Survey Paper on Optimizing Inference in Deep Neural Networks

Kruthi P R[1], Kousar Mulla[2], Madhabi Choudhury[3], Prof. Madhavi R P[4]

[1,2,3] *U.G Students,* [4]*Associate Professor, Computer Science and Engineering Department, BMS College of Engineering, Bengaluru, Karnataka, India*

***Abstract*—Deep neural networks (DNN) are an artificial neural network that imitates the working of the human brain in processing data and use it for decision-making. Nowadays, DNNs have achieved a great success in various image classification and recognition tasks. However, one of the main issues with deep neural networks is its huge computational cost and storage overhead, which constitute a serious challenge for small devices, exceeding the computing limit of those devices. During the past few years, tremendous progresses have been made in this area. In this paper we survey the recent advanced techniques for optimizing inference in DNNs through various techniques**

***Keywords* — Deep Neural Networks, Training, Inference, Optimization, Pruning, Compression, Lossy, Lossless**

I. INTRODUCTION

Deep neural networks (DNN) is an artificial neural network that imitates the working of the human brain in processing data and use it for decision-making. Training is the phase in which the network tries to learn from the given data. Inference can only happen after training. Inference is when the network uses the trained knowledge. Inference takes real-world data can only happen after training. Inference is when the network uses the trained knowledge. Inference takes real-world data and comes back with a prediction. This inference to happen quickly and also accurate becomes very important.

Training is computationally very expensive and requires terabytes of training data which cannot be done in small computational devices. So, optimizing inference becomes very important. The importance of this phase is to minimize the scope of the model by removing any parts not necessary or combine two or more layers into a single computational step. One of the main issues with deep neural networks is its huge computational cost and storage overhead, which constitute a serious challenge for small devices, exceeding the computing limit of those devices.

Thus, network compression and optimization has become a major concern and a topic of research in both academics and industries.

Therefore, to avoid all these above-mentioned issues and to optimize deep neural networks, we will be implementing the Lossy method of Filter/Channel pruning, where the network is optimized with some loss of accuracy, using Tensor Flow for optimizing popular Image nets: AlexNet, ResNet, Google Inception and Vgg.

II. MOTIVATION

Many industries (Nvidia TensorRT, Intel OpenVino) are also working and doing intense research on this optimization techniques to improve DNN performance. NVidia TensorRT successfully showed 5 ways of optimizing the network and accelerating deep learning inference.

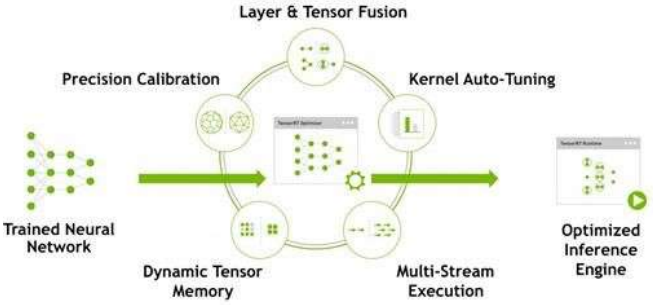

Fig 1. Courtesy [10]

III. TECHNIQUES

A. Pruning Techniques:

NNs are computationally intensive and memory intensive, making them difficult to deploy on embedded systems. Also, conventional networks fix the architecture before training starts; therefore, training cannot improve the architecture. To address these limitations, this paper has proposed a method which prunes redundant connections in 3 steps.1) Train the network to learn which connections are important.2) Prune the unimportant connections.3) Retrain the pruned network to fine tune the weights of the remaining connections. This may be repeated iteratively. [3]

Another framework namely ThiNet, which simultaneously accelerates, and compresses CNN models, establishes filter pruning (removing the whole unimportant filter) as an optimization problem and reveal that we need to prune filters using statistics information computed from its next layer, not the current layer, which differentiates ThiNet from existing

methods. Main advantage is, it does not change the network structure and higher accuracy will be preserved. [1]

One more idea is structured pruning by sparsification to help dissolve away the dense connectivity that often found at different levels in convolutional neural explored and it was found that intra-kernel striped sparsity along with convolution lowering can reduce the computational complexity. The quantize networks. The structured sparsity in deep CNN has been pruned network helps in the reduction of storage space and thus makes way for non-chip memory-based implementation of DNN. [4]

A new channel pruning method to accelerate very deep CNNs. Given a trained CNN model, there is an iterative 2 step algorithm to effectively prune each layer, by a LASSO regression-based channel selection and least square reconstruction. [8]

The success of CNNs is accompanied by a significant increase in the computation and parameter storage costs. To avoid this there is a method to prune filters with relatively low weight magnitudes to produce CNNs with reduced com- putation costs without introducing irregular sparsity. Instead of pruning with specific layer-wise hyper parameters and time consuming iterative retraining, it uses one-shot pruning and retraining strategy for simplicity an ease of implementation. [9]

B. Compression Techniques:

Existing deep CNN models are computationally expensive and memory intensive, hindering their deployment in devices with low memory resources or strict latency requirements. A solution is to perform model compression and acceleration in deep networks without decreasing the model performance. This technique is categorized into 4 schemes.1) parameter pruning and sharing 2) Lowrank factorization 3) Transferred/compact convolutional filters 4) Knowledge distillation (Training a compact neural network with distilled knowledge of large model). [2]

Most of the existing deep neural networks are structurally very complex, making them difficult to be deployed on the mobile platforms with limited computational power. To overcome this there is a compression method called dynamic network surgery, which can remarkably reduce the network complexity by properly incorporating network splicing into the whole process to avoid incorrect pruning and making it as a continual network maintenance. These two operations pruning, and splicing are integrated together by updating parameter importance whenever necessary by making our method dynamic. [5] Yet there are 2 approximations to standard convolutional neural networks: Binary Weight Networks and XNOR - Networks. 1)In the first method, they train the network that learns to find binary values for weights, which reduces the size of network by ~32x and provide the possibility of loading very DNNs into portable devices with limited memory. 2) second method, uses mostly bitwise operations to approximate convolutions.[6]

Main barriers for implementing CNNs on embedded systems are their large model size and large number of operations needed for inference. To surpass these obstacles, all the approaches can be divided into 3 groups: precision reduction, network pruning and design of compact network architectures. After presenting the main approaches in each group they conclude that the future CNN compression algorithms should be co-designed with Algorithms with hardware which will process deep learning algorithms. [7]

Studying these papers and understanding them, we can categorize optimization of DNN in 2 ways:

1) Lossless Techniques

Where the network is optimized with no loss of accuracy

a) Data Format Optimization (NHWC): Data formats refers to the structure of the image passed to a given layer. The parts of the 4D image are often referred to by the following letters:

- N refers to the number of images in a batch.
- H refers to the number of pixels in the vertical (height) dimension.
- W refers to the number of pixels in the horizontal (width) dimension.
- C refers to the channels. For example, 1 for grayscale and 3 for RGB.

Research have shown that permutation of these 4 components can improve performance. [10]

b) Layer Fusion (horizontal and vertical): First, layers with unused output are eliminated to avoid unnecessary computation. Next, where possible convolution, bias, and ReLU layers are fused to form a single layer. [10]

c) Dropout (Layer Removal): The term "dropout" refers to dropping out units (both hidden and visible) in a neural network. Dropout is a technique for reducing overfitting in neural networks by preventing complex co-adaptations on training data. This is exactly how mammalian brain works. It is a very efficient way of performing model averaging with neural networks.[12]

2) Lossy Techniques

a) Filter/Channel Pruning: In this method we remove some percentage of layer say, 10% of the layer randomly and check for the output. If the accuracy is good and the time taken is faster, the removal was successful. Else, we try the next permutation and combination of layers. [13]

IV. CONCLUSIONS

This paper presents survey for different techniques for optimizing deep neural networks. These tech- niques help in having an improved and optimized DNN and the time taken in inference would be an improved result compared to the current image networks with some negligible error, and we will be implementing pruning method which is a lossy technique using TensorFlow.

ACKNOWLEDGMENT

The work, reported in this paper, is supported by the college through the Technical Education Quality Improvement Program [TEQIP-III] of the MHRD, Government of India

REFERENCES

[1]. ThiNet: A Filter Level Pruning Method for Deep Neural Network Compression, Jian- Hao Luo, Jiaxin Wu, Weiyao Lin, International Conference on Computer Vision
[2]. A Survey of Model Compression and Acceleration for Deep Neural Networks, Yu Cheng, Duo Wang, Pan Zhou, Tao Zhang
[3]. Learning both Weights and Connections for Efficient Neural Networks, Song Han, Jeff
[4]. Structured Pruning of Deep Convolutional Neural Networks, Sajid Anwar, Kyuyeon
[5]. Dynamic Network Surgery for Efficient DNNs, Yiwen Guo, Anbang Yao, Yurong Chen, Neural Information Processing Systems, 2015
[6]. XNOR-Net: Imagenet Classification Using Binary Convolution Neural Networks, Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi, European Conference on Computer Vision, 2016
[7]. Compression of Convolutional Neural Networks, Ratko Pilipovic, Patricio Bulic, Vladimir Rissojevic
[8]. Channel Pruning for Accelerating Very Deep Neural Networks, Yihui He, Xiangyu Zhang, Sun, International Conference on Computer Vision 2017
[9]. Pruning Filters for Efficient ConvNets, Hao Li, AsimKadav, Igor Durdanovic, Hanan Samet, Hans Peter Graf, International
[10]. Conference on Learning Representations, 2017,
[11]. http://developer.nvidia.com
[12]. https://en.wikipedia.org/wiki/Dropout_(neural_networks)
[13]. https://jacobgil.github.io/deeplearning/pruning-deep-learning