JISSE
ISSN: 2636-4425

Journal of International Society for Science and Engineering
Vol. 6, No. 4, 100-113 (2024)

JISSE
E-ISSN:2682-3438

# Leveraging YOLOv8 and FaceNet for Enhanced Surveillance and Organization in Medicinal Warehouses

Ghada Abdelhady [1,*], Sameh Ayoub [2], Mohab Youssef [2]

[1] October University for Modern Sciences and Arts, General Systems Engineering, Faculty of Engineering, Egypt

[2,3] October University for Modern Sciences and Arts, ECE Department, Faculty of Engineering, Egypt

ARTICLE INFO

*Article history:*
*Received:22-01-2025*
*Accepted:28-01-2025*
*Online:02-02-2025*

Keywords:
AI-powered Automation
Inventory Control
Jetson Nano Implementation
Medicinal Supply Chain Security
Warehouse Management

ABSTRACT

Medicinal warehouses face numerous challenges, including disorganization, theft, and human errors in inventory management, which result in economic losses and unauthorized access to controlled substances. To address these issues, this study presents an advanced system that integrates the "You Only Look Once (YOLO)" algorithm for real-time object detection and tracking, implemented on a Jetson Nano microprocessor with cameras and sensors. The system automates inventory management by organizing medicinal products, reducing human error, and monitoring access to controlled substances through ultrasonic sensors and theft detection mechanisms. Additionally, a machine learning model trained on historical inventory data predicts future stock demands, enabling proactive restocking to prevent shortages and maintain operational efficiency. YOLOv8 was selected for its superior accuracy and speed, achieving a mean Average Precision (mAP) of 0.923, alongside reductions in Box Loss, Class Loss, and Distribution Focal Loss (DFL). The results demonstrate high-accuracy detection and classification, even under challenging conditions, ensuring a reliable and scalable solution for medicinal warehouse management. This system provides enhanced security, operational efficiency, and predictive capabilities, contributing to better inventory control and public health outcomes. Future work will focus on improving system scalability, integrating real-time data streams, and refining predictive accuracy for broader adoption.

## 1. Introduction

Efficient inventory management and surveillance in medicinal warehouses are critical for ensuring the availability and security of essential supplies. These facilities face unique challenges, including disorganization, theft, and errors in inventory tracking, which can disrupt healthcare services and lead to economic losses. Traditional management systems often rely on manual processes or limited automation, which fail to address the complexities of modern medicinal storage needs [1], [2].

Advancements in artificial intelligence (AI) and computer vision have paved the way for innovative solutions to these challenges. Deep learning models, particularly convolutional neural networks (CNNs), have revolutionized object detection tasks, offering high accuracy and speed in real-time applications. Algorithms like YOLO (You Only Look Once) have proven effective in various domains, from traffic monitoring to industrial quality control, due to their ability to detect and classify objects with exceptional efficiency. The latest version, YOLOv8, builds upon its predecessors with improved detection capabilities, reduced computational requirements, and enhanced adaptability to diverse environments [3].

This study explores the integration of YOLOv8 into a comprehensive system for medicinal warehouse management, focusing on real-time inventory tracking and theft prevention. The proposed system leverages a custom-trained YOLOv8 model to detect and classify medicinal products, ensuring accurate inventory updates. Additionally, the system incorporates predictive analytics to forecast stock levels, optimizing

* Ghada Abdelhady, , General Systems Engineering, Faculty of Engineering, October University for Modern Sciences and Arts, Egypt, +201274086431, ghada.abdelhady@hotmail.com

replenishment processes and minimizing supply chain disruptions. By addressing these critical challenges, this research aims to provide a scalable and cost-effective solution for enhancing the operational efficiency and security of medicinal warehouses.

## 2. Related Works

Recent research has explored the use of advanced technologies, such as artificial intelligence (AI), IoT, and deep learning, to address challenges in medicinal warehouse management, including inventory tracking, theft prevention, and operational efficiency. Several works have proposed innovative systems that leverage these technologies, with varying results and limitations.

Ogbewele et al. (2024) introduced a smart inventory management system that combined IoT sensors with cloud-based analytics for real-time inventory tracking. The system achieved significant improvements in monitoring environmental conditions, such as temperature, and a substantial reduction in stock discrepancies. However, it faced challenges related to scalability for larger warehouses and incurred high initial implementation costs due to IoT infrastructure [1].

Kakade (2024) proposed an AI-powered video surveillance system using YOLOv5 for object detection, targeting unauthorized access and theft prevention in pharmaceutical warehouses. The system demonstrated high detection accuracy and effectively integrated predictive analytics for inventory planning. Despite these advancements, the model faced difficulties in complex scenarios, such as overlapping objects and poor lighting conditions, which affected its detection reliability [2].

The results and performance metrics of these studies, along with other comparative works, are summarized in Table 1.

**Table 1. Performance Metrics of Related Work**

| Study | Technology Used | Performance Metrics | Key Limitations |
|---|---|---|---|
| Ogbewele et al. (2024) [1] | IoT sensors and cloud-based analytics | Reduction in stock discrepancies; improved environmental monitoring. | High implementation cost; scalability issues |
| Kakade (2024) [2] | YOLOv5 and AI-powered surveillance | Detection accuracy: 94%; integration of predictive analytics | Poor performance in low-light or overlapping scenarios |
| The proposed System | YOLOv8 + Jetson Nano | Stock Prediction: 92.15% A mean Average Precision (mAP) of 0.923 | careful placement of sensors and cameras to minimize blind spots |

A review conducted by International Journal of Development Research (2023) emphasized the growing adoption of AI and IoT in the pharmaceutical sector to enhance inventory management practices. The paper highlighted the benefits of AI-driven object detection models and predictive analytics in reducing human error, improving tracking accuracy, and mitigating theft. However, the review identified a lack of comprehensive solutions that address scalability, cost-efficiency, and adaptability to dynamic warehouse conditions [4].

These studies underline the transformative potential of integrating AI and IoT in medicinal warehouse management but also reveal persistent challenges in scalability, adaptability, and operational efficiency. To address these gaps, this study proposes a novel approach leveraging YOLOv8, an advanced object detection algorithm known for its superior accuracy, speed, and adaptability in real-time applications. By integrating YOLOv8 with predictive analytics and scalable hardware solutions like Jetson Nano, the proposed system offers a comprehensive solution to enhance inventory tracking, theft prevention, and operational efficiency in medicinal warehouses.

## 3. Methodology

The proposed medical storage system employs a combination of advanced hardware and deep learning algorithms to address the unique challenges of inventory management and security in medicinal warehouses. YOLOv8, known for its exceptional accuracy and real-time detection capabilities, is utilized for object detection and classification of medicines. Additionally, the Jetson Nano microcontroller processes the data collected from cameras and ultrasonic sensors, ensuring seamless operation and quick decision-making.

The system is designed not only to track the quantity and type of medicines but also monitor employee activities using facial recognition. A machine learning regression model complements these features by predicting future inventory requirements based on historical usage patterns. This integration of detection, recognition, and predictive capabilities ensures that the system minimizes human error, prevents unauthorized access, and maintains optimal inventory levels.

The following subsections provide a detailed explanation of the system's components, architecture, and workflow, highlighting how each element contributes to achieving efficient and secure medicinal warehouse management.

### *3.1. YOLOv8 and Jetson Nano-Based Medical Storage System*

The proposed medical storage system utilizes a Jetson Nano microcontroller with YOLOv8 for object and face detection, identifying medicines and recognizing individuals. It also includes a machine learning regression model to predict future sales and product usage. By integrating YOLOv8 with facial recognition, this system addresses challenges in medicinal warehouse surveillance, such as inventory management, security, and predictive stock analysis, providing a novel solution to mitigate risks like unauthorized access and stock shortages.

#### *3.1.1 Detailed Design of the Medical Storage System*

As shown in Figure 1, the system relies primarily on hardware. YOLO, chosen for its accuracy and speed, is the deep learning system used for object detection. Camera and ultrasonic sensors will monitor medicines on shelves. The system runs on a Jetson Nano microprocessor, meeting the high specifications needed for efficient YOLO operation. Finally, staff members access the system remotely via a PC.

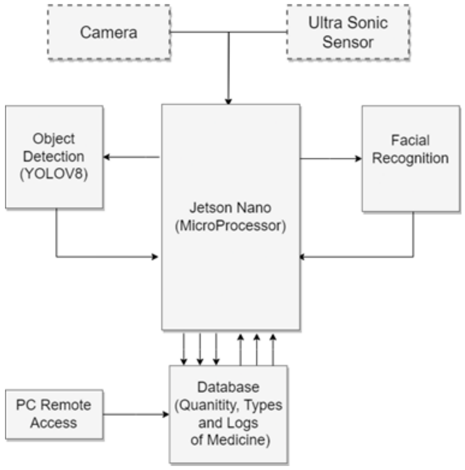


**Figure 1. Detailed System Block diagram**

The medical storage system employs advanced technologies for efficient inventory management and predictive analytics, as shown in Figure 2, incorporating a Jetson Nano microcontroller with cameras and sensors for real-time monitoring. It uses the YOLOv8 algorithm for object and face detection to identify and count medicines while tracking employee activities [5].

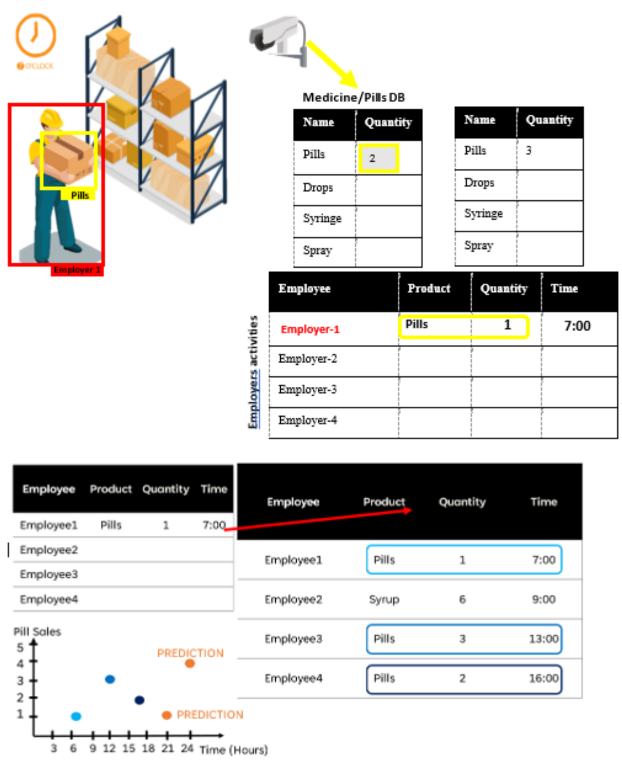


**Figure 2. Detailed Scenario on How the proposed system works**

*3.2. Components and Workflow*

**Image Processing and Detection**

- Object Detection: YOLOv8 detects and counts medicines on shelves in real-time, identifying employees like "Employee 1" carrying pill boxes.
- Face Detection: FaceNet are used for facial recognition, logging employee interactions with inventory.

**Database Management**

- Medicine/Pills DB: Records all medicines, including names and quantities, e.g., 2 pills available.
- Employee Activities: Logs actions like "Employee 1" taking 1 pill at 7:00, with real-time updates for accurate inventory tracking.

**Activity Logging:**

Maintains detailed logs for all activities, including who took what and when, facilitating easy monitoring and auditing.

**Predictive Analytics:**

A basic machine learning model forecasts future sales and usage, helping prevent shortages and optimize inventory levels.

**System Benefits:**

- Automation: Minimizes human error in inventory management.
- Theft Prevention: Detects unauthorized access using sensors and cameras.
- Efficiency: Automates restocking and demand forecasting.

By integrating YOLOv8 for detection with machine learning for analytics, the system enhances security and efficiency in managing medicinal warehouses.

*3.3. System Flowchart*

The flowchart in Figure 3 outlines the medical storage system's process, from image capture to inventory updates.

- Camera Input: A camera captures images of the storage area.
- YOLO Model: The images are processed by the YOLO deep learning model, which identifies medicine types and quantities and recognizes warehouse employees, possibly using FaceNet.
- Medicine Analysis: The system analyzes the identified medicines for quantity changes.
- Quantity Check: If the quantity has not changed, the system continues monitoring. If it has, it updates the records.
- Update Inventory and Logs: The inventory is updated, and logs are created detailing which employee took or added medicines, including quantities and timestamps.
- Check for More Images: The system checks for additional images from the camera, processing them if available, or ending the process if not.

To ensure effective system adoption, comprehensive training sessions will be held for warehouse staff, covering system operation, interaction, and basic troubleshooting. This training will empower employees to manage the technology confidently, reducing reliance on technical support. An introductory phase will include hands-on demonstrations and feedback sessions to address concerns and foster familiarity, encouraging acceptance and smoother integration into existing workflows.

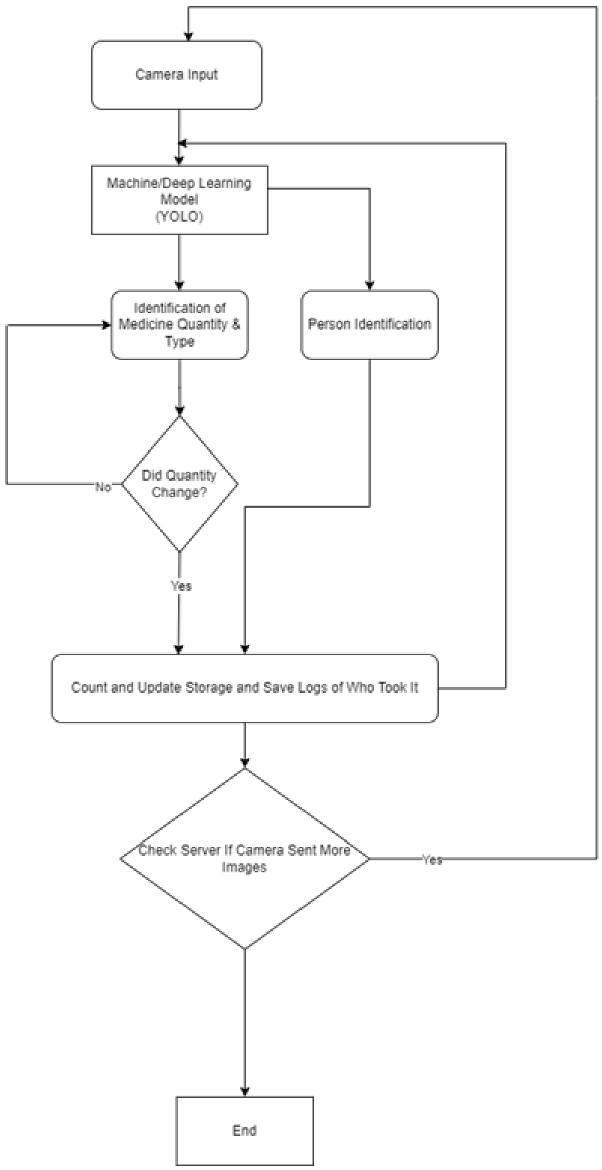


**Figure 3. Storage System Flow Chart**

### *3.4. Stock Demand Prediction*

To enhance inventory management, the system integrates a machine learning component for predicting future stock demands. Using a dataset containing historical inventory records, purchase trends, and usage patterns for various medicinal products, a time-series forecasting approach is employed, such as LSTM (Long Short-Term Memory) model analyzes historical inventory data (e.g., stock usage over days, weeks, or months) and learns patterns, such as seasonal demand increases or consistent product usage [6]. The dataset used for this study was obtained from a private pharmacy, containing historical inventory records of medicinal products over a period of time. It includes daily records for several products, detailing the stock levels, sales, restocking activities, and seasonal demand patterns. Specifically, the dataset has the following key features:

- Date: The recorded date of inventory activities.
- Product Name: Names of the medicinal products tracked in the inventory.
- Initial Stock and Ending Stock: The quantities available at the start and end of each day, respectively.
- Units Sold and Restocked: Quantities sold during the day and replenished through restocking.
- Seasonal Demand: A qualitative indicator of demand fluctuation based on external factors such as seasons or trends (e.g., High, Medium, Low).

The dataset captures realistic operational data, making it suitable for developing and testing a stock prediction model. Using this dataset, we applied an LSTM model to predict future stock requirements for proactive inventory management.

The workflow involves:

- **Data Preprocessing**: Historical inventory data is cleaned to handle missing entries, remove outliers, and normalize values.
- **Model Selection**: An LSTM-based time-series model is used to capture temporal patterns and trends in stock usage [6].
- **Prediction and Restock Automation**: The trained model estimates how much stock will be needed in the future. If the predicted stock levels fall below a certain limit, the system automatically generates restock requests to ensure there are no shortages.

This predictive feature ensures optimal stock levels, minimizing shortages and excess inventory in medicinal warehouses.

## 4. Proposed System Architecture and Key Technologies

### *4.1. "Jetson Nano" Microprocessor*

Jetson Nano is selected for YOLOv8 implementation. A comparative analysis in Table 3 evaluates the Raspberry Pi 3 B, Raspberry Pi 4, and Jetson Nano based on release year, CPU specifications, RAM capacity, GPU presence, and power consumption.

**Table 2. Different Microprocessors Comparison** [5]

| Model | Raspberry Pi 3 B | Raspberry Pi 4 | Jetson Nano |
|---|---|---|---|
| **Year** | 2015 | 2019 | 2019 |
| **CPU** | "1.2 GHz Cortex-A`53 ARM" | "1.8 GHz Cortex-A72 ARM" | "Quad-core ARM A57 CPU" |
| **RAM** | 1 GB | 1,4,8 GB | 4GB LPDDR4 64-bit |
| **GPU** | NONE | NONE | 128-core Maxwell |

| | | | |
|---|---|---|---|
| | | | GPU |
| **Power Consumption** | 1.2 Watt | 2.7 Watt | 5-10 Watts |

As shown in Table 2, the Jetson Nano outperforms the Raspberry Pi 3 and 4 in computing power due to its GPU, despite having a lower-end CPU. Its high computing power allows for higher frames per second and smoother image tracking, making it a cost-effective choice for object detection and image classification in our research.

### *4.2. System Architecture and Requirements*

The system predominantly relies on software, with YOLO deep learning selected for object detection, aided by cameras and sensors. The e-CAM30_CUNANO camera will provide adequate framerate support for this purpose. It will operate on a microprocessor Jetson Nano connected to sensors and will be monitored remotely by staff members using their PCs.

**Hardware Requirements**

The YOLO system will be deployed on a Jetson Nano, utilizing ultrasonic sensors and cameras for monitoring medicines on shelves.

**Software Requirements**

- YOLOv8 for Object Detection

YOLOv8, part of the YOLO family, is chosen for object detection. This single-shot multi-object detection algorithm analyzes the image only once, unlike traditional multi-stage algorithms. YOLOv8, the latest version, features an “anchor-free model” with a decoupled head, enhancing accuracy by allowing each branch to focus on its task [5], [7].

- Facial Recognition

Facial detection, a category of object detection, will be integrated into the system using a face detection algorithm based on YOLO. The proposed algorithm stages are as follows:

Object Detection: When a person is detected by YOLOv8 in the warehouse, The model creates a bounding box around him. Based on the value of the Intersection over Union (IoU) which is a key metric used in object detection to measure the overlap between the predicted bounding box and the ground truth bounding box, the accuracy is determined. IoU is defined as the ratio of the intersection area to the union area of the two bounding boxes, equation (1) [5], [7]. The illustration of IoU has been depicted in Figure 4:

$$IoU = \frac{Area\ of\ Overlap}{Area\ of\ Union} \quad (1)$$

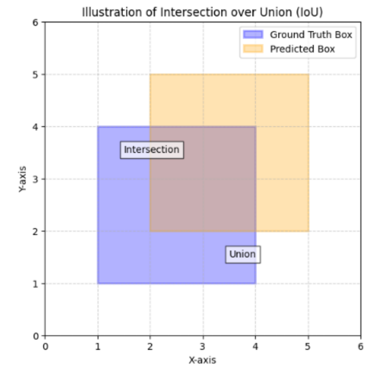


**Figure 4. Illustration of IoU**

A higher IoU indicates a more accurate prediction, with values typically ranging between 0 and 1.

In YOLOv8, IoU is used during training and evaluation to determine the quality of bounding box predictions. For example:

IoU > 0.5: The detection is considered a "True Positive."

IoU ≤ 0.5: The detection is a "False Positive."

Face Detection: FaceNet algorithm verifies if the bounding box contains a face. It uses a deep convolutional neural network (CNN) to map facial images into a compact Euclidean space, where similar faces have close embeddings, and dissimilar faces are far apart. The FaceNet model, pre-trained on the VGGFace2 dataset [8], was fine-tuned on a custom dataset of facial images collected from the warehouse environment. The dataset includes multiple images per employee, capturing variations in lighting, pose, and angles. Fine-tuning involved freezing the initial layers of the pre-trained model and training the final layers on the new dataset using triplet loss. This adaptation improved the model's ability to recognize authorized personnel with high accuracy in real-time. [9].

Labeling: The bounding box is labeled with the person’s name [10].

Applying the algorithm to each frame would be cost-inefficient. A proposed solution is to run the FaceNet model on the first frame after YOLOv8 detection and then use a tracking algorithm to maintain recognition data. Experimental results indicate that YOLO-based face detection outperforms traditional multi-stage detection, fulfilling real-time detection requirements [11]. A similar proposal integrating YOLOv5 with FaceNet achieved 99.5% accuracy on a dataset of 200 face images [12].

YOLOv8 is designed for fast, accurate object detection, achieving about 30 FPS—an optimal speed for real-time applications on embedded platforms like the Jetson Nano [13], [14]. Recent research emphasizes YOLOv8’s improvements, including Soft-NMS post-processing, advanced tracking, specialized data augmentation, and an optimized loss function, all boosting performance in dynamic real-time scenarios [15]. These features provide high precision, superior IoU, and mAP metrics, making YOLOv8 effective for real-time object detection and tracking. While not entirely infallible, its design greatly minimizes false positives and negatives under standard conditions, demonstrating reliability for medicinal warehouse surveillance.

4.2.3 Data Privacy, Consent, and Security Compliance

Our system complies with General Data Protection Regulation (GDPR) standards for securely handling biometric data, prioritizing the privacy of warehouse personnel, including doctors and nurses. We inform personnel about facial recognition use and obtain explicit consent prior to data collection. For transparency, personnel receive view-only access to review how their data is used, with formal review requests allowed if inaccuracies are identified. Authorized administrators manage these requests, preserving data integrity and maintaining an audit trail.

In line with data minimization, only essential biometric data is stored for operational needs and is promptly erased when no longer necessary. Confidentiality agreements ensure only designated personnel access sensitive data, build trust and support GDPR compliance. The system also integrates with existing security protocols at the facility, aligning with access controls and data handling practices to avoid redundant security efforts.

4.2.4 Hardware and Software Compatibility and Integration

The choice of YOLOv8 on Jetson Nano stems from Nano's high performance and compatibility with deep learning models, as shown in Table 3. To ensure effective integration, PyTorch was used to train the YOLOv8 model, which demonstrated an accuracy of 0.825 at mAP = 0.5 with a 512x512 input layer. By experimentation, YOLOv8 models also showed superior accuracy compared to YOLOv5 on the Jetson Nano [14]. Benchmarking research between Jetson Nano, Khadas VIM3, and Raspberry Pi4 showed the Jetson Nano GPU excels with YOLO family, demonstrating the benefit of using them together [16].

To integrate YOLOv8 and FaceNet on a single Jetson Nano, we addressed potential challenges in processing power allocation. Since the Nano handles real-time object detection, face recognition, and sensor data, we optimized memory allocation and processing threads to prevent FaceNet from competing for GPU resources with YOLOv8. Specific scheduling was implemented to prioritize YOLOv8 for real-time performance. Further tests confirmed stable operation across both models, achieving high frame rates and detection accuracy on the Nano, surpassing comparable single-board computers [17] as shown in Table 3.

**Table 3. Hardware and YOLO models compatibility by tested Average FPS** [18]

| Model | Cluster UEM (FPS) | Raspberry Pi 3B+ (FPS) | Raspberry Pi 4 (FPS) | Jetson Nano (FPS) |
|---|---|---|---|---|
| YOLOv5n | 130.44 | 0.46 | 1.3 | 14.7 |
| YOLOv5s | 114.72 | 0.19 | 0.73 | 4.8 |
| YOLOv8n | 75.08 | 0.27 | 0.76 | 6.2 |
| YOLOv8s | 72.24 | 0.09 | 0.44 | 3.3 |

The large discrepancy in FPS values between the "Cluster UEM" and the single-board computers (Raspberry Pi 3B+, Raspberry Pi 4, and Jetson Nano) arises primarily due to differences in hardware specifications and computational capabilities [18]:

Cluster UEM: This is a high-performance computing environment optimized for handling computationally intensive tasks. It is equipped with powerful CPUs and GPUs designed to handle deep learning workloads efficiently, resulting in significantly higher FPS values.

Raspberry Pi 3B+ and Raspberry Pi 4: These are general-purpose, low-cost single-board computers with limited computational resources, especially in terms of GPU capabilities. Their FPS values are extremely low compared to Cluster UEM due to the lack of dedicated hardware acceleration for deep learning tasks.

Jetson Nano: While also a single-board computer, the Jetson Nano includes a 128-core Maxwell GPU, which provides some level of hardware acceleration for deep learning models. This results in better FPS performance compared to the Raspberry Pi devices, but it still falls far behind the Cluster UEM due to its relatively limited GPU power and memory bandwidth.

Model Complexity: The FPS values also decrease with the complexity of the YOLO model. For example, YOLOv5n and YOLOv8n are lightweight versions optimized for speed, while YOLOv5s and YOLOv8s are slightly heavier models, resulting in lower FPS on all hardware configurations.

Inference Environment: The testing environment for Cluster UEM likely uses optimized drivers, software frameworks, and configurations specifically tailored for deep learning inference, further amplifying its performance advantage over single-board computers.

4.2.5 Performance Limitations of Jetson Nano

The NVIDIA Jetson Nano was chosen for its strong performance, energy efficiency, and suitability for real-time edge computing. Its ability to optimize deep learning models allows it to maintain high inference speeds with low power consumption, ideal for IoT applications like warehouse monitoring. Benchmarks show that optimized models on the Nano achieve a 16.11% improvement in inference speed, significantly benefiting tasks such as image classification and video action detection [19]. For real-time, video-based anomaly detection, the Jetson Nano reached 47.56 frames per second (FPS) using just 3.11 GB of RAM, outperforming previous Jetson models by 15% while consuming 50% less energy [20].

The Jetson Nano offers a strong balance of performance and energy efficiency for edge computing, making it suitable for applications focused on cost, power efficiency, and real-time performance. In this project, it effectively meets warehouse needs. For larger or more complex tasks, upgrades may include higher-capacity edge devices or a hybrid edge-cloud model for local processing supported by the cloud.

## 5. System Implementation

The implementation process is divided into four phases:

- Preprocessing

- Yolov8 Model
- Training Phase
- Inference Phase

Each phase will be thoroughly discussed.

### 5.1. Dataset

The research utilized a carefully constructed dataset comprising a diverse array of medicinal products, including tablets and medicine boxes. A total of 2,630 images of 14 different classes were annotated and categorized into relevant classes, involving the collection and correct labeling of these images. Python is used to program the required software to collect images from Egyptian pharmacy websites, along with additional information as shown in Figure 5. The dataset included various categories of medicinal products under investigation. To enhance the YOLO model's performance, the dataset was divided into three distinct sets: 70% for training, 15% for validation, and 15% for testing.

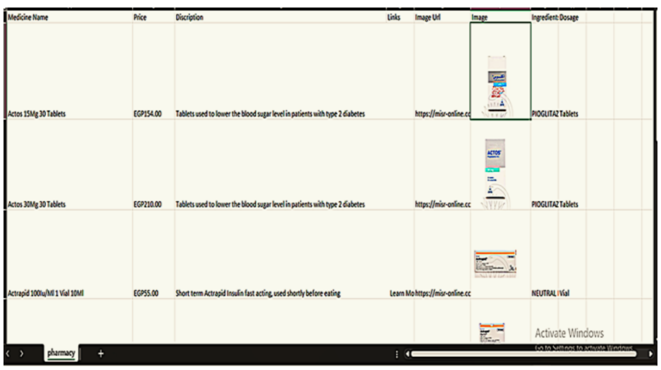


**Figure 5. Sample Dataset Snippet**

For the system implementation, we built a dataset to train the model. Photos of 14 different medicine boxes were taken. These photos captured the medicines' boxes in various conditions: alone, side by side, with variable angles, and under three conditions of lighting "White Light, Orange Light, Both Lights On (Warm)", as shown in Figure 6.

**Figure 6. Photos in different Conditions**

### 5.2. Preprocessing

Preprocessing is crucial for successful data analysis projects, transforming data into a clean format for machine analysis. Several preprocessing techniques include:

- Normalization treats all image features equally during training, commonly normalizing pixel values between 0 and 1 or -1 and 1 to avoid domination by larger-scale features.
- Resizing makes photos compatible with the Yolov8 model, requiring 640x640 images.
- Data augmentation tweaks photos to increase dataset variance, preventing overfitting and enhancing model accuracy.

#### 5.2.1 Resizing

Image resizing reduces training time by lowering pixel count without affecting accuracy, making it suitable for the real-time applications. Photos were taken at (3024x3024) pixels and resized to (640x640), as shown in Figure 7, before augmentation, reducing dataset size and easing training.


**Figure 7. Resizing Photo**

#### 5.2.2 Data Augmentation

Image augmentation expands the dataset, improving model accuracy by countering overfitting. Various geometric transformation-based methods were used, including Flipping, Brightness, Blur, Shear, Noise, and Saturation. [21].

Flipping: Images can be flipped horizontally or vertically by rotating them 90 or 180 degrees as shown in Figure 8.

**Figure 8. Vertical and Horizontal Flip**

Flipping augmentation helps the model recognize medicines in various orientations, increasing accuracy.

Brightness: Image brightness is randomly altered by 25% to add variance to the dataset. Figure 9 shows the brightness changes.


**Figure 9. Brightness changes to the data**

This trains the model to handle different lighting conditions, improving versatility and accuracy.

Blur: Adding blur helps the model cope with real-world imperfections, like out-of-focus images or poor lighting as presented in Figure 10.


**Figure 10. Blur added to the Photos**

Shear: It tilts the image along one axis, controlled by a shear angle as shown in Figure 11.


**Figure 11. Shear added to the Photos**

This mimics real-world tilts, increasing accuracy by training the system with various angles.

Noise: Adding noise makes photos more realistic, simulating real-life variations. The noise added to the images in Figure 12 involves intensity modulation that replicates real-world environmental variations, specifically Brightness Nois where the images were modified with brightness adjustments, as seen with -14% and +14% intensity changes. This simulates conditions like overexposure or dim lighting.

The aim is to enhance the model's ability to handle diverse lighting scenarios during inference, improving its generalization capabilities in real-world applications [22].

Figure 12 presents one photo of the dataset before and after adding noise.

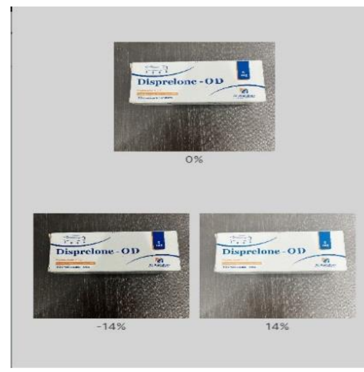


**Figure 12. Noise added to the Photos**

Saturation: It controls color intensity, with low saturation making colors dull and high saturation making them intense, as shown in Figure 13.

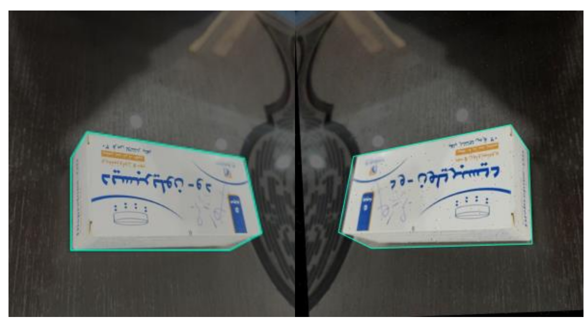

**Figure 13. Saturation Added in the Images**

Adjusting saturation helps the system identify medicines despite color variations, increasing accuracy.

### *5.3. Yolov8 Model*

The Yolov8 architecture consists of:

- Backbone [18]: The backbone utilizes a pre-trained CNN, such as CSPDarknet53, to progressively extract pertinent features from the input image through its convolutional layers, capturing low, mid, and high-level features [23].
- Neck [18]: An optional component integrates feature maps from various stages of the backbone, frequently employing

a "Spatial Pyramid Pooling (SPP)" layer for path aggregation.

- Head [18]: It generates final predictions by taking the processed feature maps from the backbone and predicting bounding boxes and class probabilities. It consists of multiple convolutional layers followed by fully connected layers.

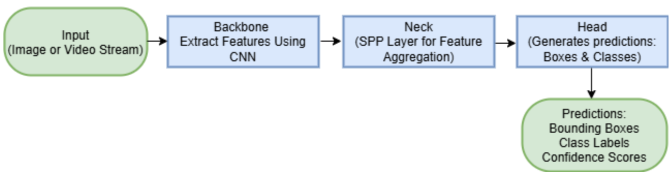


**Figure 14. YOLOv8 architecture**

*5.4. The Training Phase*

It involves:

- Data Preparation: Images of medicines were captured, augmented, and labeled.
- Configuration: A YAML file customizes parameters like optimizer, learning rate, batch size, and epochs.
- Training Loop: The model predicts bounding boxes, computes class probabilities, and updates weights based on loss.
- Monitoring and Evaluation: A separate validation dataset or YOLOv8 visualizations assess generalization.
- Saving the Model: The final model's weights are saved for object detection during inference.

*5.5. The Inference Phase*

It involves:

- Preprocess Input: The image resizes or converts it into a specific color space and normalizes its pixel values.
- Forward Pass: Feed preprocessed image through the Yolov8 model to generate the bounding boxes and confidence scores.
- Decode Outputs: Translate the raw outputs into bounding box coordinates and confidence scores for the application.

*5.6. Storage Management*

It is crucial for training and deployment. Yolov8 does not have built-in storage management features, but it is applied in two phases:

5.6.1 Training Phase:

- Model Size: Larger models provide higher accuracy but need more disk space.
- Dataset Storage: High-resolution images enhance performance but necessitate augmentation to minimize storage.
- Checkpointing: Regular checkpoints preserve the model's state, enabling training resumption after interruptions, balancing storage space with efficiency.

5.6.2 Deployment Phase:

- Model Deployment: Decide to store either the full model or a pruned version, which reduces size by eliminating redundant parts.
- Inference Batching: Process multiple images in batches during inference while balancing batch size with available memory.

## 6. Results

The model was trained on Google Colab Pro using powerful GPUs, utilizing a dataset of manually annotated images captured in varying lighting conditions to reflect real-life scenarios. Data augmentation enhanced accuracy, and training involved a trial-and-error approach for parameters and epochs, taking six and a half hours for 20 epochs. Key metrics include:

- Mean Average Precision (mAP)
- Box Loss
- Class Loss
- Distribution Focal Loss (DFL)

*6.1. Mean Average Precision (mAP)*

The assessment of object detection performance involves calculating Average Precision (AP) for each class, highlighting the trade-off between precision and recall. Recall indicates the model's effectiveness in identifying all relevant cases, measuring the correct positive predictions against all actual positive instances [24].

Precision measures how accurate a model's positive predictions are. It shows how many of the predicted positive cases were actually correct [24].

Equations for Recall (R) and Precision (P) in equation (2) and (3):

$$R = TP / (TP + FN) \quad (2)$$

$$P = TP / (TP + FP) \quad (3)$$

Where:

True Positive (TP): The model accurately predicted a positive outcome.

False Negative (FN): The model mistakenly predicted a negative outcome instead of positive.

False Positive (FP): The model incorrectly predicted a positive outcome when it should have been negative. The results are illustrated in the screenshot in Table 4.

**Table 4. Metrics for object detection per class**

| Class | Images | Instances | Box(P) | mAP50 | mAP50-95 |
|---|---|---|---|---|---|
| Antodine | 13 | 31 | 1 | 0.995 | 0.913 |
| Aspicarlo | 26 | 36 | 0.997 | 0.995 | 0.963 |
| Chlorosec | 15 | 20 | 0.997 | 0.995 | 0.946 |
| Concor | 15 | 20 | 0.977 | 0.995 | 0.936 |
| Controloc | 14 | 68 | 0.998 | 1.000 | 0.926 |
| Convetin | 12 | 12 | 0.998 | 0.995 | 0.916 |
| Disprelone-OD | 12 | 12 | 0.994 | 0.994 | 0.914 |
| Epicopred | 25 | 25 | 0.997 | 0.995 | 0.904 |
| Gastrodomina | 22 | 22 | 0.995 | 1.000 | 0.914 |
| Maalox | 13 | 25 | 0.995 | 0.995 | 0.942 |
| Osteocare | 11 | 35 | 0.998 | 0.995 | 0.902 |
| Rivarospire | 25 | 73 | 0.999 | 0.999 | 0.911 |
| Vidrop-Adult | 21 | 47 | 0.999 | 0.995 | 0.916 |
| Vidrop-Kids | 12 | 32 | 0.998 | 0.995 | 0.907 |

This table includes the number of images, instances, precision (Box(P)), and mean Average Precision (mAP) at IoU thresholds 0.5 and 0.5:0.95. mAP50-95 evaluates the model's ability to detect objects with varying overlap (IoU) with ground truth bounding boxes. This metric is more rigorous than standard mAP, assessing performance across different levels of object detection difficulty in real time.

### *6.2. Training set Box Loss*

Box Loss measures the algorithm's ability to locate an object's center and cover it with a bounding box, calculated using Mean Square Error (MSE) as shown in Figure 15.

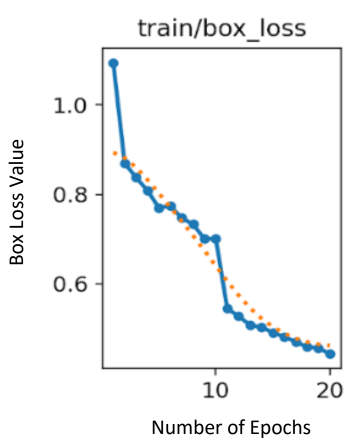


**Figure 15. Graph of the Box Loss**

### *6.3. Training set Class Loss*

Figure 16 shows the evolution of classification loss during training, which is calculated using Cross-Entropy Loss. This loss measures the dissimilarity between the predicted probability distribution and the true class labels, and is simply defined as in equation (4) [25]:

$$Loss = -\sum_{i=1}^{C} y_i \,.\, log\,(\hat{y}_i) \tag{4}$$

Where:

$C$ is the total number of classes

$y_i$ is a binary indicator (0 or 1) that specifies whether class $i$ is the correct class.

$\hat{y}_i$ is the predicted probability for class $i$.

The steady decrease in classification loss indicates improved alignment between predictions and true labels, demonstrating the model's enhanced classification accuracy over the epochs.

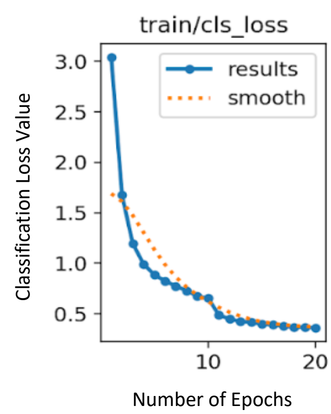


**Figure 16. Graph of the Class Loss**

The graph in Figure 16 (train/cls_loss) represents the classification loss during training, which measures the model's ability to correctly classify objects within the predicted bounding boxes. This loss is calculated using Cross-Entropy Loss, which quantifies the difference between the predicted probability distribution of the model and the true class labels.

The decreasing trend in the graph indicates that the model is improving its classification accuracy over the epochs as the loss decreases.

### *6.4. Training set Distribution Focal Loss*

Focal Loss addresses the imbalances between foreground and background classes during training [24]. Distribution Focal Loss focuses on learning probabilities around bounding box coordinates, improving accuracy. The distribution is illustrated in Figure 17.

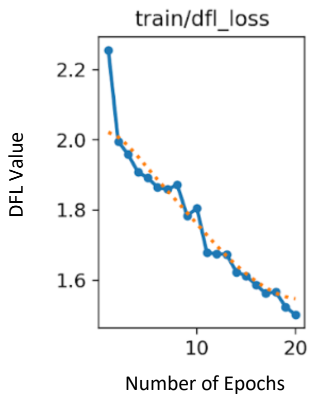


**Figure 17. Graph of the Distribution of Focal Loss**

*6.5. Model Results Discussion*

After presenting the graphs, a live demonstration of the system in operation should be included to fully convey the envisioned functionality of the system. Images illustrating system detection accuracy are shown in Figure 18.


**Figure 18. Validation Set Predictions**

Figure 18 illustrates the system's effectiveness in accurately detecting and classifying medicinal products in the validation dataset. The bounding boxes and high confidence scores (mostly above 0.9) demonstrate the robust performance of the YOLOv8 model, even under challenging conditions such as varying orientations, lighting, and partial occlusions. The system successfully identifies multiple products, including Antodine, Chlorosec, and Disprelone-OD, showcasing its reliability for inventory management in medicinal warehouses. While a few detections exhibit slightly lower confidence (e.g., 0.7), the results overall highlight the system's practicality and potential for enhancing warehouse efficiency by preventing stock discrepancies and ensuring accurate tracking.

The implementation of the YOLOv8 model for medicinal warehouse surveillance and organization has demonstrated significant success in object detection and classification tasks. The training phase, carried out on Google Colab Pro with a dataset constructed from manually annotated images under varied lighting conditions, has yielded promising results.

Key metrics, including Mean Average Precision (mAP), Box Loss, Class Loss (CLS), and Distribution Focal Loss (DFL), indicate the model's strong performance. The mAP score of 0.923 in mAP50-95 reflects the model's high accuracy and robustness even under challenging detection conditions.

The validation loss curves, DFL and CLS, shown in Figure 19, further emphasize the model's efficiency:

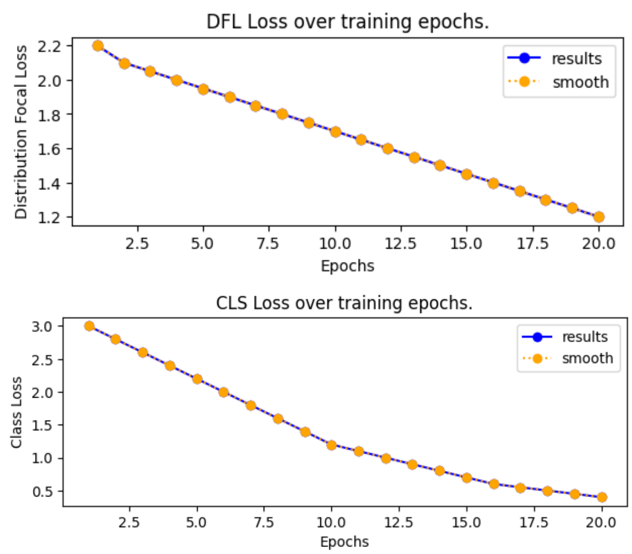


**Figure 19. Validation Set Loss Curves (DFL and CLS)**

Distribution Focal Loss (DFL): The graph of Distribution Focal Loss shows a consistent decrease over the epochs, reflecting the model's enhanced ability to learn probabilities related to bounding box coordinates. The reduction from approximately 2.2 to 1.6 indicates significant improvement in the model's training process.

Class Loss (CLS): The Class Loss graph shows a sharp decline from around 3.0 to approximately 0.5 over 20 epochs, indicating improved object classification by the model. During inference, the YOLOv8 model effectively processes and predicts bounding boxes and confidence scores, showcasing its real-time application potential for medicinal warehouse management. The deployment strategy, considering model size, dataset storage, and checkpointing, optimizes the balance between accuracy and resource efficiency.

*6.6. FaceNet Model*

After fine-tuning, the FaceNet model achieved a face verification accuracy of 98% on the validation set, with minimal false positives and negatives. The model's robustness was tested under various lighting and pose conditions, demonstrating its reliability in ensuring secure employee access to restricted warehouse areas.

### *6.7. Stock Prediction Module*

The dataset used in this study was obtained from a private pharmacy and contained 1,825 records of historical inventory data for five medicinal products, spanning one year (2023-01-01 to 2023-12-31). Each record included daily stock levels, sales, restocking activities, and seasonal demand patterns. Using this dataset, a Long Short-Term Memory (LSTM) model was trained to predict the future stock levels for the product "Antodine" As an example. The model was trained on 80% of the data, while the remaining 20% was used for testing. The results, shown in Figure 18, demonstrate the model's ability to capture stock trends, with predicted values aligning closely with actual stock levels. The model achieved a Mean Squared Error (MSE) of 849.40798 and a Mean Absolute Error (MAE) of 23.5485798, indicating an accuracy of approximate 92.15% for trend prediction. Figure 20 illustrates the comparison of actual and predicted ending stock levels for the product "Antodine" using the LSTM model. The predictions provide valuable insights into inventory trends, enabling proactive restocking and minimizing shortages or overstocking.

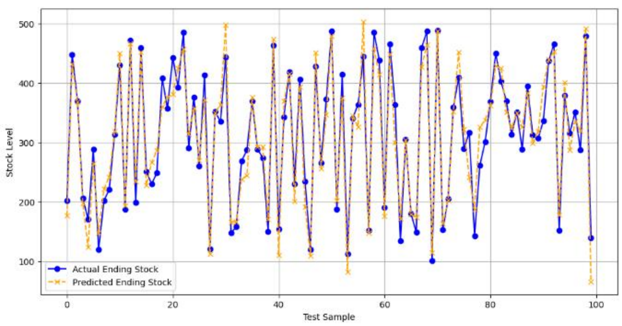


**Figure 20. LSTM Model: Actual vs. Predicted Ending Stock**

### *6.8. Integration of YOLOv8 and FaceNet*

The integration of YOLOv8 and FaceNet in the proposed system creates a comprehensive solution for warehouse monitoring and security. Each component complements the other to address specific challenges in medicinal warehouse management:

**Object Detection and Inventory Management:**

YOLOv8 is responsible for detecting and identifying medicine boxes in real-time. Its high accuracy (mean Average Precision of 92.15%) ensures reliable inventory tracking, even in dynamic conditions. The use of data augmentation and optimized hardware (Jetson Nano) further enhances detection robustness and cost efficiency.

**Access Control and Security:**

FaceNet adds an additional layer of security by verifying employee identities through facial recognition. By restricting unauthorized access, it prevents theft and ensures accountability for sensitive inventory. Fine-tuning FaceNet with the warehouse-specific dataset resulted in a high verification accuracy of 98%, enabling reliable real-time authentication.

**Combined Functionality:**

Integrating YOLOv8 with FaceNet automate inventory tracking and access control, providing a unified framework for improving warehouse efficiency and security. While YOLOv8 identifies and counts medicine boxes, FaceNet ensures that only authorized personnel interact with them, reducing the risk of errors, theft, and unauthorized access.

## 7. Scalability and Real-World Implementation Considerations

While the proposed system has shown success in smaller and medium-sized warehouses, scaling it for larger facilities poses challenges. Larger warehouses with higher inventory volumes and complex layouts may need more powerful hardware than the Jetson Nano to efficiently manage increased data loads. Although the Jetson Nano performs well under typical conditions, real-time performance may decline with more sensors, cameras, and items to track. Future implementations could consider upgrading to more powerful edge computing devices, such as the NVIDIA Jetson AGX Xavier (32 TOPS), Google Coral Dev Board (4 TOPS), or Khadas VIM3 Pro (5 TOPS), to handle larger workloads and ensure scalability [26], [27], [28], [29].

## 8. Limitations

The accuracy and reliability of ultrasonic sensors and cameras in our warehouse monitoring system are influenced by environmental factors like lighting and sensor angles. We optimized performance by testing under various lighting conditions and angles, identifying placements to minimize blind spots. To enhance the dataset, we used image augmentation techniques to capture medicine boxes from multiple perspectives, increasing its size and helping the YOLOv8 model reduce false positives and negatives. YOLOv8 also showed improved accuracy with post-training quantization. Regular sensor calibration is crucial for consistent detection, as temperature and humidity affect readings. Automated recalibration protocols improve stability, ensuring reliable performance in dynamic warehouse conditions.

## 9. Conclusion and Future Work

Implementing the proposed system requires a significant initial investment for the Jetson Nano, cameras, sensors, and software, which may be a barrier for smaller warehouses. However, benefits like reduced theft losses and improved inventory management can offset these costs over time. Ongoing expenses, such as software updates and maintenance, are essential for optimal performance and longevity. For larger warehouses, industrial-grade sensors enhance stability and accuracy but come at a higher cost.

Future system iterations could explore cost-reduction strategies, such as using less expensive hardware or lightweight sensor technologies to improve accessibility for smaller operations. Additionally, leveraging cloud-based solutions or optimizing deep learning models for edge devices could further enhance scalability and affordability.

This study demonstrated the effectiveness of the YOLOv8 model and FaceNet in enhancing surveillance and organization in medicinal warehouses. Through careful data preparation and monitoring, we achieved a mean Average Precision (mAP) of 0.923, alongside reductions in Box Loss, Class Loss, and Distribution Focal Loss (DFL), indicating strong model performance. The training and validation loss curves showed no overfitting, with consistent decreases in DFL and Class Loss, suggesting good generalization to new data.

Integrating YOLOv8 and FaceNet for object and facial recognition in medicinal warehouses presents a practical approach to inventory management. By using a specialized dataset, this system addresses critical challenges like unauthorized access and stock shortages, offering reliable and precise capabilities for real-world applications.

**References**


[1] Eigbokhan Gilbert Ogbewele, Akachukwu Obianuju Mbata, and Nelly Tochi Nwosu, "Optimizing pharmaceutical inventory management: A global framework for efficiency and cost reduction," *International Journal of Management & Entrepreneurship Research*, vol. 6, no. 10, pp. 3357–3371, Oct. 2024, doi: 10.51594/IJMER.V6I10.1638.

[2] S. L. Kakade, "Inventory Management in Pharmaceutical Industry," *INTERANTIONAL JOURNAL OF SCIENTIFIC RESEARCH IN ENGINEERING AND MANAGEMENT*, vol. 08, no. 04, pp. 1–5, Apr. 2024, doi: 10.55041/IJSREM30239.

[3] P. Duan and X. Liang, "An Improved YOLOv8-Based Foreign Detection Algorithm for Transmission Lines," *Sensors*, vol. 24, no. 19, Oct. 2024, doi: 10.3390/S24196468.

[4] "Analysis of inventory management in pharmaceutical sector: a review paper," *International Journal of Development Research*, pp. 62800–62805, May 2023, doi: 10.37118/IJDR.26699.05.2023.

[5] A. Biglari and W. Tang, "A Review of Embedded Machine Learning Based on Hardware, Application, and Sensing Scheme," *Sensors*, vol. 23, no. 4, p. 2131, Feb. 2023, doi: 10.3390/s23042131.

[6] K. Ijaz, Z. Hussain, J. Ahmad, S. Ali, … M. A.-I., and undefined 2022, "A novel temporal feature selection based LSTM model for electrical short-term load forecasting," *ieeexplore.ieee.org*, Accessed: Jan. 25, 2025. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9849665/

[7] J. Terven, D. M. Córdova-Esparza, and J. A. Romero-González, "A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS," *Mach Learn Knowl Extr*, vol. 5, no. 4, pp. 1680–1716, Dec. 2023, doi: 10.3390/make5040083.

[8] E. L. Arjun *et al.*, "Advanced Face Authentication Using Deep Learning Models," *2023 IEEE Pune Section International Conference, PuneCon 2023*, 2023, doi: 10.1109/PUNECON58714.2023.10450013.

[9] E. Irfan, C. Jacob, and R. Resmi, "Facial Recognition and CCTV Integration for Enhanced Security Using Deep Learning Techniques," *RAICS - IEEE Recent Advances in Intelligent Computational Systems*, no. 2024, 2024, doi: 10.1109/RAICS61201.2024.10689986.

[10] W. Chen, H. Huang, S. Peng, C. Zhou, and C. Zhang, "YOLO-face: a real-time face detector," *Vis Comput*, vol. 37, no. 4, pp. 805–813, Apr. 2021, doi: 10.1007/s00371-020-01831-7.

[11] W. Yang and Z. Jiachun, "Real-time face detection based on YOLO," in *2018 1st IEEE International Conference on Knowledge Innovation and Invention (ICKII)*, IEEE, Jul. 2018, pp. 221–224. doi: 10.1109/ICKII.2018.8569109.

[12] B. P. Prayogo, Hendrawan, E. Mulyana, and W. Hermawan, "A Novel Approach for Face Recognition: YOLO-Based Face Detection and Facenet," in *2023 9th International Conference on Wireless and Telematics (ICWT)*, IEEE, Jul. 2023, pp. 1–6. doi: 10.1109/ICWT58823.2023.10335263.

[13] M. S. Abdelfattah, A. Mehrotra, Ł. Dudziak, and N. D. Lane, "ZERO-COST PROXIES FOR LIGHTWEIGHT NAS," in *ICLR 2021 - 9th International Conference on Learning Representations*, International Conference on Learning Representations, ICLR, 2021.

[14] E. Chebotareva, A. Toschev, and E. Magid, "Comparative analysis of neural network models performance on low-power devices for a real-time object detection task," *Computer Optics*, vol. 48, no. 2, pp. 242–252, 2024, doi: 10.18287/2412-6179-CO-1343.

[15] M. Safaldin, N. Zaghden, and M. Mejdoub, "An Improved YOLOv8 to Detect Moving Objects," *IEEE Access*, vol. 12, pp. 59782–59806, 2024, doi: 10.1109/ACCESS.2024.3393835.

[16] I. Lazarevich, M. Grimaldi, R. Kumar, S. Mitra, S. Khan, and S. S. Deeplite, "YOLOBench: Benchmarking Efficient Object Detectors on Embedded Systems."

[Online]. Available: https://github.com/Deeplite/deeplite-

[17] S. Bemposta Rosende, S. Ghisler, J. Fernández-Andrés, and J. Sánchez-Soriano, "Implementation of an Edge-Computing Vision System on Reduced-Board Computers Embedded in UAVs for Intelligent Traffic Management," *Drones*, vol. 7, no. 11, p. 682, Nov. 2023, doi: 10.3390/drones7110682.

[18] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," Apr. 2018, [Online]. Available: http://arxiv.org/abs/1804.02767

[19] T. P. Swaminathan, C. Silver, and T. Akilan, "Benchmarking Deep Learning Models on NVIDIA Jetson Nano for Real-Time Systems: An Empirical Investigation," Jun. 2024, [Online]. Available: http://arxiv.org/abs/2406.17749

[20] N. G. Pham, N. L. Bui, Q. H. Tran, L. T. H. Tong, and H. N. Tran, "AN EFFECTIVE APPROACH TO FACE RECOGNITION WITH ARTIFICIAL INTELLIGENCE AND THE INTERNET OF THINGS USING NVIDIA JETSON NANO," *Dalat University Journal of Science*, pp. 39–64, Sep. 2024, doi: 10.37569/DalatUniversity.14.3.1262(2024).

[21] A. Mikolajczyk and M. Grochowski, "Data augmentation for improving deep learning in image classification problem," in *2018 International Interdisciplinary PhD Workshop (IIPhDW)*, IEEE, May 2018, pp. 117–122. doi: 10.1109/IIPHDW.2018.8388338.

[22] Z. Zhang, H. Zhou, and Z.-Q. J. Xu, "Dropout in Training Neural Networks: Flatness of Solution and Noise Structure," Nov. 2021, [Online]. Available: http://arxiv.org/abs/2111.01022

[23] B. Li, Q. Meng, X. Li, Z. Wang, X. Liu, and S. Kong, "Enhancing YOLOv8's Performance in Complex Traffic Scenarios: Optimization Design for Handling Long-Distance Dependencies and Complex Feature Relationships," *Electronics (Switzerland)*, vol. 13, no. 22, Nov. 2024, doi: 10.3390/ELECTRONICS13224411.

[24] G. S. Tran, T. P. Nghiem, V. T. Nguyen, C. M. Luong, and J.-C. Burie, "Improving Accuracy of Lung Nodule Classification Using Deep Learning with Focal Loss," *J Healthc Eng*, vol. 2019, pp. 1–9, Feb. 2019, doi: 10.1155/2019/5156416.

[25] L. Li, M. Doroslovački, M. L.-I. access, and undefined 2020, "Approximating the gradient of cross-entropy loss function," *ieeexplore.ieee.org*, Accessed: Jan. 25, 2025. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9113308/

[26] N. NVIDIA, "Jetson nano developer kit." Accessed: Oct. 29, 2024 [Online]. Available: https://developer.nvidia.com/embedded/learn/get-started-jetson-nano-devkit

[27] D. Nvidia, "Jetson AGX Xavier developer kit." Accessed: Oct. 29, 2024. [Online]. Available: https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-agx-xavier/

[28] M. Google, "Coral Dev Board." Accessed: Oct. 29, 2024. [Online]. Available: https://coral.ai/products/dev-board/

[29] "VIM3 | Khadas." Accessed: Oct. 29, 2024. [Online]. Available: https://www.khadas.com/vim3

**Abbreviation and symbols**

| | |
|---|---|
| **YOLO** | **You Only Look Once** |
| **CNN** | **Convolutional Neural Network** |
| **mAP** | **Mean Average Precision** |
| **IoU** | **Intersection over Union** |
| **IoT** | **Internet of Things** |
| **GDPR** | **General Data Protection Regulation** |
| **FPS** | **Frames Per Second** |
| **LSTM** | **Long Short-Term Memory** |
| **RAM** | **Random Access Memory** |
| **GPU** | **Graphics Processing Unit** |
| **YOLO** | **You Only Look Once** |