COMP47650 – Deep Learning Project

Version 3.5 – February 23, 2023

Overview

This individual project will examine your ability to apply deep learning learning architectures and address challenges relating to complex, real world problems. This project is worth 80% of the final grade.

1 Specification

The project is split into the following sections: code, report and video presentation.

1.1 Code

Projects should be completed in **either** Tensorflow (>=2.7.0) **or** Pytorch (>=1.10). Part of this project is to familiarise yourself with these frameworks, so please make sure to start early in case you run into any problems! Code should be clear, commented and most importantly, all experiments must be **reproducible**[1] in their entirety.

If you use code from elsewhere (for preprocessing, pretrained models,...etc.), make sure to cite the resource used by providing an appropriate link within the code as a footnote in the report. You are free to use pre-trained models to aid performance but you should explain the motivation behind using a particular model.

1.2 Report

The report should be in a conference paper-style format, using the NeurIPS LaTeX template (compile as "preprint") and style guide (available at `https://neurips.cc/Conferences/2020/PaperInformation/StyleFiles`). Reports should be 4–6 pages in length (excluding references) and should contain (but are not limited to) the following sections:

- Abstract
- Introduction
- Related Work
- Experimental Setup
- Results
- Conclusion & Future Work
- References

A description of these sections is included in the Resources section of this document.

1.3 Video presentation

You are required to prepare a 5 minute video where you will present your assignment using the following guidelines:

- *Introduction:* State your name and student number. Continue with a brief introduction of the topic, including problem statement and project objective.
- *Walkthrough:* Walk through the Jupyter notebook you created, highlighting the key sections and explaining your thought process as you developed the code. Evaluate cells as you progress.
- *Model explanation:* Provide an overview of the deep learning model you used, including its architecture, hyperparameters, and optimization algorithm. Explain why you chose this particular model and how it's suited to solving the problem at hand.
- *Training revelation:* Share any insights or revelations you had while training the model, including any challenges you faced and how you overcame them.
- *Test and evaluation:* Demonstrate how you evaluated the model's performance, including any metrics you used and how you interpreted the results. Show how you tested the model and explain how well it generalizes to unseen data.
- *Conclusion:* Summarize your findings and any lessons learned, Discuss how the project could be improved or extended in the future. Briefly discuss your report.

[1] A quick guide to reproducible results can be found at `https://machinelearningmastery.com/reproducible-machine-learning-results-by-default/`

Remember to keep your video concise and engaging: 5 minutes maximum. Use video conferencing software such as Zoom or MS-Teams to capture and record a screen sharing video **with an embedded self video (your face must be visible)**. Video should be encoded using the H.264 codec (widely used video compression format that provides good quality and efficient file size) in MP4 format.

2 Dataset

You are free to select a dataset **within the category randomly assigned to you** (either image, audio or text). Datasets are sourced from online repositories and present various challenges that you should address in your project. Details for dataset allocation and access is available in a separate document.

If dataset split is not available, shuffle data with random seed = 2023 and take last 20% of data for testing.

3 Experiments

3.1 Useful considerations before starting

- What pre-processing is necessary to perform before modelling data?
- What is the structure of the data? Is it sequential?
- Is the data balanced?
- If so, what techniques can be used to address this imbalance?
- What deep learning architectures are most appropriate for this data?
- Can the data be changed to fit a particular architecture?
- Are there existing pre-trained models that could be fine-tuned?
- Can the model be parallelised (make better use of GPUs for quicker training)?
- How many weights are there in the model? How much memory does this consume?

3.2 Evaluating model performance

A validation set should be created either using a subset of the training set or use the one provided to evaluate your model's ability to generalise and to perform hyper-parameter tuning. **Never use the test set in this process**. You should create a plot showing the training and validation loss on the y–axis and the iteration of training on the x–axis to identify the point where generalisation is best. The model that performs best on the validation set should then be used for testing. Explain and motivate your choice of evaluation measures which should be relevant to your dataset task. Report model performance using these measures.

It should be noted that the training procedure will likely be repeated during grading and any substantial differences (small differences are expected due to random initialisation) between the reported training loss and that obtained upon repeating the experiment will bring the results – and therefore overall project – into question.

4 Submission Details

4.1 What to submit

You are required to submit 2 files.

(i) zip file containing the following:

- All related scripts/ jupyter notebooks
- Model weights for each model trained[2].
- A 'requirements.txt'
- A short README file explaining how to run scripts.
- Your final paper in PDF format generated using the LaTeX template specified above.

Do not upload your dataset, we have this! Any feature extraction or re-writing of data into other formats should all be reproducible within the code. The submission **deadline is: 9:00am, 24th of April 2023**.

(ii) short video (5 min maximum) describing your work and report with a walkthrough of your Jupyter notebook (see above for further details). Marks will be deducted if video file is missing.

4.2 Plagiarism

All submissions will be checked for plagiarism. Please review the School's plagiarism policy here. Any suspected plagiarism will be dealt with accordingly.

5 Grading guidelines

You find below a set of guidelines relating to the project evaluation. These are indicative and non-binding.

5.1 Code (40 marks)

Algorithmic correctness (10 marks)

- Breadth & depth of investigations e.g. implementation of various models...
- Model output making sense.
- Implementation matching the description in the paper...

[2]Use either the standard Pytorch format or H5 format. More details on tensorflow to H5 here: `https://www.tensorflow.org/tutorials/keras/save_and_load`

Experimental correctness (10 marks)

- Quality of implementation.
- Correctness of methodology: appropriate use of training, test and validation set (split with seeds if not split already or according to instructions), data shuffling, correct pre-processing...
- Searching hyperparameter space: fair hyper-parameter tuning across all algorithms.

Reproducibility (10 marks)

- Inclusion of README.
- Inclusion of clear comments and (package) requirements.txt.
- Data processing reproducibility i.e. can it be run all over again automatically?
- Model reproducibility: if models are run again on another machine fulfilling requirements, roughly the same performance will be achieved[3].

Modelling and Innovation (10 marks)

- Algorithmic novelty and/or experimental novelty.
- Use of several models in comparison.
- Transfer learning if appropriate.
- Interesting data pre-processing.
- Exploring the use of raw data...etc – essentially anything *'interesting'* !

5.2 Report (60 marks)

Reports will be graded based on correctness and clarity as follows:

- Background research, Related work and References (10 marks)
- Experimental setup (15 marks)
- Results section (15 marks)
- Conclusion & Future work (5 marks)
- Evaluation & Achievements (10 marks)
- Literacy & clarity, Quality of illustrations, Graphs and tables (5 marks)

5.3 Video (30 mark deduction if missing)

- MP4 format using H.264 codec
- Screen capture with embedded video of yourself
- See above for further details

[3] ± some variance since models are randomly initialised.

6 Resources

6.1 Report: section content

- *Abstract:* General, short overview of your paper (draw in the readers attention).
- *Introduction:* Introduce the problem in more detail, the motivation for your work, a brief description of your approach and findings along with a description of how the rest of the paper will proceed.
- *Related work:* A short summary of past approaches to this problem, the state-of-the-art (if available). You should compare and contrast your own work to these existing works. Referenced results and conclusions should be used to motivate your experiments.
- *Experimental setup:* A detailed description of the dataset used, preprocessing applied (if any), proposed algorithm, baseline approach, hyper-parameter tuning done...etc. From this section, the reader should understand exactly what you have done without reading your code.
- *Results:* Explain evaluation technique and motivation behind using it (i.e. Accuracy, MSE, F1-score,...etc.) Compare your approach to baselines (i.e. a simpler model) or state of the art.
- *Conclusion & Future work:* An overview of your main findings. You should also propose how this model might be improved in future (i.e. what would you do if you had more time or could do the project again?).
- *References:* Reference any papers that you used in your related work or as a reference for your approach.

6.2 DL Frameworks and LaTeX

Pytorch:

- *Download:* `https://pytorch.org/`
- *Tutorials:* `https://pytorch.org/tutorials/`
- *Forum:* `https://discuss.pytorch.org/`

Tensorflow:

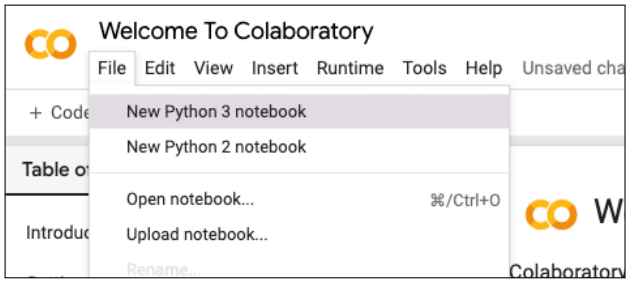
- *Download:* `https://www.tensorflow.org/install`
- *Tutorials:* `https://www.tensorflow.org/tutorials`
- *Forum:* `https://www.tensorflow.org/community/forums`

LaTeX:

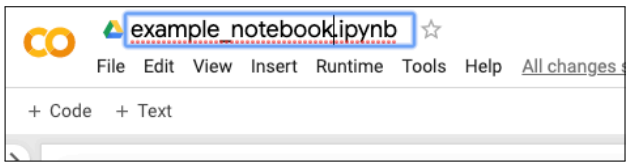
- *Online compiler:* `https://www.overleaf.com/`
- *Tutorials:* `https://www.overleaf.com/learn/latex/Tutorials`

6.3 Google Colab

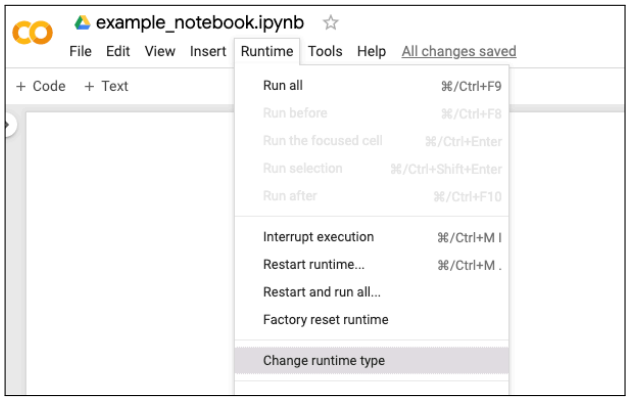
– Create new notebook



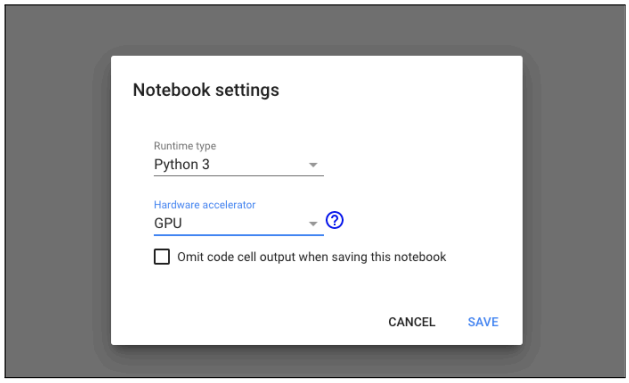
– Rename notebook



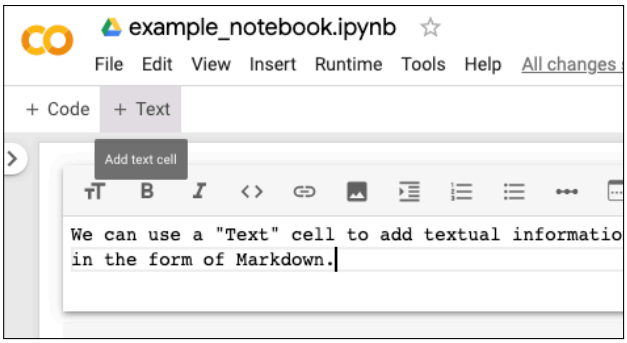
– Change runtime in order to switch between CPU, GPU and TPU



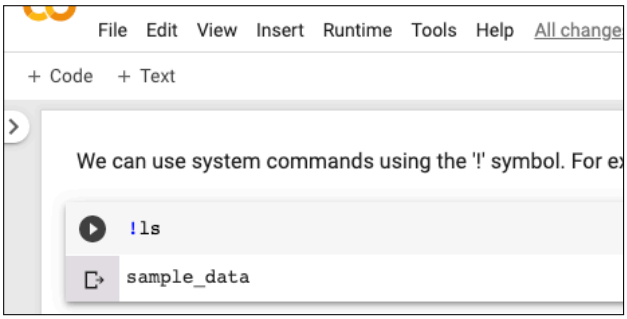
Select either None, GPU or TPU



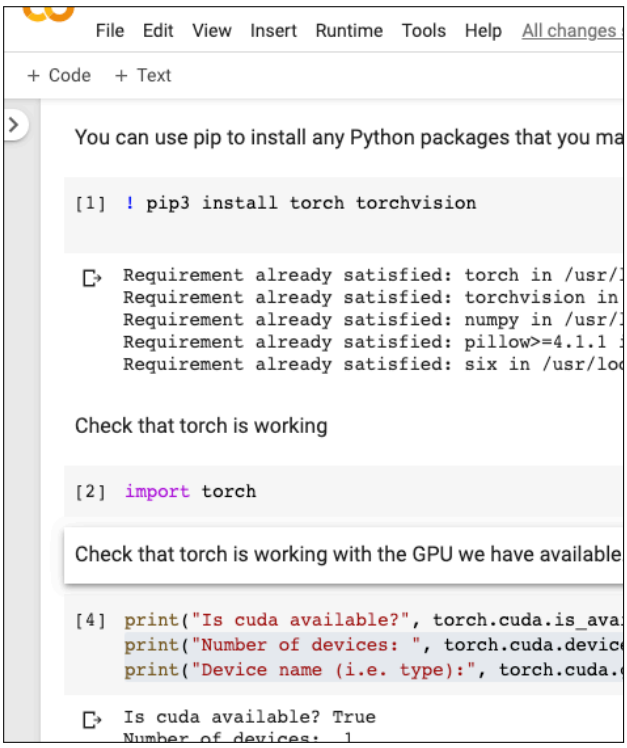
– Write details about code or additional information using the text cells.



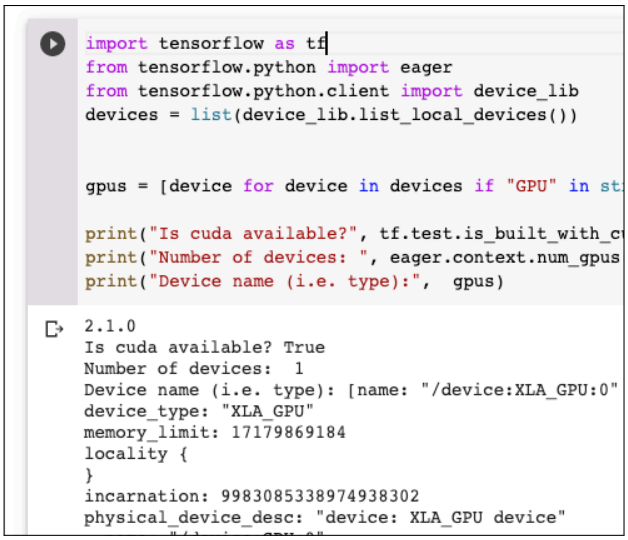
– You can use system commands (i.e. linux commands) using the '!' symbol in code cells.



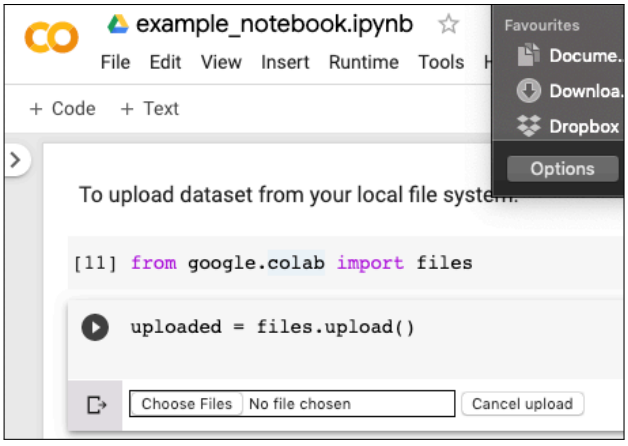
– Installing Pytorch and using it with a GPU.

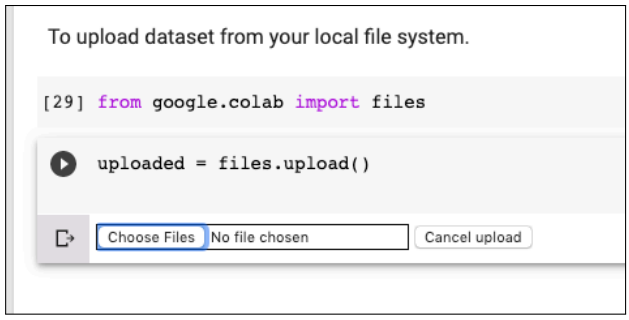


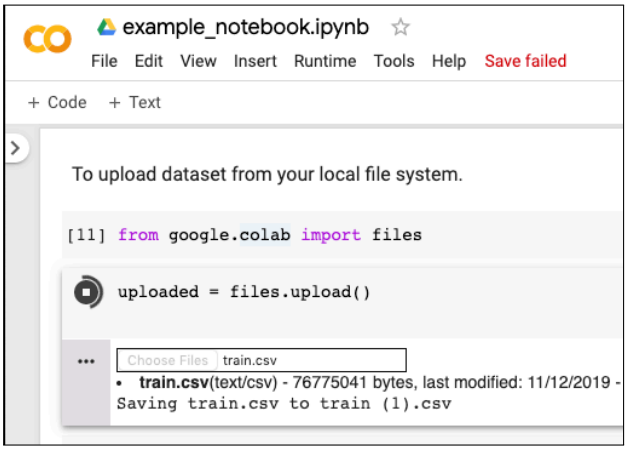
– Installing Tensorflow and using it with a GPU.



– Uploading data from your local disk.







– You can connect your Colab notebook to your google drive: https://colab.research.google.com/drive/

– You can also use the kaggle API to download data: https://www.kaggle.com/docs/api

---

oOo