ADDING OUTPUT LAYER

```
prediction = Dense(len(folders), activation='softmax')(x)

class MyLayer(tf.keras.layers.Layer):
  def call(self, inputs):
    self.add_loss(tf.abs(tf.reduce_mean(inputs)))
    return inputs
l = MyLayer()
l(np.ones((10, 1)))
l.losses
[1.0]

inputs = tf.keras.Input(shape=(10,))
x = tf.keras.layers.Dense(10)(inputs)
outputs = tf.keras.layers.Dense(1)(x)
model = tf.keras.Model(inputs, outputs)
# Activity regularization.
len(model.losses)
0


model.add_loss(tf.abs(tf.reduce_mean(x)))
len(model.losses)


inputs = tf.keras.Input(shape=(10,))
d = tf.keras.layers.Dense(10, kernel_initializer='ones')
x = d(inputs)
outputs = tf.keras.layers.Dense(1)(x)
model = tf.keras.Model(inputs, outputs)
# Weight regularization.
model.add_loss(lambda: tf.reduce_mean(d.kernel))
model.losses
[<tf.Tensor: shape=(), dtype=float32, numpy=1.0>]


input = tf.keras.layers.Input(shape=(3,))
d = tf.keras.layers.Dense(2)
output = d(input)
d.add_metric(tf.reduce_max(output), name='max')
d.add_metric(tf.reduce_min(output), name='min')
[m.name for m in d.metrics]
['max', 'min']
```

Output

Retrieves the output tensor(s) of a layer.

Only applicable if the layer has exactly one output, i.e. if it is connected to one incoming layer.