PROJECT ASSIGNMENT

Dogs vs. Cats Image Classification using Convolutional Neural Networks

ISE741

DEEP LEARNING

by

VASHISTA A N (1MS20IS131)

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

RAMAIAH INSTITUTE OF TECHNOLOGY

August 2023

Problem Statement:

Cats vs Dogs:

The cats vs dogs is a good project to start as a beginner in deep learning. You can build a model that takes an image as input and determines whether the image contains a picture of a dog or a cat.

Abstract:

In the realm of computer vision, image classification stands as a cornerstone task. With the proliferation of deep learning techniques, particularly Convolutional Neural Networks (CNNs), the accuracy and efficiency of image classification have witnessed significant advancements. This project delves into harnessing the capabilities of CNNs to distinguish between images of dogs and cats.

Utilizing the Dogs vs. Cats dataset sourced from Kaggle, which comprises 25,000 labeled images, a deep learning model was architected and trained to classify these images. The dataset was systematically partitioned into training (80%) and validation (20%) sets. The designed CNN model incorporated multiple convolutional layers paired with max-pooling layers, followed by dense layers to perform the classification.

Data augmentation techniques were employed during training to enhance the model's generalization capabilities. The model was optimized using the Adam optimizer and was trained with a binary cross-entropy loss function.

Post-training, the model's performance was evaluated on the validation set, yielding promising results in terms of accuracy and other metrics. The project not only underscores the efficacy of CNNs in image classification tasks but also provides a foundational framework for further exploration into more complex image classification challenges.

Objective:

To design and implement a deep learning model that can accurately classify images as either a dog or a cat.

Introduction:

Image classification is a fundamental task in computer vision. With the advent of deep learning and especially Convolutional Neural Networks (CNNs), the accuracy of image classification tasks has seen significant improvement. This project aims to harness the power of CNNs to classify images of dogs and cats.

Dataset:

The dataset used is the Dogs vs. Cats dataset from Kaggle. It consists of 25,000 labeled images of dogs and cats. Each image is labeled either as a dog or a cat.

Methodology:

1. **Data Preparation**:
    a. The dataset is organized into separate directories for training and validation.
    b. 80% of the images are used for training, and 20% are used for validation.
    c. Within the training and validation directories, separate subdirectories are created for each class (dogs and cats).

```
import os
import shutil
import numpy as np
from sklearn.model_selection import train_test_split
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense
from tensorflow.keras.preprocessing.image import ImageDataGenerator, load_img, img_to_array
✓ 0.0s
```

```python
def define_model():
    model = Sequential()
    model.add(Conv2D(32, (3, 3), input_shape=(64, 64, 3), activation='relu'))
    model.add(MaxPooling2D(pool_size=(2, 2)))
    model.add(Conv2D(32, (3, 3), activation='relu'))
    model.add(MaxPooling2D(pool_size=(2, 2)))
    model.add(Flatten())
    model.add(Dense(units=128, activation='relu'))
    model.add(Dense(units=1, activation='sigmoid'))
    model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
    return model
```
✓ 0.0s

2. **Model Architecture**:
    a. The model is a Convolutional Neural Network (CNN) built using TensorFlow's Keras API.
    b. The model consists of two sets of Conv2D and MaxPooling2D layers, followed by a Flatten layer and two Dense layers.

3. **Training**:
    a. Data Augmentation is used to artificially increase the size of the training dataset.
    b. The model is trained using the Adam optimizer and binary cross-entropy loss.

4. **Evaluation:** A function is implemented to classify a user-provided image.

```python
def train_model(model, train_dir, validation_dir):
    train_datagen = ImageDataGenerator(
        rescale=1./255,
        shear_range=0.2,
        zoom_range=0.2,
        horizontal_flip=True)

    test_datagen = ImageDataGenerator(rescale=1./255)

    train_generator = train_datagen.flow_from_directory(
        train_dir,
        target_size=(64, 64),
        batch_size=32,
        class_mode='binary')

    validation_generator = test_datagen.flow_from_directory(
        validation_dir,
        target_size=(64, 64),
        batch_size=32,
        class_mode='binary')

    model.fit(
        train_generator,
        steps_per_epoch=len(train_generator),
        epochs=25,
        validation_data=validation_generator,
        validation_steps=len(validation_generator))
```

✓ 0.0s

```python
def classify_image(model, image_path):
    test_image = load_img(image_path, target_size=(64, 64))
    test_image = img_to_array(test_image)
    test_image = np.expand_dims(test_image, axis=0)
    result = model.predict(test_image)

    if result[0][0] == 1:
        prediction = 'dog'
    else:
        prediction = 'cat'
    return prediction
```

✓ 0.0s

```
Found 20000 images belonging to 2 classes.
Found 5000 images belonging to 2 classes.
Epoch 1/25
625/625 [==============================] - 261s 415ms/step - loss: 0.6196 - accuracy: 0.6518 - val_loss: 0.5629 - val_accuracy: 0.6980
Epoch 2/25
625/625 [==============================] - 105s 167ms/step - loss: 0.5371 - accuracy: 0.7332 - val_loss: 0.5425 - val_accuracy: 0.7324
Epoch 3/25
625/625 [==============================] - 103s 164ms/step - loss: 0.4987 - accuracy: 0.7511 - val_loss: 0.4564 - val_accuracy: 0.7804
Epoch 4/25
625/625 [==============================] - 109s 174ms/step - loss: 0.4670 - accuracy: 0.7742 - val_loss: 0.5007 - val_accuracy: 0.7546
Epoch 5/25
625/625 [==============================] - 103s 165ms/step - loss: 0.4455 - accuracy: 0.7888 - val_loss: 0.4789 - val_accuracy: 0.7798
Epoch 6/25
625/625 [==============================] - 100s 160ms/step - loss: 0.4279 - accuracy: 0.7991 - val_loss: 0.4200 - val_accuracy: 0.8018
Epoch 7/25
625/625 [==============================] - 109s 175ms/step - loss: 0.4103 - accuracy: 0.8134 - val_loss: 0.4459 - val_accuracy: 0.7916
Epoch 8/25
625/625 [==============================] - 103s 164ms/step - loss: 0.3941 - accuracy: 0.8205 - val_loss: 0.4016 - val_accuracy: 0.8212
Epoch 9/25
625/625 [==============================] - 102s 163ms/step - loss: 0.3840 - accuracy: 0.8265 - val_loss: 0.4361 - val_accuracy: 0.7994
Epoch 10/25
625/625 [==============================] - 104s 166ms/step - loss: 0.3670 - accuracy: 0.8325 - val_loss: 0.4621 - val_accuracy: 0.7916
Epoch 11/25
625/625 [==============================] - 103s 164ms/step - loss: 0.3519 - accuracy: 0.8428 - val_loss: 0.4178 - val_accuracy: 0.8166
Epoch 12/25
...
Epoch 24/25
625/625 [==============================] - 115s 184ms/step - loss: 0.2045 - accuracy: 0.9165 - val_loss: 0.4641 - val_accuracy: 0.8344
Epoch 25/25
625/625 [==============================] - 109s 174ms/step - loss: 0.2045 - accuracy: 0.9144 - val_loss: 0.5006 - val_accuracy: 0.8270
```

Process Flow:

1. **Data Collection**: Obtain the Dogs vs. Cats dataset from Kaggle.
2. **Data Preparation**: Organize the dataset into training and validation sets.
3. **Model Definition**: Define the architecture of the CNN.
4. **Model Training**: Train the model using the training set.
5. **Model Evaluation**: Evaluate the model's performance on the validation set.
6. **User Input Classification**: Classify user-provided images.

Results:

The model's performance can be evaluated based on its accuracy on the validation set. For example, after training, the model achieved an accuracy of 90% on the validation set (this is a placeholder; you should replace it with your actual result). The accuracy metric indicates the proportion of correctly classified images out of the total images in the validation set.

```python
from sklearn.metrics import confusion_matrix, classification_report
def evaluate_model(model, validation_dir):
    test_datagen = ImageDataGenerator(rescale=1./255)
    validation_generator = test_datagen.flow_from_directory(
        validation_dir,
        target_size=(64, 64),
        batch_size=32,
        class_mode='binary',
        shuffle=False)
    true_labels = validation_generator.classes

    predictions = model.predict(validation_generator, steps=len(validation_generator))
    predicted_labels = [1 if pred > 0.5 else 0 for pred in predictions]

    conf_matrix = confusion_matrix(true_labels, predicted_labels)
    print("Confusion Matrix:")
    print(conf_matrix)

    class_report = classification_report(true_labels, predicted_labels, target_names=["cat", "dog"])
    print("\nClassification Report:")
    print(class_report)

evaluate_model(model, validation_dir)
```

✓ 11.1s

```
Found 5000 images belonging to 2 classes.
157/157 [==============================] - 11s 65ms/step
Confusion Matrix:
[[1945  555]
 [ 310 2190]]

Classification Report:
              precision    recall  f1-score   support

         cat       0.86      0.78      0.82      2500
         dog       0.80      0.88      0.84      2500

    accuracy                           0.83      5000
   macro avg       0.83      0.83      0.83      5000
weighted avg       0.83      0.83      0.83      5000
```

```python
image_path = 'test1/test1/12471.jpg'
print(f"The image is of a {classify_image(model, image_path)}")
```

✓ 0.2s

```
1/1 [==============================] - 0s 108ms/step
The image is of a dog
```

Conclusion:

The project demonstrates the effectiveness of Convolutional Neural Networks in image classification tasks. With appropriate data augmentation, optimization, and model architecture, high accuracy can be achieved in distinguishing between images of dogs and cats.