Self-Distilled Pruning of Deep Neural Networks

James O' Neill ✉, Sourav Dutta, and Haytham Assem

Huawei Ireland Research Center, Dublin, Ireland
james.o.neil@huawei-partners.com
{sourav.dutta2,haytham.assem}@huawei.com

Abstract. Pruning aims to reduce the number of parameters while maintaining performance close to the original network. This work proposes a novel *self-distillation* based pruning strategy, whereby the representational similarity between the pruned and unpruned versions of the same network is maximized. Unlike previous approaches that treat distillation and pruning separately, we use distillation to inform the pruning criteria, without requiring a separate student network as in knowledge distillation. We show that the proposed *cross-correlation objective for self-distilled pruning* implicitly encourages sparse solutions, naturally complementing magnitude-based pruning criteria. Experiments on the GLUE and XGLUE benchmarks show that self-distilled pruning increases mono- and cross-lingual language model performance. Self-distilled pruned models also outperform smaller Transformers with an equal number of parameters and are competitive against (6 times) larger distilled networks. We also observe that self-distillation (1) maximizes class separability, (2) increases the signal-to-noise ratio, and (3) converges faster after pruning steps, providing further insights into why self-distilled pruning improves generalization.

Keywords: iterative pruning · self-distillation · language models.

1 Introduction

Neural network pruning [29,16,33] zeros out weights of a pretrained model with the aim of reducing parameter count and storage requirements, while maintaining performance close to the original model. The criteria to choose which weights to prune has been an active research area over the past three decades [16,19,10,3,28]. Lately, there has been a focus on pruning models in the transfer learning setting whereby a self-supervised pretrained model trained on a large amount of unlabelled data is fine-tuned to a downstream task while weights are simultaneously pruned, referred to as *fine-pruning*. In this context, recent work proposes to learn important scores over weights with a continuous mask and prune away those that having the smallest scores [25,36]. However, these learned continuous masks double the number of parameters and gradient updates in the network [36]. Ideally, we aim to perform task-dependent fine-pruning *without* adding more parameters to the network, or at least far fewer than twice the count. Additionally, we desire pruning methods that can recover from performance degradation directly after pruning steps, faster than current pruning methods while encoding task-dependent information into the pruning process. To this end, we hypothesize self-distillation may recover performance faster after consecutive pruning steps, which becomes more important with larger performance degradation at a higher compression regime. Additionally, self-distillation has shown to encourage sparsity as the training error tends to 0 [27]. This implicit sparse regularization effect complements magnitude-based pruning.

Hence, this paper proposes to combine self-distillation and magnitude-based pruning to achieve task-dependent pruning efficiently. This is achieved by *maximizing the cross-correlation* between output representations of the fine-tuned pretrained network and a pruned version of the same network – referred to as *self-distilled pruning* (SDP). Cross-correlation maximization reduces redundancy and encourages sparse solutions [49], naturally fitting with magnitude-based pruning. Unlike typical knowledge distillation (KD) where the student is a separate network trained from random initialization, here the student is initially a masked version of the teacher. We find that SDP sets state-of-the-art results when compared to alternative magnitude-based pruning methods and equivalently sized distilled networks. We also provide three insights as to why self-distillation leads to more generalizable pruned networks. We observe that self-distilled pruning (1) *recovers performance faster* after pruning steps (i.e., improves convergence), (2) *maximizes the signal-to-noise ratio* (SNR), where pruned weights are considered as noise, and (3) *improves the fidelity* between pruned and unpruned representations as measured by mutual information of the respective penultimate layers. We focus on pruning fine-tuned monolingual *and* cross-lingual transformer models, namely BERT [6] and XLM-RoBERTa [5]. To our knowledge, this is the first study that introduces the concept of *self-distilled pruning*, analyzes iterative pruning in the mono-lingual *and* cross-lingual settings on the GLUE and XGLUE benchmarks respectively and the only work to include an evaluation of pruned model performance in the cross-lingual transfer setting.

2 Background and Related Work

Regularization-based pruning can be achieved by using a weight regularizer that encourages network sparsity. Three well-established regularizers are L_1, L_2 and L_0 weight regularization [24,23,47] for weight sparsity [11,10]. For structured pruning, Group-wise Brain Damage [18] and SSL [45] propose to use Group LASSO [48] to prune whole structures (e.g., convolution blocks or blocks within standard linear layers).Park et al. [31] avoid pruning small weights if they are connected to larger weights in consecutive layers and vice-versa, by penalizing the Frobenius norm between pruned and unpruned layers to be small.

Importance-based pruning assigns a score for each weight in the network and removes weights with the lowest importance score. The simplest scoring criteria is magnitude-based pruning (MBP), which uses the lowest absolute value

(LAV) as the criteria [33,11,10] or L_1/L_2-norm for structured pruning [23]. MBP can be seen as a zero-th order pruning criteria. However higher order pruning methods approximate the difference in pruned and unpruned model loss using a Taylor series expansion up until 1^{st} order [19,12] or the 2^{nd} order, which requires approximating the Hessian matrix [26,44,37] for scalability. Lastly, the regularization-based pruning is commonly used with importance-based pruning e.g using L_2 weight regularization alongside MBP.

Knowledge Distillation (KD) transfers the knowledge of an already trained network, such as the logit outputs [13]), and uses them as soft targets to optimize a student network. The student network is typically smaller than the teacher network and benefits from the additional information soft targets provide. There has been various extensions that involve distilling intermediate representations [34], distributions [14], maximizing mutual information between student and teacher representations [1], using pairwise interactions for improved KD [32] and contrastive representation distillation [39,30].

Self-Distillation is a special case of KD whereby the student and teacher networks have the same capacity. Interestingly, self-distilled students often generalize better than the teacher [9,46], however the mechanisms by which self-distillation leads to improved generalization remain somewhat unclear. Recent works have provided insightful observations of this phenomena. For example, Stanton et al. [38] have shown that soft targets make optimization easier for the student when compared to the task-provided one-hot targets. Allen et al. [2] view self-distillation as implicitly combining ensemble learning and KD to explain the improvement in test accuracy when dealing with multi-view data. The core idea is that the self-distillation objective results in the network learning a unique set of features that are distinct from the original model, similar to features learned by combining the outputs of independent models in an ensemble. Given this background on pruning and distillation, we now describe our proposed methodology for *SDP*.

3 Proposed Methodology

We begin by defining a dataset $\mathcal{D} := \{(X_i, y_i)\}_{i=1}^{D}$ with single samples $s_i = (X_i, \boldsymbol{y}_i)$, where each X_i (in the D training samples) consists of a sequence of vectors $X_i := (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N)$ and $\boldsymbol{x}_i \in \mathbb{R}^d$. For structured prediction (e.g., NER, POS) $y_i \in \{0,1\}^{N \times C}$, and for single and pairwise sentence classification, $y_i \in \{0,1\}^C$, where C is the number of classes. Let $\boldsymbol{y}^S = f_\theta(X_i)$ be the output prediction ($y^S \in \mathbb{R}^C$) from the student $f_\theta(\cdot)$ with pretrained parameters $\theta := \{\mathbf{W}_l, \boldsymbol{b}_l\}_{l=1}^{L}$ for L layers. The intermediate input to each subsequent layer is denoted as $\boldsymbol{z}_l \in \mathbb{R}^{n_l}$ where $\boldsymbol{z}_0 := \boldsymbol{x}$ for n_l number of units in layer l and the corresponding output activation as $\boldsymbol{A}_l = g(\boldsymbol{z}_l)$. The loss function for standard classification fine-tuning is defined as the cross-entropy loss $\ell_{CE}(\boldsymbol{y}^S, \boldsymbol{y}) := -\frac{1}{C}\sum_{i=1}^{c} \boldsymbol{y}_c \log(\boldsymbol{y}_c^s)$.

For self-distilled pruning, we also require an already fine-tuned teacher network f_Θ, that has been tuned from the pretrained state f_θ, to retrieve the soft teacher labels $y^T := f_\Theta(\boldsymbol{x})$, where $y^T \in \mathbb{R}^C$ and $\sum_c^C y_c^T = 1$. The soft label $\boldsymbol{y}^T$ can be more informative than the one-hot targets $\boldsymbol{y}$ used for standard classification as they implicitly approximate pairwise class similarities through logit probabilities. The Kullbeck-Leibler divergence ℓ_{KLD} is then used with the main task cross-entropy loss ℓ_{CE} to express $\ell_{\text{SDP}-\text{KLD}}$ as shown in Equation 1,

$$\ell_{\text{SDP-KLD}} = (1\text{-}\alpha)\ell_{\text{CE}}(\boldsymbol{y}^S, \boldsymbol{y}) + \alpha\tau^2 D_{\text{KLD}}\big(\boldsymbol{y}^S, \boldsymbol{y}^T\big) \tag{1}$$

where $D_{\text{KLD}}(\boldsymbol{y}^S, \boldsymbol{y}^T) = \mathbb{H}(\boldsymbol{y}^T) - \boldsymbol{y}^T \log(\boldsymbol{y}^S)$, $\mathbb{H}(\boldsymbol{y}^T) = \boldsymbol{y}^T \log(\boldsymbol{y}^T)$ is the entropy of the teacher distribution and τ is the softmax temperature. Following [13], the weighted sum of cross-entropy loss and KLD loss shown in Equation 1 is used as our main SDP-based KD loss baseline, where $\alpha \in [0, 1]$. After each pruning step during iterative pruning, we aim to recover the immediate performance degradation by minimizing $\ell_{\text{SDP}-\text{KLD}}$. In our experiments, we use weight magnitude-based pruning as the criteria for SDP given MBP's flexibility, scalability and miniscule computation overhead (only requires a binary tensor multiplication to be applied for each linear layer at each pruning step). However, D_{KLD} only distils the knowledge from the soft targets which may not propagate enough information about the intermediate dynamics of the teacher, nor does it penalize representational redundancy. This brings us to our proposed SDP objective.

3.1 Cross-Correlation Between Pruned and Unpruned Embeddings

Iterative pruning can be viewed as progressively adding noise $\mathbf{M}_l \in \{0, 1\}^{n_{l-1}\times n_l}$ to the weights $\mathbf{W}_l \in \mathbb{R}^{n_{l-1}\times n_l}$. Thus, as the pruning steps increase, the outputs become noisier and the relationship between the inputs and outputs becomes weaker. Hence, a correlation measure is a natural choice for dealing with such pruning-induced noise. To this end, we use a cross-correlation loss to maximize the correlation between the output representations of the last hidden state of the pruned network and the unpruned network to reduce the effects of this pruning noise. The proposed *cross-correlation SDP loss function*, ℓ_{CC}, is expressed in Equation 2, where λ controls the importance of minimizing the non-adjacent pairwise correlations between z^S and z^T in the correlation matrix $\mathcal{C}$. Here, m denotes the sample index in a mini-batch of M samples. Unlike ℓ_{KLD}, this loss is applied to the outputs of the last hidden layer as opposed to the classification logit outputs. Thus, we have,

$$\ell_{\text{CC}}(\boldsymbol{z}^S, \boldsymbol{z}^T) := \sum_i (1 - \mathcal{C}_{ii})^2 + \lambda \sum_i \sum_{j\neq i} \mathcal{C}_{ij}^2 \tag{2}$$

such that $\mathcal{C}_{ij} := \frac{\sum_m \boldsymbol{z}^S_{m,i}\boldsymbol{z}^T_{m,j}}{\sqrt{\sum_m(\boldsymbol{z}^S_{m,i})^2}\sqrt{\sum_m(\boldsymbol{z}^T_{m,j})^2}}$.

Maximizing correlation along the diagonal of $\mathcal{C}$ makes the representations invariant to pruning noise, while minimizing the off-diagonal term decorrelates the components of the representations that are batch normalized. To reiterate, $\boldsymbol{z}^S$ is obtained from the pruned version of the network (f_{θ_p}) and $\boldsymbol{z}^T$ is obtained from the unpruned version (f_θ).

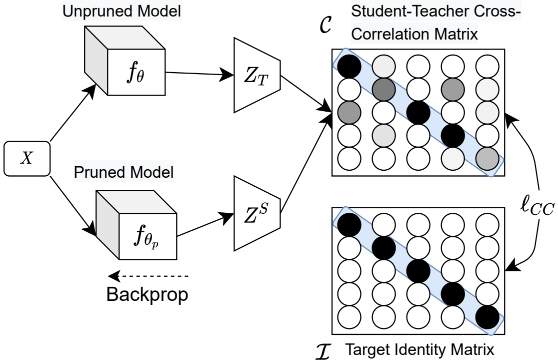


Fig. 1: **Self-Distilled Pruning with a Cross-Correlation Knowledge Distillation Loss.**

Since the learned output representations should be similar if their inputs are similar, we aim to address the problem where a correlation measure may produce representations that are instead *proportional* to their inputs. To address this, batch normalization is used across mini-batches to stabilize the optimization when using the cross-correlation loss, avoiding local optima that correspond to degenerate representations that do not distinguish proportionality. In our experiments, this is used with the classification loss and KLD distillation loss as shown in Equation 3.

$$\ell_{\text{SDP}-\text{CC}} = (1-\alpha)\ell_{\text{CE}}(\boldsymbol{y}^S, \boldsymbol{y}) + \alpha\tau^2 D_{\text{KLD}}(\boldsymbol{y}^S, \boldsymbol{y}^T) + \beta\ell_{\text{CC}}(\boldsymbol{z}^S, \boldsymbol{z}^T) \tag{3}$$

Figure 1 illustrates the proposed framework of *self-distilled pruning with cross-correlation loss* (SDP-CC), where $\mathcal{I}$ is the identity matrix. Additionally, we provide a PyTorch based pseudo-code for SDP-CC the supplementary material.

3.2 A Frobenius Distortion Perspective of Self-Distilled Pruning

To formalize the objective being minimized when using MBP with self-distillation, we take the view of *Frobenius distortion minimization* (FDM) [7] which says that layer-wise MBP is equivalent to minimizing the Frobenius distortions of a single layer. This can be described as $\min_{\mathbf{M}:||\mathbf{M}||_0=p} ||\mathbf{W} - \mathbf{M} \odot \mathbf{W}||_F$, where $\odot$ is the Hadamard product and p is a constraint of the number of weights to remove as a percentage of the total number of weights for a layer. Therefore, the output distortion is approximately the product of single layer Frobenius distortions. However, this minimization only defines a 1^{st} order approximation of pruning induced Frobenius distortions which is a loose approximation for deep networks. In contrast, the $\boldsymbol{y}^T$ targets provide higher-order information outside of the l-th layer being pruned in this FDM framework because Θ encodes information of all neighboring layers. Hence, we reformulate the FDM problem for SDP as an approximately higher-order MBP method as in Equation 4 where $\mathbf{W}^T$ are the weights in f_Θ.

$$\min_{\mathbf{M}:||\mathbf{M}||_0=p} \Big[||\mathbf{W}\text{-}\mathbf{M} \odot \mathbf{W}||_F + \lambda||\mathbf{W}^T - \mathbf{M} \odot \mathbf{W}||_F\Big] \tag{4}$$

As described in [7,12], the difference in error can be approximated with a Taylor Series (TS) expansion as $\delta\mathcal{E}_l \approx \big(\frac{\partial \mathcal{E}_l}{\partial \mathbf{W}^l}\big)^\top \delta\mathbf{W}_l + \frac{1}{2}\delta\mathbf{W}_l^\top \mathbf{H}_l \delta\mathbf{W}_l + O(||\delta\mathbf{W}_l||^3)$ where $\mathbf{H}$ is the Hessian matrix. When using SDP with a 1^{st} TS, we can further express the TS approximation for SDP as shown in Equation 5, where $\mathcal{E}_l^S$ is the error of the pruned network for task provided targets and $\mathcal{E}_l^T$ are the errors of the pruned network with distilled logits.

$$\big(\mathcal{E}_l - \mathcal{E}_l^S\big)^2 + \lambda\big(\mathcal{E}_l - \mathcal{E}_l^T\big)^2 \approx \delta\mathcal{E}_l^S + \delta\mathcal{E}_l^T \approx \Big(\frac{\partial \mathcal{E}_l^S}{\partial \theta_l}\Big)^\top \delta\theta_l + \lambda\Big(\frac{\partial \mathcal{E}_l^T}{\partial \theta_l}\Big)^\top \delta\theta_l \tag{5}$$

3.3 How Does Self-Distillation Improve Pruned Model Generalization ?

We put forth the following insights as to the advantages provided by self-distillation for better pruned model generalization, and later experimentally demonstrate their validity.

Recovering Faster From Performance Degradation After Pruning Steps. The first explanation for why self-distillation leads to better generalization in iterative pruning is that the soft targets bias the optimization and smoothen the loss surface through implicit similarities between the classes encoded in the logits. We posit this too holds true for performance recovery after pruning steps, as the classification boundaries become distorted due to the removal of weights. Faster convergence is particularly important for high compression rates where the performance drops become larger.

Implicit Maximization of the Signal-to-Noise Ratio. One explanation for faster convergence is that optimizing for soft targets translates to maximizing the margin of class boundaries given the implicit class similarities provided by teacher logits. Intuitively, task provided one-hot targets do not inform SGD of how similar incorrect predictions are to the correct class, whereas the teacher logits do, to the extent they have learned on the same task. To measure this, we use a formulation of the signal-to-noise ratio[1] (SNR) to measure the class separability and compactness differences between pruned model representations trained with and without self-distillation. We formulate SNR as Equation 6, where for a batch of inputs $\mathbf{X}$, we obtain $\mathbf{Z}$ output representations from the pruned network, which contain samples with C classes where each class has the same N number of

[1] A measure typically used in signal processing to evaluate signal quality.

samples. The numerator expresses the average ℓ_2 inter-class distance between instances of each class pair and the denominator expresses the intra-class distance between instances within the same class.

$$\frac{1/N(C\text{-}1)^2 \sum_n^N \sum_{c=1}^C \sum_{i\neq c}^C ||\sqrt{\mathbf{Z}_{c,n}}\text{-}\sqrt{\mathbf{Z}_{i,n}}||_2}{1/C(N\text{-}1)^2 \sum_{c=1}^C \sum_n^N \sum_{j\neq n} ||\sqrt{\mathbf{Z}_{c,n}}\text{-}\sqrt{\mathbf{Z}_{c,j}}||_2} \tag{6}$$

This estimation is $C-1\binom{C+1}{2}$ in the number of pairwise distances to be computed between the inter-class distances for the classes. For large output spaces (e.g., language modeling) we recommend defining the top k-NN classes for each class and estimate their distances on samples from them.

Quantifying Fidelity Between Pruned Models Trained With and Without Self-Distillation. A natural question to ask is *how much generalization power does the distilled soft targets provide when compared to the task provided one-hot targets ?* If best generalization is achieved when $\alpha = 1$ in Equation 1, this implies that the pruned network should have as high fidelity as possible with the unpruned network. However, as we will see there is a bias-variance trade-off between fidelity and generalization performance, i.e., $\alpha = 1$ is not optimal in most cases. To measure fidelity between SDP representations and standard fine-tuned representations, we compute their *mutual information* (MI) and compare this to the MI between representations of pruned models without self-distillation and standard fine-tuned models. The MI between continuous variables can be expressed as,

$$\begin{gathered}\hat{I}(\mathbf{Z}^T;\mathbf{Z}^S) = \mathrm{H}(\mathbf{Z}^T) - \mathrm{H}(\mathbf{Z}^T|\mathbf{Z}^S) = \\ -\mathbb{E}_{\boldsymbol{z}^T}[\log p(\mathbf{Z}^T)] + \mathbb{E}_{\mathbf{Z}^T,\mathbf{Z}^S}[\log p(\mathbf{Z}^T|\mathbf{Z}^S)]\end{gathered} \tag{7}$$

where $\mathrm{H}(\mathbf{Z}^T)$ is the the entropy of the teacher representation and $\mathrm{H}(\mathbf{Z}^T|\mathbf{Z}^S)$ is the conditional entropy that is derived from the joint distribution $p(\mathbf{Z}^T,\mathbf{Z}^S)$. This can also be expressed as the KL divergence between the joint probabilities and product of marginals as $I(Z^T;Z^S) = D_{\mathrm{KLD}}[p(Z^S,Z^T)||p(Z^S)p(Z^T)]$. However, these theoretical quantities have to be estimated from test sample representations. We use a k-NN based MI estimator [17,8,42,41] which partitions the supports into a finite number of bins of equal size, forming a histogram that can be used to estimate $\hat{I}(Z^S;Z^T)$ based on discrete counts in each bin. This MI estimator is given as,

$$I(z^S;z^T) \approx \epsilon\Big(\log \frac{\phi_{[\boldsymbol{z}^S]}(i,k_{[\boldsymbol{z}^S]})\phi_{[\boldsymbol{z}^T]}(i,k_{[\boldsymbol{z}^T]})}{\phi_z(i,k)}\Big) \tag{8}$$

where $\phi_{z^S}(i,k_{[z^S]})$ is the probability measure of the k-th nearest neighbour ball of $\boldsymbol{z}^S \in \mathbb{R}^{n_L}$ and $\omega_{[\boldsymbol{z}^T]}(i,k_{[\boldsymbol{z}^T]})$ is the probability measure of the k_y-th nearest neighbour ball of $\boldsymbol{z}^T \in \mathbb{R}^{n_L}$ where n_L is the dimension of the penultimate layer. In our experiments, we use 256 bins for the histogram with Gaussian smoothing and $k = 5$ (see [17] for further details).

Table 1: **GLUE benchmark results for pruned models @10% (or @20%) remaining weights.**

Compression Method	**Score**	**Single Sentence**		**Similarity and Paraphrase**			**Natural Language Inference**		
	(avg.)	**CoLA**	**SST-2**	**MNLI**	**MRPC**	**STS-B**	**QQP**	**RTE**	**QNLI**
		(mcc)	(acc)	(acc)	(f1/acc)	(pears./spear.)	(f1/acc)	(acc)	(acc)
BERT$_{\text{Base}}$ (Ours)	84.06	53.24	90.71	80.27	80.9/77.7	83.5/83.8	83.9/88.0	68.59	86.91
Knowledge Distilled Baselines (% parameters w.r.t. original BERT)									
DistilBERT (60%)	82.85	51.3	91.3	82.2	87.5/-.-	86.9/-.-	-.-/85.5	59.9	89.2
BERT-Medium (44.4%)	81.54	38.0	89.6	80.0	86.6/81.6	80.4/78.4	69.6/87.9	62.2	87.7
BERT-Small (20%)	79.02	27.8	89.7	77.6	83.4/76.2	78.8/77.0	68.1/87.0	61.8	86.4
BERT-Mini (10%)	76.97	0.0	85.9	75.1	74.8/74.3	75.4/73.3	66.4/86.2	57.9	84.1
BERT-Tiny (3.6%)	73.32	0.0	83.2	70.2	81.1/71.1	74.3/73.6	62.2/83.4	57.2	81.5
Pruning Baselines		20%	10%	10%	10%	10%	10%	10%	10%
Random	66.03	6.50	78.44	69.55	77.5/67.1	27.4/26.9	77.07/81.86	52.70	74.66
L_0-MBP	77.25	31.68	83.37	75.61	78.4/68.2	75.9/75.7	81.56/86.49	**64.26**	82.62
L_2-MBP	76.48	29.51	83.37	76.19	78.4/68.2	75.3/75.6	77.50/82.98	62.09	82.61
L_2-Global-MBP	77.16	29.25	82.83	76.40	81.2/69.9	75.1/75.5	82.77/86.70	62.01	82.24
L_2-Gradient-MBP	74.84	15.46	82.91	72.51	81.0/73.7	73.8/73.6	80.41/85.19	56.31	79.33
1^{st}-order Taylor	76.31	28.88	83.26	74.64	**83.0/74.8**	76.7/76.6	80.09/85.29	57.76	81.20
Lookahead	76.40	28.15	82.80	75.31	79.8/70.5	71.9/71.9	81.84/86.53	60.29	81.80
LAMP	74.03	20.31	83.26	74.27	72.3/63.7	73.7/74.1	79.32/85.07	58.84	81.09
Proposed Methodology									
L_2-MBP + SDP-COS	77.83	31.80	86.00	75.68	81.6/72.2	76.4/76.3	81.39/86.68	61.73	83.07
L_2-MBP + SDP-KLD	78.34	36.74	**87.96**	77.94	80.5/68.2	77.1/77.3	83.21/85.58	63.18	83.54
L_2-MBP + SDP-CC	**78.90**	**36.77**	87.84	**78.04**	81.1/71.0	**77.3/77.5**	**83.79/86.37**	62.64	**84.20**

BERT- results reported from prior work [35,15,40] and MNLI results are for the matched dataset.

4 Experimental Setup

Datasets. We perform experiments on monolingual tasks within the GLUE [43] benchmark[2] with pretrained BERT$_{\text{Base}}$ and multilingual tasks from the XGLUE benchmark [22] with pretrained XLMR$_{\text{Base}}$. In total, this covers 18 different datasets, covering pairwise classification, sentence classification, structured prediction and question answering. To our knowledge, this work is the first to analyse iterative pruning in the context of cross-lingual models and their application on multilingual datasets. Further dataset statistics can be found in supplementary material.

Iterative Pruning Baselines. For XGLUE tasks, we perform 15 pruning steps on XLM-RoBERTA$_{\text{Base}}$, one per 15 epochs, while for the GLUE tasks, we perform

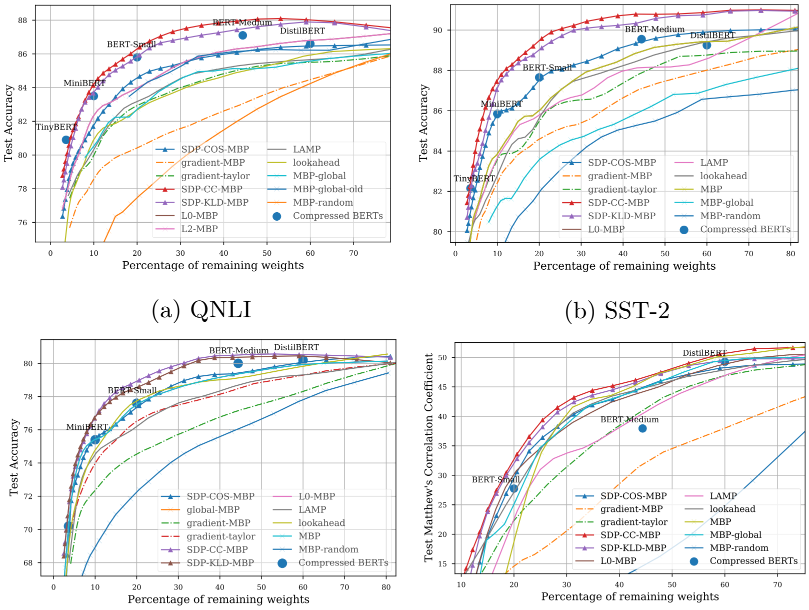
[2] WNLI is excluded for known issues, see the Q. 12 on the GLUE benchmark FAQ.

32 pruning steps on BERT$_{\mathrm{Base}}$. The compression rate and number of pruning steps is higher for GLUE tasks compared to XGLUE, because GLUE tasks involve evaluation in the *supervised classification* setting; whereas in XGLUE we report in the more challenging *zero-shot cross-lingual transfer* setting with only a single language used for training (i.e., English). At each pruning step, we uniformly pruning 10% of the parameters for both the models. Although prior work suggests non-uniform pruning schedules (e.g., cubic schedule [50]), we did not see any major differences to uniform pruning.We compare the performance of the proposed SDP-CC method against the following baselines:

- **Random Pruning** (*MBP-Random*) - prunes weights uniformly at random across all layers. Random pruning can be considered as a lower bound on iterative pruning performance.
- **Layer-wise Magnitude Based Pruning** (*MBP*) - for each layer, prunes weights with the LAV.
- **Global Magnitude Pruning** (*Global-MBP*) - prunes the LAV of all weights in the network.
- **Layer-wise Gradient Magnitude Pruning** (*Gradient-MBP*) - for each layer, prunes the weights with the LAV of the accumulated gradients evaluated on a batch of inputs.
- 1^{st} **Taylor Series Pruning** (*TS*) - prunes weights based on the LAV of |gradient × weight|.
- L_0 **norm MBP** [24] - uses non-negative stochastic gates that choose which weights are set to zero as a smooth approximation to the non-differentiable L_0-norm.
- L_1 **norm MBP** [21] - applies L_1 weight regularization and uses MBP.
- **Lookahead pruning (LAP)** [31] - prunes weight paths that have the smallest magnitude across blocks of layers, unlike MBP that does not consider neighboring layers.
- **Layer-Adaptive MBP (LAMP)** [20] - adaptively computes the pruning ratio for each layer.

For all above pruning methods we exclude weight pruning of the embeddings, layer normalization parameters and the last classification layer, as they play an important role for generalization and account for less than 1% of weights in both BERT and XLM-R$_{\mathrm{Base}}$.

Knowledge Distillation We also compare against a class of smaller knowledge distilled versions of the BERT model with varying parameter sizes on the GLUE benchmark. We report prior results of *DistilBERT* [35] and also mini-BERT models including *TinyBERT* [15], *BERT-small* [40] and *BERT-medium* [40]. In addition, we consider maximizing the cosine similarity between pruned and unpruned representations in the SDP loss, as $\ell_{\mathrm{SDP-COS}} := \alpha\ell_{\mathrm{CE}}(\boldsymbol{y}^S, \boldsymbol{y}) + \beta\big(1 - \frac{\mathbf{z}^S \cdot \boldsymbol{z}^T}{||\boldsymbol{z}^S||||\boldsymbol{z}^T||}\big)$. Unlike cross-correlation, there is no decorrelation of non-adjacent features in both representations for SDP-COS. This helps identify whether the redundancy reduction in cross-correlation is beneficial compared to the correlation loss that does not directly optimize this.



(a) QNLI

(b) SST-2

(c) MNLI

(d) CoLA

Fig. 2: **Iterative Pruning Results on GLUE tasks.**

5 Empirical Results

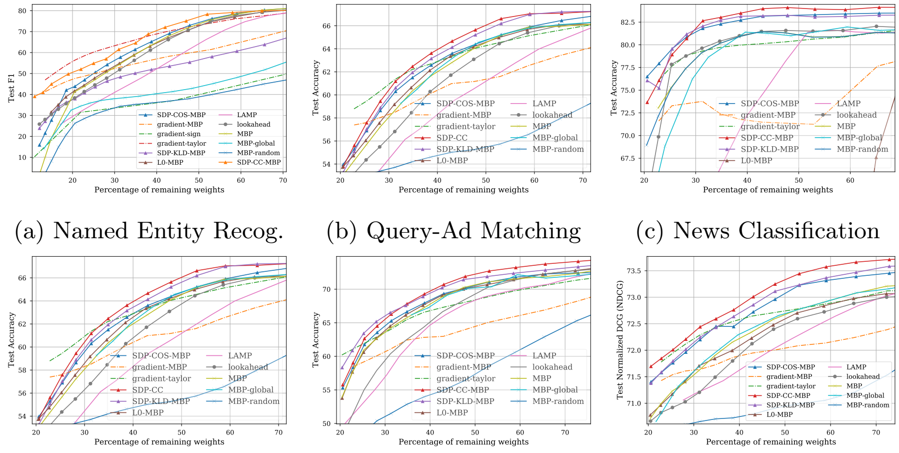
Pruning Results on GLUE. Table 1 shows the test performance across all GLUE tasks of the different models with varying pruning ratios, up to *10% remaining weights* of original $\text{BERT}_{\text{Base}}$ along with mini-BERT models [35,40] of varying size. However, for the CoLA dataset, we report at 20% pruning as nearly all compression methods have an MCC score of 0, making the compressed method performance indistinguishable. For this reason, the GLUE score (**Score**) is computed for all tasks and methods @10% apart from CoLA. The best performing compression method per task is marked in **bold**. We find that our proposed SDP approaches (all three variants) outperform against baseline pruning methods, with *SDP-CC* performing the best across all tasks. We note that for the tasks with fewer training samples (e.g., CoLA has 8.5k samples, STS-B has 7k samples and RTE has 3k samples), the performance gap is larger compared to $\text{BERT}_{\text{Base}}$, as the pruning step interval is shorter and less training data allows lesser time for the model to recover from pruning losses and also less data for teacher model to distil in the case of using SDP.

Smaller dense versions of BERT require more labelled data in order to compete with unstructured MBP and higher-order pruning methods such as 1^{st} order Taylor series and Lookahead pruning. For example, we see BERT-Mini (@10%) shows competitive test accuracy with our proposed SDP-CC on QNLI, MNLI and QQP, the three datasets with the most training samples (105k, 393k and

Table 2: **XGLUE Iterative Pruning @ 30% Remaining Weights of XLM-R$_{\text{base}}$** - Zero Shot Cross-Lingual Performance Per Task and Overall Average Score (Avg).

Prune Method	**XNLI**	**NC**	**NER**	**PAWSX**	**POS**	**QAM**	**QADSM**	**WPR**	**Avg.**
XLM-R$_{\text{Base}}$	73.48	80.10	82.60	89.24	80.34	68.56	68.06	73.32	76.96
Random	51.22	70.19	38.19	57.37	52.57	53.85	52.34	70.69	55.80
Global-Random	50.97	69.88	38.30	56.74	53.02	54.02	53.49	69.11	55.69
L_0-MBP	64.75	78.98	56.22	72.09	71.38	59.31	53.35	71.70	65.97
L_2-MBP	64.30	78.79	54.43	77.99	70.68	59.24	60.33	71.52	67.16
L_2-Global-MBP	65.12	78.64	54.47	79.13	71.37	59.26	60.61	71.80	67.55
L_2-Gradient-MBP	61.11	73.77	53.25	79.56	65.89	57.35	59.33	71.59	65.23
1^{st}-order Taylor	64.26	79.34	63.60	**82.83**	68.94	61.69	62.42	72.28	69.09
Lookahead	60.84	79.18	54.44	71.05	68.76	55.94	53.41	71.26	64.36
LAMP	58.04	63.64	51.92	66.05	67.43	55.36	52.42	71.09	60.74
L_2-MBP + SDP-COS	64.96	79.02	62.77	78.70	72.88	60.21	60.94	72.04	68.94
L_2-MBP + SDP-KLD	65.94	**80.72**	64.50	79.25	73.18	61.66	61.09	71.84	**69.77**
L_2-MBP + SDP-CC	**66.47**	79.73	**66.34**	80.03	**73.45**	**63.73**	**62.78**	**72.59**	**70.76**

364k respectively). Overall, L_2-MBP + SDP-CC achieves the highest GLUE score for all models at 10% remaining weights when compared to BERT-Base parameter count. Moreover, we find that L_2-MBP + SDP-CC achieves best performance for 5 of the 8 tasks, with 1 of the remaining 3 being from L_2MBP+SDP-KLD. This suggests that redundancy reduction via a cross-correlation objective is useful for SDP and clearly improve over SDP-COS which does not minimize correlations between off-diagonal terms. Figure 2 shows the performance across all pruning steps. Interestingly, for QNLI we observe the performance notably improves between 30-70% for SDP-CC and SDP-KLD. For SST-2, we observe a significant gap between SDP-KLD and SDP-CC compared to the pruning baselines and smaller versions of BERT, while TinyBERT becomes competitive at extreme compression (<4%). **Pruning Results on *XGLUE*.** We show the per task test performance and the *average task understanding* score on XGLUE for pruning baselines and our proposed SDP approaches in Table 2. Our proposed cross-correlation objective for SDP again achieves the best average (Avg.) score and achieves the best task performance in 6 out of 8 tasks, while standard SDP-KLD achieves best performance on one (news classification) of the remaining two. Most notably, we outperform methods which use higher order gradient information (1^{st}-order Taylor) at 30% remaining weights, which tends to be a point at which XLM-R$_{\text{Base}}$ begins to degrade performance below 10% of the original fine-tuned test performance for SDP methods and competitive baselines. In Figure 3, we can observe this trend from the various tasks within XGLUE. We note that the number of training samples used for retraining plays an important role in the rate of performance degradation. For example, of the 6 presented XGLUE



(a) Named Entity Recog. (b) Query-Ad Matching (c) News Classification

(d) Question Ans. Matching (e) XNLI (f) Web-Page Ranking

Fig. 3: **Zero-Shot Results After Iteratively Fine-Pruning XLM-R$_{\text{Base}}$ on XGLUE tasks**.

tasks, NER has the lowest number of training samples (15k) of all XGLUE tasks and also degrades the fastest in performance (from 90% to 50% Test F1 at 30% remaining weights). In comparison, XNLI has the most training samples for retraining (433k) and maintains performance relatively well, keeping within 10% of the original fine-tuned model at 30% remaining weights. **Summary of Results.** From our experiments on GLUE and XGLUE task, we find that SDP consistently outperforms pruning, KD and smaller BERT baselines. SDP-KLD and SDP-CC both outperform larger sized BERT models (BERT-Small), somewhat surprisingly, given that BERT-Small (and the remaining BERT models) have the advantage of large-scale self-supervised pretraining, while pruning only has supervision from the downstream task. For NER in XGLUE, higher order pruning methods such as Taylor-Series pruning have an advantage at high compression rates mainly due to lack of training samples (only 15k). Apart from this low training sample regime, SDP with MBP dominates at high compression rates.

Measuring Fidelity To The Fine-Tuned Model. We now analyse the empirical evidence that soft targets used in SDP may force higher fidelity with the representations of the fine-tuned model when compared to using MBP without self-distillation. As described in subsection 3.3 we measure mutual dependencies between both representations of models with the best performing hyperparameter settings of α, β and the softmax temperature τ. We note that increasing the temperature τ translates to "peakier" teacher logit distributions, encouraging SGD to learn a student with high fidelity to the teacher. From the LHS of Figure 4, we can see that SDP models have higher mutual information (MI) with the teacher compared to MBP, which performs worse for PAWS-X (similar on remaining tasks, not shown for brevity). In fact, the rank order of

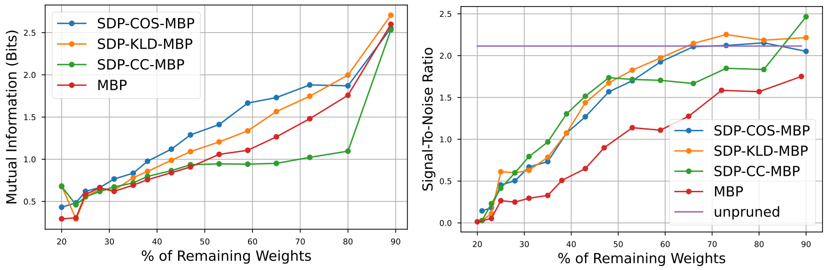


Fig. 4: **Mutual Information Between Unpruned and Pruned Representations (left) and Signal-To-Noise Ratio (right)**

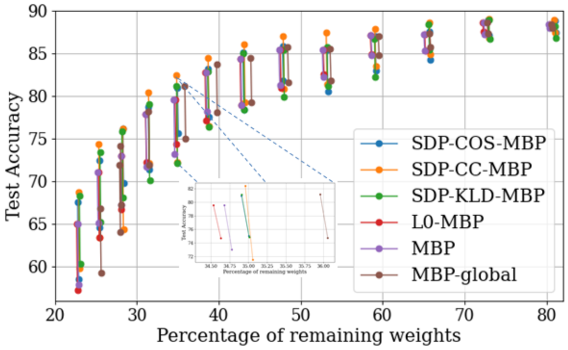


Fig. 5: PAWS-X Development Set Representations and (right) Pruning Performance Recovery with Self-Distilled Pruning.

the best performing pruned models at each pruning step has a direct correlation with MI, e.g., SDP-COS-MBP maintains highest MI and the highest test accuracy for PAWS-X for the same α. However, too high fidelity ($\alpha = 1.$) led to worse generalization compared to a balance between the task provided targets and the teacher logits ($\alpha = 0.5$).

Self-Distilled Pruning Increases Class Separability and The Signal-To-Noise Ratio (SNR). We also find that the SNR is increased at each pruning step as formulated in section 3.3. From this observation, we find that *SDP-CC-MBP* using cross-correlation loss does particularly well in the 30%-50% remaining weights range. More generally, all 3 SDP losses clearly lead to better class separability and class compactness across all pruning steps compared to MBP (i.e., no self-distillation).

Self-Distilled Pruning Recovers Faster Performance Degradation Directly After Pruning Steps. In Figure 5 we show how SDP with Magnitude pruning (SDP-MBP) recovers during training in between pruning steps. The top of each vertical bar is the recovery development accuracy and the bottom is the initial performance degradation prior to retrainng. We see that SDP pruned models degrade in performance more than magnitude pruning without self-distillation. This suggests that SDP-MBP may force weights to be closer, as there is more initial performance degradation if weights are not driven to zero. However, the recovery is faster. This may be explained by recent work that suggests the stability generalization tradeoff [4].

6 Conclusion

In this paper, we proposed a novel *self-distillation* based pruning technique based on a *cross-correlation* objective. We extensively studied the confluence between pruning and self-distillation for masked language models and its enhanced utility on downstream tasks in both monolingual and multi-lingual settings. We find that self-distillation aids in recovering directly after pruning in iterative magnitude-based pruning, increases representational fidelity with the unpruned model and implicitly maximize the signal-to-noise ratio. Additionally, we find our cross-correlation based self-distillation pruning objective minimizes neuronal redundancy and achieves state-of-the-art in magnitude-based pruning baselines, and even outperforms KD based smaller BERT models with more parameters.

References

1. Ahn, S., Hu, S.X., Damianou, A., Lawrence, N.D., Dai, Z.: Variational information distillation for knowledge transfer. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9163–9171 (2019)
2. Allen-Zhu, Z., Li, Y.: Towards understanding ensemble, knowledge distillation and self-distillation in deep learning. arXiv preprint arXiv:2012.09816 (2020)
3. Anwar, S., Hwang, K., Sung, W.: Structured pruning of deep convolutional neural networks. ACM Journal on Emerging Technologies in Computing Systems (JETC) **13**(3), 1–18 (2017)
4. Bartoldson, B.R., Morcos, A.S., Barbu, A., Erlebacher, G.: The generalization-stability tradeoff in neural network pruning. arXiv preprint arXiv:1906.03728 (2019)
5. Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., Stoyanov, V.: Unsupervised cross-lingual representation learning at scale. arXiv preprint arXiv:1911.02116 (2019)
6. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
7. Dong, X., Chen, S., Pan, S.J.: Learning to prune deep neural networks via layer-wise optimal brain surgeon. arXiv preprint arXiv:1705.07565 (2017)
8. Evans, D.: A computationally efficient estimator for mutual information. Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences **464**(2093), 1203–1215 (2008)
9. Furlanello, T., Lipton, Z., Tschannen, M., Itti, L., Anandkumar, A.: Born again neural networks. In: International Conference on Machine Learning. pp. 1607–1616. PMLR (2018)

10. Han, S., Mao, H., Dally, W.: Compressing deep neural networks with pruning, trained quantization and huffman coding. arXiv preprint (2015)
11. Han, S., Pool, J., Tran, J., Dally, W.J.: Learning both weights and connections for efficient neural networks. arXiv preprint arXiv:1506.02626 (2015)
12. Hassibi, B., Stork, D.G.: Second order derivatives for network pruning: Optimal brain surgeon. Morgan Kaufmann (1993)
13. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531 (2015)
14. Huang, Z., Wang, N.: Like what you like: Knowledge distill via neuron selectivity transfer. arXiv preprint arXiv:1707.01219 (2017)
15. Jiao, X., Yin, Y., Shang, L., Jiang, X., Chen, X., Li, L., Wang, F., Liu, Q.: Tinybert: Distilling bert for natural language understanding. arXiv preprint arXiv:1909.10351 (2019)
16. Karnin, E.D.: A simple procedure for pruning back-propagation trained neural networks. IEEE transactions on neural networks **1**(2), 239–242 (1990)
17. Kraskov, A., Stögbauer, H., Grassberger, P.: Estimating mutual information. Physical review E **69**(6), 066138 (2004)
18. Lebedev, V., Lempitsky, V.: Fast convnets using group-wise brain damage. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2554–2564 (2016)
19. LeCun, Y., Denker, J.S., Solla, S.A.: Optimal brain damage. In: Advances in neural information processing systems. pp. 598–605 (1990)
20. Lee, J., Park, S., Mo, S., Ahn, S., Shin, J.: Layer-adaptive sparsity for the magnitude-based pruning. In: International Conference on Learning Representations (2020)
21. Li, H., Kadav, A., Durdanovic, I., Samet, H., Graf, H.P.: Pruning filters for efficient convnets. arXiv preprint arXiv:1608.08710 (2016)
22. Liang, Y., Duan, N., Gong, Y., Wu, N., Guo, F., Qi, W., Gong, M., Shou, L., Jiang, D., Cao, G., et al.: Xglue: A new benchmark datasetfor cross-lingual pretraining, understanding and generation. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 6008–6018 (2020)
23. Liu, Z., Li, J., Shen, Z., Huang, G., Yan, S., Zhang, C.: Learning efficient convolutional networks through network slimming. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2736–2744 (2017)
24. Louizos, C., Welling, M., Kingma, D.P.: Learning sparse neural networks through l_0 regularization. arXiv preprint arXiv:1712.01312 (2017)
25. Mallya, A., Davis, D., Lazebnik, S.: Piggyback: Adapting a single network to multiple tasks by learning to mask weights. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 67–82 (2018)
26. Martens, J., Grosse, R.: Optimizing neural networks with kronecker-factored approximate curvature. In: International conference on machine learning. pp. 2408–2417. PMLR (2015)
27. Mobahi, H., Farajtabar, M., Bartlett, P.L.: Self-distillation amplifies regularization in hilbert space. arXiv preprint arXiv:2002.05715 (2020)
28. Molchanov, D., Ashukha, A., Vetrov, D.: Variational dropout sparsifies deep neural networks. In: International Conference on Machine Learning. pp. 2498–2507. PMLR (2017)
29. Mozer, M.C., Smolensky, P.: Skeletonization: A technique for trimming the fat from a network via relevance assessment. In: Advances in neural information processing systems. pp. 107–115 (1989)

30. Neill, J.O., Bollegala, D.: Semantically-conditioned negative samples for efficient contrastive learning. arXiv preprint arXiv:2102.06603 (2021)
31. Park, S., Lee, J., Mo, S., Shin, J.: Lookahead: A far-sighted alternative of magnitude-based pruning. arXiv preprint arXiv:2002.04809 (2020)
32. Park, W., Kim, D., Lu, Y., Cho, M.: Relational knowledge distillation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3967–3976 (2019)
33. Reed, R.: Pruning algorithms-a survey. IEEE transactions on Neural Networks **4**(5), 740–747 (1993)
34. Romero, A., Ballas, N., Kahou, S.E., Chassang, A., Gatta, C., Bengio, Y.: Fitnets: Hints for thin deep nets. arXiv preprint arXiv:1412.6550 (2014)
35. Sanh, V., Debut, L., Chaumond, J., Wolf, T.: Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. arXiv preprint arXiv:1910.01108 (2019)
36. Sanh, V., Wolf, T., Rush, A.M.: Movement pruning: Adaptive sparsity by finetuning. arXiv preprint arXiv:2005.07683 (2020)
37. Singh, S.P., Alistarh, D.: Woodfisher: Efficient second-order approximations for model compression. arXiv preprint arXiv:2004.14340 (2020)
38. Stanton, S., Izmailov, P., Kirichenko, P., Alemi, A.A., Wilson, A.G.: Does knowledge distillation really work? arXiv preprint arXiv:2106.05945 (2021)
39. Tian, Y., Krishnan, D., Isola, P.: Contrastive representation distillation. arXiv preprint arXiv:1910.10699 (2019)
40. Turc, I., Chang, M.W., Lee, K., Toutanova, K.: Well-read students learn better: On the importance of pre-training compact models. arXiv preprint arXiv:1908.08962 (2019)
41. Ver Steeg, G.: Non-parametric entropy estimation toolbox (npeet). Tech. rep., Technical Report. 2000. Available online: https://www. isi. edu/˜ gregv . . . (2000)
42. Ver Steeg, G., Galstyan, A.: Information-theoretic measures of influence based on content dynamics. In: Proceedings of the sixth ACM international conference on Web search and data mining. pp. 3–12 (2013)
43. Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., Bowman, S.R.: Glue: A multitask benchmark and analysis platform for natural language understanding. arXiv preprint arXiv:1804.07461 (2018)
44. Wang, C., Grosse, R., Fidler, S., Zhang, G.: Eigendamage: Structured pruning in the kronecker-factored eigenbasis. In: International Conference on Machine Learning. pp. 6566–6575. PMLR (2019)
45. Wen, W., Wu, C., Wang, Y., Chen, Y., Li, H.: Learning structured sparsity in deep neural networks. arXiv preprint arXiv:1608.03665 (2016)
46. Yang, C., Xie, L., Qiao, S., Yuille, A.L.: Training deep neural networks in generations: A more tolerant teacher educates better students. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 33, pp. 5628–5635 (2019)
47. Ye, J., Lu, X., Lin, Z., Wang, J.Z.: Rethinking the smaller-norm-less-informative assumption in channel pruning of convolution layers. arXiv preprint arXiv:1802.00124 (2018)
48. Yuan, M., Lin, Y.: Model selection and estimation in regression with grouped variables. Journal of the Royal Statistical Society: Series B (Statistical Methodology) **68**(1), 49–67 (2006)
49. Zbontar, J., Jing, L., Misra, I., LeCun, Y., Deny, S.: Barlow twins: Self-supervised learning via redundancy reduction. arXiv preprint arXiv:2103.03230 (2021)
50. Zhu, M., Gupta, S.: To prune, or not to prune: exploring the efficacy of pruning for model compression. arXiv preprint arXiv:1710.01878 (2017)