Assessment 2019

Emerging Technologies

Due: December 13th, 2019

This document contains the instructions for Assessment 2019 for the module Emerging Technologies. Your submission for this assessment must be in the form of a single GitHub repository and the URL for this repository must be submitted using the link on the Moodle page on or before October 11th. You may use a private repository if you wish but make sure to add `ianmcloughlin` as a collaborator, again on or before October 11th.

Your submission must be substantially your own work, otherwise it will not be considered for marking. In completing the project, you will be expected to carry out your own investigations and reference all sources used. Please be advised that all students are bound by the Quality Framework [2] at GMIT which includes the Code of Student Conduct and the Policy on Plagiarism.

Instructions

This assessment concerns the well-known MNIST [7] dataset and the Python [1] packages keras [6], flask [5], and jupyter [4]. The project is worth 100% of your marks for this module, but it is broken into three parts, as follows.

(20%) Presentation: Create a git repository and make it available online for the lecturer to clone. The repository should contain all your work for this assessment. In the last week of term, or otherwise as advised in class, you must demonstrate your repository and its code to the lecturer. This part is worth 20% of your overall mark.

(40%) Model: Create, document, and train a model that recognises hand-written digits using the MNIST dataset. This should be done using the keras and jupyter Python packages, and this part is worth 40%.

(40%) Application: Create a web application that allows a user to draw a digit using their mouse or touchscreen device. The drawing should then be then be submitted for recognition to the model you have trained above. This should be done using the flask Python package and is also worth 40%.

Minimum Viable Project

The minimum standard for this assessment is a git repository containing a README file written in Markdown [3], a jupyter notebook containing code to train your model, and a flask web application that allows users to recognise digits using your trained model. The README should contain a summary of your work and provide instructions as to how to run the jupyter notebook and the web application.

A good project will be well laid out, clear and concise, and easily understood and run. The trained model will accurately predict digits as drawn by the user, and the web application will provide a good user experience.

Marking scheme

The following marking scheme will be used to mark the project out of 100%. Students should note, however, that in certain circumstances the examiner's overall impression of the project may influence marks in each individual component.

Presentation		
20%	**Presentation**	Well-organised, easily understood repository with good documentation. Able to explain all contents of repository in discussion.
Model		
20%	**Research**	Clear evidence of research into developing the model with references to other works evident within the jupyter notebook.
20%	**Documentation**	Clear explanation and documentation of training the model using a jupyter notebook.
Application		
20%	**Code**	Simple, straight-forward code with appropriate comments and decent architecture.
20%	**User exp.**	Good user experience – easy to use, well-designed front-end with good feedback to the user and efficient submission to the model.

Advice for students

- Your git commit history should be extensive. A reasonable unit of work for a single commit is a small function, or a handful of comments, or a small change that fixes a bug. If you are well organised you will find it easier to determine the size of a reasonable commit, and it will show in your git history.
- Using information, code and data from outside sources is sometimes acceptable — so long as it is licensed to permit this, you clearly reference the source, and the

overall project is substantially your own work. Using a source that does not meet these three conditions could jeopardise your mark.

- You must be able to explain your project during and after its completion. Bear this in mind when you are writing your README. If you had trouble understanding something in the first place, you will likely have trouble explaining it a couple of weeks later. Write a short explanation of it in your README, so that you can jog your memory later.
- Everyone is susceptible to procrastination and disorganisation. You are expected to be aware of this and take reasonable measures to avoid them. The best way to do this is to draw up an initial straight-forward project plan and keep it updated. You can show the examiner that you have done this in several ways. The easiest is to summarise the project plan in your README. Another way is to use a to-do list like GitHub Issues.
- Students have problems with projects from time to time. Some of these are unavoidable, such as external factors relating to family issues or illness. In such cases allowances can sometimes be made. Other problems are preventable, such as missing the submission deadline because you are having internet connectivity issues five minutes before it. Students should be able to show that up until an issue arose, they had completed a reasonable and proportionate amount of work and took reasonable steps to avoid preventable issues.
- Go easy on yourself - this is one project in one module. It will not define you or your life. A higher overall course mark should not be determined by a single project, but rather your performance in all your work in all your modules. Here, you are just trying to demonstrate to yourself, to the examiners, and to prospective future employers, that you can take a reasonably straight-forward problem and solve it within a few weeks.

References

[1] Python Software Foundation. Welcome to python.org.
`https://www.python.org/`.

[2] GMIT. Quality assurance framework.
`https://www.gmit.ie/general/quality-assurance-framework`.

[3] GitHub Guides. Mastering markdown.
`https://guides.github.com/features/mastering-markdown/`.

[4] Project Jupyter. Project jupyter.
`https://jupyter.org/`.

[5] Pallets. Flask.
https://flask.palletsprojects.com/en/1.1.x/.

[6] Keras Team. Keras: The python deep learning library.
https://keras.io/.

[7] Christopher J.C. Burges Yann LeCun, Corinna Cortes. The mnist database of handwritten digits.
http://yann.lecun.com/exdb/mnist/.