Can use callbacks to control the training process (e.g stop training after no acc increase for 2 epochs)

* Always start with defining the callback class
* Inside the class, define the functions (These will be predefined functions , e.g ‘on\_epoch\_end’). Find out more about these functions here: <https://www.tensorflow.org/api_docs/python/tf/keras/callbacks/Callback>

Class mycallback(tf.keras.callbacks.Callback):

More on defining the function can be found here: <https://www.tensorflow.org/guide/keras/custom_callback>

**class mycallback(tf.keras.callbacks.Callback):**

**def on\_epoch\_end(self,epoch,logs={}):**

**if(logs.get('accuracy')>0.98):**

**print('\n 98% Accuracy reached!')**

**self.model.stop\_training = True**

This function will be run every time one epoch has ended.

It will get the current accuracy of the model, and if it is > 98% it will stop the training

After defining the callback class, must define the callback using

callbk = mycallback()

And then you can use it during training by specifying

model.fit(..., callbacks=[callbk])