Evaluating the Performance of YOLO Architectures for Effective Gun and Knife Detection

Suradej Intagorn[1], Mathuros Panmuang[2] Chonnikarn Rodmorn[3] and Suriya Pinitkan[4,*]

[1] Faculty of Information and Communication Technology, Mahidol University, Salaya, Phutthamonthon, Nakhon Pathom, Thailand
[2] Department of Educational Technology and Communications, Rajamangala University of Technology Thanyaburi, Rangsit-Nakornnayok Road, Klong 6 Sub-District, Thanyaburi District, Pathum Thani, Thailand
[3] Department of Applied Statistics, King Mongkut's University of Technology North Bangkok, Pracharat 1 Road, Wongsawang, Bangsue, Bangkok, Thailand
[4] Department of Computational Science and Digital Technology, Kasetsart University, Kamphaeng Saen Campus, Kamphaeng Saen Sub-District, Kamphaeng Saen District, Nakhon Pathom, Thailand

*Corresponding Email : suriya.p@ku.th

Received June 13, 2024, Revised September 7, 2024, Accepted December 15, 2025, Published June 26, 2025

Abstract. *In recent years, the rise in mass shootings in Thailand has highlighted the need for more comprehensive and cost-effective security solutions. One approach is using artificial intelligence to assist human security personnel, particularly for weapon detection through security cameras. Although advancements in deep learning and computer vision have made it possible to deploy such systems on edge computing devices, real-time weapon detection still faces challenges like accuracy and latency. This study addresses the gap in weapon detection research specific to Thailand by utilizing a dataset featuring local environments and weapons, which differ from those in existing datasets. We compare the performance of YOLO versions 5 through 8, focusing on their mean average precision (mAP) in detecting guns and knives. Since each YOLO version is developed by different research teams and may perform differently under specific conditions, our evaluation considers these variations. The findings indicate that YOLOv8 achieves the highest mAP, with scores of 0.874 on the validation set and 0.848 on the test set, demonstrating its effectiveness in the Thai context.*

Keywords:

YOLO, machine learning, weapon detection, convolutional neural network, computer vision, Thailand

1. Introduction

Recent advancements in deep learning techniques have significantly accelerated many computer vision research areas, such as face identification and object detection [1]. Deep learning and transfer learning have shifted the computer vision paradigm from manually crafted features to learning features directly from data. The progress in object detection, particularly through deep convolutional neural networks (CNNs), can largely be attributed to these deep learning advancements [2].

In recent years, there has been an increase in firearm violence incidents in Thailand, such as the tragic event at a child development center in Nong Bua Lamphu province, which was the deadliest mass shooting in Southeast Asia. Thailand has the second-highest rate of firearm-related homicides in Southeast Asia. Budi suggested that this may be due to the high rate of firearm ownership in Thailand, which ranks first in Southeast Asia with around 10.3 million firearms owned by civilians, equating to approximately 15 guns per 100 people. However, only 6.2 million of these firearms are registered [3].

The application of artificial vision algorithms to images from video surveillance systems can enhance security [4]. Consequently, our research team plans to use computer vision algorithms to automatically detect weapons such as knives and firearms from surveillance cameras, aiming to reduce the severity of firearm-related incidents in public areas in Thailand.

A real-time weapon detection system can significantly reduce the likelihood of mass shooting tragedies. While other object detection models like Mask R-CNN could potentially be applied to weapon detection, their multi-stage architecture typically requires more computational resources, leading to slower processing speeds [5]. As a result, they may be less suitable for real-time applications, such as the weapon detection system explored in this study.

This research utilizes the YOLO architecture, known for its compact size and rapid processing speed. YOLO's straightforward design enables the neural network to immediately output bounding box positions and categories. Its efficiency in real-time video detection comes from the direct use of the entire image, which helps minimize errors in distinguishing background objects [6]. Our research aims to compare several versions of YOLO—specifically YOLOv5, YOLOv6, YOLOv7, and YOLOv8—for weapon detection, establishing baselines for future research on automatic weapon detection in Thailand.

It is important to note that newer versions of YOLO should not be assumed to be universally superior to earlier versions. Each version may have been developed by different research teams with varying goals and optimizations. For instance, YOLOv6 was developed by Meituan, while YOLOv8 was developed by Ultralytics. In some cases, an earlier version may demonstrate better accuracy than a later version, depending on the specific experimental setup [7]. This comparison helps to clarify the strengths and weaknesses of each version in the context of weapon detection.Previous studies on weapon detection, such as the one in [8], often rely on datasets sourced from the internet, which may not accurately represent the characteristics of real-world images captured by surveillance systems in Thailand. For example, Figure 1 (e) and (f) show knives with shapes that are locally specific to Thailand, highlighting the differences in weapon design. Additionally, environmental factors, such as background complexity, can affect detection performance. This study aims to bridge these gaps by using real-world weapons and locations within Thailand, combined with YOLOv8, to develop a weapon detection system specifically tailored to the Thai context.

2. Related Work

Recent deep learning research in object detection, specifically in convolutional neural networks (CNNs), emphasizes various technologies. A study introduced an ensemble approach, combining multiple CNN models for plant classification across mulberry leaves, tomato leaf diseases, and corn leaf diseases datasets. Experiments investigated parameters like data augmentation, the number of models, and voting methods. Results revealed the weighted average approach in CNN voting methods as the most effective, with performance variations across datasets and ensemble methods [9].

In reference [10], the authors combined explainable AI techniques and convolutional neural networks (CNNs) to identify helmetless motorcycle riders. Employing the GRAD-CAM method for insight into the CNN's decision-making, they benchmarked performance against three baseline classifiers. Model evaluation, based on accuracy and F1-Score metrics, revealed that the CNN outperformed all models, achieving the highest F1-Score of 0.8326. Introducing the YOLOv5-Aircraft model, an enhanced version of YOLOv5, with improvements in calibration, loss function, and feature extraction. Experiments, using images from Google Earth and the Vaihingen dataset, showcased the model's effectiveness in aircraft detection, achieving a higher mean Average Precision (mAP) value of 85.25% compared to the original YOLOv5's 81.51%, demonstrating significant performance enhancement [11].

A modified version of YOLOv4, known as YOLO-SA (You Only Look Once-Small Attention), was introduced by the authors in reference [12]. This version, YOLO-SA, was used for landslide detection in satellite remote sensing images. A comparative analysis against 11 other object detectors showed that YOLO-SA outperformed its counterparts with the highest Average Precision (AP) of 0.9408 and a superior frame rate of 42. In contrast, the standard YOLOv4 exhibited an AP of 0.6560 and a frame rate of 10.

In [13], YOLOv8 is applied for rip current segmentation, with practical implications for real-world applications, especially in developing beach safety systems. The dataset includes 2,466 images from 17 videos. Experimenting with scaled versions (n, s, m, l, x), the unexpected finding was that the nano (n) version outperformed, contrary to expectations. This outcome is attributed to the dataset's specific characteristics.

In [14], the authors conducted experiments using YOLOv5, YOLOv7, and YOLOv8 to detect forest fires from images captured by UAVs. They reported that YOLOv8, specifically the scaled nano (n) version, achieved the highest mAP50-95.

Weapon detection remains a critical issue in modern surveillance and security systems. Shah et al. conducted a comprehensive review of various weapon detection methodologies in their study [15]. They highlighted multiple approaches to detecting weapons in images but noted that most of these studies relied on datasets sourced from the internet. This reliance raises concerns about the dataset’s ability to represent the challenging scenarios encountered in real-world situations.

Narejo et al. [16] explored the effectiveness of traditional Convolutional Neural Networks (CNNs), YOLOv2, and YOLOv3 in weapon detection tasks. Their dataset was also compiled from Google Images. Their findings revealed that YOLOv3 achieved the highest accuracy among the methods tested.

Grega et al. [17] identified the limitations of CCTV surveillance systems, which often suffer from poor quality, blurriness, and low resolution. They observed that weapons carried by perpetrators are typically visible only for a limited time within a scene. To address this, they designed an alarm system to assist human operators rather than fully replace them. Their primary objective was to minimize false alarms (achieve high precision), even at the expense of potentially missing some weapon-carrying events (low recall). They reasoned that an excess of false alarms would lead operators to ignore the alerts, rendering the system ineffective. Their proposed system combined various image processing techniques to extract features from CCTV frames and used a Support Vector Machine (SVM) classifier to determine whether a weapon was present. The system triggered an alarm only if a weapon was detected in consecutive frames surpassing a predefined temporal threshold.

Bhatti et al. [18] compared different techniques for detecting pistols, specifically in CCTV footage and public datasets, excluding knives. Their evaluation included sliding window techniques paired with CNN classifiers and object detection methods such as YOLOv3. Their results

indicated that YOLOv4 outperformed the other methods in terms of both accuracy and inference time.

Pullakandam et al. [8] focused on weapon detection using YOLOv8, incorporating quantization techniques to reduce inference time. Their dataset was sourced from the internet, including platforms like Google and YouTube. They reported an inference time of 7.6ms for the quantized YOLOv8 model and 9ms for the non-quantized version. However, they did not specify the sub-versions of YOLOv8 (n, s, m, l, x) used, nor did they provide examples of the images collected from the internet.

3. YOLO Architectures

Redmon et al. introduced the You Only Look Once (YOLO) model in 2015 [19]. YOLO is a one-stage object detection model, in contrast to two-stage object detection models like Faster R-CNN. A one-stage object detector simultaneously performs classification and object localization within a single stage, while two-stage object detection models first propose regions of interest in the initial stage and then classify these regions in the subsequent stage [20].

YOLO can quickly identify objects in an image, including their types and locations [6]. It uses GoogleNet as its base network rather than VGG-16, as GoogleNet offers significantly faster processing with comparable accuracy. This design choice enables YOLO to perform real-time detection while maintaining high accuracy [21]. Subsequent versions of YOLO, including YOLOv2, YOLOv3, YOLOv4, YOLOv5, YOLOv6, YOLOv7, and YOLOv8, have built upon the original model. A summary of the key characteristics of YOLOv5 to YOLOv8 is provided in Table 1.

The YOLO framework excels due to simplicity, few parameters, and fast inference, ideal for real-time applications. It starts by dividing the input image into a grid, often 7x7, following the original YOLO paper. YOLO predicts class confidence scores and attributes for multiple bounding boxes in each grid cell. In the original YOLO setup, there are 2 bounding boxes per grid cell, each with coordinates (x, y), width, height, and a confidence score indicating object presence [6].

In mathematical terms, the dimension of YOLO's output tensor is formulated as $S \times S \times (B \times 5 + C)$, where $S \times S$ represents the number of grid cells, B denotes the number of bounding boxes per grid cell, and C signifies the number of distinct object classes [22].

The YOLO architecture consists of three main components: the backbone for feature extraction, the neck for feature aggregation, and the head for object detection. The head includes subnetworks for tasks such as localization and classification, sometimes incorporating auxiliary objectives like segmentation or pose estimation

Table 1 Summary of YOLOv5, v6, v7 and v8 architectures [22]

Version	Date	Backbone	Framework	Anchor
YOLOv5	2020	YOLOv5CSPDarknet	PyTorch	Yes
YOLOv6	2022	EfficientRep	Pytorch	No
YOLOv7	2022	YOLOv7Backbone	Pytorch	No
YOLOv8	2023	YOLOv8CSPDarknet	PyTorch	No

A. YOLOv5

YOLOv5, introduced in 2020 and implemented in PyTorch by Ultralytics, embodies a three-tiered architecture, comprising the backbone, neck, and head components, as discussed earlier. Ultralytics has introduced the AutoAnchor algorithm for the dynamic adjustment of anchor boxes, which is integrated into the YOLOv5 model.

YOLOv5 can perform tasks beyond object detection such as instance segmentation. YOLOv5 incorporates data augmentation techniques such as Mosaic. The benchmarking of YOLOv5 is performed against the MS COCO dataset, achieving an Average Precision (AP) score of 50.7% [22]. YOLOv5 is also user-friendly and flexible to use [23].

B. YOLOv6

Li et al. presented their technical report on YOLOv6, which follows an architectural structure consisting of three primary components, namely the backbone, neck, and head, akin to the YOLOv5 framework. The backbone of YOLOv6 is the EfficientRep, a modified version of RepVGG [22].

Beyond architectural modifications, YOLOv6 introduces several enhancements, encompassing label assignments, loss functions, distillation strategies, and a quantization scheme. Li et al. provide eight scaled models offering a balance between speed and accuracy, suitable for diverse industrial applications across different scenarios [22][23]. The model's performance is evaluated against the MS COCO Dataset, where it achieves an Average Precision (AP) score of 57.2% at a framerate of 29 FPS. YOLOv6 surpasses previous state-of-the-art models in terms of both accuracy and speed metrics [22].

C. YOLOv7

YOLOv7 was introduced in the year 2022. The authors of YOLOv7 are the same as YOLOv4. YOLOv7 introduces two significant modifications, focusing on network architecture and the integration of novel bag-of-freebies techniques [24]. The architectural adjustments in YOLOv7 encompass the implementation of the Extended efficient layer aggregation network (E-ELAN) and a model scaling approach for concatenation-based models. The E-ELAN strategy enhances the model's efficiency in learning, while the model scaling technique adapts the architecture to the computational resources available [22] [23]. In an evaluation against the MS COCO dataset, the YOLOv7-E6 attains an Average Precision (AP) score of 55.9% while operating at a framerate of 50 frames per second [22].

D. YOLOv8

In 2023, Ultralytics, the same company responsible for the development of YOLOv5, introduced YOLOv8 [22], representing the latest version within the YOLO algorithm series [13]. YOLOv8, akin to its predecessor, offers a range of five scaled versions, commencing with the nano (n) to the extra-large (x) model. YOLOv8 can perform tasks beyond object detection such as pose estimation, segmentation, and object detection. In an evaluation against the MS COCO dataset, the YOLOv8x model attains an Average Precision (AP) score of 53.9% while achieving a framerate of 280 frames per second [22].

E. Performance Evaluation

The right metrics must be used for each issue in order to assess an object detector's performance. Drawing a bounding box around each thing that is detected in a picture makes object detection a particularly difficult problem. Equations (1) through (3) display some of the most popular measures for assessing detection performance, including precision, recall, and mAP [25].

$$Precision = \frac{TP}{TP+FP} \quad (1)$$

$$Recall = \frac{TP}{TP+FN} \quad (2)$$

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i \quad (3)$$

A true positive (TP) is when an object is correctly identified, while a false positive (FP) occurs when the network wrongly identifies an object. False negatives (FN) are objects missed by the network. The IoU method calculates overlap between predicted and ground truth bounding boxes, and detection correctness is determined by comparing IoU with a threshold. Specifying this threshold is crucial, as different IoU values yield various average precision (AP) metrics.

mAP serves as a concise measure of accuracy in object detection tasks. It's obtained by averaging AP values across all classes in the dataset, as depicted in (3). AP is calculated by computing the area under these precision-recall curves [26]. Precision-Recall curves are plotted by varying confidence thresholds for each class.

4. Experimental Setup and Results

A. Schematic Framework

Data collection precedes preprocessing with RoboFlow. Annotators label weapon locations, and the dataset is split into training, validation, and test sets. YOLOv5, 6, 7, and 8 train on the first two sets, and their performances are evaluated on the test set. The study assesses deep learning models in weapon detection systems.

B. Data Collection

The raw data consists of 1920x1080 video recordings from five standard webcams connected to USB ports. The actors comprised five individuals. To ensure processing consistency, both training set images and CCTV footage were standardized to 640x640 before inputting into the YOLO model. This resizing minimizes the impact of resolution differences between webcams and CCTV cameras. The footage includes actors simulating weapon handling, capturing gestures and movements associated with knives, handguns, and shotguns. Additionally, to provide negative examples for object detection algorithms, the actors occasionally carried unrelated objects or simply nothing, diversifying the dataset for better machine learning results. Data collection occurred on July 22, 2023, at the Child Development Center in Salaya, Phutthamonthon District, Nakhon Pathom Province, Thailand. To avoid misinterpretation of the acting as a real threat, precautionary measures were taken by informing the local police and municipality through a letter, seeking approval.

The collected dataset presents several challenges for object detection. One significant challenge is the detection of weapons that are far from the camera, making them small and difficult to identify, as illustrated in Figure 1 (d) Additionally, some weapons, such as knives, are naturally small, which further complicates detection when they are distant from the camera, as shown in Figure 1 (b). These challenges emphasize the need for robust detection algorithms capable of identifying small and distant objects within the surveillance footage.

C. Data Preprocessing

This study utilized videos from a single camera, with data preprocessing managed by the RoboFlow platform. From uploaded videos, 1,468 sampled images were extracted at a frame rate of 5 images per second. Data annotators labeled each image, identifying three classes of bounding boxes: class 0 for handguns, class 1 for knives, and class 2 for shotguns, as shown in Figure 1 (a), (b) and (c).

a b

c d

e f

Fig. 1 Examples of labeling knives, handguns, and shotguns in images.

After labeling, Roboflow randomly divided the dataset into training (70%), validation (20%), and test (10%) sets. The specific numbers are 1027 images for training, 294 for validation, and 147 for testing. Each set includes .jpg images and accompanying text files in YOLO object annotation format, specifying bounding box details for each object.

D. Experimental Setup

Our study experimented with six YOLO models: YOLOv5n, YOLOv6_lite_s, YOLOv6n, YOLOv7_tiny, YOLOv7, and YOLOv8n, training and evaluating them using respective command-line interface tools from their GitHub repositories. Some YOLO version has sub-versions ('n', 's', 'm', 'l', 'x'). We chose the smallest sub-version, 'n', for real surveillance deployment prioritizing processing speed. For edge computing environments, we used optimized sub-versions (YOLOv6 lite and YOLOv7 tiny), along with the smallest general-use sub-versions (YOLOv6n and YOLOv7). While larger versions (such as 'x') may increase accuracy, they do so at the expense of longer inference times, as shown in [8] which may be less suitable for real-time applications, such as the weapon detection system examined in this study.

In our experiments, we employed the following hardware specifications: CPU - Intel(R) Core(TM) i9-10980XE 3.00 GHz, RAM - 64.0 GB, and the graphics card - Nvidia Geforce RTX 2080 Ti with 11GB of memory.

All models were trained with default hyperparameters, except for setting 300 epochs and a batch size of 4. The image size for training and evaluation was 640 x 640. Our reporting of training hyperparameters follows the format described in reference [27].

E. Model Evaluations

We assessed object detection models on validation and test sets using command-line interface tools from their GitHub repositories. For consistency, Intersection Over Union (IoU) threshold for Non-Maximum Suppression (NMS), and confidence threshold hyperparameters were set to 0.65 across all models during evaluation. Results are presented similarly to the reference format. [13], with additional details on parameters and inference time shown in the Table 2.

In this study, we evaluated all versions of YOLO and summarized their performance metrics, including mAP, precision, recall, and inference time, as shown in Table 2. Among the versions tested, YOLOv8n demonstrated the highest overall performance, achieving an mAP50 of 0.848 and an mAP50-95 of 0.503 on the test set. It also outperformed the other models on the validation set. Due to its superior results, we have chosen to provide additional details specifically for YOLOv8n, including its confusion matrix (Table 3) and mAP for each class (Table 4). Additionally, we present the loss and performance metric curves (training and validation classification loss, precision, recall, etc.) in Figure 2. The learning curves behave as expected, with the training classification loss decreasing rapidly during the first 100 epochs and then gradually slowing down. This focus allows for a deeper analysis of the model that shows the greatest potential for our application.

Table 2 Results obtained from experiments on the validation and test dataset

	#parameters	mAP@.5	mAP@.5:.95	Precision	Recall	Inference time (CPU)	Inference time (GPU)
Results obtained from experiments on the validation dataset							
yolov5n	2.5M	0.857	0.5	0.955	0.746	75.0ms	**2.4ms**
yolov6_lite_s	0.52M	0.39	0.221	0.323	0.66	96.26ms	3.16ms
yolov6n	4.63M	0.705	0.377	**0.975**	0.55	**45.10 ms**	2.72ms
yolov7_tiny	6M	0.839	0.461	0.919	0.729	89.6ms	2.8ms
yolov7	36M	0.841	0.468	0.928	0.723	356.5ms	7.2ms
yolov8n	3M	**0.874**	**0.527**	0.947	**0.777**	75.0ms	2.7ms
Results obtained from experiments on the test dataset							
yolov5n	2.5M	0.83	0.512	0.934	0.703	74.3 ms	**3.4ms**
yolov6_lite_s	0.52M	0.292	0.192	0.323	0.5	95.69 ms	4.87ms
yolov6n	4.63M	0.678	0.378	0.642	0.74	**45.70 ms**	4.41ms
yolov7_tiny	6M	0.81	0.453	0.874	0.725	81.6 ms	4.3ms
yolov7	36M	0.807	0.443	0.875	0.683	392.0 ms	7.5ms
yolov8n	3M	**0.848**	**0.503**	**0.936**	**0.741**	75.8 ms	3.5ms

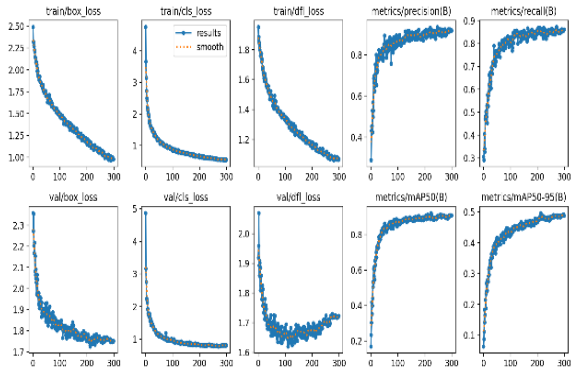

Fig. 2 Training and Validation Losses and Performance Metrics of YOLOv8n for Weapon Detection (Handgun, Knife, Shotgun) over 300 Epochs

Table 3 Experimental results of YOLOv8n on both the validation and test sets.

class	#images	#instances	Precision	Recall	mAP50	mAP50..95
Results (Validation Set)						
All	294	428	0.947	0.777	0.874	0.527
Handgun	294	126	0.951	0.77	0.874	0.452
Knife	294	161	0.954	0.64	0.807	0.449
Shotgun	294	141	0.935	0.922	0.941	0.679
Results (Test Set)						
All	147	207	0.936	0.741	0.848	0.503
Handgun	147	74	0.966	0.77	0.875	0.469
Knife	147	73	0.898	0.603	0.762	0.419
Shotgun	147	60	0.944	0.85	0.907	0.622

Table 4 Confusion Matrix of YOLOv8n on both the validation and test sets.

		True Label			
		Handgun	Knife	Shotgun	Background
predicted	Validation				
	Handgun	95	0	0	2
	Knife	1	101	0	4
	Shotgun	0	0	123	4
	Background	30	60	18	0
	Test				
	Handgun	57	1	0	1
	Knife	1	45	0	2
	Shotgun	0	0	51	1
	Background	16	27	9	0

The Confusion Matrix (Table 4) highlights greater difficulty in detecting knives, aligning with expectations due to their smaller size. Our experiments involve various weapon types, including short and long knives. Detection accuracy is influenced by size, with larger weapons being easier to detect, especially at a distance. In the test set, mAP50 values indicate shotgun detection is the most accurate, followed by handguns, with knives being the least accurate. YOLOv8n shows the highest mAP50 accuracy, but distinctions from YOLOv5n and YOLOv7 are not substantial. Discrepancies with YOLOv8's COCO dataset results may stem from our dataset's characteristics. Our system prioritizes recall over precision to minimize false negatives, crucial in preventing potential catastrophic incidents.

F. YOLOv8 Gun and Knife Detection Results and Discussion

Fig. 3 YOLOv8n Gun and Knife Detection Results and Discussion.

As shown in Figure 3, our system exhibits some misclassifications, particularly with the 'knife' class. In the validation set's confusion matrix, 161 knife objects are present, with 101 correctly identified (True Positives) and 60 misclassified as 'background' (False Negatives). The scenarios depicted in Figure 3 (a) and (b) provide insight into the nature of these misclassifications.

In scenario (a), the person is holding knives in both hands, but due to the distance from the camera and the small size of the knives in the image, the system failed to detect them, classifying the entire scene as 'background'. This scenario would be challenging even for a human observer. In scenario (b), the system partially corrected itself by detecting a knife in the person's left hand, though it still failed to identify the knife in the right hand.

These misclassifications are often due to the size and distance of the objects from the camera, which reduces the detector's ability to identify them correctly. Although the system struggles with such cases, subsequent frames (e.g., (c) in Figure 3) show successful detection as the object becomes more prominent in the frame. When considering these frames as a single incident, our system demonstrates the potential for accurate detection in dynamic scenes. We plan to address these misclassifications more rigorously in future work by quantifying the system's alert accuracy and improving its performance in challenging scenarios

5. Conclusions

In this study, we focused on identifying the most effective YOLO version in terms of accuracy (mAP) and inference time for weapon detection, specifically within our dataset. Challenges such as small weapon size and distance from the camera contributed to misclassifications and lower mAP scores. Our comparison of the smallest sub-versions of each YOLO iteration revealed that YOLOv8 achieved the highest mAP, with scores of 0.874 on the validation set and 0.848 on the test set.

Previous studies on weapon detection, particularly those using YOLOv8, such as [11], often rely on weapon images sourced from the internet, which may not accurately represent real-world environments in Thailand, where factors like angles, lighting conditions, and background complexity can vary significantly. Additionally, some weapons may be unique to Thailand, such as the local knives shown in Figure 3 (f) and (g). This study addresses these gaps by utilizing weapons and locations from real-world environments in Thailand, combined with YOLOv8, to develop a weapon detection system tailored specifically for Thailand.

For future work, we plan to explore the deployment of our system in real-world environments and integrate it into existing surveillance systems. A preliminary study deploying the system on an edge computing device (Jetson Nano Developer Kit (4GB)), as shown in Figure 3 (h), demonstrated that it could process video streams at 5-6 frames per second. We also intend to evaluate our system using additional metrics beyond the standard object detection metric, such as mAP. metrics like alert accuracy may offer a more comprehensive assessment of the system's effectiveness in real-world surveillance applications.

In real-world environments, where factors such as different angles, lighting conditions, and backgrounds can vary significantly, we expect the system to encounter challenges and make errors. To address this, we plan to record video footage of these scenarios, particularly focusing on frames where incorrect detections occur. This data will be used to retrain the models, enhancing their performance and long-term reliability over time.

Acknowledgements

The National Research Council of Thailand (NRCT) is helping to fund some of this research. The reviewers' insightful criticism and recommendations, which helped the writers improve their presentation, are also gratefully acknowledged by the authors.

References

[1] M. E. Atik, Z. Duran and R. Ozgunluk, “Comparison of YOLO versions for object detection from aerial images,” *International Journal of Environment and Geoinformatics*, vol. 9, no. 2, pp. 87-93, Jun. 2022, doi: 10.30897/ijegeo.1010741.

[2] Y. Liu, P. Sun, N. Wergeles and Y. Shang, “A survey and performance evaluation of deep learning methods for small object detection,” *Expert Systems with Applications*, vol. 172, no. 2021, pp. 114602: 1-14, Jun. 2021, doi: 10.1016/j.eswa.2021.114602.

[3] Budi, Yasmine Syifa Nabila, et al. "Daycare shooting in Thailand: time to address firearm violence as a global health issue." Public health challenges 2.2 (2023): e93.

[4] M. M. Fernandez-Carrobles, O. Deniz and F. Maroto, “Gun and knife detection based on faster R-CNN for video surveillance,” In: Morales, A., Fierrez, J., Sánchez, J., Ribeiro, B. (eds) Pattern Recognition and Image Analysis. IbPRIA 2019. *Lecture Notes in Computer Science*(), vol. 11868, Springer, 2019 Cham, pp. 441-452, Sep. 2019, doi: 10.1007/978-3-030-31321-0_38.

[5] H. Hu, C. Tang, C. Shi and Y. Qian, “Detection of residual feed in aquaculture using YOLO and mask RCNN,” *Aquacultural Engineering*, vol. 100, pp. 102304: 1 -8, Feb. 2023, doi: 10.1016/j.aquaeng.2022.102304.

[6] P. Jiang, D. Ergu, F. Liu, Y. Cai and B. Ma, “A review of Yolo algorithm developments,” *Procedia Computer Science*, vol. 199, pp. 1066-1073, 2022, doi: 10.1016/j.procs.2022.01.135.

[7] Li, Chuyi, et al. "Yolov6 v3. 0: A full-scale reloading." *arXiv preprint arXiv:2301.05586* (2023).

[8] M. Pullakandam, K. Loya, P. Salota, R. M. R. Yanamala and P. K. Javvaji, “Weapon object detection using quantized YOLOv8,” *2023 5th International Conference on Energy, Power and Environment: Towards Flexible Green Energy Technologies (ICEPE)*, Shillong, India, 2023, pp. 1-5, Jun. 2023, doi: 10.1109/ICEPE57949.2023.10201506.

[9] T. Chompookham and O. Surinta, “Ensemble methods with deep convolution neural networks for plant leaf recognition, *ICIC Express Letters*, vol. 15, no. 6, pp. 553-565, Jun. 2021, doi: 10.24507/icicel.15.06.553.

[10] S. Intagorn, S. Pinitkan, M. Panmuang and C. Rodmorn, “Helmet detection system for motorcycle riders with explainable artificial intelligence using convolutional neural network and Grad-CAM,” In: Surinta, O., Kam Fung Yuen, K. (eds) Multi-disciplinary Trends in Artificial Intelligence. *MIWAI 2022. Lecture Notes in Computer Science()*, vol. 13651, Springer, Cham, pp. 40-51, Nov. 2022, doi: 10.1007/978-3-031-20992-5_4.

[11] S. Luo, J. Yu, Y. Xi and X. Liao, “Aircraft target detection in remote sensing images based on improved YOLOv5,” *IEEE Access*, vol. 10, pp. 5184-5192, Jan. 2022, doi: 10.1109/ACCESS.2022.3140876.

[12] L. Cheng, J. Li, P. Duan and M. Wang, “A small attentional YOLO model for landslide detection from satellite remote sensing images,” *Landslides*, vol. 18, no. 8, pp. 2751-2765, May 2021, doi: 10.1007/s10346-021-01694-6.

[13] A. Dumitriu, F. Tatui, F. Miron, R. T. Ionescu and R. Timofte, “Rip current segmentation: A novel benchmark and YOLOv8 baseline results,” *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops 2023*, pp. 1261-1271.

[14] X. Jia, Y. Wang and T. Chen, “Forest fire detection and recognition using YOLOv8 algorithms from UAVs images,” *2023 IEEE 5th International Conference on Power, Intelligent Computing and Systems (ICPICS)*, Shenyang, China, pp. 646-651, Jul. 2023, doi: 10.1109/ICPICS58376.2023.10235675.

[15] Shah, Syed Atif Ali, Mahmoud Ahmad Al-Khasawneh, and M. Irfan Uddin. "Review of weapon detection techniques within the scope of

street-crimes." *2021 2nd International Conference on Smart Computing and Electronic Enterprise (ICSCEE)*. IEEE, 2021.
[16] Narejo, Sanam, et al. "Weapon detection using YOLO V3 for smart surveillance system." Mathematical Problems in Engineering 2021.1 (2021): 9975700.
[17] Grega, Michał, et al. "Automated detection of firearms and knives in a CCTV image." *Sensors* 16.1 (2016): 47.
[18] Bhatti, Muhammad Tahir, et al. "Weapon detection in real-time cctv videos using deep learning." *Ieee Access* 9 (2021): 34366-34382.
[19] Redmon, Joseph, et al. "You only look once: Unified, real-time object detection." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.
[20] Kang, Junhyung, et al. "A survey of deep learning-based object detection methods and datasets for overhead imagery." IEEE Access 10 (2022): 20118-20134.
[21] J. Du, "Understanding of object detection based on CNN family and YOLO," *Journal of Physics: Conference Series*, *2nd International Conference on Machine Vision and Information Technology (CMVIT 2018)* 23–25 February 2018, Hong Kong, vol. 1004, Feb. 2018, doi: 10.1088/1742-6596/1004/1/012029.
[22] J. R. Terven and D. M. Cordova-Esparza, "A comprehensive review of YOLO: From YOLOv1 and beyond," *Under Review in ACM Computing Surveys*, 1-27, Apr. 2023.
[23] C. Li, L. Li, H. Jiang, K. Weng, Y. Geng, L. Li, Z. Ke, Q. Li, M. Cheng, W. Nie, Y. Li, B. Zhang, Y. Liang, L. Zhou, X. Xu, X. Chu, X. Wei and X. Wei, "YOLOv6: A single-stage object detection framework for industrial applications," pp. 1-17, Sep. 2022, doi: 10.48550/arXiv.2209.02976.
[24] CY. Wang, A. Bochkovskiy and HY. Mark Liao, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR),* pp. 7464-7475, 2023.
[25] A. Malta, M. Mendes and T. Farinha, "Augmented reality maintenance assistant using YOLOv5," *Applied Sciences*, vol. 11 no. 11, pp. 4758, May 2021, doi: 10.3390/app11114758.
[26] M. Lippi, N. Bonucci, R. F. Carpio, M. Contarini, S. Speranza and A. Gasparri, "A yolo-based pest detection system for precision agriculture," *In 2021 29th Mediterranean Conference on Control and Automation (MED), PUGLIA, Italy, IEEE*, pp. 342-347, Jun. 2021, doi: 10.1109/MED51440.2021.9480344.
[27] A. Aboah, B. Wang, U. Bagci and Y. Adu-Gyamfi, "Real-time multi-class helmet violation detection using few-shot data sampling technique and yolov8." In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, 2023*, pp. 5350-5358.