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

ACM HPDC 2019

The 28th International Symposium on High-Performance Parallel and Distributed Computing

Phoenix, Arizona, USA - June 24-28, 2019

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)

June 2019

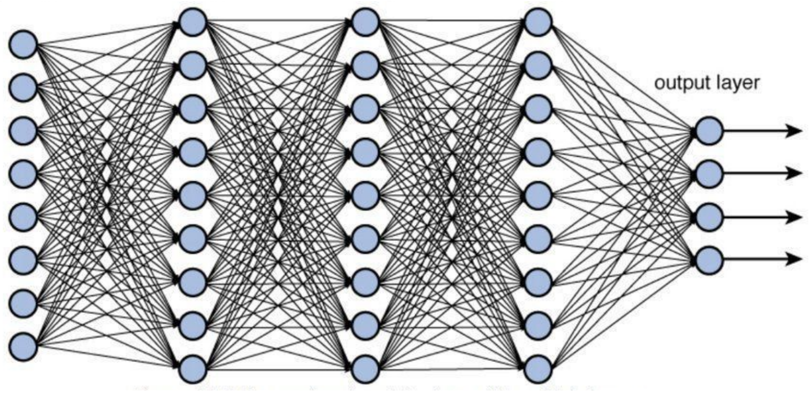
Outline

Neural Networks

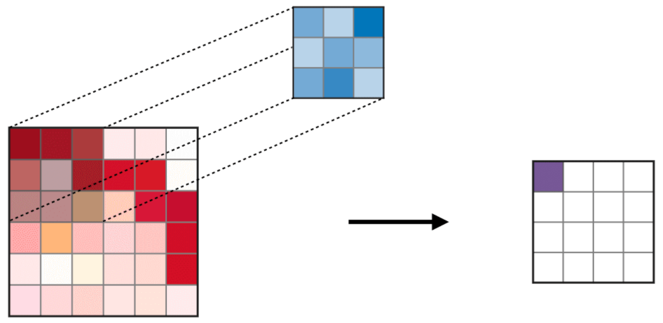
- **Typical DNNs consist of**
  - Convolutional layers. (i.e., Conv layers)
  - Fully connected layers. (i.e., FC layers)
  - Other layers. (Pooling layers etc.)
- **FC layers dominate the sizes of most DNNs**

Neural Networks	LeNet-300-100	LeNet-5	AlexNet	VGG-16
conv layers	0	3	5	13
fc-layers	3	2	3	3
ip1/fc6	300×784	500×800	4096×9216	4096×25088
ip2/fc7	100×300	10×500	4096×4096	4096×4096
ip3/fc8	10×100	-	1000×4096	1000×4096
conv fwd time	0 ms	0.5 ms	116.5 ms	149.8 ms
fc fwd time	0.30 ms	0.12 ms	2.5 ms	1.7 ms
total size	1.1 MB	1.7 MB	243.9 MB	553.4 MB
fc-layers' size (%)	100%	95.3%	96.1%	89.4%

Architectures of example neural networks



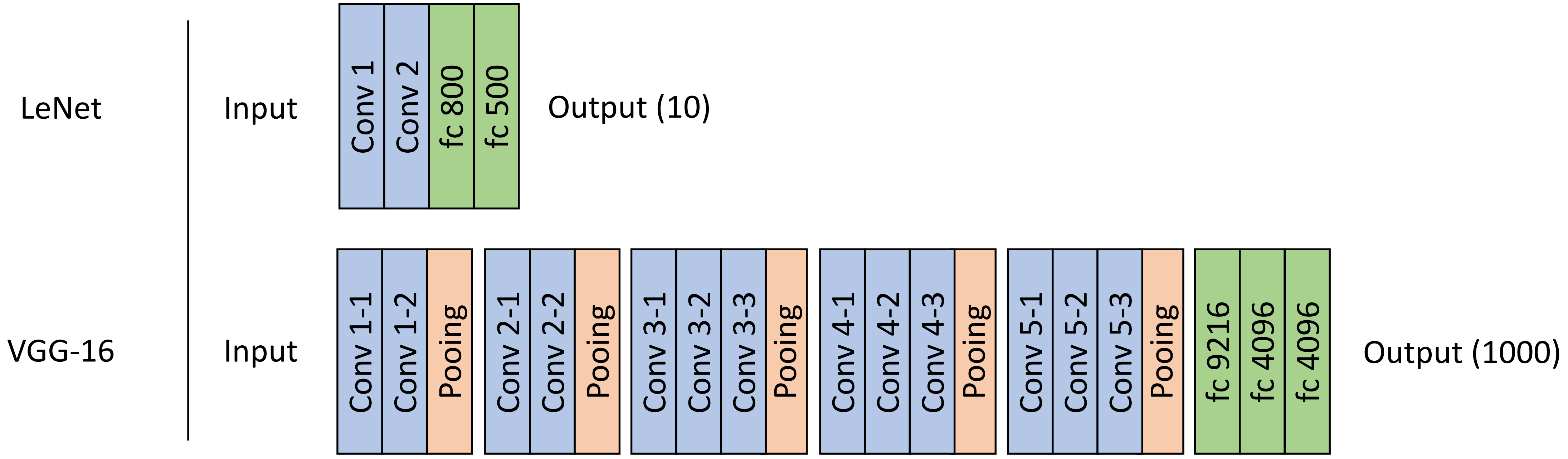
FC layers



Conv layers

Why Compress Deep Neural Networks?

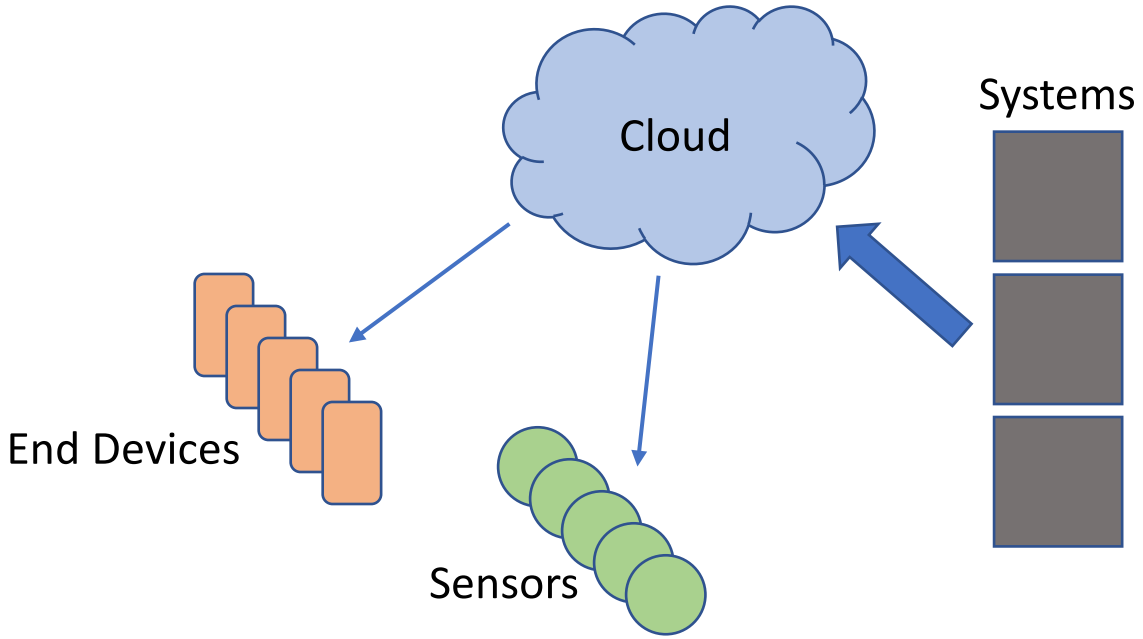
- Deep neural networks (DNNs) have **rapidly evolved** to be the state-of-the-art technique for many artificial intelligence tasks in various science and technology areas.
- Using **deeper and larger** DNNs can be an effective way to improve data analysis, but this leads to models that take up more space.



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.

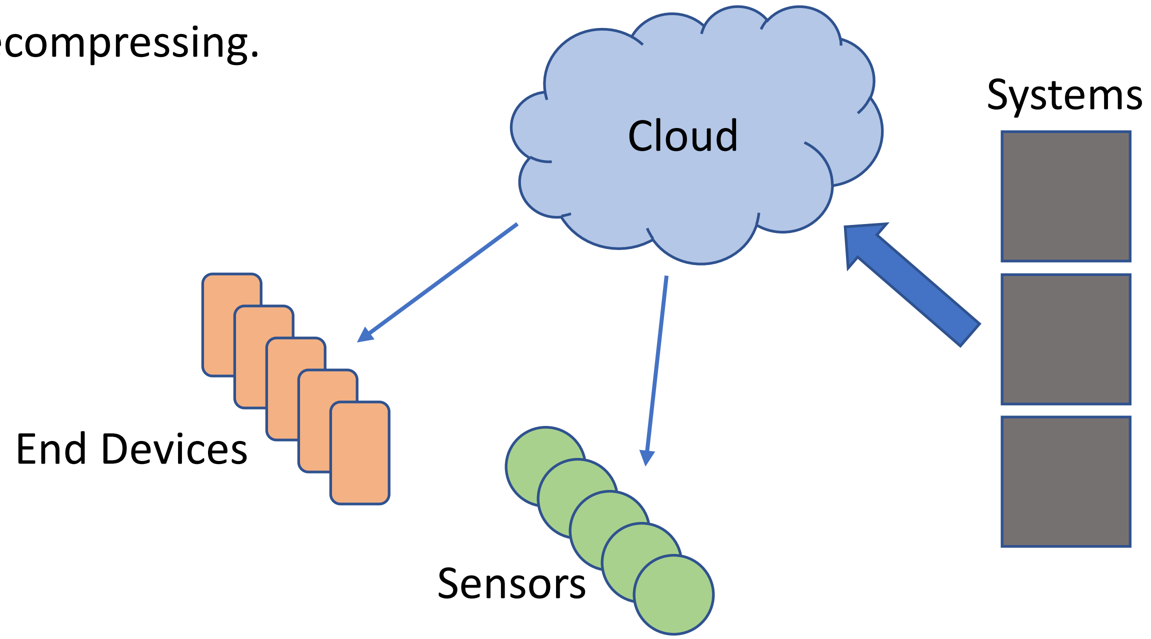


Why Compress Deep Neural Networks?

- **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.

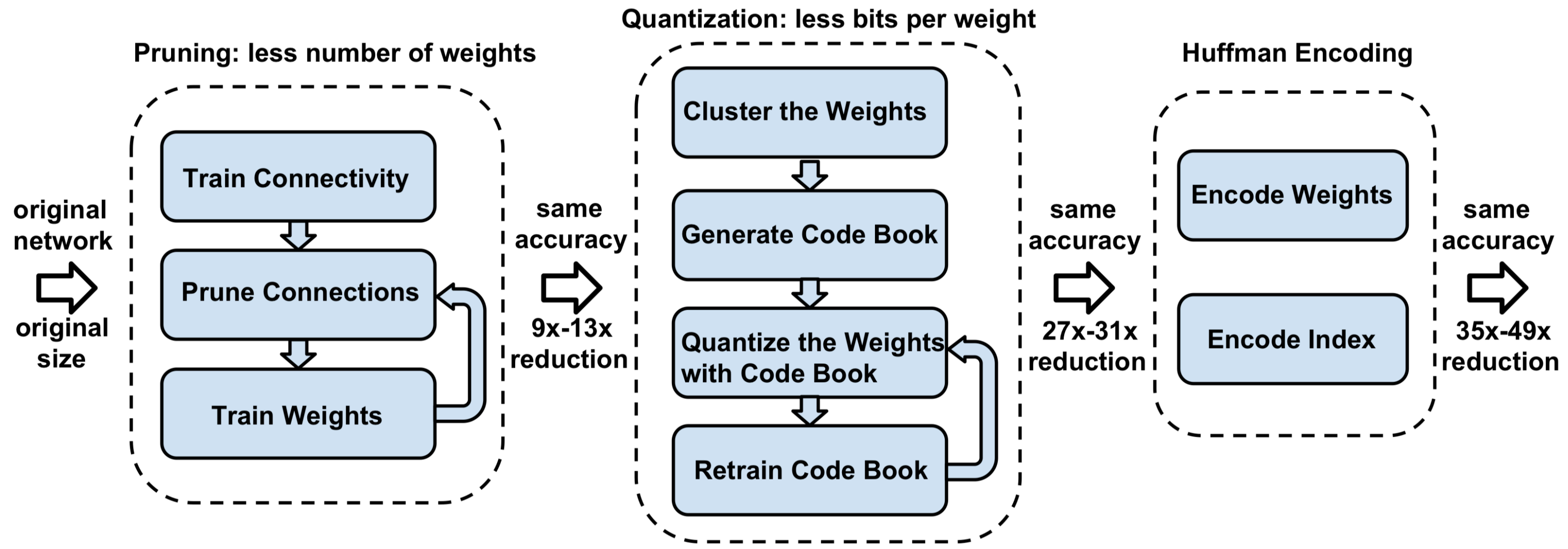


Outline

State-of-the-Art Methods

➢ **Deep Compression**

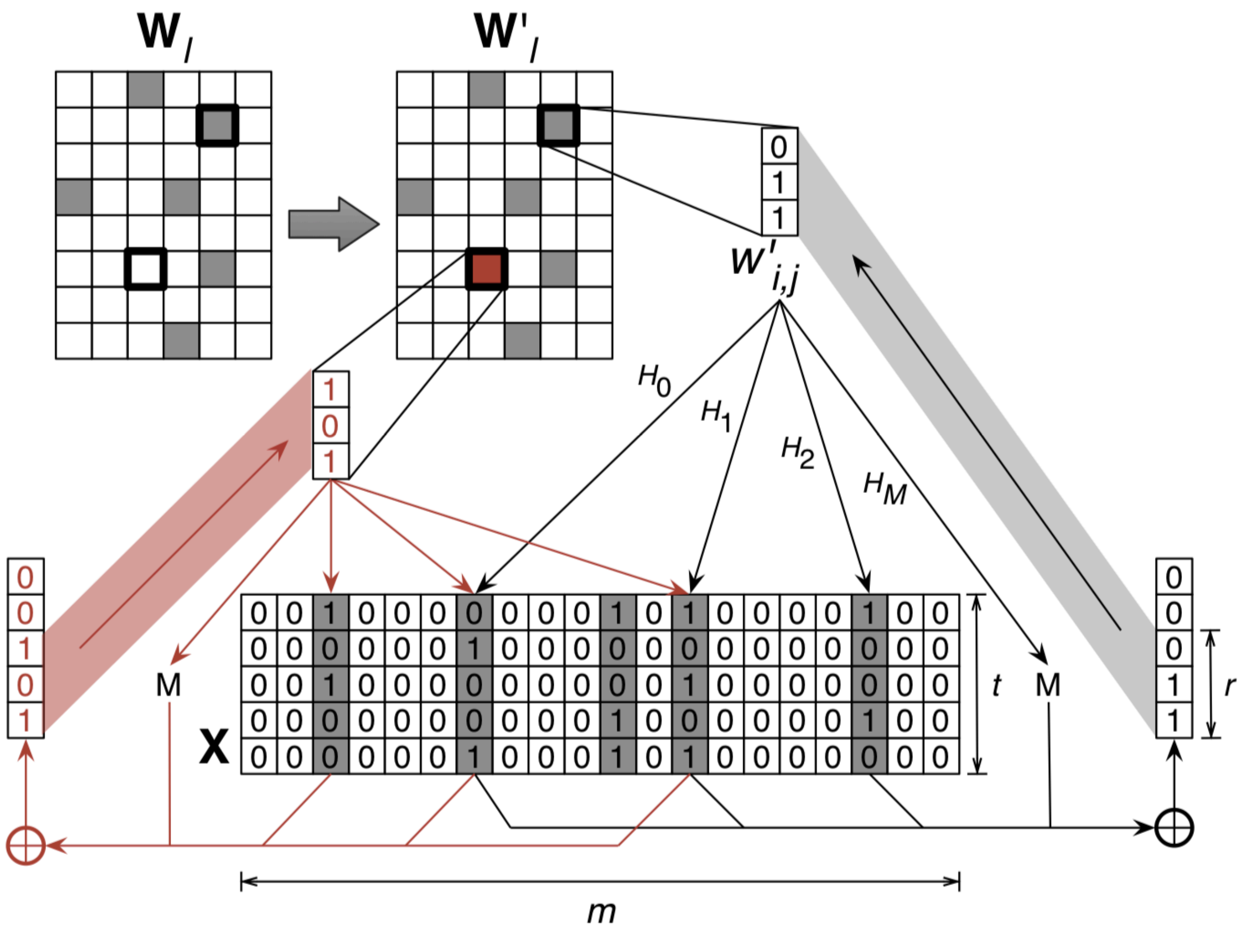
- Compression framework with three main steps: **Pruning**, **Quantization** and **Huffman Encoding**.



State-of-the-Art Methods

➢ **Weightless**

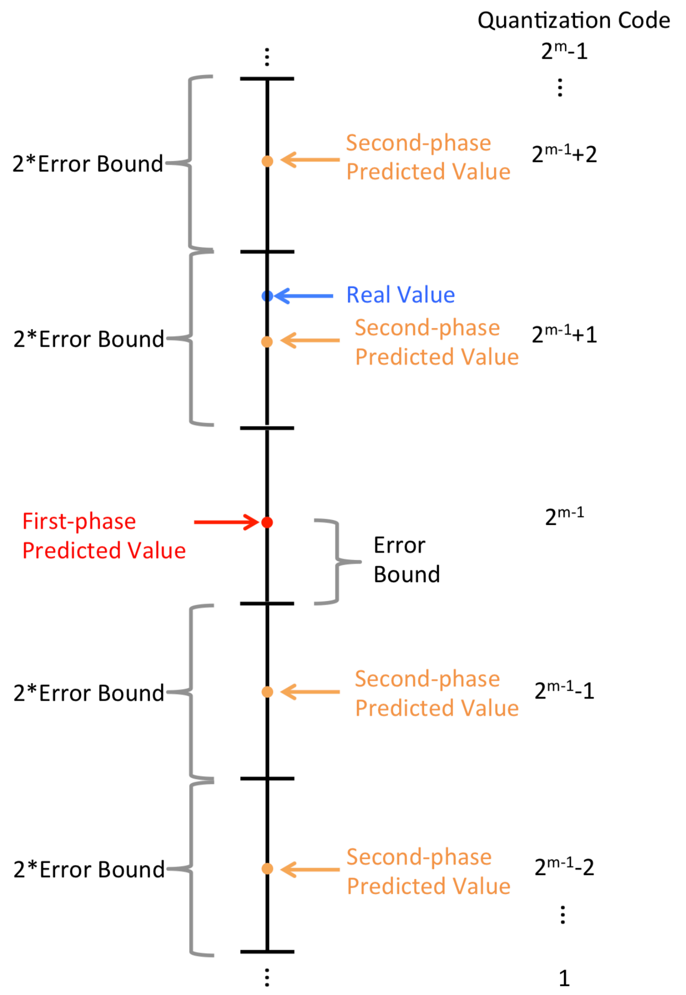
- Compression framework: **Pruning**, Encode with a **Bloomier filter**
- Decode with four Hash function



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.



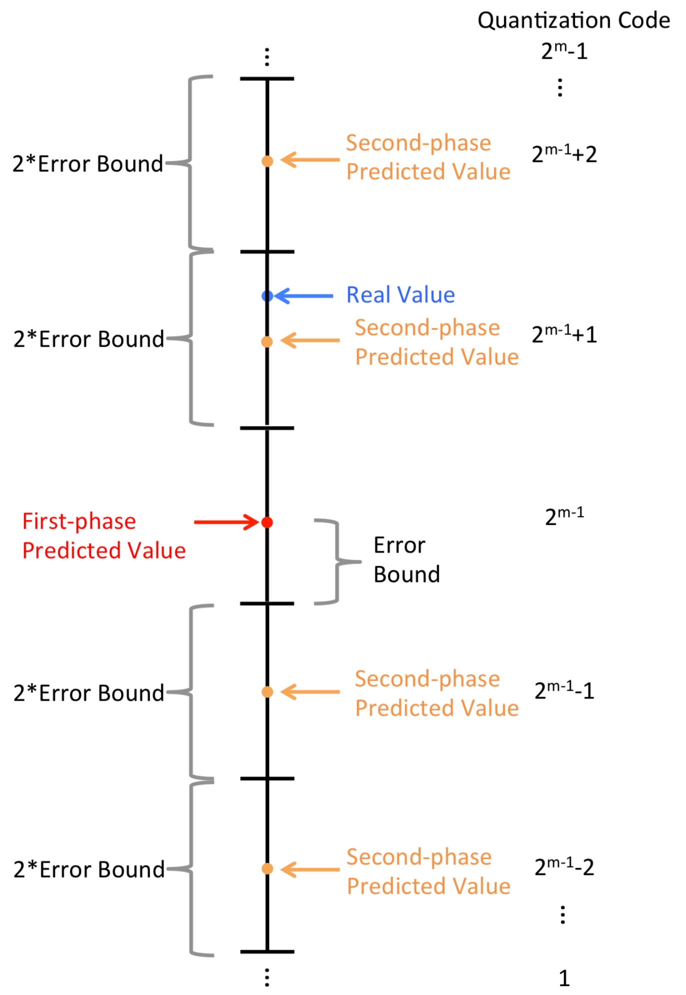
Lossy Compression for Floating-Point Data

➢ **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.

➢ **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.

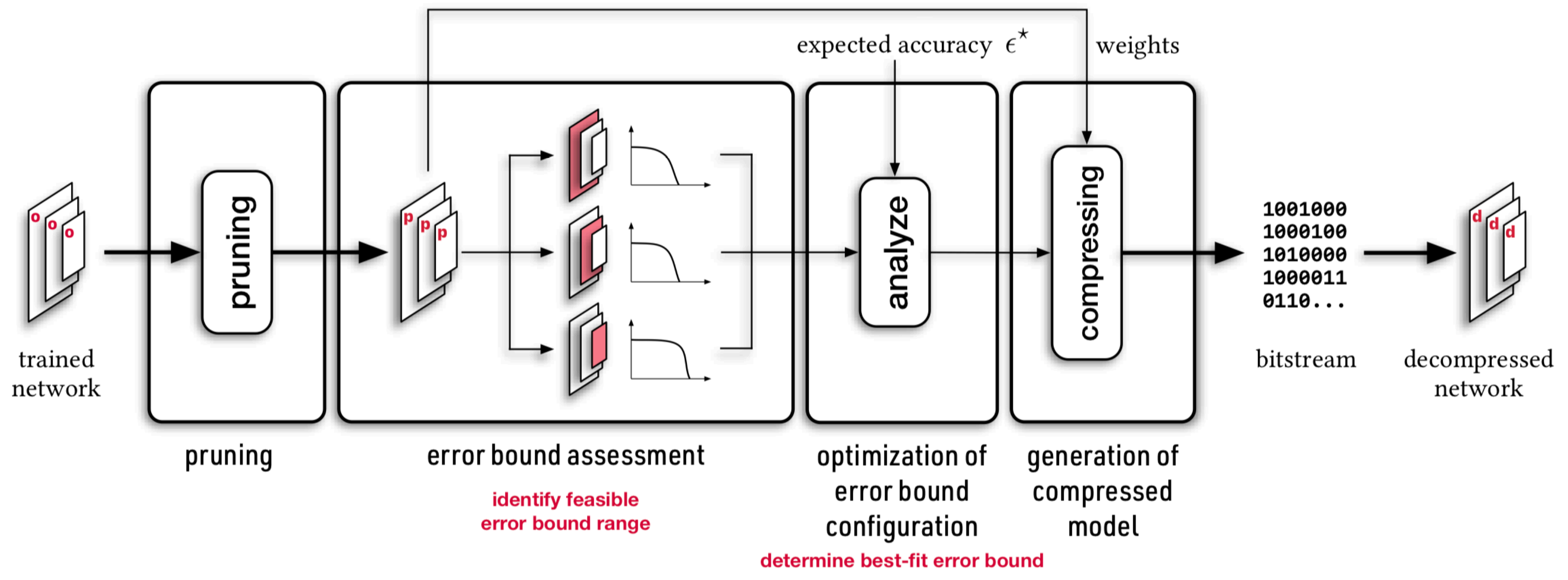
How We Solve The Problem

- **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

Overview of DeepSZ Framework



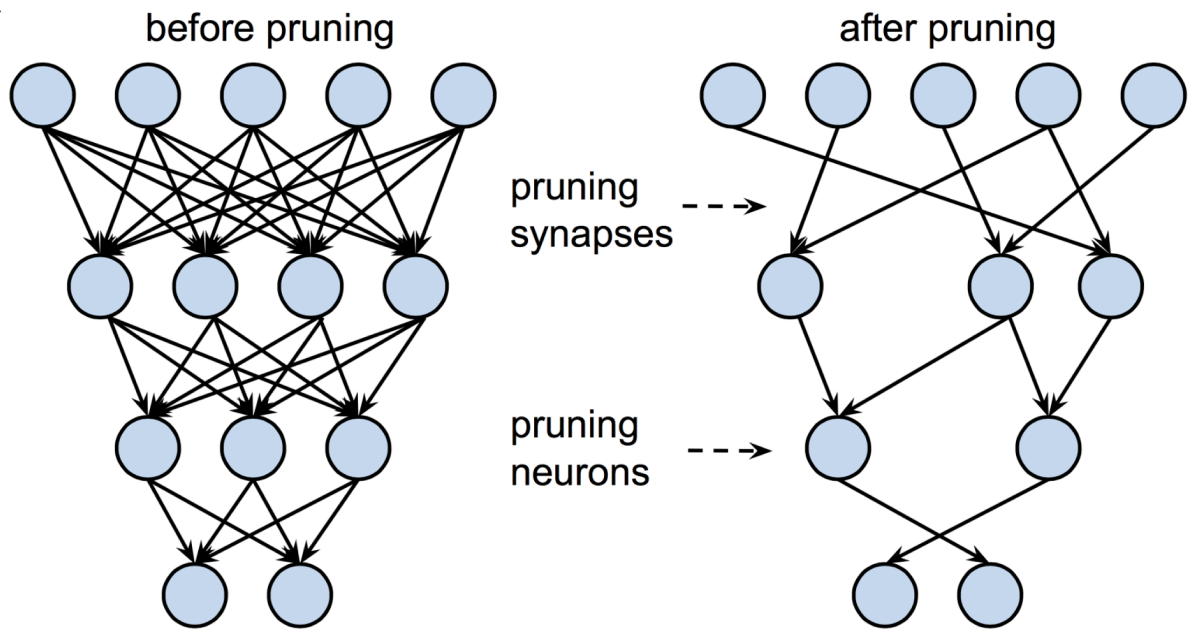
expected accuracy $\epsilon^\star$
weights
trained network
pruning
analyze
compressing
1001000
1000100
1010000
1000011
0110...
bitstream
decompressed network
pruning
error bound assessment
identify feasible error bound range
optimization of error bound configuration
determine best-fit error bound
generation of compressed model

- 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

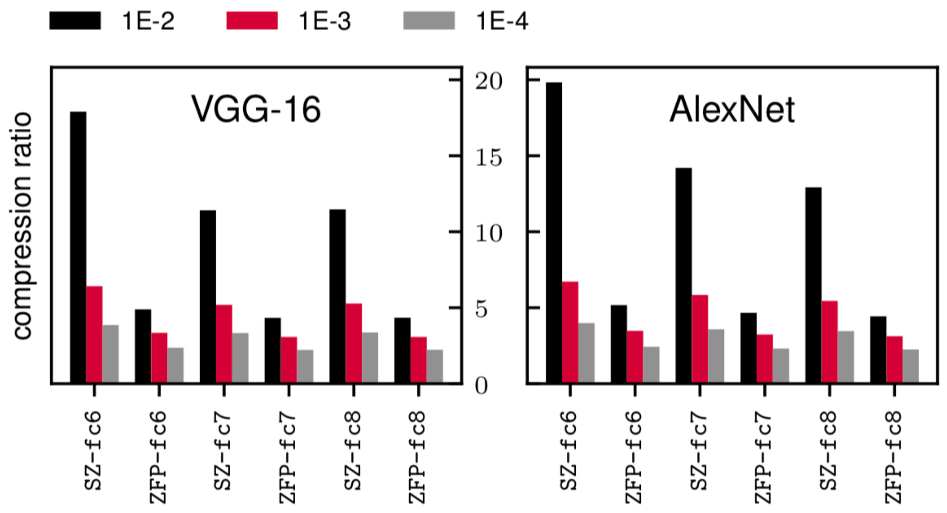
- Turning weight matrix from dense to sparse by **cutting close-zero weights to zero**, based on user defined thresholds.
- Put masks on pruned weights and **retrain** the Neural Network by tuning the rest weights.
- Represent the product by a **sparse matrix format**. In this case, one data array (32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

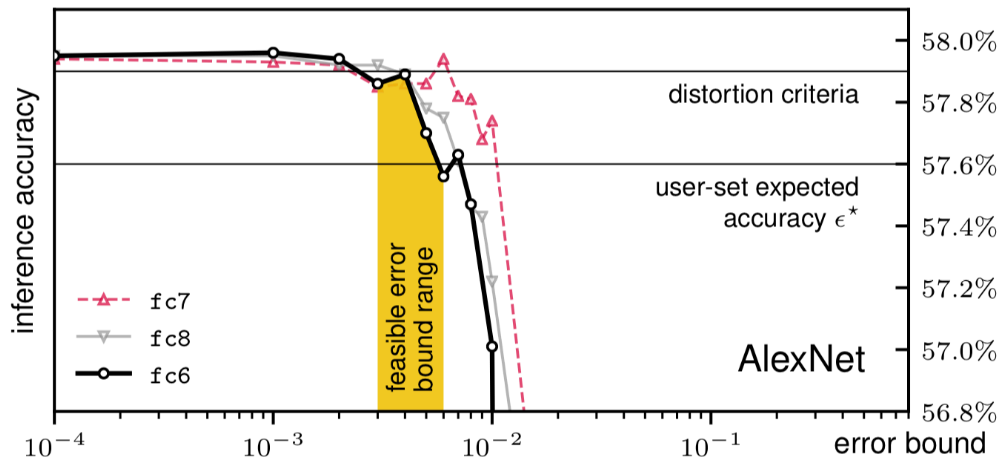


Error Bound Assessment

- 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



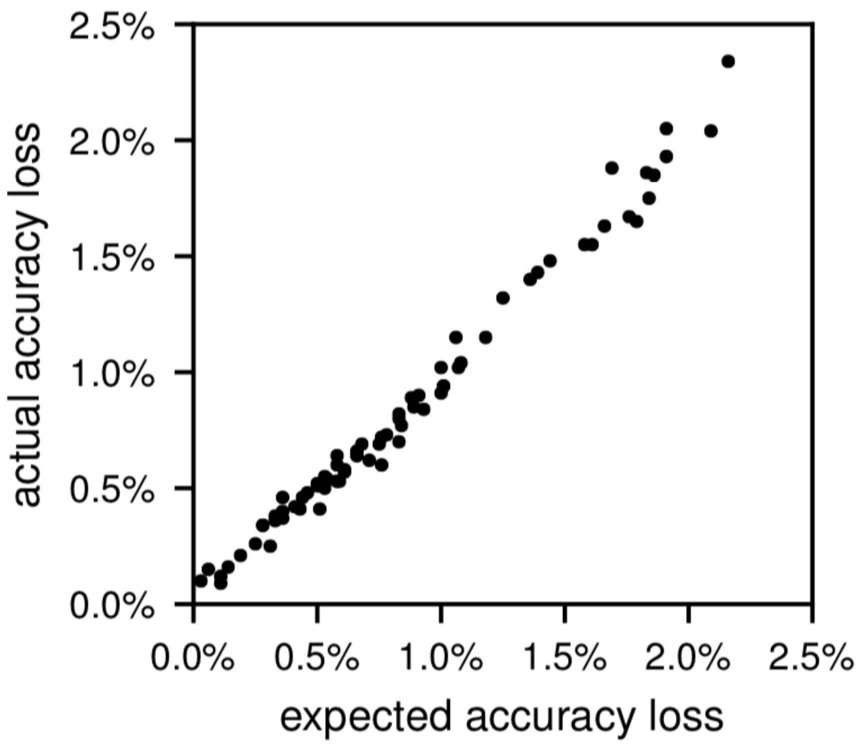
Inference accuracy of different error bounds on the fc-layers in AlexNet.

Optimization of Error Bound Configuration

$$\Delta^\star = \sum \Delta_{\ell, eb_\ell}, \qquad \Delta^\star < 2\%.$$

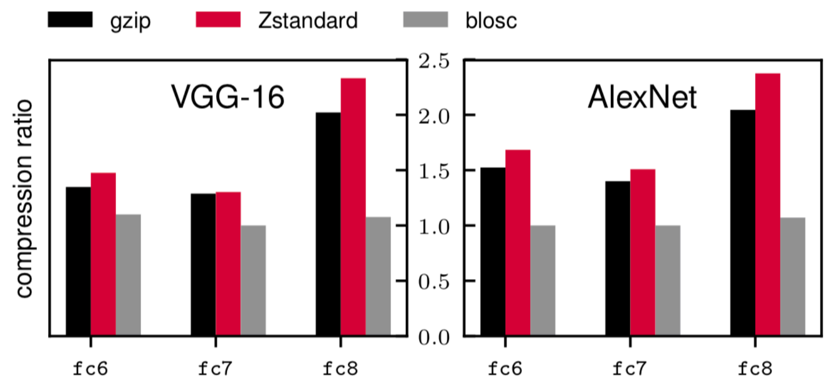
- Compression error introduced in each fc-layer has **independent impact on final network's output**.
- The relationship between final output and accuracy loss is approximately **linear**.

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

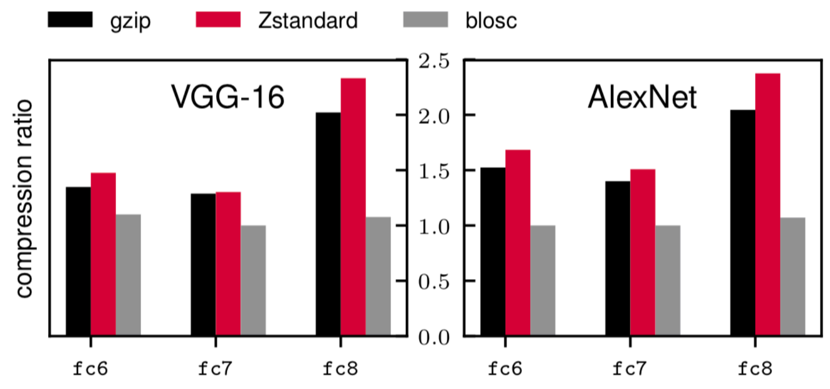
- Use SZ lossy compression on the data arrays with the error bounds (obtained in Step-3) and the best-fit lossless compression on the index arrays.



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.



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

➢ **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 **reconstruct**ed based on the decompressed data array and index array for each fc-layer.
- Decode the whole neural networks.

Outline

Experimental Configuration

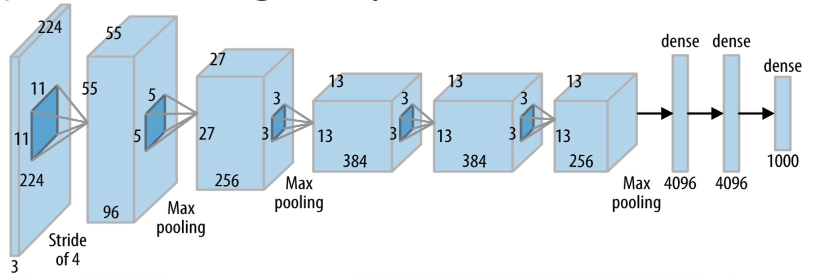
- **Hardware and Software**
  - **Four Nvidia Tesla V100 GPUs**
    - Pantarhei cluster node at the University of Alabama.
    - Each V100 has 6 GB of memory.
    - GPUs and CPUs are connected via NVLinks.
  - Intel Core i7-8750H Processors (with 32 GB of memory) for decoding analysis.
  - **Caffe**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

Experimental Configuration

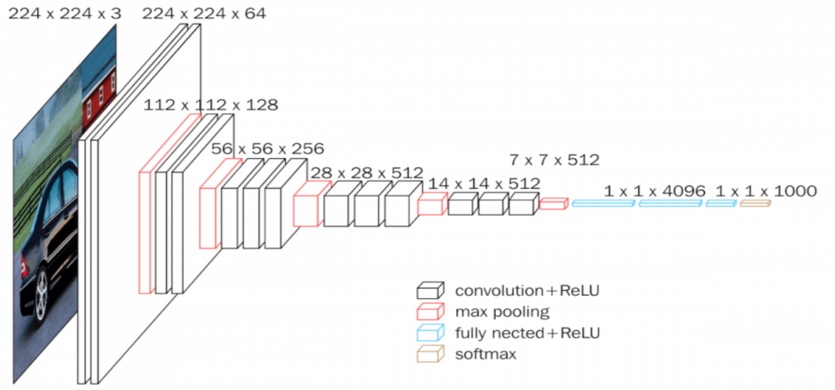
- **Hardware and Software**
  - **Four Nvidia Tesla V100 GPUs**
    - Pantarhei cluster node at the University of Alabama.
    - Each V100 has 6 GB of memory.
    - GPUs and CPUs are connected via NVLinks.
  - Intel Core i7-8750H Processors (with 32 GB of memory) for decoding analysis.
  - **Caffe**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

AlexNet

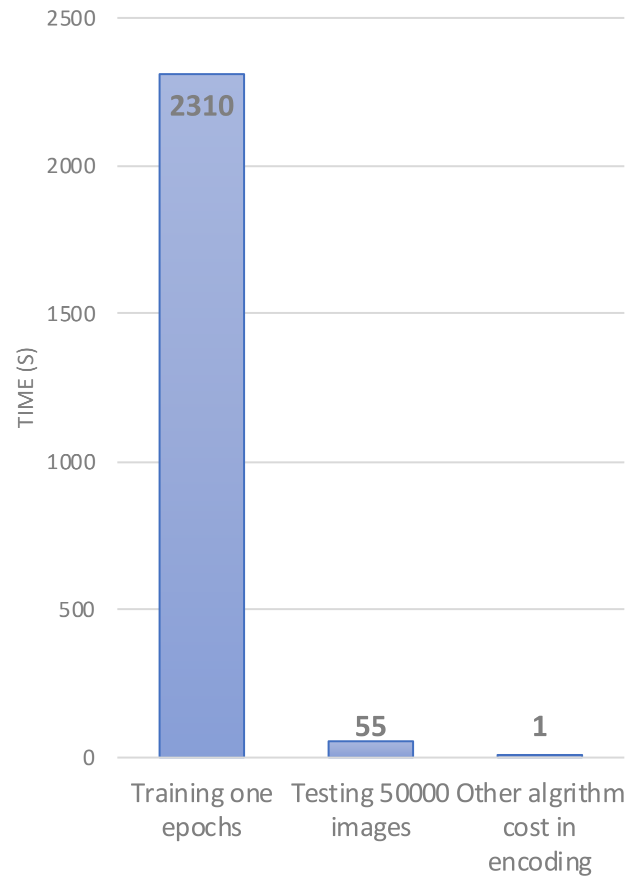


VGG-16



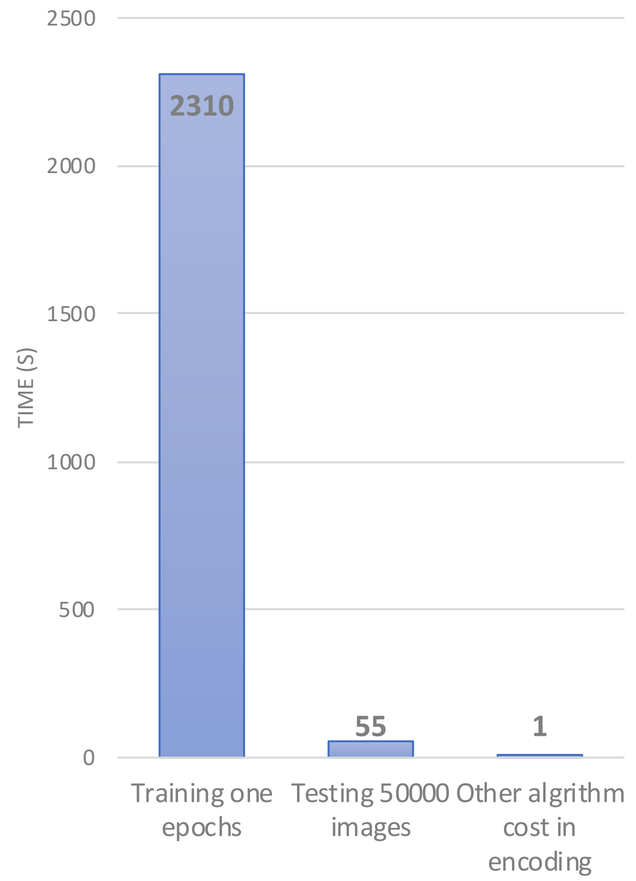
Performance Analysis of DeepSZ

- **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.



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 Θ (*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).

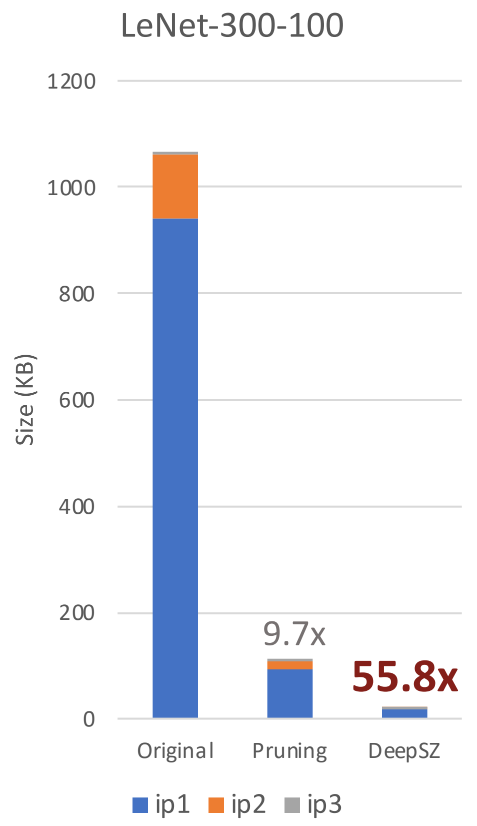
- **Deep Compression**
  - Adopts a **simple quantization** technique on the pruned weights.
  - Higher time overhead than DeepSZ does for encoding, because of **retraining**.

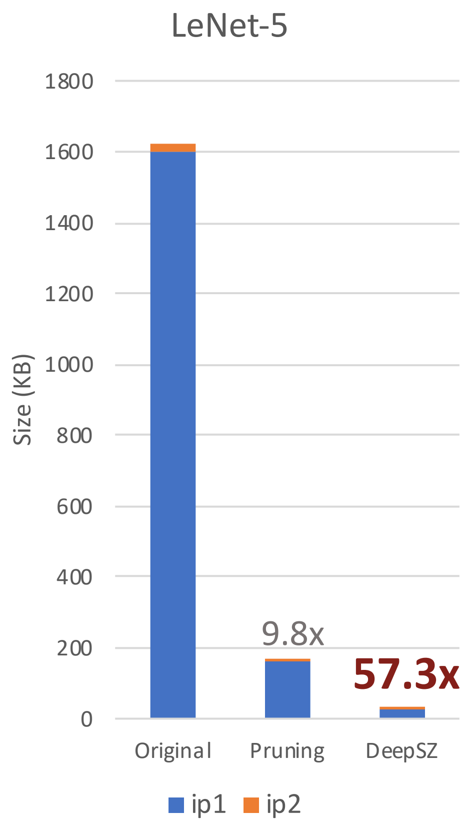
Outline

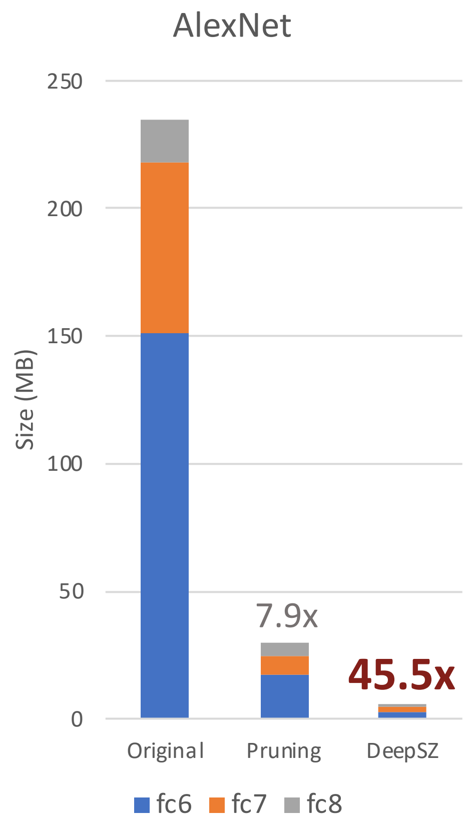
- **Introduction**
  - Neural Networks
  - Why compress Deep Neural Networks?
- **Background**
  - State-of-the-Art methods
  - Lossy Compression for floating-point data
- **Designs**
  - Overview of DeepSZ framework
  - Breakdown details in DeepSZ framework
- **Theoretical Analysis**
  - Performance analysis of DeepSZ
  - Comparison with other compressing methods
- **Experimental Evaluation**

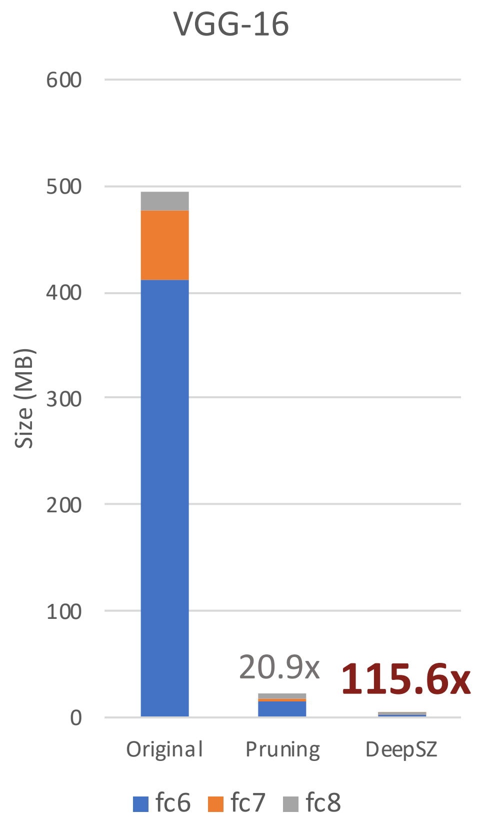
Compression Ratio Evaluation

FC-layers' compression statistics for 4 Neural Networks





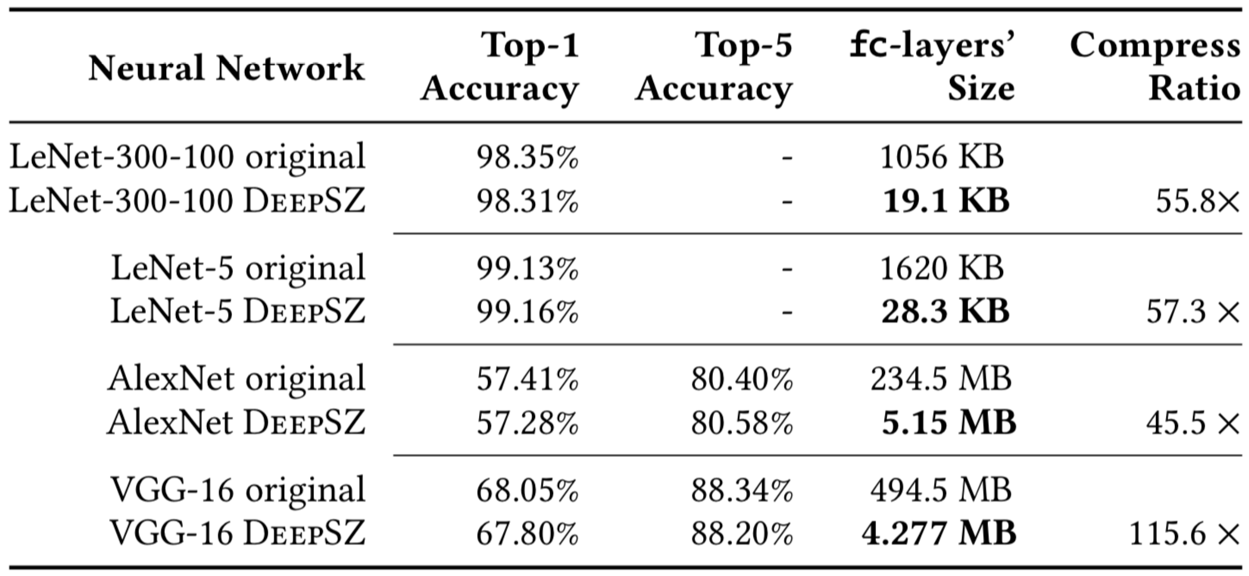




- DeepSZ shows the size of overall compression of the framework.

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.



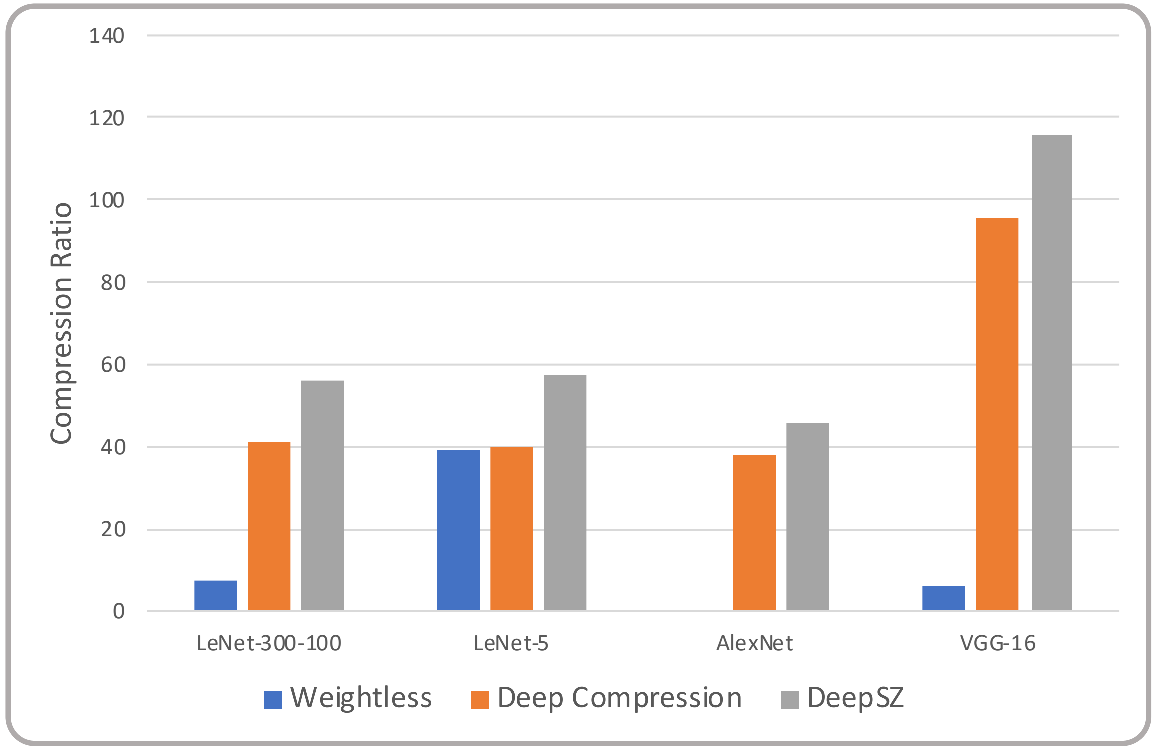
Neural Network	Top-1 Accuracy	Top-5 Accuracy	fc-layers' Size	Compress Ratio
LeNet-300-100 original	98.35%	-	1056 KB	
LeNet-300-100 DeepSZ	98.31%	-	**19.1 KB**	55.8×
LeNet-5 original	99.13%	-	1620 KB	
LeNet-5 DeepSZ	99.16%	-	**28.3 KB**	57.3 ×
AlexNet original	57.41%	80.40%	234.5 MB	
AlexNet DeepSZ	57.28%	80.58%	**5.15 MB**	45.5 ×
VGG-16 original	68.05%	88.34%	494.5 MB	
VGG-16 DeepSZ	67.80%	88.20%	**4.277 MB**	115.6 ×

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.

model	quantization (Deep Compression)	Bloomier Filter (Weightless)	SZ (DeepSZ)
LeNet-300-100	0.22%	-	0.12%
LeNet-5	0.30%	-	−0.03%[7]
AlexNet	1.56%	-	0.13%
VGG-16	2.81%	>3.0%	0.25%

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



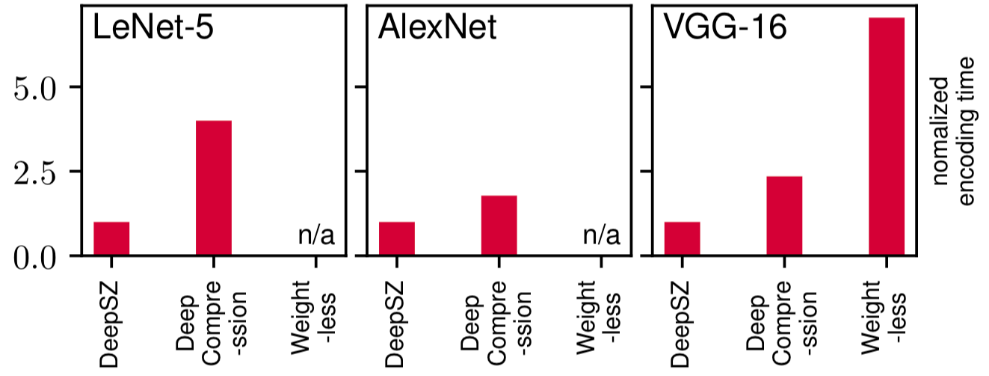
Comparison of compression ratios of different techniques on LeNet-300-100, LeNet-5, AlexNet, and VGG-16.

Performance Evaluation

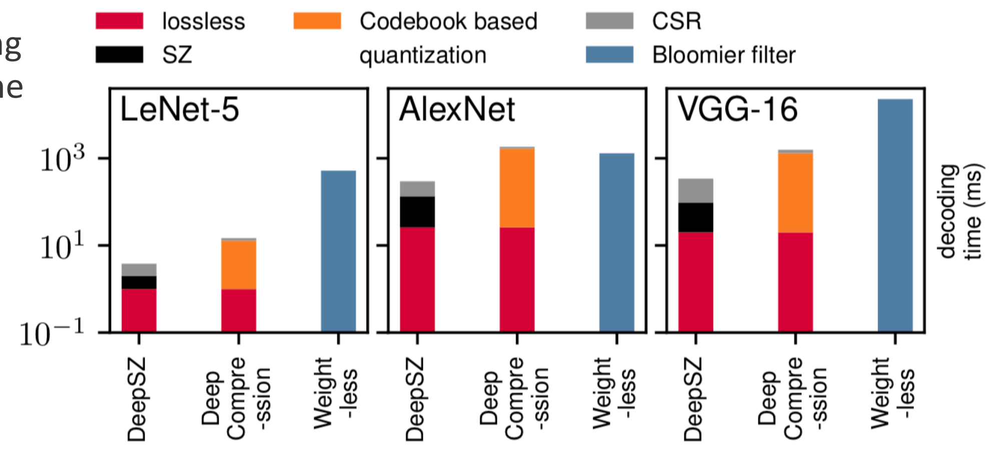
- DeepSZ has **lower encoding and decoding time overheads** than Deep Compression and Weightless
- Capable to store on end device and decompress DNNs when necessary.

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

Time breakdown of encoding and decoding with different lossy compression techniques.



(a) Normalized encoding time with different solutions on GPUs.



(b) Breakdown of decoding time with different solutions on CPU.

Conclusion and Future Work

➢ **DeepSZ**

- A novel lossy compression framework, called DeepSZ, for **effectively compressing** sparse weights in deep neural networks.
- 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**
  - A novel lossy compression framework, called DeepSZ, for **effectively compressing** sparse weights in deep neural networks.
  - 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