Deep Learning (316/616)
Assignment 1
Weightage: 3%

Submission Instructions

- Use only the given LaTeX template for answering questions
- For submitting the code and report, create a github repo with the name FirstName_LastName_RollNo and upload the code and report there itself. After you have uploaded all the files on your repository, share the github repo link over the shared form on classroom.
- Setting up the environment, running into technical problems, and figuring out their solutions is part of the learning process. Use Google, StackOverflow, and watch youtube videos. Unfortunately we will not be able to assist you.
- If you feel any portion of the problem is under-specified, you can safely assume that it has been done intentionally. You are free to make assumptions, but please specify the assumptions you make in the report. No further clarifications on the problems will be provided. However, feel free to contact the TAs in case you believe that there exists an error in the problem specification.

---

1. Which loss function, out of Cross Entropy and Mean Squared Error, works best with logistic regression because it guarantees a single best answer (no room for confusion)? Explain why this is important and maybe even show how it affects the model's training process. [3 Marks] [Theory]
2. For a binary classification task with a deep neural network (containing at least one hidden layer) equipped with linear activation functions, which of the following loss functions guarantees a convex optimization problem? Justify your answer with a formal proof or a clear argument. (a) CE (b) MSE (c) Both (A) and (B) (d) None [3 Marks] [Theory]
3. Dense Neural Network: Implement a feedforward neural network with dense layers only. Specify the number of hidden layers, neurons per layer, and activation functions. How will you preprocess the input images? Consider hyperparameter tuning strategies. [2 for implementation and 3 for explanation] [Code and Report]
4. Build a classifier for Street View House Numbers (SVHN) (Dataset) using pretrained model weights from PyTorch. Try multiple models like LeNet-5, AlexNet, VGG, or ResNet(18, 50, 101). Compare performance comment why a particular model is well suited for SVHN dataset. (You can use a subset of dataset (25%) in case you do not have enough compute.) [4 Marks] [Code and Report]
