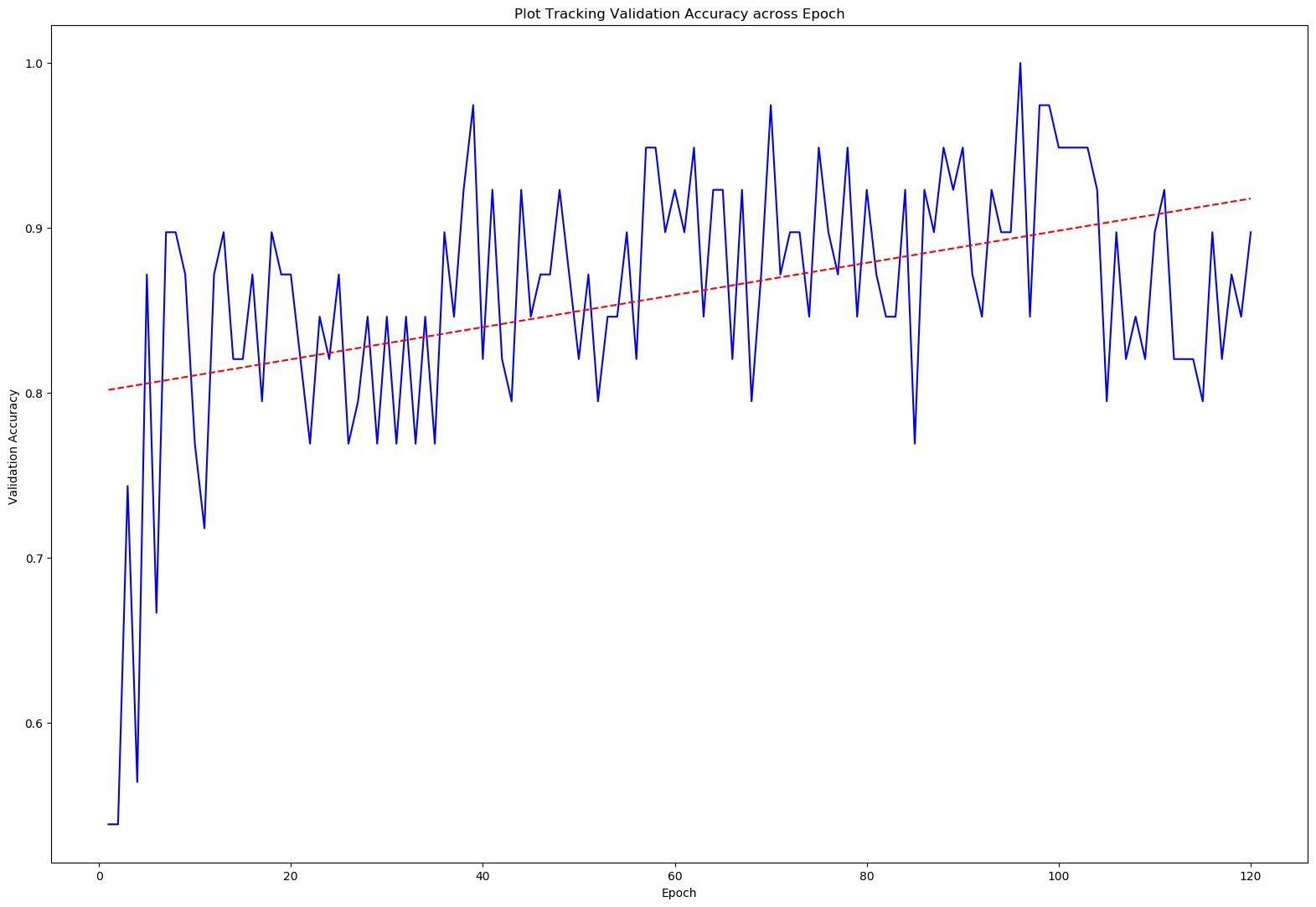
**Cone Challenge Report**

**Analyzing & Loading Data**

The training and test images were loaded into the program with a 512x512 pixel frame with 3 layers, for the RGB values. The data was normalized to fit in between 0 and 1 for yielding more accurate modeling results.

**Model Construction**

We constructed the model with 3 convolution layers using Conv2D and MaxPooing2D for each one. Later we flattened the data and fully connected it by putting it through a dense function with 128 units using the ‘relu’ activation function. One area that we added where we saw a great guide for performance was the adding of a penalty to avoid overfitting by using L2 regularizer. After that, we compiled the CNN with an adam optimizer, measured the loss with binary\_crossentropy, and measured accuracy. When fitting the data, we decided to take 10% of the training data and commit it to validation to ensure the model was behaving as designed.

****

**Figure 1**. Tracking Validation Accuracy across Epoch

By using 120 Epochs, we were able to train our model with limited data. Our output was given to us in a probability and then rounded to either a 0 or a 1. Then from there we loaded it to a pandas data frame then exported to a CSV.

**Conclusion**

After training our model with the training data, the model was run again with the testing images to predict whether a cone was present or not. After submitting output file to Kaggle, we found that the results gave a 100% accuracy. This reflects that the modeled that we created was very good at predicting whether a cone was present or not. One thing that we could do differently is doing some sort of data augmentation, like using the ImageDataGenerator class from Keras. This would have been helpful as the given dataset was small.

**References**

Image Classification using Neural Network: <https://becominghuman.ai/building-an-image-classifier-using-deep-learning-in-python-totally-from-a-beginners-perspective-be8dbaf22dd8>

Sequential Model documentation: <https://keras.io/models/sequential/>