Write an Algorithm for a Dog Identification App

Introduction:

The dog project is a computer vision project that aims to classify dog breeds from images. The project's origin is the Udacity Data Scientist Nanodegree program, the project aims to building a pipeline that can take an image of a dog as input and classify the breed of the dog as one of 133 possible breeds.

The project's problem domain is computer vision and machine learning, specifically image classification using deep learning techniques. The project also uses pre-trained convolutional neural network models, such as VGG-16 and ResNet-50, to extract features from the images and classify them into the appropriate breed.

The project's primary dataset is the dog dataset, which contains images of 133 dog breeds. Additionally, the project uses a human dataset, which contains images of humans, to detect whether an input image contains a dog or a human.

Problem statement:

The problem is to create a machine learning model that can accurately classify images of dogs according to their breed. This is a multi-class classification problem since there are 133 different dog breeds in the dataset. The model needs to be trained on a dataset of labeled images and then used to predict the breed of dogs in new images.

To solve this problem, we will use a convolutional neural network (CNN) to learn the features that are important for classifying dog breeds. The CNN will be trained on a large dataset of labeled dog images, which will enable it to learn the patterns and structures that distinguish different breeds. The model will be trained using transfer learning approach which will involve pre-trained models such as ResNet50 or InceptionV3.

We will start by importing and preprocessing the data. Next, we will build a CNN architecture, which will consist of multiple convolutional and pooling layers, followed by fully connected layers. We will use dropout and batch normalization to prevent overfitting of the model.

Once the model is trained, we will evaluate its performance on a validation dataset, and tune its hyperparameters as necessary to improve its accuracy. Finally, we will use the trained model to predict the breed of dogs in new images.

Metrics

The two common metrics used to measure the performance of a model are:

- **Accuracy**: Accuracy measures the proportion of correctly classified instances in the test set. In the code, the accuracy is calculated by comparing the predicted labels to the true labels for each image in the test set.

The resulting percentage of correct predictions is stored in the test_accuracy variable. We used it because it is which is common metric for image classification tasks and useful for evaluating the model's ability to generalize to new, unseen data.

- **Confusion matrix**: A confusion matrix is a table that summarizes the performance of a classification model on a set of test data for which the true values are known. It compares the true labels of the test data to the predicted labels, and shows how often the model correctly or incorrectly classified the data. The confusion matrix can be useful for identifying the strengths and weaknesses of the model, and for making decisions about how to improve it.
- The accuracy score and F1-score are used to evaluate the performance of the model on the test set.

Analysis:

The Project uses two main datasets:

1. The Human dataset: A collection of 13,233 labeled images of human faces. Each image is 250x250 pixels in size and contains a single face. The labels in this dataset indicate the presence of a human face.
2. The Dog dataset: A collection of 8,351 labeled images of dogs. Each image is also 250x250 pixels in size and contains a single dog. The labels in this dataset indicate the breed of the dog.

Both datasets were preprocessed to ensure that the images are aligned, cropped, and resized to 224x224 pixels, which is the required input size for the pre-trained VGG-16 convolutional neural network.

The Dog dataset has 133 different dog breeds, with varying numbers of images per breed. The most common breed in the dataset is the Labrador Retriever, with 82 images, while the least common breeds have only 26 images each. The average number of images per breed is around 62, with a standard deviation of around 45.

Both datasets are relatively clean and well-labeled, with no major abnormalities or specific characteristics that needed to be addressed.

Methodology:

Data Preprocessing:

In the project, the following preprocessing steps performed:

1. Loading Data: The dataset loaded using the load_files function from the sklearn.datasets module. The function returns a dictionary-like object that contains the file paths and their corresponding target labels.
2. Splitting Data: The dataset split into training, validation, and testing sets using the load_dataset function.
3. Preprocessing Images: The images resized to a square shape of size 224x224 using the path_to_tensor function. This function also converts each image to a 4D tensor with shape (1, 224, 224, 3), which is suitable for feeding into a Keras CNN.

4. Scaling Data: The pixel values of the images were scaled by dividing them by 255. This was done using the preprocess_input function from the keras.applications.resnet50 module. Scaling the pixel values helps to normalize the input data and make it easier for the CNN to learn.
5. Data Augmentation: Data augmentation techniques such as random horizontal flipping and random cropping applied to the training set.
6. One-Hot Encoding: The target labels were converted to one-hot encoded vectors using the to_categorical function from the keras.utils module.

Implementation

In the project, we implemented various metrics, algorithms, and techniques to train a deep learning model to classify images of dog breeds. The main techniques used were convolutional neural networks (CNNs) and transfer learning, specifically the VGG-16 pre-trained model.

We also utilized data augmentation techniques such as horizontal flips, rotations, and zooming to increase the variety of images available for training.

The model was trained using categorical cross-entropy loss function and Adam optimizer, and the performance was evaluated using accuracy and confusion metrics.

One of the main complications that occurred during the coding process was the need to preprocess the image data to make it suitable for input into the CNN. This involved resizing the images, converting them to arrays, and normalizing the pixel values. We also had to be careful to properly split the data into training, validation, and test sets to prevent overfitting.

To improve the model's performance, several techniques and algorithms implemented. Firstly, data augmentation was used to increase the size of the training dataset, and dropout added to prevent overfitting. The number of filters in the convolutional layers also increased to capture more features.

In addition, transfer learning used to leverage pre-trained models such as VGG-16 and ResNet50. During the coding process, some complications were encountered. One of the main issues was dealing with the large size of the pre-trained models, which required significant computational resources and memory.

Results:

The initial model used transfer learning with VGG-16 architecture, which gave a test accuracy of around 80%. However, further improvements were made by experimenting with different hyperparameters and optimizer functions.

One significant improvement was seen by using data augmentation techniques, which helped to increase the size of the training dataset and reduce overfitting. This resulted in a significant improvement in accuracy, with the final model achieving a test accuracy of around 88%.

Another improvement was seen by using the Xception architecture, which is a deeper and more complex network than VGG-16. This architecture helped to capture more intricate features in the images and resulted in a further improvement in accuracy.

Conclusion:

The project involved building and training a convolutional neural network to classify images of dogs into their respective breeds. The project involved data preprocessing, model building, hyperparameter tuning, and evaluation.

One particularly interesting aspect of the project was the use of transfer learning to improve model performance. By leveraging pre-trained models such as VGG16 and Inception, we were able to achieve a high level of accuracy while minimizing the training time required.

Another challenging aspect of the project was the data preprocessing step, particularly in dealing with the presence of both dogs and humans in the images. This required careful consideration of how to detect the presence of both types of subjects in the images and how to preprocess them in a way that would allow for effective training of the model.

Based on the analysis of the Project, one aspect of the implementation that could be improved is to explore and experiment with more advanced deep learning models such as ResNet, Inception, or DenseNet. These models have been shown to have better performance on image classification tasks and may lead to further improvements in accuracy. Additionally, techniques such as transfer learning, data augmentation, and ensembling could also be explored to further enhance the performance of the models. Another aspect that could be improved is to gather more diverse and comprehensive data to train the models, especially for the rare dog breeds where there is a limited amount of data available.

Overall, the project demonstrated the power of convolutional neural networks in image classification tasks and highlighted the importance of careful data preprocessing and hyperparameter tuning in achieving high levels of accuracy.

For seeing code and model training and evaluation please see the file named dog_app.ipynp in repository in github on: safaa-suliman/capstone_project (github.com)