Title: Image Classification Using CNN and Pre-Trained Models on Oxford-IIIt pet database

Introduction

My goal for this project was to build an efficient image classification model using CNN (convolutional neural networks) and pre-trained models such as VGG16, VGG19 and ResNet50. The dataset that I used for this model is the oxford_iiit_pet database, which has 37 classes with each class containing 200 images with different breeds of cats and dogs. In the project, there is data processing, data augmentation and model evaluation to achieve high accuracy to predict the breed of the dog or the cat on the given image.

Problem Description

Image classification for this database is really challenging to do as it is dealing with multiple classes and similar categories. The database includes 37 classes of pets (dogs and cats), which makes the classification complex because of the subtle differences between some of the breeds. The aim of this project is to develop a machine learning model that is going to accurately identify the breed of an image.

Solution

The solution involves following steps:

DATA PREPROCESSING

1. Enabled the GPU and mixed precision for faster training
2. Extracted the images and annotations of the dataset and put them in different directories
3. Labeled each image with a corresponding breed with indexes (0, 1, 2, 3, 4,…)
4. Split the dataset into training and validation sets (80-20 ratio)
5. Applied data augmentation to increase the variability of the training data and improve the generalization
    - Rotation (up to 30 degrees)
    - Width and height shifts (up to 20%)
    - Zoom (up to 30%)
    - Horizontal flipping
    - Brightness change (0.8 to 1.2)

MODEL IMPLEMENTATION

1. **Basic CNN Model**: Designed a convolutional neural network with three convolutional layers, max-pooling layers and dense layers for classification
2. **Pre-Trained Models**:
    - VGG16, VGG19 and ResNet50 were used as a pre-trained model to show the differences of how each of them helped the image classification. They used the pre-trained weights from the **ImageNet** dataset
    - Fine-tuned the models by freezing the base layers and adding the dense layers
    - A lower learning rate of 1e-4 (0.0001) was also used for fine-tuning to prevent overfitting

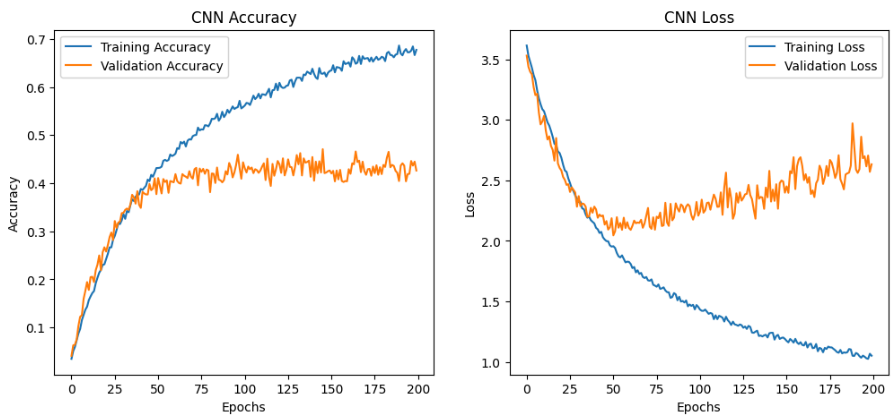
TRAINING AND EVALUATION

1. **Early Stopping**: It is implemented to monitor **validation accuracy** and to stop overfitting the data. Training is stopped if validation accuracy is not improving after 5 consecutives epochs.
2. **Training Epochs**:
    - **Basic CNN**: 200 epochs
    - **Pre-trained models**: 100 epochs
3. Evaluated the models on the validation set and compared their performance

RESULTS

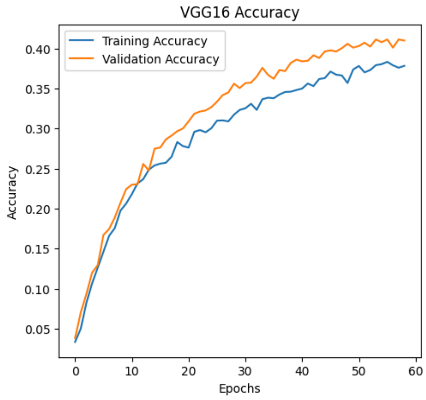
Basic CNN Accuracy and Loss

- **Validation Loss**: 2.63
- **Validation Accuracy**: 42.63%



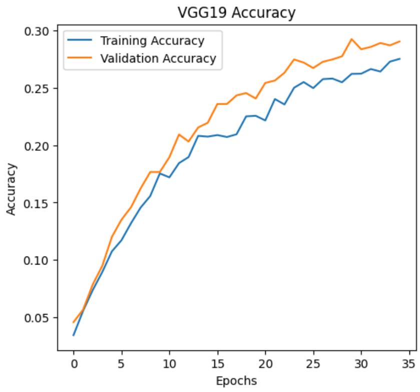
VGG16 Accuracy and Loss

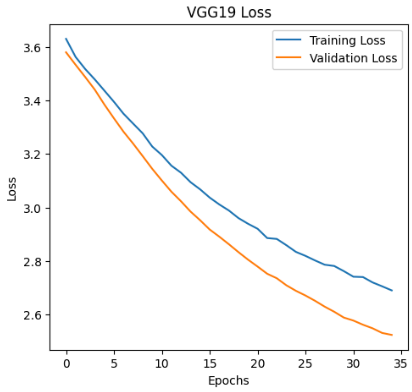
- **Validation Loss**: 2.08
- **Validation Accuracy**: 41.13%



VGG19 Accuracy and Loss

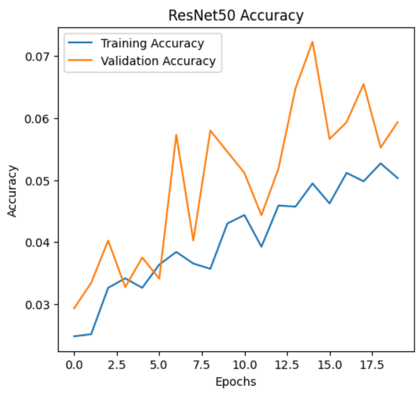
- **Validation Loss**: 2.59
- **Validation Accuracy**: 29.26%

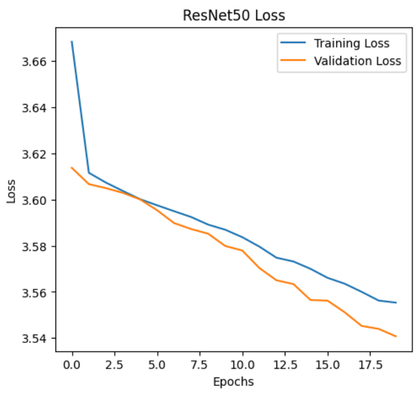




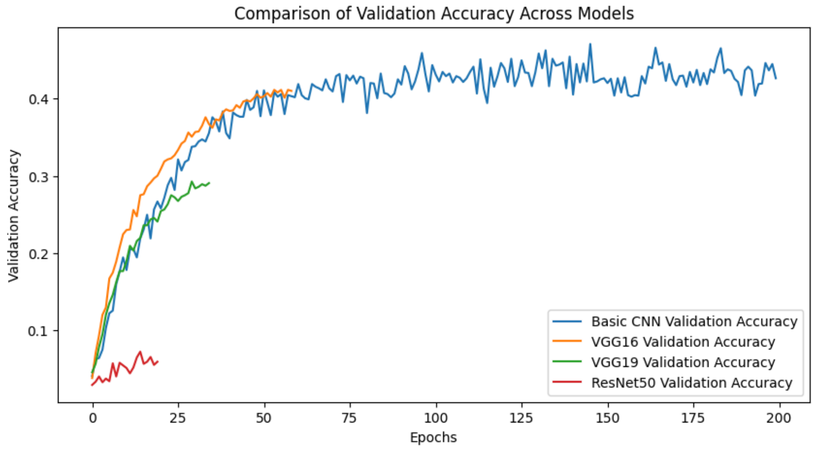
ResNet50 Accuracy and Loss

- **Validation Loss**: 3.56
- **Validation Accuracy**: 7.23%





Comparison of Validation Accuracy from all Models

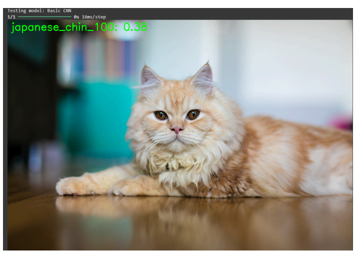


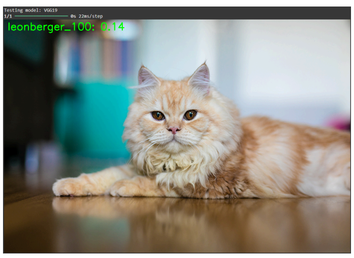
Analysis of Results

- **Basic CNN**: It achieved OK performance with validation accuracy of ~43% after 200 epochs showing that the model learned some of the features effectively
- **VGG16**: It performed similar to Basic CNN but with slightly lower accuracy. This means that the model might not have the dataset features as expected
- **VGG19**: It underperformed compared to VGG16 and Basic CNN, which may be due to overfitting or lack of data
- **ResNet50**: It showed very low accuracy, meaning the model failed to generalize

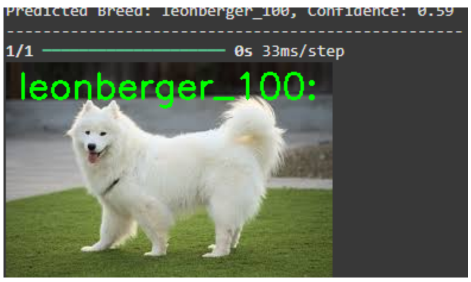
Example Predictions

- **Image1**: cat1.jpg





- **Image2**: dog1.jpg



Summary

The project was successful in showing use of both custom built CNN and pre-trained models for this image classification task. However, the pre-trained models did not perform as expected. The results highlight the importance of making the model for the particular dataset and it indicates that a simpler model can sometimes get better results for the task.

What can be improved?

- Increase the dataset size with more images to improve generalization
- Use transfer learning with unfreezing specific layers for fine-tuning
- Do other pre-trained models such as EfficientNet