Industrial Engineering (Deep Learning) 874
Post-block Assignment 2: Modern Deep Learning Fundamentals
Department of Industrial Engineering

Deadline: 13 July 2020, 23:59

Total: 160

Instructions

The focus of this assignment is to test your understanding of the concepts covered in the lectures from days 2-4. In addition, your implementation of these concepts on real-world data will also be tested in this assignment.

- Answer all the questions below.
- Where asked to provide all calculations and steps in the methodology you used, please make sure that you do. Without these calculations and insights to your methodology, no marks will be given.
- Submit your typed answers as a pdf document. Please also submit all other documents required to obtain your answer. For instance, submit your Python script that you used for any of the questions.
- Please make sure that you do and submit your own work. Plagiarism will not be tolerated.
- Note that late submissions cannot be accepted and that no extensions to the deadline can be provided.

Stochastic Gradient Descent, Mini-Batch Training **[52]**

1. What is the order of the main steps for using a gradient descent algorithm? **[2]**

1. Calculate error between the actual value and the predicted value
2. Reiterate until you find the best weights of network
3. Pass an input through the network and get values from output layer
4. Initialize random weight and bias
5. Go to each neuron which contributes to the error and change its respective values to reduce the error

A. 1, 2, 3, 4, 5

B. 5, 4, 3, 2, 1

C. 3, 2, 1, 5, 4

D. 4, 3, 1, 5, 2

2. In this question, you will implement stochastic gradient descent and mini-batch training to derive the coefficients of a linear regression model for the provided data set. In addition, you will develop a simple MLP model to compare its performance to that of the linear regression model.

i) Use the data sets trainX and trainy to fit the linear regression model. Create all the required functions in order to fit the model based on the training data for both stochastic gradient descent and the mini-batch training algorithm. Clearly show all code and equations required to obtain the coefficients of the linear regression model. Use the mean squared error (mse) as the cost function. You do not need to tune the hyper-parameters (such as learning rate or batch size) to get the perfect model. A few (e.g. 5) iterations of tuning these parameters are sufficient. Plot the cost function versus the number of iterations for the various hyper-parameter settings that are implemented. **[30]**

ii) Using the same training data sets, develop and train a multi-layer perceptron (MLP) model to approximate the input data. Any architecture may be implemented, but reasons must be provided why the parameters were selected. Plot the mean squared error of the training set during training of the network. **[10]**

iii) Evaluate the linear regression models resulting from the SGD and mini-batch algorithms as well as the MLP model on the test data sets (testX and testy) and compare their performance. Provide insights into the performance of the various models. **[10]**

Autoencoders and CNN **[88]**

1. In this question, we will be using the popular MNIST Dataset. The MNIST database (Modified National Institute of Standards and Technology database) is a large database of handwritten digits that is commonly used for training various image processing systems. the black and white images from NIST were normalized to fit into a 28x28 pixel bounding box and anti-aliased, which introduced grayscale levels. Various autoencoders models will be developed and fitted to this data set to compare their performance and characteristics.

i) Import the MNIST dataset into your environment (packages for all main programming examples exist to download and import the data set). Use 60000 images for training and 10000 images for testing.

ii) Develop the following autoencoders: For *each* of these models, following the instructions in iii)
   a. Undercomplete autoencoder
   b. Stacked autoencoder
   c. Sparse autoencoder
   d. Denoising autoencoder
   e. Convolutional autoencoder

iii) For each of the models:
    a. Develop the model according to its description (provided in ii) and provide reasons for choosing model parameters (e.g. the model architecture, hyper-parameters, etc.) **[30]**
    b. Train each model using 60000 images of the data set and test with the remaining 10000. Report the mse for the test set and plot both the training and testing mse during training. **[25]**
    c. Plot 5 of the predicted images comparing the predicted image with the original image. **[10]**

iv) Compare the mse of the test set for the various models and comment on the performance of each. Draw insights from your observations. **[5]**

2. In this question we will use the Stanford dogs dataset. You will develop a convolutional autoencoder to fit to the data set. The Stanford Dogs Dataset has over 20k images categorized into 120 breeds with uniform bounding boxes. I have decreased the number of images to use to minimize cpu resources required during training.

I. Download the *dogs* folder which contains all the 64x64 pixel cropped images of the dogs from SunLearn. Load the images into your environment. (hint: you may want to use the *ImageDataGenerator* and *flow_from_directory* functionality from Keras).

II. Develop a convolutional autoencoder to fit the data. Clearly illustrate your process of determining the best model (within reasonable time limits). Discuss why the chosen model was selected among those developed. **[15]**

III. Using your best model, reconstruct any 5 of the images in the dataset and comment on its quality. **[3]**

Restricted Boltzmann Module **[20]**

1. Provide a detailed discussion of what Restricted Boltzmann Module/Machines (RBM) are. Specifically, refer to the following.
    a. Structure/architecture of an RBM. Illustrate this visually as part of your discussion. **[5]**
    b. Differences and similarities between autoencoders and RBMs. **[5]**
    c. A high-level overview of the contrastive divergence (CD) procedure. No detailed equations are required here. **[5]**
    d. Discuss 5 use cases for RBM and why it (RBMs) is better/worse than autoencoders for these cases. **[5]**