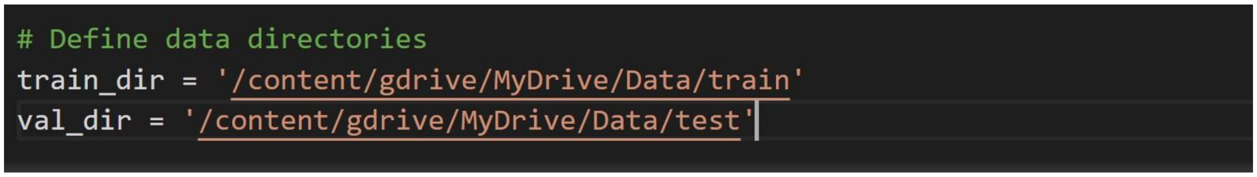
Assignment 1 Code Details

This code trains a binary classification model to classify images as either positive or negative for a certain target, using the VGG16 pre-trained model as a base.

This code is done in Google Colab, the code for this is provide in AlirezaTaraz_Assigment1.ipynb make sure you download that and follow each code cell carefully and run each cell in order. First mount your Colab notebook to Google Drive. Then imports the necessary libraries and access the training and validation image data. The dataset used for this project is Kaggle Retinal OCT images which can be found using the following link: https://drive.google.com/file/d/1DxSOyloRyd4q_kx6YpPwlrbjanmKaSWS/view , but that can be changed to your own data directories in this part of the code.



```
# Define data directories
train_dir = '/content/gdrive/MyDrive/Data/train'
val_dir = '/content/gdrive/MyDrive/Data/test'
```

Then it defines the data directories, image size, batch size, and data augmentation techniques.

Next, the pre-trained VGG16 model is loaded, and the base model layers are frozen to avoid overfitting. Additional layers are added on top of the base model and the model is compiled with an optimizer, loss function, and evaluation metric.

The model is trained using the "fit()" function on the training data, for a specified number of epochs, and the history of the training process is stored for visualization.

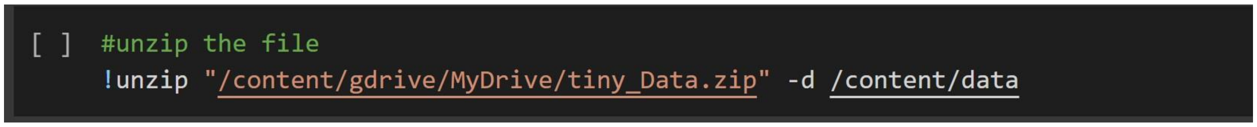
Finally, the code makes predictions on the validation data and creates a confusion matrix to evaluate the model's performance. The confusion matrix is plotted, along with the training and validation loss and accuracy curves.

Midterm Code Details

The code provided in AlirezaTaraz_midterm.ipynb is for building a Generative Adversarial Network (GAN) to generate high-resolution images from low-resolution images. The GAN consists of a generator and a discriminator. The generator takes in low-resolution images and outputs high-resolution images, while the discriminator distinguishes between real and generated high-resolution images.

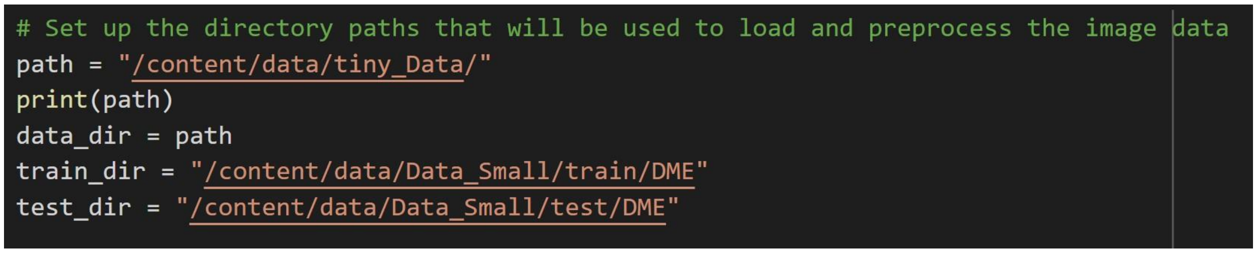
This code is done in Google Colab, the code for this is provided in AlirezaTaraz_midterm.ipynb make sure you download that and follow each code cell carefully and run each cell in order. First mount your Colab notebook to Google Drive. Next unzip your own data in Colab like shown below to make the loading process faster. The data that I used can be found using the following link: https://drive.google.com/file/d/1zGFhniov8QbpKcdg0a8jZF-QIdrNVDN6/view

Make sure it's using the right path to your file.



```
[ ]  #unzip the file
     !unzip "/content/gdrive/MyDrive/tiny_Data.zip" -d /content/data
```

Sets up the directory paths for loading and preprocessing the image data. Just make sure that the path is directed to the right location, where your dataset is in your Google Drive and change this part of the code for your own dataset.



```
# Set up the directory paths that will be used to load and preprocess the image data
path = "/content/data/tiny_Data/"
print(path)
data_dir = path
train_dir = "/content/data/Data_Small/train/DME"
test_dir = "/content/data/Data_Small/test/DME"
```

Next imports necessary libraries as shown in code.

The dataset is used for image super-resolution, which means that the model is trained to convert low-resolution images to high-resolution images. The generator network takes a low-resolution image as input and produces a high-resolution image as output. The discriminator network is used to evaluate the quality of the generated images and distinguish them from real images.

The generator network is built using a residual block architecture, which is commonly used in image super-resolution tasks. The discriminator network uses convolutional layers with batch normalization and leaky ReLU activation functions.