## **Assignment 6 - Deep Learning TensorFlow**

**August 10, 2018**

Zixuan Yu/Chung-Yang Li

### **Part A - Deep Learning model**

* Apply a Deep Learning model to your project data (if possible). Validate the accuracy.
* The Deep Learning model can be a MLP, CNN, RNN, Autoencoder, Variational autoencoder (VAE), Restricted Boltzmann machine (RBM), Deep belief network (DBN) or Generative Model. \

### **Part B - Activation function**

On *your Deep Learning model data* apply at least two different activation functions.

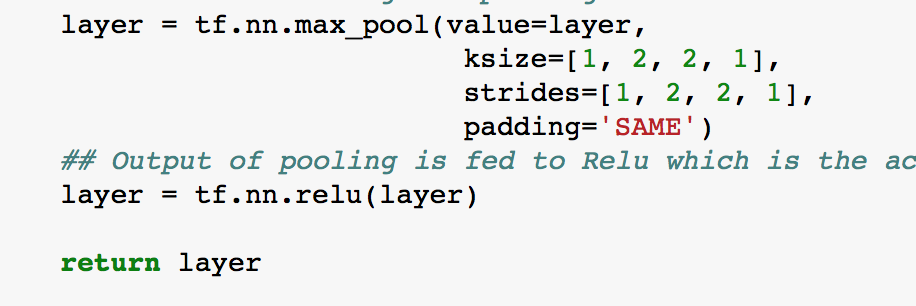
* Change the activation function. How does it effect the accuracy?

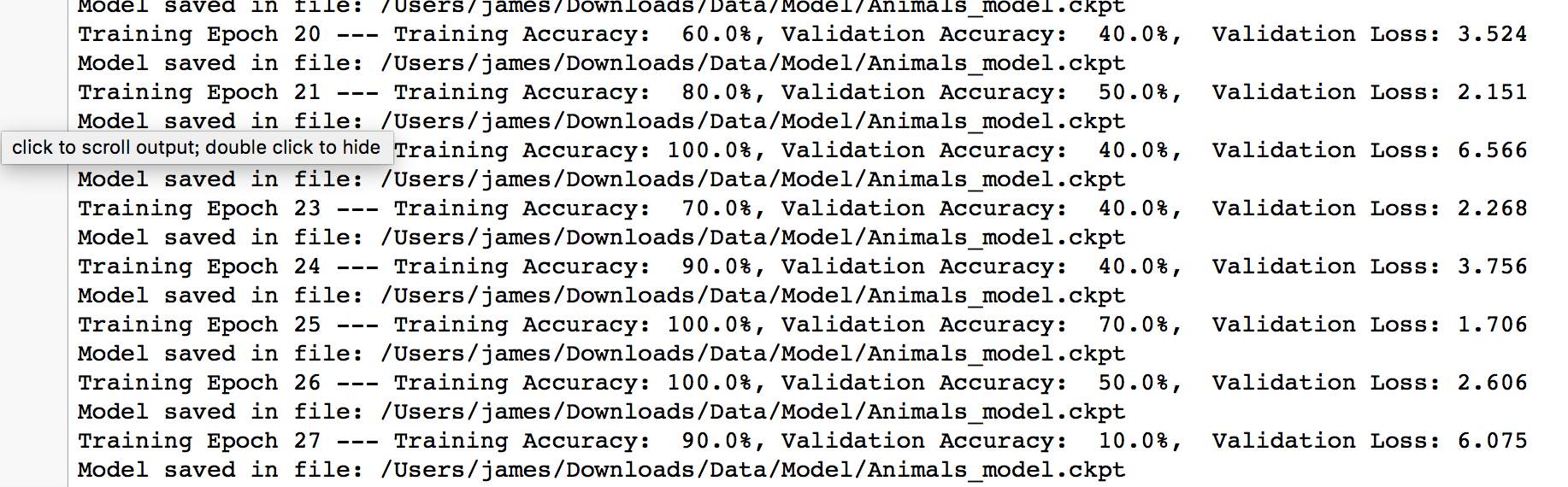
In our case, Selu activation function perform slightly better than Relu.

* How does it effect how quickly the network plateaus?

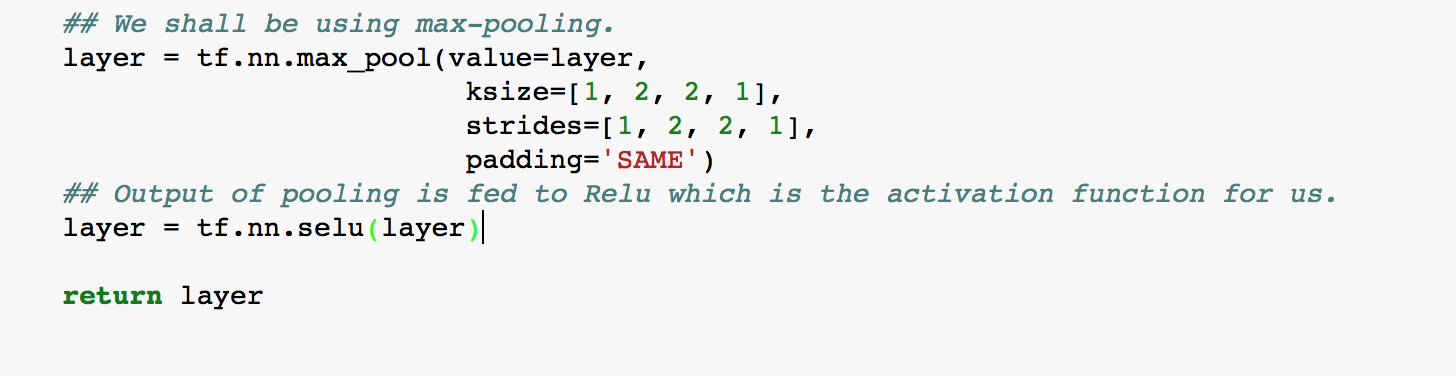
Two activation function have same pace on training.

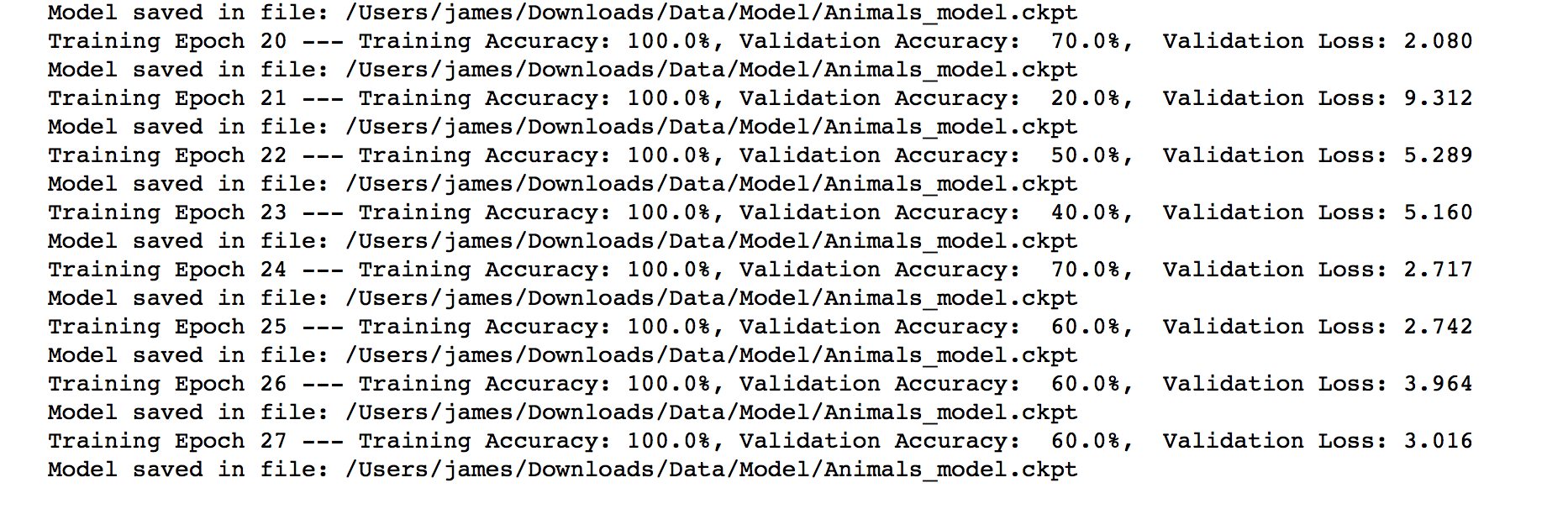
* Leaky rectified linear unit (Leaky ReLU)





* Scaled exponential linear unit (SELU)





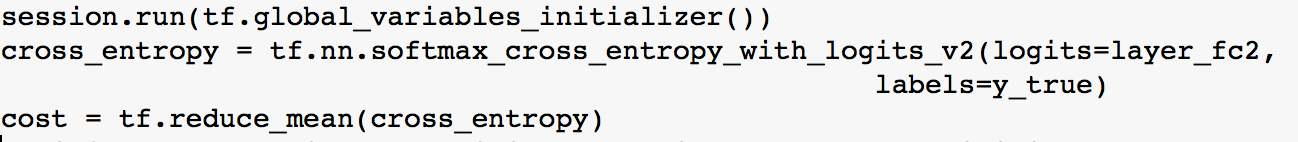
### **Part C - Cost function (10 points)**

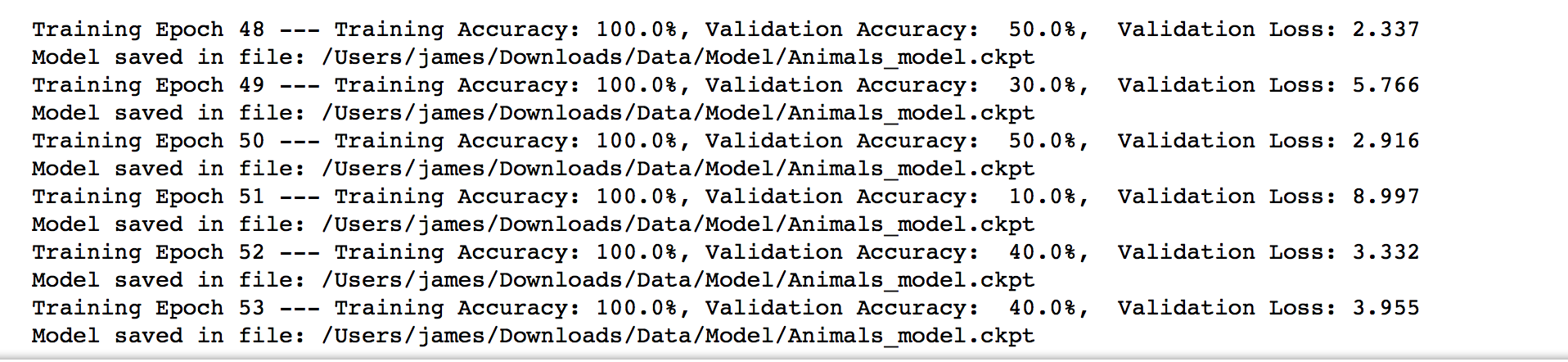
On *your Deep Learning model data* at least two different cost functions.

* Change the cost function. How does it effect the accuracy?
* How does it effect how quickly the network plateaus?

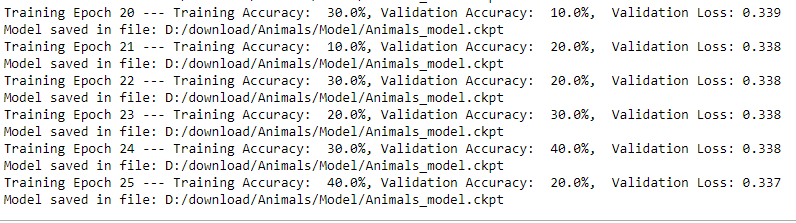
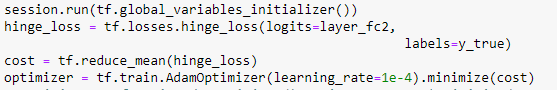
I found in my model, cross-entropy has a better accuracy.

* Cross-Entropy





* Hinge



### **Part D - Epochs**

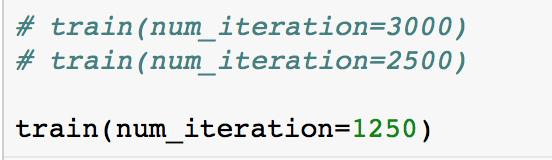
On *your Deep Learning model data*

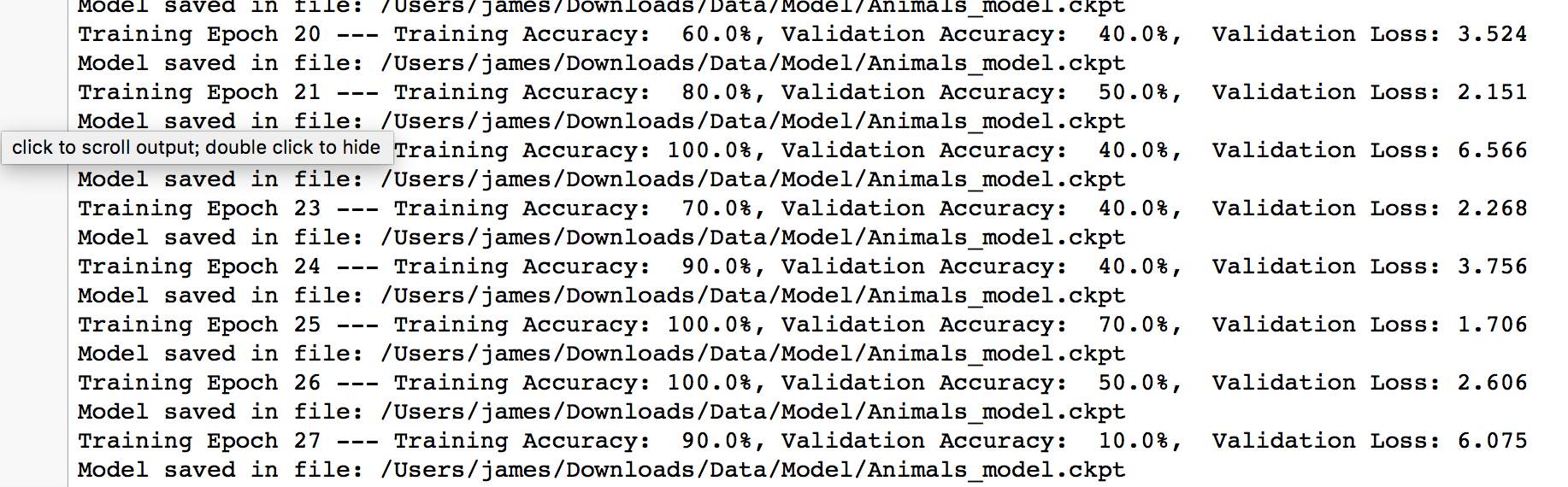
* Change the number of epochs initialization. How does it effect the accuracy?
* How quickly does the network plateau?

We change the value of iteration, 1250 and 1000.

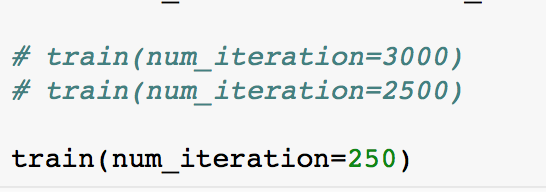
I found that when the iteration is 1000, we have better pace on training.

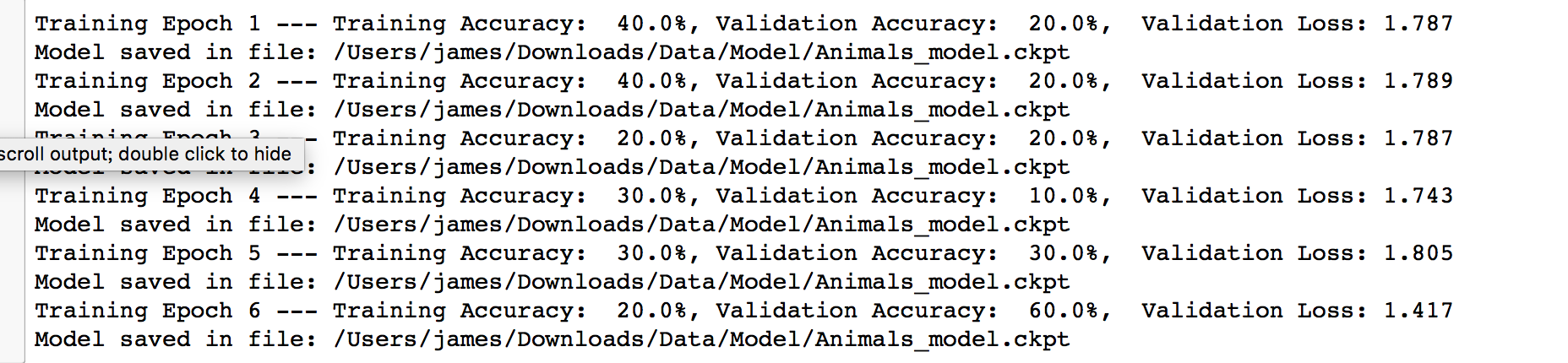
Iteration: 1250





Iteration: 250





### **Part E - Gradient estimation**

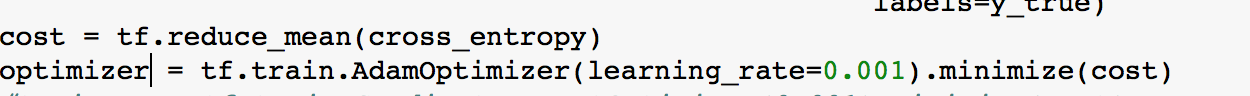
On *your Deep Learning model data* at least two qradient estimation algorithms.

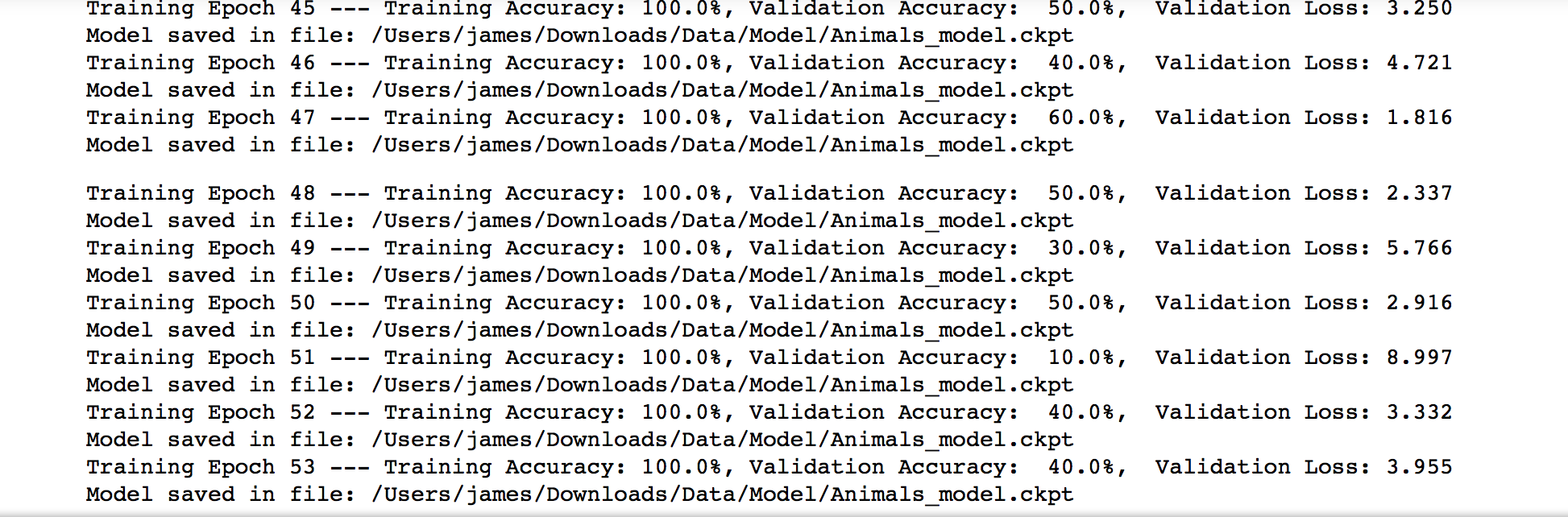
* Change the gradient estimation. How does it effect the accuracy?
* How does it effect how quickly the network plateaus?
* Various forms of gradient estimation:
* Stochastic Gradient Descent
* ADAM

we found that in my model, ADAM has better accuracy.

And I found that two ways have same pace on training.

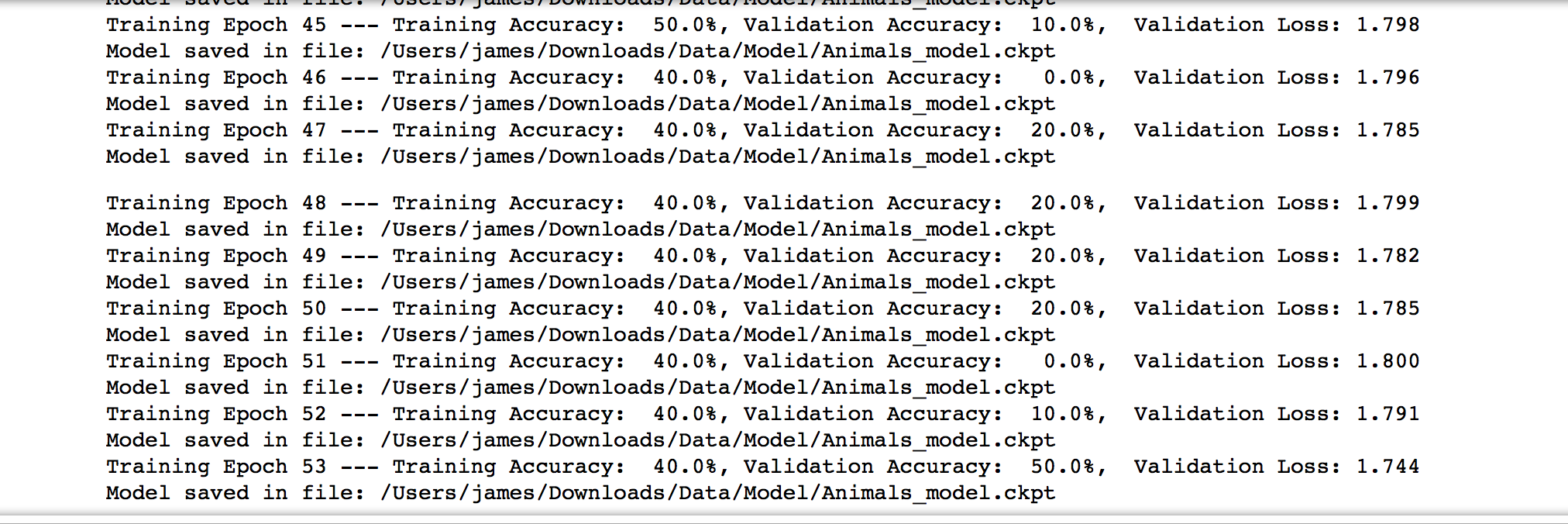
ADAM (learning rate =0.001)





Stochastic Gradient Descent( learning rate =0.001)





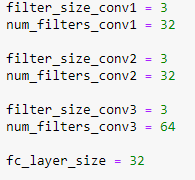
### **Part F - Network Architecture**

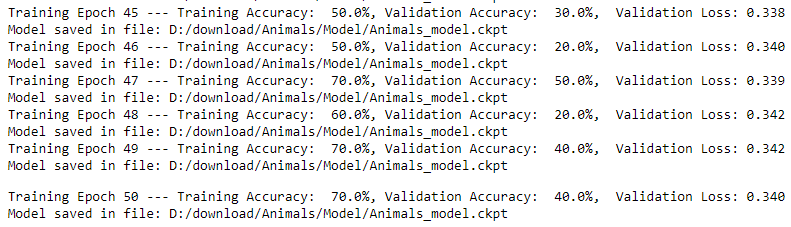
On *your Deep Learning model data*

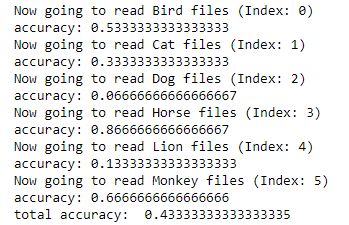
* Change the network architecture. How does it effect the accuracy?
* How does it effect how quickly the network plateaus?
* Various forms of network architecture:
* Number of layers
* Size of each layer
* Connection type
* Pre-trained components?

I tried to adjust filter size and increase the size of layer from 32 to 64, 64 to 128 and noticed that the bigger size have better training performance. The lower structure need more iteration to reach same accuracy as larger one. More number of layer do improve accuracy.

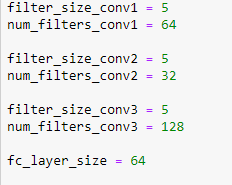
Orignal set:

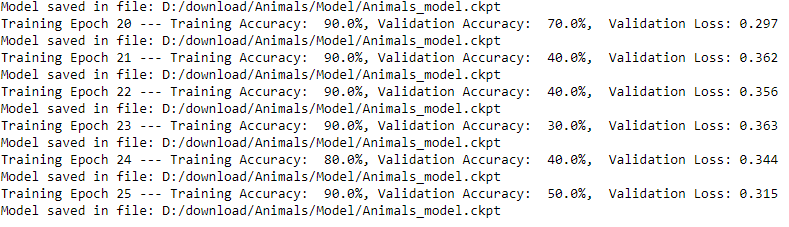


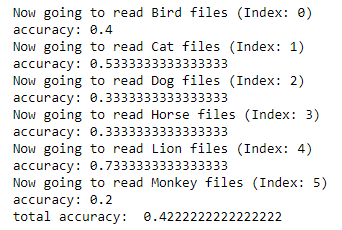




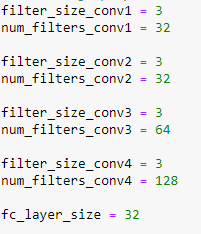
Larger layer size:

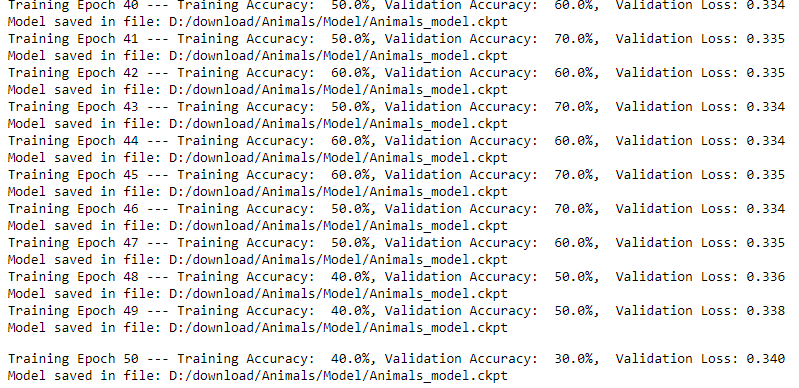


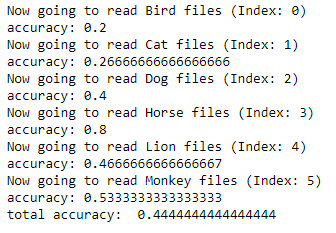




More number of layer:





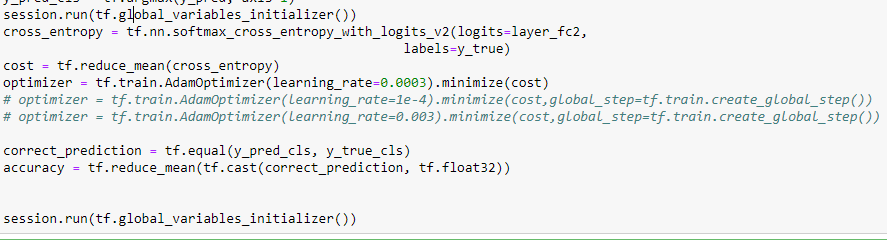


### **Part G - Network initialization**

On *your Deep Learning model data* at least two network initialization techniques.

* Change the network initialization. How does it effect the accuracy?
* How does it effect how quickly the network plateaus?
* Various forms of network initialization:
* 0
* Uniform
* Gaussian
* Xavier Glorot Initialization <http://andyljones.tumblr.com/post/110998971763/an-explanation-of-xavier-initialization>
  + Xavier Uniform
  + Xavier Gaussian

Global variable initializer:



### **Part H - TensorBoard**

Visualize Parts B thru G using TensorBoard.

