ANIMAL CLASSIFICATION USING CNN

A. Abstract

Classification and recognition of wild animals for tracking and protection purposes have become increasingly important to improve habitat, environmental, and extinction patterns. Animal image classification using CNN is a commonly used approach. This study compares the performance of three different CNN architectures MobileNetV2 [1] , ResNet18 [2], and ShuffleNetV2 [3] on three distinct animal datasets each containing images of various animals. Each dataset is preprocessed and trained separately using each CNN architecture. The accuracy of the models is assessed using the corresponding test datasets for each animal classification task. The architectures are trained from scratch and with transfer learning. Model performance was measured using testing accuracy, F1 score, precision, and recall. The results indicate that MobileNetV2 [1] outperforms the other two architectures on all three datasets when trained from scratch. However, during transfer learning, ResNet18 [2] achieves the highest accuracy among all models which is 98%. The study concludes that the choice of dataset affects the performance of CNN architectures.

B. Introduction

Efficient and reliable monitoring of wild animals in their natural habitats is essential to inform conservation and management decisions regarding wildlife species, migration patterns, and habitat protection, and is possible, to rehabilitate and group species of the same animals together. Animal classification has good applications in the field of biology, ecology, and conservation. Processing a large volume of images and videos captured from camera traps [4] manually is extremely expensive, time-consuming, and monotonous [5]. This presents a major obstacle to scientists and ecologists to monitor wildlife in an open environment. Images captured in a field represents a challenging task while classifying since they appear in a different pose, cluttered background, different lighting and climatic conditions, human photographic errors, different angles, and occlusions.

Manual animal classification can be challenging due to a large number of animal species and the variability in their appearance. However, several approaches have been proposed in the literature to address these challenges. Feature extraction methods involve manually identifying and extracting relevant features from animal images. These features can include color, texture, shape, and size. These extracted features are then used to train a classification models. Ensemble methods [6] combine the predictions of multiple classification models to improve accuracy. For example, in one study, an ensemble of support vector machines (SVMs) was used to classify animal images. Deep learning approaches, such as CNN, have been used to learn relevant features from animal images automatically. These models can achieve high accuracy, but they require large amounts of labeled data for training. Citizen science projects, such as the Zooniverse platform, involve enlisting the help of non-experts to classify animal images. These projects have been successful in classifying large numbers of images and generating labeled datasets for machine-learning models.

One of the biggest pros of existing manual animal classification is their ability to provide accurate results. Trained experts in animal classification can achieve high levels of accuracy, especially for challenging or novel species. Humans can adapt to new situations and recognize new animals without requiring extensive retraining. Humans can use their knowledge of animal behavior, habitats, and ecological relationships to provide context to animal classification. In terms of cons, Manual classification is a time-consuming process that requires significant human effort, limiting its scalability. Wildlife counts manually only provide estimates and not actual population size, because of various sources of errors such as imperfect detection, imperfect abilities to count animals that are detected [18], misidentification of species or nonexhaustive geographical coverage. When unaccounted for, these errors can introduce considerable estimation bias and obscure important ecological patterns [20], which can reduce the power to detect trends and the accuracy of any trends that are detected. Different individuals may classify the same image differently, leading to potential inconsistencies and biases. The availability of experts in animal classification may be limited, particularly in remote or under-resourced areas. Manual classification can be difficult to reproduce across different individuals or time points, leading to potential errors and inconsistencies.

To address these challenges, advances in technology have led to the development of automated animal classification techniques [7]. These methods use artificial intelligence and Deep learning techniques to analyze images. These can automatically detect and identify species, track animal movements, and even recognize individual animals based on their unique characteristics. We conducted a deep learning-based automated animal classification study that utilizes various Convolution Neural Network architectures, such as ResNet18 [2], ShuffleNetV2 [3], and MobileNetV2 [1]. This study provides valuable insights into the impact of datasets on the performance of different Convolutional Neural Networks. By evaluating the performance of MobileNetV2, ShuffleNetV2, and ResNet18 on three different datasets, we were able to observe how the choice of the dataset can significantly impact the effectiveness of each architecture.

These networks were trained on 3 different datasets using training from scratch. We have also used two techniques of transfer learning which are Deep Tuning and Fine tuning. After training, we compared the performance of the CNN models trained from scratch with the performance of the models trained using transfer learning techniques. Additionally, the quality of the data preprocessing steps performed on the datasets should be assessed, and the learned representations are visualized using t-SNE [8] to assess the quality of the learned features. These additional evaluation metrics will provide a more comprehensive understanding of the performance of the CNN models on different datasets.

B.1. Related work

The authors of this paper [9] recommend a method for automated underwater fish species classification. Popular approaches emphasize the classification of fishes outside of water because underwater classification carries several challenges such as background noises, distortion, object discrimination, image quality, and occlusion. The proposed method suggested the implementation of removing the noise in the dataset.

The paper [10], researchers evaluate the effectiveness of Convolutional Neural Networks (CNNs) for practical applications in wildlife conservation by assessing how well these models can classify animal images captured in the field. There are also discussions on the role of technology in wildlife conservation, including remote sensing technologies and their limitations. The review then focuses on the use of computer vision and deep learning techniques for identifying and tracking animals in the field, with a particular emphasis on the application of CNNs.

In this paper, author [17] briefly discussed different components of CNN. In this paper, the author has explained different CNN architectures for image classification. Through this paper, Farhana Sultana has shown advancements in CNN from LeNet-5 to the latest SENet model. In this study, they discussed the model description and training details of each model. They also drawn a comparison among those models.

In this paper [11], A Bilateral Convolution Network model for lightweight classification is proposed. The model proposed is compared with the six models of AlexNet, ResNet18 [2], MobileNetV2 [1], ShuffleNetV2 [3], and DenseNet with respect to accuracy and model size. These all models have been tested on selected classes such as elephants, lions, tigers, foxes, pandas, and wild horses are tested on the AWA dataset and obtained 85.6% accuracy.

Based on these related works, we decided to train MobileNetV2 [1], ResNet18 [2], and ShuffleNetV2 [3] due to the significant behavior with which they have been reported, supported by their results.

C. Methodology

C.1. Datasets:-

The choice of the dataset is crucial as it directly affects the accuracy of the models. We used 3 different datasets for the training. These datasets are diverse, containing various species of animals, different images from various angles, and different sizes and lighting conditions. We used three different animal image datasets for our training purpose. The first dataset, called Dataset 1 [12], contains approximately 25,000 RGB images of dogs and cats with varying pixel resolutions, ranging from 320 x 200 to 498 x 480 pixels. Originally created by Microsoft Research for a Kaggle competition in 2013, we modified this dataset and selected 5,000 images for our training. Images within this dataset are in jpg format.

Info	Dataset 1	Dataset 2	Dataset 3
Original Categories	2	25	12
Original No.of Images	25K	15.1K	17.2K
Modified Categories	2	5	12
Modified No.of Images	5k	10k	15K

The second dataset, Dataset 2 [13], was provided by a user named Saumil Agrawal in 2018. This dataset contains around 15,100 images of 25 different animal classes, with a fixed pixel resolution of 1280 x 720 pixels. We observed that this dataset was highly imbalanced, with some classes having only 60 images. To avoid potential bias towards certain classes, we selected only 5 classes which has more number of images. Images within this dataset are in jpeg format.

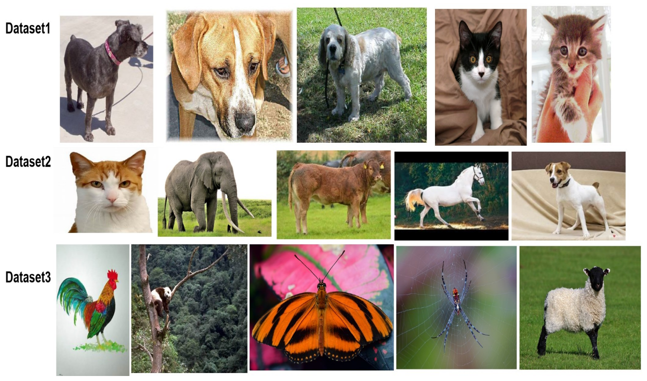

Figure 1: Sample Datasets images.

The third dataset, Dataset 3 [14], was created by Kaggle user Piyush Kumar in 2019. It contains approximately 17,200 images of various animals, including butterfly, cats, cows, elephants, hens, horses, monkeys, pandas, sheep, spiders, squirrels, among others. The size of the images are ranging from 201 x 300 to 500 x 374 pixels. Although, this dataset is relatively small compared to other image classification datasets like ImageNet [15] and CIFAR-10 [16]. Dataset 3 [14] serves as a useful benchmark for comparing the performance of different models on animal image classification tasks. For our training purpose, we selected 15,000 images from all 12 classes. Images in this dataset are in jpeg format.

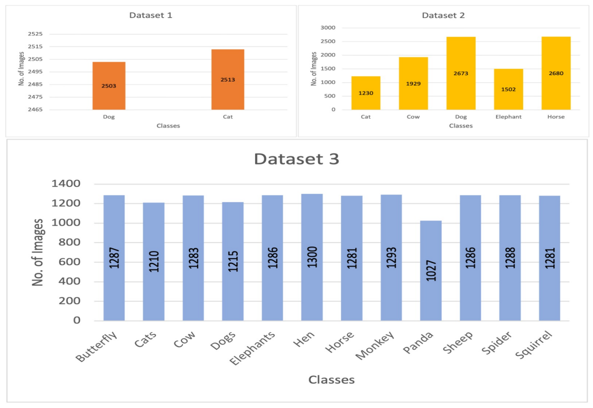

Figure 2: Dataset Wise Classes.

The class representations in all datasets vary in complexity, with some classes such as the butterfly and the elephant having relatively simple and distinct features, while other classes such as the spider and monkey have more complex and nuanced features that may be harder to learn. This makes the data set a good benchmark for training and testing the performance of deep neural networks on a range of classification tasks with varying degrees of complexity.

The dataset is divided into train, validation, and test sets in an 80:10:10 ratio, respectively. The images were randomly shuffled before the split. Test data is only used for evaluating the model.

To ensure consistent input dimensions for our neural network, we standardize images of varying sizes by resizing them to a uniform dimension of 224 x 224 x 3. Prior to feeding them into the neural network, we normalize the pixel values to fall within a pre-defined range of 0 to 1.

To address imbalanced classes within our dataset, we implement data augmentation techniques, specifically Random Cropping and Flipping. These techniques randomly crop and flip the images during training, creating additional variations of the original images to increase the diversity of the dataset. By doing so, we aim to improve the performance of the neural network in accurately classifying the various classes within the dataset.

C.2. CNN Models:-

MobileNetV2 [1] is a lightweight neural network architecture designed for mobile and embedded vision applications. The architecture can be divided into three main parts: the stem, the body, and the head. It uses inverted residual blocks. Inverted residual blocks are composed of three components: a linear bottleneck layer, a non-linear activation function, and a linear projection layer. The bottleneck layer reduces the dimensionality of the feature maps, while the projection layer expands them again. The non-linear activation function is placed between the two linear layers. Fully Connected layer has been modified. Four linear layers have been used, each followed by a ReLU activation function and a Dropout layer with a dropout probability of 0.1.

The architecture of ResNet18 [2] consists of a series of convolutional layers, followed by a global average pooling layer and a fully connected output layer with softmax activation. The ResNet18 [2] architecture consists of 18 layers, including convolutional layers, max-pooling layers, fully connected layers, and shortcut connections. The shortcut connections allow the network to bypass some of the layers, which helps to mitigate the vanishing gradient problem and makes it easier to train very deep neural networks. The ResNet18 [2] architecture uses a series of residual blocks, which are composed of two convolutional layers and a shortcut connection. Last layer has been changed. We used four linear layers, each followed by ReLU activation function and a Dropout layer with dropout probability of 0.2 and one linear layer for output.

The ShuffleNetV2 [3] architecture is composed of several building blocks, including the channel shuffle operation, depthwise separable convolution, and residual connections. The channel shuffle operation is a key component of the ShuffleNetV2 [3] architecture, and it allows for information exchange between channels while reducing computation. This operation shuffles the channels of feature maps within a group of convolutional filters, thereby allowing for the mixing of information across different channels. The depthwise separable convolution is key component of ShuffleNetV2 [3], and it involves breaking down a convolutional layer into a depthwise convolution layer followed by a pointwise convolution layer. It uses a hybrid block, which combines the depth-wise convolution operation with a traditional 1 x 1 convolution operation. This hybrid block improves the accuracy of the network while keeping the computational cost low. The output of the final building block is passed through a global average pooling layer, followed by a fully connected output layer with Softmax activation.

The estimated wall clock time for one-epoch training on a single GPU for three different datasets has been provided for three different convolutional neural network architectures. For dataset1, the one-epoch training time for MobileNetV2 [1], ResNet18 [2], and ShuffleNetV2 [3] is approximately 2.8, 2.4, and 2.6 minutes, respectively. For dataset 2, the training time for these architectures is approximately 5.6, 4.9, and 5.1 minutes, respectively. Similarly, for dataset3, the training time for MobileNetV2 [1], ResNet18 [2], and ShuffleNetV2 [3] is approximately 20, 8, and 9.2 minutes, respectively.

In terms of computational efficiency, ResNet18 [2] requires around 1.82 billion FLOPs for a single inference and has 11.2 million parameters. In comparison, MobileNetV2 [1] requires around 300 million FLOPs for a single inference and has 3.5 million parameters. ShuffleNetV2 [3], on the other hand, requires around 140 million FLOPs for a single inference and has 5.4 million parameters. Generally, MobileNetV2 [1] and ShuffleNetV2 [3] are more computationally efficient than ResNet18 [2] as they require a lower number of FLOPs. However, ResNet18 [2] is a more complex architecture than the other two.

When compared to other popular image classification models ResNet18, ShuffleNetV2, and MobileNetV2 are known for their computational efficiency while still achieving good accuracy. These architectures are especially useful for applications where computational resources are limited, such as on mobile devices or embedded systems. VGG and DenseNet, on the other hand, are known for their accuracy but require more computational resources. AlexNet is a classic architecture that is less accurate than some of the newer models but is still useful for some applications.

C.3. Optimization

The optimization algorithm chosen for a specific problem is influenced by the unique characteristics of the data involved. For our project, we employed the grid search technique to optimize the hyperparameters, including learning rates of 0.001, 0.01, and 0.1, batch sizes of 32, 16, and 64. To determine the ideal hyperparameters, we applied this combination to train MobileNetV2 on Dataset 1 Fig. A shows that best hyperparametes were 0.001 Learning Rate and Batch size is 32. It achieves 82% testing accuracy.

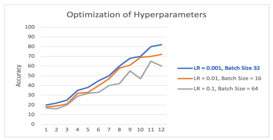

Fig.A. Optimization of Hyperparameters.

In this research study, we utilized the Adam Optimizer for optimization, which has gained prominence due to its capacity to efficiently converge on a diverse range of problems. One of the benefits of using the Adam optimizer is its ability to handle gradients that are noisy or sparse. Adam help to alleviate the impact of noisy or sparse gradients, enabling the model to converge more efficiently.

To evaluate the performance of the proposed method, we conducted a comparative analysis of different models using various metrics such as F1 score, Testing accuracy, Precision, and Recall. The results of this analysis are presented in the Results section of this paper.

D. Results

Experiment Setup: The selection of appropriate hyperparameters for training convolutional neural networks (CNNs) is a critical task that greatly impacts the performance of the model. This research determined the hyperparameters for animal classification based on the intricacy of the animal datasets, which included factors like the size of the dataset, its dimensionality, and noise levels. Additionally, the computational resources available for training were also considered. Table1 suggest the optimized hyperparameter used in the experiment.

Learning rate	0.001
Optimizer	Adam
Loss Function	Cross Entropy loss
Batch Size	32
Epochs	30

Table 1. List of Hyperparameters.

All experiments are carried out on a Google Colab and Microsoft Azure. Microsoft Azure has 6 cores, 56 GB RAM, 12 GB NVIDIA TESLA K80 GPU with 380 GB disk space and Google Colab has 13 GB RAM and 15 GB GPU with 79.2GB Disk space.

The performance of the proposed method is evaluated by comparing the different models with different metrics. The quality of CNN models is evaluated using how well they perform on test data. The sensitivity or recall corresponds to how many examples of the positive classes were labeled correctly. The precision measure is about correctness. It is how “precise” the model is out of those predicted positives and how many of them are actually positive. F1-score is determined as the harmonic mean precision and recall. A high value of this metric indicates that the model performs better on the positive class. In this experiment, all 4 majors are used for Evaluation.

Main Results and Ablative Study: The ability of a model to accurately fit the training data is an important metric for evaluating its performance. One commonly used metric to measure this ability is the training loss, which represents the difference between the predicted output and the actual output of the model on the training data.

In this study, we evaluated the performance of three different architectures, namely MobileNetV2, ShuffleNetV2, and ResNet18, on a dataset called Dataset1. We observed that the training loss decreased steadily throughout training for all three architectures, indicating an improvement in their ability to classify input images.

Among these architectures, MobileNetV2 and ShuffleNetV2 exhibited the lowest training loss, both less than 0.2 after 30 epochs. In contrast, ResNet18 had a slightly higher training loss, approximately 0.5, and also exhibited higher fluctuation in loss compared to the other architectures. These results suggests that MobileNetV2 and ShuffleNetV2 may be more efficient in terms of classifying.

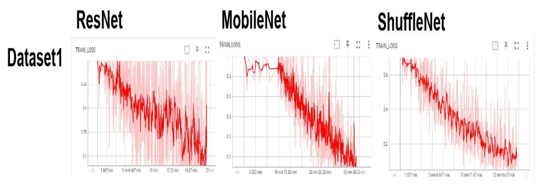

Figure 3. Training loss of dataset-1.

Figure 4 represents training loss on dataset 2. It has a similar pattern as dataset 1.

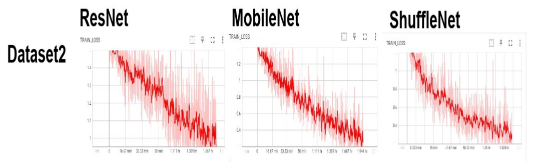

Figure 4. Training loss of dataset-2.

From Figure 5, It is observed that the training loss pattern for Dataset 3 was different compared to Dataset 1 and Dataset 2. Specifically, MobileNetV2 and ShuffleNetV2 reached convergence after 25 epochs, with their training loss becoming almost stable. In contrast, ResNet18 continued to exhibit fluctuations in loss beyond this point, indicating that it had not yet fully converged.

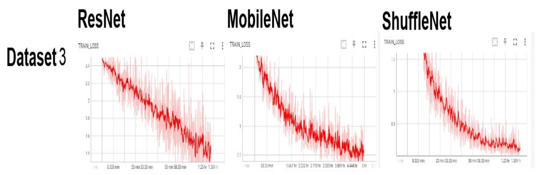

Figure 5. Training loss of Dataset-3.

The graph of losses suggests that MobileNetV2 and ShuffleNetV2 are more effective architectures for this particular application, as they were able to achieve convergence faster and more consistently compared to ResNet18.

In this study, we monitored training accuracy over time. Figure 6, 7 and 8 represents training accuracy on dataset1, dataset2, and dataset3 respectively. On dataset1 and dataset2, MobileNetV2 and ShufflenetV2 achieved almost 95% accuracy, while ResNet18 performed poorly with around 78% accuracy.

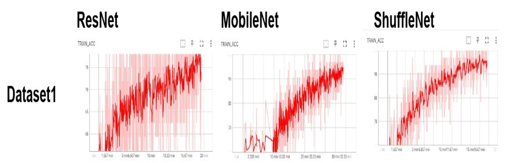

Figure 6. Training accuracy of Dataset-1.

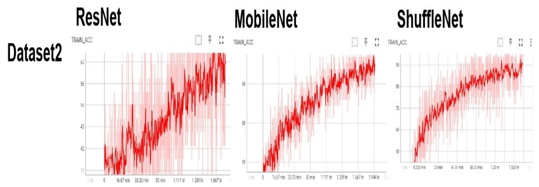

Figure 7. Training accuracy of Dataset-2.

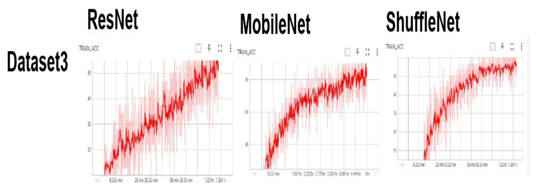

Figure 8. Training accuracy of Dataset-3.

On dataset 3, ResNet18 face issue of underfitting. It achieves 55% accuracy on training data and also, from Table 2 it has 53.50% testing accuracy. This suggests that ResNet18 was unable to capture the underlying patterns in the data and instead made high-error assumptions about the data.

Training loss and training accuracy are not enough to evaluate the performance of the model. Using the validation accuracy graph [Fig.14], Performance of different trained models has been compared and that enables us to choose the best model. After that, we tested the best models on the testing data and achieved Precision, Recall and an F1 score. From Table 2 it is clear that MobileNetV2 achieves the highest testing accuracy on all datasets. Furthermore, our study showed that the training time of MobileNetV2 was significantly more than ResNet18 and ShuffleNetV2.

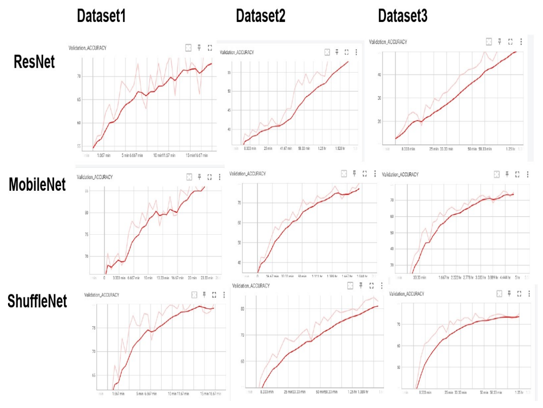

Figure 14. Validation accuracy of models.

For Dataset 1, the highest testing accuracy was achieved by MobileNetV2, which is 87.60% followed by ShuffleNetV2 and ResNet18 with 82% and 75.40%. ShuffleNetV2 takes the lowest time to train on Dataset 1 which is 20 min. MobileNetV2 achieves the highest Precision, Recall, and F1 score among all models. For Dataset 2, MobileNetV2, and ShuffleNetV2 achieves almost the same testing accuracy of around 80% and ResNet18 has the least accuracy of 61.71. However, its training time is the lowest. Similarly, on Dataset 3, Model accuracy is going down from MobileNetV2, ShuffleNetV2 to ResNet18.

This paper used t-SNE [8] to visualize the results of models that have learned high-dimensional representations of data. By plotting the high-dimensional representations of the data points in a low-dimensional space using t-SNE [8], we gain insights into how the model is representing the data and identify any patterns or clusters in the data that the model has learned. Here from Figure 9, It is clear that MobileNetV2 [1] and ShuffleNetV2 gives a Distinct, well-separated cluster. ResNet18 [2] has many outliers. Similarly, For dataset2 and dataset3 we can observe a similar data pattern.

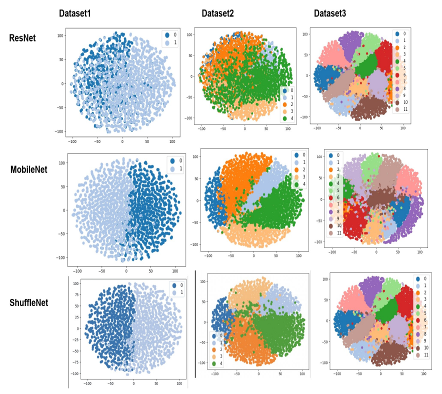

Figure 9. TSNE visilization.

Furthermore, Figure 10 represents the confusion matrix of the model with the best outcome on Dataset 1, which is MobileNetV2. In Dataset 1, the Dog class has a total of 250 images. Total 221 images are identified correctly in the dog class and the Rest of the 29 are predicted as Cat which is wrong. Similarly, Figure 11 and Figure 12, it shows Confusion Matrix for Dataset 2 and Dataset 3.

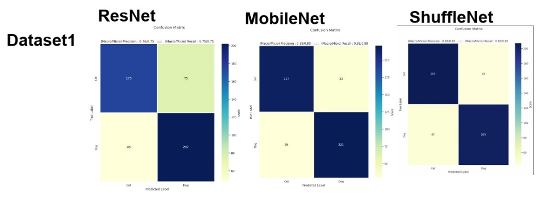

Figure 10. Confusion matrix Dataset 1.

Overall, MobileNetV2 achieved the highest accuracy on all three datasets, but the training time was longer for Dataset 3 compared to the other architectures. ShuffleNetV2 achieved competitive accuracy with faster training times, while ResNet18 had lower accuracy on all three datasets. Additionally, as the complexity of the data increased, the performance of all models decreased.

Ablative study of a CNN for animal image classification, we compared the performance of a CNN trained from scratch with the performance of the same CNN architecture using transfer learning. In this study, Deep Tuning and Fine Tuning is used as a transfer learning methods. In Fine Tuning pre-trained model on IMAGENET1K_V1 is used. For Deep Tuning, only fully connected layers were trained all other layers were freeze.

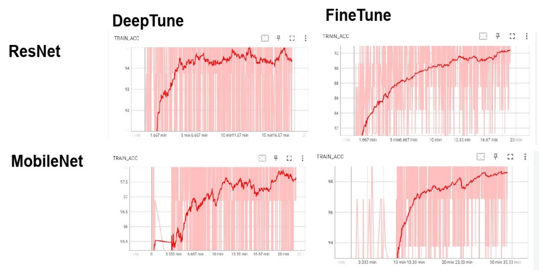

Figure 13. training accuracy of transfer learning models.

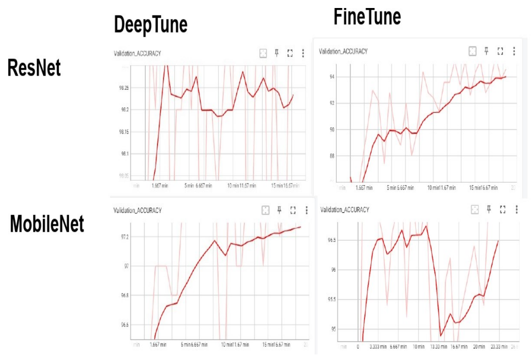

Figure 15. Validation accuracy of transfer learning models.

Fig. 6 and Fig. 13 shows difference in training accuracy for pre-trained and scratch models. In our study, the results show that transfer learning is better than training from scratch.Fig. 15 is for validation accuracy. We observed[from Table2] all models achieve more than 92% validation accuracy. ResNet18 [2] from scratch achieved 75.40% testing accuracy, while in it jumps drastically to 98.20% and 93.00% in Deep Tuning and Fine Tuning, respectively. Moreover, the Transfer Learning model converges faster than the models trained from scratch. MobileNetV2 [1] has achieved 96% testing accuracy on both methods.

References

[1] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, Liang-Chieh Chen. MobileNetV2 Inverted Residuals and Linear Bottlenecks. arXiv:1801.04381v4 [cs.CV] 21 Mar 2019. 1, 2, 3, 4, 6

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun. Deep Residual Learning for Image Recognition.arXiv:1512.03385 [cs.CV] 10 Dec 2015. 1, 2, 3, 4, 6

[3] Ningning Ma, Xiangyu Zhang, HaiTao Zheng, Jian Sun. ShuffleNet Practical Guidelines for Efficient CNN Architecture Design. arXiv:1807.11164 [cs.CV] 21 Dec 2021. 1, 2, 3, 4

[4] Thirupathi Battu, D. Sreenivasa Reddy Lakshmi, Animal image identification and classification using deep neural networks techniques, Measurement: Sensors, Volume 25, 2023, 100611, ISSN 2665-9174, https://doi.org/10.1016/j.measen.2022.100611. 1

[5] Julia Larson. Assessing Convolutional Neural Network Animal Classification Models for Practical Applications in Wildlife Conservation. Master's Theses. 5184 (2021). 1

[6] Altman, Naomi, and Martin Krzywinski. "Ensemble methods: bagging and random forests." Nature Methods 14.10 (2017): 933-935. 1

[7] Yu, Xiaoyuan, et al. "Automated identification of animal species in camera trap images." EURASIP Journal on Image and Video Processing 2013.1 (2013): 1-10. 1

[8] van der Maaten, Laurens & Hinton, Geoffrey. (2008). Visualizing data using t-SNE. Journal of Machine Learning Research. 9. 2579-2605. 2, 6

[9] Dhruv Rathi, Sushant Jain, Dr. S. Indu, "Underwater Fish Species Classification using Convolutional Neural Network and Deep Learning", International Conference of Advances in Pattern Recognition, 2017. 2

[10] Julia Larson. Assessing Convolutional Neural Network Animal Classification Models for Practical Applications in Wildlife Conservation. Master's Theses. 5184 (2021). 2

[11] Sujatha Kamepalli, Venkata Krishna Kishore Kolli, Srinivasa Rao Bandaru, "Animal Breed Classification and Prediction Using Convolutional Neural Network Primates as a Case Study", 2021 Fourth International Conference on Electrical, Computer and Communication Technologies (ICECCT), pp.1-7, 2021. 2

[12] Will Cukierski.(2013). Dogs vs. Cats. Kaggle. https://kaggle.com/competitions/dogs-vs-cats 2

[13] Saumil Agrawal. Animal Image Dataset resized. Kaggle. https://www.kaggle.com/datasets/saumilagrawal10/animal-image-dataset-resized 2

[14] Piyush Kumar. Animal Image Classification Dataset. Kaggle. https://www.kaggle.com/datasets/piyushkumar18/animal-image-classification-dataset 2, 3

[15] 1. Deng J, Dong W, Socher R, Li L-J, Li K, Fei-Fei L. Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. 2009. p. 248–55. 3

[16] Y. Krizhevsky, "Learning Multiple Layers of Features from Tiny Images," Master's thesis, Department of Computer Science, University of Toronto, 2009. 3

[17] F. Sultana, A. Sufian and P. Dutta, "Advancements in Image Classification using Convolutional Neural Network," 2018 Fourth International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN), Kolkata, India, 2018, pp. 122-129, doi:10.1109/ICRCICN.2018.8718718. 2

[18] Williams, B. K., Nichols, J. D., Conroy, M. J. (2002). Analysis and management of animal populations. Academic Press. 1

[19] Peiyi Zeng. Research on Similar Animal Classification Based on CNN Algorithm. J. Phys.: Conf. Ser. 2132 012001 (2021).

[20] Wenger, S. J. , Freeman, M. C. (2008). Estimating species occurrence, abundance, and detection probability using zero-inflated distributions. Ecology, 89(10), 2953–2959. 10.1890/07-1127.1

[21] Keiron O'Shea and Ryan Nash. An Introduction to Convolutional Neural Networks. arXiv:1511.08458v2 [cs.NE] 2 Dec 2015. 1

[22] Jiang, Bin & Huang, Wei & Wenxuan, Tu & Yang, Chao. (2019). An Animal Classification based on Light Convolutional Network Neural Network. 45-50. 10.1109/ICEA.2019.8858309.

E. Supplementary Material

Architecture	Dataset	Testing Acc.	Precision	Recall	F1-Scores	Training time
MobileNetV2	Dataset1	87.60	88	88	87	36 min
	Dataset2	80.22	80	81	80	2 hour
	Dataset3	73.69	75	74	74	5 hour
ShuffleNetV2	Dataset1	82	82	82	82	20 min
	Dataset2	79.82	82	79	79	1.2 hour
	Dataset3	72.98	72	73	72	1.3 hour
ResNet18	Dataset1	75.40	76	75	75	21 min
	Dataset2	61.71	67	56	61	1.1 hour
	Dataset3	53.50	52	54	53	1.3 hour

Table 2. Results of Model Dataset Wise.

Architecture	Dataset	Method	Testing Acc.	Precision	Recall	F1-Scores	Training time
ResNet18	Dataset1	DeepTune	98.20	98	98	98	19 min
	Dataset1	FineTune	93.00	94	93	93	30 min
MobileNetV2	Dataset1	DeepTune	96.80	97	97	97	26 min
	Dataset1	FineTune	96.60	97	97	97	34 min

Table 3 Results of transfer learning model.

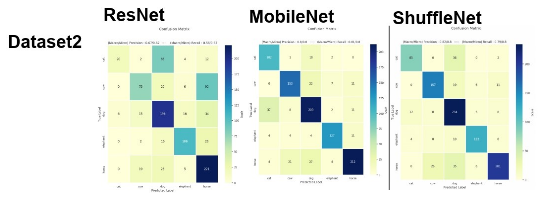

Figure 11. Confusion matrix Dataset 2.

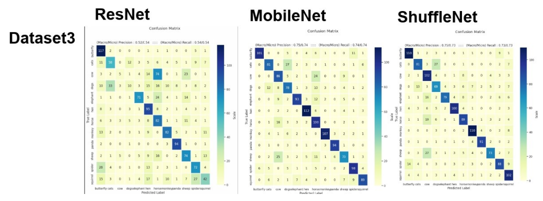

Figure 12. Confusion matrix Dataset 3.