2020 International Conference on Computational Performance Evaluation (ComPE), North-Eastern Hill University, Shillong, Meghalaya, India. July 2–4, 2020

An Incremental Pruning Strategy for Fast Training of CNN Models

Sangeeta Sarkar
Dept. of ECE
Gauhati University
Guwahati, India
sangsarkar123@gmail.com

Meenakshi Agarwalla
AI Division
Manvi R&D
Guwahati, India
meenakshi267@gmail.com

Sumit Agarwal
Dept. of EEE
IIT, Guwahati
Guwahati, India
scientist.sumit@gmail.com

Manash Pratim Sarma
Dept. of ECE
Gauhati University
Guwahati, India
mpsarma@gauhati.ac.in

***Abstract*—Deep Neural Networks have progressed significantly over the past few years and they are growing better and bigger each day. Thus, it becomes difficult to compute as well as store these over-parameterized networks. Pruning is a technique to reduce the parameter-count resulting in improved speed, reduced size and reduced computation power. In this paper, we have explored a new pruning strategy based on the technique of Incremental Pruning with less pre-training and achieved better accuracy in lesser computation time on MNIST, CIFAR-10 and CIFAR-100 datasets compared to previous related works with small decrease in compression rates. On MNIST, CIFAR-10 and CIFAR-100 datasets, the proposed technique prunes 10x faster than conventional models with similar accuracy.**

***Index Terms*—CNN, Pruning, Pre-training, Sparse Network, Compression**

I. Introduction

Deep neural network have developed gradually over past few years, and have become an important and ubiquitous tool in varied applications. These neural networks have varying sizes but one thing is common; they are large and over-parameterized. While these large neural networks have great potential, they need more memory and computation power. Thus, there lies a snag in their size as it consumes large storage, bandwidth and computational resources. In order to reduce the storage and computational cost, various techniques are used namely, pruning, quantization, weight sharing, etc. This paper proposes an improved pruning technique with better accuracy and much less training time as compared to the previous related works.

Pruning is a biologically inspired method of removing low ranking weights or filters resulting in a smaller, faster, more power efficient and more memory efficient network. These ranking of the neurons are done according to various regularization methods. Some of these regularization techniques include L1, L2, Dropout [1], Early Stopping, etc. Pruning the neural network leads to a sparser network. A sparse network has fewer links than maximum possible number of links within the network as illustrated in Fig. 1.

Pruning can be achieved in two ways. Firstly, One-shot pruning which involves training the model in consideration to a desirable accuracy followed by pruning in a single go. Secondly, Iterative Pruning involves initial training of the model followed by a small amount of gradually increased pruning and this process is repeated several times until desired pruning rate and accuracy is achieved. The disadvantage of one-shot pruning is that the network will not be able to classify correctly if pruned to a higher rate. The steps involved in One-shot pruning and Iterative Pruning are shown in Fig. 2(a) and Fig. 2(b) respectively.

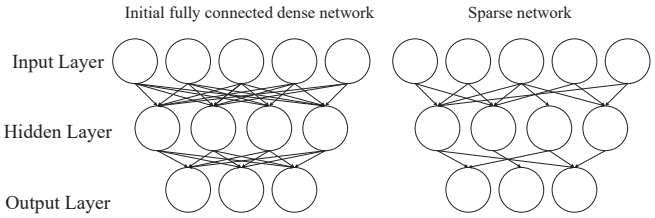


Fig. 1: A general sparse network from a fully connected network.

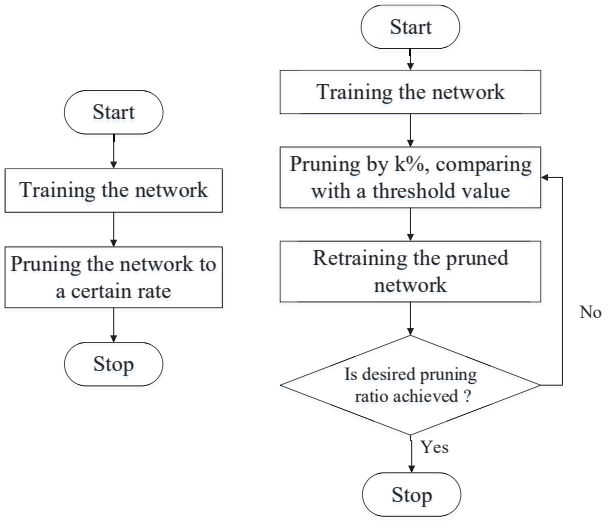
On the Modified National Institute of Standards and Technology (MNIST) dataset, our proposed method achieves an accuracy of 99.94%, 99.45% and 99.75% on pruned models proposed by Yann LeCun, like LeNet-300-100, LeNet-5 and Visual Geometry Group (VGG)-Lite respectively. On datasets from Canadian Institute For Advanced Research (CIFAR) like CIFAR-10 and CIFAR-100 datasets, our pruned VGG-Lite model achieves an accuracy of 93.71% and 99% respectively. On MNIST, CIFAR-10 and CIFAR-100 datasets, the proposed technique prunes 10x faster than conventional models.

This paper is divided into five sections. Section I introduces to the topic. Section II presents the theoretical background and Section III highlights the implementation steps. Finally, Section IV shows the results and Section V concludes the paper.

II. Related Work

The modern trend of training and testing a sparse CNN model involves pruning as the main ingredient. The objective of pruning is to obtain models with less number of parameters that can learn to classify objects at a faster rate with less memory requirement and computation power. Han [2] uses iterative pruning to obtain sparse neural networks. The pruning

978-1-7281-6644-5/20/$31.00 © 2020 IEEE



(a) One-shot Pruning (b) Iterative Pruning

Fig. 2: Different methods of pruning.

technique involves removing insignificant connections only in the fully-connected layers while keeping the weights of the convolutional layers constant. L2 regularization is used to achieve good accuracy along with dropout [1] to nullify the problem of overfitting. Molchanov [3] proposes Sparse Variational Dropout which is a modification of Variational Dropout [4]. In [4], the dropout rates can be set uniquely for each layer. On the other hand, [3] uses this technique along with L2-regularization, Batch Normalization [5] and Binary dropout achieving higher sparsity with similar accuracy on CIFAR-10 dataset and considerable loss on CIFAR-100 dataset. Narang [6] proposes a pruning technique where successively increasing values of threshold are set with each training iteration. This method enables to achieve good compression without any change in the loss. To circumvent the accuracy loss, in [7] Magnitude Pruning is proposed. A simple scheduler is used to determine the threshold value, thereby the low weighted connections are discarded achieving upto 10x reduction in weights without incurring much loss. The threshold value is increased gradually with the increase in number of pruning steps. Liu [8] proposes pruning entire convolutional filters along with its feature maps either through one-shot pruning or iterative pruning using L1-norm. At each pruning iteration, a certain number of low ranking filters are pruned globally among all the layers and reconsidered for retraining. [8] achieves a reduced computation cost and a maintained accuracy without the use of sparse convolutional complex libraries. Anwar [9] has done similar work to [8] but ranking is more complex. The network is trained, pruned and retrained but during the entire process the validation set is not considered. For further reduction, quantization technique is used. He [10] proposes Soft Filter Pruning (SFP) technique unlike the Hard Filter Pruning (HFP) adapted by [8],[11],[12],[13]. In HFP, the filters are pruned for each layers of pre-trained model, fine-tuned and retrained. These layer-wise pruning of filters degrades the performance of the model as the filters are not updated after pruning each layers filter. On the contrary, SFP keeps on updating the pruned filters during training and enables pruning the layers at the same time. SFP achieves better compression and accuracy. Li [14] suggested that for pruning algorithms, exhaustive pre-training is not required as mentioned in conventional pruning techniques. The method of Incremental Pruning Less Training (IPLT) is analogous to SFP[10]. IPLT gradually prunes the model in the pre-training phase followed by training until certain accuracy is achieved. This method reduces the computational time while maintaining the accuracy.

III. Proposed Work

This proposed model as shown in Fig. 3 has been adapted from the standard VGG-19 [14] model and VGG-like [3] model. This work is based on the methods of IPLT [14] with the difference in removing the filter weights rather than removing the entire filter. This is illustrated through an example as shown in Fig. 4 and Fig. 5 respectively. Fig. 4 shows the selection of filters and pruning in Incremental Pruning Less Training (IPLT)[14]. The first column shows the filters in the j^{th} layer. The second column shows the L2-norm values of the filter. The final column shows the pruned filters which are symbolised by dashed lines. On the other hand, Fig. 5 shows the selection of filters and pruning in our proposed work . The first column shows the filters in the j^{th} layer. The second column shows the L2-norm values of the j^{th} layer filter. The final column shows the j^{th} layer filters weights that are pruned. The weights of the filters that are pruned are kept empty. Also accuracy improvement techniques have been introduced in training phase itself. It is expected that better accuracy would be achieved in same time.

This pruning technique involves two major phases :

1) Minimum pre-training in pruning phase
2) Training phase

Fig. 6 shows the flow-chart of our proposed work.

A. Incremental Pruning Phase

The first phase of our training is the Incremental Pruning phase. It involves pre-training for a few epochs, pruning at an increased rate of 10%, then training for few epochs. The entire process is repeated until the desired pruning rate is achieved. In the pre-training stage, the least important pruning weights are chosen by regularization. Regularization impacts the pruning and retraining and prevents the model from overfitting. Overfitting problem has taken care as it degrades the model accuracy. In this approach, L2-regularization is used instead of L1-regularization because it is efficient and gives best pruning result. The weights of the filters with small L2-values are discarded. The pruning rates of the $(j+1)^{th}$ iteration can be calculated from (1).

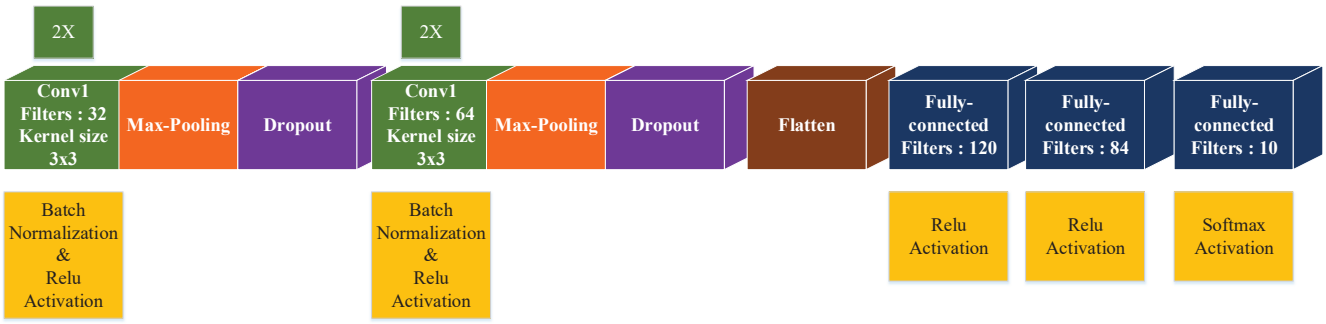


Fig. 3: VGG-Lite model.

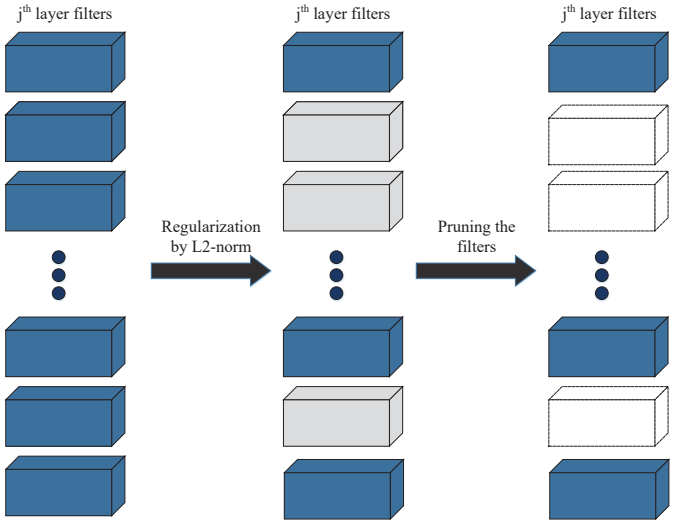


Fig. 4: Selection of filters and pruning in Incremental Pruning Less Training (IPLT)[14].

$$k_{j+1} = k_j + 10\% \, of \, N_j \tag{1}$$

where, k_j is the pruning rates of j^{th} iterations, k_{j+1} is the pruning rate of $(j+1)^{th}$ iterations, and N_j denotes the number of parameters of j^{th} iterations.

B. Training Phase

The second phase is the Training Phase. In pre-training phase, the weights are reduced. As a result, the performance of the network is hampered. Training phase helps this pruned network to recover. During the process of training, the weights are updated and the estimated error thus calculated, is governed by a hyper-parameter known as learning rate. It is very important to choose the learning rate wisely as higher or lower values of learning rate can respectively cause the model to converge too quickly to fallible solutions or cause the model to stuck-up. With each iteration of the training, the model progresses, getting closer to better accuracy. The model is trained until a desirable accuracy is achieved.

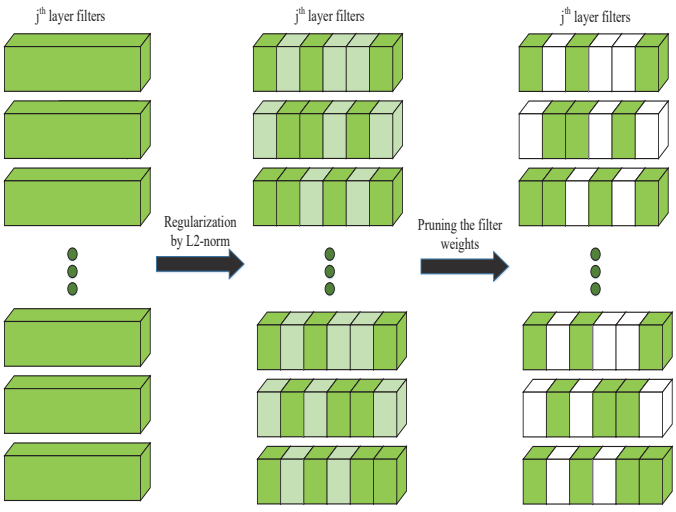


Fig. 5: Selection of filters and pruning in our proposed work

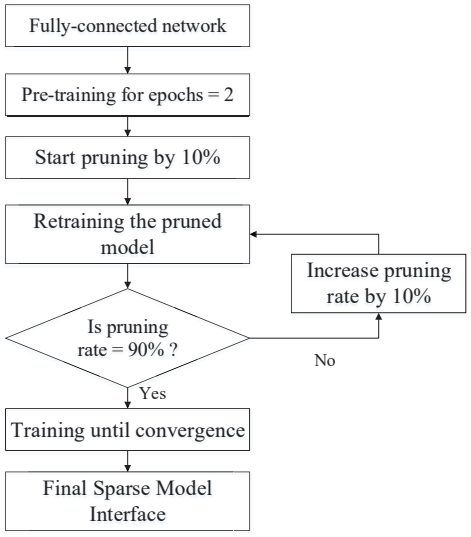


Fig. 6: The flow diagram of our proposed model.

IV. Experiments

This pruning technique is implemented using LeNet-300-100 and LeNet-5 on MNIST datasets and VGG-Lite on datasets CIFAR-10 and CIFAR-100 datasets.

A. On MNIST Dataset

MNIST dataset consists of 60,000 black and white images of size 28x28 with 10 classes. LeNet-300-100 consists of two fully-connected layers. Batch Normalization and Dropout are used after each fully-connected layer for LeNet-300-100, while for LeNet-5, Batch Normalization and static Dropout after each convolutional layer are used. L2-regularization and Adam optimizer with 10^{-5} decay, learning rate = 0.001 are used. The hyper-parameter is set to 2 and pruned from 10% to 90%.

TABLE I: Comparison of unpruned models trained on MNIST dataset

Model Name	Epochs	Accuracy (%)
LeNet-300-100 [2]	—	98.36
LeNet-300-100 (this work)	20	99.60
LeNet-5 [2]	—	99.2
LeNet-5 (this work)	20	99.76
Simple [14]	20	99.35
VGG-Lite (this work)	20	99.83

Table I presents a comparison of related works on MNIST dataset with our proposed unpruned model. Fig. 7 and Fig. 8 illustrate the training and testing accuracy and loss of MNIST on unpruned models rspectively. From Fig. 7, it is clearly visible that for 20 epochs, both training and testing accuracies are above 99%.

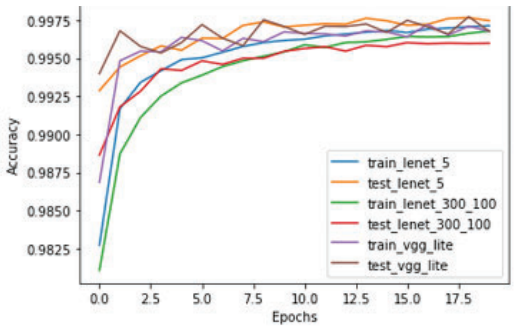


Fig. 7: The accuracy obtained on training and testing the unpruned model on MNIST datasets.

The results show that our models outperform from other models with less epochs. Table II shows that comparison of the pruned model achieving better accuracy results after pruning even with 20 epochs.

B. On CIFAR Datasets

CIFAR-10 and CIFAR-100 dataset consists of 60,000 and with 10 and 100 classes respectively of size 32x32. Both CIFAR-10 and CIFAR-100 are benchmark datasets and widely used so both of them are considered even though they only differ by their class size.

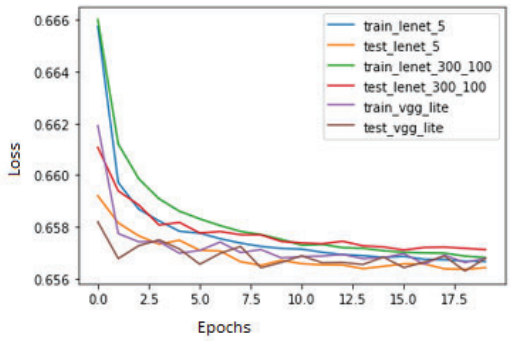


Fig. 8: The loss obtained on training and testing the unpruned model on MNIST datasets.

TABLE II: Comparison of pruned models trained on MNIST dataset

Model Name	Epochs	Accuracy (%)
LeNet-300-100 [2]	—	98.41
LeNet-300-100 [3]	200	99.08
LeNet-300-100 (this work)	20	99.94
LeNet-5 [2]	—	99.2
LeNet-5 [3]	200	99.25
LeNet-5 (this work)	20	99.45
Simple CNN [14]	20	99.35
VGG-Lite (this work)	20	99.75

TABLE III: Comparison of pruned models trained on CIFAR-10

Model Name	Epochs	Accuracy (%)
VGG-like [3]	200	92.70
VGG-19 [14]	40	94.20
VGG-Lite (this work)	20	93.71

We compare the accuracy on CIFAR dataset with different models like VGG-like and our proposed VGG-Lite model as shown in Table III. Both our unpruned and pruned model outperform other models in only 20 epochs. Table IV confirms the competency of our proposed model on CIFAR-100 dataset.

TABLE IV: Comparison of pruned models trained on CIFAR-100

Model Name	Epochs	Accuracy (%)
VGG-like [3]	200	92.70
VGG-Lite (this work)	20	99.00

TABLE V: Comparision of training time for obtaining a pruned model

Dataset	Model Name	Epochs		
		(This method)	[3]	[9]
MNIST	LeNet-300-100	20	200	—
	LeNet-5	20	200	—
	Simple CNN	20	—	—
CIFAR 10	VGG	20	200	40
CIFAR 100	VGG	20	200	—

The experiments are performed on Google Colab which uses Tesla K80 GPU. Table V shows comparison of training time for obtaining pruned models and shows that the training and pruning time in this method is much lower compared to other methods.

V. Conclusion

The pruning technique used in conventional method uses too many iterations. This work shows that large number of training and pruning steps are not necessary for obtaining a good pruned model. The incremental pruning technique proposed in this paper achieves better accuracy using much lower number of iterations. On the MNIST dataset, our proposed method achieves an accuracy of 99% and above on pruned models LeNet-300-100, LeNet-5 and VGG-Lite. On CIFAR-10 and CIFAR-100 datasets, our pruned VGG-Lite model achieve an accuracy of 93.71% and 99%. This is done with help of accuracy improvement methods integrated right into the pruning phase and pruning after limited training instead of complete training. The results of this paper would be useful in obtaining pruned models for sparsifying large and complex models in much lower amount of time. Our proposed work introduces a modified form of building pruned models which saves time, space and computation resources. It has proved to achieve better accuracy with much less training time and comparable compression ratio on some of the popular datasets like MNIST, CIFAR-10 and CIFAR-100. Also, our proposed VGG-lite model on these datasets has been able to come up with good results. These models can be applied on large datasets like ImageNet.

References

[1] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[2] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural network," in *Advances in neural information processing systems*, pp. 1135–1143, 2015.

[3] P. Molchanov, S. Tyree, T. Karras, T. Aila, and J. Kautz, "Pruning convolutional neural networks for resource efficient inference," *arXiv preprint arXiv:1611.06440*, 2016.

[4] D. P. Kingma, T. Salimans, and M. Welling, "Variational dropout and the local reparameterization trick," in *Advances in neural information processing systems*, pp. 2575–2583, 2015.

[5] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[6] S. Narang, E. Elsen, G. Diamos, and S. Sengupta, "Exploring sparsity in recurrent neural networks," *arXiv preprint arXiv:1704.05119*, 2017.

[7] M. Zhu and S. Gupta, "To prune, or not to prune: exploring the efficacy of pruning for model compression," *arXiv preprint arXiv:1710.01878*, 2017.

[8] Z. Liu, M. Sun, T. Zhou, G. Huang, and T. Darrell, "Rethinking the value of network pruning," *arXiv preprint arXiv:1810.05270*, 2018.

[9] S. Anwar, K. Hwang, and W. Sung, "Structured pruning of deep convolutional neural networks," *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, vol. 13, no. 3, pp. 1–18, 2017.

[10] Y. He, G. Kang, X. Dong, Y. Fu, and Y. Yang, "Soft filter pruning for accelerating deep convolutional neural networks," *arXiv preprint arXiv:1808.06866*, 2018.

[11] J.-H. Luo, J. Wu, and W. Lin, "Thinet: A filter level pruning method for deep neural network compression," in *Proceedings of the IEEE IEEE International Conference on Computer Vision*, pp. 5058–5066, 2017.

[12] Y. He, X. Zhang, and J. Sun, "Channel pruning for accelerating very deep neural networks," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1389–1397, 2017.

[13] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," *arXiv preprint arXiv:1510.00149*, 2015.

[14] L. Yue, Z. Weibin, and S. Lin, "Really should we pruning after model be totally trained? pruning based on a small amount of training," *arXiv preprint arXiv:1901.08455*, 2019.