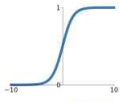
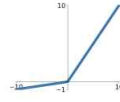
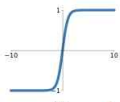
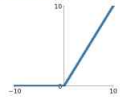
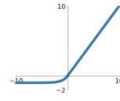
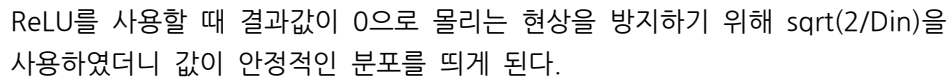


한림모여코딩 학습일지

이름	정유빈	학번	20195248	일시	2022.10.13
학습 과목	영상처리와딥러닝				
학습 계획 및 목표	<p>[Training Neural Networks]</p> <ul style="list-style-type: none"> - Activation Functions - Weight Initialization - Optimizers <p>[PyCharm]</p> <ul style="list-style-type: none"> - 디버깅 하는 법 익히기 - cnn-numpy-project 코드 분석 				
학습 내용	<p>[Training Neural Networks]</p> <ul style="list-style-type: none"> - Activation Functions <div> <div> <p>Sigmoid</p> $\sigma(x) = \frac{1}{1+e^{-x}}$  </div> <div> <p>Leaky ReLU</p> $\max(0.1x, x)$  </div> </div> <div> <div> <p>tanh</p> $\tanh(x)$  </div> <div> <p>Maxout</p> $\max(w_1^T x + b_1, w_2^T x + b_2)$ </div> </div> <div> <div> <p>ReLU</p> $\max(0, x)$  </div> <div> <p>ELU</p> $\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$  </div> </div> <p>활성화 함수의 종류는 여러 가지가 있으며 ReLU를 주로 사용한다.</p> <pre>output = np.maximum(0, x)</pre> <p><ReLU 구현></p> <p>*x < 0 일 때 gradient가 0이 되어 학습이 제대로 이루어지지 않음 -> Leaky ReLU 사용</p> <ul style="list-style-type: none"> - Weight Initialization <pre>dims = [4096] * 7 hs = [] x = np.random.randn(16, dims[0]) for Din, Dout in zip(dims[:-1], dims[1:]): W = np.random.randn(Din, Dout) * np.sqrt(2/Din) x = np.maximum(0, x.dot(W)) hs.append(x)</pre> <p><Kaiming He Initialization 구현></p>				



관성을 이용한 Momentum, 누적된 gradient 크기로 learning rate를 나누는 Adagrad / RMSProp, gradient가 0으로 치우치는 것을 막아주는 Bias correction 단계로 구성된 Adam Optimizer를 구현하였다.

```

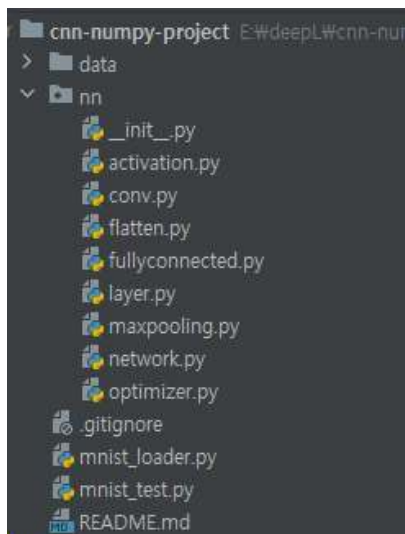
19         for l in self.layers: l: <nn.conv.Conv object at 0x000002DE4054AE48>
20             z, activation = l.forward(activation) z: [[[[ 0.00000000e+00  0.00000000e+00
21             z_stack.append(z) # 활성화 함수 거치지 않은 결과값 z
22             z_stack.append(activation) # 활성화 함수 거친 결과값 z
23
24         # cnt += 1
25
Network > train_step0 > for l in self.layers
test
onsole
Evaluate expression (Enter) or add a watch (Ctrl+Shift+Enter)
twor.py:
> activation = (ndarray: (50, 11, 11, 32)) [[[[0.00000000e+00 0.00000000e+00 0.00000000e+00 ... 0.00000000e+00,
mnist_test:
> activation = (ndarray: (50, 11, 11, 32)) [[[[0.00000000e+00 0.00000000e+00 0.00000000e+00 ... 0.00000000e+00,
> cnt = (int) 0
> l = (Conv) <nn.conv.Conv object at 0x000002DE4054AE48>
> mini_batch = (tuple: 2) (array([[[[0.,#n 0.,#n 0.,#n ...,#n 0.,#n 0.,#n 0.,#n
> mini_batch_inputs = (ndarray: (50, 28, 28, 1)) [[[[0., 0., 0., ..., 0., 0., 0., 0., 0., 0., 0., ..., 0.,
> mini_batch_outputs = (ndarray: (50, 10)) [[0. 0. 0. 0. 0. 0. 0. 1., 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0.
> self = (Network) <nn.network.Network object at 0x000002DE4055A608>
> z = (ndarray: (50, 11, 11, 32)) [[[[ 0.00000000e+00 0.00000000e+00 0.00000000e+00 ... 0.00000000e+00, 0.
> z_stack = (deque: 5) deque([array([[[[0.,#n 0.,#n 0.,#n ...,#n 0.,#n 0.,#n 0.,#n

```

activation[D][0]

	0	1	2	3	4	5
0	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
1	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
2	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
3	0.00000	0.00000	0.00007	0.00000	0.00002	0.00011
4	0.00000	0.00008	0.00059	0.00018	0.00000	0.00026
5	0.00000	0.00069	0.00080	0.00031	0.00000	0.00000
6	0.00000	0.00067	0.00042	0.00009	0.00000	0.00000
7	0.00000	0.00055	0.00013	0.00008	0.00000	0.00000
8	0.00000	0.00024	0.00015	0.00006	0.00000	0.00000
9	0.00000	0.00010	0.00000	0.00006	0.00000	0.00001
10	0.00000	0.00001	0.00000	0.00000	0.00000	0.00000

PyCharm을 통해 cnn-numpy-project 코드 분석 및 디버깅 수행
반복문이나 함수 호출을 단계별로 분석 가능 -> 코드 실행 중 원하는 위치의 값 시각적으로 확인이 가능하다.



<cnn-numpy-project 코드 구성>

학습 평가

해당 주차의 강의와 프로젝트 준비를 위해 학습 목표를 세우고, 기록하며 다시 한번 복습하는 시간을 가졌다. Neural Network의 학습 단계에서 필요한 과정들(Activation Function, Weight initialization, Optimizer)을 배우고 직접 코드로 확인하며 실행되는 과정을 디버깅을 통해 살펴보았다. PyCharm을 처음 사용해보았는데 디버깅에 유용하여 자주 사용할 것 같다.

다음 학습 계획

- 영상처리와딥러닝 cnn-numpy-project 수행
- 텍스트정보처리 기초 수학 및 전처리 부분 학습