Sparse Optimization Based on Non-convex $\ell_{1/2}$ Regularization for Deep Neural Networks

Anda Tang[1], Rongrong Ma[1], Jianyu Miao[2], and Lingfeng Niu[3](✉)

[1] School of Mathematical Sciences, University of Chinese Academy of Sciences, Beijing 100190, China
{tanganda17,marongrong16}@mails.ucas.ac.cn

[2] Henan University of Technology, Zhengzhou 450001, China
jymiao@haut.edu.cn

[3] School of Economics and Management, University of Chinese Academy of Sciences, Beijing 100190, China
niulf@ucas.ac.cn

Abstract. With the arrival of big data and the improvement of computer hardware performance, deep neural networks (DNNs) have achieved unprecedented success in many fields. Though deep neural network has good expressive ability, its large model parameters which bring a great burden on storage and calculation is still a problem remain to be solved. This problem hinders the development and application of DNNs, so it is worthy of compressing the model to reduce the complexity of the deep neural network. Sparsing neural networks is one of the methods to effectively reduce complexity which can improve efficiency and generalizability. To compress model, we use regularization method to sparse the weights of deep neural network. Considering that non-convex penalty terms often perform well in regularization, we choose non-convex regularizer to remove redundant weights, while avoiding weakening the expressive ability by not removing neurons. We borrow the strength of stochastic methods to solve the structural risk minimization problem. Experiments show that the regularization term features prominently in sparsity and the stochastic algorithm performs well.

Keywords: Deep neural networks · Sparsity · Non-convex regularizer

1 Introduction

Deep neural networks (DNN) have achieved unprecedented performance in a number of fields such as speech recognition [1], computer vision [2], and natural language processing [22]. However, these works heavily rely on DNN with a huge number of parameters, and high computation capability [5]. For instance, the work by Krizhevsky et al. [2] achieved dramatic results in the 2012 ImageNet Large Scale Visual Recognition challenge (ILSVRC) using a network containing

60 million parameters. A convolutional neural network, VGG [27], which wins the ILSVRC 2014 consists of 15M neurons and 144M parameters. This challenge makes the deployment of DNNs impractical on devices with limited memory storage and computing power. Moreover, a large number of parameters tend to decrease the generalization of the model [4,5]. There is thus a growing interest in reducing the complexity of DNNs.

Existing work on model compression and acceleration in DNN can be categorized into four types: parameter pruning and sparsity regularizers, low-rank factorization, transferred/compact convolutional filter and knowledge distillation. Among these techniques, one class focuses on promoting sparsity in DNNs. DNNs contain lots of redundant weights, occupying unnecessary computational resources while potentially causing overfitting and poor generalization. The network sparsity has been shown effective in network complexity reduction and addressing the overfitting problem [24,25].

Sparsity for DNNs can be further classified into pruning and sharing, matrix designing and factorization, randomly reducing the complexity and sparse optimization. The pruning and sharing method is to remove redundant, non-information weights with a negligible drop of accuracy. However, pruning standards require manual setup for layers, which demands fine-tuning of the parameters and could be cumbersome for some applications.

The second methods reduce memory costs by structural matrix. However, structural constraint might bring bias to the model. On the other hand, how to find a proper structural matrix is difficult. Matrix factorization uses low-rank filters to accelerate convolution. The low-rank approximation was done layer by layer. However, the implementation is computationally expensive and cannot perform global parameter compression.

The third methods randomly reduce the size of network during training. A typical method is dropout which randomly removes the hidden neurons in the DNNs. These methods can reduce overfitting efficiently but take more time for training.

Recently, training compact CNNs with sparsity constraints have achieved more attention. Those sparsity constraints are typically introduced in the optimization problem as structured and sparse regularizers for network weights. In the work [26], sparse updates such as the ℓ_1 regularizer, the shrinkage operator and the projection to ℓ_0 balls applied to each layer during training. Nevertheless, these methods often results in heavy accuracy loss. Group sparsity and the ℓ_1 norm are integrated in the work. [3,4] to obtain a sparse network with less parameters. Group sparsity and exclusive sparsity are combined as a regularization term in a recent work [5]. Experiments show that these method can achieve better performance than original network.

The key challenge of sparse optimization is the design of regularization terms. ℓ_0 regularizer is the most intuitive form of sparse regularizers. However, minimizing ℓ_0 problem is NP-hard [15]. The ℓ_1 regularizer is a convex relaxation of ℓ_0, which is popular and easy for solving. Although ℓ_1 enjoys several good properties, it may cause bias in estimation [8]. [8] proposes a smoothly clipped absolute (SCAD) penalty function to ameliorate ℓ_1, which has been proven to be unbiased. Later, many other nonconvex regularizers are proposed, including the minimax concave penalty (MCP) [16], ℓ_p penalty with $p \in (0,1)$ [9–13], ℓ_{1-2} [17,18] and transformed ℓ_1(TL1) [19–21].

The optimization methods play a central role in DNNs. Training such networks with sparse regularizers is a problem of minimizing a high-dimensional non-convex and non-smooth objective function, and is often solved by simple first-order methods such as stochastic gradient descent. Consider that the proximal gradient method is a efficient method for non-smooth programming and suitable for our model, we choose this algorithm and borrow the strengths of stochastic methods, such as their fast convergence rates and ability to avoid overfitting and appropriate for high-dimensional models.

In this paper, we consider non-convex regularizers to sparsify the network weights so that the non-essential ones are zeroed out with minimal loss of performance. We choose a simple regularization term instead of multiple terms regularizer to sparse weights and combine dropout to remove neurons.

2 Related Work

2.1 Sparsity for DNNs

There are two methodologies to make networks sparse. A class focuses on inducing sparsity among connections [3,4] to reinforce competitiveness of features. The ℓ_1 regularizer is applied as a part of regularization term to remove redundant connections. An extension of $\ell_{1,2}$ norm adopted in [5] not only achieve the same effect but also balance the sparsity of per groups.

Another class focuses on the sparsity at the neuron level. Group sparsity is a typical one [3,4,6,7], which is designed to promote all the variables in a group to be zero. In DNNs, when each group is set to denote all weights from one neuron, all outgoing weights from a neuron are either simultaneously zero. Group sparsity can automatically decide how many neurons to use at each layer, force the network to have a redundant representation and prevent the co-adaptation of features.

2.2 Non-convex Sparse Regularizers

Fan and Li [8] have discussed about a good penalty function that it should result in an estimator with three properties: sparsity, unbiasedness and continuity. And regularization terms with these properties should be nonconvex. The smoothly clipped absolute deviation (SCAD) [8] and minimax concave penalty (MCP) [16] are the regularizers that fulfil these properties. Recent years, nonconvex metrics in concise forms are taken into consideration, such as ℓ_{1-2} [17,18,31] and transformed ℓ_1 (TL1) [19–21] and ℓ_p $p \in (0,1)$ [9–13,32].

3 The Proposed Approach

We aim to obtain a sparse network, while the test accuracy has comparable or even better result than the original model. The objective function can be defined by

$$\underset{W}{min}\ \mathcal{L}(f(W), D) + \lambda \Omega(f) \tag{1}$$

where f is the prediction function which is parameterized by W and $D = \{x_i, y_i\}_{i=1}^{N}$ is a training set which has N instances, and $x_i \in \mathbb{R}^p$ is a p-dimensional input sample and $y_i \in \{1, ..., K\}$ is its corresponding class label. $\mathcal{L}$ is the loss function and Ω is the regularizer. λ is the parameter which balances the loss and the regularization term. In DNNs, W represents the set of weight matrices. As for the regularization term, it can be written as the sum of regularization on weight matrix for each layer.

ℓ_p regularization ($0 < p < 1$) is studied in the work [9–13]. The ℓ_p quasi norm of $\mathbb{R}^N$ for a variable x is defined by

$$\|x\|_p = \sum_{i=1}^{N}(|x_i|^p)^{\frac{1}{p}} \tag{2}$$

which is nonconvex, nonsoomth and non-Lipschitz.

And we use an extension of $\ell_{1,2}$ called exclusive sparse regularization to promote competition for features between different weights, making them suitable for disjoint feature sets. The exclusive regularization of $\mathbb{R}^{n\times m}$ is defined by

$$EL(X) = \sum_{i=1}^{n}(\sum_{j=1}^{m}(|x_{ij}|))^2 \tag{3}$$

The ℓ_1 norm reaches the sparsity within the group, and the ℓ_2 norm reaches the balance weight between the groups. The sparsity of each group is relatively average, and the number of non-zero weights of each group is similar.

In this work, we define the regularizer as follows,

$$\Omega(W) = (1 - \mu)\sum_{g}(\sum_{i}|w_{g,i}|)^2 + \mu \|Vec(W)\|_{\frac{1}{2}}^{\frac{1}{2}} \tag{4}$$

where $Vec(W)$ denotes vectorizing the weight matrix.

4 The Optimization Algorithm

4.1 Combined Exclusive and Half Thresholding Method

In our work, we use proximal gradient method and combine the stochastic method to solve the regularized loss function, a solution in closed form can be obtained for each iteration in our model. Considering that a minimization problem

$$\min_{W} \mathcal{L}(f(W), D) + (1-\mu)\sum_{l=1}^{4}(\sum_{i}|w_{l,i}|)^2 + \mu\lambda\sum_{l=1}^{4}\left\|Vec(W^l)\right\|_{\frac{1}{2}}^{\frac{1}{2}} \tag{5}$$

When updating W_t, as the regularizer consists of two terms, we first compute an intermediate solution by taking a gradient step using the gradient computed on the loss only, and then optimize for the regularization term while performing Euclidean projection of it to the solution space. Select a batch of samples $D_i, d_i \in D_i\}$

$$W_t = W_t - \frac{\eta_t}{size(D_i)}\sum_{i=1}^{size(D_i)}\nabla\mathcal{L}(f(W_t), d_i) \tag{6}$$

then do proximal mapping of weights after current iteration, compute the optimization problem as follows,

$$\begin{aligned} W_{t+\frac{1}{2}} &= prox_{(1-\mu)EL}(W_t) \\ &= \underset{W}{argmin}\,\frac{1}{2\lambda\eta}\left\|W - W_t\right\|_2^2 + \Omega(W) \end{aligned} \tag{7}$$

One of the attractive points of the proximal methods for our problem is that the subproblem can often be computed in closed form and the solution is usually shrinkage operator which can bring sparsity to models. The proximal operator for the exclusive sparsity regularizer, $prox_E L(W)$, is obtained as follows:

$$\begin{aligned} prox_{(1-\mu)EL}(W) &= (1 - \frac{\lambda(1-\mu)\,||W_l|\,|_1}{|w_{l,i}|})_+\, w_{l,i} \\ &= sign(w_{l,i})(|w_{g,i}| - \lambda(1-\mu)\,||W_l|\,|_1)_+ \end{aligned} \tag{8}$$

Now we consider how to compute the proximal operator of half regularization

$$\begin{aligned} W_{t+1} &= prox_{\mu\ell_{1/2}}(W_{t+\frac{1}{2}}) \\ &= \underset{W}{argmin}\,\frac{1}{2\lambda\eta}\left\|W - W_{t+\frac{1}{2}}\right\|_2^2 + \Omega(W) \end{aligned} \tag{9}$$

The combined regularizer can be optimized by using the two proximal operators at each gradient step, after updating the variable with gradient. The process is described in Algorithm 1. When training process terminates, some weights turn to zero. Then, these connections will be removed. Ultimately, a sparse architecture is yielded.

5 Experiments

5.1 Basilines

We compare our proposed method with several relevant baselines:

Algorithm 1: Combined Exclusive and Half Thresholding Regularization Method for DNN

```
Input :
        initial learning rate $\eta_0$ , initial weight $W_0$ ,
        regularization parameter $\lambda$ , balancing parameter $\mu_l$
        for each layer, mini batch size $n$, training dataset $D$ ;
output:
        the solution $w^*$
1  repeat
2  until some stopping criterion is satisfied;
3  $n$ samples randomly selected from $D$ ;
4  for layer $l$ do
5      for $\{x_i, y_i\}$ in the samples selected do
6          $L_i^{(l)} := \nabla\mathcal{L}(w_{t-1}^{(l)}, \{x_i, y_i\})$
7      end
8      $W_t^{(l)} := W_{t-1}^{(l)} - \eta_t \sum_{i=1}^{n} \frac{L_i^{(l)}}{n}$;
9      $W_t := prox_{(1-\mu)\lambda\eta_t EL}(W_t)$;
10     $W_t := prox_{\lambda\eta_t\mu\ell_{1/2}}(W_t)$;
11 end
12 $t := t + 1$
```

- ℓ_1
- Sparse Group Lasso (SGL) [4]
- Combined Group and Exclusive Sparsity (CGES) [5]
- Combined Group and TL1 Sparsity (IGTL) [28]

5.2 Network Setup

We use Tensorflow framework to implement and evaluate our models. In all cases, we choose the ReLU activation function for the network,

$$\sigma(x) = max(0, x) \tag{10}$$

One-hot encoding is used to encode different classes. We apply softmax function as activation for the output layer defined by

$$\rho(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}} \tag{11}$$

where x denotes the vector that is input to softmax, the $i_t h$ denotes the index. We initialize the weights of the network by random initialization according to a normal distribution. The size of the batch is depending on the dimensionality of the problem. We optimize the loss function by standar cross-entropy loss, which is defined as

$$\mathcal{L} = -\sum_{i=1}^{n} y_i log(f(x_i)) \tag{12}$$

5.3 Measurement

We use accuracy to measure the performance of the model, floating-point operations per second (FLOPs) to represent the computational complexity reduction of the model and parameter used to represent the percentage of parameters in the network compare to the fully connected network. The results for our experiments are reported in Table 1.

Table 1. Performance of each model on mnist

Measure	ℓ_1	SGL	CGES	IGTL	El$\ell_{1/2}$
Accuracy	0.9749	0.9882	0.9769	0.9732	**0.9772**
FLOPs	0.6859	0.8134	0.6633	0.1741	**0.1485**
Parameter used	0.2851	0.4982	0.2032	0.1601	**0.1513**

6 Conclusion

We combine exclusive sparsity regularization term and half quasi-norm and use dropout to remove neurons. We apply $\ell_{1/2}$ regularization to the neural network framework. At the same time, the sparseness brought by the regularization term and the characteristics suitable for large-scale problems are fully utilized; We also combine the stochastic method with the optimization algorithm, and transform the problem into a half thresholding problem by proximal method, so that the corresponding sparse problem can be easily solved and the complexity of the solution is reduced.

References

1. Hinton, G., et al.: Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. IEEE Sig. Process. Mag. **29**(6), 82–97 (2012)
2. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, pp. 1097–1105 (2012)
3. Alvarez, J.M., Salzmann, M.: Learning the number of neurons in deep networks. In: Advances in Neural Information Processing Systems, pp. 2270–2278 (2016)
4. Scardapane, S., Comminiello, D., Hussian, A.: Group sparse regularization for deep neural networks. Neurocomputing **241**, 81–89 (2017)
5. Yoon, J., Hwang, S.J.: Combined group and exclusive sparsity for deep neural networks. In: International Conference on Machine Learning, pp. 3958–3966 (2017)
6. Zhou, H., Alvarez, J.M., Porikli, F.: Less is more: towards compact CNNs. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9908, pp. 662–677. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46493-0_40

7. Lebedev, L., Lempitsky, V.: Fast convnets using group-wise brain damage. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 2554–2564 (2016)
8. Fan, J., Li, R.: Variable selectionvia nonconcave penalized likelihood and its oracle properties. J. Am. Stat. Assoc. **96**(456), 1348–1360 (2001)
9. Xu, Z.: Data modeling: Visual psychology approach and $L_{1/2}$ regularization theory (2010). https://doi.org/10.1142/9789814324359_0184
10. Xu, Z., Chang, X., Xu, F., Zhang, H.: $\ell_{1/2}$ regularization: a thresholding representation theory and a fast solver. IEEE Trans. Neural Netw. Learn. Syst. **23**(7), 1013 (2012)
11. Krishnan, D., Fergus, R.: Fast image deconvolution using hyper-laplacian priors. In: International Conference on Neural Information Processing Systems, pp. 1033–1041 (2009)
12. Xu, Z., Guo, H., Wang, Y., Zhang, H.: Representative of $L_{1/2}$ regularization among $L_q(0<q \leq 1)$ Regularizations: an experimental study based on phase diagram. Acta Autom. Sinica **38**(7), 1225–1228 (2012)
13. Chartrand, R., Yin, W.: Iterative reweighted algorithms for compressive sensing. Technical report (2008)
14. Lv, J., Fan, Y.: A unified approach to model selection and sparse recovery using regularized least squares. Ann. Stat. **37**(6A), 3498–3528 (2009)
15. Natarajan, B.K.: Sparse approximate solutions to linear systems. SIAM J. Comput. **24**(2), 227–234 (1995)
16. Zhang, C.H., et al.: Nearly unbiased variable selection under minimax concave penalty. Ann. Stat. **38**(2), 849–942 (2010)
17. Esser, E., Lou, Y., Xin, J.: A method for finding structure sparse solutions to nonnegative least squares problem with applications. SIAM J. Imaging Sci. **6**(4), 2010–2046 (2013)
18. Yin, P., Esser, E., Xin, J.: Ratio ad difference of ell_1 and ℓ_2 norms and sparse representation with coherent dictionaries. Commun. Inform. Syst. **14**(2), 87–109 (2014)
19. Nikolova, M.: Local strong homogeneity of a regularized estimator. SIAM J. Appl. Math. **61**(2), 633–659 (2000)
20. Zhang, S., Xin, J.: Minimization of transformed ℓ_1 penalty: Closed form representation and iterative thresholding algorithms. arXiv preprint arXiv:1412.5240 (2014)
21. Zhang, S., Xin, J.: Minimization of transformed ℓ_1 penalty: theory, differnece of convex function algorithm, and robust application in compressed sensing. Math. Program. **169**(1), 307–336 (2018)
22. Dauphin, Y.N., Fan, A., Auli, M., Grangier, D.: Language modeling with gated convolutional networks. arXiv preprint (2016). arXiv:1612.08083
23. Gong, Y., Liu, L., Yang, M., Bourdev, L.D.: Compressing deep convolutional networks using vector quantization, CoRR, vol. abs/1412.6115 (2014)
24. Gong, Y., Liu, L., Yang, M., et al.: Compressing deep convolutional networks using vector quantization. arXiv preprint arXiv:1412.6115 (2014)
25. Dinh, T., Xin, J.: Convergence of a Relaxed Variable Splitting Method for Learning Sparse Neural Networks via ℓ_0, ℓ_1 and transformed ℓ_1 Penalties arXiv preprint arXiv:1812.05719 (2018)
26. Collins, M.D., Kohli, P.: Memory bounded deep convolutional networks, arXiv preprint arXiv:1412.1442
27. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
28. Ma, R., Miao, J., Niu, L., et al.: Transformed ℓ_1 regularization for learning sparse deep neural networks. Neural Netw. **119**, 286–298 (2019)

29. Robbins, H., Monro, S.: A stochastic approximation method. Ann. Math. Stat. **22**(3), 400–407 (1951)
30. Bottou, L., Curtis, F.E., Nocedal, J.: Optimization methods for large-scale machine learning (2016)
31. Shi, Y., Miao, J., Wang, Z., et al.: Feature selection with ℓ_2, ℓ_{1-2} regularization. IEEE Trans. Neural Netw. Learn. Syst. **29**(10), 4967–4982 (2018)
32. Niu, L., Zhou, R., Tian, Y., et al.: Nonsmooth penalized clustering via ℓ_p regularized sparse regression. IEEE Trans. Cybern. **47**(6), 1423 (2017)
33. Shi, Y., Lei, M., Yang, H., et al.: Diffusion network embedding. Pattern Recognit. **88**, 518–531 (2019)