INTERSPEECH 2023
20-24 August 2023, Dublin, Ireland

CoMFLP: Correlation Measure based Fast Search on ASR Layer Pruning

Wei Liu, Zhiyuan Peng, Tan Lee

Department of Electronic Engineering, The Chinese University of Hong Kong
louislau_1129@link.cuhk.edu.hk, jerrypeng1937@gmail.com, tanlee@cuhk.edu.hk

Abstract

Transformer-based speech recognition (ASR) model with deep layers exhibited significant performance improvement. However, the model is inefficient for deployment on resource-constrained devices. Layer pruning (LP) is a commonly used compression method to remove redundant layers. Previous studies on LP usually identify the redundant layers according to a task-specific evaluation metric. They are time-consuming for models with a large number of layers, even in a greedy search manner. To address this problem, we propose CoMFLP, a fast search LP algorithm based on correlation measure. The correlation between layers is computed to generate a correlation matrix, which identifies the redundancy among layers. The search process is carried out in two steps: (1) coarse search: to determine top K candidates by pruning the most redundant layers based on the correlation matrix; (2) fine search: to select the best pruning proposal among K candidates using a task-specific evaluation metric. Experiments on an ASR task show that the pruning proposal determined by CoMFLP outperforms existing LP methods while only requiring constant time complexity. The code is publicly available at `https://github.com/louislau1129/CoMFLP`.

Index Terms: fast layer pruning, correlation measure, model compression, speech recognition.

1. Introduction

Neural network based ASR models with deep layers have shown superior performance [1–3]. This is mainly due to the recent advances in the transformer architecture [4–6] and the effectiveness of scaling laws [7]. However, large and deep models are difficult, if not impossible, to be deployed on resource-constrained devices at the edge [8–10].

Model compression [11–17] was extensively studied for realizing large models on small-footprint devices without significant performance degradation. One of the predominant approaches is pruning [15–17]. Depending on whether the pruned parameters are grouped according to inherent structures (e.g., channel, head, and layer), there are unstructured [18] and structured pruning [19–22] approaches. Unstructured pruning treats each individual weight as the unit of pruning, often resulting in a sparse model that is not suitable for speedup on current hardware [23]. Structured pruning can achieve great acceleration, especially when dropping many layers [22, 24].

It is noted that the transformer stacked with a large number of layers, e.g., DeepNet with 1,000 layers [25], demonstrates significant performance. However, it comes at the cost of higher computational resources and memory consumption. It is hypothesized that some of the many layers might be redundant and have little contribution to the overall system performance [26]. This motivates the present study to inspect the redundancy among layers and perform layer-level structured pruning, i.e., layer pruning (LP), for simplifying deep models.

Let L be the total number of layers. The core of LP is a pruning proposal that stipulates N of the L layers to be removed. The optimal pruning proposal can be found by enumerating all possible combinations of N layers and selecting the best one. For ASR systems, performance metrics like word error rate (WER) are used for determining the best proposal. Apparently, the high time complexity $\mathcal{O}(C_L^N)$ makes exhaustive search practically infeasible. In [27], the method of linear probing was applied to estimate the importance of different layers. The layer output was extracted as an input feature to a linear classifier for task-specific performance evaluation. If two consecutive layers' performances are similar, one of them would be removed. In this case, a total of L classifiers need to be trained. In [28], greedy layer pruning (GLP) was proposed to reduce the complexity of the pruning proposal search. The key idea is that the best solution for pruning $(n+1)$ layers is built on top of the already known best solution for pruning n layers, such that only $(L-n)$ possible combinations need to be evaluated. With this simplification, the complexity of the pruning proposal search is reduced from $\mathcal{O}(C_L^N)$ to $\mathcal{O}(L*N)$. Several other pruning strategies, like top-layer, middle-layer, and bottom-layer dropping, etc., were empirically studied in [29].

Inspired by [26], if two layers' outputs are similar, the layers between them are assumed to be redundant and can be discarded. In the present study, we propose the **Co**rrelation **M**easure based **F**ast Search on **L**ayer **P**runing (CoMFLP). Two correlation methods, singular vector canonical correlation analysis (SVCCA) and distance correlation (DC), are computed for any pair of layers, leading to a correlation matrix that represents the pair-wise layer similarity. Specifically, CoMFLP comprises two steps; (1) coarse search: top-K pruning proposals are determined using a layer-wise beam search on the correlation matrix; (2) fine search: WER is used to select the best one among the K candidates. The time complexity of the proposed method is approximately constant $\mathcal{O}(K)$ since step (1) has negligible computation cost. It is much faster than WER-based GLP ($\mathcal{O}(L*N)$) and has a better performance. Experiments show that the pruning proposal determined by CoMFLP can provide a very promising starting point for later fine-tuning in terms of convergence and initialization.

2. CoMFLP

We inspect a well-trained neural network for ASR and attempt to apply layer pruning to achieve good compromise between efficiency and performance. A two-stage fast search algorithm is proposed. As depicted in Fig. 1, it includes a coarse search and

10.21437/Interspeech.2023-1469

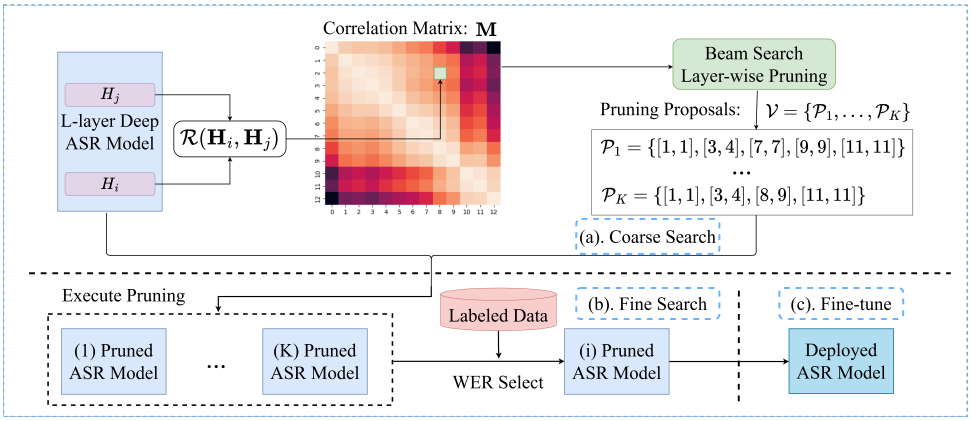


Figure 1: *The overall diagram of the proposed CoMFLP method on ASR, where L=12 is illustrated as an example.*

a fine search. Given an L-layer network, an $(L+1)$-by-$(L+1)$ correlation matrix is generated in the coarse search. With the correlation matrix, pruning proposals are evaluated, compared, and ranked. The top K proposals $\mathcal{V} = \{\mathcal{P}_1, ..., \mathcal{P}_K\}$ are retained for the subsequent fine search. In the fine search, task-specific performance metrics on labeled data are used to evaluate the K candidate proposals and select the best one. Only constant time complexity $\mathcal{O}(K)$ is required for the entire search.

2.1. Correlation matrix

The correlation matrix measures the similarity among layers. If the input and output of a block of layers are very similar, the layers within that black are considered to be redundant. Layer redundancy thus can be reflected by the correlation matrix. For speech recognition, consider a batch of B speech samples $\mathbf{S} = \{\mathbf{s}_i\}_{i=1}^{B}$. In a CTC-based ASR system, the samples are zero-padded and encoded into frame-level hidden representations $\mathbf{H}_0 \in \mathbb{R}^{B\times T_0\times D_0}$, where T_0 denotes the number of frames and D_0 represents the hidden dimension. The hidden representation $\mathbf{H}_0$ is then forwarded into an L-layer Transformer $\mathbf{Enc}$. The layer-wise forward process can be formulated as follows:

$$\mathbf{H}_i = \mathbf{Enc}^i(\mathbf{H}_{i-1});\ \ 1 \le i \le L, \tag{1}$$

where $\mathbf{H}_i \in \mathbb{R}^{B\times T_i\times D_i}$ denotes the hidden representation produced by the i-th layer of the transformer $\mathbf{Enc}$. The hidden representations are averaged along the time axis, i.e., $\bar{\mathbf{H}}_i \longleftarrow \text{TempoAverage}(\mathbf{H}_i)$, where $\bar{\mathbf{H}}_i \in \mathbb{R}^{B\times D_i}$. Finally, singular vector canonical correlation analysis (SVCCA) or distance correlation (DC) is applied on $\{\bar{\mathbf{H}}_i\}_{i=0}^{L}$ to derive the correlation matrix $\mathbf{M} \in \mathbb{R}^{(L+1)\times(L+1)}$.

2.1.1. SVCCA

SVCCA is developed to measure the similarity between two hidden layer representations [30]. It involves two steps, namely singular value decomposition (SVD) and canonical correlation analysis (CCA) [31]. SVD is applied to $\bar{\mathbf{H}}_i$ to obtain the subspace $\bar{\mathbf{H}}'_i \in \mathbb{R}^{B\times D'_i}$ that corresponds to the D'_i largest singular values (up to a pre-defined ratio of the total variance, e.g., 99%). Similarly, $\bar{\mathbf{H}}'_j \in \mathbb{R}^{B\times D'_j}$ can be obtained by applying SVD to $\bar{\mathbf{H}}_j$. Second, CCA is applied to project $\bar{\mathbf{H}}'_i$ and $\bar{\mathbf{H}}'_j$ into a shared subspace, where the correlation between the projected matrices is maximized. The correlation coefficients are averaged into a scalar in the range of 0 to 1, which is defined as the SVCCA similarity score between $\bar{\mathbf{H}}_i$ and $\bar{\mathbf{H}}_j$. A detailed description of SVCCA can be found in [30]. Applying SVCCA on all possible i, j gives the correlation matrix $\mathbf{M}$.

2.1.2. DC

DC [32, 33] provides a statistical measure of dependence between random vectors. Consider the hidden representation $\bar{\mathbf{H}}_i \in \mathbb{R}^{B\times D_i}$, where the row vector $\mathbf{h}_k^i \in \mathbb{R}^{D_i}$ corresponds to the i-th layer's response of the k-th sample. The distance matrix $\mathbf{A}^i \in \mathbb{R}^{B\times B}$ measures the normalized distances between row vectors in $\bar{\mathbf{H}}_i$. It can be computed as follows,

$$\mathbf{a}_{k,l}^i = ||\mathbf{h}_k^i - \mathbf{h}_l^i||_2, \tag{2}$$

$$\mathbf{A}_{k,l}^i = \mathbf{a}_{k,l}^i - \frac{1}{M}\sum_{l=1}^{M}\mathbf{a}_{k,l}^i - \frac{1}{M}\sum_{k=1}^{M}\mathbf{a}_{k,l}^i + \frac{1}{M^2}\sum_{k,l=1}^{M}\mathbf{a}_{k,l}^i, \tag{3}$$

where $\mathbf{A}_{k,l}^i$ refers to the element of $\mathbf{A}^i$ at k-th row and l-th column. To this end, the similarity score $\mathbf{M}_{i,j}$ between $\bar{\mathbf{H}}_i$ and $\bar{\mathbf{H}}_j$ is given by,

$$\mathbf{M}_{i,j} = \left(\frac{V_{i,j}}{\sqrt{V_{i,i}V_{j,j}}}\right)^{\frac{1}{2}}, \tag{4}$$

where the distance covariance $V_{i,j} = \frac{1}{B^2}\sum_{k,l=1}^{B}\mathbf{A}_{k,l}^i\mathbf{A}_{k,l}^j$ and $0 \le \mathbf{M}_{i,j} \le 1$.

2.2. Coarse search

The coarse search is to efficiently find out (maybe with loss of precision) the top K pruning proposals. To perform a coarse search, a method of measuring the quality of a pruning proposal is needed. In the correlation matrix $\mathbf{M}$, the element $\mathbf{M}_{i,j}$ represents the similarity between the outputs from the i-th layer and the j-th layer. It reflects the redundancy of layers $i+1$ to j, and therefore can be leveraged to measure the quality of pruning proposals. Consider a pruning proposal $\mathcal{P} = \{[s_1, e_1], ..., [s_p, e_p]\}$ that discards the layers from s_1 to e_1,$\cdots$, s_p to e_p, where $1 \le s_1 \le e_1 < \cdots < s_p \le e_p \le L$. The quality of $\mathcal{P}$ is defined as:

$$v(\mathcal{P}) = \frac{1}{p}\sum_{t=1}^{p}\mathbf{M}_{s_t-1,e_t}. \tag{5}$$

$v(\mathcal{P})$ is expected to positively correlate with the performance of the pruned model. Higher quality would likely (but not necessarily [1]) lead to a better pruned model.

Consider the goal of pruning N layers from an L-layer neural network. There could be numerous pruning proposals, making the exact search for the best one, if not intractable, time-consuming. On the other hand, the coarse search is able to efficiently find out the top K pruning proposals that can be regarded as the most potential candidates. The intuition is that a good N-layer pruning proposal is most likely to be derived from that of pruning $(N-1)$ layers. This leads to a recursive beam search algorithm, as illustrated in Algorithm 1. Specifically, suppose that a set of K proposals of pruning $(N-1)$ layers are found. For each of the proposals, we enumerate all possible choices of selecting an additional layer for pruning, leading to new proposals of pruning N layers. The new proposals are ranked according to the metric defined in Eq. (5). Finally, only the top K good proposals are selected. Algorithm 1 provides the implementation details, where a *min-queue* is used for fast ranking of the top K good proposals.

Algorithm 1: function **coarse_search**(L, N, K)

Notation L: # layers; N: # layers to be pruned; K: beam size;
Output: K pruning proposals: $\mathcal{V} = \{\mathcal{P}_1, \cdots, \mathcal{P}_K\}$
Init $\mathcal{V} = \varnothing; \mathcal{U} = \varnothing$
/* A min priority queue that keeps the K largest elements */
Init $\mathcal{L}$ as an ***min-queue*** filled with K tuples $(0, \varnothing)$
if *N is 0* **then**
  Return $\{\varnothing\}$
end
for $\mathcal{P} \in$ **coarse_search**$(L, N-1, K)$ **do**
  for *layer* $l \in [1, L] \setminus \mathcal{P}$ **do**
    $\mathcal{P}^* = \mathcal{P} \cup [l, l]$; // add a new layer to prune
    $v^* = v(\mathcal{P}^*)$; // the quality metric, refer to Eq. (5)
    if $\mathcal{P}^* \notin \mathcal{U}$ **then**
      $\mathcal{U} = \mathcal{U} \cup \{\mathcal{P}^*\}$; // deduplicate the set of $\mathcal{P}^*$
      if $v^* > \min(\mathcal{L})$ **then**
        pop($\mathcal{L}$) ; // remove the tuple with smallest v^*
        push($\mathcal{L}, (v^*, \mathcal{P}^*)$) ; // add the new tuple
      end
    end
  end
end
for $(v, \mathcal{P}) \in \mathcal{L}$ **do**
  $\mathcal{V} = \mathcal{V} \cup \{\mathcal{P}\}$
end
Return $\mathcal{V}$

2.3. Fine search

The fine search aims to select the best candidate from the K pruning proposals generated by the coarse search. To achieve this goal, we execute the pruning proposals to generate K pruned models and evaluate them on labeled data. Word error rate (WER) is used as the evaluation metric for ranking. To this end, the best pruned model is selected and fine-tuned. The underlying hypothesis is that a pruned model with a lower WER could generally speed up the fine-tuning process and is very likely to achieve better performance.

[1] E.g., SVCCA is invariant to affine transformation, which cannot be detected by correlation but may have a side effect on the performance.

3. Experimental Setup

3.1. Model, data, and metric

Two Wav2vec2-CTC models [34] from Huggingface are leveraged for layer pruning in our experiments. The first one, denoted as W2V2CTC-12[2], is a 12-layer transformer and pre-trained on AISHELL-2 [35]. The second one, named as W2V2CTC-24[3], is a 24-layer transformer and based on *wav2vec2-large-xlsr-53* [36]. Both two models are fine-tuned on AISHELL-1 [37]. The *dev* and *test* sets of AISHELL-1 are utilized for evaluation. Character error rate (CER) is used as the evaluation metric since our ASR outputs are Chinese characters.

3.2. Configuration for search

- **Correlation matrix:** In SVCCA, the batch size B should be at least 5-10 times larger than the layer dimension D_i for accurate estimation of the correlation matrix [30]. In DC, the batch size B can be small. However, it requires multiple batches (*num_batch*) of samples to estimate. The estimated correlation matrices are averaged as the final result. Our preliminary experiments suggest the hyper-parameters as follows: (1) SVCCA: $B = 9600$, 99% variance kept in SVD; (2) DC: $B = 4$, *num_batch*$= 10$. Note that the speech samples are drawn from the training set of AISHELL-1.
- **Coarse search:** the beam size K in Algorithm 1 is set to 10.
- **Fine search:** 320 utterances are randomly selected from the *test* set and denoted as the *valid* set. They are used for decoding and computing CER in fine search.

3.3. Configuration for fine-tuning

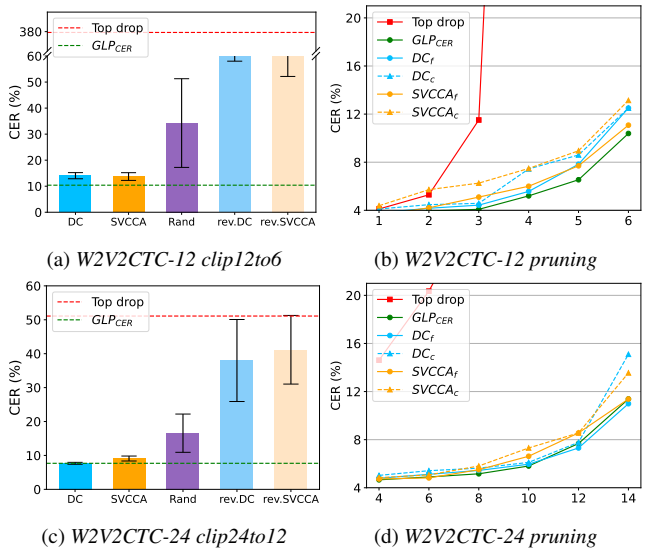
The best pruned model is fine-tuned on the training set of AISHELL-1 for 50 epochs. AdamW [38] is adopted as the optimizer with an initial learning rate of 2e-5. The scheduler *ReduceLROnPlateau* is applied with *patience* = 10 and *factor* = 0.5. Other hyperparameters are set as follows: *batch_size = 32*, *freeze_feature_extractor = False*, *clip_grad_norm = None*, and *layerdrop = 0.1*. No data augmentation is involved.

4. Results and Analysis

4.1. Performance evaluation without fine-tuning

The CER results on the $valid$ set are used to assess the effect of layer pruning. Fig. 2 compares the performance of various pruning methods. In Fig. 2 (a) and (c), *DC/SVCCA* give the coarse search results. The bar height represents the CER mean of the searched 10 pruning proposals, and the error bar denotes their standard deviation (*std.*). The prefix "rev." before *DC/SVCCA* means that the correlation-based search is reversely performed, where the worst pruning proposals are given. *Rand* is to randomly select 10 pruning proposals. Comparing *DC/SVCCA* bars with other bars, the correlation-based search CoMFLP is found to be able to distinguish the good candidates from those terrible ones. In addition, two horizontal lines are given as references. GLP_{CER} is a powerful approach that directly searches with CER in a greedy manner, achieving a relatively good balance between speed and performance. The Top drop is performed by simply dropping the top several layers. It can be observed that the *DC/SVCCA* bars exhibit comparable performance with GLP_{CER} (green line) and are signifi-

[2] kehanlu/mandarin-wav2vec2-aishell1
[3] qinyue/wav2vec2-large-xlsr-53-chinese-zn-cn-aishell1



(a) *W2V2CTC-12 clip12to6* (b) *W2V2CTC-12 pruning*

(c) *W2V2CTC-24 clip24to12* (d) *W2V2CTC-24 pruning*

Figure 2: *The CER results on valid set of different layer pruning methods. (a) and (b) are experimented on the W2V2CTC-12 model, while (c) and (d) are for the W2V2CTC-24 model.*

cantly superior to the *Top drop* (red line).

Fig. 2 (b) and (d) give the performance trend with the varied numbers of pruned layers. When dropping more layers, all methods show monotonically increasing CERs. The subscript "f" denotes the best pruning result after the fine search, while the "c" represents only the coarse search performed, and the pruning proposal indicated by the highest quality metric is output. The DC_c and $SVCCA_c$ are shown to perform worse than their counterparts with a fine search.

In conclusion, the proposed CoMFLP exhibits great power to propose a good pruning proposal that is comparable to GLP_{CER} while requiring only a constant time complexity.

4.2. Performance evaluation after fine-tuning

We then analyze the performance of the pruned model after fine-tuning (FT). Experiments are performed on the pruned W2V2CTC-12 (*clip12to6*). The results of different pruning strategies are given in Table 1. The CoMFLP series achieve the lowest CER results, significantly better than *Scratch* and *Top drop*. The GLP_{CER} achieves only 6.62% CER on *test* set despite the best performance before FT (cf. Fig. 2 (a)). Its low quality in contrast to CoMFLP (0.93 vs. 0.96) could be the reason.

In the CoMFLP block, DC_f and $SVCCA_f$ do not guarantee a better FT result. This may be due to the rather small CER *std.* shown in the current coarse search results (cf. Fig. 2 (a)). The fine search serves to filter out potentially bad proposals that are of high correlation but terrible CER. It is worth noting that $SVCCA_c$ is exactly the proposal to drop every other layer, which is quite reasonable and intuitive for the *clip12to6* case.

Note that LP can be complementary to other compression methods. Better FT results can be achieved by combining the knowledge distillation from the original deep model [39].

4.3. Training dynamics analysis

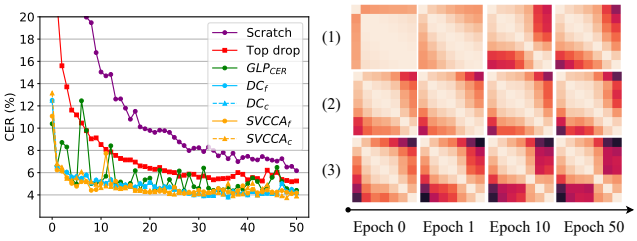
To further illustrate the superiority of CoMFLP, the training dynamics [40] are visualized in Fig. 3. It can be clearly seen that in Fig. 3 (a), the *DC/SVCCA* series converge faster and are more stable than other curves. The GLP_{CER} curve is found

Table 1: *The fine-tuning CER (%) results of different models on dev and test sets of AISHELL-1. The last column represents the pruning proposal's quality metric measured by DC.*

Model		dev	test	quality
W2V2CTC-12		5.10	5.44	-
Layer pruning to 6 layers				
Scratch		9.04	10.11	-
Top drop		7.21	7.80	0.69
GLP_{CER}		6.13	6.62	0.93
CoMFLP	DC_c	5.86	6.28	0.95
	$SVCCA_c$	**5.76**	**6.16**	**0.96**
	DC_f	5.86	6.28	0.95
	$SVCCA_f$	5.91	6.41	0.94

to have a bit of fluctuation, which may imply its sub-optimal performance.

In Fig. 3 (b), the correlation matrices of three different 6-layer models over the training process are plotted as a 3-by-4 heatmap array. They are (1) *Scratch*, (2) *Top drop*, and (3) DC_f. Each of them gives four epoch snapshots. For all three groups, the brightness of the heatmaps becomes darker as training progresses, indicating that the layer redundancy of the network tends to be reduced. From the vertical view, we can see that the DC_f's heatmap is darker than the *Top drop*'s heatmap and further darker than that of the *Scratch*. This could partially explain why the FT performance of DC_f is better than *Top drop*, and the *Scratch* performs the worst. A starting point model at epoch 0 with less redundancy means that different layer representations tend to be more diverse. This initialized diversity could lead to more effective learning during the training process, especially for models with limited capacity.



(a) *CER training curves.* (b) *Dynamic correlation matrix.*

Figure 3: *Training Dynamics with (a) CER and (b) Correlation matrix* M*. Specifically, b.(1)-b.(3) represents three 6-layer models, namely Scratch, Top drop, and* DC_f.

5. Conclusions

This paper presents a correlation measure based fast search layer pruning method called CoMFLP. Two correlation measures, i.e., DC and SVCCA have been attempted. Only constant time complexity is required for the entire search. Experiments on an ASR task show that CoMFLP can efficiently determine pruning proposals of high quality. When applying the searched pruning proposal, better convergence and initialized diversity are found in the fine-tuning process. In future work, the effectiveness and generalization of the method should be examined on deeper layers than the current setup for different tasks.

6. References

[1] J. Li *et al.*, "Recent advances in end-to-end automatic speech recognition," *APSIPA Transactions on Signal and Information Processing*, vol. 11, no. 1, 2022.

[2] Y. Zhang, D. S. Park, W. Han, J. Qin, A. Gulati, J. Shor, A. Jansen, Y. Xu, Y. Huang, S. Wang *et al.*, "Bigssl: Exploring the frontier of large-scale semi-supervised learning for automatic speech recognition," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 6, pp. 1519–1532, 2022.

[3] C. Wang, Y. Wu, S. Chen, S. Liu, J. Li, Y. Qian, and Z. Yang, "Improving self-supervised learning for speech recognition with intermediate layer supervision," in *Proc. of ICASSP*. IEEE, 2022, pp. 7092–7096.

[4] A. Gulati, J. Qin, C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu, and R. Pang, "Conformer: Convolution-augmented transformer for speech recognition," in *Proc. of Interspeech*, 2020, pp. 5036–5040.

[5] N. Chen, S. Watanabe, J. Villalba, P. Żelasko, and N. Dehak, "Non-autoregressive transformer for speech recognition," *IEEE Signal Processing Letters*, vol. 28, pp. 121–125, 2020.

[6] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[7] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei, "Scaling laws for neural language models," *arXiv preprint arXiv:2001.08361*, 2020.

[8] R. Vipperla, S. Ishtiaq, R. Li, S. Bhattacharya, I. Leontiadis, and N. D. Lane, "Learning to listen... on-device: Present and future perspectives of on-device asr," *GetMobile: Mobile Computing and Communications*, vol. 23, no. 4, pp. 5–9, 2020.

[9] Y. Shangguan, J. Li, Q. Liang, R. Alvarez, and I. McGraw, "Optimizing speech recognition for the edge," *arXiv preprint arXiv:1909.12408*, 2019.

[10] Z. Q. Lin, A. G. Chung, and A. Wong, "Edgespeechnets: Highly efficient deep neural networks for speech recognition on the edge," *arXiv preprint arXiv:1810.08559*, 2018.

[11] T. Choudhary, V. Mishra, A. Goswami, and J. Sarangapani, "A comprehensive survey on model compression and acceleration," *Artificial Intelligence Review*, vol. 53, pp. 5113–5155, 2020.

[12] Y. Guo, "A survey on methods and theories of quantized neural networks," *arXiv preprint arXiv:1808.04752*, 2018.

[13] J. Gou, B. Yu, S. J. Maybank, and D. Tao, "Knowledge distillation: A survey," *International Journal of Computer Vision*, vol. 129, pp. 1789–1819, 2021.

[14] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[15] M. Augasta and T. Kathirvalavakumar, "Pruning algorithms of neural networks—a comparative study," *Open Computer Science*, vol. 3, no. 3, pp. 105–115, 2013.

[16] W. Wang, C. Fu, J. Guo, D. Cai, and X. He, "Cop: customized deep model compression via regularized correlation-based filter-level pruning," in *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, 2019, pp. 3785–3791.

[17] D. Blalock, J. J. Gonzalez Ortiz, J. Frankle, and J. Guttag, "What is the state of neural network pruning?" *Proceedings of machine learning and systems*, vol. 2, pp. 129–146, 2020.

[18] T. Chen, J. Frankle, S. Chang, S. Liu, Y. Zhang, Z. Wang, and M. Carbin, "The lottery ticket hypothesis for pre-trained bert networks," *Advances in neural information processing systems*, vol. 33, pp. 15 834–15 846, 2020.

[19] P. Michel, O. Levy, and G. Neubig, "Are sixteen heads really better than one?" *Advances in neural information processing systems*, vol. 32, 2019.

[20] J. McCarley, R. Chakravarti, and A. Sil, "Structured pruning of a bert-based question answering model," *arXiv preprint arXiv:1910.06360*, 2019.

[21] H. Sajjad, F. Dalvi, N. Durrani, and P. Nakov, "Poor man's bert: Smaller and faster transformer models," *arXiv preprint arXiv:2004.03844*, vol. 2, no. 2, 2020.

[22] A. Fan, E. Grave, and A. Joulin, "Reducing transformer depth on demand with structured dropout," in *8th International Conference on Learning Representations, ICLR*, 2020.

[23] V. Sanh, T. Wolf, and A. Rush, "Movement pruning: Adaptive sparsity by fine-tuning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 20 378–20 389, 2020.

[24] M. Xia, Z. Zhong, and D. Chen, "Structured pruning learns compact and accurate models," in *Proc. of ACL*, 2022, pp. 1513–1528.

[25] H. Wang, S. Ma, L. Dong, S. Huang, D. Zhang, and F. Wei, "Deepnet: Scaling transformers to 1,000 layers," *arXiv preprint arXiv:2203.00555*, 2022.

[26] J. Lee, J. Kang, and S. Watanabe, "Layer pruning on demand with intermediate CTC," in *Proc. of Interspeech*, 2021, pp. 3745–3749.

[27] S. Chen and Q. Zhao, "Shallowing deep networks: Layer-wise pruning based on feature representations," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 12, pp. 3048–3056, 2018.

[28] D. Peer, S. Stabinger, S. Engl, and A. Rodríguez-Sánchez, "Greedy-layer pruning: Speeding up transformer models for natural language processing," *Pattern Recognition Letters*, vol. 157, pp. 76–82, 2022.

[29] H. Sajjad, F. Dalvi, N. Durrani, and P. Nakov, "On the effect of dropping layers of pre-trained transformer models," *Computer Speech & Language*, vol. 77, p. 101429, 2023.

[30] M. Raghu, J. Gilmer, J. Yosinski, and J. Sohl-Dickstein, "Svcca: Singular vector canonical correlation analysis for deep learning dynamics and interpretability," *Advances in neural information processing systems*, vol. 30, 2017.

[31] F. R. Bach and M. I. Jordan, "A probabilistic interpretation of canonical correlation analysis," 2005.

[32] X. Zhen, Z. Meng, R. Chakraborty, and V. Singh, "On the versatile uses of partial distance correlation in deep learning," in *Computer Vision–ECCV: 17th European Conference*. Springer, 2022, pp. 327–346.

[33] G. J. Székely, M. L. Rizzo, and N. K. Bakirov, "Measuring and testing dependence by correlation of distances," 2007.

[34] K. Lu and K. Chen, "A context-aware knowledge transferring strategy for ctc-based ASR," in *IEEE Spoken Language Technology Workshop, SLT*, 2022, pp. 60–67.

[35] J. Du, X. Na, X. Liu, and H. Bu, "Aishell-2: Transforming mandarin asr research into industrial scale," *arXiv preprint arXiv:1808.10583*, 2018.

[36] A. Conneau, A. Baevski, R. Collobert, A. Mohamed, and M. Auli, "Unsupervised cross-lingual representation learning for speech recognition," in *Proc. of Interspeech*, 2021, pp. 2426–2430.

[37] H. Bu, J. Du, X. Na, B. Wu, and H. Zheng, "Aishell-1: An open-source mandarin speech corpus and a speech recognition baseline," in *Proc. of O-COCOSDA*. IEEE, 2017, pp. 1–5.

[38] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *7th International Conference on Learning Representations, ICLR*, 2019.

[39] L. Chen, Y. Chen, J. Xi, and X. Le, "Knowledge from the original network: restore a better pruned network with knowledge distillation," *Complex & Intelligent Systems*, pp. 1–10, 2021.

[40] K. Tirumala, A. H. Markosyan, L. Zettlemoyer, and A. Aghajanyan, "Memorization without overfitting: Analyzing the training dynamics of large language models," *arXiv preprint arXiv:2205.10770*, 2022.