The Bird's Nest: Investigating Early-Bird Pruning on Transformers Using Various Pruning Techniques

Tejaswini Ramkumar Babu
Georgia Institute of Technology
tbabu8@gatech.edu

1. Target Problem

Training Transformer models like OpenAI's GPT-3, Google's Gemma, Meta's Llama, and Stability AI's generative AI models is time-consuming and costly. Identifying early-bird tickets within these models without having to complete the full training cycle could help save resources by allowing for early termination of training. However, this approach doesn't yet exist for transformers. I aim to investigate if the early-bird phenomenon exists in Transformer training, compare it with CNN training, and find better ways to decide when to stop training Transformer models. Furthermore, I will determine how much faster training could be with early-bird tickets, as well as the potential improvements in accuracy through the implementation of various pruning algorithms, such as Magnitude, Gradient, Random, and Undecayed [4].

2. Previous Work

Existing research has made significant strides in addressing the challenge of efficiency in deep neural network training, particularly through the identification of winning tickets and the development of efficient training algorithms. However, these efforts have primarily focused on convolutional neural networks [6] and computer vision tasks [1], and there's limited research on extending early-bird tickets to Transformers. Previous studies [6] [1] have demonstrated the existence of structured winning tickets in the early stages of training for language models like BERT. However, they do not directly address the unique characteristics of Transformer models used in current-day tasks. Furthermore, while these works propose efficient training algorithms leveraging early winning tickets, they do not explore the potential application of different pruning algorithms [2] to further accelerate training and reduce computational costs in Transformers. Therefore, additional research is needed to investigate the early-bird phenomena in Transformer training through various pruning algorithms.

3. Proposed Solution

My approach is to extend the early-bird pruning to Transformer models and compare the effectiveness of different pruning methods: magnitude, gradient, random, and undecayed pruning. I will employ these pruning techniques and evaluate which of them achieve better results when applied to Google's BERT [3] and Facebook's RoBERTa [5].

3.1. Milestones

1. Extend early-bird tickets to BERT.
2. Extend early-bird tickets to RoBERTa.
3. Investigate the effectiveness of the various pruning strategies on each model.
4. Assess and compare the pruned models for performance and resource efficiency.

3.2. Methodology

The study focuses on BERT and RoBERTa and how well the early-bird phenomenon performs on different architectural designs and pruning strategies. To evaluate this, I compared the accuracy of the model on our test datasets with the GLUE standard across each pruning strategy and compare it to a baseline unpruned model. The research aims to investigate the application of the Early Bird pruning technique to Transformer models, focusing on BERT and RoBERTa. This study compares the effectiveness of various pruning methods—Magnitude-Based, Gradient-Based, and Undecayed Pruning [4]—to determine their impact on the efficiency and accuracy of the model during training. A structured approach is used to train the initial model, apply pruning at different rates, and assess the performance changes.

The BERT and RoBERTa models were initially trained for 5 epochs to establish a baseline for performance comparison. This phase used a standard training routine without any pruning interventions to ensure that the model achieved

a stable state of accuracy and loss metrics, providing a solid foundation for subsequent pruning experiments.

After the initial training, three different pruning techniques were applied to the trained model:

- **Magnitude-Based Pruning (MP):** The weights with the smallest value are directly pruned, assuming that the smallest weights are less important and can be removed without significantly affecting the model's performance.

$$i = \arg\min_i |\theta_i|$$

- **Gradient-Based Pruning (GP):** The weights with the smallest gradients are pruned under the assumption that if the gradient is small, the weight is less active in updating during backpropagation.

$$i = \arg\min_i |-\theta_i \nabla_i L(\theta)|$$

- **Undecayed Pruning (UP):** UP combines both gradient and magnitude pruning in a linear combination, and it aims to strike a balance between the benefits of gradient-based pruning and the regularization effects induced by magnitude pruning. It considers both the impact of parameters on the loss function gradient and the magnitude in the pruning decision.

$$i = \arg\min_i \left(-\theta_i \nabla_i L(\theta) + \epsilon^2 \theta_i^2\right)$$

Each of these methods was applied at four different pruning rates: 0.3, 0.5, 0.7, and 0.9, which represent the percentage of weights pruned from the model.

The Early Bird [4] is a concept that a "winning ticket," or a high-performing subnetwork, can be identified early in the training process, potentially reducing the need for prolonged training durations.I extended this to BERT and RoBERTa, and I applied Early Bird pruning at the same rates as the other methods to compare its efficacy in identifying these subnetworks early in the training cycle.

In this experiment, each of the BERT and RoBERTa models' modules was looped through to identify which weights to prune based on the specific method's criteria. The weights were then pruned according to the predefined rates. This process was repeated for each pruning technique (MP, GP, and UP) and each rate (0.3, 0.5, 0.7, 0.9).

The experiment used PyTorch and the Hugging Face Transformers library for model management and training. Specific classes and functions were defined to handle the different pruning tasks, with checkpoints to assess the stability and performance of the model post-pruning. The model's performance was evaluated based on its accuracy in a classification task using GLUE's SST-2 benchmark.

4. Evaluation Results

Each model was trained for 20 epochs with an initially baseline accuracy of 0.9012 for BERT and 0.9223 for RoBERTa. Each pruning technique was applied for 20 epochs at rates of 0.3, 0.5, 0.7, and 0.9, and for undecayed pruning, the weight decay hyperparameter ϵ was set to 0.01 as it had previously [4] produced the best results.

The results of the experiment on BERT are summarized in Tables 1-3 below. The tables show the accuracy, loss, and runtime of the models pre-pruning and post-pruning for each pruning technique and rate.

Table 1. Magnitude Pruning with BERT

Prune Rate	**Metric**	**Pre-Train**	**Post-Train**
0.3	Accuracy	0.4656	0.7913
	Loss	0.7109	0.5892
	Runtime (s)	0.8689	0.841
0.5	Accuracy	0.5975	0.7787
	Loss	0.6883	0.5746
	Runtime (s)	0.8384	0.8416
0.7	Accuracy	0.5528	0.7936
	Loss	0.6786	0.5883
	Runtime (s)	0.8558	0.8415
0.9	Accuracy	0.5413	0.7878
	Loss	0.6863	0.5751
	Runtime (s)	0.8385	0.8341

Table 2. Gradient Pruning with BERT

Prune Rate	**Metric**	**Pre-Train**	**Post-Train**
0.3	Accuracy	0.8154	0.8544
	Loss	0.5136	0.5439
	Runtime (s)	0.6415	0.6414
0.5	Accuracy	0.7810	0.8268
	Loss	0.5817	0.4850
	Runtime (s)	0.6395	0.6356
0.7	Accuracy	0.5115	0.8062
	Loss	0.6758	0.5014
	Runtime (s)	0.6392	0.6363
0.9	Accuracy	0.5092	0.7924
	Loss	0.6930	0.5049
	Runtime (s)	0.6385	0.6389

Magnitude pruning shows a consistent decline in pre-training accuracy as the prune rate increases, but post-training recovery is strong for every rate. The highest recovery in accuracy is occurs at lower prune rates (0.3 and 0.5). However, post-training accuracy tends to plateau or slightly decline as the pruning magnitude reaches 0.7 and

Table 3. Undecayed Pruning with BERT

Prune Rate	Metric	Pre-Train	Post-Train
0.3	Accuracy	0.5034	0.8899
	Loss	1.4930	0.4906
	Runtime (s)	0.6399	0.6418
0.5	Accuracy	0.4908	0.7833
	Loss	1.3385	0.6158
	Runtime (s)	0.6449	0.6420
0.7	Accuracy	0.5092	0.7892
	Loss	0.8832	0.6973
	Runtime (s)	0.6501	0.6471
0.9	Accuracy	0.4908	0.7787
	Loss	0.7022	0.5585
	Runtime (s)	0.6392	0.6385

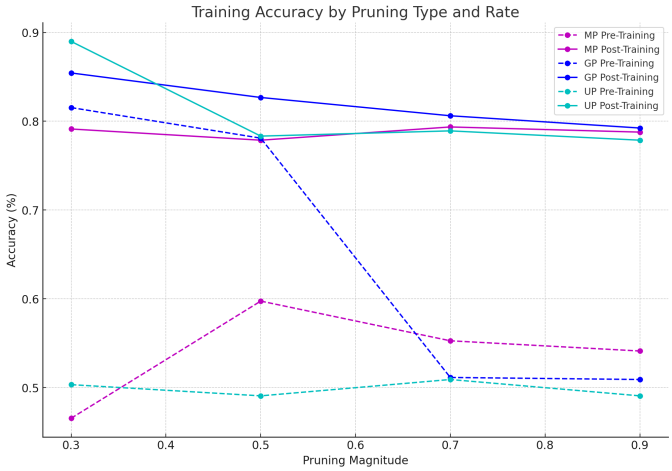

Figure 1. BERT: Training Accuracy by Pruning Type

above, possibly showing potential over-pruning where essential information is lost. Gradient pruning shows a relatively stable pre-training and post-training accuracies at lower prune rates (0.3 and 0.5), followed by a sharp decline at higher rates (0.7 and 0.9). The large decrease at higher rates indicates that the model loses important information after a certain threshold. Undecayed pruning shows the most variability, with the most noticeable drop in accuracy at the highest prune rate (0.9) There is a good recovery in post-training accuracy at lower prune rates, but the variability and significant changes at higher pruning rates raise concerns about its predictability and reliability. The runtime across different prune types and rates appears relatively stable, indicating that pruning has more impact on model accuracy and loss rather than on computational efficiency during training, indicating that the main trade-off with pruning is between model compactness and performance, not training speed.

Here are the results of the pruning time analysis for BERT:

Table 4. Pruning Time Analysis with BERT

Prune Type	Prune Rate	Time (in seconds)
Magnitude	0.3	271.9274
	0.5	269.9230
	0.7	264.7490
	0.9	263.9284
Gradient	0.3	271.3973
	0.5	270.8748
	0.7	262.8853
	0.9	262.2262
Undecayed	0.3	263.3554
	0.5	262.8618
	0.7	262.3103
	0.9	263.2686

The data shows a general trend where increasing the prune rate tends to reduce the training time across all pruning methods, which is likely because of the reduction in model size and complexity, which lessens the computational burden. Among the pruning methods, gradient pruning shows the most noticeable reduction in training time as the prune rate increases, suggesting it may be useful for tasks which requires both both model efficiency and training speed. Undecayed pruning shows consistent training times across different rates, which might be useful for tasks where predictability in training duration is important.

The results of the experiment on RoBERTa are summarized in Tables 5-7 below. The tables show the accuracy, loss, and runtime of the models pre-pruning and post-pruning for each pruning technique and rate.

Table 5. Magnitude Pruning with RoBERTa

Prune Rate	Metric	Pre-Train	Post-Train
0.3	Accuracy	0.8349	0.8704
	Loss	0.5284	0.4131
	Runtime (s)	0.8301	0.8332
0.5	Accuracy	0.5092	0.8727
	Loss	0.6937	0.3733
	Runtime (s)	0.8446	0.8295
0.7	Accuracy	0.5917	0.8291
	Loss	0.7297	0.4587
	Runtime (s)	0.8307	0.8369
0.9	Accuracy	0.4908	0.7638
	Loss	0.6985	0.5286
	Runtime (s)	0.8458	0.8226

Table 6. Gradient Pruning with RoBERTa

Prune Rate	Metric	Pre-Train	Post-Train
0.3	Accuracy	0.5092	0.7592
	Loss	0.6956	0.6968
	Runtime (s)	0.6364	0.6440
0.5	Accuracy	0.5906	0.8188
	Loss	0.9847	0.4579
	Runtime (s)	0.6390	0.6376
0.7	Accuracy	0.4908	0.7892
	Loss	0.7481	0.6977
	Runtime (s)	0.6352	0.7014
0.9	Accuracy	0.4908	0.8085
	Loss	0.6934	0.4697
	Runtime (s)	0.6377	0.6408

Table 7. Undecayed Pruning with RoBERTa

Prune Rate	Metric	Pre-Train	Post-Train
0.3	Accuracy	0.5287	0.5092
	Loss	0.8797	0.6941
	Runtime (s)	0.6398	0.6427
0.5	Accuracy	0.4908	0.5092
	Loss	1.5504	0.6959
	Runtime (s)	0.6365	0.6308
0.7	Accuracy	0.5092	0.5092
	Loss	0.7291	0.6974
	Runtime (s)	0.6392	0.6391
0.9	Accuracy	0.4908	0.5092
	Loss	0.6959	0.6977
	Runtime (s)	0.6353	0.6335

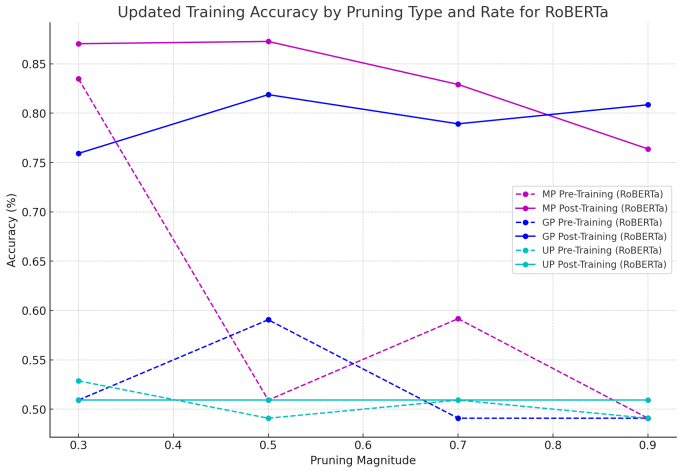

Figure 2. RoBERTa: Training Accuracy by Pruning Type

Magnitude pruning for RoBERTa shows varied pre-training accuracy for the prune rates, with the best initial accuracy at a lower prune rate of 0.3. Post-training, there is a good recovery in accuracy, but the large decrease in pre-training accuracy at the highest prune rate (0.9) suggests potential over-pruning, where important information may be lost at first but somewhat recoverable with further training. Gradient Pruning shows low pre-training accuracies across all prune rates, suggesting a loss of important parameters during the pruning process. However, there is relatively large improvement in post-training accuracy, especially at higher prune rates (0.7 and 0.9), showing that the model can compensate for the initial loss through re-training. Undecayed pruning shows the least effectiveness with a flat line, with both pre-training and post-training accuracies hovering around similar lower values across all prune rates. The small improvements or consistent results in post-training sessions show that this pruning method for RoBERTa might be retaining non-essential parameters that do not significantly enhance model performance.

Table 8. Pruning Time Analysis with RoBERTa

Prune Type	Prune Rate	Time (in seconds)
Magnitude	0.3	272.4830
	0.5	287.2703
	0.7	270.5454
	0.9	274.9230
Gradient	0.3	267.1750
	0.5	266.8891
	0.7	266.8741
	0.9	267.5033
Undecayed	0.3	264.8939
	0.5	266.6482
	0.7	268.1117
	0.9	265.6707

Unlike with BERT, the data for RoBERTa shows that, regardless of the pruning method or ratio, the time spent for training does not change. Gradient based pruning seems to have the least variability in time across various pruning ratios, but overall, unlike BERT, pruning RoBERTa does not seem to be a guaranteed way to improve the computational speed for training and inference. This may reveal that pruning as a whole is not uniformly viable as a method of reducing transformer model complexity.

5. Conclusion

In conclusion, an exploration of the various pruning strategies when applied to transformers, taken in combination with EarlyBird suggests that EarlyBird is a highly effective way to prune transformer models, due to its fast convergence over pretrained models. However, analysis

over various pruning strategies reveals that there is NOT a one-size-fits-all approach to picking a particular pruning method. In the future, I hope to investigate more models to see how pruning generalizes across transformers. In addition, I also hope to explore ways to speed up EarlyBird for transformers.

References

[1] Xiaohan Chen, Yu Cheng, Shuohang Wang, Zhe Gan, Zhangyang Wang, and Jingjing Liu. Earlybert: Efficient bert training via early-bird lottery tickets, 2021. 1

[2] Hongrong Cheng, Miao Zhang, and Javen Qinfeng Shi. A survey on deep neural network pruning-taxonomy, comparison, analysis, and recommendations, 2023. 1

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019. 1

[4] Aidan Good, Jiaqi Lin, Hannah Sieg, Mikey Ferguson, Xin Yu, Shandian Zhe, Jerzy Wieczorek, and Thiago Serra. Recall distortion in neural network pruning and the undecayed pruning algorithm, 2022. 1, 2

[5] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach, 2019. 1

[6] 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, 2022. 1