Lab 3 and Lab 4

MNIST (http://yann.lecun.com/exdb/mnist/), Modified subset of NIST (National Institute of Standards and Technology)

MNIST란 본래 미국에서 우편번호(ZIP code)를 분류할 때 기계에게 분류를 맡기려는 시도에서 비롯되었다. MNIST는 Modified subset of NIST (National Institute of Standards and Technology) 약자이다. NIST에서 수집된 DB를 약간 수정한 DB를 의미하고, NISTS의 직원 및 고등학생으로 구성된 250명에게서 수집된 손으로 쓴 숫자 이미지 집합이다.

MNIST는 간단한 컴퓨터 비전 dataset이다. 손으로 필기한 숫자 각각의 이미지가 정수로 표시되는 데이터베이스인데, 머신러닝 중 지도학습 (supervised learning) 알고리즘의 성능을 벤치마킹하는 데 사용하고 있다. 딥러닝은 MNIST에서 매우 잘 수행되며, 99.7 % 이상의 정확도를 달성하였다.

MNIST 데이터 세트에는 6만건의 사례로 구성된 교육/학습(training) 세트와 1만건의 사례로 구성된 테스트(test) 세트가 포함되어 있다. 트레이닝 세트는 올바른 레이블인 정수를 예측하는 알고리즘을 가르치기 위해 사용되며 테스트 세트는 트레이닝 된 네트워크가 얼마나 정확하게 추측 할 수 있는지 확인하는 데 사용한다.

다음 step에 따라 인공신경망을 설계하고 결과를 비교하시오.

Step 1:
random으로 Initialization(초기화)하고 1 hidden layer with SoftMax로 구현하고 결과를 비교한다.

Step 2:

random으로 Initialization하고 2 hidden layers with ReLu로 구현하고 결과를 비교한다.

Step 3:

초기화를 Xavier 또는 He가 제시한 method로 하고 2 hidden layers with ReLu로 구현하고 결과를 비교한다.

Step 4:

초기화를 Xavier 또는 He가 제시한 method로 하고 4 hidden layers with ReLu로 구현하고 결과를 비교한다.

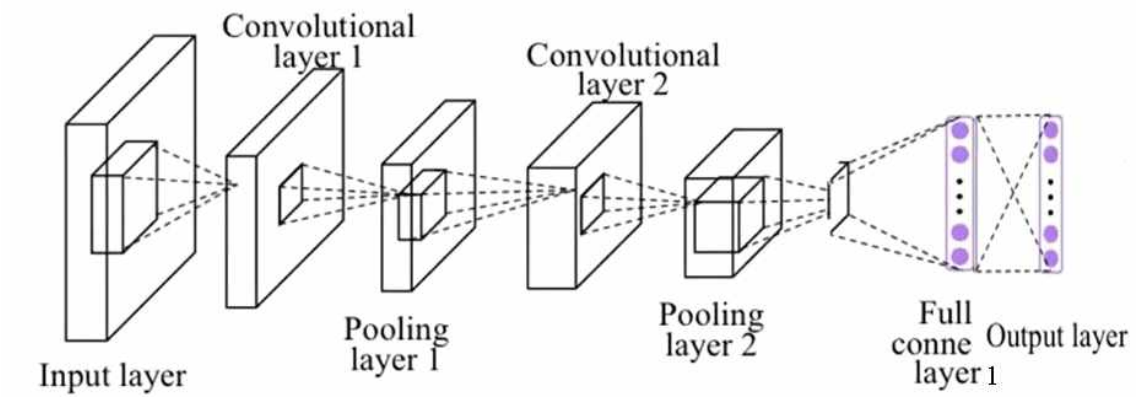
Step 5:

Step 4에서 accuracy에 문제점이 있다면 이를 심층 분석하고, 이를 해결하기 위한 방법인 Dropout을 사용한다.

또한 초기화를 Xavier 또는 He가 제시한 method로 하고 4 hidden layers with ReLu로 구현하고 결과를 비교한다.

Step 6:

2개 CNN layers without dropout - FC with 1 hidden layer



input [28x28x1]

Conv layer 1:

Conv1[28x28x32] : 32 3x3 kernels/filters/weights at stride 1, padding ='same'(입력과 출력 사이즈를 같게)
ReLu 통과 [28x28x32]
Max_pool1[14x14x32] : 1 2x2 kernels/filters/weights at stride 2, padding ='same'(입력과 출력 사이즈를 같게)

Conv layer 2:

input [14x14x32]
Conv2[14x14x64] : 64 3x3 kernels/filters/weights at stride 1, padding ='same'(입력과 출력 사이즈를 같게)
ReLu 통과 [14x14x64]
Max_pool2[7x7x64] : 1 2x2 kernels/filters/weights at stride 2, padding ='same'(입력과 출력 사이즈를 같게)

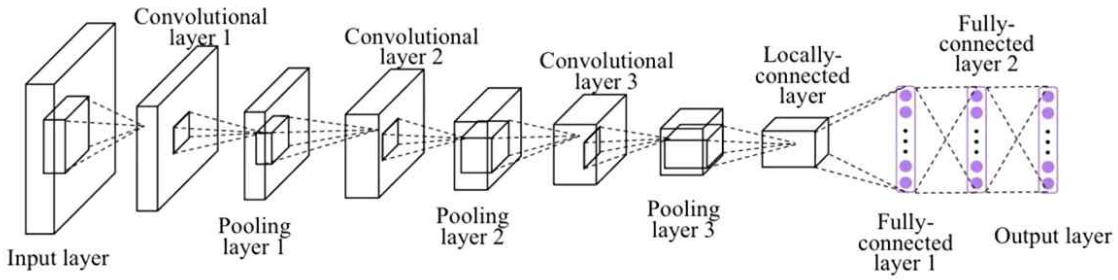
FC layer:

input [7x7x64], weight [7x7x64], b [10]

y = wx + b
cost = cross-entropy with softmax
Optimize

Step 7:

3개 CNN layers with dropout - FC with 2 hidden layers



input [28x28x1]

Conv layer 1:

Conv1[28x28x32] : 32 3x3 kernels/filters/weights at stride 1, padding ='same'(입력과 출력 사이즈를 같게)
ReLu 통과 [28x28x32]
Max_pool1[14x14x32] : 1 2x2 kernels/filters/weights at stride 2, padding ='same'(입력과 출력 사이즈를 같게)
dropout [14x14x32]

Conv layer 2:

input [14x14x32]
Conv2[14x14x64] : 64 3x3 kernels/filters/weights at stride 1, padding ='same'(입력과 출력 사이즈를 같게)
ReLu 통과 [14x14x64]
Max_pool2[7x7x64] : 1 2x2 kernels/filters/weights at stride 2, padding ='same'(입력과 출력 사이즈를 같게)

dropout [7x7x64]

Conv layer 3:

input [7x7x64]
Conv3[7x7x128] : 128 3x3 kernels/filters/weights at stride 1, padding ='same'(입력과 출력 사이즈를 같게)
ReLu 통과 [7x7x128]
Max_pool2[4x4x128] : 1 2x2 kernels/filters/weights at stride 2, padding ='same'(입력과 출력 사이즈를 같게)
dropout [4x4x128]

FC layer:

input [4x4x128]
hidden 1 node [625], weight 625 [4x4x128], b [625]
weight [625x10], b [10]

y = wx + b
cost = cross-entropy with softmax
Optimize

Step 8: Ensemble training

- Network 구조는 Step 7 과 같다. 단, 모델을 4개로 구성한다.
- ensemble 방법은 각 모델의 합을 구해 최대값으로 accuracy를 정한다.