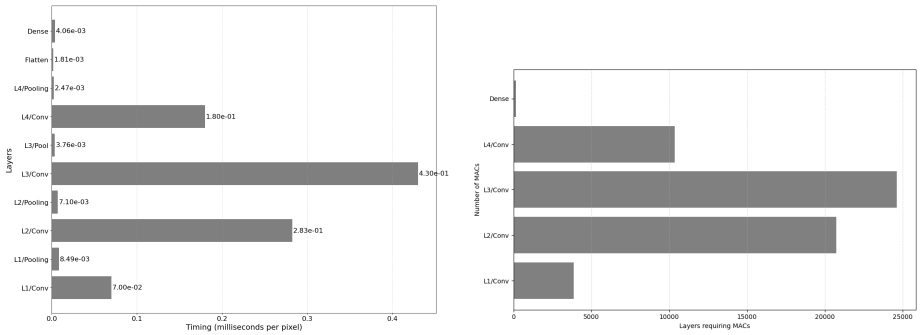
1 Acceleration Analysis on FPGAs to Reduce Inference Time

When we test *1D-Justo-LiuNet* on the Zynq-7030's CPU, segmenting an entire data cube takes around 10.78 minutes on average. This suggests that the network would benefit from future FPGA acceleration to reduce processing time and power consumption. For this, identifying the most time-consuming operation(s) on the CPU is crucial.

Consistent with the state of the art [6], results in Fig. 1a show that most of the runtime is spent on convolution operations, while pooling, flattening, and dense layers require significantly less time. Thus, convolution layers are the primary target for FPGA acceleration. When comparing the time required for each convolution layer, the deeper convolutions take longer, which is expected due to the increasing number of kernels and the use of multi-component kernels. However, in the final convolution, the processing time decreases since the pooled feature maps are smaller, saving time. The processing time of the convolutions correlates with the number of MAC operations, as shown in Fig. 1b. Therefore, the convolution layer at level 3 is the best candidate for FPGA acceleration. This can be achieved either through a fully FPGA-based design or a software-hardware co-design approach [6], where the CPU handles various operations while the FPGA handles the more time-consuming arithmetic operations.

As future work, we recommend synthesizing our C implementation into Register Transfer Level (RTL) design for the Kintex-7 FPGA within the Zynq-7030 to accelerate the convolutions. Our current C implementation is openly available in our GitHub repository at `https://github.com/NTNU-SmallSat-Lab/AI_deployment_justoliunet`. The software avoids dynamic memory allocation to make the implementation more easily synthesizable into FPGA logic. Given the promising development of High-Level-Synthesis (HLS) tools for deep learning [2, 1, 4], we propose using Xilinx's Vitis HLS. Although a comparison between Vitis AI and Vitis HLS is beyond the scope of this work, we acknowledge the availability of tools like Vitis AI for fast deployment on Xilinx devices [12], offering libraries and compatibility with frameworks like TensorFlow to streamline FPGA design and achieve quick, high-performance results [13]. However, compatibility challenges, such as implementing 1D convolutions in 1D-CNNs, must still be considered. Overall, Vitis HLS offers more hardware control. By applying *pragma* directives to perform parallel MAC operations [5, 8, 9, 7, 4, 3], Vitis HLS can significantly reduce the processing time. While MAC operations can be implemented using regular FPGA resources such as Look-Up Tables (LUTs), we highly recommend using the dedicated Digital Signal Processing (DSP) blocks, as the Kintex-7 FPGA incorporates 400 DSP blocks optimized for arithmetic acceleration [11]. These blocks can achieve up to 593 GMACs, allowing many billions of MAC operations per second. In FPGA design, analyzing data inter-dependencies is crucial [6]. Given the absence of dependencies during MAC operations, we recommend using parallel multipliers between windowed samples and kernel weights to achieve an optimal RTL design. These should be



(a) Processing times in miliseconds. (b) Number of MAC operations.

Figure 1: Comparative analysis of timing and MAC operations across *1D-Justo-LiuNet*.

followed by an efficient adder configuration such as Carry-lookahead adders [10] with adequate critical path, while maintaining clock frequency and ensuring that MAC computations are completed within a single clock cycle for proper synchronization. Further details on an FPGA implementation are beyond the scope of this work.

References

[1] Samuel Boyle, Aksel Gunderson, and Milica Orlandić. "High-level FPGA Design of Deep Learning Hyperspectral Anomaly Detection". In: *2023 IEEE Nordic Circuits and Systems Conference (NorCAS)*. IEEE. 2023, pp. 1–5.

[2] Jason Cong et al. "FPGA HLS today: successes, challenges, and opportunities". In: *ACM Transactions on Reconfigurable Technology and Systems (TRETS)* 15.4 (2022), pp. 1–42.

[3] Pedro Miguel da Costa Azevedo. "Exploring HLS Strategies in Computer Vision Algorithms". In: (2024).

[4] David Fosca Gamarra, Per Gunnar Kjeldsberg, and Henrik Sundbeck. "Lens Flare Attenuation Accelerator Design with Deep Learning and High-Level Synthesis". In: *2023 IEEE Nordic Circuits and Systems Conference (NorCAS)*. IEEE. 2023, pp. 1–7.

[5] Ioannis Kalomoiris et al. "An experimental analysis of the opportunities to use field programmable gate array multiprocessors for on-board satellite deep learning classification of spectroscopic observations from future ESA space missions". In: *European workshop on on-board data processing (OBDP2019)*. 2019.

[6] Hyeonwoo Kim et al. “An FPGA-Accelerated CNN with Parallelized Sum Pooling for Onboard Realtime Routing in Dynamic Low-Orbit Satellite Networks”. In: *Electronics* 13.12 (2024), p. 2280.

[7] HN Mahendra and S Mallikarjunaswamy. “An efficient classification of hyperspectral remotely sensed data using support vector machine”. In: *International Journal of Electronics and Telecommunications* 68 (2022).

[8] Peter Marwedel. *Embedded system design: embedded systems foundations of cyber-physical systems, and the internet of things*. Springer Nature, 2021.

[9] Romén Neris et al. “FPGA-Based implementation of a CNN architecture for the on-Board processing of very high-resolution remote sensing images”. In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 15 (2022), pp. 3740–3750.

[10] K Preethi and P Balasubramanian. “FPGA implementation of synchronous section-carry based carry look-ahead adders”. In: *2014 2nd International Conference on Devices, Circuits and Systems (ICDCS)*. IEEE. 2014, pp. 1–4.

[11] Xilinx. *Zynq-7000 SoC Data Sheet: Overview*. Version v1.11.1. Product Specification, DS190. Xilinx. 2018.

[12] AMD Xilinx. “Vitis AI User Guide (UG1414)”. In: (2022).

[13] Maciej Ziaja et al. “Benchmarking deep learning for on-board space applications”. In: *Remote Sensing* 13.19 (2021), p. 3981.