2022 데이터 크리에이터 캠프

Data Creator Camp

대학부 실습영상 - 정원기

과학기술정보통신부 NIA 한국지능정보사회진흥원

강의 목차

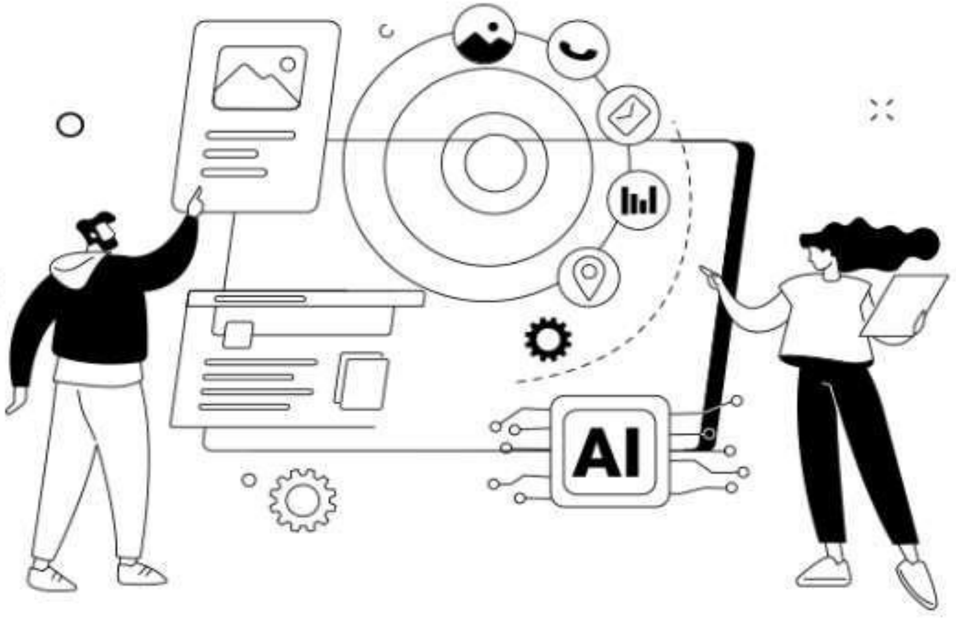
- 대회소개
- 대학부문제소개
- CNN에대한전반적인설명
- CNN을이용한Cat&DogClassification실습

대회 소개

http://creator.kbig.kr/

교육기회

- 데이터 크리에이터 캠프는 문제 해결을 위한 학습 영상 제공을 통해 교육의 기회를 제공합니다.
- 온라인 사전 학습 영상을 통해 문제 해결에 활용할 수 있는 지식을 습득하고 역량을 강화할 수 있습니다.



성장경험

- 매주 1회 데이터 전문가의 1:1(팀) 멘토링을 통해 성장의 기회를 제공합니다.
- 한 달간의 예선 기간 동안 문제 해결을 위해 다양한 접근을 경험하여 이론을 실제 활용할 수 있는 기회를 가질 수 있습니다.

대회 소개


01
4주간의 해커톤 예선전
· 예선기간 주차별 소문제 미션 진행
· 4주차에 3주간의 결과를 취합 후 PPT로 제출, 평가를 통한 본선진출팀 선발
02
온라인 사전학습 제공
· 대회 참여를 위한 사전 안내사항 및 대회 참여 가이드 제공
· 예선 기간동안 각 팀별 온라인 연수원을 통한 학습진행 (평가시 진도율 반영)
03
멘토들의 튜터링
· 팀별 멘토 배정 및 1일 1팀당 1개의 문의를 통한 문제 해결 지원
· 예선 기간 동안 총 3회 멘토링데이 운영 (매주 토요일 온라인, 팀별 50분 멘토링 진행)
04
본선
· 현장 발표와 질의응답을 통해 질의응답 후 시상 결정
· 실무형 초청 특강

대회 소개

과학기술정보통신부 NIA 한국지능정보사회진흥원

대회 소개

장 소

예선
온라인 진행

본선
한국지능정보사회진흥원
스마트스퀘어

한국지능정보사회진흥원(NIA) 스마트스퀘어: 서울특별시 중구 청계천로 14

* 정부, 지자체 행사 운영지침에 의거 철저한 방역수칙 준수를 통한 대회 운영하며 상황에 따라 변동 가능

대학부 데이터셋 소개

https://aihub.or.kr/aihubdata/data/view.do?currMenu=115&topMenu=100&aihubDataSe=realm&dataSetSn=617

- 제공데이터셋

과학기술정보통신부 NIA 한국지능정보사회진흥원

대학부 데이터셋 소개

https://aihub.or.kr/aihubdata/data/view.do?currMenu=115&topMenu=100&aihubDataSe=realm&dataSetSn=617

- 본대회에서는이중일부데이터를서브샘플링하여대회제공

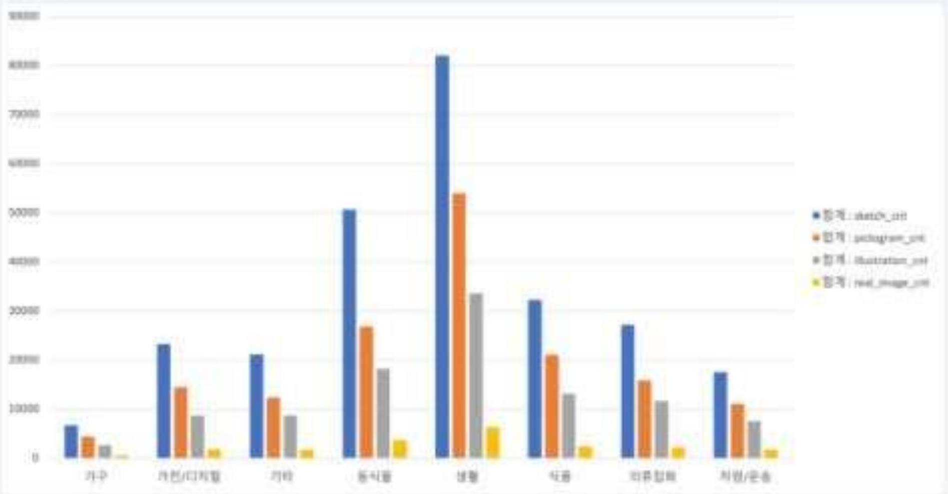
데이터 통계

추상이미지

1. 데이터 구축 규모

구분	픽토그램(아이콘)	일러스트레이션	스케치 (라인드로잉)	합계
비율(%)	30	20	50	100
데이터 수(건)	160,215	104,333	260,840	525,388

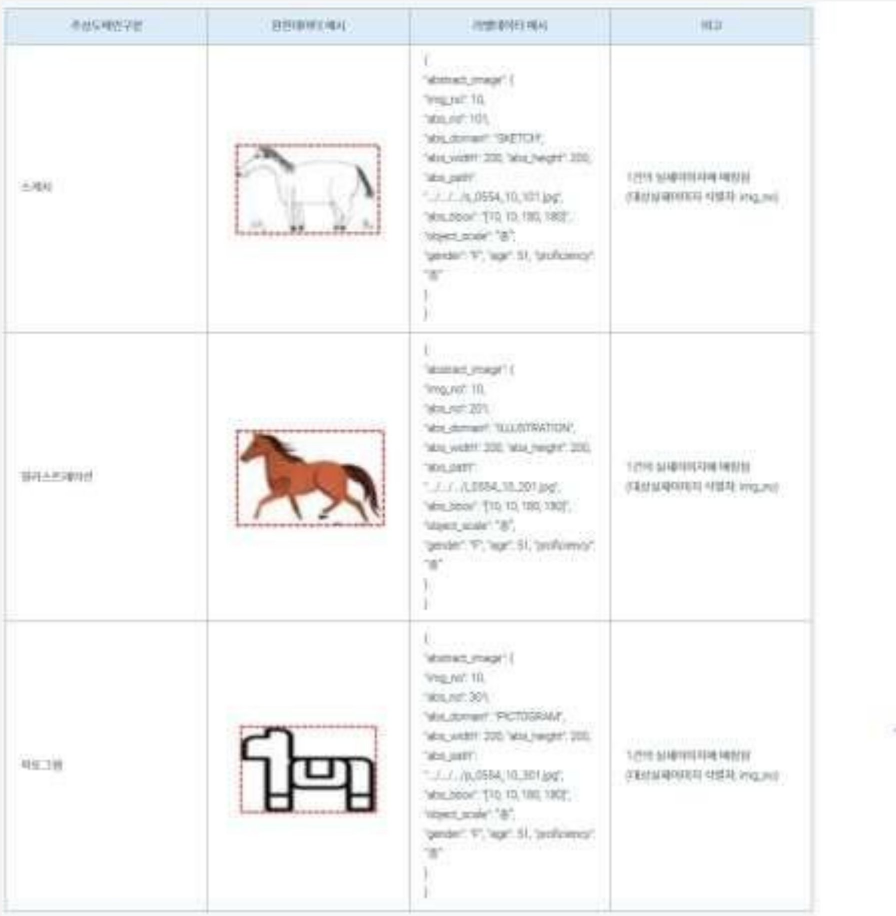
2. 데이터 분포



대학부 데이터셋 소개

https://aihub.or.kr/aihubdata/data/view.do?currMenu=115&topMenu=100&aihubDataSe=realm&dataSetSn=617

- 라벨데이터예시

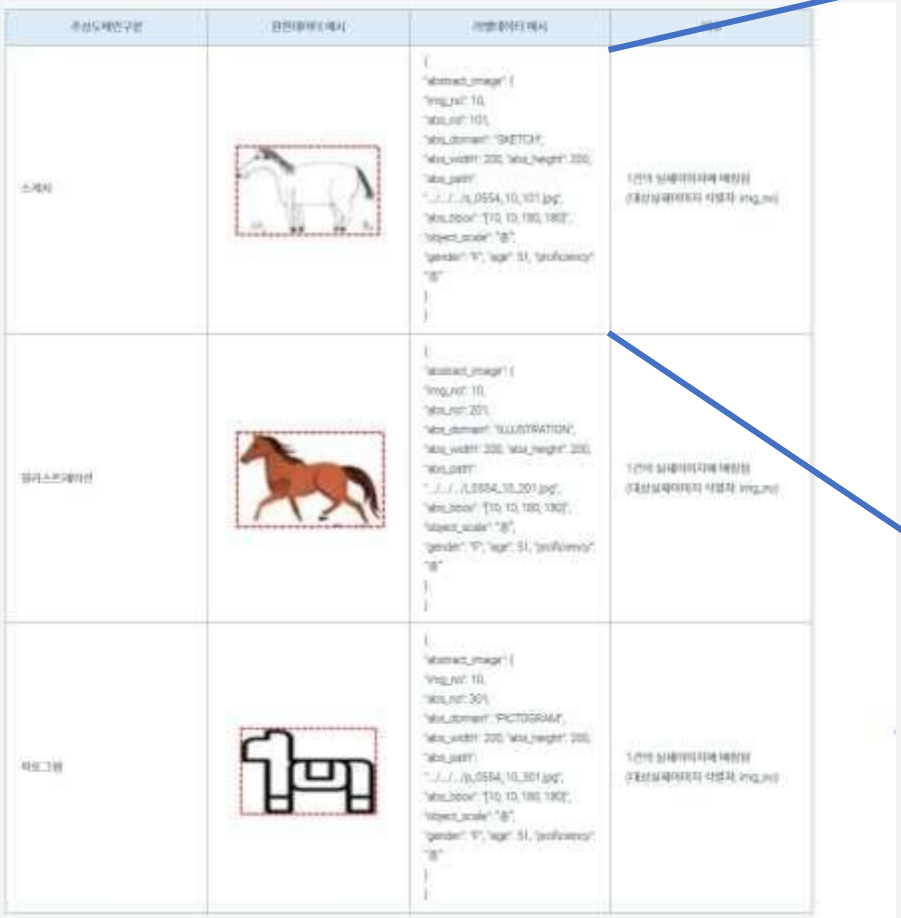


추상도메인구분	원천데이터 예시	라벨데이터 예시	비고
스케치		{ "abstract_image": { "img_no": 10, "abs_no": 101, "abs_domain": "SKETCH", "abs_width": 200, "abs_height": 200, "abs_path": "../../../s_0554_10_101.jpg", "abs_bbox": "[10, 10, 190, 180]", "object_scale": "중", "gender": "F", "age": 51, "proficiency": "중" } }	1건의 실세이미지에 매칭됨 (대상실제이미지 식별자 img_no)
일러스트레이션		{ "abstract_image": { "img_no": 10, "abs_no": 201, "abs_domain": "ILLUSTRATION", "abs_width": 200, "abs_height": 200, "abs_path": "../../../i_0554_10_201.jpg", "abs_bbox": "[10, 10, 190, 190]", "object_scale": "중", "gender": "F", "age": 51, "proficiency": "중" } }	1건의 실세이미지에 매칭됨 (대상실제이미지 식별자 img_no)
픽토그램		{ "abstract_image": { "img_no": 10, "abs_no": 301, "abs_domain": "PICTOGRAM", "abs_width": 200, "abs_height": 200, "abs_path": "../../../p_0554_10_301.jpg", "abs_bbox": "[10, 10, 190, 190]", "object_scale": "중", "gender": "F", "age": 51, "proficiency": "중" } }	1건의 실세이미지에 매칭됨 (대상실제이미지 식별자 img_no)

대학부 데이터셋 소개

https://aihub.or.kr/aihubdata/data/view.do?currMenu=115&topMenu=100&aihubDataSe=realm&dataSetSn=617

- 라벨데이터예시



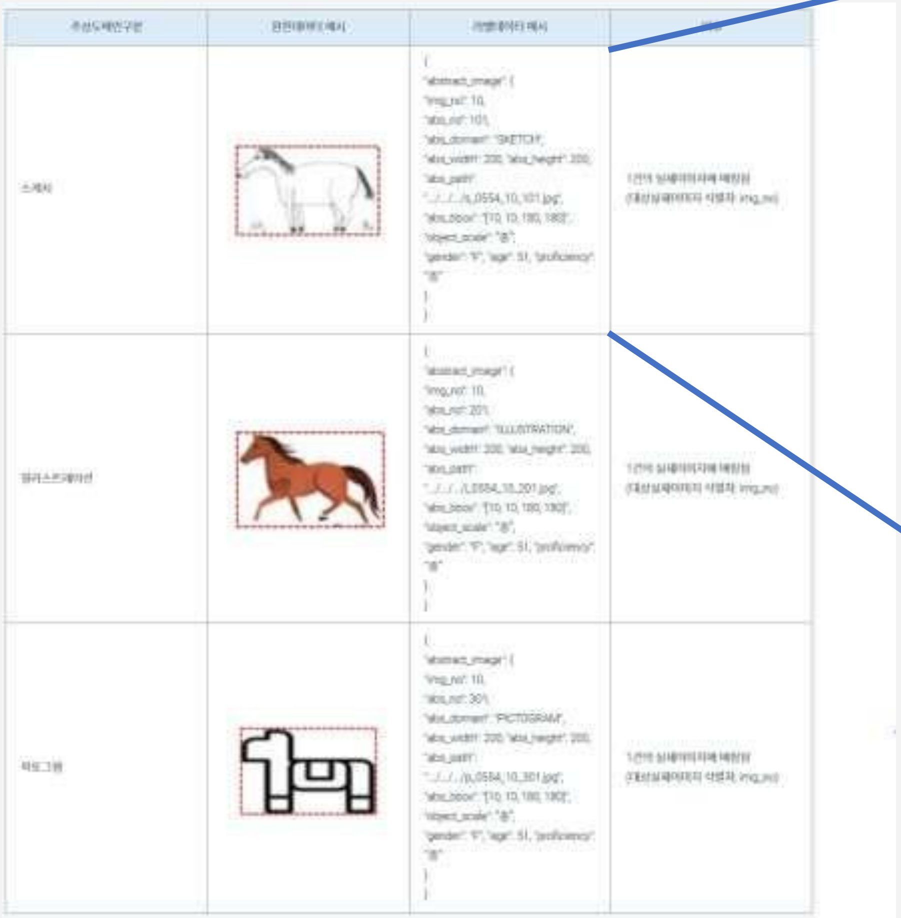

```
{
"abstract_image": {
"img_no": 10,
"abs_no": 101,
"abs_domain": "SKETCH",
"abs_width": 200, "abs_height": 200,
"abs_path":
"../../.../s_0554_10_101.jpg",
"abs_bbox": "[10, 10, 180, 180]",
"object_scale": "중",
"gender": "F", "age": 51, "proficiency":
"중"
}
}
```


JSON 파일로 구성

대학부 데이터셋 소개

https://aihub.or.kr/aihubdata/data/view.do?currMenu=115&topMenu=100&aihubDataSe=realm&dataSetSn=617

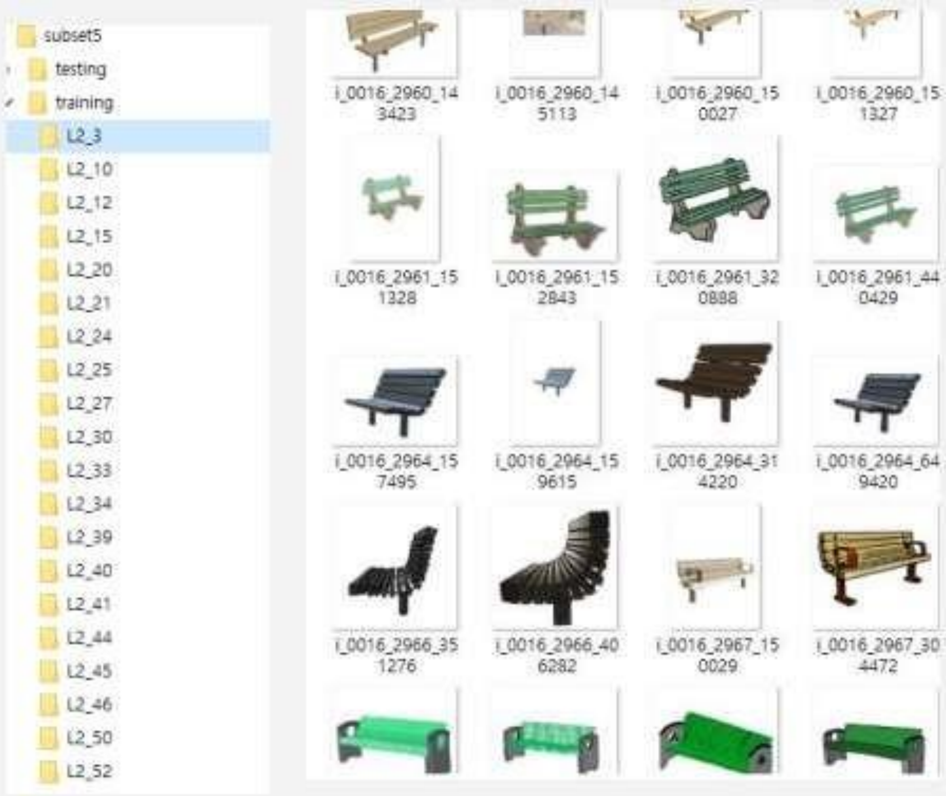
- 라벨데이터예시



{
"abstract_image": {
"img_no": 10,
"abs_no": 101,
"abs_domain": "SKETCH",
"abs_width": 200, "abs_height": 200,
"abs_path":
".../.../.../s_0554_10_101.jpg",
"abs_bbox": "[10, 10, 180, 180]",
"object_scale": "중",
"gender": "F", "age": 51, "proficiency":
"중"
}
}

대학부 데이터셋 소개

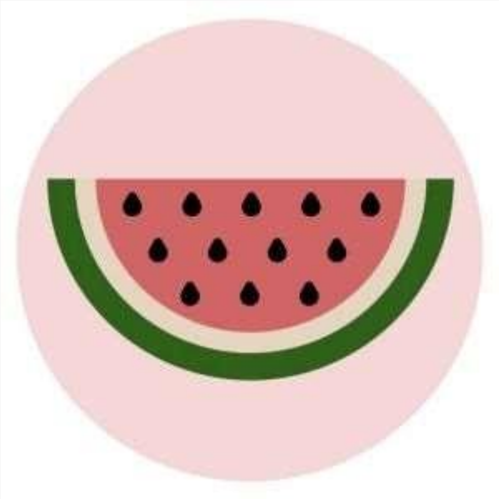
- 데이터소개
  - 픽토그램(아이콘), 일러스트레이션, 스케치 (라인드로잉)세종류의데이터로구성
- 학습데이터및평가데이터개수
  - 학습데이터: 10,000 여장
  - 평가데이터: 1,000 여장
  - 레이블정보: 20개 class



대학부 데이터셋 소개

https://m.news1.kr/articles/?3026238

- [이슈] 데이터분류상의문제(Real Image 포함)



Pictogram

Real Image

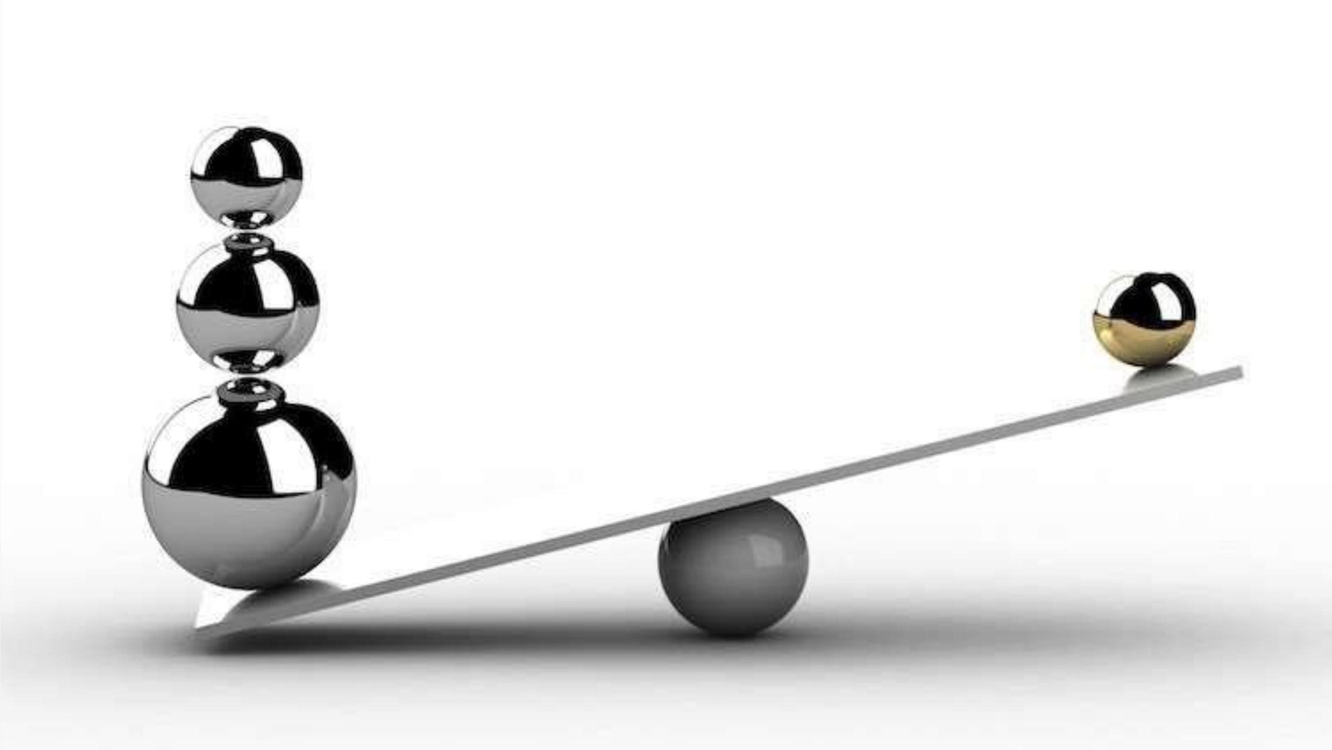
대학부 데이터셋 소개

- [이슈] 데이터분류상의문제

대학부 데이터셋 소개

https://www.kaggle.com/code/shahules/tackling-class-imbalance

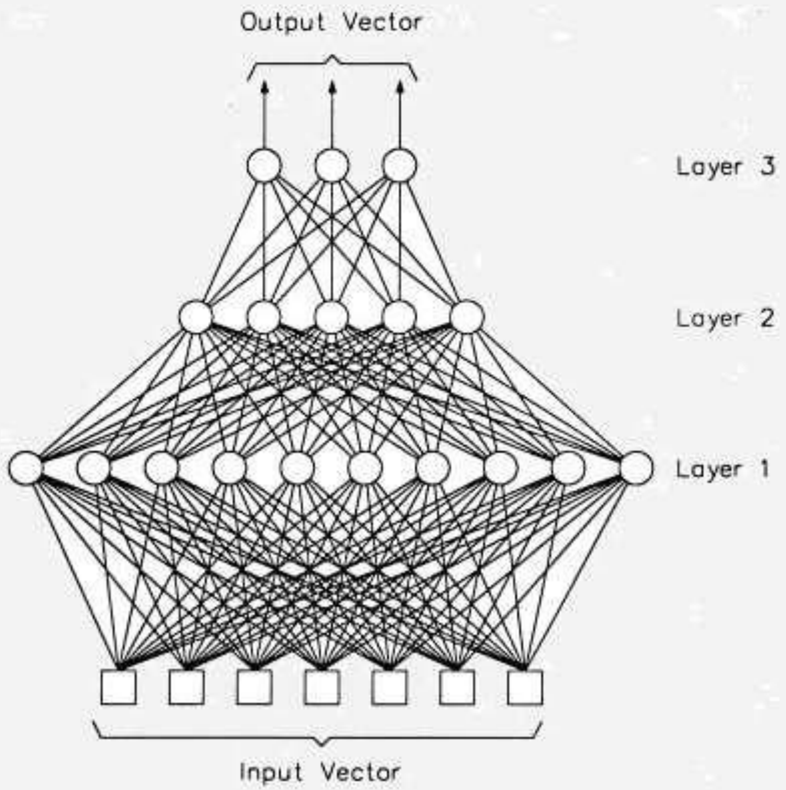
- [이슈] 클래스불균형(Class imbalance)



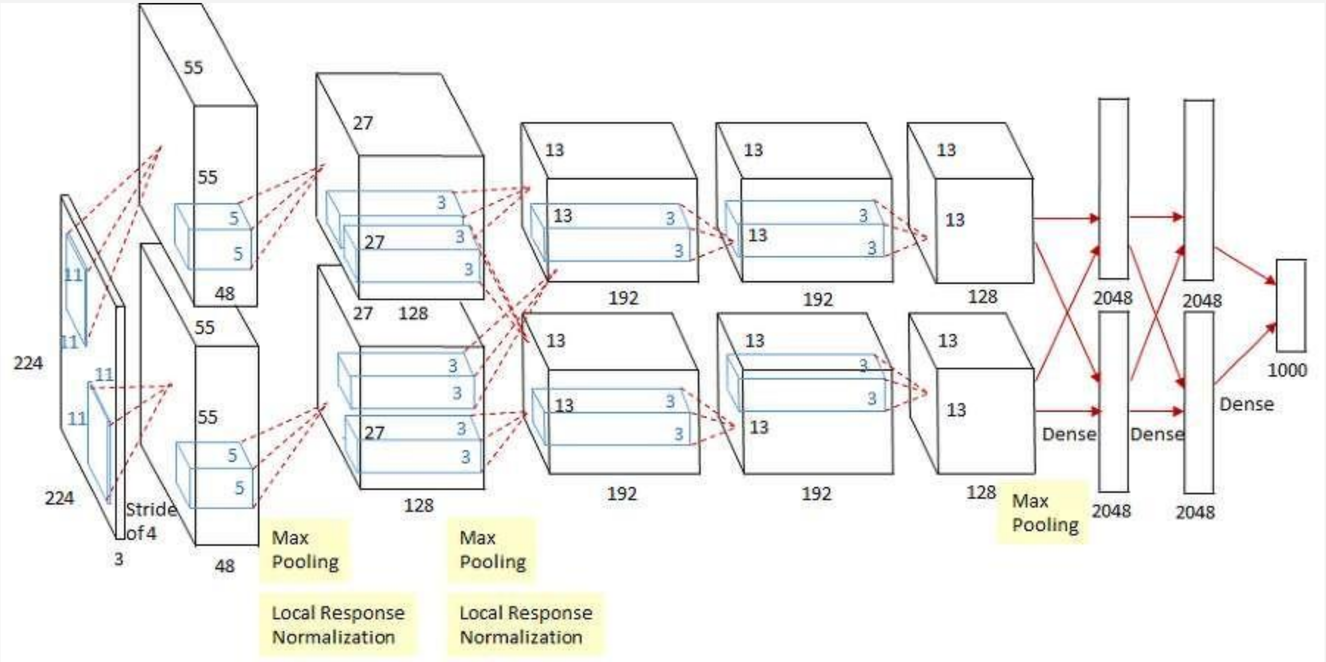
1. Oversampling or undersampling
2. Data Augmentation
3. Dropout
4. Regularization

CNN (Convolutional Neural Network)

https://deepestdocs.readthedocs.io/en/latest/004_deep_learning_part_2/0040/



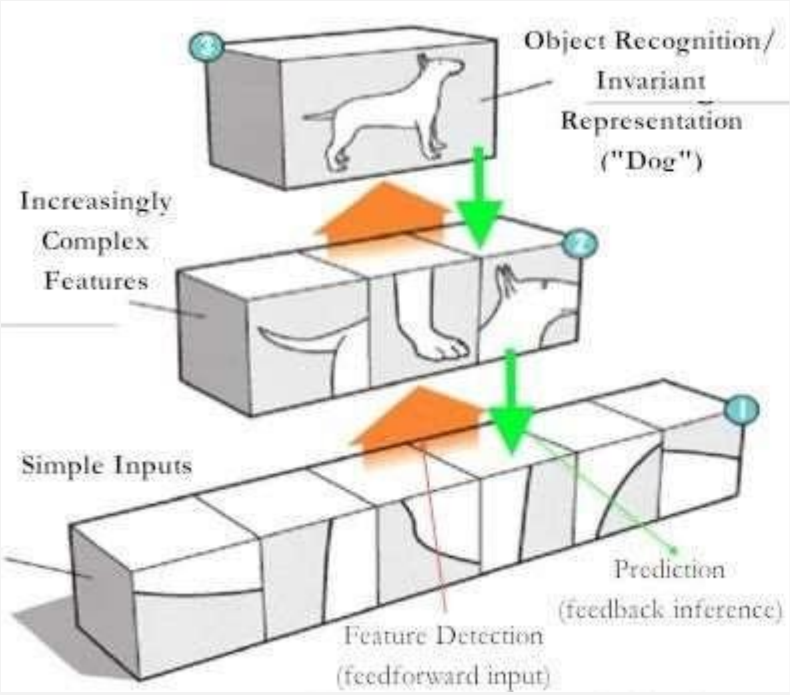
MLP(Multi-Layer Perceptron)

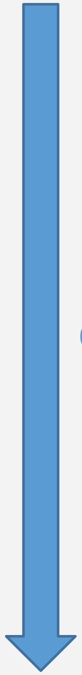


CNN (AlexNet)

CNN (Convolutional Neural Network)

https://i.pinimg.com/474x/5c/50/97/5c50979ab722d68a0b8fec2a98ba1f5c--deep-learning.jpg





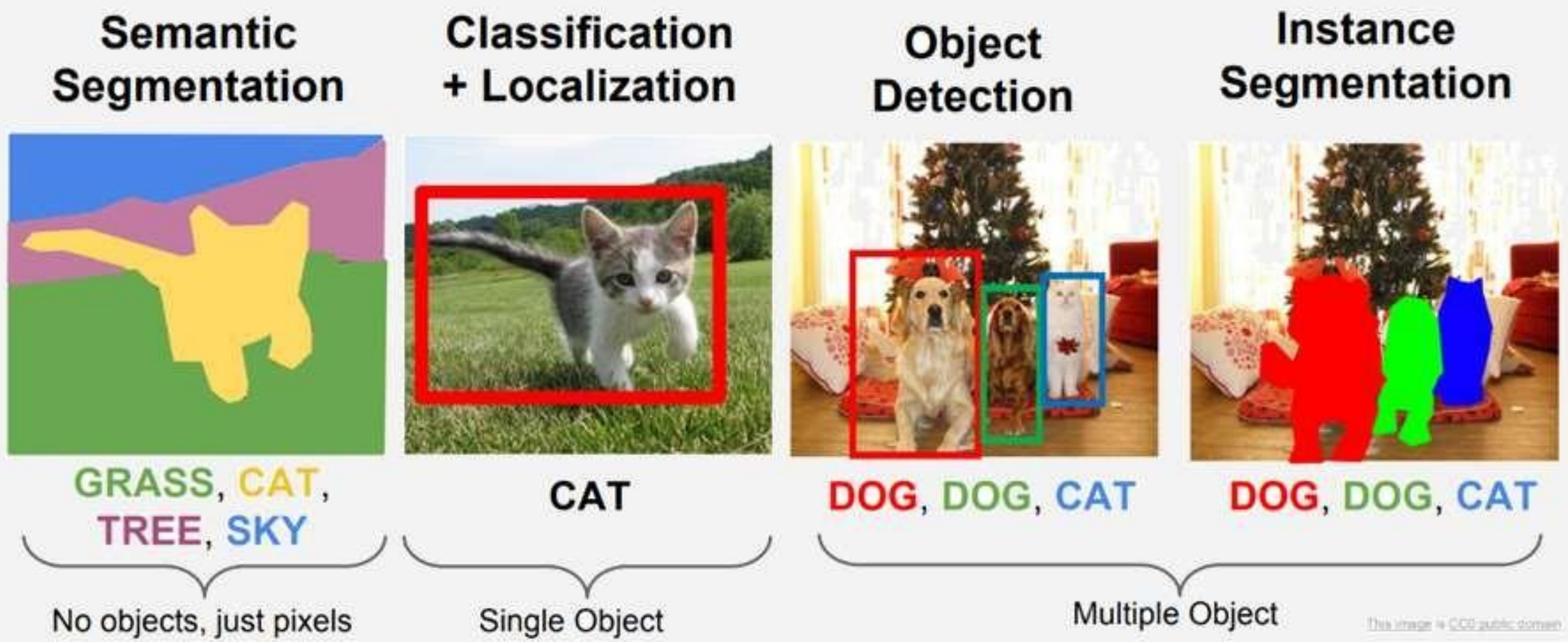
- Human vision system

  simple cells → complex cells → hyper-complex cells

CNN (Convolutional Neural Network)

https://blog.naver.com/PostView.nhn?isHttpsRedirect=true&blogId=dnjswns2280&logNo=222045224532&parentCategoryNo=&categoryNo=10&viewDate=&isShowPopularPosts=false&from=postView

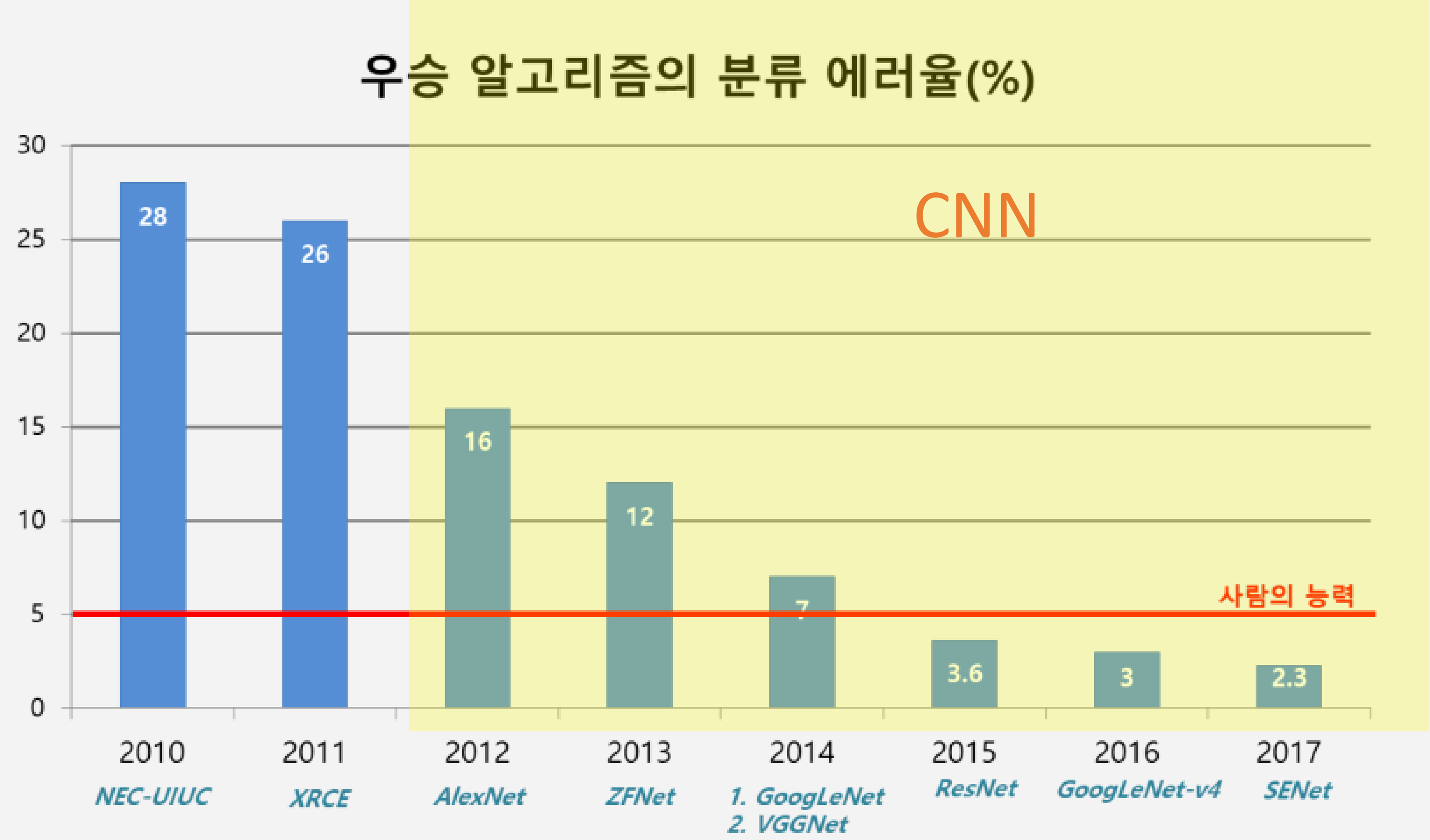
- Computer Vision



CNN (Convolutional Neural Network)

https://bskyvision.com/entry/ILSVRC-%EB%8C%80%ED%9A%8C-%EC%9D%B4%EB%AF%B8%EC%A7%80%EB%84%B7-%EC%9D%B4%EB%AF%B8%EC%A7%80-%EC%9D%B8%EC%8B%9D-%EB%8C%80%ED%9A%8C-%EC%97%AD%EB%8C%80-%EC%9A%B0%EC%8A%B9-%EC%95%8C%EA%B3%A0%EB%A6%AC%EC%A6%98%EB%93%A4

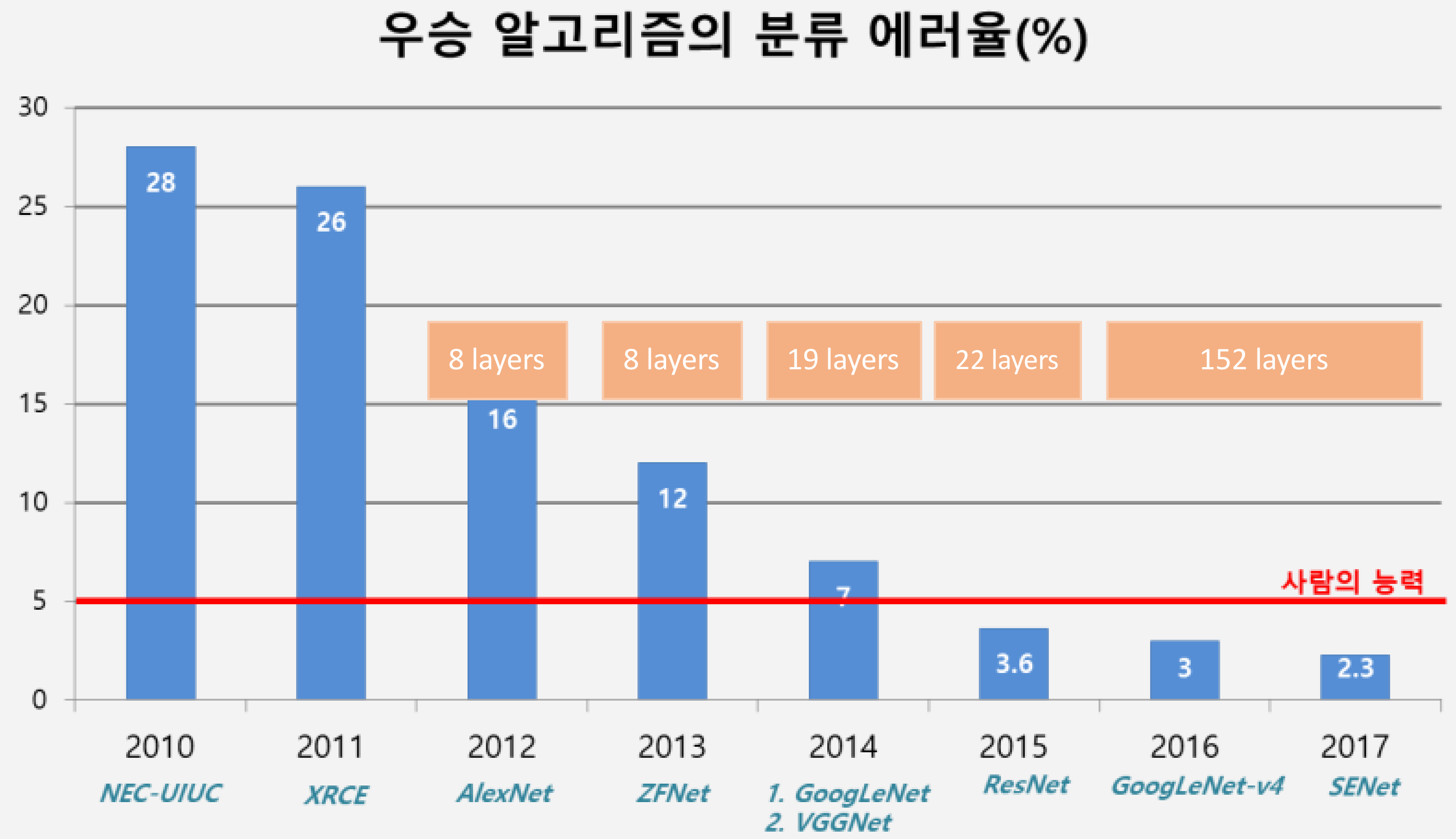
- ILSVRC(ImageNet Large Scale Visual Recognition Challenge)



CNN (Convolutional Neural Network)

https://bskyvision.com/entry/ILSVRC-%EB%8C%80%ED%9A%8C-%EC%9D%B4%EB%AF%B8%EC%A7%80%EB%84%B7-%EC%9D%B4%EB%AF%B8%EC%A7%80-%EC%9D%B8%EC%8B%9D-%EB%8C%80%ED%9A%8C-%EC%97%AD%EB%8C%80-%EC%9A%B0%EC%8A%B9-%EC%95%8C%EA%B3%A0%EB%A6%AC%EC%A6%98%EB%93%A4

- ILSVRC(ImageNet Large Scale Visual Recognition Challenge)



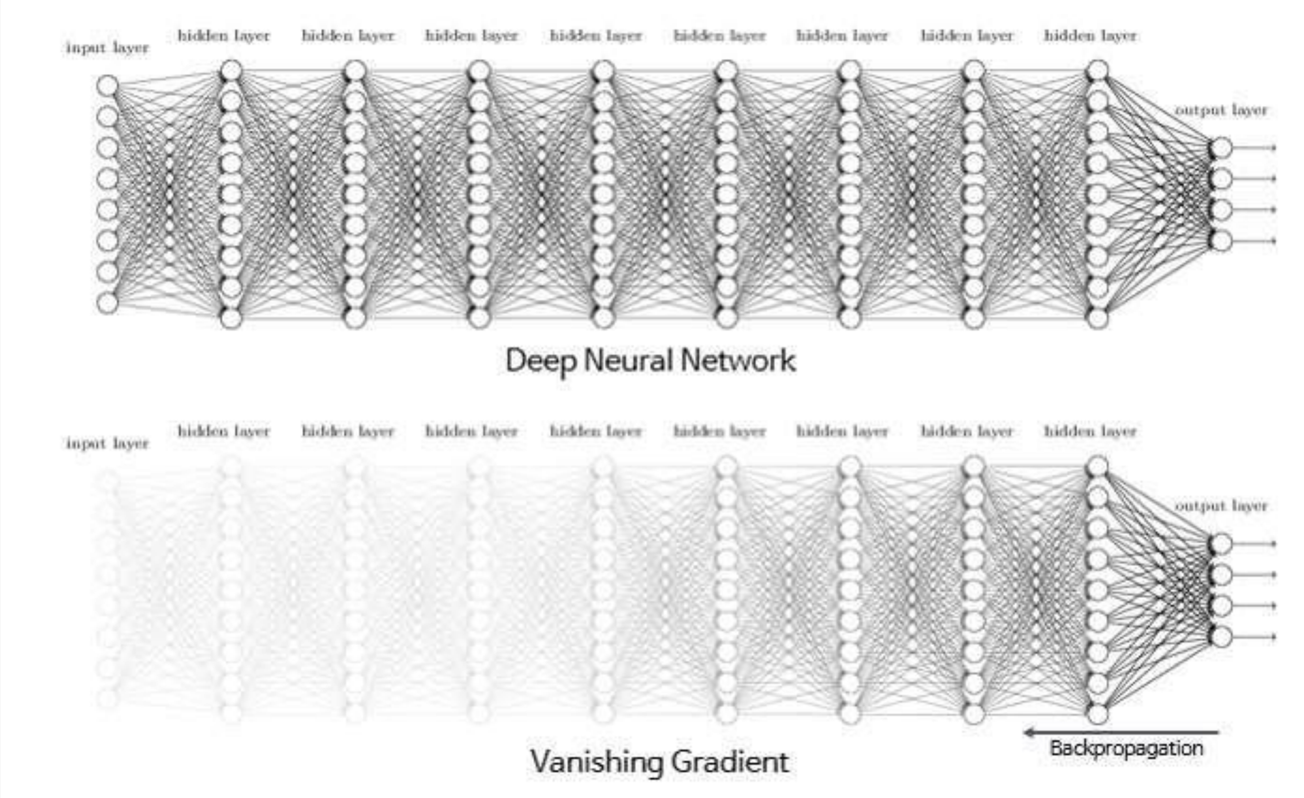
Deep 할수록 무조건 좋을까?

과학기술정보통신부 NIA 한국지능정보사회진흥원

CNN (Convolutional Neural Network)

https://heytech.tistory.com/388

- Vanishing Gradient



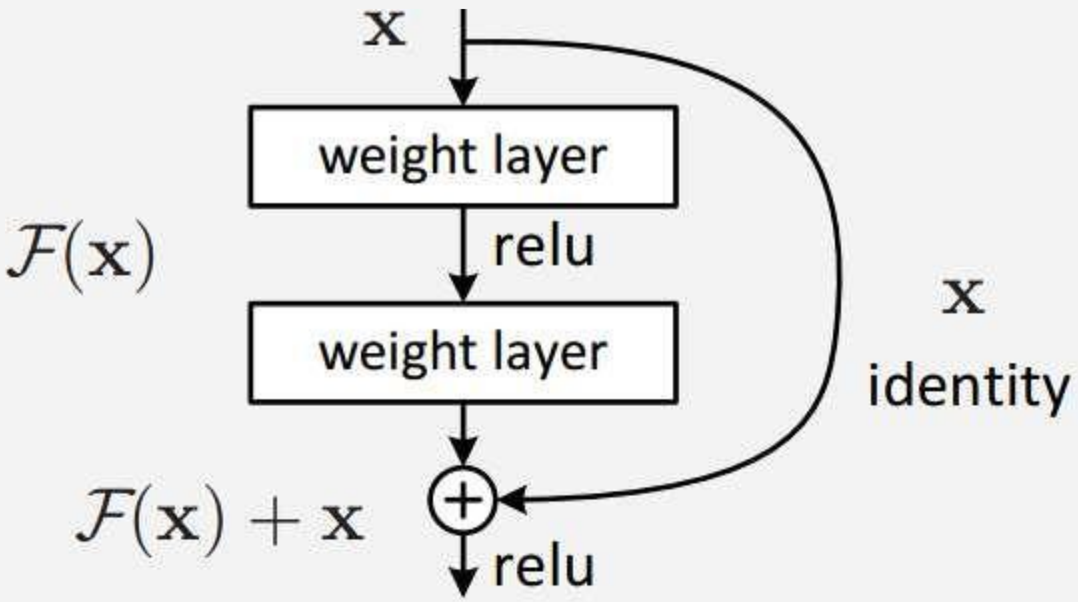
$$\theta \leftarrow \theta - \varepsilon \frac{\partial J}{\partial \theta}$$

J : cost function(loss function)
e : step size(learning rate)

CNN (Convolutional Neural Network)

https://towardsdatascience.com/residual-blocks-building-blocks-of-resnet-fd90ca15d6ec

- Skip connection

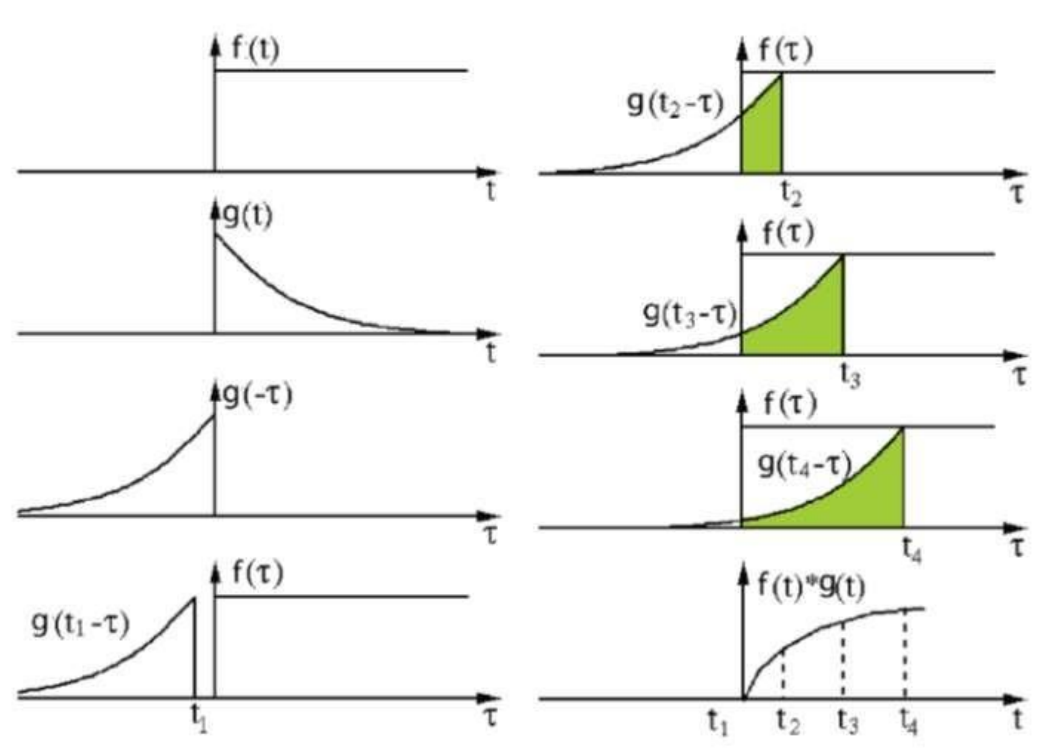


Convolution

https://towardsdatascience.com/a-comprehensive-introduction-to-different-types-of-convolutions-in-deep-learning-669281e58215

공학적 의미의 convolution : $(f * g)(t) \stackrel{\text{def}}{=} \int_{-\infty}^{\infty} f(\tau)\, g(t-\tau)\, d\tau$ $[f * g](i, j) = \sum_{p,q} f(p,q) g(i-p, j-q)$

1D-continuous 2D-discrete

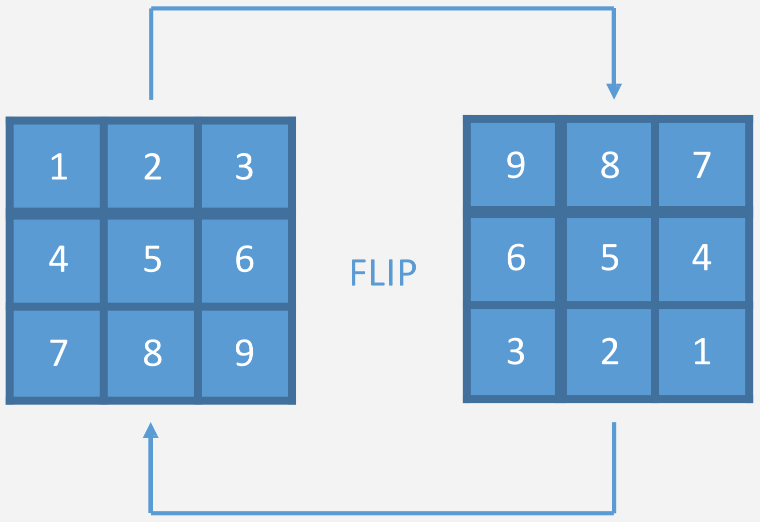


Convolution

CNN에서의 convolution : $[f * g](i,j) = \sum_{p,q} f(p,q)g(i+p,j+q)$ (엄밀히 말하면 Cross-correlation)

2D-discrete

- f : input
- g : kernel (or filter)
- output : feature map (or activation map)



- In Engineering : Convolution with kernel flipping
- CNN : Convolution without kernel flipping

Convolution

$$
\begin{bmatrix}
1 & 0 & 1 & 0 & 1 & 0 \\
0 & 1 & 1 & 0 & 1 & 1 \\
1 & 0 & 1 & 0 & 1 & 0 \\
1 & 0 & 1 & 1 & 1 & 0 \\
0 & 1 & 1 & 0 & 1 & 1 \\
1 & 0 & 1 & 0 & 1 & 0
\end{bmatrix}
*
\begin{bmatrix}
1 & 0 & 1 \\
0 & 1 & 0 \\
1 & 0 & 1
\end{bmatrix}
=
\begin{bmatrix}
\boxed{5} & & & \\
 & & & \\
 & & & \\
 & & &
\end{bmatrix}
$$

Convolution

$$
\begin{bmatrix}
1 & 0 & 1 & 0 & 1 & 0 \\
0 & 1 & 1 & 0 & 1 & 1 \\
1 & 0 & 1 & 0 & 1 & 0 \\
1 & 0 & 1 & 1 & 1 & 0 \\
0 & 1 & 1 & 0 & 1 & 1 \\
1 & 0 & 1 & 0 & 1 & 0
\end{bmatrix}
*
\begin{bmatrix}
1 & 0 & 1 \\
0 & 1 & 0 \\
1 & 0 & 1
\end{bmatrix}
=
\begin{bmatrix}
5 & 1 & & \\
 & & & \\
 & & & \\
 & & &
\end{bmatrix}
$$

Convolution

$$\begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 5 & 1 & 4 & \; \\ \; & \; & \; & \; \\ \; & \; & \; & \; \\ \; & \; & \; & \; \end{bmatrix}$$

Convolution

$$
\begin{bmatrix}
1 & 0 & 1 & 0 & 1 & 0 \\
0 & 1 & 1 & 0 & 1 & 1 \\
1 & 0 & 1 & 0 & 1 & 0 \\
1 & 0 & 1 & 1 & 1 & 0 \\
0 & 1 & 1 & 0 & 1 & 1 \\
1 & 0 & 1 & 0 & 1 & 0
\end{bmatrix}
*
\begin{bmatrix}
1 & 0 & 1 \\
0 & 1 & 0 \\
1 & 0 & 1
\end{bmatrix}
=
\begin{bmatrix}
5 & 1 & 4 & \boxed{1} \\
 & & & \\
 & & & \\
 & & &
\end{bmatrix}
$$

Convolution

$$\begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 5 & 1 & 4 & 1 \\ \boxed{3} & & & \\ & & & \\ & & & \end{bmatrix}$$

Convolution

$$
\begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix}
*
\begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}
=
\begin{bmatrix} 5 & 1 & 4 & 1 \\ 3 & 3 & 4 & 3 \\ 3 & 2 & 5 & 2 \\ 5 & 2 & 4 & 2 \end{bmatrix}
$$

input image kernel feature map

$$O = \frac{I - K + 2P}{S} + 1$$

Convolution

https://anhreynolds.com/blogs/cnn.html

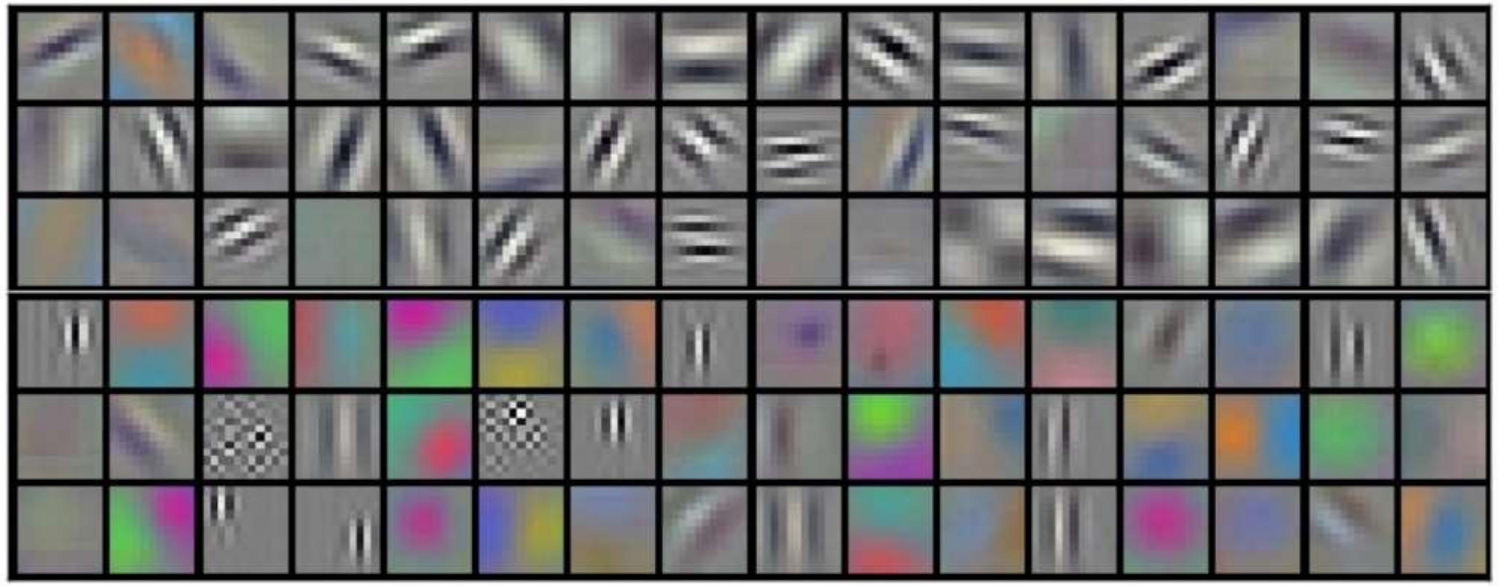
- Kernel의 역할 → feature detector

$$
\begin{bmatrix}
10 & 10 & 10 & 10 & 0 & 0 & 0 & 0 \\
10 & 10 & 10 & 10 & 0 & 0 & 0 & 0 \\
10 & 10 & 10 & 10 & 0 & 0 & 0 & 0 \\
10 & 10 & 10 & 10 & 0 & 0 & 0 & 0 \\
10 & 10 & 10 & 10 & 0 & 0 & 0 & 0 \\
10 & 10 & 10 & 10 & 0 & 0 & 0 & 0 \\
10 & 10 & 10 & 10 & 0 & 0 & 0 & 0 \\
10 & 10 & 10 & 10 & 0 & 0 & 0 & 0
\end{bmatrix}
*
\underbrace{\begin{bmatrix}
1 & 0 & -1 \\
1 & 0 & -1 \\
1 & 0 & -1
\end{bmatrix}}_{\text{Vertical}}
=
\begin{bmatrix}
0 & 0 & 30 & 30 & 0 & 0 \\
0 & 0 & 30 & 30 & 0 & 0 \\
0 & 0 & 30 & 30 & 0 & 0 \\
0 & 0 & 30 & 30 & 0 & 0 \\
0 & 0 & 30 & 30 & 0 & 0 \\
0 & 0 & 30 & 30 & 0 & 0
\end{bmatrix}
$$

vertical edge detector

Convolution

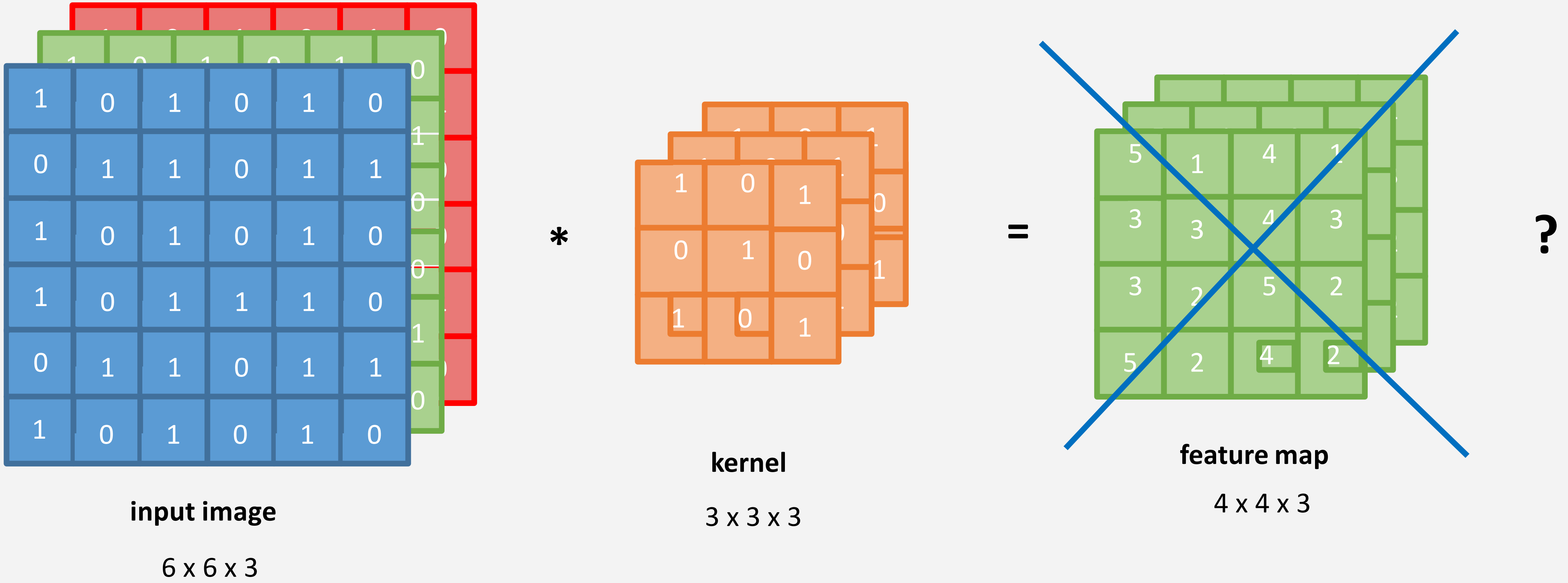
A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks”, NIPS, 2012



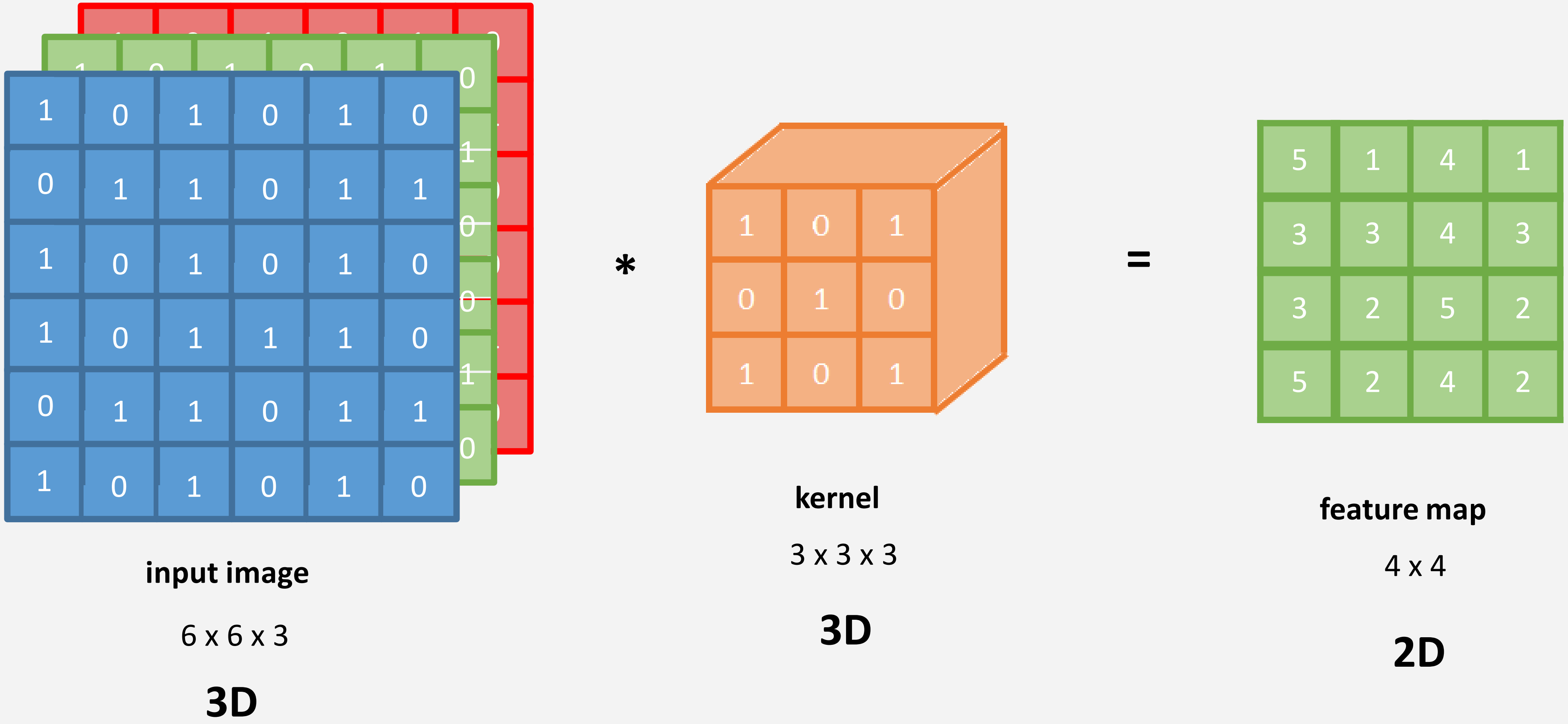
AlexNet
96 filters (size : 11 x 11)

학습해서 찾아낸 필터들

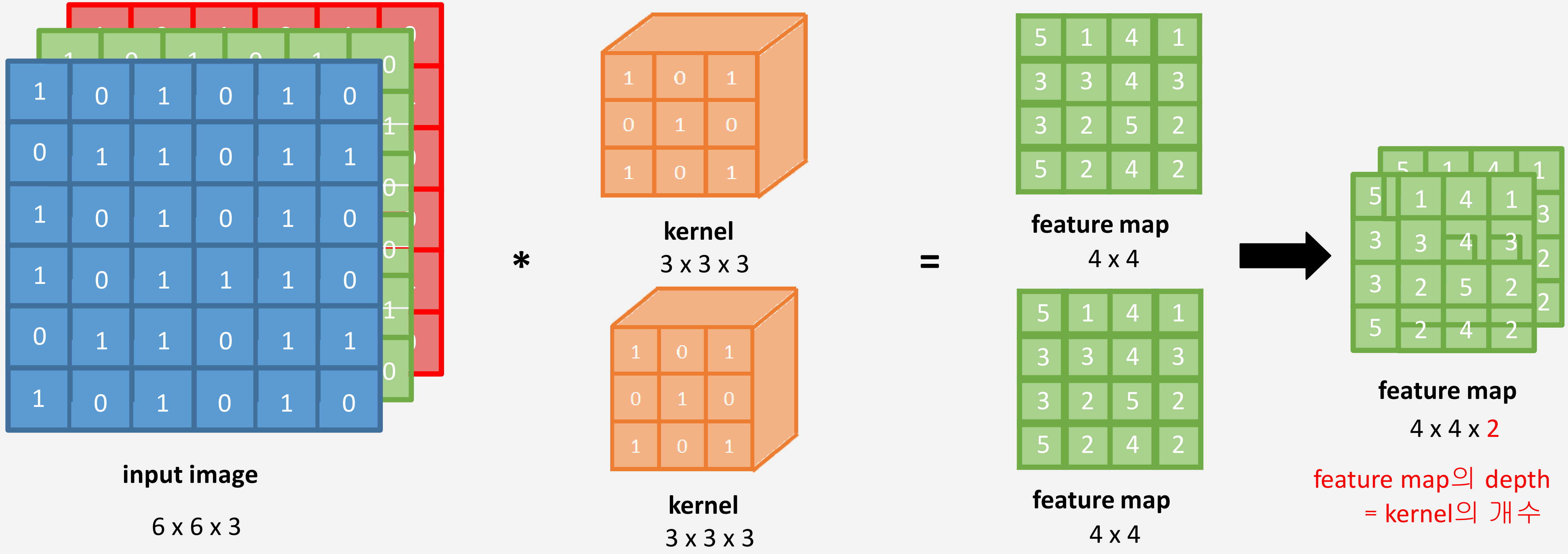
Convolution



Convolution



Convolution



Convolution

O(output) = **I(input)** * **K(kernel)**

[i , j , k] [i , j , k] [i , j , k , l]

i : channel
j : row(output)
k : column(output)

i : channel
j : row (input)
k : column(input)

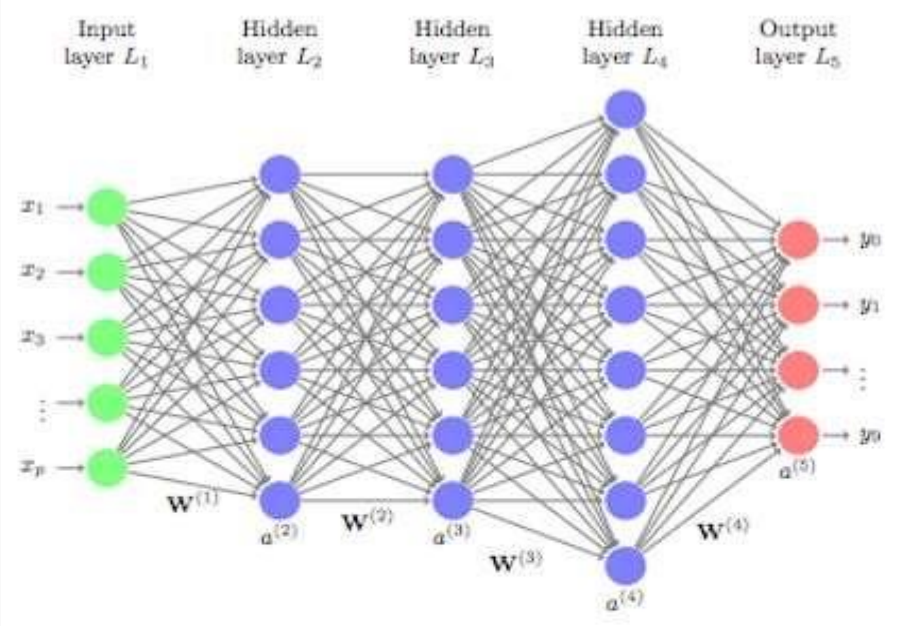
i : output channel
j : input channel
k : row(kernel)
l : column(kernel)

Ex) [3, 128, 128] * [9, 3, 5, 5] → [9, 124, 124]
[3, 360, 480] * [1, 3, 3, 3] → [1, 358, 478]
input kernel output

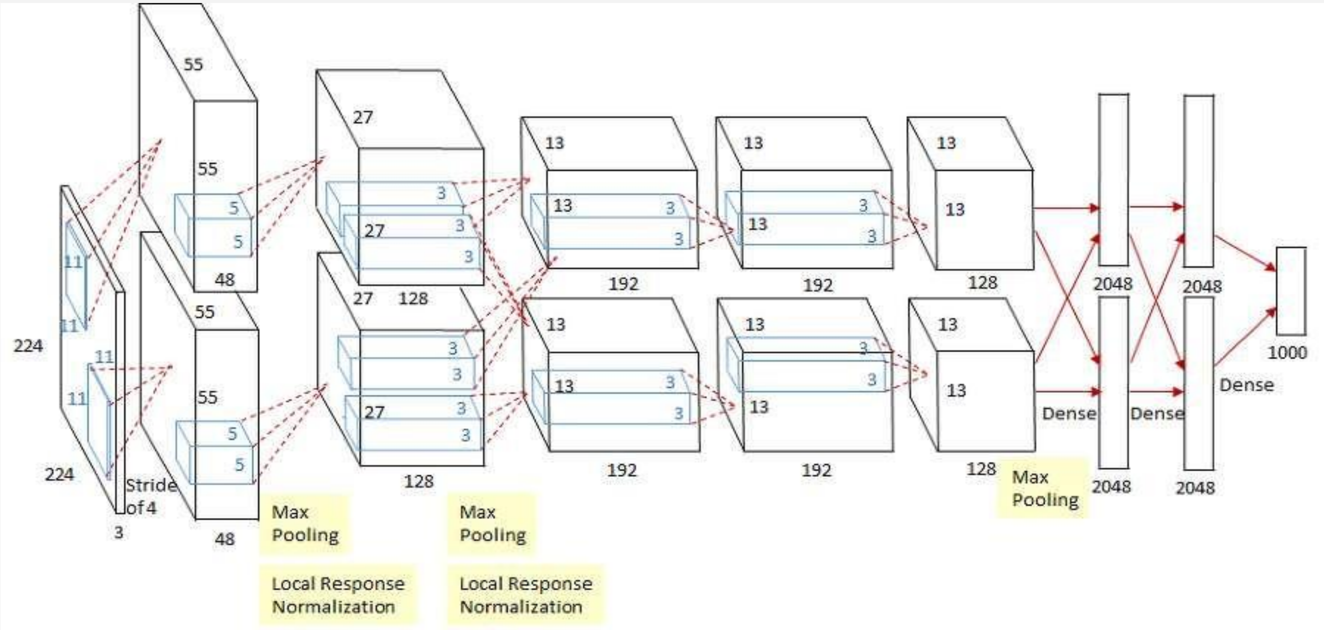
MLP vs CNN

http://uc-r.github.io/feedforward_DNN

MLP



CNN(AlexNet)



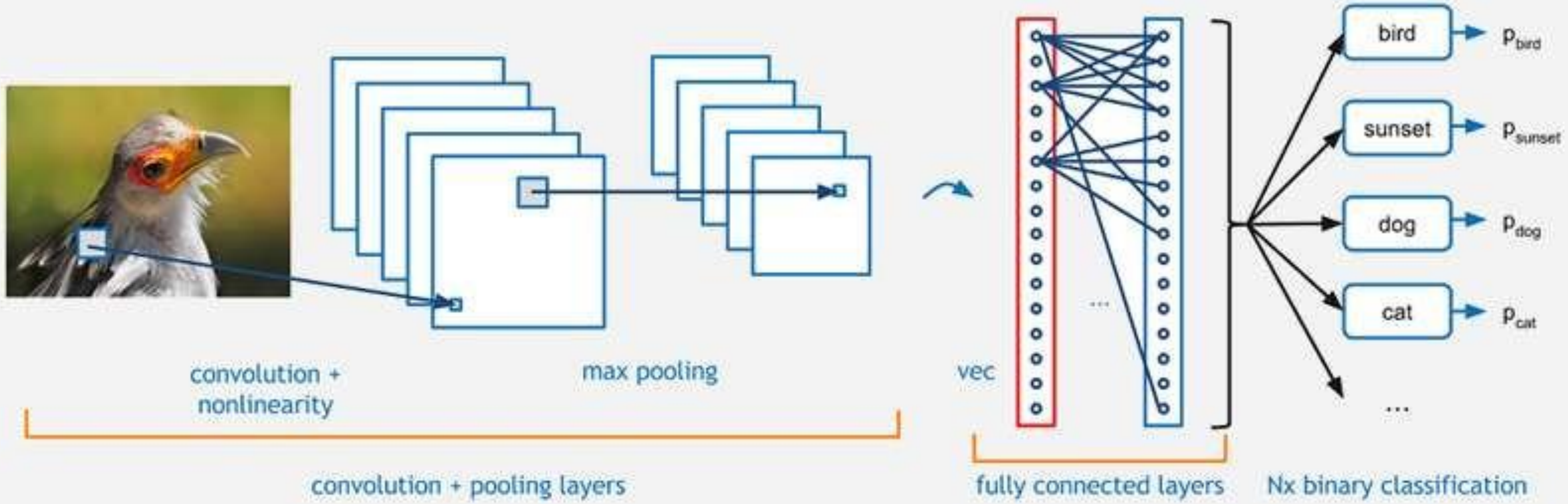
- 뉴런이 이전 layer의 모든 뉴런과 연결(fully connected)
- 뉴런이 모두 독립적으로 작용
- Overfitting이 일어나기 쉽다
- 각 뉴런의 Weight, bias를 학습한다

- 뉴런이 이전 layer의 일부 뉴런과만 연결(locally connected)
- Parameter가 서로 sharing 된다(같은 필터 사용)
- Computation 측면에서 Fully connected에 비해 훨씬 효율적
- Fully connected에 비해 overfitting도 덜 된다
- Kernel을 학습한다

과학기술정보통신부 NIA 한국지능정보사회진흥원

CNN architecture

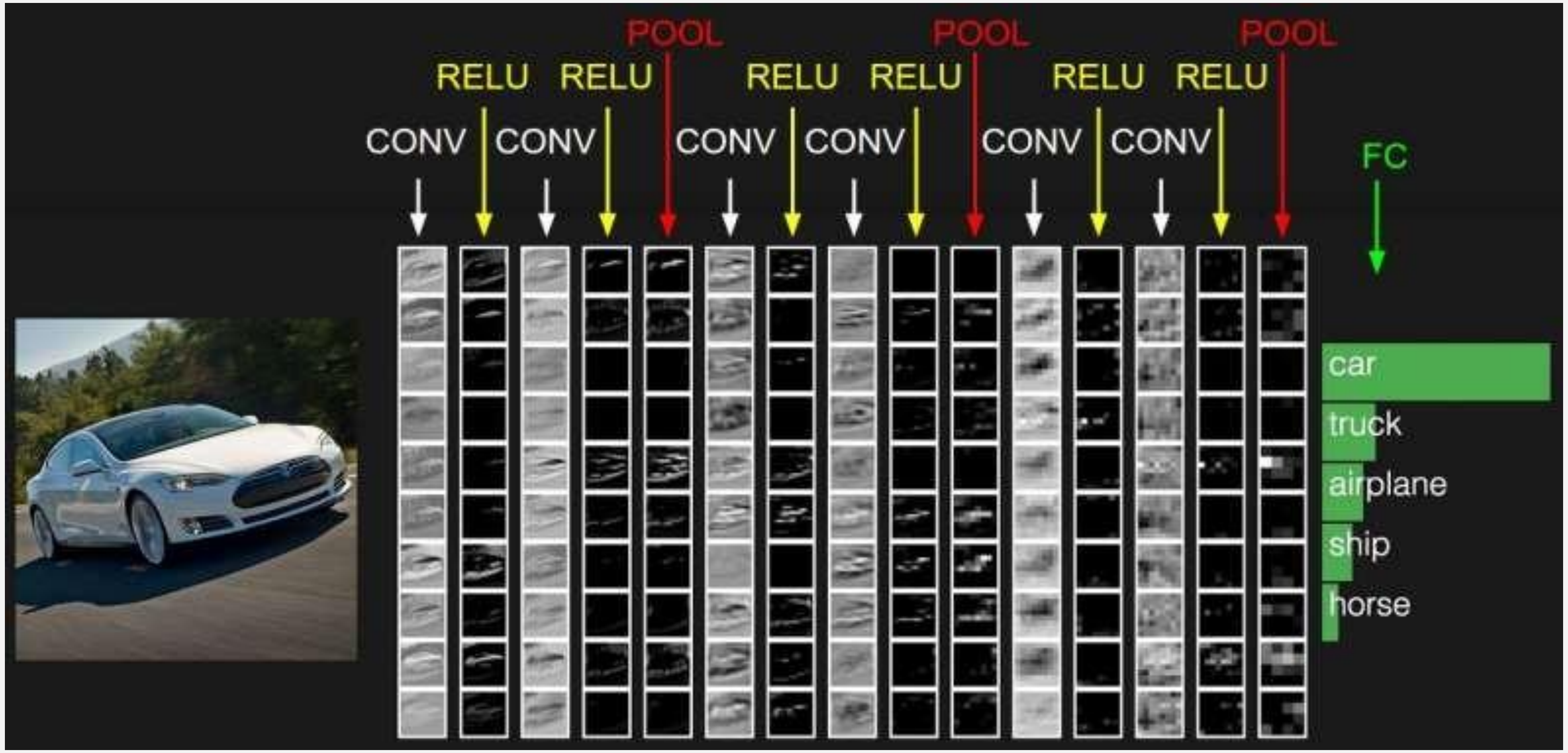
https://dalsacoo-log.tistory.com/entry/what-is-CNN



- Convolutional layer
- ReLU layer
- Pooling layer
- Fully connected layer

CNN architecture

https://cs231n.github.io/convolutional-networks/



Convolutional Layer

- Parameter : 有,   Hyperparameter : 有

  Parameter : Kernel이 parameter의 집합이다(weight, bias)

  Hyper parameter : kernel 수, kernel size 등

- CNN의 가장 중요한 layer

  kernel의 dimension으로 output의 dimension이 정해진다

  Ex) [3, 128, 128] * [9, 3, 5, 5] ➔ [9, 124, 124]

  [3, 360, 480] * [1, 3, 3, 3] ➔ [1, 358, 478]

  input   kernel   output

  하나의 **filter** = 하나의 **feature map**

Convolutional Layer

- 장점
  - Parameter Sharing
    - ✓ 같은 kernel을 사용하여 feature map이 나온다
  - Flexibility
    - ✓ 어떤 사이즈의 input이 들어오더라도 채널만 맞춰주면 학습이 가능하다
  - Efficiency
    - ✓ MLP에 비해 computation이 훨씬 적기 때문에 효율적이다

Convolutional Layer

- Zero Padding

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

효과 :

1. Output의 크기가 너무 빨리 줄어드는 것을 방지
2. 이미지의 가장자리보다 안쪽 부분에 더 집중하도록 함

ReLU Layer

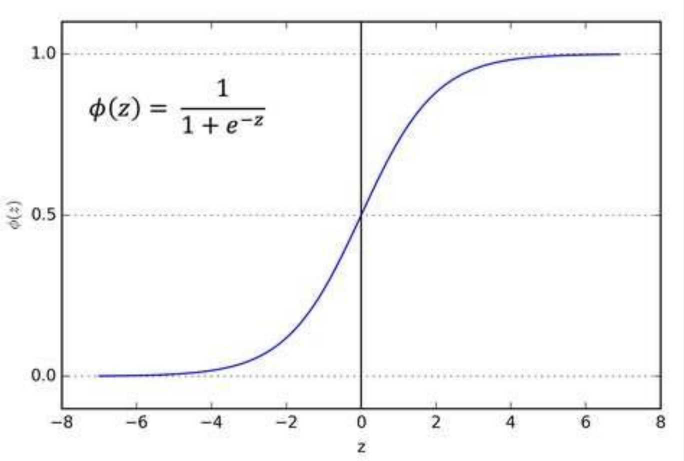
- Parameter : 無, Hyperparameter : 無
- Non-linearity를 증가시키기 위해 사용
- CNN에서 activation function은 거의 ReLU만 사용
- MLP에서도 hidden layer에서는 sigmoid 등의 다른 activation function을 쓰지 않고, ReLU만 사용

    (ex. output layer에서는 sigmoid 사용)

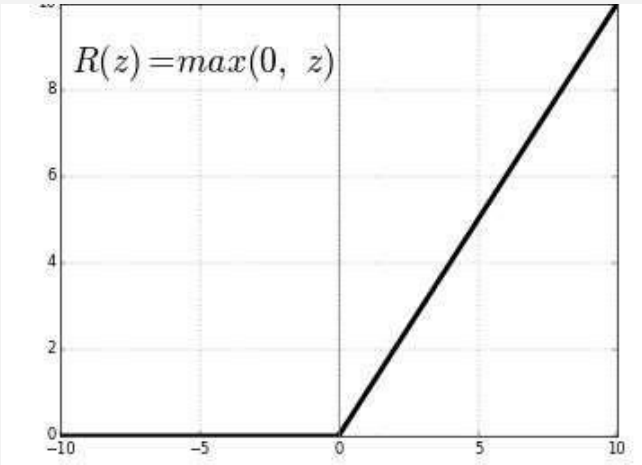
ReLU Layer

https://m.blog.naver.com/zzoyou_/222014804966

- 장점
  - Sigmoid와 달리 vanishing gradient의 문제가 없다
  - 계산이 효율적이다
  - 수렴속도가 sigmoid보다 6배정도 빠르다
- 단점
  - 중심값이 0이 아니다
  - 입력값이 0보다 작을 때 gradient가 0이 된다



Sigmoid



ReLU

Pooling Layer

- Parameter : 無,　Hyperparameter : 有

    Hyperparameter : Pooling kernel의 개수, kernel의 사이즈

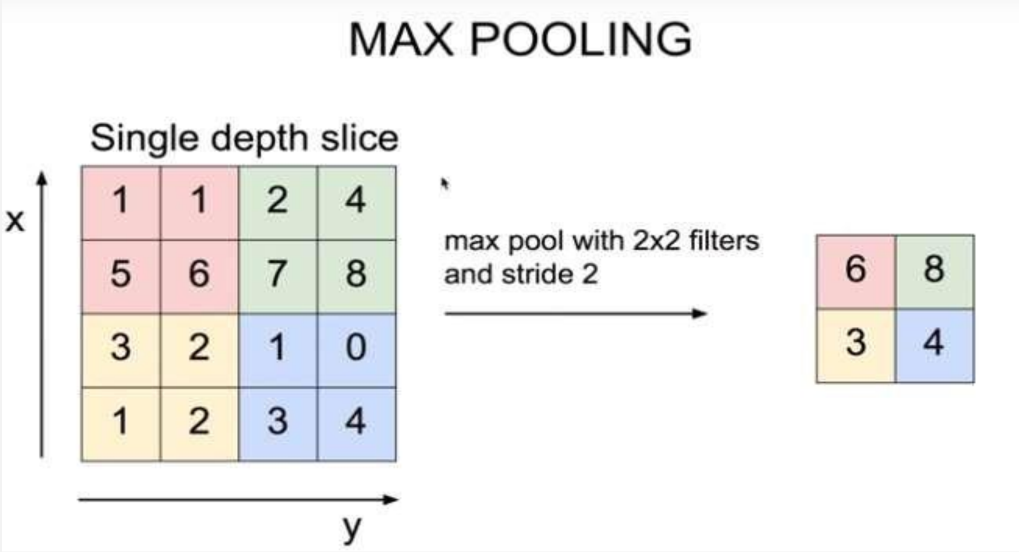
- Max pooling, Average pooling … 등등

    현재는 CNN에서는 거의 max pooling만 사용

Pooling Layer

https://hobinjeong.medium.com/cnn%EC%97%90%EC%84%9C-pooling%EC%9D%B4%EB%9E%80-c4e01aa83c83

- Row, column이 모두 절반씩으로 줄어든다(Maxpool2D)
  - 75%의 activation이 사라진다
- Depth에는 영향을 주지 않는다
- 데이터의 차원 감소
  - Neural network의 계산효율성 증가
- 강한 feature 만을 다음 layer로 넘긴다
- 최근에는 Max pooling 보다 convolutional layer에서 stride를 늘리는 방식을 채택하기도 한다



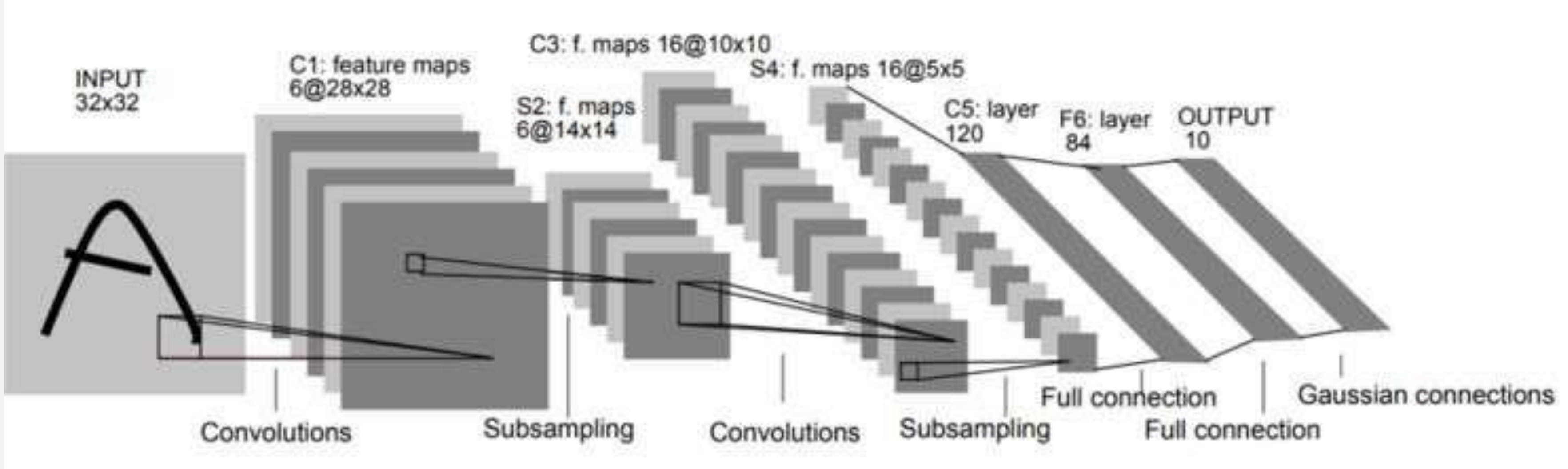
Fully-Connected Layer

- Parameter : 有,   Hyperparameter : 有 (MLP 생각)
- Dropout이 사용되기도 한다(overfitting 방지)
- Vector ➔ Vector 로 바꾸는 transformation
  - 다른 layer는 Volume ➔ Volume
- 마지막에 softmax function 사용되기도 한다
  - Multi-class classification
- CNN 모델 전체에서 대부분의 parameter는 FC layer에 존재한다
- 최근에는 FC layer의 사용을 최소화하는 추세이다

CNN architecture

Y. LeCun et al., “Gradient-based learning applied to document recognition.”, Proceedings of the IEEE, 1998

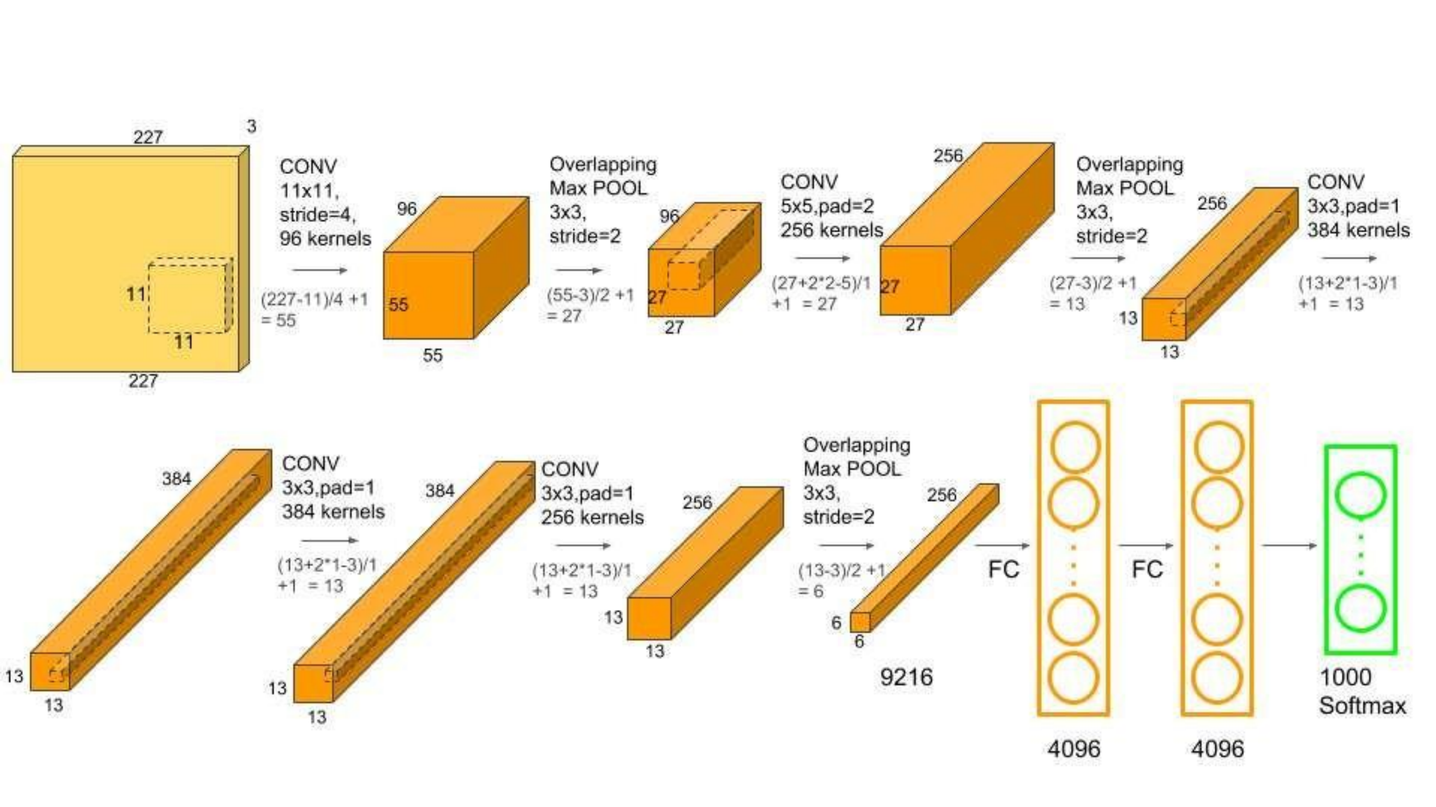
- LeNet



CNN architecture

https://seongkyun.github.io/study/2019/01/25/num_of_parameters/

- AlexNet



CNN architecture

https://vitalflux.com/what-softmax-function-why-needed-machine-learning/

- Softmax

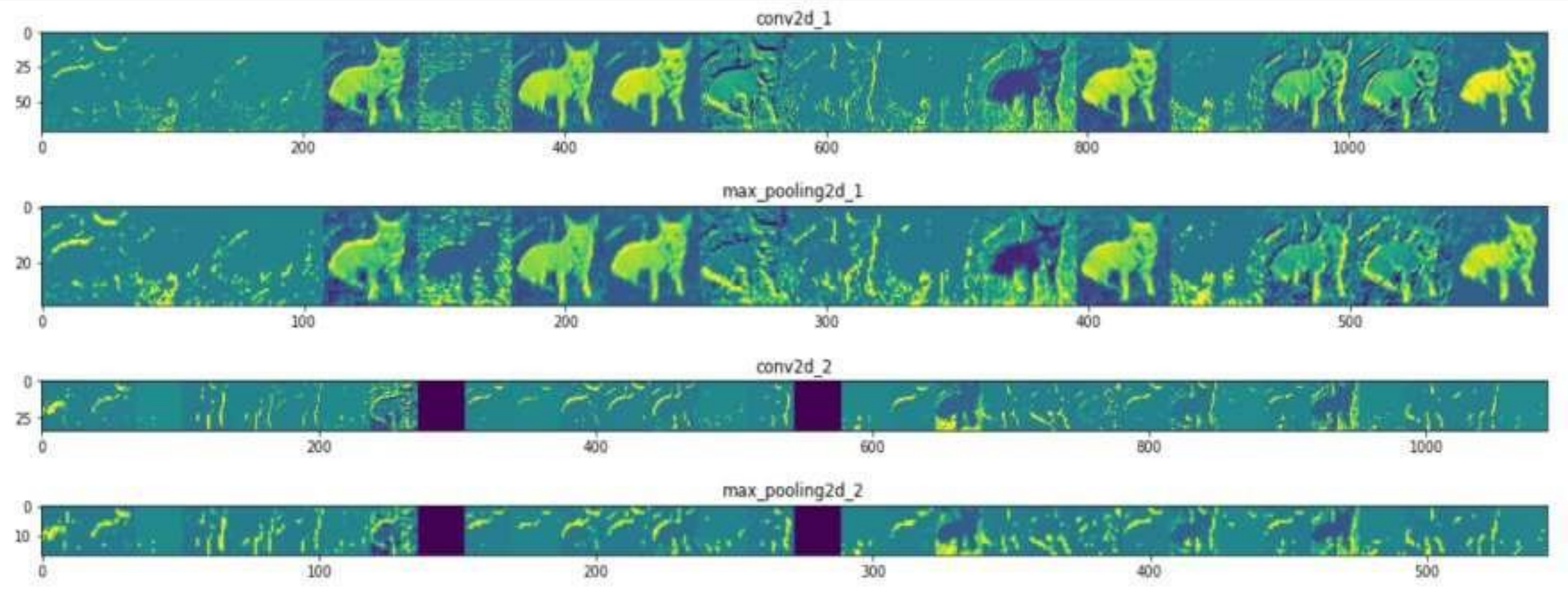
$$f_i(x) = \frac{\exp(x_i)}{\sum_j \exp(x_j)}$$

- Softmax output의 합 = 1
- 큰 건 더 크게, 작은 건 더 작게
- 확률

Convolution

https://anhreynolds.com/blogs/cnn.html

Original image



과학기술정보통신부 NIA 한국지능정보사회진흥원

Regularization

$$\min_{x} \underbrace{\|y - Hx\|^2}_{\substack{\text{Likelihood term} \\ \text{Data fidelity term}}} + \underbrace{\lambda R(x)}_{\substack{\text{Prior term} \\ \text{Regularization term}}}$$

Regularization

$$\min_{x} \underbrace{\overset{\text{MSE Loss}}{\boxed{\|y - Hx\|^2}}}_{\substack{\text{Likelihood term} \\ \text{Data fidelity term}}} + \underbrace{\overset{\text{Regularizer}}{\boxed{\lambda R(x)}}}_{\substack{\text{Prior term} \\ \text{Regularization term}}}$$

- L0 regularization
- L1 regularization
- L2 regularization

Overfitting을 방지하기 위해 **regularization**이 필요

Regularization

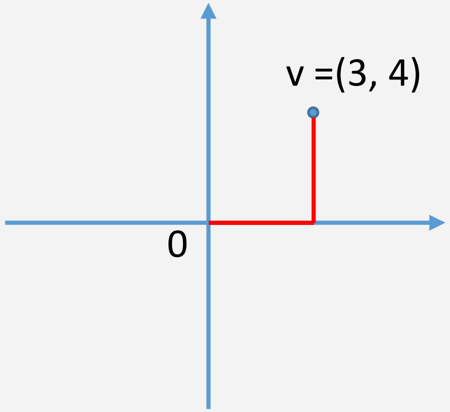
https://light-tree.tistory.com/125

- Norm

$$\|\mathbf{x}\|_p := \left(\sum_{i=1}^{n} |x_i|^p\right)^{1/p}$$

- L1-norm

$$d_1(\mathbf{p},\mathbf{q}) = \|\mathbf{p}-\mathbf{q}\|_1 = \sum_{i=1}^{n} |p_i - q_i|,\ \text{where}\ (\mathbf{p},\mathbf{q})\ \text{are vectors}\ \mathbf{p} = (p_1, p_2, \dots, p_n)\ \text{and}\ \mathbf{q} = (q_1, q_2, \dots, q_n)$$



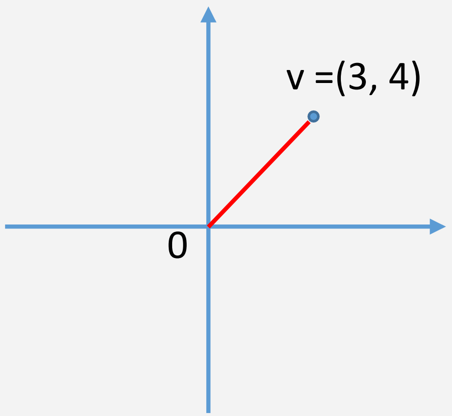
$\|v\|_1 = 3 + 4 = 7$

Regularization

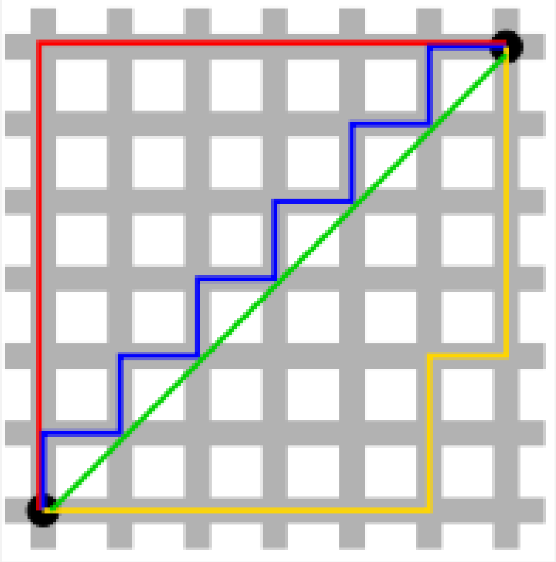
https://light-tree.tistory.com/125

- L2-norm

$$\|\boldsymbol{x}\|_2 := \sqrt{x_1^2 + \cdots + x_n^2}$$



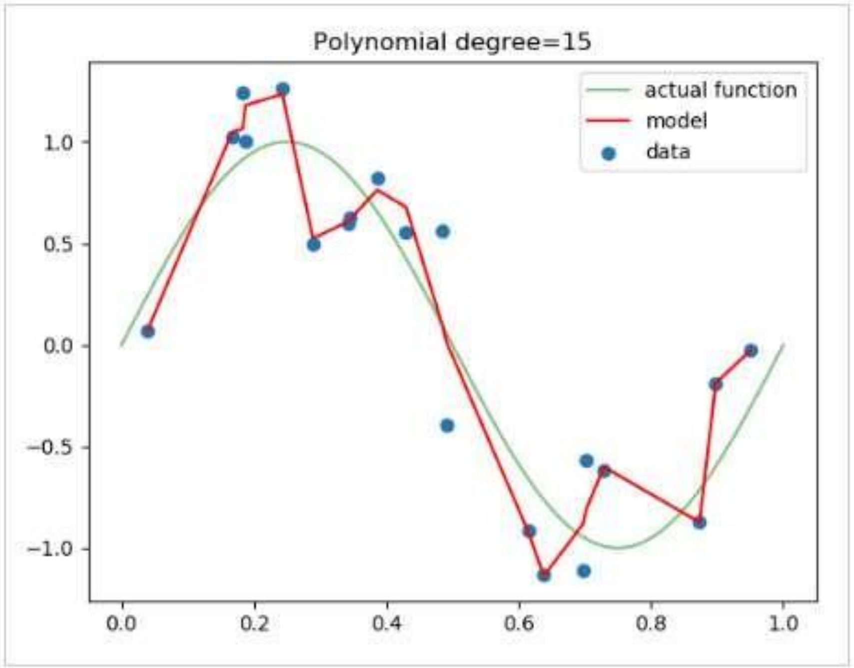
$$\|v\|_2 = \sqrt{3^2 + 4^2} - 5$$



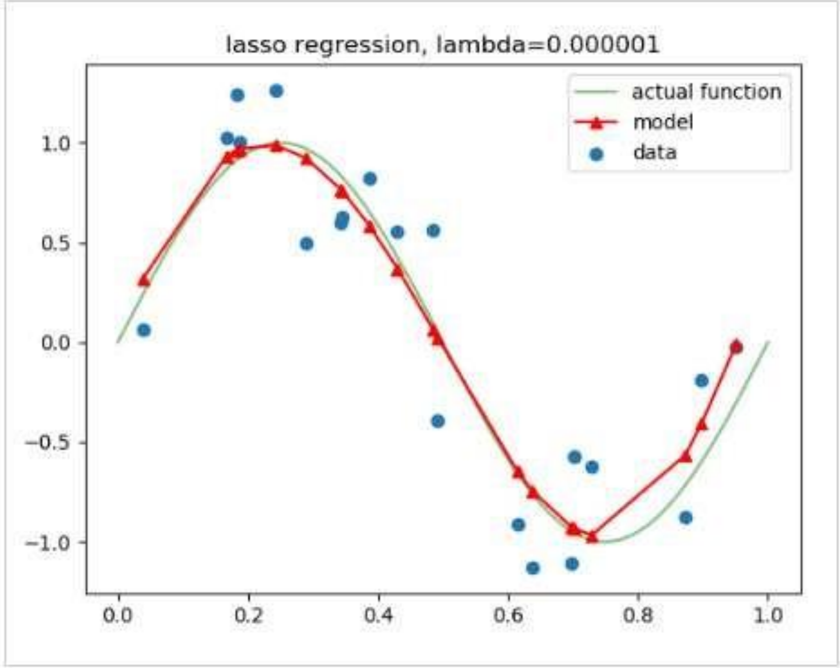
L1-Norm : 여러 경로 존재
L2-Norm : 유일한 경로 존재

Regularization

https://light-tree.tistory.com/125



Regularization



Regularization

https://light-tree.tistory.com/125

- L1 regularization

$$Cost = \frac{1}{n}\sum_{i=1}^{n}\{L(y_i, \hat{y}_i) + \frac{\lambda}{2}|w|\}$$

$L(y_i, \hat{y}_i)$: 기존의 Cost function

$$\boxed{\min_x \underbrace{\|y - Hx\|^2}_{\substack{\text{Likelihood term} \\ \text{Data fidelity term}}} + \lambda \underbrace{R(x)}_{\substack{\text{Prior term} \\ \text{Regularization term}}}}$$

LASSO Regression

(Least Absolute Shrinkage and Selection Operator)

- L2 regularization

$$Cost = \frac{1}{n}\sum_{i=1}^{n}\{L(y_i, \hat{y}_i) + \frac{\lambda}{2}|w|^2\}$$

$$\boxed{\min_x \underbrace{\|y - Hx\|^2}_{\substack{\text{Likelihood term} \\ \text{Data fidelity term}}} + \lambda \underbrace{R(x)}_{\substack{\text{Prior term} \\ \text{Regularization term}}}}$$

Ridge Regression

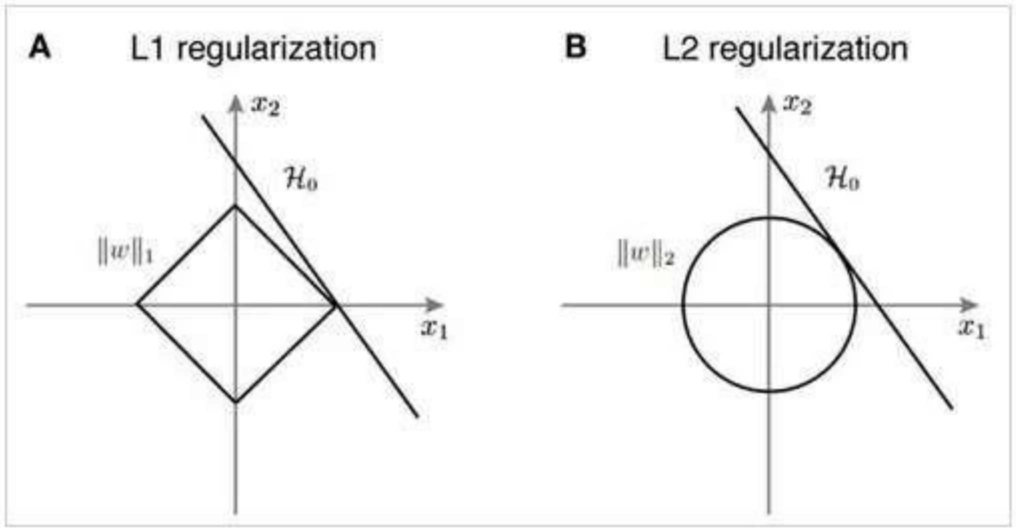
Regularization

https://light-tree.tistory.com/125

- L1 Regularization vs L2 Regularization
  - L1 regularization : Feature selection이 가능

    → Sparse coding에 적합
    → Convex optimization에 유용하게 쓰임

  - L1 regularization의 경우 미분 불가능한 점이 있기 때문에 gradient-based learning에는 주의해서 사용해야 한다



이 문서의 외부 유출 및 공유를 금합니다.

감사합니다

2022 DATA CREATOR CAMP

과학기술정보통신부 NIA 한국지능정보사회진흥원