7장. 합성곱 신경망 과제

부산대학교 전기컴퓨터공학부 정보컴퓨터공학전공
201724579 정현모

1. 개요

기존 과제에서 진행했던 신경망은 완전 연결 신경망으로 인접한 양쪽 계층의 모든 퍼셉트론 쌍을 빠짐없이 서로 연결하는 신경망이었다. 입력층, 은닉층, 출력층이 모두 완전 연결 계층으로 된 신경망의 문제점은 신경망의 크기에 따라 메모리에 큰 부담이 가고, 많은 파라미터 탓에 학습속도가 느리다는 점, 많은 파라미터의 학습을 위해 많은 양의 데이터가 필요하다는 점이 있었다. 이전 과제에서 이미지를 분류하는 완전 연결 신경망에서 이미지의 입력이 매우 커 파라미터가 아주 많아 학습 속도나 정확도가 굉장히 떨어졌었다.

이에 등장한 해결방법인 합성곱 신경망은 작은 가중치 커널을 이미지의 모든 영역에 반복 적용하는 것이다. 풀링 계층은 처리할 이미지 해상도를 단계적으로 감소시켜 다양한 크기의 패턴을 단계적으로 처리할 수 있게 한다. 신경망 파라미터 수를 획기적으로 줄이면서도, 이미지의 공간적 정보를 학습하여 품질을 향상시킬 수 있다.

2. 학습 결과

2.1. Flowers 모델 학습 비교

2.1.1. hidden layer, adam Optimizer 유무에 따른 학습 비교

파란색 하이라이트는 변경된 부분을 의미한다.

flowers_model_1

구분	내용	값
조건	Epoch	10
	hidden layer	**[30, 10]**
	Adam Optimizer	**True**
최종 결과	Accuarcy	31.1%
	Train time	160 secs
결과 화면	Model flowers_model_1 train started: Epoch 2: cost=1.460, accuracy=0.320/0.360 (31/31 secs) Epoch 4: cost=1.467, accuracy=0.313/0.330 (31/62 secs) Epoch 6: cost=1.431, accuracy=0.348/0.310 (32/94 secs) Epoch 8: cost=1.457, accuracy=0.326/0.300 (33/127 secs) Epoch 10: cost=1.403, accuracy=0.358/0.240 (33/160 secs) Model flowers_model_1 train ended in 160 secs: Model flowers_model_1 test report: accuracy = 0.311, (0 secs)	

7장. 합성곱 신경망 과제

flowers_model_2

구분	내용	값
조건	Epoch	10
	hidden layer	**[30, 10]**
	Adam Optimizer	**False**
최종 결과	Accuarcy	38.8%
	Train time	42 secs
결과 화면	Model flowers_model_2 train started: Epoch 2: cost=1.599, accuracy=0.259/0.270 (9/9 secs) Epoch 4: cost=1.524, accuracy=0.315/0.290 (8/17 secs) Epoch 6: cost=1.361, accuracy=0.397/0.360 (9/26 secs) Epoch 8: cost=1.308, accuracy=0.409/0.430 (8/34 secs) Epoch 10: cost=1.243, accuracy=0.454/0.350 (8/42 secs) Model flowers_model_2 train ended in 42 secs: Model flowers_model_2 test report: accuracy = 0.388, (0 secs)	

flowers_model_3

구분	내용	값
조건	Epoch	10
	hidden layer	**['conv', {'ksize':5, 'chn':6}],** **['max', {'stride':4}],** **['conv', 'ksize':3, 'chn':12}],** **['avg', {'stride':2}]**
	Adam Optimizer	**True**
최종 결과	Accuarcy	56.3%
	Train time	**817 secs**
결과 화면	Model flowers_model_3 train started: Epoch 2: cost=1.167, accuracy=0.532/0.540 (169/169 secs) Epoch 4: cost=0.880, accuracy=0.657/0.610 (163/332 secs) Epoch 6: cost=0.694, accuracy=0.739/0.600 (161/493 secs) Epoch 8: cost=0.546, accuracy=0.796/0.580 (162/655 secs) Epoch 10: cost=0.453, accuracy=0.829/0.620 (162/817 secs) Model flowers_model_3 train ended in 817 secs: Model flowers_model_3 test report: accuracy = 0.563, (5 secs)	

7장. 합성곱 신경망 과제

flowers_model_4

구분	내용	값
조건	Epoch	10
	hidden layer	**['conv', {'ksize':3, 'chn':6}], ['max', {'stride':2}], ['conv', {'ksize':3, 'chn':12}], ['max', {'stride':2}], ['conv', {'ksize':3, 'chn':24}], ['avg', {'stride':3}]**
	Adam Optimizer	**True**
최종 결과	Accuarcy	**64.9%**
	Train time	692 secs
결과 화면	Model flowers_model_4 train started: Epoch 2: cost=1.118, accuracy=0.569/0.530 (138/138 secs) Epoch 4: cost=0.913, accuracy=0.658/0.570 (139/277 secs) Epoch 6: cost=0.786, accuracy=0.701/0.620 (137/414 secs) Epoch 8: cost=0.669, accuracy=0.748/0.620 (138/552 secs) Epoch 10: cost=0.578, accuracy=0.787/0.700 (140/692 secs) Model flowers_model_4 train ended in 692 secs: Model flowers_model_4 test report: accuracy = 0.649, (3 secs)	

7장. 합성곱 신경망 과제

2.2. Office31 모델 학습 비교

2.2.1. hidden layer, adam Optimizer 유무에 따른 학습 비교

파란색 하이라이트는 변경된 부분을 의미한다.

office_model_1

구분	내용	값
조건	Epoch	10
	hidden layer	**['conv', {'ksize':3, 'chn':6}],** **['max', {'stride':2}],** **['conv', {'ksize':3, 'chn':12}],** **['max', {'stride':2}],** **['conv', {'ksize':3, 'chn':24}],** **['avg', {'stride':3}]**
	Conv activate function	**None**
최종 결과	Accuarcy	**87.4% + 52.6%**
	Train time	704 secs
결과 화면	Model office31_model_1 train started: Epoch 2: cost=2.611, accuracy=0.834+0.413/0.820+0.420 (140/140 secs) Epoch 4: cost=1.555, accuracy=0.900+0.640/0.830+0.440 (141/281 secs) Epoch 6: cost=0.900, accuracy=0.930+0.784/0.830+0.600 (140/421 secs) Epoch 8: cost=0.517, accuracy=0.955+0.888/0.850+0.520 (140/561 secs) Epoch 10: cost=0.319, accuracy=0.962+0.936/0.890+0.510 (143/704 secs) Model office31_model_1 train ended in 704 secs: Model office31_model_1 test report: accuracy = 0.874+0.526, (3 secs)	

7장. 합성곱 신경망 과제

office_model_2

구분	내용	값
조건	Epoch	10
	hidden layer	**['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}]**
	Conv activate function	**'sigmoid'**
최종 결과	Accuarcy	69.3% + 4%
	Train time	752 secs
결과 화면	Model office31_model_2 train started: Epoch 2: cost=4.268, accuracy=0.686+0.031/0.700+0.010 (145/145 secs) Epoch 4: cost=4.259, accuracy=0.686+0.037/0.610+0.010 (150/295 secs) Epoch 6: cost=4.262, accuracy=0.686+0.034/0.620+0.010 (151/446 secs) Epoch 8: cost=4.259, accuracy=0.686+0.034/0.670+0.020 (153/599 secs) Epoch 10: cost=4.260, accuracy=0.686+0.039/0.620+0.030 (153/752 secs) Model office31_model_2 train ended in 752 secs: Model office31_model_2 test report: accuracy = 0.693+0.040, (5 secs)	

office_model_3

구분	내용	값
조건	Epoch	10
	hidden layer	**['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}]**
	Conv activate function	**'tanh'**
최종 결과	Accuarcy	77.6 + 37%
	Train time	807 secs
결과 화면	Model office31_model_3 train started: Epoch 2: cost=3.340, accuracy=0.724+0.298/0.700+0.250 (158/158 secs) Epoch 4: cost=2.992, accuracy=0.755+0.367/0.690+0.310 (161/319 secs) Epoch 6: cost=2.706, accuracy=0.778+0.432/0.790+0.290 (159/478 secs) Epoch 8: cost=2.523, accuracy=0.785+0.473/0.740+0.300 (164/642 secs) Epoch 10: cost=2.366, accuracy=0.805+0.500/0.770+0.320 (165/807 secs) Model office31_model_3 train ended in 807 secs: Model office31_model_3 test report: accuracy = 0.776+0.370, (6 secs)	

7장. 합성곱 신경망 과제

3. 결론

3.1. 합성곱 계층

3.1.1. 커널

커널은 합성 곱 연산에 사용되는 임의 크기의 사각 영역 형태의 가중치로, 완전 연결 계층에서 합성 곱 계층으로 넘어갈 때 파라미터 수를 획기적으로 줄이면서 품질을 높이게 해주었다. 아래 **그림1**에서 3 x 3 사이즈의 가중치 영역이 커널로, 커널이 많으면 많을수록 이미지에서 여러가지 특징을 잡아낼 수 있고 커널이 크면 클수록 이미지의 한 영역의 정보를 많이 얻을 수 있다. 하지만 너무 크거나 많아지면 연산이 많아져 학습속도가 느려지게 된다.

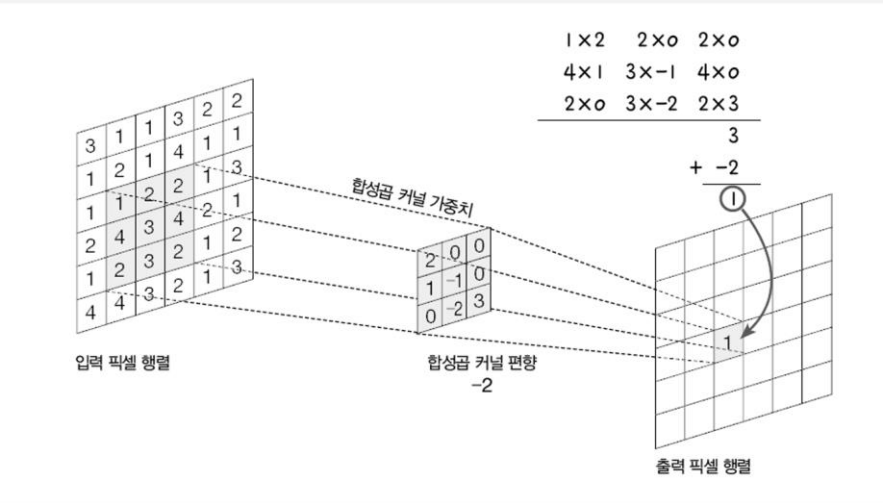


그림 1. 합성곱 커널

3.1.2. 패딩

패딩은 아래 **그림2**와 같이 합성곱 연산에서 이미지의 범위를 벗어날 때의 처리 방법이다. 입력과 같은 크기의 출력을 생성하는 SAME 패딩 방식과 입력과 다른 크기의 출력을 생성하는 VALID 패딩 방식이 존재한다. SAME 패딩 방식은 입력과 출력 크기가 동일할 뿐만 아니라, 모든 입력 픽셀이 공평하게 출력에 반영된다. 그에 반해 VALID 패딩 방식은 가장자리에 있는 픽셀이 출력에 상대적으로 적게 영향을 미친다.

7장. 합성곱 신경망 과제

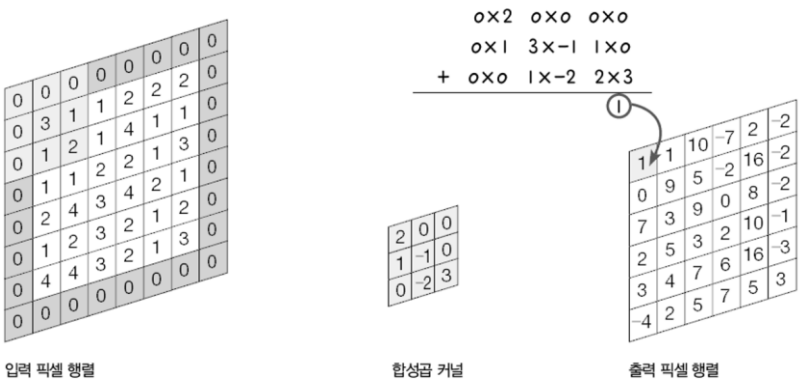
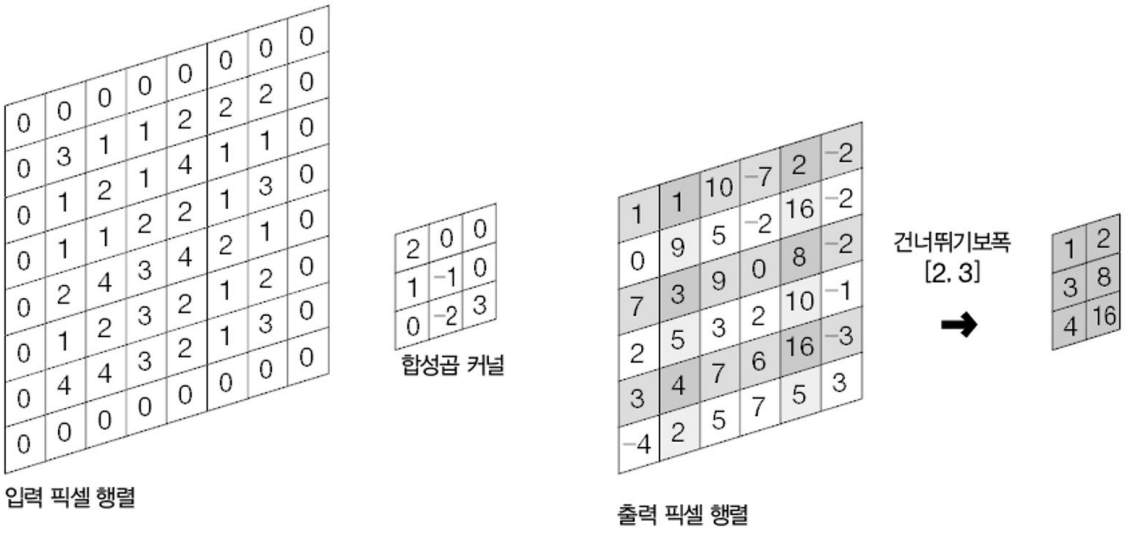


그림 2. 패딩(padding)

3.1.3. 건너뛰기 (stride)

커널이 일정한 간격으로 건너뛰면서 출력 픽셀들을 생성하며 Default로 보통 [1, 1] stride를 사용한다. Stride 크기에 따라 출력 크기가 변한다.



※ 커널 크기, 보폭: 정방형이 널리 이용되지만 꼭 그럴 필요는 없음

그림 3. 건너뛰기(strides)

7장. 합성곱 신경망 과제

3.2. 풀링 계층

커널 영역의 입력 픽셀값들의 대푯값을 구해 출력 픽셀로 생성하는 작업이다. 대푯값으로 보통 최댓값이나 평균값을 사용하고, 풀링 작업 후 보통 출력 크기가 작아진다는 특징이 있다.

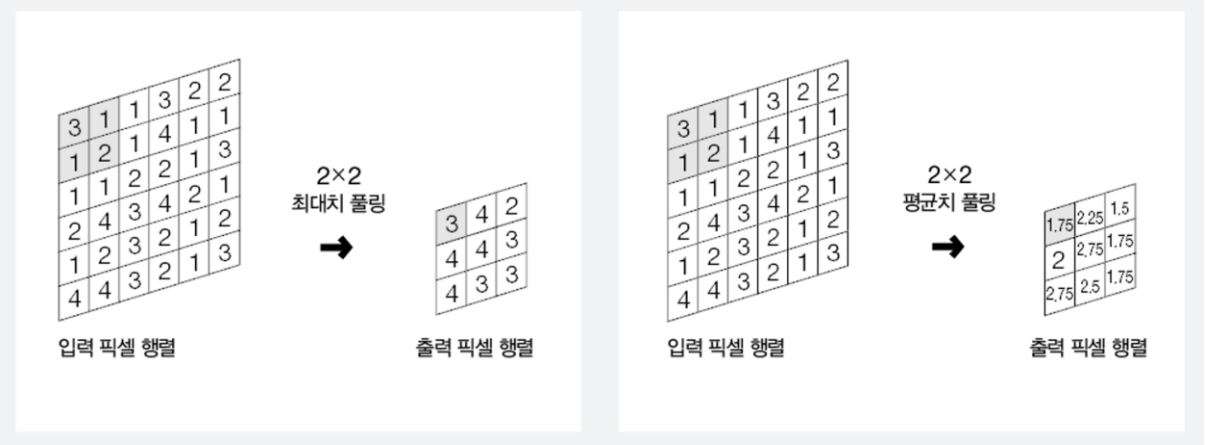


그림 4. 풀링(pooling)

3.3. 실험 결과

이전 과제에서 office31의 복합 출력이나, flowers의 이미지 인식을 완전 연결 계층으로 시도해 보았다. 하지만 학습하는데 너무 오랜 시간이 걸릴 뿐만 아니라 정확도도 크게 좋지 않았던 것을 알 수 있다. 하지만 이번에 합성곱 신경망을 사용하면서 기존 47.7% 정확도에서 64.9%까지 정확도가 증가하였다. 하지만 분명 파라미터 수가 줄었을 합성곱 신경망 모델에서 10 epoch 임에도 불구하고 시간이 월등히 많이 걸린 것을 알 수 있다. 기존 47.7%의 정확도를 보인 완전 연결 신경망의 epoch가 50에 시간이 4분정도가 걸린 것에 비해, 합성곱 신경망 모델은 epoch가 10에 12분 정도가 걸렸다. 합성곱 신경망이 연산 속도 면에서 훨씬 빠르다고 알고 있었기에 직접 계산을 해보았다.

완전 연결 계층: [320 * 240 * 3](Input) * [40 * 30 * 20 * 10] * 5(output) = 276,480,000,000개

합성곱 신경망: [320 * 240 * 3](Input) * (9 * 6) * (9* 12) * (9 * 24) * 5 = 1,451,188,224,000개

실제로 파라미터 개수를 계산해보니 합성곱 신경망의 파라미터 개수가 약 4배정도 더 많았고 시간이 더 걸린 것이 이해가 되었다. 정확도가 상승하긴 했으나, 월등한 상승은 아니었기 때문에 더 수정이 필요할 것으로 보인다.