Convolutional Neural Networks for Images

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 : Anant Shah
Roll No : EE16B105
Team Mate 2 : Siddharth Bhatia
Roll No : EE16B120

1 Best Performing Model

The parameters of our best performing model on the test set are :

1. learning rate - 0.0007
2. 6 convolution layers with the described filter sizes - [5,5,3,64], [5,5,64,64], [3,3,64,64], [3,3,64,64], [3,3,64,64], [3,3,64,128], stride 1, with padding as SAME for the first 5 layers and VALID for the last layer
3. Max-Pooling at the 2^{nd}, 4^{th} and 6^{th} convolutional layer with a stride of 2.
4. 2 Fully Connected Layers with 400 neurons in each layer
5. Batch Norm after each convolutioal layer and fully connected layer

6. L2 regularization parameter - 0.003
7. Dropout at the convolution layer with a probability of 0.8 and in the fully connected layer with a probability of 0.5
8. Batch Size - 64
9. Number of Epochs - 20
10. Initialization - Xavier Initialization
11. Data Augmentation - 3x(total) by flipping the image left to right and top to bottom.
12. Decay Learning rate - Divide the learning rate by 2 after every 10 epochs
13. Early Stopping with a patience of 5 epochs

These parameters gave us an accuracy of 56.7% on 30% of the test data-set, and gave us an accuracy of 54% on the validation set.

2 Dimensions

1. CONV 1 input - [64,64,3], CONV 1 output - [64,64,32]
2. CONV 2 input - [64,64,32]. The convolution operation is with padding "SAME" which is why there is no shrinkage in the height and width of the tensor. CONV 2 output - [64,64,32]
3. Pool1 input - [64,64,32]. The convolution operation is with padding "SAME" which is why there is no shrinkage in the height and width of the tensor. Pool1 output - [32,32,32]
4. CONV 3 input - [32,32,32], CONV 3 output - [32,32,64]
5. CONV 4 input - [32,32,64], CONV4 output - [32,32,64]
6. Pool 2 input - [32,32,64], Pool 2 output - [16,16,64]
7. CONV 5 input - [16,16,64], CONV 5 output - [16,16,64]
8. CONV 6 input - [16,16,64], CONV 6 output - [13,13,128]
9. Pool 3 input - [14,14,128], Pool 3 output - [7,7,128]
10. FC1 input - [7,7,128] which is flattened into a [6272,1] vector. Fc1 output - [256,1]
11. Softmax input - [256,1] vector which produces a [20,1] output

3 Number of Parameters

1. CONV1 Parameters - 2400
2. CONV2 Parameters - 25600
3. Pool 1 Layer - 0
4. CONV3 Paramters - 18432
5. CONV4 Parameters - 36864

6. Pool 2 Layer - 0
7. CONV5 Parameters - 36864
8. CONV6 Parameters - 73728
9. Pool 3 Layer - 0
10. FC1 Layer - 1605632
11. Softmax Layer - 5120

Total Parameters - 1804640, Fully Connected Parameters - 1610752, Convolutional Parameters - 193888

4 Number of Neurons

1. CONV1 Layer Neurons - 12288
2. CONV2 Neurons - 131072
3. Pool 1 Neurons - 131072
4. CONV3 Neurons - 32768
5. CONV4 Neurons - 65536
6. Pool 2 Neurons - 65536
7. CONV5 Neurons - 16384
8. CONV6 Neurons - 16384
9. Pool 3 Neurons - 25088
10. FC1 Neurons - 6272
11. Hidden Layer Neurons - 256
12. Softmax Neurons - 20

Total Neurons - 502676

5 Effect of Using Batch Normalization

Shown in Figure 1 and 2 are the plots for the training error and validation error versus the epoch at which it has been calculated. The parameters while running this experiment were - Batch Size - 64, Probability for Dropout in Convolutional Layer - 0.8, Probability for dropout in Fully Connected Layer - 0.5, L2 Regularization Parameter - 0.003, 3x Augmentation of data using flipped images and Xavier Initialization for the weights.

Obserevation: We observe that using Batch Norm clearly improves performance significantly over both training and validation sets.

6 Plot of CONV1 Filters

Figure 3 contains the plots of the 32 filters in the first convolutional layer. The weights of the filters have been scaled between 0 to 1 for plotting purposes. Since the filter sizes are too small(5 x 5), it is difficult to interpret the patterns captured by these filters visually.

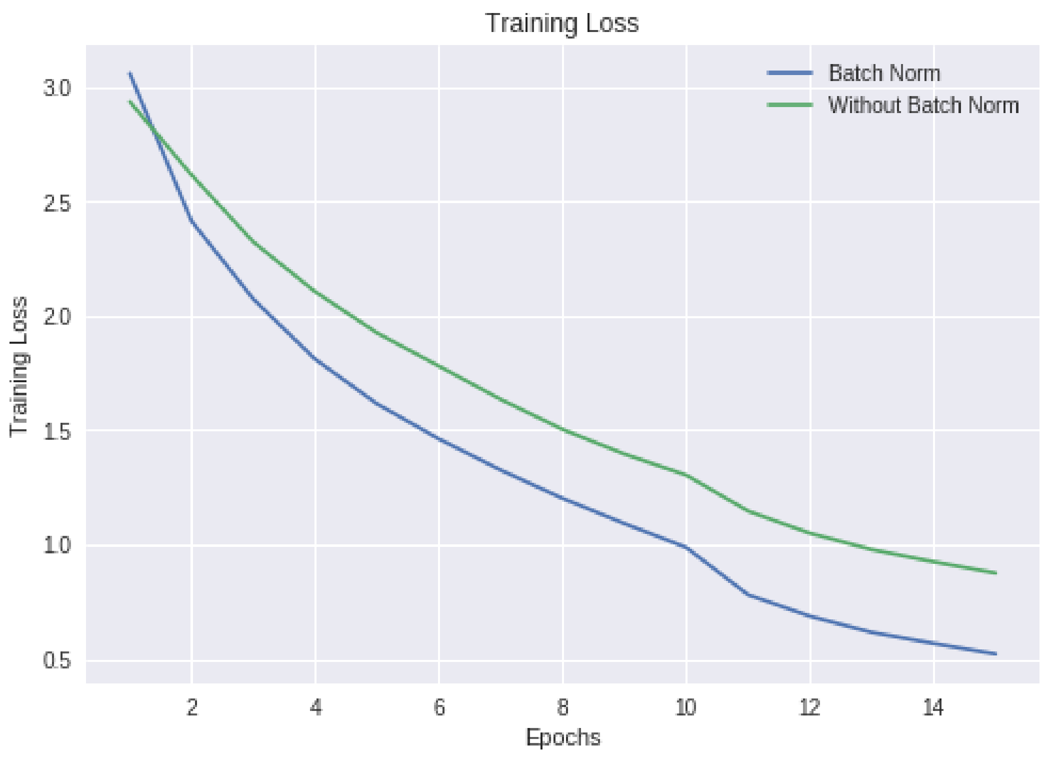

Figure 1: Analysis of the use of Batch Norm

7 Training and Validation Loss plots for Different Learning Rates

Shown in Figure 4 and 5 are the plots for the training error and validation error versus the epoch at which it has been calculated. The parameters while running this experiment were - Batch Size - 64, Probability for Dropout in Convolutional Layer - 0.8, Probability for dropout in Fully Connected Layer - 0.5, L2 Regularization Parameter - 0.003, Batch Normalization at every layer, 3x Augmentation of data using flipped images and Xavier Initialization for the weights.

Observation: For the learning rates chosen for this experiment, the training plot displays a trend of lower training loss as learning rate decreases. But the validation loss displays a different trend. lr = 0.01 is very noisy and slow to converge, whereas lr = 0.005 converges steadily without too much noise. The lower learning rate values of 0.0007 and 0.0001 initially have faster convergence, but the validation loss starts increasing after around 10 epochs.

8 Training and Validation Loss plots for Different Initialization Techniques

Shown in Figure 6 and 7 are the plots for the training error and validation error versus the epoch at which it has been calculated. The parameters while running this experiment were - Batch Size - 64, Probability for Dropout in Convolutional Layer - 0.8, Probability for dropout in Fully Connected Layer - 0.5, L2 Regularization Parameter - 0.003, Batch Normalization at every layer, 3x Augmentation of data using flipped images and a learning rate of 0.0007.

Observation: According to the training loss curve, Xavier initialization is slightly better than He initial-

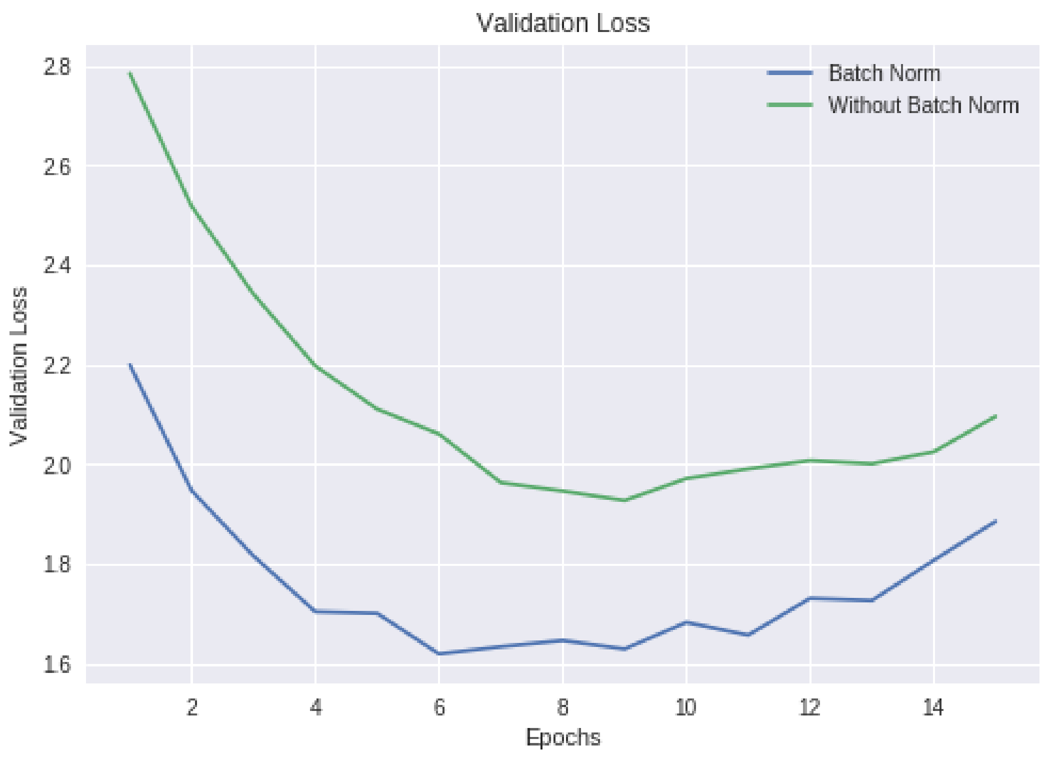

Figure 2: Analysis of the use of batch norm

ization. According to the validation loss curve, the initial performance of both strategies is similar, and He initialization slightly outperforms Xavier initialization as the training progresses.

9 Training and Validation Loss plots for Different Batch Sizes

Shown in Figure 8 and 9 are the plots for the training error and validation error versus the epoch at which it has been calculated. The parameters while running this experiment were - Probability for Dropout in Convolutional Layer - 0.8, Probability for dropout in Fully Connected Layer - 0.5, L2 Regularization Parameter - 0.003, Batch Normalization at every layer, 3x Augmentation of data using flipped images and a learning rate of 0.0007.

Observation: According to the training loss curve, the loss curves for all 4 batch sizes used in this experiment steadily decrease, with larger batch sizes(300 and 150) performing slightly worse than smaller ones(10 and 65) on the training data. According to the validation loss curves, batch size of 10 diverges significantly as compared to other three. Batch size 300 is slower to converge as compared to 10 and 65, which perform reasonably well. Both large(300) and small(10) batch sizes are not optimal.

10 Why Validation Accuracy instead of Validation Error for Early Stopping

We have used validation accuracy instead of validation error for Early Stopping. We observed the validation accuracy curve to keep decreasing while validation loss began increasing. One of the reasons for this discrepancy is the hard thresholding used for calculating the accuracy, since ultimately we are going to predict a

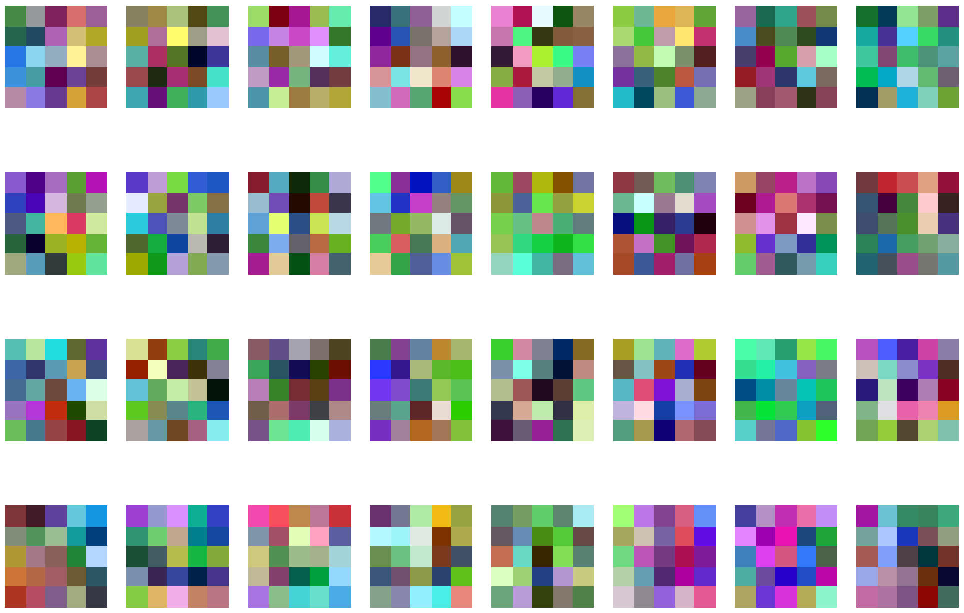
Figure 3: Filters for the first convolutional layer

single class(which is what we are going to use the neural network for, and thus, this is what we care about more) for an image even if the predicted probabilities of multiple classes are close in their values. This problem is aggravated in case of large number of classes. It is important not only to increase the probability of the correct class, but also to maximize the gap between the predicted probabilities of the correct class and that of the next lower predicted probability.

11 Guided BackPropagation

We implemented guided backpropagation on a neuron in the CONV6 layer to see which type of input maxes it fire maximally. The figure that was input to the model and the figure obtained after guided backpropagation are shown in Figure 10 and Figure 11 respectively.

Althoug the guided backprop image is not super informative when viewed separately, but when we compare it with the input image, we feel that the network is not swayed away towards irrelevant aspects of the input image, and is also able to identify something close to "green blobs/patches" which is also what the input image has.

12 Fooling The Network

We were able to fool the network by changing the input image slightly. A dog was now classified as a house after adding some noise to the image. The original image was updated based on the gradient of the predicted probability with respect to the input. The two pictures are shown in Figure 12 and Figure 13.

The changes made to the input image are minor for a human observer, who would still classify the image as a dog, but according to the network, the probability of the image being a house is increasing. If we run this for more iterations so as to increase the probability of house significantly, at some point the input image starts to distort for the human observer as well.

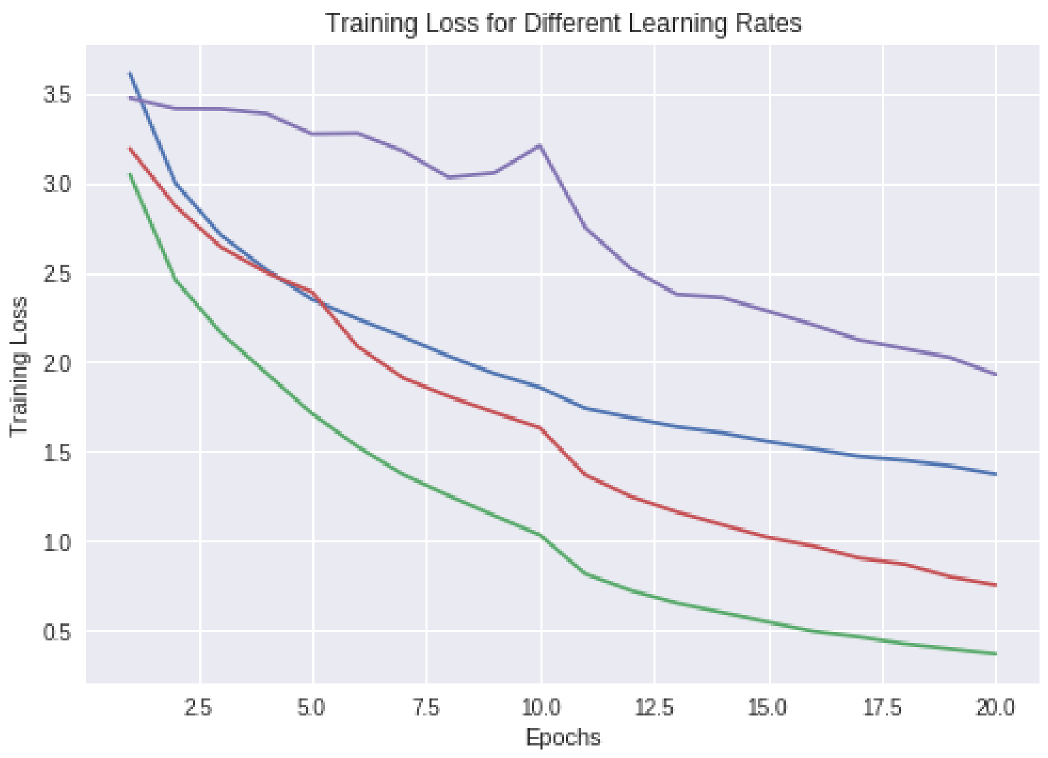

Figure 4: Training Loss for Different Learning Rates. lr-0.01(Purple), lr-0.005(Blue), lr-0.0007(Red), lr-0.0001(Green)

13 Techniques used for improvement in Accuracy

We used techniques such as dropout, data augmentation and learning rate decay to boost our accuracy on the validation and test set.

1. Data Augmentation : We tripled the size of the training data-set by flipping the original images left to right and top to bottom. We used tensorflow ops to do the same.
2. Dropout : We used dropout with a probability of 0.8 in the convolutional layers and with a probability of 0.5 in the fully connected layers. These values were taken based on a paper published by Geoff Hinton and his students.
3. Learning Rate Decay : After every 10 epochs, we decay our learning rate by a factor of 2.

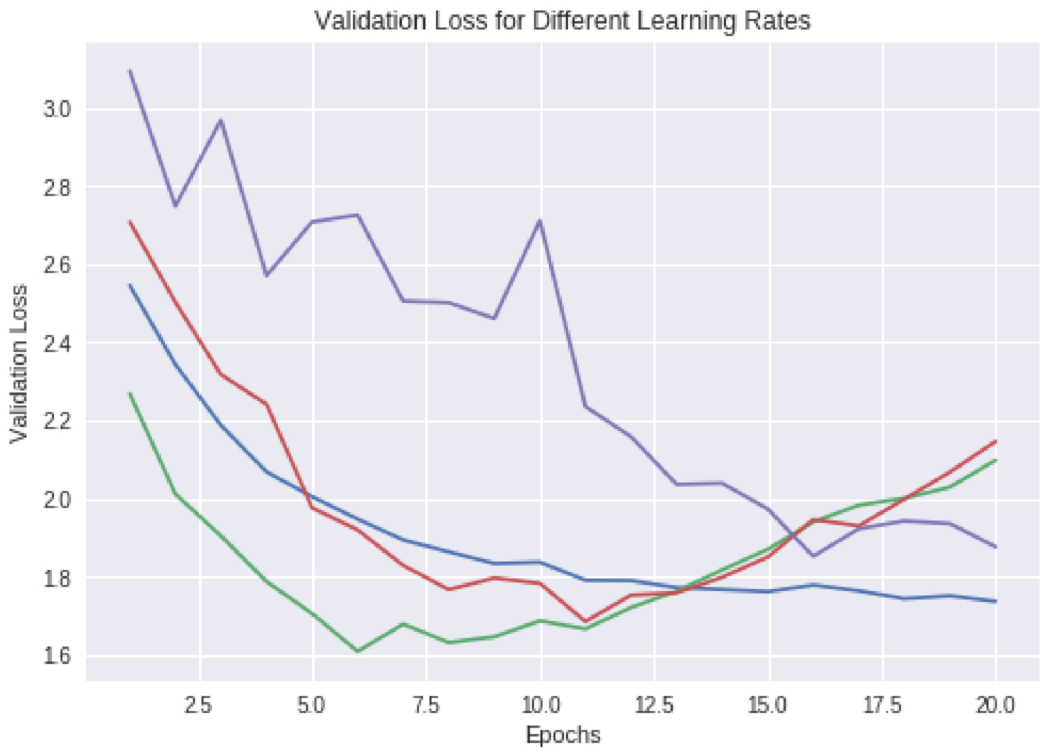

Figure 5: Validation Loss for Different Learning Rates. lr-0.01(Purple), lr-0.005(Blue), lr-0.0007(Red), lr-0.0001(Green)

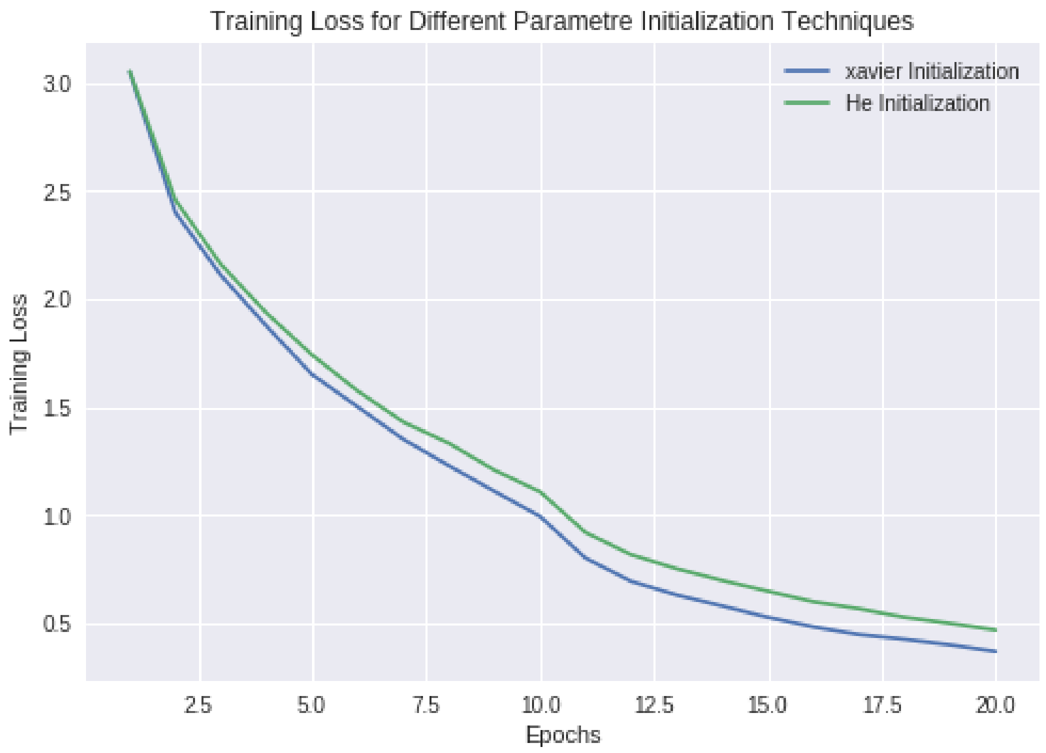

Figure 6: Training Loss for Different Initialization Techniques

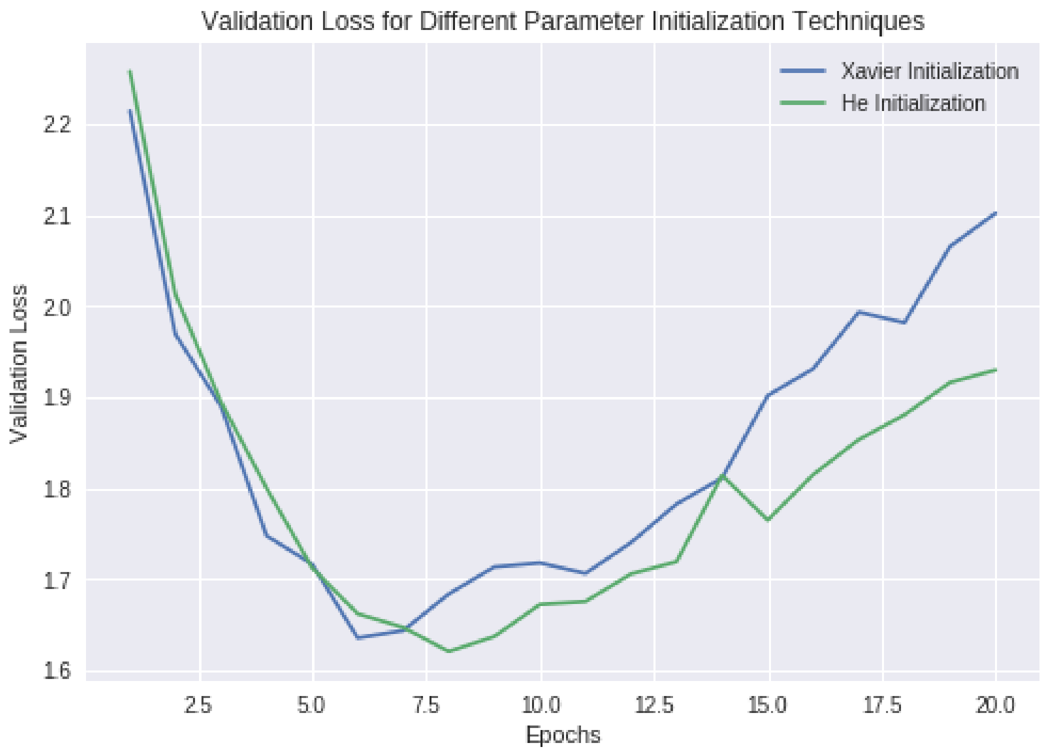

Figure 7: Validation Loss for Different Initialization Techniques

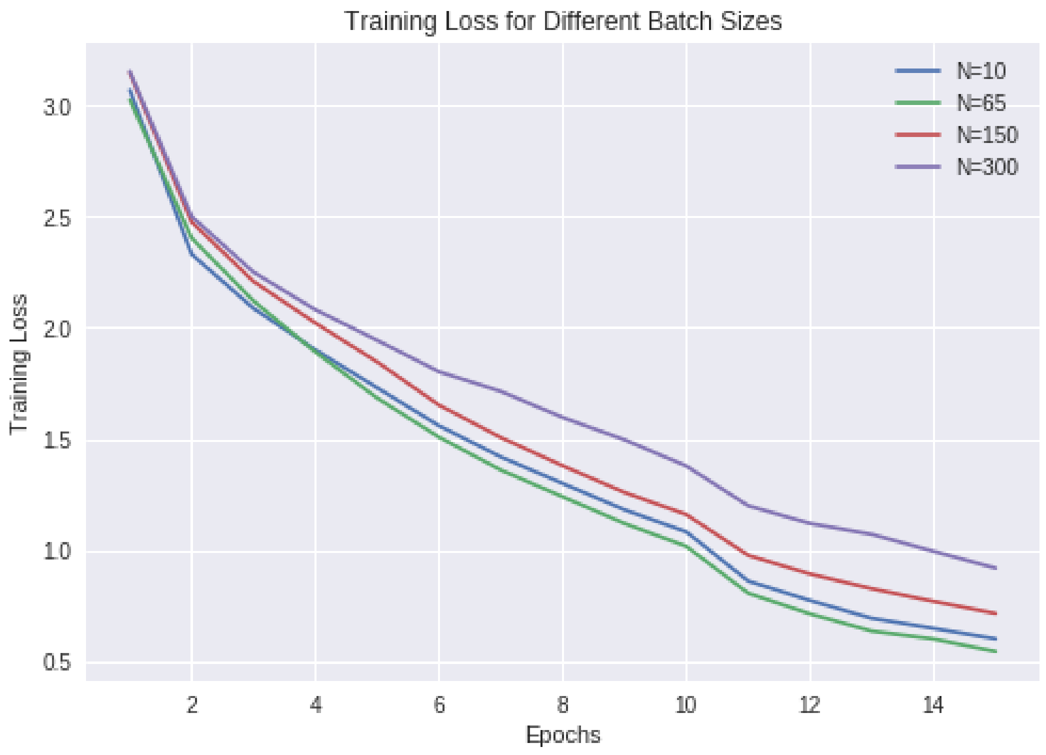

Figure 8: Training Loss for Different Batch Sizes

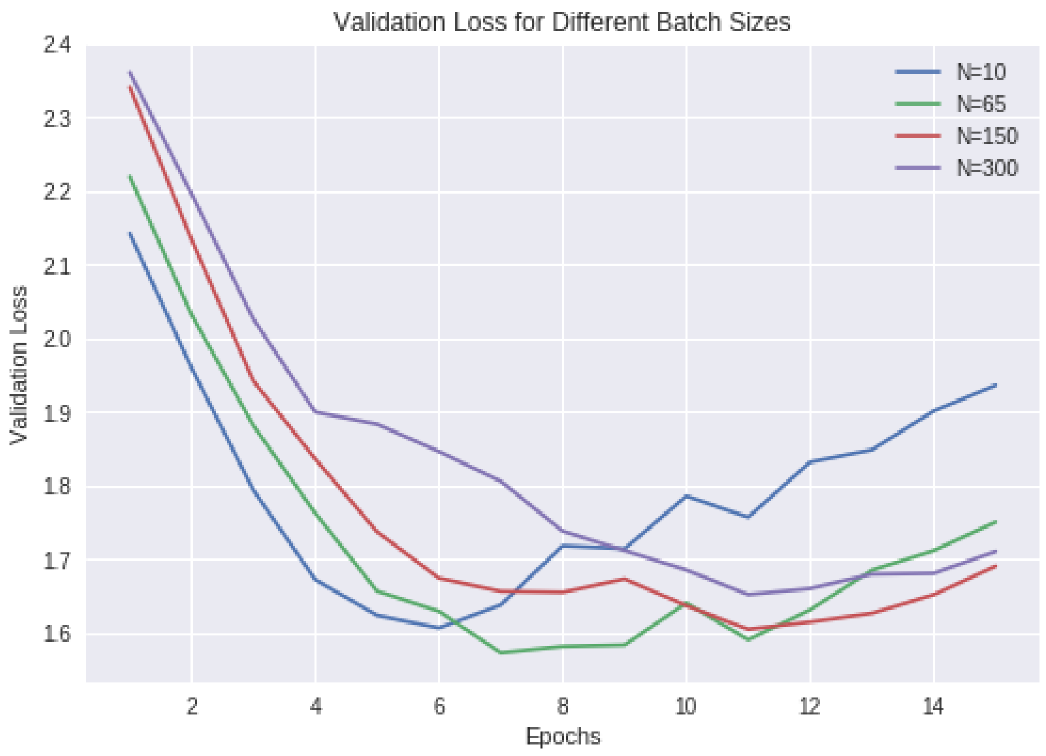

Figure 9: Validation Loss for Different Batch Sizes

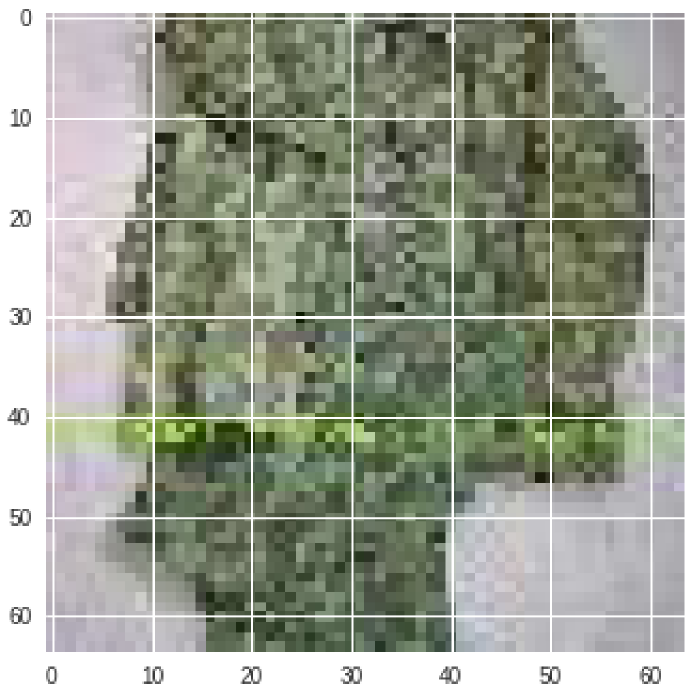

Figure 10: Original Image

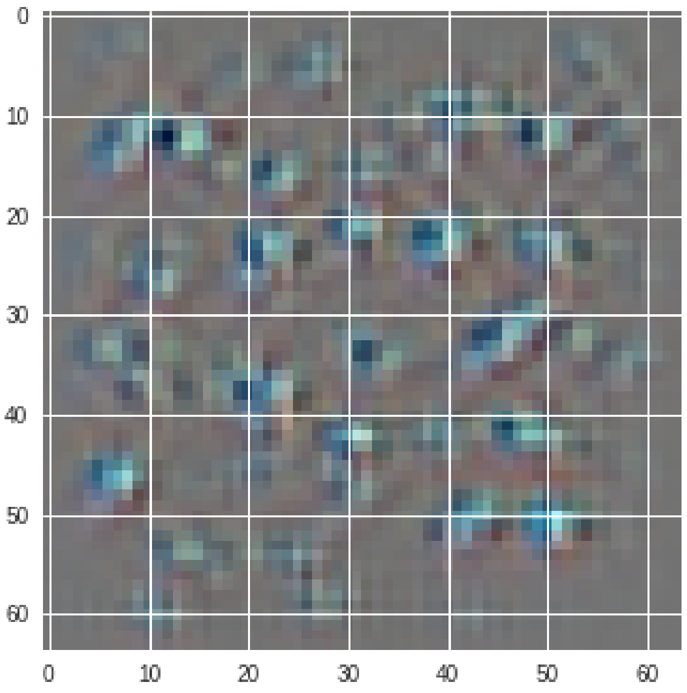

Figure 11: Image Obtained after Guided Backpropagation

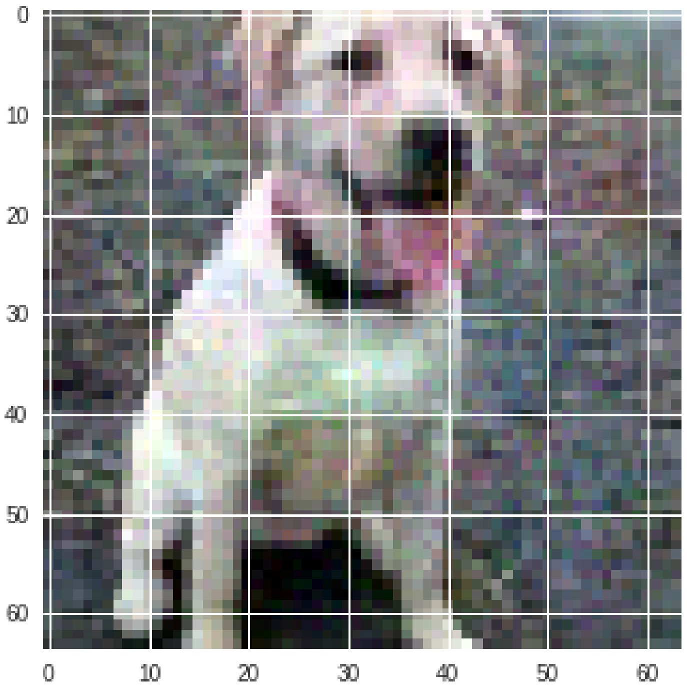

Figure 12: Original Image of a Dog

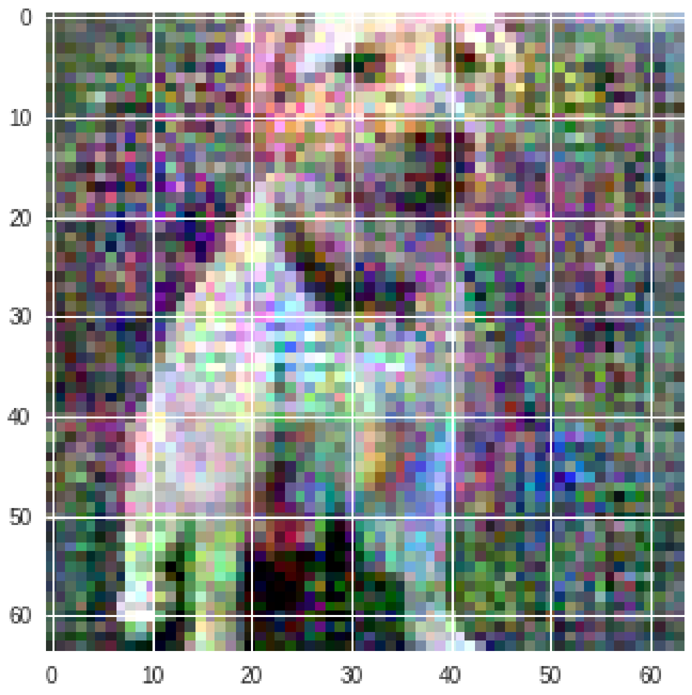

Figure 13: The same image which the network now classifies as a house