FIND A WINNING SIGN: SIGN IS ALL WE NEED TO WIN THE LOTTERY

Anonymous authors
Paper under double-blind review

ABSTRACT

The lottery ticket hypothesis (LTH) posits the existence of a sparse network (a.k.a. winning ticket) that can generalize comparably to its dense counterpart after training from initialization. However, early works fail to generalize its observation and method to large-scale settings. While recent methods, such as weight rewinding or learning rate rewinding (LRR), may have found effective pruning methods, we note that they still struggle with identifying a winning ticket. In this paper, we take a step closer to finding a winning ticket by arguing that a signed mask, a binary mask with parameter sign information, can transfer the capability to achieve strong generalization after training (i.e., generalization potential) to a randomly initialized network. We first share our observation on the subnetwork trained by LRR: if the parameter signs are maintained, the LRR-driven subnetwork retains its generalization potential even when the parameter magnitudes are randomly initialized, excluding those of normalization layers. However, this fails when the magnitudes of normalization layer parameters are initialized together. To tackle the significant influence of normalization layer parameters, we propose AWS, a slight variation of LRR to find **A** **W**inning **S**ign. Specifically, we encourage low error barriers along the linear path connecting the subnetwork trained by AWS to its counterpart with initialized normalization layer parameters, maintaining the generalization potential even when all parameters are initialized. Interestingly, we observe that across various architectures and datasets, a signed mask of the AWS-driven subnetwork can allow a randomly initialized network to perform comparably to a dense network, taking a step closer to the goal of LTH.

1 INTRODUCTION

In the field of deep learning, over-parameterization has been viewed as a key to enhancing network capacity and thus achieving better generalization (Neyshabur et al., 2019; Belkin et al., 2019). It is well known that after training an over-parameterized dense network, many redundant parameters arise that can be removed without affecting performance, using network pruning techniques (Liu et al., 2017; Lin et al., 2020; Li et al., 2020). However, pruning an initialized dense network before training often leads to a sparse network that is difficult to optimize and fails to match the original generalization (Li et al., 2017; Evci et al., 2022). Is there a sparse subnetwork (i.e., a winning lottery ticket) that can achieve generalization comparable to its dense counterpart when trained from initialization? This challenging research question, posed by Frankle & Carbin (2019) as the lottery ticket hypothesis (LTH), has garnered significant attention and inspired many follow-up studies.

Iterative magnitude pruning (Frankle & Carbin, 2019) (IMP) is the representative method to identify a winning ticket through iterating three phases: training, pruning, and rewinding. Many researchers have sought to understand how IMP finds a winning ticket. Among several insightful findings, perspectives from the loss landscape have provided valuable insights. Frankle et al. (2020a); Evci et al. (2022); Paul et al. (2023) found that IMP can find a winning ticket only when the network obtained after the training phase maintains its basin of attraction after the pruning and rewinding phases, thereby preserving strong generalization potential (i.e., generalization capability after training) in subsequent iterations. Since this condition is difficult to satisfy in relatively large-scale settings when rewinding parameters to initialization, variants of IMP bypass the challenges by either rewinding to warm-up trained parameters (*weight rewinding*) (Frankle et al., 2020a) or skipping the rewinding

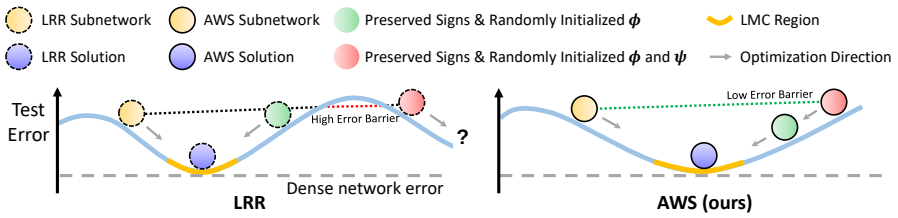


Figure 1: **Illustration of our motivation and method.** ψ and ϕ denote network parameters of normalization layers and parameters excluding those of normalization layers, respectively. The 'LMC region' refers to a region of solutions that are linearly mode-connected to the LRR or AWS solution.

phase (*learning rate rewinding*) (Renda et al., 2020). Although they found a subnetwork that performs comparably to a dense network after training, it is not at initialization, and thus not a winning ticket.

In this paper, we empirically show that an effective signed mask, a sparse mask with parameter sign information, is key to satisfying the challenging condition for finding a winning ticket. Specifically, we leverage learning rate rewinding (LRR) for its ability to find effective parameter sign and sparsity configuration (Gadhikar & Burkholz, 2024). Then, with a slight modification to LRR, we demonstrate that if parameter signs are preserved, the subnetwork obtained through our LRR variant remains within its basin of attraction even after randomly initializing its parameter magnitudes. This implies that the generalization potential of the subnetwork can be transferred to a randomly initialized network via the signed mask, possibly allowing it to generalize comparably to the dense network after training.

We observe that the original LRR fails to achieve this. As illustrated on the left side of Figure 1, the LRR subnetwork leaves its basin after randomly initializing parameter magnitudes while maintaining their signs, as indicated by the red ball and the high error barrier between the yellow and red ball, similar to the finding by Frankle et al. (2020b). However, we argue that this failure is attributed to the significant influence of normalization layer parameters when initializing their magnitudes. Interestingly, we observe that when we exclude the normalization layer parameters from initialization, maintaining the parameter signs allows the LRR subnetwork to remain in the same basin even after randomly initializing the other parameter magnitudes. This results in convergence to a solution with low error barrier along the linear path connecting it to the LRR solution (i.e., linearly mode-connected to the LRR solution), as indicated by the green ball moving towards the blue ball on the orange line region. These results indicate that when the signed mask and normalization layer parameters of the LRR subnetwork are transferred to a randomly initialized network, the resulting network can inherit the generalization potential of the LRR subnetwork.

To take a step closer to finding a winning ticket, we eliminate the need for trained normalization parameters by tackling the adverse influence of initializing normalization layer parameters. To this end, we propose AWS, a slight variation of LRR to find **A** **W**inning **S**ign, that prevents high error barriers along the linear path connecting the subnetwork trained by AWS to its counterpart with initialized normalization parameters. Specifically, at every network forward pass, AWS randomly and linearly interpolates the current normalization layer parameters with the initialized ones and uses the interpolated ones for training. As illustrated in the right side of Figure 1, we argue that the AWS subnetwork remains within its basin even after randomly initializing all parameter magnitudes while preserving their signs, as indicated by the low error barrier between the yellow and red ball. This leads the resulting network to converge to a solution that is linearly mode-connected to the AWS solution, whose performance is likely comparable to that of the dense network, indicated by the gray dotted line. Experimental results across various architectures and datasets demonstrate that transferring the signed mask of the AWS subnetwork allows a randomly initialized network to generalize comparably to the AWS solution after training, bringing us closer to the goal of LTH.

We summarize the contributions of our work as follows:

- We observe that a randomly initialized network can inherit the generalization potential of the LRR subnetwork through its signed mask and the normalization layer parameters.
- We propose AWS that eliminates the need for trained normalization parameters by preventing high error barriers between the AWS subnetwork and its counterpart with initialized normalization parameters.
- In contrast to existing methods that are limited to finding a winning ticket with predetermined initialization, we show that the signed mask acquired from AWS allows a randomly initialized network to generalize comparably to a dense network after training.

2 Related Works

Lottery Ticket Hypothesis (LTH). Frankle & Carbin (2019) propose LTH which states that within a dense network, there exists a sparse subnetwork that, when trained from initialization, can achieve performance comparable to the dense counterpart. To find such a winning lottery ticket, the authors proposed *iterative magnitude pruning* (IMP) and demonstrated that IMP successfully finds a winning ticket in a relatively small-scale setting. Follow-up works have delved into a broad range of topics related to LTH, such as theoretical support for the existence of the winning ticket (Malach et al., 2020; Orseau et al., 2020; Burkholz, 2022; da Cunha et al., 2022), efficient alternatives to IMP (You et al., 2020), searching for a winning ticket without weight training (Chen et al., 2022; Sreenivasan et al., 2022; Koster et al., 2022), and empirical analyses on winning ticket (Zhou et al., 2019; Frankle et al., 2020a; Ma et al., 2021; Sakamoto & Sato, 2022; Evci et al., 2022; Paul et al., 2023).

Insights into a Winning Ticket. One of the most important topics is investigating what makes a sparse network win the lottery. Frankle et al. (2020a) introduced the notion of mode connectivity (Freeman & Bruna, 2017; Nguyen, 2019; Draxler et al., 2018; Garipov et al., 2018; Lubana et al., 2023) into LTH to investigate the conditions under which IMP finds a winning ticket. They consider a network stable to SGD noise if, under different SGD randomness, it converges to a region of solutions that exhibit low error barriers along the linear path connecting them. Based on this definition, they demonstrated that IMP succeeds only when the rewound network is stable to SGD noise. Evci et al. (2022) found that a winning ticket can be found only when it resides in the same basin as the pruning solution used to obtain the pruning mask. Paul et al. (2023) demonstrated that a pruning mask obtained in an IMP iteration guides the subsequent pruning solution to be linearly mode-connected to the previous IMP solution, leading the consecutive pruning solutions to be piece-wise linearly mode-connected. In summary, these findings suggest that IMP can identify a winning ticket only when the network obtained from the training phase remains within its basin of attraction after the pruning and rewinding phases, thereby preserving the generalization of the original dense network during all iterations. Variants of IMP, such as *weight rewinding* (Frankle et al., 2020a) and *learning rate rewinding* (Renda et al., 2020), bypass this challenging condition by finding an effective sparse network with trained parameters rather than initialized ones, failing to find a winning ticket.

Significance of Parameter Signs in LTH. Recently, several works reported the importance of parameter signs from the perspective of representation capacity Wang et al. (2023a;b). Zhou et al. (2019) are the first to discover the role of parameter signs in the context of LTH. They empirically showed that in the parameter rewinding stage of IMP, rewinding parameter signs has a greater impact on the performance than the magnitudes. By contrast, Frankle et al. (2020b) showed that transferring the signed mask obtained by IMP to the original initialization performs worse than transferring them along with the respective magnitudes. Gadhikar & Burkholz (2024) demonstrated that IMP can fail to find a winning ticket because it loses crucial sign information during parameter rewinding and struggles to learn an effective sign configuration again due to reduced network capacity. They claimed that learning rate rewinding (Renda et al., 2020) (LRR), on the other hand, identifies more performant sparse networks by finding and maintaining the effective sign configuration during training.

We observe that the ineffectiveness of transferring parameter signs to an initialized network, as noted in Frankle et al. (2020b), is due to the adverse effect of initializing the normalization layer parameters. To address this issue, we propose a slight variant of LRR and demonstrate that the challenging conditions for finding a winning ticket suggested by Frankle et al. (2020a); Evci et al. (2022); Paul et al. (2023) can be satisfied using the effective signed mask acquired by our LRR variant.

3 Method

3.1 Notations and Background

Lottery Ticket Hypothesis. Let $\boldsymbol{\theta} \in \mathbb{R}^d$ denote the parameters of a neural network. We use $\boldsymbol{\theta}$ to also represent the neural network parameterized by $\boldsymbol{\theta}$. Consider a binary mask $\boldsymbol{m} \in \{0,1\}^d$, having the same shape as $\boldsymbol{\theta}$. Note that the mask values for parameters not targeted for pruning, such as biases, are fixed to be 1. Then, the mask $\boldsymbol{m}$ defines a sparse subnetwork of the original dense network as $\boldsymbol{\theta} \odot \boldsymbol{m}$. The lottery ticket hypothesis (LTH) (Frankle & Carbin, 2019) posits the existence of a mask with non-trivial sparsity (i.e. $\sum_i m_i \ll d$) that allows $\boldsymbol{\theta}_{\text{init}} \odot \boldsymbol{m}$, where $\boldsymbol{\theta}_{\text{init}}$ represents initialized parameters, to perform comparably to the dense network after training. Such

a sparse network at initialization is referred to as a *winning ticket*. *Iterative magnitude pruning* (IMP) (Frankle & Carbin, 2019) was suggested as a way to find the winning ticket through iterative `training` $\rightarrow$ `pruning` $\rightarrow$ `rewinding` procedures. Let $\mathcal{A}(\boldsymbol{\theta}, u)$ represent an SGD learning algorithm with an initialized learning rate scheduler that updates $\boldsymbol{\theta}$ until convergence using SGD randomness $u \sim U$ (e.g. randomness from a data loader or data augmentations). We omit u from $\mathcal{A}(\boldsymbol{\theta}, u)$ if unnecessary. At `training` phase of the t-th iteration, IMP trains the masked initial parameters, obtaining $\boldsymbol{\theta}_t^{\text{IMP}} \odot \boldsymbol{m}_{t-1}^{\text{IMP}} = \mathcal{A}(\boldsymbol{\theta}_0^{\text{IMP}} \odot \boldsymbol{m}_{t-1}^{\text{IMP}})$, where $\boldsymbol{m}_{t-1}^{\text{IMP}}$ denotes the mask obtained from the $(t-1)$-th iteration. At `pruning` phase, IMP produces the t-th mask by removing a portion of the non-zero weights in $\boldsymbol{\theta}_t^{\text{IMP}} \odot \boldsymbol{m}_{t-1}^{\text{IMP}}$ (commonly 20%) with the smallest magnitudes: $\boldsymbol{m}_t^{\text{IMP}} = \texttt{prune}(\boldsymbol{\theta}_t^{\text{IMP}} \odot \boldsymbol{m}_{t-1}^{\text{IMP}})$. At `rewinding` phase, $\boldsymbol{\theta}_t^{\text{IMP}}$ is rewound to the initial parameters, $\boldsymbol{\theta}_0^{\text{IMP}}$, and the entire process is repeated until $t = T$. Finally, IMP produces $\boldsymbol{\theta}_0^{\text{IMP}} \odot \boldsymbol{m}_T^{\text{IMP}}$, referred to as the *IMP subnetwork*, and after training the IMP subnetwork, IMP obtains $\mathcal{A}(\boldsymbol{\theta}_0^{\text{IMP}} \odot \boldsymbol{m}_T^{\text{IMP}})$, referred to as the *IMP solution*.

Variants of IMP. Several variants of IMP have been proposed to address the failure of IMP in generalizing to more challenging settings, especially focusing on the `rewinding` phase. Based on the analysis of stability to SGD noise, Frankle et al. (2020a) proposed *weight rewinding* (WR) that rewinds $\boldsymbol{\theta}_t^{\text{IMP}}$ to the warm-up trained parameters rather than to $\boldsymbol{\theta}_0^{\text{IMP}}$. *Learning rate rewinding* (LRR) Renda et al. (2020) is another variant of IMP that skips parameter rewinding and instead rewinds only the learning rate schedule. At the `training` phase of the t-th iteration, LRR trains $\boldsymbol{\theta}_{t-1}^{\text{LRR}} \odot \boldsymbol{m}_{t-1}^{\text{LRR}}$, obtaining $\boldsymbol{\theta}_t^{\text{LRR}} \odot \boldsymbol{m}_{t-1}^{\text{LRR}} = \mathcal{A}(\boldsymbol{\theta}_{t-1}^{\text{LRR}} \odot \boldsymbol{m}_{t-1}^{\text{LRR}})$, where $\boldsymbol{m}_{t-1}^{\text{LRR}}$ represents the mask obtained from the $(t-1)$-th iteration. After obtaining $\boldsymbol{m}_t^{\text{LRR}} = \texttt{prune}(\boldsymbol{\theta}_t^{\text{LRR}} \odot \boldsymbol{m}_{t-1}^{\text{LRR}})$ at the `pruning` phase, LRR skips the `rewinding` phase and continues the subsequent iterations until $t = T$. Finally, LRR produces $\boldsymbol{\theta}_T^{\text{LRR}} \odot \boldsymbol{m}_T^{\text{LRR}}$, referred to as the *LRR subnetwork*, and after training the LRR subnetwork, LRR obtains $\mathcal{A}(\boldsymbol{\theta}_T^{\text{LRR}} \odot \boldsymbol{m}_T^{\text{LRR}})$, referred to as the *LRR solution*.

Linear Mode Connectivity and Stability to SGD noise. Linear mode connectivity and stability to SGD noise have been adopted as useful tools for analyzing winning tickets (Frankle et al., 2020a; Evci et al., 2022; Paul et al., 2023). Let $\mathcal{E}(\boldsymbol{\theta})$ denote the test error of $\boldsymbol{\theta}$. We define the error barrier between two parameters, $\boldsymbol{\theta}$ and $\boldsymbol{\theta}'$, when interpolating them by a factor of α as the difference between the error of the interpolated network and the mean error:

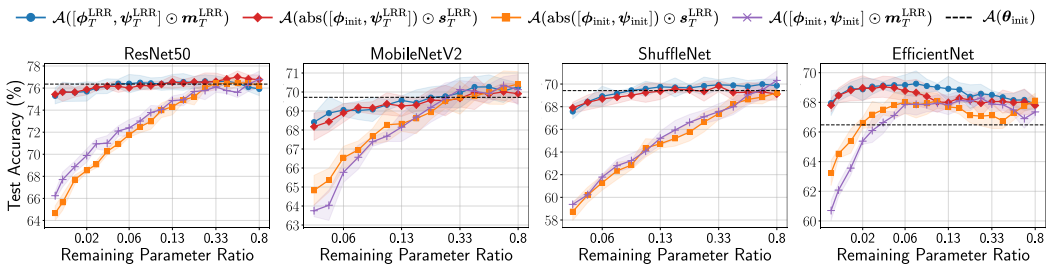
$$\mathcal{E}_\alpha(\boldsymbol{\theta}, \boldsymbol{\theta}') = \mathcal{E}(\alpha\boldsymbol{\theta} + (1-\alpha)\boldsymbol{\theta}') - (\mathcal{E}(\boldsymbol{\theta}) + \mathcal{E}(\boldsymbol{\theta}'))/2. \tag{1}$$

Then, we define $\mathcal{E}(\boldsymbol{\theta}, \boldsymbol{\theta}') = \sup_\alpha \mathcal{E}_\alpha(\boldsymbol{\theta}, \boldsymbol{\theta}')$. If $\mathcal{E}(\boldsymbol{\theta}, \boldsymbol{\theta}')$ is smaller than a sufficiently small value ϵ, we consider $\boldsymbol{\theta}$ and $\boldsymbol{\theta}'$ to be linearly mode-connected (LMC). ϵ is often determined empirically such as the standard deviation of errors of a dense network across different training seeds (Paul et al., 2023). Based on the definition of LMC, we define the stability to SGD noise as the condition where a pair of networks are LMC when trained from the same initial parameters but with different SGD randomness (Frankle et al., 2020a). Formally, $\boldsymbol{\theta}$ is considered stable to SGD noise if $\mathcal{A}(\boldsymbol{\theta}, u)$ and $\mathcal{A}(\boldsymbol{\theta}, u')$ are LMC with $u, u' \sim U$.

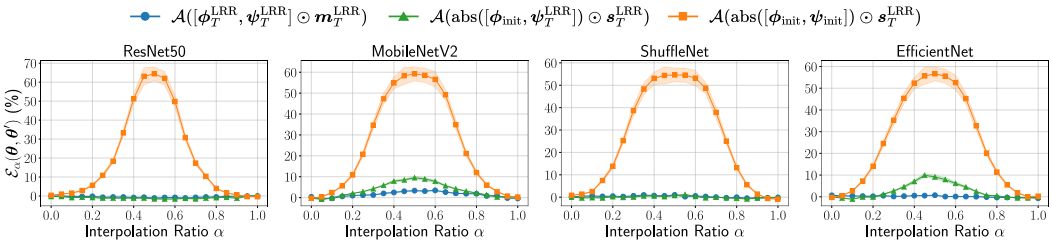
As demonstrated by previous works (Frankle et al., 2020a; Evci et al., 2022; Paul et al., 2023), IMP can identify a winning ticket only when at the t-th iteration, $\boldsymbol{\theta}_0^{\text{IMP}} \odot \boldsymbol{m}_{t-1}^{\text{IMP}}$ still resides in the basin of attraction of the $(t-1)$-th solution, $\boldsymbol{\theta}_{t-1}^{\text{IMP}} \odot \boldsymbol{m}_{t-2}^{\text{IMP}}$, even after updating $\boldsymbol{m}_{t-2}^{\text{IMP}}$ to $\boldsymbol{m}_{t-1}^{\text{IMP}}$ and rewinding $\boldsymbol{\theta}_{t-1}^{\text{IMP}}$ to $\boldsymbol{\theta}_0^{\text{IMP}}$. Paul et al. (2023) claim that if this condition is satisfied at every iteration, the consecutive solutions after the training phase are piece-wise LMC, allowing the final IMP solution to generalize comparably to the dense network. To examine this condition, previous works investigate whether a rewound network is stable to SGD noise and converges to a solution with linear mode connectivity to the network before the rewinding and pruning phases. WR and LRR satisfy the challenging condition by rewinding to warm-up trained parameters or skipping the `rewinding` phase. In contrast, we aim to transfer the strong generalization potential to a randomly initialized network through an effective signed mask, progressing toward the goal of LTH.

3.2 Motivation

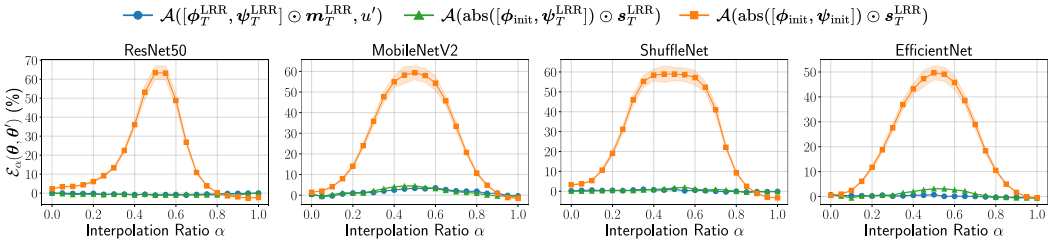
Gadhikar & Burkholz (2024) demonstrated that learning rate rewinding (LRR) successfully maintains the performance of dense networks by learning and maintaining an effective parameter sign configuration while pruning unimportant parameters, which iterative magnitude pruning (IMP) and weight rewinding (WR) fail. Several other works (Zhou et al., 2019; Frankle et al., 2020b; Chen et al., 2022; Sreenivasan et al., 2022; Koster et al., 2022; Wang et al., 2023b;a) also found the importance



(a) Test accuracy



(b) Stability to SGD-noise



(c) Linear mode connectivity with $\mathcal{A}([\phi_T^{\mathrm{LRR}}, \psi_T^{\mathrm{LRR}}] \odot m_T^{\mathrm{LRR}}, u)$

Figure 2: **Motivational experiments on CIFAR-100.** We investigate the effect of parameter initialization in the LRR subnetwork while preserving their signs with respect to **(a) test accuracy**, **(b) SGD-noise stability**, and **(c) linear mode connectivity with the LRR subnetwork.** In (b) and (c), we use a pruned network with a remaining parameter ratio of approximately 0.06. We show the mean (each point) and standard deviation (shaded area) across 3 trials.

of parameter signs in the context of the lottery ticket hypothesis or representation learning. In this work, motivated by these studies, we hypothesize that **the parameter sign information obtained through LRR can transfer the generalization potential of the LRR subnetwork to a randomly initialized network.** Let $\mathrm{sign}_0(\cdot)$ denote a function that outputs the sign of each input element if it is non-zero, and 0 otherwise. Then, $s_T^{\mathrm{LRR}} = \mathrm{sign}_0(\theta_T^{\mathrm{LRR}} \odot m_T^{\mathrm{LRR}})$ represents the signed mask of the LRR subnetwork. More specifically, our hypothesis states that applying s_T^{LRR} to a randomly initialized network, θ_{init}, will enable the resulting network to match the performance of the LRR solution after training: $\mathcal{A}(\mathrm{abs}(\theta_{\mathrm{init}}) \odot s_T^{\mathrm{LRR}}) \approx \mathcal{A}(\theta_T^{\mathrm{LRR}} \odot m_T^{\mathrm{LRR}})$ where $\mathrm{abs}(\cdot)$ and θ_{init} denote the modulus function and a randomly initialized network, respectively.

A similar idea was studied by Frankle et al. (2020b). They showed that replacing the signs of initialized parameters with those of the IMP subnetwork does not improve performance compared to using magnitude information in conjunction. We point out that this failure is attributed to ignoring the impact of parameters that may rely more on magnitudes than their signs. In the case of weight parameters, such as the weights in a convolutional layer, the sign configuration has a critical role in determining the functional mechanism of the layer as discussed in Wang et al. (2023b;a); Gadhikar & Burkholz (2024). By contrast, for parameters in a normalization layer, such as the batch normalization (Ioffe & Szegedy, 2015) or layer normalization (Ba et al., 2016), the magnitude may be much more important than the sign since the weight parameter, initialized to 1, is nearly always positive after training, and

Algorithm 1 AWS: a slight modification to LRR to find **a** **w**inning **s**ign. The modification is highlighted in red.

Require: Initialize ϕ_0, ψ_0, and $m_0 \leftarrow (1, \ldots, 1) \in \mathbb{R}^d$
for $t = 1$ to T **do**
  while not converge **do**
    $(\psi_{t-1}, \psi_{\text{init}})_\alpha = \alpha \cdot \psi_{t-1} + (1-\alpha) \cdot \psi_{\text{init}}$ where $\alpha \sim U(0,1)$ ▷ Interpolating ψ_{t-1} and ψ_{init}
    Forward pass using $[\phi_{t-1}, (\psi_{t-1}, \psi_{\text{init}})_\alpha] \odot m_{t-1}$ ▷ Forward pass with the interpolated parameters
    Update ϕ_{t-1} and ψ_{t-1} via gradient descent
  end while
  $\phi_t \leftarrow \phi_{t-1}$ and $\psi_t \leftarrow \psi_{t-1}$
  $m_t \leftarrow \texttt{prune}([\phi_t, \psi_t] \odot m_{t-1})$ ▷ Update the sparse mask
  Rewind learning rate scheduler
end for
return $\text{sign}_0([\phi_T, \psi_T] \odot m_T)$ ▷ Obtain the signed mask

the bias parameter loses the replaced signs since it is initialized to 0. We also observe that transferring the signs of parameters of normalization layers is not beneficial as analyzed in Appendix A.

Let ϕ and ψ denote the parameters excluding those of normalization layers and those of normalization layers, respectively. Then, we represent a randomly initialized network and the LRR subnetwork as $\theta_{\text{init}} = [\phi_{\text{init}}, \psi_{\text{init}}]$ and $\theta_T^{\text{LRR}} = [\phi_T^{\text{LRR}}, \psi_T^{\text{LRR}}]$, respectively. To test our conjecture, we compare two cases: in the LRR subnetwork, randomly initializing the magnitudes of both ϕ_T^{LRR} and ψ_T^{LRR} versus only ϕ_T^{LRR} while maintaining the signed mask (i.e. $\mathcal{A}(\text{abs}([\phi_{\text{init}}, \psi_{\text{init}}]) \odot s_T^{\text{LRR}})$ vs. $\mathcal{A}(\text{abs}([\phi_{\text{init}}, \psi_T^{\text{LRR}}]) \odot s_T^{\text{LRR}})$). Figure 2 shows the results on CIFAR-100 with various architectures. For details on the experiments, please refer to Section 4.1. Figure 2a shows that similar to the results in Frankle et al. (2020b), preserving the signed mask while initializing the magnitudes of all parameters randomly (indicated by the orange plots) results in performance similar to the case where sign information is not used (indicated by the purple plots), lagging far behind the performance of the LRR solution (indicated by the blue plots). This result indicates that sign information is not beneficial when the magnitudes of all parameters are randomly initialized. On the other hand, interestingly, when the parameters of normalization layers are kept intact (indicated by the red plots), the performance is comparable to the LRR solution after training, indicating that using the sign information of the LRR subnetwork is beneficial when excluding the influence of initializing the normalization layer parameters. To examine whether the resulting networks reside in the basin of attraction of the LRR subnetwork, we analyze their stability to SGD noise and linear mode connectivity with the LRR subnetwork. Figure 2b demonstrates that while randomly initializing the magnitudes of all parameters significantly ruins the stability of the LRR subnetwork (indicated by the orange plots), it is effectively preserved when ignoring the adverse influence of initializing the normalization layer parameters (indicated by the green plots). As shown in Figure 2c, we also observe that the preserved stability, in turn, leads the resulting network, $\text{abs}([\phi_{\text{init}}, \psi_T^{\text{LRR}}]) \odot s_T^{\text{LRR}}$, to converge to a solution with a low error barrier along the linear path connecting it to the LRR solution (indicated by the green plots), possibly suggesting linear mode connectivity between the two. This contrasts with the high error barrier observed when initializing all parameter magnitudes (indicated by the orange plots).

The left side of Figure 1 summarizes the observations from our motivational experiments:

- In the LRR subnetwork, randomly initializing the magnitude of all parameters while preserving their signs causes the resulting network to lose SGD noise stability of the LRR subnetwork, potentially leading to a suboptimal solution, as indicated by the red ball.
- On the other hand, when the normalization layer parameters are kept intact, the resulting network, $\text{abs}([\phi_{\text{init}}, \psi_T^{\text{LRR}}]) \odot s_T^{\text{LRR}}$, exhibits significant stability to SGD noise and converges to a solution with a low error barrier along the linear path connecting it to the LRR solution, resulting in $\mathcal{A}(\text{abs}([\phi_{\text{init}}, \psi_T^{\text{LRR}}]) \odot s_T^{\text{LRR}}) \approx \mathcal{A}([\phi_T^{\text{LRR}}, \psi_T^{\text{LRR}}] \odot m_T^{\text{LRR}})$, as indicated by the green ball.

Our observations demonstrate that when the parameter signs are preserved, the LRR subnetwork stays within its basin of attraction even after randomly initializing the other parameters, excluding those of normalization layers. In other words, a randomly initialized network can inherit the generalization potential of the LRR subnetwork through the signed mask and normalization layer parameters of the LRR subnetwork.

3.3 AWS: FINDING A WINNING SIGN

Our observations provide valuable insights into the role of the signed mask in transferring strong generalization potential to a randomly initialized network. However, the need for the trained normalization parameters still limits LRR in achieving the goal of LTH. In this subsection, we introduce a method that addresses the adverse impact of initializing the normalization layer parameters to further progress toward the goal of LTH. Our goal is to maintain the basin of attraction in which the LRR subnetwork resides when the normalization layer parameters are initialized. Two networks residing in the same basin may indicate that no high error barrier exists along the linear path connecting them (Evci et al., 2022). Thus, we propose a simple variation of LRR to find **a** **w**inning **s**ign, referred to as **AWS**, that prevents any high error barriers along the linear path connecting the LRR subnetwork and its counterpart with initialized normalization parameters. Specifically, AWS randomly and linearly interpolates the parameters of normalization layers with their initialization and uses the interpolated parameters instead of the original parameters. Formally, at every network forward pass during the t-th iteration, AWS obtains

$$(\boldsymbol{\psi}_t^{\text{AWS}}, \boldsymbol{\psi}_{\text{init}})_\alpha = \alpha \cdot \boldsymbol{\psi}_t^{\text{AWS}} + (1-\alpha) \cdot \boldsymbol{\psi}_{\text{init}}, \tag{2}$$

where $\boldsymbol{\psi}_t^{\text{AWS}}$ denotes the parameters of normalization layers during the t-th iteration of AWS and $\alpha \sim U(0,1)$. Then, AWS uses $(\boldsymbol{\psi}_t^{\text{AWS}}, \boldsymbol{\psi}_{\text{init}})_\alpha$ instead of $\boldsymbol{\psi}_t^{\text{AWS}}$ for network forwarding. We present the pseudo-code of AWS in Algorithm 1, omitting the superscript 'AWS' for simplicity. After all iterations, we transfer the resulting signed mask, $\boldsymbol{s}_T^{\text{AWS}}$, to a random initialization, obtaining $\text{abs}([\boldsymbol{\phi}_{\text{init}}, \boldsymbol{\psi}_{\text{init}}]) \odot \boldsymbol{s}_T^{\text{AWS}}$, and train it using normal training set-up until convergence.

The right side of Figure 1 shows the conceptual illustration of AWS. In contrast to LRR, AWS can allow a randomly initialized network (indicated by the red ball) to lie in the basin of attraction of the AWS subnetwork through the learned signed mask, $\boldsymbol{s}_T^{\text{AWS}}$, possibly leading it to converge to a solution that is linearly mode-connected to the AWS solution (indicated by the blue ball). Thus, we argue that **$\boldsymbol{s}_T^{\text{AWS}}$ can transfer the generalization potential of the AWS subnetwork to a randomly initialized network, possibly resulting in performance comparable to a dense network after training.**

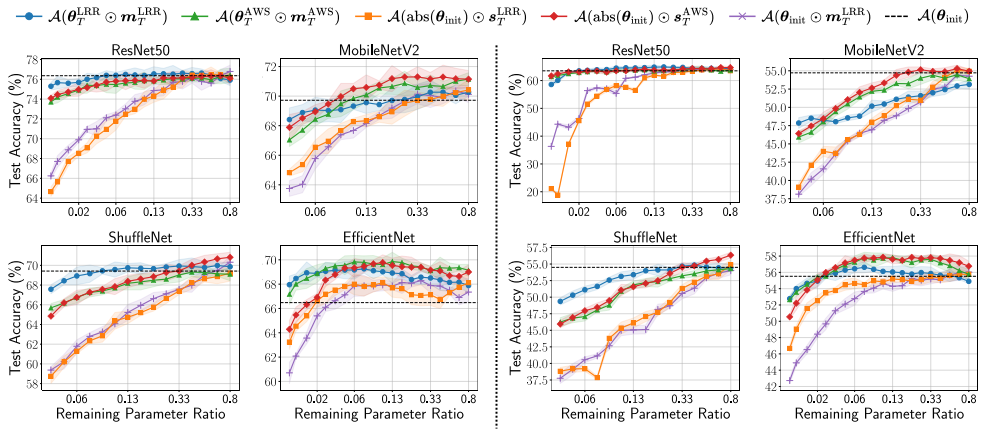
4 EXPERIMENTS

4.1 IMPLEMENTATION DETAILS

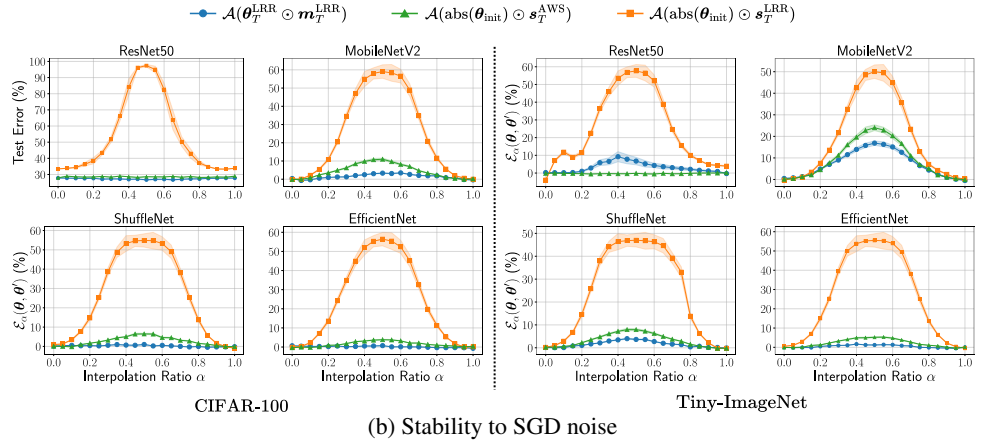
Datasets and models. Following the previous works (Ma et al., 2021; Gadhikar & Burkholz, 2024), we conduct experiments on CIFAR-100 (Krizhevsky & Hinton, 2009), Tiny-ImageNet (Le & Yang, 2015), and ImageNet (Russakovsky et al., 2015). For both CIFAR-100 and Tiny-ImageNet, we adopt ResNet-50 (He et al., 2016), MobileNetV2 (Sandler et al., 2018), ShuffleNet (Zhang et al., 2018), and EfficientNet (Tan & Le, 2019) to validate our method across various architectures. For ImageNet experiments, we use ResNet50, MobileNetV2, and MLP-Mixer (Tolstikhin et al., 2021). We adopt MLP-Mixer, which includes layer normalization, to demonstrate the generalization of our method to different types of normalization layers rather than the batch normalization layer.

Implementation. We implement all experiments using PyTorch (Paszke et al., 2019). In both learning rate rewinding (LRR) and the proposed AWS method, we observe that many training epochs and learning rate scheduling are unnecessary during the `training` phase, as the network converges quickly due to the absence of parameter rewinding, and learning rate scheduling has little impact on the performance. Thus, during the `training` phase for both LRR and AWS, we train a network for 10 epochs for CIFAR-100 and Tiny-ImageNet experiments, and 5 epochs for ImageNet experiments without learning rate scheduling. To ensure a network can converge at the early iterations, we conduct warm-up training before the first iteration. Warm-up training epochs are set to 10 for both the CIFAR-100 and Tiny-ImageNet experiments and 5 for the ImageNet experiments. After the T-th iteration, we conduct a final training for 100 epochs for the CIFAR-100 and Tiny-ImageNet experiments and 300 epochs for the ImageNet experiments.

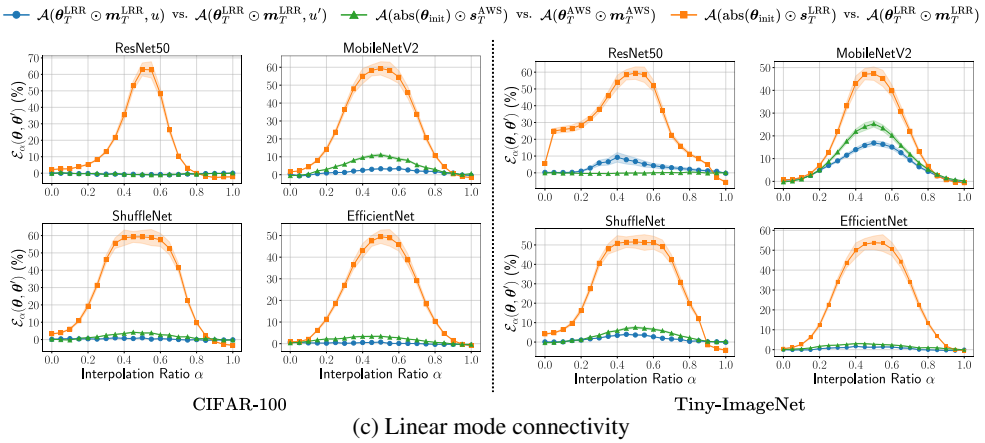
Optimization. For the CIFAR-100 and Tiny-ImageNet experiments, we use the SGD optimizer with a momentum of 0.9, a weight decay of 0.0005, and an initial learning rate of 0.1, which is decayed by a factor of 0.1 at the 50th and 75th epochs during 100 epochs. For ImageNet experiments, we use Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$, a weight decay of 0.0001 and an initial learning rate of 0.001, which is decayed from 0.001 to 0.00001 using cosine annealing. We use a batch size of 128 for the CIFAR-100 and Tiny-ImageNet experiments and 512 for the ImageNet experiments.



(a) Test accuracy



(b) Stability to SGD noise



(c) Linear mode connectivity

Figure 3: **Main results on CIFAR-100 and Tiny-ImageNet. (a):** Test accuracy of the LRR solution (blue), the AWS solution (green), a randomly initialized network trained with the LRR-driven signed mask (orange), and a randomly initialized network trained with the AWS-driven signed mask (red). **(b) and (c):** Analysis of stability to SGD noise and linear mode connectivity, respectively. A randomly initialized network trained with the AWS-driven signed mask exhibits relatively high stability to SGD noise and significant linear mode connectivity with the AWS solution (green), contrasting to the case of LRR (orange). In (b) and (c), we use a pruned network with a remaining parameter ratio of approximately 0.07 for CIFAR-100 and 0.1 for Tiny-ImageNet experiments. We show the mean (each point) and standard deviation (shaded area) across 3 trials.

4.2 RESULTS ON CIFAR-100 AND TINY-IMAGENET

Test Performance. In Figure 3a, we report the test performance on CIFAR-100 and Tiny-ImageNet. We compare the performance of six networks after training: 1) initialized dense network, $\mathcal{A}(\boldsymbol{\theta}_{\text{init}})$, 2) LRR subnetwork, $\mathcal{A}(\boldsymbol{\theta}_T^{\text{LRR}} \odot \boldsymbol{m}_T^{\text{LRR}})$, 3) AWS subnetwork, $\mathcal{A}(\boldsymbol{\theta}_T^{\text{AWS}} \odot \boldsymbol{m}_T^{\text{AWS}})$, 4) randomly initialized network masked with a signed mask of a LRR subnetwork, $\mathcal{A}(\text{abs}(\boldsymbol{\theta}_{\text{init}}) \odot \boldsymbol{s}_T^{\text{LRR}})$, 5) randomly initialized network with a signed mask of a AWS subnetwork, $\mathcal{A}(\text{abs}(\boldsymbol{\theta}_{\text{init}}) \odot \boldsymbol{s}_T^{\text{AWS}})$, and 6) randomly initialized network masked with a mask of a LRR subnetwork, $\mathcal{A}(\boldsymbol{\theta}_{\text{init}} \odot \boldsymbol{m}_T^{\text{LRR}})$. Note that $\mathcal{A}(\cdot)$ indicates a normal training algorithm without interpolating normalization layer parameters. First, we note that in most cases, the AWS solution (indicated by the green plots) achieves performance comparable to or better than the LRR solution (indicated by the blue plots). This addresses the potential concern that randomly interpolating normalization parameters in our method could adversely affect the performance of the AWS solution. Then, we transfer the signed mask from LRR ($\boldsymbol{s}_T^{\text{LRR}}$) and AWS ($\boldsymbol{s}_T^{\text{AWS}}$) to a randomly initialized network $\boldsymbol{\theta}_{\text{init}}$. In the case of LRR, the performance of a random initialized network masked with $\boldsymbol{s}_T^{\text{LRR}}$ after training (indicated by the orange plots) is similar to the case when the sign information is not used (indicated by the purple plots), lagging far behind the performance of LRR solution. On the other hand, we observe that in most cases, the performance of a randomly initialized network masked with $\boldsymbol{s}_T^{\text{AWS}}$ (indicated by the red plots) is comparable to that of AWS solution, which is similar to that of LRR solution. Finally, We observe that the signed mask obtained through AWS allows a randomly initialized network to perform comparably to a dense network after training at non-trivial sparsity as long as the AWS solution performs comparably to the dense network (indicated by the dotted line).

Stability to SGD-Noise and Linear Mode Connectivity. We further compare the effectiveness of $\boldsymbol{s}_T^{\text{AWS}}$ and $\boldsymbol{s}_T^{\text{LRR}}$ by investigating whether they can allow a randomly initialized network to reside within the basin of attraction of the AWS or LRR subnetworks. To this end, we first examine the SGD noise stability of $\mathcal{A}(\text{abs}(\boldsymbol{\theta}_{\text{init}}) \odot \boldsymbol{s}_T^{\text{AWS}})$ and $\mathcal{A}(\text{abs}(\boldsymbol{\theta}_{\text{init}}) \odot \boldsymbol{s}_T^{\text{LRR}})$. The results in Figure 3b demonstrate that $\mathcal{A}(\text{abs}(\boldsymbol{\theta}_{\text{init}}) \odot \boldsymbol{s}_T^{\text{LRR}})$, shown by the orange plots, exhibits significantly high error barriers when trained with different SGD randomness. On the other hand, $\mathcal{A}(\text{abs}(\boldsymbol{\theta}_{\text{init}}) \odot \boldsymbol{s}_T^{\text{AWS}})$, represented by the green plots, exhibits comparable or slightly higher error barriers when trained with different SGD randomness, compared to the reference stability of $\mathcal{A}(\boldsymbol{\theta}_T^{\text{LRR}} \odot \boldsymbol{s}_T^{\text{LRR}})$ shown by the blue plots. Moreover, we examine the linear mode connectivity between $\mathcal{A}(\text{abs}(\boldsymbol{\theta}_{\text{init}}) \odot \boldsymbol{s}_T^{\text{AWS}})$ or $\mathcal{A}(\text{abs}(\boldsymbol{\theta}_{\text{init}}) \odot \boldsymbol{s}_T^{\text{LRR}})$ and the AWS or LRR solution to confirm whether they are in the same basin. The results in Figure 3c demonstrates that transferring $\boldsymbol{s}_T^{\text{LRR}}$ to a randomly initialized network causes the network to converge to a solution with significantly high error barriers between the LRR solution, as indicated by the orange plots. By contrast, a random initialized network masked with $\boldsymbol{s}_T^{\text{AWS}}$ exhibits relatively low error barriers between the AWS solution, as indicated by the green plots, compared to the reference (blue plots).

Discussion. Our motivational experiments in Figure 2 show that the signed mask of the LRR subnetwork alone cannot transfer the generalization potential of the LRR subnetwork to a randomly initialized network; it is also necessary to transfer the normalization layer parameters of the LRR subnetwork. For the goal of LTH, we propose AWS that eliminates the need for the trained normalization parameters. The test performance in Figure 3a and the analysis of the SGD noise stability and linear mode connectivity in Figure 3b and Figure 3c show that a randomly initialized network masked with the signed mask of the AWS subnetwork lies in the basin of attraction of the AWS subnetwork. This demonstrates that $\boldsymbol{s}_T^{\text{AWS}}$ alone can transfer the generalization potential of the AWS subnetwork without the trained normalization layer parameters, brining us closer to the goal of LTH.

4.3 GENERALIZATION TO IMAGENET AND LAYER NORMALIZATION

To validate the effectiveness of AWS on a more challenging task, we evaluate AWS on ImageNet dataset. The results in Table 1 show that across various model and sparsity (i.e. 1- 'Remaining Params'), a randomly initialized network masked with the AWS-driven signed mask (i.e. $\mathcal{A}(\text{abs}(\boldsymbol{\theta}_{\text{init}}) \odot \boldsymbol{s}_T^{\text{AWS}})$) achieves performance comparable to both the AWS solution (i.e. $\mathcal{A}(\boldsymbol{\theta}_T^{\text{AWS}} \odot \boldsymbol{m}_T^{\text{AWS}})$) and the dense network after training, whereas masking with the LRR-driven signed mask ($\mathcal{A}(\text{abs}(\boldsymbol{\theta}_{\text{init}}) \odot \boldsymbol{s}_T^{\text{LRR}})$) fails to reach both the performance of the LRR solution (i.e. $\mathcal{A}(\boldsymbol{\theta}_T^{\text{LRR}} \odot \boldsymbol{m}_T^{\text{LRR}})$) and the dense network. Notably, we observe a similar trend in the results on MLP-Mixer, which uses only layer normalization for normalization. These results show that the signed mask of the AWS subnetwork can transfer the generalization potential of the AWS subnetwork

Table 1: **Experimental results on ImageNet.** θ_{init} indicates randomly initialized parameters and 'Remaining Params.' refers to the remaining parameter ratio. We present more results in Table 2.

Model	Method	Use AWS subnetwork	Trained from θ_{init}	Top-1 Acc. (%)	Remaining Params.
ResNet50	Dense Network	-	-	75.87 ± 0.32	1
	$\mathcal{A}(\theta_T^{\text{LRR}} \odot m_T^{\text{LRR}})$	✗	✗	76.12 ± 0.22	0.27
	$\mathcal{A}(\theta_T^{\text{AWS}} \odot m_T^{\text{AWS}})$	✔	✗	75.94 ± 0.33	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{LRR}})$	✗	✔	75.45 ± 0.31	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{AWS}})$	✔	✔	**75.89 ± 0.21**	
	$\mathcal{A}(\theta_T^{\text{LRR}} \odot m_T^{\text{LRR}})$	✗	✗	74.77 ± 0.29	0.14
	$\mathcal{A}(\theta_T^{\text{AWS}} \odot m_T^{\text{AWS}})$	✔	✗	74.81 ± 0.27	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{LRR}})$	✗	✔	72.65 ± 0.23	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{AWS}})$	✔	✔	**74.01 ± 0.32**	
MobileNetV2	Dense Network	-	-	69.13 ± 0.41	1
	$\mathcal{A}(\theta_T^{\text{LRR}} \odot m_T^{\text{LRR}})$	✗	✗	69.32 ± 0.40	0.41
	$\mathcal{A}(\theta_T^{\text{AWS}} \odot m_T^{\text{AWS}})$	✔	✗	69.02 ± 0.20	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{LRR}})$	✗	✔	67.98 ± 0.17	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{AWS}})$	✔	✔	**68.87 ± 0.39**	
	$\mathcal{A}(\theta_T^{\text{LRR}} \odot m_T^{\text{LRR}})$	✗	✗	68.42 ± 0.27	0.21
	$\mathcal{A}(\theta_T^{\text{AWS}} \odot m_T^{\text{AWS}})$	✔	✗	67.66 ± 0.27	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{LRR}})$	✗	✔	65.23 ± 0.28	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{AWS}})$	✔	✔	**67.79 ± 0.22**	
MLP-Mixer	Dense Network	-	-	58.21 ± 0.31	1
	$\mathcal{A}(\theta_T^{\text{LRR}} \odot m_T^{\text{LRR}})$	✗	✗	59.26 ± 0.32	0.21
	$\mathcal{A}(\theta_T^{\text{AWS}} \odot m_T^{\text{AWS}})$	✔	✗	60.41 ± 0.22	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{LRR}})$	✗	✔	56.80 ± 0.14	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{AWS}})$	✔	✔	**60.21 ± 0.26**	
	$\mathcal{A}(\theta_T^{\text{LRR}} \odot m_T^{\text{LRR}})$	✗	✗	58.12 ± 0.27	0.11
	$\mathcal{A}(\theta_T^{\text{AWS}} \odot m_T^{\text{AWS}})$	✔	✗	59.37 ± 0.22	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{LRR}})$	✗	✔	55.97 ± 0.26	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{AWS}})$	✔	✔	**59.10 ± 0.40**	

to a randomly initialized network on a more challenging dataset and even with layer normalization, further demonstrating the effectiveness of AWS.

4.4 Comparison to Related Work

There are existing works that also aim to search for an effective signed mask (Koster et al., 2022; Sreenivasan et al., 2022). Among them, we compare our AWS to GM Sreenivasan et al. (2022), as GM is evaluated on a more diverse and large-scale architecture than that in Koster et al. (2022). Figure 4 demonstrates that while GM performance falls short of that of the dense network across all levels of sparsity, a randomly initialized network masked with s_T^{AWS} can perform comparably to both the AWS subnetwork and the dense network until a sparsity of about 0.9. We also highlight that our work provides valuable insights into the role of parameter signs and the influence of normalization layers concerning finding a winning ticket. Moreover, it is noteworthy that the signed mask obtained by AWS can be applied to any random initialization. In contrast, GM trains a signed mask tailored for a specific initialization, which likely limits its applicability to other initialization.

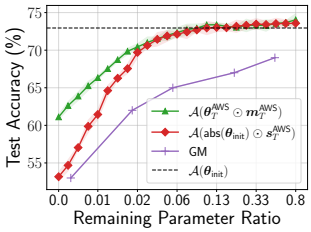


Figure 4: **Comparison to GM on CIFAR-100 with ResNet-32.** The results of GM are approximated from (Sreenivasan et al., 2022).

5 Conclusion

Finding a winning ticket is still an open problem in the field of lottery ticket hypothesis (LTH). In this work, we show that an effective signed mask, a sparse mask with sign information, is all we need to win the lottery. We observe that utilizing the signed mask and retaining normalization parameters of the subnetwork trained by learning rate rewinding (LRR) can transfer the generalization capability of the LRR subnetwork to a randomly initialized network. However, it is not truly a winning ticket when retaining a portion of trained parameters. To eliminate the reliance on the trained normalization layer parameters, we propose AWS, a slight variation of LRR, that encourages linear mode connectivity between the AWS subnetwork and its counterpart with initialized normalization parameters, allowing the AWS subnetwork to remain within its basin of attraction even after initializing all parameters. In contrast to the existing methods limited to finding a winning ticket with a predetermined initialization, we demonstrate that the signed mask of the AWS subnetwork can allow a randomly initialized network to reside within the basin of the AWS subnetwork, possibly leading the resulting network to generalize as well as the dense network. For future work, we will investigate the effectiveness of the signed mask acquired through AWS in the transfer learning scenario.

REFERENCES

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization, 2016. URL https://arxiv.org/abs/1607.06450.

Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32):15849–15854, 2019. doi: 10.1073/pnas.1903070116. URL https://www.pnas.org/doi/abs/10.1073/pnas.1903070116.

Rebekka Burkholz. Most activation functions can win the lottery without excessive depth. In *NeurIPS*, 2022.

Xiaohan Chen, Jason Zhang, and Zhangyang Wang. Peek-a-boo: What (more) is disguised in a randomly weighted neural network, and how to find it efficiently. In *ICLR*, 2022.

Arthur da Cunha, Emanuele Natale, and Laurent Viennot. Proving the lottery ticket hypothesis for convolutional neural networks. In *ICLR*, 2022.

Felix Draxler, Kambis Veschgini, Manfred Salmhofer, and Fred A. Hamprecht. Essentially no barriers in neural network energy landscape. In *ICML*, 2018.

Utku Evci, Yani A. Ioannou, Cem Keskin, and Yann Dauphin. Gradient flow in sparse neural networks and how lottery tickets win. In *AAAI*, 2022.

Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: finding sparse, trainable neural networks. In *ICLR*, 2019.

Jonathan Frankle, Gintare Karolina Dziugaite, Daniel M. Roy, and Michael Carbin. Linear mode connectivity and the lottery ticket hypothesis. In *ICML*, 2020a.

Jonathan Frankle, David J. Schwab, and Ari S. Morcos. The early phase of neural network training. In *ICLR*, 2020b.

C. Daniel Freeman and Joan Bruna. Topology and geometry of half-rectified network optimization. In *ICLR*, 2017.

Advait Gadhikar and Rebekka Burkholz. Masks, signs, and learning rate rewinding. In *ICLR*, 2024.

Timur Garipov, Pavel Izmailov, Dmitrii Podoprikhin, Dmitry Vetrov, and Andrew Gordon Wilson. Loss surfaces, mode connectivity, and fast ensembling of dnns. In *NIPS*, 2018.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

Sergey Ioffe and Christian Szegedy. Batch normalization: accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015.

Nils Koster, Oliver Grothe, and Achim Rettinger. Signing the supermask: Keep, hide, invert. In *ICLR*, 2022.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. University of Toronto, 2009.

Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015.

Bailin Li, Bowen Wu, Jiang Su, and Guangrun Wang. Eagleeye: fast sub-net evaluation for efficient neural network pruning. In *ECCV*, 2020.

Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. In *ICLR*, 2017.

Mingbao Lin, Rongrong Ji, Yan Wang, Yichen Zhang, Baochang Zhang, Yonghong Tian, and Ling Shao. Hrank: filter pruning using high-rank feature map. In *CVPR*, 2020.

Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. Learning efficient convolutional networks through network slimming. In *ICCV*, 2017.

Ekdeep Singh Lubana, Eric J. Bigelow, Robert P. Dick, David Krueger, and Hidenori Tanaka. Mechanistic mode connectivity. In *ICML*, 2023.

Xiaolong Ma, Geng Yuan, Xuan Shen, Tianlong Chen, Xuxi Chen, Xiaohan Chen, Ning Liu, Minghai Qin, Sijia Liu, Zhangyang Wang, and Yanzhi Wang. Sanity checks for lottery tickets: Does your winning ticket really win the jackpot? In *NeurIPS*, 2021.

Eran Malach, Gilad Yehudai, Shai Shalev-Shwartz, and Ohad Shamir. Proving the lottery ticket hypothesis: Pruning is all you need. In *ICML*, 2020.

Behnam Neyshabur, Zhiyuan Li, Srinadh Bhojanapalli, Yann LeCun, and Nathan Srebro. The role of over-parametrization in generalization of neural networks. In *ICLR*, 2019.

Quynh Nguyen. On connected sublevel sets in deep learning. In *ICML*, 2019.

Laurent Orseau, Marcus Hutter, and Omar Rivasplata. Logarithmic pruning is all you need. In *NeurIPS*, 2020.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019.

Mansheej Paul, Feng Chen, Brett W. Larsen, Jonathan Frankle, Surya Ganguli, and Gintare Karolina Dziugaite. Unmasking the lottery ticket hypothesis: What's encoded in a winning ticket's mask? In *ICLR*, 2023.

Alex Renda, Jonathan Frankle, and Michael Carbin. Comparing rewinding and fine-tuning in neural network pruning. In *ICLR*, 2020.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet large scale visual recognition challenge. *IJCV*, 2015.

Keitaro Sakamoto and Issei Sato. Analyzing lottery ticket hypothesis from pac-bayesian theory perspective. In *NeurIPS*, 2022.

Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: inverted residuals and linear bottlenecks. In *CVPR*, 2018.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

Kartik Sreenivasan, Jy yong Sohn, Liu Yang, Matthew Grinde, Alliot Nagle, Hongyi Wang, Eric Xing, Kangwook Lee, and Dimitris Papailiopoulos. Rare gems: Finding lottery tickets at initialization. In *NeurIPS*, 2022.

Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *ICML*, 2019.

Ilya Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, Mario Lucic, and Alexey Dosovitskiy. Mlp-mixer: An all-mlp architecture for vision. In *NeurIPS*, 2021.

Qingyang Wang, Michael A. Powell, Ali Geisa, Eric Bridgeford, Carey E. Priebe, and Joshua T. Vogelstein. Why do networks have inhibitory/negative connections? In *ICCV*, 2023a.

Qingyang Wang, Michael A. Powell, Ali Geisa, Eric W. Bridgeford, and Joshua T. Vogelstein. Polarity is all you need to learn and transfer faster. In *ICML*, 2023b.

Haoran You, Chaojian Li, Pengfei Xu, Yonggan Fu, Yue Wang, Xiaohan Chen, Richard G. Baraniuk, Zhangyang Wang, and Yingyan Lin. Drawing early-bird tickets: Towards more efficient training of deep networks. In *ICLR*, 2020.

Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *CVPR*, 2018.

Hattie Zhou, Janice Lan, Rosanne Liu, and Jason Yosinski. Deconstructing lottery tickets: Zeros, signs, and the supermask. In *NeurIPS*, 2019.

A IMPORTANCE OF PARAMETER SIGNS IN NORMALIZATION LAYERS.

In Section 3.2, we observe that preserving the signs of parameters in the LRR subnetwork while initializing their magnitudes, excluding those in normalization layers, causes the resulting network to stay within the basin of attraction of the LRR subnetwork, but it fails when the normalization layer parameters are initialized together. We claim that this occurs since the parameters in normalization layers rely more on the magnitude of parameters rather than their signs. The scaling factor in a normalization layer is nearly always positive, thus its sign information may be useless. In the case of the bias factor, it loses its sign information after initialized to 0, but it does not mean its sign information is not beneficial. To further validate our claim, we also compare the case where the bias factor in normalization layers is initialized to a constant, thus their sign information does not disappear after initialization. In Figure 5, we observe that maintaining the signs of bias factors (indicated by the purple plots) is not beneficial compared to the original initialization case (indicated by the orange plots), demonstrating our claim.

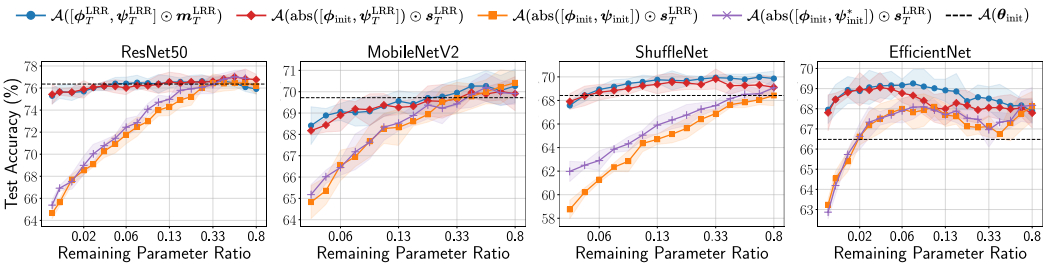
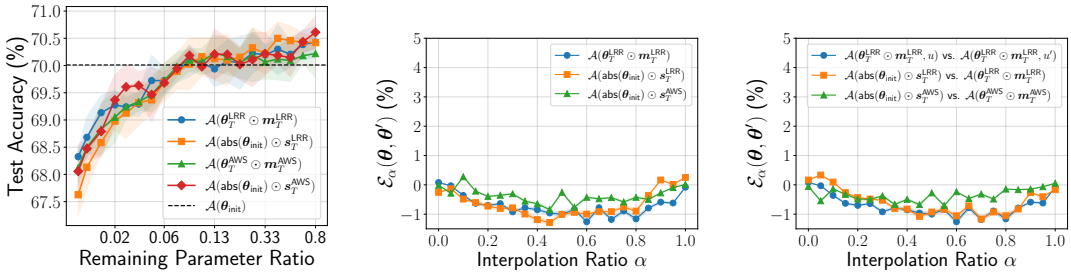


Figure 5: **Effect of transferring the sign of normalization layer parameters.** ψ^*_{init} denotes the initialized normalization layer parameters whose scaling and bias factors are set to 1 and 0.1. We conduct the experiments on CIFAR-100.

B IS AWS ALWAYS BETTER THAN LRR?

In the main manuscript, we show that after training, the performance of a randomly initialized network masked with $\boldsymbol{s}_T^{\text{LRR}}$ (i.e., $\mathcal{A}(\text{abs}(\boldsymbol{\theta}_{\text{init}}) \odot \boldsymbol{s}_T^{\text{LRR}})$) lags far behind that of the LRR solution. However, we found that for several network architectures, the performance of $\mathcal{A}(\text{abs}(\boldsymbol{\theta}_{\text{init}}) \odot \boldsymbol{s}_T^{\text{LRR}})$ is similar to that of the LRR solution. In Figure 6a, we present the test performance on VGG11-bn (Simonyan & Zisserman, 2015). We observe that $\mathcal{A}(\text{abs}(\boldsymbol{\theta}_{\text{init}}) \odot \boldsymbol{s}_T^{\text{LRR}})$ (indicated by the orange plots) achieves performance similar to that of the LRR solution (indicated by the blue plots). Thus, the performance difference between $\mathcal{A}(\text{abs}(\boldsymbol{\theta}_{\text{init}}) \odot \boldsymbol{s}_T^{\text{LRR}})$ and $\mathcal{A}(\text{abs}(\boldsymbol{\theta}_{\text{init}}) \odot \boldsymbol{s}_T^{\text{AWS}})$ (indicated by the red plots) is trivial. We also investigate the SGD noise stability of $\mathcal{A}(\text{abs}(\boldsymbol{\theta}_{\text{init}}) \odot \boldsymbol{s}_T^{\text{LRR}})$ and $\mathcal{A}(\text{abs}(\boldsymbol{\theta}_{\text{init}}) \odot \boldsymbol{s}_T^{\text{AWS}})$ and their linear mode connectivity to the corresponding LRR or AWS solution in Figure 6b and Figure 6c, respectively. We observe that the signed masks from both LRR and AWS enable a randomly initialized network to remain stable to SGD noise and converge to a solution with linear mode connectivity to the corresponding LRR or AWS solution. Thus, in some cases, LRR, without any modification, can yield an effective sign configuration that allows a randomly initialized network to win the lottery. However, we argue that AWS is more effective and generalizable to a wider range of more complex architectures as demonstrated in Figure 3.



(a) Test Accuracy (b) Stability to SGD noise (c) Linear mode connectivity

Figure 6: **Results on CIFAR-100 with VGG11-bn.**

Table 2: **Experimental results on ImageNet.** θ_{init} indicates randomly initialized parameters and 'Remaining Params.' refers to the remaining parameter ratio.

Model	Method	Use AWS subnetwork	Trained from θ_{init}	Top-1 Acc. (%)	Remaining Params.
ResNet50	Dense Network	-	-	75.87 ± 0.32	1
	$\mathcal{A}(\theta_T^{\text{LRR}} \odot m_T^{\text{LRR}})$	✗	✗	76.12 ± 0.22	0.27
	$\mathcal{A}(\theta_T^{\text{AWS}} \odot m_T^{\text{AWS}})$	✔	✗	75.94 ± 0.33	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{LRR}})$	✗	✔	75.45 ± 0.31	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{AWS}})$	✔	✔	**75.89 ± 0.21**	
	$\mathcal{A}(\theta_T^{\text{LRR}} \odot m_T^{\text{LRR}})$	✗	✗	75.62 ± 0.18	0.21
	$\mathcal{A}(\theta_T^{\text{AWS}} \odot m_T^{\text{AWS}})$	✔	✗	75.51 ± 0.13	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{LRR}})$	✗	✔	74.89 ± 0.27	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{AWS}})$	✔	✔	**75.27 ± 0.26**	
	$\mathcal{A}(\theta_T^{\text{LRR}} \odot m_T^{\text{LRR}})$	✗	✗	75.21 ± 0.32	0.17
	$\mathcal{A}(\theta_T^{\text{AWS}} \odot m_T^{\text{AWS}})$	✔	✗	75.22 ± 0.39	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{LRR}})$	✗	✔	73.99 ± 0.21	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{AWS}})$	✔	✔	**74.88 ± 0.28**	
	$\mathcal{A}(\theta_T^{\text{LRR}} \odot m_T^{\text{LRR}})$	✗	✗	74.77 ± 0.29	0.14
	$\mathcal{A}(\theta_T^{\text{AWS}} \odot m_T^{\text{AWS}})$	✔	✗	74.81 ± 0.27	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{LRR}})$	✗	✔	72.65 ± 0.23	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{AWS}})$	✔	✔	**74.01 ± 0.32**	
MobileNetV2	Dense Network	-	-	69.13 ± 0.41	1
	$\mathcal{A}(\theta_T^{\text{LRR}} \odot m_T^{\text{LRR}})$	✗	✗	69.32 ± 0.40	0.41
	$\mathcal{A}(\theta_T^{\text{AWS}} \odot m_T^{\text{AWS}})$	✔	✗	69.02 ± 0.20	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{LRR}})$	✗	✔	67.98 ± 0.17	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{AWS}})$	✔	✔	**68.87 ± 0.39**	
	$\mathcal{A}(\theta_T^{\text{LRR}} \odot m_T^{\text{LRR}})$	✗	✗	69.01 ± 0.12	0.33
	$\mathcal{A}(\theta_T^{\text{AWS}} \odot m_T^{\text{AWS}})$	✔	✗	68.89 ± 0.23	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{LRR}})$	✗	✔	67.34 ± 0.20	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{AWS}})$	✔	✔	**68.57 ± 0.33**	
	$\mathcal{A}(\theta_T^{\text{LRR}} \odot m_T^{\text{LRR}})$	✗	✗	68.88 ± 0.22	0.27
	$\mathcal{A}(\theta_T^{\text{AWS}} \odot m_T^{\text{AWS}})$	✔	✗	68.09 ± 0.23	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{LRR}})$	✗	✔	66.22 ± 0.22	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{AWS}})$	✔	✔	**68.17 ± 0.13**	
	$\mathcal{A}(\theta_T^{\text{LRR}} \odot m_T^{\text{LRR}})$	✗	✗	68.42 ± 0.27	0.21
	$\mathcal{A}(\theta_T^{\text{AWS}} \odot m_T^{\text{AWS}})$	✔	✗	67.66 ± 0.27	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{LRR}})$	✗	✔	65.23 ± 0.28	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{AWS}})$	✔	✔	**67.79 ± 0.22**	
MLP-Mixer	Dense Network	-	-	58.21 ± 0.31	1
	$\mathcal{A}(\theta_T^{\text{LRR}} \odot m_T^{\text{LRR}})$	✗	✗	59.26 ± 0.32	0.21
	$\mathcal{A}(\theta_T^{\text{AWS}} \odot m_T^{\text{AWS}})$	✔	✗	60.41 ± 0.22	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{LRR}})$	✗	✔	56.80 ± 0.14	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{AWS}})$	✔	✔	**60.21 ± 0.26**	
	$\mathcal{A}(\theta_T^{\text{LRR}} \odot m_T^{\text{LRR}})$	✗	✗	58.97 ± 0.26	0.17
	$\mathcal{A}(\theta_T^{\text{AWS}} \odot m_T^{\text{AWS}})$	✔	✗	60.24 ± 0.23	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{LRR}})$	✗	✔	56.78 ± 0.21	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{AWS}})$	✔	✔	**60.02 ± 0.19**	
	$\mathcal{A}(\theta_T^{\text{LRR}} \odot m_T^{\text{LRR}})$	✗	✗	58.34 ± 0.42	0.14
	$\mathcal{A}(\theta_T^{\text{AWS}} \odot m_T^{\text{AWS}})$	✔	✗	59.58 ± 0.38	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{LRR}})$	✗	✔	56.11 ± 0.21	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{AWS}})$	✔	✔	**59.57 ± 0.29**	
	$\mathcal{A}(\theta_T^{\text{LRR}} \odot m_T^{\text{LRR}})$	✗	✗	58.12 ± 0.27	0.11
	$\mathcal{A}(\theta_T^{\text{AWS}} \odot m_T^{\text{AWS}})$	✔	✗	59.37 ± 0.22	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{LRR}})$	✗	✔	55.97 ± 0.26	
	$\mathcal{A}(\text{abs}(\theta_{\text{init}}) \odot s_T^{\text{AWS}})$	✔	✔	**59.10 ± 0.40**	