Applying Deep Learning Models for Weapon Detection in Public Environments Through Serveil-lance Cameras

Dung Nguyen[1], Van-Dung Hoang[2], Van-Tuong-Lan Le[3]

[1] University of Sciences, Hue University, Hue city, Vietnam,
[2] HCMC University of Technology and Education, Ho Chi Minh city, Vietnam,
[3] Hue University, Hue city, Vietnam.

Correspondence: Van-Dung Hoang, dunghv@hcmute.edu.vn
Communication: received 25 Sep 2024, revised 28 Oct 2024, accepted 28 Nov 2024
DOI: 10.32913/mic-ict-research.v2025.n1.1318

***Abstract*: In the context of public security, the ability to detect weapons in real time through surveillance cameras is of utmost importance. This study focuses on applying fine-tuning techniques and data augmentation strategies to a Transformer-based model to enhance the capability of identifying weapons in public environments. The fine-tuning process optimized the model parameters, along with these augmentations, to improve weapon detection performance. Experimental results show that after fine-tuning, the model achieved an mAP0.5 score of up to 96.5%, representing a significant improvement in accuracy compared to previous object detection models. These results demonstrate the great potential for applying the model to real-time security surveillance systems, effectively detecting and addressing threats.**

Keywords: *Transformer, Real time, Weapon detection, Surveillance system*

I. Introduction

Weapon detection in public environments through surveillance cameras is a crucial and challenging task in the field of security. To address this issue, the application of modern deep learning techniques, including Convolutional Neural Networks (CNN) [1–4], and Transformer networks [5], has become a prominent research trend. These methods not only improve object recognition accuracy but also enhance processing capabilities under complex real-world conditions. In the paper [6], the authors developed a dataset for evaluating handgun detection from surveillance camera images, aiming to establish a new standard for weapon detection methods. This dataset supports deep learning models for training and testing in complex conditions, enhancing the ability to detect handguns even in crowded environments with high noise levels. However, a major limitation of this dataset is the lack of diversity in the surveillance scenarios and conditions. The handgun detection scenarios are mainly simulated in environments with a certain level of noise and complexity, but it does not cover all real-world situations, such as nighttime conditions or low-light footage.

The paper [7] introduces a multi-level feature pyramid technique for deep learning models, which helps improve the detection of atypical guns in surveillance video. This method enables the system to effectively detect guns even when they have unusual shapes and sizes, overcoming the limitations of traditional object detection methods. However, this technique requires complex computations and may face challenges when deployed in real-time systems, especially in crowded environments or when there are many objects to detect simultaneously. Furthermore, the ability to detect objects in low-light or blurry images remains limited.

In [8], the authors highlight the challenges of real-time gun detection from CCTV, particularly in scenarios with various sources of noise such as lighting changes, complex viewing angles, and crowds. This study recommends developing more robust methods to address these challenges in security surveillance. However, the paper does not provide a complete solution for issues such as handling lighting changes or difficult viewpoints, which means that current models still lack high accuracy in real-world situations.

Finally, the paper [9] presents an anomaly detection method for surveillance video, allowing the system to automatically identify abnormal or dangerous situations. This method contributes to enhancing security by assisting in the detection of abnormal behaviors related to violence or threats. However, one limitation of this approach is the classification of abnormal behaviors, especially in non-dangerous situations. The system may generate false alerts,

particularly when similar behaviors occur in contexts with no clear threat, reducing the effectiveness and reliability of the system.

Current research and methods have made significant contributions to the development of weapon and anomaly detection models in public surveillance. However, a major weakness of these models is their generalization ability in real-world scenarios, especially under complex conditions such as changing lighting, difficult viewpoints, or crowded environments. Additionally, current methods have not fully addressed the issue of accuracy in detecting objects in high-noise situations or low-quality footage. Minimizing false alerts in situations without clear threats remains a challenge that must be overcome to improve the efficiency and reliability of surveillance systems.

In this study, we focus on applying Fine-tuning techniques [10–16] to several object detection models, including Transformer-based models like RT-DETR [17] and some versions of YOLO [18–26]. RT-DETR, a variant of DETR [27], has been fine-tuned on the Weapon dataset to optimize weapon detection capabilities in public environments. Additionally, we conducted fine-tuning on YOLO, a method well-known for its speed and effectiveness in object detection. RT-DETR combines the strengths of Transformer and CNN, effectively handling spatial features and enhancing performance in real-world conditions. Meanwhile, YOLO offers fast and accurate detection capabilities. By fine-tuning both models, we expect to significantly improve the accuracy and detection capabilities of surveillance systems, thereby contributing to the enhancement of public security measures.

II. RELATED WORKS

Object Detection. In the field of computer vision, object detection is a crucial and complex task, requiring models not only to recognize objects but also to determine their locations within an image. Object detection methods are generally classified into two main categories: two-stage models and one-stage models.

Two-stage models, such as R-CNN and its variants [28–30], operate through the following process: **(1) Region Proposal Stage:** The model identifies regions in the image that are likely to contain objects, typically using a Region Proposal Network (RPN) [30]; **(2) Classification and Localization Stage:** After obtaining the proposed regions, the model classifies each region and precisely locates the objects. Two-stage models often achieve high accuracy because the detection and classification steps are clearly separated. However, the downside is slower processing speed, making it challenging to meet real-time requirements.

One-stage models, such as YOLO [18–26], SSD [31], and RetinaNet [32], are designed to detect and classify objects in a single step. These models usually have a simpler network structure, allowing for quick predictions of object positions and types directly from feature points. The biggest advantage of one-stage models is their fast speed, making them suitable for real-time applications. However, their accuracy may be lower compared to two-stage models, especially in complex situations.

Transformer-based Models. Recently, Transformer-based models such as DETR [27] and its variants [17, 33, 34] have emerged as modern solutions, combining high performance with real-time processing capabilities. DETR is one of the pioneering models that apply Transformers to the object detection task. DETR uses a Transformer encoder and decoder to directly predict bounding boxes and object labels from the input image, eliminating the need for region proposal extraction as seen in two-stage models. The architecture of DETR consists of the following key components: **(1) Backbone:** The backbone is typically a deep convolutional network, such as ResNet [35], responsible for extracting features from the input image. These features are then passed through several convolutional and pooling layers to reduce their size while enriching the spatial information; **(2) Encoder:** The encoder takes in the features from the backbone and applies a multi-head attention mechanism to model the relationships between different feature points. The encoder helps extract relevant information from the entire image, unrestricted by the size of feature regions; **(3) Decoder:** The decoder uses object queries, which are learnable embeddings, to predict the bounding boxes and labels of objects. The decoder also applies attention mechanisms to focus on regions with a high probability of containing objects, leading to more accurate predictions; **(4) Hungarian Matching:** DETR employs the Hungarian algorithm to optimize the matching of predicted bounding boxes with actual objects in the image. This approach avoids issues such as duplication or missed objects, which are common in traditional methods; **(5) Set Prediction Loss:** DETR employs a composite loss function that combines bounding box losses (L1 loss and GIoU loss) with classification label loss (cross-entropy loss). The set prediction loss allows the model to assign labels to objects more accurately and efficiently. However, DETR faces several drawbacks, including: long training times, poor performance with small objects, high computational resource requirements, and slow optimization on small datasets. Figure 1 illustrates the structure of the DETR model.

RT-DETR Model. RT-DETR [17] is an enhanced version of DETR, specifically optimized for real-time appli-

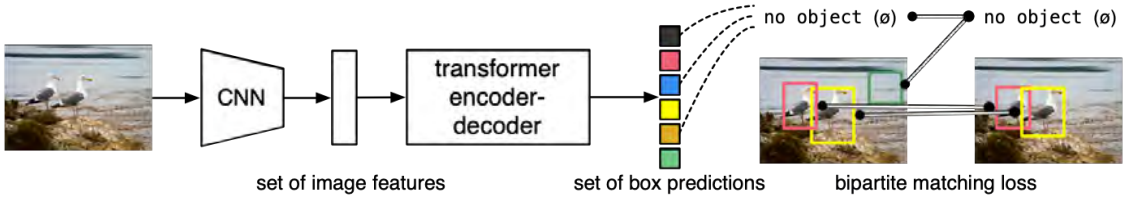

Figure 1. DETR model architecture [27]

cations. It maintains the advantages of the Transformer architecture while improving processing speed without compromising accuracy, thanks to a lighter and more efficient design. RT-DETR inherits many characteristics from DETR but is optimized for performance: **(1) Attention mechanism optimization:** The attention mechanism in RT-DETR is optimized to reduce computational complexity. Instead of applying attention over the entire image space, RT-DETR uses several strategies to offload computation, such as hierarchical attention or region-based attention; **(2) Reduction in the number of queries:** RT-DETR reduces the number of queries in the decoder, which helps speed up inference without significantly affecting the model's accuracy. The number of queries is adjusted to cover all objects in the image without wasting computational resources; **(3) Lighter backbone network:** Instead of using complex and heavy backbones like ResNet-101 [35], RT-DETR often uses lighter backbones like ResNet-50 [35] or even MobileNet [36, 37] to increase inference speed while still ensuring detection performance. With these improvements, the RT-DETR model achieves notable efficiency: **(1) High processing speed:** RT-DETR is designed to meet real-time requirements, capable of processing tens to hundreds of frames per second, depending on hardware configuration and input size; **(2) High accuracy:** Despite being optimized for speed, RT-DETR still maintains high accuracy in detecting and classifying objects; **(3) Generalization ability:** With the Transformer-based attention mechanism, RT-DETR can handle complex scenes and difficult-to-recognize objects, a challenge for models purely based on CNNs.

Fine-Tuning Technique. Fine-tuning is a powerful technique in deep learning, particularly useful when working with pre-trained models on large datasets. Fine-tuning is typically applied in scenarios where the target dataset is small or specific compared to the original dataset, or when the new task is related but distinct from the one the original model was trained on. The fine-tuning process generally begins with a model pre-trained on a large, common dataset like ImageNet [38] or COCO [39]. The main steps in the fine-tuning process include: **(1) Using the pre-trained model:** The model is loaded along with the weights learned from training on a large dataset. These weights help the model capture general features of images; **(2) Freezing the initial layers:** The early layers of the model, where general features are learned, are usually kept unchanged and not re-trained. This helps preserve the general features learned from the original dataset. Specifically, the model freezes at layer S3, meaning that the layers before S3 will remain unchanged, while the subsequent layers (from S4 onwards) will be fine-tuned to adapt to the new task; **(3) Training the later layers:** The final layers of the model, where more complex features are learned, are re-trained on the new dataset. This allows the model to adjust and learn the specific features of the new task; **(4) Tuning hyperparameters.** During fine-tuning, hyperparameters like learning rate, batch size, and optimization function can be adjusted to optimize training on the new data. Benefits of fine-tuning: **(1) Saves time and resources:** Since the model already has foundational knowledge from previous training, fine-tuning doesn't require learning from scratch, reducing both training time and computational resources; **(2) Improves performance:** Fine-tuning helps the model better adapt to the target data, improving accuracy and generalization in practical applications; **(3) Flexible application:** Fine-tuning can be applied to a variety of tasks, from image classification and object detection to natural language processing, broadening the applicability of deep learning models.

III. WEAPON DETECTION METHOD

1. IMPLEMENTATION DETAIL

The workflow of the model begins with processing the input images using data augmentation techniques such as flipping, rotating, and color adjustments to increase the diversity of the training set. After augmentation, the images are fed into the Backbone, which has been pre-trained on the ImageNet dataset [38], to extract essential features. Additionally, we use a model pre-trained on the COCO dataset [39] to initialize the parameters. The model is then fine-tuned on the dataset, as described in Section III.2.

The features are then passed through two key components: AIFI (Attention-based Intra-scale Feature Interac-

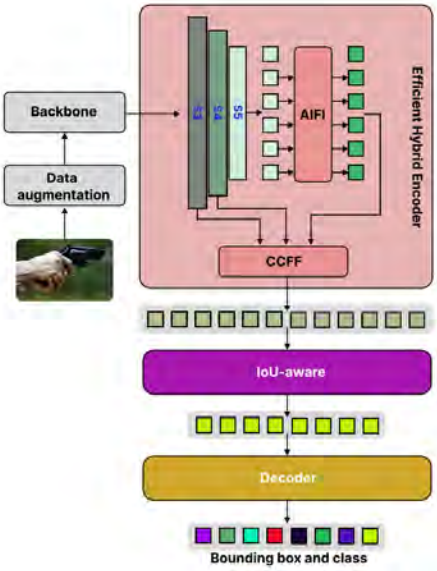

Figure 2. Structure of the model

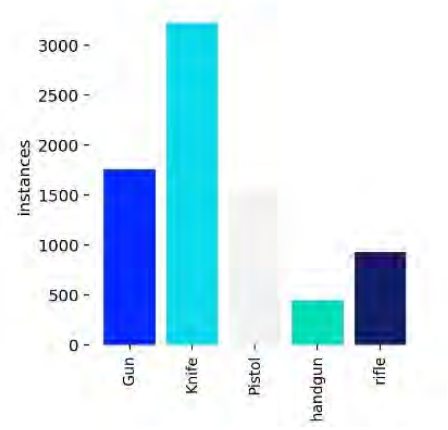

Figure 3. Object distribution by label in the training set

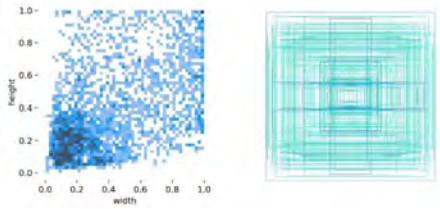

Figure 4. The ratio of bounding box area to frame area in the training set

tion) and CCFF (CNN-based Cross-scale Feature Fusion) for fusion. AIFI utilizes attention mechanisms to interact with features within the same scale, focusing on important information at each level of object detail, enhancing the key features, especially for weapon detection. CCFF combines features from multiple scales, ensuring the model accurately detects objects with varying sizes and positions in the image.

After processing through AIFI and CCFF, the features are merged into a unified representation, then passed through the IoU-Aware module to improve accuracy in predicting object locations, and finally through the Decoder to predict the labels and locations of weapons in public surveillance environments. Figure 2 illustrates the structure of the model.

2. DATASET

The Weapon Dataset [40] is a dataset containing images of various types of weapons in public environments. This dataset provides images along with labels describing the type of weapon and its location. The dataset is divided into two sets: a training set and a testing set. Figure 3 illustrates the distribution of the number of samples for objects in the training dataset.

From the chart, it can be observed that: Knife is the most prevalent object in the dataset, with over 3,000 samples, significantly surpassing other objects; Gun and Pistol have relatively similar numbers of samples, with around 1,500 samples for Gun and approximately 1,200 for Pistol; Handgun has the fewest samples in the dataset, with only around 500; Rifle has a moderate number of samples, with about 1,000. The large disparity in the number of samples between objects may impact the model's performance, particularly for objects with fewer samples, such as Handgun, which may be harder to detect accurately.

To address this imbalance, Focal Loss [32] has been employed during the training process. This loss function is particularly effective in scenarios where the dataset is imbalanced by assigning higher weights to less frequent objects. By focusing more on these underrepresented classes, Focal Loss enhances the model's ability to detect objects such as Handgun and Rifle, which have fewer samples, thereby improving overall classification accuracy and robustness in real-world applications. This technique ensures that the model does not overly favor the detection of frequent objects like Knife, contributing to a more balanced and effective weapon detection system.

Figure 4 illustrates the ratio of object bounding box areas to the areas of the images containing them in the training set. Based on the distribution of bounding box sizes relative to the frame, we observe that most detected objects are small, with widths and heights predominantly below 0.2 of the overall frame size. The high density of values in this region indicates that the model faces challenges in detecting small objects. This emphasizes the importance of fine-tuning the model to enhance sensitivity and accuracy

in detecting small objects, reducing missed or misidentified detections.

3. DATA AUGMENTATION

Data augmentation techniques are an important method in the field of deep learning, used to expand and enrich the training dataset. By creating variations of the original data, such as resizing, flipping, shifting, and adjusting colors, this technique helps the model learn to recognize objects under different conditions, thereby improving the model's generalization capability. The application of data augmentation techniques not only helps reduce overfitting but also enhances the model's performance in real-world scenarios. In this study, data augmentation techniques such as scaling, horizontal flipping, shifting, and adjusting colors in the HSV space, as described in Table I, have been used to improve the training efficiency of the RT-DETR model on the Weapon dataset.

4. MODEL TRAINING

To train the model on the Weapon dataset, the RT-DETR model was initialized from a pre-trained version on the COCO dataset. The final layers of the model were then adjusted to be compatible with the number of classes in the Weapon Dataset. The fine-tuning technique was applied by adjusting the model's parameters, as described in Table II, ensuring that the model learns the specific features of the new dataset. The AdamW optimizer was used with an appropriate learning rate, optimizing the training process without causing overfitting.

IV. EXPERIMENTS AND RESULTS

1. TRAINING MACHINE SPECIFICATIONS

In this experiment, the model was implemented on a device with the configuration described in Table III.

2. EVALUATION OF TRAINING RESULTS

Common criteria used to evaluate the performance of object detection algorithms include IoU, Precision, Recall, and mAP. Detailed definitions are listed below.

A. IoU: Intersection over Union (IoU) is calculated by taking the area of intersection between the predicted region (A) and the ground truth region (B) and dividing it by the total area of the union of these two regions. The formula can be expressed as follows:

$$IoU = \frac{A \cap B}{A \cup B} \tag{1}$$

The IoU value ranges from 0 to 1. A higher value indicates better model accuracy. Specifically, a lower numerator indicates inaccurate predictions of the ground truth region, while a larger denominator suggests that the predicted region is wider, leading to a lower IoU value.

B. Precision: Precision represents the ratio of correctly predicted samples among the set of predicted samples. It can be expressed as follows:

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

where:

- TP (True Positive): The number of correctly predicted samples
- FP (False Positive): The number of incorrectly predicted samples

C. Recall: Recall represents the ratio of correctly predicted samples among the total set of actual samples. It can be expressed as follows:

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

where:

- TP (True Positive): The number of correctly predicted samples
- FN (False Negative): The number of samples that are incorrectly predicted and not predicted at all

D. mAP: Average Precision (AP) is a measure of Precision values at different thresholds on the Precision-Recall (PR) curve, and is calculated as a weighted average. Mean Average Precision (mAP) is the average value of AP across all classes. Specifically, mAP@0.5 indicates the mAP when IoU is 0.5, and mAP@[0.5:0.95] represents the mean mAP when IoU ranges from 0.5 to 0.95.

$$mAP = \frac{1}{N} \sum_{i=1}^{N} \mathrm{AP}_i \tag{4}$$

where:

- AP: The area under the Precision-Recall (PR) curve calculated at different thresholds
- N: The number of classes

3. TRAINING RESULTS

Figure 5 illustrates the training process of the model through the error function values over epochs. Specifically, the GIoU loss during training starts at 0.55605 and fluctuates slightly across epochs, indicating that the model is gradually improving its localization ability. The Cls loss starts high (around 2.6170) and decreases over time, demonstrating that the model is learning to classify better. The L1 loss fluctuates within a narrow range, from 0.44467

Table I
A FEW DATA AUGMENTATION TECHNIQUES

Techniques	Value	Description
scale	0.5	This technique resizes the image by scaling it up or down by a certain ratio. With a factor of 0.5, the image will be reduced to 50% of its original size. This helps the model learn to recognize objects at different sizes, thus improving the model's generalization ability
fliplr	0.5	This technique flips the image horizontally with a 50% probability. This creates variations of the original image by inverting it along the horizontal axis. Horizontal flipping helps the model learn to recognize objects from different angles, potentially improving recognition in situations where the object may appear from different directions
translate	0.1	This technique shifts the image horizontally and vertically by a certain percentage. With a factor of 0.1, the image can be shifted by up to 10% of the original size in both directions. This helps the model learn to recognize objects even when they are not centered in the image, thereby improving recognition in various scenarios
hsv_h	0.015	This technique changes the hue of the image in the HSV (Hue, Saturation, Value) color space. With a factor of 0.015, the hue of the image can change by ±1.5% of its original hue value. Changing the hue helps the model learn to recognize objects under different lighting and color conditions, thereby enhancing the model's generalization capability
hsv_s	0.7	This technique adjusts the saturation of the image in the HSV color space. With a factor of 0.7, the saturation can vary by ±70% of the original saturation value. This helps the model learn to recognize objects under different color conditions, ranging from highly saturated colors to more muted tones
hsv_v	0.4	This technique changes the brightness (value) of the image in the HSV color space. With a factor of 0.4, the brightness of the image can vary by ±40% of its original brightness value. Changing the brightness helps the model recognize objects under different lighting conditions, including both low-light and high-light environments

Table II
TRAINING HYPERPARAMETERS

Parameter	Value	Description
Epoch	50	The number of epochs refers to how many times the entire training dataset is used to train the model
Learning rate	0.00001	The learning rate determines the extent to which the model's weights are adjusted after each update step based on the loss function
Batch size	4	Batch size is the number of data samples used in each step of weight updates for the model
Image size	640 x 640	Input image size for the model. All images will be resized to 640 pixels x 640 pixels before being fed into the model
Backbone pre-trained	ImageNet	The backbone is the main part of the deep learning model, and in this case, the model uses the ResNet network. The backbone was pre-trained on the ImageNet dataset, a large dataset with various types of images and object classes
Model Pre-trained	COCO	The object detection model was pre-trained on the COCO dataset, a common dataset for object detection tasks
Optimizer	AdamW	AdamW is a variant of the Adam optimization algorithm, with the addition of Weight Decay to control overfitting and improve training performance
Max-Det	300	This is the maximum number of objects the model can detect in an image
Mometum	0.937	Momentum is a parameter in the optimization algorithm that helps improve the model's convergence speed by maintaining part of the gradient direction from previous update steps
Weight decay	0.0005	Weight decay is a technique to reduce overfitting by adding a penalty to the model's weight values

Table III
A FEW SPECIFICATIONS OF THE TRAINING MACHINE

Specifications	Value
Python version	3.9
PyTorch	1.10
CUDA	11.1
GPU	Quadro RTX 4000, 7783MiB

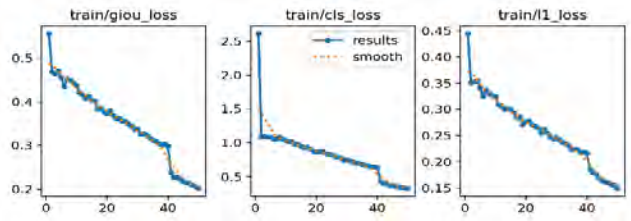

Figure 5. Training results based on Loss Functions

to 0.34084, reflecting stability in matching the predicted positions.

The three charts in Figure 6, 7, 8 provide a comprehensive view of the model's performance through precision, recall, and F1-score metrics across various confidence thresholds. The Precision-Confidence chart shows that the model's precision increases as confidence rises, with the highest value reaching 1.00 at a confidence threshold of around 0.917, indicating excellent model performance at high confidence levels. Meanwhile, the Recall-Confidence chart reveals that recall decreases as confidence increases,

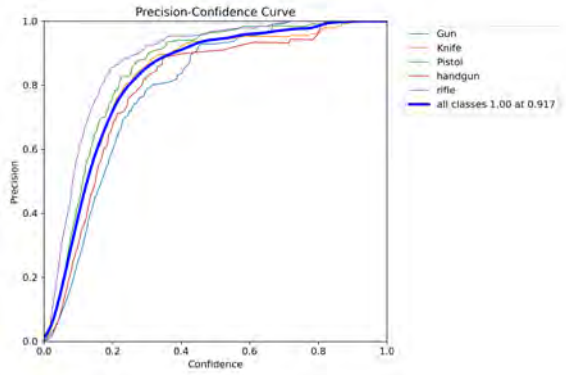

Figure 6. Precision-Confidence Curve

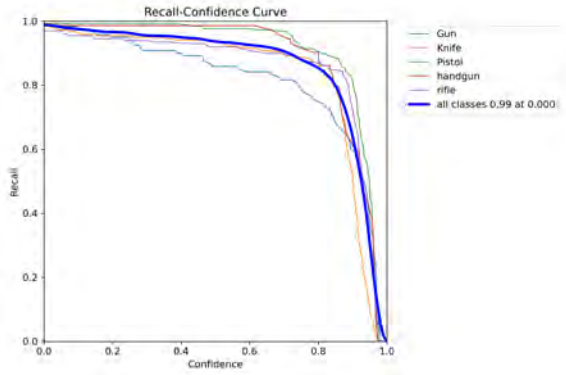

Figure 7. Recall-Confidence Curve

meaning the model overlooks more true positives at higher confidence thresholds. Finally, the F1-Confidence chart peaks at a threshold of 0.58 with an F1-score of 0.94, reflecting a balance between precision and recall at this level. Overall, the model exhibits high precision with higher confidence but has to trade off recall and F1-score as confidence continues to increase.

From Figure 9, the Precision-Recall chart shows the relationship between the model's precision and coverage across different weapon classes. The model performs exceptionally well, with an mAP@0.5 value of 0.965 (96.5%) across all classes. Classes such as Knife, Pistol, Handgun, and Rifle exhibit both high precision and recall, indicating that the model effectively detects these objects. The Gun class has slightly lower precision, at 0.947, but still maintains good performance. This demonstrates that the model operates effectively across various object types, with high accuracy and coverage.

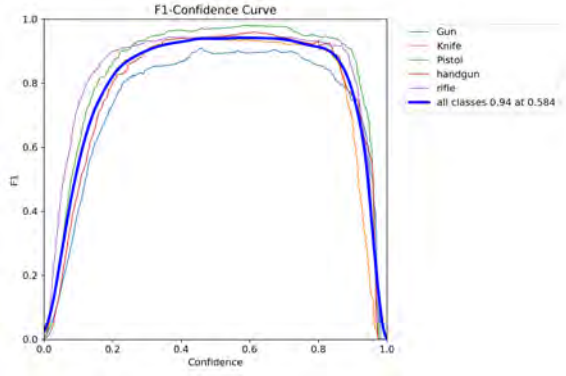

Figure 8. F1-Confidence Curve

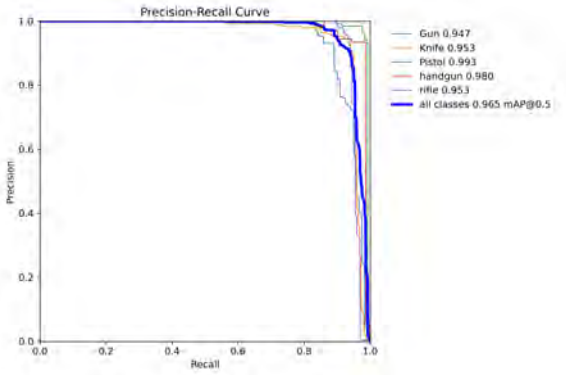

Figure 9. Precision-Recall Curve

Figure 10 illustrates the confusion matrix of the model after training, showing the classification performance across different object categories. The confusion matrix indicates that the model classifies weapons with fairly good performance for the main categories such as Gun, Knife, and Pistol, with high accuracy rates of 96%, 96%, and 99%, respectively. However, there is a slight confusion between different types of weapons, such as confusion between gun and background (29%) and between knife and background (30%). These errors suggest that the model struggles to distinguish objects that may visually resemble weapons. On the other hand, the background category has the highest confusion rate with all weapon types, indicating that the model still needs improvement in detecting non-weapon environments.

Figure 11, 12 illustrates the model's prediction results on several images from the test set.

Table IV compares the performance and efficiency of the YOLOv8, YOLOv9, YOLOv10, RT-DETR, Faster RCNN + FPN, and UGR models on a specific dataset. For small models, YOLOv8-n, YOLOv9-t, and YOLOv10-n achieve a good balance between accuracy and inference speed. Among them, YOLOv9-t stands out by reaching a mAP@0.5 of 85.5% and mAP@0.5-0.95 of 64.3%, while maintaining an inference time of 18.9ms. YOLOv8-n offers a faster inference time of 9.8ms but has a lower mAP@0.5-0.95 of 61.1%. For large models, YOLOv8-l, YOLOv9-c,

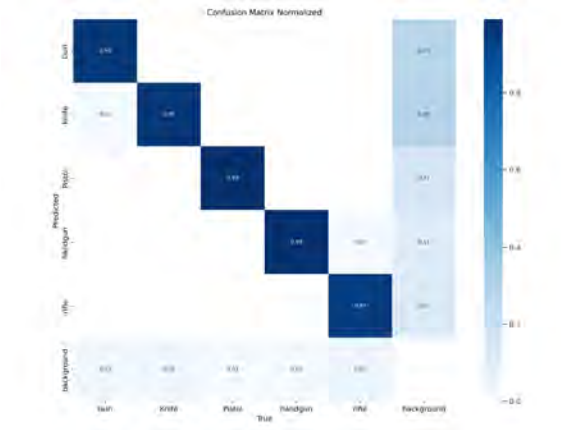

Figure 10. Confusion Matrix on the Test Dataset

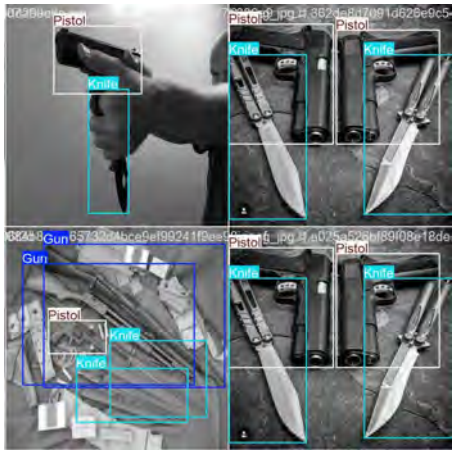

Figure 11. Ground-Truth

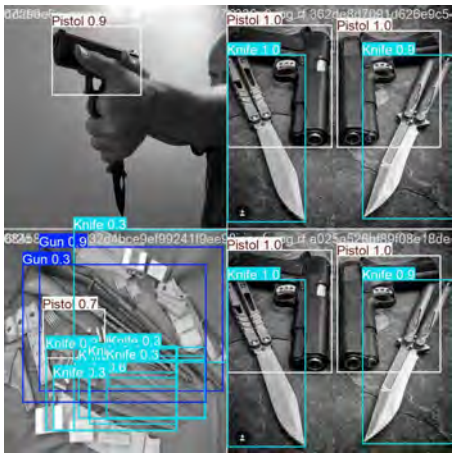

Figure 12. Predicted

and YOLOv10-l require higher computational resources. Notably, YOLOv8-l, with the highest GFlops, achieves a mAP@0.5-0.95 of only 64.0%, which is lower compared to the smaller, more efficient models. In contrast, RT-DETR-l excels in terms of accuracy, achieving a mAP@0.5 of 96.5% and mAP@0.5-0.95 of 79.7%, although it has a slower inference time of 45.1ms. Compared to Faster RCNN + FPN and UGR, RT-DETR-l outperforms both in accuracy, with Faster RCNN + FPN achieving 91.7% mAP@0.5 and UGR reaching 92.6% mAP@0.5. However, both models have lower mAP@0.5-0.95 values (68.3% for UGR), indicating that RT-DETR-l provides better generalization across detection thresholds. This comparison highlights that while RT-DETR-l demands more computational resources, it is the most accurate model, making it highly suitable for tasks that prioritize precision.

V. Conclusions

In this paper, we conducted fine-tuning of the RT-DETR model on the Weapon dataset to improve the detection of weapons in public environments through surveillance cameras. The experimental results show that the fine-tuned RT-DETR model achieved a mAP@0.5 score of up to 96.5%, demonstrating high effectiveness in detecting and classifying weapon objects. This indicates that RT-DETR can accurately and consistently recognize objects in various situations, enhancing the reliability of security surveillance systems. The use of fine-tuning significantly improved the model's performance compared to the original version, especially in real-world conditions where there is significant variation in the shapes and sizes of objects. These results not only affirm the feasibility of RT-DETR in challenging object detection tasks but also open up new avenues for applying Transformer models in surveillance and security systems.

In the future, several improvements and follow-up research can be pursued to further enhance the effectiveness of weapon detection models: (1) Dataset expansion: Expanding the dataset with more diverse weapon types and scenarios can improve the model's accuracy, especially under different environmental conditions. This expansion could also include gathering data from various sources to increase the model's generalization capability; (2) Hyperparameter tuning: Further tuning the hyperparameters of the RT-DETR model, such as the size of the Transformer network and training parameters, could help achieve better performance and reduce training time; (3) Combining with other techniques: Researching the combination of RT-DETR with other deep learning techniques, such as machine learning augmentation methods or lighter deep learning models, could improve inference speed and reduce

Table IV
TRAINING HYPERPARAMETERS

Model	Layers	Parameters	GFlops	Epoch	mAP@0.5	mAP@0.5-0.95	Inference time
YOLOv8-n	255	03.2M	8.90	50	85.5%	61.1%	09.8ms
YOLOv9-t	917	02.0M	7.90	50	85.5%	64.3%	18.9ms
YOLOv10-n	385	02.8M	8.70	50	80.3%	59.3%	28.6ms
Faster RCNN + FPN [9]	N/A	N/A	N/A	N/A	91.7%	N/A	N/A
UGR [8]	N/A	N/A	N/A	N/A	92.6%	68.3%	N/A
YOLOv8-l	365	43.7M	165.7	50	84.9%	64.0%	52.2ms
YOLOv9-c	618	25.5M	104.0	50	85.0%	63.7%	45.0ms
YOLOv10-l	628	25.9M	127.9	50	81.5%	63.5%	42.9ms
RT-DETR-l	**673**	**33.0M**	**108.0**	50	96.5%	**79.7%**	**45.1ms**

computational resource requirements; (4) Application in real-world scenarios: Deploying the model in real-world security surveillance systems and evaluating its performance in live environments could help identify existing issues and improve the model based on the specific requirements of the application. Continued research and development in these areas will not only improve weapon detection technology but also contribute to the overall advancement of more intelligent and efficient security surveillance systems.

ACKNOWLEDGEMENT

This research is funded by Vietnam National Foundation for Science and Technology Development (NAFOSTED) under grant number 102.05-2021.04.

REFERENCES

[1] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.

[3] K. Simonyan, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[4] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[5] A. Vaswani, "Attention is all you need," *Advances in Neural Information Processing Systems*, 2017.

[6] S. Yellapragada, Z. Li, K. B. Doshi, P. M. Mhasakar, H. Fan, J. Wei, E. Blasch, B. Zhang, and H. Ling, "Cctv-gun: Benchmarking handgun detection in cctv images," *arXiv preprint arXiv:2303.10703*, 2023.

[7] J. Lim, M. I. Al Jobayer, V. M. Baskaran, J. M. Lim, J. See, and K. Wong, "Deep multi-level feature pyramids: Application for non-canonical firearm detection in video surveillance," *Engineering applications of artificial intelligence*, vol. 97, p. 104094, 2021.

[8] J. L. S. González, C. Zaccaro, J. A. Álvarez-García, L. M. S. Morillo, and F. S. Caparrini, "Real-time gun detection in cctv: An open problem," *Neural networks*, vol. 132, pp. 297–308, 2020.

[9] W. Sultani, C. Chen, and M. Shah, "Real-world anomaly detection in surveillance videos," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 6479–6488.

[10] C. B. Do and A. Y. Ng, "Transfer learning for text classification," *Advances in neural information processing systems*, vol. 18, 2005.

[11] L. Torrey and J. Shavlik, "Transfer learning," in *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*. IGI global, 2010, pp. 242–264.

[12] L. Zhao, S. Pan, E. Xiang, E. Zhong, Z. Lu, and Q. Yang, "Active transfer learning for cross-system recommendation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 27, no. 1, 2013, pp. 1205–1211.

[13] K. Weiss, T. M. Khoshgoftaar, and D. Wang, "A survey of transfer learning," *Journal of Big data*, vol. 3, pp. 1–40, 2016.

[14] N. Agarwal, A. Sondhi, K. Chopra, and G. Singh, "Transfer learning: Survey and classification," *Smart Innovations in Communication and Computational Sciences: Proceedings of ICSICCS 2020*, pp. 145–155, 2021.

[15] A. Hosna, E. Merry, J. Gyalmo, Z. Alom, Z. Aung, and M. A. Azim, "Transfer learning: a friendly introduction," *Journal of Big Data*, vol. 9, no. 1, p. 102, 2022.

[16] Z. Lin, D. Liu, W. Pan, and Z. Ming, "Transfer learning in collaborative recommendation for bias reduction," in *Proceedings of the 15th ACM Conference on Recommender Systems*, 2021, pp. 736–740.

[17] Y. Zhao, W. Lv, S. Xu, J. Wei, G. Wang, Q. Dang, Y. Liu, and J. Chen, "Detrs beat yolos on real-time object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 16 965–16 974.

[18] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 779–788.

[19] J. Redmon and A. Farhadi, "Yolo9000: Better, faster, stronger," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 6517–6525.

[20] J. Redmon, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[21] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *arXiv preprint arXiv:2004.10934*, 2020.

[22] G. Jocher, A. Stoken, J. Borovec, L. Changyu, A. Hogan, L. Diaconu, F. Ingham, J. Poznanski, J. Fang, L. Yu *et al.*, "ultralytics/yolov5: v3. 1-bug fixes and performance improvements," *Zenodo*, 2020.

[23] C. Li, L. Li, Y. Geng, H. Jiang, M. Cheng, B. Zhang, Z. Ke, X. Xu, and X. Chu, "Yolov6 v3. 0: A full-scale reloading," *arXiv preprint arXiv:2301.05586*, 2023.

[24] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-

time object detectors," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 7464–7475.
[25] C.-Y. Wang, I.-H. Yeh, and H.-Y. Mark Liao, "Yolov9: Learning what you want to learn using programmable gradient information," in *European Conference on Computer Vision*. Springer, 2025, pp. 1–21.
[26] A. Wang, H. Chen, L. Liu, K. Chen, Z. Lin, J. Han, and G. Ding, "Yolov10: Real-time end-to-end object detection," *arXiv preprint arXiv:2405.14458*, 2024.
[27] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European conference on computer vision*. Springer, 2020, pp. 213–229.
[28] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 580–587.
[29] R. Girshick, "Fast r-cnn," in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1440–1448.
[30] 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 & Machine Intelligence*, vol. 39, no. 06, pp. 1137–1149, June 2017.
[31] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*. Springer, 2016, pp. 21–37.
[32] T.-Y. Ross and G. Dollár, "Focal loss for dense object detection," in *proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2980–2988.
[33] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, "Deformable detr: Deformable transformers for end-to-end object detection," *arXiv preprint arXiv:2010.04159*, 2020.
[34] D. Meng, X. Chen, Z. Fan, G. Zeng, H. Li, Y. Yuan, L. Sun, and J. Wang, "Conditional detr for fast training convergence," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 3651–3660.
[35] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
[36] A. G. Howard, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.
[37] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4510–4520.
[38] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.
[39] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*. Springer, 2014, pp. 740–755.
[40] J. Assalim, "Fireguns and knifes detector dataset," https://universe.roboflow.com/joo-assalim/fireguns-and-knifes-detector, nov 2023, visited on 2024-11-29. [Online]. Available: https://universe.roboflow.com/joo-assalim/fireguns-and-knifes-detector

Dung Nguyen was born on June 13, 1988 in Thua Thien Hue. He graduated with a bachelor's degree in information technology from University of Sciences, Hue University in 2010. In 2013, he graduated with a master's degree in computer science from University of Sciences, Hue University. Currently he works at University of Sciences, Hue University. Research fields: Software technology, artificial intelligence, machine learning, deep learning, databases.
Email: nguyendung@hueuni.edu.vn

Van-Dung Hoang received a Ph.D. degree in Electrical and Computer Engineering from University of Ulsan, Ulsan city, Korea, in 2015. After a year of experience with Telecom SudParis as a Postdoctoral Research Fellow, he joined the Faculty of Information Technology, HCMC University of Technology and Education, Ho Chi Minh City, Vietnam, where he is currently serving as a professor of Artificial Intelligence Department. His research interests cover a wide area, which focuses on computer vision, robotics, medical image processing, autonomous vehicles, and ambient intelligence.
Email: dunghv@hcmute.edu.vn

Van-Tuong-Lan Le was born on November 10, 1974, in Thua Thien Hue. In 1996, he graduated with a degree in Mathematics and Informatics from the College of Sciences, Hue University. He obtained a Master's degree in Information Technology from Hanoi University of Science and Technology in 2002 and earned a Ph.D. in Computer Science from the College of Sciences, Hue University, in 2018. He is currently working at the Department of Training and Student Affairs, Hue University.
Email: lvtlan@hueuni.edu.vn