CNN (Convolutional Neural Network)

- CNN 사용 이유
기존 다층 신경망 : 기본적으로 1차원 형태의 데이터를 사용
따라서, 이미지가 입력될 경우 이것을 1차원 벡터로 변환하게 되는데 이 과정에서 이미지의 공간적 정보가 손실됨
-> 이를 해결하기 위해 CNN 사용

입력값은 matrix로 표현된 이미지
목적 : 입력값 이미지의 모든 영역에 같은 필터를 반복 적용해 패턴을 찾아 처리하는 것
특징 추출 신경망이 깊을수록 인식 성능이 좋아짐

- Filter
Convolution Layer에 존재
이미지의 특징을 찾아내기 위한 공용 파라미터
입력된 이미지를 지정된 간격으로 순회
픽셀이 흑백일 경우 2차원이지만, 입력값 이미지가 3차원인 경우도 존재 (ex. 컬러 이미지)
Channel : 색 성분 / 각 픽셀을 RGB 3개의 실수로 표현

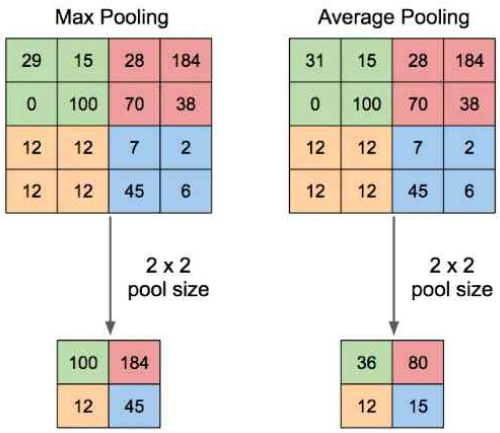
- Stride
필터를 얼마만큼 움직여 주는가
1이 기본값이고 1보다 큰 값이 될 수도 있음
stride 값이 커질 경우 필터가 이미지를 건너뛰는 칸이 커짐
-> 결과값 이미지의 크기는 작아짐(= 정보가 손실됨)

- Padding
Convolution 처리를 하면서 손실되는 부분이 발생하는데 이를 해결하기 위해 0으로 구성된 테두리를 이미지 가장자리에 채워넣음 (입력값과 결과값의 크기가 동일하게 유지되게 해줌)

CNN의 구조
- Convolution Layer
패턴 분석을 담당 (이를 위해 쓰이는 도구 : Filter)
입력된 이미지를 대상으로 여러개의 필터를 사용하여 결과값(새로운 이미지, feature map)을 얻음
이 후, Filter에서 도출한 결과값에 활성함수를 적용할 수 있음(ex. ReLU function)
-> Convolution Layer = Convolution 처리와 활성함수로 구성
+) 왜 활성함수를 쓰는가? 데이터 손실을 방지하기 위해

- Pooling Layer
convolution 과정을 통해 많은 수의 결과값을 생성 -> 한 개의 이미지에서 너무 많은 결과값이 도출됨, Pooling은 overfitting을 막기 위한 과정
Pooling : 각 영역에서 특정값 하나를 뽑아오는 것

대표적으로 Max Pooling, Average Pooling이 있음
ex) Pool의 크기가 2x2인 경우 2x2 크기의 matrix에서 max나 average 값을 가져와 결과값의 크기를 반으로 줄임



- 평탄화
이전 레이어의 출력을 일자 형태의 데이터로 펼쳐주는 과정
1차원 데이터로 변형해도 상관없는 이유 : pooling layer에서 얻어낸 이미지들은 입력된 이미지에서 얻어온 특이점 데이터가 됨, 따라서 1차원 벡터 데이터로 변형시켜도 무관한 상태

- Fully Connected Layer
입력된 이미지가 무엇인지 softmax 함수를 통해 분류

CNN의 특징
- Locality : 근접한 픽셀끼리 종속성을 가짐
- Shared Weights : topology 변화와 관계 없이 항상성을 얻을 수 있음