개발자를 위한 머신러닝&딥러닝

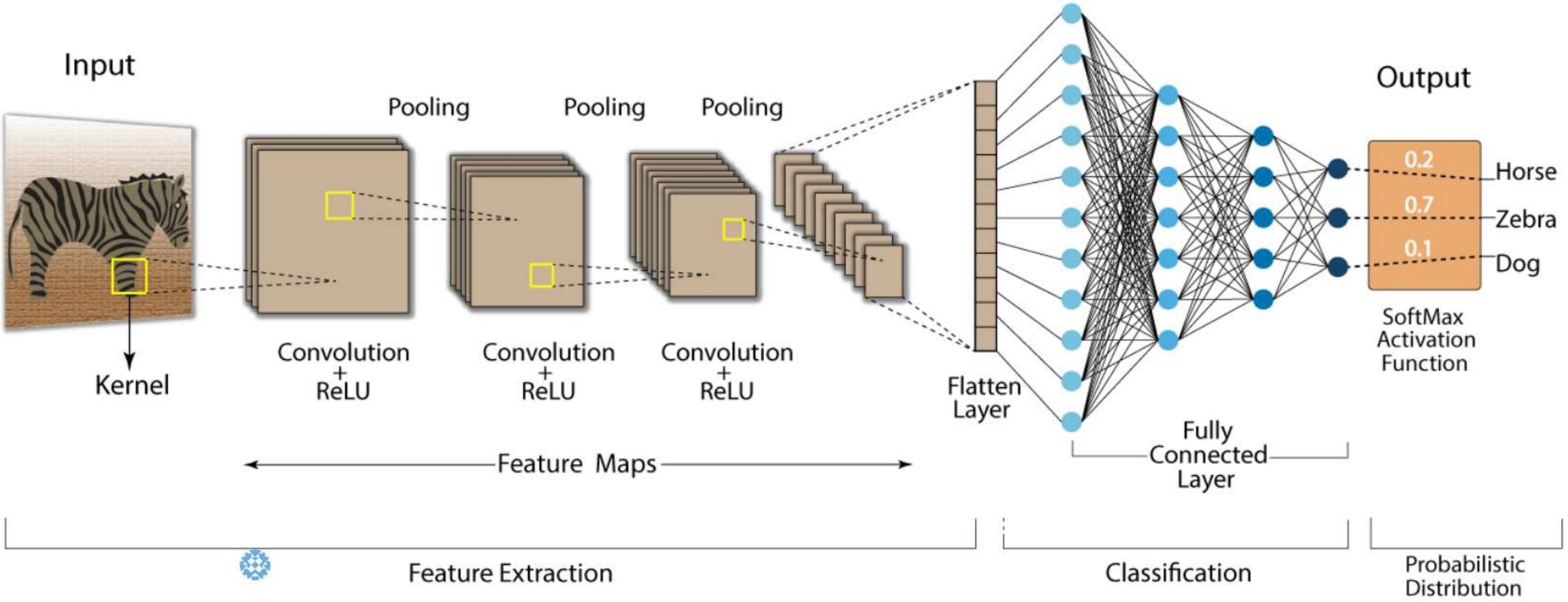
v0.0.1

2022-12-17

written by A.K.A whatwant (whatwant@whatwant.com)

CHAPTER 03
고급 컴퓨터 비전
: 이미지에서 특징 감지하기

CNN (합성곱 신경망, Convolutional Neural Network)



※ 참고 : https://developersbreach.com/convolution-neural-network-deep-learning/

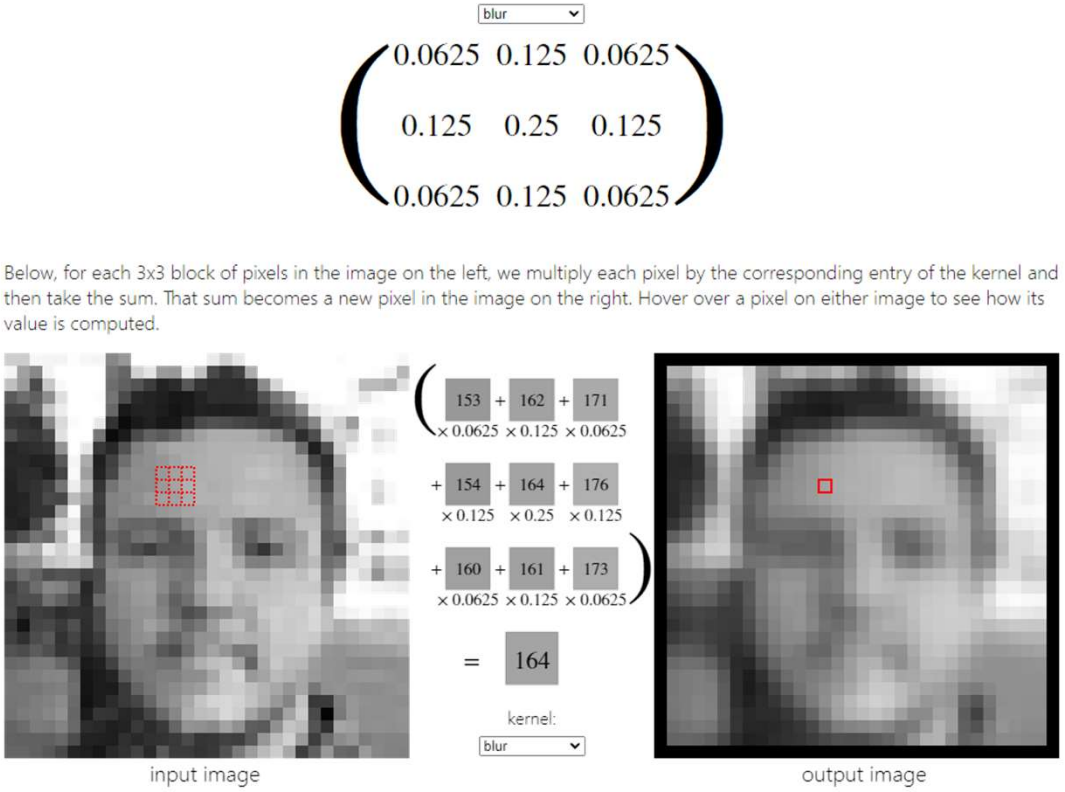
Image Kernels Explained Visually

- https://setosa.io/ev/image-kernels/

- Kernel

  . 이미지 처리에서 커널, 컨볼루션 행렬 또는 마스크는 흐리게, 선명하게 하기, 엠보싱, 가장자리 감지 등에 사용되는 작은 행렬

  . = filter = window = mask



※ 참고 : https://setosa.io/ev/image-kernels/

Stride & Padding

- https://calvinfeng.gitbook.io/machine-learning-notebook/supervised-learning/convolutional-neural-network/convolution_operation

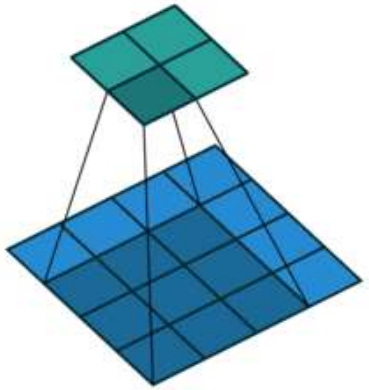
- Stride : 입력데이터에 커널(필터)를 적용할 때 커널이 이동할 간격

- Padding : 합성곱(convolution) 연산을 수행하기 전, 입력데이터 주변을 특정 값으로 채워 늘리는 것

  . 주로 출력데이터의 공간적(Spatial)크기를 조절하기 위해 사용

  . 패딩을 할 때, 채울 값은 hyperparameter로 어떤 값을 채울지 결정. 주로 zero-padding 사용

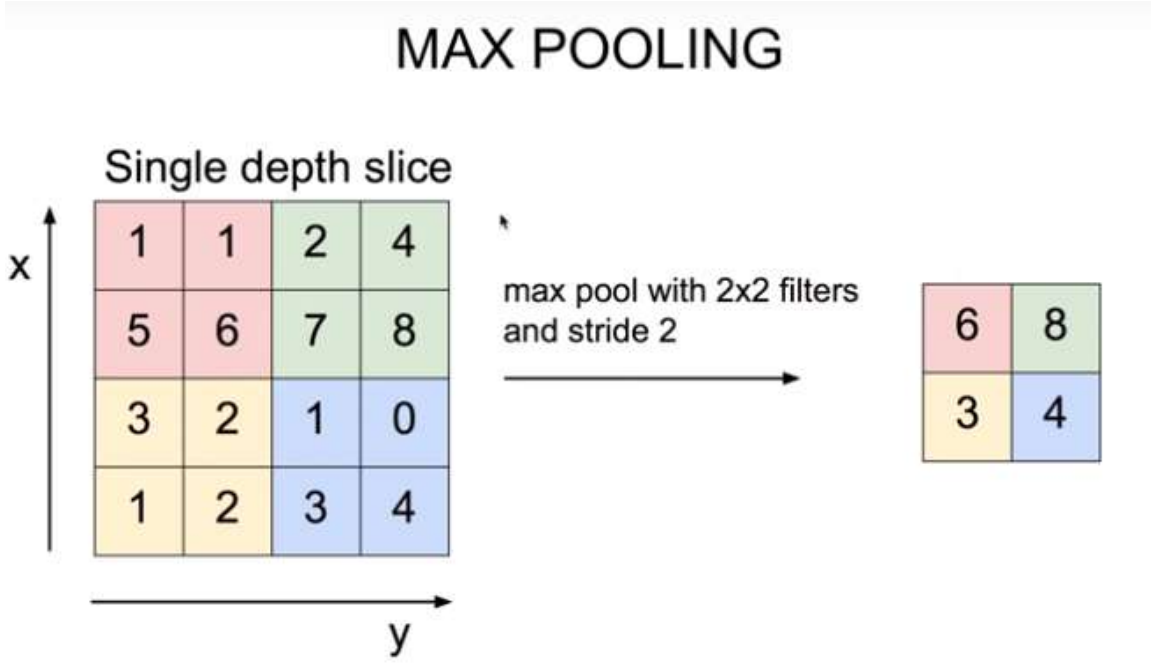
Using 3x3 filter applied on a single channel 4x4 image



※ 참고 : https://calvinfeng.gitbook.io/machine-learning-notebook/supervised-learning/convolutional-neural-network/convolution_operation

pooling

- Convolution을 거쳐서 나온 activation maps이 있을 때, 이를 이루는 convolution layer을 resizing하여 새로운 layer를 얻는 것



※ 참고 : hobinjeong.medium.com/cnn에서-pooling이란-c4e01aa83c83

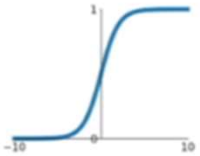
Activation Functions (활성화 함수)

- Neural Network에서 노드에 입력된 값들을 비선형 함수에 통과시킨 후 다음 레이어로 전달할 때 사용하는 함수

- 선형 함수가 아니라 비선형 함수를 사용하는 이유는 딥러닝 모델의 레이어 층을 깊게 가져갈 수 있기 때문

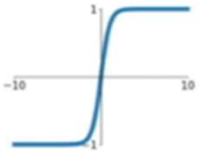
Sigmoid

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



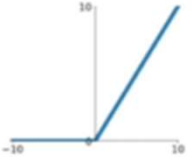
tanh

$$\tanh(x)$$



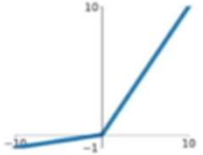
ReLU

$$\max(0, x)$$



Leaky ReLU

$$\max(0.1x, x)$$



Maxout

$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

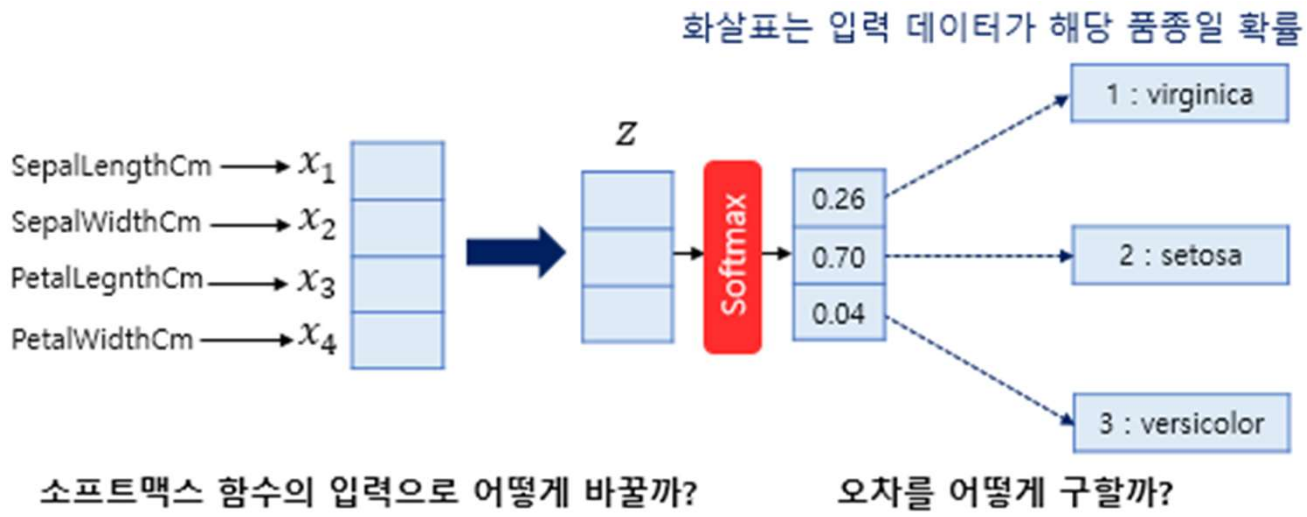
ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$

※ 참고 : https://blog.naver.com/PostView.naver?isHttpsRedirect=true&blogId=handuelly&logNo=221824080339

softmax

- 선택지의 총 개수를 k라고 할 때, k차원의 벡터를 입력받아 각 클래스에 대한 확률을 추정



※ 참고 : https://wikidocs.net/35476

What does reshape(60000, 28, 28, 1) mean?

What does reshape(60000, 28, 28, 1) mean?

Asked 3 years, 3 months ago Modified 3 years, 3 months ago Viewed 8k times

7

I'm learning convolutional networks and python all at once.

I have a problem with the following code:

```
import tensorflow as tf
print(tf.__version__)
mnist = tf.keras.datasets.fashion_mnist
(training_images, training_labels), (test_images, test_labels) = mnist.load_data()
training_images=training_images.reshape(60000, 28, 28, 1)
```

I don't understand what `reshape(60000, 28, 28, 1)` means.

What are 60000 and 28 and 28 and 1?

I will get 60000 arrays of 28 columns by 28 rows... and the 1 is...

python numpy conv-neural-network

12

Think about it, how would you store 60k images 28 by 28 pixels if it was RGB?

For each pixel you would need 3 scalars (each for one channel), so it would be 60000x28x28**x3**.

And how many channels you need when the image is in greyscale? Just one, so it would be 60000x28x28**x1**

Of course, in case of one channel this could be simplified even more to 60000x28x28, but I would say the former approach is better because you give explicitly information about how many channels the image has and looks like some ML frameworks require that information to operate correctly.

Share Follow edited Sep 24, 2019 at 13:34 answered Sep 24, 2019 at 13:33

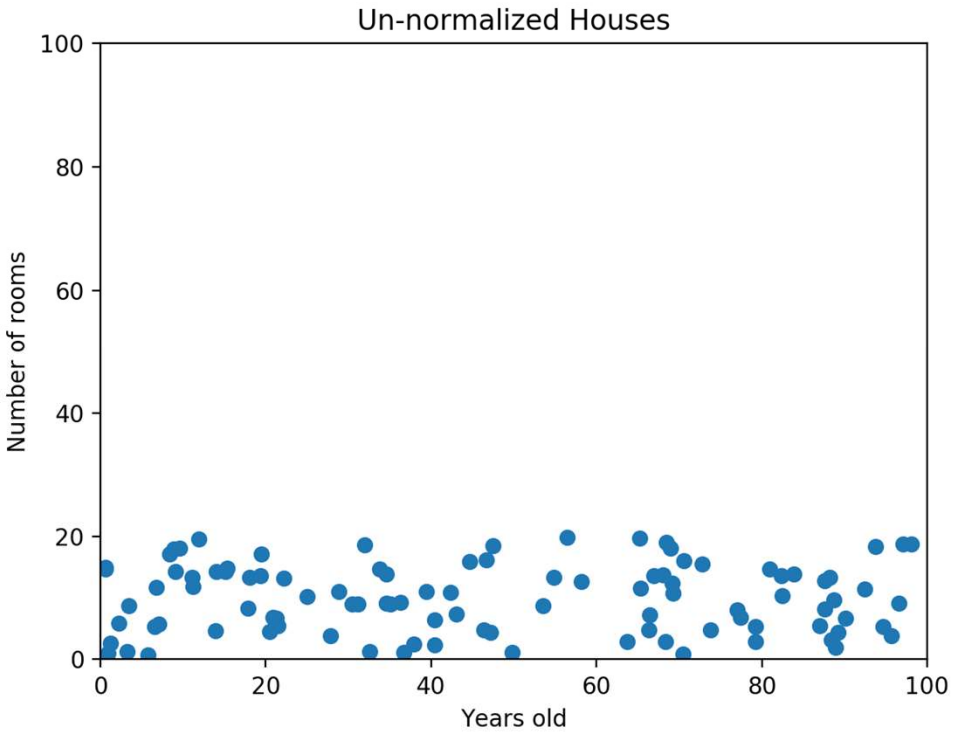
Yuras
484 ● 1 ● 3 ● 11

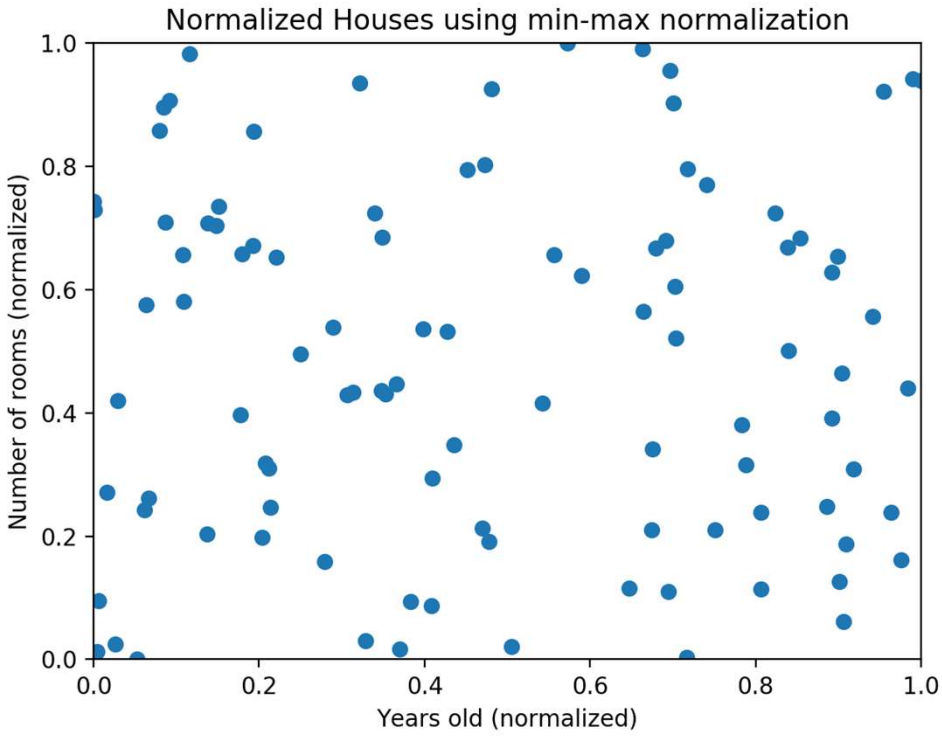
3 Also, if you want your code to work with both colored and grayscale images, you should make sure to not leave out this "empty dimension". – Nils Werner Sep 24, 2019 at 13:34

※ 참고 : https://stackoverflow.com/questions/58081284/what-does-reshape60000-28-28-1-mean

Normalization (정규화)

- 데이터가 동일한 정도의 스케일(중요도)로 반영되도록 해주는 게 정규화(Normalization)의 목표





※ 참고 : https://hleecaster.com/ml-normalization-concept/

ImageDataGenerator

- 이미지를 학습시킬 때 학습데이터의 양이 적을 경우 학습데이터를 조금씩 변형시켜서 학습데이터의 양을 늘리는 방식

keras.preprocessing.image.**ImageDataGenerator**(

featurewise_center=False, **samplewise_center**=False, **featurewise_std_normalization**=False,
samplewise_std_normalization=False, **zca_whitening**=False, **zca_epsilon**=1e-06, **rotation_range**=0,
width_shift_range=0.0, **height_shift_range**=0.0, **brightness_range**=None, **shear_range**=0.0, **zoom_range**=0.0,
channel_shift_range=0.0, **fill_mode**='nearest', **cval**=0.0, **horizontal_flip**=False, **vertical_flip**=False, **rescale**=None,
preprocessing_function=None, **data_format**=None, **validation_split**=0.0, **dtype**=None)

▶ rescale: 크기 재조절 인수. 디폴트 값은 None입니다. None 혹은 0인 경우 크기 재조절이 적용되지 않고,
그 외의 경우 (다른 변형을 전부 적용한 후에) 데이터를 주어진 값으로 곱합니다.

example) rescale = 1./255 : 값을 0과 1 사이로 변경

※ 참고 : https://acdongpgm.tistory.com/169

※ 참고 : https://keras.io/ko/preprocessing/image/

ImageDataGenerator().**flow_from_directory - 1/2**

- 이미지를 학습시킬 때 학습데이터의 양이 적을 경우 학습데이터를 조금씩 변형시켜서 학습데이터의 양을 늘리는 방식

flow_from_directory(
directory, **target_size**=(256, 256), **color_mode**='rgb', **classes**=None, **class_mode**='categorical',
batch_size=32, **shuffle**=True, **seed**=None, **save_to_dir**=None, **save_prefix**='', **save_format**='png',
follow_links=False, **subset**=None, **interpolation**='nearest')

▶ **target_size**: 정수 튜플 (높이, 넓이), 디폴트 값: (256, 256). 모든 이미지의 크기를 재조정할 치수.

▶ **class_mode**: 반환될 라벨 배열의 종류 결정
- "categorical" : (default) 2D 형태의 원-핫 인코딩 라벨
- "binary" : 1D 형태의 이진 라벨
- "sparse" : 1D 형태의 정수 라벨
- "input" : 입력 이미지와 동일한 이미지 (주로 자동 인코더와 함께 사용).
- None : 어떤 라벨도 반환되지 않음

※ 참고 : https://keras.io/ko/preprocessing/image/

ImageDataGenerator().**flow_from_directory - 2/2**

▶ classes: 클래스 하위 디렉토리의 선택적 리스트 (예. ['dogs', 'cats']). 디폴트 값: None.

- None: 하위 디렉토리의 이름/구조에서 자동으로 유추 (순서는 영숫자 순서 기준)

※ class_indices 속성을 통해서 클래스 이름과 클래스 색인 간 매핑을 담은 딕셔너리를 얻을 수 있음

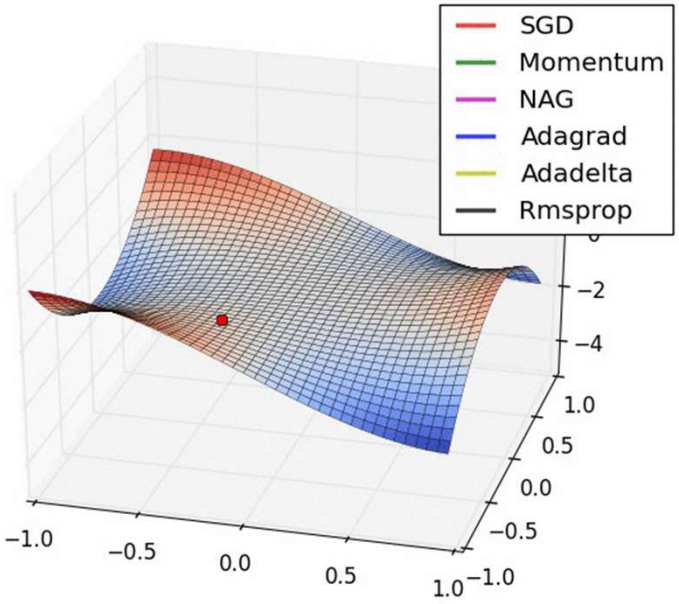
[return value]

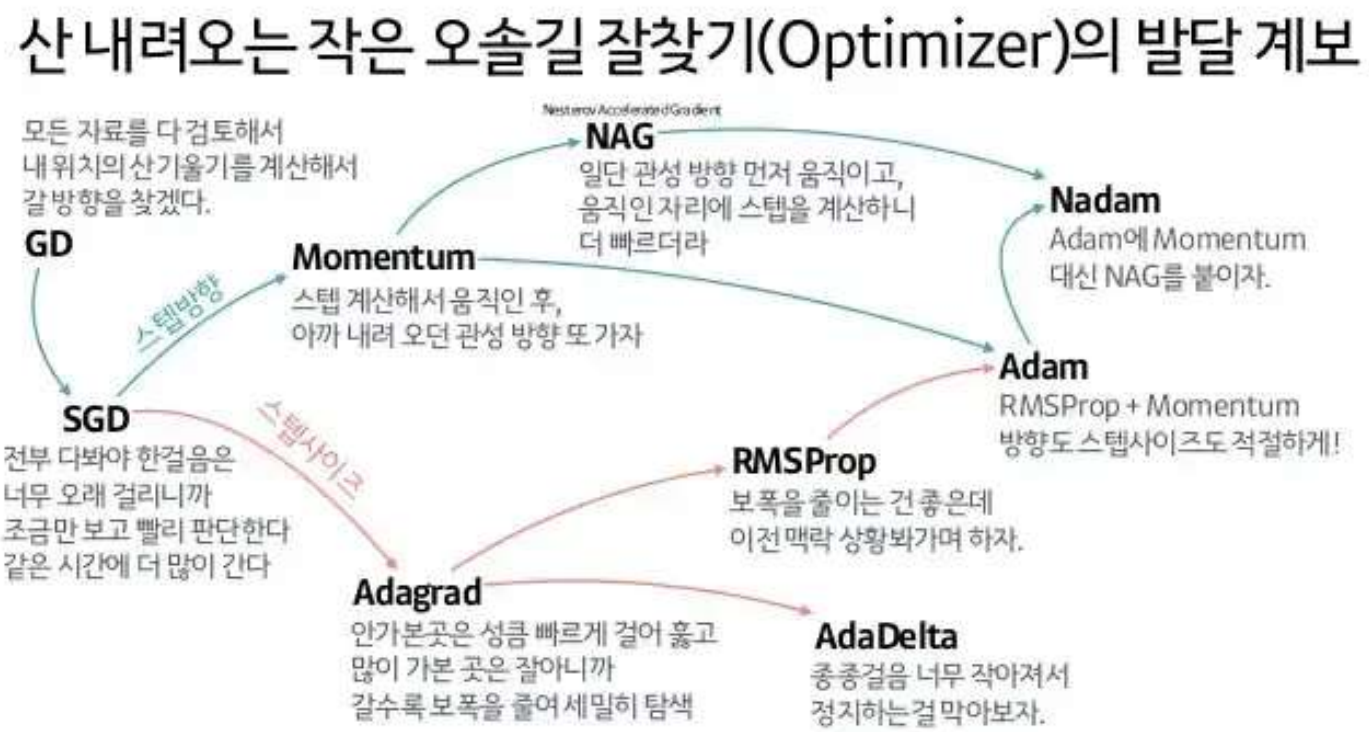
- (x, y) 튜플을 만들어내는 DirectoryIterator
  . x : (배치 크기, *표적 크기, 채널) 형태의 이미지 배치로 구성된 numpy 배열
  . y : 라벨로 이루어진 numpy 배열

※ 참고 : https://keras.io/ko/preprocessing/image/

Optimizer

- 손실 함수를 줄여 나가면서 학습하는 방법





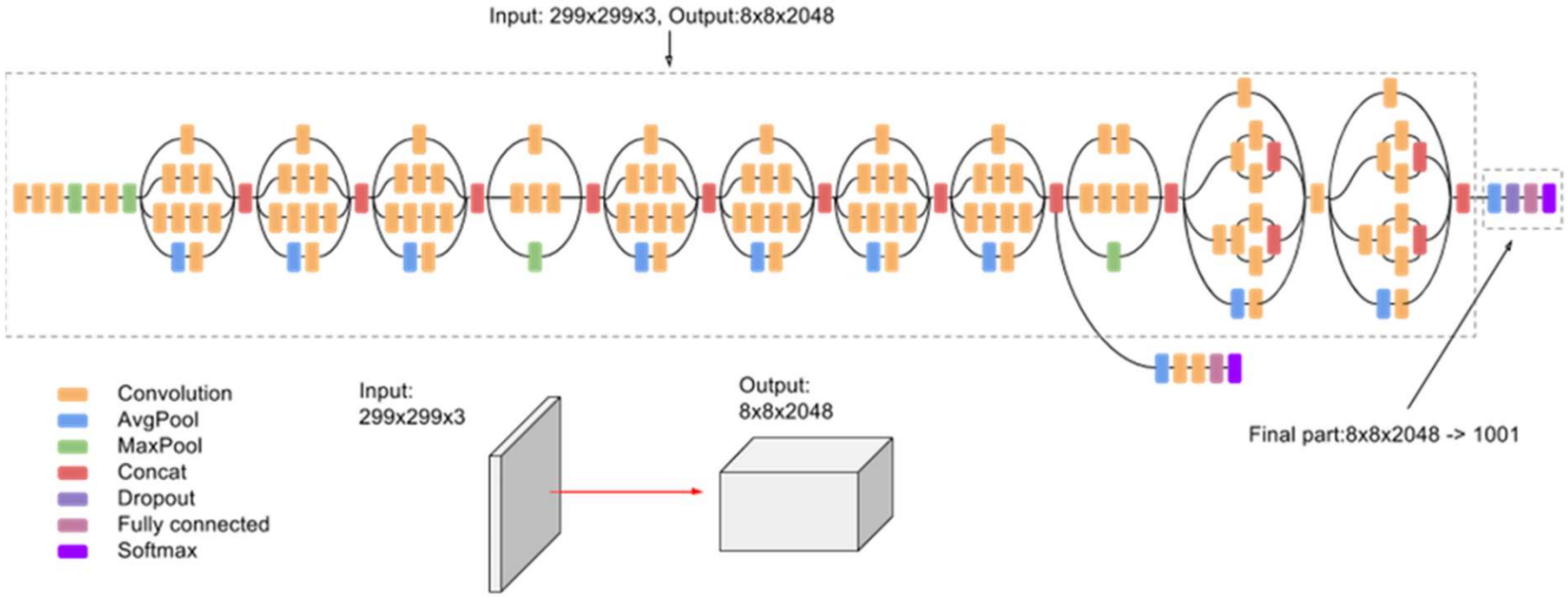
※ 참고 : https://heeya-stupidbutstudying.tistory.com/entry/ML-신경망에서의-Optimizer-역할과-종류

Inception.v3

- 대표적인 image classification용 CNN 모델 중 하나

  . GoogleNet이라고 불리우기도 한 이 모델은, 구글에서 만듦

  . 299×299 크기의 이미지를 입력 받아 풍선, 딸기, 고양이 등 사전에 정의된 여러 가지 사물 유형 중 하나로 분류



※ 참고 : https://cloud.google.com/tpu/docs/inception-v3-advanced?hl=ko

※ 참고 : https://norman3.github.io/papers/docs/google_inception.html