# SE395 Programming Assigment2 Report

201811118 이 구

***1. For all networks (1) 2-layer CNN, (2) 3-layer CNN, (3) 2-layer CNN using DL-framework, (4) 3-layer CNN using DL-framework and (5) 3-layer NN with ReLU (PA1), show the results and compare***

DL framwork를 사용하지 않은 모델의 경우 MNIST train dataset 60000장을 1 epoch 학습하였다. 학습 중 validation에 걸리는 시간을 줄이기 위하여 2-layer 모델의 경우 MNIST test dataset 중 랜덤한 2000장만 이용하였고, 3-layer 모델의 경우 랜덤한 500장만 이용하였다. DL framework를 사용한 모델의 경우 MNIST train dataset 60000장을 5 epoch 씩 학습하였고, validation에는 MNIST train dataset 10000장을 모두 사용하였다.

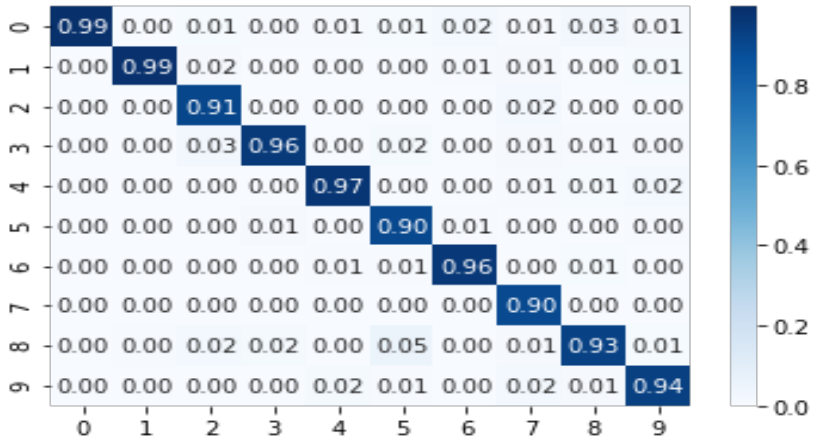
3-layer 모델의 경우, 각 layer의 forward, backward를 구현하였고, model 에서의 forward, backward 및 train이 정상적으로 동작하였지만, accuracy가 어느 순간부터 증가하지 않고, local minima에 빠진 듯한 결과를 보였다. 적절한 learning rate를 찾아 문제를 해결하고자 하였으나, due date가 촉박하여 model의 train accuracy가 약 27% 인 상태에서 학습을 중단시킨 결과를 사용하였다.

(1) 2-layer CNN, (2) 3-layer CNN, (3) 2-layer CNN using DL-framework, (4) 3-layer CNN using DL-framework 와 (5) 3-layer NN with ReLU (PA1)의 성능을 비교하였을 때, 학습이 충분히 진행되지 않은 (2) 모델을 제외한 모든 CNN model의 성능이 NN model보다 높았다. 이는, CNN이 단순히 이미지의 각각의 pixel 값들을 이용하여 예측하는 NN과 다르게, filter를 사용하여 이미지의 각 pixel의 상하좌우 context 정보들을 예측에 사용하기 때문일 것이다.

각 모델의 결과는 다음 페이지부터 첨부하였다. 이때, 각 모델의 추론 결과는 MNIST test dataset 10000장을 모두 사용하여 얻은 결과이다.

### 1) 2-layer CNN

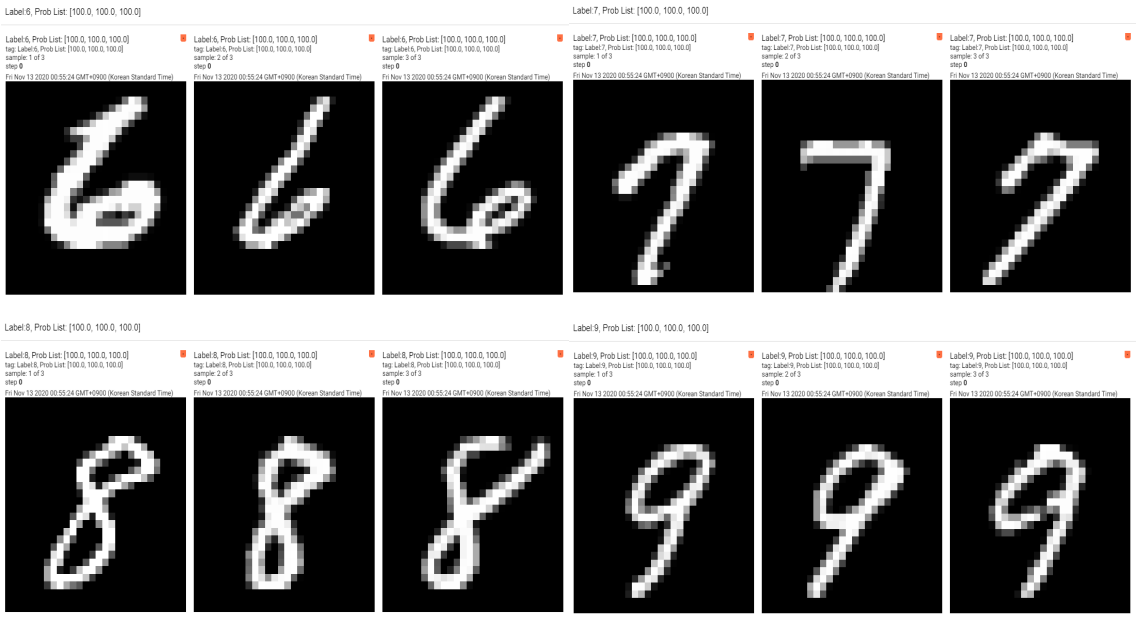
(a) 10x10 Confusion Matrix,




<confusion matrix: 2-layer CNN model>

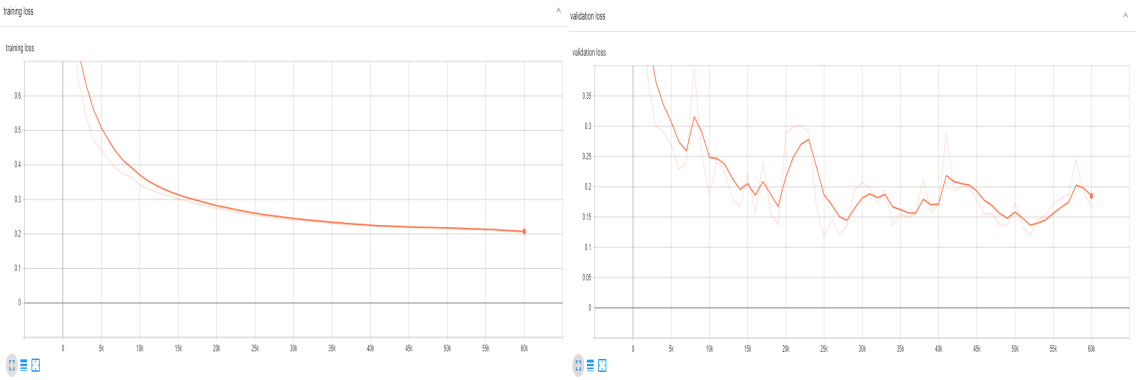
(b) Top 3 score images (all classes), using Tensorboard

구현한 2-layer CNN 모델의 결과 중, 각 class 별 top 3 score image를 Tensorboard 상에서 확인하였다. 각 이미지의 태그는 해당 이미지의 label과 세 이미지의 결과 값 리스트로 하였다. 앞에서부터 차례로 첫 번째, 두 번째, 세 번째 이미지의 결과 값이다.

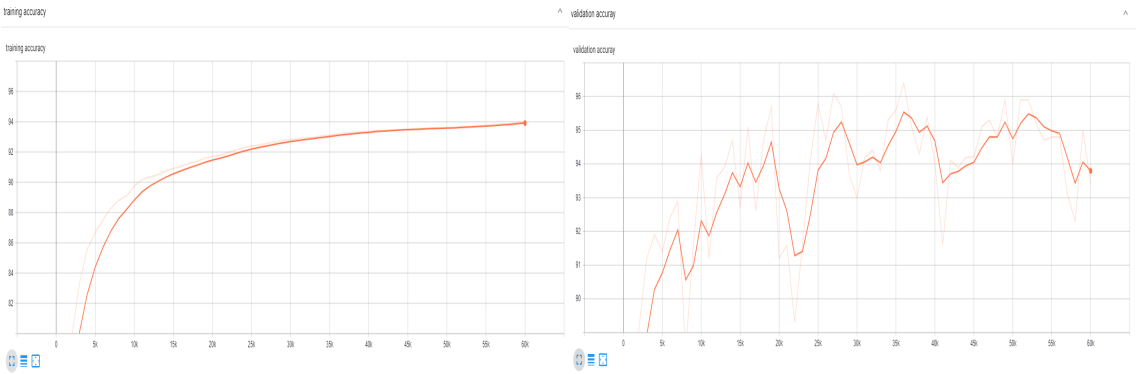



<top 3 score images: 2-layer CNN model>

(c) Training Loss graph using Tensorboard



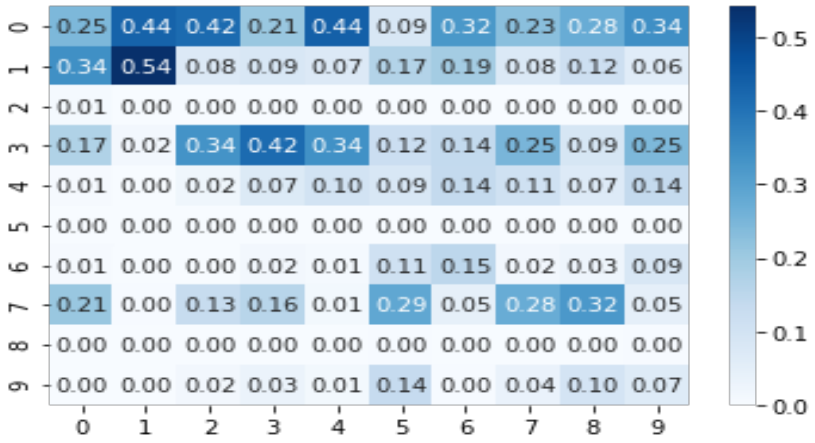

<2-layer CNN model training loss graph > <2-layer CNN model valid loss graph>




<2-layer CNN model train acc graph> <2-layer CNN model valid acc graph >

## 2) 3-layer CNN

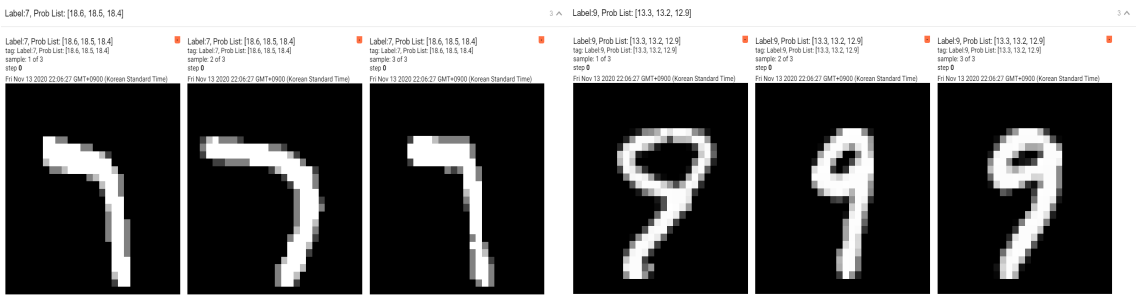
(a) 10x10 Confusion Matrix




<confusion matrix: 3-layer CNN model>

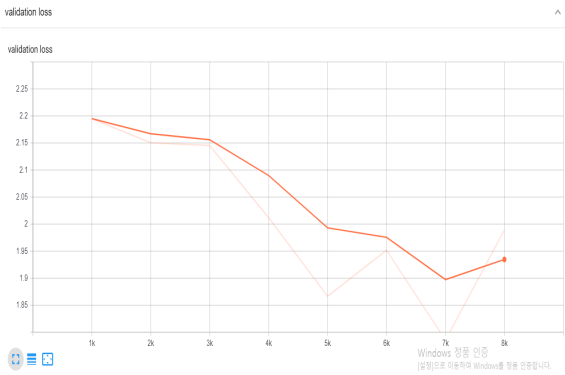
(b) Top 3 score images (all classes), using Tensorboard

구현한 3-layer CNN 모델의 결과 중, 각 class 별 top 3 score image를 Tensorboard 상에서 확인하였다. 각 이미지의 태그는 해당 이미지의 label과 세 이미지의 결과 값 리스트로 하였다. 앞에서부터 차례로 첫 번째, 두 번째, 세 번째 이미지의 결과 값이다. 이 중, 2와 8의 경우 옳게 예측된 이미지가 하나도 존재하지 않았고, 5의 경우 한 장의 이미지만이 옳게 예측되었다.

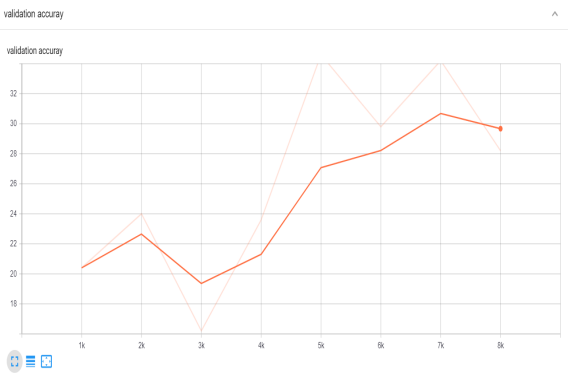



<top 3 score images: 3-layer CNN model>

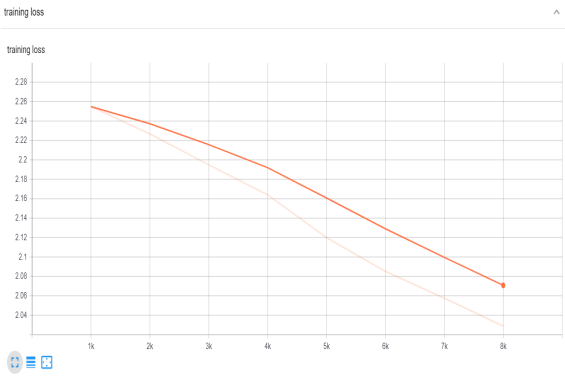
(c) Training Loss graph using Tensorboard



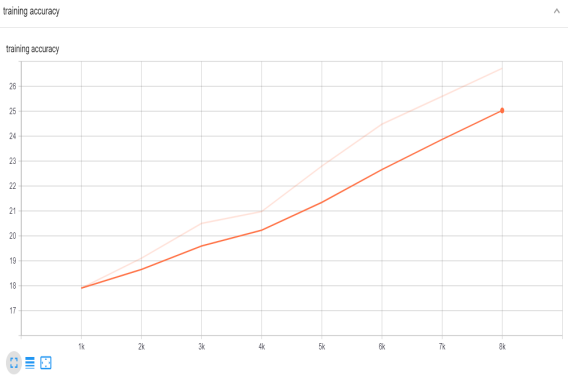

<3-layer CNN model valid loss graph>




<3-layer CNN model valid acc graph>



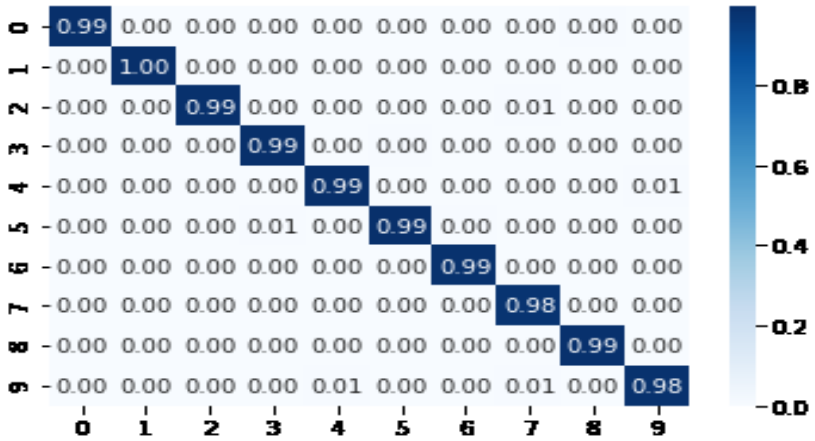

<3-layer CNN model train loss graph >




<3-layer CNN model train acc graph>

### 3) 2-layer CNN using DL-framework

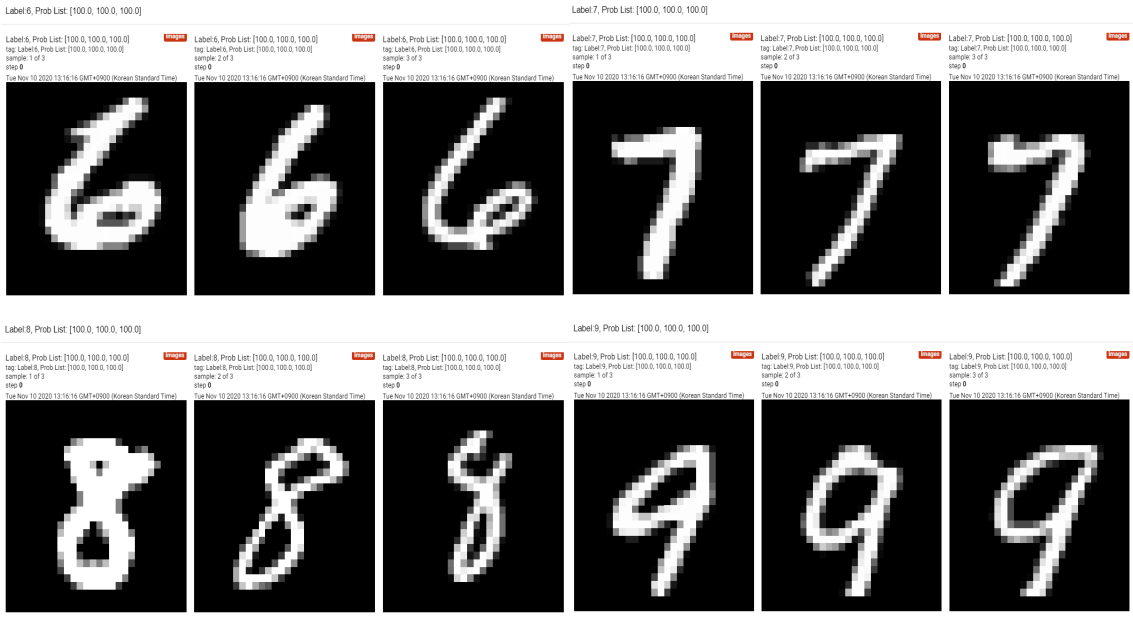
(a) 10x10 Confusion Matrix,




<confusion matrix: 2-layer CNN model with DL framework>

(b) Top 3 score images (all classes), using Tensorboard

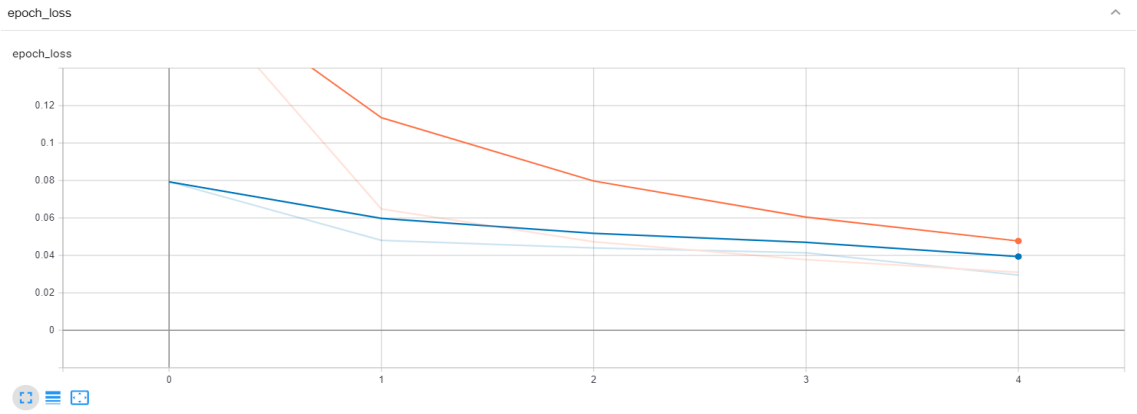
Keras를 이용하여 구현한 2-layer CNN 모델의 결과 중, 각 class 별 top 3 score image를 Tensorboard 상에서 확인하였다. 각 이미지의 태그는 해당 이미지의 label과 세 이미지의 결과 값 리스트로 하였다. 앞에서부터 차례로 첫 번째, 두 번째, 세 번째 이미지의 결과 값이다.



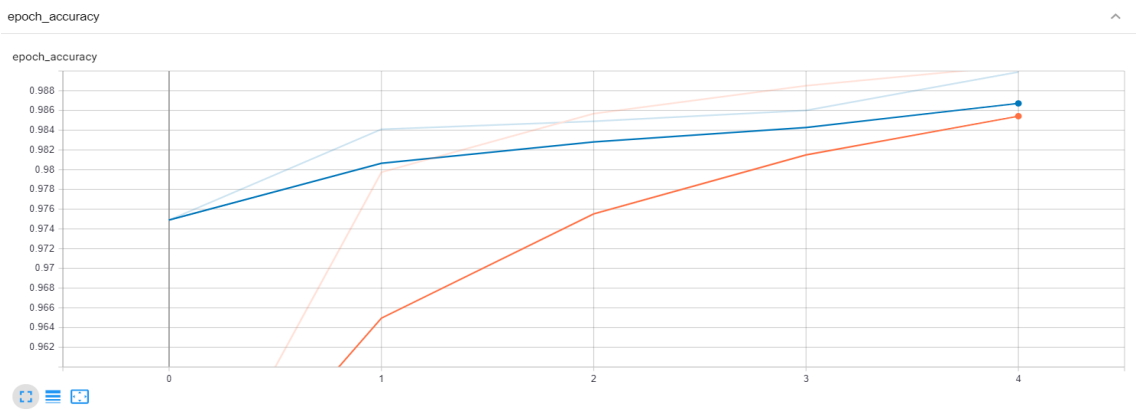

<top 3 score images: 2-layer CNN model with DL framework>

(c) Training Loss graph using Tensorboard

Tensorboard 상에서 확인한 2-layer CNN 모델(with DL framework)의 train loss, val loss, train accuracy, val accuracy는 아래와 같다.



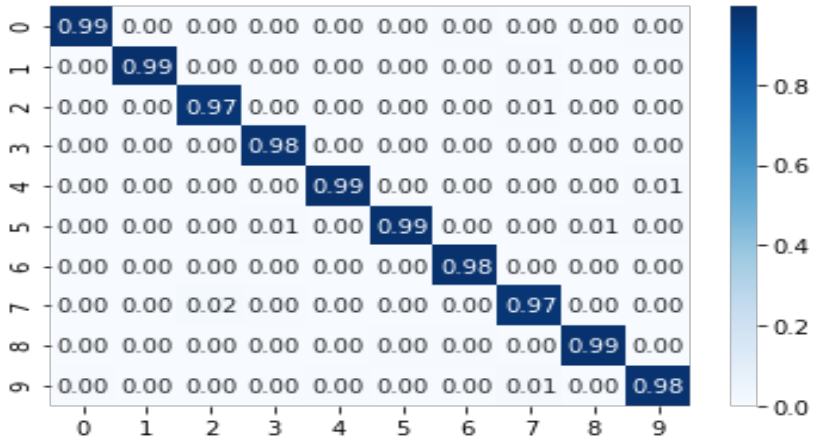

<2-layer CNN model with DL framework training, valid loss graph on Tensorboard>




<2-layer CNN model with DL framework training, valid acc graph on Tensorboard>

4) 3-layer CNN using DL-framework

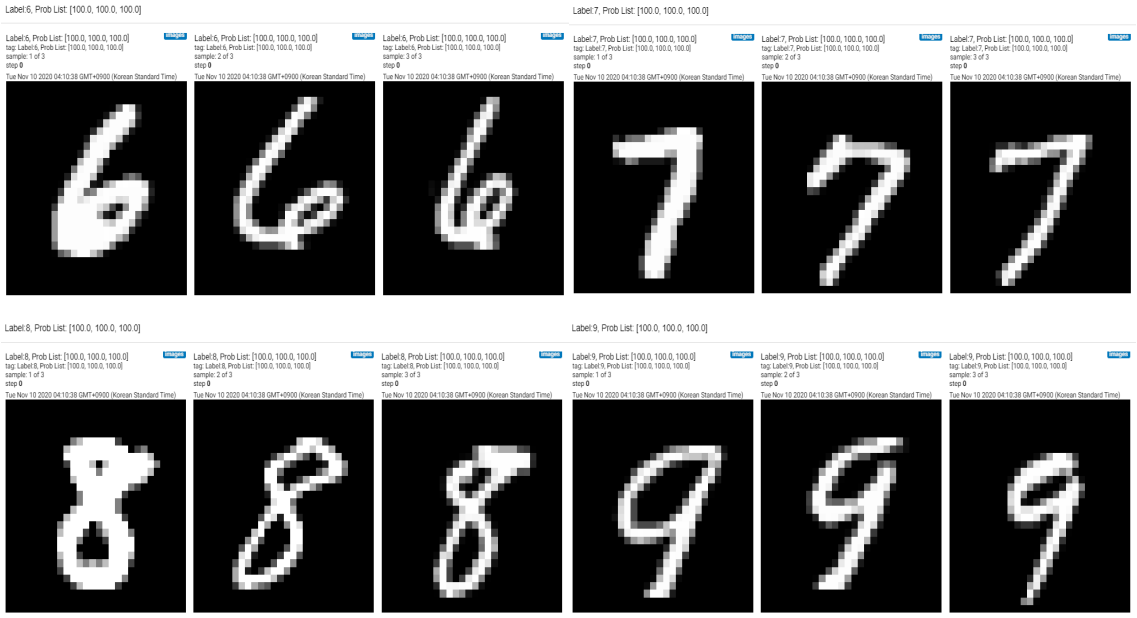
(a) 10x10 Confusion Matrix,




<confusion matrix: 3-layer CNN model with DL framework>

(b) Top 3 score images (all classes), using Tensorboard

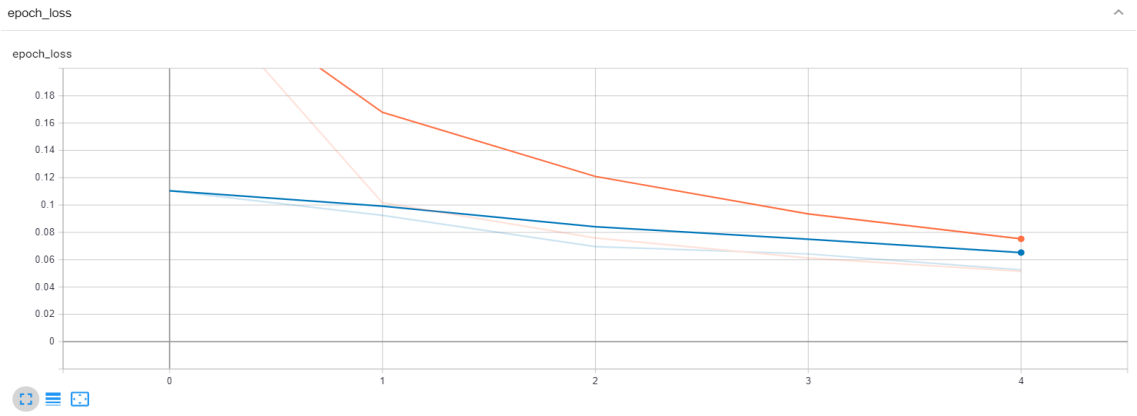
Keras를 이용하여 구현한 3-layer CNN 모델의 결과 중, 각 class 별 top 3 score image를 Tensorboard 상에서 확인하였다. 각 이미지의 태그는 해당 이미지의 label과 세 이미지의 결과 값 리스트로 하였다. 앞에서부터 차례로 첫 번째, 두 번째, 세 번째 이미지의 결과 값이다.



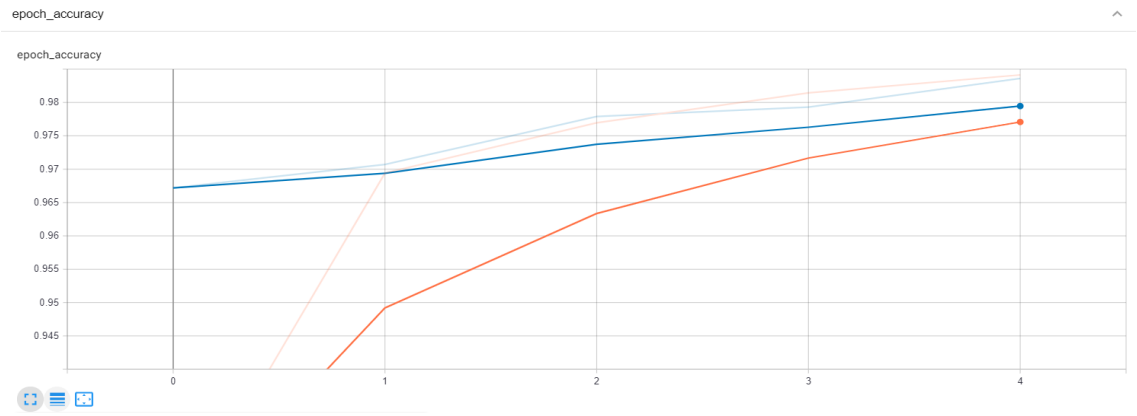

<top 3 score images: 3-layer CNN model with DL framework>

(c) Training Loss graph using Tensorboard

Tensorboard 상에서 확인한 3-layer CNN 모델(with DL framework)의 train loss, val loss, train accuracy, val accuracy graph는 아래와 같다.



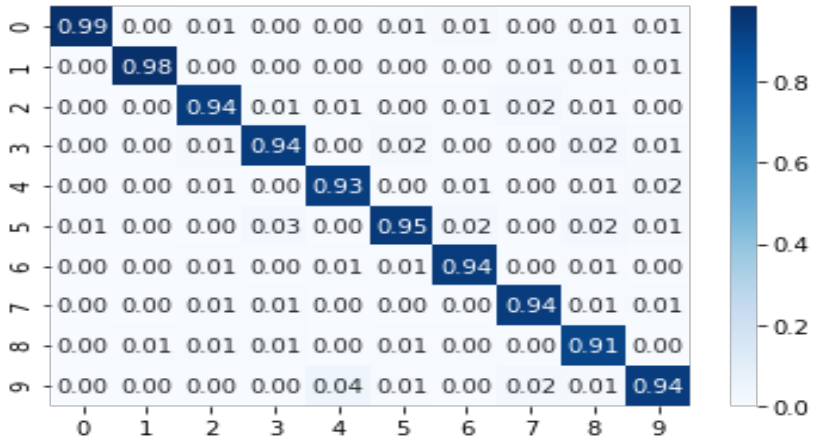

<3-layer CNN model with DL framework training, valid loss graph on Tensorboard>




<3-layer CNN model with DL framework training, valid acc graph on Tensorboard>

5) 3-layer NN with ReLU

(a) 10x10 Confusion Matrix,




<confusion matrix: 3-layer nn model with DL framework>

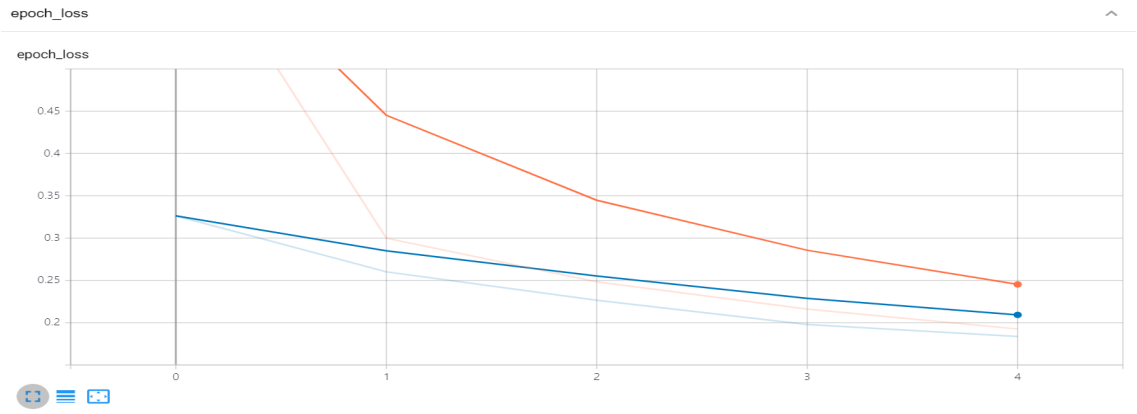
(b) Top 3 score images (all classes), using Tensorboard

Keras를 이용하여 구현한 3-layer NN 모델의 결과 중, 각 class 별 top 3 score image를 Tensorboard 상에서 확인하였다. 각 이미지의 태그는 해당 이미지의 label과 세 이미지의 결과 값 리스트로 하였다. 앞에서부터 차례로 첫 번째, 두 번째, 세 번째 이미지의 결과 값이다.

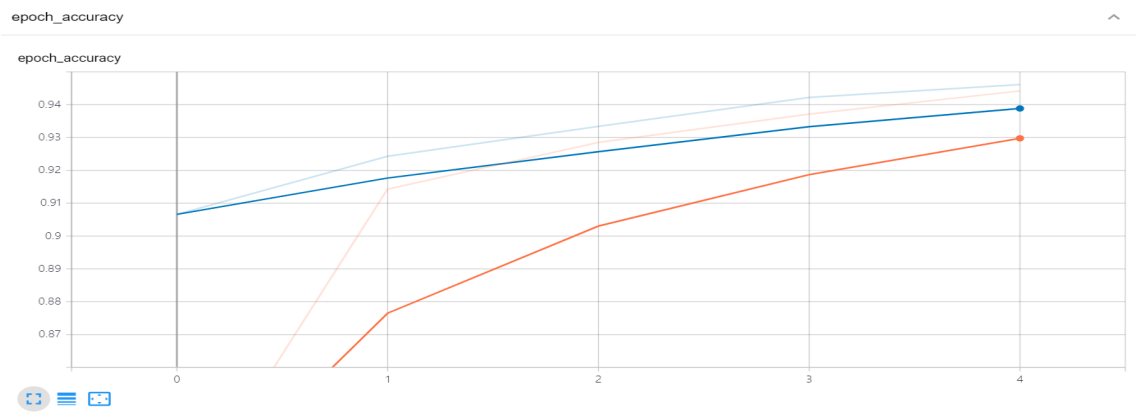

<top 3 score images: 2-layer nn model with framework>

(c) Training Loss graph using Tensorboard

Tensorboard 상에서 확인한 3-layer NN 모델(with DL framework)의 train loss, val loss, train accuracy, val accuracy graph는 아래와 같다.




<3-layer NN model with DL framework training, valid loss graph on Tensorboard>

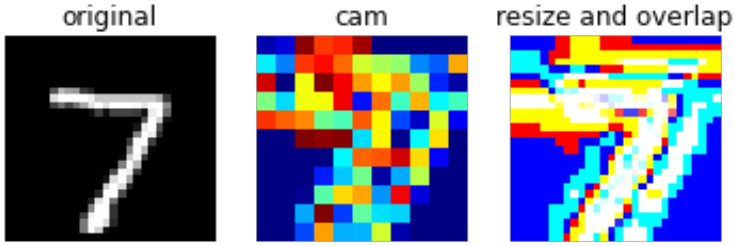



<3-layer CNN model with DL framework training, valid acc graph on Tensorboard>

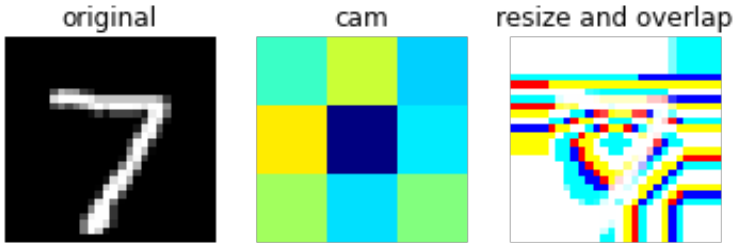
*2. (Optional – extra 30%) Visualize four Class Activation Map (CAM) from above four CNNs (1)-(4) with the same sample.*

아래의 각 모델의 Class Activation Map은 MNIST test dataset의 첫 번째 이미지를 사용하여 얻은 결과이다. 2-layer CNN 모델의 경우 마지막 Convolution Layer의 output의 shape이 (num_channel, 11, 11) 이고, 3-layer CNN 모델의 경우 마지막 Convolution Layer의 output의 shape이 (num_channel, 3, 3) 이다.
아래의 이미지들은 순서대로 각 모델의 Class Activation Map을 얻기 위해 사용한 original 이미지, Class Activation Map, Class Activation Map을 resize한 후 원본 이미지와 overlap 한 결과이다.
2-layer CNN 모델들의 경우 original 이미지에서, 7의 윤곽을 따라 activation 되어있는 것을 확인할 수 있었지만, 3-layer CNN 모델들의 경우 CAM의 size가 (3, 3)이기에 유의미한 의미를 찾기 힘들었다.

1) 2-layer CNN




2) 3-layer CNN




3) 2-layer CNN with DL framework


4) 3-layer CNN with DL framework

**reference**

아래 사이트들을 참고하였습니다.

https://tutorials.pytorch.kr/intermediate/tensorboard_tutorial.html
https://www.tensorflow.org/tensorboard/image_summaries
https://github.com/nvs-abhilash/CAM-keras/blob/master/cam_keras.py
https://github.com/tkwoo/ClassActivationMap_Keras
https://github.com/vzhou842/cnn-from-scratch
https://github.com/AlessandroSaviolo/CNN-from-Scratch