Accurate classification of forest fires in aerial images using ensemble model

Ch Raga Madhuri[1], Sravya Sri Jandhyala[1], Deepthi Meenakshi Ravuri[1], Vunnava Dinesh Babu[2]
[1]Department of Computer Science and Engineering, Velagapudi Ramakrishna Siddhartha Engineering College, Vijayawada, India
[2]Department of Computer Science and Engineering, RV Institute of Technology, Guntur, India

Article Info

Article history:

Received Apr 19, 2023
Revised Dec 7, 2023
Accepted Feb 12, 2024

Keywords:

Convolutional neural network
InceptionV3
Max ensembling
ResNet50
VGG-16

ABSTRACT

This paper proposes a method to identify forest fires in aerial images using three different convolutional neural networks (CNNs). Unlike general approaches that make use of a single CNN to classify the images, the proposed solution uses the outcomes of different CNNs and considers the most predicted class. This method overcomes the problems associated with using a single CNN, such as low accuracy due to the drawbacks associated with that model. The three different classifiers used here are InceptionV3, VGG-16, and ResNet50. Classification is carried out based on the presence of fire or smoke features in the images. The individual predictions are combined using max-ensembling. The performance is analyzed using metrics like precision, recall, accuracy and F1-score. From the work, it was found that the combined model resulted in an accuracy of 95.8%. The results confirm that the final model provides greater classification accuracy than the individual models. The proposed method can be used to predict forest fires from live aerial images more accurately and help reduce the damage caused.

Corresponding Author:

Ch Raga Madhuri
Department of Computer Science and Engineering, Velagapudi Ramakrishna Siddhartha Engineering College
Vijayawada, India
Email: madhuri.chandra1209@gmail.com

1. INTRODUCTION

Forest fires can lead to significant harm to the environment, including the loss of wildlife habitats and the destruction of timber resources [1]. They also pose a threat to human life and property. Today, the technology for detecting forest fires has significantly improved with the use of satellites, drones, and sophisticated sensors [2]. The use of deep learning methods is one such advancement towards detecting forest fires. It involves training deep neural networks with large amounts of data, allowing the network to automatically learn and improve from experience [3]. Convolutional neural networks (CNNs) are used in deep learning for processing and classifying images [4]. They use multiple layers of interconnected neurons to learn features and patterns in image data. They use a series of pooling layers, convolutional layers, and fully connected layers to extract features from the given input image and classify it into one or more categories. CNNs have two main parts: the feature extraction part and the classification part [5]. The feature extraction part consists of a set of convolutional and pooling layers [6]. The convolutional layers of the CNN model perform feature extraction using convolutions (filters) of different sizes. Pooling layers deal with downsampling the feature maps and reducing the count of parameters in the network [7]. The final feature map is flattened and sent to the classification part of CNN. The classification part consists of a fully connected network that deals with the actual classification of the image. The fully connected network contains several layers of neurons that have a bias and the activation function. An activation function produces an output based on the input feature map values, weights of the edges from the previous layer, and bias. Based on the activation function output, the neurons are fired, and the value is sent to the next layer [8]. The last layer of the CNN is called the output layer, which has as many neurons as the number of output classes possible. Optimizers are used in CNNs to converge to the output faster [9]. They reduce the computational time by only considering the important parameters for classification which is done by observing the loss function.

Each classification model has its own strengths and weaknesses, and relying on a single model can limit its predictive power. However, using an ensemble of models can mitigate individual drawbacks and improve overall accuracy. Ensembling is the process of combining the predictions of multiple machine learning (ML) models to produce a final prediction [10]. The main objective of this research work is to identify forest fires in aerial images based on the presence of smoke or fire using three different CNN models: InceptionV3, VGG-16, and ResNet50. The results of these three models are combined, and a final result is produced. These three models were chosen because Resnet is a deep network with many layers, which is more accurate. Inception is a rather wider, not deeper, network that computes results in less time. VGG is a small network that yields excellent results. Each of these has different features, and hence their combination would lead to better results. The three models are trained using transfer learning [11]. Each of these models is pre-trained on the "ImageNet" dataset, which is a very large dataset that contains up to 100 different classes of images [12]. The fully connected network at the end of these models supports the "ImageNet" dataset classes and weights. The top layers of the model are removed and the initial layers are retained. The initial layers are frozen to retain their weights so that the powerful feature extraction capability of the models can be utilized. The final dense layers are created according to our work and the model is trained.

Some of the related works are discussed below. Each of these research works employed more than one model for recognizing forest fires and combined the results. Xu *et al.* [13] used YOLOv5, EfficientDet, and EfficientNet. YOLOv5 was effective in detecting large-area fires and EfficientDet was implemented to identify the small-ones carefully. Additionally, EfficientNet was utilized to prevent the system from being tricked by fire-like objects such as the sun. In their study, Ghali *et al.* [14] employed an ensemble of two deep learning models, EfficientNet-B5 and DenseNet-201, to classify images as either non-fire or fire. The image is classified using a Sigmoid function. Both models extract features with different complexity levels. Huang *et al.* [15] used an ensemble of Lightweight YOLOX-L and the defogging method to detect forest fires even in cases where fog obstructs the view of the flames in the images. This ensemble approach allowed for improved detection accuracy but did not provide satisfactory results in situations where the fog density is high.

Bahhar *et al.* [16] utilized a binary classification model with a binary cross-entropy loss and an ensembled CNN network of Xception, MobileNetV2, ResNet-50, and DenseNet121 to improve the accuracy of detecting fire and smoke. The results were promising, but the scalability of this approach for large-scale wildfire detection may be limited. Sathishkumar *et al.* [17] utilized transfer learning in two stages: pre-training and fine-tuning. Pre-trained models, namely CGG-16, Inception V3, and Xception were employed to extract features from the dataset. The learning without forgetting (LwF) method was utilized to preserve the knowledge learned during the pre-training stage. But the method is computationally expensive and may require significant computing resources for training. Khan and Khan [18] developed a CNN-based FFireNet model to classify forest fire images. MobileNetV2 was used as the backbone network. Sigmoid and rectified linear unit activation functions were used to add non-linearity and prevent overfitting. However, the model had difficulty classifying smaller-scale fires. Bouguettaya *et al.* [19] reviewed various computer vision techniques, traditional ML and deep learning algorithms such as object detection, semantic segmentation and image classification. However, the study did not examine generative adversarial networks (GANs) and long short-term memory (LSTM) algorithms. Oliver *et al.* [20] used LeNet-5, AlexNet and VGG-16 models for forest fire prediction and classification, which required less data-training than other supervised ML algorithms. However, the training data may not adequately represent the full range of forest fire conditions. Geetha *et al.* [21] explores different image processing techniques and CNNs to detect forest smokes and fires in real-time data. All the above works explored ensemble approaches to detecting forest fires using different models. However, this paper explores the use of three contrasting models that completely differ in their architecture, depth of feature extraction, and time taken to produce results.

2. METHOD

Figure 1 shows an overview of the proposed architecture of the system. The whole process of development is divided into three sections. The first step is data collection and preparation. The next step is to train the individual models and build the ensemble model, followed by the last step, which is testing. Each of these is clearly discussed in the following sections.

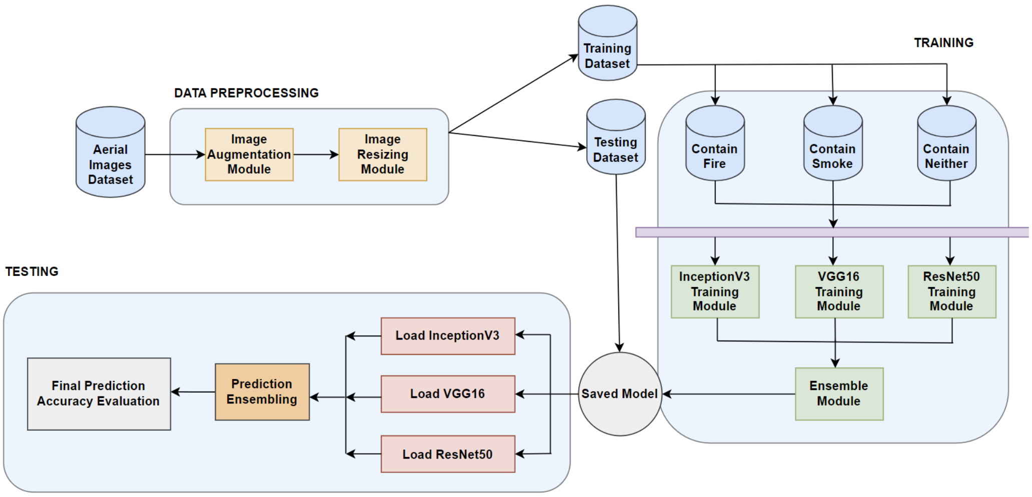

Figure 1. Proposed architecture of the system

2.1. Dataset collection and preparation

A dataset titled "forest-fire" that includes a number of aerial images of the forest fires and smoke was gathered along with all other images from the web sources. The collection includes 2,000 test images, 1,200 aerial images with fire, and 12,000 images with smoke. Training and testing sets were created from the dataset. Following that, the images from both sets were divided into three categories: fire, smoke, and non-fire. Some sample images from the collection are shown here in Figure 2.

Figure 2. Some sample images from 'forest-fire' dataset

To avoid overtraining the model, the dataset is divided in such a way that the training set has 900 images per each class and the testing set has 188 images per each class. To help the classifier identify the class that each training image belongs to, annotation of the images is done [22]. All the training images were resized to dimensions [150x150x3] to suit the models. Image augmentation was performed to increase the size of training dataset and the model's performance. Augmentation is the process of making small changes to the existing dataset images [23]. In this way, the dataset size increases and the model will be able to handle the test images taken from different angles, with different lighting. The three main augmentation techniques performed are re-scaling, shearing, and zooming.

2.2. Training InceptionV3 model

InceptionV3 uses "Inception modules" that enable the network to capture features at different scales [24]. A module in Inception contains multiple convolutional filters with differing kernel sizes that are applied to the same input. This will allow the module to capture the features at different scales and with different levels of detail. Inception modules also include 1x1 convolutional layers that reduce the dimensionality of the input, which helps to reduce computational costs and overfitting. InceptionV3 also uses batch normalization to improve training speed and reduce overfitting. This technique normalizes the input to each layer so that the mean and variance of the input are close to 0 and 1, respectively. This makes the training process more stable and helps prevent the network from getting stuck in local minima. The network also includes auxiliary classifiers to predict the output class at intermediate stages. By providing additional feedback during training, the auxiliary classifiers help the network learn more quickly and reduce overfitting.

This work uses the model by training it on forest fire image data using the transfer learning technique [25]. Half of the hidden layers were frozen, and the other half were trained on our work. The final dense layers of the pre-trained model are not considered. New dense layers are created, with the output layer containing only three neurons for classifying the image into one of the three classes required for our problem: fire, non-fire, and smoke. The rectified linear activation function (ReLU) is used for all the layers except for the output layer to maintain non-linearity in the entire network [26]. The softmax function is utilized in the final layer to produce output as probabilities for each class [27]. The pooling technique used is average pooling, and the image size required for this model is 150x150x3. The optimizer used in this model is a stochastic gradient descent (SGD) optimizer, and the loss considered is categorical cross-entropy loss [28]. Figure 3 shows how the accuracy of the model has improved throughout the training. The x-axis shows the number of epochs in which are trained in the model and the y-axis shows the accuracy. We used 10 epochs to train all the three models.

2.3. Training VGG-16 model

VGG-16, a CNN architecture, stands for visual geometry group. The VGG-16 architecture has 16 layers, comprising 3 fully linked layers and 13 convolutional layers [29]. The subsequent convolutional layers include 128, 256, and 512 filters, but the first two convolutional layers have only 64 filters each. All of the convolutional layers use a 3x3 filter size, and the stride is fixed at 1 pixel. After the convolutional layers, there are three fully connected layers with 4096 units each. The result of the final fully connected layer is passed through a softmax activation function, which produces a probability distribution over the 1,000 classes in the ImageNet dataset. By using small 3x3 filters throughout the network, the architecture is able to learn a rich set of features without using more complex techniques such as pooling or in-network down-sampling. VGG-16 uses batch normalization which normalizes the input to each layer so that the mean and variance of the input are close to 0 and 1, respectively. This helps stabilize the training process and reduce overfitting. The VGG-16 model is trained similar to the InceptionV3 model using transfer learning, except that all the hidden layers are frozen in the VGG-16 model. The optimizer used in VGG-16 is the adaptive moment estimation (Adam) optimizer. The image size and the activation functions used are the same for all three models. Figure 4 shows the increase in accuracy of VGG-16 model while training.

2.4. Training ResNet50 model

ResNet50 uses residual connections, allowing the network to learn more easily by enabling it to skip over certain layers during training. The residual connections are added after every two convolutional layers, allowing the network to learn more complex features without increasing the risk of overfitting [30]. The final layer of the network uses a softmax activation function to produce a probability distribution over the classes in the dataset. Overall, ResNet50 is a highly effective and widely used neural network architecture for image classification tasks. Due to the more number of layers in the model, it suffers from the vanishing gradient problem when the output is calculated using activation functions. Skip connections are used to avoid this problem, where the output of one layer is passed to another layer while skipping some layers in between.

The ResNet50 model is also trained similarly to the networks trained previously using transfer learning. Due to the skip connections problem in Resnet that can affect accuracy, all the hidden layers are trained in this case to apply that to our problem. The optimizer used in ResNet is also an SGD optimizer. The image size, activation functions used are similar to those of the other networks. Figure 5 demonstrates the progressive enhancement in accuracy achieved by the ResNet model throughout the training process.

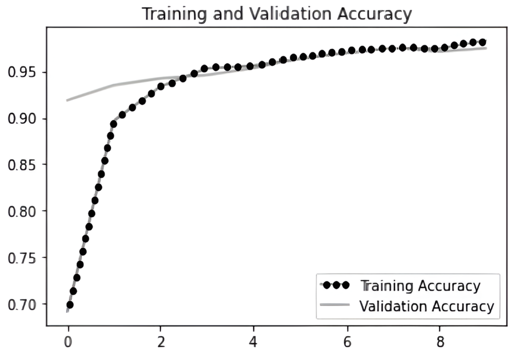

Figure 3. Graph showing the improvement in accuracy of InceptionV3 model during training

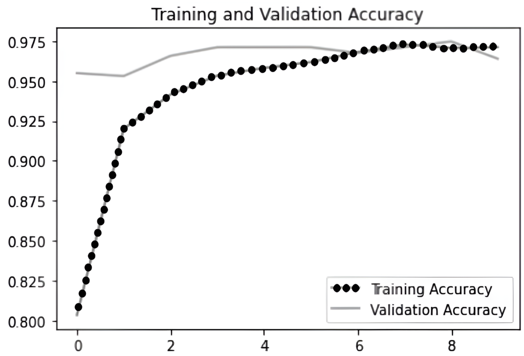

Figure 4. Graph showing the improvement in accuracy of VGG-16 model during training

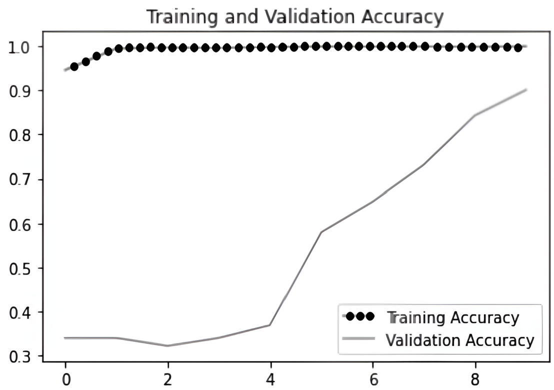

Figure 5. Graph showing the improvement in accuracy of ResNet model during training

2.5. Developing the ensemble model

Each of the three trained classification models performs classification in the following way: they take an aerial image of a forest as input and produce a class index for the predicted class as output, i.e., 0 for fire, 1 for non-fire, and 2 for smoke. The first step is pre-processing the input image. The given input image is resized to 150x150x3. The image is re-scaled to make the pixel ranges of [0,255] fall between [0,1] for normalization. The next step is to extract the features of the image. Convolutions of different sizes (of size NxN) are applied on the input image (of size MxM) to obtain different feature maps (of size M-N+1 x M-N+1) at all the convolutional layers. Average pooling is performed to minimize the high dimensional feature map at pooling layers. This is done as per (1):

$$Y_{i,j,k} = \frac{1}{m * n} \sum_{p=0}^{m-1} \sum_{q=0}^{n-1} X_{i+p,j+q,k} \tag{1}$$

where $Y_{i,j,k}$ is the output at the position i,j,k in the output feature vector, $X_{i+p,j+q,k}$ is the input at the position i+p,j+q,k in the input feature vector, m and n are the height and width of the pooling window. ReLu activation function is used to maintain non-linearity. In (2) shows the ReLu function.

$$f(x) = max(x, 0) \tag{2}$$

The above feature extraction steps are repeated until the final multidimensional feature map is obtained. The obatined feature map of dimensions (HxWxD) is flattened into a single dimensional feature vector of size H*W*D to pass it on to the classification layers. The next step is used to classify the image based on the feature map obtained. Based on the input feature values, edge weights, bias of the neurons and the ReLu activation function, the input is propogated forward and classified. Adam optimizer (for VGG-16) or SGD optimizer (for ResNet50 and InceptionV3) are used that use categorical cross entropy loss to converge faster to the result. Softmax activation function is used to transform the raw outputs of the model into probabilities for each class. In (3) shows the softmax activation function.

$$f_i(\vec{a}) = \frac{e^{a_i}}{\sum_k e^{a_k}} \tag{3}$$

where a is the raw output of the model for that class and output is the probability of image to fall in that class. The class with the highest probability is found and its class index is returned. The next step followed is max ensembling. Each of the model produces a index of the class that it predicted i.e. 0 for fire, 1 for non-fire and 2 for smoke. Among the three predictions, the most predicted class is considered as output.

3. RESULT AND DISCUSSION

The experimental findings that shows the effectiveness of our proposed method and the obtained detection results are presented in this section. The results that are displayed are meant to show how accurate and effective our approach is. True positive (TP), false positive (FP), true negative (TN), and false negative (FN) were calculated from the confusion matrices of the models [31]. TP value increases when a model predicts the positive result and actual result were also positive. FP refers to a case where the model predicted the positive result, but the actual obtained result was negative. TN increases when model predicts a negative result and actual result was also negative. FN refers to a case where the model predicted a negative result, but the actual result was positive. Accuracy is used as a performance metric which is defined as the ratio of number of correct predictions made by the proposed model to the total number of predictions made on it. Accuracy can be mathematically expressed as in (4):

$$Accuracy = \frac{(TP + TN)}{(TP + FP + TN + FN)} \tag{4}$$

The precision metric is used to measure the ratio of genuine positives among all the model's positive predictions. It can be mathematically expressed as in (5):

$$Precision = \frac{TP}{(TP + FP)} \tag{5}$$

The percentage of TPs obtained among all positive cases that the model has successfully identified is measured by recall. It can be expressed mathematically as in (6):

$$Recall = \frac{TP}{(TP + FN)} \tag{6}$$

An evaluation of a model's performance in binary classification tasks is done by calculating its F1-score, that was derived from the balanced average of recall and precision. The F1-score can have values between 0 and 1, where 1 is the greatest possible score. It can be expressed mathematically as in (7):

$$\text{F1-score} = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{7}$$

Table 1 shows the quantitative results of classification for the VGG-16 model, which includes the precision, recall, and F1-score while Tables 2 and 3 provide the same information for ResNet50 and InceptionV3 models respectively. VGG-16 model achieves an accuracy rate of 95.1% in total while ResNet50 achieves 93.3% and InceptionV3 achieves 90%. The results show that VGG-16 showed greater performance than the remaining models.

Table 1. Numerical results of VGG-16 model

	Precision	Recall	F1-score
Fire	0.96	0.94	0.95
Non-fire	0.93	0.96	0.94
Smoke	0.99	1.00	1.00

Table 2. Numerical results of ResNet50 model

	Precision	Recall	F1-score
Fire	0.99	0.72	0.84
Non-fire	0.80	0.98	0.88
Smoke	0.95	1.00	0.97

The combined model showed an overall accuracy of 98.5%. The evaluation metrics for the combined model are presented in Table 4. The combined model is tested and shown for a few sample images, and the results are shown in Figure 6. The proposed work was compared with the other related works in terms of accuracy. The task performed, the dataset used, and the accuracies obtained are listed in Table 5. It can be stated that the combination of models employed in this approach showed improvement in classification accuracy.

Table 3. Numerical results of InceptionV3 model

	Precision	Recall	F1-score
Fire	0.97	0.83	0.89
Non-fire	0.87	0.98	0.92
Smoke	0.97	0.99	0.98

Table 4. Numerical results of the combined model

	Precision	Recall	F1-score
Fire	0.98	0.90	0.94
Non-fire	0.91	0.98	0.94
Smoke	0.99	1.00	1.00

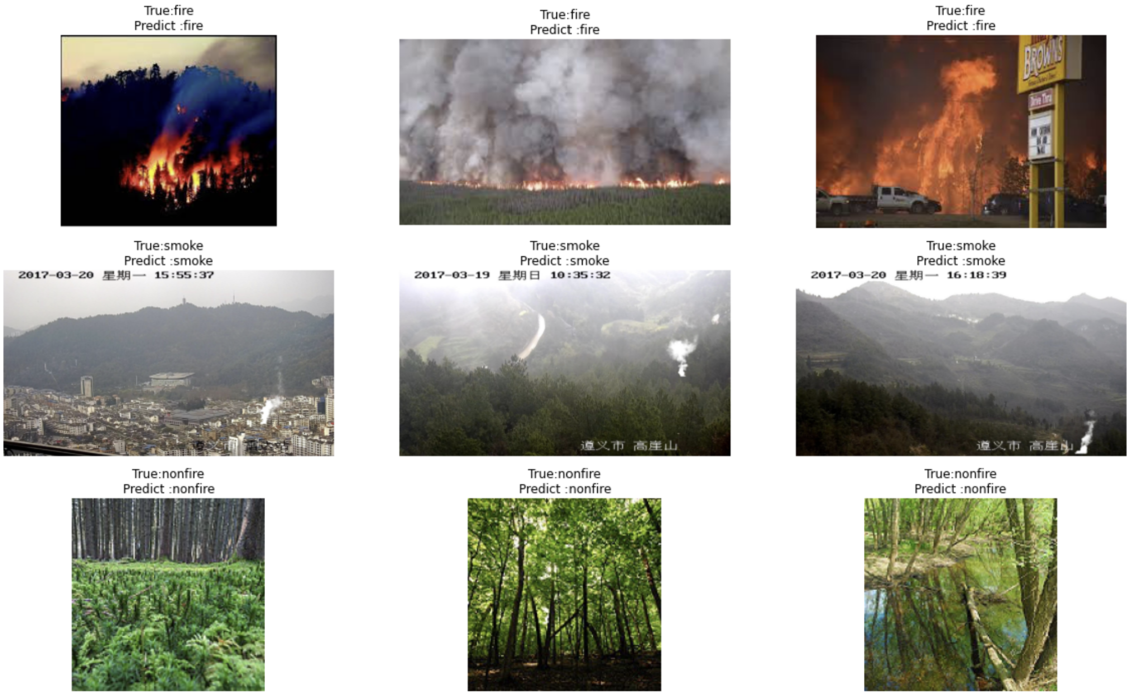

Figure 6. Image showing some predictions of the ensemble model on test images

Table 5. Comparison of previous related implementations with the proposed work

Related work	Task	Dataset	Accuracy (%)
Xu *et al.* [13]	Forest fire classification and detection	Bow fire, FD-dataset, forestry images, VisiFire	90.12
Ghali *et al.* [14]	Forest fire classification, detection, and segmentation	FLAME	85.12
Huang *et al.* [15]	Forest fire classification, detection, and smoke detection	Own dataset	Fire: 84 Smoke: 89.62
Bahhar *et al.* [16]	Forest fire classification, detection, and smoke detection	FLAME	93.15
Sathishkumar *et al.* [17]	Forest fire classification, fire, and smoke detection	Dataset collected from MODIS, VIIRS, Sentinel-2 and Landsat-8	91.41
Khan and Khan [18]	Forest fire classification and detection	Own dataset	98.42
Oliver *et al.* [20]	Detection of forest fires using CNN	Own dataset	94.3
Our work	Forest fire and smoke classification	Forest fire	95.8

4. CONCLUSION

The purpose of this work was to develop and evaluate the efficiency of an ensemble model for classification of the forest fires based on the existence of fire or smoke features. Transfer learning is used to retrain three separate image classification algorithms, VGG-16, ResNet50, and InceptionV3, which are pre-trained on "ImageNet" dataset. The individual predictions of all the three classes are considered, and the most predicted class is considered the final predicted class. The final ensemble model produced an overall accuracy of 95.8% which exceeded the accuracies of the individual classifiers 95.1%, 90.0%, and 93.3% for VGG-16, ResNet50, and InceptionV3 respectively. The work indicated that the use of transfer learning techniques on the strong architectures of the three different models and combining their outputs, along with the use of Adam and SGD optimizers, led to higher accuracy results. The future work deals with considering other strong image classifiers and including their predictions in deciding the final output.

REFERENCES

[1] M. D. Flannigan, B. D. Amiro, K. A. Logan, B. J. Stocks, and B. M. Wotton, "Forest fires and climate change in the 21st century," *Mitigation and Adaptation Strategies for Global Change*, vol. 11, no. 4, pp. 847-859, Jul. 2006, doi: 10.1007/s11027-005-9020-7.

[2] A. A. A. Alkhatib, "A review on forest fire detection techniques," *International Journal of Distributed Sensor Networks*, vol. 10, no. 3, p. 597368, Mar. 2014, doi: 10.1155/2014/597368.

[3] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015, doi: 10.1038/nature14539.

[4] D. Lu and Q. Weng, "A survey of image classification methods and techniques for improving classification performance," *International Journal of Remote Sensing*, vol. 28, no. 5, pp. 823–870, Mar. 2007, doi: 10.1080/01431160600746456.

[5] P. Kim, "Convolutional neural network," in *MATLAB Deep Learning*, Berkeley, CA: Apress, 2017, pp. 121–147, doi: 10.1007/978-1-4842-2845-6_6.

[6] M. D. Levine, "Feature extraction: a survey," *Proceedings of the IEEE*, vol. 57, no. 8, pp. 1391–1407, Aug. 1969, doi: 10.1109/PROC.1969.7277.

[7] M. Sun, Z. Song, X. Jiang, J. Pan, and Y. Pang, "Learning pooling for convolutional neural network," *Neurocomputing*, vol. 224, pp. 96–104, Feb. 2017, doi: 10.1016/j.neucom.2016.10.049.

[8] R. Poojary and A. Pai, "Comparative study of model optimization techniques in fine-tuned CNN models," in *2019 International Conference on Electrical and Computing Technologies and Applications (ICECTA)*, IEEE, Nov. 2019, pp. 1–4, doi: 10.1109/ICECTA48151.2019.8959681.

[9] A. D. Rasamoelina, F. Adjailia, and P. Sincak, "A review of activation function for artificial neural network," in *2020 IEEE 18th World Symposium on Applied Machine Intelligence and Informatics (SAMI)*, IEEE, Jan. 2020, pp. 281–286, doi: 10.1109/SAMI48414.2020.9108717.

[10] T. Hastie, R. Tibshirani, and J. Friedman, "Ensemble learning," in *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, New York, NY: Springer, 2009, pp. 605–624, doi: 10.1007/978-0-387-84858-7_16.

[11] K. Weiss, T. M. Khoshgoftaar, and D. Wang, "A survey of transfer learning," *Journal of Big Data*, vol. 3, no. 1, pp. 1–40, May 2016, doi: 10.1186/s40537-016-0043-6.

[12] J. Deng, W. Dong, R. Socher, L.-J. Li, Kai Li, and Li Fei-Fei, "ImageNet: a large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Jun. 2009, pp. 248–255, doi: 10.1109/CVPR.2009.5206848.

[13] R. Xu, H. Lin, K. Lu, L. Cao, and Y. Liu, "A forest fire detection system based on ensemble learning," *Forests*, vol. 12, no. 2, p. 217, Feb. 2021, doi: 10.3390/f12020217.

[14] R. Ghali, M. A. Akhloufi, and W. S. Mseddi, "Deep learning and transformer approaches for UAV-based wildfire detection and segmentation," *Sensors*, vol. 22, no. 5, p. 1977, Mar. 2022, doi: 10.3390/s22051977.

[15] J. Huang, Z. He, Y. Guan, and H. Zhang, "Real-time forest fire detection by ensemble lightweight YOLOX-L and defogging method," *Sensors*, vol. 23, no. 4, p. 1894, Feb. 2023, doi: 10.3390/s23041894.

[16] C. Bahhar *et al.*, "Wildfire and smoke detection using staged YOLO model and ensemble CNN," *Electronics*, vol. 12, no. 1, p. 228, Jan. 2023, doi: 10.3390/electronics12010228.

[17] V. E. Sathishkumar, J. Cho, M. Subramanian, and O. S. Naren, "Forest fire and smoke detection using deep learning-based learning without forgetting," *Fire Ecology*, vol. 19, no. 1, pp. 1–17, Feb. 2023, doi: 10.1186/s42408-022-00165-0.

[18] S. Khan and A. Khan, "FFireNet: deep learning based forest fire classification and detection in smart cities," *Symmetry*, vol. 14, no. 10, p. 2155, Oct. 2022, doi: 10.3390/sym14102155.

[19] A. Bouguettaya, H. Zarzour, A. M. Taberkit, and A. Kechida, "A review on early wildfire detection from unmanned aerial vehicles using deep learning-based computer vision algorithms," *Signal Processing*, vol. 190, p. 108309, Jan. 2022, doi: 10.1016/j.sigpro.2021.108309.

[20] A. S. Oliver, U. Ashwanthika, and R. Aswitha, "Detection of forest fire using convolutional neural networks," in *2020 7th International Conference on Smart Structures and Systems (ICSSS)*, IEEE, Jul. 2020, pp. 1–6, doi: 10.1109/ICSSS49621.2020.9202200.

[21] S. Geetha, C. S. Abhishek, and C. S. Akshayanat, "Machine vision based fire detection techniques: a survey," *Fire Technology*, vol. 57, no. 2, pp. 591–623, Mar. 2021, doi: 10.1007/s10694-020-01064-z.

[22] A. Hanbury, "A survey of methods for image annotation," *Journal of Visual Languages & Computing*, vol. 19, no. 5, pp. 617–627, Oct. 2008, doi: 10.1016/j.jvlc.2008.01.002.

[23] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, vol. 6, no. 1, p. 60, Dec. 2019, doi: 10.1186/s40537-019-0197-0.

[24] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, Jun. 2016, pp. 2818–2826, doi:

10.1109/CVPR.2016.308.
[25] S. S. Jandhyala, R. R. Jalleda, and D. M. Ravuri, "Forest fire classification and detection in aerial images using Inception-V3 and SSD models," in *2023 International Conference on Intelligent Data Communication Technologies and Internet of Things (IDCIoT)*, Bengaluru, India: IEEE, Jan. 2023, pp. 320–325, doi: 10.1109/IDCIoT56793.2023.10053522.
[26] A. F. Agarap, "Deep learning using rectified linear units (ReLU)," *arXiv*, Mar. 2018, doi: 10.48550/arXiv.1803.08375
[27] A. Khalifa and A. Al-Hamadi, "A survey on loss functions for deep face recognition network," in *2021 IEEE 2nd International Conference on Human-Machine Systems (ICHMS)*, IEEE, Sep. 2021, pp. 1–7, doi: 10.1109/ICHMS53169.2021.9582652.
[28] P. Netrapalli, "Stochastic gradient descent and its variants in machine learning," *Journal of the Indian Institute of Science*, vol. 99, no. 2, pp. 201–213, Jun. 2019, doi: 10.1007/s41745-019-0098-4.
[29] S. Tammina, "Transfer learning using VGG-16 with deep convolutional neural network for classifying images," *International Journal of Scientific and Research Publications (IJSRP)*, vol. 9, no. 10, pp. 143–150, Oct. 2019, doi: 10.29322/IJSRP.9.10.2019.p9420.
[30] B. Koonce, "ResNet 50," in *Convolutional Neural Networks with Swift for Tensorflow*, Berkeley, CA: Apress, 2021, pp. 63–72, doi: 10.1007/978-1-4842-6168-2_6.
[31] M. Hossin and M. N. Sulaiman, "A review on evaluation metrics for data classification evaluations," *International Journal of Data Mining & Knowledge Management Process*, vol. 5, no. 2, pp. 1–11, Mar. 2015, doi: 10.5121/ijdkp.2015.5201.

BIOGRAPHIES OF AUTHORS

Ch Raga Madhuri is an accomplished Assistant Professor at the Department of Computer Science and Engineering at Velagapudi Ramakrishna Siddhartha Engineering College, Vijayawada, having held this position since April 2015. Alongside her teaching responsibilities, she is currently pursuing a Ph.D. degree from JNTUK, Kakinada. With over 8 years of teaching experience, she has gained extensive knowledge and expertise in artificial intelligence, machine learning, data analytics, and IoT fields. She has a distinguished research record, having published 9 papers in many national and international conferences supported by IEEE and Scopus. In addition, she has published research papers in Scopus and UGC Indexed journals. She also holds one patent in her name. She can be contacted at email: madhuri.chandra1209@gmail.com.

Sravya Sri Jandhyala is a student at Velagapudi Ramakrishna Siddhartha Engineering College, Kanuru, Vijayawada, India, pursuing her Bachelor of Technology in Computer Science and Engineering. Her research interests lie in machine learning and artificial intelligence. She published two papers in international conferences supported by IEEE and Scopus. She can be contacted at email: sravyajandhyala2001@gmail.com.

Deepthi Meenakshi Ravuri is a student at Velagapudi Ramakrishna Siddhartha Engineering College, Kanuru, Vijayawada, India, pursuing her Bachelor of Technology in Computer Science and Engineering. Her research interests lie in machine learning and artificial intelligence. She published a paper in an international conference supported by IEEE and Scopus. She can be contacted at email: vkdmeenakshi@gmail.com.

Vunnava Dinesh Babu has been working as the Head of Computer Science and Engineering Department at RV Institute of Technology, Guntur, Andhra Pradesh. He is a seasoned professional with passion for continuous learning to enhance his teaching. He has done B.Tech. from NRI Institute of Technology, M.Tech. in Computer Science from St. Mary's Group of Institutions and Ph.D. from Saveetha Institute of Medical and Technical Science of Chennai. He has a rich 12+ years of experience in Academics, Research, Teaching, Training and Consultancy. Also, he is actively involved in guiding and performing research on various facets of Technologies. As a resolute educationalist, he has published numerous text books on cryptography and network security along with research papers which outline his educational tendencies. He can be contacted at email: dineshbabuvunnava@gmail.com.