Predictive Modeling of FPGA Resource and Power Consumption for Configurable CNN Operators

Philippe Magalhães, Virginie Fresse, Benoît Suffran, Olivier Alata

Predictive Modeling of FPGA Resource and Power Consumption for Configurable CNN Operators

1st Philippe MAGALHÃES
Lab. H. Curien, UMR 5516
CNRS, IOGS, Univ. J. Monnet
Saint Etienne, France
philippe.magalhaes@univ-st-etienne.fr

2nd Virginie FRESSE
Lab. H. Curien, UMR 5516
CNRS, IOGS, Univ. J. Monnet
Saint Etienne, France
virginie.fresse@univ-st-etienne.fr

3rd Benoît SUFFRAN
ST Microelectronics
Grenoble, France
benoit.suffran@st.com

4th Olivier ALATA
Lab. H. Curien, UMR 5516
CNRS, IOGS, Univ. J. Monnet
Saint Etienne, France
olivier.alata@univ-st-etienne.fr

***Abstract*—As Convolutional Neural Networks (CNNs) continue to grow in complexity and accuracy, their deployment on embedded platforms requires hardware-aware optimizations to meet stringent constraints on logic resources, power, and latency. FPGAs offer an attractive solution because of their parallelism, reconfigurability, and energy efficiency. However, conventional FPGA design flows remain time-consuming and often lack early-stage estimation capabilities. This work introduces a library of parameterizable Intellectual Properties (IPs) for convolution, activation, and pooling, developed in VHDL and optimized for fixed-point arithmetic. IPs were designed to address various architectural trade-offs involving logic usage, DSP allocation strategy, parallelism, and power efficiency, while also supporting faster development through modular reuse. To accelerate design space exploration, a methodology is proposed for generating predictive mathematical models capable of estimating key FPGA resource metrics as functions of input bit widths. The models were validated with low prediction errors and coefficient of determination (R^2) values greater than 0.94, allowing accurate estimation of the resources and dynamic power without synthesis. This supports fast architectural decisions and paves the way for automated, resource-aware CNN deployment on FPGAs.**

***Index Terms*—CNN, FPGA, Mathematical Modeling, Resource Dimensionin**

I. Introduction

Convolutional Neural Networks (CNNs) are widely used in computer vision, natural language processing, and other machine learning tasks. To meet increasing complexity and computational demands [1] [2], CNNs are often deployed on GPUs for their parallelism and mature software support, although power usage limits them in energy-constrained scenarios. CPUs are accessible and easy to program, but lack parallelism and bandwidth. ASICs offer high efficiency and performance for specific tasks, yet have high development costs and no reconfigurability [3] [4] [5]. FPGAs balance parallelism, efficiency, and reconfigurability, making them suitable for embedded CNNs, though development typically requires low-level languages. Toolchains and abstraction layers can ease development, albeit reducing fine-grained control.

Recent studies have targeted the optimization of CNN deployments on FPGAs. For instance, Xu et al. [5] proposed a DSE model to estimate resource usage, enabling dynamic reconfiguration, while Kokkinis et al. [10] developed a high-level hls4ml-based model to predict MLP resource consumption. Li et al. [6] apply DSE to identify computation-intensive regions and propose optimal partitioning strategies to reduce energy. Shi et al. [7] address limited hardware by dynamically reconfiguring FPGA regions. Shao et al. [1] combine INT8 and INT16 quantization with systolic arrays to improve energy efficiency. Fuketa et al. [8] avoid DSPs by combining INT8 with residual vector quantization and dot-product approximation. Gundrapally et al. [9] describe an ultra-low-power accelerator for deep learning, emphasizing energy-aware design.

Despite optimization strategies and metrics for resource estimation, many FPGA-based CNN implementations still suffer from unbalanced resource usage. Design efforts often prioritize logic reduction, throughput, or power savings, while proportional utilization is overlooked, leading to overuse of some resources and underuse of others. Networks are frequently mismatched with the target FPGA, causing either underutilization or saturation. Moreover, early-stage power estimation is uncommon, and the development flow remains complex, requiring hardware expertise and lengthy synthesis, placement, and routing phases that hinder design space exploration.

To address these limitations, a reusable IP library was developed in VHDL for convolution, activation, and pooling, featuring parameterizable options to support diverse scenarios. Development time can be reduced while high performance is maintained through the reuse of these IPs. In addition, a predictive modeling framework was introduced to estimate resource and power consumption, from CNN parameters, enabling network adaptation to available FPGA resources or supporting platform selection. Although this study employed the Xilinx Zynq UltraScale+ ZCU104 and Vivado, the methodology is tool-independent and adaptable to other technologies, enabling improved design strategies and more efficient resource allocation in CNN deployment.

II. FPGA Hardware Platform

FPGAs provide a versatile and efficient platform for implementing both logical and arithmetic operations. Their architecture is based on configurable logic blocks, which include Look-Up Tables (LUTs), which can be optimized for combinational logic (LLUTs) or for local storage (MLUTs), flip-flops

(FFs) for data storage and synchronization, and a network of multiplexers that enable flexible routing and interconnection.

To support high-speed arithmetic, carry chain structures (CChains) are employed to propagate carry signals efficiently between adjacent logic elements, significantly accelerating addition and other arithmetic operations.

Memory resources are also varied and include Block RAM (BRAM) for centralized storage, UltraRAM for higher capacity and bandwidth, and distributed memory, which offers low latency storage close to the computation units.

Finally, Digital Signal Processing (DSP) blocks are dedicated hardware units designed to perform complex operations, such as multiplication and accumulation, with high precision.

Since FPGA architectures and resource distributions vary across models, families, and manufacturers, a thorough understanding of these characteristics is essential to effectively tailor implementations and maximize hardware utilization.

III. IP Library for CNN Components

To streamline CNNs deployment on FPGAs, a VHDL-based library of parameterizable IPs was developed. It includes convolution modules with fixed-point arithmetic, serial loading of 3×3 kernel coefficients stored locally to save memory, and parallel data input for higher throughput. The library also includes modules for ReLU, ReLU6, MaxPooling, and AveragePooling. To improve adaptability across FPGA architectures, multiple versions of convolution and pooling IPs were designed using distinct coding strategies, providing a range of resource footprints suited to various design constraints. Table I summarizes the key characteristics.

TABLE I
Characteristics of Developed IPs.

IP	DSP Usage	Logic Usage	Key Features
$Conv_1$	None	High	Only logic and Carry Chains; one convolution at a time.
$Conv_2$	1 DSP	Low	It reduces the use of logic; one convolution at a time.
$Conv_3$	1 DSP	Moderate	Two parallel convolutions; limited up to 8-bit operands.
$Conv_4$	2 DSPs	Moderate	Two parallel convolutions; each using a single DSP.
ReLU	None	Low	Only logic and Carry Chains.
ReLU6	None	Low	Only logic and Carry Chains.
MaxPooling 2x2 and 3x3	None	Low	It performs comparison using only logic and carry chains.
AvgPooling 2x2	None	Low	It uses shift operation instead of division.
AvgPooling $3x3_1$	None	Moderate	It uses shift operation instead of division.
AvgPooling $3x3_2$	1 DSP	Low	It uses DSP to reduce logic usage.

IV. Framwork for Prediting models

This section presents the framework used to construct predictive models for estimating the FPGA resource and power consumption from CNN IP configurations. The methodology is summarized in Fig. 1, which describes the modeling process. Each step plays a key role: synthesis-based data collection captures resource usage across configurations; data analysis reveals trends and correlations; regression fitting generates predictive equations; and validation ensures model accuracy.

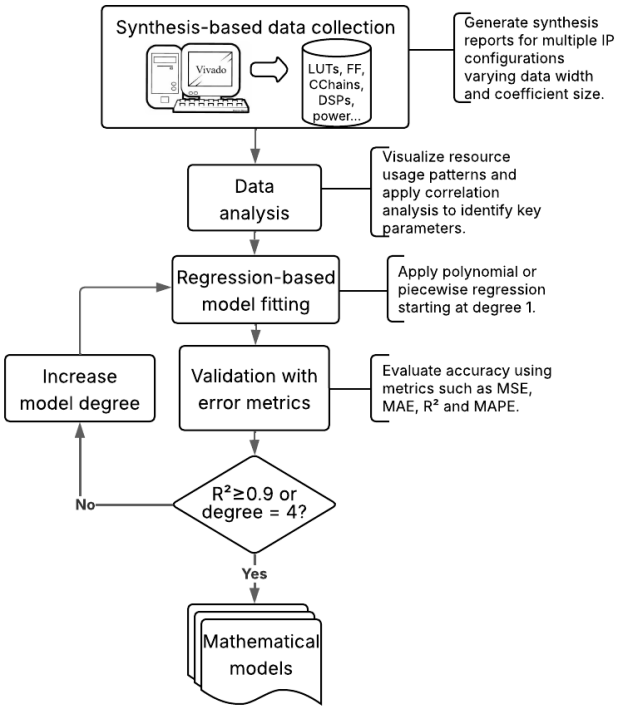


Fig. 1. Methodological Framework.

A. Synthesis-Based Data Collection

For the convolution functions, 196 configurations were used, generated by varying the data and coefficient widths from 3 to 16 bits, and for the IPs of the activation and pooling functions, the input bit widths were varied from 3 to 16 bits, resulting in 14 configurations. Synthesis was performed using Vivado 2024.2 on the Zynq UltraScale+ ZCU104 platform at 200MHz, and resource usage metrics, including LLUT, MLUT, FF, CChains, DSPs and dynamic power consumption, were recorded for each configuration and IP type. These measurements are specific to the target FPGA family and may vary with different architectures or synthesis tools.

B. Data Analysis

Pearson's correlation was employed for its simplicity and interpretability to capture linear relationships between data and coefficient widths and resource usage. Most convolution IPs exhibited strong correlations, with LLUT and MLUT values above 0.65, indicating that the input size significantly affects logic and memory utilization. FF usage also showed a strong dependence with coefficient size. In contrast, $Conv_3$ showed a lower correlation (0.497), suggesting a non-linear relationship likely caused by internal DSP parallelism. For the remaining CNN functions, the correlation between input width and LLUT usage exceeded 0.91. These results inform model selection, as strong correlations favor polynomial fitting, whereas weaker ones suggest nonlinear or piecewise approaches.

C. Model Construction

For this work, the usage of LLUTs was selected as the target variable due to their central role in combinational logic, their strong correlation with other resources, and their ability to reflect overall usage. Polynomial regressions [11] were applied to IPs whose resource usage was linearly correlated with input sizes, while a piecewise regression was used for IPs $Conv_3$, ReLU6 and MaxPooling 3x3.

Fig. 2 shows the scatter plot of the LLUT consumption for all configurations of the IP $Conv_1$, along with the surface generated from the obtained model, where d and c correspond to the number of data bits and coefficient bits, respectively. For $Conv_2$ and $Conv_4$, similar behaviors were observed, and the corresponding models are given in Equations 1 and 2. The piecewise model used for $Conv_3$ is shown in Fig. 3.

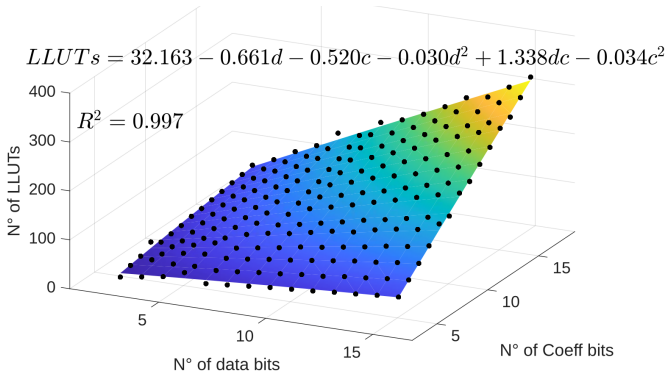


Fig. 2. LLUTs surface adjustment - $Conv_1$.

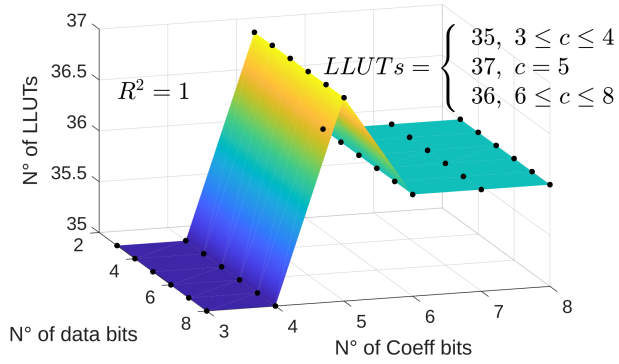


Fig. 3. LLUTs surface adjustment - $Conv_3$.

$$Conv_2 : LLUTs = 17.053 + 0.476d + 0.516c \quad (1)$$

$$Conv_4 : LLUTs = 20.886 + 1.004d + 1.037c \quad (2)$$

Figures 4 and 5 illustrate polynomial regression models fitted to the LLUT consumption data of the remaining CNN functions. These plots are two-dimensional, since only the input data width is varied, and first-degree polynomials were sufficient in all cases. For $ReLU6$ and $MaxPooling_{3x3}$,

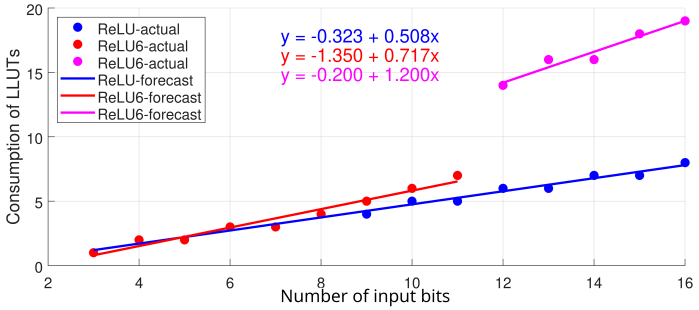


Fig. 4. LLUTs adjustment - Activation operations.

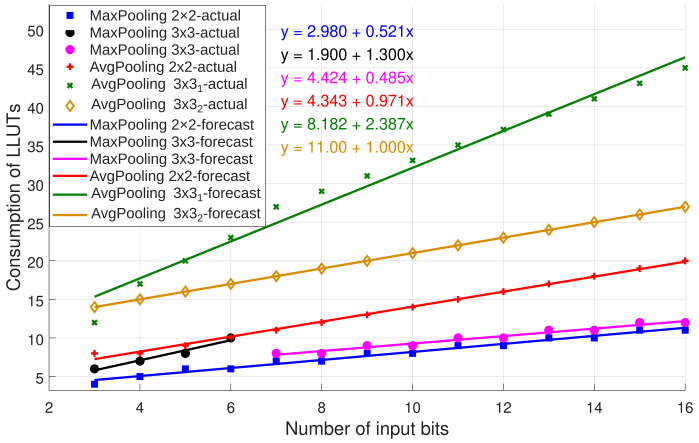
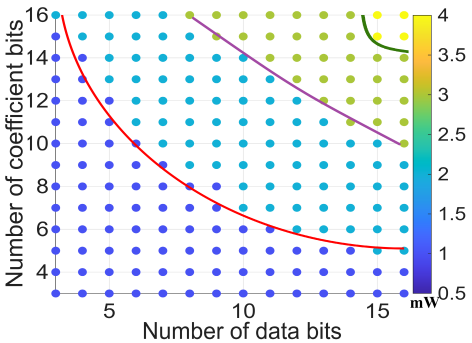


Fig. 5. LLUTs adjustment - Pooling operations.

piecewise regression was adopted due to observed discontinuities, likely caused by the internal structure of the LLUTs.

The dynamic power consumption pattern for the IP $Conv_1$ is illustrated in Fig. 6. Four consumption levels , ranging from 1 to 4 mW, each grouping a subset of IP configurations. In this case, polynomial regression was used to determine the threshold between levels. When the output of the function reaches or exceeds 0.5, it indicates a transition from one consumption level to the next. For example, if $f(d, c) \geq 0.5$, the consumption changes from 1mW to 2mW.



$f(d, c) = -0.0048d^2 - 0.005c^2 - 0.0007dc + 0.1562d + 0.1785c - 1.5084$
$g(d, c) = 0.0034d^2 + 0.0052c^2 + 0.0106dc - 0.1232d - 0.1527c + 1.0154$
$h(d, c) = -0.003d^2 - 0.003c^2 + 0.0028dc + 0.0562d + 0.0562c - 0.46$

Fig. 6. Power consumption - $Conv_1$.

For $Conv_2$ and $Conv_3$, the dynamic power remains constant in all configurations at 2mW. For $Conv_4$, two levels occur, 3mW and 4mW, with thresholds defined by Equation 3. For the remaining IPs, the dynamic power consumption remained constant for all configurations. ReLU, ReLU6 and MaxPooling consumed less than 1mW, Average Pooling 2×2 consumed 1mW, while both 3×3 versions consumed 2mW.

$$Conv_4 : f(d,c) = -0.0036d^2 - 0.0024c^2 + 0.0005dc + 0.1241d + 0.119c - 1.1894 \quad (3)$$

Note that the developed models apply specifically to the Xilinx UltraScale+ family. For other FPGA families or platforms from other manufacturers, new models must be constructed.

V. Validation

Validation was carried out in two stages: first, by predicting resource consumption for the convolutional, activation, and pooling layers of an arbitrary 8-bit CNN architecture; and second, by evaluating the accuracy using performance metrics.

A. Resource Consumption Predictions

Table II demonstrates how the models assist in selecting CNN IPs based on available FPGA resources. Different IP versions for the same function present distinct resource footprints, allowing better adaptation to hardware constraints. For the CNN architecture example, the difference between predicted and actual values was under 1.3% for most resources. The exception was the use of CChain, with a deviation of 9.5%, still acceptable for early-stage estimation.

TABLE II
Estimated resource usage for arbitrary CNN layers.

Layer	IP Used	# IPs	LLUT	FF	CChain	DSP	Power (mW)
Conv1	$Conv_1$	8	834	427	74	0	8
Activ1	$ReLU$	8	29	56	0	0	2
Pool1	Max_{2x2}	8	57	88	8	0	2
Conv2	$Conv_3$	8	288	246	0	8	16
Activ2	$ReLU6$	16	70	107	0	0	4
Pool2	Avg_{2x2}	16	192	208	32	0	16
Conv3	$Conv_2$	32	799	685	0	32	64
Activ3	$ReLU$	32	118	224	0	0	8
Pool3	Max_{2x2}	32	228	352	32	0	8
Total	—	—	**2615**	**2393**	**146**	**40**	**128**
Error	—	—	**1.25%**	**1.29%**	**9.5%**	**0%**	**0%**

B. Performance Evaluation of the Predictive Models

The accuracy of the models was assessed using the Root Mean Square Error (RMSE), Mean Absolute Error (MAE), Coefficient of Determination (R^2), and Mean Absolute Percentage Error (MAPE). Each model was initially fitted with a first-degree polynomial, with the degree increased up to 4 if R^2 was below 0.9, ensuring that the model reflected actual behavior. Table III shows high R^2 values and low errors for LLUT predictions. All models achieved R^2 over 0.94 and MAPE less than 8%, confirming their effectiveness for early estimation of FPGA resource and power consumption.

TABLE III
Error Metrics for Zynq UltraScale+ ZCU104 LLUT Models.

IP	MSE	MAE	R^2	MAPE (%)
$Conv_1$	16.244	3.054	0.997	3.038
$Conv_2$	0.498	0.538	0.941	2.134
$Conv_3$	0.00	0.00	1.00	0.00
$Conv_4$	0.379	0.518	0.989	1.342
$ReLU$	0.061	0.243	0.985	7.38
$ReLU6$	0.141	0.311	0.996	7.93
$MaxPooling$ 2x2	0.086	0.266	0.980	3.80
$MaxPooling$ 3x3	0.064	0.244	0.979	2.68
$AvgPooling$ 2x2	0.053	0.146	0.996	1.51
$AvgPooling$ $3x3_1$	1.833	1.050	0.980	4.66
$AvgPooling$ $3x3_2$	0.00	0.00	1	0.00

VI. Conclusion

This work introduced a library of parameterizable CNN IPs developed in VHDL and a predictive modeling framework to estimate FPGA resource and dynamic power consumption. The IPs provide flexibility to match the heterogeneous architectures of the Xilinx UltraScale+ family. Polynomial and piecewise regression models achieved R^2 over 0.94 for LLUT usage and dynamic power, enabling early-stage estimation to guide IP selection, optimize resource utilization, and reduce synthesis iterations. Future work includes expanding the IP library with new CNN functions, generalizing the models to support other FPGA vendors, incorporating factors like operating frequency, and integrating the modeling flow into an automated toolchain for resource-aware CNN deployment.

Acknowledgment

This work was sponsored by a public grant overseen by Auvergne-Rhône-Alpes region, Grenoble Alpes Metropole and BPIFrance, as part of project I-Démo Région "Green AI".

References

[1] Y. Shao, J. Shang, Y. Li, Y. Ding, M. Zhang, K. Ren and Y. Liu. "A Configurable Accelerator for CNN-Based Remote Sensing Object Detection on FPGAs". IET Computers & Digital Techniques, 2024.

[2] H. Ye. "Accelerating convolutional neural networks: Exploring FPGA-based architectures and challenges". Journal of Physics: Conference Series. 2786. 012004, 2024, doi: 10.1088/1742-6596/2786/1/012004.

[3] J. Jiang, Y. Zhou, Y. Gong, H. Yuan and S. Liu. "FPGA-based Acceleration for Convolutional Neural Networks: A Comprehensive Review", 2025, doi: 10.48550/arXiv.2505.13461.

[4] L. Chen, F. Luo, F. Wang and L. Lv. "Weather Recognition Based on a Field Programmable Gate Array and Lightweight Convolutional Neural Network". Electronics. 14(9):1740, 2025, 10.3390/electronics14091740

[5] Y. Xu, J. Luo and W. Sun. "Flare: An FPGA-Based Full Precision Low Power CNN Accelerator with Reconfigurable Structure". Sensors. 24. 2239, 2024, doi: 10.3390/s24072239.

[6] Z. Li and S. Bilavarn. "Improving the Energy Efficiency of CNN Inference on FPGA using Partial Reconfiguration". DASIP, 2024.

[7] K. Shi, M. Wang, X. Tan, Q. Li and T. Lei. "Efficient Dynamic Reconfigurable CNN Accelerator for Edge Intelligence Computing on FPGA". Information, 14, 194, 2023, doi: 10.3390/info14030194.

[8] H. Fuketa, T. Katashita, Y. Hori and M. Hioki, "Multiplication-Free Lookup-Based CNN Accelerator Using Residual Vector Quantization and Its FPGA Implementation," in IEEE Access, vol. 12, 2024.

[9] A. Gundrapally, Y. A. Shah, N. Alnatsheh, and K. K. Choi. "A High-Performance and Ultra-Low-Power Accelerator Design for Advanced Deep Learning Algorithms on an FPGA". Electronics, 13(13), 2024.

[10] A. Kokkinis, K. Siozios. "Fast Resource Estimation of FPGA-Based MLP Accelerators for TinyML Applications". Electronics 2025, 14, 247.

[11] D. C. Montgomery, E. A. Peck and G. G. Vining, "Introduction to Linear Regression Analysis", 6th, John Wiley & Sons, Inc., 2021, p.704