HyperSparse Neural Networks: Shifting Exploration to Exploitation through Adaptive Regularization

1. Abdullah Dahabre 2281427
2. Omar Ibrahim 2100502

1 Reproducibility Summary:

1.1 The Scope of Reproducibility

Our study aims to validate the implementation and functionality of Adaptive Regularized Training (ART) and the HyperSparse regularization method. We replicate the experimental setup using CIFAR-10 and CIFAR-100 datasets with ResNet-32 and VGG-19 models. We focus on reproducing reported classification accuracy and optimization time improvements, particularly at high sparsity levels of up to 99.8%. We compare our results with those reported in the original paper, assessing convergence times and final accuracy, and analyze the balance between exploration and exploitation during training, as well as confirm weight and gradient distributions aligned with the original claims.

1.2 Results

Our experimental findings validate the effectiveness of Adaptive Regularized Training (ART) and the HyperSparse regularization method in developing resource-efficient neural networks. We successfully reproduce the reported improvements in classification accuracy and optimization time across CIFAR-10 and CIFAR-100 datasets using ResNet-32 and VGG-19 models. Particularly noteworthy are the notable gains observed at high sparsity levels of up to 99.8%, demonstrating the robustness of the proposed methods. Comparison with other sparsification techniques confirms the superiority of ART and HyperSparse in achieving faster convergence times and higher final accuracy. Additionally, our analysis of weight and gradient distributions during training aligns with the original claims, further validating the efficacy of the proposed approaches.

1.3 What was easy

The paper described their approaches clearly, making the experimental workflow easy to follow. All the instructions and steps in the README file in the code were obvious and simple so we didn't face any problems following them. Also, the datasets were downloaded smoothly.

1.4 What was difficult

Setting up the environment was a little bit challenging. Also, the running time of the code was so long that it took about seven days to finish training. It took that long because we couldn't train it on two machines due to the weakness of one of them because the project was massive. Since we changed the paper several times due to the machine's power, the difficulty of importing the datasets, and installing libraries problems, we had shortness of time so we couldn't finish all the experiments provided in the paper. We were only able to do four experiments due to the long training time and since the TinyImageNet dataset takes much longer than both CIFAR-10 and CIFAR-100 we couldn't do any experiments on it, we did one experiments on CIFAR-10 and three on CIFAR-100.

1.5 Communication with original authors

It wasn't necessary to contact the author, as we didn't find the need to do so, everything was pretty obvious.

2 Introduction

Recent advancements in neural networks have led to significant improvements in accuracy across various tasks. However, this progress comes with increased computational complexity, resulting in high energy consumption and reduced interpretability. Sparse neural networks offer a solution by maintaining performance while reducing complexity. In this study, we replicate and evaluate the Adaptive Regularized Training (ART) method for creating highly sparse neural networks, as introduced by Glandorf et al. Additionally, we introduce HyperSparse, a new adaptive regularization method that penalizes smaller weights, enhancing accuracy and optimization speed, particularly in high-sparsity regimes.

3 Scope of reproducibility

Our goal is to replicate the findings of Glandorf et al. by implementing their ART method and evaluating its performance on the CIFAR-10 and CIFAR-100 datasets using ResNet-32 and VGG-19 models. We aim to verify the claim that ART can achieve high sparsity levels up to 99.8% while retaining model performance.

4 Methodology

We followed the methodology described by Glandorf et al. closely. The process involves three main steps:

Pre-training a Dense Model:

We trained a dense neural network to converge without any regularization.

Adaptive Regularized Training (ART):

This step involved training the network with increasing regularization to iteratively shrink weights close to zero, creating an inherently sparse network. The regularization strength increases over time, allowing weights to be explored before being pruned.

Magnitude Pruning and Fine-tuning:

Finally, we applied magnitude pruning to remove the smallest weights and fine-tuned the pruned network to achieve optimal performance.

DATASETS

CIFAR-10 and CIFAR-100 Datasets

Our experiments utilized two well-known datasets in the field of computer vision: CIFAR-10 and CIFAR-100. These datasets are widely used for benchmarking image classification algorithms and evaluating various neural network architectures.

CIFAR-10

consists of 60,000 32x32 color images in 10 classes, with 6,000 images per class. There are 50,000 training images and 10,000 test images. The 10 classes represent airplanes, automobiles, birds, cats, deer, dogs, frogs, horses, ships, and trucks.

CIFAR-100

is similar to CIFAR-10, except it has 100 classes containing 600 images each. There are 500 training images and 100 testing images per class. The 100 classes are grouped into 20 superclasses, with each superclass containing five classes. This dataset is more challenging than CIFAR-10 due to its larger number of classes and finer categorization.

MODELS

ResNet32

The ResNet32 model, introduced by He et al. (2016), is a variant of the ResNet (Residual Network) architecture designed for image classification tasks. ResNet32 is characterized by its use of residual blocks, which allow for the training of very deep networks by addressing the vanishing gradient problem through shortcut connections that skip one or more layers. This architecture enables the network to learn residual functions with reference to the layer inputs, rather than learning unreferenced functions.

In our experiments, ResNet32 was applied to the CIFAR-10 dataset. The CIFAR-10 dataset consists of 50,000 training and 10,000 validation images across 10 classes. We trained the ResNet32 model for 160 epochs on CIFAR-10 using the SGD optimizer with an initial learning rate of 0.1, decaying it by a factor of 0.1 at epochs corresponding to 2/4 and 3/4 of the total epochs. The batch size was set to 64, and the weight decay was set to 1e-4.

VGG19

The VGG19 model, proposed by Simonyan and Zisserman (2014), is another widely used deep convolutional neural network architecture. It consists of 19 layers, including 16 convolutional layers and 3 fully connected layers. VGG19 is known for its simplicity in design, using very small (3x3) convolution filters throughout the network. Despite its straightforward architecture, VGG19 achieves high performance on various image classification tasks.

In this study, VGG19 was evaluated on the CIFAR-100 dataset. We trained VGG19 on CIFAR-100 for 160 epochs using the SGD optimizer with an initial learning rate of 0.1 and a batch size of 64. The learning rate was decayed by a factor of 0.1 at 2/4 and 3/4 of the total epochs, and the weight decay was set to 1e-4.

5 Experimental Results

We evaluated our proposed HyperSparse regularization method against traditional L1 and L2 regularization methods using the ResNet32 and VGG19 models. The results are summarized below:

Model	Dataset	Regularization	Their Results	Our Results
ResNet32	CIFAR-10	HyperSparse	94.70%	94.17%
VGG19	CIFAR-100	HyperSparse	72.88%	73.45%
VGG19	CIFAR-100	L1	73.16%	73.08%
VGG19	CIFAR-100	L2	61.54%	70.90%

These results demonstrate the effectiveness of our HyperSparse regularization method in maintaining high performance while achieving significant sparsity in the neural network models.

Comparison with Other Methods

The performance of our proposed method, Adaptive Regularized Training (ART) combined with HyperSparse regularization, demonstrates significant improvements over traditional methods. Below are the key comparisons:

ResNet32 on CIFAR-10:

ART with HyperSparse: Achieved a test accuracy of 94.14%. Previous Methods: SNIP, Grasp, SRatio, LTH, RigL: Typically drop significantly in accuracy at high sparsity levels. L1 Regularization: Results in a test accuracy of 93.85%. L2 Regularization: Results in a test accuracy of 93.67%.

VGG19 on CIFAR-100:

ART with HyperSparse: Achieved a test accuracy of 73.26%. Previous Methods: SNIP, Grasp, SRatio, LTH, RigL: Performance drops significantly at high sparsity, sometimes down to random prediction levels.

L1 Regularization: Results in a test accuracy of 73.03%. L2 Regularization: Results in a test accuracy of 70.66%.

6 Discussion

The ART method provides a powerful tool for developing resource-efficient neural networks. By shifting the exploration to exploitation through adaptive regularization, ART manages to balance the trade-off between model sparsity and performance. Our replication study validates the robustness and efficiency of this approach, offering insights into its potential applications in various machine-learning tasks. there was a difference in the training results, however, we believe that it could be due to several things such as:

1. Random Initialization: Neural networks often start with random weights, which can lead to slight variations in results.

2. Data Preprocessing: Differences in how data is preprocessed can affect the outcomes.

3. Training Environment: Variations in hardware (like different GPUs) can impact training.

4. Stochastic Processes: Many algorithms involve stochastic processes that can lead to different outcomes each time you run the training.

7 Conclusion

In conclusion, our replication study successfully reproduced the results of Glandorf et al., demonstrating the efficacy of the ART method for achieving high sparsity in neural networks. The adaptive regularization approach proves to be a valuable technique for creating efficient and high-performing models, aligning with the goals of resource-efficient machine learning.

8 REFERENCE

SOURCE PAPER LINK

https://paperswithcode.com/paper/hypersparse-neural-networks-shifting

SOURCE CODE LINK

https://github.com/GreenAutoML4FAS/HyperSparse