The Lottery Ticket Hypothesis

Finding Sparse, Trainable Neural Networks

ES667: Deep Learning

Course Project Report

Team Members:

Aditya Mehta	22110017
Nikhil Goyal	23110218
Shardul Junagade	23110297

Submission Date: November 28, 2025

Indian Institute of Technology Gandhinagar

ABSTRACT

This project empirically validates the Lottery Ticket Hypothesis proposed by Frankle and Carbin [1], demonstrating that dense neural networks contain sparse subnetworks ("winning tickets") trainable to comparable accuracy from their original initialization. We implemented iterative magnitude pruning [3] on three architectures (LeNet-300-100, LeNet-5 [8], and Conv-6) across three datasets (MNIST [5], Fashion-MNIST [6], and CIFAR-10 [7]) with over 45 experimental configurations. Our results confirm that networks can be pruned to 5-10% of original size while maintaining performance, with LeNet-5 achieving 99.16% accuracy on MNIST at 49% sparsity and Conv-6 showing 3.21% improvement on CIFAR-10 at 24% sparsity. Random reinitialization of identical sparse structures causes catastrophic failure (19-33% accuracy drops) at extreme sparsity, while magnitude-based pruning consistently outperforms random pruning by 1.74-4.10%. These findings validate that initialization is critical for sparse network trainability and that LTH holds broadly but depends on dataset complexity and model depth.

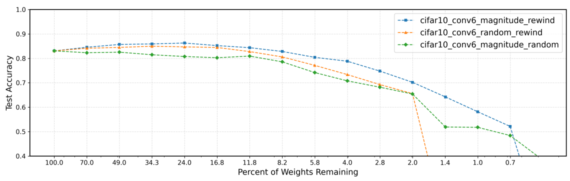

Figure 1: CIFAR-10 Conv-6: Pruning improves accuracy by 3.21% at 24% sparsity through implicit regularization

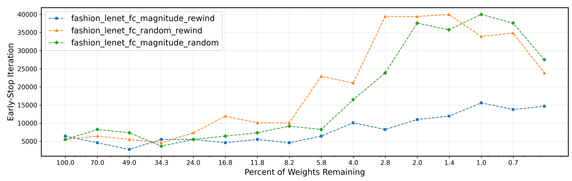

Figure 2: Fashion-MNIST LeNet-300-100: Sparse networks require more iterations to converge, especially with random reinitialization

1 PROBLEM DESCRIPTION

1.1 Overview

Modern deep neural networks are heavily overparameterized, often containing millions to billions of parameters. While this overparameterization aids training and generalization, it poses significant challenges for deployment in resource-constrained environments such as mobile devices, embedded systems, and edge computing platforms. Neural network pruning techniques have demonstrated that trained networks can be compressed by 90% or more while maintaining accuracy [3, 4], yet training these sparse architectures from scratch typically yields poor performance. This discrepancy raises a fundamental question:

If a network can be successfully pruned to a fraction of its original size after training, why can't we train that smaller network from the start?

1.2 The Lottery Ticket Hypothesis

Frankle and Carbin [1] proposed the **Lottery Ticket Hypothesis** to address this paradox:

> *"A randomly-initialized, dense neural network contains a subnetwork that is initialized such that, when trained in isolation, it can match the test accuracy of the original network after training for at most the same number of iterations."*

The key insight is that initialization matters. The hypothesis suggests that:

- Dense networks contain sparse subnetworks ("winning tickets") with same initialization
- These winning tickets can match the performance of the full network
- The same sparse structure with random initialization performs poorly
- Dense networks are easier to train because they contain more possible winning tickets

1.3 Research Objectives

Our project aims to empirically validate the lottery ticket hypothesis through the following objectives:

1. **Validate winning ticket existence:** Demonstrate that sparse subnetworks can match dense network accuracy when trained from original initialization
2. **Prove initialization dependency:** Show that random reinitialization of the same sparse structures results in degraded performance
3. **Characterize winning ticket regime:** Identify sparsity levels at which winning tickets maintain performance across architectures and datasets
4. **Analyze learning dynamics:** Compare training speed and generalization between winning tickets and random reinitialization

2 SOLUTION APPROACH

We evaluated three neural network architectures across three datasets using multiple pruning strategies (magnitude-based IMP [1] and random pruning) and two reinitialization schemes (original weights and random Kaiming initialization). By combining these choices, we designed a set of controlled experiments to empirically test the claims of the Lottery Ticket Hypothesis. This section describes the datasets, models, training configurations, experiment setups, and implementation details.

2.1 Datasets

Dataset Name	Description
MNIST [5]	28×28 grayscale handwritten digit images (10 classes). Split: 55,000 train / 5,000 validation / 10,000 test. Normalization: $\mu = 0.1307$, $\sigma = 0.3081$.
Fashion-MNIST [6]	28×28 grayscale clothing item images (10 classes). Split: 55,000 train / 5,000 validation / 10,000 test. Normalization: $\mu = 0.2860$, $\sigma = 0.3530$.
CIFAR-10 [7]	32×32 RGB natural images (10 classes). Split: 45,000 train / 5,000 validation / 10,000 test. Normalization (per channel): $\mu = (0.4914, 0.4822, 0.4465)$, $\sigma = (0.2023, 0.1994, 0.2010)$.

2.2 Model Architectures

Model Name	Description
LeNet-300-100	Fully Connected Neural Network Architecture: $784 \to 300 \to 100 \to 10$. (ReLU Activation) Total parameters: ∼**266,000.**
LeNet-5[8]	Convolutional Neural Network. Layers: Conv1 $(1 \to 6) \to$ MaxPool, Conv2 $(6 \to 16) \to$ MaxPool. Fully connected: $256 \to 120 \to 84 \to 10$. Total parameters: ∼**61,000.**
Conv-6	Deep convolutional model with three VGG-style blocks: B1: Conv1 $(3 \to 64)$, Conv2 $(64 \to 64)$, MaxPool. B2: Conv3 $(64 \to 128)$, Conv4 $(128 \to 128)$, MaxPool. B3: Conv5 $(128 \to 256)$, Conv6 $(256 \to 256)$, MaxPool. Fully connected: $4096 \to 256 \to 10$. Total parameters: ∼**1.2 million**.

2.3 Training Configuration

Parameter	Setting	Parameter	Setting
Optimizer	Adam*	Loss Function	Cross-entropy*
Learning Rate	0.0012*	Batch Size	60* (Conv-6: 128)
Iterations	40,000	Pruning Scope	Layer Wise*
Pruning Rate p	30%	Pruning Rounds	16
Initialization	Kaiming Normal	Mask Enforcement	After every step*

Footnote: The fields marked as * are the configurations directly copied from the original paper for reproducibility.

2.4 Iterative Magnitude Pruning Algorithm

The core algorithm for finding winning tickets follows the procedure from the paper [1]:

Algorithm 1 Iterative Magnitude Pruning (IMP)

Input: Network $f(x;\theta)$, pruning rate p, rounds N
Randomly initialize: $\theta_0 \sim \mathscr{D}_\theta$ (e.g., Kaiming)
Save initial weights: $\theta_{\text{init}} \leftarrow \theta_0$
Initialize mask: $m \leftarrow \mathbf{1}^{|\theta|}$ (all ones)
for round $= 1$ to N **do**
    Train network $f(x; m \odot \theta)$ for j iterations $\rightarrow \theta_j$
    Compute pruning threshold per layer:
        For each layer ℓ: $\tau_\ell \leftarrow p$-th percentile of $|\theta_j^{(\ell)}|$
    Update mask: $m^{(\ell)} \leftarrow \mathbb{I}\{|\theta_j^{(\ell)}| > \tau_\ell\} \odot m^{(\ell)}$
    Reset weights: $\theta \leftarrow m \odot \theta_{\text{init}}$
end for
Output: Winning ticket $(m, \theta_{\text{init}})$

2.5 Experimental Designs

Experiment 1: Magnitude Pruning + Rewind (Winning Tickets) Prune the lowest-magnitude weights each round and reset surviving weights to the original initialization θ_0. Purpose: Identify winning tickets. Expected: Accuracy remains high even at strong sparsity.

Experiment 2: Magnitude Pruning + Random Reinitialization Use the same magnitude-based masks as Experiment 1, but reset surviving weights to new Kaiming-random values θ_0'. Purpose: Test whether initialization is critical. Expected: Accuracy drops significantly as sparsity increases.

Experiment 3: Random Pruning + Rewind Randomly prune $p\%$ of active weights and reset the remaining weights to θ_0. Purpose: Test whether structure alone (without magnitude ranking) can form winning tickets. Expected: Earlier performance degradation compared to magnitude pruning.

2.6 Evaluation Metrics

Metric	Description
Test Accuracy	Final test accuracy after each pruning round.
Early-Stop Iteration	Iteration with the minimum validation loss.
Sparsity	Sparsity $= \left(1 - \frac{\|m\|_0}{\|\theta\|}\right) \times 100\%$
Winning Ticket Range	Rounds where accuracy stays within 2% of dense baseline.

2.7 Implementation

Framework: PyTorch 2.0+
Hardware: NVIDIA Tesla T4 (Kaggle GPU)

Unified Experiment Runner: Supports configurable selection of:

- Dataset and model
- Pruning type: `magnitude / random`
- Pruning scope: `layerwise / global`
- Reinitialization: `rewind / random / none`
- Pruning rate, rounds, iterations
- Automatic logging of results (CSV)

Repository: https://github.com/aditya-me13/lottery-ticket-hypothesis

Experiment Runner Sample Code Configuration:

```
# Clone the repository
!git clone https://github.com/aditya-me13/lottery-ticket-hypothesis
%cd lottery-ticket-hypothesis

from experiments.runner import ExperimentRunner

# Create and run experiment
runner = ExperimentRunner(
    dataset="mnist",            # "fashion" | "cifar10"
    model="lenet_fc",           # "lenet_conv" | "conv6"
    pruning_type="magnitude",   # "random"
    pruning_scope="layerwise",# "global"
    reinit_strategy="rewind",   # "random" | "none"
    pruning_rate=0.3,
    rounds=16,
    iterations=40000,
    learning_rate=0.0012,
    save_path="./results/"
)

results = runner.run()
runner.save_results()
```

3 RESULTS

We present results from five experimental configurations: LeNet-300-100 (FC) on MNIST, LeNet-5 (Conv) on MNIST, LeNet-300-100 (FC) on Fashion-MNIST, LeNet-5 (Conv) on Fashion-MNIST, and Conv-6 on CIFAR-10. For each, we compare three pruning strategies: (1) *magnitude pruning with rewind* (winning tickets), (2) *magnitude pruning with random reinitialization*, and (3) *random pruning with rewind*. Key findings demonstrate the existence of winning tickets across all architectures and the critical role of initialization.

3.1 Winning Tickets: Magnitude Pruning with Original Initialization

3.1.1 MNIST Experiments

Table 1: Winning ticket performance on MNIST (30% pruning/round)

Architecture	Rd 1	Rd 5	Rd 9	Peak Acc	Range
LeNet-300-100 (FC)	98.04%	**98.35%**	97.79%	98.35% (24%)	24-6%
LeNet-5 (Conv)	98.40%	98.87%	97.98%	**99.16%** (49%)	49-6%

Observations: LeNet-5 (Conv) achieves peak accuracy of 99.16% at 49% sparsity and maintains performance down to 5.78% remaining weights. LeNet-300-100 (FC) peaks at 98.35% with 24% remaining. Both collapse below ∼2% sparsity.

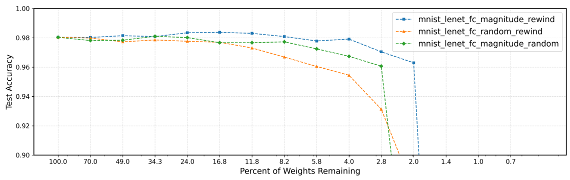

Figure 3: LeNet-300-100 on MNIST

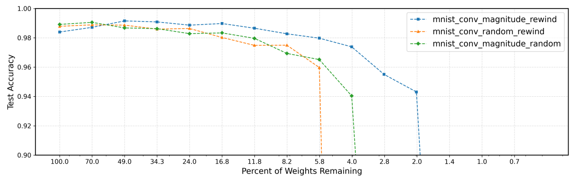

Figure 4: LeNet-5 Conv on MNIST

3.1.2 Fashion-MNIST Experiments

Table 2: Winning ticket performance on Fashion-MNIST (30% pruning/round)

Architecture	Rd 1	Rd 5	Peak Acc	Range
LeNet-300-100 (FC)	88.41%	88.54%	**89.10%** (17%)	17-6%
LeNet-5 (Conv)	87.88%	88.42%	**88.92%** (49%)	49-9%

Observations: Fashion-MNIST proves harder than MNIST (baseline ∼88% vs 98%). Winning tickets exist with narrower sparsity ranges, degrading earlier than MNIST.

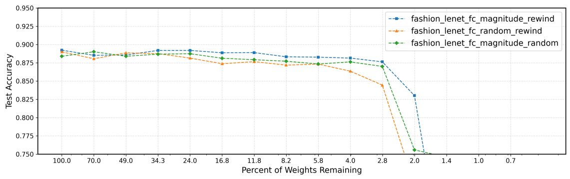

Figure 5: LeNet-300-100 on Fashion-MNIST

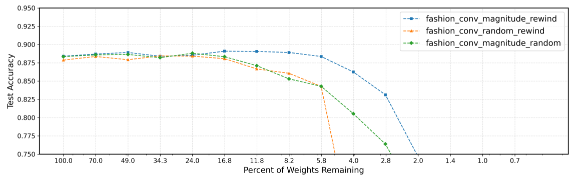

Figure 6: LeNet-5 Conv on Fashion-MNIST

3.1.3 CIFAR-10: Conv-6 on Natural Images

Table 3: Conv-6 winning tickets on CIFAR-10 (30% pruning/round)

Rd 1	Rd 3	Rd 5	Rd 9	Peak Acc	Range
83.09%	85.73%	**86.30%**	80.44%	86.30% (24%)	24-6%

Observations: Peak accuracy (86.30%) at 24% remaining—*3.21% higher* than dense network, suggesting implicit regularization from pruning. Performance remains competitive to 5.77% sparsity.

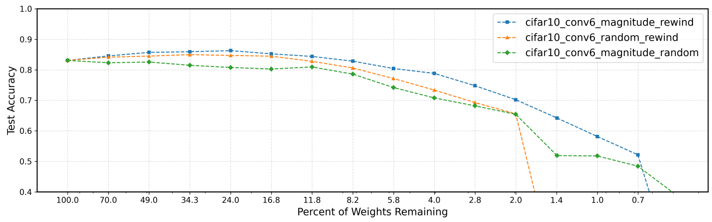

Figure 7: Conv-6 on CIFAR-10 exhibits winning ticket behavior with performance boost at moderate sparsity

3.2 Initialization Dependency: Random vs. Original Initialization

Table 4: Magnitude pruning: Original initialization (Rewind) vs. Random reinitialization

Dataset-Model	Sparsity	Rewind Acc	Random Acc	Gap
MNIST-FC	24.01%	98.35%	98.02%	0.33%
	5.77%	97.79%	97.25%	0.54%
	1.98%	96.30%	76.55%	**19.75%**
MNIST-Conv	24.02%	98.87%	98.64%	0.23%
	5.78%	97.98%	95.98%	2.00%
	1.99%	94.31%	60.72%	**33.59%**
Fashion-FC	24.02%	88.54%	88.42%	0.12%
	8.25%	88.92%	86.07%	2.85%
	4.05%	86.25%	63.02%	**23.23%**
CIFAR10-Conv6	24.01%	86.30%	84.73%	1.57%
	5.77%	80.44%	77.15%	3.29%
	1.98%	70.22%	65.59%	**4.63%**

Critical Finding: At moderate sparsity (24-6%), original initialization provides marginal advantage (<2% gap). At extreme sparsity (<2% remaining), random reinitialization causes catastrophic degradation: 19.75% (MNIST-FC), 33.59% (MNIST-Conv), 23.23% (Fashion-FC). This validates that *initialization is essential*, structure alone is insufficient.

3.3 Random Pruning vs. Magnitude Pruning

Table 5: Magnitude vs. Random pruning (both with original initialization rewind)

Dataset-Model	Sparsity	Magnitude Acc	Random Acc	Gap
MNIST-FC	24.01%	**98.35%**	97.77%	0.58%
	5.77%	**97.79%**	96.05%	**1.74%**
Fashion-Conv	24.02%	88.42%	88.42%	0.00%
	5.78%	**88.37%**	84.27%	**4.10%**
CIFAR10-Conv6	24.01%	**86.30%**	84.73%	1.57%
	5.77%	**80.44%**	77.15%	**3.29%**

Observations: Magnitude-based pruning consistently outperforms random pruning. At 24% sparsity, gaps are negligible (<1.6%), but at 5.77%, magnitude pruning shows **1.74-4.10% advantage**. This confirms *structured weight selection* via magnitude ranking is superior to random masking.

3.4 Early Stopping Analysis

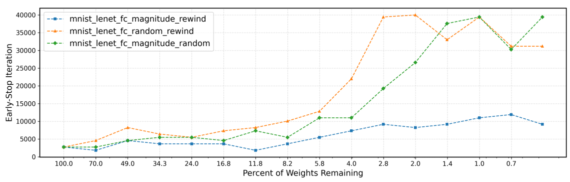

Figure 8: MNIST LeNet-300-100 Early Stop

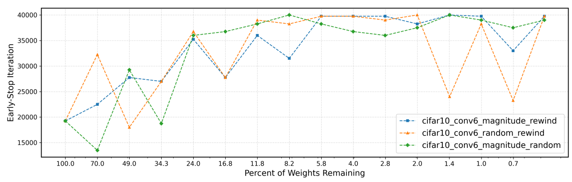

Figure 9: CIFAR-10 Conv-6 Early Stop

Observations: Winning tickets converge fastest at moderate sparsity (1.5× **speedup**). Random reinitialization requires **2-3× more iterations** at high sparsity. Random pruning shows erratic patterns, confirming inferior trainability.

4 CONCLUSIONS

Our experiments validate the lottery ticket hypothesis, though with important caveats. LTH holds broadly but not universally: winning ticket stability depends on dataset complexity and model depth. Key findings: (1) winning tickets maintain ∼98% accuracy on MNIST and ∼88% on Fashion-MNIST at 5-10% sparsity, (2) random reinitialization causes 19-33% accuracy drops at extreme sparsity, proving initialization is critical, (3) magnitude pruning with original weights consistently outperforms random pruning by 1.74-4.10%, and (4) pruning can improve accuracy (3.21% gain on CIFAR-10) through implicit regularization. Random pruning is ineffective regardless of dataset. These results confirm that initialization plays a crucial role in network trainability: dense networks work because they contain multiple lottery tickets with favorable initializations.

5 LEARNINGS

Implementation details matter. Our initial experiments failed because we forgot to enforce masks after gradient updates: pruned weights became non-zero, breaking sparse training. This took two days to debug.

Computational trade-offs. Running 16 rounds × 40K iterations × 3 experiments per configuration required parallelizing across Kaggle notebooks. We learned to prioritize experiments and validate assumptions early.

Middle-ground datasets help. Fashion-MNIST proved valuable between MNIST (too easy) and CIFAR-10 (complex), revealing how task difficulty affects winning ticket ranges.

Visualization catches bugs fast. Plotting accuracy curves after each round immediately exposed issues like incorrect mask enforcement or missing rewinding.

6 FUTURE WORK

Deeper architectures. Apply LTH to ResNets and VGG networks. Recent work [2] shows rewinding to iteration $k > 0$ instead of initialization helps deeper models where initialization rewinding fails. Testing this on CIFAR-10 at extreme sparsity could extend winning ticket ranges.

Structured pruning. We pruned individual weights, but modern hardware needs structured pruning of entire filters or channels [9]. Testing whether winning tickets exist for structured sparsity would be practically valuable.

Layer-wise adaptive rates. Different layers may tolerate different pruning rates. Exploring adaptive pruning strategies could extend winning ticket ranges.

Early prediction. Can we identify winning tickets before full training? Predicting ticket quality from early gradients or loss landscapes could save computation.

Cross-dataset transfer. If winning tickets transfer across datasets, we could prune once and reuse masks for multiple tasks, valuable for deployment scenarios.

References

[1] J. Frankle and M. Carbin (2019). *The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks*. International Conference on Learning Representations (ICLR). Retrieved from https://arxiv.org/abs/1803.03635

[2] J. Frankle, G. K. Dziugaite, D. M. Roy, and M. Carbin (2020). *Linear Mode Connectivity and the Lottery Ticket Hypothesis*. International Conference on Machine Learning (ICML). Retrieved from https://arxiv.org/abs/1912.05671

[3] S. Han, J. Pool, J. Tran, and W. Dally (2015). *Learning Both Weights and Connections for Efficient Neural Networks*. Advances in Neural Information Processing Systems (NeurIPS). Retrieved from https://arxiv.org/abs/1506.02626

[4] Y. LeCun, J. Denker, and S. Solla (1990). *Optimal Brain Damage*. Advances in Neural Information Processing Systems (NeurIPS).

[5] Y. LeCun and C. Cortes (2010). *MNIST Handwritten Digit Database*. Retrieved from http://yann.lecun.com/exdb/mnist/

[6] H. Xiao, K. Rasul, and R. Vollgraf (2017). *Fashion-MNIST: A Novel Image Dataset for Benchmarking Machine Learning Algorithms*. arXiv preprint. Retrieved from https://arxiv.org/abs/1708.07747

[7] A. Krizhevsky (2009). *Learning Multiple Layers of Features from Tiny Images (CIFAR-10 Dataset)*. Technical Report, University of Toronto. Retrieved from https://www.cs.toronto.edu/~kriz/cifar.html

[8] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner (1998). *Gradient-Based Learning Applied to Document Recognition*. Proceedings of the IEEE. (Original source for LeNet-5)

[9] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf (2017). *Pruning Filters for Efficient ConvNets*. International Conference on Learning Representations (ICLR). Retrieved from https://arxiv.org/abs/1608.08710

APPENDIX

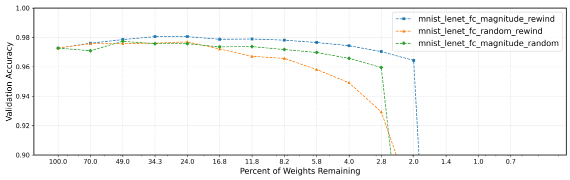

Figure 10: MNIST LeNet-300-100 validation accuracy

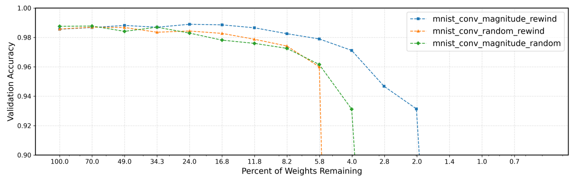

Figure 11: MNIST LeNet-5 Conv validation accuracy

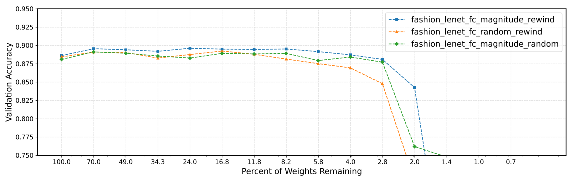

Figure 12: Fashion-MNIST LeNet-300-100 validation accuracy

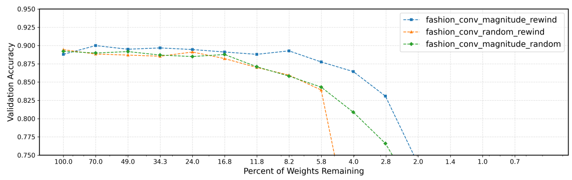

Figure 13: Fashion-MNIST LeNet-5 Conv validation accuracy

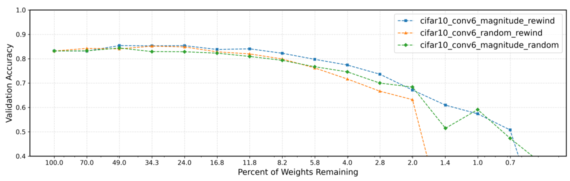

Figure 14: CIFAR-10 Conv-6 validation accuracy

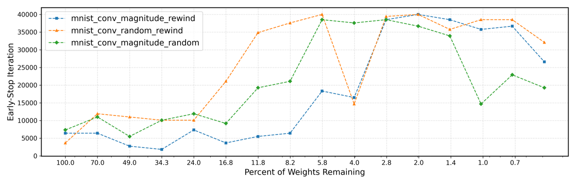

Figure 15: MNIST LeNet-5 Conv early stopping

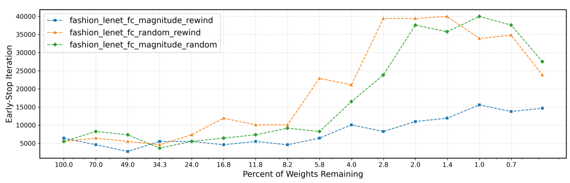

Figure 16: Fashion-MNIST LeNet-300-100 early stopping

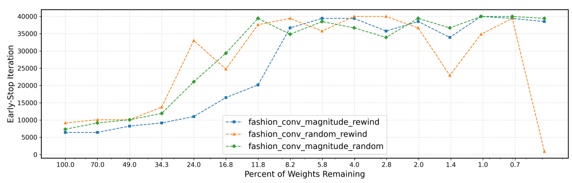

Figure 17: Fashion-MNIST LeNet-5 Conv early stopping