Deep Learning – Assignment 2025

Value: 60% - Due Date Wednesday 23rd April at 8pm

Version 1.0

Overview

This is the specification for the main assignment of the Deep Learning class.

The goal of the project is to perform a systematic investigation of a number of Deep Learning methods in the context of image processing tasks.

This assignment has been designed to give you a range of task elements that you can use to build up your skills in Deep Learning. As such this project specification is detailed and descriptive.

The outputs for this project will be a report, source code, and trained models. Details on what is to be included in each of these pieces is detailed below. The report along with a code archive (.zip or .tar.gz), is to be submitted via Brightspace. The code archive should not include your trained models - instead demo code should be able to retrieve your models from a public URL as needed.

Your implementation should be made in Keras / TensorFlow or PyTorch as you see fit with Notebook files such that your code can be opened and run within Google Colab -- though it is noted that training may in practice need to be run on a local machine for full training to take place.

Note that you cannot use any alternative data set or coding framework to what is specified in this assignment brief.

Task Specification

Part 1: Simple Age Estimator - 50% (of project total)

The core of this project is based around a simple task – building and evaluating a model for age estimation based on the “facial age” dataset on Kaggle:

https://www.kaggle.com/datasets/frabbisw/facial-age

Note that there are a number of facial age datasets on kaggle, so please make sure you are using the link above to get the correct one.

The dataset has a simple structure consisting of numerous directories corresponding to individual ages.

The first task will simply be to prepare the dataset for suitable processing and save for reuse. There are several ways that this can be done, but make sure to do the following:

(a) Save out 10% of the data to be used as a test set. This test set should not be used for validation but instead be held out for end testing. Remember that this should be randomly selected.
(b) Make a training validation set between the remainder of the dataset. The specific split here is up to you, but a validation set of roughly 10-20% of the total data probably makes sense for a dataset this size.
(c) Folder names give the ages for each group but you will want to create a CSV file that actually contains the ages explicitly. While you are creating this, you should also create additional explicit labels to class the ages into broad groups: e.g., child 0-12, teen 13-19, youth 20-30, mid: 31-45; mature 45-60; older 61+. The specific categories you use are up to you, but try to use a smaller well balanced set of categories to improve performance below on the classification task.

This main part of the task is to perform a number of analyses based on training from scratch to predict age both as a real-valued number and age-category as a classification task.

Below I've set out several different comparisons that you are to perform. You can think of these as being individual chunks of analysis. These do not necessarily build on each other. It is up to you to decide whether these are done in isolation or not. The key point is that you understand yourself what you are doing, and that this understanding is reflected both in well documented code, and in your report.

Your models should be designed to minimize overfitting as appropriate.

You should consider the investigation of hyperparamters, though full grid search will not be feasible.

This initial set of models should be based on a hand-crafted network only. Pre-trained networks for this task do exist, and backbone networks could improve performance a lot, but the goal here is for you to explore network variations for yourself.

In all cases you should record your results as graphs for Training and Validation data and report the test result after training has been completed. Please select whatever loss functions or metrics you think are appropriate based on notes provided in class, or more widely based on what is appropriate in a regression or classification task.

CNN Kernel Variations

- Use different activation functions.
- Use different kernel sizes.

Network Variations:

- Use different network depths.
- With and without pooling.
- With and without skip connections.
- Using different numbers of fully connected layers.

Dataset Variations:

- 1 channel vs 3 channel
- Data Augmentations to enhance the dataset

Variation of your choice – come up with your own variation in model design or architecture for investigation. A wide amount of variation is possible here.

Model variations both for regression (specific age) and age category should be produced. For age-category, investigate the different types of output layer activation functions that are possible.

Model Saving
From the various models above, save the two top performing models. Your demo code should be able to load these saved models and demonstrate performance on the held-out dataset.

Your test dataset should be saved and made available for your demo code to use via a google drive download or similar.

Part 2: Using Backbone Models and Fine Tuning (35%)

In Part 2 of the assignment, we switch from a direct focus on the age prediction task and instead look at the production of a pre-trained model based on an autoencoder which you will then apply to the age categorization task.

For this part of the task, you should take the non-test data and split it into two subsets Block 1 and Block 2. One of these subsets will be used to train the autoencoder and the other will be used to build an age classification model. Your first task is to make these subsets and save these subsets to disk for use later. Make sure to randomly select the images.

As with Part 1, the set of tasks below do not build on each other and can in some ways be considered independent.

Autoencoder Modelling: For Block 1 Images

Building on your best performing model to this point, construct an autoencoder model to attempt to reproduce input images. Discuss the appropriate use of loss functions and metrics for evaluation.

Save the model notebook for the demo notebook and use later.

Transfer Learning for Block 2 Images

Using the model saved above, investigate the use of transfer learning to build a network for Block 2 images for the age classification task. Compare results for Transfer Learning based implementations to those from your original models from part 1.

Backbone Model Re-Use

Using the Block 1 data, fine tune a generic image processing model to perform the age category classification task. Note that the generic model should for example be pre-trained on imagenet eather than a specific age classification task. Compare this backbone-based model to your own homegrown models.

Part 3: Bias in Data (15%)

Using the original model architecture from part 1, and the original data splits, train a model where one gender or another has been omitted from the training / validation data, but has been left in the test data. Compare such a model to another model trained with roughly the same number of training instances. This approach only has to be tried for the age classification task and build this around your best performing architecture / network design from part 1.

Document Specification

Your document should be 13-15 pages in length and formatted as per the current document, i.e., standard margins, 11pt Arial font, 1.15 spacing between lines, and a 50% spacing (approximately) between paragraphs.

The first page is a standalone cover page. The cover page should include:

- Your name
- Your student number
- A link to a zip file that contains all your code (not models) collected data files etc.
- Links to the models
- The following statement:

“I confirm that the document and related code here is all my own work and that I did not engage in unfair practice in any way.”

Your code submission should consist of two elements. First each component above should have its own training notebook. This notebook should clearly demonstrate the data processing and training of your models. Saving and testing of your work can also be explored in these notebooks. In addition to these individual notebooks, one single extra notebook should be included as a demo notebook. The demo notebook should not contain training code, but should contain code to load your best models from the web, load your test data from the web, and demo the models at work. Submitting a non-working version of this demo notebook will impact marking.

When submitting notebooks, make sure they are purged of data. Otherwise, notebook files can be REALLY BIG and be a problem for you to submit via brightspace. Submitting notebooks with embedded data may lead to marks being lost.

This report should be a well written, suitably structured, and appropriately referenced document that justifies decisions, provides results as well as your analysis. The report should not give a background explanation of Deep Neural Networks or related models.

The following sections should minimally be included:

- Part 1
- Part 2
- Part 3
- Discussion

Appendices should be avoided. Do keep in mind that the report is important should not simply be a reiteration of the code.

Submission Details

This assignment is due for submission at the date shown at the top of this document. Extensions will not be granted except in the case of documented and approved personal circumstances. Late submissions will be deducted 2 percent per day for each of the first two days late, and then 1 percent per day for each day thereafter until a final deadline of May 5th.

All taught (not research) students should be available to answer questions on their assignment on dates to be confirmed in weeks 1 or 2 of May (subject to change). Arrangements for this will be made later, but are expected to be on MS Teams.

Marking Rubric

Marks for Parts 1, and 2 of this assignment are split in the ratio shown in the section headings above. For each Part, marks will in turn be broken down in a split between documentation and quality of modelling / analysis. The Rubric below details how marks will be split out of 100 for each of the three parts.

0 - 19: The student failed to provide a working implementation of code for achieving this part of the assignment -- or an unacceptable amount of code or documentation indicated unfair practice.

20 - 39: The student provided a working implementation of this Part of the assignment, but the documentation or interview failed to show the student had a clear understanding of the methods employed.

40 - 59: The student provided a working implementation of this Part of the assignment, and the documentation and interview showed that the student had a clear understanding of the methods employed.

60 - 79: The student provided a high-quality implementation of this part of the assignment with a high quality report that was detailed in analysis and well-motivated.

80 - 100: The student provided an excellent implementation of this Part of the assignment with a report of quality that is worthy for submission for publication at a national conference. To reach this grade the work will generally need a non-trivial original element that is not already covered by the assignment specification -- this is in addition to meeting the other criteria.

All students should be able to defend / explain their submission orally when requested to do so. Any failure to do so will be judged as a failure in the documentation to show the student had a clear understanding of the methods employed. It should be noted that the interview does impact on the marking scheme.

Unfair practice of any type is not accepted. Do not copy text into your report without appropriate citations. Even with citations, no more than 10% of a report should be based around existing material. Note that this also includes ‘self-plagiarism’. Similarly, no copying blocks of code into your code without appropriate acknowledgement. Again, even with acknowledgements, only a small portion of code should be copied directly. Unfair practice of this type will be subject to a grade of 0 on relevant parts of the assignment, and, or, full disciplinary procedures. If you are in doubt - ask me.

Group Submission

Group submission of this project is possible. A group submission is a single submission made by one person on behalf of a team of 2 people from the same class cohort (i.e., PhD or MSc). It is not possible to have a team with a mix of PhD and MSc students. Note that larger teams are not possible.

There is a 4% absolute discount on the total grade for a group submission. Team members should contribute equally to all major aspects of the project. Each team member does however need to be able to attend an interview on the submission and answer all questions on that submission. In other words, while a group can work on a submission, it is not possible for a person to avoid understanding how any part of the submission works.

In the case of a group submission two additional criteria hold:

- Each team member needs to submit a 5 minute presentation (recorded with voice over) to give an overview of their own learning experience from the project highlighting what they believe worked well in the analysis and what did not.
- A peer review survey between group members also needs to be completed. The peer review survey results in a weighting that can adjust individuals within a group up and down.