International Journal of Applied Information Systems (IJAIS) – ISSN : 2249-0868
Foundation of Computer Science FCS, Delaware, USA
Volume 13– No. 2, February 2026 – www.ijais.org

A Mask RCNN-based Bottle Recognition and Counting System for Retail Shelf Intelligence

Cesar Robles
International Brewery PLC, Nigeria

Temitope Oladokun
International Brewery PLC, Nigeria

Chukwuemeka C. Ugwu
Department of Data Science, The Federal University of Technology, Akure, Nigeria

Gbolahan Kolawole
International Brewery PLC, Nigeria

Taiwo Adelakin
International Brewery PLC, Nigeria

Saleem Adebayo
International Brewery PLC, Nigeria

ABSTRACT
Accurate identification and counting of bottles are important for effective inventory management in bottling operations. However, many bottling operations still rely on manual audits and conventional object detection techniques that struggle in visually complex environments. In this study, deep learning-based computer vision approach is proposed to enable precise and automated real-time bottle identification and counting. An automated system was developed using Mask R-CNN-based instance segmentation approach. The methodology includes the collection of chiller bottle images, image pre-processing, mask RCNN model training, evaluations, and deployment. The performance of the proposed model was compared with faster RCNN using two backbone networks, ResNet-50 and ResNet-101. The result obtained shows that mask RCNN outperformed faster RCNN in extracting region of interest (ROI) extraction, with significant difference in average precision of 5.07% and 2.08% for the ResNet50 and ResNet101 respectively. These findings show the effectiveness of Mask R-CNN for accurate bottle detection and counting in automated bottling inventory systems.

Keywords
Region of Interest, Convolution Neural Network, Recurrent CNN, Deep learning

1. INTRODUCTION
Effective inventory management plays a vital role in many industries by helping organizations maintain appropriate stock levels, reduce waste, and improve overall profitability [1]. When inventory is well managed, firms are better positioned to make informed decisions, control operational costs, and respond quickly to changing market demands [2]. This is particularly important in sectors with rapid stock turnover, such as retail, manufacturing, and bottling, where even minor inefficiencies can lead to significant losses [3][4]. Through consistent tracking of goods and well-structured replenishment processes, companies can limit stockouts and overstocking, reduce production disruptions, and ultimately improve customer satisfaction [5].

In bottling companies, inventory management goes beyond production facilities to real-time monitoring in retail environments. Field marketers play a critical role in ensuring product availability, optimizing shelf presence, and analyzing competitor activity [6][7]. A key component of this process is bottle counting and tracking, which offers detailed insights into SKU movement and shelf dynamics. Automated systems increasingly complement field marketers' efforts by reducing the reliance on manual audits and providing real-time, unit-level visibility, improving shrinkage detection and forecasting.

Traditionally, bottle counting relied on manual audits where operators manually count and sort the bottles [8][9]. This process has been accurate, but as operations scaled, manual process became inefficient and error-prone, thereby ushering in the era of bottle counting automation. The automation of bottle counting using computer vision (CV), machine learning (ML), and IoT technologies has transformed manual tasks into a real-time, high-precision operation [10][11]. Early CV methods include laplacian of gaussian (LoG), prewitt operator for edge detection; Harris corner, features from accelerated segment test for corner and interest point detection; Scale-Invariant Feature Transform (SIFT) and Speeded-Up Robust Features (SURF) for feature descriptors among others. The limitations of these methods spans from computation intensiveness (SIFT/SURF), to sensitivity to noise and lighing variations (LoG). The combination of machine learning with other conventional CV methods for feature matching and classification tends to address these limitations by introducting structure, robustness, object-level discrimation, and semantic meaning [12][13]14]. However, their relaince on manually designed features constrains their performances in complex and dynamic environments.

Deep learning emerged in computer vision as a data-driven alternative to handcrafted feature-based methods, enabling automatic hierarchical feature learning and end-to-end optimization. Through convolutional neural networks, deep learning models achieve superior robustness and accuracy in complex visual environments, making them well suited for real-world applications such as automated bottle identification and inventory management.

Early deep learning–based object detectors, such as region-based convolutional neural networks (R-CNN) and their successors, introduced the concept of learning region proposals and object classifications within a unified framework. Faster R-CNN further enhanced this approach by integrating a region proposal network (RPN), enabling efficient and accurate object localization. While Faster R-CNN marked a major advancement over traditional methods, its reliance on bounding-box-based detection limited its effectiveness in scenarios involving overlapping objects and cluttered backgrounds, common characteristics of retail shelves and bottling lines.

Mask R-CNN was introduced to address these limitations by extending Faster R-CNN with an additional instance segmentation branch. This segmentation head predicts a pixel-level mask for each detected object, enabling precise delineation of object boundaries rather than coarse rectangular localization. For bottle identification and counting, this capability is particularly important, as it allows individual bottles to be separated even when they are closely packed or partially occluded. By isolating object pixels from the background, Mask R-CNN improves region-of-interest extraction, reduces false detections, and enhances counting accuracy

Motivated by the limitations of conventional feature-based methods and the inability of bounding-box detectors to reliably separate densely packed objects, this study adopts Mask R-CNN to achieve accurate instance-level bottle identification and counting in real-world bottling environments.

2. LITERATURE REVIEW

Several literature have carried out in improving object detection models in inventory managements. The authors in [16] developed a bottle counting system that applied depth image processing with a Kinect-2 sensor to capture 3D images of bottles on a recycling line. The method utilized calibration, morphological operations, and connected component analysis to achieve an accuracy of 99%. However, the developed model faced adaptability issues due to its reliance on specific hardware and sensitivity to changes in operational conditions. Similarly, [14] presented an intelligent sorting system that used computer vision to detect and classify cans and bottles, achieving a 91.33% identification accuracy. However, its effectiveness diminished when handling diverse object shapes and sizes.

In [17], a two-stage tracking system for crate and bottle detection achieved 99.9% accuracy but introduced computational complexity that may not suit high-speed production environments. Other works employed deep learning-based object detection models, particularly the convolution neural network (CNN) models [15]. CNN models have proven to be effective with variants like faster RCNN, Mask RCNN, YOLO and more, high level feature extractions from images, and turning raw visual data into actionable insights where made possible. [18], applied faster R-CNN on a football dataset for improved detection. though performance on small objects was limited. Enhancements like batch normalization were incorporated into SSD models [19], improving mAP scores but showing reduced generalization with small batch sizes. Meanwhile, models combining YOLO-v3 [20] and ResNet [21] achieved moderate precision, while adaptations of YOLO-v3 for lane detection [22] demonstrated improvements by modifying grid density and detection scales.

The authors in [23] compared classical machine learning and deep learning models for UAV images and proposed lightweight detection models using techniques like convolution decomposition and model pruning, but Mask R-CNN was not considered in their analysis. In [24], the authors presented the deployment of RetinaNet and PointPillars on edge AI platforms, emphasizing runtime evaluation but overlooked the development phase.

[25] developed and compared YOLOv6 and YOLOv7 models for plastic bottle waste detection model on the water surface. The authors applied FloW-Img images. The dataset consists of 500,000 bottle objects in 2,000 images. The YOLOv6 and YOLOv7 models were evaluated using mAP and running time. The results showed that YOLOv6 and YOLOv7 can handle bottle waste detection well, with mAP values of 0.873 and 0.512, respectively. In addition, YOLOv6 (4.21 m/s) has a higher detection speed than YOLOv7 (13.7 m/s). However, when it comes to accuracy and precise understanding of object shapes and instances at the pixel level, Mask RCNN performs better.

Similarly, [26] also worked on an automated approach to waste counting, focusing on using YOLOv8 object detection model to acquire accurate counts of plastic bottles as they advent down rivers and streams but encountered high false positives which were filtered out by post-processing algorithm.

[27] and [28] worked on bottle level fill detection. While [28] applied image processing with machine learning, [27] utilized sequential neural network model for its bottle level fill detection. However, both studies have limited restricted adaptability and generalization as neither approach addresses varying bottle shapes, occlusions, real-time processing constraints, or deployment challenges in unconstrained production environments

Across these works, it is evident that advancements in both region-based networks like R-CNN and its successors (Faster R-CNN, Mask R-CNN) and regression-based networks like YOLO and SSD have substantially improved detection accuracy and reduced inference time. However, region-based models maintain a consistent edge in detection accuracy, especially with improvements in architectures like Faster R-CNN and Mask R-CNN. These strengths, combined with Mask R-CNN's segmentation capabilities, make it a compelling choice for real-time object detection tasks in bottling environments. Persistent challenges such as background noise and poor detection head designs remain areas of concern that must be addressed for optimal performance.

Given these motivations, this study aims to develop an automated bottle recognition and counting system using Mask R-CNN. Specifically, it seeks to evaluate the model's effectiveness in improving detection accuracy and handling typical challenges in bottling inventory management, ultimately supporting better stock tracking and real-time field marketing insights

3. METHODOLOGY

Figure 1 shows the architecture of the system. The system is basically in 5 phases namely, the data source and collection, data pre-processing, model training, inference, and the deployment phase

3.1 Data Source and Collection

One of the fundamental pillars of a successful computer vision research is the availability of high-quality data to ensure good performance and generalizability of computer vision models. In this study we employed an in-house data collection strategy as data were collected using cameras and as well as leveraging existing data. The images captured are composed of different bottle objects in a chiller. A total of 405 images were collected.

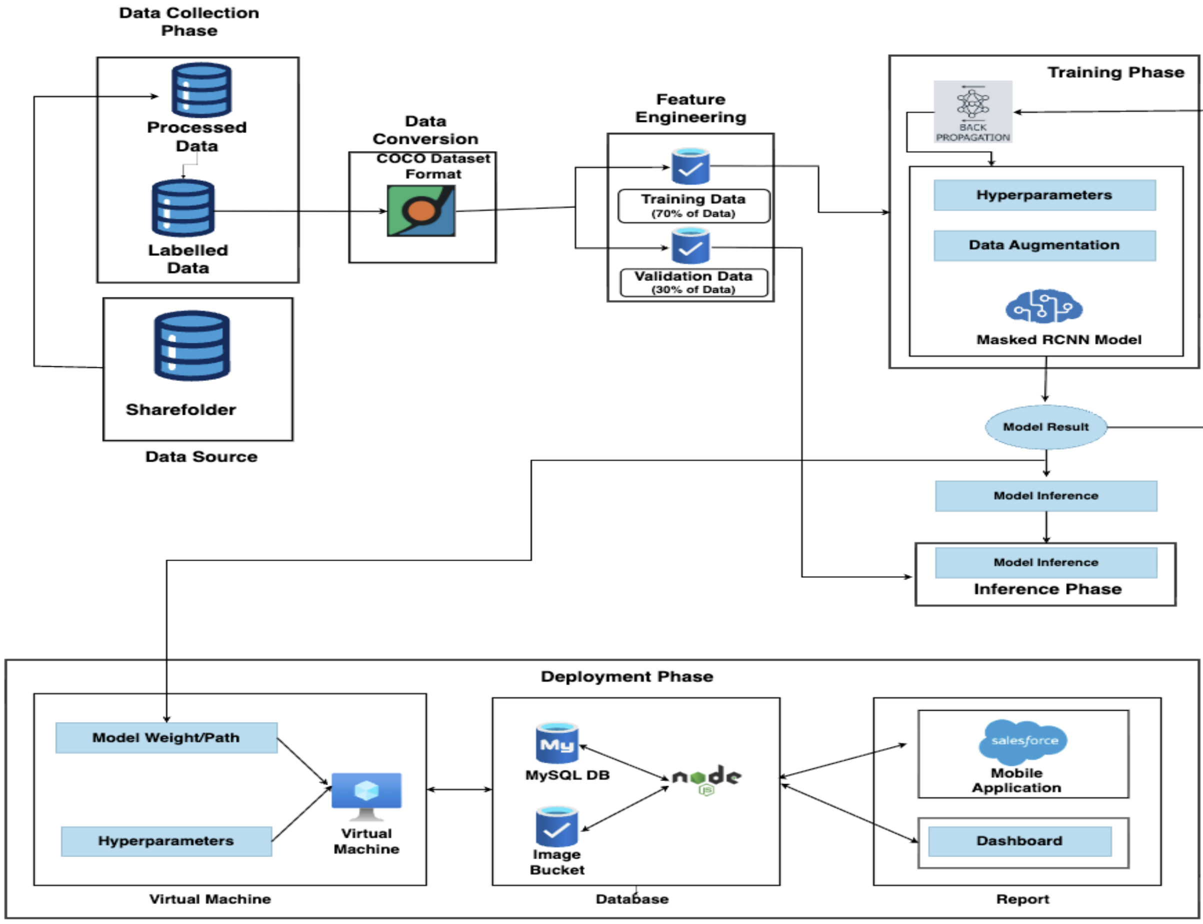

Fig 1: Functional System Architecture

3.2 Data Preprocessing

This phase revolves around converting raw data into a format conducive to training and evaluation. This process encompasses several key steps: Image resizing, Selection of Quality Images, Noise Removal, Data Annotation, and COCO data conversion.

3.2.1 . Image Resize

Most images captured are in variable sizes, therefore the need to establish a base size is required to ensure compatibility during model training. Hence, this study selected a resolution of 1080 by 2040 and size of 1MB for all the images because any metric below this specification will indicate an error in the development and if the sizes are bigger the computational power will not be able to handle it.

3.2.2 . Quality Image Selection

It is said that the performance of a model is dependent of the quality of the dataset used. In this regard, we carried out data cleaning on the obtained dataset. Histogram and correlation coefficient methods were applied to select the quality images. The images were firstly converted into histograms before correlation coefficient captured in equation (1) was applied to obtain paired image similarities. The coefficient ranges between -1 to 1 images below a specified threshold value were discarded.

$$r(h_x, h_y) = \frac{Cov(h_x, h_y)}{\sqrt{Var(h_x) * Var(h_y)}} \qquad (1)$$

where, h_x and h_y are histogram results of two images, $Var(h_x)$ is the covariance of h_x, $Var(h_y)$ is the covariance of h_y, and $Cov(h_x, h_y)$ is the covariance between h_x and h_y.

3.2.3 Noise Removal

To mitigate the impact of noise especially salt and pepper noise in the images, we applied median filtering. As shown in Equation (2), this type of noise produces fixed values of 0 (pepper) and 255 (salt). The filtering process involves two steps: first, pixels were classified as noisy (np) or non-noisy (nnp) based on Equation (3). Then, a filtering window is applied over each pixel x_i, replacing noisy pixels with the median of surrounding values, as defined in Equation (4). Here, x_i represents the noisy image, $\phi_{i,j}$ the original pixel value, and p_n, p_p the probabilities of pepper and salt noise, respectively. The filtering window $\bar{x}^w_{i,j}$ contains pixels in a $w * w$ neighborhood, where w is an odd integer ≥ 3.

$$x_i = \begin{cases} 0 & with\ probability \quad p_n \\ 255 & with\ probrobability \quad p_p \\ \phi_{i,j} & with\ probability\ 1 - (p_n + p_n) \end{cases} \qquad (2)$$

$$x_i = \begin{cases} nnp & if\ 0 < x_i < 255 \\ np & if\ x_i = 0\ or\ x_i = 255 \end{cases} \qquad (3)$$

$$\bar{x}^w_{i,j} = \{x_{k,i} | i - \tfrac{w-1}{2} \le k \le i + \tfrac{w-1}{2},$$

$$j - \tfrac{w-1}{2} \le l \le j + \tfrac{w-1}{2}\} \qquad (4)$$

3.2.4 Data Annotation

Object annotation is essential for computer vision tasks like object detection. In this study, LabelMe was selected for its simplicity, lightweight design, and compatibility with the dataset. Using its interface, bottle objects in each image were manually outlined with polygons, generating a corresponding JSON file containing metadata such as label names, base64-encoded image data, file paths, image dimensions, and object coordinates. To ensure annotation quality, validation checks were performed to confirm image presence, correct JSON-image linking, proper labeling, and adherence to image size requirements.

3.2.5 COCO Data Conversion

Converting LabelMe annotations to COCO format was essential, as the study uses Faster R-CNN and Mask R-CNN models pre-trained on the COCO dataset. After conversion, the data was split into training and test sets using an 82:18 ratio, resulting in 332 training images and 73 test images. Each image contains various bottle brands. Table 1 presents the class distribution across the training, test, and total datasets

Table 1. Class count for the training, test, and the entire dataset

S/N	Class	Total Instance Count	Train Instanc e Count	Test Instance Count
1	BetaCAN	252	176	76
2	BetaPET	220	172	48
3	BetaRGB	734	607	127
4	BudCAN	191	154	37
5	BudRGB	2073	1728	345
6	CastleLiteRGB	507	414	93
7	EagleRGB	421	323	98
8	EagleStoutRGB	428	361	67
9	FlyingFishRGB	581	509	72
10	GrandCAN	113	91	22
11	GrandPET	204	158	46
12	GrandRGB	379	313	66
13	HeroCAN	394	338	56
14	HeroRGB	2260	1872	388
15	TrophyCAN	531	438	93
16	TrophyRGB	3569	2978	591
17	TrophyStoutCAN	79	55	24
18	TrophyStoutRGB	1133	896	237
	TOTAL	**14069**	**11583**	**2486**

3.3 Mask R-CNN

Mask R-CNN extends Faster R-CNN by adding a parallel branch for predicting segmentation masks, alongside class labels and bounding box offsets. As illustrated in Figure 2, the architecture comprises a backbone for feature extraction, a Region Proposal Network (RPN) for generating Regions of Interest (RoIs), and multibranch heads: two fully connected layers for classification and regression, and a fully convolutional network (FCN) for mask prediction. The backbone processes an input image x into feature maps f_q as shown in Equation (5):

$$f_q = ReLU(W_q \odot x + b) \quad (5)$$

where W_q is the weight, b is the bias, and $\odot$ denotes the convolution operator, $ReLU$ is the activation function. The RPN generates proposals using achors by sliding a window over the feature maps and evaluates anchors based on the intersection over Union (IoU) with ground truth boxes. Anchors with IoU < 0.3 are label negative, and those with IoU > 0.7are positive. The RPN is trained using a multi-task loss function L, combining classification L_{cls} and regression L_{reg} losses:

$$L(p_i, t_i) = \frac{1}{N_{cls}} \sum_{i=1}^{N_{cls}} L_{cls}(p_i, p_i^*) + \lambda \frac{1}{N_{reg}} \sum_{i=1}^{N_{reg}} p_i^* L_{reg}(t_i, t_i^*) \quad (6)$$

with L_{cls} defined as:

$$L_{cls}(p_i, p_i^*) = -log[p_i p_i^* + (1 - p_i^*)(1 - p_i)] \quad (7)$$

andL_{reg}as:

$$L_{reg}(t_i, t_i^*) = R(t_i - t_i^*) \quad (8)$$

using the smooth L1 function:

$$R(x) = \begin{cases} 0.5x^2, & |x| < 1 \\ |x| - 0.5, & others \end{cases} \quad (9)$$

where i represents the index of the anchor, $p_i$is the probability of the ith anchor being an object, , t_i is the vector representing the 4 parametrized coordinates of the predicted bounding box, t_i^* is the is the coordinate vector of the corresponding ground-truth bounding box. N_{cls} and N_{reg} are the normalization coefficients of L_{cls} and L_{reg} respectively, while λ is the weight parameter between L_{cls} and L_{reg}.

The region proposals generated from RPN require RoIAlign to adjust their dimension to meet the multibranch prediction networks. RoIAlign uses bilinear interpolation to extract the corresponding features of each region proposal on the feature map. During the model training process, the loss function of the Mask R-CNN model for each proposal would be given as follows:

$$L = L_{cls} + L_{reg} + L_{mask} \quad (10)$$

3.4 Evaluation Metrics

Average Precision (AP) and its variants, such as AP50, AP75, APm, and Apl were used to evaluate the performance of the developed model in the context of object localization. These metrics provide insights into how well a model detects objects across different thresholds of confidence and object sizes.

3.4.1 . Average Precision (AP)

This metric explains the precision-recall curve obtaine by varying the confidence threshold for detection. The AP captured in equation 13 provides an overall measure of detection performance, considering all bottle sizes and confidence thresholds.

$$mAP = \frac{1}{n} \sum_{k=1}^{k=n} AP_k \quad (11)$$

where n represents the number of bottle brands and AP_k represents the precision of detecting the brands

3.4.2 . Average Precision at IoU = 0.5 (AP50)

The AP at IoU of 50 considers only detections with an IoU threshold of 0.5 or higher. This implies that the metric focuses how the model detects bounding boxes that overlaps with the ground thruth by atleast 50%.

3.4.3 . Average Precision at IoU = 0.75 (AP75)
AP at IoU of 75 computes the average precision, but only for detections with an IoU threshold of 0.75 or higher. This metric emphasizes more precise detections, where the bounding box alignment with ground truth annotations is stricter.

3.4.4 . AP Medium (APm) and AP large (APl)
There are bottles that are partially visible (medium-sized) and botles that are close to the camera (large-sized). APm and APl measures performance separately for medium-sized and large-sized bottles.

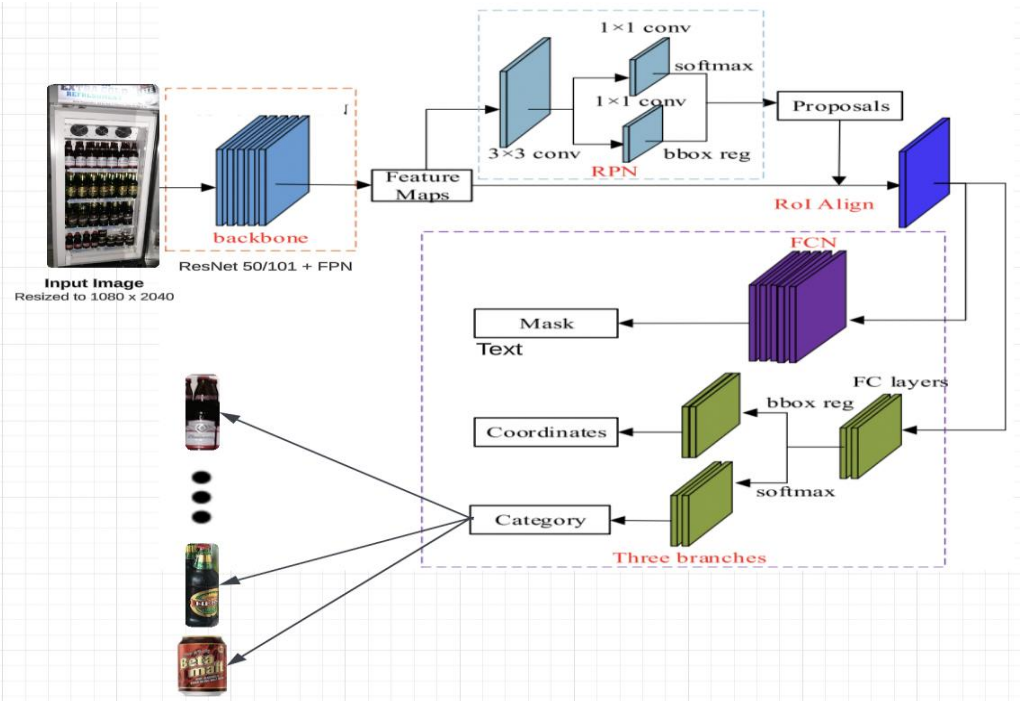

Fig 2. The structure of Mask R-CNN (Adapted from [29])

4. RESULTS AND DISCUSSION
This subsection provides detailed information about on the preprocessing and model training implementation as well as the model results.

4.1 Preprocessing Implementation
The pre-processing was implemented using Python language with libraries like OpenCV for key image processing tasks, including RGB-to-grayscale conversion (Figure 3), histogram computation, correlation coefficient calculation (Figure 4), and median filtering. Figure 4 shows grayscale histograms for two images in Figure 3, with pixel intensity on the x-axis (0–255) and normalized frequency on the y-axis. Gray Image 1 (blue) displays a wide range of intensities, while Gray Image 2 (orange) is skewed toward darker values, indicating stronger contrast. The correlation coefficient between the histograms is 0.22, suggesting less similarity. This comparison is important for identifying and removing near-duplicate images during pre-processing, where a threshold (typically 0.8) ensures only distinct images are retained for model training.

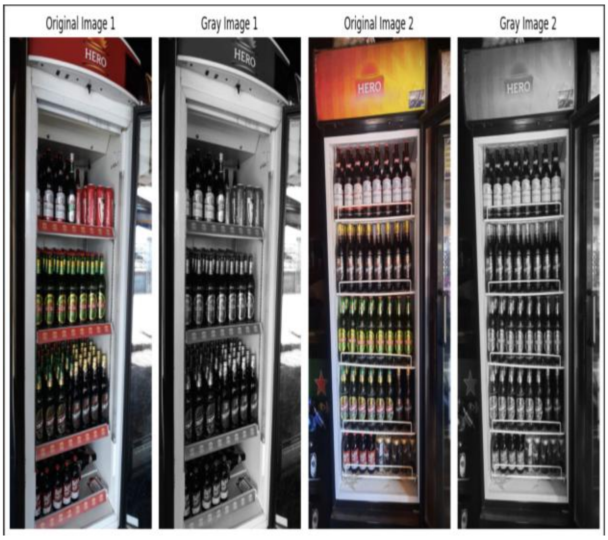

Fig 3: Original and gray image representation

International Journal of Applied Information Systems (IJAIS) – ISSN : 2249-0868
Foundation of Computer Science FCS, Delaware, USA
Volume 13– No. 2, February 2026 – www.ijais.org

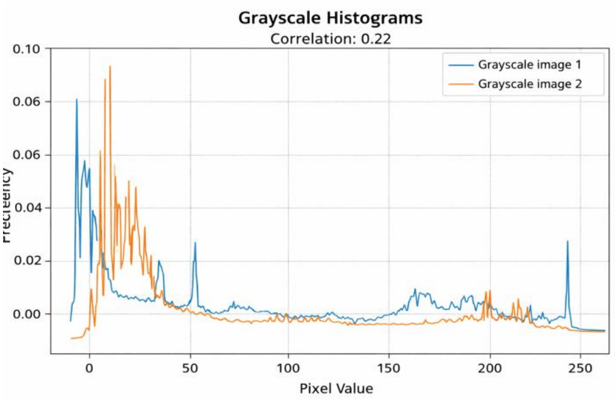

Fig 4: Histogram plot and correlation value of two image pair

4.2 Model Implementation

For model development and comparison, four architectures were implemented using Python on Google Colab with T4 GPU support. Two Mask R-CNN models with ResNet50 and ResNet101 backbones were developed for segmentation, and two Faster R-CNN models with identical backbones were used for object detection comparison. Mask-RCNN-50-FPN uses a ResNet50 backbone with Feature Pyramid Network (FPN), offering a good trade-off between accuracy and efficiency. It is pretrained on the COCO dataset and optimized for multi-scale segmentation. Training was conducted over 2100 and 4200 iterations. Figures 5 shows instance-level recognition where each bottle is detected and segmented. Figure 6 shows the bounding boxes across each detected SKUs with a mask on each SKU by Mask R-CNN.

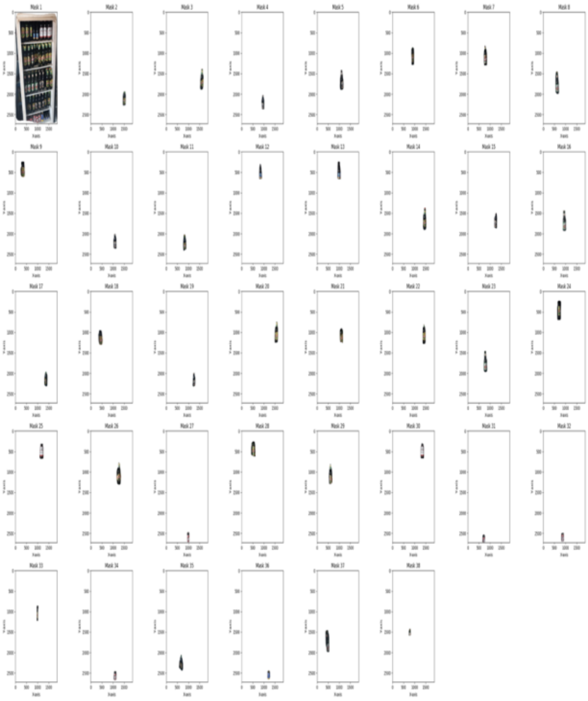

Fig 5: Segmentation of bottle objects

Fig 6: Mask RCNN Bottle Recognition

4.2.1 Mask RCNN Result

The configuration in Table 2 recorded AP scores of 78.48 (training) and 73.61 (test), while Table 3 achieved 79.43 and 76.40, respectively indicating improved test performance. For AP50, AP75, APm, and APl, Mask-RCNN-50-FPN with 4200 iterations outperformed its 2100-iteration counterpart, showing that more iterations enhance performance. Similarly, Table 4 and Table 5 show that Mask-RCNN-101-FPN-4200 achieved higher AP (79.99 vs. 75.64), AP75 (92.54 vs. 92.19), APm (75.16 vs. 73.26), and APl (75.42 vs. 73.76) compared to Mask-RCNN-101-FPN-2100. However, the 2100-iteration model slightly surpassed in AP50 (96.36 vs. 95.90). Notably, Mask-RCNN-101-FPN-4200 performed better across most metrics, confirming that increasing training iterations to 4200 leads to better model convergence and improved accuracy.

Table 2. Mask-RCNN-50-FPN-2100

Avg_precision	train_bbox	test_bbox
AP	78.47732997	73.60552252
AP50	98.8739745	95.65393791
AP75	97.12464754	91.25333672
APm	78.26219095	74.63868459
APl	78.71998536	72.56778292

Table 3. Mask-RCNN-50-FPN-4200

avg_precision	train_bbox	test_bbox
AP	79.42782462	76.40177745
AP50	99.29428972	96.58291207
AP75	98.29572919	92.04375171
APm	79.16649144	75.89640926
APl	79.78742336	74.16572085

Table 4. Mask-RCNN-101-FPN-2100

avg_precision	train_bbox	test_bbox
AP	78.91189762	75.64451106
AP50	99.19268383	96.35942846
AP75	97.75510432	92.19023644
APm	77.84324182	73.26082711
APl	79.45720712	73.76064335

Table 5. Mask-RCNN-101-FPN-4200

avg_precision	train_bbox	test_bbox
AP	80.83398067	79.98656375
AP50	99.44086422	95.90302178
AP75	98.20962014	92.53580787
APm	80.94024799	75.1648137
APl	80.8851921	75.41511509

4.2.2 Faster RCNN Result

The test bounding box results in Tables 6 and 7 show that Faster-RCNN-50-FPN-4200 outperforms Faster-RCNN-50-FPN-2100 across most metrics. It achieved higher AP (74.32 vs. 73.27), AP50 (96.05 vs. 95.97), AP75 (92.90 vs. 92.08), and APl (74.49 vs. 71.80). Although the 2100-iteration model slightly led in APm (75.01 vs. 74.84), the margin was negligible. Overall, the 4200-iteration version demonstrated better performance, suggesting that more iterations improve detection accuracy. Similarly, from Tables 8 and 9, Faster-RCNN-101-FPN-4200 consistently outperformed Faster-RCNN-101-FPN-2100. It recorded higher values for AP (74.92 vs. 72.60), AP50 (96.22 vs. 96.11), AP75 (92.98 vs. 91.24), APm (75.09 vs. 73.16), and APl (75.55 vs. 73.09). These results confirm that increasing iterations enhances the model's overall accuracy, especially in detecting objects of varying sizes with greater precision.

Table 6. Faster-RCNN-50-FPN-2100

avg_precision	train_bbox	test_bbox
AP	78.15516637	73.26877351
AP50	99.15256922	95.96915454
AP75	97.75934251	92.08215428
APm	77.95397403	75.01055686
APl	78.42238425	71.7960679

Table 7. Faster-RCNN-50-FPN-4200

avg_precision	train_bbox	test_bbox
AP	79.13583897	74.31675417
AP50	99.30115991	96.04708129
AP75	97.89445206	92.9034866
APm	78.97267039	74.84340211
APl	79.36147405	74.48551125

Table 8. Faster-RCNN-101-FPN-2100

avg_precision	train_bbox	test_bbox
AP	77.72450892	72.59915444
AP50	99.29134501	96.1053842
AP75	97.28627569	91.24033191
APm	77.09099044	73.15538255
APl	77.79318186	73.0876105

Table 9. Faster-RCNN-101-FPN-4200

avg_precision	train_bbox	test_bbox
AP	80.13142348	74.92066094
AP50	99.34772307	96.21902327
AP75	98.21573054	92.9811194
APm	80.29793937	75.08849928
APl	80.38144504	75.5459625

4.2.3 Mask RCNN and Faster RCNN Comparison

Figure 7 shows that Mask R-CNN consistently outperforms or matches Faster R-CNN across all setups, indicating the added mask branch enhances performance without compromising bounding box accuracy. Performance gains from ResNet-50 to ResNet-101 are more significant in Mask R-CNN, reflecting its greater benefit from deeper features. Additionally, increasing iterations from 2100 to 4200 improves both models, with Mask R-CNN showing more noticeable gains. The best results were from Mask R-CNN (R101/4200) with AP of 79.99 and Faster R-CNN (R101/4200) with AP of 74.92.

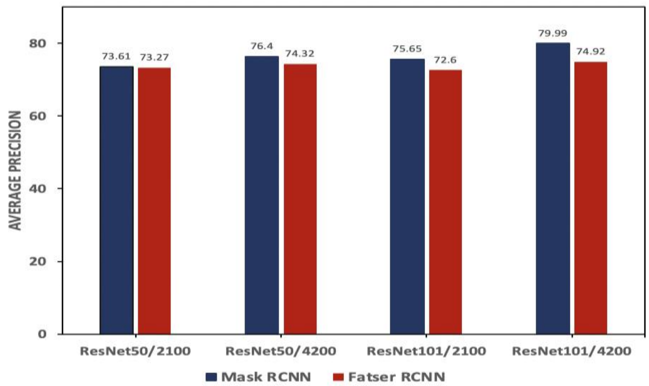

Fig 7: The Average Precision values for Mask RCNN and Faster RCNN

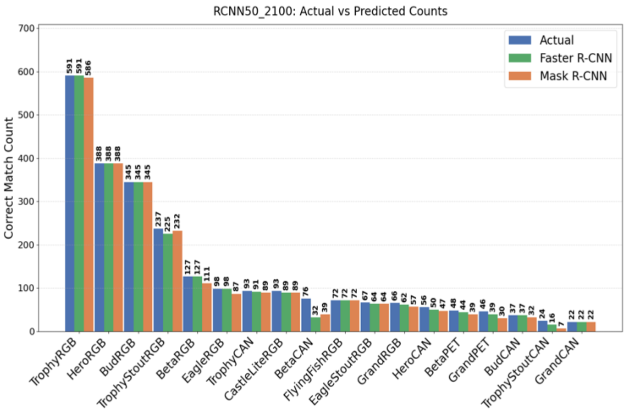

Fig 8: R50_2100: Actual vs Predicted Counts

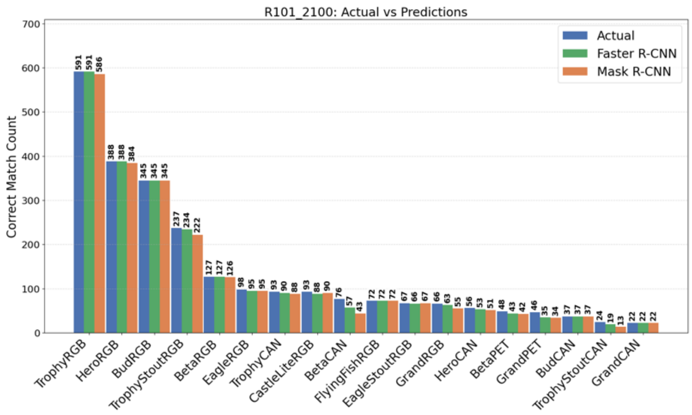

Fig 9: R101_2100: Actual vs Predicted Counts

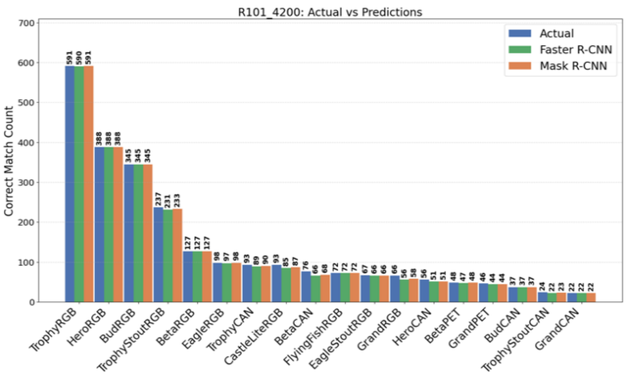

Fig 10: R101_4200: Actual vs Predictions

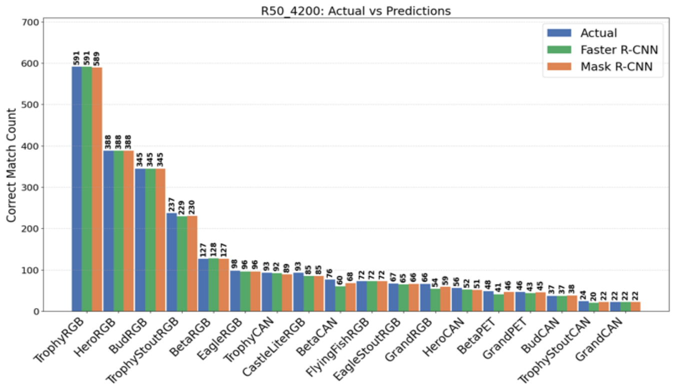

Fig 11: R50_4200: Actual vs Predictions

The comparative evaluation of the four plots (Fig 8-11) revealed clear patterns in detection stability, counting accuracy, and model convergence behavior across SKU categories. A consistent observation across all plots is that both Faster R-CNN and Mask R-CNN perform exceptionally well on high-frequency SKUs such as TrophyRGB, HeroRGB, and BudRGB. For these categories, predicted counts closely match the ground truth values regardless of backbone depth or training iteration. This suggests that when sufficient training examples are available, both architectures learn highly discriminative representations and generalize effectively in dense retail shelf environments.However, the differences between models become more pronounced when examining low-frequency SKUs such as BetaCAN, GrandCAN, and TrophyStoutCAN. In the RCNN50_2100 configuration, there is clear underestimation of several minority classes, indicating that the model had not yet fully converged. Increasing the training iterations to 4200 significantly improves performance, particularly in reducing undercounting errors. This demonstrates that training duration plays a critical role in achieving stable SKU-level detection, especially in imbalanced datasets typical of retail shelves where some products dominate visually and numerically. Comparing backbone architectures, the ResNet101-based models consistently outperform their ResNet50 counterparts. The deeper architecture appears to provide richer feature representations, enabling better discrimination between visually similar bottle types. This improvement is evident in the R101_2100 configuration, which already shows more stable predictions than RCNN50_2100, and becomes even more pronounced in R101_4200. The R101_4200 model exhibits the closest alignment with actual counts across nearly all SKU categories, indicating stronger generalization and more reliable object localization. When comparing detection frameworks, Mask R-CNN demonstrated slightly more stable counting behavior across most SKUs. Its predictions tend to remain consistently close to the ground truth, particularly in medium- and high-frequency categories. Nevertheless, Faster R-CNN performed competitively and occasionally matches or slightly exceeds Mask R-CNN in certain mid-range SKU counts.

4.2.4 Mask RCNN and Faster RCNN Cumulative Processing Time

Figure 12 illustrates the cumulative processing times for both Mask RCNN and Faster RCNN models with ResNet50 and ResNet101 backbones on the test data. As shown in the figure, Mask RCNN with the ResNet101 backbone exhibits the shortest cumulative processing time of 67.5 seconds, likely due to its more efficient architecture. In contrast, Mask RCNN with the ResNet50 backbone has the longest processing time of 187.5 seconds.

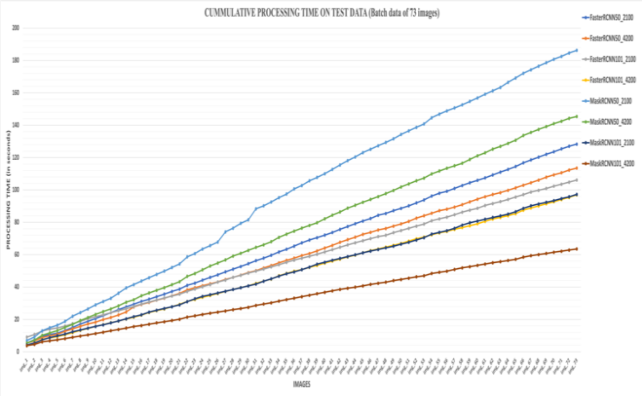

Fig 12: Cummulative Processing Time

4.3 Dashboard Display

Figure 13 presents the JSON file generated after model recognition, which contains detailed analyses of recognized bottle brands. These analyses include the coordinates of detected bottles, image quality assessment, identification of competitive brands (if any), the total count of detected bottles, and the count for each bottle class. Bottle classes that are not present in the image are recorded as zero (0). It is important to note that the model only focuses on bottles visible at the front, excluding those positioned at the back. This JSON file serves as the data source for the dashboard display of the image, as shown in Figure 14. The breakdown of detected bottle counts by class is as follows: "TrophyRGB" (Trophy) with 11 bottles, "BudRGB" (Budweiser) with 4 bottles, "HeroRGB" (Hero) with 3 bottles, "TrophyStoutRGB" (Trophy Stout) with 3 bottles, and "CastleLiteRGB" (Castle Lite) with 3 bottles. No additional bottle classes were detected in the image

```
{"coordinate": {"BetaRGB330": [[169.4341278076172, 426.56256103515625],
                               [113.96194458007812, 425.06890869140625],
                               [188.2237091064453, 427.2603759765625],
                               [132.3041229248047, 425.9122314453125],
                               [151.0463409423828, 426.2078857421875],
                               [207.1446533203125, 427.49383544921875],
                               [226.4468994140625, 427.89215087890625]],
               "BudRGB600": [[130.48753356933594, 148.36492919921875],
                             [106.85459899902344, 148.21475219726562]],
               "CastleLiteRGB600": [[237.28713989257812, 148.68121337890625],
                                    [213.436279296875, 149.90623474121094]],
               "FlyingFishRGB420": [[192.06158447265625, 151.49456787109375],
                                    [151.86663818359375, 153.16065979003906],
                                    [171.28994750976562, 153.79933166503906]],
               "Glass": [[170.71990966796875, 254.44113159179688]],
               "HeroRGB600": [[115.69380187988281, 253.64865112304688],
                              [159.8741455078125, 254.45828247070312],
                              [227.9324951171875, 255.67613220214844],
                              [137.9950408935547, 253.62200927734375],
                              [204.6958770751953, 254.65847778320312],
                              [182.182861328125, 255.69305419921875]],
               "TrophyRGB600": [[179.87379455566406, 347.4387512207031],
                                [223.3402557373047, 348.6338806152344],
                                [201.12063598632812, 348.2432861328125]],
               "TrophyStoutRGB600": [[158.47601318359375, 347.619140625],
                                     [115.73289489746094, 347.3199462890625],
                                     [137.33018493652344, 347.23095703125]]},
"competitor_presence": False,
 "image_quality": "Good",
 "no_of_bootles": 26,

 "present_classes": {"BetaCAN330": 0,
                     "BetaPET330": 0,
                     "BetaRGB330": 7,
                     "BudCAN330": 0,
                     "BudRGB600": 2,
                     "CastleLiteRGB600": 2,
                     "EagleRGB600": 0,
                     "EagleStoutRGB600": 0,
                     "FlyingFishRGB420": 3,
                     "Glass": 1,
                     "GrandCAN330": 0,
                     "GrandPET330": 0,
                     "GrandRGB330": 0,
                     "HeroCAN330": 0,
                     "HeroCAN500": 0,
                     "HeroRGB600": 6,
                     "TrophyCAN500": 0,
                     "TrophyRGB600": 3,
                     "TrophyStoutCAN330": 0,
                     "TrophyStoutRGB600": 3}
```


Fig 13: JSON with detected class and count

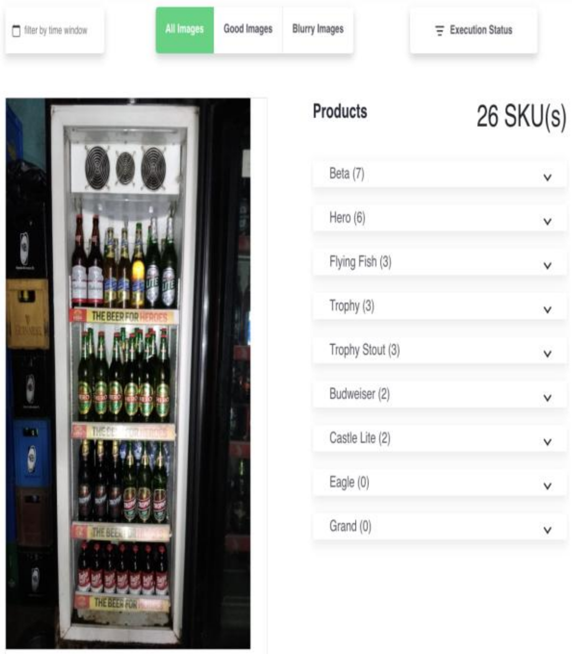

Fig 14: Image of Chiller with bottle identifcation and count

5. CONCLUSION

In conclusion, this study had developed a Mask R-CNN-based recognition and counting system tailored for bottle inventory management and compared its performance with Faster R-CNN. Through a process involving data collection, preprocessing, model training, evaluation, and scalability testing, the study demonstrated that Mask R-CNN outperforms Faster R-CNN in detecting bottle objects two backbones (ResNet50 and ResNet101) with an mAP of 79.99 as compared with Faster RCNN mAP of 74.92. The results highlight the effectiveness of the Mask R-CNN architecture, particularly its ability to leverage the added mask prediction branch without compromising bounding box detection accuracy. This makes Mask R-CNN a better choice for tasks requiring object detection benefiting from the richer feature extraction and multi-task learning. However, other backbones like EfficientNet or Swin Transformer can be used, which might offer better performance or computational efficiency.

Future research may explore more advanced and computationally efficient backbone architectures such as EfficientNet or Swin Transformer, which could potentially enhance detection accuracy while reducing computational complexity. Additionally, optimisation techniques including model pruning, quantisation, and knowledge distillation may be investigated to facilitate real-time deployment on edge devices or mobile platforms.

6. REFERENCES

[1] Munyaka, J. B., Yadavalli, V. S. S. Inventory management concepts and implementations: a systematic review. *South African Journal of Industrial Engineering*, **2022**, Vol. *33*, pp. 15-36

[2] Rubel, K. Increasing the Efficiency and Effectiveness of Inventory Management by Optimizing Supply Chain through Enterprise Resource Planning Technology. *EFFLATOUNIA-Multidisciplinary Journal*, **2021**, Vol. *5*, pp. 1739-1756.

[3] Ubabudu, M. C., Ozoemena, P. C., Anam, B. E. Effectiveness of Inventory Management on the Profitability of Manufacturing Sectors in Nigeria Bottling Company, Kaduna. *European Journal of Business and Innovation Research*, **2024**, *12*, pp.1-14.

[4] Manoharan, G., Kumar, V., Karthik, A., Asha, V., Mohan, C. R., & Nijhawan, G. IoT-Based Smart Inventory Management System Using Machine Learning Techniques. In *International Conference on Advances in Computing, Communication and Applied Informatics (ACCAI)*. **2024,** pp. 1-5.

[5] Inprasit, T., Tanachutiwat, S. Reordering point determination using machine learning technique for inventory management. In *International Conference on Engineering, Applied Sciences, and Technology (ICEAST)*. **2018**, pp. 1-4.

[6] Dwivedi, Y. K., Ismagilova, E., Hughes, D. L., Carlson, J., Filieri, R., Jacobson, J., ... & Wang, Y. Setting the future of digital and social media marketing research: Perspectives and research propositions. *International journal of information management*, **2021**, *59*, 102168.

[7] Pramodhini, R., Kumar, S., Bhardwaj, S., Agrahari, N., Pandey, S., Harakannanavar, S. S. E-Commerce Inventory Management System Using Machine Learning Approach. In *2023 International Conference on Data Science and Network Security (ICDSNS)*. **2023**, pp. 1-7.

[8] Dhaliwal, N., Tomar, P. K., Joshi, A., Reddy, G. S., Hussein, A., Alazzam, M. B. A detailed Analysis of Use of AI in Inventory Management for technically better management. In *2023 3rd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*. **2023**, pp. 197-201.

[9] Waheed, F., Omar, M., Ibrahim, S. Z., Chugtai, R., & Aejaz, S. H. Application of Industrial IoT in Developing a Sustainable and Automatic Liquid Filling Plant. *Pakistan Journal of Scientific Research*, **2023,** *3*(1), 130-140.

[10] Lemley, J., Bazrafkan, S., & Corcoran, P. Deep learning for consumer devices and services: pushing the limits for machine learning, artificial intelligence, and computer vision. *IEEE Consumer Electronics Magazine*, **2017**, Volume *6*, pp. 48-56.

[11] Zhao, X., Wang, L., Zhang, Y., Han, X., Deveci, M., & Parmar, M. A review of convolutional neural networks in computer vision. *Artificial Intelligence Review*, **2024**, Vol. *57*, pp. 1- 43.

[12] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In NIPS, **2015**, pp. 1- 7

[13] Okwu, M. O., & Emovon, I. (2018). Artificial Neural Network (ANN) and Greedy Heuristic (GH) Approach to Transshipment Model in a Bottling Plant. *Operations Research and Information Engineering*, **2018,** Vol 2, pp. 51-60.

[14] Yani, I., & Budiman, I. Development of identification system of cans and bottle. In *Journal of Physics: Conference Series*. **2015**, Vol. 622, p. 012053. IOP Publishing.

[15] A. Voulodimos, N. Doulamis, A. Doulamis, & E. Protopapadakis, Deep learning for computer vision: A brief review. *Computational intelligence and neuroscience*, **2018**,1, 7068349.

[16] Kucukyıldız, G., and Ocak, H. (2022). Bottle counting system for a re-cycling facility based on processing of depth images. *Sigma Journal of Engineering and Natural Sciences*, **2022,** *Vol. 40*, pp. 464-474.

[17] L. Remmelzwaal, “Object detection and tracking for crate and bottle identification in a bottling plant using deep learning,” In *Artificial Intelligence and Applications*, **2023**, Vol. 1, pp. 191-195.

[18] X. Zhou, W. Gong, W. Fu and F. Du, "Application of deep learning in object detection," *2017 IEEE/ACIS 16th International Conference on Computer and Information Science (ICIS)*, Wuhan, China, **2017**, pp. 631-634.

[19] Shuai, Q.,Wu, X. Object detection system based on SSD algorithm. In *international conference on culture-oriented science & technology (ICCST)*. **2020**, pp. 141-144.

[20] Malhotra, P., & Garg, E. Object detection techniques: a comparison. In *2020 7th international conference on smart structures and systems (ICSSS)*. **2020,** pp. 1-4.

[21] Tarimo, W., Sabra, M. M., & Hendre, S. Real-time deep learning-based object detection framework. In *2020 IEEE Symposium Series on Computational Intelligence (SSCI)* **2020**, pp. 1829-1836.

[22] Ji, G., Zheng, Y. Automatic Lane Line Detection System Based on Artificial Intelligence. *Journal of Electrical and Computer Engineering*, **2022**, pp. 1-10

[23] Shi, Z. Object detection models and research directions. In IEEE International Conference on Consumer Electronics and Computer Engineering (ICCECE). **2021**, pp. 546-550.

[24] Stäcker, L., Fei, J., Heidenreich, P., Bonarens, F., Rambach, J., Stricker, D., & Stiller, C. Deployment of deep neural networks for object detection on edge AI devices with runtime optimization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* **2021**, pp. 1015-1022.

[25] Kirana, N. L., & Kurnianingtyas, D. (2024). A Deep Learning Approach to Plastic Bottle Waste Detection on the Water Surface using YOLOv6 and YOLOv7. *Engineering, Technology & Applied Science Research*, *14*(6), 18623-18630.

[26] Heller, A., Jacobs, M., Acosta-González, G., Basola, A., Beck, J., Garnes, W., ... & Davison, J. H. (2025). Plastic water bottle detection model using computer vision in aquatic environments. *Scientific Reports*, *15*(1), 24851.

[27] Martins, O. O., Abdulhamid, M. M., Lawal, M. O., Olugbenga, O. T., & Okikiola, O. E. (2023). Development of a Sequential Neural Network Model for Bottle-Fill Level Detection and Classification.

[28] Anush, C., Yashwanth, K., Shashank, S., Reddy, M. V., & Kumar, A. (2021, August). Bottle line detection using digital image processing with machine learning. In *Journal of Physics: Conference Series* (Vol. 1998, No. 1, p. 012033). IOP Publishing.

[29] Zhou, Z., Zhang, M., Chen, J., & Wu, X. Detection and classification of multi-magnetic targets using mask-RCNN. *IEEE Access*, **2020**,*8*, pp.187202-187207.