CS412 Homework 4 Report

Uğur ÖZTUNÇ 28176

Colab Notebook Link

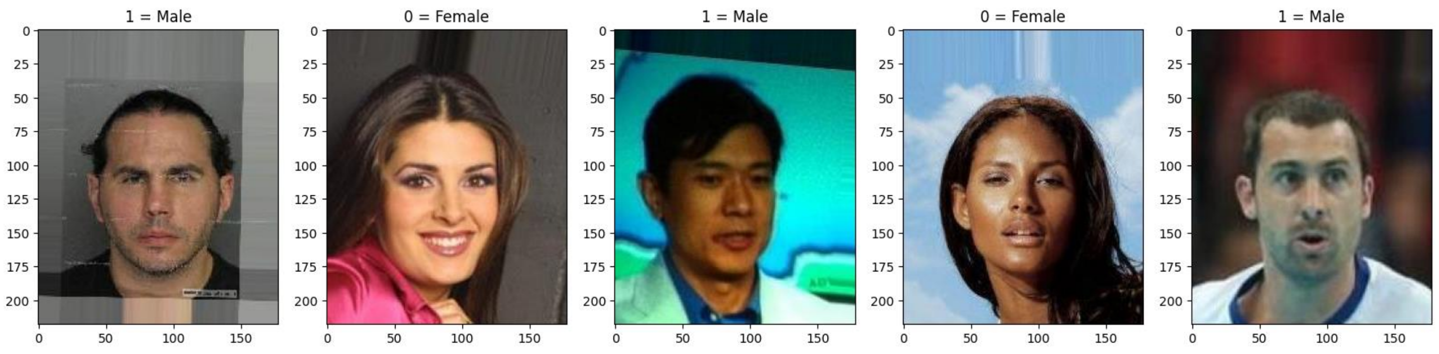
https://colab.research.google.com/drive/1o3BUVPVWQhohq9bkRQKwVHjmfTFRIe0f

Introduction

In this assignment, we were asked to build a model for gender classification using transfer learning. Transfer learning is a technique that utilizes pre-trained models, mostly models that are trained with a lot of data for long times, to solve new tasks with limited training data. In this case, we were asked to utilize the VGG-16 architecture, a deep neural network widely used for image recognition, and adapted it to classify the gender of individuals in face images.

Dataset

For this task, we were provided with a subset of the CelebA dataset, which contains over 200,000 celebrity images annotated with various attributes. The subset used consisted of 30,000 RGB face images. The gender attribute was extracted from the dataset for the gender classification task. Here are five randomly selected images from the dataset:

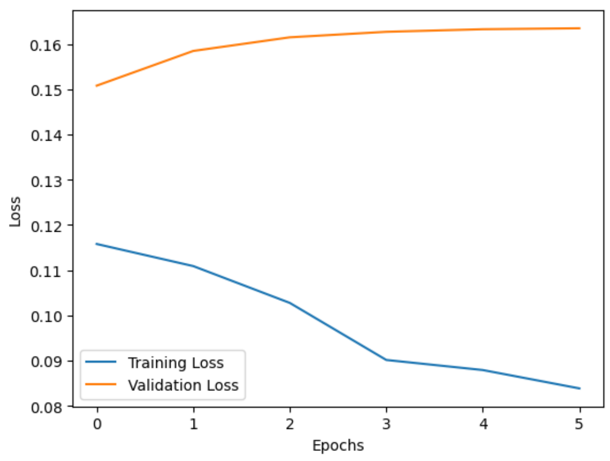


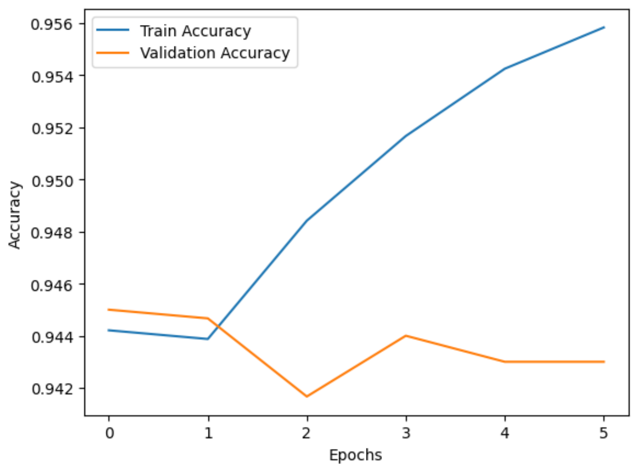
Methodology

The transfer learning model is created by utilizing the pre-trained VGG-16 model. The VGG-16 model was originally trained on the large-scale ImageNet dataset, enabling it to learn rich representations of images. The original head of the VGG-16 model was replaced to adapt it for binary gender classification. Firstly, the dataset is loaded and split into train (80% = 24000 images), validation (10% = 3000 images), and test sets (10% = 3000 images). Then, data augmentation and preprocessing are performed using the ImageDataGenerator class. The input images were resized to (224, 224) to match the input size expected by the VGG-16 model. The VGG-16 model is loaded and its layers are frozen to retain the pre-trained weights. A new classification head is added on top of the base model, consisting of dense layers including dropout regularization, and sigmoid activation layer to output the gender prediction. The model is compiled with SDG optimizer and binary cross entropy loss function. Training is performed using the train and validation generators, with early stopping and learning rate reduction callbacks. The model's performance is evaluated using the training and validation accuracy and loss curves. Finally, the model is tested on unseen test set, and different threshold values are tested to find the optimal threshold for gender classification accuracy. The best threshold and corresponding test accuracy are reported as the final results.

Experiments

The experiment was conducted using a learning rate of 0.001 and a batch size of 16 for a total of 10 epochs of training. The initial accuracy at the beginning of training was approximately 0.8, and it gradually increased to around 0.94 by the end of the training period. Although precise values for different parameter configurations could not be recorded due to limitations with Google Colab's GPU acceleration and runtime resources, it is observed that tuning the parameters resulted in an improvement in the performance of the model. During the tuning phase, the accuracy increased from 0.94 to approximately 0.96 over 6 epochs (early stopped) of training. Here are the training curves:





However, upon evaluating the model on the test set, the accuracy dropped significantly to around 0.52. This indicates that the model may have become overfit to the training set and fails to generalize well to unseen data.

- I want to emphasize again that the inconsistent behavior of Google Colab's GPU acceleration (such as blocking me from using the hardware acceleration feature for hours due to long times of using GPU runtime) hindered the recording of precise values for different parameter combinations. This limitation impacted the ability to perform a comprehensive analysis of the effects of varying the parameters.

Conclusion

In conclusion, the experiment involved training a model on a dataset of male and female facial images using transfer learning with VGG-16. By tuning the hyperparameters, including a learning rate of 0.001 and a batch size of 16, the model's performance improved during the training phase. However, when evaluating the model on the test set, the accuracy indicated a potential overfitting to the training data. This low accuracy on the test set compared to the training phase showed that there is a need for further investigation and adjustments to find a solution for the overfitting issue. For improving the model's performance, some additional steps can be taken, such as using regularization techniques or trying various combinations of hyperparameter values. These could help to mitigate the overfitting problem of the model and enhance the model's ability to generalize to unseen data.