Spectral Tensor Layer for Model-Parallel Deep Neural Networks

Zhiyuan Wang[1], Ziyi Xia[2], Xinghang Sun[2], Xiao-Yang Liu[2]

[1]Huazhong University of Science and Technology, Wuhan, China.
[2]Columbia University, New York, US.

Introduction

We propose a novel *spectral tensor layer* for model-parallel deep neural networks, which can be trained in the spectrum domain [1][2]. A spectral tensor neural network consists of a number of small, weak subnetworks, which can be trained in an asynchronous parallel manner. The final output is the ensemble of the subnetworks' outputs, e.g., average over all subnetworks, or weighted average of top-k subnetworks. Compared with conventional neural networks, spectral tensor neural networks have intrinsical data parallelism as well as model parallelism, which is very suitable for distributed training, and the ensemble method performs remarkably well.

Spectral Tensor Neural Network

Data Preprocessing: Assume that the original input has size $H \times W$ and we split it into B independent spectrals as follows:

- **Data tensor**: Reorganize the data into a tensor of size $H' \times W' \times B$, where $H'W'B = HW$.
- **Spectrum tensor**: Perform discrete cosine transform (DCT) along the third dimension of the data tensor to obtain the spectrum tensor with B spectrals (frontal slices). Each frontal slice is a spectral of size $H' \times W'$.

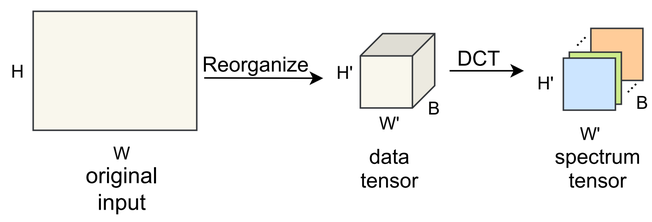

Fig. 1. Data preprocessing.

Training Process: Train B subnetworks independently with the B spectrals and the corresponding labels, respectively, using standard packages.

Ensemble Method: In inference stage, we split a sample into B spectrals and feed them into the B subnetworks correspondingly. As shown in Fig. 3, we ensemble the B outputs using weighted averaging:

$$\boldsymbol{y} = \sum_{i=1}^{B} w_i \boldsymbol{y_i}, \tag{1}$$

where $\boldsymbol{y}$ is the output of the spectral tensor neural network, $\boldsymbol{y_i}$ and w_i are the component subnetworks' output and corresponding weight, $i = 1, 2, ..., B$.

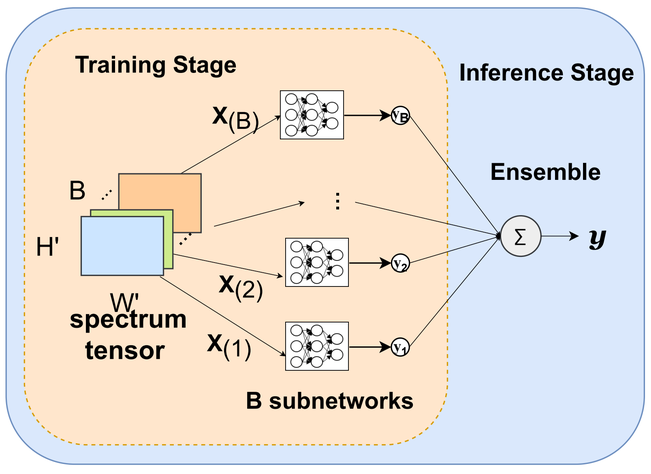

Fig. 2. Training stage and inference stage of spectral tensor neural networks.

Spectral tensor layer brings following advantages:

- **Data parallelism and Model parallelism**: The B spectrals contain the information of different frequencies, making it reasonable to separate the spectrals and independently train B subnetworks for the B spectrals, respectively.
- **Distributed training**: The B separatable spectrals can be placed at B nodes with the corresponding subnetwork.
- **Asynchronous parallelism**: There is no need to mix the intermediate information at different frequencies. Therefore, there is no communication overhead among the parallel subnetworks during the training process.

MNIST Data Set

Data set: MNIST handwritten digit classification data set [3]. Each image has size 28×28.
Preprocessing: Each image is preprocessed into a spectrum tensor of size $7 \times 7 \times 16$. The spectrum tensor is split into 16 spectrals where each spectral is of size 7×7.
Experiment Settings:

- Independently train 16 subnetworks with 8 fully connected layers, of which the number of neurons in each hidden layer is 49.
- Optimizer is Adam and learning rate is 1×10^{-3}.
- Set the batch size as 128.

Results: As given in Table 1, we measure the performance by computing the classification accuracy on the test data set. As shown in Fig. 3, we record the loss of ensemble case and subnetworks.

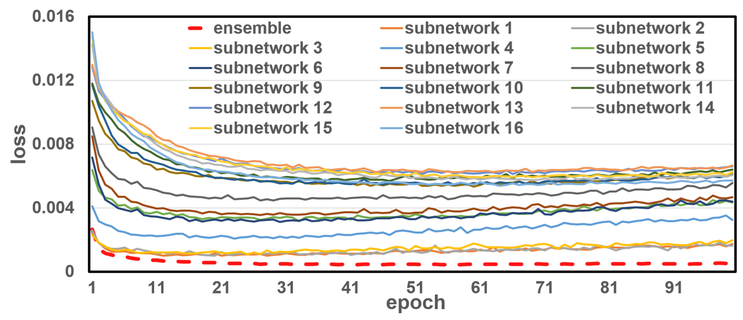

Fig. 3. Test loss on MNIST data set.

ImageNet Data Set

Data set: ImageNet data set (ILSVRC2012) [4]. Each image has size $224 \times 224 \times 3$.
Preprocessing: Each image is preprocessed into a spectrum tensor of size $56 \times 56 \times 16 \times 3$. The spectrum tensor is split into 16 spectrals where each spectral is of size $56 \times 56 \times 3$.
Experiment Settings:

- Independently train 16 subnetworks with 12 convolution layers.
- Optimizer is Adam, the initial learning rate is 0.1 and we adjust the learning rate periodically.
- Set the batch size as 1024.

Results: As given in Table 1, we got an accuracy of 61.24% in ensemble method, which is remarkably better than 56.82%, the highest accuracy of the subnetworks.

CIFAR-10 Data Set

Data set: CIFAR-10 classification data set [5]. Each image has size $32 \times 32 \times 3$.
Preprocessing: Each image is preprocessed into a spectrum tensor of size $16 \times 16 \times 4 \times 3$. The spectrum tensor is split into 4 spectrals where each spectral is of size $16 \times 16 \times 3$.
Experiment Settings:

- Independently train 4 subnetworks with 10 convolution layers.
- Optimizer is Adam and learning rate is 1×10^{-3}.
- Set the batch size as 128.

Results: As given in Table 1, we measure the performance by computing the classification accuracy on the test data set. And as shown in Fig. 4, we record the loss of ensemble case and subnetworks.

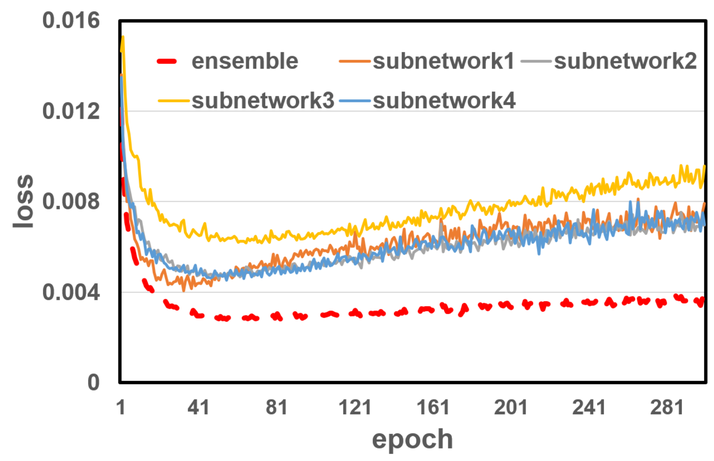

Fig. 4. Test loss on CIFAR-10 data set.

Experiment Results

We summarize the mean and highest accuracy of the component subnetworks and the accuracy of our ensemble method in Table 1, namely *Mean Acc.*, *Highest Acc.* and *Acc. Ens.*, respectively. *Baseline* is the accuracy of conventional neural networks. The conventional neural networks have the same layer as the component subnetworks but have more parameters because it deals with input of larger size.

Table 1. Experiment results.

Data sets	Mean Acc.	Highest Acc.	Ens. Acc.	Baseline
MNIST [3]	82.74%	96.32%	98.36%	98.71%
CIFAR-10 [5]	81.67%	85.54%	89.64%	91.86%
ImageNet [4]	45.48%	56.82%	61.24%	62.62%

Conclusions

In this poster, we propose spectral tensor layer for model-parallel deep neural networks. By performing DCT on the data, we can split the data to achieve data parallelism as well as model parallelism. The great potential for distributed training without communication makes spectral tensor layer for neural networks very promising. Based on that, we've shown the ensemble result is remarkably better than the best result of the component subnetworks.

References

[1] Tao Zhang, Xiao-Yang Liu, Xiaodong Wang, and Anwar Walid. cuTensor-Tubal: Efficient primitives for tubal-rank tensor learning operations on gpus. *IEEE Transactions on Parallel and Distributed Systems*, 31(3):595–610, 2019.

[2] Tao Zhang, Xiao-Yang Liu, and Xiaodong Wang. High performance gpu tensor completion with tubal-sampling pattern. *IEEE Transactions on Parallel and Distributed Systems*, 31(7):1724–1739, 2020.

[3] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010. URL `http://yann.lecun.com/exdb/mnist/`.

[4] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.

[5] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical Report 0, University of Toronto, Toronto, Ontario, 2009.

Contact information:
Zhiyuan Wang
vinlee624@gmail.com