OPEN ACCESS
Indo-JC Indonesia Journal of Computing
ISSN 2460-9056
socj.telkomuniversity.ac.id/indojc

Ind. Journal on Computing
Vol. 7, Issue. 2, August 2022. pp. 65-78
doi:10.34818/indojc.2022.7.2.654

Video Based Fire Detection Method Using CNN and YOLO Version 4

Muhammad Salman Farhan [#1], Febryanti Sthevanie [#2], Kurniawan Nur Ramadhani [#3]

#School of Computing, Telkom University
Jl. Telekomunikasi No. 1, Bandung 40257, Indonesia

[1] salmanfarhan@student.telkomuniversity.ac.id
[2] sthevanie@telkomuniversity.ac.id
[3] kurniawannr@telkomuniversity.ac.id

Abstract

Fire detection is one of the technological efforts to prevent fire incidents, this is very important because the damage caused by fires can be minimized by having a fire detector. There are two types of fire detection, namely traditional-based and computer vision-based. Traditional-based fire detection has many shortcomings, one of which requires a close fire distance for activation. Hence, computer vision-based fire detection is made to cover the shortcomings of traditional-based fire detection. Therefore, in this research, we propose a video-based fire detection using a Convolutional Neural Network (CNN) Deep Learning approach supported by You Only Look Once (YOLO) object detection model version four. This research used a dataset of various fire scenarios in the form of images and videos. The fire detection built in this research has an accuracy of above 90% with an average detection speed of 34.17 Frame Per Second (FPS).

Keywords: CNN, Deep Learning, Fire Detection, Object Detection, YOLO

Abstrak

Mendeteksi api merupakan salah satu upaya teknologi untuk mencegah terjadinya kebakaran, hal ini sangat penting karena dengan memiliki pendeteksi api kerusakan yang ditimbulkan oleh kebakaran dapat diminimalisir. Terdapat terdapat dua jenis pendeteksi api, yaitu berbasis tradisional dan berbasis visi komputer. Pendeteksi api berbasis tradisional memiliki banyak kekurangan salah satunya membutuhkan jarak api yang dekat untuk aktivasinya sehingga dibuat sebuah pendeteksi api berbasis visi komputer untuk menutupi kekurangan yang dimiliki oleh pendeteksi api berbasis tradisional. Oleh karena itu, pada penelitian ini membuat pendeteksi api berdasarkan video menggunakan pendekatan visi komputer berbasis *deep learning Convolutional Neural Network* (CNN) yang didukung oleh model deteksi objek *You Only Look Once* (YOLO) versi empat. Penelitian ini menggunakan dataset berbagai skenario api dalam bentuk citra dan video. Pendeteksi api yang dibangun pada penelitian ini memiliki akurasi di atas 90% dengan kecepatan deteksi rata-rata 34,17 *Frame Per Second* (FPS).

Kata Kunci: CNN, Deep Learning, Deteksi Objek, Mendeteksi Api, YOLO

I. Introduction

Fire can quickly cause significant property damage as well as injuries to living things [1]. Fires can also start at variety of place, from forest to warehouses. This has led to the development of fire detection technologies for preventing and mitigating the impacts of these fires. Therefore, the purpose of this research is to build fire detection as an early warning tool to reduce and prevent the impact of damage caused by fires.

In general, fire detection can be divided into two categories, namely traditional fire detection and computer vision fire detection. Traditional fire detection techniques used smoke or heat sensors and require proximity to objects to activate. These sensors also require human intervention to confirm a fire has broken out. Furthermore,

Received on April, 2022. Accepted on July, 2022

the system needs to collect information about fire's size, location, and temperature. To overcome this limitation, researchers have investigated computer vision-based methods in combination with various types of additional sensors [2-5]. Technologies in this category allow people to see the fire without coming to the scene and provide detailed fire information such as location, size, and temperature level, resulting in greater monitoring coverage, less human intervention, and more responsiveness. Although using the computer vision method has many advantages, some problems still occur with this method. Therefore, the researchers attempted to address the problem in terms of computer vision technology [6].

The computer vision–based fire detection method starts from the color of the fire using the Hue Saturation Intensity (HIS) and Red Green Blue (RGB) color models [7] to extract areas that have the possibility of fire to determine the fire area [8]. However, color-based fire detection is vulnerable to environmental factors such as lighting and shadows [6]. Exploring the characteristics and dynamics of different types of fire is very difficult and requires extensive knowledge of fire. However, this can be substituted by deep learning approaches with sufficient data to avoid overfitting. CNN is commonly used in fire detection as a deep learning model. However, the lack of dataset causes CNN not to perform well. After introducing the Imagenet dataset and other datasets. The CNN model is proven to have very good and accurate performance compared to other computer vision models [9].

Object detection technology is rapidly developing to get high accuracy and speed. There are several object detections commonly used, including You Only Look Once (YOLO), EfficientDet, Single Shot MultiBox Detector (SSD), and Faster Region-Based Convolutional Neural Network (Faster-RCNN) [10]. YOLO version three was proven to have the highest accuracy and the fastest fire detection process compared to other models in the case of fire detection [11]. However, when compared to YOLO version four, the difference in performance using YOLO version four has a very significant increase, namely an increase in Average Precision (AP) by 10% and Frame Per Second (FPS) by 12% better than using YOLO version three [12]. Therefore, this research used a CNN deep learning model supported by YOLO version four to build a fire detection system.

II. Literature Review

A. CNN (Convolutional Neural Network) Architecture

LeCun (1989) explained Convolutional Neural Network (CNN) is a special type of neural network for processing data with a grid-like topology. An example of such data is an image. An image can be thought of as a two-dimensional grid of pixels. The name “Convolutional neural network” denotes a mathematical operation called convolution [12]. This research used the CNN deep learning model because deep learning has better performance than the shallow learning model and CNN is the most suitable deep learning method for this research since it detects important features automatically without any human involvement [7].

Figure 1 shows the CNN architectural design for the fire detection system.

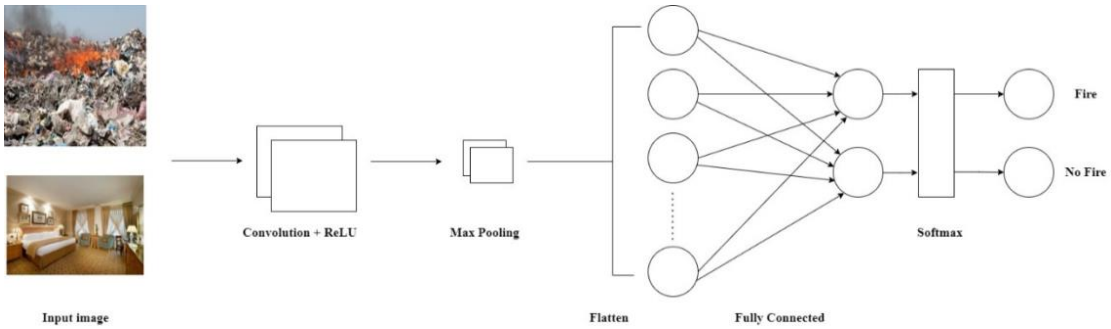

Fig. 1. CNN Architecture

The following is an explanation of the CNN architectural design for the fire detection system as shown in Figure 1:

1. Input
   Input of this system is the form of images to train the model, this component used video clips to detect fire and measure the performance of the model.
2. Convolutional Layer + ReLU
   The convolutional layer is the core layer of the CNN with RelU as the function. There are several components in this layer, namely input data, filters, and feature maps. One example of input is a color image that has three dimensions, namely width, height, and channel. The filter moved across the image and perform a "dot" operation between the input and the value of the filter to produce an output called a feature map [10]. This process is known as convolution. The final result of this convolution process is an output array.
3. Pooling Layer
   This layer used to extract the dominant features. This layer is also used for reducing dimensions and parameters. There are two types of pooling function: maximum pooling and average pooling. Maximum pooling returns the maximum value of the image, and average pooling returns the average value of the image.
4. Fully Connected Layer
   The feature map is produced in the form of a multidimensional array from pooling layer, which must be transformed into a vector (flatten) in order to be input into a fully connected layer. Activation function softmax is used in this layer to perform classification. The final result of softmax is a probability value with a scale of zero to one.

B. Object Detection

Ivan Vasilev (2019) explained that object detection is the process of finding object instances of a particular class, such as trees, faces, and cars in videos or images. Different from classification, object detection can detect multiple objects and their position in the image. Object detection returned a list of detected objects along with object class information, probabilities, and coordinates for each object [13]. it can be interpreted that detection is the process of finding one or more objects and the location of the objects that have been found in the image or video. The results obtained from object detection are object class information and object locations.

The object detection has several models that are still frequently used, including You Only Look Once (YOLO), EfficientDet, Single Shot Multibox Detector (SSD), and Faster Region-Based Convolutional Neural Network (Faster-RCNN). The accuracy produced by the object detection models is varied because it is strongly influenced by the type and condition of the data used when modeling. In this research, we used the YOLO object detection because YOLO sees the fire is not a small object in the entire image, making YOLO one of the best choices for detecting fire [7]. This research used YOLO version four as an object detection model because YOLO version four is a state-of-the-art (best) object detection model, so it can produce high accuracy with fast performance. YOLO version four is also open-source, making it easier to configure object detection model according to research needs, and this object detection model has a very high generalization capability so the model does not require a lot of data to produce high accuracy [12].

C. YOLOv4 Architecture

You Only Look Once (YOLO) is a method of detecting objects with one stage or which can be interpreted as only seeing the image once [10]. As shown in Figure 2, the following are the four components of this one-stage process:

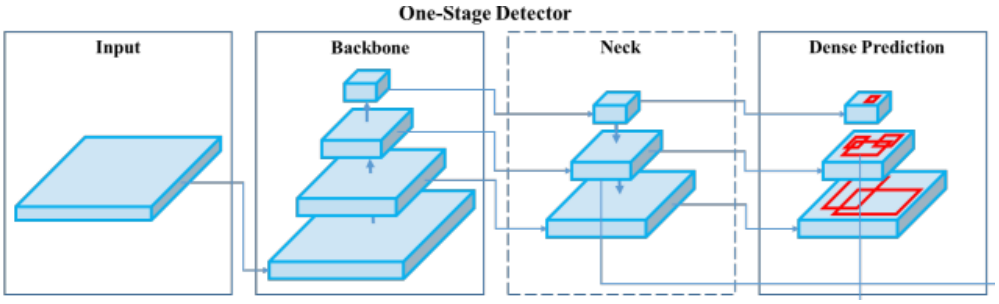

Fig. 2. Yolo Architecture

1. Input
   The Input component, the image size is adjusted according to the input resolution layer. Resolution can be determined by the rule that it can be divided by 32. In general, the image received by this component is a color image.
2. Backbone
   This component performs feature extraction. Several backbones can be used, namely CSPReNEXT50, CSPDarknet53, and Efficient-B3. In this research, we used the CSPDarknet53 backbone because this model is the most optimal model by separating the most important context features and hardly reducing the speed of network operations [12].
3. Neck
   This component works to add a layer between the backbone and dense prediction. With this component, object detection can be performed on objects of different sizes.
4. Dense Prediction
   This component determined the bounding box and classify what is contained in the bounding box. This is done by dividing the received image into a grid of cells with each cell containing an anchor box. After that, predictions are made on each anchor box.
5. Bag of Freebies
   This component YOLO used to improve model performance. Bag of Freebies (BoF) is a collection of methods that aim to improve the accuracy of the detection results without increasing inference costs. One of the method used is the method of augmentation and regularization. BoF is used in backbone and detector components.
6. Bag of Specials
   Same as BoF this component YOLO used to improve model performance. Bag of Specials (BoS) is a set of methods that aims to significantly increase the accuracy of the detection results by increasing the inference cost by a small amount. BoS is also used on the same components as BoF.

III. Research Method

The system we built is a video-based fire detection system using the Convolutional Neural Network (CNN) deep learning method supported by You Only Look Once (YOLO) version four. Figure 3 shows the architectural design for a fire detection system and Figure 7 is an architectural design for a fire detection system deployment.

A. Fire Detection System Modeling

The fire detection system modeling in this research was built based on the diagram as shown in Figure 3. All processes in the diagram were carried out in stages to avoid errors in building a model of the fire detection system. This stage is carried out to train the model in detecting fire using images.

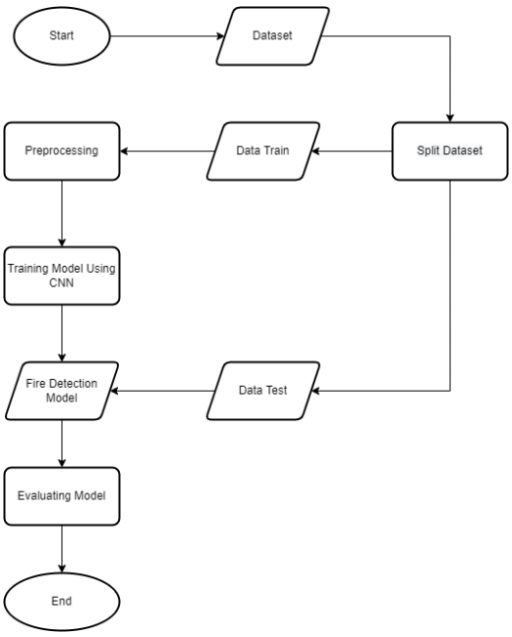

Fig. 3. Research Method Design I

1. Dataset

   In this research, we used image dataset from various fire scenarios that contain fire 1200 images, and non-fire 800 images. The datasets are divided into 1200 training set, 200 validation set, and 400 testing set.

2. Preprocessing

   The following are some of the image preprocessing method we used in this research:

   a) Histogram equalization

      Image histograms visualize the distribution of intensity in pixels, which reflects the quantity distribution in pigments within an image. We used Histogram equalization to increase the local contrast without reducing the overall contrast to highlight some details in an image [14]. The image that has been preprocessed using histogram equalization is shown in Figure 4.

Fig. 4. Comparison of Original Image (Left) and After Using Histogram Equalization (Right)

   b) Brightness Adjustment

      Enhancing brightness is one of the most effective preprocessing techniques for refining images [15]. We used brightness levels between -25% to 25% of the original image

brightness to train the model so it can detect fire at different brightness levels. Figure 5 shows the image that has been preprocessed using brightness image preprocessing.

Fig. 5. Comparison of Original Image Brightness (Middle) and After Brightness Adjustment (Left and Right)

c) Image Rotation

Rotating the image allows the model to detect fires from all angles by using image preprocessing. in this stage, we used an image rotation of 90° Clockwise and Counter-Clockwise. The image that has been preprocessed using rotate image preprocessing is shown in Figure 6.

Fig. 6. Example of Fire Image After Using Image Preprocessing Rotate

3. CNN Model Training
   a. Model Scenario

      In this research we created a model with three different scenarios, model 1 was built without implemented the image preprocessing, model 2 was built used image preprocessing and model 3 was built without implemented the image preprocessing but with a different input layer resolution size from model 1 and model 2.

   b. Hyper-parameters

      To train the model, we propose two different hyper-parameters, model 1 and 2 used 16 subdivisions with input layer resolution size of 416x416, Model 3 used a subdivision with a larger input layer resolution size due to the device's inability to train using the same subdivision as model 1, thus model 3 used 24 subdivisions with input layer resolution size of 608x608. Table I show All hyper-parameters configurations used in this research.

TABLE I
HYPER-PARAMETERS COMPARISON

Model	Model 1 (original) and Model 2 (Image Preprocessing)	Model 3 (Layer Size 608)
Batch	64	64
Subdivision	16	24
Width	416	608
Height	416	608
Momentum	0.9	0.9
Decay	0.0005	0.0005
Learning rate	0.001	0.001
Burn in	1000	1000
Max batches	6000	6000
Pretrained Weights	Yolov4.conv.137	Yolov4.conv.137

c. Training and Testing

Based on the research of N. Tajbakhsh [16] the pre-training model produce a better model than using from scratch so in this research all model were trained using the Yolov4.conv.137 pre-trained model. All training and testing processes are carried out using a Google Colab GPU with specifications describe on Table II.

TABLE II
HARDWARE SPECIFICATION

Model	Training	Testing
RAM	16 GB	16 GB
GPU	Tesla T4	Tesla T4

4. Fire Detection Model

The CNN model produced by training stage were then tested using our testing set.

5. Evaluating Model

The confusion matrix is a parameter that measure the performance of the model. As shown in Table III the number of correct and incorrect predictions are displayed in the confusion matrix table broken down by each class. The confusion matrix provided information not only on the errors made by the model but also on the type of errors [17].

TABLE III
CONFUSION MATRIX

		Predicted Class	
		Positive	Negative
Actual Class	Positive (Fire)	True Positive (TP)	False Negative (FN)
	Negative (Non-Fire)	False Positive (FP)	True Negative (TN)

The following is an explanation from Table III of the confusion matrix:

- True Positive (TP)

  TP represents the number of samples accurately predicted by the model. In this case, the model predicts that there is a fire in the video or image and it is true that there is a fire in the video or image.

- False Negative (FN)
  FN represents the number of samples that are positive but predicted negative by the model. In this case, the model predicts that there is no fire in the video or image but there is actually a fire in the video or image.
- False Positive (FP)
  FP represents the number of samples that are negative but predicted to be positive by the model. In this case, the model predicts that there is a fire in the video or image but actually there is no fire in the video or image.
- True Negative (TN)
  TN represents the number of negative samples that the model predicts accurately. In this case the model predicts that there is no fire in the video or image and it is true there is no fire in the video or image.

After obtaining the Confusion matrix, the following values can be calculated:

- Accuracy
  Accuracy measures how accurate the model is in classifying correctly (positive and negative). Here is how to calculate accuracy [17]:

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

- Precision
  Precision is the ratio of a truly positive prediction to an overall positive predicted result. Here is how to calculate precision [17]:

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

- Recall
  Recall is the ratio of a truly positive prediction compared to completely positive overall data. Here is how to calculate Recall [17]:

$$Recall = \frac{TP}{TP + FN} \quad (3)$$

- F-Measure
  F-Measure is a comparison of the average precision and recall. Here is how to calculate the F-Measure [17]:

$$F - Measure = \frac{2 * Precision * Recall}{Precision + Recall} \quad (4)$$

B. Application of Fire Detection System Modeling

The deployment modeling of the fire detection system was built following the flow of the diagram as shown Figure 7. This stage was carried out to test the fire detection that shown in Figure 3 to be applied to video for the fire detection process.

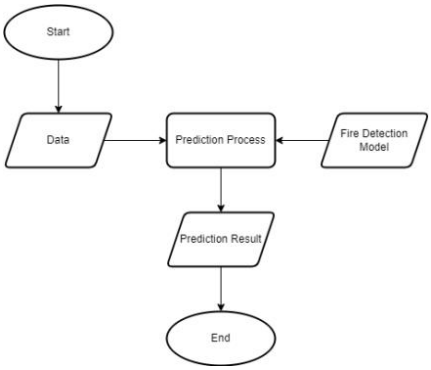

Fig. 7. Research Method Design II

1. Data
   The data used in this flowchart is an offline video.
2. Fire Detection Model
   The fire detection model has been shown in Figure 3.
3. Prediction Process
   The process of detecting fire on video used the fire detection model that shown in Figure 3 to detect the presence of fire on video.
4. Prediction Result
   The results of the prediction process with the output accuracy and frames/second from the system model.

IV. RESULTS AND DISCUSSION

The research has been completed by carrying out three different research scenarios to get the best fire detection model results.

A. Result Training Process

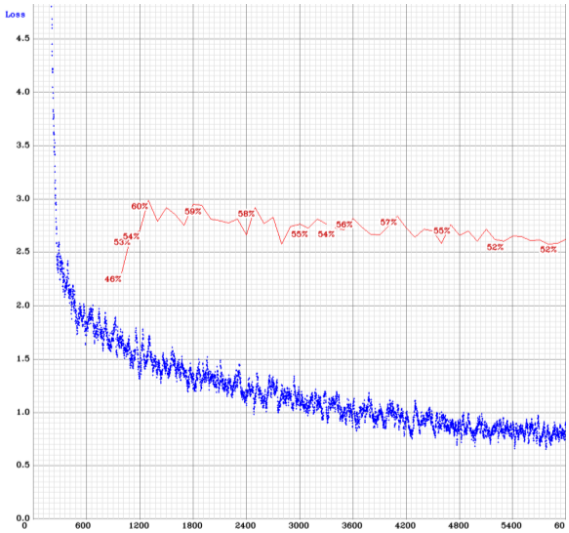

Fig. 8. Validation Score From Training Process Model 1

Figure 8. Shows the training process for model 1, the original model without using image preprocessing, this model produces the highest mean average precision of 59.76%.

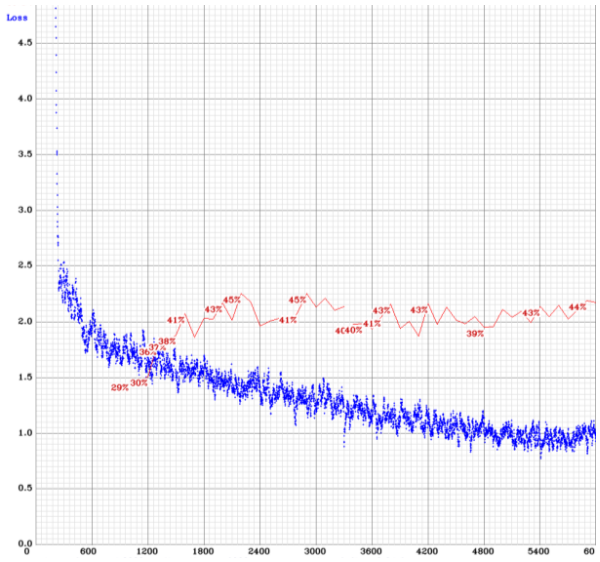

Fig. 9. Validation Score From Training Process Model 2

Figure 9. Shows the training process of model 2, the model that used image preprocessing, this model produces the highest mean average precision of 45.1%.

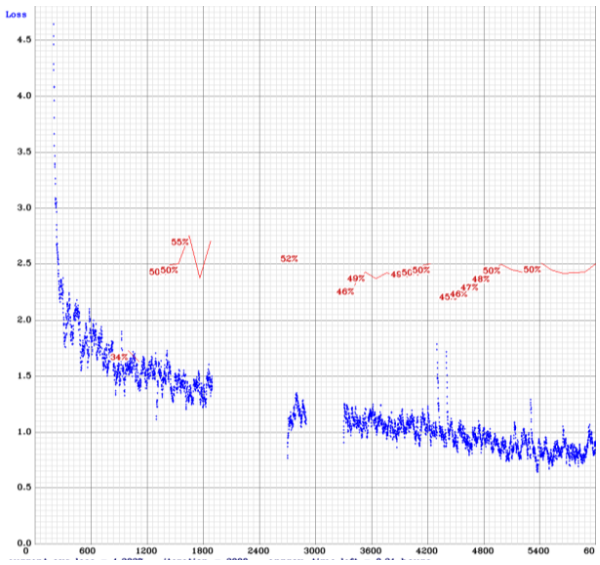

Fig. 10. Validation Score From Training Process Model 3

Figure 10. Shows the training process of model 3, the model that used an input layer resolution size of 608x608, this model produces the highest mean average precision of 55.01%. Figure 8 to Figure 10 show that the longer the training process is carried out, the smaller the loss generated by the model.

B. Mean Average Precision (mAP)

From the training process that had been carried out, this research produces the best model with a mean average precision of 59.76% on the validation set, Figure 11 shows the percentage comparison of mAP owned by each model that has been built.

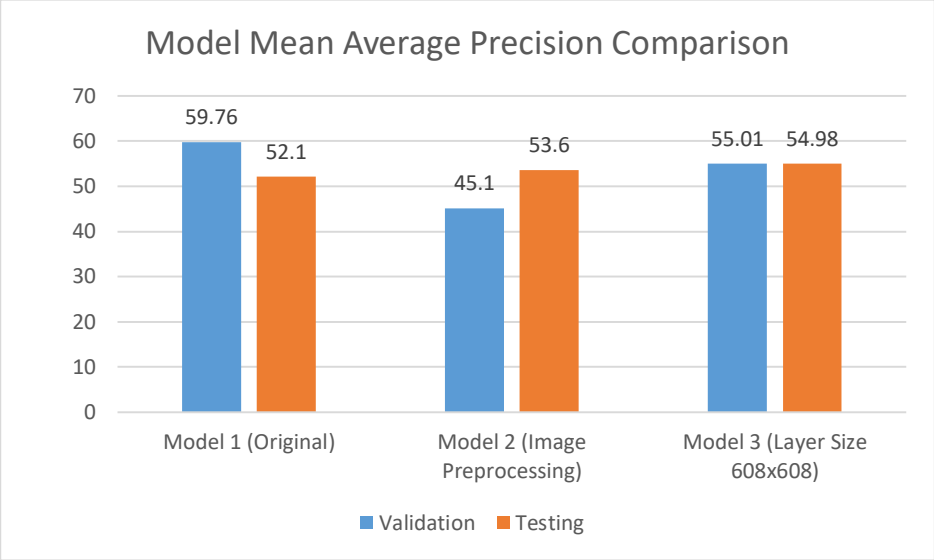

Fig. 11. Model Mean Average Precision Comparison

As shown in Figure 11, model 1 has the best mean average precision of 59.76% on a validation dataset but produces only 52.1% on the testing set, which indicates a mean average precision difference of 7.66%. While model 3 has a lower mean average precision than model 1 on a validation set with a 55.01% mean average precision, but this model only has a mean average precision difference of 0.03% on the testing set with a mean average precision of 54.98%, indicating that model 3 is more stable.

C. Model Comparison

All comparisons of results between the models we have built is shown in Table IV. The training time required for each model depends on its configuration. Model 3 required a longer training time than model 1 and model 2 because it used an input layer with resolution size of 608x608.

TABLE IV
OVERALL RESULT MODEL COMPARISON

Model	Model 1 (Original)	Model 2 (Image Processing)	Model 3 (Layer Size 608)
Training Time	10 Hour	10 Hour	30 Hour
Best Epoch Training	1370	2901	1649
Validation Set AP(mAP@50)	59.76	45.10	55.01
Testing Set AP (mAP@50)	52.1	53.6	54.98
TP	122	155	150
FP	72	104	59
FN	155	122	192
Precision	0.63	0.6	0.72
Recall	0.44	0.56	0.44
F1-Score	0.52	0.58	0.54

D. Video Testing

All model that have been built are tested on each video with a duration of five seconds to measure the performance of each model.

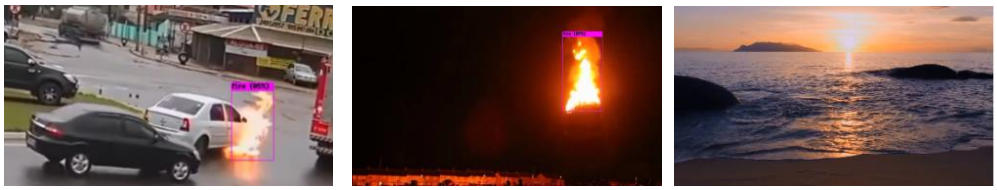

Fig. 12. Video Result of Model 1 (Original), Left (Video A), Middle (Video B), Right (Video C)

As shown in Figure 12, model 1 (original) detected fire well on light and dark backgrounds and it knows that there is no fire in video C with a detection accuracy of above 90%.

Fig. 13. Video Result of Model 2 (Image Processing)

in Figure 13, model 2 shows detection accuracy results above 90% as well but this model had an error in detecting video C because this model detected a fire that is not exist in video C.

Fig. 14. Video Result of Model 3 (Layer Size 608x608)

As shown in Figure 14, although model 3 knows in video C there is no fire, the performance of this model only has a detection accuracy above 70%.

Table V is a comparison of the average FPS performance of all videos from each model, model 1 has an average detection speed of 34.17 FPS, while model 2 is the model with the highest average FPS with an average detection speed of 35.1 FPS, this happened because model 2 is built used image preprocessing while model 1 is built without implemented the image preprocessing. The model 3 only has an average detection speed of 33.5 FPS because this model used an input layer with resolution size of 608x608 so it affected performance of the model by used more memory usage. Table V shows the models that have been built used different preprocessing and hyper-parameters that produce varied results.

TABLE V
VIDEO RESULT COMPARISON

Model	Model 1 (Original)	Model 2 (Image Processing)	Model 3 (Layer Size 608)
Average Fps Video A	37.7 FPS	36.2 FPS	34.6 FPS
Average Fps Video B	33.6 FPS	38.1 FPS	34.3 FPS
Average Fps Video C	31.2 FPS	31 FPS	31.6 FPS
Total Average Fps	34.17 FPS	35.1 FPS	33.5 FPS

V. CONCLUSION

We have conducted three different scenarios to build a fire detection model using CNN and Yolo version four with 2000 different images as dataset. As stated from the scenario result, model A gained the highest average accuracy above 90% with an average detection speed of 34.17 FPS. Based on our experiments it can be concluded in terms of performance and accuracy, fire detection models are not always improved by using image preprocessing but adding more dataset and using image augmentation might improve the model. Increasing the size of the input resolution in hyper-parameters not only increases the precision of the model but also improves the stability of the model when given a new dataset.

ACKNOWLEDGMENT

The author would like to thank all those who have provided support and the opportunity for us to complete this research.

References

[1] Chi, R., Lu, Z. M., & Ji, Q. G. (2017). Real-time multi-feature based Fire Flame Detection In Video. IET Image Processing, 11(1), 31–37. https://doi.org/10.1049/iet-ipr.2016.0193
[2] Qiu, T., Yan, Y., & Lu, G. (2012). An autoadaptive edge-detection algorithm for Flame and fire image processing. IEEE Transactions on Instrumentation and Measurement, 61(5), 1486–1493. https://doi.org/10.1109/tim.2011.2175833
[3] Che-Bin Liu, & Ahuja, N. (2004). Vision based fire detection. Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004. https://doi.org/10.1109/icpr.2004.1333722
[4] Celik, T., Demirel, H., Ozkaramanli, H., & Uyguroglu, M. (n.d.). Fire detection in video sequences using statistical color model. 2006 IEEE International Conference on Acoustics Speed and Signal Processing Proceedings. https://doi.org/10.1109/icassp.2006.1660317
[5] Ko, B. C., Ham, S. J., & Nam, J. Y. (2011). Modeling and formalization of fuzzy finite automata for detection of irregular fire flames. IEEE Transactions on Circuits and Systems for Video Technology, 21(12), 1903–1912. https://doi.org/10.1109/tcsvt.2011.2157190
[6] Kim, B., & Lee, J. (2019). A video-based fire detection using Deep Learning Models. Applied Sciences, 9(14), 2862. https://doi.org/10.3390/app9142862
[7] Shen, D., Chen, X., Nguyen, M., & Yan, W. Q. (2018). Flame detection using Deep Learning. 2018 4th International Conference on Control, Automation and Robotics (ICCAR). https://doi.org/10.1109/iccar.2018.8384711
[8] Wang, T., Shi, L., Yuan, P., Bu, L., & Hou, X. (2017). A new fire detection method based on flame color dispersion and similarity in consecutive frames. 2017 Chinese Automation Congress (CAC). https://doi.org/10.1109/cac.2017.8242754
[9] Dua, M., Kumar, M., Singh Charan, G., & Sagar Ravi, P. (2020). An improved approach for fire detection using Deep Learning Models. 2020 International Conference on Industry 4.0 Technology (I4Tech). https://doi.org/10.1109/i4tech48345.2020.9102697
[10] Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT Press.
[11] Li, P., & Zhao, W. (2020). Image fire detection algorithms based on Convolutional Neural Networks. Case Studies in Thermal Engineering, 19, 100625. https://doi.org/10.1016/j.csite.2020.100625
[12] Hidayatullah, P. (2021). Buku Sakti Deep Learning Computer Vision Menggunakan Yolo untuk Pemula. Stunning Vision AI Academy.
[13] Vasilev, I., Slater, D., Spacagna, G., Roelants, P., & Zocca, V. (2019). Python deep learning: Exploring deep learning techniques and neural network architectures with pytorch, Keras, and tensorflow. Packt Publishing.
[14] Cao, C., Tan, X., Huang, X., Zhang, Y., & Luo, Z. (2021). Study of flame detection based on improved Yolov4. Journal of Physics: Conference Series, 1952(2), 022016. https://doi.org/10.1088/1742-6596/1952/2/022016
[15] Mukhiddinov, M., Abdusalomov, A. B., & Cho, J. (2022). Automatic fire detection and notification system based on improved Yolov4 for the blind and visually impaired. Sensors, 22(9), 3307. https://doi.org/10.3390/s22093307
[16] Tajbakhsh, N., Shin, J. Y., Gurudu, S. R., Hurst, R. T., Kendall, C. B., Gotway, M. B., & Liang, J. (2016). Convolutional Neural Networks for medical image analysis: Full training or fine tuning? IEEE Transactions on Medical Imaging, 35(5), 1299–1312. https://doi.org/10.1109/tmi.2016.2535302
[17] N Prabhu Ram, R Gokul Kannan, V Gowdham, R Arul Vignesh. (2020). Fire Detection Using Cnn Approach. International Journal Of Scientific & Technology Research (IJSTR).