Assignment 2

Q1. Train VGG-16 CNN from scratch using MNIST dataset (which contains images of handwritten digits categorized into classes of 0 to 9)

a) (Which Activation function is better?) Fix the number of epochs (to 50 epochs) and fixate any learning rate, optimizer, batch size, train this network, with layers having:

- only *RELU* activation,
- only leaky RELU activation,
- only *sigmoid* activation,
- only *tanh* activation.

You will need to produce the type of class using Softmax after the last layer, in all the above cases. Tabulate/Plot and discuss the convergence of the network upon these different activations.

b) (What learning rate is better?) Fix the optimizer, batch size, and activation function,the number of epochs (to 50 epochs), train this network first using different learning rates (of your choice). Tabulate/Plot and discuss the convergence of the network upon varying learning rate.

c) (Role of optimizer) Fix the number of epochs, learning rate, batch size, activation function, train this network first using the following optimizers:

- Vanilla Gradient descent
- Momentum based gradient descent (momentum value of your choice)
- Adam.
- RMS prop.

Tabulate/Plot and discuss the convergence of the network upon using these different optimizers.

d) (**Transfer Learning**) So far, you would have understood the best/optimal hyperparameters that yield the best performance in your set of experiments. Save the weights/configuration of this model (this is what we will call the pretrained model). In this experiment, rather than classifying the images into classes 0 to 9, we will slightly change the problem statement in the following way:

- The dataset with digits 0-9 has mainly two types of information: either the handwritten digit is made using a curved stroke or a straight stroke (or maybe both). Hence, divide the dataset into two classes viz., the set of digits with curved stroke(s) **C: {0,2,3,5,6,8,9}** and the set of digits with straight stroke(s), **S: {1,4,7}.**

- **Modify the** last layer of the network so that now it can classify the digit into class '**C**' or class '**S**'**. Initialize the network with the weights of the pre trained model**

(except the weights of the penultimate layer that must be trained). In other words**, retrain only the last layer of the network** keeping the weight of all other layers same as were in the pretrained model.

Use the optimal hyperparameters (of your choice). Tabulate/Plot and discuss the convergence of the network in this task of transfer learning.

Q2. Train a CNN network with ResNet-18 as a backbone from scratch with CIFAR-10 and note down the performance.

a. Initialize the ResNet-18 network with pre-trained weights from ImageNet and then try to use these weights to improve the training for the CIFAR-10 dataset. Try to come up with different ways of using these weights to improve the performance and play with the hyper-parameters to get the best performance. Document the results of your experiments.
b. Train the network from scratch with the Tiny-CIFAR-10 dataset. Try using as many data augmentation techniques as you can think of to try to improve the performance. Try dropout after different layers and with different dropout rates. Document the results of your experiments.
c. Visualize the activations of the CNN for a few test examples in each of the above cases. How are the activation in the first few layers different from the later layers?

For the Tiny-CIFAR-10 dataset, take 500 images per class from CIFAR-10 for training. Use the same 10,000 images for testing as per the CIFAR-10 dataset.

Suggested Frameworks: Tensorflow/Pytorch.

NOTE: Submissions will be done on moodle in the form of a pdf report and a zip containing the source code with input files considered and output files generated. Only one of the group members will upload the assignment. The deadline for this assignment is 12th March 2023.