Activation-Based Pruning of Neural Networks

Subjects: Computer Science, Artificial Intelligence

Contributor: Tushar Ganguli , Edwin K. P. Chong

A novel technique is presented for pruning called *activation-based* pruning to effectively prune fully connected feedforward neural networks for multi-object classification. The technique is based on the number of times each neuron is activated during model training. Further analysis demonstrated that activation-based pruning can be considered a dimensionality reduction technique, as it leads to a sparse low-rank matrix approximation for each hidden layer of the neural network. The rank-reduced neural network generated using activation-based pruning has better accuracy than a rank-reduced network using principal component analysis. After each successive pruning, the amount of reduction in the magnitude of singular values of each matrix representing the hidden layers of the network is equivalent to introducing the sum of singular values of the hidden layers as a regularization parameter to the objective function.

machine learning network pruning dimensionality reduction computer vision

1. Introduction

Deep neural networks are used to solve real-world problems in various domains such as image classification, text classification, and speech recognition. These networks often require millions of parameters and billions of floating-point operations to make accurate predictions. Network pruning has emerged as an important technique for improving the efficiency of deep neural networks by removing redundant structures. Pruning reduces the number of parameters of a neural network, resulting in a reduction of the computational resource required to run the network. Some of the most-popular pruning methods are magnitude-based pruning, structured pruning , pruning based on the lottery ticket hypothesis, and dynamic pruning. Of these methods, magnitude-based pruning has been proven to be successful for producing compact models and has witnessed widespread acceptance. However, prior work on magnitude-based pruning contains certain deficiencies. Magnitude-based pruning does not inherently induce a low-rank structure in the hidden layers of neural networks. More-rigorous constraints are needed to drive rank reduction. Integrating it with structured pruning or low-rank regularizers is likely necessary to fully exploit the compression property.

These limitations are addressed through activation-based pruning, which achieves results comparable to magnitude-based pruning in terms of training, validation, and testing accuracy, and functions as a dimensionality reduction technique. This method effectively reduces the hidden layers of neural networks to sparse low-rank matrix approximations. Activation-based pruning is achieved using both labeled and unlabeled data where the data should be a representative of the same distribution as the training data. Activation-based pruning is equivalent to introducing a weighted nuclear norm as a regularization parameter during the minimization of the objective function in image classification tasks. Consequently, activation-based pruning eliminated the need for additional regularizers to induce a low-rank structure in the hidden layers of the feedforward network. Activation-based pruning selectively targets weights that contributed minimally to the network's training process. This pruning strategy results in the formation of sparse, low-rank matrix approximations, effectively reducing the full-rank matrices of a trained network. After the pruned networks undergo retraining, they preserve the low-rank characteristics of these matrices, especially when previously pruned (zeroed) weights are allowed to be reutilized. In contrast, observations with magnitude-based pruning indicate a different behavior: During the retraining phase, this method engages nearly the entire spectrum of weights within the network. This distinction highlights the unique impact of activation-based pruning on the network's weight optimization during retraining.

2. Overview

2.1. Overview of Low-Rank Matrix Approximation

Various pruning methods [2][3][4] have resulted in sparse matrices, low-rank matrix approximation, or a hybrid of both. Han et al. [2] compressed deep neural networks by simultaneously pruning both network weights and connections to reduce computational cost and memory usage by inducing a regularization term that encourages sparsity during training. Weights with small magnitudes are pruned based on a specified threshold. This technique resulted in sparse weight matrices. Swaminathan et al. [3] proposed a novel method called sparse low rank (SLR) to compress the dense layers of deep neural networks by improving upon truncated singular-value decomposition (SVD). The key idea was to induce structured sparsity into the decomposed matrices from SVD based on the significance of the input/output neurons. Neuron significance is estimated by absolute weights, activations, or a change in cost when removed. Yang et al. [4] proposed a method called SVD training, which first decomposed each layer into the form of its full-rank SVD and, then, performs training on the decomposed weights. Low rank is encouraged by applying sparsity-inducing regularizers on the singular values of each layer. Singular-value pruning is applied at the end to explicitly reach a low-rank model. Activation-based pruning achieves sparsity in low-rank matrix approximation by assigning scores to each neuron to identify significant and insignificant neurons. The method avoids the computationally expensive process of decomposing matrices using SVD, making it advantageous for large matrices.

2.2. Overview of Structured/Unstructured Pruning

Pruning can be classified into structured [5][6][7][8][9][10][11][12], and semi-structured [2][13] pruning. Structured pruning removes filters, channels, or layers to induce structured sparsity patterns. It is commonly used in convolutional neural networks (CNNs), where entire filters or channels (groups of neurons) can be pruned. In unstructured pruning, individual weights without structural constraints are considered for removal. It can be applied to any layer of a neural network, including fully connected layers and convolutional layers. Semi-structured pruning is a hybrid approach that combines aspects of both structured and unstructured pruning. It involves removing entire structures, such as filters or channels, but within those structures, individual weights may be pruned. Activation-based pruning is unstructured, where individual neurons are assigned a score for pruning.

2.3. Overview of Importance-Based Pruning

Pruning can also be categorized based on the importance assigned to the weights, filters, and neurons of the network. These techniques induce sparsity by removing connections or filters based on criteria such as the weight magnitude [6][7][8] or sensitivity scores [14][15]. The sparse architectures are then retrained to regain accuracy. Zhu and Gupta [1] explored model pruning as a means of model compression by implementing magnitude-based pruning, where the weights with the smallest absolute values are pruned. In activation-based pruning, a score is assigned to each neuron, which guides the decision of pruning.

2.4. Overview Of Iterative/One-Shot Pruning

Pruning can also be categorized as either iterative [2][16][17][18] or one-shot [15]. Iterative pruning is a multi-step process of assigning a score, pruning the network, and retraining. Han et al. [2] proposed a three-step pipeline to prune redundant connections in neural networks without affecting accuracy. They first trained the dense network, then pruned low-weight connections below a threshold to obtain a sparse network, and finally, retrained the sparse network to learn the weight parameters. Guo et al. [16] introduced a two-step process called pruning and splicing, where weight connections can be removed and added back, based on solving a constrained optimization problem for each layer. Han et al. [17] used a three-pronged approach of pruning, quantization, and Huffman coding, to achieve substantial compression of the network. Yuan et al. [18] demonstrated how networks can be grown and pruned dynamically during the training phase using structured continuous sparsification. Growing and pruning of the network are often performed by introducing a regularization parameter in the cost function and relaxing the initial optimization problem. Liu et al. [15] implemented one-shot pruning to prune weights in a single step. Activation-based pruning is an iterative method where a score is assigned to each neuron, and subsequently, the neurons with the lowest score are pruned. Afterward, the network is retrained, and this cycle is repeated multiple times.

2.5. Overview of When to Prune

Pruning can also be classified based on when the pruning occurs, before [2][14][16][19], during [7][13][18][20], or after training [1][17]. The motivation for pruning before training is to eliminate the cost of pretraining. Pruning during training iteratively prunes and retrains the network to induce sparsity by updating the weight magnitudes or filters and channels, and pruning after training generally takes a pretrained network and, subsequently, prunes and retrains multiple times. Activation-based pruning occurs during training.

3. Conclusion

Activation-based pruning successfully reduces the size of feedforward networks. This pruning technique can be applied with either labeled or unlabeled data, as long as the data are drawn from the same distribution used to train the original feedforward network. Activation-based pruning is adaptable to supervised, semi-supervised, or unsupervised learning algorithms. Furthermore, each layer of the pruned network serves as a sparse low-rank matrix representation of the fully trained original network. Empirical evidence support the hypothesis that activation-based pruning can be interpreted as introducing a regularization parameter of the weighted nuclear norm of the hidden layers. Additionally, considering the architectural and implementation characteristics of activation-based pruning, this technique has the potential to be applied to various types of neural networks.

References

1. Zhu, M.; Gupta, S. To prune, or not to prune: Exploring the efficacy of pruning for model compression. arXiv 2017, arXiv:1710.01878.
2. Han, S.; Pool, J.; Tran, J.; Dally, W. Learning both Weights and Connections for Efficient Neural Network. In Advances in Neural Information Processing Systems; Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., Garnett, R., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2015; Volume 28.
3. Swaminathan, S.; Garg, D.; Kannan, R.; Andres, F. Sparse low rank factorization for deep neural network compression. Neurocomputing 2020, 398, 185–196.
4. Yang, H.; Tang, M.; Wen, W.; Yan, F.; Hu, D.; Li, A.; Li, H.; Chen, Y. Learning Low-Rank Deep Neural Networks via Singular Vector Orthogonality Regularization and Singular Value Sparsification. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, Virtually, 14–19 June 2020.
5. Hu, H.; Peng, R.; Tai, Y.W.; Tang, C.K. Network Trimming: A Data-Driven Neuron Pruning Approach towards Efficient Deep Architectures. arXiv 2016, arXiv:1607.03250.
6. Li, H.; Kadav, A.; Durdanovic, I.; Samet, H.; Graf, H.P. Pruning Filters for Efficient ConvNets. arXiv 2017, arXiv:1608.08710.
7. Frankle, J.; Carbin, M. The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks. In Proceedings of the International Conference on Learning Representations, New Orleans, LA, USA, 6–9 May 2019.
8. He, Y.; Zhang, X.; Sun, J. Channel Pruning for Accelerating Very Deep Neural Networks. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 1398–1406.
9. Gray, S.; Radford, A.; Kingma, D.P. GPU Kernels for Block-Sparse Weights. arXiv 2017, arXiv:1711.09224.
10. Kalchbrenner, N.; Elsen, E.; Simonyan, K.; Noury, S.; Casagrande, N.; Lockhart, E.; Stimberg, F.; van den Oord, A.; Dieleman, S.; Kavukcuoglu, K. Efficient Neural Audio Synthesis. In Proceedings

of the 35th International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018; Dy, J., Krause, A., Eds.; Proceedings of Machine Learning Research. Volume 80, pp. 2410–2419.

11. Frankle, J.; Dziugaite, G.K.; Roy, D.M.; Carbin, M. Stabilizing the Lottery Ticket Hypothesis. arXiv 2020, arXiv:1903.01611.
12. Yu, R.; Li, A.; Chen, C.F.; Lai, J.H.; Morariu, V.I.; Han, X.; Gao, M.; Lin, C.Y.; Davis, L.S. NISP: Pruning Networks Using Neuron Importance Score Propagation. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 9194–9203.
13. Molchanov, D.; Ashukha, A.; Vetrov, D. Variational Dropout Sparsifies Deep Neural Networks. In Proceedings of the 34th International Conference on Machine Learning, Sydney, NSW, Australia, 6–11 August 2017; Precup, D., Teh, Y.W., Eds.; Proceedings of Machine Learning Research. Volume 70, pp. 2498–2507.
14. Lee, N.; Ajanthan, T.; Torr, P. SNIP: Single-Shot Network Pruning Based on Connection Sensitivity. In Proceedings of the International Conference on Learning Representations, New Orleans, LA, USA, 6–9 May 2019.
15. Liu, Z.; Sun, M.; Zhou, T.; Huang, G.; Darrell, T. Rethinking the Value of Network Pruning. In Proceedings of the International Conference on Learning Representations, New Orleans, LA, USA, 6–9 May 2019.
16. Guo, Y.; Yao, A.; Chen, Y. Dynamic Network Surgery for Efficient DNNs. In Advances in Neural Information Processing Systems; Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., Garnett, R., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2016; Volume 29.
17. Han, S.; Mao, H.; Dally, W.J. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding. arXiv 2016, arXiv:1510.00149.
18. Yuan, X.; Savarese, P.; Maire, M. Growing Efficient Deep Networks by Structured Continuous Sparsification. arXiv 2020, arXiv:2007.15353.
19. Wang, C.; Zhang, G.; Grosse, R. Picking Winning Tickets Before Training by Preserving Gradient Flow. arXiv 2020, arXiv:2002.07376.
20. Tanaka, H.; Kunin, D.; Yamins, D.L.; Ganguli, S. Pruning neural networks without any data by iteratively conserving synaptic flow. In Advances in Neural Information Processing Systems; Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2020; Volume 33, pp. 6377–6389.