

## **FIRE EARLY WARNING SYSTEM VIA CCTV CAMERA USING CONVOLUTIONAL NEURAL NETWORK**

**Safri Adam<sup>1\*</sup>, Anggi Syahrul Kurniawan<sup>2</sup>, Nurul Fadillah<sup>3</sup>**

<sup>1,2,3</sup> Teknik Informatika, Jurusan Teknik Elektro Politeknik Negeri Pontianak  
emai: <sup>1</sup>safriadam@polnep.ac.id, <sup>2</sup>anggiekurniawan99@email.com,  
<sup>3</sup>nurulfadillah@polnep.ac.id

### **ABSTRAK:**

*At the Informatics Engineering Laboratory of Pontianak State Polytechnic, there is a fire detection system using smoke and heat sensors in certain rooms, while CCTV cameras are available throughout the room. Therefore, a fire detection system using CCTV cameras is proposed as an alternative fire detection tool to support sensor-based fire detection. Convolutional Neural Networks (CNNs) are a class of deep learning algorithms specifically designed for processing structured grid data, such as images. They are particularly effective in tasks related to image recognition and classification due to their ability to automatically learn spatial hierarchies of features from input images. This system works by processing video from CCTV cameras and classified using a Convolutional Neural Network (CNN) model that has been previously trained to recognize visual signs related to fire on video from CCTV cameras. Detected fire will be sent a notification via the user's Telegram application. These results show that the system works as expected with an average confidence level of 91.9% accuracy and 20.8% loss. The system was successfully developed into a fire detection application using the Convolutional Neural Network (CNN) model integrated with CCTV cameras and notification features via the Telegram application.*

**Keyword:** *Internet of Things (IoT), deep learning, Convolutional Neural Network (CNN), fire detection system, CCTV, Informatics Engineering Laboratory.*

### **I. INTRODUCTION**

Fire is a disaster caused by uncontrolled flames, and it is also one of the disasters that frequently occur around us, generally causing harm and being difficult to control (Ramadah, Wibawa, & Rizal, 2022). The causes of fires usually stem from human negligence, natural factors, and so on (Damkar Banda Aceh, 2020). According to data on the Grenfell Tower fire in London in 2017, the losses incurred from the fire included the deaths of 72 fatalities, dozens injured, hundreds displaced, making it the deadliest fire in Britain in the last century (Barak, 2020). In the State Polytechnic Pontianak (POLNEP) environment itself, a fire occurred in January 2022 in a machine workshop due to electrical short circuiting. There were no

causalties from the fire, only some workshop equipment damaged as a result (Admin, 2022). Rapid fire handling is crucial to extinguish it promptly. Anticipatory measures can mitigate the risk of significant losses of fire, such as installing fire alarm warning system or automatic fire extinguishing system with water (C Bhuvaneshwari, 2022).

Currently, fire anticipation technology involves fire detection alarms using smoke sensors (Hendri & Wahyuni, 2018) (Irsyad Dzikhrollah, 2023). Smoke sensor-based fire detection alarms are conventional fire detection systems where the system has to wait for smoke to reach the sensor before it can respond. However, this method is not very effective or efficient as it takes a considerable amount of time for smoke from a fire to reach the sensor. Smoke sensor-based systems cannot be used in open area due to their reliance on ceilings or walls. Additionally, in open areas, there are many combustible materials such as trees and fuel that can interact with the oxygen in the air, causing rapid fire growth and spread (Bayaoumi, et al., 2013) Thus, these sensor-based systems cannot function optimally in detecting fires. Moreover, the cost of implementing sensor-based fire detection systems is significant, and they can only be used in small areas. If a larger area requires fire detection, multiple systems need to be installed, which is not cost-effective in today's system context.

To address this issue, minimising the cost of fire detection can be achieved by utilising CCTV cameras with the assistance of the OpenCV library. This system utilises digital image processing on a computer. The system works by capturing video data using CCTV, which is then processed by the computer frame by frame. If the computer detects frames indicating potential fire, the system sends a notification through the Telegram application. This system is expected to complement the existing fire detection systems in the Polnep Informatics Engineering Laboratory, which uses smoke and heat sensors (Fitri Wibowo S. S., 2022) (Fitri Wibowo S. S., 2024). However, not all rooms are equipped with smoke and heat sensors. However, not all rooms are equipped with smoke and heat sensors due to cost limitations. Nonetheless, every room in the Informatics Engineering Laboratory at Polnep is equipped with CCTV to monitor conditions and activities. Thus, the CCTV-based fire detection system is expected to serve as an alternative fire detection system in the Polnep Informatics Engineering Laboratory.

## II. RESEARCH METHOD

There are several references that have been obtained, the first is written by G. sathyakala, V.Kirhika, and B. Aiswarya entitled "Computer Vision Based Fire Detection with a Video" in 2018. This journal discusses a computer vision-based fire detection system using CCTV cameras by breaking down the video frame by

frame for easy processing by the computer (Gunawaardena, Ruwanthika, & Jayasekara, 2016). In this journal, the authors adopt the method of breaking down the video frame by frame for processing using OpenCV.

The second reference is written by Sahar Bayoumi, Iham Al Sobky, Monerah Almohsin, Manahel Altwaim, Walikota Monira and Hadiyah Alkahtani entitled “A Real-time Fire Detection and Notification System Based on Computer Vision” in 2013 (Bayaoumi, et al., 2013). This journal discusses the development of a computer vision-based fire detection system that can provide real-time notifications via email if a fire occurs using smoke sensors and CCTV cameras. In this journal, the authors adopt a system that can provide real-time notifications.

The third reference is written by Muhammad Hendri and Elvira Sukma Wahyuni entitled “Design of Smoke and Fire Detection System Using Image Processing: in 2018 (Hendri & Wahyuni, 2018). In this journal, the design of a smoke and fire detection system using image processing with a CNN mode is discussed, which is capable of solving problems in object recognition and object detection with high accuracy and precision. In this journal, the authors adopt a CNN model to detect fires because this model has high accuracy and precision in detecting objects.

The fourth reference is written by Oxyta Sri Giyanto and Wahyat entitled “Prototype Fire Detection Device with Telegram Notification and IoT-based Alarm)” in 2022 (Giyanto & Wahyat, 2022). This journal discusses the design and creation of a device that can detect fire and smoke which can be applied indoors or outdoors. This system can be integrated online, making it easier for information to be conveyed quickly with an internet connection. The system has features that can send notifications to the fire detection bot on Telegram to mobile users. The system design to be developed is a combination of several features, methods, and models from the above references, namely a fire detection system through CCTV using the OpenCV library. This system is created using a CNN model to manage frames from the CCTV video output and generate output that will sent via notifications on the Telegram application, with implementation done in the Computer Engineering Laboratory at Polnep.

The research method to design and build a fire detection system through CCTV cameras using the OpenCV library consists of several stages as follows:

### **Data Collection**

Data collection is a stage to gather all the necessary information related to the system to be developed. In this stage, there are several methods of data

collection, such as downloading the required datasets. The datasets needed include images of fire and non-fire. The example of the required data can be seen in Figure 1.



(a) Fire (b) Non Fire

Figure 1. Dataset Sample

**Model Development**

In this stage, the model to be created is determined according to the needs. The model created must be able to classify images with and without fire. In the model development stage, a computer server or Google Colab can be used to process data and run CNN algorithms.

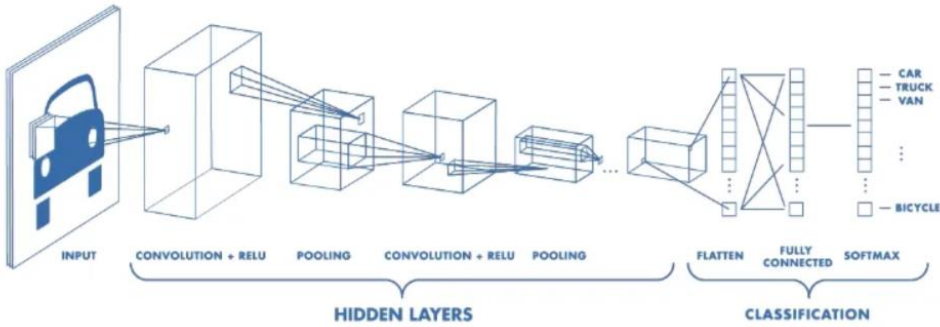


Figure 1. CNN Algorithms

The CNN algorithms has been widely used in image processing research. This method has a high level of accuracy in detecting objects and images classification. Therefore, in this study, an early fire detection mehid based on image using the CNN algorithm is developed (Hendri & Wahyuni, 2018).

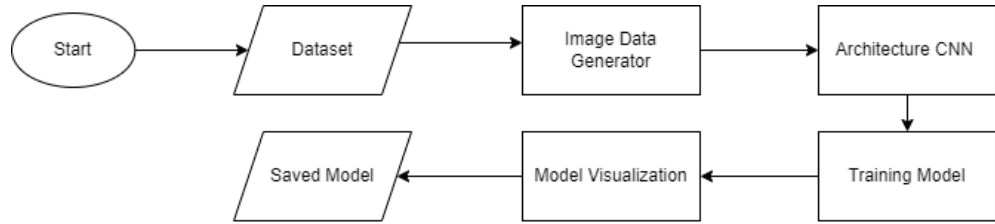


Figure 2. The Proses of Creating CNN

In Figure 3, there is a design for creating a CNN model for the development of a fire detection system. Here is an explanation of the processes involved in creating the model.

1) *Dataset Collection*

This step involves collecting data that will be used to train and test the CNN model. The dataset can consist of images or pictures to create object recognition patterns. The collected dataset consists of fire and non-fire images from Deep Quest AI, as well as some images taken as needed for training the model. Here are the details of data used for training and validation in the system model to be created.

Table 1. Dataset percentage

<i><b>Dataset</b></i>	<i><b>Fire</b></i>	<i><b>Non-Fire</b></i>
<i>Training</i>	694 (80.23%)	286 (80.79%)
<i>Validation</i>	171 (19.77%)	68(19.21%)
<b>Total</b>	<b>865</b>	<b>354</b>

In the dataset details, there are 865 fire images, which account for 80.23% for training the model and 19.97% for validating the model. Meanwhile, for the non-fire images, there are 354 images, consisting of 80.79% for training the model and 19.21% for validating the model.

2) *Image Data Generator*

The Image Data Generator is a class from the Keras API that is highly useful in image data processing. This generator willload images from the provided dataset and can apply various transformations to help enrich the dataset and prepare it for training. Some transformations applied to the training data include rescale (normalization), horizontal\_flip (horizontal flipping), rotation\_range (rotation), height\_shift\_range (random vertical shifting), and fill\_mode (method for filling empty pixels). Then, the training generator resizes the images to (224,224) with categorical type and divides the batch data as neede. Training data undergoo augmentation, while validation data only undergo normalisation.

3) *CNN Architecture:*

The construction of the CNN architecture adopts the research by Dhruvil Shah, as the study achieved a training accuracy of 96.83% and a validation accuracy of 94.98% (Shah, 2020). The training and validation losses were 0.09% and 0.13%, respectively, after training for 50 epochs. The CNN architecture consists of several convolutional layers, pooling layers, dropout layers, and fully connected layers for binary classifications of images. This model can be trained using preprocessed data with the Image Data Generator and evaluated based on predefined accuracy and loss metrics. A summary of the CNN architecture can be seen in Figure 4.

Layer (type)	Output Shape	Param #
conv2d_3 (Conv2D)	(None, 107, 107, 96)	34944
max_pooling2d_3 (MaxPooling2D)	(None, 53, 53, 96)	0
conv2d_4 (Conv2D)	(None, 49, 49, 256)	614656
max_pooling2d_4 (MaxPooling2D)	(None, 24, 24, 256)	0
conv2d_5 (Conv2D)	(None, 20, 20, 384)	2457984
max_pooling2d_5 (MaxPooling2D)	(None, 9, 9, 384)	0
flatten_1 (Flatten)	(None, 31104)	0
dropout_3 (Dropout)	(None, 31104)	0
dense_3 (Dense)	(None, 2048)	63703040
...		
Total params: 68,910,850		
Trainable params: 68,910,850		
Non-trainable params: 0		

Figure 3. the CNN Architecture Summary (Shah, 2020)

The displayed model architecture consists of various type of layers arranged sequentially to process images in a classification task. The model starts with a convolutional layer (conv2d\_3) applying 96 filters of size 11x11 on the input image, followed by a max-pooling layer (max\_pooling2d\_3) to reduce the image dimensions. Subsequently, the next convolutional layer (conv2d\_4) applies 256 filters of size 5x5, followed by another max-pooling layer (max\_pooling2d\_4).

Next, the final convolutional layer (conv2d\_5) is applied with 384 filters of size 5x5, followed by another max-pooling layer (max\_pooling2d\_5). The processed data is then flattened into a one-dimensional vector (flatten\_1) before passing through three dropout layers (dropout\_3) to prevent overfitting. Three fully connected layers (dense\_3, dense\_4, and dense\_5) successively have 2048, 1024, and 2 neurons, with ReLU activation on the first and second layers and softmax activation on the last layer.

Overall, this model has around 69,910,850 adjustable parameters with no untrainable parameters. Using the Adam optimizer, and ‘categorical\_crossentropy’ loss, this model is intended for image classification tasks with two classes and will be evaluated using accuracy during training.

4) *Model Training:*

The CNN model that has been constructed will be trained using previously prepared data. Training involves feeding data input through the model, calculating loss, and adjusting model parameters based on optimization algorithms such as SGD and Adam to minimize loss and enhance model quality. Model training serves to discover patterns within the data that will form the basis of the model's knowledge. This process utilizes TensorFlow and Keras to train the CNN model in fire detection.

The CNN Model Training adopts research by Dhruvil Sjai, as the study achieved a training accuracy of 96.83% and a validation accuracy of 94.98%. The training and validation losses were 0.09% and 0.13% respectively, after training for 50 epochs<sup>[9]</sup>. The CNN architecture consists of several convolutional layers, pooling layers, and fully connected layers for binary classification of images. This model can be trained using preprocessed data with the Image data Generator and evaluated based on predefined accuracy and loss metrics.

#### 5) *Model Visualization*

Model Visualisation is the process of visualizing various aspects of a machine learning or deep learning model, such as the model architecture, performance during training, and the model's response to input data. Its goal is to provide a better understanding of how the model operates, processes information, and makes predictions.

#### 6) *Testing the Model*

This stage involves testing the developed and trained model. Testing is done by inputting videos containing fire into the model to determine whether the system can detect or not.

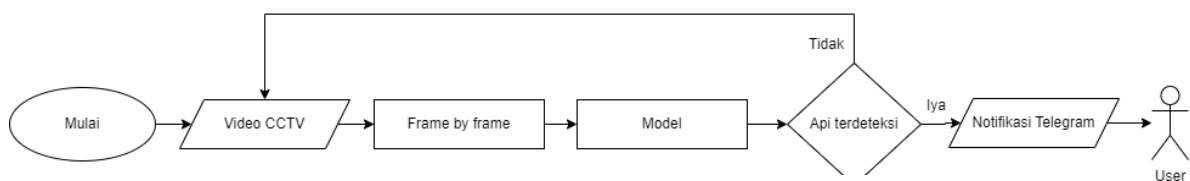


Figure 5. Testing the Model

In figure 5, there is a flowchart of the fire detection system for testing the model. The system will start by capturing real-time video output from CCTV cameras, then the video will be processed by breaking it down frame by frame. Subsequently, each frame will be processed by the model. If the model detects a fire frame among the processed frames, the system will send a notification via the Telegram application by sending potential fire frame that could cause a fire, which can be viewed on the user's device.

### III. RESULT AND DISCUSSION

## Results

After completing the system development, there are several outcomes that will be presented as follows:



Figure 6. Fire Detection Application Through CCTV

In Figure 6, there are the design and development results of the fire detection application through CCTV cameras. The image above shows the interface of the application. This application is created using the Python programming language, and its interface is built on the Flask Web framework. Flask is a web framework built using the Python programming language. To run the fire detection application through CCTV cameras, you can execute the application file created in an Integrated Development Environment (IDE) such as Visual Studio Code. To run the application interface, you can open a browser like Google Chrome and select the video to be processed by the application using the “Choose File” button.

“Choose File” is an element or button on a web page that allows users to select and upload files from their devices to the web server. Once the desired file has been selected, users can click the submit button to send file to the web server.





Figure 7. *Running the Application*

In Figure 7, the fire detection application through CCTV cameras has been executed. In this display, the application shows the video output from the CCTV, which is then processed by the fire detection application to provide predictions based on the processed video. If the fire detection application through CCTV cameras predicts the presence of a fire in the processed video, then the application will send a notification to the Telegram application, specifically to the Telegram Bot that has been created.

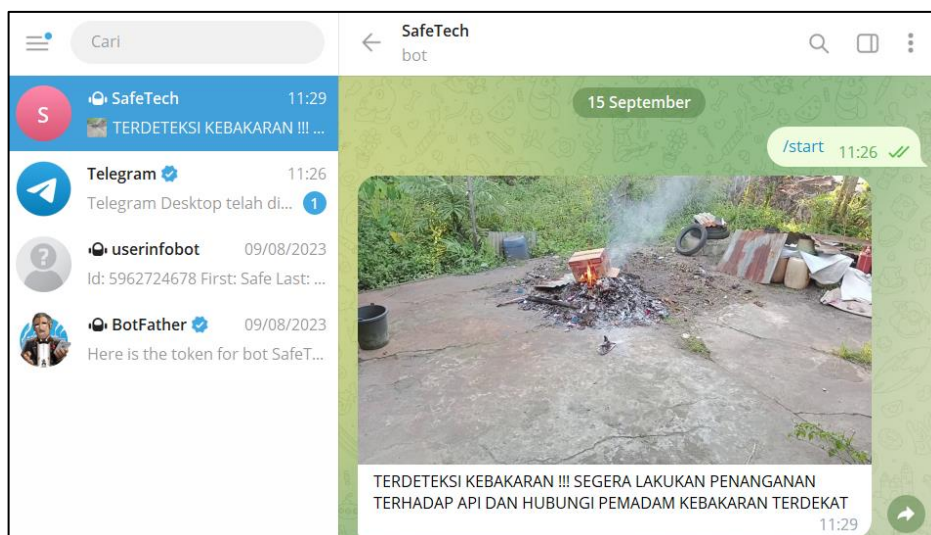


Figure 8. *Telegram Bot*

In Figure 8, it displays notifications on the Telegram application sent by the fire detection application through CCTV cameras. The notifications sent contain frames or images that include a fire within them. If the fire detection application predicts a fire while running, it will immediately send a notification to the Telegram application through the previously created Telegram Bot. The notification sent

includes an image containing the fire and a message to inform the user to quickly take precautionary or firefighting measures. By sending images directly when a fire occurs, users can promptly validate the fire to prevent false alarms. By sending these notifications, firefighting measures can be promptly implemented.

**Discussion**

At this stage, the results of designing and building a fire detection application through CCTV cameras using the OpenCV library are discussed. The process consists of several stages as follows:

**Model Creation**

In this stage, the developed model should be able to classify images of fire and non-fire. In creating the model, the author utilized the Integrated Development Environment (IDE), Visual Studio Code, running on a server computer and employing the CNN algorithms. Based on the training results, as the last epoch, namely epochs 50/50, it was found that the trained model met expectations with an average confidence level or accuracy at 91.9% and a loss of 20.8%. Training and validation visualizations can be seen in Figure 9.

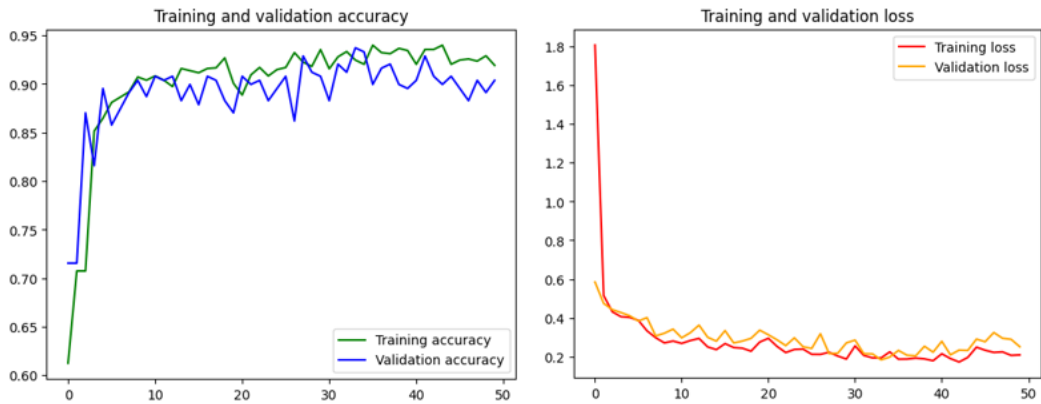






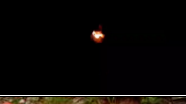


Figure 9. Data Training and Validation

During the training process, changes in loss and accuracy values are observed. Initially, the loss is generally high and the accuracy is low, but as the epochs progress, the model starts to understand patterns in the training data and successfully optimizes parameters to reduce loss and increase accuracy. This is the result of weight optimization within the model. By the end of the 50 epochs, the model has shown significant improvement. Loss has decreased and accuracy has increased, indicating that the model has successfully learned from the training data and can more accurately classify images in the validation data.

**Testing**

The test results were obtained from experiments conducted by the author. Testing was carried out by preparing several videos depicting fire and non-fire scenarios. The method chosen by the author is the Black Box testing method. This Black Box testing method assesses the functionality of the tested application. The generated output is then compared with the expected output. The next step is to input the results of the Black Box testing into the test table.

*Table 2. Testing result*

Testing Activities	Angle of Capture			The Expected Realisation	Testing Results	Confidence Score	Conclusiom	
	Distance	Camera Height	Lighting	Time needed/estimation			Suitable	Not Suitable
	1 Meter	1 Meter	23.601 lux	17 Seconds	Detecting a Fire	“Fire Detected”	76,7%	✓
	3 Meters	2,5 Meters	1.080 lux	50 Seconds	Detecting a Fire	“ Fire Detected ”	54,3%	✓
	0,5 Meter	0,3 Meter	44 lux	9 Seconds	Detecting a Fire	“ Fire Detected ”	80,9%	✓
	3,5 Meters	1,5 Meter	1.080 lux	108 Seconds	Detecting a Fire	“ Fire Detected ”	62%	✓
	3 Meters	2,5 Meter	0 lux	-	Detecting a Fire	“ Fire Not Detected ”	-	✓
	0,5 Meter	1 Meter	7.633 lux	-	Not Detecting a Fire	“ Fire Not Detected ”	-	✓
	10 Meters	1,5 Meters	43.698 lux	-	Not Detecting a Fire	“ Fire Not Detected ”	-	✓

Based on table 2 of the black box testing, the testing was conducted with several parameters such as distance, camera height, lighting, and time. To measure the level of lighting in the testing environment, a light sensor on a smartphone device was used with the assistance of the Illuminance application to measure light intensity (Ramadah, Wibawa, & Rizal, 2022). There are also expected realizations and testing results to compare the output of the testing with different inputs and confidence scores to measure how confident the model is in predicting fire and non-fire categories.

Based on the testing data in the testing environment, it can be concluded that distance, camera height, and lighting level are factors that greatly influence the time required for the application to detect fire images. The farther the distance from the camera to the fire object, the higher the camera position, and the lower the lighting level, the longer the image capture time will be. Differences in camera configurations can also affect the capture time. Additionally, different lighting levels at the same distance can result in differences in the time to detect fire images.

#### IV. CONCLUSIONS

Based on the development and testing conducted in the Fire detection Application through CCTV Cameras, the following conclusions were drawn:

- 1) The constructed model is quite effective in detecting fire. Based on the training results, it was found that the system operates as expected with an average confidence level of accuracy at 91.9% and a loss of 20.8%.
- 2) The application can classify fire because it has been trained using image data containing fire images. Classification is based on the visual patterns present in these images.
- 3) The time required by the application to detect fire images is influenced by several factors such as distance, camera height, and lighting level. The greater the distance, the higher the camera, and the lower the lighting, the longer the required time. Additionally, different camera configurations can also affect detection time. Different lighting settings at the same time distance can also result in different detection times.
- 4) The video output from should not exceed the set resolution, which is 1920x1080 pixels. This resolution limit is applied to ensure smooth application operation, as excessively high resolutions may require greater computational resources.
- 5) The application requires a stable internet connection as it sends data such as video from CCTV and notifications to Telegram. A stable connection is necessary for consistent data transmission and timely notifications.
- 6) The application is capable of effectively detecting fire when the color of the fire in the image resembles actual fire colors such as red, orange, and yellow.

The expected recommendations to aid the development of the fire detection system through CCTV cameras to make it better are as follows:

- 1) Enhance the model development to improve detection accuracy. By adding more training data and considering more complex model architectures, the resulting model will be much better.
- 2) Optimise application performance by enhancing the capacity and specifications of hardware and software to run the application. Although the application can run on the current devices for future and long-term implementations.
- 3) Refine notifications by sending notifications not only through the Telegram application but also through other platforms such as WhatsApp, which are more frequently and widely used in daily life.

Integrate with other security systems existing in the Polnep Infomatics Engineering Laboratory, such as smoke and heat-based fire detection systems, sound alarms, or automatic firefighting equipment, to provide a more comprehensive responses to fire threats.

## v. REFERENCES

Admin. (2022, January 4). (*BERITA VIDEO*) *Ruang Mesin Polnep Pontianak Terbakar*. (ninemedia) Dipetik May 18, 2023, dari <https://www.ninemedia.id/2022/01/berita-video-ruang-mesin-polnep.html>

Barak, H. (2020, Juni 14). *Api Melalap Apartemen Grenfell London, 72 Orang Tewas*. (LIPUTAN 6) Dipetik March 15, 2023, dari <https://www.liputan6.com/global/read/4277948/14-6-2017-api-melalap-apartemen-grenfell-london-72-orang-tewas>

Bayaoumi, S., Alsoubky, E., Almohsin, M., Altwaim, M., Alkaldi, M., & Alkahtani, M. (2013). A Real-time Fire Detection and Notification System Based on Computer Vision. *Fire Detection*.

C Bhuvaneswari, M. K. (2022). Implementation of Intelligent Residential Fire Extinguisher System. *2022 4th International Conference on Smart Systems and Inventive Technology (ICSSIT)*. Tirunelveli, India.

*Damkar Banda Aceh*. (2020, July 13). (Pemerintah Kota Banda Aceh) Dipetik March 15, 2023, dari <https://damkar.bandaacehkota.go.id/2020/07/13/faktor-penyebab-kebakaran-dan-upaya-pencegahan-kebakaran/>

Fitri Wibowo, S. S. (2022). Design and Implementation of IoT-Based Smart Laboratory Using ESP32 and Thingsboard to Improve Security and Safety in POLNEP Informatics Engineering Laboratory. *Jurnal ELIT*, 3(2), 13-21.

Fitri Wibowo, S. S. (2024). An IoT-Enabled Smart Energy Management System to Improve Energy Efficiency in University Laboratory. *Sinkron: jurnal dan penelitian teknik informatika*, 8(2), 1038-1046.

Giyanto, O. S., & Wahyat. (2022). Prototype Alat Pendeteksi Kebakaran Dengan Notifikasi Telegram. *Seminar Nasional Industri dan Teknologi (SNIT)*, 260-29.

Gunawaardena, A., Ruwanthika, R., & Jayasekara, A. (2016). Computer Vision Based Fire Alarming System . 1.

- Hendri, M., & Wahyuni, E. S. (2018). PERANCANGAN SISTEM DETEKSI ASAP DAN API MENGGUNAKAN PEMROSESAN CITRA.
- Irsyad Dzikhruallah, Z. B. (2023). Rangkaian Rancang Bangun Alat Pendeteksi Kebakaran berdasarkan Asap dan Suhu pada Dapur Restoran Berbasis Arduino dan Internet of Things. *Jurnal Teknologi Informasi dan Komunikasi*, 7(3), 465-471.
- Ramadah, F., Wibawa, P. D., & Rizal, A. (2022). Fire Detection System Using Image Processing on Webcam with Yolov3 Method. *e-Proceeding of Engineering*, 9, 228.
- Shah, D. (2020, July 6). *Early Fire detection system using deep learning and OpenCV*. (towardsdatascience) Dipetik April 17, 2023, dari <https://towardsdatascience.com/early-fire-detection-system-using-deep-learning-and-opencv-6cb60260d54a>