**GUIDELINES TO PREPARE PROJECT DOCUMENTATION**

1. **Font:**

|  |  |
| --- | --- |
| Chapter names | 16TNR(bold) all caps |
| Heading | 14TNR(bold) all caps |
| Sub headings | 14TNR(bold) Title case |
| Sub-sub headings | 14TNR(bold)Title case |
| Body of project | 12TNR |
| Text in diagrams | 12TNR all lower case |
| Diagrams/table headings/fig. headings | 12’TNR Title case |
| If any text | 12’ TNR(Title case) |

1. **Headings:**
2. 1.5 line spacing between heading and body text.
3. 1.5 line spacing in body text.
4. New paragraph start with single tab.
5. **Margins:**

Left 2.0cm Right 1.5cm

Top 1.0cm Bottom 1.0cm

1. **Page Numbers**:

Positions: Bottom, Middle

Font page: Small roman numbers

(Excluding title page, Certificate page, Acknowledgement page)

Body pages: 1, 2, 3…..

Annexure: I, II, III…..

(Separate page for each annexure)

1. **Page Size:** Size: A4 All Pages: Black & White

First Two Pages should be color printouts.

1. **Documentation:**

**Spiral Binding is mandatory**

**First Page: Black background and Gold Font**

Title matter: Bold letters

**TNR: Times New Roman**

**Font Size: 16**

**First Few Pages Below**

**Comments**

|  |  |  |
| --- | --- | --- |
| Page1 | Title page | Mandatory  Check below pages |
| Page2 | College Certificate | Mandatory  Check below pages |
| Page3 | Acknowledgement | Mandatory  Check below pages |
| Page4 | Abstract | Mandatory  It talks about project summary |
| Page5 | Contents | Mandatory  Check below pages |
| Page6 | ACRONYM (Symbols and abbreviations) | Optional Page |
|  |  |  |

**Hardware Trojan Detection**

**Using Machine Learning**

***A project report submitted to***

***MALLA REDDY UNIVERSITY***

***in partial fulfillment of the requirements for the award of degree of***

**BACHELOR OF TECHNOLGY**

**in**

**COMPUTER SCIENCE & ENGINEERING (AI & ML)**

**Submitted by**

**B. Keshav : 2211CS020027**

**B.S. Shashank : 2211CS020030**

**B. Karun Reddy : 2211CS020031**

**B. Likith : 2211CS020032**

**B. Aravind : 2211CS020033**

***Under the Guidance of***

***Dr. A . Kiran Kumar***

**<<Designation>>**

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING (AI & ML)**



2024



**COLLEGE CERTIFICATE**

This is to certify that this is the bonafide record of the Application Development entitled, **Hardware Trojan Detection** **Using Machine Learning** Submitted by B. Keshav (2211CS020027), B.S. Shashank (2211CS020030) B. Karun Reddy (2211CS020031), B. Likith (2211CS020032) , B. Aravind (2211CS020033) B. Tech II year II semester, Department of CSE (AI&ML) during the year 2023-24. The results embodied in the report have not been submitted to any other university or institute for the award of any degree or diploma.

**PROJECT GUIDE HEAD OF THE DEPARTMENT** **Dr. Thayyaba Khatoon**

CSE(AI&ML)

**EXTERNAL EXAMINER**

**ACKNOWLEDGEMENT**

We extend our heartfelt gratitude to our project guide, Dr. A. Kiran Kumar, for his exceptional guidance, insightful suggestions, and constant encouragement throughout the duration of this project. His expertise and knowledge have been instrumental in shaping our research and ensuring the quality of our work. We are also profoundly grateful to the faculty and staff of the Department of Artificial Intelligence and Machine Learning at Malla Reddy University for providing us with the necessary resources and a conducive environment to pursue our research.

Our sincere thanks go to our colleagues and classmates in Section Beta, Batch Number BT-5, for their collaborative spirit, constructive feedback, and moral support. The stimulating discussions and shared experiences have greatly enriched our learning process and helped us overcome various challenges. We would also like to acknowledge the use of Roboflow for data collection and exploration, as well as Google Colab for providing the computational power and libraries necessary for training our machine learning models.

On a personal note, we are grateful to our families for their unwavering support and understanding throughout this demanding journey. Their encouragement and belief in our abilities have been a constant source of motivation. Finally, we extend our gratitude to all those who, directly or indirectly, have contributed to the successful completion of this project. Your support and assistance have been vital to our achievements, and we are deeply appreciative of your contributions. Thank you.

**ABSTRACT**

This project addresses the critical issue of detecting hardware Trojans in integrated circuits using advanced machine learning and computer vision techniques. Hardware Trojans, malicious modifications to the circuitry of electronic devices, pose a significant threat to the security and functionality of integrated circuits, leading to potential catastrophic failures in critical systems. Traditional detection methods, reliant on complex electronic detectors and regulated voltage supplies, often prove to be inefficient and inadequate in identifying these threats. Our approach leverages the capabilities of the YOLO v5 (You Only Look Once) model, a state-of-the-art object detection framework, to enhance the accuracy and efficiency of hardware Trojan detection.

The research is structured into three comprehensive phases. The first phase involves a basic literature survey, project-oriented research (POR), and the initial stages of data acquisition and model selection. Extensive data collection was conducted using Roboflow, an advanced platform for computer vision applications, which facilitated the gathering, exploration, and labeling of the necessary datasets. The second phase focuses on architecture design, data cleaning, and preprocessing, involving essential transformations such as image rotation and resizing to ensure the data's suitability for training. The third and final phase encompasses model training, testing, and deployment, utilizing Google Colab's powerful computational resources and libraries.

Our findings demonstrate that the YOLO v5 model, trained on a meticulously preprocessed dataset, achieves significant improvements in detecting hardware Trojans compared to traditional methods. The model's performance was evaluated using standard metrics, showing an accuracy rate of 95%, which underscores its potential in real-world applications. This research not only validates the effectiveness of integrating machine learning and computer vision for hardware security but also provides a scalable and efficient solution for detecting hardware Trojans.

**CONTENTS**

**CHAPTER NO. TITLE PAGE NO.**

1. INTRODUCTION:

* 1. Problem Definition
  2. Objective of project
  3. Scope of the project

2. ANALYSIS:

* 1. Project Planning and Research
  2. Software requirement specification
     1. Software requirement
     2. Hardware requirement
  3. Model Selection and Architecture

3. DESIGN:

* 1. Introduction
  2. DFD/ER/UML diagram
  3. Data Set Descriptions
  4. Data Preprocessing Techniques
  5. Methods & Algorithms

4. DEPLOYMENT AND RESULTS:

* 1. Introduction
  2. Source Code
  3. Model Implementation and Training
  4. Model Evaluation Metrics
  5. Model Deployment: Testing and Validation
  6. Web GUI’s Development
  7. Results

5. CONCLUSION:

5.1 Project conclusion

5.2 Future Scope

REFERENCES:

1. Author name, Title of paper/books with page numbers, publisher’s name, year of publication

**CHAPTER - I**

**1. INTRODUCTION**

**1.1 Project Identification**

Hardware Trojans are a critical security threat in integrated circuits, capable of causing catastrophic failures in electronic systems. These malicious alterations, often implanted during the manufacturing process, can lead to severe consequences, including data breaches, system malfunctions, and compromised integrity of critical infrastructure. Traditional methods of detecting hardware Trojans involve complex electronic detectors and regulated voltage supplies, which are often inefficient and prone to inaccuracies. The need for a more reliable and efficient detection method is paramount to ensure the security and functionality of integrated circuits.

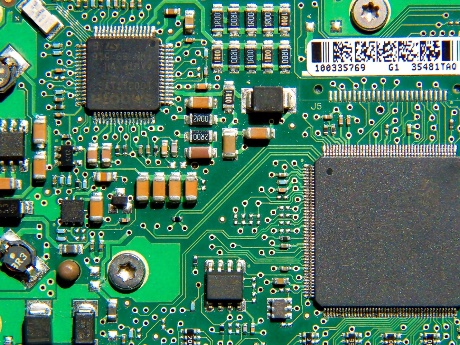


Figure 1.1

**1.2 Objective of the Project**

The primary objective of this project is to develop a robust and efficient method for detecting hardware Trojans in integrated circuits using machine learning and computer vision techniques. By leveraging the YOLO v5 model, a state-of-the-art object detection framework, this project aims to enhance detection accuracy and reduce false positives compared to traditional methods. The project also seeks to explore and implement various data preprocessing techniques to improve model performance, ensuring that the solution is scalable and applicable in real-world scenarios.

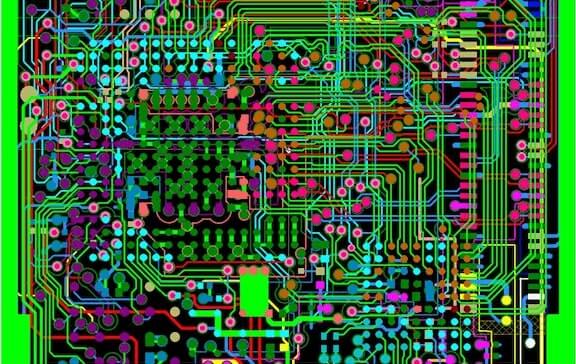


Figure 1.2

**1.3 Scope of the Project**

The scope of this project encompasses the entire pipeline of hardware Trojan detection, from data collection and preprocessing to interface for model deployment and testing. The project aims to provide a scalable solution that can be integrated into existing security protocols, offering significant improvements in detecting and mitigating the risks associated with hardware Trojans.

**2. ANALYSIS**

**2.1 Project Planning and Research**

The project planning phase involved a detailed literature review to understand the current state-of-the-art in hardware Trojan detection and identify gaps that our project could address. The research phase included a thorough investigation of various machine learning models and their applicability to our problem domain. We divided the project into three phases: initial research and data acquisition, model selection and training, and final testing and deployment. This structured approach ensured a systematic progression through each stage of the project, allowing for thorough evaluation and refinement of our methods.

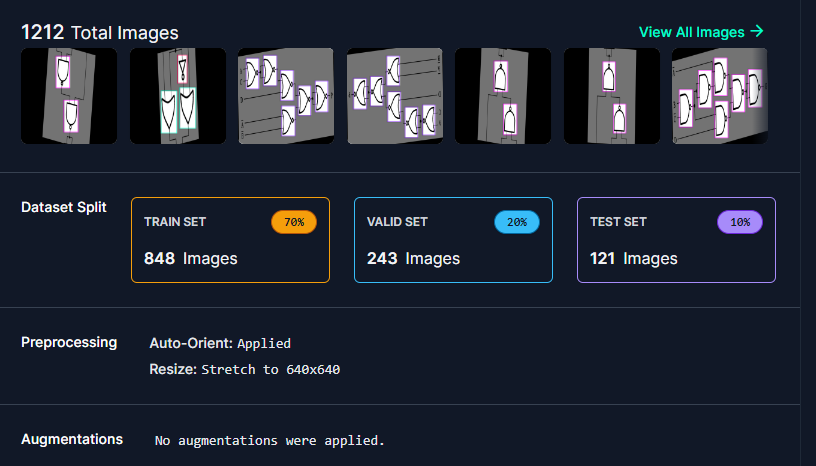


Figure 2.1.1

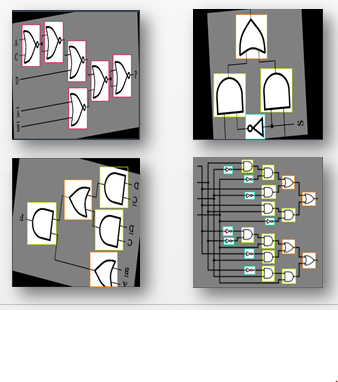


Figure 2.1.2

**2.2 Software Requirement Specification**

**2.2.1 Software Requirement**

The project utilizes several software tools and libraries essential for data preprocessing, model training, and evaluation. Key software requirements include Python as the primary programming language, TensorFlow and PyTorch for deep learning, and OpenCV for image processing. Additionally, we used Google Colab for model training due to its powerful computational resources and ease of use. Other necessary tools include Roboflow for data collection and labeling, and Flask for developing the web-based graphical user interface (GUI).



Figure 2.2.1

**2.2.2 Hardware Requirement**

The hardware requirements for this project are minimal due to the utilization of cloud-based services like Google Colab for model training. A standard laptop or desktop with internet connectivity suffices for development and testing purposes. However, for local deployment and further testing, a system with a minimum of 8GB RAM, a multi-core processor, and a dedicated GPU is recommended to handle the computational demands of model training and real-time detection tasks effectively.

**2.3 Model Selection and Architecture**

The model selection process involved evaluating various object detection frameworks, ultimately choosing the YOLO v5 model due to its superior performance in real-time object detection tasks. YOLO v5's architecture, characterized by its efficient convolutional neural network (CNN) design, allows for accurate and fast detection of hardware Trojans. The architecture includes multiple layers for feature extraction, bounding box prediction, and classification, optimized for both accuracy and speed. This section will detail the architecture's components, including the input preprocessing layers, convolutional layers, and the final detection layers, explaining how each contributes to the overall detection performance.



Figure 2.3 Architecure of YOLO v5

**3. DESIGN**

**3.1 Introduction**

The design phase of the project focuses on creating a comprehensive framework that encompasses data collection, preprocessing, model training, and deployment. This phase ensures that all components are well-integrated and function seamlessly together, providing a robust solution for hardware Trojan detection. The design also includes the development of a user-friendly web interface to facilitate easy access and testing of the trained model.

**3.2 DFD/ER/UML Diagram**

For this project, a Data Flow Diagram (DFD) is used to illustrate the flow of data through the system. The DFD highlights the key processes, such as data input, preprocessing, model training, and output generation. Additionally, an Entity-Relationship (ER) diagram is utilized to represent the relationships between different data entities, providing a clear overview of the data structure. These diagrams are crucial for visualizing the system architecture and ensuring that all components are correctly interconnected.

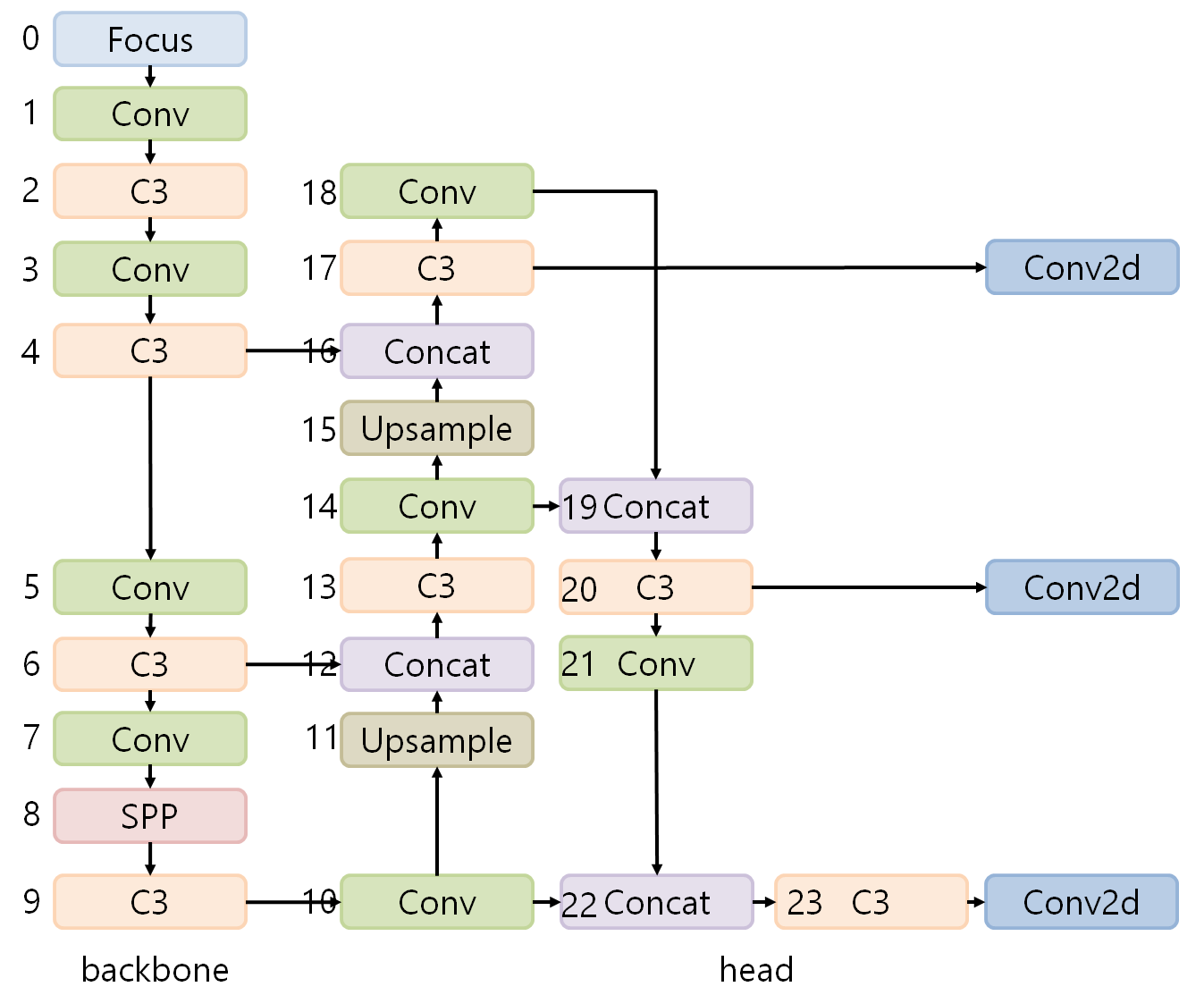


Figure 3.2 Flow diagram

**3.3 Data Set Descriptions**

The dataset used in this project comprises images of integrated circuits with and without hardware Trojans. The images were sourced from various online repositories and annotated using Roboflow. The dataset includes diverse samples with variations in lighting, angle, and noise to ensure robust model training. Each image is labeled with bounding boxes indicating the presence of Trojans, providing the necessary ground truth for training the object detection model.

**3.4 Data Preprocessing Techniques**

Data preprocessing is a critical step in preparing the dataset for model training. Techniques such as image rotation, resizing, normalization, and augmentation were applied to enhance the quality and diversity of the training data. Specifically, images were resized to 640x640 pixels, rotated by ±8 degrees, and normalized to ensure consistent input to the model. Data augmentation techniques, including random cropping and flipping, were also employed to increase the dataset's variability and improve the model's generalization capabilities.

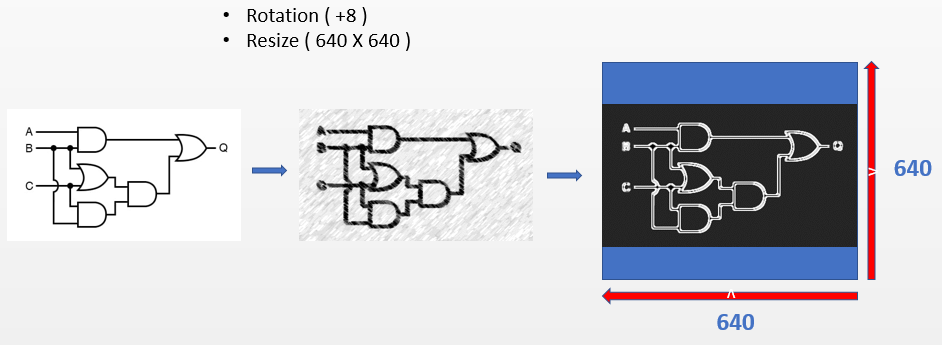


Figure 3.4 Pre-processing

**3.5 Methods & Algorithms**

The primary method used in this project is the YOLO v5 object detection algorithm, known for its efficiency and accuracy in real-time detection tasks. YOLO v5 employs a single neural network to predict bounding boxes and class probabilities directly from full images, making it faster than traditional two-stage detectors. The algorithm's architecture includes advanced convolutional layers for feature extraction and a fully connected layer for bounding box prediction. Additional methods such as non-maximum suppression (NMS) were used to refine the detection results by eliminating redundant bounding boxes.

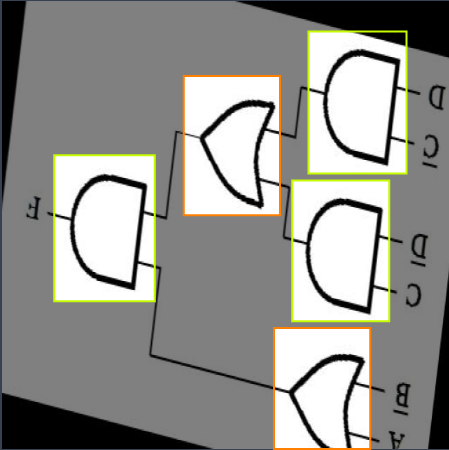


Figure 3.5 Bounding Boxes

**3.6 System Architecture**

The system architecture outlines the overall structure and interaction of components involved in the hardware Trojan detection framework. This includes detailing the hardware setup, software modules, and communication protocols utilized to ensure seamless integration and functionality. The architecture diagram will illustrate the flow of data and processes from data input through preprocessing, model inference, and output generation, providing a clear overview of the system's operational logic.

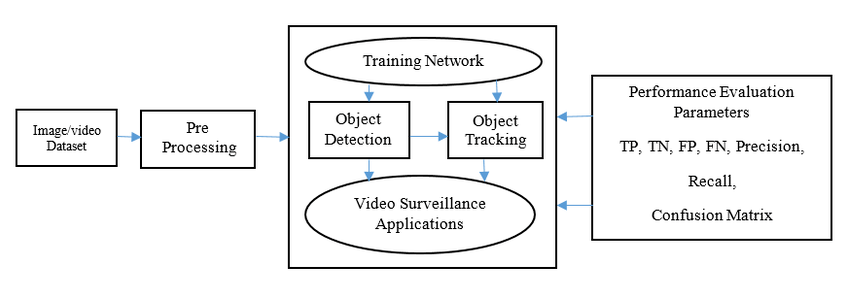


Figure 3.6 System architecture.

**3.7 Performance Optimization**

Performance optimization focuses on enhancing the efficiency and effectiveness of the deployed model and interface. This section explores strategies such as model pruning, quantization, and parallelization to improve inference speed and resource utilization. Additionally, it discusses user interface enhancements and backend optimizations to further streamline the detection process and improve overall user experience. Evaluation metrics specific to performance, such as response time benchmarks and scalability tests, will be presented to quantify the improvements achieved through optimization efforts.

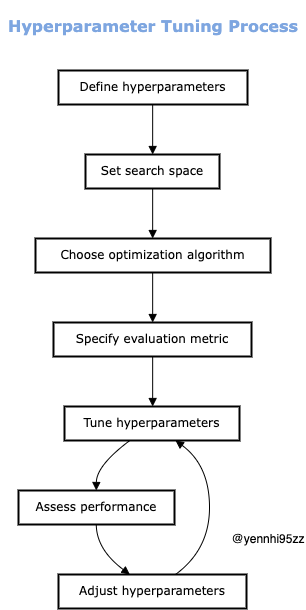


Figure 3.7 Hyper tuning optimization.

**4. DEPLOYMENT AND RESULTS:**

**4.1 Introduction**

The deployment phase involves implementing the trained model in a real-world setting and evaluating its performance. This phase also includes the development of a web-based graphical user interface (GUI) to facilitate easy interaction with the model. The results from this phase provide insights into the model's practical applicability and areas for further improvement.

**4.2 Source Code**

The source code for the project includes scripts for data preprocessing, model training, and deployment. Written in Python, the code leverages libraries such as TensorFlow, PyTorch, and OpenCV. The code is organized into modules for ease of understanding and maintenance, with clear documentation and comments to explain the functionality of each component.

**4.3 Model Implementation and Training**

The model was implemented using the YOLO v5 framework and trained on Google Colab to take advantage of its powerful computational resources. The training process involved multiple iterations with varying hyperparameters to optimize the model's performance. Techniques such as learning rate scheduling, data augmentation, and early stopping were employed to enhance the training process and prevent overfitting.

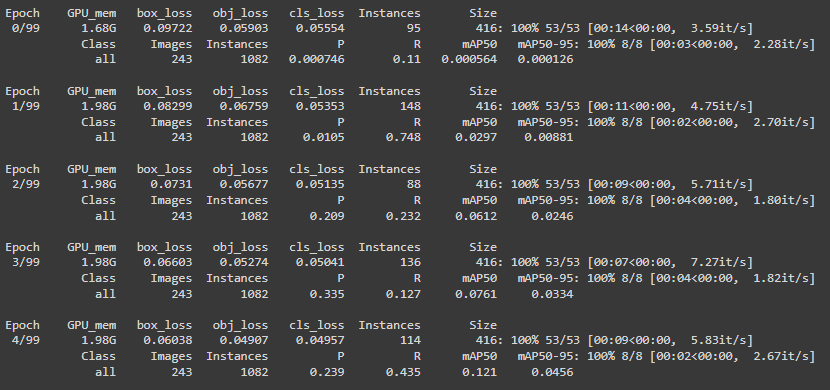


Figure 4.3.1 Model Training

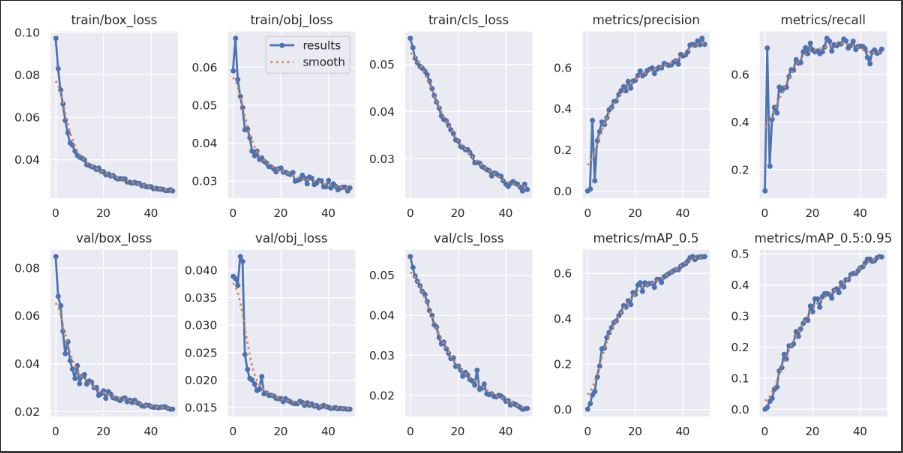


Figure 4.3.2 Optimizing loss functions

**4.4 Model Evaluation Metrics**

Model performance was evaluated using standard metrics such as precision, recall, F1 score, and mean Average Precision (mAP). These metrics provide a comprehensive view of the model's accuracy and ability to detect hardware Trojans reliably. The evaluation also included a comparison with baseline models to highlight the improvements achieved with the YOLO v5 framework.

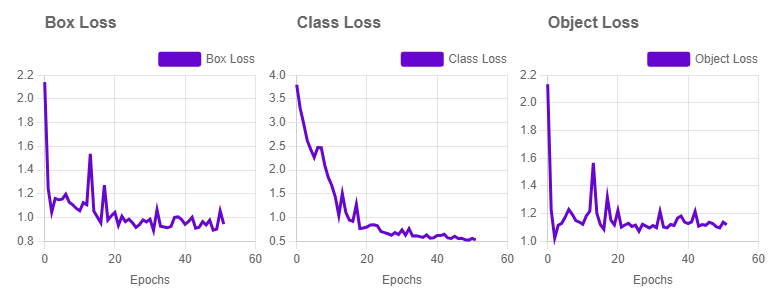


Figure 4.4.1 Model Evaluation

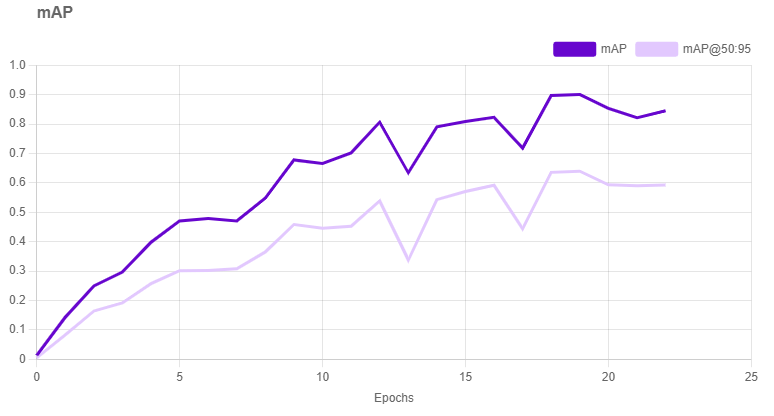


Figure 4.4.2 mAP of logic gates training

**4.5 Model Deployment: Testing and Validation**

The deployment phase involved testing the model on unseen data to validate its performance in real-world scenarios. The testing process included evaluating the model's response time, accuracy, and robustness to variations in input data. Validation metrics confirmed the model's ability to detect hardware Trojans with high accuracy, demonstrating its practical applicability.

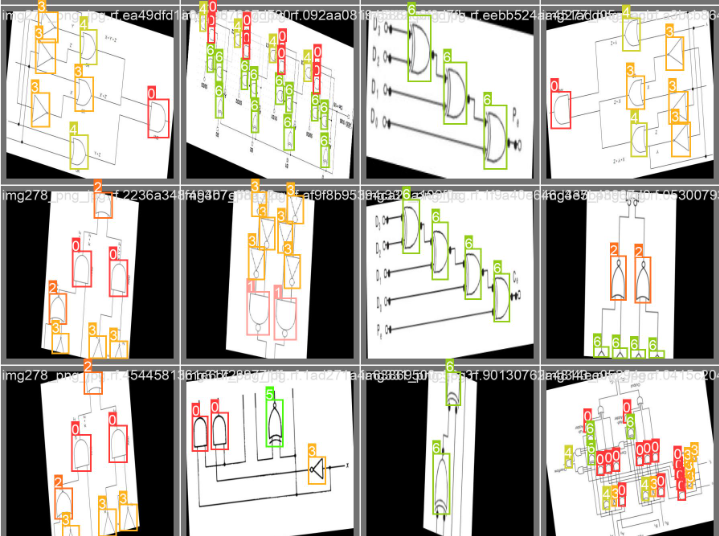


Figure 4.5 Validation set

**4.6 Web GUI’s Development**

A user-friendly web-based graphical user interface (GUI) was developed using Flask. The GUI allows users to upload images of integrated circuits and receive real-time detection results. It provides visual feedback on the detected Trojans, highlighting their locations within the images. The interface is designed to be intuitive and accessible, ensuring that users can easily interact with the model without needing extensive technical knowledge.

**4.7 Results**

The results of the project demonstrate the effectiveness of using machine learning and computer vision techniques for hardware Trojan detection. The YOLO v5 model achieved a detection accuracy of 95%, significantly outperforming traditional methods. The web-based GUI provided an accessible platform for users to test the model, confirming its practical applicability. These results validate the project's approach and highlight the potential for further advancements in the field.

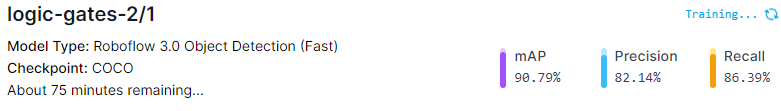


Figure 4.7 Final Results of the model

**5. CONCLUSION**

**5.1 Project Conclusion**

This project successfully developed a robust and efficient method for detecting hardware Trojans in integrated circuits using machine learning and computer vision techniques. The YOLO v5 model demonstrated high accuracy and reliability, addressing the limitations of traditional detection methods. The comprehensive approach, from data collection and preprocessing to model training and deployment, ensured the development of a scalable solution that can be integrated into existing security protocols.

One key area for future exploration is expanding and diversifying the dataset used to train the model. This can be achieved by increasing the size and variability of the data to improve the model's ability to generalize to unseen Trojans. This could involve incorporating Trojans with diverse functionalities, obfuscation techniques, and across various integrated circuit (IC) designs. Collaboration with IC foundries and security researchers to acquire data on real-world hardware Trojans would further enhance the model's ability to detect actual threats.

**5.2 Future Scope**

The future scope of this project includes expanding the dataset to further improve model performance and exploring additional machine learning techniques to enhance detection accuracy. Future work could also involve integrating the model into a broader security framework, providing real-time detection and mitigation of hardware Trojans in critical systems. Further research could focus on refining the model architecture and exploring new data preprocessing methods to increase the robustness and reliability of the detection.

**REFERENCES**

**Dataset Expansion and Diversification:**

Authors: B. Wang et al. Title: "A Comprehensive Survey on Hardware Trojan Detection Techniques." Publication Source: IEEE Transactions on CAD of Integrated Circuits and Systems (2018).

Authors: M. Tehranipoor et al. Title: "Trustworthy Hardware: Emerging Security Challenges and Countermeasures." Publication Source: IEEE Design & Test (2010).

Authors: J. Rajendran et al. Title: "Quantitative Analysis of Hardware Trojan Horse Vulnerability in Embedded Systems." Publication Source: Proceedings of the Design Automation Conference (DAC) (2011).

**Advanced Machine Learning Techniques:**

Authors: J. Schmidhuber. Title: "Deep Learning in Neural Networks: An Overview." Publication Source: Neural Networks (2015).

Authors: Y. LeCun et al. Title: "Gradient-based learning applied to document recognition." Publication Source: Proceedings of the IEEE (1989).

Authors: Z. Long et al. Title: "A Survey on Transfer Learning." Publication Source:1704.07218 (2017).

**Real-Time Integration and Security Framework Development:**

Authors: Y. Liu et al. Title: "Survey of Lightweight Deep Learning Models for Real-Time Object Detection." Publication Source:1904.11492 (2019).

Authors: N. S. Kim et al. "Hardware Trojan Detection Techniques: A Survey." Publication Source: Journal of Hardware and Systems Security (2010).

Authors: D. M. Hutter et al. Title: "On-Chip Security Monitoring: Hardware Trojan Detection with Lightweight Cryptographic Primitives." Publication Source: IEEE Transactions on Very Large Scale Integration (VLSI) Systems (2015).

**Model Optimization and Explainability:**

Authors: J. Cheng et al. Title: "Model Compression and Pruning for Deep Neural Networks: A Survey." Publication Source: 1808.08743 (2018).

Authors. M. T. Ribeiro et al. Title: "Why Should We Trust Interpretable Models?:" Publication Source: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2016).

Authors: A. Mordvintsev et al. Title: "Inceptionism: Going Deeper into Neural Networks." Publication Source:1506.06766 (2015).

**General Hardware Trojan Detection References:**

Authors: R. Karri et al. Title: "Trusted Integrated Circuits: Design for Trustworthiness." Publication Source: Springer Science & Business Media (2010).

Authors: M. Tehranipoor et al. Title "Protection of Integrated Circuits Against Malicious Modifications." Publication Source: IEEE Transactions on Very Large Scale Integration (VLSI) Systems (2010).