

# Convolutional Neural Network With TensorFlow

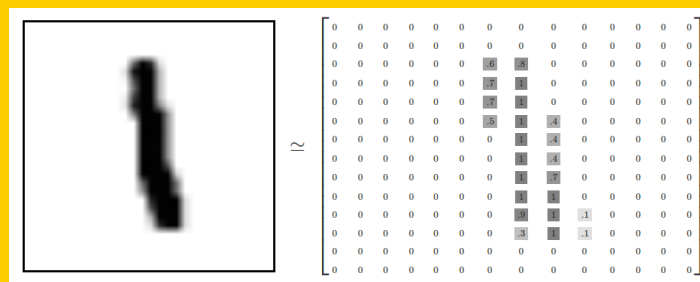
**텐서플로우로 합성곱 신경망 구현하기**

SEUNGWOO LEE

지금까지 **밑바닥**에서부터 코드 짜 봤으니까  
실제로 한번 적용해봅시다

# 텐서플로우로 간단하게 손글씨 숫자<sup>1)</sup>를

## 판별해보는 코드를 한번 짜봅시다



- 1) MNIST 데이터셋 입니다, 28\*28의 이미지로 이루어져 있습니다.  
(다만 이미지가 있는건 아니고 이미지를 Array로 변환한 데이터가 포함되어 있습니다)

**FloydHub**를 사용해서 실습을 해 보겠습니다

**딥러닝**을 **클라우드**에서 할 수 있습니다

가격	CPU	GPU	
	0.0432\$	0.432\$	Per hour

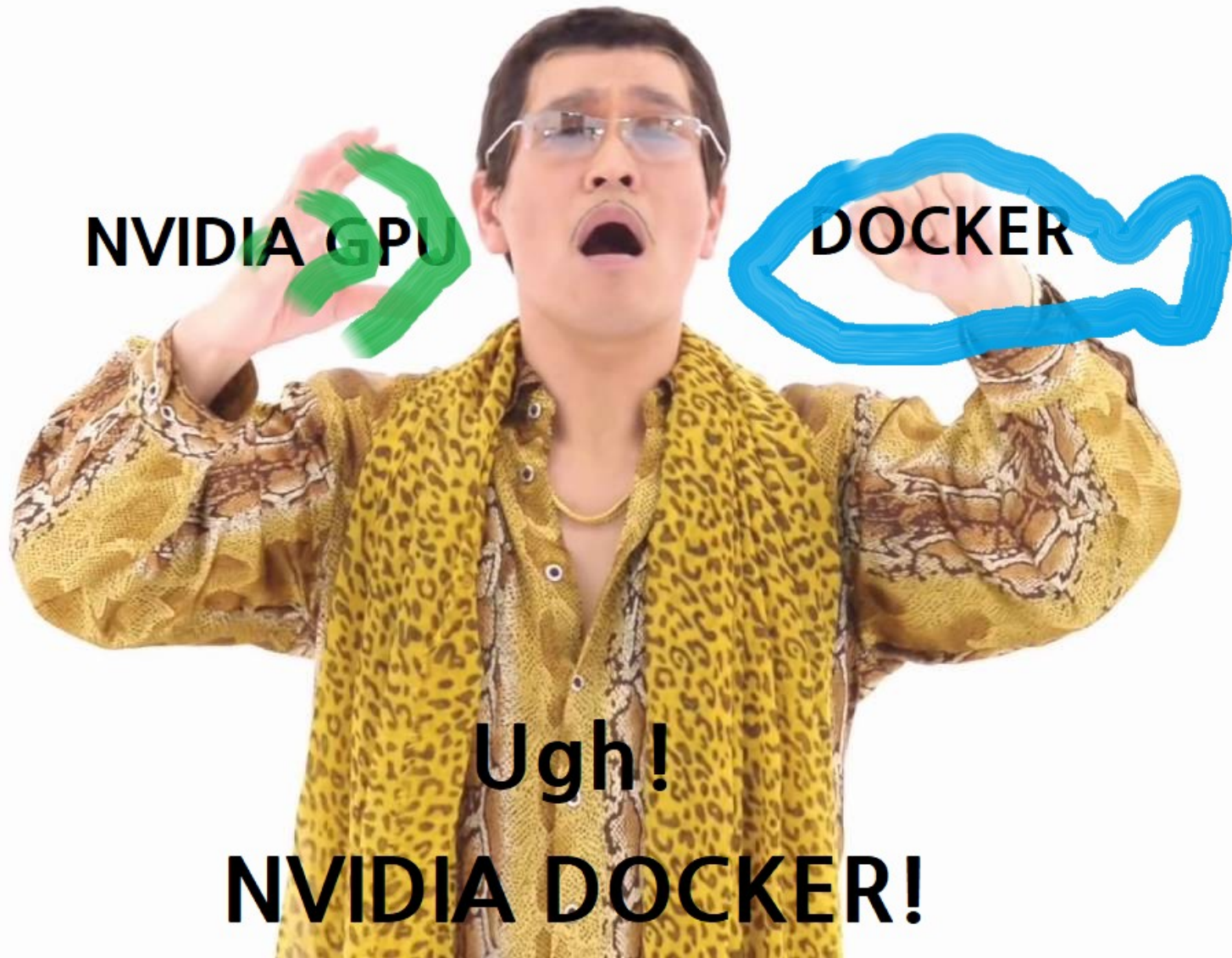
# 가격

## 기본 사양

Size	Desc	vCPUs	Mem	GPU Count	GPU MEM	DISK	COST Per Sec	COST Per Hour
C1	CPU 전용	1	8G	-	-	100G	0.0 <sup>4</sup> 12	0.0432
G1	CPU+GPU 전용	4	61G	1 <sup>1)</sup>	12G <sup>1)</sup>	100G	0.0 <sup>3</sup> 12	0.432

1) NVIDIA Tesla K80 입니다

# 구동은 이 친구가 도와줍니다



NVIDIA GPU

DOCKER

Ugh!

NVIDIA DOCKER!

서버에서

도커로 가상화 되어서

구동됩니다!

클라우드에서 학습을 하고, 학습된 데이터를  
다운 받거나 API로 사용할 수 있습니다

가입시 **100시간 무료**로 줍니다!



# 플랜

- Trial(채험판)
  - 무료 GPU 체험입니다 (100시간)
  - 1개의 잡을 동시에 돌릴 수 있으며, 24시간 Max. 입니다.
- Individual(개인용)
  - 연회비는 **없**으며, 내가 사용한 대로 냅니다.
  - 3개의 잡을 동시에 돌릴 수 있으며, 7일이 Max. 입니다.
- Team(팀용)
  - Waiting List에 등록 하셔야 합니다.

# 기본 사용법

From Python!

먼저 Sign Up을 합니다!  
(**카드번호** 필요 없어요!)

**그리고**

이메일 인증을 하고 로그인을 하면

다음과 같은 창이 뜹니다

# Welcome

**A**

## Install floyd-cli 설치

```
# Install floyd-cli
$ pip install -U floyd-cli
```

Trouble installing the cli? See the [installation guide](#)



Login to your account cmd 에서 floyd login을 입력하면

```
$ floyd login
```

*# Follow the instructions on your CLI*



Copy and paste the CLI authentication token in your terminal

\_\_\_\_\_

COPY TO CLIPBOARD

**D**

## Train your first model on FloydHub

Follow the [Quick Start Guide](#) to learn the basics of using FloydHub. Train a CNN model to recognize handwritten digits using Tensorflow on GPU and the MNIST database.

**<http://floydhub.com/welcome>**

C:\Windows\system32\cmd.exe

Microsoft Windows [Version 10.0.15063]  
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\John>floyd login

C:\Windows\system32\cmd.exe

Microsoft Windows [Version 10.0.15063]  
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\John>floyd login

Error:

Your version of CLI (0.9.10) is no longer compatible with server. Run:

    pip install -U floyd-cli  
to upgrade to the latest version (0.10.1)

C:\Users\John>\_





**설치한지 하루만에 업데이트라구요?**

C:\Windows\system32\cmd.exe

Microsoft Windows [Version 10.0.15063]  
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\John>floyd login

Error:

Your version of CLI (0.9.10) is no longer compatible with server. Run:

    pip install -U floyd-cli  
to upgrade to the latest version (0.10.1)

C:\Users\John>pip install -U floyd-cli

**까짓거 업데이트 하죠 뭐...**

C:\Windows\system32\cmd.exe

>=0.5.1->floyd-cli)

Requirement already up-to-date: certifi>=2017.4.17 in c:\miniconda3\lib\site-packages (from requests>=2.12.4->floyd-cli)

Requirement already up-to-date: idna<2.6,>=2.5 in c:\miniconda3\lib\site-packages (from requests>=2.12.4->floyd-cli)

Requirement already up-to-date: urllib3<1.22,>=1.21.1 in c:\miniconda3\lib\site-packages (from requests>=2.12.4->floyd-cli)

Requirement already up-to-date: chardet<3.1.0,>=3.0.2 in c:\miniconda3\lib\site-packages (from requests>=2.12.4->floyd-cli)

Requirement already up-to-date: six in c:\miniconda3\lib\site-packages (from pathlib2>=2.2.1->floyd-cli)

Building wheels for collected packages: floyd-cli

Running setup.py bdist\_wheel for floyd-cli ... done

Stored in directory: C:\Users\John\AppData\Local\pip\Cache\wheels\d2\41\cc\4dbb290b534a0f517630851540895ab183baea27029c09c8b5

Successfully built floyd-cli

Installing collected packages: floyd-cli

Found existing installation: floyd-cli 0.9.10

Uninstalling floyd-cli-0.9.10:

Successfully uninstalled floyd-cli-0.9.10

Successfully installed floyd-cli-0.10.1

C:\Users\John>floyd login\_

C:\Windows\system32\cmd.exe - floyd login

Requirement already up-to-date: certifi>=2017.4.17 in c:\miniconda3\lib\site-packages (from requests>=2.12.4->floyd-cli)

Requirement already up-to-date: idna<2.6,>=2.5 in c:\miniconda3\lib\site-packages (from requests>=2.12.4->floyd-cli)

Requirement already up-to-date: urllib3<1.22,>=1.21.1 in c:\miniconda3\lib\site-packages (from requests>=2.12.4->floyd-cli)

Requirement already up-to-date: chardet<3.1.0,>=3.0.2 in c:\miniconda3\lib\site-packages (from requests>=2.12.4->floyd-cli)

Requirement already up-to-date: six in c:\miniconda3\lib\site-packages (from pathlib2>=2.2.1->floyd-cli)

Building wheels for collected packages: floyd-cli

Running setup.py bdist\_wheel for floyd-cli ... done

Stored in directory: C:\Users\John\AppData\Local\pip\Cache\wheels\d2\41\cc\4dbb290b534a0f517630851540895ab183baea27029c09c8b5

Successfully built floyd-cli

Installing collected packages: floyd-cli

Found existing installation: floyd-cli 0.9.10

Uninstalling floyd-cli-0.9.10:

Successfully uninstalled floyd-cli-0.9.10

Successfully installed floyd-cli-0.10.1

C:\Users\John>floyd login

Authentication token page will now open in your browser. Continue? [Y/n]:

## CLI authentication token

Copy and paste the token in your terminal

COPY TO CLIPBOARD

COPY를 눌러서 복사한 후에 웹에  
우클릭 해서 붙여넣기 (안보이지만  
붙여넣기 된거니까 걱정 마세요)

```
C:\Windows\system32\cmd.exe

Running setup.py bdist_wheel for floyd-cli ... done
Stored in directory: C:\Users\John\AppData\Local\pip\Cache\wheels\d2\41\cc\4dbb290b534a0f517630851540895ab183baea27029c09c8b5
Successfully built floyd-cli
Installing collected packages: floyd-cli
  Found existing installation: floyd-cli 0.9.10
    Uninstalling floyd-cli-0.9.10:
      Successfully uninstalled floyd-cli-0.9.10
Successfully installed floyd-cli-0.10.1

C:\Users\John>floyd login
Authentication token page will now open in your browser. Continue? [Y/n]: Y
Please copy and paste the authentication token.
This is an invisible field. Paste token and press ENTER:
Error: Invalid Token

C:\Users\John>floyd login
Authentication token page will now open in your browser. Continue? [Y/n]: y
Please copy and paste the authentication token.
This is an invisible field. Paste token and press ENTER:
Login Successful

C:\Users\John>_
```

**로그인 완료**

# Project & Dataset 생성하는 법





Projects

Datasets



New Project



[jaeseung172/projects/cnn-project](#)

empty cnn project

★ 0    Last updated unknown

✳ None

☰ 0

🌿 None

🕒 -

← Prev

Next →

[Projects](#)[Datasets](#)

## Create a new project

A project contains all your jobs and data

**Project path**

**Project name**

**Description (optional)**

**Visibility Level**



**Public**

Anyone can see this project



**Private**

Only you can see this project

Create project





## Command line instructions

### Initialize project locally ⓘ

Command to link your jobs to this FloydHub project

```
floyd init cnn-project
```

### Run a job ⓘ

Run a job in this project

```
floyd run <insert-command-here>
# To use GPU
floyd run --gpu <insert-command-here>
# To use a different environemnt
floyd run --gpu --pytorch <insert-command-here>
```

[Projects](#)[Datasets](#)

## Datasets

[New Dataset](#)

### No datasets found

Explore other's datasets or create your own

[← Prev](#)[Next →](#)

[Projects](#)[Datasets](#)

## Create a new dataset

A dataset is a collection of files and the story that makes them compelling

**Dataset path**

**Dataset name**

**Description (optional)**

**Visibility Level**



**Public**

Anyone can see this dataset



**Private**

Only you can see this dataset

Create dataset






jaeseung172 / datasets / datasets-sample 

★ Star

0

↗ Share ▾

 Overview

 Versions

 CLI

 Settings

## Command line instructions

### Initialize dataset locally

Command to link your data to this FloydHub dataset

```
floyd data init datasets-sample
```

### Upload data

Run data to this dataset

```
floyd data upload
```

# 실행은 이렇게 합니다

CPU: 공백

GPU: --gpu

```
floyd run --env {Environment} {CPU or GPU?} {FILE EXT}
              실행 환경                                "python FILE_NAME"
```

# Environment

TensorFlow: tensorflow-1.2, tensorflow-1.2:py2,  
tensorflow, tensorflow:py2, tensorflow-1.0,  
tensorflow-1.0:py2, tensorflow-0.12, tensorflow-  
0.12:py2

Theano: theano-0.8, theano-0.8:py2, theano-0.9,  
theano-0.9:py2

**(Keras는 이 두버전을 임포트 했을때 사용 가능)**

Caffe(rc4): caffe, caffe:py2

Torch(Torch7): torch, torch:py2

PyTorch(0.1.9): pytorch, pytorch:py2



다음의 환경에는 자동으로 이 패키지들이  
설치되어 있습니다

**h5py, iPython, Jupyter, matplotlib, numpy,  
OpenCV, Pandas, Pillow, scikit-learn, scipy,  
sklearn**

# 데이터셋에 관한 명령어

floyd data init	데이터셋을 Initialize 합니다
floyd data upload	데이터셋을 업로드 합니다
floyd data status	데이터셋을 리스팅 합니다
<b>floyd data clone</b>	<b>데이터셋을 클론 합니다 그러나 이 커맨드에 대한 설명은 없음</b>
floyd data delete	데이터를 삭제 합니다
floyd data output	아웃풋 파일을 브라우저에서 봅니다 (기본으로 셋팅한 브라우저에서 열림)
<b>커맨드</b>	<b>정보</b>

# floyd data init

앞서 웹 대쉬보드에서 지정했던 이름으로

지금 있는 디렉토리를 설정합니다

floyd data init 데이터셋\_이름

# floyd data upload

앞서 지정했던 폴더에 있는

데이터를 업로드 합니다

floyd data upload

# floyd data status

앞서 업로드했던 데이터의 상태를 봅니다  
(뒤에 NAME이나 ID를 붙이면 자세한 정보가  
나오게 됩니다)

# floyd data delete

앞서 설정하고, 업로드하고, 체크해봤던

데이터셋을 삭제합니다

floyd data delete {NAMES or IDS}

# floyd data output

데이터 **Output**을 볼 수 있습니다

floyd data output ID

(다만, 옆에 -u를 붙이게 되면

**URL**만 던져 줍니다)

**floyd run 시에 파라미터 설정 방법**



파라미터		설명
--cpu/--gpu	디폴트는 cpu	디폴트로 CPU로 구동됩니다만, GPU로 설정할 때는 GPU로 구동됩니다
--data <ID:mount>		ID는 당신이 연결하고 싶은 데이터 소스를 의미하고, mount는 정확한 path를 말합니다
--mode [jupyter serve]	command	Jupyter로 실행하는지, REST API로 Serve하는지 설정합니다
--no-open		주피터 실행할 때 브라우저 안뜨게 합니다, URL만 출력
--env [tensorflow, ...]		환경설정입니다, 앞서 설명드렸습니다
--tensorboard		세롭게 추가된 텐서보드 기능입니다

**주피터 노트북** 지원합니다!

floyd run --mode jupyter

(단! 주피터는 계속 돌아가니까 사용하신 후에

**floyd stop {ID}** 시켜줘야 합니다, 지갑이...)

# floyd requirements

- Floyd에서 구동될 Dependencies 를 설정합니다.
- 문법은 PyPi requirements를 작성하는 것과 똑같습니다.
- 자동적으로 Run 커맨드를 돌릴 때에 실행됩니다.
- 파일 이름을 floyd\_requirements.txt 로 지정해 놓으면 됩니다.

# 더 읽어보기 (영문)

Environment

<http://docs.floydhub.com/guides/environments>

Data Mounting

[http://docs.floydhub.com/guides/mounting\\_data/](http://docs.floydhub.com/guides/mounting_data/)

외부 라이브러리 사용 관련

[http://docs.floydhub.com/guides/installing\\_dependencies/](http://docs.floydhub.com/guides/installing_dependencies/)

# Convolutional Neural Network

**한국어로**

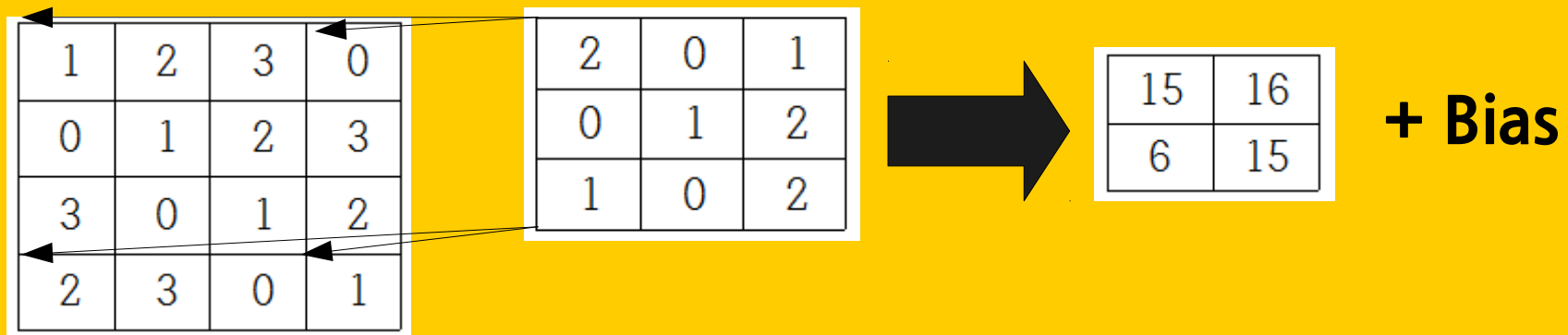
**합성곱 신경망!**

**합성곱 신경망의 과정은**

**Conv - Pool ... 로부터 시작합니다**

**핵심은 합성곱 연산에 있습니다**





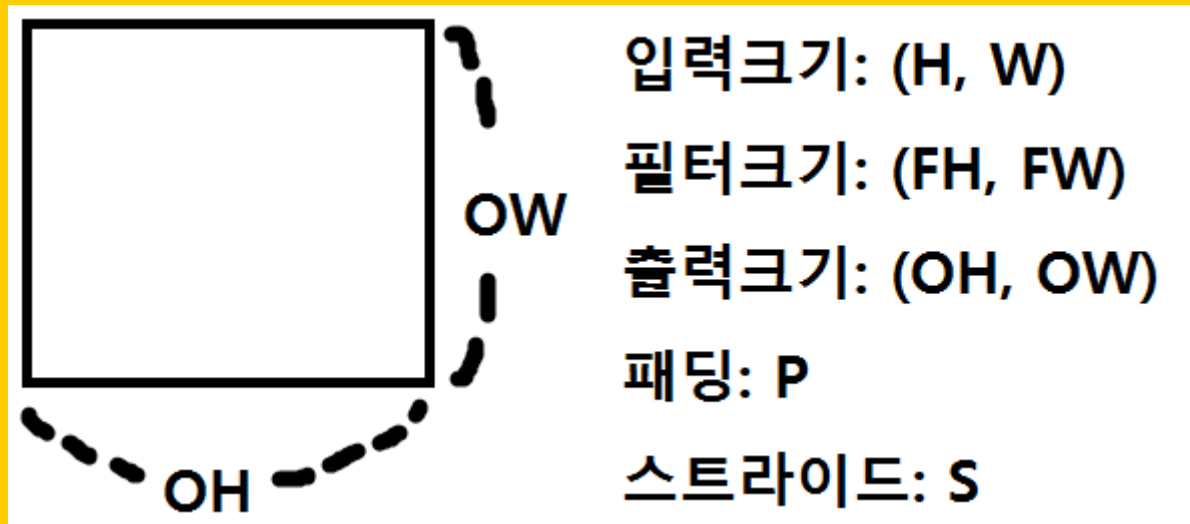
```
conv1 = conv2d(x, weights['wc1'], biases['bc1'])  
# 5x5 conv, 1 input, 32 outputs  
'wc1': tf.Variable(tf.random_normal([5, 5, 1, 32])),  
'bc1': tf.Variable(tf.random_normal([32])),
```

# 합성곱이란 무엇일까요?

```
def conv2d(x, W, b, strides=1):  
    # 컨브 2차원 래퍼, 렐루 함수를 사용하고, 웨이트 바이어스 값을 인풋으로 한다, 스트라이드는 1로 설정한다.  
    # 스트라이드는 발자국을 의미한다.  
    x = tf.nn.conv2d(x, W, strides=[1, strides, strides, 1], padding='SAME')  
    x = tf.nn.bias_add(x, b)  
    # Relu 함수 때려서 계산, 0이랑 같거나 0보다 큰것만 리턴하고 0보다 작은 음수는 0을 리턴함  
    return tf.nn.relu(x)
```

패딩: 합성곱 연산을 할때 데이터 주변을 특정 값(보통 0)으로 채우기도 한다, 출력 크기를 조정할 목적으로 사용한다. (데이터 손실을 방지하기 위하여 사용한다)  
(그러나 여기서는 'SAME' 이기 때문에 28\*28 그대로 들어가게 된다.

스트라이드: 필터를 적용하는 위치의 간격을 이야기한다, 스트라이드를 키우면 출력 크기는 작아진다.



$$OH = \frac{H + 2P - FH}{S} + 1 \quad OW = \frac{W + 2P - FW}{S} + 1$$

# 합성곱 연산 Formula

**출력 크기가 정수로 맞아 떨어져야 한다**

**(프레임워크 중에는 반올림하는 경우가 있다)**

**3차원 데이터는 합성곱 연산을 어떻게 할까요?**

**3차원은 각 필터끼리 서로 곱하고 더한 후에**

**3개를 모두 더해줍니다**

**각 필터당 하나의 출력  
레이어가 생성된다**

**Conv를 했으면 풀링을 해봅시다**

**Pooling은 간단합니다**



# 앞에서 했던 Conv.에서 가장 x한 값을 찾아 내는걸 말합니다

Mean-pooling도 있고, Max-pooling도 있습니다.  
그래서 x라고 한 겁니다

**이제 거의 다 왔습니다!**

**TensorFlow로 구현해봅시다!**