# 기초인공지능 과제4 보고서

**20181264 노영현**

## 1. 3번 문제를 해결하기 위해 본인이 시도한 방법

input_data 행렬의 column이 1024로 고정되어서 들어오기 때문에 우선 이를 나타내는 변수 s를 선언해 1024를 저장한다. layers 라는 리스트에 nn.Linear Multi-Layer Perceptron과 activation function을 번갈아면서 삽입해준다. 위 과정을 out_feat_list에 들어있는 MLP의 layer들을 다 삽입할 때 까지 반복한다. 반복 하기 위해서 for문을 사용한다. 반복문 마다 out_feat_list에서 인자 i를 받아서 layers에 nn.Linear(s,i)를 append 해준다. 여기서 s는 중간 layer의 column을 의미하고 i는 바꾸고자 하는 column을 의미한다. 그러므로 반복문마다 append 해준 뒤 s를 i로 업데이트해준다. class내의 activatioin 변수 내에 삽입하고자 하는 activation function이 저장되어 있으므로 layers에 activation을 append하고 반복문을 끝마친다.

## 2. 4번 문제를 해결하기 위해 본인이 시도한 방법. 가장 성능이 높았던 방법에 대해서 소개. 최종적인 test 성능

우선 성능을 높이기 위해 activation fuction을 변경해보았다. 다른 것들은 그대로 유지한 체 activation function만 변경했을 때 sigmoid, tanh, relu 3개 중 relu를 사용했을 때 성능이 가장 높은 것을 확인했다. 따라서 activation function으로 relu를 사용하기로 했다. 하지만 아직 최종 성능이 40%를 넘지 않아 다른 조건도 변경시켜야 했다. LEARNING_RATE 를 변경시키기 위해서 초기 설정 값이 아닌 0.1, 0.01로 변경시켜 실행해보았는데 두 값 모두 초기 값이 0.001보다 훨씬 낮은 성능이 나왔다. 그래서 out_feat_list를 늘리기 위해서 num_layer값을 8로 바꾸고 512, 128, 32를 추가해 out_feat_list를 [1024, 512, 256, 128, 64, 32, 16, 10]으로 설정해주었다. 그리고 실행시키자 최종 성능 41.37%로 테스트 정확도 40%를 넘었다.