How Redundant Is the Transformer Stack in Speech Representation Models?

Albert Kjøller Jacobsen* akjja@dtu.dk
Teresa Dorszewski* tksc@dtu.dk
Lenka Tětková lenhy@dtu.dk
Lars Kai Hansen lkai@dtu.dk

Section for Cognitive Systems, DTU Compute
Technical University of Denmark
2800 Kongens Lyngby, Denmark

Abstract

Self-supervised speech representation models, particularly those leveraging transformer architectures, have demonstrated remarkable performance on downstream tasks. Recent studies revealed high redundancy of transformer layers, potentially allowing for smaller models and more efficient inference. We perform a detailed analysis of layer similarity in speech models, leveraging three similarity metrics. Our findings reveal a block-like structure of high similarity, suggesting significant redundancy within the blocks along with two main processing steps that are both found to be critical for maintaining performance. We demonstrate the effectiveness of pruning transformer-based speech models without post-training, achieving up to 40% reduction in transformer layers while maintaining 95% of the model's predictive capacity. Lastly, we show that replacing the transformer stack with a few simple layers can reduce the network size by up to 95% and inference time by up to 87%, significantly reducing the computational footprint with minimal performance loss, revealing the benefits of model simplification for downstream applications.

1 Introduction

Recent advancements in speech representation models, particularly those leveraging transformer architectures, have demonstrated remarkable performance across various downstream tasks (1; 2; 3), however, inference with these models often comes with significant computational costs due to large model sizes. This paper investigates the redundancy present within transformer layers of speech representation models, exploring the potential of pruning, thereby using smaller and more efficient networks for inference. Several studies have shown that transformer models contain a substantial amount of redundancy (4; 5; 6; 7; 8) and recent research on large language models (LLMs) has revealed that many layers can be pruned without significantly impacting performance (9; 10; 11). This phenomenon is not limited to LLMs; 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, high linearity was observed in transformer models, further indicating potential redundancy (14). They demonstrated that the embedding transformations between sequential layers exhibit near-perfect linearity, suggesting that many layers may perform redundant operations. They retain model performance while removing the most linear layers or replacing them with linear approximations. This paper systematically investigates redundancy in speech models. Our main contributions include:

1. A detailed analysis of similarity in speech representation models, leveraging three similarity metrics. We find a block-like similarity structure suggesting two main processing steps.

*Equal contribution

4th NeurIPS Efficient Natural Language and Speech Processing Workshop (ENLSP-IV 2024).

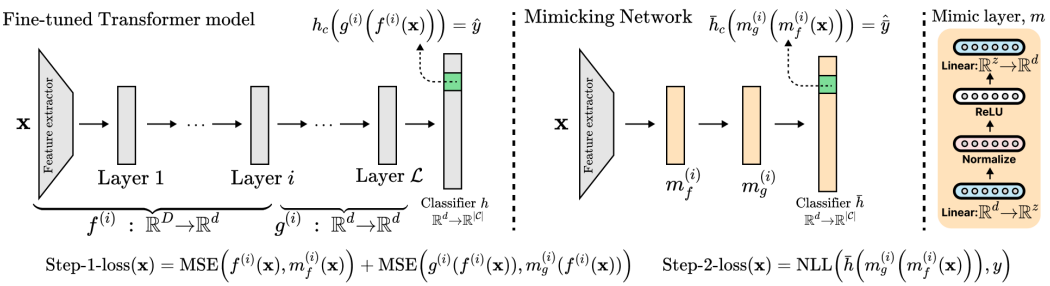

Figure 1: **Overview of our analysis framework.** *Left*: We let the fine-tuned transformer to layer i be $f^{(i)}$, from layer i to $\mathcal{L}$ be $g^{(i)}$, and the classifying layer be h. *Middle:* 2-layer mimicking network where $m_f^{(i)}$ and $m_g^{(i)}$ mimic $f^{(i)}$ and $g^{(i)}$, learned via the MSE loss. We fine-tune the m's and $\overline{h}$ with NLL loss. *Right:* A mimic layer maps a d-dimensional representation to z dimensions and back.

2. Evidence showing that up to 45% of transformer-based speech model layers can be structurally pruned without post-training, with low performance drop. We find that to maintain performance, parts of both blocks identified in the similarity analysis need to be present.
3. Significant reduction in the computational footprint of transformer-based speech models while keeping 95% predictive capacity, by replacing the transformer stack with a few layers.

2 Methods

2.1 Layer similarity

We perform an extensive analysis of latent representations of speech representation models. We extract and compare representations of audio input after each transformer block to identify redundancy. All scores depend on the input batch $X \in \mathbb{R}^{n \times D}$, which we omit in the notation for simplicity, thus $A = f^{(i)}(X) \in \mathbb{R}^{n \times D}$ and $B = f^{(j)}(X) \in \mathbb{R}^{n \times D}$ are the representations of the batch at layers $i, j \in \{1, \ldots, \mathcal{L}\}$. We center the the representations batch-wise and use cosine similarity, $\mathrm{S}_{cos}(i,j) = \frac{1}{n} \sum_{l=1}^{n} A_{l,\cdot}^T B_{l,\cdot} / \left(||A_{l,\cdot}|| \cdot ||B_{l,\cdot}||\right)$, along with two other metrics detailed below.

Centered Kernel Alignment (CKA) (15) holds desirable properties for neural networks, namely invariance to isotropic scaling and orthogonal transformations which implies permutation invariance. In the linear form, CKA between representations of X at layers i and j uses the Frobenius norm

$$\mathrm{S}_{CKA}(i,j) = ||B^T A||_F^2 / \left(||A^T A||_F ||B^T B||_F\right) \tag{1}$$

Mutual nearest-neighbor alignment (mutual kNN) (16) captures local structure between representations. If we let $\mathcal{N}_k(A_{l,\cdot})$ be the set of indices for the k-nearest samples of $A_{l,\cdot}$ in the batch, then mutual kNN similarity of layers i and j is defined

$$\mathrm{S}_{kNN}(i,j) = \frac{1}{n} \sum_{l=1}^{n} \left(\frac{1}{k} \left| \mathcal{N}_k(A_{l,\cdot}) \cap \mathcal{N}_k(B_{l,\cdot}) \right| \right) \tag{2}$$

2.2 Pruning

We investigate the relation between feature similarity patterns and model redundancy by heuristically pruning the transformer stack. The heuristics considered are *forward* and *backward* which removes layers sequentially from the beginning or end of the transformer stack, respectively. Additionally we prune by the minimum block-influence score (11) - i.e. $\mathrm{BI}(i) = 1 - \mathrm{S}_{cos}(i-1, i)$ - and a version based on mutual kNN similarity. We remove transformer layers in an order determined by the heuristic until the stack is empty, while always keeping the first layer. *No* post-training is done.

2.3 Mimicking networks - knowledge distillation

We propose a simple strategy (Figure 1) for distilling knowledge from fine-tuned audio models based on reproducing latent representations with a single 1- or 2-layer *mimicking network*. It learns

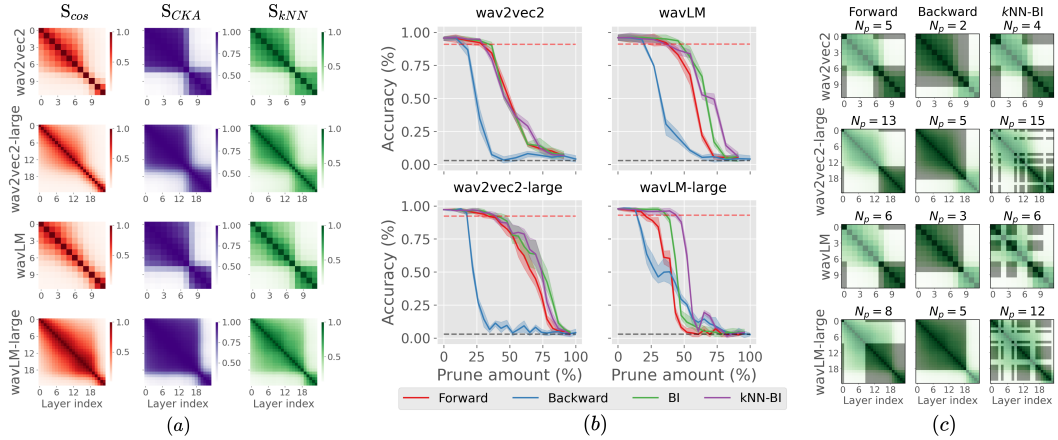

Figure 2: **Analysis of redundancy of layers**: (a) Similarity between layers of `wav2vec2` and `wavLM` averaged across batches. All three metrics reveal a block structure. (b) Effect of pruning on performance for different heuristics. Up to 45% of layers can be pruned while maintaining 95% of accuracy (- -). Performance drops to random chance (- -) with too much pruning. The [2.5, 97.5] empirical quantiles from $N = 5$ runs are provided. (c) kNN similarity matrix overlayed with a pruning mask. White layers indicate pruned layers before the performance hits 50% of the full capacity. For- and backward pruning only preserve performance while both blocks are present. kNN-BI pruning mainly considers the first block.

to reproduce representations of a speech model's last layer and optionally also an intermediate layer. We experiment with *transformer* and *mimicking* layers and their hidden dimensionality, $z \in \{32, 768, 4096\}$, while ensuring weight-sharing along the temporal axis. Each model follows a 2-stage training procedure of *1) a mimicking phase* using a *mean-squared error* (MSE) objective and *2) an adaption phase* for fine-tuning to the downstream task through *negative log likelihood* (NLL) on the log-probabilities. Additionally, we examine *non-mimicking* networks that only learn representations via the adaption phase, i.e. they are randomly initialized and fine-tuned.

In the *mimicking phase*, models are trained on a GPU using a batch size of 128 for 50 epochs, i.e. 33,150 steps and Adam. We regularly evaluate the model on 1024 random validation set samples. All models converged. Next, the *adaption phase* trains for further 30 epochs, i.e. 19,890 steps, and stores the best weights before potential overfitting. We fix the learning rate to 10^{-3} (determined from pilot experiments). We evaluate models by individually predicting the 4482 test set samples after running 300 forward passes for warming up the GPU to ensure running time comparability between models. We report average accuracies and inference times along with the standard errors.

2.4 Data & models

All analyses consider a word classification task from the *speech commands v0.02* dataset (17) (data splits available at *huggingface.co*) which features 35 words spoken by >400 speakers. We resample audio inputs to 16kHz and pad/restrict to 1 second and exclude the *_silence_* class for analyses. We consider the `small` and `large` fine-tuned versions of `wav2vec2` (3) and `wavLM` (2), respectively having 12 and 24 layers in the transformer stack. These were fine-tuned to classify the 35 words, with a learning rate of $2\text{x}10^{-5}$ for 10,000 steps, resulting in accuracies of 98.32 / 97.21% for `wav2vec2` (base/large) and 97.22 / 98.86% for `wavLM` (base/large).

3 Results & discussion

3.1 Layer similarity

Our analysis reveals that all models exhibit two primary blocks characterized by highly similar latent representations (see Figure 2a). The second block typically comprises the final 4-5 layers. The high similarities suggest a significant degree of redundancy, raising questions on the necessity of keeping all layers. Comparing the three similarity metrics, CKA and mutual kNN show the block structure

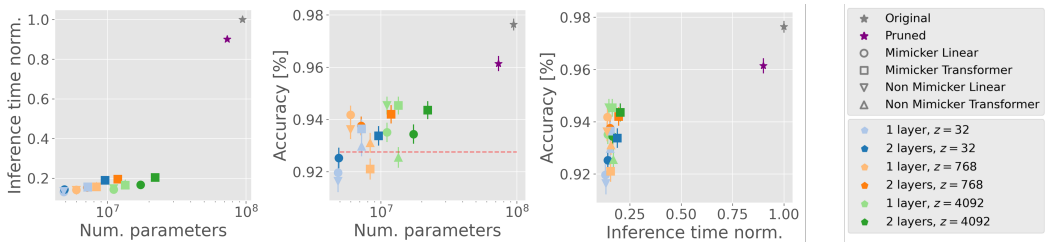

Figure 3: **Simplification of `wav2vec2-small`'s transformer stack using mimicking networks.** Reduction in inference time (up to 87%) and number of parameters (up to 95%) using mimicking networks, while maintaining 95% of the original accuracy (- -). The pruned model removes 3 layers according to kNN-BI. Inference time is normalized wrt. `wav2vec2-small` and the results for other models look very similar. The associated data is provided in Appendix A along with similar illustrations for other models.

more clearly than cosine similarity. Consistent with recent debates (15; 16), our findings indicate that CKA and mutual kNN better capture similarity structures. Mutual kNN reveals additional blocks details, suggesting the potential benefit of pruning based on local rather than global similarity.

3.2 Layer-wise pruning

Given the high similarity , we investigate how many layers can be pruned without impacting performance. For all models, we can prune a substantial amount 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 from the second layer and forward, we observe that we can prune almost the same amount of layers, indicating redundancy of the early layers of the first similarity block. When pruning backward the performance drops after only 1-4 pruned layers, depending on the model size. In Figure 2c, it becomes apparent that performance is maintained only while parts of both blocks are present, highlighting the importance of both processing steps for classification. However, a recent study exploring the same fine-tuned models found that *with* post-training, backward pruning can be successful without any loss in performance (13).

3.3 Mimicking networks

Based on the two blocks identified in the similarity analysis, we let the intermediate mimicking layer, $m^{(i)}$, mimic the last layer of the first block. We compare with a 1-layer mimicking network, directly mimicking $f^{(\mathcal{L})}$ as well as with versions that only learn via the adaption phase. Substitution of the transformer stack leads to reductions in parameters and inference times of 76.6-94.8% and 79.6-86.8%, respectively, while most models retain over 95% of the performance. We found that increasing the hidden dimensionality, z, slightly improved performance, with transformer mimic layers generally performing best. Interestingly, no significant performance difference was observed for 1- or 2-layer networks, suggesting intermediate representations to be non-essential for the downstream task. 1-layer networks without the mimicking phase demonstrated impressive performance, suggesting the transformer stack's exact representations to be non-critical, yet, removing it completely and using only the fine-tuned classification layer dropped accuracy to 79%, indicating the need for some non-linearity. These findings suggest that the transformer stack can be simplified to a single non-linear layer for the downstream application.

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 similarity blocks identified in our analysis. The ability to prune 15-45% of the transformer layers without significant loss in performance further underscores the redundancy present in transformer layers within speech representation models. We reveal a relation between block-like similarity patterns and predictive performance; the two main blocks found in the similarity analysis

seem to be critical for the task, as fully pruning either block results in a massive drop in performance. However, many layers can be pruned within these blocks, suggesting high redundancy within each block. Our exploration of mimicking networks - supported by recent studies on knowledge distillation in speech representation models (18; 19; 20; 21) - suggests that the entire transformer stack can be replaced with a much smaller and faster network for efficient inference while maintaining over 95% of performance. This highlights the potential for leveraging large, complex models for on-device applications in resource-constrained environments which might otherwise be infeasible.

Limitations & future work. Our findings from spoken word classification might vary across tasks / domains, yet the analyses broadly apply to investigating efficient inference for transformer models. We suggest extending to vision transformers or CLIP, as initial studies revealed block-like similarity.

Acknowledgments and Disclosure of Funding

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".

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] 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.

[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] 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.

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

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

[18] 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.

[19] 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.

[20] 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.

[21] 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.

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`

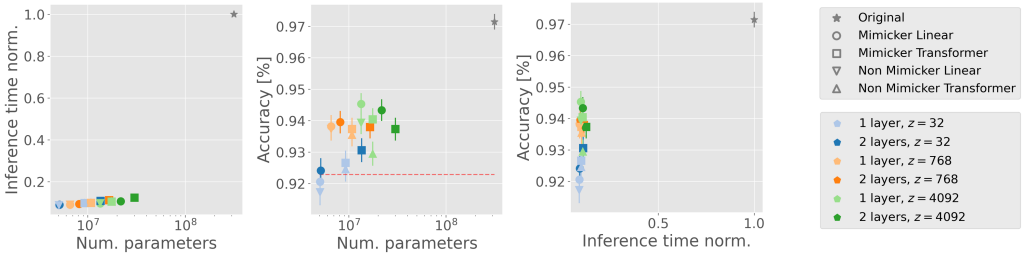

Figure 4: `wav2vec2-large`

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`

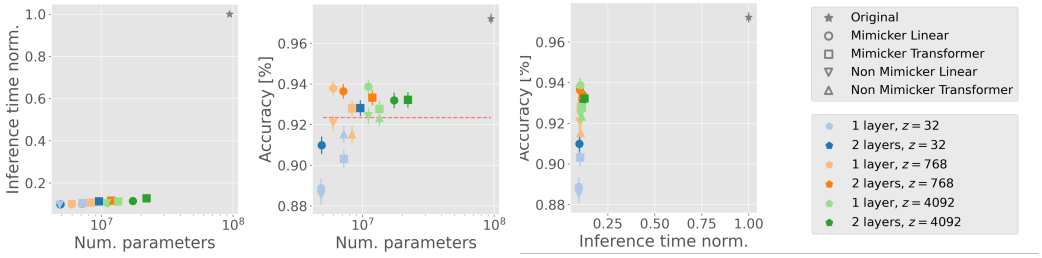

Figure 5: `wavLM-small`

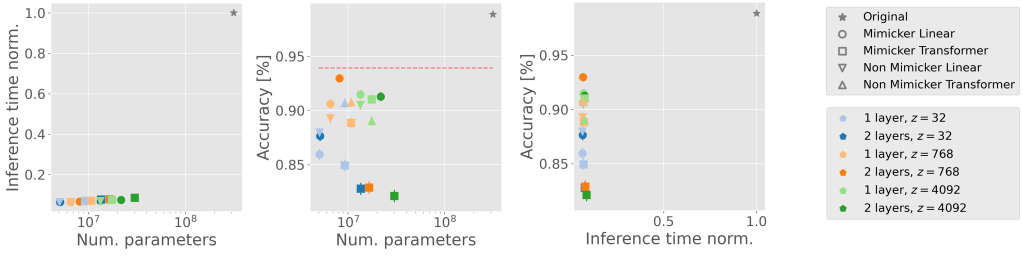

Figure 6: `wavLM-large`

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`