딥러닝 및 응용

DAE 실험

2015004475 김태훈

1. 구동 환경

   OS: Windows 10 64비트

   Language: Python 3.7

   환경설정은 conda를 이용하였으며 필요 패키지는 requirements.txt에 기재됨.

2. 코드 설명

   batch_size: 미니배치 사이즈, 100.

   learning_rate: gradient descent learning rate, 0.01.

   epoch_num: 총 반복할 epoch 수, 20.

   noise_level: 인풋 데이터에 가할 노이즈 레벨, 변동.

   n_input: 인풋 레이어 차원, 784.

   n_hidden1: 1층 히든레이어(인코더1, 디코더1) 차원, 256.

   n_hidden2: 2층 히든레이어(인코더2, 디코더2) 차원, 32.

   X: mnist 이미지를 담을 [None, 784] 플레이스홀더.

   Y: 마찬가지로 mnist 이미지가 올 플레이스홀더.

encoder1: 784 -> 256

encoder2: 256 -> 32

decoder1: 32 -> 256

decoder2: 256 -> 784

cost: loss함수로 square 사용.

opt: Adam 옵티마이저 사용.

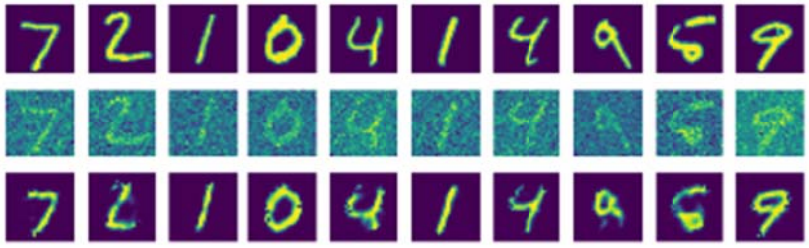
세션 가동...

1. 텐서 variable 랜덤 초기화.
2. epoch_num회의 epoch을 반복하며
    A. 해당 epoch의 평균 코스트를 초기화.
    B. 미니배치 개수만큼 반복하며
        i. 배치 사이즈만큼의 데이터를 뽑아냄
        ii. 미니배치 크기만큼 mnist 트레이닝 데이터를 가져와 batch_xs에 할당. 이 실험의 경우 라벨링 된 y는 필요 없으므로 따로 할당하지 않음.
        iii. 가져온 batch_xs에 noise_level에 따른 랜덤 노이즈를 부여해 batch_xs_noisy에 할당.
        iv. X에 batch_xs_noisy, Y에 batch_xs를 주어 학습.
            - 노이즈 있는 인풋으로부터 feature를 학습하고 노이즈 없는 원본을 복원할 수 있도록 학습.
        v. 평균 코스트 계산.
    C. epoch 번호와 평균 코스트를 출력

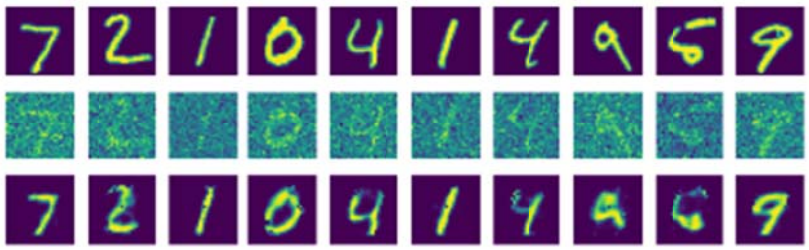
3. 테스트 데이터를 10개 가져와 노이즈 부여.
4. decoder2에 테스트 데이터를 넣어 이미지 generate.
5. 테스트 데이터의 원본 이미지, 노이즈 부여한 이미지, generate 된 결과 이미지를 출력.

3. 실험 결과

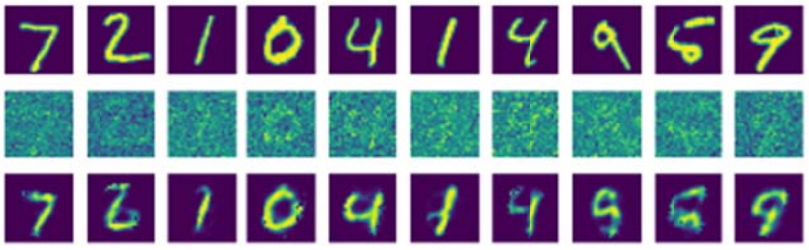
A. 784 -> 256 -> 784; noise level 0.5



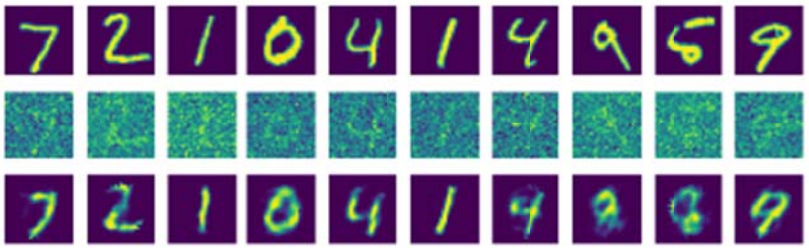
B. 784 -> 256 -> 784; noise level 0.7



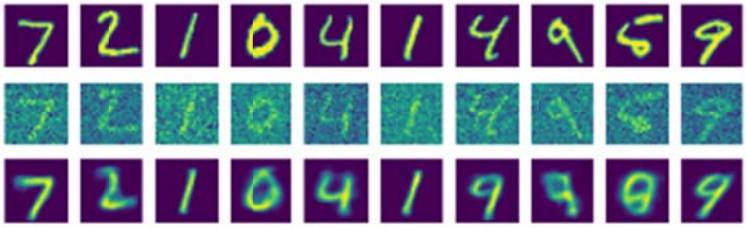
C. 784 -> 256 -> 784; noise level 0.9



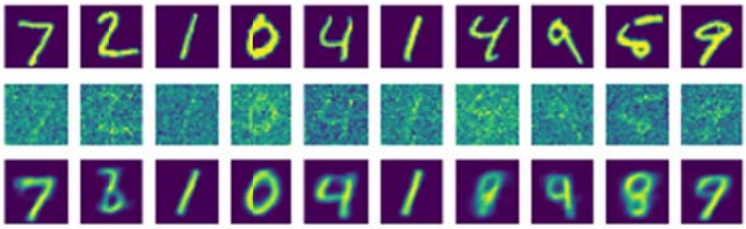
D. 784 -> 256 -> 784; noise level 0.99



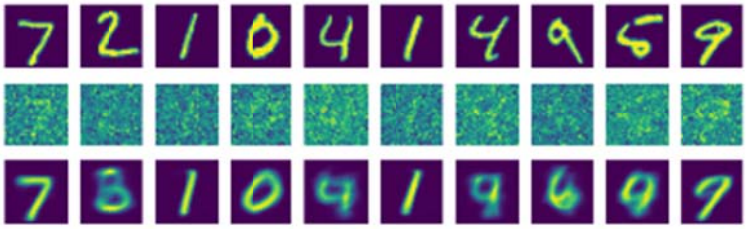
E. 784 -> 256 -> 32 -> 256 -> 784; noise level 0.5



F. 784 -> 256 -> 32 -> 256 -> 784; noise level 0.7



G. 784 -> 256 -> 32 -> 256 -> 784; noise level 0.9



결론:

노이즈가 심해질수록 결과물의 선명도가 떨어지는 패턴을 보이며, 높은 노이즈에서는 원본과 다른 숫자를 생성하는 경우도 나타남. 하지만 대체로 주요 feature 자체는 잘 학습하는 것으로 보임.

깊이를 늘렸을 때 같은 노이즈 레벨을 갖는 단층 디코더에 비해 결과물이 흐리고 부정확해짐. 오토인코더가 lossy한 특성을 갖고 차원 감소를 일으키기 때문에 32까지 줄어드는 만큼 256에 비해 결과물의 질이 떨어지는 것은 당연함.