HOW REDUNDANT IS THE TRANSFORMER STACK IN SPEECH REPRESENTATION MODELS?

Teresa Dorszewski, Albert Kjøller Jacobsen*, Lenka Tětková, Lars Kai Hansen*

Technical University of Denmark
DTU Compute, Section for Cognitive Systems
{tksc,akjja,lenhy,lkai}@dtu.dk

ABSTRACT

Self-supervised speech representation models, particularly those leveraging transformer architectures, have demonstrated remarkable performance across various tasks such as speech recognition, speaker identification, and emotion detection. Recent studies on transformer models revealed high redundancy between layers and the potential for significant pruning, which we will investigate here for transformer-based speech representation models. We perform a detailed analysis of layer similarity in speech representation models using three similarity metrics: cosine similarity, centered kernel alignment, and mutual nearest-neighbor alignment. Our findings reveal a block-like structure of high similarity, suggesting two main processing steps and significant redundancy of layers. We demonstrate the effectiveness of pruning transformer-based speech representation models without the need for post-training, achieving up to 40% reduction in transformer layers while maintaining over 95% of the model's predictive capacity. Furthermore, we employ a knowledge distillation method to substitute the entire transformer stack with *mimicking layers*, reducing the network size by 95-98% and the inference time by up to 94%. This substantial decrease in computational load occurs without considerable performance loss, suggesting that the transformer stack is almost completely redundant for downstream applications of speech representation models.

Index Terms— Redundancy, Layer Similarity, Transformers, Speech Representation Learning, Pruning

1. INTRODUCTION

Recent transformer-based speech representation models have shown impressive performance in numerous tasks including speech recognition, speaker identification, and emotion detection [1, 2, 3]. However, these models often come with significant computational costs due to their large size and complexity. This paper investigates the redundancy present within transformer layers of speech representation models, exploring the potential to prune or replace these layers and thereby create smaller, more efficient networks suitable for on-device automatic speech recognition tasks.

Several studies have shown that transformer-based speech representation models contain a substantial amount of redundancy [4, 5, 6, 7]. Recent research on large language models (LLMs) has revealed that many layers and neurons can be pruned without significantly impacting performance [8, 9, 10, 11]. Similar findings have been observed in speech representation models, where pruning or informed layer selection can lead to reduced computational requirements and faster inference times while retaining or even improving performance [12, 13].

Moreover, a high degree of linearity was observed in transformer models, further indicating potential redundancy [14]. It was demonstrated that the embedding transformations between sequential layers exhibit near-perfect linearity, suggesting that many of these layers may be performing redundant operations. By identifying and removing the most linear layers or replacing them with linear approximations, they show that it is possible to remove a few layers without loss in performance.

Recent studies on knowledge distillation (KD) of speech representation models have shown the potential to significantly reduce the number of parameters while mostly maintaining performance in many downstream tasks [15, 16, 17, 18]. Zampierin et al. [18] utilize a similarity-aware strategy, leveraging redundancy of layers to reduce the number of layers needed.

With this paper, we aim to investigate this redundancy systematically and leverage it in a pruning and KD approach for speech representation models. Our main contributions include:

(1) A detailed analysis of similarity in speech representation models, leveraging three similarity metrics. We find high similarity between layers that presents itself in a block-like structure, suggesting two main processing steps and high redundancy of layers.

(2) Evidence of the effectiveness of pruning transformer-based speech representation models without the need for post-training. We can remove up to 45% of the transformer layers without significant loss in performance by structurally selecting layers leveraging the block influence score [11]. We find that to maintain performance, parts of both blocks identified using similarity metrics need to be present.

(3) Significant reduction in the computational footprint of transformer-based speech representation models while maintaining 95% of the models' predictive capacity. By employing a knowledge distillation approach and replacing the whole transformer stack with *mimicking layers*, we can decrease the network size by an order of magnitudes.

Our experiments consistently show high redundancy in the transformer layers of speech representation models, indicating that the

* Equal contribution.

This work was supported by the Pioneer Centre for AI, DNRF grant number P1, the DIREC Bridge project Deep Learning and Automation of Imaging-Based Quality of Seeds and Grains, Innovation Fund Denmark grant number 9142-00001B, and the Novo Nordisk Foundation grant NNF22OC0076907 "Cognitive spaces - Next generation explainability".

transformer stack can be considerably minimized for downstream applications to create more resource-efficient networks.

2. METHODS

We divide the transformer-based network into three parts. Let $f^{(i)} : \mathbb{R}^D \to \mathbb{R}^d$ for $i \in \{1, \dots, \mathcal{L}\}$ be the network up to layer i (with embedding dimension d), let $g^{(i)} : \mathbb{R}^d \to \mathbb{R}^d$ be the remaining part of the network from layer i to $\mathcal{L}$, and $h : \mathbb{R}^d \to \mathbb{R}^{|\mathcal{C}|}$ the classification layer. The composite function $\left(h \circ \left(g^{(i)} \circ f^{(i)}\right)\right)(\boldsymbol{x})$ is then the complete forward pass for an input $\boldsymbol{x} \in \mathbb{R}^D$. For the rest of this section, for simplicity we fix the notation: $i, j \in \{1, \dots, \mathcal{L}\}$ indices of transformer layers, $X \in \mathbb{R}^{n \times D}$ a matrix of input data, and $A^{(i)} = f^{(i)}(X) \in \mathbb{R}^{n \times d}$ representation of input after layer i.

2.1. Layer Similarity

First, we perform an extensive analysis of latent representations after each transformer layer of several speech representation models (see subsection 2.4), where we investigate the similarity between layers to identify redundant information. We extract latent representations of audio input after each transformer block and compare the representations across layers, leveraging three similarity metrics, namely cosine similarity, centered kernel alignment (CKA), and mutual nearest neighbor alignment (mutual kNN). All metrics are invariant to isotropic scaling and orthogonal transformations implying permutation invariance. All scores depend on the input data X, which we omit in the notation for simplicity. Moreover, we center all the representations.

The cosine similarity score between representations of X at layers i and j is defined as

$$\mathrm{S}_{cos}(i,j) = \frac{1}{n} \sum_{l=1}^{n} \frac{\left(A_{l,\cdot}^{(j)}\right)^T A_{l,\cdot}^{(i)}}{\left\|A_{l,\cdot}^{(i)}\right\| \left\|A_{l,\cdot}^{(j)}\right\|}. \tag{1}$$

We additionally consider the CKA metric [19]. In the linear form, CKA is given by

$$\mathrm{S}_{CKA}(i,j) = \frac{\left\|\left(A^{(j)}\right)^T A^{(i)}\right\|_F^2}{\left\|\left(A^{(i)}\right)^T A^{(i)}\right\|_F \left\|\left(A^{(j)}\right)^T A^{(j)}\right\|_F}. \tag{2}$$

Acknowledging the ongoing discussion on the validity of similarity metrics for learned representations [20], we consider the locally-aware mutual kNN similarity score given by

$$\mathrm{S}_{kNN}(i,j) = \frac{1}{n} \sum_{l=1}^{n} \left(\frac{1}{k} \left|\mathcal{N}_k\left(A_{l,\cdot}^{(i)}\right) \cap \mathcal{N}_k\left(A_{l,\cdot}^{(j)}\right)\right|\right), \tag{3}$$

where $\mathcal{N}_k\left(A_{l,\cdot}^{(i)}\right)$ is the set of indices for the k-nearest samples of $A_{l,\cdot}^{(i)}$ in the batch. We chose $k = 8$ based on the robustness of initial experiments and previous studies [20].

2.2. Pruning

We investigate the relation between feature similarity patterns and model redundancy by heuristically pruning the transformer stack. The considered heuristics include *forward* and *backward* pruning, i.e. pruning starting with the first or last transformer block, along with pruning by the minimum Block Influence (BI) score $\mathrm{BI}(i) = 1 - \mathrm{S}_{cos}(i-1, i)$ [11] and a modified, locally-aware version relying on mutual kNN similarity, S_{kNN}, rather than cosine similarity, S_{cos}. For all heuristics, we prune by deleting whole transformer blocks in the order determined by the heuristic, the first transformer block is never pruned. The remaining blocks are stitched together without any post-training.

2.3. Mimicking Networks - Knowledge Distillation

We propose a simple strategy for distilling knowledge from fine-tuned audio models (teacher networks) based on reproducing intermediate representations. We introduce a so-called *mimicking network*, which is trained to replace the transformer stack using 1 or 2 *mimicking layers*. As illustrated in Figure 1, the mimicking network learns to reproduce representations in the last layer $\mathcal{L}$ of the teacher transformer stack and optionally also in the layer i. As experimental parameters, we consider the layer type, i.e. *Transformer* (`TransformerEncoderLayer` from PyTorch) or *linear mimic* layers (Figure 1 right), as well as their hidden dimensionality, $z \in \{32, 768, 4096\}$. We ensure weight-sharing along the temporal dimension. Each model is trained with a 2-stage procedure consisting of *1) a mimicking phase* using a mean squared error (MSE) objective and *2) an adaptation phase* for fine-tuning to the downstream task using a negative log likelihood (NLL) objective on the log-probabilities (detailed loss function in Figure 1). For evaluating the mimicking approach, we additionally explore randomly initialized models that only consider the adaptation phase.

In the *mimicking phase*, models are trained for 50 epochs using the training set with a batch size of 128, resulting in 33150 steps. Model evaluation is carried out regularly on 1024 inputs randomly sampled from the validation set. All models converged within the training horizon. Subsequently, the *adaption phase* exploits the optimal weight set and trains for 30 epochs, i.e. 19890 steps. The optimal models based on the validation set, i.e. before potential overfitting, are saved. Based on initial experiments, all models are trained with a fixed learning rate of 10^{-3}.

All models are evaluated on the test set by individually predicting the 4482 samples, and the inference time is measured after a GPU warm-up of 300 steps (to ensure comparable conditions). We report average performances and inference times including uncertainty estimates given by the standard error of the mean.

2.4. Data & Models

All analyses consider a word classification task using the *speech commands v0.02* dataset [21], which features 35 words (i.e. classes) spoken by more than 400 speakers. The dataset and its splits are obtained from *huggingface.co*. All audio files are resampled to 16kHz and padded/restricted to 1 second, and the *_silence_* class is excluded for the analyses.

We consider the `base` and `large` fine-tuned versions of `wav2vec2` [3] and `wavLM` [2]. All models follow the same transformer architecture [22] with 12 or 24 transformer stacks and were fine-tuned on the train set to perform classification of the 35 words, with a learning rate of $2\mathbf{x}10^{-5}$ for 10000 steps. The final accuracy on the test set of the models is 98.32 / 97.21% for `wav2vec2` (base/large) and 97.22 / 98.86% for `wavLM` (base/large).

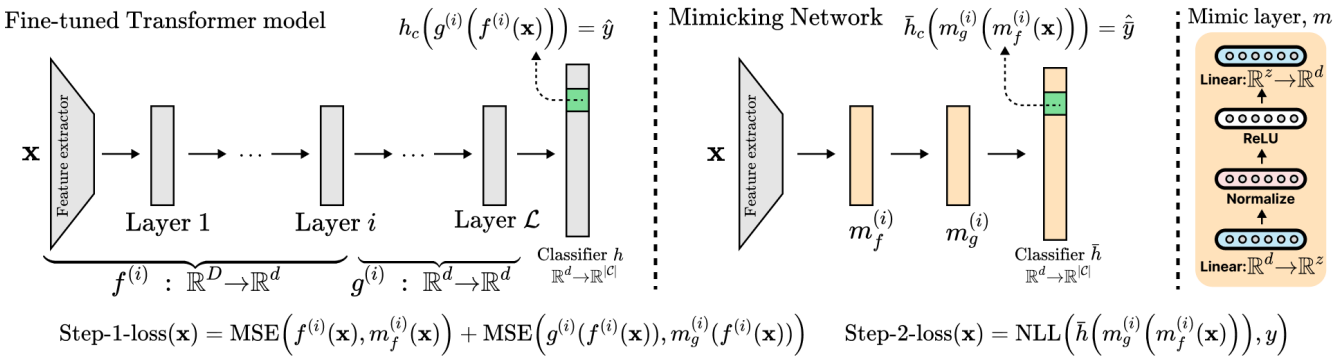

Fig. 1. **Conceptual overview of mimicking networks.** ***Left:*** The general transformer model, consisting of a feature extractor module, the transformer stack and a classification/probing layer. We denote the first part of the network until layer i by $f^{(i)}$ the remaining part of the network from layer i to $\mathcal{L}$ by $g^{(i)}$, and the classification layer by h. ***Middle:*** A 2-layer mimicking network where $m_f^{(i)}$ and $m_g^{(i)}$ represent mimicked representations of $f^{(i)}$ and $g^{(i)}$, respectively, learned via the step-1-loss. In step 2, the classifier, $\bar{h}$, and mimicking layers are then finetuned for the downstream task. ***Right:*** Design of the mimicking layer that maps an input embedding of dimension d to a z-dimensional representation before mapping it back to the original shape.

3. RESULTS & DISCUSSION

3.1. Layer Similarity

First, we explore the similarity structure between layers. Our analysis reveals that all models exhibit two primary blocks characterized by highly similar latent representations throughout each block (see Figure 2a). The first block consists of approximately two-thirds to three-quarters of the transformer layers, while the second block typically comprises the final four to five layers. These findings suggest a significant degree of redundancy within these blocks, raising questions about the necessity of all layers. Similar block structures have been observed (but not further investigated) in convolutional models [19] and other speech representation models [18], with varying numbers and dimensions of the blocks depending on the models and tasks.

When comparing the three similarity metrics, CKA and mutual kNN exhibit the block structure more distinctly than cosine similarity. Consistent with recent debates on similarity metrics [19, 20], our findings indicate that CKA and mutual kNN more accurately capture similarity structures, and mutual kNN reveals additional details within the blocks. This further indicates the potential benefit of pruning according to local rather than global similarity structure.

3.2. Layer-Wise Pruning

Given the high similarity and therefore potential redundancy of the transformer layers, we investigate how many layers can be pruned, i.e. simply deleted without retraining, before significantly impacting performance. For all models, we can prune a substantial number of layers before we see a significant drop in performance (see Figure 2b). When using BI or kNN-BI to prune the least important layers first, we can prune 25-42% of layers while maintaining 95% of the original performance. Interestingly, when pruning forward (not pruning the very first layer), we see very similar results, where we can prune almost up to the same amount of layers, which indicates that especially early layers within the first block are redundant.

When pruning backward or forward, the performance drops completely after removing either of the blocks identified earlier (with exception of wavLM large, where the performance already drops after removing half of the first block). In Figure 2c it becomes apparent that the performance is maintained only as long as parts of both blocks are still present, highlighting the importance of both processing steps.

Other studies have shown that by retraining after layer-wise pruning, models can regain full performance or, in some cases, even improve performance in speech representations models [13] and LLMs [23]. This highlights how redundant layers are not only a computational burden, but are also unnecessary or even harmful to keep.

3.3. Mimicking Networks

Is the transformer stack completely redundant? To answer this question, we substitute the transformer stack with mimicking layers. Based on the two blocks we find during the similarity analysis, we first test two mimicking layers, with the intermediate mimicked layer i being the last layer of the first block. We compare these results to only using one mimicking layer, directly learning the final representations, as well as to using just a linear or transformer layer instead, directly trained for classification. These substitutions lead to a parameter reduction of up to 95% in base models and up to 98% in large models and a reduction of the inference time of up to 94% (seeFigure 3 and Figure 4-6 in the appendix). Most models retain more than 95% of the performance while achieving these high reductions in size and therefore also inference time. This is a higher reduction, while maintaining comparable performance, than in recent studies on KD of speech representation models [15, 16, 17, 18] and a much higher reduction in size than in pruning studies on speech representation models [5, 12, 13]. Although our method does not bring significant improvements over other state-of-the-art distillation methods, it demonstrates the high redundancy of the transformer stack and highlights the potential for significant reductions in size and computational load.

Interestingly, we do not observe a significant difference in performance between using one or two mimicking layers, indicating that intermediate representations are not essential for the downstream task. We found that increasing the hidden dimensionality of the mimicking layer (z) slightly improved performance, although $z = 4096$ often led to model overfitting. While transformer mimicker layers generally produced the best results, using

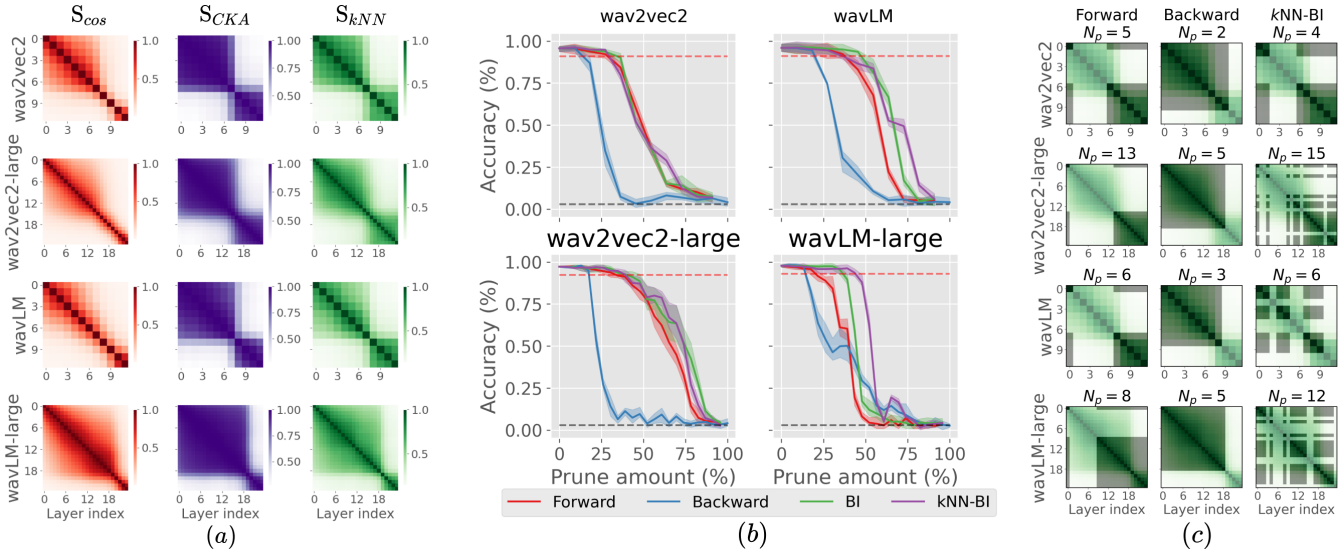

Fig. 2. **Analysis of redundancy of layers** using similarity measures and pruning of layers: (a) Similarity between layers of `wav2vec2` and `wavLM`. All three metrics (cosine similarity, CKA, mutual kNN) reveal a block structure. (b) Effect of pruning on performance, using four different pruning objectives. Up to 45% of layers can be pruned while maintaining 95% of accuracy (- -). After pruning most layers, the model performance drops to random chance (- -). Uncertainty ranges cover the empirical 2.5 and 97.5 quantiles obtained from $N = 5$ runs. (c) Visualisation of pruned layers on top of the kNN similarity matrix (with 50% performance threshold). Light layers indicate pruned layers, dark layers are still present. Backward and forward pruning (left+middle) only preserve performance as long as both blocks are still present. Pruning based on kNN-BI (right) prunes layers mainly in the first block.

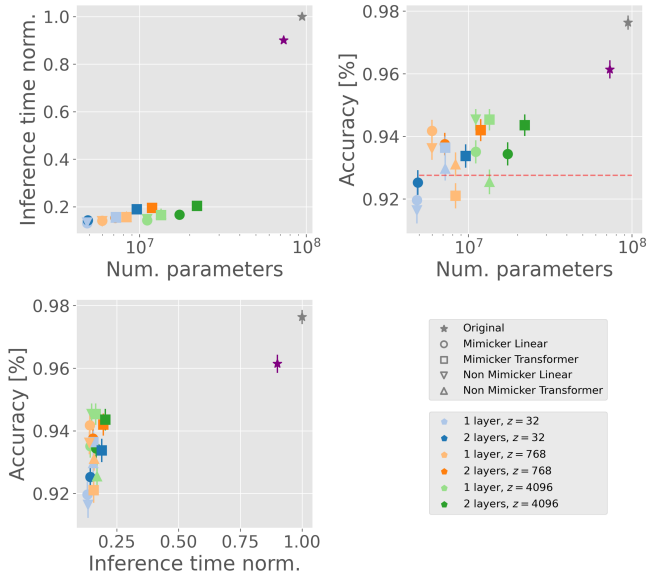

Fig. 3. **Simplification of transformer stack using mimicking networks.** Reduction in inference time (up to 87%) and number of parameters (up to 95%) using mimicking networks, while maintaining 95% of the original accuracy (- -). We test transformer and linear layers with different dimensions z. Inference time is normalized to 1 (inference time of original model), the pruned model has 3 layers pruned using kNN-BI. These results are for `wav2vec2`, results for other models are in the appendix.

a single transformer or linear layer without mimicking steps also demonstrated impressive performance, suggesting the exact representations learned by the transformer stack to be non-critical. However, eliminating the transformer stack and relying solely on the final linear layer for classification resulted in a substantial performance drop to 79% accuracy, implying some degree of non-linearity is required to maintain performance. These findings suggest that the transformer stack is not needed for downstream applications and can be replaced by a single non-linear layer. The redundancy of the transformer stack has also been found by Kostas et al. [24] in an EEG transformer model, where the transformer stack was beneficial for pretraining but not critical or even harmful for downstream applications.

4. CONCLUSION

Our findings indicate a significant degree of redundancy within the transformer layers of speech representation models. This redundancy is evident from the high similarity between layers, particularly within the two primary blocks identified in our analysis. The ability to prune 15-45% of transformer layers without retraining while mostly maintaining performance further underscores the significant redundancy present in transformer layers within speech representation models. The two main blocks found in the similarity analysis seem to be critical for performance, as fully pruning either block results in a massive drop in performance. However, within these blocks many layers can be pruned, suggesting high redundancy within each block.

Our exploration of mimicking networks suggests that the entire transformer stack can be replaced with a much smaller and faster network, while maintaining over 95% of performance, highlighting the high redundancy of the transformer stack in speech representa-

tion models. This potential of reducing the model to a model order of magnitudes smaller and faster than the original model is also supported by recent studies on KD in speech representation models [15, 16, 17, 18]. All of these findings highlight the potential for substantial reduction in size and computational load enabling practical applications in on-device automatic speech recognition and resource-constrained environments.

5. REFERENCES

[1] Shu-Wen Yang et al., "SUPERB: speech processing universal performance benchmark," in *Interspeech 2021, 22nd Annual Conference of the International Speech Communication Association*. 2021, pp. 1194–1198, ISCA.

[2] Sanyuan Chen et al., "Wavlm: Large-scale self-supervised pre-training for full stack speech processing," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 6, pp. 1505–1518, 2022.

[3] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," *Advances in neural information processing systems*, vol. 33, pp. 12449–12460, 2020.

[4] Andy T Liu, Shang-Wen Li, and Hung-yi Lee, "Tera: Self-supervised learning of transformer encoder representation for speech," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 2351–2366, 2021.

[5] Yifan Peng, Kwangyoun Kim, Felix Wu, Prashant Sridhar, and Shinji Watanabe, "Structured pruning of self-supervised pre-trained models for speech recognition and understanding," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023, pp. 1–5.

[6] Shucong Zhang, Erfan Loweimi, Peter Bell, and Steve Renals, "On the usefulness of self-attention for automatic speech recognition with transformers," in *2021 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2021, pp. 89–96.

[7] Hassan Sajjad, Fahim Dalvi, Nadir Durrani, and Preslav Nakov, "On the effect of dropping layers of pre-trained transformer models," *Computer Speech & Language*, vol. 77, pp. 101429, 2023.

[8] Fahim Dalvi, Hassan Sajjad, Nadir Durrani, and Yonatan Belinkov, "Analyzing redundancy in pretrained transformer models," *arXiv preprint arXiv:2004.04010*, 2020.

[9] Yifei Yang, Zouying Cao, and Hai Zhao, "Laco: Large language model pruning via layer collapse," *arXiv preprint arXiv:2402.11187*, 2024.

[10] Andrey Gromov, Kushal Tirumala, Hassan Shapourian, Paolo Glorioso, and Daniel A Roberts, "The unreasonable ineffectiveness of the deeper layers," *arXiv preprint arXiv:2403.17887*, 2024.

[11] Xin Men, Mingyu Xu, Qingyu Zhang, Bingning Wang, Hongyu Lin, Yaojie Lu, Xianpei Han, and Weipeng Chen, "Shortgpt: Layers in large language models are more redundant than you expect," *arXiv preprint arXiv:2403.03853*, 2024.

[12] Ankita Pasad, Bowen Shi, and Karen Livescu, "Comparative layer-wise analysis of self-supervised speech models," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023, pp. 1–5.

[13] Teresa Dorszewski, Lenka Tětková, and Lars Kai Hansen, "Convexity-based pruning of speech representation models," *2024 IEEE 34th International Workshop on Machine Learning for Signal Processing (MLSP)*, 2024.

[14] Anton Razzhigaev, Matvey Mikhalchuk, Elizaveta Goncharova, Nikolai Gerasimenko, Ivan Oseledets, Denis Dimitrov, and Andrey Kuznetsov, "Your transformer is secretly linear," *arXiv preprint arXiv:2405.12250*, 2024.

[15] Xiaoyu Yang, Qiujia Li, and Philip C Woodland, "Knowledge distillation for neural transducers from large self-supervised pre-trained models," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 8527–8531.

[16] Yifan Peng, Yui Sudo, Shakeel Muhammad, and Shinji Watanabe, "Dphubert: Joint distillation and pruning of self-supervised speech models," *arXiv preprint arXiv:2305.17651*, 2023.

[17] Kuan-Po Huang, Tzu-Hsun Feng, Yu-Kuan Fu, Tsu-Yuan Hsu, Po-Chieh Yen, Wei-Cheng Tseng, Kai-Wei Chang, and Hung-Yi Lee, "Ensemble knowledge distillation of self-supervised speech models," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023, pp. 1–5.

[18] Luca Zampierin, Ghouthi Boukli Hacene, Bac Nguyen, and Mirco Ravanelli, "Skill: Similarity-aware knowledge distillation for speech self-supervised learning," *arXiv preprint arXiv:2402.16830*, 2024.

[19] Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton, "Similarity of neural network representations revisited," in *International conference on machine learning*. PMLR, 2019, pp. 3519–3529.

[20] Minyoung Huh, Brian Cheung, Tongzhou Wang, and Phillip Isola, "The platonic representation hypothesis," *arXiv preprint arXiv:2405.07987*, 2024.

[21] P. Warden, "Speech Commands: A Dataset for Limited-Vocabulary Speech Recognition," *ArXiv e-prints*, Apr. 2018.

[22] A Vaswani, "Attention is all you need," *Advances in Neural Information Processing Systems*, 2017.

[23] Chun Fan, Jiwei Li, Xiang Ao, Fei Wu, Yuxian Meng, and Xiaofei Sun, "Layer-wise model pruning based on mutual information," *arXiv preprint arXiv:2108.12594*, 2021.

[24] Demetres Kostas, Stephane Aroca-Ouellette, and Frank Rudzicz, "Bendr: Using transformers and a contrastive self-supervised learning task to learn from massive amounts of eeg data," *Frontiers in Human Neuroscience*, vol. 15, pp. 653659, 2021.

A. APPENDIX / SUPPLEMENTAL MATERIAL

We present the data behind Figure 3 (see Table 1) along with experiments of mimicking networks for the `wav2vec2-large`, `wavLM-small` and `wavLM-large`. Note that the L and T respectively denote linear and transformer layers, while z is the hidden dimension of the layer(s).

Results of simplification of networks using *mimicking networks*. In `wav2vec2-large` (Figure 4 and Table 2) and `wavLM-small` (Figure 5 and Table 3) the models keep over 95% of their original performance while reducing the number of parameters by 95-98% and the inference time by up to 91%. In `wavLM-large` (Figure 6 and Table 4) the performance is still above 90% of the original performance while reducing the size by 98% and the inference time by 94%.

Network type	Layer type	N layers	z	Number of parameters	Inference time (normalized)	Accuracy
Original	T	12	-	94577571	1	0.976 ± 0.002
Mimicker	L	1	32	4851331	0.13	0.919 ± 0.004
Mimicker	L	2	32	4901283	0.14	0.925 ± 0.004
Mimicker	L	1	768	5984035	0.14	0.942 ± 0.003
Mimicker	L	2	768	7165219	0.14	0.938 ± 0.004
Mimicker	L	1	4096	11105827	0.15	0.935 ± 0.004
Mimicker	L	2	4096	17402147	0.16	0.934 ± 0.004
Mimicker	T	1	32	7216707	0.14	0.936 ± 0.004
Mimicker	T	2	32	9632099	0.16	0.934 ± 0.004
Mimicker	T	1	768	8347939	0.15	0.921 ± 0.004
Mimicker	T	2	768	11894563	0.16	0.942 ± 0.003
Mimicker	T	1	4096	13463075	0.16	0.945 ± 0.003
Mimicker	T	2	4096	22124835	0.18	0.944 ± 0.003
Non-mimicker	L	1	32	4851331	0.13	0.916 ± 0.004
Non-mimicker	L	1	768	5984035	0.14	0.936 ± 0.004
Non-mimicker	L	1	4096	11105827	0.14	$\mathbf{0.945 \pm 0.003}$
Non-mimicker	T	1	32	7216707	0.14	0.93 ± 0.004
Non-mimicker	T	1	768	8347939	0.15	0.931 ± 0.004
Non-mimicker	T	1	4096	13463075	0.16	0.925 ± 0.004

Table 1. `wav2vec2-small`

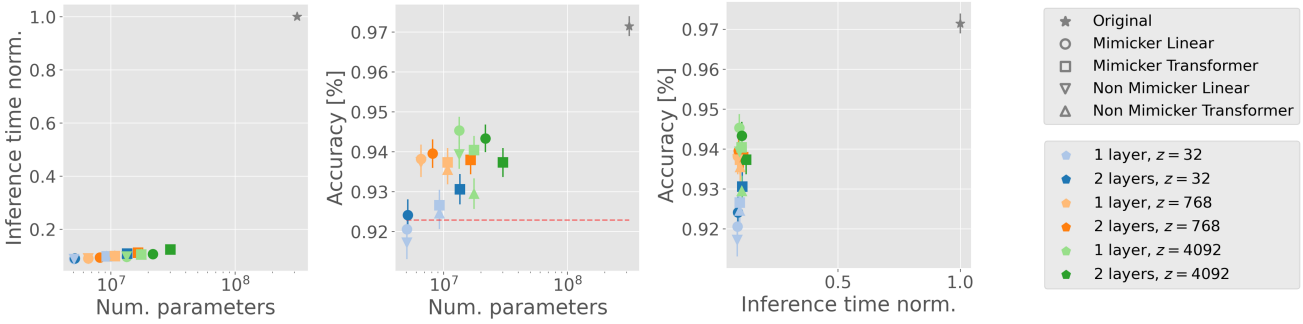

Fig. 4. `wav2vec2-large`

Network type	Layer type	N layers	z	Number of parameters	Inference time (normalized)	Accuracy
Original	T	24	-	315700387	1	0.971 ± 0.002
Mimicker	L	1	32	5064835	0.089	0.921 ± 0.004
Mimicker	L	2	32	5131427	0.091	0.924 ± 0.004
Mimicker	L	1	768	6574371	0.09	0.938 ± 0.004
Mimicker	L	2	768	8149027	0.094	0.94 ± 0.004
Mimicker	L	1	4096	13400099	0.097	$\mathbf{0.945 \pm 0.003}$
Mimicker	L	2	4096	21793827	0.11	0.943 ± 0.003
Mimicker	T	1	32	9267267	0.098	0.927 ± 0.004
Mimicker	T	2	32	13536355	0.11	0.931 ± 0.004
Mimicker	T	1	768	10775331	0.099	0.937 ± 0.004
Mimicker	T	2	768	16552483	0.11	0.938 ± 0.004
Mimicker	T	1	4096	17594403	0.11	0.94 ± 0.004
Mimicker	T	2	4096	30190627	0.12	0.937 ± 0.004
Non-mimicker	L	1	32	5064835	0.088	0.917 ± 0.004
Non-mimicker	L	1	768	6574371	0.09	0.937 ± 0.004
Non-mimicker	L	1	4096	13400099	0.097	0.939 ± 0.004
Non-mimicker	T	1	32	9267267	0.098	0.925 ± 0.004
Non-mimicker	T	1	768	10775331	0.1	0.936 ± 0.004
Non-mimicker	T	1	4096	17594403	0.11	0.929 ± 0.004

Table 2. `wav2vec2-large`

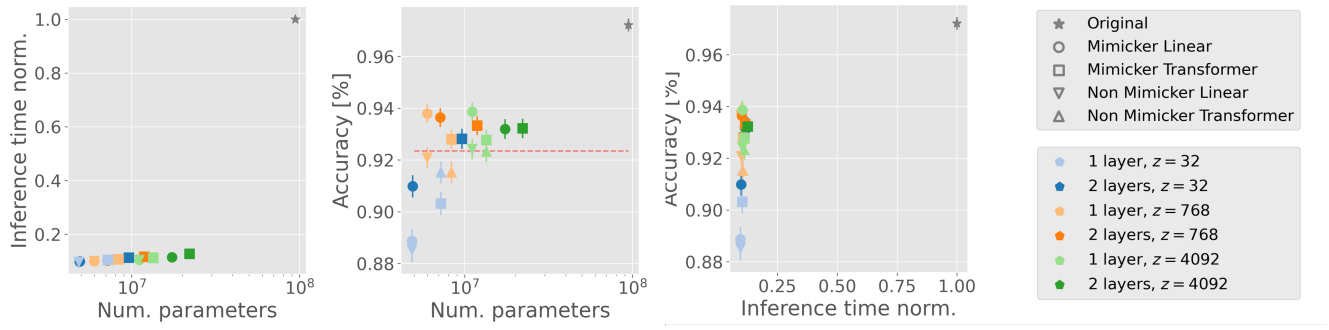

Fig. 5. `wavLM-small`

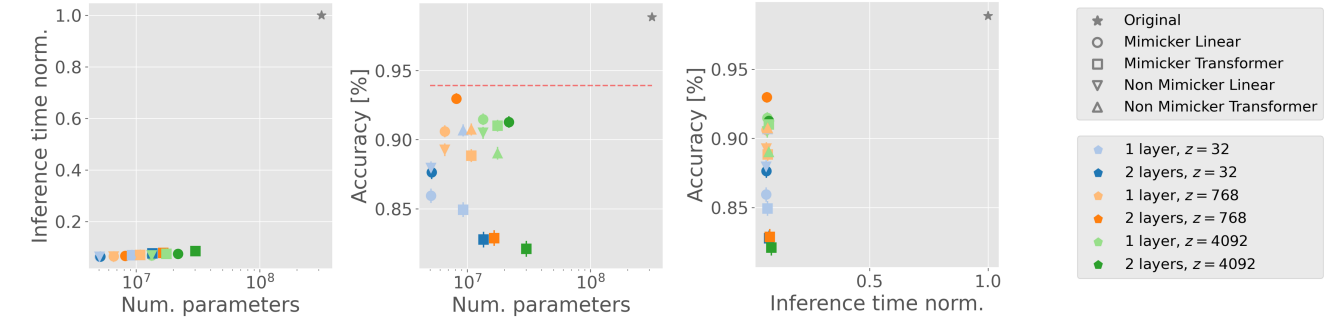

Fig. 6. `wavLM-large`

Network type	Layer type	N layers	z	Number of parameters	Inference time (normalized)	Accuracy
Original	T	12	-	94587795	1	0.972 ± 0.002
Mimicker	L	1	32	4851331	0.097	0.889 ± 0.005
Mimicker	L	2	32	4901283	0.099	0.91 ± 0.004
Mimicker	L	1	768	5984035	0.1	0.938 ± 0.004
Mimicker	L	2	768	7165219	0.1	0.936 ± 0.004
Mimicker	L	1	4096	11105827	0.1	$\mathbf{0.939 \pm 0.004}$
Mimicker	L	2	4096	17402147	0.11	0.932 ± 0.004
Mimicker	T	1	32	7216707	0.1	0.903 ± 0.004
Mimicker	T	2	32	9632099	0.11	0.928 ± 0.004
Mimicker	T	1	768	8347939	0.11	0.928 ± 0.004
Mimicker	T	2	768	11894563	0.12	0.933 ± 0.004
Mimicker	T	1	4096	13463075	0.11	0.928 ± 0.004
Mimicker	T	2	4096	22124835	0.13	0.932 ± 0.004
Non-mimicker	L	1	32	4851331	0.097	0.885 ± 0.005
Non-mimicker	L	1	768	5984035	0.1	0.921 ± 0.004
Non-mimicker	L	1	4096	11105827	0.1	0.924 ± 0.004
Non-mimicker	T	1	32	7216707	0.1	0.915 ± 0.004
Non-mimicker	T	1	768	8347939	0.11	0.915 ± 0.004
Non-mimicker	T	1	4096	13463075	0.11	0.923 ± 0.004

Table 3. `wavLM-small`

Network type	Layer type	N layers	z	Number of parameters	Inference time (normalized)	Accuracy
Original	T	24	-	315724515	1	0.989 ± 0.002
Mimicker	L	1	32	5070979	0.064	0.859 ± 0.005
Mimicker	L	2	32	5137571	0.065	0.876 ± 0.005
Mimicker	L	1	768	6580515	0.065	0.906 ± 0.004
Mimicker	L	2	768	8155171	0.067	$\mathbf{0.93 \pm 0.004}$
Mimicker	L	1	4096	13406243	0.069	0.915 ± 0.004
Mimicker	L	2	4096	21799971	0.075	0.913 ± 0.004
Mimicker	T	1	32	9273411	0.07	0.849 ± 0.005
Mimicker	T	2	32	13542499	0.076	0.828 ± 0.006
Mimicker	T	1	768	10781475	0.07	0.888 ± 0.005
Mimicker	T	2	768	16558627	0.079	0.829 ± 0.006
Mimicker	T	1	4096	17600547	0.075	0.91 ± 0.004
Mimicker	T	2	4096	30196771	0.086	0.821 ± 0.006
Non-mimicker	L	1	32	5070979	0.064	0.88 ± 0.005
Non-mimicker	L	1	768	6580515	0.064	0.893 ± 0.005
Non-mimicker	L	1	4096	13406243	0.069	0.905 ± 0.004
Non-mimicker	T	1	32	9273411	0.07	0.907 ± 0.004
Non-mimicker	T	1	768	10781475	0.071	0.907 ± 0.004
Non-mimicker	T	1	4096	17600547	0.075	0.89 ± 0.005

Table 4. `wavLM-large`