**EIE4105 Multimodal Human Computer Interaction Technologies**

**Lab: Harness AI with Google Colab**

E.1.13.  
Capture the results for different images and put them into your report.

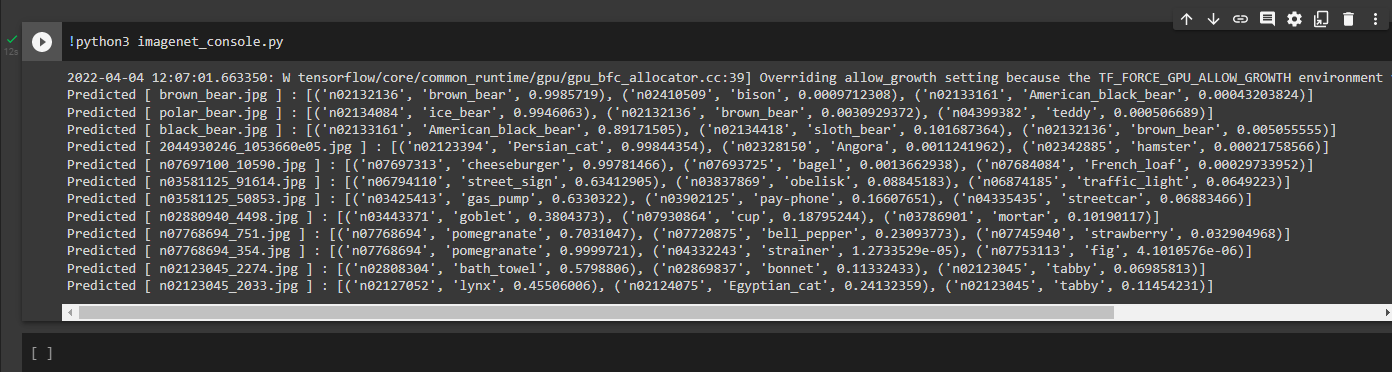
Graphical user interface, application

Description automatically generated

E.1.14.   
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”.

A picture containing text, different

Description automatically generated



E.2.17.  
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.

|  |  |
| --- | --- |
| Graphical user interface, text  Description automatically generated[Default: 3 hidden layers with 200 nodes in each layer]  (Train Accuracy: 99.62%, Test Accuracy: 97.89%) | |
| [1 hidden layer with 200 nodes in the layer] (Train Accuracy: 99.62%, Test Accuracy: 98.12%) | [10 hidden layers with 200 nodes in each layer] (Train Accuracy: 98.89%, Test Accuracy: 97.78%) |
| [3 hidden layers with 20 nodes in each layer] (Train Accuracy: 96.63%, Test Accuracy: 95.47%) | [3 hidden layers with 2000 nodes in each layer] (Train Accuracy: 99.51%, Test Accuracy: 98.15%) |
| [1 hidden layer with 2000 nodes in the layer] (Train Accuracy: 99.68%, Test Accuracy: 98.15%) | [10 hidden layers with 20 nodes in each layer] (Train Accuracy: 62.15%, Test Accuracy: 61.92%) |

From the testing result, we can see the trend that the less the hidden layers we have, and the more the nodes we have, we obtain a better result.

E.2.19.  
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.

|  |  |
| --- | --- |
| [Default] Accuracy: 99.03% | |
| [Less Layers] Accuracy: 98.66% | [More Layers] Accuracy: 19.62% |
| [Less Nodes] Accuracy: 98.37% | [More Nodes] Accuracy: 99.15% |
| [sigmoid] Accuracy: 98.59% | [softmax] Accuracy: 98.81% |
| [tanh] Accuracy: 98.62% | [exponential] Accuracy: 9.80% |

From the result above, the best setting should be having more nodes, less layers, and having an activation function ‘relu’.

E.2.20.

Record your observations.

|  |
| --- |
| [Digit 0] |
| [Digit 1] |

E.2.21.  
Modify the program “mnist\_cnn.py” so that it uses 1D-CNN instead of 2D-CNN.  
a) Compare the performance of the 1D-CNN against the 2D-CNN.

|  |  |
| --- | --- |
| [1D-CNN] Accuracy: 97.59% | [2D-CNN] Accuracy: 99.03% |

b) 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?

|  |  |
| --- | --- |
| [1D-CNN] | [2D-CNN] |

Conv1D and Conv2D are in different dimension, the result of the kernel convoluting on them are also different as they are in different dimension.

c) 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?

As the activation size of 1D-CNN is larger than that of 2D-CNN in this case.

d) Why 1D-CNN is not appropriate for classifying hand-written digits?

As 1D-CNN is not suitable for pattern recognition (like hand-written digits in this case).