

KLE Society's
KLE Technological University



An Industry Training Report

On

**Machine Learning-Based Real-Time Conveyor Inspection and
Performance Improvement for Steel Industries**

Submitted in partial fulfillment of the requirement for the degree of

**Bachelor of Engineering in
Computer Science and Engineering**

Submitted By

**Shreyas Joshi
01fe20bcs145**

**Under the guidance
of**

Mrs. Pratibha R Malagatti

**SCHOOL OF COMPUTER SCIENCE &ENGINEERING,
HUBBLLI-580 031 (India).**

Academic year 2023-24



**B. V. Bhoomaraddi College Campus, Vidyanagar, Hubballi - 580031.
Karnataka (India)**

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that Industry Training entitled “**Machine Learning-Based Real-Time Conveyor Inspection and Performance Improvement for Steel Industries**” is a bonafide work carried out by the student **Mr. Shreyas Joshi** bearing USN **01fe20bcs145** in partial fulfillment of the completion of 8th semester B. E. course during the year 2023 – 24 at **DocketRun Tech Private Limited**. The Industry Training report has been approved as it satisfies the academic requirement with respect to the Training work prescribed for the above said course.

Name of the Guide
Mrs. Pratibha R Malagatti

Head of SoCSE
Dr. Vijayalakshmi M.

Name of the examiners

1 -----
2 -----

Signature with date

1 -----
2 -----



DOCKETRUN TECH PRIVATE LIMITED

1st Floor, R.H. Kulkarni Building, KLE Tech University, Vidyanagar Hubballi Karnataka – 580031

CIN: U72900KA2019PTC128107

GST: 29AAHCD4356L1Z6

PAN: AAHCD4356L

Mob: +91-9449034387

Email: info@docketrun.com

Website: www.docketrun.com

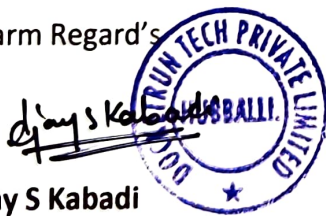
INTERNSHIP COMPLETION CERTIFICATE

We are delighted to certify that **Mr. Shreyas Joshi**, a student of Computer Science Engineering, **BVB College, KLE Tech University, Vidyanagar Hubballi**, has successfully completed the internship program (08/01/2024 – 31/05/2023) at **DocketRun Tech Pvt Ltd**.

During the internship, he played a pivotal role in multiple projects, focusing on AI development & completed numerous assignments, showcasing adaptability and excellence. Notably, he was a key contributor to the "**Machine Learning-Based Real-Time Conveyor Inspection and Performance Improvement for Steel Industries**" project.

We would like to express our sincere gratitude for your exemplary performance and wish you continued success in all your future endeavors.

Warm Regard's



Ajay S Kabadi

Founder, C.E.O

DocketRun Tech Pvt Ltd.

DECLARATION

I hereby declare that the Industry Training Report entitled **“Machine Learning Based Real-Time Conveyor Inspection and Performance Improvement for Steel Industries”** is an authentic record of my own work as requirements of Industry Training, during the period from 08-01-2024 to 31-05-2024 for the award of the degree of B.E. Under the guidance of Mrs. Pratibha R Malagatti.

(Signature of student)

Shreyas Joshi
01fe20bcs145

DATE:

Acknowledgement

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of a number of individuals whose professional guidance and encouragement helped me in the successful completion of this report work.

I also take this opportunity to thank Dr. Vijaylakshmi M, Professor and Head, School of Computer Science and Engineering for having provided us academic environment which nurtured our practical skills contributing to the success of our project.

I sincerely thank our guide Mrs. Pratibha R Malagatti, School of Computer Science and Engineering for her guidance and wholehearted co-operation during the course of completion.

I sincerely thank Mr. Huchhareddi, Docketrun for his support, inspiration and wholehearted co-operation during the course of completion.

My gratitude will not be complete without thanking our beloved parents, our seniors and our friends who have been a constant source of aspirations.

Shreyas Joshi

About Company

DocketRun is an AI-powered video analytics platform that enables organizations across industries to streamline operations, ensure safety and security at the workplace, enhance customer experiences, leverage data to drive decisions, and thus supercharge their businesses. Headquartered in Deshpande Startups (India's Largest Incubation Centre), Hubballi, Karnataka, and backed by the mentors like Santosh Huralikoppi and CM Patil, DocketRun has customers across segments like e-commerce, retail, manufacturing plants, educational institutes, and so on, with customers like Maruti Suzuki, Tata Hitachi, Fabwoods, KLE University, Deshpande Educational Trusts, Alie Global, among several others. With the organizations across industries shifting from analog-heavy to intelligent IP video surveillance products and solutions, DocketRun is well-positioned to take advantage of this shift.

DocketRun offers a comprehensive product and software solution called "DocketRun" which is India's first real-time advanced AI system for monitoring and control of safety protocols, standard operating procedures (SOPs), and inspection processes. The solution can be seamlessly integrated with any existing CCTV camera infrastructure at plant or facility locations, making it a plug-and-play offering. DocketRun's core offerings include: a) Safety Eye: For monitoring employee safety and adherence to safety protocols b) Electrical Eye: For monitoring electrical panels, including 6.6 KV and 33 KV panels c) Mech Eye: For monitoring mechanical operations and related SOPs d) Other customizable solutions for industrial SOPs and product quality inspections The system is designed to trigger various alert mechanisms in case of violations, such as violation snapshots, hooter systems, relay systems, and voice announcement systems. DocketRun's solutions span multiple use cases, including personal protective equipment (PPE) monitoring, fire detection, traffic counting, man-machine interface monitoring, security and surveillance, and smart wearable system integration. The platform supports integration with any CCTV camera brand, allowing flexibility in deployment across diverse industrial environments. DocketRun's AI-powered video analytics platform enables organizations to streamline operations, enhance workplace safety and security, optimize customer experiences, and drive data-driven decision-making, ultimately leading to business growth and efficiency. The Electrical Panels SOP Monitoring solution features real-time video monitoring, voice-enabled alerts and notifications, live CCTV camera grid analytics, and complete job history with image proofs. DocketRun boasts a unique selling proposition as the world's first video analytics platform to analyze the complete process of electrical and mechanical jobs in steel plants, with the ability to process from 10 to 10,000+ cameras without lag. We also offers real-time IP65 graded feedback systems to control violations across multiple cameras, along with hooter systems, relay systems, and voice announcement systems to

announce violations in any regional language with custom messages.

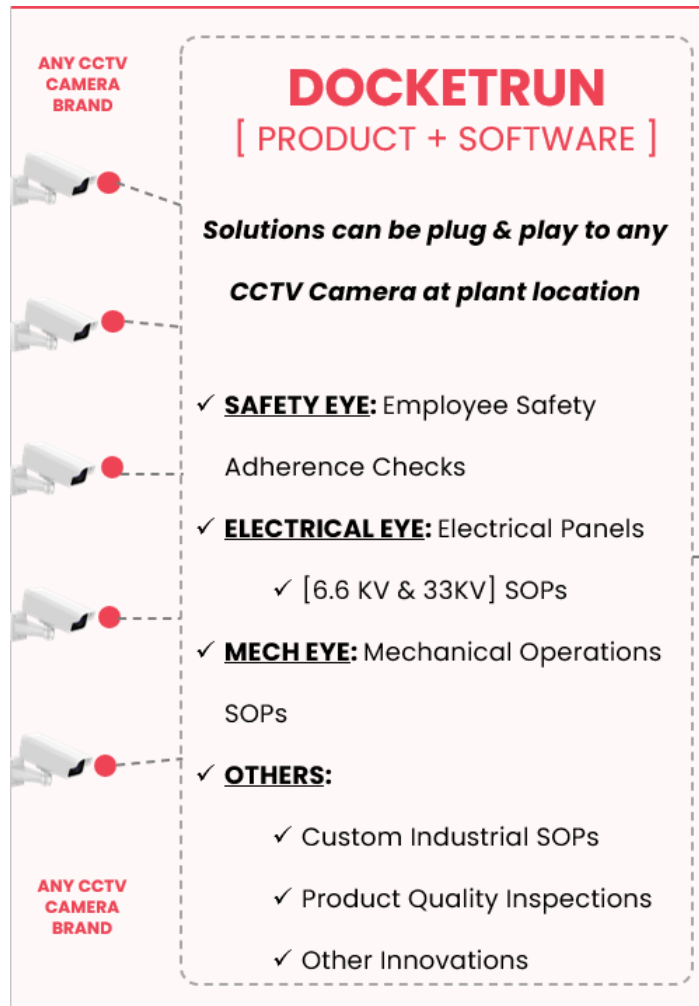


Figure 1: DocketRun product details

CONTENTS

Acknowledgement	i
About Company	ii
CONTENTS	v
LIST OF FIGURES	vi
1 INTRODUCTION	1
1.1 Video Analytics for Industrial Application	1
1.2 Introduction to AI/ML	2
1.2.1 Supervised Learning	2
1.2.2 Unsupervised Learning	3
1.2.3 Reinforcement Learning	4
1.2.4 Semi-Supervised Learning	4
1.3 Computer Vision	5
1.3.1 Image Classification	5
1.3.2 Image Detection	6
1.3.3 Image Segmentation	7
1.4 Role of ML Intern	9
2 Tools and Technology Used	10
2.1 Data Collection and Preprocessing	10
2.1.1 Data acquisition	11
2.1.2 Data cleaning and preprocessing	11
2.2 Image Processing	11
2.2.1 Image Filtering	11
2.2.2 Image Transformation	12
2.2.3 Color Space Conversion	12
2.2.4 Image Enhancement	13
2.3 Machine Learning Models	13
2.3.1 Visual Geometry Group (VGG)	13
2.3.2 Inception V3	14
2.3.3 Residual Networks (ResNet)	14
2.3.4 You Only Look Once (YOLO)	15

2.4	Deployment and Integration	16
2.4.1	Oracle VirtualBox	16
2.4.2	LabelImg	16
2.4.3	Tensorflow	17
2.4.4	PyTorch	17
2.4.5	React	18
2.4.6	PostgreSQL	18
2.4.7	Tkinter	18
2.4.8	OpenCV	19
3	Snapshots	20
3.1	Module implementation	20
3.2	Training metric results	22
3.2.1	Loss Graphs	23
3.2.2	Metric Graphs	24
4	Results and Discussions	25
5	Conclusion and Future Scope	26

LIST OF FIGURES

1	DocketRun product details	iii
1.1	Machine learning classification	2
1.2	Segmentation classification	8
2.1	Virtual machine architecture	16
2.2	Application of opencv	19
3.1	Block of code to create database table	20
3.2	Block of code to connect to database	21
3.3	Block of code for execution of main process	22
3.4	Training metrics results	23

Chapter 1

INTRODUCTION

1.1 Video Analytics for Industrial Application

Video analytics has emerged as a transformative technology, revolutionizing various industries by providing actionable insights from video data. It involves the automatic interpretation of video streams to detect and determine temporal and spatial events, finding applications in security and surveillance, retail analytics, healthcare, and more. This technology is instrumental in bolstering security and safety through its real-time processing capabilities of data from cameras, swiftly detecting, tracking, and responding to various events and anomalies. It can also analyze behavior patterns, such as loitering or erratic movements, and trigger alerts accordingly. In crowded environments like airports or stadiums, video analytics aids in crowd management by monitoring density and detecting potential safety hazards, contributing to traffic monitoring by detecting violations and improving traffic flow, and providing crucial situational awareness during emergency situations. One of its key components is machine learning, particularly deep learning, which plays a vital role in detecting and recognizing objects, faces, and activities in videos. Artificial intelligence (AI) algorithms are used to analyze video content, extract meaningful information, and make intelligent decisions based on the analysis. Tools and technologies like OpenCV and YOLO (You Only Look Once) are instrumental in video analytics, with OpenCV providing various functions for image and video processing, and YOLO being a state-of-the-art, real-time object detection system that can detect multiple objects in an image or video stream. Segmentation, another computer vision technique involving dividing an image into segments, simplifies its representation or locates objects and boundaries, used in medical imaging, satellite image analysis, and autonomous driving, among other applications.

During my internship, I had the opportunity to delve deep into these tools and technologies, gaining hands-on experience in their implementation. I explored the intricacies of machine learning and AI, understanding how these technologies are leveraged to analyze video data and extract meaningful insights. Additionally, I experimented with different object detection models, gaining valuable insights into their strengths and limitations. This experience not only enhanced my technical skills but also broadened my perspective on the possibilities and challenges in the field of video analytics.

1.2 Introduction to AI/ML

Artificial Intelligence (AI) and Machine Learning (ML) have become integral components of modern technology, revolutionizing industries across the globe. At its core, AI simulates human intelligence processes using machines, enabling them to learn from data, adapt to new inputs, and perform tasks autonomously. Machine Learning, a subset of AI, focuses on developing algorithms that allow computers to learn from and make predictions or decisions based on data.

AI and ML have permeated various sectors, from healthcare and finance to manufacturing and transportation, reshaping the way businesses operate and adding unprecedented efficiency and accuracy to processes. Understanding the fundamentals of AI and ML is essential for grasping their potential applications and unlocking their benefits. Types of Machine Learning are:

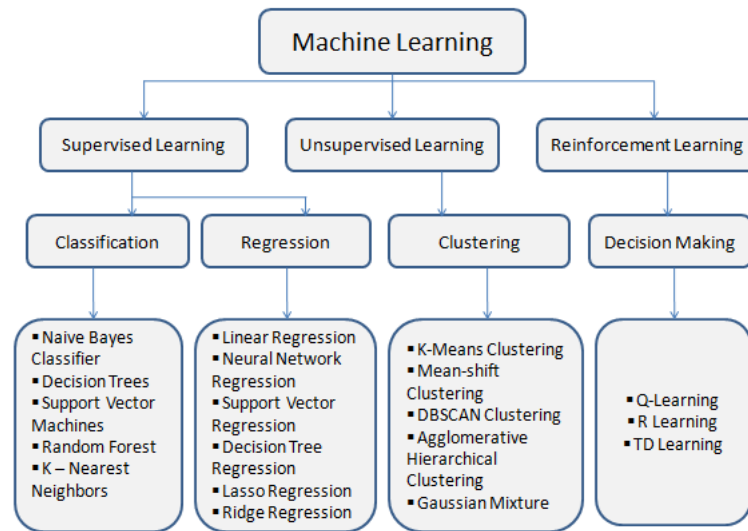


Figure 1.1: Machine learning classification

1.2.1 Supervised Learning

Supervised learning involves training a model on a labeled dataset, where each input is paired with the corresponding output. The model learns to map inputs to outputs, making predictions on new data. An example of supervised learning is image recognition, where the model learns to classify images into predefined categories based on labeled training data. Real-world application: Email spam filtering, where the model learns to classify emails as spam or non-spam based on labeled examples.

Supervised learning is the cornerstone of many computer vision applications, relying on labeled datasets where each image is paired with the correct output label.

- **Image Classification:** Model learns to categorize images into predefined classes, like identifying whether an image contains a cat or a dog. Object detection goes a step further by not only recognizing the objects but also localizing them within the image, typically by drawing bounding boxes around them.
- **Object Detection:** Identifies and localizes objects within images. This means the system not only recognizes which objects are present but also pinpoints where each object is situated by drawing bounding boxes around them. This dual capability is essential for understanding the context and dynamics of the scene in various applications.
- **Semantic Segmentation:** Classifying each pixel in an image into a category, which is crucial for understanding complex scenes, such as those in autonomous driving where different parts of the road scene need to be identified. Instance segmentation combines these techniques to distinguish between multiple objects of the same category within an image, identifying each instance separately.
- **Models:** Convolutional Neural Networks (CNNs) are particularly effective for these tasks, as they can capture and hierarchically process spatial features in images, making them ideal for various supervised learning applications in computer vision.

1.2.2 Unsupervised Learning

In unsupervised learning, the model is presented with unlabeled data and must find structure or patterns within it. Unlike supervised learning, there are no predefined outcomes, and the algorithm explores the data to discover hidden insights. Clustering algorithms, such as K-means, are a common example of unsupervised learning. Real-world application: Market segmentation in marketing, where unsupervised learning algorithms group customers based on similarities in behavior or preferences.

Unsupervised learning in computer vision involves exploring and identifying patterns in data without the use of labeled examples.

- **Clustering:** Grouping similar images together based on their visual features, which can be useful for tasks like organizing large photo collections or discovering new categories within a dataset.
- **Dimensionality Reduction:** Reducing the number of random variables under consideration (e.g., using techniques like Principal Component Analysis (PCA) or t-SNE to visualize high-dimensional image data). It reduces the complexity of image data, making it easier to visualize and interpret high-dimensional data.

- **Anomaly detection:** The system identifies images or parts of images that deviate from the normal patterns observed in the majority of the dataset, such as detecting defects in manufacturing processes.
- **Generative models:** Models like Generative Adversarial Networks (GANs) and Autoencoders are popular in unsupervised learning, as they can generate new images and learn efficient representations of the input data, respectively.

1.2.3 Reinforcement Learning

Reinforcement learning involves training an agent to interact with an environment in order to maximize cumulative rewards. The agent learns through trial and error, receiving feedback in the form of rewards or penalties based on its actions. Reinforcement learning powers autonomous systems and game-playing algorithms. Real-world application: Self-driving cars, where reinforcement learning algorithms enable vehicles to learn optimal driving strategies through experience in various traffic scenarios.

Reinforcement learning (RL) in computer vision involves training an agent to make a sequence of decisions based on visual inputs by rewarding desired actions and punishing undesired ones. This approach is particularly useful in dynamic and interactive environments where decision-making is crucial.

- **Robotics and Control:** Using visual input to perform tasks like navigation, manipulation, and interaction with objects (e.g., a robot learning to pick up objects using visual feedback).
- **Game Playing:** Training agents to play games using visual information from the game environment (e.g., agents learning to play video games like Atari based on screen pixels).
- **Autonomous Driving:** Enabling vehicles to make driving decisions based on visual data from cameras (e.g., lane following and obstacle avoidance).

RL models learn from the environment through exploration and exploitation, and techniques like Deep Q-Networks (DQNs) and Proximal Policy Optimization (PPO) are popular in this domain.

1.2.4 Semi-Supervised Learning

Semi-supervised learning combines elements of supervised and unsupervised learning, leveraging both labeled and unlabeled data. This approach is particularly useful when labeled data is scarce or expensive to obtain. Semi-supervised learning algorithms can extract additional information from unlabeled data to improve model performance. Real-world application: Speech

recognition, where a model trained on a small set of labeled speech samples can leverage a larger pool of unlabeled data to enhance its accuracy.

1.3 Computer Vision

Computer vision is a multidisciplinary field of artificial intelligence (AI) that enables computers and systems to interpret and understand the visual world. By using digital images from cameras and videos, along with deep learning models, machines can accurately identify and classify objects, detect events, and make decisions based on visual inputs. This technology plays a crucial role in enhancing safety systems across multiple sectors by enabling real-time monitoring, detection, and response to potential hazards. In the automotive industry, it powers advanced driver-assistance systems (ADAS) and autonomous vehicles, helping to prevent accidents by detecting pedestrians, recognizing traffic signals, and monitoring driver behavior. In industrial settings, computer vision systems can identify unsafe conditions, such as malfunctioning equipment or workers without proper safety gear, thereby reducing workplace accidents. Additionally, in public security, computer vision aids in identifying suspicious activities and improving crowd management during large events. By providing a layer of intelligent, automated oversight, computer vision significantly enhances the effectiveness and reliability of safety systems.

1.3.1 Image Classification

Image classification is a fundamental task in machine learning and computer vision, where the goal is to assign a label or category to an input image. This task involves analyzing the visual content of the image and classifying it into predefined categories. For example, an image classification model can be trained to recognize and classify images of different animals, such as cats, dogs, and birds. The process typically begins with the acquisition and preprocessing of image data, followed by the application of machine learning algorithms to learn patterns and features from the data. The trained model can then be used to predict the class of new, unseen images with a certain level of accuracy.

The development of image classification models has been significantly advanced by the introduction of deep learning techniques, particularly Convolutional Neural Networks (CNNs). CNNs are designed to automatically and adaptively learn spatial hierarchies of features from input images. They consist of multiple layers, including convolutional layers, pooling layers, and fully connected layers, each playing a role in feature extraction and representation. Early layers typically learn low-level features such as edges and textures, while deeper layers capture high-level features and more abstract concepts. Models like AlexNet, VGG, and ResNet have

set benchmarks in image classification tasks, achieving remarkable accuracy on large datasets like ImageNet.

Despite the successes, image classification faces several challenges. One of the main challenges is the need for large labeled datasets to train robust and accurate models. Collecting and annotating such datasets can be time-consuming and expensive. Additionally, image classification models need to be robust to variations in lighting, viewpoint, occlusion, and other real-world conditions. Overfitting is another common issue, where the model performs well on the training data but fails to generalize to new data. To address these challenges, techniques such as data augmentation, regularization, and transfer learning are employed. Data augmentation artificially increases the size and diversity of the training dataset through transformations like rotation, scaling, and flipping. Transfer learning leverages pre-trained models on large datasets and fine-tunes them for specific tasks, reducing the need for extensive labeled data. These methods help improve the performance and generalization capabilities of image classification models, making them more applicable to real-world scenarios.

1.3.2 Image Detection

Image detection is a fundamental task in computer vision that involves identifying and locating objects within an image. It is crucial for various applications, including autonomous driving, surveillance, and healthcare. The process begins with preprocessing steps like resizing, normalization, and augmentation to enhance the quality and variability of the input data. Traditional techniques for image detection, such as Haar cascades and Histogram of Oriented Gradients (HOG), rely on handcrafted features and statistical models to detect objects. These methods, while effective in controlled environments, often struggle with variations in scale, lighting, and occlusion present in real-world scenarios.

In recent years, the advent of deep learning has revolutionized image detection. Convolutional Neural Networks (CNNs) have become the backbone of modern detection algorithms due to their ability to learn hierarchical features directly from raw pixel data. State-of-the-art models like YOLO (You Only Look Once), SSD (Single Shot Multibox Detector), and Faster R-CNN have significantly improved detection accuracy and speed. YOLO, for instance, divides the image into a grid and predicts bounding boxes and class probabilities simultaneously, achieving real-time performance. These models are trained on large annotated datasets like COCO or Pascal VOC, enabling them to generalize well to diverse object categories and complex scenes.

Despite these advancements, image detection still faces several challenges. Detecting small objects, handling dense scenes with multiple overlapping objects, and ensuring robustness

under varying environmental conditions are areas of ongoing research. Additionally, the computational requirements for training and deploying deep learning models can be prohibitive, necessitating the development of more efficient algorithms and hardware accelerators. Future directions in image detection may involve integrating contextual information, improving transfer learning techniques, and leveraging unsupervised and semi-supervised learning to reduce dependency on large labeled datasets.

1.3.3 Image Segmentation

Image segmentation is a critical process in computer vision that involves partitioning an image into distinct regions or segments, each representing a different object or part of an object. This technique is essential for various applications such as medical imaging, where precise segmentation of anatomical structures like organs or tumors is necessary for accurate diagnosis and treatment planning. Unlike image detection and recognition, which focus on identifying and classifying objects, segmentation provides a pixel-wise classification, offering a more granular understanding of the image content. There are different types of segmentation, including semantic segmentation, which labels each pixel with a class label, and instance segmentation, which not only identifies the classes but also distinguishes between individual instances of objects.

Traditional methods of image segmentation relied heavily on techniques like thresholding, edge detection, and clustering. Thresholding involves separating pixels based on intensity values, while edge detection methods like the Canny edge detector identify boundaries by detecting discontinuities in gradients. Clustering algorithms, such as k-means, group pixels into clusters based on their features. However, these approaches often struggled with complex images containing varying textures, colors, and lighting conditions. The advent of machine learning, and particularly deep learning, has significantly advanced the field of image segmentation. Convolutional Neural Networks (CNNs) have enabled more accurate and efficient segmentation through architectures specifically designed for this purpose, such as Fully Convolutional Networks (FCNs), U-Net, and Mask R-CNN.

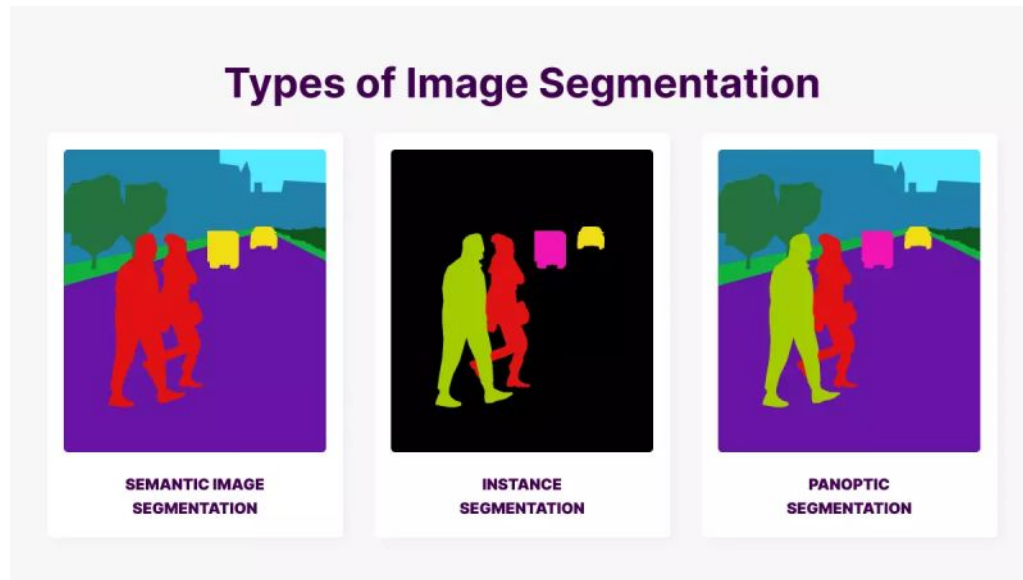


Figure 1.2: Segmentation classification

- **Semantic Segmentation** Semantic segmentation assigns a class label to each pixel in an image, ensuring that pixels belonging to the same class (e.g., "cat", "dog", "tree") are grouped together. However, it does not differentiate between individual instances of the same class. For example, in a scene with multiple cats, semantic segmentation will label all cat pixels with the same class, without distinguishing between different cats. This type of segmentation is widely used in applications like autonomous driving, where identifying the presence and boundaries of different objects in the environment is crucial.
- **Instance Segmentation** Instance segmentation extends semantic segmentation by not only classifying each pixel but also distinguishing between different instances of the same object class. This means that in an image with multiple cats, each cat will be segmented as a separate instance. Techniques like Mask R-CNN are commonly used for instance segmentation. This type of segmentation is particularly useful in scenarios where precise object counting and differentiation are required, such as in crowd analysis or inventory management.
- **Panoptic Segmentation** Panoptic segmentation is a relatively recent approach that combines the strengths of both semantic and instance segmentation. It provides a comprehensive scene understanding by generating a unified output that includes both the semantic segmentation of background regions (like sky, road, and buildings) and the instance segmentation of foreground objects (like people, cars, and animals). This holistic approach is beneficial in applications like robotics and augmented reality, where a complete understanding of both the scene and individual objects is necessary.

1.4 Role of ML Intern

As an ML intern tasked with creating safety detection software, my role involves developing an end-to-end system that ensures comprehensive safety monitoring and timely alerts. This includes designing and training machine learning models to accurately detect potential safety hazards or incidents from video footage or images. Using state-of-the-art object detection techniques, I will identify and localize safety concerns, such as unauthorized access, equipment malfunctions, or hazardous conditions. Once detected, the system will trigger automated alerts through integrated email and WhatsApp notifications, ensuring that relevant personnel are informed immediately. My responsibilities also encompass the deployment and maintenance of the software, ensuring it operates seamlessly in real-time environments. This involves setting up the necessary infrastructure, optimizing the system for performance, and conducting thorough testing to ensure reliability and accuracy.

Chapter 2

Tools and Technology Used

During my internship, I had the opportunity to work with a variety of tools and technologies that are integral to the field of artificial intelligence and video analytics. One of the primary tools I worked with was OpenCV, an open-source computer vision and machine learning software library. OpenCV provided me with a rich set of functions for image and video processing, allowing me to manipulate and analyze video data effectively.

Another key technology I worked with was YOLO (You Only Look Once), a state-of-the-art, real-time object detection system. YOLO enabled me to detect multiple objects in video streams with high accuracy and speed, making it ideal for applications requiring real-time processing of video data.

In addition to OpenCV and YOLO, I also gained experience with other machine learning and deep learning frameworks, such as TensorFlow and PyTorch. These frameworks are widely used for developing and training machine learning models, and I used them extensively to experiment with different object detection models and algorithms.

2.1 Data Collection and Preprocessing

As ML system interns, gathering and preparing data is a crucial part of our work. We used a number of techniques intended to record different aspects of the steel manufacturing process in order to gather a variety of data. We used IoT devices with sensors strategically positioned throughout the manufacturing facility in addition to CCTV cameras in the RTSP format. These sensors recorded vibration, humidity, pressure, and other pertinent factors in addition to temperature fluctuations. Moreover, we incorporated data logging systems straight into the conveyor system, which allowed us to obtain operating metrics instantly. In order to provide a comprehensive understanding of the production lifecycle, we also collaborated with already-existing plant systems to gain access to historical data archives. We created a solid dataset with both historical and real-time insights by utilising this variety of data collection techniques. .

2.1.1 Data acquisition

The acquisition of numerous data sources was a critical component of our project's success during the data collecting and preparation phase. First and foremost, we collected visual data via Real-Time Streaming Protocol (RTSP)-compliant CCTV cameras. We were able to obtain a thorough grasp of the steel production process by using this technology, which gave us real-time access to the conveyor machines. By adding temperature information gathered from sensors inside the sinter plant, we also expanded our knowledge. Our understanding of these diverse data kinds has been enhanced, and the necessity of adaptability when managing different data sources in machine learning applications has been highlighted by our experience navigating and integrating them.

2.1.2 Data cleaning and preprocessing

To guarantee the accuracy and consistency of our datasets, we have learned to use strong data cleaning and preprocessing procedures. We learned techniques like imputation and value deletion from the dataset to deal with missing data. Outlier detection and removal is another crucial procedure that entails locating and adjusting data points that substantially depart from the norm in order to reduce their possible influence on model performance. Standardisation and normalisation approaches are essential for preventing any one characteristic from controlling the learning process of the model. These strategies involve scaling features to a constant range. We also learned about feature engineering, which is essential to preprocessing and involves adding new features or altering current ones to improve the prediction ability of the model. During this stage, methods like scaling numerical features or one-hot encoding for categorical variables are frequently used.

2.2 Image Processing

Image processing techniques encompass a wide range of methods and algorithms used to analyze, manipulate, enhance, or interpret digital images. These techniques include filtering, transformations, color space conversion, feature extraction, segmentation, enhancement, restoration, compression, morphological operations, and feature matching. The ability to extract important information from visual input for additional analysis and decision-making renders them essential for computer vision.

2.2.1 Image Filtering

Image filtering techniques are essential tools in image processing used to extract specific features from images. The different filters used in our project are: the blur filter, which is used

to reduce noise or detail in an image. Gaussian blur, for example, applies a weighted average to each pixel's neighborhood, resulting in a smooth, blurred effect. Median blur replaces each pixel's value with the median value of its neighborhood, effectively removing outliers and preserving edges. Bilateral blur is a more sophisticated technique that preserves edges while reducing noise by considering both spatial and intensity differences between pixels.

Conversely, sharpening filters enhance edges and detail in an image, making it appear more defined and crisp. These filters work by accentuating high-frequency components in the image, effectively increasing contrast along edges.

Edge detection filters, such as Sobel, Prewitt, and Canny, are used to identify edges or boundaries within an image. Sobel and Prewitt operators compute the gradient magnitude of the image, highlighting regions of significant intensity changes. The Canny edge detection algorithm is more advanced and robust, utilizing multiple stages including noise reduction, gradient computation, non-maximum suppression, and edge tracking by hysteresis to accurately detect edges while suppressing noise and false positives.

2.2.2 Image Transformation

Image transformation encompasses a range of techniques used to manipulate images to achieve desired effects or prepare them for specific tasks. Resizing is a common transformation that alters the dimensions of an image, either scaling it up or down while preserving its aspect ratio. This is useful for adjusting the size of images to fit within certain constraints or to match the dimensions of other images in a dataset. Rotation involves rotating the image by a specified angle, allowing for adjustments to orientation or alignment. Cropping is another transformative process that involves removing unwanted parts of an image, focusing only on the regions of interest. This can help improve composition, remove distractions, or extract specific features. Warping is a more complex transformation that distorts the image to correct perspective distortions or align features. It is commonly used in tasks such as image registration, where multiple images need to be aligned spatially. Collectively, these image transformation techniques provide valuable tools for preprocessing images and manipulating their appearance or structure to suit various applications in computer vision.

2.2.3 Color Space Conversion

Color space conversion is a fundamental aspect of image processing that involves converting images from one color representation to another. One common conversion is from RGB (Red, Green, Blue) to grayscale, which represents each pixel with a single intensity value corresponding to its brightness. Grayscale images simplify processing tasks by reducing the dimensionality of the data and removing color information. Another important conversion

is from RGB to HSV (Hue, Saturation, Value) or HSL (Hue, Saturation, Lightness), which represent colors in terms of hue, saturation, and either value or lightness. These color spaces are particularly useful for color-based segmentation tasks, where objects of interest are identified based on their color characteristics. Additionally, color quantization is a technique used to reduce the number of colors in an image while preserving its overall appearance. This is beneficial for simplifying image analysis tasks or reducing memory usage in applications where color fidelity is not critical. Overall, color space conversion techniques are essential tools in image processing, enabling efficient manipulation and analysis of color information in digital images.

2.2.4 Image Enhancement

Image enhancement techniques are utilized in image processing to improve the visual quality and interpretability of images for better analysis or presentation. Contrast enhancement is a common technique aimed at increasing the dynamic range of pixel intensities, resulting in a more visually appealing and informative image. This adjustment enhances the distinction between different features and details within the image, making it easier to perceive and interpret. Noise reduction is another essential enhancement process that involves the removal of unwanted noise or artifacts from the image, which may result from various sources such as sensor limitations, transmission errors, or environmental factors. Techniques such as filters or statistical methods are employed to effectively suppress noise while preserving important image features. Additionally, image restoration techniques are applied to recover degraded or damaged images, restoring them to a more pristine state. Methods such as deconvolution or inpainting are commonly used to reconstruct missing or corrupted image regions, improving overall image quality and usability.

2.3 Machine Learning Models

2.3.1 Visual Geometry Group (VGG)

The Visual Geometry Group (VGG) network is renowned for its simplicity and effectiveness in image classification tasks, making it a seminal convolutional neural network architecture. VGG16, one of its most popular architectures, consists of 16 layers, including 13 convolutional layers and 3 fully connected layers. Notably, the convolutional layers are structured with small 3x3 filters, maintaining a consistent filter size throughout the network. This uniformity simplifies the architecture and enables deeper networks while reducing the number of parameters, which is advantageous for object detection tasks. Additionally, VGG16's deep stack of convolutional layers followed by max-pooling layers for spatial downsampling allows

it to capture increasingly abstract features as the network progresses deeper, making it well-suited for detecting objects with varying complexities.

2.3.2 Inception V3

Inception V3 is a powerful convolutional neural network architecture renowned for its effectiveness in various computer vision tasks, including object detection. Developed by Google, Inception V3 stands out for its innovative design, featuring inception modules that enable efficient multi-scale feature extraction. These modules consist of parallel convolutional layers with different receptive fields, allowing the network to capture both local and global features effectively. Additionally, Inception V3 incorporates techniques such as factorized convolutions and dimensionality reduction to reduce computational complexity while maintaining high performance. Its ability to extract rich hierarchical features at different scales makes it particularly well-suited for object detection tasks where detecting objects of varying sizes and complexities is crucial. Moreover, Inception V3's architecture facilitates transfer learning, enabling the model to be fine-tuned on specific detection datasets with relatively few training samples. Overall, Inception V3's combination of efficient feature extraction, computational scalability, and transfer learning capabilities makes it a highly effective choice for object detection tasks in diverse real-world scenarios.

2.3.3 Residual Networks (ResNet)

Residual Networks, commonly known as ResNet, have revolutionized the field of deep learning, particularly in the domain of image detection. Developed by Microsoft Research, ResNet introduces the concept of residual connections, which allow for the training of extremely deep neural networks. These residual connections enable the direct propagation of information from earlier layers to later layers, mitigating the vanishing gradient problem and facilitating the training of networks with hundreds or even thousands of layers. In the context of detection tasks, ResNet's deep architecture enables the extraction of hierarchical features of increasing complexity, crucial for accurately detecting objects in images. Moreover, ResNet architectures come in various depths, such as ResNet-50, ResNet-101, and ResNet-152, offering flexibility to choose models based on computational resources and performance requirements. With their ability to effectively capture intricate features and their scalability to accommodate deep architectures, ResNet models have become a cornerstone in the field of object detection, consistently achieving state-of-the-art results on benchmark datasets and real-world applications.

2.3.4 You Only Look Once (YOLO)

You Only Look Once (YOLO) represents a groundbreaking approach to object detection, renowned for its speed and accuracy. Unlike traditional detection methods that rely on region proposal algorithms followed by classification, YOLO takes a unified approach by simultaneously predicting bounding boxes and class probabilities for objects within an image. This single-shot detection mechanism allows YOLO to operate in real-time, making it particularly well-suited for scenarios where speed is paramount, such as video surveillance, autonomous vehicles, and real-time monitoring systems. YOLO's architecture divides the input image into a grid and predicts bounding boxes and class probabilities for each grid cell, significantly reducing computational complexity compared to region-based approaches. Furthermore, YOLO achieves impressive detection performance across a wide range of object classes and sizes, making it a versatile choice for various detection tasks. Its ability to deliver fast and accurate results in real-time scenarios has made YOLO a popular choice for applications requiring rapid object detection and tracking, where timely decision-making is critical.

YOLOv5, an evolution of the YOLO family, represents a significant advancement in object detection. Developed by Ultralytics, YOLOv5 builds upon the speed and accuracy of its predecessors while introducing several key improvements. One notable feature is its streamlined architecture, which consists of a single model size capable of achieving competitive performance across various datasets and tasks. This simplicity allows for easier deployment and integration into production systems. Additionally, YOLOv5 introduces novel techniques such as AutoML and automated hyperparameter optimization, enabling efficient model scaling and customization. Furthermore, YOLOv5 leverages advanced data augmentation and training strategies to improve generalization and robustness, particularly in scenarios with limited training data. With its combination of speed, accuracy, and versatility, YOLOv5 has quickly gained traction in the computer vision community, offering a state-of-the-art solution for object detection in real-world applications.

YOLOv8 is the latest iteration of the popular YOLO object detection algorithm, offering significant improvements over previous versions like YOLOv5. Developed by Ultralytics, YOLOv8 boasts faster inference speeds, better accuracy, and enhanced capabilities compared to its predecessors. It leverages cutting-edge techniques like decoupled head for improved performance and model efficiency. YOLOv8 also introduces new features such as instance segmentation, allowing for pixel-level object segmentation. With its modular design and seamless integration with various deep learning frameworks, YOLOv8 offers a powerful and flexible solution for real-time object detection and computer vision applications, making it a preferred choice over older versions like YOLOv5.

2.4 Deployment and Integration

2.4.1 Oracle VirtualBox

Oracle VirtualBox is an open-source virtualization platform that allows users to run multiple operating systems simultaneously on a single physical machine. It provides a convenient and cost-effective solution for testing software, developing applications, and running legacy or incompatible software without the need for separate hardware. VirtualBox supports a wide range of guest operating systems, including various versions of Windows, Linux, macOS, and more. It offers features such as snapshotting, which allows users to save the current state of a virtual machine and revert to it later if needed, and cloning, which enables the quick duplication of virtual machines for deployment or testing purposes. VirtualBox also provides seamless integration between host and guest systems through features like shared folders, clipboard sharing, and drag-and-drop functionality, enhancing user experience and productivity. With its user-friendly interface, extensive documentation, and active community support, Oracle VirtualBox is a versatile and popular choice for individuals, developers, and organizations seeking to leverage the benefits of virtualization for their computing needs.

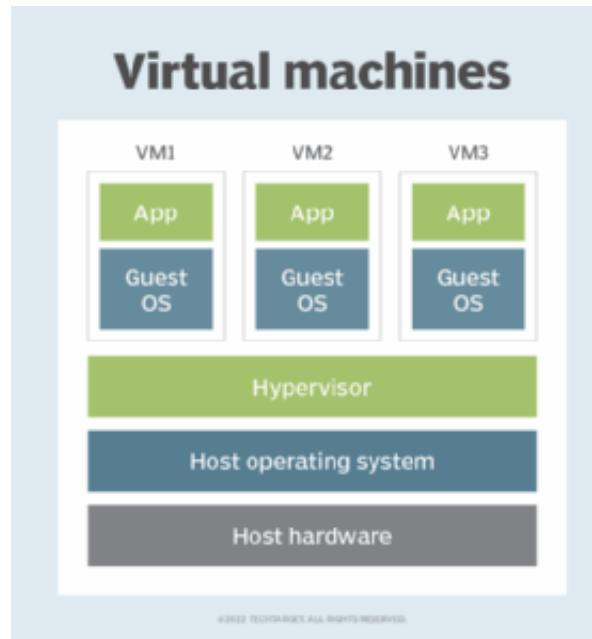


Figure 2.1: Virtual machine architecture

2.4.2 LabelImg

LabelImg is an open-source graphical image annotation tool used for labeling images with bounding boxes for object detection tasks. Developed by Tzutalin, LabelImg offers an intuitive and user-friendly interface, making it easy for annotators to draw precise bounding

boxes around objects of interest in images. It supports various annotation formats, including Pascal VOC and YOLO, ensuring compatibility with different machine learning frameworks and pipelines. Labellmg also provides useful features such as zooming, panning, and keyboard shortcuts for efficient annotation workflow. Additionally, it allows users to save and load annotation projects, facilitating collaboration and iterative labeling processes. With its simplicity, versatility, and active community support, Labellmg is used for annotating datasets used in training object detection models for computer vision applications.

2.4.3 Tensorflow

TensorFlow is an open-source machine learning framework developed by Google Brain. It is widely used for building, training, and deploying machine learning models, particularly deep learning models, across a range of tasks and industries. TensorFlow offers a flexible and scalable platform for developing both research prototypes and production-ready applications. One of its key features is its computational graph abstraction, where operations are represented as nodes and data flows as edges between these nodes, enabling efficient parallel execution on CPUs, GPUs, or even specialized hardware like TPUs. TensorFlow provides high-level APIs like Keras for easy model development and low-level APIs for more fine-grained control and optimization. Its extensive library of pre-built models and modules, known as TensorFlow Hub, allows developers to leverage pre-trained models for various tasks, speeding up development and reducing the need for large datasets. TensorFlow also supports distributed training, allowing models to be trained across multiple devices or servers for improved performance and scalability. With its rich ecosystem, comprehensive documentation, and active community support, TensorFlow has become one of the most popular and widely adopted frameworks for machine learning and deep learning applications.

2.4.4 PyTorch

PyTorch is an open-source machine learning framework developed by Facebook's AI Research lab. It is renowned for its flexibility, simplicity, and dynamic computational graph, making it particularly popular among researchers and developers. PyTorch allows for seamless integration into Python, enabling rapid prototyping and experimentation with machine learning models. Its dynamic computational graph feature enables developers to define and modify computational graphs on-the-fly, facilitating easier debugging and experimentation. It provides a rich set of tools and utilities for building deep learning models, including automatic differentiation, a variety of optimization algorithms, and support for GPU acceleration. It also provides a rich ecosystem of libraries and tools tailored for computer vision tasks, such as torchvision, which offers pre-trained models, datasets, and common utilities for image pro-

cessing.

2.4.5 React

React is a JavaScript library developed by Facebook for building interactive and dynamic user interfaces. Its component-based architecture allows developers to create reusable UI components, simplifying the development process and promoting code reusability. React's declarative syntax enables developers to describe how the UI should look based on the application's state, making it easier to manage complex UIs and maintain consistency across the application. Additionally, React's virtual DOM efficiently updates the actual DOM, resulting in improved performance and faster rendering of UI changes. With its extensive ecosystem and strong community support, React provides developers with a robust set of tools and libraries for building modern web applications. Its flexibility, scalability, and efficiency make it a popular choice for developing single-page applications, progressive web apps, and other interactive web experiences.

2.4.6 PostgreSQL

PostgreSQL is a powerful open-source relational database management system (RDBMS) known for its reliability, scalability, and extensive feature set. It offers robust support for ACID (Atomicity, Consistency, Isolation, Durability) transactions, ensuring data integrity and reliability. PostgreSQL's advanced indexing capabilities, including B-tree, hash, and GiST (Generalized Search Tree), enable efficient querying and high-performance data retrieval. Its support for complex data types such as JSON, XML, and arrays makes it suitable for handling diverse data structures. Additionally, PostgreSQL provides comprehensive SQL support along with support for stored procedures, triggers, and user-defined functions, enabling developers to implement complex business logic within the database. With its active community, frequent updates, and adherence to SQL standards, PostgreSQL is a popular choice when data integrity, performance, and scalability are paramount.

2.4.7 Tkinter

Tkinter is a built-in Python library for creating graphical user interfaces (GUIs) quickly and easily. It provides a simple and intuitive way to design and implement GUI applications, making it ideal for both beginners and experienced developers. Tkinter offers a variety of widgets, such as buttons, labels, entry fields, and text boxes, which can be easily customized to suit the specific requirements of the application. Its event-driven programming model allows developers to define callbacks to handle user interactions, such as button clicks or key presses. Tkinter's seamless integration with Python's object-oriented programming paradigm

facilitates the creation of modular and maintainable code. With its lightweight and portable nature, Tkinter applications can run on various platforms without requiring additional dependencies.

2.4.8 OpenCV

OpenCV, short for Open Source Computer Vision Library, is a widely-used open-source library for computer vision and image processing tasks. It offers a vast array of functionalities, including image and video processing, object detection and tracking, feature extraction, and machine learning algorithms. OpenCV provides a comprehensive set of tools and utilities for working with images and videos, enabling developers to perform complex operations such as edge detection, image filtering, and geometric transformations with ease. Its Python interface, in particular, has made it highly accessible to developers, allowing them to leverage the power of OpenCV alongside the flexibility and simplicity of Python programming. With its extensive documentation, active community support, and cross-platform compatibility, OpenCV has become a go-to choice for computer vision applications.

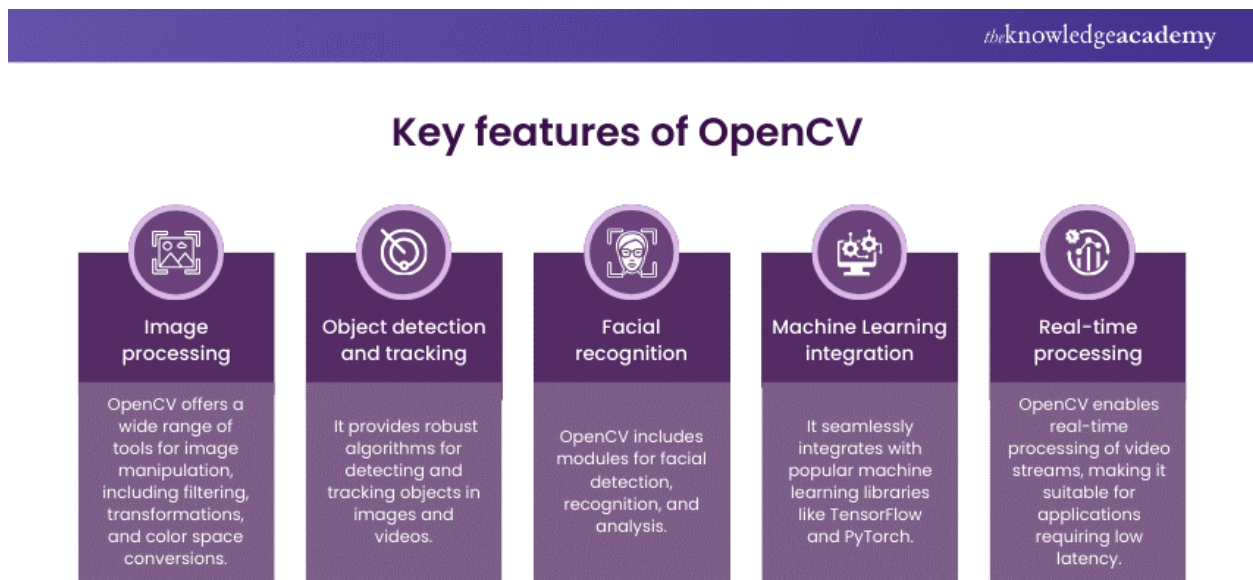


Figure 2.2: Application of opencv

Chapter 3

Snapshots

3.1 Module implementation

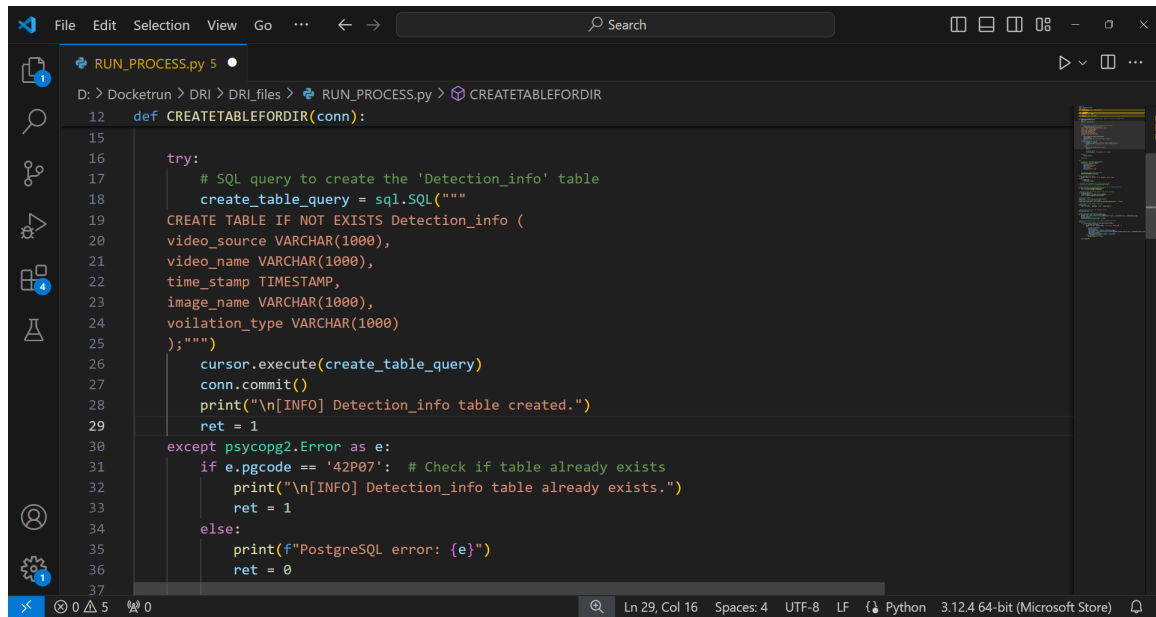
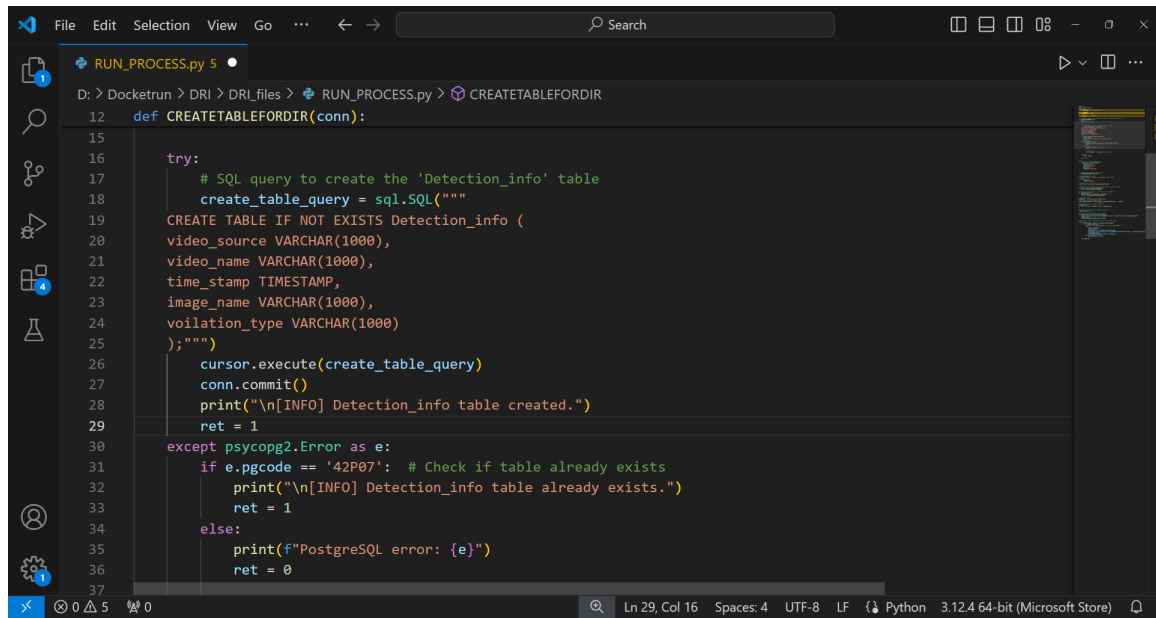


Figure 3.1: Block of code to create database table

The code block defines a function named *CREATETABLEFORDIR* that aims to ensure the presence of a table called *Detection_info* in a PostgreSQL database. It takes a database connection object as an argument, constructs a SQL query to create the table with specified columns if it does not already exist, and executes this query. If the table already exists, it catches the specific error indicating this and prints an informational message. This function ensures the necessary table structure is in place for storing detection information, handling both creation and existing scenarios efficiently.



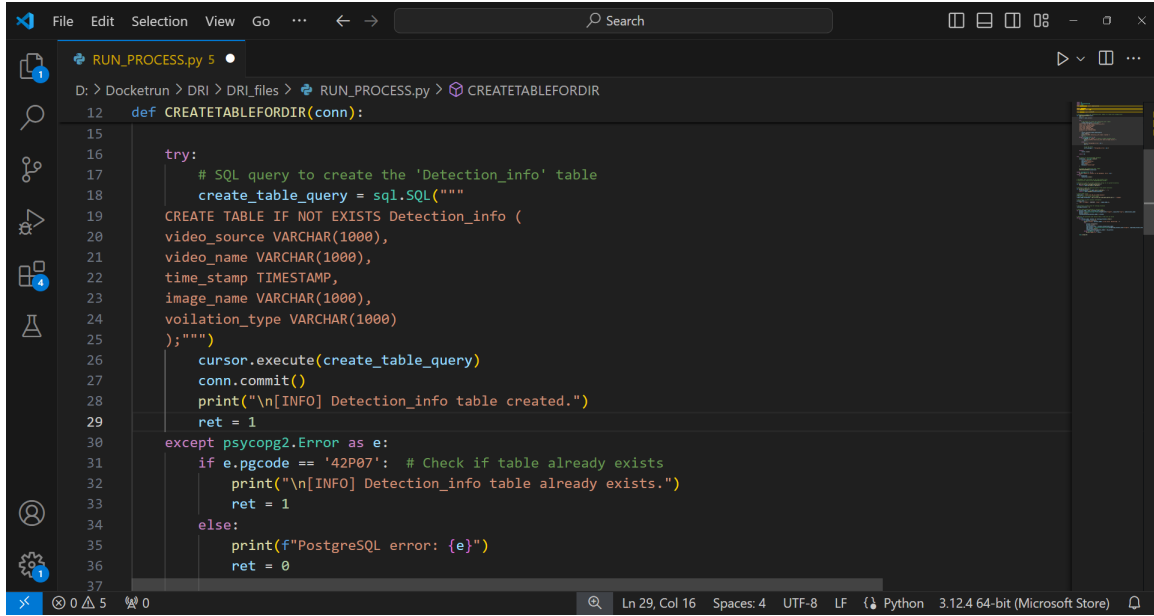
```

12 def CREATETABLEFORDIR(conn):
13
14     try:
15         # SQL query to create the 'Detection_info' table
16         create_table_query = sql.SQL("""
17         CREATE TABLE IF NOT EXISTS Detection_info (
18         video_source VARCHAR(1000),
19         video_name VARCHAR(1000),
20         time_stamp TIMESTAMP,
21         image_name VARCHAR(1000),
22         voilation_type VARCHAR(1000)
23         );""")
24         cursor.execute(create_table_query)
25         conn.commit()
26         print("\n[INFO] Detection_info table created.")
27         ret = 1
28     except psycopg2.Error as e:
29         if e.pgcode == '42P07': # Check if table already exists
30             print("\n[INFO] Detection_info table already exists.")
31             ret = 1
32         else:
33             print(f"PostgreSQL error: {e}")
34             ret = 0
35
36
37

```

Figure 3.2: Block of code to connect to database

The code establishes a connection to a PostgreSQL database and creates a table named *Detection_info*. The main benefit of this code is its ability to interact with a PostgreSQL database programmatically. By connecting to the database and creating tables, it enables seamless data storage and retrieval for various applications. This functionality is crucial for applications that require persistent data management, such as web applications, data analysis tools, or any system that needs to store and manipulate structured data. The script handles connection errors gracefully by catching exceptions and displaying an error message if the connection fails. It also ensures that the database connection is properly closed, even in the event of an exception, by utilizing the finally block. This practice promotes efficient resource management and prevents potential resource leaks.



```

12 def CREATETABLEFORDIR(conn):
13
14     try:
15         # SQL query to create the 'Detection_info' table
16         create_table_query = sql.SQL("""
17         CREATE TABLE IF NOT EXISTS Detection_info (
18         video_source VARCHAR(1000),
19         video_name VARCHAR(1000),
20         time_stamp TIMESTAMP,
21         image_name VARCHAR(1000),
22         voilation_type VARCHAR(1000)
23         );""")
24         cursor.execute(create_table_query)
25         conn.commit()
26         print("\n[INFO] Detection_info table created.")
27         ret = 1
28     except psycopg2.Error as e:
29         if e.pgcode == '42P07': # Check if table already exists
30             print("\n[INFO] Detection_info table already exists.")
31             ret = 1
32         else:
33             print(f"PostgreSQL error: {e}")
34             ret = 0
35
36
37

```

Figure 3.3: Block of code for execution of main process

The code snippet demonstrates the implementation of a process monitoring and management system using Python's multiprocessing module. It starts by iterating over a dictionary called *process_info*, which contains the necessary information (target function, arguments, and process name) to initialize and start multiple processes. For each process, it creates a new multiprocessing.Process instance and adds it to a dictionary called *running_processes* for tracking purposes. The code then enters an infinite loop to continuously monitor the running processes. Within this loop, it checks the status of each process by calling the *is_alive()* method. If a process is found to be not alive, it prints a message indicating that the process is being restarted. It then attempts to terminate the old process using *terminate()* and joins it using *join()*. Subsequently, it retrieves the process information from *process_info* and creates a new process instance with the same details, effectively replacing the terminated process in *running_processes*. If an exception occurs during the process restart, it prints an error message. Finally, the code introduces a delay of 10 seconds using *time.sleep(10)* before checking the processes again in the next iteration of the loop. This approach ensures that the application remains resilient and can recover from process failures or crashes by automatically restarting the affected processes, thereby maintaining high availability and fault tolerance.

3.2 Training metric results

Once our machine learning model is trained, we assess various metrics to evaluate its performance and generalizability for future use. Key metrics include different types of losses and

performance parameters such as precision, recall, and mean average precision. These metrics help us understand how accurately the model identifies objects, how well it detects all relevant objects, and how robustly it performs across different scenarios. By analyzing these metrics, we can ensure the model is effective and reliable for real-world applications.

3.2.1 Loss Graphs

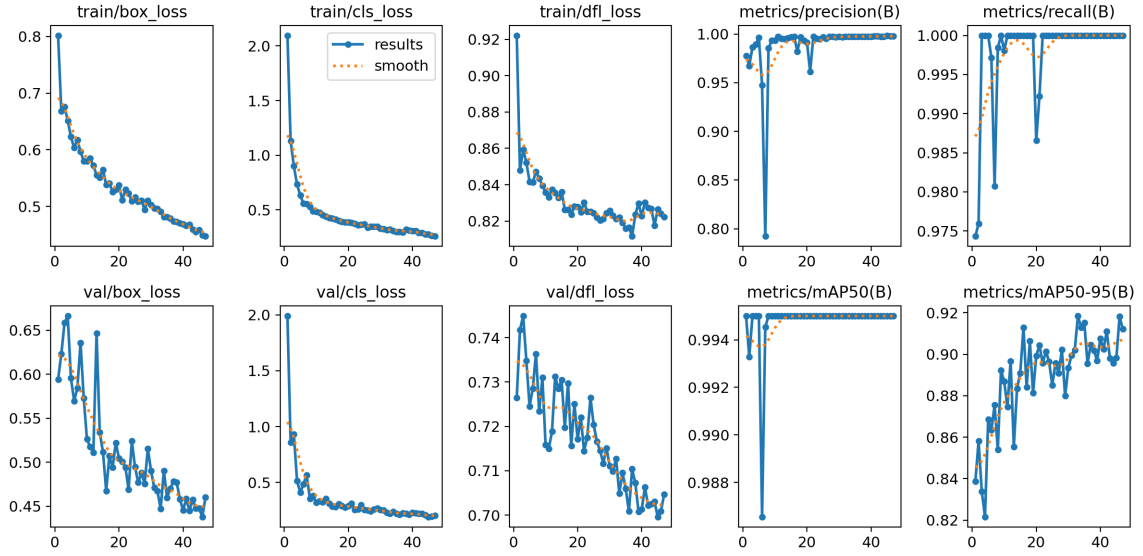


Figure 3.4: Training metrics results

1. train/box_loss: The graph shows a steady decrease, indicating that the model is learning to predict bounding boxes more accurately over time.
2. train/cls_loss: A steady decrease in this graph indicates that the model is improving its object classification accuracy during training.
3. train/df_l_loss: The graph shows a consistent decline, indicating that the model is getting better at refining the bounding box predictions over the training epochs.
4. val/box_loss: The decrease and stabilization in this loss suggest that the model generalizes well to unseen data in terms of predicting bounding boxes.
5. val/cls_loss: The downward trend indicates that the model is maintaining good classification accuracy on the validation set, showing good generalization.
6. val/df_l_loss: The decrease in this loss shows that the model is effectively refining its bounding box predictions on the validation data, similar to the training data.

3.2.2 Metric Graphs

1. metrics/precision(B): The graph shows high precision values, close to 1.0, indicating that the model makes very few false positive errors. This is excellent for applications where false positives are costly.
2. metrics/recall(B): The recall values are also very high, close to 1.0, indicating that the model detects nearly all the objects it is supposed to, with very few false negatives.
3. metrics/mAP50(B): The graph shows near-perfect mAP50 values, indicating excellent object detection performance at this IoU threshold.
4. metrics/mAP50-95(B): The graph shows high and steadily improving mAP50-95 values, indicating robust performance across various IoU thresholds. This suggests the model performs well in both loose and strict matching conditions.

The model has high precision and recall values, which means it accurately detects objects with few false positives and false negatives. This is ideal for object detection tasks where both accuracy and completeness are important. The decreasing and stabilizing loss values in both training and validation indicate that the model is learning effectively and generalizing well to unseen data.

Chapter 4

Results and Discussions

During the internship, the training approach was project-based rather than having a dedicated training period. From the very first day, interns were assigned projects and tasks to work on, immersing them directly into real-world applications and challenges. The manager provided comprehensive briefings on these tasks and the problems that needed to be solved, ensuring that interns had a clear understanding of their objectives and expected outcomes. This hands-on approach was designed to foster practical learning and immediate application of skills. To further support the learning process, a senior Data Engineer was assigned as a mentor to guide and support the interns throughout their projects. This mentorship role was crucial in providing expert advice, troubleshooting assistance, and professional insights.

The interns were introduced to the systems in use and given a high-level understanding of the codebase and relevant concepts. This introduction included an overview of the architecture, key components, and operational workflows, which laid the groundwork for their independent exploration. However, the primary emphasis was on self-learning and independently grasping the concepts and codebase. The mentor held periodic discussions with the interns to address any issues or questions that arose during the learning process, facilitating a deeper understanding through interactive dialogue and practical demonstrations. These discussions were tailored to the interns' progress and specific challenges, ensuring personalized guidance.

Due to the complexity of the problem statements in industrial environments, it was acknowledged that a more thorough understanding of these systems required additional time. Interns often encountered sophisticated technical issues and intricate workflows that demanded a deeper dive into the system's intricacies. Despite this challenge, the manager and mentor provided unwavering support and guidance to ensure a smooth learning experience for the interns. They offered additional resources, arranged supplementary sessions, and provided constructive feedback to help interns navigate these complexities effectively. Standup meetings were conducted daily to discuss progress and any challenges or issues encountered. These meetings were a platform for interns to share their experiences, receive immediate feedback, and collaboratively address any roadblocks they were facing. The regular interaction with the team facilitated a collaborative learning environment and ensured that interns remained aligned with project goals while continuously improving their skills.

Chapter 5

Conclusion and Future Scope

The internship offered an in-depth understanding of how a leading AI solutions company addresses diverse problem cases. It provided the opportunity to work on tasks involving various systems and services, offering hands-on experience with a wide array of tools and technologies. This immersive experience exposed interns to the intricacies of solving detection problems in environments where data clarity is an issue. By working with cutting-edge technologies and tools used in data processing, storage, and analysis, interns gained practical experience and a deep understanding of the challenges and considerations involved in handling large-scale data systems.

A notable aspect of the internship was the opportunity to collaborate with professionals at different levels of experience. This created a rich learning environment where interns could leverage the expertise of seasoned professionals while contributing their own unique perspectives and ideas. Such collaboration fostered a culture of knowledge sharing, allowing interns to learn from the collective experiences of their colleagues. Engaging with diverse teams provided valuable insights into various problem-solving approaches and enhanced the overall learning experience.

The internship also provided interns with the freedom to implement their own creative and innovative solutions to problems, contributing to the company's codebase. This autonomy encouraged critical thinking, innovation, and a sense of ownership over their work. Interns were able to showcase their skills, learn from their successes and failures, and make tangible contributions to the company's projects. Overall, the internship was a hands-on learning experience where independent learning was encouraged, supported by a team of mentors. The exposure to various systems, collaboration with experienced professionals, and the opportunity to contribute to real-world projects made the internship an invaluable learning opportunity for all participants.