Delta-DNN: Efficiently Compressing Deep Neural Networks via Exploiting Floats Similarity

Zhenbo Hu[§,*], Xiangyu Zou[§,*], Wen Xia[§,$], Sian Jin[ζ], Dingwen Tao[ζ], Yang Liu[§,$], Weizhe Zhang[§,$], Zheng Zhang[§]

[§]School of Computer Science and Technology, Harbin Institute of Technology, Shenzhen, China
[$]Cyberspace Security Research Center, Peng Cheng Laboratory, Shenzhen, China
[ζ]School of Electrical Engineering and Computer Science, Washington State University, WA, USA

The 49th International Conference on Parallel Processing (ICPP 2020)

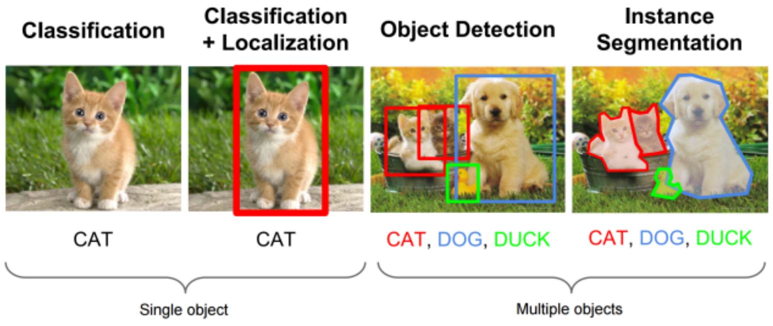
August 17-20, 2020

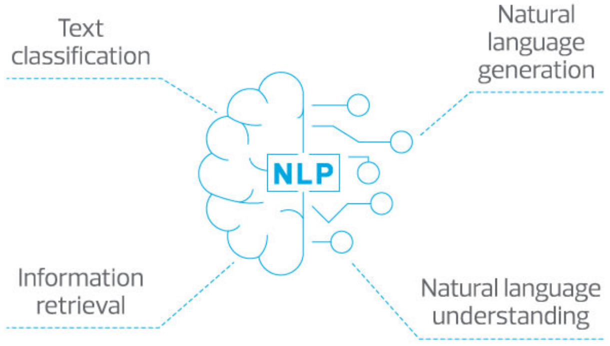
Outline

- **Introduction**
  - Neural Networks
  - Why compress Neural Networks?
- **Background and motivation**
  - Data compression techniques & compressing DNNs
  - Observation and motivation
- **Design and implementation**
  - Overview of Delta-DNN framework
  - Breakdown details in Delta-DNN framework
- **Typical application scenarios**
- **Performance evaluation**

Neural Networks

- ➢ **Deep Neural Networks are designed to solve complicated and non-linear problems**
- ➢ **Typical Deep Neural Networks Applications**
  - **computer vision** (i.e., Image classification, Image classification + localization, Object detection, Instance Segmentation, etc.)
  - **natural language processing** (i.e., Text classification, Information retrieval, Natural language generation, Natural language understanding, etc.)

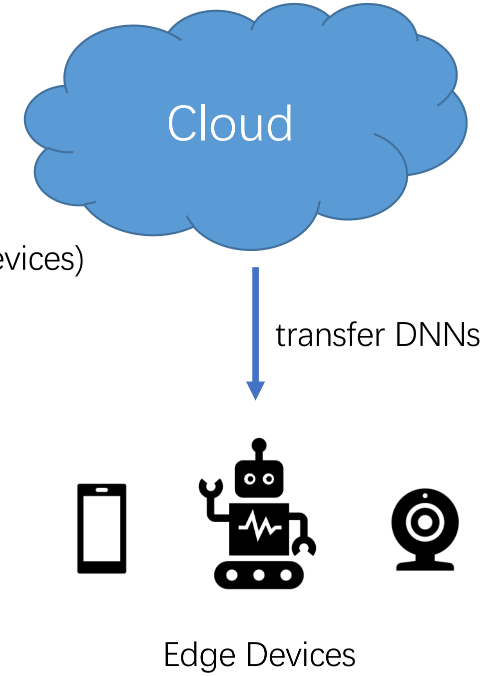




Why compress Neural Networks?

- **To further improve the inference accuracy, DNNs are becoming deeper and more complicated**

- **A DNNs Practical Application**
  - To train a DNN in cloud servers with high-performance accelerators
  - Then transfer the trained DNN model to the edge devices (i.e., mobile devices, IoT devices)
  - The edge devices run the DNN model



Compressing Neural Networks is an effective way to reduce the transfer cost.

Data compression techniques

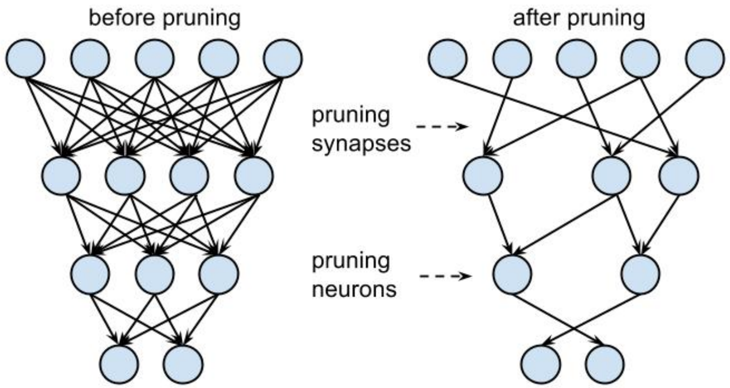
- **Data compression techniques are especially important for data reduction.**

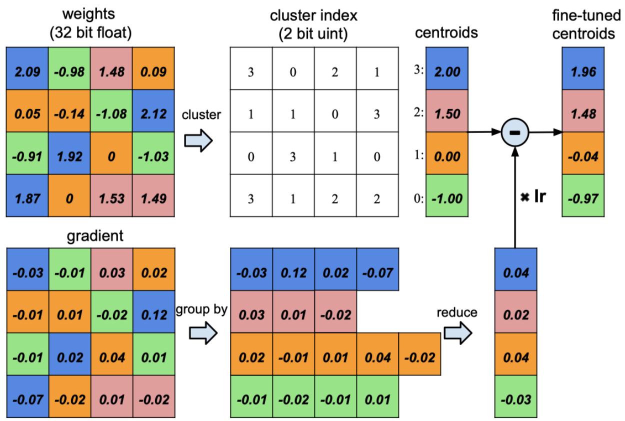
- **Lossless compression**
  - Usually deal with data as byte streams, and reduce data at the bytes/string level based on classic algorithm such as Huffman coding, dictionary coding, etc.
  - Delta compression observes the high data similarity (data redundancy), then only records the delta data for space savings.

- **Lossy compression**
  - Typical lossy compressors are for images, such as JPEG2000.
  - Lossy compression of floating-point data from HPC, such as ZFP, SZ, etc.
  - SZ lossy compression with a data-fitting predictor and a point-wise error bound controlled quantizator.

Compressing DNNs

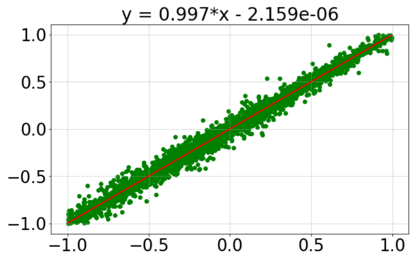
- **Compressing DNNs means compressing a large amount of very random floating-point numbers**
- **Special technologies for compressing DNNs**
    - Pruning (removing some unimportant parameters)
    - Quantization (transforming the floats parameters into low bits numbers)



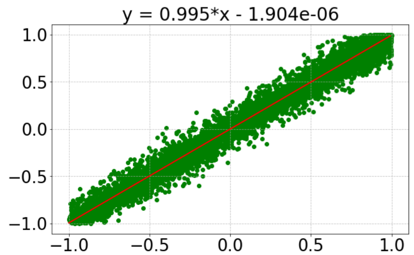


Observation and motivation

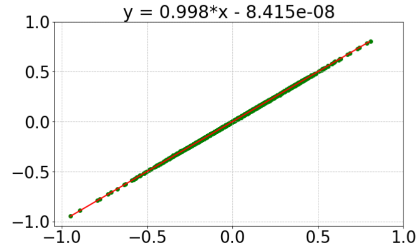
- **The floating–point numbers of the neighboring networks are very similar**
  - Linear fitting close to $y = x$ & SSIM close to 1.0



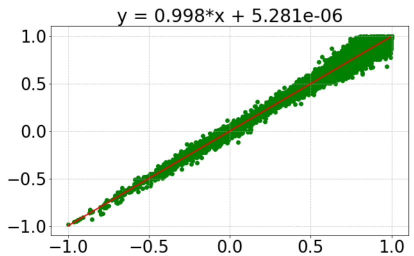
(a) VGG-16, SSIM: 0.99994



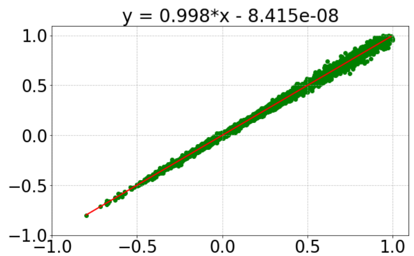
(b) ResNet101, SSIM: 0.99971



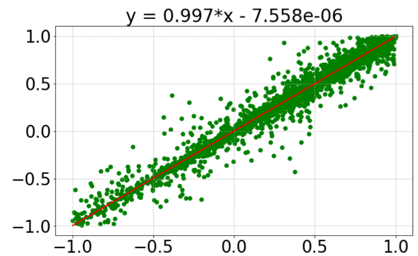
(c) GoogLeNet, SSIM: 0.99999



(d) EfficientNet, SSIM: 0.99624



(e) MobileNet, SSIM: 0.99998



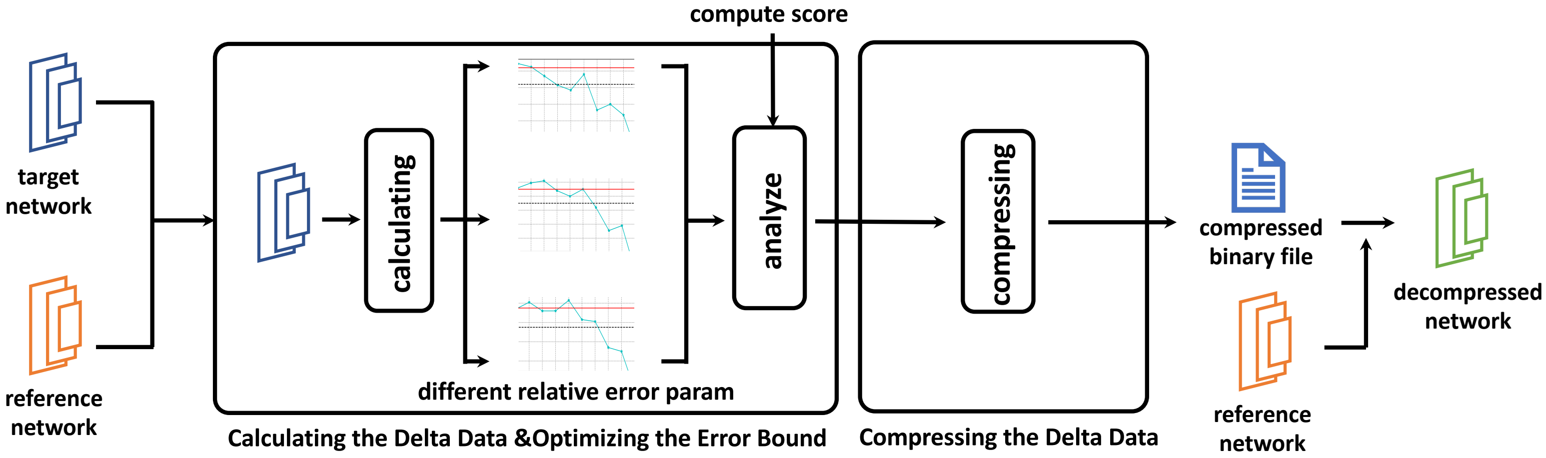
(f) ShuffleNet, SSIM: 0.99759

Observation and motivation

➢ Motivation

- Inspired by the **delta compression** technique, we calculate the **delta data** of the similar floats between two neighboring neural networks.
- We employ the ideas of **error-bound SZ lossy compression**, i.e., a data-fitting predictor and an error-controlled quantizator, to compress the delta data.

Overview of Delta-DNN framework



- **Calculating the Delta Data:** calculate the lossy delta data of the target and reference networks (including all layers).
- **Optimizing the Error Bound:** select the suitable error bound used for maximizing the lossy compression efficiency.
- **Compressing the Delta Data:** reduce the delta data size by using lossless compressors.

Calculating the Delta Data

- **Following the idea of SZ lossy compressor**

convert float-point numbers to integers & most integers are equal to zero

- Calculate and quantize

$$M_i = \left\lfloor \frac{A_i - B_i}{2 \cdot log(1+\epsilon)} + 0.5 \right\rfloor$$

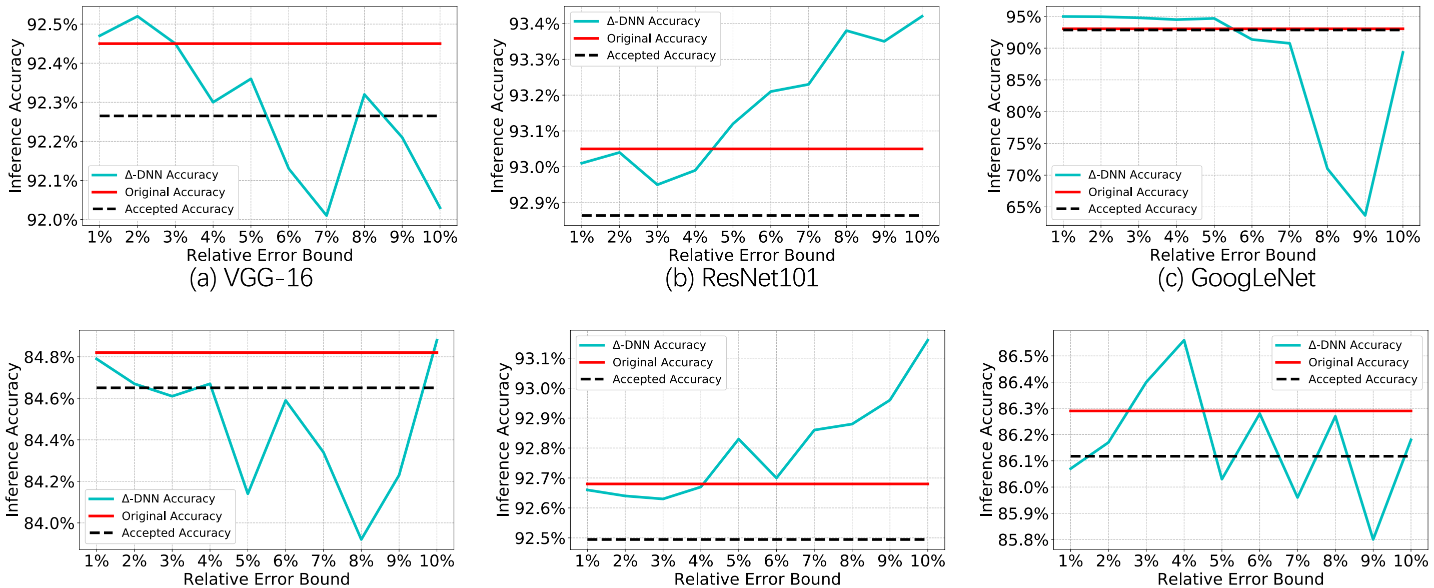
- Recover the parameters

$$A_i' = 2 \cdot M_i \cdot log(1+\epsilon) + B_i$$

A_i is a parameter from target network, B_i is the corresponding parameters from reference network, ϵ is the predefined relative error bound, and is an integer for recording the delta data of A_i and B_i.

Optimizing the Error Bound

- **How to get a reasonable relative error bound to maximize the compression ratio without compromising DNNs' inference accuracy?**
  - **Two key metrics:** compression ratio, inference accuracy loss

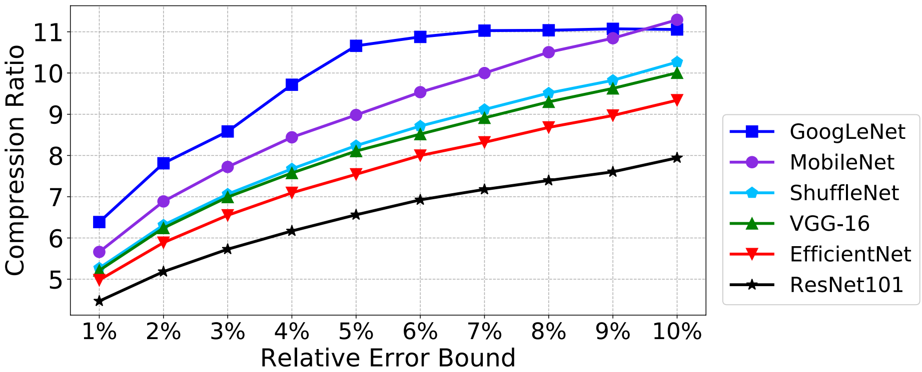


(a) VGG-16 (b) ResNet101 (c) GoogLeNet

(d) EfficientNet (e) MobileNet (f) ShuffleNet

the impact of inference accuracy with different error bounds

Optimizing the Error Bound



The impact of compression ratio with different error bounds

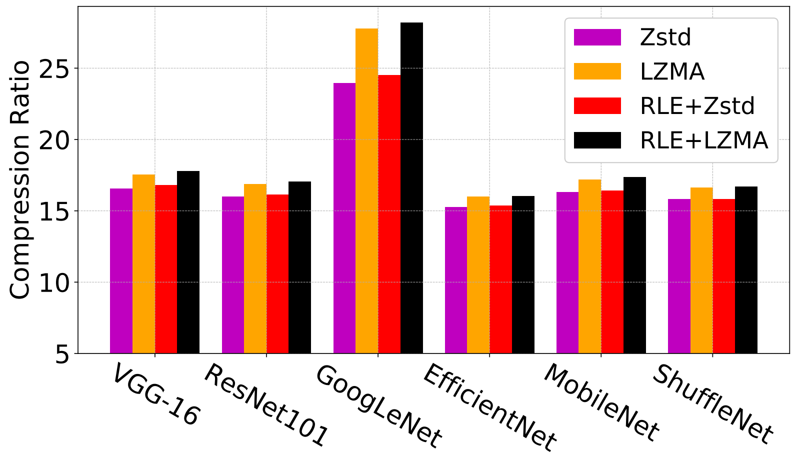
➢ **Our solution:**

- Collecting the results of compression ratio and the inference accuracy degradation along with the available error bounds
- Assessing the collected results to select an optimal error bound according to Formula as below

$$Score = \alpha \cdot \Phi + \beta \cdot \Omega, \qquad (\alpha + \beta = 1)$$

Compressing the Delta Data

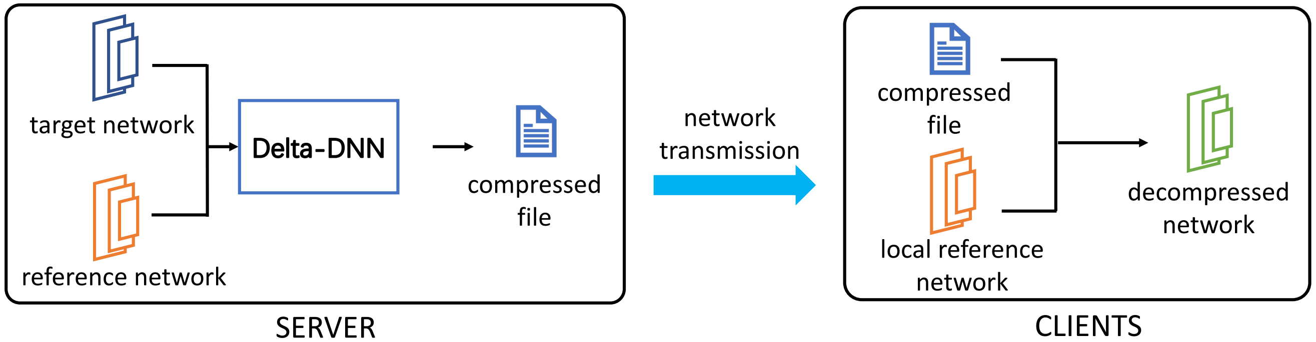
- **To further reduce the delta data space**
  - Zstd
  - LZMA
  - Run-Length Encoding (RLE) + Zstd
  - Run-Length Encoding (RLE) + LZMA



Compression ratios of Delta-DNN running 4 compressors

Optimizing Network Transmission for DNNs

- **DNNs are trained on the server and deployed locally on the client (such as mobile device and IoT device)**
  - **Bottleneck:** network transmission for DNNs

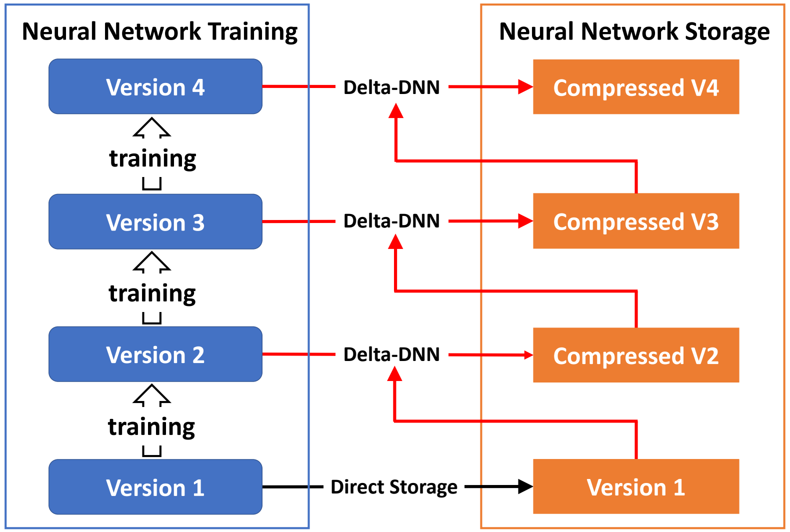


Delta-DNN for reducing network transmission

Saving Storage Space for DNNs

- **In some situations, DNNs need be continuously trained and updated**
  - Transfer Learning
  - Incremental Learning

- **Saving multiple snapshots or versions of DNNs**
  - Using Delta-DNN to save storage space



Delta-DNN for reducing storage cost

Experimental Setup

➢ **Hardware and Software**

- a NVIDIA TITAN RTX GPU with 24 GB of memory.
- an Intel Xeon Gold 6130 processor with 128 GB of memory.
- Pytorch deep learning framework.
- SZ lossy compression library.

➢ **DNNs and Datasets**

- CIFAR-10 dataset.
- VGG-16, ResNet101, GoogLeNet, EfficientNet, MobileNet, and ShuffleNet.

Compression Performance of Delta-DNN

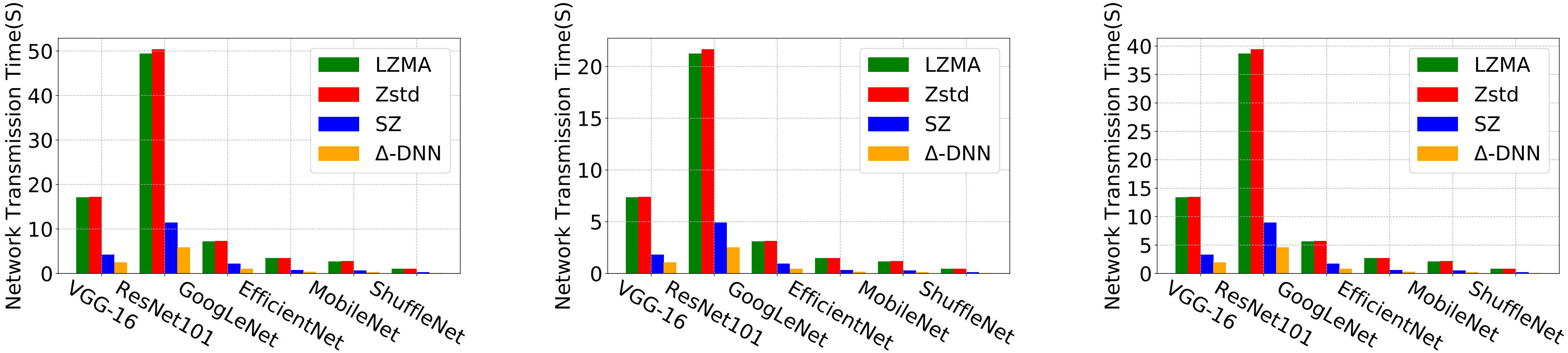
- **Compression ratio results of the four compressor on six popular DNNs** (Default relative inference accuracy loss less than 0.2%)

Networks	Original Size	Compression Ratio (and the error bound)				Inference Accuracy (and the differences)		
		LZMA	Zstd	SZ	Δ-DNN	Original	SZ	Δ-DNN
VGG-16	56.2 MB	1.096	1.088	4.415 (7%)	7.394 (8%)	92.45%	92.31% (–0.15%)	92.32% (–0.15%)
ResNet101	162.6 MB	1.098	1.078	4.192 (5%)	9.341 (10%)	93.05%	92.87% (–0.19%)	93.44% (+0.42%)
GoogLeNet	23.6 MB	1.097	1.078	3.565 (2%)	7.811 (2%)	94.95%	94.88% (–0.07%)	94.95% (+0.00%)
EfficientNet	11.3 MB	1.099	1.078	3.204 (1%)	10.266 (10%)	84.82%	84.76% (–0.07%)	84.88% (+0.07%)
MobileNet	8.9 MB	1.101	1.077	3.788 (3%)	9.627 (9%)	92.68%	92.57% (–0.12%)	93.16% (+0.52%)
ShuffleNet	3.5 MB	1.097	1.076	3.192 (1%)	11.291 (10%)	86.29%	86.19% (–0.12%)	86.18% (–0.13%)

Delta-DNN achieves about **2x~10x** higher compression ratio compared with the state-of-the-art approaches, LZMA, Zstd, and SZ.

Case 1: Optimizing Network Transmission

➢ **Using Delta-DNN to reduce network transmissions**



(a) Mobile Broadband Downloading (b) Fixed Broadband Downloading (c) Fixed Broadband Uploading

Delta-DNN **significantly reduces** the network consumption of six neural networks.

The network bandwidth data is from the global average network bandwidth on SPEEDTEST in January 2020.

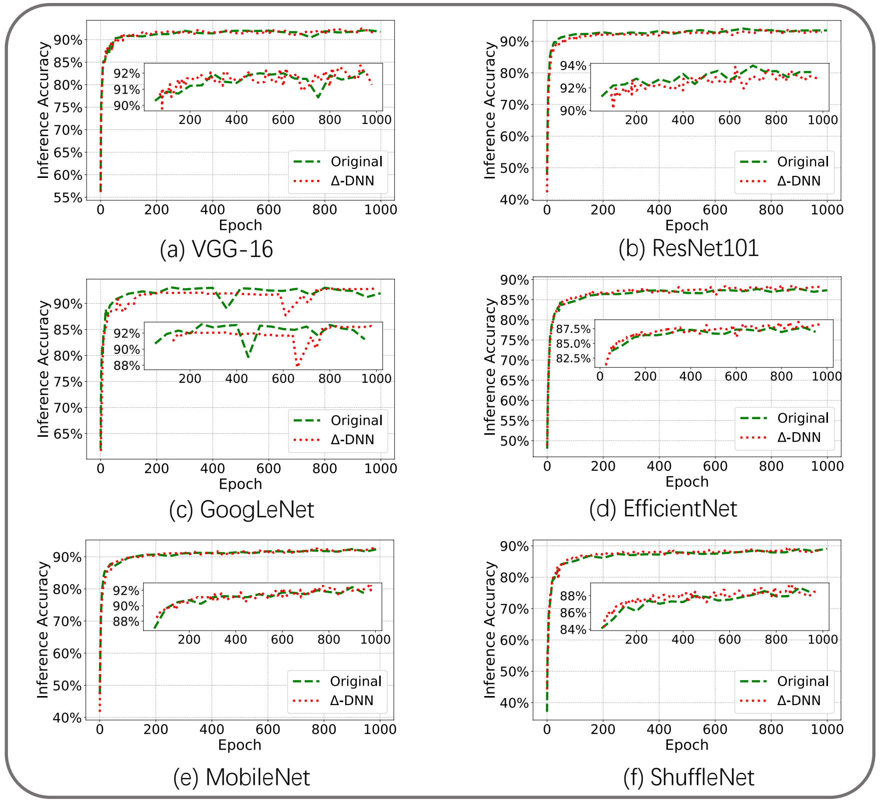
Case 2: Saving Storage Space

- Using **Delta-DNN** to save storage space

Storage space consumption before and after using Delta-DNN

Network	Epochs	Total Size		Comp. Ratio	Accuracy Loss
		Original	Δ-DNN		
VGG-16	95	5.21 GB	693 MB	7.702	-0.0003%
ResNet101	89	14.1 GB	2.18 GB	6.488	-0.0015%
GoogLeNet	83	1.91 GB	191 MB	10.259	-0.0009%
EfficientNet	110	1.21 GB	208 MB	5.946	0.0001%
MobileNet	115	1.00 GB	140 MB	7.311	-0.0004%
ShuffleNet	113	391 MB	73 MB	5.302	0

Delta-DNN can effectively reduce the storage size by **5x~10x**, while the average inference accuracy loss is **negligible**.



Inference accuracy before and after using Delta-DNN

Conclusion and future work

- **Delta-DNN**
  - A novel delta compression framework for DNNs, called Delta-DNN, which can significantly reduce the size of DNNs by exploiting the floats similarity existing in neighboring networks in training.
  - Our evaluation results on six popular DNNs suggest Delta-DNN achieves 2x~10x higher compression ratio compared with Zstd, LZMA, and SZ approaches.
  - Controllable between inference accuracy and compression ratio.

- **Future work**
  - Evaluate our proposed Delta-DNN on more neural networks and more datasets.
  - Further improve the compression ratio combining other model compression techniques.
  - Extend Delta-DNN framework into more scenarios, like deep learning in the distributed systems.

Thank you!

哈爾濱工業大學（深圳）
HARBIN INSTITUTE OF TECHNOLOGY, SHENZHEN