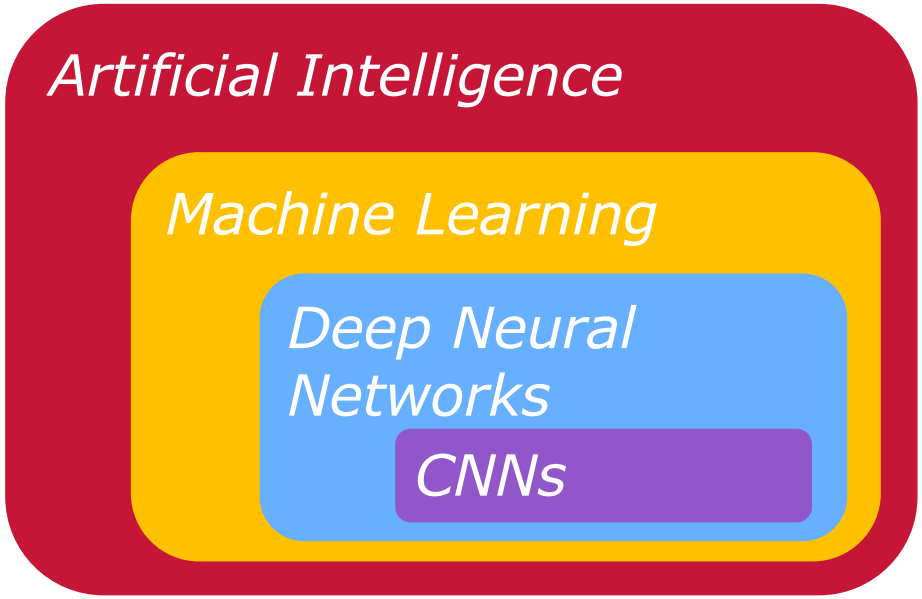
Imperial College London

Tool-flows for mapping CNNs into FPGAs: Trends and Challenges

Christos Bouganis
ccb98@ic.ac.uk

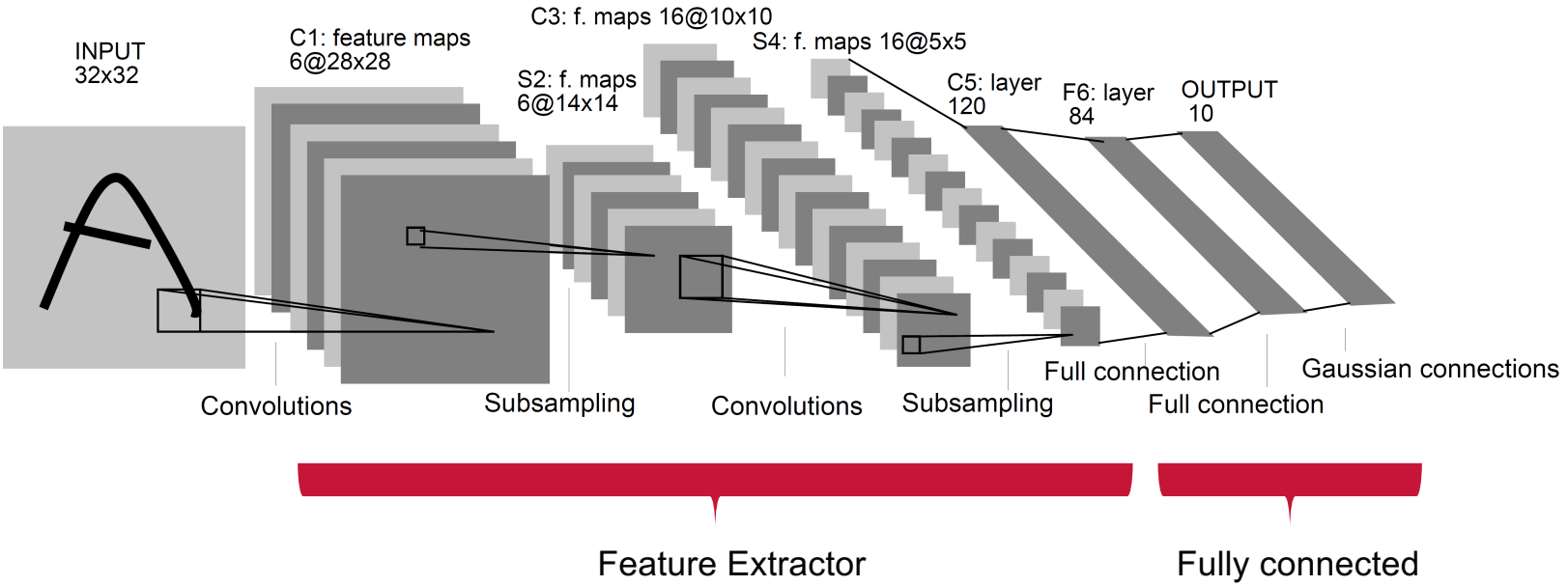
Intelligent Digital Systems Lab
Electrical and Electronic Engineering Department
Imperial College London


Artificial Intelligence
Machine Learning
Deep Neural Networks
CNNs

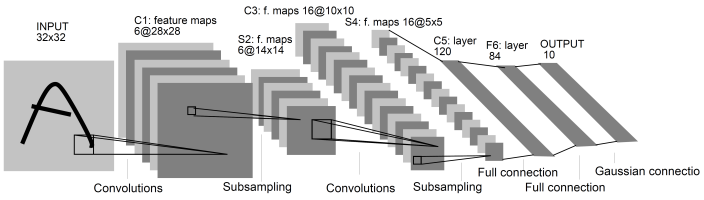

YOLO v2
http://pjreddie.com/yolo

YOLO v2

Convolutional Neural Network



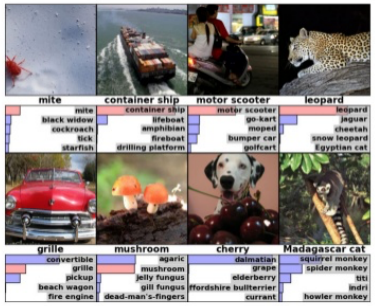
Convolutional Neural Network - Trends



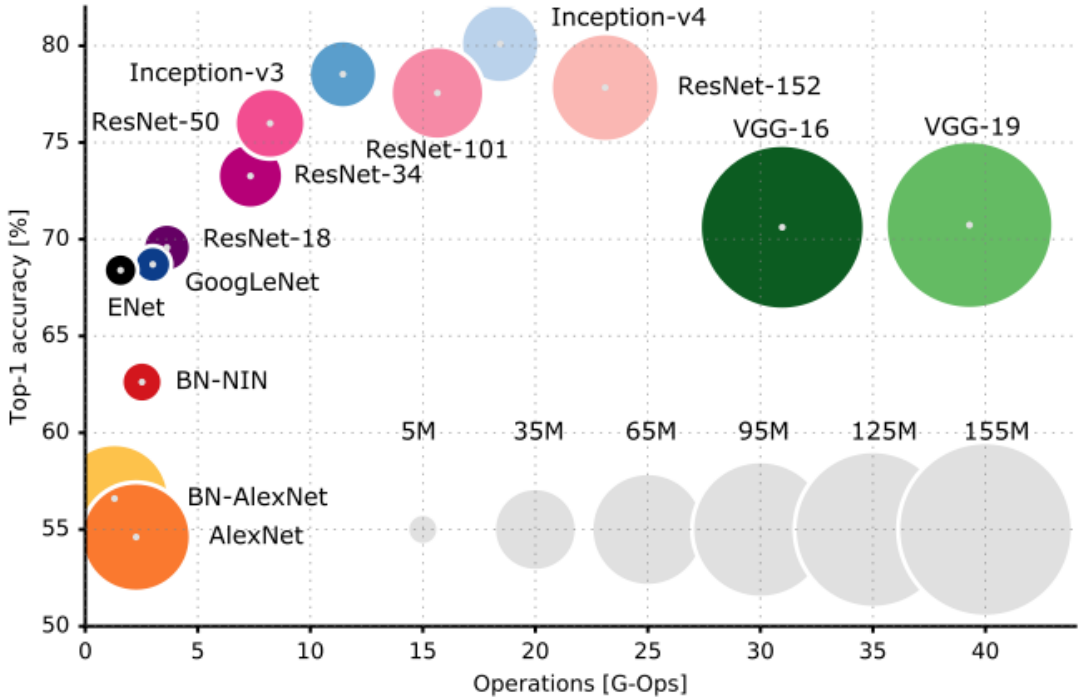
ImageNet Challenge

IMAGENET

- 1,000 object classes (categories).
- Images:
  - 1.2 M train
  - 100k test.



4



A Deep Learning Software Ecosystem

First Wave

Second Wave

Caffe
(UC Berkeley)

Caffe2
(Facebook)

Microsoft CNTK
(Microsoft)

TensorRT
(NVIDIA)

torch
(NYU / Facebook)

PYTORCH
(Facebook)

mxnet
(Amazon)

Neural Network Toolbox
MATLAB

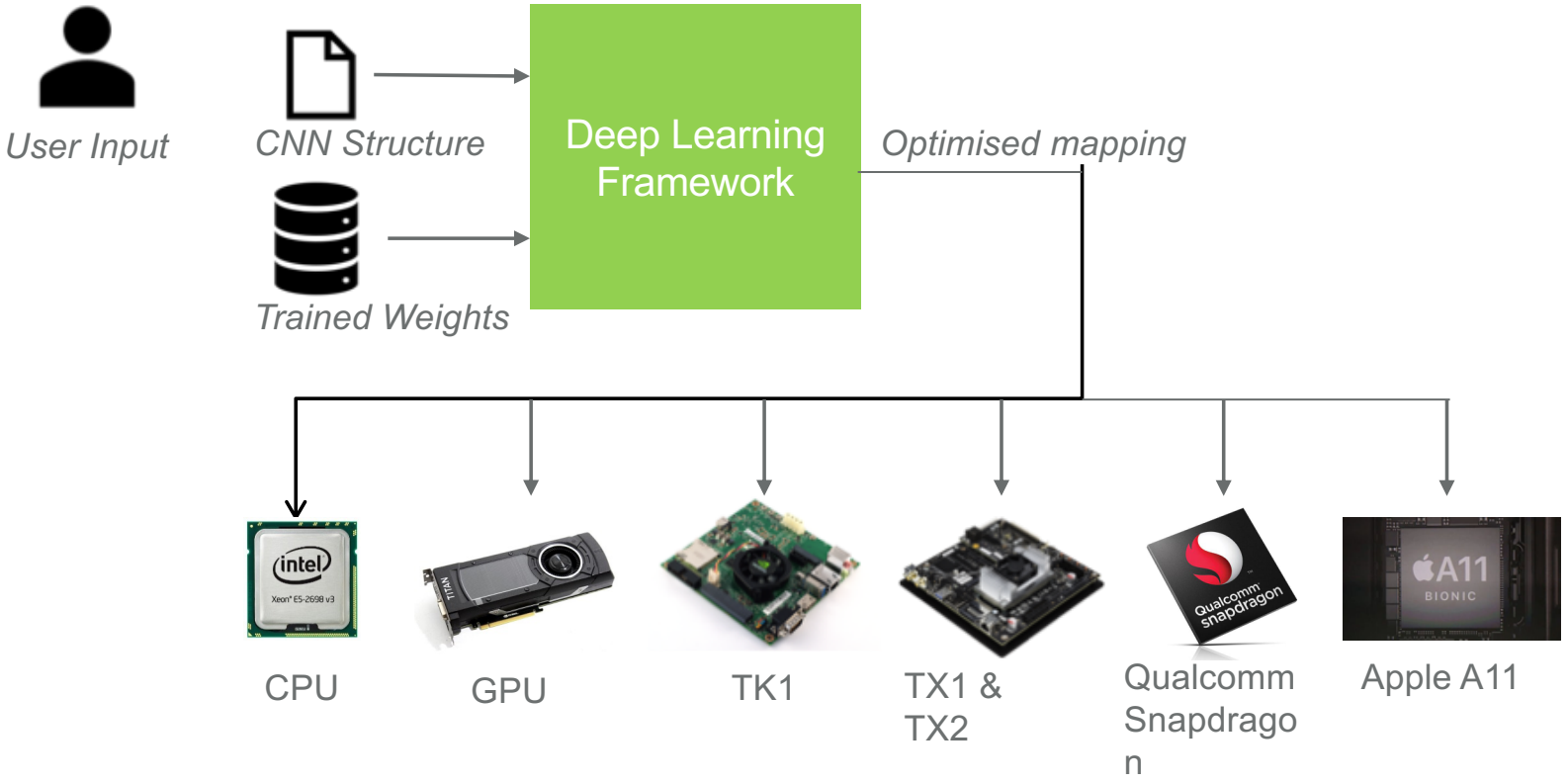
theano
(Univ. of Montreal)

TensorFlow
(Google)

CoreML
(Apple)

Training and Inference

CNN Deployment Flow



CNN Deployment Flow

FPGA platform resources

User Input

CNN Structure

Trained Weights

Deep Learning Framework

Optimised mapping

CPU

GPU

TK1

TX1 & TX2

Qualcomm Snapdragon

Apple A11

FPGA

CNNs: An opportunity for tool-flows for Reconfigurable Hardware

In the last few years, significant progress in generic FPGA HLS tools

- Vivado HLS, Intel OpenCL, MaxCompiler, LegUp, etc.
- Generate designs based on the mapping and scheduling of low-level primitive operations → Large design space.
- Low-level entry point

CNN workloads are highly structured

- Layers with pre-defined types and parametrisation

Opportunity for domain-specific frameworks for CNNs

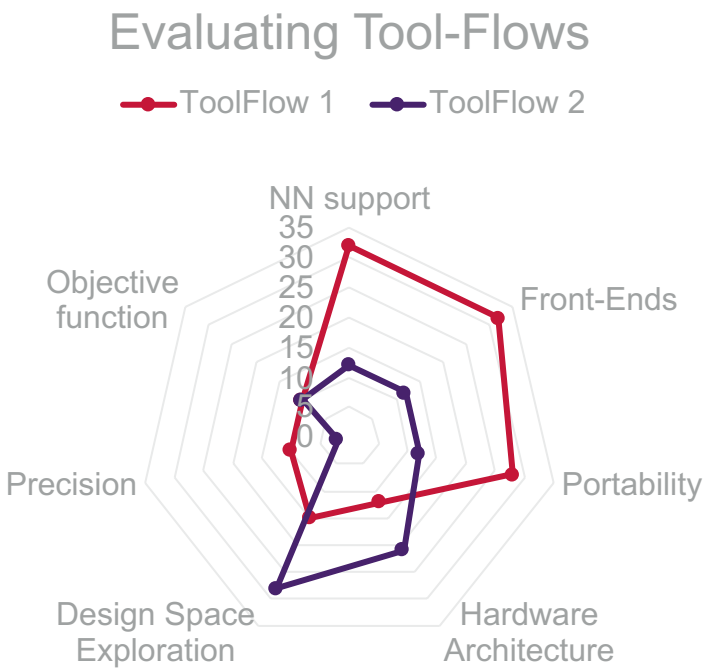
- Generate optimised architectures

Existing CNN-to-FPGA tool-flows

Name	*Input description*	*Year*	*Publication*
fpgaConvNet	Caffe & Torch	May 2016	FCCM 2016
DeepBurning	Caffe	June 2016	DAC 2016
Angel-Eye	Proprietary	July 2016	FPGA 2016
ALAMO	Proprietary	August 2016	FPL 2016
DnnWeaver	Caffe	October 2016	MICRO 2016
Caffeine	Caffe	November 2016	ICCAD 2016
FINN	Theano	February 2017	FPGA 2017
FP-DNN	TensorFlow	May 2017	FCCM 2017
SysArrayAccel	C	June 2017	DAC 2017

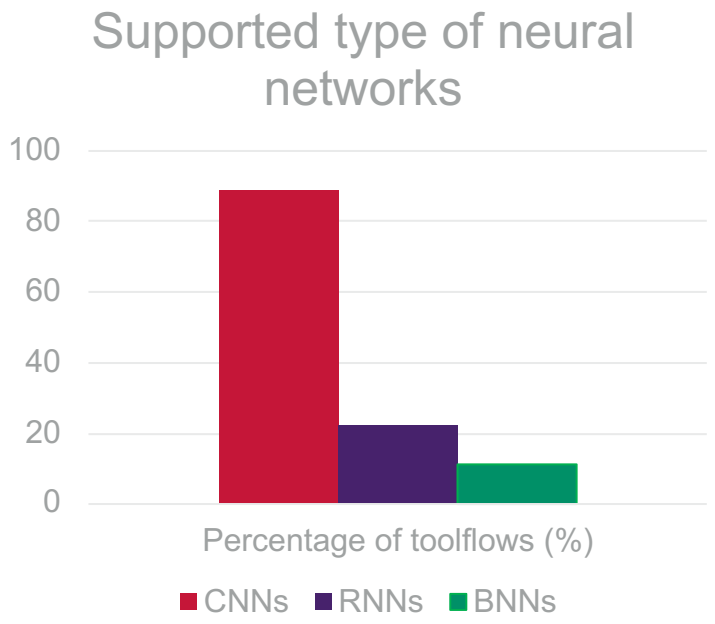
Key aspects for consideration – How do they compare?

- Key aspects for consideration
  - Neural Network Support
  - Front-End Support
  - Design Portability
  - Hardware Architectures
  - Design Space Exploration
  - Precision support
  - Objective function



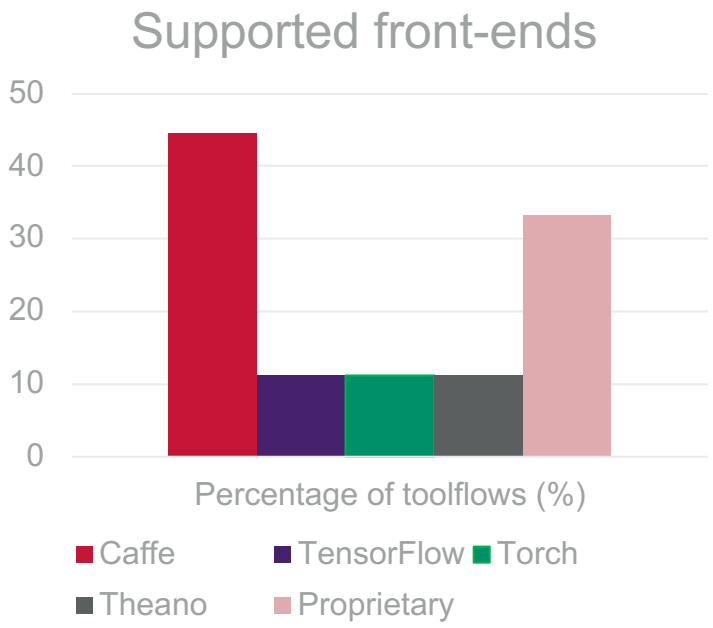
Supported Neural Network models

- Mainstream models include
  - CNNs
  - Recurrent Neural Networks (RNNs)
  - Binarised Neural Networks (BNNs)
  - Ternary Neural Networks
- DeepBurning and FP-DNN
  - RNNs
  - LSTMs
  - Residual connections in CNNs (FP-DNN)



Supported Front-Ends

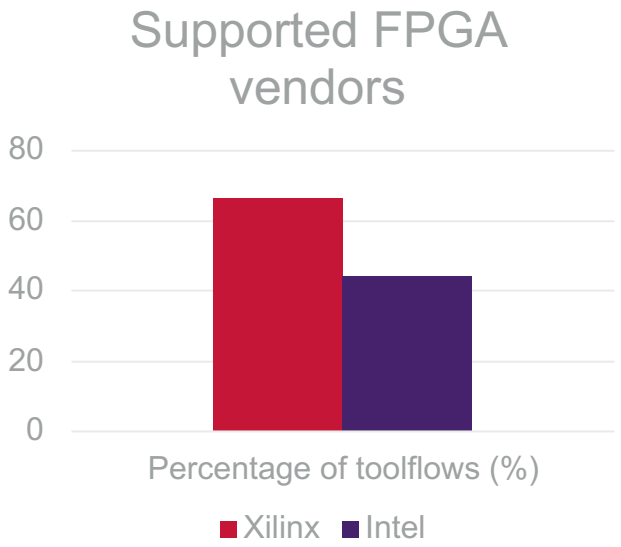
- Critical for reaching a wide user base
- Two options
  - Integration with existing frameworks
    - FpgaConvNet, DeepBurning, DNNWeaver Caffeine
  - Proprietary front-ends
    - SysArrayAccel, Angle-Eye, CNN RTL

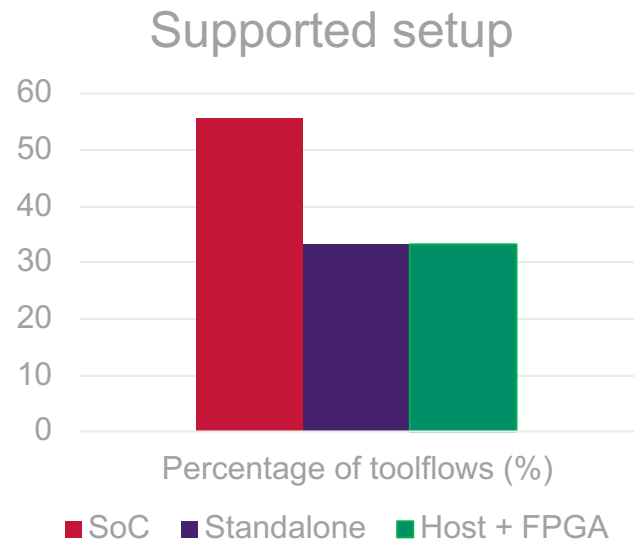


Design Portability

Def.: The degree to which a tool-flow can target:

1. devices by multiple vendors and families
2. different setups (SoCs, host-FPGA servers, stand-alone FPGAs).





Highest portability: DnnWeaver

Hardware Architecture

Two types of architectures

1. *Streaming architectures*

2. *Single computation engine*

Hardware Architecture – Streaming Architectures

Characteristics:

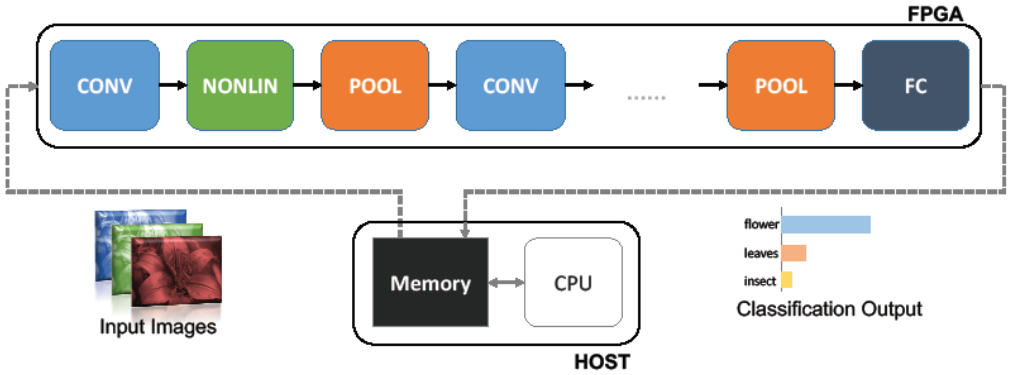
- Coarse pipeline of hardware stages
- One hardware stage per layer
- *fpgaConvNet, DeepBurning, FINN*

Advantages:

+ Customisation
+ Concurrent execution of layers
+ Flexible allocation of resources per layer, tailored to the target network

Disadvantages:

– Flexibility
– New bitsream for each CNN → long compilation times



Hardware Architecture – Single Computation Engine

Characteristics:

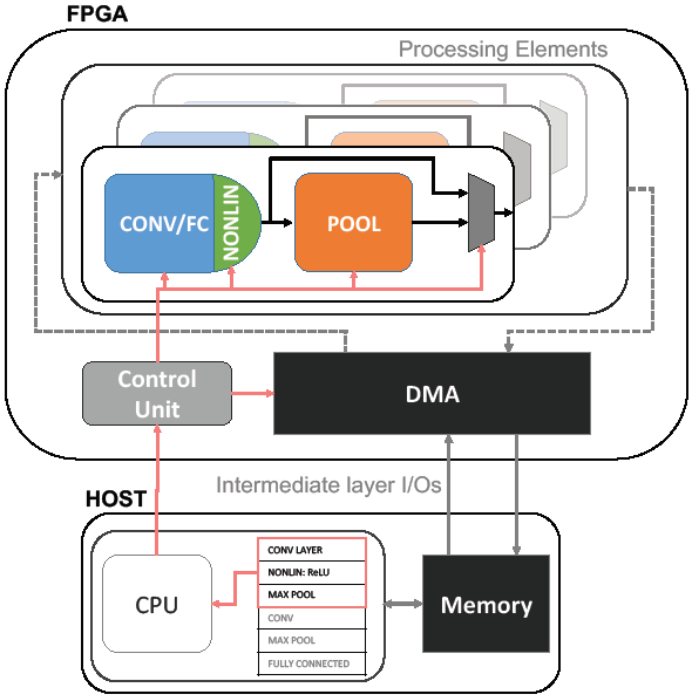
- Processing element-based design
- Fixed architecture, time-shared between layers
- Control via microinstructions
- *Angel-Eye, ALAMO, DnnWeaver, Caffeine, FP-DNN, SysArrayAccel*

Advantages:

\+ Flexibility
\+ One bistream can target many CNNs

Disadvantages:

– Customisation
– High performance variability across CNNs
– Inefficiencies due to processor-like control mechanisms



Design Space Exploration

Each toolflow defines an architectural design space

- *Parameter Space:*
  - » Which trade-offs and alternative designs can be explored?

- *Exploration Method*
  - » How it the design space traversed?
  - » Which objectives are optimised?

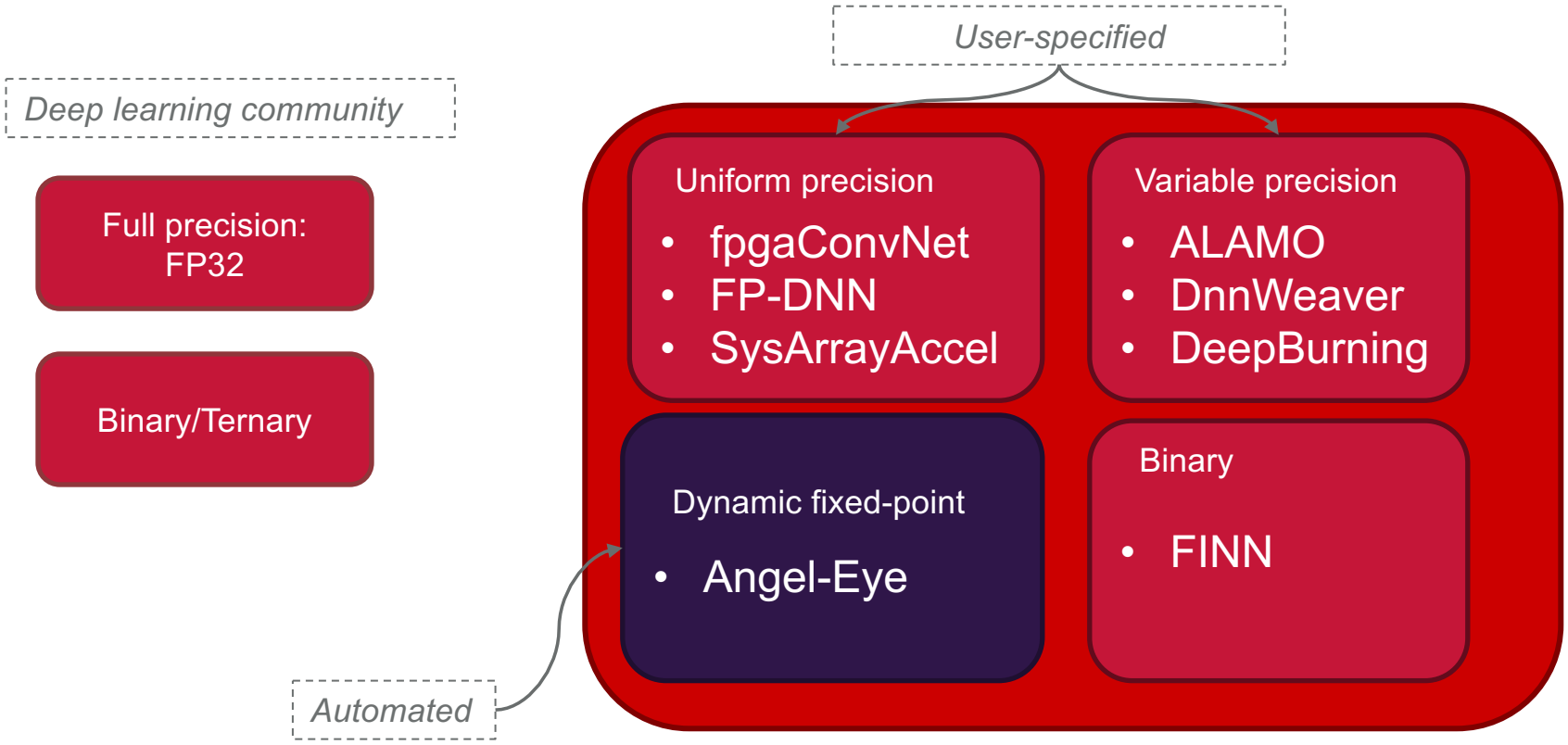
Observations

- Toolflows with streaming architectures define a finer-grained space.
  - » Structure of the pipeline
  - » Allocation of resources among hardware stages
- Single computation engine toolflows focus on:
  - » Scaling of the computation engine with HW resources
  - » CNN-to-microinstructions mapping

Design Space Exploration

Toolflow	Formulation	Solver	Objectives
fpgaConvNet	Mathematical optimisation s.t. the rsc budget	Global Optimiser (Simulated Annealing)	Throughput, latency or multiobjective criteria
DnnWeaver	-//-	Heuristic	Throughput
Caffeine	Roofline model	Enumeration	Throughput
SysArrayAccel	Mathematical optimisation s.t. the rsc budget	Pruning + Enumeration	Throughput
FINN	Rule-based: rate-balancing	Heuristic	Throughput, latency
FP-DNN	Rule-based: bandwidth-matching	Heuristic	Throughput
ALAMO		Heuristic	Throughput, latency
DeepBurning	Rule-based: rate-balancing	Heuristic	Throughput, latency
Angel-Eye	Rule-based	Heuristic	Throughput, latency

Arithmetic Precision



Performance (Quality of Result)

Achieved QoR is the most critical factor.

QoR can be evaluated with respect to two factors:

1) Comparison of QoR between toolflows for the same CNN-FPGA pair,
2) Comparison with hand-tuned accelerator for the same CNN-FPGA pair.

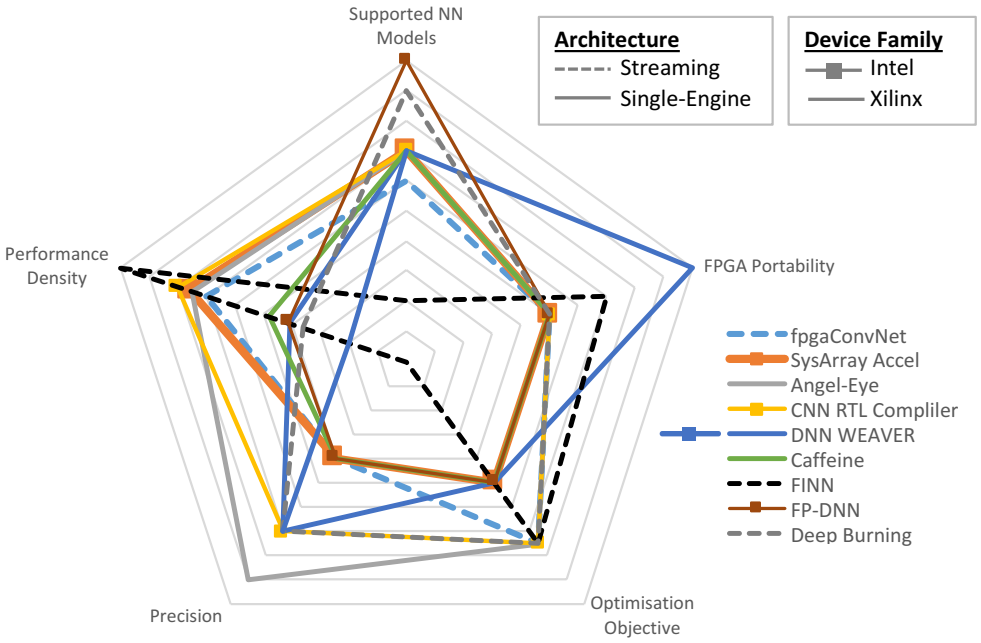
Fair comparisons: Each toolflow to target the same CNN-FPGA pair.

However, the majority of existing toolflows are not open-sourced, or provide limited support.

- DnnWeaver targeting Zynq XC7Z020 (limited support, open sourced)
- FINN targeting Xilinx PYNQ-Z1 board (specific BNNs)
- Angel-Eye used internally by DeePhi.
- fpgaConvNet (webpage with up-to-date benchmarking results)

High-level Performance Observations

1) RTL-based designs tend to outperform their HLS counterparts.
2) Finer-grained DSE tends to offer an advantage in terms of obtained performance.
3) Single computation engines tend to reach higher performance for CNNs with a uniform structure.
4) Comparable or even better performance than hand-tuned designs



THE FUTURE OF CNN-TO-FPGA TOOLFLOWS

Objectives of a CNN-to-FPGA Toolflow

Objective 1. Targeting next-generation CNN models.

Trends:

1) Increased depth.
   » 8-layer AlexNet (2012) → 152-layer ResNet-152 (2016), 161-layer DenseNet-161 (2017)
2) Increased workload.
   » 20x in GOps/input from AlexNet (2012) to VGG-16 (2014)
3) Novel compound components.
   » Inception module (GoogLeNet), residual block (ResNet), dense block (DenseNet), residual Inception block (Inception-v4).

Challenge

- Irregular layer connectivity challenges the automation of mapping to hardware.

Objectives of a CNN-to-FPGA Toolflow

Objective 1. Targeting next-generation CNN models.

Support for Recurrent Neural Networks (RNNs):

- TPU paper by Google – around 95% of NN workloads are RNNs.

Challenge

- RNNs are memory-bounded and require different design approach to CNNs.

Objectives of a CNN-to-FPGA Toolflow

Objective 2. Support for compressed and sparse CNNs

Numerous schemes exploit redundancy in the model to reduce inference time.

- Low-rank approximations, pruning, sparsification, quantisation.
- ASIC accelerators have introduced designs for such networks (e.g. zero-skipping compute units, bit-serial arithmetic, etc.)

Challenge

- Methods such as pruning can break the uniformity of computation and memory accesses.

Objectives of a CNN-to-FPGA Toolflow

Objective 3. FPGA-based CNN training

GPUs are the current standard for CNN training.

- Next-generation FPGAs demonstrate promising performance and power efficiency (Stratix 10, UltraScale).
- Recent advances in low-precision NN training offers space for customisation and variable-precision arithmetic that suits FPGAs.
- Slightly explored by the F-CNN framework.

Challenge

- Demonstrate gains of FPGAs over GPUs for CNN training.

Objectives of a CNN-to-FPGA Toolflow

Objective 4. Hardware-Network co-design

End-to-end toolchain, from dataset and application to network and hardware design.

- Expose hardware performance and power consumption to the training phase, co-optimise the CNN model and the hardware under a holistic view.

Challenge

- Long-term objective for the community towards the efficient hardware execution of high-performing neural networks.

Thank you - Questions

…soon published in *ACM Computing Surveys*

Toolflows for Mapping Convolutional Neural Networks on FPGAs: A Survey

STYLIANOS I. VENIERIS, ALEXANDROS KOURIS AND CHRISTOS-SAVVAS BOUGANIS,
Imperial College London

In the past decade, Convolutional Neural Networks (CNNs) have demonstrated state-of-the-art performance in various Artificial Intelligence tasks. To accelerate the experimentation and development of CNNs, several software frameworks have been released, primarily targeting power-hungry CPUs and GPUs. In this context, reconfigurable hardware in the form of FPGAs constitutes a potential alternative platform that can be integrated in the existing CNN ecosystem to provide a tunable balance between performance, power consumption and programmability. In this paper, a survey of the existing CNN-to-FPGA toolflows is presented, comprising a comparative study of their key characteristics which include the supported applications, architectural choices, design space exploration methods and achieved performance. Moreover, major challenges and objectives introduced by the latest trends in CNN algorithmic research are identified and presented. Finally, an evaluation methodology is proposed, aiming at the comprehensive, complete and in-depth evaluation of CNN-to-FPGA toolflows.

References

- Stylianos I. Venieris and Christos-Savvas Bouganis. 2016. fpgaConvNet: A Framework for Mapping Convolutional Neural Networks on FPGAs. In FCCM. DOI:http://dx.doi.org/10.1109/fccm.2016.22
- Stylianos I. Venieris and Christos-Savvas Bouganis. 2017. fpgaConvNet: Automated Mapping of Convolutional Neural Networks on FPGAs (Abstract Only). In FPGA. DOI:http://dx.doi.org/10.1145/3020078.3021791
- Stylianos I. Venieris and Christos-Savvas Bouganis. 2017. Latency-Driven Design for FPGA-based Convolutional Neural Networks. In FPL.
- Ying Wang, Jie Xu, Yinhe Han, Huawei Li, and Xiaowei Li. 2016. DeepBurning: Automatic Generation of FPGA-based Learning Accelerators for the Neural Network Family. In DAC. Article 110. DOI:http://dx.doi.org/10.1145/2897937.2898003
- KaiyuanGuo,LingzhiSui,JiantaoQiu,SongYao,SongHan,YuWang,andHuazhongYang.2016.Angel-Eye:ACompleteDesign Flow for Mapping CNN onto Customized Hardware. In ISVLSI. DOI:http://dx.doi.org/10.1109/isvlsi.2016.129
- Jiantao Qiu and others. 2016. Going Deeper with Embedded FPGA Platform for Convolutional Neural Network. In FPGA. ACM, 26–35. DOI:http://dx.doi.org/10.1145/2847263.2847265
- Yufei Ma, Naveen Suda, Yu Cao, Jae sun Seo, and Sarma Vrudhula. 2016. Scalable and Modularized RTL Compilation of Convolutional Neural Networks onto FPGA. In FPL .DOI:http://dx.doi.org/10.1109/FPL.2016.7577356
- Hardik Sharma, Jongse Park, Emmanuel Amaro, Bradley Thwaites, Praneetha Kotha, Anmol Gupta, Joon Kyung Kim,Asit Mishra, and Hadi Esmaeilzadeh. 2016. Dnnweaver: from High-level Deep Network Models to Fpga Acceleration.

References

- Hardik Sharma, Jongse Park, Divya Mahajan, Emmanuel Amaro, Joon Kyung Kim, Chenkai Shao, Asit Mishra, andHadi Esmaeilzadeh. 2016. From High-Level Deep Neural Models to FPGAs. In MICRO. DOI:http://dx.doi.org/10.1109/micro.2016.7783720

- Chen Zhang, Zhenman Fang, Peipei Zhou, Peichen Pan, and Jason Cong. 2016. Caffeine: Towards Uniformed Representation and Acceleration for Deep Convolutional Neural Networks. In ICCAD. Article 12, 8 pages. DOI:http://dx.doi.org/10.1145/2966986.2967011

- Yaman Umuroglu, Nicholas J. Fraser, Giulio Gambardella, Michaela Blott, Philip Leong, Magnus Jahre, and KeesVissers. 2017. FINN: A Framework for Fast, Scalable Binarized Neural Network Inference. In FPGA. DOI:http://dx.doi.org/10.1145/3020078.3021744

- NicholasJ.Fraser,YamanUmuroglu,GiulioGambardella,MichaelaBlott,PhilipLeong,MagnusJahre,andKeesVissers.2017. Scaling Binarized Neural Networks on Reconfigurable Logic. In PARMA-DITAM. DOI:http://dx.doi.org/10.1145/3029580.3029586

- Yijin Guan, Hao Liang, Ningyi Xu, Wenqiang Wang, Shaoshuai Shi, Xi Chen, Guangyu Sun, Wei Zhang, and JasonCong. 2017. FP-DNN: An Automated Framework for Mapping Deep Neural Networks onto FPGAs with RTL-HLS Hybrid Templates. In FCCM. DOI:http://dx.doi.org/10.1109/fccm.2016.22

- Xuechao Wei, Cody Hao Yu, Peng Zhang, Youxiang Chen, Yuxin Wang, Han Hu, Yun Liang, and Jason Cong. 2017. Automated Systolic Array Architecture Synthesis for High Throughput CNN Inference on FPGAs. In DAC. DOI:http://dx.doi.org/10.1145/3061639.3062207