Instructor: Dr. Naresh Manwani	Homework Assignment 2 Deep Learning: Theory and Practices (CS7.601), Spring 2021 Submission Deadline: **11.55 PM, 19/03/2021**	Total Marks : 30 Submit at: courses.iiit.ac.in

Instructions

1. This is a programming assignment and all questions are compulsory to attempt. Total Marks for the assignment is x.
2. Your submissions are expected to be a .tar.gz file containing the folowing:
   - main.ipynb - A Jupyter notebook with Code and explaination. Please ensure that you include a **detailed report** of the results within the notebook using **markdown cells**.
   - Auxiliary python scripts (plotting, helper functions which need not appear in the main notebook.)
3. It is advised that you try out Google Colaboratory to ensure the code is reproducible.
4. **Use of Pytorch, Tensorflow or any other deep learning framework is not allowed.** If you are using any other languages, please contact the TAs before you proceed.

Gradient Descent Method [6 Points]

Problem 1. Consider learning a linear regression function (without bias term) as follows.

$$y = 3x_1 + 4x_2 + \epsilon \tag{1}$$

Generate 500 samples in uniformly random manner with $x_1, x_2 \in [-10, 10]$. Add a gaussian noise ϵ with mean $= 0$ and variance $= 0.01$ to each generated sample value y obtained from x_1, x_2.

- Consider learning a linear regression function (without bias term). Hence, there will be two learnable parameters only. Assuming squared error loss function, write the objective function J to be minimized. **[0.5 marks]**
- Plot the error curve with respect to the parameters for this problem. **[0.5 marks]**
- Consider minimizing J using gradient descent with constant step size η_{opt}. Calculate and explain the optimal learning rate η_{opt}. **[1 mark]**
- Try gradient descent with following step sizes $\eta = \frac{0.9\eta_{opt}}{2}, \frac{1.5\eta_{opt}}{2}, \eta, 1.5\eta_{opt}$. For definiteness, we consider convergence to be complete $J < 0.001$. For every case,
  - Plot the error vs epoch curve. **[2 marks (0.5 for each η)]**
  - Using contour plots show the convergence path. **[2 marks (0.5 for each η)]**

Gradient Descent and Variants - 1 [4 Points]

Problem 2. Consider the Rosenbrock function $f(x, y) = x^2 + 100(y - x^2)^2$, which is used to benchmark optimization algorithms and the following variant admits a global minimum at $(0, 0)^T$. Use random initialization of the parameters.

- Run gradient descent with constant step size to minimize $f(x, y)$. Show contour plot of the function. After every update, using arrow show the movement in the contour plots. Do it till convergence. [1 Point]
- Use gradient descent with Polyak's momentum method to minimize $f(x, y)$. Show contour plot of the function. After every update, using arrow show the movement in the contour plots. Do it till convergence. [1 Point]
- Minimize $f(x, y)$ using Nesterov accelerated gradient descent. Show contour plot of the function. After every update, using arrow show the movement in the contour plots. Do it till convergence. [1 Point]
- Minimize $f(x, y)$ using Adam optimizer. Show contour plot of the function. After every update, using arrow show the movement in the contour plots. Do it till convergence. [1 Point]

Gradient Descent and Variants - 2 [4 Points]

Problem 3. Consider the following function

$$f(x,y) = \frac{50}{9}(x^2+y^2)^3 - \frac{209}{18}(x^2+y^2)^2 + \frac{59}{9}(x^2+y^2).$$

This function has global minimum at $(0,0)^T$ and local minima at $x^2+y^2=1$. Consider minimizing $f(x,y)$ using the methods below. Use random initialization of the parameters.

- Use gradient descent with constant step size to minimize $f(x,y)$. Show contour plot of the function. After every update, using arrow show the movement in the contour plots. Do it till convergence. [1 Point]
- Use Polyak's momentum method to minimize $f(x,y)$. Show contour plot of the function. After every update, using arrow show the movement in the contour plots. Do it till convergence. [1 Point]
- Minimize $f(x,y)$ using Nesterov accelerated gradient descent. Show contour plot of the function. After every update, using arrow show the movement in the contour plots. Do it till convergence. [1 Point]
- Minimize $f(x,y)$ using Adam optimizer. Show contour plot of the function. After every update, using arrow show the movement in the contour plots. Do it till convergence. [1 Point]

Initialization Matters [5 Marks]

Problem 4. [5 marks]

	ω_1			ω_2			ω_3		
sample	x_1	x_2	x_3	x_1	x_2	x_3	x_1	x_2	x_3
1	0.28	1.31	-6.2	0.011	1.03	-0.21	1.36	2.17	0.14
2	0.07	0.58	-0.78	1.27	1.28	0.08	1.41	1.45	-0.38
3	1.54	2.01	-1.63	0.13	3.12	0.16	1.22	0.99	0.69
4	-0.44	1.18	-4.32	-0.21	1.23	-0.11	2.46	2.19	1.31
5	-0.81	0.21	5.73	-2.18	1.39	-0.19	0.68	0.79	0.87
6	1.52	3.16	2.77	0.34	1.96	-0.16	2.51	3.22	1.35
7	2.20	2.42	-0.19	-1.38	0.94	0.45	0.60	2.44	0.92
8	0.91	1.94	6.21	-0.12	0.82	0.17	0.64	0.13	0.97
9	0.65	1.93	4.38	-1.44	2.31	0.14	0.85	0.58	0.99
10	-0.26	0.82	-0.96	0.26	1.94	0.08	0.66	0.51	0.88

Figure 1: Data

Create a 3-1-1 sigmoidal network with bias to be trained to classify patterns from w_1 and w_2 in the table above (Figure-1). Use back-propagation with learning rate $\eta = 0.1$.

1. Initialize all weights randomly in the range $-1 \leq w \leq +1$. Plot a learning curve of the training error as a function of epoch. **[2 marks]**
2. Now repeat (1) but with weights initialized to be the same throughout each level. In particular, let all input-to-hidden weights be initialized with $w_{ji} = 0.5$ and all hidden-to-output weights with $w_{kj} = -0.5$. **[2 marks]**
3. Explain the source of the differences between your learning curves. **[1 mark]**

Backpropagation, Resiliant Propagation and Quickprop [6 Points]

Problem 5. Download the Wine Quality Dataset (white wine). This dataset can be viewed as a classification or regression task. Let us assume as a regression problem. There are a total of 4898 examples with 11 variables. Randomly split the data in training set (70% of total points) and testing set (30% of total points).
Train a 3-layer neural network (with three variations of number of hidden nodes 25, 50, 75) that can predict the value of Wine Quality, given the 11 inputs. Use following activation functions:
(a) tanh, (b) Relu. Use squared error loss function at the output. Train using RProp, Quickprop and batch Back-propagation (BP) for 1000 epochs. Report the following.

1. For every setting (number of hidden nodes, activation function), plot MSE versus number of epochs curves for (a) Rprop, (b) Quickprop and (c) Backpropagation. Comment on your observations.

2. Find the final training and test MSE values for each setting.

Preprocessing the data speeds up learning [5 Points]

Problem 6. Demonstrate that pre-processing data can lead to significant reduction in time of learning. Consider a single linear output unit for a two-category classification task, with teaching values ± 1 and squared error criterion.

1. Write a program to train three weights based on training samples.

2. Generate training set having 400 samples, 200 from each of the two categories $P(w_1) = P(w_2) = .5$ and $p(\mathbf{x}|w_i) \sim \mathcal{N}(\mu_i, \Sigma)$, where $\mathcal{N}$ is Gaussian distribution, $\mu_1 = (-3, 4)^T$ and $\mu_2 = (4, -3)^T$ and

$$\Sigma = \begin{bmatrix} 16 & 0 \\ 0 & 9 \end{bmatrix}$$

   Generate 200 test samples, 100 from each category.

3. Find the optimal learning rate empirically by trying a few values. Plot error versus epoch curve for both training and test sets. [2 Points]

4. Train to minimum error. Why is there no danger of overtraining in this case? [0.5 Point]

5. Why can we be sure that it is at least possible that this network can achieve the minimum (Bayes) error. [0.5 Point]

6. Now pre-process the data by subtracting off the mean and scaling with standard deviation in each dimension. Repeat the above, and find the optimal learning rate. [2 Points]