Paying more Attention to Snapshots of Iterative Pruning: Improving Model Compression via Ensemble Distillation

Duong H. Le
1610580@hcmut.edu.vn
Trung-Nhan Vo
1612372@hcmut.edu.vn
Nam Thoai
namthoai@hcmut.edu.vn

Ho Chi Minh City University of Technology
Ho Chi Minh, Vietnam

Abstract

Network pruning is one of the most dominant methods for reducing the heavy inference cost of deep neural networks. Existing methods often iteratively prune networks to attain high compression ratio without incurring significant loss in performance. However, we argue that conventional methods for retraining pruned networks (i.e., using small, fixed learning rate) are inadequate as they completely ignore the benefits from snapshots of iterative pruning. In this work, we show that strong ensembles can be constructed from snapshots of iterative pruning, which achieve competitive performance and vary in network structure. Furthermore, we present a simple, general and effective pipeline that generates strong ensembles of networks during pruning with *large learning rate restarting*, and utilizes knowledge distillation with those ensembles to improve the predictive power of compact models. In standard image classification benchmarks such as CIFAR and Tiny-Imagenet, we advance state-of-the-art pruning ratio of structured pruning by integrating simple ℓ_1-norm filters pruning into our pipeline. Specifically, we reduce 75-80% of total parameters and 65-70% MACs of numerous variants of ResNet architectures while having comparable or better performance than that of original networks. Code is available at https://github.com/lehduong/kesi.

1 Introduction

Motivation Researchers have extensively exploited deep and wide networks for the sake of achieving superior performance on various tasks. Most of state-of-the-art networks are extremely computationally expensive and require excessive memory. However, real-world applications usually require running deep neural networks on edge devices for various reasons: user privacy, security, real-time analysis, offline capability, reducing cost for server deployment, and so on. Adopting large and cumbersome networks to such resource-constrained environments is challenging due to restrictions of memory, computational power, energy consumption, and so on.

Background Network pruning [14, 24, 26, 36] reduces a cumbersome and over-parameterized network to compact one by removing unnecessary weights and connections of networks. It is widely believed that small networks pruned from large, over-parameterized networks achieve superior performance than those trained from scratch [10, 26, 31, 37]. A plausible explanation to this phenomenon is the lottery ticket hypothesis [10] i.e. large, over-parameterized networks contain many optimal sub-networks i.e. winning tickets. In particular, network pruning could be done in two manners: *one-shot pruning* - prune a network with the desired compression ratio and retrain it only **one** time, or *iterative pruning* - only prune small ratio of the original network, retrain and repeat that process until the target size is reached. It has been shown that iterative pruning could lead to a greater compression ratio compare to one-shot pruning approaches [14, 26, 31, 37]. Furthermore, Frankle *et al*. [10] point out that iteratively-pruned-winning-tickets learn faster and reach higher test accuracy at smaller network size.

On the other hand, ensembles of neural networks are known to be much more robust and accurate than individual networks [2, 20, 41]. In spite of their superior performance, the tremendous cost of training and inference of ensembles makes them less attractive in practice. For the purpose of accelerating training time of ensembles, prior works proposed methods encouraging models to converge to different local minimums during training [12, 20, 44]. To reduce inference time of ensembles, one could use a single network to mimic behavior of ensembles as pioneered by born-again tree [6] and knowledge distillation [5, 7, 19, 32]. In above approaches, although small networks can not achieve comparable performance with ensembles of networks, *dark knowledge* transferred from teachers to student network could bridge the gap between their predictive powers.

Our proposal While existing methods of iterative pruning are more effective than one-shot pruning, the snapshots at each pruning iteration are mostly overlooked. We consider leveraging the snapshots of iterative pruning to take the performance of compact models to the next level.

In this work, we propose a simple pipeline for model compression by slightly modifying the standard approach. Specifically, we make use of *large learning rate restarting* at each pruning iteration to retrain pruned networks. Hence, each retraining step could be considered as a *cycle* of *Snapshot ensemble* [20]. Utilizing both large learning rate restarting and pruning foster the diversity between snapshots, thus, constructing strong ensembles. Once achieved the desired compression ratio, we then distill the knowledge from the ensembles of snapshots of iterative pruning to the final model. Our method acquires the advantages of network pruning, ensembles learning, and knowledge distillation. To the best of our knowledge, this is the first work attempting to exploit snapshots of iterative pruning to further improve the performance of pruned networks.

Our main contributions The contributions of our work are summarized as below:

1. We empirically show that fine-tuning with large learning rate restarting can achieve competitive or better results than the common strategy (i.e. small, fixed learning rate) on a range of standard datasets and architectures. Surprisingly, such simple modification can create very strong baselines for both structured and unstructured pruning.

2. We demonstrate that snapshots of iterative pruning could construct strong ensembles.

3. We propose a simple pipeline to combine knowledge distillation from ensembles and iterative pruning. We empirically show that our approach can achieve state-of-the-art pruning ratio by reducing $75-80\%$ of parameters and $65-70\%$ MACs on numerous variants of ResNet while having comparable or better results than original networks.

2 Related Work

Knowledge Distillation The approach of training small, efficient student network to mimic behavior of large, over-parameterized network has been proposed for a long time [7] and was recently repopularized in [3, 19]. Later, knowledge distillation was extended to various aspects, transferring knowledge from intermediate layers [38, 48], allowing teachers and students to guide each others [50], using teacher and student with the same architecture [4, 11, 43, 44], distilling knowledge in multiple steps [34]. To address the cost of training two networks in knowledge distillation, [44, 50, 51] propose online approaches to train the student and teacher networks in one generation. Furthermore, Anil *et al*. [1] adopt knowledge distillation to accelerate the training of large scale neural networks. Universally Slimmable networks [46] provide an ensemble of sub-networks that has implicit knowledge distillation through shared weights.

Network Pruning The idea behind network pruning is to reduce the redundant weights and connections of original network to achieve compact networks without losing much performance [14, 26]. In general, pruning can be divided into two categories: structured pruning and unstructured pruning. Unstructured pruning [13, 14, 15, 24, 42] always results in sparse weight matrices, which can not directly accelerate the inference efficiency without specialized hardware/libraries. In contrast, structured pruning approaches [18, 26, 28, 35, 47] remove the redundant weights at the level of filters/channels/layers, thus, speeding up the inference of networks directly. There are numerous approaches to determine redundant filters/weights: [31] use statistic information of the next filters to select unimportant filters, [26] prune the filters that have smallest norms in each layer, [35] select the filters to minimize the construction loss estimated with Taylor expansion. As these criteria are rough estimations of weight's importance, pruning a large number of filters/weights at once might break down and lead to inferior performance compare to iterative pruning [14, 26]. Recently, Liu *et al*. [29] empirically show that training the pruned model from scratch can also achieve comparable or even better performance than fine-tuning. While the efficacy of network pruning remains an open question, in this work, we propose exploiting the benefit of having multiple networks through iterative pruning for constructing ensembles of networks.

3 Knowledge Distillation

Consider the classification problem in which we need to determine the correct category for input image $\mathbf{x}$ among M classes. The probability of class m for sample $\mathbf{x}_n$ given by neural network f parameterized by θ is computed as:

$$p_m(\mathbf{x}_n;\theta,\tau) = \frac{\exp(\frac{f_m(\mathbf{x}_n;\theta)}{\tau})}{\sum_{i=1}^{M}\exp(\frac{f_i(\mathbf{x}_n;\theta)}{\tau})} \tag{1}$$

Where τ is the temperature of softmax function, higher values of τ lead to softer output distribution. Conventional approaches optimize the parameters θ by sampling mini-batches $\mathcal{B}$ from the dataset and update the parameters to minimize cross-entropy objective:

$$\mathcal{L}_{NCE}(\mathcal{B};\theta) = -\frac{1}{N}\sum_{n=1}^{N}\sum_{m=1}^{M} y_m \log p_m(\mathbf{x}_n;\theta,1) \tag{2}$$

The target distribution of a sample is usually represented by *one-hot vector* i.e. only the true class is 1 and all other classes are 0. Since input images might differ in term of noise, complexity, and multi-modality, enforcing networks to excessively fit the delta distribution of ground truth for all samples might deteriorate their generalization. Besides that, the similarity between classes provides rich information for learning and potentially prevent overfitting [43]. Knowledge distillation [7, 19] uses a trained (*teacher*) network, which usually has high capacity, to guide the training of other (*student*) network. Let $q_m(\mathbf{x}_n)$ be the probability of class m for image $\mathbf{x}_n$ given by the teacher network, which is parameterized by ψ. The objective function of knowledge distillation is defined as:

$$\mathcal{L}_{KD}(\mathcal{B};\theta,\tau,\psi) = -\frac{\tau^2}{N}\sum_{n=1}^{N}\sum_{m=1}^{M} q_m(\mathbf{x}_n;\psi,\tau)\log\frac{q_m(\mathbf{x}_n;\psi,\tau)}{p_m(\mathbf{x}_n;\theta,\tau)} \tag{3}$$

In case the teacher is an ensemble of K networks, the target distribution of knowledge distillation is the average of outputs of all networks: $\bar{q}_m(\mathbf{x}_n;\psi_{1:K},\tau) = \frac{1}{K}\sum_{k=1}^{K} q_m(\mathbf{x}_n;\psi_k,\tau)$.

An alternative approach is optimizing the mean of Kullback-Leibler divergence between the student and each teacher network:

$$\mathcal{L}'_{KD}(\mathcal{B};\theta,\tau,\psi_{1:K}) = -\frac{\tau^2}{KN}\sum_{n=1}^{N}\sum_{m=1}^{M}\sum_{k=1}^{K} q_m(\mathbf{x}_n;\psi_k,\tau)\log\frac{q_m(\mathbf{x}_n;\psi_k,\tau)}{p_m(\mathbf{x}_n;\theta,\tau)} \tag{4}$$

We experimented with two above objectives but did not observe significant difference in performance of student networks, thus, we only report results of the *second* approach.

4 Snapshots of Iterative Pruning

In contrast to previous works, which mainly focus on the aforementioned usage of iterative pruning (i.e. alleviating the noise of weight's importance estimation), we exploit the benefits of generating multiple models varying in structure and capacity to construct strong ensembles.

Inspired by the prior works of [30, 39] in which the authors show that promising local optimums could be found in a small number of epochs after restarting the learning rate. Furthermore, Huang *et al*. [20] demonstrate that utilizing large learning rate restarting during training can construct strong ensembles without much additional cost.

Broadly speaking, the performance of ensembles depends on: the performance of individual network and the diversity of them. On the other hand, network pruning generates snapshots varying in structure and achieving competitive performance. *Hence, if pruned networks could achieve minimal loss in predictive power relative to the original network, the ensemble of them could potentially outperforms the ensemble of networks having identical architecture (and trained with large learning rate restarting).*

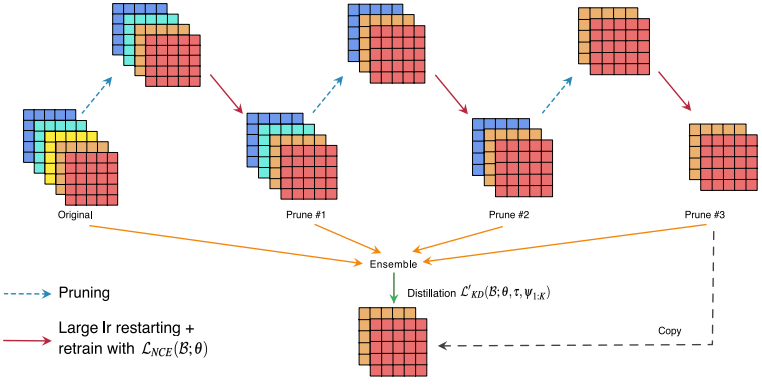


Figure 1: Overview of our approach combining the advantage of knowledge distillation, ensembles of networks, and network pruning. At the start, we prune the filters/weights according to some criteria (ℓ_1-norm, Taylor approximation,...). With KESI, we retrain the pruned networks with large learning rate and minimize the conventional supervised loss function. Once we achieve the desired pruning ratio, we use knowledge distillation to transfer the knowledge from ensembles of snapshots of iterative pruning to the final model.

Prior works such as [14, 29, 35] retrain the pruned networks for T more epochs with a fixed learning rate, which is usually the final learning rate of the training. However, this approach might result in multiple snapshots being stuck in similar local optimums, thus, leading to very weak ensembles as shown in our experiments. Similar to [20], we adopt the large learning rate restarting at each pruning iteration to encourage each snapshot to converge to different optimum. For learning rate restarting, we utilize the One-cycle policy [40], which is proved to increase convergence speed of several models. Due to the similarity of our proposed method and *Snapshot Ensembling* [20], we refer to each pruning and retraining step as a *cycle*. One-cycle policy adjusts learning rate at each mini-batch update and has two phases:

INCREASING LEARNING RATE The learning rate and momentum of optimizer will be initialized to $\eta_{initial}$ and $\beta_{initial}$ respectively. During the first T iterations of fine-tuning, learning rate and momentum gradually increase from initial values to η_{max}, β_{max}. The learning rate and momentum at i-th step with *cosine annealing strategy* are given by:

$$\eta_i = \eta_{max} + \frac{\eta_{initial} - \eta_{max}}{2}(1 + \cos(\frac{i}{T} \cdot \pi)) \tag{5}$$

$$\beta_i = \beta_{max} + \frac{\beta_{initial} - \beta_{max}}{2}(1 + \cos(\frac{i}{T} \cdot \pi)) \tag{6}$$

DECREASING LEARNING RATE After T iterations, learning rate and momentum will be gradually decreased from η_{max} and β_{max} to η_{min} and β_{min} in $L-T$ iterations where L is total number of iterations for fine-tuning.

$$\eta_i = \eta_{min} + \frac{\eta_{max} - \eta_{min}}{2}(1 + \cos(\frac{i-T}{L-T} \cdot \pi)) \tag{7}$$

$$\beta_i = \beta_{initial} + \frac{\beta_{max} - \beta_{initial}}{2}(1 + \cos(\frac{i-T}{L-T} \cdot \pi)) \tag{8}$$

It is worth noticing that differs from previous works [20, 44], which use *cosine annealing schedule*, by using One-cycle policy, we also *"warm-up"* learning rate at the start of each cycle. In our experiments, warming up learning rate is extremely important to achieve high accuracy with deep and large networks.

Surprisingly, retraining with One-cycle policy does not only generate significantly stronger ensembles, but also consistently **outperforms** the standard strategy for finetuning in terms of predictive accuracy of individual snapshots. We hypothesize that the (local) optimums of pruned networks are actually far from those of original networks, thus, large learning rate is needed to guarantee the convergence of pruned networks. We leave rigorous evaluation to investigate this phenomenon for future works.

5 Effective Pipeline for Model Compression

Since we already obtain strong ensembles during pruning, it is straightforward to distill the knowledge from them to the final pruned network. Our proposed pipeline can be summarized as follow:

Algorithm 1: Knowledge Distillation from Ensemble of Snapshots of Iterative pruning

1. TRAIN the baseline model to completion.
2. PRUNE redundant weights of the network based on some criteria.
3. RETRAIN the pruned network with **large learning rate**.
4. REPEAT step 2 and 3 until desired compression ratio is reached.
5. DISTILL knowledge from ensembles of snapshots of pruning.

From now, we refer to our pipeline for model compression as *Knowledge Distillation from Ensembles of Snapshots of Iterative Pruning* (KESI). An overview of our approach is depicted in Figure 1. Our approach is extremely simple, easy to implement and can be adopted with any pruning mechanisms. We discuss the reasons why ensembles of snapshots of pruning are naturally suited for knowledge distillation.

Quality of Teacher In knowledge distillation, student can either learn to jointly optimize the supervised loss (Equation 2) and knowledge distillation loss (Equation 4) or only optimize the distillation objective. In the former case, if the teacher is poorly trained, mathematically speaking, the two objectives will conflict with each other. In the latter case, a poor teacher provides weak supervision (noisy label), making it's harder to learn from the student's perspective. Furthermore, ensembles provide more robust predictions on noisy labeled datasets [25] and out-of-distribution examples [23].

Student and Teacher Gap Although ensembles of snapshots have superior performance than the original network, it is not sufficient to guarantee the improvement in the performance of the student network with Knowledge Distillation. In fact, many works such as [8, 33, 43] show that a powerful teacher might impair the performance of its student if there is a large gap between their predictive powers. However, ensembles of snapshots of pruning consist of

models varying in capacity. Hence, teacher's predictions of hard-to-learn samples (because of their complexity, multi-modality) will have softer distributions as the small networks could not "remember" those samples and would be more uncertain about them.

In this work, we only investigate knowledge distillation from ensembles of fixed-weights teachers, however, we can also jointly train all models and allow them to guide each other, which is referred to as *deep mutual learning* [50].

6 Experiments

We conduct experiments on CIFAR-10, CIFAR-100 [22] and Tiny-Imagenet [1] datasets.

The two CIFAR datasets [22] consist of colored natural images sized at 32×32 pixels. CIFAR-10 (C10) and CIFAR-100 (C100) images are drawn from 10 and 100 classes, respectively. For each dataset, there are 50,000 training images and 10,000 images reserved for testing.

The Tiny ImageNet dataset consists of a subset of ImageNet images [9]. There are 200 classes, each of which has 500 training images and 50 validation images. Each image is resized to 64×64 and augmented with random crops, horizontal mirroring, and RGB intensity scaling.

We run each experiment 3 times then report mean and standard deviation of each network. In our experiments, we prune all networks in 5 *cycles* unless otherwise stated.

6.1 Experiment setup

Training baselines

We adopt the training and pruning code from [29] [2]. We train all networks with Stochastic Gradient Descent (SGD), learning rate is dropped from 0.1 to 0.01 at 50% training and to 0.001 at 75%. The batch size is set to 128 and weight decay is 0.0001 similar to [16, 17].

CIFAR In order to create strong baseline models, we extend the training schedule of all models to 300 epochs. For WideResnet, we use same configurations as described in [49].

Tiny-Imagenet we adopt Pytorch's pretrained models on ImageNet and only replace the last fully-connected layer and train networks for $T = 100$ more epochs. We warm up learning rate from 0.01 to 0.1 in 10 epochs. Other configurations are adopted from CIFAR training recipe.

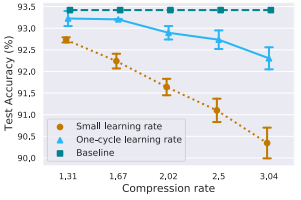
Pruning

Structured pruning we use ℓ_1-norm based filters pruning [26] for simplicity. In each layer, a fixed number of filters having smallest ℓ_1-norm will be pruned. Since the bulk of networks tend to be last layers, we increase the percentage of filters that will be pruned as the layer goes deeper to achieve higher compression ratio.

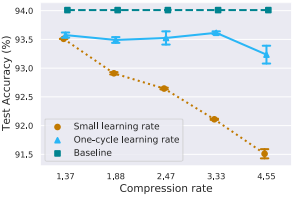
Unstructured pruning, we exploit (global) magnitude-based weight pruning [14] i.e. pooling parameters across all layers and pruning weights with lowest magnitude. Specifically, we only prune parameters of convolutional layers similar to [29].

[1] https://tiny-imagenet.herokuapp.com

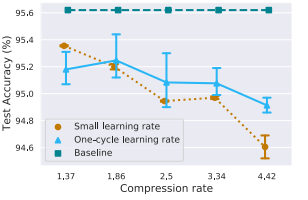
[2] https://github.com/Eric-mingjie/rethinking-network-pruning



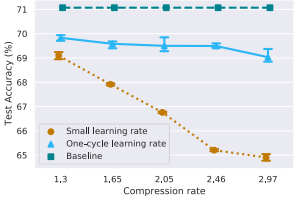
(a) Resnet-56 on CIFAR-10



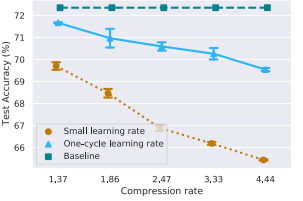
(b) Resnet-110 on CIFAR-10



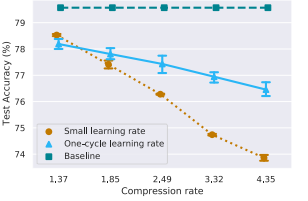
(c) WideResnet-16-8 on CIFAR-10



(a) Resnet-56 on CIFAR-100



(b) Resnet-110 on CIFAR-100



(c) WideResnet-16-8 on CIFAR-100

Figure 2: **ℓ_1-norm filters pruning** [26] with standard small, fixed learning rate and One-cycle learning rate.

Retraining

The budget for fine-tuning of each cycle is $T = 40$ and $T = 25$ epochs on CIFAR and Tiny-Imagenet datasets respectively regardless of model architectures. In *standard policy*, the learning rate is set to 0.001 and fixed during retraining.

For *One-cycle policy*, we set the initial learning rate $\eta_{initial} = 0.01$, gradually increase it to the maximum learning rate $\eta_{max} = 0.1$ in 10% of total (retrain) epochs, then decrease it to the minimum learning rate $\eta_{min} = 0.0001$ for remaining epochs. Other configurations are identical to those of training.

Knowledge Distillation

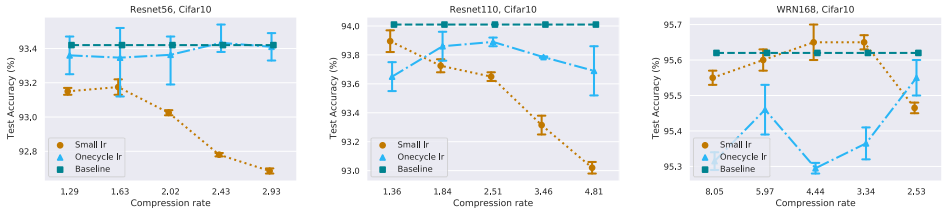
We use *Adam* optimizer [21] for ensemble distillation since it gives better results than vanilla SGD in our experiments. For knowledge distillation, we also adopt One-cycle policy where we set $\eta_{initial}, \eta_{max}, \eta_{min}$ to $1e{-}4, 1e-3, 1e{-}6$ respectively. We do not explicitly use regularization for knowledge distillation. Other configurations *e.g.* batch size, number of retraining epochs,... are similar to normal finetuning.

In our experiments, we use temperature $\tau = 5$. The teachers i.e. ensembles of snapshots consist of 6 models including the original (unpruned) network and 5 snapshots of pruning.

6.2 Results

6.2.1 Effectiveness of large learning rate

We conduct experiments to empirically evaluate the performance of pruned networks trained with large learning rate compare to networks fine-tuned with small learning rate. Figure 2 and 3 demonstrate results of pruned networks with different compression ratios for both structured and unstructured pruning. Exhaustive results are reported in supplementary documents.



(a) Resnet-56 on CIFAR-10 (b) Resnet-110 on CIFAR-10 (c) WideResnet-16-8 on CIFAR-10

Figure 3: **Unstructured magnitude-based pruning** [14] with standard small, fixed learning rate and One-cycle learning rate.

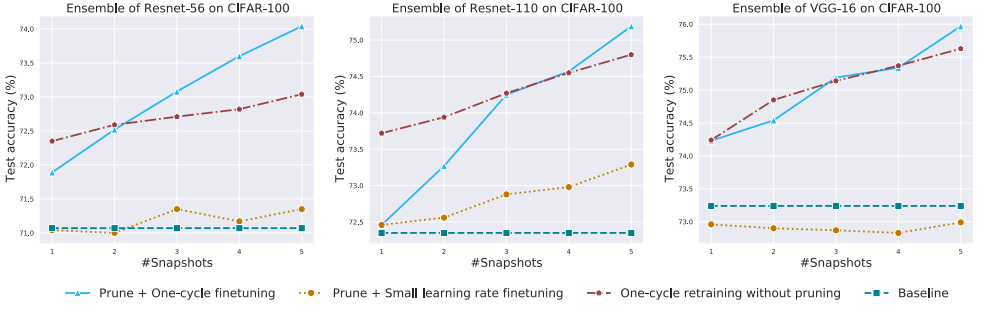


Figure 4: Performance of ensembles of snapshots with different approaches on CIFAR-100.

6.2.2 Performance of ensembles of snapshots

We compare the performance of ensembles of snapshots with different approaches: snapshots of pruned networks trained with small learning rate, snapshots of pruned networks trained with large learning rate restarting and snapshots of unpruned networks retrained with large learning rate (i.e. all snapshots have same architecture as the original network). Figure 4 presents the result of this experiment.

We can see that although the network capacity is decreased at each *cycle*, the ensembles of snapshots of iterative pruning achieve competitive or even better than snapshots of networks with same architecture. Detailed results of performance of ensembles are reported in supplementary documents.

6.2.3 Performance of compact networks trained with our pipeline

In this section, we demonstrate that the smaller models trained with our pipeline (KESI) achieve comparable or even better results than the original model. Each final model is iteratively pruned and retrained in 5 *cycles* with different strategies. Table 2 and 3 present the performance of compact models on CIFAR-10, CIFAR-100 and Tiny-Imagenet. Specifically, we compare the iteratively-pruned-models retrained with small learning rate, large learning rate and our pipeline (i.e. large learning rate + knowledge distillation). Our pipeline consistently outperforms the standard strategy by a large margin for both structured and unstructured pruning.

Although our approach is general and can be applied to any (iterative) pruning mechanism, we also give a comparison of model trained with our pipeline and conventional approaches in table 1. We conduct experiment to compare performance of student networks

Model	Methods	% Params ↓	% FLOPs ↓	baseline	pruned
Resnet-56	CP [18]	-	50.6	92.80	91.80
	FPEC [26]	14.1	27.6	93.04	93.06
	NISP [47]	42.4	35.5	93.26	93.01
	GAL-0.8 [28]	65.9	60.2	93.26	91.58
	GBN [45]	66.7	**70.3**	93.10	93.07
	HRank [27]	42.4	50.0	93.26	93.17
	PFEC+KESI (our)	**67.1**	61.5	**93.42**	**93.34 ± 0.05**
Resnet-110	PFEC [26]	32.6	38.7	93.53	93.30
	GAL-0.5 [28]	44.8	48.5	93.50	92.74
	HRank [27]	68.7	**68.6**	93.52	92.65
	PFEC+KESI (our)	**77.5**	65.4	**94.01**	**94.01 ± 0.22**

Table 1: Comparing performance of pruned networks with other approaches on CIFAR-10 dataset.

Model	Structured Pruning					Unstructured Pruning			
	Method	# Params(M)	% MACs ↓	C10	C100	Method	# Params(M)	C10	C100
Resnet-56	baseline	0.85	0.00	93.42	71.07	baseline	0.85	93.42	71.07
	PFEC [26]	0.28	61.5	90.35 ± 0.36	64.91 ± 0.14	MWP [14]	0.29	92.69 ± 0.02	69.53 ± 0.07
	PFEC+One-cycle	0.28	61.5	92.31 ± 0.26	69.03 ± 0.24	MWP+One-cycle	0.29	93.41 ± 0.08	70.46 ± 0.30
	PFEC+KESI(our)	0.28	61.5	**93.34 ± 0.05**	**70.95 ± 0.11**	MWP+KESI(our)	0.29	**93.90 ± 0.10**	**72.27 ± 0.09**
Resnet-110	baseline	1.73	0.00	94.01	72.35	baseline	1.73	94.01	72.35
	PFEC [26]	0.39	65.38	91.51 ± 0.08	65.44 ± 0.04	MWP [14]	0.36	93.02 ± 0.04	68.90 ± 0.08
	PFEC+One-cycle	0.39	65.38	93.24 ± 0.16	69.54 ± 0.07	MWP+One-cycle	0.36	93.69 ± 0.17	71.59 ± 0.30
	PFEC+KESI(our)	0.39	65.38	**94.01 ± 0.22**	**72.12 ± 0.11**	MWP+KESI(our)	0.36	**94.44 ± 0.11**	**73.12 ± 0.25**
Preresnet-164	baseline	1.70	0.00	95.06	76.35	baseline	1.70	95.06	76.35
	PFEC [26]	0.31	69.23	92.05 ± 0.11	69.20 ± 0.04	MWP [14]			
	PFEC+One-cycle	0.31	69.23	94.15 ± 0.06	73.99 ± 0.06	MWP+One-cycle			
	PFEC+KESI(our)	0.31	69.23	**94.30 ± 0.53**	**75.84 ± 0.32**	MWP+KESI(our)			
WideResnet-16-8	baseline	10.96	0.00	95.62	79.57	baseline	10.96	95.62	79.57
	PFEC [26]	2.48	64.52	94.61 ± 0.09	73.82 ± 0.10	MWP [14]	2.53	95.47 ± 0.07	77.92 ± 0.16
	PFEC+One-cycle	2.48	64.52	94.91 ± 0.04	76.46 ± 0.27	MWP+One-cycle	2.53	95.55 ± 0.05	78.82 ± 0.11
	PFEC+KESI(our)	2.48	64.52	**95.68 ± 0.12**	**79.01 ± 0.20**	MWP+KESI(our)	2.53	**95.97 ± 0.05**	**80.08 ± 0.06**
VGG-16	baseline	14.99	0.00	94.23	73.24	baseline	14.99	94.23	73.24
	PFEC [26]	2.71	45.16	93.88 ± 0.12	68.37 ± 0.09	MWP [14]	1.02	93.47 ± 0.22	68.39 ± 0.21
	PFEC+One-cycle	2.71	45.16	94.10 ± 0.09	71.95 ± 0.04	MWP+One-cycle	1.02	93.53 ± 0.10	71.74 ± 0.15
	PFEC+KESI(our)	2.71	45.16	**94.59 ± 0.09**	**73.52 ± 0.20**	MWP+KESI(our)	1.02	**94.01 ± 0.06**	**73.91 ± 0.09**

Table 2: Accuracy (%) of pruned networks on CIFAR-10 and CIFAR-100 datasets trained with different strategies. PFEC (or MWP) are models pruned with ℓ_1-norm filters pruning [26] (or magnitude-based weights pruning [14]) and fine-tuned with small learning rate. PFEC/MWP+One-cycle are pruned networks retrained with large learning rate restarting. PFEC/MWP+KESI are pruned networks retrained with our pipeline

Table 3: Performance of compact models on Tiny-Imagenet

Model	Method	#Params (M)	MACs(G)	Acc
Resnet-18	baseline	11.01	1.82	67.22
	FPEC [26]	2.71	0.83	61.06 ± 0.32
	FPEC+One-cycle	2.71	0.83	64.70 ± 0.33
	FPEC+KESI (our)	2.71	0.83	**66.87 ± 0.26**
Resnet-34	baseline	21.39	3.68	68.81
	FPEC [26]	5.40	1.57	64.93 ± 0.15
	FPEC+One-cycle	5.40	1.57	67.26 ± 0.21
	FPEC+KESI (our)	5.40	1.57	**70.02 ± 0.43**

Table 4: Knowledge distillation with ensembles teacher and single model teacher

Model	Method	#Params (M)	C10	C100
Resnet-56	baseline	0.85	93.42	71.07
	single teacher	0.28	93.13 ± 0.04	70.29 ± 0.14
	ensemble teacher	0.28	**93.34 ± 0.05**	**72.27 ± 0.09**
Resnet-110	baseline	1.73	94.01	72.35
	single teacher	0.39	93.48 ± 0.05	71.50 ± 0.11
	ensemble teacher	0.39	**94.01 ± 0.22**	**73.12 ± 0.25**
WRN-16-8	baseline	19.96	95.62	79.57
	single teacher	2.48	95.37 ± 0.21	78.71 ± 0.24
	ensemble teacher	2.48	**95.68 ± 0.12**	**79.01 ± 0.20**

trained with single teacher (i.e. original/unpruned networks) and ensembles teacher in table 4 for ablation study. We can see that compact models learn from ensembles outperform those learn from a single teacher by a large margin.

7 Conclusion

We propose a simple pipeline by slightly modifying the standard approach to acquire the advantages of network ensembles, knowledge distillation and network pruning. Our experiments show that small and compact networks trained with our pipeline significantly outperform the standard approach and create very strong baselines for model compression. Specifically, our method reduces nearly 80% of parameters and 70% FLOPs of several models by structured pruning without incurring loss in performance.

Acknowledgement The authors thank anonymous reviewers and area chairs for their useful feedback. We also want to express our appreciation to Ms. Le Thi Tham Quynh for her valuable aids in the final preparation of the paper.

References

[1] Rohan Anil, Gabriel Pereyra, Alexandre Passos, Robert Ormandi, George E Dahl, and Geoffrey E Hinton. Large scale distributed neural network training through online distillation. *arXiv preprint arXiv:1804.03235*, 2018.

[2] Arsenii Ashukha, Alexander Lyzhov, Dmitry Molchanov, and Dmitry Vetrov. Pitfalls of in-domain uncertainty estimation and ensembling in deep learning. *arXiv preprint arXiv:2002.06470*, 2020.

[3] Jimmy Ba and Rich Caruana. Do deep nets really need to be deep? In *Advances in neural information processing systems*, pages 2654–2662, 2014.

[4] Hessam Bagherinezhad, Maxwell Horton, Mohammad Rastegari, and Ali Farhadi. Label refinery: Improving imagenet classification through label progression. *arXiv preprint arXiv:1805.02641*, 2018.

[5] Anoop Korattikara Balan, Vivek Rathod, Kevin P Murphy, and Max Welling. Bayesian dark knowledge. In *Advances in Neural Information Processing Systems*, pages 3438–3446, 2015.

[6] Leo Breiman and Nong Shang. Born again trees.

[7] Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 535–541, 2006.

[8] Jang Hyun Cho and Bharath Hariharan. On the efficacy of knowledge distillation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4794–4802, 2019.

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[10] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018.

[11] Tommaso Furlanello, Zachary C Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. Born again neural networks. *arXiv preprint arXiv:1805.04770*, 2018.

[12] Timur Garipov, Pavel Izmailov, Dmitrii Podoprikhin, Dmitry P Vetrov, and Andrew G Wilson. Loss surfaces, mode connectivity, and fast ensembling of dnns. In *Advances in Neural Information Processing Systems*, pages 8789–8798, 2018.

[13] Yiwen Guo, Anbang Yao, and Yurong Chen. Dynamic network surgery for efficient dnns. In *Advances in neural information processing systems*, pages 1379–1387, 2016.

[14] Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. In *Advances in neural information processing systems*, pages 1135–1143, 2015.

[15] Stephen José Hanson and Lorien Y Pratt. Comparing biases for minimal network construction with back-propagation. In *Advances in neural information processing systems*, pages 177–185, 1989.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer, 2016.

[18] Yihui He, Xiangyu Zhang, and Jian Sun. Channel pruning for accelerating very deep neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1389–1397, 2017.

[19] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[20] Gao Huang, Yixuan Li, Geoff Pleiss, Zhuang Liu, John E Hopcroft, and Kilian Q Weinberger. Snapshot ensembles: Train 1, get m for free. *arXiv preprint arXiv:1704.00109*, 2017.

[21] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[22] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[23] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in neural information processing systems*, pages 6402–6413, 2017.

[24] Yann LeCun, John S Denker, and Sara A Solla. Optimal brain damage. In *Advances in neural information processing systems*, pages 598–605, 1990.

[25] Jisoo Lee and Sae-Young Chung. Robust training with ensemble consensus. *arXiv preprint arXiv:1910.09792*, 2019.

[26] Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. *arXiv preprint arXiv:1608.08710*, 2016.

[27] Mingbao Lin, Rongrong Ji, Yan Wang, Yichen Zhang, Baochang Zhang, Yonghong Tian, and Ling Shao. Hrank: Filter pruning using high-rank feature map. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1529–1538, 2020.

[28] Shaohui Lin, Rongrong Ji, Chenqian Yan, Baochang Zhang, Liujuan Cao, Qixiang Ye, Feiyue Huang, and David Doermann. Towards optimal structured cnn pruning via generative adversarial learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2790–2799, 2019.

[29] Zhuang Liu, Mingjie Sun, Tinghui Zhou, Gao Huang, and Trevor Darrell. Rethinking the value of network pruning. *arXiv preprint arXiv:1810.05270*, 2018.

[30] Ilya Loshchilov and Frank Hutter. Sgdr: stochastic gradient descent with restarts. corr abs/1608.03983 (2016). *arXiv preprint arXiv:1608.03983*, 2016.

[31] Jian-Hao Luo, Jianxin Wu, and Weiyao Lin. Thinet: A filter level pruning method for deep neural network compression. In *Proceedings of the IEEE international conference on computer vision*, pages 5058–5066, 2017.

[32] Andrey Malinin, Bruno Mlodozeniec, and Mark Gales. Ensemble distribution distillation. *arXiv preprint arXiv:1905.00076*, 2019.

[33] Seyed-Iman Mirzadeh, Mehrdad Farajtabar, Ang Li, Nir Levine, Akihiro Matsukawa, and Hassan Ghasemzadeh. Improved knowledge distillation via teacher assistant.

[34] Seyed-Iman Mirzadeh, Mehrdad Farajtabar, Ang Li, and Hassan Ghasemzadeh. Improved knowledge distillation via teacher assistant: Bridging the gap between student and teacher. *arXiv preprint arXiv:1902.03393*, 2019.

[35] Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. Pruning convolutional neural networks for resource efficient inference. *arXiv preprint arXiv:1611.06440*, 2016.

[36] Russell Reed. Pruning algorithms-a survey. *IEEE transactions on Neural Networks*, 4 (5):740–747, 1993.

[37] Alex Renda, Jonathan Frankle, and Michael Carbin. Comparing rewinding and fine-tuning in neural network pruning. *arXiv preprint arXiv:2003.02389*, 2020.

[38] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. *arXiv preprint arXiv:1412.6550*, 2014.

[39] Leslie N Smith. No more pesky learning rate guessing games. *CoRR, abs/1506.01186*, 5, 2015.

[40] Leslie N Smith and Nicholay Topin. Super-convergence: Very fast training of neural networks using large learning rates. In *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications*, volume 11006, page 1100612. International Society for Optics and Photonics, 2019.

[41] Jasper Snoek, Yaniv Ovadia, Emily Fertig, Balaji Lakshminarayanan, Sebastian Nowozin, D Sculley, Joshua Dillon, Jie Ren, and Zachary Nado. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. In *Advances in Neural Information Processing Systems*, pages 13969–13980, 2019.

[42] Suraj Srinivas and R Venkatesh Babu. Data-free parameter pruning for deep neural networks. *arXiv preprint arXiv:1507.06149*, 2015.

[43] Chenglin Yang, Lingxi Xie, Siyuan Qiao, and Alan L Yuille. Training deep neural networks in generations: A more tolerant teacher educates better students. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5628–5635, 2019.

[44] Chenglin Yang, Lingxi Xie, Chi Su, and Alan L Yuille. Snapshot distillation: Teacher-student optimization in one generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2859–2868, 2019.

[45] Zhonghui You, Kun Yan, Jinmian Ye, Meng Ma, and Ping Wang. Gate decorator: Global filter pruning method for accelerating deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 2133–2144, 2019.

[46] Jiahui Yu and Thomas S Huang. Universally slimmable networks and improved training techniques. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1803–1811, 2019.

[47] Ruichi Yu, Ang Li, Chun-Fu Chen, Jui-Hsin Lai, Vlad I Morariu, Xintong Han, Mingfei Gao, Ching-Yung Lin, and Larry S Davis. Nisp: Pruning networks using neuron importance score propagation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9194–9203, 2018.

[48] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. *arXiv preprint arXiv:1612.03928*, 2016.

[49] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

[50] Ying Zhang, Tao Xiang, Timothy M Hospedales, and Huchuan Lu. Deep mutual learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4320–4328, 2018.

[51] Xiatian Zhu, Shaogang Gong, et al. Knowledge distillation by on-the-fly native ensemble. In *Advances in neural information processing systems*, pages 7517–7527, 2018.