Efficient Lottery Ticket Finding: Less Data is More

Zhenyu Zhang[*1] **Xuxi Chen**[*2] **Tianlong Chen**[*2] **Zhangyang Wang**[2]

Abstract

The lottery ticket hypothesis (LTH) (Frankle & Carbin, 2018) reveals the existence of winning tickets (sparse but critical subnetworks) for dense networks, that can be trained in isolation from random initialization to match the latter's accuracies. However, finding winning tickets requires burdensome computations in the train-prune-retrain process, especially on large-scale datasets (e.g., ImageNet), restricting their practical benefits. This paper explores a new perspective on finding lottery tickets more efficiently, by doing so only with a specially selected subset of data, called **Pr**uning-**A**ware **C**ritical set (PrAC set), rather than using the full training set. The concept of PrAC set was inspired by the recent observation, that deep networks have samples that are either *hard to memorize* during training, or *easy to forget* during pruning. A PrAC set is thus hypothesized to capture those most challenging and informative examples for the dense model. We observe that a high-quality winning ticket can be found with training and pruning the dense network on the very compact PrAC set, which can substantially save training iterations for the ticket finding process. Extensive experiments validate our proposal across diverse datasets and network architectures. Specifically, on CIFAR-10, CIFAR-100, and Tiny ImageNet, we locate effective PrAC sets at $35.32\% \sim 78.19\%$ of their training set sizes. On top of them, we can obtain the same competitive winning tickets for the corresponding dense networks, yet saving up to $82.85\% \sim 92.77\%$, $63.54\% \sim 74.92\%$, and $76.14\% \sim 86.56\%$ training iterations, respectively. Crucially, we show that a PrAC set found is **reusable** across different network architectures, which can amortize the extra cost of finding PrAC sets, yielding a practical regime for efficient lottery ticket finding.

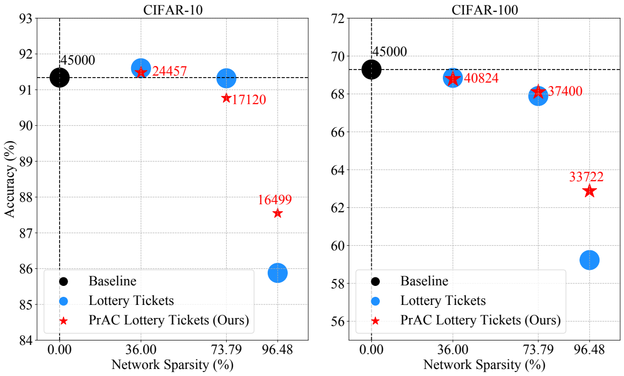

Figure 1. Test accuracy of found subnetworks from ResNets at different sparsity levels on CIFAR-10 and CIFAR-100. Black dots (●) represent the performance of unpruned baselines; blue dots (●) indicate the performance of vanilla lottery tickets found with full data (Frankle & Carbin, 2018), and red star (★) are established by our PrAC lottery tickets. Red numbers are the number of samples in the PrAC set. The lottery tickets found on the PrAC sets could perform on par with the vanilla ones at moderate sparsity levels, and even outperform the latter at the highest sparsity of 96.48%.

1. Introduction

Deep neural networks (DNNs) have revolutionized the performance bar of various tasks, yet suffer from substantial over-parameterization (Voulodimos et al., 2018). Parameter-counts are frequently measured in billions rather than millions, with the time and financial outlay necessary to train these models growing in concert. Once trained, they can be pruned of excessive capacity (Han et al., 2015; Tang et al., 2020). However, conventional approaches first train dense DNNs, and then prune the trained them to high levels of sparsity. Those methods significantly reduce the inference complexity yet cost even greater computational resources and memory footprints at training.

An emerging subfield has explored the prospect of directly training smaller, sparse subnetworks in place of the full models without sacrificing performance. The key idea is to reuse the sparsity pattern found through pruning and train a sparse network from scratch. The seminal work (Frankle & Carbin, 2018) hypothesized that standard DNNs contain sparse matching subnetworks, often called *winning ticket*, capable of training in isolation to full accuracy. In other words, we could have trained smaller networks from the start if only we had known which subnetworks to choose. In larger-scale real-world settings, current methods often

[*]Equal contribution [1]University of Science and Technology of China [2]University of Texas at Austin. Correspondence to: Zhangyang Wang <atlaswang@utexas.edu>.

Proceedings of the 38th International Conference on Machine Learning, PMLR 139, 2021. Copyright 2021 by the author(s).

empirically choose winning tickets by *Iterative Magnitude Pruning* (**IMP**), sometimes at an early training point called "rewinding" (Frankle et al., 2019b; 2020a). Other works also showed sparsity might emerge at the initialization (Lee et al., 2018; Wang et al., 2020), or at the early training stage (You et al., 2020). However, it was observed in (Frankle et al., 2020b) that IMP still outperforms those carefully designed alternatives by clear margins, and remain as the most effective lottery ticket finding approach. However, the cumbersome train-prune-train cycle required by IMP makes it extremely expensive to find lottery tickets from large models and datasets, and also questioning the practical efficiency benefits of finding lottery tickets.

In parallel to seeking *model sparsity* during training, another complementary and promising line of ideas exploits *data sparsity*, *i.e*, reducing training costs by the informed selection of training samples (Tsang et al., 2005; Har-Peled & Kushal, 2007). Such techniques often select a small but critical *core set* from a large dataset, by which way a significant fraction of examples can be omitted from training while still maintaining the trained models' generalization (Zhao & Zhang, 2015; Katharopoulos & Fleuret, 2018; Toneva et al., 2019; Mirzasoleiman et al., 2020). Also related to the core set approach is the dataset distillation (Wang et al., 2018) that aims to summarize training images into a handful of synthetic images, ensuring that DNNs trained on the latter generalize almost as well as trained on the former.

1.1. Research Questions & Our Contributions

However, the questions below are not yet clear:

(Q_1) How will the "model sparsity" (e.g. LTH) and "data sparsity" (e.g., core set) interplay? Can one help the other? Can they possibly be jointly utilized to push training efficiency to the next level?

To answer the above question (Q_1), we first formulate and address a prerequisite question (Q_0):

(Q_0) What samples are considered as "core" for finding a lottery ticket (trainable sparse DNN)?

A typical "coreset" (Mirzasoleiman et al., 2020) aims to guarantee that models fitting the coreset also provide a good fit for the original data, and finding it is treated as an approximation problem such as sampling or clustering. To find a sparse subnetwork that can *match* the performance of the full model, the challenge level is escalated higher since sparse DNNs are way tougher to train (Evci et al., 2019), and the core samples have also to identify the trainable sparse connectivity patterns. In other words, the new core set needs to encode not only the full dataset's knowledge, but also the trainability.

In this paper, we first attempt to address (Q_0) by investigating a new concept called **Pr**uning-**A**ware **C**ritical set (**PrAC set**), that targets to characterize important samples for finding lottery tickets that are both same *generalizable* and *trainable*. Considering that the lottery ticket iterates between two steps: *(re-)training*, and *pruning*. Conceptually, we hope a PrAC set to capture two types of samples:

- Samples that are *hard to memorize*, during (re-)training of the (original or pruned) DNN. Recent observations by (Toneva et al., 2019; Yao et al., 2020; Xia et al., 2021; Han et al., 2020) reveal that certain examples are memorized easily during training, but some others are repeatedly forgotten. Such (un)forgettable examples generalize across different architectures in the same dataset. The forgetting dynamics also suggest one can train a DNN on a dataset with a large fraction of the least forgotten examples removed.
- Samples that are *easy to forget*, during pruning the dense DNN into a sparse DNN. Pruning steps are essential to the final (trainable) sparsity, yet hampering both memorization and generalization. Moreover, it has been observed by (Hooker et al., 2020a) that pruning disproportionately impacts the model performance on a narrow subset of the dataset, e.g., the atypical, semantically ambiguous or underrepresented images.

During lottery ticket finding, by calculating the forgotten dynamics for each sample within training and the prediction differences after each pruning, we can effectively collect those most informative samples and build a PrAC set. In fact, our approach is a *co-design between data and model sparsity*, which we feel essential due to the hardness of (Q_0).

Equipped with PrAC sets, we then examine (Q_1) and present a comprehensive set of experiments, integrating the PrAC set with an efficient lottery ticket finding and training framework. In general, we find PrAC sets to help find comparable winning tickets with much higher training efficiency, compared to the vanilla IMP scheme using the full set, with little performance drop (sometimes even with performance gains)[1]. We summarize our main findings as follows:

- We identify winning tickets and PrAC sets broadly across different datasets (CIFAR-10, CIFAR-100, Tiny ImageNet) and architectures (ResNet-20, ResNet-56, and VGG-16). High-quality winning tickets can be found on the PrAC sets while saving training time and costs. Specifically, we save $82.85\% \sim 92.77\%$ on CIFAR-10, $63.54\% \sim 74.92\%$ on CIFAR-100, and $76.14\% \sim 86.56\%$ on Tiny ImageNet in training iterations, while maintaining or even boosting their achievable accuracies.
- PrAC sets show great transferability across architectures on the same dataset, which can amortize the cost of finding PrAC sets in practice. Taking ResNet-20 as the source architecture, the PrAC set found in CIFAR-10 and CIFAR-100 can locate winning tickets in ResNet-56 and VGG-19

[1] Our implementations are available at: `https://github.com/VITA-Group/PrAC-LTH`

with almost no performance degradation. We further visualize the PrAC set samples, conclude their patterns, and compare them with multiple sample selection methods.

- On CIFAR-10, the PrAC winning ticket (79.03%) are sparser than tickets from random pruning (48.80%). Our ticket finding also outperforms other efficient network pruning methods. For example, at 93.13% sparsity, our PrAC lottery tickets can outperform SynFlow (Tanaka et al., 2020) by 1.51%, SNIP (Lee et al., 2018) by 5.47%, and GraSP (Wang et al., 2020) by 18.73%.

2. Related Work

Lottery Ticket Hypothesis (LTH). LTH (Frankle & Carbin, 2018) has drawn lots of attention. Later on, (Frankle et al., 2019a; Renda et al., 2020) scaled up LTH to larger models by early weight rewinding that relaxes the use of original random initialization. Another intriguing property of lottery tickets, the transferability, has also been thoroughly examined (Mehta, 2019; Morcos et al., 2019; Desai et al., 2019; Chen et al., 2020b;a). Zhou et al. (2019) investigated different components in LTH and observed supermasks in winning tickets. LTH has also been extended to various applications (Gale et al., 2019; Chen et al., 2020b; Yu et al., 2020; Chen et al., 2021c; Kalibhat et al., 2020; Chen et al., 2021a; Ma et al., 2021; Gan et al., 2021; Chen et al., 2021b) beyond image classification.

Unstructured IMP (Han et al., 2015; Frankle & Carbin, 2018) serves as an effective method to find these winning tickets, and Dynamic Sparse Training (Mostafa & Wang, 2019; Mocanu et al., 2018; Evci et al., 2020) is also capable of identifying subnetworks with promising performance. However their computational expensiveness motivates many efficient alternatives that hope to locate sparse trainable subnetworks at random initialization or early training stage, with less or no training (Lee et al., 2018; You et al., 2020; Wang et al., 2020; Tanaka et al., 2020; Frankle et al., 2020b). Unfortunately, those sparse subnetworks found at beginning usually have clearly *inferior performance* to the IMP-found winning tickets, leaving IMP still the mainstream LTH scheme. This paper explores a complementary new perspective on finding lottery tickets more efficiently by co-designing a specially crafted subset. Our method secures winning tickets of *fully comparable performance* to the full IMP scheme, and it can also be straightforwardly combined with those efficient pruning methods if needed.

Active Learning and Core-Set Approaches. Another closely related literature is the problem of active learning (Settles, 2009; 2012) and core-set selection (Tsang et al., 2005; Har-Peled & Kushal, 2007; Bachem et al., 2017; Sener & Savarese, 2017). Specifically, Zhao & Zhang (2015); Katharopoulos & Fleuret (2018); Toneva et al. (2019); Wang et al. (2018); Mirzasoleiman et al. (2020); Hooker et al. (2020a;b) select core-sets by the importance sampling. Zhao & Zhang (2015); Katharopoulos & Fleuret (2018) sort the samples according to the magnitude of its loss gradient with respect to parameters of the network. Toneva et al. (2019) samples the examples based on the forgetting dynamics during the course of learning. Mirzasoleiman et al. (2020) constructs core-set that provides an approximately low-rank Jacobian matrix. Wang et al. (2018) generates synthetic examples to distill the knowledge from the entire dataset, and Hooker et al. (2020a;b) find pruning can cause disproportionately high errors on a small subset. We draw inspirations from several of those ideas, and extend the idea of core-set to be co-optimized with LTH.

3. Methodology

In this section, we present our framework to co-design model and data sparsity, which works in an iterative fashion of two alternative steps: i) constructing the Pruning-Aware Critical (PrAC) set with pruned models, which selects the most challenging and informative examples; ii) utilizing PrAC sets to identify critical subnetworks, (*i.e.*, lottery tickets), which takes much less training iterations. In this way, the burdensome computations of the train-prune-retrain process in tickets finding, can be substantially reduced. The overall pipeline is summarized in Algorithm 1.

Algorithm 1 Data and Model Sparsity Co-Design

```
Input: Full training data $\mathcal{D}_0$, a threshold for the number of
    forgets $\mathcal{E}_{\mathrm{F}}$, a network $f(\boldsymbol{\theta}_0, )$ with initialization weights
    $\boldsymbol{\theta}_0$, pruning ratios $\rho$, and the desired sparsity level $s$.
Output: Sparse mask $\boldsymbol{m}$ ($\|\boldsymbol{m}\|_0 \ll \|\boldsymbol{\theta}_0\|$), pruning-aware
    critical (PrAC) set $\mathcal{P}$ ($|\mathcal{P}| \ll |\mathcal{D}_0|$)
1:  Set $\boldsymbol{m} = \mathbf{1} \in \mathbb{R}^{\|\boldsymbol{\theta}_0\|_0}$, and $\mathcal{D} = \mathcal{D}_0$
2:  while $(1 - \frac{\|\boldsymbol{m}\|_0}{\|\boldsymbol{\theta}_0\|_0} \leq s)$ do
3:      # Data slimming to construct PrAC sets
4:      Set $\mathcal{P} = \varnothing$
5:      Train $f(\boldsymbol{m} \odot \boldsymbol{\theta}_0, \cdot)$ on $\mathcal{D}$ for T epochs and update
        the forgetting statistics for all training samples in $\mathcal{D}$
6:      Select samples from $\mathcal{D}$ with forgetting statistics
        greater than $\mathcal{E}_{\mathrm{F}}$, and add them into $\mathcal{P}$
7:      # Model slimming to locate critical subnetworks
8:      Prune $\rho = 20\%$ remaining weights of subnetworks
        $f(\boldsymbol{m} \odot \boldsymbol{\theta}_{\mathrm{T}}, \cdot)$, and update $\boldsymbol{m}$ accordingly
9:      # Data slimming to construct PrAC sets
10:     Select samples from $\mathcal{D}_0$ that full model and subnetworks disagree with, and add them into into $P$
11:     Set $\mathcal{D} = \mathcal{P}$
12: end while
```

3.1. Identifying the Pruning-Aware Critical (PrAC) Set

This section shows the details about how to shrink the training set to proposed Pruning-Aware Critical set, which illustrates the process in lines 3-6, 10 of Algorithm 1.

Rationale I: Critical Examples for Training. In the network training, each batch of data has its own and likely different statistics. Therefore, they can be regarded as different "tasks". Catastrophic forgetting happens (Toneva et al., 2019) during the training process so that certain examples are memorized easily during training while some others are repeatedly forgotten. Different behaviors on samples reveal the difficulty of them, providing a natural way to select critical examples, *i.e.*, the degree of difficult-to-forget of each training sample. As pointed out by (Toneva et al., 2019), training models on a dataset with a large fraction of the least forgotten examples removed can yield extremely competitive performance as training on the full data.

Approach I: Calculating the Forgetting Statistics. To measure how easy for a model to forget a sample, we use *forgetting statistics* (Toneva et al., 2019) as the metric. Specifically, the forgetting statistics for a sample is the number of transition from a correctly to incorrectly classified sample. We sort the number of forgetting statistics of all training data, and select those have statistics greater than a pre-defined threshold into the PrAC set, in lines 3-6 of Alg. 1.

Rationale II: Critical Examples for Pruning. Although the performance of located sparse lottery tickets can match the performance of the full model, the increased number of zero weights might have hampered the memorization and generalization ability of models. Such conjecture has been supported by recent observation (Hooker et al., 2020a), which demonstrates there exists pruning-aware examples that have different prediction between the full and pruned model. These examples are semantically ambiguous and hard for the pruned model to memorize.As a consequence, we merge these easy-to-forget samples into the PrAC set we construct to remedy such capacity loss.

Approach II: Utilizing the Disagreement between Full and Pruned Models. For each sample in the training set, we calculate the predicted class of $\boldsymbol{x}$ by full dense models and pruned subnetworks, *i.e.*, $f(\boldsymbol{\theta}, \boldsymbol{x})$ and $f(\boldsymbol{m} \odot \boldsymbol{\theta}, \boldsymbol{x})$, where $f(\boldsymbol{\theta}, \cdot)$ is a model with parameters $\boldsymbol{\theta}$, and $\boldsymbol{m}$ is a sparse mask. If two predictions are different, then we include this sample to the PrAC set (line 10 of Algorithm 1).

3.2. Efficient Lottery Tickets Finding

Matching Subnetworks and Lottery Ticket. A subnetwork within a dense network $f(\boldsymbol{\theta}, \cdot)$ is defined as $f(\boldsymbol{m} \odot \boldsymbol{\theta}, \cdot)$, where $\boldsymbol{m} \in \{0,1\}^{\|\boldsymbol{\theta}\|_0}$ is a binary mask indicating the sparsity levels, and $\odot$ is the element-wise product. Let $\boldsymbol{\theta}_0$ be the initial weights, and $\boldsymbol{\theta}_i$ be the weights after i training steps. Following Frankle & Carbin (2018), we define the *matching network* as a subnetwork $f(\cdot, \boldsymbol{m} \odot \boldsymbol{\theta})$, with $\boldsymbol{\theta}_t$ being the initialization of $\boldsymbol{\theta}$, that can reach the comparable performance to the full network within a similar training iterations; a *winning ticket* is defined as a matching subnetwork where $\boldsymbol{\theta_0}$ as the initial weights.

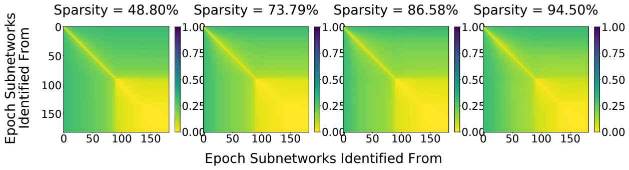

Figure 2. Results of the pairwise hamming distance between identified subnetworks on CIFAR-10 with ResNet-20

Identifying Subnetworks. To identify subnetworks, we adopt an iterative magnitude pruning method (Han et al., 2015). We follow a conventional iterative train-prune-retrain process in Frankle & Carbin (2018), yet with our PrAC set: We train the model $f(\boldsymbol{m} \odot \boldsymbol{\theta}, \cdot)$ on our PrAC set, prune a certain percent of the weights, reset and retrain the model, and repeat the process until we meet the sparsity requirement.

Turning PrAC set into actual training efficiency. Using PrAC sets for training can save training cost, firstly because less training data directly lead to fewer training iterations per training epoch. However, **the gains are way beyond linear** - since less training data could also imply easier fitting and faster convergence, e.g., less number of epochs. To fully leverage the potential of PrAC sets for efficient ticket finding, we introduce two training strategies for PrAC:

i) ***Dynamic training iterations***. After constructing the PrAC set, we will tune the training iterations according to the size of the PrAC set. We linearly scale down the number of iterations using the following formula to decide a new number of training iterations: $N = \frac{|\mathcal{P}|}{|\mathcal{D}_0|} N_0$, where $\mathcal{D}_0$ and $\mathcal{P}$ are the full training set and the PrAC set respectively, and N_0 is the original training iterations.

   In practice, we also tune the learning rate scheduler using the above adjustment formula to re-calculate the decay schedule for learning rates. By scaling down the required training iterations, we can gain training efficiency in a simple but meaningful way.

ii) ***Early stopping***. We build an early stopping mechanism upon the dynamic training iterations technique by introducing the Early Bird Ticket (You et al., 2020). It was originally designed for one-shot pruning; however, we reformulate and extend it to our iterative pruning context. As shown by You et al. (2020), winning tickets will emerge at the early period of the training process, which provides empirical support for using the early stopping technique. In our work, we calculate sparsity masks for the model after every epoch of training and monitor the distance between masks as a criterion for early stopping.

   The distance metric for matrices we use is the Hamming distance, *i.e.*, the number of different elements in two masks. Once the distance becomes smaller than a threshold, we interrupt the training, prune the network and update the sparsity mask, and use it for further retraining. The Hamming distances between masks at different sparsities on different architectures are shown in Figure 2. The graph validates the convergence of Hamming distance between sparsity masks at about half of training.

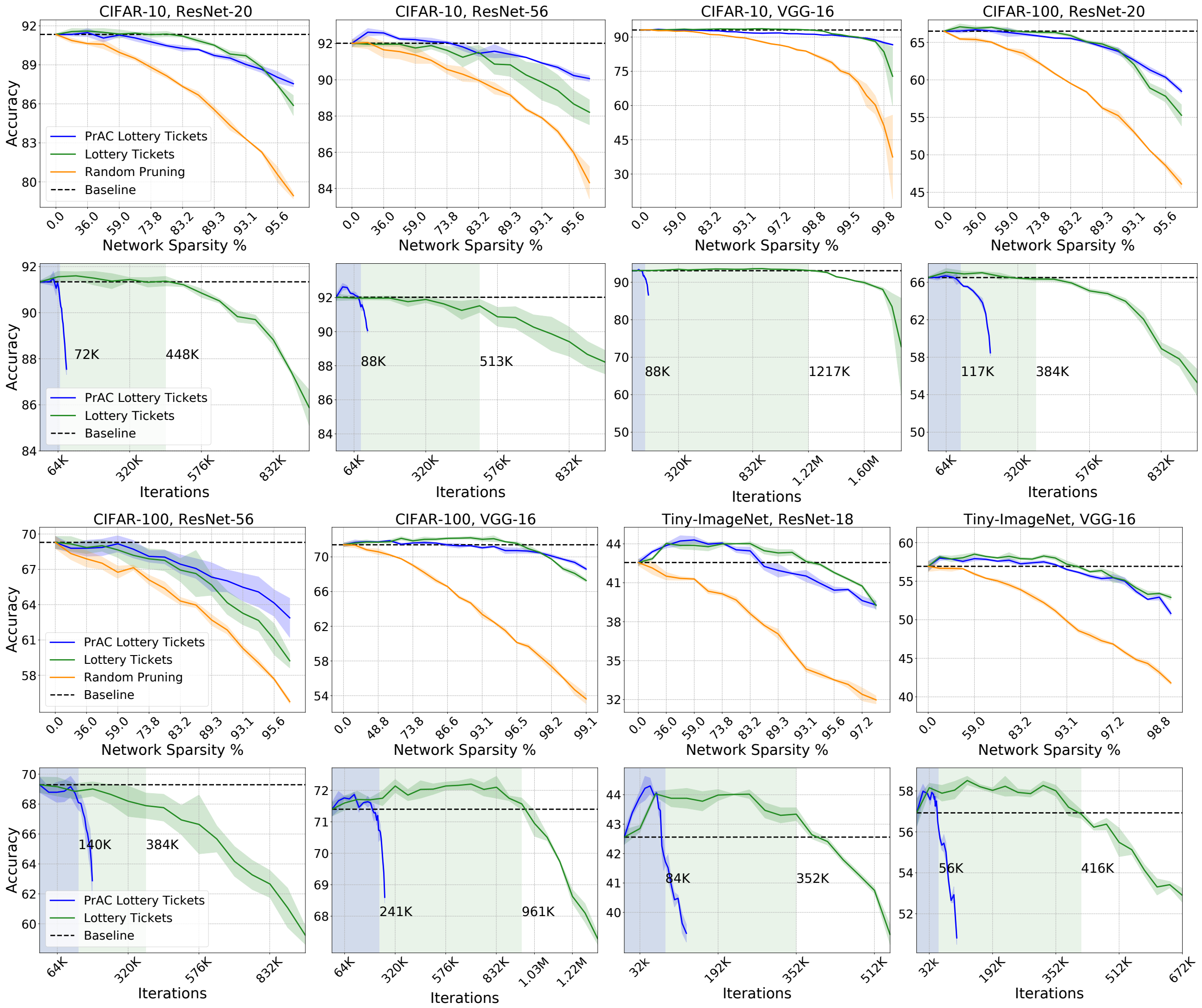

Figure 3. Testing accuracy of subnetworks at a range of sparsity levels from 0% to 99.85% (the first and third rows) and the training iterations for finding each subnetwork (the second and fourth rows) on CIFAR-10, CIFAR-100, and Tiny-ImageNet with ResNet-18, ResNet-20, ResNet-56, and VGG-16. Blue, Green, Orange and **Black** curves represent our PrAC lottery tickets, vanilla lottery tickets, random pruning, and dense network, respectively. The solid line and shading are the mean and standard deviation of testing accuracy. The numbers within figures are the iterations used to find subnetworks with the **same sparsity** and **comparable performance**, which indicate our achieved training resources saving. We consider PrAC lottery tickets to achieve a matched performance as vanilla lottery tickets when the performance of PrAC lottery tickets is within one standard deviation of the performance of vanilla lottery tickets.

Integrating the above two techniques with the PrAC set, we build our data-model sparsity co-design framework to efficiently find matching subnetworks, termed as *PrAC lottery ticket*, with much less training resources.

4. Experiments

General Setup. We summarize the key setups and hyperparameters of our implementation in Table 1, and refer readers to Appendix A1 for more details. Our experiments use two popular architectures, ResNet (He et al., 2016) and VGG (Simonyan & Zisserman, 2014), on three representative datasets, *i.e.*, CIFAR-10 (Krizhevsky et al.,

Table 1. Implementation Details. For ResNet-20 and ResNet-56, we adopt three different training settings: standard, *low* and *warmup* (Frankle et al., 2019a). The low variant means a lower learning rate, and the warmup variant adopts a warm-up method that linearly increases the learning rate from zero.

Network	Variant	Dataset	Batch Size	Learning Rate	Warmup
ResNet-20	Standard Low Warmup	CIFAR10 & CIFAR100	128	0.1 0.01 0.03	0 0 15 epochs
ResNet-56	Standard Low Warmup	CIFAR10 & CIFAR100	128	0.1 0.01 0.03	0 0 15 epochs
VGG-16	- -	CIFAR10 & CIFAR100 Tiny-ImageNet	128 512	0.1 0.1	0 0
ResNet-18	-	Tiny-ImageNet	512	0.1	0

2009), CIFAR-100 (Krizhevsky et al., 2009) and Tiny-ImageNet (Wu et al., 2017). Specifically, we train networks for 182 epochs with a multi-step learning rate schedule, which decays the learning rate to its one-tenth at epoch 91 and 136, respectively. We evaluate the quality of obtained subnetworks, *i.e.*, lottery tickets, by testing accuracy after independently trained from the same random initialization or early rewound weights (Frankle et al., 2019a). All reported results are averaged over three independent runs.

4.1. Identifying Winning Tickets with PrAC Sets

We evaluate our data and model sparsity co-design framework across diverse datasets and architectures with a total of eight combinations, specifically, CIFAR-10 with {ResNet-20, ResNet-56, VGG-16}, CIFAR-100 with {ResNet-20, ResNet-56, VGG-16}, and Tiny-ImageNet with {ResNet-18, VGG-16}. We consider vanilla lottery tickets (LT) method (Frankle & Carbin, 2018) and random pruning for comparisons. Figure 3 collects the achieved performance of subnetworks with different sparsity and their training effort for identifying each subnetwork, in terms of training iterations. Several observations can be drawn as follows:

- Our PrAC lottery tickets match the performance as vanilla lottery tickets in all combinations while notably less training costs, specifically achieving training iteration saving of 83.93% and 69.53% for ResNet-20, 82.85% and 63.54% for ResNet-56, 92.77% and 74.92% for VGG-16 on CIFAR-10 and CIFAR-100, respectively; 76.14% for ResNet-18, and 86.56% for VGG-16 on Tiny-ImageNet. As shown in Figure 3, we record the number of training iterations for the PrAC lottery tickets and the vanilla LT before reaching the highest sparsity that the former can match. And we color the area of the graph according to the number of training iterations for better demonstration.
- Somehow surprisingly, PrAC lottery tickets can even outperform the vanilla lottery tickets at some very high sparsity levels. This intriguing phenomenon implies that utilizing the data-level sparsity by PrAC sets, in addition to efficiency purpose, may even have additional regularization effects on improving the found model's generalization. We will leave further investigation for future work.
- The numbers of examples in PrAC sets across different datasets are adaptively varying. On CIFAR-10, the percentage of the number of the PrAC sets ranges from 35.32% to 37.07%, from 69.55% to 78.19% on CIFAR-100, and from 68.23% to 75.10% on Tiny ImageNet. The ratios of training iterations saved also vary between datasets. On CIFAR-10, we can save training iterations more than 80% but no more than 75% on CIFAR-100, which means that it requires more training effort to find PrAC lottery ticket on CIFAR-100 than CIFAR-10.
- Different architectures show the different percentage of training iteration saving and indicates the speed of matching subnetworks emerge. On VGG-16, our method can save the highest percentage of training iterations, indicating the highest speed to find lottery tickets. On ResNet-56 and ResNet-18, the speed to find lottery ticket is slower; On CIFAR-10 our framework can save 82.85% of training iterations on ResNet-56 while 92.77% on VGG-16; On Tiny ImageNet our framework can save 76.14% on ResNet-18 while 86.56% on VGG-16.

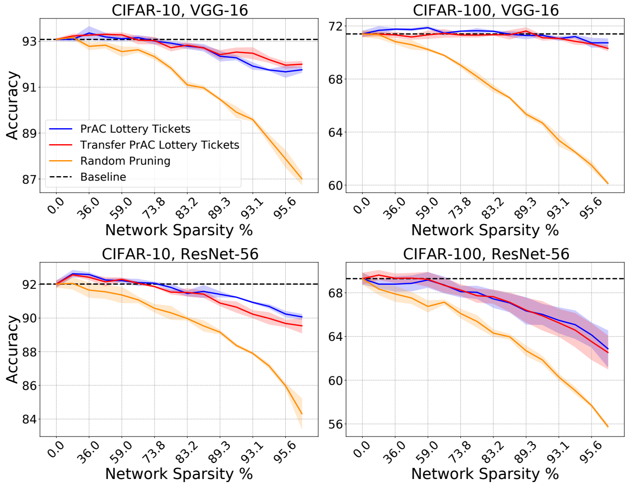

Figure 4. The transferability study of PrAC sets on CIFAR-10 and CIFAR-100. Blue, Red, Orange and **Black** curves represent our PrAC lottery tickets, PrAC tickets found with transferred PrAC sets, random pruning and full network. Each curve contains the mean and standard deviation of test accuracy of subnetworks.

4.2. PrAC Sets Are Transferable Across Models

The construction of PrAC sets seems model-dependent, relying on a given full dense network and pruned subnetworks. It motives us to investigate to what extent the PrAC sets depend on those factors. As shown in Figure 4, we conduct transferability studies of PrAC sets across network architectures. Specifically, taking ResNet-20 as the source architecture to build PrAC sets on CIFAR-10 and CIFAR-100, and then finding PrAC lottery tickets in ResNet-56 and VGG-16 (target architectures) with transferred PrAC sets.

Results in Figure 4 demonstrate that *Transfer PrAC Lottery Tickets* present competitive performance to *PrAC Lottery Tickets*. They show similar accuracies at most sparsity levels, and both surpass randomly pruned subnetworks by a significant performance margin. It demonstrates that PrAC sets are surprisingly transferable for identifying lottery tickets across diverse architectures, which opens up promising avenues of efficiently finding winning tickets in huge models with compact PrAC sets constructed by tiny networks.

4.3. Comparisons with Strong Baselines.

Core-set and active learning. Natural comparative baselines, *i.e.*, core-set and active learning approaches, are considered to assess the quality of PrAC sets further. In spe-

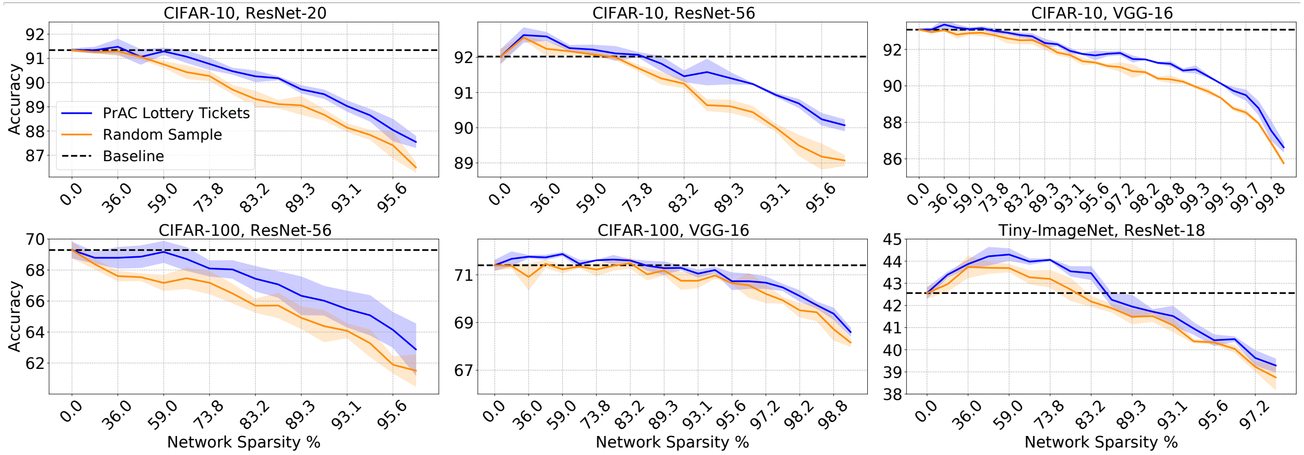

Figure 5. Comparison results of our PrAC lottery tickets with subnetworks identified with subsets of random sampling across different architectures and datasets. **More results** can be found at Figure A10.

cific, we adopt two representative methods, active learning via maximum entropy sampling (Lewis & Gale, 1994; Settles, 2012) termed as "Entropy", and core-set selection via proxy (Coleman et al., 2020) named as "SVP". Meanwhile, random sampling is designed for a sanity check. For fair comparisons, we keep the training iterations and the number of data in baselines consistent with our sparsity co-design approach. Figure A10 collects the achieved performance of independently trained subnetworks from different approaches on CIFAR-10 with ResNet-20 and Figure 5 further provides a comprehensive comparison with random sampling across different datasets and architectures. Results demonstrate that utilizing PrAC sets is capable of finding consistently better subnetworks with higher accuracies across diverse sparsity. It suggests that our co-design of data and model sparsity produces more informative pruning-aware subsets, which benefits to locate high-quality winning tickets.

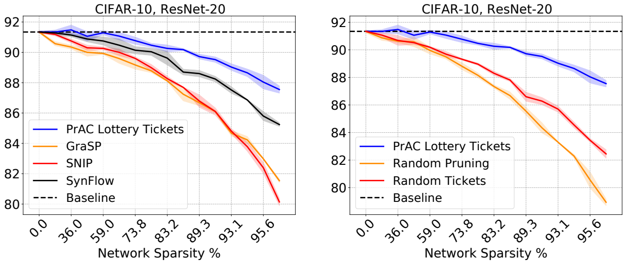

Figure 6. Comparison results with strong baselines. *Left:* Comparison of our PrAC lottery tickets with other pruning methods. *Right:* Comparison of our methods with random pruning or initialization.

Other efficient network pruning approaches. Recent proposed SNIP (Lee et al., 2018), GraSP (Wang et al., 2020), and SynFlow (Tanaka et al., 2020) aim to prune networks at initialization, thereby saving resources at training stages. They usually only require a single batch of training examples with certain effective pruning criterion to find subnetworks in one-shot, which can be enhanced with more data and training budgets (Wang et al., 2020; Tanaka et al., 2020). For fair comparisons, we implement these methods in an iterative manner (usually better than one-shot (Han et al., 2015; Frankle & Carbin, 2018)) with the training iterations and the number of training data consistent with our approaches. As shown in Figure 6 (*Left*), only our approach is able to identify winning tickets with matched performance to full unpruned models (*i.e.,* Baseline), and obtain a consistent performance margin compared to other pruning methods. Specifically, PrAC lottery tickets with 93.13% sparsity surpass SynFlow, SNIP, and GraSP by 1.51%, 4.22% and 4.36% test accuracy. With the computation consumption remains constant for all algorithms, this achieved significant performance gap verifies the superiority of our proposal.

Random tickets with random re-initialization. To exclude the possibility of trivial solutions, we consider the commonly adopted baseline, random tickets trained from randomly re-initialized weights, from the LTH literature (Frankle & Carbin, 2018). From Figure 6 (*Right*), we observe that PrAC lottery tickets hold overwhelming advantages. For example, with a 1% accuracy gap against the full model, our identified matching subnetworks with a sparsity of 79.03%, which are much sparser than both random pruning (48.80%) and random tickets (48.80%).

4.4. Ablation Study

The Two Components in the PrAC set. To investigate the individual effect of critical examples for training (CET) and pruning (CEP), we only utilize CET to identify matching subnetworks, as presented in Figure A12. Results show that without the assistance of CEP, the found subnetworks consistently incur $\sim 1\%$ performance drop. Table A3 collects the number of samples in CET and CEP. We observe that as the sparsity grows, the number of CEP keeps increasing; meanwhile, CEP shares fewer overlap images with CET, which indicates gradually detached distributions of critical samples during training and pruning.

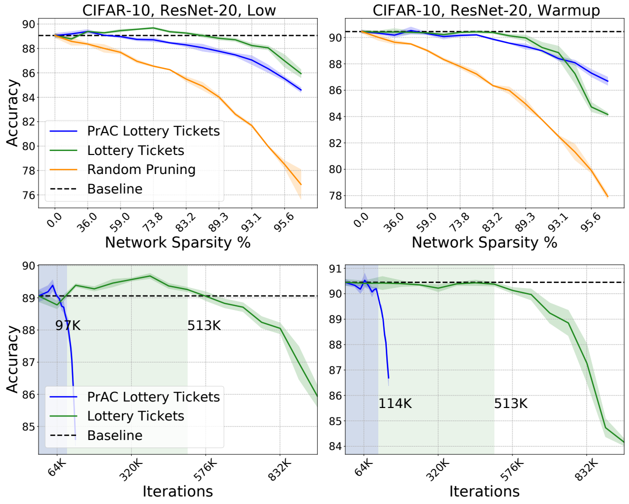

Figure 7. Testing accuracy of subnetworks at a range of sparsity levels from 0% to 96.48% (the first row) and the training iterations for finding each subnetwork (the second row) on CIFAR-10 with ResNet-20 under different lottery ticket settings. The numbers within figures are the iterations used to find the subnetworks with the **same sparsity** and **comparable performance**. **More results** can be found at Figure A11.

With or without early stopping. We adopt the early stopping (You et al., 2020) technique in our framework to find PrAC lottery tickets more efficiently. To understand its effect, we implement the variant, PrAC *w.o.* Early Stop, that disables the early stopping in our methods. As shown in Figure 8, we observe that PrAC *w.o.* Early Stop finds subnetworks with the same sparsity level and similar performance as vanilla lottery tickets, achieving 40.63% training resources saving. Equipped with the early stopping, PrAC lottery tickets at the same sparsity, obtain 83.93% training resources saving at the cost of $\leq 0.50\%$ accuracy loss.

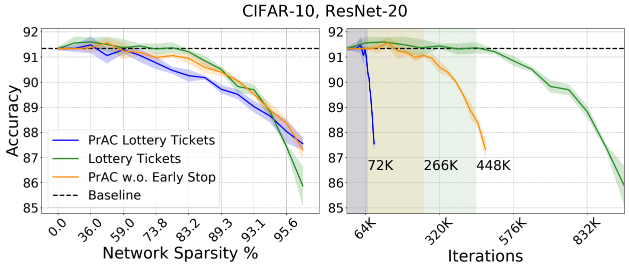

Figure 8. Ablation studies of vanilla lottery tickets and our PrAC lottery tickets *w/w.o.* early stopping on CIFAR-10 with ResNet-20. *Left:* Testing accuracy of subnetworks with different sparsity. *Right:* Training iterations for finding each subnetwork. And the numbers within the figure are the iterations used to find the corresponding subnetworks with the **same sparsity** and **comparable performance**. (72k, 266k, 448k represent PrAC lottery tickets, PrAC *w.o.* Early Stop and vanilla lottery tickets, respectively.)

Validating across diverse lottery ticket settings. Here we further evaluate our framework under two additional lottery ticket settings proposed by Frankle & Carbin (2018), *i.e.* **low** and **warmup**, with ResNet-20 and ResNet-56 on CIFAR-10 and CIFAR-100, respectively. Detailed hyperparameters are listed in Table 1. Figure 7 and A11 shows that to find subnetworks with similar performance under the **low** and **warmup** settings, our methods only cost $18.91\% \sim 22.22\%$ and $34.33\% \sim 38.01\%$ training resources on CIFAR-10 and CIFAR-100, compared to vanilla lottery tickets. These consistently achieved training savings further verify the efficiency of PrAC lottery tickets, and the effectiveness of our sparsity co-design framework.

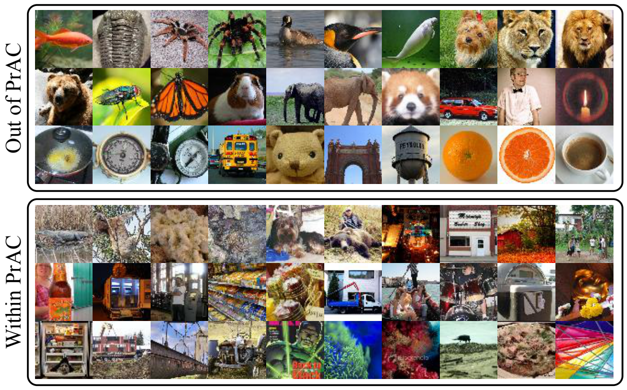

Figure 9. Visualization of examples out of (*upper*) and within (*bottom*) the final PrAC set on Tiny-ImageNet.

4.5. Visualization and Analyses of PrAC Sets

Visualization of examples out of and within PrAC sets on Tiny-ImageNet is provided in Figure 9, and the class-wise ratios of images in PrAC sets can be found in Figure A15. As shown in Figure 9, the images out of the PrAC set show less complexity where objects are centered and easily distinguishable, such as identifying an orange from white backgrounds. In contrast, the images within PrAC sets contain multiple ambiguous elements, including lower quality, depict multiple objects, complicated backgrounds and resulting in a challenging recognition even for a human. In addition, the distribution of PrAC set's classes in Figure A15 is quite balanced, where the number of images is in the same order. Such observations may provide possible insights on why PrAC sets are capable of locating critical subnetworks, *i.e.*, PrAC tickets, with satisfying performance.

5. Conclusion

In this paper, we explore a new perspective to finding lottery tickets more efficiently by doing so on small pruning-aware critical (PrAC) subsets, which is constructed via data and model sparsity co-design. Extensive experiments verify the effectiveness of our proposals with diverse network architectures on multiple common datasets, including CIFAR-10, CIFAR-100, and Tiny ImageNet. High-quality winning tickets, can be identified efficiently on such compact PrAC sets and enjoys significant training cost reduction.

References

Bachem, O., Lucic, M., and Krause, A. Practical coreset constructions for machine learning. *arXiv preprint arXiv:1703.06476*, 2017.

Chen, T., Frankle, J., Chang, S., Liu, S., Zhang, Y., Carbin, M., and Wang, Z. The lottery tickets hypothesis for supervised and self-supervised pre-training in computer vision models. *arXiv preprint arXiv:2012.06908*, 2020a.

Chen, T., Frankle, J., Chang, S., Liu, S., Zhang, Y., Wang, Z., and Carbin, M. The lottery ticket hypothesis for pretrained bert networks. *arXiv preprint arXiv:2007.12223*, 2020b.

Chen, T., Cheng, Y., Gan, Z., Liu, J., and Wang, Z. Ultra-data-efficient gan training: Drawing a lottery ticket first, then training it toughly. *arXiv preprint arXiv:2103.00397*, 2021a.

Chen, T., Sui, Y., Chen, X., Zhang, A., and Wang, Z. A unified lottery ticket hypothesis for graph neural networks. *arXiv preprint arXiv:2102.06790*, 2021b.

Chen, X., Zhang, Z., Sui, Y., and Chen, T. Gans can play lottery tickets too. In *International Conference on Learning Representations*, 2021c.

Coleman, C., Yeh, C., Mussmann, S., Mirzasoleiman, B., Bailis, P., Liang, P., Leskovec, J., and Zaharia, M. Selection via proxy: Efficient data selection for deep learning. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=HJg2b0VYDr.

Desai, S., Zhan, H., and Aly, A. Evaluating lottery tickets under distributional shifts. In *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP*, 2019.

Evci, U., Pedregosa, F., Gomez, A., and Elsen, E. The difficulty of training sparse neural networks. *arXiv*, abs/1906.10732, 2019.

Evci, U., Gale, T., Menick, J., Castro, P. S., and Elsen, E. Rigging the lottery: Making all tickets winners. In *International Conference on Machine Learning*, pp. 2943–2952. PMLR, 2020.

Frankle, J. and Carbin, M. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*, 2018.

Frankle, J., Dziugaite, G. K., Roy, D. M., and Carbin, M. Linear mode connectivity and the lottery ticket hypothesis. *arXiv*, abs/1912.05671, 2019a.

Frankle, J., Dziugaite, G. K., Roy, D. M., and Carbin, M. Stabilizing the lottery ticket hypothesis. *arXiv preprint arXiv:1903.01611*, 2019b.

Frankle, J., Dziugaite, G. K., Roy, D., and Carbin, M. Linear mode connectivity and the lottery ticket hypothesis. In *International Conference on Machine Learning*, pp. 3259–3269. PMLR, 2020a.

Frankle, J., Dziugaite, G. K., Roy, D. M., and Carbin, M. Pruning neural networks at initialization: Why are we missing the mark? *arXiv preprint arXiv:2009.08576*, 2020b.

Gale, T., Elsen, E., and Hooker, S. The state of sparsity in deep neural networks. *arXiv*, abs/1902.09574, 2019.

Gan, Z., Chen, Y.-C., Li, L., Chen, T., Cheng, Y., Wang, S., and Liu, J. Playing lottery tickets with vision and language. *arXiv preprint arXiv:2104.11832*, 2021.

Han, B., Niu, G., Yu, X., Yao, Q., Xu, M., Tsang, I., and Sugiyama, M. Sigua: Forgetting may make learning with noisy labels more robust. In *International Conference on Machine Learning*, pp. 4006–4016. PMLR, 2020.

Han, S., Mao, H., and Dally, W. J. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.

Har-Peled, S. and Kushal, A. Smaller coresets for k-median and k-means clustering. *Discrete & Computational Geometry*, 37(1):3–19, 2007.

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

Hooker, S., Courville, A., Clark, G., Dauphin, Y., and Frome, A. What do compressed deep neural networks forget? *arXiv preprint arXiv:1911.05248*, 2020a.

Hooker, S., Moorosi, N., Clark, G., Bengio, S., and Denton, E. Characterising bias in compressed models. *arXiv preprint arXiv:2010.03058*, 2020b.

Kalibhat, N. M., Balaji, Y., and Feizi, S. Winning lottery tickets in deep generative models, 2020.

Katharopoulos, A. and Fleuret, F. Not all samples are created equal: Deep learning with importance sampling. *arXiv preprint arXiv:1803.00942*, 2018.

Krizhevsky, A. et al. Learning multiple layers of features from tiny images. 2009.

Lee, N., Ajanthan, T., and Torr, P. H. Snip: Single-shot network pruning based on connection sensitivity. *arXiv preprint arXiv:1810.02340*, 2018.

Lewis, D. D. and Gale, W. A. A sequential algorithm for training text classifiers. In *SIGIR'94*, pp. 3–12. Springer, 1994.

Ma, H., Chen, T., Hu, T.-K., You, C., Xie, X., and Wang, Z. Good students play big lottery better. *arXiv preprint arXiv:2101.03255*, 2021.

Mehta, R. Sparse transfer learning via winning lottery tickets. *arXiv*, abs/1905.07785, 2019.

Mirzasoleiman, B., Bilmes, J., and Leskovec, J. Coresets for data-efficient training of machine learning models. In *International Conference on Machine Learning*, pp. 6950–6960. PMLR, 2020.

Mocanu, D. C., Mocanu, E., Stone, P., Nguyen, P. H., Gibescu, M., and Liotta, A. Scalable training of artificial neural networks with adaptive sparse connectivity inspired by network science. *Nature communications*, 9 (1):1–12, 2018.

Morcos, A., Yu, H., Paganini, M., and Tian, Y. One ticket to win them all: generalizing lottery ticket initializations across datasets and optimizers. In *Advances in Neural Information Processing Systems 32*, 2019.

Mostafa, H. and Wang, X. Parameter efficient training of deep convolutional neural networks by dynamic sparse reparameterization. In *International Conference on Machine Learning*, pp. 4646–4655. PMLR, 2019.

Renda, A., Frankle, J., and Carbin, M. Comparing rewinding and fine-tuning in neural network pruning. In *8th International Conference on Learning Representations*, 2020.

Sener, O. and Savarese, S. Active learning for convolutional neural networks: A core-set approach. *arXiv preprint arXiv:1708.00489*, 2017.

Settles, B. Active learning literature survey. 2009.

Settles, B. Active learning. *Synthesis lectures on artificial intelligence and machine learning*, 6(1):1–114, 2012.

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

Tanaka, H., Kunin, D., Yamins, D. L., and Ganguli, S. Pruning neural networks without any data by iteratively conserving synaptic flow. *arXiv preprint arXiv:2006.05467*, 2020.

Tang, Y., Wang, Y., Xu, Y., Tao, D., Xu, C., Xu, C., and Xu, C. Scop: Scientific control for reliable neural network pruning. *arXiv preprint arXiv:2010.10732*, 2020.

Toneva, M., Sordoni, A., des Combes, R. T., Trischler, A., Bengio, Y., and Gordon, G. J. An empirical study of example forgetting during deep neural network learning. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=BJlxm30cKm.

Tsang, I. W., Kwok, J. T., and Cheung, P.-M. Core vector machines: Fast svm training on very large data sets. *Journal of Machine Learning Research*, 6(Apr):363–392, 2005.

Voulodimos, A., Doulamis, N., Doulamis, A., and Protopapadakis, E. Deep learning for computer vision: A brief review. *Computational intelligence and neuroscience*, 2018, 2018.

Wang, C., Zhang, G., and Grosse, R. Picking winning tickets before training by preserving gradient flow. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SkgsACVKPH.

Wang, T., Zhu, J.-Y., Torralba, A., and Efros, A. A. Dataset distillation. *arXiv preprint arXiv:1811.10959*, 2018.

Wu, J., Zhang, Q., and Xu, G. Tiny imagenet challenge. Technical report, 2017.

Xia, X., Liu, T., Han, B., Gong, C., Wang, N., Ge, Z., and Chang, Y. Robust early-learning: Hindering the memorization of noisy labels. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=Eql5b1_hTE4.

Yao, Q., Yang, H., Han, B., Niu, G., and Kwok, J. T.-Y. Searching to exploit memorization effect in learning with noisy labels. In *International Conference on Machine Learning*, pp. 10789–10798. PMLR, 2020.

You, H., Li, C., Xu, P., Fu, Y., Wang, Y., Chen, X., Baraniuk, R. G., Wang, Z., and Lin, Y. Drawing early-bird tickets: Toward more efficient training of deep networks. In *8th International Conference on Learning Representations*, 2020.

Yu, H., Edunov, S., Tian, Y., and Morcos, A. S. Playing the lottery with rewards and multiple languages: lottery tickets in rl and nlp. In *8th International Conference on Learning Representations*, 2020.

Zhao, P. and Zhang, T. Stochastic optimization with importance sampling for regularized loss minimization. In *international conference on machine learning*, pp. 1–9, 2015.

Zhou, H., Lan, J., Liu, R., and Yosinski, J. Deconstructing lottery tickets: Zeros, signs, and the supermask. *arXiv preprint arXiv:1905.01067*, 2019.