CNN (Convolution Neural Network)

2018년 10월 5일 금요일 오후 4:08

Fully Connected Layer[1] 만으로 구성된 인공 신경망의 입력 데이터는 1차원(배열) 형태로 한정됩니다.
한 장의 컬러 사진은 3차원 데이터입니다. 배치 모드에 사용되는 여러장의 사진은 4차원 데이터입니다.
사진 데이터로 전연결(FC, Fully Connected) 신경망을 학습시켜야 할 경우에, 3차원 사진 데이터를 1차원으로 평면화시켜야 합니다.
사진 데이터를 평면화 시키는 과정에서 공간 정보가 손실될 수밖에 없습니다.

결과적으로 이미지 공간 정보 유실로 인한 정보 부족으로 인공 신경망이 특징을 추출 및 학습이 비효율적이고 정확도를 높이는데 한계가 있습니다.

이미지의 공간 정보를 유지한 상태로 학습이 가능한 모델이 바로 CNN(Convolutional Neural Network)입니다.
CNN(Convolutional Neural Network)은 기존 Fully Connected Neural Network와 비교하여 다음과 같은 차별성을 갖습니다.

- 각 레이어의 **입출력 데이터의 형상 유지**
- 이미지의 **공간 정보를 유지**하면서 **인접 이미지와의 특징을 효과적으로 인식**
- **복수의 필터**로 **이미지의 특징 추출 및 학습**
- 추출한 **이미지의 특징을 모으고 강화하는 Pooling 레이어**
- 필터를 공유 파라미터로 사용하기 때문에, 일반 인공 신경망과 비교하여 **학습 파라미터가 매우 적음**

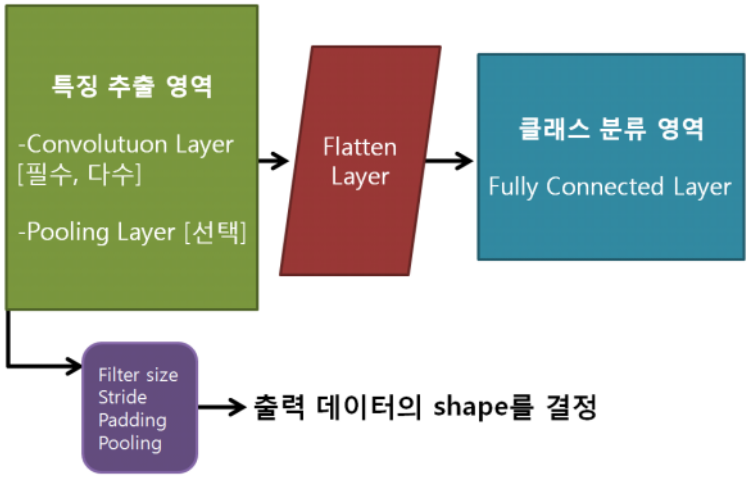
CNN은 위 이미지와 같이 이미지의 **특징을 추출하는 부분**과 **클래스를 분류하는 부분**으로 나눌 수 있습니다.

1. **특징 추출 영역**

특징 추출 영역은 **Convolution Layer와 Pooling Layer를 여러 겹 쌓는 형태로 구성**됩니다.[2]
Convolution Layer는 입력 데이터에 필터를 적용 후 활성화 함수를 반영하는 필수 요소입니다. Convolutional Layer 다음에 위치하는 Pooling Layer는 선택적인 레이어입니다.

2. **클래스 분류 영역**

CNN 마지막 부분에는 이미지 분류를 위한 **Fully Connected 레이어가 추가**됩니다. 이미지의 **특징을 추출하는 부분과 이미지를 분류하는 부분 사이에** 이미지 형태의 데이터를 배열 형태로 만드는 **Flatten 레이어가 위치** 합니다.



CNN은 이미지 특징 추출을 위하여 <그림1>과 같이 입력데이터를 필터가 순회하며 합성곱을 계산하고, 그 계산 결과를 이용하여 Feature map을 만듭니다. Convolution Layer는 **Filter 크기, Stride, Padding 적용 여부, Max Pooling 크기에 따라서 출력 데이터의 Shape이 변경**됩니다.

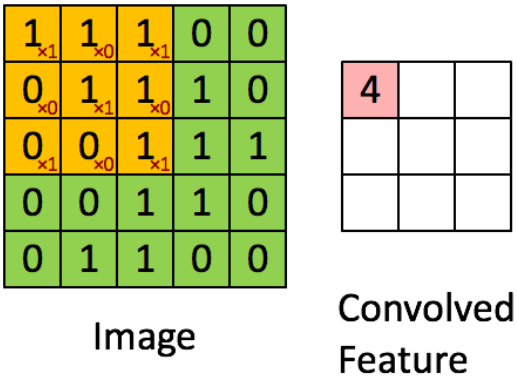
1. CNN의 주요 용어 정리

CNN에는 다음과 같은 용어들이 사용됩니다.

- Convolution(합성곱)
- 채널(Channel)
- 필터(Filter)
- 커널(Kernel)
- 스트라이드(Strid)
- 패딩(Padding)
- 피처 맵(Feature Map)
- 액티베이션 맵(Activation Map)
- 풀링(Pooling) 레이어

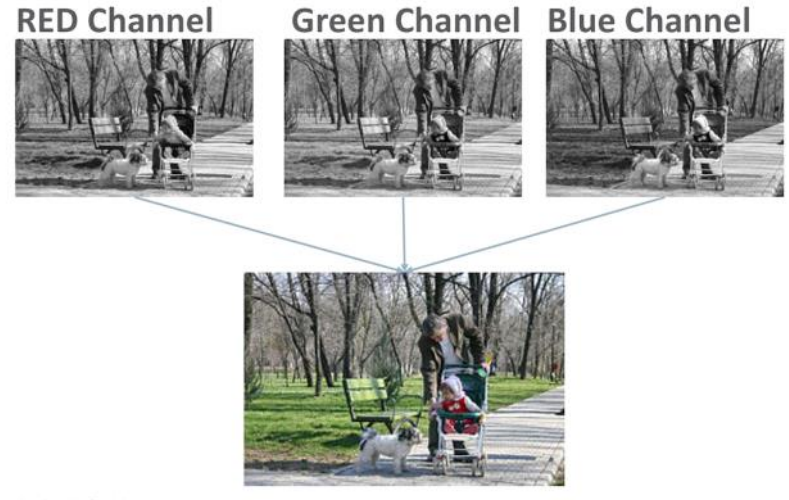
1.1 합성곱, Convolution

필터를 이용해 합성곱 연산을 수행하여, 합성곱 처리 결과로 부터 Feature Map을 만든다.



1.2 채널, Channel

이미지 픽셀 하나하나는 실수입니다. 컬러 사진은 천연색을 표현하기 위해서, 각 픽셀을 RGB 3개의 실수로 표현한 3차원 데이터입니다.(<그림 2> 참조) 컬러 이미지는 3개의 채널로 구성됩니다. 반면에 흑백 명암만을 표현하는 흑백 사진은 2차원 데이터로 1개 채널로 구성됩니다. 높이가 39 픽셀이고 폭이 31 픽셀인 컬러 사진 데이터의 shape은 (39, 31, 3)[3]으로 표현합니다. 반면에 높이가 39픽셀이고 폭이 31픽셀인 흑백 사진 데이터의 shape은 (39, 31, 1)입니다.

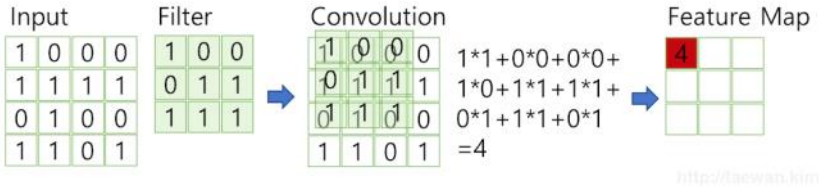


이미지 출처: https://en.wikipedia.org/wiki/Channel_(digital_image)

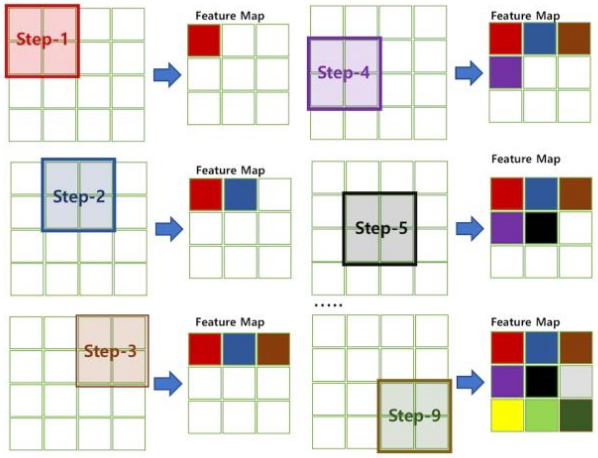
Convolution Layer에 유입되는 입력 데이터에는 한 개 이상의 필터가 적용됩니다. 1개 필터는 Feature Map의 채널이 됩니다. Convolution Layer에 **n개**의 필터가 적용된다면 출력 데이터는 **n개**의 채널을 갖게 됩니다.

1.3 필터(Filter) & Stride

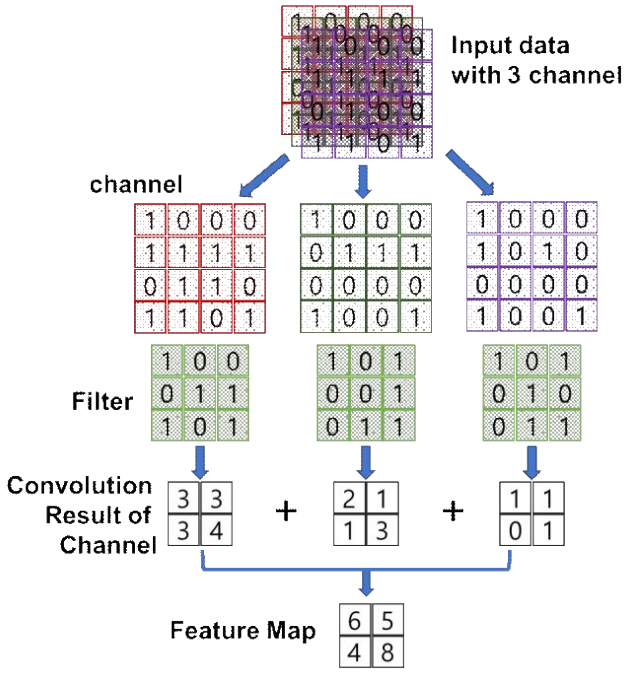
필터는 이미지의 특징을 찾아내기 위한 공용 파라미터입니다. Filter를 Kernel이라고 하기도 합니다. CNN에서 Filter와 Kernel은 같은 의미입니다. 필터는 일반적으로 (4, 4)이나 (3, 3)과 같은 정사각 행렬로 정의됩니다. CNN에서 학습의 대상은 필터 파라미터입니다. <그림 1>과 같이 입력 데이터를 지정된 간격으로 순회하며 채널별로 합성곱을 하고 모든 채널(컬러의 경우 3개)의 합성곱의 합을 Feature Map로 만듭니다. 필터는 지정된 간격으로 이동하면서 전체 입력데이터와 합성곱하여 Feature Map을 만듭니다. <그림 3>은 채널이 1개인 입력 데이터를 (3, 3) 크기의 필터로 합성곱하는 과정을 설명합니다.



필터는 입력 데이터를 지정한 간격으로 순회하면서 합성곱을 계산합니다. 여기서 지정된 간격으로 필터를 순회하는 간격을 Stride라고 합니다. <그림 4>는 strid가 1로 필터를 입력 데이터에 순회하는 예제입니다. strid가 2로 설정되면 필터는 2칸씩 이동하면서 합성곱을 계산합니다.



입력 데이터가 여러 채널을 갖을 경우 필터는 각 채널을 순회하며 합성곱을 계산한 후, 채널별 피처 맵을 만듭니다. 그리고 각 채널의 피처 맵을 합산하여 최종 피처 맵으로 반환합니다. **입력 데이터는 채널 수와 상관없이 필터 별로 1개의 피처 맵이 만들어집니다.** <그림 5 참조>



하나의 Convolution Layer에 크기가 같은 여러 개의 필터를 적용할 수 있습니다. 이 경우에 Feature Map에는 필터 갯수 만큼의 채널이 만들어집니다. 입력데이터에 적용한 필터의 개수는 출력 데이터인 Feature Map의 채널이 됩니다.
Convolution Layer의 입력 데이터를 필터가 순회하며 합성곱을 통해서 만든 출력을 Feature Map 또는 Activation Map이라고 합니다. Feature Map은 합성곱 계산으로 만들어진 행렬입니다. Activation Map은 Feature Map 행렬에 활성 함수를 적용한 결과입니다. 즉 Convolution 레이어의 최종 출력 결과가 Activation Map입니다.

1.4 패딩(Padding)

Convolution 레이어에서 Filter와 Stride에 작용으로 Feature Map 크기는 입력데이터 보다 작습니다. Convolution 레이어의 출력 데이터가 줄어드는 것을 방지하는 방법이 패딩입니다. 패딩은 입력 데이터의 외각에 지정된 픽셀만큼 특정 값으로 채워 넣는 것을 의미합니다. 보통 패딩 값으로 0으로 채워 넣습니다.

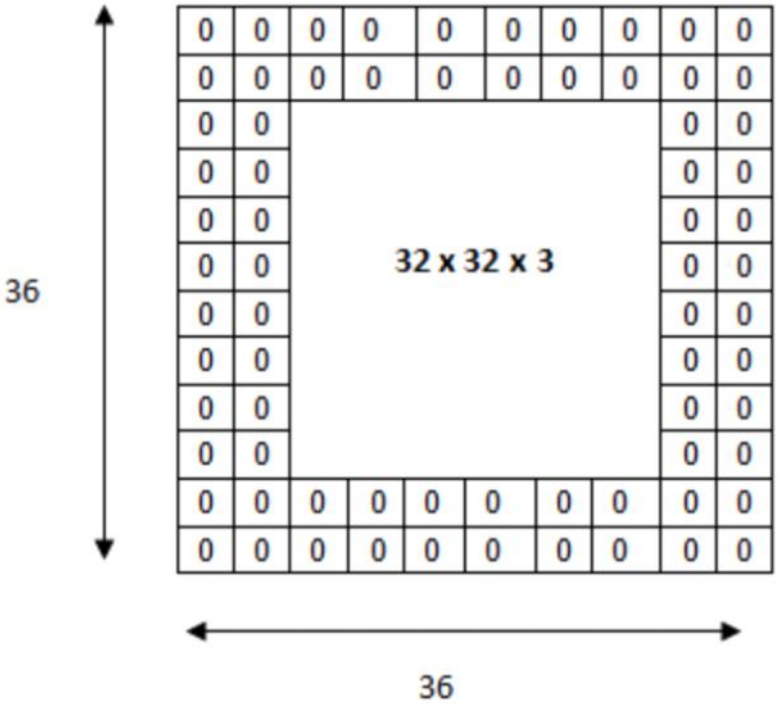


그림 6: padding 예제: 2pixel 추가

<그림 6>은 (32, 32, 3) 데이터를 외각에 2 pixel을 추가하여 (36, 36, 3) 행렬을 만드는 예제입니다. Padding을 통해서 Convolution 레이어의 출력 **데이터의 사이즈를 조절하는 기능**이 외에, 외각을 "0"값으로 둘러싸는 특징으로 부터 인공 **신경망이 이미지의 외각을 인**

식하는 학습 효과도 있습니다.

1.5 Pooling 레이어

풀링 레이어는 컨볼류션 레이어의 출력 데이터를 입력으로 받아서 출력 데이터(Activation Map)의 크기를 줄이거나 **특정 데이터를 강조하는 용도로 사용**됩니다. 플링 레이어를 처리하는 방법으로는 Max Pooling과 Average Pooning, Min Pooling이 있습니다. 정사각 행렬의 특정 영역 안에 값의 최댓값을 모으거나 특정 영역의 평균을 구하는 방식으로 동작합니다. <그림 7>은 Max pooling과 Average Pooling의 동작 방식을 설명합니다. 일반적으로 Pooing 크기와 Stride를 같은 크기로 설정하여 모든 원소가 한 번씩 처리 되도록 설정합니다.

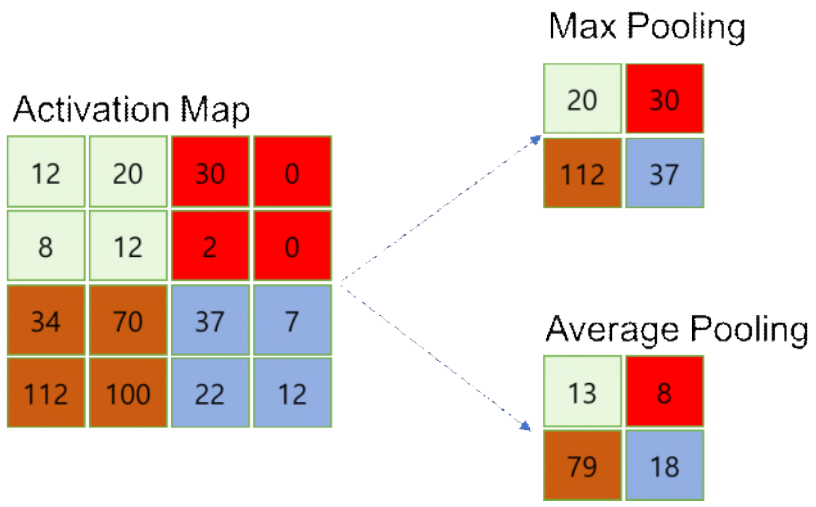


그림 7: Pooling 예제: Max Pooling, Average Pooling

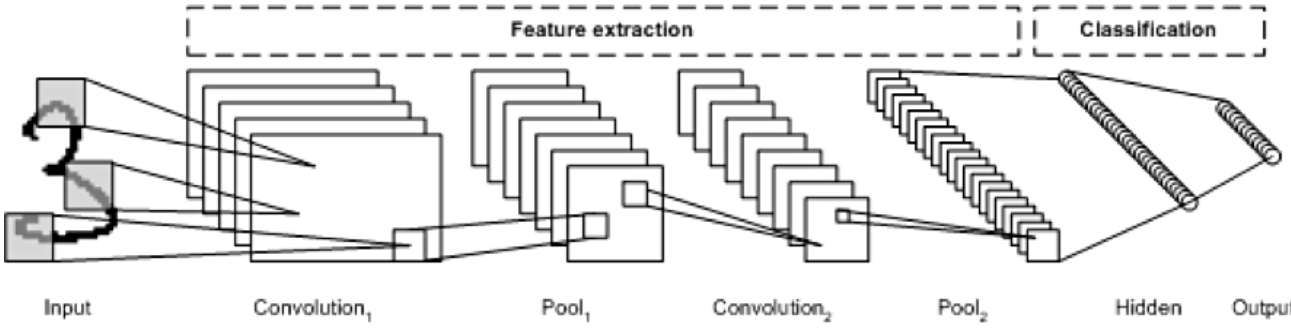
Pooling 레이어는 Convolution 레이어와 비교하여 다음과 같은 특징이 있습니다.

- 학습대상 파라미터가 없음
- Pooling 레이어를 통과하면 행렬의 크기 감소
- Pooling 레이어를 통해서 채널 수 변경 없음

CNN에서는 주로 Max Pooling을 사용합니다.

CNN, Convolution Neural Network 요약

January 04, 2018 Machine Learning



Fully Connected Layer[1] 만으로 구성된 인공 신경망의 입력 데이터는 1차원(배열) 형태로 한정됩니다. 한 장의 컬러 사진은 3차원 데이터입니다. 배치 모드에 사용되는 여러장의 사진은 4차원 데이터입니다. 사진 데이터로 전연결(FC, Fully Connected) 신경망을 학습시켜야 할 경우에, 3차원 사진 데이터를 1차원으로 평면화시켜야 합니다. 사진 데이터를 평면화 시키는 과정에서 공간 정보가 손실될 수밖에 없습니다. 결과적으로 이미지 공간 정보 유실로 인한 정보 부족으로 인공 신경망이 특징을 추출 및 학습이 비효율적이고 정확도를 높이는데 한계가 있습니다. 이미지의 공간 정보를 유지한 상태로 학습이 가능한 모델이 바로 CNN(Convolutional Neural Network)입니다.

CNN(Convolutional Neural Network)은 기존 Fully Connected Neural Network와 비교하여 다음과 같은 차별성을 갖습니다.

- 각 레이어의 입출력 데이터의 형상 유지
- 이미지의 공간 정보를 유지하면서 인접 이미지와의 특징을 효과적으로 인식
- 복수의 필터로 이미지의 특징 추출 및 학습
- 추출한 이미지의 특징을 모으고 강화하는 Pooling 레이어

- 필터를 공유 파라미터로 사용하기 때문에, 일반 인공 신경망과 비교하여 학습 파라미터가 매우 적음

CNN은 위 이미지와 같이 이미지의 특징을 추출하는 부분과 클래스를 분류하는 부분으로 나눌 수 있습니다. 특징 추출 영역은 Convolution Layer와 Pooling Layer를 여러 겹 쌓는 형태로 구성됩니다.[2] Convolution Layer는 입력 데이터에 필터를 적용 후 활성화 함수를 반영하는 필수 요소입니다. Convolutional Layer 다음에 위치하는 Pooling Layer는 선택적인 레이어입니다. CNN 마지막 부분에는 이미지 분류를 위한 Fully Connected 레이어가 추가됩니다. 이미지의 특징을 추출하는 부분과 이미지를 분류하는 부분 사이에 이미지 형태의 데이터를 배열 형태로 만드는 Flatten 레이어가 위치 합니다.
CNN은 이미지 특징 추출을 위하여 <그림1>과 같이 입력데이터를 필터가 순회하며 합성곱을 계산하고, 그 계산 결과를 이용하여 Feature map을 만듭니다. Convolution Layer는 Filter 크기, Stride, Padding 적용 여부, Max Pooling 크기에 따라서 출력 데이터의 Shape이 변경됩니다.

1. CNN의 주요 용어 정리

CNN에는 다음과 같은 용어들이 사용됩니다.

- Convolution(합성곱)
- 채널(Channel)
- 필터(Filter)
- 커널(Kernel)
- 스트라이드(Strid)
- 패딩(Padding)
- 피처 맵(Feature Map)
- 액티베이션 맵(Activation Map)
- 풀링(Pooling) 레이어

간략하게 각 용어에 대해서 살펴 보겠습니다.

1.1 합성곱, Convolution

합성곱, Convolution의 위키피디아 정의는 다음과 같습니다.

> *합성곱 연산은 두 함수 f, g 가운데 하나의 함수를 반전(reverse), 전이(shift)시킨 다음, 다른 하나의 함수와 곱한 결과를 적분하는 것을 의미한다. 출처: **https://ko.wikipedia.org/wiki/%ED%95%A9%EC%84%B1%EA%B3%B1***

위키피디아의 정의는 상당히 난해합니다. <그림 1>은 2차원 입력데이터(Shape: (5,5))를 1개의 필터로 합성곱 연산을 수행하는 과정을 소개합니다. 합성곱 처리 결과로 부터 Feature Map을 만듭니다.

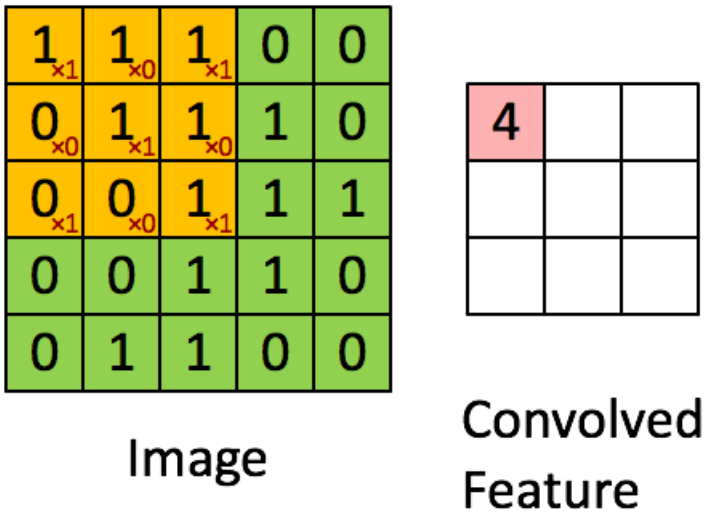
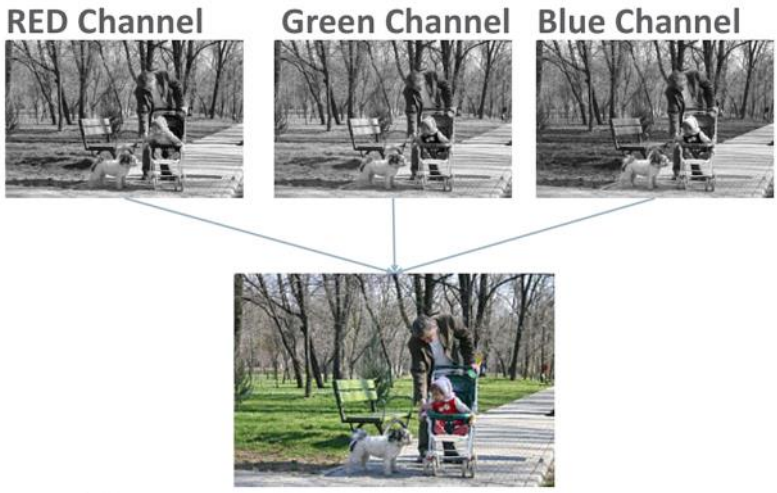


***그림 1**: 합성곱 처리 절차, 출처: http://deeplearning.stanford.edu/wiki/index.php/Feature_extraction_using_convolution*

1.2 채널, Channel

이미지 픽셀 하나하나는 실수입니다. 컬러 사진은 천연색을 표현하기 위해서, 각 픽셀을 RGB 3개의 실수로 표현한 3차원 데이터입니다.(<그림 2> 참조) 컬러 이미지는 3개의 채널로 구성됩니다. 반면에 흑백 명암만을 표현하는 흑백 사진은 2차원 데이터로 1개 채널로 구성됩니다. 높이가 39 픽셀이고 폭이 31 픽셀인 컬러 사진 데이터의 shape은 (39, 31, 3)[3]으로 표현합니다. 반면에 높이가 39픽셀이고 폭이 31픽셀인 흑백 사진 데이터의 shape은 (39, 31, 1)입니다.



이미지 출처: https://en.wikipedia.org/wiki/Channel_(digital_image)

***그림 2**: 3개의 채널로 만들어진 컬러 사진*

Convolution Layer에 유입되는 입력 데이터에는 한 개 이상의 필터가 적용됩니다. 1개 필터는 Feature Map의 채널이 됩니다. Convolution Layer에 **n개**의 필터가 적용된다면 출력 데이터는 **n개**의 채널을 갖게 됩니다.

1.3 필터(Filter) & Stride

필터는 이미지의 특징을 찾아내기 위한 공용 파라미터입니다. Filter를 Kernel이라고 하기도 합니다. CNN에서 Filter와 Kernel은 같은 의미입니다. 필터는 일반적으로 (4, 4)이나 (3, 3)과 같은 정사각 행렬로 정의됩니다. CNN에서 학습의 대상은 필터 파라미터 입니다. <그림 1>과 같이 입력 데이터를 지정된 간격으로 순회하며 채널별로 합성곱을 하고 모든 채널(컬러의 경우 3개)의 합성곱의 합을 Feature Map로 만듭니다. 필터는 지정된 간격으로 이동하면서 전체 입력데이터와 합성곱하여 Feature Map을 만듭니다. <그림 3>은 채널이 1개인 입력 데이터를 (3, 3) 크기의 필터로 합성곱하는 과정을 설명합니다.

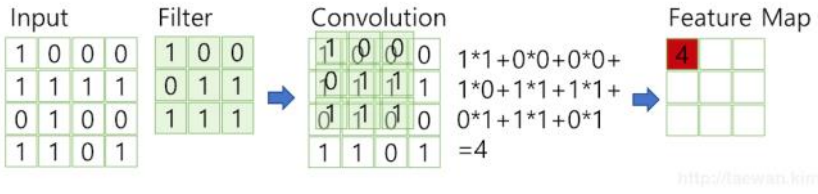


***그림 3**: 합성곱 계산 절차*

필터는 입력 데이터를 지정한 간격으로 순회하면서 합성곱을 계산합니다. 여기서 지정된 간격으로 필터를 순회하는 간격을 Stride라고 합니다. <그림 4>는 strid가 1로 필터를 입력 데이터에 순회하는 예제입니다. strid가 2로 설정되면 필터는 2칸씩 이동하면서 합성곱을 계산합니다.

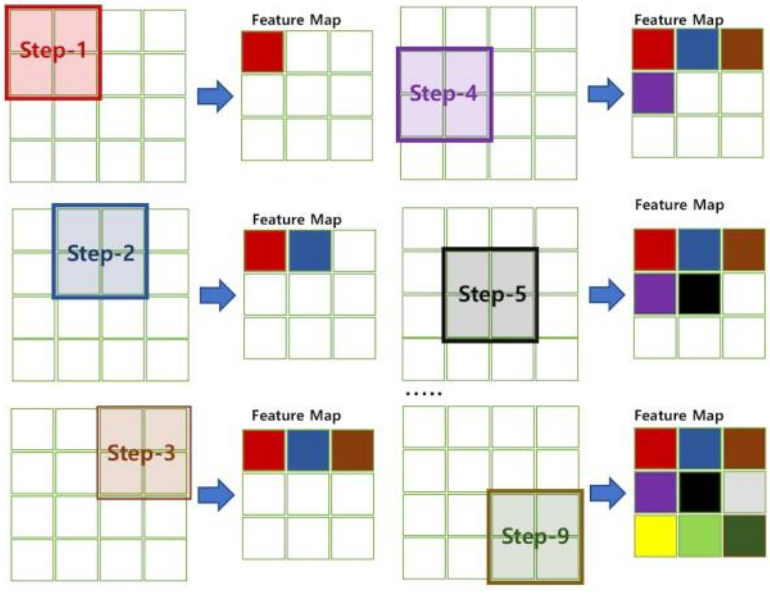


***그림 4**: Feature Map 과정*

입력 데이터가 여러 채널을 갖을 경우 필터는 각 채널을 순회하며 합성곱을 계산한 후, 채널별 피처 맵을 만듭니다. 그리고 각 채널의 피처 맵을 합산하여 최종 피처 맵으로 반환합니다. 입력 데이터는 채널 수와 상관없이 필터 별로 1개의 피처 맵이 만들어 집니다. <그림 5 참조>

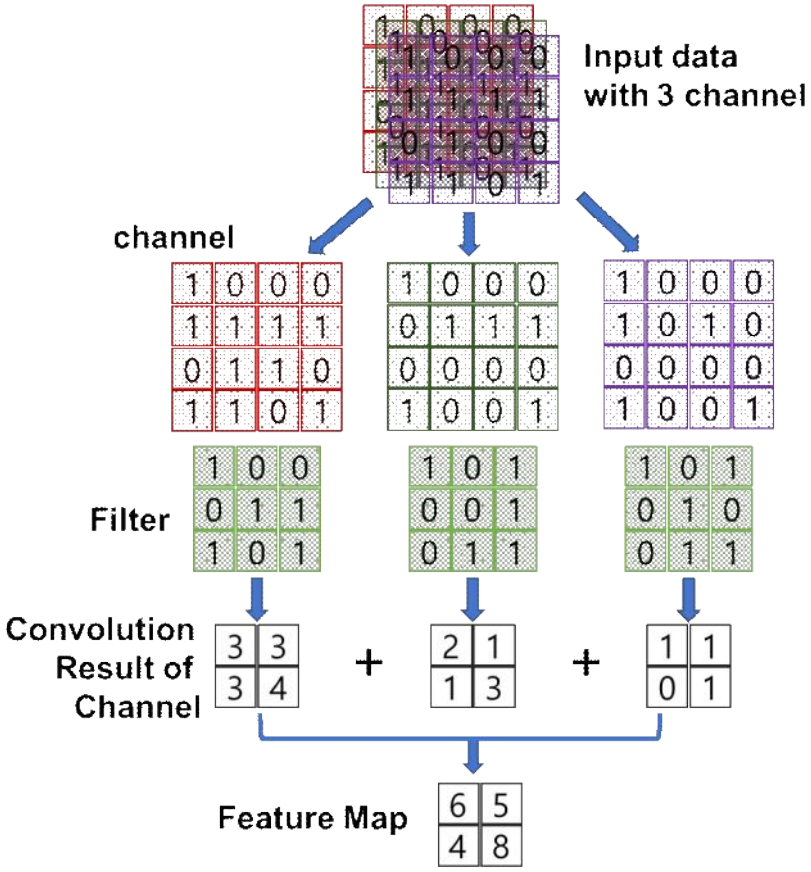


그림 5: 멀티 채널 입력 데이터에 필터를 적용한 합성곱 계산 절차

하나의 Convolution Layer에 크기가 같은 여러 개의 필터를 적용할 수 있습니다. 이 경우에 Feature Map에는 필터 갯수 만큼의 채널이 만들어집니다. 입력데이터에 적용한 필터의 개수는 출력 데이터인 Feature Map의 채널이 됩니다.
Convolution Layer의 입력 데이터를 필터가 순회하며 합성곱을 통해서 만든 출력을 Feature Map 또는 Activation Map이라고 합니다. Feature Map은 합성곱 계산으로 만들어진 행렬입니다. Activation Map은 Feature Map 행렬에 활성 함수를 적용한 결과입니다. 즉 Convolution 레이어의 최종 출력 결과가 Activation Map입니다.

1.4 패딩(Padding)

Convolution 레이어에서 Filter와 Stride에 작용으로 Feature Map 크기는 입력데이터 보다 작습니다. Convolution 레이어의 출력 데이터가 줄어드는 것을 방지하는 방법이 패딩입니다. 패딩은 입력 데이터의 외각에 지정된 픽셀만큼 특정 값으로 채워 넣는 것을 의미합니다. 보통 패딩 값으로 0으로 채워 넣습니다.

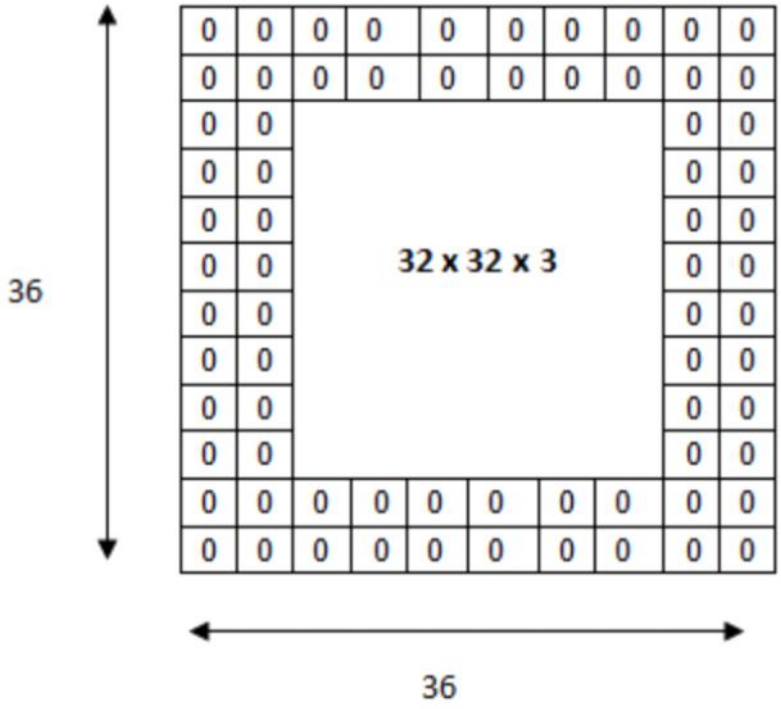


그림 6: padding 예제: 2pixel 추가

<그림 6>은 (32, 32, 3) 데이터를 외각에 2 pixel을 추가하여 (36, 36, 3) 행렬을 만드는 예제입니다. Padding을 통해서 Convolution 레이어의 출력 데이터의 사이즈를 조절하는 기능이 외에, 외각을 "0"값으로 둘러싸는 특징으로 부터 인공 신경망이 이미지의 외각을 인식하는 학습 효과도 있습니다.

1.5 Pooling 레이어

풀링 레이어는 컨볼류션 레이어의 출력 데이터를 입력으로 받아서 출력 데이터(Activation Map)의 크기를 줄이거나 특정 데이터를 강조하는 용도로 사용됩니다. 플링 레이어를 처리하는 방법으로는 Max Pooling과 Average Pooning, Min Pooling이 있습니다. 정사각 행렬의 특정 영역 안에 값의 최댓값을 모으거나 특정 영역의 평균을 구하는 방식으로 동작합니다. <그림 7>은 Max pooling과 Average Pooling의 동작 방식을 설명합니다. 일반적으로 Pooing 크기와 Stride를 같은 크기로 설정하여 모든 원소가 한 번씩 처리 되도록 설정합니다.

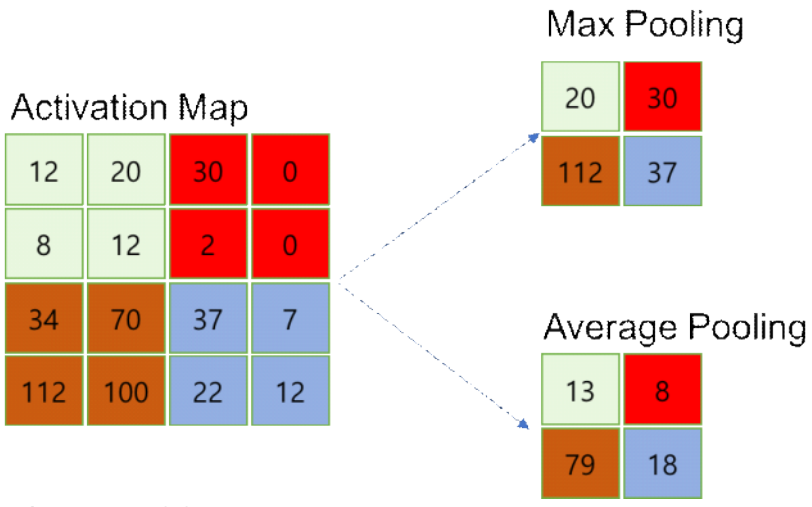


그림 7: Pooling 예제: Max Pooling, Average Pooling

Pooling 레이어는 Convolution 레이어와 비교하여 다음과 같은 특징이 있습니다.

- 학습대상 파라미터가 없음
- Pooling 레이어를 통과하면 행렬의 크기 감소
- Pooling 레이어를 통해서 채널 수 변경 없음

CNN에서는 주로 Max Pooling을 사용합니다.

2. 레이어별 출력 데이터 산정

Convloution 레이어와 Pooling 레이버의 출력 데이터 크기를 계산하는 방법을 소개합니다.

2.1 Convolution 레이어 출력 데이터 크기 산정

입력 데이터에 대한 필터의 크기와 Stride 크기에 따라서 Feature Map 크기가 결정됩니다. 공식은 다음과 같습니다.

- 입력 데이터 높이: H
- 입력 데이터 폭: W
- 필터 높이: FH
- 필터 폭: FW
- Strid 크기: S
- 패딩 사이즈: P

    - *식 1. 출력 데이터 크기 계산*OutputHeight=OH=(H+2P−FH)S+1OutputWeight=OW=(W+2P−FW)S+1OutputHeight=OH=(H+2P−FH)S+1OutputWeight=OW=(W+2P−FW)S+1

<식 1>의 결과는 자연수가 되어야 합니다. 또한 Convolution 레이어 다음에 Pooling 레이어가 온다면, Feature Map의 행과 열 크기는 Pooling 크기의 배수여야 합니다. 만약 Pooling 사이즈가 (3, 3)이라면 위 식의 결과는 자연수이고 3의 배수여야 합니다. 이 조건을 만족하도록 Filter의 크기, Stride의 간격, Pooling 크기 및 패딩 크기를 조절해야 합니다.

2.2 Pooling 레이어 출력 데이터 크기 산정

Pooling 레이어에서 일반적인 Pooling 사이즈를 정사각형입니다. Pooling 사이즈를 Stride 같은 크기로 만들어서, 모든 요소가 한번씩 Pooling되도록 만듭니다. 입력 데이터의 행 크기와 열 크기는 Pooling 사이즈의 배수(나누어 떨어지는 수)여야 합니다. 결과적으로 Pooling 레이어의 출력 데이터의 크기는 행과 열의 크기를 Pooling 사이즈로 나눈 몫입니다. Pooling 크기가 (2, 2) 라면 출력 데이터 크기는 입력 데이터의 행과 열 크기를 2로 나눈 몫입니다. pooling 크기가 (3, 3)이라면 입력데이터의 행과 크기를 3으로 나눈 몫이 됩니다. <식 2 참조>

- *식 2. 출력 데이터 크기 계산*
  OutputRowSize=InputRowSizePoolingSizeOutputColumnSize=InputColumnSizePoolingSizeOutputRowSize=InputRowSizePoolingSizeOutputColumnSize=InputColumnSizePoolingSize

3. CNN 구성

<그림 8>은 전형적인 CNN 구성입니다. CNN은 Convolution Layer와 Max Pooling 레이어를 반복적으로 stack을 쌓는 특징 추출(Feature Extraction) 부분과 Fully Connected Layer를 구성하고 마지막 출력층에 Softmax를 적용한 분류 부분으로 나뉩니다.

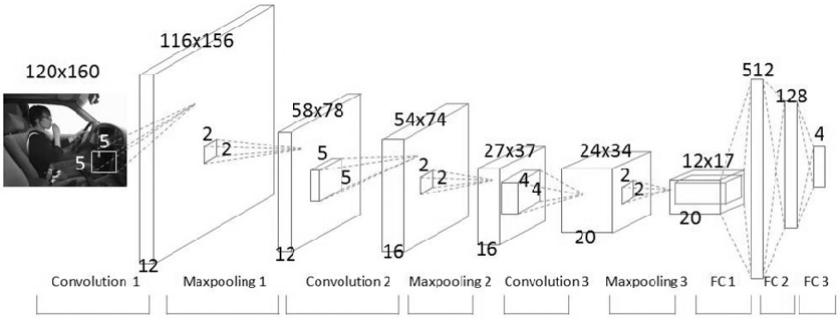


***그림 8**: 전형적인 CNN*

CNN을 구성하면서 Filter, Stride, Padding을 조절하여 특징 추출(Feature Extraction) 부분의 입력과 출력 크기를 계산하고 맞추는 작업이 중요합니다. <코드 1>은 <그림 8>을 Keras로 CNN 모델로 구현한 코드입니다.

코드 1. CNN 모델 예제 코드 (Keras)

```
from keras.models import Sequential
from keras.layers.convolutional import Conv2D
from keras.layers.convolutional import MaxPooling2D
from keras.layers import Dense
from keras.layers import Flatten
model = Sequential()
model.add(Conv2D(12, kernel_size=(5, 5), activation='relu', input_shape=(120, 60, 1)))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(16, kernel_size=(5, 5), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(20, kernel_size=(4, 4), activation='relu'))
```

```
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dense(4, activation='softmax'))
```

4. CNN 입출력, 파리미터 계산

다음과 조건과 같은 이미지를 학습하는 CNN의 각 레이어별 입력 데이터와 출력 데이터의 Shape을 계산해 보고 네트워크가 학습시키는 파라미터의 개수를 계산해 보겠습니다. 예제로 사용할 CNN 모델 정보는 다음과 같습니다.

- 입력데이터 Shape: (39, 31, 1)
- 분류 클래스: 100

layer	Filter	Stride	Max Pooling	activation function
Convolution Layer 1	(4, 4, 20)	1	X	relu
Max Pooling Lyaer 1	X	2	(2, 2)	X
Convolution Layer 2	(3, 3, 40)	1	X	relu
Max Pooling Lyaer 2	X	2	(2, 2)	X
Convolution Layer 3	(3, 3, 60)	1	1	relu
Max Pooling Lyaer 3	X	2	(2, 2)	X
Convolution Layer 4	(2, 2, 80)	1	1	relu
Flatten	X	X	X	X
fully connected Layer	X	X	X	softmax

4.1 Layer 1의 Shape과 파라미터

Layer 1은 1개의 Convolution Layer와 1개의 Pooling Layer로 구성됩니다. 두 레이어의 출력 데이터 shape과 파라미터는 다음과 같이 계산할 수 있습니다.

4.1.1 Convolution Layer 1

Convolution Layer 1의 기본 정보는 다음과 같습니다.

- 입력 데이터 Shape = (39, 31, 1)
- 필터=(4, 4, 20)
- Stride = 1

입력 이미지에 Shape이 (4, 4)인 필터 20개를 적용할 경우에, 출력 데이터(Activation Map)의 Shape을 계산하는 과정은 <식 3>과 같습니다.

  - *식 3. Convolution Layer 1의 Activation Map 크기 계산*RowSize=N−FStrid+1=39−41+1=36ColumnSize=N−FStrid+1=31−41+1=28RowSize=N−FStrid+1=39−41+1=36ColumnSize=N−FStrid+1=31−41+1=28

<식 3>로 계산된 출력 데이터(Activation Map)의 Shape은 (36, 28, 20) 입니다. Convolution Layer 1에서 학습시킬 대상은 (4, 4) 필터 20개 입니다. 따라서 이 레이어의 학습 파라미터는 320개 (4X4X20) 입니다.

- 출력 데이터(Activation Map) Shape: (36, 28, 20)
- 학습 파라미터: 320개 (4X4X20)

4.1.2 Max Pooling Layer 1

Max Pooling Layer 1의 입력 데이터의 Shape은 (36, 28, 20)입니다. Max Pooling 크기가 (2, 2)이기 때문에 출력 데이터 크기는 <식 4>와 같이 계산될 수 있습니다.

  - *식 4. Max Pooling Layer 1의 출력 데이터 크기 계산*RowSize=362=18ColumnSize=282=14RowSize=362=18ColumnSize=282=14

<식 4>으로 계산된 출력 데이터의 Shape은 (18, 14, 20) 입니다. Max Pooling Layer에서 학습 파라미터가 없습니다.

- 출력 데이터 Shape: (18, 14, 20)
- 학습 파라미터: 0

4.2 Layer 2의 Shape과 파라미터

Layer 2는 1개의 Convolution Layer와 1개의 Pooling Layer로 구성됩니다. 두 레이어이 출력 데이터 shape과 파라미터는 다음과 같이 계산할 수 있습니다.

4.2.1 Convolution Layer 2

Convolution Layer 2의 기본 정보는 다음과 같습니다.

- 입력 데이터 Shape = (18, 14, 20)
- 필터=(3, 3, 40)
- Stride = 1

입력 이미지에 Shape이 (3, 3)인 필터 40개를 적용할 경우 출력 데이터(Activation Map)의 Shape을 계산하는 과정은 <식 5>과 같습니다.

  - *식 5. Convolution Layer 2의 Activation Map 크기 계산*RowSize=N−FStrid+1=18−31+1=16ColumnSize=N−FStrid+1=14−31+1=12RowSize=N−FStrid+1=18−31+1=16ColumnSize=N−FStrid+1=14−31+1=12

<식 5>로 계산된 출력 데이터(Activation Map)의 Shape은 (16, 12, 40)입니다. Convolution Layer 1에서 학습시킬 대상은 (3, 3) 필터 40개입니다. 따라서 이 레이어의 학습 파라미터는 360개 (3X3X40) 입니다.

- 출력 데이터(Activation Map) Shape: (16, 12, 40)
- 학습 파라미터: 360개 (3X3X40)

4.2.2 Max Pooling Layer 2

Max Pooling Layer 2의 입력 데이터의 Shape은 (16, 12, 40)입니다. Max Pooling 크기가 (2, 2)이기 때문에 출력 데이터 크기는 <식 6>와 같이 계산될 수 있습니다.

  - *식 6. Max Pooling Layer 2의 출력 데이터 크기 계산*RowSize=162=8ColumnSize=122=6RowSize=162=8ColumnSize=122=6

<식 6>로 계산된 출력 데이터(Activation Map)의 Shape은 (8, 6, 40) 입니다. Max Pooling Layer에서 학습 파라미터가 없습니다.

- 출력 데이터 Shape: (8, 6, 40)
- 학습 파라미터: 0

4.3 Layer 3의 Shape과 파라미터

Layer 3도 1개의 Convolution Layer와 1개의 Pooling Layer로 구성됩니다. 두 레이어이 출력 데이터 shape과 파라미터는 다음과 같이 계산할 수 있습니다.

4.3.1 Convolution Layer 3

Convolution Layer 3의 기본 정보는 다음과 같습니다.

- 입력 데이터 Shape = (8, 6, 40)
- 필터=(2, 2, 60)
- Stride = 1

입력 이미지에 Shape이 (3, 3)인 필터 60개를 적용할 경우 출력 데이터(Activation Map)의 Shape을 계산하는 과정은 <식 7>과 같습니다.

  - *식 7. Convolution Layer 3의 Activation Map 크기 계산*RowSize=N−FStrid+1=8−21+1=6ColumnSize=N−FStrid+1=6−21+1=4RowSize=N−FStrid+1=8−21+1=6ColumnSize=N−FStrid+1=6−21+1=4

<식 7>로 계산된 출력 데이터(Activation Map)의 Shape은 (6, 4, 60)입니다. Convolution Layer 1에서 학습시킬 대상은 (3, 3) 필터 60개입니다. 따라서 이 레이어의 학습 파라미터는 540개 (3X3X60) 입니다.

- 출력 데이터(Activation Map) Shape: (6, 4, 60)
- 학습 파라미터: 540개 (3X3X60)

4.3.2 Max Pooling Layer 3

Max Pooling Layer 3의 입력 데이터의 Shape은 (6, 4, 60)입니다. Max Pooling 크기가 (2, 2)이기 때문에 출력 데이터 크기는 <식 8>과 같이 계산될 수 있습니다.

  - *식 8. Max Pooling Layer 3의 출력 데이터 크기 계산*RowSize=62=3ColumnSize=42=2RowSize=62=3ColumnSize=42=2

<식 8>로 계산된 출력 데이터(Activation Map)의 Shape은 (3, 2, 60)입니다. Max Pooling Layer에서 학습 파라미터가 없습니다.

- 출력 데이터 Shape: (3, 2, 60)
- 학습 파라미터: 0

4.4 Layer 4의 Shape과 파라미터

Layer 3도 1개의 Convolution Layer로 구성됩니다. 이 레이어의 출력 데이터 shape과 파라미터는 다음과 같이 계산할 수 있습니다.

4.4.1 Convolution Layer 4

Convolution Layer 4의 기본 정보는 다음과 같습니다.

- 입력 데이터 Shape = (3, 2, 60)
- 필터=(2, 2, 80)
- Stride = 1

입력 이미지에 Shape이 (2, 2)인 필터 80개를 적용할 경우 출력 데이터(Activation Map)의 Shape을 계산하는 과정은 <식 9>과 같습니다.

- *식 9. Convolution Layer 3의 Activation Map 크기 계산*RowSize=N−FStrid+1=3−21+1=2ColumnSize=N−FStrid+1=2−21+1=1RowSize=N−FStrid+1=3−21+1=2ColumnSize=N−FStrid+1=2−21+1=1

<식 9>로 계산된 출력 데이터(Activation Map)의 Shape은 (2, 1, 80)입니다. Convolution Layer 1에서 학습시킬 대상은 (2, 2) 필터 80개입니다. 따라서 이 레이어의 학습 파라미터는 320개 (2X2X80)입니다.

- 출력 데이터(Activation Map) Shape: (2, 1, 80)
- 학습 파라미터: 320개 (2X2X80)

4.5 Flatten Layer의 Shape

Flatten Layer는 CNN의 데이터 타입을 Fully Connected Neural Network의 형태로 변경하는 레이어입니다. Flatten 레이어에는 파라미터가 존재하지 않고, 입력 데이터의 Shape 변경만 수행합니다.

- 입력 데이터 Shape =(2, 1, 80)
- 출력 데이터 Shape =(160, 1)

4.6 Softmax Layer

이 레이어의 입력 데이터 Shape은 (160, 1)입니다. 이 네트워크의 분류 클래스가 100개이기 때문에 최종 데이터의 Shape은 (100, 1)입니다.

- 입력 데이터의 shape: (160, 1)
- 출력 데이터의 shape: (100, 1)

이때 Weight Shape은 (100, 160)입니다. Softmax 레이어이 파라미터는 160,000개 (100X160)입니다.

4.7 전체 파라미터 수와 레이어별 Input/Output 요약

layer	Filter	Stride	Pooling	활성함수	Input Shape	Output Shape	파라미터 수
Convolution Layer 1	(4, 4, 20)	1	X	relu	(39, 31, 1)	(36, 28, 20)	320
Max Pooling Lyaer 1	X	2	(2, 2)	X	(36, 28, 20)	(18, 14, 20)	0
Convolution Layer 2	(3, 3, 40)	1	X	relu	(18, 14, 20)	(16, 12, 40)	360
Max Pooling Lyaer 2	X	2	(2, 2)	X	(16, 12, 40)	(8, 6, 40)	0
Convolution Layer 3	(3, 3, 60)	1	1	relu	(8, 6, 40)	(6, 4, 60)	540
Max Pooling Lyaer 3	X	2	(2, 2)	X	(6, 4, 60)	(3, 2, 60)	0
Convolution Layer 4	(2, 2, 80)	1	1	relu	(3, 2, 60)	(2, 1, 80)	320
Flatten	X	X	X	X	(2, 1, 80)	(160, 1)	0
fully connected Layer	X	X	X	softmax	(160, 1)	(100, 1)	160,000

이 CNN을 그림 9와 같이 표현할 수 있습니다.

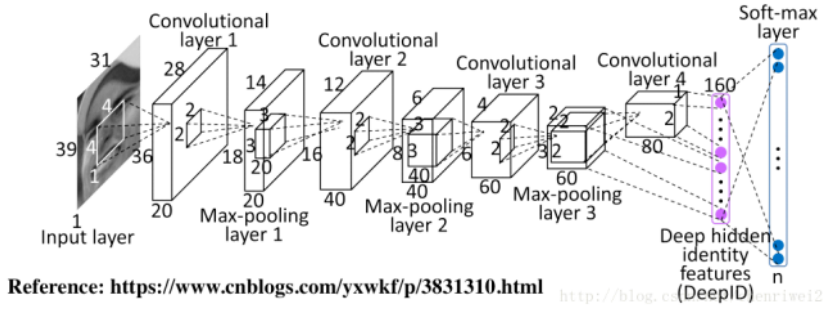


그림 9: 예제 CNN 이미지

5. CNN과 FC Neural Network 파라미터 비교

앞에서 다룬 CNN의 총 파라미터 수는 161,540개입니다. 이 CNN과 유사한 4개의 은닉 레이어를 갖는 FC(Fully Connected) Neural Network을 <그림 10>과 같이 만들 수 있습니다.

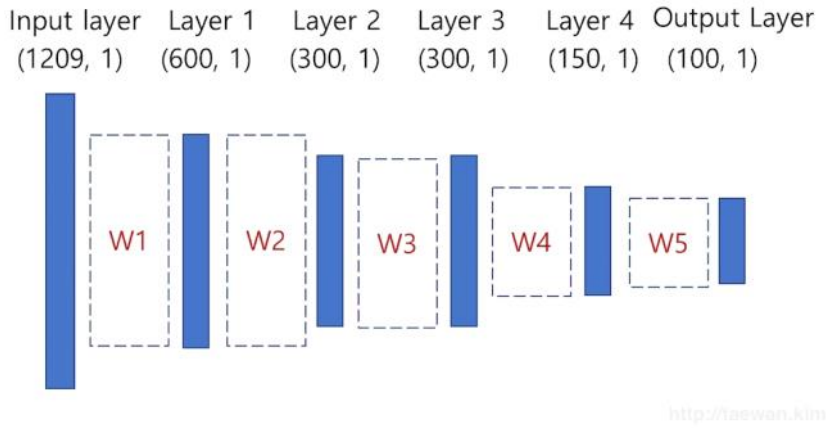


***그림 10**: FC 신경망 - 비교*

<그림 10>의 신경망을 계산하면 파라미터는 다음과 같이 계산 가능합니다.

레이어	입력 노드	출력 노드	Weight Shape	파라미터 수
Layer 1	1209	600	(1209,600)	725,400
Layer 2	600	300	(600,300)	180,000
Layer 3	300	300	(300,300)	90,000
Layer 4	300	150	(300,150)	45,000
Output	150	100	(150,100)	15,000
합계				1,055,400

<그림 10> 인공 신경망의 총 파라미터는 백만 개가 넘습니다. 예제 CNN 파라미터와 비교하면 10배 이상의 학습 파라미터를 갖습니다. 은닉층을 더 깊게 만들 경우 Fully Connected Neural Network과 CNN과의 학습 파라미터의 차이는 더 급격하게 늘어납니다.
CNN은 Fully Connected Neural Network과 비교하여 다음과 같은 특징을 갖습니다.

- CNN은 학습 파라미터 수가 매우 작음
- 학습 파라미터가 작고, 학습이 쉽고 네트워크 처리 속도가 빠름

6. 요약

CNN(Convolutional Neural Network)은 이미지의 공간 정보를 유지하면서 인접 이미지와의 특징을 효과적으로 인식하고 강조하는 방식으로 이미지의 특징을 추출하는 부분과 이미지를 분류하는 부분으로 구성됩니다. 특징 추출 영역은 Filter를 사용하여 공유 파라미터 수를 최소화하면서 이미지의 특징을 찾는 Convolution 레이어와 특징을 강화하고 모으는 Pooling 레이어로 구성됩니다.
CNN은 Filter의 크기, Stride, Padding과 Pooling 크기로 출력 데이터 크기를 조절하고, 필터의 개수로 출력 데이터의 채널을 결정합니다.
CNN는 같은 레이어 크기의 Fully Connected Neural Network와 비교해 볼 때, 학습 파라미터양은 10% 규모입니다. 은닉층이 깊어질수록 학습 파라미터의 차이는 더 벌어집니다. CNN은 Fully Connected Neural Network와 비교하여 더 작은 학습 파라미터로 너 높은 인식률을 제공합니다.