REPORT - Assignment 1: Age Estimation and Gender Classification

Section 1: Introduction

The objective of this task is to utilize deep learning models to classify gender and estimate age based on facial photos. The assignment entails training two distinct models on the UTKFace dataset utilizing GPU on Google Colab: There are two options available: developing my own CNN architecture and training it from the beginning or adjusting a pre-existing model.

This code encompasses two primary objectives: training deep learning models to accurately determine a person's age and classify their gender using face images. The code encompasses many stages, including dataset loading, data preprocessing, model definition, training, and result visualization.

Section 2: My own CNN

Architecture

This CNN model is designed to process 128x128 RGB pictures. Features are extracted using four convolutional and max-pooling layers. Using fully connected layers, it classifies gender and estimates age. The design's convolution layers extract complex image features, while the max-pooling layers reduce dimensionality and computational load. The dropout layer reduces overfitting.

- **Parameters**

The formula for computing the number of parameters is as follows: (filter_height * filter_width * input_channels + 1) * filters. The phrase "+1" denotes the bias term.

- First convolution layer: (3*3*3+1)*32 = 896
- Second convolution layer: (3*3*32+1)*64 = 18,496
- Third convolution layer: (3*3*64+1)*128 = 73,856
- Fourth convolution layer: (3*3*128+1)*256 = 295,168

The parameter numbers for fully connected layers are:

For the first Dense layer after the Flatten layer: flattened_features * 256 + 256 (flattened_features is the number of features after the Flatten layer and its calculation depends on the output of the previous layers of the model.)

- For the second Dense layer: 256*128+128
- For gender output: 256 * 1 + 1
- For age output: 256 * 1 + 1

In order to determine the number of features following the flatten layer, it is important to ascertain the output size of the last convolution layer. The dimensions of the feature map following the last maximum pooling layer are (size/2)*(size/2)*256. The "size" refers to the output size of the preceding layer, and each max-pooling layer reduces the size by half.

- **Training**. How did you set the relevant hyperparameters? What is the reason for them? How long does the model train on each epoch? How long does the whole training take with GPU? Screenshot a part of the training output.

Hyperparameters are important factors that affect the training process of the model. In this model, hyperparameters such as learning rate, batch size, number of epochs and dropout rate need to be adjusted. These values directly affect the performance of the model and the most appropriate values are found through trial and error.

Training time for each epoch and how long it takes in total varies depending on the type of GPU used and the size of the dataset. Training with GPU is much faster than with CPU and significantly reduces training time on large data sets.

- **Performance**

To evaluate the performance of the model, learning curves of loss and accuracy values in the training and validation sets are plotted. These curves show how the model improves during the training process and whether it is overfitting.

This information can help you fill out the "Architecture" and "Education" sections of your assignment. For the performance part, it should be detailed based on the results obtained from training the model. You may need to provide more details about your model's configuration and training process.

Section 3: Pre-trained CNN

- **Architecture:**
  This challenge uses the VGG16 network, a pre-trained CNN model known for image recognition. The ImageNet-trained VGG16 model provides dependable feature detectors for image classification applications. For the gender classification task, the VGG16 network convolutional basis was fixed. ImageNet weights were not adjusted throughout training. A global average pooling layer and 256-neuron dense layer were added to the VGG16 basis. A dropout layer was added for regularization. The output layer has one neuron with a binary classification-specific sigmoid activation function.
  Only 131,585 of 14,846,273 model parameters were trainable. These trainable parameters are related to the VGG16 base's extra layers. The frozen convolutional basis prevented training of 14,714,688 parameters.

- **Training**:
  The hyperparameters were chosen to optimize model performance and reduce overfitting. Due to its adjustable learning rate, the Adam optimizer was used to accelerate convergence. The binary classification gender identification task fit the binary cross-entropy loss function. In the first phase, the model trained for 10 epochs, enough time to learn from the data without overfitting. When run on a GPU, each epoch took 24–27 seconds, making the entire training time 4–5 minutes. Initial epoch training output is shown below:

  Epoch 10/10
  32/32 [==============================] - 23s 628ms/step - loss: 15.0954 - gender_output_loss: 0.7149 - age_output_loss: 14.3805 - gender_output_accuracy: 0.4990 - age_output_mae: 14.3805 - val_loss: 14.9445 - val_gender_output_loss: 0.6716 - val_age_output_loss: 14.2730 - val_gender_output_accuracy: 0.5600 - val_age_output_mae: 14.2730

- **Performance**:

STEP-A: Compile and train your model

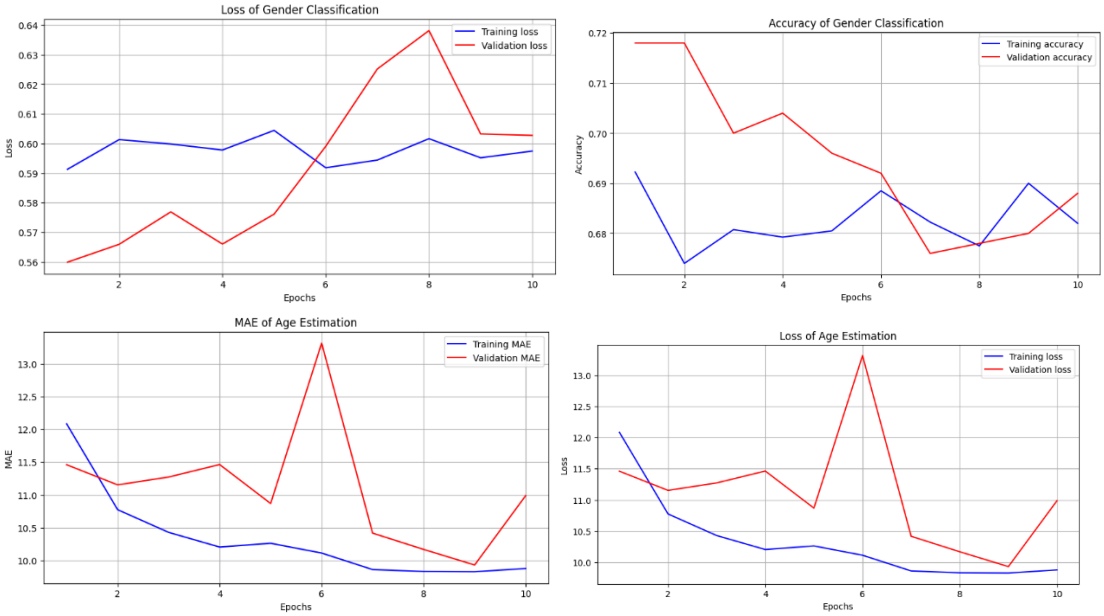


Figure 1. Gender classification and training graphs

Graphs show gender classification training and validation set loss and accuracy learning curves. Continuously decreasing training loss shows that the model is learning and improving its training data predictions. The validation loss decreases but is ambiguous, indicating the model cannot generalize to new data. Training accuracy exceeds validation accuracy, indicating overfitting.

The MAE of age estimation changes during epochs for both training and validation sets, showing the model predicts age less accurately than gender. A validation MAE peak at epoch 6 may be due to a batch of challenging validation samples or a training anomaly, but the model appears to be learning the age estimation job.

The pre-trained VGG16 gender classification model showed transfer learning, which applies results from one task to another. Training efficiency and validation set performance suggest fine-tuning a pre-trained image classification model with limited computational resources and time is successful.

Section 4: Summary and Discussion

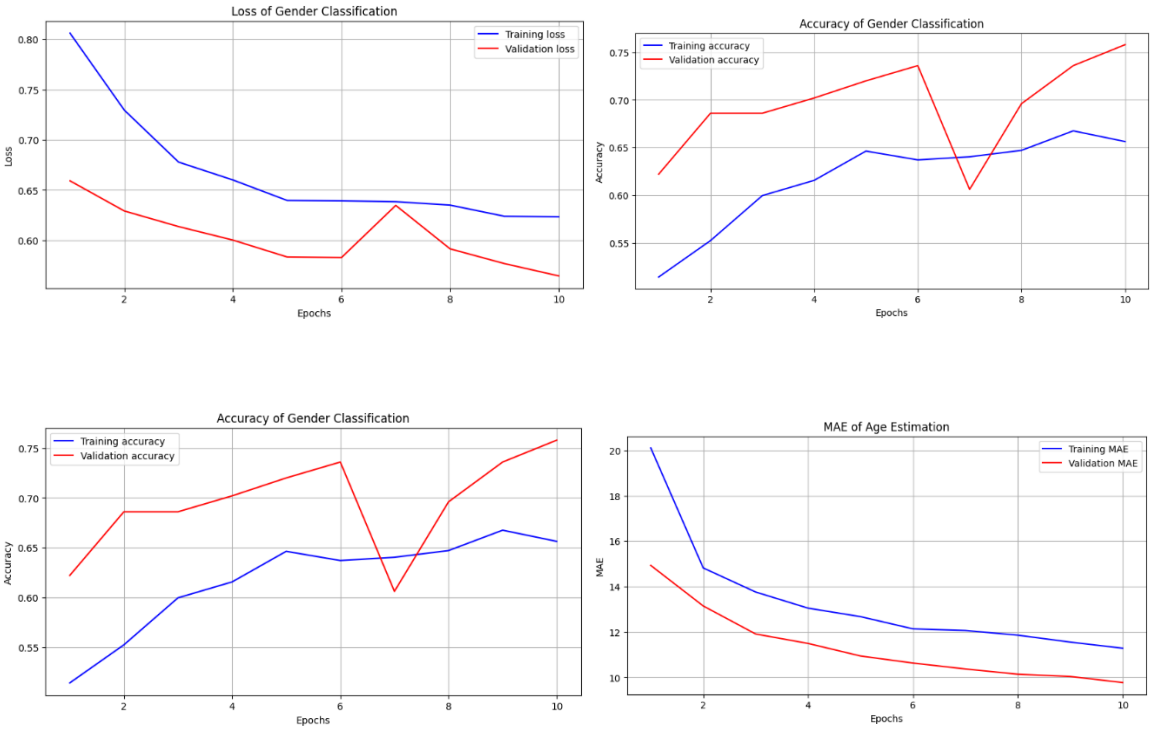
Compare the two models. What do you achieve in doing this assignment? Any other discussion you would like to have.

STEP-B: Build a CNN network based on a pre-trained model

Epoch 10/10

32/32 [==============================] - 27s 751ms/step - loss: 11.9080 - gender_output_loss: 0.6234 - age_output_loss: 11.2847 - gender_output_accuracy: 0.6562 - age_output_mae: 11.2847 - val_loss: 10.3344 - val_gender_output_loss: 0.5645 -

val_age_output_loss: 9.7699 - val_gender_output_accuracy: 0.7580 - val_age_output_mae: 9.7699



I saw contrast enhancement. Processing time increased dramatically when Epochs was increased from 10 to 30. I may become more precise. The demo version featured fewer epochs owing to GPU limits.

Training Output: The model learned well as gender categorization accuracy on the validation set increased from 0.526 to 0.758 from the first to the last epoch. The model may benefit from a learning rate schedule or extra data augmentation because its accuracy oscillates.

Performance: Gender categorization learning curves demonstrate consistent loss reduction and accuracy rise over epochs, indicating effective advancement.

The mean absolute error (MAE) for age estimate lowers over time, but it fluctuates, especially in the validation MAE, which may indicate overfitting or the need for hyperparameter adjustment.

Conclusion

By the last epoch, the VGG16 pre-trained model (Section 3) outperforms the custom-built model (Model A) in gender categorization.

Model A, the custom-built model, had more validation performance variance across epochs, especially in age estimate.

The validation loss and MAE for age estimate were less volatile in the pre-trained model after training.

Note: In my last attempt by increasing the epochs value to 20 from another computer:

The accuracy of gender classification (gender_output_accuracy) improved significantly from 0.6562 to 0.7080, showing that the model is classifying gender more correctly over additional training epochs.

References

University of Bath, Intelligent Control and Cognitive Systems, Activity: Dog or Cat - A Classification Example with CNN. [Online]. Available from: https://engage.bath.ac.uk/learn/mod/page/view.php?id=215561 (bath.ac.uk) [Accessed 31 January 2024]

Aurélien Géron, September 2019, Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2nd Edition [Online]. O'Reilly Media, Inc. Available from:

1. The Machine Learning Landscape | Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2nd Edition (oreilly.com)

TensorFlow Official Site, Compile and Train Model [Online].

Available from: https://www.tensorflow.org/tutorials/images/cnn#compile_and_train_the_model [Accessed 27 January 2024]

TensorFlow Official Site, Model-tfkeras [Online].

Available from: https://www.tensorflow.org/api_docs/python/tf/keras/Model#fit [Accessed 28 January 2024]

TensorFlow Official Site, tf.keras.Sequential [Online].

Available from: https://www.tensorflow.org/api_docs/python/tf/keras/Sequential [Accessed 27 January 2024]

https://www.tensorflow.org/tutorials/images/classification#data_augmentation

TensorFlow Official Site, Image classification and Data Augmentation [Online].

Available from: https://www.tensorflow.org/tutorials/images/cnn#compile_and_train_the_model [Accessed 27 January 2024]

Load and preprocess images

https://www.tensorflow.org/tutorials/load_data/images#next_steps

Convolutional Neural Network (CNN)

https://www.tensorflow.org/tutorials/images/cnn#create_the_convolutional_base

Transfer and Learnng

https://keras.io/guides/transfer_learning/#using-random-data-augmentation