Artificial Intelligence for Human-Technologies-Economy Sustainable Development
23–25 May 2024
Lublin, Poland
make learn
TIIM
PIConf
Management, Knowledge and Learning
International Conference
Technology, Innovation and Industrial Management
Sustainable Development

Enhancing Poultry Welfare and Production through Advanced Chicken Detection, Tracking and Counting: A Deep Learning Approach

Suraj Salihu
Gombe State University, Nigeria
surajsalihu@gsu.edu.ng

Suleiman Salihu Jauro
Gombe State University, Nigeria
ssjauro@yahoo.com

Mohammed Nasir Salihu
Teffund, Nigerian
salihumn@gmail.com

Abstract

The integration of deep learning into chicken welfare and rearing demonstrates the potential to transform poultry farming methods and increase the financial worth of this significant agricultural industry. As a result, implementing deep learning algorithms for monitoring chickens offers farmers prospects to boost production and profit and ensure the poultry industry's long-term growth. This study presents a deep-learning learning approach for chicken detection using YOLOv8, YOLOv7, and YOLOv5 models based on MAP scores at different IoU thresholds. We proposed a chicken tracking and counting pipeline using YOLOv8 + DeepSORT. Detected chickens are tracked using the Kalman filter, and an ID is assigned to each detected chicken; a specific colour is assigned to the tracked chicken based on a unique ID. A chicken dataset of V1, V2, V3, V4 and V5 was built containing 673 images; YOLOv8+V4 demonstrates remarkable performance, outperforming other models with mAP50 of 97.4% and mAP50-95 of 75.4%. Notably, it excels in identifying chicks with mAP50 of 98.3% and mAP50-95 of 76.5%, making it a reliable model for the Chick and Chicken classes. YOLOv8+V5 also performs well, achieving mAP50 and mAP50-95 values of 95.9% and 71.1%, respectively. YOLOv7 models (V1 and V4) show consistent performance, with YOLOv7+V4 performing equally well overall, with a mAP50 of 96.0% and mAP50-95 of 70.6%. YOLOv7+V5 slightly lag behind with mAP50 of 95.9%. YOLOv5+V4 with mAP50 of 96.4% demonstrates respectable performance, surpassing YOLOv7 models in mAP scores.

Keywords: chicken detection, yolo, CNN, deep learning and DeepSORT

1. Introduction

Farmers engage in poultry farming due to its economic value and profitability. Chickens offer a convenient source of high-quality protein and essential nutrients for maintaining a healthy body. They require low energy and have lower cholesterol levels compared to other meat types [1]. This makes chicken a favorable option for those concerned about cardiovascular health and seeking a balanced diet. Poultry meat contains various nutrients such as vitamin B12, niacin, iron, and zinc [2]. The global poultry market has been on a growth trajectory, with a Compound Annual Growth Rate (CAGR) of 10.1%, reaching $350.87 billion in the previous year. Predictions estimate a further increase to $493.21 billion in 2026, indicating a strong market demand [3]. Traditional methods of managing chicken welfare are labor-intensive and wasteful of resources. To address this, Artificial Intelligence (AI) has emerged as a promising technology for enabling efficient and intelligent poultry farming [4]

Real-time detection, tracking, and counting of chickens are crucial in the poultry industry [5] Maintaining an accurate count of the chicken population is vital for effective decision-making. In large farms, chickens might be hidden or unaccounted for, leading to disease spread. Implementing cameras and networked detection algorithms can locate hidden chickens, aiding their care and well-being. Monitoring chicken activities in real-time allows identifying health issues early on [1]. Detecting sick chickens is a challenge that requires constant vigilance. Signs like body swelling, limping, nasal discharge, and changes in behavior indicate sickness. Maintaining a balance between healthy and sick flocks can be difficult for humans to monitor in real time. Camera-based systems can provide continuous feedback and aid in evaluating chicken activities without causing stress [6].

Automated chicken monitoring is gaining attention for its potential to enhance animal welfare, disease control, breeding standards, and waste reduction [7]. Automated systems utilize AI to analyze data from wearables, cameras, and sensors to detect health problems early, thereby improving welfare and minimizing disease spread [8]. AI-driven real-time disease outbreak detection and behavioral analysis enable prompt intervention [9] and the assessment of avian behavior for signs of stress or discomfort [10]. AI-based systems optimize the environment in real time to create a comfortable and healthy atmosphere for the chickens [11]. The wealth of data generated by AI aids decision-making by integrating health, environmental, and productivity data to enhance management techniques and welfare protocols. However, real-time assessment of chickens poses challenges in large poultry farms due to complex backgrounds, occlusions, and variations in lighting [12].

Traditional methods relying on handcrafted feature engineering are slow, lack generalization, and demand extensive human labor. To overcome these limitations, CNNs have shown promise in image analysis tasks. Unlike Fully Connected Networks, CNNs handle nonlinear data like images effectively by reducing the number of trainable parameters [13]. The proposed approach in this paper aims to enhance poultry welfare and production through a hybrid deep-learning model for real-time chicken detection, tracking, and counting. YOLOv5 [14], YOLOv7 [15], and YOLOv8 [16] detection accuracies are compared, followed by integrating the best model with Deepsort for tracking and counting pipelines.

2. Related works

A hybrid deep learning model called ChickTrack based on Yolov5 for chicken identification in video frames and Deepsort for tracking and counting was proposed [7]. The model uses class probabilities and bounding box coordinates for chicken recognition, trajectory tracking and counting. It achieved good performance with a maximum tracking ID of 323 against ground truth of 310. The LC-DenseFCN model for poultry detection and counting was introduced by [17]; it utilizes a full convolutional network and DenseNet backbone for feature extraction. The method achieved a high detection accuracy of 93.84% and a counting accuracy of 97% with a processing speed of 9.27 fps. The use of pixel-level predictions and deconvolutional operations helps preserve spatial information. A deep learning procedure called TabNet was developed by [18] that incorporates wearable sensors (accelerometer and RFID microchips) placed on chickens to track their movement in real-time. They use Generative Adversarial Networks (GANs) to recreate missing records from the dataset and achieve a high accuracy of 97% in differentiating between sick and healthy chickens. An end-to-end deep learning approach for estimating the density and counting the number of chickens in poultry called ChickenNet was presented by [19]. The model utilizes a pixel-level classification approach and employs Fully Connected Network (FCN) for density map estimation. ChickenNet outperforms the YOLOv3 model in terms of accuracy percentages (P1 and P2) related to pixel prediction and classification. The ChickenNet model addresses the problem of chicken density estimation and achieves better results (1.3% and 2.5%) compared to YOLOv3 of 16% and 9 for P1 and P2 respectively. An hybrid system for tracking and detecting chickens using a modified YOLOv4 model and Kalman filters was proposed by [20]. The system achieves a high accuracy rate of 99.9% in identifying chickens based on tracking their movement using low-resolution grayscale video footage. The hybrid system combining YOLOv4 and Kalman filters shows promising results for chicken tracking and detection. A broiler counting method using Faster R-CNN with ResNet-101 was presented by [21]. They compared different CNN architectures and foundResNet- 101 to be more accurate with 92.% accuracy at AP of 50% and 78.1% as 75%.
[22] recommended using YOLOv3 for chicken detection and gender classification. The method employed DeblurGAN network and PSS-NSC defuzzification technique to enhance image quality and remove motion blur. The study evaluated different CNN backbones and achieved a detection accuracy of 92.23% and classification accuracy of 96.85% using VGG-19. Table I provide a breakdown of the dataset used, accuracy and the drawback as follows:

Table I: Description and Contribution, Dataset used, Accuracy and the Drawbacks of some of the methods from related literature

N/S	Algorithms	Authors	Description and Contributions	Dataset collection and volume	Accuracy	Drawback
1	DeepSortYolov5 with CSP-Backbone and PANet	[7]	Designed for real-time chicken identification, tracking, and counting.	Private dataset: 72 chickens at 1280x720	323 IDs	Struggles with detecting smaller and closer objects.

2	LC-DenseFCN Using DenseNet Backbone	[17]	Counts chickens using a localization loss-based algorithm. DenseNet backbone used for feature extraction.	Private dataset: 1200 images with 1080p resolution	93.84%	Potential memory requirements
3	TabNet Deep Learnin Model	[18]	Deep learning classifier with wearable sensors to monitor and classify chicken health. Utilizes	Artificial dataset: 10,000 samples	97%	Technological limitations, cost, private dataset.

N/S	Algorithms	Authors	Description and Contributions	Dataset collection and volume	Accuracy	Drawback
			synthetic data for addressing class imbalance.			
4	ChickenNet + VGG16	[19]	FCN-based model for estimating density and counting of crowded chickens. Uses pixel-level classification and regression.	Dataset: 100 images	P1: 1.3%, P2: 2.5%	Large model size, lengthy training time, dataset not available.
5	YOLOv4 + Kalman-filter	[20]	Hybrid system for chicken identification and tracking. YOLOv4 for detection, Kalman filter for tracking.	Dataset: 1296 frames with 1000 annotated frames	99.9 %	Difficulties with smaller objects, grayscale images, and pre-trained network.
6	YOLOv3	[22]	Estimates gender ratios and distribution of chickens. Utilizes VGG-16, VGG-19, ResNet-18, ResNet-34, DenseNet-121,	Dataset: 800 chicken photos at 2992x2000 Resolution	92.23 %	Difficulties with smaller objects, dataset not available.
7	Faster RCNN with ResNet-101 Backbone	[21]	Optimized object detection model for counting broilers. Utilizes the Detectron2 model trained on COCO dataset.	Dataset: 600 images with annotations	92.8 %	RPN trained with all anchors in one image, slower convergence.

3. Material and Methodology

Figure 1 gives an overview of our pipeline; we start by adding video frames to our framework. In order to identify the chicken in each frame, object detection is used. However, detecting the chicken is insufficient to determine the number of chickens in a video, we pass the detections to our tracking model, associations are made between the frames by the tracker. For instance, if chicken "a" appears in the two consecutive frames, the tracker connects

them and assigns them a unique ID of say 1. Once a specific chicken has been accomplished and tracked across the frames, it will be counted.

3.1 Chicken Detection Models

In this study, we investigated three object detection models of YOLOv5, YOLOv7, and YOLOv8 to meet the prerequisite of chicken detection. The YOLOv5 [14] design comprises three core elements: the Head, Neck, and Backbone. The Backbone's CNN extracts multi-level image features. The Neck combines these features for creating prediction feature maps, while the Head predicts classes and bounding boxes using Neck's feature maps. YOLOv5 improves on YOLOv4 with enhanced model structure, training, and performance. It adopts CSP Darknet53, reducing redundant computations and boosting efficiency. With 29 convolutional layers of 3x3 size, YOLOv5 has 27.6 million parameters and a receptive field of 725x725. An SPP block extends CSPDarknet53's receptive field without slowdown, and PANet enriches features by merging low-level with high-level attributes.

The YOLOv7 [15] surpasses YOLOv5 through design advancements. This includes planned/re-parameterized convolution, coarseness for auxiliary loss, compound scaling, Extended Efficient Layer Aggregation Network (EELAN), and fineness for lead loss. EELAN enhances learning while preserving gradient approach. YOLOv7 introduces Trainable Bag of Freebies (TBoF) training method, substantially boosting object detection accuracy and generalization across detectors like RetinaNet, SSD, and YOLOv3. It increases accuracy and speed (5 - 160 FPS), surpassing GPU V100 performance of YOLOR, Scaled-YOLOv4, DETR, Deformable DETR, ViT-Adapter-B, but YOLOv7's limitations include model structure, training data, and hyperparameters.

YOLOv8 [16] combines features of multiple real-time object detectors, retaining PAN- FPN, SPPF module, and CSPall from YOLOv5, with significant innovations. It introduces a new SOTA model with object recognition networks at P5:640 and P6:1280 resolutions, creating models at various scales based on YOLOv5-like scaling coefficient for diverse applications. YOLOv8's C2f module, derived from YOLOv7's ELAN, maintains YOLOv5's original concept. The backbone is CSPDarknet53, followed by C2f module, and YOLOv8 predicts bounding boxes with an anchor-free detection head. A larger feature map and improved convolutional network enhance precision and speed, with feature pyramid networks aiding recognition of various-sized objects. YOLOv8 features a user-friendly API and its extensibility allows easy compatibility and performance evaluation with other YOLO versions (Lou et al., 2023).

3.2 Chicken Tracking

In this study, a DeepSORT Multi-Object Tracking (MOT) algorithm was employed to fulfil the requirement of real-time chickens tracking due to speed-accuracy trade-offs in tracking- by-detection methods which integrate a tracking pipeline with a detection pipeline. DeepSORT represents an advanced iteration of the Simple-Object & Real-Time Tracking (SORT) approach [23]. It introduces an innovative association mechanism that reduces frequent ID switching and enhances object tracking accuracy, especially when objects are occluded for extended periods. FairMOT [24] might have trouble with really quick object motions, while TrackNet [25] may not

Be able to handle complicated occlusions or quick appearance changes. Objects with unusual forms or orientations may be difficult for CenterTrack [26] to handle and StarTrack [27] handles occlusions, retains object identities, and concentrates more on benchmarking than real-time tracking. DeepSORT involves using data association to assign unique IDs to detected chickens from the YOLO model.

3.3 Chicken Counting

A method to count chicken using Deep Learning (DL) involves two main steps: first, chicken are detected using Convolutional Neural Networks (CNN), then the instances detected are counted. This method involves locating specific instances of each detected chickens using a visual object detector. To count the poultry in each frame, the DeepSORT algorithm is used. It initializes by processing the detected chicken information, updates their positions as they move around frames, tracked chicken data is then used to initiate counting which is done by defining a region of interest using the frame's height and width.

3.4 Working flow of Chicken Detection, Tracking and Counting

DeepSORT offers reliable and precise multi-object tracking by integrating various approaches, even in difficult situations where objects may be obscured for lengthy periods of time, to successfully track chickens after it was detected using Yolo, the DeepSORT algorithm utilizes the Kalman Filter for prediction, Deep Appearance Descriptor for feature extraction, the Mahalanobis Distance for data association, Hungarian algorithm for the link between detections and tracks**.** The workflow describes the step-by-step procedure of identifying, tracking, and counting chickens in image or video frames as follows**:**

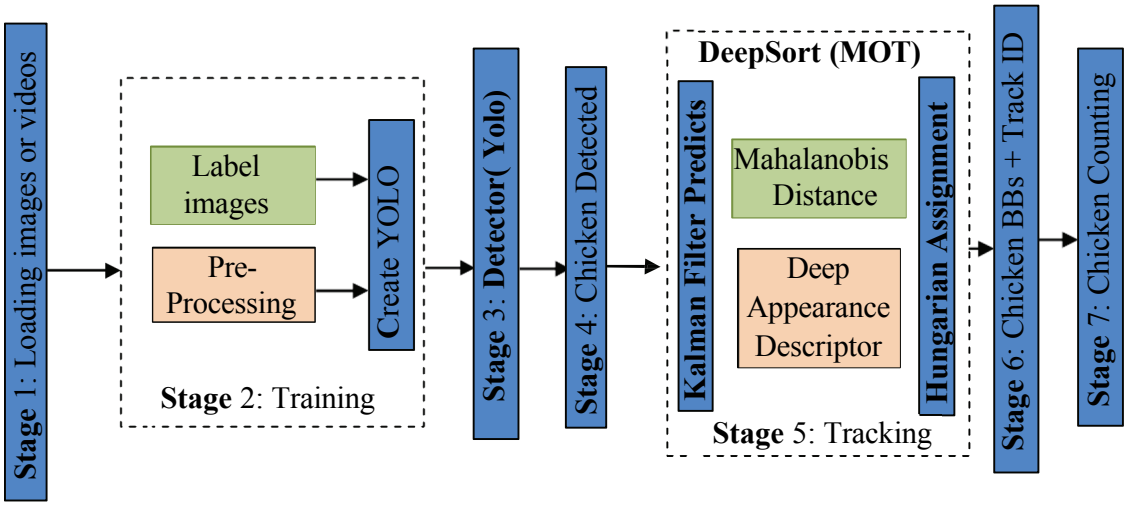

Figure 1: *The working flow of the proposed Yolo Chicken Detector, Tracker and Counting*

Stage 1: Describe Chicken Localization this is the process of loading images or frames containing chickens to be analyzed.

Stage 2: Label Images and Preprocessing: this stage is responsible for pre-processing and stretching the input frames or images to 640 x 640 required by the Yolo model and compares them with the labeled data for subsequent stages.

Stage 3: YOLO Detector: Apply the YOLO object detection algorithm to the preprocessed images or frames, then Non-Maximum Suppression (NMS) is applied to remove redundant and overlapping bounding boxes, finally the model Detect and localize chickens within the frames, retain only those bounding boxes with confidence scores above a certain threshold.

Stage 4 Detected Chickens: this stage uses the output of the YOLO detector to identify the detected chickens in the videos frames and collect the bounding box coordinates and confidence scores for each detected chicken.

Stage 5 is a Chicken Tracking Management phase based on:

1. Kalman Filter Predicts

Kalman Filter [28] is a central component of the DeepSORT tracking algorithm. It is used to predict the future position and state of tracked chickens, based on their previous motion. Kalman filter calculates centroids for bounding boxes in the first frame, maintaining the same ID in subsequent frames. When a new chicken appeared, a new ID was assigned, and the old one was dropped. Tracking became challenging due to appearance/disappearance across frames and occlusions. This problem was tackled using centroid analysis across frames and estimating distance from the previous centroid. During the prediction phase, the Kalman Filter forecasts the positions of the tracks chicken in subsequent frames. This prediction is crucial for maintaining the continuity of chicken tracks and updating their states as new detections are obtained and erase tracks that have not been connected to any detection for an extended period. Kalman filter predicted velocity and future position, following equation describes the movement of chicken tracking:

$$\mathrm{X}t = Q\mathrm{X}_{t-1} + LU_t + \mathrm{W}_t \tag{1}$$

Where $\mathrm{X}t$ represents the anticipated state at time t, W_t represents the estimated error, while the matrix Q denotes the changes X from the preceding step and for updating acceleration U_t, a different matrix L is employed. Whereas the measurements are modelled using the following:

$$Z_t = H\mathrm{X}_t + V_t \tag{2}$$

In the above equation, Zt is a measurement vector, like a position vector, H stands for the transformation matrix, while the noise is measured using Vt.

2. Mahalanobis Distance

Mahalanobis Distance [29] is a metric used in the data association step of the DeepSORT tracking algorithm. It quantifies the dissimilarity between feature vectors of chicken detections and existing tracks; the Mahalanobis Distance provides a robust measure of similarity that takes into account the correlation between different feature dimensions. This distance metric aids in determining the best matching pairings between new chicken detections and pre-existing tracks,

facilitating accurate data association even in scenarios where objects might be partially occluded or exhibit variations in appearance and is defined as:

$$D^2 = (x - \mu)^T . C^{-1}(x - \mu) \quad (3)$$

Where D^2 represents the Mahalanobis distance, x is the feature vector of the detected chicken, and μ is the mean feature vector of the tracked chicken. C^{-1} is the inverse of the covariance matrix calculated from the tracked chicken's feature vectors.

3. Deep Appearance Descriptor

The Deep Appearance Descriptor is a technique utilized in DeepSORT for feature extraction. It is employed to capture the visual attributes of tracked chickens. By generating a representation of a chicken's appearance based on its bounding box, this descriptor enables the comparison of visual characteristics between detections and tracks. These appearance features are essential for associating newly detected chickens with their corresponding previous tracks. The process ensures that even in situations where a chicken's appearance varies due to occlusion or other factors, the correct association can be maintained.

4. Hungarian Assignments

The Hungarian algorithm is utilized in the data association stage of DeepSORT to establish optimal assignments between new detections and pre-existing tracks. This algorithm efficiently solves the assignment problem by determining the best matching pairs that minimize the overall cost based on the Mahalanobis Distance between appearance features and predicted positions. By considering all possible combinations and selecting the assignment with the lowest total cost, the Hungarian algorithm links chicken detections and tracks in a manner that ensures reliable and accurate multi-object tracking. This process enables DeepSORT to effectively handle challenging scenarios where objects are occluded or disappear from view for extended periods.

Stage 6 Bounding Boxes and Track IDs: update bounding boxes for the actively tracked chickens and assign unique track IDs to each tracked chicken to maintain identity across frames.

Stage 7 Chicken Counting: count the number of uniquely tracked chickens based on detection by defining a zone, continuously update the chicken count as new chicken detections are associated and tracked

3.5 Dataset Collection

Our dataset comprises of 673 images of chickens samples of which half of the dataset were sample collected from internet and the remaining were locally from five different farms using cameras, to have robust dataset data augmentation and exploration are applied in order to gain insight into the data. The dataset is labelled in Yolo (.txt) format necessary for the architectures using labelImg software, to addressed the problem of small object detection associated with Yolo models we divide our dataset into two classes of chick and chicken, out of the total 3732 annotations 673 for chick and 3059 for chicken respectively. The image samples were taken at various times with different lighting situations as shown in Figure 2 and 3.

Figure 2: *Dataset of chicken taken in Landscape Orientation*

Figure 3: *Dataset of chicken taken in Portrait Orientation*

3.6 Data Augmentation

The primary element influencing how the trained model behaves with new data is the quantity of labelled data. The difficulty of assembling a sizable labelled dataset can be overcome with the use of data augmentation model seems to be more generalizable when the scale and the data quality are both increased. However, while gathering data in practice, it is typically difficult to capture all circumstances, such as various lighting situations. Data augmentation is a method for creating new data by modifying and changing the underlying data. Data augmentation with lighting variation must be taken into account while training the model because it is challenging to control the illumination during data collection. Data augmentation involves modifying or learning from imperfect and modest datasets. Five versions of datasets (V1 to V5)were generated using different augmentation techniques as shown in Table 2, the augmentation methodsincluding flip (horizontal and vertical), crop, rotation, shear, greyscale, hue, saturation, brightness, exposure, blur, noise, cutout, and mosaic as contained in figure 4, 5 and 6, these approaches are applied to the image in a variety of ratios and scopes.

Table 2: From the table“ □ ” indicates augmentation technique applicable while "-" indicates that an augmentation technique is not applicable for the version, F: Flip, R: Rotation, C: Crop, S: Shear, G: Grayscale, H: Hue, S: Saturation, Br: Brightness, E: Exposure, Bl: Blur, N: Noise, C: Cut-out, M: Mosaic, BB: Bounding Box, BBF: Bounding Box Flip.

Dataset Version + Augmentation Techniques Applied	**F**	**Cr**	**R**	**S**	**G**	**H**	**S**	**Br**	**E**	**Bl**	**N**	**Cu**	**M**	**BB**	**BBF**
	Horizontal	0% Min, 20% Max Zoom	-15° to +15°	±15° Horizontal, ±15° Vertical	Apply to 25% images	-25° to +25°	-25% to +25%	-25% to +25%	-25% to +25%	Up to 2.5 px	Up to 5% pixels	3 boxes, 10% size each	Applied	Rotation -15° to +15°	Vertical
V5	□	□	□	□	□	□	□	□	□	□	□	□	□	-	-
V4	□	-	-	-	□	□	□	□	-	-	□	□	□	-	-
V3	□	-	-	-	□	□	□	□	-	-	□	-	□	□	□
V2	□	-	-	-	-	□	-	□	-	-	-	-	-	□	-
V1	□	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Figure 4: *Greyscale, Mosaic, Noise and Cut-out Augmentations*

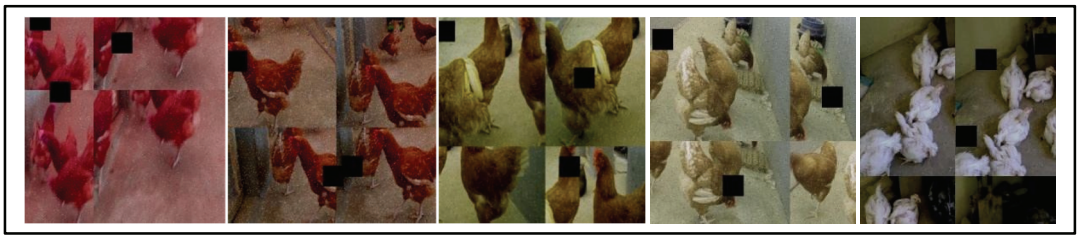

Figure 5:*Hue, Noise, Mosaic, Brightness, Saturation, Flip, Exposure and Cut-out Augmentations*

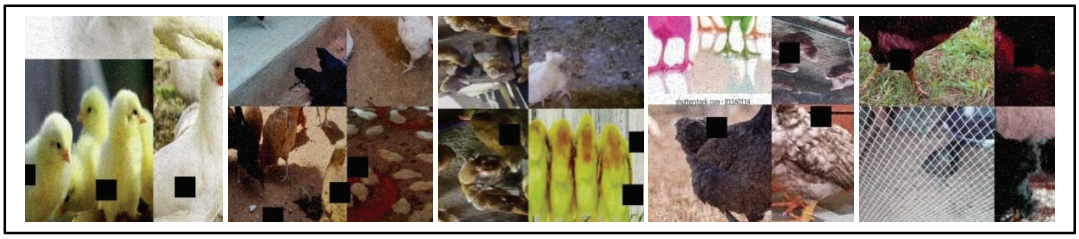

Figure 6: *Cut-out, Noise, Brightness, Saturation and Mosaic Augmentations from different images*

Table 3: *Hyperparameters used during training of the models*

Hyperparameter	YOLOv5	YOLOv7	YOLOv8
lr0	0.01	0.01	0.01
Lrf	0.01	0.1	0.1
Momentum	0.937	0.937	0.937
weight_decay	0.0005	0.0005	0.0005
warmup_epochs	3.0	3.0	3.0
warmup_momentum	0.8	0.8	0.8
warmup_bias_lr	0.1	0.1	0.1
Box	0.05	0.05	7.5
Cls	0.5	0.3	0.3
Mosaic	1.0	1.0	1.0
Mixup	0.0	0.15	0.15
Optimiser	SGD	SGD	SGD

3.7 Data Pre-processing

The size of the input image is primarily determined by the backbone receiving the image. Pre-processing in object detection involves a set of techniques applied to the input data to enhance the quality of the data, reduce noise, and extract features before feeding the data into a deep learning model, among the common pre-processing techniques employed are resize all the images to 640x640 and auto orient before they fed into the Yolo models.

3.8 Models Training

The dataset was divided into 88%, 8% and 4% representing 1400, 169 and 67 images for training, validation, and testing accordingly. The augmentation techniques described in Table 2 were only applied to the training set and models are then trained after that. After training, validation is carried out on the testing dataset to evaluate the model's overall detection accuracy. The outcomes of each model are then compared and analyzed. The three models were all trained

using genetic algorithms' best hyperparameters. All models were trained using a batch size of 16 with image size of 640x640 over the course of 50 epochs; Table3 gave the highlights of the hyperparameters turning as follows:

3.9 Performance Metrics

Four metrics of AP, Precision (P), Recall (R), F1 score were used for the evaluation of our Yolo models. AP is calculated at different IoU thresholds (mAP50), aggregated (mAP50:90). IoU measures how similar two sets are and is calculated as the intersection to union ratio of the two sets as in equation 3 below.

$$IoU(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{4}$$

Our classification relies on four states: true positive (TP), false positive (FP), true negative (TN), and false negative (FN).

$$Precision = \frac{TP(C\square ick) + TP(C\square icken)}{[TP(C\square ick) + TP(C\square icken)] + [FP(C\square ick) + FP(C\square icken)]} \tag{5}$$

The recall is calculated as the sum of all true positives (relevant items) divided by all true positives. Equation (5) presents its mathematical representation.

$$Recall = \frac{TP(C\square ick) + TP(C\square icken)}{[TP(C\square ick) + TP(C\square icken)] + [FN(C\square ick) + FN(C\square icken)]} \tag{6}$$

The F1 score is computed by averaging precision and recall values between 0 and 1 and is represented by equation (6).

$$F_1 = \frac{(2 * (Precision(c\square ick) * Recall(c\square ick) + Precision(c\square icken) * Recall(c\square icken)))}{(Precision(c\square ick) * Recall(c\square ick) + Precision(c\square icken) * Recall(c\square icken) + Precision(c\square ick) + Precision(c\square icken))} \tag{7}$$

The final step computes the Average Precision (AP), also called the P-R curve, as the area under the precision-recall curve averaged across different IoU thresholds shown in AP equations below.

$$AP = \int_0^1 p(c\square ick \text{ or } c\square icken) d(c\square ick \text{ or } c\square iken) \tag{8}$$

During training, the mean average precision (mAP) is computed by averaging each class's average precision (AP). This is done across various IoU thresholds for each class, resulting in the final mAP metric for the test data, calculated by averaging the mAP values of all classes using Equation (8).

$$mAP = \frac{1}{N} \sum_{i=1}^{N} AP_i \tag{9}$$

Where N is the overall number of classes with background as a class type and AP_i is the average precision of every single class.

4. Results and Discussion

4.1 Chicken Detection Results

The performance of the chicken detection for YOLOv8, YOLOV7, YOLOV5 models was evaluated based on mAP scores at different IoU thresholds (mAP50 and mAP50- 95).Interesting insights into the advantages and disadvantages of each model are revealed by comparative examination of the findings. Beginning with the YOLOV8 models, it is clear that version 4 of dataset shown remarkable performance, outperforming all other models in terms of mAP ratings with a mAP50 of 0.974 and a mAP50-95 of 0.754. Notably, it performed exceptionally well in the "Chick" class with a mAP50 of 0.983 and mAP50-95 of 0.765, demonstrating its efficacy in correctly identifying and localizing chicks. YOLOv8V4's performance was consistent across the "Chick," and "Chicken" classes, making it trustworthy model. YOLOv8V5 also performed well, with mAP50 and mAP50-95 values of 0.959 and 0.711, respectively.

YOLOv7 models followed suit, with V1 achieving a mAP50 of 0.961 and mAP50-95 of 0.687, demonstrating consistent performance throughout the "Chick," and "Chicken" categories. Despite the fact that YOLOv7V1 did well, but YOLOv7V4 performed equally good overall with mAP50 of 0.96 and mAP50-95 of 0.706, while YOLOv7V5 performance is slightly behind by obtaining a mAP50 of 0.959. YOLOV5 models showed respectable performance in one of it versions (V4) which performance is beyond all YOLOv7 models, despite not receiving the highest mAP scores, with the mAP50 scores ranging from 0.939 to 0.964 when compare with YOLOV7 and YOLOV8. With the accuracy of 0.974 and 0.734 of, V4 YOLOv8 stood out as having the ability to distinguish between hens and chicks with higher accuracy and outperforms his competitors in terms of mAP50, mAP50-95, recall, and precision, indicating its capacity to precisely recognize objects while avoiding false positives and false negatives. While YOLOv7 and YOLOv5 also perform well in these criteria, they lag behind YOLOv8 a little Table 4 presents these findings.

YOLOv8 have been demonstrated to perform accurately in practical applications, which makes it ideal for our chicken detection, tracking and counting pipeline. The outcomes of employing Mosaic as shown in Figure 5, 6 and 7 makes use of four images by flipping, altering the color gamut as well scaling of four images and combined them to create a single image. It offers two biggest benefits: First, it may enhance the background and tiny object detection, which at the same time expanding the training data and enhancing model robustness. Second, during batch normalization, the data of four images are calculated at once, reducing the dependence on batch size.

Table 4: *Chicken Detection Experimental Results of YOLOv8, YOLOv7 and YOLOv5 Models*

Model	Experimental Datasets	Category	Precision (P)	Recall (R)	mAP@0.5 (mAP50)	mAP@0.5:0.95 (mAP50-95)
YOLO V8	V1	**All**	**0.914**	**0.898**	**0.946**	**0.698**
		Chick	0.959	0.859	0.940	0.693
		Chicken	0.868	0.938	0.952	0.704
	V2	**All**	**0.942**	**0.887**	**0.943**	**0.687**
		Chick	0.961	0.868	0.937	0.681
		Chicken	0.924	0.907	0.949	0.692
	V3	**All**	**0.896**	**0.897**	**0.931**	**0.688**
		Chick	0.899	0.877	0.918	0.676
		Chicken	0.893	0.917	0.943	0.701
	V4	**All**	**0.940**	**0.939**	**0.974**	**0.754**
		Chick	0.964	0.948	0.983	0.765
		Chicken	0.915	0.931	0.965	0.743
	V5	**All**	**0.909**	**0.917**	**0.959**	**0.711**
		Chick	0.910	0.920	0.957	0.714
		Chicken	0.909	0.913	0.961	0.709
YOLO V7	V1	**All**	**0.938**	**0.921**	**0.961**	**0.687**
		Chick	0.967	0.915	0.968	0.690
		Chicken	0.909	0.927	0.954	0.684
	V2	**All**	**0.912**	**0.906**	**0.942**	**0.681**
		Chick	0.914	0.915	0.939	0.688
		Chicken	0.91	0.896	0.945	0.674
	V3	**All**	**0.894**	**0.931**	**0.958**	**0.671**
		Chick	0.903	0.915	0.952	0.644
		Chicken	0.885	0.947	0.963	0.698
	V4	**All**	**0.941**	**0.913**	**0.96**	**0.706**
		Chick	0.943	0.936	0.961	0.709
		Chicken	0.939	0.89	0.959	0.703
	V5	**All**	**0.918**	**0.947**	**0.959**	**0.683**
		Chick	0.925	0.958	0.958	0.678
		Chicken	0.911	0.913	0.961	0.709
YOLO V5	V1	**All**	**0.892**	**0.909**	**0.939**	**0.689**
		Chick	0.874	0.915	0.939	0.683
		Chicken	0.91	0.902	0.939	0.694

	V2	**All**	**0.914**	**0.915**	**0.939**	**0.688**
		Chick	0.928	0.852	0.94	0.679
		Chicken	0.909	0.916	0.956	0.692
	V3	**All**	**0.931**	**0.922**	**0.960**	**0.689**
		Chick	0.948	0.930	0.967	0.676
		Chicken	0.913	0.914	0.952	0.702
	V4	**All**	**0.950**	**0.931**	**0.964**	**0.734**
		Chick	0.964	0.931	0.968	0.744
		Chicken	0.936	0.930	0.960	0.724
	V5	**All**	**0.912**	**0.916**	**0.960**	**0.693**
		Chick	0.914	0.895	0.957	0.674
		Chicken	0.911	0.938	0.964	0.713

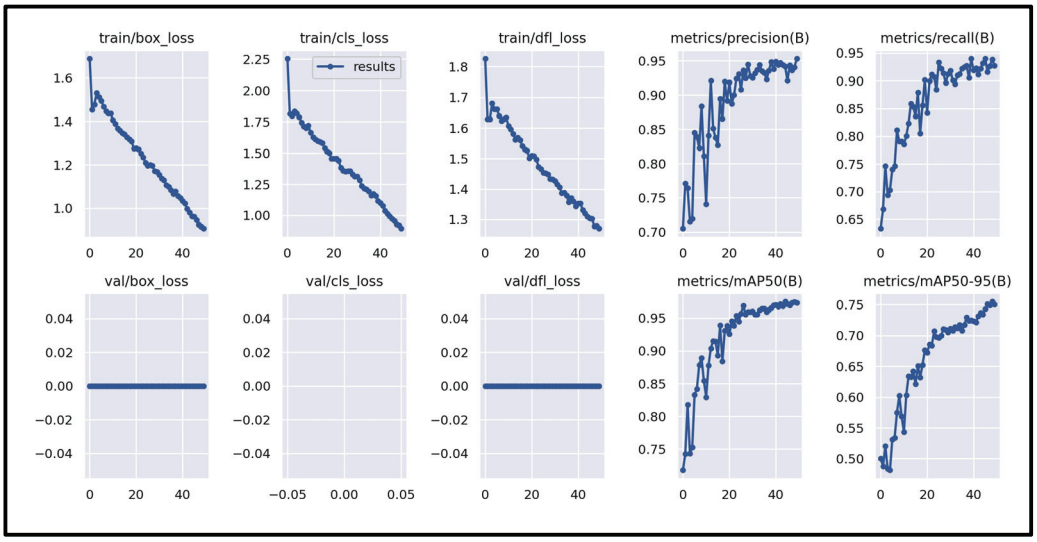

Figure 7: *Training/validation Results from Yolov8V4*

Figure 8: *Chicken Detection Result obtained from Yolov8V4*

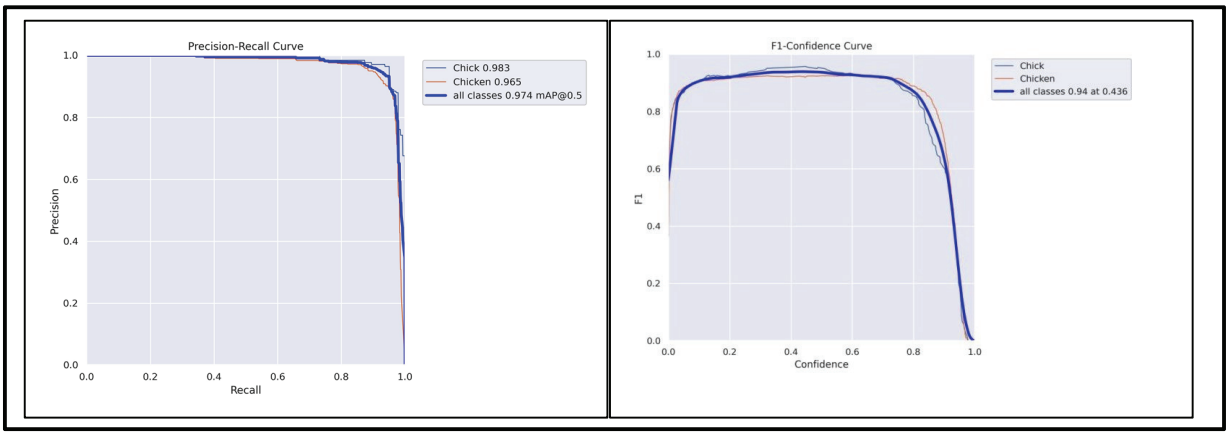

Figure 9: *Precision-Recall Curve and F1 Score from Yolov8V4*

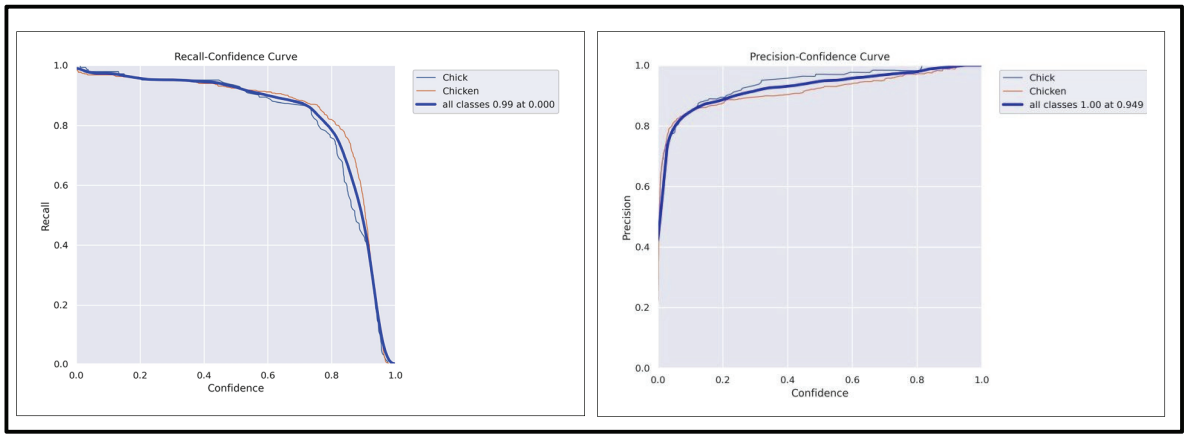

Figure 10: *Precision-Confidence Curve Recall Confidence Curve from Yolov8V4*

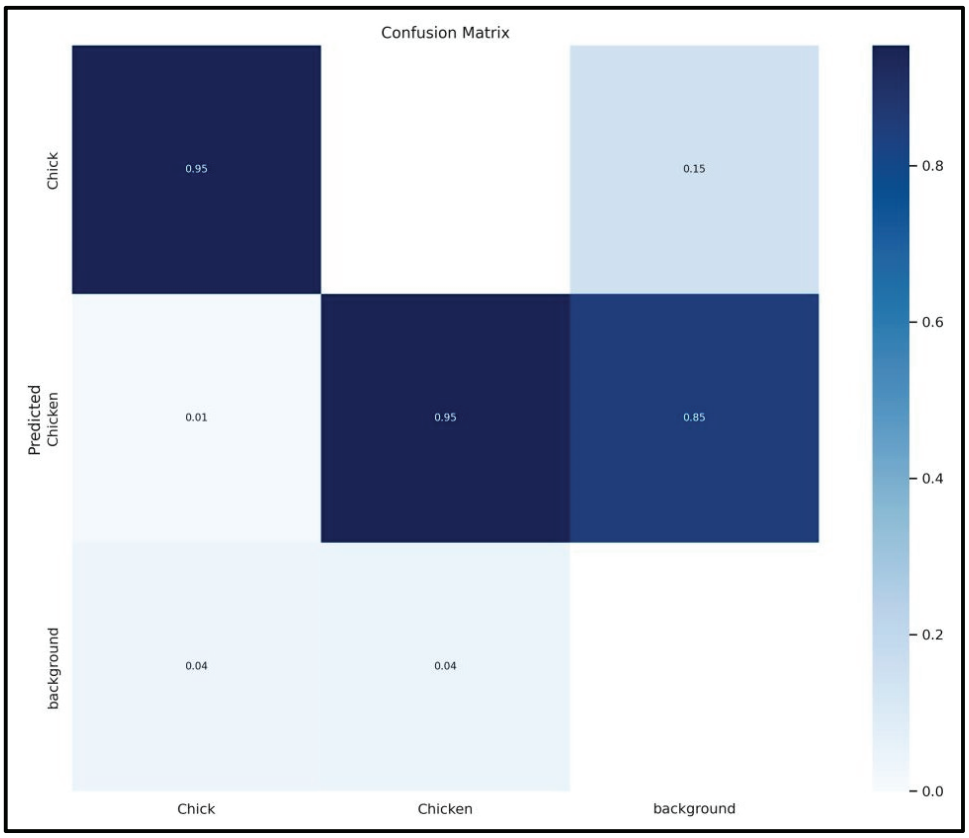
Confusion Matrix
Predicted
Chick
Chicken
background
0.95
0.15
0.01
0.95
0.85
0.04
0.04
Chick
Chicken
background
0.8
0.6
0.4
0.2
0.0

Figure 11: *Confusion Matrix from Yolov8 V4*

4.2 Comparative Analysis of Detection frameworks

Object detection is a common problem in MOT design. The whole process of tracking an object will be in worthless if objects are not appropriately detected, here, Table 3 shows the comparative analysis of the proposed detection framework with the earlier frameworks. Our Model Yolov8 + DeepSort (Proposed) obtained 97.4%, outperforming LC-DenseFCN (Cao et al., 2021) at 93.84%, YOLOv3 (Yao et al., 2020) at 92.23%, and Faster R-CNN with Detectron2 (Čakić et al., 20) at 97%, while being slightly below Faster RCNN with ResNet-101 (Azzahrawan & Dj, 2022) at 92.8% in terms of detection accuracy."

Table 5: Comparison of Chicken Detection, Tracking and Classification Pipelines

4.3 Chicken Tracking and Counting Results

N/S	Algorithms	Authors	Detection	Tracking
	LC-DenseFCN	[17]	93.84 %	97 %
	TabNet	[18]	97%	-
	Faster RCNN, ResNet-101	[21]	92.8 %	-
	YOLOv3	[22]	92.23 %	-
	Yolov8 + DeepSort	**Our Studies**	**97.4 %**	-

In our scenario, the detected chicken acquired from the detector of YOLOv8 is used to track the movement of the chicken. The trajectory of certain chickens that weren't included in the training dataset was determined using the video data. Videos were divided into frames, and the YOLOv8 algorithm find hens in each frame, using the Kalman filter which track individual and multiple chickens, an ID was assigned to the chicken or multiple chicken detected. Additionally, a specific color is assigned to the tracked chicken based on the number and an image calculation; the detected chicken is also given an ID, which is subsequently removed after the chicken is no longer present in the frame, the height and width of the frame were used as region of interest to define a zone and initiate the chicken counting as shown in figure 11. It should be stressed that the effectiveness of tracking and the elimination of false positives are strongly correlated with the effectiveness of the hens' detection module. As a result, the tracking algorithm's performance requires careful optimization through substantial dataset training. By including visual tracking into the IoU, it is possible to compensate for falsely negative chicken recognition and detection, this is the reason why we strongly emphasize on the chicken detection pipeline in this study.

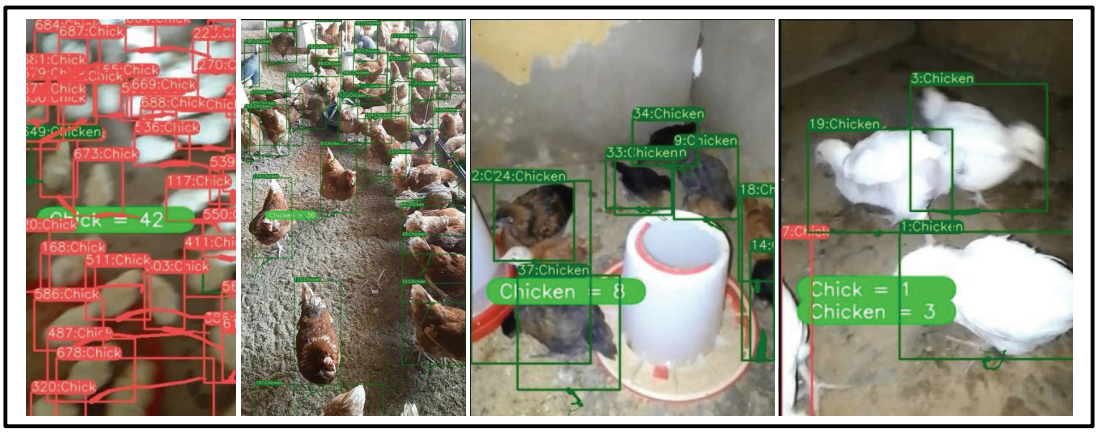

Figure 12: *Chicken Tracker and Counting pipeline*

5. Conclusion

In this paper, we evaluation of YOLOv8, YOLOv7, and YOLOv5 models for chicken detection shows promising results. YOLOv8 version 4 demonstrates remarkable performance, surpass other models with high mAP scores, particularly excelling in correctly identifying and localizing chicks. YOLOv7 models show consistent performance, with V1 and V4 performing equally well. YOLOv5+V4 also demonstrates respectable performance, surpassing YOLOv7 models in certain aspects. The implementation of YOLOv8 + DeepSORT in a chicken tracking and counting pipeline, along with the use of the Kalman filter for tracking, proves effective. The pipeline efficiently tracks individual and multiple chickens, assigns IDs, and removes IDs when chickens are no longer present. The incorporation of the Mosaic technique enhances background and tiny object detection, expanding training data and improving model robustness; the proposed approach could be implemented using IoT and smartphones for remote monitoring of chicken birds. The ability to remotely detect, track, and count chickens can be beneficial for poultry farming and other scenarios where monitoring from a distance is required.

In future, we emphasize the importance of optimizing the tracking algorithm's performance through substantial dataset training. As well collect real-time remote data of individual chickens through the integration smart farming systems and analyse the for signs of illness, stress, or abnormal behaviour and monitor factors like temperature, humidity, and lighting to enhance chicken comfort, growth, and egg production. By including visual tracking into the IoU, falsely negative chicken recognition and detection can be compensated, making the proposed pipeline effective for practical chicken tracking and counting applications.

6. References

[1] C. Okinda, I. Nyalala, T. Korohou, C. Okinda, J. Wang, T. Achieng, P. Wamalwa, T. Mang, M. Shen, A review on computer vision systems in the monitoring of poultry: A welfare perspective, Artificial Intelligence in Agriculture. 4 (2020) 184–208.

[2] M. Kumar, S.P. Dahiya, P. Ratwan, Backyard poultry farming in India: A tool for nutritional security and women empowerment, Biological Rhythm Research. 52 (2021) 1476–1491. https://doi.org/10.1080/09291016.2019.1628396.

[3] The Poultry Site, (2023). https://www.thepoultrysite.com/news/2022/06/global-poultry-market-to-grow-to-493-21-billion-by-2026 (accessed June 18, 2023).

[4] M.H. Lashari, A.A. Memon, S.A.A. Shah, K. Nenwani, F. Shafqat, Iot based poultry environment monitoring system, in 2018 IEEE International Conference on Internet of Things and Intelligence System (IOTAIS), IEEE, 2018: pp. 1–5.

[5] S. Neethirajan, Automated tracking systems for the assessment of farmed poultry Animals. 12 (2022) 232.

[6] M.D.N. Brochu, Pathogen and Disease Prevalence, and Demographic Characteristics of Ontario Small Poultry Flocks, (2019).

[7] S. Neethirajan, ChickTrack–a quantitative tracking tool for measuring chicken activity, Measurement. 191 (2022) 110819.

[8] K.J. Dsouza, S.A. Muthukumaraswamy, A Robotics-Based Surveillance System for Livestock Wellbeing and Early Disease Detection in Poultry Farms, in 2023 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE), IEEE, 2023: pp. 221–225.

[9] G. Abbas, S. Jaffery, A.H. Hashmi, A.J. Tanveer, M. Arshad, Q.A. Amin, M.I. Saeed, M. Saleem, R.A.M. Qureshi, A.A. Khan, PROSPECTS AND CHALLENGES OF ADOPTING AND IMPLEMENTING SMART TECHNOLOGIES IN POULTRY PRODUCTION., Pakistan Journal of Science. 74 (2022).

[10] S. Gurung, Development of Novel Euthanasia and Depopulation Methods for Neonatal Poultry and Caged Laying Hens, (2017).

[11] D. Sinwar, V.S. Dhaka, M.K. Sharma, G. Rani, AI-based yield prediction and smart irrigation, Internet of Things and Analytics for Agriculture, Volume 2. (2020) 155–180.

[12] N. Li, Z. Ren, D. Li, L. Zeng, Automated techniques for monitoring the behaviour and welfare of broilers and laying hens: towards the goal of precision livestock farming, Animal. 14 (2020) 617–625.

[13] W. Shang, K. Sohn, D. Almeida, H. Lee, Understanding and improving convolutional neural networks via concatenated rectified linear units, in: International Conference on Machine Learning, PMLR, 2016: pp. 2217–2225.

[14] G. Jocher, A. Stoken, J. Borovec, L. Changyu, A. Hogan, L. Diaconu, J. Poznanski, L. Yu, P. Rai, R. Ferriday, ultralytics/yolov5: v3. 0, Zenodo. (n.d.).

[15] C.-Y. Wang, A. Bochkovskiy, H.-Y.M. Liao, YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors, ArXiv E-Prints. (2022) arXiv-2207.

[16] G. Jocher, A. Chaurasia, J. Qiu, YOLO by Ultralytics. 2023, URL: Https://Github. Com/Ultralytics/Ultralytics. (2023).

[17] L. Cao, Z. Xiao, X. Liao, Y. Yao, K. Wu, J. Mu, J. Li, H. Pu, Automated chicken counting in surveillance camera environments based on the point supervision algorithm: LC- DenseFCN, Agriculture. 11 (2021) 493.

[18] G. Ahmed, R.A.S. Malick, A. Akhunzada, S. Zahid, M.R. Sagri, A. Gani, An approach towards IoT-based predictive service for early detection of diseases in poultry chickens, Sustainability. 13 (2021) 13396.

[19] D. Cheng, T. Rong, G. Cao, Density map estimation for crowded chicken, in: Image and Graphics: 10th International Conference, ICIG 2019, Beijing, China, August 23–25, 2019, Proceedings, Part III 10, Springer, 2019: pp. 432–441.

[20] A.L.R. Siriani, V. Kodaira, S.A. Mehdizadeh, I. de Alencar Nääs, D.J. de Moura, D.F. Pereira, Detection and tracking of chickens in low-light images using YOLO network and Kalman filter, Neural Computing and Applications. 34 (2022) 21987–21997.

[21] M.R. Azzahrawan, E.C. Djamal, Counting Broiler from Poultry House Video Using Faster Region-based Convolutional Neural Networks, in: 2022 International Seminar on Intelligent Technology and Its Applications (ISITIA), IEEE, 2022: pp. 199–203.

[22] Y. Yao, H. Yu, J. Mu, J. Li, H. Pu, Estimation of the gender ratio of chickens based on computer vision: Dataset and exploration, Entropy. 22 (2020) 719.

[23] A. Bewley, Z. Ge, L. Ott, F. Ramos, B. Upcroft, Simple online and realtime tracking, in: 2016 IEEE International Conference on Image Processing (ICIP), IEEE, 2016: pp. 3464– 3468.

[24] Y. Zhang, C. Wang, X. Wang, W. Zeng, W. Liu, Fairmot: On the fairness of detection and re-identification in multiple object tracking, International Journal of Computer Vision. 129 (2021) 3069–3087.

[25] C. Li, G. Dobler, X. Feng, Y. Wang, Tracknet: Simultaneous object detection and tracking and its application in traffic video analysis, ArXiv Preprint ArXiv:1902.01466. (2019).

[26] X. Zhou, V. Koltun, P. Krähenbühl, Tracking objects as points, in: European Conference

on Computer Vision, Springer, 2020: pp. 474–490.

[27] S. Doll, N. Hanselmann, L. Schneider, R. Schulz, M. Enzweiler, H. Lensch, STAR-Track: Latent Motion Models for End-to-End 3D Object Tracking with Adaptive Spatio- Temporal Appearance Representations, ArXiv Preprint ArXiv:2306.17602. (2023).

[28] K. Saho, Kalman filter for moving object tracking: Performance analysis and filter design, Kalman Filters-Theory for Advanced Applications. (2017) 233–252.

[29] H. Ghorbani, Mahalanobis distance and its application for detecting multivariate outliers, Facta Universitatis, Series: Mathematics and Informatics. (2019) 583–595.