The Lottery Ticket Hypothesis for Object Recognition

Sharath Girish* sgirish@cs.umd.edu
Shishira R Maiya* shishira@umd.edu
Kamal Gupta kampta@umd.edu
Hao Chen chenh@umd.edu
Larry Davis lsd@umiacs.umd.edu
Abhinav Shrivastava abhinav@cs.umd.edu
University of Maryland, College Park

Abstract

Recognition tasks, such as object recognition and keypoint estimation, have seen widespread adoption in recent years. Most state-of-the-art methods for these tasks use deep networks that are computationally expensive and have huge memory footprints. This makes it exceedingly difficult to deploy these systems on low power embedded devices. Hence, the importance of decreasing the storage requirements and the amount of computation in such models is paramount. The recently proposed Lottery Ticket Hypothesis (LTH) states that deep neural networks trained on large datasets contain smaller subnetworks that achieve on par performance as the dense networks. In this work, we perform the first empirical study investigating LTH for model pruning in the context of object detection, instance segmentation, and keypoint estimation. Our studies reveal that lottery tickets obtained from Imagenet pretraining do not transfer well to the downstream tasks. We provide guidance on how to find lottery tickets with up to 80% overall sparsity on different sub-tasks without incurring any drop in the performance. Finally, we analyse the behavior of trained tickets with respect to various task attributes such as object size, frequency, and difficulty of detection.

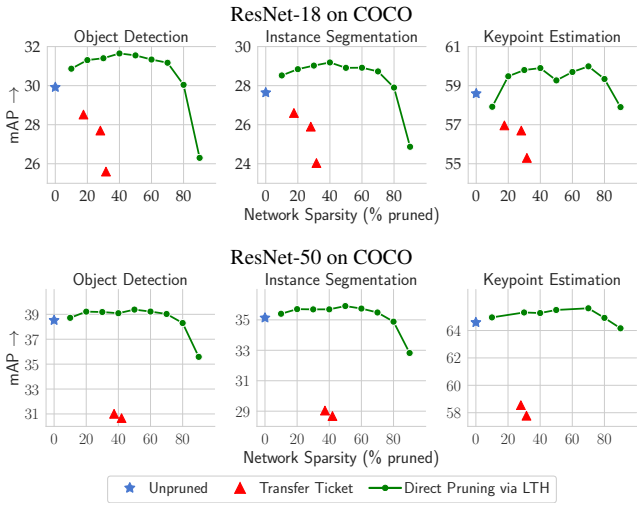

Figure 1: Performance of lottery tickets discovered using direct pruning for various object recognition tasks. Here we have used a Mask R-CNN model with ResNet-18 backbone (top) and ResNet-50 backbone (bottom) to train models for object detection, segmentation and human keypoint estimation on the COCO dataset. We show the performance of the baseline dense network, the sparse subnetwork obtained by transferring ImageNet pre-trained "universal" lottery tickets, as well as the subnetwork obtained by task-specific pruning. Task-specific pruning outperforms the universal tickets by a wide margin. For each of the tasks, we can obtain the same performance as the original dense networks with only 20% of the weights.

1. Introduction

Recognition tasks, such as object detection, instance segmentation, and keypoint estimation, have emerged as canonical tasks in visual recognition because of their intuitive appeal and pertinence in a wide variety of real-world problems. The modus operandi followed in nearly all state-of-the-art visual recognition methods is the following: (i) Pre-train a large neural network on a very large and diverse image classification dataset, (ii) Append a small task-specific network to the pre-trained model and fine-tune the weights jointly on a much smaller dataset for the task. The introduction of ResNets by He *et al*. [22] made the training of very deep networks possible, helping in scaling up model capacity, both in terms of depth and width, and became a well-established instrument for improving the performance of deep learning models even with smaller datasets [25]. As a result, the past few years have seen increasingly large neural network architectures [35, 55, 47, 23], with sizes often exceeding the memory limits of a single hardware accelerator. In recent years, efforts towards reducing the memory and computation footprint of deep networks have followed three seemingly parallel tracks with common objectives: weight quantization, sparsity via regularization, and network pruning; Weight Quantization [24, 16, 44, 6, 30]

*Equal contribution.

methods either replace weights of a trained neural network with lower precision or arithmetic operations with bit-wise operations to reduce the memory up to an order of magnitude. Regularization approaches, such as dropout [45, 2] or LASSO [48], attempt to discourage an over-parameterized network from relying on a large number of features and encourage learning a sparse and robust predictor. Both quantization and regularization approaches are effective in reducing the number of weights in a network or the memory footprint, but usually at the cost of increased error rates [20, 30]. In comparison, pruning approaches [28, 19] disentangle the learning task from pruning by alternating between weight optimization and weight deletion. The recently proposed Lottery Ticket Hypothesis ·(LTH) [12] falls in this category.

According to LTH, an over-parameterized network contains sparse sub-networks which not only match but sometimes even exceed the performance of the original network, all by virtue of a "lucky" random initialization before training. The original paper was followed up with tips and tricks to train large-scale models under the same paradigm [15]. Since then, there has been a large, growing body of literature exploring its nuances. Although some of these recent works have tried to answer the question – how well do the tickets transfer across domains [38, 37], when it comes to vision tasks – the buck stops at image classification.

In this work, we aim to extend and explore the analysis of lottery tickets to fundamental visual recognition tasks of object detection, instance segmentation, and keypoint detection. Popular methods for such recognition tasks use a two-stage detection pipeline, with a supervised pre-trained convolutional neural network (ConvNet) backbone, a region proposal network (RPN), and one or more region-wise task-specific neural network branches. Loosely speaking, a ConvNet backbone is the most computationally intensive part of the architecture, and pre-training is the most time-consuming part. Therefore, as part of this study, we explore the following questions: (a) Are there *universal* sub-networks within the ConvNet backbone that can be transferred to the downstream object recognition tasks? (b) Can we train sparser and more accurate sub-networks for each of the downstream tasks? And, (c) How does the behavior or properties of these sub-networks change with respect to the corresponding dense network? We investigate these questions under the dominant settings used in object recognition frameworks. Specifically, we use ImageNet [7] pre-trained ResNet-18 and ResNet-50 [22] backbones, Faster R-CNN [41] and Mask R-CNN [21] modules for object recognition on Pascal VOC [11] and COCO [31] datasets. Our contributions are as follows:

- We show that tickets obtained from ImageNet training don't transfer to object recognition in case of COCO, *i.e.*, there are no *universal* tickets in pre-trained ImageNet models that can be used for downstream recognition tasks without a drop in performance. This is in contrast with previous works related to ticket transfer in vision models [37, 38]. In case of smaller datasets such as Pascal VOC, we are able to find winning tickets from ImageNet pre-training with upto 40% sparsity.

- With direct pruning, we can find "task-specific" tickets with up to 80% sparsity for each of the datasets and backbones. We also investigate the efficacy of methods introduced by [12, 38, 13, 42] such as iterative magnitude pruning, late resetting, early bird training, and layerwise pruning in the context of object recognition.

- Finally we analyse the behavior of tickets obtained for object recognition tasks, with respect to various task attributes such as object size, frequency, and difficulty of detection, to make some expected (and some surprising) observations.

2. Related Work

Model Compression: Ever since deep neural networks started gaining traction in real-world applications, there have been serious attempts made to reduce their parameters, intending to attain lower memory footprints [16, 52, 24, 44, 6, 30], higher inference speeds [49, 8, 18] and potentially better generalization [1]. Amongst the various proposed techniques, model pruning approaches are predominant mainly due to their simplicity and effectiveness. One line of methods follow an unstructured process where insignificant weights are set to zero and are frozen for the rest of the training. The significance of weights are quantified either by magnitude [19] or gradients during training time [29]. In structured pruning methods, relationships between pruned weights are taken into consideration, leading to pruning them in groups. Methods like [51] utilize Group Lasso regularization to prune redundant filter weights to enable structural sparsity, [33] uses explicit $L0$ regularization to make weights within structures have exact zero values, and network slimming [32] learns an efficient network by modelling the scaling factor of batch normalization layer.

The Lottery Ticket Hypothesis: The introduction of Lottery Ticket Hypothesis by [15] opened a pandora's box of immense possibilities in the field of pruning and sparse models. The original paper was followed by [14] where the authors introduce the concept of "late resetting" which enabled the application of the hypothesis to larger and deeper models. [56] followed up by proposing an extensive, in-depth analysis where they show that the resetting of the weights need not be to the exact initialization, but just need to the initial signs. [13] probes the aspect of resetting further to show that the reason why LTH works is because of its ability to make the subnetwork stable to SGD noise. As far as theoretical guarantees are considered, [36] offers

strong theoretical proofs for the experimental evidence of LTH. [17] probed an orthogonal question about the number of possible tickets from a network. They showed that a single initialization had multiple winning tickets with low overlap and empirically conclude that there exists an entire "distribution" of winning lottery tickets.

Complementary to LTH[15], [29] and [50] offer algorithms that can pick the winning ticket without the need for training. But they do not match the performance of the original procedure. The problem of longer training using LTH was effectively tackled by [53] which introduced the concept of "early bird tickets" where the authors show that the winning tickets and their masks are obtained in the first few epochs of training, foregoing the need to train the original initialization till convergence. The intriguing properties of LTH led to a glut of works which investigated its eclectic aspects. [38] scrutinize the generalization properties of winning tickets and offer empirical evidence that winning tickets can be transferred across datasets and optimizers, in the realm of image classification. The authors also discuss the learnt "inductive biases" of the tickets which may lead to worse performance of transferred tickets when compared with a ticket obtained from the same dataset. [37] then proposed a variation of the theory titled "transfer ticket hypothesis" where they investigate the effectiveness of transferring a mask generated from source dataset to a target dataset. [9] shows that the winning tickets do not perform simple overfitting to any domain and carry forward certain inherent biases which can prove useful for other domains too. There have been many applications of LTH in the fields of NLP [5] [39] [9][3] and Reinforcement Learning [54] [53] as well. The work of [43] briefly analyzes LTH on single stage detectors such as YOLOv3 [40] and achieves 90% winning tickets, while maintaining the mAP on the Pascal VOC 2007 dataset. However, as they evaluate on light-weight and fast detectors, their mAP (~ 56) is much lower compared to networks like Faster R-CNN [41] which reach mAP of ~ 69 with just a ResNet-18 backbone. They are also limited to object detection and do not provide a detailed analysis of LTH for this task. The idea for transferring subnetworks obtained from ImageNet to object detection tasks was concurrently discussed by [4]. For small datasets such as Pascal VOC, [4] observes that ImageNet tickets transfer for detection and segmentation tasks. However, we extend the analysis to the larger COCO dataset and show that this observation doesn't hold. We further build upon these results, to test out the generalization and transfer capabilities of winning lottery tickets across different object recognition datasets and tasks in computer vision.

3. Background: Lottery Ticket Hypothesis

LTH states that dense randomly-initialized neural networks contain sparse sub-networks which can be trained in isolation and can match the test accuracy of the original network. These sub-networks are called winning tickets and can be identified using an algorithm called Iterative Magnitude Pruning (IMP). Suppose the number of iterations for pruning is T and we wish to prune $p\%$ of the network weights. The weights/parameters are represented by $w \in \mathbb{R}^n$ and the pruning mask by $m \in \{0,1\}^n$ where n is the total number of weights in the network. The complete algorithm is presented in 1.

Algorithm 1 Iterative Pruning for LTH

1: Randomly initialize network f with initial weights w_0, mask $m_0 = \mathbb{1}$, prune target percentage p, and T pruning rounds to achieve it.
2: **while** $i < T$ **do**
3: Train network for N iterations $f(x; m_i \odot w_0) \rightarrow f(x; m_i \odot w_i)$
4: Prune bottom $p^{\frac{1}{k}}\%$ of $m_i \odot w_i$ and update m_i.
5: Reset to initial weights w_0
6: $i \leftarrow i + 1$ ▷ next round

This pruning method can be one-shot when it proceeds for only a single iteration or it can proceed for multiple iterations, k, pruning $p^{\frac{1}{k}}\%$ each round. The authors also use other techniques such as learning rate warmup and show that finding winning tickets is sensitive to the learning rate. While this method obtains winning tickets for smaller datasets, like MNIST [27], CIFAR10 [26], they fail to generalize to deeper networks, such as ResNets, and larger vision benchmarks, such as ImageNet [7]. [14] shows that IMP fails when resetting to the original initialization. They claim that resetting instead to the network weights after a few iterations of training provides greater stability and enables them to find winning tickets in these larger networks. They show that rewinding/late resetting to $3-7\%$ into training yields subnetworks which are 70% smaller in the case of ResNet-50, without any drop in accuracy.

4. LTH for Object Recognition

In this section, we extend the Lottery Ticket Hypothesis to several object recognition tasks, such as Object Detection, Instance Segmentation, and Keypoint Detection. In §4.1, we describe the datasets, models, and metrics we use in our paper. §4.2 examines the transfer of the lottery tickets obtained from ImageNet training to the downstream recognition tasks. §4.3 investigates direct pruning on the downstream tasks. §4.4 analyzes the various properties of winning tickets obtained using direct pruning.

4.1. Experimental setup

We evaluate LTH primarily on the 2 datasets - Pascal VOC 2007 and COCO. We deal with the 3 tasks of object

Table 1: Performance on the COCO dataset for ImageNet transferred tickets for ResNet-18 backbone at varying sparsity. The results for VOC are averaged over 5 runs with the standard deviation in parantheses. We obtain winning tickets at higher sparsity for smaller datasets like VOC compared to COCO.

Prune %	COCO Detection			COCO segmentation			COCO Keypoint			VOC Detection	
	Network sparsity	mAP	AP50	Network sparsity	mAP	AP50	Network sparsity	mAP	AP50	Network sparsity	mAP
90%	31.61%	25.59	43.69	31.61%	24.03	40.89	21.47%	55.30	79.30	79.49%	63.91(±0.41)
80%	28.10%	27.70	46.50	28.10%	25.90	43.70	19.09%	56.70	81.10	70.66%	65.82(±0.23)
50%	17.57%	28.52	47.54	17.57%	26.60	44.66	11.94%	56.96	80.83	44.16%	68.06(±0.11)
0%	0%	29.91	49.05	0%	27.64	46.00	0%	58.59	82.04	0%	68.53(±0.29)

Table 2: Performance on the COCO dataset for ImageNet transferred tickets for ResNet-50 backbone at various levels of pruning. The results for VOC are averaged over 5 runs with the standard deviation in parantheses. We obtain higher levels of sparsity compared to ResNet-18 transferred tickets which can be expected as it has fewer redundant parameters. Additionally, tickets for VOC have much higher sparsity with no drop in mAP compared to unpruned model.

Prune %	COCO Detection			COCO segmentation			COCO Keypoint			VOC Detection	
	Network sparsity	mAP	AP50	Network sparsity	mAP	AP50	Network sparsity	mAP	AP50	Network sparsity	mAP
90%	41.99%	30.66	50.75	41.99%	28.68	47.76	31.49%	57.78	82.25	65.37%	71.20(±0.21)
80%	37.33%	31.01	50.98	37.33%	29.04	47.90	28%	58.55	83.06	58.11%	71.08(±0.20)
0%	0%	38.5	59.29	0%	35.13	56.39	0%	64.59	86.48	0%	71.21(±0.32)

detection, instance segmentation, and keypoint detection for COCO and only object detection for VOC. We use Mask-RCNN for object detection and segmentation for COCO, Keypoint-RCNN for keypoint detection, and Faster-RCNN for object detection on VOC. We compare the results for ResNet-18 and ResNet-50 backbones. Note that while we use the term mean Average Precision (mAP) as a performance metric for all the tasks and datasets, the actual calculation of mAP is done using code provided in the respective datasets (and cannot be compared across the tasks).

4.2. Transfer of ImageNet Tickets

Many object recognition tasks utilize pre-trained networks whose backbones are trained on the ImageNet dataset. This is because ImageNet features and weights have shown the ability [21] to generalize well to several downstream vision tasks. A plethora of works exists which perform LTH for the ImageNet classification task and obtain winning tickets. Therefore, tickets for standard convolutional backbones, such as ResNets, are readily available. This raises the pertinent question of whether pruned ImageNet trained models transfer directly to object recognition tasks. In order to answer this question, we transfer the pruned model to the backbone of the RCNN-based network and fine-tune the full network while ensuring the pruned weights in the backbone remain as zeros.

We perform experiments for the two architectures: ResNet-18 and ResNet-50, where we obtain $10\%, 20\%$, and 50% tickets on ImageNet by following the approach of [14], and then transfer the model to the backbones of the three R-CNN based networks. All models were trained on COCO, encompassing the three recognition tasks and the results are summarized in Tables 1, 2. Additionally, we also perform similar experiments on the smaller Pascal VOC 2007 dataset for object detection to verify whether ImageNet tickets transfer without a significant drop in mAP. The results are shown in the last column of Tables 1, 2. Note that pruning percentage of the ImageNet ticket is not equal to the actual network sparsity of the various networks as only the backbone of the networks are transferred and they make up a fraction of the total weights.

We see that ImageNet tickets transferred to COCO show a noticeable drop in mAP even with low levels of sparsity. We also note that another drawback of training transferred tickets on COCO is that they require careful tuning of learning rate and batch size for different tasks. On the other hand, for smaller datasets such as Pascal VOC, winning tickets are easily obtained at higher levels of sparsity which is $\sim 45\%$ for ResNet-18 and $\sim 65\%$ for ResNet-50. The larger networks can be pruned to a greater extent for both datasets.

ImageNet transferred tickets offer very little sparsity as the backbone of the networks usually do not make up most of the weights. For example, the ResNet-18 based Mask-RCNN for COCO Detection and Segmentation has only 44% of the parameters in the backbone, and hence, the overall network sparsity reaches 31% when pruning 90% of the backbone weights, as shown in Table 1. The rest of the weights are usually from the fully-connected layers of the network. It is therefore imperative to prune layers in addition to the backbone to increase network sparsity without decrease in mAP. As a consequence, we look into directly pruning the full network using LTH.

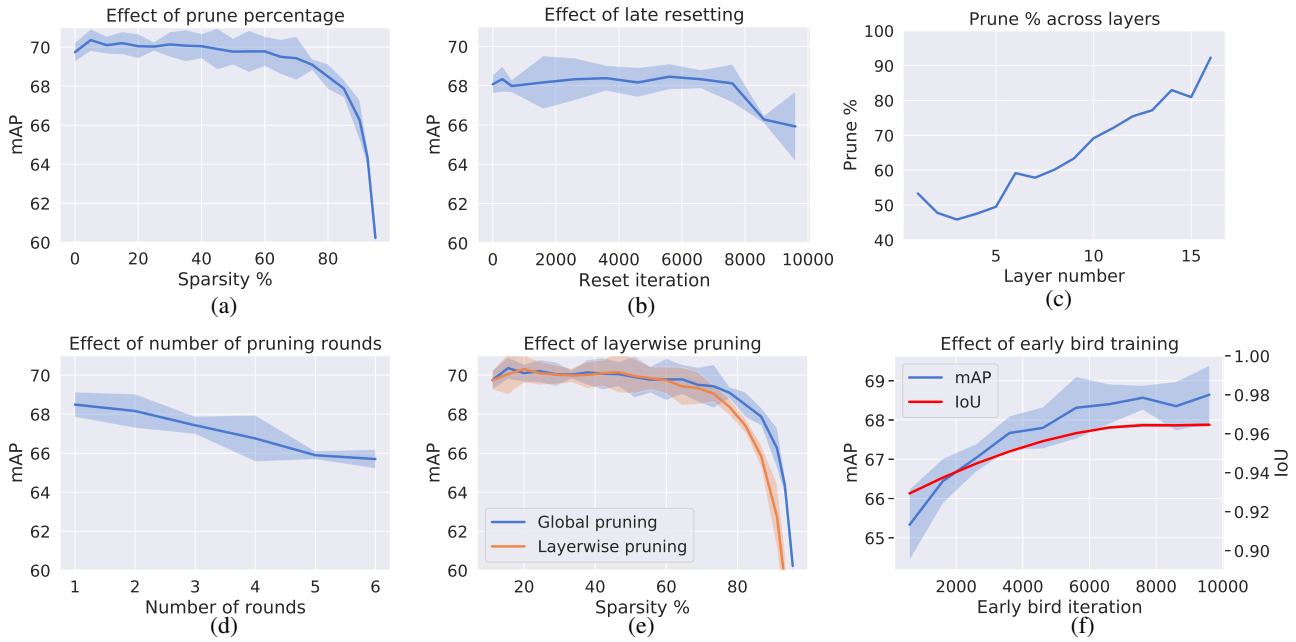

Figure 2: Effect of varying different hyperparameters for pruning Faster RCNN with ResNet-18 backbone on the Pascal VOC 2007 [11] dataset. All solid lines reported are the values averaged over 5 runs and the error bands are within 3 times the standard deviation.

4.3. Direct Pruning for Downstream Task

In this section, we analyze the effect of various hyperparameters and pruning strategies for detection networks in order to obtain winning tickets. We primarily use the ResNet-18 backbone for Faster-RCNN trained on VOC for all our experiments in this section, unless mentioned otherwise. Even though the ResNet-18 backbone is smaller than other backbone networks such as ResNet-50, we find that similar conclusions hold for the larger networks as well.

The Faster RCNN network consists of parameters which we group into 4 main modules: Base Convolutions, Classification Network Convolutions (Top), Region Proposal Network (RPN), Classification network box and classification fully connected heads (Box and Cls Head). We provide a detailed analysis of pruning these 4 groups and their effect on winning tickets. Additionally, we also analyze different pruning strategies and the role played by hyperparameters.

Varying Pruning Percentage: We evaluate the network performance at varying levels of sparsity. We prune different percentages of parameters in the Base and Top modules which include 88% of the total network parameters. The results are plotted in Fig.2a. We achieve performance within one standard deviation of the baseline, with 70% sparsity. Our models outperform the baseline mean, thereby proving that we can indeed obtain high performance winning tickets at much higher levels of sparsity for detection. Additionally, we see that at any given sparsity, direct pruning yields much better results compared to ImageNet transferred tickets. We also show that these observations pan other vision tasks by obtaining winning tickets for Mask-RCNN and Keypoint-RCNN with both ResNet-18 and ResNet-50 backbones on the COCO dataset. We additionally prune FC layers in these set of experiments in order to achieve desired sparsity levels as they take up $\sim 50\%$ of the total weights. The results for ResNet-18 are shown in Fig. 1. We obtain winning tickets with 80% sparsity on all the three tasks while outperforming the unpruned network for lower levels of sparsity. Additionally, we consistently outperform the different ImageNet transferred tickets (50%, 80%, 90%) by a large margin supporting our claim that direct training of tickets on downstream tasks yield better results than ImageNet tickets.

Effect of Early/Late Resetting: [14] states that resetting the network to a few iterations through training instead of the initialization stabilizes the winning ticket training. We evaluate whether this holds true for detection tasks as well. We show the performance of winning tickets as a function of resetting at various stages of training in Fig. 2b and observe that resetting during the earlier or even mid stages of training does not have a very strong effect on the final mAP. This is likely because the backbones of detection networks are initialized with ImageNet weights and are not random as is the case with other papers dealing with LTH in the classification setting. Therefore, the weights are more stable and late resetting is not necessary. We also additionally analyze effects of resetting towards the end of training and notice that there is a sharp drop in the performance after $8k$ iterations. This is because the learning rate is decayed at this stage of training and the parameters change significantly right after. A similar case holds when we perform

Table 3: Performance on Pascal VOC by pruning different modules of a ResNet-18 Faster-RCNN network. The results are averaged over 5 runs with the standard deviation in parantheses. ✓ represents the module being pruned, while Param % represents the percentage of parameters occupied by the modules being pruned.

Base	Top	RPN	Box, Cls Head	Param %	Network Sparsity	mAP
-	-	-	-	0	0%	69.74 (±0.16)
-	-	-	✓	0.65	0.52%	70.30 (±0.14)
-	-	✓	-	9.71	7.77%	70.02 (±0.19)
-	-	✓	✓	10.36	8.29%	70.08 (±0.10)
✓	-	-	-	21.93	17.55%	69.32 (±0.07)
✓	-	-	✓	22.59	18.07%	69.60 (±0.19)
✓	-	✓	-	31.64	25.31%	69.39 (±0.25)
✓	-	✓	✓	32.29	25.83%	69.47 (±0.15)
-	✓	-	-	66.39	53.11%	69.02 (±0.19)
-	✓	-	✓	67.04	53.63%	68.74 (±0.21)
-	✓	✓	-	76.09	60.88%	68.88 (±0.25)
-	✓	✓	✓	76.75	61.40%	68.93 (±0.26)
✓	✓	-	-	88.32	70.66%	68.45 (±0.21)
✓	✓	-	✓	88.97	71.18%	68.54 (±0.23)
✓	✓	✓	-	98.03	78.42%	68.51 (±0.23)
✓	✓	✓	✓	98.68	78.94%	68.47 (±0.10)

learning rate warmup but do late resetting before the learning rate is fully warmed up. The performance drops significantly as the learning rate keeps fluctuating showing that late resetting is quite sensitive to learning rate.

Pruning different Faster-RCNN modules: We prune 20% of the parameters of the various modules within the Faster-RCNN network and analyze their effects on the mAP. We also try different combinations of pruning with the modules and report the results in Table 3. Pruning the Box and Classification head (which takes up only 65% weights) outperforms the baseline case of no pruning, but does not always improve performance when other modules are being pruned. Additionally, pruning the RPN module increases the performance slightly even though it comprises of only 10% of the network weights. Next, pruning the Base module and/or the Top module of the backbone leads to a drop in performance, which is expected as they consist of 22% and 66% of the weights respectively. Pruning the Base alone, excluding the Top, performs nearly as well as the baseline, while including the Top yields a lower mAP.

Performance of Early-bird tickets: [53] showed that tickets can be found at early stages of training. We visualize this by obtaining masks at various stages in training and evaluating their performance. We also plot each masks' Intersection over Union (IoU) with the default mask obtained at the end of training. This IoU shows the overlap in the parameters being pruned. The results are visualized in Fig. 2f. We see that within 50% of network training we find tickets whose performance is within a standard deviation of the performance of the default ticket (obtained at the end of training). This is because the IoU becomes more or less stable at around 0.96 during the middle stages of training and the mask is unchanged as training advances. This allows us to cut down on the number of training iterations significantly with very little cost to the network performance.

Effect of number of rounds of pruning: [12] states that iterative pruning performs better than one-shot pruning on the classification task with small datasets and networks. We show that this does not necessarily hold true for detection and larger backbones. We plot the network's performance against various rounds of pruning and observe that one-shot pruning outperforms iterative methods in Fig. 2d.

Layer-wise *vs*. global pruning: [14] performs global pruning for larger datasets and networks and claims that pruning at the same rate in lower layers as compared to higher layers, is detrimental to the network performance. We evaluate the two methods of pruning on the detection task and show the results in Fig. 2e. Additionally, for global pruning, we plot the percentage of parameters pruned in each layer of the backbone network in Fig. 2c. Layer-wise pruning does as good as global pruning for lower levels of sparsity. However, there is a noticeable performance gap for sparsity levels above 60%. This is because layer-wise pruning forces lower layers with very few parameters to have high sparsity percentages. But as per Fig. 2c, for global pruning, we see that lower layers are pruned less as they are crucial to both the RPN and Classification stages of the network.

4.4. Properties of Winning Tickets

In Section 4.3, we showed that we can discover sparser networks within our two-stage Mask-RCNN detector if we directly prune on the task itself. We build upon those results to further probe the properties of winning tickets.

Effect of backbone architecture: In Fig. 3, we show how winning tickets behave for 2 different backbones, ResNet-18 and ResNet-50, at different sparsity levels (50%, 80%, 90%). We make two observations: First, the breaking point for both networks is $\sim$ 80% sparsity. However, performance of ResNet-18 drops more sharply than ResNet-50 afterwards. This is intuitive since ResNet-18 has fewer redundant parameters and over-pruning leads to drop in the performance. Second, as we gradually increase the sparsity of the networks, mAP increases for all tasks in case of both networks. However, gains for ResNet-18 models are consistently more than ResNet-50.

Do winning tickets behave differently for varying object sizes? Using the definition from [31], we categorize bounding boxes into small (area $< 32^2$), medium ($32^2 <$ area $< 96^2$), and large (area $> 32^2$). To understand how sparse networks behave for different sized objects, we plot the percentage gain or drop from the mAP of a dense network. Figure 4 shows the percentage change in mAP for different levels of sparsity in the Mask R-CNN model. We can

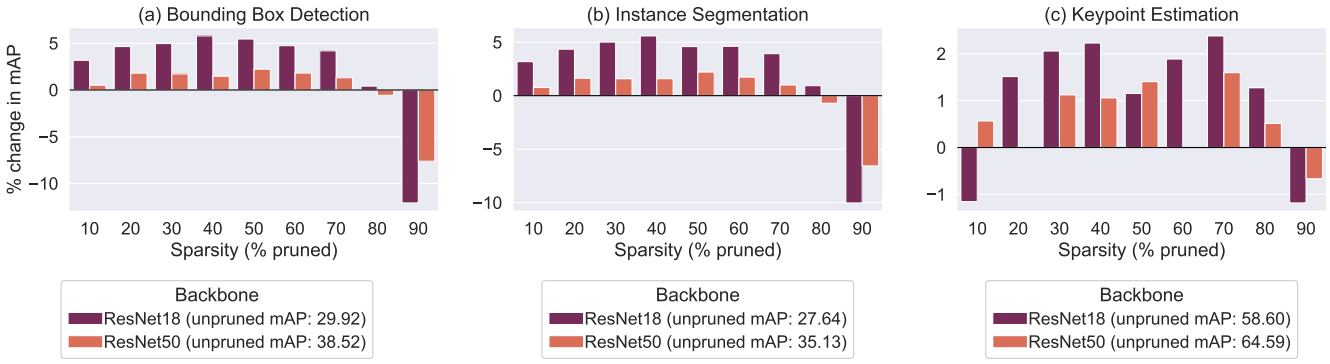

Figure 3: ResNet-18 *vs*. ResNet-50. We analyse change in mAP by using LTH on Mask R-CNN with different backbones.

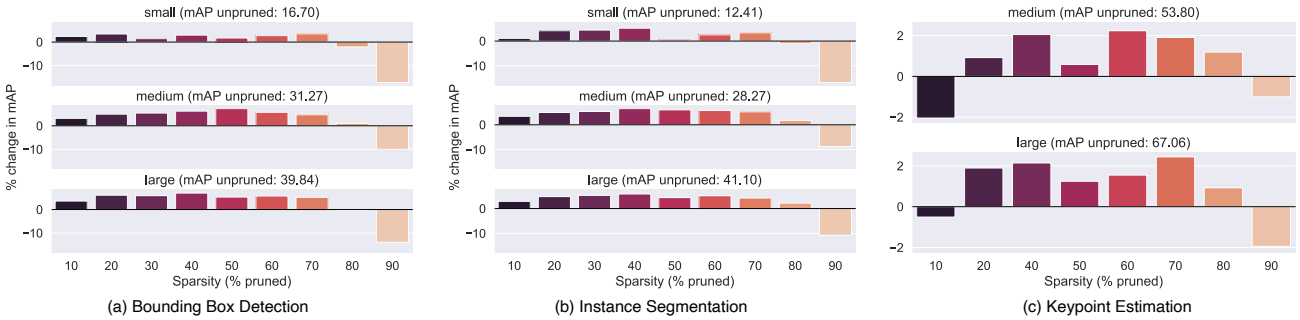

Figure 4: Comparison of Mean Average Precision (mAP) of pruned model for different object sizes in case of Object Detection, Instance Segmentation, Keypoint Estimation. x-axis shows the sparsity of the subnetwork (or the percentage of weights removed). y-axis shows the percentage drop in mAP as compared to the unpruned network. For all tasks, and object sizes, performance doesn't drop till about 80% sparsity. After which, small objects are hit slightly harder as compared to medium and large objects.

observe that in each case, the model performance increases with sparsity, until sparsity reaches 80%, after which, mAP sharply declines. We note that the percentage drop for small boxes is more, with winning tickets (10% of weights) showing a drop of over 17% in case of detection and segmentation tasks while medium sized objects show smaller drops than large objects for all tasks.

How does the performance of the pruned network vary for rare *vs*. frequent categories? We sort the 80 object categories in COCO by their frequency of occurrence in training data. We consider networks with 80% and 90% of their weights pruned and observe the percentage change in the bounding box mAP of the model with respect to the unpruned network for each of the categories. Figure 5(a) depicts the behavior with a bar graph. While for most categories, winning tickets are obtained at 80% sparsity, performance drops sharply with more pruning in case of rare categories (such as toaster, parking meter, and bear) as compared to common categories (such as person, car, and chair).

Do the winning tickets behave differently on easy vs hard categories? For a machine learning model, an object can be easy or hard to recognize because of a variety of reasons. We have already discussed two reasons that influence the performance — number of instances available in the training data, and size of the object. There can also be other causes that can render an object unrecognizable in given surroundings. Camouflage or occlusion, poor camera quality, light conditions, distance from the camera, or just variations within different instances or views of the object are few of them. Since exhaustive analyses of these causes is intractable, we rank object categories based on performance of an unpruned Mask R-CNN model. We do this categorization for detection and segmentation models as shown in Figure 5(b) and (c). Note that 'easy' and 'hard' categories from these two definitions have an overlap but they are not the same. For example, knife, handbag, and spoon are the categories with lowest bounding box mAP, and giraffe, zebra, and stop signs are one with the highest (excluding 'hair drier' which has 0 mAP). On the other hand, skis, knife, and spoon have the lowest segmentation mAP, while stop sign, bear, and fire hydrant have the highest. From the Figure 5(b) and (c), we make the following observations — (i) tickets with 80% sparsity can actually increase mAP for certain categories like snowboard by as much as 38%, (ii) Going from 80% to 90% sparsity, mAP drops significantly for easy categories, (iii) categories that are hit the hardest such as skis, hot dog, spoon, fork, handbags usually have long, thin appearance in images.

Do winning tickets transfer across tasks? We showed that

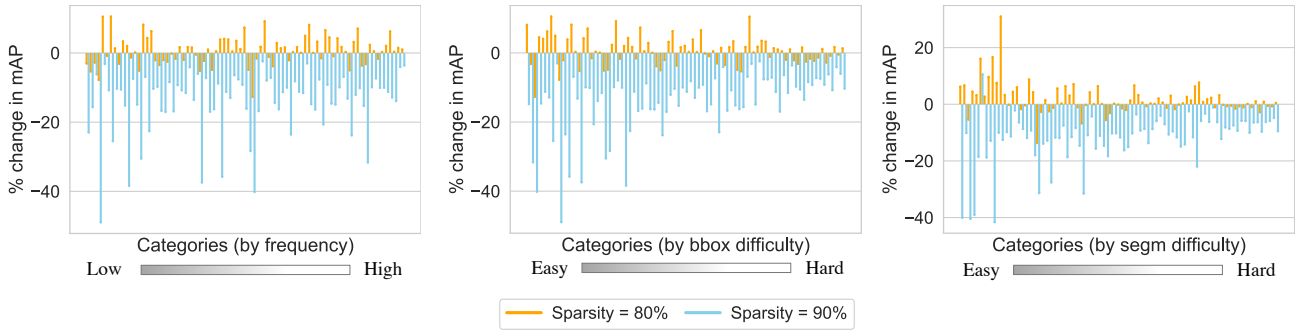

Figure 5: Comparison of Mean Average Precision (mAP) of pruned model for 80 COCO object categories. x-axis in each of the plot is a list of categories (sorted using different criteria). y-axis shows the percentage drop in mAP as compared to the unpruned network.

Table 4: Effect of ticket transfer across tasks. Transferred tickets do worse than direct training as expected, but still do not result in drastic drops in the mAP or AP50. Here we do task transfer using the 80% pruned model.

Target task	Source task	Network sparsity	mAP	AP50
Det	Det/Seg	78.4%	30.04	49.40
	Keypoint	50.11%	23.94	41.08
Seg	Det/Seg	78.4%	27.90	46.68
	Keypoint	50.11%	23.02	39.01
Keypoint	Det/Seg	76.98%	58.31	81.53
	Keypoint	79.4%	59.34	82.36

ImageNet tickets transfer to a limited extent to downstream tasks. We further study whether the tickets obtained from the downstream task of detection/segmentation transfer to keypoint estimation and vice-versa. We train Mask-RCNN and Keypoint-RCNN respectively for the two tasks on the COCO dataset while maintaining a sparsity level of 80%. For both the tasks we transfer all values till box head modules, after which the model structures differ. The results are shown in Table 4. We can observe that the drop is marginal for the transfer of tickets between detection-segmentation to keypoint task, as compared with the reverse case which registers a significant drop. This might be because the ticket is obtained on the keypoint task which is trained only on 'human' class and it fails to transfer well for the detection task which uses the entire COCO dataset.

5. Discussion

[37, 38] show that winning tickets transfer well across datasets. However, the study in [37] was limited to smaller datasets, like CIFAR-10 and FashionMNIST, and both [37, 38] are limited to classification tasks. We obtain contrasting results when transferring tickets across tasks as shown in Sec. 4.2. ImageNet tickets transfer with approximately 40% sparsity to fall within one standard deviation of the baseline network. This is likely due to the fact that winning tickets retain inductive biases from the source dataset which are less likely to transfer to a new domain and task.

Additionally, we show that unlike prior LTH works, iterative pruning degrades the performance of subnetworks on detection and one-shot pruning provides the best networks. We also observe that due to the use of pre-trained weights from ImageNet for the backbone of detection networks, late resetting is not necessary for finding winning tickets. This is in contrast to the [14], which is restricted to the classification task involving random initialization for the networks. Like previous works, in our experiments as well, we find that sparse lottery tickets often outperform the dense networks themselves. However, we make another interesting observation — in each of object recognition tasks, tickets with fewer parameters such as ResNet-18 show more gains in performance as compared to tickets with more parameters (ResNet-50). We also find that small and infrequent objects face higher performance drop as the sparsity increases.

6. Conclusion

We investigate the Lottery Ticket Hypothesis in the context of various object recognition tasks. Our study reveals that the main points of original LTH hold for different recognition tasks, *i.e.*, we can find subnetworks or winning tickets in object recognition pipelines with up to 80% sparsity, without any drop in performance on the task. These tickets are task-specific, and pre-trained ImageNet model tickets don't perform as well on the downstream recognition tasks. We also analyse claims made in recent literature regarding training and transfer of winning tickets from an object recognition perspective. Finally, we analyse how the behavior of sparse tickets differ from their dense counterparts. In the future, we would like to investigate how much speed up can be achieved using these sparse models with various hardware [34] and software modifications [10]. Extending this analyses for even bigger datasets such as JFT-300M [46] or IG-1B [35] and for self-supervised learning techniques is another direction to pursue.

Acknowledgements. This work was partially supported by DARPA GARD #HR00112020007 and a gift from Facebook AI.

References

[1] Sanjeev Arora, Rong Ge, Behnam Neyshabur, and Yi Zhang. Stronger generalization bounds for deep nets via a compression approach. *arXiv preprint arXiv:1802.05296*, 2018.

[2] Jimmy Ba and Brendan Frey. Adaptive dropout for training deep neural networks. In *Advances in neural information processing systems*, pages 3084–3092, 2013.

[3] C Brix, P Bahar, and H Ney. Successfully applying the stabilized lottery ticket hypothesis to the transformer architecture. *ACL*, 2020.

[4] Tianlong Chen, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Michael Carbin, and Zhangyang Wang. The lottery tickets hypothesis for supervised and self-supervised pre-training in computer vision models. *arXiv preprint arXiv:2012.06908*, 2020.

[5] T Chen, J Frankle, S Chang, S Liu, Y Zhang, Z Wang, and M Carbin. The lottery ticket hypothesis for pre-trained bert networks. In *NeurIPS*, 2020.

[6] Zhiyong Cheng, Daniel Soudry, Zexi Mao, and Zhenzhong Lan. Training binary multilayer neural networks for image classification using expectation backpropagation. *arXiv preprint arXiv:1503.03562*, 2015.

[7] 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.

[8] Emily L Denton, Wojciech Zaremba, Joan Bruna, Yann LeCun, and Rob Fergus. Exploiting linear structure within convolutional networks for efficient evaluation. In *Advances in neural information processing systems*, pages 1269–1277, 2014.

[9] Shrey Desai, Hongyuan Zhan, and Ahmed Aly. Evaluating lottery tickets under distributional shifts. *arXiv preprint arXiv:1910.12708*, 2019.

[10] Erich Elsen, Marat Dukhan, Trevor Gale, and Karen Simonyan. Fast sparse convnets. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14629–14638, 2020.

[11] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

[12] J Frankle and M Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *ICLR*, 2019.

[13] Jonathan Frankle, Gintare Karolina Dziugaite, Daniel M Roy, and Michael Carbin. Linear mode connectivity and the lottery ticket hypothesis. *arXiv preprint arXiv:1912.05671*, 2019.

[14] Jonathan Frankle, Gintare Karolina Dziugaite, Daniel M Roy, and Michael Carbin. Stabilizing the lottery ticket hypothesis. *arXiv preprint arXiv:1903.01611*, 2019.

[15] J Frankle, G K Dziugaite, D M Roy, and M Carbin. Stabilizing the lottery ticket hypothesis. In *ICML*, 2020.

[16] Yunchao Gong, Liu Liu, Ming Yang, and Lubomir Bourdev. Compressing deep convolutional networks using vector quantization. *arXiv preprint arXiv:1412.6115*, 2014.

[17] Kathrin Grosse and Michael Backes. How many winning tickets are there in one dnn? *arXiv preprint arXiv:2006.07014*, 2020.

[18] 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*, 2015.

[19] 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.

[20] Trevor Hastie, Robert Tibshirani, and Martin Wainwright. *Statistical learning with sparsity: the lasso and generalizations*. CRC press, 2015.

[21] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[22] 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.

[23] Yanping Huang, Yonglong Cheng, Dehao Chen, HyoukJoong Lee, Jiquan Ngiam, Quoc V. Le, and Zhifeng Chen. Gpipe: Efficient training of giant neural networks using pipeline parallelism. *CoRR*, abs/1811.06965, 2018.

[24] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Quantized neural networks: Training neural networks with low precision weights and activations. *The Journal of Machine Learning Research*, 18(1):6869–6898, 2017.

[25] Simon Kornblith, Jonathon Shlens, and Quoc V Le. Do better imagenet models transfer better? In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2661–2671, 2019.

[26] Alex Krizhevsky et al. Learning multiple layers of features from tiny images. 2009.

[27] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 1998.

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

[29] N Lee, T Ajanthan, and P Torr. SNIP: Single-shot network pruning based on connection sensitivity. In *ICLR*, 2019.

[30] Rundong Li, Yan Wang, Feng Liang, Hongwei Qin, Junjie Yan, and Rui Fan. Fully quantized network for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2810–2819, 2019.

[31] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[32] Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. Learning efficient convolutional networks through network slimming. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2736–2744, 2017.

[33] Christos Louizos, Max Welling, and Diederik P Kingma. Learning sparse neural networks through l_0 regularization. *arXiv preprint arXiv:1712.01312*, 2017.

[34] Liqiang Lu, Jiaming Xie, Ruirui Huang, Jiansong Zhang, Wei Lin, and Yun Liang. An efficient hardware accelerator for sparse convolutional neural networks on fpgas. In *2019 IEEE 27th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)*, pages 17–25. IEEE, 2019.

[35] Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens van der Maaten. Exploring the limits of weakly supervised pretraining. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 181–196, 2018.

[36] Eran Malach, Gilad Yehudai, Shai Shalev-Shwartz, and Ohad Shamir. Proving the lottery ticket hypothesis: Pruning is all you need. *arXiv preprint arXiv:2002.00585*, 2020.

[37] R Mehta. Sparse transfer learning via winning lottery tickets. In *NeurIPS*, 2020.

[38] Ari Morcos, Haonan Yu, Michela Paganini, and Yuandong Tian. One ticket to win them all: generalizing lottery ticket initializations across datasets and optimizers. In *Advances in Neural Information Processing Systems*, pages 4932–4942, 2019.

[39] Rajiv Movva and Jason Y. Zhao. Dissecting lottery ticket transformers: Structural and behavioral study of sparse neural machine translation. *ArXiv*, abs/2009.13270, 2020.

[40] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[41] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

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

[43] Andrey Salvi and Rodrigo Barros. An experimental analysis of model compression techniques for object detection. In *Anais do VIII Symposium on Knowledge Discovery, Mining and Learning*, pages 49–56, Porto Alegre, RS, Brasil, 2020. SBC.

[44] Daniel Soudry, Itay Hubara, and Ron Meir. Expectation backpropagation: Parameter-free training of multilayer neural networks with continuous or discrete weights. In *Advances in neural information processing systems*, pages 963–971, 2014.

[45] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[46] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting Unreasonable Effectiveness of Data in Deep Learning Era. In *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[47] Mingxing Tan and Quoc V Le. Efficientnet: Rethinking model scaling for convolutional neural networks. *arXiv preprint arXiv:1905.11946*, 2019.

[48] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

[49] Vincent Vanhoucke, Andrew Senior, and Mark Z Mao. Improving the speed of neural networks on cpus. 2011.

[50] Chaoqi Wang, Guodong Zhang, and Roger Grosse. Picking winning tickets before training by preserving gradient flow. *arXiv preprint arXiv:2002.07376*, 2020.

[51] Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Learning structured sparsity in deep neural networks. In *Advances in neural information processing systems*, pages 2074–2082, 2016.

[52] Jiaxiang Wu, Cong Leng, Yuhang Wang, Qinghao Hu, and Jian Cheng. Quantized convolutional neural networks for mobile devices. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4820–4828, 2016.

[53] H You, C Li, P Xu, Y Fu, Y Wang, X Chen, R G Baraniuk, Z Wang, and Y Lin. Drawing early-bird tickets: Toward more efficient training of deep networks. In *ICLR*, 2020.

[54] H Yu, S Edunov S, Y Tian Y, and A S Morcos. Playing the lottery with rewards and multiple languages: lottery tickets in RL and NLP. In *ICLR*, 2020.

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

[56] Hattie Zhou, Janice Lan, Rosanne Liu, and Jason Yosinski. Deconstructing lottery tickets: Zeros, signs, and the supermask. In *Advances in Neural Information Processing Systems*, pages 3597–3607, 2019.