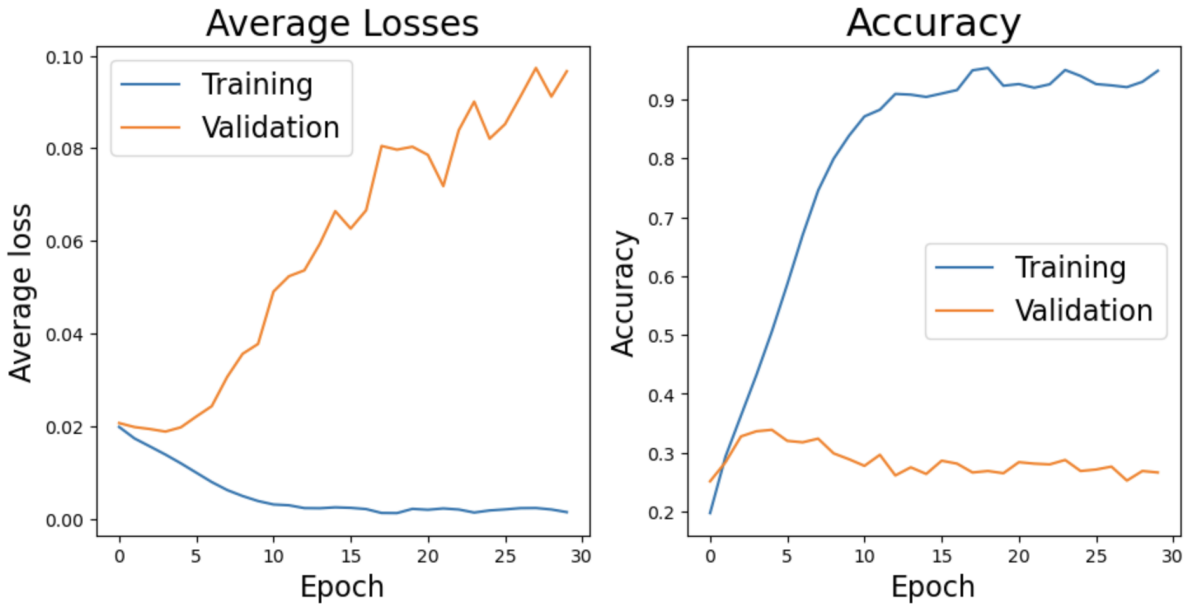
Question 1: Why is it important to have separate train and validation sets? What might happen if you don't have a validation set?

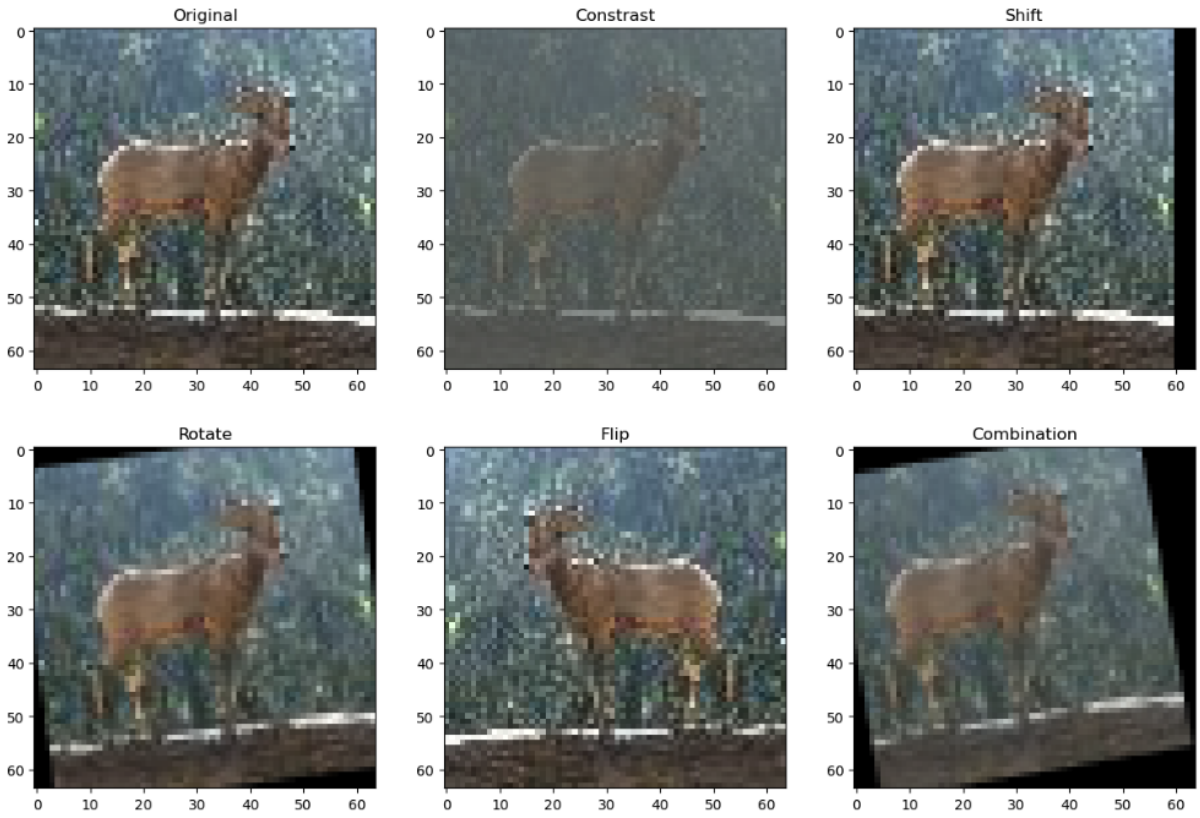
It is important to have separate training and validation sets because the ultimate goal of a model is not to learn a function that fits the training data as best as it can, but learn a function that can explain/fit the data well enough so that it learns the underlying function that generated the data (and therefore enables it to correctly fit new data). If you do not have a validation set, the model will try and optimize for the first goal we mentioned (fitting the training data as best as it can), which will lead to the model possibly overfitting to the training data and not being able to generalize well onto new unseen data during test time. By having a validation set, you are pushing the model to optimize for goal number two (fit well enough to generalize) rather than goal one.

Question 2a: Include the plots generated from training the baseline model from scratch (i.e., without loading in pretrained weights). What do you notice about the train vs. validation performance? What might this mean about the model?



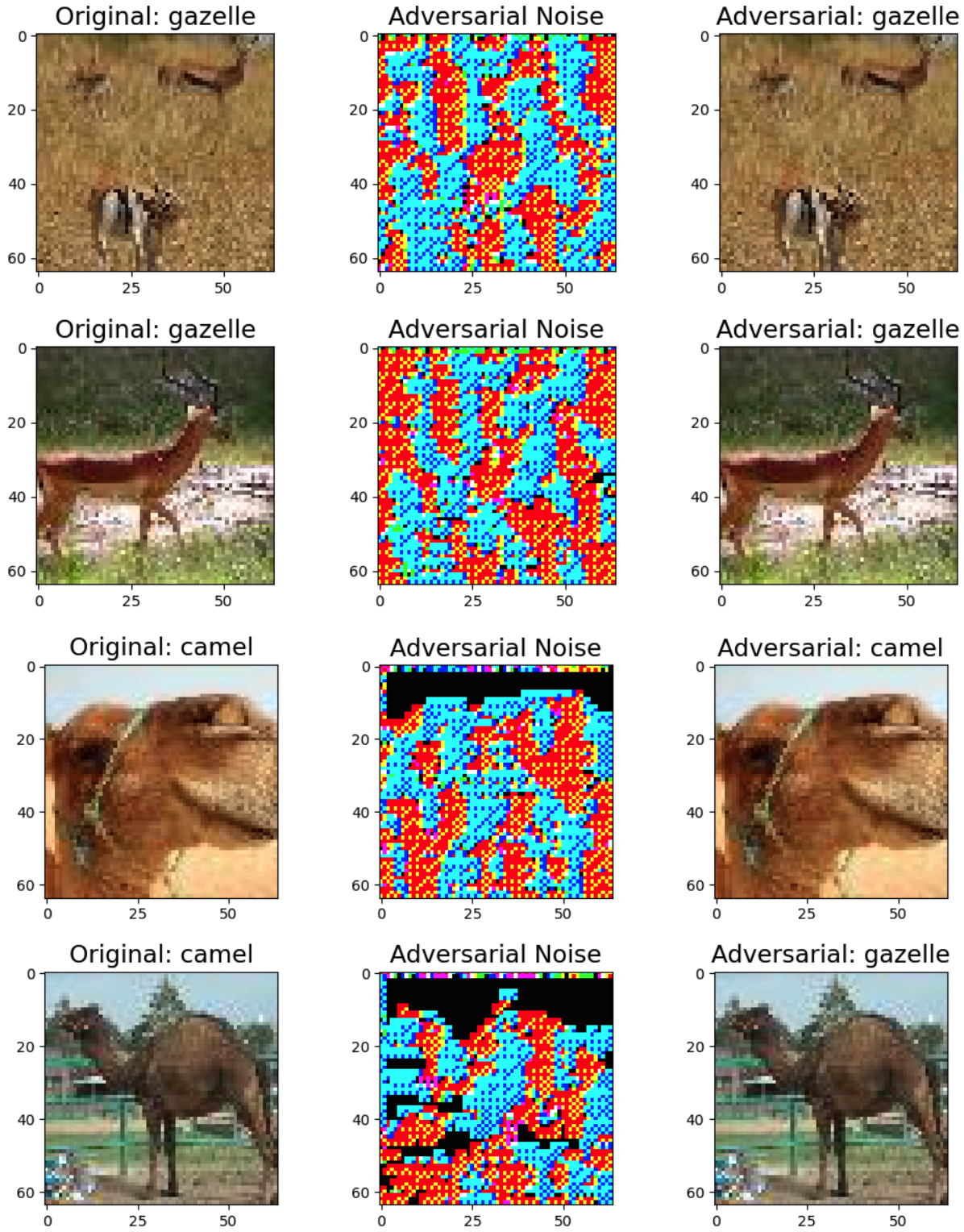
Looking at the plots above, we can see that as we train for more epochs, the training performance is getting better and better while the validation performance is getting progressively worse. This likely means that our model is overfitting to our training data and is unable to generalize well onto unseen data (the validation data in this case). Perhaps the number of data points we are training on is too limited and we need a larger set of data points in order for the model to truly be able to generalize well.

Question 2b: Here is an image and various transformations that we have applied to the image as a means of data augmentation…



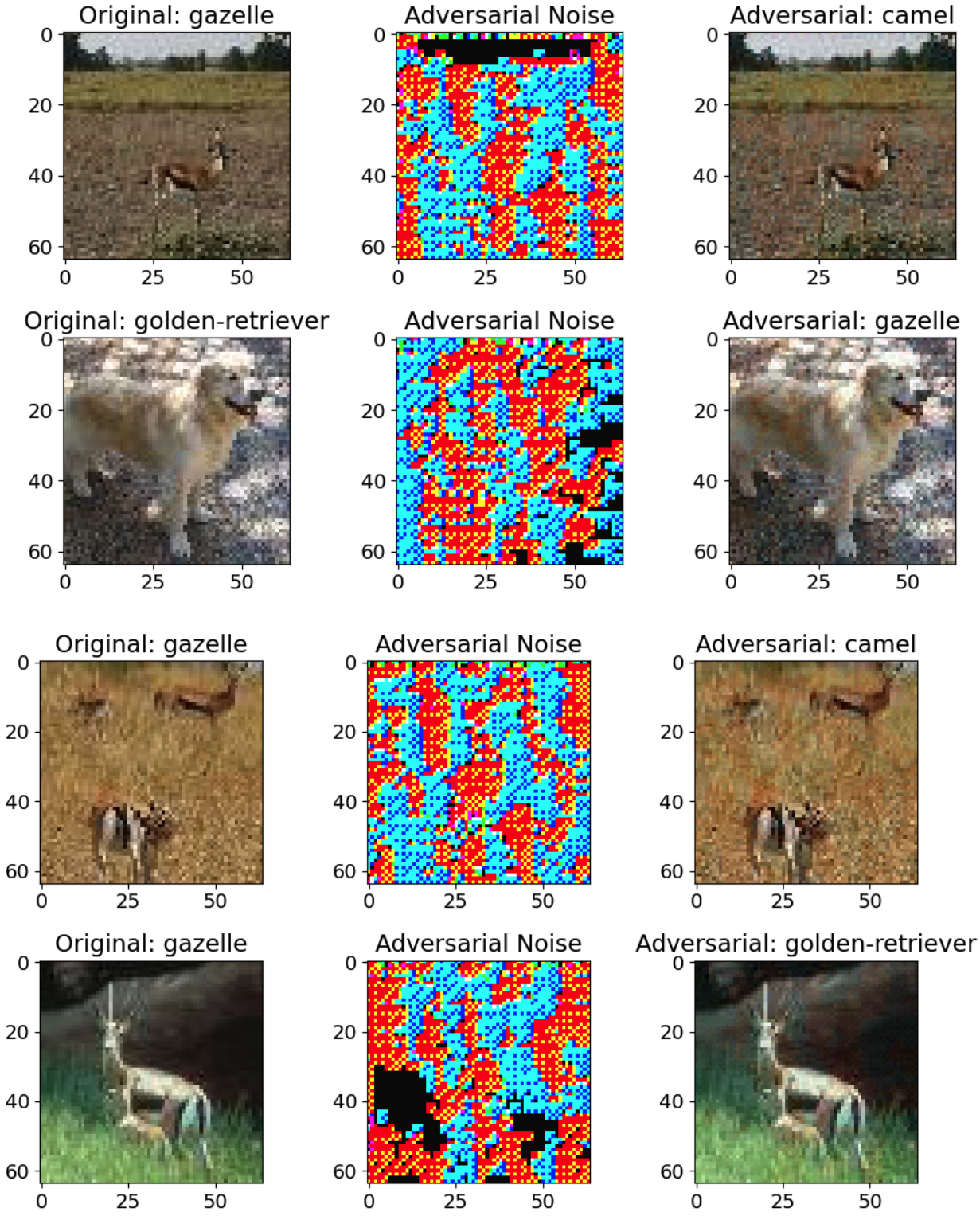
Question 4: Include 3 adversarial image plots from the cells below in your writeup: one with epsilon=0.005, one with epsilon=0.02, and one with epsilon=0.15. As you increase epsilon, what happens to the misclassification rate? Why?

Epsilon=0.005 Test:



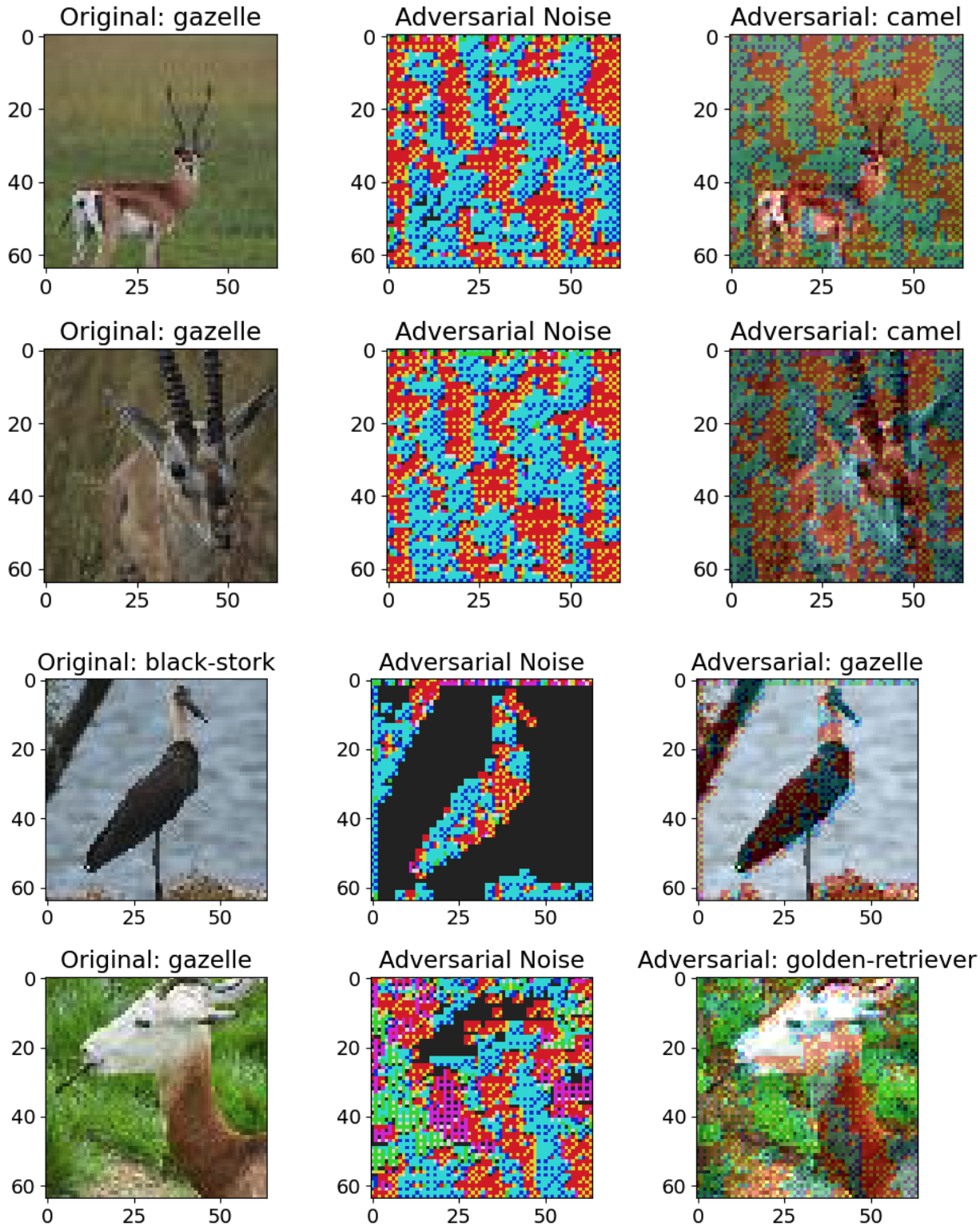
Misclassification Rate: Out of total 82 images generated, 32.9268% of adversarial images misclassified.

Epsilon=0.02 Test:



Misclassification Rate: Out of total 82 images generated, 90.2439% of adversarial images misclassified.

Epsilon=0.15 Test:



Misclassification Rate: Out of total 82 images generated, 100.0% of adversarial images misclassified.

As we increase the epsilon value, the misclassification rate increases. This is because the epsilon value controls the magnitude of the noise we are adding to the original image to create the adversarial image (the higher the epsilon value, the greater the magnitude of the noise). Given that the direction of this noise is the direction of the gradient, adding noise to our image will increase the loss, and the magnitude of the noise (as controlled by epsilon) will determine how much we increase the loss. The greater the magnitude, the greater the loss, and ultimately the higher the misclassification rate. Therefore it follows that increasing epsilon will increase the misclassification rate.