COMPUTER VISION - LAB 7

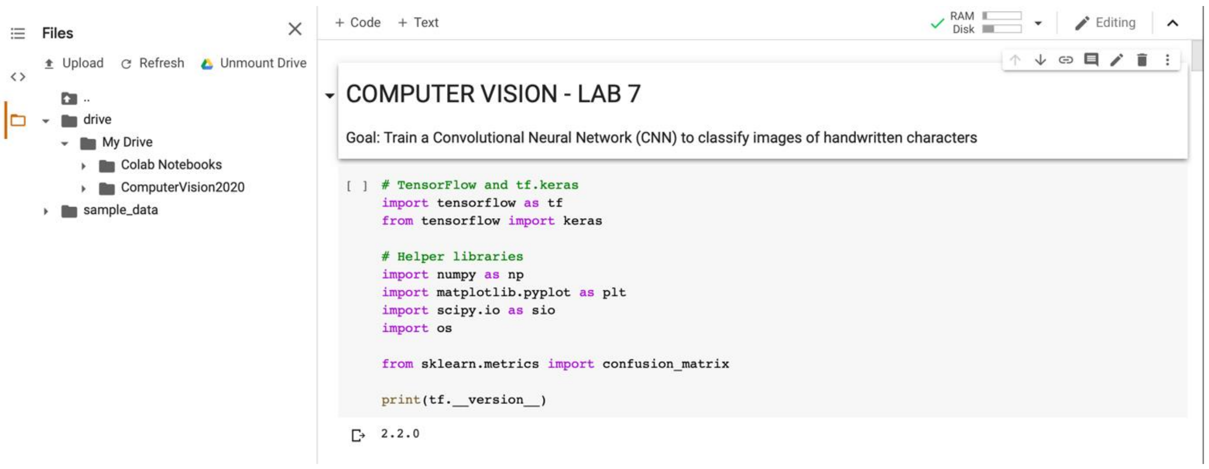
Topics: Image classification with deep learning
Goal: Train a Convolutional Neural Network (CNN) to classify images of handwritten characters

This lab experience builds on the deep network training example on the MNIST dataset recently discussed. The EMNIST dataset (https://www.nist.gov/itl/products-and-services/emnist-dataset) is an extended version of MNIST containing handwritten character digits and letters. The structure of the dataset is similar, and the images composing the dataset have the same format: 28x28 [pxl], grayscale. For this lab, we suggest you to use the Colab environment, that is freely available (see the notes at the end of the document) and includes the Keras library.

You are requested to train a deep network on the EMNIST dataset. As discussed in class, Python will be used to complete this task. Some code is already provided, with some helper functions that are described in the related comments. Starting from that code, you essentially need to work in the main function. The workflow is as follows:

1. Download the dataset from e-learning.
2. Feed the dataset to the Keras framework: you need to reshape the data in a format compatible with a CNN model (the provided data are arranged into 1D vectors). You can use the helper function ‘ `load_dataset(folder)` ‘ provided by the instructor to load and reshape the data.
3. Create a network to classify the dataset. You can start from the same network used for MNIST, but you should at least modify the last layer, in order to provide a number of outputs that is coherent with the number of classes of the problem (one for each of the 36 characters to be recognized).
4. Train the network and check the results. After training, you should check if the network showed a good performance, or if overfitting or underfitting occurred. For a better understanding of the results you achieved, you can visualize the learning process using the functions:
   a. `plot_history([('CNN_classifier', history)])`
   b. `print_confusion_matrix(model, test_images, test_labels)`
   c. `plot_example_errors(model, test_images, test_labels)`
5. After checking the performance of the first network you trained, you can modify its structure and complexity. For example, you can change the number of convolution filters for each layer, add or remove convolutional layers, change the number of neurons in the fully connected hidden layer. Each time you modify the network, you should train it and compare the performance with other structures you tried.

Notes:
If you use Google Colab to create and train the network, you can easily upload the dataset (download at step 2.) in you Google Drive account and access it in your Colab notebook from the left panel under the “Files” tab as in the figure below. Be sure to select the GPU acceleration under the Edit -> Notebook settings dialog box.



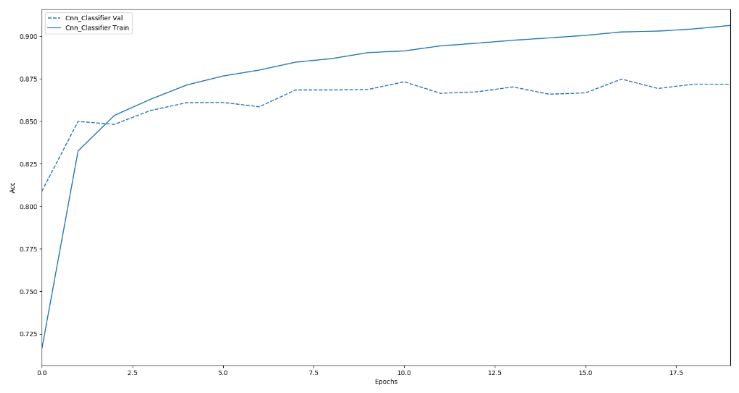
COMPUTER VISION - LAB 7

Goal: Train a Convolutional Neural Network (CNN) to classify images of handwritten characters

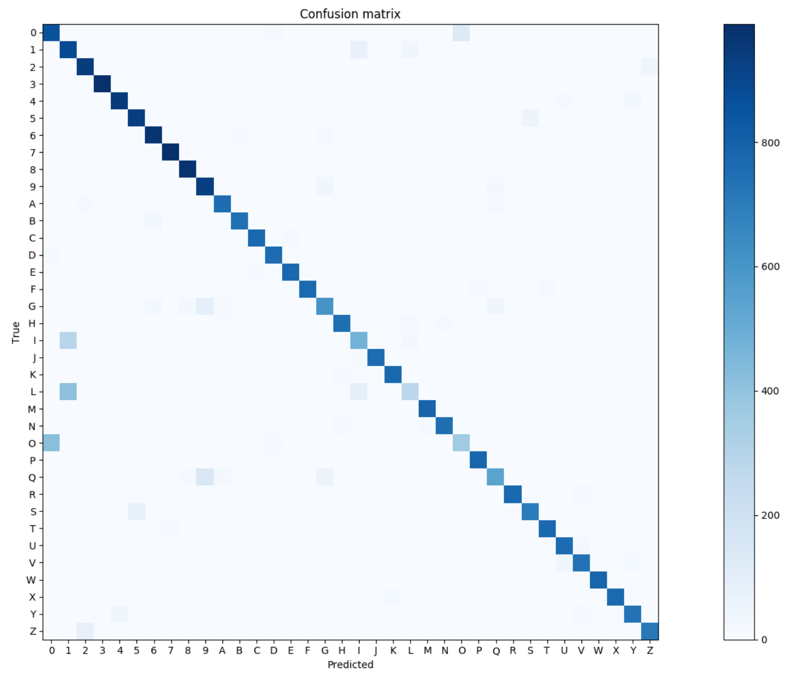
```
[ ] # TensorFlow and tf.keras
    import tensorflow as tf
    from tensorflow import keras

    # Helper libraries
    import numpy as np
    import matplotlib.pyplot as plt
    import scipy.io as sio
    import os

    from sklearn.metrics import confusion_matrix

    print(tf.__version__)
```

```
2.2.0
```

Example of how to load the dataset from Google Drive



Sample behavior of the training and validation error



Sample confusion matrix on the EMNIST dataset