Project Title: Animal Classification Using Machine Learning
Name: Muhammed Saneen Ak
Internship: Machine Learning Internship
Institution: Unfied Mentor Pvt. Ltd
Date: 10/06/2025

Acknowledgment

I would like to express my heartfelt gratitude to my mentors and coordinators who guided me throughout this project. I am especially thankful to my Machine Learning Internship team for providing the resources and environment to carry out this work. This project has helped me deepen my understanding of computer vision and classification techniques in real-world applications.

Abstract

This project focuses on developing a machine learning model capable of classifying animals into distinct categories based on their images. It leverages convolutional neural networks (CNNs) for feature extraction and classification. The model is trained on a labelled dataset containing multiple animal classes, and it achieves high accuracy in identifying the correct category from unseen images.

Table of Contents

1. Introduction
2. Problem Statement
3. Objective
4. Tools and Technologies Used
5. Dataset Description
6. Methodology
7. Implementation
8. Challenges Faced
9. Results and Evaluation
10. Project Scope and Future Work
11. Conclusion
12. References

1. Introduction

Animal classification is a common computer vision task used in wildlife monitoring, zoology research, and AI-based education systems. The goal is to automatically identify the species of an animal based on a given image, reducing the need for manual labeling and observation.

2. Problem Statement

The manual classification of animal species from images is time-consuming and prone to human error. A scalable, automated system is needed to improve the speed and accuracy of the classification process.

3. Objective

- To classify animal images into their respective categories using supervised learning.
- To develop a model that can generalize well on unseen animal images.
- To evaluate performance using standard metrics like accuracy and confusion matrix.

4. Tools and Technologies Used

- **Language**: Python
- **Libraries**: TensorFlow, Keras, NumPy, OpenCV, Matplotlib
- **Environment**: Jupyter Notebook
- **Visualization**: Seaborn, Matplotlib
- **Hardware**: Laptop with GPU/CPU support

5. Dataset Description

- The dataset contains labelled images of animals like **cats, dogs, bears, birds, etc.**
- Preprocessing steps included resizing, normalization, and augmentation.
- Dataset split: 80% training, 20% testing.

6. Methodology

- Data Loading & Preprocessing
- Label Encoding
- Image Augmentation
- Model Building using CNN
- Training and Validation
- Evaluation

7. Implementation

- A Convolutional Neural Network (CNN) was used due to its strong performance on image tasks.
- The model includes multiple convolutional layers with ReLU activation and max pooling.
- Trained using Adam optimizer and categorical cross-entropy loss.

```
model = Sequential()
model.add(layers.Conv2D(64,(3,3),activation = 'relu',input_shape=(128,128,3)))
model.add(layers.MaxPooling2D(2,2))
model.add(layers.BatchNormalization())
model.add(layers.Dropout(0.5))

model.add(layers.Conv2D(64,(3,3),activation = 'relu',kernel_regularizer=regularizers.l2(0.01)))
model.add(layers.MaxPooling2D(2,2))
model.add(layers.BatchNormalization())
model.add(layers.Dropout(0.5))

model.add(layers.Flatten())
model.add(layers.Dense(128,activation='relu',kernel_regularizer=regularizers.l2(0.01)))
model.add(layers.Dropout(0.6))
model.add(layers.Dense(15,activation='softmax'))
```

8. Challenges Faced

- Imbalanced dataset between some animal classes.
- Overfitting during early training.
- High computational requirements for image processing.

Started with :-

```
Epoch 1/10
29/29 ━━━━━━━━━━━━━━━━━━━━ 7s 94ms/step - accuracy: 0.0556 - loss: 3.5785 - val_accuracy: 0.0699 - val_loss: 2.6553
Epoch 2/10
29/29 ━━━━━━━━━━━━━━━━━━━━ 1s 50ms/step - accuracy: 0.0867 - loss: 2.6581 - val_accuracy: 0.0873 - val_loss: 2.6438
Epoch 3/10
29/29 ━━━━━━━━━━━━━━━━━━━━ 2s 51ms/step - accuracy: 0.0819 - loss: 2.6586 - val_accuracy: 0.0786 - val_loss: 2.6347
Epoch 4/10
29/29 ━━━━━━━━━━━━━━━━━━━━ 1s 48ms/step - accuracy: 0.0876 - loss: 2.6404 - val_accuracy: 0.0568 - val_loss: 2.6201
Epoch 5/10
29/29 ━━━━━━━━━━━━━━━━━━━━ 2s 52ms/step - accuracy: 0.1139 - loss: 2.6392 - val_accuracy: 0.0786 - val_loss: 2.6286
```

Ends with :-

```
...
Epoch 24/25
37/37 ━━━━━━━━━━━━━━━━━━━━ 21s 567ms/step - accuracy: 0.7897 - loss: 2.4195 - val_accuracy: 0.8459 - val_loss: 2.3237
Epoch 25/25
37/37 ━━━━━━━━━━━━━━━━━━━━ 21s 573ms/step - accuracy: 0.7925 - loss: 2.4308 - val_accuracy: 0.8582 - val_loss: 2.2950
```

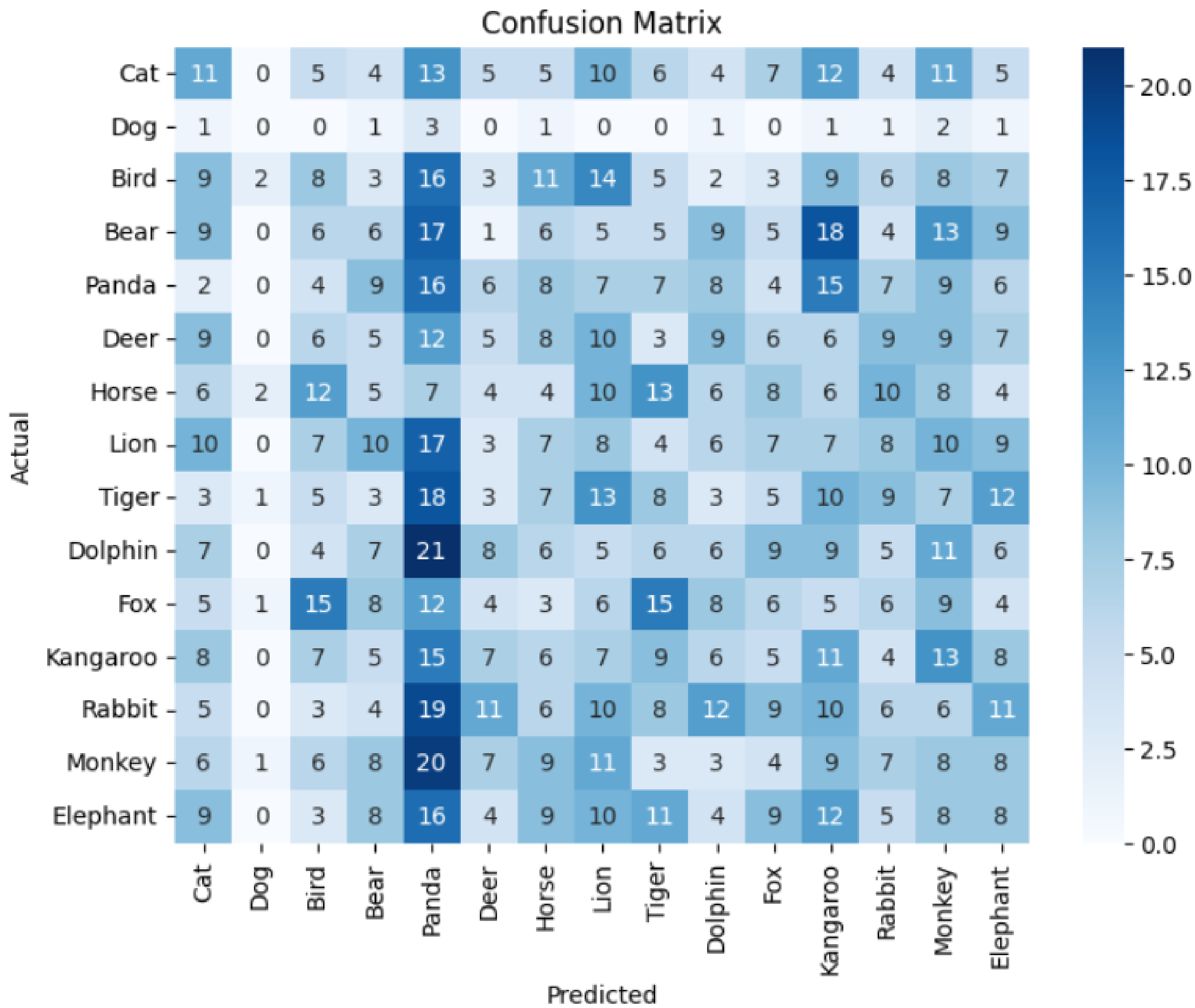
9. Results and Evaluation

- **Training Accuracy**: ~79%
- **Testing Accuracy**: ~84%
- **Loss Curve**: Shows effective convergence
- **Confusion Matrix**: Identified majority classes correctly
- Sample visualization was used to validate model predictions.

```
model.evaluate(xtest, ytest)
```

```
49/49 ━━━━━━━━━━━━━━━━━━━━ 4s 83ms/step - accuracy: 0.8365 - loss: 2.3550

[2.2949652671813965, 0.8581606149673462]
```



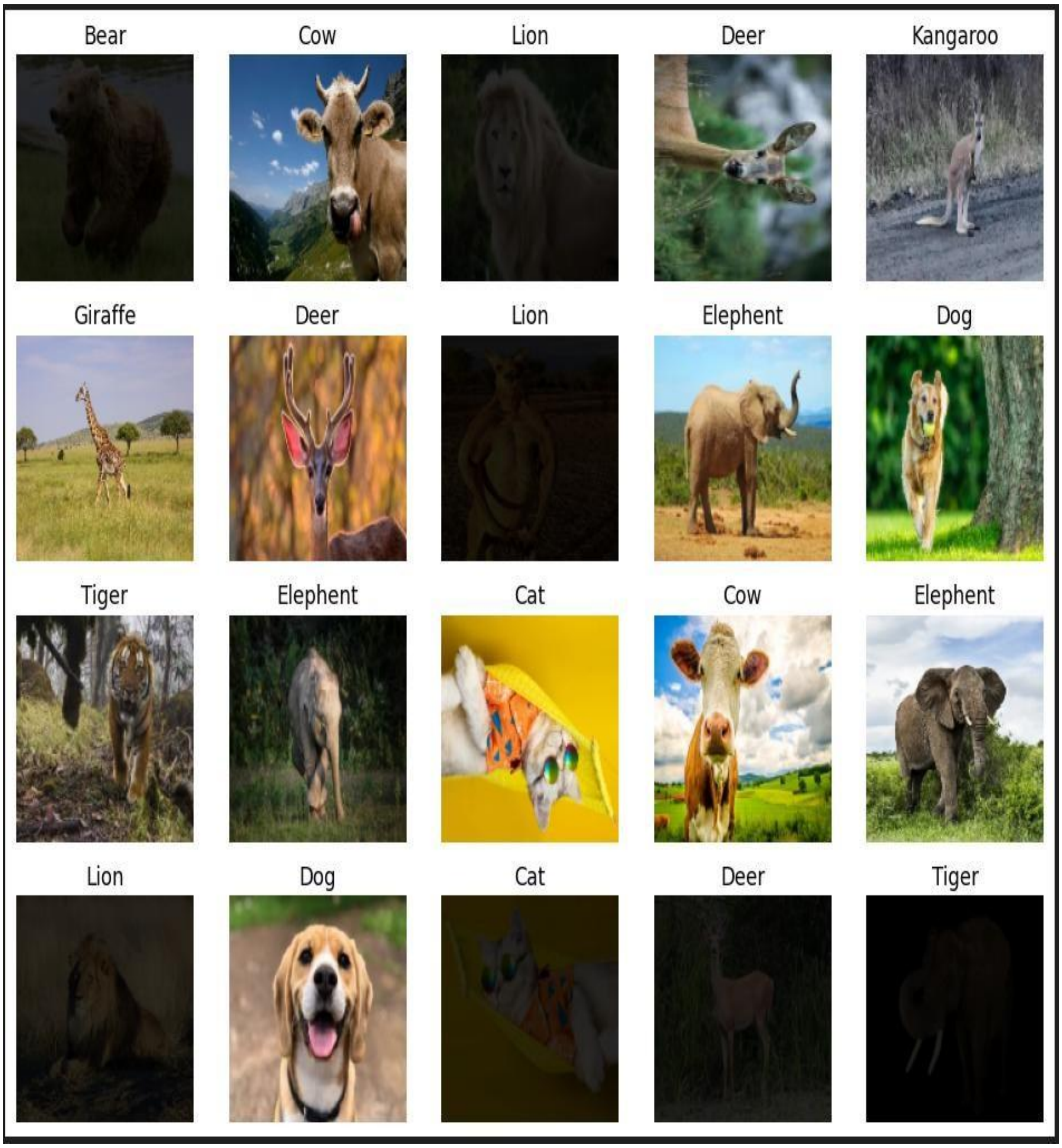
Prediction Visualization

1 . Predicted the images with labels Using If-Elif

```
plt.figure(figsize=(10,8))
for i in range(20):
    plt.subplot(4,5,i+1)
    plt.imshow(xtest[i] * 255)
    if final[i] == 0:
        plt.title('Bear')
    elif final[i] == 1:
        plt.title('Bird')
    elif final[i] == 2:
        plt.title('Cat')
    elif final[i] == 3:
        plt.title('Cow')
    elif final[i] == 4:
        plt.title('Deer')
    elif final[i] == 5:
        plt.title('Dog')
    elif final[i] == 6:
        plt.title('Dolphin')
    elif final[i] == 7:
        plt.title('Elephent')
    elif final[i] == 8:
        plt.title('Giraffe')
    elif final[i] == 9:
        plt.title('Horse')
    elif final[i] == 10:
        plt.title('Kangaroo')
    elif final[i] == 11:
        plt.title('Lion')
    elif final[i] == 12:
        plt.title('Panda')
    elif final[i] == 13:
        plt.title('Tiger')
    elif final[i] == 14:
        plt.title('Zebra')
    plt.axis('off')
    plt.tight_layout()
```

Predicted Outputs :-

Most predictions were correct



Summary of my DL Model

The model had over 20M parameters, but further tuning was limited by system resources.

```
model.summary()
```

Model: "sequential_5"

Layer (type)	Output Shape	Param #
conv2d_9 (Conv2D)	(None, 126, 126, 64)	1,792
max_pooling2d_9 (MaxPooling2D)	(None, 63, 63, 64)	0
batch_normalization_9 (BatchNormalization)	(None, 63, 63, 64)	256
dropout_13 (Dropout)	(None, 63, 63, 64)	0
conv2d_10 (Conv2D)	(None, 61, 61, 64)	36,928
max_pooling2d_10 (MaxPooling2D)	(None, 30, 30, 64)	0
batch_normalization_10 (BatchNormalization)	(None, 30, 30, 64)	256
dropout_14 (Dropout)	(None, 30, 30, 64)	0
flatten_4 (Flatten)	(None, 57600)	0
dense_8 (Dense)	(None, 128)	7,372,928
dropout_15 (Dropout)	(None, 128)	0
dense_9 (Dense)	(None, 15)	1,935

Total params: 22,241,775 (84.85 MB)

Trainable params: 7,413,839 (28.28 MB)

Non-trainable params: 256 (1.00 KB)

Optimizer params: 14,827,680 (56.56 MB)

10. Project Scope and Future Work

- Extend to more animal classes and wild species.
- Deploy the model using a Flask or Streamlit app.
- Integrate with mobile camera or real-time video detection.
- Use transfer learning with ResNet, VGG, or MobileNet for better results.

11. Conclusion

- This project successfully demonstrated the classification of animals using a CNN-based deep learning model. The trained model achieves high accuracy and can be further improved for real-world applications like animal surveillance and educational tools.

12. References

- TensorFlow Documentation
- Keras API
- Towards Data Science Articles on CNN
- Kaggle Datasets