Finding Meta Winning Ticket to Train Your MAML

Dawei Gao
Alibaba Group
gaodawei.gdw@alibaba-inc.com

Yuexiang Xie
Alibaba Group
yuexiang.xyx@alibaba-inc.com

Zimu Zhou
Singapore Management University
zimuzhou@smu.edu.sg

Zhen Wang
Alibaba Group
jones.wz@alibaba-inc.com

Yaliang Li
Alibaba Group
yaliang.li@alibaba-inc.com

Bolin Ding
Alibaba Group
bolin.ding@alibaba-inc.com

ABSTRACT

The lottery ticket hypothesis (LTH) states that a randomly initialized dense network contains sub-networks that can be trained in isolation to the performance of the dense network. In this paper, to achieve rapid learning with less computational cost, we explore LTH in the context of meta learning. First, we experimentally show that there are sparse sub-networks, known as meta winning tickets, which can be meta-trained to few-shot classification accuracy to the original backbone. The application of LTH in meta learning enables the adaptation of meta-trained networks on various IoT devices with fewer computation. However, the status quo to identify winning tickets requires iterative training and pruning, which is particularly expensive for finding meta winning tickets. To this end, then we investigate the inter- and intra-layer patterns among different meta winning tickets, and propose a scheme for early detection of a meta winning ticket. The proposed scheme enables efficient training in resource-limited devices. Besides, it also designs a lightweight solution to search the meta winning ticket. Evaluations on standard few-shot classification benchmarks show that we can find competitive meta winning tickets with 20% weights of the original backbone, while incurring only 8%-14% (Conv-4) and 19%-29% (ResNet-12) computation overhead (measured by FLOPs) of the standard winning ticket finding scheme.

CCS CONCEPTS

• **Computing methodologies** → *Machine learning approaches.*

KEYWORDS

Meta Learning; Network Pruning; Lottery Ticket Hypothesis

ACM Reference Format:
Dawei Gao, Yuexiang Xie, Zimu Zhou, Zhen Wang, Yaliang Li, and Bolin Ding. 2022. Finding Meta Winning Ticket to Train Your MAML. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '22), August 14–18, 2022, Washington, DC, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3534678.3539467

1 INTRODUCTION

Meta learning is a prevailing paradigm for rapid learning of new tasks from limited data [17, 34]. Specifically, a backbone is meta-trained to generate a weight initialization that can be adapted to unseen tasks in few shots [9]. Such capability of fast adaptation has become increasingly important in Internet-of-Things (IoT) applications such as autonomous cars, personal robots, and smart home appliances to deliver adaptive and personalized services.

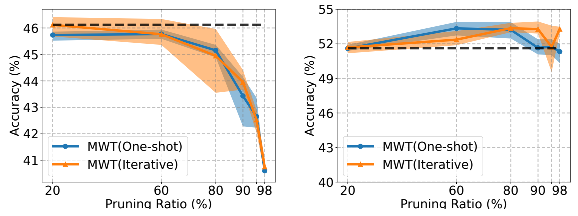
However, it is non-trivial to bring fast model adaptation to IoT platforms, since the backbones widely adopted in meta learning [9, 16] easily overwhelm the memory system of modern IoT devices, which is typically several *KB* to *MB* [3]. The existing meta learning schemes do not alter the backbone architecture [8].

To tackle such issue, in this study, we explore pruning the backbones for meta learning via the lens of the Lottery Ticket Hypothesis (LTH) [10]. In particular, we focus on **the existence and fast identification of sparse sub-networks within a dense backbone, known as meta winning tickets, that can be meta-trained in isolation by prior gradient-based meta learning methods to comparable few-shot learning accuracy as the original backbone.**

The motivation of our problem scope can be summarized as: (1) Gradient-based meta learning algorithms such as model-agnostic meta learning (MAML) [9] and its variants [22, 24] are fit for fast adaptation in IoT applications since they support diverse learning tasks and recent studies have shown the feasibility of gradient-based training on memory-constrained platforms [13]. (2) LTH [10], which was first revealed in vanilla supervised learning, has been extended to multiple other learning paradigms [4, 5, 37]. It wide applicability implies that it may also hold for meta learning. (3) The status quo to identify the sparse sub-network, *i.e.*, the winning ticket, involves pre-training, pruning, and re-training [10]. Such a process can be expensive for detecting a meta winning ticket, due to massive meta updates and the calculation of second-order derivatives in MAML and its variants. Early detection of meta winning tickets may notably reduce the workload to obtain a compact, meta-trained network.

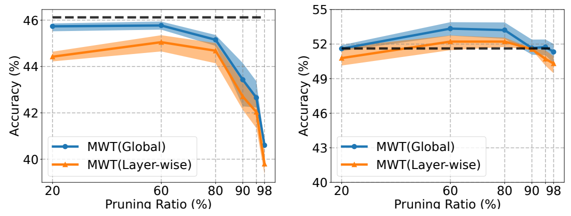
We take an empirical approach to investigating the existence and early detection of meta winning tickets, and experiment with diverse backbones [9, 16], meta learning algorithms [9, 22, 24], and few-shot learning benchmarks [26, 28]. Our main contributions and results are summarized as follows:

- Meta winning tickets, *i.e.*, sparse sub-networks within the original backbone with competitive few-shot classification performance, empirically exist, and they can be extracted



(a) Conv-4 (b) ResNet-12

Figure 1: Comparison of iterative and one-shot pruning for finding meta winning tickets. Results obtained with MAML, *mini*ImageNet, 5-way-1-shot.



(a) Conv-4 (b) ResNet-12

Figure 2: Comparison of layer-wise and global pruning for finding meta winning tickets. Results obtained with MAML, *mini*ImageNet, 5-way-1-shot.

via the standard winning ticket finding method as in the original LTH [10].

- Meta winning tickets exhibit distinctive characteristics from the winning tickets in vanilla supervised learning. Specifically, multiple meta winning tickets derived from the same backbone share similar per-layer pruning ratios, and have large overlaps in their mask locations in the first layer, which we call as *primary masks*.
- Noticing that the primary masks of a meta winning ticket emerge early during meta training, we propose PMT (primary masked ticket), a simple algorithm that approximately identifies a meta winning ticket once the masks in the first layer stabilize.
- Evaluations show that PMT accelerates meta winning ticket finding by 7.02 to 16.67× on Conv-4, and 3.51 to 6.15× on ResNet-12. It also reduces the computation overhead of meta-training a sparse network by 2.07 to 2.78× on Conv-4 and 1.77 to 1.94× on ResNet-12.

2 PROBLEM STATEMENT

The lottery ticket hypothesis (LTH) [10] is first uncovered for vanilla supervised learning in image classification tasks and has recently been investigated in different learning paradigms and tasks [4, 5, 12, 37]. In this section, we state our hypothesis in the context of meta learning.

2.1 Meta Learning Primer

Meta learning proves effective for few-shot learning [34]. We focus on gradient-based meta learning [9, 22, 24] for its applicability in diverse tasks [17]. For ease of discussion, we first briefly review model-agnostic meta-learning (MAML) [9], a prevailing gradient-based meta-learning algorithm in the context of few-shot classification.

MAML enables fast adaptation to unseen classes, *i.e.*, few-shot classification by learning from abundant few-shot classification episodes. It meta-trains a *backbone* $f(\boldsymbol{\theta}_0)$ with initial parameters $\boldsymbol{\theta}_0$ to generate a model $f(\boldsymbol{\theta}_M)$ such that the meta-trained parameters $\boldsymbol{\theta}_M$ can adapt to novel classes with few training samples. Meta-training is formulated as a two-tier optimization process upon large amounts of tasks $\{\mathcal{T}_n\}$ sampled from a distribution $p(\mathcal{T})$ via stochastic gradient descent. For each sampled task within $\{\mathcal{T}_n\}$, there exists a dataset $\mathcal{D}_n = \{\mathcal{S}_n, \mathcal{Q}_n\}$ with $\mathcal{S}_n$ (support set) for training and $\mathcal{Q}_n$ (query set) for testing.

During the k-*th* iteration of meta-training, also known as *meta epoch*, MAML samples N tasks from $p(\mathcal{T})$ and updates the parameters $\boldsymbol{\theta}_k$ as:

$$\text{inner loop:} \quad \boldsymbol{\theta}_k^n = \boldsymbol{\theta}_k - \alpha \nabla_{\boldsymbol{\theta}_k} \mathcal{L}\left(\boldsymbol{\theta}_k; \mathcal{S}_n\right), \tag{1}$$

$$\text{outer loop:} \quad \boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \frac{\beta}{N} \sum_n \nabla_{\boldsymbol{\theta}_k} \mathcal{L}\left(\boldsymbol{\theta}_k^n; \mathcal{Q}_n\right), \tag{2}$$

where α and β represent the inner and outer loop learning rates respectively, and $\mathcal{L}$ denotes the loss function. In the inner loop, the *task-specific* parameters $\boldsymbol{\theta}_k^n$ are optimized from the meta initialization $\boldsymbol{\theta}_k$ on the support set for a few update steps. In the outer loop, $\boldsymbol{\theta}_k$ is updated according to the averaged loss across the sampled tasks on the query set. It is worth pointing out that meta-training demands massive meta updates due to the training instability [2], and involves computation-intensive calculations of the second order derivatives $\boldsymbol{\theta}_k$ [2, 24], which makes meta training rather costly. Recent studies such as ANIL [24] and BOIL [22] show the viability to omit the inner loop for the body or head of the backbone without degrading few-shot accuracy, which in effect reduces the meta-training overhead.

2.2 Meta Winning Ticket Hypothesis

According to LTH [10], a dense network $f(\boldsymbol{\theta}_0)$ with proper initialization $\boldsymbol{\theta}_0$ contains a sparse sub-network $f(\boldsymbol{\theta}_0 \odot \boldsymbol{m})$, where $\boldsymbol{m}$ is a binary mask and $\odot$ denotes element-wise product, such that training $f(\boldsymbol{\theta}_0 \odot \boldsymbol{m})$ in isolation can match the performance with training $f(\boldsymbol{\theta}_0)$. For complex networks, LTH holds by rewinding the parameters to those after k training iterations *i.e.*, $\boldsymbol{\theta}_k$, rather than the initial $\boldsymbol{\theta}_0$ [11]. $\boldsymbol{\theta}_k \odot \boldsymbol{m}$ that yields comparable performance to the dense network is called a *winning ticket* [10, 11].

We explore LTH in gradient-based meta learning [9, 22, 24]. Specifically, we hypothesize that there exists *meta winning ticket* $\boldsymbol{\theta}_k \odot \boldsymbol{m}$ in the dense backbone $f(\boldsymbol{\theta}_0)$ such that meta-training $f(\boldsymbol{\theta}_k \odot \boldsymbol{m})$ as in Eq. (1) and Eq. (2) achieves similar few-shot classification accuracy as meta-training $f(\boldsymbol{\theta}_0)$. We focus on *(i)* the *empirical existence* of meta winning tickets (Sec. 3) and *(ii)* their *efficient detection algorithms* (Sec. 4).

3 EXISTENCE OF META WINNING TICKETS

In this section, we empirically show meta winning tickets by applying magnitude based weight pruning, the status quo to identify winning tickets in vanilla supervised training [10]. The magnitude

Table 1: Hyper-parameters setups for MAML, ANIL and BOIL.

Method	Dataset	Backbone	Outer Loop			Inner Loop		
			LR	Optimizer	Iterations	LR	Optimizer	Update Steps
MAML	*mini*ImageNet	Conv-4	0.001	Adam	80,000	0.01	SGD	5
		ResNet-12	0.001	Adam	80,000	0.01	SGD	5
	*tiered*ImageNet	Conv-4	0.001	Adam	80,000	0.01	SGD	5
		ResNet-12	0.001	Adam	80,000	0.01	SGD	5
ANIL	*mini*ImageNet	Conv-4	0.001	Adam	200,000	0.01	SGD	5
		ResNet-12	0.001	Adam	200,000	0.01	SGD	5
BOIL	*mini*ImageNet	Conv-4	0.001	Adam	200,000	1	SGD	1
		ResNet-12	0.001	Adam	200,000	1	SGD	1

based weight pruning method has also been adopted to find winning tickets in other domains [4, 5, 12, 37].

3.1 Measurement Settings

Previous studies [10] propose to find a winning ticket $\boldsymbol{\theta}_k \odot \boldsymbol{m}$ in a dense network $f(\boldsymbol{\theta}_0)$ with initialization $\boldsymbol{\theta}_0$ via magnitude based pruning [14]. We adapt the procedure to gradient-based meta-learning and evaluate the performance of tickets on widely-used few-shot classification benchmarks. The learning rates adopted in the inner and outer loop follow the settings in [9, 22, 24].

Note that there are different pruning settings in the original LTH, including one-shot or iterative, global or layer-wise, and choices of learning rate. We first conduct experiments to compare different pruning settings. Specifically, Fig. 1 shows the comparison between one-shot pruning and iterative pruning with MAML on *mini*ImageNet (5-way-1-shot setting). For both Conv-4 and ResNet-12, iterative pruning and one-shot pruning yield similar performances. However, the computation cost of iterative pruning is nearly 7 times that of one-shot pruning. We choose one-shot pruning since one application of our solution is to reduce the workload of sparse meta training. Fig. 2 shows the comparison between global pruning and layer-wise pruning with MAML on *mini*ImageNet (5-way-1-shot setting). For both Conv-4 and ResNet-12, the global pruning outperforms layer-wise pruning with all pruning ratios by about 1%. The possible reason is the global pruning can adjust the pruning ratios for different layers adaptively, and the layer-wise pruning is just a special case of the global pruning. Based on the above observations, in this study we conduct experiments with *one-shot global* pruning.

Meta Learning Setups. We evaluate three gradient-based *meta learning algorithms*, whose different datasets and backbones are summarized in Table 1, including **MAML** [9]: the two-tier optimization as Eq. (1) and Eq. (2); **ANIL** [24]: omitting MAML's inner loop for backbone *head*; and **BOIL** [22]: omitting MAML's inner loop for backbone *body*.

We adopt two two widely-used *backbone architectures* $f(\cdot)$ for meta learning, including: **Conv-4** [9]: It is a 4-layer convolutional network with 3×3 convolutions followed by batch normalisation, ReLU, and 2×2 max-pooling; and **ResNet-12** [16]: It contains 4 residual blocks, each containing three 3×3 convolutional layers. In each residual block, the first two convolution layers are followed by batch normalisation and ReLu, and the last convolution layer is followed by batch normalisation and a skip connection. A 2×2 max-pooling is used after each residual block. The number of filters in each residual block is 64, 128, 256, and 512. The backbones are initialized as $f(\boldsymbol{\theta}_0)$ by the Kaiming Normal [15], a widely-used initialization for deep neural networks [16, 19, 25, 27].

```
Algorithm 1: MWT: find a meta winning ticket
Input: $\theta_0$: initial weights; $p(\mathcal{T})$: task distribution; $\mathcal{M}$: meta
       training algorithm; $p\%$: pruning ratio; $s$: random
       seed;
Output: A winning ticket $\theta_k \odot m$
1 Initialize $\theta$ with $\theta_0$
2 while $\theta$ not converge do
3     Sample batch of tasks $\{\mathcal{T}_i\} \sim p(\mathcal{T})$ by random seed $s$
4     Meta train $\theta$ on tasks $\{\mathcal{T}_i\}$ by $\mathcal{M}$
5 end
6 Prune $p\%$ weights of $\theta$ with small magnitude, and obtain the
    mask $m$
7 Rewind weight back to $\theta_k$
8 Return the ticket $\theta_k \odot m$
```

Pruning Setups. Following previous studies [10], the standard method (**MWT**) to find a meta winning ticket $\boldsymbol{\theta}_k \odot \boldsymbol{m}$ given $f(\boldsymbol{\theta}_0)$ and a pruning ratio $p\%$ (percentage of weights to be pruned) contains three steps: *(i)* Train $f(\boldsymbol{\theta}_0)$ by a meta-learning algorithm for M meta epochs to obtain $\boldsymbol{\theta}_M$; *(ii)* Prune $p\%$ weights based on their magnitude to get a mask $\boldsymbol{m}$; *(iii)* Rewind the weights back to $\boldsymbol{\theta_k}$, which yields a ticket $\boldsymbol{\theta}_k \odot \boldsymbol{m}$; *(iv)* Retrain the ticket by the meta-learning algorithm to resume performance. We provide the pseudo code of MWT in Algorithm 1.

In order to confirm the tickets found by **MWT** are competitive, we compare **MWT** with other pruning schemes including (1) **Random Re-initialization (RR)**, which is the same as **MWT** except that it re-initializes the weights with a new random Kaiming Normal when retaining (*i.e.*, step *(iv)*); and (2) **Random Mask (RM)**, which generates a random mask across different layers rather than pruning based on the magnitude of the meta-trained weights.

Benchmarks. We evaluate meta winning tickets on two few-shot classification benchmarks: *mini*ImageNet and *tiered*ImageNet:

- *mini*ImageNet: It is a subset of ImageNet for image classification. It consists of 60,000 colour images with 100 classes, each having 600 images of size 84×84 [26]. We follow the

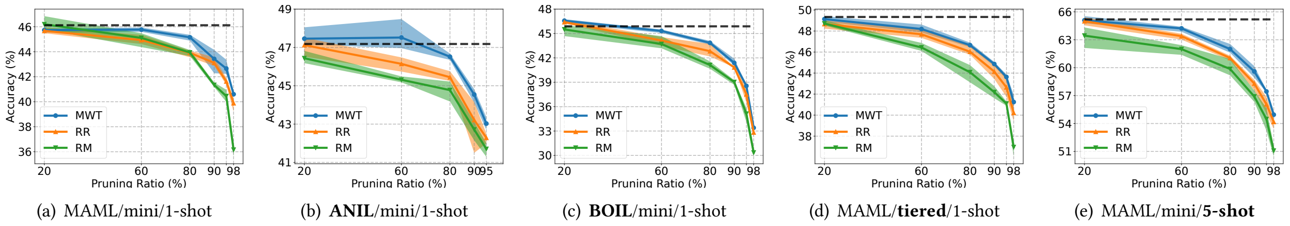


Figure 3: Few-shot accuracy of tickets extracted from Conv-4 backbone architecture via different methods.

(a) MAML/mini/1-shot (b) **ANIL**/mini/1-shot (c) **BOIL**/mini/1-shot (d) MAML/**tiered**/1-shot (e) MAML/mini/**5-shot**

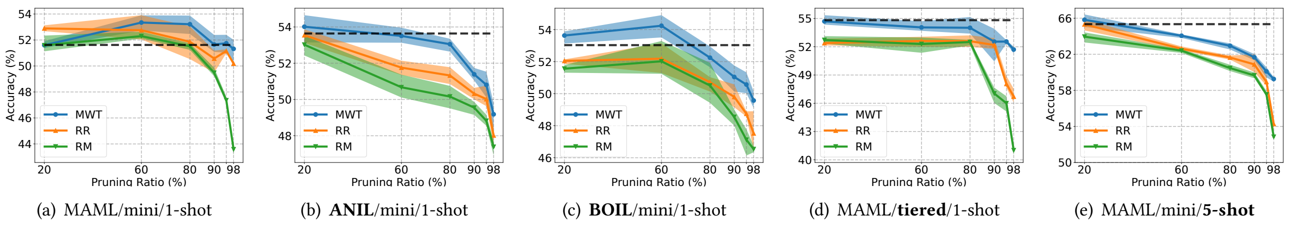


Figure 4: Few-shot accuracy of tickets extracted from ResNet-12 backbone architecture via different methods.

(a) MAML/mini/1-shot (b) **ANIL**/mini/1-shot (c) **BOIL**/mini/1-shot (d) MAML/**tiered**/1-shot (e) MAML/mini/**5-shot**

original data partition, where the dataset is divided into 64 training classes, 12 validation classes and 24 test classes.

- *tiered*ImageNet: It is a larger image classification dataset with 608 classes and 779, 165 images [28]. This dataset first clusters similar classes into the same category. Then it is divided into 20 categories (351 classes) for training, 6 categories (97 classes) for validation and 8 categories (160 classes) for testing.

In our experiments, we adopt the 5-way 1-shot setting by default and also test 5-way 5-shot for comparison.

3.2 Observations

Since a winning ticket consists of the rewind weight $\boldsymbol{\theta}_k$ and the mask matrix $\boldsymbol{m}$, we are interested in their impact on a meta winning ticket.

Impact of Mask *m*. As shown in Fig. 3 and Fig. 4, we plot the few-shot classification accuracy of the network with various combinations of meta-training methods (MAML, ANIL, BOIL), and pruning methods (MWT, RR, RM) on different backbones (Conv-4 and ResNet-12), datasets (*mini*ImageNet and *tiered*ImageNet) and few-shot settings (5-way-1-shot and 5-way-5-shot). The bold dotted line in the figures denotes the accuracy of the originally meta-trained backbone without pruning. Overall, MWT is able to identify tickets with 80% of weights pruned, while incurring a few-shot accuracy drop within 3% across backbones, datasets, and few-shot settings.

At the same pruning ratios, the tickets found by RR incur an accuracy drop of 4%, and those found by RM introduce an accuracy drop of 5%. For example, at pruning ratio 80% with Conv-4, *mini*ImageNet and 5-way 1-shot settings, MWT only incurs 1% drop in accuracy, while the accuracies of RR and RM decrease by over 3%. From the experimental results we can conclude that MWT identifies competitive tickets in meta learning, so-called meta winning tickets. Another important observation is that RM performs notably worse than RR, which highlights the contribution of mask m to a meta winning ticket.

Impact of Rewind *k*. Previous studies [11] show that the optimal rewind k may differ for small and large networks in vanilla supervised training. To evaluate the impact of rewind in meta learning, we conduct experiments on *mini*ImageNet with MAML for meta-training and MWT for identifying the tickets, with pruning ratios from 20% and 98%. The experimental results, including the few-shot accuracy with different rewind k for Conv-4 and ResNet-12, are illustrated in Fig. 5. It can be observed that an overly large rewind k tends to induce notable accuracy drop at all pruning ratios. With a small rewind k (greater than zero), the accuracy at certain pruning ratios might increase, and the increments become marginal at high pruning ratios, *e.g.*, 95%.

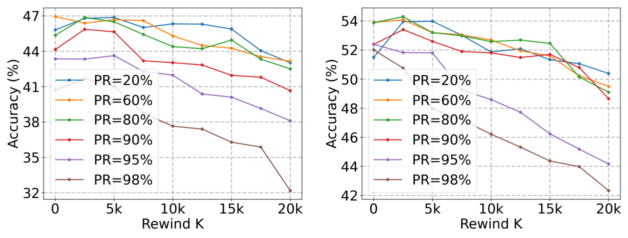
Takeaways. The experiments show that (1) Winning tickets empirically exist in meta learning in few-shot learning benchmarks, and they can be identified by the standard methods for finding winning tickets in vanilla supervised training; (2) The mask m is essential in a meta winning ticket $\boldsymbol{\theta}_k \odot \boldsymbol{m}$, yet the rewind k is less important. We can set k to zero without significantly damaging the performance of meta winning tickets, particularly at high pruning ratios.

4 FINDING META WINING TICKET EARLY

We further explore methods to identify meta winning tickets earlier than applying the standard MWT (*i.e.*, Algorithm 1). Such methods hold promise to notably reduce the workload of sparse meta-training (see Sec. 5). Inspired by the observation that there exist multiple winning tickets in a backbone [7], we propose to identify a meta winning ticket early via exploiting the commonality among meta winning tickets.

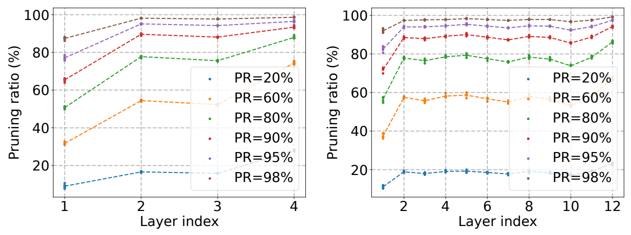
4.1 Obtaining Multiple Meta Winning Tickets

We first conduct experiments to find multiple meta winning tickets from the same backbone, and then analyze the commonality among them. Specifically, we repeat the meta training steps in Algorithm 1 (*i.e.*, lines 2-4) using the same initialization $\boldsymbol{\theta}_0$ but different random seeds, and obtain a series of well-trained networks $\{\boldsymbol{\theta}_T^1, \cdots, \boldsymbol{\theta}_T^g\}$.



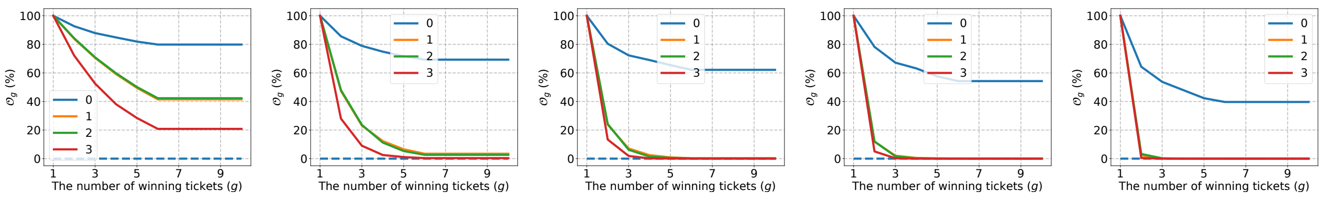
(a) Conv-4 (b) ResNet-12

Figure 5: Impact of rewind k.



(a) Conv-4 (b) ResNet-12

Figure 6: Distributions of per-layer pruning ratios of different meta winning tickets.



(a) $PR = 20\%$ (b) $PR = 60\%$ (c) $PR = 80\%$ (d) $PR = 90\%$ (e) $PR = 98\%$

Figure 7: Overlap of per-layer mask locations among different meta winning tickets for Conv-4 backbone.

Then we perform magnitude-based pruning on them to generate different winning tickets, denoted as $\{\boldsymbol{\theta}_0 \odot \boldsymbol{m}^1, \cdots, \boldsymbol{\theta}_0 \odot \boldsymbol{m}^g\}$. We generate 10 meta winning tickets for the Conv-4 and ResNet-12 backbone respectively, and report the detailed performance of them in Appendix A.1. The results show that the behaviors of these meta winning tickets are similarly as the those shown in Fig. 3(a) and Fig. 4(a).

4.2 Mask Patterns among Meta Winning Tickets

Further, we analyze the commonality among the meta winning tickets. Since these tickets shares the same $\boldsymbol{\theta}_0$, we focus on the patterns of their masks $\{\boldsymbol{m}^1, \cdots, \boldsymbol{m}^g\}$.

Inter-Layer Mask Patterns. Following previous study [31], which shows the importance of layer-wise pruning ratios [31], we investigate the per-layer pruning ratios of different meta winning tickets. Specifically, for a preset global pruning ratio $p\%$ in Algorithm 1, we plot the pruning ratios at each layer for the 10 meta winning tickets obtained from the Conv-4 and ResNet-12 backbones, as in Fig. 6. It can be observed that given a global pruning ratio, the per-layer pruning ratios of meta winning tickets are almost the same. Similar phenomena can be observed in the conducted experiments using different global pruning ratios.

The observations above imply the feasibility to configure the per-layer pruning ratios of a meta winning ticket via a *data-independent* scheme [31]. However, the per-layer pruning ratios alone fail to identify a meta winning ticket (see empirical evidence and discussion in Sec. 5), which motivates us to further investigate intra-layer mask patterns.

Intra-Layer Mask Patterns. Among the meta winning tickets, we are interested in their mask locations at each layer *i.e.*, where $\boldsymbol{m}_i = 1$. We define O_g as the percent of overlapped mask locations within the first g meta winning tickets $\{\boldsymbol{\theta}_0 \odot \boldsymbol{m}^1, \cdots, \boldsymbol{\theta}_0 \odot \boldsymbol{m}^g\}$ as $O_g := \frac{\left|\{i \mid \boldsymbol{m}_i^j = 1, \forall 1 \le j \le g\}\right|}{||\boldsymbol{m}||_0} \times 100\%$. We show the values of O_g at each layer with different pruning ratios for 10 meta winning tickets for the Conv-4 backbone in Fig. 7. Similar results for ResNet-12 are in Appendix A.2. From the results of O_g we can conclude: (1) Given a global pruning ratio, the overlapped mask locations among tickets deceases at deeper layers. (2) With high global pruning ratios, the overlapped mask locations almost only exist in the first layer. For example, when $PR \ge 60\%$, the overlapped mask locations drop to zero in the layers other than the first. When $PR = 98\%$, the overlap is still 40% in the first layer.

Observing such distinction between the first and the other layers, we hypothesize that the shared mask locations in the first layer, named as *primary masks*, are crucial for a meta winning ticket. To confirm our hypothesis, we empirically show the difference between the shared masks in the first layer (called primary masks) and the other layers (called secondary masks). For a given global pruning ratio, we first obtain the per-layer pruning ratios (see Sec. 4.3), and then create the following tickets:

- **MWT**: it is the meta winning tickets, which contains both the primary and secondary masks.
- **SR**: the mask locations in all layers are randomly picked.
- **OMT**: it randomly picks mask locations in the first layer and uses the secondary masks for other layers.
- **PM**: it adopts the primary masks in the first layer while the other mask locations are randomly picked.

As shown in Fig. 8, we plot the few-shot accuracy of these tickets under the settings of MAML, *mini*ImageNet, and 5-way-1-shot. MWT outperforms all other tickets when pruning more than 60% weights. We observe that PM outperforms SR, which implies the importance of primary masks. We also notice that OMT achieves similar performance with the random ticket SR, which implies the secondary masks are replaceable for a meta winning ticket. Based

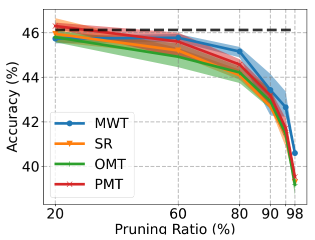
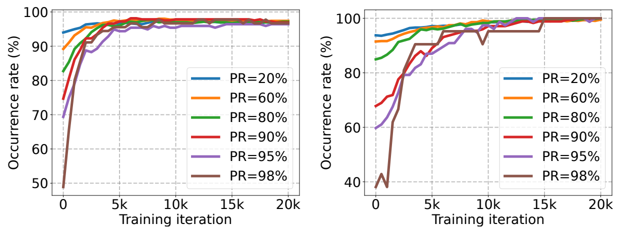


Figure 8: Ablation study of primary masks.



(a) Conv-4 (b) ResNet-12

Figure 9: Emergence of primary masks during meta training.

on the above observation, we attach importance to the primary masks within meta winning tickets.

4.3 Fast Meta Winning Ticket Finding Algorithm

The inter- and intra-layer mask patterns among meta winning tickets discussed above motivate us to identify a meta winning ticket early by exploiting the commonality among meta winning tickets.

Idea. EBTrain [36] shows that the masks of a winning ticket for vanilla supervised training tend to stabilize early in the training stage. Inspired by EBTrain, we demonstrate that the *primary masks* of a *meta winning ticket* emerge early during meta training.

We investigate the emergence of meta winning tickets by plotting the occurrence rate of primary masks during meta training. The occurrence rate is the percent of primary masks that occur in the current mask of the first layer. Specifically, Fig. 9 shows the occurrence rate after every 500 training iterations for the Conv-4 and ResNet-12 backbones, where the primary masks are extracted as in Sec. 4.2. We observe that most mask locations in the primary masks appear at the beginning. Over 90% mask locations of the primary masks are fixed within $3,000$ iterations for Conv-4, and $7,500$ training iterations for ResNet-12. For comparison, the complete meta training process of both Conv-4 and ResNet-12 contains 80,000 training iterations, which shows that the mask locations of the primary masks are nearly fixed when running only 4%-9% of total iterations.

Algorithm Sketch. Motivated by the observation above, we propose a simple yet effective method named **P**rimary **M**asked **T**icket (**PMT**), which identifies meta winning tickets once the mask locations in the first layer of the backbone become stable. Note that it is intractable to directly monitor the emergence of primary masks, since the emergence of primary masks is obtained via comparing different meta winning tickets, which are unknown beforehand. Instead, we monitor all the mask locations in the first layer of the backbone to recognize the primary masks when it stabilizes. The intuition is that, when the masks in the first layer are fixed, the primary masks must have emerged.

The detailed procedure is illustrated in Algorithm 2. We first initialize the backbone and a queue with length l. Then we repeat sampling a batch of tasks from $p(\mathcal{T})$, and training the backbone for γ iterations. After that, we obtain the mask of the first layer by magnitude-based pruning, and push the overlap of the last two masks of the first layer into Q. The training will be terminated

Algorithm 2: PMT: find a meta winning ticket early exploiting primary masks

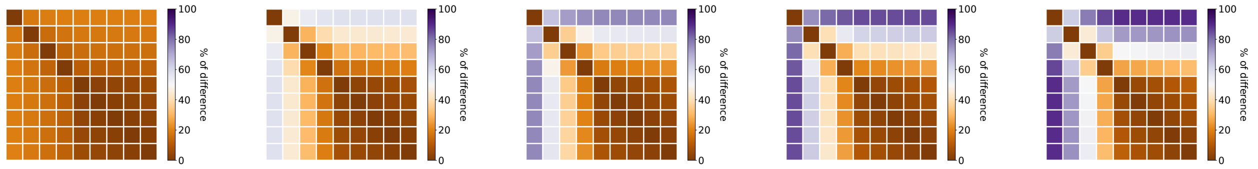
```
Input: $\mathcal{CR}$: compression ratio; $\theta_0$: initial weights; $p(\mathcal{T})$:
       distribution over tasks; $\mathcal{M}$: meta training algorithm;
       $\gamma$: training interval; $l$: length of queue; $\delta$: threshold
Output: A winning ticket $\theta_0 \cdot \boldsymbol{m}$
1  Initialize $\theta$ with $\theta_0$
2  Initialize a queue $Q$ with length $l$
3  Get keep ratio for the 1st layer as in [31]
4  while $MAX(Q) \leq \delta$ do
5      for $\gamma$ training iterations do
6          Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
7          Meta train $\theta$ on tasks $\mathcal{T}_i$
8      end
9      Prune the first layer in $\theta$ and obtain the mask $\boldsymbol{m}$
10     Compute the overlap $c$ between $\boldsymbol{m}$ and the last masks,
         and push it into $Q$
11 end
12 Return the ticket $\theta_0 \cdot \boldsymbol{m}$
```

when the maximum value within Q is larger than δ. Note that the pruning ratio of the first layer is obtained based on the smart ratios proposed by [31]. Specifically, for a L-layer backbone, the keep ratio (i.e., 1 minus the pruning ratio) of the l-th layer is proportional to $(L-l+1)^2 + (L-l+1)$.

Discussion. Although both the proposed PMT and EBTrain [36] identify a winning ticket at the early stage of training, they differ in two aspects. Firstly, EBTrain [36] relies on the observation that the mask locations of *all layers* stabilize in the early stage of training.

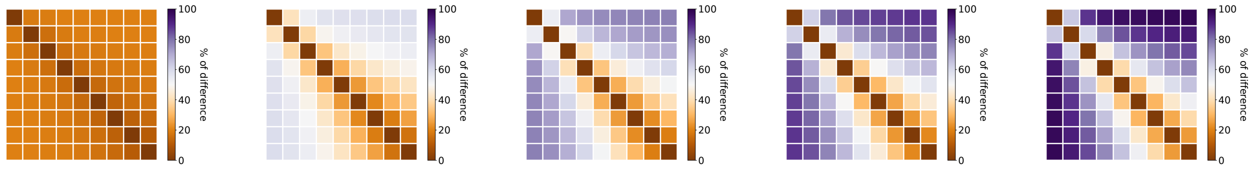
However, this observation does not hold in meta training. To show the differences between the proposed PMT and EBTrain, we conduct the same observation in meta learning scenario for EBTrain. Specifically, we record the meta training produce for both Conv-4 and ResNet-12. Then we calculate and plot the hamming distance among these checkpoints with different pruning ratios. The results on Conv-4 and ResNet-12 are shown in Fig. 10 and Fig. 11. For Conv-4, the mask matrix gets stable fast with low pruning ratio, but much slower with high pruning ratio, which is similar with observation in [36]. However, the mask matrix of larger backbone (ResNet-12) keeps changing during meta training, which implies the infeasibility of EBTrain to find meta winning ticket.

Secondly, our experiments demonstrate the importance of the masks in the *first layer* in a meta winning ticket, which motivates



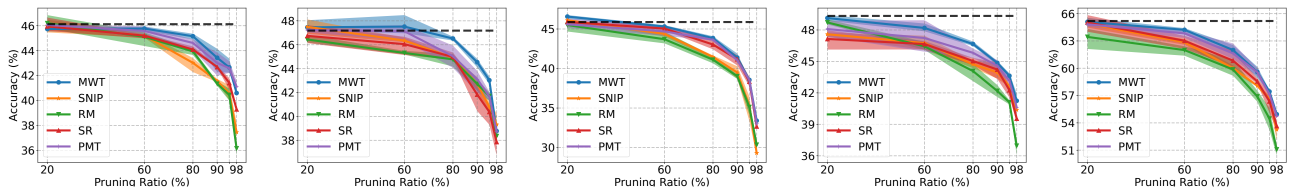
(a) $PR = 20\%$ (b) $PR = 60\%$ (c) $PR = 80\%$ (d) $PR = 90\%$ (e) $PR = 98\%$

Figure 10: The distance of mask matrix for Conv-4 in meta learning.



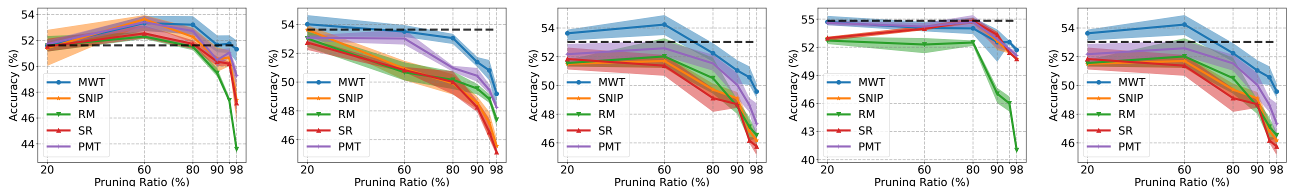
(a) $PR = 20\%$ (b) $PR = 60\%$ (c) $PR = 80\%$ (d) $PR = 90\%$ (e) $PR = 98\%$

Figure 11: The distance of mask matrix for ResNet-12 in meta learning.



(a) MAML/mini/1-shot (b) **ANIL**/mini/1-shot (c) **BOIL**/mini/1-shot (d) MAML/**tiered**/1-shot (e) MAML/mini/**5-shot**

Figure 12: Evaluation of tickets extract from Conv-4 backbone architecture via different methods.



(a) MAML/mini/1-shot (b) **ANIL**/mini/1-shot (c) **BOIL**/mini/1-shot (d) MAML/**tiered**/1-shot (e) MAML/mini/**5-shot**

Figure 13: Evaluation of tickets extract from ResNet-12 backbone architecture via different methods.

us to propose PMT for finding meta winning tickets at the early stage of meta-training.

5 EXPERIMENT

In this section, we experiment with various combinations of datasets (*mini*ImageNet and *tiered*ImageNet), backbones (Conv-4 and ResNet-12), meta training algorithms (MAML, ANIL and BOIL), and training manner (5-way-1-shot and 5-way-5-shot). The details of the experimental settings can be found in Sec. 3.1. All the neural networks are trained and tested on NVIDIA RTX 2080Ti with 12GB memory.

5.1 Baselines

We compare PMT with the following baselines:

- **Random Mask (RM)**. It randomly prunes weights across layers, and rewinds the remaining weights to the initialization. It serves as the baseline of a random ticket.
- **Smart Ratio (SR)**. It follows the per-layer pruning ratios of the meta winning ticket, yet randomly prunes weights within each layer. It is the baseline that only accounts for the inter-layer mask patterns of meta winning tickets.
- **SNIP**. It is a "pruning at initialization" scheme [18], which calculates the gradients of weights and obtains mask matrix by removing weights with gradient in small magnitude.
- **MWT**. It is the standard method to find a winning ticket [10] and used in Algorithm 1. It serves as the upper bound of a meta winning ticket in terms of accuracy.

5.2 Performance on Few-Shot Accuracy

We show the few-shot classification accuracy of the tickets obtained by various combinations of different meta-training methods (MAML, ANIL, BOIL) and different pruning algorithms (PMT, RM, SR, SNIP, MWT) in Fig. 12 and Fig. 13. For Conv-4, when pruning 20% weights, different pruning methods have similar performances.

Table 2: Computational cost for meta winning ticket finding and sparse meta training with Conv-4 backbone.

Setup	Ticket Finding FLOPS ($\times10^{15}$)				Sparse Meta Training FLOPS ($\times10^{15}$)					
	MWT	PMT	PMT	PMT	PR=60%		PR=80%		PR=90%	
		PR=60%	PR=80%	PR=90%	MWT	PMT	MWT	PMT	MWT	PMT
MAML/mini/1-shot	0.78	$0.05_{\times16.67}$	$0.06_{\times13.33}$	$0.06_{\times13.3}$	1.14	$0.41_{\times2.78}$	1.24	$0.52_{\times2.37}$	1.24	$0.52_{\times2.39}$
ANIL/mini/1-shot	1.94	$0.20_{\times9.52}$	$0.28_{\times7.02}$	$0.28_{\times7.02}$	2.93	$1.20_{\times2.46}$	3.23	$1.56_{\times2.07}$	3.23	$1.57_{\times2.06}$
BOIL/mini/1-shot	0.39	$0.04_{\times10.00}$	$0.05_{\times7.69}$	$0.05_{\times7.69}$	0.58	$0.23_{\times2.55}$	0.63	$0.29_{\times2.16}$	0.62	$0.28_{\times2.19}$
MAML/**tiered**/1-shot	0.78	$0.06_{\times13.33}$	$0.08_{\times9.41}$	$0.08_{\times9.41}$	1.13	$0.41_{\times2.76}$	1.23	$0.53_{\times2.30}$	1.20	$0.51_{\times2.36}$
MAML/mini/**5-shot**	3.89	$0.27_{\times14.29}$	$0.32_{\times12.31}$	$0.32_{\times12.31}$	5.71	$2.09_{\times2.73}$	6.22	$2.65_{\times2.35}$	6.20	$2.65_{\times2.34}$

Table 3: Computational cost for meta winning ticket finding and sparse meta training with ResNet-12 backbone.

Setup	Ticket Finding FLOPS ($\times10^{15}$)				Sparse Meta Training FLOPS ($\times10^{15}$)					
	MWT	PMT	PMT	PMT	PR=60%		PR=80%		PR=90%	
		PR=60%	PR=80%	PR=90%	MWT	PMT	MWT	PMT	MWT	PMT
MAML/mini/1-shot	20.08	$3.26_{\times6.15}$	$3.76_{\times5.33}$	$4.14_{\times4.85}$	30.62	$13.80_{\times2.22}$	33.73	$17.42_{\times1.94}$	34.13	$18.19_{\times1.88}$
ANIL/mini/1-shot	50.19	$13.80_{\times3.64}$	$14.43_{\times3.48}$	$15.18_{\times3.31}$	78.05	$41.66_{\times1.87}$	79.30	$43.54_{\times1.82}$	78.30	$43.29_{\times1.81}$
BOIL/mini/1-shot	10.04	$2.51_{\times4.00}$	$2.66_{\times3.78}$	$2.86_{\times3.51}$	15.46	$7.93_{\times1.95}$	16.06	$8.88_{\times1.81}$	16.46	$9.29_{\times1.77}$
MAML/**tiered**/1-shot	20.08	$3.51_{\times5.71}$	$3.89_{\times5.16}$	$4.64_{\times4.32}$	31.37	$14.81_{\times2.12}$	33.53	$14.34_{\times1.93}$	33.13	$17.69_{\times1.87}$
MAML/mini/**5-shot**	100.38	$21.33_{\times4.71}$	$22.59_{\times4.44}$	$26.98_{\times3.72}$	154.21	$75.16_{\times2.05}$	165.13	$88.34_{\times1.87}$	162.12	$88.71_{\times1.82}$

However, when the pruning ratio is increased from 60% to 98%, PMT outperforms RM, SR, and SNIP in few-shot classification accuracy. Compared with MWT (the upper bound), PMT incurs at most 1% drop in accuracy at the same pruning ratios. For ResNet-12, PMT also outperforms the baselines by a noticeable margin. The accuracy drop of PMT compared to MWT is within 2%, while other baselines incur an accuracy drop of at most 12%. The experimental results demonstrate the effectiveness of the proposed PMT. To balance few-shot accuracy and computational cost, PMT identifies meta winning tickets at the early stage of meta training based on primary masks. These experimental results show that the meta winning tickets found by PMT achieve competitive few-shot accuracy compared to MWT.

5.3 Performance on Sparse Meta-Training

In this section, we show the computation overhead for meta winning ticket finding and end-to-end meta-training of a sparse backbone. Note that we only report the results of PMT and MWT, since we empirically find that other methods, *i.e.*, RM, SR and SNIP, induce notable few-shot accuracy drop and fail to identify a competitive meta winning ticket.

To be more specific, we first identify a meta winning ticket and meta-train it till the loss on the validation dataset is minimized. Then we measure the FLOPS (including the forward and back propagation) [20] of both meta ticket finding and the meta training process of the sparse network. Note that sparse matrix multiplication can be supported by different libraries [1, 23].

The experimental results are shown in Table 2 and Table 3. From the results we can observe that the computational cost for finding meta winning tickets increases marginally with larger pruning ratios, since more weights are removed with large pruning ratios. For MAML, PMT accelerates ticket finding by at least 12.31× (MAML/mini/5-shot) on Conv-4 and 3.72× (MAML/mini/5-shot) on ResNet-12. For ANIL, PMT achieves at least 7.02× acceleration on Conv-4 and 3.31× on ResNet-12. For BOIL, PMT achieves at least 7.69× acceleration on Conv-4 and 3.51× on ResNet-12. The acceleration with ANIL is the lowest because ANIL only updates the weights of the output layer, making it converge slower than MAML and BOIL.

Compared with meta ticket finding, the acceleration of PMT on end-to-end sparse meta training decreases, because our method does not alter the meta-training process after meta ticket finding. However, PMT still achieves 2.06 − 2.39× acceleration on Conv-4 and 1.77 − 1.94× acceleration on ResNet-12 than using the standard MWT for ticket finding. The experimental results demonstrate the advantages of PMT in terms of computational cost.

6 RELATED WORK

We present the related works in following categories.

Meta Learning. Meta learning is a promising solution to few-shot learning [17, 34]. Meta learning methods include optimization-based [2, 9, 35], black-box/model-based [21, 29], and metric-based [30, 32] methods, and we focus on optimization-based schemes that learns an initialization by gradients [9, 22, 24, 38]. Gradient-based optimization such as MAML [9] is advantageous, since it can be applied to not only classification, but also regression and reinforcement learning. It can be fit for on-device adaptation given the advances in memory-efficient gradient descent implementations [13]. MAML and its variants [9, 22, 24, 38] only meta-trains the weights of the given backbone without optimizing its architecture. Recent study [33] proposes to perform magnitude-based pruning on the backbone to mitigate meta-overfitting. However, the authors follow the computation-intensive procedure, which consists of pre-training, pruning, and re-training. Different from previous studies, we optimize the backbone architecture in the lens of the lottery ticket hypothesis and propose a novel method to identify meta winning tickets early.

Lottery Ticket Hypothesis. The original LTH is studied in vanilla supervised learning with applications on network pruning [10]. LTH has been extended to sparsify the backbones in more sophisticated learning tasks including natural language processing [37], object detection [12], self-supervised pre-training [4], lifelong learning [5], etc. Our study is aligned with this emerging trend, yet focuses on meta learning, which is promising for fast model adaptation on low-resource platforms. Some pioneer studies apply LTH for efficient training. For example, EBTrain [36] uncovers winning tickets that appear early in vanilla supervised learning, whereas EarlyBERT [6] explores early winning ticket finding in pre-training language models. However, these methods cannot be applied in meta learning, since the observation on early-bird tickets for vanilla supervised learning does not hold in meta learning. We empirical confirm the existence of winning tickets in meta learning, and further propose to identify them early to balance few-shot accuracy and computational cost.

7 CONCLUSION

In this paper, we study the existence and fast finding of winning tickets for gradient-based meta learning. We empirically show that there exist meta winning tickets, which can be meta-trained in isolation to achieve comparable few-shot classification accuracy to the original backbone. Further, we propose Primary Mask Ticket (PMT) for fast ticket detection via exploiting the commonality among meta winning tickets, which balances few-shot accuracy and computational cost for finding meta winning tickets. Experiments show that the meta wining tickets found by PMT can achieve better performance than baselines. In addition, compared with the standard winning ticket detection methods, PMT significantly reduces the computational overhead in both meta ticket detection and end-to-end sparse meta-training.

8 ACKNOWLEDGMENTS

We thank the reviewers for their valuable comments and suggestions. Zimu Zhou's research was supported by the Lee Kong Chian Fellowship awarded by Singapore Management University.

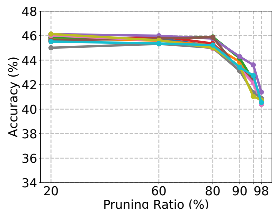
REFERENCES

[1] Martín Abadi, Paul Barham, Jianmin Chen, and et al. 2016. TensorFlow: A System for Large-Scale Machine Learning. In *OSDI*.
[2] Antreas Antoniou, Harrison Edwards, and Amos J. Storkey. 2019. How to train your MAML. In *ICLR*.
[3] Colby Banbury, Chuteng Zhou, Igor Fedorov, and et al. 2021. Micronets: Neural network architectures for deploying tinyml applications on commodity microcontrollers. *Proceedings of Machine Learning and Systems* 3 (2021).
[4] Tianlong Chen, Jonathan Frankle, Shiyu Chang, and et al. 2021. The lottery tickets hypothesis for supervised and self-supervised pre-training in computer vision models. In *CVPR*.
[5] Tianlong Chen, Zhenyu Zhang, Sijia Liu, and et al. 2021. Long live the lottery: The existence of winning tickets in lifelong learning. In *ICLR*.
[6] Xiaohan Chen, Yu Cheng, Shuohang Wang, and et al. 2021. Earlybert: Efficient bert training via early-bird lottery tickets. In *ACL*.
[7] James Diffenderfer and Bhavya Kailkhura. 2021. Multi-prize lottery ticket hypothesis: Finding accurate binary neural networks by pruning a randomly weighted network. In *ICLR*.
[8] Thomas Elsken, Benedikt Staffler, Jan Hendrik Metzen, and et al. 2020. Meta-learning of neural architectures for few-shot learning. In *CVPR*.
[9] Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*.
[10] Jonathan Frankle and Michael Carbin. 2019. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *ICLR*.
[11] Jonathan Frankle, Gintare Karolina Dziugaite, Daniel M. Roy, and et al. 2020. Linear mode connectivity and the lottery ticket hypothesis. In *ICML*.
[12] Sharath Girish, Shishira R. Maiya, Kamal Gupta, and et al. 2021. The lottery ticket hypothesis for object recognition. In *CVPR*.
[13] Mary Gooneratne, Khe Chai Sim, Petr Zadrazil, and et al. 2020. Low-rank gradient approximation for memory-efficient on-device training of deep neural network. In *ICASSP*.
[14] Song Han, Huizi Mao, and William J. Dally. 2016. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding. In *ICLR*.
[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and et al. 2015. Delving deep into rectifiers: surpassing human-Level performance on imageNet classification. In *ICCV*.
[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and et al. 2016. Deep residual learning for image recognition. In *CVPR*.
[17] Timothy M. Hospedales, Antreas Antoniou, Paul Micaelli, and et al. 2020. Meta-learning in neural networks: A survey. arXiv:2004.05439
[18] Namhoon Lee, Thalaiyasingam Ajanthan, and Philip H. S. Torr. 2019. Snip: Single-shot network pruning based on connection sensitivity. In *ICLR*.
[19] Xuezhe Ma and Eduard H. Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *ACL*.
[20] Pavlo Molchanov, Stephen Tyree, Tero Karras, and et al. 2017. Pruning Convolutional Neural Networks for Resource Efficient Inference. In *ICLR*.
[21] Tsendsuren Munkhdalai and Hong Yu. 2017. Meta networks. In *ICML*.
[22] Jaehoon Oh, Hyungjun Yoo, ChangHwan Kim, and et al. 2021. Boil: Towards representation change for few-shot learning. In *ICLR*.
[23] Adam Paszke, Sam Gross, Francisco Massa, and et al. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *NeurIPS*.
[24] Aniruddh Raghu, Maithra Raghu, Samy Bengio, and et al. 2020. Rapid learning or feature reuse? Towards understanding the effectiveness of MAML. In *ICLR*.
[25] Vivek Ramanujan, Mitchell Wortsman, Aniruddha Kembhavi, and et al. 2020. What's hidden in a randomly weighted neural network?. In *CVPR*.
[26] Sachin Ravi and Hugo Larochelle. 2017. Optimization as a model for few-Shot learning. In *ICLR*.
[27] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and et al. 2016. You only look once: Unified, real-time object detection. In *CVPR*.
[28] Mengye Ren, Eleni Triantafillou, Sachin Ravi, and et al. 2018. Meta-learning for semi-supervised few-shot classification. In *ICLR*.
[29] Adam Santoro, Sergey Bartunov, Matthew Botvinick, and et al. 2016. Meta-learning with memory-augmented neural networks. In *ICML*.
[30] Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *NeurIPS*.
[31] Jingtong Su, Yihang Chen, Tianle Cai, and et al. 2020. Sanity-checking pruning methods: Random tickets can win the jackpot. In *NeurIPS*.
[32] Flood Sung, Yongxin Yang, Li Zhang, and et al. 2018. Learning to compare: Relation network for few-shot learning. In *CVPR*.
[33] Hongduan Tian, Bo Liu, Xiao-Tong Yuan, and et al. 2020. Meta-learning with network pruning. In *ECCV*.
[34] Yaqing Wang, Quanming Yao, James T. Kwok, and et al. 2020. Generalizing from a few examples: A survey on few-shot learning. *Comput. Surveys* 53, 3 (2020), 63:1–63:34.
[35] Huaxiu Yao, Xian Wu, Zhiqiang Tao, and et al. 2020. Automated relational meta-learning. In *ICLR*.
[36] Haoran You, Chaojian Li, Pengfei Xu, and et al. 2020. Drawing early-bird tickets: Towards more efficient training of deep networks. In *ICLR*.
[37] Haonan Yu, Sergey Edunov, Yuandong Tian, and et al. 2020. Playing the lottery with rewards and multiple languages: lottery tickets in RL and NLP. In *ICLR*.
[38] Luisa M. Zintgraf, Kyriacos Shiarlis, Vitaly Kurin, and et al. 2019. Fast context adaptation via meta-learning. In *ICML*.

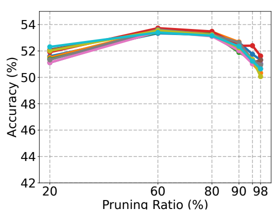
A ADDITIONAL RESULTS FOR FAST FINDING META WINNING TICKETS

A.1 Details of 10 Meta Winning Tickets

As shown in Fig. 14(a) and Fig. 14(b), we plot 10 meta winning tickets for the Conv-4 and ResNet-12 backbone, respectively, under the settings of MAML, *mini*ImageNet, and 5-way-1-shot. We can observe that the behaviors of these meta winning tickets are similarly as the those shown in Fig. 3(a) and Fig. 4(a).



(a) Conv-4

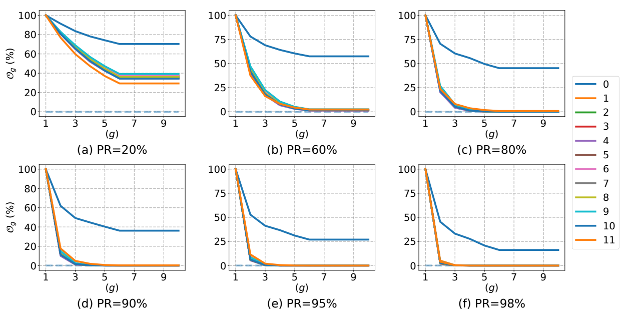


(b) ResNet-12

Figure 14: Performance of 10 different meta winning tickets.

A.2 Intra-Layer Mask Patterns for ResNet-12

We plot the values of O_g at each layer at different pruning ratios for the 10 meta winning tickets using the ResNet-12 backbone in Fig. 15. The results are similar to those using the Conv-4 backbone (shown in Fig. 7). There is a clear distinction between the overlap of mask locations at the first layer and the remaining layers.



(a) PR=20% (b) PR=60% (c) PR=80% (d) PR=90% (e) PR=95% (f) PR=98%

Figure 15: Overlap of per-layer mask locations among different meta winning tickets for ResNet-12 backbone.