**EIE4105 Multimodal Human Computer Interaction Technologies**

**Lab: Harness AI with Google Colab**

**A. Objectives and Outcomes**

After finishing this lab, you should be able to perform the following:

* Use Google Colab to implement image classifiers
* Use Keras on top of TensorFlow to develop DNN and CNN for handwritten digit classification
* Deploy the DNN and CNN for handwritten digit classification on Colab

**B. Assessment Criteria**

* Able to use and develop DNN and CNN on Google Colab
* Able to use a pre-trained model for image classification
* Able to explain the capability of different deep neural networks
* Able to write a clear report

**C. Submission and Demonstration**

* Copy and paste the graphs and images that you obtain in this lab and **convert it to PDF**.
* Submit your report to Blackboard before the deadline specified in Blackboard.

**D. Google Colab**

Google Colab is a free cloud service with free GPU. You may use Colab to develop deep learning applications using popular libraries such as Keras, TensorFlow, PyTorch, and OpenCV.

**E. Procedures**

***E.1 Prepare Colab Environment***

1. Colab runs on browsers. You need a Google account to use Colab. If you do not have one, visit https://support.google.com/mail/answer/56256?hl=en.
2. Display the Google Drive page in your browser. Create the following directory structure in your Google Drive:

My Drive/Learning/EIE4105/lab3

Later, you may mount your folder in Google Drive to the working environment of Colab.

1. Download <http://bioinfo.eie.polyu.edu.hk/download/EIE4105/lab3/python.zip>. Decompress the .zip file to a folder in your desktop/notebook computer folder. You should see the folder python/.
2. Copy the folder “python/” to your google drive folder “My Drive/Learning/EIE4105/lab3”. After finishing this step, you should see the folder structure like this on your Google Drive:

Application

Description automatically generated with medium confidence

1. Read the file “Drive/Learning/EIE4105/lab3/imagenet/imagenet-console.py” using Google TextEditor. It will use the pre-trained model ResNet50, which is a ResNet with 50 layers.

***E.2 Image Classification using Pre-Trained CNN***

1. In your Google Drive page, under the folder “EIE4105/lab3/python”, click the “New” button to create a Colab ipython file “EIE4105-lab3.ipynb”

Graphical user interface, text, application

Description automatically generated

1. Check if Keras and TensorFlow has been installed

Text

Description automatically generated with medium confidence

1. Mount the Google Drive to the IPython Notebook as follows:

A screenshot of a cell phone

Description automatically generated

Click the link and follow the instruction. Put the key in the edit box and press the “Return” key. You should see the following after mounting.

A screenshot of a cell phone

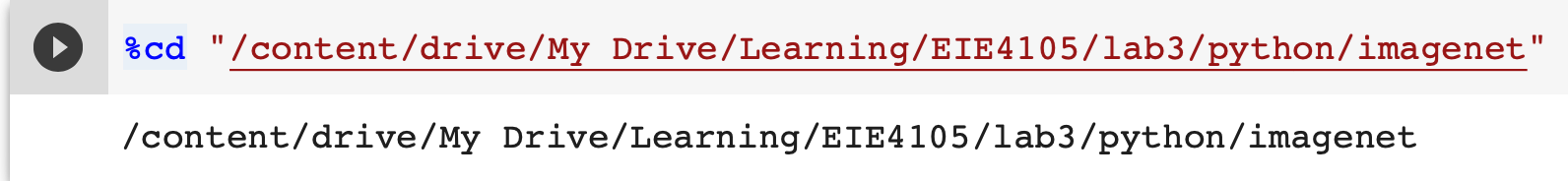
Description automatically generated

1. Configure the Colab to use Python3 and GPU by clicking Edit 🡪 Notebook settings. Select “GPU” in the pop-up window:

Graphical user interface, text, application

Description automatically generated

1. Double click the file “imagenet-console.py” under the folder “imagenet” on the left panel of Colab. You should see an edit window that allows you to edit the python file. You may need to change the “img\_path” so that the program can read the .jpg files in the “images” directory.
2. Click the “+ Code” button on the top-left to create a new command box and put the following command in the box to change the folder.



1. Check your current folder:

Graphical user interface, text

Description automatically generated

1. Execute the python file:

Graphical user interface, text

Description automatically generated

Capture the results for different images and put them into your report.

1. Download some more images (.jpg) files from the ImageNet ([www.image-net.org](http://www.image-net.org)) and see if the program can recognize the objects. Put the images into your report and state the recognition results (including the class name and confidence level). You may also find some images in <http://bioinfo.eie.polyu.edu.hk/download/EIE4105/lab3/images.zip>. Then, copy the images to “My Drive/Learning/EIE4105/lab3/python/imagenet/images”.

***E.3 DNN and CNN for Handwritten Digit Recognition***

1. Study the file “EIE4105/lab3/python/mnist/mnist\_dnn.py.” in your Google Drive.
2. Change directory to “/content/drive/My Drive/Learning/EIE4105/lab3/python/mnist”. Execute the python file “mnist\_dnn.py”, which trains and tests a DNN. You should see something like this:

Text

Description automatically generated

Read the code in “mnist\_dnn.py” carefully to understand how to implement a DNN.

1. Change the number of hidden layers and the number of nodes in the hidden layers to see if the changes could improve the classification accuracy. Report the accuracy you obtain.
2. Open the file “mnist\_cnn.py”. This script trains and tests a convolutional neural network (CNN). Read the code carefully to understand how to implement a CNN. Create a folder “models” in your Google Drive to store the CNN as an .h5 file.



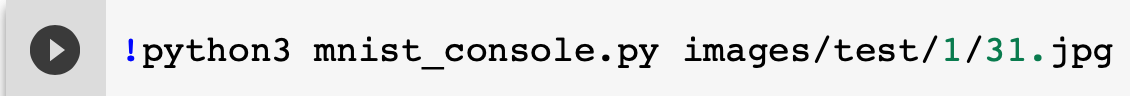
1. Execute the script as follows:



The script also saves the CNN in .h5 format in the folder “mnist/models”. Change the number of layers and the number of nodes to investigate how these parameters affect performance. Also, change the activation function to see if you can get better performance.

1. Open the file "mnist\_console.py." Study the code carefully to see how to use a pre-trained CNN for handwritten digit recognition from .jpg files. Execute the program as follows:





Record your observations.

You may display the digit on your browser by using the following code:

Text

Description automatically generated

1. Modify the program “mnist\_cnn.py” so that it uses 1D-CNN instead of 2D-CNN.
   1. Compare the performance of the 1D-CNN against the 2D-CNN.
   2. Compare the output shape of each layer in the 1D-CNN against the output shape of the corresponding layer in the 2D-CNN. Why they are different?
   3. Why is the number of parameters in 1D-CNN is larger than that of 2D-CNN if the kernel size of the two CNN is the same?
   4. Why 1D-CNN is not appropriate for classifying hand-written digits?

***Hints***: Refer to the documentation of tensorflow.keras.layers.Conv1D and tensorflow.keras.layers.MaxPooling1D. Note that the input\_shape should be 2D rather than 3D, and the first dimension (row) is the time dimension. Also, you define the kernel\_size and the pool\_size in the time dimension only.

**References:**

[1] <https://medium.com/deep-learning-turkey/google-colab-free-gpu-tutorial-e113627b9f5d>

[2] <https://pyimagesearch.com/2019/10/21/keras-vs-tf-keras-whats-the-difference-in-tensorflow-2-0/>

-- END --