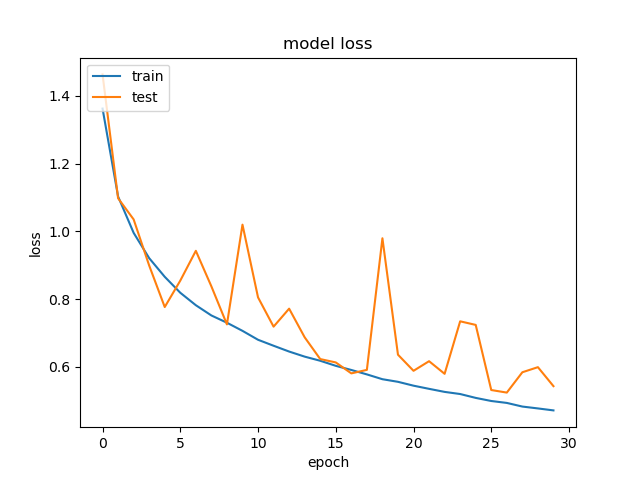
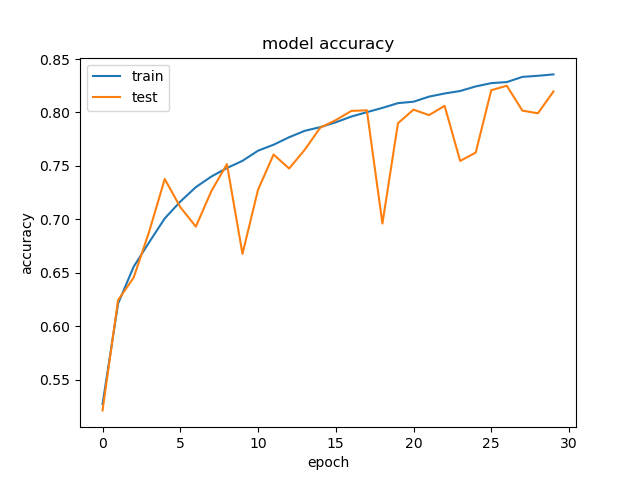
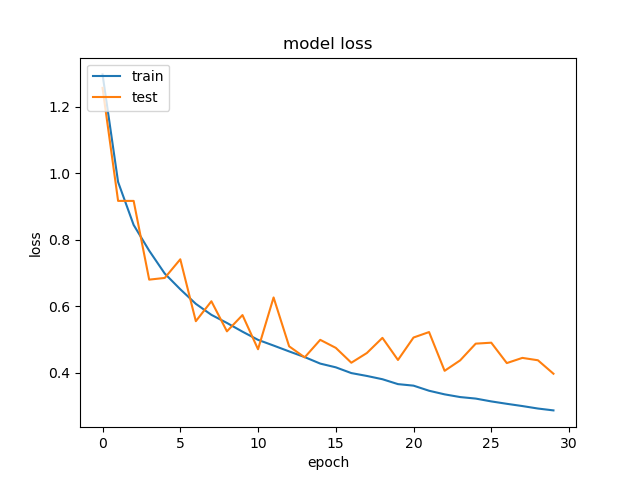
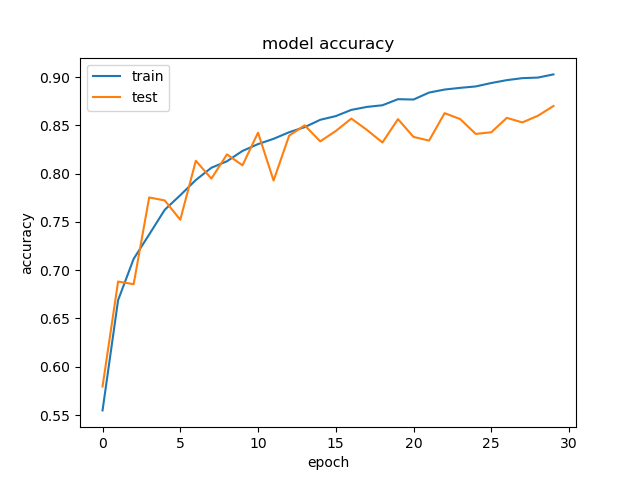
**모든 Epoch는 30, Batchsize는 32라고 가정.**



**Bn\_conv 2와 4를 제거**하여 실험을 진행했다. 즉, 각 레이어 단계에서 2번씩 훈련하는 것을 1번으로 줄였는데 결과는 다음과 같았다.

226s 145ms/step - loss: 0.4720 - acc: 0.8356 - val\_loss: 0.5434 - val\_acc: 0.8196

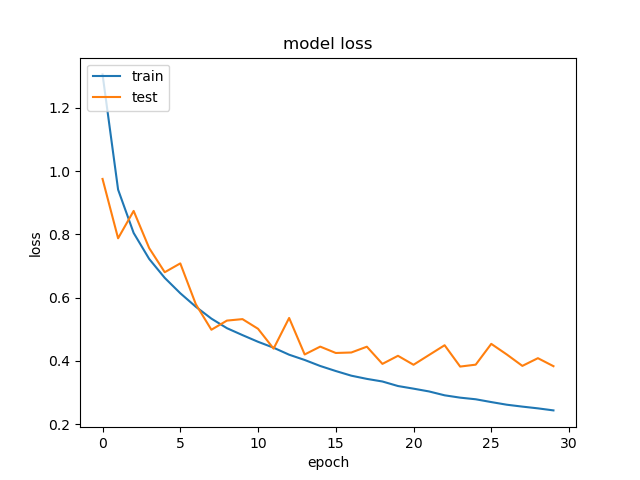
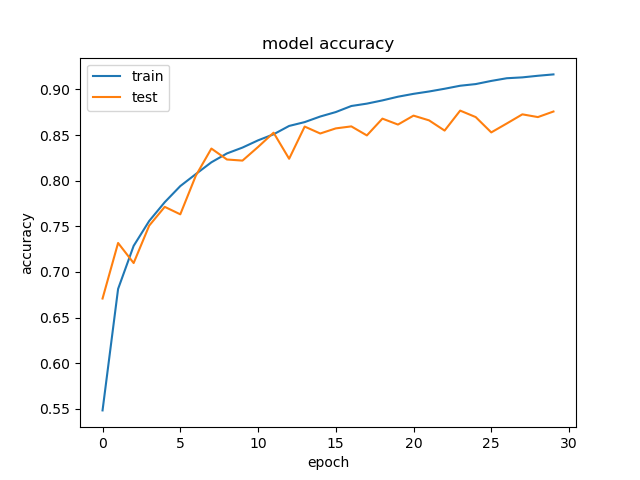
**훈련을 1번씩 줄이니 기존의 실험보다 acc가 낮은 결과가** 나왔다.



**다음은 Bn\_conv 3,6을 추가시켜** 실험을 진행했다. 즉, 각 레이어 단계에서 2번씩 훈련하는 것을 3번으로 늘렸다. 결과는 다음과 같았다.

1562/1562 [==============================] - 713s 456ms/step - loss: 0.2861 - acc: 0.9028 - val\_loss: 0.3966 - val\_acc: 0.8700

**훈련시간이 3배나 늘었지만, 그만큼 acc도 늘었다**. 과적합이 실험 1에 비해 더 늘어났다?

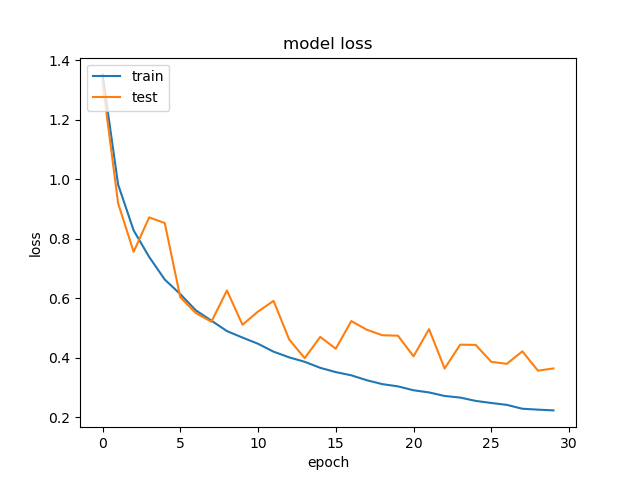
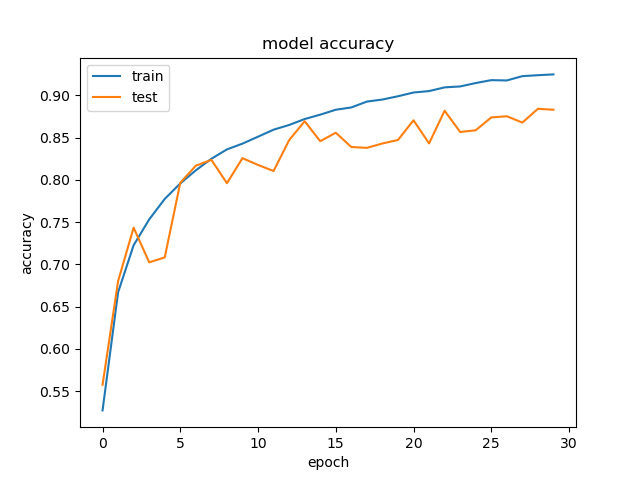


**다음은 각 레이어 단계가 2번이었는데, 3번으로 (32, 64 레이어 + 128 레이어) 늘려서 진행했다**.

각 레이어 단계는 2번씩 훈련했다.

1562/1562 [==============================] - 23s 15ms/step - loss: 0.2435 - accuracy: 0.9165 - val\_loss: 0.3832 - val\_accuracy: 0.8758

전 실험에서 너무 많은 시간이 걸려, CUDA를 통해 돌리니 시간이 매우 단축됐다. **Acc가 향상됐다.**



**마지막은 각 레이어 단계를 4번으로 늘렸다. (32, 64 레이어 + 128, 256 레이어)**

1562/1562 [==============================] - 24s 16ms/step - loss: 0.2222 - accuracy: 0.9247 - val\_loss: 0.3634 - val\_accuracy: 0.8829

CUDA로 인해 시간의 차이가 별로 안난다.(안중요한듯) **acc 역시도 향상됐다. 과적합이 벌어지긴 하나, val\_acc(검증 데이터\_test)가 객관적인 수치가 높아지고 있기 때문에 많이 향상됐다고 볼 수 있다.**