딥러닝 및 응용					
학번	2018023390	**이름**	이서연	**제출일**	2021.05.29

1. Table 1

	BinaryCrossentropy	MeanSquaredError
Accuracy (with train set)	0.9998	0.9991
Accuracy (with test set)	1	1
Train time(s)	742.45	682.15

2. Table 2

	SGD result	RMSProp	Adam
Accuracy (with train set)	0.9998	0.9998	0.9996
Accuracy (with test set)	1	1	1
Train time	742.45	802.29	742.25

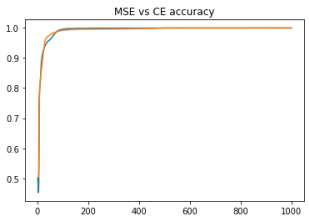
3. Table 3

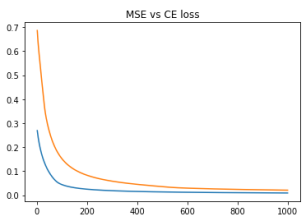
	mini batch = 4	mini batch = 32	mini batch = 128
Accuracy (with train set)	0.9999	0.9988	0.9986
Accuracy (with test set)	1	1	1
Train time	2062.17 s	742.45	142.2 s

4. Discussion

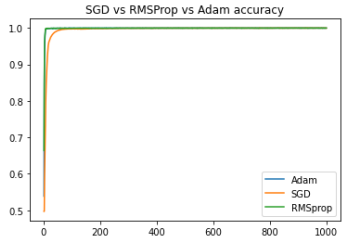
사실 모든 부분에서 Total Accuracy 의 큰 차이가 있지는 않았다. 문제가 어렵지 않고, 많이 반복을 했기 때문인 것으로 보인다.

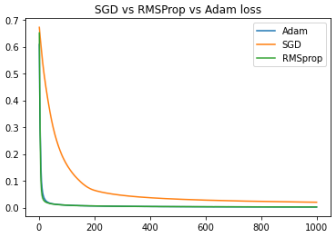
Table 1 번에서는 Binary cross entropy 와 MSE 를 비교했는데, Binary cross entropy 가 살짝 더 빠르게 끝났는데, MSE 가 더 계산적으로 덜 또는 더 expensive 해서 그런 것은 아닌 것 같고, training 자체가 모종의 이유로 늦게 시작한 것 같다. 1 epoch 당 학습하는 시간이 정확히 똑같이 나왔기 때문이다. Accuracy 와 Loss 가 epoch 별로 떨어지는 것을 확인했는데 어떻게 보면 MSE 가 좀 더 빠르게 떨어지는 것 처럼 보일 수도 있지만 사실 MSE 는 제곱값이라는 점을 감안하면 사실 당연한 것으로 보인다. accuracy 가 수렴하는 시점은 비슷하다.





Optimizer 에 따라서도 Epoch 에 따른 Loss 를 비교해보았는데 SGD 는 천천히 떨어지며 수렴하는 반면 Adam 과 RMSprop 은 급격하게 떨어져서 한번에 수렴하는 것을 볼 수 있다. 1000 번을 epoch 을 돌고도 SGD 는 Adam 과 RMSProp 보다 loss 가 떨어지지 못했다.





같은 minibatch size(=32)를 이용했음에도, SGD 는 mini batch 마다 loss 의 변동이 크게 나타났는데, RMSprop 과 Adam 은 처음에는 큰폭으로 움직이다가 수렴하고 나서부터는 loss 가 움직이는 폭이 줄어든다. 이는 RMSprop 과 Adam 이 작동하는 방식과 연관이 있는 것으로 생각된다. SGD 의 경우 learning rate 를 한번 정하면 변하지 않는데, RMSprop 이나 Adam 은 minibatch 마다 dW, db 를 업데이트하는 정도가 학습을 할수록 바뀌게 된다. 그래서 optimal 에 수렴한 이후로는 거의 변하지 않는데 SGD 는 한번에 수렴하지도 못하고, 큰 폭으로 움직이게 된다.

batch 마다 loss 를 얻는 부분은 이곳 (https://stackoverflow.com/questions/47479557/keras-record-loss-and-accuracy-of-train-and-test-for-each-batch) 에서 참고했다.

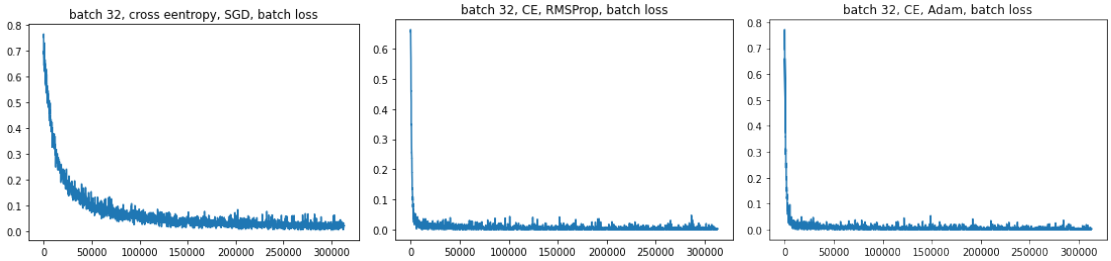
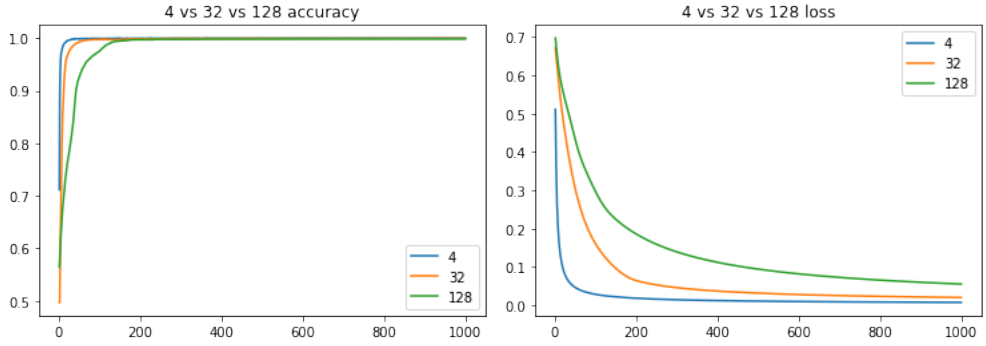
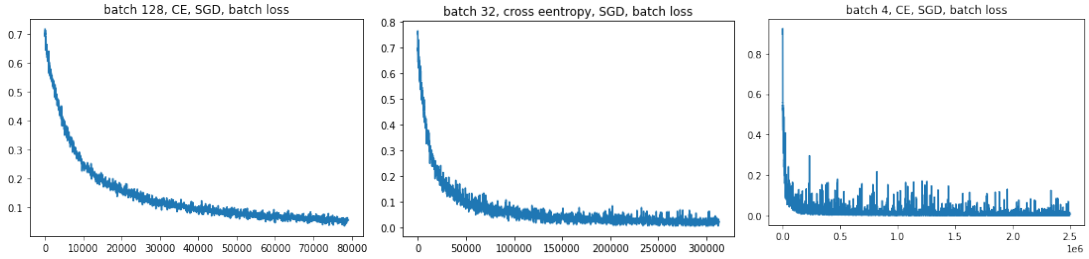


Table 3 에서 비교한 mini batch size 별로도 비교를 진행했는데, 당연하게도 4 에서 가장 학습에 오랜 시간이 소요되었고, 128 이 가장 먼저 끝났다. accuracy, loss 모두 batch size = 4 에서 가장 좋았다. 하지만 실제로 사용한다면, 아무리 성능이 좋아도시간이 너무 오래걸려서 저 정도로 작은 batch 로는 학습이 힘들 것으로 생각된다. 더 feature 가 많고, 복잡한 네트워크였다면 이것보다 훨씬 오래 걸렸을 것이다.





batch 별 loss 를 가지고 그래프를 그려봤을 때, batch size 가 적으니 변동폭이 굉장히 컸다. 사실 batch gradient descent 는 이러한 변동 폭이 나타나지 않으니까, 실험을 해보기 전에는 batch size 가 작으면 작을수록 더 변동폭이 작게 나타나지 않을까 생각했는데, 생각해보니 batch 가 작아서 한번의 batch 를 돌 때 오히려 새로운 데이터를 많이 접하지 못하니까 변동폭이 크게 나타날 수도 있을 것 같다.