Hyunjin Lee 2020320023
Professor Seungryong Kim
COSE474-01
20 October 2021

Project 1: MLP Implementation

Development Environment: Google Colab

1) Code Explanations: (also commented in the python file)

I have implemented loss, train, prediction functions in neural_net.py for two-layer net classifying images on CIFAR-10 dataset. Implementing the loss function was straightforward. The fully connected layer was implemented with dot product between input and weight matrices and adding bias through broadcasting. The Relu layer was implemented by creating a mask with false values for values smaller than or equal to 0. Cross entropy loss was used to find the loss value for the model. The gradients for the weight and bias were calculated using backpropagation and local gradients.

For the train function, I first selected array of random indices using choice() function from numpy.random. I ran the loss function with the minibatch training data and labels and performed stochastic gradient descent on the weight matrices stored in the params dictionary. To predict the labels, I ran the loss function using the given input and selected the indices with the largest value using argmax() function from numpy.

2) Results

The two-layer net showed good performance on classifying CIFAR-10 datasets. By tuning the hyperparameters using random search, I was able to reach 46.6% accuracy on the validation set and 48% accuracy on the test set. The random search was first done by using a coarse search across the log space for the learning rate and regularization strength. After finding the general area of where the “true” hyperparameters were located, I then ran a finer search to find the best values for the hyperparameters.

3) Discussions

Although the two-layer net was good at classifying CIFAR-10 dataset, there were many ways to improve the performance. For example, the model showed some signs of overfitting as the training accuracy reached almost 70% while the validation accuracy was 46.6%. The variance problem can be solved by introducing dropout or a batch normalization layer between the first fully connected layer and the nonlinear activation function. However, fixing the variance problem only allows the test accuracy to get closer to the training accuracy. In order to improve the model’s performance as a whole, the model’s complexity needs to be increased. This can be done by introducing another full connected layer with nonlinearity into the model.