Pattern Recognition and Machine Learning

Lab - 9 Assignment
Neural Networks

Guidelines for submission

1. Perform all tasks in a single colab file.
2. Create a report regarding the steps followed while performing the given tasks. The report should not include excessive unscaled preprocessing plots.
3. Try to modularize the code for readability wherever possible
4. In-Lab Submission: Google Classroom

   Naming convention: **RollNo_InLabSubmission_9.ipynb**

   Submit the In-lab work after the lab itself.
5. Submit the colab[.ipynb], python[.py] and report[.pdf] files in the Google Classroom with proper naming conventions.

   Please follow the naming conventions:

   RollNo_LabAssignment_9.ipynb

   RollNo_LabAssignment_9.py

   RollNo_LabAssignment_9.pdf
6. Plagiarism will not be tolerated
7. For the final submission, you may take up to 4 extra days. No penalty will be there.

Guidelines for Report:

1. The report should be to the point. Justify the space you use!

2. Explanations for each task should be included in the report. You should know the ‘why’ behind whatever you do.

3. Do not paste code snippets in the report.

Question 01: [25 marks]: A Basic Neural Network - Digit Recognition

A. Download the MNIST dataset using torch-vision. Split into train, test and validation dataset. Apply Augmentions to images:
   a. Training dataset: RandomRotation(5 degrees), RandomCrop(size=28, padding = 2), ToTensor and Normalize.
   b. Testing dataset and validation dataset: ToTensor and Normalize
B. Plot a Few Images from each class. Create a data loader for the training dataset as well as the testing dataset.
C. Write a 3-Layer MLP using PyTorch all using Linear layers. Print the number of trainable parameters of the model.
D. Train the model for 5 epochs using Adam as the optimizer and CrossEntropyLoss as the Loss Function. Make sure to evaluate the model on the validation set after each epoch and save the best model as well as log the accuracy and loss of the model on training and validation data at the end of each epoch.
E. Visualize correct and Incorrect predictions along with Loss-Epoch and Accuracy-Epoch graphs for both training and validation.

Question 02: [60 Marks]: ANN from Scratch!!

In this exercise, you need to predict the life of Abalone - a kind of shellfish, based on a number of characteristics (sex, length, diameter, height, weights in different forms, etc.). Model it as a classification problem to predict the class (based on the number of rings). The dataset is available here.

A. For this classification, you need to use a multi-layer perceptron.
   a. Preprocess & visualize the data. Create train, val, and test splits but take into consideration the class distribution (Hint: Look up stratified splits). ~ [5]
B. b. Implement a multi-layer perceptron from scratch. This would include the following ~[40]
   a. Write activation functions.
   b. Forward propagate the input.
   c. Back propagate the error.
   d. Train the network using stochastic gradient descent.
   e. Predict the output for a given test sample and compute the accuracy.
C. Now experiment with different activation functions (at least 3 & to be written from scratch) and comment (in the report) on how the accuracy varies. Create plots to support your arguments. ~[5]
D. Experiment with different weight initialization: Random, Zero & Constant. Create plots

to support your arguments. ~[5]

E. Change the number of hidden nodes and comment upon the training and accuracy. Create plots to support your arguments.Add a provision to save and load weights in the MLP.~[5]

References: BackPropagation , Neural Networks

Question 03: [5 Marks]: Experiment with Architecture

Enough with all the coding. Let us use a few of the available tools/resources online to get a better understanding of neural networks. Add screenshots of your experiments and observations in the report and address the following questions **in your own words**.

A. Google PlayGround
    a. Does increasing the model size improve the fit, or how quickly it converges? Does this change how often it converges to a good model?
    b. Run the model as given four or five times. Before each trial, hit the **Reset the network** button to get a new random initialization. (The **Reset the network** button is the circular reset arrow just to the left of the Play button.) Let each trial run for at least 500 steps to ensure convergence. What shape does each model output converge to? What does this say about the role of initialization in non-convex optimization? Try making the model slightly more complex by adding a layer and a couple of extra nodes. Repeat the trials from Task 1. Does this add any additional stability to the results?
    c. Train the best model you can, using just X_1 and X_2. Feel free to add or remove layers and neurons, change learning settings like learning rate, regularization rate, and batch size. What is the best test loss you can get? How smooth is the model output surface?Even with Neural Nets, some amount of feature engineering is often needed to achieve best performance. Try adding in additional cross product features or other transformations like $\sin(X_1)$ and $\sin(X_2)$. Do you get a better model? Is the model output surface any smoother?

B. Neural Nets V2 Desmos Visualized
    a. Try tweaking the learning rate and report your observations regarding the stability of training by observing the decision boundary.