M.G.M's College of Engineering, Nanded
Department of Computer Science & Engineering
Academic Year: 2024-25 (EVEN SEM)

---

# Machine Learning Project Report

## Title: "Image Classification Animal Dataset Using – CNN"

**Submitted by:**

**Name** : Ms.Sanskruti Suhasrao Pande

**Class** : TY CSE- A

**Roll_No** : 118

**Subject Incharge:**

Ms. Nitu L.Pariyal

---

## Abstract

This project explores the use of Convolutional Neural Networks (CNNs) for the classification of animal images. A CNN model is implemented using TensorFlow and trained on a custom dataset consisting of multiple animal classes. The primary goal is to correctly identify the class of new, unseen images. With a wellstructured CNN architecture and proper training, the model achieved high accuracy. This project demonstrates the effectiveness of deep learning for image classification and lays the foundation for future improvements such as transfer learning.

---

## Objective

To build and train a Convolutional Neural Network (CNN) to classify images of animals into their respective categories with high accuracy. The project aims to explore basic deep learning techniques and visualize how CNNs perform in realworld classification problems.

---

## Dataset

- The dataset contains labeled folders, each representing a different animal class (e.g., dogs, cats, elephants, lions).
- Images are collected in separate folders and organized for training and validation purposes.
- Data is loaded using TensorFlow's image_dataset_from_directory() function.
- The dataset is split into: ◦ **80% for training** ◦ **20% for validation**

---

## Preprocessing

- All images are resized to **128×128 pixels** for uniform input size.
- Pixel values are normalized to a range of **0–1** using the Rescaling(1./255) layer.
- The data is shuffled and batched during loading to ensure better model performance.

---

## CNN Model Architecture

1. **Input Layer**: Rescaling layer to normalize pixel values.
2. **Convolutional Layer 1**: 16 filters, kernel size 3×3, activation = ReLU ◦

MaxPooling Layer (2×2)

3. **Convolutional Layer 2**: 32 filters, kernel size 3×3, activation = ReLU 。

   MaxPooling Layer (2×2)

4. **Convolutional Layer 3**: 64 filters, kernel size 3×3, activation = ReLU 。

   MaxPooling Layer (2×2)

5. **Flatten Layer**

6. **Dense Layer**: 64 neurons, activation = ReLU

7. **Output Layer**: Softmax activation for multi-class classification

**Input Shape**: (128, 128, 3)

**Output Shape**: (Number of animal classes)

---

## Training Details

- **Optimizer**: Adam
- **Loss Function**: Sparse Categorical Crossentropy
- **Epochs**: 5
- **Batch Size**: 32
- **Final Accuracy**: 。Training: ~90% 。Validation: ~85%

- No signs of overfitting were observed within 5 epochs.

---

## Visualization

- Accuracy vs. Epoch and Loss vs. Epoch graphs were plotted.
- Both training and validation accuracy increased steadily.

- Visualization confirmed the model learned generalizable patterns.

---

## Prediction

- The trained model (animal_cnn_model.h5) is used to predict the class of new images.
- Test images are resized and normalized before feeding into the model.
- The predicted class is displayed as output.

---

## Tools and Libraries Used

- **TensorFlow & Keras** – For model building and training
- **Matplotlib** – For plotting graphs
- **Python 3.x** – Programming language

---

## Results and Evaluation

- The model achieved high training and validation accuracy with minimal training.
- No overfitting observed during training.
- The current model performs well for clean and centered images.
- Additional evaluation metrics like confusion matrix and precision-recall can be added for deeper analysis.

---

## Conclusion

The CNN model developed for animal image classification performs efficiently with good accuracy. This project demonstrates how deep learning and CNNs can

be used effectively for visual recognition tasks. The model is lightweight and performs well even with a limited number of training epochs, making it suitable for basic applications.

---

## Future Improvements

- Apply **data augmentation** to improve generalization
- Train for **more epochs** to increase accuracy
- Use **transfer learning** with pre-trained models like **VGG16** or **MobileNet**
- Add **confusion matrix**, **F1-score**, and **classification report**
- Deploy the model as a simple **web app** for interactive usage

---

## References


[1] Tom M. Mitchell, Machine Learning, 1st Edition, McGraw-Hill Education, 1997. ISBN: 978-0070428072.

[2] François Chollet, Deep Learning with Python, 2nd Edition, Manning Publications, 2021. ISBN: 978-1617296864.

[3] Ian Goodfellow, Yoshua Bengio, and Aaron Courville, Deep Learning, 1st Edition, MIT Press, 2016. ISBN: 978-0262035613.

[4] Christopher M. Bishop, Pattern Recognition and Machine Learning, 1st Edition, Springer, 2006. ISBN: 978-0387310732.

[5] Ethem Alpaydin, Introduction to Machine Learning, 4th Edition, MIT Press, 2020. ISBN: 978-0262043793.


---