CSE/ECE 343/543: Machine Learning
Assignment-3 Neural Networks
Max Marks: 130

Instructions

- Keep collaborations at high level discussions. Copying/Plagiarism will be dealt with strictly.
- Late submission penalty: As per course policy.
- Your submission should be a single zip file **2018xxx_HW3.zip** (Where *2018xxx* is your roll number).
- Include only the **relevant files** arranged with proper names. A **.pdf report** explaining your codes with relevant graphs and visualization and theory questions.
- Do **NOT** include data files in your submission. It makes your files unnecessarily big while downloading.
- Ensure that everything required for a particular question is present in their respective files in terms of functions (not comments). Failure to do so would result in a penalty. Follow the following file structure for submission:

  2018xxx_HW3
  |– Q1.py
  |– Q2.py
  |– Q3.py
  |– Q4.py
  |– Report.pdf
  |– Weights (folder)
  |– Plots (folder)

- Remember to **turn in** after uploading on google classroom.
- Resolve all your doubts from TA's in their office hours **two days before the deadline.**
- **Document** your code. Lack of comments and documentation or improper file names would result in loss of 20% of the *obtained* score.

---

1. (55 points) You have to implement a general algorithm for Neural Networks. You can only use the numpy library. Use the attached **Q1.py** for implementing the algorithm.

    1. (6) The network should have the following parameters
    n_layers: Number of Layers (int)
    layer_sizes: array of size n_layers which contains the number of nodes in each layer. (array of int)
    activation: activation function to be used (string)
    learning_rate: the learning rate to be used (float)
    weight_init: initialization function to be used
    batch_size: batch size to be used (int)
    num_epochs: number of epochs to be used (int)
    2. (3+3+3+3+3) Implement the following activation functions with their gradient calculation too: ReLU, sigmoid, linear, tanh, softmax.
    3. (3+3+3) Implement the following weight initialization techniques for the hidden layers:
    zero: Zero initialization
    random: Random initialization with a scaling factor of 0.01
    normal: Normal(0,1) initialization with a scaling factor of 0.01
    4. (10+5+5+5) Implement the following functions with bias=0 and cross entropy loss as the loss function. You can create other helper functions too.
    fit(): accepts input data & input labels and trains a new model
    predict_proba(): accepts input data and returns class wise probability
    predict(): accepts input data and returns the prediction using the trained model
    score(): accepts input data and their labels and returns accuracy of the model

2. (35 points) Use the **MNIST** dataset for training and testing the neural network model created in Question 1 **ONLY**. Use the train dataset for calculating the training error and test dataset for calculating the testing error.

    1. (5) Use the following architecture [#input, 256, 128, 64, #output], learning rate=0.1, and number of epochs=100. Use normal weight initialisation as defined in the first question. Save the weights of the trained model separately for each activation function defined above and report the test accuracy.
    2. (12) Plot training error vs epoch curve and testing error vs epoch curve for ReLU, sigmoid, linear and tanh activation function. Finally, you should have 4 graphs for the 4 activation functions.
    3. (5) In every case, what should be the activation function for the output layer? Give reasons to support your answer.
    4. (2) What is the total number of layers and number of hidden layers in this case?
    5. (5) Visualise the features of the final hidden layer by creating tSNE plots for the model with highest test accuracy. You can use sklearn for visualization.

6. (6) Now, use sklearn with the same parameters defined above and report the test accuracy obtained using ReLU, sigmoid, linear and tanh activation functions. Comment on the differences in accuracy, if any.

3. (20 points) For the DATASET provided, use the following hyperparameter settings to train each neural network (using PyTorch) described below-

Parameter	Value
# of Hidden units	4
Weight Initialization	Random
Learning Rate	0.01

1. **Hidden Units:**
   For the hyperparameters mentioned in the table above except the number of hidden units, train a single hidden layer neural network changing the value of the number of hidden units to 5, 20, 50, 100 and 200. Run the optimization for 100 epochs each time.
   (a). (6) Plot the average training cross-entropy (sum of the cross-entropy terms over the training set divided by the total number of training examples) on the y-axis vs number of hidden units on the x-axis. In the same figure, plot the average validation cross-entropy.
   (b). (4) Examine and comment on the plots of training and validation cross-entropy. What is the effect of changing the number of hidden units?
2. **Learning Rate:**
   For the hyperparameters mentioned in the table above except the learning rate, train a single hidden layer neural network changing the value of the learning rate to 0.1, 0.01 and 0.001. Run the optimization for 100 epochs each time.
   (a). (6) Plot the average training cross-entropy on the y-axis vs the number of epochs on the x-axis for the mentioned learning rates. In the same figure, plot the average validation cross-entropy loss. Make a separate figure for each learning rate.
   (b). (4) Examine and comment on the plots of training and validation cross-entropy. How does adjusting the learning rate affect the convergence of cross-entropy of each dataset?

4. (20 points) Use the binary **CIFAR 10** subset for this part.

1. (3) Conduct Exploratory Data Analysis (EDA) on the CIFAR-10 dataset. Report the class distribution.
2. (10) Use the existing AlexNet Model from PyTorch (pretrained on ImageNet) as a feature extractor for the images in the CIFAR subset. You should use the fc8 layer as the feature, which gives a 1000 dimensional feature vector for each image.
3. (5) Train a Neural Network with 2 hidden layers of sizes 512 and 256, and use the fc8 layer as input to this Neural Network for the classification task.
4. (2) Report the test accuracy along with the confusion matrix and the ROC curve.