# **Assignment 1**

## **Noise and Normalization layers in Convolution Neural Networks**

**Noise layers:**

While training deep neural networks, the distribution of each layer changes during training, as the parameters of the previous layers change. This makes it a complicated task.

We aim to get a high accuracy of the model for the given input. In case of CIFAR-10, we want the predicted label of the tested dataset to match the input label.

To Improve the accuracy of the model, we add a noise to the input layer. Adding noise to the input layer avoids overfitting of the model. It makes the outputs impervious to the small perturbations in the input.

Allowing some inaccuracy while training the deep neural network improves the accuracy and the training performance of the model. This improves the accuracy on the test data.

Relaxing consistency introduces stochastic noise into the training process. This implicitly mitigates over-fitting of the model and generalizes the model better to classify test data.

Noise also eliminates the memorization effect of a deep neural network and hence allows the model to capture the general observation of the training data that can be applied.

**Normalization layers:**

Normalization layers are added so that all the inputs are in a comparable range. Since the parameters of the previous layers are changing, it slows down the training by requiring lower learning rates and careful parameter initialization. This makes it very hard to train models. Hence we need to normalize layer inputs.

Batch Normalization allows us to use much higher learning rates and be less careful about initialization. It also acts as a regularizer, in some cases eliminating the need for Dropout.

To be able to take full advantage of Batch Normalization, we need to Increase the Learning rates and eliminate the dropouts.

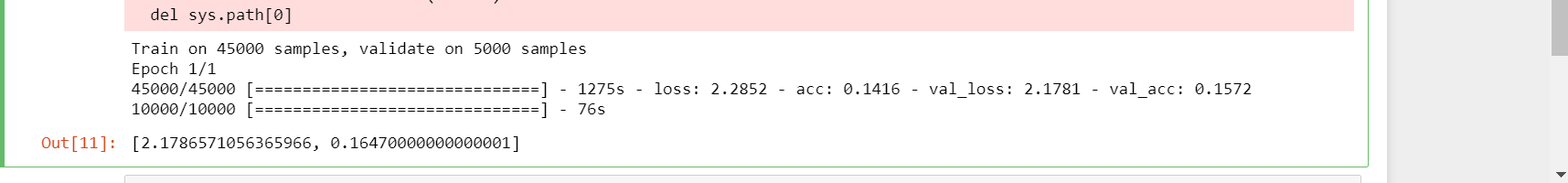
For the CIFAR-10 dataset, I have implemented the CNN for three types of noise layers with batch Normalization:

* White gaussian Noise
* White gaussian dropout
* Alpha dropout

The results are compared to the implementation without any noise or normalization.

The results are as shown below:

With Gaussian Noise:



Without Gaussian Noise:

