Annals of Emerging Technologies in Computing (AETiC)
Vol. 6, No. 3, 2022

Research Article

An Edge Computing Environment for Early Wildfire Detection

Ahmed Saleem Mahdi* **and Sawsen Abdulhadi Mahmood**

Mustansiriyah University, Baghdad, Iraq
asm_msc@uomustansiriyah.edu.iq; sawsenhadi@uomustansiriyah.edu.iq
*Correspondence: asm_msc@uomustansiriyah.edu.iq

Received: 25th March 2022; Accepted: 25th June 2022; Published: 1st July 2022

Abstract: Recently, an increasing demand is growing for installing a rapid response system in forest regions to enable an immediate and appropriate response to wildfires before they spread across vast areas. This paper introduces a multilevel system for early wildfire detection to support public authorities to immediately specify and attend to emergency demands. The presented work is designed and implemented within Edge Computing Infrastructure. At the first level; the dataset samples of wildfire represented by a set of video sequences are collected and labelled for training mode purposes. Then, YOLOv5 deep learning model is adopted in our framework to build a trained model for distinguishing the fire event against non-fire events in binary classification. The proposed system structure comprises IoT entities provided with camera sensor capabilities and NVIDIA Jetson Nano Developer kit as an edge computing environment. At the first level, a video camera is employed to assemble environment information received by the micro-controller middle level to handle and detect the possible fire event presenting in the interested area. The last level is characterized as making a decision by sending a text message and snapshot images to the cloud server. Meanwhile, a set of commands are sent to IoT nodes to operate the speakers and sprinklers, which are strategically assumed to place on the ground to give an alarm and prevent wildlife loss. The proposed system was tested and evaluated using a wildfire dataset constructed by our efforts. The experimental results exhibited 98% accurate detection of fire events in the video sequence. Further, a comparison study is performed in this research to confirm the results obtained from recent methods.

Keywords: ***Early Wildfire Detection System; Edge Computing; Jetson Nano; YOLOv5***

1. Introduction

Wildfires are unintentional flames that occur in naturalistic environments such as woods, prairies, and meadows. Wildfires are frequently started by humans or natural phenomena such as lightning, and they can occur at any time or in any region. It is unknown how half of the wildfires reported originated. Most ecosystems all around the world are being ravaged by wildfires[1]. The majority of flames begin small and quickly spread, making them incredibly difficult to control. Early wildfire detection is critical to combating the enormous size of wildfires raging worldwide. The development of real-time wildfire detection systems based on video surveillance has recently sparked much interest [1]. In order to mitigate the losses of forests resources, an effective development for monitoring wildfires detection system is required. The key point of early wildfire detection systems represents by minimizing the time spent between fire detection and alerting the appropriate authorities. Deep learning methods are widely employed for wildfire detection task with extensive dataset allowing researchers to precisely predict the wildfire event and prevent occurring. However, there is no benchmark dataset was constructed in the previous researches to employ in the training mode of deep learning models. Therefore, a suitable and

1 https://www.who.int/health-topics/wildfires

Ahmed S. Mahdi and Sawsen A. Mahmood, "An Edge Computing Environment for Early Wildfire Detection", *Annals of Emerging Technologies in Computing (AETiC)*, Print ISSN: 2516-0281, Online ISSN: 2516-029X, pp. 56-68, Vol. 6, No. 3, 1st July 2022, Published by International Association for Educators and Researchers (IAER), DOI: 10.33166/AETiC.2022.03.005, Available: http://aetic.theiaer.org/archive/v6/v6n3/p5.html.

collected wildfire dataset provided with ground truth of fire events is constructed and adopted in this research. The main contributions of this research can be summarized as follows:

1. Provide a useful wildfire dataset supported by ground truth (labelled frames) to evaluate the proposed system. A wildfire images dataset was constructed and adopted in our framework which collected from internet videos. An annotation process is applied over each frame to provide the ground truth labelled frames.
2. An improved structure of YOLOv5 deep learning model is implemented to achieve higher accuracy detection of fire events along with video sequence on both offline (local images and videos) and real-time testing modes. Commonly, YOLOv5 network is based mainly on the COCO dataset[2] in the training phase, which includes specific classes such as (cars, faces, bicycles, aeroplanes…etc.). Based on our experiments, the wildfire class does not exist in the COCO dataset. As a result, the trained model of YOLOv5 network could be achieved detection accuracy of fire objects by around 75%. Therefore, we have suggested using YOLOv5 network for object detection task with new initialization of weights within [0, 1] randomly to get a higher detection rate of wildfire objects in the acquired images. Further, the dropout regularization technique is added at the last layer to overcome the overfitting drawbacks.
3. Performing multi-object detection on a single frame automatically detecting the fire event on the acquired planetary images.

The rest of this paper is organised as follows; the most related works are stated in section 2, the proposed methodology is presented in section 3, while the experimental results and discussions are presented in section 4, the main conclusions of this paper are illustrated in section 5.

2. Related Works

In this section, the previous and most related works is presented. Many studies had been presented in Wildfire Detection which adopted the common CNN architecture, such as U-Net [2-3], Alex Net [4-5], Google Net [6], and YOLO (You Only Look Once) [7-8] in addition to the researches that used R-CNN [9].

Jiao *et al.* developed a wildfire detecting algorithm based on UAV-based aerial imagery using YOLOv3. The available processing capacity on the onboard hardware is used to create small-scale CNNs. The algorithm's recognition rate is around 0.83, the detection frame rate can exceed more than 3.2 frames per second. According to the testing results, the method has a lot of benefits for using UAVs to detect forest fires in real-time. The disadvantage of this algorithm is that it is sensitive to large-area wildfires [10]. On other hand the study [11] proposes Edge and Fog computing fundamentals to the UAV-based wildfire detection system domain via a hierarchical architecture, this ecosystem brings together cloud computing's powerful resources, fog computing's rich resources, and UAVs' sensing abilities, the key challenges posed by the early wildfire detection are effectively addressed by these layers' efficient collaboration.

Based on the Google Net architecture, researchers in the study [6] presented a low wildfire detection CNN framework for surveillance cameras. It has a fair computational complexity and is fit for the intended purpose when compared to other computational cost networks such as Alex Net. Experiment results on benchmark fire datasets demonstrate the suggested framework's effectiveness, as well as its appropriateness for wildfire detection in CCTV surveillance systems compared to current methods. The issue with this study is that it requires a computer with high specifications to operate. Zhang *et al*. used a CNN-based spatial prediction model for wildfire susceptibility to eliminate the class imbalance, over-sampling was employed, and the (training, validation) sample libraries were built using proportional stratified sampling, to improve prediction accuracy for wildfire susceptibility prediction, a CNN architecture was built and hyperparameters were tuned. The test dataset was then fed into the trained model, which was used to create a spatial prediction map of wildfire susceptibility in Yunnan Province. Lastly, statistical metrics such as the Wilcoxon signed-rank test and the receiver operating characteristic curve are used, and the area under the curve was used to evaluate the proposed model's prediction performance (AUC), The limitation of this study is that the effects of various architectures of CNN, such

2 https://www.cocodataset.org

as VGG-net, RES-net and GoogleNet, on wildfire prediction results have not been thoroughly investigated [5].

Hossain *et al*. presented a new method for detecting forest fires based on one ANN and a colour and multi-colour space local binary pattern of both fire and smoke signatures. The images used in this paper for training and evaluation mainly were recorded from UAVs in difficult conditions such as minuscule flame pixels, varying lighting and range, complicated backgrounds while keeping a processing speed at 19 fps, occluded fire and smoke regions, and smoke merging into the background, the proposed approach got F1 ratings of 0.84 for fire and 0.90 for smoke. It outperformed support vector machines, random forests, Bayesian classifiers, and YOLOv3 in detecting difficult fire and smoke regions of various sizes, colours, textures, and opacity. The primary limitation is that the weight versus loss graph does not have a pleasing form in real [12]. The researchers in [13] used distributed sensor networks, to create a low-power, a low-cost monitoring system for a wildfire that included a camera as well as humidity and smoke sensors, in the system's implementation, determining the differences between images captured with and without fire/smoke was a big issue, CNNs are a viable solution for processing camera images and detecting a wildfire/smoke situation, using a large wildfire image database resulting a high degree of accuracy.

Pan *et al*. [1] proposed a deep CNN for wildfire detection using cameras. To improve the fire detection rate, they used transfer learning to neural network train and a window-based analysis strategy to improve system performance. The frequency response of the filters in the dense layer and convolutional layer is calculated. They eliminate filters with low-energy impulse responses. In addition, they use the cosine similarity measure in the frequency domain to compare convolutional filters in the Fourier domain and remove identical filters, reducing storage requirements for edge devices. They tested the neural network using a range of wildfire videos, and the pruned system outperforms the regular network in daytime wildfire detection, as well as some nighttime wildfire video clips[14]. In the study [3] the researchers provide a drone-collected fire image dataset from Arizona pine forest prescribed burning, the dataset contains video footage as well as thermal heatmaps captured by infrared sensors, and researchers can easily apply their wildfire detection and modelling algorithms by (annotating and labelling) captured videos and images frame by frame, the precision and recall of our FLAME method approached 92 % and 84 %, respectively.

Bjånes *et al*. [15] established a novel Ensemble model focused on 2 previously presented deep learning networks, which produced impressive results for wildfire susceptibility risks. Their model is compared to each sub-model, two other deep learning networks, and two other machine learning benchmarks, XGBoost and SVM. To detect a wildfire at an early stage. The researchers in [16] proposed combining fog computing and CNN with Unmanned Aerial Vehicles (UAVs). Because of its demonstrated ability for such recognition tasks, a high-efficiency CNN model was utilized to recognize fire images. As a result of the proposed model's use of Alex Net and other architectures, image analysis tasks have improved in capability, to the point where a pre-trained model can perform as well as a primate. By employing those architectures, they trained the model and applied it on a fog device, as a result of which response time and accuracy are improved.

3. Methodology

The presumptive scenario presented in this study aims to locate the problem of early wildfire detection using YOLOv5 deep learning model within edge computing infrastructure. Initially, we built a dataset composed of wildfire images named the WILDFIRE-I dataset based on our efforts. Then, a modified deep learning model based on YOLOv5 network is adopted in our framework to train, validate and test the modified model. All the processing regards the captured images or videos are handled at the edge computing level including; multi-object detection of a fire event at a single frame in terms of the binary classification task. Finally, the images with fire objects will be sent to the cloud server (Google drive) which represents the decision-making level. Fig.1 visualizes the workflow of the wildfire detecting system based on YOLOv5 deep learning model.

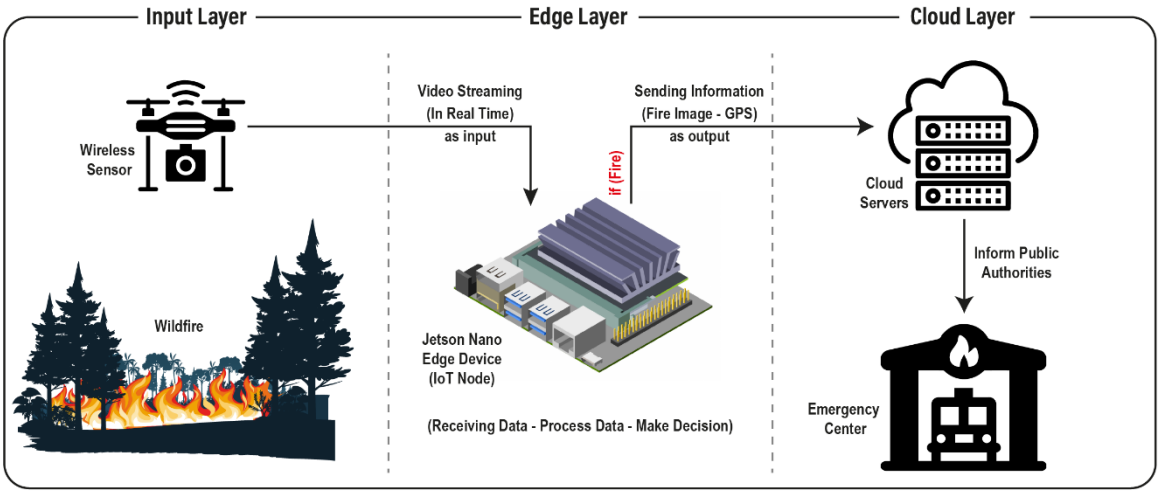

Figure 1. Proposed wildfire detection system workflow

3.1. System Requirements

3.1.1. Hardware Requirements

Dealing with deep learning methods requires higher specifications. The essential hardware specifications used in the proposed system are; Lenovo laptop with (CPU: Ryzen 5, GPU: RTX 3050 4 GB, RAM: 16 GB), In order to simulate the framework of edge computing Infrastructure we have used NVIDIA Jetson Nano Developer Kit with these specifications[3]: (GPU 128-core Maxwell, CPU Quad-core ARM A57 1.43 GHz, Memory 4 GB 64-bit LPDDR4 25.6 GB/s), Logitech USB Camera: 5MP, 720 HD, 5V 3A Power Supply Charger Type-C / Micro USB, SanDisk 64GB Ultra Micro SD HC Class 10 Memory Card and USB WIFI Adapter. These specifications are available at minimum cost.

3.1.2. Software Requirements

The proposed system required specific software applications for implementation such as: (OS: Windows 10 on a laptop, Ubuntu 18.04 on Jetson Nano, and Programming Language: Python). Additional Software Requirements are: (PyCharm Community, Anaconda 3, PyTorch 1.9 with Torchvision 0.11.0, Cuda 10.2, Last Version of Python Libraries, Image Annotation Lab and FastStone Photo Resizer (for Image processing).

3.2. Dataset Construction

The process of constructing the wildfire images dataset involved four main steps; images collection, pre-processing techniques to eliminate noise and correct inconsistencies, images annotation and data splitting step. The collected dataset is available on Mendeley Website[4]. Fig. 2 illustrates the main steps required to build the WILDFIRE-I dataset.

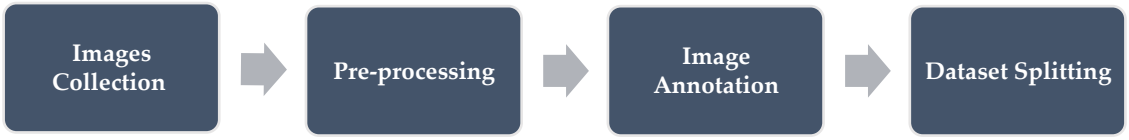

Figure 2. WILDFIRE-I dataset construction workflow

3.2.1. Images Collection

The process of images collection was performed through searching over the Internet about Wildfire images taking into consideration the following concepts; view angles (Top, Side), perspectives (Day, Night) addition to variant size of images and distances for more significant variance. We have faced the problem of limited Wildfire images required for training mode [13,17], and there is no standard wildfire dataset to utilize [4,18].

3 https://developer.nvidia.com/embedded/jetson-nano-developer-kit
4 https://data.mendeley.com/datasets/9kz5pfw4xm/3

The constructed dataset WILDFIRE-I is composed of 3,436 images taken from different sources such as:

1. www.forestryimages.org, "Fire"
2. www.nnmtl.cn/EFDNet , "Forest Fire".
3. Images from the internet, keywords: "Wildfire, Wildfire from the top, Wildfire side view, Forest fire", Day and Night Scenes.
4. Selected frames from YouTube videos, keywords: "Wildfire, Wildfire captured with drones".

3.2.2. Preprocessing

The input images are preprocessed to get high-quality images through noise removal and correct inconsistencies. Image quality always plays an essential role in object recognition methods. A higher image quality gives a better detection rate than any unprocessed noisy image [19]. Thus, extracting features from unprocessed photos is problematic because it effects the rate of object detection or categorization. Often, pre-processing techniques are applied before the feature extraction process to assist machine learning methods to get a higher detection rate with cleaned images samples. We have used the following pre-processing techniques for the input images:

1. Ignore images with sizes smaller than 224×224.
2. Convert input images into JPG type using the FastStone Photo application.
3. Resize the import images to a fixed size: 224×224 using the FastStone Photo Resizer application.

3.2.3. Image Annotation

The annotation process involves drawing a bounding box around the interesting object in each image and determining object position (coordinates) even with presenting more than one interesting object in the image[5]. Then, the class label is identified as a parameter in our framework such as; (class 0- for a fire object) and (class 1- for a non-fire object) to achieve the labelling process of the acquired images.

In our framework, the rectangular bounding box technique is utilized for the annotation process because it is commonly used in computer vision systems that provide the location of the target object. Bounding boxes are generally composed of 2 coordinates (x1, y1) and (x2, y2), or by one coordinate (x1, y1) and the bounding box's width (w) and height (h)[6], as shown in Fig. 3-a. For each image file in the same directory, a .txt file with the same name is created in YOLOv5 labelling format as shown in Fig. 3-b. Each (.txt) file includes the annotations form of the desired image file as: (object class, coordinates, and height and width).

Image Annotation Lab is adopted to annotate 6872 Images including; 3436 images with Fire class, and 3436 images with No-Fire class as shown in Fig. 3-c. The main steps of the annotation process can be summarized as follows:

- Specify class labels (Fire) and (No Fire).
- Draw a rectangular box around the fire region as Fire class, and on No-Fire objects for the No-Fire
- Select the class label for each bounding box.
- Export the annotations as Yolo format.

3.2.4. Dataset Splitting

To evaluate the modified deep learning model, the WILDFIRE-I dataset is divided into 80% for the training mode and 20% for the testing mode, as clarified in Table1. The dataset splitting operation is implemented with the assistance of Python code that randomly splits the dataset into training and validation sets with given ratios.

Table1. Dataset Splitting Ratio and number of images in each category

Category	Training 80%		Testing 20%	Total
	Train 80%	Valid 20%		
Fire	2199	550	687	3436
No Fire	2199	550	687	3436
Total	4398	1100	1374	6872

5 https://viso.ai/computer-vision/image-annotation/

6 https://towardsdatascience.com/image-data-labelling-and-annotation-everything-you-need-to-know-86ede6c684b1

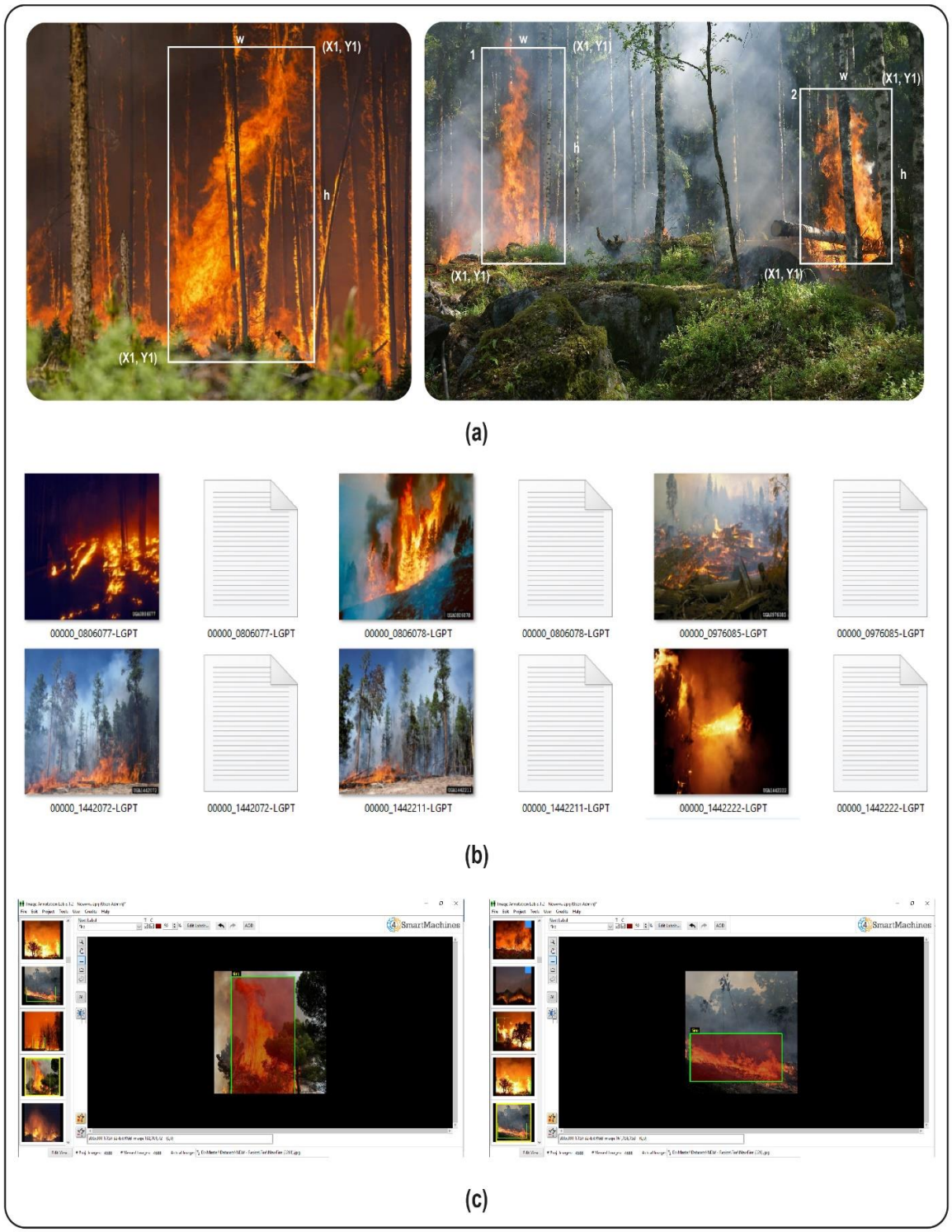

Figure 3. Simulation results of annotation process (a) Annotation coordinates (b) Annotation files (image and text) (c) Image Annotation Lab Software

3.3. Learning Model – Based Wildfire Detection

Specifically, the wildfire detection task in the acquired images or video sequence is considered an object detection problem that must be addressed and resolved. This study adopts YOLOv5 learning model for wildfire detection tasks due to its effective and fastest learning model-based object detection and classification tasks[7]. Object detectors based on CNNs are primarily used in recommendation systems,

7 https://betterprogramming.pub/machine-learning-model-api-using-yolov5-with-fast-api-192f1290a982

which could be used to detect objects in a real-time manner with lower processing power[8]. Practically, the YOLO network workflow divides the input image into grids, each of which detects objects within its borders. Fig. 4 illustrates the main structure of the original YOLOv5 architecture.

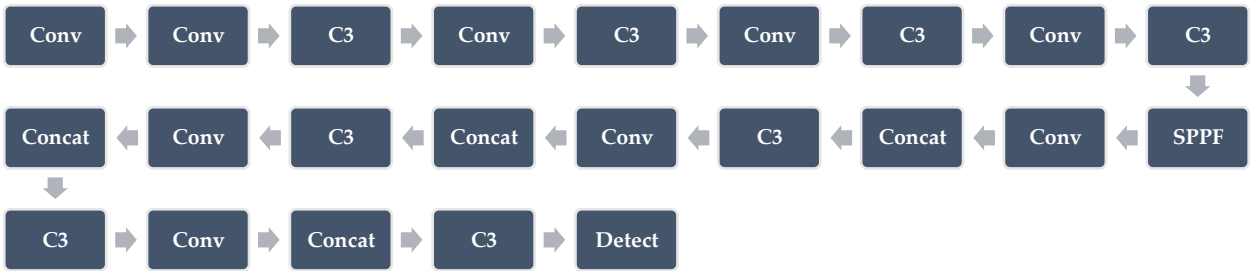

Figure 4. YOLOv5s Model Architecture

In Table 2, the description of each term mentioned in Fig. 4 (YOLOv5 structure) is stated[9]:

Table2. YOLOv5s Model Architecture Terms Explanation

Term	Description
Conv	Standard Convolution Layer
C3	CSP Bottleneck with 3 Convolutions Layers, consists primarily of three (Conv) modules and a module that is cascaded by multiple Bottlenecks.
Concat	A splicing layer is used to join two layers together.
SPPF	Spatial Pyramid Pooling Layer
Detect	The Network Output

To obtain an accurate detection rate with lower processing time, we have initialized the weights randomly within the range [0, 1] and added a dropout layer at the last layer of YOLOv5 to minimize the activated nodes and yet lower consuming time the time for detection result. The dropout layer's fundamental advantage is that it prevents all neurons in a recent layer from maximizing their weights simultaneously, as shown in Fig. 5. This adaptation, achieved in random groupings, prevents all neurons from converging to the same objective, thereby de-correlating the weights. The second attribute revealed for the use of dropout is that the activations of the hidden units become sparse, which is also a desired trait[10].

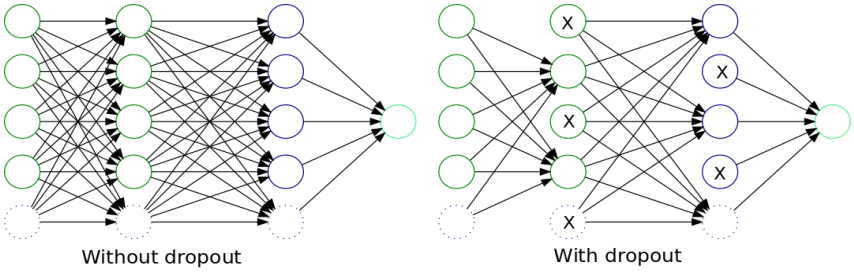

Figure 5. Dropout Layer Advantages

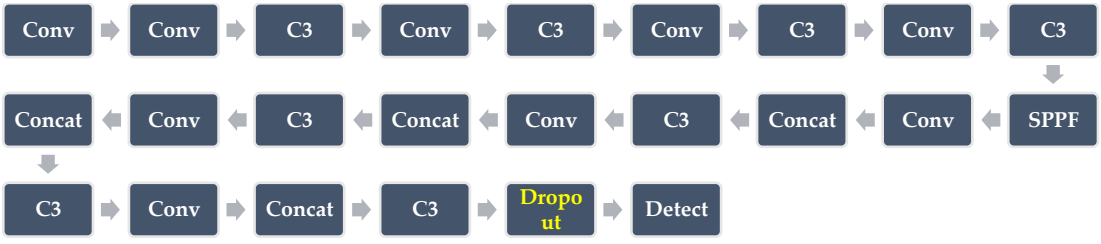

Figure 6. YOLOv5 Model with Dropout layer Architecture

3.4. Edge Computing Level

Realizing the processing data in a place close to the data source and its destination, edge computing infrastructure is hopeful of handling the problems in different delay-critical applications, such as real-time

8 https://medium.com/analytics-vidhya/object-detection-algorithm-yolo-v5-architecture-89e0a35472ef
9 https://www.fatalerrors.org/a/0Nt90T4.html
10 https://www.oreilly.com/library/view/machine-learning-for/9781786469878/252b7560-e262-49c4-9c8f-5b78d2eec420.xhtml

human surveillance. Using ubiquitously placed cameras and smart devices permits video analysis at the edge server. We have suggested using the NVIDIA Jetson Nano Developer kit as an edge computing environment [20] equipped with Docker Container to install a trained YOLOv5 model-based wildfire detection system.

This way, the video would be streamed through the camera attached to the accelerator system. The YOLOv5 code is written in Python, and the deep learning framework is used in PyTorch. To optimize the neural network layers NVIDIA TensorRT is used for faster inference during runtime. NVIDIA TensorRT is based on NVIDIA CUDA 10.2 for parallel computing.

In this paper, we have employed Nvidia Jetson Nano Developer Board as edge computing infrastructure due to its advantages which include a GPU and enabling to run deep learning model networks quickly. Fig. 7 describes the Nvidia Jetson Nano Developer Kit.

Figure 7. Nvidia Jetson Nano Developer Kit [11]

3.5. Cloud Computing Level

At the cloud computing level of the proposed system, the fire images and locations are sending to the relevant authorities by uploading the target images to the cloud. Our approach used Google Drive to deliver data quickly and ensure a quick response. Google Drive can connect a user-end client application to the company's services via a virtual network by launching virtual machine instances that can be easily connected to Internet gateways, subject to firewall and access control rules, offer programmable interfaces for managing virtual machine instances and changing access control rules [21], experiments show that sending images to the cloud takes about 1.7 seconds in Real-Time and 1.0 second with local images.

4. Experiments and Evaluation

The performance evaluation of the proposed wildfire detection system is conducted based on the WILDFIRE-1 dataset. The confusion matrix[12] benchmark is utilized in our evaluation which included four main metrics (TP, TN, FP, FN) to give a clear simulation result as shown in Fig. 8. It enables us to assess how well our model performed, identify where it went wrong, and provide guidance on correcting our course.

4.1. Performance Evaluation Metrics[13]

4.1.1. Accuracy

Accuracy metric refers to the total number of correct predictions divided by the total number of predictions made of the entire dataset predictions and calculated according to Eq. 1:

$$\text{Accuray} = \frac{\text{TP+TN}}{\text{TP+TN+FP+FN}} \quad (1)$$

4.1.2. Precision

It is defined as the ratio of true positives to the total positives predicted by object detection model and calculated according to Eq. 2:

$$\text{Precision} = \frac{\text{TP}}{\text{TP+TN}} \quad (2)$$

11 https://developer.nvidia.com/embedded/jetson-nano-developer-kit

12 https://www.analyticsvidhya.com/blog/2020/04/confusion-matrix-machine-learning/

13 https://www.kdnuggets.com/2020/04/performance-evaluation-metrics-classification.html

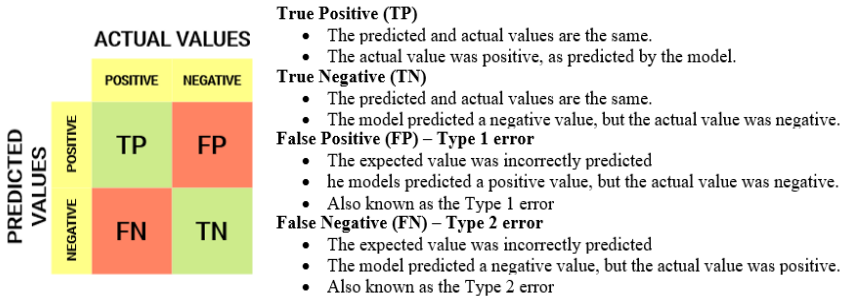

Figure 8. Confusion Matrix with 2 classes [14]

4.1.3. Recall

It is defined the number of real positive outcomes divided by the number of actual positive outcomes and calculated using Eq. 3:

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

4.1.4. F1 Score

F1-Score refers to the harmonic mean of precision and recall and calculated using Eq. 4:

$$\text{F1 Score} = \frac{2*(\text{precision}*\text{recall})}{(\text{precision}+\text{recall})} \tag{4}$$

4.2. Experiments Implementation

To confirm the effectivity of the modified YOLOv5 network-based wildfire detection system, we have achieved two experiments including:

4.2.1. Experiment 1

In this experiment, YOLOv5 and YOLOv3 networks were implemented separately using (WILDFIRE-I) dataset. In the training mode, both networks achieved an optimized configuration of hyperparameters while preserving the original structure. The obtained results of this experiment are illustrated in Figure 9. Based on the experiment 1 results, the training loss was close to 0.03 with precision 84% and Recall of 99% for YOLOv5 deep learning model, while the training loss was 0.03 with precision 83% and Recall 81% for YOLOv3 deep learning model. Subsequence, the trained models YOLOv5 and YOLOv3 networks were saved to employ later (separately) at the edge computing infrastructure (Jetson Nano) to detect the wildfire object in the streaming video. The experimental results of experiment 1 for the testing mode was obtained based on implementing the trained models of YOLOV5 and YOLOv3 networks separately, as clarified in Table 3.

Table 3. Test phase results of YOLOv5 and YOLOv3

Term		YOLOv5 On Laptop	YOLOv5 On Jetson Nano	YOLOv3 on Laptop
Test Time	pre-process (ms)	0.3	1.8	0.2
	Inference (ms)	9.7	77.1	13.2
	NMS per image (ms)	1.1	10.1	0.9
True Positives		685	685	676
True Negatives		672	672	673
False Positives		2	2	11
False Negatives		15	15	14
Sensitivity		97.86%	97.86%	97.97%
Specificity		99.70%	99.70%	98.39%
Precision		99.71%	99.71%	98.40%
Accuracy		98.76%	98.76%	98.18%
F1 Score		98.77%	98.77%	98.18%

4.2.2. Experiment 2

In this experiment, the modified YOLOv5 network was trained with dropout layer and implemented to achieve the desired object (fire object) detection task. The qualitative results of training mode-based on

14 https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62

exterminate 2 are visualized in Fig. 10, while the quantitate results of the testing mode are illustrated Table 4.

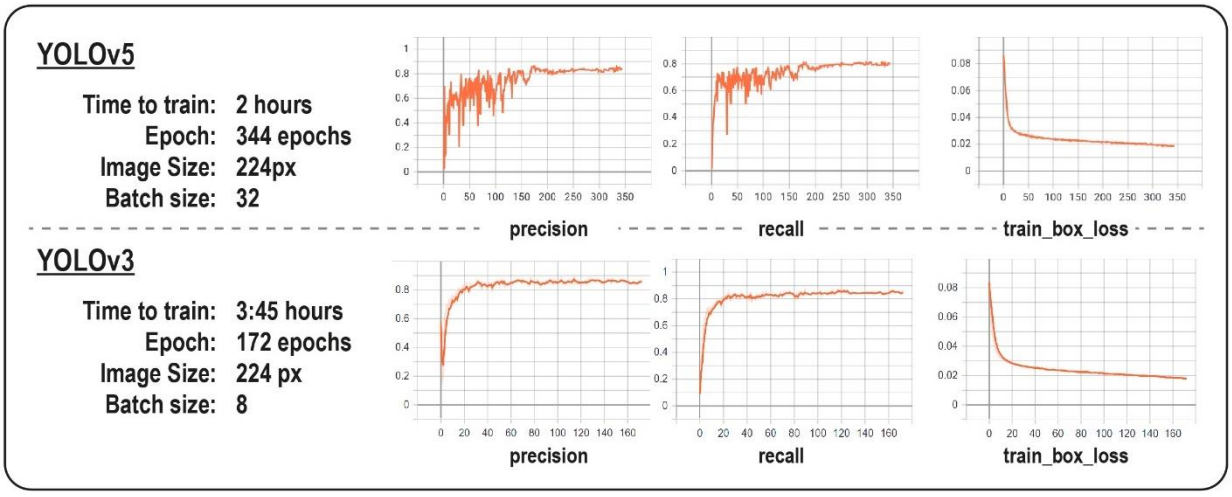

Figure 9. Results demonstration based on weights initialization and hypermeter configuration of YOLOv5 and YOLOv3 networks

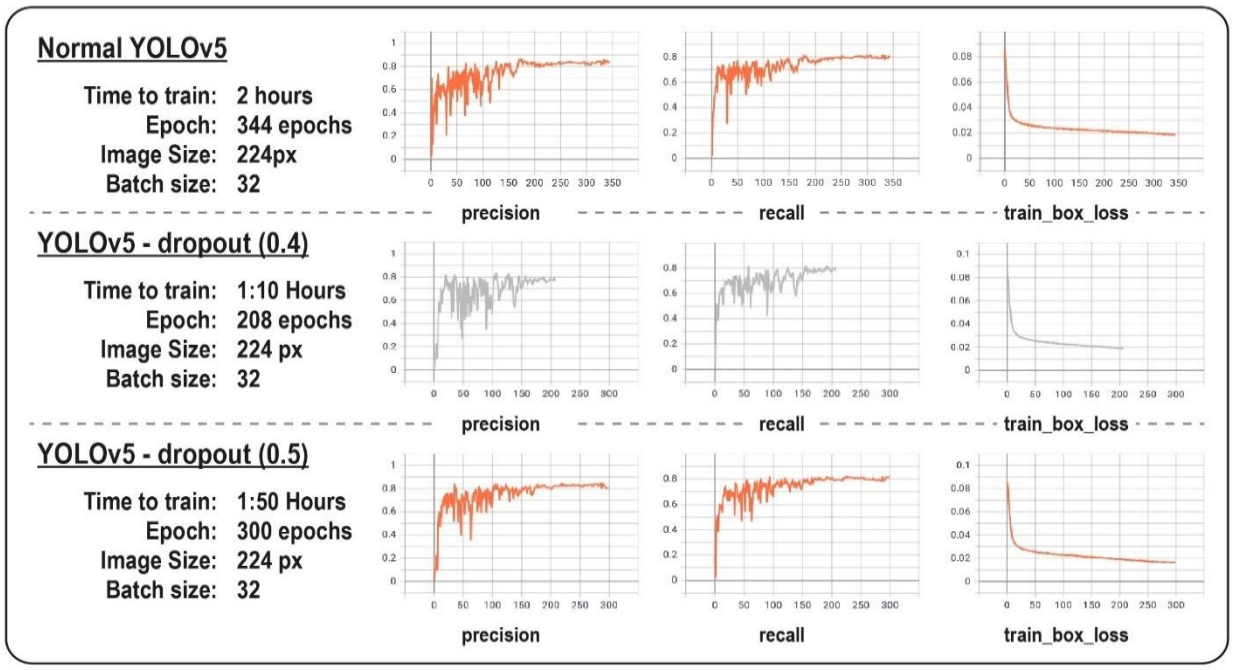

Figure 10. Training results of modified YOLOv5

Table 4. Testing results of YOLOv5s and modified YOLOv5

Term		YOLOv5 On Laptop	dropout (0.4) on Laptop	dropout (0.4) on Jetson Nano
Test Time	pre-process (ms)	0.3	0.2	1.4
	Inference (ms)	9.7	9.5	66.0
	NMS per image (ms)	1.1	0.7	8.1
True Positives		685	676	676
True Negatives		672	665	665
False Positives		2	11	11
False Negatives		15	22	22
Sensitivity		97.86%	96.85%	96.85%
Specificity		99.70%	98.37%	98.37%
Precision		99.71%	98.40%	98.40%
Accuracy		98.76%	97.60%	97.60%
F1 Score		98.77%	97.62%	97.62%

4.3. Comparison Study

Based on our knowledge, the first step in this study was to select an appropriate and effective deep learning model to conduct fire object detection task in the input images or video sequence. Thus, YOLOv5 and YOLOv3 models were selected, trained and evaluated using WILDFIRE dataset. Based on the obtained results shown in Fig. 11 and table 2, we have employed YOLOv5 deep learning model in the proposed system. YOLOv3 network takes longer time to train the entire dataset samples compared toYOLOv5 network. However, Yolov3 uses a smaller number of Epochs to be trained when compared with YOLOv5 due to epoch or batch size used in each model structure. Through experimental results and testing phase, we found that YOLOv5 outperformed YOLOv3 in term of accuracy, which is the most important evaluation metric we want to achieve, as shown in Fig. 11.

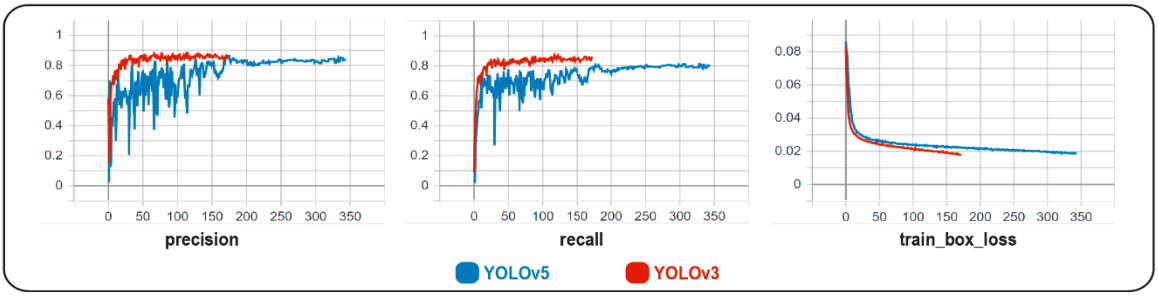

Figure 11. Comparison results of the training phase for YOLOv5 and YOLOv3 networks

On the other hand, after selecting the training model, the study took it upon itself to improve the algorithm by adding a dropout layer to the original structure of Yolov5. Several experiments were conducted to reach to the valid dropout ratio. Fig. 11 depicts the evaluation metrics for Yolov5 and Yolov3 networks with different ratios of dropout Layer. The results exhibited an improvement of training time with dropout ratio 0.5, while the dropout ratio 0.4 is the best in which trims the training time into half. The modified yolov5 specified detection accuracy of fire object close to 98% as illustrated in table 5, while the processing time was dropped to 0.4 ms for each frame. As a result, the run time was improved and decreased about 36%, especially when using real-time on edge device. Further, a comparison study between the proposed study and state of art methods (most interesting studies) was performed as stated in Table 4 to show and demonstrate the progress of the proposed study in terms of accuracy, precision and recall evaluation metrics.

Table 5. Comparison study between proposed study and state of art methods

Study	Method	Dataset	Accuracy	Precision	Recall
[3]	ANN	FLAME Dataset	76 %	92 %	84 %
[6]	CNN	Foggia Dataset	94 %	82 %	98 %
[7]	YOLOv2	Collected Dataset	96 %	97 %	97 %
[10]	YOLOv3	Collected Dataset	83 %	82 %	79%
[11]	CNN + ANN	NASA Global Land Data Assimilation System (GLDAS)	95%	93%	95%
[12]	ANN	UAV Image	89 %	89 %	80 %
[15]	CNN	GIS database	90 %	90 %	87%
Our Study	YOLOv3	WILDFIRE-I	98 %	84%	99%
Our Study	YOLOv5	WILDFIRE-I	99 %	83 %	81%
Our Study	YOLOv5 with dropout layer	WILDFIRE-I	98 %	88 %	99 %

5. Conclusion

We have presented a proposed wildfire detection system based on YOLOv5 deep learning model within edge computing infrastructure. The following conclusions were reached as a result of our experiments; advisedly that YOLOv5 takes a lower time than the YOLOv3 method in the training mode as shown in Fig. 11 and Table 3. Based on our experiments YOLOv3 obtained higher speed detection, while YOLOv5 method obtained a higher detection accuracy. The batch size parameter was effect mainly to get higher accuracy detection. Computer specifications used in the training process vary significantly in terms of time and training results. Using a GPU makes a significant difference in AI applications. Adding a dropout layer with a (0.4) ratio was much better than adding a dropout layer with a (0.5) ratio in terms of

training time and detection time. The small size of the images contributed to minimizing the processing time of training and testing processes.

Acknowledgements

This research was supported by Mustansiriyah University hosted by the Department of Computer Science in the college of education, Baghdad, Iraq.

References

[1] Hongyi Pan, Diaa Badawi and Ahmet Enis Cetin, "Computationally Efficient Wildfire Detection Method Using a Deep Convolutional Network Pruned via Fourier Analysis", *Sensors*, Vol. 20, No. 10, p. 2891, May 2020, DOI: 10.3390/s20102891.

[2] Gabriel Henrique de Almeida Pereira, Andre Minoro Fusioka, Bogdan Tomoyuki Nassu and Rodrigo Minetto, "Active fire detection in Landsat-8 imagery: A large-scale dataset and a deep-learning study", *ISPRS Journal of Photogrammetry and Remote Sensing*, Vol. 178, pp. 171–186, August 2021, DOI: 10.1016/j.isprsjprs.2021.06.002.

[3] Alireza Shamsoshoara, Fatemeh Afghah, Abolfazl Razi, Liming Zheng, Peter Z. Fulé *et al*., "Aerial Imagery Pile Burn Detection Using Deep learning: the FLAME Dataset", *Computer Networks*, Vol. 193, p. 108001, July 2021, DOI: 10.1016/j.comnet.2021.108001.

[4] Qingjie Zhang, Jiaolong Xu, Liang Xu and Haifeng Guo, "Deep Convolutional Neural Networks for Forest Fire Detection", in *Proceedings of the 2016 International Forum on Management, Education and Information Technology Application*, Guangzhou, China, January 2016, pp. 568–575. DOI: 10.2991/ifmeita-16.2016.105.

[5] Guoli Zhang, Ming Wang and Kai Liu, "Forest Fire Susceptibility Modeling Using a Convolutional Neural Network for Yunnan Province of China", *International Journal of Disaster Risk Science*, Vol. 10, No. 3, pp. 386–403, September 2019, DOI: 10.1007/s13753-019-00233-1.

[6] Khan Muhammad, Jamil Ahmad, Irfan Mehmood, Seungmin Rho and Sung Wook Baik, "Convolutional Neural Networks Based Fire Detection in Surveillance Videos", *IEEE Access*, Vol. 6, pp. 18174–18183, 2018, DOI: 10.1109/access.2018.2812835.

[7] Sergio Saponara, Abdussalam Elhanashi and Alessio Gagliardi, "Real-time video fire/smoke detection based on CNN in antifire surveillance systems", *Journal of Real-Time Image Processing*, Vol. 18, No. 3, November 2020, DOI: 10.1007/s11554-020-01044-0.

[8] Renjie Xu, Haifeng Lin, Kangjie Lu, Lin Cao and Yunfei Liu, "A Forest Fire Detection System Based on Ensemble Learning", *Forests*, Vol. 12, No. 2, p. 217, February 2021, DOI: 10.3390/f12020217.

[9] Qi-xing Zhang, Gao-hua Lin, Yong-ming Zhang, Gao Xu and Jin-jun Wang, "Wildland Forest Fire Smoke Detection Based on Faster R-CNN using Synthetic Smoke Images", *Procedia Engineering*, Vol. 211, pp. 441–446, 2018, DOI: 10.1016/j.proeng.2017.12.034.

[10] Zhentian Jiao, Youmin Zhang, Jing Xin, Lingxia Mu, Yingmin Yi *et al*., "A Deep Learning Based Forest Fire Detection Approach Using UAV and YOLOv3", in *Proceedings of the 1st International Conference on Industrial Artificial Intelligence (IAI)*, Shenyang, China, July 2019, pp. 1–5. DOI: 10.1109/ICIAI.2019.8850815.

[11] Guoli Zhang, Ming Wang and Kai Liu, "Deep neural networks for global wildfire susceptibility modelling", *Ecological Indicators*, Vol. 127, p. 107735, August 2021, DOI: 10.1016/j.ecolind.2021.107735.

[12] F. M. Anim Hossain, Youmin M. Zhang and Masuda Akter Tonima, "Forest fire flame and smoke detection from UAV-captured images using fire-specific color features and multi-color space local binary pattern", *Journal of Unmanned Vehicle Systems*, Vol. 8, No. 4, pp. 285–309, December 2020, DOI: 10.1139/juvs-2020-0009.

[13] Yuanbin Wang, Langfei Dang and Jieying Ren, "Forest fire image recognition based on convolutional neural network", *Journal of Algorithms & Computational Technology*, Vol. 13, p. 174830261988768, January 2019, DOI: 10.1177/1748302619887689.

[14] Sawsen Abdulhadi Mahmood, Azal Monshed Abid and Sadeq H. Lafta, "Anomaly event detection and localization of video clips using global and local outliers", *Indonesian Journal of Electrical Engineering and Computer Science*, Vol. 24, No. 2, p. 1063, November 2021, DOI: 10.11591/ijeecs.v24.i2.pp1063-1073.

[15] Alexandra Bjånes, Rodrigo De La Fuente and Pablo Mena, "A deep learning ensemble model for wildfire susceptibility mapping", *Ecological Informatics*, Vol. 65, p. 101397, November 2021, DOI: 10.1016/j.ecoinf.2021.101397.

[16] Kethavath Srinivas and Mohit Dua, "Fog Computing and Deep CNN Based Efficient Approach to Early Forest Fire Detection with Unmanned Aerial Vehicles", in *International Conference on Inventive Computation Technologies*, Coimbatore, India, November 2019, Vol. 98, pp. 646–652. DOI: 10.1007/978-3-030-33846-6_69.

[17] Abdelmalek Bouguettaya, Hafed Zarzour, Amine Mohammed Taberkit and Ahmed Kechida, "A review on early wildfire detection from unmanned aerial vehicles using deep learning-based computer vision algorithms", *Signal Processing*, Vol. 190, p. 108309, January 2022, DOI: 10.1016/j.sigpro.2021.108309.

[18] Yanhong Chen, Youmin Zhang, Jing Xin, Guangyi Wang, Lingxia Mu *et al*., "UAV Image-based Forest Fire Detection Approach Using Convolutional Neural Network", in *14th IEEE Conference on Industrial Electronics and Applications (ICIEA)*, Xi'an, China, June 2019, Vol. 2019. DOI: 10.1109/iciea.2019.8833958.

[19] Hironobu Fujiyoshi, Tsubasa Hirakawa and Takayoshi Yamashita, "Deep learning-based image recognition for autonomous driving", *IATSS Research*, Vol. 43, No. 4, pp. 244–252, December 2019, DOI: 10.1016/j.iatssr.2019.11.008.

[20] Sebastián Valladares, Mayerly Toscano, Rodrigo Tufiño, Paulina Morillo and Diego Vallejo-Huanga, "Performance Evaluation of the Nvidia Jetson Nano Through a Real-Time Machine Learning Application", in *International Conference on Intelligent Human Systems Integration*, Palermo, Italy, February 2021, pp. 343–349. DOI: 10.1007/978-3-030-68017-6_51.

[21] Hussain M. J. Almohri and Mohammad Qasem, "An Efficient Deception Architecture for Cloud-based Virtual Networks", *Kuwait Journal of Science*, Vol. 46, No. 3, August 2019.