Rev	Changes	Notes
2.1	Tried using my own data. Data started in drive_data folder from ~two loops in each direction.	Trains, does not perform well.
3.1	Went back to trying the basic pipeline from the videos. Using a dense FC=1.	Trains and runs.
3.2	Added lambda normalization and image augmentation via flipping.	I am choosing to ignore the side camera images for now. I may come back to these later but it doenst seem like they should be nessesary to get the model up and running. The colors are inverted. (see link below). I am going to ignore this for now. Also I found a great resource here that addresses some issues that I foresee: https://discussions.udacity.com/t/behavioral-cloning-non-spoiler-hints/233194
3.3	Added keras cropping layer, takes out 50 pixels at the top and 20 at the bottom.	Seems to be working ok.
3.4	Trying to add Lenet convolutional architecture. 2 conv blocks, 6 channels, 5x5 stride. FC120,FC84,FC1	Right now I am having trouble connecting to AWS, so I am training on my MacBook for 3 epochs. It is taking about 330s (5.5mins) per epoch (not too bad). (At least I don't have to worry about it disconnecting from AWS...) The model is sort-of driving, but defintly not very well. Although I think this is the best I have done with a convnet.
3.5	Trying the same but with 6 epochs.	
3.6	Trying NVIDIA network. Conv:24(w/2x2subsample),36(w/2x2subsample),48(w/2x2subsample),64,64, Dense: 100,50,10. 3 Epochs.	Working pretty well! Makes it past bridge. Has trouble with water line. Update: I accidentally overwrote the 3.6 model file and have not been able to duplicate that performance.
3.7	Inceasing number of epochs with NVIDIA model.	Signs of overfitting.
3.8	Adding visualizations of training and validation loss. Put in a batch size of 512. 2 Epochs.	Best performance yet.
3.9	Adding a histogram of steering angles. Trying 3 Epochs.	Lots of zeros... might need to filter some of these out or try to get better data. Best performance yet! Note that I tried running at different speeds in the drive.py file (see drive_tevis_edits.py) and got the best performance out of speeds less than 20.
3.10	Changed first FC layer to 200 units.	3 epoch did pretty well. 5 not as well, overfitting.
4.1	Added single dropout layer after all convolutions. 0.5 keep_prob.	No improvment.
4.2	Simplified the model a bit.	No improvment.
5.1	Implimented Canny transform in notebook and in drive_tevis_edits.py, on top of rev3.8 architecture. High threshold = 160 Low threshold = 1 kernel = 5	Good performance, great proof of concept, but still fails at the curve after the bridge. Sometimes has trouble before the bridge as well.
IDEA	Make it save every epoch then I can just over-train every time and then just go back and pick the best model, delete the others.	Looking into this. http://machinelearningmastery.com/check-point-deep-learning-models-keras/
5.2	Trained for 5 epochs.	Validation accuracy kept going down, but the driving performance is worse than 5.1.
5.3	Changed canny high threshold to 60. Changed keras cropping to only take to 60 rows out (instead of 70). Training for 4 epochs.	Doesn't make the first sharp turn.
IDEA	Go back to 3 input channels, but instead oof 3 colors, feed in 3 different canny maps of varying complexity.	
5.4	More epochs (7)	No improvment.
5.5	High Threshold 100, kernel 7.	Not good. Seems underfit? I'm going to see how many epochs it takes before validation score begins to drop.
Skype meeting with Aman (4/28)	Try more epochs on 3.9. Look at colorspace. Try dropout. numpy to extract color channel image[:,:,n] selects only n channel.	
3.11	Changed back to 3.9 arcitecture. Added second histogram for before and after flipping. Performed downsampling of zero-angle records at keep prob of 0.33. Train for 4 epochs.	Looks promising but it has trouble on the bridge, gets too close to the left side.
3.12	Tried .5 downsampling,	No significant improvment.
3.13	Changed cropping to 60 from 70.	
6.1	Adding dropout to 3.9. 5 Epochs.	Seemingly good performance, possibly best yet. Almost made the corner after the bridge.
6.2	Implimented checkpoint system.	

6.3	Updates to checkpoint system.	Checkpoint system working well. Helping to learn the differences between underfitting and overfitting. It seems that the best sign of UNDERFITTING is an inability to corner sharply enough (drifting off the road), whereas a signs of OVERFITTING are jerkyness, over-correction, and steering into the wall on the bridge.
NEW DATA	Collecting new data.	First run: 10100. Second Run:10500 (bridge and hard corner). Third Run: 10990 (bridge and hard corner). Fourth run: 13378 (Backwards lap).
7.1	Using new data on 3.9 architecture	
7.2	Adding dropout 0.5. Started from a restart after Epoch 1, so epochs are one more than they claim to be in file names.	Signs of overfitting.
NOTE	Made some cosmetic/jistogram changes on R7.2. Start from here from now on for new models.	
7.3	Put in 0.25 dropout layer between each hidden layer of the network.	Not training at all. (Car doesn't steer.)
8.1	Trying image resizing, using cv2.resize(). Resized to 64x64. Starting with Udacity data on 3.9 architecture. (with7.2 processing updates). Had to take out last 2 conv layers though due to dimentional reduction.	MUCH FASTER Training time. Makes it all the way around, but with some errors, particularly in the hard curve. Like the best model so far at 5 epochs, but seems a fit jerky (possibly overtrained).
8.2	Added padding and put last 2 nividia conv layers back in.	Not working well.
8.3	Added dropout between FC layer, 0.5 keep_prob.	Not working well.
8.4-8.9	Various tweaks with the resized images, nothing working very well.	
9.1	Trying mutiple cameras. Starting with 8.1 architecture.	Shows promise. Needs less epochs of training.
9.2	Adding one more conv layer. (The first 64).	Probably the best yet. Model 9.2.01 goes all the way around the track, and handles the usual hard spots quite well, but has a couple of its own tough spots. 9_2_3-00: Best yet. Steering correction = 0.3.
#####	Model 9_2_3_1-00 makes it all the way around the track without ever hitting a curb (except just very slightly brushing it in one spot). This model may be sufficient to pass the project.	