DNNWEAVER: From High-Level Deep Network Models to FPGA Acceleration

Hardik Sharma  Jongse Park  Divya Mahajan  Emmanuel Amaro
Joon Kyung Kim  Chenkai Shao  Asit Mishra†  Hadi Esmaeilzadeh

Alternative Computing Technologies (ACT) Lab
Georgia Institute of Technology

†Intel Corporation

MICRO’16
Deep Neural Networks: A Move Towards Artificial Intelligence
Programmability is a First-Order Concern

Different applications require different DNNs!
FPGAs are Hard to Program!
Over 10,000 lines of code for DNN hardware templates in 14 months!

DNNs are Big, on-chip FPGA storage is small!
AlexNet is 119 MBytes whereas Arria 10 has 5 MB!
Bridging the Semantic Gap

High-level Model Specification

Translator

Macro Dataflow Graph

Design Planner

Execution Schedule

Accelerator Configuration

Design Weaver

Accelerator Core

Integrator

FPGA

Synthesizable Accelerator
1. Translator

High-Level DNN model – Caffe*

```
layer{
    name: Pool,
    type: POOLING,
    params{...}
}
layer{
    name: Conv,
    type: CONVOLUTION,
    params{...}
}
layer{
    name: Inner-Product,
    type: INNER_PRODUCT,
    params{...}
}
```

ISA: Abstracting DNNs

Abstract DNNs as a macro-dataflow graph
Template Architecture

Accelerator Core

PU₀   𝑃𝐸₀   Normalization
      𝑃𝐸₁   Pooling
      …     …
      𝑃𝐸_{n−1}   Activation

PU_{m−1}   𝑃𝐸₀   Normalization
           𝑃𝐸₁   Pooling
           …     …
           𝑃𝐸_{n−1}   Activation

Lookup \( f(x) \)
Processing Unit

Flexible #PEs

PU₀

Normalization

Pooling

Activation

Lookup \( f(x) \)

Exchangeable Components
Executing the Macro Dataflow Graph
Slicing the Dataflow Graph

Conv

Mult Mult ...
Mult Mult Mult Mult
Mult Mult Mult
Add Add Add

Convolution Node
Output

Sliced Output
2. Design Planner

Macro-Dataflow

ISA

Pool ➔ Conv ➔ Inner-Product

FPGA Specification

- # of DSPs
- # of BRAMs
- BRAM capacity
- # LUTs

Co-optimize Hardware and Execution Schedule
Using FPGA’s Programmability

Significant variations between different DNN models: # layers, Model Size, Operations, etc.

Specialize for each DNN model!
Co-optimization

![Graph showing speedup over Xeon E3 baseline for Overfeat and AlexNet with different PEs-per-PU.](image-url)

- Overfeat
- AlexNet

<table>
<thead>
<tr>
<th>PEs-per-PU</th>
<th>Speedup over Xeon E3 baseline</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>1.2</td>
</tr>
<tr>
<td>10</td>
<td>1.4</td>
</tr>
<tr>
<td>12</td>
<td>2.0</td>
</tr>
<tr>
<td>14</td>
<td>2.5</td>
</tr>
<tr>
<td>15</td>
<td>3.0</td>
</tr>
<tr>
<td>20</td>
<td>3.5</td>
</tr>
<tr>
<td>25</td>
<td>4.0</td>
</tr>
<tr>
<td>30</td>
<td>4.5</td>
</tr>
<tr>
<td>35</td>
<td>5.0</td>
</tr>
<tr>
<td>40</td>
<td>5.5</td>
</tr>
</tbody>
</table>
3. Design Weaver

Design Planner

Accelerator Configuration

Execution Schedule

Verilog

Decode

Accelerator Core

PU0

PE0 PE1 ... PE_{n-1}

Normalization

Pooling

Activation

Lookup f(x)

PU_{m-1}

PE0 PE1 ... PE_{n-1}

Normalization

Pooling

Activation

Lookup f(x)
4. Integrator

Memory Interface

Accelerator Core

PU_0

PE_0

Normalization

Pooling

Activation

Lookup f(x)

PU_{m-1}

PE_0

Normalization

Pooling

Activation

Lookup f(x)
## Benchmark DNNs

<table>
<thead>
<tr>
<th>Name</th>
<th># Layers</th>
<th>Model Size (MB)</th>
<th># Operations (Mops)</th>
<th>Lines of Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>VGG-16</td>
<td>36</td>
<td>324.0 MB</td>
<td>16362 MOps</td>
<td>347</td>
</tr>
<tr>
<td>OverFeat</td>
<td>16</td>
<td>278.0 MB</td>
<td>2798 MOps</td>
<td>196</td>
</tr>
<tr>
<td>VGG-CNN-S</td>
<td>19</td>
<td>196.0 MB</td>
<td>2666 MOps</td>
<td>200</td>
</tr>
<tr>
<td>AlexNet</td>
<td>20</td>
<td>119.0 MB</td>
<td>1147 MOps</td>
<td>278</td>
</tr>
<tr>
<td>Djinn</td>
<td>13</td>
<td>48.4 MB</td>
<td>25 MOps</td>
<td>105</td>
</tr>
<tr>
<td>NiN</td>
<td>28</td>
<td>14.5 MB</td>
<td>1106 MOps</td>
<td>516</td>
</tr>
<tr>
<td>LeNet</td>
<td>7</td>
<td>0.8 MB</td>
<td>2 MOps</td>
<td>128</td>
</tr>
<tr>
<td>CIFAR-10Full</td>
<td>12</td>
<td>0.2 MB</td>
<td>12 MOps</td>
<td>156</td>
</tr>
</tbody>
</table>
Platforms Tested

High Performance

- FPGA
  - Altera Arria 10
  - Altera Stratix V

- CPU
  - Intel Xeon E3
  - ARM Cortex 15

- GPU
  - Tesla K40
  - GTX 650 Ti
  - Tegra K1

Low Power

- FPGA
  - Xilinx ZC702

- CPU
  - Altera Stratix V

- GPU
  - Altera Arria 10

Low Power

- CPU
  - ARM Cortex 15

- GPU
  - Tegra K1
Performance vs CPUs

Compared to Xeon, Arria10 is **5.9x** faster, and Zynq is **0.6x** faster.
Compared to TeslaK40, Zynq is **1.6x**, and Arria10 is **1.3x** more power efficient.
Conclusion

FPGAs are a promising option for low-mid power range

However, there is a semantic gap between the high-level DNN models and FPGA acceleration

**DNNWEAVER** is an initial step in making FPGAs more accessible to DNN programmers
Questions?
Backup Slides
Per Layer Speedup CPU

<table>
<thead>
<tr>
<th>Layer Type</th>
<th>Cifar10 Full</th>
<th>LeNet</th>
<th>NiN</th>
<th>Djinn ASR</th>
<th>AlexNet</th>
<th>VGG-CNN-S</th>
<th>Overfeat</th>
<th>VGG-16</th>
<th>Geomean</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conv+Pool</td>
<td>1.6</td>
<td>1.4</td>
<td>0.3</td>
<td>1.6</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.5</td>
<td>0.5</td>
</tr>
<tr>
<td>IP+Act</td>
<td>17.4</td>
<td>7.7</td>
<td>6.7</td>
<td>7.6</td>
<td>12.6</td>
<td>12.1</td>
<td>12.1</td>
<td>13.4</td>
<td>13.4</td>
</tr>
<tr>
<td>Norm</td>
<td>15.8</td>
<td>6.7</td>
<td>7.7</td>
<td>6.7</td>
<td>12.6</td>
<td>12.1</td>
<td>12.1</td>
<td>13.4</td>
<td>13.4</td>
</tr>
<tr>
<td>Conv+Pool</td>
<td>168.8</td>
<td>168.8</td>
<td>168.8</td>
<td>168.8</td>
<td>168.8</td>
<td>168.8</td>
<td>168.8</td>
<td>168.8</td>
<td>168.8</td>
</tr>
<tr>
<td>IP+Act</td>
<td>1499.3</td>
<td>1499.3</td>
<td>1499.3</td>
<td>1499.3</td>
<td>1499.3</td>
<td>1499.3</td>
<td>1499.3</td>
<td>1499.3</td>
<td>1499.3</td>
</tr>
<tr>
<td>Norm</td>
<td>1499.3</td>
<td>1499.3</td>
<td>1499.3</td>
<td>1499.3</td>
<td>1499.3</td>
<td>1499.3</td>
<td>1499.3</td>
<td>1499.3</td>
<td>1499.3</td>
</tr>
<tr>
<td>Conv+Pool</td>
<td>1765.5</td>
<td>1765.5</td>
<td>1765.5</td>
<td>1765.5</td>
<td>1765.5</td>
<td>1765.5</td>
<td>1765.5</td>
<td>1765.5</td>
<td>1765.5</td>
</tr>
<tr>
<td>IP+Act</td>
<td>1680.3</td>
<td>1680.3</td>
<td>1680.3</td>
<td>1680.3</td>
<td>1680.3</td>
<td>1680.3</td>
<td>1680.3</td>
<td>1680.3</td>
<td>1680.3</td>
</tr>
<tr>
<td>Norm</td>
<td>1680.3</td>
<td>1680.3</td>
<td>1680.3</td>
<td>1680.3</td>
<td>1680.3</td>
<td>1680.3</td>
<td>1680.3</td>
<td>1680.3</td>
<td>1680.3</td>
</tr>
</tbody>
</table>

Speedup / Xeon E3 (log)
Performance vs GPU

![Graph showing performance comparison between different hardware and software configurations.](image-url)
Performance-per-Watt vs CPU

- ARM A15
- DW-Zynq
- DW-Stratix
- DW-Arria

<table>
<thead>
<tr>
<th>Model</th>
<th>Cifar10 full</th>
<th>LeNet</th>
<th>NiN</th>
<th>Djinn</th>
<th>ASR</th>
<th>AlexNet</th>
<th>VGG-CNN-S</th>
<th>Overfeat</th>
<th>VGG-16</th>
<th>Gmean</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cifar10</td>
<td>3.3</td>
<td>2.3</td>
<td>1.4</td>
<td>1.7</td>
<td>1.1</td>
<td>5.0</td>
<td>0.9</td>
<td>1.0</td>
<td>1.0</td>
<td>1.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>6.7</td>
<td>7.8</td>
<td>7.5</td>
<td>8.0</td>
<td>16.0</td>
</tr>
<tr>
<td></td>
<td>34</td>
<td>38</td>
<td>17</td>
<td>33</td>
<td>11</td>
<td>10</td>
<td>13</td>
<td>10</td>
<td>8.3</td>
<td>16.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>6.7</td>
<td>7.8</td>
<td>7.5</td>
<td>8.0</td>
<td>13.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3.6</td>
<td>6.3</td>
<td>6.3</td>
<td>8.0</td>
<td>7.9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3.6</td>
<td>6.3</td>
<td>6.3</td>
<td>8.0</td>
<td>7.9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1.6</td>
<td>7.9</td>
<td>8.3</td>
<td>8.0</td>
<td>13.0</td>
</tr>
</tbody>
</table>
## Platforms Tested

<table>
<thead>
<tr>
<th>Platform</th>
<th>CPU/GPU</th>
<th>TDP</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>Altera Arria 10</td>
<td></td>
<td>35W</td>
<td>$4495</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Altera Stratix V</td>
<td></td>
<td>25W</td>
<td>$6999</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Xilinx Zynq ZC702</td>
<td></td>
<td>2W</td>
<td>$129</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Intel Xeon E3-1276 V3</td>
<td></td>
<td>84W</td>
<td>$339</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ARM Cortex 15</td>
<td></td>
<td>5W</td>
<td>$191</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tegra K1 GPU</td>
<td></td>
<td>10W</td>
<td>$191</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GTX 650 Ti</td>
<td></td>
<td>110</td>
<td>$150</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tesla K40</td>
<td></td>
<td>235</td>
<td>$5499</td>
</tr>
</tbody>
</table>
Deep Neural Networks (DNNs)

Convolution and Inner-Product are the Learnable Layers
Deep Neural Networks (DNNs)

DNNWeaver supports all these five layers.
Scheduling Operations on Hardware

Convolution

Inner-product

Batch 0
Batch 1
Batch 2
Batch 3

PU0
PU1
PU2
PU3
Performance vs GPU
Performance-per-Watt vs CPU
Executing the Slice

Mult \rightarrow \text{Add} \rightarrow \text{Mult}

PU_0:
- Normalization
- Pooling
- Activation
- Lookup $f(x)$
Compared to Xeon, Arria10 is 5.9x faster, Stratix V is 2.8x faster, and Zynq is 0.6x faster.
## Performance-per-Watt vs GPUs

<table>
<thead>
<tr>
<th></th>
<th>Tegra K1</th>
<th>Tesla K40</th>
<th>DW-Zynq</th>
<th>DW-Stratix</th>
<th>DW-Arria</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Cifar10</strong></td>
<td>6</td>
<td>5</td>
<td>3.1</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>full LeNet</td>
<td>2.5</td>
<td>2.7</td>
<td>3.2</td>
<td>2.8</td>
<td>2.7</td>
</tr>
<tr>
<td>NiN</td>
<td>1.8</td>
<td>1.1</td>
<td>1.5</td>
<td>1.6</td>
<td>1.2</td>
</tr>
<tr>
<td>Djinn ASR</td>
<td>1.3</td>
<td>1.3</td>
<td>1.2</td>
<td>1.1</td>
<td>1.4</td>
</tr>
<tr>
<td>AlexNet</td>
<td>2.0</td>
<td>2.1</td>
<td>2.0</td>
<td>1.4</td>
<td>1.9</td>
</tr>
<tr>
<td>VGG-CNN-S</td>
<td>2.3</td>
<td>3.8</td>
<td>9.5</td>
<td>4.0</td>
<td>2.0</td>
</tr>
<tr>
<td>Overfeat</td>
<td>4.6</td>
<td>4.5</td>
<td>3.4</td>
<td>3.0</td>
<td>5.3</td>
</tr>
<tr>
<td>VGG-16</td>
<td>4.5</td>
<td>4.7</td>
<td>3.8</td>
<td>3.4</td>
<td>5.3</td>
</tr>
<tr>
<td><strong>Gmean</strong></td>
<td>3.2</td>
<td>3.2</td>
<td>2.7</td>
<td>2.7</td>
<td>2.7</td>
</tr>
</tbody>
</table>

Compared to GTX650, Zynq is **3.2x**, Stratix V is **1.5x**, and Arria10 is **2.7x** more power efficient.
Co-optimize Hardware and Execution Schedule

![Diagram showing hardware and schedule optimization]

- Hardware
  - PE<sub>0</sub>
  - PE<sub>1</sub>
  - ... PE<sub>n-1</sub>
- Schedule
  - PU<sub>0</sub>
    - Normalization
    - Pooling
    - Activation: Lookup \( f(x) \)
Processing Engine

Buffer

ALU

Accumulate

FIFO