
A MODEL GENERALIZATION STUDY IN LOCALIZING INDOOR COWS WITH COW LOCALIZATION (COLO) DATASET

A PREPRINT

 **Mautushi Das**
School of Animal Sciences
Virginia Tech
Blacksburg, VA 24061
mautushid@vt.edu

 **Gonzalo Ferreira**
School of Animal Sciences
Virginia Tech
Blacksburg, VA 24061
gonf@vt.edu

 **C. P. James Chen ***
School of Animal Sciences
Virginia Tech
Blacksburg, VA 24061
niche@vt.edu

January 22, 2025

ABSTRACT

1 Precision livestock farming (PLF) increasingly relies on advanced object localization techniques
2 to monitor livestock health and optimize resource management. This study investigates the gener-
3 alization capabilities of YOLOv8 and YOLOv9 models for cow detection in indoor free-stall barn
4 settings, focusing on varying training data characteristics such as view angles and lighting, and model
5 complexities. Leveraging the newly released public dataset, COws LOcalization (COLO) dataset,
6 we explore three key hypotheses: (1) Model generalization is equally influenced by changes in
7 lighting conditions and camera angles; (2) Higher model complexity guarantees better generalization
8 performance; (3) Fine-tuning with custom initial weights trained on relevant tasks always brings
9 advantages to detection tasks. Our findings reveal considerable challenges in detecting cows in
10 images taken from side views and underscore the importance of including diverse camera angles
11 in building a detection model. Furthermore, our results emphasize that higher model complexity
12 does not necessarily lead to better performance. The optimal model configuration heavily depends
13 on the specific task and dataset, highlighting the need for careful model selection tailored to the
14 particular application. Lastly, while fine-tuning with custom initial weights trained on relevant tasks
15 offers advantages to detection tasks, simpler models do not benefit similarly from this approach. It
16 is more efficient to train a simple model with pre-trained weights without relying on prior relevant
17 information, which can require intensive labor efforts. Future work should focus on adaptive methods

*Corresponding author: James Chen <niche@vt.edu>

18 and advanced data augmentation to improve generalization and robustness. This study provides
19 practical guidelines for PLF researchers on deploying computer vision models from existing studies,
20 highlights generalization issues, and contributes the COLO dataset containing 1254 images and 11818
21 cow instances for further research.

22 **Keywords** Object detection · Cows · Model generalization · Model selection

23 **1 Introduction**

24 **Object Localization and Its Applications**

25 Localizing livestock individuals from images or videos has become an essential task in precision livestock farming
26 (PLF) Fernandes et al. [2020]. Such techniques allow farm operators to manage animal well-being and health in
27 real-time, optimizing their resource management and improving sustainability Morrone et al. [2022], Hao et al. [2023].
28 Technically speaking, in the field of computer vision (CV), which is a subfield of artificial intelligence (AI) that focuses
29 on translating visual information into actionable insights, localization tasks can be further categorized into object
30 detection, object segmentation, and pose estimation. Object detection is the simplest form among these tasks, localizing
31 objects of interest by enclosing them within a rectangular bounding box defined by x and y coordinates, pixel width,
32 and pixel height Viola and Jones [2001]. Successful instances in this category include YOLO (You Only Look Once)
33 Redmon et al. [2016], Faster R-CNN (Region Convolutional Neural Networks) Girshick [2015], and SSD (Single Shot
34 MultiBox Detector) Liu et al. [2016]. These models have been adopted and applied by animal scientists for detection in
35 precision livestock farming. For example, a study Yu et al. [2022] leveraged the DRN-YOLO model Xu and Wu [2020]
36 to predict the eating behavior of dairy cows. This approach automates the assessment of feeding behavior, a critical
37 indicator of cow health and productivity, and has saved labor efforts in complex farm settings. Another notable work is
38 presented in Nasirahmadi et al. [2019], where the authors developed a posture detection system for pigs using deep
39 learning models such as Faster R-CNN, SSD, and R-FCN, coupled with 2D imaging. This system accurately identifies
40 standing and lying postures of pigs under commercial farm settings.

41 To achieve finer localization, object segmentation is employed to outline object contours pixel-wise, while pose
42 estimation is performed by orienting and marking the key points of the object Hariharan et al. [2015]. Some popular
43 object segmentation models include Mask R-CNN He et al. [2017], MS R-CNN Huang et al. [2019], and U-Net
44 Siddique et al. [2021]. This method of segmentation has also been applied in the field of PLF. In the study Noe et al.
45 [2022], the authors developed a method using Mask R-CNN He et al. [2017] to segment and outline cattle in feedlots.
46 Their technique enhances images and extracts key frames to accurately detect cattle, achieving superior precision
47 with a mean pixel accuracy of 0.92. This supports advanced, real-time monitoring of cattle in PLF. Another study
48 group Tu et al. [2021] developed the PigMS R-CNN framework Huang et al. [2019] to enhance the monitoring of
49 group-housed pigs. This framework employs a 101-layer residual network along with a feature pyramid network and
50 soft non-maximum suppression to effectively detect and segment pigs, thereby improving the accuracy of identifying
51 and locating individual pigs in complex environments.

52 **Model Generalization, Pre-Training, and Fine-Tuning**

53 Although implementing image-based systems in livestock production has become more common, current studies
54 primarily focus on accuracy in homogenous environments and rarely address the challenges of model generalization.
55 How a model can generalize to new environments is critical when farm operators deploy existing CV models in their
56 own settings. Good generalization performance ensures that the model can reproduce similar results as reported
57 in the original study, even in new environments with different conditions. Factors such as camera angles and the
58 presence of occlusions can impact generalization in the deployment environment. Deploying the same model in a new
59 environment does not necessarily guarantee the same performance as reported in the original study. Li et al. Li et al.
60 [2021] also pointed out that the lighting conditions on farms in real applications can be highly variable, leading to poor
61 generalization performance.

62 One explanation for poor generalization is the discrepancy between the pre-training process and the specific use case.
63 Most CV models are released with pre-trained weights, obtained from training on a large-scale dataset. For example,
64 the COCO dataset Lin et al. [2014] is a general-purpose dataset containing over 200,000 images and a wide range of
65 object categories, such as vehicles and household items. Directly deploying a model pre-trained on the COCO dataset
66 to detect cows in a farm setting may not ensure satisfactory performance, as the dataset does not contain enough cow
67 instances in different view angles or occlusions. To alleviate this discrepancy, fine-tuning is a common practice that
68 modifies the prediction head of the pre-trained model and updates the weights on a new dataset more relevant to the
69 specific use case. Most application studies have adopted this approach to improve model generalization on their specific
70 tasks Han et al. [2021], Guirguis et al. [2022], Gupta et al. [2023].

71 Nevertheless, fine-tuning is not guaranteed to be successful, as the outcome depends on both the quantity and quality of
72 the annotated dataset. For example, Zin et al. Zin et al. [2020] deployed an object detection model to recognize cow
73 ear tags in a dairy farm. Although the model achieved a high accuracy of 92.5% in recognizing the digits on the ear
74 tags, more than 10,000 images were required for fine-tuning. Assembling such a large dataset is labor-intensive and
75 requires specific training in annotating the images. The annotated dataset is rigorously organized in a specific format.
76 For example, the COCO annotation format Lin et al. [2014] stores image information, object class, and annotations of
77 the entire dataset in one nested JSON format. In contrast, the YOLO format Ultralytics [2023], another common format
78 for object localization, stores information of one image in one text file, with each line representing one object instance
79 in the image. Additionally, unlike the COCO format that stores bounding box coordinates in absolute pixel values,
80 the YOLO format stores the coordinates in relative values to the image size. These technical details are key to valid

81 annotations, which are usually facilitated by professional annotation tools such as Labelme Massachusetts Institute of
82 Technology [2023], CVAT OpenCV [2023], or Roboflow Roboflow [2023].

83 **Model Complexity and Performance**

84 Another factor affecting model generalization is model complexity. Generally, model complexity is quantified by the
85 number of learnable parameters in a model Hu et al. [2021]. A more complex model can often generalize better to
86 unseen data with high accuracy. However, this high complexity also comes at the cost of computational resources in the
87 form of memory or time Justus et al. [2018]. The computational cost may further limit how models can be deployed
88 in real-world applications, where real-time processing or edge computing is desired for fast or compact systems. For
89 instance, the VGG-16 model Simonyan and Zisserman [2014] has 138 million parameters and requires a video memory
90 of at least 8GB, while the ResNet-152 He et al. [2016] has around 60 million parameters with a recommended video
91 memory of 11GB. Recent models for object detection, such as YOLOv8 Ultralytics [Januray 2023] and YOLOv9 Wang
92 and Liao [2024], have been developed in different sizes, providing a flexible choice for researchers to balance between
93 generalization performance and computational cost. In YOLOv8, the spectrum of model complexity ranges from the
94 highly intricate YOLOv8x, containing 68.2 million parameters, to more streamlined variants like YOLOv8n with only
95 3.2 million parameters. The memory demand for the model architecture alone, without considering the intermediate
96 results during training or inference, is larger by a factor of 21 for YOLOv8x (136.9 megabytes) compared to YOLOv8n
97 (6.5 megabytes). Therefore, the trade-off between model complexity and computational cost is a critical factor to
98 consider when deploying CV models in real-world scenarios.

99 **YOLO Models**

100 Before YOLO, object detection methods typically involved either using “sliding windows with classifier” or “region
101 proposals with classifier.” The sliding windows method required running the classifier hundreds or thousands of times
102 per image. On the other hand, advanced region proposal-based approaches divided the task into two steps: first,
103 identifying potential object regions (i.e., region proposals) and then applying a classifier to these regions. In contrast,
104 YOLO models are capable of performing object detection in a single pass through the network, which is why the
105 acronym YOLO stands for “You Only Look Once.”

106 YOLOv8 Ultralytics [Januray 2023], building on the YOLOv5 Jocher [2020] architecture, incorporates the C2F
107 module (cross-stage partial bottleneck with two convolutions), a refinement of the CSPLayer of YOLOv5 featuring
108 two convolutional operations. It employs SiLU activation over traditional ReLU and Sigmoid Elfwing et al. [2018] for

109 smoother gradient flow, enhancing CNN performance. The module divides input from a convolutional layer, processes
110 one half through bottleneck layers (offering two types: with and without shortcuts similar to ResNet Targ et al. [2016]),
111 then merges it back for further convolution. This design, along with a spatial pyramid pooling fast (SPPF) layer in its
112 backbone, supports efficient feature pooling and multi-scale detection by using three distinct heads, thereby optimizing
113 object detection across varying sizes. Furthermore, YOLOv8 innovates with an anchor-free approach, directly predicting
114 bounding boxes and confidence scores, thus simplifying the network and reducing computational overhead Law and
115 Deng [2018], Duan et al. [2019], Tian et al. [2019].

116 Deep learning models, including the YOLO family, encounter an information bottleneck issue Tishby and Zaslavsky
117 [2015], Tishby et al. [2000], where the retention of input information diminishes as data is compressed into features.
118 This loss is exacerbated in deeper network layers, often leading to reduced model efficacy. One approach to mitigate
119 this involves expanding the model's width, i.e., increasing the number of parameters, which allows for broader feature
120 mapping and potentially recaptures lost information. However, simply increasing model size can lead to unreliable data
121 outputs and does not proportionally enhance model performance.

122 YOLOv9 addresses these challenges through innovations like Programmable Gradient Information (PGI) and the
123 Generalized Efficient Layer Aggregation Network (GELAN) Wang and Liao [2024]. PGI optimizes gradient generation
124 to minimize deep layer information loss, featuring a main branch for inference and auxiliary branches for enhanced
125 training. GELAN, by integrating and pooling convolutional layers, ensures robust feature retention. This adaptive
126 system notably boosts inference speed by 20% Wang and Liao [2024] on the COCO dataset Lin et al. [2014], while its
127 multi-level auxiliary information facilitates the detection of objects across varying sizes, making YOLOv9 particularly
128 effective in identifying smaller objects compared to its predecessors.

129 **Public Datasets**

130 A public dataset helps the community to develop methodology based on the same baseline. One famous example in
131 computer vision is the ImageNet dataset Deng et al. [2009], which serves as a benchmark for image classification.
132 AlexNet Krizhevsky et al. [2017], the winner of the ImageNet Large Scale Visual Recognition Challenge in 2012,
133 demonstrated its outstanding capability to classify images in the ImageNet dataset using Rectified Linear Units (ReLU)
134 as the activation function, rather than the traditional sigmoid function. The success of AlexNet accelerated the
135 development of CV models in the following years, such as VGG Karen [2014], GoogLeNet Szegedy et al. [2015],
136 ResNet Targ et al. [2016], and DenseNet Huang et al. [2017]. However, similar to the challenges that pre-trained models

137 face in specific use cases, a generic public dataset, such as ImageNet Deng et al. [2009] and COCO Lin et al. [2014],
138 may not be sufficient for PLF applications.

139 There have been efforts to create public datasets for livestock scenarios. For example, the CattleEyeView dataset was
140 collected to support applications like cattle pose estimation and behavior analysis, providing extensive annotations
141 across 30,703 frames from top-down video sequences of cows Ong et al. [2023]. Another study T. Psota et al. [2020]
142 leverages a public dataset for pigs comprising 3600 images from 12 videos of group-housed pigs. The dataset is
143 particularly designed for applications such as pig tracking. Additionally, the "OpenCows2020" dataset, developed
144 by researchers from the University of Bristol, is a public dataset specifically designed for advancing non-intrusive
145 monitoring of cattle. It supports precision farming applications such as automated productivity assessment, health and
146 welfare monitoring, and veterinary research, including behavioral analysis and disease outbreak tracing. The dataset
147 consists of 11,779 images with 13,026 labeled objects, mainly focusing on cattle Ninja [2024].

148 **Study Objectives**

149 This study aims to use YOLO-family models to explore model generalization across varying environmental settings
150 and model complexities within the context of indoor cow localization. Object detection, being the simplest form
151 of localization, serves as an ideal baseline for extending to more complex tasks such as segmentation and posture
152 estimation. Starting with object detection allows for a clear and foundational understanding of model behavior and
153 performance, which can then inform and enhance the approach to more complex tasks. Consequently, this study
154 examines three hypotheses:

- 155 • **Model generalization is equally influenced by changes in lighting conditions and camera angles.** Should
156 camera angles prove more impactful than lighting conditions, it would be advisable to prioritize camera
157 placement when deploying CV models in new environments.
- 158 • **Higher model complexity guarantees better generalization performance.** If a highly complex model does
159 not ensure superior performance, future studies might consider adopting less computationally demanding
160 models that still enhance performance.
- 161 • **The advantages of using fine-tuned models as initial training weights are persistent over pre-trained
162 models.** If the advantages diminish as the training sample size increases in a similar cow localization task but
163 different environments, the fine-tuning efforts may be deemed unnecessary when the deployment environment
164 varies over multiple locations on a farm.

165 To facilitate these investigations, a public dataset named COws LOcalization (COLO) COL [2023] will be developed and
166 made available to the community. The findings of this study are expected to provide practical guidelines for Precision
167 Livestock Farming (PLF) researchers on deploying CV models, considering available resources and anticipated
168 performance.

169 **2 Materials and Methods**

170 **Cow Husbandry**

171 All procedures involving cow handling and image capturing were conducted in accordance with ethical guidelines and
172 approved by the Virginia Tech Institutional Animal Care and Use Committee (IACUC #22-146). The cows studied were
173 part of the dairy herd at the Virginia Tech Dairy Complex in Blacksburg, Virginia, USA, which comprises approximately
174 80% Holstein and 20% Jersey cows. For the 'External' setting, the study included 100% Holstein cows. The milking
175 cows were housed in pens within a free-stall barn, featuring two rows of sand-bedded stalls, headlocks at the feed bunk,
176 and two water troughs per pen. The stocking density was maintained at 100% (i.e., one cow per stall). Heat stress was
177 managed using automatic 48-inch diameter fans positioned over the stalls and feeding alleys. Cows were milked twice
178 daily at 1:00 am and 12:00 pm in a double-twelve parallel milking parlor. They were fed ad libitum (with less than
179 5% refusals) once daily at 8:00 am with a total mixed ration (TMR) consisting of approximately 42% corn silage, 8%
180 grass hay, and 50% concentrate on a dry matter basis. Manure from the stalls was removed at each milking session by
181 personnel driving the cows to milking. Manure from the walking alleys within the pen was cleared two or three times
182 daily using an automatic flushing system with recycled water. Fresh or recycled sand was added on a weekly basis.

183 **Image Dataset**

184 The images in this study were collected using the Amazon Ring camera model Spotlight Cam Battery Pro (Ring Inc.),
185 which offers a real-time video feed of dairy cows. Three cameras were installed in the barn: two at a height of 3.25
186 meters (10.66 feet) above the ground covering an area of 33.04 square meters (355.67 square feet). One camera provided
187 a top view while the other was angled approximately 40 degrees from the horizontal to offer a side view of the cows.
188 These are hereafter referred to as *the Top-View camera* and *the Side-View camera*, respectively. A third camera, termed
189 *the external camera*, was set at a lower height of 2.74 meters (9.00 feet) and covered a larger area of 77.63 square
190 meters (835.56 square feet). Positioned 10 degrees downward from the horizontal, it captured a challenging perspective
191 prone to occlusions among cows.

192 Images were captured using an Ring Application Programming Interface (API) [Greif, 2024], configured to record a
193 ten-second video clip every 30 minutes continuously for 14 days. Since the image quality relies on the camera's internet
194 connection, which was occasionally unstable, some images were found to be tearing or unrecognizable. Hence, the
195 resulting dataset was manually curated for consistent quality, comprising 504 images from *the Top-View camera*, 500
196 from *the Side-View camera*, and 250 from *the external camera*. These images were further categorized based on the
197 lighting conditions: for *the Top-View camera*, 296 images were captured during daylight, 118 in the evening under
198 artificial lighting, and 90 as near-infrared images without artificial light. From *the Side-View camera*, 113 images were
199 taken in the evening, and 97 as near-infrared images. All images from *the external camera* were captured during the
200 day. Image examples are shown in **Figure 1a**.

201 The image annotations were conducted using an online platform, Roboflow Roboflow [2023], to define cow positions in
202 the images. The bounding boxes were manually drawn to enclose the cow contours, providing the coordinates of the
203 top-left corners and the width and height of the boxes. If cows were partially occluded, the invisible parts were inferred
204 based on the adjacent visible parts. If the cow position was too far from the camera, making important body features
205 such as the head, tail, and legs unrecognizable, the cow was excluded from the dataset. The final annotations were
206 saved in the YOLO format Ultralytics [2023], where annotations were stored in a text file with one row per cow in the
207 image, each row containing the cow's class, center coordinates, width, and height of the bounding box. The graphical
208 representation of the annotated images is shown in **Figure 1b**.

209 **Model Training**

210 The model training was implemented using the Python library Ultralytics Ultralytics. The model hyperparameters
211 were set to the default values in the library. The training epochs were set to 100, and the batch size was set to 16. The
212 implemented data augmentation included randomly changing the image color hue, saturation, and exposure to improve
213 the model's generalization to different lighting conditions. Geometry augmentation was also applied by randomly
214 flipping the images horizontally, copying and pasting to mix up object instances across multiple images to increase data
215 diversity, and randomly scaling the images to simulate different distances between the camera and the cows. The details
216 of the hyperparameters are shown in **Table 3**. The training was conducted on an NVIDIA A100 GPU (NVIDIA, USA)
217 with 80GB video memory provided by Advanced Research Computing at Virginia Tech.

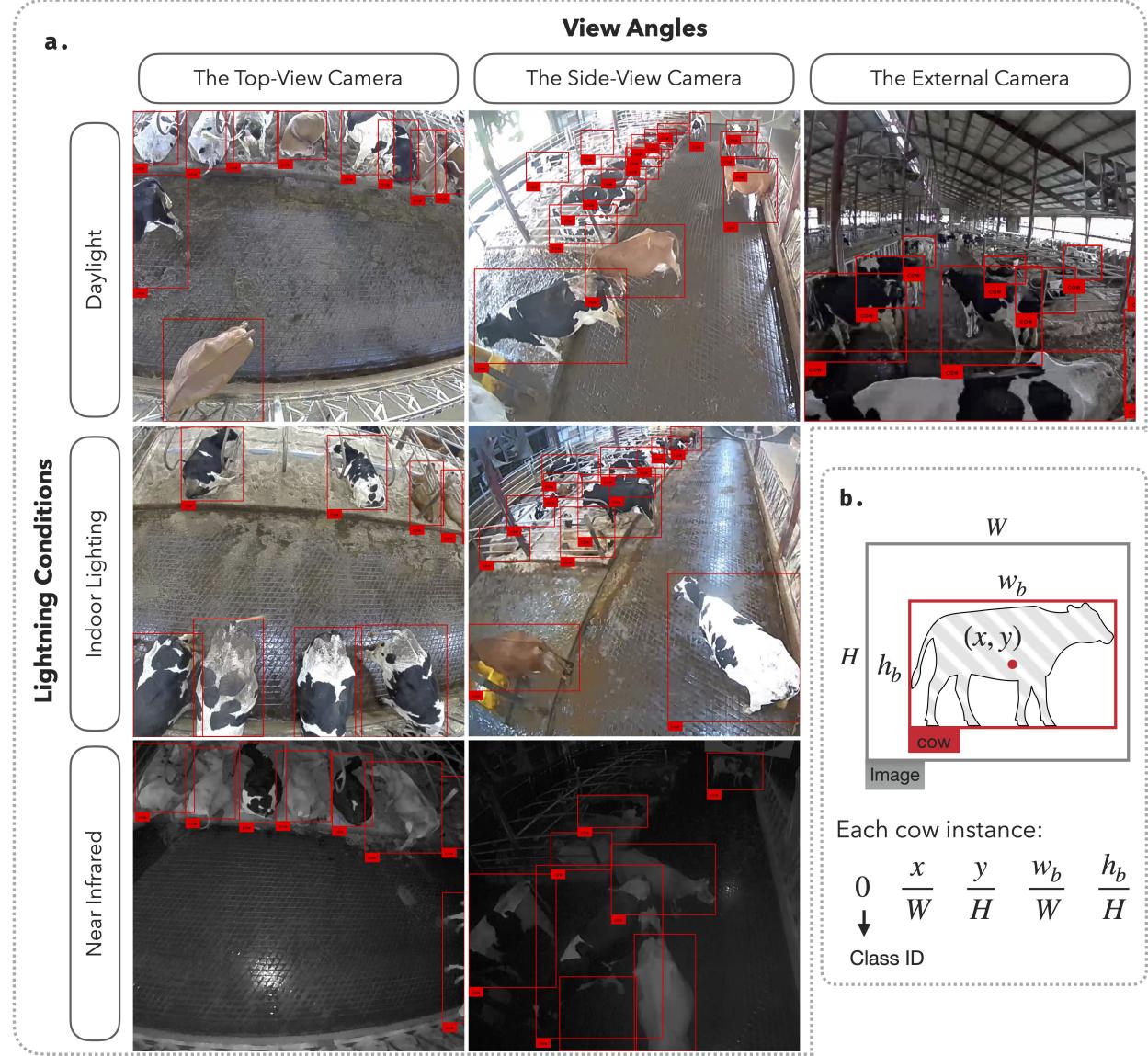


Figure 1: Overview of the COLO dataset. 1a. Seven instance images from the dataset with red bounding boxes labeling the location of cows. The columns show three different view angles: top-view, side-view, and external. The rows show three different lighting conditions: daylight, indoor, and near-infrared. 1b. An example of the annotated image in YOLO format. W , H , w_b , and h_b represent the width, height, width of the bounding box, and height of the bounding box, respectively. x and y represent the center coordinates of the bounding box.

218 Model Evaluation

219 The examined YOLO models are object detection models that return positions of detected objects (i.e., cows in this
 220 study) for the evaluated images. The detections are represented by a list of bounding boxes. Regardless of specific
 221 procedures among YOLO variants for computational efficiency, such as YOLOv8, which integrates objectness scores
 222 and conditional class probabilities into a single confidence score, each detection generally consists of $4 + c$ elements:

223 the xy-coordinates, width, and height of the bounding box, and the c confidence scores indicating the probability of
224 the object belonging to each of the c classes. The class with the highest confidence score is considered the predicted
225 class of the object. To evaluate the model performance, two aspects are considered: the localization accuracy and
226 the classification accuracy. The localization accuracy is measured by the Intersection over Union (IoU) between the
227 predicted bounding box and the ground truth bounding box. On the other hand, the classification accuracy is measured
228 by the precision and recall given the confidence threshold. If the confidence score of a detection is higher than the
229 threshold, the detection is considered a positive detection. Otherwise, the detection is neglected. Combining the
230 localization and classification accuracy, the mean Average Precision (mAP) averages the area under the precision-recall
231 curve across all the classes. The curve is generated by varying the confidence threshold from 0 to 1 given an IoU
232 threshold. In this study, four metrics were used in the evaluation: the precision and recall at the confidence threshold of
233 0.25 and IoU threshold of 0.5, the mAP at the IoU threshold of 0.5 (noted as $mAP@0.5$), and the averaged mAP at
234 varying IoU thresholds ranging from 0.5 to 0.95 (noted as $mAP@0.5:0.95$).

235 **Study 1: Benchmarking Model Generalization Across Different Environmental Conditions**

236 To compare the performance drop between different view angles and lighting conditions, we designed a cross-validation
237 strategy where models were trained on one dataset configuration and tested on another. There are five training
238 configurations in this study (**Figure 2**):

239 • **Baseline:** The model was trained and evaluated on the dataset characterized for all conditions, including
240 top-view, side-view, daylight, evening, and near-infrared images. The images did not overlap between the
241 training and evaluation sets.

242 • **Top2Side:** The model was trained on the top-view images and evaluated on the side-view images.

243 • **Side2Top:** The model was trained on the side-view images and evaluated on the top-view images.

244 • **Day2Night:** The model was trained on the daylight images and evaluated on the evening images, including
245 both artificial lighting and near-infrared images.

246 • **External:** The model was trained on images collected by the Top-View and Side-View cameras and evaluated
247 on the external camera images.

248 To study how the training sample size affects model performance in each configuration, the testing set in the cross-
249 validation was fixed to the same 100 images. Then, the training set size was iteratively altered from 16 to 512 images,

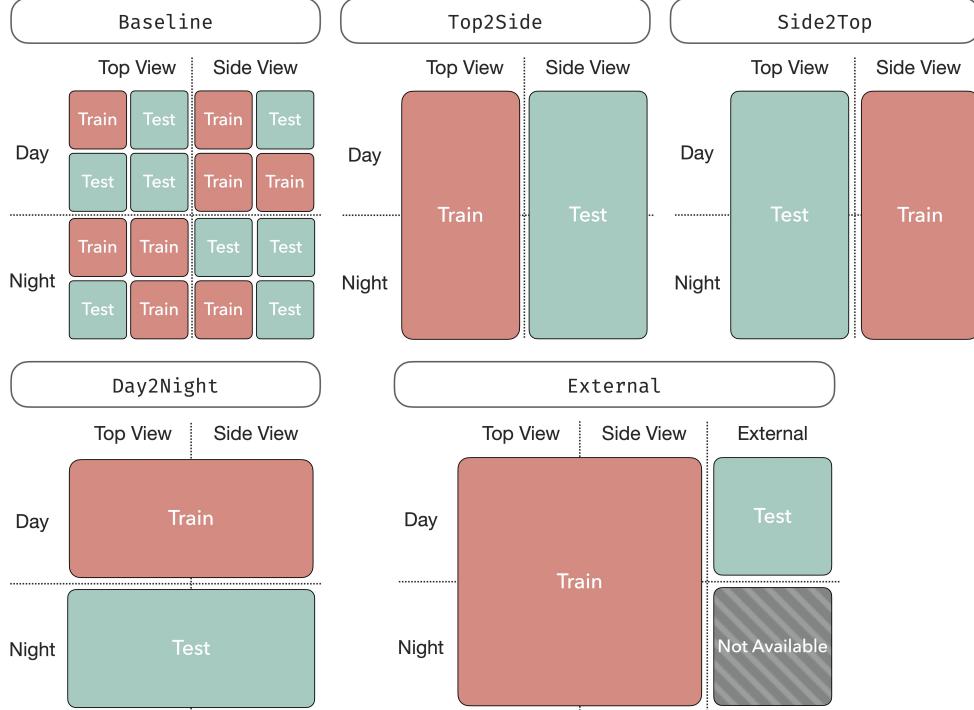


Figure 2: Cross-validation configurations. The training and testing sets were split into five different configurations: Baseline, Top2Side, Side2Top, Day2Night, and External.

250 with the sample size doubled at each step. Each training sample size was repeated 50 times with different random seeds
 251 to ensure the robustness of the results. The YOLOv9e, which is the most capable model in the YOLO family to date
 252 according to its performance on the COCO dataset, was used as the base model for this study.

253 **Study 2: The Correlation Between Model Complexity and Performance on the Tasks of Localizing Cows**

254 To investigate whether model performance increases with model complexity, five YOLO-family models were examined
 255 in this study. Three of the models were selected from the YOLOv8 family: YOLOv8n, YOLOv8m, and YOLOv8x. All
 256 YOLOv8 models share a similar architecture, differing in their depth multiplier, width multiplier, and ratio factor, which
 257 collectively determine their parameter counts of 3.2 million (m), 25.9m, and 68.2m, respectively. The depth multiplier
 258 determines how many convolutional layers are repeated in a C2F module, the novelty of YOLOv8. The width multiplier
 259 and ratio factor collectively specify the channel numbers in the convolutional operations. Correspondingly, YOLOv8n,
 260 YOLOv8m, and YOLOv8x are defined by depth multipliers of 0.33, 0.67, and 1.0, respectively. The width multipliers
 261 are 0.25, 0.75, and 1.25, while the ratio factors are 2.0, 1.5, and 1.0 v8y [2023]. These variations enable the models to
 262 achieve different balances between computational efficiency and accuracy.

263 The remaining two models were YOLOv9c and YOLOv9e, the latest models in the YOLO family, with parameter
264 counts of 25.6M and 58.2M, respectively. Unlike YOLOv8 models, these models have slightly different backbone
265 architectures. Although the majority of the components between YOLOv9c and YOLOv9e are the same, they primarily
266 differ in their layer counts, module complexities, and depth configurations. YOLOv9c has 618 layers and uses simpler
267 modules, resulting in a more efficient model with lower computational demands. Conversely, YOLOv9e has 1225
268 layers and employs more advanced modules v9y [2023].

269 All models were trained on 500 images in the five cross-validation configurations: Baseline, Top2Side, Side2Top,
270 Day2Night, and External (**Figure 2**). In addition to model performance, computing speed was also evaluated. The
271 training speed was recorded in seconds per 100 epochs on NVIDIA A100 GPUs (NVIDIA, USA), and the inference
272 time was recorded as frames per second (FPS) on both the CPU and GPU (Apple M1 Max chip, Apple Inc., USA).
273 The relationship between model complexity and time consumption was analyzed to provide insights into the trade-off
274 between model performance and computational cost.

275 **Study 3: Assessing the Advantages of Using Fine-Tuned Model Over the Pre-Trained Model as Initial Model**

276 **Weights**

277 Most models are released with pre-trained weights obtained from large datasets containing millions of object instances
278 (e.g., COCO Lin et al. [2014] and ImageNet Deng et al. [2009]). The pre-trained models have a general capability in
279 recognizing common objects such as vehicles, animals, and household items. When the model is required to recognize
280 specific objects (i.e., cows in this study), a model trained on a smaller but specific dataset is expected to have better
281 performance. However, such advantages may not necessarily persist as the training sample size increases. Having an
282 equally large number of samples for both the pre-trained and fine-tuned models could diminish the performance gap
283 between the two models. To investigate this hypothesis, this study evaluated the performance of fine-tuned models with
284 two different initial weights. The first initial weight was the default weight from the pre-trained model on the COCO
285 dataset, while the second initial weight was the weight from the fine-tuned model on the opposite view angle. The
286 cross-validation settings are described in **Table 1**.

287 The backbones of all models (i.e., YOLOv8n, YOLOv8m, YOLOv8x, YOLOv9c, and YOLOv9e) were fine-tuned
288 across different training sample sizes: 16, 32, 64, 128, 256, and 500. The goal was to determine whether the advantage
289 of using the fine-tuned weights persists under the interaction between model complexity and different fine-tuning
290 samples. The performance of the models was evaluated using mAP@0.5:0.95.

Table 1: Finetuning configurations with different initial weights

Finetuning and Prediction Task	Initial Weights
Top-View Camera	COCO (pre-trained) Side-View Camera (fine-tuned)
Side-View Camera	COCO (pre-trained) Top-View Camera (fine-tuned)
External Camera	COCO (pre-trained) Top-View and Side-View Cameras (fine-tuned)

291 3 Results and Discussion

292 Public Dataset: COLO

293 The COLO dataset contains 1254 images and 11818 cow instances captured from an indoor farm environment.
 294 The dataset is organized in YOLO and COCO formats and published on the online platforms GitHub
 295 (<https://github.com/Niche-Squad/COLO/>) and Huggingface (<https://huggingface.co/datasets/Niche-Squad/COLO>). The
 296 dataset consists of eight configurations (**Table 2**): *0_all*, *1_top*, *2_side*, *3_external*, *a1_t2s*, *a2_s2t*, *b_light*, and
 297 *c_external*. The *0_all* configuration serves as the baseline for this study, featuring non-overlapping training and
 298 testing images collected from both the Top-View Camera and Side-View Camera. The *1_top*, *2_side*, and *3_external*
 299 configurations contain images from their respective cameras. The *a1_t2s*, *a2_s2t*, and *b_light* configurations include
 300 training/testing splits for the Top2Side, Side2Top, and Day2Night scenarios, respectively. The *c_external* configuration
 301 features training images from the Top-View and Side-View Cameras, with testing images from the External Camera.
 302 The dataset hosted on GitHub is available as a compressed zip file for public access. In contrast, the dataset on
 303 Huggingface requires the Python package "datasets" Lhoest et al. [2021] to download. The Huggingface version offers
 304 additional functionality to resize the images and annotations to specific resolutions, providing greater flexibility for
 305 various applications.

Table 2: Description of the COLO dataset configurations.

Configuration	Training Samples	Testing Samples
<i>0_all</i>	Top-View + Side-View	Top-View + Side-View
<i>1_top</i>	Top-View	Top-View
<i>2_side</i>	Side-View	Side-View
<i>3_external</i>	External	External
<i>a1_t2s</i>	Top-View	Side-View
<i>a2_s2t</i>	Side-View	Top-View
<i>b_light</i>	Day	Night
<i>c_external</i>	Top-View + Side-View	External

306 Evaluation Metrics

307 To assess the performance of the YOLO models, we used four key metrics: mAP@0.5:0.95, mAP@0.5, precision, and
308 recall. These metrics provide a comprehensive understanding of how well the models detect and localize cows in the
309 images from the COLO dataset. A pair-wise comparison of these metrics is presented in **Figure 3** to illustrate their
310 interrelationships.

311 The mAP@0.5:0.95 metric is the most stringent, requiring the model to achieve both high positioning accuracy (i.e.,
312 high IoU) and high precision across IoU thresholds from 0.5 to 0.95. Because it is less likely to be influenced by
313 high-confidence predictions alone, it serves as a reliable indicator of overall model performance. Achieving an accuracy
314 greater than 0.90 on this metric is generally unrealistic; typically, a value of 0.7 is considered good and is sufficient to
315 yield precision and recall of around 0.9.

316 In contrast, mAP@0.5 is more lenient, requiring high confidence but only moderate IoU. It measures the average
317 precision at an IoU threshold of 0.5. For applications where counting cows is more important than precise positioning,
318 an mAP@0.5 value of 0.9 is sufficient. For example, our results showed that the YOLOv8n model, trained on 32
319 samples, achieved an mAP@0.5 of 0.9, making it suitable for such applications.

320 Precision and recall metrics focus on the accuracy and completeness of the detections. Precision is the ratio of true
321 positive detections to the total number of positive detections (true positives + false positives), measuring how accurate
322 the positive predictions are. Recall is the ratio of true positive detections to the total number of actual positives (true
323 positives + false negatives), measuring the model's ability to detect all relevant objects. Generally, higher precision is
324 associated with higher recall. However, in some configurations, such as Side2Top and External with smaller sample
325 sizes, models exhibited high recall but low precision. This indicates a tendency to misclassify non-cow objects as cows
326 more frequently than missing actual cows, suggesting a tendency to overestimate rather than underestimate the number
327 of cows in the images.

328 Our observations emphasize that for applications where counting cows is more critical than precise positioning,
329 achieving a high mAP@0.5 is adequate, while the stringent mAP@0.5:0.95 metric serves as a comprehensive indicator
330 of overall model performance. These metrics provide insights into both the localization and classification capabilities of
331 the models, helping to identify strengths and weaknesses under different environmental conditions and camera angles.

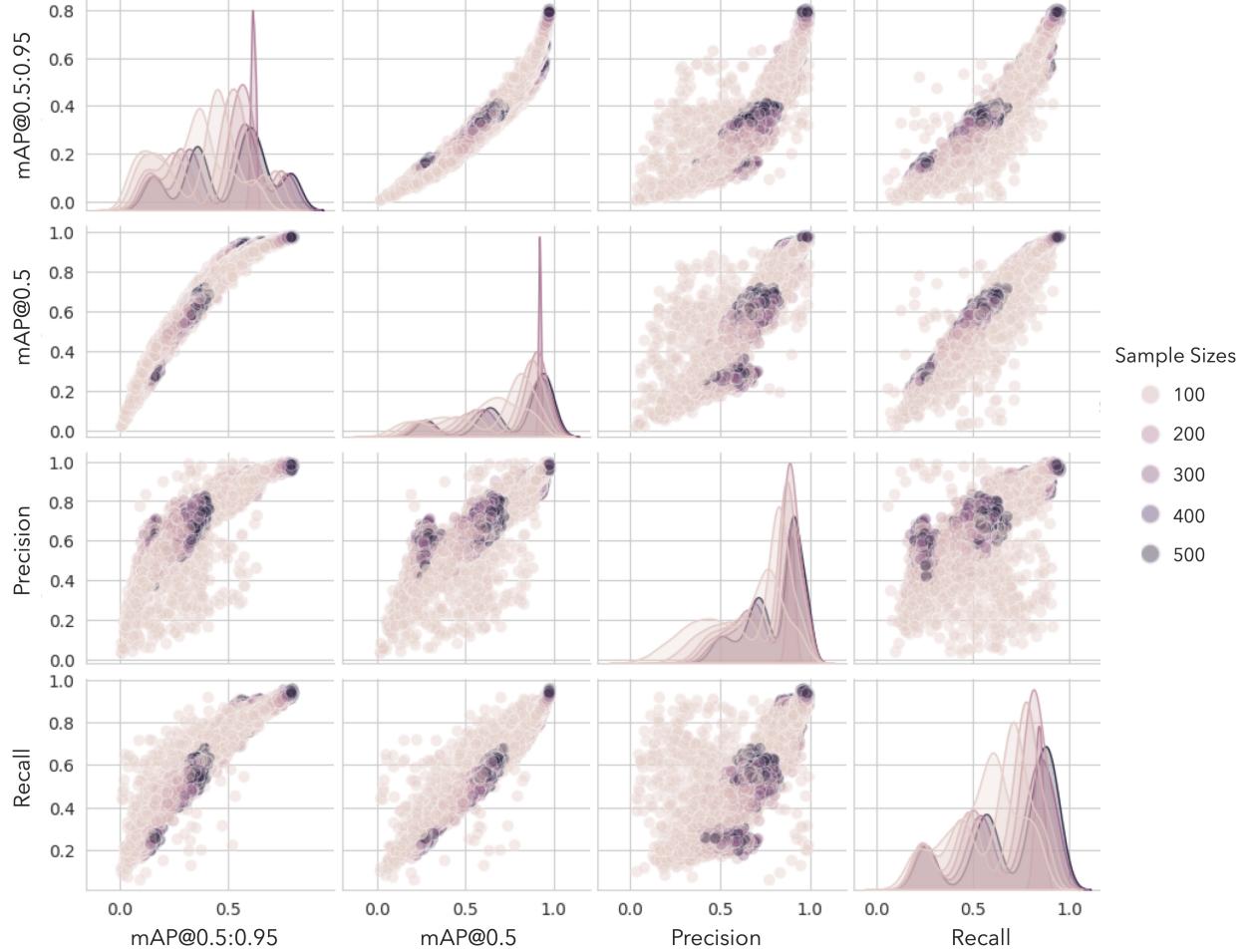


Figure 3: Pairwise scatter plots of the evaluation metrics: mAP@0.5:0.95, mAP@0.5, precision, and recall. Each point represents a different model configuration, with the color indicating the training sample size.

332 Study 1: The Changes in Camera View Angles Dramatically Affect Model Performance

333 The baseline training configuration showed good generalization capability, with over 90% of the predictions correctly
 334 positioning cows at the 50% IoU criterion (mAP@0.5). The generalization performance can be dissected into changes
 335 in view angles (i.e., Top2Side and Side2Top) and lighting conditions (i.e., Day2Night). Changes in lighting conditions
 336 did not dramatically affect model performance across all four metrics. However, changing camera views resulted
 337 in a performance drop of approximately 30% and 60% in mAP@0.5 for the Side2Top and Top2Side configurations,
 338 respectively. Across all metrics and training sample sizes, the Top2Side configuration consistently showed the worst
 339 performance.

340 From the perspective of precision and recall, changing the camera view from Top2Side resulted in the model missing
 341 more than 7 out of every 10 cows, with only 50% of the detections being correct. For the 'External' configuration,

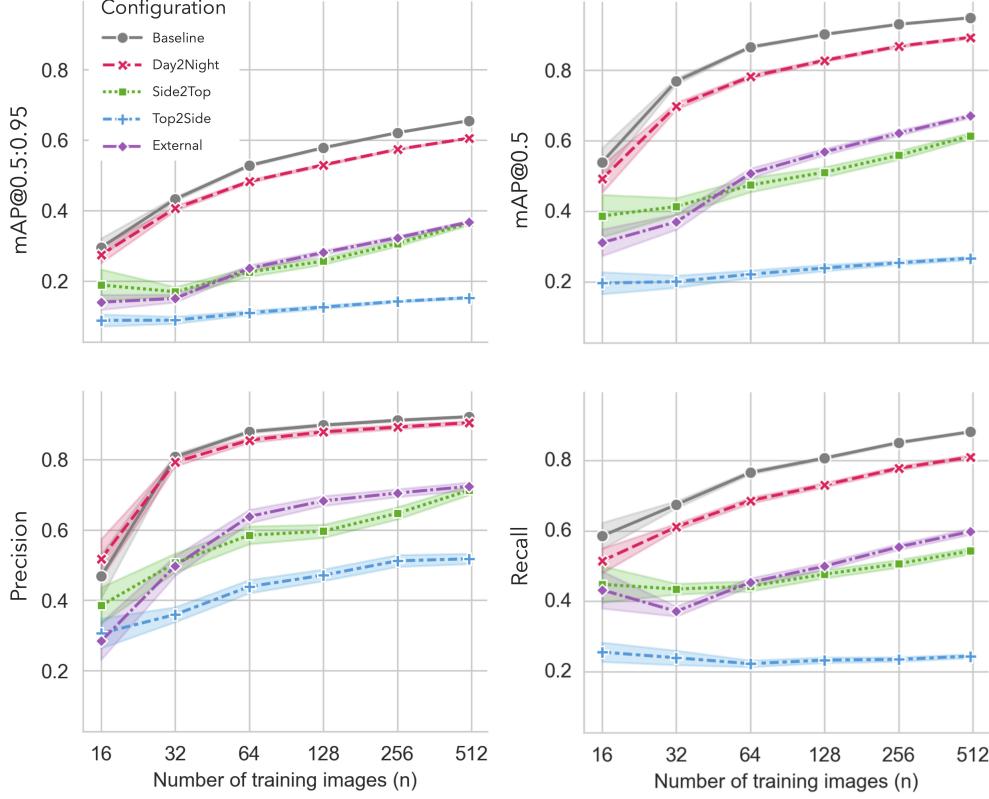


Figure 4: The generalization performance of YOLOv9e across various data configurations and training sample sizes. Sample sizes are depicted on the horizontal axis using a logarithmic scale with a base of 2, and the data configurations are represented by different colors and marker shapes. The upper left and right plots display the metrics mAP@0.5:0.95 and mAP@0.5, respectively, for different training samples across diverse data configurations. The lower left and right plots depict precision and recall values, also for varying training samples and configurations.

- 342 our model identified 6 out of every 10 cows, which is not ideal but also not the worst performance observed. Notably,
 343 performance in the Day2Night configuration was close to the baseline in terms of precision, which only considers
 344 predictions with high confidence compared to mAP@0.5. Hence, by excluding low-confidence predictions, changes in
 345 lighting conditions did not affect model performance. Regardless of the configuration and evaluation metrics, model
 346 performance always increased as the training sample sizes increased.
- 347 This study provides a comparative analysis of the behavior of the model in various data configurations. It is clear
 348 that the ‘Day2Night’ configuration shows much better performance relative to the heterogeneous viewpoint-oriented
 349 configurations ‘Top2Side’ and ‘Side2Top’.
- 350 Despite the various challenges in adapting models from day to night conditions, the ‘Day2Night’ configuration
 351 consistently maintains high precision, closely mirroring the ‘Baseline’ configuration across all training sample sizes.
 352 This suggests that changes in lighting have less impact on the model’s ability to detect objects compared to changes in
 353 viewpoint. This robustness to lighting could be attributed to the inclusion of diverse lighting conditions in the training

354 phase. Specifically, model performance benefited from pixel-wise augmentation techniques such as adjustments to hue,
355 saturation, and value (HSV). These augmentations introduced a variety of color variations to the images, enhancing the
356 model’s ability to generalize across different visual conditions. Moreover, these YOLO models benefit from pre-training
357 on the COCO dataset, which is characterized by a wide array of images with varied lighting, aiding their adaptability to
358 shifts in light.

359 On the other hand, the models perform suboptimally in scenarios involving changes in viewpoint. Each new viewpoint
360 introduces fundamentally different object features that are not replicated through standard data augmentation methods
361 such as lighting or affine transformations. For example, when the camera is placed at a lower angle, cows are more
362 frequently occluded by stalls and fences. These additional objects introduce variations that cannot be mitigated by
363 augmentations in HSV space or image translation. Consequently, ‘Top2Side’ performs the worst, as it is particularly
364 challenging to identify cows from the side. Even for the ‘External’ configuration, the model struggles to generalize well
365 despite being trained on the ‘Baseline’ configuration because the camera angle is changed again in the ‘External’ setup.
366 In summary, camera view angle is crucial for model generalization, with side views being the most challenging.

367 **Study 2: A Higher Model Complexity Does Not Always Lead to Better Performance**

368 The study found that the training configuration significantly affects the relationship between model complexity and
369 performance. Based on Study 1, predicting images from a side view using a model trained on Top-View camera images
370 is one of the most challenging tasks. In this configuration, increasing model complexity generally resulted in poorer
371 generalization, with simpler models often performing better. However, in other configurations that demonstrated better
372 generalization in Study 1, the peak performance was not always achieved by the most complex model. For example,
373 in the baseline configuration, the YOLOv9e model performed best in terms of mAP@0.5:0.95, mAP@0.5, and recall,
374 while the YOLOv8m model excelled in precision. Neither of these models had the highest parameter counts compared
375 to YOLOv8x. It is also worth noting that different model architectures showed different performance trends with varying
376 complexities. The YOLOv8-family models tended to perform best with mid-sized models (i.e., YOLOv8m), whereas
377 larger models in the YOLOv9 family usually performed better. Hence, the study concluded that model performance is
378 determined by both the training configuration and the model architecture.

379 The study results, as shown in **Figure 5**, indicate that although both YOLOv8 Ultralytics [Januray 2023] and YOLOv9
380 Wang and Liao [2024] models exhibit an increase in mAP@0.5 with more parameters when trained on the COCO
381 dataset Lin et al. [2014], this does not support a definitive conclusion that more parameters consistently improve model

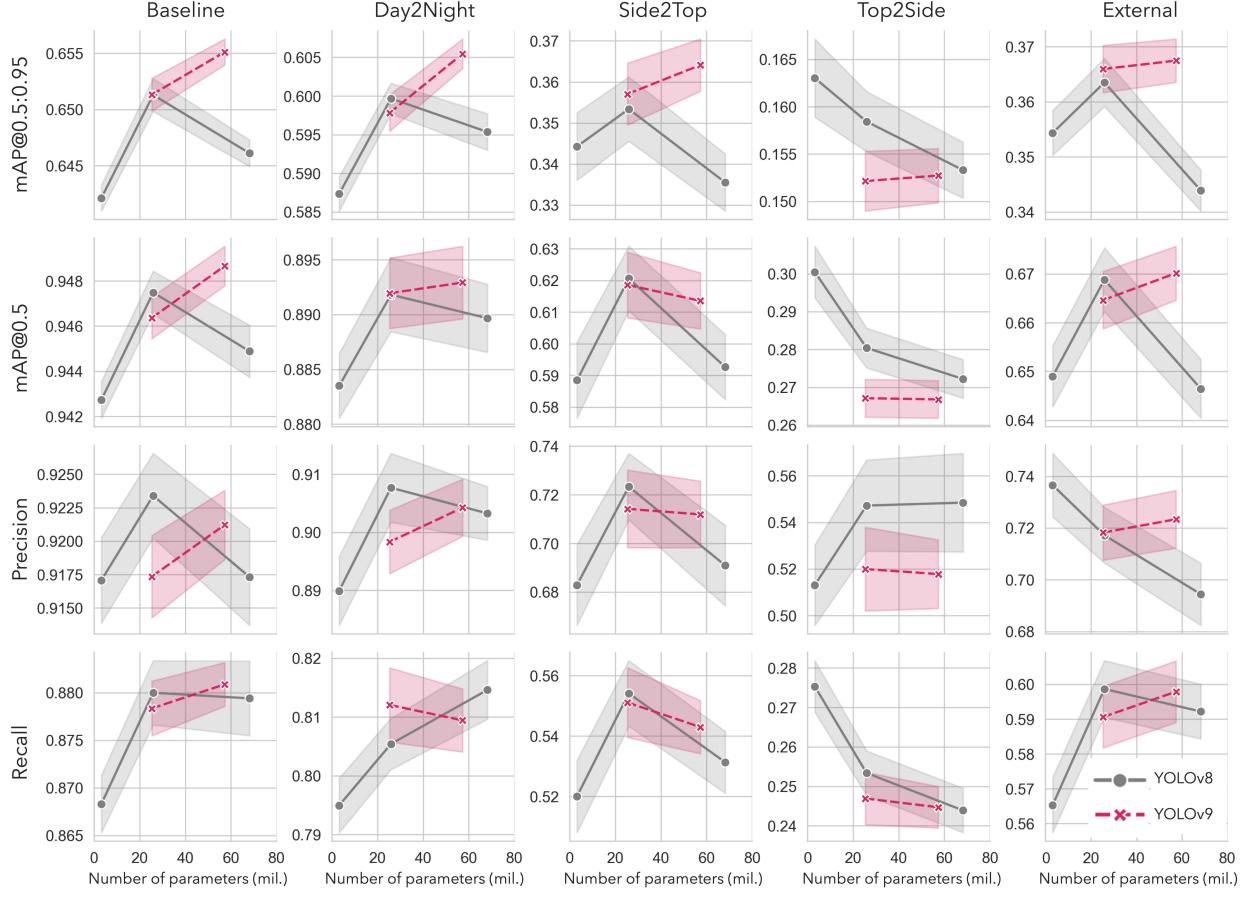


Figure 5: The performance of YOLOv8 and YOLOv9 models across various model parameters and data configurations, evaluated using four metrics: mAP@0.5:0.95, mAP@0.5, precision, and recall. Each column indicates a different data configuration, starting from top left to bottom right: ‘Baseline’, ‘Day2Night’, ‘Side2Top’, ‘Top2Side’, and ‘External’. The horizontal axis of all plots indicates the number of model parameters.

382 performance. This may be because the prior work’s findings were based on the COCO dataset, which includes 80
 383 classes and mainly features standalone images. In contrast, this study uses an indoor farm dataset focused exclusively
 384 on a single class: cows. Consequently, the model may not need as many parameters to effectively detect cows. This
 385 suggests that researchers should not rely solely on public dataset performance, as model generalization is specific to the
 386 task and dataset.

387 Additionally, this study found that a small model such as YOLOv8n, with only 3.2M parameters, can yield 90% accuracy
 388 with a relatively small size of training samples. This indicates that when one encounters a simple and homogenous task
 389 like positioning cows, deploying a small model is optimal in balancing computing time and prediction accuracy. This
 390 further underscores the importance of considering the specific characteristics of the task and dataset when choosing a
 391 model, rather than defaulting to more complex models under the assumption they will perform better.

392 Overall, our findings emphasize that higher model complexity does not necessarily lead to better performance. The
 393 optimal model configuration depends heavily on the specific task and dataset, highlighting the need for careful model
 394 selection tailored to the particular application at hand.

395 **Study 3: The Advantages of Custom Initial Weights are Limited When the Model is Simple**

396 The results presented in Figure 6 indicate that the benefit of using fine-tuned initial weights is minimal for simpler
 397 models. Specifically, when employing YOLOv8n, the performance difference between the default and fine-tuned
 398 weights was insignificant when fine-tuning data from the Top-View Camera and Side-View Camera. However, as model
 399 complexity increased, a greater number of fine-tuning samples were required for the two different initial weights to
 400 achieve similar performance. For instance, in the case of YOLOv9e, the performance gap was eliminated when the
 401 number of fine-tuning samples reached 128 and 64 for the Top-View Camera and Side-View Camera data sources,
 402 respectively. A similar trend was observed with the External camera, where a significant performance gap of more than
 403 25% in mAP@0.5:0.95 was observed for YOLOv9e when the sample size was 16. It is also noted that, although the
 404 performance gap was closed to zero for the Top-View Camera and Side-View Camera data sources, the gap was never
 405 closed for the External camera.

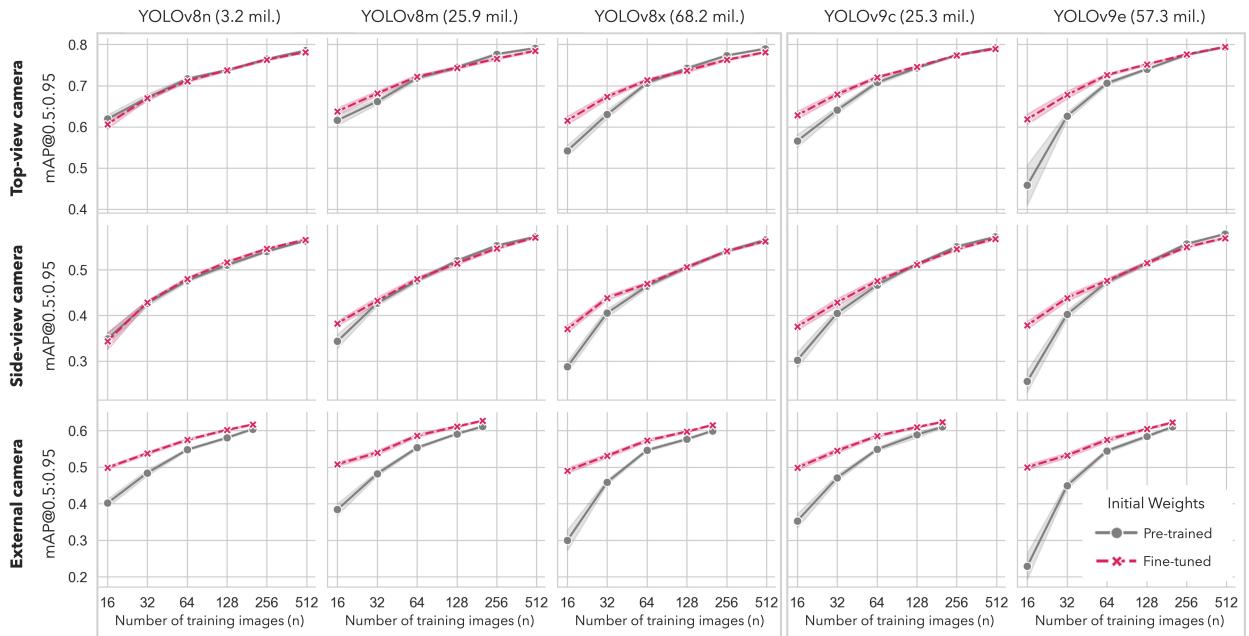


Figure 6: Varied generalization performance in mAP@0.5:0.95 with different initial weights. Red lines represent instances where weights were initialized with fine-tuned weights from other data configuration, while grey lines indicate scenarios employing pre-trained weights (i.e., trained with the COCO dataset). The horizontal axis indicates the number of training samples used for the fine-tuning procedure.

406 This study suggests that, for YOLO models with fewer parameters, such as YOLOv8n and YOLOv8m, the choice of
407 weight initialization does not make a significant difference in fine-tuning performance. In contrast, larger models like
408 YOLOv8x, YOLOv9c, and YOLOv9e exhibit improved performance when weights are initialized from a model that has
409 been previously fine-tuned in a similar data configuration, as described in Table 1. Therefore, when fine-tuning larger
410 models with a limited dataset, it is beneficial to utilize weights previously fine-tuned on various data configurations.

411 This approach leverages the additional learned features and adaptability from the initial fine-tuning, resulting in better
412 performance even with a small amount of new data. For example, our results showed that YOLOv9e achieved optimal
413 performance with fewer fine-tuning samples when initialized with fine-tuned weights compared to default weights.

414 Conversely, for smaller models, the weight initialization strategy does not significantly impact fine-tuning performance.
415 This is likely due to the lower complexity and fewer parameters of these models, which makes them less dependent on
416 the initial weight configuration to achieve good performance. In practical terms, this means that for simpler models,
417 researchers can save time and computational resources by directly fine-tuning without the need for customized weight
418 initialization.

419 The analysis of Figure 6 also provides insight into performance across homogeneous viewpoint data configurations,
420 specifically ‘Top-View Camera’ and ‘Side-View Camera’. The data demonstrates that the ‘Top-View Camera’ configu-
421 ration consistently yields higher mAP values regardless of the training sample size and weight initialization conditions.

422 This implies that the ‘Side-View Camera’ configuration, where both training and test images are captured from the side
423 view, presents a more formidable challenge for cow detection compared to the ‘Top-View Camera’ configuration. The
424 side view poses difficulties due to occlusions by neighboring cows and additional distractions, such as obstacles in
425 aisles and fences. Furthermore, cows located further away in side-view images may not be as visible, complicating
426 feature extraction. In contrast, the ‘Top-View Camera’ configuration benefits from an unobstructed aerial perspective,
427 ensuring that the top view of all cow instances is clearly visible and free from such obstructions. This distinction in
428 visibility between the two configurations contributes to the ease of feature extraction and ultimately, the performance
429 disparity observed.

430 These findings align with the results from Study 1, which demonstrated that changes in camera view angles dramatically
431 affect model performance. In Study 1, we found that models trained on Top-View datasets struggled the most to detect
432 cows from side-view images, with performance dropping by approximately 60% in mAP@0.5. This significant drop in
433 performance is attributed to the same reasons identified in Study 3: the side view introduces occlusions and distractions
434 that are not present in the top view, making feature extraction more challenging.

435 This study highlights that when working with external or unseen datasets, fine-tuning with custom initial weights trained
 436 on relevant tasks brings advantages to the detection tasks. On the other hand, simpler models do not benefit much from
 437 customized weights, suggesting that it is more efficient to train a simple model with pre-trained weights without relying
 438 on prior relevant information, which sometimes requires intensive labor efforts.

439 **Computational Resource Requirements**

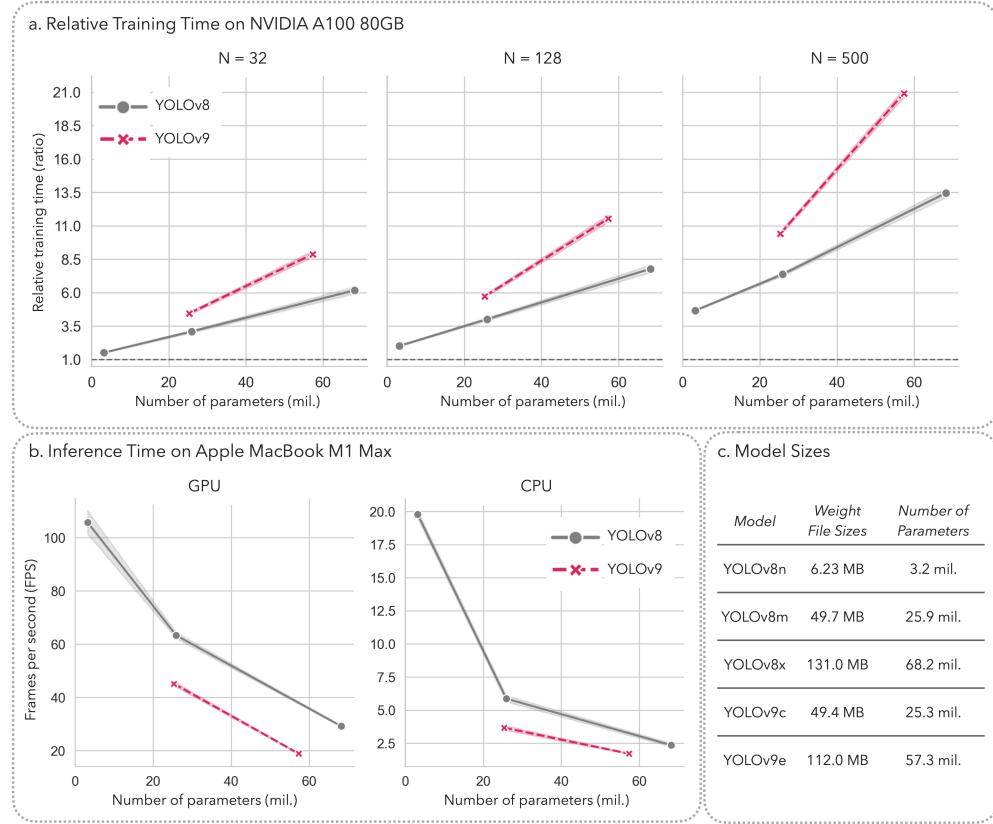


Figure 7: Comparative evaluation of computational resource requirements. (a) Training time (expressed as a multiple of baseline time) versus number of parameters for YOLOv8 and YOLOv9 models, presented for training sample sizes of 32 (left), 128 (middle), and 500 (right). (b) Inference frequency versus number of parameters for YOLOv8 and YOLOv9 models on GPU (left) and CPU (right). (c) A table displaying the weight sizes and parameter counts of various YOLOv8 and YOLOv9 models.

440 The evaluation of computational resource requirements is crucial for understanding the feasibility of deploying YOLO
 441 models in real-world applications, especially in environments with limited computational resources. This section
 442 compares training time (Figure 7a), inference time (Figure 7b), and model weight sizes (Figure 7c) for various YOLO
 443 models.

444 The training time for each model was measured and expressed as a multiple of the baseline training time, which is
 445 the time required to train the YOLOv8n model with 32 samples. The results indicated that using the largest model,

446 YOLOv8x, which has 20 times more parameters, increased training time by 4 to 6 times, depending on the training
447 sample size. Additionally, the YOLOv9 models generally required more training time and had slower inference frames
448 per second (FPS) compared to the YOLOv8 models. The gap in training time expanded as the number of training
449 samples increased.

450 Inference time was measured as the average FPS in a batch of 64 images. Running the models on a CPU with the
451 smallest model (YOLOv8n) was slower than running the largest model (YOLOv8x) on a GPU. For example, the FPS
452 for the small, YOLOv8n, on a CPU was 19.77, while the FPS for YOLOv8x on a GPU was 29.21. High FPS models
453 are essential for real-time inference, which usually requires a model with an FPS higher than 30. The results indicate
454 that implementing YOLO models on a CPU may not meet real-time requirements, especially for larger models.

455 Lastly, model weight sizes were also considered, impacting memory requirements and deployment feasibility, especially
456 in edge computing environments. The weight sizes and parameter counts of various YOLO models are displayed in
457 Figure 7c.

458 In conclusion, this evaluation highlights the trade-offs between model complexity and computational efficiency. The
459 larger YOLO models, while offering potentially better performance, require significantly more computational resources.
460 This analysis helps researchers and practitioners select the appropriate model based on the available computational
461 resources and the specific requirements of their application.

462 4 Conclusion

463 This study examined the impact of various training configurations and model complexities on the performance of
464 YOLOv8 and YOLOv9 models for cow detection in indoor farm environments. Our results indicate that model
465 performance is highly dependent on camera viewpoints, with side views presenting the greatest challenges. Additionally,
466 fine-tuning models with weights from similar datasets substantially enhances performance, particularly for complex
467 models in scenarios with limited data. We also introduce a public cow localization dataset, 'COLO', to support the
468 research community.

469 The findings indicate that while increasing model complexity can improve performance, this is not always the case,
470 especially in challenging configurations like 'Top2Side', which predict images from a side view using a model trained
471 on top-view images. Models trained on a single viewpoint exhibit limited generalization, underscoring the importance
472 of incorporating diverse and consistent camera angles in the training data.

473 Despite the promising results, this study has certain limitations. The models' performance was evaluated under specific
474 indoor farm conditions, which may not generalize to all livestock environments. Moreover, the reliance on pre-defined
475 configurations may limit the applicability of our findings to more dynamic settings.

476 Future work should explore adaptive methods for enhancing model generalization across varied viewpoints and
477 environmental conditions. Additionally, investigating the integration of advanced data augmentation techniques and
478 more diverse datasets could further improve detection accuracy and robustness.

479 In conclusion, this study offers practical insights into reproducing model performance in new environmental settings
480 and provides the public 'COLO' dataset to facilitate further research and advancements in the field.

481 **Acknowledgments**

482 This research was supported by the USDA Hatch Research Project funding VA-160196. The authors acknowledge
483 Advanced Research Computing at Virginia Tech for providing computational resources and technical support that have
484 contributed to the results reported within this paper. URL: <https://arc.vt.edu/>. During the preparation of this work the
485 author(s) used ChatGPT in order to ensure the grammar and clarity of the manuscript. After using this tool/service, the
486 author(s) reviewed and edited the content as needed and take(s) full responsibility for the content of the publication.

487 References

- 488 Colo: A large-scale dataset for conversational question answering over knowledge graphs. <https://huggingface.co/datasets/Niche-Squad/COLO>, 2023. Accessed: 2024-04-09.
- 489 Ultralytics v8 models. <https://github.com/ultralytics/ultralytics/blob/main/ultralytics/cfg/models/v8/yolov8.yaml>, 2023.
- 490 Ultralytics v9 models. <https://github.com/ultralytics/ultralytics/tree/main/ultralytics/cfg/models/v9>, 2023.
- 491 Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.
- 492 K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian. Centernet: Keypoint triplets for object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6569–6578, Seoul, Republic of Korea, 27 October–2 November 2019.
- 493 Stefan Elfwing, Eiji Uchibe, and Kenji Doya. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural networks*, 107:3–11, 2018.
- 494 Arthur Francisco Araújo Fernandes, João Ricardo Rebouças Dórea, and Guilherme Jordão de Magalhães Rosa. Image analysis and computer vision applications in animal sciences: an overview. *Frontiers in Veterinary Science*, 7:551269, 2020.
- 495 Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.
- 496 Dusty Greif. dgreif/ring, April 2024. URL <https://github.com/dgreif/ring>. original-date: 2018-10-12T22:53:01Z.
- 497 Karim Guirguis, Ahmed Hendawy, George Eskandar, Mohamed Abdelsamad, Matthias Kayser, and Jürgen Beyerer. Cfa: Constraint-based finetuning approach for generalized few-shot object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4039–4049, 2022.
- 498 Chhaya Gupta, Nasib Singh Gill, Preeti Gulia, and Jyotir Moy Chatterjee. A novel finetuned yolov6 transfer learning model for real-time object detection. *Journal of Real-Time Image Processing*, 20(3):42, 2023.
- 499 Xu Han, Zhengyan Zhang, Ning Ding, Yuxian Gu, Xiao Liu, Yuqi Huo, Jiezhong Qiu, Yuan Yao, Ao Zhang, Liang Zhang, et al. Pre-trained models: Past, present and future. *AI Open*, 2:225–250, 2021.

- 515 Wangli Hao, Chao Ren, Meng Han, Li Zhang, Fuzhong Li, and Zhenyu Liu. Cattle body detection based on yolov5-ema
516 for precision livestock farming. *Animals*, 13(22):3535, 2023.
- 517 Bharath Hariharan, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. Hypercolumns for object segmentation and
518 fine-grained localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages
519 447–456, 2015.
- 520 Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings*
521 *of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- 522 Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international*
523 *conference on computer vision*, pages 2961–2969, 2017.
- 524 Xia Hu, Lingyang Chu, Jian Pei, Weiqing Liu, and Jiang Bian. Model complexity of deep learning: A survey. *Knowledge*
525 *and Information Systems*, 63:2585–2619, 2021.
- 526 Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional
527 networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708,
528 2017.
- 529 Zhaojin Huang, Lichao Huang, Yongchao Gong, Chang Huang, and Xinggang Wang. Mask scoring r-cnn. In
530 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6409–6418, 2019.
- 531 Glenn Jocher. YOLOv5 by Ultralytics. <https://github.com/ultralytics/yolov5>, 2020. Accessed on: 28
532 February 2023.
- 533 Daniel Justus, John Brennan, Stephen Bonner, and Andrew Stephen McGough. Predicting the computational cost of
534 deep learning models. In *2018 IEEE international conference on big data (Big Data)*, pages 3873–3882. IEEE, 2018.
- 535 Simonyan Karen. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv: 1409.1556*,
536 2014.
- 537 Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural
538 networks. *Communications of the ACM*, 60(6):84–90, 2017.
- 539 H. Law and J. Deng. CornerNet: Detecting objects as paired keypoints. In *Proceedings of the European Conference on*
540 *Computer Vision (ECCV)*, pages 734–750, Munich, Germany, 8–14 September 2018.
- 541 Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien
542 Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, Joe Davison, Marija Čuklina, Simon Brandeis, Teven

- 543 Le Scao, Philipp Schmid, Sylvain Gugger, Clément Delangue, Théo Matussière, Lysandre Debut, Idan Ben-Ami,
544 Olga Filippova, Martin d’Hoffschildt, Sébastien Gérard, Brendan Lane, Leo Ansell, Lars Buitinck, Damien
545 Esposito, Mathis Raison, Jacob Klein, Thibault Nguyen, Tomoki Mikami, Victor Sanh, Vishrav Chaudhary, Nicolas
546 Patry, Wilson Y. Chang, Julien Froment, Jonas Buhmann, Quentin Malartic, Victor Winschel, Charlie Watson,
547 Rajarshi Pradeep, Gunjan Chhablani, Manuela Rohrbach, Maxim Jenny, John Bolton, Jason Phang, Theo Löw,
548 Alexander Rush, and Thomas Wolf. Datasets: A community library for natural language processing. <https://github.com/huggingface/datasets>, 2021.
- 550 Guoming Li, Yanbo Huang, Zhiqian Chen, Gary D Chess Jr, Joseph L Purwell, John Linhoss, and Yang Zhao.
551 Practices and applications of convolutional neural network-based computer vision systems in animal farming: A
552 review. *Sensors*, 21(4):1492, 2021.
- 553 Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence
554 Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference,*
555 *Zurich, Switzerland, September 6–12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014.
- 556 Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg.
557 Ssd: Single shot multibox detector. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The*
558 *Netherlands, October 11–14, 2016, Proceedings, Part I 14*, pages 21–37. Springer, 2016.
- 559 Massachusetts Institute of Technology. Labelme: Image annotation tool. <http://labelme.csail.mit.edu/>
560 Release3.0/, 2023.
- 561 Sarah Morrone, Corrado Dimauro, Filippo Gambella, and Maria Grazia Cappai. Industry 4.0 and precision livestock
562 farming (plf): an up to date overview across animal productions. *Sensors*, 22(12):4319, 2022.
- 563 Abozar Nasirahmadi, Barbara Sturm, Sandra Edwards, Knut-Håkan Jeppsson, Anne-Charlotte Olsson, Simone Müller,
564 and Oliver Hensel. Deep learning and machine vision approaches for posture detection of individual pigs. *Sensors*,
565 19(17):3738, 2019.
- 566 Dataset Ninja. Visualization tools for opencow2020 dataset. <https://datasetninja.com/opencows2020>, may
567 2024. URL <https://datasetninja.com/opencows2020>. visited on 2024-05-21.
- 568 Su Myat Noe, Thi Thi Zin, Pyke Tin, and Ikuo Kobayashi. Automatic detection and tracking of mounting behavior in
569 cattle using a deep learning-based instance segmentation model. *Int. J. Innov. Comput. Inf. Control*, 18(1):211–220,
570 2022.

- 571 Kian Eng Ong, Sivaji Retta, Ramarajulu Srinivasan, Shawn Tan, and Jun Liu. Cattleeyevew: A multi-task top-down
572 view cattle dataset for smarter precision livestock farming. In *2023 IEEE International Conference on Visual*
573 *Communications and Image Processing (VCIP)*, pages 1–5. IEEE, 2023.
- 574 OpenCV. Cvat: Computer vision annotation tool. <https://www.cvcat.ai/>, 2023.
- 575 Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object
576 detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.
- 577 Roboflow. Roboflow: Organize, annotate, and improve machine learning datasets. <https://roboflow.com/>, 2023.
- 578 Nahian Siddique, Sidike Paheding, Colin P Elkin, and Vijay Devabhaktuni. U-net and its variants for medical image
579 segmentation: A review of theory and applications. *Ieee Access*, 9:82031–82057, 2021.
- 580 Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv*
581 *preprint arXiv:1409.1556*, 2014.
- 582 Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent
583 Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on*
584 *computer vision and pattern recognition*, pages 1–9, 2015.
- 585 Eric T. Psota, Ty Schmidt, Benny Mote, and Lance C. Pérez. Long-term tracking of group-housed livestock using
586 keypoint detection and map estimation for individual animal identification. *Sensors*, 20(13):3670, 2020.
- 587 Sasha Targ, Diogo Almeida, and Kevin Lyman. Resnet in resnet: Generalizing residual architectures. *arXiv preprint*
588 *arXiv:1603.08029*, 2016.
- 589 Z. Tian, C. Shen, H. Chen, and T. He. Fcos: Fully convolutional one-stage object detection. In *Proceedings of the*
590 *IEEE/CVF International Conference on Computer Vision*, pages 9627–9636, Seoul, Republic of Korea, 27 October–2
591 November 2019.
- 592 Naftali Tishby and Noga Zaslavsky. Deep learning and the information bottleneck principle. In *2015 ieee information*
593 *theory workshop (itw)*, pages 1–5. IEEE, 2015.
- 594 Naftali Tishby, Fernando C Pereira, and William Bialek. The information bottleneck method. *arXiv preprint*
595 *physics/0004057*, 2000.
- 596 Shuqin Tu, Weijun Yuan, Yun Liang, Fan Wang, and Hua Wan. Automatic detection and segmentation for group-housed
597 pigs based on pigms r-cnn. *Sensors*, 21(9):3251, 2021.
- 598 Ultralytics. Ultralytics models documentation. <https://docs.ultralytics.com/models/>. Accessed: 2024-05-21.

- 599 Ultralytics. Ultralytics datasets documentation. <https://docs.ultralytics.com/datasets/detect/>, 2023.
- 600 Ultralytics. YOLOv8 — docs.ultralytics.com. <https://docs.ultralytics.com/models/yolov8/#overview>,
- 601 Januray 2023.
- 602 Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the*
603 *2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001*, volume 1, pages
604 I–I. Ieee, 2001.
- 605 Chien-Yao Wang and Hong-Yuan Mark Liao. YOLOv9: Learning what you want to learn using programmable gradient
606 information. 2024.
- 607 Danqing Xu and Yiquan Wu. Improved yolo-v3 with densenet for multi-scale remote sensing target detection. *Sensors*,
608 20(15):4276, 2020.
- 609 Zhenwei Yu, Yuehua Liu, Sufang Yu, Ruixue Wang, Zhanhua Song, Yinfa Yan, Fade Li, Zhonghua Wang, and Fuyang
610 Tian. Automatic detection method of dairy cow feeding behaviour based on yolo improved model and edge computing.
611 *Sensors*, 22(9):3271, 2022.
- 612 Thi Thi Zin, Moe Zet Twint, Pann Thinzar Seint, Shin Thant, Shuhei Misawa, Kosuke Sumi, and Kyohiro Yoshida.
613 Automatic Cow Location Tracking System Using Ear Tag Visual Analysis. *Sensors*, 20(12):3564, January 2020.
614 ISSN 1424-8220. doi:10.3390/s20123564. URL <https://www.mdpi.com/1424-8220/20/12/3564>. Number:
615 12 Publisher: Multidisciplinary Digital Publishing Institute.

616 Appendix

617 Hyperparameters in Ultralytics library

618 The table below show the hyperparameters used in the Ultralytics library for training the models in this study.

Table 3: Hyperparameters for the training procedure

Hyperparameters	Description	Value
epochs	Number of training epochs	100
batch	Number of images in each batch	16
optimizer	Optimizer used for training	auto
hsv_h	Altering the hue value of the image	0.015
hsv_s	Altering the saturation of the image by a fraction	0.7
hsv_v	Altering the brightness of the image by a fraction	0.4
translate	Randomly translating the image by a fraction of the image size	0.1
scale	Randomly scaling the image by a fraction of the image size	0.5
fliplr	Randomly flipping the image horizontally with the given probability	0.5
mosaic	Combining four images into one mosaic image with the given probability	1.0
mixup	Randomly mixing up the object instances across multiple images with the given probability	0.15
copy_paste	Randomly copying and pasting the object instances across multiple images with the given probability	0.3