IAES International Journal of Artificial Intelligence (IJ-AI)
Vol. 14, No. 3, June 2025, pp. 2083~2095
ISSN: 2252-8938, DOI: 10.11591/ijai.v14.i3.pp2083-2095

A hybrid framework for wild animal classification using fine-tuned DenseNet121 and machine learning classifiers

Dhanushree Vijayendrakumar, Balakrishna Kempegowda
Department of Electronics and Communication Engineering, Maharaja Institute of Technology Mysore, Visvesvaraya Technological University, Belagavi, India

Article Info

Article history:

Received Sep 17, 2024
Revised Feb 4, 2025
Accepted Mar 15, 2025

Keywords:

Classification
Deep learning
Finetuned
Hybrid model
Machine learning
Pre-trained
Transfer learning

ABSTRACT

Over the past few decades, wildlife monitoring has become an active research area. Various topics like animal-vehicle collision, human-animal conflict, animal poaching, population demography, and tracking of animal behaviour come under wildlife monitoring. Different methods have been used for wild animal monitoring, out of which machine learning (ML) and deep learning (DL) are widely used for automatic detection and classification of species from digital images. Both ML and DL have their advantages and disadvantages. A hybrid model has been proposed by integrating the advantage of DL and ML to detect and classify animals automatically. Publicly available datasets of five wild animals are used to train the model, and for testing the model, a dataset is created by capturing the images with the help of a camera and mobile device in different locations and under various environmental conditions. Two approaches are proposed using a hybrid model: a pre-trained dense convolution neural network 121 (DenseNet121) model is used in the first approach, and a finetuned DenseNet121 model is used in the second approach for feature extraction. Extracted features from the pre-trained and finetuned DenseNet121 model are fed into ML classifiers such as extreme gradient boosting (XGBoost), random forest (RF), and naïve Bayes (NB) for classification. When the performance was analysed, the second approach performed better than the first.

This is an open access article under the CC BY-SA license.

Corresponding Author:

Balakrishna Kempegowda
Department of Electronics and Communication Engineering, Maharaja Institute of Technology Mysore
Visvesvaraya Technological University
Belagavi 590018, Karnataka, India
Email: balakrishnak_ece@mitmysore.in

1. INTRODUCTION

The human population is increasing significantly yearly, resulting in faster natural resource consumption. In addition to the rapid growth of population, urbanization, pollution, hunting, poaching, and climate change threaten the survival and habitats of wildlife, particularly wild animals. Many national and international organizations like the World Wide Fund for Nature-India (WWF-India), the Wildlife Trust of India (WTI), the International Union for Conservation of Nature (IUCN), the Wild Life Conservation Society (WCS), the World Wildlife Fund (WWF), Conservation International (CI), and Fauna and Flora International, work with the government and law enforcement to maintain balanced biodiversity by protecting the species and its habitat. Earlier, wild animals were monitored with the help of direct observation, data records, and acoustic devices. It was very time-consuming. Nowadays, wild animal detection from camera trap images and videos plays a vital role in monitoring their distribution. Detection of wild animals from digital images is difficult due to complex and ever-changing environments like lighting, weather conditions, and cluttered backgrounds [1]. There are several approaches to monitoring wild animals, among which machine learning (ML) and deep learning (DL) techniques have become popular in animal detection and classification.

In recent years, computer vision systems have developed a lot because they attempt to train the computer to perform actions similar to humans by recognizing patterns. ML, a subset of computer vision, has gained attention in medical diagnosis, agriculture, robotics, object recognition and classification, speech recognition, and stock market trading. In the field of wildlife monitoring, manually identifying the species is a very time-consuming and difficult task. Where with the help of ML algorithms, this task becomes easy so that animals can be monitored efficiently; thus, we can prevent human-animal conflict, human-vehicle collision, animal theft, and protect the agriculture field from wild animals. Conventional ML involves the following steps for detection and classification: preprocessing to improve the image quality, segmentation to break the image into a discrete group of pixels, feature extraction, and classification. Where ML requires separate algorithms for feature extraction and classification. Feature extraction plays a crucial role in improving the accuracy of the ML model. The feature can be binary, categorical, numerical, textual, or ordinal [2]. Different techniques are used to extract the features, such as locally linear embedding (LLE), autoencoders, principal component analysis (PCA), non-negative matrix factorization (NNMF), local binary pattern (LBP), and singular value decomposition (SVD). The extracted features are used for classification. Classification is a supervised ML method, and various algorithms are used for classification. Some of them are support vector machine (SVM), naïve Bayes (NB), extreme gradient boosting (XGBoost), random forest (RF), logistic regression (LR), decision tree (DT), and k-nearest neighbour (KNN).

DL, a subset of ML, gained a lot of attention in the early 2000s for its improvement in the performance of predictive models. DL algorithms can handle almost any type of data, but solving complex problems requires a large amount of data and processing power. Different DL algorithms exist, and convolutional neural networks (CNN) are the most commonly employed for object detection and categorization. Identifying and classifying wild animals from digital images is a challenging task because most of the images contain highly cluttered background environments with inappropriate poses of animals. In addition to this, images include large aerial data, and most of the time, these images are fuzzy. Even with these lot constraints, CNNs can accurately classify and locate wild animals in an image because of their capacity to analyze information directly from raw data and automatically extract the complex features from the previous layer, which are combined and passed to the next layer using convolutional operations. In recent years, pre-trained DL model architectures and transfer learning approaches have played a significant role in object detection, localization, and classification [3]. The most considerable advantage of DL over ML is that it eliminates the need for handcrafted features during the training of the dataset. Even though DL has a lot of advantages, it also has its challenges. First, it is greedy for datasets and requires a relatively considerable amount of data compared to ML to train the model because the number of learning parameters is very high. The second and most crucial part is tuning the hyperparameter because it affects the model's accuracy. Compared to ML models, DL models take considerable time to find the optimal setting [4]. Features significantly impact the performance of both ML and DL models, where feature extraction and selection require domain knowledge in ML. In contrast, in DL, feature extraction is performed automatically by considering DL and ML advantages and disadvantages.

In this research, a hybrid model is proposed by combining the strengths of both DL and ML models to monitor the wild animals because wild animal monitoring is used in a variety of applications such as wild animal intrusion detection to farm fields, industrial areas, residential area, prevent animal pouching, to maintain population demography of wild animal so that it helps to prevent the animal from becoming endangered, and avoid collision between vehicle and wild animal. In the hybrid model, two approaches are used in the first approach after preprocessingfeatures are extracted using the pre-trained DenseNet121 model, and those features are fed to ML classifiers to perform classifications [5]. In the second approach, finetuning is performed on the pre-trained DenseNet121 model, which is used for feature extraction. Those features are fed to ML classifiers to perform classifications. The hybrid approach helps to increase the interpretability of the classifications made by the ML classifiers [6]. In addition to this, the hybrid model performs well in different domains on low-volume and high-dimensional datasets, which can be seen in the literature review section.

2. LITERATURE REVIEW

In digital image classification tasks, better features always result in better categorization. Previously, several techniques were employed for feature extraction, such as PCA, scale invariant feature transform (SIFT), linear discriminant analysis (LDA), histogram of oriented gradients (HOG), and speed robust feature

(SURF). ML provides various supervised classifier algorithms that are used to categorize the data into a predefined category based on the feature. DL and transfer learning has grown in prominence in feature extraction and classification in recent years.

Kotwal *et al.* [7] proposed a model that combines DL and ML to classify microscopic bacterial images. They considered a dataset of 200 images of four bacterial species, with 50 images from each species for experimentation. Two data preprocessing techniques are used: resizing is used to maintain the uniformity of the dataset by resizing all the images in the dataset to 128×128, followed by augmentation techniques like right and left rotation, cropping, flipping, and scaling to increase the dataset to 2000. Here, a LBP, HOG, residual network 50 (ResNet-50), and visual geometry group 16 (VGG-16) are used for feature extraction. Five ML algorithms, such as KNN, NB, DT, SVM, and RF are used for classification. The best accuracy performance was achieved when VGG-16 features with an SVM classifier were used.

Sarizeybek and Isik [8] put forth a hybrid model to classify the ten different cattle species. The custom dataset was used for experimentation, with an average of 350 images of each species. Here, data preprocessing techniques were used to resize the images to 224×224. The seven different augmentation techniques were used for upsampling the data, and the synthetic minority oversampling technique (SMOTE) algorithm was used to solve the multiclass imbalance problem. The VGG-16 model was used for feature extraction, and supervised ML algorithms like RF and XGBoost were used for classification. The model achieved the highest accuracy when VGG-16 and RF combinations were used.

Mavaie *et al.* [9] propose a hybrid model in which features extracted from a deep network are fed into a non-DL method to classify low-volume, high-dimensional data. For experimentation purposes, they made use of six different datasets. A pre-trained DL VGG model was used for feature extraction, and non-DL models like XGBoost and SVM was used for classification. The research's main objective was to identify the layer in the DL model that is best for feature extraction. From experimentation, they noticed that features extracted from higher levels lead to more accurate classification.

Celik and Inik [10] proposed a brain tumor detection and classification model using DL and optimized ML algorithms. For experimentation purposes, 7,023 MRI images from three publicly available datasets, BrH35, SARTAJ, and Figshare, were resized after dataset collection to provide sufficient input size. The authors proposed a hybrid model composed of a CNN model for feature extraction and a hyper-tuned machine ML algorithms such as DT, NB, SVM, and KNN for classification. Hyper-tuning was performed using a Bayesian optimization algorithm.

Al-Badri *et al.* [11] proposed a model to classify rumex obtusifolius is present in the grassland. Two standard datasets are used for experimentation purposes. Dataset 1 consists of 900 images and dataset 2 consists of 677 images. Several image sizes were compared, and it was identified that the proposed model provided the optimal solution when the size was 224×224. To avoid overfitting, augmentation and resizing were performed. The hybrid model consists of three CCN extractors, which serve as the backbone during classification. The extractors are VGG-16, ResNet-50, and Inception-v3. Each variable is passed through three extractors, and the output of the three extractors is fed to the ensemble model. The ensemble model selects the output with high accuracy and a low error rate. The proposed model helps increase classification accuracy by overcoming overlapping, occlusion, and illumination effects common in real-world images.

Natarajan *et al.* [12] proposed a hybrid model to send an alert message to the end user based on the activity of wild animals. About 45 K images of eight wild animals were taken from publicly available datasets such as the wild animal datasets, CDnet, camera trap, and hoofed. All the images were resized to 224×224, and denoising was performed on the rescaled images. The hybrid model is the integration of the bidirectional long short-term memory (Bi-LSTM) framework and VGG-19. The hybrid model achieves good accuracy and helps protect wild animals and humans during conflict by sending an alert message to the end user.

From the several state-of-the-art works, we can see that most of the researchers have used ML, DL, and few attempted for hybrid approach for detection and classification. However, keen observation leads to identifying the gaps, such as the majority of work being done using publicly available datasets, which have fewer constraints due to limiting challenges while dealing with datasets. Based on our literature survey, we have come to know that not a considerable amount of research has been carried out for the detection and classification of wild animals. Even the work carried out so far in the relevant areas has considered workflow using pre-trained DL models being used for feature extraction and most hybrid approaches being used in the medical field, compared to other fields. However, in the proposed method, the hybrid approach is used for wild animal monitoring. Instead of using a publicly available dataset for both training and testing, it is used only to train the model and to test the model-created datasets, which are captured in real-time at different locations used and compare the model accuracy by using both pre-trained DL and fine tunned DL model in terms of accuracy.

3. MATERIALS AND METHOD

The primary goal of the hybrid model is the accurate detection and classification of wild animals from the digital image. Here, a DL model is used for feature extraction, and ML algorithms are used for classification. Figure 1 shows the workflow of the proposed system which involves five steps that are discussed in detail in this section.

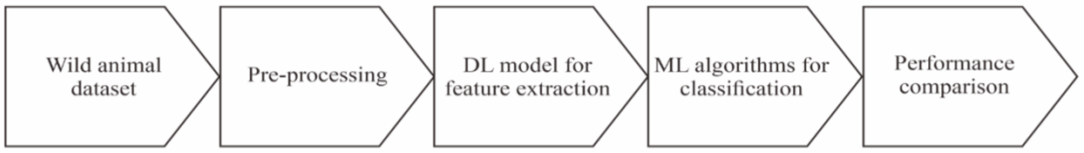

Wild animal dataset
Pre-processing
DL model for feature extraction
ML algorithms for classification
Performance comparison

Figure 1. Workflow of the proposed system

3.1. Dataset

A dataset of five wild animals, such as bears, deer, lions, tigers, and elephants, is used for experimentation. For training purposes, datasets are collected from Images.cv, Kaggle, and Google, and for testing purposes, the data has been captured in real-time using a camera and mobile in different locations like Bannerghatta National Park, Mudumalai National Park, Sakrebyle elephant camp, Sri Chamarajendra Zoological Gardens, Bandipur Safari, Kabini Wildlife Safari, Lion and Tiger Safari, and zoo at Shivamogga. The classification model should deal with real-life scenarios; hence, the dataset for testing is collected in a different location, which includes a range of challenges, like different backgrounds, different postures of animals, and different brightness levels due to various weather conditions.

3.2. Preprocessing

Classification performance depends critically on the dataset's quality, and preprocessing helps to improve the dataset's quality. The proposed model uses four data preprocessing types: resizing, normalization, augmentation, and one hot encoding. The critical step in preprocessing is resizing. The CNN model demands a fixed input size; all the images in the entire dataset should be the same size. Image size affects both training time and accuracy. Here, we resized whole images in the dataset to 300×300. Normalization ensures that every input pixel has a similar data distribution. Normalization transforms the original image pixel into a range between 0 to 1. Thus, it accelerates the training process, improves convergence speed, and enhances performance. Since colored images are considered for the classification problem, each input pixel will range from 0 to 256. A side from 0, the range of pixel values is 255. Normalization was performed on training and testing datasets by dividing all the input pixels by 255. The imbalance in the data set affects the model's performance by causing overfitting. So, it is necessary to increase the original dataset quantity synthetically, which is achieved with the help of augmentation. The two primary categories of data augmentation are color transformation and geometric transformation. The geometric transformation modifies the pixel locations. On the other hand, color transformation changes the pixel values in the image. In the proposed method, geometric transformations like rotation, horizontal flipping, shear range, and zoom range are applied to increase the training dataset.

3.3. Deep learning model

The DL model improves accuracy but requires a larger dataset and training time. Transfer learning is commonly used to address this problem. The strategy of the transfer learning approach is that it transfers the gained weights, or knowledge of the model, which was pre-trained on the large labelled dataset, to the related task. Thus, it provides a shortcut to decrease the computation time through the random initialization of weights [13]. Recent studies have shown that even when the dataset is small, model performance has improved with the help of transfer learning [14]. The proposed method DL model, such as dense convolution neural network 121 (DenseNet121), is used for the feature extraction.

3.3.1. DenseNet121

A DenseNet was proposed by Huang *et al.* [15], to improve performance by overcoming the problem of vanishing gradients. Its accuracy was compared on numerous state-of-the-art image classification datasets, such as CIFAR-10, ImageNet, SVHN, and CIFAR-100. Different variants of DenseNet and DenseNet121 were used for experimentation. DenseNet121 comprises 121 layers with about 8 million parameters. The DenseNet121 architecture is divided into a dense block and a transition layer to facilitate feature concatenation and downsampling. The architecture of DenseNet121 is shown in Figure 2. The input is fed to a 7×7 convolution layer with stride 2, followed by a 3×3 max pool layer for dimensionality reduction with stride 2. The output of pooling layer is fed to a dense block.

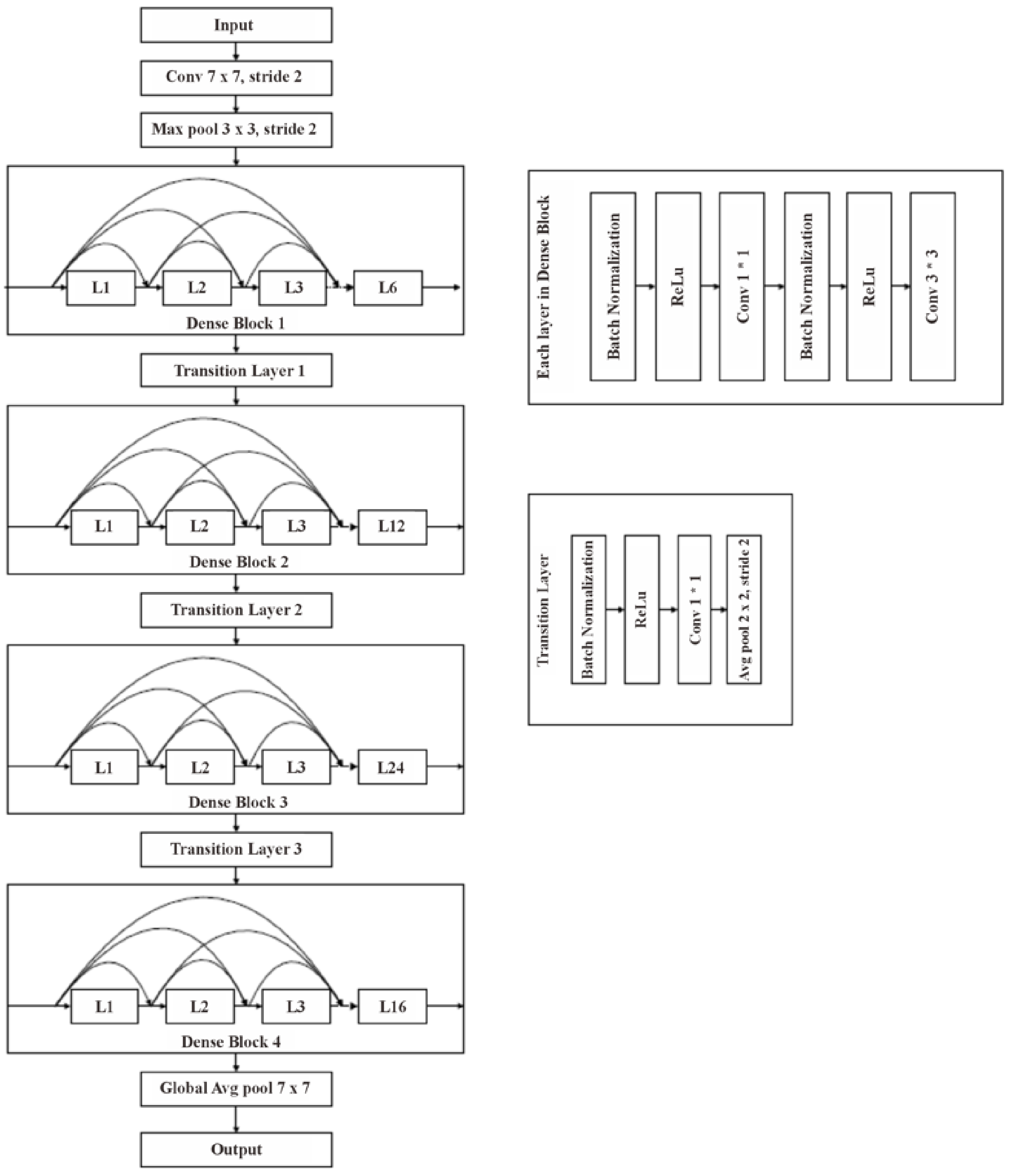


Figure 2. Architecture of DenseNet121

3.3.2. Dense block

DenesNet121 is comprised of four dense blocks with different numbers of layers, such as 6, 12, 24, and 16 layers [16]. The number of filters between dense blocks varies, while the feature map's size remains constant. After computing the nonlinear operations like batch normalization, rectified linear unit (ReLU), and convolution, the resulting feature maps of each layer are sent as inputs to succeeding connected layers. Each dense block's last layer will have the feature map's concatenation. If there are L layers as in (1), the L^{th} layer receives the feature map from all preceding layers.

$$Y_L = H_L[y_0, y_1, y_2, \dots y_{L-1}] \quad (1)$$

Where $y_0, y_1, y_2, \dots y_L$ represents the input, $[y_0, y_1, y_2, \dots y_{L-1}]$ represent the concatenation of input, and H_L is defined as a composite function that applies sequential operations as shown in the content of each denseblock in Figure 2.

Even if each layer produces only k feature maps as output. There may be many inputs for future layers. A 1×1 convolution layer is added as a bottleneck layer before each 3×3 convolution in the dense block to boost computational efficiency and speed.

3.3.3. Transition layer

There are three transition layers in the DenseNet121 architecture to perform downsampling [17]. Each transition layer consists of batch normalization, ReLu activation function, and a 1×1 convolutional layer, followed by a 2×2 average pooling layer. They are placed between dense blocks to reduce the feature map size, a crucial component of convolutional networks.

3.3.4. Fully connected layer

It is the last layer of the DenseNet121 architecture. When the input is a binomial probability distribution, it uses the sigmoid function, or when the input is a multinomial probability distribution, it uses a SoftMax function for classification. SoftMax function is shown in (2).

$$S(\vec{z})_i = \frac{e^{\mathrm{Zi}}}{\sum_{j=1}^{n} e^{\mathrm{Zj}}} \quad (2)$$

The input to the SoftMax function is the output of the last hidden layer, a vector of a real number. e^{Zi} represents the standard exponential function of the input vector, e^{Zj} represents the standard exponential function of the output vector, n represents the number of classes in a multiclass classification problem. The exponential function in the numerator acts as a nonlinear function, which is normalized by dividing it by the sum of the exponential function in the denominator. The output of the SoftMax function is in terms of probability.

3.4. Machine learning classification algorithms

ML algorithms are used for different purposes, like regression, classification, and prediction. The algorithms are chosen based on the type of task. Since we are using ML for classification purposes, this section discusses the classifier algorithms used in experimentation.

3.4.1. Random forest

RF is a supervised ML algorithm based on ensemble learning used in regression and classification problems. A RF comprises multiple DT to solve the problem of overfitting, which usually occurs in a single DT. RF combines randomized feature selection and bagging techniques to build an extensive collection of de-correlated DT. Let M and k represent the training dataset's data instances and features. Each DT in the ensemble is trained on a new dataset built by randomly sampling m instances and randomly selecting k'<k features [18]. Each node will be tested during the tree's growth to see if it is pure or impure for further splitting. Pure nodes will not be divided further, while impure nodes will be divided further [19]. Entropy and Gini index are two methods used to check the purity of the node, which is shown in (3) and (4).

$$E = \sum_{i=1}^{N} -p_i log_2(p_i) \quad (3)$$

$$G = 1 - \sum_{i=1}^{N} (p_i)^2 \quad (4)$$

Where N represents the number of classes, $p_i$represents the relative frequency of the class.

Usually, the Gini index is preferred over entropy since entropy uses a logarithmic function for calculation. Which leads to more computation time. During inference, the final prediction made by the RF depends on the majority of ensemble votes cast by each DT for the selected class.

3.4.2. Naïve Bayes

The NB classifier is another supervised ML algorithm based on the Bayes theorem, mainly used for classification. It has the name naïve because this classifier assumes conditional independence between all input features in the class [20]. Three primary methods are employed under NB: Bernoulli, Gaussian, and multinomial. Multinomial NB is used if the input feature vectors are discrete. Gaussian NB are used if the input feature vectors are continuous. If the input future vector is binary, then Bernoulli NB is used. The Bayes theorem is represented in (5).

$$P(M|N) = \frac{P(N|M)*P(M)}{P(N)} \tag{5}$$

Where P(M|N) is the probability of event M given that the probability of the event N has already occurred, P(N|M) is the likelihood probability, P(M) is posterior probability, and P(N) is marginal probability.

Using the concept of Bayes theorem, the NB classifier is shown in (6).

$$P(J|k_1, k_2 \dots k_n) = \frac{P(J)\prod_{i=1}^{n} P(k_i|J)}{P(k_1)P(k_2)\dots P(k_n)} \tag{6}$$

Where J is the class variable, and $k_1, k_2 \dots k_n$ is a dependent feature vector. Since the denominator remains constant for every record, in (6) can be can be represented by (7). In (8), Argmax is used to find the highest probability class.

$$P(J|k_1, k \dots . k_n)\alpha[\, P(J) \prod_{i=1}^{n} P(k_i|J) \tag{7}$$

$$\hat{J} = argmax[P(J) \prod_{i=1}^{n} P(k_i|J)] \tag{8}$$

3.4.3. Extreme gradient boosting

As its name implies, it is based on a gradient-boosting architecture that addresses regression and classification problems. It uses DT as base learners and leverages regularization approaches to improve model generalization [21]. In gradient boosting, weights are adjusted iteratively to minimize the loss by using adaptive modeling, in which a DT is added one at a time. The output of a new tree is added to that of previous trees until the loss is reduced to a threshold or defined limit of trees. In XGBoost, the root node of the DT is split into only two leaves, regardless of the number of categories. Before splitting the leaf node further, each node's similarity score and gain are calculated. In (9) represents the similarity score where λ is a hyperparameter, and P_r is probability obtained by dividing the possible outcome by a number of classes. Gain is calculated using (10).

$$sc = \frac{\sum(residuals)^2}{\sum \Pr(1-pr)^2+\lambda} \tag{9}$$

$$G = sc(Left\ child\ node) + sc(rigt\ child\ node) - sc(root\ node) \tag{10}$$

XGBoost uses a technique called tree pruning to avoid overfitting problems. This technique involves chopping down branches of the tree that don't significantly impact the model's performance. Tree pruning is calculated by using the cover value [22]. The cover value gives the minimum acceptable gain of the node; The cover value is represented in (11).

$$cv = \sum \Pr(Pr - 1) \tag{11}$$

If the node has more gain than the cover value, then that node will be retained; otherwise, the node will be pruned.

4. PROPOSED METHODOLOGY

The proposed hybrid model for multiclass classification is shown in Figure 3. It is divided into five steps. In the first step, a dataset of five animals, like bears, elephants, deer, tigers, and lions, is collected from the publicly available dataset for training, and a test dataset is collected manually in different locations with the help of a DSLR camera and mobile. In the second step, preprocessing operations like resizing, normalization, augmentation, and one-hot encoding are applied to the training dataset. For the test dataset, all the preprocessing that is mentioned above is applied except for augmentation. In the third step, feature extraction is performed using a pre-trained DenseNet121 model and finetuned DenseNet121. In the fourth step, extracted features are fed to the ML classifiers such as RF, XGBoost and NB for classification. In the fifth step, the performance metrics are applied to evaluate the proposed model.

To be specific, two approaches are used for feature extraction and classification. In the first approach, a pre-trained DenseNet121 model is used for feature extraction and ML classifiers are used for classification. In the second approach, a finetuned DenseNet121 model is used for feature extraction and ML classifiers are used for classification.

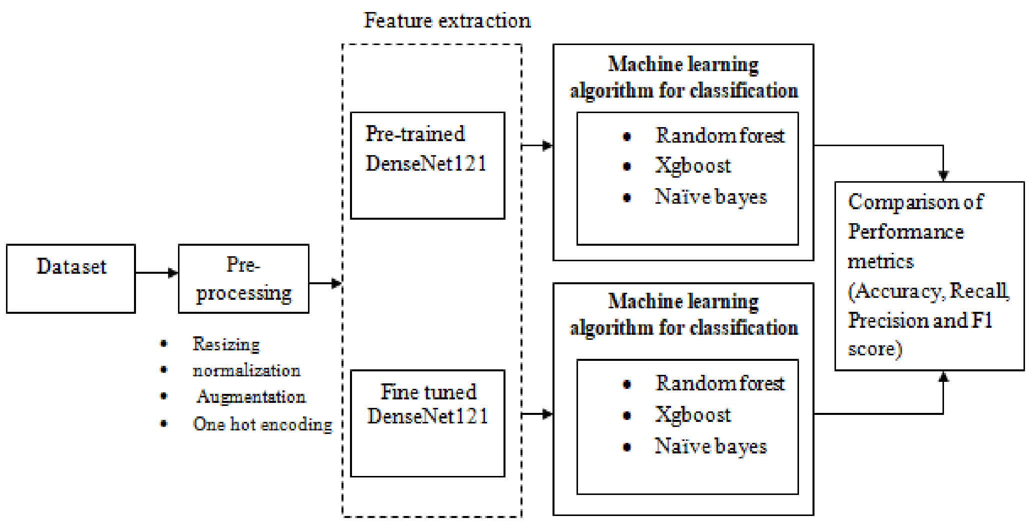


Figure 3 A hybrid model for multiclass classification

4.1. Pre-trained DenseNet121 model with machine learning classifier

This is the first approach. In this approach, after preprocessing the dataset, the hybrid model, which is shown in Figure 4, is used for feature extraction and classification. With the help of the transfer learning approach, the pre-trained DenseNet121, which was trained on the ImageNet dataset, is utilized for feature extraction [23]. Since the pre-trained model is used for feature extraction, it consists of 0 trainable and 7037504 non-trainable parameters. The output of the last layer of pre-trained DenseNet121, which is the global average pooling layer, is fed into ML classifiers such as RF, XGBoost and NB for classification, and the performance of each combination is noted.

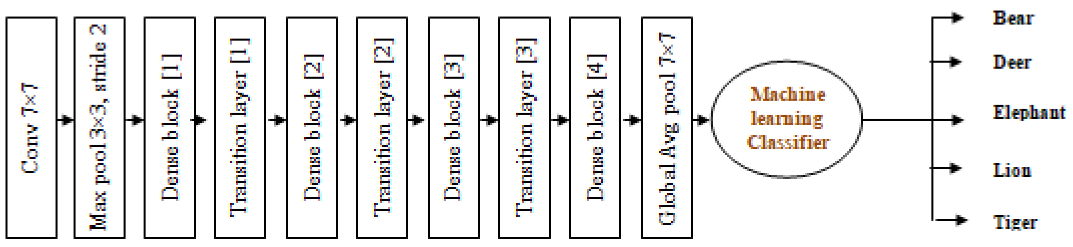


Figure 4. A hybrid model for classification using pre-trained DenseNet121 and ML classifiers

4.2. Finetuned DenseNet121 mode with machine learning classifier

This is the second approach; here, finetuning is performed on top of the pre-trained DenseNet121 model. Figure 5 shows the finetuned version of the model, which is discussed in Figure 4. Since the convolution base extracts the basic features like edges, shapes, and contracts, the convolution base is freed, and finetuning is performed only on the top layer [24]. In the fine-tuned model, a fatten layer is added after the global average pooling layer, followed by a dropout of 20%, a dense layer of size 512 with the ReLu activation function, a dropout of 20% and a Dense layer of 256. During experimentation, the following parameters were held constant: adagrad optimizer with a learning rate of 0.01, batch size of 16, dropout of 0.2, and epochs 10. The feature extracted from the last dense layer is fed as input to the ML classifier for classification purposes, and the performance of each combination is noted.

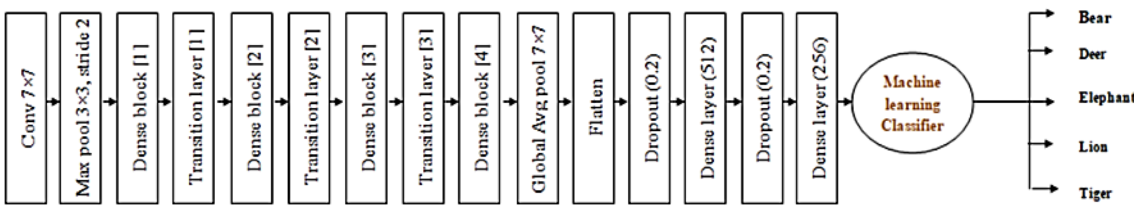


Figure 5. A hybrid model for classification using finetuned DenseNet121 and ML classifiers

4.3. Performance metrics

Various performance matrices are used to evaluate the performance of the ML model. Multiclass classification is evaluated using the confusion or error matrix to assess the proposed model's performance. A confusion matrix is a square matrix that varies in size depending on the number of classes [25]. The diagonal element of the confusion matrix represents the accurate prediction of classes, while non-diagonal elements represent the miss classification. Various performance matrices can be defined using a confusion matrix, out of which precision, F1 score, recall, and accuracy are used. The following notations represent: i) true positive (TP): the number of positive classes in the test data is accurately classified as positive; ii) true negative (TN): the number of negative classes in test data is accurately classified as negative; iii) false positive (FP): the number of negative classes in test data is misclassified as positive; and iv) false negative (FN): the number of positive classes in test data is misclassified as a negative.

– Accuracy: it measures how accurately the model predicts the test data. Equation (12) indicates the number of samples accurately predicted to the total number of samples in the test dataset.

$$Accuracy = \frac{True\ positive+True\ nagative}{Trur\ positive+False\ positive+False\ Nagative+True\ nagative} \tag{12}$$

– Precision: the ratio of the TP to the total number of positive predictions made in the test data. In (13) represents the precision.

$$Precision = \frac{True\ positive}{True\ positive+False\ positive} \tag{13}$$

– Recall: it is a measure of the ability of a prediction model to accurately detect the samples in the test dataset, and it is represented by (14).

$$Recall = \frac{True\ positive}{True\ posive+False\ nagative} \tag{14}$$

– F1-score: precision and recall are used to determine the F1-score as in (15). The model performs better when its F1 score is higher.

$$F1 - score = \frac{2*Recall*Precision}{Recall+Precision} \tag{15}$$

4.4. Experimental setup

The proposed model is executed in Google Colab Pro using NVIDIA graphics processing unit (GPU) on the desktop. Google Colaboratory free online closed based environment which enables the training of ML and DL models on central processing unit (CPU), GPU, and tensor processing unit (TPU). Table 1 shows the list of hardware and software components used for experimentation.

Table 1. Hardware and software details

Hardware	Software
Processor	11th generation Intel core i5 processor
Random access memory	8GB
Operating system	64bit
Windows edition	10-pro
CUDA version	12.2
GPU name	Tesla T4
Languages	Python 3.10.12 version
Library	TensorFlow, seaborn, NumPy, matplotlib, CV2, OS, Scikit-learn

5. RESULTS AND DISCUSSION

The multiclass classification of 5 wild animals was proposed by combining the DL and ML models. The performance of two hybrid approaches compared. In the first approach, feature extraction is performed using a Pre-trained DenseNet121 model, and extracted features are fed into ML classifiers such as XGBoost, RF, and NB. In the second approach, feature extraction is performed using a finetuned DenseNet121 model, and extracted features are fed into ML classifiers such as XGBoost, RF, and NB.

5.1. Analysis of hybrid model (pre-trained DenseNet121 model with ML classifier)

5.1.1. Confusion matrix

The performance of the hybrid model is evaluated using a confusion matrix as shown in Figure 6. The confusion matrix is a square matrix, a plot of accurate and predicted values that displays the true and false predictions made by the model [26]. Each column and row represent the class name: bear, deer, elephant, lion, and tiger. The number of images the model precisely classifies is determined using the diagonal values. Figures 6(a) to 6(c) represent the confusion matrix of the pre-trained DenseNet121model with RF classifier, NB classifier, and XGBoost classifier. The pre-trained DenseNet121 model performs better with the XGBoost classifier than the RF and NB classifiers.

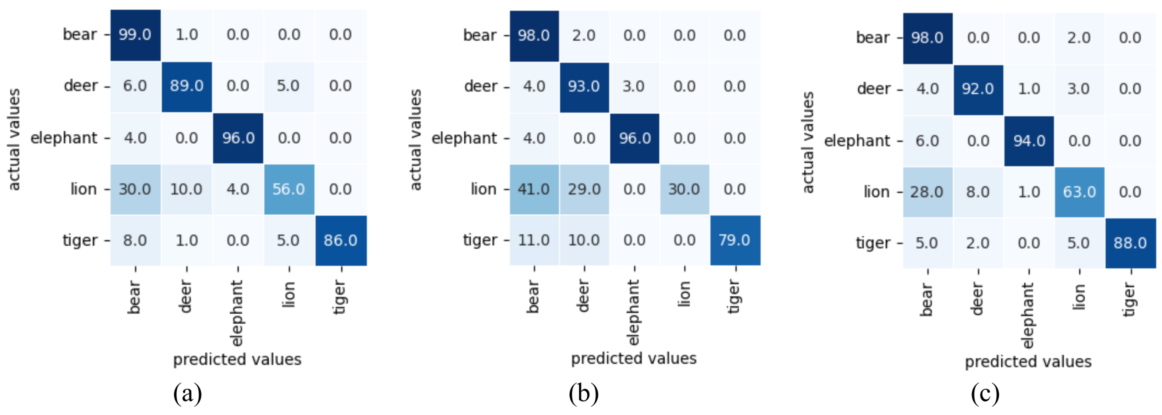


Figure 6. Confusion matrix of pre-trained DenseNet121model with ML classifiers of (a) RF, (b) NB, and (c) XGBoost

5.1.2. Analysis of performance metrics

Accuracy is the most often used approach for measuring a model's performance. In addition to accuracy, the F1 score is used as the primary component of performance evaluation. The F1 score is a combination of precision and recall, with precision indicating how effective the model is at identifying positive class samples from predicted ones. Recall is the number of positive class samples in the dataset that the model correctly identifies. Table 2 shows an average performance comparison of the hybrid mode. It can be observed that better and more stable performance is achieved when the pre-trained DenseNet121 model is used with the XGBoost classifier.

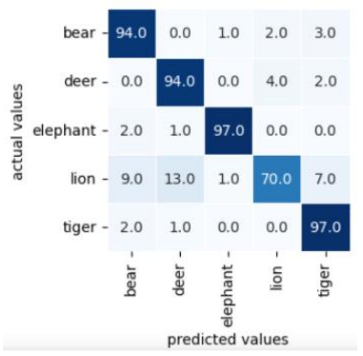
Table 2. Analysis of performance metrics of the hybrid model

Model	Performance matrix			
	Precision	Recall	F1-score	Accuracy
Pre-trained DenseNet121+RF	0.87	0.85	0.85	0.85
Pre-trained DenseNet121+NB	0.86	0.79	0.77	0.79
Pre-trained DenseNet121+XGBoost	0.89	0.87	0.87	0.87

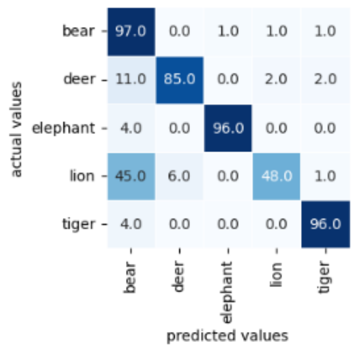
5.2. Analysis of hybrid model (finetuned DenseNet121 model with machine learning classifier)

5.2.1. Confusion matrix

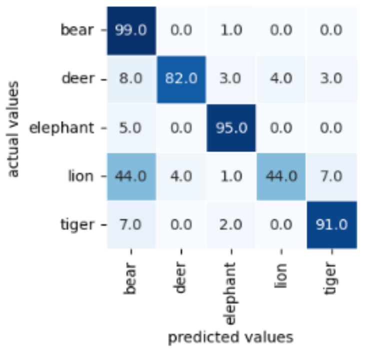
The confusion matrix depicts the prediction model's performance in a tabular format as shown in Figure 7. The diagonal members of the matrix show that the predicted classes match perfectly with the actual classes, whereas the off-diagonal elements represent misclassification. The performance of the classifier is directly proportional to the diagonal elements. The higher the count on the diagonal, the better the classifier's performance. Figures 7(a) to 7(c) show the confusion matrix of the finetuned DenseNet121 model with RF classifier, NB classifier, and XGBoost classifier. The performance of DensNet121 with XGBoost is better when compared to the other two.



(a)



(b)



(c)

Figure 7. Confusion matrix of finetuned DenseNet121 model with ML classifiers of (a) RF, (b) NB, and (c) XGBoost

5.2.2. Analysis of performance matrix (precision, recall, F1-score, and accuracy)

From (13), it can be observed that FP affect precision, and from (14), it can be observed that FN impact recall. To evaluate the performance of a model, it is vital to consider both precision and recall, along with accuracy, where the F1-score is often used to balance precision and recall. Table 3 shows the average performance comparison of the hybrid model with average precision, average recall, average F1-score, and average accuracy. It can be observed that better and more stable performance is achieved when the finetuned DenseNet121 model is used with the RF classifier.

The hybrid model that is proposed is not widely used in wildlife conservation but is most regularly used in other fields. So, authors claim that the work carried to classify the wild animal have not been done so far. Even if it is used in other applications, the pre-trained DL models are used for feature extraction, but in the proposed model, a finetuned DenseNet121 model is used for feature extraction. Most of the work on the hybrid model is done by using the publicly available dataset, but here, the publicly available dataset is used only to train the model, and for testing the model, that dataset is collected in a different location by using a DSLR camera and mobile phone. Since the testing dataset is collected in real-time, it includes various challenges such as different backgrounds, different weather conditions, and different poses of the animals.

Table 3. Analysis of the performance matrix of the hybrid model

Model	Performance matrix			
	Precision	Recall	F1-score	Accuracy
Fine-tuned DenseNet121+RF	91	90	90	91
Fine-tuned DenseNet121+NB	89	84	84	84
Fine-tuned DenseNet121+XGBoost	86	82	82	82

6. CONCLUSION AND FUTUREWORK

A hybrid approach-based classification algorithms are performed on five wild animals like bears, elephants, deer, tigers, and lions. For experimentations, testing datasets were created by our own visiting different locations like Bannerghatta National Park, Mudumalai National Park, Sakrebyle elephant camp, Sri Chamarajendra Zoological Gardens, Bandipur Safari, Kabini Wildlife Safari, Lion and Tiger Safari, and zoo at Shivamogga using DSLR cameras and mobile cameras. Whereas, for training used publicly available datasets stored in repository. A DL model is used for feature extraction, and ML classifiers such as RF, NB, and XGBoost are used for classification. The DeseNet121 model is used to extract the feature. In the first approach, the DenseNet121 model, which was pre-trained on the ImageNet dataset, is used for feature extraction, and the extracted features are fed into ML classifiers for classification. In the second approach, the pre-trained DenseNe121 model is finetuned on the top layer by freezing the convolutional base. It is observed that the XGBoost classifier performs better than the RF and NB classifiers in the pre-trained approach, whereas RF performs better in the finetuned approach. When the finetuned model was used for feature extraction, the classifier was classified more accurately compared to pre-trained model features. In the future, wild animals limited to five classifications can be increased to more numbers in consideration with the local areas and also one can try to work for night vision images. The application once developed can be used for census of wild animals in forest areas and also avoid the conflict between human and animals.

ACKNOWLEDGMENTS
We thank each one of the staff of Maharaja Institute of Technology Mysore and our parents for directly and indirectly supporting in one or other way for our research work.

FUNDING INFORMATION
The authors did not receive support from any organization for the submitted work.

AUTHOR CONTRIBUTIONS STATEMENT
This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Dhanushree Vijayendrakumar	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	
Balakrishna Kempegowda	✓	✓		✓	✓	✓		✓		✓		✓		

C	: **C**onceptualization	I	: **I**nvestigation	Vi	: **Vi**sualization
M	: **M**ethodology	R	: **R**esources	Su	: **Su**pervision
So	: **So**ftware	D	: **D**ata Curation	P	: **P**roject administration
Va	: **Va**lidation	O	: Writing -**O**riginal Draft	Fu	: **Fu**nding acquisition
Fo	: **Fo**rmal analysis	E	: Writing - Review &**E**diting		

CONFLICT OF INTEREST STATEMENT
The authors have no conflicts of interest to declare relevant to this article's content.

DATA AVAILABILITY
The data that support the findings of this study are available on request from the corresponding author, [BK]. The data, which contain information that could compromise the privacy of research participants, are not publicly available due to certain restrictions.

REFERENCES
[1] S. Li, H. Zhang, and F. Xu, “Intelligent detection method for wildlife based on deep learning,” *Sensors*, vol. 23, no. 24, Dec. 2023, doi: 10.3390/s23249669.
[2] K. Liakos, P. Busato, D. Moshou, S. Pearson, and D. Bochtis, “Machine learning in agriculture: a review,” *Sensors*, vol. 18, no. 8, Aug. 2018, doi: 10.3390/s18082674.
[3] S. B. Islam, D. Valles, T. J. Hibbitts, W. A. Ryberg, D. K. Walkup, and M. R. J. Forstner, “Animal species recognition with deep convolutional neural networks from ecological camera trap images,” *Animals*, vol. 13, no. 9, 2023, doi: 10.3390/ani13091526.
[4] K. Balakrishna and V. Dhanushree, “A review on animal detection and classification using computer vision techniques: scope for future enhancement to application,” in *2023 International Conference on Recent Trends in Electronics and Communication (ICRTEC)*, IEEE, pp. 1–6, Feb. 2023, doi: 10.1109/ICRTEC56977.2023.10111888.
[5] K. Balakrishna, “Fusion approach-based horticulture plant diseases identification using image processing,” in *Applications of Advanced Machine Intelligence in Computer Vision and Object Recognition: Emerging Research and Opportunities*, IGI Global Scientific Publishing, pp. 119–132, 2020, doi: 10.4018/978-1-7998-2736-8.ch005.
[6] K. Balakrishna, “WSN, APSIM, and communication model-based irrigation optimization for horticulture crops in real time,” in *Artificial Intelligence and IoT-Based Technologies for Sustainable Farming and Smart Agriculture*, IGI Global Scientific Publishing, pp. 243–254, 2021, doi: 10.4018/978-1-7998-1722-2.ch015.
[7] S. Kotwal, P. Rani, T. Arif, and J. Manhas, “Machine learning and deep learning based hybrid feature extraction and classification model using digital microscopic bacterial images,” *SN Computer Science*, vol. 4, no. 5, 2023, doi: 10.1007/s42979-023-02138-9.
[8] A. T. Sarizeybek and A. H. Isik, “Detection of bovine species on image using machine learning classifiers,” *Gazi University Journal of Science*, vol. 37, no. 1, pp. 137–148, Mar. 2024, doi: 10.35378/gujs.1203685.
[9] P. Mavaie, L. Holder, and M. K. Skinner, “Hybrid deep learning approach to improve classification of low-volume high-dimensional data,” *BMC Bioinformatics*, vol. 24, no. 1, Nov. 2023, doi: 10.1186/s12859-023-05557-w.

[10] M. Celik and O. Inik, "Development of hybrid models based on deep learning and optimized machine learning algorithms for brain tumor multi-classification," *Expert Systems with Applications*, vol. 238, 2024, doi: 10.1016/j.eswa.2023.122159.
[11] A. H. Al-Badri, N. A. Ismail, K. Al-Dulaimi, A. Rehman, I. Abunadi, and S. A. Bahaj, "Hybrid cnn model for classification of rumex obtusifolius in grassland," *IEEE Access*, vol. 10, pp. 90940–90957, 2022, doi: 10.1109/ACCESS.2022.3200603.
[12] B. Natarajan, R. Elakkiya, R. Bhuvaneswari, K. Saleem, D. Chaudhary, and S. H. Samsudeen, "Creating alert messages based on wild animal activity detection using hybrid deep neural networks," *IEEE Access*, vol. 11, pp. 67308–67321, 2023, doi: 10.1109/ACCESS.2023.3289586.
[13] T. S. Qaid, H. Mazaar, M. Y. H. Al-Shamri, M. S. Alqahtani, A. A. Raweh, and W. Alakwaa, "Hybrid deep-learning and machine-learning models for predicting COVID-19," *Computational Intelligence and Neuroscience*, vol. 2021, no. 1, Jan. 2021, doi: 10.1155/2021/9996737.
[14] S. A. Albelwi, "Deep architecture based on DenseNet-121 model for weather image recognition," *International Journal of Advanced Computer Science and Applications*, vol. 13, no. 10, 2022, doi: 10.14569/IJACSA.2022.0131065.
[15] G. Huang, Z. Liu, L. V. der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," *arXiv-Computer Science*, pp. 1-9, Aug. 2016.
[16] I. Marin, S. Mladenović, S. Gotovac, and G. Zaharija, "Deep-feature-based approach to marine debris classification," *Applied Sciences*, vol. 11, no. 12, Jun. 2021, doi: 10.3390/app11125644.
[17] M. Sowmya, M. Balasubramanian, and K. Vaidehi, "Classification of animals using mobilenet with SVM classifier," in *Computational Methods and Data Engineering*, Springer, Singapore, pp. 347–358, 2023, doi: 10.1007/978-981-19-3015-7_25.
[18] A. Raza *et al.*, "A hybrid deep learning-based approach for brain tumor classification," *Electronics*, vol. 11, no. 7, Apr. 2022, doi: 10.3390/electronics11071146.
[19] Y. Wu, Y. He, and Y. Wang, "Multi-class weed recognition using hybrid CNN-SVM classifier," *Sensors*, vol. 23, no. 16, Aug. 2023, doi: 10.3390/s23167153.
[20] K. Hemachandran, S. Tayal, P. M. George, P. Singla, and U. Kose, *Bayesian reasoning and gaussian processes for machine learning applications*. Boca Raton: Chapman and Hall/CRC, 2022, doi: 10.1201/9781003164265.
[21] A. Asselman, M. Khaldi, and S. Aammou, "Enhancing the prediction of student performance based on the machine learning XGBoost algorithm," *Interactive Learning Environments*, vol. 31, no. 6, pp. 3360–3379, 2023, doi: 10.1080/10494820.2021.1928235.
[22] J. Li *et al.*, "Application of XGBoost algorithm in the optimization of pollutant concentration," *Atmospheric Research*, vol. 276, Oct. 2022, doi: 10.1016/j.atmosres.2022.106238.
[23] M. Chhabra and R. Kumar, "A smart healthcare system based on classifier DenseNet 121 model to detect multiple diseases," in *Mobile Radio Communications and 5G Networks*, Springer, Singapore, pp. 297–312, , 2022, doi: 10.1007/978-981-16-7018-3_23.
[24] B. Li, "Facial expression recognition by DenseNet-121," in *Multi-Chaos, Fractal and Multi-Fractional Artificial Intelligence of Different Complex Systems*, Elsevier, pp. 263–276, , 2022, doi: 10.1016/B978-0-323-90032-4.00019-5.
[25] K. Balakrishna, B. R. G. Prasad, N. D. Dhanyashree, V. Balaji, and N. M. Krishna, "IoT based class attendance monitoring system using RFID and GSM," in *2021 IEEE International Conference on Mobile Networks and Wireless Communications (ICMNWC)*, IEEE, pp. 1–5, Dec. 2021, doi: 10.1109/ICMNWC52512.2021.9688482.
[26] J. Li, H. Sun, and J. Li, "Beyond confusion matrix: learning from multiple annotators with awareness of instance features," *Machine Learning*, vol. 112, no. 3, pp. 1053–1075, Mar. 2023, doi: 10.1007/s10994-022-06211-x.

BIOGRAPHIES OF AUTHORS

Mrs. Dhanushree Vijayendrakumar earned her bachelor's degree in electronics and communication from VTU, Belagavi, in 2013. She has obtained her master's degree in M.Tech. (signal processing) from VTU, Belgaum in 2015. Currently, she is a research scholar at Maharaja Institute of Technology Mysore, where she is doing her Ph.D. in electronics and communication. She has attended many workshops, FDPs, and induction programs conducted by various universities. Her areas of interest are artificial intelligence, IoT, and signal processing. She can be contacted at email: dhanushreev@gmail.com.

Dr. Balakrishna Kempegowda received his B.E. and M.Tech. degree from the VTU, Belagavi, Karnataka, India in 2011 and 2013, and his Ph.D. degree at the University of Mysore, Karnataka, India. He is presently working as associate professor in the Department of Electronics and Communication Engineering at Maharaja Institute of Technology Mysore. He has published about 34 research papers in reputed journals and conferences. His research interests his artificial intelligence and internet of things. He has 10 years of research and teaching experience. Presently supervising 4 student towards thier doctral degree in the area of artificial intelligence and internet of things. He can be contacted at email: balakrishnak_ece@mitmysore.in.