7장. 합성곱 신경망 모델: 꽃 이미지 분류 신경망

부산대학교 항공우주공학과

정대현 201527137

1. 개요

이전에 수행한 이미지 분류는 다층 퍼셉트론을 이용하여 학습했으나 좋은 성능을 보이지 못했습니다. 사람에게 사진에 있는 어떤 물체를 위치에 상관없이 찾는 것이 어렵지 않지만 다층 퍼셉트론은 같은 사물과 관계없이 위치가 달라지면 다른 이미지로 인식합니다. 이를 해결하기 위해 모든 위치에 특정 사물이 존재하는 사진을 준비하는 방법은 데이터를 준비하는데 엄청난 비용과 학습시간을 요구합니다. 이를 해결하기 위해 고안한 방법은 합성곱 신경망 구조(Convolutional Neural Networks) 입니다. 합성곱 계층에 있는 커널이라는 작은 가중치 텐서를 이미지의 모든 영역에 반복 적용하여 패턴을 찾으며 이미지의 해상도를 떨어뜨려 패턴을 단계적으로 처리하게 됩니다. 학습에 요구되는 파라미터를 획기적으로 줄이면서도 각 요소의 효율성을 높여 품질은 유지합니다.

2. 합성곱 신경만 구조(CNN, Convolution Neural Networks)

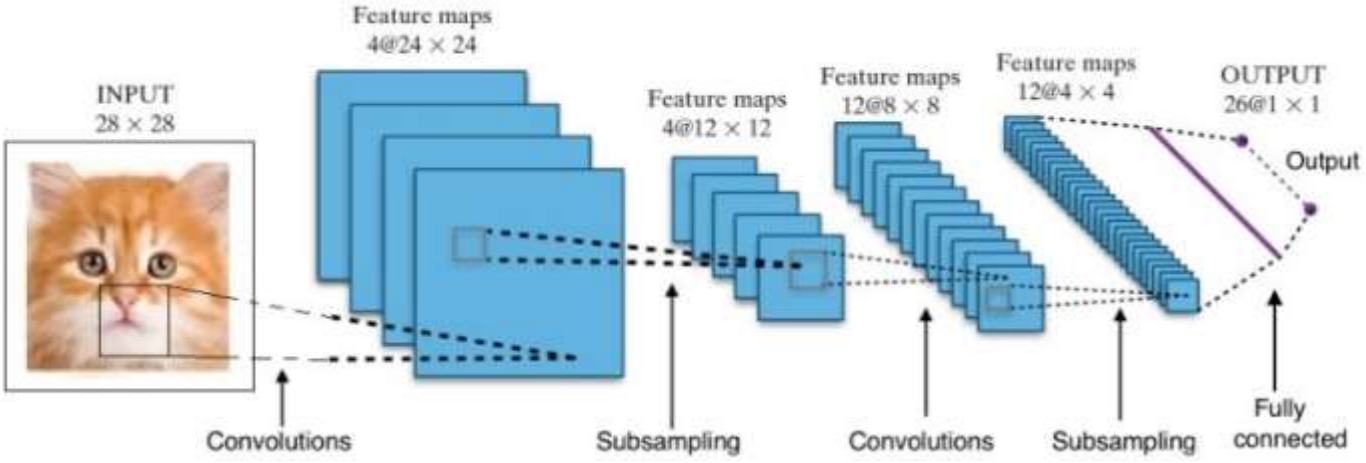

Figure 1. 합성곱신경망 레이어 구성

합성곱 계층은 커널이라는 작은 사각 영역 모양의 가중치 파라미터를 이용해 입력 픽셀값들로부터 출력 픽셀값을 계산합니다. 다층 퍼셉트론에서는 출력 픽셀 개별적으로 각각 입력 픽셀 전체로부터 계산됩니다. 합성곱 계층에서는 이와 달리 작은 영역 안에 있는 입력 픽셀 값들만 이용해 출력 픽셀을 계산합니다. 합성곱 계층의 특징은 작은 커널이 반복 적용하면서 출력 픽셀 전체를 계산을 합니다. Fig.1의 그림을 기준으로 Input 이미지가 28x28일 경우 완전 연결 계층이라고 가정할 경우 필요한 전체 파라미터 수는 28 x 28 + 28 = 812가 되야 합니다. 하지만 합성곱 연산에 사용되는 파라미터 수는 커널 가중치 행렬과 커널 편향을 합쳐 4 x 4 + 1 = 17에 불과합니다. 매우 적은 수의 파라미터로 같은 크기의 출력을 생성할 수 있습니다.

이런 커널을 이용한처리는 두가지 장점이 있습니다.

1. 커널의 반복 이용으로 커널에 대한 학습 횟수가 늘어나는 효과가 생겨서 학습의 속도가 향상됩니다.
2. 같은 커널이 모든 위치에 적용되기 때문에 한 곳에서 포착된 패턴이 다른 곳에 이용될 수 있습니다.

3. 풀링 계층(pooling layer)

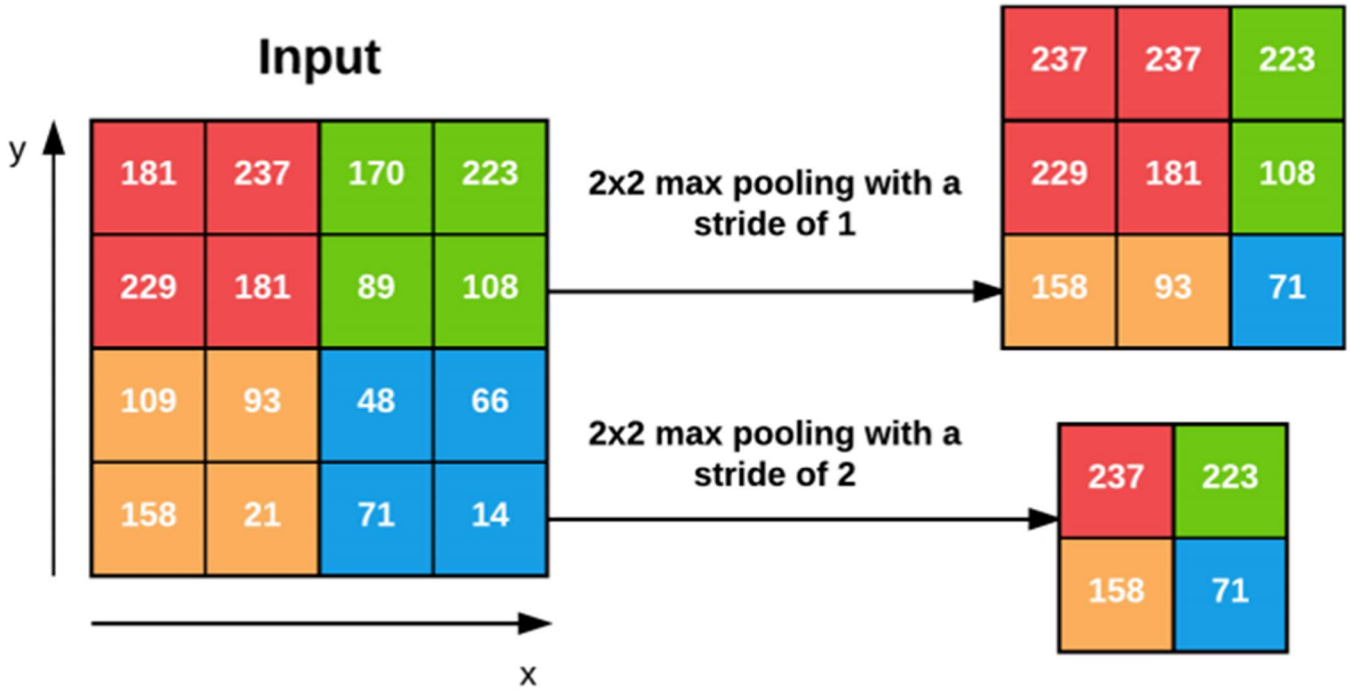

Figure 2. 풀링 계층의 구성 맥스풀링

풀링은 Fig.2 그림과 같이 일정 영역의 입력 픽셀 값 들로부터 그 최대치나 평균치 같은 대푯값을 구해 출력 픽셀로 생성하는 처리를 뜻합니다. Fig.2에서 보는 것처럼 일정한 영역 안에 있는 입력 픽셀 값을 대푯값으로 바꾸게 되며 일정한 영역의 크기를 커널이라고 합니다. 이때 사용하는 대푯값으로는 최대값 또는 평균값을 사용합니다. 풀링의 장점 두가지로 첫번째는 과적합을 방지할 수 있습니다. 풀링하는 과정에서 input size가 줄어드는 것은 불필요한 파라미터를 제거할 수 있으며 대표적인 값 추려 가지고 대상을 추정할 수 있습니다. 두번째는 첫번째와 비슷한 이유로 대푯값만 뽑아내기에 특정한 모양을 더 잘 인식할 수 있으며 모델 성능 향상을 기대할 수 있습니다.

4. 실험

4.1 Reference 5장 은닉계층 3개 폭 [50, 25, 10] epoch 10회 [CNN 미적용]

```
Epoch 2: cost=1.373, accuracy=0.390/0.500 (6/6 secs)
Epoch 4: cost=1.250, accuracy=0.454/0.400 (7/13 secs)
Epoch 6: cost=1.188, accuracy=0.479/0.370 (6/19 secs)
Epoch 8: cost=1.144, accuracy=0.516/0.460 (6/25 secs)
Epoch 10: cost=1.096, accuracy=0.543/0.370 (7/32 secs)
Model flowers_model_3 train ended in 32 secs:
Model flowers_model_3 test report: accuracy = 0.472, (0 secs)
추정확률분포 [ 7, 8,42, 6,37] => 추정 rose : 정답 tulip => X
추정확률분포 [12,43, 9,25,11] => 추정 dandelion : 정답 tulip => X
추정확률분포 [12, 8, 9,25,47] => 추정 tulip : 정답 tulip => O
```

4.2 size 5, ch 6, max stride 4 + size 3, ch 12, avg stride 2 epoch 10회 [CNN 적용]

```
Epoch 2: cost=1.167, accuracy=0.532/0.540 (137/137 secs)
Epoch 4: cost=0.880, accuracy=0.657/0.610 (145/282 secs)
Epoch 6: cost=0.692, accuracy=0.739/0.590 (148/430 secs)
Epoch 8: cost=0.547, accuracy=0.795/0.550 (148/578 secs)
Epoch 10: cost=0.440, accuracy=0.837/0.620 (147/725 secs)
Model flowers_model_3 train ended in 725 secs:
Model flowers_model_3 test report: accuracy = 0.561, (5 secs)
추정확률분포 [ 1, 5,33, 0,60] => 추정 tulip : 정답 tulip => O
추정확률분포 [38,54, 1, 4, 3] => 추정 dandelion : 정답 dandelion => O
추정확률분포 [86, 0, 1,12, 1] => 추정 daisy : 정답 daisy => O
```

4.3 size 3, ch 6, max stride 2 + size 3, ch 12, max stride 2

+ size 3, ch 24, avg stride 3 epoch 10회 [CNN 적용]

```
Epoch 2: cost=1.118, accuracy=0.569/0.530 (133/133 secs)
Epoch 4: cost=0.913, accuracy=0.660/0.570 (148/281 secs)
Epoch 6: cost=0.785, accuracy=0.700/0.610 (153/434 secs)
Epoch 8: cost=0.672, accuracy=0.753/0.600 (156/590 secs)
Epoch 10: cost=0.581, accuracy=0.782/0.710 (151/741 secs)
Model flowers_model_4 train ended in 741 secs:
Model flowers_model_4 test report: accuracy = 0.652, (2 secs)
추정확률분포 [11,34,13,20,21] => 추정 dandelion : 정답 dandelion => O
추정확률분포 [ 1, 6,15,49,28] => 추정 sunflower : 정답 tulip => X
추정확률분포 [ 3, 0,95, 0, 1] => 추정 rose : 정답 tulip => X
```

4.4 size 3, ch 6, max stride 2 + size 3, ch 12, max stride 2 + size 3, ch 24, max stride 2

+ size 3, ch 48, avg stride 3 epoch 10회 [CNN 적용]

```
Epoch 2: cost=1.143, accuracy=0.534/0.590 (150/150 secs)
Epoch 4: cost=0.947, accuracy=0.632/0.610 (145/295 secs)
Epoch 6: cost=0.840, accuracy=0.673/0.660 (147/442 secs)
Epoch 8: cost=0.774, accuracy=0.706/0.650 (147/589 secs)
Epoch 10: cost=0.683, accuracy=0.739/0.630 (140/729 secs)
Model flowers_model_4 train ended in 729 secs:
Model flowers_model_4 test report: accuracy = 0.664, (3 secs)
추정확률분포 [ 2, 6, 3,85, 4] => 추정 sunflower : 정답 tulip => X
추정확률분포 [ 8,90, 1, 0, 1] => 추정 dandelion : 정답 dandelion => O
추정확률분포 [84, 5, 0,10, 1] => 추정 daisy : 정답 daisy => O
```

4.5 size 3, ch 6, max stride 2 + size 3, ch 12, max stride 2 + size 3, ch 24, max stride 2

+ size 3, ch 48, avg stride 3 epoch 25회 [CNN 적용]

```
Model flowers_model_4 train started:
Epoch 2: cost=1.143, accuracy=0.534/0.590 (117/117 secs)
Epoch 6: cost=0.840, accuracy=0.673/0.660 (127/372 secs)
Epoch 10: cost=0.683, accuracy=0.739/0.630 (133/637 secs)
Epoch 14: cost=0.480, accuracy=0.826/0.550 (132/901 secs)
Epoch 18: cost=0.305, accuracy=0.892/0.570 (131/1166 secs)
Epoch 22: cost=0.176, accuracy=0.943/0.720 (131/1433 secs)
Model flowers_model_4 train ended in 1631 secs:
Model flowers_model_4 test report: accuracy = 0.644, (3 secs)
추정확률분포 [99, 0, 0, 0, 1] => 추정 daisy : 정답 daisy => O
추정확률분포 [24,45,22, 0, 9] => 추정 dandelion : 정답 dandelion => O
추정확률분포 [ 0, 0, 0, 2,98] => 추정 tulip : 정답 tulip => O
```

4.6 size 3, ch 6, max stride 2 + size 3, ch 12, max stride 2 + size 3, ch 24, max stride 2

+ size 3, ch 48, avg stride 3 epoch 50회 [CNN 적용]

```
Epoch 15: cost=0.427, accuracy=0.844/0.640 (343/975 secs)
Epoch 25: cost=0.153, accuracy=0.950/0.660 (354/1673 secs)
Epoch 35: cost=0.070, accuracy=0.977/0.550 (349/2388 secs)
Epoch 45: cost=0.062, accuracy=0.980/0.670 (325/3059 secs)
Epoch 50: cost=0.058, accuracy=0.982/0.600 (323/3382 secs)
Model flowers_model_4 train ended in 3382 secs:
Model flowers_model_4 test report: accuracy = 0.647, (2 secs)
추정확률분포 [ 0,100, 0, 0, 0] => 추정 dandelion : 정답 dandelion => O
추정확률분포 [100, 0, 0, 0, 0] => 추정 daisy : 정답 daisy => O
추정확률분포 [99, 1, 0, 0, 0] => 추정 daisy : 정답 daisy => O
```

4.7 size 3, ch 6, max stride 2 + size 3, ch 12, max stride 2

+ size 3, ch 24, avg stride 2 epoch 20회 [CNN 적용]

```
Model flowers_model_4 train started:
Epoch 4: cost=0.925, accuracy=0.642/0.590 (230/230 secs)
Epoch 8: cost=0.718, accuracy=0.730/0.710 (261/491 secs)
Epoch 12: cost=0.517, accuracy=0.807/0.680 (267/758 secs)
Epoch 16: cost=0.364, accuracy=0.867/0.670 (259/1017 secs)
Epoch 20: cost=0.231, accuracy=0.912/0.710 (263/1280 secs)
Model flowers_model_4 train ended in 1280 secs:
Model flowers_model_4 test report: accuracy = 0.666, (3 secs)
추정확률분포 [ 0, 0, 0, 0,100] => 추정 tulip : 정답 tulip => O
추정확률분포 [ 0, 0,24, 0,76] => 추정 tulip : 정답 rose => X
추정확률분포 [ 6,79,12, 1, 2] => 추정 dandelion : 정답 daisy => X
```

4.8 size 3, ch 6, max stride 2 + size 3, ch 12, max stride 2

+ size 3, ch 24, avg stride 2 epoch 50회 [CNN 적용]

```
Model flowers_model_4 train started:
Epoch 12: cost=0.436, accuracy=0.838/0.650 (279/788 secs)
Epoch 20: cost=0.160, accuracy=0.947/0.620 (261/1309 secs)
Epoch 28: cost=0.103, accuracy=0.967/0.670 (266/1838 secs)
Epoch 36: cost=0.094, accuracy=0.968/0.640 (274/2388 secs)
Epoch 44: cost=0.055, accuracy=0.982/0.540 (268/2934 secs)
Epoch 48: cost=0.115, accuracy=0.963/0.570 (276/3210 secs)
Model flowers_model_4 train ended in 3344 secs:
Model flowers_model_4 test report: accuracy = 0.612, (3 secs)
추정확률분포 [ 2,72,27, 0, 0] => 추정 dandelion : 정답 rose => X
추정확률분포 [ 0, 0, 1,99, 0] => 추정 sunflower : 정답 tulip => X
추정확률분포 [ 0, 0,100, 0, 0] => 추정 rose : 정답 rose => O
```

실험	은닉층or합성곱계층	epoch	시간	정확도
1	[50, 25, 10]	10	32s	0.472
2	[size 5, ch 6, max stride 4] + [size 3, ch 12, avg stride 2]	10	725s	0.561
3	size 3, ch 6, max stride 2 + size 3, ch 12, max stride 2 + size 3, ch 24, avg stride 3	10	741s	0.652
4	size 3, ch 6, max stride 2 + size 3, ch 12, max stride 2 + size 3, ch 24, max stride 2 + size 3, ch 48, avg stride 3	10	729s	0.664
5	size 3, ch 6, max stride 2 + size 3, ch 12, max stride 2 + size 3, ch 24, max stride 2 + size 3, ch 48, avg stride 3	25	1631s	0.644
6	size 3, ch 6, max stride 2 + size 3, ch 12, max stride 2	50	3382s	0.647

	+size 3, ch 24, max stride 2 + size 3, ch 48, avg stride 3			
7	size 3, ch 6, max stride 2 + size 3, ch 12, max stride 2 + size 3, ch 24, avg stride 2	20	1280s	0.666
8	size 3, ch 6, max stride 2 + size 3, ch 12, max stride 2 + size 3, ch 24, avg stride 2	50	3344s	0.612

5. 결론

이번 실험의 목적은 CNN모델의 이해도 있지만 기존 다층 퍼셉트론에 비해 성능향상이 이루어졌는지 살펴보는 것입니다. 실험 4.1은 5장 실험에서 가장 성능이 좋았던 모델을 들고 왔으며 정확도는 47.2%였습니다. 여기에서부터 CNN을 적용했을 때 얼마나 성능이 향상되는지 실험을 반복합니다. 실험 4.2에서는 간단한 합성곱신경망을 사용했을 뿐이지만 정확도는 56.1%로 다층퍼셉트론에는 나오지 않던 정확도를 바로 볼 수 있었습니다. 합성곱신경망을 여러 층으로 만들고 연산이 반복이 될수록 정확도 역시 빠르게 상승했습니다. 어느정도 합성곱계층을 생성한 실험의 경우 모두 기본적으로 60%이상의 성능을 보였습니다. 어떤 식으로 해도 다층 퍼셉트론보다는 우수한 성능을 보이기에 이미지 영상 처리라면 큰 고민 없이 CNN을 사용해도 좋은 결과를 얻을 수 있습니다.

하지만 이어서 계속해서 합성곱계층과 반복횟수를 늘려보았을 때 합성곱신경망의 특성상 같은 크기의 데이터를 사용해도 학습시간이 오래 걸립니다. 이후에는 학습시간이 크게 걸리지만 정확도는 소폭 증가하거나 오히려 감소했습니다. 실험 7과 3를 비교하면 학습시간은 두배 안되는 정도로 늘어났으나 성능 향상은 0.013에 그쳤습니다. 여기서 더 반복 횟수를 늘려 보았지만 정확도는 소폭 감소했으며 더 이상의 학습은 이루어지지 않는 모습을 보입니다. 어디까지나 "CNN은 모든 위치의 데이터를 준비하지 않아도 된다" 였지만 데이터셋으로 사용한 4천여개의 꽃 사진으로는 다소 부족한 모습을 보였습니다.