DeepSZ: A Novel Framework to Compress Deep Neural Networks by Using Error-Bounded Lossy Compression

Sian Jin (The University of Alabama)
Sheng Di (Argonne National Laboratory)
Xin Liang (University of California, Riverside)
Jiannan Tian (The University of Alabama)
Dingwen Tao (The University of Alabama)
Frank Cappello (Argonne National Laboratory)
Outline

- **Introduction**
  - Neural Networks
  - Why compress Deep Neural Networks?

- **Background**
  - State-of-the-Art methods
  - Lossy Compression for floating-point data

- **Designs**
  - Overview of DeepSZ framework
  - Breakdown details in DeepSZ framework

- **Theoretical Analysis**
  - Performance analysis of DeepSZ
  - Comparison with other compressing methods

- **Experimental Evaluation**
Neural Networks

- **Typical DNNs consist of**
  - Convolutional layers. (i.e., Conv layers)
  - Fully connected layers. (i.e., FC layers)
  - Other layers. (Pooling layers etc.)

- **FC layers dominate the sizes of most DNNs**

<table>
<thead>
<tr>
<th>Neural Networks</th>
<th>LeNet-300-100</th>
<th>LeNet-5</th>
<th>AlexNet</th>
<th>VGG-16</th>
</tr>
</thead>
<tbody>
<tr>
<td>conv layers</td>
<td>0</td>
<td>3</td>
<td>5</td>
<td>13</td>
</tr>
<tr>
<td>fc-layers</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>ip1/fc6</td>
<td>300 × 784</td>
<td>500 × 800</td>
<td>4096 × 9216</td>
<td>4096 × 25088</td>
</tr>
<tr>
<td>ip2/fc7</td>
<td>100 × 300</td>
<td>10 × 500</td>
<td>4096 × 4096</td>
<td>4096 × 4096</td>
</tr>
<tr>
<td>ip3/fc8</td>
<td>10 × 100</td>
<td>-</td>
<td>1000 × 4096</td>
<td>1000 × 4096</td>
</tr>
<tr>
<td>conv fwd time</td>
<td>0 ms</td>
<td>0.5 ms</td>
<td>116.5 ms</td>
<td>149.8 ms</td>
</tr>
<tr>
<td>fc fwd time</td>
<td>0.30 ms</td>
<td>0.12 ms</td>
<td>2.5 ms</td>
<td>1.7 ms</td>
</tr>
<tr>
<td>total size</td>
<td>1.1 MB</td>
<td>1.7 MB</td>
<td>243.9 MB</td>
<td>553.4 MB</td>
</tr>
<tr>
<td>fc-layers’ size (%)</td>
<td>100%</td>
<td>95.3%</td>
<td>96.1%</td>
<td>89.4%</td>
</tr>
</tbody>
</table>
Why Compress Deep Neural Networks?

- Deep neural networks (DNNs) have **rapidly evolved** to be the state-of-the-art technique for many artificial intelligence tasks in various science and technology areas.
- Using **deeper and larger** DNNs can be an effective way to improve data analysis, but this leads to models that take up more space.

---

**LeNet**

<table>
<thead>
<tr>
<th>Input</th>
<th>Conv 1</th>
<th>Conv 2</th>
<th>fc 800</th>
<th>fc 500</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output (10)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

**VGG-16**

<table>
<thead>
<tr>
<th>Input</th>
<th>Conv 1-1</th>
<th>Conv 1-2</th>
<th>Pooling</th>
<th>Conv 2-1</th>
<th>Conv 2-2</th>
<th>Pooling</th>
<th>Conv 3-1</th>
<th>Conv 3-2</th>
<th>Conv 3-3</th>
<th>Pooling</th>
<th>Conv 4-1</th>
<th>Conv 4-2</th>
<th>Conv 4-3</th>
<th>Pooling</th>
<th>Conv 5-1</th>
<th>Conv 5-2</th>
<th>Conv 5-3</th>
<th>Pooling</th>
<th>fc 9216</th>
<th>fc 4096</th>
<th>fc 4096</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output (1000)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Why Compress Deep Neural Networks?

➢ Resource-limited platforms

• Train DNNs in the cloud using high-performance accelerators.
• Distribute the trained DNN models to end devices for inferences.
• Limited storage, transfer bandwidth and energy lost on fetching from external DRAM.
Why Compress Deep Neural Networks?

- **Resource-limited platforms**
  - Train DNNs in the cloud using high-performance accelerators.
  - Distribute the trained DNN models to end devices for inferences.
  - **Limited storage, transfer bandwidth** and **energy lost** on fetching from external DRAM.

- **Compressing neural networks**
  - Inferences accuracy after compressing and decompressing.
  - Compression ratio.
  - Encoding time.
  - Decoding time.

- **Challenges**
  - Achieve high compression ratio while remaining the accuracy.
  - Ensure fast to encode and decode.
Outline

- **Introduction**
  - Neural Networks
  - Why compress Deep Neural Networks?

- **Background**
  - State-of-the-Art methods
  - Lossy Compression for floating-point data

- **Designs**
  - Overview of DeepSZ framework
  - Breakdown details in DeepSZ framework

- **Theoretical Analysis**
  - Performance analysis of DeepSZ
  - Comparison with other compressing methods

- **Experimental Evaluation**
State-of-the-Art Methods

- **Deep Compression**
  - Compression framework with three main steps: **Pruning**, **Quantization** and **Huffman Encoding**.
State-of-the-Art Methods

➢ Weightless

- Compression framework: **Pruning**, Encode with a **Bloomier filter**
- Decode with four Hash function
Lossy Compression for Floating-Point Data

How SZ works

- Each data point’s value is predicted based on its neighboring data points by an adaptive, best-fit prediction method.
- Each floating-point weight value is converted to an integer number by a linear-scaling quantization based on the difference between the real value and predicted value and a specific error bound.
- Lossless compression is applied to reduce the data size thereafter.
How SZ works

• Each data point’s value is predicted based on its neighboring data points by an adaptive, best-fit prediction method.
• Each floating-point weight value is converted to an integer number by a linear-scaling quantization based on the difference between the real value and predicted value and a specific error bound.
• Lossless compression is applied to reduce the data size thereafter.

Advantages

• Higher compression ratio on 1D data than other state-of-the-art methods (such as ZFP).
• Error-bounded compression.
How We Solve The Problem

- DeepSZ
  - A lossy compression framework for DNNs.
  - Perform error-bounded lossy compression (SZ) on the pruned weights.
How We Solve The Problem

- **DeepSZ**
  - A *lossy compression framework* for DNNs.
  - Perform error-bounded lossy compression (SZ) on the pruned weights.

- **Challenges**
  - How can we determine an appropriate error bound for each layer in the neural network?
  - How can we maximize the overall compression ratio regarding different layers in the DNN under user-specified loss of inference accuracy?
Outline

- Introduction
  - Neural Networks
  - Why compress Deep Neural Networks?

- Background
  - State-of-the-Art methods
  - Lossy Compression for floating-point data

- Designs
  - Overview of DeepSZ framework
  - Breakdown details in DeepSZ framework

- Theoretical Analysis
  - Performance analysis of DeepSZ
  - Comparison with other compressing methods

- Experimental Evaluation
Overview of DeepSZ Framework

- **Prune**: remove unnecessary connections (i.e., weights) from DNNs and retrain the network to recover the inference accuracy.
- **Error bound assessment**: implement different error bounds on different FC layers in DNN and test their impacts on accuracy degradation.
- **Optimization**: use the result from last step to optimize error bound strategy for each FC layer.
- **Encode**: generate the compressed DNN models without retraining (in comparison: other approaches require another retrain process, which is highly time-consuming).
Network Pruning

• Turning weight matrix from dense to sparse by cutting close-zero weights to zero, based on user defined thresholds.
• Put masks on pruned weights and retrain the Neural Network by tuning the rest weights.
• Represent the product by a sparse matrix format. In this case, one data array (32 bits per value) and one index array (8 bits per value).

Reduce the size of fc-layers by about 8× to 20× if the pruning ratio is set to be around 90% to 96%.
Error Bound Assessment

- Test the inference accuracy with **only one compressed layer** in every test, dramatically reducing the test times.
- **Dynamically decide the testing range** of error bound to further reduce test times.
- Collect the data from testing.

Comparation of SZ and ZFP

Inference accuracy of different error bounds on the fc-layers in AlexNet.
Optimization of Error Bound Configuration

\[ \Delta^* = \sum \Delta_{\ell,eb_{\ell}}, \quad \Delta^* < 2\%. \]

- Compression error introduced in each fc-layer has **independent impact on final network’s output.**
- The relationship between final output and accuracy loss is approximately **linear.**

Determine the best-fit error bound for each layer by a dynamic planning algorithm. Based on expected accuracy loss or expected compression ratio.
Generation of Compressed Model

• Use SZ lossy compression on the data arrays with the error bounds (obtained in Step-3) and the best-fit lossless compression on the index arrays.

Compression ratios of different layers’ index arrays with different lossless compressors on AlexNet and VGG-16.
Generation of Compressed Model

- Use SZ lossy compression on the data arrays with the error bounds (obtained in Step-3) and the best-fit lossless compression on the index arrays.

Decoding

- **Decompress** the data arrays using the SZ lossy compression and the index arrays using the best-fit lossless compression.
- The sparse matrix can be **reconstructed** based on the decompressed data array and index array for each fc-layer.
- Decode the whole neural networks.

Compression ratios of different layers’ index arrays with different lossless compressors on AlexNet and VGG-16.
Outline

- **Introduction**
  - Neural Networks
  - Why compress Deep Neural Networks?

- **Background**
  - State-of-the-Art methods
  - Lossy Compression for floating-point data

- **Designs**
  - Overview of DeepSZ framework
  - Breakdown details in DeepSZ framework

- **Theoretical Analysis**
  - Performance analysis of DeepSZ
  - Comparison with other compressing methods

- **Experimental Evaluation**
Experimental Configuration

Hardware and Software

• Four Nvidia Tesla V100 GPUs
  ▪ Pantarhei cluster node at the University of Alabama.
  ▪ Each V100 has 6 GB of memory.
  ▪ GPUs and CPUs are connected via NVLinks.
• Intel Core i7-8750H Processors (with 32 GB of memory) for decoding analysis.
• Caffe deep learning framework.
• SZ lossy compression library (v2.0).
Experimental Configuration

- **Hardware and Software**
  - **Four Nvidia Tesla V100 GPUs**
    - Pantarhei cluster node at the University of Alabama.
    - Each V100 has 6 GB of memory.
    - GPUs and CPUs are connected via NVLinks.
  - Intel Core i7-8750H Processors (with 32 GB of memory) for decoding analysis.
  - **Caffe** deep learning framework.
  - **SZ lossy compression** library (v2.0).

- **DNNs and Datasets**
  - **LeNet-300-100, LeNet-5, AlexNet, and VGG-16.**
  - LeNet300-100 and LeNet-5 on the MNIST dataset.
  - AlexNet and VGG-16 on the ImageNet dataset.
Encoding

- The computational cost is focused mostly on performing the tests with different error bounds to check the corresponding accuracies.
- Performing the tests is still much faster than retraining.
Performance Analysis of DeepSZ

- **Encoding**
  - The computational cost is focused mostly on performing the tests with different error bounds to check the corresponding accuracies.
  - Performing the tests is still much faster than retraining.

- **Decoding**
  - The overall time complexity of DeepSZ’s decoding is $\Theta(n)$.
  - Still comparatively low even on end devices.
Comparison with Other Methods

- **Weightless**
  - Weightless has higher time overhead for encoding than DeepSZ does because of *retraining*.
  - Weightless has higher time overhead for decoding than DeepSZ does because of *Bloomier filter structure*.
  - **Only one layer is compressible** (usually the largest layer).
Comparison with Other Methods

- **Weightless**
  - Weightless has higher time overhead for encoding than DeepSZ does because of *retraining*.
  - Weightless has higher time overhead for decoding than DeepSZ does because of *Bloomier filter structure*.
  - *Only one layer is compressible* (usually the largest layer).

- **Deep Compression**
  - Adopts a *simple quantization* technique on the pruned weights.
  - Higher time overhead than DeepSZ does for encoding, because of *retraining*.
Outline

- **Introduction**
  - Neural Networks
  - Why compress Deep Neural Networks?

- **Background**
  - State-of-the-Art methods
  - Lossy Compression for floating-point data

- **Designs**
  - Overview of DeepSZ framework
  - Breakdown details in DeepSZ framework

- **Theoretical Analysis**
  - Performance analysis of DeepSZ
  - Comparison with other compressing methods

- **Experimental Evaluation**
Compression Ratio Evaluation

FC-layers’ compression statistics for 4 Neural Networks

- **DeepSZ** shows the size of overall compression of the framework.
Experimental Evaluation

- Top-1 Accuracy means the top class (the one having the highest probability) is the same as the target label.
- Top-5 Accuracy means the target label is one of the top 5 predictions with the highest prediction probability.
- **Compression ratio of 45x to 116x with top-1 accuracy loss lower than 0.25%**.
- Note for LeNet, as the network is much simpler, features decent compression ratio with almost no accuracy loss.

<table>
<thead>
<tr>
<th>Neural Network</th>
<th>Top-1 Accuracy</th>
<th>Top-5 Accuracy</th>
<th>fc-layers’ Size</th>
<th>Compress Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>LeNet-300-100 original</td>
<td>98.35%</td>
<td>-</td>
<td>1056 KB</td>
<td></td>
</tr>
<tr>
<td>LeNet-300-100 DEEPSZ</td>
<td>98.31%</td>
<td>-</td>
<td><strong>19.1 KB</strong></td>
<td>55.8×</td>
</tr>
<tr>
<td>LeNet-5 original</td>
<td>99.13%</td>
<td>-</td>
<td>1620 KB</td>
<td></td>
</tr>
<tr>
<td>LeNet-5 DEEPSZ</td>
<td>99.16%</td>
<td>-</td>
<td><strong>28.3 KB</strong></td>
<td>57.3×</td>
</tr>
<tr>
<td>AlexNet original</td>
<td>57.41%</td>
<td>80.40%</td>
<td>234.5 MB</td>
<td></td>
</tr>
<tr>
<td>AlexNet DEEPSZ</td>
<td>57.28%</td>
<td>80.58%</td>
<td><strong>5.15 MB</strong></td>
<td>45.5×</td>
</tr>
<tr>
<td>VGG-16 original</td>
<td>68.05%</td>
<td>88.34%</td>
<td>494.5 MB</td>
<td></td>
</tr>
<tr>
<td>VGG-16 DEEPSZ</td>
<td>67.80%</td>
<td>88.20%</td>
<td><strong>4.277 MB</strong></td>
<td>115.6×</td>
</tr>
</tbody>
</table>
Experimental Evaluation

- **Higher compression ratio** compared to other compression methods.
- Much lower accuracy loss before retraining.
- More **flexibility** on tradeoff between accuracy and compression ratio.

<table>
<thead>
<tr>
<th>model</th>
<th>quantization (Deep Compression)</th>
<th>Bloomier Filter (Weightless)</th>
<th>SZ (DeepSZ)</th>
</tr>
</thead>
<tbody>
<tr>
<td>LeNet-300-100</td>
<td>0.22%</td>
<td>-</td>
<td>0.12%</td>
</tr>
<tr>
<td>LeNet-5</td>
<td>0.30%</td>
<td>-</td>
<td>-0.03%</td>
</tr>
<tr>
<td>AlexNet</td>
<td>1.56%</td>
<td>-</td>
<td>0.13%</td>
</tr>
<tr>
<td>VGG-16</td>
<td>2.81%</td>
<td>&gt;3.0%</td>
<td>0.25%</td>
</tr>
</tbody>
</table>

Inference accuracy degradation of different techniques based on comparable compression ratio.

Comparison of compression ratios of different techniques on LeNet-300-100, LeNet-5, AlexNet, and VGG-16.
Performance Evaluation

- DeepSZ has **lower encoding and decoding time overheads** than Deep Compression and Weightless
- Capable to store on end device and decompress DNNs when necessary.

For example, DeepSZ spends 26 ms in lossless decompression, 108 ms in SZ lossy decompression, and 162 ms in reconstructing the sparse matrix on AlexNet. As a comparison, the time for one forward pass with 50 images per batch takes 1,100 ms on AlexNet
Conclusion and Future Work

DeepSZ

- Avoid the costly retraining process after compression, leading to a significant **performance improvement** in encoding DNNs.
- **Controllable** tradeoff between accuracy and compression ratio.
Conclusion and Future Work

- **DeepSZ**
  - Avoid the costly retraining process after compression, leading to a significant **performance improvement** in encoding DNNs.
  - **Controllable** tradeoff between accuracy and compression ratio.

- **Future Work**
  - Evaluate our proposed DeepSZ on more neural network architectures.
  - DeepSZ evaluation on convolutional layers.
  - **Use DeepSZ for improving GPU memory utilization.**
Thank you!

Any questions are welcome!

Contact

Dingwen Tao: tao@cs.ua.edu
Sian Jin: sjin6@crimson.ua.edu