Image Classification for Smoke and Flame Recognition Using CNN and Transfer Learning on Edge Device

Endah Kristiani[1,2], Yi-Chun Chen[1,4], Chao-Tung Yang[1,3(✉)], and Chia-Hsin Li[4]

[1] Department of Computer Science, Tunghai University, Taichung 407224, Taiwan, Republic of China
ctyang@thu.edu.tw
[2] Department of Informatics, Krida Wacana Christian Univeristy, Jakarta 11470, Indonesia
[3] Research Center for Smart Sustainable Circular Economy, Tunghai University, No. 1727, Sec.4, Taiwan Boulevard, Taichung 407224, Taiwan, Republic of China
[4] iAMBITION TECHNOLOGY CO., LTD., 3F., No. 159-1, Sec. 1, Zhongxing Rd., Dali, Taichung 41267, Taiwan, Republic of China

Abstract. This paper implemented image classification for smoke and flame detection. CNN model was trained in three topologies of InceptionV3, MobileNet, and VGG16. These three models were then tested on Raspberry Pi 4 with Intel Neural Compute Stick 2 (NCS 2). The experimental results demonstrated that MobileNetV2 is a superior model to the other two models in terms of training and inference, even if the accuracy rate of the three was as high as 94% when utilizing the test set for evaluation.

Keywords: image classification · CNN · Imagnet · edge computing · transfer learning

1 Introduction

In recent years, there has been a significant increase in the development of computer image vision. In addition, image recognition has significantly improved the widespread adoption of convolutional neural networks. Neural Networks can be broken down into subcategories, one of which is called Convolutional Neural Networks (commonly abbreviated as CNN or ConvNet). These networks are typically implemented for tasks involving image and speech recognition. The high dimensionality of images can be reduced by its built-in convolutional layer without any information being lost in the process. That is the reason why CNNs are particularly well-suited for image classification problems [1–3].

This study uses smoke and flame images to create deep-learning models for detecting the early signs of fire. We trained the models based on CNN and

D.-J. Deng et al. (Eds.): SGIoT 2022, LNICST 497, pp. 150–158, 2023.
https://doi.org/10.1007/978-3-031-31275-5_16

inferred the models on the edge device. In detail, our objectives are listed as follows.

- To train the models based on three types of CNN networks, InceptionV3, MobileNet, and VGG16.
- To infer the trained models on Raspberry Pi 4 and Intel NCS 2.

2 Background Review and Related Works

This section discussed the methods and previous research works as our references, such as image classification and CNN.

2.1 Image Classification

Image Classification (often referred to as Image Recognition) is the process of connecting a given image with a single (single-label classification) or several (multilabel classification) labels. Classification of images is a challenging task that allows for the evaluation of modern architectures and methodologies in the field of computer vision. The most often encountered classification job in supervised image classification is single-label classification. As the name implies, single-label categorization assigns a single label or comment to each picture. As a result, the model generates a single value or prediction for each image seen. The model returns a vector of length equal to the number of classes and a value indicating whether the picture corresponds to that class.

2.2 Convolution Neural Network (CNN)

CNN has always been a significant part of Deep Learning. Its power in image and image recognition is mighty, and it can even be said to be even more potent than human eye recognition. Many image recognition models are also formed from the extension of the CNN architecture. A convolutional neural network uses convolution to use color and size in the image as the input data of the neural network. Compared with the most active neural network, the most significant difference is the sharing of weights. The basic idea of CNN is to use a diversified image database as training data. Then, transfer the image to the output end of the grid using neural network parameters, and calculate the error value of the target and prediction at the output end learned by backpropagation. CNN method constantly updates the weight value of the neural network because this calculation allows the convolutional neural network to solve the problem of large amounts of data. Therefore, convolutional neural network research for high variability and dimensions is very suitable for image recognition. Convolutional neural network architecture often includes single or multiple convolution layers, pooling layers, and a fully connected layer at the output end. The function of the convolutional layer is to capture images. The action of taking features, finding the best features, and then classifying, and the pooling layer is often added between the convolutional layers.

2.3 Smart Edge

Edge Computing is a distributed computing architecture. The data originally processed by the central node is cut and distributed to the edge nodes, which can be closer to the user's terminal equipment and speed up the data. The processing and speed of the network reduce the delay [4]. Edge computing decomposes the work that was originally unified from the center of the network, which is the cloud, and distributes it to the edge nodes for processing. This makes the processing and transmission of data faster and reduces the network transmission The resulting delay. Therefore, when the network connection is interrupted, part of the calculations or services handed over to the edge processing can continue to be completed, achieving high availability of the overall service. In previous applications where AI and terminals were combined, terminal devices' storage and computing capabilities were very limited, so data must be sent back to the cloud for processing through the network. This data will be sent back to the central data center if traditional cloud computing is used. Centralized processing, and then transmitted back to the user's equipment, this method has greatly increased the demand for network bandwidth. However, with the continuous advancement of hardware equipment and algorithms in recent years, edge computing is the key to accelerating the Internet of Things. After the edge node receives the data, it directly analyzes and processes it, thereby reducing cloud computing resources and improving transmission efficiency [5].

2.4 Related Works

Due to the limitations of mobile devices, Jadon et al. [8] proposed a portable method for fire identification and use. Most automatic fire alarm systems will detect fires caused by sensors such as heat, smoke, or flame. One of the new ways to solve this problem is to perform detection using images. The imaging method is promising because it does not require a specific sensor and can be easily embedded in different devices.

In the paper by Jareerat et al. [6], an early warning must be given to reduce the loss of life and property caused by fire. A fire detection system based on light detection and analysis is proposed. The system uses HSV and YCbCr color models under given conditions to separate orange, yellow, and high-brightness light from the background and ambient light. Analyze and calculate fire growth based on frame differences. The overall accuracy of the experiment has exceeded 90%.

Thou-Ho Chen [7] presents a smoke-detection method for early fire-alarming systems based on video processing. The basic strategy of smoke-pixel judgment is composed of two decision rules: a chromaticity-based static decision rule and a diffusion-based dynamic characteristic decision rule. The chromatic decision rule is deduced by the grayish color of the smoke and the dynamic decision rule is dependent on the spreading attributes of smoke.

3 Methods

3.1 Dataset

The research dataset for this study was compiled using web crawlers from daily fire scenes and aerial photos of big forest fires. It will be kept on the server for administration and data enhancement to support later trials. There are 1056 smoke photographs and 2305 fire images in the original collection of the fire dataset. The initial inadequate dataset is supplemented with Keras. The original image has been rotated, turned upside down, and moved to the left and right. Based on the augmentation, 8000 additional photographs were included. Therefore, the dataset increased to 11361 images. Then, for training purposes, the dataset has a distribution ratio of 60% train set, 20% validation set, and 20% test set.

3.2 Model's Training and Inference

After the image preprocessing, we began the construction of the Deep Learning model, using three different CNN networks to train fire image data, that is VGG16, InceptionV3, and MobileNetV3. Tensorflow was used as the basic framework to train the models. Imagenet was used as the transfer learning model in the training. Then, the OpenVINO tool was used for model optimization to speed up operational efficiency in the edge device. Figure 1 shows the Raspberry Pi with Intel NCS 2 as edge devices.

Fig. 1. Raspberry Pi and NCS

3.3 Optimization Development

The Raspberry Pi is being used as an edge device with the optimization development methodology. To accelerate the training and inference procedures, our system made advantage of open-source AI software. The Intel Distribution of the OpenVINO Toolkit optimizes model deployment for inference processing by adapting and optimizing trained models for the downstream hardware target. It supports models trained in TensorFlow, Caffe, and MXNet on the CPU, integrated GPU, VPU (Movidius Myriad 2/Neural Compute Stick), and FPGA.

The system used the TensorFlow framework to improve deep learning modules. The Intel Python distribution and the Data Analytics Acceleration Library are required for machine learning. Deep neural network (DNN) libraries that are open-source offer CPU-optimized functions.

4 Results

4.1 Fire Scene Classification Result

This work selects three algorithms of different depths for transfer learning training, namely Inceptionv3, VGG16, and MobileNetV2. Before training, we used K-fold cross-validation for the collection of datasets. The method is to randomly divide the data into k sets and then use one set as Testing data. The remaining k-1 sets as Training data, repeated until each set is regarded as Testing data, and the final results (Prediction results) are compared with the ground truth (Performance Comparison). Training on scenes of fire and normal conditions. After 100 training cycles, we observed the training curve through the training log until the loss value reached the minimum value and did not change. The VGG16 training curve is shown in Fig. 2 and Table 1, Inceptionv3 training curve is shown in Fig. 3 and Table 2, MobileNetV2 training curve such as Fig. 4 and Table 3.

1. VGG16 Models

Table 1. VGG16 Accuracy and Loss

	Accuracy	Loss
Train	0.9546	0.1284
Validation	0.9234	0.1991

2. InceptionV3 Models

Table 2. InceptionV3 Accuracy and Loss

	Accuracy	Loss
Train	0.9844	0.0412
Validation	0.9820	0.0438

3.MobileNetV2 Models

Three different image classification models through the test set of experimental results, The accuracy of VGG16 is as high as 94%, and the accuracy of

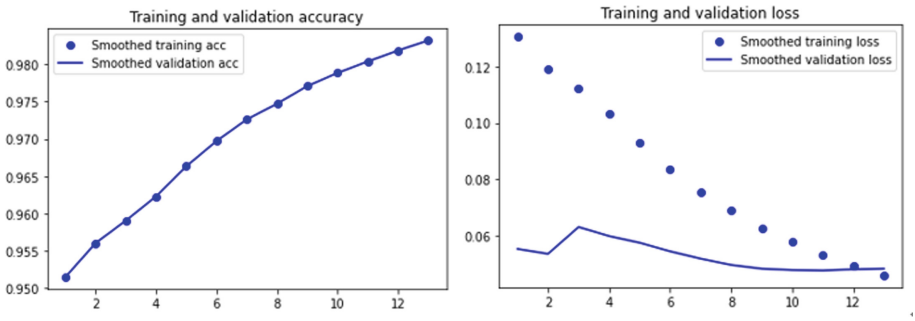

Fig. 2. VGG16 Acc and Loss

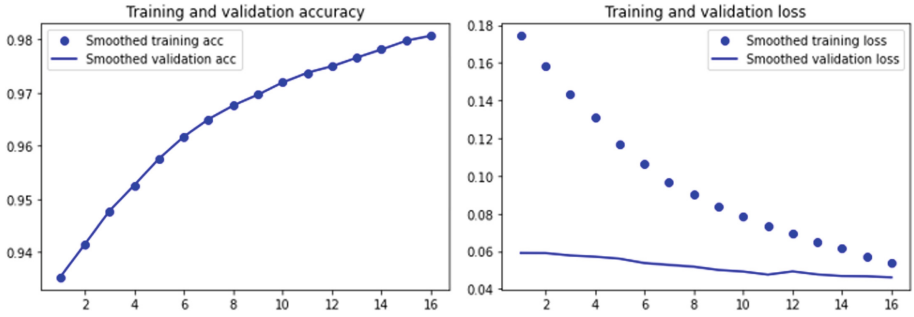

Fig. 3. InceptionV3 Acc and Loss

Table 3. MobileNetV2 Accuracy and Loss

	Accuracy	Loss
Train	0.9880	0.0396
Validation	0.9813	0.0417

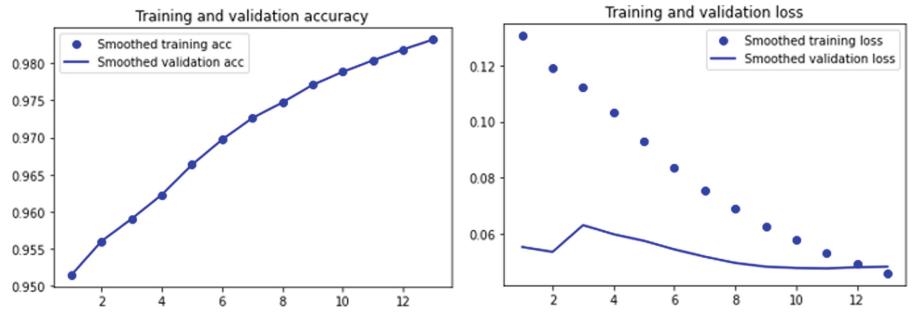

Fig. 4. MobileNetV2 Acc and Loss

Table 4. Compare the accuracy of the test set between different models

	True	False	Total	Accuracy
VGG16	685	37	722	0.9487
InceptionV3	713	9	722	0.9875
MobileNetV2	712	10	722	0.9889

InceptionV3 and MobileNetV2 is as high as 98%, such as Table 4 In addition to comparing the accuracy of the model on the test set, related evaluation indicators are also calculated to verify which model actually performs best, as shown in Fig. 5 and Table 5.

Table 5. Compare evaluation metrics between different models

	Recall	Precision	F1-score
VGG16	0.95	0.96	0.94
InceptionV3	0.99	0.98	0.99
MobileNetV2	0.99	0.99	0.99

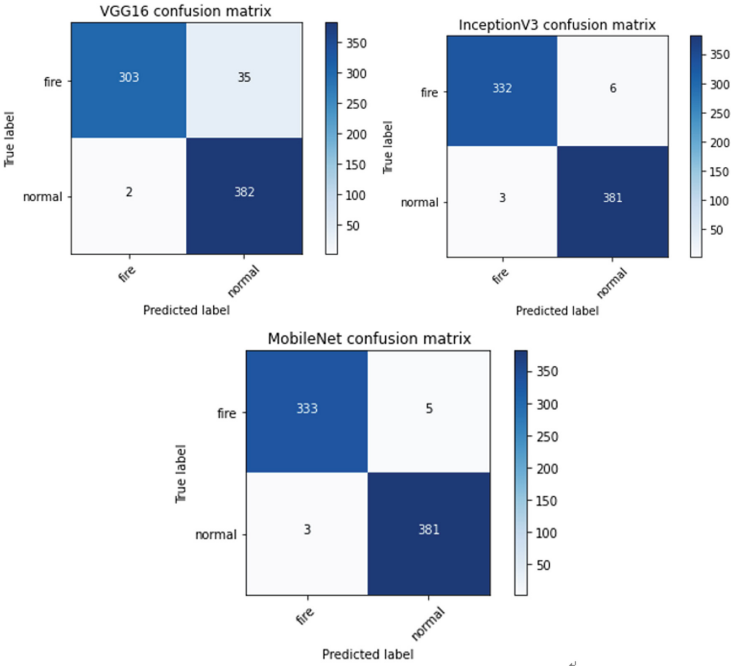

Fig. 5. Models confusion matrix

In addition to the comparison accuracy rate, FP16 (half-precision) and FP32 (single-precision) of the model were also compared. It can be seen from Table 6, the performance of VGG16 is far inferior to the other two models whether it is FP16 or FP32.

Table 6. Compare the operating efficiency between different models

Model Type	FPS
VGG16 FP32	5
VGG16 FP16	5
InceptionV3 FP32	19
InceptionV3 FP16	18
MobileNetV2 FP32	23
MobileNetV2 FP16	25

4.2 Classification Model Comparison

This paper uses three algorithms of CNN, VGG16, InceptionV3, and MobileNetV2. After training the models, we compared these three models' accuracy and related evaluation indicators. Among the three models, MobileNetV2 has the best accuracy and operating efficiency, with an accuracy rate of 98% and an FPS of 25 INF when using the FP16 model. Therefore, this paper chooses to use this model as the final model for subsequent fire scene classification in fire scene classification. The test results are shown in Fig. 6.

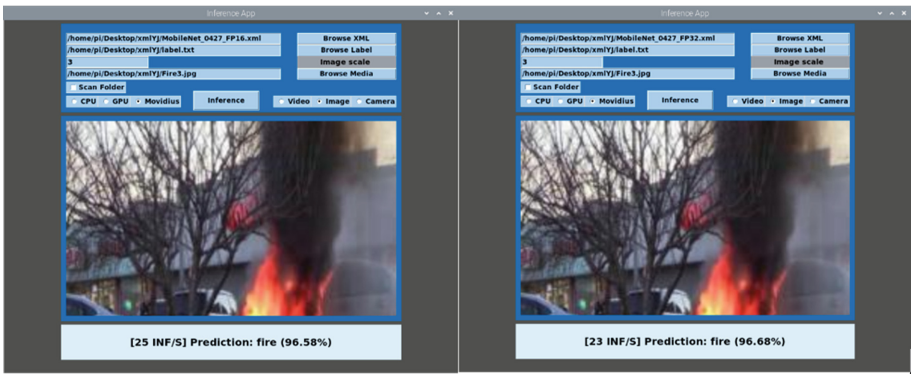

Fig. 6. MobileNetV2 FP16 and FP32 test

5 Conclusions

This paper implemented image classification for smoke and flame recognition. The fire detections employed an image classification method using the CNN model with three networks of VGG16, InceptionV3, and MobileNetV2. Although the accuracy rate of the three was as high as 94%, it can be seen that when using the test set for evaluation, the experimental data showed that MobileNetV2 is an excellent model compared to the other two models in terms of training and inference.

Acknowledgements. This work was supported by the National Science and Technology Council (NSTC), Taiwan (R.O.C.), under grants number 111-2622-E-029-003-, 111-2811-E-029-001-, 111-2621-M-029-004-, and 110-2221-E-029-020-MY3.

References

1. Gaur, A., Singh, A., Kumar, A., Kumar, A., Kapoor, K.: Video flame and smoke based fire detection algorithms: a literature review. Fire Tech. **56**(5), 1943–1980 (2020)
2. Li, P., Zhao, W.: Image fire detection algorithms based on convolutional neural networks. Case Stud. Thermal Eng. **19**, 100625 (2020)
3. Shi, F., et al.: A fire monitoring and alarm system based on YOLOv3 with OHEM. In: 2020 39th Chinese Control Conference (CCC). IEEE (2020)
4. Tang, J., Liu, S., Liu, L., Yu, B., Shi, W.: LoPECS: a low power edge computing system for real-time autonomous driving services. IEEE Access **8**, 30467–30479 (2020)
5. Huynh, L.N., Lee, Y., Balan, R.K.: DeepMon: mobile GPU based deep learning framework for continuous vision applications. In: MobiSys 2017 - Proceedings of 15th Annual Internationsl Conference on Mobile System, Applications and Services, pp. 82–95 (2017)
6. Seebamrungsat, J., Praising, S., Riyamongkol, P.: Fire detection in the buildings using image processing. In: 2014 Third ICT International Student Project Conference (ICT-ISPC), pp. 95–98 (2014). https://doi.org/10.1109/ICT-ISPC.2014.6923226
7. Chen, T., Yin, Y., Huang, S., Ye, Y.: The smoke detection for early fire-alarming system base on video processing. In: 2006 International Conference on Intelligent Information Hiding and Multimedia, pp. 427-430 (2006). https://doi.org/10.1109/IIH-MSP.2006.265033
8. Jadon, A., Varshney, A., Ansari, M.S.: Low-complexity highperformance deep learningdeep learning model for real-time low-cost embedded fire detection systems. Procedia Comput. Sci. **171**(2019), 418–426 (2020)