Journal of Theoretical and Applied Information Technology
15st July 2026. Vol.104. No.13
© Little Lion Scientific
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

# ADAPTIVE POST-PROCESSING FUSION FOR MULTI-MODEL YOLO-BASED INSECT DETECTION

**AKARID ABDERRAHIM, AIT EL ASRI SMAIL, EL ADIB SAMIR, RAISSOUNI NAOUFAL**

Remote Sensing Systems and Telecommunications, National School for Applied Sciences, Abdelmalek Essaadi University UAE/U01 ENSATe, Tetuan, Morocco

E-mail: a.akarid@uae.ac.ma , smail.aitelasri@etu.uae.ac.ma , seladib@uae.ac.ma , naoufal.raissouni.ensa@gmail.com

**ABSTRACT**

In recent years, object detection has made tremendous progress, yet the best current detectors still fall short of achieving ultimate performance. Insect detection is vital for precision agriculture but remains challenging due to species similarity and complex environmental conditions. This study introduces a generic adaptive post-processing fusion framework that dynamically combines predictions from multiple heterogeneous object detectors using Non-Maximum Suppression (NMS) and Weighted Box Fusion (WBF). The proposed approach was evaluated on the IP102 dataset, focusing on 10 pest species. Individual You Only Look Once (YOLO) models (YOLOv5nu, YOLOv8n and YOLOv10n) attained accuracies of 95.73%, 95.37%, and 95.37% with Mean Average Precision (mAP) at Intersection over Union (IoU) 0.5 scores of 92.91%, 91.32%, and 92.57%, respectively. The adaptive fusion approach outperformed all single models, reaching a mAP at IoU 0.5 of 97.22% and an accuracy of 98.58%, representing a gain of approximately 5% and 2.85%, respectively, and surpassing fixed-weight NMS and WBF baselines by approximately 6% in mAP. These findings demonstrate that combining YOLO models through adaptive fusion significantly enhances insect detection performance, providing a reliable building block that can be integrated into future pest monitoring pipelines for smart agriculture systems.

**Keywords:** *Object Detection, YOLO, Weighted Box Fusion, Adaptive Ensemble Learning, Insect Pest Detection*

## 1. INTRODUCTION

Insect recognition is not considered an ordinary visual task. This is an apparently easy operation, but is based on essential scientific disciplines, such as entomology, entomological taxonomy, insect ecology, biodiversity, and conservation sciences. These fields provide the theoretical and methodological foundations necessary for accurate species identification, understanding species' interactions with their environment, as well as identifying the impact that species exert on environmental systems as a whole and on agricultural systems specifically [1]. Additionally, in entomology and agricultural science, insect recognition is a fundamental tool in pest control, ensuring the early identification of insect infestations, minimizing crop damage, identifying invasive or exotic insects, and contributing to research developments in entomology, including the development of modern taxonomic databases. Insect species recognition also plays a fundamental role in food security and agricultural economic sustainability, which requires scientific knowledge within agricultural sectors [2]. In addition, the process of insect species identification differs significantly from common object and animal classification tasks due to certain attributes [3]-[4].

Insect recognition, identification, and classification are inherently challenging tasks in entomological science, primarily due to visual similarities between species, which make it difficult to differentiate between species. To respond to these issues, numerous approaches have been proposed for identifying and classifying insects [5], contributing to addressing these challenges and providing deeper insights. Indeed, in many fields of research, such as entomological sciences, the environment, and agricultural engineering, computer vision techniques play a crucial role [6]-[7]-[8].

This study proposes a generic adaptive post-processing fusion framework designed to improve the accuracy and robustness of any multi-detector object detection system. The scope of this work is confined to post-processing fusion applied to three complementary nano YOLO versions evaluated on ten pest classes from the IP102 dataset [27], a large-scale benchmark widely used for fine-grained pest detection.

The new knowledge created by this work directly addresses a critical and underexplored limitation in existing fusion methods: standard post-processing techniques such as Non-Maximum Suppression (NMS) and Weighted Box Fusion (WBF) assign fixed, uniform weights to all contributing models regardless of their actual prediction reliability on any given image. This static treatment fails to account for per-image variability in detector confidence, particularly in challenging scenarios involving visually similar species, cluttered agricultural backgrounds, and multi-scale insect instances. To overcome this limitation, this work introduces: an image-adaptive confidence-based weighting mechanism that dynamically adjusts each model's contribution at inference time based on its mean prediction confidence, and a consensus-driven confidence-boosting strategy that rewards multi-model agreement to reduce false negatives, both applied entirely without additional model training.

Although individual models (YOLOv5nu, YOLOv8n, and YOLOv10n) provide strong performance, their predictions often vary across species and image conditions. To address these inconsistencies, we investigate both NMS and WBF for model ensembling and introduce an adaptive fusion strategy that assigns image-specific weights based on detector confidence. We expect that dynamically assigning image-specific confidence-based weights to multiple YOLO detectors during post-processing fusion will yield superior detection performance compared to both individual single-model detectors and fixed-weight ensemble baselines, without requiring any additional model training.

The paper is organized as follows: section 2 presents the materials and methods and describes the proposed approach; section 3 presents the numerical results and discussion; section 4 concludes the paper and suggests future work and directions.

## 2. RELATED WORK

Deep learning (DL) has been extensively used for insect detection in recent years, with features including image classification and object detection. Building on foundational neural network models [9]-[10], convolutional neural networks (CNNs) [11] emerged as the dominant architecture. The rise of DL continued with variants such as DNN, LSTM, and RNN, further expanding DL applications [12]–[13]. This growth is largely a result of large training datasets like ImageNet [14], coupled with advances in high-performance computing systems such as graphics processing units (GPU) clusters, which have enabled more efficient model training.

Simultaneously, various network architectures have been extensively researched to enhance performance, including AlexNet [15], GoogLeNet [16], and visual geometry group (VGG) [17]. These innovations have enabled the training of large-scale CNNs on extensive datasets such as MS COCO [18] and pattern analysis statistical modelling and computational learning-visual object classes (PASCAL VOC) [19]. Multi-stage detectors like R-CNN and R-FCN use two-stage pipelines to extract regions of interest before classifying and refining object localization.

Ding and Taylor [20] proposed a sliding window method that divides images into overlapping windows to evaluate each for the presence of moths. This approach scans images at multiple scales to detect objects of varying sizes but can be computationally demanding and less accurate compared to modern DL methods. Recent advancements in DL-based object detection have been particularly beneficial for, focusing on disease and pest detection. Significant changes in object detection methods have replaced traditional sliding window approaches with region proposals. Recent progress in object detection has introduced key architectures including YOLO [22], Faster R-CNN [21] which integrates region proposal networks (RPN) to improve speed over R-CNN and the Fast Region-Based Convolutional Neural Network (Fast R-CNN).Subsequently, R-FCN was introduced to enhance detection accuracy, and Single Shot Multibox Detector (SSD) [23] has significantly enhanced both the accuracy and speed of detection systems. However, predictions from a single model often remain limited by model bias and uncertainty, particularly in complex scenarios involving overlapping or small objects. To overcome these limitations, several studies have proposed ensemble and fusion techniques to combine predictions from multiple models, thereby improving the robustness and overall accuracy of detection systems.

A commonly used post-processing method for combining detections is non-maximum suppression (NMS) [24]. Originally introduced to eliminate redundant bounding boxes, NMS retains the box with the highest confidence score while removing others whose overlap (Intersection over Union - IOU) exceeds a given threshold. Variants such as soft non-maximum suppression (Soft-NMS) [25] were later proposed to reduce the confidence scores of overlapping boxes instead of discarding them entirely, allowing potentially correct detections to be preserved. NMS-based methods may sometimes discard valuable information when several high-confidence detections represent the same object. To address this limitation, weighted boxes fusion (WBF) was introduced by Solovyev et al. (2021) [26]. Unlike NMS, WBF does not suppress overlapping boxes but rather fuses them by computing a weighted average of their coordinates according to their confidence scores. This approach preserves more spatial information and often leads to more accurate object localization, especially when combining results from different models.

Despite the progress surveyed above, a clear gap remains in the literature [24]–[26]: existing post-processing fusion methods for object detection typically assign fixed, uniform weights to all contributing models regardless of their actual reliability on a given image, and their application to fine-grained insect pest detection on large-scale benchmarks such as IP102 [27] remains largely unexplored. The present study directly addresses this gap by proposing a training-free adaptive post-processing fusion framework that dynamically weights model contributions based on per-image confidence, enabling robust multi-species pest detection without any architectural modification or additional training.

## 3. MATERIALS AND METHODS

### 3.1 Data Description and Pre-Processing

Collection and capture of pest images represent a particularly complex task due to morphological variations throughout their life cycle, making their identification and classification difficult. The dataset used in this study is derived from the extensive IP102 dataset released by Wu et al. [27]. This dataset includes 102 categories of insect pests, totaling 75222 samples containing over 19000 labeled images. Table 1 presents the selected dataset, which focuses on 10 weakly labeled classes, comprising a total of 2810 images with 3650 instances, indicating that some images contain multiple objects. The dataset is divided into training, validation, and test sets, containing 2248 (2955 instances), 281 (341 instances), and 281 (354 instances) images, with proportions of 80%, 10%, and 10%, respectively, to ensure a balanced evaluation and address identification and classification challenges.

The image resolutions vary significantly, with some being small and others large. However, the average image size is 0.18 megapixels (180,000 pixels), and the median aspect ratio (Width × Height) indicates that most images tend to be around 500 pixels wide and 369 pixels tall.

*Table 1: Distribution of images and annotation across the ten selected pest classes*

| Class | Label | Images | Annotation |
|---|---|---|---|
| Black Cutworm | 18 | 300 | 317 |
| Red Spider | 21 | 160 | 168 |
| Aphids | 24 | 879 | 1395 |
| Flea Beetle | 37 | 345 | 384 |
| Tarnished Plant Bug | 47 | 348 | 365 |
| Pseudococcus Comstocki Kuwana | 64 | 51 | 73 |
| Ampelophaga | 66 | 279 | 279 |
| Lycorma Delicatula | 67 | 338 | 338 |
| Chrysomphalus Aonidum | 79 | 50 | 183 |
| Aleurocanthus Spiniferus | 82 | 60 | 148 |
| TOTAL | | 2810 | 3650 |

### 3.2 Data Augmentation

One of the principal techniques employed in computer vision applications is data augmentation. The quantity of data is crucial, as it significantly influences the performance of models. Data augmentation increases the dataset size without requiring additional manual labeling. Techniques presented in Table 2, such as horizontal flips and 90-degree rotations, are applied to improve the model's ability to recognize insects from different perspectives. The grayscale filter, applied to 100% of images by adding a conversion, transforms images into black and white and removes color. This transformation was applied to all images, converting them to grayscale and increasing dataset diversity. With four distinct augmentations used before training, we generate more than 4000 different images from the same dataset.

Throughout the model's learning process, additional augmentations are applied, such as blur, median blur, to gray, and contrast-limited adaptive histogram equalization (CLAHE). These

© Little Lion Scientific
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

augmentations allow each image to be represented in multiple variants for each given image, showcasing different perspectives and features.

*Table 2: Data augmentation techniques applied to datasets during training.*

| Augmentation | Description | Data generated |
|---|---|---|
| Fliplr | Horizontal symmetry | 4 095 images |
| Flipud | Vertical symmetry | |
| Rot 90 | 90-degree rotation | |
| Grayscale | Removes colors | |
| Blur | Sharpness of an image | |
| MedianBlur | Noise and smooth out images | |
| ToGray | Removing color information | |
| CLAHE | Enhancing visibility in bright and dark areas. | |

### 3.3 Architecture Model

In this work, we aimed to leverage various object detection models, broadly categorized into different versions of YOLO [22], a well-known single-stage object detection framework. Specifically, we explored YOLOv5nu, YOLOv8n, and YOLOv10n. YOLO models are recognized for their single-stage detection process, which excels in real-time applications by directly predicting bounding boxes and class probabilities. Over successive versions, YOLO has seen significant improvements: YOLOv5 emphasizes speed and ease of use, YOLOv8 introduces anchor-free detection and advanced optimization techniques, and YOLOv10 further refines the architecture to enhance both accuracy and efficiency.

The approach applied in our work builds upon this YOLO architecture as well as prior studies in DL for object detection [28], with a particular focus on insect detection. For instance, Amrani et al. [29] explored insect detection using a YOLOv3-based adaptive feature fusion convolutional network. To further enhance performance, we employed YOLOv5, YOLOv8, and YOLOv10, each building upon previous YOLO versions and utilizing a CNN-based architecture comprising three main components: the backbone, neck, and head, as illustrated in Figure 1.

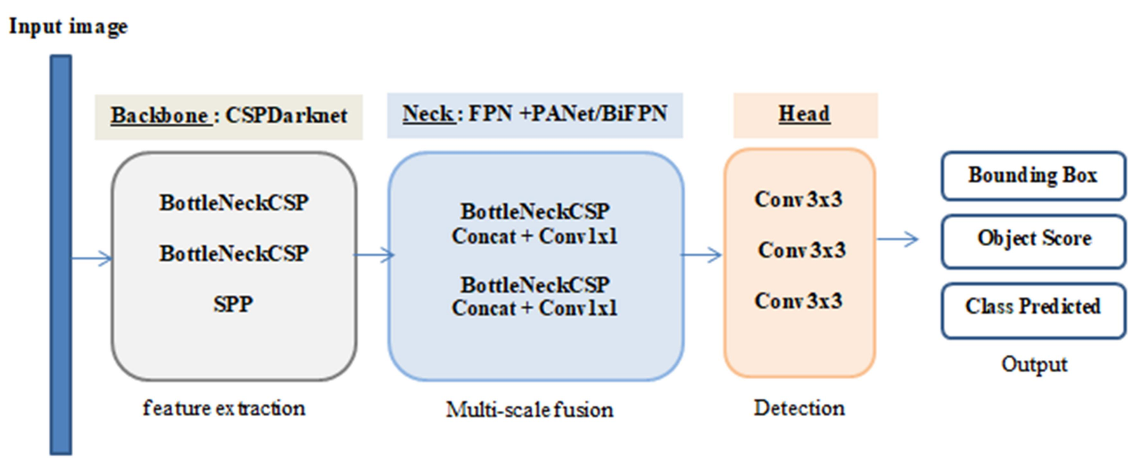



*Figure 1: Architectural Elements of YOLO-Based Object Detection.*

The first stage is a Backbone Network for feature extraction. This backbone extracts high-level features from the input images. YOLOv5 uses Darknet53 with cross stage partial network (CSPN), which reduces computational complexity by optimizing gradient combinations [30]. As a result, YOLOv5 lowers memory usage and inference time while improving model accuracy. It also provides multiple model versions by adjusting the depth and width of the feature extractor. Additionally, YOLOv5 utilizes Feature Pyramid Network (FPN)[31], enabling multi-scale object detection through a hierarchical structure to enhance accuracy. YOLOv8 and later versions incorporate more lightweight and efficient variants, such as Improved CSPDarknet. YOLOv10 enhances CSPNet for better gradient flow and efficiency.

The Neck structure in YOLO architectures is based on the FPN and the Path Aggregation Network (PAN)[32], enabling multi-scale feature fusion. YOLOv5 employs FPN with PAN to enhance multi-scale detection, while YOLOv8 and later versions optimize this approach by incorporating the Bidirectional Feature Pyramid Network (BiFPN)[33], which improves feature fusion and propagation. The continuous evolution of YOLO architectures has led to progressive enhancements in the Neck to strengthen detection performance. For instance, YOLOv10 refines BiFPN by improving feature weighting.

The second stage comprises the detection Head, responsible for bounding box prediction and objectness score estimation. The objectness score reflects the model's confidence in the predicted bounding box containing an object, along with the associated class probabilities for each object class present in the image. These outputs collectively represent the model's predictions. YOLOv5 utilizes an anchor-based detection head, relying on predefined anchor boxes for bounding box prediction. In contrast, YOLOv8 introduces an anchor-free approach, inspired by CenterNet [34] and fully convolutional one-stage detection (FCOS) [35], which simplifies the prediction process by directly estimating object centers, sizes, and confidence scores, reducing computational complexity. YOLOv10 further refines this anchor-free method by enhancing object localization and class prediction through improved feature learning techniques.

### 3.4 Proposed Approach: Adaptive WBF Using YOLO Models

Our proposed approach introduces an enhancement of the standard WBF technique, termed the adaptive fusion approach, a generic adaptive post-processing fusion framework designed to improve the accuracy and robustness of object detection when combining multiple YOLO models. Unlike traditional WBF, which assigns fixed weights to models during fusion, our adaptive fusion approach adjusts the fusion weights according to the mean confidence scores. This allows models with higher confidence on a specific image to have greater influence on the final fused output. Additionally, a confidence-boosting mechanism reinforces predictions when multiple models detect the same class, promoting consistency across models and refining the aggregation of bounding box predictions based on both confidence scores and intersection over union (IOU) overlaps.

As depicted in Figure 2, our workflow begins by evaluating the individual YOLO models separately to establish baseline performance. In the second stage, standard fusion techniques such as NMS and WBF are applied to combine pairs of models, followed by experiments with fixed weight configurations to explore their influence on the fusion process. Building on these analyses, the adaptive fusion approach consolidates the strengths of the different model combinations and provides a more flexible and context-aware fusion mechanism capable of adjusting to varying detection scenarios.

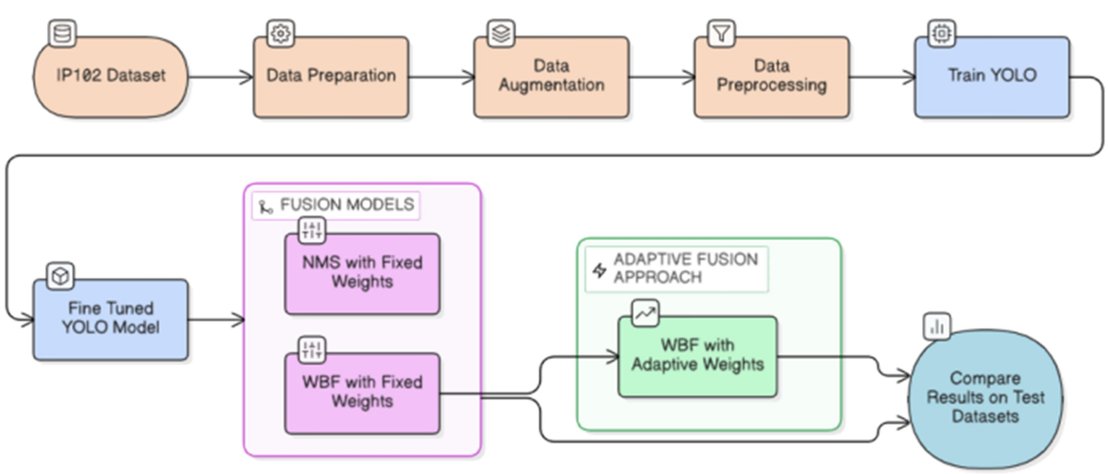



*Figure 2: Adaptive Fusion Approach Pipeline.*

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

The optimization procedure of the proposed adaptive fusion strategy consists of several well-defined stages, summarized as follows:

#### 3.4.1 Initialization

We begin with the standard YOLO architectures. Network weights are initialized using their pretrained backbones to ensure stable convergence. Each YOLO model (YOLOv5nu, YOLOv8n, and YOLOv10n) is then trained on the selected subset of the IP102 dataset using the augmented training set described in Table 2. This stage establishes strong base detectors with diverse feature representations.

#### 3.4.2 Initial Training and Fine-Tuning

Models are subsequently fine-tuned on the target dataset to improve detection of visually similar insect species, ensuring consistent performance across classes.

Before introducing adaptive behavior, we apply WBF as the fundamental mechanism for merging overlapping detections. WBF aggregates bounding box coordinates using confidence-weighted averaging, providing a more stable ensemble compared to NMS.

#### 3.4.3 Adaptive weight computation:

Unlike standard fusion, our method computes image-specific adaptive weights for each model. These weights reflect the reliability of each detector on the current image based on its mean confidence score.

- **Mean confidence score and Adaptive weight rule.**

For a given model $i$, let it output $n_i$ bounding boxes with confidence scores $s_{i,1}, s_{i,2}, \dots, s_{i,n_i}$. The mean confidence score and the adaptive weight rule are defined in Equations (1) and (2), respectively.

$$\bar{s}_i = \frac{1}{n_i} \sum_{j=1}^{n_i} s_{i,j} \tag{1}$$

$$w_i = \min(W_{\max}, W_{\text{base}} + \bar{s}_i) \tag{2}$$

With:

- $W_{\text{base}} = 1.0$
- $W_{\max} = 2.0$

If a model produces no detections for an image, it is assigned a neutral weight $w_i = 1.0$

This mechanism ensures that models performing reliably on a specific image contribute more strongly to the fusion process, while preventing any single detector from dominating.

#### 3.4.4 Adaptive weighted fusion

For each detector, the predicted bounding boxes, their corresponding confidence scores, and the predicted labels are collected as inputs to the fusion process. Boxes belonging to the same class and overlapping with an IoU threshold of 0.55 are grouped into a cluster. This value follows the recommendation of Solovyev et al. [26], who report it as close to an optimal threshold for WBF at mAP@0.5.

The fused bounding box and the fused confidence score are then computed from these clusters using confidence-weighted averaging, as defined in Equations (3) and (4), respectively.

$$B_f = \frac{\sum_{i=1}^{N} \sum_{j \in \mathcal{C}} w_i \, s_{i,j} \, b_{i,j}}{\sum_{i=1}^{N} \sum_{j \in \mathcal{C}} w_i \, s_{i,j}} \tag{3}$$

Where:

$N$ : The number of models.
$b_{i,j}$ : The $jth$ bounding box produced by model $i$.
$s_{i,j}$ : Its confidence score.
$w_i$ : The adaptive weight computed earlier

$$S_f = \frac{\sum_{i=1}^{N} w_i \; s_i^{max}}{\sum_{i=1}^{N} w_i} \tag{4}$$

Let:

$$s_i^{max} = \max_{j \in \mathcal{C}} s_{i,j}$$

#### 3.4.5 Confidence boosting and evaluation:

After fusion, when at least two detectors assign the same class label to the merged box, an additive confidence bonus $\beta = 0.1$ is added to the fused score and clipped to a maximum of 1.0, as defined in Equation (5). This consensus-driven reinforcement reduces false negatives by rewarding multi-model agreement and promoting prediction consistency across detectors.

The final fused predictions are then evaluated on the test set using accuracy, precision, recall, F1-score, and mAP@0.5.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

$$S_f = \min(1.0\ ,\ S_f + \beta) \tag{5}$$

Here, β = 0.1 is a small additive bonus applied only when at least two detectors predict the same class label for the merged box. The min(·) operator ensures the fused confidence remains within the valid [0, 1] range.

### 3.4.6 Experiments and Evaluation Metrics

In this study, experiments were conducted using Python 3.10, Ultralytics version 8.1, and PyTorch 2.1 as the deep learning framework. All experiments were executed on Google Colab. All models were trained on an NVIDIA Tesla T4 GPU with CUDA 12.2 support for a total of 50 epochs. This hardware configuration ensured efficient training while maintaining reproducibility across all model variants.

As described in the dataset section, the experimental dataset contains images from 10 insect pest classes, partitioned into 2,248 training samples (80%), 281 validation samples (10%), and 281 test samples (10%). The input resolution was fixed at 640×640 for all YOLO variants.

Training was performed using the AdamW optimizer, which automatically determined optimal hyperparameters during training. The final configuration resulted in a learning rate of 0.000714, a momentum term of 0.9, and a weight decay of 0.0001 for selected layers. Data loading employed two workers and incorporated extensive augmentation strategies: horizontal and vertical flips, rotations, grayscale conversion, blur, median blur, CLAHE, and Mosaic augmentation, implemented through the Albumentations and Mosaic libraries. These augmentations enhanced robustness against variations in lighting, scale, orientation, and background complexity.

- **Evaluation metrics:**

Object detection performance is commonly assessed using several metrics, including precision, recall, F1-score, and mean Average Precision (mAP). These metrics are derived from the comparison between predicted bounding boxes and ground-truth annotations, using IoU as the spatial matching criterion.

A confusion matrix is constructed using the following components:

- True Positives (TP): Correctly detected objects.
- False Positives (FP): Incorrectly detected objects.
- False Negatives (FN): Missed detections.
- True Negatives (TN): Correctly identified background.

Precision reflects the proportion of correct positive predictions, and recall quantifies the proportion of actual objects successfully detected. They are defined as (6) and (7):

$$\text{Precision} = \frac{TP}{TP + FP} \tag{6}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{7}$$

Average precision (AP) and Mean average precision (mAP). Average Precision (AP) evaluates detection performance for a single class by computing the area under the precision--recall curve. Mean Average Precision (mAP) aggregates AP values across all classes and is widely used as the principal evaluation metric in object detection benchmarks (8):

$$\text{mAP} = \frac{\sum_{c=1}^{C} AP(c)}{C} \tag{8}$$

Where $C$ is the total number of classes.

F1-Score combines precision and recall into a single metric by computing their harmonic mean. It is informative in class-imbalance scenarios because it penalizes models that achieve high precision but low recall (or vice versa):

$$F1 = \frac{2 \times (\text{Precision} \times \text{Recall})}{\text{Precision} + \text{Recall}} \tag{9}$$

Although mean Average Precision (mAP) is the standard evaluation metric in object detection, accuracy was additionally included as a complementary indicator of overall detection performance.

The accuracy metric is defined as:

$$\text{Accuracy} = \frac{TP}{\text{TP} + \text{FP} + \text{FN}} \tag{10}$$

The selection of these metrics is motivated by the requirements of object detection evaluation.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

mAP@0.5 is adopted as the primary metric for direct comparability with established benchmarks such as PASCAL VOC [19] and IP102 [27]. Precision, Recall, F1-score, and accuracy are additionally reported to provide a comprehensive assessment of detection performance, balancing localization accuracy, classification effectiveness, and overall predictive capability [36]-[37].

## 4. RESULTS AND DISCUSSION

### 4.1 Performance Evaluation of YOLOv8n On 97 Classes

The initial phase of our work involved training the YOLOv8 model on 97 classes from the IP102 dataset. The mAP@0.5 for each class and confusion matrix results are illustrated in Figures 3 and 4, while the model's performance in terms of precision, recall, mAP@0.5, and mAP@0.5-95 is presented in Table 3.

The results show that YOLOv8, applied to 97 classes, performs well on the 10 selected insect pests, with precision and recall values close to 100% for class Lycorma Delicatula and class Ampelophaga as presented in Table 3. The mAP@0.5 values for all classes are 82%, generally elevated for the majority of insect species as shown in Figure 4.

*Table 3: Performance of YOLOv8 on 97 pest classes.*

| Class | label | images | Instances(%) | P (%) | R(%) | mAP@0.5(%) |
|---|---|---|---|---|---|---|
| All | | 15178 | 17839 | 77 | 77 | 82 |
| Black Cutworm | 18 | 15178 | 250 | 66 | 90 | 86 |
| Red Spider | 21 | 15178 | 131 | 84 | 85 | 93 |
| Aphids | 24 | 15178 | 1113 | 84 | 94 | 96 |
| Flea Beetle | 37 | 15178 | 310 | 95 | 98 | 99 |
| tarnished plant bug | 47 | 15178 | 294 | 85 | 89 | 94 |
| Pseudococcus comstocki Kuwana | 64 | 15178 | 60 | 73 | 98 | 91 |
| Ampelophaga | 66 | 15178 | 223 | 97 | 1 | 99 |
| Lycorma Delicatula | 67 | 15178 | 270 | 99 | 1 | 99 |
| Chrysomphalus Aonidum | 79 | 15178 | 144 | 83 | 88 | 91 |
| Aleurocanthus Spiniferus | 82 | 15178 | 121 | 78 | 82 | 88 |

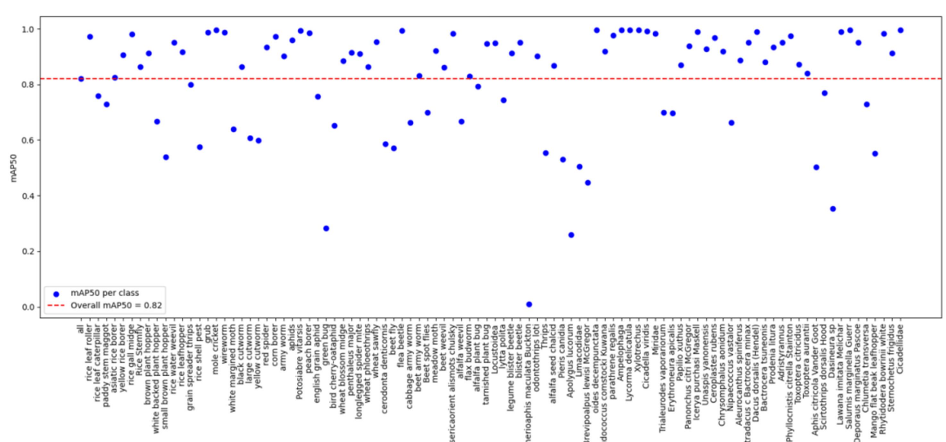



*Figure 3: mAP@0.5 performance for 97 classes using YOLOv8n.*

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

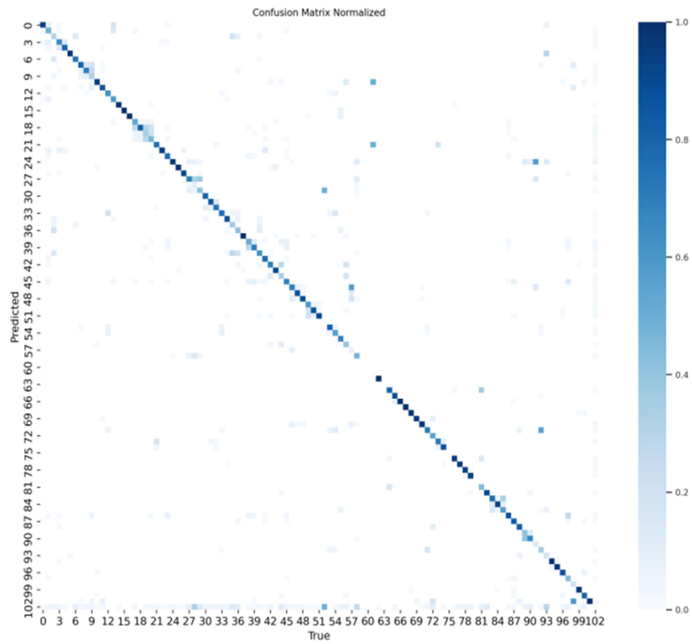



*Figure 4: Confusion matrix for 97 classes using YOLOv8n.*

### 4.2 Training Evaluation on the Selected Classes

Table 4, summarizes the training performance of the three YOLO models. YOLOv10n obtains the highest Precision and mAP@0.5 (86%), reflecting improved localization accuracy and fewer false positives. Its Recall (76%), however, is slightly lower than that of YOLOv5nu and YOLOv8n (both 79%), indicating a modest tendency to miss some object instances. YOLOv8n offers a minor precision improvement over YOLOv5nu but does not increase mAP@0.5. Overall, the results show that YOLOv10n provides the strongest detection accuracy, while YOLOv5nu is preferable in scenarios where training time or limited computational resources are a constraint.

### 4.3 Fine-tuned Models and Fixed-weight Fusion Models Results.

The results of the pre-trained models on the test datasets are presented in Table 5, which includes both the performance of individual models and those obtained through model fusion using NMS and WBF. The mAP@0.5 values remain relatively similar across the different configurations, while the accuracy shows an improvement of approximately +1% to +3%.

To optimize the contribution of each model, eleven different weight combinations ranging from 0.0 to 1.0 were systematically tested. For each configuration, inference was performed on the test dataset, and evaluation metrics including Precision, Recall, F1-score, Accuracy, and mAP@0.5 were computed. The best-performing combination, based on mAP@0.5 was selected, as summarized in Table 6.

*Table 4: Performance of models during training phase.*

| Metrics | YOLOv5nu | YOLOv8n | YOLOv10n |
|---|---|---|---|
| Precision (%) | 80 | 83 | 86 |
| Recall (%) | 79 | 79 | 76 |
| mAP@0.50 (%) | 83 | 83 | 86 |
| Time | 1h 21m | 1h 38m | 1h 35m |

*Table 5: Performance comparison of individual YOLO models and pairwise WBF/NMS fusion on test datasets.*

| Model | Precision (%) | Recall (%) | F1-Score (%) | mAP (%) | Accuracy (%) |
|---|---|---|---|---|---|
| YOLOv5nu | 96.6 | 96.4 | 96.4 | 92.9 | 95.7 |

| | | | | | |
|---|---|---|---|---|---|
| YOLOv8n | 97.2 | 97.1 | 97.1 | 91.3 | 95.3 |
| YOLOv10n | 96.9 | 96.7 | 96.7 | 92.5 | 95.3 |
| Fusion Methods | | | | | |
| WBF+(v8n+v10n) | 97.6 | 97.4 | 97.4 | 91.0 | 96.8 |
| WBF+(v8n+v5n) | 98.9 | 98.9 | 98.9 | 91.0 | **98.5** |
| WBF+(v5nu+v10n) | 98.2 | 98.2 | 98.1 | 90.5 | 97.8 |
| NMS+(v8n+v10n) | 97.6 | 97.4 | 97.4 | 91.0 | 96.8 |
| NMS+(v8n+v5nu) | 97.9 | 97.8 | 97.8 | 91.0 | 97.9 |
| NMS+(v5nu+v10n) | 98.2 | 98.2 | 98.1 | 90.5 | 97.8 |

*Table 6: Performance of NMS and WBF fusion models with fixed weights on the test set*

| Fusion method+ Models | weights | Precision (%) | Recall (%) | F1-Score (%) | mAP (%) | Accuracy (%) |
|---|---|---|---|---|---|---|
| WBF v8n+v10n | [0.6,0.4] | 97.9 | 97.8 | 97.8 | 91.0 | 97.1 |
| WBF v8n +v5nu | [0.5,0.5] | 98.9 | 98.9 | 98.9 | 91.0 | **98.9** |
| WBF v5n+v10n | [0.5,0.5] | 98.2 | 98.2 | 98.1 | 90.5 | 97.8 |
| NMS v8n+ v10n | [0.4,0.6] | 97.6 | 97.4 | 97.4 | 91.0 | 96.8 |
| NMS v8n+v5nu | [0.5,0.5] | 97.9 | 97.8 | 97.8 | 91.0 | 97.5 |
| NMS v5nu+v10n | [0.5,0.5] | 98.2 | 98.2 | 98.1 | 90.5 | 97.8 |

### 4.4 Our Adaptive Fusion Approach Results.

During the three conducted tests, it was observed that the mAP@0.5 performance did not exceed 91%. In contrast, the results obtained from our proposed approach demonstrate a clear improvement, both in terms of accuracy, reaching values comparable to the best-performing models (approximately 98%), and in terms of mAP@0.5, where it significantly outperforms them, as shown in Table 7 and Figure 5. Furthermore, this improvement is achieved without requiring any additional training, relying solely on post-processing adjustments at inference time.

*Table 7: Performance comparison between individual YOLO models and proposed adaptive fusion approach on test set.*

| Model | Fusion method | mAP@0.5 (%) | Accuracy (%) |
|---|---|---|---|
| YOLOv5nu | - | 92.9 | 95.7 |
| YOLOv8n | - | 91.3 | 95.3 |
| YOLOv10n | - | 92.5 | 95.3 |
| Our Adaptive Fusion Approach | | | |
| YOLOv8n +YOLOv5nu | WBF (ours) | 97.2 | 98.5 |
| YOLOv5nu +YOLOv10n | WBF (ours) | 96.8 | 98.2 |
| YOLOv8n +YOLOv5nu | NMS (ours) | 97.3 | 98.2 |

- **Accuracy:**

The adaptive fusion approach reached 98.58%, showing an improvement of +2.85% compared to individual models, and a comparable performance to the fusion model (YOLOv8n + YOLOv5nu) with WBF, and fixed-weights [0.5, 0.5] which achieved 98.95%.

- **Mean average precision at IoU 0.5:**

As reported in Table 8, the adaptive approach achieved a mAP@0.5 of 97.22%, representing an improvement of approximately +5% over all individual YOLO models and roughly +6% over the standard WBF and NMS fusion methods.

In terms of precision, recall, the proposed approach, whether implemented with WBF, NMS, or Soft-NMS, consistently outperforms the individual YOLO models. The highest precision and recall achieved by a single model is 97.28% for YOLOv8n, whereas our adaptive fusion (YOLOv8n+YOLOv5nu) reaches 98.58% for both precision and recall, corresponding to an improvement of approximately +1%.For the F1-score, YOLOv8n attains 97.10%, while the adaptive fusion achieves 98.55%, again reflecting a gain of roughly +1%.

Figure 6 compares the number of correct, incorrect, and non-detected predictions across the three individual YOLO models and the proposed adaptive fusion approach. Overall, although the performance levels are relatively close, the approach consistently provides a measurable and systematic improvement compared to the individual models.

Journal of Theoretical and Applied Information Technology
15st July 2026. Vol.104. No.13
© Little Lion Scientific
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Figure 7 illustrates representative examples of detection comparison between YOLO models and our adaptive approach.

*Table 8: Performance comparison between the proposed adaptive fusion and individual YOLO models on the test dataset.*

| Model | YOLOv5nu | YOLOv8n | YOLOv10n | Our adaptive fusion (YOLOv5nu+YOLOv8n) |
|---|---|---|---|---|
| Accuracy (%) | 95.7 | 95.3 | 95.3 | 98.58 |
| Precision (%) | 96.6 | 97.2 | 96.9 | 98.5 |
| Recall (%) | 96.4 | 97.1 | 96.7 | 98.5 |
| F1 Score (%) | 96.4 | 97.1 | 96.7 | 98.55 |
| mAP@0.5 (%) | 92.9 | 91.3 | 92.5 | 97.2 |
| Correct | 269 | 268 | 268 | 277 |
| Incorrect | 10 | 8 | 9 | 4 |
| Non-Detected | 2 | 5 | 4 | 0 |

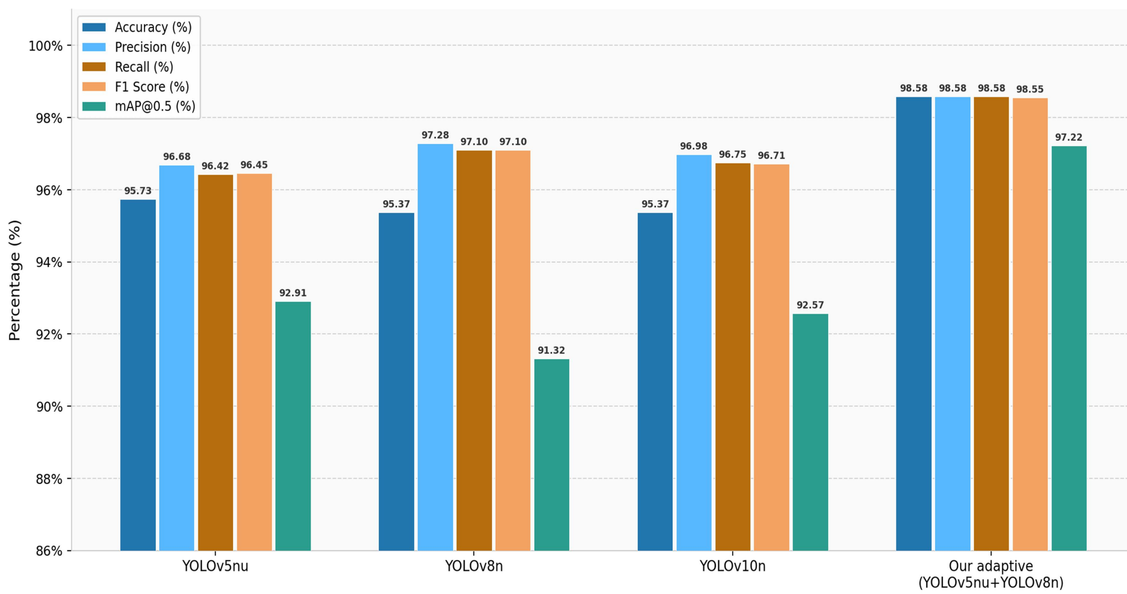
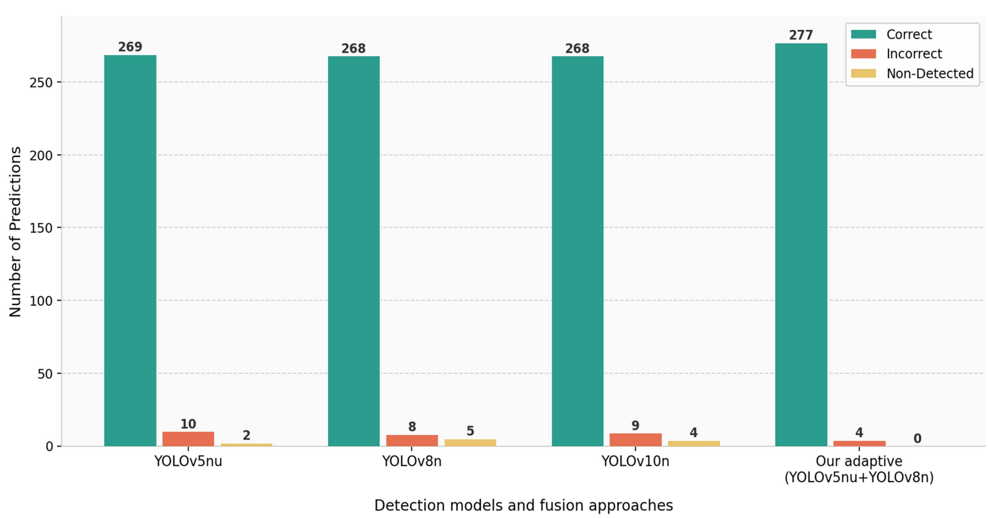



*Figure 5: Comparative Performance of YOLO Models and The Adaptive Fusion Approach.*




.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

*Figure 6: Prediction Outcomes Per Detection Models and our Adaptive Fusion Approach.*

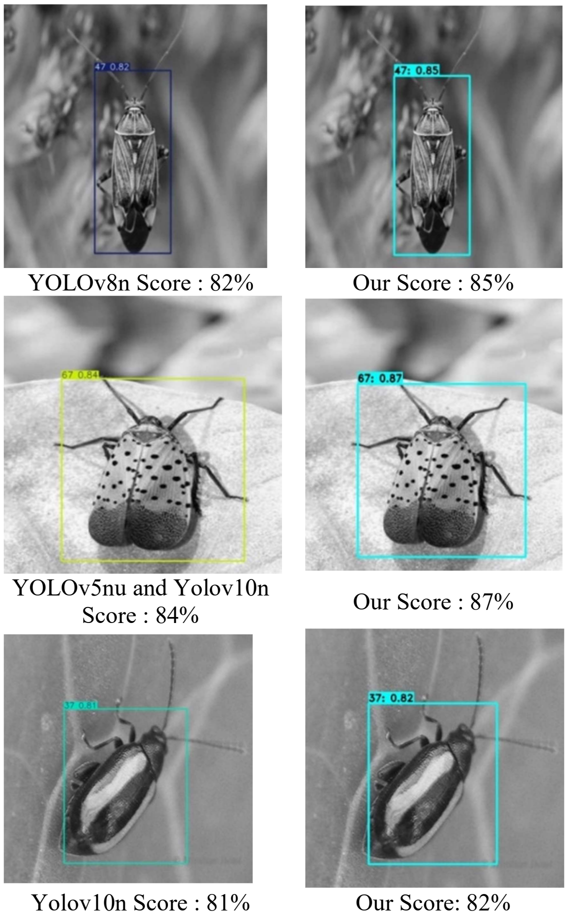



*Figure 7: Examples of Comparative Analysis of Accuracy on Test Datasets Between YOLO Models and The Adaptive Fusion Approach.*

Our proposed adaptive fusion approach, based on fine-tuning YOLO models combined with bounding box post-processing methods, achieves 97.2% mAP@0.5 and 98.5% accuracy with WBF on the YOLOv8n + YOLOv5nu configuration. This surpasses the individual models tested and state-of-the-art techniques considered in this paper, as shown in Table 9, and it should be noted that the comparisons are indicative rather than strictly equivalent. The referenced methods differ in dataset composition, number of target classes, training configurations, and evaluation protocols, making direct performance equivalence impossible to establish. These comparisons are therefore included to provide a general sense of positioning relative to the state of the art rather than to serve as rigorous benchmarks.

This study set out to exploit the complementary strengths of multiple YOLO detectors through adaptive post-processing fusion to achieve superior detection precision and mAP compared to single-model detectors. The results confirm the full achievement of this objective: the proposed adaptive fusion approach reached 97.22% mAP@0.5 and 98.58% accuracy, surpassing all individual models by approximately +5% and +2.85% respectively, and outperforming fixed-weight NMS and WBF baselines by approximately +6% in mAP all without any additional model training. When positioned against prior work in Table 9, the proposed framework achieves substantially higher mAP@0.5 than methods such as YOLOv5s [38] (64%) and YOLO-Pest [40] (57.1%) evaluated on IP102, while operating on a focused ten-class subset. It should be noted, as acknowledged earlier, that these comparisons are indicative rather than strictly equivalent due to differences in dataset partitions, class counts, and evaluation protocols; nonetheless, they provide meaningful context for situating the proposed contribution within the current landscape of pest detection research.

The comparative analysis of recent pest-detection studies highlights clear disparities in performance across different DL paradigms. Earlier works such as ResNet-50 and Attention U-Net achieved reasonable accuracy on small or simplified datasets but demonstrated limited scalability when confronted with the visual diversity and class imbalance characteristic of large agricultural benchmarks. Likewise, improvements introduced in modified YOLOv5 architectures, such as attention modules, refined backbone structures, or enhanced feature-fusion mechanisms, which achieved limited performance on large benchmarks like IP102 with mAP@0.5 values frequently below 65%, more sophisticated attention-enhanced detectors, including CornerNet with DenseNet-100 or ViT-integrated approaches such as YOLO-Pest, showed competitive results on curated datasets. However, their performance typically degrades when evaluated on broader multi-species collections, suggesting limitations in generalization. Even high-capacity architectures such as YOLOv5x deliver exceptional results only on constrained datasets (e.g., IP-23), without demonstrating similar robustness on more heterogeneous pest environments.

In contrast, the proposed method substantially improves detection accuracy and robustness by integrating the complementary strengths of multiple YOLO architectures through dynamic weighting and ensemble aggregation using WBF, yielding the

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

highest performance among the configurations tested. While direct comparison with prior IP102 studies is not strictly possible due to differences in dataset partitions, class counts, and evaluation protocols, the results in Table 9 position our framework favorably within the current landscape of pest-detection methods.

It is worth acknowledging that the literature presents differing perspectives on ensemble methods. Some studies suggest that sufficiently large single-model architectures can approach ensemble-level performance without multi-model inference overhead, while others argue that end-to-end learned fusion strategies are more principled than post-processing approaches. The present study contributes empirical evidence that adaptive post-processing fusion, even with lightweight nano-scale detectors, yields meaningful performance gains at a fraction of the cost of larger architectures or learned fusion methods.

*Table 9: Performance our proposed approach vs. state-of-the-art on pest datasets using map@0.5 and accuracy.*

| Model / Technique | Year | Data | Classes | mAP@0.5 (%) | Accuracy (%) |
|---|---|---|---|---|---|
| ResNet-50 - SVM [36] | 2025 | Custom | 10 | — | 94 |
| SVM baseline [36] | 2025 | Custom | 10 | — | 85.2 |
| U-Net(DMSAU-Net)[37] | 2024 | IP102 | 14 | — | 92.1 |
| YOLOv5s [38] | 2023 | IP102 | 40 | 64 | - |
| DenseNet100+CornerNet [39] | 2023 | IP102 | 8 | 57.2 | 68.7 |
| YOLO-Pest (CAC3+SE+ConvNeXt) [40] | 2023 | IP102 | 28 | 57.1 | — |
| YOLO-Pest (CAC3+SE+ConvNeXt) [40] | 2023 | Teddy Cup pest | 28 | 91.9 | — |
| Our adaptive approach. | | | | | |
| (v8n+v5nu) + WBF | 2026 | IP102 | 10 | 97.2 | 98.5 |
| (v5nu+v8n+v10n) + WBF | | | | 96.5 | 97.8 |
| (v5nu+v10n) + WBF | | | | 96.8 | 98.2 |
| (v8n+v5nu) + NMS | | | | 97.3 | 98.2 |
| (v8n+v5nu) + Soft-NMS | | | | 97.8 | 96.9 |

These findings demonstrate that ensemble-based adaptive fusion provides a more reliable and scalable solution for pest detection in real agricultural settings, effectively mitigating the weaknesses of single-model detectors by enhancing feature diversity and maintaining high accuracy even in the presence of complex backgrounds, overlapping instances, and subtle inter-class variations. Overall, the results position adaptive fusion as a strong candidate for next-generation intelligent pest-monitoring systems.

## 5. CONCLUSION

In this study, we presented a generic adaptive post-processing fusion framework that dynamically combines predictions from heterogeneous YOLO detectors (YOLOv5nu, YOLOv8n, and YOLOv10n), using per-image adaptive weights over NMS and WBF outputs, applied to ten pest classes from the IP102 dataset. The results demonstrate that the proposed method improves detection performance compared to these standard post-processing techniques, highlighting its effectiveness in real-world insect detection tasks.

The experimental results confirm this expectation: The generic adaptive post-processing fusion approach achieved 98.58% accuracy and 97.22% mAP@0.5, outperforming all individual models by approximately +2.85% and +5% respectively, and surpassing standard fixed-weight WBF and NMS baselines by roughly +6% in mAP@0.5. This approach maximizes detection accuracy by optimally combining the strengths of multiple models through weighted fusion. By dynamically adjusting model contributions based on confidence scores, it ensures more consistent and reliable detections. The fusion reduces the impact of weaker individual predictions, leading to more precise bounding box localization across diverse insect classes.

Despite these promising results, some limitations should be considered. The study is limited to ten pest classes from IP102, three nano-scale YOLO variants and the generalization of the approach to larger datasets, multi-stage detectors or transformer-based architectures is limited. The confidence boosting parameter β = 0.1 and the IoU threshold of 0.55 are used without systematic ablation and the substantial class imbalance among some species warrants further investigation. Furthermore, the framework has been validated only in a controlled offline setting and the deployment in real-world scenarios under variable field conditions needs to be evaluated.

In future work, the framework will be extended to a larger number of insect classes and subclasses, which will enhance its applicability for insect detection and classification. Future efforts will also include a comparative study between single-stage and multi-stage detectors, such as Faster R-CNN, along with alternative backbone architectures, to further strengthen the benchmarking of the proposed approach. Overall, the adaptive fusion framework introduced in this study can serve as benchmark architecture for future research and as a reliable building block for future pest-monitoring pipelines in smart agriculture systems.

## ACKNOWLEDGEMENTS

We would like to express our sincere gratitude to the researchers who developed the IP102 dataset. This dataset has been an invaluable resource for our research, providing a comprehensive benchmark for insect pest detection and classification. We appreciate the efforts of Wu et al. (2019) in curating and making the dataset publicly available, which has significantly contributed to advancements in deep learning-based pest identification.

## CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

## DATA AVAILABILITY

The IP102 dataset used in this study is available at: https://github.com/xpwu95/IP102

## REFERENCES:

[1] F. Fina, P. Birch, R. Young, J. Obu, B. Faithpraise, and C. Chatwin, “automatic plant pest detection and recognition using k-means clustering algorithm and correspondence filters,” 2013. [Online]. Available: http://www.bipublication.com

[2] H. Al Hiary, S. Bani Ahmad, M. Reyalat, M. Braik, and Z. ALRahamneh, “Fast and Accurate Detection and Classification of Plant Diseases,” Int. J. Comput. Appl., vol. 17, no. 1, pp. 31–38, Mar. 2011, doi: 10.5120/2183-2754.

[3] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie,“The Caltech-UCSD Birds-200-2011 Dataset,” Caltech, vol. 42, no. 15, pp. 50–57, 2011, Accessed: Nov. 14, 2024. [Online]. Available: https://authors.library.caltech.edu/27452

[4] J. Krause, M. Stark, J. Deng, and L. Fei-Fei, “3D object representations for fine-grained categorization,” in Proceedings of the IEEE International Conference on Computer Vision, IEEE, Dec. 2013, pp. 554–561. doi: 10.1109/ICCVW.2013.77.

[5] N. Larios et al., “Automated insect identification through concatenated histograms of local appearance features: Feature vector generation and region detection for deformable objects,” Mach. Vis. Appl., 2008, doi: 10.1007/s00138-007-0086-y.

[6] S. N. Yaakob and L. Jain, “An insect classification analysis based on shape features using quality threshold ARTMAP and moment invariant,” Appl. Intell., 2012, doi: 10.1007/s10489-011-0310-3.

[7] J. Wang, C. Lin, L. Ji, and A. Liang, “A new automatic identification system of insect images at the order level,” Knowledge-Based Syst., 2012, doi: 10.1016/j.knosys.2012.03.014.

[8] P. J. D. Weeks, M. A. O’Neill, K. J. Gaston, and I. D. Gauld, “Species-identification of wasps using principal component associative memories,” Image Vis. Comput., 1999, doi: 10.1016/S0262-8856(98)00161-9.

[9] W. S. McCulloch and W. Pitts, “A logical calculus of the ideas immanent in nervous activity,” Bull. Math. Biophys., 1943, doi: 10.1007/BF02478259.

[10] G. E. Hinton, S. Osindero, and Y. W. Teh, “A fast learning algorithm for deep belief nets,” Neural Comput., 2006, doi: 10.1162/neco.2006.18.7.1527.

[11] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” Proc. IEEE, 1998, doi: 10.1109/5.726791.

[12] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” Neural Comput., vol. 9, no. 8, pp. 1735–1780, 1997.

[13] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Distributed Representations of Words and Phrases and their Compositionality," arXiv preprint arXiv:1310.4546, 2013.

[14] Jia Deng, Wei Dong, R. Socher, Li-Jia Li, Kai Li, and Li Fei-Fei, “ImageNet: A large-scale

hierarchical image database," 2009. doi: 10.1109/cvprw.2009.5206848.

[15] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in Advances in Neural Information Processing Systems, 2012.

[16] C. Szegedy et al., "Going deeper with convolutions," in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2015. doi: 10.1109/CVPR.2015.7298594.

[17] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings, 2015.

[18] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2016. doi: 10.1109/CVPR.2016.90.

[19] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (VOC) challenge," Int. J. Comput. Vis., 2010, doi: 10.1007/s11263-009-0275-4.

[20] W. Ding, and G. Taylor, "Automatic moth detection from trap images for pest management," Computers and Electronics in Agriculture, vol. 123, pp. 17–28, Apr. 2016, doi: 10.1016/j.compag.2016.02.003.

[21] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in Advances in Neural Information Processing Systems, 2015.

[22] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2016. doi: 10.1109/CVPR.2016.91.

[23] W. Liu et al., "SSD: Single shot multibox detector," in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2016. doi: 10.1007/978-3-319-46448-0_2.

[24] D. Cremers, I. Reid, H. Saito, and M. H. Yang, "Computer Vision-ACCV 2014:12th Asian Conference on Computer Vision Singapore, Singapore, November 1–5, 2014 Revised Selected Papers, Part I," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 9003, no. April, 2014, doi: 10.1007/978-3-319-16865-4.

[25] N. Bodla, B. Singh, R. Chellappa, and L. S. Davis, "Soft-NMS - Improving Object Detection with One Line of Code," Proc. IEEE Int. Conf. Comput. Vis., vol. 2017–October, pp. 5562–5570, 2017, doi: 10.1109/ICCV.2017.593.

[26] R. Solovyev, W. Wang, and T. Gabruseva, "Weighted boxes fusion: Ensembling boxes from different object detection models," Image Vis. Comput., vol. 107, 2021, doi: 10.1016/j.imavis.2021.104117.

[27] X. Wu, C. Zhan, Y. K. Lai, M. M. Cheng, and J. Yang, "IP102: A large-scale benchmark dataset for insect pest recognition," in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2019. doi: 10.1109/CVPR.2019.00899.

[28] A. Fuentes, S. Yoon, S. C. Kim, and D. S. Park, "A robust deep-learning-based detector for real-time tomato plant diseases and pests recognition," Sensors (Switzerland), 2017, doi: 10.3390/s17092022.

[29] A. Amrani, F. Sohel, D. Diepeveen, D. Murray, and M. G. K. Jones, "Insect detection from imagery using YOLOv3-based adaptive feature fusion convolution network," Crop Pasture Sci., 2023, doi: 10.1071/CP21710.

[30] C. Y. Wang, H. Y. Mark Liao, Y. H. Wu, P. Y. Chen, J. W. Hsieh, and I. H. Yeh, "CSPNet: A new backbone that can enhance learning capability of CNN," in IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, 2020. doi: 10.1109/CVPRW50498.2020.00203.

[31] T. Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, 2017. doi: 10.1109/CVPR.2017.106.

[32] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path Aggregation Network for Instance Segmentation," in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2018. doi: 10.1109/CVPR.2018.00913.

[33] M. Tan, R. Pang, and Q. V. Le, "EfficientDet: Scalable and efficient object detection," in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2020. doi: 10.1109/CVPR42600.2020.01079.

[34] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian, "CenterNet: Keypoint triplets for object detection," in Proceedings of the IEEE International Conference on Computer Vision, 2019. doi: 10.1109/ICCV.2019.00667.

[35] Z. Tian, C. Shen, H. Chen, and T. He, "FCOS: Fully convolutional one-stage object detection," in Proceedings of the IEEE International Conference on Computer Vision, 2019. doi: 10.1109/ICCV.2019.00972.

[36] P. K. Saket and N. Gupta, "Eksplorium Insect Classification Using Deep Learning and Machine Learning : A Comparative Study for Agricultural Applications Eksplorium," vol. 46, no. 2, pp. 302–313, 2025.

[37] X. Wang, S. Zhang, and T. Zhang, "Crop insect pest detection based on dilated multi-scale attention U-Net," Plant Methods, vol. 20, no. 1, Dec. 2024, doi: 10.1186/s13007-024-01163-w.

[38] Y. Yan, M. Xie, D. Duan, Z. Xie, and Y. Li, "Agricultural Pest Small Target Detection Algorithm Based on Improved YOLOv5 Architecture Haiying Song Guangdong Polytechnic Normal University," pp. 1–15, 2023, [Online]. Available: https://doi.org/10.21203/rs.3.rs-3109779/v1

[39] W. Albattah, M. Masood, A. Javed, M. Nawaz, and S. Albahli, "Custom CornerNet: a drone-based improved deep learning technique for large-scale multiclass pest localization and classification," Complex Intell. Syst., 2023, doi: 10.1007/s40747-022-00847-x.

[40] Q. Xiang, X. Huang, Z. Huang, X. Chen, J. Cheng, and X. Tang, "YOLO-Pest: An Insect Pest Object Detection Algorithm via CAC3 Module," Sensors, vol. 23, no. 6, Mar. 2023, doi: 10.3390/s23063221.