Journal of Advanced College of Engineering and Management

Comparative Study of CNN and MobileNetV2 for Forest Fire Detection with Data Augmentation

[1*]Jatin Raut, [2]Rajad Shakya
[1,2]*Department of Electronics and Computer Engineering, Thapathali Campus, Kathmandu, Nepal*
Corresponding email: [*]jatinraut3@gmail.com

DOI: 10.3126/jacem.v12i01.93905

Abstract

Forest fires pose a significant environmental threat, and quick, reliable detection is essential to reduce damage. This paper compares lightweight custom Convolutional Neural Network (CNN) with a fine-tuned MobileNetV2 for automated forest fire detection. Both models were trained on a Kaggle dataset containing 5,050 labeled forest images. The images underwent preprocessing that included resizing and normalization, and they were improved through extensive data augmentation like rotation, zoom, shifting, and flipping. The CNN achieved an accuracy of 92% with an F1-score of 0.90 for detecting fire. In comparison, MobileNetV2 achieved a higher accuracy of 95% with an F1-score of 0.94, cutting false negatives by 39%. Both models reached a ROC-AUC of 0.99, but MobileNetV2 showed better sensitivity, making it more suitable for real-time monitoring systems like drones and satellites. The results show that while CNN is robust with fewer false positives, MobileNetV2 strikes a better balance between precision and recall, making it the preferred choice for forest fire detection.

Keywords*—Convolutional Neural Network, Deep Learning, Forest Fire, Image Processing, Surveillance Systems*

1. INTRODUCTION

Forest fires are among the most destructive natural disasters. They cause significant ecological and economic losses and threaten human life. They spread quickly, release greenhouse gases, and harm biodiversity. This makes early detection of forest fires essential for effective suppression efforts. Traditional methods like satellite monitoring, sensor networks, and human patrols often face delays, limited coverage, or interference from the environment, which decreases their reliability for early detection.

The increased use of computer vision and deep learning has enabled image-based forest fire detection in real time. Unlike methods based on handcrafted features, Convolutional Neural Networks (CNNs) and transfer learning models like MobileNetV2 can learn important features from images automatically. This improves accuracy and robustness. However, there hasn't been enough exploration of the trade-offs between lightweight custom CNN models and advanced pre-trained models for forest fire detection.

In this study, we provide a comparison of two models: (1) a lightweight custom CNN designed from scratch, and (2) a fine-tuned MobileNetV2 model trained on a balanced dataset of forest fire images. By using systematic preprocessing and data augmentation, we assess their performance with multiple evaluation metrics such as accuracy, precision, recall, F1-score, and ROC-AUC. Our findings show that MobileNetV2 consistently outperforms the CNN, particularly in reducing false negatives, which is crucial for real-world fire detection. This study demonstrates the potential of MobileNetV2 as a strong and practical solution for real-time wildfire detection systems.

2. LITERATURE REVIEW

Forest fire detection has been studied using traditional sensor-based and vision-based methods. Wang et al. [1] noted that infrared and smoke sensors work well for localized monitoring, but they struggle in large-scale scenarios due to interference. Shang et al. [2] showed that satellite remote sensing can cover wider areas, but it faces delays and limited time resolution. Neelakandan et al. [3] used vision-based features like color and texture. However, as Madhuri et al. [4] pointed out, these methods often result in false positives.

The introduction of deep learning, particularly CNNs, led to significant improvements. Seyd et al. [5] and Zhao et al. [6] achieved high accuracy by automatically learning features from image datasets. Zhang et al. [7] expanded on this by combining CNNs with recurrent models to manage temporal changes in video. To address dataset limitations, Alkhatib et al. [8] and Verma et al. [9] emphasized the use of data augmentation and synthetic datasets for added strength. Du et al. [10] compared three MobileNet-based architectures and showed that lightweight transfer learning models offer competitive accuracy while significantly lowering computational costs, making them very suitable for wildfire detection.

3. METHODOLOGY

A. Dataset Description

The dataset used in this project is called "Forest Fire Images." It is publicly available on Kaggle and was curated by Mohnish Sai Prasad [11]. This dataset has 5,050 RGB images sorted into two categories: 'fire' and 'non Fire.' The fire category includes images with visible flames or smoke. The non fire category consists of forest or nature scenes without any signs of fire. The images differ in environmental conditions like lighting, camera angle, and vegetation type, adding complexity to the classification task. All images are resized to 128×128 pixels to maintain uniform input sizes for the CNN model. Each image's pixel values are normalized to the range [0, 1], which helps in faster and more stable training of the neural network. The dataset is divided into training and validation sets with an 80:20 ratio. Stratified sampling is used to keep the class balance. To increase the diversity of the training data and avoid overfitting, we apply various augmentation techniques with Keras's ImageDataGenerator.

These techniques include random rotations of up to 15 degrees, zooming by up to 10%, horizontal and vertical shifts of up to 10% of the image dimensions.

B. Image Preprocessing

The first step in the methodology is to prepare the image dataset for model training. All images from the two classes, 'fire' and 'non fire' are loaded from a publicly available dataset titled "Forest Fire Images" obtained from Kaggle. These images are resized to a fixed dimension of 128×128 pixels with three color channels (RGB) to ensure a uniform input size for the CNN. This resizing uses bilinear interpolation. After resizing, each image is turned into a NumPy array, and pixel values are normalized by scaling them to the range [0, 1]. This normalization speeds up convergence during training and improves overall model stability. The class labels are encoded as numbers, with 1 for fire and 0 for non fire. Finally, the complete dataset is split into training and validation sets using an 80:20 split, making sure that both classes are equally represented in each subset.

The general formula for mapping coordinates from the resized image back to the original image (used during interpolation) is:

$$X_{orig} = \frac{x_{new}}{W_{target}} \times W_{orig}, \quad Y_{orig} = \frac{y_{new}}{H_{target}} \times H_{orig} \#(1)$$

Where:

- W= width, H=height
- $\left(X_{new}, Y_{new}\right): pixelinresizedimage$
- $\left(X_{orig}, Y_{orig}\right): mappedpixelinresizedimage$

$$Normalized\ Pixel = \frac{Original\ Pixel}{255} \#(2)$$

C. Data Augmentation

To improve the diversity of the training data and reduce overfitting, data augmentation with Keras's ImageDataGenerator was used. Augmentation only occurs on the training data, not on the validation set. The following transformations are applied:

- **Rotation Range of 15°**: This randomly rotates images within a ±15-degree window to simulate different viewing angles.
- **Zoom Range of 10%**: This helps the model be more robust to various distances from the fire source.
- **Width and Height Shift Range of 10%**: This mimics camera displacement and variability in forest scenes.
- **Horizontal Flip**: This mirrors images, making the model indifferent to directional orientation.

This augmentation strategy increases the effective size and variability of the training dataset without needing to manually collect more images. It enables the model to generalize better to real-world situations.

Figure 1: Data Augmentation

D. Model

a. Custom CNN

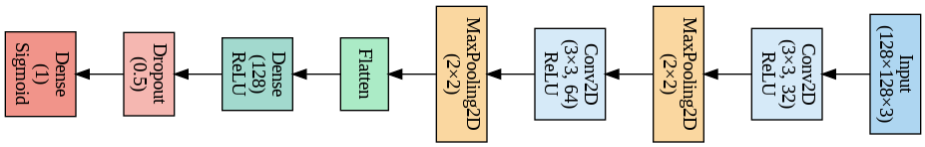

Figure 2: Custom CNN

The system uses a custom Convolutional Neural Network (CNN) created with Keras's Sequential API. The architecture starts with a Conv2D layer that has 32 filters of size 3×3, using the ReLU activation function to add non-linearity. This is followed by a MaxPooling2D layer with a pool size of 2×2, which reduces the dimensions of the feature maps. The second stage includes another Conv2D layer with 64 filters and ReLU activation, followed again by a 2×2 max pooling operation. In the third block, a Conv2D layer with 128 filters and ReLU activation is applied, followed by another MaxPooling2D layer. These convolutional and pooling layers work together to extract and downsample important spatial features from the input images. After the final pooling operation, a Flatten layer converts the resulting 2D feature maps into a 1D feature vector, which serves as the input for the fully connected part of the network. A Dense layer with 128 neurons and ReLU activation is added next to capture complex feature interactions. This is followed by a Dropout layer with a dropout rate of 50% to reduce overfitting by randomly deactivating neurons during training. The output layer is a Dense layer with one neuron and a sigmoid activation function, which provides a probability score indicating the likelihood that the input image contains fire. This CNN architecture balances computational complexity and performance, making it suitable for real-time applications where resources may be limited.

b. MobileNetV2

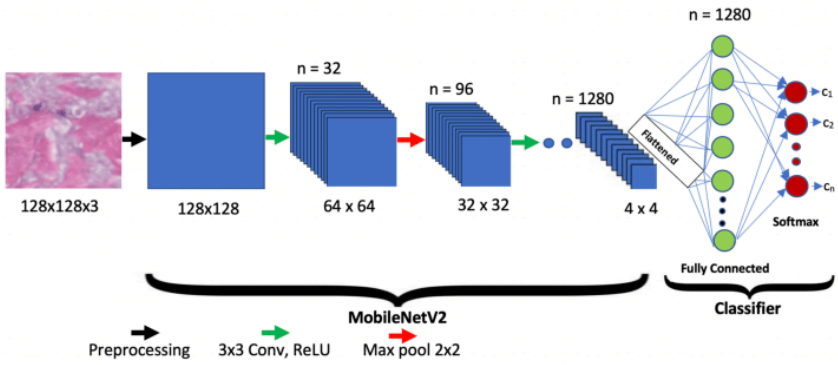

Figure 3: MobileNetV2 Architecture (adapted from Sandler et al. [12])

MobileNetV2 is a lightweight and efficient deep learning architecture created by Google. It is specifically designed for mobile and embedded vision applications with limited computational resources. This version improves on the original MobileNet by introducing key features like inverted residual blocks and linear bottlenecks. These innovations help it maintain high accuracy while greatly reducing the number of parameters and computational demands. The architecture also depends on depthwise separable convolutions to achieve its efficiency.

MobileNetV2 is commonly used as a flexible backbone for various computer vision tasks. These tasks include image classification, object detection often used with frameworks like SSDLite and semantic segmentation. Its ability to balance performance and efficiency makes it a great choice for real-time processing on devices such as smartphones and edge devices. It is suitable for applications like face recognition, augmented reality, medical image analysis—like skin cancer or COVID-19 detection and even waste classification.

E. Model Compilation and Training

The Adam optimizer, which is renowned for its quick convergence and flexibility, is used to compile the model. Binary cross-entropy is chosen as the loss function because it is suitable for binary classification tasks. Accuracy serves as the main evaluation metric during training. Using the augmented training data generator and the non-augmented validation data generator, the model is trained using the .fit() function. A batch size of 32 is used, and training is spread across 25 to 50 epochs. If, after a predetermined number of epochs (patience), the model stops improving, an EarlyStopping callback is used to track the validation loss and end training early. This lessens the chance of overfitting and pointless calculations.

$$L_{BCE} = -\frac{1}{N}\sum_{i=1}^{N}[y_i \, log(\hat{y}_i) \; + \; (1 - y_i) log(1 - \hat{y}_i)] \tag{3}$$

Where:

- N = Total number of samples
- $y_i = Truelabelforsamplei\left(y_i \in 0, 1\right)$
- $\hat{y}_i = Predicted\ probability\ for\ class\ 1\ for\ sample\ i\ (\hat{y}_i \in (0, 1))$

4. RESULTS AND DISCUSSION

This section provides a comparison of the performance of two different deep learning models made for forest fire detection: a custom Convolutional Neural Network (CNN) and a fine-tuned MobileNetV2 model. Both models were trained on a dataset of forest images, which were divided into 'Fire' and 'Non Fire' classes. They were tested under the same experimental conditions to make sure the comparison was fair.

A. Model Performance Overview

Both the custom CNN and MobileNetV2 models were trained for 10 epochs with an Adam optimizer and binary cross-entropy loss. Data augmentation techniques, such as rotation, zoom, width and height shifts, and horizontal flips, were applied to the training set for both models. This improved their ability to generalize.

The custom CNN model quickly increased its training accuracy, reaching about 93.15% by the 10th epoch. Its validation accuracy peaked at 93.07%. The MobileNetV2 model also showed solid training progress, achieving an accuracy of 95.00% on the test set. Both models demonstrated good generalization and minimal overfitting, as shown by their validation curves. The Area Under the Curve (AUC) score was 0.99 for both models, which indicates excellent ability to distinguish between classes.

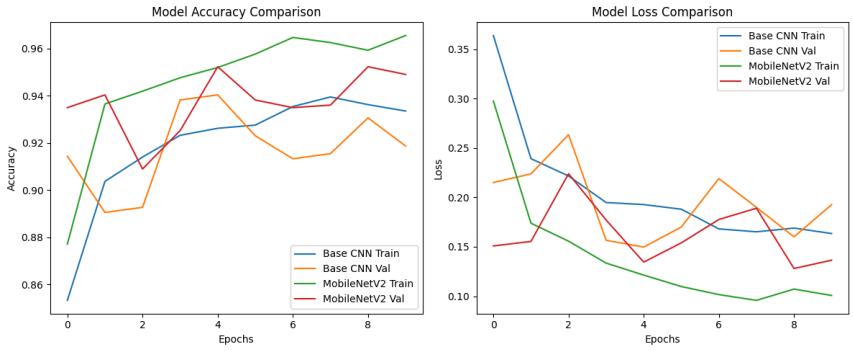

Figure 4: Comparison of accuracy and loss

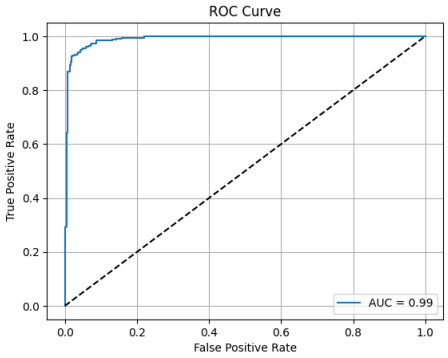

Figure 5: ROC Curve of Custom CNN

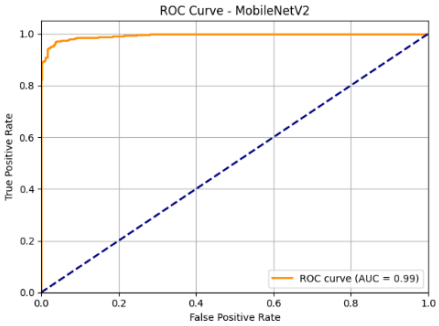

Figure 6: ROC Curve of MobileNetV2

B. Comparative Classification Performance

Table 1: Custom CNN Classification Report

Class	Precision	Recall	F1-Score	Support
0 (Non_Fire)	0.87	0.99	0.93	500
1 (Fire)	0.99	0.83	0.90	423
Accuracy			0.92	493
Weighted Avg	0.93	0.91	0.92	493
Macro Avg	0.93	0.91	0.92	493

Table 2: MobileNetV2 Classification Report

Class	Precision	Recall	F1-Score	Support
0 (Non_Fire)	0.92	0.99	0.95	500
1 (Fire)	0.99	0.90	0.94	423
Accuracy			0.95	923
Weighted Avg	0.95	0.95	0.95	923
Macro Avg	0.95	0.95	0.95	923

The trained models were carefully tested on the unseen test set. We used classification reports and confusion matrices for evaluation.

Comparing the classification reports in Table 1 and Table 2, MobileNetV2 achieved a slightly higher overall accuracy of 95% than the custom CNN's 92%. For the 'Non Fire' class, the custom CNN showed a very high recall of 0.99 and a precision of 0.87, indicating it was effective at identifying non-fire instances. MobileNetV2 also did well for 'Non Fire', with a precision of 0.92 and a recall of 0.99.

For the important 'Fire' class, the custom CNN had an exceptionally high precision of 0.99, which suggests a very low rate of false positives. Its recall for 'Fire' was 0.83. In contrast, MobileNetV2 maintained high precision at 0.99 for 'Fire', and it had a superior recall of 0.90, meaning it correctly identified a larger portion of actual fire instances. The F1-scores for the 'Fire' class were 0.90 for the custom CNN and 0.94 for MobileNetV2. This highlights that MobileNetV2 had a slightly better balance between precision and recall for this class.

C. Analysis of Misclassifications

The confusion matrices in Figure 7: MobileNetV2 Confusion Matrix and Figure 8: CNN Confusion Matrix showed a detailed breakdown of correct and incorrect classifications for each model. The custom CNN misclassified only 4 'Non Fire' images as 'Fire' (false positives), which shows its strong ability to avoid false alarms. However, it missed 71 actual 'Fire' instances (false negatives). MobileNetV2 had the same number of false positives, misclassifying 4 'Non Fire' images as 'Fire', but it significantly reduced false negatives to 43. This trade-off is important in fire detection. The custom CNN is more cautious in predicting fire, which minimizes false alarms, while MobileNetV2 is more aggressive and reduces the risk of missing real fires.

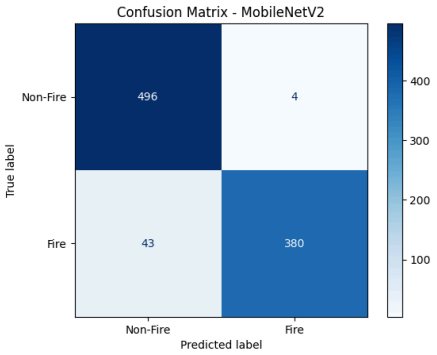

Figure 7: MobileNetV2 Confusion Matrix

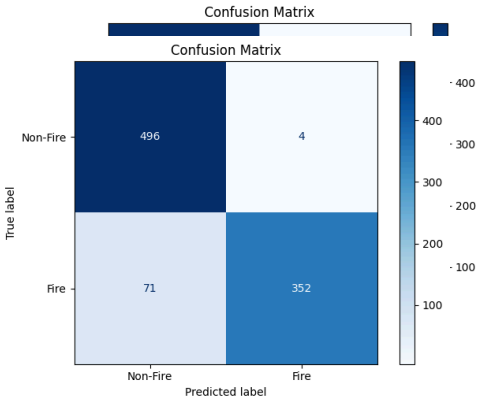

Figure 8: CNN Confusion Matrix

5. CONCLUSION AND FUTURE WORK

This paper presented a deep learning approach for forest fire detection using both a custom CNN and a lightweight MobileNetV2 model enhanced with data augmentation techniques. The MobileNetV2 model achieved a higher accuracy of 95% compared to 92% for the

custom CNN, along with superior recall (90% vs. 83%) for fire class. This demonstrates MobileNetV2's better performance and efficiency, making it the preferred choice for real-time monitoring applications such as drones, satellite imagery, and surveillance systems. The system effectively distinguished fire from non-fire images through careful preprocessing and training.

Future work includes expanding the dataset with more diverse and real-time imagery, integrating environmental sensor data for multimodal detection, extending classification to fire severity levels, optimizing MobileNetV2 for deployment on low-power edge devices, and incorporating real-time alert systems to improve forest fire management effectiveness.

6. ACKNOWLEDEGEMENT

I would like to thank Er. Rajad Shakya for supervision and my family for their support. The Department of Electronics and Computer Engineering, Thapathali Campus, Institute of Engineering provided the necessary infrastructure and academic environment.

REFERENCES

[1] Wang, Y., Tee, K. M., & Yoong, R. T., A systematic literature review of vision-based fire detection, prediction, and forecasting, Jurnal Kejuruteraan, vol. 37, no. 1, pp. 191–218, 2025.

[2] Shang, D., et al., Deep learning-based forest fire risk research on monitoring and early warning algorithms, Fire, vol. 7, no. 4, p. 151, 2024.

[3] Neelakandan, P. M., et al., Deep learning-based wildfire image detection and classification systems for controlling biomass, Applied Mathematics and Information Sciences, vol. 17, no. 1, pp. 127–136, 2023.

[4] Madhuri, C. R., et al., Accurate classification of forest fires in aerial images using ensemble model, Bulletin of Electrical Engineering and Informatics, vol. 13, no. 4, pp. 2650–2658, 2024.

[5] Seyd, S., et al., Fire-net: A deep learning framework for active forest fire detection, Journal of Electrical and Electronics Engineering, 2022.

[6] Zhao, J., et al., SVM based forest fire detection using static and dynamic features, Lecture notes in computer science, pp. 1–8, 2020.

[7] Zhang, Q., et al., Deep learning method for forest fire detection using unmanned aerial vehicles, Proceedings of the International Conference on Artificial Intelligence, Big Data, Computing and Data Communication Systems, pp. 151–157, 2019.

[8] Alkhatib, A., & Jaber, K., Advances in forest fire detection, prediction and behavior: A comprehensive survey, Journal of Computer Science, 2024.

[9] Verma, D., et al., Synthetic augmentation for robust forest fire detection via CNN, IEEE Access, vol. 10, pp. 123456–123468, 2025.

[10] Du, S., Li, J., & Noto, M., Comparison and Analysis of Three MobileNet-Based Models for Wildfire Detection, Journal of Advances in Information Technology, vol. 15, no. 4, pp. 511–518, 2024.

[11] Prasad, M. S., Forest fire images, On-line, Available: https://www.kaggle.com/datasets/mohnishsaiprasad/forest-fire-images, 2021.

[12] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. C., MobileNetV2: Inverted residuals and linear bottlenecks, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4510–4520, 2018.