Research on Model Compression Based on Convolutional Neural Network

Minglan Xu

Cardiff Sixth Form College, Cardiff, UK

*Corresponding Author

Keywords: Convolutional neural network, Deep learning, Benchmark architecture vgg16, Data set mnist

Abstract: Deep learning methods have achieved remarkable success in the variety of applications with various variants. The most popular instance is perhaps Convolutional neural networks(CNN) consisting numerous of numbers of convolutional layers to proceed image based input to yield desired output. Typically, CNNs contains enormous number of parameters and requires huge number of float operations for inference. Hence how to filter out redundant parameters become more and more necessary. In this paper, we study how to compress CNN architectures based on sparsity-inducing regularization optimization. We validate the method on one benchmark architecture VGG16 and dataset MNIST.

1. Introduction

This paper investigates the exploration process of compressing models with different methods of CNNs. A Convolutional Neural Network CNN is constituted of one or more convolutional layers and then followed by one or more fully connected layers as in a standard multilayer neural network, as our environment is filled with neural networks and deep learning models, thus, CNNs become one of the most popular ways analyzing visual imagery. However, there are still a few common problems of CNNs; for example, pooling function can't always work though it is very popular in CNN because the outcomes keep approximately constant when small changes appear [1].

In the last few decades, Convolutional Neural Networks CNN have demonstrated rapid developments and significant successes in many areas such as video and image recognition, recommender systems[2], image classification, medical image analysis, natural language processing[3] and so on. Therefore, among various application, handwritten digit recognition on MNIST is one of them which base on CNN. On one hand, talking about CNNs' advantages, it is necessary to know that CNN has numerous parameters so that it is able to find optimized solutions with a extreme fast speed. Obviously, it benefits from most of the advantages but suffers from high computational cost. On the other hand, there are a lot of excessive parameters so it would be interesting to filter out those redundant parameters.

Therefore, what can we do, aiming to overcome those difficulties above? Sparsity-inducing optimization can be a perfect way to solve these questions, this method is aiming to use or obtain some easy and typical datas or models. It is used for linear changeable selection but nowadays innumerable extensions emerge, like structured sparsity or kernel selection. Many relative estimation problems can be project as convex optimization problems by regulating empiric risk with flexible non-smooth norms. To be more specific, it can make the models be expressed easily or computationally cheaper to use (ie. even if the model does not need to space, it can still look for the perfect sparse approximation) In addition, it also gives signal in advance whether the model needs to be spares or not. Thus, sparsity-inducing optimization absolutely can be utilized to compress models.

The structure of the paper is organized as follows: in section II, we present some related work of the model compression optimization and demonstrate further information about two datasets models. Section III I will show some concrete steps about the method on one benchmark architecture

  DOI: 10.25236/cseem.2020.026

VGG16 and dataset MNIST then give out the summary before explaining more further research about potential directions in the future.

2. Related Work

In the recent years, the model compression has been received special attentions with numerous methods designed for CNNs, which can be largely categorized into the following branches:

Bayesian compression: Researches may construct some statistical models to predict the compression ratio of each convolutional neural networks.

Bayesian compression is based on Bayes' theorem which describes the probability of an event which depends on some of former conditions related to the event. [4] We trim a large part of networks through sparsity inducing priors, there are mainly two novelties which are hierarchical priors and posterior uncertainties. [9]

Filter ranking by importance score: Another idea is to rank the importance score of each filters and remove the filters that exhibits less fundamental to the model's predictive powers.

Tucker Decomposition in [8] computes a higher n-D Tensor along its each modes/dimensions. The first 2 modes are the output and input when the convolution layer is a 4D Tensor, and also, the rest 2 dimensions are special in CNNs. Due to the 4-D Tensor in Tucker Decomposition, there is a set of 2D-matrics U along each of the dimensions of the tensor and a core tensor G. It is able to vary the ranks of the output core tensor and factor metrics so that a trade-off betweenez space and accuracy can be happened [7][8].

$$K_{i,j,s,t} = \sum_{r_1=1}^{R_1} \sum_{r_2=1}^{R_2} \sum_{r_3=1}^{R_3} \sum_{r_4=1}^{R_4} G_{r_1,r_2,r_3,r_4} \times U^1_{i,r_1} \times U^2_{j,r_2} \times U^3_{s,r_3} \times U^4_{t,r_4}$$

Filter Pruning from convolutional layers can be a perfect and common approach of compressing CNNs [6]. It depends on its importance to decide filters which can be deleted through a few methods. We followed the skills in [6] and minimize those filters by the Taylor series expansion of the error reduces by removing a filter to yield better conclusion. Finally it provides a balanced number of filters between space and accuracy though the tradeoff of threshold parameters [7].

Knowledge distillation: The knowledge distillation is designed to train a huge teacher network which would further used to train a much smaller student networks with the same level of accuracy.

In [11], the author believes that the model can be regarded as a black box, and knowledge can be regarded as a mapping relationship from input to output. Therefore, we can first train a teacher network, and then use the output result of the teacher's network q as the target of the student network, and train the student network so that the result of the student network pred is close to q.

3. Method and Experiments

We develop a model compression method based on sparse optimization. In general, the sparse optimization, e.g., l1-regulization, is able to yield solutions of high sparsity, i.e., including zero elements. The sparsity of one kernel can be interpreted as the redundancy of the current kernel, so that we may use it as some compression ratio to shrink the whole kernel into a smaller size. To deliver such high sparse solutions, there exist numerous methods, e.g., proximal stochastic gradient descent method Prox-SG and its variants. In the remaining paper, we present how we utilize Prox-SG to conduct model compression on CNN and benchmark datasets.

3.1 Experimental Environment

We performed our experiments on environment PyTorch, importing with torch and torchvision. In our experiments, we will mainly describe the processes to concretely validate the method on one benchmark architecture VGG16 and dataset MNIST.

3.2 Datasets

MNIST[12] is a benchmark dataset, which contains 70,000 images totally where 10,000 for testing and 60,000 for training, In their original paper, a support-vector machine was used to get an error rate of 0.8%.[16]

3.3 Experimental Setup

We mainly used PyTorch to train for the MNIST dataset, firstly create a PyTorch environment followed by defining hyperparameters for optimizer, i.e., Prox-SG. As the experiments are repeatable, we have to set random seeds for anything applying stochastic number generation; next TorchVision actually takes part in using a batch size of 1000 for testing and 64 for training with a result of global mean and standard deviation produced in this dataset. It is much easier to analyze and recognize images after some trainings for the PyTorch dataloaders. While building a network such as VGG16, we use two 2-D convolutional layers followed by two fully-connected layers to create a ideal new class. Rectified linear units and two dropout layers are chosen for the activation function and the mean of regularization respectively then import these prepared codes. For ensuring our network has in its training mode, it is necessary to iterate over it per epoch. We propagate a new set of gradients into the parameters after calculating a negative log likelihood to measure the deviation between the output and the ground truth label.

In the end, we loop over number of epochs to evaluate our model with random initialized parameters [12][13]. That's one of the examples how CNN Architectures can be compressed based on sparsity-inducing regularization optimization.

In the whole process we train the VGG16 on MNIST by the optimizer Prox-SG, and reach the accuracy of 99% with high sparse solution about 70%, which is further used to construct a smaller model of which the size is only about 70% compared to the original one. Then we retrain this compressed model and observe no regression on accuracy which demonstrate the effectiveness of the sparse optimization on model compression.

4. Conclusion

We propose a model compression approach by sparse optimization, where the sparse optimization is able to yield high sparse solution and used as compression ratio for each kernel in the convolutional neural network. In numerical experiments, we successfully compress the VGG16 by about 70% on benchmark dataset MNIST.

References

[1] Zhang, Xingpeng , and X. Zhang . "Global Learnable Pooling With Enhancing Distinctive Feature for Image Classification." IEEE Access, vol.8, pp.98539-98547, 2020.

[2] M. Slaney. "Web-scale multimedia analysis: Does content matter?" MultiMedia, IEEE, vol.18, no.2, pp.12-15, 2011.

[3] Collobert, Ronan , and J. Weston . "A unified architecture for natural language processing: Deep neural networks with multitask learning." Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5-9, 2008.

[4] LAZalta, Edward N. "The Stanford Encyclopedia of Philosophy." Philpapers Org, vol.1. no.1, pp.69-76, 2013.

[5] Douglas Hubbard "How to Measure Anything: Finding the Value of Intangibles in Business" pg. 46, John Wiley & Sons, 2007

[6] Molchanov P , Tyree S , Karras T , et al. Pruning Convolutional Neural Networks for Resource Efficient Inference. 2016..

[7] Goyal, Saurabh , A. R. Choudhury , and V. Sharma . “Compression of Deep Neural Networks by Combining Pruning and Low Rank Decomposition.” 2019 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW) IEEE, 2019..

[8] Kim Y D , Park E , Yoo S , et al. Compression of Deep Convolutional Neural Networks for Fast and Low Power Mobile Applications. Computer ence, vol.71, no.2 pp.576-584, 2015.

[9] Christos Louizos, Karen Ullrich, Max Welling, Bayesian Compression for Deep Learning. NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing SystemsDecember 2017, pp.3290–3300, 2017.

[10] D. P. Kingma, T. Salimans, and M. Welling. Variational dropout and the local reparametrization trick. Advances in Neural Information Processing Systems, 2015.

[11] Geoffrey Hinton, Oriol Vinyals, Jeff Dean, Distilling the Knowledge in a Neural Network arXiv:1503.02531

[12] Chouaib H , Cloppet F , Vincent N . Fast Feature Selection for Handwritten Digit Recognition, International Conference on Frontiers in Handwriting Recognition. IEEE Computer Society, 2012..

[13] Küttler, Heinrich, Nardelli N , Lavril T , et al. TorchBeast: A PyTorch Platform for Distributed RL. 2019..

[14] K. Simonyan and A. Zisserman from the University of Oxford, Very Deep Convolutional Networks for Large-Scale Image Recognition.

[15] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton, collected CIFAR-10 and CIfar-100 datasets, 2009.

[16] LeCun, Yann, Léon Bottou, Yoshua Bengio and Patrick Haffner, gradient based learning applied to document recognition.