PoET-BiN: Power Efficient Tiny Binary Neurons

Sivakumar Chidambaram\textsuperscript{1}, J.M. Pierre Langlois\textsuperscript{2}, Jean Pierre David\textsuperscript{1}

Department of Electrical Engineering \textsuperscript{1}
Department of Computer and Software Engineering \textsuperscript{2}
Polytechnique Montréal
Montréal, Canada

03 March 2020
Contents

1. Introduction
2. Background
3. PoET-BiN
4. Experimental setup and results
5. Conclusion
Real-time deep learning use cases

Autonomous Driving

CCTV Monitoring

Required Attributes:
- Accuracy
- Latency and Throughput
- Power and Energy constraints
- Memory and Hardware costs

Translation
Source: www.firebase.google.com/docs/ml-kit/translation
Computation needs

**Exponential Growth in the Training of Artificial Intelligence Programs**

Source: https://openai.com/blog/ai-and-compute/
Note: A petaFLOPS is a unit of computing speed equal to one quadrillion FLOPS, floating operations per second, a measure of computer performance.
Current Deep Learning Software Acceleration Techniques

Quantized Neural Networks
- Quantization of weights and activations
- Binarizing, Ternarization, Multi-bit quantization
- Helps in generalization on the unseen data

Pruning - Remove certain neurons from the vanilla neural network
- A bagging technique that averages various randomly pruned networks
- Introduces noise in the system that helps perform better on unseen data

Sparsification - Sparse matrix multiplication
- Removing connections between neurons
- Reduces the number of multiplication and additions
- Reduces number of memory reads

- Implemented on hardware devices such as FPGA, microprocessors, microcontrollers etc.
Hardware: FPGAs

- Goto device for rapid prototyping of accelerators
- FPGAs consist of Arithmetic Logical Modules (ALMs), programmable interconnects, IOs and BRAMs
- ALMs are the main computational unit
- 100,000s of ALMs in a typical FPGA
- Each ALM has a LookUp Table (LUT) with up to 8 inputs and up to 2 outputs
- Programmed using Hardware Description Languages

Source: https://hackaday.com/2018/03/01/another-introduction-to-fpgas/
Problem Definition and Objectives

Problem Definition:
- Vanilla neural networks are computation, power and area intensive.
- Current acceleration approaches are still computationally intensive.
- Quantized neural networks and pruning are not optimized for FPGAs.

Objectives/Contributions:
- A modified Decision Tree training algorithm to better match LUTs with a fixed number of inputs.
- The Reduced Input Neural Circuit (RINC): A LUT-based architecture founded on modified Decision Trees and the hierarchical version of the well known Adaboost algorithm to efficiently implement a network of binary neurons.
- A sparsely connected output layer for multiclass classification.
- The PoET-BiN architecture consisting of multiple RINC modules and a sparsely connected output layer.
- Automatic VHDL code generation of the PoET-BiN architecture for FPGA implementation.
Binary Decision Trees

**What are Binary DTs:**
- Inputs(X) and Outputs(Y) are binary
- Node wise creation of Decision Tree from root to leaves
- The feature that most reduces the entropy is chosen
- Divides the representation space to classify data

**Challenges:**
- To classify large datasets - need larger Decision Trees
- Results in large implementations on the hardware- complex and high power consumption
- To effectively implement on FPGAs we need small Decision Trees of $\leq 6$ inputs to fit in one LUT
RINC-0 : Modified Decision Tree Algorithm

- Modified DT algorithm - level based entropy reduction rather than node based
- Decision Trees are restricted by the number of inputs (I)
- A node-wise off-the-shelf 6-input Decision Tree would have only 7 leaf nodes
- Level-wise Decision Tree will have $2^6 = 64$ leaf nodes
- More granularity
RINC-1 : Incorporating Adaboost

- A single Decision Tree is a weak classifier
- Ensemble methods such as Boosting and Bagging are used to create strong classifiers from weak classifiers
- We use the well-know Adaboost algorithm

Adaboost Classifier Working Principle with Decision Stump as a Base Classifier

- The weak classifiers are created serially
- The samples are given equal weight initially
- The first weak classifier is trained on the data
- The mis-classified sample’s weights are increased
- Subsequent classifier focuses on the incorrect samples
- Each classifier is assigned a weight based on the number of correctly classified samples
- A weighted sum of all the weak classifier outputs forms the strong classifier

Adaboost Algorithm

\[ H = \text{sign} \left( 0.38 \times + \right) + 0.58 \times - + 0.87 \times - \]

RINC-1 Module

- The MAC and threshold operations can be implemented in a LUT
- Can group up to a maximum of $P$ Decision Trees
- However, $P$ Decision Trees with $P^2$ inputs are not enough compared to a MAC operation in a neural network
- Neuron in a neural network can have up to 4096 inputs as compared 36 (when $P=6$) in RINC-1 modules
- Hence, we introduce the hierarchical Adaboost algorithm
The RINC-2 modules have adequate capacity to represent MAC operations.

- Can only be used for binary classifications.
Binary to Multiclass Classification

Current Methods:
- Multiclass DTs: costly to implement
- One-vs-All classification: leads to reduction in accuracy

Our Approach:
- A sparsely connected intermediate layer before the final output layer for multiclass classification
- Only P neurons of the intermediate layer connected to each neuron in the output layer
- The neurons in the output layer need to have multiple bits to represent the probabilities and cannot be binary values
- Implemented as LUTs
### Experimental Setup

**Vanilla Network (A₁)**
- **FE** → **Hidden Layers of Classifier** → **Output layer**

**Binary Feature Representation Network (A₂)**
- **FE** → **BIN Act.** → **Hidden Layers of Classifier** → **Output layer**

**Teacher Network (A₃)**
- **FE** → **BIN Act.** → **Hidden Layers of Classifier** → **Inter. Layer** → **BIN Act.** → **Output layer**

**Final Architecture (A₄)**
- **FE** → **BIN Act.** → **RINC Classifiers** → **Sparsely Connected Output layer**

---

**TABLE – Network architecture**

<table>
<thead>
<tr>
<th>ARCHITECTURE (ARCH.)</th>
<th>SYMBOL</th>
<th>DATASET</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>LeNET</em>&lt;sub&gt;FE&lt;/sub&gt; – (512FC) – (10FC)</td>
<td>M1</td>
<td>MNIST</td>
</tr>
<tr>
<td><em>VGG11</em>&lt;sub&gt;FE&lt;/sub&gt; – (4096FC) – (4096FC) – (10FC)</td>
<td>C1</td>
<td>CIFAR-10</td>
</tr>
<tr>
<td><em>VGG11</em>&lt;sub&gt;FE&lt;/sub&gt; – (2048FC) – (2048FC) – (10FC)</td>
<td>S1</td>
<td>SVHN</td>
</tr>
</tbody>
</table>
Results: Accuracy

- $A_1$: Vanilla network, $A_2$: Network with binary features, $A_3$: Teacher network with intermediate layer, $A_4$: PoET-BiN

<table>
<thead>
<tr>
<th>ARCH.</th>
<th>DATASET</th>
<th>$A_1$ (%)</th>
<th>$A_2$ (%)</th>
<th>$A_3$ (%)</th>
<th>$A_4$ (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>M1</td>
<td>MNIST</td>
<td>99.20</td>
<td>99.06</td>
<td>98.93</td>
<td>98.15</td>
</tr>
<tr>
<td>C1</td>
<td>CIFAR-10</td>
<td>91.02</td>
<td>89.88</td>
<td>89.10</td>
<td>92.64</td>
</tr>
<tr>
<td>S1</td>
<td>SVHN</td>
<td>97.36</td>
<td>96.98</td>
<td>96.22</td>
<td>95.13</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IMPLEMENTATION</th>
<th>ACCURACY (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNIST</td>
<td></td>
</tr>
<tr>
<td>CIFAR-10</td>
<td></td>
</tr>
<tr>
<td>SVHN</td>
<td></td>
</tr>
<tr>
<td><strong>BINARYNET(2016)</strong></td>
<td>98.97 89.76 95.06</td>
</tr>
<tr>
<td><strong>POLYBINN(2018)</strong></td>
<td>97.52 91.58 94.97</td>
</tr>
<tr>
<td><strong>NDF(2015)</strong></td>
<td>99.42 90.46 95.20</td>
</tr>
<tr>
<td><strong>OUR WORK</strong></td>
<td>98.15 92.64 95.13</td>
</tr>
</tbody>
</table>

There is a reduction in accuracy for each modification introduced.

Comparable accuracy with other state-of-the-art networks.

Same feature extractor.

BinaryNet - Neural Network approach.

POLYBiNN - Decision Tree approach.

NDF - Hybrid approach.

Same feature extractor for fair comparisons.
Results: Power Consumption

- Measurements from Xilinx Power Analyzer tool
- Power consumption of the classification layers only

<table>
<thead>
<tr>
<th>TABLE – RINC power</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DATA SET</strong></td>
</tr>
<tr>
<td><strong>DYNAMIC(W)</strong></td>
</tr>
<tr>
<td><strong>STATIC(W)</strong></td>
</tr>
<tr>
<td><strong>TOTAL(W)</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TABLE – Number of arithmetic operations</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>OP.</strong></td>
</tr>
<tr>
<td><strong>ADD.</strong></td>
</tr>
<tr>
<td><strong>MULT.</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TABLE – Single arithmetic operation power</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>OPERATION</strong></td>
</tr>
<tr>
<td><strong>MULTIPL. (16 BITS)</strong></td>
</tr>
<tr>
<td><strong>MULTIPL. (32 BITS)</strong></td>
</tr>
<tr>
<td><strong>ADD. (16 BITS)</strong></td>
</tr>
<tr>
<td><strong>ADD. (32 BITS)</strong></td>
</tr>
<tr>
<td><strong>MULTIPL. (FP)</strong></td>
</tr>
<tr>
<td><strong>ADD. (FP)</strong></td>
</tr>
</tbody>
</table>
Results: Energy Consumption, Latency and Hardware Costs

- The networks were implemented on a Spartan-6 Xilinx FPGA
- Energy reduction by up to three orders of magnitude when compared to recent binary quantized neural networks

<table>
<thead>
<tr>
<th>TECHNIQUE</th>
<th>ENERGY (J) MNIST</th>
<th>ENERGY (J) CIFAR-10</th>
<th>ENERGY (J) SVHN</th>
</tr>
</thead>
<tbody>
<tr>
<td>VANILLA</td>
<td>$8.0 \times 10^{-5}$</td>
<td>$5.7 \times 10^{-3}$</td>
<td>$1.6 \times 10^{-3}$</td>
</tr>
<tr>
<td>1-BIT QUANT</td>
<td>$2.1 \times 10^{-7}$</td>
<td>$3.9 \times 10^{-5}$</td>
<td>$9.2 \times 10^{-6}$</td>
</tr>
<tr>
<td>16-BIT QUANT</td>
<td>$8.5 \times 10^{-6}$</td>
<td>$6.0 \times 10^{-4}$</td>
<td>$1.0 \times 10^{-4}$</td>
</tr>
<tr>
<td>32-BIT QUANT</td>
<td>$1.7 \times 10^{-5}$</td>
<td>$1.2 \times 10^{-3}$</td>
<td>$3.6 \times 10^{-4}$</td>
</tr>
<tr>
<td>PoET-BiN</td>
<td>$8.2 \times 10^{-9}$</td>
<td>$5.4 \times 10^{-9}$</td>
<td>$4.1 \times 10^{-9}$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TABLE – Implementation results</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATA SET</td>
</tr>
<tr>
<td>LATENCY (NS)</td>
</tr>
<tr>
<td>NUMBER OF LUTs</td>
</tr>
</tbody>
</table>
Conclusion

- Proposed a Power-efficient Tiny Binary Neuron architecture
- Removed all MAC operations and memory access in classification layers
- Achieved comparable accuracies to other state-of-the-art works

Advantages

- Reduction in energy by up to three orders of magnitude when compared to recent binary quantized neural networks
- Can be implemented in any hardware, not just FPGAs

Further Work

- Implementation for the convolutional layers
- Results for larger datasets
We thank
- Ahmed Abdelsalam for his suggestions and comments throughout the project
- MITACS and ReSMiQ for partially sponsoring the project
Thanks!

Questions???