

7장. 간단한 합성곱 모델 : 꽃 이미지 분류 신경망

1. 개요

꽃 이미지 분류 신경망 실험에서는 5장에서 사용된 꽃 이미지 데이터셋을 활용해 합성곱 신경망 사용 효과에 대해 알아보는 작업을 수행하였다. 합성곱 신경망(CNN)은 이미지 처리에 맞게 다층 퍼셉트론을 변경시킨 신경망이라 할 수 있다. 합성곱 신경망에는 합성곱 계층과 풀링 계층이 존재하는데, 합성곱 계층은 작은 가중치 텐서를 이미지의 모든 영역에 반복적으로 적용하여 위치에 관계 없이 작은 영역 안의 패턴을 찾아 처리하고, 풀링 계층은 처리할 이미지 해상도를 단계적으로 감소시켜 다양한 크기의 패턴을 단계적으로 처리할 수 있게 한다. 따라서 신경망 파라미터 수를 획기적으로 줄이면서도 전문성을 높여 품질을 향상시키는 효과를 낼 수 있다.¹ 이 실험에서는 합성곱 신경망 사용 유무에 따른 정확도 차이를 비교해보고, 합성곱 계층, 풀링 계층, 활성화 함수 등을 변화시키면서 차이를 분석해보고자 한다.

2. 학습 결과


2.1 합성곱 신경망 사용 유무

다층 퍼셉트론 사용과 합성곱 신경망 사용 경우의 정확도 차이를 살펴보고자 한다.

(1) 모델1




구분	내용
조건	- 에포크 횟수(epoch_count) : 10 - 은닉 계층 개수 = 2 - 계층 이름, 구성 정보 명시적 표기 유무 : X

¹ 합성곱 계층은 다층 퍼셉트론의 완전 연결 계층에 비해 크게 줄어든 수의 파라미터를 가지고, 풀링 계층은 아예 파라미터를 가지지 않는다. 합성곱 신경망은 소수의 파라미터를 집중적으로 학습시키는 방법으로 이미지 처리에서의 학습 품질을 크게 향상시킬 수 있다.

데이터 정확도	31.1%
소요 시간	262s
결과 화면	<pre>fm1 = CnnBasicModel('flowers_model_1', fd, [30, 10]) fm1.exec_all(epoch_count = 10, report = 2)</pre> <p>Model flowers_model_1 train started: Epoch 2: cost=1.460, accuracy=0.320/0.360 (52/52 secs) Epoch 4: cost=1.467, accuracy=0.313/0.330 (52/104 secs) Epoch 6: cost=1.431, accuracy=0.348/0.310 (54/158 secs) Epoch 8: cost=1.457, accuracy=0.326/0.300 (52/210 secs) Epoch 10: cost=1.403, accuracy=0.358/0.240 (52/262 secs) Model flowers_model_1 train ended in 262 secs: Model flowers_model_1 test report: accuracy = 0.311, (1 secs)</p> <p>Model flowers_model_1 Visualization</p>  <p>추정확률분포 [15,23,14,25,22] => 추정 sunflower : 정답 daisy => X 추정확률분포 [15,23,14,25,22] => 추정 sunflower : 정답 sunflower => 0 추정확률분포 [15,23,14,25,22] => 추정 sunflower : 정답 tulip => X</p>

(2) 모델2

구분	내용
조건	<ul style="list-style-type: none"> - 에포크 횟수(epoch_count) : 10 - 은닉 계층 개수 = 2 - 계층 이름, 구성 정보 명시적 표기 유무 : O
데이터 정확도	38.8%
소요 시간	84s

결과 화면	<pre> fm2 = CnnBasicModel('flowers_model_2', fd, [['full', {'width':30}], ['full', {'width':10}]]) fm2.use_adam = False fm2.exec_all(epoch_count = 10, report = 2) </pre> <p>Model flowers_model_2 train started:</p> <p>Epoch 2: cost=1.599, accuracy=0.259/0.270 (15/15 secs)</p> <p>Epoch 4: cost=1.524, accuracy=0.315/0.290 (17/32 secs)</p> <p>Epoch 6: cost=1.361, accuracy=0.397/0.360 (17/49 secs)</p> <p>Epoch 8: cost=1.308, accuracy=0.409/0.430 (18/67 secs)</p> <p>Epoch 10: cost=1.243, accuracy=0.454/0.350 (17/84 secs)</p> <p>Model flowers_model_2 train ended in 84 secs:</p> <p>Model flowers_model_2 test report: accuracy = 0.388, (1 secs)</p> <p>Model flowers_model_2 Visualization</p> <div style="display: flex; justify-content: space-around; align-items: center;">    </div> <p>추정확률분포 [0, 0.77, 0.23] => 추정 rose : 정답 rose => 0</p> <p>추정확률분포 [11,25,22,19,22] => 추정 dandelion : 정답 dandelion => 0</p> <p>추정확률분포 [19,27,16,18,19] => 추정 dandelion : 정답 sunflower => X</p>
-------	--

(3) 모델3

구분	내용
조건	<ul style="list-style-type: none"> - 에포크 횟수(epoch_count) : 10 - 은닉 계층 : 합성곱 계층 - [5,5](6),[3,3](12) / 풀링 계층 - [4,4],[2,2]
데이터 정확도	56.1%
소요 시간	1599s

결과 화면

```
fm3 = CnnBasicModel('flowers_model_3', fd,
                    [['conv', {'ksize':5, 'chn':6}],
                     ['max', {'stride':4}],
                     ['conv', {'ksize':3, 'chn':12}],
                     ['avg', {'stride':2}]],
                    True)
fm3.exec_all(epoch_count = 10, report = 2)
```

Model flowers_model_3 train started:

Epoch 2: cost=1.167, accuracy=0.532/0.540 (329/329 secs)

Epoch 4: cost=0.880, accuracy=0.657/0.610 (318/647 secs)

Epoch 6: cost=0.692, accuracy=0.739/0.590 (317/964 secs)

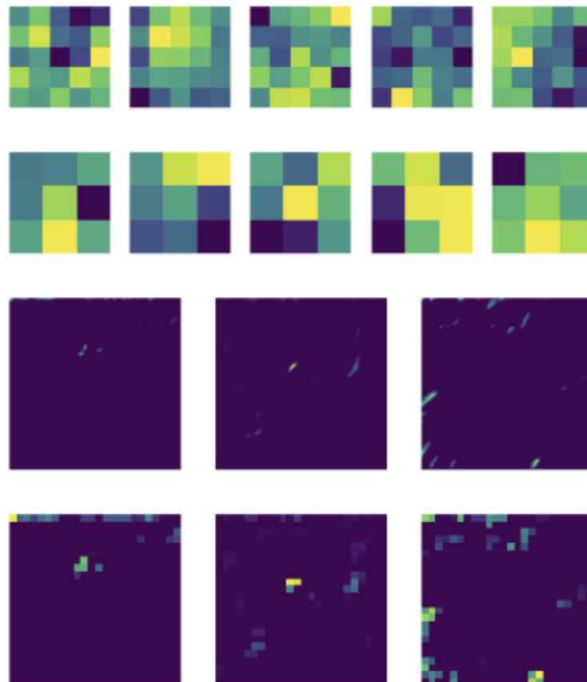
Epoch 8: cost=0.547, accuracy=0.795/0.550 (320/1284 secs)

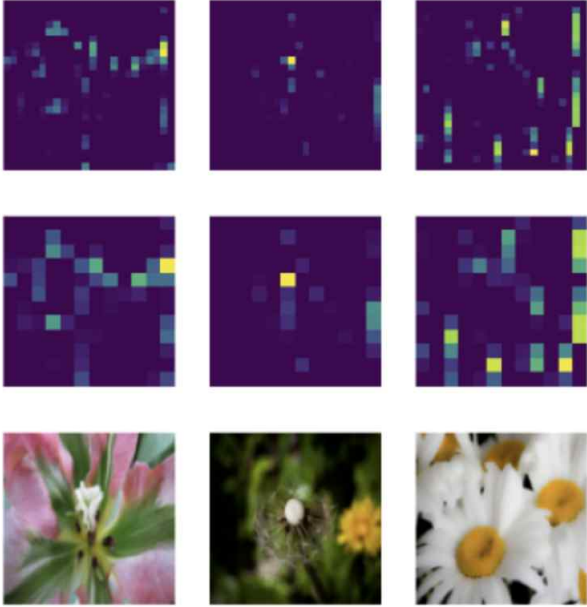
Epoch 10: cost=0.440, accuracy=0.837/0.620 (315/1599 secs)

Model flowers_model_3 train ended in 1599 secs:

Model flowers_model_3 test report: accuracy = 0.561, (9 secs)

Model flowers_model_3 Visualization




	 <p> 추정확률분포 [1, 5,33, 0,60] => 추정 tulip : 정답 tulip => 0 추정확률분포 [38,54, 1, 4, 3] => 추정 dandelion : 정답 dandelion => 0 추정확률분포 [86, 0, 1,12, 1] => 추정 daisy : 정답 daisy => 0 </p>
--	---

2.2 은닉 계층 수에 따른 비교

(1) 모델4

구분	내용
조건	- 에포크 횟수(epoch_count) : 10 - 은닉 계층 : 합성곱 계층 - [5,5](6),[3,3](12) / 풀링 계층 - [4,4],[2,2]
데이터 정확도	56.1%
소요 시간	1599s
결과 화면	<pre> fm3 = CnnBasicModel('flowers_model_3', fd, [['conv', {'ksize':5, 'chn':6}], ['max', {'stride':4}], ['conv', {'ksize':3, 'chn':12}], ['avg', {'stride':2}]], True) fm3.exec_all(epoch_count = 10, report = 2) </pre> <p> Model flowers_model_3 train started: Epoch 2: cost=1.167, accuracy=0.532/0.540 (329/329 secs) Epoch 4: cost=0.880, accuracy=0.657/0.610 (318/647 secs) Epoch 6: cost=0.692, accuracy=0.739/0.590 (317/964 secs) Epoch 8: cost=0.547, accuracy=0.795/0.550 (320/1284 secs) Epoch 10: cost=0.440, accuracy=0.837/0.620 (315/1599 secs) Model flowers_model_3 train ended in 1599 secs: Model flowers_model_3 test report: accuracy = 0.561, (9 secs) </p> <p>Model flowers_model_3 Visualization</p>

	 <p> 추정확률분포 [1, 5,33, 0,60] => 추정 tulip : 정답 tulip => 0 추정확률분포 [38,54, 1, 4, 3] => 추정 dandelion : 정답 dandelion => 0 추정확률분포 [86, 0, 1,12, 1] => 추정 daisy : 정답 daisy => 0 </p>
--	---

(2) 모델5

구분	내용
조건	<ul style="list-style-type: none"> - 에포크 횟수(epoch_count) : 10 - 은닉 계층 : 합성곱 계층 - [3,3](6),[3,3](12),[3,3](24) / 풀링 계층 - [2,2],[2,2],[3,3]
데이터 정확도	64.7%
소요 시간	1240s
결과 화면	<pre> fm4 = CnnBasicModel('flowers_model_4', fd, [['conv', {'ksize':3, 'chn':6}], ['max', {'stride':2}], ['conv', {'ksize':3, 'chn':12}], ['max', {'stride':2}], ['conv', {'ksize':3, 'chn':24}], ['avg', {'stride':3}]] fm4.exec_all(epoch_count = 10, report = 2) </pre> <p> Model flowers_model_4 train started: Epoch 2: cost=1.118, accuracy=0.569/0.530 (290/290 secs) Epoch 4: cost=0.913, accuracy=0.660/0.570 (269/559 secs) Epoch 6: cost=0.778, accuracy=0.702/0.630 (226/785 secs) Epoch 8: cost=0.670, accuracy=0.747/0.610 (227/1012 secs) Epoch 10: cost=0.582, accuracy=0.783/0.680 (228/1240 secs) Model flowers_model_4 train ended in 1240 secs: Model flowers_model_4 test report: accuracy = 0.647, (5 secs) </p> <p>Model flowers_model_4 Visualization</p>  <p> 추정확률분포 [6,73, 9, 3, 9] => 추정 dandelion : 정답 dandelion => 0 추정확률분포 [2,17,13,32,36] => 추정 tulip : 정답 tulip => 0 추정확률분포 [3, 0,96, 0, 2] => 추정 rose : 정답 tulip => X </p>

2.2.1 하이퍼파라미터에 따른 비교 - 에포크 횟수 증가




(1) 모델6

구분	내용
조건	<ul style="list-style-type: none"> - 에포크 횟수(epoch_count) : 30 - 은닉 계층 : 합성곱 계층 - [3,3](6),[3,3](12),[3,3](24) / 풀링 계층 - [2,2],[2,2],[3,3]
데이터 정확도	59.8%
소요 시간	3222s
결과 화면	<pre>fm4 = CnnBasicModel('flowers_model_4', fd, [['conv', {'ksize':3, 'chn':6}], ['max', {'stride':2}], ['conv', {'ksize':3, 'chn':12}], ['max', {'stride':2}], ['conv', {'ksize':3, 'chn':24}], ['avg', {'stride':3}]]) fm4.exec_all(epoch_count = 30, report = 6)</pre> <p>Model flowers_model_4 train started: Epoch 6: cost=0.789, accuracy=0.693/0.590 (663/663 secs) Epoch 12: cost=0.463, accuracy=0.828/0.620 (636/1299 secs) Epoch 18: cost=0.236, accuracy=0.915/0.560 (626/1925 secs) Epoch 24: cost=0.144, accuracy=0.948/0.630 (632/2557 secs) Epoch 30: cost=0.083, accuracy=0.974/0.640 (665/3222 secs) Model flowers_model_4 train ended in 3222 secs: Model flowers_model_4 test report: accuracy = 0.598, (4 secs)</p> <p>Model flowers_model_4 Visualization</p> <div style="display: flex; justify-content: space-around; align-items: center;">    </div> <p>추정확률분포 [0, 0, 0, 100, 0] => 추정 sunflower : 정답 dandelion => X 추정확률분포 [19, 0, 80, 0, 0] => 추정 rose : 정답 tulip => X 추정확률분포 [0, 100, 0, 0, 0] => 추정 dandelion : 정답 dandelion => 0</p>

2.3 활성화 함수에 따른 비교

(1) 모델 7

구분	내용
조건	<ul style="list-style-type: none"> - 에포크 횟수(epoch_count) : 10 - 활성화 함수 : sigmoid 함수 - 은닉 계층 : 합성곱 계층 - [3,3](6),[3,3](12),[3,3](24) / 풀링 계층 - [2,2],[2,2],[3,3]
데이터 정확도	50.5%
소요 시간	1290s

결과 화면	<pre> fm4 = CnnBasicModel('flowers_model_4', fd, [['conv', {'ksize':3, 'chn':6, 'actfunc':'sigmoid'}], ['max', {'stride':2}], ['conv', {'ksize':3, 'chn':12, 'actfunc':'sigmoid'}], ['max', {'stride':2}], ['conv', {'ksize':3, 'chn':24, 'actfunc':'sigmoid'}], ['avg', {'stride':3}]] fm4.exec_all(epoch_count = 10, report = 2) </pre> <p>Model flowers_model_4 train started: Epoch 2: cost=1.590, accuracy=0.243/0.280 (255/255 secs) Epoch 4: cost=1.453, accuracy=0.363/0.240 (258/513 secs) Epoch 6: cost=1.386, accuracy=0.399/0.420 (259/772 secs) Epoch 8: cost=1.308, accuracy=0.441/0.450 (259/1031 secs) Epoch 10: cost=1.197, accuracy=0.484/0.400 (259/1290 secs) Model flowers_model_4 train ended in 1290 secs: Model flowers_model_4 test report: accuracy = 0.505, (7 secs)</p> <p>Model flowers_model_4 Visualization</p> <div style="display: flex; justify-content: space-around; align-items: center;">    </div> <p> 추정확률분포 [3, 2,55, 1,39] => 추정 rose : 정답 tulip => X 추정확률분포 [1, 0,45, 2,52] => 추정 tulip : 정답 rose => X 추정확률분포 [24,20,33, 4,19] => 추정 rose : 정답 sunflower => X </p>
-------	--

(2) 모델 8

구분	내용
조건	<ul style="list-style-type: none"> - 에포크 횟수(epoch_count) : 10 - 활성화 함수 : 쌍곡탄젠트 함수 - 은닉 계층 : 합성곱 계층 - [3,3](6),[3,3](12),[3,3](24) / 풀링 계층 - [2,2],[2,2],[3,3]
데이터 정확도	53.8%
소요 시간	1326s

결과 화면

```
fm4 = CnnBasicModel('flowers_model_4', fd,
    [['conv', {'ksize':3, 'chn':6, 'actfunc':'tanh'}],
    ['max', {'stride':2}],
    ['conv', {'ksize':3, 'chn':12, 'actfunc':'tanh'}],
    ['max', {'stride':2}],
    ['conv', {'ksize':3, 'chn':24, 'actfunc':'tanh'}],
    ['avg', {'stride':3}]])
fm4.exec_all(epoch_count = 10, report = 2)
```

Model flowers_model_4 train started:
 Epoch 2: cost=1.218, accuracy=0.494/0.450 (264/264 secs)
 Epoch 4: cost=1.133, accuracy=0.543/0.570 (267/531 secs)
 Epoch 6: cost=1.084, accuracy=0.567/0.440 (265/796 secs)
 Epoch 8: cost=1.042, accuracy=0.584/0.520 (265/1061 secs)
 Epoch 10: cost=1.006, accuracy=0.606/0.460 (265/1326 secs)
 Model flowers_model_4 train ended in 1326 secs:
 Model flowers_model_4 test report: accuracy = 0.538, (8 secs)

Model flowers_model_4 Visualization



추정확률분포 [13,11, 3,61,12] => 추정 sunflower : 정답 sunflower => O
 추정확률분포 [21,60,13, 1, 4] => 추정 dandelion : 정답 daisy => X
 추정확률분포 [13, 2,32, 4,49] => 추정 tulip : 정답 daisy => X

3. 결과 분석

합성곱 신경망 사용 (다층 퍼셉트론 vs 합성곱 신경망)에 따른 차이를 표로 정리하면 다음과 같다.

	은닉 계층	명시적 표기	아담 알고리즘	데이터 정확도 (%)	소요 시간 (s)
다층 퍼셉트론	[30,10]	X	O	31.1	262
		O	X	38.8	84
합성곱 신경망	합성곱: [5,5](6),[3,3](12) 풀링:[4,4],[2,2]	-	O	56.1	1599

우선, 위 표에서 볼 수 있듯이 다층 퍼셉트론을 사용했을 때에도 계층 이름, 구성 정보 명시적 표기 유무에 따라 정확도에 차이가 있음을 알 수 있다. 정보를 명시적으로 표기함에 따라 정확도가 7.7% 상승하는 효과가 있었으며, 소요 시간에서 큰 단축이 있었다. 정보를 명

시적으로 따로 표기하지 않았을 때 CnnBasicModel 클래스 객체 초기화 함수에서의 self.use_adam default값 변경으로 아담 알고리즘이 적용되고 있어 이런 현상이 나타났을 것으로 예상된다. 즉, 원인이 이 때문이라면 이 신경망 모델은 아담 알고리즘을 이용할 때보다 일반 경사하강법을 이용할 때 성능이 더 나아지는 예외적인 사례라 볼 수 있을 것이다.

다층 퍼셉트론을 이용했을 때와 합성곱 신경망을 이용했을 때를 비교해보면, 우선 정확도의 차이가 크게 나타난 것을 확인할 수 있다. 합성곱 신경망을 이용했을 때 약 20%가량 높은 정확도를 보임을 확인할 수 있다. 또, 소요 시간이 합성곱 신경망을 이용했을 때 많이 늘었는데, 같은 아담 알고리즘을 사용했을 때와 비교해보아도 약 6배나 더 걸린 모습을 볼 수 있다. 이는 사용된 파라미터 수의 비교를 통해 다시 살펴볼 필요가 있다.

우선 다층 퍼셉트론을 이용한 신경망의 경우 입력크기가 $96 \times 96 \times 3 = 27,648$ 이고 출력크기가 5이므로 사용된 파라미터의 수는 $(27,648 \times 30 + 30) + (30 \times 10 + 10) + (10 \times 5 + 5) = 829,835$ 개이다.

합성곱 신경망의 경우, $[10, 96, 96, 3]$ 의 입력을 $[5, 5]$ 크기의 커널을 갖는 합성곱 계층이 $[10, 96, 96, 6]$ 형태로 만든 후, $[4, 4]$ 의 풀링 계층을 통해 $[10, 24, 24, 6]$ 형태로 바꾸는 과정을 거친다. 그 다음 $[3, 3]$ 크기의 커널을 갖는 합성곱 계층이 $[10, 24, 24, 12]$ 형태로 만든 후, $[2, 2]$ 의 풀링 계층을 통해 $[10, 12, 12, 12]$ 형태로 바꾸게 된다. 따라서 첫번째 합성곱 계층의 파라미터 수는 $5 \times 5 \times 3 \times 6 + 6 = 456$ 개가 되고, 두번째 합성곱 계층은 $3 \times 3 \times 6 \times 12 + 12 = 660$ 개의 파라미터를 가진다. 풀링계층은 파라미터를 가지지 않고, 출력계층은 $[12 \times 12 \times 12, 5]$ 의 가중치 행렬과 $[5]$ 의 편향 정보를 가지므로, 출력계층의 파라미터 수는 $12 \times 12 \times 12 \times 5 + 5 = 8645$ 개가 된다. 따라서 총 파라미터의 수는 $456 + 660 + 8645 = 9761$ 개로 10,000개가 되지 않는 것이다.

다층 퍼셉트론을 이용했을 때, 약 83만개의 파라미터가 사용된 것을 고려하면, 합성곱 신경망의 경우 파라미터의 개수가 아주 적은 것을 알 수 있다. 소요 시간을 다시 생각해보면, 1/80이 안되는 소수의 파라미터를 6배 시간으로 학습시킨 것이므로, 합성곱 신경망의 파라미터를 훨씬 더 많은 시간동안 학습시킨 셈이다. 이를 통해 정확도가 약 20% 높게 나타날 수 있었음을 알 수 있다.

합성곱 신경망에서 은닉 계층을 달리했을 때의 결과는 다음과 같다.

	은닉 계층		데이터 정확도 (%)	소요 시간(s)
	합성곱 계층	풀링 계층		
1	$[5, 5](6), [3, 3](12)$	$[4, 4], [2, 2]$	56.1	1599
2	$[3, 3](6), [3, 3](12), [3, 3](24)$	$[2, 2], [2, 2], [3, 3]$	64.7	1240

은닉계층을 늘렸을 때, 8.6%의 정확도가 높게 나타나고, 소요시간 또한 줄어든 것을 확인할 수 있다. 파라미터 수를 다시 살펴보면 다음과 같다. 우선 1번의 경우, 위에서 구한 것과 같이 파라미터는 9761개가 될 것이다.

2번의 경우, [3,3]커널, 6채널의 합성곱 계층과 [2,2]의 풀링 계층이 [10,48,48,6] 형태를 만들고, [3,3]커널, 12채널의 합성곱 계층과 [2,2]의 풀링 계층이 [10,24,24,12] 형태를 만들 것이다. 마지막으로 [3,3]커널, 24채널의 합성곱 계층과 [3,3]의 풀링 계층이 [10,8,8,24] 형태를 만들게 되므로 출력계층은 $[8*8*24,5]$ 의 가중치 행렬로 [10,5]의 최종 출력을 나타낸다. 파라미터 수를 살펴보면, 풀링 계층은 파라미터를 갖지 않고, 첫번째 합성곱 계층의 경우 $3*3*3*6+6=168$ 개, 두번째 합성곱 계층은 $3*3*6*12+12=660$ 개, 세번째 합성곱 계층은 $3*3*12*24+24 = 2616$ 개를 갖게 된다. 출력 계층의 경우는 $8*8*24*5+5 = 7685$ 개를 가지므로 총 파라미터 수는 $168+660+2616+7685=11,129$ 개가 된다.

즉, 파라미터 수는 1.14배 증가했고, 소요 시간은 0.77배가 되었으며, 정확도도 8.6% 증가했으므로 성능 개선이 있음을 알 수 있다.

학습 횟수를 늘렸을 때, 정확도가 높아질 것으로 예상하고 에포크 횟수를 30회로 늘려본 결과 오히려 정확도가 다소 감소하는 모습을 보였다. 이를 통해 적절한 횟수의 학습에서 좋은 성능을 나타낸다는 것을 예측해볼 수 있다.

활성화 함수에 따른 비교 결과는 다음과 같다. 은닉 계층은 동일하게 합성곱 계층 -[3,3](6), [3,3](12),[3,3](24)/풀링 계층 - [2,2],[2,2],[3,3] 조건을 사용하였다.

활성화 함수	데이터 정확도(%)	소요 시간(s)	시각화 출력 정답
ReLU 함수	64.7	1240	2/3
시그모이드 함수	50.5	1290	0/3
쌍곡 탄젠트 함수	53.8	1326	1/3

시그모이드 함수를 사용했을 때, 50.5%의 정확도가 나왔지만, 시각화 출력을 살펴본 결과 3개 중 하나도 맞지 않는 모습을 보였다. 이를 통해 학습이 제대로 이루어지지 않았음을 짐작할 수 있고, 은닉 계층의 비선형 활성화 함수로 시그모이드 함수를 이용하는 것이 바람직하지 않음을 알 수 있다. 쌍곡탄젠트 함수를 이용한 경우, 시그모이드 함수를 이용할 때보다 나은 결과가 얻어졌지만, 여전히 ReLU 함수를 이용할 때에 비하면 정확도가 낮은 결과를 보여주었다. 따라서 시그모이드 함수와 쌍곡탄젠트 함수 모두 ReLU 함수를 대신하여 은닉 계층의 비선형 활성화 함수를 이용하기는 바람직하지 못할 것으로 예상된다.

3-1. 결론

위 실험을 통해 다층 퍼셉트론을 사용한 경우, 예외적으로 이 실험에서 아담 알고리즘을 사용한 경우가 경사하강법을 이용한 경우보다 결과가 좋지 못했음을 알 수 있었다. 또한, 다층 퍼셉트론과 합성곱 신경망을 비교했을 때 합성곱 신경망이 정확도 면에서 20% 높은 결과를 보였으며, 이는 파라미터 수와 소요 시간 등을 통해 이런 현상이 나타난 이유를 짐작해 볼 수 있었다.

또, 합성곱 신경망의 은닉 계층 수가 증가했을 때, 8.6%의 정확도가 더 높아지는 모습을 관찰 할 수 있었으며, 학습 횟수가 늘어남에 따라 정확도가 높아질 것으로 예상하였으나, 적절한 횟수의 학습에서 좋은 결과를 나타낼 수 있음을 실험을 통해 확인할 수 있었다.

은닉계층의 비선형 활성화 함수를 변경해 본 결과, 시그모이드 함수와 쌍곡탄젠트 함수 모두 ReLU 함수를 이용했을 때보다 정확도가 낮게 나타났다. 따라서 두 함수 모두 ReLU 함수를 대신하여 은닉 계층의 비선형 활성화 함수를 이용하기는 바람직하지 못할 것으로 예상된다.