Deploying Deep Neural Networks in the Embedded Space

Stylianos I. Venieris, Alexandros Kouris, Christos-Savvas Bouganis

2nd International Workshop on Embedded and Mobile Deep Learning (EMDL)

MobiSys, 15 June 2018
Who we are

Stylianos I. Venieris
Machine Learning

Alexandros Kouris
Machine Learning, Robotics

Konstantinos Boikos
Computer Vision, SLAM

Manolis Vasileiadis
Computer Vision

Mudhar Bin Rabieah
Machine Learning

Nur Ahmadi
Brain-Machine Interface

Christos-Savvas Bouganis
Lab Director
Reader at
Imperial College London
Focus: Couple the design of the ML algorithm with the design of the computational platform to improve performance and enable the deployment of AI systems.

- Absolute power consumption
- Performance-per-Watt
Conventional Embedded Platforms for Neural Networks

GPUs – Tegra K1, X1 and X2
DSPs – Qualcomm Hexagon,
Apple Neural Engine, ...

FPGAs
- Custom datapath
- Custom memory subsystem
- Programmable interconnections
- Reconfigurability

✓ High throughput
✗ Low latency
✗ Low power

✓ High throughput
✓ Low latency
✓ Low power

Challenge: Huge design space
Our Approach: Automated toolflows
Research Areas / Challenges

- Mapping Automation
- Multiple CNN Mapping
- Time-constrained Inference
- Privacy-aware Deep Learning
Challenge #1: Mapping Automation
Challenge #1: Mapping Automation

Little knowledge about FPGAs
Ease of deployment
“Good” designs

Deep Learning Developers

Challenges:
- High-dimensional design space
- Diverse application-level needs
- Utilise the FPGA resources
- Design automation

Would like to:
- Target FPGAs
- Optimise for high performance
Challenge #1: Automated CNN-to-FPGA Toolflow

- Network Description
- Performance Requirements
- FPGA Target Platform Specifications

Automated Design Space Exploration

Network Hardware Mapping

Supplied by Deep Learning Expert
• Synchronous Dataflow Modelling
  – Capture hardware mappings as matrices
  – Transformations as algebraic operations
  – Analytical performance model
  – Cast design space exploration as a mathematical optimisation problem

\[
\begin{align*}
\mathbf{r}_1 &= \begin{bmatrix}
1 & -1 & 0 & 0 & 0 & 0 & 0 \\
0 & K^2 & -K^2 & 0 & 0 & 0 & 0 \\
0 & 4K^2 & -4K^2 & 0 & 0 & 0 & 0 \\
0 & 0 & 4 & -4 & 0 & 0 & 0 \\
0 & 0 & 0 & 4 & -4 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 4 & -4 \\
0 & 0 & 0 & 0 & 0 & 0 & 4p^2 - 4p^2
\end{bmatrix} \\
\mathbf{r}_2 &= \begin{bmatrix}
1 & -1 & 0 & 0 & 0 & 0 & 0 \\
0 & K^2 & -K^2 & 0 & 0 & 0 & 0 \\
0 & 4K^2 & -4K^2 & 0 & 0 & 0 & 0 \\
0 & 0 & 2K^2 & -2K^2 & 0 & 0 & 0 \\
0 & 0 & 2 & -2 & 0 & 0 & 0 \\
0 & 0 & 0 & 2 & -2 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 2p^2 - 2p^2
\end{bmatrix}
\end{align*}
\]

\[
t_{\text{total}}(B, N_P, \Gamma) = \sum_{i=1}^{N_P} t_i(B, \Gamma_i) + (N_P - 1) \cdot t_{\text{reconfig}}.
\]
Meeting the performance requirements
Comparison with Embedded GPUs: Same absolute power constraints (5W)

fpgaConvNet vs Embedded GPU (GOp/s) for the same absolute power constraints (5W)

- Latency-driven scenario $\Rightarrow$ batch size of 1
- Up to 6.65× speedup with an average of 3.95× (3.43× geo. mean)

- Throughput-driven scenario $\Rightarrow$ favourable batch size
- Up to 5.53× speedup with an average of 3.32× (3.07× geo. mean)
Challenge #2: Multi-CNN Systems
Challenge #2: Multi-CNN Systems – Autonomous Drones

Camera

Set of CNNs
- Object Detection
- Semantic Segmentation
- Navigation
- Monitoring
- Domain Task

Target Platform
- FPGA
- GPU
- DSP

Mapping?
Challenge #2: Multi-CNN System

Challenges:
- Resource allocation among CNNs
- Design automation

Why?
- Models with different performance constraints, e.g. required throughput and latency
- Competing for the same pool of resources
- High-dimensional design space

\[ f\text{-CNN}^x \]

Set of CNNs → Per-CNN Performance Requirements → Optimised Mapping → Target Platform Specifications

Supplied by Deep Learning Expert
Multi-CNN FPGA design

- One customised hardware engine per CNN
- Explore both on-chip resource allocation and different memory access schedules
Comparison with Embedded GPUs: Same absolute power constraints (5W)

- Latency-driven scenario → batch size of 1
- Up to 9.68× speedup with an average of 5.25× (geo. mean)

- Latency-driven scenario → batch size of 1
- Up to 19.09× speedup with an average of 6.85× (geo. mean)
Challenge #3: Time-constrained Inference
Challenge #3: Time-constrained Inference

![Image of a city street scene with cars and traffic lights]

- Camera/Sensor
- CNN
- LSTM
- Decision/Action

Diagram showing the process of processing data from a camera/sensor through a CNN and LSTM to a decision/action.

Graph illustrating the metric of interest over time, comparing current approaches to a target.
• Approximate LSTMs
  – Iterative refinement using SVD + Pruning.
  – Parametrised with respect to:
    • Number of iterations
    • Level of pruning

• Parametrised hardware architecture, tailored for approximate LSTMs

• Co-optimise given a user-defined time budget
Impact on LSTM-based Image Captioning

Input Image

![Input Image](image_url)

---

0) a brown dog laying on top of a piece of luggage. (p=0.000051)
1) a brown dog laying on top of a pile of luggage. (p=0.000042)
2) a brown dog laying on top of a pile of shoes. (p=0.000028)
3) a brown dog laying on top of a pile of books. (p=0.000015)
4) a brown dog laying on top of a pile of shoes. (p=0.000001)
Impact on LSTM-based Image Captioning

Input Image

0) a man is sitting on a <UNK> with a <UNK>. (p=0.000000)
1) a man is sitting on a <UNK> with a <UNK>. (p=0.000000)
2) a man is sitting on a <UNK> with a small dog. (p=0.000000)
3) a man is sitting on a <UNK> with a small dog. (p=0.000000)
4) a man is sitting on a <UNK> on the ground. (p=0.000000)

[Graph showing BLEU scores over runtime for different NZ values]

NZ = 64
NZ = 128
NZ = 256
NZ = 512
NZ = 1024
Base model
Impact on LSTM-based Image Captioning

Input Image

0) a man is sitting on a <UNK> with a <UNK>. (p=0.000000)
1) a man is sitting on a <UNK> with a <UNK>. (p=0.000000)
2) a man is sitting on a <UNK> with a small dog. (p=0.000000)
3) a man is sitting on a <UNK> with a small dog. (p=0.000000)
4) a man is sitting on a <UNK> with a <UNK> on the ground. (p=0.000000)

0) a brown dog laying on top of a pile of luggage. (p=0.000031)
1) a brown dog laying on top of a pile of shoes. (p=0.000015)
2) a brown dog laying on top of a rug. (p=0.000015)
3) a brown dog laying on top of a pile of clothes. (p=0.000010)
4) a dog is laying on the floor next to a stuffed animal. (p=0.000007)
Challenge #4: Privacy-aware Deep Learning
Challenge #4: Privacy-restricted Optimisation

**Aim:** Design an optimised HW system (performance and accuracy)

**Given:**
- A High-Level CNN Description (i.e. Caffe)
- A target FPGA platform
- Train\(\times\) Data privacy, availability
- Testing Data
- Target metric (top1/top-5 accuracy, ...)

➡️ quantisation with retraining step

*Limited quantisation opportunities*
Challenge #4: Privacy-aware Deep Learning

The graph shows the Top-5 ImageNet Accuracy (%) for different wordlengths (bits) for VGG-16, AlexNet, and CompRoof. The x-axis represents the wordlength in bits (16, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2), and the y-axis represents the Top-5 ImageNet Accuracy (%). The performance (GOp/s) is shown on the right side of the graph, ranging from 0 to 2500.
• Pushing quantization below limits of acceptable accuracy to gain performance (high throughput)
• Evaluation of Quality of Prediction to identify and correct error introduced by quantization
Challenge #4: Privacy-aware Deep Learning

- CNN Description
- FPGA Platform
- Layers, Weights, ...
- LUTs, DSPs, Memory BW, On-chip, ...

Layers:
- QUANTISATION
- EVALUATOR
- Roofline Model
- HW Architecture

HighPrec, LowPrec

Validation Set

User-Input: maxError

FPGA Implementation
Challenge #4: Privacy-aware Deep Learning

![Graph showing speed-up vs classification error](image)

- VGG-16, Zynq
- VGG-16, UltraScale+
- AlexNet, Zynq
- AlexNet, UltraScale+

Classification Error compared to a faithful 8-bit implementation (%)
Research topics

- Mapping Automation
- Multiple CNN Mapping
- Time-constrained Inference
- Privacy-aware Deep Learning

www.imperial.ac.uk/idsl
Alexandros Kouris, Stylianos I. Venieris, and Christos-Savvas Bouganis. 2018. **CascadeCNN: Pushing the performance limits of quantisation.** In SysML.

Alexandros Kouris, Stylianos I. Venieris, and Christos-Savvas Bouganis. 2018. **CascadeCNN: Pushing the Performance Limits of Quantisation in Convolutional Neural Networks.** In 2018 28th International Conference on Field Programmable Logic and Applications (FPL).


S. I. Venieris and C. S. Bouganis. 2018. **f-CNNx: A Toolflow for Mapping Multiple Convolutional Neural Networks on FPGAs.** In 2018 28th International Conference on Field Programmable Logic and Applications (FPL).