Comparison and Analysis of Three MobileNet-Based Models for Wildfire Detection

Shiyan Du [1,*], Jiacheng Li [2], and Masato Noto [2]

[1] Field of Electrical, Electronics and Information Engineering, Graduate School of Engineering, Kanagawa University, Yokohama, Japan

[2] Department of Applied Systems and Mathematics, Kanagawa University, Yokohama, Japan

Email: r202370157bj@jindai.jp (S.D.); lijiacheng@kanagawa-u.ac.jp (J.L.); noto@kanagawa-u.ac.jp (M.N.)

*Corresponding author

***Abstract*—The dynamic equilibrium of ecosystems can be maintained through controlled burning, but excessive wildfires can lead to severe consequences. Therefore, the use of Internet of Things (IoT) devices equipped with deep image processing models for wildfire detection has recently become a trend. Conventional deep image processing models suffer from accuracy issues and large model sizes, limiting their applicability on small IoT devices. To address this challenge, we utilized lightweight deep image processing models such as the MobileNet series to train a wildfire database. Furthermore, we evaluated three different versions of MobileNet (V2, V3 Large, and V3 Small) using a cross-entropy loss function to compare their accuracy and training times. Through data analysis, recommendations for deploying MobileNet models on IoT devices are provided. The results indicate that the ranking of MobileNet's accuracy from highest to lowest is V2, V3 Large, and V3 Small; the ranking of loss values from lowest to highest is V2, V3 Large, and V3 Small; and the ranking of training times from fastest to slowest is V3 Large, V2, and V3 Small.**

***Keywords*—wildfire detection, MobileNet model, deep learning, comparison and analysis**

I. Introduction

In the natural forest ecosystem, wildfires are a major disrupting factor. Wildfires not only cause significant damage to ecosystems but also have a profound impact on human societies. With the influence of climate change, the trends, frequency, and severity of wildfires in many regions are astonishingly increasing. However, controlling wildfires comes with high costs, and firefighting itself is a risky task. Additionally, wildfires generate various airborne particles, affecting human health and contributing to atmospheric pollution. Tran *et al.* [1] described in their paper the airborne particles produced by recent wildfires on the west coast of the United States. These particles can remain suspended in the atmosphere for days to weeks, passing through Heating, Ventilation, and Air Conditioning (HVAC) systems of various buildings before settling on equipment and facilities inside buildings. As a result, wildfires pose a threat not only to human health but also lead to a decline in air quality. To address the issues of wildfire occurrence and prevention, researchers have proposed various methods for predicting and detecting wildfires. Xie *et al.* [2] mentioned that the Fuel Moisture Content (FMC) method used in assessing wildfire risk can greatly help understand fire behavior and conduct firefighting management activities by estimating the range of FMC. This method achieves real-time detection of FMC through satellite remote sensing, providing an effective means for early wildfire warning. Shah *et al.* [3] used CubeSats satellites with a modular and cost-effective architecture in Low Earth Orbit (LEO), achieved early detection of wildfires within 30 min using multispectral visible to infrared cameras. Shaik *et al.* [4] classified fuel types using high-resolution images from PRecursore IperSpettrale della Missione Applicativa (PRISMA), demonstrating the effectiveness of the Scott/Burgan fuel model system through ground assessments, satellite measurements, and simulated concentration displays of summer data. Remote sensing technology is a widely used and effective approach. While satellite remote sensing has demonstrated considerable efficacy in wildfire prevention and control, its applicability is limited across all countries due to high technical barriers and associated costs. Consequently, researchers are actively exploring alternative methods for wildfire detection and prediction.

Ferreyra *et al.* [5] established a model predicting the emission of pollutants into the atmosphere from wildfires through simulating chemical and transport processes, aiding in the assessment and detection of wildfires. The research by Kotroni *et al.* [6] describes the DISARM project (Drought and fIre ObServatory and eArly waRning system), an atmospheric-fire model system that demonstrates strong support in suppressing wildfires, providing a crucial background for assessing wildfire risks in future climates. Zhang *et al.* [7] proposed a wildfire risk model that optimizes monitoring tower systems through integrating visual analysis, location allocation, and multiple coverage of high-risk wildfire areas, enhancing the effectiveness in multiple coverage of high-risk wildfire areas.

With the development of deep learning technology, new object recognition and image processing techniques

Manuscript November 20, 2023; revised December 13, 2023; accepted January 17, 2024; published April 16, 2024.

continue to emerge. Kashika and Venkatapur [8] adopted the DarkNet-53 (a 53-layer Deep Convolutional Neural Network) method and integrated it into a safety helmet detection system based on You Only Look Once v3 (YOLOv3), achieving better real-time detection of safety helmets. Xie *et al.* [9] proposed an image processing method based on three-frame difference, particle swarm optimization, and Otsu enhancement for detecting planktonic organisms. Experimental results showed that this method could rapidly and accurately acquire clear planktonic organism targets. Ye *et al.* [10] introduced a novel All-in-One Dehazing (AOD) Network based on Convolutional Neural Network (CNN) and Long Short Term Memory (LSTM) networks, demonstrating advantages in accuracy and efficiency for AOD tasks and providing good recommendations for robot action training.

In addition, IoT devices and temperature sensors play a crucial role in wildfire monitoring. These smart tools can track environmental temperature changes in real-time and identify potential signs of wildfires. Meanwhile, drones equipped with high-resolution cameras and infrared imaging technology have become crucial in wildfire detection. They can rapidly cover vast areas, providing real-time data on potential fire sources and significantly enhancing the capabilities of emergency response and wildfire management. In the context of drone-based forest fire detection, Xie and Huang [11] introduced an approach using a modified Faster Regions with Convolutional Neural Network (RCNN) algorithm based on transfer learning, improving performance by more than 18% compared to traditional methods. This technological advancement enhances monitoring and control of wildfires, strengthening our ability to address wildfire risks and effectively support prevention and management.

In summary, as technology evolves, the ways of detecting and preventing wildfires have become increasingly diverse. When choosing a detection method, it is crucial to select the most effective one as based on the current situation. For wildfires, the current use of Internet of Things (IoT) or drone devices for real-time detection represents a cost-effective and balanced approach. Therefore, the use of devices equipped with lightweight deep image processing models for wildfire detection is receiving increased research attention.

The main contribution of this paper can be given as the utilization of three versions of the lightweight deep image processing model MobileNet (V2, V3 Large, and V3 Small) for training the wildfire database. Through a comprehensive evaluation, we analyzed the results, encompassing accuracy and loss values on both the training and testing datasets. Additionally, we conducted a comparative examination of the training times for these three models. Based on these results, we present recommendations for deploying them on IoT or drone devices that require robust deep image processing capabilities.

The rest of the paper is structured as follows: Section II briefly summarizes the related work and provides a brief introduction to the MobileNet model. Section III provides details about the computer environment used for training the model and the methodology employed for evaluating the model in this study. In Section IV, we present the experimental results and analysis. Finally, Section V contains the conclusions and recommendations.

II. Related Works

In the rapidly advancing field of deep image processing technology today, employing deep image processing models to identify wildfire events has become a significant trend. Wu and Zhang [12] utilized classical object detection methods, including Faster RCNN, various variants of YOLO, and Single Shot MultiBox Detector (SSD), for forest fire detection. They introduced new categories for smoke and changes in fire areas to reduce false alarms, enhancing the accuracy of fire detection by modifying the YOLO structure. Bai and Wang [13] proposed a precise method for forest fire recognition, combining classification and anomaly detection models. They optimized an algorithm based on the Visual Geometry Group (VGG) network, trained smoke and flame recognition models through transfer learning, and improved early flame detection with an optimized YOLO network. This method achieved an average precision (mAP) of 96.5%, meeting the requirements for real-time wildfire detection.

However, traditional deep learning models, including VGG, typically face challenges of significant model size when deployed on resource-constrained portable mobile devices, especially in fields requiring real-time applications such as wildfire recognition. To address this challenge, with the continuous progress in the field of computer vision, lightweight deep image processing models, such as the MobileNet series, have emerged. These models are specifically designed to operate efficiently on mobile devices by reducing model size and the number of parameters while maintaining relatively high-performance levels. Therefore, the image processing technology of MobileNet models has also been applied in various fields.

In agriculture, Dan *et al.* [14] proposed the SE-MobileNet V2 model by combining MobileNet V2 and Squeeze-and-ExcitationNetworks (SENet), effectively identifying pests in crops. Masykur *et al.* [15] used MobileNet as a framework to develop a rice plant disease and pest classification model and an object detection method based on leaf color, achieving a detection accuracy of 97%. Rajbongshi *et al.* [16] utilized the MobileNet model to study four types of rose diseases, achieving an accuracy of 95.63%. Yao *et al.* [17] proposed an underwater sea cucumber detection method based on an improved MobileNet-SSD (MD-SSD) to enhance the accuracy of MobileNet-SSD in sea cucumber detection. This is important for the rapid promotion of autonomous sea cucumber harvesting robots. In the medical field, Gasa *et al.* [18] introduced a skin disease diagnosis system and used the MobileNet CNN on a Raspberry Pi to identify types of skin lesions. Reddy *et al.* [19] proposed and compared an automatic melanoma cancer detection system based on the MobileNet architecture algorithm and CNN algorithm, showing that the MobileNet architecture

provided 75% accuracy, while CNN provided 65% accuracy. In other fields, Sun and Luo [20] proposed using a fine-tuned MobileNet V2 model as a feature extraction method for palm vein recognition, while Zhou *et al.* [21] and Tang *et al.* [22] respectively proposed an improved MobileNet model and an enhanced Mobilenet-SSD method for face recognition. They all mentioned that these methods are not only accurate but also more convenient to deploy on IoT devices.

The mentioned MobileNet models have a common advantage across different fields, namely, their small model size, making them convenient for deployment on IoT devices. In the study by Kavyashree *et al.* [23], the baseline architecture was modified to further reduce its size to 2.3 MB while maintaining an outstanding accuracy of 89.13%. This streamlined model is currently applicable to a wide range of mobile devices and embedded visual platforms. Additionally, Thin MobileNet and Slim MobileNet models were proposed separately by Sinha *et al.* [24] and Bouguezzi [25]. The size of the Thin MobileNet model is 9.9 MB, achieving an accuracy of 85.61%. The Slim MobileNet model, with a size of 7.3 MB, achieves an accuracy of 73.12% while reducing parameter count by over 41%. Compared to traditional image recognition models, these MobileNet models are more suitable for deployment on IoT devices. The compact size and high accuracy of these models are of significant importance for devices involved in wildfire recognition in wilderness environments.

A. *MobileNet V1 and MobileNet V2*

Howard *et al.* [26] introduced MobileNet V1, the first generation of MobileNet, featuring the DepthWise Separable Convolution technique. This technique significantly reduces computational complexity and model parameters through two key steps: DepthWise convolution, applying filters to each input channel, and pointwise convolution, effectively combining channel results while preserving model performance. DepthWise Separable Convolution includes DepthWise Convolution, applying filters separately to input channels, reducing computational load while retaining channel feature information, and Pointwise Convolution, combining DepthWise Convolution results to produce the final output feature map, as illustrated in Fig. 1.

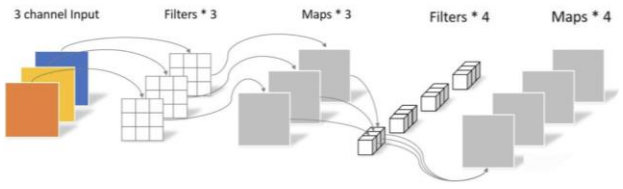

Fig. 1. DepthWise convolution and pointwise convolution.

Sandler *et al.* [27] introduced MobileNet V2, which brought forth a series of significant improvements to enhance the performance and versatility of deep learning models. This version of the MobileNet series inherits the advantages of its predecessor, MobileNet V1, and introduces innovative design elements that bolster its performance, non-linearity, and applicability. A crucial improvement in MobileNet V2 is the introduction of the Inverted Residuals structure, as depicted in Fig. 2 [28]. This structure represents a major architectural improvement in deep learning models. The core of this design is to reevaluate the traditional residual structure, making the model more nonlinear and performance-oriented.

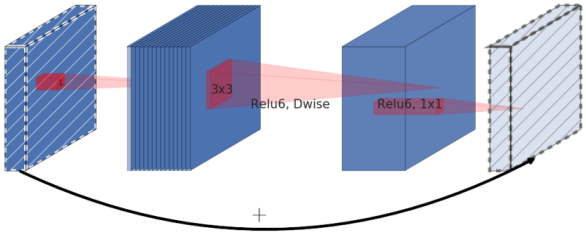

Fig. 2. Inverted residuals.

The release of MobileNet V2 not only represents the continuous evolution of the MobileNet series but also provides the deep learning community with more powerful tools to meet the growing demands of applications. Its improvements in performance, non-linearity, and versatility have opened up new possibilities for research and applications in the field of computer vision. As a result, MobileNet V2 has become a top choice for numerous deep learning research and practical applications.

B. *MobileNet V3 Large & Small*

Howard *et al*. [28] introduced MobileNet V3, featuring a key innovation known as the Squeeze-and-Excitation (SE) module, aimed at improving the model's performance. The SE module is another key component of MobileNet V3 with significant functionality. As depicted in Fig. 3 [30], its role is to adaptively readjust the importance of different channel features within the neural network, thereby enhancing the model's performance.

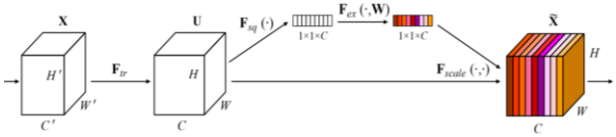

Fig. 3. Squeeze-and-Excitation (SE) block.

The SE block operates through the following four key steps:

(1) **F***tr* (Feature Map): This step involves transforming the input feature map $\boldsymbol{X}$ into a new feature map $\boldsymbol{U}$ using standard convolution. This convolutional operation helps capture various features from the input data.

(2) **F***sq* (Squeeze): The "squeeze" operation performs global average pooling on the $\boldsymbol{U}$ feature map, resulting in a $1 \times 1 \times C$ output $\boldsymbol{Z}$, where C represents the number of channels. $\boldsymbol{Z}$ is a $1 \times 1 \times C$ tensor containing global information. This step aggregates global information from all feature layers.

(3) **F***ex* (Excitation): The excitation step comprises two fully connected layers and a sigmoid activation function. It applies a non-linear transformation to the $\boldsymbol{Z}$ tensor and produces a $1 \times 1 \times C$ output s, where s is a tensor containing channel weight information. This step is designed to learn the importance of each feature layer by modeling the relationships between them.

(4) **F***scale* (Scale): In this step, a channel-wise multiplication operation scales each feature layer in $\boldsymbol{U}$ by its corresponding weight s. This results in an output tensor $\tilde{X}$ with dimensions $H \times W \times C$. The purpose of this step is to adjust the importance of each feature layer based on the learned importance values.

The SE block enhances the model's performance and representation through four key steps. It adaptively adjusts channel feature weights, capturing crucial features, and improving effectiveness across diverse tasks. Within the SE module recalibrates features through two steps: squeezing and exciting. Squeezing involves global average pooling, compressing channel features into a single value, representing the average importance. In the exciting step, a small neural network redistributes these importance values. This allows the SE module to learn channel importance and adjust feature responses. SE modules make MobileNet V3 more intelligent and efficient. By dynamically adjusting channel feature importance, the model focuses on informative features and suppresses less relevant ones. This adaptability improves overall performance, making it versatile across tasks and data distributions, leading to continuous development.

MobileNet V3 comes in two versions, MobileNet V3 Large and MobileNet V3 Small, catering to different needs and application scenarios. This provides users with more choices and flexibility.

- MobileNet V3 Large: Suited for tasks demanding high model accuracy. It is meticulously designed to excel in applications requiring high precision, such as image classification and object recognition.
- MobileNet V3 Small: Optimized for low-latency real-time applications. This version is designed to be lightweight, ensuring efficient performance in resource-constrained environments. MobileNet V3 Small is an ideal choice for real-time tasks on mobile devices with limited computational resources.

In summary, MobileNet V3 offers users the choice to find an ideal balance for different application scenarios. MobileNet V3 Large excels in tasks requiring high accuracy, while MobileNet V3 Small focuses on low latency and efficiency, making it particularly suitable for real-time applications on resource-constrained devices.

In the context of IoT devices dedicated to wildfire recognition in outdoor environments, careful consideration of computational resources is crucial. Therefore, choosing lightweight deep learning models is a critical strategy. Lightweight models have smaller model sizes and fewer parameters, allowing them to operate efficiently in resource-constrained environments. This is extremely important for outdoor applications, as IoT devices need to recognize wildfires in real-time, respond quickly to potential hazards, and take the necessary actions. Thus, IoT devices deployed in the outdoors must strike a balance between performance and computational resources to ensure their reliability and practicality. Choosing lightweight deep learning models has become a key strategy in achieving this goal.

III. Materials and Methods

A. Experimental Environments and Wildfire Database

In this study, we used the MobileNet model from the PyTorch library for wildfire database feature recognition. The hardware setup for the experiments includes a computer equipped with a 12th Gen Intel(R) Core(TM) i7-12700KF 3.61 GHz processor and an NVIDIA GeForce RTX 3060 Ti graphics card. The operating system used for the experiments is Windows 11 Professional, and GPU acceleration was utilized for training. The software environment includes NVIDIA CUDA 11.8 and PyTorch 2.0.1, with the programming language being Python 3.10.

The wildfire database is a compilation of data from various sources, including the "Forest fire" photo collection from the Kaggle website and data obtained by searching for wildfire-related photos. In total, there are 4,526 wildfire photos and 10,799 non-wildfire photos. To conduct training and testing, the wildfire database was divided into two main categories: "Train" and "Test". The Train set comprises 3,168 wildfire photos and 7,478 non-wildfire photos, while the Test set contains 1,358 wildfire photos and 3,321 non-wildfire photos. The Train set is primarily utilized for training the wildfire model, while the Test set is used to evaluate the model's performance. Fig. 4 is the samples from wildfire database.

Fig. 4. Samples from wildfire database.

B. Evaluation of Model Performance

Ruby and Yendapalli [29] recommended that the evaluation metrics used for assessing model performance include accuracy and the cross-entropy loss function.

C. Accuracy

Accuracy is an important indicator for measuring the classification performance of a model. It measures the

proportion of the number of samples correctly classified by the model to the total number of samples. After each training and testing epoch, we extract predictions from the model's output. The output of the model is a vector containing the predicted probabilities for each class, where each element represents the probability of the corresponding class. Next, we find the category label corresponding to the highest probability by comparing the predicted probabilities of each category. This can be achieved with the following code:

preds = torch.max(outputs, 1)

where *outputs* represent the model's prediction output, and *preds* are the model's predicted class labels.

Then, we compare the predictions of the wildfire detection model with the actual labels to determine which wildfire samples were correctly classified. For each sample, if the model's prediction matches the actual label, we consider it correctly classified. Finally, in each iteration cycle, we accumulate the count of samples that were correctly classified. This can be achieved with the following code:

train_acc + = torch.sum(preds == labels.data)

where *train_acc* is the number of correct classifications used to accumulate the training set.

Regarding the calculated accuracy value of the wildfire detection model, once we accumulate the number of correctly classified samples, we can calculate the accuracy. The accuracy is calculated by

$$Train\ Accuracy = \frac{train\_acc}{train\_sum} \tag{1}$$

$$Test\ Accuracy = \frac{test\_acc}{test\_sum} \tag{2}$$

This is a very common performance metric used to evaluate a model's performance in classification tasks. The higher the accuracy, the better the model's performance.

For the training set, we divide *train_acc* by the total number of samples in the training set to calculate the training set's accuracy. For the test set, we divide *test_acc* by the total number of samples in the test set to calculate the test set's accuracy. The accuracy value is a proportion between 0 and 1: when accuracy is close to 1, it means the model correctly classifies most of the samples, and when it is close to 0, it indicates a poorer performance.

In our experiments, accuracy is an important performance evaluation metric used to assess the wildfire detection model's performance. We monitor both the training set accuracy and the test set accuracy, providing insights into the model's classification performance on different datasets. The training set accuracy reflects the model's classification ability on the training data. As training progresses, the training set accuracy gradually improves, indicating that the model is continuously enhancing its classification performance on the training data. This is a positive sign because it means the model is learning more features and patterns from the known data. In contrast, the test set accuracy is more general because it evaluates the model's performance on previously unseen new data, demonstrating its generalization ability. When the test set accuracy increases, it indicates that our wildfire detection model performs exceptionally well on new data and can effectively generalize to different situations and scenarios. This is crucial for practical applications because the model needs to detect wildfires in unknown environments.

Through the calculation and analysis of accuracy, we can quantitatively assess the performance of the wildfire detection model to determine its effectiveness and accuracy in classification tasks.

D. CrossEntropyLoss

The cross-entropy loss function (CrossEntropyLoss) is a common loss function used for classification problems to measure the difference between the predicted probability distribution of the model and the actual label distribution. In the wildfire detection classification task, there are two classes for each sample: "fire" and "no fire". The actual class label is represented by a vector y_i of length 2, where only one element is 1, indicating the true class. The wildfire detection model's prediction is represented by a probability distribution vector p_i, also of length 2, representing the model's predicted probabilities for each class.

The cross-entropy loss is calculated by

$$L(y_i, p_i) = -\sum_{j=1}^{2} y_i^j \, log\left(p_i^j\right) \tag{3}$$

where y_i^j is the *j*-th element of the actual label, and p_i^j is the *j*-th element of the model's predicted probability. Cross-entropy loss measures the loss by summing the log-probability product of each element of the true label and the corresponding predicted probability. During training, we calculate the cross-entropy loss for all samples and average them to obtain the average loss for the entire dataset. The smaller the average loss, the closer the prediction of the wildfire detection model is to the true label, and the better the performance of the wildfire detection model.

In summary, the cross-entropy loss function is used to evaluate the performance of the model on the classification task. By comparing the difference between the model's predicted probability and the true label, we can measure the accuracy of the model. The smaller the loss value, the better the model performs in the classification task.

IV. Result and Analysis

In this study, we used three deep learning models for wildfire database recognition training: MobileNet V2, MobileNet V3 Large, and MobileNet V3 Small. These models were chosen on the basis of their widespread applications and performance in deep image processing. Training was performed for 60 rounds. Our main findings are discussed below. Figs. 5 and 6 respectively show the training accuracy and test accuracy, where the horizontal axis in both figures represents the number of training rounds, and the vertical axis represents accuracy.

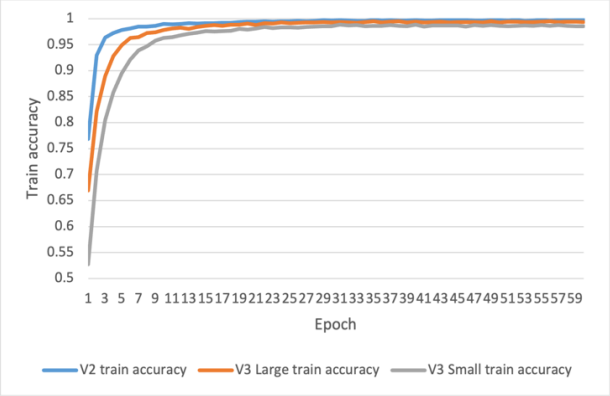

Fig. 5. Train accuracy of three MobileNet models.

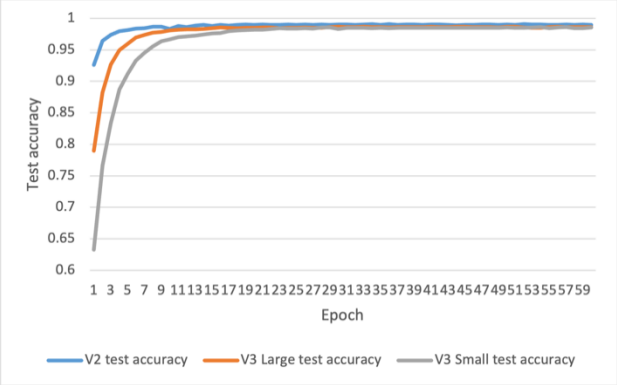

Fig. 6. Test accuracy of three MobileNet models.

MobileNet V2 exhibited the highest classification accuracy, indicating its outstanding performance in wildfire image recognition. MobileNet V3 Large followed, with MobileNet V3 Small ranking last in terms of performance. These results highlight the impact of model architecture on performance, especially in tackling the challenging task of wildfire image recognition. Figs. 7 and 8 respectively show the training loss and test loss, where the horizontal axis in both figures represents the number of training rounds, and the vertical axis represents the loss values.

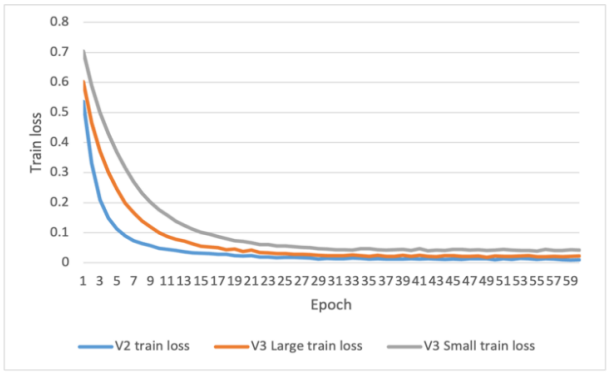

Fig. 7. Train loss of three MobileNet models.

MobileNet V2 exhibited the lowest loss during training, indicating that it converged to the optimal solution more quickly. MobileNet V3 Large followed, while MobileNet V3 Small performed less efficiently in this regard. These results have practical significance for the training and optimization of deep learning models.

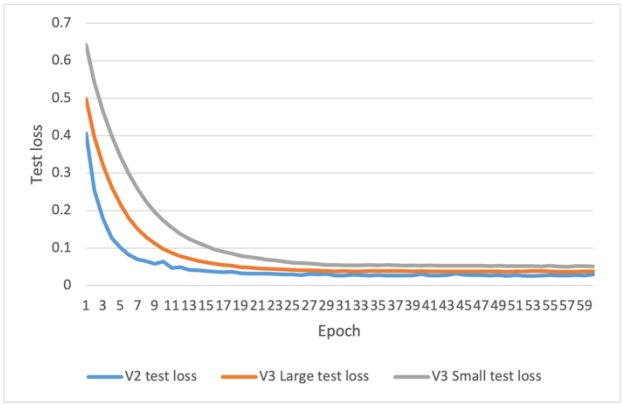

Fig. 8. Test loss of three MobileNet models.

We also calculated the training times for the three models, as shown in Fig. 9. The horizontal axis represents the three different models (MobileNet V2, MobileNet V3 Large, and MobileNet V3 Small), and the vertical axis represents the training time, specifically the time required for 60 rounds of training. By observing these time data, we can gain insight into the training speed of different models. In terms of time efficiency, MobileNet V3 Large exhibited an excellent performance, followed by MobileNet V2, while MobileNet V3 Small required more time to complete the same number of training rounds. These results provide valuable clarity as to the efficiency of these models during the training process.

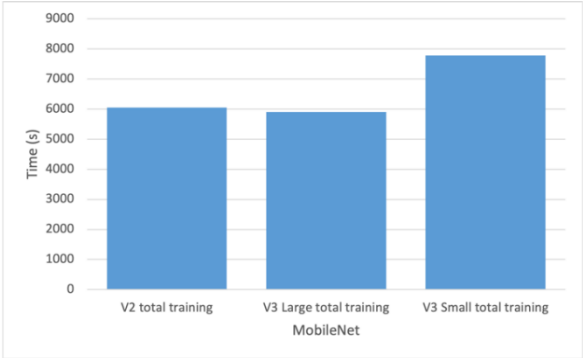

Fig. 9. Training times (for 60 epochs) of three MobileNet models.

These findings on loss values and training times provide us with important information about the performance and training efficiency of different models, enabling the informed selection of the best model for specific needs.

V. Conclusion

The results of this study provide valuable insights on the selection and configuration of models used in IoT devices for deep image processing. For IoT devices deployed in outdoor environments, portability, ease of installation, and uninstallation are crucial factors.

Depending on the varying performance and computational resource constraints of IoT devices, the following conclusions can be drawn:

- When IoT devices have sufficient computational resources, prioritizing the high-performance MobileNet V2 model is the best choice to achieve the highest accuracy and reliability.
- When IoT devices have limited computational resources, MobileNet V3 Large becomes a more prudent choice, as it delivers good performance within a shorter training time.
- For embedded IoT devices that require frequent updates of the wildfire detection model, the shorter training time of MobileNet V3 Large is a significant advantage.

Further research and optimization can enhance the performance and efficiency of deep learning models on IoT devices to meet practical demands such as wildfire detection.

These conclusions offer strong guidance for the selection and configuration of IoT devices, underscoring the flexibility and adaptability of deep learning models in different application scenarios.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

AUTHOR CONTRIBUTIONS

Shiyan Du, Jiacheng Li, and Masato Noto conducted the research; Shiyan Du did the experiments and wrote the paper; Jiacheng Li analyzed the data and also wrote the paper; Masato Noto gave his advice; all authors had approved the version.

REFERENCES

[1] Q. Tran, R. Gayhardt, T. Nguyen, and A. Zaman, "Recent US west coast wildfire disasters: Impact on the reliability assessment of optical transceivers," in *Proc. 2022 IEEE International Reliability Physics Symposium*, 2022, pp. 1–4.

[2] Q. Xie, X. Quan, and B. He, "Wildfire danger assessment over southwest China based on short-term features of weather and fuel variables," in *Proc. 2021 IEEE International Geoscience and Remote Sensing Symposium*, 2021, pp. 8648–8651.

[3] S. B. Shah, T. Grübler, L. Krempel *et al*., "Real-time Wildfire detection from space—A trade-off between sensor quality, physical limitations and payload size," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 42, pp. 209–213, 2019.

[4] R. U. Shaik, L. Giovanni, and L. Fusilli, "Dynamic wildfire fuel mapping using sentinel–2 and PRISMA hyperspectral imagery," in *Proc. IEEE International Geoscience and Remote Sensing Symposium*, pp. 5973–5976, 2022.

[5] M. F. G. Ferreyra, G. Curci, L. D. Ceca *et al.*, "Monitoring air pollution from wildfires using ground data, satellite products and modeling: The austral summer 2016–2017 in Argentina," in *Proc. IEEE International Geoscience and Remote Sensing Symposium*, 2019, pp. 7630–7633.

[6] V. Kotroni, C. Cartalis, S. Michaelides *et al*., "DISARM early warning system for wildfires in the eastern mediterranean," *Sustainability*, vol. 12, no. 16, 6670, 2020.

[7] F. Zhang, P. Zhao, S. Xu *et al*., "Integrating multiple factors to optimize watchtower deployment for wildfire detection," *Science of the Total Environment*, vol. 737, 139561, 2020.

[8] P. H. Kashika and R. B. Venkatapur, "Deep learning technique for object detection from panoramic video frames," *International Journal of Computer Theory and Engineering*, vol. 14, no. 1, pp. 20–26, 2022.

[9] X. Xie, H. Li, and F. Hu, "The flocs target detection algorithm based on the three frame difference and enhanced method of the OTSU," *International Journal of Computer Theory and Engineering*, vol. 7, no. 3, 2015.

[10] N. Ye, R. Wang, and N. Li, "A novel active object detection network based on historical scenes and movements," *International Journal of Computer Theory and Engineering*, vol. 13, no. 3, pp. 79–83, 2021.

[11] F. Xie and Z. Huang, "Aerial forest fire detection based on transfer learning and improved faster RCNN," in *Proc. 2023 IEEE 3rd International Conference on Information Technology, Big Data and Artificial Intelligence*, 2023, vol. 3, pp. 1132–1136.

[12] S. Wu and Z. Libing, "Using popular object detection methods for real-time forest fire detection," in *Proc. 2018 11th International Symposium on Computational Intelligence and Design*, 2018, vol. 1, pp. 280–284.

[13] X. Bai and Z. Wang, "Research on forest fire detection technology based on deep learning," in *Proc. 2021 International Conference on Computer Network, Electronic and Automation*, 2021, pp. 85–90.

[14] B. Dan, X. Sun, and L. Liu, "Identification of diseases and pests in Lycium Barbarum using SE-MobileNet V2 algorithm," in *Proc. 2019 12th International Symposium on Computational Intelligence and Design*, 2019, vol. 1, pp. 121–125.

[15] F. Masykur, K. Adi, and O. D. Nurhayati, "Nurhayati, classification of paddy leaf disease using MobileNet model," in *Proc. 2022 IEEE 8th International Conference on Computing, Engineering and Design*, 2022, pp. 1–4.

[16] A. Rajbongshi, T. Sarker, M. M. Ahamad, and M. M. Rahman, "Rose diseases recognition using MobileNet," in *Proc. 2020 4th International Symposium on Multidisciplinary Studies and Innovative Technologies*, 2020, pp. 1–7.

[17] Y. Yao, Z. Qiu, and M. Zhong, "Application of improved MobileNet-SSD on underwater sea cucumber detection robot," in *Proc. 2019 IEEE 4th Advanced Information Technology, Electronic and Automation Control Conference*, 2019, pp. 402–407.

[18] Z. N. F. Gasa, P. A. Owolawi, T. Mapayi, and K. Odeyemi, "MobileNet neural network skin disease detector integrated with raspberry pi and telegram," in *Proc. 2020 International Conference on Artificial Intelligence, Big Data, Computing and Data Communication Systems*, 2020, pp. 1–5.

[19] K. V. Reddy and L. R. Parvathy, "An innovative analysis of predicting melanoma skin cancer using MobileNet and convolutional neural network algorithm," in *Proc. 2022 2nd International Conference on Technological Advancements in Computational Sciences*, 2022, pp. 91–95.

[20] Q. Sun and X. Luo, "A new image recognition combining transfer learning algorithm and MobileNet V2 model for palm vein recognition," in *Proc. 2022 4th International Conference on Frontiers Technology of Information and Computer*, 2022, pp. 559–564.

[21] Y. Zhou, Y. Liu, G. Han, and Y. Fu, "Face recognition based on the improved MobileNet," in *Proc. 2019 IEEE Symposium Series on Computational Intelligence*, 2019, pp. 2776–2781.

[22] J. Tang, X. Peng, X. Chen, and B. Luo, "An improved MobileNet-SSD approach for face Detection," in *Proc. 2021 40th Chinese Control Conference*, 2021, pp. 8072–8076.

[23] P. S. P. Kavyashree, M. El-Sharkawy, "Compressed MobileNet V3: A lightweight variant for resource-constrained platforms," in *Proc. 2021 IEEE 11th Annual Computing and Communication Workshop and Conference*, 2021, pp. 104–107.

[24] D. Sinha and M. El-Sharkawy, "Thin MobileNet: An enhanced MobileNet architecture," in *Proc. 2019 IEEE 10th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference*, 2019, pp. 0280–0285.

[25] S. Bouguezzi, H. Faiedh, and C. Souani, "Slim MobileNet: An enhanced deep convolutional neural network," in *Proc. 2021 18th International Multi-Conference on Systems, Signals & Devices*, 2021, pp. 12–16.

[26] A. G. Howard, M. Zhu, B. Chen *et al*., "MobileNets: Efficient convolutional neural networks for mobile vision applications," arXiv preprint, arXiv:1704.04861, 2017.

[27] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4510–4520.

[28] A. Howard, M. Sandler, G. Chu *et al*., "Searching for MobileNetV3," in *Proc. the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 1314–1324.
[29] U. Ruby and V. Yendapalli, "Binary cross-entropy with deep learning technique for image classification," *Int. J. Adv. Trends Comput. Sci. Eng*., vol. 9, no. 10, 2020.
[30] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7132–7141.