Report for Programming Assignment-2

March 31, 2019

Checklist :

☑ checked.

☐ unchecked.

☒ not done.

The task you need to ensure before submission.

- ☑ We have read all the instruction carefully and followed them to our best ability.
- ☑ We have written the name, roll no in report.
- ☑ Run sanity_check.sh.
- ☑ We will be submitting only single submission on behalf of our team.
- ☑ We have not included unnecessary text, pages, logos in the assignment.
- ☑ We have not used any high level APIs(Keras, Estimators for e.g.).
- ☑ We have not copied anything for this assignment.

Team Mate 1 : Anshul Kumar
Roll No : ME15B082
Team Mate 2 : Ashutosh Raj
Roll No : ME15B086

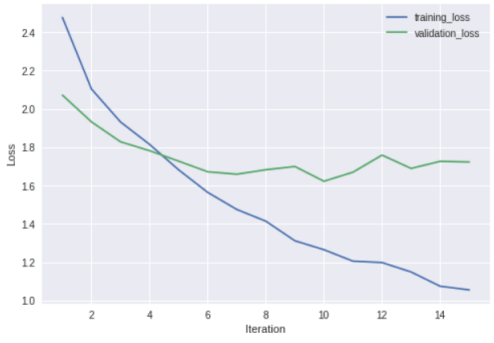
1 Configuration and training details of your best performing model

The default configuration gave the best results. The configuration is as follows:

Layer	Input Size	Output Size	Kernel Size	Padding	Stride
Conv1	64*64*3	62*62*32	32*3*5*5	1	1
Conv2	64*64*32	60*60*32	32*32*5*5	1	1
Pool1	60*60*32	30*30*32	-	1	2
Conv3	30*30*32	30*30*64	64*32*3*3	1	1
Conv4	30*30*64	30*30*64	64*64*3*3	1	1
Pool2	30*30*64	15*15*64	-	1	2
Conv5	15*15*64	15*15*128	128*64*3*3	1	1
Conv6	15*15*128	13*13*128	128*128*3*3	0	1
Pool3	13*13*128	7*7*128	-	1	2
FC1	6272	256	-	-	-
FC2	256	20	-	-	-

Training Details:
Learning Rate = 0.01
Batch Size = 64
Init = 2 (He)
Epochs = 20
Early Stopping = 1
Data Augmentation = 1 (Horizontal and Vertical Flipping)
Dropout keep probability = 0.5 (during training), 1 (during validation and testing)
Batch Normalization = on after every activation except in last layer (before softmax in last layer)

2 Plot of best training model

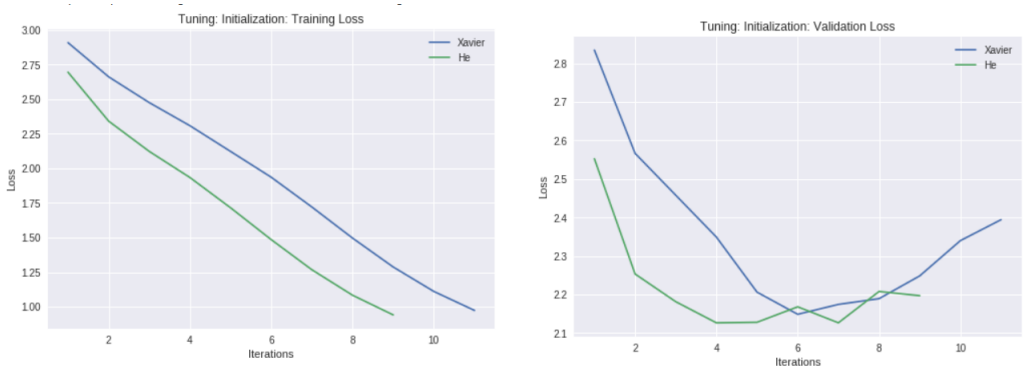


3 Performance on test data by best model

Our model gave a score of 0.5567 on Kaggle Public Leaderboard. It fairly matches with Validation score of 0.5531

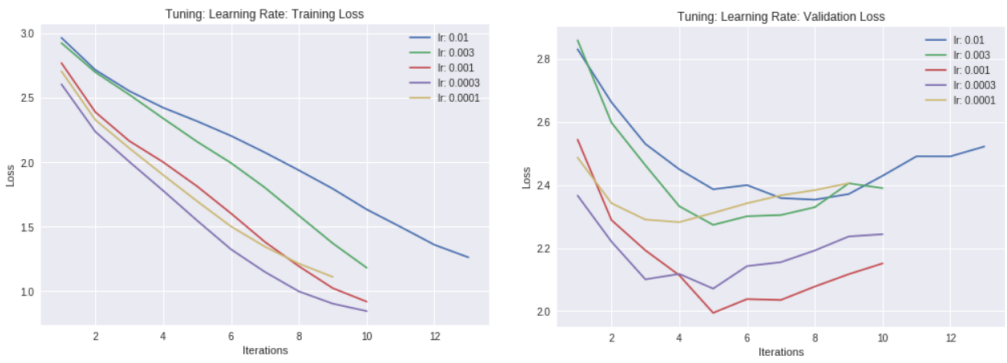
4 Hyperparameter Tuning

4.1 Initialization



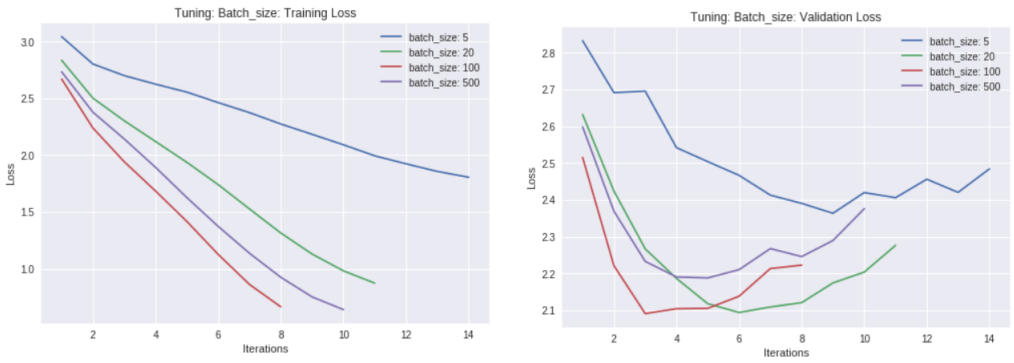
The model was trained with two initializations: 'Xavier' and 'He'. As visible from the training loss plot , the model learns faster in the case of 'He' initialisation 'He' initialization. Also, the validation loss is lower in the case of 'He' initialisation. This performance difference is because 'ReLu' activation is better suited with 'He' than 'Xavier' initialization.

4.2 Learning Rate



We observe that the learning rate of 0.001 gives best performance on the validation set. For learning rates higher than this, the model learns quite slowly with poor validation accuracy(more loss). And for the learning rates less than this, although the model learns faster (training error reduces faster), the model performs very poorly on the validation set. Also, we must note that the curves that some curves terminate at an early epoch because we have employed early stopping.

4.3 Batch size



The best batch size for the model can be seen to be 100. Here too, like the case of learning rate, the lower batch size results in slower training and high validation loss while higher batch size (more than 100) results in a comparable learning speed but very high validation loss. The batch size of 100 gives the provides fastest learning and least validation loss. However, we have tried using batch sizes in the multiples of 2 also, and have found that batch size of 64 outperforms all the tested batch sizes. Hence, in the best model, we have chosen batch size of 64.

5 Dimensions of input and output at each layers

Our model has the following architecture :

$$Image \rightarrow Conv1(F5{-}32) \rightarrow Conv2(F5{-}32) \rightarrow Pool1 \rightarrow Conv3(F3{-}64) \rightarrow Conv4(F3{-}64) \rightarrow Pool2$$

$$\rightarrow Conv5(F3{-}64) \rightarrow Conv6(F3{-}128) \rightarrow Pool3 \rightarrow FC1(256) \rightarrow SOFTMAX(20)$$

$Image\ size : 64 * 64 * 3 \quad (input\ to\ Conv1)$
$O/P\ Conv1 : 62 * 62 * 32 \quad (input\ to\ Conv2)$
$O/P\ Conv2 : 60 * 60 * 32 \quad (input\ to\ Pool1)$
$O/P\ Pool1 : 30 * 30 * 32 \quad (input\ to\ Conv3)$
$O/P\ Conv3 : 30 * 30 * 64 \quad (input\ to\ Conv4)$
$O/P\ Conv4 : 30 * 30 * 64 \quad (input\ to\ Pool2)$
$O/P\ Pool2 : 15 * 15 * 64 \quad (input\ to\ Conv5)$
$O/P\ Conv5 : 15 * 15 * 64 \quad (input\ to\ Conv6)$
$O/P\ Conv6 : 13 * 13 * 128 \quad (input\ to\ Pool3)$
$O/P\ Pool3 : 7 * 7 * 128 \quad (input\ to\ FC1)$
$I/P\ FC1 : 6272 \quad (fattening\ 7 * 7 * 128)$
$O/P\ FC1 : 256 \quad (input\ to\ softmax)$
$O/P\ Softmax : 20 * 1\ vector\ of\ class\ probability\ values$

6 Number of parameters in the network

Number of Parameters in Conv1 = 5 x 5 x 3 x 32 + 32 = 2432
Number of Parameters in Conv2 = 5 x 5 x 32 x 32 + 32 = 25632
Number of Parameters in Conv3 = 3 x 3 x 32 x 64 + 64 = 18496
Number of Parameters in Conv4 = 3 x 3 x 64 x 64 + 64 = 36928
Number of Parameters in Conv5 = 3 x 3 x 64 x 64 + 64 = 36928
Number of Parameters in Conv6 = 3 x 3 x 64 x 128 + 128 = 73856
Number of Parameters between Pool3 and FC1 = 6272 x 256 + 256 = 1605888
Number of Parameters between FC1 and Softmax layer = 256 x 20 + 20 = 5140
Thus, total parameters in Conv layers = 194272
Total parameters in FC layers = 1611028
Total parameters in all = 1805300 (approx. 1.8 mn)

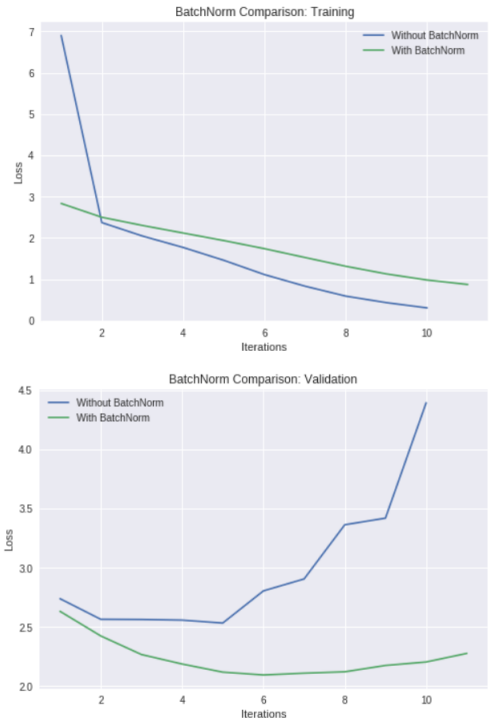
7 Number of neurons in the network

The number of neurons in the model is equal to the sum of the sizes of all the layers starting from the Conv1 uptil the softmax layer. Thus,
Neurons in Conv1 = 62 x 62 x 32 = 123008
Neurons in Conv2 = 60 x 60 x 32 = 115200
Neurons in Pool1 = 30 x 30 x 32 = 28800
Neurons in Conv3 = 30 x 30 x 64 = 57600
Neurons in Conv4 = 30 x 30 x 64 = 57600
Neurons in Pool2 = 15 x 15 x 64 = 14400
Neurons in Conv5 = 15 x 15 x 64 = 14400
Neurons in Conv6 = 13 x 13 x 128 = 21632
Neurons in Pool3 = 7 x 7 x 128 = 6272
Neurons in FC1 = 256
Neurons in Softmax = 20
Total neurons in Convolutional layers = 438912 Total Neurons in FC layers = 276 Total neurons = 439188

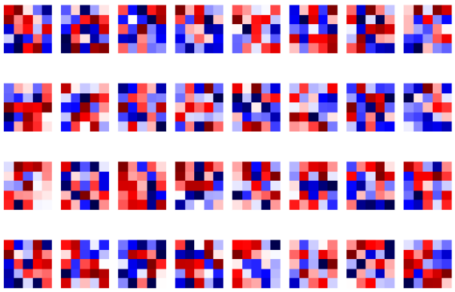
8 Effect of using Batch Normalization

It is a technique by which the inputs to a layer is standardized. It tries to keep the distribution of the inputs to the hidden layer same as that of the standardized inputs in the input layer. It also acts as a regularizer since it adds some noise to the inputs of the hidden layers.
We observe that for the given data, the network performs only slightly better in training without using batch normalization for the case of training. However, we can observe a lot of difference in the performances of the model under the two cases in the validation loss case. Thus, we can conclude that here batch

normalization has more pronounced effect as a regularizer than its effect on improving model's parameter learning speed.

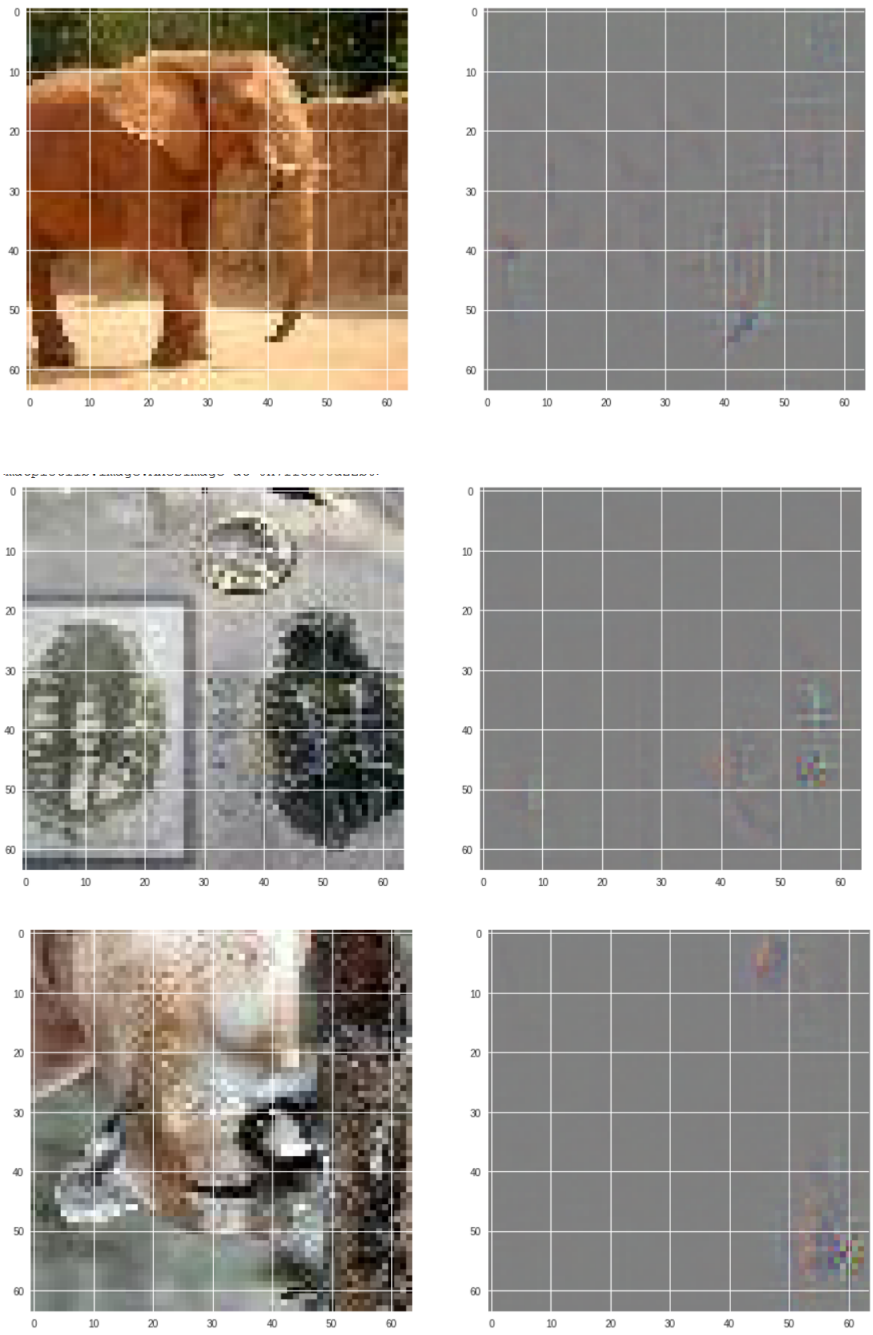


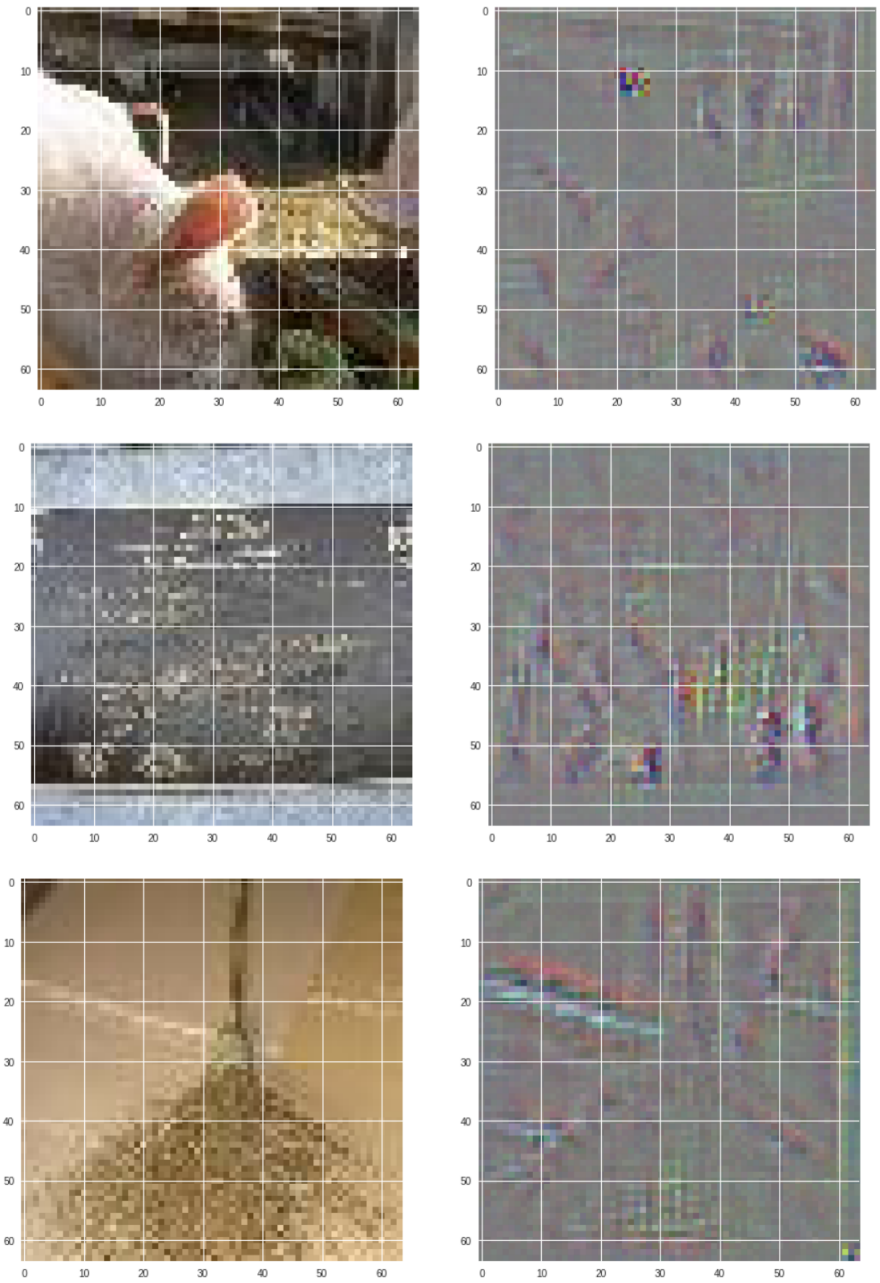
9 Plot of 32 layer-1 (Conv1) Filters

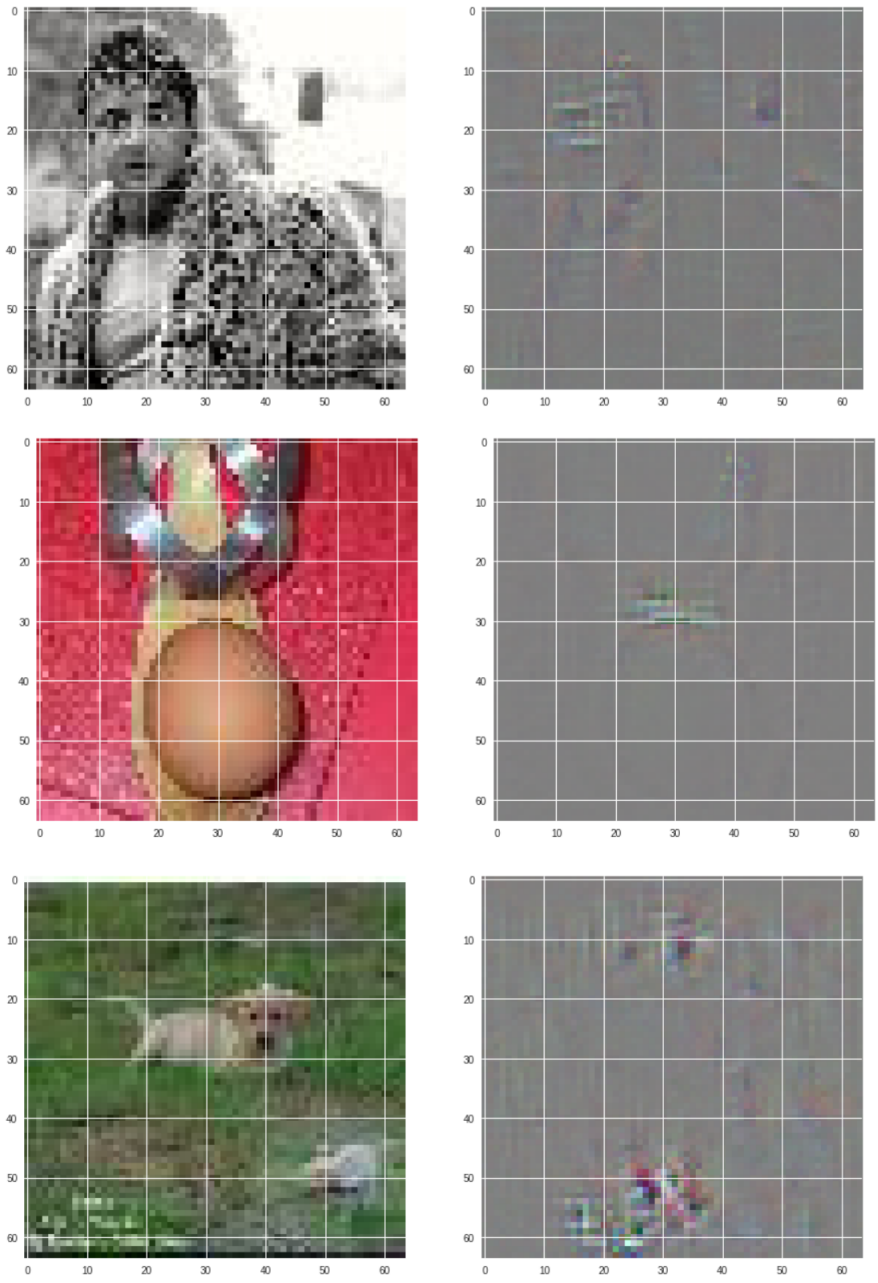


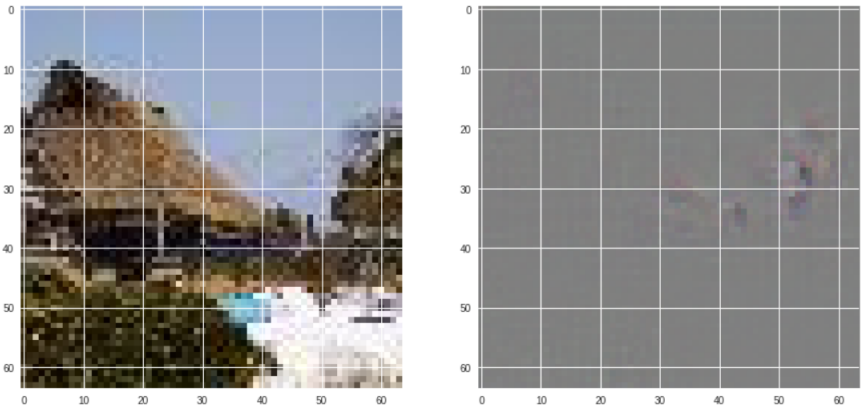
The plot of the 32 layer-1 filters we observe that the filters detect various form of information from the image along different orientation. It detects pattern along the highlighted part (red) from the images and send them forward through the convolutional layers. Various neurons are zeroed down as seen by blue region.

10 Guided Backpropagation









The neurons essentially act as feature detectors, and using guided backpropagation, we try to visualize which features (if any) are detected by some 10 random neurons in the Conv6 layer. We have plotted 10 images and the corresponding guided backpropagated images for the 10 random neurons in Conv6. We can see that in the first image of an elephant, we are interestingly able to see that some of these neurons get excited in the area corresponding to the trunk of the elephant. Similarly, in other images too, we find such features being detected. In some cases, these features are more prominent and intuitive (i.e. we might intuitively want the network to detect a feature like the trunk of the elephant for a labelling task(say)) like in the case of first image, while in others they are not. This helps us relate the working of the network to our intuition and may help us improve model's performance.

11 Additional redularization to improve model performance : Dropout

We have used Dropout of 0.5 on FC layer 1 as an additional regualrization method in order to improve the model performance. This is infact in the list of parameters for the best performing model. Dropout helps prevent overfit.