세미나(5)

컨볼루션 신경망

정보통계학과 이현섭

전달 내용

1 컨볼루션 신경망이란?

2 컨볼루션 신경망의 구조 및 학습

3 향상을 위한 알고리즘

컨볼루션 신경망이란?

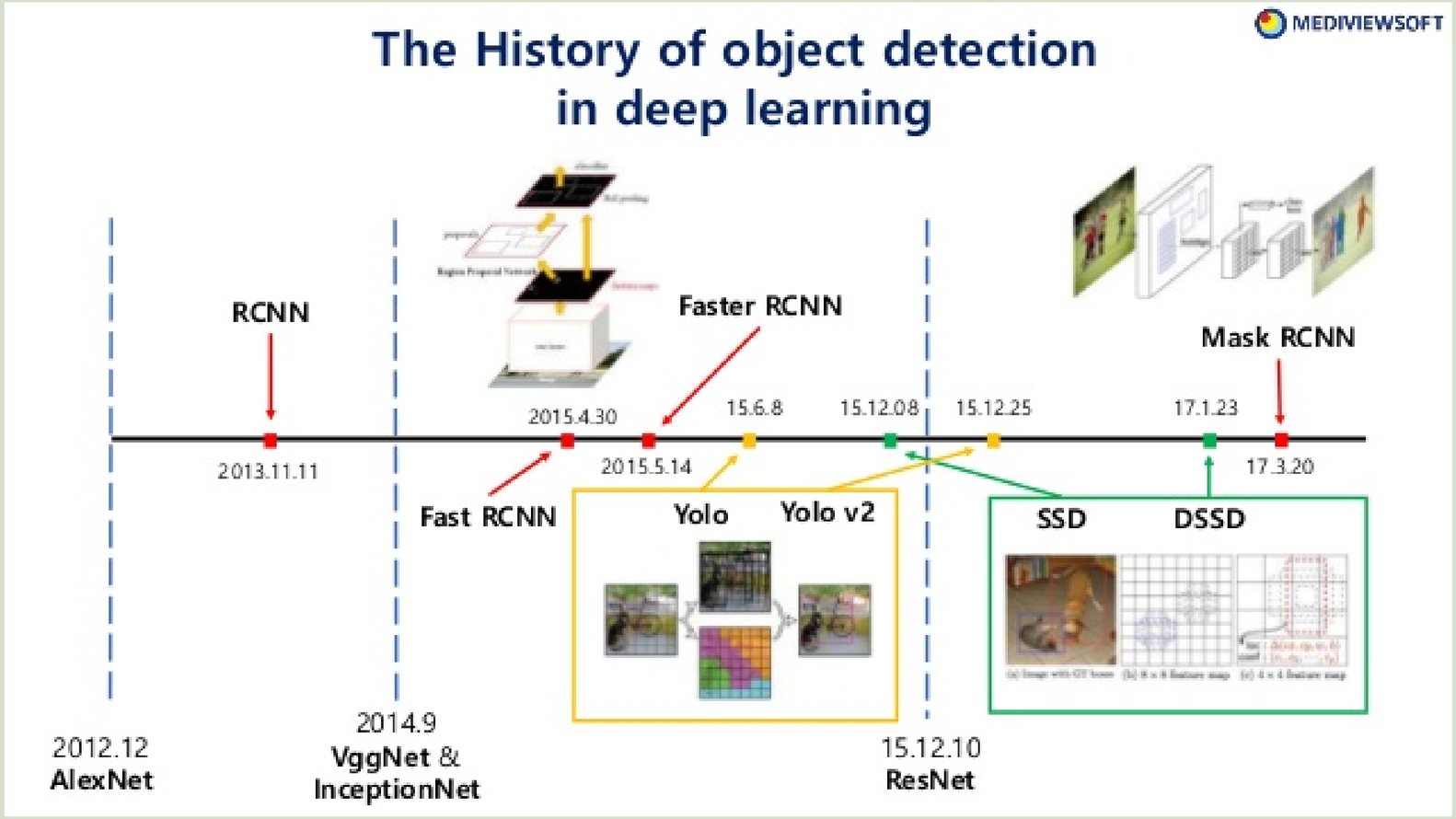
CNN 발상

0	0	0	0
0	1	0	0
1	1	0	0
0	0	0	0

0	0	0	0
0	0	1	1
0	0	0	1
0	0	0	0

- 왼쪽의 그림을 회전하여 오른쪽의 그림처럼 만들어도 삼각형을 검출할 수 있음.
- 이를 만약에 1차원으로 Flatten한다면? 삼각형의 모양은 사라질 것.(다층 퍼셉트론에 사용하기 위해)
- 이는 화소의 연결성을 쓸 수 없어 개별 화소를 보고 분류할 수 밖에 없음.(모폴로지 연산)
- 따라서 DNN으로 cifar-10 데이터셋의 정확도를 파악할 때 낮았던 이유.

CNN의 역사



- 1998년 수표를 인식하는 실용 시스템 제작에 활용한 사례가 발표됨.
- 2012년 ILSVRC 대회의 자연 영상 인식에서 AlexNet이 우승을 차지하며 널리 알려짐.
- 이후 여러 대회를 거쳐서 나온 많은 VGGNet, ResNet 등 여러 컨볼루션 신경망이 나타남.

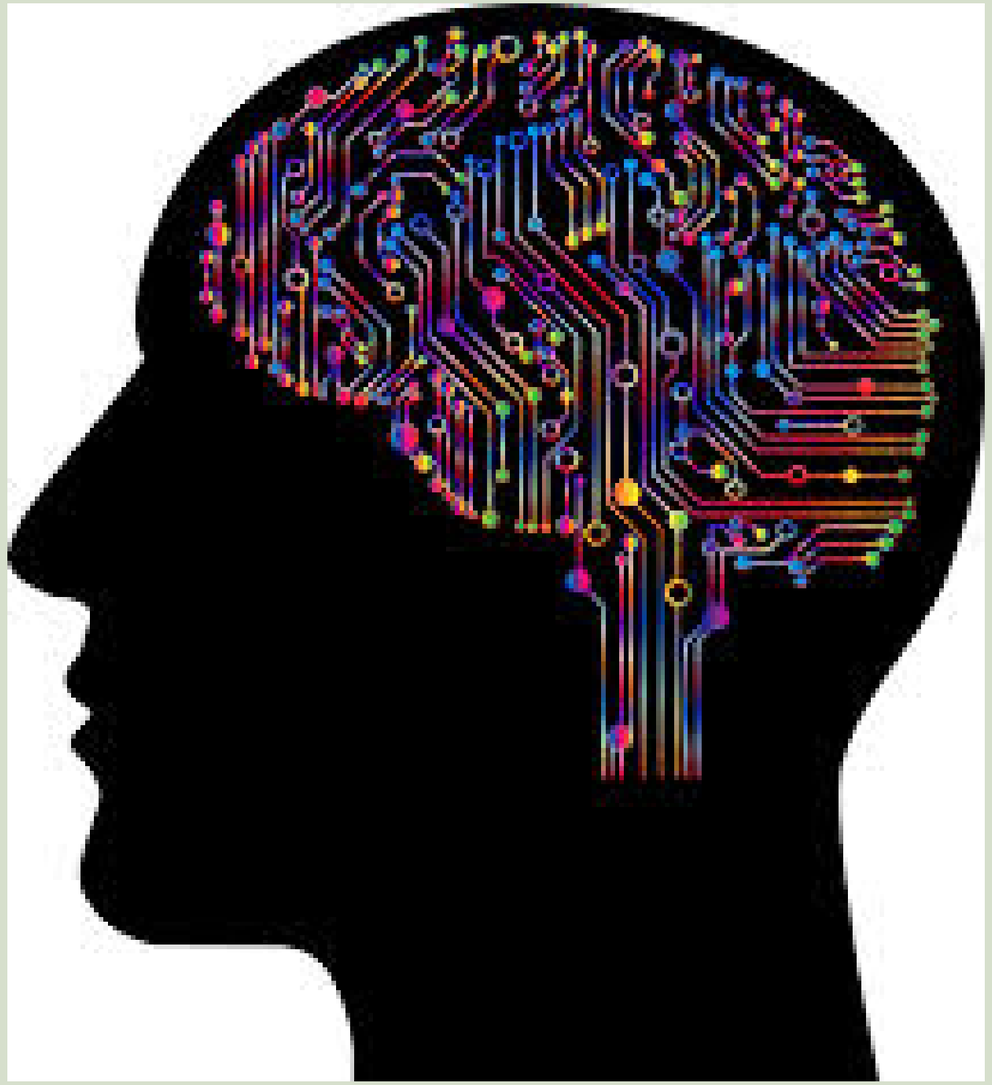
CNN의 성공 요인

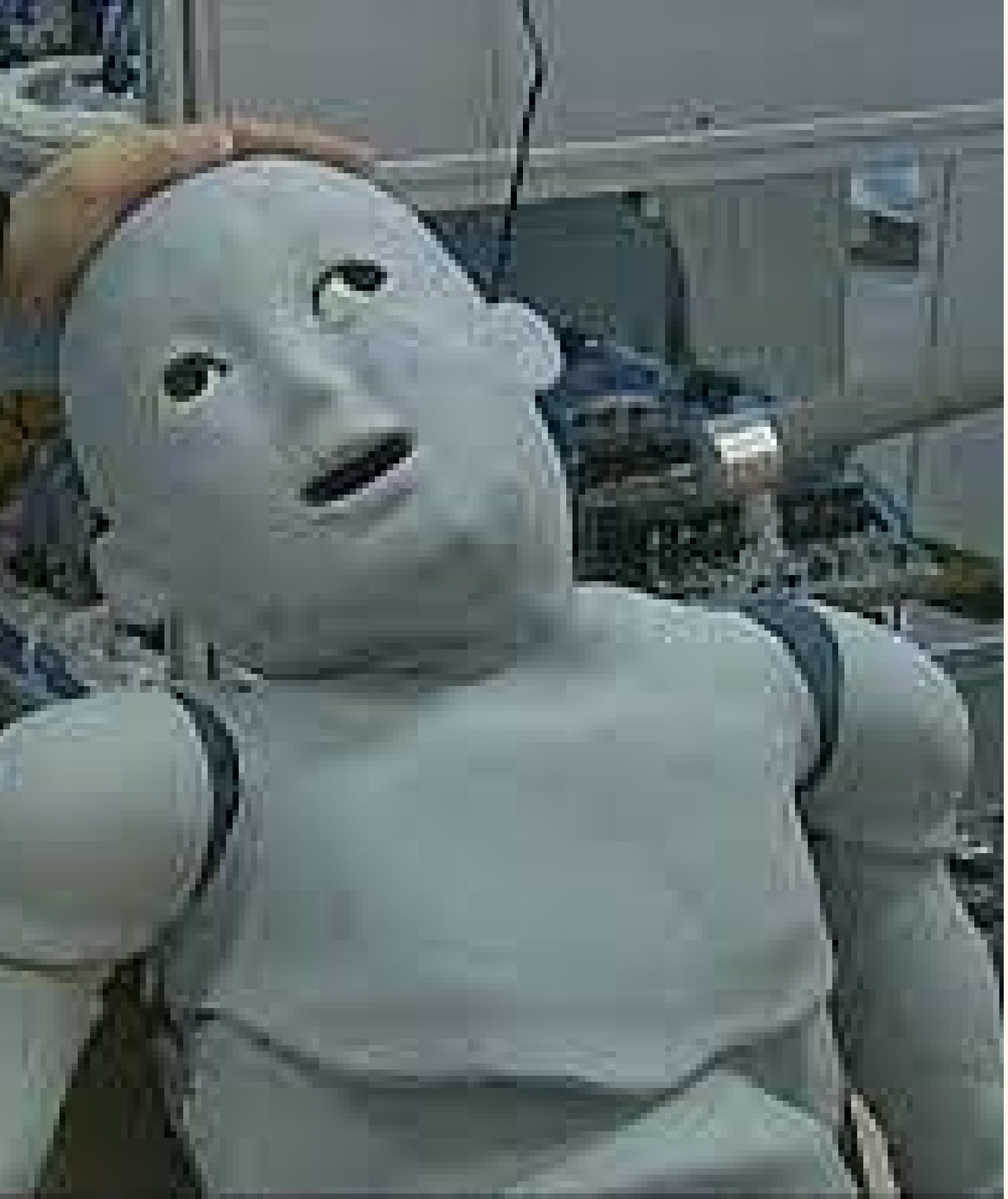
- 데이터셋이 커졌다. 인터넷의 발달로 인해 데이터셋 수집이 용이해졌고 확기적으로 방대한 데이터셋이 생겼다.(ImageNet은 초기 발전에 가장 큰 공헌을 한 데이터셋)
- GPU의 병렬 처리로 학습 시간이 대폭 상승.

신경망이 깊어지고 데이터셋이 커지면 학습 시간이 길어지지만 GPU를 사용 시 학습 시간이 10~100배 빨라진다.

- 좋은 학습 알고리즘이 개발되었다.

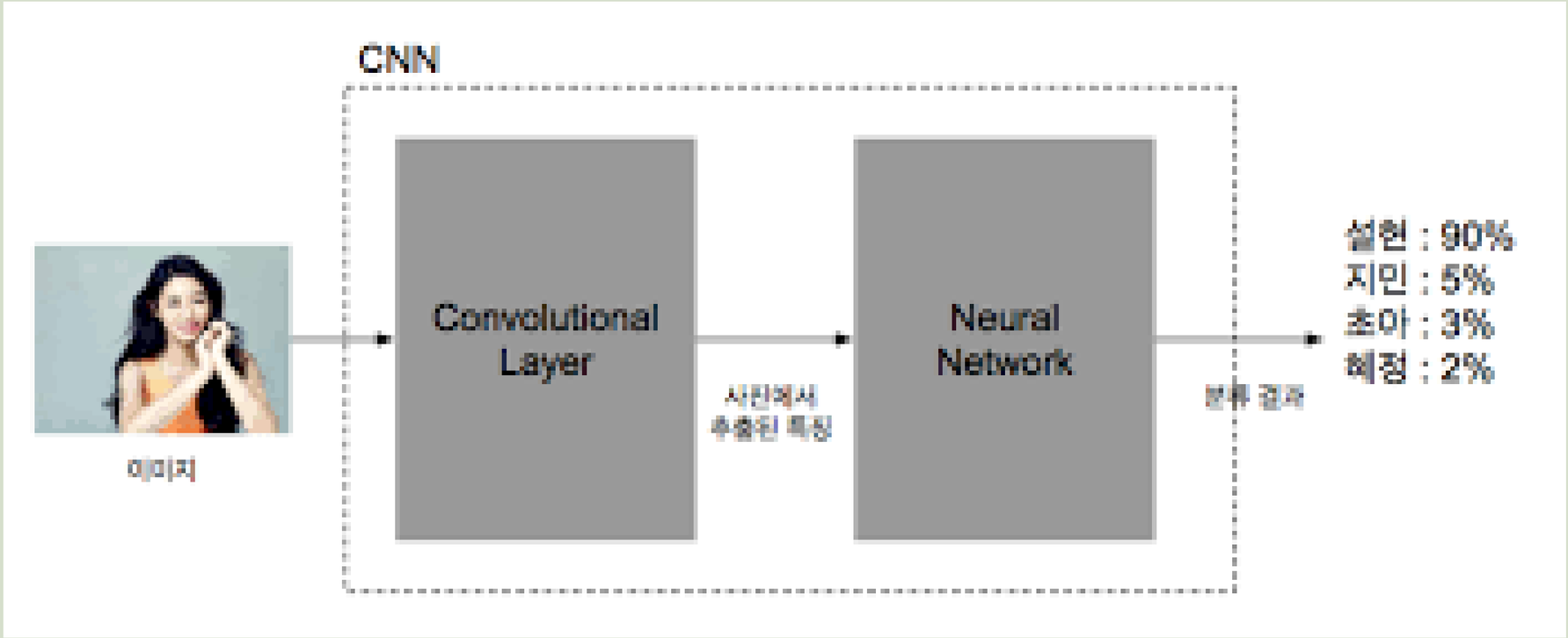
활성 함수와 규제기법. 손실 함수와 옵티마이저의 발전이 요인이다.





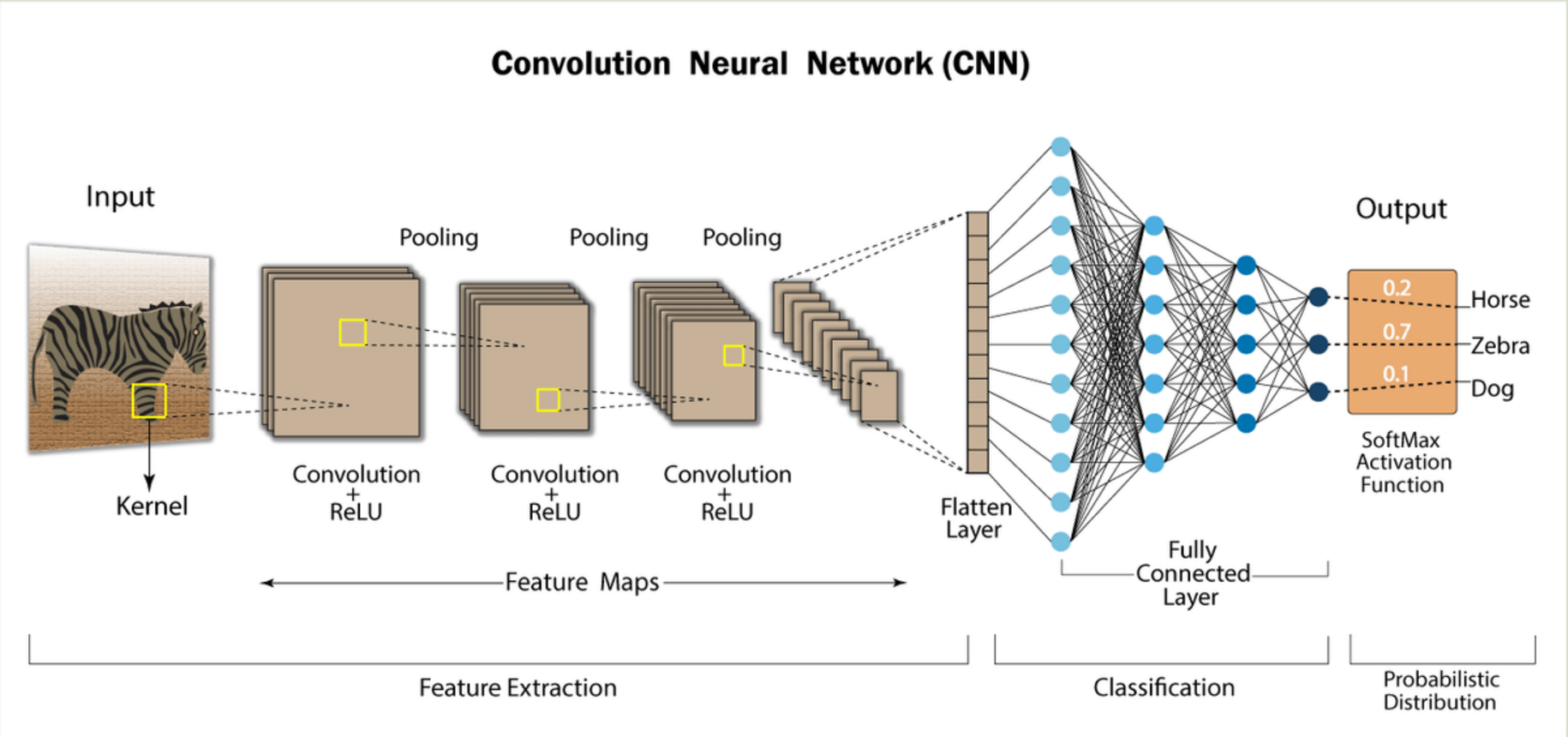
컨볼루션 신경망의 구조 및 학습

CNN의 구조



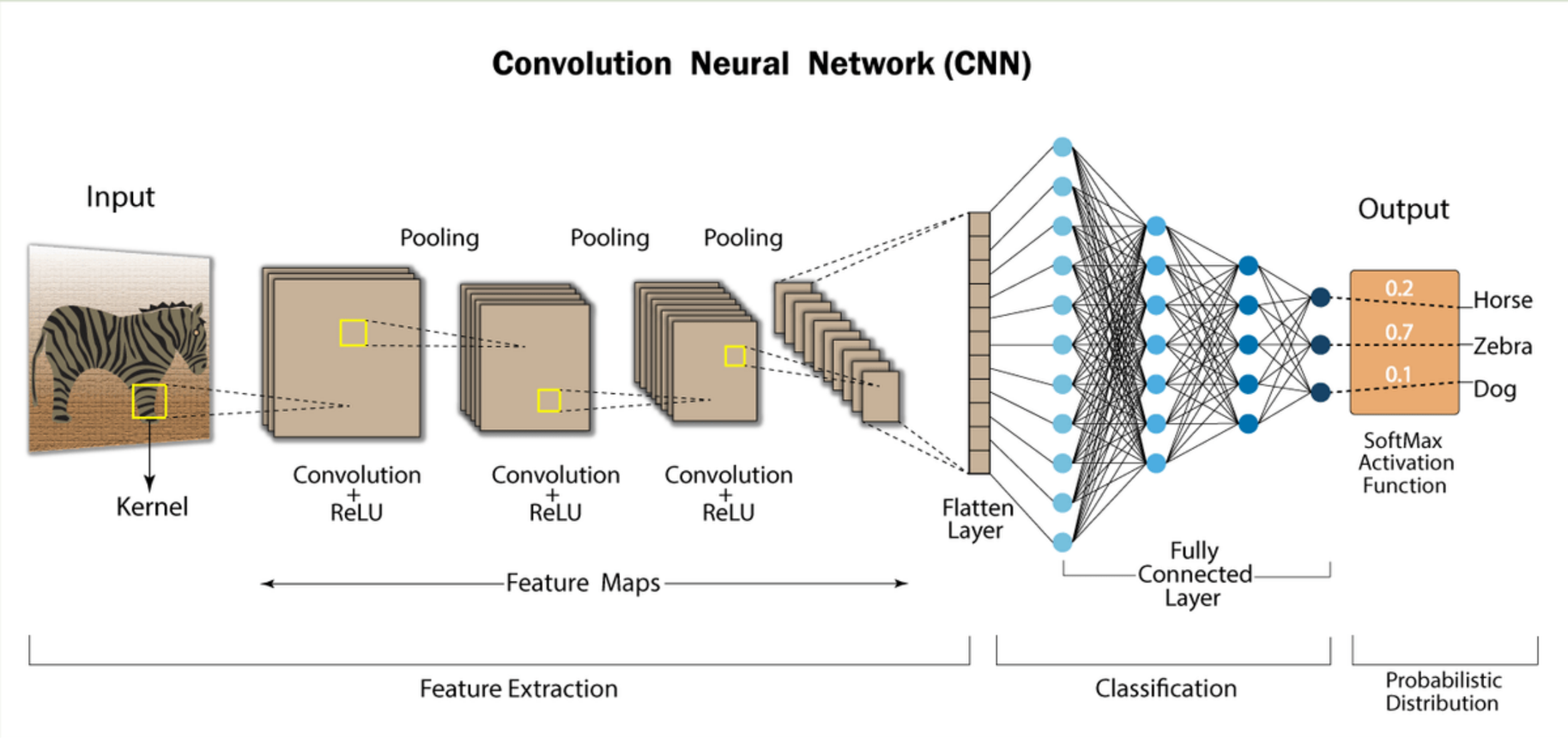
- 간단한 설명으로 CNN은 기본적인 Neural Network 앞에 여러 계층의 Convolution Layer을 붙인 형태와 같음.
- 컨볼루션 층 : 특징 추출을 위한 신경망
- 입력 영상의 형상을 직접 모델링하지 않아도 CNN이 특징 추출과 분류 역할을 모두 자동으로 이루어내는 것.

CNN의 구조



- 컨볼루션 층은 흔히 컨볼루션이라는 수학적 연산 + Relu라는 활성화 함수에 의해 이루어짐.
- 풀링(Pooling)층은 생략되기도 하지만 Max Pooling이나 Average pooling이 많이 사용됨.
- 완전 연결 층은 Input layer와 Hidden layer, Output layer로 이루어진 신경망

CNN의 구조

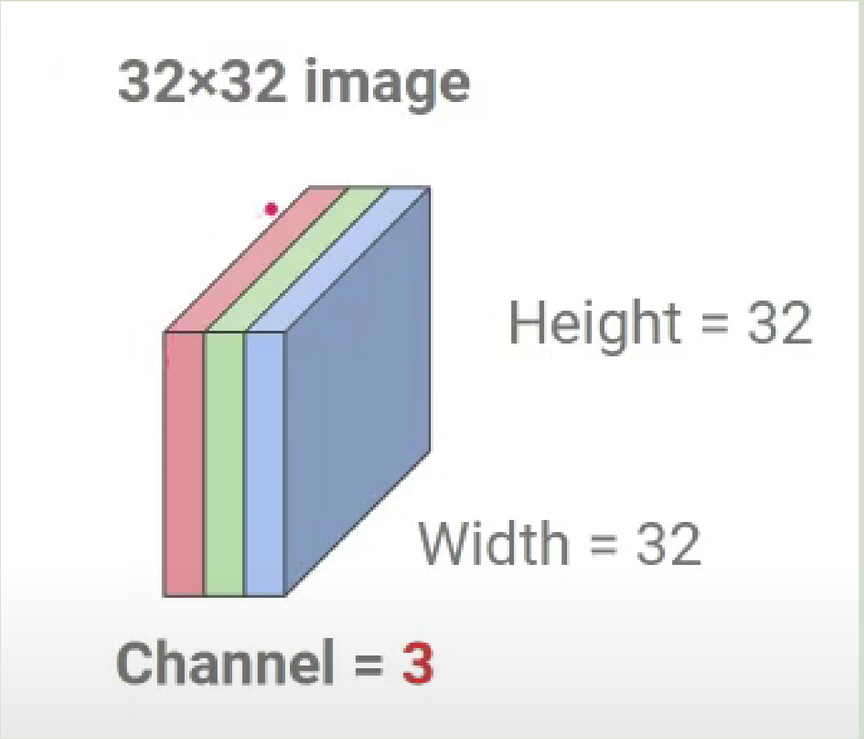


- 특징 추출을 위해서 2부분(컨볼루션 + 활성화 함수), 분류를 위한 3부분(Flatten, 완전 연결, 소프트 맥스)로 이루어져 있다.
- 입력 영상인 동물 영상에 대해 특징 추출을 먼저 진행.(컨볼루션 층과 풀링 층을 여러 번에 걸쳐 진행)
- 완전 연결 층은 1차원 데이터를 처리하기에 Flatten을 거친다.
- 완전 연결 층의 출력층에서 Softmax 함수를 사용하여 최종 분류(확률, 0~1)

컨볼루션 층

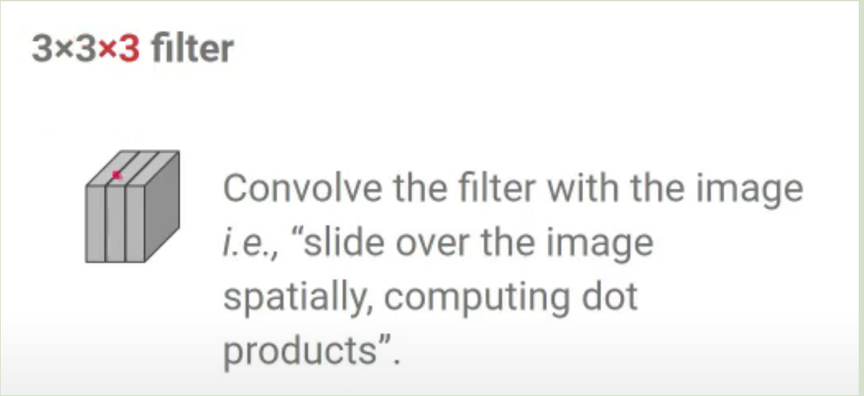
• 컨볼루션 층은 m x n x k 모양의 3차원 텐서
(k개 채널로 구성되어 있는 m x n 맵)

• 필터의 깊이는 입력 특징 맵과 같이 k
(크기는 h x h, 보통 3 or 5를 사용)

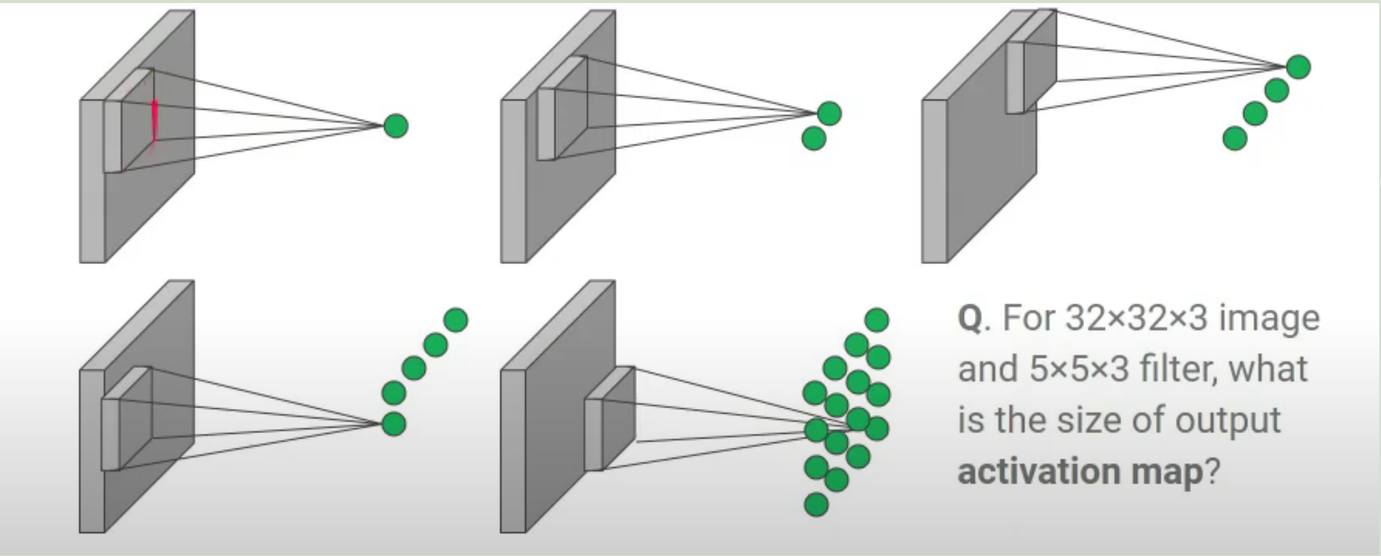


• Ex. 3*3 필터를 넣는다고 가정하였을 때,
사실 3*3 필터가 아닌 3*3*3 필터라는 점을 고려.

• 또한 파라미터는 (m x n x k)+1(bias)로 계산되는 점.
(bias가 있어야 제대로 계산이 가능!)



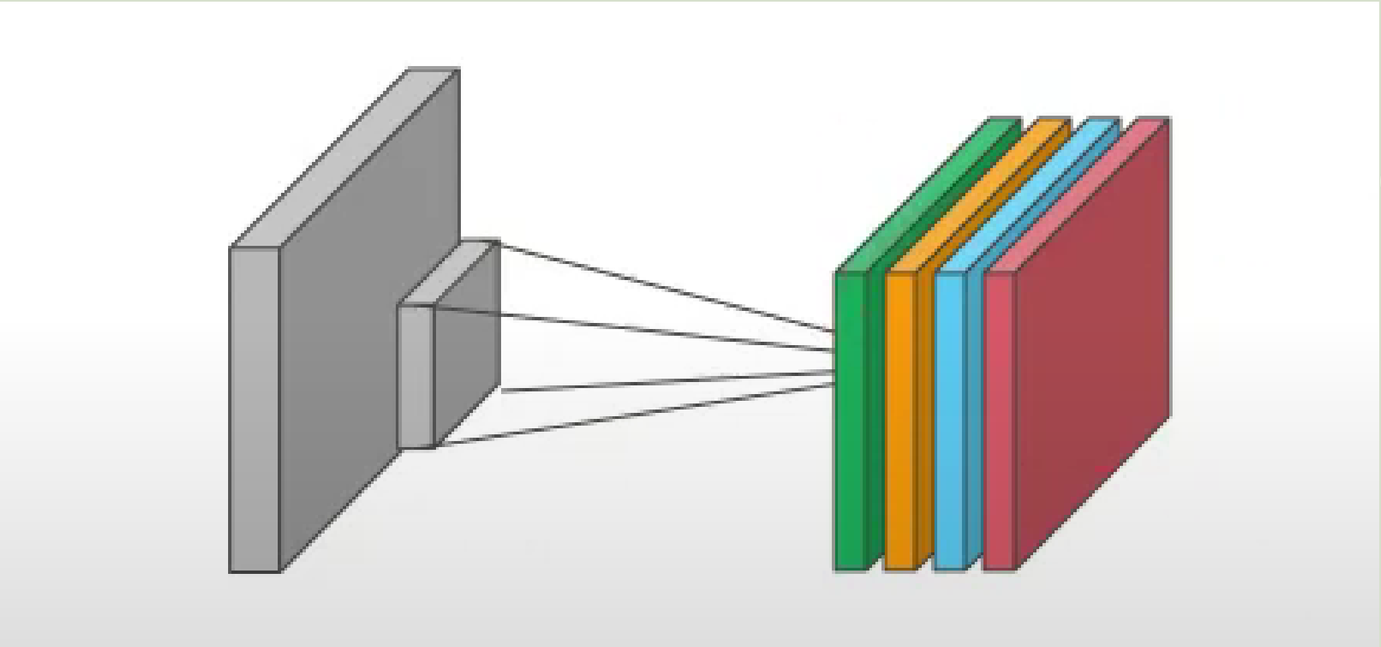
컨볼루션 층



- Q. 32 * 32 * 3 image에 5 * 5 * 3 filter를 집어넣을 경우.

activation map의 출력 사이즈는 어떻게 되는가?

- 고려사항 : Bias는 activation map에 계산되지 않고 파라미터 개수를 계산할 때 사용.



- Q. 32 * 32 * 3 image에 5 * 5 * 3 filter 4개를 집어넣을 경우.

activation map의 출력 사이즈는 어떻게 되는가?

- 고려사항 : 원래는 RGB가 채널이었지만 여기에서는 4개가 채널의 개수가 되는 것.

컨볼루션

- 영상 위에서 필터/커널을 슬라이딩하면서 겹쳐지는 부분의 각 원소값을 곱해서 모두 더한 값을 출력으로 하는 수학적인 연산

1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

Input

1*1	1*0	1*1	0	0
0*0	1*1	1*0	1	0
0*1	0*0	1*1	1	1
0	0	1	1	0
0	1	1	0	0

Input * Filter

1	1	1	0	0
0	1	1	1	0
0	0	1*1	1*0	1*1
0	0	1*0	1*1	0*0
0	1	1*1	0*0	0*1

Input * Filter

1	0	1
0	1	0
1	0	1

Filter / Kernel

4		

Feature Map

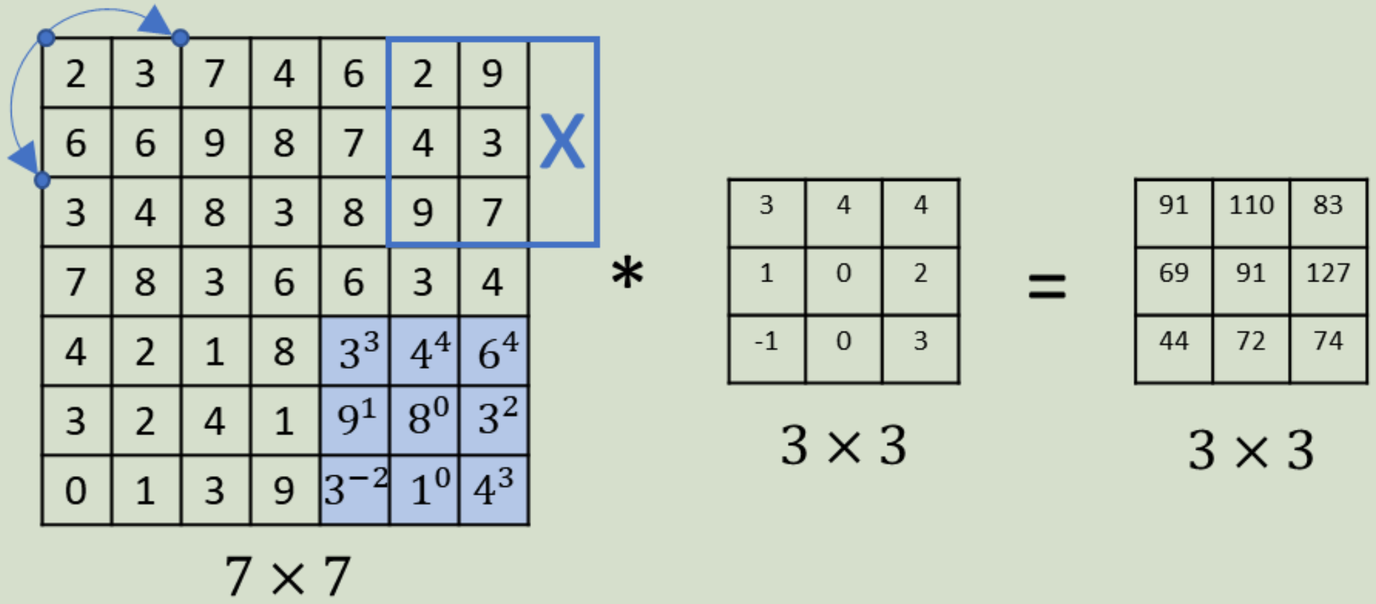
4	3	4
2	4	3
2	3	4

Feature Map

컨볼루션(Stride)

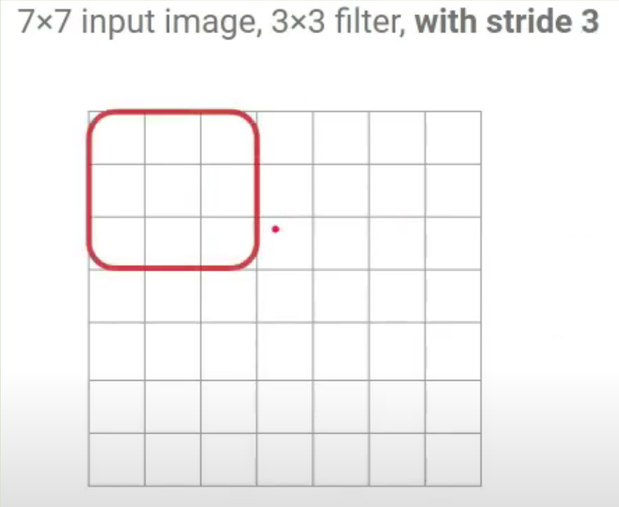
- 일반적으로 필터(커널)가 input image에서 1pixel만큼 이동하며 연산한다.

Stride를 주게되면 필터 이동시 해당 pixel만큼 이동하면서 연산하게 된다.



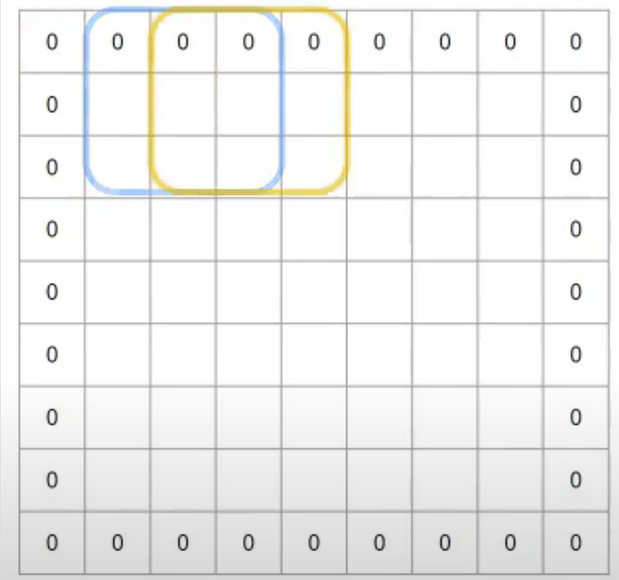
- stride를 키우면 공간적인 feature 특성을 손실할 가능성이 높음.
- 불필요한 특성을 제거하는 효과를 가져올 수 있음
- Convolution 연산 속도를 향상시키기도 함.

컨볼루션(padding)



- 해당 input image에 stride 3으로 설정할 시 이미지 추출이 가능할까?
- No! stride 3을 진행하였지만 영역이 남게되어 적절한 Stride가 아님.

- 이를 stride하여 나눌 수 있도록 하기 위해 Padding을 진행!

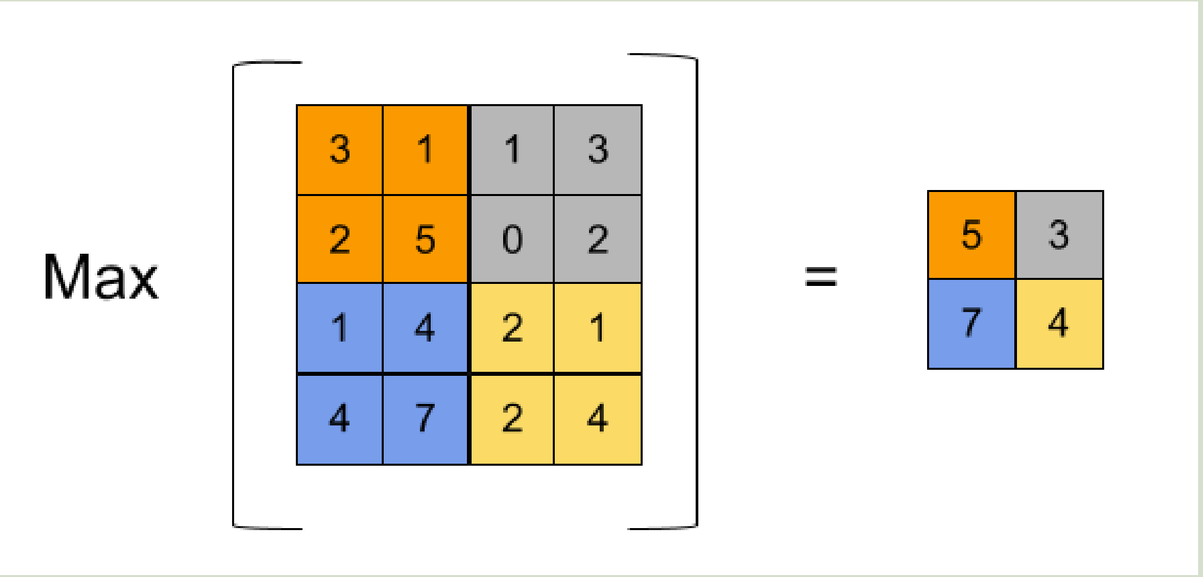


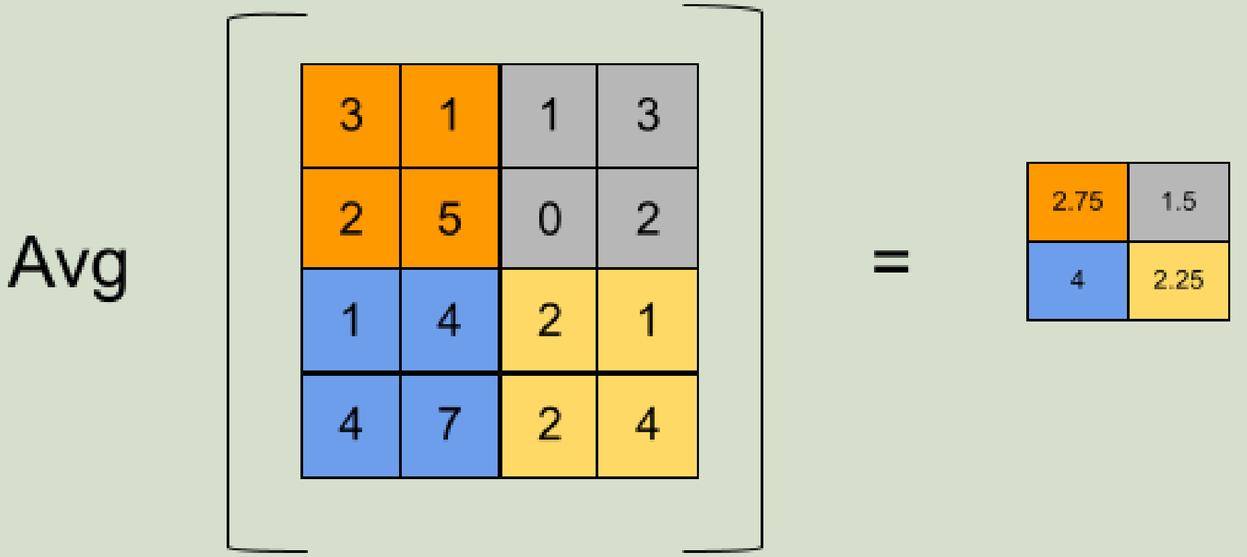
- Filter 적용 전 보존하려는 Feature map 크기에 맞게 입력 Feature map의 좌우 끝과 상하 끝에 0으로 값을 채워 Feature map의 size를 키움.

- 노이즈가 생길 수 있지만 큰 영향은 없음 + 모서리 주변 feature들의 특징을 보다 강화하는 장점이 있음.

컨볼루션(pooling)

- 위의 그림과 같이 stride 크기와 pool size를 동일하게 하여 pool size 내에서 특정 값만 가져오는 것.





- pooling의 종류에는 Max pooling과 Average pooling이 존재
(주로 max pooling을 사용)

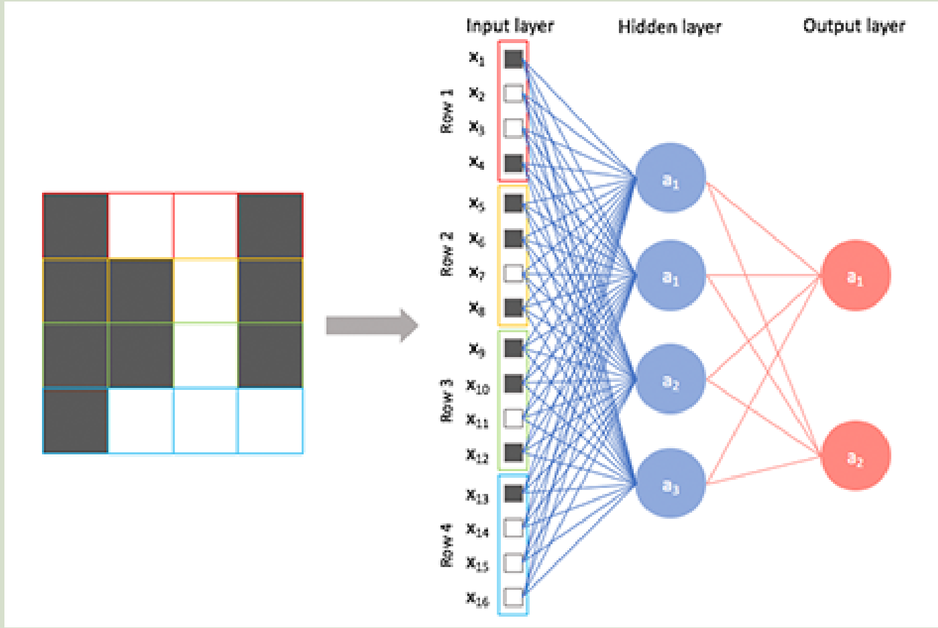
- pooling은 비슷한 feature들이 서로 다른 이미지에서 위치가 달라지면서 다르게 해석되는 현상을 중화.
- Feature map 크기가 줄어 계산 속도가 향상됨.

- 최근에는 손실되는 것이 생기는 우려가 있어 Pooling을 무조건 사용하는 것은 아니라고 함.

컨볼루션(Fully-Connected Layer)

- 이전 층에서 모든 뉴런이 다음 층에서 모든 뉴런과 연결된 계층을 말한다.

컨볼루션 층과 풀링 층으로부터 얻어진 특징 맵으로부터 영상을 분류하는데 사용되는 층이다.



- 완전 연결 층에서는 2차원의 특징 맵을 1차원의 배열 형태로의 평탄화(Flattening)을 통해 영상을 분류.

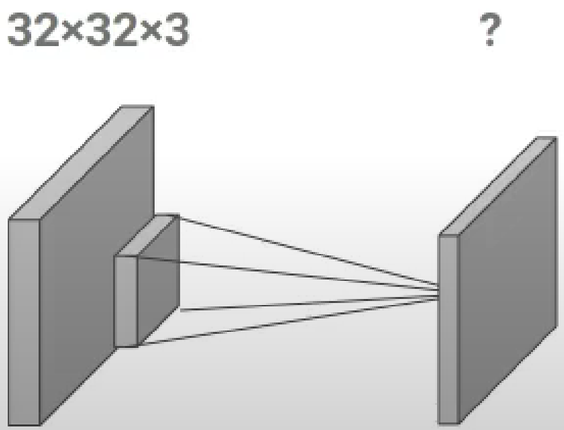
- 이 과정을 거쳤을 때 처음 Input에서 특징만 추출한 최종적인 Layer가 완성된다.

컨볼루션(Fully-Connected Layer) 예시

- 아래의 그림에서 주어진 이미지 크기. Stride와 padding 크기를 고려하여 문제를 풀어보시오.

Given an input volume of **32×32×3**, apply **10 5×5 filters** with **stride 1**, **pad 2**.

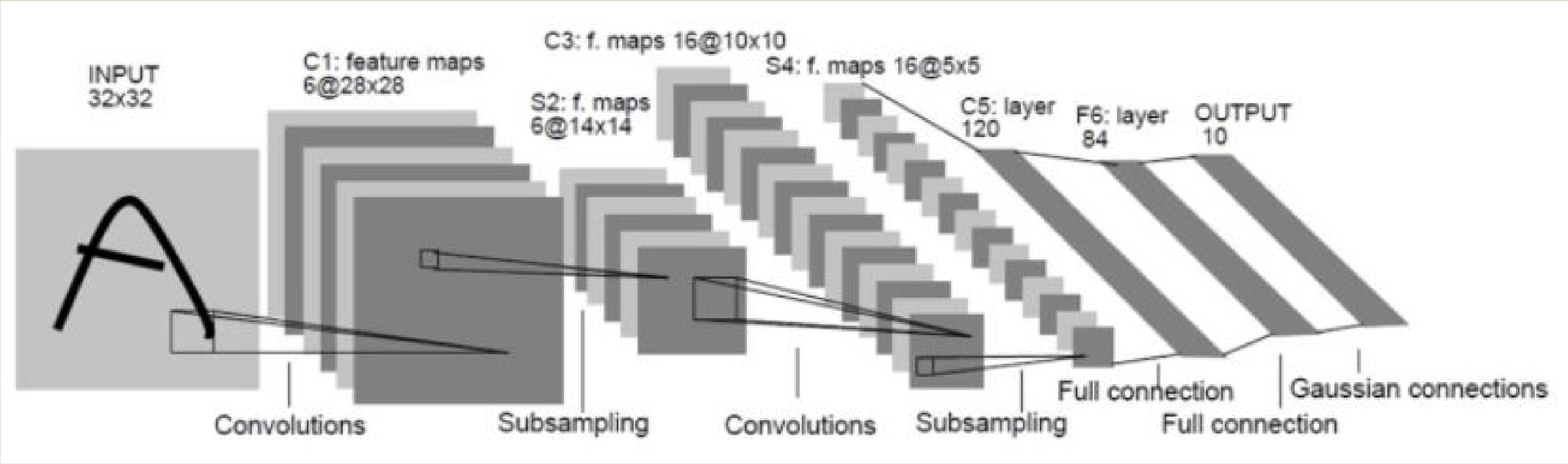
What is the output size?
Number of parameters?
cf. Number of parameters if fully-connected?



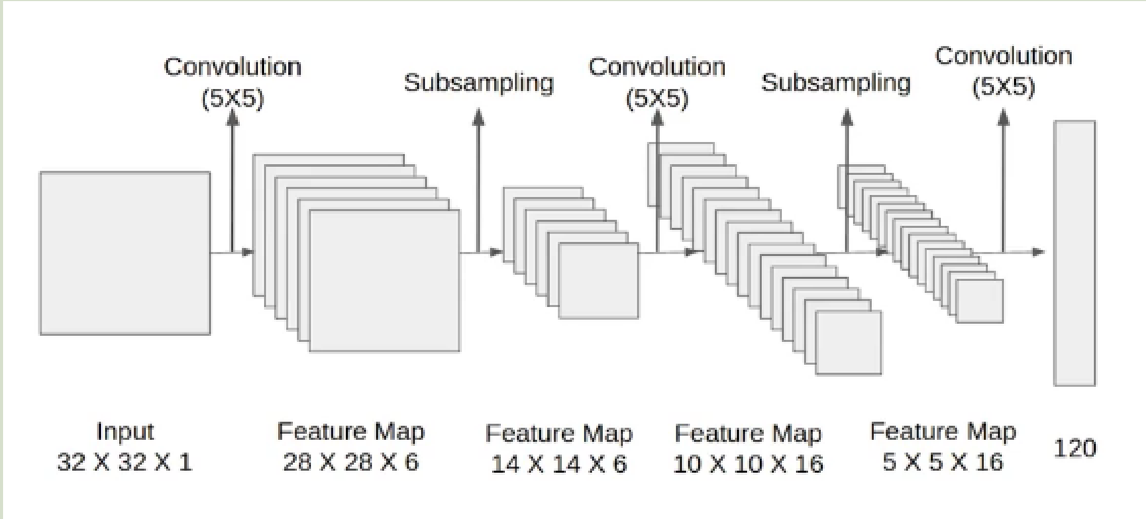
- 고려사항 : parameter의 개수와 fully-connected의 parameter 개수는 다를 것이다.
- 추가사항 : parameter개수를 알아보는 단서는 filter에 존재할 것이다.

LeNet-5 사례

- 1998년에 개발한 CNN 알고리즘(CNN 알고리즘에서는 최초의 모델)

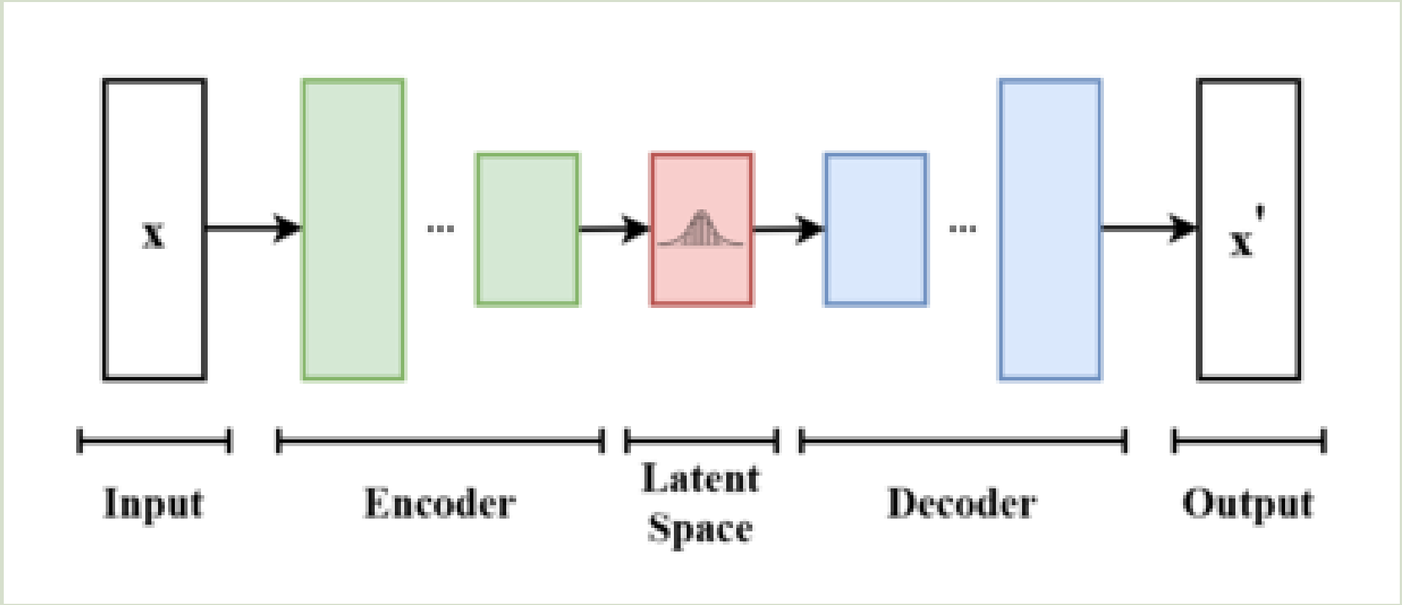


- 기본적인 구조 : C–P–C–P–FC–FC



LeNet-5를 통한 CNN의 장점

- 유연한 구조 : 컨볼루션층과 풀링층, 완전연결층을 쌓아 만들기 때문에 데이터에 따라 or 풀어야 하는 문제에 따라 다양한 모양으로 조립할 수 있다는 장점이 있음.



- 오토인코더를 둔 예시 : 인코더는 특징 맵을 점점 작게 함 + 디코더는 다시 키워 원래 영상을 복원함.

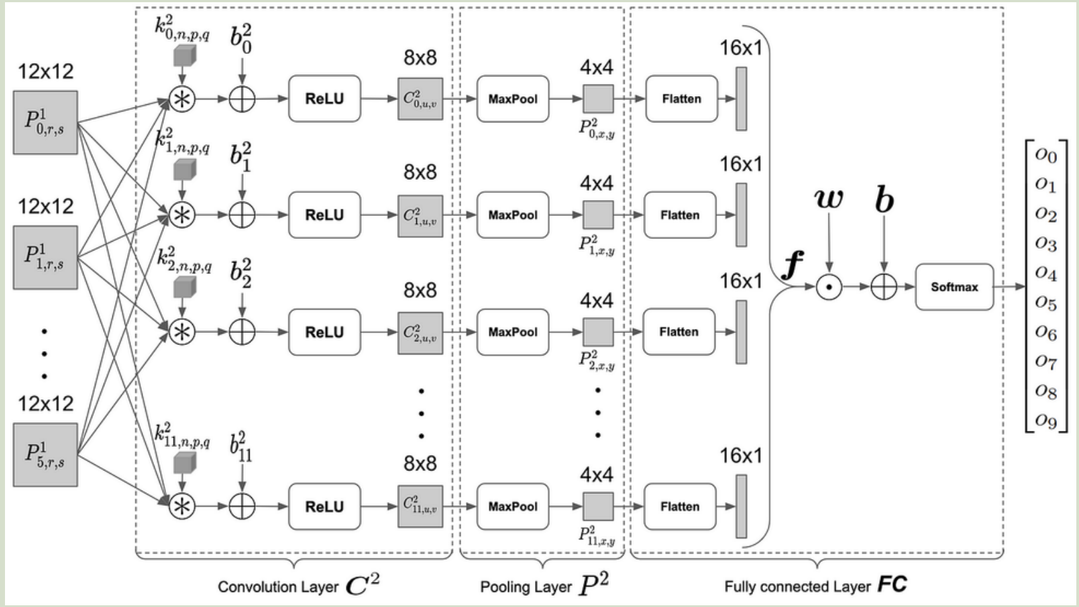
(이는 대칭을 이루도록 설계)

- 신경망의 가운데 있는 층은 원래 영상보다 크기가 훨씬 축소된 특징 맵을 생성.

- 따라서 영상의 특징 추출기 or 영상 압축기 등에 활용할 수 있음.

CNN이 사용하는 역전파 알고리즘

- CNN 역시 역전파 알고리즘을 사용한다. 전방 계산 – 손실 함수 계산 – 역전파 과정을 거쳐서 오류를 줄이는 방향으로 가중치를 갱신.



- 다층 퍼셉트론을 위한 역전파 알고리즘이 그대로 적용

(다만 계산이 다르기 때문에 계산식에 따라 미분만 다르게 수행되는 것.)

- 참고 사이트 : https://metamath1.github.io/cnn/index.html

CNN의 특징학습과 통째학습

- 특징 학습 : 사람이 합리적 사고를 통해 그럴듯해 보이는 필터를 설계하고 데이터의 특성을 무시한 채 모든 데이터에 일괄 적용하는 고전적인 접근 방법에서 '주어진 데이터셋을 인식하는데 최적인 필터를 찾아냄'으로 진행되는 학습.

- 통째 학습 : 특징 학습을 다른 관점에서 바라볼 때.

LBP,SIFT,HOG 등은 수작업 특징 중에 유명한 것들인데 이는 특징 추출프로그램과 분류기 프로그램을 결합하여 인식기를 완성하는 패러다임이다.

하지만 딥러닝 패러다임에서는 통합된 신경망에서 특징 학습과 분류기 학습을 통째로 진행함.

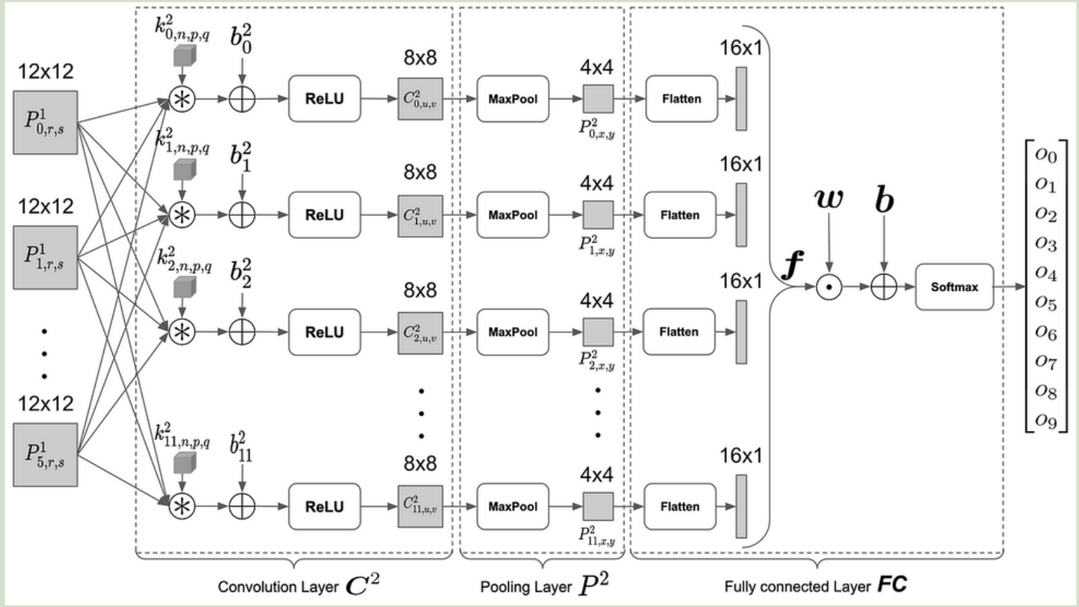
- 컨볼루션 신경망이 우수한 이유

1. 데이터의 원래 구조를 유지한다.
2. 특징 학습을 통해 최적의 특징을 추출한다.
3. 신경망의 깊이를 깊게 할 수 있다.

향상을 위한 알고리즘

CNN이 사용하는 역전파 알고리즘

- CNN 역시 역전파 알고리즘을 사용한다. 전방 계산 – 손실 함수 계산 – 역전파 과정을 거쳐서 오류를 줄이는 방향으로 가중치를 갱신.



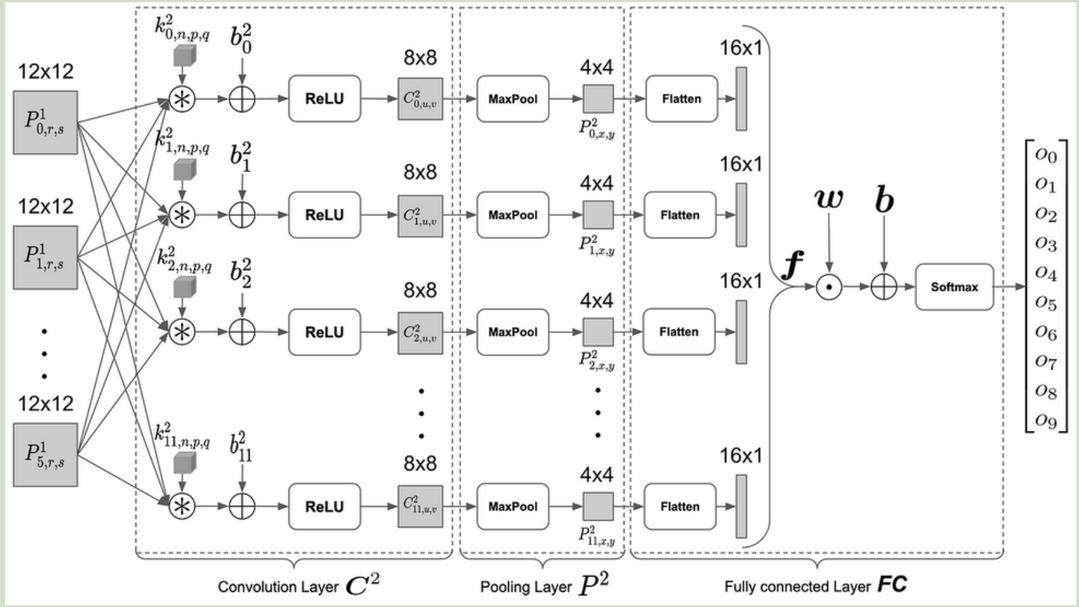
- 다층 퍼셉트론을 위한 역전파 알고리즘이 그대로 적용

(다만 계산이 다르기 때문에 계산식에 따라 미분만 다르게 수행되는 것.)

- 참고 사이트 : https://metamath1.github.io/cnn/index.html

CNN이 사용하는 역전파 알고리즘

- CNN 역시 역전파 알고리즘을 사용한다. 전방 계산 – 손실 함수 계산 – 역전파 과정을 거쳐서 오류를 줄이는 방향으로 가중치를 갱신.



- 다층 퍼셉트론을 위한 역전파 알고리즘이 그대로 적용

(다만 계산이 다르기 때문에 계산식에 따라 미분만 다르게 수행되는 것.)

- 참고 사이트 : https://metamath1.github.io/cnn/index.html

CNN의 학습 알고리즘 향상

- 손실함수

- 교차 엔트로피

두 개의 확률분포 P와 Q에 대해 하나의 사건 X가 갖는 정보량으로 정의.
실제 데이터의 확률 분포인 P와 학습된 모델에 의해 얻어진 확률 분포 Q의 차이를 구하는데 사용된다.

$$H(P,Q) = -\sum_{x}^{n} P(x) log_2 Q(x)$$

$$= -\sum_{i=1,...k}^{k} P(e_i) log_2 Q(e_i)$$

- Q는 모델의 예측 확률 분포를 의미하며 P는 모델의 참값에 대한 확률 분포를 나타낸다.

CNN의 학습 알고리즘 향상

- **손실함수 문제 예시**

다음 교차 엔트로피 손실함수를 계산해보시오.

Q = [0.1 0.2 0.6 0.1]
P = [0 0 1 0]

$$H(P,Q) = -\sum_{x}^{n} P(x) log_2 Q(x)$$

$$= -\sum_{i=1,...k}^{k} P(e_i) log_2 Q(e_i)$$

- 정보 이론에 따라 확률이 낮은 사건일수록 값어치있는 정보라는 의미. 반대의 경우에는 더 적은 정보를 전달.
- 따라서 확률(P(x))과 정보량(I(x))은 반비례 관계

- 엔트로피는 사건들에 대한 평균적인 정보량을 나타냄. 사건이 일어날 확률이 모두 같을 때 가장 높다.

CNN의 학습 알고리즘 향상

- Focal 손실 함수

- 부류 불균형이 심한 경우에 주로 사용하는 손실 함수.
- 참값이 물체인 화소 : o = (p, 1−p), y = (1,0)
- 참값이 배경인 화소 : o = (p, 1−p), y = (0,1)

Pt = { p, 참값이 물체인 화소
1−p, 참값이 배역인 화소}

$$FL = -(1 - P_t)^{\gamma} \log(P_t)$$

$$CE = -\log(P_t)$$

CNN의 학습 알고리즘 향상

- Focal 손실 함수와 Cross Entropy loss가 다른 이유?

- Focal loss에서는 마지막에 출력되는 각 클래스의 P(x)를 이용해 CD Loss에 통과된 최종 확률값이 큰 EASY 케이스의 Loss를 크게 줄이고 최종 확률 값이 낮은 HARD 케이스의 Loss를 낮게 줄이는 역활을 한다.

- 보통 Cross Entropy loss는 확률이 낮은 케이스에 패널티를 주는 역활만 하고 확률이 높은 케이스에는 어떠한 보상도 줌.

- 그러나 Focal Loss는 확률이 높은 케이스에는 확률이 낮은 케이스보다 Loss를 더 크게 낮추는 보상을 주는 차이점이 존재.

관련 내용 및 코드 구현 : https://velog.io/@heaseo/Focalloss-%EC%84%A4%EB%AA%85

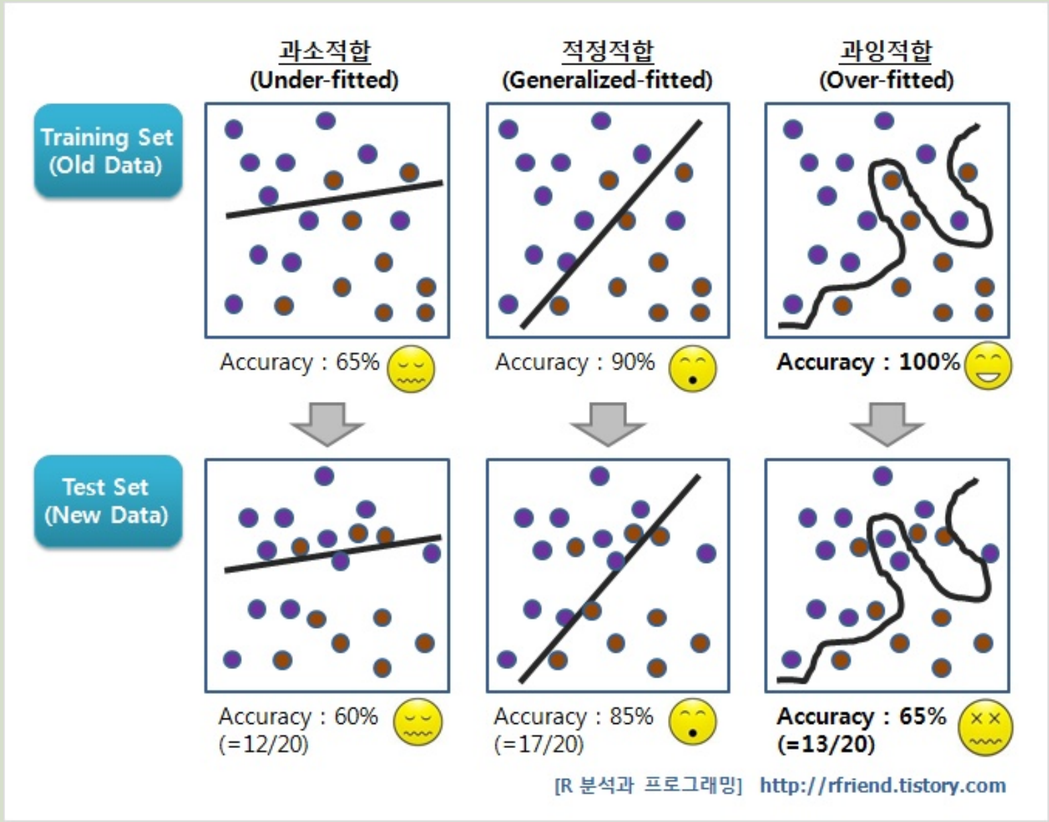
CNN의 학습 알고리즘 향상

- 규제

- 딥러닝은 층이 깊은 신경망을 사용하기 때문에 학습 알고리즘이 추정할 가중치가 매우 많음.
- 데이터 부족은 과잉 적합으로 나타날 수 있는데 딥러닝은 신경망 모델의 용량을 크게 유지하면서 여러 규제 기법을 적용해 과잉 적합을 방지하는 전략을 사용한다.

- 과잉 적합

- 데이터에 비해 용량이 너무 작아 모델링이 제대로 되지 못하는 것 -> 과소 적합
- 반대로 고차원의 모델은 훈련 집합에 대해서는 거의 완벽하게 모델링하는데 훈련에 참여하지 않은 데이터에 대해 낮은 성능을 보이는 것 -> 과잉 적합
- 이 책에서는 데이터를 늘리면 과잉 적합 현상을 누그러뜨릴 수 있다는 사실을 보여준다. 훈련 집합 크기를 20에서 30,50,100으로 늘리면 과잉 적합이 점점 누그러지기 때문.



CNN의 학습 알고리즘 향상

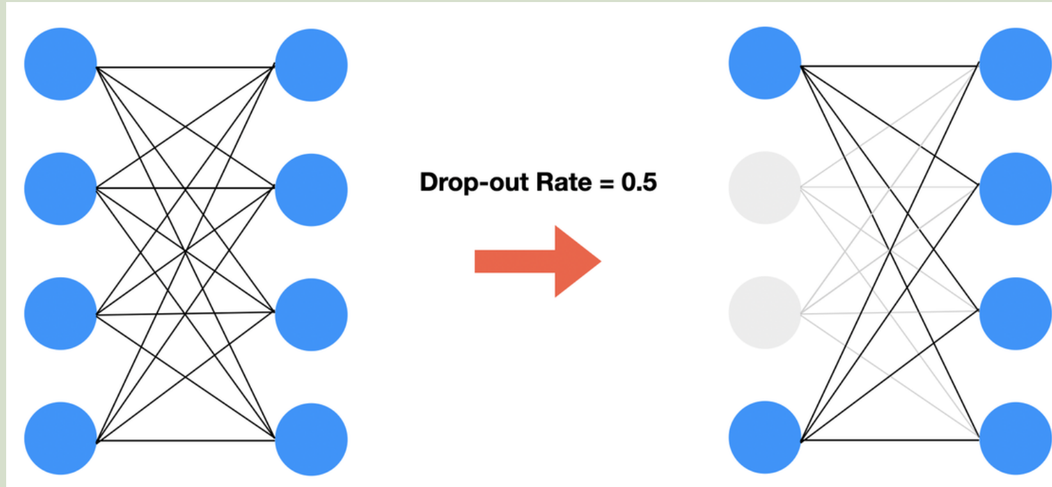
- 데이터 증강

- 데이터를 늘리면 과잉 적합을 방지할 수 있다. 하지만 데이터 수집에는 많은 비용과 시간이 들기에 데이터를 증강하는 방법이 존재.
- 영상에 약간의 이동. 회전. 크기. 명암 변환을 랜덤하게 적용하여 데이터를 증강한다.

- 드롭 아웃

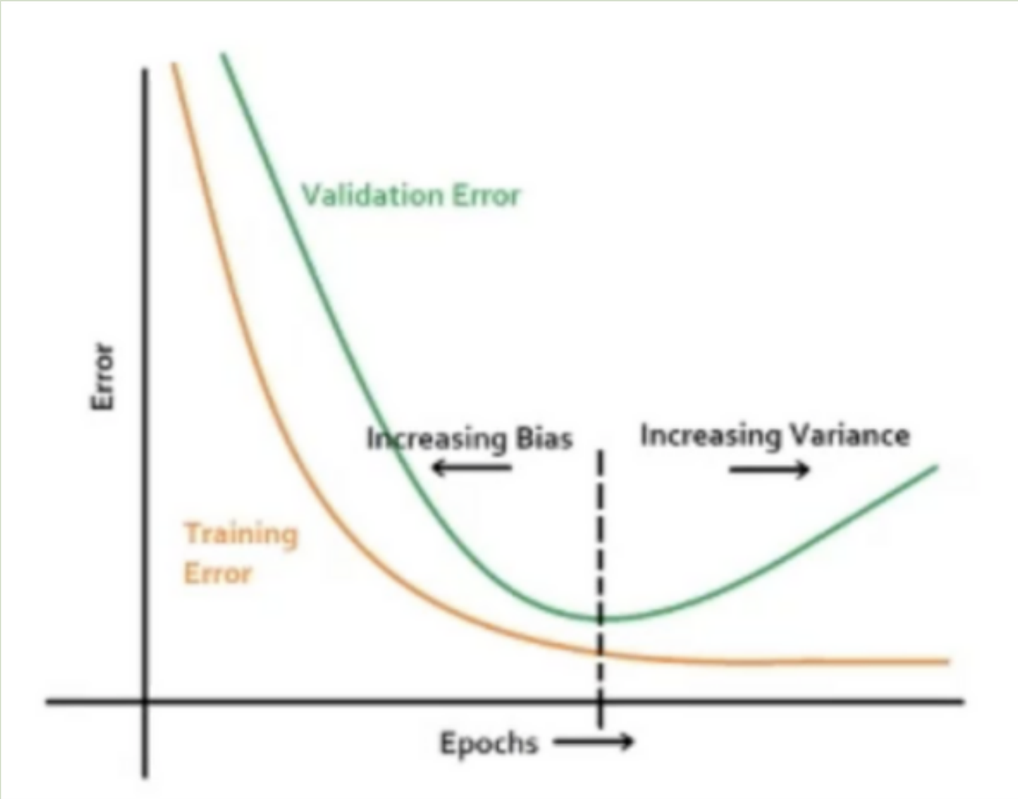
- 특징 맵을 구성하는 요소 중 일부를 랜덤 선택하여 0으로 설정하는 단순한 아이디어인데 과잉 적합을 방지하는데 매우 효과적.
- Ex. r = 0.3으로 설정하면 30% 가량이 0이 된다.
- Ex. r = 0.5로 설정하면 50%가량이 0이 되는 것!

- 드롭아웃은 학습할 때만 적용하고 예측할 때는 적용하지 않는다.



CNN의 학습 알고리즘 향상

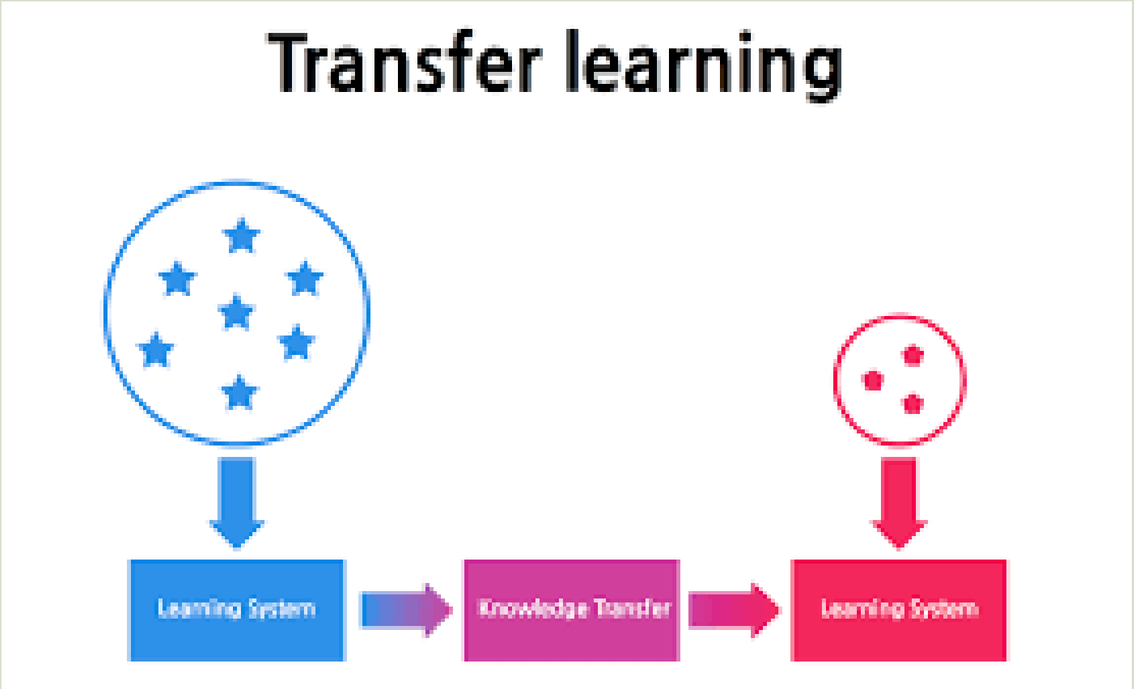
- 조기 멈춤(Early Stopping)



- 훈련 집합은 꾸준히 성능이 개선되지만 검증 집합에선 성능 개선이 없다...? -> 과잉 적합이 발생!
- 이런 상황에 '검증 집합에 대한 성능 개선이 더 없거나 퇴화'한다면 학습을 마치는 전략!
- 검증집합의 오류가 최저인 점에서 학습을 멈춤.

CNN의 학습 알고리즘 향상

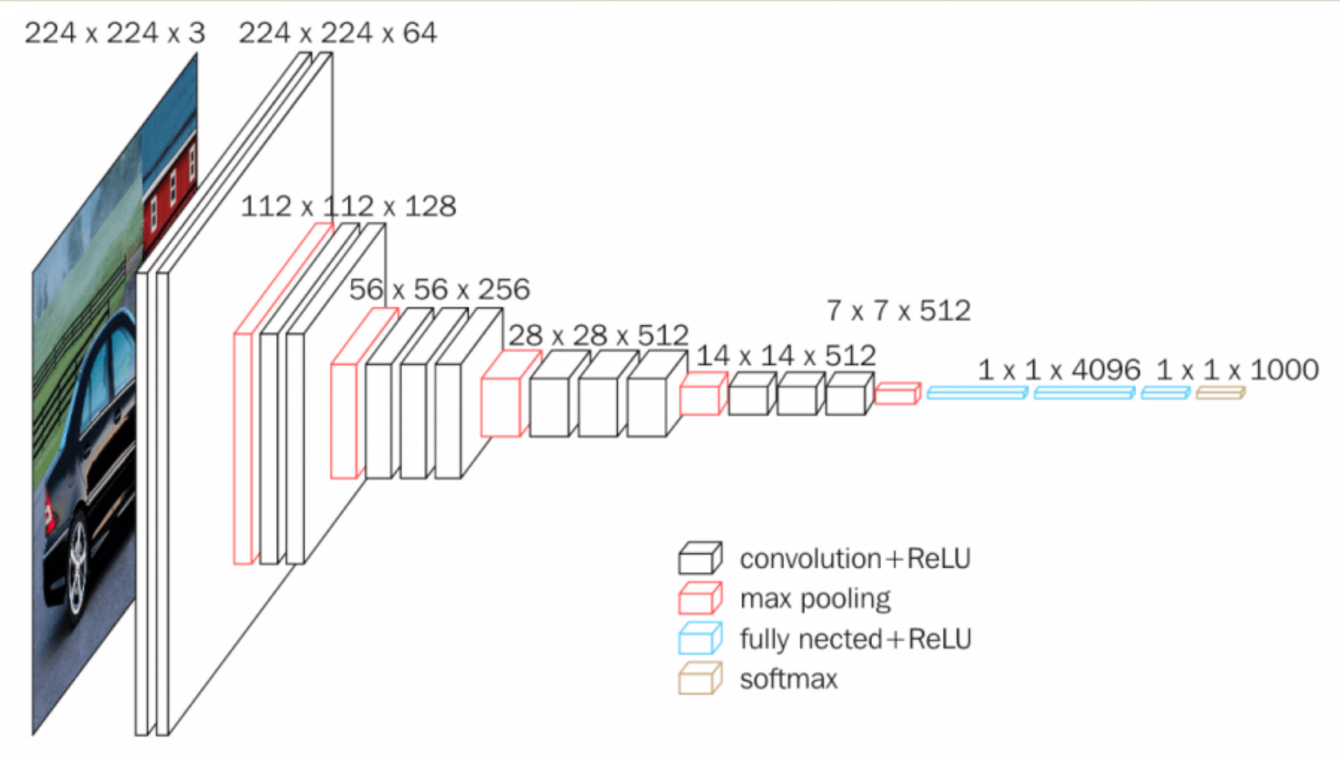
- 전이 학습



- 방안으로 나오게 된 배경 : 너무 많은 정보와 데이터들을 학습하는데 있어 오랜 시간이 걸림 + 좋은 하이퍼 매개변수를 찾으려면 학습을 수십 번 시도해야 하는 부담
- 전이 학습은 어떤 도메인의 데이터로 학습한 모델을 다른 도메인에 적용해 성능을 높이는 방법
- Ex. ImageNet 데이터셋으로 학습한 모델로 개의 품종을 인식하는데 전이 학습을 사용 시 높은 성능을 쉽게 확보할 수 있다.

CNN의 학습 알고리즘 향상

- VGGNet



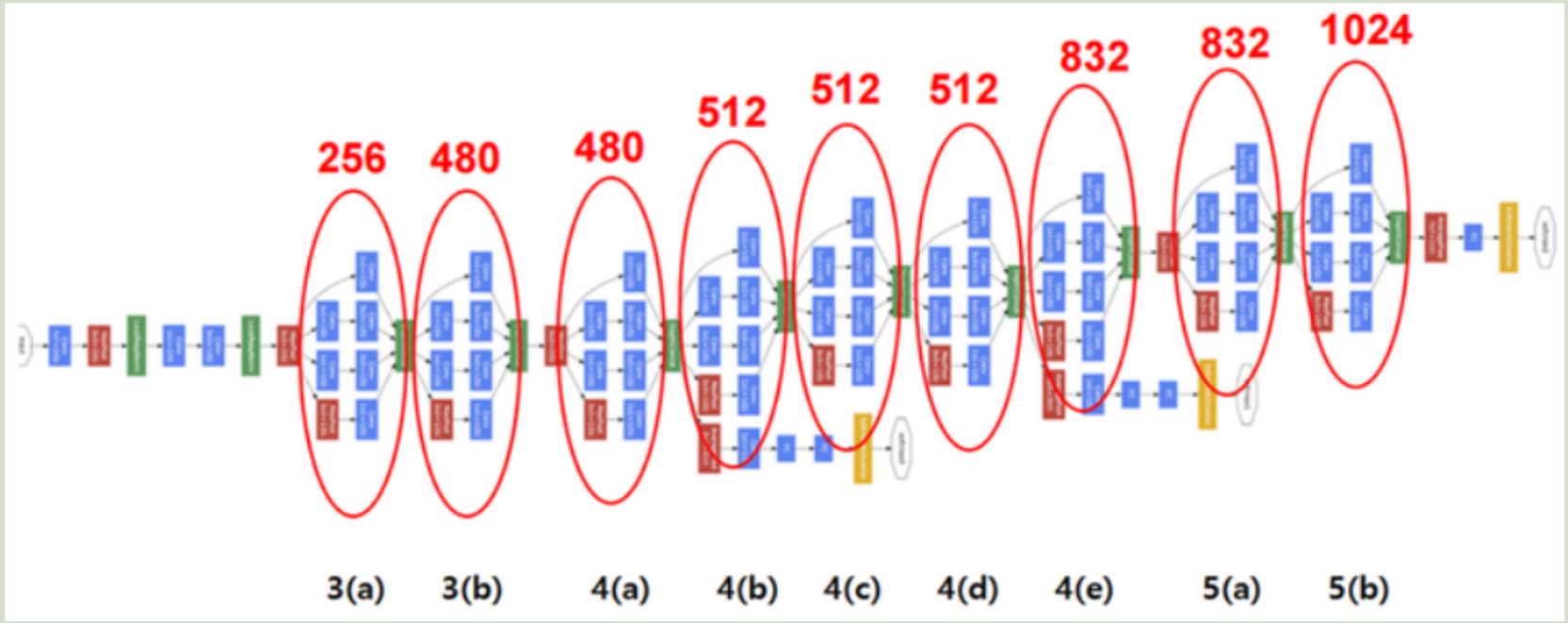
- 3*3 Conv으로만 진행하지만 대신 층이 더욱 깊어지는 특징이 있음.

(가장 많은 메모리가 초기 Conv에 소비되고 가장 많은 파라미터가 늦은 FC에 존재)

- Dropout은 0.5로 규정하며 Batch Size는 256으로 규정하였다.
- 사용한 옵티마이저는 SGD + Momentum으로 0.9로 정하여 lr=0.01

CNN의 학습 알고리즘 향상

- GoogLeNet



- GoogLeNet 이외에도 InceptionNet이라는 별명을 가지고 있는 모델(컨볼루션 층 안에 다층 퍼셉트론을 둔 네트워크 속 네트워크가 있기 때문)
- 22층으로 구성 (9 inception modules * 2 layers + 3 Conv + 1 FC)
- Parameter가 적은 이유가 FC 2개를 지운 것이 주요하였음.
- SGD+Momentum = 0.9 lr은 8 epochs 마다 4% 감소

CNN의 학습 알고리즘 향상

- ResNet

- ResNet은 152개의 층을 가지고 있으며 GoogLeNet의 22개의 층보다 더욱 깊어진 특징이 있다.

Residual Block을 제안하여 입력값을 출력값에 더해줄 수 있도록 지름길을 하나 만들어 줌.

- Dropout은 없으며 Batch Size는 256으로 규정하였다.
- 사용한 옵티마이저는 SGD + Momentum으로 0.9로 정하였고 lr=0.1