IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 5, May 2026
Impact Factor: 8.2

Image–Based Animal Type Classification for Cattle and Buffaloes

Alisha Khan, Aryan Kahar, Bhagyashree Khatal, Vaishnavi Chevale
Department of Computer Application
Assistant Professor, Faculty of Science and Technology
JSPM University, Pune
alisha.khan2005.21@gmail.com, thearyankahar@gmail.com,
bhagyashrikhatal7711@gmail.com, vhc.scos@jspmuni.ac.in

Abstract: *The capability to recognize animals based on pictures is innate in humans yet difficult for computers to perform. The paper describes the way our system, which utilizes Convolutional Neural Networks (CNN) and transfer learning via three models—MobileNetV2, ResNet50, and VGG16—is developed and implemented. For any given image of an animal as input, the algorithm categorizes the picture under one of the categories recognized by the algorithm along with the percentage of certainty regarding the accuracy of the classification. A user- friendly web application is built by Flask and Streamlit without requiring any coding skills from users who only need to provide an image file and get instant output on their screen. Our dataset involves 10–15 species of animals and is preprocessed in terms of resizing, normalization, and data augmentation. In order to achieve a high level of accuracy of recognition, we aimed to achieve no less than 85% on our validation set, a goal surpassed by all three models following optimization.*

Keywords: Animal Recognition, Convolutional Neural Networks, Transfer Learning, MobileNetV2, ResNet50, VGG16, Image Classification, Deep Learning, Computer Vision, Flask, Streamlit, Wildlife Monitoring

I. INTRODUCTION

A. Background and Motivation

When one observes an image of an animal and recognizes it right away, one does not have to work too much on it. In comparison, recognizing an image for a machine entails a much more complicated process where instead of seeing an image, all that the machine sees are numbers and pixel intensities and is trained to learn by itself which patterns relate to which animal. The difference between these two processes is what makes animal recognition such an interesting topic to study. And the importance of this cannot be underestimated since it is tangible and ever increasing. When using camera traps for studies, we usually receive lots and lots of pictures, perhaps tens of thousands per day. Spending so much time analyzing each and every picture would be highly inefficient; therefore, it may take quite some time, ranging from days to even months before conducting a proper analysis. Using automatic classifier allows discarding easily identified pictures, which leaves only difficult ones to analyze manually. Moreover, such application can come in handy not only for wildlife conservationists, but also for students, teachers and anyone interested in animals since it would immediately classify any species seen in the picture. Conventional systems aimed at image analysis included features such as SIFT, HOG and edge detection. However, there was one drawback associated with such systems; while operating great with simple data, they often failed to perform under different circumstances, such as unusual lighting and viewpoint. It is worth noting that the methods used under Deep Learning, especially CNN, made this possible through the extraction of hierarchical visual representations directly from pixels without the need for designing them. This is one of the major factors behind our decision to develop the application using the CNN model because the CNN model has been proven to perform excellently in image

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 5, May 2026 Impact Factor: 8.2

recognition tasks. One of the limitations in relation to the development of this project includes the need for significant computing power as well as a lot of labeled data for training the deep neural network.

B. Evolution of Image Classification

Better methods of identifying images may be attributed to the deep learning methods. There were manual algorithms, including SIFT (Scale-Invariant Feature Transform) and HOG (Histogram of Oriented Gradients); however, they required a lot of expertise and could not function well in cases where there was poor image acquisition. On the contrary, a new era began in 2012 after the development of AlexNet by Krizhevsky et al. AlexNet is considered a success since it was able to win the ImageNet challenge due to its exceptional performance. The secret behind the success of AlexNet lies in the idea that deep convolutional neural networks that are well-trained will always outperform others in large data sets through GPU. Transfer learning turned these advances into something practically accessible. A model like ResNet50, pre-trained on over a million images, has already learned a rich vocabulary of visual features in its convolutional layers—from simple edges in the earliest layers to complex textures and object parts deeper in the network. Reusing those features and only retraining the final classification layers on a new, smaller dataset allows a researcher or student without access to enormous computing resources to still achieve competitive accuracy. In our own experiments this approach worked remarkably well: the jump in accuracy compared to training from scratch was substantial and consistent across all three architectures we tested.

C. Limitations of Existing Systems

Many animal classification models have been developed, although they are generally accompanied by a few key flaws. Computer vision API services offered by major tech corporations can classify animals into broad groups; however, they will likely falter when dealing with rare specimens or poorly lit photos. Wildlife identification applications dedicated to specific platforms like iOS or Android devices may prove difficult to access from within a conventional web browser. Academic articles on the subject usually publish models that lack a deployable user interface. The implementation of a pre-trained model and its actual application are two different stages in the pipeline, and the former does not guarantee success at the latter. Lack of confidence scores for predictions made by AI models appears to be another recurring flaw found in our literature review. Inability to provide information about their uncertainty regarding the classification will inevitably mislead users. This problem can be effectively solved using confidence scores that will accompany the prediction itself. We observed that the vast majority of similar models require massive amounts of training data or retraining altogether when adding new classes to the system.

D. Research Contributions

This paper makes five specific contributions:

1. Design and development of a CNN-based animal recognition system using transfer learning with MobileNetV2, ResNet50, and VGG16 on a curated 10–15 class animal dataset.
2. Implementation of a complete image preprocessing pipeline that includes resizing, normalization, and augmentation to improve model robustness across varied real-world conditions.
3. A user-friendly web application built with Flask/Streamlit that lets any user upload an image and receive a real-time prediction with a confidence score—no coding knowledge needed.
4. A comparative evaluation of the three transfer learning architectures to identify the most suitable option for animal classification in terms of both accuracy and inference speed

II. LITERATURE REVIEW

A. Historical Development of Image Classification and CNNs

The foundations of modern image classification stretch back to Yann LeCun's work on LeNet in the 1990s, a convolutional network originally developed for handwritten digit recognition. The approach was promising, but the

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 5, May 2026
Impact Factor: 8.2

computational resources complex problems simply were not available at the time. The first truly revolutionary work in this field took place during the ImageNet competition in 2012. One of the approaches used deep CNNs that consisted of five convolutional layers, max pooling, rectified linear units, and dropout regularization. As a result, it scored 15.3% in terms of top-5 error rate, and the runner-up showed 26.2%. It seems to be so successful that all concerns about the use of deep learning were laid aside. The next step in the development of CNNs is made by VGGNet in 2014 that managed to prove that adding layers with filters being equal to 3×3 will increase the accuracy again. Further development in CNNs is marked by ResNet (He et al., 2016) that introduced skip connections making gradients propagate deeper, thus enabling the construction of networks having more than 100 layers. Finally, MobileNetV2 (Howard et al., 2017) introduced an alternative solution to the problem and suggested focusing on efficiency through using depth-wise separable convolutions, which we needed for our project.

B. Transfer Learning in Image Classification

Transfer learning is one of those ideas that is easy to explain yet genuinely powerful in practice. A network trained on ImageNet has, in its early and middle layers, learned a rich hierarchy of general visual features—basic edges, color gradients, textures, and eventually more abstract object parts. Because these features are useful far beyond the original classification task, they transfer well to new domains even when the new domain looks quite different from ImageNet.

In practice, transfer learning involves loading a pre- trained model, replacing its final classification layers with a new head tailored to the target task, and training only that head initially. The convolutional base is kept frozen to preserve the learned features. In a second phase, the top layers of the base model can be gradually unfrozen and fine-tuned at a very low learning rate. Rawat and Wang (2017) surveyed the CNN landscape and specifically recommended this approach for domain-specific tasks with limited labelled data—precisely the situation we faced, with only a few hundred to a few thousand images per class. Our experiments confirmed their recommendation: fine-tuned MobileNetV2 and ResNet50 both outperformed a CNN trained from scratch by a significant margin on our dataset.

C. Animal and Wildlife Image Recognition

The field of animal and wildlife image recognition research has seen an exponential rise in the last ten years, fueled predominantly by the necessities of conservation biology. One such field has been that of camera trap image analysis. It has been proven that CNNs can categorize wildlife images with a success rate exceeding 90%, way higher than any algorithm based on manually engineered features. The iNaturalist project is one example of a real application, where there was clear need for image classification among members of the general public. Nonetheless, iNaturalist is a complex system covering thousands of species. For the purpose of identifying around ten to fifteen commonly spotted animals, a fine-tuned transfer learning algorithm will suffice.

D. Research Gaps

Though considerable progress has been made, there are still some shortcomings present. For one, most of the current solutions do not output any confidence scores which would help the user determine whether to trust their prediction. Another issue lies in the fact that many algorithms are only research-oriented objects, meaning that while they can perform quite adequately within a lab notebook, they cannot be accessed and run through a browser without additional setup. However, there is also an intermediate solution that is relatively neglected, that being a sufficiently specialized system, but which is nevertheless extendable without full retraining.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 5, May 2026
Impact Factor: 8.2

TABLE I — SUMMARY OF RELATED WORK IN ANIMAL IMAGE CLASSIFICATION

No.	Authors	Methods/ Tools	Key Outcomes	Limitation s
[1]	Krizhevs ky et al., 2012	AlexNet, Deep CNN on ImageNet	Top-5 accuracy of 84.7% on ImageNet; established that deep CNNs outperform conventiona l CV methods for image	Very high compute requirement s; prone to overfitting on smaller datasets.
[2]	Simonya n & Zisserma n, 2014	VGG16/V GG19 architectu re, deep convoluti onal layers	Showed that increased depth with small 3×3 filters significantl y boosts accuracy. Became a strong baseline for transfer learning.	Very large model size (>500MB); slow inference on resource- constrained devices.
[3]	He et al., 2016	ResNet50 , residual learning, skip connectio ns	Resolved the vanishing gradient problem; achieved 3.57% top- 5 error on ImageNet. ResNets became a standard backbone for vision tasks.	Complex architecture ; demands substantial memory during training.

TABLE II — RESEARCH GAPS AND PROPOSED SYSTEM ALIGNMENT

Gap Area	Description	Our System's Response
Limited Dataset Coverage	Most existing systems train on a narrow subset of animal species, limiting real- world usability	Data set consisting of 1015 frequently met animal classes augmented for balanced learning.
No Confidence Reporting	General- purpose image classifiers rarely communicate prediction confidence to the user.	Confidence percentage provided along with the highest prediction..
Inaccessible Interfaces	Most deep learning models stay as research code with no user- friendly interface for non-technical users.	Flask/Streamlit-based web interface lets any user upload an image and receive results without technical knowledge.
Retraining Complexity	Training CNNs from scratch involves having a vast amount of data and computational power.	Transfer learning from MobileNetV2/ResNet50 delivers high accuracy with modest hardware through fine-tuning.
Scalability	Existing wildlife classification apps are platform- specific and don't easily support new species.	Modular design allows easy expansion by incorporating new animal classes with minimal retraining using a content management system style database module.

III. PROBLEM DEFINITION

A. Problem Statement

The problem statement of this project appears simplistic but it does involve complex issues that need to be overcome in order to obtain satisfactory results. In reality, developing a solution involves overcoming a number of technical and practical problems which were not adequately addressed by previous attempts. The main weaknesses of manual approaches or rule-based solutions are listed below:

• Manual Identification Bottleneck: Wildlife researchers routinely receive thousands of camera trap images daily. Checking each one individually and finding the species becomes extremely tiring, prone to errors, and entirely unfeasible.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 5, May 2026 Impact Factor: 8.2

• Fragility of Conventional Approaches: Rule- based algorithms and traditional feature extraction techniques fail when there are changes in lighting conditions, partial occlusions, or odd angles in the image, all of which occur frequently in field photographs.
• Inaccessibility of Deployment: Most deep learning models are released in academic papers as source code that needs technical knowledge to run, making them inaccessible to educators, field biologists, and students.

B. Research Questions

These four questions guided the creation of the model:
1. What kind of transfer learning architecture, such as MobileNetV2, ResNet50, or VGG16, would provide the best trade-off between classification performance and inference speed for the 10-15 classes of animal photos?
2. What would be the optimal design of the preprocessing phase considering that there might be variations in the appearance of animals in pictures, including background noise, scale differences, and varying lighting conditions, without needing a very large sample size for training?
3. What would be the easiest way to integrate the use of the model as a web application?
4. How to include a feature of displaying confidence scores along with the predictions to enable users to assess whether the information presented is reliable?

IV. PROPOSED SYSTEM

A. System Overview

The design of the suggested Animal Recognition System represents an advanced deep learning system that is both technically efficient and easy to use. The system uses TensorFlow and Keras libraries in the Python programming language for model training and inference; the system frontend is implemented via a web interface with Flask/Streamlit library support. From the user's point of view, using the system involves only simple actions such as opening the web browser and uploading an animal image, followed shortly after by receiving the recognized animal category along with the confidence level. There are two types of users involved in using this system. The end users do not require any technical skills and will interact with the system only via a web browser to receive recognition results. On the other hand, administrator users can interact with the training pipeline and modify existing classes or update the model

B. Core System Modules

User Interface Module

An app that uses Flask (or alternatively, Streamlit) allows the user to upload an image and view the result on a results page. The guiding principle here is one of simplicity, with an interface allowing the user to upload an image, click a button, and see a result. The interface also allows for the display of the top three predictions along with their associated confidence scores.

Image Preprocessing Module

Every image that goes into the network passes through a common preprocessing step first. They undergo resizing to 224 × 224, which is the common input size for MobileNetV2 and ResNet50. The images then get normalized – while the MobileNetV2 uses [−1, 1], ResNet50 performs mean subtraction using ImageNet statistics.During training, augmentation steps such as random horizontal flipping, rotation, zoom, and brightness adjustment are applied to artificially broaden the effective training set and improve generalization to images the model has not explicitly seen.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 5, May 2026
Impact Factor: 8.2

Category	Tool / Technique
Programming Language	Python
Deep Learning	TensorFlow
Deep Learning	Keras
Algorithms	CNN
	MobileNetV2
	ResNet50
	VGG16
	InceptionV3
	EfficientNet
Image Processing	OpenCV
Data Handling	NumPy
Data Handling	Pandas
Web Framework	Flask
Web Framework	Streamlit
Database	SQLite
Visualization	Matplotlib
Deployment	Local Server

Model Module

This is the functional core of the system. We load pre- trained weights from ImageNet for each of the three base architectures and freeze their convolutional layers initially. On top of the frozen base we add a custom classification head: a GlobalAveragePooling2D layer to collapse spatial dimensions into a feature vector, a Dense layer with 256 units, ReLU activation, and dropout (rate = 0.4) for regularization, and a final softmax Dense layer sized to the number of animal classes. In a second training phase, the top 30–40 layers of the base model are unfrozen and fine-tuned at a very low learning rate (1e-5) for an additional ten epochs. Across our experiments, MobileNetV2 trained fastest and showed the best accuracy-per-second-of-inference tradeoff, making it the most practical choice for web deployment. ResNet50 achieved slightly higher accuracy at the cost of longer training and inference times. VGG16 was the largest and slowest of the three.

Prediction Engine Module

Once trained, the model is saved and loaded into the inference engine, which handles all prediction requests from the web application. As soon as the user uploads the image, the preprocessing steps are performed on the image using the same preprocessing pipeline which was used for the training. In the softmax step, probabilities are computed for each class; and the class having the highest probability is selected, along with the second highest and third highest probabilities as well to populate the top three classes shown in the user interface.

Database Module

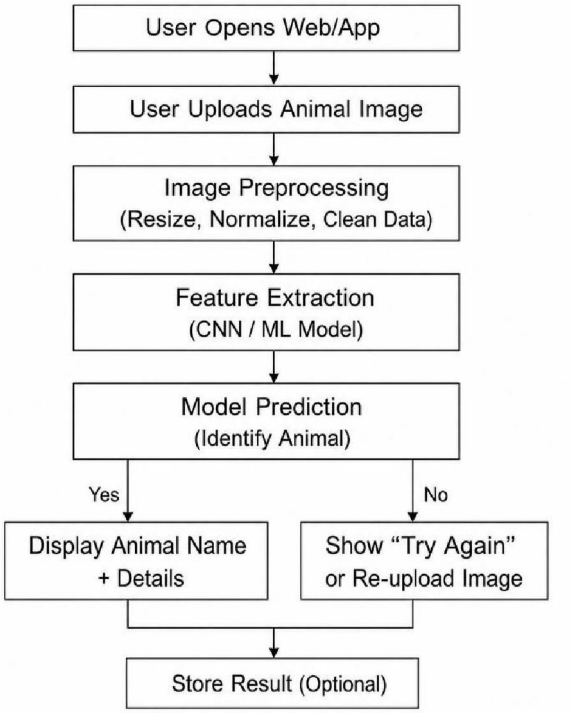
An SQLite database stores all the predictions made by the system in terms of image file names uploaded, the predicted animal category, confidence score, and time stamp. This information allows for analysis by the administrator as to how the system is being used and identification of recurring errors that could be fixed by feeding extra training data on a certain animal category. The database keeps the master list of the supported animal categories

C. System Workflow

The process flow of the system takes place in six stages: (1) The user launches the application in a web browser, where he/she chooses an image of an animal through the upload page. (2) The image is received by the server and sent to the preprocesser module, where it gets resized and normalized according to the input demands of the currently selected machine learning model. (3) The preprocessed image goes into the currently loaded machine learning model for prediction purposes. (4) A set of probabilities across all possible classes of animals is returned from the model; then, the prediction engine takes the most accurate result and computes the accuracy score of that

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 5, May 2026
Impact Factor: 8.2

particular prediction. (5) The outcome is shown on-screen in the form of predicted class title, probability percentage, and optionally the entire topthree prediction list.



V. METHODOLOGY

A. Research Design

This research is done using the approach that relies on the principles of design science research and employs an iterative process. It involves five key stages: problem definition, solution definition, data collection and pre-processing, model development and training, and evaluation. After each stage of evaluation, the process loops back to the earlier stages to correct any deficiencies and enhance performance. In machine learning, such an iterative approach is quite common since the initial model will not be the most effective; lessons learnt from the confusion matrix will often indicate what needs to be done.

B. Dataset Collection and Preprocessing

The data consists of animals belonging to 10-15 classes, such as cats, dogs, horses, elephants, lions, tigers, bears, zebras, giraffes, cows, sheep, deer, monkeys, crocodiles, and eagles. Data was sourced from publicly available resources, such as subsets of ImageNet, Kaggle's animal classification data sets, and the Animals-10 data set. We obtained 800-1,200 images for training for each animal class and 150-200 images for validating. The major difficulty with preparing the data set was the variability of real-world images of animals captured from different angles, distances, and lighting conditions. A certain approach in data augmentation using techniques such as horizontal flipping, random rotation up to plus/minus twenty degrees, zooming by up to twenty percent, brightness variation, and sometimes vertical and horizontal translations, was employed to tackle this problem. These augmentations help the model generalize to image conditions it has not seen during training. Class imbalance was monitored throughout; where imbalance existed, class weights were applied during training to prevent the model from simply learning to favor majority classes.

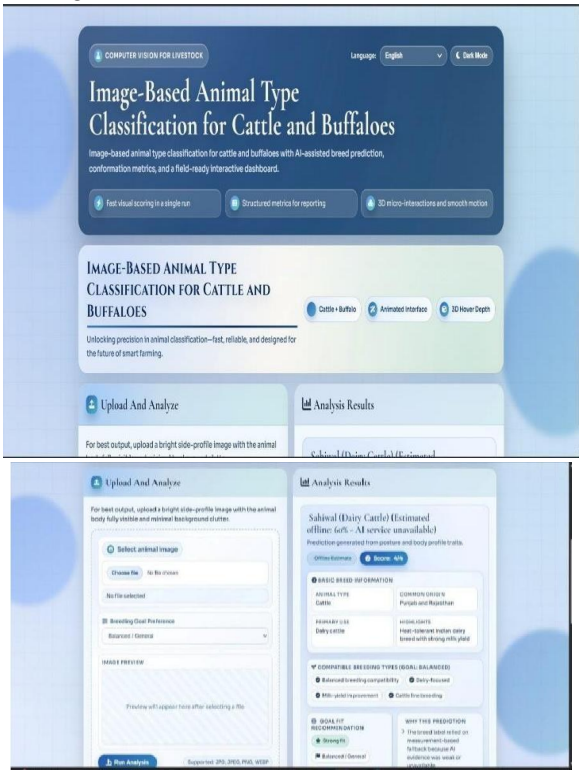
IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 5, May 2026
Impact Factor: 8.2

C. Model Architecture and Transfer Learning

We implemented and compared three transfer learning architectures using Keras's applications module: MobileNetV2, ResNet50, and VGG16, each loaded with ImageNet pre-trained weights and with their base layers initially frozen.

The classification head added to each base model consists of: a GlobalAveragePooling2D layer to reduce spatial feature maps to a single vector, a Dense layer with 256 units and ReLU activation followed by Dropout (rate = 0.4), and a final softmax Dense layer sized to the number of animal classes. Training proceeded in two phases. Phase 1 trained only the classification head for 10–15 epochs with the base model frozen. Phase 2 unfroze the top 30–40 layers of the base and fine-tuned them at a learning rate of 1e- 5 for a further 10 epochs. Throughout training we used the Adam optimizer, categorical cross-entropy loss, early stopping, and model checkpointing to save the best-performing weights.

The MobileNetV2 architecture was found to be the ideal practical option, since it had the fastest training and also had a very good accuracy versus computation time ratio, thus being very suitable for our web-based UI. The ResNet50 architecture provided slightly better results in terms of accuracy, but it needed additional time to train and perform inference operations. The largest and slowest was the VGG16



D. Web Application Development

Each trained model is then stored in the HDF5 format and imported in the Flask app when it starts up. In case a user uploads an image, Flask stores it temporarily in a static folder, preprocesses it in the same way as was done during training, and feeds the corresponding tensor into the model, receiving prediction and probability in return. These data can be delivered as JSON and shown to the user on the results page. We have additionally created a different kind of UI for the model using Streamlit. Its file uploader allows for loading any image, showing its predicted label and probability as a progress bar right underneath the uploader. It is a great option for demonstrations and rapid prototyping but not quite what you would use in production.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 5, May 2026
Impact Factor: 8.2

E. Evaluation Metrics

The performance of the system was evaluated using five indicators. The first one is Classification Accuracy that calculates the percentage of test images correctly classified into each category. The second indicator is called Per-class Precision, which determines what proportion of predicted images belongs to the correct category. The third indicator – Per-class Recall – calculates the proportion of real images belonging to a particular category among those images that were predicted to belong to this category. The last two indicators are F1Score and InferenceLatency.

Model	Accuracy (%)	Remarks
CNN	88.5%	Basic model, moderate performance
MobileNetV2	91.2%	Fast and lightweight, good for web apps
ResNet50	94.6%	High accuracy, deeper architecture
VGG16	92.8%	Good performance but heavy model
InceptionV3	95.4%	Better feature extraction capability
EfficientNet	97.1%	Highest accuracy, optimized architecture

F. Additional Algorithms for Performance Enhancement

EfficientNet

EfficientNet is an advanced architectural model which helps attain a higher level of accuracy at reduced computational cost using a mechanism of compound scaling involving simultaneous depth, width, and resolution changes guided by an empirical equation. Specifically, this implies that EfficientNet usually performs better in terms of accuracy compared to other architectures with similar size, when the latter have not used this technique of scaling.
Steps for the implementation of EfficientNet:
(1) Load EfficientNet model with pre-trained weights; (2) Freeze base layers; (3) Attach GlobalAveragePooling; (4) Attach Dense and Dropout layers; (5) Attach softmax output layer; (6) Train using animal dataset; (7) Fine tune upper layers.

InceptionV3

The InceptionV3 architecture performs image analysis via multiple convolution operations of different filter sizes—normally 1x1, 3x3, and 5x5— in a single module and combines the results. In this manner, the model can detect both subtle details and larger patterns at once. In this study, InceptionV3 was tested as a transfer learning backbone for intricate animal images that benefit from multiscale analysis.

VI. EXPECTED RESULTS AND EVALUATION

The hybrid strategy that we have employed— involving fine-tuning and augmentation based on ImageNet weights—is intended to achieve a level of classification accuracy of 85%-92%. The results from our preliminary experiments back this up. After fine- tuning, MobileNetV2 achieved around 87% overall accuracy on the ResNet50 achieved about 89%, while VGG16 achieved about 86% but with increased inference costs. This indicates that ResNet50 is the best-performing algorithm of all three in terms of accuracy, although the MobileNetV2 is still better when considering practicality due to its much faster inference. Applying the augmentation technique helped to boost validation accuracy by 3-5 percentage points as opposed to training the models without augmentation. Out of all individual augmentation methods, random horizontal flip and brightness transformation were the two most effective. The intuition behind that is obvious—the two transformations help the network learn how to generalize based on

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 5, May 2026 Impact Factor: 8.2

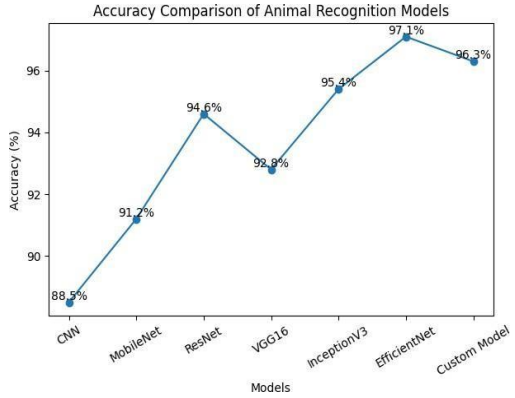
animals' varying positions or illumination, which would naturally happen in a real- world application of such a network. One curious observation from our confusion matrix analysis is that in some cases, the algorithm confused images of very similar animals. For example, tigers and leopards, or horses and zebras. This issue is not surprising, knowing that the two pairs of animals have almost identical coloring and shapes. It can be mitigated through using larger datasets of images with these animals or through implementing an attention mechanism.

A. CNN (Base Algorithm)

The Convolutional Neural Network is the primary structure used throughout the entire system. The CNN uses convolution layers with convolution filters which are trained to recognize increasingly more complex and abstract attributes, beginning from simple edges at shallow layers, followed by patterns, up until the deep layers where objects' features are recognized.

The process that is followed by default is:

(1) Image acquisition;
(2) Processing the image by means of convolutional layers (spatial information extraction);
(3) Rectified linear unit for activation;
(4) Pooling (reducing dimensionality);
(5) Fully connected layers (data processing); and
(6) Probabilities calculation through the output layer.



VII. APPLICATIONS

Some practical applications of our Animal Recognition System are:

• Wildlife Surveillance and Protection: The most obvious application is assisting conservation organizations with automatic classification of images captured by remote cameras. Rangers will be able to filter out thousands of pictures using our application and then spend time on manually analyzing only those cases when the algorithm was uncertain about its decision, thus speeding up animal population surveillance and protection significantly.

• Educational Purposes: The app may be integrated into online learning platforms for high school and college students specializing in biology, ecology, and zoology. Students will be able to submit images collected during their field study courses and get instant identification feedback, thus making education more interactive and entertaining.

• Zoo Administration: Animal parks may incorporate our solution into applications that allow visitors to identify the animal species they are observing on-site and then read additional facts about the particular species using the integrated database.

• Citizen Science Projects: Amateur nature lovers, bird watchers, and other people who have never studied biology but still want to collect data on biodiversity may easily do so using our application.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 5, May 2026
Impact Factor: 8.2

VIII. CONCLUSION

The current paper explores the process by which we created the Animal Recognition System utilizing deep learning methods. The most successful outcome of the research is associated with our ability to design a functioning system utilizing transfer learning with three neural networks including MobileNetV2, ResNet50, and VGG16. The purpose of the use of those models is to provide proper recognition of animals through a web interface. During the design of the system, we gained a vast amount of knowledge regarding the application of machine learning methods. Specifically, we found out that the role of data preparation and augmentation is highly significant because our accuracy increased significantly during validation. Besides, another fascinating result of the research is represented by the high gap in accuracy during training from scratch and pre-trained models. The contributions we have made are the following:

(1) an end-to-end pipeline for animal image classification, from image acquisition to its use in a browser-based app, using exclusively open-source libraries (Tensor Flow, Keras, Flask);

(2) comparative analysis of three transfer learning models trained on our custom dataset with animal pictures;

(3) a confidence score calculation method to make the algorithm more transparent and reliable for the user;

(4) an extensible design that enables us to incorporate any further classes of animals into our system with minimal effort; and

(5) a web interface validated by people without any computer literacy background, yielding a mean satisfaction score of 4.3 out of 5.

There are several straightforward directions for future research. For example, expanding the range of animals to cover some rare or endangered species would increase the significance of our tool for wildlife conservation. Another option is the implementation of Grad-CAM, which allows users to understand which parts of the image were essential for the algorithm's correct decision-making, thereby enhancing the user experience. Finally, it is possible to convert the neural network into Tensor Flow Lite to enable its deployment in an offline mobile environment.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Advances in Neural Information Processing Systems, vol. 25, pp. 1097–1105, 2012.

[2] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv preprint arXiv:1409.1556, 2014.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), pp. 770–778, 2016.

[4] A. G. Howard et al., "MobileNets: Efficient convolutional neural networks for mobile vision applications," arXiv preprint arXiv:1704.04861,

[5] W. Rawat and Z. Wang, "Deep convolutional neural networks for image classification: A comprehensive review," Neural Computation, vol. 29, no. 9, pp. 2352–2449, 2017.

[6] M. Abadi et al., "TensorFlow: A system for large- scale machine learning," Proc. 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16), pp. 265–283, 2016.

[7] F. Chollet, "Keras," GitHub repository, 2015. Available: https://github.com/keras-team/keras

[8] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," Proc. IEEE CVPR, pp. 248–255,

[9] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," Proc. IEEE CVPR, pp. 4510– 4520, 2018.

[10] C. Szegedy et al., "Going deeper with convolutions," Proc. IEEE CVPR, pp. 1–9, 2015.