BOOSTING TICKET: TOWARDS PRACTICAL PRUNING FOR ADVERSARIAL TRAINING WITH LOTTERY TICKET HYPOTHESIS

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent research has proposed the lottery ticket hypothesis, suggesting that for a deep neural network, there exist trainable sub-networks performing equally or better than the original model with commensurate training steps. While this discovery is insightful, finding proper sub-networks requires iterative training and pruning. The high cost incurred limits the applications of the lottery ticket hypothesis. We show there exists a subset of the aforementioned sub-networks that converge significantly faster during the training process and thus can mitigate the cost issue. We conduct extensive experiments to show such sub-networks consistently exist across various model structures for a restrictive setting of hyperparameters ($e.g.$, carefully selected learning rate, pruning ratio, and model capacity). As a practical application of our findings, we demonstrate that such sub-networks can help in cutting down the total time of adversarial training, a standard approach to improve robustness, by up to 49% on CIFAR-10 to achieve the state-of-the-art robustness.

1 INTRODUCTION

Pruning has served as an important technique for removing redundant structure in neural networks (Han et al., 2015b;a; Li et al., 2016; He et al., 2017). Properly pruning can reduce cost in computation and storage without harming performance. However, pruning was until recently only used as a post-processing procedure, while pruning at initialization was believed ineffective (Han et al., 2015a; Li et al., 2016). Recently, Frankle & Carbin (2019) proposed the lottery ticket hypothesis, showing that for a deep neural network there exist sub-networks, when trained from certain initialization obtained by pruning, performing equally or better than the original model with commensurate convergence rates. Such pairs of sub-networks and initialization are called winning tickets.

This phenomenon indicates it is possible to perform pruning at initialization. However, finding winning tickets still requires iterative pruning and excessive training. Its high cost limits the application of winning tickets. Although Frankle & Carbin (2019) shows that winning tickets converge faster than the corresponding full models, it is only observed on small networks, such as a convolutional neural network (CNN) with only a few convolution layers. In this paper, we show that for a variety of model architectures, there consistently exist such sub-networks that converge significantly faster when trained from certain initialization after pruning. We call these *boosting tickets*.

We observe the standard technique introduced in Frankle & Carbin (2019) for identifying winning tickets does not always find boosting tickets. In fact, the requirements are more restrictive. We extensively investigate underlining factors that affect such boosting effect, considering three state-of-the-art large model architectures: VGG-16 (Simonyan & Zisserman, 2014), ResNet-18 (He et al., 2016), and WideResNet (Zagoruyko & Komodakis, 2016). We conclude that the boosting effect depends principally on three factors: (i) learning rate, (ii) pruning ratio, and (iii) network capacity; we also demonstrate how these factors affect the boosting effect. By controlling these factors, after only one training epoch on CIFAR-10, we are able to obtain 90.88%/90.28% validation/test accuracy (regularly requires $>$30 training epochs) on WideResNet-34-10 when 80% parameters are pruned.

We further show that the boosting tickets have a practical application in accelerating adversarial training, an effective but expensive defensive training method for obtaining robust models against

adversarial examples. Adversarial examples are carefully perturbed inputs that are indistinguishable from natural inputs but can easily fool a classifier (Szegedy et al., 2013; Goodfellow et al., 2015).

We first show our observations on winning and boosting tickets extend to the adversarial training scheme. Furthermore, we observe that the boosting tickets pruned from a weakly robust model can be used to accelerate the adversarial training process for obtaining a strongly robust model. On CIFAR-10 trained with WideResNet-34-10, we manage to save up to 49% of the total training time (including both pruning and training) compared to the regular adversarial training process.

Our contributions are summarized as follows:

1. We demonstrate that there exists boosting tickets, a special type of winning tickets that significantly accelerate the training process while still maintaining high accuracy.
2. We conduct extensive experiments to investigate the major factors affecting the performance of boosting tickets.
3. We demonstrate that winning tickets and boosting tickets exist for adversarial training scheme as well.
4. We show that pruning a non-robust model allows us to find winning/boosting tickets for a strongly robust model, which enables accelerated adversarial training process.

2 Background and Related Work

2.1 Lottery Ticket Hypothesis

Network pruning has been extensively studied as a method for compressing neural networks and reducing resource consumption (Li et al., 2016; He et al., 2017; LeCun et al., 1990; Han et al., 2015b; Guo et al., 2016; Zhou et al., 2016). However, it was previously believed that pruned networks cannot be trained from the start (Han et al., 2015b; Li et al., 2016).

Surprisingly, recent research has shown it is possible to prune a neural network at the initialization and still reach similar performance as the full model (Liu et al., 2018; Lee et al., 2018). Within this category, the lottery ticket hypothesis (Frankle & Carbin, 2019) states a randomly-initialized dense neural network contains a sub-network that is initialized such that, when trained in isolation, learns as fast as the original network and matches its test accuracy.

In Frankle & Carbin (2019), an iterative pruning method is proposed to find such sub-networks. Specifically, this approach first randomly initializes the model. The initialization is stored separately and the model is trained in the standard manner until convergence. Then a certain proportion of the weights with the smallest magnitudes are pruned while remaining weights are reset to the previously stored initialization and ready to be trained again. This train-prune-reset procedure is performed several times until the target pruning ratio is reached. Using this pruning method, they show the resulting pruned networks can be trained to similar accuracy as the original full networks, which is better than the model with the same pruned structure but randomly initialized.

One of the limitations of the lottery ticket hypothesis, as pointed in (Frankle et al., 2019), is that winning tickets are found by unstructured pruning which does not necessarily yield faster training or executing time. In addition, finding winning tickets requires training the full model beforehand, which is time-consuming as well, especially considering iterative pruning. In this paper, we aim to show that there exists a subset of winning tickets, namely boosting tickets, that not only performs equally well as the original model but also converges much faster.

2.2 Adversarial Examples

Given a classifier $f : \mathcal{X} \to \{1, \ldots, k\}$ for an input $\mathbf{x} \in \mathcal{X}$, an adversarial example $\mathbf{x}_{\text{adv}}$ is a perturbed version of $\mathbf{x}$ such that $\mathcal{D}(\mathbf{x}, \mathbf{x}_{\text{adv}}) < \epsilon$ for some small $\epsilon > 0$, yet being mis-classified as $f(\mathbf{x}) \neq f(\mathbf{x}_{\text{adv}})$. $\mathcal{D}(\cdot, \cdot)$ is some distance metric which is often an ℓ_p metric, and in most of the literature ℓ_∞ metric is considered, so as in this paper.

The procedure of constructing such adversarial examples is often referred to as adversarial attacks. One of the simplest attacks is a single-step method, Fast Gradient Sign Method (FGSM) (Goodfellow et al., 2015), manipulating inputs along the direction of the gradient with respect to the outputs: $\mathbf{x}_{\text{adv}} = \Pi_{\mathbf{x}+\mathcal{S}}(\mathbf{x} + \alpha(\nabla_{\mathbf{x}} L(\theta, \mathbf{x}, y))$, where $\Pi_{\mathbf{x}+\mathcal{S}}$ is the projection operation that ensures adversarial examples stay in the ℓ_p ball $\mathcal{S}$ around $\mathbf{x}$. Although this method is fast, the attack is weak and can be defended easily. On the other hand, its multi-step variant, Projected Gra-

dient Descend (PGD), is one of the strongest attacks (Kurakin et al., 2016; Madry et al., 2017): $\mathbf{x}_{\text{adv}}^{t+1} = \Pi_{\mathbf{x}+\mathcal{S}}(\mathbf{x}_{\text{adv}}^{t} + \alpha(\nabla_{\mathbf{x}} L(\theta, \mathbf{x}, y))$, where $\mathbf{x}$ is initialized with a random perturbation. Since PGD requires to access the gradients for multiple steps, it will incur high computational cost.

On the defense side, currently the most successful defense approach is constructing adversarial examples via PGD during training and add them to the training sets as data augmentation, which is referred to as adversarial training (Madry et al., 2017). One caveat of adversarial training is its computational cost due to performing PGD attacks at each training step. Alternatively, using FGSM during training is much faster but the resulting model is robust against FGSM attacks but vulnerable against PGD attacks (Kurakin et al., 2016). In this paper, we show it is possible to combine the advantages of both and quickly train a strongly robust model benefited from the boosting tickets.

2.3 Connecting robustness and compactness

Prior studies have shown success in achieving both compactness and robustness of the trained networks (Guo et al., 2018; Ye et al., 2018; Zhao et al., 2018; Dhillon et al., 2018; Sehwag et al., 2019; Wijayanto et al., 2019). However, most of them will either incur much higher training cost or sacrifice robustness from the full model. On the contrary, our approach is able to reduce training time while obtaining similar/higher robust accuracy than the original full network.

3 Empirical Study of Boosting tickets

We first investigate boosting tickets on the standard setting without considering adversarial robustness. In this section, we show that with properly chosen hyperparameters, we are managed to find boosting tickets on VGG-16 and ResNet that can be trained much faster than the original dense network. Detailed model architectures and the setup can be found in Supplementary Section A.

3.1 Existence of Boosting Tickets

To find the boosting tickets, we use a similar algorithm for finding winning tickets, which is briefly described in the previous section and will be detailed here. First, a neural network is randomly initialized and saved in advance. Then the network is trained until convergence, and a given proportion of weights with the smallest magnitudes are pruned, resulting in a mask where the pruned weights indicate 0 and remained weights indicate 1. We call this train-and-prune step *pruning*. This mask is then applied to the saved initialization to obtain a sub-network, which are the boosting tickets. All of the weights that are pruned (where zeros in the mask) will remain to be 0 during the whole training process. Finally, we can retrain the sub-networks.

The key differences between our algorithm and the one proposed in Frankle & Carbin (2019) to find winning tickets are (i) we use a small learning rate for pruning and retrain the sub-network (tickets) with learning rate warm-up from this small learning rate. In particular, for VGG-16 we choose 0.01 for pruning and warmup from 0.01 to 0.1 for retraining; for ResNet-18 we choose 0.05 for pruning and warmup from 0.05 to 0.1 for retraining; (ii) we find it is sufficient to prune and retrain the model only once instead of iterative pruning for multiple times. In Supplementary Section B, we show the difference of boosting effects brought from the tickets found by iterative pruning and one-shot pruning is negligible. Note warmup is also used in Frankle & Carbin (2019). However, they propose to use warmup from small learning rate to a large one during pruning as well, which hinders the boosting effect as shown in the following experiments.

First, we show the existence of boosting tickets for VGG-16 and ResNet-18 on CIFAR-10 in Figure 1 and compare to the winning tickets. In particular, we show the boosting tickets are winning tickets, in the sense that they outperform the randomly initialized models. When compared to the winning tickets, boosting tickets demonstrate equally good performance with a higher convergence rate. Similar results on MNIST can be found in Supplementary Section C.

To measure the overall convergence rate, early stopping seems to be a good fit in the literature. It is commonly used to prevent overfitting and the final number of steps are used to measure convergence rates. However, early stopping is not compatible with learning rate scheduling we used in our case where the total number of steps is determined before training.

This causes two issues in our evaluation in Figure 1: (i) Although the boosting tickets reach a relatively high validation accuracy much earlier than the winning ticket, the training procedure is

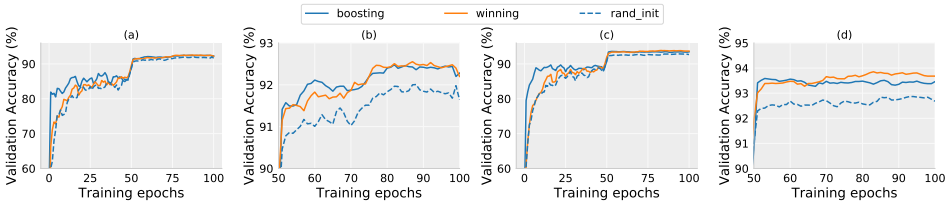

Figure 1: Validation accuracy during the training process on VGG-16 (a, b) and ResNet-18 (c, d) for winning tickets, boosting tickets, and randomly initialized weights. In both models, the boosting tickets show faster convergence rate and equally good performance as the winning tickets.

then hindered by the large learning rate. After the learning rate drops, the performance gap between boosting tickets and winning tickets becomes negligible. As a result, the learning rate scheduling obscures the improvement on convergence rates of boosting tickets; (ii) Due to fast convergence, boosting tickets tend to overfit, as observed in ResNet-18 after 50 epochs.

To mitigate these two issues without excluding learning rate scheduling, we conduct another experiment where we mimic the early stopping procedure by gradually increasing the total number of epochs from 20 to 100. The learning rate is still dropped at the 50% and 75% stage. In this way, we can better understand the speed of convergence without worrying about overfitting even with learning rate scheduling involved. In figure 2, we compare the boosting tickets and winning tickets in this manner on VGG-16.

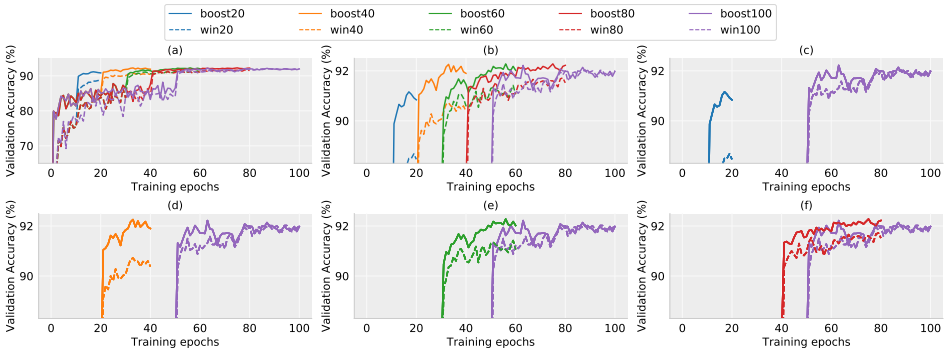

Figure 2: Validation accuracy when the total number of epochs are 20, 40, 60, 80, 100 for both the boosting tickets (straight lines) and winning tickets (dash lines) on VGG-16. Plot (a) and (b) contains the validation accuracy for all the training epochs in different scales. Plot (c,d,e,f) compare the validation accuracy between models trained for fewer epochs and the one for 100 epochs.

While the first two plots in Figure 2 show the general trend of convergence, the improvement of convergence rates is much clearer in the last four plots. In particular, the validation accuracy of boosting tickets after 40 epochs is already on pair with the one trained for 100 epochs. Meanwhile, the winning tickets fall much behind the boosting tickets until 100 epochs where two finally match.

We further investigate the test accuracy at the end of training for boosting and winning tickets in Table 1. We find the test accuracy of winning tickets gradually increase as we allow for more training steps, while the boosting tickets achieve the highest test accuracy after 60 epochs and start to overfit at 100 epochs.

Table 1: Final test accuracy of winning tickets and boosting tickets trained in various numbers of epochs on VGG-16.

# of Epochs	20	40	60	80	100
Test Accuracy on Winning Tickets (%)	88.10	90.03	90.96	91.79	92.00
Test Accuracy on Boosting Tickets (%)	91.25	91.84	92.13	92.14	92.05

Summarizing the observations above, we confirm the existence of boosting tickets and state the boosting ticket hypothesis: *A randomly initialized dense neural network contains a sub-network that is initialized such that, when trained in isolation, converges faster than the original network and other winning tickets while matches their performance.*

In the following sections, we investigate three major components that affect the boosting effects.

3.2 LEARNING RATE

As finding boosting tickets requires alternating learning rates, it is natural to assume the performance of boosting tickets relies on the choice of learning rate. Thus, we extensively investigate the influence of various learning rates.

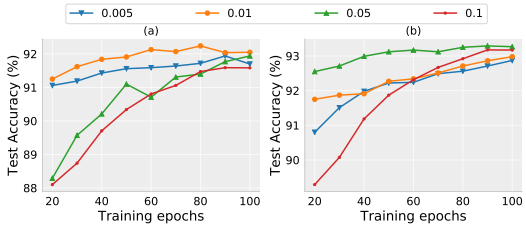

Figure 3: The final test accuracy achieved when total number of epochs vary from 20 to 100 on four different tickets. Each line denotes one winning ticket found by learning rate 0.005, 0.01, 0.05, and 0.1 for VGG-16 (a) and ResNet-18 (b).

We use similar experimental settings in the previous section, where we increase the total number of epochs gradually and use the test accuracy as a measure of convergence rates. We choose four different learning rates 0.005, 0.01, 0.05 and 0.1 for pruning to get the tickets. All of the tickets found by those learning rates obtain the accuracy improvement over randomly reinitialized sub-model and thus satisfy the definition of winning tickets (i.e., they are all winning tickets).

As shown in the first two plots of Figure 3, tickets found by smaller learning rates tend to have stronger boosting effects. For both VGG-16 and ResNet-18, the models trained with learning rate 0.1 show the least boosting effects, measured by the test accuracy after 20 epochs of training. On the other hand, training with too small learning rate will compromise the eventual test accuracy at a certain extent. Therefore, we treat the tickets found by learning rate 0.01 as our boosting tickets for VGG-16, and the one found by learning rate 0.05 as for ResNet-18, which converge much faster than all of the rest while achieving the highest final test accuracy.

3.3 PRUNING RATIO

Pruning ratio has been an important component for winning tickets (Frankle & Carbin, 2019), and thus we investigate its effect on boosting tickets. Since we are only interested in the boosting effect, we use the validation accuracy at early stages as a measure of the strength of boosting to avoid drawing too many lines in the plots. In Figure 4, we show the validation accuracy after the first and fifth epochs of models for different pruning ratios for VGG-16 and ResNet-18.

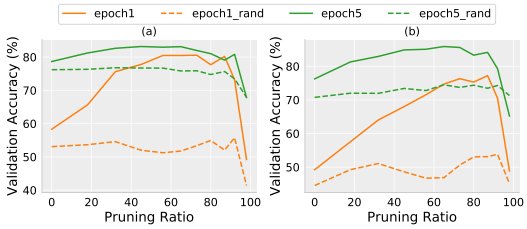

Figure 4: Under various pruning ratios, the changes of validation accuracy after the first and fifth training epoch, trained from the original initialized weights of boosting tickets and randomly reinitialized ones for VGG-16 (a) and ResNet-18 (b).

For both VGG-16 and ResNet-18, boosting tickets always reach much higher accuracy than randomly reinitialized sub-models, demonstrating their boosting effects. When the pruning ratio falls into the range from 60% to 90%, boosting tickets can provide the strongest boosting effects which obtain around 80% and 83% validation accuracy after the first and the fifth training epochs for VGG-16 and obtain 76% and 85% validation accuracy for ResNet-18. On the other hand, the increase of validation accuracy between the first training epoch and the fifth training epoch become smaller when boosting effects appear. It indicates their convergence starts to saturate due to the large learning rate at the initial stage and is ready for dropping the learning rate.

3.4 MODEL CAPACITY

We finally investigate how model capacity, including the depth and width of models, affects the boosting tickets. We use WideResNet (Zagoruyko & Komodakis, 2016) either with its depth or width fixed and vary the other factor. In particular, we keep the depth as 34 and increases the width from 1 to 10, comparing their boosting effect. Then we keep the width as 10 and increase the depth from 10 to 34. The changes of validation accuracy of the models are shown in Figure 5.

Overall, Figure 5 shows models with larger capacity have a more significant boosting effect, though the boosting effects keep the same when the depth is larger than 22. Notably, we find the largest model WideResNet-34-10 achieves 90.88% validation accuracy after only one training epoch.

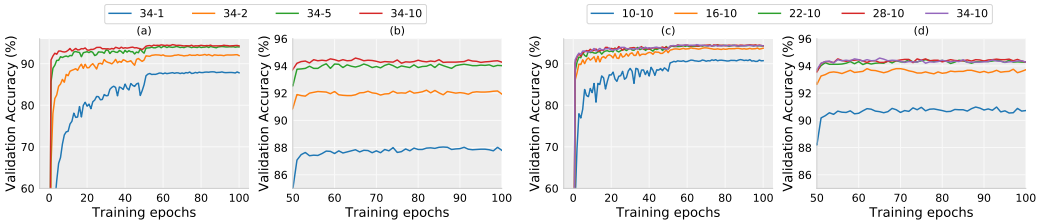

Figure 5: Plot (a) and (b) correspond to boosting tickets for various of model widths. Plot (c) and (d) correspond to boosting tickets for various of model depths. While a wider model always boosts faster, deep models have similar boosting effect when the depth is large enough.

4 Boosting Tickets in Adversarial Settings

Although the lottery ticket hypothesis is extensively studied in Frankle & Carbin (2019) and Frankle et al. (2019), the same phenomenon in adversarial training setting lacks thorough understanding.

In this section, we show two important facts that make boosting tickets suitable for the adversarial scheme: (1) the lottery ticket hypothesis and boosting ticket hypothesis are applicable to the adversarial training scheme; (2) pruning on a weakly robust model allows to find the boosting ticket for a strongly robust model and save training cost.

4.1 Applicability for Adversarial Training

In the following experiment, we use a naturally trained model, that is trained in the standard manner, and two adversarially trained models using FGSM and PGD respectively to obtain the tickets by pruning these models. Then we retrain these pruned models with the same PGD-based adversarial training from the same initialization. In Figure 6, we report the corresponding accuracy on the original validation sets and on the adversarially perturbed validation examples, noted as clean accuracy and robust accuracy. We further train the pruned model from random reinitialization to validate lottery ticket hypothesis.

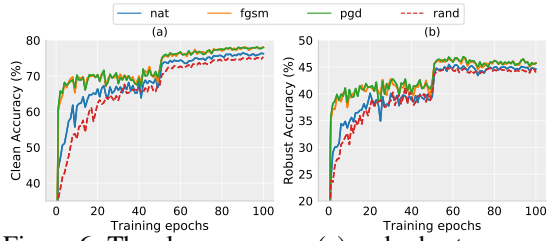

Figure 6: The clean accuracy (a) and robust accuracy (b) of pruned models on the validation set. The models are pruned based on different training methods (natural training, FGSM-based adversarial training, and PGD-based adversarial training). For each obtained boosting ticket, it is retrained with PGD-based adversarial training with 100 training epochs.

Unless otherwise stated, in all the PGD-based adversarial training, we keep the same setting as Madry et al. (2017). The PGD attacks are performed in 10 steps with step size $2/255$ (PGD-10). The PGD attacks are bounded by $8/255$ in its ℓ_∞ norm. For the FGSM-based adversarial training, the FGSM attacks are bounded by $8/255$.

Both models trained from the boosting tickets obtained with FGSM- and PGD-based adversarial training demonstrate superior performance and faster convergence than the model trained from random reinitialization. This confirms the lottery ticket hypothesis and boosting ticket hypothesis are applicable to adversarial training scheme on both clean accuracy and robust accuracy. More interestingly, the performance of the models pruned with FGSM- and PGD-based adversarial training are almost the same. This observation suggests it is sufficient to train a weakly robust model with FGSM-based adversarial training for obtaining the boosting tickets and retrain it with stronger attacks such as PGD.

This finding is interesting because FGSM-based adversarial trained models will suffer from label leaking problems as learning weak robustness Kurakin et al. (2016). In fact, the FGSM-based adversarially trained model from which we obtain our boosting tickets has 89% robust accuracy against FGSM but with only 0.4% robust accuracy against PGD performed in 20 steps (PGD-20). However, Figure 6 shows the following PGD-based adversarial retraining on the boosting tickets obtained from that FGSM-based trained model is indeed robust. Further discussions can be found in Section 5.

In Ye et al. (2019), the authors argued that the lottery ticket hypothesis fails to hold in adversarial training via experiments on MNIST. We show they fail to observe winning tickets because the models they used have limited capacity. In the adversarial setting bounded by $L_\infty \leq 0.3$, small models such as a CNN with two convolutional layers used in Ye et al. (2019) can not yield even winning tickets when pruning ratio is large. In Figure 7, plot (a) and (b) are the clean and robust accuracy of the pruned models when the pruning ratio is 80%. The pruned model degrades into a trivial classifier where all example are classified into the same class with $11.42\%/11.42\%$ valid/test accuracy. However, when we use VGG-16, as shown in plot (c) and (d), the winning tickets are found again. This can be explained as adversarial training requires much larger model capacity than standard training (Madry et al., 2017), thus pruning small models could undermine their performance. Since MNIST is a simple dataset, adversarial training converges quickly at the first few epochs for both the tickets and randomly initialized models. Therefore, there is no winning tickets performing obvious boosting effect which we can identify as a boosting ticket on MNIST.

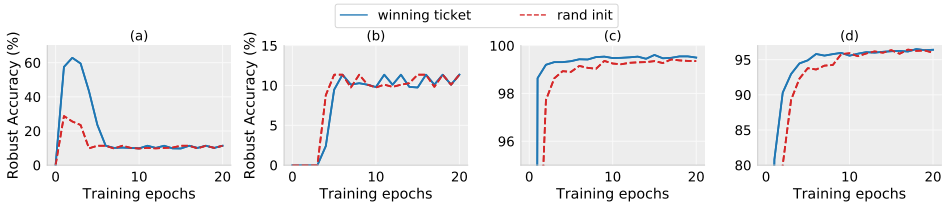

Figure 7: We show clean (a,c) and robust accuracy (b,d) for both winning tickets and randomly initialized weights on LeNet (a,b) and Vgg-16 (c,d) on MNIST with adversarial training.

4.2 Convergence Speedup

We then conduct the same experiment as in Figure 2 but in the adversarial training setting to better show the improved convergence rates. The results for validation accuracy and test accuracy are presented in Figure 8 and Table 2 respectively. It suggests it is sufficient to train 60 epochs to achieve similar robust accuracy as the full model trained for 100 epochs.

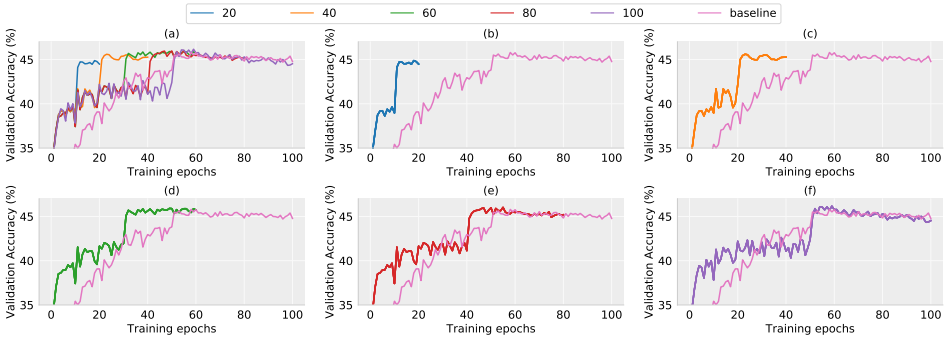

Figure 8: Validation robust accuracy of pruned models with PGD-based adversarial training on VGG-16 where the total number of epochs are 20, 40, 60, 80, 100 respectively. Plot (a) and (b) show all the results while plot (c), (d), (e), (f) compare each model with the baseline model. The baseline model is obtained by 100-epoch PGD-based adversarial training on the original full model.

Table 2: Best test clean and robust accuracy for PGD-based adversarial training on boosting tickets obtained by FGSM-based adversarial training in various numbers of epochs on VGG-16. Baseline model is obtained by 100-epoch PGD-based adversarial training on original full model.

# of Epochs	20	40	60	80	100	Baseline
Robust Test Accuracy	44.49	45.27	**45.73**	45.20	44.53	44.78
Clean Test Accuracy	75.15	76.28	76.48	77.60	**78.07**	77.21

4.3 Boosting Ticket Applications on adversarially trained WideResNet-34-10

Until now, we have confirmed that boosting tickets exist consistently across different models and training schemes and convey important insights on the behavior of pruned models. However, in the natural training setting, although boosting tickets provide faster convergence, it is not suitable for accelerating the standard training procedure as pruning to find the boosting tickets requires training

full models beforehand. On the other hand, the two observations mentioned in Section 4 enable boosting tickets to accelerate adversarial training. In particular, we can find boosting tickets with FGSM-based adversarial training that and they can significantly accelerate the PGD-based adversarial training. Note that the cost of FGSM-based training is only 1/10 times of the standard 10-step PGD-based one and thus is almost negligible compared to the time saved due to the boosting effect.

In Table 3, we apply adversarial training to WideResNet-34-10, which has the same structure used in Madry et al. (2017), with the proposed approach for 40, 70 and 100 epochs and report the best accuracy/robust accuracy under various attacks among the whole training process. In particular, we perform 20-step PGD, 100-step PGD as white-box attacks where the attackers have the access to the model parameters. More experimental results are included in the Appendix.

Table 3: Best test clean accuracy, robust accuracy, and training time for PGD-based adversarial training on boosting tickets obtained by FGSM-based one in various numbers of epochs on WideResNet-34-10. Overall, our training strategy based on boosting tickets can save up to 49% of the total training time while performing better compared to regular adversarial training on the full model.

	Test Accuracy(%)			Consumed Time(s)			
Models	Clean	PGD-20	PGD-100	Pruning	Training	Total	Ratio
Madry's	86.21	50.07	49.32	-	134,764	134,764	-
Ours-40	87.72	50.37	49.28	15,462	54,090	**69,552**	**0.51**
Ours-70	**87.85**	**50.48**	**49.58**	15,462	94,796	110,258	0.82
Ours-100	87.35	49.92	49.11	15,462	137,105	152,567	1.13

We report the time consumption for training each model to measure how much time is saved by boosting tickets. We run all the experiments on a workstation with 2 V100 GPUs in parallel. From Table 3 we observe that while our approach requires pruning before training, it is overall faster as it uses FGSM-based adversarial training. In particular, to achieve its best robust accuracy, original Madry et al.'s training method (Madry et al., 2017) requires 134,764 seconds on WideResNet-34-10. To achieve that, our boosting ticket only requires 69,552 seconds, including 15,462 seconds to find the boosting ticket and 54,090 seconds to retrain the ticket, saving 49% of the total training time.

5 Discussion and Future Work

Not knowledge distillation. It may seem that winning tickets and boosting tickets behave like knowledge distillation (Ba & Caruana, 2014; Hinton et al., 2015) where the learned knowledge from a large model is transferred to a small model. This conjecture may explain the boosting effects as the pruned model quickly recover the knowledge from the full model. However, the lottery ticket framework seems to be distinctive to knowledge distillation. If boosting tickets simply transfer knowledge from the full model to the pruned model, then an FGSM-based adversarially trained model should not find tickets that improves the robustness of the sub-model against PGD attacks, as the full model itself is vulnerable to PGD attacks. Yet in Section 4.1 we observe an FGSM-based adversarially trained model still leads to boosting tickets that accelerates PGD-based adversarial training. We believe the cause of boosting tickets requires further investigation in the future.

Accelerate adversarial training. Recently, Zhang et al. (2019); Shafahi et al. (2019) propose to reduce the training time for PGD-based adversarial training by recycling the gradients computed for parameter updates and constructing adversarial examples. While their approach focuses on reducing the computational time for each epoch, our method focuses more on the convergence rate (i.e., reduce the number of epochs required for convergence). Therefore, our approach is compatible with theirs, making it a promising future direction to combine both to further reduce the training time.

6 Conclusion

In this paper, we investigate boosting tickets, sub-networks coupled with certain initialization that can be trained with significantly faster convergence rate. As a practical application, in the adversarial training scheme, we show pruning a weakly robust model allows to find boosting tickets that can save up to 49% of the total training time to obtain a strongly robust model that matches the state-of-the-art robustness. Finally, it is an interesting direction to investigate whether there is a way to find boosting tickets without training the full model beforehand, as it is technically not necessary.

REFERENCES

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

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, pp. 39–57. IEEE, 2017.

Guneet S Dhillon, Kamyar Azizzadenesheli, Zachary C Lipton, Jeremy Bernstein, Jean Kossaifi, Aran Khanna, and Anima Anandkumar. Stochastic activation pruning for robust adversarial defense. *arXiv preprint arXiv:1803.01442*, 2018.

Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rJl-b3RcF7.

Jonathan Frankle, Gintare Karolina Dziugaite, Daniel M Roy, and Michael Carbin. The lottery ticket hypothesis at scale. *arXiv preprint arXiv:1903.01611*, 2019.

Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015. URL http://arxiv.org/abs/1412.6572.

Yiwen Guo, Anbang Yao, and Yurong Chen. Dynamic network surgery for efficient dnns. In *Advances In Neural Information Processing Systems*, pp. 1379–1387, 2016.

Yiwen Guo, Chao Zhang, Changshui Zhang, and Yurong Chen. Sparse dnns with improved adversarial robustness. In *Advances in neural information processing systems*, pp. 242–251, 2018.

Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015a.

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*, pp. 1135–1143, 2015b.

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*, pp. 770–778, 2016.

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*, pp. 1389–1397, 2017.

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

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. pp. 448–456, 2015. URL http://jmlr.org/proceedings/papers/v37/ioffe15.pdf.

Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016.

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

Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Namhoon Lee, Thalaiyasingam Ajanthan, and Philip HS Torr. Snip: Single-shot network pruning based on connection sensitivity. *arXiv preprint arXiv:1810.02340*, 2018.

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

Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks. *arXiv preprint arXiv:1611.02770*, 2016.

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

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*, 2016.

Vikash Sehwag, Shiqi Wang, Prateek Mittal, and Suman Jana. Towards compact and robust deep neural networks. *arXiv preprint arXiv:1906.06110*, 2019.

Ali Shafahi, Mahyar Najibi, Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! *arXiv preprint arXiv:1904.12843*, 2019.

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

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Arie Wahyu Wijayanto, Jun Jin Choong, Kaushalya Madhawa, and Tsuyoshi Murata. Towards robust compressed convolutional neural networks. In *2019 IEEE International Conference on Big Data and Smart Computing (BigComp)*, pp. 1–8. IEEE, 2019.

Shaokai Ye, Siyue Wang, Xiao Wang, Bo Yuan, Wujie Wen, and Xue Lin. Defending dnn adversarial attacks with pruning and logits augmentation. 2018.

Shaokai Ye, Kaidi Xu, Sijia Liu, Hao Cheng, Jan-Henrik Lambrechts, Huan Zhang, Aojun Zhou, Kaisheng Ma, Yanzhi Wang, and Xue Lin. Second rethinking of network pruning in the adversarial setting. *arXiv preprint arXiv:1903.12561*, 2019.

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

Dinghuai Zhang, Tianyuan Zhang, Yiping Lu, Zhanxing Zhu, and Bin Dong. You only propagate once: Painless adversarial training using maximal principle. *arXiv preprint arXiv:1905.00877*, 2019.

Yiren Zhao, Ilia Shumailov, Robert Mullins, and Ross Anderson. To compress or not to compress: Understanding the interactions between adversarial attacks and neural network compression. *arXiv preprint arXiv:1810.00208*, 2018.

Hao Zhou, Jose M Alvarez, and Fatih Porikli. Less is more: Towards compact cnns. In *European Conference on Computer Vision*, pp. 662–677. Springer, 2016.

A Model Architectures and Setup

Setup. We use BatchNorm Ioffe & Szegedy (2015), weight decay, decreasing learning rate schedules ($\times 0.1$ at 50% and 75%), and augmented training data for training models. We try to keep the setting the same as the one used in Frankle & Carbin (2019) except we use one-shot pruning instead of iterative pruning. It allows the whole pruning and training process to be more practical in real applications. On CIFAR-10 dataset, we randomly select 5,000 images out of 50,000 training set as validation set and train the models with the rest. The reported test accuracy is measured with the whole testing set.

All of our experiments are run on four Tesla V100s, 10 Tesla P100s, and 10 2080 Tis. For all the time-sensitive experiments like adversarial training on WideResNet-34-10 in Section 4.3, we train each model on two Tesla V100s with data parallelism. For the rest ones measuring the final test accuracy, we use one gpu for each model without parallelism.

In Table 4, we summarize the number of parameters and parameter sizes of all the model architectures that we evaluate with including VGG-16 Simonyan & Zisserman (2014), ResNet-18 He et al. (2016), and the variance of WideResNets Zagoruyko & Komodakis (2016).

Table 4: Number of parameters and parameter sizes for various architectures.

	# of Parameters	Size (MB)
VGG-16	29,975,444	114.35
ResNet-18	11,173,962	42.63
WideResNet-34-10	46,160,474	176.09
WideResNet-28-10	36,479,194	139.16
WideResNet-22-10	26,797,914	102.23
WideResNet-16-10	17,116,634	65.29
WideResNet-10-10	7,435,354	28.36
WideResNet-34-5	11,554,074	44.08
WideResNet-34-2	1,855,578	7.08
WideResNet-34-1	466,714	1.78

B Iterative Pruning vs One Shot Pruning

In Figure 9, we track the training of models obtained from both iterative pruning and one shot pruning. We find the performance of both, in terms of the boosting effects and final accuracy, is indistinguishable.

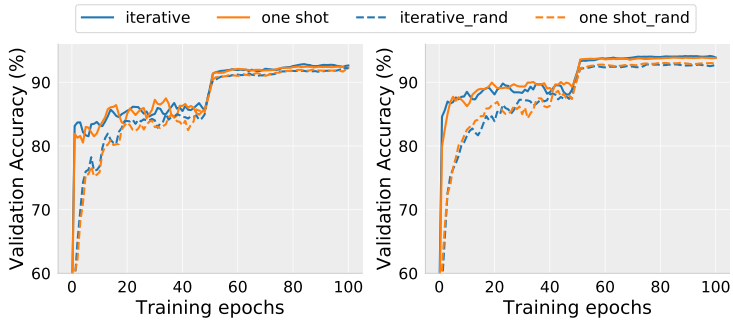

Figure 9: We compare tickets obtained via iterative pruning and one shot pruning on VGG-16 (left) and ResNet-18 (right). We plot the validation accuracy of models from both approaches and the corresponding randomly initialized models.

C EXPERIMENTS ON MNIST

In this section, we report experiment results on MNIST (LeCun et al., 1998) for the standard setting, where we use LeNet (LeCun et al., 1998) with two convolutions and two fully connected layers for the classification task.

As for MNIST we do not use learning rate scheduling, early stopping is then used to determine the speed of convergence. In 5, we report the epochs when early stopping happens and the test accuracy to illustrate the existence of boosting tickets for MNIST. While winning tickets converge at the 18th epoch, boosting tickets converge at the 11th epoch, indicating faster convergence.

Table 5: The epochs when early stopping happens and the corresponding accuracy for the full model, winning tickets, boosting tickets, and randomly initialized model based on LeNet with two convolutional layers and two fully connected layers.

	Full Model	Winning	Boosting	Rand Init
Early Stopping	20	18	11	16
Test Accuracy	99.18	99.24	99.23	98.97

D ADDITIONAL ROBUSTNESS EVALUATION

It might be suspicious if the resulting models from pruning and adversarial training are indeed robust against strong attacks, as the pruning mask is obtained from a weakly robust model. We conduct extensive experiments on CIFAR-10 with WideResNet-34-10 to evaluate the robustness of this model and compare to the robust model trained with Madry et al's method Madry et al. (2017). In addition to the experimental results reported in the main text, in Table 6 we include results for C&W attacks Carlini & Wagner (2017) and transfer attacks Papernot et al. (2016); Liu et al. (2016) where we attack one model with adversarial examples found by 20-step PGD based on other models.

We find the adversarial examples generated from one model can transfer to another model with a slight decrease on the robust error. This indicates our models and Madry's model share adversarial examples and further share decision boundaries.

Table 6: Best test clean accuracy (the first row), robust accuracy (the second to fourth rows), transfer attack accuracy (the middle four rows), and training time for PGD-based adversarial training (the last four rows) on boosting tickets obtained by FGSM-based adversarial training in various of numbers of epochs on WideResNet-34-10. Overall, our adversarial training strategy based on boosting tickets is able to save up to 49% of the total training time while achieving higher robust accuracy compared to the regular adversarial training on the original full model.

	Test Accuracy(%)			
Models	Madry's	Ours-40	Ours-70	Ours-100
Natural	86.21	87.72	**87.85**	87.35
PGD-20	50.07	50.37	**50.48**	49.92
PGD-100	49.32	49.28	**49.58**	49.11
C&W	50.46	**50.92**	50.82	50.37
Madry's	-	58.16	57.39	57.63
Ours-40	58.69	-	54.04	56.11
Ours-70	58.77	54.60	-	55.23
Ours-100	58.61	56.62	55.20	-
Pruning Time(s)	0	15,462	15,462	15,462
Training Time(s)	134,764	54,090	94,796	137,105
Total Time(s)	134,764	**69,552**	110,258	152,567
Ours/Madry's	-	**0.51**	0.82	1.13