Coursework for “Deep Learning” [CSC8637]

This document outlines the coursework for the Deep Learning module (CSC8637). Four separate tasks are involved, carrying a different percentage of the total mark. In the following, each task is described along with submission requirements demo requirements and the marking scheme.

You are also required to perform a short (15 minute) demonstration of your work. These will be scheduled for after your coursework submission.

Hand-in deadline: 4.30pm Friday 23rd February 2024.

---

1. Using an Existing Model [30%]:

An important part of any deep learning practitioner’s job is the ability to run existing model architectures on different types of data. In this part of the coursework, you will demonstrate your ability to work with models by training a CycleGAN model on specific datasets.

CycleGAN has very visually impressive results and is widely known and used within the machine learning community. You can learn more about the approach by reading the paper: https://arxiv.org/pdf/1703.10593.pdf. The project page for CycleGAN is also worth looking at: https://junyanz.github.io/CycleGAN/.

The paper outlining CycleGAN is one of the two papers covered in the “Paper Reading” workshops held as part of the module.

You are not actually expected to write the code for CycleGAN yourself (but you are free to do so if you wish – though there are no extra marks for doing this). Various implementations of CycleGAN are available in different frameworks (Keras, TensorFlow and PyTorch). Some of the examples are as follows:

- Keras:
  - https://keras.io/examples/generative/cyclegan/
  - https://github.com/simontomaskarlsson/CycleGAN-Keras
- PyTorch:
  - https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix
- TensorFlow:
  - https://github.com/leehomyc/cyclegan-1
  - https://github.com/architrathore/CycleGAN/

There are numerous other implementations available and you can choose whichever you think best!

The objective is for you to train CycleGAN to transfer between human faces and cats/dogs.

The dataset of human faces is a combination of two datasets:

- UTK Face Cropped: https://www.kaggle.com/datasets/abhikjha/utk-face-cropped
- Inceptionv3: https://www.kaggle.com/code/bmarcos/image-recognition-gender-detection-inceptionv3/data

You must download both datasets from Canvas:
https://ncl.instructure.com/courses/49739/files/7871434?module_item_id=3133483

The dataset of cats and dogs is a combination of two datasets:

- Dog Face Dataset: https://images.cv/dataset/dog-face-image-classification-dataset
- Cat Dataset: https://www.kaggle.com/datasets/spandan2/cats-faces-64x64-for-generative-models

You must download both datasets from Canvas:

- Dog Face Dataset: https://ncl.instructure.com/courses/49739/files/7870876?module_item_id=3133358
- Cat face Dataset: https://ncl.instructure.com/courses/49739/files/7870820?module_item_id=3133334

You need to be able to process the data appropriately and train the model correctly (without overfitting or underfitting) so it can produce visually plausible results.

You will submit the following:

- Short report - no more than half a page - on what you did to process the data and train the model and the hyper-parameters within the model you have used/changed.
- Some examples of the results of the model - this includes images of yourself and module staff members stylised as cats/dogs using your trained model. You can select any image of yourself (optional) and you can find staff images in the intro lecture 00. If you do not wish to use an image of yourself, you can use an image of any celebrity.
- For testing we will provide you with some images, so make sure your code will allow this.
- Code and checkpoints (trained model weights) along with instructions on how to run your model to reproduce your submitted results.

Marking:

- **20/30:** Code (with trained model weights) that runs correctly and re-produces the submitted results.
- **10/30:** Visual quality of the results. Stylised images should be visually plausible and contextually look like cats/dogs. Your images should have a minimum resolution of 64x64 (larger images are acceptable if they are not too blurry or noisy).

Demo:

- We will provide you with a set of images from either domain. You should be able to load these into your chosen tool and convert them to the new domain. You can either display them directly or save to a file and open them.

2. Fine-Grained Classification [30%]:

The fine-Grained Image Classification tasks distinguish subtle differences between classes. In both academia and industry, it is highly important to get the best possible performance out of any model. The objective here is to train a network that can accurately classify models of airplanes.

The dataset you will be using for this task is from:
https://www.kaggle.com/datasets/seryouxblaster764/fgvc-aircraft/code
You must download the dataset from Canvas:
https://ncl.instructure.com/courses/49739/files/7871440?module_item_id=3133487

The dataset comes with a training set, a validation set and a test set. You are only allowed to train your model using the training set. Having a validation set is helpful to optimise your model hyper-parameters. The test set is only used to generate the final accuracy values.
You should not use the test set for training under any circumstances.

You can use any model architecture you want. You may use any of the seminal classification architectures or you can design your own network. It is recommended that you use transfer learning (by using weights pre-trained on ImageNet). You can try to freeze some layers of your network during fine-tuning or you can finetune the entire model.

You should find the best hyper-parameters you can, so you get the best results possible (this is what the validation set is for).

At the end, after model training is complete, you should assess the performance of your classifier using the test set and report **Accuracy**, **Precision** and **Recall**. You should also create a **Confusion Matrix**.

Getting a decent level of accuracy should be relatively easy as the data distributions to which the training set and the test set belong are aligned. However, you are encouraged to try to get the best performance out of the model as all submissions are ranked based on model performance and 5 marks are given based on the rank of the submission.

Additionally, another 10 marks is dedicated to reaching certain levels of accuracy. Any submission which achieves an accuracy value above 90% on the test set will receive the full 10 marks. Submissions that produce accuracy levels between 85 to 90%, will get a mark of 8 out of 10. Any submission with accuracy results between 80 to 85% will receive 6 marks. Any submission with results from 75 to 80% accuracy will get 4 marks. All submissions with accuracy values from 70% to 75% will receive 2 mark and all submission below 70% accuracy will receive zero out of the 10 marks dedicated to this.

You will submit the following:

- Short report – no more than half a page – on what you did to process the data and train the model and the hyper-parameters you have used within your model. The reasons for choosing the hyper-parameters must be explained (report if you have run any experiments or searches or tell us if you just guessed what the best hyper-parameters were). Your final results (accuracy, precision, recall and confusion matrix – the confusion matrix does not need to be within the half-page limit) should be included in this report.
- Full training and test code along with accurate instructions as to how they should be run. Your results should be reproducible. [*Hint: be careful about random number generators introducing stochasticity. Seeds are your friend.*]
- Model checkpoints (trained model weights) should also be submitted to ensure reproducibility of the reported results.

Marking:

- **15/30:** Code (with trained model weights) that runs correctly and re-produces the submitted results.
- **10/30:** Marks distributed according to the performance of your model (based on the accuracy metric). 10 for above 90% accuracy. 8 for 85-90. 6 for 80-85. 4 for 75-80. 1 for 75-80. 0 for any submission with an accuracy of less than 70%.
- **5/30:** Type of approach used to achieve the final result, e.g., network architecture design, method of hyper-parameter selection, or type of transfer learning.

Demo:

- You will be given a set of images (and labels). You need to be able to pass these through your Deep Learning model and provide accuracy, precision, recall, F1 and confusion matrix for these images along with the individual labels for each image.

---

3. Time-Series Model [30%]:

Much of the data we need to process is time-series data. In this part of the coursework we give you twelve months' worth of data for the logins and logouts of students to a mythical university set of computers. The dataset can be downloaded from:

The file is a comma separated value (CSV) file with the following columns:

- Date and time of the event
- The event type – either LOGIN or LOGOUT
- The cluster on which the event occurred
- The duration – for a LOGIN this is the number of milli-seconds the user was logged in for, for a LOGOUT this is zero
- The total number of users logged in at that point in time

The CSV file can be downloaded from:
https://ncl.instructure.com/courses/49739/files/7910028?module_item_id=3151207

There are a lot of things you can predict from this dataset, for the purpose of this coursework you will be assessed on the following 3:

1. Given a sequence of events (in the format listed above) predict the next 100 login / logout times.
2. Given a sequence of events (in the format listed above) predict the next 100 values for the number of students using the computers.
3. Given a sequence of events (in the format listed above) predict the next 100 cluster names for either logins / logouts.

You will submit the following:

- Short report - no more than half a page - on what you did to process the data and train the model and the hyper-parameters you have used within your model. You may include a few examples of the outputs of your model.
- Full training and test code along with accurate instructions as to how they should be run. Your test code should be written in a way that allows the user to provide a sequence of events (in a CSV file) and then outputs the appropriate result.
- Model checkpoints (trained model weights) should be submitted to enable the testing of the model (text generation).

Marking:

- **20/30:** Code (with trained model weights) that runs correctly and is capable of generating the results presented.
- **5/30:** Type of approach used to achieve the final result, e.g., network architecture design, method of hyper-parameter selection, or data processing method.
- **5/30:** Quality of the generated sentences. To receive the full 5 marks, the generated results should follow the pattern of the true labels.

Demo:

- You will be expected to demo three results. This could be three separate networks:
- Given a CSV file of N events, remove the last 100 of these (for testing). Run your model on the N-100 events and predict the last 100 login / logout times. Produce an R^2, MSE, MAE values for these last 100 predictions.
- Given a CSV file of N events, remove the last 100 of these (for testing). Run your model on the N-100 events and predict the number of students using the system for the next 100 events. Produce an R^2, MSE, MAE value for these last 100 predictions.
- Given a CSV file of N events, remove the last 100 of these (for testing). Run your model on the N-100 events and predict the cluster names for the next 100 events. Produce an accuracy, precision, recall and F1 score for this.

4. Reflecting on the Paper Reading Workshop [10%]:

The area of machine learning is extremely fast-paced and the state of the art in most tasks and applications changes at least every few months. As such, reading a few textbooks and learning the fundamentals, while necessary, is not enough and it is very important to keep up to the date with the literature. Reading papers is thus an essential skill, not unlike many other areas of computer science.

In a short report, you will reflect on what you have learnt from the paper reading workshops held as part of this module. You will outline how the skill and knowledge you have gained has affected your process of reading papers and share your insight on the subject.

You will submit the following:

- Short report - no more than half a page - on what you have learned about reading academic papers and how this will affect your future as a deep learning specialist whether working in industry or academia. The report will also include a very brief summary of the papers we have read (no more than a few sentences).

Marking:

- **5/10:** A clear description of the key skills that you have learnt for identifying pertinent papers and how you would then go about gaining the most information from these papers.
- **5/10:** A short summary of the two papers covered.