A Universal Compression Algorithm for Deep Neural Networks

Fraunhofer Heinrich Hertz Institute (HHI)
Machine Learning Group
Dr. Wojciech Samek
Deep Learning "Revolution"

Ingredients for the success
1. Large volumes of data
2. Large (Deep) Models
3. Large Computing Power

Images, Text, Speech, Games ...

Fraunhofer Heinrich Hertz Institute
A Universal Compression Algorithm for Deep Neural Networks
Complexity of DNN is Growing
Large Computational Resources Needed

Common carbon footprint benchmarks

in lbs of CO2 equivalent

<table>
<thead>
<tr>
<th>Category</th>
<th>CO2 Equivalent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Roundtrip flight b/w NY and SF (1 passenger)</td>
<td>1,984</td>
</tr>
<tr>
<td>Human life (avg. 1 year)</td>
<td>11,023</td>
</tr>
<tr>
<td>American life (avg. 1 year)</td>
<td>36,156</td>
</tr>
<tr>
<td>US car including fuel (avg. 1 lifetime)</td>
<td>126,000</td>
</tr>
<tr>
<td>Transformer (213M parameters) w/ neural architecture search</td>
<td>626,155</td>
</tr>
</tbody>
</table>

Chart: MIT Technology Review • Source: Strubell et al. • Created with Datawrapper
Processing at the “Edge”

Wired
(Feb 2018)

Cameras and radar generate ~6 gigabytes of data every 30 seconds.

Self-driving car prototypes use approximately 2,500 Watts of computing power.

Generates wasted heat and some prototypes need water-cooling!

[slide from V. Sze]
Processing at the “Edge”

On-device deep learning

Distributed Data & Privacy

Latency & bandwidth constraints
We need techniques to reduce the computational complexity (i.e., storage, memory, energy, runtime, communication overhead)
MPEG-7 Part 17

Standard on "Compression of Neural Networks for Multimedia Content Description and Analysis"
Outline of this talk

1. Background: Quantization & Encoding
2. DeepCABAC
3. Compression in Federated Learning
Background: Quantization & Encoding
Source Coding

Represent a signal with the minimum number of (binary) symbols without exceeding an “acceptable level of distortion”.

Lossy Step + Lossless Step
Source Coding

**Goal**: Minimize the rate-distortion objective:

\[
C^* = \arg \min_C E_{P(w)} [D(w, q) + \lambda L_C(b)]
\]

where \( b = (B \circ Q)(w) \) and \( q = (Q^{-1} \circ Q)(w) \).
Lossless Coding

The minimum information required to fully represent a sample \( w \) that has probability \( P(w) \) is of \(-P \log_2 P(w)\) bits (Shannon).

Challenges:
- Decoder does not know \( P(w) \)
- \( P(w) \) may be non-stationary
Lossless Coding: Desired Properties

**Universality:** The code should have a mechanism that allows it to adapt its probability model to a wide range of different types of input distributions, in a sample-efficient manner.

**Minimal redundancy:** The code should produce binary representations of minimal redundancy with regards to its probability estimate.

**High efficiency:** The code should have high coding efficiency, meaning that encoding/decoding should have high throughput.
Video Coding Standards

A Universal Compression Algorithm for Deep Neural Networks
NN Coding

In NN coding, things are more complicated:

- complex distortion term (non-linear accumulation of errors)
- no clear structure in NN weights (e.g. in video high correlation between frames and neighboring pixels)
- more flexibility (e.g. fine-tuning, sparsification, structural changes)
A Universal Compression Algorithm for Deep Neural Networks

NN Coding

is developing a standard on "Compression of Neural Networks for Multimedia Content Description and Analysis"
Lossy Coding

Finding the optimal code is in most cases NP-hard

$$C^* = \arg \min_C \mathbb{E}_{P(w)} \left[ D(w, q) + \lambda L_C(b) \right]$$

Idea: Fix the binarization map B by selecting a particular (universal) lossless code. Then just need to find a scalar quantizer

$$\left(Q, Q^{-1}\right)^* = \arg \min_{(Q, Q^{-1})} \mathbb{E}_{P(w_j)} \left[ D(w_j, q_j) + \lambda L_Q(b_j) \right]$$
A Universal Compression Algorithm for Deep Neural Networks

From Source Coding to NN Coding

\[ (Q, Q^{-1})^* = \arg \min_{(Q, Q^{-1})} \sum_{(x, y) \in \mathcal{D}} \mathcal{L}(y'', y') + \lambda L_Q(b) \]

\[ y' \sim P(y' | x, w) \]

\[ y'' \sim P(y'' | x, q) \]
From Source Coding to NN Coding

\[(Q, Q^{-1})^* = \arg \min_{(Q, Q^{-1})} \sum_{(x, y) \in \mathcal{D}} \mathcal{L}(y'', y') + \lambda L_Q(b)\]

\[\Rightarrow\]

\[(Q, Q^{-1})^* = \arg \min_{(Q, Q^{-1})} \sum_{(x, y) \in \mathcal{D}} D_{KL}(y'' || y') + \lambda L_Q(b)\]

Use KL-divergence as distortion measure
From Source Coding to NN Coding

\[(Q, Q^{-1})^* = \arg \min_{(Q, Q^{-1})} \sum_{(x, y) \in D} \mathcal{L}(y'', y') + \lambda L_Q(b)\]

\[(Q, Q^{-1})^* = \arg \min_{(Q, Q^{-1})} \sum_{(x, y) \in D} D_{KL}(y'' || y') + \lambda L_Q(b)\]

\[(Q, Q^{-1})^* = \min_{(Q, Q^{-1})} (q - w) F(q - w)^T + \lambda L_Q(b)\]

If the output distributions do not differ too much, we can approximate KL with the Fisher Information Matrix (FIM)
From Source Coding to NN Coding

\[
(Q, Q^{-1})^* = \arg \min_{(Q, Q^{-1})} \sum_{(x, y) \in \mathcal{D}} \mathcal{L}(y''; y') + \lambda L_Q(b)
\]

\[
(Q, Q^{-1})^* = \arg \min_{(Q, Q^{-1})} \sum_{(x, y) \in \mathcal{D}} D_{KL}(y'' || y') + \lambda L_Q(b)
\]

\[
(Q, Q^{-1})^* = \min_{(Q, Q^{-1})} (q - w)F(q - w)^T + \lambda L_Q(b)
\]

\[
(Q, Q^{-1})^* = \arg \min_{(Q, Q^{-1})} F_i(q_i - w_i)^2 + \lambda L_Q(b)
\]

Approximate FIM by only its diagonal elements
DeepCABAC: Weighted RD-based Quantization + CABAC

Parametrize each weight parameter as Gaussian. $F_i = 1/\sigma_i$

DeepCABAC-v1

$\arg \min_{(Q,Q^{-1})} \sum_i F_i(q_i - w_i)^2 + \lambda L_Q(b)$

$q_k = \Delta I_k$

https://github.com/fraunhoferhhi/DeepCABAC

[Wiedemann et al. 2019, ODML-CDNNR]
best paper award

A Universal Compression Algorithm for Deep Neural Networks
DeepCABAC: Uniform Quantization + CABAC

\[ (Q, Q^{-1})^* = \arg \min_{(Q, Q^{-1})} (q_i - w_i)^2 \]

\[ q_k = \Delta I_k \]

DeepCABAC-v3

\[ F_j = 1 \quad \forall j \quad \lambda = 0 \]

https://github.com/fraunhoferhhi/DeepCABAC

[Wiedemann et al. 2019, ODML-CDNNR]
best paper award
Properties of CABAC

Binarization: represents each unique input value as a sequence of binary decisions.
Context modelling: probability model for each decision, which is updated on-the-fly by the local statistics of the data -> universality.
Arithmetic coding: arithmetic coding for each bit -> minimal redundancy + high efficiency
<table>
<thead>
<tr>
<th>Sparse Models (sparsity [%])</th>
<th>Org. Acc. Top1 [%]</th>
<th>Os_size [MB]</th>
<th>DeepCABAC (acc. [%])</th>
</tr>
</thead>
<tbody>
<tr>
<td>VGG16 (9.85)</td>
<td>69.43</td>
<td>553.43</td>
<td>1.57 (69.43)</td>
</tr>
<tr>
<td>ResNet50 (74.12)</td>
<td>74.09</td>
<td>102.23</td>
<td>4.74 (73.65)</td>
</tr>
<tr>
<td>Small-VGG16 (7.57)</td>
<td>91.35</td>
<td>60.01</td>
<td>1.6 (91.00)</td>
</tr>
<tr>
<td>LeNet5 (1.90)</td>
<td>99.22</td>
<td>1.72</td>
<td>0.72 (99.16)</td>
</tr>
</tbody>
</table>
### Some Results

<table>
<thead>
<tr>
<th>Sparse Models (sparsity [%])</th>
<th>Org. Acc. Top1 [%]</th>
<th>Os_size [MB]</th>
<th>DeepCABAC (acc. [%])</th>
</tr>
</thead>
<tbody>
<tr>
<td>VGG16</td>
<td>69.43</td>
<td>553.43</td>
<td>1.57</td>
</tr>
</tbody>
</table>

VGG16 **553.4MB** \(\rightarrow\) **8.7MB** at an acc. **69.43%**

ResNet50 **102.2MB** \(\rightarrow\) **4.85MB** at an acc. **73.65%**

<table>
<thead>
<tr>
<th>Small-VGG16 (7.57)</th>
<th>91.35</th>
<th>60.01</th>
<th>1.6 (91.00)</th>
</tr>
</thead>
<tbody>
<tr>
<td>LeNet5 (1.90)</td>
<td>99.22</td>
<td>1.72</td>
<td>0.72 (99.16)</td>
</tr>
</tbody>
</table>
Compression in Federated Learning
Federated Learning

Download

Server

Expensive communication!

Upload

Server

Expensive communication!

Data

Data

Data

Data
Federated Learning
Plug & Play compression by DeepCABAC

[Neumann et al. 2020, IEEE ICIP]
Federated Learning

Total Communication = [#Communication Rounds] x [#Parameters] x [Avg. Codeword length]

Case Study: VGG16 on ImageNet

- Number of Iterations until Convergence: 900,000
- Number of Parameters: 138,000,000
- Bits per Parameter: 32

→ Total Communication = 496.8 Terabyte (Upload+Download)
Federated Learning

Total Communication = [#Communication Rounds] × [#Parameters] × [Avg. Codeword length]

Compression Methods
- Communication Delay
- Lossy Compression: Unbiased
- Lossy Compression: Biased
- Efficient Encoding
Federated Learning

Sattler, et al. "Sparse binary compression: Towards distributed deep learning with minimal communication." 2019 International Joint Conference on Neural Networks (IJCNN).
Federated Learning

Validation Error vs # transferred Bits (log-log)

[Sattler et al. 2019, IEEE IJCNN]
Federated Learning

Reduction in communication from 125 TB to 3.35 GB for every participating client.

[Sattler et al. 2019, IEEE IJCNN]
### Federated Learning

<table>
<thead>
<tr>
<th>Compression Method</th>
<th>Method</th>
<th>Baseline</th>
<th>DGC $^3$</th>
<th>Fed. Avg.$^4$</th>
<th>SBC (1)</th>
<th>SBC (2)</th>
<th>SBC (3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>LeNet5-Caffe @MNIST</td>
<td>Accuracy</td>
<td>0.9946</td>
<td>0.994</td>
<td>0.994</td>
<td>0.994</td>
<td>0.994</td>
<td>0.991</td>
</tr>
<tr>
<td></td>
<td>Compression</td>
<td>$\times 1$</td>
<td>$\times 718$</td>
<td>$\times 500$</td>
<td>$\times 2071$</td>
<td>$\times 3166$</td>
<td>$\times 24935$</td>
</tr>
<tr>
<td>ResNet18 @CIFAR10</td>
<td>Accuracy</td>
<td>0.946</td>
<td>0.9383</td>
<td>0.9279</td>
<td>0.9422</td>
<td>0.9435</td>
<td>0.9219</td>
</tr>
<tr>
<td></td>
<td>Compression</td>
<td>$\times 1$</td>
<td>$\times 768$</td>
<td>$\times 1000$</td>
<td>$\times 2369$</td>
<td>$\times 3491$</td>
<td>$\times 31664$</td>
</tr>
<tr>
<td>ResNet34 @CIFAR100</td>
<td>Accuracy</td>
<td>0.773</td>
<td>0.767</td>
<td>0.7316</td>
<td>0.767</td>
<td>0.7655</td>
<td>0.701</td>
</tr>
<tr>
<td></td>
<td>Compression</td>
<td>$\times 1$</td>
<td>$\times 718$</td>
<td>$\times 1000$</td>
<td>$\times 2370$</td>
<td>$\times 3166$</td>
<td>$\times 31664$</td>
</tr>
<tr>
<td>ResNet50 @ImageNet</td>
<td>Accuracy</td>
<td>0.737</td>
<td>0.739</td>
<td>0.724</td>
<td>0.735</td>
<td>0.737</td>
<td>0.728</td>
</tr>
<tr>
<td></td>
<td>Compression</td>
<td>$\times 1$</td>
<td>$\times 601$</td>
<td>$\times 1000$</td>
<td>$\times 2569$</td>
<td>$\times 3531$</td>
<td>$\times 37208$</td>
</tr>
<tr>
<td>WordLSTM @PTB</td>
<td>Perplexity</td>
<td>76.02</td>
<td>75.98</td>
<td>76.37</td>
<td>77.73</td>
<td>78.19</td>
<td>77.57</td>
</tr>
<tr>
<td></td>
<td>Compression</td>
<td>$\times 1$</td>
<td>$\times 719$</td>
<td>$\times 1000$</td>
<td>$\times 2371$</td>
<td>$\times 3165$</td>
<td>$\times 31658$</td>
</tr>
<tr>
<td>WordLSTM* @WIKI</td>
<td>Perplexity</td>
<td>101.5</td>
<td>102.318</td>
<td>131.51</td>
<td>103.95</td>
<td>103.95</td>
<td>104.62</td>
</tr>
<tr>
<td></td>
<td>Compression</td>
<td>$\times 1$</td>
<td>$\times 719$</td>
<td>$\times 1000$</td>
<td>$\times 2371$</td>
<td>$\times 3165$</td>
<td>$\times 31657$</td>
</tr>
</tbody>
</table>
Next Standard?

INTERNATIONAL ORGANISATION FOR STANDARDISATION
ORGANISATION INTERNATIONALE DE NORMALISATION
ISO/IEC JTC1/SC29/WG11
CODING OF MOVING PICTURES AND AUDIO

ISO/IEC JTC1/SC29/WG11/N19228
April 2020, Alpbach, AT

Source  Video
Status    Approved
Title     Call for Incremental NNR Test Materials
Conclusion

Efficiency in storage, memory, energy, runtime, communication..
DeepCABAC based on established compression technology
Different options (e.g. fine-tuning, structural changes, NAS)
Hardware co-design is crucial
MPEG standardization is moving forward
References

Neural Network Compression

http://dx.doi.org/10.1109/JSTSP.2020.2969554

https://arxiv.org/abs/1912.08881


http://dx.doi.org/10.1109/IJCNN.2019.8852119
References

Efficient Deep Learning

http://dx.doi.org/10.1109/TNNLS.2019.2910073

https://dx.doi.org/10.1109/CVPRW50498.2020.00368

https://dx.doi.org/10.1109/CVPRW50498.2020.00369
References

F Federated Learning

https://www.itu.int/en/journal/2020/001/Pages/07.aspx


http://dx.doi.org/10.1109/TNNLS.2019.2944481

References

Federated Learning

http://dx.doi.org/10.1109/ICASSP40776.2020.9054676

http://dx.doi.org/10.1109/IJCNN.2019.8852172

Slides and Papers available at

www.federated-ml.org
Efficient Neural Network Representation
Efficient Representation

**Goal:** Find a representation for the neural network, which is:

1) efficient with regard to storage / memory
2) efficient with regard to inference complexity
3) efficient with regard to energy consumption
Fixed-Point Neural Nets

Reduce Precision

32-bit float: 0110010100000000101000000000100

8-bit fixed: 01100110

Binary: 0

No guarantee that DNN algorithm designer will use a given approach. Need flexible hardware!

Advantages

- Arithmetic with lower bit-depth is faster

- From 32-bits to 8-bits, we get (almost) 4x reduction in memory

- Lower bit-widths we can squeeze more data into caches/registers

- Floating point arithmetic not always be supported on microcontrollers on some ultra low-power embedded devices
Goal: Find a representation for the weight matrices of a neural network, which is:
1) efficient with regard to storage
2) efficient with regard to inference complexity
Efficient Representation Format

$$M = \begin{pmatrix}
0 & 3 & 0 & 2 & 4 & 0 & 0 & 2 & 3 & 4 & 0 & 4 \\
4 & 4 & 0 & 0 & 0 & 4 & 0 & 0 & 4 & 4 & 0 & 4 \\
4 & 0 & 3 & 4 & 0 & 0 & 0 & 4 & 0 & 2 & 0 & 0 \\
0 & 0 & 0 & 4 & 4 & 4 & 0 & 3 & 4 & 4 & 0 & 0 \\
0 & 4 & 4 & 0 & 0 & 4 & 0 & 4 & 0 & 0 & 0 & 0 \\
0 & 4 & 4 & 0 & 0 & 4 & 0 & 4 & 0 & 0 & 0 & 0
\end{pmatrix}$$

Storage requirements: 60 entries
dense format
Matrix Formats: Dense Format

\[
M = \begin{pmatrix}
0 & 3 & 0 & 2 & 4 & 0 & 0 & 2 & 3 & 4 & 0 & 4 \\
4 & 4 & 0 & 0 & 0 & 4 & 0 & 0 & 4 & 4 & 0 & 4 \\
4 & 0 & 3 & 4 & 0 & 0 & 4 & 0 & 2 & 0 & 0 \\
0 & 0 & 0 & 4 & 4 & 4 & 0 & 3 & 4 & 4 & 0 & 0 \\
0 & 4 & 4 & 0 & 0 & 4 & 0 & 4 & 0 & 0 & 0 & 0 \\
\end{pmatrix}
\]

\[
* \begin{pmatrix}
a_1 \\
a_2 \\
\cdot \\
\cdot \\
a_{12}
\end{pmatrix}
\]

Storage requirements: 60 entries

Scalar product (second row \(M\), vector \(a\)):
- 24 load
- 12 multiply
- 11 add
- 1 write operations
Compressed neural networks have *weight sharing* property.

**Trick:**

\[
    z_i^l = \sum_{j} w_{ij}^l a_{j}^{l-1} \quad \rightarrow \quad z_i^l = \sum_{k} w_k^l \sum_{j \in J_{ik}^l} a_j^{l-1}
\]
Matrix Formats: Compressed Entropy Row Format

\[\Omega : [0, 4, 3, 2]\]
\[\text{colI} : [4, 9, 11, 1, 8, 3, 7, 0, 1, 5, 8, 9, 11, 0, 3, 7, 2, 9, 3, 4, 5, 8, 9, 7, 1, 2, 5, 7]\]
\[\OmegaPtr : [0, 3, 5, 7, 13, 16, 17, 18, 23, 24, 28]\]
\[\text{rowPtr} : [0, 3, 4, 7, 9, 10]\]

CER format

Storage requirements: 49 entries

Scalar product (second row \(M\), vector \(a\)):
- 17 load
- 1 multiply
- 5 add
- 1 write operations

\[4(a_1 + a_2 + a_6 + a_9 + a_{10} + a_{12})\]

[Wiedemann et al. 2020, IEEE TNNLS]
A Universal Compression Algorithm for Deep Neural Networks

Storage Efficiency

\[ H = - \log_2 p_0 \]

\[ \log_2 K \]

Spike-and-Slab distributions

Dense

CER/CSER

Sparse

Storage complexity

\[ 10^9 \]

Operations complexity

\[ 10^8 \]

Time complexity

\[ 10^11 \]

Energy complexity

\[ 10^11 \]

Wiedemann et al. 2020, IEEE TNNLS

Fraunhofer Heinrich Hertz Institute