Google Neural Network Models for Edge Devices: Analyzing and Mitigating Machine Learning Inference Bottleneck

Amirali Boroumand  Saugata Ghose  Berkin Akin
Ravi Narayanaswami  Geraldo F. Oliveira  Xiaoyu Ma
Eric Shiu  Onur Mutlu

PACT 2021

Seminar in Computer Architecture 2022 Fall
Yeejoo Han
03.11.2022
Executive summary

Background

ML inference of different models are pushed to the edge devices

Problem

The state-of-art Edge TPU has three shortcomings

- It operates significantly below its peak throughput
- It operates significantly below its peak energy efficiency
- It has inefficient memory system

Key Insight

The shortcomings arise from heterogeneity of NN layers and monolithic design of the edge TPU.

Goal

Design an edge accelerator that can fully exploit the variation of layers

Key Mechanism

Mensa: a new framework for designing accelerator

- Mensa consists of a collection of smaller HW accelerators that are specialized to specific subsets of layers

Key Results

G-Mensa was designed and tested on Google edge NN models

- Performance and energy efficiency was improved
- Inference latency was reduced
ML inference computation is pushed directly to edge devices
Machine Learning on Edge Devices

Benefits of edge computation

- Enhanced data security
- Real-time responses
- Reduced operational costs
- No network constraints

Constraints of edge computation

- Limited power budget
- Limited computational resources

Specialized accelerators for edge computation are needed
Edge Neural Network Models

<table>
<thead>
<tr>
<th>Model Type</th>
<th>Examples</th>
<th>Model Type</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNN</td>
<td>Face detection, 15 CNNs</td>
<td>LSTM</td>
<td>Language Translation, 2 LSTMs</td>
</tr>
<tr>
<td>RNN, Transducers</td>
<td>Speech Recognition, 4 Transducers</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RCNN</td>
<td></td>
<td></td>
<td>Image captioning, 3 RCNNs</td>
</tr>
</tbody>
</table>

**Challenge:**
Edge accelerators must inference **efficiently** across a **wide variety** of NN models.
Outline

1 Introduction
2 Edge TPU and Model Characterization
3 Mensa
4 Mensa-G
5 Evaluation
6 Conclusion
7 Strength & Weakness
8 Discussion
There are 3 major shortcomings of Google Edge TPU

1. Underutilization of **processing elements**

2. **Energy efficiency** far below theoretical maximum

3. Inefficient and ineffective **memory system**
Underutilization of Processing Elements

The Edge TPU utilizes only 24% of its peak throughput in average.

- CNNs and RCNNs: 40.7% of peak throughput
- LSTMs and Transducers: Less than 1% of peak throughput
The Edge TPU achieves only **37.2% of its peak energy efficiency**

CNNs and RCNNs: *50.7% of the maximum efficiency at best*

LSTMs and Transducers: *33.8% of the maximum efficiency at best*
Inefficient memory system

On-chip buffers and off-chip memory access account for a significant portion of energy use.

Off-chip parameter traffic to DRAM
46% of total energy

Parameter distribution
31% of total energy
Layer-level study of Google Edge Models

Analysis of the models in significant detail, at the granularity of individual layers

Key Insight

There are significant variations in terms of layer characteristics

1. Diversity across the models
2. Diversity within the models
Insight 1: There is **significant variance** in terms of layer characteristics across the models

- **Parameter footprint**
  - Larger footprints in LSTMs and Transducers

- **Computational complexity**
  - Lower computational complexity in LSTM and Transducers

- **Data reuse pattern**
  - No data reuse in LSTM

- **Inter/intra-cell dependencies**
  - Special dependencies exist in LSTM layers
Diversity Across the Models

**Insight 1:** There is **significant variance** in terms of layer characteristics **across** the models.

Layers from LSTMs and Transducers have significantly larger footprints than layers from CNNs.
Diversity Within the Models

**Insight 2:** Even within model, the layers exhibit **significant heterogeneity** in terms of layer characteristics

- **Parameter footprint/ Computational complexity**
  - Varies significantly within CNN models

- **Data reuse pattern**
  - Variation according to input/output activation depth and number of kernels
Diversity Within the Models

**Insight 2:** Even within model, the layers exhibit significant heterogeneity in terms of layer characteristics.

FLOP/B ratio varies across different layers of CNN models by a factor of $\times 244$. 

**Layers from CNN13**
<table>
<thead>
<tr>
<th></th>
<th>Root causes of Edge TPU shortcomings</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1</strong></td>
<td>PE Underutilization</td>
</tr>
<tr>
<td></td>
<td>• Large fixed PE array, single dataflow</td>
</tr>
<tr>
<td><strong>2</strong></td>
<td>Poor Energy Efficiency</td>
</tr>
<tr>
<td></td>
<td>• Large overprovisioned on-chip buffer</td>
</tr>
<tr>
<td></td>
<td>• High cost of off-parameter traffic</td>
</tr>
<tr>
<td><strong>3</strong></td>
<td>Memory System Issues</td>
</tr>
<tr>
<td></td>
<td>• Caching Failure</td>
</tr>
<tr>
<td></td>
<td>• Wasted energy for buffer access</td>
</tr>
</tbody>
</table>
Key Takeaways

To achieve high utilization and energy efficiency, **key components of an edge accelerator must be customized!**
### Mensa Framework

**Mensa** is a new machine learning accelerator design framework

<table>
<thead>
<tr>
<th>Problem</th>
<th>Goal</th>
<th>Mechanism</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accelerator performance is degraded due to monolithic design of the accelerator</td>
<td>Build an accelerator that can efficiently run inference across a wide range of NN models</td>
<td>a collection of smaller HW accelerators specialized toward different layer types</td>
</tr>
</tbody>
</table>
Mensa Overview

**Google Edge TPU**

- Model A
- Model B
- ... (ellipsis)
- Model C

**Mensa**

- Model A
- Model B
- ... (ellipsis)
- Model C

**Runtime Scheduler**

- Family 1
- Accelerator 1
- Family 2
- Accelerator 2
- Family 3
- Accelerator 3

Group the layers into families

Each accelerator caters to a specific family

All layers are executed in a single accelerator

Each accelerator caters to a specific family
The goal of SW runtime scheduler is to identify which accelerator each model should run on.

- Accelerator Characteristics
- Layer Characteristics
- Runtime Scheduler
- Layer Mapping

- Generated once during the initial setup
- Assigns the destination accelerator for each layer
- Also accounts for communication overhead using a simple cost algorithm
1 Introduction
2 Edge TPU and Model Characterization
3 Mensa
4 Mensa-G
5 Evaluation
6 Conclusion
7 Strength & Weakness
8 Discussion
Identifying Layer Families

We group layers into smaller number of layer families based on layer characteristics.

<table>
<thead>
<tr>
<th>Family</th>
<th>Parameter footprint</th>
<th>Data reuse rate</th>
<th>MAC intensity</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 &amp; 2</td>
<td>Low</td>
<td>High</td>
<td>High</td>
<td>Compute-centric</td>
</tr>
<tr>
<td>3, 4 &amp; 5</td>
<td>High</td>
<td>Low</td>
<td>Low</td>
<td>Data-centric</td>
</tr>
</tbody>
</table>
Identifying Layer Families

We group layers into smaller number of layer families based on layer characteristics.

<table>
<thead>
<tr>
<th>Family</th>
<th>Parameter footprint</th>
<th>Data reuse rate</th>
<th>MAC intensity</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 &amp; 2</td>
<td>Low</td>
<td>High</td>
<td>High</td>
<td>Compute-centric</td>
</tr>
<tr>
<td>3,4 &amp; 5</td>
<td>High</td>
<td>Low</td>
<td>Low</td>
<td>Data-centric</td>
</tr>
</tbody>
</table>
Hardware Accelerators

We design three accelerators based on the key characteristics of families

**Pascal**
- 32x32 PE array
  - 2 TFLOP/s
- 256kB Activation buffer
  - 8x Reduction
- 128kB Parameter buffer
  - 32x Reduction
- On-chip accelerator

**Families 1 & 2**

**Pavlov**
- 8x8 PE array
  - 128 GFLOP/s
- 128kB Activation buffer
  - 16x Reduction
- No Parameter buffer
  - 4MB in Baseline
- Near-data accelerator

**Jacquard**
- 16x16 PE array
  - 256 GFLOP/s
- 128kB Activation buffer
  - 16x Reduction
- 128kB Parameter buffer
  - 32x Reduction
- Near-data accelerator

**Families 3**

**Families 4&5**
Hardware Accelerators

We design three accelerators based on the key characteristics of families

**Pascal**
- 32x32 PE array
- 2 TFLOP/s
- 256kB Activation buffer
- 8x Reduction
- 128kB Parameter buffer
- 32x Reduction
- On-chip accelerator

**Families 1 & 2**

**Pavlov**
- 8x8 PE array
- 128 GFLOP/s
- 128kB Activation buffer
- 16x Reduction
- No Parameter buffer
- 4MB in Baseline
- Near-data accelerator

**Jacquard**
- 16x16 PE array
- 256 GFLOP/s
- 128kB Activation buffer
- 16x Reduction
- 128kB Parameter buffer
- 32x Reduction
- Near-data accelerator

**Families 4&5**
Hardware Accelerators

We design three accelerators based on the key characteristics of families

**Families 1 & 2**
- 32x32 PE array
  - 2 TFLOP/s
- 256kB Activation buffer
  - 8x Reduction
- 128kB Parameter buffer
  - 32x Reduction
- On-chip accelerator

**Pascal**
- 32x32 PE Array
- 32 GB/s
- 32kB Activation buffer
- 128kB Parameter buffer
- 32x Reduction
- On-chip accelerator

**Families 3**
- 8x8 PE array
  - 128 GFLOP/s
  - 128kB Activation buffer
  - 16x Reduction
  - No Parameter buffer
  - 4MB in Baseline
  - Near-data accelerator

**Jacquard**
- 16x16 PE Array
- 128kB Parameter buffer
  - 16x Reduction
  - Near-data accelerator

**Families 4 & 5**
- 16x16 PE array
  - 256 GFLOP/s
- 128kB Activation buffer
  - 16x Reduction
- 128kB Parameter buffer
  - 32x Reduction
- Near-data accelerator
1 Introduction
2 Edge TPU and Model Characterization
3 Mensa
4 Mensa-G
5 Evaluation
6 Conclusion
7 Strength & Weakness
8 Discussion
Energy Analysis

Baseline Google Edge TPU

Baseline Google Edge TPU + high bandwidth off-chip memory
Energy Analysis

Mensa-G improves average energy efficiency by 3.0x compared to baseline.
Mensa-G improves average throughput by 3.1x compared to baseline
More in the paper

• Details about Mensa Runtime Scheduler
• Details about Pascal, Pavlov, and Jacquard’s dataflows
• Energy comparison with Eyeriss v2
• Mensa-G’s utilization results
• Mensa-G’s inference latency results
Executive summary

**Background**
ML inference of different models are pushed to the edge devices

**Problem**
The state-of-art Edge TPU has three shortcomings
- It operates significantly below its peak throughput
- It operates significantly below its peak energy efficiency
- It has inefficient memory system

**Key Insight**
The shortcomings arise from **heterogeneity of NN layers** and **monolithic design** of the edge TPU.

**Goal**
Design an edge accelerator that can **fully exploit the variation of layers**

**Key Mechanism**
**Mensa**: a new framework for designing accelerator
- Mensa consists of a collection of smaller HW accelerators that are specialized to specific subsets of layers

**Key Results**
G-Mensa was designed and tested on Google edge NN models
- Performance and energy efficiency was improved
- Inference latency was reduced
Strengths

Novelty

- First to conduct an inter- and intra-model analysis in **layer-level granularity**
  - Quantify the **significant layer variation** in edge NN models
  - Cluster the layers
- Propose a new framework for **heterogeneous ML inference acceleration** (on-chip + near data accelerators)
  - provide and evaluate an example heterogeneous accelerator design for Google edge NN models

Mechanism

- **Well-designed** specialized hardware based on layer clusters
- **Practical and applicable**
  - Mensa built based on an existing framework
  - **Flexibility**: accelerators and schedulers can be tailored to different target models
Strengths

Evaluation

- Well analyzed the shortcomings of Google Edge TPU and its causes
- Successfully identified layer characteristic to design specialized HW accelerators
- Created an example design to thoroughly analyze the performance

Results

- Achieved high performance and energy efficiency compared to Google Edge TPU and Eyeriss v2
Weaknesses

Mechanism

- Framework built based on **existing NN models**
  - New models/families of layers might emerge
- **Suboptimal scheduler algorithm** could lead to potential scheduler overhead
  - Use of very simple cost algorithm might cause inefficiency
- **Underutilization of HW resources**
  - Concurrent operation of multiple HW accelerators was not considered

Evaluation

- Evaluation based only on 24 Google NN models
- **Simulated evaluation results**, not considering other HW constraints
Discussion

Would multi-accelerator framework be the optimal solution of rising heterogeneity of neural network models?

- Better scheduling
- Increase on-chip memory
- Better on-chip interconnect
- Higher memory bandwidth
Discussion

Could we think of a better scheduling algorithm for Mensa framework?

- Mensa scheduling algorithm

- Phase 1
  - Identifies the ideal HW accelerator for each layer

- Phase 2
  - Use a simple cost algorithm, sequentially map the layers
  - For layer $i$
    - If communication cost is higher than the penalties using $i-1$ accelerator, use $i-1$ accelerator
    - Else, use the ideal accelerator
What would be the main challenge of edge ML acceleration?

- Moore’s law slowing down
  - Using more transistor to boost performance might lead to power and cost rise

- Neural Network workloads evolving rapidly
  - Different footprints, data reuse patterns, data type
  - Programmability and flexibility

- Combination of HW accelerators and SW platforms
  - Google Edge TPU can only run Tensorflow models
  - Development of workflow to build cloud-to-edge pipelines
  - Developing new models appropriate for edge computation
Figure 1. Throughput roofline (left) and energy roofline (right) for the Edge TPU across all Google edge neural network models.