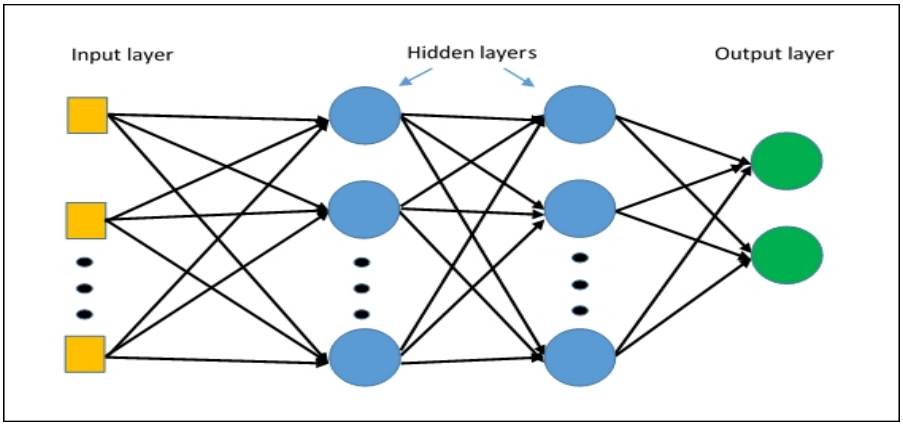
WHAT IS AN MLP?
A multilayer perceptron (MLP) is a class of feedforward artificial neural network. An MLP consists of, at least, three layers of nodes: an input layer, a hidden layer and an output layer. Except for the input nodes, each node is a neuron that uses a nonlinear activation function. MLP utilizes a supervised learning technique called backpropagation for training.



Step1: Split the code into train and test data (used Cifar_10 Data)
Step2: Reshaped the X_train to (50000*3072)
Reshaped the X_test into (10000*3072)
Step3: Converted the y test into categorical variables using keras.to.
categorical_
Step4: Adding layers to Neural Networks Using Sequential ()
Step5: Printing the accuracy and plotting

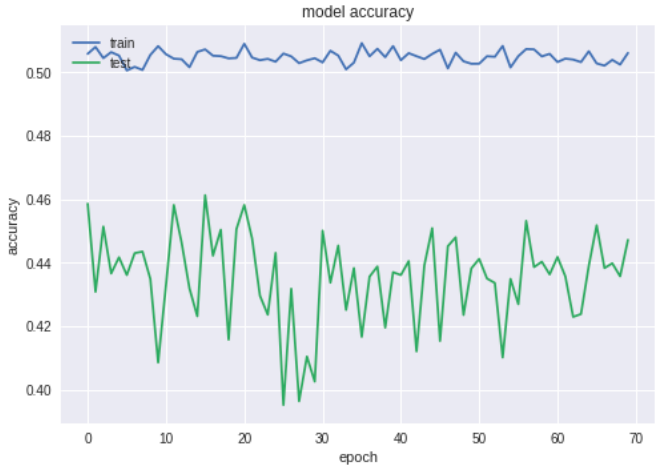
Base Model Configurations:
No of epochs to be 20
Batch Size 128
Number of classes = 10
No_of_hidden layers = 3
No_of_neurons = 512
Activation for hidden leayers = Relu
Activations for classification = SoftMax

Results : Test loss: 1.4791696659088134
Test accuracy: 0.4869
but this returned fluctuating results. Like when I ran for the first time it gave me an accuracy of 35% and then for the second time it gave me an accuracy of 45%. So, I decided to go for higher epochs like 70,150

Epochs 1):
First I ran the model for 70 epochs
Batch Size 128
Number of classes = 10
No_of_hidden layers = 3
No_of_neurons = 512
Activation for hidden leayers = Relu
Activations for classification = SoftMax

Test loss: 1.6852296886444091

```
Test accuracy: 0.4472
```



I got an accuracy of 44.72 % which was pretty. Much same each time I ran the 70 epochs.

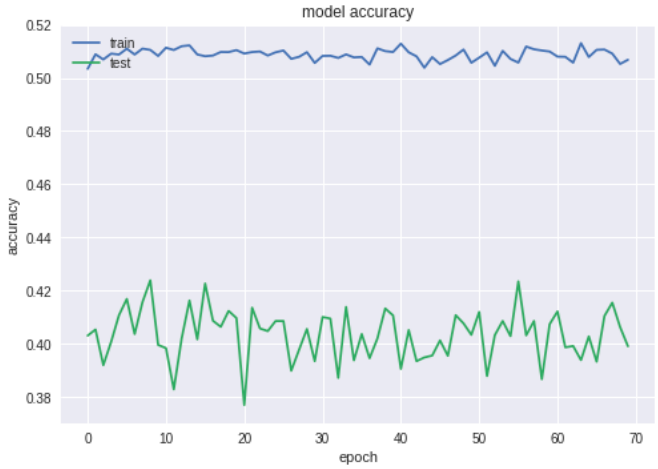
Epochs 2):
Next, I ran the model for 150 epochs
Batch Size 128
Number of classes = 10
No_of_hidden layers = 3
No_of_neurons = 512
Activation for hidden leayers = Relu
Activations for classification = SoftMax

```
Test loss: 1.9728069534301758
Test accuracy: 0.4179
```

The accuracy decreased so I decided to go with 70 epochs.
As the number of epochs increased the accuracy decreased.

Batch_Size 1):
Next, I ran the model for 70 epochs
Batch Size 200
Number of classes = 10
No_of_hidden layers = 3
No_of_neurons = 512
Activation for hidden leayers = Relu
Activations for classification = SoftMax

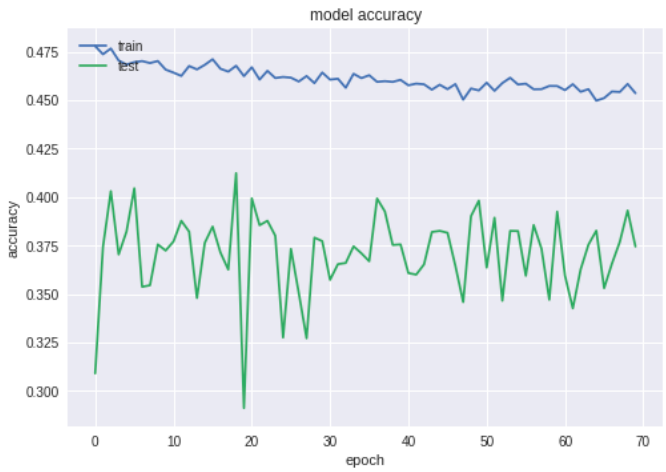
```
Test loss: 2.092425735092163
Test accuracy: 0.3991
```



The accuracy decreased and reached at 39.91%

Batch_Size 2):
Next, I ran the model for 70 epochs
Batch Size 70
Number of classes = 10
No_of_hidden layers = 3
No_of_neurons = 512
Activation for hidden leayers = Relu
Activations for classification = SoftMax

```
Test loss: 2.1960160053253173
Test accuracy: 0.3745
```

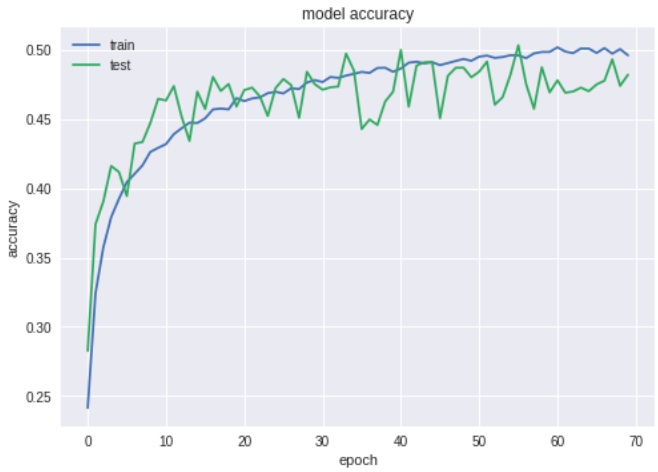


The accuracy further decreased and reached at 37.45%

Batch Size 3):
Next, I ran the model for 150 epochs
Batch Size 128
Number of classes = 10
No_of_hidden layers = 3
No_of_neurons = 512
Activation for hidden leayers = Relu
Activations for classification = SoftMax

```
Test loss: 2.093388455581665
Test accuracy: 0.4079
```

Increasing number of neurons 700
```
Test loss: 1.5205495515823364
Test accuracy: 0.4821
```



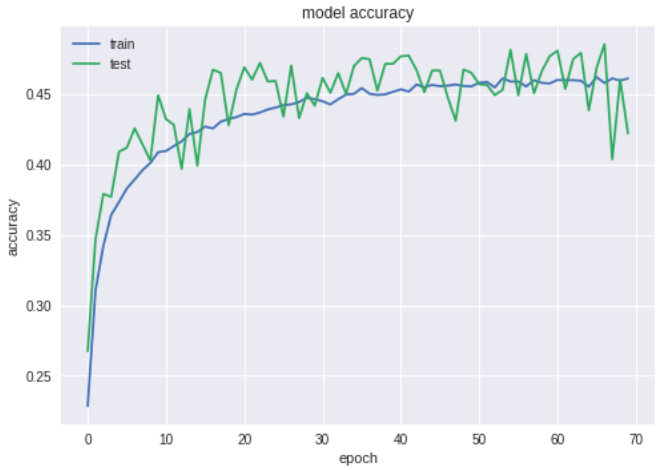
Accuracy increased to 48.21

By increasing number of number of 1000

```
Test loss: 1.531395346069336
Test accuracy: 0.4652
```

NOW INCREASING NUMBER OF HIDDEN LAYERS BY 4:
BATCH_SIZE=128
Epochs=70
NEURONS=700
hidden layers=2

```
Test loss: 1.610419565963745
Test accuracy: 0.4222
```

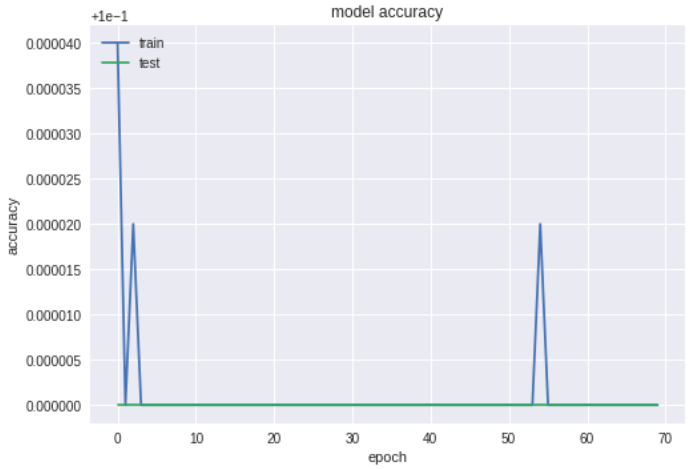


By decreasing the hidden layer to 2 accuracy decreased.

```
Test loss: 1.54212713470459
Test accuracy: 0.4566
```

Learning rate=0.1
BATCH_SIZE=128
Epochs=70
NEURONS=700
hidden layers=1
learning rate =0.1
Worst/Very low accuracy

```
Test loss: 14.506285720825195
Test accuracy: 0.1
```



BATCH_SIZE=128
Epochs=70
NEURONS=700
hidden layers=1
learning rate =0.001

Gave a good accuracy

```
Test loss: 1.5045540014266967
Test accuracy: 0.4804
```

BATCH_SIZE=128
Epochs=70
NEURONS=700
hidden layers=3
learning rate =0.0001

By further decreasing the learning rate the accuracy increases

```
Test loss: 1.344419214630127
Test accuracy: 0.5642
```

Trying the different hidden layers =3
Hidden layers =1
First layer = ‘tanh’
Second layer = ‘relu’
Third layer = ‘selu’
Final Layer= ‘soft max’

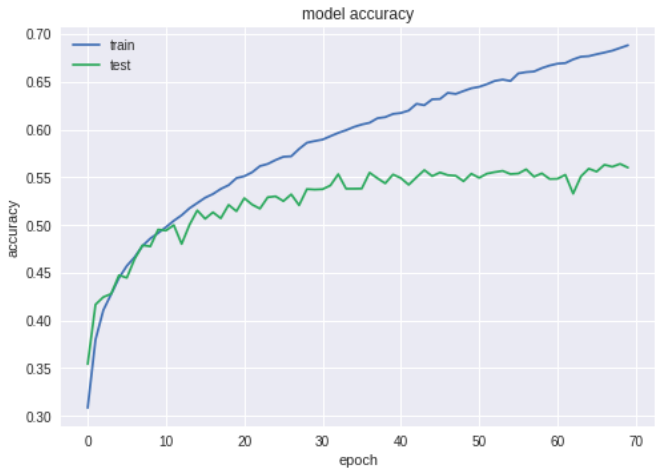
```
Test loss: 1.3187185523986817
Test accuracy: 0.5649
```

Now trying with 2 hidden layer
First layer= ‘selu’
Second layer= ‘relu’
Final Layer= ‘soft max’

```
Test loss: 1.2824515655517579
Test accuracy: 0.5701
```

Now trying
First Layer= ‘soft sign’
Second Layer= ‘relu’
Final layer= ‘ soft max’

```
Test loss: 1.2604014684677125
Test accuracy: 0.5601
```



Next trying
First Layer= ‘Hard Sigmoid’
Second Layer= ‘relu’
Output layer = ‘softmax’

```
Test loss: 1.2267157320022584
Test accuracy: 0.5685
```

Next I tried

First Layer= ‘tanh’
Second Layer= ‘relu’
Final Layer= ‘softmax’

```
Test loss: 1.3550836372375488
Test accuracy: 0.5554
```

I took the best model

Batch_size =128
Epoch = 70
'Hard Sigmoid'
'relu'
'softmax'
Lr = 0.0001
Droup out =0.2

Now tried drop out = 0.4

```
Test loss: 1.2492784006118773
Test accuracy: 0.5568
```

Now drop out = 0.6

```
Test loss: 1.3365159990310669
Test accuracy: 0.5296
```

Epochs: As the epochs increased the accuracy of the model decreased

Batch Size: As batch size increased / decreased there was significant change in the accuracy of the model

(Batch size of 128 was good for the model)

No of neurons: as I increased the neurons the accuracy of the model decreased. But after getting the final tweaking the number of neurons did not matter

By increasing the number of hidden layers (all with relu activation) the accuracy of the model decreased. But when started experimenting number of neurons with different activation function the accuracy increased depending upon the activation functions used.

But for the best model by tweaking the drop outs the accuracy only decreased

I choose the model with
EPOCH = 70
BATCH SIZE = 128
FIRST LAYER = ‘HARD SIGMOID’
SECOND LAYER = ‘RELU’
THIRD LAYER = ‘SOFT MAX’
LR = 0.0001
DROP OUT = 0.4

```
Test loss: 1.2492784006118773
Test accuracy: 0.5568
```

Not only gave me a good accuracy but gave has a good fit to the data

Most of my models were overfitting. Hence increase of neurons at the beginning only made it worse.
But as I tweaked the activation layers, reduced the learning rate made average number of neurons (not large not small) . Models started to become better