DENNIS PRANGLE

INTRODUCTION TO DEEP LEARNING USING TENSORFLOW: PRACTICALS

JUMPINGRIVERS.COM

Preliminaries

Structure of handout

Most of the chapters in this document cover several chapters of the notes. Each chapter here has a **main exercise** with a Jupyter notebook containing a solution which you can consult after doing the exercise, or if you get stuck! Each chapter also has several optional exercises without solutions. For chapters A-C there are also notebooks detailing how to create/download the dataset required for the main exercise.

Python packages

The following packages are used in most examples:

- Tensorflow, version 1.4 or later (The latest version 1.13 causes some deprecation warnings, but these can simply be ignored.)
- Keras (I recommend installing the standalone library rather than using the version included with tensorflow)
- numpy (Version 1.14 causes some warnings when used with tensorflow but these can simply be ignored.)
- matplotlib

The following packages are also used briefly in some examples:

- sklearn
- h5py

A Linear and logistic regression in Keras (Chapters 1 and 2)

Main exercise: concentric circles

- Fit logistic regression to the **concentric circles** dataset. Code to create this is given in a notebook.
- It is hard to classify this dataset by logistic regression using the input variables provided. Can you come up with some data **features** (i.e. transformed variables) to improve performance?

Optional exercise: Boston housing example

- Compare the Keras results in Section 2.1 with linear regression output from `sklearn.linear_model.LinearRegression`.
- Investigate how weight penalisation affects the Boston housing linear regression results.

B Neural networks in Keras (Chapters 3 and 4)

Main exercise: concentric circles

- Fit a neural network to the concentric cirlces dataset. Unlike logistic regression, you should be able to get good performance without feature engineering.
- Investigate how the classification boundary changes for different choices of activation function.

Optional exercise: Boston housing example

- Try the Boston housing example again with neural network regression.
- Investigate how your results differ with identity/softplus activation for final output.

C Training in Keras (Chapters 5 and 6)

Main exercise: fashion MNIST

The **fashion MNIST** dataset is similar to MNIST but, instead of images of digits, it is made up of images of fashion items (T-shirts, dresses, trainers etc.) Otherwise, its format is designed to be as similar to MNIST as possible.

Use a similiar neural network to Section 6.5 of the notes to compare the performance of different optimisers. For example, compare SGD with a constant learning rate and a decaying one. When using SGD, it will take some experimentation to find a good learning rate. You can also try algorithms with adaptive learning rates like Adam.

It's a good idea to compare how these methods perform on **training loss** rather than validation loss. While validation loss is our best judge of overall performance, it is affected by things like overfitting and regularisation choices. Here we just want to look at optimiser performance and this is best measured by training loss, since this is the criterion we ask the optimiser to minimise.

Optional exercises

- Use the tensor playground website `http://playground.tensorflow.org` to get some intuition for how training choices affect neural network performance.
- Use tensorboard on any example we've been through so far.
- Try to improve the performance of the MNIST and fashion MNIST examples.

D Tips for neural networks in practice (Chapter 7)

Main exercise: MNIST

Revisit the MNIST example of Section 6.5 and:

- Implement early stopping.
- Investigate whether not preprocessing the data reduces performance.

Optional exercises

Using any dataset, investigate the other techniques from this chapter:

- Feature engineering
- Gradient clipping
- Investigate whether sorting the training data by one of the variables – an extreme case of not shuffling! – reduces performance.

E Low-level Tensorflow (Chapter 8/Appendix A)

If you are less interested in low-level tensorflow then feel free to skip these exercises and instead do some of the optional exercises for the earlier chapters.
Note: you can do this exercise with (see Chapter 8) or without (see Appendix A) eager execution.

Main exercise: Boston Housing

Code up a multilayer perceptron in low-level tensorflow and use it to fit the Boston Housing dataset again. Hint: activation functions are within the `tf.nn` package `https://www.tensorflow.org/api_docs/python/tf/nn`.

Optional exercises

Extend the multilayer perceptron to do the following:

- Penalise the weights using L_1 penalisation, or something more unusual, like penalising the maximum weight, which is not available as a default in Keras.
- Use $\tan^{-1}$ as an activation function.
- Train using mini-batches.
- Print the validation loss during training.

F Convolutional neural networks (Chapters 9 and 10)

Main exercise: IMDB

Try to improve performance on the IMDB example from Chapter 10.
Some things to try are:

- Various types of regularisation
- Gradient clipping
- Different architectures
- Alter `max_words` and `max_len`

Optional exercises

- Try to improve performance on the MNIST example from Chapter 10
- Adapt the MNIST example to the fashion MNIST dataset (see Chapter C above) and try to optimise performance