WHAT WOULD SHANNON DO?
BAYESIAN COMPRESSION FOR DL

KAREN ULLRICH
UNIVERSITY OF AMSTERDAM

DEEP LEARNING AND REINFORCEMENT LEARNING
SUMMER SCHOOL MONTREAL 2017
Motivation
Motivation

- 1 Wh costs 0.0225 cent
- running a Titan X for 1h: 5.625 cent

- facebook has 1.86 billion active users
- VGG takes ~147ms/16 predictions

- making one prediction for all users costs 20 k€
Motivation - Summary

- mobile devices have limited hardware
- energy costs for predictions
- bandwidth transmitting models
- speeding up inference for real time processing
- relation to privacy
Practical view on compression

A : Sparsity learning

- Structured Pruning:
  - CR: \( \frac{|w|}{2|w\neq 0|} \)

- (Unstructured) Pruning

- Structured Pruning:
  - CR: \( \frac{|w|}{|w\neq 0|} \)

\[
A = [\begin{bmatrix}
1 & 2 & 3 & 0 & 1 & 2 \\
0 & 1 & 2 & 3 & \vdots & 1
\end{bmatrix}] \]

IC = [0 1 3 0 1 2]
IR = [3, 0, 2, 3, ..., 1]
Practical view on compression

B : Bit per weight reduction

- precision quantisation
- CR: 32/10 = 3
- PRO: fast inference
- CON: savings is not too big

- Set quantisation by clustering
- CR: 32/4 = 8
- PRO: extreme compressible with e.g. further Hoffman encoding
- CON: inference?
Practical view on compression
Summary - Properties

<table>
<thead>
<tr>
<th></th>
<th>Set quantisation</th>
<th>Bit quantisation</th>
</tr>
</thead>
</table>
| **Unstructured pruning** | - highest compression  
- flop and energy savings moderate                                                                                                               |                                                                                                                                                   |
| **Structured pruning**   |                                                                                                                                                  | - lowest expected compression  
- BUT will save considerable amount of flops and thus energy                                                                                   |
### Practical view on compression

#### Summary - Applications

<table>
<thead>
<tr>
<th>Unstructured pruning</th>
<th>Structured pruning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set quantisation</td>
<td></td>
</tr>
<tr>
<td>Bit quantisation</td>
<td></td>
</tr>
</tbody>
</table>

**Unstructured pruning**
- “ZIP”-format for NN
- transmitting via limited channels
- save millions of nets efficiently

**Structured pruning**
- inference at scale
- real time predictions
- hardware limited devices
Variational lower bound

\[
\log p(D) \geq \mathcal{L}(q(w), w)) = \mathbb{E}_{q(w)}[\log \frac{p(D,w)}{q(w)}] \\
= \mathbb{E}_{q(w)}[\log p(D|w))] - KL(q(w)||p(w))
\]

MDL principle and Variational Learning

The best model is the one that compresses the data best. There are two costs, one for transmitting a model and one for reporting the data misfit.

Jorma Rissanen, 1978
Variational lower bound

\[
\log p(\mathcal{D}) \geq \mathcal{L}(q(w), w)) = \mathbb{E}_{q(w)}[\log \frac{p(\mathcal{D}, w)}{q(w)}] = \mathbb{E}_{q(w)}[\log p(\mathcal{D}|w)] - KL(q(w)\|p(w))
\]

transmitting
transmitting
transmitting

data misfit
the model

Variational lower bound

\[
\log p(D) \geq \mathcal{L}(q(w), w)) = \mathbb{E}_{q(w)}[\log \frac{p(D, w)}{q(w)}]
\]

\[
= \mathbb{E}_{q(w)}[\log p(D|w)] - KL(q(w)||p(w))
\]

\[
p(D|w) = p(T|X, w) = \prod_{n=1}^{N} \mathcal{N}(t_n|x_n, w)
\]

\[
KL(q(w)||p(w)) = \mathbb{E}_{q(w)}[-\log p(w)] - H(q(w))
\]

\[
H(q(w)) = -\int_{\Omega} q(w) \log q(w) \, dw = -\int_{\mathbb{R}^I} \mathcal{N}(w|0, \sigma I) \log \mathcal{N}(w|0, \sigma I) = \left[\log(2\pi e\sigma^2)\right]^I.
\]

# Practical view on compression

## Summary - Properties

<table>
<thead>
<tr>
<th>Unstructured pruning</th>
<th>Bit quantisation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set quantisation</td>
<td>- lowest expected compression</td>
</tr>
<tr>
<td></td>
<td>- BUT will save considerable amount of flops and thus energy</td>
</tr>
<tr>
<td>Unstructured pruning</td>
<td>- highest compression</td>
</tr>
<tr>
<td></td>
<td>- flop and energy savings moderate</td>
</tr>
<tr>
<td>Structured pruning</td>
<td></td>
</tr>
</tbody>
</table>

**KAREN ULLRICH, MAR 2017**
Solution: train a neural network with **gaussian mixture model prior**

\[ q(w) = \prod q(w_i) = \delta(w_i | \mu_i) \]

\[ p(w) = \prod_i \sum_j \pi_j \mathcal{N}(w_i | \mu_j, \sigma_j^2). \]

- Pruning by setting one component to zero with high mixing proportion

**Soft weight-sharing for NN compression**

KAREN ULLRICH, EDWARD MEED & MAX WELLING  
ICLR 2017

| Model       | Method                | Top-1 Error[\%] | $\Delta$ [\%] | $|W|[10^6]$ | $\frac{|W_{\neq 0}|}{|W|}$ [%] | CR |
|-------------|-----------------------|-----------------|---------------|-------------|-------------------------------|----|
| LeNet-300-100 | Han et al. (2015a)    | 1.64 $\rightarrow$ 1.58 | 0.06          | 0.2         | 8.0                           | 40 |
|             | Guo et al. (2016)     | 2.28 $\rightarrow$ 1.99 | -0.29         | 0.2         | 1.8                           | 56 |
|             | Ours                  | 1.89 $\rightarrow$ 1.94 | -0.05         | 0.2         | 4.3                           | 64 |
| LeNet-5-Caffe| Han et al. (2015a)    | 0.80 $\rightarrow$ 0.74 | -0.06         | 0.4         | 8.0                           | 39 |
|             | Guo et al. (2016)     | 0.91 $\rightarrow$ 0.91 | 0.00          | 0.4         | 0.9                           | 108|
|             | Ours                  | 0.88 $\rightarrow$ 0.97 | 0.09          | 0.4         | 0.5                           | 162|
| ResNet (light) | Ours                 | 6.48 $\rightarrow$ 8.50 | 2.02          | 2.7         | 6.6                           | 45 |
## Practical view on compression

### Summary - Properties

<table>
<thead>
<tr>
<th>Unstructured pruning</th>
<th>Bit quantisation</th>
</tr>
</thead>
<tbody>
<tr>
<td>- highest compression</td>
<td>- lowest expected compression</td>
</tr>
<tr>
<td>- flop and energy savings moderate</td>
<td>- BUT will save considerable amount of flops and thus energy</td>
</tr>
</tbody>
</table>
Bayesian Compression for Deep Learning

CHRISTOS LOUIZOS, KAREN ULLRICH & MAX WELLING
UNDER SUBMISSION NIPS 2017

- **Idea:** use dropout to learn architecture
- the variational version of dropout learns the dropout rate

- **Solution:** Learn dropout rate for each weight structure, when weights have a high dropout rate we can safely ignore them
- uncertainty in left over weights to compute bit precision

Bayesian Compression for Deep Learning

CHRISTOS LOUIZOS, KAREN ULLRICH & MAX WELLING
UNDER SUBMISSION NIPS 2017

\[ q(z) = \prod q(z_i) = \mathcal{N}(z_i | \mu^z_i, \alpha_i) \]

\[ q(w|z) = \prod q(w_i|z_i) = \mathcal{N}(w_i | z_i \mu_i, z_i^2 \sigma^2_i) \]

force high dropout rates

push to zero for high dropout rates
Bayesian Compression for Deep Learning

CHRISTOS LOUIZOS, KAREN ULLRICH & MAX WELLING
UNDER SUBMISSION NIPS 2017

\[ q(z) = \prod q(z_i) = \mathcal{N}(z_i | \mu_i, \alpha_i) \]

\[ q(w | z) = \prod q(w_i | z_i) = \mathcal{N}(w_i | z_i \mu_i, z_i^2 \sigma_i^2) \]

\[ p(w) = \int p(z) p(w | z) dz \]

\[ p(w) \propto \int \frac{1}{|z|} \mathcal{N}(w | 0, z^2) dz = \frac{1}{|w|} \]
## Bayesian Compression for Deep Learning

**CHRISTOS LOUIZOS, KAREN ULLRICH & MAX WELLING**

UNDER SUBMISSION NIPS 2017

<table>
<thead>
<tr>
<th>Network &amp; size</th>
<th>Method</th>
<th>Pruned architecture</th>
<th>Bit-precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>LeNet-300-100</td>
<td>Sparse VD</td>
<td>512-114-72</td>
<td>8-11-14</td>
</tr>
<tr>
<td>784-300-100</td>
<td>BC-GNJ</td>
<td>278-98-13</td>
<td>8-9-14</td>
</tr>
<tr>
<td></td>
<td>BC-GHS</td>
<td>311-86-14</td>
<td>13-11-10</td>
</tr>
<tr>
<td>LeNet-5-Caffe</td>
<td>Sparse VD</td>
<td>14-19-242-131</td>
<td>13-10-8-12</td>
</tr>
<tr>
<td></td>
<td>GD</td>
<td>7-13-208-16</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>GL</td>
<td>3-12-192-500</td>
<td>-</td>
</tr>
<tr>
<td>20-50-800-500</td>
<td>BC-GNJ</td>
<td>8-13-88-13</td>
<td>18-10-7-9</td>
</tr>
<tr>
<td></td>
<td>BC-GHS</td>
<td>5-10-76-16</td>
<td>10-10-14-13</td>
</tr>
<tr>
<td>VGG</td>
<td>BC-GNJ</td>
<td>63-64-128-128-245-155-63-26-24-20-14-12-11-11-15</td>
<td>10-10-10-10-8-8-8-5-5-5-5-5-6-7-11</td>
</tr>
<tr>
<td>(2× 64)-(2× 128)-</td>
<td>BC-GHS</td>
<td>51-62-125-128-228-129-38-13-9-6-5-6-6-6-6-20</td>
<td>11-12-9-14-10-8-5-5-6-6-8-11-17-10</td>
</tr>
<tr>
<td>-(3× 256)-(8× 512)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Bayesian Compression for Deep Learning

CHRISTOS LOUIZOS, KAREN ULLRICH & MAX WELLING

UNDER SUBMISSION NIPS 2017

<table>
<thead>
<tr>
<th>Model</th>
<th>Original Error %</th>
<th>Method</th>
<th>Compression Rates (Error %)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>$</td>
<td>w</td>
</tr>
<tr>
<td>LeNet-300-100</td>
<td>1.6</td>
<td>DC</td>
<td>8.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DNS</td>
<td>1.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SWS</td>
<td>4.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sparse VD</td>
<td>2.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BC-GNJ</td>
<td>10.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BC-GHS</td>
<td>10.6</td>
</tr>
<tr>
<td>LeNet-5-Caffe</td>
<td>0.9</td>
<td>DC</td>
<td>8.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DNS</td>
<td>0.9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SWS</td>
<td>0.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sparse VD</td>
<td>0.7</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BC-GNJ</td>
<td>0.9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BC-GHS</td>
<td>0.6</td>
</tr>
<tr>
<td>VGG</td>
<td>8.4</td>
<td>BC-GNJ</td>
<td>6.7</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BC-GHS</td>
<td>5.5</td>
</tr>
</tbody>
</table>
Bayesian Compression for Deep Learning

CHRISTOS LOUIZOS, KAREN ULLRICH & MAX WELLING
UNDER SUBMISSION NIPS 2017

![Graph showing average time and energy consumption per forward-pass for different devices and methods.](image-url)
Warning: Don’t be too enthusiastic!

These algorithms are merely proposals, little can be realised by common frameworks today.

- Architecture pruning 😊
- Sparse matrix support 😞 (partially in big frameworks)
- Reduced bit precision 😞 (NVIDIA is starting)
- Clustering 😞
Thank you for your attention. Any questions?

KARENULLRICH.INFO
@KAREN_ULLRICH