MLBlocks: FPGA Blocks for Machine Learning Applications

SeyedRamin Rasoulinezhad, David Boland, and Philip H.W. Leong
School of Electrical and Information Engineering, The University of Sydney

THE UNIVERSITY OF SYDNEY

Motivation

› Observation:

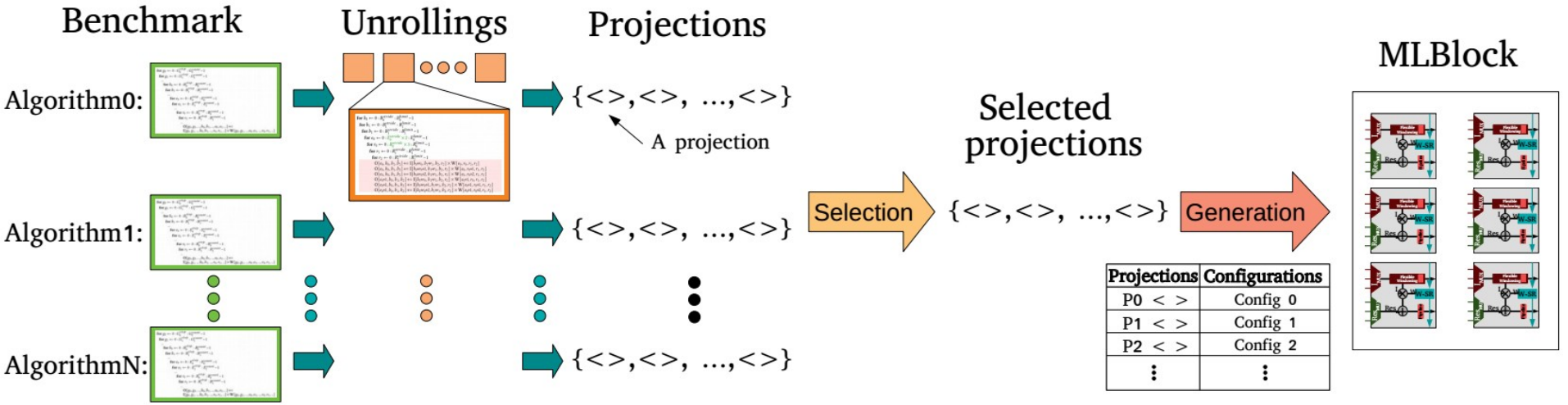
- FPGA architectures are Optimized for networking, signal/image processing using high-precision DSP blocks
- DSP48 → 27×18 multiplier, 48-bit Accumulation or two one input shared 8x8 MACs

› Previous works:

- **Enhancing existing DSP blocks**: PIR-DSP [1], Boutrous et al [2], Intel Agilex Architecture
  - Works on logic elements: LUXOR [3], Boutros et al. [4]
- **Integrating domain-specific engines:** Xilinx Versal AI-engines
  - Fundamental issue: They do not address the shortcomings of current FPGA architectures
- **Designing a new embedded block:** Hamamu [5], Achronix MLP72

› Aim: How to design a new block by a systematic fashion targeting future workloads

Overview and Generalized nested loop model



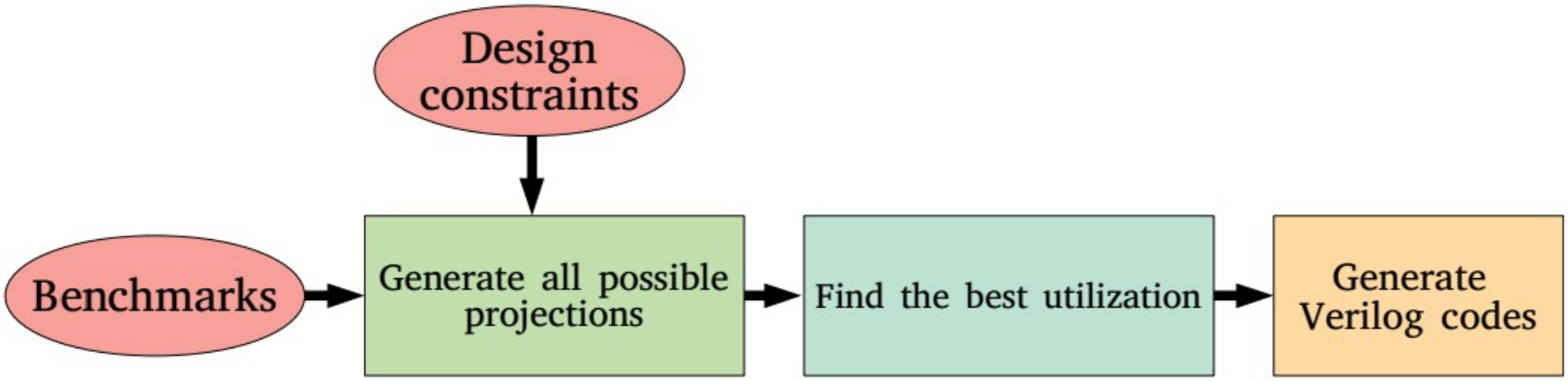
› Four loop variable groups:

- **R**eduction(IW) → reuse of temporary O
  - Like: dot product
- **E**xpansion(WO) → reuse of I
- **B**atching(IO) → reuse of W
- **G**rouping(IWO) → computation replication
  - Used in depth-wise and grouped convolution

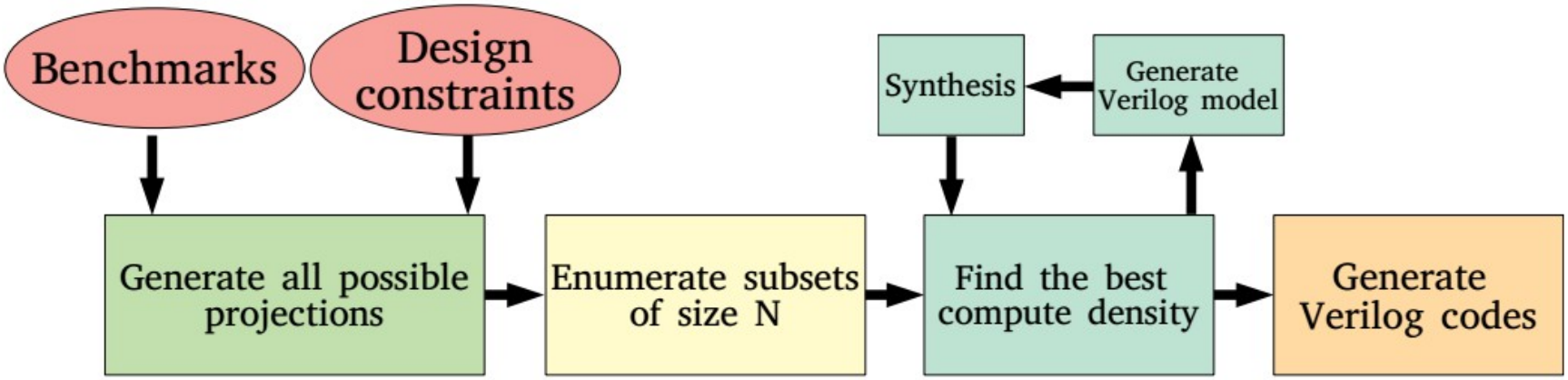
Algorithm 1: Pseudo code of the generalized nested loop model

$$
\begin{aligned}
&\textbf{for } g_0 \leftarrow 0 : G_0^{stride} : G_0^{limit} - 1 \\
&\quad \textbf{for } g_1 \leftarrow 0 : G_1^{stride} : G_1^{limit} - 1 \\
&\quad\quad \ldots \\
&\quad\quad\quad \textbf{for } b_0 \leftarrow 0 : B_0^{stride} : B_0^{limit} - 1 \\
&\quad\quad\quad\quad \textbf{for } b_1 \leftarrow 0 : B_1^{stride} : B_1^{limit} - 1 \\
&\quad\quad\quad\quad\quad \ldots \\
&\quad\quad\quad\quad\quad\quad \textbf{for } e_0 \leftarrow 0 : E_0^{stride} : E_0^{limit} - 1 \\
&\quad\quad\quad\quad\quad\quad\quad \textbf{for } e_1 \leftarrow 0 : E_1^{stride} : E_1^{limit} - 1 \\
&\quad\quad\quad\quad\quad\quad\quad\quad \ldots \\
&\quad\quad\quad\quad\quad\quad\quad\quad\quad \textbf{for } r_0 \leftarrow 0 : R_0^{stride} : R_0^{limit} - 1 \\
&\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad \textbf{for } r_1 \leftarrow 0 : R_1^{stride} : R_1^{limit} - 1 \\
&\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad \ldots \\
&\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad \mathbf{O}[g_0, g_1, .., b_0, b_1, .., e_0, e_1, ..] \mathrel{+}= \\
&\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad \mathbf{I}[g_0, g_1, .., b_0, b_1, .., r_0, r_1, ..] \times \mathbf{W}[g_0, g_1, .., e_0, e_1, .., r_0, r_1, ..]
\end{aligned}
$$

Projection selections

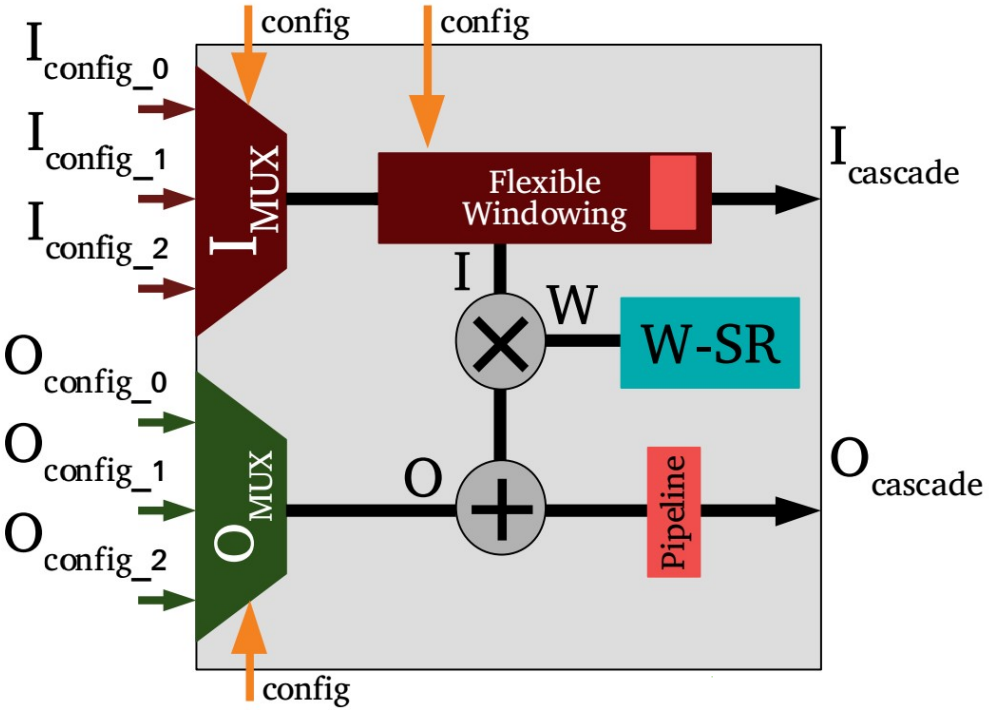


(a) Greedy search

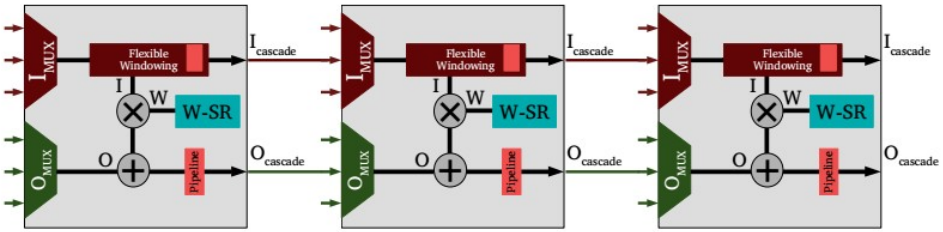


(b) N-Config search

Micro Architecture



(a) A MAC unit



(c) Cascading MAC units using I_{out} and Res_{out} signals

+ Serial Multiplier

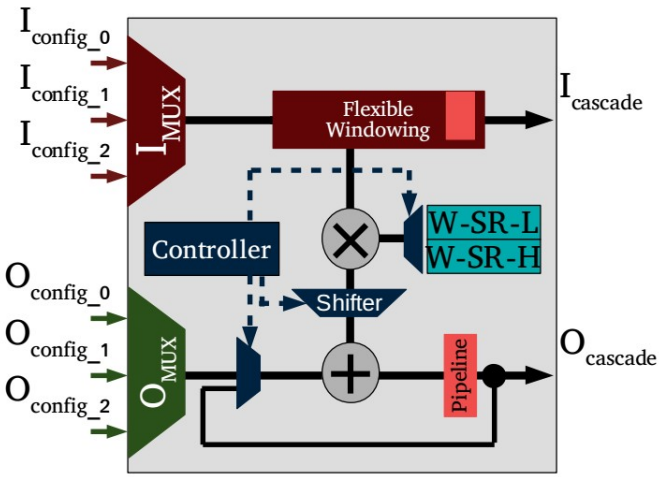
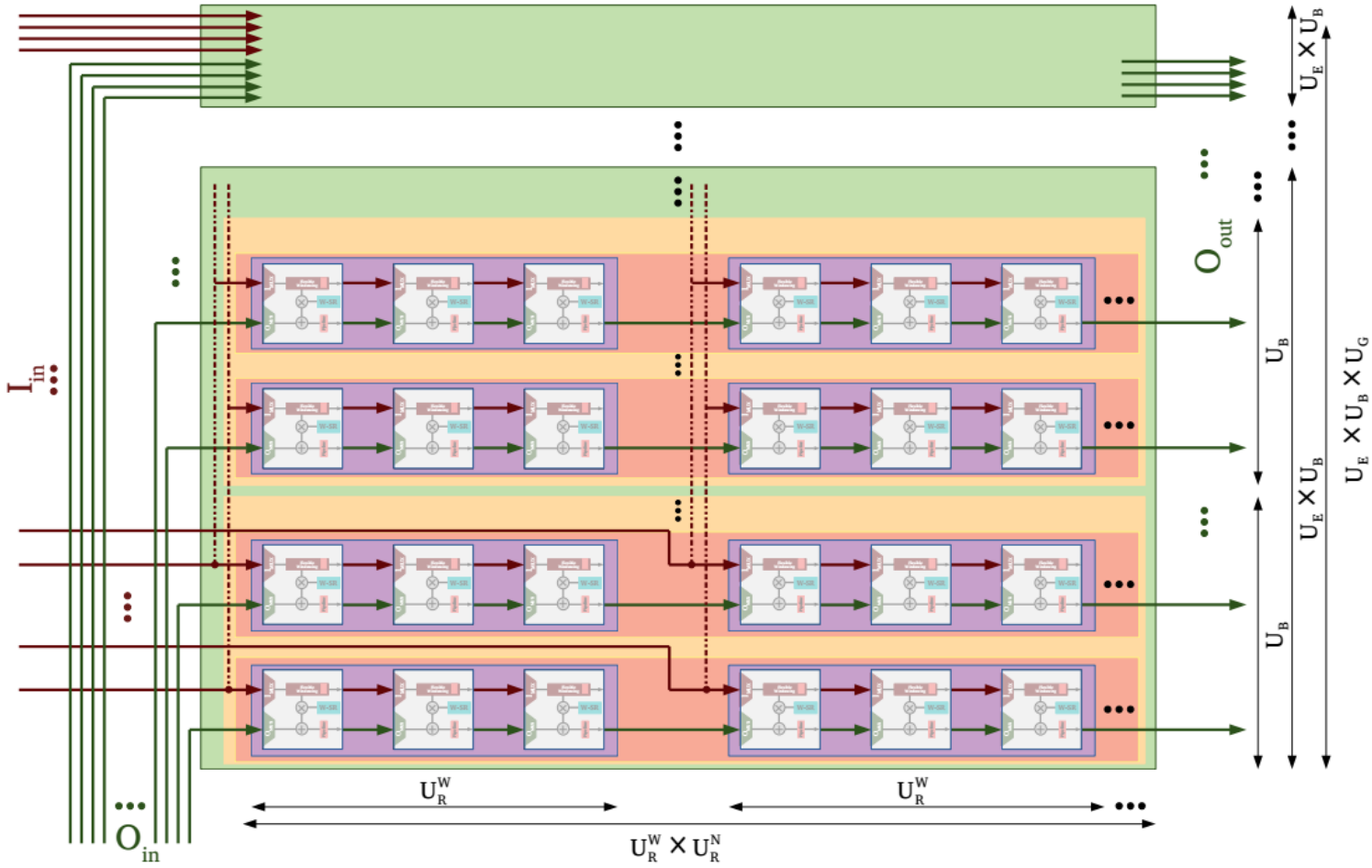


Figure 6: A serial multiplier-armed MAC unit

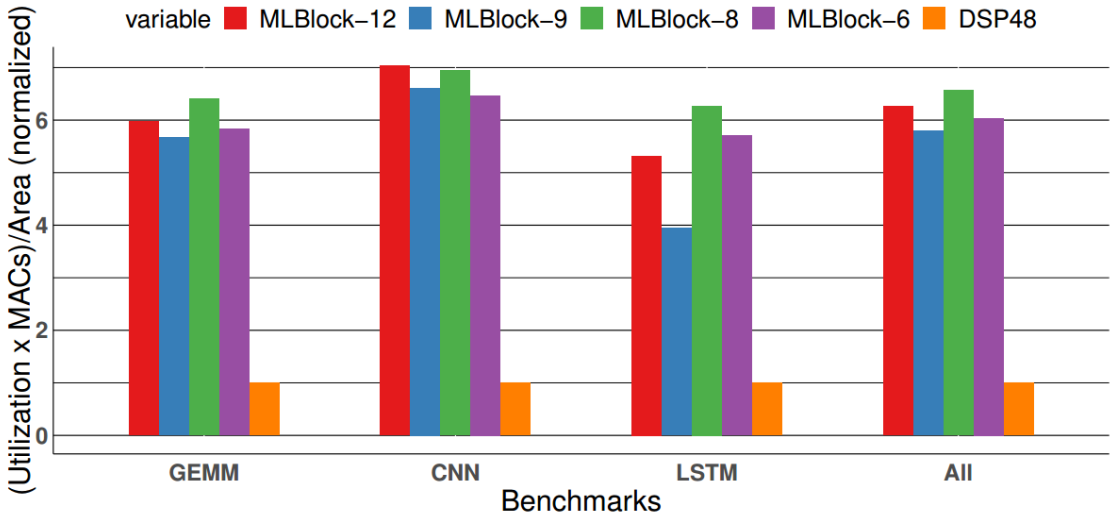
Macro Architecture

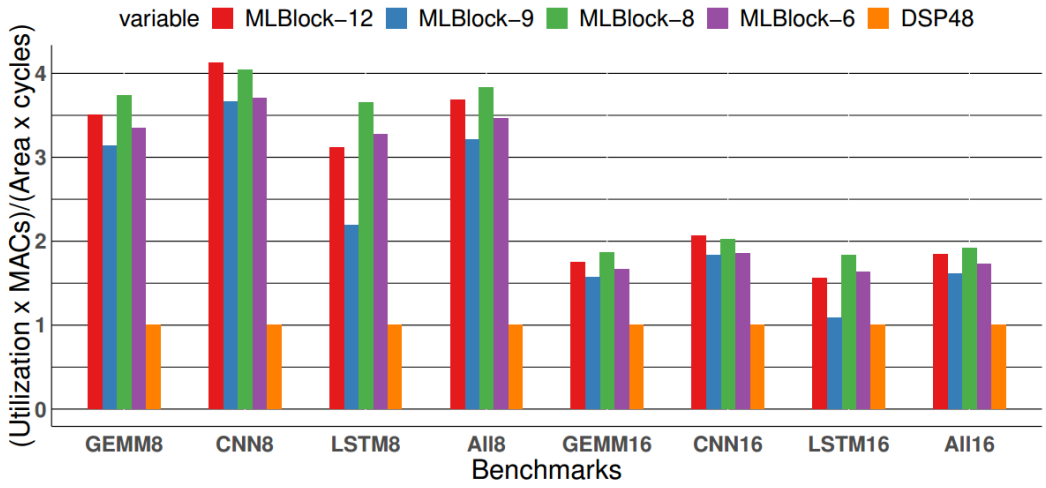
- Each selected projection maps to an MLBlock configuration
- A projection routing:



Results – DSP48-like MLBlocks

- Assumptions:
  - Greedy projection selection
  - DSP48 IO and Area constraints
- Compute density improvement:
  - Only 8x8 (top)
  - 8x8, 8x16, 16x8, 16x16 (down)





THE UNIVERSITY OF SYDNEY

Conclusion

- Briefly our contributions are:
  - A Methodology for designing coarse-grained embedded blocks for machine learning applications.
  - MLBlocks, a parameterized embedded block architecture
  - Two configuration selection techniques, called Greedy and heuristic
  - A python based framework to generate hardware description of an efficient MLBlock instance for a given set of constraints

- Conclusions:
  - Using Xilinx DSP48 constraints, our approach results to embedded blocks with 6 times more compute density

References

- [1] S. Rasoulinezhad, H. Zhou, L. Wang, and P. H. W. Leong, PIR-DSP: an FPGA DSP block architecture for multi-precision deep neural networks, FCCM 2019,
- [2] A. Boutros, S. Yazdanshenas, and V. Betz, Embracing diversity: Enhanced DSP blocks for low-precision deep learning on FPGAs, FPL 2018
- [3] S. Rasoulinezhad, Siddhartha, H. Zhou, L. Wang, D. Boland, and P. H. W. Leong, LUXOR: an FPGA logic cell architecture for efficient compressor tree implementations, in FPGA '20
- [4] A. Boutros, M. Eldafrawy, S. Yazdanshenas, and V. Betz, Math doesn't have to be hard: Logic block architectures to enhance low-precision multiply-accumulate on FPGAs, FPGA 2019
- [5]A. Aror, Z. Wei2, and L. K. John, Hamamu: Specializing FPGAs for ML Applications by Adding Hard Matrix Multiplier Blocks, ASAP20