Report for Assignment 3 (CIFAR100) Weekly Report

Tanish Singhal - 2017CH10258

Shubh Jaroria – 2017CS10380

5th Week:

For the 5th week, we tried some modifications in the VGG-16 network with data augmentation on the **test data** but the accuracy achieved wasn't able to beat our previous week's accuracy score. Hence we are submitting the same predictions and code as of previous week.
Previous Accuracy = 66.8%

4th Week:

For the 4th week, we tried some modifications in the VGG-16 network and even tried implementing a basic resnet but the accuracy achieved wasn't able to beat our previous week's accuracy score. Hence we are submitting the same predictions and code as of previous week.
Previous Accuracy = 66.8%

3rd Week:

For the 3rd week, we tried a new network – VGG16 [1], a deeper network than the previous ones we had tried. We also tried one 'ELU' model [2] which promised above 70% accuracy, but the training was taking a very large time and hence we switched from it.

References:

[1] https://github.com/geifmany/cifar-vgg

[2] https://medium.com/@birdortyedi_23820/deep-learning-lab-episode-5-cifar-100-a557e19219ba

Test Set Accuracy achieved – 66.8%

The details of the model are summarized as follows:

Layer Name	Output Shape	Number of parameters

Conv2D_1	(None, 32, 32, 64)	1792
Activation – 'relu'_1	(None, 32, 32, 64)	0
Batch Normalization_1	(None, 32, 32, 64)	256
Dropout_0.3_1	(None, 32, 32, 64)	0
Conv2D_2	(None, 32, 32, 64)	36928
Activation – 'relu'_2	(None, 32, 32, 64)	0
Batch Normalization_2	(None, 32, 32, 64)	256
Max Pooling2D_1	(None, 16, 16, 64)	0
Conv2D_3	(None, 16, 16, 128)	73856
Activation – 'relu'_3	(None, 16, 16, 128)	0
Batch Normalization_3	(None, 16, 16, 128)	512
Dropout_0.4_2	(None, 16, 16, 128)	0
Conv2D_4	(None, 16, 16, 128)	147584
Activation – 'relu'_4	(None, 16, 16, 128)	0
Batch Normalization_4	(None, 16, 16, 128)	512
Max Pooling2D_2	(None, 8, 8, 128)	0
Conv2D_5	(None, 8, 8, 256)	295168
Activation – 'relu'_5	(None, 8, 8, 256)	0
Batch Normalization_5	(None, 8, 8, 256)	1024
Dropout_0.4_3	(None, 8, 8, 256)	0
Conv2D_6	(None, 8, 8, 256)	590080
Activation – 'relu'_6	(None, 8, 8, 256)	0
Batch Normalization_6	(None, 8, 8, 256)	1024
Dropout_0.4_4	(None, 8, 8, 256)	0
Conv2D_7	(None, 8, 8, 256)	590080
Activation – 'relu'_7	(None, 8, 8, 256)	0
Batch Normalization_7	(None, 8, 8, 256)	1024
Max Pooling2D_3	(None, 4, 4, 256)	0
Conv2D_8	(None, 4, 4, 512)	1180160
Activation – 'relu'_8	(None, 4, 4, 512)	0
Batch Normalization_8	(None, 4, 4, 512)	2048
Dropout_0.4_5	(None, 4, 4, 512)	0
Conv2D_9	(None, 4, 4, 512)	2359808
Activation – 'relu'_9	(None, 4, 4, 512)	0
Batch Normalization_9	(None, 4, 4, 512)	2048
Dropout_0.4_6	(None, 4, 4, 512)	0
Conv2D_10	(None, 4, 4, 512)	2359808
Activation – 'relu'_10	(None, 4, 4, 512)	0
Batch Normalization_10	(None, 4, 4, 512)	2048
Max Pooling2D_4	(None, 2, 2, 512)	0
Conv2D_11	(None, 2, 2, 512)	2359808
Activation – 'relu'_11	(None, 2, 2, 512)	0
Batch Normalization_11	(None, 2, 2, 512)	2048
Dropout_0.4_7	(None, 2, 2, 512)	0
Conv2D_12	(None, 2, 2, 512)	2359808
Activation – 'relu'_12	(None, 2, 2, 512)	0

Batch Normalization_12	(None, 2, 2, 512)	2048
Dropout_0.4_8	(None, 2, 2, 512)	0
Conv2D_13	(None, 2, 2, 512)	2359808
Activation – ‘relu’_13	(None, 2, 2, 512)	0
Batch Normalization_13	(None, 2, 2, 512)	2048
Max Pooling2D_5	(None, 1, 1, 512)	0
Dropout_0.5_9	(None, 1, 1, 512)	0
Flatten_1	(None, 512)	0
Dense_1	(None, 512)	262656
Activation – ‘relu’_14	(None, 512)	0
Batch Normalization_13	(None, 512)	2048
Dropout_0.5_10	(None, 512)	0
Dense_2	(None, 100)	51300
Activation – ‘relu’_15	(None, 100)	0

===

Total params: 15,047,588

Trainable params: 15,038,116

Non-trainable params: 9,472

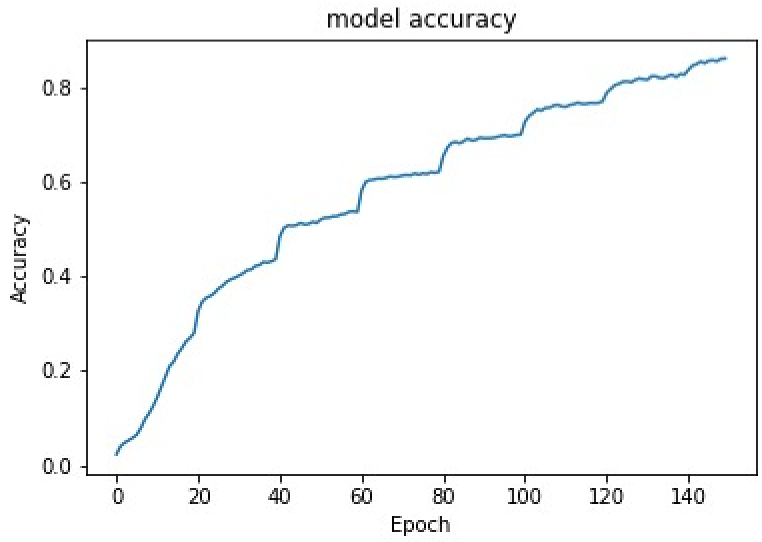
____

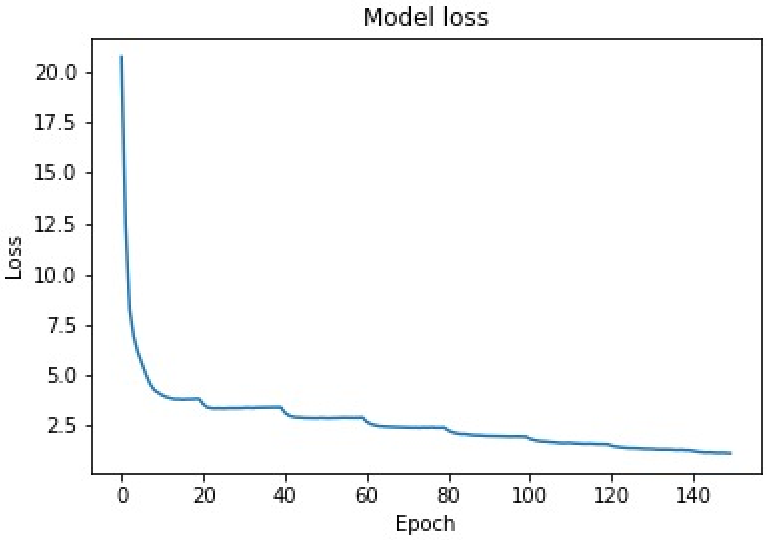
All the convolution layers have:

1. padding = ‘same’
2. kernel_size = 3X3
3. L2 regularization on kernel with weight decay = 0.0005

Data Augmentation was used to increase the train data without new data and to avoid over-fitting. In data augmentation, horizontal flip was done along rotation, and width and height shifting.

We obtained the following plots for training accuracy and loss:





Model Characteristics:

1. No. of epochs = 100
2. Learning rate = 0.1
3. Learning Rate Decay = 1e-6
4. Lr_drop = 20
5. Batch_Size = 128
6. Loss = Categorical Cross Entropy
7. Steps per epoch = 390
8. Momentum for sgd optimizer = 0.9

2nd Week:

For the 2nd week, we tried a new concept, merging random forest with the output of our previous CNN's flattened dense layer. We were able to achieve training accuracy of **99.9%** but the validation accuracy was dropped down to around 23% which indicates **overfitting** which we tried reducing by reducing the number of epochs and varying the parameters of the random forest. We were still not able to achieve a better accuracy and hence we are submitting the same predictions as last time for the competition part, although we are attaching our Random forest code also just for reference.

1st Week:

For the first week, rather than jumping directly into SOTA architectures, we implemented our model which was quite similar to the one submitted in CIFAR10 to understand and grasp the role of different layers in our model. We implemented a simple non-SOTA architecture due to the above reasons and to get a benchmark performance for our further models.

The model implemented: **ACCURACY ACHIEVED = 56.63%**

Model Summary:

Layer Name	Output Shape	Number of parameters
Conv2D_1	(None, 32, 32, 32)	896
Activation – 'elu'_1	(None, 32, 32, 32)	0
Batch Normalization_1	(None, 32, 32, 32)	128
Conv2D_2	(None, 32, 32, 32)	9248
Activation – 'elu'_2	(None, 32, 32, 32)	0
Batch Normalization_2	(None, 32, 32, 32)	128
Max Pooling2D_1	(None, 16, 16, 32)	0
Dropout_0.2_1	(None, 16, 16, 32)	0
Conv2D_3	(None, 16, 16, 64)	18496
Activation – 'elu'_3	(None, 16, 16, 64)	0
Batch Normalization_3	(None, 16, 16, 64)	256
Conv2D_4	(None, 16, 16, 64)	36928
Activation – 'elu'_4	(None, 16, 16, 64)	0
Batch Normalization_4	(None, 16, 16, 64)	256
Max Pooling2D_2	(None, 8, 8, 64)	0
Dropout_0.3_2	(None, 8, 8, 64)	0
Conv2D_5	(None, 8, 8, 128)	73856

Activation – ‘elu’_5	(None, 8, 8, 128)	0
Batch Normalization_5	(None, 8, 8, 128)	512
Conv2D_6	(None, 8, 8, 128)	147584
Activation – ‘elu’_6	(None, 8, 8, 128)	0
Batch Normalization_6	(None, 8, 8, 128)	512
Max Pooling2D_3	(None, 4, 4, 128)	0
Dropout_0.4_3	(None, 4, 4, 128)	0
Flatten_1	(None, 2048)	0
Dense_1	(None, 100)	204900

==

Total params: 493,700

Trainable params: 492,804

Non-trainable params: 896

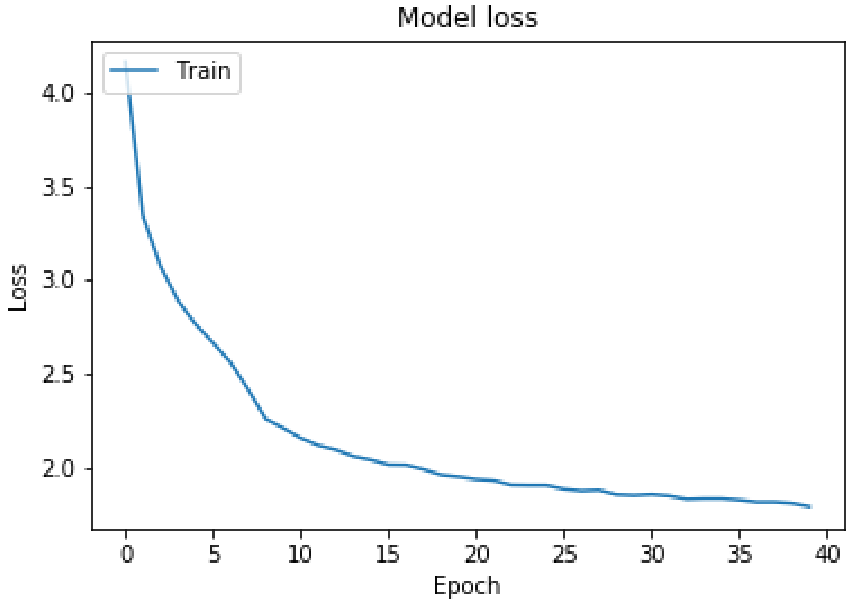
____

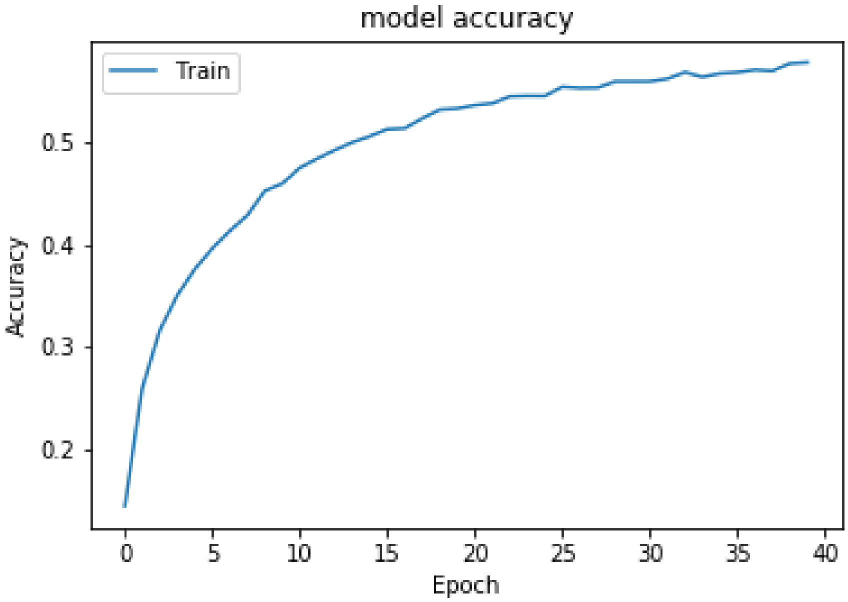
All the convolution layers have:

4. padding = ‘same’
5. kernel_size = 3X3
6. L2 regularization on kernel with weight decay = 1e-4

Data Augmentation was used to increase the train data without new data and to avoid over-fitting. In data augmentation, horizontal flip was done along with width and height scaling.

We obtained the following plots for training accuracy and loss:





Model Characteristics:

1. No. of epochs = 40
2. Learning rate = 0.001
3. Batch_Size = 64
4. Loss = Categorical Cross Entropy
5. Steps per epoch = 1000