# Fine-tuning faster region-based convolution neural networks for detecting poultry feeding behaviors

Xue Hui[1†], Delin Zhang[2†], Wei Jin[3], Yichang Ma[3], Guoming Li[4*]

(1. *Beijing Eastmage Architectural Design Go. Ltd., Beijing 100083, China*;
2. *Asset Management company of China Agricultural University, Beijing 100083, China*;
3. *College of Animal Science and Technology, China Agricultural University, Beijing 100092, China*;
4. *Department of Poultry Science, University of Georgia, Athens, Georgia State, 30602, USA*)

**Abstract:** Poultry feeding behaviors provide valuable information for system design and farm management. This study developed poultry feeding behavior detectors using the faster region-based convolution neural network (faster R-CNN). Twenty 50-day-old Jingfen layer pullets were kept in four experimental compartments and could freely move between adjacent ones. Four light colors (white, red, green, and blue) were supplied to create environmental variations for detector development. A camera was installed atop each compartment to capture images for detector development. Several hyperparameters were fine-tuned to determine the optimal one. Based on the trade-off strategies between detection accuracy and processing speed, the following strategies were deployed to develop the detector: feature extractor of inception V2, the model trained with common objects in context dataset, fixed_shape_resizer with the size of 600×600 pixels, kernel stride of 8300 proposals, and dynamic learning rate. The final detector had 95.7% recall, 94.2% average precision, 94.9% F1 score, 23.5 mm root mean square error, and 8.3 fps processing speed, indicating decent performance for detecting poultry feeding behaviors. With the trained detector, temporal and spatial feeding behaviors of individual poultry can be successfully characterized. It is concluded that the faster R-CNN should be a useful tool to continuously monitor poultry feeding behaviors in group settings.

**Keywords:** faster R-CNN, feeding behavior, poultry, feature extractor, hyperparameter
**DOI:** 10.25165/j.ijabe.20251801.6344

**Citation:** Hui X, Zhang D L, Jin W, Ma Y C, Li G M. Fine-tuning faster region-based convolution neural networks for detecting poultry feeding behaviors. Int J Agric & Biol Eng, 2025; 18(1): 64–73.

## 1 Introduction

Poultry feeding is undoubtedly among core concerns for farmers as it can influence economic benefits and reflect poultry physiological and welfare status. Poultry feeding responses vary with different environmental stimuli[1], rearing systems, genetics[2], social interactions[3], and sensory factors[4]. Abnormal feeding behaviors are identified as a sign of illness for food animal[5]. As such, assessment of poultry feeding behaviors provides scientific evidence for welfare-based management and efficacy of resource allowance. In modern intensive poultry production systems, monitoring individual poultry's feeding behaviors during the whole lifetime is almost impossible for farmers and researchers, due to the labor- and time-intensive nature of the task. Automated solutions to assist farm management are warranted.

Current precision livestock farming (PLF) technologies provide a possibility to automatically measure poultry feeding behaviors. For example, weighing scales were utilized to monitor poultry feed intake in real-time[1]. Although the systems could accurately inspect the feeding behaviors of group-housed poultry in the weighing area, they could not differentiate individual feeding poultry, thus missing valuable information of individual variations. Radio frequency identification (RFID) systems offered solutions to detect multiple individual feeding birds by registering tagged birds in detecting ranges of antennas[6,7]. However, the systems are rather expensive and complex to implement in poultry houses. The attached tags need frequent adjustment to avoid bird discomfort as poultry grow bigger, and they may get lost during the inspection period. Furthermore, the sensitivity of the systems may be blocked by litter, metal materials, and animals. These challenges limit the system application in lab scales rather than commercial scales. Image processing was another technology used to continuously monitor poultry feeding in commercial scales because it was low-cost and non-invasive[8,9]. Nevertheless, the processing algorithms were subject to complexity of image background, environmental conditions, bird sizes, and bird postures, thus resulting in poor generalization. Additionally, the above-mentioned methods mainly recorded temporal poultry feeding behaviors (e.g., feeding time, feeding bout, feeding duration per bout, etc.) and ignore spatial information including inter-distance of feeding birds and bird distribution along feeders, which hinders better understanding of poultry feeding behaviors and resource allowance. Convolutional neural networks (CNNs) are increasingly applied in livestock farming to facilitate farm management and may have the potential to overcome the above-mentioned drawbacks of the technologies[10,11].

Convolutional neural networks can detect poultry without introducing human interference and invasion. The CNNs can adapt

**Received date:** 2022-12-07 **Accepted date:** 2024-07-11
**Biographies: Xue Hui**, Assistant Professor, research interest: precision Livestock and Farm design, Email: 350047289@qq.com; **Delin Zhang**, Senior engineer, research interest: Agricultural Engineering, Email: 332511898@qq.com; **Wei Jin**, research assistant, research interest: precision Livestock and Farm design, Email: jinwei_cau@126.com; **Yichang Ma**, Research Assistant, research interest: precision livestock and farm design Email: mayichang1993@126.com.
†Authors make the same contribution to this manuscript
***Corresponding author: Guoming Li**, Assistant Professor, research interest: Precision Livestock and Poultry Farming, Agriculture Engineering, Computer visions, Applied Artificial Intelligence. 120 D.W. Brooks Drive, Athens, GA 30602, USA. Tel: 1-706-542-0150; Email: gmli@uga.edu.

to various detection environments if they are fine-tuned[10]. Based on some geometric features of bounding boxes around objects of concern, the CNNs may have the potential to track individual poultry continuously. Quite a few efficient CNN models have been developed in recent years, varying in function, performance, and architecture. Among them, the faster region-based CNN (faster R-CNN) had decent accuracy and processing speed for object detection based on our previous experiment[11]. The faster R-CNN has been widely applied for a variety of agriculture applications and showed good generalization among these applications (Table 1). Thus, the faster R-CNN may have the potential for detecting poultry feeding behaviors, though the detection performance remains to be verified.

**Table 1 Applications of faster region-based convolutional neural network in precision agriculture**

| Feature extractor | Detected object | Performance | Reference |
|---|---|---|---|
| — | Fuji apple | 94.8% AP and 89.8% F1 score | [12] |
| VGG | Apple | 81.4% recall, 83.5% accuracy, 81.4% AP, 81.8% F1 score, and 86.3% IOU | [13] |
| VGG19 | Apple | 87.2% precision, 83.6% recall, 85.2% F1 score, 85.8% IOU, and 7.9 fps processing speed | [14] |
| Darknet53 | Litchi | 97.6% AP and 1.5 fps processing speed | [15] |
| Inception V2 | Floor egg | 91.9%-94.7% precision, 99.8-100.0% recall, and 91.9%-94.5% accuracy | [16] |
| Resnet50 and Darknet53 | Aphid | 85.4% AP, 61.7% mAP, and 9.0 fps processing speed | [17] |
| ResNet50 | Pullet drinking behavior | 88.2% precision, 88.7% recall, 89.4% specificity, and 89.1% accuracy | [18] |
| Inception V2, ResNet50, ResNet101 and Inception-ResNet V2 | Pig standing and lying behaviors | 77.0%-93.0% precision and 80.0%-91.0% mAP | [19] |
| — | Passion fruit | >91.5% accuracy | [20] |
| — | Dairy goat | 92.5% average precision and 20.0 fps processing speed | [21] |
| — | Cow in pasture and feedlot | 90.0%-95.0% accuracy and 80.0-89.0% AP | [22] |
| ZF-net | Pig feeding behavior | 99.6% precision and 86.9% recall | [23] |
| MobileNet | Sow drinking, urination, mounting behaviors | 93.5%-97.0% accuracy, 95.1% mAP, and 0.3 fps processing speed | [24] |
| — | Sow standing, sitting, recumbency behaviors | 95.0%-99.1% precision, 93.9-99.3% recall, 88.3% -95.5% accuracy, and 9.0-26.3 fps processing speed | [25] |

Note: — indicates missing information in a reference. AP is average precision, mAP is mean average precision, IOU is intersection over union, and fps is frames per second.

The objective of this research was to develop faster R-CNN feeding behavior detectors to detect feeding behavior using layer pullets as examples. The performance of feeding detection was compared for various feature extractors, pre-trained models, image resizers, sizes of kernel stride, number of proposals, and learning rate. With the trained detector, temporal and spatial feeding behaviors of individual poultry were characterized.

## 2 Materials and methods

### 2.1 Animal, housing, and management

The experiment was conducted at China Agricultural University. Twenty 50-day-old Jingfen layer pullets (Jingfen; Beijing Huadu Yukou Poultry Co., Ltd., Beijing, China) were kept in a lighting preference test system containing four compartments with each measuring 1.2 m (length)×0.96 m (width)×2.0 m (height)[26]. When 20 pullets were present together in a compartment, the stocking density was 361 cm²/pullet, which was higher than that recommended by Hy-Line International 2013)[27] for cage-reared pullets (100 to 200 cm²/pullet). The major purpose of this research is to evaluate the model rather than analyze bird behavior. Dynamic numbers of birds in each compartment can create variations for model development. The four compartments were arranged in a straight line, and adjacent compartments were connected with an open small door (0.3 m in width and 0.4 m in height). Such a design can ensure that birds go through different compartments and produce various numbers of feeding birds in a compartment, which can create image variations for detector development. One trough feeder (0.96 m wide) was installed with each compartment, and an infrared camera (V1.1.0, Zhejiang Dahua Technology Co., Ltd., Hangzhou, China) was mounted atop each compartment to monitor pullet activity. Four light colors (white, green, red, and blue) were assigned to respective compartments and introduced extra image variations for detector development. White light is typically used in poultry production. Green and blue lights are beneficial for bird body development, and red light can stimulate bird movement. Understanding the feeding behaviors under the four light colors provides critical insights in precision poultry management. The light intensity was 0.1 W/m² at bird head level, and the light program was 12L:12D (lights ON at 8:00 and OFF at 20:00). Temperature and relative humidity were maintained at (23.1±0.5)°C and (20±1)%.

### 2.2 Feeding behavior definition and labeling

The recorded videos were converted into images at a rate of one frame per second (1 fps), and each image was 1280×720 pixels. Pullets were defined as feeding when their heads were present atop the feeder trough. Images from 9-10 d with 10 min intervals were labeled using the open-source software (LabelImg), and annotations were saved as .xml files in PASCAL VOC format for further processing. Five thousand images containing at least one feeding bird per image were selected from each compartment, resulting in a total of 20 000 images for detector training, validation, and testing.

### 2.3 Faster region-based convolutional neural network

Faster R-CNN is an extension of R-CNN and fast R-CNN proposed by Ren et al.[28]. The structure of the network is shown in Figure 1. Faster R-CNN has two stages: the first stage is the region proposal network (RPN), and the second stage is the box classifier using the proposed regions for prediction. In Figure 1, an input image is fed into the feature extractor to produce feature maps. The RPN runs on the maps to proposed regions which include target objects, feeding birds in this case. The regions are tiled onto the feature maps. Then regions of interest (ROI) are proposed and

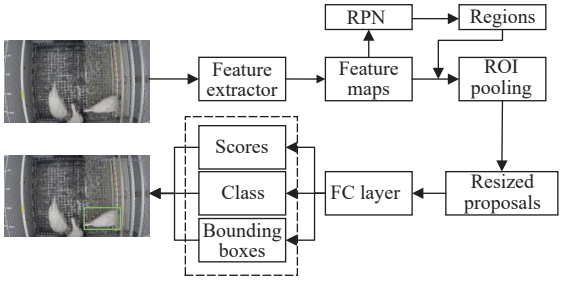



Figure 1 Schematic illustration of the faster region-based convolutional neural network (RPN is region proposal network and ROI is region of interest. A feeding bird is enclosed within a green bounding box after the network processing)

pooled using average/max pooling. As a result, the ROI/proposals are resized into uniform sizes, and the resized proposals are connected with fully-connected (FC) layers to produce scores, class, and bounding boxes, which are visualized in the original images simultaneously.

### 2.4 Overall strategy for training, validation, and testing

The 20 000 images were stratified into training, validation, and testing sets. The training set had 13 322 images and 28 347 feeding birds, the validation set had 1200 images and 3508 feeding birds, and the testing set had 4000 images and 10 376 feeding birds, as listed in Table 2. The detectors were trained with the training set, and resultant models were saved during training in specific iteration periodically and validated with the validation set. The training and validation losses, which can reflect the amount of deviation between ground truth and prediction, were evaluated as proposed by Ren et al.[28] and visualized in an open-source platform, Tensor Board. Based on observation, if the training and validation losses both kept decreasing, the models may be underfitted and need more training; while if the training loss kept decreasing but validation loss rebounded to increase, the models may be overfitted. To that end, training should be stopped, and final models were saved accordingly for evaluation on the hold-out dataset, the testing set. The losses were other aspects to be observed during the training, and the training was deemed to be completed when training and validation losses tended to be stable within several iterations. The computer system used for detector training, validation, and testing was equipped with 32 GB RAM, Intel(R) Core (TM) i7-8700K processor, and NVIDIA GeForce GTX 1080 GPU card (Dell Inc., Round Rock, TX, USA).

**Table 2 Data distribution for detector training, validation, and testing**

| | Training | Validation | Testing |
|---|---|---|---|
| Number of images | 13 322 | 1200 | 4000 |
| Number of feeding birds | 28 347 | 3508 | 10 376 |

### 2.5 Modifications for detector development

The modifications for detector development included feature extractor, pre-trained model, image resizer, kernel stride, number of region proposals, and learning rate. Unless specified in the sections, the following modifications were trained with ResNet101 feature extractor, COCO-trained model, keep_aspect_ratio_resizer with 600×600 pixels, 16 kernel stride, 300 proposals, and dynamic learning rate.

#### 2.5.1 Feature extractor

Six feature extractors were embedded into the faster R-CNN detector and evaluated to determine the most efficient ones, which were Inception V2, ResNet50, ResNet101, and Inception ResNet V2. The overall structure of these extractors is shown in Figure 2. Inception V2 deploys a factorized design and multiple-sized filters in parallel[29]. The resultant features are concatenated to form feature maps. ResNet50 and ResNet101 are the ResNets with 50 and 101 layers, respectively. The ResNets used a shortcut connection to build convolution blocks and feed-forward convolution to process inputs[30]. Inception ResNet V2 is the Inception network adding residual connection[31,32]. The order of complexity for these feature extractors is: Inception V2 < ResNet50 < ResNet101 < Inception ResNet V2.

#### 2.5.2 Pre-trained model

The pre-trained models involved in this study were the models trained with the COCO dataset[33], KITTI dataset[34], FGVC dataset, and AVA dataset[35]. Hereafter, they are abbreviated as COCO-trained model, KITTI-trained model, FGVC-trained model, and AVA-trained model, respectively. The COCO is a generic dataset, the KITTI is a bird- and car-based dataset, the FGVC is a car-based dataset, and the AVA is a human-based dataset. These are commonly used datasets and easily accessed.

#### 2.5.3 Image resizer

Three modes of resizers combined with five sizes were evaluated. The mode of "identity_resizer" did not change the image size, which was 1280×720 pixels. The other two modes were 'keep_aspect_ratio_resizer' and 'fixed_shape_resizer', combining with the two sizes of 600×600 pixels and 1500×1500 pixels. Under the former mode, the length and width for an input image were enlarged/diminished to the required sizes, in which the length-to-width ratio (16/9 in this case) was changed. Under the latter mode, the ratio of the image was consistent and differences between the original image and resized images were padded with zero matrices. Samples of resized images with the resizers can be found in Figure 3.

#### 2.5.4 Size of kernel strides for feature extraction and grid anchor generation

Figure 4 shows an example of the convolution process in which a 4×4 kernel with stride of 8 runs on a 12×12 input results in a 2×2 output. The kernel sizes of 8 and 16 were trained and evaluated to determine the optimal one since they are commonly used for deep learning[28].

#### 2.5.5 Numbers of proposals after region proposal network

Three numbers of region proposal network were tested: 100, 300, and 500. The 300 was the default setting by the original author[28]. The 100 proposals were deemed sufficient to cover a maximum of 20 pullets in a compartment. The 500 was a level higher than the default setting and used for comparison. Higher than 500 proposals were thought to diminish the processing speed and were not considered in this case.

#### 2.5.6 Overall learning rate

Two types of learning rates were trained and compared. The first one was a constant learning rate of 0.003, and the second one was a dynamic learning rate with 0.003 for 0-10 000 iterations, 0.0003 for 10 000-20 000 iterations, and 0.000 03 for 20 000-30 000 iterations. In sum, a step-decrease learning rate can help to optimize the training loss and obtain an optimal model[36]. The two learning strategies are shown in Figure 5.

After performance comparison of the above-mentioned feature extractors and training hyperparameters, the optimal ones were selected to develop the feeding behavior detector.

### 2.6 Evaluation metrics

After the detectors were trained and validated, the hold-out testing set was used for evaluating the trained detectors as described in Section 2.4. The intersection over union (IOU) was used to determine whether a feeding bird was correctly detected (Equation 1), with greater than 0.5 being true positive.

$$\mathrm{IOU} = \frac{Area\ of\ ground\ truth\ box \cap Area\ of\ predicted\ box}{Area\ of\ ground\ truth\ box \cup Area\ of\ predicted\ box} \quad (1)$$

Precision, recall, and F1 score for detecting each feeding bird in the images were calculated using Equations 2-4. Precision is the ratio of correctly detected feeding birds to total detected feeding birds. Recall is the ratio of correctly detected feeding birds to total manually labeled feeding birds. F1 score is the harmonic mean of precision and recall and a balance metric on comprehensively evaluating false feeding and non-feeding cases. For the three

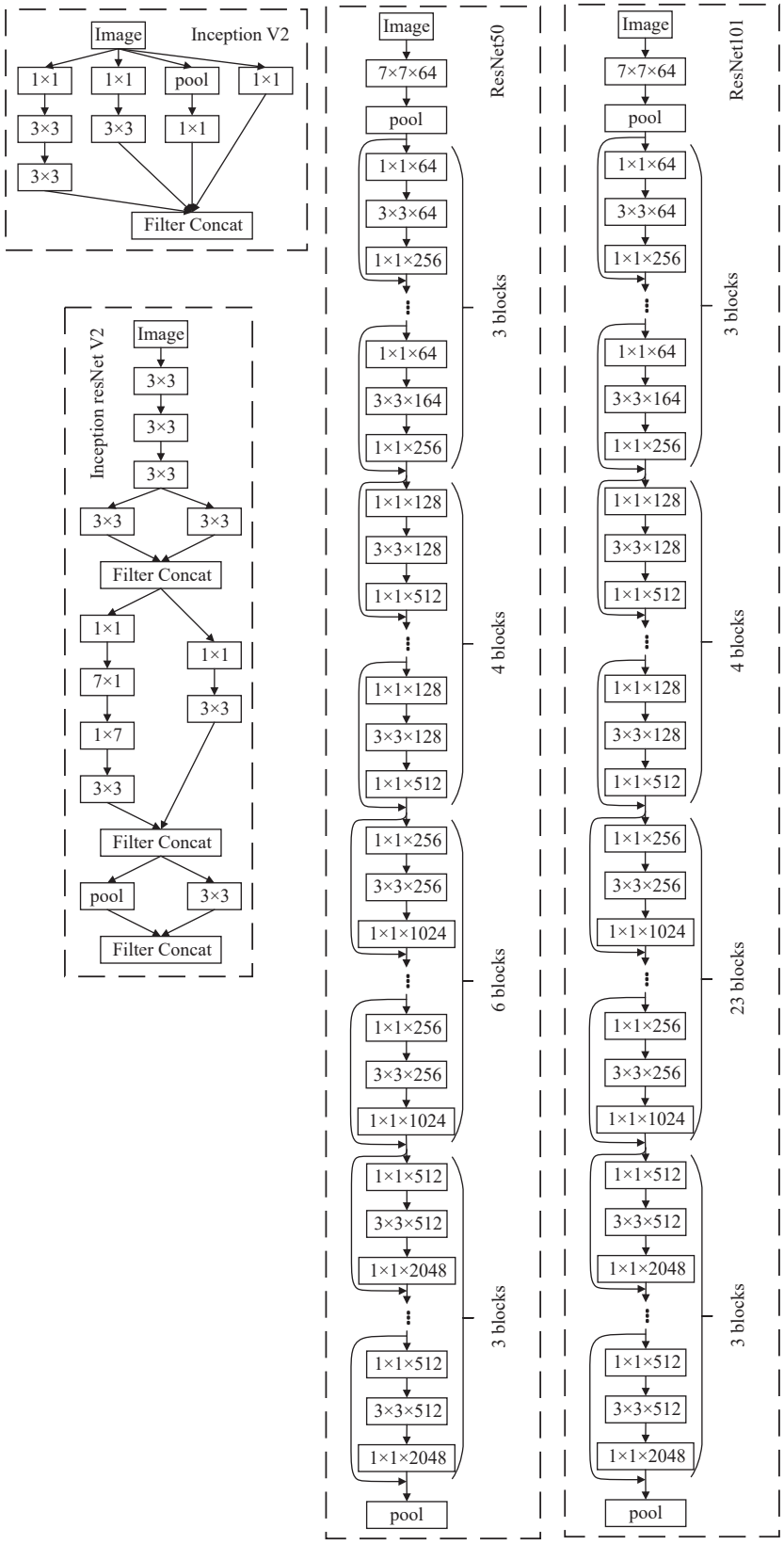



Figure 2 Architecture for the feature extractors of Inception V2, ResNet50, ResNet101, and Inception ResNet V2 ("ResNet" is residual network and "concat" is concatenate)

metrics, a closer to 100% value reflects better detection performance of a detector.

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

$$\text{F1 score} = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{4}$$

where, $TP$ is true positive, i.e., number of cases that a detector successfully detects an existent feeding bird in an image with IOU greater than 0.5; $FP$ is false positive, i.e., number of cases that a detector reports a nonexistent feeding bird in an image, or IOU is less than 0.5; and $FN$ is false negative, i.e., number of cases that a

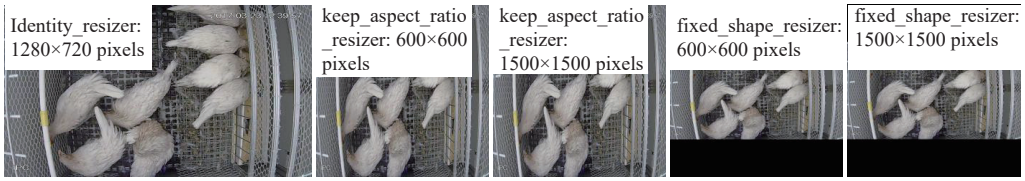



Figure 3 Samples of resized images with the five resizers. It should be noted that the images are arranged for presentation and are not in real sizes (The real sizes of the resized images can be found in the white solid rectangles)

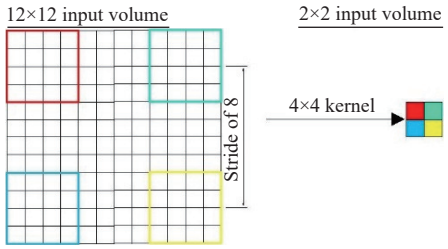



Figure 4 Schematic illustration of the convolution process with a 12×12 input, a 4×4 kernel, stride of 8, and 2×2 output

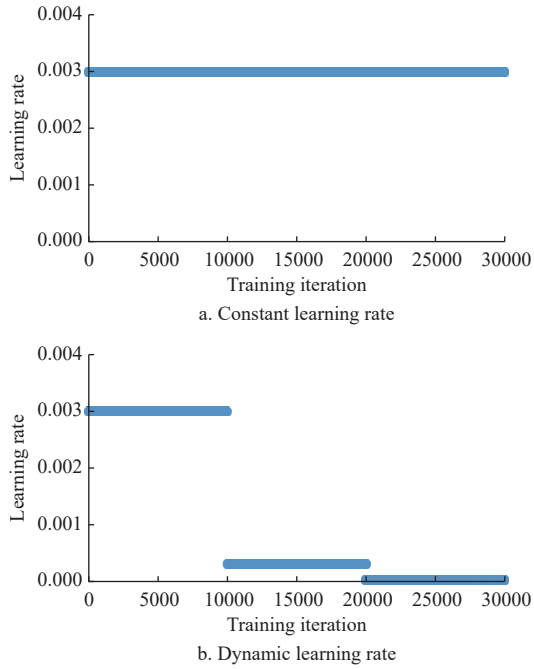



Figure 5 Learning rates during the 30 000 training iterations

detector fails to detect an existent feeding bird in an image.

Average precision (AP) summarizes the shape of the precision-recall curve and is defined as the mean precision at a set of 11 equally spaced recall levels [0, 0.1, …, 1] (Equations (5) and (6))[37].

$$\mathrm{AP}\left[\%\right] = \frac{1}{11}\sum_{r\in\{0,0.1,\ldots,1\}} P_{interp}(r) \tag{5}$$

$$P_{interp}(r) = \max_{\tilde{r}:\tilde{r}\geq r} P(\tilde{r}) \tag{6}$$

where, $r$ is level of recall at {0, 0.1, …, 1}; $P_{interp}(r)$ is interpolated precision in the precision-recall curve when recall is $r$; $\tilde{r}$is recall within a wiggle piece; and $P(\tilde{r})$ is measured precision at recall $\tilde{r}$.

Root mean square error (RMSE) of the feeding bird location predicted by the detectors was calculated using Equation 7. The RMSE reflects the location deviation of a predicted feeding bird from its actual location. A conversion factor of 1.8 mm/pixel was validated and used to convert pixel-based coordinates to real distance.

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{N}\left\{(\hat{x}_i - x_i)^2 + (\hat{y}_i - y_i)^2\right\}}{N}} \tag{7}$$

where, $\hat{x}_i$ and $\hat{y}_i$ are the predicted center coordinates of $i^{\text{th}}$ egg; $x_i$ and $y_i$are the $i^{\text{th}}$ manually labeled center coordinates; and $N$ is the total number of feeding birds in the images.

Processing time reported by Python 3.6 was used to evaluate the processing speed of the detectors for processing 4000 images. Processing speed (fps) was obtained by dividing the total images (4000) by processing time.

Because the threshold of performance difference varied among previous investigations, the threshold was set specifically for the dataset. The precision, recall, F1 score, and AP were different among the settings when the differences were over 2%, which can help to differentiate the effects of different settings. The RMSE was different among the settings when the difference was over 10 mm, which was 3% of the width of the poultry body. The processing speed was different among the settings when the difference was over 1 fps.

### 2.7 Automated behavior measurement

With the final detector, one hour (11:00-12:00) of feeding data was measured for individual pullets at 50 d of age and under white light. The feeding behaviors were characterized as total feeding time (min), the total number of feeder visits (times), average feeding duration per visit (min/time), frequency of individual feeding duration (min/time), frequency of the number of simultaneously feeding birds, frequency of inter-bird distance of feeding birds (mm), and spatial distribution of feeding birds along the feeder trough.

## 3 Results

### 3.1 Detection samples

With the trained detector, individual feeding birds were identified and enclosed with green bounding boxes, accompanied with a class name and a confidence score. As shown in Figure 6, the feeding birds were well categorized automatically under different light colors and feeding scenarios.

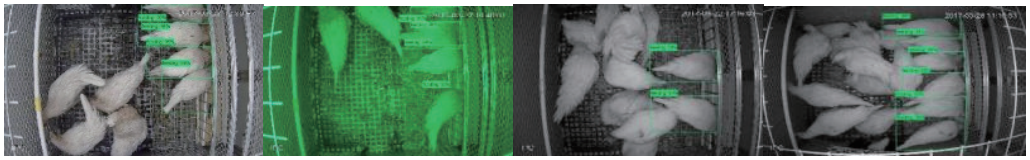


Figure 6 Samples of detected feeding birds using the faster region-based convolution neural network feeding behavior detector (light colors from left to right figures are white, red, green, and blue light, respectively)

### 3.2 Performance of five feature extractors

Table 3 shows the detection of various feature extractors. The AP (98.7-98.8%) and RMSE (22.1-24.9 mm) were similar among the four feature extractors. The Inception V2 had comparable precision (95.7%), recall (97.2%), and F1 score (96.4%) with ResNet50 and ResNet101 but the fastest processing speed (8.1 fps). The ResNet50 and ResNet101 performed similarly in feeding detection with regard to all detection performance indicators. The Inception ResNet V2 had the highest recall (98.7%) but the lowest precision (87.7%) and F1 score (92.9%) and the slowest processing speed (2.1 fps). As the Inception V2 performed on equivalent or higher levels for most of the detection performance indicators and had the highest processing speed, it was selected as the feature extractor to develop the feeding behavior detector in this case.

**Table 3 Detection performance of pullet feeding behaviors using five feature extractors**

| Feature extractor | Precision/ % | Recall/ % | F1 score/% | AP/% | RMSE/ mm | Processing speed/fps |
|---|---|---|---|---|---|---|
| Inception V2 | 95.7 | 97.2 | 96.4 | 98.7 | 22.1 | 8.1 |
| ResNet50 | 95.0 | 97.4 | 96.2 | 98.8 | 25.8 | 5.2 |
| ResNet101 | 96.8 | 96.0 | 96.4 | 98.8 | 26.5 | 4.7 |
| Inception ResNet V2 | 87.7 | 98.7 | 92.9 | 98.7 | 24.9 | 2.1 |

Note: AP is average precision and RMSE is root mean square error.

### 3.3 Performance of the detectors trained with the weights from four pre-trained models

Table 4 shows the detection performance using different pre-trained models. The COCO-trained model performed similarly, in terms of all performance indicators, with the KITTI-trained model. The FGVC-trained model had the lowest recall (89.0%); second lowest precision (63.0%), F1 score (73.8%), and AP (82.2%); second highest RMSE (38.9 mm); and second slowest processing speed (4.9 fps). The AVA-trained model had the lowest precision (39.0%), F1 score (55.5%), and AP (70.0%), largest RMSE (51.9 mm), and slowest processing speed (4.2 fps). As the COCO-trained model had comparable or even higher performance than others and was trained with the most generic benchmark dataset, COCO dataset, it was utilized to develop the feeding behavior detector.

**Table 4 Detection performance of pullet feeding behaviors trained with the weights from four pre-trained models**

| Pre-trained models | Precision/ % | Recall/ % | F1 score/% | AP/ % | RMSE/ mm | Processing speed/fps |
|---|---|---|---|---|---|---|
| COCO-trained model | 92.9 | 97.9 | 95.3 | 99.0 | 22.4 | 5.1 |
| KITTI-trained model | 94.2 | 97.2 | 95.7 | 98.5 | 21.8 | 5.2 |
| FGVC-trained model | 63.0 | 89.0 | 73.8 | 82.2 | 38.9 | 4.9 |
| AVA-trained model | 39.0 | 96.0 | 55.5 | 70.0 | 48.7 | 4.2 |

Note: AP is average precision and RMSE is root mean square error.

### 3.4 Performance of various image resizers

Table 5 shows that the average performance of keep_aspect_ratio_resizer and fixed_shape_resizer was similar. However, the image sizes of 600×600 pixels had at least 2.1% higher recall and 1 fps faster processing speed than other sizes. The precision, F1 score, AP, and RMSE were mostly similar among all the resizers, except for the keep_aspect_ratio with the size of 1500×1500 pixels, which had the lowest F1 score of 93.3%. Hence, the fixed_shape_resizer with the size of 600×600 pixels was selected to develop the detector.

### 3.5 Performance of various kernel strides

The performance of the two kernel strides was compared (Table 6). The strides of 8 had 2.5% higher precision (96.0%) but 1.4 fps lower processing speed (3.7 fps) than those of 16. Other performance indicators were similar between the two strides. As the two strides had comparable detection performance, the one (stride of 8) with less erroneous detection was preferred, despite slightly compromising processing speed.

**Table 5 Detection performance of pullet feeding behaviors for different image resizers**

| Mode of image resizers | Size/ pixels | Precision/ % | Recall/ % | F1 score/% | AP/ % | RMSE/ mm | Processing speed/fps |
|---|---|---|---|---|---|---|---|
| Identity_resizer | 1280×720 | 94.3 | 95.4 | 94.9 | 98.9 | 24.1 | 3.9 |
| Keep_aspect_ ratio_resizer | 600×600 | 94.2 | 97.6 | 95.9 | 99.0 | 22.1 | 5.1 |
| | 1500×1500 | 93.0 | 93.6 | 93.3 | 98.4 | 26.2 | 3.6 |
| Fixed_shape_resizer | 600×600 | 94.7 | 97.5 | 96.1 | 99.1 | 23.8 | 4.9 |
| | 1500×1500 | 94.0 | 95.8 | 94.9 | 98.8 | 26.1 | 2.8 |

Note: AP is average precision and RMSE is root mean square error.

**Table 6 Detection performance of pullet feeding behaviors using two kernel strides**

| Kernel strides | Precision/ % | Recall/ % | F1 score/% | AP/ % | RMSE/ mm | Processing speed/fps |
|---|---|---|---|---|---|---|
| 8 | 96.0 | 97.1 | 96.6 | 99.2 | 21.6 | 3.7 |
| 16 | 93.5 | 97.9 | 95.7 | 99.1 | 23.0 | 5.1 |

Note: AP is average precision and RMSE is root mean square error.

### 3.6 Performance of the three numbers of proposals

Table 7 shows the performance of the detector with three proposals. Among the three, the detector with 100 proposals had the lowest recall (92.2%) and AP (96.3%) but the fastest processing speed (7.7 fps). The one with 500 proposals had the lowest precision (90.5%), F1 score (92.8%), and processing speed (3.7 fps). The 300 proposals had the highest recall (97.1%) and F1 score (96.0%), comparable precision (95.0%), AP (99.0%), and RMSE (22.9 mm), and middle processing speed (5.1 fps), thus was chosen for detector development.

**Table 7 Detection performance of pullet feeding behaviors with three numbers of proposals**

| Numbers of proposals | Precision/ % | Recall/ % | F1 score/% | AP/ % | RMSE/ mm | Processing speed/fps |
|---|---|---|---|---|---|---|
| 100 | 96.5 | 92.2 | 94.3 | 96.3 | 19.5 | 7.7 |
| 300 | 95.0 | 97.1 | 96.0 | 99.0 | 22.9 | 5.1 |
| 500 | 90.5 | 95.4 | 92.8 | 97.4 | 27.3 | 3.7 |

Note: AP is average precision and RMSE is root mean square error.

### 3.7 Performance of two types of learning rates

Table 8 shows the performance of the two types of learning rates. The constant learning rate had lower precision (91.7%) than the dynamic one, while other performance indicators were similar between the two rates, thus the latter was selected for detector development.

**Table 8 Detection performance of pullet feeding behaviors with two types of learning strategies**

| Learning rate | Precision/ % | Recall/ % | F1 score/% | AP/ % | RMSE/ mm | Processing speed/fps |
|---|---|---|---|---|---|---|
| Constant | 91.7 | 97.4 | 94.5 | 98.8 | 24.0 | 5.0 |
| Dynamic | 94.0 | 97.5 | 95.7 | 99.1 | 21.8 | 5.1 |

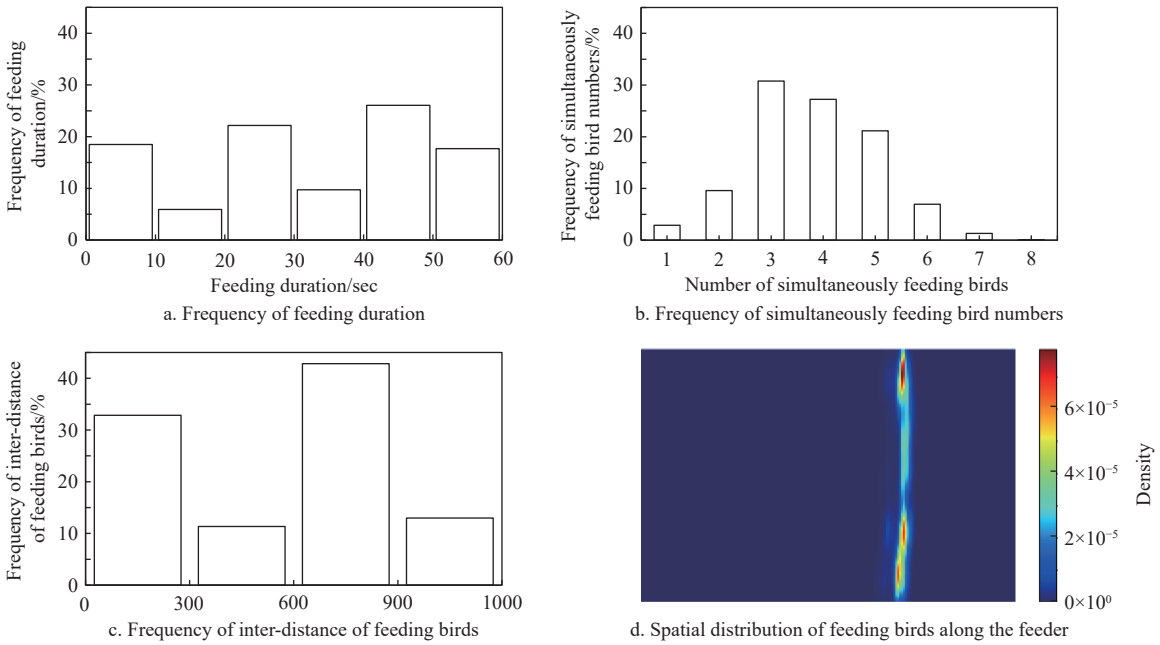
Note: AP is average precision and RMSE is root mean square error.

The following strategies were deployed to develop the final feeding behavior detector: feature extractor of Inception V2, COCO-

trained model, fixed_shape_resizer with the size of 600×600 pixels, kernel stride of 8, 300 proposals, and dynamic learning rate. The final detector had 95.7% recall, 94.2% AP, 94.9% F1 score, 23.5 mm RMSE, and 8.3 fps processing speed, which were comparable with most of the above-mentioned performance indicators.

### 3.8 Individual pullet feeding behaviors

Figure 7 shows the feeding behaviors of individual pullets of 50 d of age during 1 h and under white light. Total feeding time was 364.4 min, total feeder visits were 918 times, and average feeding duration was 0.4 min per visit. For 26.1% of the time (the highest among the 6 categories), the pullets ate for 40-50 s during each feeder visit; while for 5.9% of the time (the lowest among the 6 categories), the pullets were at feeder for 10-20 s. Three birds choosing to eat simultaneously accounted for the highest proportion (30.8%) of the time, while the cases of 8 simultaneous feeding birds took up the lowest proportion (0.08%). For 42.8% of the time (the most among the four categories), the pullets chose to stay 600-900 mm away from other feeding birds. The pullets preferred to eat at the two sides of the feeder trough rather than the middle.




a. Frequency of feeding duration

b. Frequency of simultaneously feeding bird numbers

c. Frequency of inter-distance of feeding birds

d. Spatial distribution of feeding birds along the feeder

Note: The inter-distance is the centroid distance between pairs of pullets. The white rectangle in Figure (d) represents the feeder trough, and non-unit density value in the same figure represents what is the probability for a feeding bird to present at a specific location.

Figure 7 Pullet feeding behaviors within one hour of 50 d of age and under white light

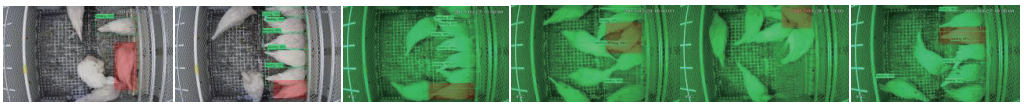
## 4 Discussion

### 4.1 Evaluation metrics

Appropriate evaluation metrics are critical for model development, and the metrics selections should be specified with different applications[38]. The true negative rate or specificity was not calculated in this case because pullets could move between compartments and the exact number of non-feeding birds in each frame was not clear. The generic evaluation metric, accuracy, was not considered either due to the lack of true negatives. Although AP and F1 score were comprehensive metrics to evaluate false positives and false negatives, they cannot be the only metrics for model evaluation. The two were high (mostly>95%) since they were averaged with the precision and recall; however, the corresponding precision or recall sometimes was not optimal. Multiple evaluation metrics should be considered for machine learning model development rather than single ones[39]. The RMSE can tell us how much deviation the model made to locate feeding birds and whether the detector is qualified for evaluating spatial feeding behaviors of poultry. The processing speed (2.1-8.1 fps in this case) provided evidence on how fast the model can process the images and whether it has potential for real-time applications.

### 4.2 Erroneous detection

The faster R-CNN detectors mostly performed well in detecting pullet feeding behaviors but still had some erroneous detections. As shown in Figure 8, pullet adhesion, overlapping, and occlusion around the feeder trough resulted in the detector misidentifying



Feeding bird #: 2 False negative #: 1 | Feeding bird #: 6 False negative #: 1 | Feeding bird #: 6 False negative #: 1 | Feeding bird #: 5 False negative #: 1 | Feeding bird #: 2 False negative #: 1 | Feeding bird #: 5 False negative #: 1

Feeding bird #: 2 False negative #: 1 | Feeding bird #: 6 False negative #: 1 | Feeding bird #: 6 False negative #: 1 | Feeding bird #: 2 False negative #: 1 | Feeding bird #: 4 False negative #: 2 | Feeding bird #: 4 False negative #: 1

Figure 8 False negative detection (False negatives are marked with solid red rectangles)

target objects[13,15]. Such an error became more obvious as more birds clustered at feeder but may be overcome by increasing model complexity, which may have more capacity to hold enough features of objects, thus being more robust. Kang and Chen[13] developed a deep learning network based on ResNet101 and evaluated it with less false negatives, compared with the lightweight network architecture, single shot detector (SSD). The detectors could wrongly recognize some features of non-feeding birds as those of the feeding ones, causing false positives. Previous investigations also reported that similar features between unconcern objects and concern objects could lead to false positive detection[19,25]. The false positive scenarios were summarized by stating that non-feeding birds falsely detected as feeding birds may have one of the following features: having a low confidence score of <99%, being 600 mm away from the feeder trough, and being parallel with the feeder trough (Figure 9). Those false positives may be ruled out with thresholds of the three features. It is worth noting that the detection performance could be further improved via tuning feature extractor and training hyperparameters, as shown in our results.

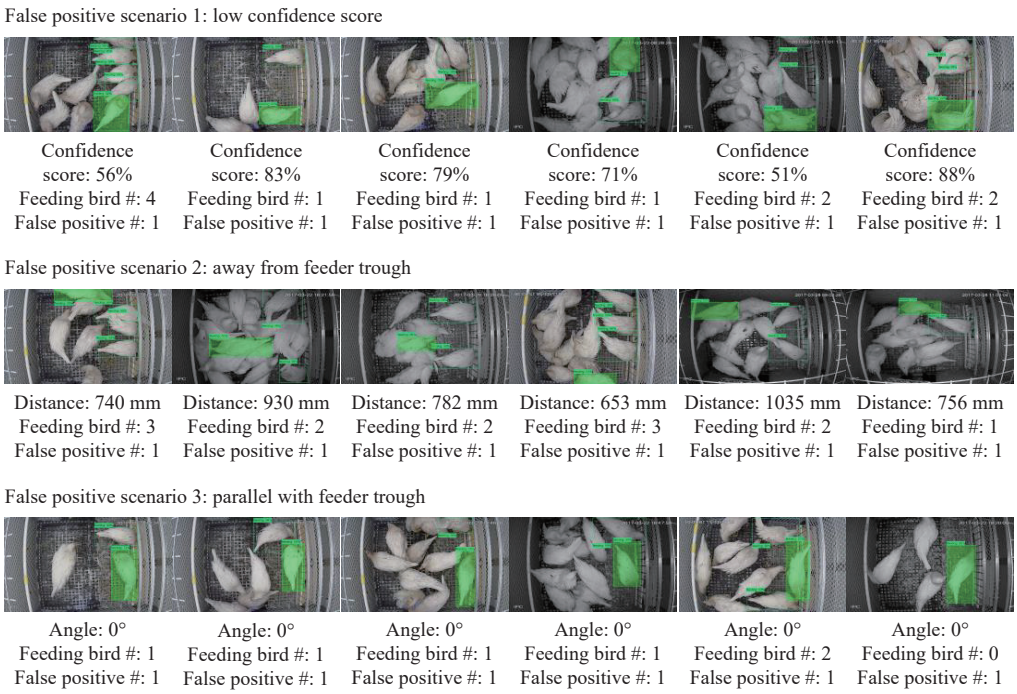



Figure 9 False positive detection under different scenarios (False positives are marked with solid green rectangles)

### 4.3 Feature extractors

Feature extractors can amplify aspects of input that are important to discriminate and classify target objects and suppress irrelevant variations[20], and those with higher complexity can retain more features. Among the four feature extractors, the most complex one, Inception ResNet V2, was more robust due to less false negative detection, but also more sensitive for objects of concern, leading to many false positives. Li et al.[16] compared the performance of SSD, faster R-CNN, and region-based fully convolutional network (R-FCN) for floor egg detection and found that the most complicated R-FCN over-predicted objects. The complexity of the feature extractors needs to be balanced with the dataset of single object of interest. With fewer layers stacked onto the model, the lightweight feature extractor, Inception V2, processed the images faster[12,15]. In sum, proper CNN architectures are critical for developing a robust and efficient detector as they can affect detection performance.

### 4.4 Pre-trained models

Transfer learning efficiency may depend on the similarity between the customized dataset and previous datasets[40]. Among the four datasets, COCO dataset is the most generic and largest one that contained chickens as well. The KITTI dataset included thousands of birds which also shared parts of similar features (e.g., feather, wing, beak, etc.) with pullets in this case. In contrast, the FGVC and AVA dataset mainly consisted of cars and humans, respectively. The texture, color, and edges for such objects were significantly different from those of chickens, and for this reason the pre-trained models from these two datasets had poorer performance. The processing speed of the detectors with AVA-trained model was even compromised because the detectors detected too many non-feeding birds within an image. To boost transfer learning efficiency, a survey is recommended to investigate the similarity between the customized dataset and the dataset of pre-trained models.

### 4.5 Image resizers

Mode of image resizers did not influence detection results in this case. Typically, if the shapes of the objects of interest are distorted after resizing, it may mismatch desired features and downgrade the detection performance[18]. In this case, feeding birds were large relative to the whole image because of low camera installation height, and the birds in the resized images may still maintain clear features for processing. Interestingly, the detector with larger image size performed relatively poorly, which was also reported by Li et al.[17] . The feeding birds may lose some critical features after the images were enlarged. The processing speed was even diminished with large image sizes because of more pixels being convoluted compared with smaller sizes.

### 4.6 Kernel strides

Smaller kernel strides may result in larger feature maps and retain more pixel information. The features of objects of interest could be relatively completed in such maps, which promoted the ability of the detector to differentiate the feeding birds from non-feeding birds in this case. That is why the stride of 8 had higher

precision than the stride of 16. Ren et al.[28] also reported that the stride of 10 produced more accurate results than the stride of 16. However, smaller kernel strides also indicated more convolution steps and led to compromised processing speed. The trade-off strategies need to be considered for detection accuracy and processing speed.

### 4.7 Numbers of proposals

Different numbers of proposals can be proposed by setting the thresholds of NMS and adjusting the designated number[28]. With less proposals, the target feeding birds may be filtered out, resulting in less true feeding birds being detected, which in turn improved processing speed. That is why the 100 proposals had lower recall but higher processing speed. With more proposals, some unnecessary proposals (e.g., non-feeding birds in this case) were retained and processed by the detector, leading to lower precision and processing speed for 500 proposals in this case. The middle number of proposals (300 proposals) seemed to balance different aspects of evaluation metrics and obtain decent performance. However, Wang et al.[21] compared the proposals of 30, 300, and 2000 for dairy recognition and found that the least number of proposals had the highest AP. The optimal proposal number may be specific for objects and verified for real applications.

### 4.8 Learning rate

Learning rate determines the update degree in each step. Optimizing the training with the same rate could lead to the oscillation away from minimum loss and further suboptimal models. Especially in this case, the model with a constant learning rate of 0.003 was suboptimal and caused many false positives. This could be improved by reducing the learning rates in advanced steps of training[36]. It should be noted that the learning rate strategies also depend on model architecture. Nasirahmadi et al.[16] compared the learning rates of 0.03, 0.003, and 0.0003 for faster R-CNN, R-FCN, and SSD, and reported that different models with these learning rates could result in various mAP for pig behavior detection.

### 4.9 Automated feeding behavior measurement

Individual feeding pullets could be continuously monitored with the trained detectors. The extracted behavior information showed that the pullets showed temporal and spatial preference on the feeding during the testing period. These behavior measures may provide valuable insights into farm management and facility design. For example, by evaluating the simultaneous feeding bird number and inter-distance of pairs of feeding birds, we could understand what the dimensions of the feeder trough and stocking density need to be to fit poultry preference[3]. In sum, the faster R-CNN feeding behavior detector is a useful tool to evaluate poultry feeding behaviors.

### 4.10 Major innovation and future application

To the best knowledge of the authors, this research is the first to examine feeding behaviors of poultry inside cage systems, which provide critical insights into precision poultry management. The current method relies on the camera installed atop cage systems. In future application, such a method can be used to monitor feeding behaviors of birds on the top layers of enriched colony systems.

## 5 Conclusions

This study developed faster R-CNN feeding behavior detectors by fine-tuning the feature extractors, pre-trained model, image resizer, kernel stride, number of proposals, and learning rate. Except for some cases (i.e., FGVC-trained and AVA-trained models), the detectors performed well in detecting poultry feeding behaviors. The precision, recall, F1 score, and AP were mostly over 90%. The RMSE ranged from 19.5 to 27.3 mm for most of the hyperparameters, indicating small location errors of detected feeding birds. The processing speeds were 2.1 to 8.1 fps and depended on architecture complexity of feature extractors and hyperparameter tuning. With the trade-off strategies for detection accuracy and processing speed, we finally deployed the following to develop the detector: feature extractor of Inception V2, COCO-trained model, fixed_shape_resizer with the size of 600×600 pixels, kernel stride of 8300 proposals, and dynamic learning rate. With the trained detector, the temporal and spatial feeding behaviors of individual pullets could be monitored and characterized. Overall, the faster R-CNN is a useful tool to monitor the feeding behaviors of individual poultry.

## Acknowledgements

This research was funded by China Scholar Council (Grant No. 201808410600). We acknowledge Prof. Zhengxiang Shi and Prof. Baoming Li of China Agricultural University for supporting the experiment of this study.

## [References]

[1] Wang S Y, Jiang G P, Pan C H, Santos T, Elhadidi Y, Jado A, et al. Light demand characteristics, production performance, and changes in the feeding patterns of broilers. Int J Agric & Biol Eng, 2024; 17(2): 68–73.

[2] Howie J A, Tolkamp B J, Avendano S, Kyriazakis I. The structure of feeding behavior in commercial broiler lines selected for different growth rates. Poultry Science, 2009; 88(6): 1143–1150.

[3] Collins L M, Sumpter D J T. The feeding dynamics of broiler chickens. Journal of the Royal Society Interface, 2006; 4(12): 65–72.

[4] Ferket P R, Gernat A G. Factors that affect feed intake of meat birds: A review. International Journal of Poultry Science, 2006; 5(10): 905–911.

[5] Urton G, Keyserlingk M A G V, Weary D M. Feeding behavior identifies dairy cows at risk for metritis. Journal of Dairy Science, 2005; 88(8): 2843–2849.

[6] Li L, Zhao Y, Oliveira J, Verhoijsen W, Liu K, Xin H. A UHF RFID system for studying individual feeding and nesting behaviors of group-housed laying hens. Transactions of the ASABE, 2017; 60(4): 1337–1347.

[7] Li G, Zhao Y, Hailey R, Zhang N, Liang Y, Purswell J L. An ultra-high frequency radio frequency identification system for studying individual feeding and drinking behaviors of group-housed broilers. Animal, 2019; 13(9): 2060–2069.

[8] Yang X, Dai H, Wu Z, Bist R B, Subedi S, Sun J, et al. An innovative segment anything model for precision poultry monitoring. Computers and Electronics in Agriculture, 2024; 222: 109045.

[9] Li G, Zhao Y, Chesser D, Lowe J W, Purswell J L. Image processing for analyzing broiler feeding and drinking behaviors. 2019 ASABE Annual International Meeting, 2019. DOI:10.13031/aim.201900165.

[10] Zou X G, Yin Z L, Li Y H, Gong F, Bai Y G, Zhao Z H, et al. Novel multiple object tracking method for yellow feather broilers in a flat breeding chamber based on improved YOLOv3 and deep SORT. Int J Agric & Biol Eng, 2023; 16(5): 44–55.

[11] Zhou M, Zhu J H, Cui Z H, Wang H Y, Sun X Q. Detection of abnormal chicken droppings based on improved Faster R-CNN. Int J Agric & Biol Eng, 2023; 16(1): 243–249.

[12] Gené-Mola J, Vilaplana V, Rosell-Polo J R, Morros J R, Ruiz-Hidalgo J, Gregorio E. Multi-modal deep learning for Fuji apple detection using RGB-D cameras and their radiometric capabilities. Computer and Electronic in Agriculture, 2019; 162: 689–698.

[13] Kang H, Chen C. Fast implementation of real-time fruit detection in apple orchards using deep learning. Computer and Electronic in Agriculture, 2020; 168: 105108.

[14] Kang H, Chen C. Fruit detection, segmentation and 3D visualisation of environments in apple orchards. Computer and Electronic in Agriculture, 2020; 171: 105302.

[15] Liang C, Xiong J, Zheng Z, Zhong Z, Li Z, Chen S, Yang Z. A visual detection method for nighttime litchi fruits and fruiting stems. Computer and Electronic in Agriculture, 2020; 169: 105192.

[16] Li G, Ji B, Li B, Shi Z, ZhaoY, Dou Y, et al. Assessment of layer pullet

drinking behaviors under selectable light colors using convolutional neural network. Computer and Electronic in Agriculture, 2020; 172: 105333.

[17] Li R, Wang R, Xie C, Liu L, Zhang J, Wang F, et al. A coarse-to-fine network for aphid recognition and detection in the field. Biosystem Engineering, 2019; 187: 39–52.

[18] Li G, Xu Y, Zhao Y, Du Q, Huang Y. Evaluating convolutional neural networks for cage-free floor egg detection. Sensors, 2020; 20(2): 332.

[19] Nasirahmadi A, Sturm B, Edwards S, Jeppsson K H, Olsson A C, Müller S, et al. Deep learning and machine vision approaches for posture detection of individual pigs. Sensors, 2019; 19(17): 3738.

[20] Tu S, Xue Y, Zheng C, Qi Y, Wan H, Mao L. Detection of passion fruits and maturity classification using Red-Green-Blue Depth images. Biosystem Engineering, 2018; 175: 156–167.

[21] Wang D, Tang J, Zhu W, Li H, Xin J, He D. Dairy goat detection based on Faster R-CNN from surveillance video. Computer and Electronic in Agriculture, 2018; 154: 443–449.

[22] Xu B, Wang W, Falzon G K, Wan P, Guo L, Chen G, et al. Automated cattle counting using Mask R-CNN in quadcopter vision system. Computer and Electronic in Agriculture, 2020; 171: 105300.

[23] Yang Q, Xiao D, Lin S. Feeding behavior recognition for group-housed pigs with the Faster R-CNN. Computer and Electronic in Agriculture, 2018; 155: 453–460.

[24] Zhang Y, Cai J, Xiao D, Li Z, Xiong B. Real-time sow behavior detection based on deep learning. Computer and Electronic in Agriculture, 2019; 163: 104884.

[25] Zhu X, Chen C, Zheng B, Yang X, Gan H, Zheng C, et al. Automatic recognition of lactating sow postures by refined two-stream RGB-D faster R-CNN. Biosystem Engineering, 2020; 189: 116–132.

[26] Li G, Li B, Shi Z, Zhao Y, Ma H. Design and evaluation of a lighting preference test system for laying hens. Computer and Electronic in Agriculture, 2018; 147: 118–125.

[27] Hy-Line International 2013. Growing management of commercial pullets. http://www.hyline.com/UserDocs/Pages/TB_PULLET_MGMT_ENG.pdf. Accessed on [2021-08-09] .

[28] Ren S, He K, Girshick R, Sun J. Faster R-CNN: Towards real-time object detection with region proposal networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2017; 39(6): 1137–1146.

[29] Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z. Rethinking the inception architecture for computer vision. IEEE Conference on Computer Vision and Pattern Recognition, 2016; pp.2818–2826. DOI:10.1109/CVPR.2016.308

[30] He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition, Proceedings of the Institute of Electrical and Electronics Engineers Conference on Computer Vision and Pattern Recognition, 2016; pp.770–778. DOI:10.1109/cvpr.2016.90

[31] Szegedy C, Ioffe S, Vanhoucke V, Alemi A A. Inception-v4, inception-resnet and the impact of residual connections on learning. 31st AAAI Conference on Artificial Intelligence, 2017; pp.4278–4284. DOI: 10.48550/arXiv.1602.07261

[32] Lotter W, Sorensen G, Cox D. A multi-scale CNN and curriculum learning strategy for mammogram classification. Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support, Springer, 2017; 10553: 169–177.

[33] Lin T Y, Maire M, Belongie S, Hays J, Perona P, Ramanan D, et al. Microsoft coco: Common objects in context. 13th European Conference on Computer Vision, Springer, 2014; 8693: 740–755.

[34] Geiger A, Lenz P, Stiller C, Urtasun R. Vision meets robotics: The kitti dataset. The International Journal of Robotics Research, 2013; 32(11): 1231–1237.

[35] Gu C, Sun C, Ross D A, Vondrick C, Pantofaru C, Li Y, et al. AVA: A video dataset of spatio-temporally localized atomic visual actions. IEEE Conference on Computer Vision and Pattern Recognition, 2018; pp.6047–6056. DOI:10.1109/CVPR.2018.00633

[36] Ruder S. An overview of gradient descent optimization algorithms, arXiv. preprint, 2016; p. 12. DOI:10.1109/CVPR.2018.00633

[37] Everingham M, Van G L, Williams C K, Winn J, Zisserman A. The pascal visual object classes (voc) challenge. International Journal of Computer Vision, 2010; 88(2): 303–338.

[38] Bylinskii Z, Judd T, Oliva A, Torralba A, Durand F. What do different evaluation metrics tell us about saliency models? IEEE Transactions on Pattern Analysis and Machine Intelligence, 2018; 41(3): 740–757.

[39] Dinga R, Penninx B W, Veltman D J, Schmaal L, Marquand A F. Beyond accuracy: Measures for assessing machine learning models, pitfalls and guidelines. BioRxiv, 2019; 743138. DOI:10.1101/743138

[40] Razavian A S, Azizpour H, Sullivan J, Carlsson S. CNN features off-the-shelf: an astounding baseline for recognition. IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, 2014; pp.512–519. DOI:10.1109/CVPRW.2014.131