HW-hyper-tuning

EE3 KUAN-YU-LIN b04901110

May 2018

1 Introduction

This is a report about hyper-parameter tuning in machine learning. The data set of this experiment is mnist and the illustration of the model is shown in figure 1. The hyper parameters we will tune include learning-rate, transfer function, batch size, epochs, optimizer, and weight initialization.

2 Method

Without specifying, the hyper parameter' default values are as follows: learning rate = 1e-4, transfer function = relu, batch size = 512, epochs = 40, optimizer = Adam, kernel initializer = random uniform. we will try to modify each value of the hyper parameter to find the best result. and the ready-to-be assigned value of each hyper-parameter is explicitly shown below.

learning rate	1e-2	1e-3	1e-4	1e-5	1e-6		
transfer function	selu	relu	tanh	sigmoid	linear		
batch size	20	30	50	100	200		
epochs	100	200	300	400	500		
optimizer	SGD	RMSprop	Adagrad	Adadelta	Adam	Adamax	Nada
initializer	lec uniform	he uniform	glor uniform	lec normal	he normal	glor normal	

3 Outcome

1. learning-rate - accuracy, shown in figure 2
2. transfer function - accuracy, 2
3. .batch size - accuracy, shown in figure 4
4. epochs - accuracy, shown in figure 5
5. optimizer - accuracy, shown in figure 6
6. weight initialization - accuracy, shown in figure 7

4 Discussion

1. According to figure 2, we can see once learning rate is too large, the harder for the model to converge and hence not able to perform as well as smaller one.
2. elu, selu, relu are the best performers in this test, due to their similarity.
3. figure 5 shows that batch size is not a decisive hyper parameters.
4. accuracy drop after epochs ¿ 400, due to over tuning
5. Except Adadeita, all perform similarly.
6. Except he uniform, all perform similarly.

5 Conclusion

the best resultant hyper parameters are respectively:

1. learning-rate = 0.0001
2. transfer function = relu
3. batch size = 100
4. epochs = 400
5. optimizer = Adam
6. weight initialization = random uniform

6 Best result

combine the best hyper parameters we can get 0.978 accuracy, and the plot of training history of the best hyper-parameters are provided at figure 8

```
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 160)               125600
_________________________________________________________________
dense_2 (Dense)              (None, 250)               40250
_________________________________________________________________
dense_3 (Dense)              (None, 100)               25100
_________________________________________________________________
dense_4 (Dense)              (None, 160)               16160
_________________________________________________________________
dense_5 (Dense)              (None, 10)                1610
=================================================================
Total params: 208,720
Trainable params: 208,720
Non-trainable params: 0
```

Figure 1: model structure

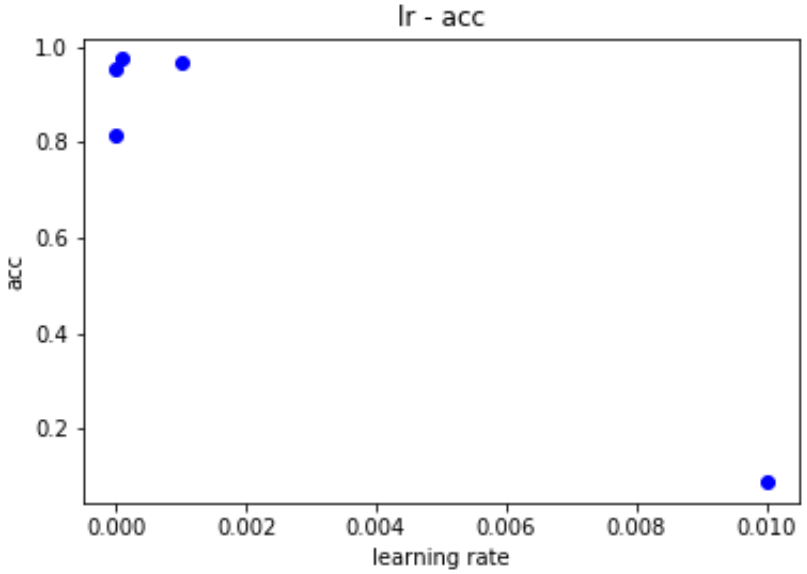

Figure 2: learning rate-accuracy

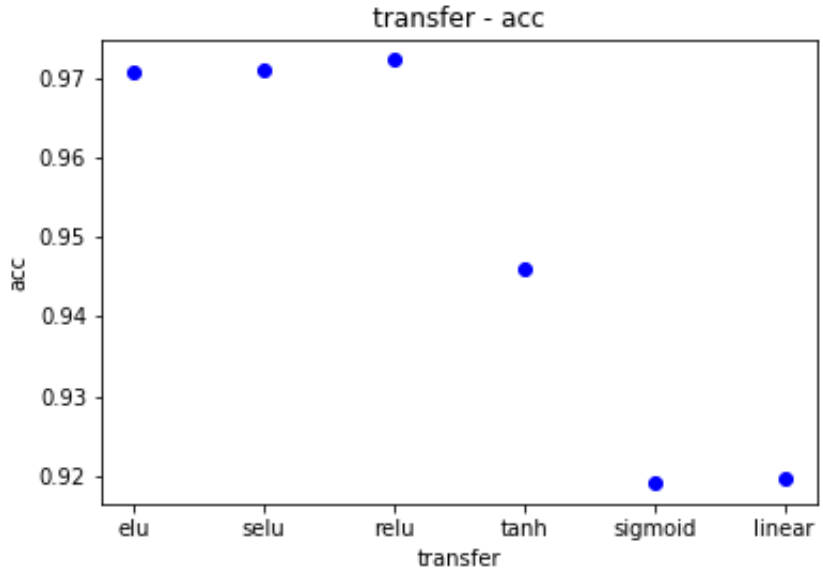

Figure 3: transfer function - accuracy

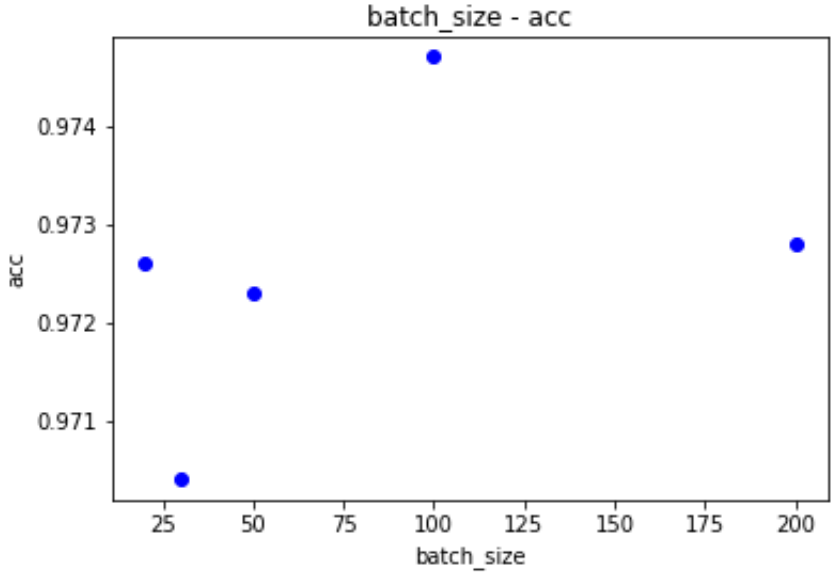

Figure 4: batch size - accuracy

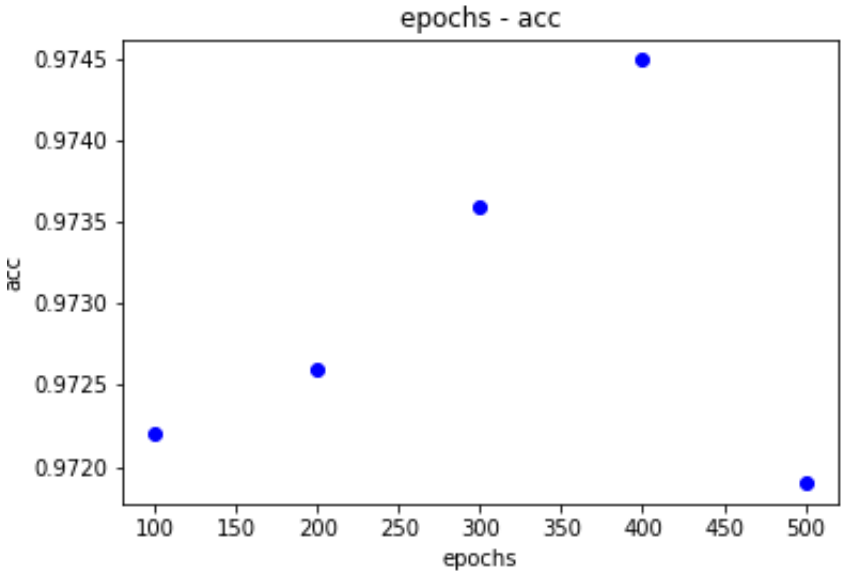

Figure 5: epochs - accuracy

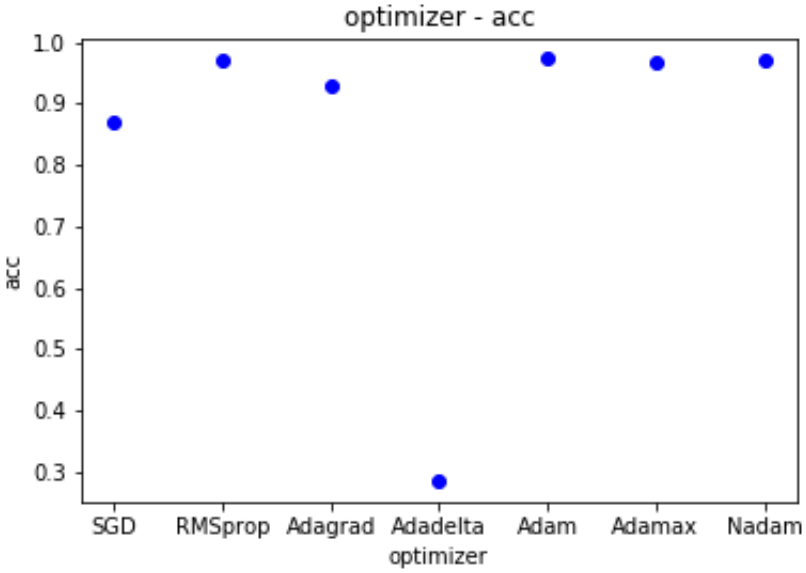

Figure 6: optimizer - accuracy

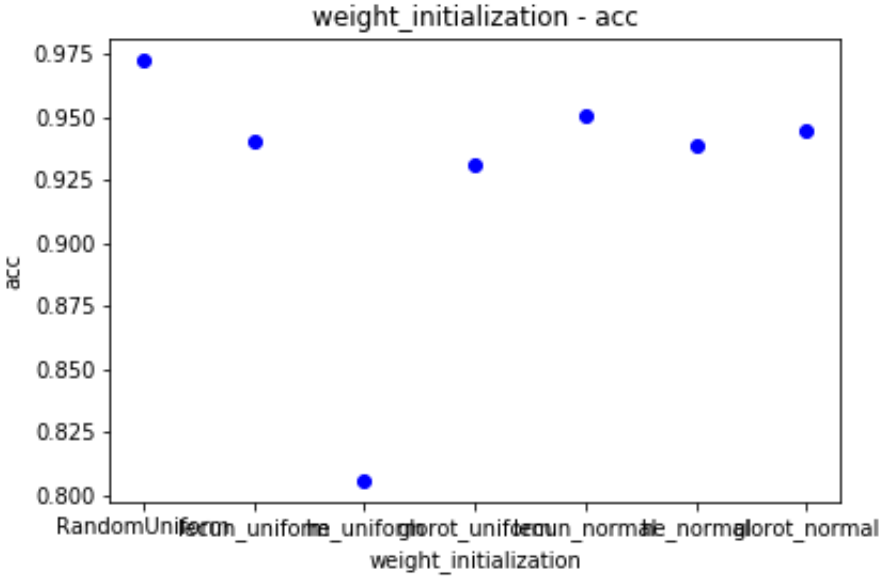

Figure 7: weight initialization - accuracy

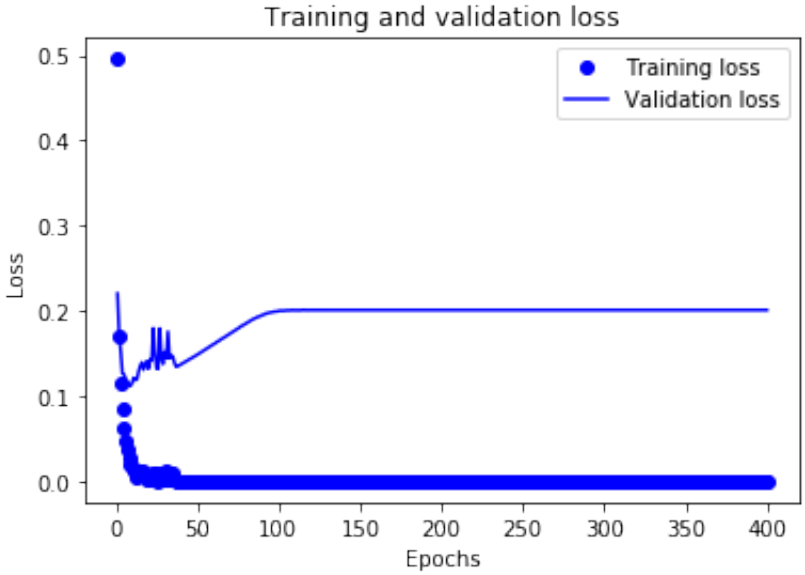

Figure 8: training history of the best hyper parameters