LABEL LENS : Label Detection using Machine Learning

# INDEX

1. **Introduction**

**2. Problem Definition**

**2.1. Problem Statement**

**2.2. Literature Review**

**3. Objectives**

**3.1. Primary Objectives**

**3.2. Secondary Objectives**

**4. Methodology**

**4.1. Approach**

**4.2. Procedures**

**5. Project Execution**

**5.1. Planning and Design**

**5.2. Implementation**

**6. Tools and Techniques Used**

**6.1. Tools**

**6.2. Techniques**

**7. Partial Results**

**7.1. Initial Findings**

**7.2. Improvements with GUI Integration**

**8. Results and Discussion**

**8.1. Final Results**

**8.2. Discussion**

**9. Prototype (Hardware/Software)**

**9.1. Prototype Description**

**9.2. Development Process**

**9.3. Testing and Validation**

**10. Conclusion**

**10.1. Summary**

**10.2. Personal Reflection**

**11. Visuals**

# **Introduction**

In the contemporary manufacturing and packaging industry, the accuracy and clarity of product labels are paramount for quality control. Labels provide crucial information such as compliance marks, recycling instructions, and branding details, which are essential for regulatory adherence, consumer safety, and brand reputation. Traditional manual inspection processes are labor-intensive and prone to human error, leading to inefficiencies and inaccuracies. Addressing these challenges, our project aims to develop an automated label detection system using advanced machine learning techniques. This system is designed to streamline the inspection process, significantly enhancing efficiency and reliability. Our system leverages YOLOv5, a state-of-the-art object detection model known for its high speed and accuracy. YOLOv5 was trained on a dataset of images from smartphone boxes of two brands: Samsung and OnePlus. To create a robust training dataset, the images were meticulously annotated using LabelImg, an open-source annotation tool. This comprehensive preparation enabled the YOLOv5 model to accurately learn and identify different labels.

The ultimate goal of our project is to integrate the trained YOLOv5 model into a user-friendly graphical user interface (GUI) application. Developed using CustomTkinter and OpenCV, the application provides real-time label detection and verification. Users can quickly and accurately inspect and validate labels on smartphone boxes, significantly reducing the manual effort involved. By automating the inspection process, our system boosts productivity and reliability in label verification, offering a valuable tool for the manufacturing sector. The intuitive design of the GUI ensures that operators with minimal technical expertise can easily use the system, making it a practical and efficient solution for real-world applications. Our project stands out by addressing a critical need in the manufacturing industry: the need for an efficient and reliable method to ensure the accuracy of product labels. By leveraging the power of machine learning and advanced computer vision techniques, we provide a solution that not only meets but exceeds the capabilities of traditional manual inspection methods. The integration of these technologies into a cohesive system offers significant benefits, including reduced labor costs, increased inspection speed, and enhanced accuracy, all of which contribute to improved overall productivity.

In conclusion, the development of an automated label detection system using YOLOv5, CustomTkinter, and OpenCV represents a significant advancement in the field of quality control for the manufacturing industry. This project not only addresses the inherent limitations of manual inspection processes but also sets a new standard for efficiency and reliability in label verification.

# **Problem Definition**

## **Problem Statement**

The primary challenge our project aims to address is the inefficiency and inaccuracy associated with the manual inspection of product labels in the manufacturing industry. Labels on products serve multiple critical purposes: they provide regulatory information, brand identification, and important usage instructions. Ensuring that these labels are correctly applied and legible is essential for compliance with regulations, maintaining brand integrity, and ensuring consumer safety. In many manufacturing settings, label inspection is still a predominantly manual process. This manual inspection process is labor-intensive, time-consuming, and prone to human error, which can lead to mislabeled products, regulatory fines, recalls, and damage to brand reputation.

In the real-world context, the accuracy and reliability of product labels are crucial. Incorrect or missing labels can result in severe consequences, including legal penalties for non-compliance with regulatory standards, customer dissatisfaction, and potential safety hazards if the labels contain usage or safety information. For example, incorrect recycling instructions could lead to environmental harm, while mislabeling food products can pose health risks to consumers. Furthermore, the manual inspection process significantly increases labor costs and reduces overall operational efficiency. In a competitive market, these inefficiencies can affect the profitability and sustainability of a business.

Our project aims to solve this problem by developing an automated label detection system using machine learning techniques. The system leverages advanced object detection models to automatically inspect and verify product labels in real-time. By automating the label inspection process, our solution seeks to eliminate the inefficiencies and inaccuracies associated with manual inspection, thereby enhancing productivity, reducing costs, and ensuring that all products meet regulatory and quality standards consistently. This automated approach not only improves operational efficiency but also ensures that the labels on products are consistently accurate and compliant, thus safeguarding consumer safety and brand reputation.

## **Background Information**

The challenge of accurately and efficiently inspecting product labels has been a persistent issue in the manufacturing industry for many years. Historically, label inspection was conducted entirely by human workers, who manually checked each product for correct labeling. This method, while straightforward, is fraught with potential for human error, particularly in high-volume production environments where workers are required to inspect hundreds or thousands of items per day. The high likelihood of fatigue and oversight makes manual inspection an unreliable method for ensuring label accuracy and compliance.

Over time, various technologies have been introduced to aid in the inspection process. Early attempts included the use of simple optical character recognition (OCR) systems and barcode scanners. While these technologies provided some improvement over manual inspection, they were limited in their capabilities and often struggled with issues such as varying label designs, poor print quality, and the presence of multiple labels on a single product. Additionally, these systems required significant manual intervention to set up and maintain, which limited their overall effectiveness and scalability.

With the advent of machine learning and artificial intelligence, new possibilities emerged for automating the label inspection process. Machine learning models, particularly those specializing in object detection and recognition, have demonstrated remarkable capabilities in accurately identifying and classifying objects within images.

YOLOv5, the latest iteration of the YOLO series, builds on this foundation by offering improved speed and accuracy over its predecessors. YOLOv5's architecture allows it to process images in a single pass, significantly reducing the time required to detect and classify objects. This makes it particularly well-suited for real-time applications such as automated label inspection, where rapid processing of high volumes of images is essential. The robustness of YOLOv5, combined with its ability to accurately detect a wide variety of objects, makes it an excellent choice for our label detection system.

In developing our system, we have drawn on these advances in machine learning and computer vision to create a solution that addresses the specific challenges of label inspection in the manufacturing industry. By using YOLOv5, we leverage its strengths in real-time object detection to accurately and efficiently inspect labels on products. Our system is trained on a diverse dataset of labeled images, ensuring that it can recognize and verify a wide range of labels with high accuracy. This training process involves the use of LabelImg, an open-source tool that allows us to annotate images with precise label information, providing the necessary data for the model to learn and generalize from.

Previous attempts to solve the problem of label inspection have laid the groundwork for our project, but many of these solutions have fallen short due to limitations in technology and implementation. Our approach, which combines the latest advancements in machine learning with a practical, user-friendly interface, represents a significant step forward in the quest to automate and optimize the label inspection process. By addressing the limitations of past solutions and incorporating cutting-edge technology, our project aims to set a new standard for efficiency and accuracy in the manufacturing industry, ultimately benefiting both producers and consumers alike.

# **Objectives**

## **Primary Objectives**

The primary objectives of our project revolve around developing a comprehensive, automated label detection system that enhances efficiency, accuracy, and reliability in the manufacturing industry. These objectives are aligned with addressing the core challenges associated with manual label inspection processes. The main expected outcomes and goals are as follows:

**1. Automate Label Detection :** The foremost objective is to fully automate the label detection process using advanced machine learning techniques. By leveraging YOLOv5, a state-of-the-art object detection model, we aim to develop a system capable of accurately identifying and verifying various labels on product packaging in real-time. This automation will significantly reduce the dependency on manual labor, thereby decreasing labor costs and mitigating human errors.

**2. Enhance Accuracy and Consistency :** The system aims to achieve high accuracy in label detection, ensuring that every label on a product is correctly identified and verified. This accuracy is crucial for maintaining regulatory compliance, consumer safety, and brand integrity. Consistency in label verification across different products and batches is also a critical outcome, as it ensures uniform quality control standards.

**3. Increase Operational Efficiency :** By automating the label inspection process, the project aims to drastically improve operational efficiency within manufacturing environments. The system will be designed to handle high volumes of products at a rapid pace, ensuring that the inspection process does not become a bottleneck in the production line. This increased efficiency will contribute to higher throughput and reduced production times.

**4. Develop a User-Friendly Interface :** Another primary objective is to create a graphical user interface (GUI) that is intuitive and easy to use. Developed using CustomTkinter and OpenCV, the GUI will enable users with minimal technical expertise to operate the system effectively. The interface will provide real-time feedback and visualization, making it easier for operators to monitor the inspection process and make necessary adjustments.

**5. Ensure Scalability and Flexibility** **:** The system will be designed to be scalable and flexible, capable of adapting to different types of products and labels. This flexibility ensures that the solution can be applied across various manufacturing settings and can accommodate future changes in labeling requirements without significant modifications.

**6. Real-Time Processing :** The system should be capable of processing images and detecting labels in real-time, allowing for immediate feedback and corrections. This real-time capability is essential for maintaining a smooth and efficient production workflow.

## **3.2. Secondary Objectives**

In addition to the primary objectives, there are several secondary objectives that the project aims to achieve. These additional goals enhance the overall functionality and impact of the system, providing further value to the manufacturing process. The secondary objectives include:

**1. Support for Multiple Label Types :** The system will be designed to support a wide variety of label types and designs, including compliance marks, branding details, recycling instructions, and more. This comprehensive support ensures that the system can be applied to different products with diverse labeling requirements.

**2. Integration with Existing Systems :** Another secondary objective is to ensure that the label detection system can be easily integrated with existing manufacturing and quality control systems. This integration will facilitate seamless data exchange and improve overall process coordination. By providing APIs and other integration tools, the system can work in conjunction with other software and hardware used in the production line.

**3. Data Logging and Reporting :** The system will include features for data logging and reporting, enabling users to track and analyze inspection results over time. This capability is essential for quality control and process improvement, as it provides insights into common issues and trends. Detailed reports can help identify areas for improvement and ensure that the inspection process remains effective and efficient.

**4. Error Handling and Alerts :** To further enhance reliability, the system will incorporate robust error handling and alert mechanisms. If the system detects a problem, such as a missing or incorrect label, it will generate an alert to notify the operator. This immediate feedback allows for quick resolution of issues, preventing defective products from reaching the market.

**5. Training and Adaptation :** The project aims to develop a system that can be easily trained and adapted to new products and labels. By providing tools and guidelines for training the model with new datasets, users can extend the system's capabilities to accommodate evolving labeling requirements. This adaptability ensures the longevity and relevance of the solution.

**6. User Training and Documentation :** Comprehensive training materials and documentation will be developed to ensure that users can effectively operate and maintain the system. This includes user manuals, training videos, and technical support resources. Proper training ensures that users can maximize the benefits of the system and address any issues that may arise.

**7. Compliance with Industry Standards :** The system will be designed to comply with relevant industry standards and regulations for label inspection and quality control. Ensuring compliance not only enhances the system's credibility but also provides assurance to users that the solution meets established benchmarks for performance and reliability.

**8. Environmental Impact Considerations :** Finally, the project aims to consider the environmental impact of the label detection system. By improving the accuracy of recycling instructions and other eco-friendly labels, the system can contribute to better environmental practices. Additionally, reducing the need for rework and waste in the production process aligns with sustainability goals.

In summary, the primary and secondary objectives of our project collectively aim to develop a robust, efficient, and user-friendly automated label detection system. By addressing the challenges of manual inspection and leveraging advanced machine learning techniques, the project seeks to transform the quality control process in the manufacturing industry, ultimately leading to improved operational efficiency, accuracy, and reliability.

# **4. Methodology**

## **4.1 Approach**

The overall approach to developing our automated label detection system involves leveraging state-of-the-art machine learning techniques, particularly YOLOv5 (You Only Look Once version 5), for real-time object detection. The strategy focuses on creating a robust, accurate, and efficient system capable of identifying and verifying various product labels in a manufacturing environment. Below is a detailed flowchart and explanation of the theoretical frameworks and models used in our approach.

**Flowchart**

1. **Data Collection**: The first step involves collecting a comprehensive dataset of images that contain the labels we aim to detect. This includes images of smartphone boxes from two brands, Samsung and OnePlus, each containing multiple specific labels.
2. **Data Annotation**: Using the LabelImg tool, each image in the dataset is annotated to mark the location and category of each label. This step is crucial for creating a training dataset that the machine learning model can use to learn label characteristics.
3. **Model Training**: The annotated dataset is used to train the YOLOv5 model. This involves feeding the labeled images into the model and iteratively adjusting the model parameters to minimize detection errors.
4. **Model Evaluation**: After training, the model is evaluated on a separate validation set to assess its accuracy and performance. Metrics such as precision, recall, and mean average precision (mAP) are used to gauge the model's effectiveness.
5. **Integration with GUI**: Once the model is trained and validated, it is integrated into a graphical user interface (GUI) developed using CustomTkinter and OpenCV. This GUI allows users to interact with the system, providing real-time label detection and verification.
6. **Testing and Validation**: The integrated system is tested in a real-world environment to ensure it meets the performance criteria and user requirements. Any necessary adjustments are made based on the test results.
7. **Deployment and Monitoring**: Finally, the system is deployed in the manufacturing setting, and its performance is continuously monitored to ensure sustained accuracy and reliability.

**Theoretical Frameworks and Models**

* **YOLOv5 Model**: YOLOv5 is a cutting-edge object detection model known for its high speed and accuracy. Unlike traditional models that use a sliding window approach to detect objects, YOLOv5 processes the entire image in a single pass, making it exceptionally fast and efficient for real-time applications.
* **Supervised Learning**: The model training process uses supervised learning, where the annotated images serve as the training data, and the model learns to map input images to the correct labels.
* **Convolutional Neural Networks (CNNs)**: YOLOv5 is based on convolutional neural networks, which are particularly effective for image recognition tasks due to their ability to capture spatial hierarchies in images.

## **Procedures**

The procedures for executing the project are detailed below, including timelines and milestones to ensure systematic progress and timely completion.

**1. Project Planning :**

* **Define Objectives**: Clearly outline the primary and secondary objectives of the project.
* **Project Scope**: Establish the project scope, including the specific labels to be detected and the expected outcomes.
* **Resource Allocation**: Identify and allocate necessary resources, including hardware, software, and personnel.

**2. Data Collection and Preparation :**

* **Image Collection**: Collect images of smartphone boxes from Samsung and OnePlus, ensuring a diverse and representative dataset.
* **Data Annotation**: Use LabelImg to annotate the collected images, marking the location and category of each label. This step is crucial for creating a high-quality training dataset.

**3. Model Training and Evaluation :**

* **Data Splitting**: Split the annotated dataset into training, validation, and test sets to ensure unbiased evaluation of the model.
* **Model Training**: Train the YOLOv5 model using the training dataset. This involves multiple iterations to fine-tune the model parameters for optimal performance.
* **Model Evaluation**: Evaluate the trained model on the validation set using metrics such as precision, recall, and mean average precision (mAP). Adjust the model as necessary to improve performance.

**4. GUI Development :**

* **GUI Design**: Design the graphical user interface using CustomTkinter and OpenCV, focusing on user-friendliness and real-time functionality.
* **Integration**: Integrate the trained YOLOv5 model into the GUI, ensuring seamless interaction between the model and the interface.
* **Testing**: Conduct initial testing of the integrated system to identify and fix any issues.

**5. System Testing and Validation :**

* **Real-World Testing**: Test the integrated system in a real-world manufacturing environment to assess its performance under actual operating conditions.
* **User Feedback**: Gather feedback from users to identify any usability issues or areas for improvement.
* **System Refinement**: Make necessary adjustments to the system based on test results and user feedback.

**6. Deployment and Monitoring :**

* **Deployment**: Deploy the finalized system in the manufacturing setting, ensuring it is fully operational and integrated with existing processes.
* **Monitoring**: Continuously monitor the system's performance to ensure sustained accuracy and reliability. Implement any necessary updates or maintenance.

**7. Documentation and Training :**

* **Documentation**: Develop comprehensive documentation, including user manuals and technical guides, to support system operation and maintenance.
* **User Training**: Conduct training sessions for users to ensure they are proficient in operating the system and addressing any potential issues.

# **Project Execution**

## **5.1 Planning and Design**

The initial planning and design phase of our project, "Automated Label Detection System," was a collaborative effort among our team. We focused on leveraging each member's strengths and knowledge to create an efficient and effective solution. This phase involved brainstorming sessions, creation of design drafts, and strategic planning to ensure a smooth execution.

* **Brainstorming Sessions:**

We began with several brainstorming sessions to discuss the scope and objectives of our project. These sessions were held to:

1. **Identify Challenges:** We discussed the current challenges in manual label inspection in the manufacturing sector, such as inefficiencies, human error, and the need for regulatory compliance.
2. **Explore Solutions:** We explored potential solutions using machine learning, specifically focusing on object detection models that could provide real-time and accurate label detection.
3. **Define Requirements:** We established the primary requirements for our system, which included high accuracy, real-time processing, user-friendly interface, and scalability.

Each team member contributed ideas based on their understanding and research, and we collectively decided on the approach to be taken. We identified YOLOv5 as the best model for our needs due to its balance of speed and accuracy.

**Design Drafts**

Following the brainstorming sessions, we created detailed design drafts to visualize our ideas and plan the project systematically. These drafts included:

**1. System Architecture:** We sketched the overall system architecture, highlighting the interaction between components such as data collection, the machine learning model, and the graphical user interface (GUI).

**2. Data Flow Diagrams:** We created data flow diagrams to map out the data processing pipeline from image capture to label detection and verification. This helped us understand the sequence of operations and the dependencies between different modules.

**3. User Interface Mockups:** We designed mockups for the GUI using tools like Figma. These mockups focused on ensuring the interface was intuitive, with clear sections for displaying real-time detection results, user controls for starting/stopping the inspection process, and indicators for pass/fail status.

**4. Technical Specifications:** We documented the technical specifications, including the hardware requirements (camera setup, computational resources) and software requirements (dependencies for YOLOv5, CustomTkinter, OpenCV, etc.). This ensured we had a clear understanding of what resources we needed and how to set them up.

**Strategic Planning:**

With the design drafts in place, we developed a strategic plan to guide the project's execution. This plan included:

* **Timeline and Milestones:** We created a detailed timeline with specific milestones for each phase of the project, such as data collection, model training, GUI development, testing, and deployment. This helped us stay on track and ensure timely progress.
* **Resource Allocation:** We assigned specific tasks to each team member based on their strengths and areas of expertise. This included roles such as data collection and annotation, model training and evaluation, GUI development, and system integration.
* **Risk Management:** We identified potential risks, such as data quality issues, model accuracy problems, and integration challenges. We discussed mitigation strategies for each risk to ensure we were prepared to handle any issues that arose.
  1. **Implementation**

The implementation phase involved executing the planned steps, developing prototypes, and refining our system based on testing and feedback. Here’s a detailed account of how we executed our project:

**1. Data Collection and Annotation:**

* **Image Collection:** We collected images of smartphone boxes from Samsung and OnePlus, ensuring a diverse dataset. We used smartphones and a high-resolution camera to capture clear images.
* **Annotation:** Using the LabelImg tool, we meticulously annotated each image to mark the location and category of each label. This step was critical for creating a high-quality training dataset for the YOLOv5 model.

**2. Model Training and Evaluation:**

* **Data Preparation:** We split our annotated dataset into training, validation, and test sets to ensure unbiased evaluation of our model.
* **Model Training:** We trained the YOLOv5 model using our annotated dataset. This involved multiple iterations to fine-tune the model parameters and optimize performance. We used Google Colab with GPU support to speed up the training process.
* **Evaluation:** After training, we evaluated the model on the validation set, using metrics like precision, recall, and mean average precision (mAP).

**3. GUI Development:**

* **Design and Development:** We developed the GUI using CustomTkinter and OpenCV. The design focused on ease of use, with clear sections for real-time label detection results, user controls, and pass/fail indicators.
* **Integration:** We integrated the trained YOLOv5 model into the GUI, ensuring seamless interaction between the model and the interface. This required careful coding to ensure real-time processing and accurate display of results.

**4. Testing and Refinement:**

* **Real-World Testing:** We tested the system in a simulated manufacturing environment to assess its performance under actual operating conditions. This helped us identify any practical issues and make necessary adjustments.
* **User Feedback:** We gathered feedback from potential users, such as classmates and instructors, to identify any usability issues or areas for improvement. Based on this feedback, we refined the system to enhance user experience and functionality.

**5. Final Deployment and Monitoring:**

* **Deployment:** Once we were satisfied with the system's performance, we deployed it in a simulated real-world setting. This involved setting up the camera and ensuring the GUI was fully operational.
* **Monitoring:** We continuously monitored the system's performance to ensure it maintained accuracy and reliability. We also documented any issues that arose and made further refinements as needed.

# **Tools and Techniques Used**

## **6.1 Tools**

In our project, we employed a range of tools, software, and hardware to achieve our objectives effectively. Each tool was selected based on its relevance and functionality within the context of our project goals.

**1. YOLOv5:**

* **Purpose:** YOLOv5 (You Only Look Once version 5) is a state-of-the-art object detection model used to detect and classify objects in images.
* **Description:** YOLOv5 was pivotal for our project as it provided the necessary speed and accuracy for real-time label detection. We trained the model on a custom dataset of labeled images to recognize various labels on smartphone boxes.

**2. LabelImg:**

* **Purpose:** LabelImg is an open-source graphical image annotation tool.
* **Description:** We used LabelImg to annotate our dataset. By marking the bounding boxes around labels in our images, we created the training data required for YOLOv5. This tool was essential for preparing our dataset accurately and efficiently.

**3. CustomTkinter:**

* **Purpose:** CustomTkinter is a modern, customizable version of the Tkinter library used for creating graphical user interfaces in Python.
* **Description:** We used CustomTkinter to develop a user-friendly GUI for our system. It allowed us to create a visually appealing interface where users could view real-time label detection results and interact with the system easily.

**4. OpenCV:**

* **Purpose:** OpenCV (Open Source Computer Vision Library) is a library of programming functions primarily aimed at real-time computer vision.
* **Description:** OpenCV was used for image processing and handling video streams from the camera. It enabled us to capture and process frames in real-time, which were then passed to the YOLOv5 model for label detection.

**5. PIL (Python Imaging Library):**

* **Purpose:** PIL is a library in Python for opening, manipulating, and saving various image file formats.
* **Description:** We used PIL to handle image conversions and display images in the GUI. It facilitated the seamless integration of images within the CustomTkinter interface.

**6. Google Colab:**

* **Purpose:** Google Colab is a cloud-based service that allows the execution of Python code using Google’s hardware, including GPUs.
* **Description:** We utilized Google Colab for training our YOLOv5 model. The availability of free GPU resources significantly sped up the training process, allowing us to experiment with different model configurations and optimize performance.

**7. Camera Setup:**

* **Purpose:** A high-resolution camera was used to capture images and video streams of smartphone boxes.
* **Description:** The camera was essential for real-time label detection, providing the input data needed by our system. We used a high-resolution camera to ensure the clarity and detail required for accurate label recognition.

## **Techniques**

The successful execution of our project required the application of various techniques and methods. These techniques were chosen based on their effectiveness in addressing our project's specific requirements and challenges.

**1. Data Annotation and Preparation:**

* **Technique:** Manual annotation of images using LabelImg.
* **Application:** We meticulously annotated each image to create a high-quality dataset. This process involved marking the location and category of each label on the smartphone boxes. Accurate annotation was crucial for training the YOLOv5 model to recognize labels correctly.

**2. Model Training and Fine-Tuning:**

* **Technique:** Training the YOLOv5 model on a custom dataset.
* **Application:** Using the annotated dataset, we trained the YOLOv5 model to detect and classify labels. The training involved multiple iterations to fine-tune the model parameters and optimize performance. We evaluated the model using metrics like
* precision, recall, and mean average precision (mAP) to ensure it met our accuracy requirements.

**3. Real-Time Image Processing:**

* **Technique:** Processing video frames using OpenCV.
* **Application:** OpenCV was used to capture and process video frames from the camera in real-time. This involved converting frames to the appropriate color format, resizing them, and feeding them into the YOLOv5 model for label detection. The processed frames were then displayed in the GUI, providing real-time feedback to users.

**4. GUI Development:**

* **Technique:** Creating an interactive interface using CustomTkinter.
* **Application:** We designed and developed the GUI to be user-friendly and intuitive. The interface included real-time detection results, user controls for starting and stopping the inspection process, and visual indicators for pass/fail status. The GUI was essential for making the system accessible to users with minimal technical expertise.

**5. Model Integration:**

* **Technique:** Integrating the trained YOLOv5 model into the GUI.
* **Application:** We ensured seamless interaction between the YOLOv5 model and the CustomTkinter interface. This involved writing code to load the model, process video frames, and update the GUI with detection results in real-time. The integration was critical for achieving the desired functionality and performance of our system.

**6. Testing and Evaluation:**

* **Technique:** Systematic testing and evaluation of the system.
* **Application:** We conducted extensive testing to evaluate the system’s performance under various conditions. This included real-world testing in a simulated manufacturing environment, gathering user feedback, and refining the system based on the findings. Testing was vital for identifying and fixing any issues, ensuring the system met all performance and usability requirements.

By carefully selecting and applying these tools and techniques, we were able to develop a robust and effective automated label detection system. Each component played a crucial role in achieving our project goals, from data preparation and model training to real-time processing and user interface development. The combination of advanced machine learning techniques and practical software solutions enabled us to create a valuable tool for improving efficiency and accuracy in label inspection processes.

# **Partial Results**

## **7.1 Initial Findings**

In the initial stages of our project, the focus was primarily on developing and training the YOLOv5 model to accurately detect labels on phone boxes. We used the annotated images collected through LabelImg to train the model. After several iterations of training and fine-tuning, the model achieved a satisfactory level of accuracy. When tested on video feeds, the model successfully detected and identified labels on the phone boxes, displaying bounding boxes around the detected labels.

The video output demonstrated that the model could process frames in real-time, correctly identifying and highlighting the labels on the boxes. However, this initial setup had its limitations. The output was confined to a video stream, making it difficult to interact with or manipulate the detection process in real-time. Users could only observe the detection results without any means to control or customize the detection parameters based on their specific needs.

## **Improvements with GUI Integration**

Recognizing the need for a more interactive and user-friendly system, we integrated the trained YOLOv5 model with a custom GUI built using CustomTkinter. This integration brought several significant improvements to the project:

**1. User Interaction:**

The CustomTkinter-based GUI provided an intuitive platform for users to interact with the label detection system. Users could easily select the brand of the phone box (Samsung or OnePlus) from a dropdown menu, which dynamically adjusted the labels to be detected based on the selected brand. This flexibility allowed for a more targeted and efficient label inspection process.

**2. Real-Time Feedback:**

The GUI displayed real-time detection results, providing instant feedback to the users. Detected labels were shown with visual indicators: a checkmark for present labels and a cross for missing labels. This immediate feedback enabled users to quickly verify the presence of all required labels and make necessary adjustments if any labels were missing.

**3. Enhanced Usability:**

The integration of the GUI significantly enhanced the overall usability of the label detection system. The user-friendly interface made it easier for users to operate the system without needing extensive technical knowledge. The clear and organized layout of the GUI ensured that users could navigate through the system effortlessly, making the label inspection process more efficient and less time-consuming.

**4. Additional Features:**

The GUI included a comprehensive summary feature that provided a "PASS" or "FAIL" indicator based on whether all required labels were detected. This feature allowed users to quickly assess the overall inspection results and take appropriate actions as needed.

**5. Scalability:**

The integration also laid the groundwork for future enhancements, such as adding more brands or labels, improving detection algorithms, and incorporating additional functionalities.

Overall, the integration of the GUI with the YOLOv5 model transformed our project from a basic label detection system into a sophisticated and user-friendly tool. The improvements in user interaction, real-time feedback, enhanced usability, and additional features significantly increased the system's practical applications, making it a valuable asset for label inspection tasks.