Low Latency YOLO v3-Tiny Accelerator for Low-Cost FPGA

Enhanced Architecture for Edge Computing

Table of Contents

01 Low Latency YOLO v3-Tiny Accelerator for Low-Cost FPGA

02 Background and Motivation

03 Proposed Accelerator Architecture

04 General Matrix Multiplication (GEMM)

05 Systolic Array Architecture

06 Custom Instruction Set

07 Results and Performance

08 Conclusion and Future Applications

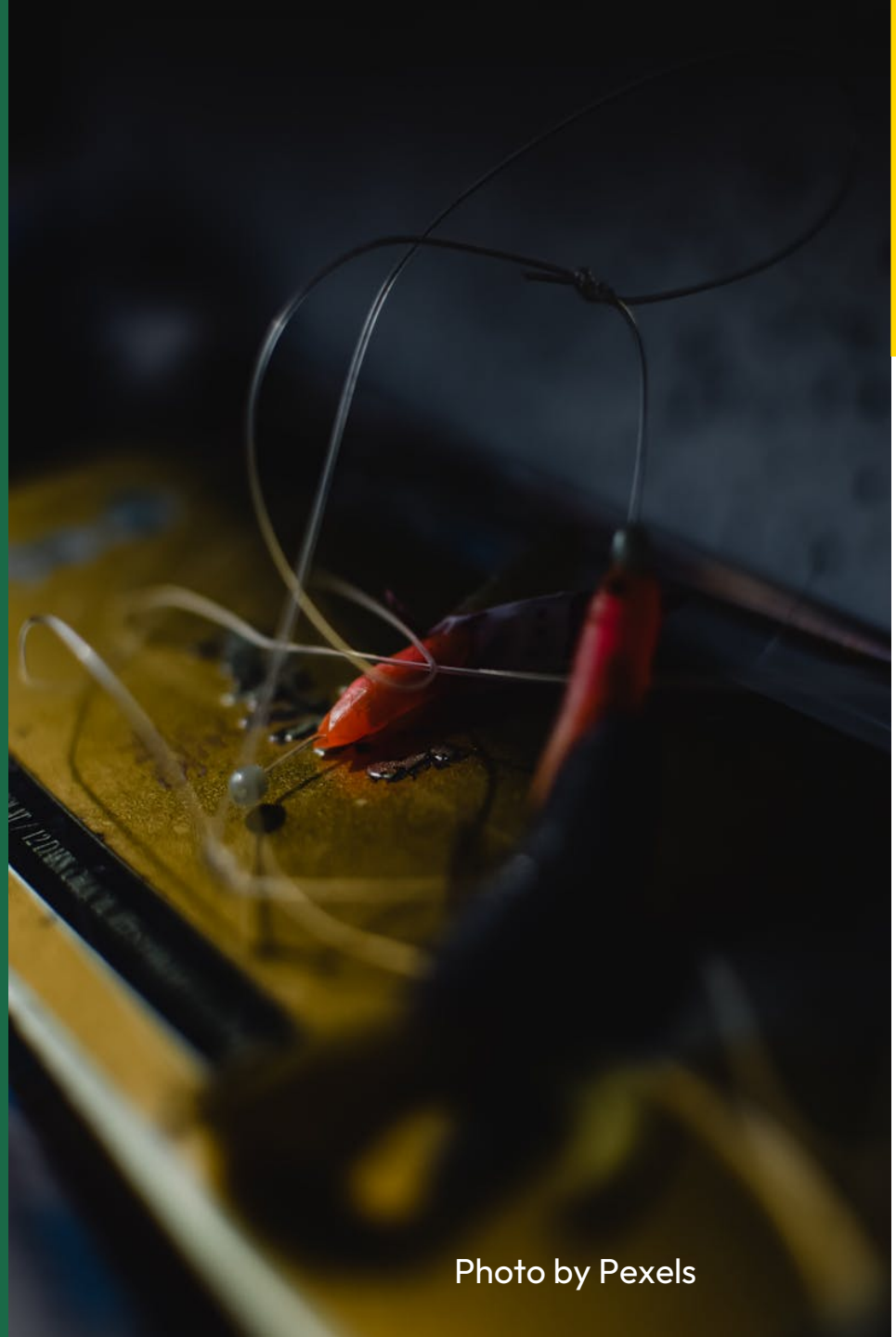


Photo by Pexels

1

Low Latency YOLO v3-Tiny Accelerator for Low-Cost FPGA

Leveraging Dataflow, Control Flow, and GEMM in FPGA

- Authors: TRIO ADIONO, ADIWENA PUTRA, NANA SUTISNA, INFALL SYAFALNI, and RAHMAT MULYAWAN
- Institut Teknologi Bandung

- Publication: IEEE Access

Photo by Pexels

2

Background and Motivation

Challenges in Edge AI

- Edge AI needs: Low latency, low energy, high computation efficiency.
- Traditional solutions struggle with latency and power in IoT applications.
- Objective: Efficiently implement YOLO v3-Tiny on low-cost FPGA using innovative data handling and processing.


Photo by Pexels

3

Proposed Accelerator Architecture

Dataflow, Control Flow, and Custom Instructions

- Dataflow architecture ensures efficient data movement across processing elements.
- Control flow manages execution order and allows for FPGA scalability.
- Custom instruction set optimizes key layers for low-cost hardware.

Photo by Pexels

4

General Matrix Multiplication (GEMM)

Matrix Multiplication Simplifies Convolution

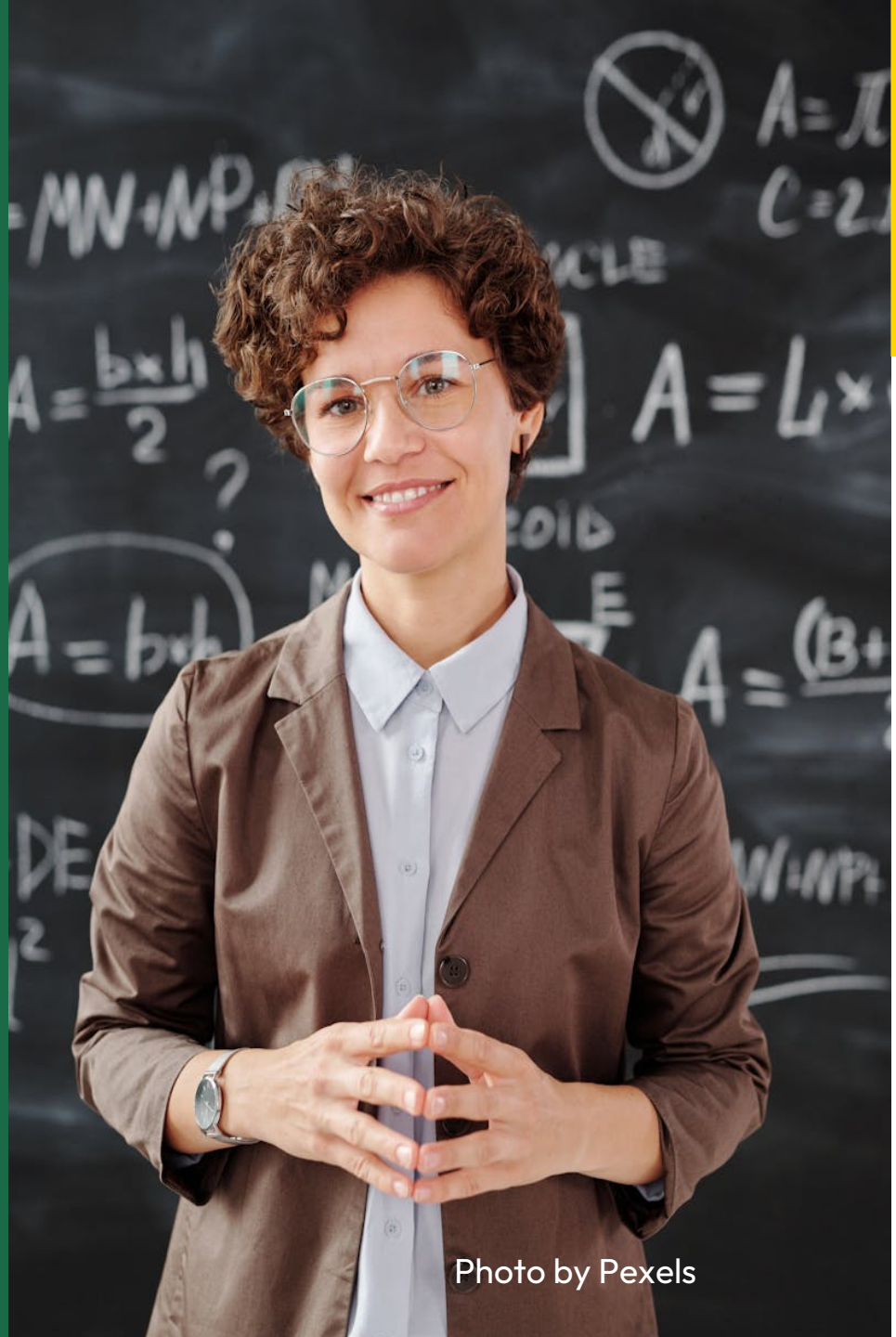
- GEMM allows for convolution by reshaping feature maps and weights into matrices.
- Efficient processing: reduces custom hardware needs for variable kernel sizes.
- Example: 3x3 kernel reshaped for matrix multiplication.

to by Pexels

Systolic Array Architecture

Optimizing Matrix Multiplication

- Array of processing elements enables pipeline-based parallel processing.
- Efficiently handles matrix multiplication in GEMM, enabling scalability.
- Data flows through the array with minimal control overhead, ideal for FPGA.



Custom Instruction Set

Specialized Instructions for YOLO Layers

- Optimizes convolution, batch normalization, and quantization layers.
- Reduces complexity and boosts processing efficiency by merging layers and optimizing precision.
- Example: merging batch normalization with convolution to minimize operations.

Results and Performance

Accelerator Efficiency

- Performance: 8.3 FPS, 31.5 GOPS, 69.3x faster than CPU.
- Up to 1.75x better clock cycle ratio than other commercial accelerators.
- Use Cases: Traffic analysis, ADAS, real-time applications.

Photo by Pexels

8

Conclusion and Future Applications

Impact of Accelerator Design

- Low-cost, scalable FPGA solution optimized for edge applications.
- Applications: surveillance, autonomous vehicles, smart cities.