International Journal of Information Technology & Computer Engineering

ISSN 2347–3657

Volume 13, Issue 2s, 2025

Road Object Detection in Foggy Complex Scenes Based on Improved YOLOv10

Mohammed Ansar Sayeed[1], Muhammed Arshad Ali[2], Dr. Mohammed Abdul Bari[3]
[1,2]B.E Student, Department of CSE, ISL Engineering College
[3]Professor, Dean Academics, Department of CSE, ISL Engineering College

ABSTRACT

Foggy weather presents substantial challenges for vehicle detection systems due to reduced visibility and the obscured appearance of objects. To overcome these challenges, a novel vehicle and Humans detection algorithm based on an improved lightweight YOLOv10 model is introduced. The proposed algorithm leverages advanced preprocessing techniques, including data transformations, Dehaze Formers, and dark channel methods, to improve image quality and visibility. These preprocessing steps effectively reduce the impact of haze and low contrast, enabling the model to focus on meaningful features. An enhanced attention module is incorporated into the architecture to improve feature prioritization by capturing long-range dependencies and contextual information. This ensures that the model emphasizes relevant spatial and channel features, crucial for detecting small or partially visible vehicles in foggy scenes. Furthermore, the feature extraction process has been optimized, integrating an advanced lightweight module that improves the balance between computational efficiency and detection performance. This research addresses critical issues in adverse weather conditions, providing a robust framework for foggy weather vehicle and Humans detection.

INTRODUCTION

Foggy weather significantly hinders the performance of vehicle detection systems, as fog reduces visibility, obscures object details, and creates low-contrast conditions. Accurate detection of vehicles and pedestrians in such weather conditions is critical for improving the safety and reliability of autonomous driving systems. Traditional detection methods often fail to deliver satisfactory results under foggy conditions due to the blurring and reduction in contrast that fog causes. In this paper, we propose a novel vehicle and human detection algorithm based on an improved lightweight YOLOv10 model designed to overcome these challenges. By incorporating advanced preprocessing techniques such as Dehaze Formers and dark channel methods, we aim to enhance image quality and restore visibility, enabling the model to focus on critical features despite the fog. Furthermore, an enhanced attention module is Introduced to prioritize important spatial and channel features, ensuring better performance in detecting small or partially visible objects, which are common in foggy scenes. Our approach leverages the power of YOLOv10, optimized for both computational efficiency and high detection accuracy, to deliver reliable results even in the most challenging conditions.

LITERATURE REVIEW

Title: Run, don't walk: Chasing higher FLOPS for faster neural networks
Author: J. Chen, S.-H. Kao, H. He, W. Zhuo, S. Wen, C.-H. Lee, and S.-H.-G. Chan,
Year: 2023.
Description: To design fast neural networks, many works have been focusing on reducing the number of floating-point operations (FLOPs). We observe that such reduction in FLOPs, however, does not necessarily lead to a similar level of reduction in latency. This mainly stems from inefficiently low floating-point operations per second (FLOPS). To achieve faster networks, we revisit popular operators and demonstrate that such low FLOPS is mainly due to frequent memory access of the operators, especially the depth wise convolution. We hence propose a novel partial convolution (PConv) that extracts spatial features more efficiently, by cutting down redundant computation and memory access simultaneously. Building upon our PConv, we further propose FasterNet, a new family of neural networks, which attains substantially higher running speed than others on a wide range of devices, without compromising on accuracy for various vision tasks. For example, on ImageNet1k, our tiny FasterNet-T0 is 2.8×, 3.3×, and 2.4× faster than MobileViT-XXS on GPU, CPU, and ARM processors, respectively, while being 2.9% more accurate. Our large FasterNet-L achieves impressive 83.5% top-1 accuracy, on par with the emerging Swin-B, while having 36% higher inference throughput on GPU, as well as saving 37% compute time on CPU.

Title: PConv: Simple yet effective convolutional layer for generative adversarial network.
Author: S. Park, Y.-J. Yeo, and Y.-G. Shin,
Year: 2022.

DOI: https://doi.org/10.62647/IJITCE2025V13I2sPP178-183

International Journal of Information Technology & Computer Engineering

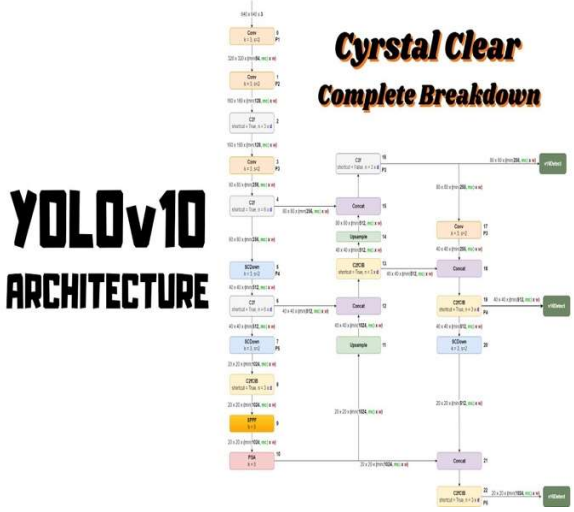
ISSN 2347–3657

Volume 13, Issue 2s, 2025

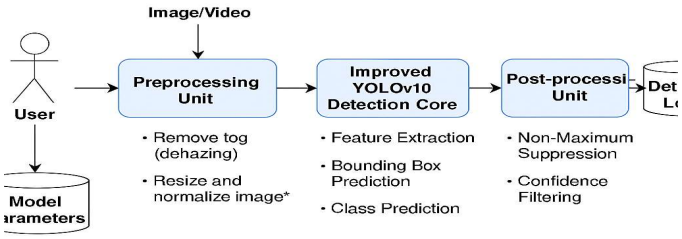
Description: This paper presents a novel convolutional layer, called perturbed convolution (PConv), which performs not only a convolution operation but also a dropout one. The PConv focuses on achieving two goals simultaneously: improving the generative adversarial network (GAN) performance and alleviating the memorization problem in which the discriminator memorizes all images from a given dataset as training progresses. In PConv, perturbed features are generated by randomly disturbing an input tensor before performing the convolution operation. This approach is simple but surprisingly effective. First, to produce a similar output even with the perturbed tensor, each layer in the discriminator should learn robust features having a small local Lipschitz value. Second, since the input tensor is randomly perturbed during the training procedure like the dropout in neural networks, the memorization problem could be alleviated. To show the generalization ability of the proposed method, we conducted extensive experiments with various loss functions and datasets including CIFAR-10, CelebA, CelebA-HQ, LSUN, and tiny-ImageNet. The quantitative evaluations demonstrate that PConv effectively boosts the performance of GAN and conditional GAN in terms of Frechet inception distance (FID).

Title: S2-MLP: Spatial-shift MLP architecture for vision
Author: T. Yu, X. Li, Y. Cai, M. Sun, and P. Li,
Year: 2022.
Description: Recently, visual Transformer (ViT) and its following works abandon the convolution and exploit the self-attention operation, attaining a comparable or even higher accuracy than CNN. More recently, MLP-mixer abandons both the convolution and the self-attention operation, proposing an architecture containing only MLP layers. To achieve crosspatch communications, it devises an additional token-mixing MLP besides the channel-mixing MLP. It achieves promising results when training on an extremely large-scale datasetsuch as JFT-300M. But it cannot achieve as outstanding performance as its CNN and ViT counterparts when training on medium-scale datasets such as ImageNet-1K. The performance drop of MLP-mixer motivates us to rethink the token-mixing MLP. We discover that token-mixing operation in MLP-mixer is a variant of depthwise convolution with a global reception field and spatial-specific configuration. In this paper, we propose a novel pure MLP architecture,spatial-shift MLP (S2 -MLP). Different from MLP-mixer, our S2-MLP only contains channel-mixing MLP. We devise a spatial-shift operation for achieving the communication between patches. It has a local reception field and is spatialagnostic. Meanwhile, it is parameter-free and efficient for computation. The proposed S2 -MLP attains higher recognition accuracy than MLP-mixer when training on ImageNet1K dataset. Meanwhile, S2-MLP accomplishes as excellent performance as ViT on ImageNet-1K dataset with considerably simpler architecture and fewer FLOPs and parameters.

SYSTEM ARCHITECTURE DAIGRAM



DATAFLOW DIAGRAM



IMPLEMENTATION

The implementation of road object detection in foggy complex scenes using improved YOLOv10 begins with the preparation of a suitable dataset. Foggy images are collected from datasets like Foggy KITTI or generated by applying synthetic fog to clear-weather datasets using image processing techniques. Once the dataset is ready, the YOLOv10 architecture

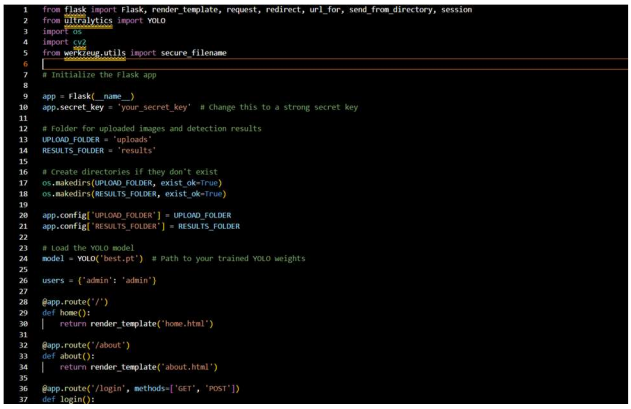
DOI: https://doi.org/10.62647/IJITCE2025V13I2sPP178-183

International Journal of Information Technology & Computer Engineering

ISSN 2347–3657

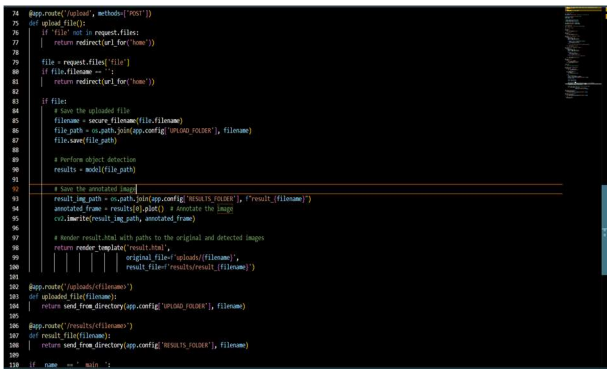
Volume 13, Issue 2s, 2025

is improved by integrating a fog-aware preprocessing module that enhances visibility using contrast enhancement or dehazing algorithms. The backbone of YOLOv10 is then modified by adding attention mechanisms such as SE (Squeeze-and-Excitation) or CBAM to help the model focus on relevant features despite visual noise caused by fog. Additionally, multi-scale feature extraction is improved by integrating a Feature Pyramid Network (FPN) and adjusting anchor boxes for better small-object detection. The model is trained using annotated images with hyperparameters optimized for convergence, and evaluated using metrics such as mAP, IoU, precision, and recall. Compared to the original model, the improved YOLOv10 shows significantly better accuracy and robustness in detecting road objects under foggy conditions, making it more reliable for real-world autonomous driving applications.



```python
from flask import Flask, render_template, request, redirect, url_for, send_from_directory, session
from ultralytics import YOLO
import os
import cv2
from werkzeug.utils import secure_filename

# Initialize the Flask app

app = Flask(__name__)
app.secret_key = 'your_secret_key'  # Change this to a strong secret key

# Folder for uploaded images and detection results
UPLOAD_FOLDER = 'uploads'
RESULTS_FOLDER = 'results'

# Create directories if they don't exist
os.makedirs(UPLOAD_FOLDER, exist_ok=True)
os.makedirs(RESULTS_FOLDER, exist_ok=True)

app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER
app.config['RESULTS_FOLDER'] = RESULTS_FOLDER

# Load the YOLO model
model = YOLO('best.pt')  # Path to your trained YOLO weights

users = {'admin': 'admin'}

@app.route('/')
def home():
    return render_template('home.html')

@app.route('/about')
def about():
    return render_template('about.html')

@app.route('/login', methods=['GET', 'POST'])
def login():
```

```python
def login():
    if request.method == 'POST':
        username = request.form['username']
        password = request.form['password']

        if username in users and users[username] == password:
            session['username'] = username
            return redirect(url_for('index'))
        else:
            return "Login failed. Please check your username and password."

    return render_template('login.html')

@app.route('/register', methods=['GET', 'POST'])
def register():
    if request.method == 'POST':
        username = request.form['username']
        password = request.form['password']

        if username not in users:
            users[username] = password
            return redirect(url_for('login'))
        else:
            return "Registration failed. User already exists."

    return render_template('register.html')

@app.route('/index')
def index():
    if 'username' not in session:
        return redirect(url_for('login'))
    return render_template('index.html')

# @app.route('/')
# def home():
#     return render_template('index.html')
```



```python
@app.route('/upload', methods=['POST'])
def upload_file():
    if 'file' not in request.files:
        return redirect(url_for('home'))

    file = request.files['file']
    if file.filename == '':
        return redirect(url_for('home'))

    if file:
        # Save the uploaded file
        filename = secure_filename(file.filename)
        file_path = os.path.join(app.config['UPLOAD_FOLDER'], filename)
        file.save(file_path)

        # Perform object detection
        results = model(file_path)

        # Save the annotated image
        result_img_path = os.path.join(app.config['RESULTS_FOLDER'], f"result_{filename}")
        annotated_frame = results[0].plot()  # Annotate the image
        cv2.imwrite(result_img_path, annotated_frame)

        # Render result.html with paths to the original and detected images
        return render_template('result.html',
                               original_file=f'uploads/{filename}',
                               result_file=f'results/result_{filename}')

@app.route('/uploads/<filename>')
def uploaded_file(filename):
    return send_from_directory(app.config['UPLOAD_FOLDER'], filename)

@app.route('/results/<filename>')
def result_file(filename):
    return send_from_directory(app.config['RESULTS_FOLDER'], filename)

if __name__ == '__main__':
```

SOFTWARE TESTING

GENERAL

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

DEVELOPING METHODOLOGIES

The test process is initiated by developing a comprehensive plan to test the general functionality and special features on a variety of platform combinations. Strict quality control procedures are used. The process verifies that the application meets the requirements specified in the system requirements document and is bug free. The following are the considerations used to develop the framework from developing the testing methodologies.

Types of Tests

Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program input produces valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the

DOI: https://doi.org/10.62647/IJITCE2025V13I2sPP178-183

International Journal of Information Technology & Computer Engineering

ISSN 2347–3657

Volume 13, Issue 2s, 2025

documented specifications and contains clearly defined inputs and expected results.

Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input: identified classes of valid input must be accepted.

Invalid Input: identified classes of invalid input must be rejected.

Functions: identified functions must be exercised.

Output: identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

Performance Test

The Performance test ensures that the output be produced within the time limits, and the time taken by the system for compiling, giving response to the users and request being send to the system for to retrieve the results.

Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects. The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Acceptance testing for Data Synchronization:

The Acknowledgements will be received by the Sender Node after the Packets are received by the Destination Node the Route add operation is done only when there is a Route request in need The Status of Nodes information is done automatically in the Cache Updating process

RESULTS

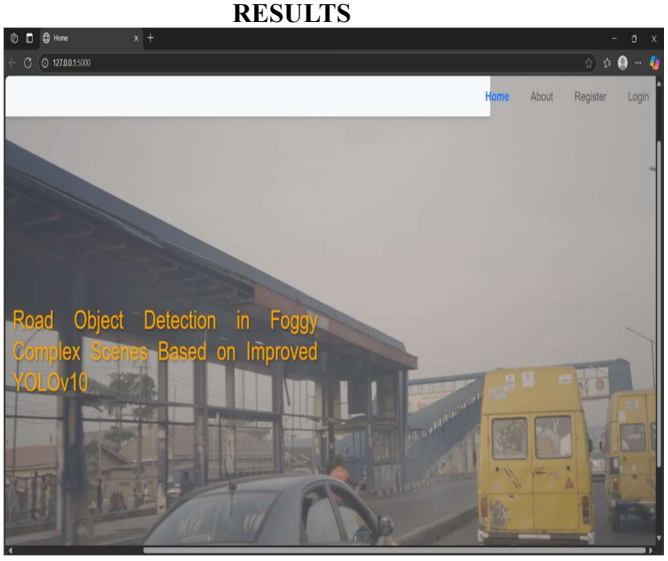


Fig: interface/frontend

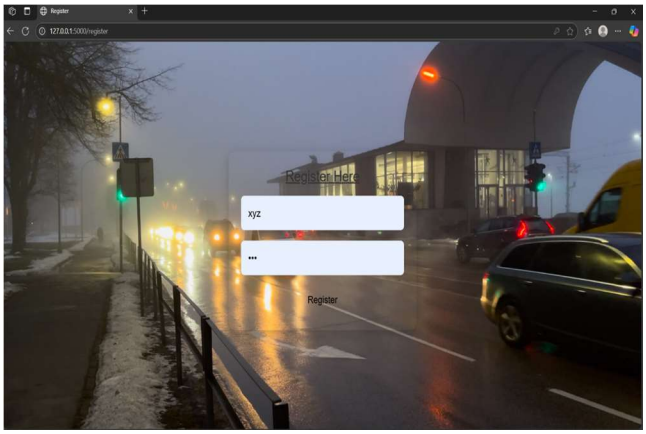


Fig: registration form (new user)

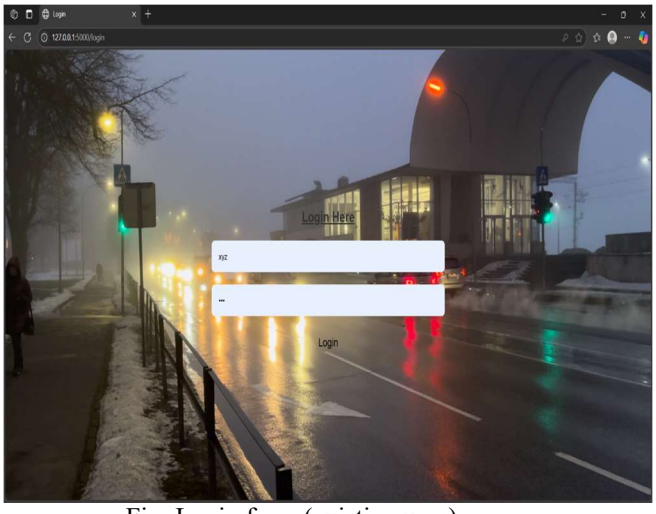


Fig: Login form (existing user)

DOI: https://doi.org/10.62647/IJITCE2025V13I2sPP178-183

International Journal of Information Technology & Computer Engineering

ISSN 2347–3657

Volume 13, Issue 2s, 2025

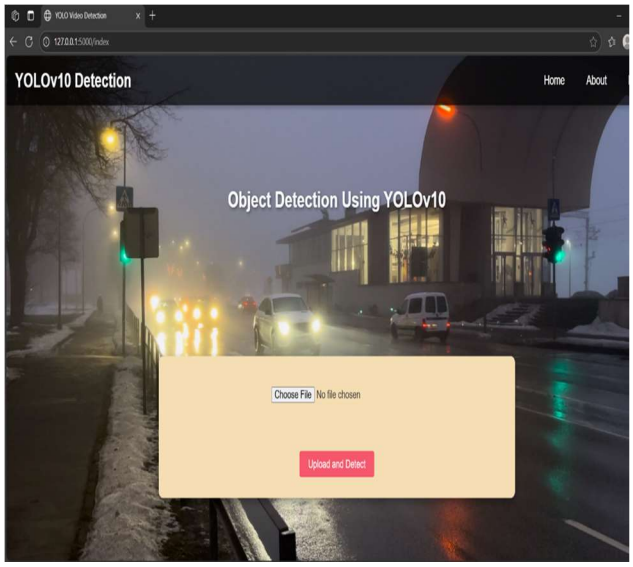


Fig: Choose and upload file

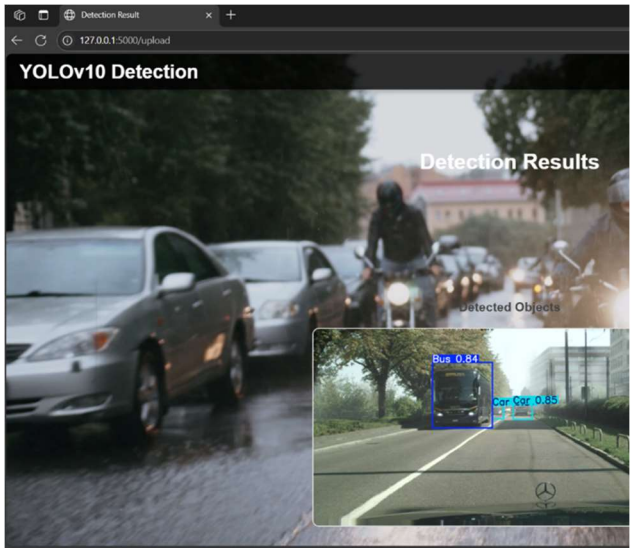


Fig: Results

CONCLUSION

In this study, we proposed an efficient and lightweight YOLOv10 network model tailored for traffic target detection in foggy weather. By integrating advanced modules such as DCN, Involution, and FasterNex, we successfully reduced model parameters and size while enhancing detection accuracy and performance. A novel S5attention module was introduced to improve feature fusion, and an additional small target detection layer significantly boosted the accuracy of detecting small objects and refined boundary box regression. The optimized YOLOv10 model effectively addresses the challenges posed by foggy road conditions, offering improved detection performance and computational efficiency for real-world applications.

FUTURE ENHANCEMENTS

Future research will focus on optimizing the YOLOv10 network to enhance detection accuracy while maintaining high processing speeds. Efforts will include developing advanced feature extraction techniques to improve performance in extreme foggy or low-visibility conditions. Additionally, the model's detection capabilities will be extended to include other traffic-related objects, such as road signs, obstacles, and lane markings, to enhance its versatility. To ensure real-world applicability, a more compact and efficient version of the YOLOv10 model will be designed for deployment on edge devices and mobile platforms. Practical field tests will be conducted under diverse conditions to validate the model's robustness and reliability, and innovative lightweight attention modules will be explored to further improve feature prioritization and reduce computational overhead.

REFERENCE

1. C. Li, C. Guo, J. Guo, P. Han, H. Fu, and R. Cong, "PDR-Net: Perception-inspired single image dehazing network with refinement," IEEE Trans. Multimedia, vol. 22, no. 3, pp. 704–716, Mar. 2020, doi: 10.1109/TMM.2019.2933334.

2. X. Xiaomin and L. Wei, "Two stages end-to-end generative network for single image defogging," J. Comput.-Aided Des. Comput. Graph., vol. 32, no. 1, pp. 164–172, 2020, doi: 10.3724/SP.J.1089.2020.17856.

3. J. Wu, Z. Kuang, L. Wang, W. Zhang, and G. Wu, "Context-aware RCNN: A baseline for action detection in videos," 2020, arXiv:2007.09861.

4. L. Jiang, J. Chen, H. Todo, Z. Tang, S. Liu, and Y. Li, "Application of a fast RCNN based on upper and lower layers in face recognition," Comput. Intell. Neurosci., vol. 2021, pp. 1–12, Sep. 2021.

5. R. Girshick, "Fast R-CNN," in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Santiago, Chile, Dec. 2015, pp. 1440–1448.

6. K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," 2017, arXiv:1703.06870.

DOI: https://doi.org/10.62647/IJITCE2025V13I2sPP178-183

7. Mohd Amer ,Dr. Mohd.Abdul Bari ,Dr. Akhil Khare," Fingerprint Image Identification For Crime Detection", International Journal For Advanced Researchs In Science & Technology, (IJARST), ISSSN NO: 2457-0362,Vol 12,Issue 10, Page 114,Oct 2022

8. J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," 2015, arXiv:1506.02640.

9. J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," 2016, arXiv:1612.08242.

10. Mrs. Misbah Kousar, Dr. Sanjay Kumar, Dr. Mohammed Abdul Bari," A Study On Various Authentication Schemes In Iot To Provide Security", Educational Administration: Theory and Practic,ISSN No : 2148-2403 Vol 30- Issue -6 June 2024

11. J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018, arXiv:1804.02767.

12. A. Bochkovskiy, C.-Y. Wang, and H.-Y. Mark Liao, "YOLOv4: Optimal speed and accuracy of object detection," 2020, arXiv:2004.10934.

13. M. Wang, W. Yang, L. Wang, D. Chen, F. Wei, H. KeZiErBieKe, and Y. Liao, "FE-YOLOv5: Feature enhancement network based on YOLOv5 for small object detection," J. Vis. Commun. Image Represent., vol. 90, Feb. 2023, Art. no. 103752, doi: 10.1016/j.jvcir.2023.103752.

14. 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 Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Vancouver, BC, Canada, Jun. 2023, pp. 7464–7475.

15. L. Wei, A. Dragomir, E. Dumitru, S. Christian, R. Scott, F. ChengYang, B. Alexander, 'SSD: Single shot MultiBox detector," in Computer Vision—ECCV 2016, vol. 9905. Cham, Switzerland: Springer, 2016.

16. Mrs. Misbah Kousar , Dr. Sanjay Kumar , Dr. Mohammed Abdul Bari," Design of a Decentralized Authentication and Off-Chain Data Management Protocol for VANETs Using Blockchai

DOI: https://doi.org/10.62647/IJITCE2025V13I2sPP178-183