DAT405 Introduction to Data Science and AI

Assignment 7 Neural Networks with Keras and TensorFlow:

Download the notebook for this assignment. The notebook will provide a basis that you can use to solve the exercises.

This assignment will work with the MNIST data set. The MNIST dataset is a standard benchmark where small 28x28 pixel grayscale images of handwritten digits. Each image was manually assigned to a class label, an integer from 0 to 9, by the US Census Bureau. The task associated with the dataset is building **a model that takes a new image (of the same size) and returns a class label – that is, an integer from 0 to 9**.

For this assignment, we will use the Keras framework to construct our neural networks. You can read more about the Keras framework here https://keras.io/. You can find information regarding the different layers and regularizers there. In this assignment, you can use free GPU leases on Google Colab or deepnote to speed up training.

If there are any questions regarding the assignment or the Keras/TensorFlow framework, please contact the TAs. Use the lab-sessions to ask questions!

There will be 10 points for this assignment.

1) **1 point. *Preprocessing*.**
   In the notebook, the data is downloaded from an external server imported into the notebook environment using the `mnist.load_data()` function call.
   Explain the data pre-processing high-lighted in the notebook.

2) **4 points. *Network model, training, and changing hyper-parameters*.**

   A) How many layers does the network in the notebook have? How many neurons does each layer have? What activation functions and why are these appropriate for this application? What is the total number of parameters for the network? Why does the input and output layers have the dimensions they have?

   B) What loss-function is used to train the network? What is the functional form (mathematical expression) of the loss function? and how should we interpret it? Why is it appropriate for the problem at hand?

   C) Train the network for 10 epochs and plot the training and validation accuracy for each epoch.

   D) Update model to implement a three-layer neural network where the hidden-layers has 500 and 300 hidden units respectively. Train for 40 epochs. What is the best validation accuracy you can achieve? – Geoff Hinton (a co-pioneer of Deep learning) claimed this network could reach a validation accuracy of 0.9847 (http://yann.lecun.com/exdb/mnist/) using **weight decay** (L2 regularization of weights (kernels): https://keras.io/api/layers/regularizers/). Implement weight decay on hidden units and train and select 5 regularization factors from 0.000001 to 0.001. Train 3 replicates networks for each regularization factor. Plot the final validation accuracy with standard deviation (computed from the replicates) as a function of the regularization factor. How close do you get to Hintons result? – If you do not get the same results, what factors may

influence this? (hint: What information is not given by Hinton on the MNIST database that may influence Model training)

3) 2 points. Convolutional layers.

A) Design a model that makes use of at least one convolutional layer – how performant a model can you get? -- According to the MNIST database it should be possible reach to 99% accuracy on the validation data. If you choose to use any layers apart from convolutional layers and layers that you used in previous questions, you must describe what they do. If you do not reach 99% accuracy, report your best performance and explain your attempts and thought process.

B) Discuss the differences and potential benefits of using convolutional layers over fully connected ones for the particular application?

In the last question we will explore an unsupervised learning problem for denoising images. We will not be needing the labels during training of this model.

4) 3 points. Auto-Encoders for denoising.

A) The notebook implements a simple denoising deep autoencoder model. Explain what the model does: use the data-preparation and model definition code to explain how the goal of the model is achieved. Explain the role of the loss function? Draw a diagram of the model and include it in your report. Train the model with the settings given.

B) Add increasing levels of noise to the test-set using the salt_and_pepper()-function (0 to 1). Use matplotlib to visualize a few examples (3-4) in the original, "seasoned" (noisy), and denoised versions. (Hint: for visualization use imshow(), use the trained autoencoder from 4A to denoise the noisy digits). At what noise level does it become difficult to identify the seasoned digits for you? At what noise level does the denoising stop working?

C) Test whether denoising improves the classification with the best performing model you obtained in questions 2 or 3. Plot the accuracy as a function of noise-level for the seasoned and denoised datasets. Discuss your results.

D) Explain how you can use the decoder part of the denoising auto-encoder to generate synthetic "hand-written" digits? – Describe the procedure, implement it and show examples in your report.

What to submit

- All Python code written.
- A report that includes the figures produced and the descriptions/discussions that are requested in the questions.

If you upload a zip file, please also upload any PDF files separately (so that they can be viewed more conveniently in Canvas).

Self-check

Is all the required information on the front page? Have you answered all questions to the best of your ability? Anything else you can easily check? (details, terminology, arguments, clearly stated answers etc.?)

Grading will be based on a qualitative assessment of each assignment. It is important to:

- Present clear arguments
- Present the results in a pedagogical way

    - Should it be table/plot? What kind of plot? Is everything clear and easy to understand?
- Show understanding of the topics
- Give correct solutions.
- Make sure that the code is well commented.
    - Important parts of the code should be included in the running text and the full code uploaded to Canvas.