## horizontal line

Deep Learning Exercise 1

George Tsiatsios

# Overview

The CIFAR-10 Dataset consists of 60000 color images from 10 classes (airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck). Within the training dataset there are 50000 tiny images, with each image having the dimensions of 32 pixels by 32 pixels structure ( 32 \* 32 \* 3), which correspond to width , height, RGB. The pixel values are in range of 0 to 255 for each read,green and blue channels.The test dataset has identical components but only 1000 images. The main challenge is to learn a multi-class classifier that can classify a given image into a unique class.

# Data Preprocessing

Zero-Centre and Normalization are very simple pre-processing technique to regulate the image data. The zero-centre subtracts the mean across every individual feature in the data and normalization scales data points to the range 0 to 1 by dividing each value by the maximum observation which is 255. The data are loaded as integers , so we must cast it to floating point values in order to perform the division.

In the fully connected neural networks the input data should be a vector instead of several 32\*32\*3 images. The images are reshaped to 32\*32\*3 = 3074 vector. On the other hand flattening process could be used and reshaping could be avoided. The labels for this dataset are numerical values from 0 to 9 so it is important to our network to threat these elements as items ,rather than ordinal values. When we are making predictions about categorical data the best practice is to use ‘one-hot encoded’ vector. This means that we create a vector as long as the number of categories we have, and force the model to set exactly one of the positions in the vector to 1 and the rest to 0.

# Neural Network Architectures

## How many layers ? How many nodes in each layer?

In this section I am trying to archive a baseline network. Relu has has been chosen as SOTA activation function Categorical cross entropy has been chosen as loss function due to the multiclass problem. and the simplest optimization algorithm : Stochastic Gradient Descent. Adam or RMSprop could also been chosen as fast convergence algorithm but it’s common to use SGD as SOTA performance. All the networks have the same input and output layer. The input layer is a vector of 3074 entries. The output layer uses softmax as activation function.This allows us to treat those ten output values as probabilities, and the largest one is selected as the prediction for the one-hot vector. Beside that I use 32 batch size and 15 epoch for faster training in order to review the networks. I experiment with multiple architectures. Such as [1024,512,512] , [1024,600], [1024,512,128], [256,256], [512,512] ,[60,30],[60,30,20]. Network [1024,512,512] and [1024,600] archive the highest validation accuracy over 50%. We observe that By increasing the number of neurons and layers we are able to archive higher accuracy. In the experiment I proceed with the best model [1024,512,512], model 0.

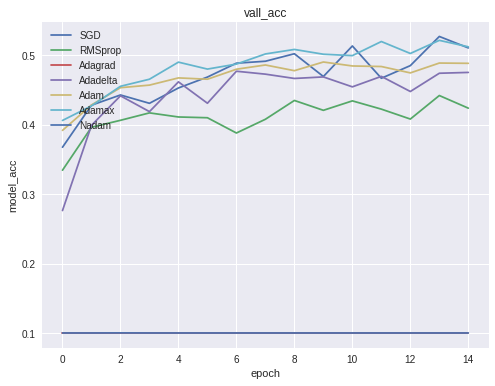
## 2. Choice of activation and loss functions

In order to choose the best neuron activation function in the hidden layer.Here we require softmax function in the output and cross entropy as loss function for the multiclass problem I should experiment with all the possible [relu , tanh, sigmoid] but as we observe relu is thes best activation function. All the hidden layers had the same function.

## 3.Regularisation (Dropout, Batch Normalization , L1, L2)

Overfit occurs when the data are too tightly coupled to the model are representing. To overcome this we use Dropout, Batch Normalization , L1 or L2. Dropout should be placed after each layer.As we will see in next section when we are trying to find the optimal number of epoch and batch size when the epoch are over 80 the model start to overfit. From the regularization the overall accuracy decreases but we ensure that the model is not overfitting. I apply all the Regularisation functions in the hidden layer. I experiment with Dropout in rates (0.2,0,5,0.6,0.7,0.8) and observate that with 0.2 the best validation is archived and the lowest error.

Furthermore L1 and L2 with [0.01,0.04,0.001,0.0001] regularisation and Batch Normalisation between Hidden layers and the results are that with the least values of the regularisation we archive the best model with the lowest loss. The rates are not the optimal because here I am trying to find the best model with 15 epochs with 100 or 1000 it may overfit so in the final models I will try to tune the Dropout and L1 , L2 rates to find the best model.

3. Choice of Training Optimization Algorithm

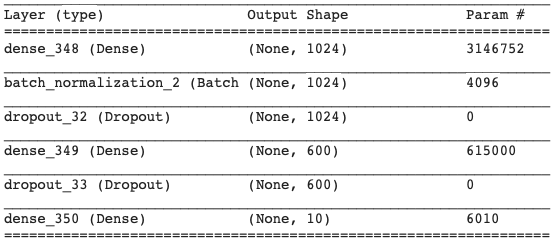
By using the best model from section 1 we see that SGD is close with the adaptive algorithms in 15 epochs , the reason of is that these are fast converging algorithms in 100 epochs may be not so good.

4.Hyper-Parameter configuration

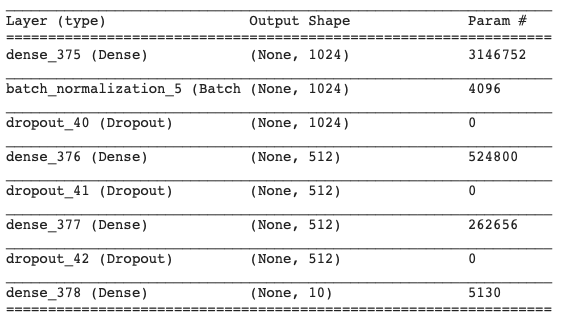
For hyper-parameters I have been experimenting with Learning rate [0.001, 0.01, 0.1, 0.2, 0.3] , which controls how much to update the weight at the end of each batch ,for the best optimizers and momentum[0.1,0.4,0.9] in SGD which ,controls how much to let the previous update influence the current weight update. With batch size [32,64,128] and epochs [15,50,100] as we see with 100 epochs we overfit so in the final model we will try to tune the regularisers properly. Generally I try to find the best learning rate and momentum, nesterov momentum( for SGD) and then the optimal epochs and batchsize. The best hyper parameters for SGD are [lr=0.001, momentum=0.9]. For Adam learning rate 0.001.

For Batch size 32 and epochs 100 we have the best validation accuracy. Ideally we need to add the learning rate when trying to find the optimal epochs and batchsize to see how much the batch size and learning rate differ across the epochs. By experiment with learning rates[0.001, 0.01, 0.1], batch size [32, 64, 128, 256] and epochs [50, 100]. Regarding the epoch as the learning rate increases the model overfits faster.

5. Final Models

For the final model considering all above experimentations the best model is: 1024,512,512,10 with dropout in between all layers , batch size and epochs 100 With Dropout as Regulariser with rate 0.5 the model does not overfit. Relu as neuron activation function , softmax as output activation function and lr=0.001, momentum=0.9,nesterov=True. This model give me:

**Test loss: 1.3955701761245727 Test acc: 0.5108**

Another model could be:

This model has the same hyperparameters as above but layers 1024,600,10. Finally after 100 epochs I archived:

**Test loss: 1.2978966299057006  
Test accuracy: 0.5479**

lr=0.001, momentum=0.9,nesterov=False

in the test set which is low. I hope with more than 100 epochs this model will get more than that.