A Lightweight YOLOv2: A Binarized CNN with a Parallel Support Vector Regression for an FPGA

Hiroki Nakahara, Haruyoshi Yonekawa, Tomoya Fujii, Shimpei Sato
Tokyo Institute of Technology, Japan
Outline

• Background
  • Convolutional Neural Network (CNN)

• Mixed-precision CNN for a Lightweight YOLOv2
  • Binary precision CNN
  • Half precision support vector regression (SVR)

• FPGA Implementation

• Experimental Results

• Conclusion
Deep Learning is Everywhere
Convolutional Neural Network (CNN)

- Convolutional + Fully connected + Pooling
- State-of-the-art performance in an image recognition task
- Widely applicable

Image Recognition Tasks

Easy

• Classification
  • Answer “category” of the object in an image

Hard

• Object Detection
  • Classification + localization

Baby (44%)
Son (23%)
Daughter (33%)
Applications

• Robotics, autonomous driving, security, drones...
Demo

Available at https://www.youtube.com/watch?v=_iMboyu8iWc
## Requirements in Embedded System

<table>
<thead>
<tr>
<th>Cloud</th>
<th>Embedded</th>
</tr>
</thead>
<tbody>
<tr>
<td>Many classes (1000s)</td>
<td>Few classes (&lt;10)</td>
</tr>
<tr>
<td>Large workloads</td>
<td>Frame rates (15-30 FPS)</td>
</tr>
<tr>
<td>High efficiency (Performance/W)</td>
<td>Low cost &amp; low power (1W-5W)</td>
</tr>
<tr>
<td>Server form factor</td>
<td>Custom form factor</td>
</tr>
</tbody>
</table>

J. Freeman (Intel), “FPGA Acceleration in the era of high level design”, HEART2017
Deep Learning Inference Device

• Flexibility: R&D costs for keeping on evolving algorithms
• Power performance efficiency
• FPGA has flexibility & better performance
Outline

• Background
  • Convolutional Neural Network (CNN)

• Mixed-precision CNN for a Lightweight YOLOv2
  • Binary precision CNN
  • Half precision support vector regression (SVR)

• FPGA Implementation

• Experimental Results

• Conclusion
Object Detection Problem

• Detecting and classifying multiple objects at the same time
• Evaluation criteria (from Pascal VOC):

Ground truth annotation

Detection results:
>50% overlap of bounding box (BBox) with ground truth
One BBox for each object
Confidence value for each object

Ground truth annotation

Person (50%)

Detection results:
>50% overlap of bounding box (BBox) with ground truth
One BBox for each object
Confidence value for each object

\[
\text{precision} = \frac{\# \text{ Correct detect.}}{\# \text{all detect.}}
\]

\[
\text{recall} = \frac{\# \text{ Correct detect.}}{\# \text{all objects}}
\]

Average Precision (AP):

\[
AP = \frac{1}{11} \sum_{r \in \{0, 1, \ldots, 11\}} P_{\text{interp}(r)}
\]
YOLOv2 (You Only Look Once version 2)

- Single CNN (One-shot) object detector
  - Both a classification and a BBox estimation for each grid

2D Convolutional Operation

- Computational intensive part of the YOLOv2
Binarized CNN

\[ w_0 (\text{Bias}) \]

\[ w_1 \]
\[ x_1 \]
\[ w_2 \]
\[ x_2 \]
\[ \vdots \]
\[ w_n \]
\[ x_n \]

\[ Y = x_1 x_2 Y \]
\[ z = f_{\text{sgn}}(Y) \]

<table>
<thead>
<tr>
<th>x1</th>
<th>x2</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>-1</td>
<td>1</td>
</tr>
<tr>
<td>-1</td>
<td>+1</td>
<td>-1</td>
</tr>
<tr>
<td>+1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>+1</td>
<td>+1</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>x1</th>
<th>x2</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Improvements by Binarization

\[ w_0 (\text{Bias}) \]

\[ w_1 \]

\[ x_1 \]

\[ w_2 \]

\[ x_2 \]

\[ \vdots \]

\[ w_n \]

\[ x_n \]

\[ \sum \]

\[ Y \]

\[ f_{\text{sgn}}(Y) \]

\[ Z \]

EXNORs → Many MACs

Binary Precision → On-chip Memory

<table>
<thead>
<tr>
<th>x1</th>
<th>x2</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>-1</td>
<td>1</td>
</tr>
<tr>
<td>-1</td>
<td>+1</td>
<td>-1</td>
</tr>
<tr>
<td>+1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>+1</td>
<td>+1</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>x1</th>
<th>x2</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

EXNORs → Many MACs

Binary Precision → On-chip Memory
Near Memory Realization by Binarization

- High bandwidth (Left)
- Less power consumption (Right)

J. Dean, “Numbers everyone should know”
Source: https://gist.github.com/2841832

**Typical CNN for Classification**

Input image → Feature maps

- **CONV+Pooling**
- **CONV+Pooling**

Feature extraction layers → Classification layers

- Feature maps with varying dimensions:
  - Input image: 5
  - Feature extraction layers:...
  - Classification layers: 2, 0, 4, 6, 7, 1, 5, 3, 8, 9
Hypothesis

• Does binarized feature map has a location? → Yes
Problem

- Low precision NN is hard to regress a function
- Example: sin(x) regression using a NN (3-layers)

(a) Float 32 bit for activation and weight
(b) Float 32 for activation and binary weight
(c) All binarized

Miss localization
Proposed YOLOv2

- Feature extraction layer: Binary precision
- Localization and classification layer: Half precision
Support Vector Regression (SVR)

- Regression version of the Support Vector Machine (SVM)*1
- Passive aggressive (On-line) training is supported*2
- Model decompression (sparse like) can be applied*3

\[ y = \sum_{i=1}^{n} \langle w, x_i \rangle + b \]

\( w \): weight, \( x_i \): i-th input, and \( b \): bias.

Outline

• Background
  • Convolutional Neural Network (CNN)

• Mixed-precision CNN for a Lightweight YOLOv2
  • Binary precision CNN
  • Half precision support vector regression (SVR)

• FPGA Implementation

• Experimental Results

• Conclusion
Parallel Binarized CNN Circuit

Update $N_i+1$ bits  
Read $N_i$ bits

$2L+K$ bits Shift Register

Binarized MACs

Sign

Write Ctrl. Logic

Counter

On Chip Memory
Parallel SVR Circuit

Feature Extracted CNN

- Binarized Feature map memory
- SVR for class_20
- SVR for class_1
- SVR for conf
- SVR for w
- SVR for h
- SVR for y
- SVR for x

Parallel SVR

Circuit for a SVR

Localization and Classification

Weight cache

Index

From Binarized F. Map memory

-1×w

w

0

1

Reg

Clear

bias

Circuit for SVR

25
Overall Architecture

Host ARM Processor

AXI4-BUS

F. map memory

Streaming Binarized 2D Conv. Circuit

Binarized Weight Cache

Shift Register (2L+K bits)

Binarized Weight Cache

Weight Cache (W.C.)

Index

SVR for class_20

SVR for conf.

SVR for x

W.C.

W.C.

W.C.

Reg

Clear

-1 \times w

w

0

1

b
Outline

• Background
  • Convolutional Neural Network (CNN)

• Mixed-precision CNN for a Lightweight YOLOv2
  • Binary precision CNN
  • Half precision support vector regression (SVR)

• FPGA Implementation

• Experimental Results

• Conclusion
Training Result

• Environment
  • CNN: Proposed YOLOv2
    • Binarized Darknet19+SVR
  • Dataset: Pascal VOC2007
    • 21 class, 224x224 image size

• Framework
  • Binary precision CNN: GUINNES*1
  • Half precision SVR: Pegasos*2

• Accuracy (mAP)
  • 67.6%

Implementation Setup

- Board: Xilinx Inc. Zynq UltraScale+ MPSoC zcu102 evaluation board
  - Zynq UltraScale+ MPSoC FPGA
- Design tool: SDSoc 2017.4
  - Timing constraint: 299.97MHz

<table>
<thead>
<tr>
<th>Module</th>
<th>#LUTs</th>
<th>#FFs</th>
<th>#18Kb BRAMs</th>
<th>#DSP48Es</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary CNN (2D bin. Conv.)</td>
<td>108,138</td>
<td>358,868</td>
<td>1680</td>
<td>135</td>
</tr>
<tr>
<td></td>
<td>(103,924)</td>
<td>(313,839)</td>
<td>(0)</td>
<td>(0)</td>
</tr>
<tr>
<td>Parallel SVR</td>
<td>27,243</td>
<td>11,431</td>
<td>26</td>
<td>242</td>
</tr>
<tr>
<td>Total (%)</td>
<td>135,381</td>
<td>370,299</td>
<td>1,706</td>
<td>377</td>
</tr>
<tr>
<td></td>
<td>(49.3)</td>
<td>(67.5)</td>
<td>(93.5)</td>
<td>(14.9)</td>
</tr>
</tbody>
</table>
## Comparison

<table>
<thead>
<tr>
<th>Platform</th>
<th>Embedded CPU</th>
<th>Embedded GPU</th>
<th>FPGA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device</td>
<td>Quad-core ARM Cortex-A57</td>
<td>256-core Pascal GPU</td>
<td>Zynq UltraScale+ MPSoC</td>
</tr>
<tr>
<td>Clock Freq. [GHz]</td>
<td>1.9</td>
<td>1.3</td>
<td>0.3</td>
</tr>
<tr>
<td>Memory</td>
<td>32 GB eMMC Flash</td>
<td>8GB LPDDR4</td>
<td>32.1 Mb BRAM</td>
</tr>
<tr>
<td>Time [msec] (FPS) [sec⁻¹]</td>
<td>4210.0 (0.23)</td>
<td>715.9 (1.48*)</td>
<td>24.5 (40.81)</td>
</tr>
<tr>
<td>Dynamic Power [W]</td>
<td>4.0</td>
<td>7.0</td>
<td>4.5</td>
</tr>
<tr>
<td>Efficiency [FPS/W]</td>
<td>0.057</td>
<td>0.211</td>
<td>9.060</td>
</tr>
</tbody>
</table>

* Chainer (version 1.24.0), source code: https://github.com/leetenki/YOLOv2
Conclusion

• Lightweight YOLOv2 for a real-time object detector
  • Mixed-precision CNN
    • Binary precision CNN: Feature extraction
    • Half precision SVR: Classification and localization

• FPGA Implementation
  • Outperforms an embedded GPU and a CPU

• Future Work: Applied to CNN-based applications
  • Single-shot object detector (SSD, PVANet)
  • Semantic segmentation (FCN, U-Net)
  • Pose estimation (OpenPose)
  • CNN SLAM
Thank you!

Contact:
Hiroki Nakahara, Tokyo Institute of Technology
nakahara@ict.e.titech.ac.jp