用賽靈思 FPGA 加速機器學習推斷

張 帆
資深全球AI方案技術專家
2019年4月25日
AI在edge端應用中落地面臨的挑戰

1080p Object Detection (SSD) @ 30 FPS

= 43 TOPS

< 10W, < 50 ms latency, < $50
Challenges Are Opportunities

ML expertise with FPGA

<5%

ML expertise with GPU/CPU/DSP

25%

No expertise
Looking for near “turn-key” solutions

>70%

Number of Opportunities
Xilinx AI Solution

- Algorithm Model Zoo
- DNNDK
- Xilinx AI SDK
- DPU and Accelerating IPs

- Surveillance
- Data center
- ADAS/AD

- Reference solutions
- Deep learning libraries
- Cross-compiling tools
- BSP for boards

- Compilation
  - Compiler
  - Assembler

- Runtime
  - Core API
  - Loader
  - Driver
  - Tracer

- Zynq-7020
- Zynq-7020
- ZU2
- ZU2
- ZU9
## Algorithm Model Zoo

<table>
<thead>
<tr>
<th>Network</th>
<th>Backbone</th>
<th>DPU Deployment</th>
<th>Input Size</th>
<th>OPs</th>
<th>Paras</th>
<th>Training Set</th>
<th>Val Set</th>
<th>Eva Norm Float</th>
<th>Eva Norm Fixed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resnet50_v1</td>
<td>Resnet</td>
<td>Yes</td>
<td>224*224</td>
<td>7.7G</td>
<td>25.6M</td>
<td>ImageNet</td>
<td>ImageNet</td>
<td>0.7483</td>
<td>0.7338</td>
</tr>
<tr>
<td>Inception_v3</td>
<td>Inception</td>
<td>No</td>
<td>299*299</td>
<td>11.43G</td>
<td>23.8M</td>
<td>ImageNet</td>
<td>ImageNet</td>
<td>0.7401</td>
<td>0.7347</td>
</tr>
<tr>
<td>SSD</td>
<td>VGG16</td>
<td>No</td>
<td>300*300</td>
<td>62.77G</td>
<td>26.3M</td>
<td>Voc07+12</td>
<td>Voc07</td>
<td>77.19%</td>
<td></td>
</tr>
<tr>
<td>RefineDet</td>
<td>VGG-16</td>
<td>Support</td>
<td>480*360</td>
<td>123.9G</td>
<td>29.6M</td>
<td>Coco2014</td>
<td>Coco2014</td>
<td>70.14%</td>
<td></td>
</tr>
<tr>
<td>Densebox</td>
<td></td>
<td>Yes</td>
<td>320*320</td>
<td>492M</td>
<td></td>
<td>Private</td>
<td>Private</td>
<td>97.92%</td>
<td>97.50%</td>
</tr>
<tr>
<td>Yolo_v3</td>
<td>Yolo</td>
<td>Yes</td>
<td>512*288</td>
<td>53.7G</td>
<td>61.8M</td>
<td>Cityscape</td>
<td>Cityscape</td>
<td>53.7%</td>
<td>53.1%</td>
</tr>
</tbody>
</table>

- **DPU deployment**
  - **Yes**: the model is successfully deployed on DPU.
  - **Support**: the model is supported but not deployed. Similar model structure is deployed and test successfully.
  - **No**: the model is not supported by DPU right now mainly due to some special operations or layers.
DNNDK – Deep Neural Network Development Kit

> DECENT
  >> Flt32 to Int8 quantization with one line command

> DNNC
  >> Automatic layer fusion to avoid frequently data read and write

> Runtime N²Cube
  >> Various APIs to facilitate specific application

> Profiler Dsight
  >> Powerful tool as failure analysis and optimization
DNNDK Dev Flow

01 Model Compression
02 Model Compilation
03 Programming
04 Hybrid Compilation
05 Execution

Five Steps with DNNDK
Step 1: Model Compression With DECENT

- Consists of two separate tools
  - Quantization Tool
  - Pruning Tool

- Effects
  - Compress model size
    - 5x – 100x
  - Compress running time
    - 1.5x – 10x

- Platform
  - Caffe, Darknet
  - TensorFlow
    - Quantization Tool Beta version
    - Pruning Tool Internal version

---

decent – Deep compression Tool
decent_q – Quantization Tool
decent_p – Pruning Tool
Step 2: Model Compilation With DNNC

Programmable tensor-level DPU instruction set
- Compatible for Caffe/TensorFlow frameworks
- Flexible & scalable for various CNN layers
Step 3: Programming with DNNDK API

```c
int main(int argc, char *argv[])
{
    DPUKernel *kernel_conv;
    DPUKernel *kernel_fc;
    DPUTask *task_conv;
    DPUTask *task_fc;
    char *input_addr;
    char *output_addr;

    /* DNNDK API to attach to DPU driver */
    dpuInit();

    /* DNNDK API to create DPU kernels for CONV & FC networks */
    kernel_conv = dpuLoadKernel("resnet50_conv", 224, 224);
    kernel_fc = dpuLoadKernel("resnet50_fc", 1, 1);

    /* Create tasks from CONV & FC kernels */
    task_conv = dpuCreateTask(kernel_conv);
    task_fc = dpuCreateTask(kernel_fc);

    /* Set input tensor for CONV task and run */
    input_addr = dpuGetTensorAddress(dpuGetTaskInputTensor(task_conv));
    setInputImage(Mat &image, input_addr);
    dpuRunTask(task_conv);
    output_addr = dpuGetTensorAddress(dpuGetTaskOutputTensor(task_conv));

    /* Run average pooling layer on CPU */
    run_average_pooling(output_addr);

    /* Set input tensor for FC task and run */
    input_addr = dpuGetTensorAddress(dpuGetTaskInputTensor(task_fc));
    setFCInputData(task_fc, input_addr);
    dpuRunTask(task_fc);
    output_addr = dpuGetTensorAddress(dpuGetTaskOutputTensor(task_fc));

    /* Display the Classification result from FC task */
    displayClassificationResult(output_addr);

    /* DNNDK API to destroy DPU tasks/kernels */
    dpuDestroyTask(task_conv);
    dpuDestroyTask(task_fc);
    dpuDestroyKernel(kernel_conv);
    dpuDestroyKernel(kernel_fc);

    /* DNNDK API to detach from DPU driver and free DPU resources */
    dpuFini();
    return 0;
}
```
Step 4: DNNDK Hybrid Compilation Model
Acceleration on Hardware - SSD

Pruning Speedup on Hardware (2xDPU-4096@ZU9)
VGG_SSD 4 classes detection @Deephi surveillance data

<table>
<thead>
<tr>
<th>Pruning Iterations</th>
<th>Operations(G)</th>
<th>mAP(%)</th>
<th>fps</th>
</tr>
</thead>
<tbody>
<tr>
<td>baseline</td>
<td>117</td>
<td>61.5</td>
<td>18</td>
</tr>
<tr>
<td>1</td>
<td>63.4</td>
<td>57</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>63.5</td>
<td>37</td>
<td>20</td>
</tr>
<tr>
<td>3</td>
<td>63.4</td>
<td>27</td>
<td>40</td>
</tr>
<tr>
<td>4</td>
<td>62.4</td>
<td>23</td>
<td>60</td>
</tr>
<tr>
<td>5</td>
<td>62</td>
<td>19</td>
<td>80</td>
</tr>
<tr>
<td>6</td>
<td>61.5</td>
<td>17</td>
<td>100</td>
</tr>
<tr>
<td>7</td>
<td>61.1</td>
<td>15.6</td>
<td>120</td>
</tr>
<tr>
<td>8</td>
<td>61</td>
<td>14.6</td>
<td>140</td>
</tr>
<tr>
<td>9</td>
<td>60.8</td>
<td>13.6</td>
<td>160</td>
</tr>
<tr>
<td>10</td>
<td>59.2</td>
<td>12.2</td>
<td>180</td>
</tr>
<tr>
<td>11</td>
<td>60.4</td>
<td>11.6</td>
<td>200</td>
</tr>
</tbody>
</table>

Operations(G) / mAP / fps
Xilinx AI SDK

- Xilinx AI SDK to enable low touch engagements for customers
  - Encourage top customers to use SDK to build applications and solutions
  - Only use generic SDK release to support low priority customers

### Xilinx AI SDK
- Libraries and reference solutions
- Cross-compiling tools
- BSP for boards

### Xilinx AI Suite
- Algorithm Model Zoo
- DNNDK
- DPU and accelerating IPs
DPU Scalability

Peak Perf
INT8 (OPS)

6.8T  
5.5T  
4.1T  
3.5T  
2.9T  
2.8T  
2.4T  
1.7T  
1.6T  
1.2T  
700G  
576G  
230G  
115G  
102G  
56G  

ZU15  
ZU11  
ZU9  
ZU7  
ZU6  
ZU5  
ZU4  
ZU3  
ZU2  
ZU14S/ZU15  
ZU12S  
ZU10  

DPU Configuration

<table>
<thead>
<tr>
<th></th>
<th>LUTs</th>
<th>Registers</th>
<th>BRAM</th>
<th>DSP</th>
</tr>
</thead>
<tbody>
<tr>
<td>B256 (8x4x4)</td>
<td>16132</td>
<td>25064</td>
<td>43</td>
<td>66</td>
</tr>
<tr>
<td>B256 (2x8x8)</td>
<td>15286</td>
<td>22624</td>
<td>53.5</td>
<td>50</td>
</tr>
<tr>
<td>B288 (4x6x6)</td>
<td>15812</td>
<td>23689</td>
<td>46</td>
<td>62</td>
</tr>
<tr>
<td>B512 (4x8x8)</td>
<td>20177</td>
<td>31782</td>
<td>69.5</td>
<td>98</td>
</tr>
<tr>
<td>B1024 (8x8x8)</td>
<td>27377</td>
<td>46241</td>
<td>101.5</td>
<td>194</td>
</tr>
<tr>
<td>B1152 (4x12x12)</td>
<td>28698</td>
<td>46906</td>
<td>117.5</td>
<td>194</td>
</tr>
<tr>
<td>B1600 (8x10x10)</td>
<td>30877</td>
<td>56267</td>
<td>123</td>
<td>282</td>
</tr>
<tr>
<td>B2304 (8x12x12)</td>
<td>34379</td>
<td>67481</td>
<td>161.5</td>
<td>386</td>
</tr>
<tr>
<td>B3136 (8x14x14)</td>
<td>38555</td>
<td>79867</td>
<td>203.5</td>
<td>506</td>
</tr>
<tr>
<td>B4096 (8x16x16)</td>
<td>40865</td>
<td>92630</td>
<td>249.5</td>
<td>642</td>
</tr>
</tbody>
</table>

* B256/288/512/3136 work in progress
DPU IP Integration

The DPU TRD (Targeted Reference Design) has been published in Xilinx.com.
Key Takeaway

1. Xilinx offers advanced Edge ML solution
2. DNNDK makes neural network deployment easily on FPGA
3. DPU IP is ready to download from Xilinx website for free
Adaptable.
Intelligent.