Exercise 17 - Investigating Deep Neural Network architectures for cryptocurrency price predictions (in R & Tensorflow)

March 06, 2018

- *Installed TensorFlow CPU with R wrapper via Python on Anaconda.*
- *Installed TensorFlow GPU with CUDA via cuDNN package.*

Used raw TensorFlow for predicting ETH (ETH-7, BTC-3):

Performance = 0.54 [single layer]

- *Installed TensorFlow CPU with R wrapper via Keras API accessing Python on Anaconda with C++ core.*

Used Keras API for TensorFlow for predicting ETH (ETH-7, BTC-3):

Performance = 0.54 (150 it), 0.52 (1500 it. - overfitting) [1 hidden layer with 10 hidden nodes]

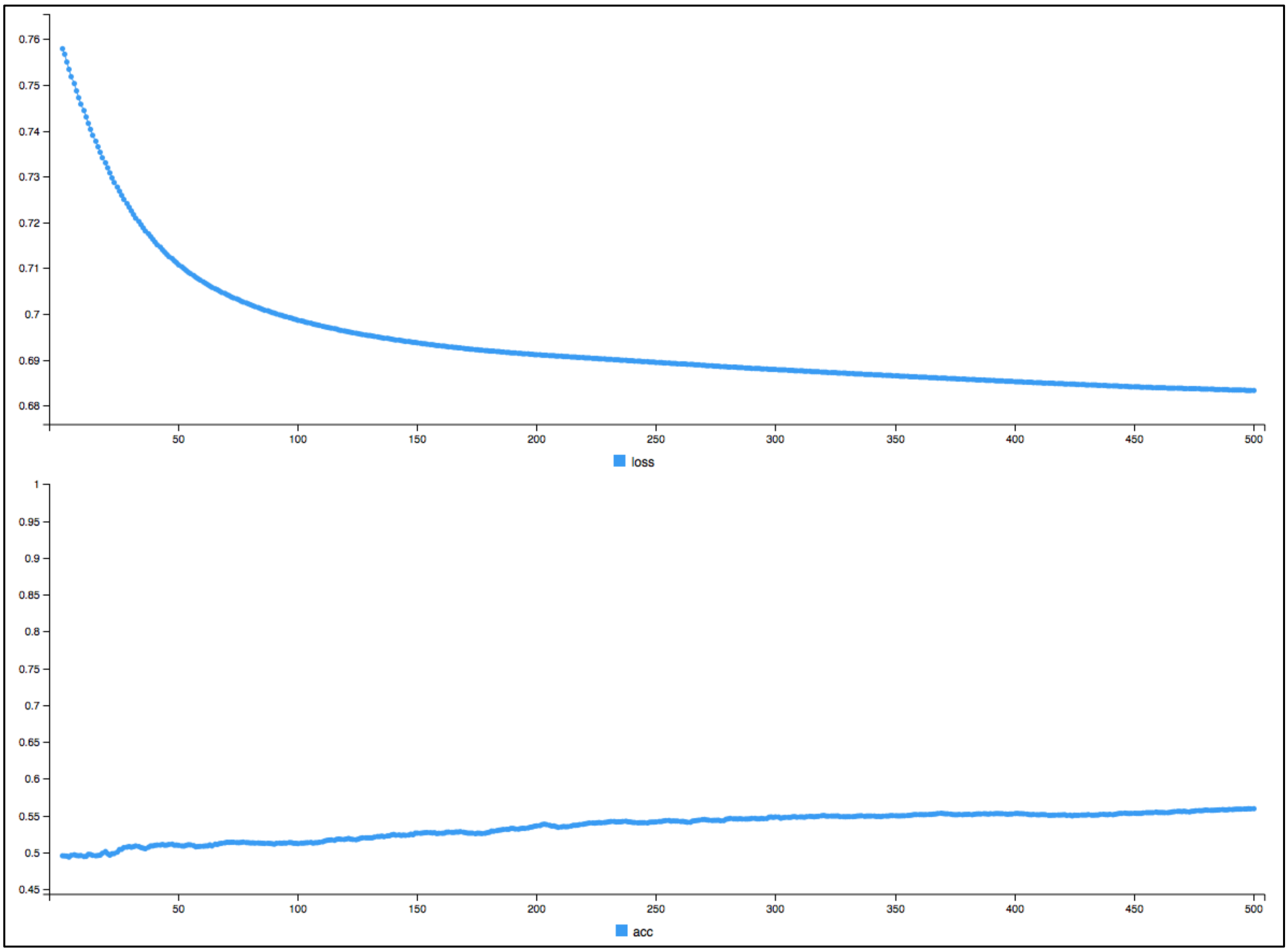


Figure 1 - Plots of cost function and accuracy for ETH prediction

Trying by shuffling the input data + printing probabilities:

Performance = 0.53 for 300 iterations [1 hidden layer with 10 hidden nodes]

Trying by shuffling the input data + only taking the 60% confidence or above:

Performance = 0.59 for 100 iterations [1 hidden layer with 10 hidden nodes] (198 preds of 768 in test set)

Using Keras API for TensorFlow GPU (CUDA, cuDNN) via Python on Anaconda (C++ Core) for predicting DASH (DASH Close to DASH-7, BTC-1, BTC-2, BTC-3)

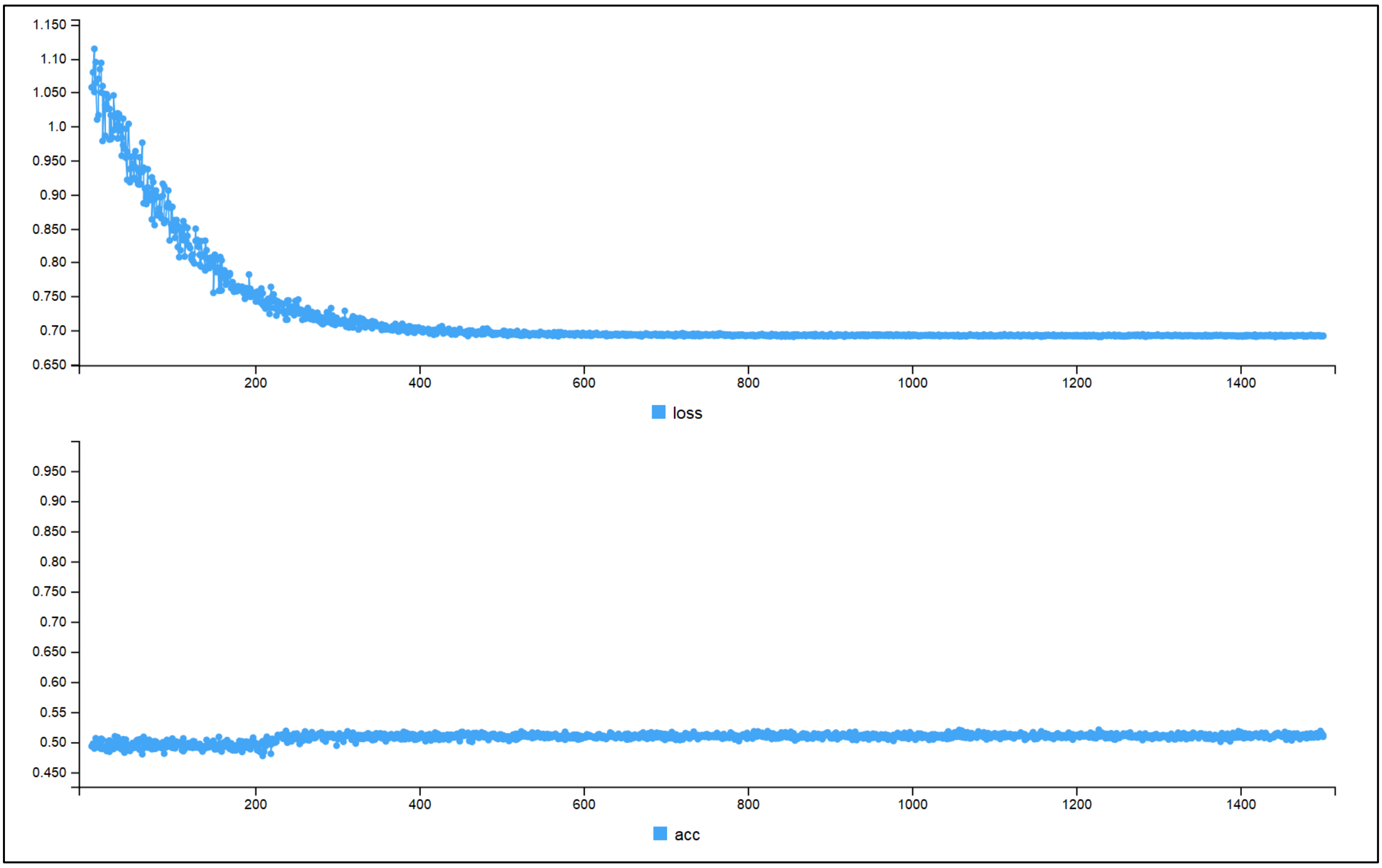
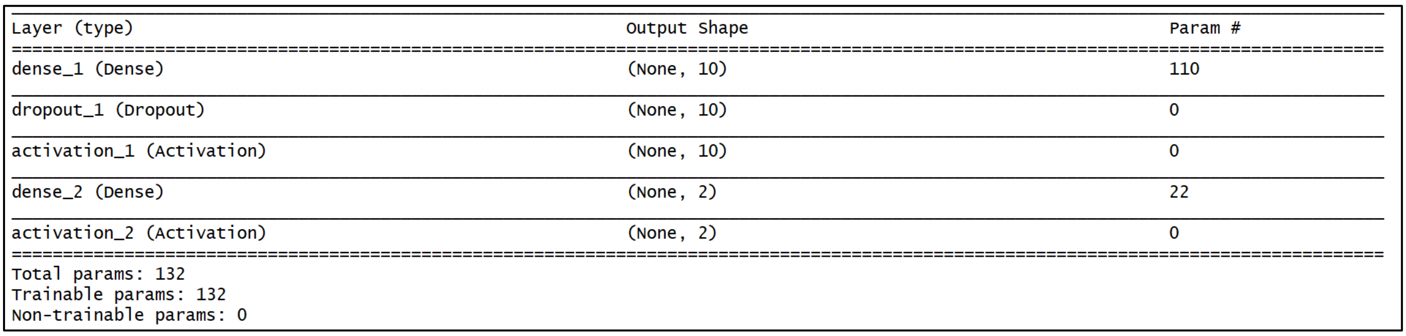


Figure 2 - Plots of cost function and accuracy for DASH prediction



```
Layer (type)                                      Output Shape                                 Param #
==========================================================================================================
dense_1 (Dense)                                   (None, 10)                                   110
__________________________________________________________________________________________________________
dropout_1 (Dropout)                               (None, 10)                                   0
__________________________________________________________________________________________________________
activation_1 (Activation)                         (None, 10)                                   0
__________________________________________________________________________________________________________
dense_2 (Dense)                                   (None, 2)                                    22
__________________________________________________________________________________________________________
activation_2 (Activation)                         (None, 2)                                    0
==========================================================================================================
Total params: 132
Trainable params: 132
Non-trainable params: 0
```

Figure 3 - Summary of model

Performance = 0.52 (1500 it.) [1 hidden layer with 10 hidden nodes] dropout = 0.9

Trying by shuffling the input data into mini-batches + printing probabilities:

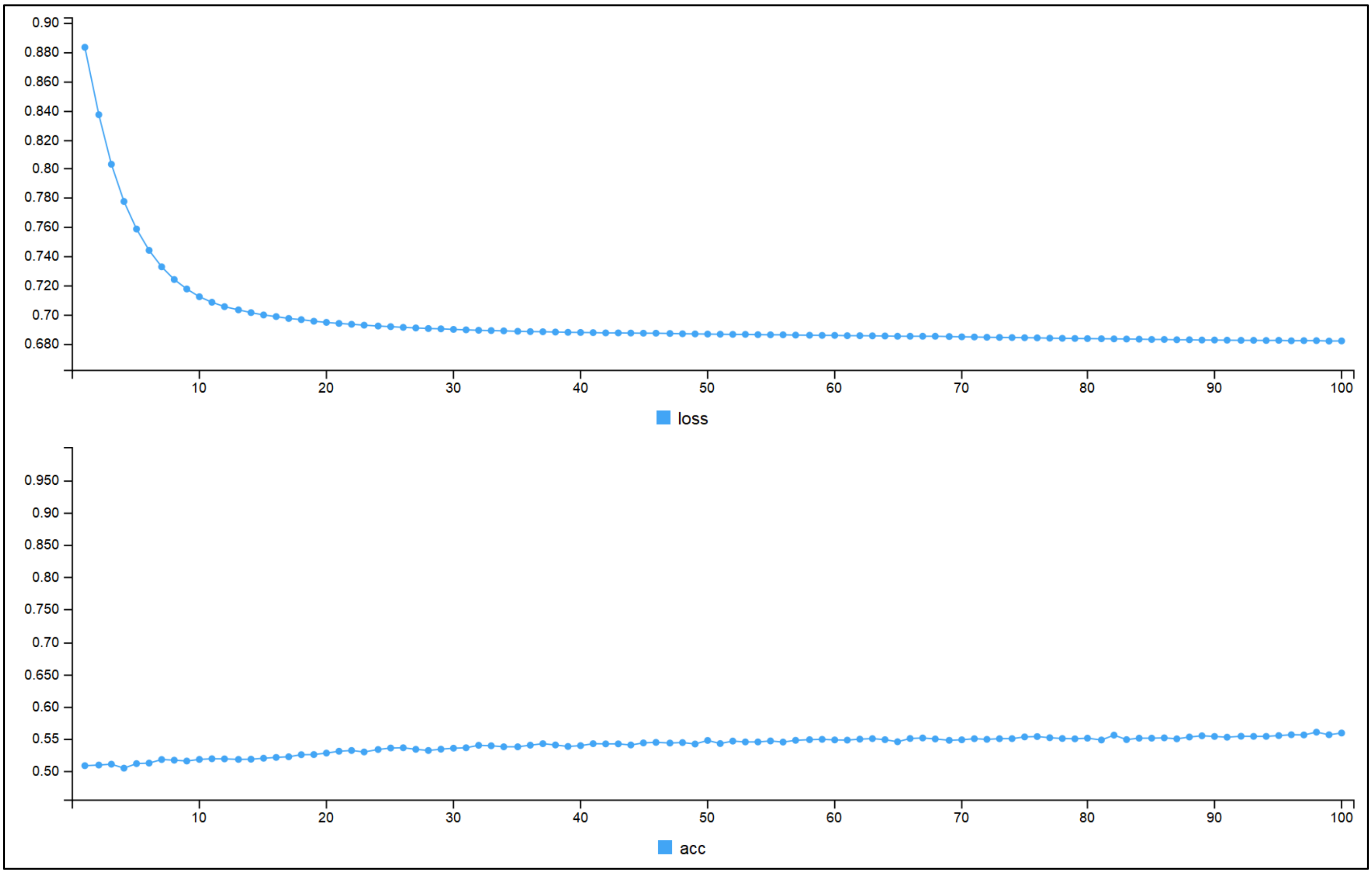
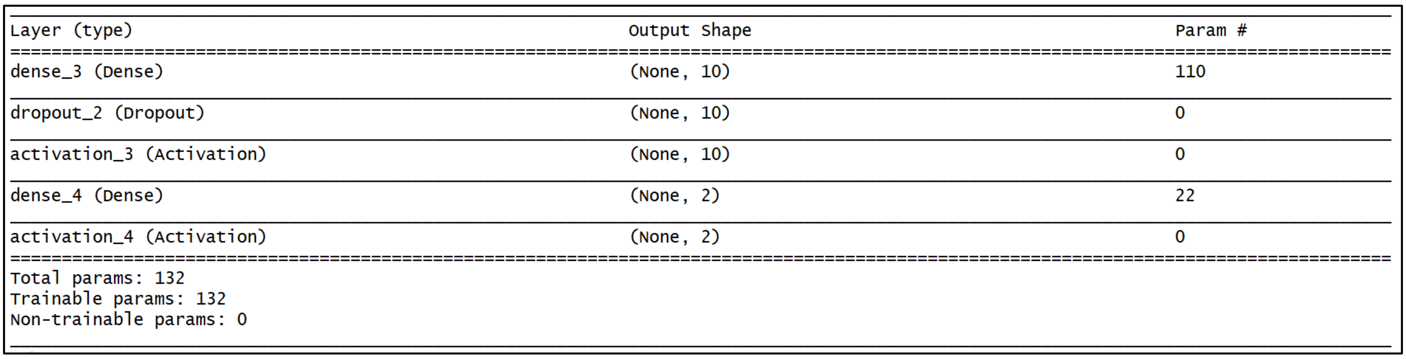


Figure 2 - Plots of cost function and accuracy for DASH prediction



```
Layer (type)                                Output Shape                           Param #
==============================================================================================
dense_3 (Dense)                             (None, 10)                             110

dropout_2 (Dropout)                         (None, 10)                             0

activation_3 (Activation)                   (None, 10)                             0

dense_4 (Dense)                             (None, 2)                              22

activation_4 (Activation)                   (None, 2)                              0
==============================================================================================
Total params: 132
Trainable params: 132
Non-trainable params: 0
```

Figure 3 - Summary of model

Performance = 0.57 for 100 iterations [1 hidden layer with 10 hidden nodes]

Trying by shuffling the input data + only taking the 52% confidence or above:

Performance = 0.60 for 100 iterations [1 hidden layer with 10 hidden nodes]
(216 predictions of 719 in test set)