ECE 4300.01 Final Project

Multi-Platform Image Object Detection with YOLOv3 and RISC-V

Team Members:
Gloria Tovar, Ryan Agatep, Jose Hernandez, Daniel Pedro

Faculty Advisor:
Professor: Mohamed El-Hadedy Aly

Email: mealy@cpp.edu, gtovar@cpp.edu, rlagatep@cpp.edu, jmhernandez3@cpp.edu, dpedro@cpp.edu

California State Polytechnic University, Pomona
Department of Electrical and Computer Engineering, College of Engineering

***Abstract*—This project is centered around the deployment of the YOLOv3 (You Only Look Once Version 3) object detection system on the Nexys A7-100T FPGA board, integrating it with Vivado RISC-V capabilities. Leveraging the FPGA platform provides a unique advantage due to its parallel processing capabilities, rendering it well-suited for computationally intensive tasks like the object detection functionality offered by YOLOv3. The Nexys A7-100T board, equipped with the Xilinx Artix-7 FPGA, strikes a favorable balance between performance and available resources. The development process is facilitated by Vivado, an extensive development environment provided by Xilinx, enabling the design and implementation of FPGA projects. The incorporation of the RISC-V architecture enhances the adaptability and scalability of the YOLOv3 system. It serves to evaluate resource utilization, execution time performance, and the accuracy of object detection. The project involves comprehensive testing on various platforms, including a Mac computer, Windows Subsystem for Linux, and systems powered by AMD Ryzen and Intel Core processors. This multi-platform testing strategy ensures the versatility and robustness of the YOLOv3 implementation, assessing its performance across different computing environments.**

I. Introduction

This project investigates the implementation and performance evaluation of YOLOv3 [3] (You Only Look Once, Version 3), a cutting-edge object detection system, across various computing environments. At the heart of this study is the deployment of YOLOv3 on the Nexys A7-100T FPGA board, which employs the Xilinx Artix-7 FPGA with Vivado RISC-V integration [5]. This configuration facilitates an exploration of parallel processing capabilities essential for intensive tasks such as object detection, with a focus on accuracy and execution time.

Furthermore, the study extends to include diverse hardware configurations and operating systems, encompassing macOS, Windows Subsystem for Linux (WSL), and virtualized environments on AMD Ryzen and Intel Core processors [4]. This comparative approach aims to comprehend the influence of different setups on the operational efficiency of YOLOv3, emphasizing its adaptability and potential optimization in various applications. It is important to note that the team would be using darknet [2] to install YOLOv3.

The methodology involves rigorous experimentation, benchmarking, and detailed performance analysis, striving to strike a balance between theoretical exploration and empirical insights. This paper aims to present findings and insights that contribute to a broader understanding of the capabilities and potential applications of YOLOv3 in the fields of computer vision and machine learning.

II. Methodology

FPGA - The initial methodology to benchmark darknet (image detection software) is to use reconfigurable hardware to host soft-cores of different architectures: RISC-V, MIPS, and ARM. Since the logic slices, and internal memories are limited in the FPGA available for testing (Nexys-A7-100t), a single-core processor. This single-core processor(s), however, is a pipelined microarchitecture, like most modern microprocessors. Although the RISC-V soft-cores were successfully programmed onto the FPGA, the practicality to run demos was not adequate, as the kernel image and distribution (Debian) lacked basic functionalities such as gcc, make, and even internet connection.

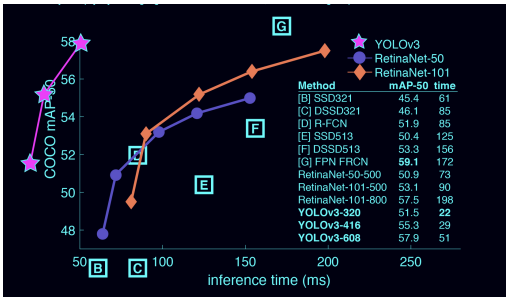

Fig. 1. Darknet Calculation with COCO-map50

QEMU (Emulation) - QEMU emulation [1] is a tool that uses dynamic machine language translation in conjunction with hardware virtualization technology. This was accomplished through the use of raw linux images loaded on a directory, and then executed with the QEMU command and parameters. Again, there were limitations, such as ram and storage space, and basic tools, that limited the testing of yolo models on these virtual environments.

Baremetal (Linux) - Running linux on VMs was viable because it offered all the tools and resources that the former two methods did not. Additionally, this method of testing was attractive to the particular machines available for testing, because they did not have to use their respective graphics card. The performance metrics are limited only to CPU usage, as this would make the control variables much more fair to this objective.Also, since the same linux distros were used (Ubuntu), there was more control of the biases at the end results.

Once again, the goal was to minimize biases between the different chips, so the base yolov3 with darknet was used. CUDA with darknet was not used because not all the chips had a complimentary GPU unit to accelerated the vectorized instructions.

III. Mathematical Model

To compare results, the team went to discover a website on how darknet with YOLOv3 is used on a COCO-map50. In Figure 1 demonstrates on how YOlOv3 is accurate and and fast when COCO-map50 is being used. The x-axis represents the time on how long YOLOv3 took to caputure the image results. The orange line and purple line represent on how YOLOv3 changes when there is a model change. From these analysis, the team will be doing different platforms to view the results to see how YOLOv3 would work in platforms and comparing the results to one another.

IV. Analysis

A. YOLOv3 Results Summary

YOLOv3 - Darknet and the object detection system YOLOv3 are tested on various operating systems, which are

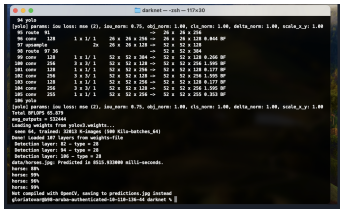

Fig. 2. YOLOv3 Horses Analysis in Mac Computer

Fig. 3. YOLOv3 Horses Ubuntu Analysis in Mac Computer

Ubuntu using a Virtual Machine and Windows Subsystem for Linux (WSL), macOS, and Windows. The goal is to see which operating system can finish detecting objects on an image the fastest. Running the YOLOv3 system and using the same image of a group of horses on each of these operating systems shows that execution time on macOS is the fastest being about 8.5 seconds, while Ubuntu running on a Virtual Machine through macOS is the slowest being about 30 seconds.

B. MacBook Pro 2019 Computer

For the MacBook Pro in question, the processor boasts a formidable 2.4 GHz 8-Core Intel Core i9. Despite the absence of a dedicated GPU in the MacBook Pro lineup, darknet was successfully uploaded and subjected to testing. The primary objective for this MacBook Pro revolves around determining the optimal operating system capable of swiftly detecting objects within images. Additionally, an intriguing exploration involves assessing the feasibility of leveraging both OpenCV and darknet for real-time camera detection on the Mac platform.

1) YOLOv3 on macOS: Darknet successfully underwent the upload process onto the macOS, accompanied by the addition of more images to the designated 'data' folder. Upon execution, YOLOv3 demonstrated remarkable proficiency in accurately identifying objects in the majority of the uploaded images. Notably, the processing time exhibited a notable improvement, showcasing a faster runtime. Among all the devices employed for this evaluation, it was observed that macOS outperformed the others, emerging as the swiftest in detecting objects within the images. In Figure 2, shows the results on how it perform when it was evaluating "horses" image.

2) YOLOv3 on Ubuntu 23.10: To facilitate the testing of YOLOv3 using darknet on a macOS computer, a Virtual Machine, specifically VMware Fusion, was employed to integrate Ubuntu 23.10. Darknet was effectively uploaded and configured on Ubuntu 23.10, mirroring the process on

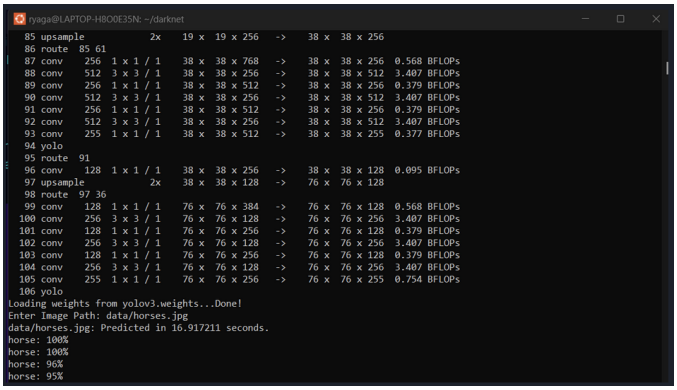

Fig. 4. YOLOv3 results in Windows Subsystem for Linux for Horses

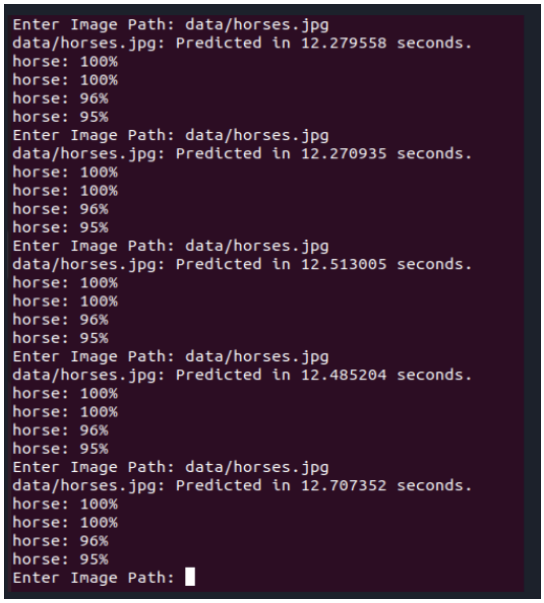

Fig. 5. YOLOv3 running on Ubuntu on x86-64

the macOS. Subsequently, a set of additional images was introduced into the designated 'data' folder for comprehensive testing. In Figure 3, demonstrates the results to the "horses" image. However, it's noteworthy that the runtime for running YOLOv3 on Ubuntu 23.10 did not exceeded that on the macOS, resulting in a considerably longer processing time.

C. Windows Subsystem for Linux

Windows Subsystem for Linux was used to install the Ubuntu 20.04 terminal on the Windows 11 system. From there, darknet was installed along with the YOLOv3 system and multiple images from the "data" folder were tested. The time it took to detect the objects in the "dog" image took 16.513504 seconds. In Figure 4, shows the results of "horses" images which includes the percentage that it was recognize and how long it took to process the timing of the image.

D. Ubuntu on x86-64

For bare metal Ubuntu on x86-64, there were competitive execution time(s). 12th Gen Intel(R) Core(TM) i7-12800HX

was the CPU used to run x86-64 benchmarks. It has a heterogenous core structure with 8 performance cores and 8 efficient cores, 24 total threads, and 25 MB of total cache. Using only YOLOv3 with darknet basic image object detection, this CPU was able to execute the "dog" image on an average of 11.6 seconds. At the end, x86-64 tended to perform similarly under a small margin. This reflects the methods used to acquire the results, as only CPUs were used, with no use of GPUs (CUDA), or OpenCV. In Figure 5, shows the final results when "horses" images was evaluated.

E. AMD Ryzen 9 5900HS processor

YOLOv3's performance was evaluated using Darknet on an AMD Ryzen 9 5900HS processor within an Ubuntu virtual machine under two distinct operational conditions. Initially, five trials each for dog and horse images were conducted after the virtual machine had been running continuously and utilized for various tasks. This was aimed at simulating a real-world, prolonged usage scenario, yielding average prediction times of 12.880 seconds for dog images and 12.451 seconds for horse images, reflecting consistent performance under extended operation. Conversely, a separate experiment was performed on a cold machine immediately after startup, representing a fresh operational state. In this setup, the first run typically extended to about 17 seconds, likely due to initial system resource allocation and stabilization, but subsequent trials showed average times nearly identical to those in the prolonged use scenario. This analysis demonstrates the AMD Ryzen 9 5900HS's robust performance in both extended and fresh operational states, highlighting its capability to maintain consistent efficiency in complex machine learning tasks like YOLOv3, irrespective of different system usage patterns.

V. Conclusion

In this investigation into the implementation and performance evaluation of YOLOv3, a leading object detection system, across various computing environments, we have unveiled significant insights. Central to our study was the deployment of YOLOv3 on the Nexys A7-100T FPGA board, integrated with Vivado RISC-V, which revealed the potential and limitations of FPGA platforms in handling intensive tasks like object detection. While our research highlighted the FPGA's promise, particularly in parallel processing, practical challenges related to internal memory and basic functionality constraints were evident. In contrast, more traditional computing environments, such as macOS, Windows Subsystem for Linux, and virtualized setups on AMD Ryzen and Intel Core processors, demonstrated a higher level of adaptability and efficiency in running YOLOv3. Remarkably, macOS proved to be the most efficient in execution time, suggesting its suitability for object detection tasks even without GPU support. The study's comparative analysis across various hardware and software configurations not only underscored the adaptability of YOLOv3 but also its consistent performance on x86-64 platforms, irrespective of the lack of GPU acceleration and OpenCV integration. This research, therefore, not only contributes to a deeper understanding of the capabilities and potential applications of YOLOv3 in the realms of computer vision and machine learning but also highlights the critical role of the computing environment in optimizing the operational efficiency of advanced object detection systems. Future research directions could include enhancing the usability of FPGA for similar tasks or delving into the integration of GPUs and advanced libraries to further refine performance across a spectrum of computing platforms.

VI. Roles and Acknowledgement

A. Ryan Agatep

Responsible for the performance analysis of YOLOv3 on Ubuntu WSL using the Windows 11 system, which includes conducting tests and analyzing data on multiple images to be able to compare with other systems. Contributed by writing the project's abstract and various parts of the analysis.

B. Jose Hernandez

Responsible for the performance analysis of YOLOv3 on the AMD Ryzen 9 5900HS, including setup on an Ubuntu VM, conducting tests, and analyzing data. Contributed to writing the project's introduction and part of the analysis.

C. Daniel Pedro

Performance metrics for YOLOv3 on Intel I7-1200HX. Tasks include compiling darknet on native intel chip, as well as running image detection on machine for performance metrics. RISC-V soft-core on FPGA was implemented on Nexys A7-100t, which booted from sd card with kernel image.

D. Gloria Tovar

Responsible for the performance analysis of YOLOv3 in a Mac Computer. Set up a darknet in the macOS and tested YOLOv3 from the terminal. To get more testing done, set up Ubuntu on Mac Computer in VMWare Fusion. Compared results from Linux and macOS system for two different images to compare from each other and from other data as well. Also, contributed in writing the presentation and the paper for the Multi-Platform Image Object Detection with YOLOv3 and RISC-V project.

References

[1] Michael Howard and R. Bruce Irvin. Esp32: Qemu emulation within a docker container. In Kohei Arai, editor, *Proceedings of the Future Technologies Conference (FTC) 2023, Volume 3*, pages 63–80, Cham, 2023. Springer Nature Switzerland.

[2] Joseph Redmon. Darknet: Open source neural networks in c. http://pjreddie.com/darknet/, 2013–2016.

[3] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement, 2018.

[4] M. Subramon, D. Kramer, and I. Paul. Amd ryzen™ 7040 series: Technology overview. In *2023 IEEE Hot Chips 35 Symposium (HCS)*, pages 1–27, Los Alamitos, CA, USA, aug 2023. IEEE Computer Society.

[5] E. Tarassov. Amd/xilinx vivado block designs for fpga risc-v soc running debian linux distro, 2023. Accessed on Nov. 30, 2023.