Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 05, May-2025

A Novel Flame and Smoke Detection Algorithm Based on ODCONVBS-YOLOV5X6

Shaik Irfan
Dept. of Computer Science (AI&ML)
Vignana Bharathi Institute of Technology,
Hyderabad, India

Rathod Maheshwar
Dept. of Computer Science (AI&ML)
Vignana Bharathi Institute of
Technology, Hyderabad, India

Sunkoju Aryan Sriraj
Dept. of Computer Science
(AI&ML) Vignana Bharathi Institute
of Technology,
Hyderabad, India

R.V Sai Chaitanya
Dept. of Computer Science
(AI&ML) Vignana Bharathi Institute
of Technology, Hyderabad, India

Soma Manish Goud
Dept. of Computer Science
(AI&ML) Vignana Bharathi Institute
of Technology,
Hyderabad, India

Abstract: **This study presents an enhanced flame and smoke detection technique utilizing "YOLOv5s, such as ODConvBS (ordinary Convolutional Blocks with Spatial attention)" to optimize the extraction of salient features from the convolutional kernel. The model also integrates Gnconv in the Neck to beautify the extraction of high-order spatial information. Conventional algorithms for flame and smoke detection are plagued by low accuracy, improved pass over costs, diminished detection efficiency, and inadequate performance in identifying small objects, resulting in considerable losses during hearth- associated events. The enhanced YOLOv5s model proven a massive growth in "mean average precision (mAP)" whilst using a dataset of flame and smoke. The accuracy rate, recall price, and detection pace skilled significant improvements, hence. The counseled set of rules rectifies the deficiencies of traditional flame and smoke detection techniques while demonstrating full-size performance enhancements, rendering it a possible answer for actual-time and precise fire detection. The use of innovative factors within the version structure enhances feature extraction, resulting in improved detection performance regarding accuracy, recall, and velocity. The mission complements its performance by integrating sophisticated detection algorithms with YOLO v5x6 and YOLOv8, with YOLO v5x6 attaining significant metrics of "79.2% mAP, 74% recall, and 80.6% precision".**

"Index terms - YOLOv5s, object detection, Gnconv, attention mechanism, ODConvBS".

1. INTRODUCTION

The fire will cause sufficient damage to human lives and assets in everyday activities and also prevents healthy improvement of society. However, the flames
can be easily turned off during the first level of the fire disaster. Therefore, it is in a hurry and properly detecting flames and smoke can be reduced by fire damage to maintain everyday level of production. By regularly detecting the initial flame, the data on flame and smoke are collected through various temperature sensors, smoke sensors and light sensors to determine the impact on the stove. The sensor's installation role, effective range, outside mild, and ambient humidity considerably impact the accuracy of flame and smoke detection. Item detection involves identifying and categorizing all target items inside a picture, representing an essential task within the domain of pc vision. Currently, item detection strategies are labeled into classes: two-stage and One-level. The two-stage layout produces pre-defined containers that potentially incorporate objects for detection and extraction through features, followed by class and regression localization. The two-level architecture removes the necessity for generating pre-selected packing containers and might immediately extract characteristics within the network to be expecting each the classification and vicinity of an object.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 05, May-2025

As imaginary and presentation algorithms and hardware of computers develop, they mainly developed deep learning techniques to detect flames and smoke improved traditional manual strategies, allowing models to remove other abstract and intense functions from better generalization than traditional processes. In 2010 Fizzi et al.] Deep learning beds Complete the task of Flame and smoke detection consists of three components: classification (detection of the presence of flame or smoke in the input image), detection (determination of flame and smoke appearance and long designation) and division (look at the appearance of flame (detect flame and illustrate.

In 2019, Lin et al. [6] established a unified detection framework that integrates quicker "R-CNN and 3D CNN," in which faster R-CNN facilitates smoke localization the usage of static spatial records, while 3D CNN [7] accomplishes smoke reputation by means of synthesizing dynamic spatiotemporal statistics. This method considerably enhances the detection accuracy of smoke compared to conventional convolutional detection algorithms. In 2020, Li et al. [8] employed faster "R-CNN [9], R-FCN [10], SSD [11], and YOLOv3 [12]" for flame detection, concluding that the CNN-based model attained a superior equilibrium of accuracy and detection efficacy.

To mitigate the version's increased leave out detection fee, "recursive gated convolution (Gnconv)" is integrated into the "feature pyramid network (FPN)", resulting in a unique Gnconv-FPN architecture. This enhancement augments the model's capacity to method better-order data, replicates the efficacy of the self-attention mechanism, prevents the lack of target data, and similarly enhances the detection accuracy of small goals. In order to beautify the possibility of functional extraction of the version, the version includes the mechanism of extreme visual attention (attention) the size of the PUT-FPN, which combines all functions and aspects of operations to replace conversion channels of functions. Quickly or later, Siou [27], which is characteristic of enlargement of the school period and convergence for a version that evaluates the vector angle between regression.

2. LITERATURE SURVEY

Object identification, intently linked to video analysis and photograph comprehension, has garnered significant research interest in latest years. Methods of detection of traditional elements depend on handmade functions and superficial trained architecture. Their performance without problems is stabilized through complex clothing that includes multiple low spot images with reference to product detectors and visual classification. Rapid development in deep learning has brought about the emergence of more sophisticated tools capable of obtaining semantic, high-level, and deeper traits to tackle issues inherent in older architectures. These models showcase variations in network structure, schooling technique, and optimization feature, among other factors. This paper [1] Deep learning -full of object detection framework. This assessment begins with a brief assessment of deep learning and its history of its basic instruments, the "Convolutional Neural network (CNN)" [13, 16]. In the end, we give attention to trendy everyday item detection architectures, incorporating enhancements and powerful strategies to beautify detection overall performance. For the reason that numerous specific detection obligations possess particular homes, we will also succinctly evaluation a few such obligations, this includes the detection of the main object, facial detection and walking detection. Experimental research is carried out to explore specific strategies and draw large conclusions. In the long run, various potential avenues and objectives are offered to manual future endeavors in each item detection and associated neural community-based learning structures.

Item detection is a simple issue in visible popularity within laptop imaginative and prescient and has been substantially researched during the last decades. Visual object popularity seeks to perceive gadgets of unique goal instructions internal a photo, accurately localizing each prevalence and assigning the correct class label. In light of the great achievements in deep learning-based photo type, item detection methodologies using deep learning were considerably researched in current years. This paintings [2] offers a radical assessment of latest advancements in visible object popularity with deep learning. Through an extensive exam of cutting-edge literature, we comprehensively examine the contemporary object detection frameworks and categorize the survey into three important sections: "(i) detection components, (ii) learning methodologies, and (iii) packages and benchmarks". The survey comprehensively examines several parameters influencing detection performance, which includes detector architectures, feature learning, concept era, and sampling methodologies. in the end, we examine many prospective avenues to decorate and stimulate future research in visible item detection via deep learning. Keywords: object Detection, Deep mastering, Deep Convolutional Neural Networks.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 05, May-2025

This research provides access to video mammals through multifunction merger. The temporary and spatial properties of flames, including standard flame dynamics and shelter indicators, are used in an error as a detection method of flashing [8] colored video sequences contained in the identification framework. First, the sophisticated "GAUSI AGIGATE" model is first used to remove dynamic objects in the foreground from the static story of detection scenes; Finally, renowned transmission devices are marked with candidates and unrelated areas of flame that use a set of flame filtering with rules; Finally, the flame fibrillation is used by identity based on the statistical assessment of the frequency to separate the real flames from the gadget in the video algorithm. The effects that seek that the proposed algorithms are effective, strong and green. Access to flame detection can occur with an image of 320 × 240 pixels on a computer device with AMD 2.04 GHz processor at the price of 24 frames per second.

Fire consequences from chemical reactions between oxygen and atmospheric additives, which can result in ecological imbalance on a wide scale by producing by-merchandise together with smoke. On the grounds that 1990, nearly ninety million fire occurrences have been said, resulting in fatalities and jeopardizing endangered and susceptible sources. The cause is frequently indeterminate; nonetheless, international warming is regularly identified as a contributing component in times of forest fires. On this look at [4], we create a multidisciplinary system that detects traces of fireplace and smoke in any isolation. The machine employs computer vision to analyze and stumble on fire or smoke in actual-time the use of a Deep learning set of rules [3, 4, and 2, 1] and offers notifications upon detection.

international fireplace disasters inflict social, environmental, and economic damage, rendering early discovery and prompt reporting crucial for the renovation of human lives and assets. Smoke detection is critical for early fire identity; but, maximum present day technology are restricted to either indoor or outside tracking settings, exhibiting subpar overall performance in hazy situations. This research [5] introduces a "Convolutional Neural network (CNN)"-based totally framework for smoke detection and segmentation in each clean and hazy situations. We make use of a green CNN structure, known as EfficientNet, for smoke detection, achieving advanced accuracy as compared to preceding tactics. We separate the smoke areas using DeepLabv3+, which employs green encoders and decoders along with a pixel-sensible classifier for top of the line localization. Our consequences of smoke detection reveal a good increase in accuracy of up to several% and a decrease in zero.46% inside the false alarm rate (ways), while segmentation indicates a huge increase of 2% in global accuracy and 1% means "penetration over the Union (iou"). This makes our method optimal for smoke detection and segmentation in practical supervision.

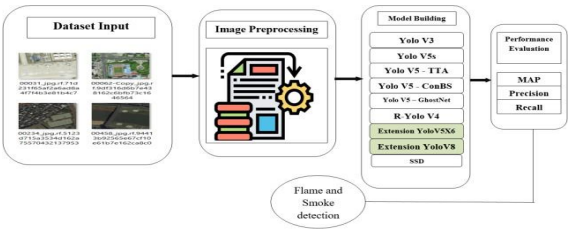
3. METHODOLOGY

i) Proposed Work:

The study offers a method of flame and smoke detection that uses "Odconvbs" in "Yolov5 [25, 26, 27]". Originally, the standard conference blocks in the Yolov5 spine with ODCONVBS are replaced to facilitate the restoration of vigilant data from the converting kernel. Second, GnConv is part of the neck to increase the model of the model to extract the spatial data of high orders. The task investigates the usage of "YOLO v8 and YOLO v5x6" models to improve the efficacy of flame and smoke detection, targeting more desirable accuracy in final predictions, with YOLO v5x6 reaching sizeable metrics of 79.2% mAP, 74% recall, and 80.6% precision. Those superior models enhance talents and increase object detection, as a result augmenting the overall effectiveness of the detection method. Furthermore, it proposes the creation of an intuitive front end utilizing the Flask framework to enhance user testing and engagement.

ii) System Architecture:

The computer is produced a wide model of flame and smoke detection, shown in Figure 1. Discover the Pram and the smoke image [25,26,27] The SPPF module is used on the spinal grid terminal, which works with higher speed, to standardize the dimensions of function maps obtained from the rear network and work to decorate the accuracy of functional extraction. Functional maps are finally transmitted to the neck network "(GNCONV-FPN)" for processing and fusion, which facilitates spatial spatial data within the map of the characteristics and meets the device for self-harm. The confidence in the neck network is used to increase data integration between different groups. Ultimately, the records is transmitted to the primary network to finalize the item detection process. Parent 1 illustrates the enhanced network model shape.

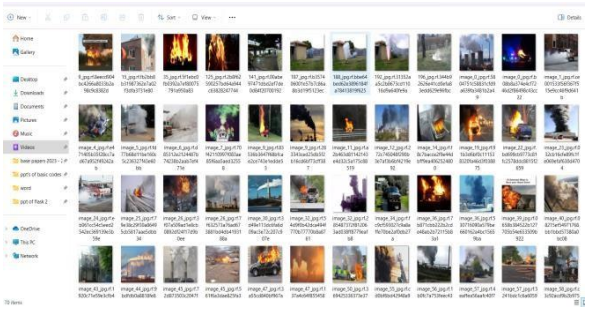


"Fig 1Proposed Architecture"

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 05, May-2025

iii) Dataset collection:

The limited quantity of images, singular scenario, and low resolution in the public flame and smoke dataset avert the enhancement of the model's generalization functionality. To enhance version generalization and tiny target recognition capabilities, it's miles essential to enhance the diversity of datasets. This paper [20] utilizes web crawlers to collect flame and smoke snap shots from the internet, then labeling these images into datasets for version training and evaluation. The gathering of four, 998 photographs changed into divided into training, validation, and test sets in an 8:1:1 ratio, encompassing a selection of flame smoke scenarios relevant to the study's focus



"Fig 2 Dataset images"

iv) Image Processing:

Image processing is essential for object recognition in independent using systems, involving numerous critical phases. The preliminary stage entails transforming the enter photo into a blob item, so optimizing it for further analysis and modification. Subsequently, the categories of items to be recognized are established, specifying the precise classifications that the algorithm intends to recognize. Concurrently, bounding boxes are established, delineating the areas of hobby in the photo where objects are predicted to be situated. The analyzed data is subsequently transformed into a NumPy array, a crucial process for effective numerical computation and analysis.

The next phase includes loading a pre-trained model, utilising prior knowledge from comprehensive datasets. This involves reading the network layers of the pre-trained model, which encompass learning characteristics and parameters essential for specific item detection. Moreover, output layers are derived, yielding conclusive predictions and facilitating efficient item identification and categorization.

Moreover, within the photograph processing pipeline, the picture and annotation file are blended, making certain whole facts for next evaluation. The color space is modified via changing from BGR to RGB, and a masks is generated to emphasize pertinent traits. The photo is ultimately scaled to decorate its suitability for subsequent processing and analysis. This distinct image processing workflow offers a robust basis for reliable and specific item recognition in the evolving environment of autonomous riding structures, hence improving safety and decision-making on the road.

v) Data Augmentation:

Data augmentation is a crucial method for enhancing the diversity and resilience of training datasets for machine learning models, especially in image processing and computer vision. The system encompasses three main alterations to enhance the authentic dataset: randomization of the image, rotation of the image, and transformation of the image.

Randomizing the image provides variety thru the application of random alterations, which include modifications in brightness, comparison, or colour saturation. This stochastic method enhances the version's potential to generalize to unfamiliar input and varied environmental conditions.

Rotating the image includes altering the orientation of the original image by means of various levels. This augmentation method assists in teaching the version to discover matters from various viewpoints, emulating discrepancies in real-world situations.

Image transformation encompass geometric changes like scaling, shearing, and flipping. These modifications enhance the dataset by include distortions that replicate real-world variances in item appearance and orientation.

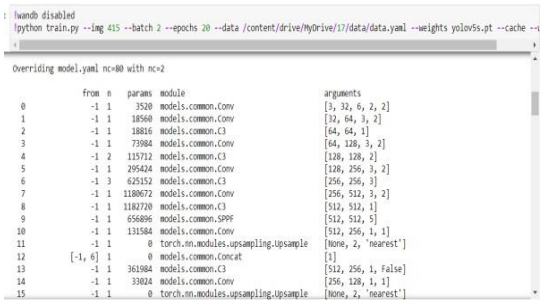
Utilizing these data augmentation strategies enhances the training dataset's comprehensiveness, enabling the model to acquire resilient features and patterns. This, as a consequence, enhances the model's capability to generalize and feature proficiently throughout varied and annoying test conditions. data augmentation is an crucial technique for reducing overfitting, improving model overall performance, and increasing the general reliability of machine learning models, in particular in applications inclusive of image recognition for autonomous driving systems.

vi) Algorithms:

YOLOv5s "You only look once (YOLO)" is a compact and efficient shape of the YOLO object detection technique, selected for its equilibrium among speed and precision. It directly forecasts bounding boxes and class possibilities, rendering it appropriate for real-time flame and smoke detection [17].

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 05, May-2025

YoloV5s



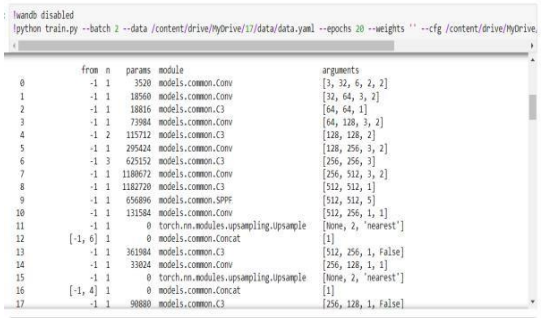
```
!wandb disabled
!python train.py --img 415 --batch 2 --epochs 20 --data /content/drive/MyDrive/17/data/data.yaml --weights yolov5s.pt --cache --
```

```
Overriding model.yaml nc=80 with nc=2

                 from  n    params  module                                  arguments
  0                -1  1      3520  models.common.Conv                      [3, 32, 6, 2, 2]
  1                -1  1     18560  models.common.Conv                      [32, 64, 3, 2]
  2                -1  1     18816  models.common.C3                        [64, 64, 1]
  3                -1  1     73984  models.common.Conv                      [64, 128, 3, 2]
  4                -1  2    115712  models.common.C3                        [128, 128, 2]
  5                -1  1    295424  models.common.Conv                      [128, 256, 3, 2]
  6                -1  3    625152  models.common.C3                        [256, 256, 3]
  7                -1  1   1180672  models.common.Conv                      [256, 512, 3, 2]
  8                -1  1   1182720  models.common.C3                        [512, 512, 1]
  9                -1  1    656896  models.common.SPPF                      [512, 512, 5]
 10                -1  1    131584  models.common.Conv                      [512, 256, 1, 1]
 11                -1  1         0  torch.nn.modules.upsampling.Upsample    [None, 2, 'nearest']
 12          [-1, 6]  1         0  models.common.Concat                    [1]
 13                -1  1    361984  models.common.C3                        [512, 256, 1, False]
 14                -1  1     33024  models.common.Conv                      [256, 128, 1, 1]
 15                -1  1         0  torch.nn.modules.upsampling.Upsample    [None, 2, 'nearest']
```

“Fig 3 YOLOV5s”

YOLOv5 – TTA test-time augmentation enhances YOLOv5 by integrating augmentation techniques during inference. This method improves the model's resilience and precision by implementing a couple of adjustments to enter photos, facilitating more dependable flame and smoke detection across diverse real-time settings.

Yolo V5 - TTA



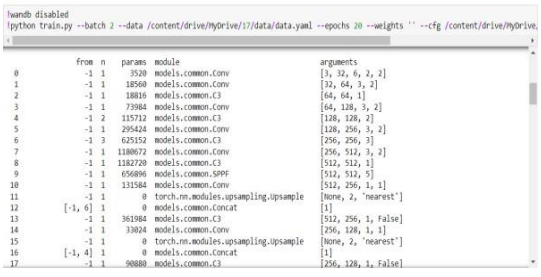
```
!wandb disabled
!python train.py --batch 2 --data /content/drive/MyDrive/17/data/data.yaml --epochs 20 --weights '' --cfg /content/drive/MyDrive/
```

```
                 from  n    params  module                                  arguments
  0                -1  1      3520  models.common.Conv                      [3, 32, 6, 2, 2]
  1                -1  1     18560  models.common.Conv                      [32, 64, 3, 2]
  2                -1  1     18816  models.common.C3                        [64, 64, 1]
  3                -1  1     73984  models.common.Conv                      [64, 128, 3, 2]
  4                -1  2    115712  models.common.C3                        [128, 128, 2]
  5                -1  1    295424  models.common.Conv                      [128, 256, 3, 2]
  6                -1  3    625152  models.common.C3                        [256, 256, 3]
  7                -1  1   1180672  models.common.Conv                      [256, 512, 3, 2]
  8                -1  1   1182720  models.common.C3                        [512, 512, 1]
  9                -1  1    656896  models.common.SPPF                      [512, 512, 5]
 10                -1  1    131584  models.common.Conv                      [512, 256, 1, 1]
 11                -1  1         0  torch.nn.modules.upsampling.Upsample    [None, 2, 'nearest']
 12          [-1, 6]  1         0  models.common.Concat                    [1]
 13                -1  1    361984  models.common.C3                        [512, 256, 1, False]
 14                -1  1     33024  models.common.Conv                      [256, 128, 1, 1]
 15                -1  1         0  torch.nn.modules.upsampling.Upsample    [None, 2, 'nearest']
 16          [-1, 4]  1         0  models.common.Concat                    [1]
 17                -1  1     90880  models.common.C3                        [256, 128, 1, False]
```

“Fig 4 YOLOV5-TTA”

The YOLOv5s the spine consists of the Oponvbs module, which uses Omni-Dimensional dynamic conversion with batchnormalization and power activation. At the same time, this method improves functional extraction by means of a contractual key area, so the accuracy of flame and smoking detection improves.

Yolo V5 - ConBS



```
!wandb disabled
!python train.py --batch 2 --data /content/drive/MyDrive/17/data/data.yaml --epochs 20 --weights '' --cfg /content/drive/MyDrive/
```

```
                 from  n    params  module                                  arguments
  0                -1  1      3520  models.common.Conv                      [3, 32, 6, 2, 2]
  1                -1  1     18560  models.common.Conv                      [32, 64, 3, 2]
  2                -1  1     18816  models.common.C3                        [64, 64, 1]
  3                -1  1     73984  models.common.Conv                      [64, 128, 3, 2]
  4                -1  2    115712  models.common.C3                        [128, 128, 2]
  5                -1  1    295424  models.common.Conv                      [128, 256, 3, 2]
  6                -1  3    625152  models.common.C3                        [256, 256, 3]
  7                -1  1   1180672  models.common.Conv                      [256, 512, 3, 2]
  8                -1  1   1182720  models.common.C3                        [512, 512, 1]
  9                -1  1    656896  models.common.SPPF                      [512, 512, 5]
 10                -1  1    131584  models.common.Conv                      [512, 256, 1, 1]
 11                -1  1         0  torch.nn.modules.upsampling.Upsample    [None, 2, 'nearest']
 12          [-1, 6]  1         0  models.common.Concat                    [1]
 13                -1  1    361984  models.common.C3                        [512, 256, 1, False]
 14                -1  1     33024  models.common.Conv                      [256, 128, 1, 1]
 15                -1  1         0  torch.nn.modules.upsampling.Upsample    [None, 2, 'nearest']
 16          [-1, 4]  1         0  models.common.Concat                    [1]
 17                -1  1     90880  models.common.C3                        [256, 128, 1, False]
```

“Fig 6 YOLOV% convbs”

YOLOv5 with GhostNet it is an altered iteration of the "YOLOv5" item detection model, integrating GhostNet modules. GhostNet seeks to limit computational expenses by offering a streamlined design characterized by decreased parameters and computations, all while preserving excellent accuracy. The project employs "YOLOv5" GhostNet to achieve an equilibrium among efficiency and performance, rendering it appropriate for real-time packages and environments with constrained hardware resources. The integration guarantees efficient object detection with enhanced computational efficacy.

Yolo V5 - GhostNet

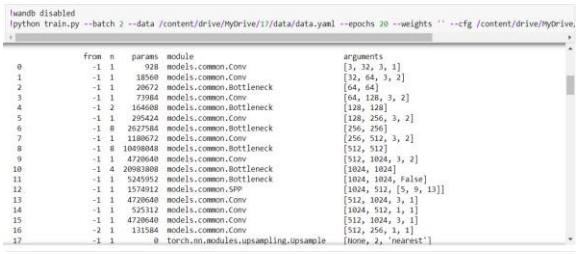
```
!wandb disabled
!python train.py --batch 2 --data /content/drive/MyDrive/17/data/data.yaml --epochs 20 --weights '' --cfg /content/yolov5/models/
```

```
                 from  n    params  module                                  arguments
  0                -1  1      3520  models.common.Conv                      [3, 32, 6, 2, 2]
  1                -1  1     10144  models.common.GhostConv                 [32, 64, 3, 2]
  2                -1  1      9656  models.common.C3Ghost                   [64, 64, 1]
  3                -1  1     38720  models.common.GhostConv                 [64, 128, 3, 2]
  4                -1  2     40160  models.common.C3Ghost                   [128, 128, 2]
  5                -1  1    151168  models.common.GhostConv                 [128, 256, 3, 2]
  6                -1  3    165024  models.common.C3Ghost                   [256, 256, 3]
  7                -1  1    597248  models.common.GhostConv                 [256, 512, 3, 2]
  8                -1  1    564672  models.common.C3Ghost                   [512, 512, 1]
  9                -1  1    656896  models.common.SPPF                      [512, 512, 5]
 10                -1  1     69248  models.common.GhostConv                 [512, 256, 1, 1]
 11                -1  1         0  torch.nn.modules.upsampling.Upsample    [None, 2, 'nearest']
 12          [-1, 6]  1         0  models.common.Concat                    [1]
 13                -1  1    208608  models.common.C3Ghost                   [512, 256, 1, False]
 14                -1  1     18240  models.common.GhostConv                 [256, 128, 1, 1]
 15                -1  1         0  torch.nn.modules.upsampling.Upsample    [None, 2, 'nearest']
 16          [-1, 4]  1         0  models.common.Concat                    [1]
 17                -1  1     53104  models.common.C3Ghost                   [256, 128, 1, False]
```

“Fig 7 YOLOv5 with GhostNet”

YOLOv4 it is a better iteration of the YOLO object identification algorithm, recognized for its superior accuracy and performance. It contains architectural upgrades, consisting of the CSPDarknet53 backbone and PANet for characteristic aggregation, ensuing in superior object detection performance. YOLOv4 is presumably employed in the assignment due to its superior capabilities, providing super accuracy in flame and smoke detection [14].

YoloV4



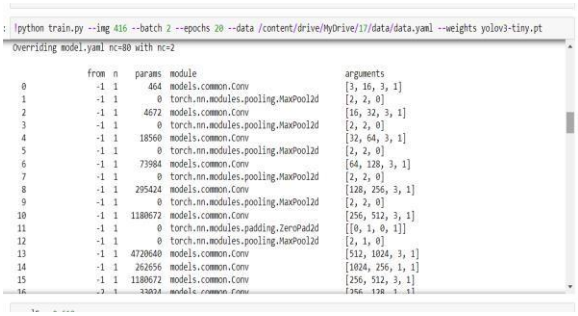
```
!wandb disabled
!python train.py --batch 2 --data /content/drive/MyDrive/17/data/data.yaml --epochs 20 --weights '' --cfg /content/drive/MyDrive/
```

```
                 from  n    params  module                                  arguments
  0                -1  1       928  models.common.Conv                      [3, 32, 3, 1]
  1                -1  1     18560  models.common.Conv                      [32, 64, 3, 2]
  2                -1  1     20672  models.common.Bottleneck                [64, 64]
  3                -1  1     73984  models.common.Conv                      [64, 128, 3, 2]
  4                -1  2    164608  models.common.Bottleneck                [128, 128]
  5                -1  1    295424  models.common.Conv                      [128, 256, 3, 2]
  6                -1  8   2627584  models.common.Bottleneck                [256, 256]
  7                -1  1   1180672  models.common.Conv                      [256, 512, 3, 2]
  8                -1  8  10498048  models.common.Bottleneck                [512, 512]
  9                -1  1   4720640  models.common.Conv                      [512, 1024, 3, 2]
 10                -1  4  20983808  models.common.Bottleneck                [1024, 1024]
 11                -1  1   5245952  models.common.Bottleneck                [1024, 1024, False]
 12                -1  1   1574912  models.common.SPP                       [1024, 512, [5, 9, 13]]
 13                -1  1   4720640  models.common.Conv                      [512, 1024, 3, 1]
 14                -1  1    525312  models.common.Conv                      [1024, 512, 1, 1]
 15                -1  1   4720640  models.common.Conv                      [512, 1024, 3, 1]
 16                -2  1    131584  models.common.Conv                      [512, 256, 1, 1]
 17                -1  1         0  torch.nn.modules.upsampling.Upsample    [None, 2, 'nearest']
```

“Fig 8 YOLOV4”

YOLOv3 is a prior iteration of the YOLO collection, recognized for its rapidity and precision in real-time object identification. It utilizes a Darknet-53 backbone and incorporates "feature pyramid networks (FPN)" to beautify feature representation across varying scales. YOLOv3 is probably incorporated into the task due to its demonstrated efficacy in swift and unique object recognition, rendering it appropriate for real-time flame and smoke detection applications.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 05, May-2025



```
!python train.py --img 416 --batch 2 --epochs 20 --data /content/drive/MyDrive/17/data/data.yaml --weights yolov3-tiny.pt
Overriding model.yaml nc=80 with nc=2

                 from  n    params  module                                  arguments
  0                -1  1       464  models.common.Conv                      [3, 16, 3, 1]
  1                -1  1         0  torch.nn.modules.pooling.MaxPool2d      [2, 2, 0]
  2                -1  1      4672  models.common.Conv                      [16, 32, 3, 1]
  3                -1  1         0  torch.nn.modules.pooling.MaxPool2d      [2, 2, 0]
  4                -1  1     18560  models.common.Conv                      [32, 64, 3, 1]
  5                -1  1         0  torch.nn.modules.pooling.MaxPool2d      [2, 2, 0]
  6                -1  1     73984  models.common.Conv                      [64, 128, 3, 1]
  7                -1  1         0  torch.nn.modules.pooling.MaxPool2d      [2, 2, 0]
  8                -1  1    295424  models.common.Conv                      [128, 256, 3, 1]
  9                -1  1         0  torch.nn.modules.pooling.MaxPool2d      [2, 2, 0]
 10                -1  1   1180672  models.common.Conv                      [256, 512, 3, 1]
 11                -1  1         0  torch.nn.modules.padding.ZeroPad2d      [0, 1, 0, 1]]
 12                -1  1         0  torch.nn.modules.pooling.MaxPool2d      [2, 1, 0]
 13                -1  1   4720640  models.common.Conv                      [512, 1024, 3, 1]
 14                -1  1    262656  models.common.Conv                      [1024, 256, 1, 1]
 15                -1  1   1180672  models.common.Conv                      [256, 512, 3, 1]
```

"Fig 9 YOLOV3"

SSD it is a quick object detection technique that forecasts object locations and class chances in a single iteration. Its efficacy renders it suitable for actual-time applications, facilitating speedy and powerful flame and smoke detection in the project [11].

```
# get the model using our helper function
model = get_model_bbox(num_classes)

'''
Use this to reset all trainable weights
model.apply(reset_weights)
'''

# move model to the right device
model.to(device)

# construct an optimizer
params = [p for p in model.parameters() if p.requires_grad]
optimizer = torch.optim.SGD(params, lr=0.005,  # Feel free to play with values
                            momentum=0.9, weight_decay=0)

# Defining learning rate scheduler
lr_scheduler = torch.optim.lr_scheduler.StepLR(optimizer,
                                               step_size=20,
                                               gamma=0.2)


result_mAP = []
best_epoch = None

# Let's train!
for epoch in range(num_epochs):


    # train for one epoch, printing every 10 iterations
    train_one_epoch(model, optimizer, data_loader, device, epoch, print_freq=50)
    # update the learning rate
    lr_scheduler.step()
    # evaluate on the test dataset
    results =  evaluate(model, data_loader_test, device=device)
    # saves results of mAP @ IoU = 0.5
    result_mAP.append(results.coco_eval['bbox'].stats[1])
    #save the best model so far
    if result_mAP[-1] == max(result_mAP):
        best_save_path = os.path.join(f'bestmodel_noaug_sgd(wd=0)_8batch-epoch{epoch}.pth')
        torch.save(model.state_dict(), best_save_path)
        best_epoch = int(epoch)
        print(f'\n\nmodel from epoch number {epoch} saved!\n result is {max(result_mAP)}\n\n')

# Saving the last model
save_path = os.path.join(f'noaug_sgd_2batch-lastepoch{num_epochs-1}.pth')
torch.save(model.state_dict(), save_path)
print(f'model from last epoch(no.{num_epochs-1}) saved')
```

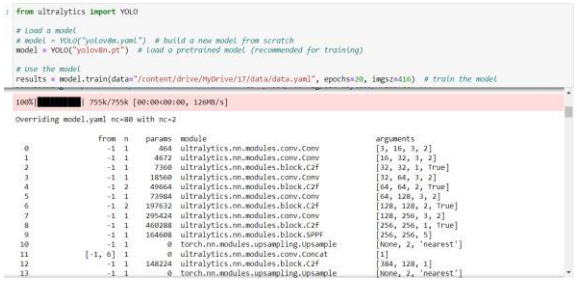
"Fig 10 SSD"

Faster R-CNN it is a -stage object detection approach identified for its advanced accuracy. It suggests potential zones inside a photo and enhances them, providing strong detection efficacy. The project pick faster R-CNN for its accuracy, guaranteeing meticulous and precise analysis of flame and smoke occurrences in various settings [9].

```
# get the model using our helper function
model = get_model_bbox(num_classes)

'''
Use this to reset all trainable weights
model.apply(reset_weights)
'''

# move model to the right device
model.to(device)

# construct an optimizer
params = [p for p in model.parameters() if p.requires_grad]
optimizer = torch.optim.SGD(params, lr=0.005,  # Feel free to play with values
                            momentum=0.9, weight_decay=0)

# Defining learning rate scheduler
lr_scheduler = torch.optim.lr_scheduler.StepLR(optimizer,
                                               step_size=20,
                                               gamma=0.2)


result_mAP = []
best_epoch = None

# Let's train!
for epoch in range(num_epochs):


    # train for one epoch, printing every 10 iterations
    train_one_epoch(model, optimizer, data_loader, device, epoch, print_freq=50)
    # update the learning rate
    lr_scheduler.step()
    # evaluate on the test dataset
    results =  evaluate(model, data_loader_test, device=device)
    # saves results of mAP @ IoU = 0.5
    result_mAP.append(results.coco_eval['bbox'].stats[1])
    #save the best model so far
    if result_mAP[-1] == max(result_mAP):
        best_save_path = os.path.join(f'bestmodel_noaug_sgd(wd=0)_8batch-epoch{epoch}.pth')
        torch.save(model.state_dict(), best_save_path)
        best_epoch = int(epoch)
        print(f'\n\nmodel from epoch number {epoch} saved!\n result is {max(result_mAP)}\n\n')

# Saving the last model
save_path = os.path.join(f'noaug_sgd_2batch-lastepoch{num_epochs-1}.pth')
torch.save(model.state_dict(), save_path)
print(f'model from last epoch(no.{num_epochs-1}) saved')
```

"Fig 11 Faster RCNN"

YOLOv8 YOLOv8 is distinguished for its superiority in object identification, segmentation, and posture estimation, selected for its ideal equilibrium of pace, precision, and adaptability. This advanced model is optimal for real-time flame and smoke detection in many settings, owing to its superior performance and intuitive design.

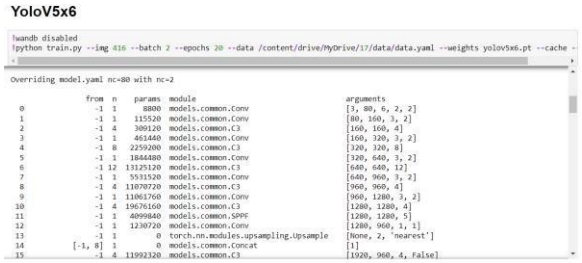


```
from ultralytics import YOLO

# Load a model
# model = YOLO("yolov8m.yaml")  # build a new model from scratch
model = YOLO("yolov8n.pt")  # load a pretrained model (recommended for training)

# Use the model
results = model.train(data="/content/drive/MyDrive/17/data/data.yaml", epochs=20, imgsz=416)  # train the model

100%|██████████| 755k/755k [00:00<00:00, 126MB/s]
Overriding model.yaml nc=80 with nc=2

                   from  n    params  module                                       arguments
  0                  -1  1       464  ultralytics.nn.modules.conv.Conv             [3, 16, 3, 2]
  1                  -1  1      4672  ultralytics.nn.modules.conv.Conv             [16, 32, 3, 2]
  2                  -1  1      7360  ultralytics.nn.modules.block.C2f             [32, 32, 1, True]
  3                  -1  1     18560  ultralytics.nn.modules.conv.Conv             [32, 64, 3, 2]
  4                  -1  2     49664  ultralytics.nn.modules.block.C2f             [64, 64, 2, True]
  5                  -1  1     73984  ultralytics.nn.modules.conv.Conv             [64, 128, 3, 2]
  6                  -1  2    197632  ultralytics.nn.modules.block.C2f             [128, 128, 2, True]
  7                  -1  1    295424  ultralytics.nn.modules.conv.Conv             [128, 256, 3, 2]
  8                  -1  1    460288  ultralytics.nn.modules.block.C2f             [256, 256, 1, True]
  9                  -1  1    164608  ultralytics.nn.modules.block.SPPF            [256, 256, 5]
 10                  -1  1         0  torch.nn.modules.upsampling.Upsample         [None, 2, 'nearest']
 11             [-1, 6]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 12                  -1  1    148224  ultralytics.nn.modules.block.C2f             [384, 128, 1]
 13                  -1  1         0  torch.nn.modules.upsampling.Upsample         [None, 2, 'nearest']
```

"Fig 12 YOLOv8"

YOLOv5x6, A variation of YOLOv5, it excels in real-time object identification owing to its rapidity, accuracy, and light-weight architecture. Featuring CSPDarknet for feature extraction and PANet for feature fusion, it excels in efficient and precise flame and smoke detection, especially suitable for mobile device deployment.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 05, May-2025



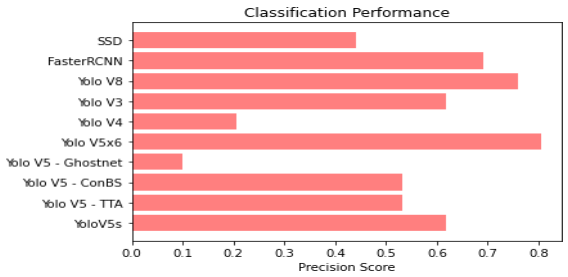
"Fig 13 YOLOV5X6"

4. EXPERIMENTAL RESULTS

Precision: Precision assesses the proportion of accurately categorized cases among the ones identified as positive. Consequently, the formula for calculating precision is expressed as:

"Precision = True positives/ (True positives + False positives) = TP/(TP + FP)"

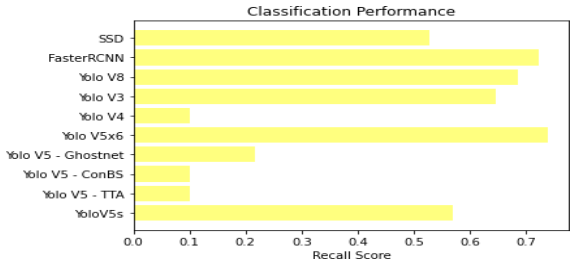
$$\text{Precision} = \frac{True\ Positive}{True\ Positive + False\ Positive}$$



"Fig 14 Precision comparison graph"

Recall: Remember that "machine learning" has a calculation that considers the model's ability to recognize all relevant times in a particular class. Composing the right positive comments for the general actual positivity and providing insight into the model efficiency to identify events in a particular class.

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



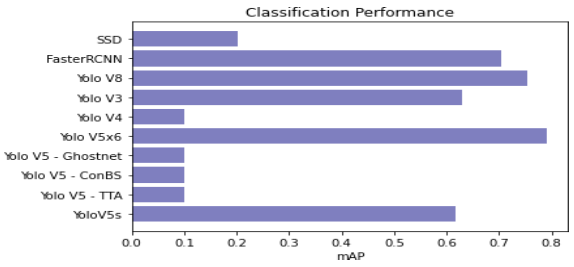
"Fig 15 Recall comparison graph"

MAP: "Average average accuracy (map)" is statistics for evaluation of "quality of evaluation". It evaluates the number of relevant proposals and their location in the list. The K Map is determined because the arithmetic mean "average accuracy (AP)" in OK for all users or questions.

$$mAP = \frac{1}{n}\sum_{k=1}^{k=n} AP_k$$

$$AP_k = \text{the AP of class } k$$

$$n = \text{the number of classes}$$



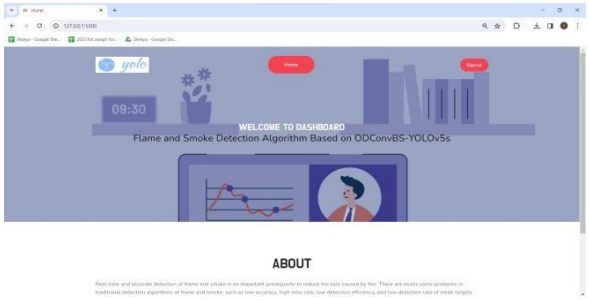
"Fig 16 mAP comparison graph"

Published by :
http://www.ijert.org

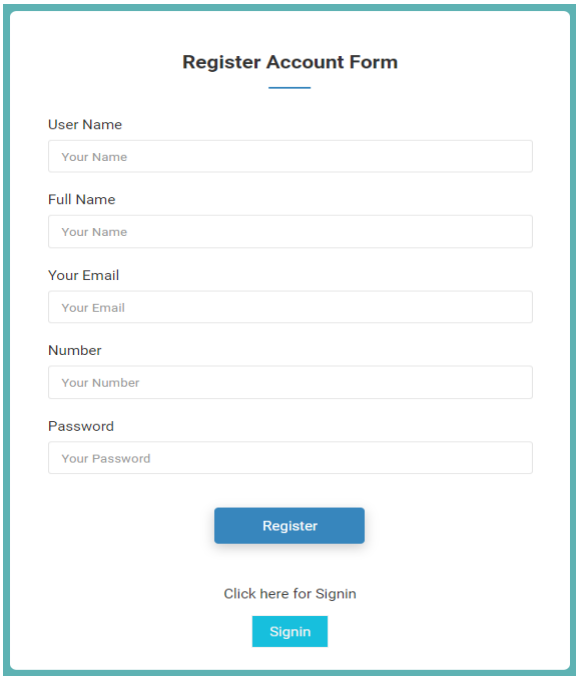
International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 05, May-2025

ML Model	Precision	Recall	mAP
YoloV5s	0.618	0.570	0.617
Yolo V5 - TTA	0.533	0.100	0.100
Yolo V5 - ConBS	0.533	0.100	0.100
Yolo V5 - Ghostnet	0.100	0.217	0.100
Extension Yolo V5x6	0.806	0.740	0.792
Yolo V4	0.207	0.100	0.100
Yolo V3	0.618	0.646	0.630
Extension Yolo V8	0.759	0.685	0.753
Faster RCNN	0.692	0.722	0.705
SSD	0.440	0.527	0.202

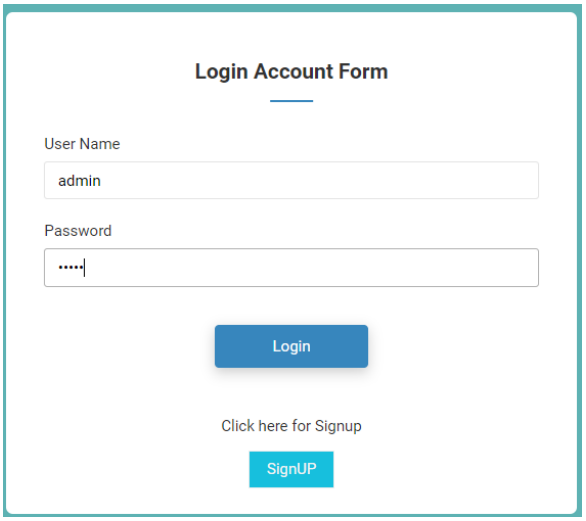
"Fig 17 Performance Evaluation table"



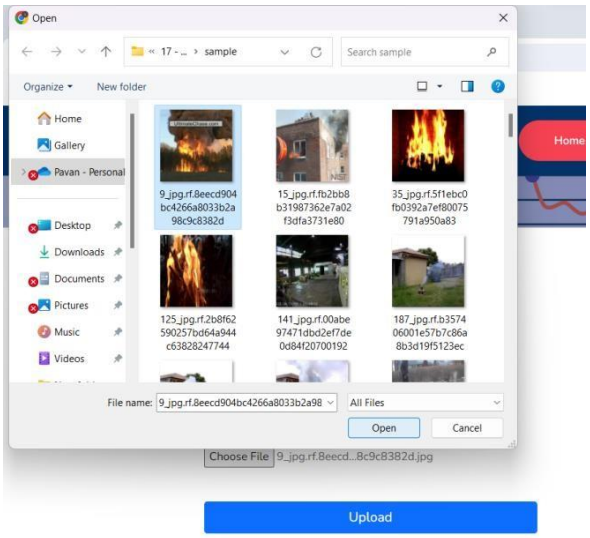
"Fig 18 Home page"



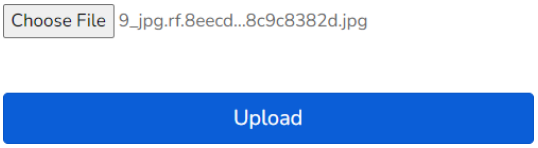
"Fig 19 Registration page"



"Fig 20 Login page"



"Fig 21 Input image folder"



'Fig 22 Upload input image"

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 05, May-2025

"Fig 23 Predict result for given input'

5. CONCLUSION

by examining many superior models, such as YOLO V5 variations, YOLO V8, SSD, and faster R-CNN The task attained a thorough comprehension of its efficacy in figuring out fires and smoke in far off aerial satellite imagery. The experiment broadened its attain by assessing the ODConvBS-YOLOv5s version, explicitly engineered for flame and smoke detection. This unique algorithm turned into evaluated in opposition to other models to decide its efficacy in improving detection accuracy. The deployment of a Flask-based web interface with SQLite authentication improves consumer experience, rendering it accessible for those seeking an intuitive platform for satellite image analysis. YOLO v5x6, an extension, demonstrates a great mAP of 0.792, making it an outstanding solution for real-time flame and smoke detection. Its exceptional overall performance positions it as a finest approach in object detection, showcasing reliability and efficiency across various packages. The developed system serves the unique characteristic of detecting flames and smoke in remote aerial satellite photos while additionally organizing a basis for destiny upgrades and applications in satellite image processing, benefiting researchers and cease-users across several industries.

6. FUTURE SCOPE

investigate the integration of supplementary sensor records, consisting of infrared and thermal imaging, to improve the system's capacity to pick out flame and smoke in diverse environmental situations, such as low-light or obstructed situations. Increase the mission to contain an actual-time decision support system that leverages discovered flame and smoke records to supply prompt data for emergency response teams, thereby enhancing the rate and quality of decision-making throughout fire conditions. Have a look at the implementation of the detection method on part computing devices to facilitate decentralized processing. This can enhance the system's responsiveness and lessen reliance on centralized servers, rendering it greater appropriate for allotted packages. Consistently augment and diversify the dataset with various flame and smoke scenarios to always beautify the model's overall performance. This ensures variation to changing climatic circumstances and new problems in fire detection, fostering a resilient and future-ready system.

REFERENCES

[1] Z.-Q. Zhao, P. Zheng, S.-T. Xu, and X. Wu, ''Object detection with deep learning: A review,'' IEEE Trans. Pattern Anal. Mach. Intell., vol. 30, no. 11, pp. 3212–3232, Nov. 2019.

[2] X. Wu, D. Sahoo, and S. C. Hoi, ''Recent advances in deep learning for object detection,'' Neurocomputing, vol. 396, pp. 39–64, Jul. 2020.

[3] J. Chen, Y. He, and J. Wang, ''Multi-feature fusion based fast video flame detection,'' Building Environ., vol. 45, no. 5, pp. 1113–1122, May 2010.

[4] A. Pawar, ''A multi-disciplinary vision-based fire and smoke detection system,'' in Proc. 4th Int. Conf. Electron., Commun. Aerosp. Technol. (ICECA), Nov. 2020, pp. 900–904.

[5] S. Khan, K. Muhammad, and T. Hussain, ''Deepsmoke: Deep learning model for smoke detection and segmentation in outdoor environments,'' Expert Syst. Appl., vol. 182, Nov. 2021, Art. no. 115125.

[6] G. Lin, Y. Zhang, G. Xu, and Q. Zhang, ''Smoke detection on video sequences using 3D convolutional neural networks,'' Fire Technol., vol. 55, no. 5, pp. 1827–1847, Sep. 2019.

[7] L. Wang, W. Li, and W. Li, ''Appearance-and- relation networks for video classification,'' in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 1430–1439.

[8] G. Lin, Y. Zhang, G. Xu, and Q. Zhang, ''Smoke detection on video sequences using 3D convolutional neural networks,'' Fire Technol., vol. 55, no. 5, pp. 1827–1847, Sep. 2019.

[9] L. Wang, W. Li, and W. Li, ''Appearance-and- relation networks for video classification,'' in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 1430–1439.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 05, May-2025

[10] P. Li and W. Zhao, ‘‘Image fire detection algorithms based on convolutional neural networks,’’ Case Stud. Thermal Eng., vol. 19, Jun. 2020, Art. no. 100625.

[11] S. Ren, K. He, and R. Girshick, ‘‘Towards real- time object detection with region proposal networks,’’ 2019, arXiv:1506.01497.

[12] J. Dai, Y. Li, and K. He, ‘‘R-FCN: Object detection via region-based fully convolutional networks,’’ in Proc. Adv. Neural Inf. Process. Syst., vol. 29, 2016, pp. 1–11.

[13] W. Liu, D. Anguelov, and D. Erhan, ‘‘SSD: Single shot multibox detector,’’ in Proc. Eur. Conf. Comput. Vis. Cham, Switzerland: Springer, 2016, pp. 21–37.

[14] J. Redmon and A. Farhadi, ‘‘YOLOv3: An incremental improvement,’’ 2018, arXiv:1804.02767.

[15] S. Saponara, A. Elhanashi, and A. Gagliardi, ‘‘Real-time video fire/smoke detection based on CNN in antifire surveillance systems,’’ J. Real-Time Image Process., vol. 18, no. 3, pp. 889–900, 2021.

[16] M. Torabian, H. Pourghassem, and H. Mahdavi- Nasab, ‘‘Fire detection based on fractal analysis and spatio-temporal features,’’ Fire Technol., vol. 57, pp. 2583–2614, May 2021.

[17] K. Avazov, M. Mukhiddinov, F. Makhmudov, and Y. I. Cho, ‘‘Fire detection method in smart city environments using a deep-learning-based approach,’’ Electronics, vol. 11, p. 73, Dec. 2022.

[18] S. Majid, F. Alenezi, S. Masood, M. Ahmad, E. S. Gündüz, and K. Polat, ‘‘Attention based CNN model for fire detection and localization in real-world images,’’ Expert Syst. Appl., vol. 189, Mar. 2022, Art. no. 116114.

[19] Z. Xue, H. Lin, and F. Wang, ‘‘A small target forest fire detection model based on YOLOv5 improvement,’’ Forests, vol. 13, no. 8, p. 1332, Aug. 2022.

[20] Y. Hu, J. Zhan, and G. Zhou, ‘‘Fast forest fire smoke detection using MVMNet,’’ Knowl.-Based Syst., vol. 241, Jan. 2022, Art. no. 108219.

[21] J. Li, G. Zhou, and A. Chen, ‘‘Adaptive linear feature-reuse network for rapid forest fire smoke detection model,’’ Ecolog. Informat., vol. 68, May 2022, Art. no. 101584.

[22] O. Khudayberdiev, J. Zhang, and S. M. Abdullahi, ‘‘Light-FireNet: An efficient lightweight network for fire detection in diverse environments,’’ Multimedia Tools Appl., vol. 81, pp. 24553–24572, Mar. 2022.

[23] A. Hosseini, M. Hashemzadeh, and N. Farajzadeh, ‘‘UFS-Net: A unified flame and smoke detection method for early detection of fire in video surveillance applications using CNNs,’’ J. Comput. Sci., vol. 61, May 2022, Art. no. 101638.

[24] T. Liang, H. Bao, and W. Pan, ‘‘Traffic sign detection via improved sparse R-CNN for autonomous vehicles,’’ J. Adv. Transp., vol. 2022, pp. 1–16, Mar. 2022.

[25] Y. Gu and B. Si, ‘‘A novel lightweight real-time traffic sign detection integration framework based on YOLOv4,’’ Entropy, vol. 24, no. 4, p. 487, Mar. 2022.

[26] J. Wang, Y. Chen, and Z. Dong, ‘‘Improved YOLOv5 network for realtime multi-scale traffic sign detection,’’ Neural Comput. Appl., vol. 35, pp. 7853– 7865, Dec. 2022.

[27] F. Dadboud, V. Patel, and V. Mehta, ‘‘Single- stage UAV detection and classification with YOLOV5: Mosaic data augmentation and PANet,’’ in Proc. 17th IEEE Int. Conf. Adv. Video Signal Based Surveill. (AVSS), Nov. 2021, pp. 1–8.

[28] C. Si, Z. Zhang, and F. Qi, ‘‘Better robustness by more coverage: Adversarial and mixup data augmentation for robust finetuning,’’ in Proc. Findings Assoc. Comput. Linguistics, (ACL-IJCNLP), 2021, pp. 1569–1576.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 05, May-2025

[29] Z. Gevorgyan, ‘‘SIoU loss: More powerful learning for bounding box regression,’’ 2022, arXiv:2205.12740.

[30] T. Y. Lin, P. Dollár, and R. Girshick, ‘‘Feature pyramid networks for object detection,’’ in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jul. 2017, pp. 2117–2125.

[31] Z. Zheng, P. Wang, and D. Ren, ‘‘Enhancing geometric factors in model learning and inference for object detection and instance segmentation,’’ IEEE Trans. Cybern., vol. 52, no. 8, pp. 8574–8586, Aug. 2022.

[32] C. Li, A. Zhou, and A. Yao, ‘‘Omni-dimensional dynamic convolution,’’ 2022, arXiv:2209.07947.

[33] Q. L. Zhang and Y. B. Yang, ‘‘SA-Net: Shuffle attention for deep convolutional neural networks,’’ in Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP), Jun. 2021, pp. 2235–2239.

[34] N. Ma, X. Zhang, and H. T. Zheng, ‘‘ShuffleNet V2: Practical guidelines for efficient CNN architecture design,’’ in Proc. Eur. Conf. Comput. Vis. (ECCV), Sep. 2018, pp. 116–131.

[35] Y. Rao, W. Zhao, Y. Tang, J. Zhou, S.-N. Lim, and J. Lu, ‘‘HorNet: Efficient high-order spatial interactions with recursive gated convolutions,’’ 2022, arXiv:2207.14284.

[36] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, ‘‘An image is worth 16×16 words Transformers for image recognition at scale,’’ 2020, arXiv:2010.11929.