Assignment 2: Autoencoders

Welcome to your second assignment. Upload the completed scripts task1.py and task2.py and the document assignment2.pdf to the VLE before the deadline.

Standard criteria for valid assignments

In general, an assignment is considered valid if the following criteria are met:

- The delivered program uses Python and Tensorflow.
- Only one code file is used per task, which is a modified version of the provided script, with the same file names. Apart from the two code files, you should also provide a PDF with screenshots of the charts shown in each task.
- The way the final test error is calculated in the provided script is left as is.
- Only the provided dataset is used and without modification (although the data may be transformed for use by the model).
- Your program may not run for longer than 10 minutes on CPU.
- The delivered program code is organised and easy to understand.
- The delivered program does not use any random number generator seeds.
- When the delivered program is run, the model is trained from scratch using gradient descent (or an extension of gradient descent).
- The test set does not influence training in any way.
- The delivered program code is not excessively modified. I will be correcting your program by checking what was changed from the original provided scripts and needless modification makes corrections slower.
- At the end of training you will be shown the test accuracy, duration, and number of parameters in your model. Your aim should be to deliver the fastest, smallest, and best performing program possible with the given task constraints.

You will get marks for:

- Neatness.
- Using correct implementations of techniques shown in class.
- Use of sensible hyperparameters.
- Use of regularisation.
- The model's general performance.
- Any interesting ideas you include (for bonus marks).

Task 1: Autoencoder

Summary: Learn a simple autoencoder neural network that generates MNIST handwritten digits.

You are provided with a file "dataset.txt" that contains a collection of flat bitmap images of hand written digits together with a half-finished Python script "task1.py". Your task is to complete the script in order to learn a simple autoencoder using gradient descent with Tensorflow which accurately regenerates the given digits in the dataset. You may split the dataset into separate validation sets if needed and you may use any technique discussed during the lectures.

The script also includes code to project the thought vector into a 2 dimensional space using the T-SNE algorithm such that the closer the dots are in the 2 dimensional space, the more similar the thought vectors were. Different coloured dots represent thought vectors of different digits. A good autoencoder should cluster digit thought vectors in a sensible way such that similar looking digits appear closer to each other than other digits.

Accuracy is the percentage of pixels in the output that match the pixels in the input after rounding. A good accuracy is 97%.

Constraints

This task in particular is considered valid if the following criteria are met:

- The model is a simple autoencoder neural network.
- The thought vector is 256 elements long with a tanh activation function.
- Output uses sigmoid.

Task 2: Multi-task learning

Summary: Learn a multi-task model that performs two tasks given an MNIST handwritten digits: acts as a simple autoencoder neural network and classifies the digit encoded in the thought vector of the autoencoder.

Like the previous task, create a simple autoencoder. From the thought vector extend separate layers to classify the digit encoded in the thought vector (do not use a separate input, the idea is to force the representation of the thought vector to be useful for both generating the digit and classifying it). The script's name is "task2.py".

Note that it might be necessary to use one of the advanced optimisers for this task rather than simple gradient descent. Accuracy is the percentage of pixels in the output that match the pixels in the input after rounding. A good accuracy is 97%.

Constraints

This task in particular is considered valid if the following criteria are met:

- Part of the model is a simple autoencoder.
- Another part of the model is a classifier that starts from the autoencoder's thought vector such that the thought vector is a shared layer.
- All information about the image being passed to the classifier must come from the thought vector only.
- The thought vector is 256 elements long with a tanh activation function.
- Output of the generator uses sigmoid and output of the classifier uses softmax.