BIRMINGHAM CITY
University

CMP6228 - Unknown Assignment

Untitled CMP6228 Report

Lewis Higgins - Student ID 22133848

CMP6228 - Deep Neural Networks

Module Coordinator: Khalid Ismail

Contents

Introduction

This module consists of two assignments in a similar fashion to CMP6230. By week 8, you must have found and written a **proposal** (20%) detailing the problem to be solved with deep learning.

The second assignment is worth 80%, and is a full report detailing the design, development and evaluation of a deep learning model to solve the proposed problem. From this, an immediate observation is once again the medical field - image classification of people with some kind of disease from an X-ray dataset of some sort. Recall previous data sources - Kaggle, IEEE DataPort, UCI ML. Try to avoid the same topics as others if possible, though that's perhaps even less likely than it was in other modules if image classification is the main objective.

Assignment 2's deadline is presumably in Week 12, or maybe even later. ***A draft version is due in Week 11, however.***

It is unlikely this module can be completed on your laptop, as TensorFlow will be used. Colab is proposed as an option, which could perhaps be feasible enough, though that may likely depend on the size of the chosen dataset, of which no specific value count was given. (Nor feature count, though this is likely irrelevant in an image classification scenario). Colab ran epochs $\tilde{3}$0-40% quicker at first glance if the runtime was changed to a GPU. You should fine-tune a good amount of epochs; more epochs just increase training time, but will they actually enhance the model that much or will it just plateau? A 3060 Ti processes twice as fast as Colab's free GPU.

Because of this assignment being almost identical to CMP6202 (only difference is the model and the fact that you'll likely be doing image classification rather than text), it's likely that elements of it can be adapted to Assignments 1 and 2. It's likely to be a very math-heavy module.

The Moodle page for this module is not immediately available; as such, you can't really plan ahead for what will be done on a week-by-week basis. Because of this uncertainty, it might be best to prioritise this module over CMP6207.

Tech stack

- Previously seen libraries:
  - Conda
  - Numpy
  - Scipy (sort-of seen before but not properly)
  - Scikit-learn
- Pillow
  - An image processing library.
- h5py
  - Appears to be used for serialization of models, like Pickle was in CMP6230.
- Keras & TensorFlow
  - Keras apparently uses TensorFlow to maximise efficiency with tensor calculations.
  - (Or the other way around. Keras is a part of TensorFlow now.)
  - A tensor is a multidimensional array or matrix.

Keras has two ways of composing models: Sequential and Functional. **Only sequential will be used in the scope of this module.**

Sequential

Multiple predefined models are stacked in a linear pipeline like a stack or queue:

```
model = Sequential()
model.add(Dense(N_HIDDEN, input_shape=(784,)))
model.add(Activation('relu'))
model.add(Dropout(DROPOUT))
model.add(Dense(N_HIDDEN))
model.add(Activation('relu'))
model.add(Dropout(DROPOUT))
model.add(Dense(nb_classes))
model.add(Activation('softmax'))
model.summary()
```

Functional

Uses an API where complex models can be defined such as DAGs (!!!), models with shared layers, or multi-output models. Not much info was given on this, likely requiring your own research. **Although it's good to know that these exist, they won't be used in this module.**

Conclusion

Overall, something was done...

Appendix A - Windows Subsystem for Linux

As of TensorFlow 2.10's release in September 2022, Windows is no longer natively supported with GPU access (TensorFlow, 2025b). To circumvent this issue, the official TensorFlow Docker container can be pulled and used on a Windows host (TensorFlow, 2025a). Docker aims to mitigate issues of software incompatibility across devices through containers, which act similarly to virtual machines in that they emulate another system with the necessary dependencies on a host. In this specific case, Docker uses the Windows Subsystem for Linux to virtualize an Ubuntu client that supports TensorFlow's latest releases with GPU support.

I'm not going to remove this section until I make a final decision on the following:

- My RTX 3060 Ti seems to work at the same speed, if not slower, than a Colab GPU.
- My CPU works over 1000x faster than both, but my research shows that this is true for small datasets and the time needed grows exponentially.
- It is possible that the GPU speed issues are due to low VRAM allocation, but this cannot actually be resolved without **DUAL-BOOTING LINUX.**
- Though, is that even necessary? Tensorflow 2.10 is a version that supports the 3060Ti. If half of the things online use completely outdated Python versions, why shouldn't I use a slightly outdated TensorFlow version?
- All in all, the decision between Colab and local use will depend entirely on the size of the chosen dataset, though it's likely Colab will win out.

Bibliography

TensorFlow (2025a). *Install TensorFlow with Docker*. TensorFlow. URL: https://www.tensorflow.org/install/docker (visited on 30/01/2025).

TensorFlow (2025b). *Install TensorFlow with pip*. TensorFlow. URL: https://www.tensorflow.org/install/pip (visited on 30/01/2025).