TRANSFER LEARNING WITH FINE-GRAINED SPARSE NETWORKS: FROM AN EFFICIENT NETWORK PERSPECTIVE

Xiaomin Li [1] **Cody Blakeney** [1] **Ziliang Zong** [1]

ABSTRACT

The deployment of Deep Neural Networks (DNNs) on edge-devices has been hindered by their high computation cost and memory footprint. Moreover, the lack of high-quality labeled training data also prevents DNNs from being effectively implemented for various mobile apps running on resource-constrained devices. Pruning can reduce the model size and calculation time by removing less important parameters in DNNs. Transfer learning allows a learned model to quickly adapt to a similar new task with less demand on training data. However, can we leverage the advantages from both pruning and transfer learning still remains largely unknown. This study strives to answer two fundamental research questions: 1) Can we combine pruning and transfer learning? 2) How to effectively integrate pruning and transfer learning? To answer these questions, we implement an unstructured gradual pruning method and image classification transfer learning task on several widely used DNN models (e.g. ResNet and VGG) and datasets (ImageNet, CIFAR10, and Food101). We find that 1) An appropriately pruned model (with moderate sparsity ratio) can endow even better generalization ability on transfer learning tasks. 2) The order of pruning and transfer learning has large impact on DNNs' accuracy and training time. Our experiments show that pruning a ResNet34 model (pre-trained on ImageNet) to 50% sparsity, transfer it to CIFAR10 dataset, then further prune it to 80% sparsity can increase its accuracy by 0.5% and reduce training time by 3x. The training time can be reduced by 70x with 0.37% better accuracy if we transfer the model first then prune it to 80% sparsity. 3) Pruning is expensive thus trade-offs between accuracy gain and training cost need to be considered in practice.

1 INTRODUCTION

In recent years, Deep Neural Networks (DNNs) have solved many challenging tasks in computer vision, speech recognition, and natural language processing. Despite the breakthroughs various DNNs have made in different fields, two factors are hindering (or slowing down) the deployment of DNN applications on resource-constrained devices such as mobile phones, drones, etc. The first limiting factor is the growing size and deeper architecture of DNN models, which requires larger storage space, longer training/inference time and consumes more power. Therefore, it is paramount to develop techniques that can alleviate the greedy resource needs of dense and deep DNN models without compromising accuracy. The second issue is the lack of high-quality labeled data, which prevents DNNs from being widely and quickly implemented in real practice. Training a model from scratch for a new task takes substantial amount of time to reach convergence. Meanwhile, the insufficient training data will hurt the model's accuracy. Model pruning has been proposed to address the first problem by removing or masking less important parameters in a DNN model. With moderate pruning rate, it can significantly reduce the demand of DNN models for resources while keeping its accuracy. Transfer learning, which transfers knowledge learned from a well-trained model to a new domain of interest, has been verified as an effective solution to the second problem. The transfer learning process usually only requires minor adjustments of the original model but often outperforms training from scratch while significantly reducing the amount of training data and training time.

A natural question to ask is can we combine pruning with transfer learning to further promote the deployment of DNNs on resource constrained devices? Furthermore, if we can take advantage of both techniques, what is the best way to integrate them? To the best of our knowledge, the answers to these questions remain unknown. No existing literature has thoroughly investigated this problem yet because combining pruning and transfer learning seems risky at the first glance. During the pruning process, a large portion of parameters in a DNN model are marked as less important

[*]Equal contribution [1]Department of Computer Science, Texas State University, Texas, United States. Correspondence to: Ziliang Zong <ziliang@txstate.edu>.

Resource-Constrained Machine Learning (ReCoML) Workshop of MLSys 2020 Conference, Austin, TX, USA, 2020.

thus removed or masked. This might hurt the performance of the transer learning task because the pruned parameters could be useful for the generalization ability of the DNN model. Making decisions about which parameters are less important is not trivial and largely based on the input data. Therefore, a pruned model may have better performance on specific features of input data but performs poorly on transfer learning tasks.

To verify if the aforementioned concerns are true or not, we explore the possibility of combining model pruning with transfer learning in this study. Specifically, we strive to answer the following two key research questions: 1) Can we combine transfer learning and pruning? 2) If we can benefit from the combination of pruning and transfer learning, what is the best way to integrate them? To answer these questions, we pick two widely used convolutional neural networks (VGG16 (Simonyan & Zisserman, 2014) and ResNet34 (He et al., 2016)) and choose the transfer learning task that classifies images for various size of datasets (e.g. ImageNet (Deng et al., 2009), CIFAR10 (Krizhevsky et al., 2009), and Food101(Bossard et al., 2014)). During the investigation, we implement a simple gradual pruning approach (Zhu & Gupta, 2017) that requires minimal tuning and can be seamlessly incorporated with various model architectures. For transfer learning, we change the number of classes in the last layer and take the rest of the layers as fixed feature extractors for the new datasets.

The primary findings are summarized below:

- Pruning and transfer learning can work together. Training a model on a large dataset and prune it not over aggressively (with the sparsity ratio less than 90%) is able to improve the generalization ability of the transfer learning task comparing to its dense counterpart.

- The orders of conducting pruning and transfer learning have significant impact on model accuracy and training time. Among three different orders (i.e. prune before transfer learning, prune after transfer learning, prune before and after transfer learning), our experimental results show that prune the ResNet34 model (pre-trained on ImageNet) to 50% sparsity first, and conduct transfer learning to CIFAR10 dataset then further prune the transferred model to 80% sparsity has the best performance, which can increase the accuracy by 0.5% and reduce the training time by 3x.

- Pruning is resource demanding and time consuming, especially when the input dataset is large. The tradeoffs between the accuracy gain and computation cost when combine transfer leaning with pruned models should be carefully considered.

2 Related Work

Parameter Pruning Pruning induces sparsity in weights and activations of neural networks to reduce parameter size and computation times. Optimal Brain Damage (LeCun et al., 1990) and Optimal Brain Surgeon (Hassibi & Stork, 1993) are the earliest works that proposed the idea of network pruning in the 1990s. (Han et al., 2015) pruned network weights with a small magnitude threshold and introduced a three-step iterative pruning pipeline approach, which brought network pruning to public attention. Afterward, pruning further diverged into unstructured pruning and structured pruning. Unstructured pruning increases the sparsity of weights in neural networks and generates fine-grained sparse weight tensors. The drawbacks are it only generates smaller models when special sparse matrix storage methods are used and only reduces computation cost as well as run-time memory when deployed on specialized hardware. For structured pruning, there are vector level, kernel level, filter level, and channel level pruning strategies (Mao et al., 2017). The coarser the pruning level is, the smaller the final models are. Compared to unstructured pruning, structured pruning directly changes network architectures and does not require any special hardware. However, it tends to suffer from lower accuracy. The most recently works (Frankle & Carbin, 2018; Liu et al., 2018) started to explore the essence of pruning, which claim sub-networks of the original network can achieve comparable accuracy from random re-initialization comparing to the original one.

Knowledge Transfer Training a DNN from scratch with randomly initialized parameters is computationally intensive and time consuming. In addition, it requires massive high quality labeled data as input, which could be hard to collect in practice. Knowledge transfer can leverage the learned representations from other models and generalize its knowledge for a new task with less training data (Pan & Yang, 2009). For example, the DNNs trained on a large-scale image classification dataset such as ImageNet (Deng et al., 2009) has proven to be excellent feature extractors for a broad range of visual tasks such as image captioning (Karpathy & Fei-Fei, 2015; Lu et al., 2017) and visual question answering (Xu & Saenko, 2016). Depending on whether the labeled data are available in a source domain and target domain, transfer learning can be divided into three categories, which are inductive, transductive and unsupervised transfer learning (Pan & Yang, 2009). The method used in our study belongs to inductive transfer learning because it has both labeled data in the source and target domain.

The combination of pruning and transfer learning seems to be counter-intuitive. After all, the pruning process aims to reduce model size by removing or masking unimportant weights (i.e. part of the model information will be erased). Transfer learning, on the other hand, strives to utilize the

learned representations from other datasets as much as possible to learn a new task in a more efficient way (i.e. any missing information of the model could negatively affect its generalization ability in a transfer learning task). Therefore, to the best of our knowledge, none of the previous work had conducted a comprehensive study on the feasibility of combining pruning with transfer learning. In this paper, we prove that pruning not only works well with transfer learning but also performs better than dense models if they are integrated in an appropriate order.

3 METHODOLOGY

In this section, we describe the methodologies in detail on conducting transfer learning with fine-grained sparse networks.

3.1 Automatic gradual unstructured pruning

Comparing to other unstructured pruning methods, the automatic gradual pruning is the preferred method for our experiments as it is succinct, convenient and flexible. This method was first proposed by (Zhu & Gupta, 2017), which concluded that large sparse models tend to achieve better performance than small dense models at the same memory footprint. Unlike other magnitude-based weight pruning methods (Han et al., 2015; See et al., 2016; Narang et al., 2017), the automatic gradual pruning strategy does not require the pre-processing and parameter analysis steps because it prunes each layer equally to the desired sparsity. We leverage this method to further explore how well large sparse models perform on transfer learning tasks. Assuming the sparsity ratio of a DNN model is S, automatic gradual pruning prunes the model from an initial sparsity S_i to a final sparsity value S_f within n pruning steps, starting at training step t_0 with pruning frequency Δt. The temporary sparsity S_t at each step is represented as equation 1.

$$s_t = s_f + (s_i - s_f)\left(1 - \frac{t - t_0}{n\Delta t}\right)^3 \quad \text{for } t \in \{t_0, t_0 + \Delta t, \ldots, t_0 + n\Delta t\} \tag{1}$$

The intuition behind equation 1 is to prune the network rapidly in the initial phase when the redundant connections are abundant and gradually reduce the pruning rate as if the redundancy of weights is dropping gradually. Figure 1 shows the pruning curve of a ResNet20 model trained on the CIFAR10 dataset and pruned to 95% sparsity within 75 epochs, from which we can observe the decreasing pruning slope when the model is approaching to the expected sparsity. To be consistent, we prune all layers except the first input layer and the last fully connected layer in the experiments shown in Section 4. For every layer pruned, a binary mask variable with the same size and shape as the layer's weight tensor is added and it determines which weights participate in the forward execution, as demonstrated in (Zhu & Gupta, 2017). Since the binary weight masks are updated in each Δt step as the network is trained to gradually increase the sparsity of the network, it allows the model to recover from any pruning-induced loss in accuracy. To accurately illustrate how does pruning affect model generalization ability, we set $\Delta t = 1$, which disables the capability of network to adjust their accuracy in pruning steps.

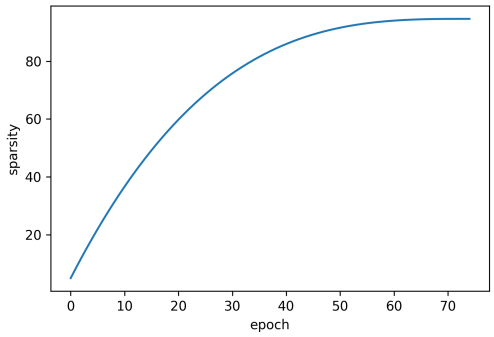

Figure 1. Gradual Pruning on ResNet20 model

We prune VGG16 and ResNet34 models that are pre-trained on ImageNet using the automatic gradual pruning approach. The initial learning rate of the pruning process is set to 0.005, and it reduces 5% in every epoch. In the pruning before transfer learning process (PB), we prune pre-trained models for 10 epochs from sparsity S_i to sparsity S_f and let the networks adjust their parameters for another 5 epochs. In the pruning after transfer learning process (PA), we prune transferred models for 25 epochs and let the network adjust for 25 epochs. The batch size of all Resnet34 pruning is set as 64 whereas the batch size of pruning VGG16 varies from 16 to 128.

3.2 Transfer learning

To accelerate the learning process, it is typical to use a pre-trained model on a very large dataset and take their parameters as initialization for the new image classification tasks. Based on the similarity and the relative size of the new dataset comparing to the original dataset, the transferred models can be viewed as the fixed feature extractors (learned parameters are frozen) or fine-tuned feature extractors (learned parameters can be changed to fit new tasks) on the new dataset.

Three image classification datasets are used in our experiments, which are **CIFAR10** (Krizhevsky et al., 2009), **Food101** (Bossard et al., 2014) and **ImageNet** (Deng et al., 2009). For all input images, we crop and resize them to the 224x224 resolution and use a standard data augmentation scheme (He et al., 2016; Huang et al., 2016; Lin

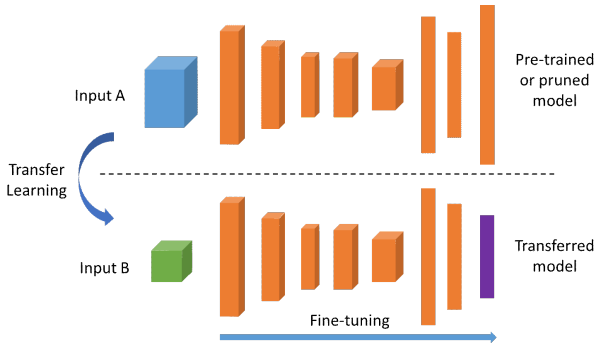

Figure 2. Transfer learning process

et al., 2013) (shifting/mirroring) to pre-process the data. All pruning and transfer learning experiments are conducted on two popular network architectures - **VGGNet** (Simonyan & Zisserman, 2014) and **ResNet** (He et al., 2016). The pre-trained VGG16 and ResNet34 models on ImageNet from the Pytorch (Paszke et al., 2017) framework are selected as baseline models. For all transfer learning processes, we change the number of neurons in the last fully connected layer into the number of classes of the target dataset and fine tune all weights (See Figure 2). We set the initial learning rate as 0.005 and it degrades 5% in every epoch afterwards. The batch size is 128 and the models are trained for 50 epochs respectively. As Table 1 shows, training ResNet34 and VGG16 models with CIFAR10 and Food101 dataset from scratch gets much worse accuracy comparing to training the corresponding transferred models that are pre-trained on ImageNet.

Model	Dataset	Scratch Acc%	Transfer Acc%
ResNet34	Cifar10	88.720	**96.130**
	Food101	70.657	**82.178**
VGG16	Cifar10	88.210	**93.080**
	Food101	72.412	**78.547**

Table 1. Accuracy comparison between training from scratch and transfer learning. Scratch Acc% represents the accuracy of training the models from CIFAR10 or Food101 datasets with random initialization. Transfer Acc% represents the accuracy of training the models on ImageNet dataset then transfer to the CIFAR10 and Food101 datasets respectively.

4 Experiments and Results

In this section, we design a set of experiments to answer two key research questions. First, what is the impact of pruning on transfer learning? Can they work together? Second, what is the best order (prune before transfer learning, prune after transfer learning, or prune before and after transfer learning) to integrate pruning with transfer learning? The experimental results are presented to quantitatively analyze the impact of different choices to accuracy and training time.

In our experiments, all pruning tasks (using the pre-trained ImageNet models) are completed on a server with 2 Titan X Nvidia GPUs with 12GB on-chip memory each. All transfer learning tasks are conducted on a Dell workstation with one GeForce GTX 1080 GPU that has 8GB on-chip memory.

4.1 Does pruning affect the generalization ability of transfer learning?

The biggest concern to combine pruning and transfer learning is that pruning may damage the generalization ability of a DNN model by removing or masking a great portion of parameters that are considered as less important. To investigate if this is a valid concern, we conduct transfer learning on pruned models with various sparsities and compare the accuracy of transfer learning tasks with their dense counterparts.

For this experiment, we select the standard ResNet34 and VGG16 provided by Pytorch pre-trained on ImageNet, and prune them into 50%, 70%, 80%, and 90% sparsity respectively. For each sparse model we transfer them to both the CIFAR10 and Food101 image classification datasets. To ensure the correctness of the results, all experiments are conducted with the open-source Distiller tool (Zmora et al., 2018), a package especially designed for neural network compression research.

Figure 3 shows the accuracy comparison of transfer learning on dense models (i.e. the orininal model without pruning) and several sparse models with 50%, 70%, 80%, and 90% sparsities. It can be observed that pruned models perform better than or at least equally well when sparsity ratio is 80% or less. This is probably because pruning successfully eliminates some of the redundancy in the original network, which helps to alleviate the over-fitting problem in the dense model. It is also evident that the accuracy of different CNN models and transfer learning tasks show a trend of increasing first and decreasing later between 50% and 80% sparsity. However, the accuracy of transfer learning task is still better than using the dense model when the pruning rate is not over aggressive (e.g. sparsity ratio is greater than 90%). (Han et al., 2015) reported that accuracy dropped when the model is pruned too aggressively. Our findings align well with their results and extend the existing findings by proving that aggressive pruning will hurt the accuracy of transfer learning tasks as well.

Therefore, we can conclude that the concern pruning will hurt the generalization ability of transfer learning is unnecessary. In fact, transfer learning can benefit from a pruned model as long as it is not too aggressively pruned. The generalization ability of a model can be improved by a moderate pruning process, which possibly because the relief of

over-fitting problem.

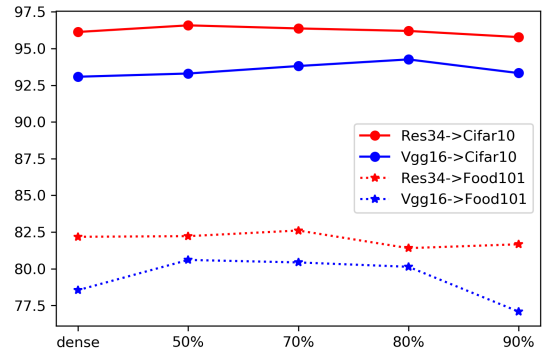

Figure 3. Accuracy comparison among transfer learning on dense model and pruned models with 50%, 70%, 80%, and 90% sparsity respectively. Pruned models perform better than or at least equally well when sparsity ratio is 80% or less.

4.2 Does the order of pruning and transfer learning affect model accuracy?

Transfer learning is usually conducted with the models pre-trained on a large dataset. This is done in part to transfer features that generalize well to a new task or target domain. However, when we consider adding pruning into the transfer learning pipeline, there are multiple options when the model should be pruned, as outlined in Figure 4. Here, "PB", "PA", and "PBA" represent pruning before transfer learning, pruning after transfer learning and prune the model both before and after transfer learning, respectively. A set of experiments are designed to evaluate the performance of each sequence, which are conducted as follows. We keep the results from Section 4.1 and use that as our 'PB' results. We take the baseline dense model that has been transferred to the new dataset and prune it to the same sparsities while fine-tuning on the new dataset. Please note that in all sequences, if a green block (pruning) shows before the orange block (transfer learning), the original domain of images are used as inputs. Similarly, new domain data are used as inputs if the green block shows after the orange block.

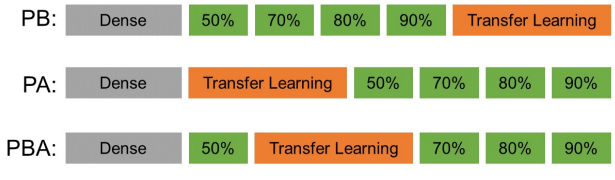

Figure 4. Pruning and transfer learning sequences. Green blocks represent sparsity a DNN model is pruned to. Orange blocks indicate where in the sequence transfer learning occurs.

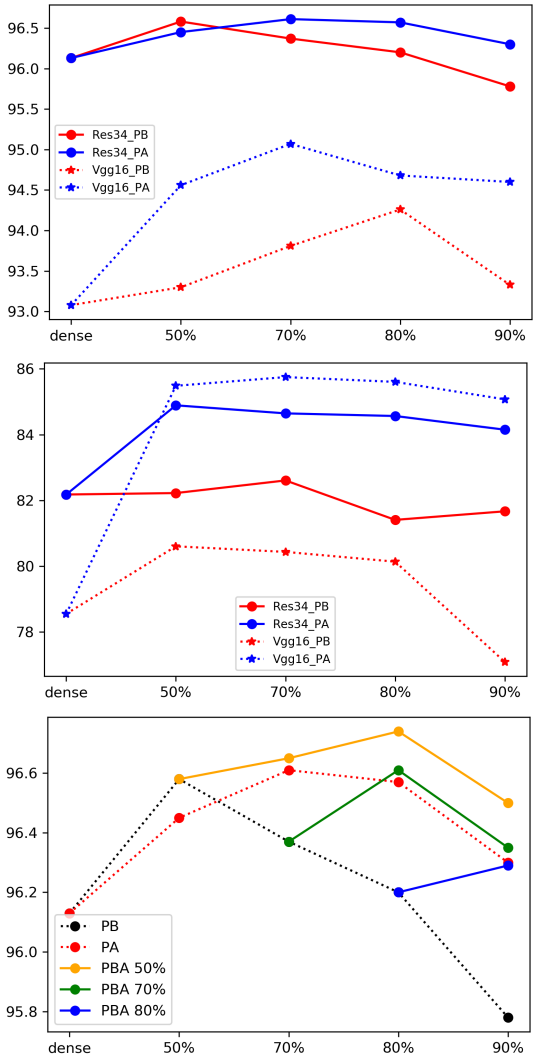

Figure 5. The performance of different pruning and transfer learning sequences. Top: PB/PA on CIFAR10 with ResNet34 and VGG16. Middle: PB/PA on Food101 with ResNet34 and VGG16. Bottom: PBA on CIFAR10 with ResNet34

Figure 5 plots the results of transfer learning on CIFAR10 and Food101 dataset using ResNet34 and VGG16 respectively, both pre-trained on ImageNet. We can observe from the top and middle charts that PA outperforms PB in most cases. The bottom chart of Figure 5 shows the comparison results of PB, PA, and PBA, where the black dash line represents the results of PB for ResNet34 on CIFAR10 dataset and the red dash line shows the corresponding results of PA. The yellow, green and blue solid lines represent the results of different PBA pipelines starting from various pruned ResNet34 models and transferred to CIFAR10 dataset, then further pruned to higher sparsity. It is surprising to see that PBA performs even better than PB and PA. It clearly demonstrates that pruning an already sparse model after transfer learning can further increase its accuracy. Among all viable

Pipeline	P-Time(h)	T-Time(h)	Total(h)	Acc%
$D\|\|T\|\|50\|\|70\|\|80$	5.38	2.44	**7.82**	96.57
$D\|\|50\|\|T\|\|70\|\|80$	186.28	2.44	188.72	**96.74**
$D\|\|50\|\|70\|\|T\|\|80$	367.18	2.44	369.62	96.61
$D\|\|50\|\|70\|\|80\|\|T$	548.08	2.44	550.52	96.20

Table 2. The estimated time on pruning and transfer learning on training the Resnet34_CIFAR10 80% sparsity model. The pipeline names represent the sequence of pruning and transfer learning. For example, "$D\|\|50\|\|T\|\|70\|\|80$" indicates the pipeline starts from a pre-trained dense model (D) and prune it to 50% sparsity then transfer it to CIFAR10 dataset, and then iteratively prune the transferred model to 70% and 80% sparsity. "P-Time(h)" represents the pruning time in hours while the "T-Time" means the transfer learning time in hours.

sparsities, a moderately pruned model with 50% sparsity for PBA achieves the best accuracy.

4.3 What are the trade-offs between accuracy gain and training time cost?

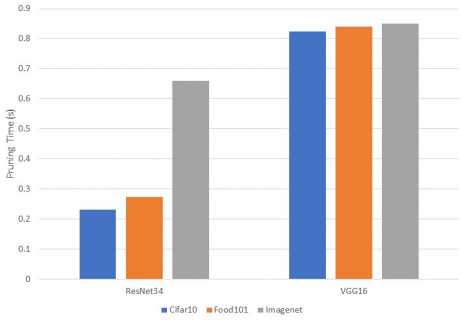

Figure 6. Average pruning time per mini-batch

As discussed in Section 4.2, there are a variety of possible orders to build the prune-transfer pipeline. Since pruning is computationally expensive and time consuming, it will be less ideal if we only use accuracy as the single metric to evaluate which order is the best. For example, Table 2 illustrates the estimated time of several valid pipelines to prune a ResNet34 model (pre-trained on ImageNet) to 80% sparsity and transfer it to CIFAR10. The estimated time is calculated by the average pruning time and transfer learning time for each mini-batch multiplied by the total input size. The results show that the "$D\|\|50\|\|T\|\|70\|\|80$" pipeline has the highest accuracy (96.74%) but needs more than 188 hours to train. Although the accuracy of the "$D\|\|T\|\|50\|\|70\|\|80$" pipeline is slightly lower (96.57%), it is trained much faster (finishes in less than 8 hours). There are two primary reasons why pruning before transfer learning takes substantial amount of time. First, the ImageNet dataset is used as inputs if pruning before transfer learning, which has significantly more images than the CIFAR10 or Food101 dataset (used as inputs if pruning after transfer learning). Second, during the pruning and transfer learning experiments, we notice that the processing time in each mini-batch is longer when pruning with ImageNet dataset as input, even though the hardware and batch size are identical (see Figure 6). All ResNet34 models are pruned on a single GeForce GTX 1080 GPU and all VGG16 models are pruned on a single Titan X GPU. Despite all images are resized to the 224x224 resolution and all mini-batch size is set to 64, we can still observe that the pruning time for each mini-batch is longer on ImageNet than CIFAR10 and Food101. This difference on VGG16 is not as obvious as on ResNet34 because VGG16 has two very large fully connected layers. Therefore, increasing the number of classes on VGG16 has a relatively minimal impact on the mini-batch execution time. However, increasing the class number from 10 to 1000 for ResNet34 has a much larger impact on each mini-batch execution time.

To summarize, the trade-offs between accuracy gain and training time need to be carefully considered when implementing the pruning-transfer learning pipeline. Generally speaking, pruning a well pre-trained model on a large dataset (e.g. ImageNet) endows it better generalization ability. Developers may be able to deploy sparse models with less aggressive pruning to get satisfied accuracy improvements and arithmetical reductions. However, pruning a model trained on large datasets is very time consuming. If the primary goal is to get the highest accuracy or best generalization ability for various transfer learning tasks, we recommend pruning with a less sparse model (e.g. sparsity less than 50%) before transfer learning. Whereas, if training time or training cost is critical (e.g. developers might need to prune and implement multiple sparse models to fulfill development requirements), a better solution would be transfer learning first then pruning, as demonstrated by the "$D\|\|T\|\|50\|\|70\|\|80$" pipeline example in Table 2.

5 Conclusions

In this paper, we systematically study the feasibility of combining pruning and transfer learning. We demonstrate that a moderately pruned model can improve the generalization ability of transfer learning. In addition, the orders of integrating pruning and transfer learning have large impact on the models' accuracy and training time. Our experiments show that pruning a ResNet34 model (pre-trained on ImageNet) to 50% sparsity, transfer it to CIFAR10 dataset, then further prune it to 80% sparsity can increase its accuracy by 0.5% and reduce training time by 3x. The training time can be reduced by 70x with 0.37% better accuracy if we transfer the model first then prune it to 80% sparsity. Furthermore, pruning requires extensive computational resource and elongate training time, the trade-offs between accuracy gain and computation cost should be carefully considered when integrating pruning with transfer learning.

ACKNOWLEDGMENTS

The work reported in this paper is supported by the U.S. National Science Foundation under Grant No. CNS-1908658.

REFERENCES

Bossard, L., Guillaumin, M., and Van Gool, L. Food-101 – mining discriminative components with random forests. In *European Conference on Computer Vision*, 2014.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Frankle, J. and Carbin, M. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018.

Han, S., Pool, J., Tran, J., and Dally, W. Learning both weights and connections for efficient neural network. In *Advances in neural information processing systems*, pp. 1135–1143, 2015.

Hassibi, B. and Stork, D. G. Second order derivatives for network pruning: Optimal brain surgeon. In *Advances in neural information processing systems*, pp. 164–171, 1993.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Huang, G., Sun, Y., Liu, Z., Sedra, D., and Weinberger, K. Q. Deep networks with stochastic depth. In *European conference on computer vision*, pp. 646–661. Springer, 2016.

Karpathy, A. and Fei-Fei, L. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3128–3137, 2015.

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

LeCun, Y., Denker, J. S., and Solla, S. A. Optimal brain damage. In *Advances in neural information processing systems*, pp. 598–605, 1990.

Lin, M., Chen, Q., and Yan, S. Network in network. *arXiv preprint arXiv:1312.4400*, 2013.

Liu, Z., Sun, M., Zhou, T., Huang, G., and Darrell, T. Rethinking the value of network pruning. *arXiv preprint arXiv:1810.05270*, 2018.

Lu, J., Xiong, C., Parikh, D., and Socher, R. Knowing when to look: Adaptive attention via a visual sentinel for image captioning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 375–383, 2017.

Mao, H., Han, S., Pool, J., Li, W., Liu, X., Wang, Y., and Dally, W. J. Exploring the regularity of sparse structure in convolutional neural networks. *arXiv preprint arXiv:1705.08922*, 2017.

Narang, S., Elsen, E., Diamos, G., and Sengupta, S. Exploring sparsity in recurrent neural networks. *arXiv preprint arXiv:1704.05119*, 2017.

Pan, S. J. and Yang, Q. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10): 1345–1359, 2009.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. Automatic differentiation in pytorch. 2017.

See, A., Luong, M.-T., and Manning, C. D. Compression of neural machine translation models via pruning. *arXiv preprint arXiv:1606.09274*, 2016.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Xu, H. and Saenko, K. Ask, attend and answer: Exploring question-guided spatial attention for visual question answering. In *European Conference on Computer Vision*, pp. 451–466. Springer, 2016.

Zhu, M. and Gupta, S. To prune, or not to prune: exploring the efficacy of pruning for model compression. *arXiv preprint arXiv:1710.01878*, 2017.

Zmora, N., Jacob, G., Zlotnik, L., Elharar, B., and Novik, G. Neural network distiller, June 2018. URL https://doi.org/10.5281/zenodo.1297430.