CSE-4/574 Programming Assignment-5

Divya Pandey
dpandey@buffalo.edu

Manish Addanki
maddanki@buffalo.edu

AkshayPandey
pandey5@buffalo.edu

Abstract

In this project we are building convolution neural networks on ASL dataset. The main tasks of this project is to build CNN with different regularizations such as L1 and L2 and dropout to achieve good accuracy.

1. Loading Dataset:

We have loaded the given ASL dataset in code and split it into training, test, and validation. Below is the output:

```
Found 4176 images belonging to 10 classes.
Found 1392 images belonging to 10 classes.
Found 1392 images belonging to 10 classes.
Number of samples in Train Folder:  4176
Number of samples in Valid Folder:  1392
Number of samples in Test Folder:  1392
```

2. Building Neural network and train with 8 layers

2.1 and 2.2:

We have built Convoluted Neural Network with following different type of layers -

i) 1st layer : filter size of 96 , kernel size =11*11, & padding value is of type 'same' followed by max pooling with pool size of 2x2, step size of 2x2 and padding type = 'valid'

ii) 2nd layer- filter size of 256 , step size =2x2 , kernel size =11*11, & padding value is of type 'same' followed by max pooling with pool size of 2x2, step size of 2x2 and padding type = 'valid'

iii) 3rd layer- filter size of 384 , step size =2x2 , kernel size =3x3, & padding value is of type 'same'

iv) 4th layer: filter size of 256 , step size =2x2 , kernel size =3x3, & padding value is of type valid followed by max pooling with pool size of 2x2, step size of 2x2 and padding type = 'valid'

v) 5th layer: Flattened followed by Dense layer of size 4096 and adding dropout value 0.4

vi) 6th layer : Flattened followed by Dense layer of size 4096 and adding dropout value 0.6

vii) 7th layer: Flattened followed by Dense layer of size 1000 and adding dropout value 0.5

viii) Final layer of softmax to give 10 output classes.

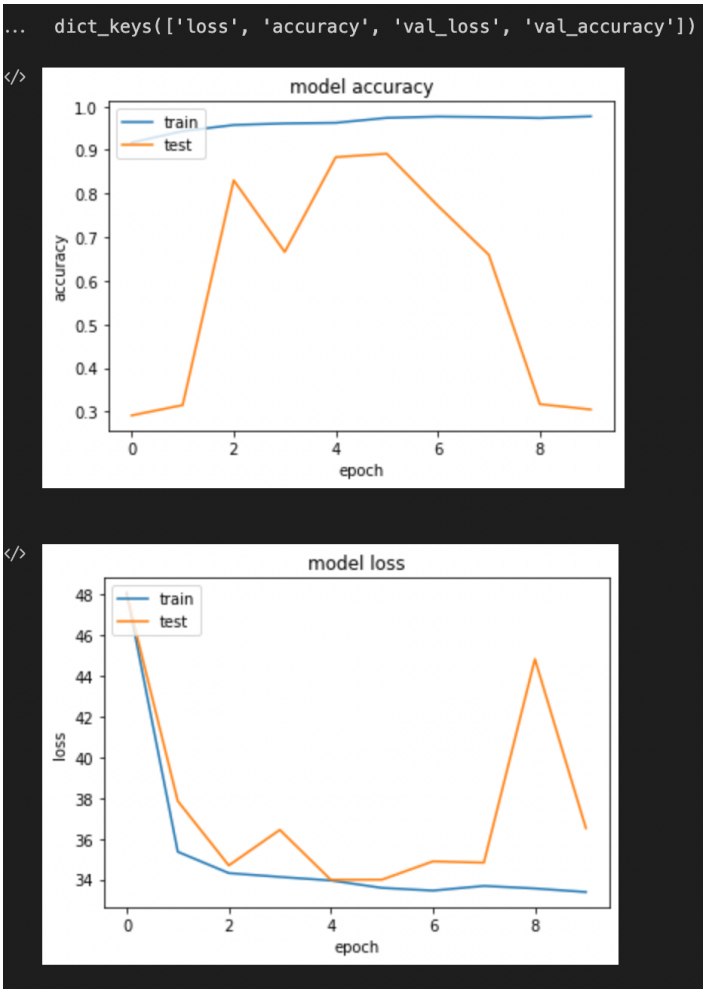
After each layer we added batch normalization and the activation function used across layers is ReLU

We have tried many combinations , incrementally added new layers and increased the number of nodes. Finally we were able to achieve an accuracy of greater than 80 percent i.e. 85.13 percent with the above layer configuration.

2.3 L1 regularization

We tried with regularization terms of 0.01,0.1,1 and 5 and got best results for 0.01 with an accuracy of 65.37 percent which is lesser than that without l1 regularization.

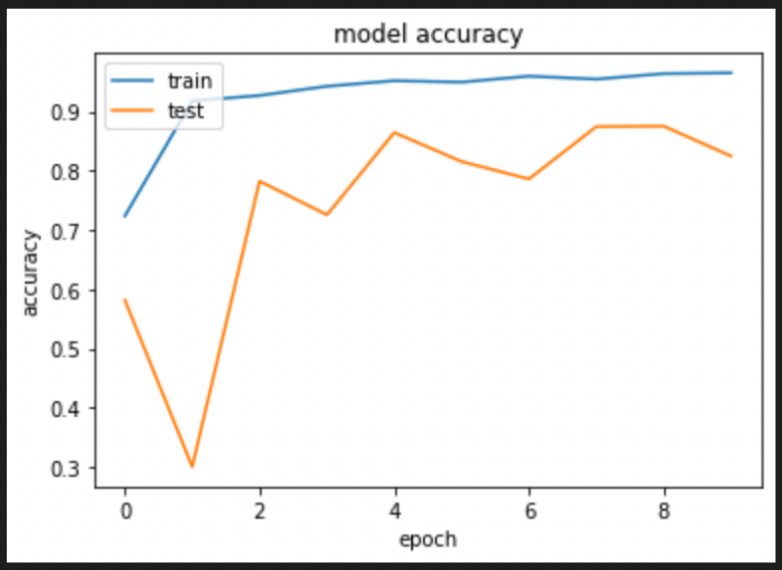
Graphs of Accuracy and Loss for lambda = 0.01 - L1 regularization:

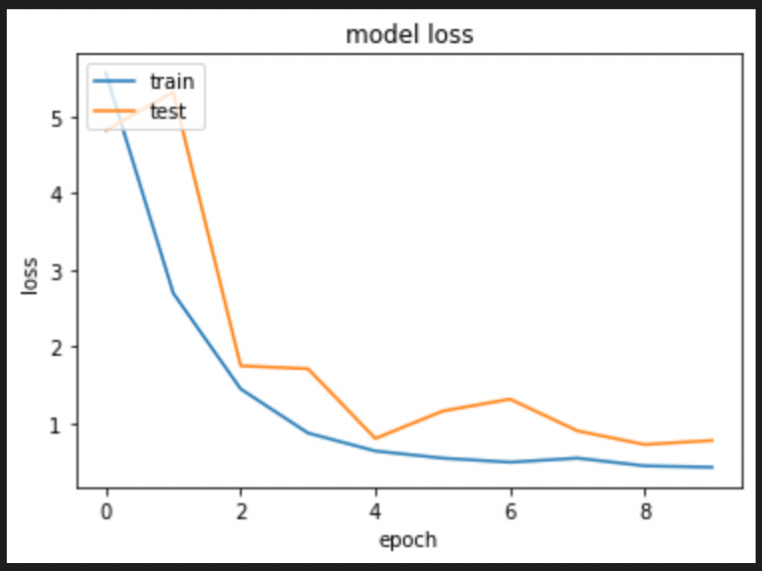


2.4 L2 regularization

We tried with regularization terms of 0.01,0.1,1 and 5 and got best results for lambda 0.01 which is lesser than that without any regularization but better than model with l1 regularization i.e. 79.96 percent.

```
..  dict_keys(['loss', 'accuracy', 'val_loss', 'val_accuracy'])
```

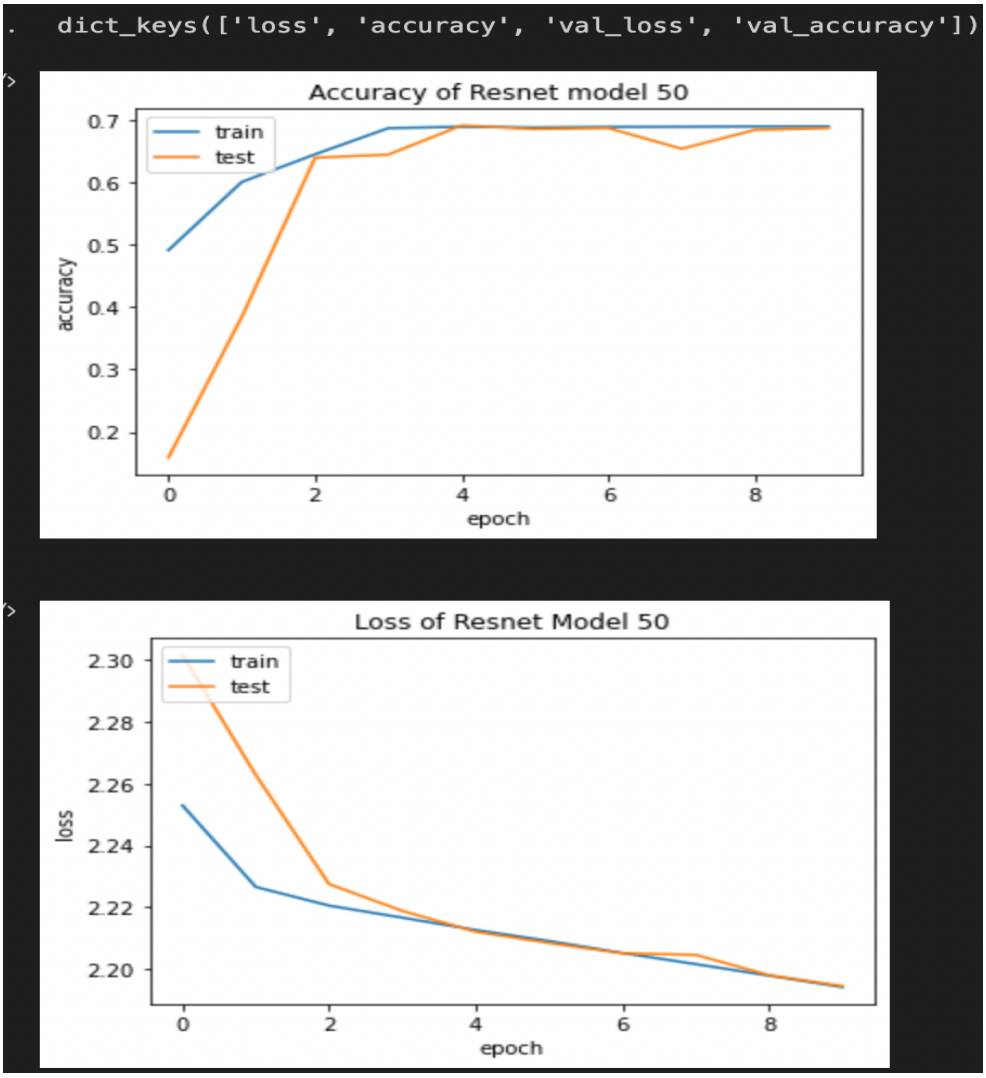




Part-3 : Fine Tuning ResNet

Steps:

- We loaded the Resnet 50 data using keras with input shape of 224* 224* 3.
- Built the model passing parameters weight value as “imagenet” & input tensor of size defined above in input shape.
- After we define the model we add one dense layer to tune the to fit our project class size of 10.
- Now, the final model is built which has 10 classes of the input images.
- We froze the top 70 layers of our model as trainable value ==”False”.
- We compiled our model of loss type categorical_cross entropy & metrics as accuracy.
- Finally, we trained the model.
- We achieved an accuracy of 68.10 percent.



References:

- https://www.freecodecamp.org/news/asl-using-alexnet-training-from-scratch-cfec9a8acf84/
- https://stackoverflow.com/questions/41668813/how-to-add-and-remove-new-layers-in-keras-after-loading-weights
- https://pyimagesearch.com/2020/04/27/fine-tuning-resnet-with-keras-tensorflow-and-deep-learning/
- https://machinelearningmastery.com/display-deep-learning-model-training-history-in-keras/
- Professor Kenneth Joseph lectures