**3. CNN**

**3.1 CNN(Convolutional Neural Network)이란?**

CNN이란 말 그대로 'convolution'이라는 작업이 들어가는 Neural Network이다. CNN은 이미지를 인식하기 위해 패턴을 찾는 데 특히 유용하다. 데이터에서 직접 학습하고 패턴을 사용해 이미지를 분류한다. 즉, 특징을 수동으로 추출할 필요가 없는 것이다. 이러한 장점 덕분에 자율주행자동차, 얼굴 인식과 같은 객체 인식 등에 많이 사용된다.

**3.2 합성곱 층** (Convolutional Layer)

DNN의 가장 기본적인 구조는 완전연결 계층(Fully Connected Layer, FC layer)으로 전 layer와 후 layer가 모두 연결된 구조이다.

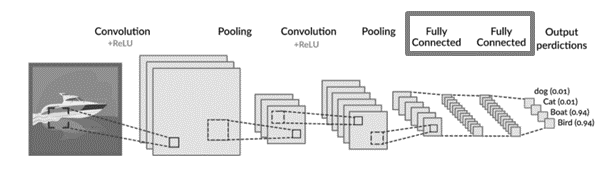


Figure 1 - Convolutional layer (ebb and flow’s tistory, Aug 19, 2020)

위의 그림 CNN 과정에서 노란색으로 표시된 부분이 Fully Connected 과정이며 이것은 CNN을 이용해 추출한 feature를 최종적으로 어떤 이미지인지 분류하기 위해서 연결되어 있는 layer이다. CNN은 위와 같이 마지막 부분에 FC layer를 이용하는 경우가 많다.

CNN에서의 합성곱 층이 아닌 완전연결 계층(FC layer)만을 이용해 MNIST(Mixed National Institute of Standards and Technology) 데이터셋(필기 숫자의 분류를 위한 학습 데이터 집합)을 분류하는 모델을 만든다고 가정하면, 각각 높이, 폭, 채널로 이루어진 3차원의 데이터(28, 28, 1)를 입력층으로 넣기 위해 평평한 데이터인 1차원으로 만드는 차원 축소 과정을 거쳐야 한다.

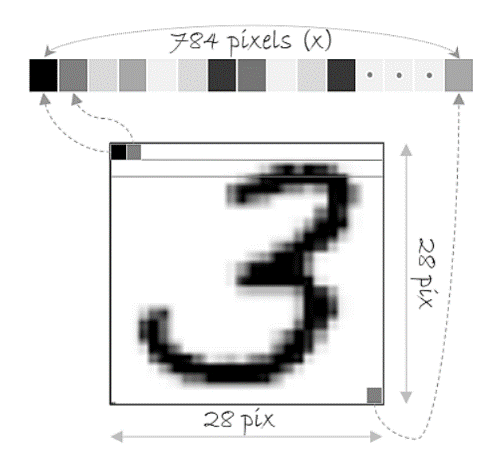


Figure 2 - 28 \* 28 pixel (Wikidocs, Aug 19, 2020)

28 x 28 x 1 픽셀의 3차원 데이터를 1차원으로 만들면 784개의 픽셀로 이루어진 데이터가 만들어진다. 하지만 차원 축소로 인해 데이터의 형상이 무시된다는 문제점이 생긴다. 3차원 데이터로 존재했을 때에는 공간적 구조를 가져 3차원 공간 정보를 가지지만 1차원일 때에는 그러한 정보가 사라지게 된다. 이와 같은 문제점을 해결하기 위해 CNN에서 완전연결 계층 대신 합성곱 층을 사용한다.

합성곱 층은 입력 데이터의 형상을 유지하기 때문에 3차원의 이미지를 입력층으로 입력 받고 3차원 데이터로 출력하여 다음 layer로 전달한다. 따라서 CNN은 이미지 데이터와 같이 형상을 가지고 있는 데이터를 다른 신경망보다 더 정확하게 학습한다.

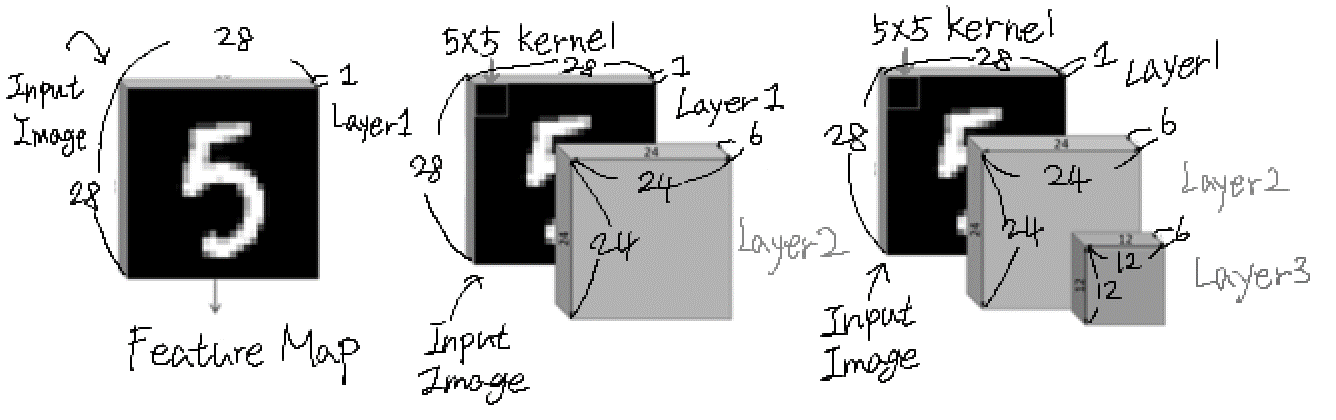


Figure 3 - Convolutional layer (Excelsior-JH’s tistory, Aug 19, 2020)

합성곱 층에서는 완전연결 신경망과 같이 각 유닛(뉴런)이 앞 계층의 모든 유닛과 연결되어 있지 않고 각각의 유닛은 이전 계층에서 근접해 있는 몇 개의 유닛들에만 연결이 된다. 즉 합성곱 층은 입력 이미지의 모든 픽셀과 연결되는 것이 아니라 수용영역(receptive field) 이라고도 불리는 필터(Filter) 안에 있는 픽셀에만 연결되어 있다. 또한 모든 유닛은 이전 계층에 동일한 방법으로 연결되어 있어 같은 값의 가중치와 구조를 공유한다. 그리고 그 연결 사이에 합성곱 연산이 있고 그 신경망을 합성곱 신경망이라고 한다.

**합성곱 층 연산**

합성곱 연산은 필터를 일정한 간격으로 이동해가며 계산한다.

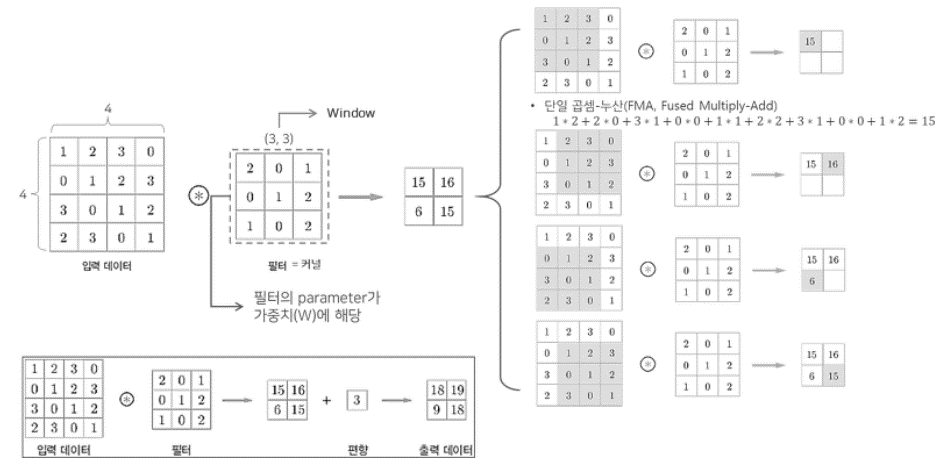


Figure 4 - Fused Multiply-Add (FMA) (Excelsior-JH's tistory, Aug 19, 2020)

위의 그림처럼 합성곱 연산은 입력 데이터와 필터간에 서로 대응하는 원소끼리 곱한 후 총합을 구하는 것이고 이를 Fused Multiply-Add(FMA) 라고 한다. 편향(bias)은 필터를 적용한 후에 더해주어 계산한다.

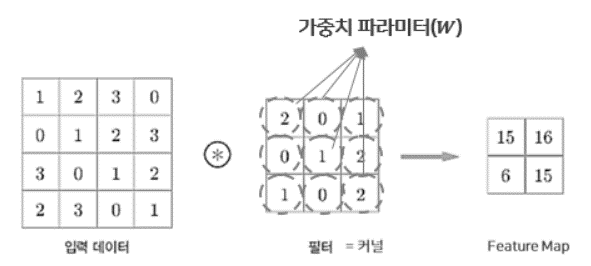
**3.3 필터**

Figure 5 - CNN Filter (Excelsior-JH's tistory, Aug 19, 2020)

수용영역을 합성곱 층에서 필터(Filter) 또는 커널(Kernal) 또는 윈도우(window)라고 하며 필터는 그 특징이

데이터에 존재하는지 검출해주는 함수이다.

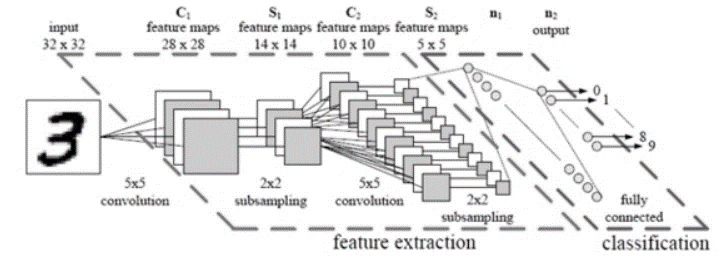
필터는 합성곱 층에서 가중치 파라미터인 w에 해당한다. 학습 단계에서 적절한 필터를 찾도록 학습되며 합성곱 층에서 입력 데이터에 필터를 적용하여 필터와 유사한 이미지의 영역을 강조하는 특성맵(Feature Map)을 출력하여 다음 층으로 전달한다.

Figure 6 - Convolutional layer (안재홍, CNN 기반, Multi-class 이미지 분류 기술, Aug 19, 2020)

정리하면 입력 데이터를 필터로 합성곱 연산을 하여 데이터의 특징(feature)을 추출하고 이러한 합성곱 layer를 여러 계층으로 연결한다. 합성곱 layer을 지날 때마다 저수준의 특징들이 점차 고수준의 특징들로 만들어진다. 그런 후에 마지막에는 완전연결 계층으로 최종 결과를 학습한다.

**3.4 Channel**

우리가 흔히 알고 있는 색의 3원색인 RGB는 Red, Green, Blue이다. 이를 이미지의 채널에도 적용할 수 있는데, 우리가 생각하는 컬러 이미지는 Red채널, Green채널, Blue채널로 이루어져 있다. 즉, 컬러 사진인 경우, 각 픽셀을 RGB 3개의 실수로 표현한 3차원 데이터이고 3개의 채널로 구성되며, 흑백 사진의 경우에는 2차원 데이터로 1개의 채널로 구성된다. 보통 연산량(오차)을 줄이기 위해 전처리에서 이미지를 흑백으로 만들어 처리한다. 만약 높이가 30픽셀이고 폭이 20픽셀인 컬러 사진 데이터의 shape은 (30, 20, 3)으로 표현하고, 흑백인 경우엔 (30, 20, 1)로 표현한다.

**3.5 Pooling**

이미지의 크기를 계속 유지하며 **FC(Fully Connected) layer**로 가게 된다면 연산량은 기하급수적으로 늘어날 것이다. 적당히 크기도 줄이고, 특징을 강조할 수 있어야 하는데 그 역할을 하는 것이 **Pooling**이다.

Pooling은 쉽게 convolution layer를 거쳐서 나온 activation maps가 있을 때, 이를 이루는 convolution layer를 resizing하여 새로운 layer를 얻어 특정 데이터를 강조하는 용도로 사용된다.

또는, matrix연산을 사용하지 않고 각 픽셀에서 하나의 값을 뽑아내는 과정으로도 이해할 수 있다.

Pooling의 처리방법으로는 최댓값을 뽑아내는 max pooling, 평균값을 뽑아내는 mean pooing, 최솟값을 뽑아내는 Min pooling이 있는데 CNN에서는 주로 **Max Pooling**을 사용한다.

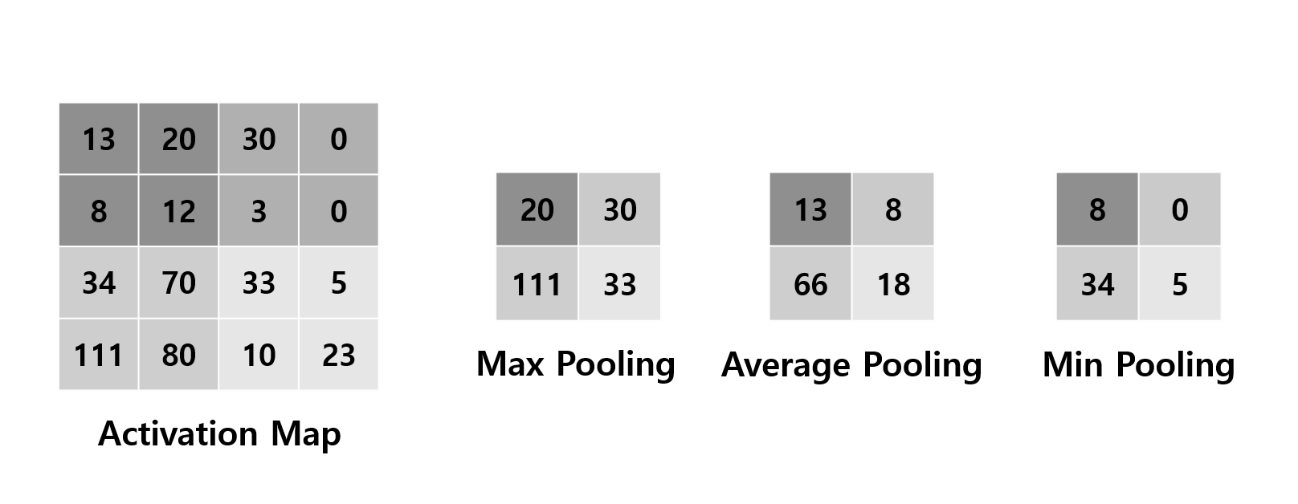
일반적으로 pooling의 크기와 stride의 크기를 같게 설정하여 모든 원소가 한 번씩은 처리되도록 한다.

Figure 7 - Pooling (GRU's github, Aug 19, 2020)

**3.6 완전 연결형 레이어 (Fully Connected layer)**

Fully Connected layer (FC layer) 는 완전 연결형 레이어로 합성곱 층의 마지막에 존재한다. FC layer는 앞의 Convolution, Relu, Pooling의 과정을 반복하면서 나온, 가장 마지막 activation map들의 pooling까지 끝났을 때, 각각의 클래스마다 점수를 도출해 내기 위해 그 이미지를 입력으로 받는다. 마지막 Conv layer의 출력은 3차원 volume으로 이루어진다. 이를 전부 펴서 1차원 벡터로 만들어 FC layer의 입력으로 사용한다. 전부 다 하나로 통합시키고, 최종적인 추론을 하는 것이다.

FC layer는 일반적인 input layer, hidden layer, output layer를 가지는 부분이다. 이 부분은 실제로 분류를 해주기 위해 존재한다. CNN은 보통 이미지 분류와 같은 영역에서 자주 사용되는데, 이때 주어진 이미지를 분류해야 하므로 FC layer를 이용해 실제 분류를 해내는 것이다. 즉, FC layer의 앞에 있는 Conv layer는 주어진 이미지 데이터에 대해 N개의 필터를 이용해 올바른 특징을 뽑아내는 역할을 하고, FC layer는 그 특징들을 기반으로 실제 분류를 실행한다.

**3.7 CNN의 Parameter**

**- Convolution filter의 개수**

각 layer에서의 연산시간을 비교적 일정하게 유지하며 시스템의 균형을 맞추는 것이 좋다. 보통 Pooling layer를 거치면 출력이 1/4로 줄어들기 때문에 convolution layer의 결과인 feature map의 개수를 4배 정도 증가시키면 된다.

**- Filter의 Size**

작은 filter를 여러 개 중첩하면 원하는 특징을 더 부각하며 연산량을 줄일 수 있다.

**- Padding 여부와 Padding Size**

Padding을 사용하게 되면 입력 이미지의 크기를 줄이지 않을 수 있다. 아래에 관련 정보를 추가했다.

**- Stride의 간격**

stride는 filter의 이동 간격을 조절하는 매개변수이다. 이 값이 커지면 결과 데이터의 크기가 작아지게 된다.

**- Pooling의 크기**

데이터의 크기를 줄이며 특징을 강조해준다.

(Convolution layer 다음에 Pooling layer가 온다면, Feature map의 행과 열 크기는 Pooling 크기의 배수여야 한다. 이 조건을 만족하도록 CNN의 Parameter를 조절해야 한다.)

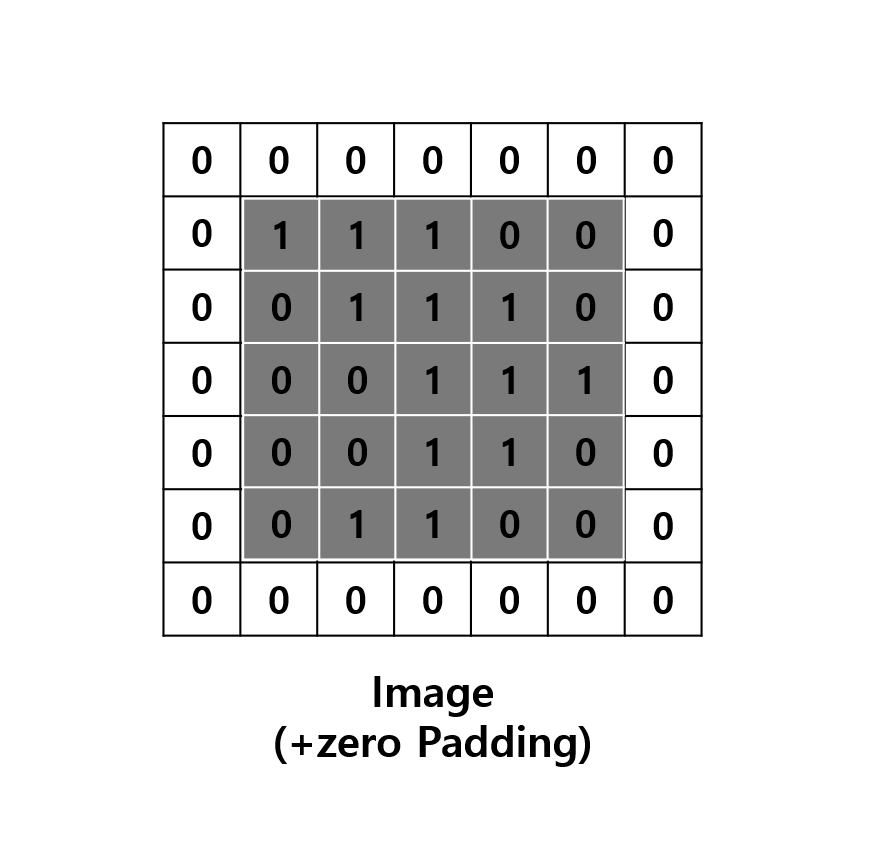
**Padding**

Figure 8 - Padding (GRU's github, Aug 19, 2020)

Padding은 Convolution을 수행하기 전, 입력 데이터(이미지) 주변을 특정 픽셀 값(보통은 0)으로 채워 늘리는 것이다. Padding을 사용하게 되면 입력 이미지의 크기를 줄이지 않을 수 있다.

Convolution에서는 filter와 stride의 작용으로 convolution layer는 입력 데이터(이미지)보다 작아지게 된다. 이렇게 입력 데이터보다 출력 데이터의 값이 작아지는 것을 방지하는 방법이 바로 Padding이다. Padding을 하면 convolution을 해도 크기가 작아지지 않는다.

**3.8 Convolution layer 출력 데이터 크기**

 입력 데이터 높이: H  입력 데이터 폭: W

 필터 높이: FH  필터 폭: FW

 Stride 크기: S  Padding 크기: P

위 식의 결과는 자연수가 되어야 한다. 또한, Convolution layer 다음에 Pooling layer가 온다면, Feature map의 행과 열 크기는 Pooling 크기의 배수여야 한다. 예를 들어, 만일 Pooling의 사이즈가 (3, 3)이라면 위 식의 결과는 자연수이고 3의 배수여야 한다. 이를 만족하도록 Parameter (filter의 크기, stride의 간격, pooling의 크기 및 패딩 크기)를 조절하여야 한다.

**4. 정리 및 고찰**

딥러닝 알고리즘 중 이미지 분류에 특화된 CNN을 이용하여 CIFAR-10 데이터 셋을 분류했고, 코드는 b스카이비전의 웹사이트를 참고했습니다. CNN은 3차원 데이터를 입력하면 3차원으로 출력하기 때문에 더 정확하게 학습할 수 있다는 장점을 가지고 있어 이미지 분류에 사용하였습니다. 3차원 데이터를 입력하면 커널(필터)를 이용하여 Convolution 과정과 Max Pooling 과정을 차례로 반복하며 데이터의 feature를 추출하였고 여기에서 데이터 출력을 위해 relu 라는 활성화 함수를 이용하였습니다. 마지막 과정으로 완전연결 계층을 이용해 최종적으로 어떤 이미지인지 분류했고 여기서 softmax 활성화 함수를 이용하였습니다. 이렇게 학습된 모델을 이용해 이미지의 class를 예측하고 정확도를 나타내는 모델을 만들 수 있었습니다.

CIFAR-10으로 인공지능 모델을 구현했을 때, 정확도가 약 70% ~ 80%였습니다. 정확도를 더 늘리기 위해서는 학습률이나 레이어 층의 개수, 해당 레이어의 노드 개수 등의 하이퍼 파라미터를 조정해야 할 필요가 있었습니다. 레이어를 깊이 만드는 것도 정확도를 높일 수 있는 요인으로 작용하나, 기울기 손실 문제 혹은 가중치 초기값 문제를 다루어야 합니다. 또한, 레이어가 깊을수록 모델이 구현되는 시간도 기하급수적으로 늘어나기 때문에 GPU환경의 컴퓨팅이 필요합니다. 즉, 인공지능 구현에 있어서 하드웨어적인 부분도 필수적입니다. 향후 하이퍼 파라미터를 조정하는 강화학습과 CNN이외에도 RNN, LSTM등의 딥러닝을 더 연구할 것이며, 이미지 분류뿐만 아니라 이미지 인식, 이미지 분할 등과 같이 더 심화된 컴퓨터 비전을 설계할 것입니다.

이번 학술지 작성을 통해 CNN에 관한 전반적인 지식을 새롭게 얻고, 복습할 수 있었으며, 여러 프로그램을 보고 직접 만들며 인공지능 분야에 한 발짝 다가갈 수 있었습니다. 처음엔 어려워 보여서 조금 걱정이 됐지만, 기초부터 차근차근 공부하며 이해하니 재미를 붙일 수 있었습니다. 비슷한 생각을 갖고 계신 분들이 이 학술지를 보고 조금이나마 이미지분류에 관심을 갖고 공부하셨으면 좋겠다는 생각을 갖고, 최대한 이해하기 쉽도록 작성하려 노력했습니다. 이미지 분류, CNN을 처음 공부하시는 분들께 이 글이 도움이 된다면 좋겠습니다.