Computation Reuse in DNNs by Exploiting Input Similarity

Marc Riera, Jose Maria Arnau, Antonio González
Sequence Processing Applications

Speech Audio Signal

Time

Frequency
Sequence Processing Applications

- frame 0
- frame 1
- frame 2
- frame 3
- frame 4

Frequency

10 ms

Time

4/06/2018
Sequence Processing Applications

![Diagram showing sequence processing with DNN execution and frames over time.](image)
Sequence Processing Applications

- DNN Exec 1
- DNN Exec 2

- Frame 0
- Frame 1
- Frame 2
- Frame 3

Frequency

10 ms

Time

4/06/2018
ISCA 2018
Speech Recognition DNN executions to classify a sequence of audio frames in phonemes
## Benchmarks

<table>
<thead>
<tr>
<th>DNN Name</th>
<th>DNN Type</th>
<th>DNN Application</th>
<th>#Parameters</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kaldi</td>
<td>MLP</td>
<td>Acoustic Scoring</td>
<td>4.7M</td>
<td>89.04%</td>
</tr>
<tr>
<td>EESEN</td>
<td>RNN</td>
<td>Speech Recognition</td>
<td>11M</td>
<td>68.85%</td>
</tr>
<tr>
<td>C3D</td>
<td>CNN</td>
<td>Video Classification</td>
<td>78M</td>
<td>93.48%</td>
</tr>
<tr>
<td>AutoPilot</td>
<td>CNN</td>
<td>Self-Driving Cars</td>
<td>1.6M</td>
<td>99.63%</td>
</tr>
</tbody>
</table>
Input Similarity

<table>
<thead>
<tr>
<th>System</th>
<th>Input Similarity (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kaldi</td>
<td>45%</td>
</tr>
<tr>
<td>C3D</td>
<td>69%</td>
</tr>
<tr>
<td>Autopilot</td>
<td>77%</td>
</tr>
<tr>
<td>EESEN</td>
<td>52%</td>
</tr>
<tr>
<td>Average</td>
<td>61%</td>
</tr>
</tbody>
</table>
Exploiting Temporal Similarity Example

**Baseline**

\[
O^i = I_0^i w_0 + I_1^i w_1 + I_2^i w_2 + b
\]

**Frame i**

- \(I_0^i\) to \(N\) with weight \(w_0\)
- \(I_1^i\) to \(N\) with weight \(w_1\)
- \(I_2^i\) to \(N\) with weight \(w_2\)

\(O^i = I_0^i w_0 + I_1^i w_1 + I_2^i w_2 + b\)

**Frame i+1**

- \(I_0^{i+1}\) to \(N\) with weight \(w_0\)
- \(I_1^{i+1}\) to \(N\) with weight \(w_1\)
- \(I_2^{i+1}\) to \(N\) with weight \(w_2\)

\(O^{i+1} = I_0^{i+1} w_0 + I_1^{i+1} w_1 + I_2^{i+1} w_2 + b\)
Exploiting Temporal Similarity Example

Proposal

Frame $i$

$O^i = I_0^i w_0 + I_1^i w_1 + I_2^i w_2 + b$

Frame $i+1$

$O^{i+1} = O^i + (I_2^{i+1} - I_2^i) w_2$

Number of computations before = 6
Number of computations after = 2

Note: Subtraction of the inputs is almost negligible since its performed once per input
Computation Reuse

Kaldi: 53%
C3D: 74%
Autopilot: 79%
EESEN: 55%
Average: 66%
DNN Processing Unit

Weights Buffer
- Bank 1
- Bank...
- Bank N

I/O Buffer

Data Master

Compute Engine
- FP MUL Array
- FP ADD/SUB Array

Control Unit
- REC
- SQRT
- EXP
- MAX
- MIN

Tile
FC Execution in the Reuse Accelerator (1)

Weights Buffer

weights layer L

Bank 0

Bank 1
FC Execution in the Reuse Accelerator (2)

I/O Buffer

Current Inputs layer L

Bank 0

Current Inputs layer L

Bank 1

Previous Inputs

Outputs layer L

$X_1, X_2, X_3, X_4, \ldots, X_M$

$idx_1, idx_2, \ldots, idx_M$

$Z_1, Z_2, Z_3, Z_4, \ldots, Z_M$
FC Execution in the Reuse Accelerator (3)

Start Layer L

More Inputs?

Yes

Read previous Input from I/O Buffer

Read current Input from I/O Buffer

Compare Inputs

Diff==0?

Yes

End Layer L

More neurons?

Yes

Write Outputs to I/O Buffer

Perform M Corrections

Read M Weights

No

No
Other Supported Layers

**Convolutional Neural Network (CNN)**

**Recurrent Neural Network (RNN)**
Evaluation Methodology

- Simulator to evaluate the performance and energy of the accelerator
- Design Compiler to obtain power and delay of logic modules
  - 28/32nm library from Synopsys and the DesignWare logic modules
- CACTI used for SRAM and eDRAM memories
- MICRON LPDDR4 for main Memory
- Accelerator Configuration:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Technology</td>
<td>32 nm</td>
</tr>
<tr>
<td>Frequency</td>
<td>500 MHz</td>
</tr>
<tr>
<td># of Tiles</td>
<td>4</td>
</tr>
<tr>
<td># of 32-bit multipliers</td>
<td>128</td>
</tr>
<tr>
<td># of 32-bit adders</td>
<td>128</td>
</tr>
<tr>
<td>Weights Buffer</td>
<td>36 MB</td>
</tr>
<tr>
<td>I/O Buffer Size</td>
<td>1152KB (Baseline) / 1280KB (Reuse)</td>
</tr>
</tbody>
</table>
Results: SpeedUp

- Kaldi: 1.9
- C3D: 4.7
- Autopilot: 5.2
- EESEN: 2.1
- Average: 3.5
Results: Energy Savings

Normalized Energy (%)

- Kaldi: 50%
- C3D: 49%
- Autopilot: 77%
- EESEN: 76%
- Average: 51%
- Average: 63%
Conclusions

• More than 60% of the inputs remain unmodified respect the previous execution

• Our proposed scheme checks which inputs have changed:
  • Unmodified inputs are ignored, avoiding computations and memory accesses
  • Modified inputs are used to correct the previous output of each neuron

• On average, 63% energy savings and 3.5x speedup

• Small area overhead of less than 1% mainly for additional storage
Computation Reuse in DNNs by Exploiting Input Similarity

Marc Riera, Jose Maria Arnau, Antonio González