LogicNets: Co-Designed Neural Networks and Circuits for Extreme-Throughput Applications

Nicholas J. Fraser

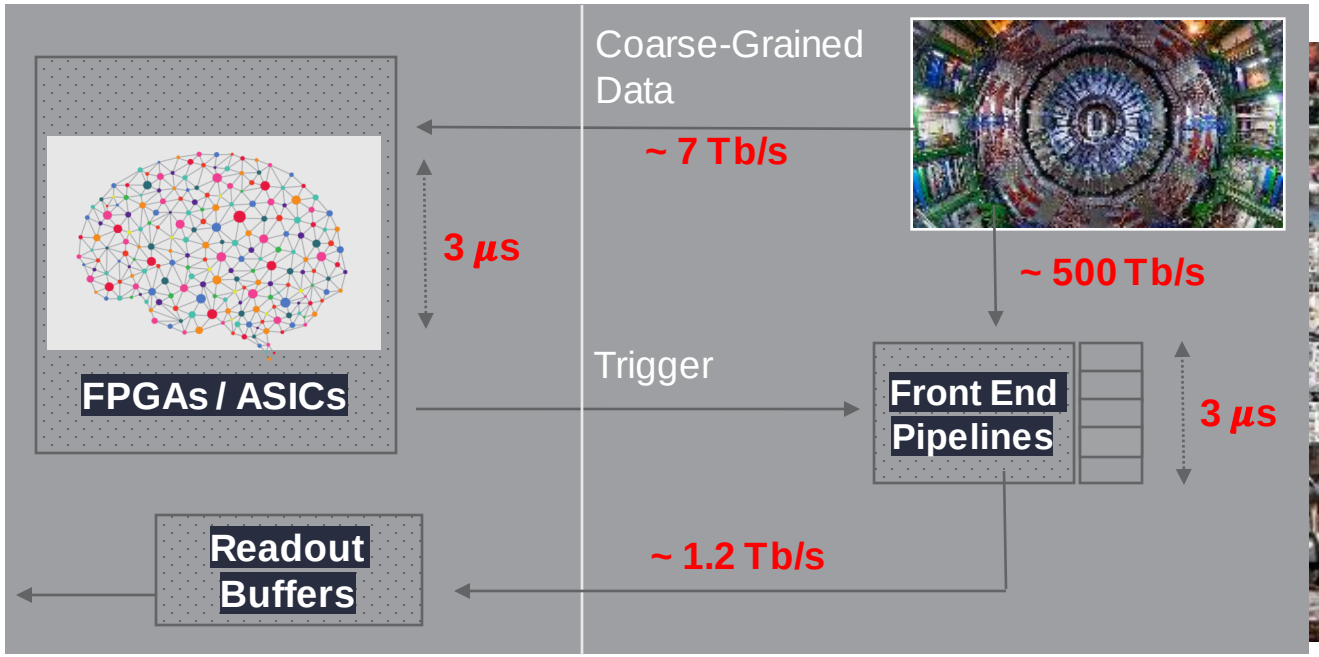
RCML Workshop @ FPL'22, 2022-09-01

AMD
together we advance_

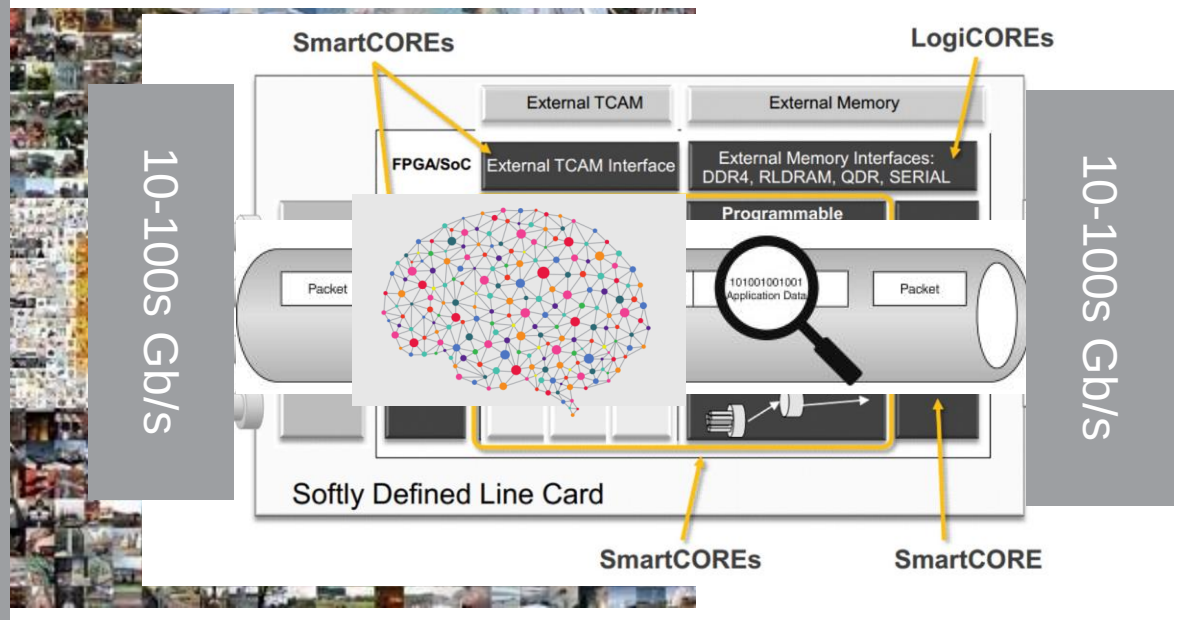
Agenda

1. ML in Extreme-Throughput Applications
2. LogicNets Methodology
3. Incompletely-Specified Boolean Functions
4. Conclusions

DNNs in Extreme-Throughput Applications



Source: Thomas James, CERN



CERN CMS Experiment

Network Intrusion Detection

- How do we mix DNNs into extreme-throughput applications?
  - Need DNNs running at 100Ms of FPS, sub-microsecond latency

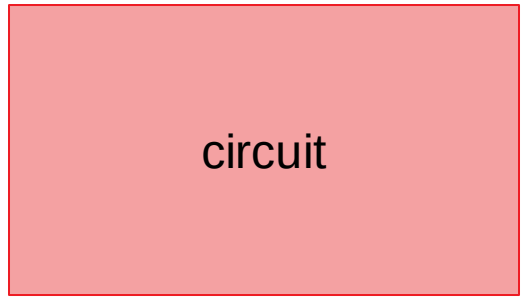
LogicNets at a Glance

Specialized DNN
Topology

(with particular constraints)

PyTorch

convert

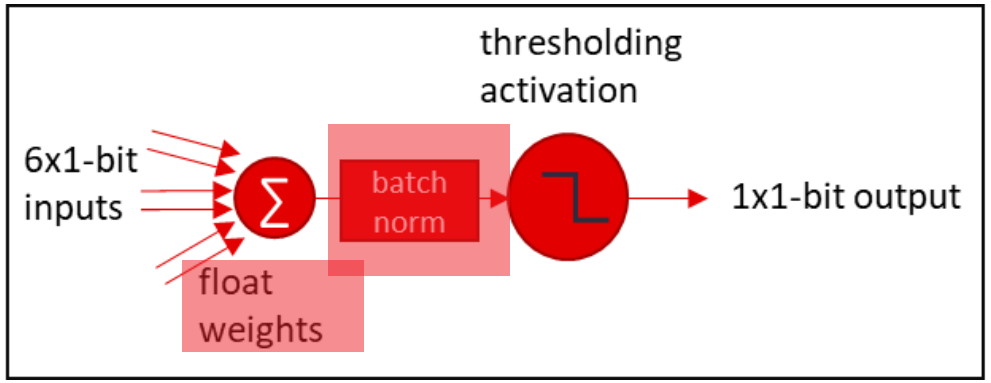


Fully-spatial
Implementation

II=1
low logic depth, high Fclk
100M's of samples per second

FPGA

Quantized Neurons as Truth Tables



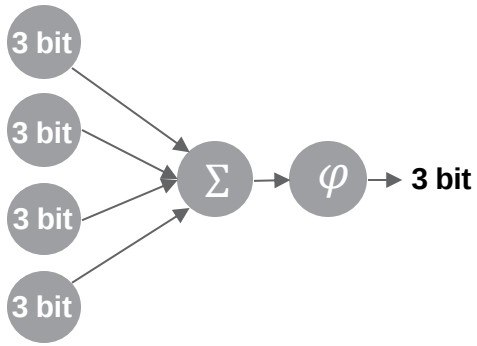
convert
(enumerate inputs)

Total input: 6 bits
Total output: 1 bit

Total input: 6 bits
Total output: 1 bit

PyTorch **FPGA**

Cost of Implementing Large Truth Tables



0	1	0	. . .	0	1	1	0
0	0	1		0	1	1	1
.							.
.							.
.							.
0	1	1		0	0	0	0
1	1	1	. . .	0	1	1	0

Total input: 12 bits
Total output: 3 bit

Truth Table Size: $4096 \times 15 = 2^{3*4} \times 3 * 5$

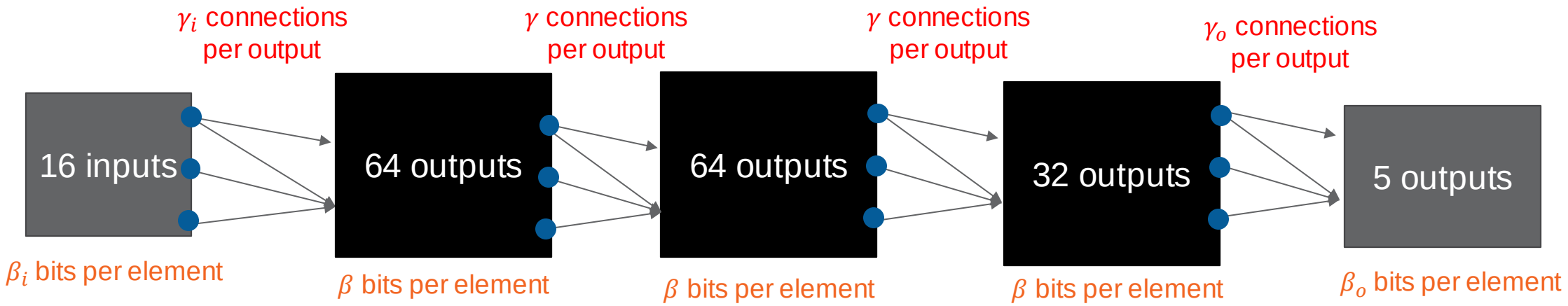
Co-design DNN topology to avoid intractably large LUTs.

LUT6 cost of the neuron:
~4095 LUT6s

Designing for Low Fan-In

- How do we ensure each neuron has low fan-in to avoid LUT explosion?

Sparsity: Each neuron connects to only γ previous neurons.



Activation Quantization: Each neuron only produces β bits.

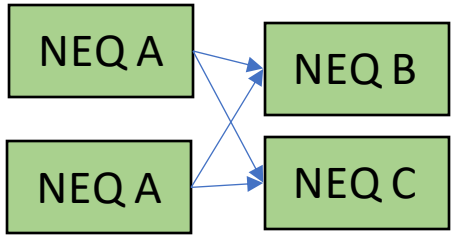
Constrain fan-in $\beta \cdot \gamma \leq 15$

LogicNets Design Flow

Decide Model → Train Model → Generate Truth Table → Generate Verilog Code → Synthesize with Vivado

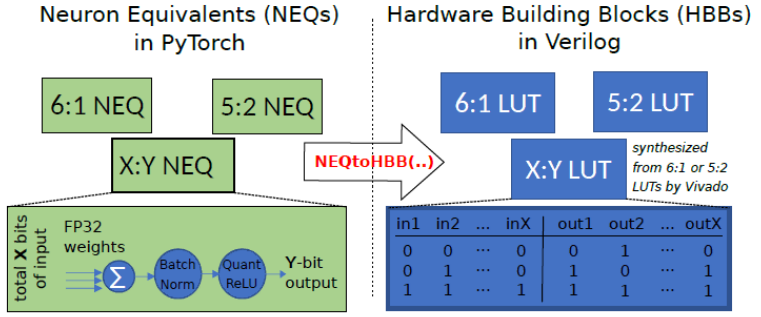
Model instantiated and trained using PyTorch

- Can engineer topology using LUT cost estimates

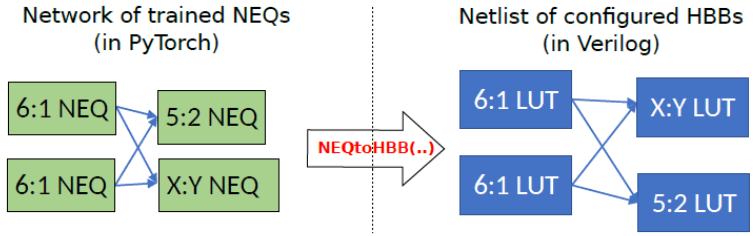


Define topology using NEQs in PyTorch and train

Model conversion and Synthesis using Python, then Vivado



(a) HBBs and NEQs



(b) Training and converting a network of NEQs to a netlist

Evaluation

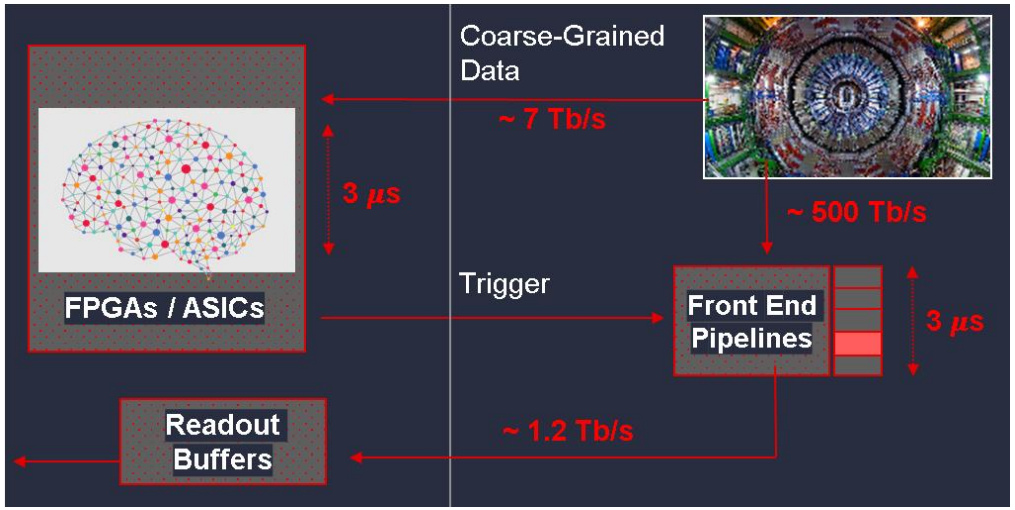
- On two tasks with extreme-throughput requirements:
  - **High-Energy Physics**: Jet Substructure Classification
  - **Network Security**: Intrusion Detection

- Design space exploration: selection of 2 to 4-layer networks
  - Topologies kept small to facilitate exploration + low resource cost
  - Used cost model to focus on relevant parts of design space

- All generated with LogicNets flow
  - Train in PyTorch, convert to Verilog
  - Out-of-context synthesis with Vivado 2019.2 targeting `xcu280-fsvh2892-2L-e`

- All results can be reproduced using our open source release:
  - https://github.com/Xilinx/logicnets

Evaluation - Jet Substructure Classification (JSC)



hls4ml JSC dataset [1]
16-input, 5-output classification problem

1 neuron = 1 6-LUT, since $\beta \cdot \gamma = 6$
very compact & fast, with non-trivial accuracy

Config	Acc %	LUT	Fmax	Latency
4-layer $\beta = 2, \gamma = 3$	69.4	241	1,400 MHz (666 MHz)	7.5 ns
4-layer $\beta = 3, \gamma = 4$	71.9	15.0k	584 MHz	8.6 ns
5-layer $\beta = 3, \gamma = 4$	73.0	36.3k	396 MHz	15 ns

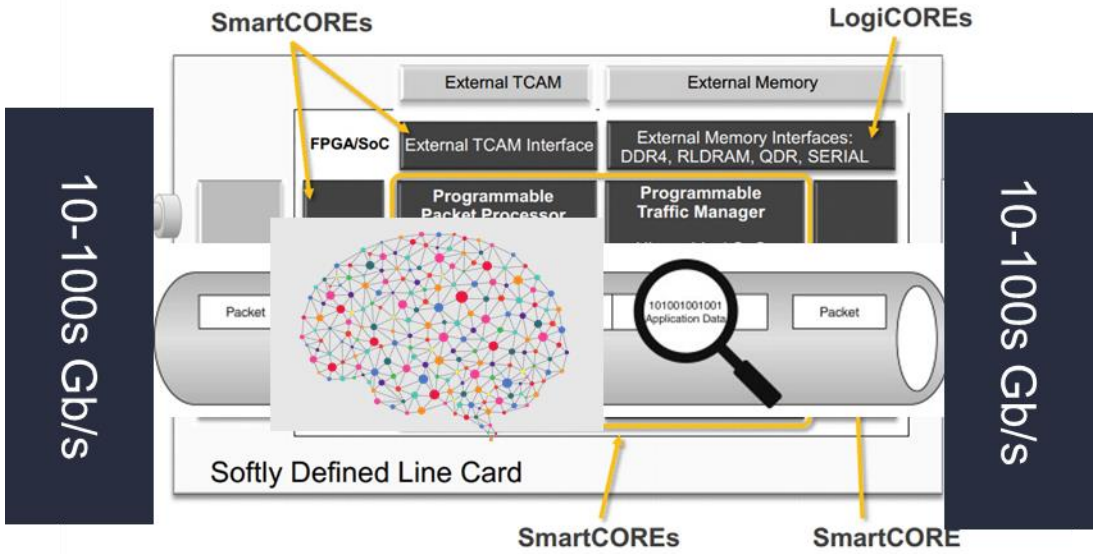
Prior FPGA work [1]:

50 ns latency @ 200 MHz
88k LUTs, 1k DSPs
75% accuracy with 16-bit

2.0x higher throughput, **3.3x** lower latency
No DSPs, less than half the LUTs
-2% points accuracy

[1] J. Duarte, et al, "Fast inference of deep neural networks in FPGAs for particle physics," Journal of Instrumentation, 2018.

Evaluation - Network Intrusion Detection (NID)



UNSW-NB15 Network Intrusion Detection dataset [1]

49-input, 1-output classification problem

High-throughput, small-footprint intrusion detection for e.g. SmartNICs

Config	Acc %	LUT	Fmax	Latency
2-layer $\beta = 2, \gamma = 7$	89.4	110	1183 MHz (666 MHz)	3 ns
4-layer $\beta = 2, \gamma = 7$	92.6	3.1k	801 MHz (666 MHz)	7.5 ns
4-layer $\beta = 3, \gamma = 5$	93.1	9.1k	498 MHz	10 ns

Prior FPGA work [2]:

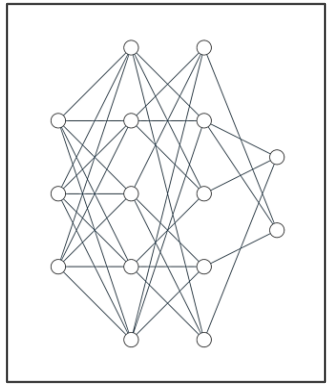
19.6 ns latency @ 51 MHz
51k LUTs
90% accuracy with 1-bit

13x higher throughput, **2.6x** lower latency
16.5x fewer LUTs
+2.6% points better accuracy

[1] N. Moustafa and J. Slay, "UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," MilCIS, 2015.
[2] T. Murovič and A. Trost, "Massively parallel combinational binary neural networks for edge processing," Elektrotehniski Vestnik, 2019.

LogicNets with Incompletely-Specified Boolean Functions

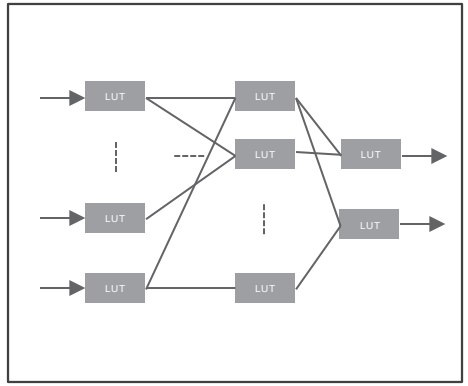
From Lossless to Lossy Transformations



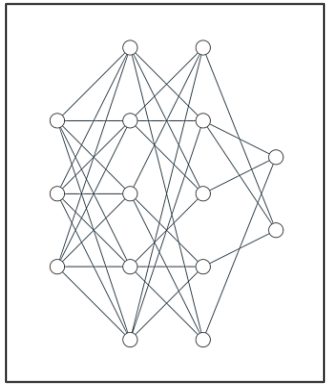
Network in PyTorch

Lossless transformation

Fully enumerated truth tables



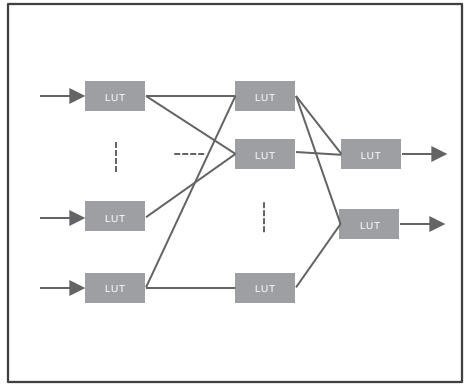
Technology-mapped network



Network in PyTorch

Lossy transformation

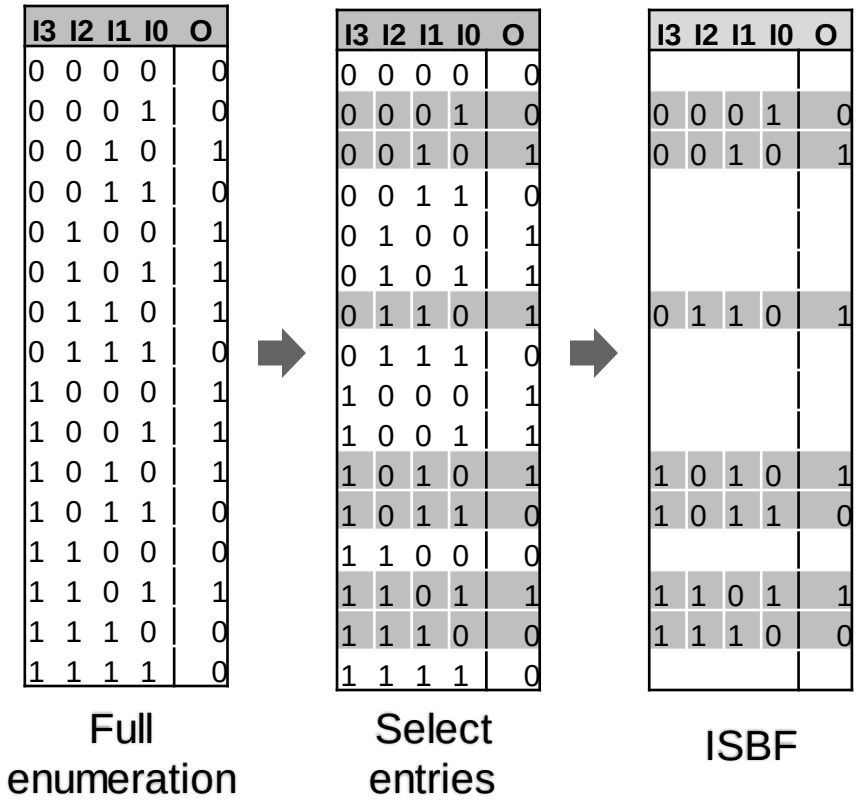
Incomplete truth tables



Technology-mapped network

Incompletely Specified Boolean Functions (ISBFs)

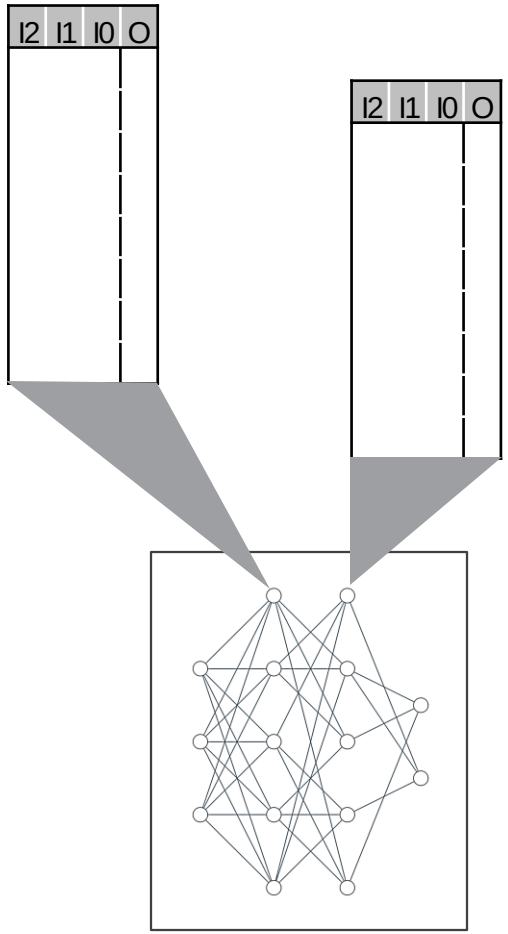
- Not every combination in the Truth-Table is important [1]
- Generate an ISBF based on a **care_set** including the **relevant** combinations
  - **Preserve functionality** of the neuron
  - **Decrease logic** cost of the neuron
- Explore logic minimisation through ISBFs



[1] Nazemi, Mahdi, Ghasem Pasandi, and Massoud Pedram. "NullaNet: Training deep neural networks for reduced-memory-access inference." arXiv preprint arXiv:1807.08716 (2018).

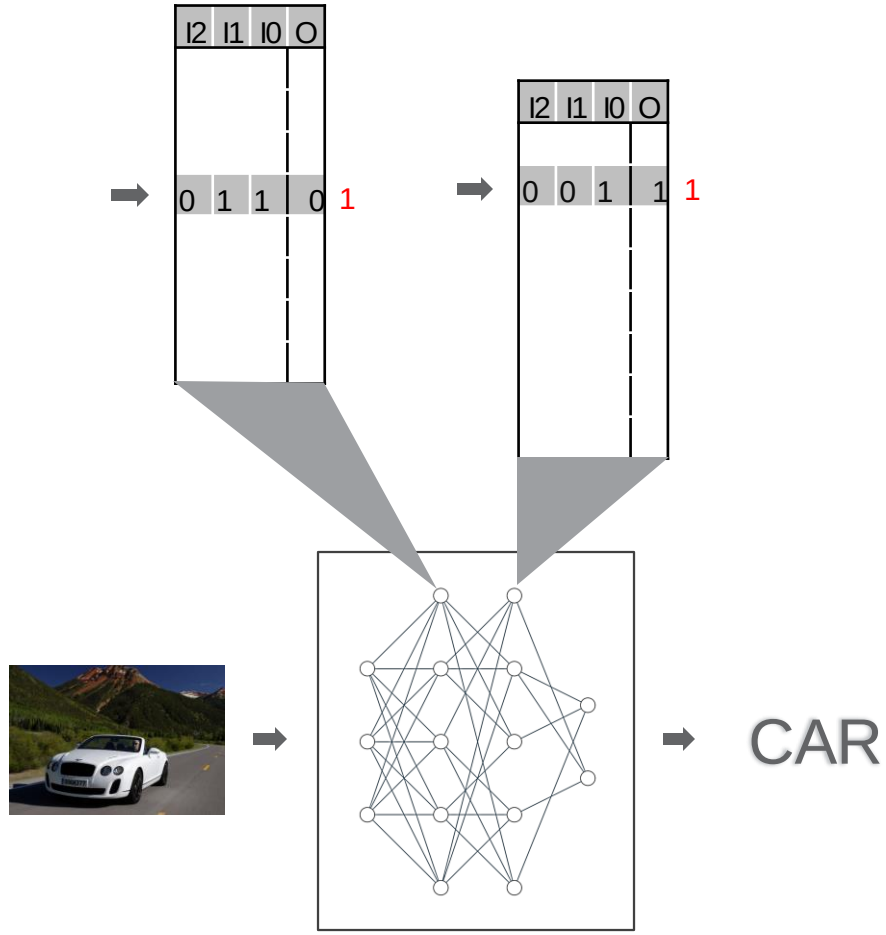
Generating the Care Set

- Monitor every I/O combination of every neuron over the dataset



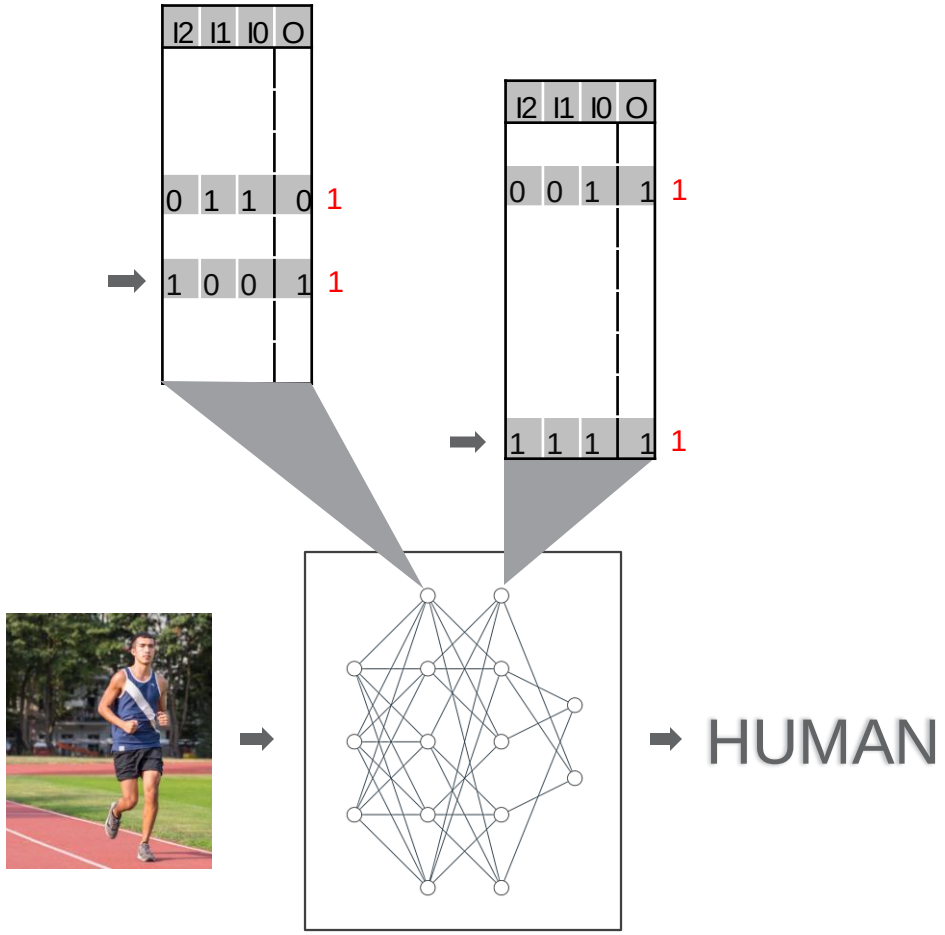
Generating the Care Set

- Monitor every I/O combination of every neuron over the dataset
- Calculate frequencies for input combinations



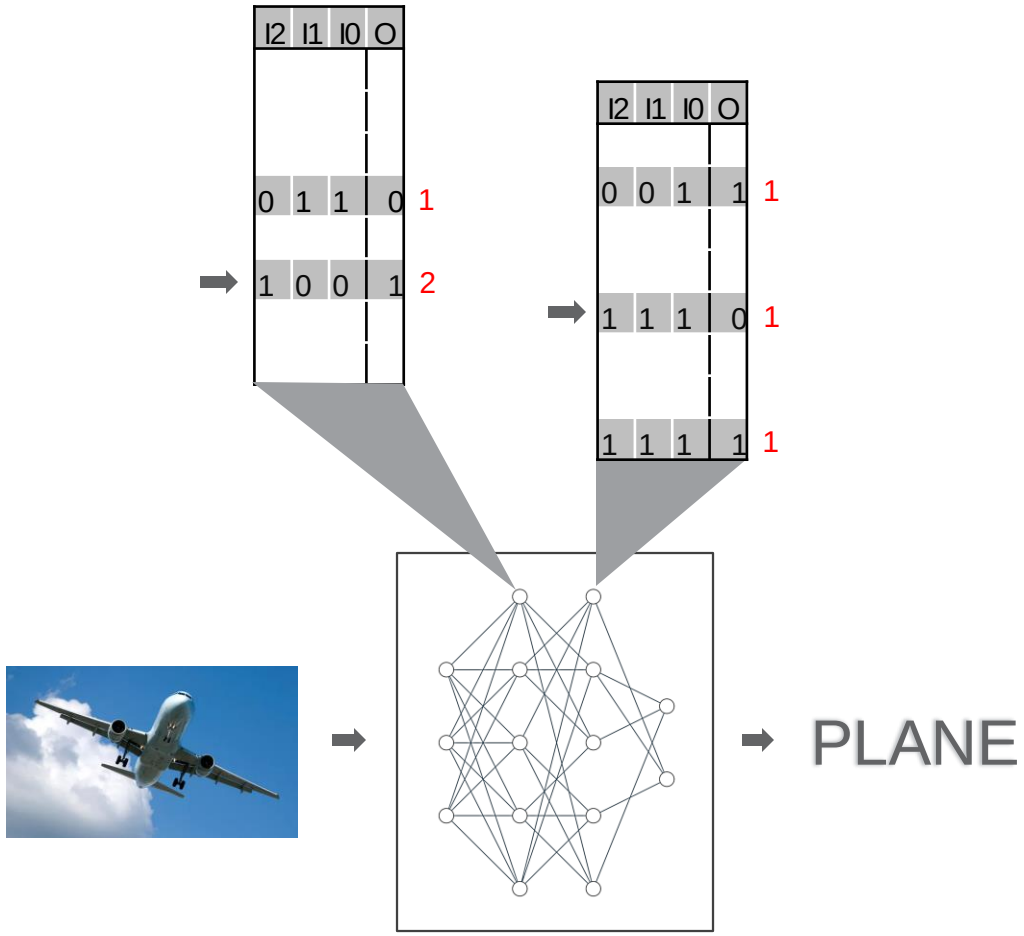
Generating the Care Set

- Monitor every I/O combination of every neuron over the dataset
- Calculate frequencies for input combinations



Generating the Care Set

- Monitor every I/O combination of every neuron over the dataset
- Calculate frequencies for input combinations
- Set cut-off frequency values to decrease the care_set size
- Size vs Accuracy trade-off



Evaluation

- On two tasks with extreme-throughput requirements:
  - **High-Energy Physics**: Jet Substructure Classification

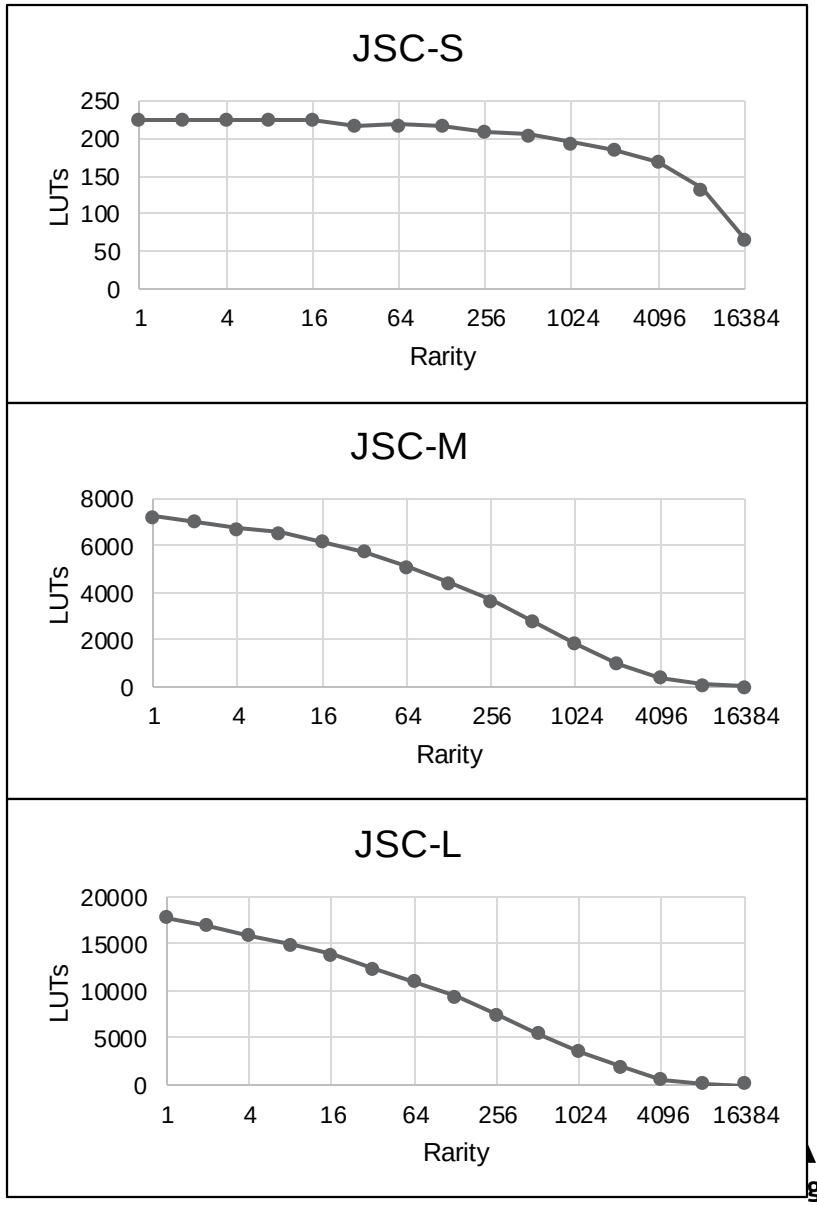
- Design space exploration: selection of 2 to 4-layer networks
  - Topologies kept small to facilitate exploration + low resource cost
  - Used cost model to focus on relevant parts of design space

- All generated with LogicNets flow
  - Train in PyTorch, convert to Verilog
  - Vivado 2021.2 used for baseline synthesis
  - ABC used for ISBF logic synthesis and technology mapping
    - Some additional customisations [1]

[1] Miyasaka, et al, "Synthesizing Practical Boolean Functions Using Truth Tables," IWLS, 2022.

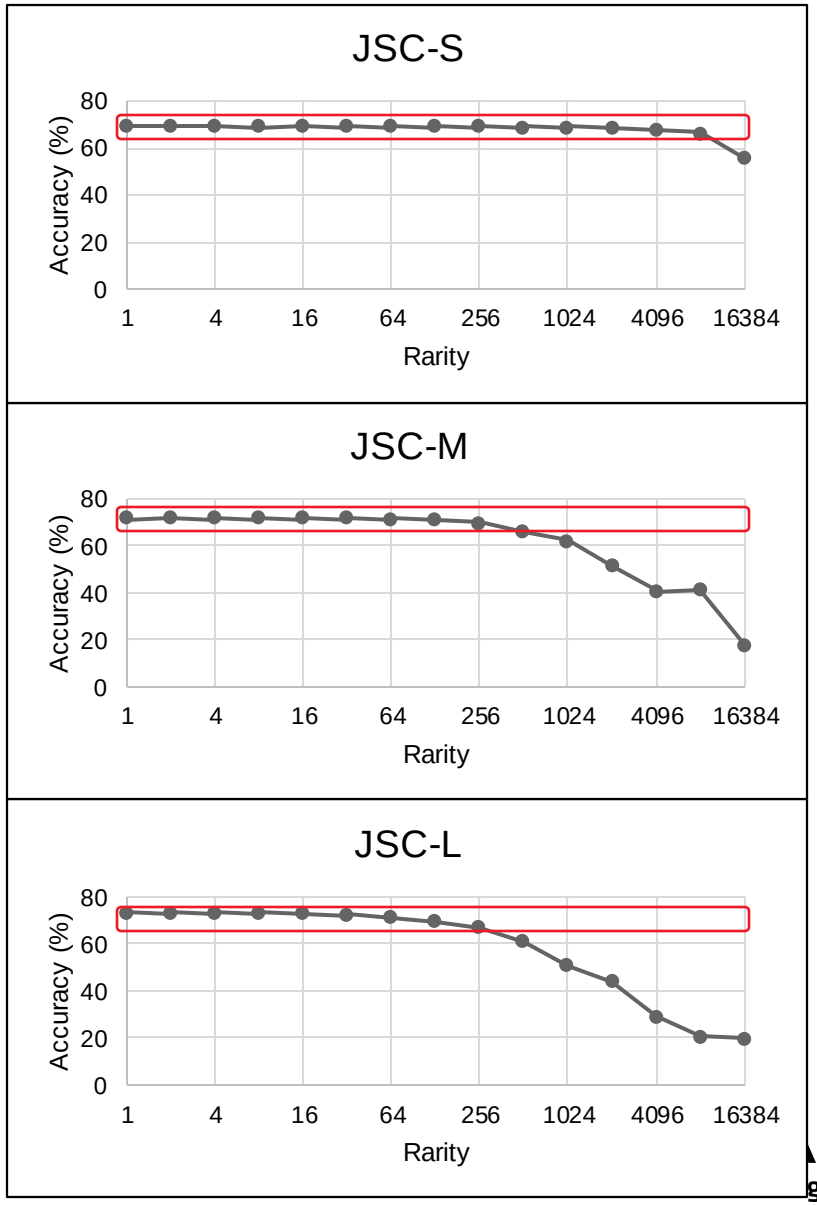
Results – Jet Substructure Classification

- **Resource Usage:**
  - Steady decrease in resource usage for larger networks (JSC-M/L)
  - JSC-S resource usage changes only slightly for $rarity \leq 512$



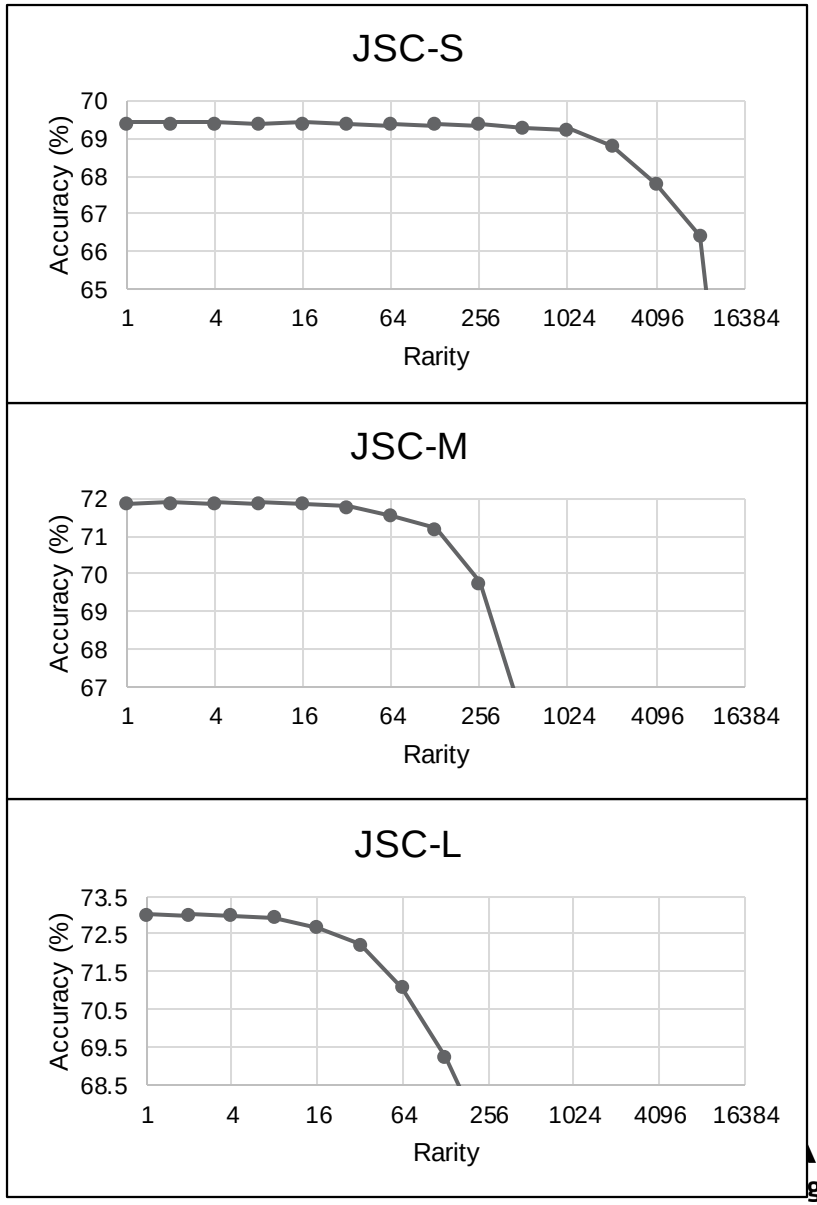
Results – Jet Substructure Classification

- **Resource Usage**:
  - Steady decrease in resource usage for larger networks (JSC-M/L)
  - JSC-S resource usage changes only slightly for $rarity \leq 512$
- **Accuracy:**
  - Network accuracy eventually “falls off the cliff” over some rarity threshold



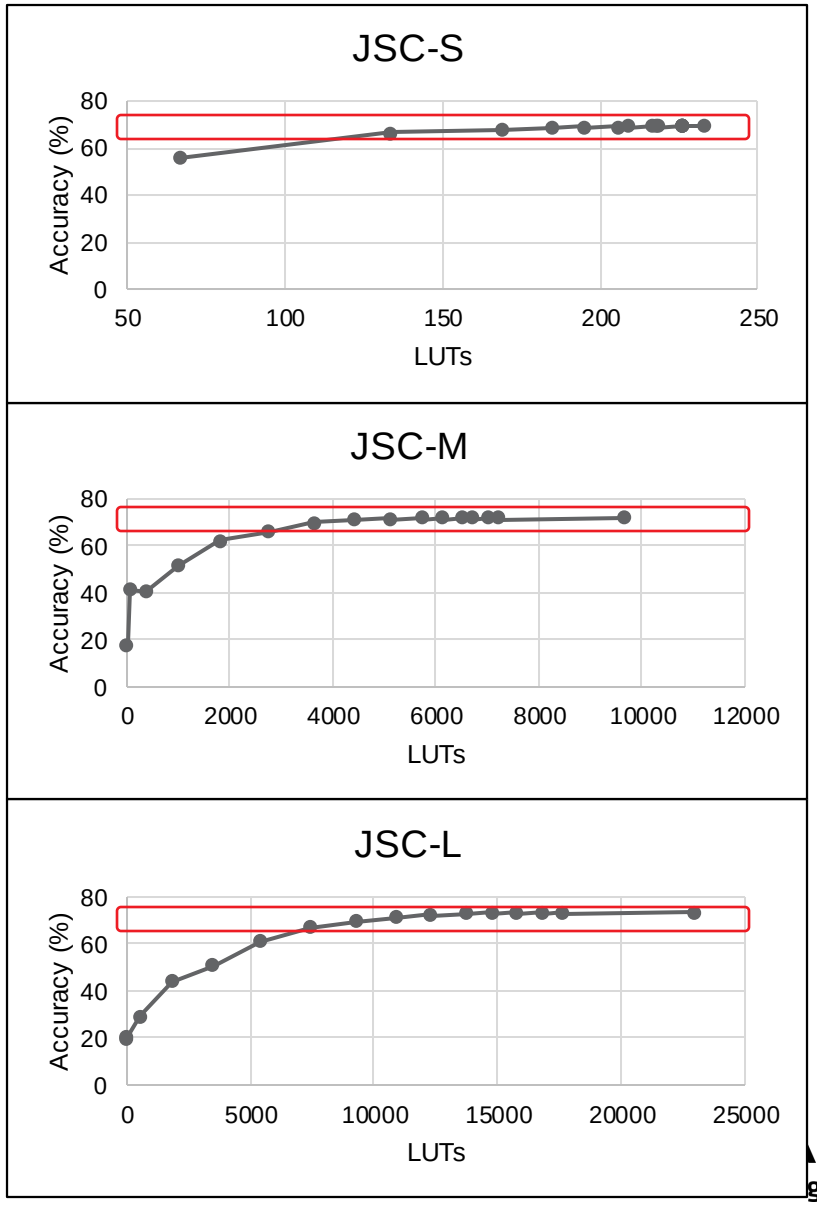
Results – Jet Substructure Classification

- **Resource Usage**:
  - Steady decrease in resource usage for larger networks (JSC-M/L)
  - JSC-S resource usage changes only slightly for $rarity \leq 512$
- **Accuracy:**
  - Network accuracy eventually “falls off the cliff” over some rarity threshold



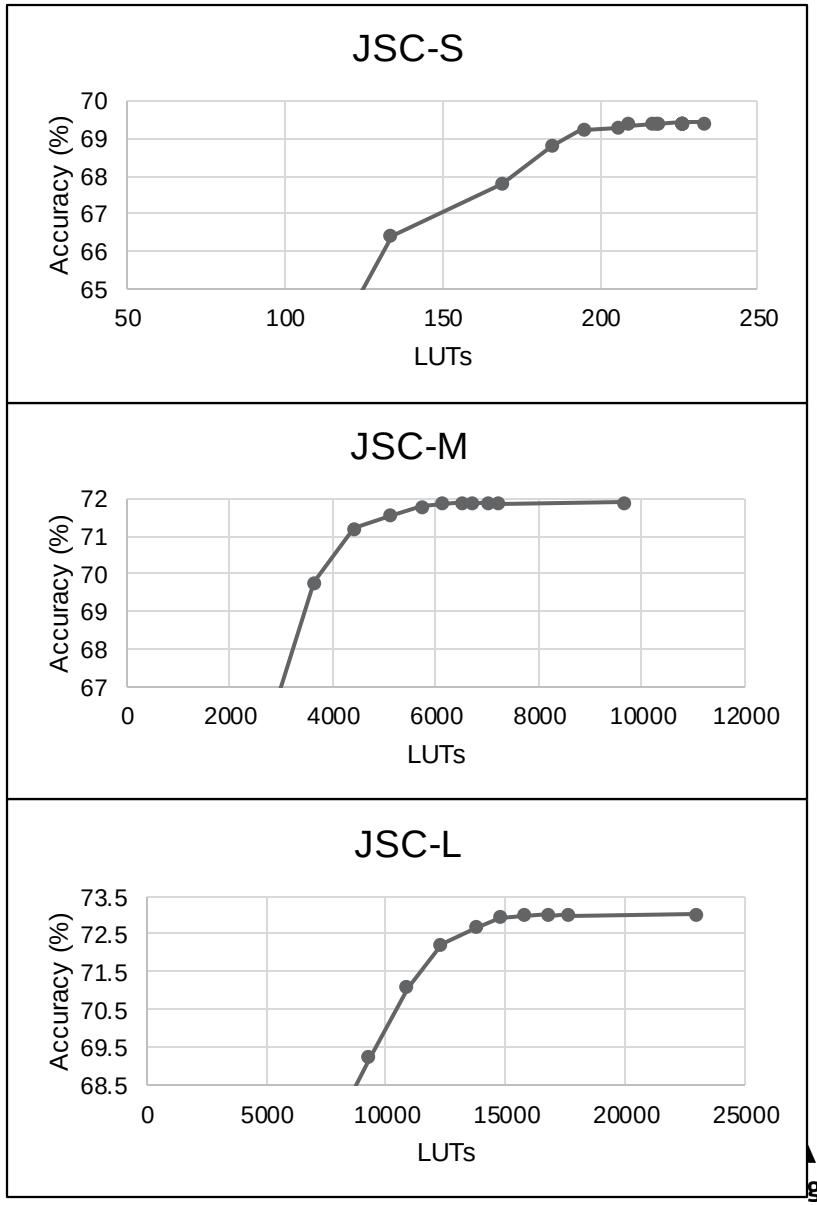
Results – Jet Substructure Classification

- **Resource Usage**:
  - Steady decrease in resource usage for larger networks (JSC-M/L)
  - JSC-S resource usage changes only slightly for $rarity \leq 512$
- **Accuracy:**
  - Network accuracy eventually "falls off the cliff" over some rarity threshold
  - Good accuracy/resource trade-offs can be achieved for small rarity values
    - After "falling off the cliff" it's better to implement a smaller topology
    - Biggest improvement for larger models



Results – Jet Substructure Classification

- **Resource Usage**:
  - Steady decrease in resource usage for larger networks (JSC-M/L)
  - JSC-S resource usage changes only slightly for $rarity \leq 512$
- **Accuracy:**
  - Network accuracy eventually "falls off the cliff" over some rarity threshold
  - Good accuracy/resource trade-offs can be achieved for small rarity values
    - After "falling off the cliff" it's better to implement a smaller topology
    - Biggest improvement for larger models



Results – Jet Substructure Classification

- **Resource Usage**:
  - Steady decrease in resource usage for larger networks (JSC-M/L)
  - JSC-S resource usage changes only slightly for $rarity \leq 512$
- **Accuracy:**
  - Network accuracy eventually "falls off the cliff" over some rarity threshold
  - Good accuracy/resource trade-offs can be achieved for small rarity values
    - After "falling off the cliff" it's better to implement a smaller topology
    - Biggest improvement for larger models

Network	Rarity Thr.	Acc. (%)	LUTs
JSC-S	0	69.4	227
JSC-S	1024	69.3	195
JSC-M	0	71.9	14.9k
JSC-M	32	71.8	5.5k
JSC-L	0	73.0	35.4k
JSC-L	8	72.9	14.8k

~60% reduction in LUTs for ~0.1% less accuracy

Conclusion

- DNNs for extreme-throughput applications through co-design
  - Implement quantized neurons as truth tables
  - Sparsity + few-bit quantization to keep compact size
  - Demonstrated 100M's of inferences/second on two simple datasets

- LogicNets makes use of the native resources of the FPGA
  - FPGA hardware building blocks (LUTs, routing resources) used directly

- Further gains possible if incompletely-specified Boolean functions
  - Up to 60% fewer resources for larger LogicNets models

- Available on Github: https://github.com/xilinx/logicnets

Future Work

- Incorporate more advances sparsification schemes
- Improve LogicNets topology design
- Utilise different quantisation schemes and/or different activation function
- Explore more synthesis techniques optimised to LogicNet networks

Acknowledgements

- Colleagues in my team (Michaela Blott's team)
  - In particular, Yaman, and former interns: Jon-Ander, Yash Akhauri
  - Others in AMD AECG Research Labs

- Imperial College London
  - George Constantinides

- UC Berkeley
  - Alan Mishchenko, Yukio Miyasaka, John Wawrzynek


AMD