Invocation-driven Neural Approximate Computing with a Multiclass-Classifier and Multiple Approximators

Haiyue Song, Chengwen Xu, Qiang Xu, Zhuoran Song, Naifeng Jing, Xiaoyao Liang, and Li Jiang
Advanced Computer Architecture Laboratory
Shanghai Jiao Tong University
Approximate Computing

- Many applications are error tolerant
- Neural network (NN) is suitable to approximate a code block/function
  - Amdahl law: performance limited by serial code
  - NN has high parallelism, e.g., FPGA, ASIC, GPU
  - An interesting facts: Neural network can approximate any continuous function
Related works

- Model based quality control for Approximate Computing [ISCA’15, ISLPED’16, DATE’16]
  - Classifier: predict the data is “approximatable” or not
  - Approximator (Accelerator): approximately compute data at fast speed and low power consumption
  - Error: the gap between the output of approximator and that of original program

With quality control architecture

NN-based

Classifier

Input

Error < \( \theta \)

Accelerator

Output

Error > \( \theta \)

Original program
Related works

- Model based quality control for Approximate Computing [ISCA’15, ISLPED’16, DATE’16]
  - Classifier: predict the data is “approximatable” or not
  - Approximator (Accelerator): approximately compute data at fast speed and low power consumption
  - Error: the gap between the output of approximator and that of original program

- Question:
  How to train NN-based classifier and approximator?
Related works

- One-pass training [ISCA’16]
  - Train Approximator and Classifier separately
  - Ignore the correlation between the two NNs
Related works

- **Iterative training [DAC’17]**
  - Train Approximator and Classifier together using iterative training
  - Classifier correlate with Approximator
  - Data with low error is easy to predict
Motivation

- Problems
  - Even iterative training, some data still fail to be approximated (red part in the figure)
  - Single Approximator may overfit one cluster/distribution of input sample

Do we really have to give up those data?
Motivation

- Problems
  - Even iterative training, some data still fail to be approximated (red part in the figure)
  - Single Approximator may overfit one cluster/distribution of input sample

- Motivation
  - Multiple approximators may be complementary, and make invocation higher

Do we really have to give up those data?
Multiple Cascaded Classifiers and Approximators (MCCA)

- **Training Process**
  - The original input samples are used to train classifier C1 and approximator A1.
Multiple Cascaded Classifiers and Approximators (MCCA)

- Training Process
  - The original input samples are used to train classifier C1 and approximator A1.
  - Feed the remaining input samples not yet to be recognized by C1 (Data nC) to classifier C2 and approximator A2.
Multiple Cascaded Classifiers and Approximators (MCCA)

- **Training Process**
  - The original input samples are used to train classifier C1 and approximator A1.
  - Feed the remaining input samples not yet to be recognized by C1 (Data nC) to classifier C2 and approximator A2.
  - Repeat until a specific pair of Cn and An cannot converge.
Multiple Cascaded Classifiers and Approximators (MCCA)

- Inference Process
  - If C1 approves, the input data are sent to A1.
Multiple Cascaded Classifiers and Approximators (MCCA)

- **Inference Process**
  - If C1 approves, the input data are sent to A1.
  - If C1 disapproves, the input data are sent to the next classifier C2.
Inference Process

- If C1 approves, the input data are sent to A1.
- If C1 disapproves, the input data are sent to the next classifier C2.
- Repeat until Cn approves.
Multiple Cascaded Classifiers and Approximators (MCCA)

- Inference Process
  - If C1 approves, the input data are sent to A1.
  - If C1 disapproves, the input data are sent to the next classifier C2.
  - Repeat until Cn approves.

- Demerit
  - The time spending on inference is too long
Multiclass-classifier and Multiple Approximators (MCMA)

- **Inference Process**
  - The multiclass-classifier predicts which approximator can approximate the input data.
Complementary training

- Test A1 with all data, produce the label C1 for any input sample that A1 can safely approximate.
Multiclass-classifier and Multiple Approximators (MCMA)

- Complementary training
  - Test A1 with all data, produce the label C1 for any input sample that A1 can safely approximate
  - Test A2 with the remaining data, produce the label C2 for any input sample that A2 can safely approximate

Training Process:
- Origin Data
- Complementary Allocation
- Select & Train
- Discarded Data for A1
  - Data for A2
  - Data for An
- Multiclass classifier
- Training Process
Multiclass-classifier and Multiple Approximators (MCMA)

**Complementary training**

- Test A1 with all data, produce the label C1 for any input sample that A1 can safely approximate.
- Test A2 with the remaining data, produce the label C2 for any input sample that A2 can safely approximate.
- Repeat until test An, the remaining input samples without any label are labeled as nC.
Multiclass-classifier and Multiple Approximators (MCMA)

- Complementary training
  - Test A1 with all data, produce the label C1 for any input sample that A1 can safely approximate
  - Test A2 with the remaining data, produce the label C2 for any input sample that A2 can safely approximate
  - Repeat until test An, the remaining input samples without any label are labeled as nC.
  - Train the multiclass-classifier and approximators using iterative training.
Multiclass-classifier and Multiple Approximators (MCMA)

- **Competitive training**
  - Test A1 with all data, obtain the approximation error.
  - Test A2 with all data, obtain the approximation error.
  - ...
  - Test An with all data, obtain the approximation error.
Multiclass-classifier and Multiple Approximators (MCMA)

- Competitive training
  - Test A1 with all data, obtain the approximation error.
  - Test A2 with all data, obtain the approximation error.
  - ...
  - Test An with all data, obtain the approximation error.
  - Generate the label for each data according to the lowest approximation error.
- **Competitive training**
  - Test A1 with all data, obtain the approximation error.
  - Test A2 with all data, obtain the approximation error.
  - ...
  - Test An with all data, obtain the approximation error.
  - Generate the label for each data according to the lowest approximation error.
  - Train the multiclass-classifier and approximators using iterative training.
Hardware design

- Add a Controller to control the weight buffer inside the PE.
Hardware design

- Add Controller to control multiple approximators.
- Weight buffer receives the signal from the controller, and then schedule approximators.
Hardware design

- Read data from Input FIFO.

Data flow
Hardware design

- Conduct vector multiplication in PE.

Data flow
Hardware design

- Controller send signal to CPU or Approximator.
If approximator invoked, fetch corresponding approximator’s weight.
Hardware design

- Conduct vector multiplication in PE.

Data flow
Hardware design

- Send back the result from PE to output FIFO.
Hardware design

- Load the weights layer by layer.
### Experimental setup

- Compared with One-pass[ISCA’16] and Iterative training[DAC’17]
- 8 benchmark applications

<table>
<thead>
<tr>
<th></th>
<th>Benchmark</th>
<th>Domain</th>
<th>Train Data</th>
<th>Test Data</th>
<th>Approximator Topology</th>
<th>Classifier Topology</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Black-Scholes</td>
<td>Financial Analysis</td>
<td>70K options</td>
<td>30K options</td>
<td>6-&gt;8-&gt;1</td>
<td>6-&gt;8-&gt;2(4)</td>
</tr>
<tr>
<td>2</td>
<td>FFT</td>
<td>Signal Processing</td>
<td>8K fp numbers</td>
<td>3K fp numbers</td>
<td>1-&gt;2-&gt;2-&gt;2</td>
<td>1-&gt;2-&gt;2(4)</td>
</tr>
<tr>
<td>3</td>
<td>Inversek2j</td>
<td>Robotics</td>
<td>70K (x,y) pairs</td>
<td>30K (x,y) pairs</td>
<td>2-&gt;8-&gt;2</td>
<td>2-&gt;8-&gt;2(4)</td>
</tr>
<tr>
<td>4</td>
<td>Jmeint</td>
<td>3D gaming</td>
<td>70K triangles</td>
<td>30K triangles</td>
<td>18-&gt;32-&gt;16-&gt;2</td>
<td>18-&gt;16-&gt;2(4)</td>
</tr>
<tr>
<td>5</td>
<td>JPEG encoder</td>
<td>Compression</td>
<td>512×512 pixel color image</td>
<td>512×512 pixel color image</td>
<td>64-&gt;16-&gt;64</td>
<td>64-&gt;16-&gt;2(4)</td>
</tr>
<tr>
<td>6</td>
<td>K-means</td>
<td>Machine Learning</td>
<td>100K pairs of (r,g,b) points</td>
<td>50K pairs of (r,g,b) points</td>
<td>6-&gt;8-&gt;4-&gt;1</td>
<td>6-&gt;8-&gt;4-&gt;2(4)</td>
</tr>
<tr>
<td>7</td>
<td>Sobel</td>
<td>Image Processing</td>
<td>512×512 pixel color image</td>
<td>512×512 pixel color image</td>
<td>9-&gt;8-&gt;1</td>
<td>9-&gt;8-&gt;2(4)</td>
</tr>
<tr>
<td>8</td>
<td>Bessel</td>
<td>Scientific Computing</td>
<td>70K fp pairs</td>
<td>30K fp pairs</td>
<td>2-&gt;4-&gt;4-&gt;1</td>
<td>2-&gt;4-&gt;2(4)</td>
</tr>
</tbody>
</table>
Experiment Results

- Invocation increase **20%~30%** on average.
- Invocation increase **40%+** in sobel or kmeans benchmark.
The approximation error is **below** the error bound in **most** benchmarks.
The average speedup is 1.23x compared with one-pass method.
Experiment Results

• The average energy reduction is $1.15x$ compared with one-pass method.
Experiment

- Almost all samples have a corresponding approximator that can approximate it.
Invocation-driven Neural Approximate Computing with a Multiclass-Classifier and Multiple Approximators

Zhuoran Song (宋卓然)
Professor Li Jiang (蒋力)
Advanced Computer Architecture Laboratory
Shanghai Jiao Tong University

Thanks for listening!