On the impact of training stability on the Lottery Ticket Hypothesis

Anonymous authors
Paper under double-blind review

Abstract

The Lottery Ticket Hypothesis states that there exist sparse subnetworks (called 'winning' Lottery Tickets) within dense networks that, when trained under the same regime, achieve similar or better validation accuracy as the dense network. It has been shown that for larger networks and more complex datasets, an additional pretraining step is required for winning lottery tickets to be successfully found. Previous work linked the amount of pretraining required to instability to SGD noise via Linear Mode Connectivity (LMC). In this paper, we will dive deeper in the factors of training that influence SGD instability and as such the requirements for finding 'winning' tickets. We show that several techniques that have a positive influence on dense network generalization increase SGD instability, and as such hinder the extraction of 'winning' tickets. By dampening this instability via smart hyper-parameter selection, we show that we can extract 'winning' tickets that are more sparse and even outperform several tickets found with pretraining. The additional stability to SGD noise has as unexpected side effect that useful features for classification are encoded in the winning tickets purely by pruning. We show that these features do not emerge when trained with more instability, and that they are transferable to different datasets, as well as enable faster training of the ticket.

1 Introduction

The performance increase of AI models on different benchmarks has been significant over time, but this came at a cost. Small increases in benchmark scores, are typically the results of a large upscaling in model parameters, or training resources. Following a recent study by (Sevilla et al., 2022) the computational resources required to train a state-of-the-art neural network model has doubled roughly every 4 to 9 months. This unsustainable growth results in models that can only be trained by large institutions.

However, often these models are severely overparameterized and can be significantly pruned without loss of model performance. **?** These approaches follow a typical train-prune-finetune pipeline, which still requires training the model.

Recent work by Frankle & Carbin (2019) introduced the Lottery Ticket Hypothesis (LTH), which states that within an initialized neural network, there exists a sparse subnetwork that can be trained to similar accuracy as the dense network. However, this came under scrutiny, as it was found that for larger models and datasets, this hypothesis no longer holds. To remedy this, a new criteria called LMC has been introduced in Frankle et al. (2020), which measures the error across a linear interpolation path between two networks trained from the same initialization with different seeds. This lead to the inclusion of late-rewinding, pretraining the dense network to a state of SGD stability, and finding Lottery Tickets within that network. This approach has been called the Lottery Ticket Rewinding (LTR).

1.1 Contributions

- We show the influence of different parameters during the training process on the LMC metric, which shows how stable an initialization is to SGD noise.

- We further highlight the impact of these parameters on the predictive quality of Lottery Tickets, and show that the configuration with the best performing dense network, not necessarily leads to the best winning ticket.
- With careful hyperparameter selections, we can find Lottery Tickets that outperform tickets found with late rewinding, albeit at extreme sparsities.
- We highlight that the tickets found with stability can function as feature extractors when frozen, and that this property is transferable to other datasets.

2 Related Work

2.1 The Lottery Ticket Hypothesis

Empirical Analysis. Introduced by Frankle & Carbin (2019), the Lottery Ticket Hypothesis posits that within a dense neural network, there exists a sparse neural network that can achieve commensurate accuracy when trained in the same circumstances. This has been further extended by Frankle et al. (2020) to include late rewinding for more complex settings. Zhou et al. (2019) studies the different components of the LTH methodology, concludes that masking can be seen as training the network, and introduces binary Supermasks which can be applied on a random network to achieve significantly better validation accuracy than random chance. These Supermasks were later improved upon in (Koster et al., 2022) by using ternary masks. Ma et al. (2021) introduces hard sanity checks for Lottery Tickets, and find that a high learning rate hinders the discovery of Lottery Tickets. Evci et al. (2022) studies the Lottery Ticket Hypothesis in combination with gradient flow. They find that sparse networks generally train poorly due to limited gradient flow, however sparse tickets that suffer from the same issue do train faster and better. The authors hypothesize that this is due to the ticket and trained ticket being located in the same loss basin. Paul et al. (2022) ... Maene et al. (2021) ... T et al. (2022) ...

Transferability. Transferability of a Lottery Ticket to a different setting as the one for which it as extracted, has been a thorougly studied subject. By achieving this, it means that some of the computational costs incurred by finding the Lottery Ticket can be offset due to its versatility. Morcos et al. (2019); Mehta (2019) find that Lottery Tickets can transfer well between different natural image datasets. Sabatelli. et al. (2021) further extends this by transferring from natural image dataset to small-size non-natural image datasets such as medical or art datasets. They find that in some cases the tickets underperform compared to a ticket found for the target datasets, but they all outperform the dense network trained on the target dataset. These results reinforce the observation that Lottery Tickets outperform dense networks when trained on limited data (T et al., 2022). Chen et al. (2021) further extends the study of transferability, by considering transferability between different tasks such as classification, object detection and semantic segmentation. In the context of NLP, Desai et al. (2021) discovers similarly that Lottery Tickets transfer well under different data distributions.

Variants. Talk about the SLTH, elastic lottery ticket hypothesis, Early-bird tickets, ...

2.2 (Linear) Mode Connectivity.

Frankle et al. 2021 "Linear Mode Connectivity and the Lottery Ticket Hypothesis"

Zhou et al. 2023 "Going Beyond Linear Mode Connectivity: The Layerwise Linear Feature Connectivity"

Ainsworth et al. 2023 "GIT RE-BASIN: MERGING MODELS MODULO PERMUTATION SYMMETRIES"

Entezari et al. 2022 "The role of permutation invariance in linear mode connectivity of neural networks."

Fort et al. 2022 "Deep learning versus kernel learning: an empirical study of loss landscape geometry and the time evolution of the Neural Tangent Kernel"

Ferbach et al. 2024 "Proving linear mode connectivity of neural networks via optimal transport"

Iyer et al. 2024 "Linear Weight Interpolation Leads to Transient Performance Gains"

Adilova et al. 2024 "LAYER-WISE LINEAR MODE CONNECTIVITY"

3 Methodology

We start from commonly used training configurations (see **??**) for the Lottery Ticket Hypothesis, taken from Frankle et al. (2021), and study the impact of different components of the training configuration on train-time metrics, such as Linear Mode Connectivity Frankle et al. (2020), forgetting scores Toneva et al. (2019), and convergence speed. Next, we use the relationships between configuration parameters and the train-time metrics to explain phenomenons occurring in sparse lottery tickets.

3.1 The Lottery Ticket Hypothesis

Algorithm 1 Lottery Ticket Hypothesis with late rewinding.

1: Initialize a neural network with weights $\theta_0 \in \mathbb{R}^d$.
2: Initialize pruning mask $M = 1^d$.
3: Train θ_0 for p steps to θ_p. ▷ *Pretraining*
4: **for** $n \in \{1, \ldots, N\}$ **do** ▷ *Mask Search*
5: Train $M \odot \theta_p$ to convergence.
6: Prune the 20% of weights lowest magnitude.
Let $M[i]$=0 if the corresponding weight i is pruned.
7: **end for**
8: Train the final network $M \odot \theta_p$. ▷ *Sparse Training*

Defined by an iterative procedure (see Algorithm 1), the algorithm consists of two (or three) phases. In the optional *Pretraining* phase, the network is lightly trained to provide a more stable rewind point Frankle et al. (2020) for the next phase. The second phase is a *Mask Search* phase, in which the sparse network is repeatedly trained with the same training configuration as the dense network. The goal of this training is to identify parameters for pruning, as the lowest magnitude parameters at the end of this phase contribute least to the predictive performance of the network. Next, a fixed percentage of the lowest magnitude parameters are pruned and the remaining weights are reset to the rewind point (either the initialization or the pretrained network). Finally, the *Sparse Training* phase, is the phase in which the ticket is trained until completion, after which the network is usable in prediction. A winning lottery ticket is then defined as a pruned network that can attain similar (or better) validation accuracy as the dense network in commensurate training.

It is evident that the trainability of a Lottery Ticket depends on both the initial weights and the pruning mask. This directly translates to a dependency on the *Pretraining* phase for the initial weights and a dependency on the *Mask Search* phase for the pruning mask. As such, modifications to the configurations used in *Mask Search*, can greatly impact the final ticket. While it is prohibitively expensive to conduct a grid search on all possible parameters, due to the expensive nature of this procedure, instead we focus on the impact of a few parameters. This will allow us to better understand the impact on the trainability and predictive quality of the resulting lottery tickets.

Batch Size. In the case of large datasets, it is often infeasible to use the whole dataset during each update step of the network. As such, it is common to use minibatches of the dataset to update a network iteratively. Intuitively, if the batch size is smaller, then the resulting gradient is more influenced by the subsampling of the dataset and the batch composition can be distinctly different from the composition of the full dataset. It has been shown in Keskar et al. (2017) that using a larger batch size leads to a sharp minimizer which has a negative impact on the generalization of a neural network. We show similar impacts on generalization in the form of lower validation accuracy when training the dense networks.

Momentum. Introduced as a technique to speed up gradient descent, momentum employs a factor $\mu \in [0, 1]$, and allows gradient information to be carried over from previous batches in the weight update. As such, the

updates function similar to a exponential moving average. Recent work to understand the role of momentum in the generalization of Neural Networks has been undertaken by Jelassi & Li (2022), in which the authors argue that momentum leads to classifiers generalizing on small-margin samples, rather than memorizing those samples. The authors additionally find that the impact of momentum is more significant with higher batch sizes.

$$v_{t+1} = \mu v_t - \epsilon \nabla f(\theta_t) \tag{1}$$

$$\theta_{t+1} = \theta_t + v_{t+1} \tag{2}$$

Training epochs. *Hypothesis: limiting the number of training epochs, also is a way to limit the training instability, as we have less epochs, there is less noise that can influence the outcomes, and as such it is possible for the networks to still be in the same loss basin. While this has a negative effect on the accuracy (obviously!), this is also the fastest approach to generate stable lottery tickets, as we can limit the training time.*

3.2 Train-time Metrics

Linear Mode Connectivity. Introduced by Frankle et al. (2020), LMC is a metric that measures the error on a linear interpolating path between two networks trained from the same initialization. If there is a significant increase in error alongside the path, the initialization is said to be unstable to SGD noise. With this justification came the introduction of the *Pretraining* phase, as pretraining for a small amount results in a network stable to SGD noise.

Forgetting events. During training, a network learns to predict labels for its input samples. However, the learning is not monotonic, meaning that if a sample is predicted correctly at some iteration i, it possible that an iteration $i+k$ it is no longer correctly predicted. To measure these 'forgetting events', Toneva et al. (2019) introduced forgetting scores, which record whether a sample is learned correctly each time the network is fed the sample. Samples that are never forgotten once learned are called 'unforgettable samples'.

Convergence speed. Measuring how fast a neural network reaches optimal accuracy given a certain training configuration is a difficult task. We measure this by calculating the validation accuracy before training (acc_0) and after every epoch (acc_{ep}). We then measure the rate of increase, by averaging $\text{acc}_{ep} - \text{acc}_0$ over the duration of the training. To account for different optimal accuracies, we rescale this value. This formula can be interpreted as the (rescaled) area under the validation curve of the network during training.

$$AUC = \frac{1}{t} \sum_{ep=1}^{t} \frac{\text{acc}_{ep} - \text{acc}_0}{\text{acc}_t - \text{acc}_0} \tag{3}$$

3.3 Experimental setup

The main bulk of our experiments will be done on ResNet-18 + CIFAR-10. However, the experiments of Section 4 are repeated on ResNet-18 + CIFAR-100 and ResNet-34 + TinyImageNet to demonstrate applicability on datasets of different complexities.

We use configurations from T et al. (2022) as a starting point. More specifically, this means that we use a total training budget of 200 epochs in each iteration, after which we prune 20% of the weights. In the case of Late-Rewinding, this budget includes a *Pretraining* phase of 2 epochs. Training is done by minimizing the Cross-Entropy Loss with SGD, starting with a learning rate of 0.1 (or 0.2 for TinyImageNet) which is cosine annealed, and a weight decay of 5E-4. A full breakdown of the training configuration can be found in the appendix.

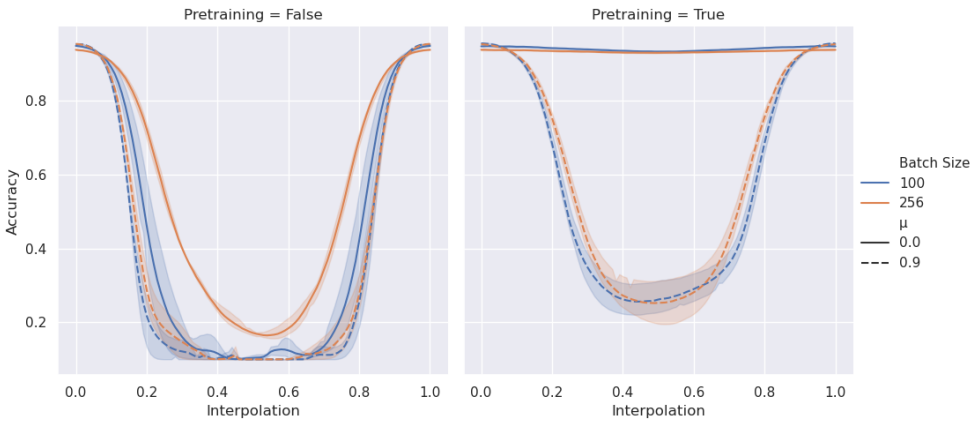


Figure 1: Linear Mode Connectivity for a dense ResNet-18 model trained on CIFAR-10 with **(left)** and without **(right)** pretraining, given different training configurations.

4 Experiments

4.1 Training Stability

Linear Mode Connectivity. We explore the influence of the momentum and batch size parameters on train-time performance of ResNet-18 models on CIFAR-10. For this, we start with a randomly initialized network and train this several times with different SGD noise. This allows us to record the LMC between different trained versions in Figure 1. Approaches that achieve better stability, i.e., achieving less severe accuracy drops along the interpolation path, also suffer from a reduced dense network accuracy (see 0.00% sparsity column in Table 2). This hints that while increased stability is linked to an increased performance of Lottery Tickets, instability is required for neural networks to achieve their best performance.

We notice that without late-rewinding, no configuration achieves a minimal error path between two convergences, showing that the networks are not stable to SGD noise. When adding two epochs of pretraining, stability in general increases. Approaches without momentum ($\mu = 0.0$) attain perfect stability, while approaches with momentum do not longer make random chance predictions during interpolation. To study when these approaches attain perfect LMC, we conduct a more thorough study in the appendix.

Example Forgetting. A similar indication of stability can be seen when recording the forgetting events for different training configurations. We notice that samples are much more often forgotten during training with momentum, indicating a more volatile approach to learning. This frequent forgetting and re-learning of certain samples seems to be coupled to better generalization, as we see an increase in evaluation accuracy. We hypothesize that the samples that are often forgotten are samples close to the decision boundary, which are generalized upon, rather than memorized under the application of momentum, as shown by Jelassi & Li (2022). Regarding batch size, we notice that in general, forgetting events occur more often at lower batch sizes, however at an extreme batch size, the number of forgetting events increases again. The number of unforgettable samples is also significantly impacted by batch size, where a very large batch size has a significantly lower number of unforgettable samples.

We further highlight the relationship between example forgetting and LMC in the appendix.

Limited Training. Instability to SGD noise is a function of the training time of a network. Indeed, if we start from the same initialization and we train multiple networks with different SGD noise for a limited number of epochs, the resulting networks will be linearly connected (*Show in a figure*). The likely reason for this, is that neural networks first learn general representations and then more specific features. As after a limited number of epochs, these networks are not yet well-converged, they have only learned the more general representations.

Table 1: Forgetting statistics and convergence statistics for different configurations used in training ResNet-18 models on CIFAR-10.

μ	Batch Size	Forgetting events	Unforgettable samples	Train Convergence	Val. Convergence
0.0	32	5.27 ± 7.43	34.87%	0.9567	0.9503
0.0	100	2.84 ± 4.18	**43.14%**	0.9776	0.9643
0.0	256	**2.22 ± 3.11**	42.76%	0.9752	0.9459
0.0	1024	3.32 ± 3.62	19.92%	0.9639	0.9520
0.9	100	8.94 ± 10.69	23.22%	0.9230	0.9132
0.9	256	5.64 ± 7.73	30.35%	0.9458	0.9338
0.9	1024	3.07 ± 4.05	32.84%	0.9639	0.9505

Table 2: Lottery Ticket performances for different training parameters at different sparsity levels.

Ticket Parameters				Sparsity				
Dataset	μ	Pretrain	Batch Size	0.00%	67.23%	89.26%	96.48%	98.84%
CIFAR-10	0.0	✗	100	94.58%	94.39%	93.93%	92.95%	91.10%
CIFAR-10	0.0	✗	256	93.63%	94.02%	93.93%	**93.39%**	**92.67%**
CIFAR-10	0.9	✗	100	**95.24%**	**94.99%**	**94.39%**	93.25%	91.41%
CIFAR-10	0.9	✗	256	95.18%	94.98%	94.29%	93.36%	91.39%
CIFAR-10	0.0	✓	100	94.62%	95.06%	**94.95%**	**94.71%**	**93.95%**
CIFAR-10	0.0	✓	256	93.54%	93.97%	93.94%	93.50%	92.77%
CIFAR-10	0.9	✓	100	**95.30%**	95.13%	94.85%	93.90%	92.31%
CIFAR-10	0.9	✓	256	95.24%	**95.16%**	94.55%	93.57%	91.86%

4.2 Impact on Lottery Tickets

In Table 2, we have listed the validation accuracies of Lottery Tickets found under different parameter configurations at certain critical sparsities. In the next paragraphs, we use these performances to highlight the different stages at which the SGD instability of the training procedure impacts the resulting Lottery Tickets. For a full overview of the results, as well as results on ResNet-18 + CIFAR-100, see the appendix.

Without late-rewinding. When not considering the additional stability of rewinding to a set of pretrained weights, we notice that unstable configurations dominate in the low sparsity regime. However, at deeper sparsities, we notice that stable configurations gain the upper hand. Following the observations that the batch size during training introduces less instability than momentum, we notice that the first equilibrium, at 89.26% sparsity, is in relation to batch size, while at 96.48% sparsity the advantages of momentum are wiped out compared to the most stable configuration ($\mu = 0.0$, batch size = 256). Finally, at an extreme sparsity (98.84%), the differences are even more pronounced. We should note that while unstable training leads to better performing dense networks, from which the performance

Hypothesis: We see for TinyImageNet, that the effect of batch size is much more pronounced. This is either related to the difficulty of the samples, or more likely, to the number of classes.

With late-rewinding. This approach provides stability by design and this is noticeable in a significantly increase in accuracy at most sparsities compared to the approach without pretraining. Interestingly, we once again notice that configurations with $\mu = 0.9$ begin to underperform at 95.60% sparsity, while the effect of smaller batch sizes is much less pronounced. In fact, at all sparsities we notice that smaller batch sizes outperform larger batch sizes, which differs from the observations without late-rewinding. Full results at all tested sparsities are in the appendix.

Table 3: Linear probing results at different locations in a ResNet-18 ticket extracted under different training configurations.

Ticket parameters			89.26% sparsity		96.48% sparsity	
μ	Pretraining	Batch Size	Block 4	Block 8	Block 4	Block 8
0.0	✗	100	33.61 ± 1.60%	30.93 ± 0.82%	33.45 ± 0.41%	30.51 ± 0.38%
0.0	✗	256	**51.38 ± 0.81%**	**82.87 ± 0.84%**	**52.17 ± 1.22%**	**80.85 ± 1.57%**
0.9	✗	100	29.48 ± 2.03%	28.65 ± 0.92%	29.39 ± 1.81%	25.71 ± 1.03%
0.9	✗	256	30.70 ± 0.52%	27.83 ± 1.55%	29.87 ± 1.42%	28.15 ± 0.21%
0.0	✓	100	52.08 ± 0.95%	73.11 ± 1.16%	49.32 ± 1.95%	65.81 ± 1.77%
0.0	✓	256	**56.27 ± 0.48%**	**84.53 ± 1.07%**	**55.91 ± 0.69%**	**83.71 ± 1.35%**
0.9	✓	100	39.27 ± 1.58%	48.92 ± 3.44%	36.74 ± 0.67%	42.49 ± 2.03%
0.9	✓	256	37.27 ± 5.31%	46.29 ± 7.13%	33.97 ± 1.76%	38.62 ± 3.30%
Permuted Baseline			36.92%	23.38%		

4.3 Less training epochs

Throughout training, the effects of SGD noise accumulates, leading to more instability. As such, by limiting the number of epochs trained, we can limit the instability incurred by these tickets, and limit the accuracy drop when pruning. We note that while there is a smaller drop in accuracy of tickets w.r.t. the dense network, these tickets are not practically useful, as the dense network is not fully converged.

5 Consequences of SGD stability

5.1 Linear evaluation

To determine the expressivity of the features within the tickets (pretrained weights + pruning mask), we devise a linear evaluation experiment inspired by linear probing (Alain & Bengio, 2017). At certain layers in the network, we insert a linear probe, which is a channel-wise pooling operation followed by a linear classification layer. While the linear probe is training, we freeze all other parameters in the network, such that we do not modify any of the features and the resulting accuracy correctly reflects the predictive quality of the features. For consistency between different parameter configurations, we use a single set of parameters to train the linear layer. While this might have some minimal influence on the validation accuracy of the probe, the general trends are explicit enough to warrant not exploring different hyperparameter settings for the probes.

An overview of the results can be found in Table 3. We notice that the stable configurations result in potent feature extractors at initialization. This shows that simply by pruning, useful features can be found from initialization or weights in early training. Additionally, using late-rewinding has a positive influence on the linear probing accuracy. We compare the results with a baseline, which consists of the best performing ticket with a random permuted mask. This baseline preserves the initialization and the layerwise sparsity, and thus shows that the remarkable performance can not be explained by those factors alone. Full results with additional visualizations can be found in the appendix.

The best performing masks can be likened to the Supermasks found by Zhou et al. (2019). The key difference between our observation and Supermasks is that our stable tickets are used as feature extractor, so a linear layer is finetuned on top, rather than evaluated without any training. In general, the Supermasks have been only extracted in simpler settings, namely for 3-layer fully connected networks on MNIST, or 2-/4-/6-layer CNNs on CIFAR-10, so no indication exists of applicability on more complex settings. Initial accuracies of a Supermask were limited to 80% on MNIST, or 24% on CIFAR-10, but by optimizing the mask, while keeping the initialization the same (reminiscent of the Strong Lottery Ticket Hypothesis), accuracies of up to 95.3% on MNIST and 65.4% on CIFAR-10 could be achieved.

Table 4: Validation accuracies of a 89.26% sparse ResNet-18 ticket when trained on a CIFAR-10 subset.

Ticket parameters			Subset sizes			
μ	Pretraining	Batch Size	1%	2%	5%	10%
0.0	✗	100	44.22 ± 1.47%	53.37 ± 1.41%	68.40 ± 0.96%	78.90 ± 0.30%
0.0	✗	256	**85.43 ± 0.58%**	**88.03 ± 0.62%**	**90.58 ± 0.21%**	**91.82 ± 0.11%**
0.9	✗	100	42.93 ± 2.12%	52.80 ± 3.05%	71.35 ± 1.49%	80.96 ± 0.29%
0.9	✗	256	37.02 ± 1.03%	47.38 ± 1.35%	63.93 ± 4.25%	75.44 ± 1.05%
0.0	✓	100	68.02 ± 2.81%	74.49 ± 0.60%	83.43 ± 0.25%	88.69 ± 0.48%
0.0	✓	256	**85.84 ± 0.70%**	**88.09 ± 0.47%**	**90.40 ± 0.21%**	**91.68 ± 0.11%**
0.9	✓	100	53.85 ± 3.47%	59.29 ± 2.70%	71.10 ± 0.55%	81.51 ± 0.90%
0.9	✓	256	45.06 ± 4.52%	53.55 ± 5.55%	66.54 ± 0.32%	76.99 ± 0.75%
Best Dense Network			42.54 ± 1.15%	51.25 ± 0.78%	67.15 ± 0.34%	78.42 ± 0.53%

5.2 Few-shot learning performances

Starting from the ticket, we next determine few-shot learning performances by using coresets during the *Sparse Training* phase. As we focus on small dataset sizes, we will use the random selection method, as this has been shown to work best in those cases (Guo et al., 2022).

We consider the following subset sizes: [1% , 2%, 5%, 10%]. Each class is equally represented in the dataset subset. To allow for comparability between TinyImageNet and CIFAR-10, we have chosen 1% as a bottom limit, as TinyImageNet features 500 images per class, while CIFAR-10 features 5000 images per class. Going lower than 1% could be feasible for CIFAR-10, but will be difficult for TinyImageNet, as then the subset selection process will significantly impact the performance of the trained network. Results for tickets at 89.26% sparsity on CIFAR-10 are listed in Table 4. Additional results and visualizations can be found in the appendix.

We notice that a significant portion of the validation accuracy attained with the full training dataset can be recovered in scenarios with higher SGD stability. In the most extreme case, we can attain 85% validation accuracy by training with 50 randomly chosen images per class of CIFAR-10. This is a gain of 43% over training the dense network with that subset. When increasing the number of samples in the subset, the resulting accuracy gain in these scenarios is not linear, but rather slows down. This is exemplified by the transition of 1% to 2%, for which the accuracy gain is higher than that of 5% to 10%. In the unstable scenarios we notice that the validation accuracy of the ticket often lies in a similar range as that of the dense network, or is even slightly lower. Additionally, the accuracy gain when increasing the number of samples shows a much more linear trend.

5.3 Dataset Transferability

Previous research has demonstrated that Lottery Tickets generalize well to other datasets. We further explore the observations made in Section 5.1 that certain tickets can function as feature extractors when frozen, and analyze the transferability of those features to different datasets. Starting from tickets extracted on ResNet-18 + CIFAR-10, we replace the linear layer with a new linear layer with a target-specific number of outputs. While this procedure leads to a loss of sparsity in the classification layer, this is the only possible approach short of resparsifying the linear layer, which might induce side effects.

Transferring is done to MNIST, CIFAR-100, TinyImageNet and EuroSat. The reasoning for these datasets is as following. MNIST is an easy dataset, albeit monochrome, which features different instances from CIFAR-10 (numbers, rather than objects). CIFAR-100 has been generated in the same manner as CIFAR-10, but images and classes are mutually exclusive between the datasets, and it contains more classes. TinyImageNet is a more challenging dataset, with more classes and less instances per class than CIFAR-10. Finally, EuroSat

Table 5: Transferability of frozen ResNet-18 tickets extracted on CIFAR-10 with different configurations.

Ticket parameters			Target dataset			
μ	Pretraining	Batch Size	MNIST	CIFAR-100	TinyImageNet	EuroSat
0.0	✗	100	82.11 ± 0.48%	16.66 ± 0.20%	7.42 ± 0.14%	70.46 ± 0.94%
0.0	✗	256	**97.25 ± 0.15%**	**40.57 ± 1.07%**	**18.13 ± 0.33%**	**85.21 ± 0.56%**
0.9	✗	100	77.03 ± 0.31%	12.73 ± 0.82%	5.49 ± 0.21%	61.00 ± 1.81%
0.9	✗	256	77.30 ± 1.60%	13.35 ± 0.74%	6.28 ± 0.92%	60.74 ± 5.21%
0.0	✓	100	96.90 ± 0.03%	36.18 ± 1.02%	18.48 ± 0.79%	82.40 ± 0.98%
0.0	✓	256	**97.45 ± 0.21%**	**41.63 ± 0.91%**	**21.56 ± 1.21%**	**85.47 ± 0.98%**
0.9	✓	100	90.85 ± 1.60%	17.34 ± 1.80%	8.46 ± 0.29%	69.37 ± 1.06%
0.9	✓	256	92.79 ± 2.67%	18.96 ± 4.22%	10.47 ± 1.91%	70.86 ± 1.43%
Finetuned dense network			71.17 ± 0.81%	..	4.41 ± 0.25%	..
Retrained dense network			99.58 ± 0.04%	77.40 ± 0.37%	..	98.51 ± 0.53%

features landscape images, rather than object images. Each of these datasets has characteristics not present in CIFAR-10, which serves to demonstrate the versatility of the features present within the lottery ticket.

We show the results for 89.26% sparsity tickets in Table 5, where we additionally compare with two baselines. The first baseline (*Finetuned Dense Network*) is achieved by replacing the last linear layer of a fully-trained dense network, freezing the other layers, and finetuning on the target dataset. The second baseline (*Retrained dense network*) uses the same training parameters to train on the target dataset from scratch. We show the full training curves in the appendix.

It can be clearly seen that the tickets with the least instability transfer best to the different datasets. While pretraining aids in increasing the transfer performance, it is unable to close the gap between the more unstable tickets and the best performing tickets. Additionally, we notice that in each case the features encoded in the sparse tickets transfer considerably better than those encoded in the dense network, even though the dense network is trained, while the tickets are not. This shows that by iteratively pruning, features emerge in the ticket that are more transferable than those obtained by training.

5.4 Linear Mode Connectivity of Tickets

6 Discussion

6.1 Loss Basin interpretation

Satisfying Linear Mode Connectivity can be interpreted as having two solutions that lie in the same linear connected loss basin. When extracting stable lottery tickets, we notice that the winning tickets all lie in the same loss basin (see Figure 2).

6.2 Instability via momentum or batch size

We have noticed empirically that using a the commonly used $\mu = 0.9$ for the momentum parameter, or a smaller batch size, results in a lower stability. This has been shown both in experiments with linear mode connectivity, and in the LTH itself. However, the cause of this instability is not precisely determined. We hypothesize that this is related to difficult-to-classify samples within the training dataset. We can however infer that the instability incurred by a lower batch size is less severe than using $\mu = 0.9$. This is can be seen in the LMC experiments, where pretraining is not sufficient to fully overcome the instability of momentum.

Small Batch Size. When using a small batch size, the gradient of a single (or multiple) such samples, influences the total batch gradient in a much greater fashion. This can lead to a gradient signal away from

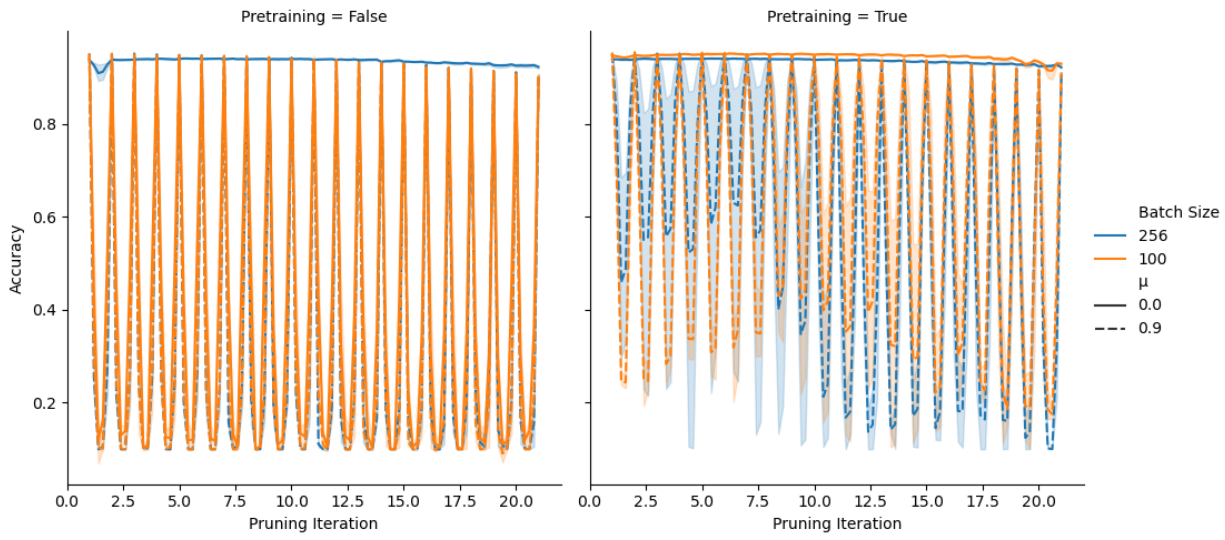


Figure 2: Caption

the easy optimization path. While this is desirable in the dense training regime, as this leads to better generalization, when optimizing a sparse network this leads to worse results.

Momentum. A similar observation can be made for the momentum parameter, albeit a little less straightforward. By applying momentum, the optimization step incorporates information of previous batches, thus amplifying the gradient effect of outliers in batches.

6.3 Frozen Feature Extractor

We notice in several experiments that, when instability is low enough, the tickets found via the LTH contain useful features without additional training. This suggests that some information of the dataset is encoded within the ticket, by pruning, when instability is low enough. The encoded information can be used to recover a large percentage (more than 90% in the case of CIFAR-10) of the validation accuracy of a fully trained dense network, by finetuning a linear classifier on the features. Additionally, these features aid in speeding up the convergence of the validation accuracy when training such a ticket.

This effect has been observed independently of the pretraining applied, albeit that with a pretrained ticket, the effects are amplified. Furthermore, we have shown that these features are not specific for the source dataset of the ticket, but can instead be transferred to other datasets, both in similar settings, or in radically different settings.

This behavior is reminiscent of the Strong Lottery Ticket Hypothesis, which poses that any trained network can be approximated by pruning connections from a sufficiently large untrained network. There is however a nuanced difference, in that the stable tickets found by our approach require a finetuned classification layer to achieve good validation accuracy.

7 Conclusion

In this research we study a number of training configurations and their influence on Lottery Tickets. We notice that commonly used hyperparameters, such a momentum and lower batch sizes have positive effect on the generalization of dense networks as already reported in the literature, but can negatively impact the Lottery Ticket Hypothesis. This impact is linked to the instability to SGD noise created by these hyperparameters during the training process. This is exemplified in a sharp increase of forgetting events

during training, the lack of Linear Mode Connectivity between different seeds of SGD noise, and lower convergence speeds in training and validation accuracy.

When applying these insights in training instability on the Lottery Ticket Hypothesis, we notice that in the cases with the least training instability, winning lottery tickets can be found at higher sparsities without any pretraining necessary. Even when considering additional stability in the form of pretraining, the more stable approaches show less accuracy degradation at higher sparsities.

On top of the higher validation accuracy at deeper sparsities, we show that tickets found with a higher stability exhibit several nice properties encoded by the pruned initialization. We notice that these tickets achieve better few-shot generalizability, and can in fact be used as frozen feature extractors to a remarkable accuracy.

Future work. While we have shown that stable tickets can be good feature extractors when frozen, we have not highlighted how exactly this behavior emerges, and is linked to training stability. We believe that further exploring this property might lead to tickets that can be extracted faster, albeit at a small accuracy cost.

References

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. In *International Conference on Learning Representations Workshops*, 2017.

Tianlong Chen, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Michael Carbin, and Zhangyang Wang. The lottery tickets hypothesis for supervised and self-supervised pre-training in computer vision models. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2021. doi: 10.1109/CVPR46437.2021.01604.

Shrey Desai, Hongyuan Zhan, and Ahmed Aly. Evaluating lottery tickets under distributional shifts. In *DeepLo@EMNLP-IJCNLP 2019 - Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource Natural Language Processing - Proceedings*, 2021. doi: 10.18653/v1/d19-6117.

Utku Evci, Yann Dauphin, Yani Ioannou, and Cem Keskin. Gradient flow in sparse neural networks and how lottery tickets win. In *Proceedings of the Thirty-Sixth AAAI Conference on Artificial Intelligence*, pp. 3082–3101, 2022.

Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *7th International Conference on Learning Representations, ICLR 2019*, 2019.

Jonathan Frankle, Gintare Karolina Dziugaite, Daniel Roy, and Michael Carbin. Linear mode connectivity and the lottery ticket hypothesis. pp. 3259–3269. PMLR, 2020. ISBN 2640-3498.

Jonathan Frankle, Gintare Karolina Dziugaite, Daniel Roy, and Michael Carbin. Pruning neural networks at initialization: Why are we missing the mark? In *International Conference on Learning Representations*, 2021.

Chengcheng Guo, Bo Zhao, and Yanbing Bai. Deepcore: A comprehensive library for coreset selection in deep learning. In *International Conference on Database and Expert Systems Applications*, pp. 181–195. Springer, 2022.

Samy Jelassi and Yuanzhi Li. Towards understanding how momentum improves generalization in deep learning. In *International Conference on Machine Learning*, pp. 9965–10040. PMLR, 2022.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *International Conference on Learning Representations*, 2017. URL `https://openreview.net/forum?id=H1oyRlYgg`.

Nils Koster, Oliver Grothe, and Achim Rettinger. Signing the supermask: Keep, hide, invert. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=e0jtGTfPihs`.

Xiaolong Ma, Geng Yuan, Xuan Shen, Tianlong Chen, Xuxi Chen, Xiaohan Chen, Ning Liu, Minghai Qin, Sijia Liu, and Zhangyang Wang. Sanity checks for lottery tickets: Does your winning ticket really win the jackpot? *Advances in Neural Information Processing Systems*, 34, 2021.

Jaron Maene, Mingxiao Li, and Marie-Francine Moens. Towards understanding iterative magnitude pruning: Why lottery tickets win. *arXiv preprint arXiv:2106.06955*, 2021.

Rahul Mehta. Sparse transfer learning via winning lottery tickets. *arXiv preprint arXiv:1905.07785*, 2019.

Ari Morcos, Haonan Yu, Michela Paganini, and Yuandong Tian. One ticket to win them all: generalizing lottery ticket initializations across datasets and optimizers. *Advances in neural information processing systems*, 32, 2019.

Mansheej Paul, Brett Larsen, Surya Ganguli, Jonathan Frankle, and Gintare Karolina Dziugaite. Lottery tickets on a data diet: Finding initializations with sparse trainable networks. *Advances in Neural Information Processing Systems*, 35:18916–18928, 2022.

Matthia Sabatelli., Mike Kestemont., and Pierre Geurts. On the transferability of winning tickets in non-natural image datasets. In *Proceedings of the 16th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 5: VISAPP,*, pp. 59–69. SciTePress, 2021. ISBN 978-989-758-488-6. doi: 10.5220/0010196300590069.

Jaime Sevilla, Lennart Heim, Anson Ho, Tamay Besiroglu, Marius Hobbhahn, and Pablo Villalobos. Compute trends across three eras of machine learning. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8. IEEE, 2022.

Mukund Varma T, Xuxi Chen, Zhenyu Zhang, Tianlong Chen, Subhashini Venugopalan, and Zhangyang Wang. Sparse winning tickets are data-efficient image recognizers. In Alice H Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022.

Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J. Gordon. An empirical study of example forgetting during deep neural network learning. In *International Conference on Learning Representations*, 2019.

Hattie Zhou, Janice Lan, Rosanne Liu, and Jason Yosinski. Deconstructing lottery tickets: Zeros, signs, and the supermask. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

A Full Lottery Ticket results

In Figure 3, we show the validation accuracies of Lottery tickets at different sparsities, when trained with different configurations. For convenience, we have highlighted the sparsities discussed in the main table with a dotted line. We notice that overall the rate of performance decay with momentum is higher than without momentum. Even though these tickets start with a higher accuracy than those without momentum, at a certain sparsity they underperform w.r.t. those without momentum, indicating that a more stable training process provides better quality tickets at high sparsities.

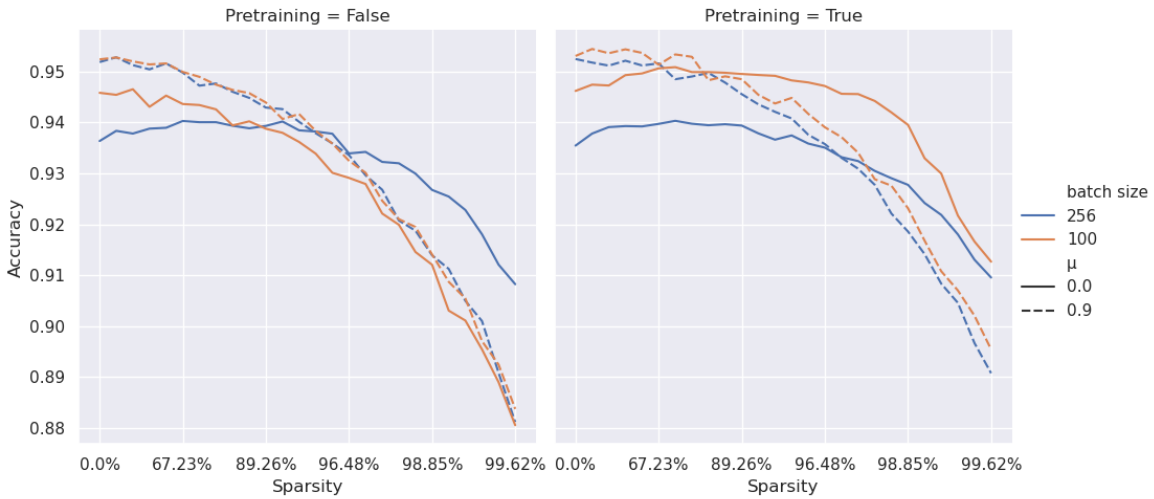


Figure 3: Validation accuracies at different sparsities for Lottery Tickets extracted from a ResNet-18 model on the CIFAR-10 dataset.

B Finding optimal rewind points

For several parameter configurations of ResNet-18, the default rewind point of 2 epochs was insufficient to provide stability to SGD noise. As such, we conduct a stability study to determine at which points (measured in epochs) SGD stability emerges. We follow the same definition of stability as in Frankle et al. (2020), meaning if the error across a linear path is less than 2%.

To speed up this procedure, we use a binary search algorithm to determine the first stable rewind point for each configuration. This allows us to find this rewind point in $\lceil log_2 200 \rceil + 1 = 9$ steps. We employ three random initializations for the rewind point, and to measure stability we use three different sets of SGD noise, for a total of 9 runs per configuration.

We list the results in Table 6. We notice that, as discussed in the main paper, both momentum and lower batch size impact stability negatively. In particular, the use of momentum has a significant impact, where for each configuration, we notice that Linear Mode Stability only emerges at >10% of the total training epochs.

Table 6: Stability to SGD noise for different parameter configurations of a ResNet-18 on CIFAR-10.

	$\mu = 0.0$				$\mu = 0.9$			
	32 BS	100 BS	256 BS	1024 BS	32 BS	100 BS	256 BS	1024 BS
First stable epoch	*5*	<=2	1	6.7 ± 1.2	88.7 ± 1.2	46.7 ± 0.9	22.0 ± 0.8	..

C Validation Accuracy Convergence.

In Figure 4 we have visualized the validation accuracy convergence of Lottery Tickets at different sparsities. We notice that the convergence scores without momentum are significantly better, and seem to suffer less from a decreasing factor than those with momentum. *conclusion here that these converge better.*

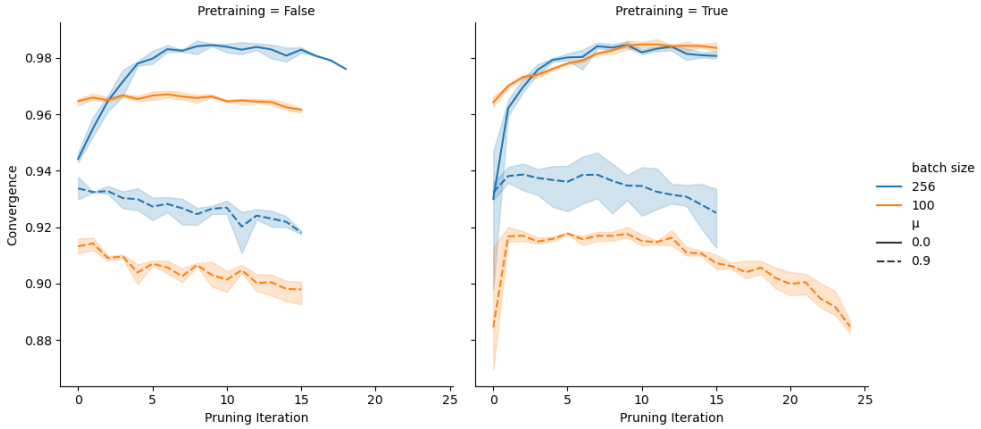


Figure 4: The convergence scores at different sparsities for Lottery Tickets extracted from a ResNet-18 model on the CIFAR-10 dataset.

D Forgetting events and instability

As shown in Toneva et al. (2019), a subset of the CIFAR-10 dataset, selected by removing samples with a low forgetting score, can be used to train a ResNet-18 model without loss of generalization, as compared to the full dataset. The authors show that this is true for removing up to the 30% of the samples with the lowest forgetting events. To see how the removal of these samples influences the instability to SGD noise, we repeat the training of a ResNet-18 model with $\mu = 0.9$, batch size $= 256$ at different subset levels.

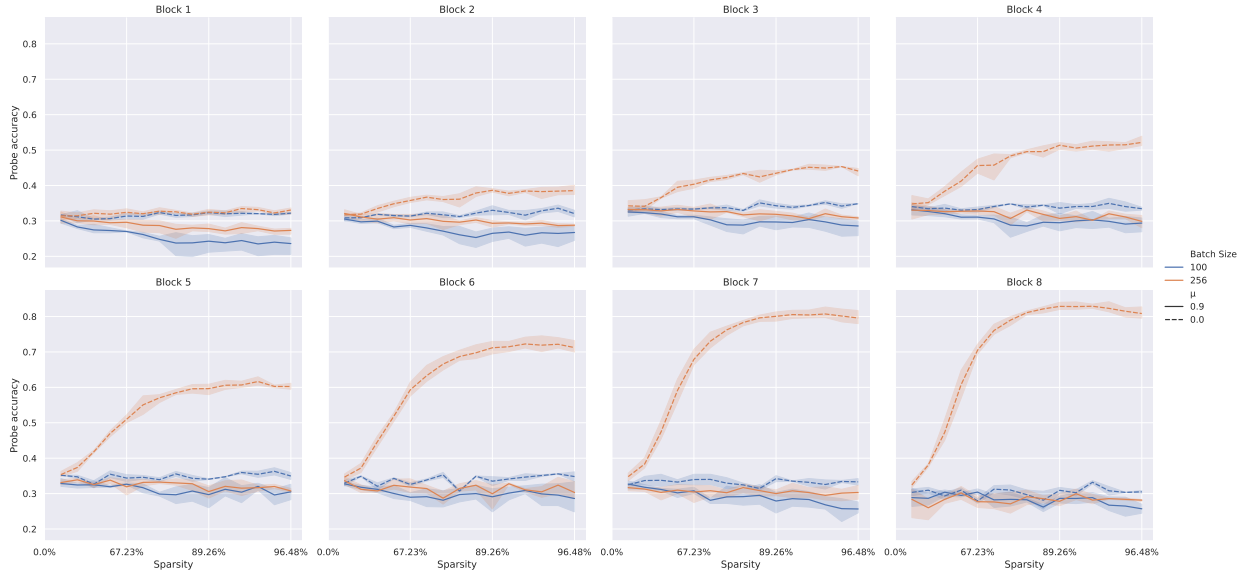
We notice in Table 7 that removing the 20% easiest samples has no significant impact on the validation accuracy, but that removing the 20% hardest samples significantly impacts the generalization. Via SGD stability analysis in **??**, we additionally notice that removing the harder samples leads to more stability, while removing the easier samples ...

Table 7: Validation accuracy of ResNet-18 + CIFAR-10 on different settings of samples removed.

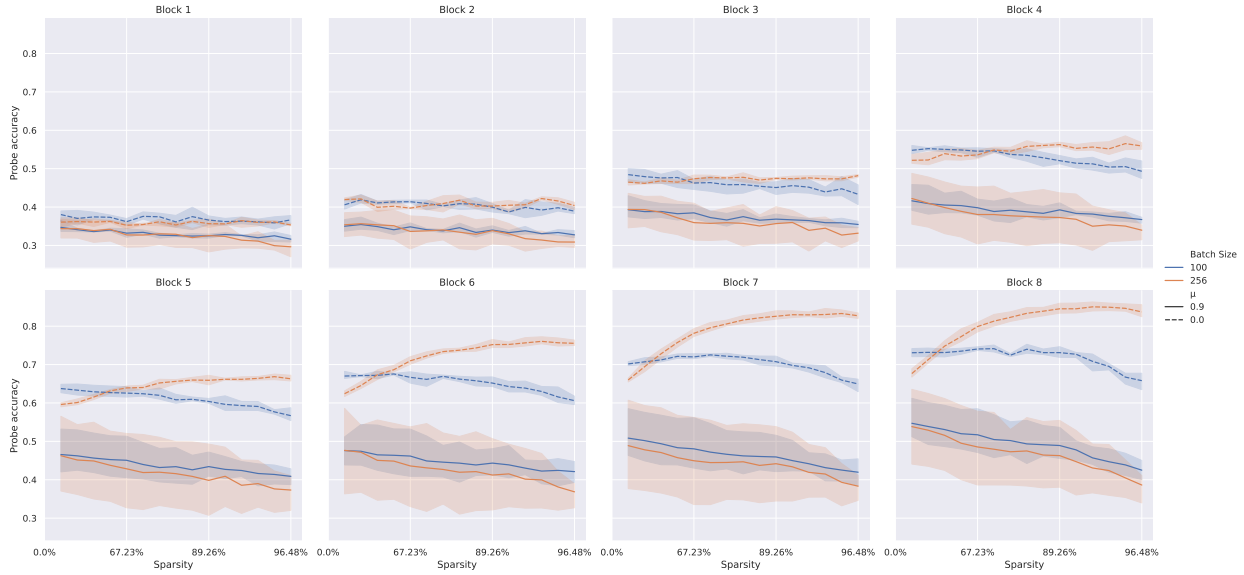
	Easiest samples removed		Hardest samples removed	
Original	20%	50%	20%	50%
95.18%	95.04%	93.79%	92.43%	87.43%

E Linear Probing

Full results of the linear probing experiment for different parameter configurations can be found in Figure 5. This includes results at each residual block and for each sparsity up until 96.48% (after which each configuration begins to lose accuracy rapidly).



(a)



(b)

Figure 5: Linear Probe accuracy for a ResNet-18 ticket given different parameter configurations. **(a)** Without late rewinding. **(b)** With late rewinding.

F Additional tables

Table 8: Caption

Ticket Parameters				Sparsity				
Dataset	μ	Pretrain	Batch Size	0.00%	67.23%	89.26%	96.48%	98.84%
CIFAR-100	0.0	✗	100	76.66 ± 0.22%	75.57 ± 0.18%	73.53 ± 0.52%		
CIFAR-100	0.0	✗	256	74.39 ± 0.23%	75.03 ± 0.36%	**74.46 ± 0.13%**		
CIFAR-100	0.9	✗	100	**78.54 ± 0.12%**	**76.60 ± 0.08%**	73.91 ± 0.28%		
CIFAR-100	0.9	✗	256	77.74 ± 0.37%	76.47 ± 0.10%	74.07 ± 0.17%		
CIFAR-100	0.0	✓	100	76.57 ± 0.26%	76.76 ± 0.20%	76.23 ± 0.39%		
CIFAR-100	0.0	✓	256	74.41 ± 0.18%	74.62 ± 0.16%	73.54 ± 0.19%		
CIFAR-100	0.9	✓	100	**78.18 ± 0.11%**	77.20 ± 0.32%	74.66 ± 0.42%		
CIFAR-100	0.9	✓	256	77.71 ± 0.25%	**77.57 ± 0.47%**	**77.24 ± 0.35%**		

Table 9: Caption

Ticket Parameters				Sparsity				
Dataset	μ	Pretrain	Batch Size	0.00%	67.23%	89.26%	96.48%	98.8
TImgNet	0.0	✗	100	61.14 ± 0.42%	59.76 ± 0.14%	57.79 ± 0.13%	54.47 ± 0.33%	
TImgNet	0.0	✗	256	59.33 ± 0.09%	59.95 ± 0.48%	**60.20 ± 0.19%**	**58.89 ± 0.17%**	
TImgNet	0.9	✗	100	**63.74 ± 0.36%**	**62.46 ± 0.30%**	35.95 ± 30.70%	35.69 ± 30.51%	
TImgNet	0.9	✗	256	61.57 ± 0.86%	60.37 ± 0.31%	56.93 ± 1.45%	52.76 ± 3.98%	
TImgNet	0.0	✓	100	60.91 ± 0.43%	62.90 ± 0.56%	**62.67 ± 0.54%**	**60.37 ± 0.17%**	
TImgNet	0.0	✓	256	59.27 ± 0.41%	60.01 ± 0.08%	60.08 ± 0.36%	58.59 ± 0.66%	
TImgNet	0.9	✓	100	**63.45 ± 0.53%**	**63.02 ± 0.33%**	60.10 ± 0.81%	57.73 ± 0.56%	
TImgNet	0.9	✓	256	61.66 ± 0.39%	61.22 ± 0.45%	58.95 ± 0.10%	55.91 ± 0.74%	