Animal Image Classification using CNN & Transfer Learning

Shravani Bande

Project Description

This project explores the use of Convolutional Neural Networks (CNNs) and Transfer Learning (MobileNetV2) to classify animal images into 15 categories such as Cat, Dog, Lion, Elephant, etc. It documents the entire learning journey — from basic CNN to an optimized pretrained model.

Dataset Structure

```
dataset/
 train/
     Cat/
     Dog/
     Elephant/
     Lion/
     ... (15 total classes)
```

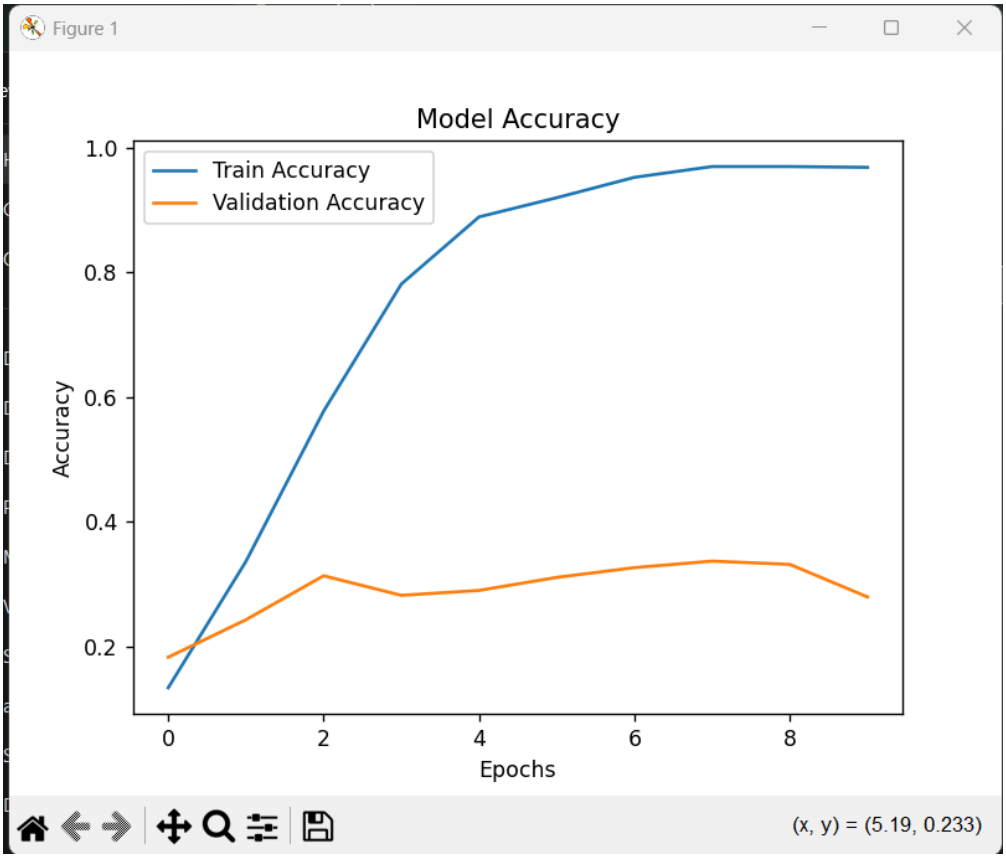
First Attempt: main.py (Basic CNN)

Model Summary: A very basic CNN model with no dropout or augmentation.

Code:

```
model = Sequential([
    Conv2D(32, (3,3), activation='relu', input_shape=(150,150,3)),
    MaxPooling2D(2,2),
    Flatten(),
    Dense(64, activation='relu'),
    Dense(num_classes, activation='softmax')
])
```

Result: Very low validation accuracy (30%), training accuracy high. The model overfit quickly and didn't generalize well.



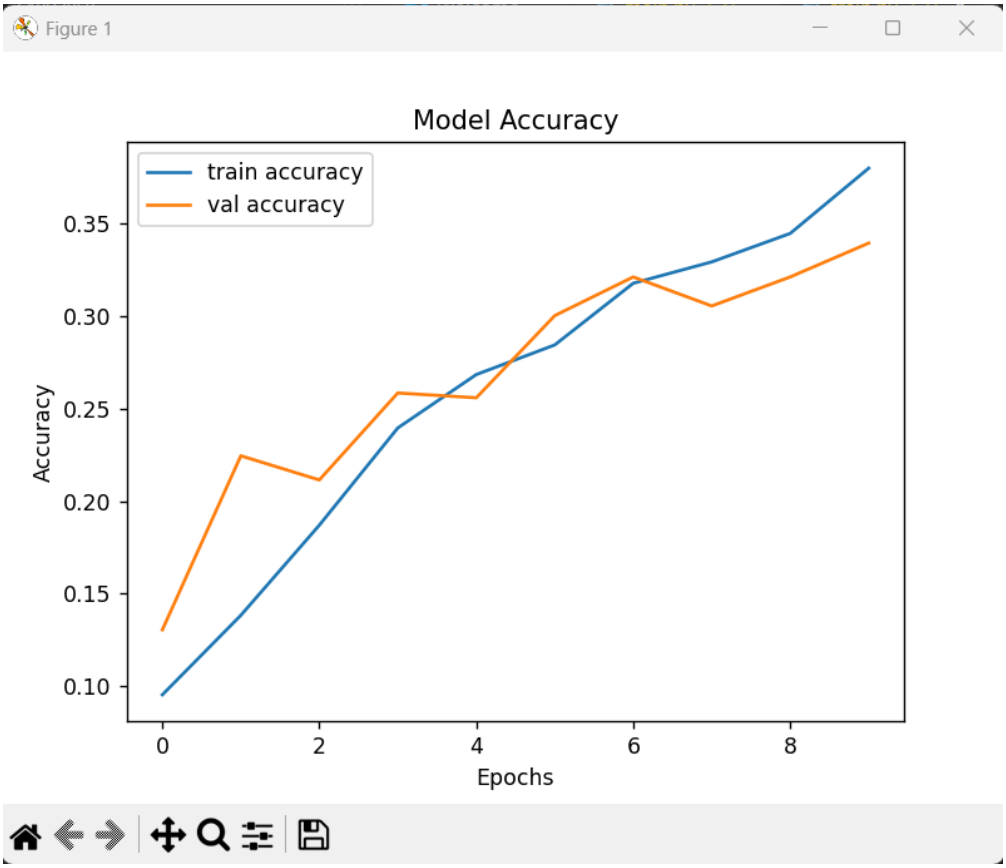
Second Attempt: train.py (Improved CNN with Dropout)

Model Summary: Added dropout and data augmentation.

Code:

```
model = Sequential([
    Conv2D(32, (3,3), activation='relu', input_shape=(150,150,3)),
    MaxPooling2D(2,2),
    Dropout(0.3),
    Conv2D(64, (3,3), activation='relu'),
    MaxPooling2D(2,2),
    Dropout(0.3),
    Flatten(),
    Dense(128, activation='relu'),
    Dropout(0.5),
    Dense(num_classes, activation='softmax')
])
```

Result: Slight improvement in validation accuracy (35%) and less overfitting, but still far from ideal.



Problems Encountered

- **Script Execution Blocked:** Error while activating virtual environment.
- **OpenCV Error:** `ModuleNotFoundError: No module named 'cv2'`
- **Input Shape Mismatch:** Predicted image size didn't match model expectation.
- **Large File Error on GitHub:** `tensorflow/.pyd file > 900MB` blocked push.

Final Solution: train_transfer.py (MobileNetV2)

Model Summary: Used MobileNetV2 as base, added dense layers, frozen convolution layers.

Code:

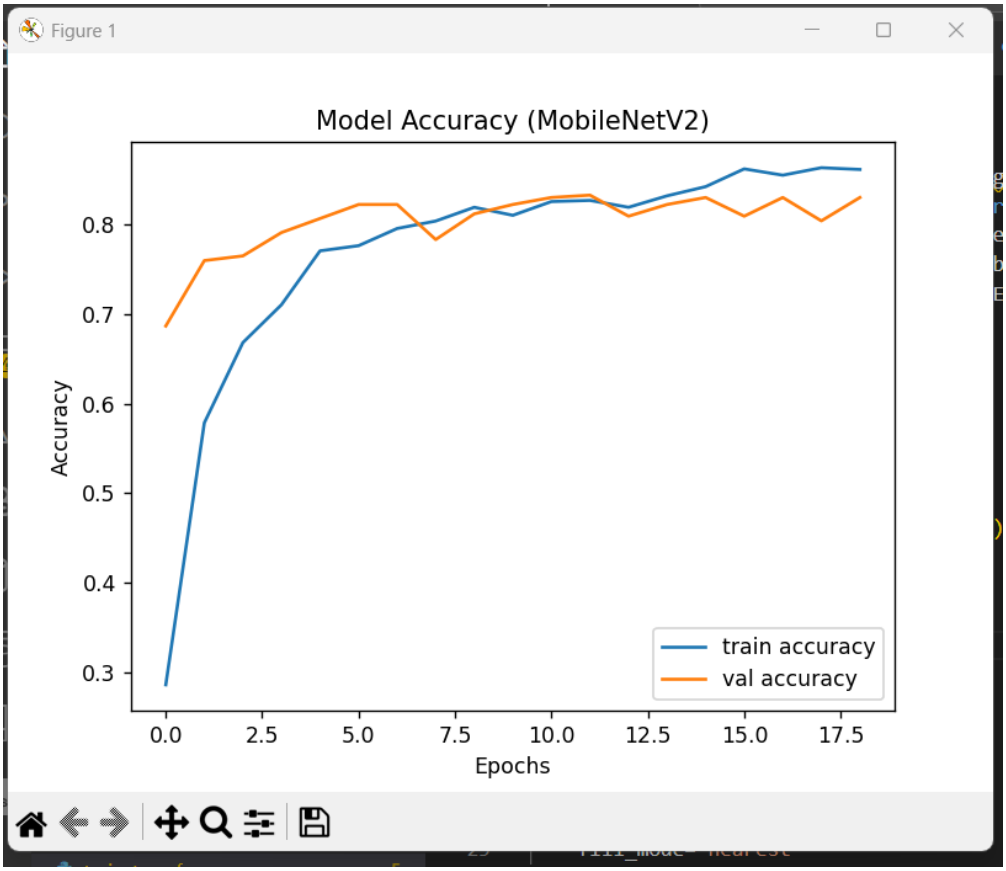
```
base_model = MobileNetV2(input_shape=(224, 224, 3), include_top=
    False, weights='imagenet')
base_model.trainable = False

x = base_model.output
x = GlobalAveragePooling2D()(x)
x = Dropout(0.3)(x)
```

```
x = Dense(128, activation='relu')(x)
x = Dropout(0.5)(x)
output = Dense(num_classes, activation='softmax')(x)

model = Model(inputs=base_model.input, outputs=output)
```

Training Accuracy: 88%
Validation Accuracy: 85%



Prediction Sample

```
Enter the image filename (e.g., test_dolphin.jpg):
 Prediction: Dolphin
```

How to Run the Project

```
# Clone the repo
git clone https://github.com/savi-08/animal-image-classification.git
cd animal-image-classification

# Setup virtual environment
```

```
python -m venv venv
venv\Scripts\activate
pip install -r requirements.txt

# Train model
python train_transfer.py

# Make predictions
python predict.py
```

Conclusion

This project highlights a progressive learning journey. From an overfitting CNN to a robust pretrained MobileNetV2-based model, it shows how experimentation, debugging, and optimization lead to better performance.

Author

Shravani Bande
Akola, Maharashtra
GitHub: `https://github.com/savi-08`