A Survey of Related Research on Compression and Acceleration of Deep Neural Networks

Xin Long\(^1\)*, Zongcheng Ben\(^2\), Yan Liu\(^3\)

\(^1\)National University of Defense Technology
Changsha, Hunan 410073

\(^2\)National University of Defense Technology
Changsha, Hunan 410073

\(^3\)National University of Defense Technology
Changsha, Hunan 410073

*longxin_gtkd@163.com  \(^b\)514827991@qq.com  \(^c\)641806177@qq.com

ABSTRACT: Deep networks have achieved great success in many areas in recent years. However, with the increasing sophistication of deep neural networks (DNNs), the memory consumption and computational cost expand exponentially, greatly hindering their application in mobile devices and other limited resources. Therefore, there is impending necessity to consider model compression and acceleration without affecting the inference accuracy. In this paper, we review the recent popular techniques for compressing and accelerating deep networks. Those methods could be broadly divided into four categories: parameter pruning and sharing, low rank approximation, sparse regularization constraints and network weight low-bit quantization. The advantages and disadvantages of different compression and acceleration methods are also described in detail, other types of approach are also introduced in our paper, and future prospects for the field are given finally.

1. INTRODUCTION

In recent years, the world has witnessed in a new round of artificial intelligence revolution, especially in deep learning, which has attracted attention of researchers from various countries. From handwritten character recognition to image recognition and speech recognition, the deep learning system has the ability to approximate the biological brain and can be applied to enhance simple tasks that humans or animals can perform. The rise of deep learning benefits from the success of deep learning networks, especially the development of deep convolutional neural networks (CNNs), providing advanced performance for many practical applications. Under the current background of big data, massive data provide a large number of training sets for DNNs. Complex calculation models also enhance the ability of data fitting. CNNs have achieved practical achievements that traditional methods cannot achieve.

All of these achievements are rely on complex deep neural network models. Hundreds of millions of parameters and high-performance computing GPUs play important roles. In the 2012, Krizhevsky [1] from Toronto university constructed a multi-layers network with over 60 million parameters which...
exceeds all previous models in its classification accuracy. But it took two to three days to train the entire network. DNNs are mainly composed of convolution layers, activation layers, pooling layers and full connection layers. With the complicate network structure, DNN will introduce more layers and neurons, while bringing the increase in the size of the model, the large memory requirements and the huge demand for energy consumption. Therefore, how to embed these performances into power-limited devices is a big challenge. We need to consider model compression and acceleration without impact of inference accuracy. Effective compression methods can have significant impacts on artificial intelligence-related distribution systems, embedded devices, or mobile devices.

There are many ways to solve this problem, including machine learning, computational structures, data compression, and hardware design. In this paper, we study the recent software methodologies of compressing and accelerating deep learning network models, and these methods have achieved certain achievements in some fields. In general, these methods can be roughly divided into following four categories: parameter pruning and sharing, low-rank approximation, sparse regularization, and weight quantization. The parameter pruning and sharing method mainly refers to find and delete the redundant parameters and connections in the network model, and keep the connection with the greater weight to achieve the purpose of reducing the storage and computation complexity. Low-rank approximation refers to decompose the high-dimensional weight tensors into the product of several low-order vectors and keep significant eigenvalues to maintain the accuracy of models. The sparse regularization method refers to the introduction of regularization constraints to the full-connect layer or convolutional layer of the network model to achieve sparseness, so as to reduce redundant convolution kernels and channels, and even convolutional layers. Weight quantification means that the weights of deep networks are represented by several discrete values, which could reduce the floating-point calculations in the actual computation process and the storage space. In the following sections we will describe each network compression and acceleration method in detail, as well as related research.

2. PRUNING AND SHARING

DNNs generally contain plenty of parametric redundancy [2], which will consume storage resources and computing devices, increasing of the power consumption in embedded devices. Previous research shows that pruning and parameter sharing have excellent performance in reducing model complexity and preventing network overfitting [3].

Initially, Han [4,5] proposed three-stage compression method (Fig.1) to prune the initial network structure to remove redundant or unimportant connections, then retrain sparsely connected networks obtained after pruning, and do parameter sharing (parameter sharing process refers to the use of k-means clustering algorithm, keep same parameter in one cluster). Finally, Huffman coding is used to maintain the accuracy while compressing the network. Experiments show that the above method achieves a better compression effect in Alexnet without losing the accuracy, the original 240 MB parameter amount is processed through above steps, which becomes 27 MB, 8.9 MB, and 6.9 MB, respectively.

However, the experimental network structure used for pruning in the above method is relatively small, which is generally applicable to the clipping of full-connection networks, and cannot achieve significant acceleration effect in large scale convolutional network layers [6]. With the increasing number of convolutional layers in a neural network, the compression of the convolutional layers is particularly important. For example, in [7], only 3.99% of the parameters of ResNet come from the fully connected layer. The others are parameters of the convolution layer. Based on this, Liu [8] used sparse decomposition to eliminate redundant parameters of the neural network model, successfully reduced the parameter amount by 90%, while the accuracy only lost about 1%.

In practice, this method has some drawbacks indeed. Pruning requires multiple iterations before it converges, which consumes lots of training time. Fine-tuning of parameters is also very tedious in the application, and increase the complexity of the calculation. Weight sharing would also reduce the training accuracy of the model in some extent.
3. LOW RANK APPROXIMATION

In DNNs, the computational cost is dominated by the convolution operation. Therefore, reducing the number of convolutional layers or compressing the convolutional layer means that the entire model is compressed and accelerated. In generally, any weight vector in the convolutional layer can be regarded as a four-dimensional tensor, and a large amount of redundant information existing in the four-dimensional tensors which make the low-rank approximation possible. For fully connected network, this method still works.

The idea of low-rank theory used for accelerating the convolution operation has already existed. For example, the high-dimensional discrete cosine transform and wavelet transform are represented by the product of one-dimensional discrete cosine and one-dimensional wavelet transform, respectively. In DNNs, two-dimensional tensors can be expressed in low rank by using the SVD (singular value decomposition) method, expressed as the product between orthogonal matrix with diagonal matrix. The higher-order tensor can be extended to two-dimensional tensor by stretching, and then low rank expressions could be achieved according to the above method. In [9], Denton et al. used this method to achieve a $2 \times$ speedup on both the GPU and the CPU, with a loss of accuracy of less than 1%.

Then Jaderberg [10] assumed that the original high-dimensional convolution kernel could be decomposed into the products of several low-dimensional kernel and proposed a solution to the problem of accelerating the DNNs. Firstly, convolution kernels with a rank of 1 are used to act on the input to generate M basic feature maps (Figure 1), and then the output feature maps are reconstructed by using linear combinations. This method could achieve the approximation of a 2-dimensional convolution kernel, which can also be extended to 4-dimensional. The acceleration effect of this method in text recognition reaches 4.5 times, while the accuracy only decreases by 1%.

Figure 1. The three-stage compression method proposed in [5].

Figure 2. Low rank approximation.
Low-rank decomposition is generally non-convex problem and difficult to calculate. Aim to obtain suitable solutions, Jaderberg [10] used iterative methods to obtain local solutions. In [11], Cheng proposed a new decomposition method based on the above work to find a global closed solution. Experiments show that the method can achieve equivalent or better results in large-scale networks.

Although low-rank approximation can achieve direct acceleration and compression in practice, there are some limitations. On the one hand, the decomposing operation of the convolution kernel takes time and effort. On the other hand, the current method is to perform low-rank compression at each layer, rather than global compression, resulting in different degree of deformation of each layer. In the end, this method requires continuous Fine-tuning and cross-validation to seek the balance between inference accuracy and training speed.

4. SPARSE REGULARIZATION

Except the above two methods, there are also sparse regularization methods that have attracted scholars' extensive research interest in recent years. This chapter will focus on it, which has been widely used in statistics, signal and image processing, computer vision and other fields. There is still a broad application space for deep network compression and acceleration.

4.1 Regularization Items

This method originates from the linear inverse problem:

\[ y = Ax + n \]  

Where \( y \) is the observed vector, \( x \) is the target vector needed to be acquired, \( A \) is the perceptual matrix or design matrix and \( n \) is the set to noise. Since the problem is ill-conditioned in most of the time, it needs to be solved by regularization methods. There are mainly the following three model methods [12]:

Methods of Tikhonov:

\[ \min f(x) + \tau \varphi(x) \]  

Methods of Morozov:

\[ \min \varphi(x) \quad s.t. \quad f(x) \leq \varepsilon \]  

Methods of Ivanov:

\[ \min f(x) \quad s.t. \quad \varphi(x) \leq \varepsilon \]  

Where \( \varphi(x) \) is the regularization constraints of the optimization function which is used to promote certain characteristics of the final solution, including sparsity, smooth, and so on. There are many kinds of function expressions for the regularization constraints. For example, the regularization term used the L2 norm, which have been proposed in [13] and [14]. This method is widely used in statistics because of its simple model and easy solution process.

The L1 norm gradually developed subsequently. Tibshirani [15] proposed the LASSO method in the field of statistics, Chen [16] proposed base trace in the field of signal processing, Tao [17] and Donoho [18] proposed the theory of compressed sensing [19]. The L1 norm has attracted a large number of scholars for its powerful ability to promote sparse solutions. In recent years, scholars have paid attention not only to the sparse property of the solution but also the structural characteristics of this sparse property, including group sparseness [20], block sparseness [21], and structure sparseness [22-25], etc. The experimental facts prove that these regularization constraints can find the solution of the problem better and faster, and obtain some structural features at the same time.

4.2 Sparsity regularization for DNNs

In recent several years, some scholars began to use the regularization constraints to sparse the DNNs. In the optimization process, the \( l_0 \) norm, \( l_1 \) norm, and \( l_2 \) norm regularization terms are mainly used to promote sparse structure, which is widely applied for image feature selection, facial recognition and multitasking learning and other fields [26-28].

Initially, Lebedev [29] used a variety of different pruning methods in the process of network learning for the brain injury problem. It was found that the computation cost of convolutional
computation is greatly reduced and the acceleration effect is significant when using the sparse regularization term.

It is not a unique instance, but has its counterpart. In [30], Zhou found that large networks have redundancy. To optimize the model structure, a method similar to [29] was adopted to introduce a regularization term into the objective function and use the norm:

$$g(w) = \lambda \sum_{(m,l) \in \mathcal{E}} \|w_{lm}\|$$

(5)

Experiments showed that for the networks of LeNet and AlexNet, the amount of neurons and parameters in the network were greatly reduced, while the accuracy was not lost.

Lately, Wen [6] added another special regularization term to each network layer and removed redundant convolution kernels, channels and even network layers successfully. After adding the regularization constraint, the objective function is expressed as:

$$E(W) = L(W) + \lambda R(W) + \eta \sum_{i=1}^{l} P(W^{(i)})$$

(6)

Where $L(W)$ is the term of data loss, $R(\cdot)$ is a non-structural sparse regularization term (such as $l_2$ norm), and $P(\cdot)$ is a structural sparse constraint term. Since Group Lasso has the feature of promoting group sparseness effectively [31, 32], it is quoted here:

$$P(W) = \sum_{g=1}^{G} \sqrt{\sum_{i=1}^{l} (w_{i}^{(g)})^2}$$

(7)

Where $G$ is the number of groups contained; $w_{i}^{(g)}$ is the weight in the group.

The introduction of regularization items on the target optimization function makes the network structure more compact and training efficiency greatly improved. However, there are still some disadvantages. The problem of using complex constraint needs constant iteration to reach the optimal solution.

5. WEIGHT QUANTIZATION

The DNN model is designed to serve practical applications such as scene monitoring, medical diagnostics, and unmanned vehicles. For these scenarios, there are strict constraints on physical size and energy consumption requirements. In order to facilitate the migration of DNNs on mobile devices (such as programmable logic arrays (FPGAs)), the relative research is weight quantization. In this way, the original expensive floating-point multiplication operation computation cost can be replaced by a fixed-point number shift operation. This not only reduces the memory requirements, but also accelerates the computing speed of the network and facilitates the operation on miniaturized platforms.

The low-bit quantization of neural networks can be traced back to the nineties of the nineteenth century [33,34], but there was no available deep network model and large data sets at the time to verify the effectiveness of it. In recent years, with the vigorous development of deep learning, weight quantization technology has re-entered our vision. A great deal of research has focused on the use of binary and ternary weights to train low-bit networks. In 2015, Prof. Yoshua took the lead in proposing the Binary Connect model [35], which attracted wide attention and opened the era of the binary network. In this method, the network weights in the forward propagation and the backward propagation are binarized into 1 or -1, while the weights are still kept in the floating point during the updating of parameters, which can save lots of matrix operations, and greatly optimize the training time and memory space. Experimentally proved that Binary Connect achieved excellent results on the MNIST, CIFAR-10, and SVHN, but for large datasets, it does not maintain good accuracy.

Then researches based on quantization have gradually increased. In [36], Rastegari innovatively proposed XNOR-Net's idea which approximates binarization of all weights and inputs simultaneously. If all operations in the convolution operation are binary, the point multiplication of two binary vectors is equivalent to a shift operation, which can greatly reduce the computational cost and memory savings. This is followed by the study of a three-valued network, including ternary weight networks (TWN) and trained ternary quantization (TTQ) [37-38]. Since the weighted quantization of the weights implies more information than the binarization, the inference accuracy is improved compared to the binary network.
Last year, Ali iDST team proposed the latest neural network algorithm based on low-bit network compression and acceleration technology [39]. This method is different from the binarization and the ternary network, but approximates the network weight as discrete value such as power-of-two form, and compresses the original multi-bit floating point weight into 1-3 bit weight. The final problem becomes a optimization problem with discrete constrained conditions, which can be solved by ADMM algorithm. Same as the above method, this technique replaces the original floating point multiplication operation with a fixed-point number shift operation, which improves the training speed and reduces energy consumption.

6. OTHER TYPES OF APPROACHES
There have been other attempts to reduce the number of parameters and model size of neural networks by designing special model architecture. Constraining the model architecture itself by reducing convolutional layers, removing fully connected layers, using convolutional filters of small size, such as NIN [40], Squeeze-Net [41] and Mobile-Net [42].

Other approaches to reduce the convolutional overheads contain above methods combinations. In [43] and [44], they firstly used special regularization to train the network and obtain a model with some structural characteristics , then retrain the net with parameter sharing by clustering algorithms. Experimental results showed that the amount of parameters are greatly reduced.

7. FUTURE OUTLOOK
Although considerable progress has been made in the research on the compression and acceleration of deep learning network models in recent years, it can be seen that this field is still in a preliminary development stage and faces many challenges.

(1) At present, most of the methods are based on mature deep neural network models, so the degree of freedom is small when changes are made to the network structure and hyperparameters. In order to deal with complicated and varied learning tasks, we need to find more robust methods.

(2) Layer compression causes different levels of deformation in each layer, which makes training difficult and leads to degradation in accuracy. Therefore, it is necessary to find an adaptive compression method based on the different situation.

(3) For low-rank approximation and sparse regularization method, the process of finding the optimal solution iteratively is complex, and it may be time-consuming to converge, requiring multiple iterations. It requires continuous fine-tuning and cross-validation to find the balance between accuracy and computational cost. The future requires more prior knowledge of the network to find the optimal solution with faster and more accurate way.

(4) Hardware constraints exist in most small platform devices. DNNs need to be implemented on smaller and more efficient platforms. Dedicated compression methods which make full use of limited computing resources are future research directions.

(5) Unified compression and accelerated evaluation methods. At present, the evaluation methods for compression and acceleration of various types of deep network models are scattered and there is no uniform measurement method. Subsequent researches needs to propose a unified evaluation criterion to apply for different models of different data sets.

8. CONCLUSION
For all DNNs, the computational cost and storage capacity are key factors that directly affect the learning performance of deep networks. The compression and acceleration of deep learning networks aims to reduce the redundancy of complex models and the computational cost. Its purpose is of great significance to the application of deep learning network miniaturization. Based on this, this paper summarizes the four main methods, including parameter pruning and sharing, low-rank approximation, regularization constraints, and weight quantization in the academic community in recent years, and analyzes its advantages and disadvantages. Finally, we make a simple outlook. We hope to help readers understand the field and expect more workers to join in.
REFERENCES

[1] A Krizhevsky, I Sutskever, and G Hinton, et al. Imagenet classification with deep convolutional neural networks. In Neural Information Processing Systems, 2012.

[2] Denil M, Shakibi B, Dinh L, et al. Predicting parameters in deep learning. In Neural Information Processing Systems, 2013: 2148-2156.

[3] Y Gong, L Liu, M. Yang, and L D Bourdev. Compressing deep convolutional networks using vector quantization, arXiv preprint arXiv: 1412. 6115, 2014.

[4] Han S, Pool J, Tran J, et al. “Learning both weights and connections for efficient neural networks,” Neural Information Processing Systems, 2015: 1135-1143.

[5] Han S, H Mao, and W J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding. In International Conference on Learning Representations (ICLR), 2016.

[6] W Wen, C Wu, Y Wang, Y Chen, and H Li. Learning structured sparsity in deep neural networks. In Advances in Neural Information Processing Systems, 2016.

[7] K M He, X Y Zhang, S Q Ren, and J Sun. Deep residual learning for image recognition, arXiv preprint arXiv: 1512. 03385, 2015.

[8] B Y Liu, M Wang, Hassan Foroosh, Marshall Tappen, and Marianna Pensky, et al. Sparse convolutional neural networks. In the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015.

[9] E Denton, W Zaremba, J Bruna, Y LeCun, and R Fergus. Exploiting linear structure within convolutional networks for efficient evaluation, arXiv preprint arXiv: 1404. 0736, 2014.

[10] Jaderberg M, Vedaldi A, Zisserman A. Speeding up convolutional neural networks with low rank expansions. In 26th British machine vision conference (BMVC). Swansea: IEEE, 2015, 100-109.

[11] Cheng T, Tong X, X G Wang, and Weiman E. Convolutional neural networks with low-rank regularization, arXiv preprint arXiv: 1511. 06067, 2015.

[12] D Lorenz and N Worliczek. Necessary conditions for variational regularization schemes. Inverse Problems, vol. 29, 2013.

[13] N Tikhonov. Solution of incorrectly formulated problems and the regularization method. In Soviet Math. Doklady, vol.4, pp. 1035–1038, 1963.

[14] E Hoerl and R W Kennard. Ridge Regression: Biased Estimation for Nonorthogonal Problems. In Technometrics, vol. 12, no. 1, 1970, pp. 55–67.

[15] R Tibshirani, Regression shrinkage and selection via the lasso. In Journal of the Royal Statistical Society, pp. 267-288, 1996.

[16] S S Chen, D L Donoho, and M A Saunders. Atomic decomposition by basis pursuit. In SIAM Journal on Scientific Computing, vol. 20, pp.33–61, 1998.

[17] E J Candes and T Tao. Decoding by linear programming. In IEEE Transactions on Information Theory, vol.51, pp. 4203-4215, 2005.

[18] D L Donoho. Compressed sensing. In IEEE Transactions on Information Theory, vol. 52, pp. 1289-1306, 2006.

[19] Shi G M, Liu D H, Gao D H, Liu J, Wang L J. Advances in Theory and Application of Compressed Sensing, ACTA ELECTRONICA SINICS, vol.37, pp. 1070-1081, 2009.

[20] M Yuan and Y Lin. Model selection and estimation in regression with grouped variables. In Journal of the Royal Statistical Society, vol. 68, pp. 49-67, 2005.

[21] Y Eldar and H Boleskei. Block-sparsity: Coherence and efficient recovery. In IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 2885-2888, 2009.

[22] C Micchelli, J Morales, and M Pontil. Regularizers for structured sparsity. In Advances in Computational Mathematics, pp. 1-35, 2010.

[23] J Huang, T Zhang, and D Metaxas. Learning with structured sparsity. In The Journal of Machine Learning Research, pp. 3371-3412, 2011.
[24] R Jenatton, J Y Audibert, and F Bach. Structured variable selection with sparsity-inducing norms. Journal of Machine Learning Research, vol.12, pp. 2777–2824, 2011.
[25] Liu J W, Cui L P, Liu Z Y, Luo X L. Survey on the Regularized Sparse Models, Chinese Journal of Computer, Vol.37, pp.1307-1325, 2015.
[26] A F T Martins , N A Smith, P M Q. Aguiar and M A T Figueiredo. Structured Sparsity in Structured Prediction. Proc. of Empirical Methods for Natural Language Processing, 2011.
[27] R Caruana, Multitask learning. Machine Learning, vol.28, pp.41-75, 1997.
[28] G Obozinski, B Taskar and M. Jordan. Joint covariate selection and joint subspace selection for multiple classification problems. Statistics and Computing, vol.20, pp. 231-252, 2010.
[29] V Lebedev and V S Lempitsky. Fast convnets using group-wise brain damage. In 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016.
[30] H Zhou, J M Alvarez, and F Porikli, Less is more: Towards compact cnns. In European Conference on Computer Vision, Amsterdam, the Netherlands, October 2016, pp. 662–677.
[31] M Yuan and Y Lin. Model selection and estimation in regression with grouped variables. Journal of the Royal Statistical Society, 68(1): 49–67, 2006.
[32] Seyoung Kim and Eric P Xing. Tree-guided group lasso for multi-task regression with structured sparsity. In Proceedings of the 27th International Conference on Machine Learning, 2010.
[33] E Fiesler, A Choudry, and H J Caulfield. Weight discretization paradigm for optical neural networks. In The Hague’90, 12-16 April. International Society for Optics and Photonics, 1990, pp. 164–173.
[34] M Marchesi, G Orlandi, F Piazza, and A Uncini. Fast neural networks without multipliers. IEEE transactions on Neural Networks, vol. 4, no. 1, pp. 53–62, 1993.
[35] M Courbariaux, Y Bengio, and J P David. Binary connect: Training deep neural networks with binary weights during propagations. Advances in Neural Information Processing Systems, 2015.
[36] M Rastegari, V Ordonez, J Redmon, and Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. European Conference on Computer Vision (ECCV), 2016.
[37] F Li and B Liu. Ternary Weight Networks. In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, June 27-30, 2016.
[38] C Z Zhu, S Han, H Z Mao, and William J. Trained Ternary Quantization, arXiv preprint arXiv: 1612. 01064, 2017.
[39] C Leng, Z S Dou, S H Zhu, and R Jin. Extremely low bit neural: squeeze the last bit out with ADMM, arXiv preprint arXiv: 1707. 09870, 2017.
[40] Min Lin, Qiang Chen, Shuicheng Yan. Network in network. arXiv preprint arXiv:1602.07360, 2016.
[41] Forrest N. Iandola, Song Han, Matthew W. Moskewicz, Khalid Ashraf, William J. Dally, Kurt Keutzer. SqueezeNet: Alexnet-level accuracy with 50x fewer parameters and < 0.5MB model size. arXiv preprint arXiv:1602.07360, 2016.
[42] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. MobileNets: Efficient convolutional neural networks for mobile vision applications, arXiv preprint arXiv:1704.04861, 2017.
[43] Wu J, Wang Y, Wu Z, et al. Deep k-Means: Re-Training and Parameter Sharing with Harder Cluster Assignments for Compressing Deep Convolutions. arXiv preprint arXiv:1806.09228, 2018.
[44] Fard M M, Thonet T, Gaussier E. Deep k-Means: Jointly Clustering with k-Means and Learning Representations. arXiv preprint arXiv:1806.10069, 2018.