Interspeech 2024
1-5 September 2024, Kos, Greece

SparseWAV: Fast and Accurate One-Shot Unstructured Pruning for Large Speech Foundation Models

Tianteng Gu[1], *Bei Liu*[1], *Hang Shao*[1], *Yanmin Qian*[1*]

[1]Auditory Cognition and Computational Acoustics Lab
MoE Key Lab of Artificial Intelligence, AI Institute
Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China
{995999277,beiliu,hangshao99,yanminqian}@sjtu.edu.cn

Abstract

Self-supervised speech representation learning has shown remarkable capability in automatic speech recognition. However, it requires substantial computations and storage capacity. Pruning is an effective method for model compression. In this work, we propose SparseWAV, a fast and accurate unstructured pruning framework designed for large speech foundation models, which can efficiently remove unimportant parameters without sacrificing performance. It adaptively determines the sparsity ratio for each weight matrix within pre-trained models and updates the remaining parameters to compensate for the eliminated ones. Experiments on Librispeech demonstrate the proposed method can remove **80%** of the parameters of pre-trained large speech foundation models with negligible performance loss. Compared to previous works, our resulting models achieves up to **30%** improvement in performance under similar parameters. Meanwhile, the compression algorithm's time consumption is reduced by up to **1080x**.

Index Terms: model compression, unstructured pruning, speech recognition

1. Introduction

Self-supervised speech representation learning has demonstrated impressive results in automatic speech recognition. [1, 2, 3, 4]. However, these pre-trained models come with significant computational expenses and extensive storage needs, hindering their widespread adoption in consumer products. Recently, there has been considerable interest in compressing these models.

The current two mainstream methods are knowledge distillation and network pruning. Knowledge distillation utilizes a large teacher model to guide a smaller student model. Previous studies such as DistilHuBERT [5] and FitHuBERT [6] have achieved promising results. However, distillation requires an amount of training time and the design of student model needs specific expertise to achieve better performance. Network pruning discovers a compact subnetwork from a large model by removing unimportant weights. Recent works like DPHUBERT [7, 8, 9] apply L0 regularization based structured pruning to compress speech models. However, previous methods require an additional round of training on the dataset to determine the pruning mask, which introduces extra computational overhead. Moreover, structured pruning often leads to greater performance degradation as it uses structures rather than individual weights as the basic unit of pruning, which might result in the removal of some important parameters.

*Corresponding author

In comparison, unstructured pruning offers higher degrees of freedom, allowing non-important parameters in the weight matrix to be individually removed. Thus, models pruned with unstructured approaches generally exhibit better performance. PARP [10] is one of the methods for unstructured pruning speech models. However, PARP has only been validated on low-resource speech recognition tasks and causes significant performance degradation under high sparsity conditions.

In this paper, we propose SparseWAV, a fast and accurate unstructured pruning method suitable for large speech foundation models. It can efficiently remove the majority of parameters within pre-trained speech models without performance degradation and performs well under high sparsity conditions. Specifically, we first introduce Optimal Brain Surgeon (OBS)[11] framework, tailed for pruning large speech models. To further enhance performance, a refined saliency criterion that integrates first-order information is proposed. In addition, we present a scheme to dynamically assign varying sparsity ratios to each weight matrix according to the Hessian based sensitivity. Finally, two different post-pruning finetune strategies are developed to recover the performance of the pruned models.

Experimental results on LibriSpeech dataset demonstrate that our method can be applied to various pre-trained speech models, including wav2vec2-base, large and wavlm. Up to 80% parameters can be removed with negligible performance degradation. In addition, a thorough comparison with previous works reveals that our method achieves a new state-of-the-art performance with similar parameters. Moreover, our approach also significantly reduces the time consumption compared to previous strategies. Unlike the extensive GPU hours required for knowledge distillation, our method efficiently prunes wav2vec2-base in merely 80 seconds and wav2vec2-large in 210 seconds on a single RTX 4090. Notably, at sparsity levels of 50% or lower, models pruned with our technique achieve the same performance as their original dense counterparts without the need for fine-tuning.

2. Methods

2.1. Preliminary

OBS is a second information based pruning method, it prunes one parameter then updates the remaining to compensate for the performance degradation until the target sparsity is reached. The saliency of parameters which measures the change in the loss function due to pruning and the updates of the remaining parameters $\boldsymbol{\delta}$ are derived through Taylor expansion and Lagrange multipliers. More precisely

$$\text{saliency}_p = \frac{w_p^2}{[\mathbf{H}^{-1}]_{pp}}, \quad \boldsymbol{\delta_p} = -\frac{w_p}{[\mathbf{H}^{-1}]_{pp}} \cdot [\mathbf{H}^{-1}]_{:,p}, \quad (1)$$

10.21437/Interspeech.2024-607

where w_p denote the value of the pth paramter, $\mathbf{H}$ denote the Hessian matrix of model's loss function, $[\mathbf{H}^{-1}]_{pp}$ denotes the pth diagonal entry of the inverse Hessian, and $[\mathbf{H}^{-1}]_{:,p}$ is its pth column. Note that the OBS framework assumes that the model has converged, so it neglects the first-order term of the Taylor expansion. At the same time, it assumes that higher-order terms are small and can be ignored. Therefore, the saliency formula only includes second-order terms.

To reduce the computation cost of OBS due to the calculation of the Hessian of the whole model, we follow [12, 13] to decompose the pruning task into a series of layer-wise pruning and reconstruction of a single weight matrix $\mathbf{W}$:

$$\text{argmin}_{\widehat{\mathbf{W}}} \quad ||\mathbf{W}\mathbf{X} - \widehat{\mathbf{W}}\mathbf{X}||_2^2 \quad \text{s.t.} \quad \text{sparsity}(\widehat{\mathbf{W}}) \geq S. \quad (2)$$

where $\widehat{\mathbf{W}}$ denotes the pruned weight matrix, $\mathbf{X}$ is the input of dimension $d_{\text{col}} \times N$ and S is the target sparsity. Using Formula 2 as the loss function, we find that each row of matrix W has the same Hessian matrix: $2\mathbf{X}\mathbf{X}^{\mathbf{T}}$. This can be calculated using a formula, without the need for complex framework-based differentiation, thereby saving time.

2.2. Improved Saliency with First-order Information

Through analysis and experimentation, we found that adding the first-order term of the Taylor expansion to the saliency formula describing parameter importance can improve the performance of pruning methods employing approximations. The improved saliency is formulated as follow:

$$\text{improved saliency}_p = |w_p \mathbf{G}_p| + \frac{w_p^2}{[\mathbf{H}^{-1}]_{pp}} \quad (3)$$

where $\mathbf{G}_p$ denotes the gradient of the loss function with respect to the p-th parameter. In the case of layer-wise pruning, the loss function is Formula 2 and $\mathbf{G}_{ij} = 2(\widehat{\mathbf{W}} - \mathbf{W})_{i,:}(\mathbf{X}\mathbf{X}^{\mathbf{T}})_{:,j}$ for the j-th parameter in the i-th column of the weight matrix.

We have several reasons for doing this: First, OBS itself introduces errors when computing the updates of the remaining parameters by neglecting higher-order terms. Additionally, layer-wise pruning, by considering only the linear part and neglecting the effect of the non-linear activation function, further increases the error in the updates of the remaining parameters. After pruning a certain number of parameters, the model no longer converges on the dataset due to accumulated errors. At this point, the model's gradient with respect to the loss function is no longer zero, rendering OBS's assumption invalid.

2.3. Mixed Sparsity Pruning

Each layer of the speech SSL model contributes differently to the overall performance of the model. Therefore, it's reasonable to assign different sparsities for different weight matrices to maximize the retention of the model's performance. We develop a novel method capable of dynamically setting the sparsity ratio for each weight matrix. This approach involves calculating the sensitivity of each weight matrix by approximating the trace of the Hessian matrix with respect to the loss function, and then determining the sparsity level for each weight matrix based on this sensitivity.

First, we estimate the trace of model's Hessian matrix using stochastic linear algebra methods [14] and the Hutchinson algorithm [15], because accurately computing the Hessian matrix of a speech SSL model is time and space-consuming.

$$\begin{aligned} \text{Tr}(\mathbf{H}) &= \text{Tr}(\mathbf{H}\mathbf{I}) = \text{Tr}(\mathbf{H}\text{E}[\mathbf{z}\mathbf{z}^{\mathbf{T}}]) = \text{E}[\text{Tr}(\mathbf{H}\mathbf{z}\mathbf{z}^{\mathbf{T}})] \\ &= \text{E}[\mathbf{z}^{\mathbf{T}}\mathbf{H}\mathbf{z}] \approx \frac{1}{N}\sum_{i=1}^{N} \mathbf{z}_i^T \mathbf{H}\mathbf{z}_i \end{aligned} \quad (4)$$

where $\mathbf{I}$ denotes identity matrix, and $\mathbf{z}$ is a random vector sampled from a standard Gaussian distribution, N denotes the number of sampled data points.

Then, we compute the average trace of the parts of the Hessian matrix related to each weight matrix as the sensitivity of that weight matrix.

$$\text{sensitivity} = \frac{1}{n}\Sigma_i^n \text{Tr}(\mathbf{H})_i \quad (5)$$

where n denotes the number of diagonal elements in the corresponding part of Hessian matrix.

After computing the sensitivity of all weight matrices, we sort them and assign sparsity to each weight matrix based on their rankings. Due to the constraints that are difficult to satisfy in sparsity assignment, such as ensuring that the sparsity of weight matrices falls within the range of 0 to 1 and that the weighted sum equals the target sparsity of the model as a whole, we adopted a simple approach: using an arithmetic progression to allocate sparsity. The detailed process is as follows:

$$\text{Lower bound} = s - \alpha, \;\; \text{Higher bound} = s + \alpha \quad (6)$$

$$\text{sparsity}_i = \text{Lower bound} + \text{rank}_i \times \frac{2\alpha}{N-1} \quad (7)$$

where sparsity_i denotes the allocated sparsity for i-th weight matrix, s denotes the target sparsity of the whole model, α is a hyperparameter that controls the range of sparsity, N denotes the total number of weight matrix in the model, rank_i denotes the ranking (base 0) of sensitivity of the i-th weight matrix from highest to lowest. Thus, weight matrices with higher sensitivity are assigned lower sparsity, allowing for the retention of more important parameters.

2.4. Post-pruning Fine-tuning

While our proposed method can compensate for the performance degradation by updating the remaining parameters, the pruned model's performance still suffers a significant reduction due to approximation errors, particularly under conditions of high sparsity($\geq 70\%$). To recover the performance, we introduce two strategies for fine-tuning after pruning:

One-pass. We directly prune the model to the target sparsity level and then fine-tune the pruned model in a one-pass way.

Progressive. Models are pruned and finetuned in an iterative manner until the target sparsity is reached. We refer to this method as **SparseWAV-progress**. For example, if the target sparsity is 80%, SparseWAV-progress first prunes the dense model to 70% and finetunes the pruned model, then subsequently prunes it from 70% sparsity to 80% sparsity and finetunes it. Take models with 90% sparsity as another example, its sparsity increases progressively in increments of 0%-70%-80%-90%.

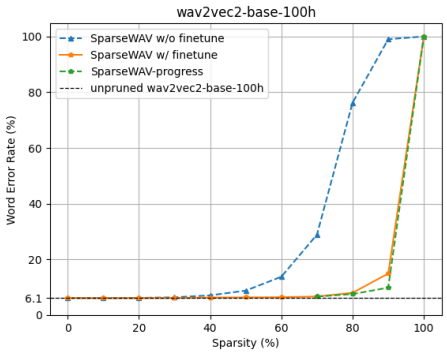

Figure 1: *The ASR performance of wav2vec2-base-100h pruned at different sparsities using SparseWAV.*

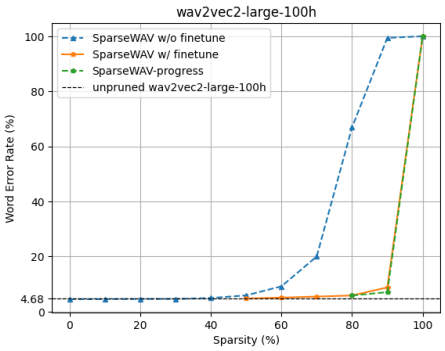

Figure 2: *The ASR performance of wav2vec2-large-100h pruned at different sparsities using SparseWAV.*

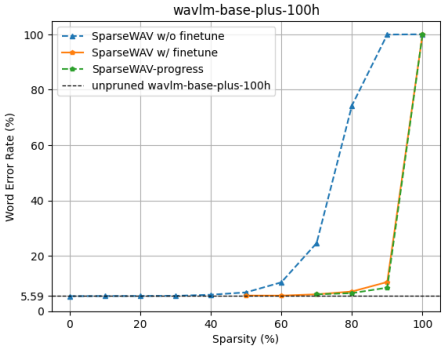

Figure 3: *The ASR performance of wavlm-base-plus-100h pruned at different sparsities using SparseWAV.*

3. Experiment

3.1. Experimental Setup

Toolkits.Our method is implemented using the PyTorch [16] and Hugging Face [17]. Pretrained and finetuned models are downloaded from fairseq [18] and Hugging Face.

Data. The train-clean-100h set of LibriSpeech [19] is utilized for finetuning, while the test-clean set is utilized for evalution. The calibration dataset consists of 2048 random samples from the LibriSpeech train-clean-100h dataset

Model. We examine the performance of the proposed method on wav2vec2-base, wav2vec2-large and wavlm-base-plus. The pre-trained weights of these models are downloaded from Hugging Face and fairseq and re-loaded using AutoModeForCTC from the Hugging Face's transformer library. It is worth noting that a CTC decoder without language models is employed.

Pruning. Pruning is performed on a single RTX 4090. The α for SparseWAV was set to 0.1. It takes 80 seconds to prune wav2vec2-base, 210 seconds to prune wav2vec2-large.

Fine-tuning. The configuration of fine-tuning depends on the performance loss of the pruned model, typically requiring more epochs for models with higher sparsity. For example, with a sparsity of 50% for wav2vec2-base, we conduct fine-tuning for 40 epochs using the Adam optimizer with betas=(0.9, 0.999) and epsilon=1e-08. The learning rate was set to 3e-5, with a warm-up of 1000 steps, and a batch size of 16.

3.2. Main Results

In this section, we examine the performance of SparseWAV on various large speech foundation models, including wav2vec2-base/large [2] and wavlm-base-plus. Overall, our method can prune over 70% of the parameters of the model with almost no loss in performance.

From Figure 1, 2 and 3, it can be clearly seen that the proposed pruning method can remove over 70% of the parameters of the model with negligible performance loss. For wav2vec2-base (Figure 1), when the sparsity is less than or equal to 70%, the pruned models do not show significant performance degradation, and in some cases, models with lower sparsity even outperform the original model. This observation is consistent with previous work [9, 10], suggesting that pruning may provide regularization that enhances model generalization. At 90% sparsity, the model pruned using SparseWAV-progress only incurs a 3.69% absolute WER increase. As Figure 2 and 3 display, similar phenomena can been observed on wav2vec2-large-100h and wavlm-base-plus-100h respectively, illustrating its broad applicability across various pre-trained speech models.

Furthermore, we find that the better the performance of the original model, the smaller the performance loss caused by pruning. Figure 4 illustrates the performance of one-shot pruned wav2vec2-large models finetuned on different datasets. These models were not finetuned after pruning. Models trained more extensively have more accurate parameters, allowing SparseWAV to more effectively offset the impact of pruning through updates to the remaining parameters. Notably, when the sparsity is below 50%, the performance of the pruned wav2vec2-large model remains consistent with the original model even without post-pruning fine-tuning.

3.3. Ablation Study

In this section, we analyze the effects of the suggested pruning techniques on performance. Taking wav2vec2-base-100h as an example, the sparsity ratio is set to 75% in the experiments. Table 1 clearly demonstrates that the pruning methods we have introduced can lead to consistent improvements in the performance of the pruned models.

We take the OBS with uniform sparsity as the baseline for comparison. As shown in the second row of Table 1, the WER of the model pruned uniformly reached 57.95%, far exceeding that of the dense model at 6.1%. The performance of the model is severely degraded and cannot be used.

Mixed Sparsity. SparseWAV assigns specific sparsity to each weight matrix. Model with mixed sparsity (third row) outper-

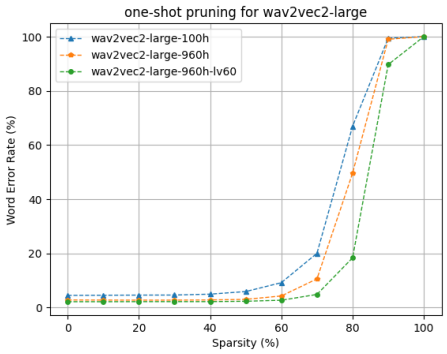

Figure 4: *The ASR performance comparison of wav2vec2-large models finetuned on datasets of varying sizes, without post-pruning finetuning.*

Table 1: *Ablation study conducted to verify the effectiveness of the improvements we proposed. The experiments were carried out on the wav2vec2-base-100h model with 75% transformer sparsity*

Method	ASR w/o LM
	WER(%) ↓
wav2vec2-base-100h	6.1
OBS [11]	57.95
+mixed sparsity	53.79
++improved saliency	53.01
+++post-pruning finetune	7.54
++++progress	**7.11**

formes model with uniform sparsity with a WER of 53.79% compared to 57.95% for uniform sparsity., proving the necessity of mixed sparsity. This is because different weight matrices contribute differently to model performance.

Improved Saliency. Saliency criterion is improved in SparseWAV. The forth row shows that improved saliency could more accurately identify parameters that can be pruned.

Post-pruning Finetune. SparseWAV finetunes models after pruning when target sparsity is high. The fifth row demonstrates that post-pruning finetune can significantly enhance the performance of pruned models from 53.01% to 7.54%.

Progress. The performance of pruned model could be further recovered by SparseWAV-progress. The sixth row shows that iterative pruning could improved perfomance from 7.54% to 7.11%, closer to the dense model.

3.4. Comparison with Other Compression Methods

This section compares the performance of SparseWAV and previous model compression methods for speech models. We prune 85% of the parameters in the transformer part of the wav2vec2 model to ensure that the pruned model's size is similar to the previously compressed model. The WER of the previous method originates from leaderboard of SUPERB [21]. As illustrated in table 2, SparseWAV significantly outperforms distillation methods and L0 regularization-based structured pruning methods by at least 2% absolute WER. This indicates that

Table 2: *Performance comparison of our method versus previous compression methods on WER. SparseWAV2VEC2 and SparseWAVLM+ are pruned from wav2vec2-base and wavlm-base-plus respectively with 85% transformer sparsity.*

Method	Params	ASR w/o LM
	Millions	WER(%) ↓
Baselines		
wav2vec2 Base [2]	95.04	6.43
HuBERT Base [3]	94.68	6.42
WavLM Base+ [4]	94.70	5.59
Prior Compression Methods		
DistilHuBERT [5]	23.49	13.37
FitHuBERT [6]	22.49	12.09
12-Layer Half [20]	26.87	10.96
DPHuBERT [7]	23.59	10.47
DPWavLM [7]	23.59	10.19
Wang et al. [8]	26.57	10.61
Ours		
SparseWAV2VEC2	22.27	**8.08**
SparseWAVLM+	22.98	**7.12**

SparseWAV can better preserve the speech recognition performance of the original model.

Table 3: *Time-consuming comparison of our method versus previous compression methods on WER. We measure the time costed by SparseWAV to one-shot prune models.*

Method	Model	Time	Speedup
Base model			
DistilHuBERT [5]	HuBERT-base	55 Hours	–
DPHuBERT [7]	HuBERT-base	24 Hours	1x
SparseWAV (ours)	wav2vec2-base	80 Seconds	**1080x**
Large model			
DPHuBERT [7]	HuBERT-large	60 Hours	1x
SparseWAV (ours)	wav2vec2-large	210 Seconds	**1028x**

Regarding algorithm efficiency, we list the time consumption in Table 3. Obviously, the time required for SparseWAV is significantly lower than that of previous methods, offering an acceleration of at least 1080x. It should be noted that previous methods require distillation on 960h data and fine-tuning on 100h data, whereas our approach merely involves fine-tuning on 100h of data. This indicates the effectiveness and superiority of the proposed methods.

4. Conclusion

In this work, we introduce SparseWAV, a fast and accurate method for pruning large speech models. It is a general and can prune speech models of various sizes such as wav2vec2-base/large and wavlm base+. SparseWAV effectively preserves the performance of the original model and surpasses previous model compression methods for speech models. In the future, we will continue to explore methods to determine the sparsity of each layer of the model to further improve performance.

5. Acknowledgement

This work was supported in part by China STI 2030-Major Projects under Grant No. 2021ZD0201500, in part by China NSFC projects under Grants 62122050 and 62071288, and in part by Shanghai Municipal Science and Technology Commission Project under Grant 2021SHZDZX0102.

6. References

[1] A. Mohamed, H.-y. Lee, L. Borgholt, J. D. Havtorn, J. Edin, C. Igel, K. Kirchhoff, S.-W. Li, K. Livescu, L. Maaløe, T. N. Sainath, and S. Watanabe, "Self-supervised speech representation learning: A review," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 6, pp. 1179–1210, 2022.

[2] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 12 449–12 460. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2020/file/92d1e1eb1cd6f9fba3227870bb6d7f07-Paper.pdf

[3] W.-N. Hsu, B. Bolte, Y.-H. H. Tsai, K. Lakhotia, R. Salakhutdinov, and A. Mohamed, "Hubert: Self-supervised speech representation learning by masked prediction of hidden units," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 3451–3460, 2021.

[4] S. Chen, C. Wang, Z. Chen, Y. Wu, S. Liu, Z. Chen, J. Li, N. Kanda, T. Yoshioka, X. Xiao, J. Wu, L. Zhou, S. Ren, Y. Qian, Y. Qian, J. Wu, M. Zeng, X. Yu, and F. Wei, "Wavlm: Large-scale self-supervised pre-training for full stack speech processing," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 6, pp. 1505–1518, 2022.

[5] H.-J. Chang, S.-w. Yang, and H.-y. Lee, "Distilhubert: Speech representation learning by layer-wise distillation of hidden-unit bert," in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 7087–7091.

[6] Y. Lee, K. Jang, J. Goo, Y. Jung, and H. R. Kim, "FitHuBERT: Going Thinner and Deeper for Knowledge Distillation of Speech Self-Supervised Models," in *Proc. Interspeech 2022*, 2022, pp. 3588–3592.

[7] Y. Peng, Y. Sudo, S. Muhammad, and S. Watanabe, "DPHuBERT: Joint Distillation and Pruning of Self-Supervised Speech Models," in *Proc. INTERSPEECH 2023*, 2023, pp. 62–66.

[8] H. Wang, S. Wang, W.-Q. Zhang, S. Hongbin, and Y. Wan, "Task-Agnostic Structured Pruning of Speech Representation Models," in *Proc. INTERSPEECH 2023*, 2023, pp. 231–235.

[9] Y. Peng, K. Kim, F. Wu, P. Sridhar, and S. 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)*, 2023, pp. 1–5.

[10] C.-I. J. Lai, Y. Zhang, A. H. Liu, S. Chang, Y.-L. Liao, Y.-S. Chuang, K. Qian, S. Khurana, D. Cox, and J. Glass, "Parp: Prune, adjust and re-prune for self-supervised speech recognition." in *NeurIPS*, 2021.

[11] B. Hassibi and D. Stork, "Second order derivatives for network pruning: Optimal brain surgeon," in *Advances in Neural Information Processing Systems*, S. Hanson, J. Cowan, and C. Giles, Eds., vol. 5. Morgan-Kaufmann, 1992. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/1992/file/303ed4c69846ab36c2904d3ba8573050-Paper.pdf

[12] E. Frantar, S. P. Singh, and D. Alistarh, "Optimal Brain Compression: a framework for accurate post-training quantization and pruning," *Advances in Neural Information Processing Systems*, vol. 36, 2022.

[13] E. Frantar and D. Alistarh, "SparseGPT: Massive language models can be accurately pruned in one-shot," *arXiv preprint arXiv:2301.00774*, 2023.

[14] M. W. Mahoney, 2011.

[15] H. Avron and S. Toledo, "Randomized algorithms for estimating the trace of an implicit symmetric positive semi-definite matrix," *Journal of the ACM*, p. 1–34, Apr 2011. [Online]. Available: http://dx.doi.org/10.1145/1944345.1944349

[16] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035.

[17] 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. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush, "Huggingface's transformers: State-of-the-art natural language processing," 2019.

[18] M. Ott, S. Edunov, A. Baevski, A. Fan, S. Gross, N. Ng, D. Grangier, and M. Auli, "fairseq: A fast, extensible toolkit for sequence modeling," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, W. Ammar, A. Louis, and N. Mostafazadeh, Eds. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 48–53. [Online]. Available: https://aclanthology.org/N19-4009

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

[20] T. Ashihara, T. Moriya, K. Matsuura, and T. Tanaka, "Deep versus Wide: An Analysis of Student Architectures for Task-Agnostic Knowledge Distillation of Self-Supervised Speech Models," 2022.

[21] 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.