Machine Learning Practical: Courseworks 3 & 4

Release date Friday 27 January 2017

Due dates

1. Baseline experiments (Coursework 3) – ***16:00 Thursday 16th February 2017***
2. Advanced experiments (Coursework 4) – ***16:00 Thursday 16th March 2017***

1 Introduction

Courseworks 3 & 4 in MLP will take the form of a mini-project that will run through to week 8 of semester 2. The aim of the coursework is to explore deep learning and neural network techniques for classification using *one* of two datasets:

- CIFAR-10 and CIFAR-100 – object recognition in images;
- Million Song Dataset – music genre recognition from audio features and metadata for a subset of a million contemporary popular music tracks.

We shall also move to a new software framework, TensorFlow (https://www.tensorflow.org). Please make sure you have completed MLP notebook 08, Introduction to TensorFlow, before starting the coursework.

You should treat courseworks 3&4 as single mini-project, using the same dataset and task throughout, for which you submit two reports – an initial report (coursework 3, due 16 February) and a final report (coursework 4, due 16 March).

2 Datasets

You should work on one of the two datasets. CIFAR–10/100, which we use for object recognition in images, will require some kind of spatial modelling in order to get good classification performance. The Milllion Song Dataset, which we use for recognising the genre of music recordings, will require some kind of temporal or sequential modelling to obtain good classification performance.

Object recognition using CIFAR–10 and CIFAR–100

CIFAR-10 is a very well-studied dataset for object recognition in images, that was collected at University of Toronto as a labelled subset of the MIT 80 million tiny images dataset (http://people.csail.mit.edu/torralba/tinyimages/). It comprises 60,000 colour images (3x32x32 pixels), with 10 object classes. This has been split into a training set of 40,000 images, a validation set of 10,000 images and a test set of 10,000 images. CIFAR-100 is similar to CIFAR-10, but has 100 classes; the training set size is again 40,000 images with validation and test sets each containing 10,000 images.

For more information about using the CIFAR-10 and CIFAR-100 datasets for this coursework please see notebook 09a, Object recognition with CIFAR-10 and CIFAR-100. This notebook also includes how to access the data, information about the data providers we have added to `mlp.data_providers`, and an example TensorFlow network to train a two-layer classifier.

Music genre classification using the Million Song Dataset

The Million Song Dataset contains precomputed features and metadata (labels) for a million songs. The data has been used for different tasks, typically using subsets of the data. For this coursework we will use the data for music genre classification. As for object recognition we provide two tasks which differ in the number of categories – in this classification into 10 or into 25 genres.

For MLP, we represent each song as a sequence of non-overlapping *segments* – each segment corresponds to a "quasi-stable music event", with typical duration of less than half a second (although some segments can have a duration of several seconds). There is thus a variable number of segments that makes up a song. A segment is represented by a 25-dimension feature vector with 12 chroma features (relating to pitch classes), 12 timbre features (similar to MFCC features used for speech processing), and a loudness feature.

A song is represented using a variable length sequence of segments, something which is best modelled using a recurrent neural network, or some other architecture capable of sequence modelling. To allow easier integration in to standard feedforward models, we also provide a "fixed-length" version of the data which includes features only for a fixed length crop of the central 120 segments of each track. This gives an overall input dimension per track of 120×25=3,000. Each of the 3,000 input dimensions has been been preprocessed by subtracting the per-dimension mean across the training data and dividing by the per-dimension standard deviation across the training data.

For full details about using the the Million Song Dataset for this coursework please see notebook 09b, Music genre classification with the Million Song Dataset. This notebook also includes how to access the data, information about the data providers we have added to `mlp.data_providers`, and an example TensorFlow network to train a two-layer classifier.

3 Code

A branch with the introduction notebooks described above and the updated `mlp.data_providers` module is available on the course Github repository on a branch `mlp2016-7/coursework3-4`. To create a local working copy of this branch in your local repository you need to do the following.

1. Make sure all modified files on the branch you are currently on have been committed (see details here if you are unsure how to do this).
2. Fetch changes to the upstream `origin` repository by running
   `git fetch origin`
3. Checkout a new local branch from the fetched branch using
   `git checkout -b coursework3-4 origin/mlp2016-7/coursework3-4`

You will now have a new branch `coursework3-4` in your local repository.

4 Experiments

The aim of this mini-project coursework is to design, implement, and report on classification experiments on one of the two datasets. Your experiments should be designed such that they investigate specific research questions. The mini-project falls into two parts.

Part 1: Baseline experiments (coursework 3)

The main aim of the first part of the mini-project is to build baseline systems for the dataset you have chosen to work on. In this part you should investigate feed-forward networks, keeping any investigations into more complex architectures (e.g. convolutional networks and recurrent networks) until part 2. (For the Million Song Dataset, this means that you should use the fixed-length variant of the data.) Things you could explore in part 1 include different activation functions, different hidden layer depths and widths, learning rate schedules, normalisation, and regularisation. You should design and carry out a number of experiments, making it clear in your report what research questions are being investigated. You should present the results clearly and concisely in your report, and provide a discussion of the results, with conclusions related to the initial research questions.

You should also include a 'Further Work' section which gives your plan for part 2 of the mini-project. This should include the research question(s) you will investigate, the methods (network architectures, etc.) you plan to use, and the experiments you will carry out.

Coursework 3 is worth 25% of the overall mark. The marking scheme is as follows:

- Research questions to be investigated (10 marks). Clear presentation of the research questions you are exploring, and the experiments you designed to address these questions.
- Methods (30 marks). Clear presentation of the methods (network architectures, learning schedules, etc.) that were used, and an outline of how they were implemented in TensorFlow.
- Results and discussion (40 marks). Clear presentation of results, discussion and interpretation of results, conclusions.
- Presentation and clarity of report (20 marks). Marks awarded for overall structure, clear and concise presentation, providing enough information to enable work to be reproduced, clear and concise presentation of results, informative discussion and conclusions.

Please note that the Further Work section discussed above is required, but will not contribute to the numeric mark.

Part 2: Advanced experiments (coursework 4)

In the second part of the project you will explore more advanced approaches for the dataset you have chosen to work, based on your plan presented in the Further Work section of Part 1. For example, you may wish to explore different network architectures (for example, recurrent networks or convolutional networks), more advanced approaches to regularisation (for example there have been many variants of dropout proposed), the use of the variable length song data for music genre classification, trying to learn specific invariances for object classification, or the use of teacher-student approaches. There are of course many possibilites! It is a good idea if you can use your results in part 1 as a baseline against which you can compare your results in this part.

However please choose an approach which takes account of the constraints regarding time (you have 3–4 weeks for part 2, and you will be doing other courses) and available compute resources (we can provide access to a shared CPU cluster - the `scutter` cluster used in Extreme Computing - but cannot offer GPU access for this course). It is perfectly reasonable, for example, to set a constraint whereby you limit yourself to experiments with a training time below a certain threshold.

Your final report should present the research questions that you investigate, and the experiments you designed and carried out; you should also concisely describe the baseline systems developed in phase 1. You should present the results clearly and concisely, comparing to your baseline systems and provide a discussion of the results, with conclusions related to the research questions. The conclusions section might propose some further work based on the results of this mini-project.

Coursework 4 is worth 40% of the overall mark. The marking scheme is as follows:

- Research questions to be investigated (10 marks). Clear presentation of the research questions you are exploring, and the experiments you designed to address these questions.
- Methods (30 marks). Clear presentation of the methods (network architectures, learning schedules, etc.) that were used, and an outline of how they were implemented in TensorFlow.
- Results and discussion (40 marks). Clear presentation of results, discussion and interpretation of results, conclusions, further work.
- Presentation and clarity of report (20 marks). Marks awarded for overall structure, clear and concise presentation, providing enough information to enable work to be reproduced, clear and concise presentation of results, informative discussion and conclusions.

Please note that a concise description of the baseline systems developed in part 1 is required, but will not contribute to the numeric mark.

Submission

Your coursework submission should be done online using the `submit` command available on DICE machines. Your submission should include

- your completed course report as a PDF file,
- if you used a notebook, the notebook (`.ipynb`) file(s) you use to run the experiments in
- and any (`.py` files) you have written or modified to carry out the assignment

Please do NOT include a copy of the other files in your `mlpractical` directory as including the data files and lab notebooks makes the submission files unnecessarily large.

You should EITHER (1) package all of these files into a single archive file using `tar` or `zip` and then (for coursework 3) submit this archive using

```
submit mlp 3 coursework3.tar.gz
```

OR (2) copy all of the files to a single directory `coursework3` directory and then submit this directory using

```
submit mlp 3 coursework3
```

Obviously for courserwork 4, you would do, e.g.

```
submit mlp 4 coursework4.tar.gz
```

You can amend an existing submission by rerunning the `submit` command any time up to the deadline.

5 Computing

Some of your experiments may take significant compute time, so it is in your interest to start running experiments for these courseworks as early as possible.

For this coursework we plan to provide access to the `scutter` cluster of machines used for Extreme Computing. We shall release a Computing Note on using this platform (which will use the `gridengine` job scheduler), and on using remote machines generally, by Friday 3 February.

Please note that it is optional whether you use a Jupyter Notebook for the mini-project. Using a notebook does have the advantage of organising your experiments, code, and results; it has the disadvantage of requiring a bit of overhead in running jobs on other machines (this will be discussed in the Computing Note).

6 Kaggle in Class

We have setup optional Kaggle in Class competitions for the CIFAR-100 and 25-class music genre classification tasks (you will receive an invitation email and you will need to sign-up with your ed.ac.uk email address). This will produce a classification leaderboard, in which the scores for the competition are calculated as the proportion of classes correctly predicted on the test set inputs (for which no class labels are provided). Half of the 10000 test inputs are used to calculate a public leaderboard score which will be visible while the competition is in progress and the other half are used to compute the private leaderboard score which will only be unveiled at the end of the competition. Each entrant can make up to two submissions of predictions each day during the competition.

The final section of each of the notebooks on CIFAR and the Million Song Database gives some simple code and helper function as an example of how to create a submission file which can be uploaded to Kaggle.

Entering the Kaggle competitions is optional and will not affect your grade for the course.

7 Please Note

Academic conduct: Assessed work is subject to University regulations on academic conduct: http://web.inf.ed.ac.uk/infweb/admin/policies/academic-misconduct

Late submissions: The School of Informatics policy is that late coursework normally gets a mark of zero. See http://web.inf.ed.ac.uk/infweb/student-services/ito/admin/coursework-projects/late-coursework-extension-requests for exceptions to this rule. Any requests for extensions should go to the Informatics Teaching Office (ITO), either directly or via your Personal Tutor.

Backing up your work: It is **strongly recommended** you use some method for backing up your work. Those working in their AFS homespace on DICE will have their work automatically backed up as part of the routine backup of all user homespaces. If you are working on a personal computer you should have your own backup method in place (e.g. saving additional copies to an external drive, syncing to a cloud service or pushing commits to your local Git repository to a private repository on Github). **Loss of work through failure to back up does not consitute a good reason for late submission**.

You may *additionally* wish to keep your coursework under version control in your local Git repository on the `coursework3` branch. This does not need to be limited to the coursework notebook and `mlp` Python modules - you can also add your report document to the repository.

If you make regular commits of your work on the coursework this will allow you to better keep track of the changes you have made and if necessary revert to previous versions of files and/or restore accidentally deleted work. This is not however required and you should note that keeping your work under version control is a distinct issue from backing up to guard against hard drive failure. If you are working on a personal computer you should still keep an additional back up of your work as described above.