Wiley
International Transactions on Electrical Energy Systems
Volume 2025, Article ID 8669289, 20 pages
https://doi.org/10.1155/etep/8669289

WILEY

Research Article

A Systematic YOLO-Specific Model Selection for Mechanical Fault Identification in High-Voltage Insulators

Arailym Serikbay, Venera Nurmanova, Yerbol Akhmetov, Amin Zollanvari, and Mehdi Bagheri

Electrical and Computer Engineering Department, School of Engineering and Digital Sciences, Nazarbayev University, Astana 010000, Kazakhstan

Correspondence should be addressed to Arailym Serikbay; arailym.serikbay@nu.edu.kz and Mehdi Bagheri; mehdi.bagheri@nu.edu.kz

Received 18 November 2024; Revised 11 July 2025; Accepted 9 August 2025

Academic Editor: Jagabar Sathik

Regular monitoring of outdoor insulators is crucial to ensure the reliable functioning of the power grid. With recent progress in computer vision technologies, traditional manual and expensive visual inspections can now be replaced by automated analysis using images captured by unmanned aerial vehicles (UAVs). In such applications, a practitioner might opt to choose a state-of-the-art object detection and classification deep learning architecture, including You Look Only Once (YOLO). The variety of existing YOLO architectures *per se* makes selecting the best application-dependent YOLO model challenging. However, selecting the best architecture solely based on performance without considering the model complexity limits its deployment on resource-limited embedded devices. Consequently, we conduct a rigorous, systematic model selection based on the performance–complexity trade-off across 13 YOLO architectures to determine the most effective model for detecting common mechanical faults in insulators using images captured by UAVs. A dataset comprising 15,000 images of insulators, categorized into normal condition, bird-pecking damage, cracks, and missing caps, has been compiled for training the models. Specifically, all considered YOLO architectures are compared using model complexity and the mAP@0.5:0.95. During the model selection stage, YOLOv8l proved to be the best model in terms of mAP@0.5:0.95, while YOLOv5n was the model of choice in terms of complexity at the expense of a slight reduction in performance. Alongside YOLOv8l and YOLOv5n, an "optimal" model (OP-YOLO) was selected using a multicriteria decision-making approach, balancing detection accuracy and computational efficiency. In particular, in terms of test-set performance, YOLOv8l, YOLOv5n, and OP-YOLO achieved 0.919, 0.901, and 0.896 mAP@0.5:0.95, respectively. Although YOLOv8l reported a higher mAP@0.5:0.95, YOLOv5n requires ~20.9 times less memory and ~40.2 times less floating-point operations per second (FLOPs). Also, YOLOv5n outperforms the OP-YOLO model, still requiring ~12 times less memory and ~19 times less FLOPs.

Keywords: faster R-CNN; insulator diagnostics; mechanical damage; missing cap; UAV image detection; YOLO

1. Introduction

An overhead line insulator is a crucial electrical component utilized for fastening and isolating the conductors from earthed overhead transmission and distribution lines. Porcelain, glass, or polymers are commonly used materials in insulator structures in the high-voltage industry. The linear insulators are used to fix the rigid or flexible current-carrying parts, hardware insulators separate the current-carrying parts from neutral or grounded circuit elements, and checkpoint insulators are utilized to pass a conductor or a busbar through the structures via separating the conductor from the construction structure and other grounded equipment. Overhead power lines and outdoor insulators usually operate under harsh environmental conditions and are exposed to different pollutants along with other natural factors, such as lightning strikes [1]. Such conditions can degrade the insulating properties and

mechanical integrity [2] of power line components, potentially leading to reduced power quality, blackouts, or even irreversible damage to the power grid infrastructure [3]. Therefore, timely and thorough maintenance and fault detection are among the highest priorities for utility managers, owners, and electricity providers [4].

The degradation of insulating materials due to environmental aging, contamination, and mechanical stress has been extensively investigated in recent studies focusing on composite materials for high-voltage applications [5, 6]. For example, Ersoy et al. [5] analyzed the influence of particle size on the performance of polyurethane–mica composites used in insulation, while their subsequent work [6] introduced a novel solid dielectric material with enhanced electrical and mechanical properties for transformer windings.

The operation of overhead power lines includes maintenance (operational maintenance), overhaul, and elimination of the damage on overhead lines. Currently, there are various assessment methods developed for the insulator diagnostic, such as electrical, mechanical, galvanizing, and routine tests [7]. They can be classified into contact and noncontact, electrical and nonelectric, sound and visual, and combined methods. Contact methods are mainly used when the line is de-energized, and insulators are offline. Such methods are accompanied by the removal of insulators from the supports, which has its own risk and cost for transmission line companies and personnel safety threats. Alternatively, noncontact methods can be applied without de-energizing the power lines, thus reducing the risks associated with dismantling insulators at high altitudes [8]. Therefore, besides conventional standardized diagnostic tests, advanced noncontact inspection techniques have been actively developed and applied: ultrasonic method [9], ultraviolet (UV) [10], infrared (IR) image test [11], radio interference (RI) test [12], self-normalized photothermal radiometry [13], manned aerial inspection [14], and unmanned aerial vehicle (UAV) imaging methods are some of those techniques to name a few [15].

Among these novel techniques, insulator diagnosis via UAV imaging has recently been introduced and is even being actively applied in the industry since it offers an economical, personnel safety, and low-maintenance cost solution. Nonetheless, in recent years, a class of machine learning models known as deep neural networks (DNNs) has become quite desirable for UAV image processing. It is proposed that the detection of insulator faults using UAV images takes a major leap through the application of DNNs [16]. This state of affairs is generally attributed to the efficiency and the accuracy of DNN-based image processing compared to visual inspection or even compared with conventional image processing techniques based on filtering, segmentation, and spatial conversion [16].

Commonly, deep learning–based object detection algorithms can be classified into one-stage and two-stage methods, each having its benefits and drawbacks. As the name suggests, one-stage object detection performs both classification and bounding box regression using a single neural network in a unified pipeline. This distinction does not apply to two-stage detectors, where the first neural network generates the region of interest (RoI), and the second network performs the classification and bounding box prediction [17]. The cases of two-stage object recognition algorithms are region-based convolutional neural networks (R-CNN), fast R-CNN, faster R-CNN, and Mask R-CNN, whereas one-stage detectors are RetinaNet, single-shot multibox detector (SSD) [17], and You Only Look Once (YOLO) [17]. Normally, one-stage models require lower computational costs and high-speed processes, while two-stage detectors have high detection and classification precision.

Hu and Li [18] applied channel attention along with faster R-CNN and U-Net architectures for insulator detection, reaching 0.919 mean average precision (mAP). Wang and Yi [19] reported 0.945 mAP in insulator detection while incorporating a Gaussian distribution into bounding box prediction in YOLOv3. Han et al. [20] suggested the enhancement of YOLOv3-tiny via the introduction of the spatial pyramid pooling (SPP) architecture and RoI for the detection and localization of the missing cap fault. Sampedro et al. [21] proposed an architecture combined with a fully convolutional Up-Net network for segmentation, CNN, and Siamese CNN for insulator diagnostic purposes, resulting in 0.993 mAP. Xuan et al. [22] proposed one-stage center mask-based object detection to define insulator missing plates from UAV-taken and noise-introduced images, reporting accurate and high-speed performance of the proposed technique. Wei et al. [23] compared the performance of Faster R-CNN, RetinaNet, and YOLOv3 models to detect and localize the string insulator missing cap as a result of self-explosion. Miao et al. [24] applied the SSD model to aerial images to classify between porcelain and composite insulators, reporting 0.938 and 0.853 precision in detecting porcelain and composite insulators, respectively. Zan et al. [25] observed that the traditional YOLOv4-tiny architecture surpasses faster R-CNN and EfficientNet in the detection of the broken overhead line insulator, resulting in 0.916 mAP compared to 0.828 and 0.861 for the latter two architectures, respectively. Feng et al. [26] applied YOLOv5 and its variants on UAV-taken images to identify and localize the defective part of the string insulator and reached the maximum mAP of 0.995. Serikbay et al. [27] trained a CNN model to classify outdoor insulator surface contamination types. The proposed model selection and classifier size optimization method led to ~3 times lighter architecture at the expense of a 6.5% accuracy reduction. Mussina et al. [28] utilized a fusion convolutional network (FCN) for UAV-based insulator surface condition monitoring. In particular, deep CNN and multilayer neural networks (MNN) are combined for image classification. The training image dataset included original insulator images and augmented data, including images with Gaussian, Poisson, Salt and Pepper, and Speckle noise.

The YOLOv8 model was used to examine the condition of high-voltage insulators with mechanical faults, such as cracks, as well as insulators with surface contamination [29]. For training, publicly accessible datasets, including the Chinese Power Line Insulator and Insulator Defect Image

datasets, were used. As a result, the selected model achieved mAP values of 0.861 and 0.989, respectively. Additionally, an enhanced version of YOLOv8 incorporating the ResPANet module and a feature extraction module developed using GhostNet was introduced [30]. Compared to the conventional YOLOv8, which has a mAP values of 0.892, the enhanced architecture achieved an accuracy of 0.939.

This study employs various versions of YOLOv3, YOLOv5, and YOLOv8 models to perform both insulator identification and classification of mechanical defects. Specifically, the evaluation involves YOLOv3 variants (YOLOv3, YOLOv3-tiny, YOLOv3-SPP), YOLOv5 variants (YOLOv5n, YOLOv5s, YOLOv5m, YOLOv5x, YOLOv5l), and YOLOv8 variants (YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8x, YOLOv8l) for the classification of bird-pecking damage, cracks, and missing cap faults.

The contributions of this work are threefold:

1. Creation of a pipeline to identify the mechanical integrity of HV insulators using YOLO technology, along with the creation of a dataset of 15,000 images of different mechanical damages on polymer, porcelain, and glass insulators, particularly images of bird-pecking damage, cracks, and missing cap faults at different background scenery, imaging angles, and brightness.
2. Using UAV-captured images and employing YOLOv3, YOLOv5, and YOLOv8 architectures and their variants for insulator detection and mechanical damage classification.
3. Analysis of performance and computational complexity, including a comparative evaluation of implemented architectures and deployment techniques.

The rest of this study is also organized as follows: Section 2 discusses the YOLOv3, YOLOv5, and YOLOv8 architectures used for the object detection and classification. Section 3 discusses the utilized test objects and data collection procedure. Model selection is described in Section 4. The experimental results and discussion are provided in Section 5 and Section 6, respectively. Section 7 concludes the entire analysis and observations.

2. YOLO Object Detection Models

The constructed intelligent insulator detection and fault classification in this work is based on YOLOv3 [31], YOLOv5 [32], and YOLOv8 [33] architectures and their variants, including YOLOv3-tiny, YOLOv3-SPP, YOLOv5n, YOLOv5s, YOLOv5m, YOLOv5l, YOLOv5x, YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8x, and YOLOv8l. As discussed earlier, YOLO is a single-stage object detection approach that performs end-to-end processing, where both object detection and classification tasks are conducted in one single neural network [34].

In the YOLO framework, the input image is segmented into an $S \times S$ grid, as shown in Figure 1. The orange rectangle in Figure 1 is the predicted bounding box generated around the object. The center of the bounding box, indicated by a red dot, lies inside the prediction cell highlighted with green color. For $S \times S$ grid, the YOLO model generates an output feature map of size $S \times S \times [B * (C + 5)]$, where B is the number of boundary box predictions per cell and C is the number of classes. In particular, each of B bounding boxes is composed of the following elements: objectness score p_O (the likelihood of box having an object), t_x and t_y represent the center of the bounding box; t_h and t_w represent height and width of the bounding box, respectively; and p_C is the class probability (the likelihood of the object belonging to a certain class)—the "5" in the feature mentioned above map size is due to the coordinates of the bounding box, bounding box dimensions, and objectness score [34]. In YOLO, bounding box dimensions are determined by predicting the offsets (aforementioned t_x, t_y, t_w, and t_h) from universally defined 9 anchor box priors (i.e., 3 anchor box priors with different dimensions and shapes for each prediction scale) [34]. The sizes and shapes of the anchor box priors were predefined in the original YOLOv3 paper [31] by deploying k-means clustering over the COCO dataset.

2.1. YOLOv3, YOLO-Tiny, and YOLO-SPP. YOLOv3 is based on YOLO and YOLOv2, originally introduced by [31]. YOLOv3 utilizes the concept of deep residual network, skip connections, and upsampling layers and is based on the deeper CNN Darknet-53. The ultimate architecture of YOLOv3 consists of 106 layers, where object detection is carried out in layers 79, 91, and 103, and the feature maps are generated in layers 82, 94, and 106, see Figure 2.

As the name states, YOLOv3-tiny is a simplified variant of the YOLOv3, where the prediction accuracy is slightly decreased as a trade-off for decreased processing time, hardware, and memory requirements [35]. In YOLOv3-tiny, the downsampling of feature maps is conducted in max-pooling layers, in contrast to the original YOLOv3, where it is carried out in the convolutional layers [36].

YOLOv3-SPP is an enhanced version of YOLOv3 by adding an SPP module to the original architecture (between fifth and sixth convolution layers), which can result in a higher prediction accuracy while maintaining high processing speed [36]. SPP module generates feature vectors without image cutting or scaling and is robust to the diversity of feature map sizes. Therefore, the addition of an SPP module facilitates the accurate detection of different-sized insulators from a given image. Although the downsampling is conducted in convolutional layers in both YOLOv3 and YOLOv3-SPP, the feature selection is performed in an improved SPP [36].

2.2. YOLOv5n, YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x. YOLOv5 is a one-stage object recognition architecture that enhances the capabilities of its predecessor, YOLOv4. YOLOv5 is aggregated from five main modules: input, backbone (CSP), neck (PAnet), head, and output. YOLOv5 has a focus layer incorporated into the backbone CNN that reduces model parameters, floating-

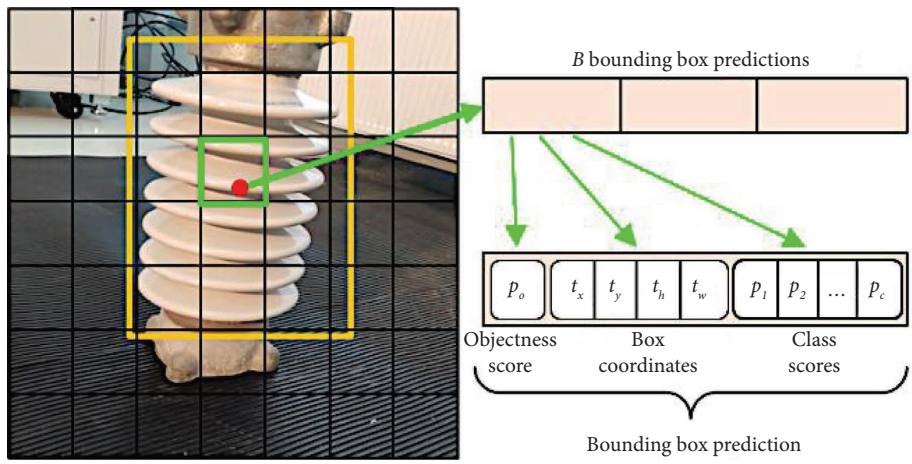


Figure 1: Bounding box prediction with an $S \times S$ grid.

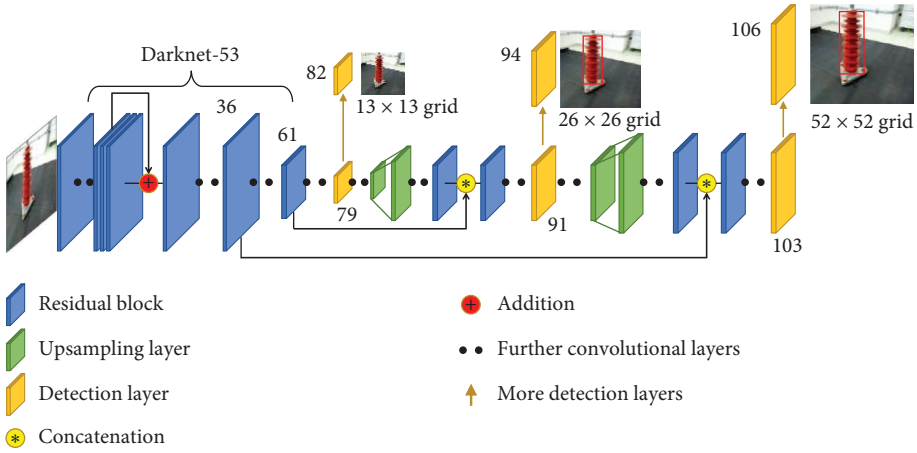


Figure 2: Structure of the YOLOv3 detection framework.

point operations per second (FLOPS), memory, and processing time while maintaining high prediction accuracy [37].

YOLOv5 has several versions based on the memory requirements [38]. This work focuses on YOLOv5n (nano), YOLOv5s (small), YOLOv5m (medium), YOLOv5l (large), and YOLOv5x (extra-large) variants, for insulator mechanical damage detection [36]. YOLOv5n has a more compact architecture and lower network depth and width, while YOLOv5x requires more memory and an almost large network size at the expense of the computational speed [38]. Other than memory requirements, these variants also vary in the number of model parameters, feature extraction modules, and convolution kernels [38, 39].

2.3. YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l, and YOLOv8x. Similar to YOLOv5, YOLOv8 [33] includes several variations, each differing in terms of architectural complexity, performance, and memory requirements: YOLOv8n (nano), YOLOv8s (small), YOLOv8m (medium), YOLOv8l (large), and YOLOv8x (extra-large). YOLOv8n is the smallest and fastest model, achieving around 105 frames per second on a GTX 1060 GPU [31]. In contrast, YOLOv8x is the most accurate but the slowest architecture, operating at 17 FLOPs, with a significantly higher memory requirement, highlighting the speed and accuracy trade-off. YOLOv8 incorporates several pretrained models for applications as detecting objects, segmenting individual instances, and classifying images. These models, along with the corresponding checkpoints for instance segmentation, were trained using the COCO detection dataset with an image size of 640×640 pixels [33].

3. Data Collection

As part of the data acquisition process, an image dataset containing suspension and post-type insulators exhibiting various mechanical defects is collected to enable effective training and evaluation of the selected YOLO architectures. For this purpose, three frequently encountered insulator mechanical damages, including bird pecking, crack, and missing cap faults, are inflicted on composite, glass, and porcelain insulators, see Figure 3. Insulator parameters are given in Table 1.

The goal was to create an insulator dataset that would allow for controlled modeling of faults while ensuring that

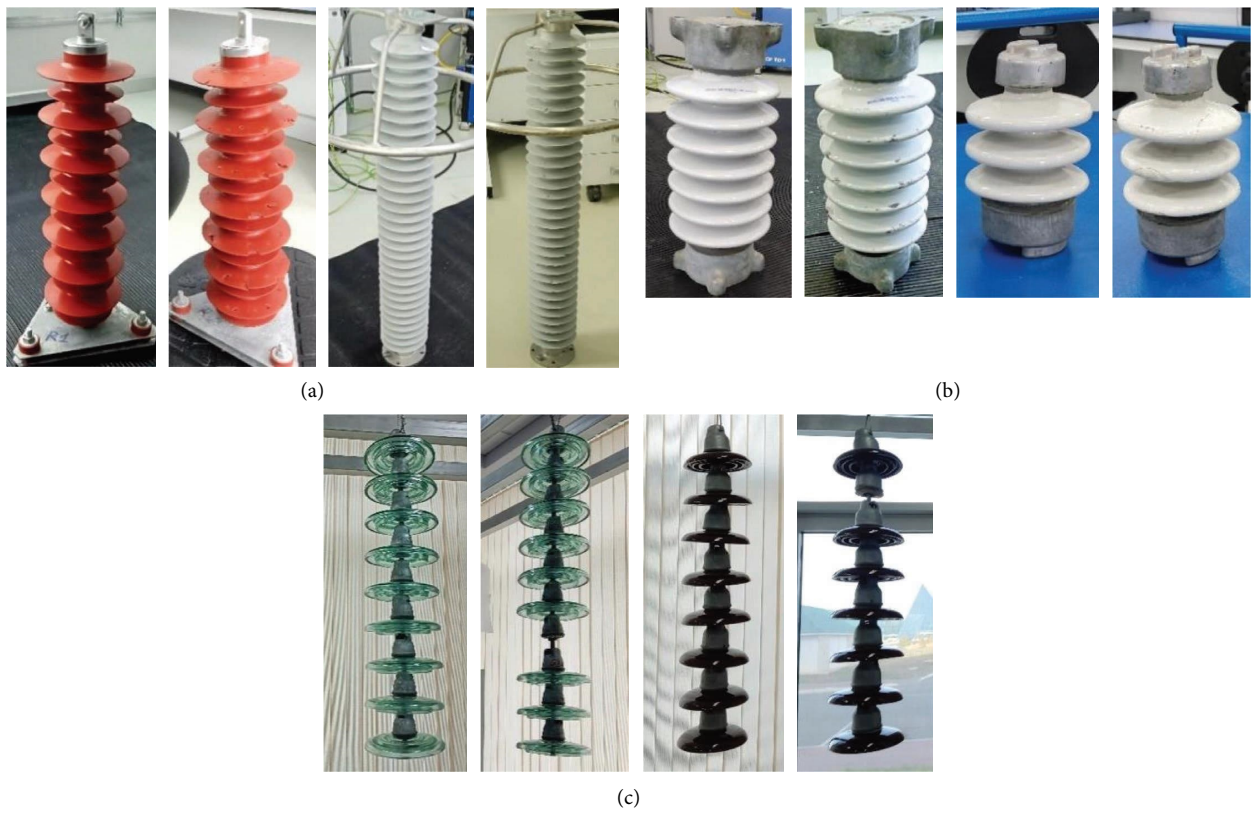


FIGURE 3: Various types of mechanical damage in insulators: (a) Bird-pecking damage on polymer insulators. (b) Cracking observed in porcelain insulators. (c) Instances of missing caps in both glass and porcelain insulators.

the model could also be utilized for outdoor real conditions. The images are collected in a high-voltage laboratory environment under various illumination conditions and camera perspectives. Indeed, the dataset was collected in an indoor environment with a temperature of ~20°C and a relative humidity of ~50% in varying lighting conditions. For the missing cap faults, the caps of both glass and brown porcelain insulators were removed step by step, altering the position of the missing cap along the insulator string. This method enabled the simulation of all possible scenarios that are encountered in real practical conditions. For cracks and bird-pecking damages, the proposed faulty insulator models with different severities were emulated using proper tools, ensuring that a wide spectrum of damage severities, ranging from light to severe, are represented. Finally, the number of collected images for each insulator category and type of mechanical damage is reported in Table 2. Overall, 8886 images of normal and damaged post- and suspension-type insulators are captured.

As shown in Table 2, insulator images with mechanical fault classes (bird pecking damage, missing cap, and cracks) contain ~1000–200 images, whereas the normal insulator class for the insulator type has only ~300–400 images. To address the class imbalance, a targeted data augmentation strategy was deployed. To prevent data leakage, the original dataset was first split into training, validation, and test subsets, and then, data augmentation was applied separately within each subset. Specifically, 15% of the entire dataset was randomly sampled to form the test set. The remaining 85% was then randomly divided into the training and validation sets using a ratio of 85% for training and 15% for validation. This split strategy resulted in three subsets consisting of 6412 (training), 1136 (validation), and 1338 (test) images, respectively. This approach was selected to ensure consistency across all model evaluations and enable reproducible comparison under identical conditions. Fixed splits are widely adopted in object detection research, particularly with YOLO-based models, due to the high computational cost of full k-fold cross-validation and the tendency of performance metrics to stabilize with sufficiently large training sets. Several recent studies have followed similar non–cross-validated strategies while achieving strong detection performance, for example, Han et al. [40] with an 80/10/10 split, Zhang et al. [41] with 70/15/15, and Chen et al. [42] with 81/9/10, each reporting mAP scores above 90% without cross-validation. Given our dataset scale and the consistency of this approach in the literature, we consider this split both practical and robust for evaluating model performance.

After splitting, data augmentation was applied separately within each subset. In particular, single-sample independent augmentation techniques (rotation, shifts, zoom, shear, and flip) were employed to synthetically increase class-specific sample size to 1250 images, ensuring a balanced dataset and minimizing prediction bias (see Table 3). These augmentation techniques, selected based on prior studies [43],

Table 1: Insulator parameters.

Insulator	Type	Operating voltage	Usage
Red polymer	Post	33 kV	Lightning and switching surge protection in power substations, in Figure 3(a)
White polymer	Post	110 kV	Protection of electrical equipment against switching and lightning surges, in Figure 3(a)
White porcelain	Post	138 kV	Structural support and insulation of power lines in power plants and stations, in Figure 3(b)
White porcelain small	Post	10 kV	Mechanical support and insulation of current-carrying parts in high-voltage disconnecting switches and switchgears, in Figure 3(b)
Glass	Suspension	11 kV/disc	Electrical insulation and support of conductors and thunder-proof cables on transmission lines and in switchgears in power systems, in Figure 3(c)
Brown porcelain	Suspension	11 kV/disc	Mechanical support and insulation of overhead power lines and switchgears in substations and power plants, in Figure 3(c)

Table 2: The dataset of insulators with different mechanical damages.

Insulator type	Normal	Bird pecking	Crack	Missing cap
Red polymer	369	1207	—	—
White polymer	373	1256	—	—
White porcelain	408	—	967	—
White porcelain small	258	—	1197	—
Glass	316	—	—	1058
Brown porcelain	366	—	—	1011
Total number of original images (before augmentation)	2090	2463	2164	2169
Number of artificially augmented images	5410	37	336	331
Total dataset size after augmentation	7500	2500	2500	2500

simulate real-world variations, helping the model learn more generalized and invariant features. For example, rotation and shifts improve spatial robustness, zoom and shear enhance scale and distortion tolerance, and flipping introduces symmetry awareness. Additionally, input rescaling (1/255) was applied for normalization.

To ensure balanced representation across all classes, a class-specific augmentation strategy was applied to increase each class to 1250 images. Augmentation was performed independently within the training, validation, and test subsets, proportionally to their original distributions, resulting in the addition of 4426, 776, and 912 augmented images to the training, validation, and test sets, respectively. This approach preserved the statistical structure of the original data and prevented data leakage between subsets. To mitigate overfitting, augmentation techniques (e.g., rotation, shift, zoom, shear, flip) were applied with randomized parameters per image, enhancing variation and model generalization.

As a result of the augmentation, the final dataset size increased to 15,000 images of clean and damaged insulators (see Table 2). Accordingly, the training, validation, and test sets consisted of 10,838, 1912, and 2250 images, respectively, and were used for model training (see Section 4), selection, and evaluation (see Section 5). The training set was used to adjust and optimize the weights of each YOLO model, the validation set was used for model selection, and the test set was reserved for final evaluation. Access to the dataset used in this study is available upon request to the corresponding author.

4. Model Training and Selection

In this study, three architectures of YOLOv3, five architectures of YOLOv5, and five architectures of YOLOv8 are examined in insulator detection and mechanical damage classification. Specifically, YOLOv3 [29], YOLOv3-SPP (SPP-based YOLOv3) [34], YOLOv3-tiny [37], YOLOv5n, YOLOv5s, YOLOv5m, YOLOv5l, YOLOv5x [36], YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l, and YOLOv8x [31] models are trained and compared.

In the training process, each model is optimized by updating the weights via the backpropagation algorithm using the training set. All training procedures were conducted using 8 × NVIDIA Tesla V100 graphics processors, each equipped with 32 GB of VRAM. The system had dual Intel Xeon E5-2698v4 CPUs (20 cores/40 threads, 2.2 GHz base), 512 GB DDR4 RAM, and utilized Ubuntu Linux as the operating system with Python 3.8.6 (built with GCCcore 10.2.0) and CUDA 11.7.0. The TensorFlow version 2.13.1 served as a deep learning framework and the training configuration details are provided in Table 4.

4.1. Model Selection by Performance and Complexity. The deployed model selection is a two-stage process based on the model performance on the validation set. In Stage I, the best epoch for each model is identified; that is, the epoch that results in the highest mAP@0.5:0.95. Stage II determines the "best" model out of the 13 best individual models identified in Stage I.

Stage I: Each YOLO architecture training was performed over 100 epochs with a batch size of 16 using the training dataset. The models were trained for an input image resolution of 640 × 640. We employed the Adam optimizer [44] with an initial learning rate of 0.01 with the learning rate decay strategy. Regularization across all models was achieved primarily through weight decay (set to 0.0005), which is supported by default in YOLOv3, YOLOv5, and YOLOv8 implementations. Dropout is not typically used in YOLO architectures and was not employed in this study. Early stopping was deliberately avoided to maintain consistent training conditions across all models and prevent premature termination, especially given that the goal was to compare complete training trajectories. Instead, we tracked validation loss and mAP trends over all 100 epochs to evaluate the robustness and stability of each model's performance (see Figure 4(a)). Deployed models were trained for 100 epochs to ensure sufficient convergence. Cross-validation was not used due to the high computational cost of training 13 models for 100 epochs each.

After each training epoch, the performance of each model was evaluated using mAP@0.5:0.95 score computed on the validation set (see Table 5). Finally, the epoch that resulted in the highest mAP@0.5:0.95 was designated as the best epoch, and the corresponding trained model was stored. Figure 4(a) presents mAP@0.5:0.95 as a function of epoch for all deployed architectures. For better visualization and comparison, Figure 4(b) illustrates the variation of validation set-specific mAP@0.5:0.95 in the last 25 training epochs. As it can be seen from Figure 4, accuracy typically plateaus between 40 and 100 epochs and reaches a peak performance. Within this range, additional training shows marginal

Table 3: Data augmentation techniques.

Parameter	Description and value	Purpose
Rotation	Randomly rotates images within a range of 25°	Simulates variation in object orientation
Width/height shift	Shifts the image horizontally/vertically/by up to 10% of the width/height and randomly zooms	Scales variations and varying object distances
Zoom	Uses shear transformation	Adds robustness
Shear	Applies random horizontal mirroring	Enhances symmetry learning
Horizontal flip	Normalizes image pixel intensity values to fall within the normalized range of (0, 1)	Improves training stability and convergence speed
Rescale		

Table 4: Training configuration details.

Component	Details
GPUs	8 × NVIDIA Tesla V100
CPUs	Dual Intel Xeon E5-2698v4
RAM	512 GB DDR4
Operating system	Ubuntu Linux
Python version	Python 3.8.6
CUDA version	CUDA 11.7.0
Deep learning framework	TensorFlow 2.13.1

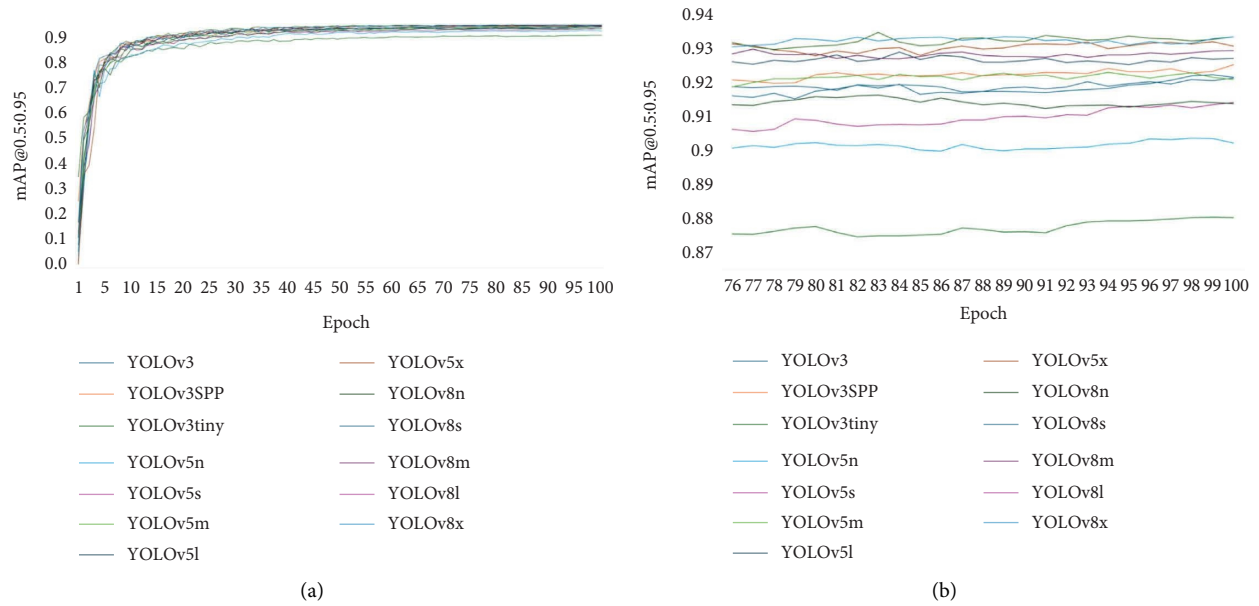


Figure 4: The progression of mAP@0.5:0.95 scores over 100 epochs for 13 YOLO models. (a) Graph illustrating performance trends throughout the complete training duration. (b) Graph illustrating performance trends over the last 25 epochs for better readability.

Table 5: Performance metrics of various YOLO models with 640 × 640 resolution inputs.

Models	Precision	Recall	mAP@0.5	mAP@0.5:0.95	Epoch	Layers	Memory (MBFP16)	GFLOPs
YOLOv3	0.994	0.991	0.994	0.923	100	261	118.0	154.8
YOLOv3-SPP	0.990	0.992	0.994	0.925	94	269	120.0	155.4
YOLOv3-tiny	0.968	0.974	0.992	0.892	97	48	16.9	12.9
YOLOv5n	0.979	0.971	0.993	0.909	98	157	4.0	4.1
YOLOv5s	0.985	0.984	0.993	0.918	100	157	14.0	15.8
YOLOv5m	0.989	0.990	0.994	0.924	97	212	41.0	47.9
YOLOv5l	0.991	0.992	0.994	0.929	84	267	89.0	107.7
YOLOv5x	0.992	0.991	0.994	0.931	76	322	166.0	203.8
YOLOv8n	0.983	0.986	0.993	0.919	83	168	6.2	8.1
YOLOv8s	0.983	0.978	0.993	0.923	55	168	21.5	28.4
YOLOv8m	0.991	0.992	0.994	0.931	63	218	49.7	78.7
YOLOv8l	0.992	0.989	0.994	**0.933**	83	268	83.7	164.8
YOLOv8x	0.992	0.987	0.994	0.932	89	268	131.0	257.4

Note: The bold value refers to the best model with the highest mAP@0.5:0.95 value.

enhancements in performance. The figure illustrates that most architectures exhibit relatively stable mAP values toward the end of training, indicating convergence. Notably, YOLOv8n and YOLOv5n demonstrate slightly lower overall performance compared to others. The highest validation results obtained by the previously mentioned YOLO models are summarized in Table 5.

Stage II: According to the results obtained in Stage I and based on the common "winner-takes-all" strategy, YOLOv8l should be naturally selected as the "best" model for insulator mechanical damage detection, as it achieved the highest mAP@0.5:0.95 of 0.933 on the validation set. The runner-up YOLO architecture was YOLOv8x with an mAP@0.5:0.95 of 0.932. At the same time, the lightest

architectures in terms of the memory requirements and GFLOPs, including YOLOv5n and YOLOv8n, demonstrated an mAP@0.5:0.95 of 0.909 and 0.919, which is only about 2.55% and 1.51% less than the YOLOv8l. Thus, to select the "best" model, it is beneficial to assess the computational demands of the models in conjunction with their mAP@0.5:0.95 results. Therefore, complexity-related parameters, such as the number of layers, memory requirements for half-precision floating-point computations in MB (MBFB16), and the number of floating-point calculations in GFLOPs, are reported for all studied YOLO models in Table 5 in addition to corresponding mAP@0.5:0.95 scores.

To better visualize these results, Figure 5 shows the memory requirements of models plotted against the validation set mAP@0.5:0.95. From this figure, it can be observed that YOLOv8l, YOLOv8x, YOLOv5x, and YOLOv8m exhibit virtually the same performance in terms of mAP@0.5:0.95. It is also observed that the high-performance group (e.g., YOLOv8l, YOLOv8x, YOLOv5l, YOLOv5x, YOLOv8m) provides higher mAP, but requires more memory. In contrast, the lower-memory group (e.g., YOLOv5n, YOLOv5s, YOLOv8n, YOLOv3-tiny) requires significantly less memory at the cost of a slight decrease in mAP.

The performance–complexity consideration for model selection becomes important in practice since embedded devices such as UAVs tend to have limited computational capacities to store parameters of the trained models and perform prediction. As a result, we do not make a definite selection among the 13 YOLO architectures that prefer one vs. the others. Instead, we propose two models that can be used in practice, depending on performance–complexity requirements. In this regard, YOLOv8l architecture, which achieved a mAP@0.5:0.95 of 0.933, is the model of choice if the highest performance is desired, and YOLOv5n is the best model in terms of the lowest memory requirements and FLOPs at the expense of a slight reduction in performance. In particular, YOLOv5n requires only 4.0 MB memory for half-precision computation (~20.9 times less memory than YOLOv8l) and has 4.1 GFLOPs to classify the image (~40.2 times less than YOLOv8l). We propose to use YOLOv5n in hardware-restrained computational devices and to deploy YOLOv8l model in scenarios requiring optimal performance. In the following section, the performance of both models is assessed on the test set and the results are summarized.

4.2. Optimal Model Selection via Multicriteria Evaluation. In addition to identifying the two models in the previous section (YOLOv8l and YOLOv5n), we select an "optimal" model, further referred to as optimal-YOLO (OP-YOLO), where a multicriteria decision-making strategy is employed that balances detection performance with model efficiency. Seven key metrics were evaluated as follows: precision, recall, mAP@0.5, mAP@0.5:0.95, memory usage, GFLOPs, and layer count, normalized via Min-Max scaling to ensure comparability across different scales.

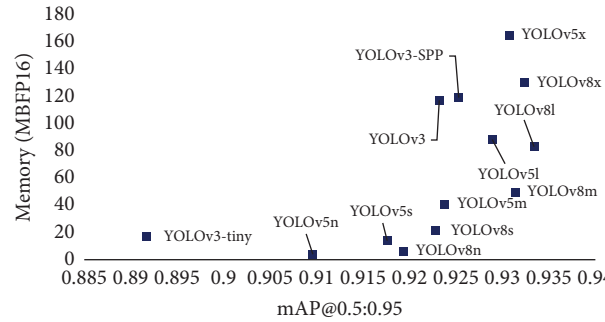


Figure 5: Memory requirements of each YOLO model versus observed performance in mAP@0.5:0.95.

Since there is no universally accepted standard for metric weighting in such multiobjective evaluations, each metric was weighted according to its practical importance: accuracy metrics (mAP@0.5:0.95: 25%, mAP@0.5: 20%, precision and recall: 15% each) and resource metrics (memory: 10%, GFLOPs: 10%, layers: 5%). The final composite score for each model was calculated as the weighted sum of normalized metrics.

We assigned the highest weights to mAP@0.5:0.95 (25%) and mAP@0.5 (20%) as they are standard accuracy metrics in object detection and offer a comprehensive view of precision–recall performance across thresholds. In UAV-based inspection, accuracy is often prioritized over efficiency, as recent studies have shown that moderate increases in computational load are acceptable for improved detection reliability [45, 46]. Precision and recall (15% each) remain crucial for minimizing false alarms and missed detections in industrial inspection, though they were slightly deprioritized relative to mAP, which provides a more integrated performance assessment. Efficiency metrics, memory usage and GFLOPs (10% each), were included to account for resource constraints in edge AI deployment. However, they were weighted lower, as modern UAVs can tolerate moderate trade-offs in power and latency for better detection quality. Model depth (5%) reflects architectural complexity and potential inference delay but was given minimal weight, since size and weight are secondary in UAVs capable of supporting heavier AI models [47].

This analysis revealed that the OP-YOLO model (YOLOv8m) offers the best overall trade-off, combining high detection accuracy (mAP@0.5:0.95 = 0.931) with moderate computational demands, reaching a maximum value of 0.868 for the composite score (see Table 6). In contrast, YOLOv3-tiny showed the lowest score of 0.262. Although some models, such as YOLOv5l and YOLOv5x, demonstrated high detection accuracy (mAP@0.5:0.95 = 0.929 and 0.931, respectively), they also incurred significantly higher resource demands, including memory consumption, GFLOPs, and layer complexity. When these models were evaluated within our multimetric scoring framework, which integrates both accuracy and efficiency considerations, their overall scores were lower than models with better performance–efficiency trade-offs. Figure 6 presents a bar chart comparing all models by their final

Table 6: Ranking of YOLO models based on a multicriteria decision-making framework.

Model	Score
YOLOv3	0.734
YOLOv3-SPP	0.715
YOLOv3-tiny	0.262
YOLOv5n	0.541
YOLOv5s	0.698
YOLOv5m	0.799
YOLOv5l	0.782
YOLOv5x	0.728
YOLOv8n	0.709
YOLOv8s	0.642
YOLOv8m	**0.868**
YOLOv8l	0.784
YOLOv8x	0.730

Note: The bold value refers to the model that reached the highest score used for the multicriteria decision-making framework.

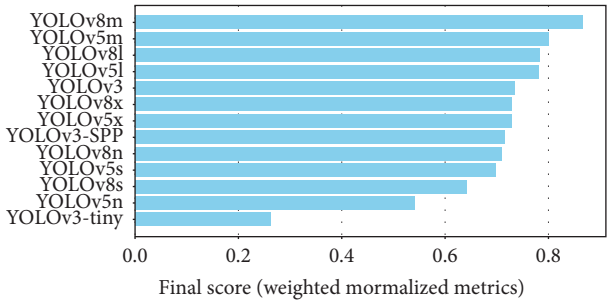


Figure 6: YOLO model comparison (accuracy + efficiency trade-off).

composite scores, offering a clear visualization of the performance–efficiency trade-offs.

To evaluate the robustness of our model selection, we further tested all candidate models under five different weighting scenarios: (i) our proposed balanced-performance-focused configuration, (ii) an accuracy-centric configuration (emphasizing mAP, precision, and recall), (iii) a real-time deployment configuration (emphasizing GFLOPs and memory usage), (iv) a lightweight deployment scenario (prioritizing model depth and size), and (v) a slightly modified balanced configuration. The weights used in each case are detailed in Supporting Table S3.

Across all configurations, the top-performing models, especially YOLOv8m, consistently ranked among the highest, indicating that our model selection is robust to reasonable variations in evaluation priorities. This consistency supports the reliability of YOLOv8m as a practical solution for accurate and efficient UAV-based inspection in diverse deployment scenarios.

5. Results

Following the model selection procedure detailed in the prior section, the selected YOLOv8l, YOLOv5n, and OP-YOLO architectures are evaluated on the test set. The performance evaluation is based on several metrics as presented in Table 7. According to the test-set results, the highest mAP of 0.995 is reported by YOLOv8l at IOU = 0.5, whereas both YOLOv5n and OP-YOLO reach the mAP of 0.994 at IOU = 0.5. Furthermore, YOLOv8l, YOLOv5n, and OP-YOLO achieved a mAP@0.5:0.95 of 0.919, 0.901, and 0.896, respectively. Figure 7 illustrates the estimated confusion matrices for the test dataset for YOLOv8l (a), YOLOV5n (b), and OP-YOLO (c) models.

It can be noted that the **YOLOv5n** model has a slight tendency to misclassify insulators with **cracks** as **normal insulators**, as seen by a background misclassification rate of 0.35 for this class, higher than in the other two models. Furthermore, **YOLOv5n** demonstrates a higher confusion rate for the **normal** class with the **background** (0.61), indicating less robustness in distinguishing insulators from their background compared to **YOLOv8l** (0.79) and **OP-YOLO** (0.77).

The **YOLOv8l** model performs best in identifying the **crack** class, achieving a perfect score (1.00), and shows slightly better separation between the **normal** and **background** classes compared to other models. **OP-YOLO**, while generally comparable to YOLOv8l, shows a slightly lower precision in the **crack** class (0.99) and a slightly better performance than YOLOv5n in minimizing confusion with the background class across all categories.

Overall, **YOLOv8l** and **OP-YOLO** offer stronger robustness and less confusion between critical fault classes and background, while **YOLOv5n**, despite high overall accuracy, appears slightly more susceptible to background interference and crack misclassification.

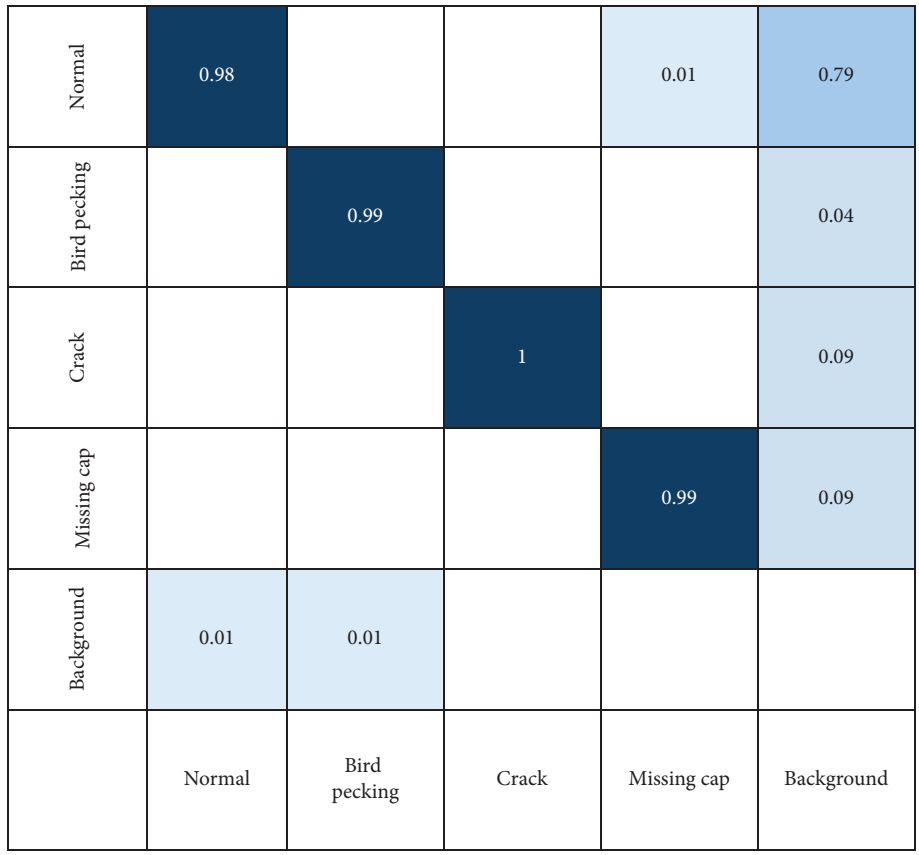
Figure 8 shows localization and classification performed by trained YOLOv8l on some original test insulator images. Images were collected in a laboratory under varying illumination and camera perspectives, with ambient conditions of around 20°C temperature and 50% relative humidity. As can be seen, the trained model effectively generates bounding boxes regardless of the type and size of the insulator under the test. The model also reports predicted fault type and corresponding confidence level, which is identified as the product of the objectness probability p_O and class probability p_C.

Figure 9 presents sample images from the original test set with visually complex scenes. These examples were selected to highlight challenging conditions such as cluttered backgrounds, the presence of multiple insulators, low lighting, and long-distance imaging. Figure 9 demonstrates the confident performance of the YOLOv8l model and its robustness to additional background elements and multiple insulators in one image.

5.1. Performance on Original UAV Imagery. The trained YOLOv8l model was also examined using UAV-captured insulator images. The database was collected using DJI Mavic II drone as depicted in Figure 10. The UAV was equipped with a 12-megapixel camera with 2x optical zoom capabilities.

Table 7: Test result comparison.

Metric	YOLOv5n	YOLOv8l	OP-YOLO
mAP@0.5	0.994	0.995	0.994
mAP@0.5:0.95	0.901	0.919	0.896
Average precision (IOU = 0.5)	0.984	0.995	0.992
Average recall (IOU = 0.5)	0.986	0.996	0.989



(a)

Figure 7: Continued.

Normal	0.96			0.01	0.61
Bird pecking		0.99			0.01
Crack	0.03		1		0.35
Missing cap				0.99	0.03
Background	0.01	0.01			
	Normal	Bird pecking	Crack	Missing cap	Background

(b)

Normal	0.98		0.01		0.77
Bird pecking		0.99			0.02
Crack			0.99		0.20
Missing cap				1	0.02
Background	0.01	0.01			
	Normal	Bird pecking	Crack	Missing cap	Background

(c)

Figure 7: Confusion matrix for the test set at IOU = 0.5. (a) YOLOv8l. (b) YOLOv5n. (c) OP-YOLO.

Figure 8: Detection and classification of insulator mechanical damage using YOLOv8l on the test set.

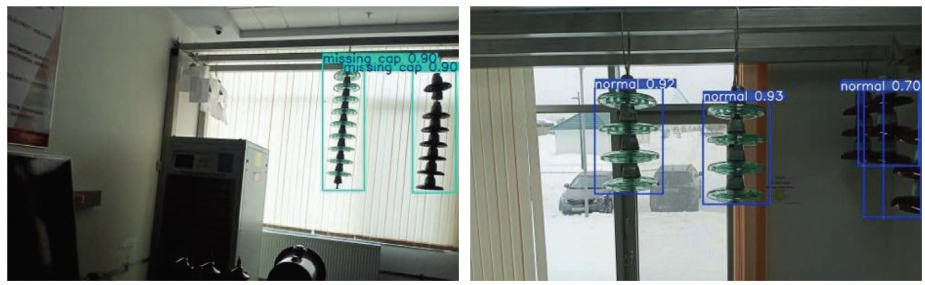


Figure 9: YOLOv8l model performance at complex scenery.

Figure 10: UAV-based insulator image collection.

To examine the worst-case scenario and evaluate the performance of the model, a very low-quality photo setting was selected, and the original image size of the UAV camera was set to 1280 × 720 pixels (~1-megapixel). Detection and classification results of the YOLOv8l model using UAV-captured images are illustrated in Figure 11. It is observed that the YOLOv8l image detection algorithm works on single object insulator images, as well as multiple object images and different insulator conditions. Notably, YOLOv8l remains effective even when processing low-quality images, making it suitable for scenarios with

Figure 11: Test using UAV-captured images.

limited camera capabilities. This high-performance outcome became possible since the aforementioned YOLO architectures were trained with an input image resolution of 640×640 pixels (i.e., all the trained and tested images were resized to 640×640 pixels).

The obtained results show that the trained YOLOv8l model successfully identifies the bird pecking damage in polymer insulators, cracks on ceramic insulators, and missing caps in string insulators. In summary, it should be highlighted that although YOLOv8l exhibited the highest classification performance, the significantly lower memory and computational requirements of YOLOv5n will make it an attractive choice to be deployed over resource-limited embedded devices.

5.2. Robustness to Noise and Occlusion. To evaluate the robustness of the selected YOLO models (YOLOv8l, YOLOv5n, and OP-YOLO) under challenging visual conditions, two augmented versions of the original test set were created: a noise-augmented test set and an occlusion-augmented test set. These augmentations simulate common real-world issues encountered during UAV inspections, such as sensor noise and physical obstructions.

The noise-augmented test set was generated by adding Gaussian noise [48] with a standard deviation of 5 and zero mean to each image in the original test set. This process introduces realistic noise artifacts that can occur due to low-light conditions or sensor imperfections while maintaining the original object locations and corresponding annotations. The labels from the original test set were preserved without modification.

For the occlusion-augmented test set, random black rectangles were overlaid on each image to mimic partial occlusions caused by objects such as vegetation, poles, or dust [43]. Specifically, three black patches of varying size and random positions were applied to each image, simulating realistic scenarios where insulator surfaces might be partially obstructed. As with the noise set, the original annotations were retained without changes. Sample noise-augmented and occlusion-augmented test set images are given in Figure 12.

These augmented datasets were stored separately and used to re-evaluate the selected YOLO models (YOLOv8l, YOLOv5n, and OP-YOLO) (see Tables 8 and 9). By comparing the model performance on noise and occlusion test sets with their baseline results on original test data, we were able to analyze the robustness of each model to image degradation and partial obstruction commonly found in real deployment environments.

When comparing the results from Table 8 (occlusion-augmented test set) with those from Table 7 (original test set), all three models experienced a noticeable drop in performance due to simulated occlusions. YOLOv8l maintained the highest robustness, with only a ~4.8% decrease in mAP@0.5 and ~17.6% drop in mAP@0.5:0.95, indicating its strong ability to handle partial obstructions. OP-YOLO also demonstrated solid robustness with relatively small performance degradation. YOLOv5n, while still effective, showed a slightly larger decline across all metrics, particularly in average recall. These results suggest that YOLOv8l and OP-YOLO are better suited for deployment in visually complex environments where occlusions are common.

By contrasting the results presented in Table 9 (noise-augmented test set) with those in Table 7 (original test set), all three YOLO models showed a considerable decline in performance, indicating their sensitivity to image noise. YOLOv8l, which achieved the highest accuracy on clean data, experienced a ~48.8% drop in mAP@0.5 and over 50% decrease in mAP@0.5:0.95. YOLOv5n also had a sharp decrease across all metrics, particularly in average recall. Notably, OP-YOLO maintained comparatively higher scores in this noisy setting, with the smallest drop in average precision and recall. The significant performance degradation across all models highlights a key limitation in their current training setup. To improve resilience against sensor imperfections or low-light conditions, future work should focus on incorporating noise-augmented data into the training process.

6. Discussion

Having a variety of existing YOLO architectures makes selecting the best application-dependent YOLO model challenging for practical object detection and classification tasks. At the same time, selecting the best architecture merely based on the performance, without considering the model complexity, limits its deployment on resource-limited

(a) (b)

Figure 12: Sample noise-augmented and occlusion-augmented test-set images. (a) Occlusion-augmented test-set images. (b) Noise-augmented test-set images.

embedded devices. In this study, three architectures of YOLOv3, five variations of YOLOv5, and five variations of YOLOv8 are compared in terms of model complexity–performance trade-off for the identification of insulator faults and mechanical damage. Specifically, we examined YOLOv3 [29], YOLOv3-SPP [34], YOLOv3-tiny [37], YOLOv5n, YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x [36], YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l, and YOLOv8x architectures [31]. These single-stage detection models are widely known for their speed and computational efficiency. The results of the previous section show that although YOLOv8l reported higher mAP@0.5:0.95, YOLOv5n requires ~20.9 times less memory and ~40.2 times less FLOPs for image classification, resulting in a slight ~2.6% reduction in mAP@0.5:0.95, making YOLOv5n a favorable option for deployment on resource-limited platforms such as UAVs. Also, YOLOv5n outperforms the OP-YOLO model, still requiring ~12 times less memory and ~19 times less FLOPs.

6.1. Benchmarking Against Faster R-CNN. To ensure a fair comparison, faster R-CNN models (ResNet50, Inception-ResNetV2, and ResNet152V1) [49–51] were trained using the same training, validation, and testing configurations as the YOLO-based models (see Table 10). This includes identical image resolution (640×640), number of training epochs (100), and batch size (16), using Adam as the optimizer [44] (initial learning rate of 0.01) and applying a weight decay of 0.0005 for regularization. No early stopping or dropout was used. The same preprocessing steps and targeted data augmentation techniques were applied, maintaining consistency across all evaluated models. The training was conducted on the same GPU environment described in Section 4 to ensure consistent hardware usage and reproducibility.

Comparison of Tables 7 and 10 shows that the identified YOLO-based architectures consistently perform better than all variations of Faster R-CNN models. In particular, YOLOv8l, YOLOv5n, and OP-YOLO achieved 0.994, 0.993,

Table 8: Test result comparison with the occlusion-augmented test set.

Metric	YOLOv5n	YOLOv8l	OP-YOLO
mAP@0.5	0.913	0.947	0.935
mAP@0.5:0.95	0.719	0.757	0.748
Average precision (IOU = 0.5)	0.892	0.948	0.939
Average recall (IOU = 0.5)	0.856	0.889	0.873

Table 9: Test result comparison with the noise-augmented test set.

Metric	YOLOv5n	YOLOv8l	OP-YOLO
mAP@0.5	0.512	0.509	0.548
mAP@0.5:0.95	0.402	0.452	0.492
Average precision (IOU = 0.5)	0.384	0.452	0.547
Average recall (IOU = 0.5)	0.486	0.507	0.582

Table 10: Comparative test results with faster R-CNN models.

Metric	Faster R-CNN		
	ResNet50	InceptionResNetV2	ResNet152V1
mAP@0.5	0.975	0.869	0.890
Average precision (IOU = 0.5)	0.821	0.667	0.890
Average recall (IOU = 0.5)	0.881	0.790	0.721

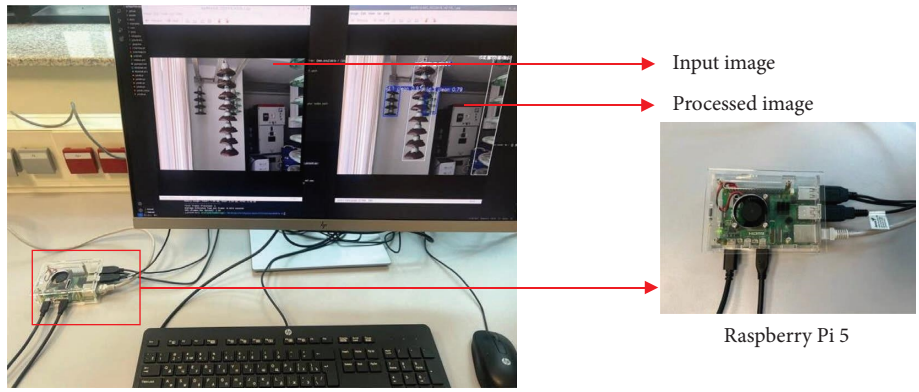


Figure 13: Raspberry Pi 5 setup with cooling fan, power supply, camera, monitor, keyboard, and mouse connections.

and 0.994 mAP@0.5, respectively, while the best faster R-CNN (ResNet50) achieved a maximum mAP@0.5 of 0.975. Regarding the average precision metric, YOLOv8l, YOLOv5n, and OP-YOLO achieved 0.992, 0.979, and 0.991, respectively, while the highest average precision among the faster R-CNN models was only 0.89 (ResNet152V1). Finally, regarding the average recall at IOU = 0.5, YOLOv8l, YOLOv5n, and OP-YOLO achieved 0.989, 0.971, and 0.992, respectively, while the best faster R-CNN (ResNet50) achieved 0.881.

6.2. Edge Deployment Feasibility. In real-world scenarios, both memory usage and prediction accuracy are key factors. For instance, UAVs typically have hardware limitations that restrict their ability to store large pretrained model weights and carry out predictions efficiently. As outlined in the model training and selection section, no final decision was made between the YOLOv8l, YOLOv5n, and OP-YOLO models. However, considering a multicriteria decision-making strategy to balance between model complexity and accuracy, OP-YOLO appears to be the most favorable choice.

This section explores deploying the selected models on resource-limited edge devices for real-time classification of high-voltage insulator mechanical fault detection. While modern GPUs handle deep learning tasks well, devices including Raspberry Pi [52] face challenges such as limited memory, power, and battery life. YOLOv8l, YOLOv5n, and OP-YOLO were tested on a Raspberry Pi 5 [52] Model B (4 GB RAM, 5 V/4A), chosen for its cost-effectiveness and suitability for edge computing. The setup included a 16 GB MicroSD card, adapters, and a cooling fan (see Figure 13), with the software environment based on Raspberry Pi OS Bookworm 12 and PyTorch 2.5.1.

Table 11 presents the inference performance of YOLOv8l, YOLOv5n, and OP-YOLO on video data using Raspberry Pi 5. YOLOv5n delivers the fastest inference

Table 11: Performance of YOLOv8l, YOLOv5n, and OP-YOLO models.

	Inference (s)	FPS	Memory (GB)
YOLOv8l	3.6	0.2	0.275
YOLOv5n	0.2	4.0	0.337
OP-YOLO	1.8	0.6	0.245

(0.2 s) and highest frame rate (4 FPS), but with slightly higher memory usage (0.337 GB). OP-YOLO offers a well-balanced trade-off, achieving moderate speed (1.8 s), reasonable frame rate (0.6 FPS), and the lowest memory consumption (0.245 GB). In contrast, YOLOv8l demonstrates the slowest performance (3.6 s and 0.2 FPS) with higher memory usage (0.275 GB), making it suboptimal for real-time edge deployment. Overall, OP-YOLO emerges as a more efficient and practical choice for resource-constrained platforms such as Raspberry Pi 5.

7. Conclusion

Outdoor electrical insulators are essential components of the power grid, serving both as structural supports and as electrical insulators for power lines. To facilitate their uninterrupted operation, performing regular inspections of insulators is crucial. This study provided an in-depth discussion of various insulator diagnostic techniques. The recent development of deep learning and computer vision facilitates the implementation of automated UAV-based insulator condition diagnostic techniques, which can replace the costly old-style manual inspection. In particular, computer vision-based aerial image analysis helps reduce inspection time and prevent subjective assessments by on-site maintenance operators. Practically, widely accepted YOLOv3, YOLOv5, and YOLOv8 architectures were trained and evaluated in the classification of commonly encountered insulator mechanical defects, including bird pecking, cracking, and missing cap faults.

A two-stage model selection procedure was deployed to identify the best YOLO model based on the prediction accuracy. Based on the performance of the validation set, YOLOv8l outperformed other models and achieved a mAP@0.5:0.95 of 0.933. Moreover, the "lightest" architecture, including YOLOv5n, required ~20.9 times less memory and ~40.2 times fewer FLOPs than YOLOv8l and achieved a mAP@0.5:0.95 of 0.909 on the validation set. Alongside YOLOv8l and YOLOv5n, an "optimal" model, OP-YOLO, was selected using a multicriteria decision-making approach that balances detection accuracy and computational efficiency. OP-YOLO model offers the best overall trade-off, with high detection accuracy (mAP@0.5:0.95 of 0.931) and moderate computational demands, reaching a maximum value of 0.868 for the defined composite score.

On the test set, YOLOv8l, YOLOv5n, and OP-YOLO achieved an mAP@0.5:0.95 of 0.919, 0.901, and 0.896 and a remarkably high mAP@0.5 of 0.995, 0.994, and 0.994, respectively. Based on the result, it was recommended to deploy YOLOv8l when the optimal performance is a priority, and YOLOv5n in resource-constrained hardware, such as embedded devices used in UAVs. Also, it was recommended to deploy the OP-YOLO model, selected based on a multicriteria decision-making framework, when the trade-off between performance and computational efficiency is desired.

One key limitation of the presented approach is the significant drop in performance when evaluated on noise-augmented data, indicating limited robustness to sensor imperfections and low-light conditions. This degradation suggests that the current training setup lacks sufficient variability to generalize across real-world challenges. To overcome this limitation, future research should aim to enlarge the training dataset by incorporating noise-augmented and occlusion-augmented data, thereby improving model robustness. Additionally, incorporating multimodal inputs, such as IR or thermal imaging, could enhance detection capabilities under extreme conditions such as poor visibility or adverse weather, enabling the development of more robust fault detection systems in practical UAV deployments.

Despite these limitations, the results demonstrate the high effectiveness of YOLOv8l, YOLOv5n, and OP-YOLO models in insulator damage classification from various complex insulator scenes, such as multiple objects in the frame, various camera angles and distances, different illumination levels, and cluttered backgrounds. Furthermore, YOLOv8l, YOLOv5n, and OP-YOLO showed a marked advantage in terms of performance over the leading two-stage models (faster R-CNN with different backbone architectures). Combining the proposed aerial image-based inspection method with already existing diagnostic techniques, such as insulator flashover, porosity, leakage current, impulse withstand, and dielectric dissipation factor measurement, can improve the prediction performance and reliability of the insulator damage diagnosis.

Data Availability Statement

The data collected during the current study are available from the corresponding author upon reasonable request.

Conflicts of Interest

The authors declare no conflicts of interest.

Funding

This work was supported in part by the Collaborative Research Project (CRP), Nazarbayev University, under Grant 211123CRP1604, and the Faculty Development Competitive Research Grant (FDCRG), Nazarbayev University, under Grant 201223FD8811.

Acknowledgments

This work was supported in part by the Collaborative Research Project (CRP), Nazarbayev University, under Grant 211123CRP1604, and the Faculty Development Competitive Research Grant (FDCRG), Nazarbayev University, under Grant 201223FD8811.

References

[1] A. Y. Alqudsi, R. A. Ghunem, and E. David, "The Viability of the Filler Barrier Effect During the DC Dry-Band Arcing on Silicone Rubber," *IEEE Transactions on Dielectrics and Electrical Insulation* 29, no. 5 (October 2022): 1873–1881, https://doi.org/10.1109/TDEI.2022.3198754.

[2] J. Cao, Y. Du, Y. Ding, et al., "Lightning Protection With a Differentiated Configuration of Arresters in a Distribution Network," *IEEE Transactions on Power Delivery* 38, no. 1 (February 2023): 409–419, https://doi.org/10.1109/TPWRD.2022.3192482.

[3] A. Andreotti, A. Pierno, and V. A. Rakov, "An Analytical Approach to Calculation of Lightning Induced Voltages on Overhead Lines in Case of Lossy Ground—Part II: Comparison with Other Models," *IEEE Transactions on Power Delivery* 28, no. 2 (April 2013): 1224–1230, https://doi.org/10.1109/TPWRD.2013.2241085.

[4] X. Lu, W. Quan, S. Gao, et al., "An Outdoor Support Insulator Surface Defects Segmentation Approach via Image Adversarial Reconstruction in High-Speed Railway Traction Substation," *IEEE Transactions on Instrumentation and Measurement* 71 (2022): 1–19, https://doi.org/10.1109/TIM.2022.3211558.

[5] A. Ersoy, F. Atalar, and H. Hiziroğlu, "A Study on Particle Size Effect of Polyurethane-Mica Composites," *IEEE Access* 12 (2024): 679–688, https://doi.org/10.1109/access.2023.3347368.

[6] A. Ersoy, F. Atalar, and A. Aydogan, "Investigation of Novel Solid Dielectric Material for Transformer Windings," *Polymers* 15, no. 24 (2023): 1–14.

[7] American National Standard, *Test Methods for Electrical Power Insulators* (National Electrical Manufacturers Association, July 2013).

[8] D. Ivanov, M. Sadykov, D. Yaroslavsky, A. Golenishchev-Kutuzov, and T. Galieva, "Non-Contact Methods for High-Voltage Insulation Equipment Diagnosis During Operation," *Energies* 14, no. 18 (September 2021): 5670, https://doi.org/10.3390/en14185670.

[9] C. Yuan, C. Xie, L. Li, F. Zhang, and S. M. Gubanski, "Ultrasonic Phased Array Detection of Internal Defects in Composite Insulators," *IEEE Transactions on Dielectrics and Electrical Insulation* 23, no. 1 (2016): 525–531, https://doi.org/10.1109/tdei.2015.005225.

[10] N. Davari, G. Akbarizadeh, and E. Mashhour, "Intelligent Diagnosis of Incipient Fault in Power Distribution Lines Based on Corona Detection in UV-Visible Videos," *IEEE Transactions on Power Delivery* 36, no. 6 (2021): 3640–3648, https://doi.org/10.1109/tpwrd.2020.3046161.

[11] H. Zheng, Y. Liu, Y. Sun, et al., "Arbitrary-Oriented Detection of Insulators in Thermal Imagery via Rotation Region Network," *IEEE Transactions on Industrial Informatics* 18, no. 8 (2022): 5242–5252, https://doi.org/10.1109/tii.2021.3123107.

[12] S. Anjum, S. Jayaram, A. El-Hag, and A. N. Jahromi, "Detection and Classification of Defects in Ceramic Insulators Using RF Antenna," *IEEE Transactions on Dielectrics and Electrical Insulation* 24, no. 1 (2017): 183–190, https://doi.org/10.1109/tdei.2016.005867.

[13] L. Liu, H. Mei, C. Guo, Y. Tu, and L. Wang, "Pixel-Level Classification of Pollution Severity on Insulators Using Photothermal Radiometry and Multiclass Semisupervised Support Vector Machine," *IEEE Transactions on Industrial Informatics* 17, no. 1 (2021): 441–449, https://doi.org/10.1109/tii.2020.2984642.

[14] L. Yang, J. Fan, Y. Liu, E. Li, J. Peng, and Z. Liang, "A Review on State-of-the-Art Power Line Inspection Techniques," *IEEE Transactions on Instrumentation and Measurement* 69, no. 12 (2020): 9350–9365, https://doi.org/10.1109/tim.2020.3031194.

[15] V. N. Nguyen, R. Jenssen, and D. Roverso, "Intelligent Monitoring and Inspection of Power Line Components Powered by Uavs and Deep Learning," *IEEE Power and Energy Technology Systems Journal* 6, no. 1 (2019): 11–21, https://doi.org/10.1109/jpets.2018.2881429.

[16] Z. D. Zhang, B. Zhang, Z. C. Lan, et al., "FINet: An Insulator Dataset and Detection Benchmark Based on Synthetic Fog and Improved YOLOv5," *IEEE Transactions on Instrumentation and Measurement* 71 (2022): 1–8, https://doi.org/10.1109/TIM.2022.3194909.

[17] M. Carranza-García, J. Torres-Mateo, P. Lara-Benítez, and J. García-Gutiérrez, "On the Performance of One-Stage and Two-Stage Object Detectors in Autonomous Vehicles Using Camera Data," *Remote Sensing*, 13, no. 1, 89, https://doi.org/10.3390/rs13010089.

[18] X. Hu and Y. N. Li, "Hybrid Model Insulator Fault Detection Based on Faster R-CNN and U-Net," *Video Engineering* 45, no. 5 (January 2021): 125–130, https://doi.org/10.16280/j.videoe.2021.05.034.

[19] Q. Wang and B. S. Yi, "Insulator Defect Recognition in Aerial Images Based on Gaussian YOLOv3," *Laser Optoelectron. Prog.* 58, no. 12 (June 2021): 254–260, https://doi.org/10.3788/LOP202158.1210022.

[20] J. Han, Z. Yang, H. Xu, et al., "Search Like an Eagle: A Cascaded Model for Insulator Missing Faults Detection in Aerial Images," *Energies* 13, no. 3 (February 2020): 713, https://doi.org/10.3390/en13030713.

[21] C. Sampedro, J. Rodriguez-Vazquez, A. Rodriguez-Ramos, A. Carrio, and P. Campoy, "Deep Learning-Based System for Automatic Recognition and Diagnosis of Electrical Insulator Strings," *IEEE Access* 7 (2019): 101283–101308, https://doi.org/10.1109/access.2019.2931144.

[22] Z. Xuan, J. Ding, and J. Mao, "Intelligent Identification Method of Insulator Defects Based on CenterMask," *IEEE Access* 10 (2022): 59772–59781, https://doi.org/10.1109/ACCESS.2022.3179975.

[23] B. Wei, Z. Xie, Y. Liu, K. Wen, F. Deng, and P. Zhang, "Online Monitoring Method for Insulator Self-Explosion Based on Edge Computing and Deep Learning," *CSEE Journal of Power and Energy Systems*, https://doi.org/10.17775/CSEEJPES.2020.05910.

[24] X. Miao, X. Liu, J. Chen, S. Zhuang, J. Fan, and H. Jiang, "Insulator Detection in Aerial Images for Transmission Line Inspection Using Single Shot Multibox Detector," *IEEE Access* 7 (2019): 9945–9956, https://doi.org/10.1109/ACCESS.2019.2891123.

[25] W. Zan, C. Dong, J. Zhao, F. Hao, D. Lei, and Z. Zhang, "Defect Identification of Power Line Insulator Based on an Improved yolov4-Tiny Algorithm," in *2022 5th International Conference on Renewable Energy and Power Engineering (REPE)* (2022), 35–39, https://doi.org/10.1109/REPE55559.2022.9949418.

[26] Z. Feng, L. Guo, D. Huang, and R. Li, "Electrical Insulator Defects Detection Method Based on YOLOv5," in *2021 IEEE 10th Data Driven Control and Learning Systems Conference (DDCLS)* (2021), 979–984, https://doi.org/10.1109/DDCLS52934.2021.9455519.

[27] A. Serikbay, M. Bagheri, A. Zollanvari, and B. T. Phung, "Accurate Surface Condition Classification of High Voltage Insulators Based on Deep Convolutional Neural Networks," *IEEE Transactions on Dielectrics and Electrical Insulation* 28,

no. 6 (December 2021): 2126–2133, https://doi.org/10.1109/TDEI.2021.009648.

[28] D. Mussina, A. Irmanova, P. K. Jamwal, and M. Bagheri, "Multi-Modal Data Fusion Using Deep Neural Network for Condition Monitoring of High Voltage Insulator," *IEEE Access* 8 (2020): 184486–184496, https://doi.org/10.1109/ACCESS.2020.3027825.

[29] S. Panigrahy and S. Karmakar, "Real-Time Condition Monitoring of Transmission Line Insulators Using the YOLO Object Detection Model With a UAV," *IEEE Transactions on Instrumentation and Measurement* 73 (2024): 1–9, https://doi.org/10.1109/TIM.2024.3381693.

[30] S. He, C. Zang, J. He, et al., "Detecting Contaminated Insulators by UV Imaging Technology" (2010).

[31] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," (2018), https://arxiv.org/abs/1804.02767.

[32] Ultralytics, "Yolov5 Focus Layer Discussion #3181·Ultralytics/yolov5," *GitHub* https://github.com/ultralytics/yolov5/discussions/3181m1.

[33] Ultralytics, "Ultralytics/Ultralytics: New-yolov8 in Pytorch > Onnx > Coreml > TFLite," *GitHub* https://github.com/ultralytics/ultralytics.

[34] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," *arXiv* (2015).

[35] X. Wang, S. Wang, J. Cao, and Y. Wang, "Data-Driven Based Tiny-YOLOv3 Method for Front Vehicle Detection Inducing SPP-Net," *IEEE Access* 8 (2020): 110227–110236, https://doi.org/10.1109/ACCESS.2020.3001279.

[36] X. Zhang, Y. Gao, H. Wang, and Q. Wang, "Improve YOLOv3 Using Dilated Spatial Pyramid Module for Multi-Scale Object Detection," *International Journal of Advanced Robotic Systems* 17, no. 4 (2020): 1729881420936062, https://doi.org/10.1177/1729881420936062.

[37] H. Zhang, M. Tian, G. Shao, J. Cheng, and J. Liu, "Target Detection of Forward-Looking Sonar Image Based on Improved YOLOv5," *IEEE Access* 10 (2022): 18023–18034, https://doi.org/10.1109/ACCESS.2022.3150339.

[38] H. K. Jung and G.-S. Choi, "Improved Yolov5: Efficient Object Detection Using Drone Images Under Various Conditions," *Applied Sciences* 12, no. 14 (2022): 7255, https://doi.org/10.3390/app12147255.

[39] W. He, Z. Huang, Z. Wei, C. Li, and B. Guo, "TF-YOLO: An Improved Incremental Network for Real-Time Object Detection," *Applied Sciences* 9, no. 16 (2019): 3225, https://doi.org/10.3390/app9163225.

[40] G. Han, M. He, M. Gao, J. Yu, K. Liu, and L. Qin, "Insulator Breakage Detection Based on Improved YOLOv5," *Sustainability* 14, no. 10 (May 2022): 6066, https://doi.org/10.3390/su14106066.

[41] N. Zhang, J. Su, Y. Zhao, and H. Chen, "Insulator-YOLO: Transmission Line Insulator Risk Identification Based on Improved YOLOv5," *Processes* 12, no. 11 (November 2024): 2552, https://doi.org/10.3390/pr12112552.

[42] Y. Chen, H. Liu, J. Chen, J. Hu, and E. Zheng, "Insu-YOLO: An Insulator Defect Detection Algorithm Based on Multiscale Feature Fusion," *Electronics* 12, no. 15 (July 2023): 3210, https://doi.org/10.3390/electronics12153210.

[43] C. Shorten and T. Khoshgoftaar, "A Survey on Image Data Augmentation for Deep Learning," *Journal of Big Data* 6, no. 1 (2019): 60, https://doi.org/10.1186/s40537-019-0197-0.

[44] D. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *3rd International Conference on Learning Representations, ICLR 2015-Conference Track Proceedings* (2014), https://arxiv.org/pdf/1412.6980.

[45] T. Y. Lin, M. Maire, S. Belongie, et al., "Microsoft COCO: Common Objects in Context," in *Proceeding of the European Conference on Computer Vision (ECCV)* (Zurich, Switzerland, 2014), 740–755, https://doi.org/10.1007/978-3-319-10602-1_48.

[46] G. Jocher, A. Chaurasia, and J. Qiu, "YOLOv5," *Ultralytics, GitHub repository* (2020): https://github.com/ultralytics/yolov5.

[47] M. Canpolat Şahin and A. Kolukısa Tarhan, "Evaluation and Selection of Hardware and AI Models for Edge Applications: A Method and a Case Study on Uavs," *Applied Sciences (Basel)* 15, no. 3 (January 2025): 1026, https://doi.org/10.3390/app15031026.

[48] D. Hendrycks and T. Dietterich, "Benchmarking Neural Network Robustness to Common Corruptions and Perturbations," in *Proceedings of the International Conference on Learning Representations (ICLR)* (2019).

[49] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection With Region Proposal Networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39, no. 6 (June 2017): 1137–1149, https://doi.org/10.1109/TPAMI.2016.2577031.

[50] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (Las Vegas, NV, 2016), 770–778, https://doi.org/10.1109/CVPR.2016.90.

[51] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning" (2016).

[52] G. Anand and A. K. Kumawat, "Object Detection and Position Tracking in Real Time Using Raspberry Pi," *Materials Today: Proceedings* 47, no. 11 (2021): 3221–3226, https://doi.org/10.1016/j.matpr.2021.06.437.