CS 615 - Deep Learning

Assignment 4 - Exploring Hyperparameters
Winter 2022

Introduction

In this assignment we will explore the effect of different hyperparameter choices and apply a multi–class classifier to a dataset.

Programming Language/Environment

As per the syllabus, we are working in Python 3.x and you must constrain yourself to using numpy, matplotlib, pillow and opencv–python add–on libraries.

Allowable Libraries/Functions

In addition, you **cannot** use any ML functions to do the training or evaluation for you. Using basic statistical and linear algebra functions like *mean*, *std*, *cov* etc.. is fine, but using ones like *train*, *confusion*, etc.. is not. Using any ML–related functions, may result in a **zero** for the programming component. In general, use the "spirit of the assignment" (where we're implementing things from scratch) as your guide, but if you want clarification on if can use a particular function, DM the professor on Discord.

Grading

Part 1 (Theory)	10pts
Part 2 (Visualizing an Objective Function)	10pts
Part 3 (Exploring Model Initialization Effects)	20pts
Part 4 (Exploring Learning Rate Effects)	20pts
Part 5 (Adaptive Learning Rate)	20pts
Part 6 (Multi–class classification)	20pts

Table 1: Grading Rubric

Datasets

MNIST Database The MNIST Database is a dataset of hand–written digits from 0 to 9. It contains 60,000 training samples, and 10,000 testing samples, each of which is a 28×28 image.
You have been provided two *CSV* files, one with the training data, and one with the testing data. This file is arranged so that each row pertains to an observation, and in each row, the first column is the *target class* $\in \{0, 9\}$. The remaining 784 columns are the *features* of that observation, in this case, the pixel values.

For more information about the dataset, you can visit: http://yann.lecun.com/exdb/mnist/

1 Theory

Whenever possible, please leave your answers as fractions so the question of rounding and loss of precision therein does not come up.

1. What would the *one–hot encoding* be for the following set of multi–class labels (5pts)?

$$Y = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 2 \\ 3 \\ 0 \end{bmatrix}$$

2. Given the objective function $J = \frac{1}{4}(x_1 w_1)^4 - \frac{4}{3}(x_1 w_1)^3 + \frac{3}{2}(x_1 w_1)^2$ (*I know you already did this in HW3, but it will be relevant for HW4 as well*):

   (a) What is the gradient $\frac{\partial J}{\partial w_1}$ (1pt)?

   (b) What are the locations of the extrema points for your objective function if $x_1 = 1$? Recall that to find these you set the derivative to zero and solve for, in this case, w_1. (3pts)

   (c) What does J evaluate to at each of your extrema points, again when $x_1 = 1$ (1pts)?

2 Visualizing an Objection Function

For the next few parts we'll use the objective function $J = \frac{1}{4}(x_1 w_1)^4 - \frac{4}{3}(x_1 w_1)^3 + \frac{3}{2}(x_1 w_1)^2$ from the theory section. First let's get a look at this objective function. Using $x_1 = 1$, plot w_1 vs J, varying w_1 from -2 to +5 in increments of 0.1. You will put this figure in your report.

3 Exploring Model Initialization Effects

Let's explore the effects of choosing different initializations for our parameter(s). In the theory part you derived the partial of $J = \frac{1}{4}(x_1 w_1)^4 - \frac{4}{3}(x_1 w_1)^3 + \frac{3}{2}(x_1 w_1)^2$ with respect to the parameter w_1. Now you will run gradient descent on this for four different initial values of w_1 to see the effect of weight initialization and local solutions.

Perform gradient descent as follows:

- Run through 100 epochs.
- Use a learning rate of $\eta = 0.1$.
- Evaluate J at each epoch so we can see how/if it converges.
- Assume our only data point is $x = 1$

Do this for initialization choices:

- $w_1 = -1$.
- $w_1 = 0.2$.
- $w_1 = 0.9$.
- $w_1 = 4$.

In your report provide the four plots of epoch vs. J, superimposing on your plots the final value of w_1 and J once 100 epochs has been reached. In addition, based on your visualization of the objective function in Section 2, describe why you think w_1 converged to its final place in each case.

4 Explore Learning Rate Effects

Next we're going to look at how your choice of learning rate can affect things. We'll use the same objective function as the previous sections, namely $J = \frac{1}{4}(x_1 w_1)^4 - \frac{4}{3}(x_1 w_1)^3 + \frac{3}{2}(x_1 w_1)^2$.

For each experiment initialize $w_1 = 0.2$ and use $x = 1$ as your only data point and once again run each experiment for 100 epochs.

The learning rates for the experiments are:

- $\eta = 0.001$
- $\eta = 0.01$
- $\eta = 1.0$
- $\eta = 5.0$

And once again, create plots of *epoch* vs J for each experiment and superimpose the final values of w_1 and J.

NOTE: Due to the potential of overflow, you likely will want to have the evaluation of your J function in a try/except block where you break out of the gradient decent loop if an exception happens.

5 Adaptive Learning Rate

Finally let's look at using an adaptive learning rate, á la the Adam algorithm.

For this part of your homework assignment we'll once again look to learn the w_1 that minimizes $J = \frac{1}{4}(x_1 w_1)^4 - \frac{4}{3}(x_1 w_1)^3 + \frac{3}{2}(x_1 w_1)^2$ given the data point $x = 1$. Run gradient descent *with ADAM* adaptive learning on this objective function for 100 epochs and produce a graph of epoch vs J. Ultimately, you are implementing ADAM from scratch here.

Your hyperparameter initializations are:

- $w_1 = 0.2$
- $\eta = 5$
- $\rho_1 = 0.9$
- $\rho_2 = 0.999$
- $\delta = 10^{-8}$

In your report provide a plot of epoch vs J.

6 Multi–Class Classification

Finally, in preparation for our next assignment, let's do multi–class classification. For this we'll use the architecture:

$$Input \rightarrow \text{Fully Connected} \rightarrow \text{Softmax} \rightarrow \text{Output w/ Cross–Entropy Objective Function}$$

Download the MNIST dataset from BBlearn. Read in the training and testing (validation) data. We will leave other design and hyperparameter decisions to you, although they should be documented in your report. You likely might want to take into account the things explored in this assignment (if so you'll likely need to change how some of your modules work, and that's fine!).

Train your system using the training data, keeping track of the value of your objective function with regards to the training and validation sets as you go.

In your final report provide:

- Your design/hyperparameter decisions.
- A graph of epoch vs. J for both training and validation.
- Your final training and validation *accuracy*.

NOTE: You may have some features with zero standard deviation. To avoid a divide–by–zero situation, I suggest setting those standard deviations one one

Submission

For your submission, upload to Blackboard a single zip file containing:

1. PDF Writeup
2. Source Code
3. readme.txt file

The readme.txt file should contain information on how to run your code to reproduce results for each part of the assignment.

The PDF document should contain the following:

1. Part 1:
   (a) Your solutions to the theory question
2. Part 2:
   (a) Your plot.
3. Part 3:
   (a) Your four plots of epoch vs. J with the terminal values of x and J superimposed on each.
   (b) A description of why you think x converged to its final place in each case, justified by the visualization of the objective function.
4. Part 4:
   (a) Your four plots of epoch vs. J with the terminal values of x and J superimposed on each.
5. Part 5:
   (a) Your plot of *epoch* vs J.
6. Part 6:
   (a) Any additional design decisions.
   (b) Hyperparameter choices.
   (c) Graph of J vs *epoch* for both training and validation data sets.
   (d) Final training and validation accuracies.