ITE4053 Deep Learning course

Practice 4

2016026080 Dajin Han

Learning_rate = 1.0,

	BinaryCrossentropy	MeanSquaredError
Acc_train	0.984600	0.985200
Acc_test	0.987000	0.982000

	SGD	RMSprop	Adam
Acc_train	0.983900	0.985300	0.940100
Acc_test	0.988000	0.973000	0.931000
Time_train	5.122176	6.413932	5.869797
Time_test	0.249455	0.257780	0.249931
Loss type	Binary CE	Binary CE	Binary CE

	Python Result (local)	CPU (local)	GPU (gpu server)
Acc_train	0.984600	0.984700	0.985400
Acc_test	9.770000	0.986000	0.987000
Time_train	0.8349459	4.460400	5.611204
Time_test	0.0001789	0.341082	0.180154
Loss type	CE	CE	CE
Optimizer type	SGD	SGD	SGD

	Mini-batch=1	Mini-batch=32	Mini-batch=128	Mini-batch=1000
Acc_train	0.989000	0.994600	0.995600	0.985800
Acc_test	0.980000	0.993000	0.992000	0.987000
Time_train	1813.202614	80.576750	21.302118	2.872409
Time_test	0.118682	0.113498	0.112499	0.115892
Loss type	CE	CE	CE	CE
Optimizer type	SGD	SGD	SGD	SGD

- **Best Optimizer :** When the value of learning_rate is 1.0, SGD optimizer usually performed best. But when I changed the learning_rate smaller, adam optimizer performed best. Adam optimizer is the best option for slow access to the most well trained model.
- **Mini-batch :** In my case, the mini-batch performed better than the full batch. Maybe mini batch can identify small commonalities of small dataset, and train model with them and this makes model specific, while full batch only train model roughly