Introduction and Problem statement:

Have you ever wondered if a computer could watch a wildlife video and tell you what animals it sees? In this project, we set out to do exactly that — build a machine learning system that can automatically analyze videos, detect if an animal is present in each frame, and, if so, identify what kind of animal it is.

The main goal was to process a video frame-by-frame using two neural networks: one to determine whether an animal is present, and another to classify the type of animal (such as a cat, dog, or fox). The project combines ideas from computer vision, deep learning, and practical deployment of models to real-world video data.

Methodology and Data:

We used a two-stage neural network approach:

1. **Animal Presence Detection Model**
   This model is a simple convolutional neural network (CNN) trained to classify images as either "animal" or "no animal". It was trained using a custom-labeled dataset with folders for animal images and images without animals.

2. **Animal Classification Model**
   The second model is a transfer learning model based on MobileNetV2, trained on 90 animal classes. This model only runs when the first model detects an animal in a frame.

Preprocessing

- All frames were resized to match the input size expected by the models (128×128 for the presence model and 96×96 for the classification model).

- Pixel values were normalized to fall between 0 and 1.

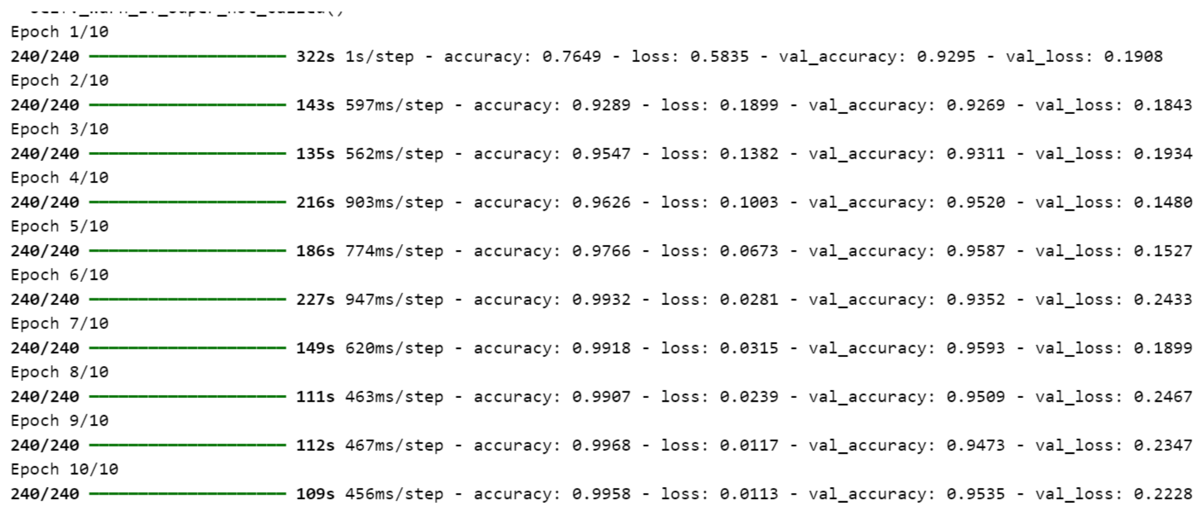
- Videos were read frame-by-frame using OpenCV

Data

- Animals data set
  - https://www.kaggle.com/datasets/iamsouravbanerjee/animal-image-dataset-90-different-animals
- Non animal data set
  - https://www.kaggle.com/datasets/adityajn105/flickr8k

Results

Animal presence model:



```
  self._warn_if_super_not_called()
Epoch 1/10
240/240 ━━━━━━━━━━━━━━━━━━━━ 322s 1s/step - accuracy: 0.7649 - loss: 0.5835 - val_accuracy: 0.9295 - val_loss: 0.1908
Epoch 2/10
240/240 ━━━━━━━━━━━━━━━━━━━━ 143s 597ms/step - accuracy: 0.9289 - loss: 0.1899 - val_accuracy: 0.9269 - val_loss: 0.1843
Epoch 3/10
240/240 ━━━━━━━━━━━━━━━━━━━━ 135s 562ms/step - accuracy: 0.9547 - loss: 0.1382 - val_accuracy: 0.9311 - val_loss: 0.1934
Epoch 4/10
240/240 ━━━━━━━━━━━━━━━━━━━━ 216s 903ms/step - accuracy: 0.9626 - loss: 0.1003 - val_accuracy: 0.9520 - val_loss: 0.1480
Epoch 5/10
240/240 ━━━━━━━━━━━━━━━━━━━━ 186s 774ms/step - accuracy: 0.9766 - loss: 0.0673 - val_accuracy: 0.9587 - val_loss: 0.1527
Epoch 6/10
240/240 ━━━━━━━━━━━━━━━━━━━━ 227s 947ms/step - accuracy: 0.9932 - loss: 0.0281 - val_accuracy: 0.9352 - val_loss: 0.2433
Epoch 7/10
240/240 ━━━━━━━━━━━━━━━━━━━━ 149s 620ms/step - accuracy: 0.9918 - loss: 0.0315 - val_accuracy: 0.9593 - val_loss: 0.1899
Epoch 8/10
240/240 ━━━━━━━━━━━━━━━━━━━━ 111s 463ms/step - accuracy: 0.9907 - loss: 0.0239 - val_accuracy: 0.9509 - val_loss: 0.2467
Epoch 9/10
240/240 ━━━━━━━━━━━━━━━━━━━━ 112s 467ms/step - accuracy: 0.9968 - loss: 0.0117 - val_accuracy: 0.9473 - val_loss: 0.2347
Epoch 10/10
240/240 ━━━━━━━━━━━━━━━━━━━━ 109s 456ms/step - accuracy: 0.9958 - loss: 0.0113 - val_accuracy: 0.9535 - val_loss: 0.2228
```

Animal Classification model:

0.76% Accuracy

If we had more time to run our final model to test for accuracy we definitely would, however we are generally experiencing a pretty good success rate with a small sample size for testing.

Discussion

One of the biggest challenges was managing different input sizes for each model. We had to preprocess each frame twice — once for each model — to ensure the input shapes matched. We also noticed that false positives were more common in scenes with clutter or shadows, which sometimes tricked the presence model into thinking an animal was present.

Training the classifier on a limited dataset meant the model occasionally confused similar animals, like dogs and foxes. This could be improved with more diverse training data or by adding temporal consistency (tracking animals across frames).

Comparing our results to similar work, we found that this two-stage approach (presence detection → classification) is commonly used in real-time surveillance and animal tracking applications.

Conclusion

This project shows that it's possible to build a relatively simple but effective system for detecting and classifying animals in videos using neural networks. With more data and model fine-tuning, it could be expanded to more species or integrated into wildlife monitoring systems. The separation of detection and classification keeps the system efficient and flexible for real-world use.