Acceleration of Approximate KNN Indexing on High-Dimensional Vectors

This project has received funding from the Hellenic Foundation for Research and Innovation (HFRI) and the Genal Secretariat for Research and Technology (GSRT) under grant agreement no 2212- CloudAccel

https://cloudaccel.github.io/

Dimitrios Danopoulos
Christoforos Kachris
Dimitrios Soudris
Motivation

- Performance
- Reliability
- Low Power
Motivation

❖ Performance
❖ Reliability
❖ Low Power

Software: Machine Learning
Motivation

- Performance
- Reliability
- Low Power

Software: Machine Learning

Hardware: FPGAs
Overview

01 Introduction
Typical KNN, Approximate KNN, Programming with OpenCL-FPGAs

02 A case study on FAISS framework
FAISS operation, Application profiling, Choosing function to accelerate

03 FPGA Accelerator Design
Algorithm implementation, HW/SW optimizations, Integration with FAISS

04 Evaluation and Results
Performance of HW function, Performance of full system, Real-world metrics
Introduction

Typical KNN (K-Nearest Neighbor) algorithm

- A classification algorithm that finds the k closest points to an object
- Euclidean distance is often used as the distance metric
- It is exhaustive because all distances from the object need calculation
Introduction

Typical KNN (K-Nearest Neighbor) algorithm

• A classification algorithm that finds the k closest points to an object
• Euclidean distance is often used as the distance metric
• It is exhaustive because all distances from the object need calculation

It’s a lazy algorithm with high computation cost.
Impractical for the industry!
Introduction

Approximate KNN algorithm

- A classification algorithm that approximates the k closest points to an object
- It uses techniques like data reduction, vector quantization, clustering, etc.
- It is often as good as the exact one!

Low memory and computation cost.
Ideal for the industry.
Introduction

Approximate KNN algorithm

- A classification algorithm that approximates the k closest points to an object
- It uses techniques like data reduction, vector quantization, clustering, etc.
- It is often as good as the exact one!

Low memory and computation cost.
Ideal for the industry.

Training of approximate KNN is challenging!
Introduction

Programming with OpenCL-FPGAs
A case study on FAISS framework

FAISS (Facebook AI Similarity Search)

What is it?
A library for efficient similarity search and clustering of dense vectors written in C++/Python.

How it does it?
It’s built around an index type that stores and clusters vectors and then it can search more efficiently.

The tradeoff:
Ultra-fast similarity search
Long Index creation times

Execution times
- Train
- Search
- Other
A case study on FAISS framework

Choosing function for acceleration

- Training involves building an index that clusters data into Voronoi cells.
- It has many MAC operations. Specifically BLAS-level3 functions. (i.e. GEMM)

Tip

At search time only the vectors contained in the cell that the query is mapped to along with a few neighboring ones are compared

Voronoi partition: for each centroid above there is a corresponding region consisting of all points closer to that point
A case study on FAISS framework

Choosing function for acceleration

- Training involves building an index that clusters data into Voronoi cells.

- It has many MAC operations. Specifically BLAS-level3 functions. (i.e. GEMM)

  Ideal candidate for FPGAs!

Tip

Voronoi partition: for each centroid above there is a corresponding region consisting of all points closer to that point

At search time only the vectors contained in the cell that the query is mapped to along with a few neighboring ones are compared
FPGA Accelerator Design

Algorithm Design

1. Define a scalable function for acceleration
   custom column-major GEMM

2. Optimize host, kernels and memory

3. Integrate it with FAISS framework

The mathematical equation of our GEMM

\[ C = \alpha \ast T(A) \ast B + \beta \ast C \]
FPGA Accelerator Design

Algorithm Design

M dim: list size of the index
Defines how many Voronoi cells we partition from the database vectors.

K dim: the vector dimension

N dim: the number of vectors to cluster

column-major GEMM
FPGA Accelerator Design

- Optimizations

- OpenCL kernel task sync
- 512-bit interface
- Use all 4 DDRs
- Traverse both arrays in K dim
- Pipeline with II=1 in every loop
- Allocate BRAMs for faster memory
FPGA Accelerator Design

- **Optimizations**

  - OpenCL kernel task sync
  - 512-bit interface
    - Use all 4 DDRs
  - Traverse both arrays in K dim
  - Pipeline with II=1 in every loop
  - Allocate BRAMs for faster memory

**HOST**

- Make device context global

**Memory**

- Transfer bigger chunks of memory to DDRs
- Ensure data is in physically contiguous memory

**Kernel**

- Avoid SLR crossing
- Create larger CUs than more and smaller
FPGA Accelerator Design

- Integration with FAISS
Evaluation and Results

- **Kernel freq**: 300MHz
- **DDRs used**: 4
- **Kernels**: 4

**Performance of HW**

- 115x vs single CPU (accelerator-only)
- 2.4x vs 36-thread Xeon (end-to-end execution)

**FPGA utilization summary per kernel**

<table>
<thead>
<tr>
<th>Name</th>
<th>BRAM</th>
<th>DSP</th>
<th>FF</th>
<th>LUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total</td>
<td>502</td>
<td>1004</td>
<td>157131</td>
<td>89096</td>
</tr>
<tr>
<td>(%)</td>
<td>11</td>
<td>14</td>
<td>6</td>
<td>7</td>
</tr>
</tbody>
</table>

Good enough?

Yes, because most datasets usually need more than ~2K cells to cluster them sufficiently.
Evaluation and Results

- Real-world metrics

Comparing accuracy for two kinds of indices

1. The fraction of the total vectors we search from database is \( \frac{n_{probe}}{n_{cell}} \)

2. So, for larger probe we have better accuracy as we search more cells

3. Our FPGA accelerated indices do not suffer from accuracy drop. They maintain same accuracy for same \( \frac{n_{probe}}{n_{cell}} \) fraction

“R-recall at R” is used to measure KNN model accuracy
Evaluation and Results

Power metrics

Power efficiency comparison

- $4.1 \frac{GFLOPs}{Watt}$ vs Xeon CPU
- $1.4 \frac{GFLOPs}{Watt}$ vs Kepler-Class GPU

Conclusion

Better performance and performance/watt
Thank you!

This project was funded from the Xilinx University program and the Hellenic Foundation for Research and Innovation (HFRI) and the General Secretariat for Research and Technology (GSRT), under grant agreement No 2212- Hardware Acceleration of Machine Learning Applications in the Cloud.