AI School 6기 5주차

파이썬 기초 – 웹 크롤링

CNN 기초2

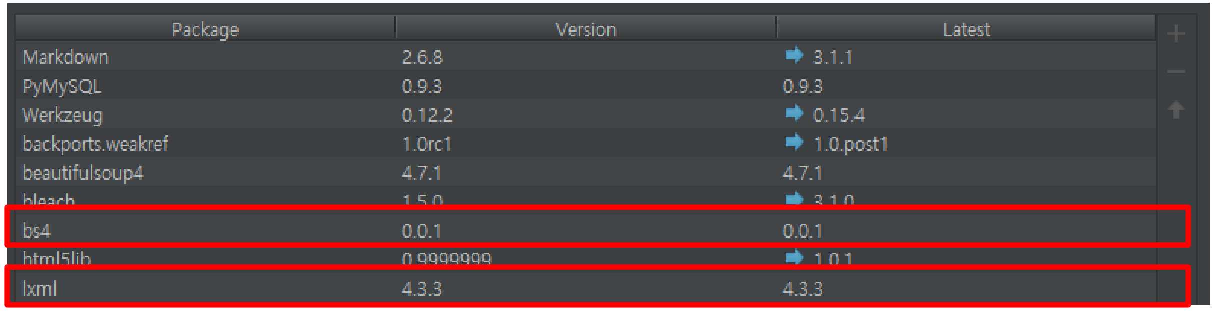
CNN 모델을 활용한 객체 분류

AI School 6기 5주차

파이썬 기초 – 웹 크롤링

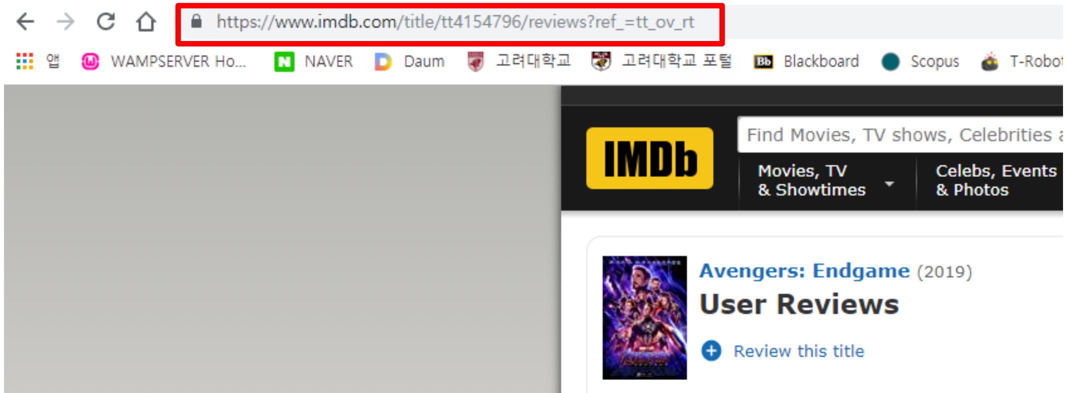
웹 크롤링

- BeuatifulSoup



Package	Version	Latest
Markdown	2.6.8	➡ 3.1.1
PyMySQL	0.9.3	0.9.3
Werkzeug	0.12.2	➡ 0.15.4
backports.weakref	1.0rc1	➡ 1.0.post1
beautifulsoup4	4.7.1	4.7.1
bleach	1.5.0	➡ 3.1.0
bs4	0.0.1	0.0.1
html5lib	0.9999999	➡ 1.0.1
lxml	4.3.3	4.3.3

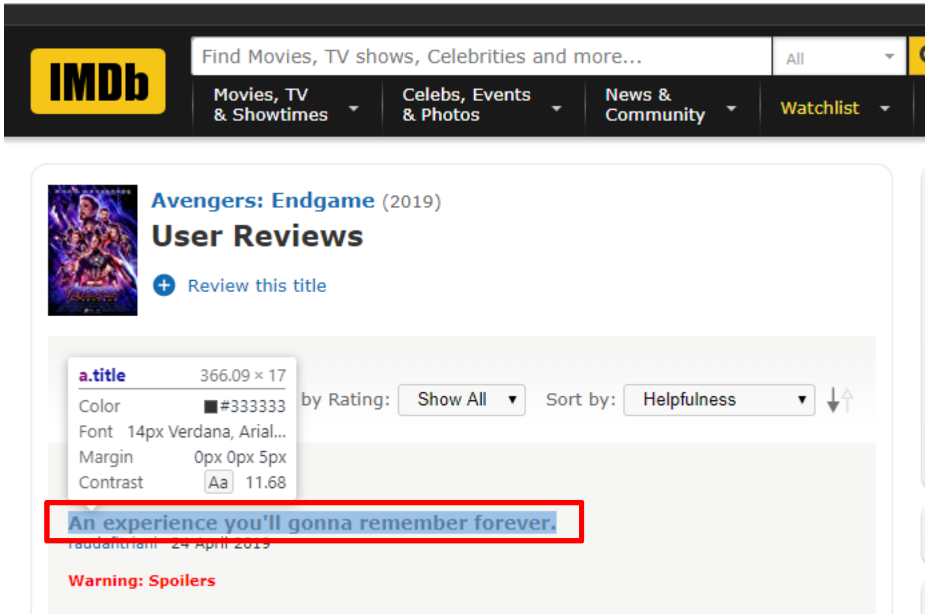
```
from bs4 import BeautifulSoup
import urllib.request

url = "https://www.imdb.com/title/tt4154796/reviews?ref_=tt_ov_rt"
htmlData = urllib.request.urlopen(url)
bs = BeautifulSoup(htmlData, 'lxml')
print(BeautifulSoup.prettify(bs))
```

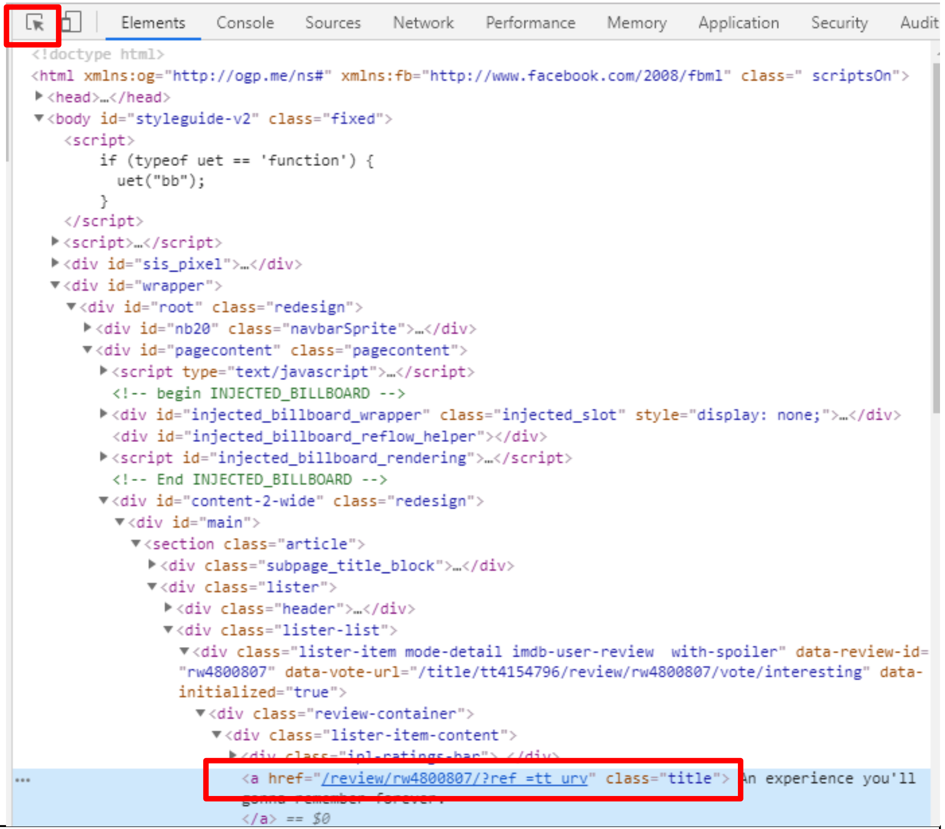


https://www.imdb.com/title/tt4154796/reviews?ref_=tt_ov_rt

웹 크롤링

- 개발자 도구 (F12)



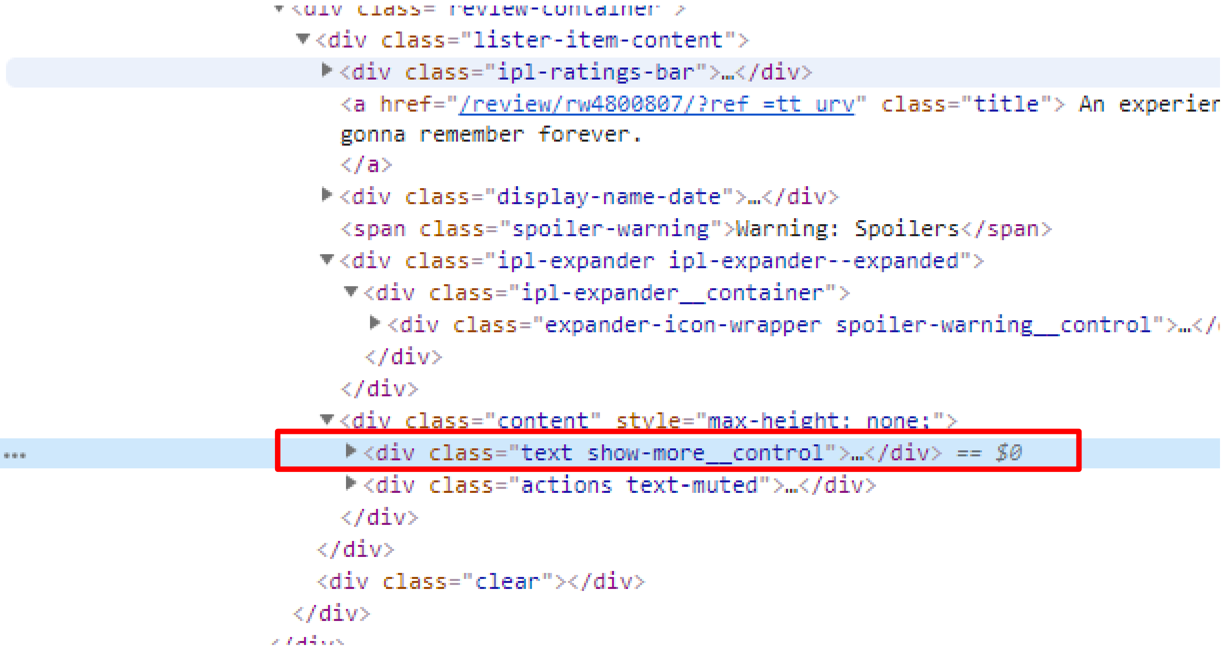


```
<!doctype html>
<html xmlns:og="http://ogp.me/ns#" xmlns:fb="http://www.facebook.com/2008/fbml" class=" scriptsOn">
 ▶<head>…</head>
 ▼<body id="styleguide-v2" class="fixed">
    <script>
        if (typeof uet == 'function') {
          uet("bb");
        }
    </script>
   ▶<script>…</script>
   ▶<div id="sis_pixel">…</div>
   ▼<div id="wrapper">
     ▼<div id="root" class="redesign">
       ▶<div id="nb20" class="navbarSprite">…</div>
       ▼<div id="pagecontent" class="pagecontent">
         ▶<script type="text/javascript">…</script>
          <!-- begin INJECTED_BILLBOARD -->
         ▶<div id="injected_billboard_wrapper" class="injected_slot" style="display: none;">…</div>
          <div id="injected_billboard_reflow_helper"></div>
         ▶<script id="injected_billboard_rendering">…</script>
          <!-- End INJECTED_BILLBOARD -->
         ▼<div id="content-2-wide" class="redesign">
           ▼<div id="main">
             ▼<section class="article">
               ▶<div class="subpage_title_block">…</div>
               ▼<div class="lister">
                 ▶<div class="header">…</div>
                 ▼<div class="lister-list">
                   ▼<div class="lister-item mode-detail imdb-user-review  with-spoiler" data-review-id=
                    "rw4800807" data-vote-url="/title/tt4154796/review/rw4800807/vote/interesting" data-
                    initialized="true">
                     ▼<div class="review-container">
                       ▼<div class="lister-item-content">
                         ▶<div class="ipl-ratings-bar">…</div>
...                       <a href="/review/rw4800807/?ref_=tt_urv" class="title"> An experience you'll
                          gonna remember forever.
                          </a> == $0
```

```
title_list = bs.findAll('a', 'title')

for title in title_list:
    print(title.getText())
```

웹 크롤링

- bs.findAll([tag], [class명])

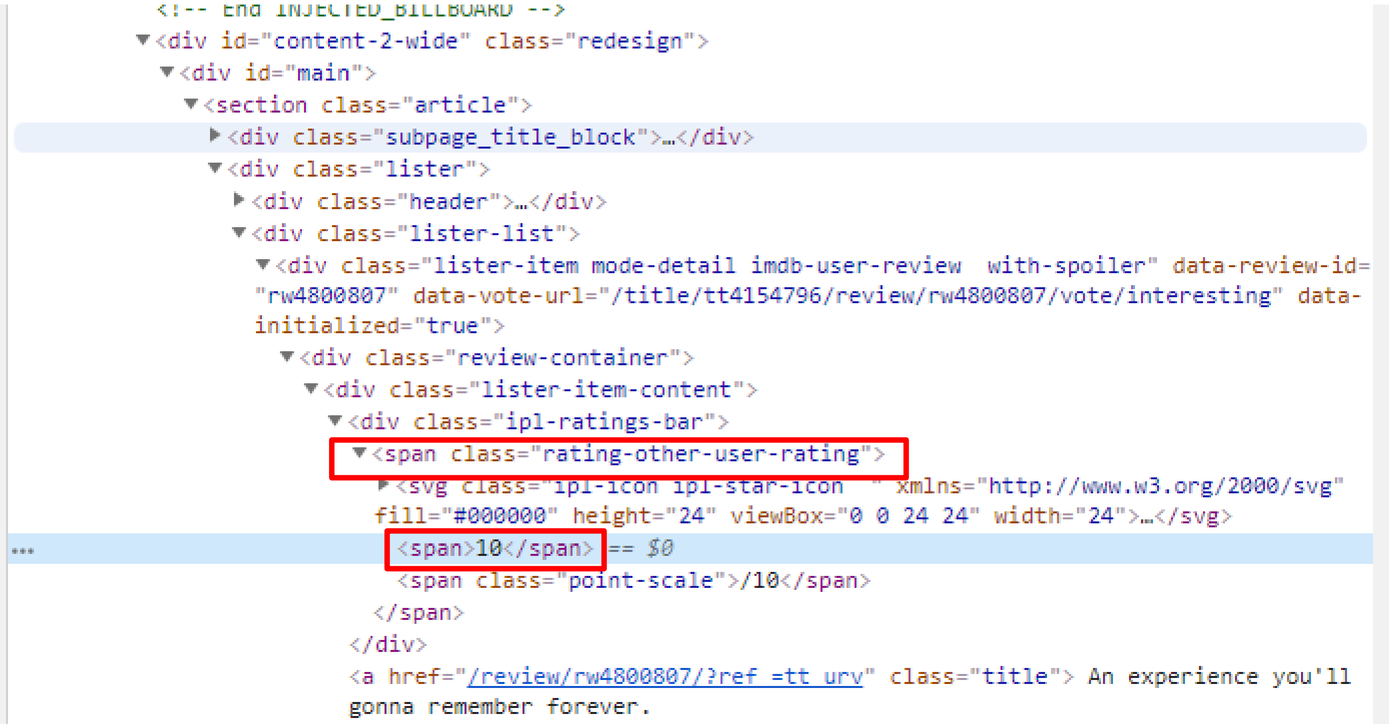


```
▼<div class="review-container">
  ▼<div class="lister-item-content">
    ▶<div class="ipl-ratings-bar">…</div>
     <a href="/review/rw4800807/?ref_=tt_urv" class="title"> An experier
     gonna remember forever.
     </a>
    ▶<div class="display-name-date">…</div>
     <span class="spoiler-warning">Warning: Spoilers</span>
    ▼<div class="ipl-expander ipl-expander--expanded">
      ▼<div class="ipl-expander__container">
        ▶<div class="expander-icon-wrapper spoiler-warning__control">…</
       </div>
     </div>
    ▼<div class="content" style="max-height: none;">
...   ▶<div class="text show-more__control">…</div> == $0
      ▶<div class="actions text-muted">…</div>
     </div>
   </div>
   <div class="clear"></div>
  </div>
</div>
```

```
review_list = bs.findAll('div', 'text show-more__control')

for content in review_list:
    print(content.getText()+"₩n")
```

웹 크롤링

- bs.findAll([tag], [class명])



```
<!-- End INJECTED_BILLBOARD -->
▼<div id="content-2-wide" class="redesign">
  ▼<div id="main">
    ▼<section class="article">
      ▶<div class="subpage_title_block">…</div>
      ▼<div class="lister">
        ▶<div class="header">…</div>
        ▼<div class="lister-list">
          ▼<div class="lister-item mode-detail imdb-user-review  with-spoiler" data-review-id=
          "rw4800807" data-vote-url="/title/tt4154796/review/rw4800807/vote/interesting" data-
          initialized="true">
            ▼<div class="review-container">
              ▼<div class="lister-item-content">
                ▼<div class="ipl-ratings-bar">
                  ▼<span class="rating-other-user-rating">
                    ▶<svg class="ipl-icon ipl-star-icon  " xmlns="http://www.w3.org/2000/svg"
                    fill="#000000" height="24" viewBox="0 0 24 24" width="24">…</svg>
...                  <span>10</span> == $0
                     <span class="point-scale">/10</span>
                   </span>
                 </div>
                 <a href="/review/rw4800807/?ref_=tt_urv" class="title"> An experience you'll
                 gonna remember forever.
```

```
score_list = bs.findAll('span', 'rating-other-user-rating')

for score in score_list:
    print(score.span.getText())
```

전처리

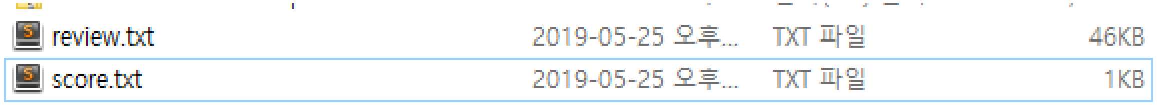
- 알파벳 외 문자 제거, 특수 문자 등 분리, 소문자 변환

```
import re

def clean_str(string):
    string = re.sub(r"[^A-Za-z0-9(),!?₩'₩`]", " ", string)
    string = re.sub(r"₩'s", " ₩'s", string)
    string = re.sub(r"₩'ve", " ₩'ve", string)
    string = re.sub(r"n₩'t", " n₩'t", string)
    string = re.sub(r"₩'re", " ₩'re", string)
    string = re.sub(r"₩'d", " ₩'d", string)
    string = re.sub(r"₩'ll", " ₩'ll", string)
    string = re.sub(r",", " , ", string)
    string = re.sub(r"!", " ! ", string)
    string = re.sub(r"₩(", " ₩( ", string)
    string = re.sub(r"₩)", " ₩) ", string)
    string = re.sub(r"₩?", " ₩? ", string)
    string = re.sub(r"₩s{2,}", " ", string)
    return string.strip().lower()
```

File 출력

- f = open(“파일 경로”, “읽기/쓰기”)

```
print(len(title_list))
f = open("./data/review.txt", "w", encoding='UTF8')
for i in range(len(title_list)):
    f.write(clean_str(title_list[i].getText())+" "+clean_str(review_list[i].getText())+"₩n")
f.close()

f = open("./data/score.txt", "w", encoding='UTF8')
for i in range(len(score_list)):
    f.write(score_list[i].span.getText()+"₩n")
f.close()
```

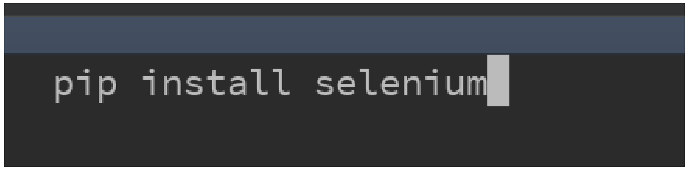


```
review.txt    2019-05-25 오후...    TXT 파일    46KB
score.txt     2019-05-25 오후...    TXT 파일     1KB
```

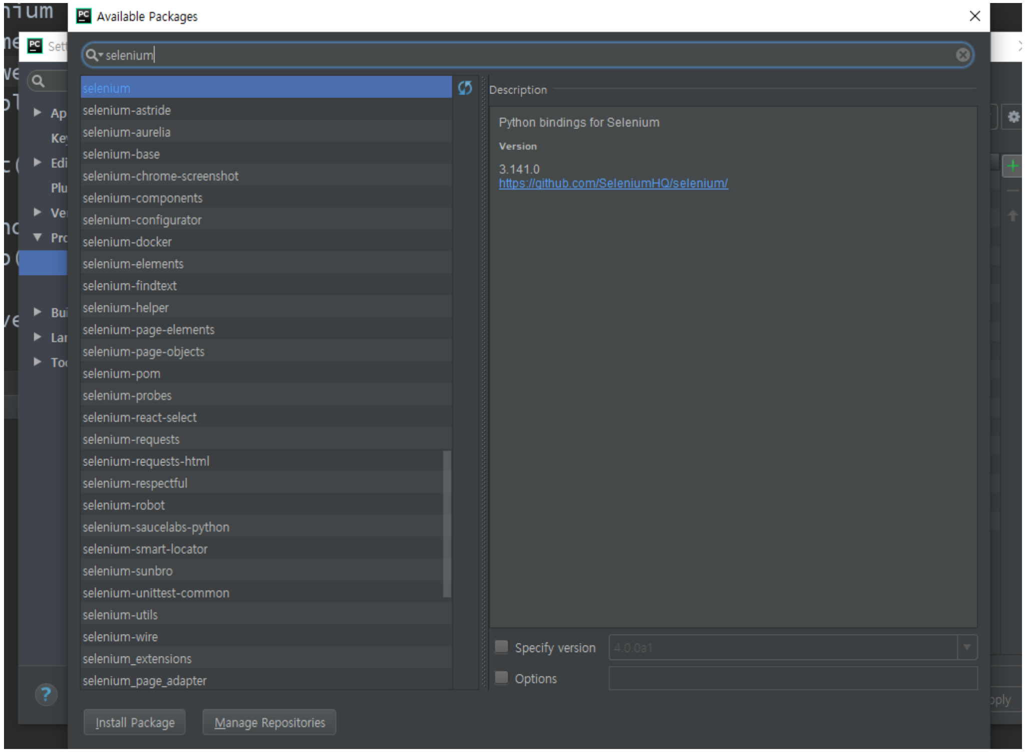
동적 크롤링

- Selenium

  - 웹앱을 테스트하는데 이용하는 프레임 워크.
  - Webdriver라는 API를 통해 운영체제에 설치된 Chrome등의 브라우저를 제어



```
pip install selenium
```



Search: selenium

selenium
selenium-astride
selenium-aurelia
selenium-base
selenium-chrome-screenshot
selenium-components
selenium-configurator
selenium-docker
selenium-elements
selenium-findtext
selenium-helper
selenium-page-elements
selenium-page-objects
selenium-pom
selenium-probes
selenium-react-select
selenium-requests
selenium-requests-html
selenium-respectful
selenium-robot
selenium-saucelabs-python
selenium-smart-locator
selenium-sunbro
selenium-unittest-common
selenium-utils
selenium-wire
selenium_extensions
selenium_page_adapter

Description

Python bindings for Selenium

Version

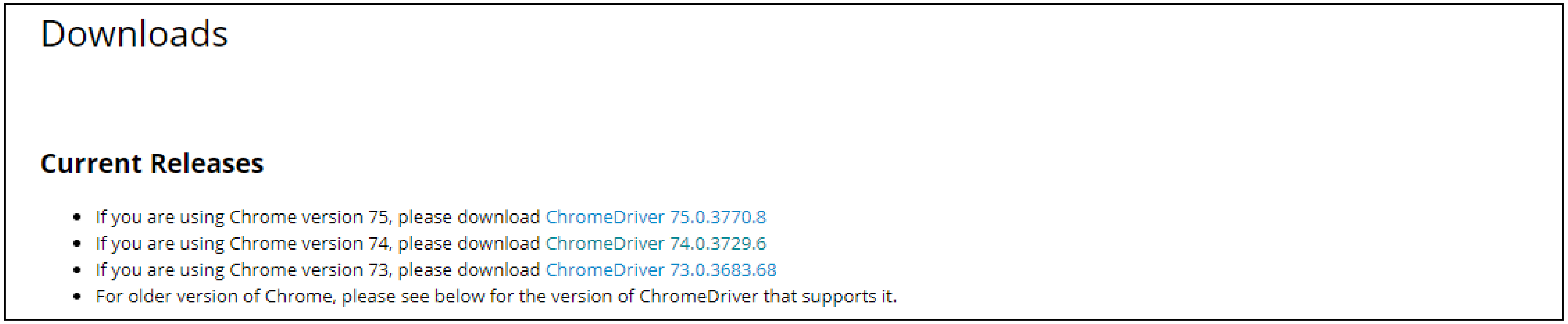
3.141.0
https://github.com/SeleniumHQ/selenium/

□ Specify version 4.0.0a1

□ Options

동적 크롤링

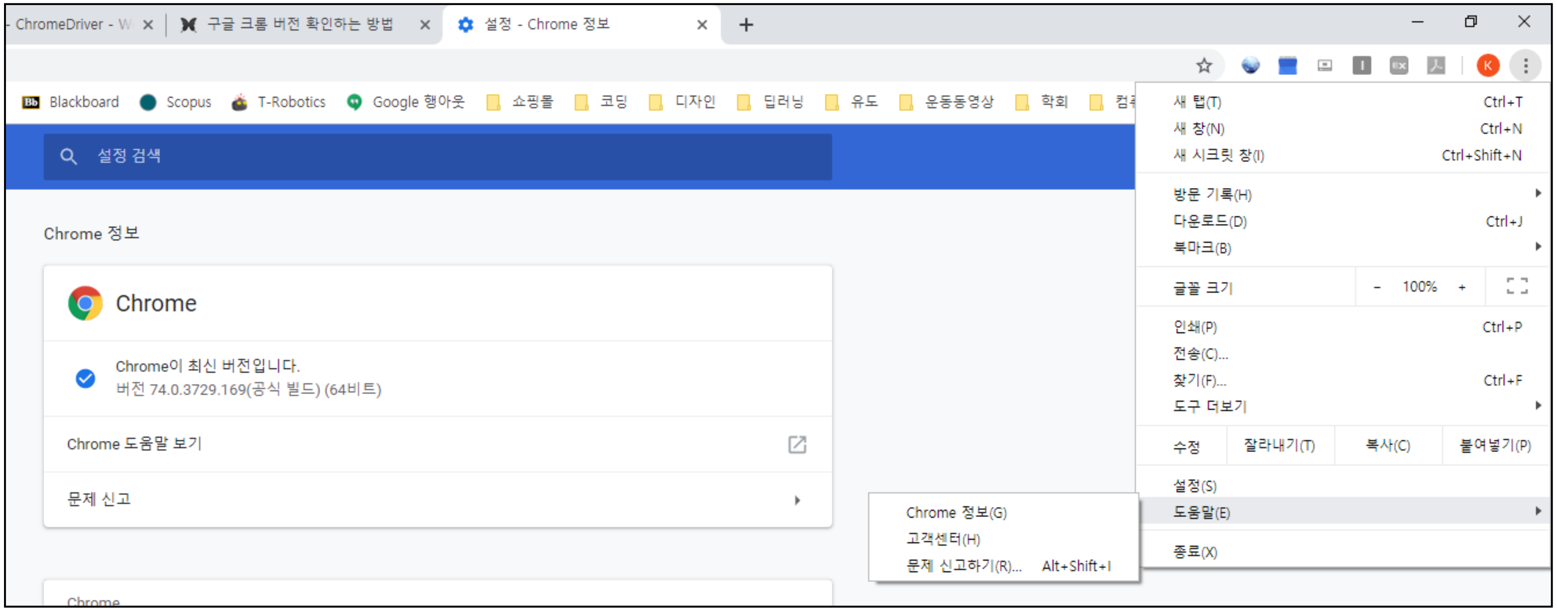
- Chrome WebDriver
  - https://sites.google.com/a/chromium.org/chromedriver/downloads
  - 본인의 Chrome 버전 확인 후 버전에 맞는 webdriver 다운로드



Downloads

Current Releases

- If you are using Chrome version 75, please download ChromeDriver 75.0.3770.8
- If you are using Chrome version 74, please download ChromeDriver 74.0.3729.6
- If you are using Chrome version 73, please download ChromeDriver 73.0.3683.68
- For older version of Chrome, please see below for the version of ChromeDriver that supports it.



Chrome 정보

Chrome

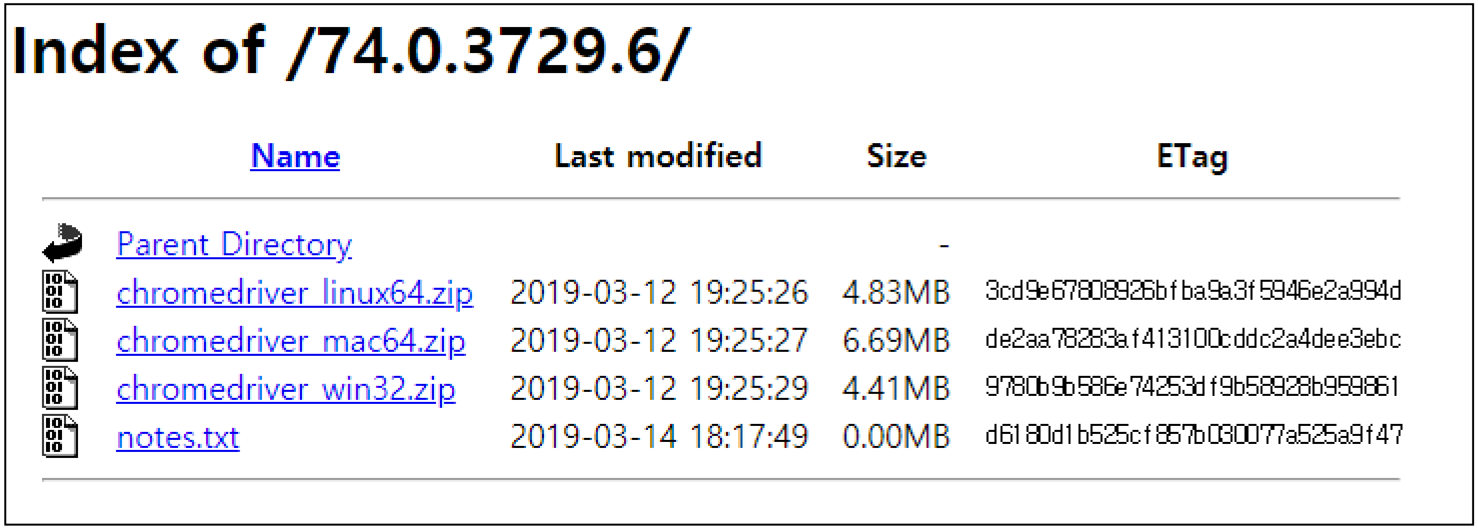
Chrome이 최신 버전입니다.
버전 74.0.3729.169(공식 빌드) (64비트)

Chrome 도움말 보기

문제 신고

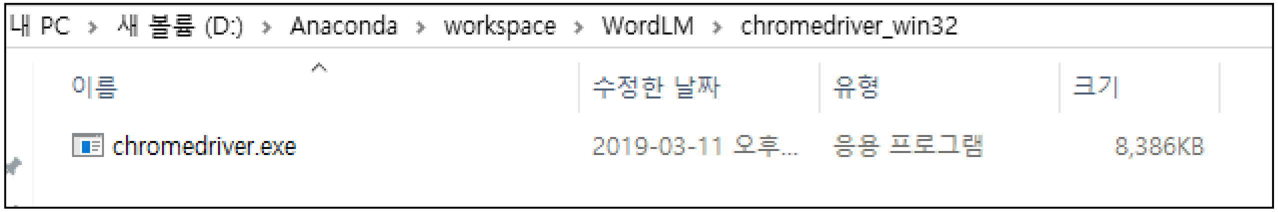
동적 크롤링

- Chrome WebDriver
  - 운영체제에 맞는 webdriver 다운로드
  - 원하는 위치에 압축 풀기



Index of /74.0.3729.6/

Name	Last modified	Size	ETag
Parent Directory		-	
chromedriver_linux64.zip	2019-03-12 19:25:26	4.83MB	3cd9e67808926bfba9a3f5946e2a994d
chromedriver_mac64.zip	2019-03-12 19:25:27	6.69MB	de2aa78283af413100cddc2a4dee3ebc
chromedriver_win32.zip	2019-03-12 19:25:29	4.41MB	9780b9b586e74253df9b58928b959861
notes.txt	2019-03-14 18:17:49	0.00MB	d6180d1b525cf857b030077a525a9f47



내 PC › 새 볼륨 (D:) › Anaconda › workspace › WordLM › chromedriver_win32

이름	수정한 날짜	유형	크기
chromedriver.exe	2019-03-11 오후...	응용 프로그램	8,386KB

동적 크롤링

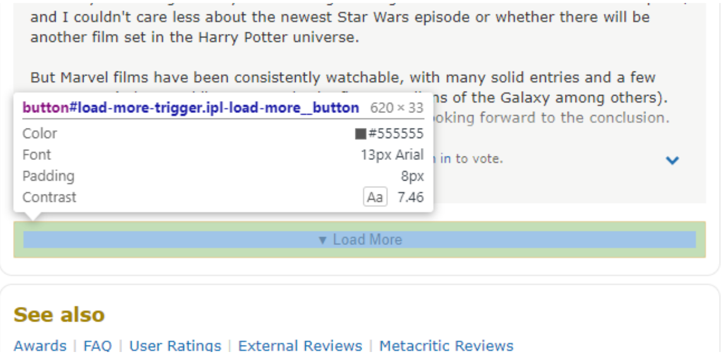
- Chrome WebDriver
  - https://sites.google.com/a/chromium.org/chromedriver/downloads
  - 본인의 Chrome 버전 확인 후 버전에 맞는 webdriver 다운로드

```
import selenium
from selenium import webdriver
import time
driver = webdriver.Chrome("D:/Anaconda/workspace/WordLM/chromedriver_win32/chromedriver")
driver.implicitly_wait(3)

driver.get('https://www.imdb.com/title/tt4154796/reviews?ref_=tt_ov_rt')
```

동적 크롤링

- 버튼 클릭 및 html source 받아오기

```
driver.find_element_by_xpath('//*[@id="load-more-trigger"]').click()
time.sleep(10)

click_list = driver.find_elements_by_xpath("//div[@class='expander-icon-wrapper show-more__control']")
for click in click_list:
    if click.is_displayed():
        click.click()

req = driver.page_source

bs=BeautifulSoup(req, 'lxml')
```



and I couldn't care less about the newest Star Wars episode or whether there will be another film set in the Harry Potter universe.

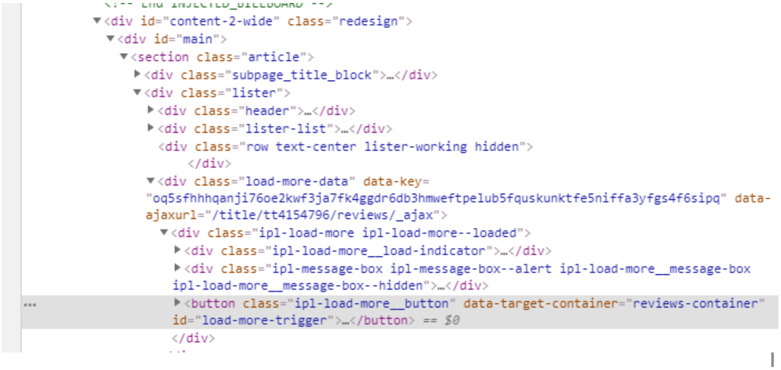
But Marvel films have been consistently watchable, with many solid entries and a few
...ns of the Galaxy among others).
...oking forward to the conclusion.

...in to vote.

button#load-more-trigger.ipl-load-more__button 620 × 33

Color	■#555555
Font	13px Arial
Padding	8px
Contrast	Aa 7.46

▼ Load More
See also
Awards | FAQ | User Ratings | External Reviews | Metacritic Reviews



```
<!-- End INJECTED_BILLBOARD -->
▼<div id="content-2-wide" class="redesign">
  ▼<div id="main">
    ▼<section class="article">
      ▶<div class="subpage_title_block">…</div>
      ▼<div class="lister">
        ▶<div class="header">…</div>
        ▶<div class="lister-list">…</div>
         <div class="row text-center lister-working hidden">
            </div>
        ▼<div class="load-more-data" data-key=
        "oq5sfhhhqanji76oe2kwf3ja7fk4ggdr6db3hmweftpelub5fquskunktfe5niffa3yfgs4f6sipq" data-
        ajaxurl="/title/tt4154796/reviews/_ajax">
          ▼<div class="ipl-load-more ipl-load-more--loaded">
            ▶<div class="ipl-load-more__load-indicator">…</div>
            ▶<div class="ipl-message-box ipl-message-box--alert ipl-load-more__message-box
            ipl-load-more__message-box--hidden">…</div>
...         ▶<button class="ipl-load-more__button" data-target-container="reviews-container"
            id="load-more-trigger">…</button> == $0
            </div>
```

동적 크롤링

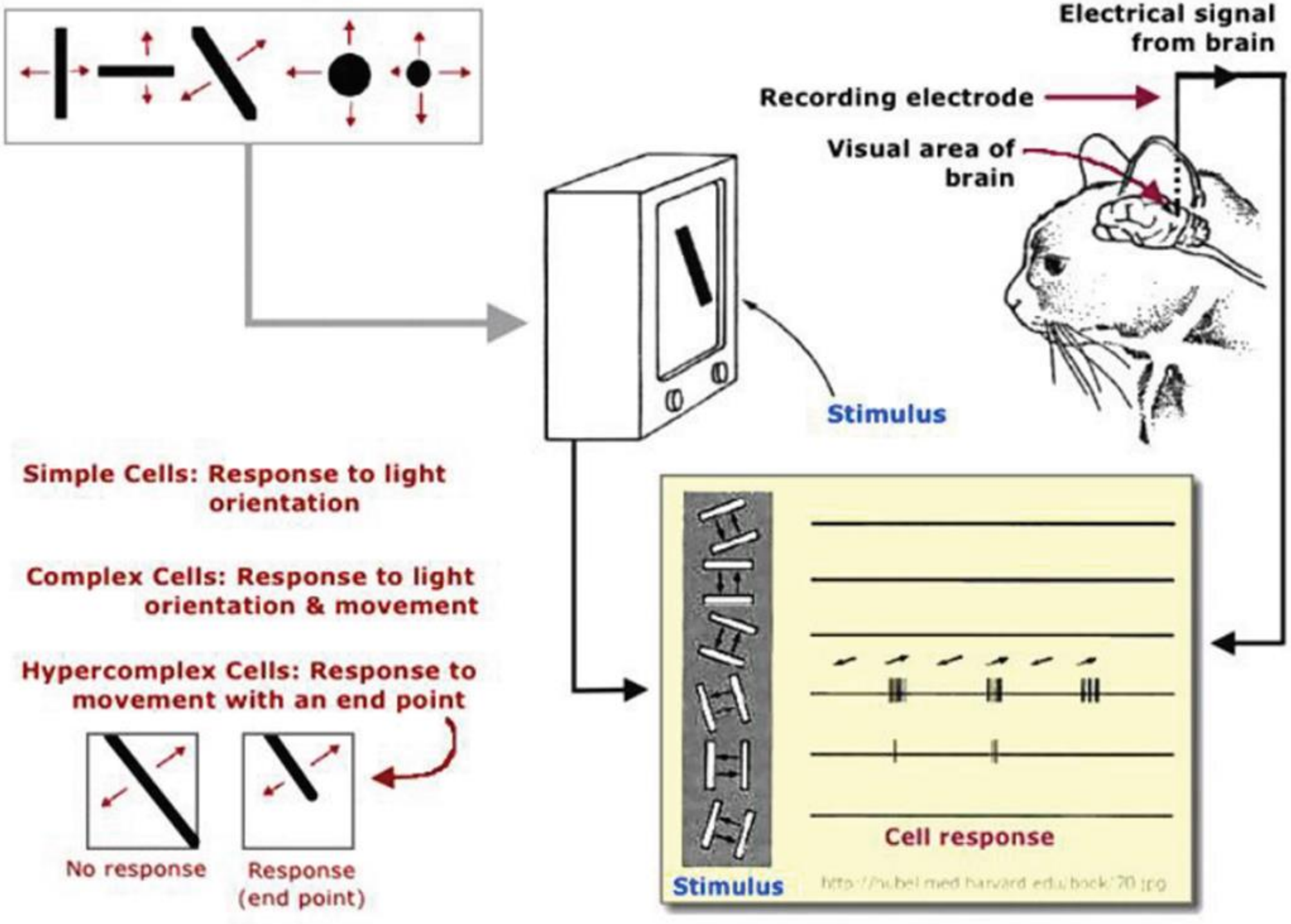
- 뒤 부분 동일

```
title_list = bs.findAll('a', 'title')
review_list = bs.findAll('div', 'text show-more__control')
score_list = bs.findAll('span', 'rating-other-user-rating')
.
.
.
f = open("./data/score.txt", "w", encoding='UTF8')
for i in range(len(score_list)):
    f.write(score_list[i].span.getText()+"₩n")
f.close()
```

AI School 5기 5주차

CNN 기초2

Convolutional Neural Networks

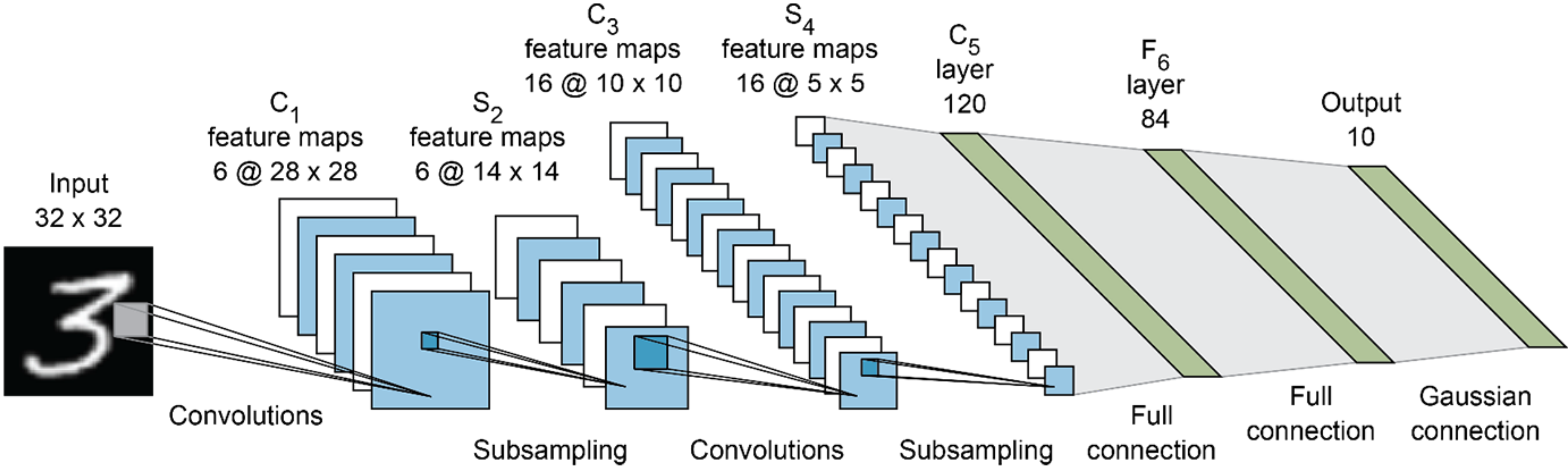


http://hubel.med.harvard.edu/book/70.jpg

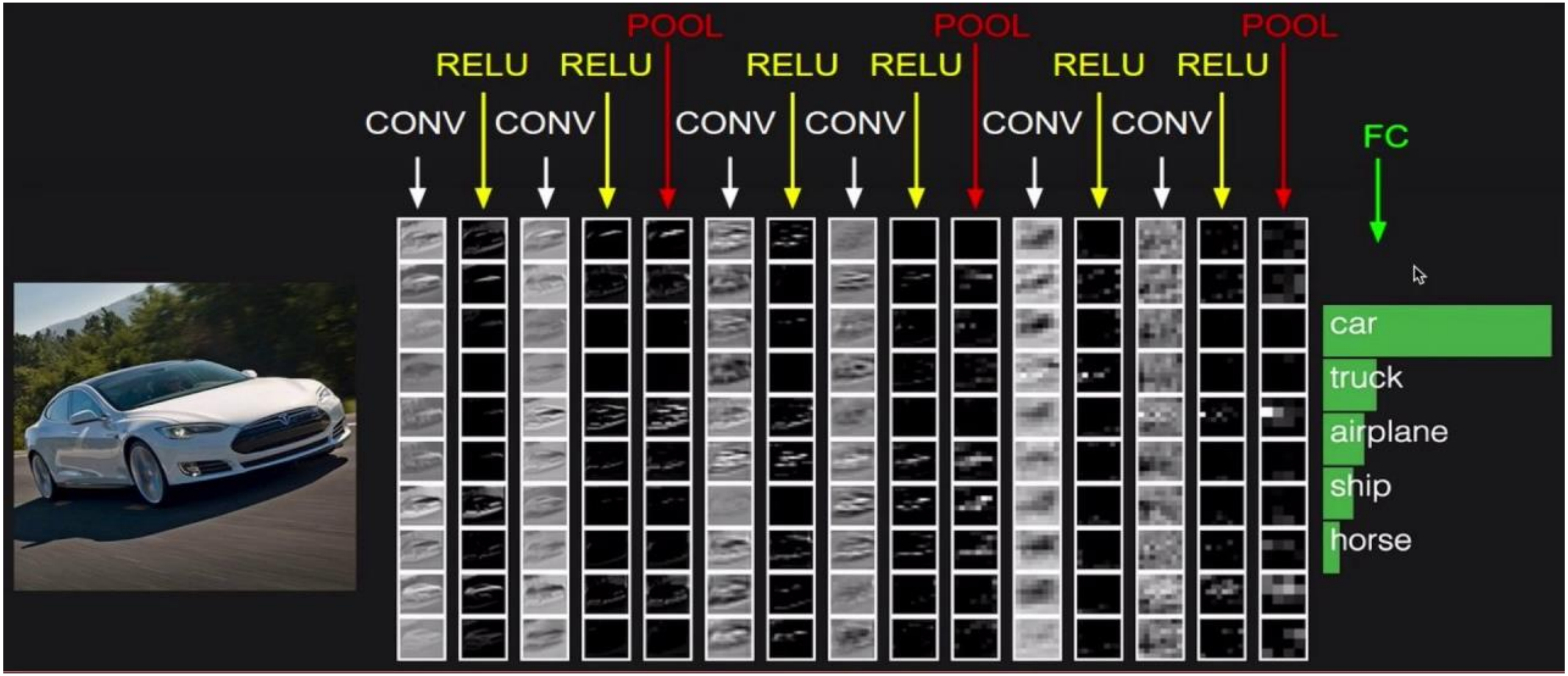
Hubel & Wiesel, 1959

Convolutional Neural Networks

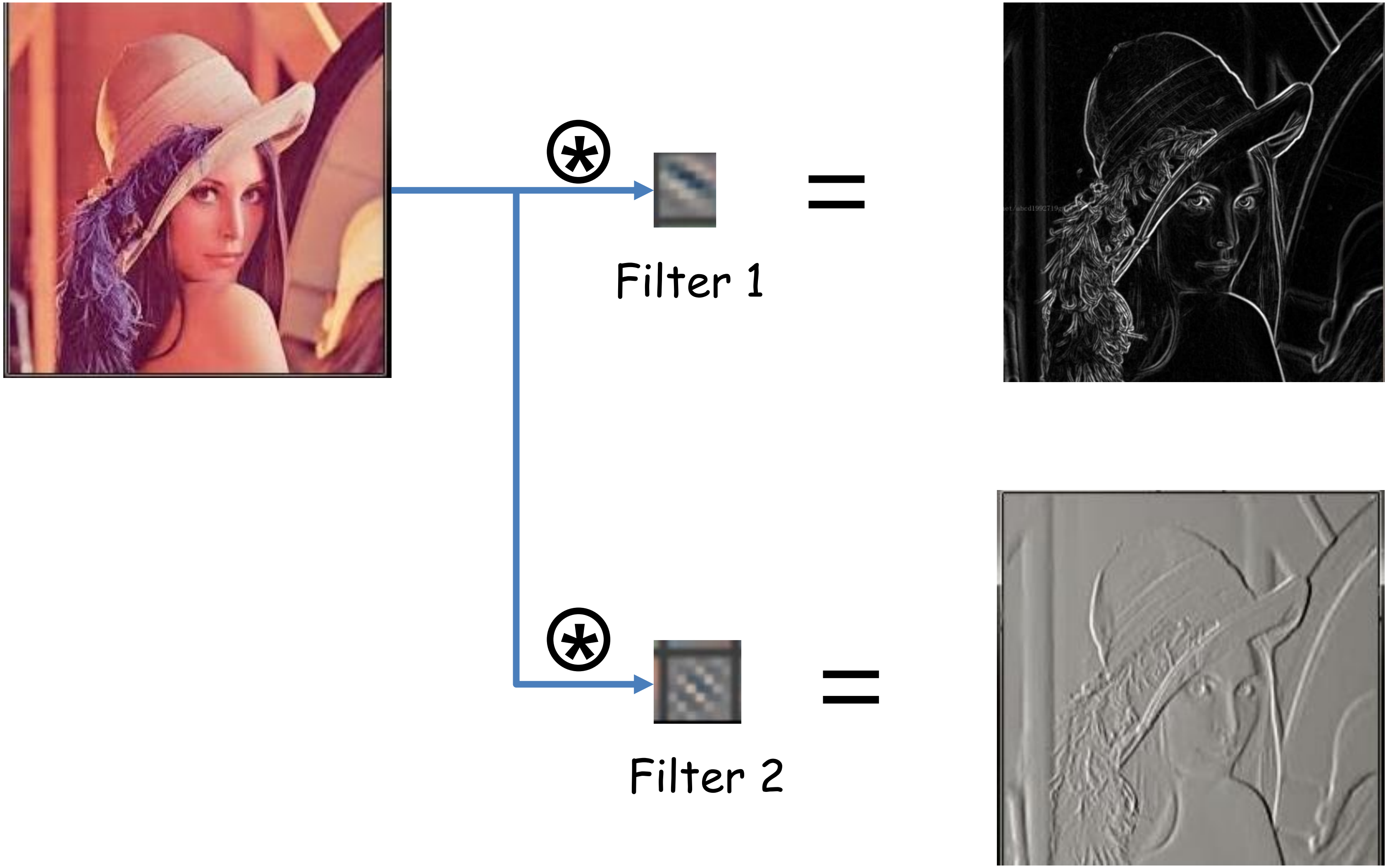
- Convolution과 Pooling을 반복하여 상위 Feature를 구성
- Convolution은 Local영역에서의 특정 Feature를 얻는 과정
- Pooling은 Dimension을 줄이면서도, Translation-invariant 한 Feature를 얻는 과정



Convolutional Neural Networks



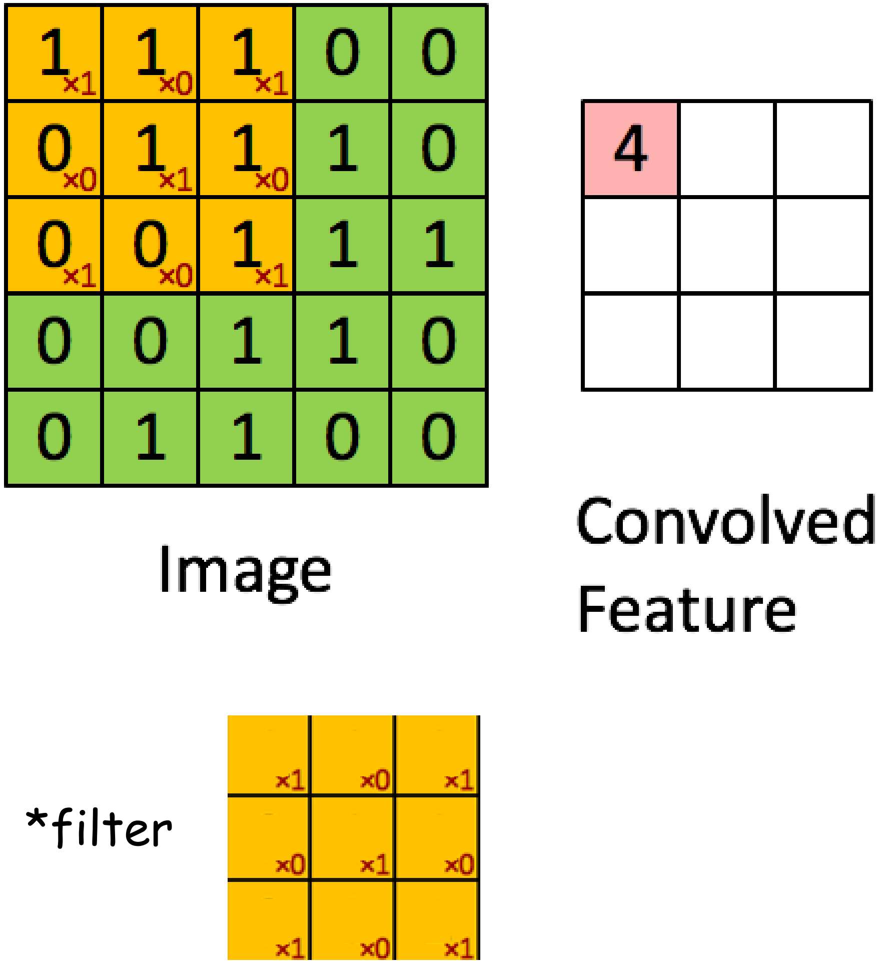
Convolution, Filter



Filter 1

Filter 2

Convolution, Filter



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

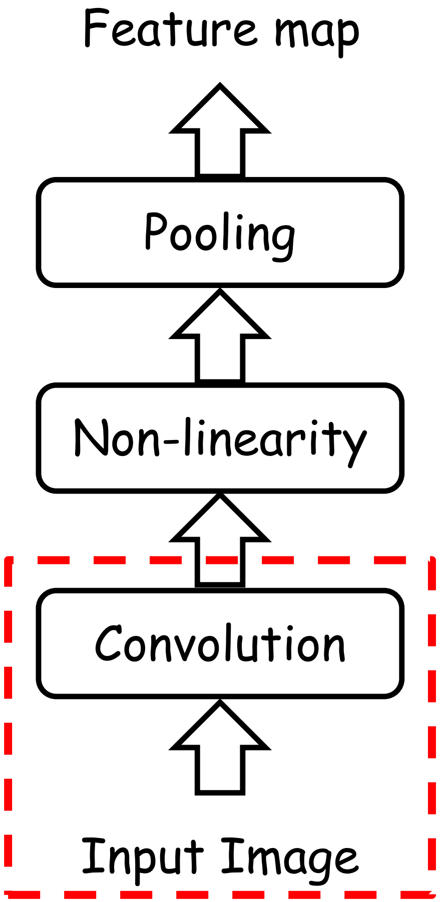
Image

4		

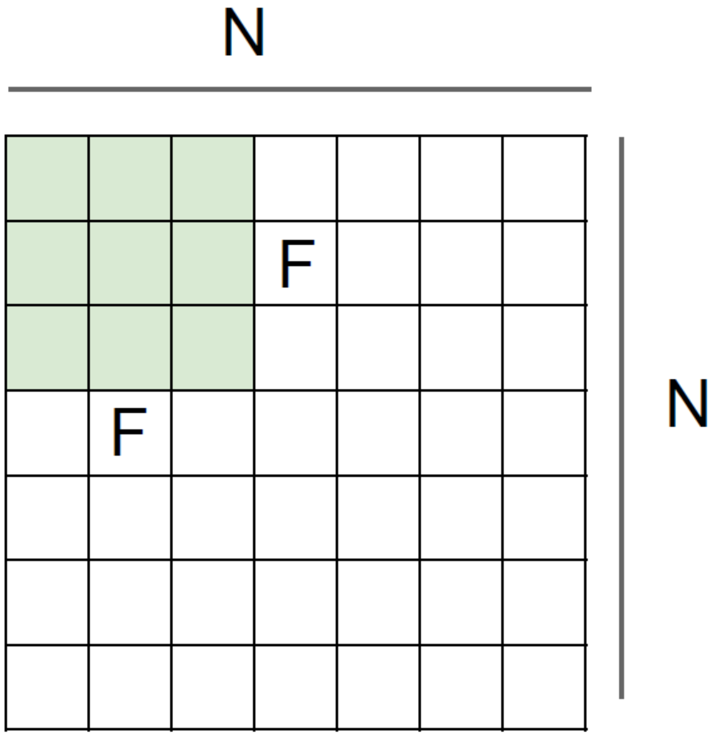
Convolved Feature

*filter

×1	×0	×1
×0	×1	×0
×1	×0	×1



Convolution, Filter

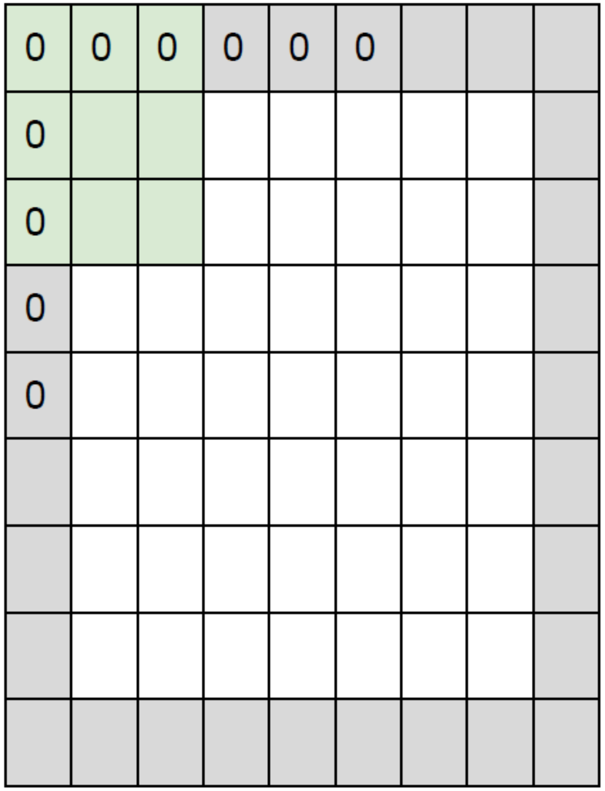


Output size:
(N - F) / stride + 1

e.g. N = 7, F = 3:
stride 1 => (7 - 3)/1 + 1 = 5
stride 2 => (7 - 3)/2 + 1 = 3
stride 3 => (7 - 3)/3 + 1 = 2.33 :\

Convolution, Filter

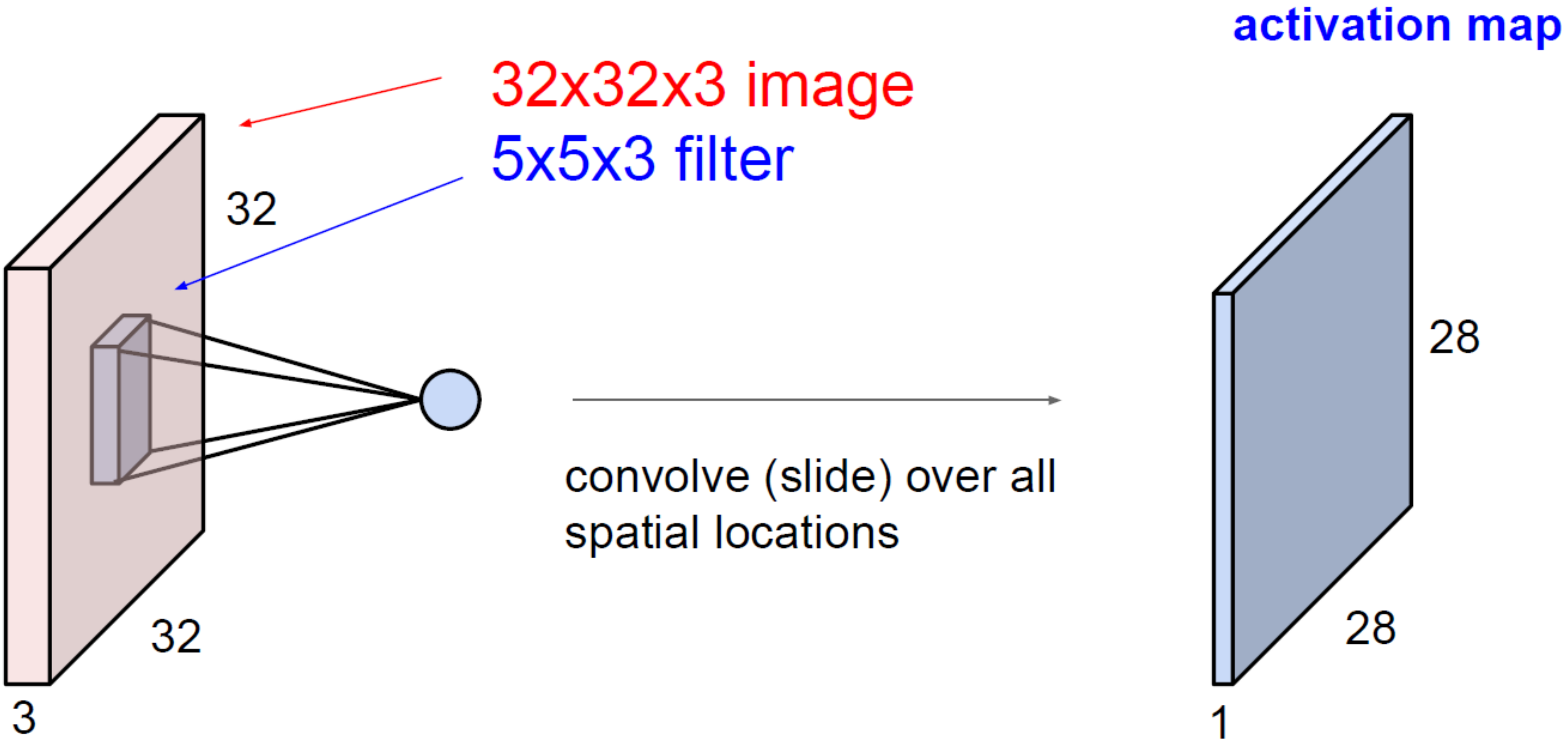
In practice: Common to zero pad the border



e.g. input 7x7
3x3 filter, applied with **stride 1**
pad with 1 pixel border => what is the output?

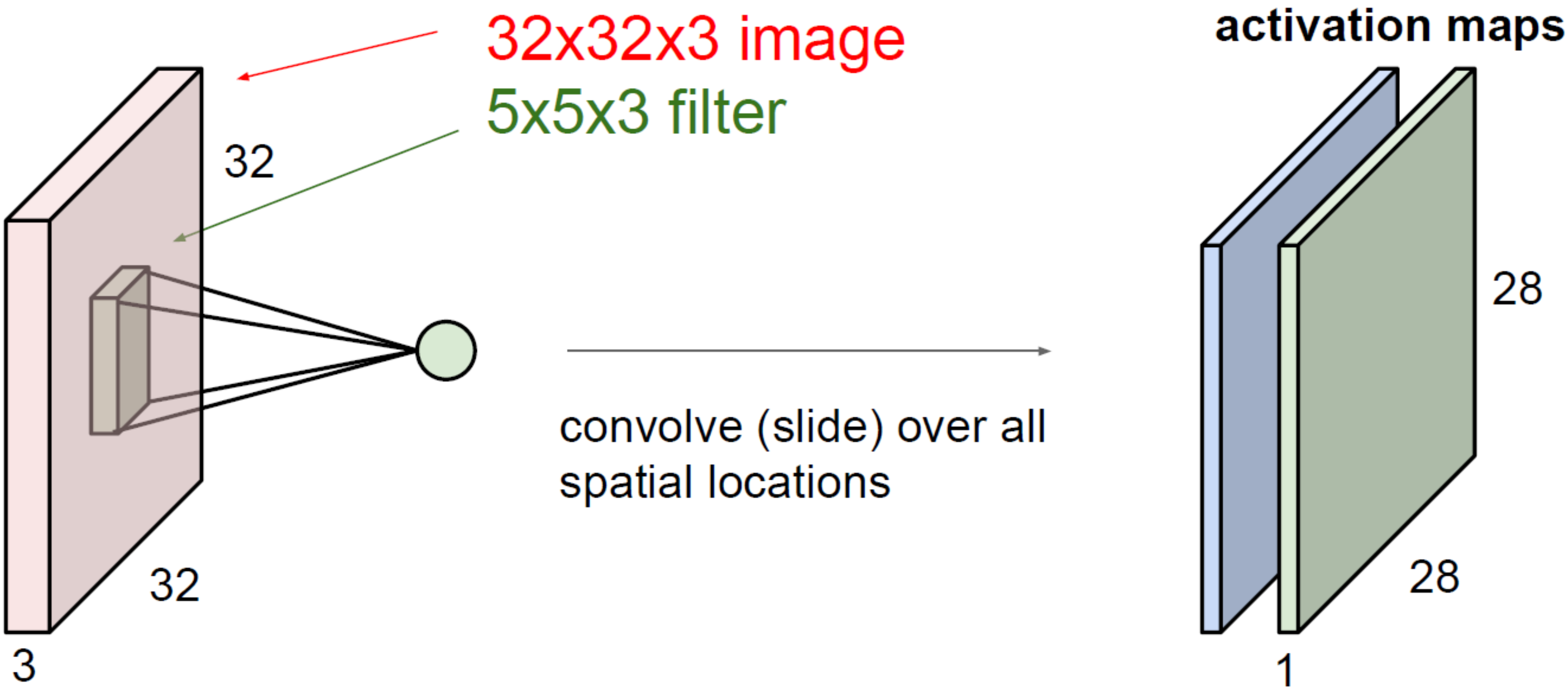
(recall:)
(N - F) / stride + 1

Convolution, Filter



Convolution, Filter

consider a second, green filter



32x32x3 image

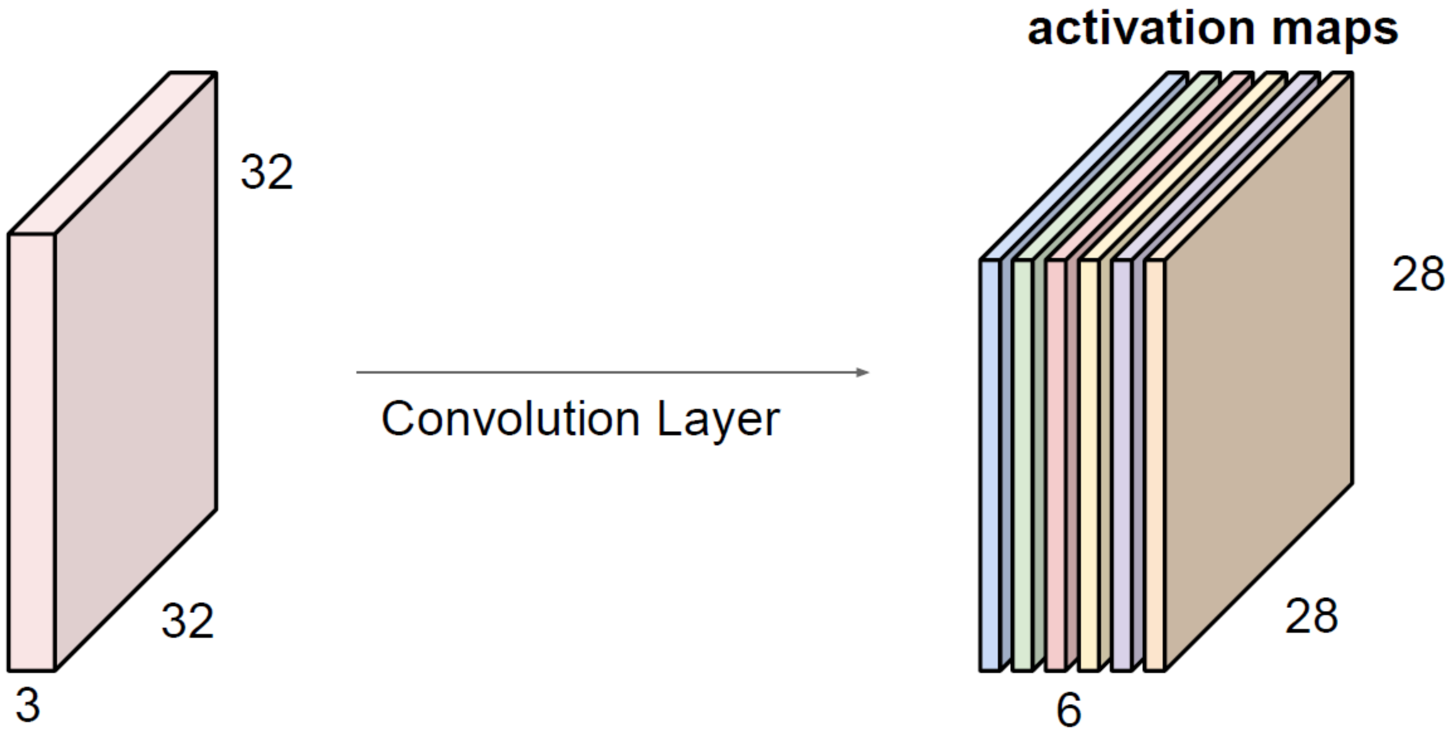
5x5x3 filter

convolve (slide) over all spatial locations

activation maps

Convolution, Filter

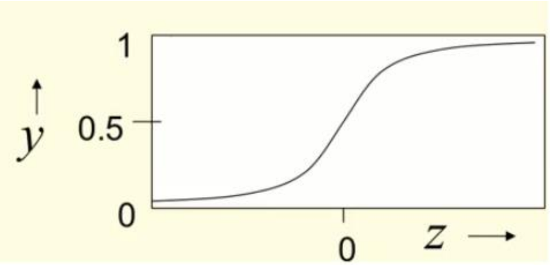
For example, if we had 6 5x5 filters, we'll get 6 separate activation maps:



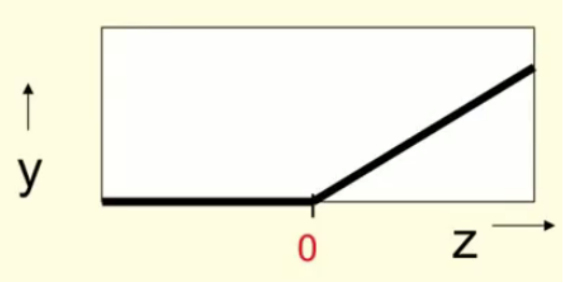
We stack these up to get a "new image" of size 28x28x6!

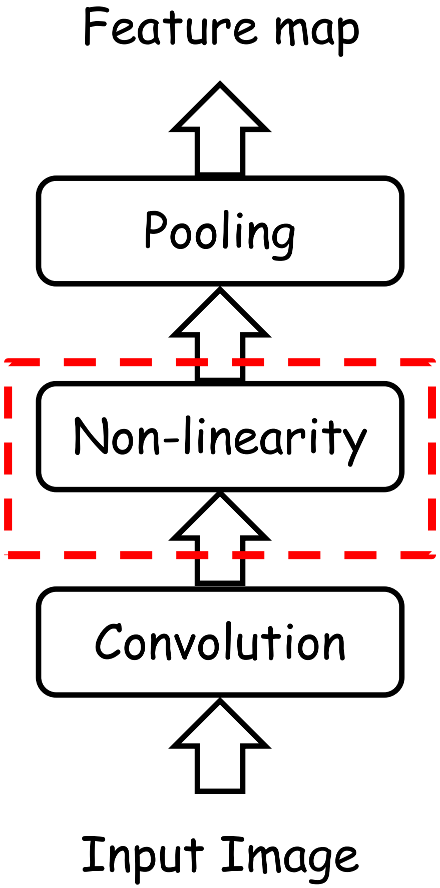
Non-linearity

Sigmoid

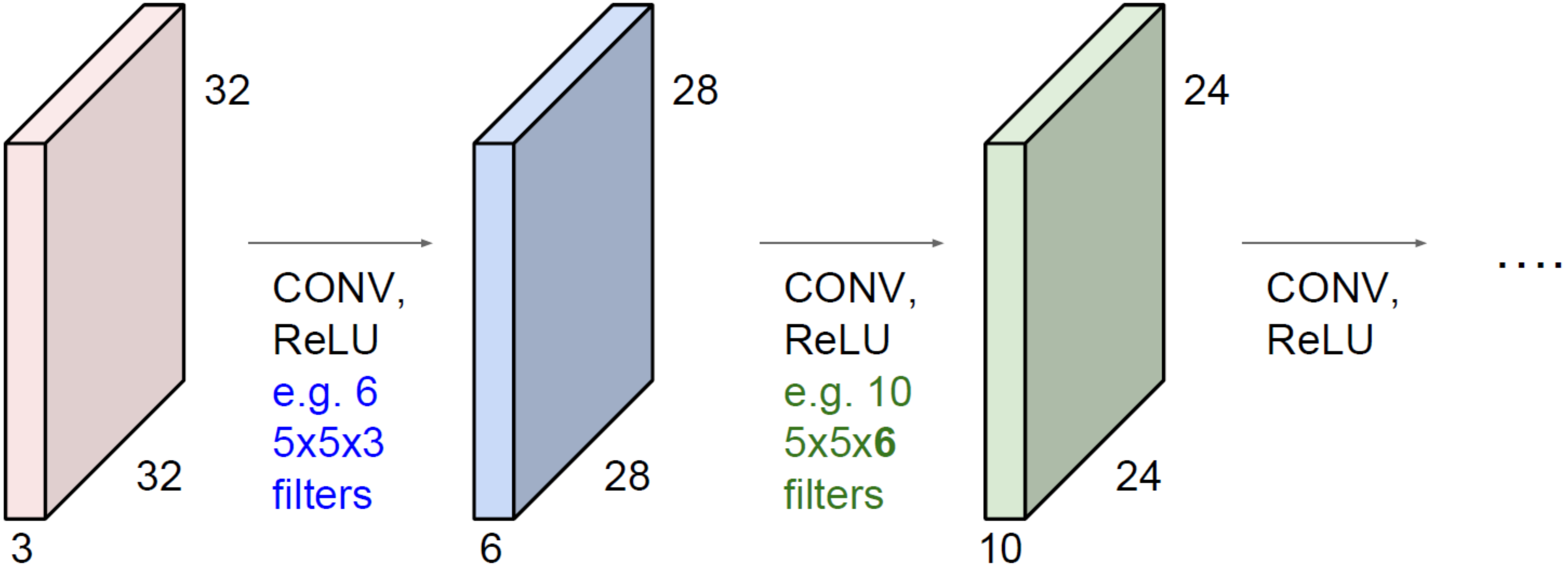


Rectified linear unit



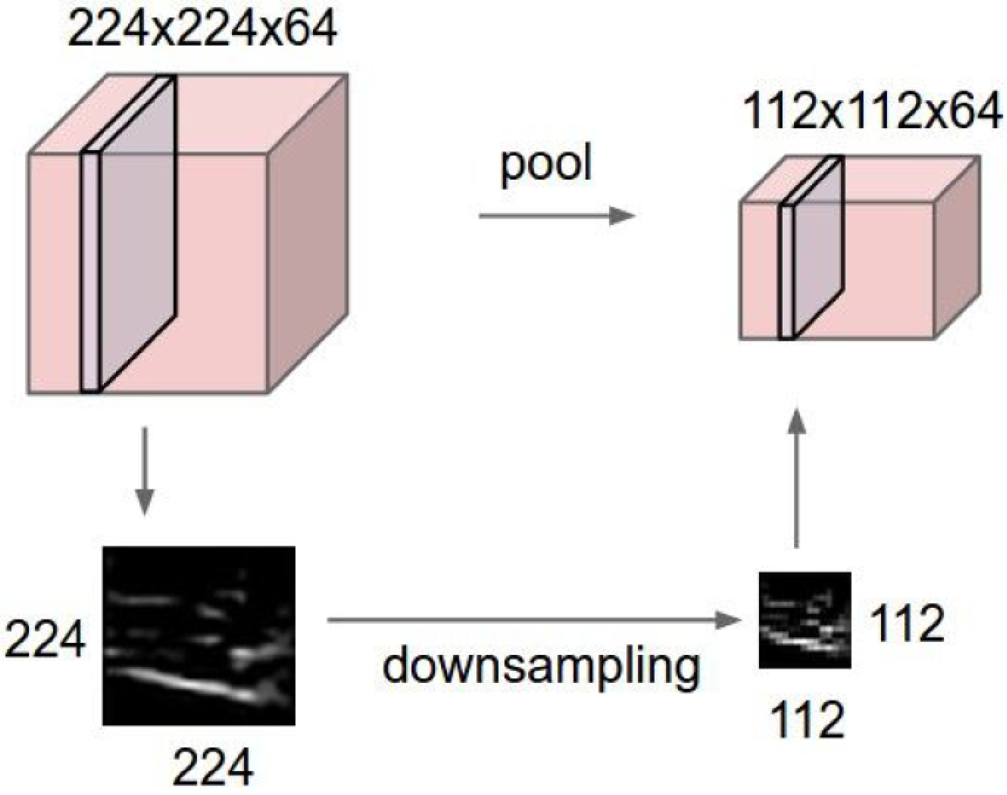


Convolution, Filter

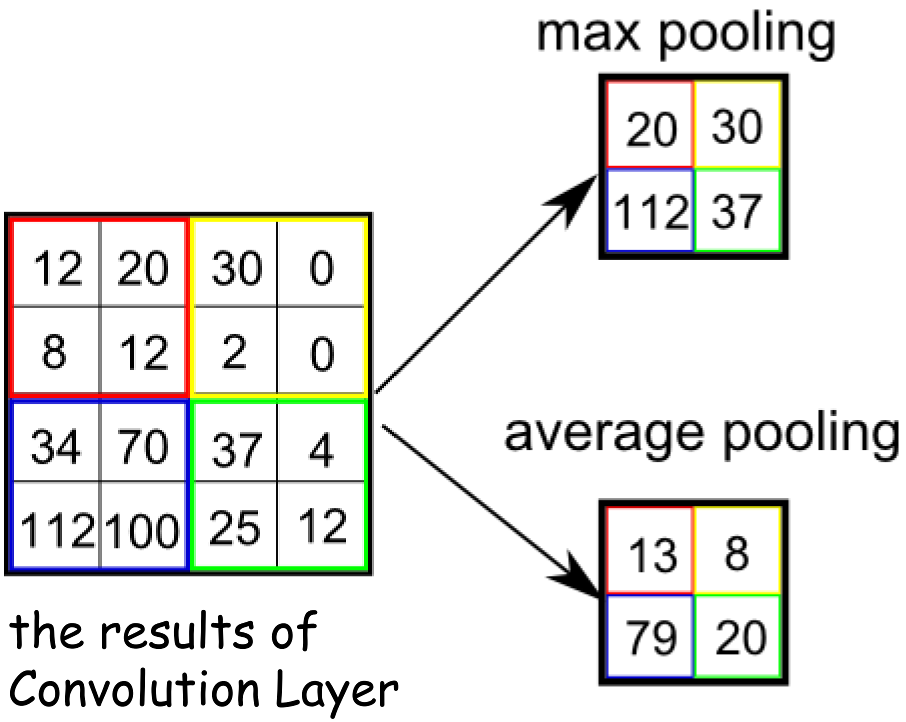


Pooling

- makes the representations smaller and more manageable
- operates over each activation map independently:



Pooling



12	20	30	0
8	12	2	0
34	70	37	4
112	100	25	12

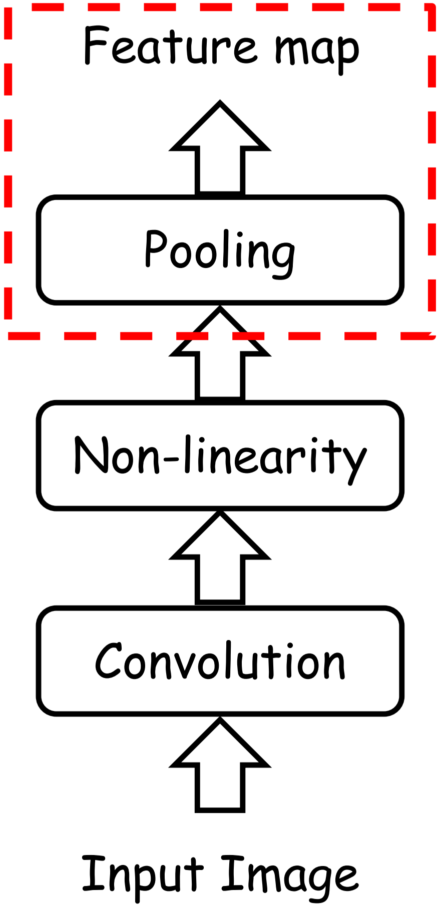
the results of
Convolution Layer

max pooling

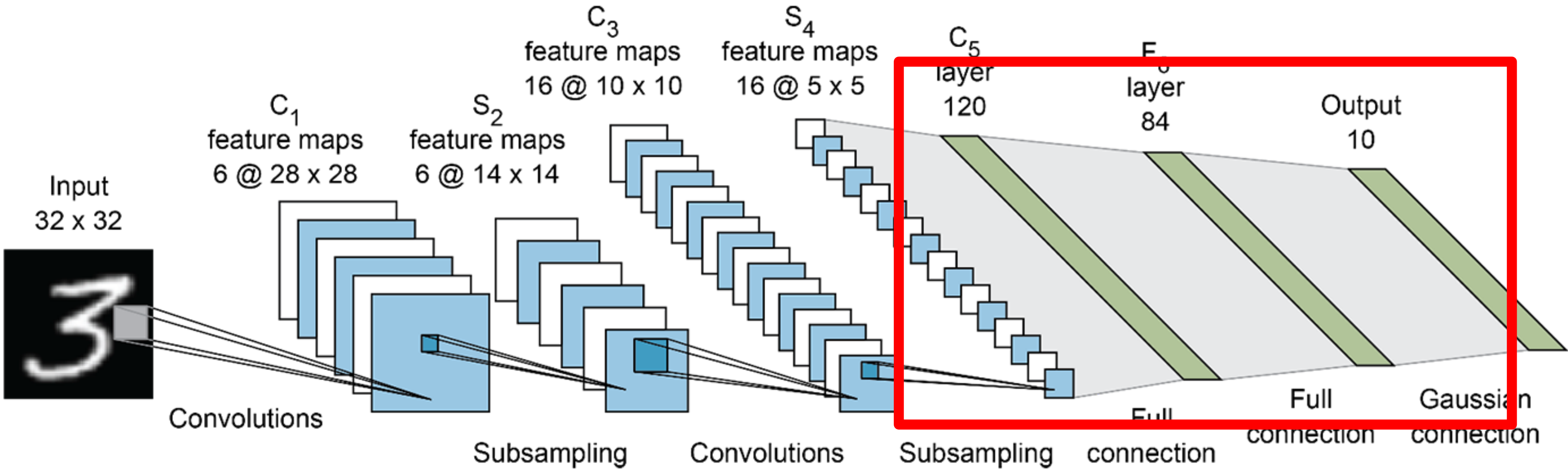
20	30
112	37

average pooling

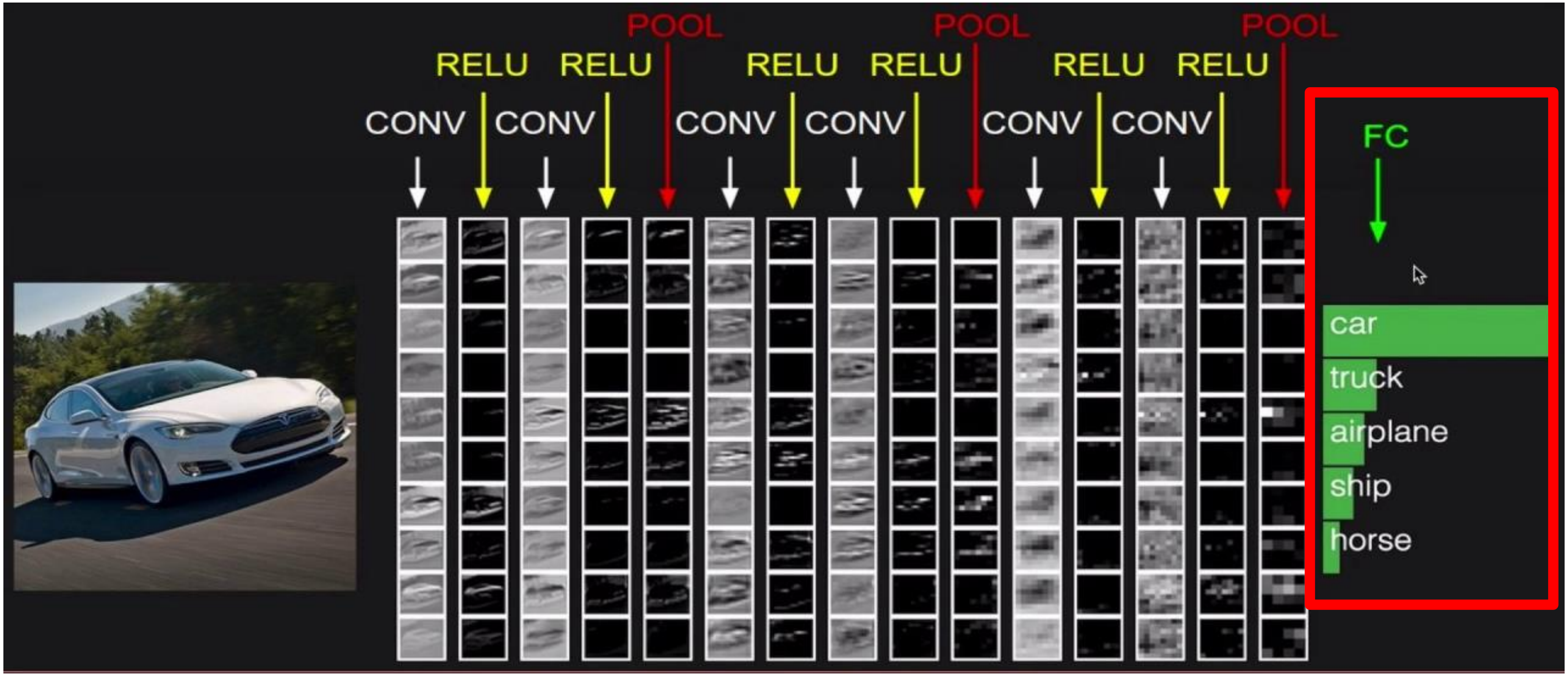
13	8
79	20



Fully connected layer

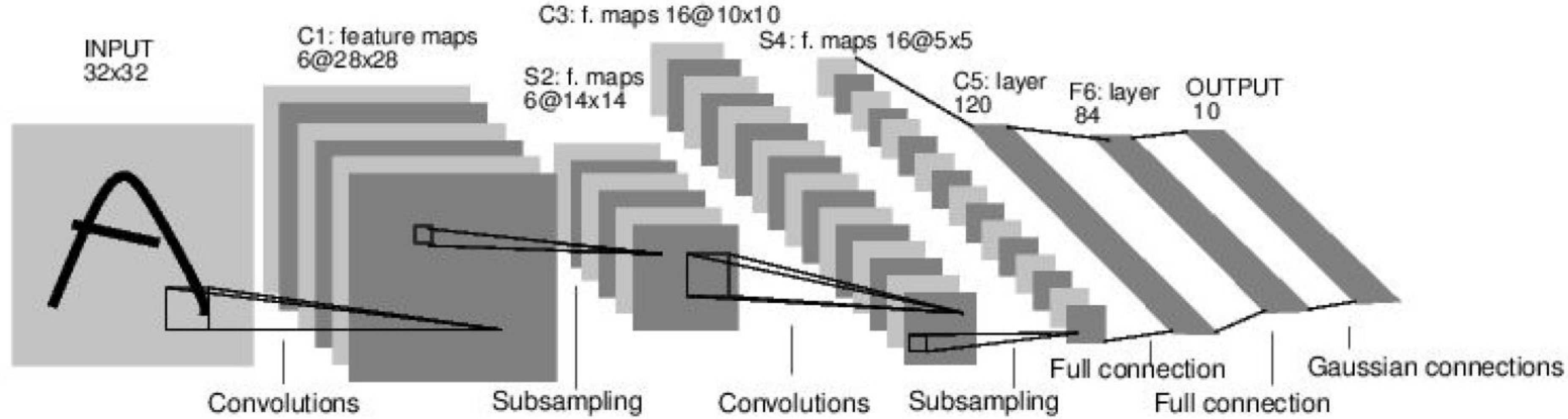


Fully connected layer



LeNet-5

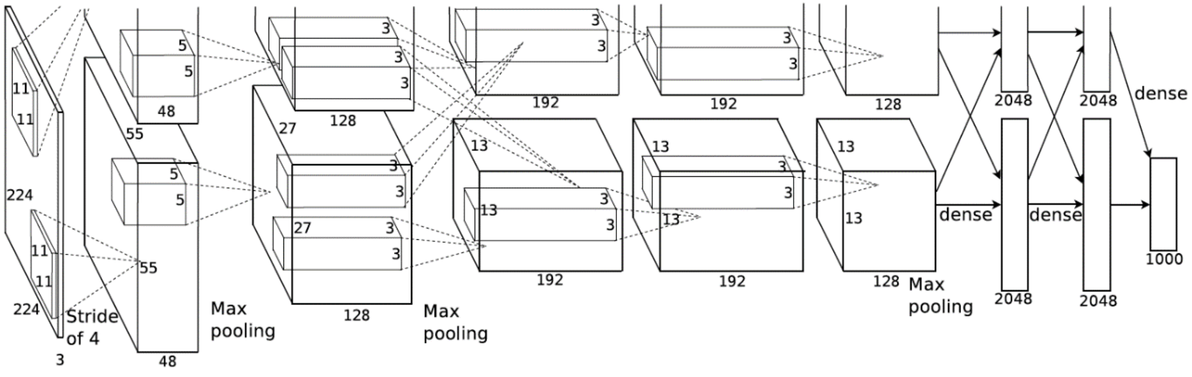
- LeNet [LeCun et al., 1998]



Conv filters were 5x5, applied at stride 1
Subsampling (Pooling) layers were 2x2 applied at stride 2
i.e. architecture is [CONV-POOL-CONV-POOL-CONV-FC]

CNN for object recognition

- AlexNet [Krizhevsky et al., 2012]

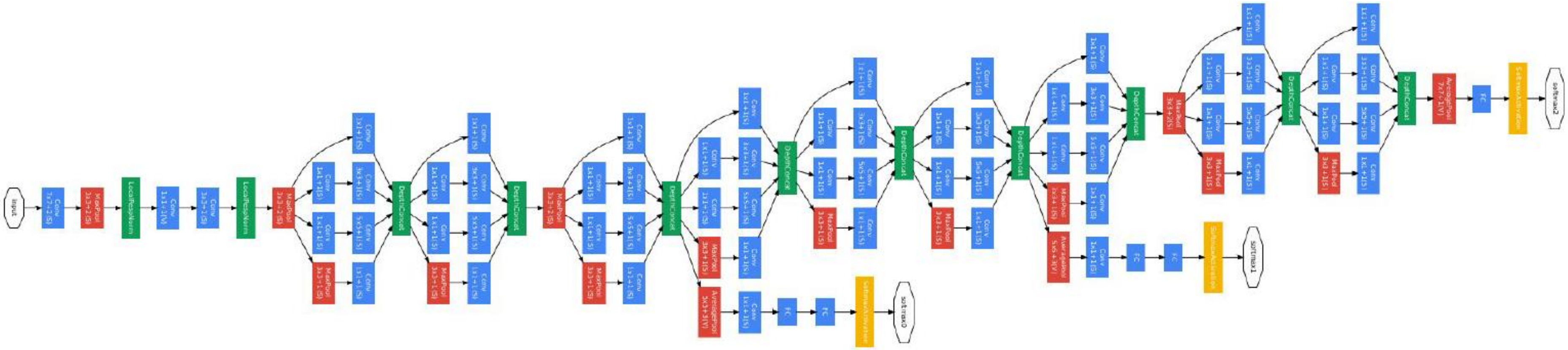


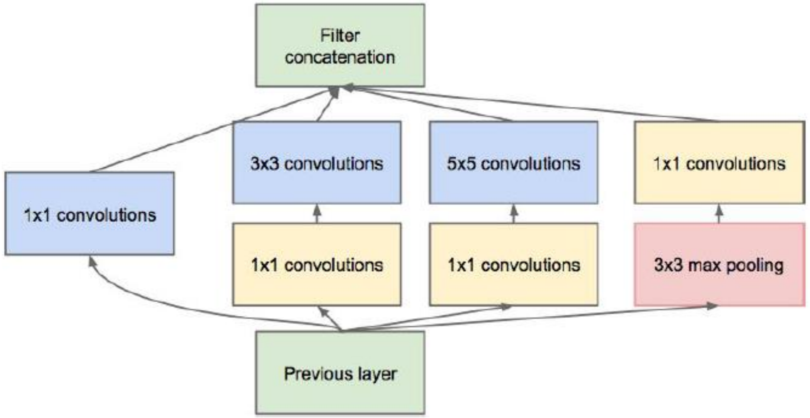
Full (simplified) AlexNet architecture:
[227x227x3] INPUT
[55x55x96] CONV1: 96 11x11 filters at stride 4, pad 0
[27x27x96] MAX POOL1: 3x3 filters at stride 2
[27x27x96] NORM1: Normalization layer
[27x27x256] CONV2: 256 5x5 filters at stride 1, pad 2
[13x13x256] MAX POOL2: 3x3 filters at stride 2
[13x13x256] NORM2: Normalization layer
[13x13x384] CONV3: 384 3x3 filters at stride 1, pad 1
[13x13x384] CONV4: 384 3x3 filters at stride 1, pad 1
[13x13x256] CONV5: 256 3x3 filters at stride 1, pad 1
[6x6x256] MAX POOL3: 3x3 filters at stride 2
[4096] FC6: 4096 neurons
[4096] FC7: 4096 neurons
[1000] FC8: 1000 neurons (class scores)

Details/Retrospectives:
- first use of ReLU
- used Norm layers (not common anymore)
- heavy data augmentation
- dropout 0.5
- batch size 128
- SGD Momentum 0.9
- Learning rate 1e-2, reduced by 10 manually when val accuracy plateaus
- L2 weight decay 5e-4
- 7 CNN ensemble: 18.2% -> 15.4%

CNN for object recognition

- GoogleNet [Szegedy et al., 2014]





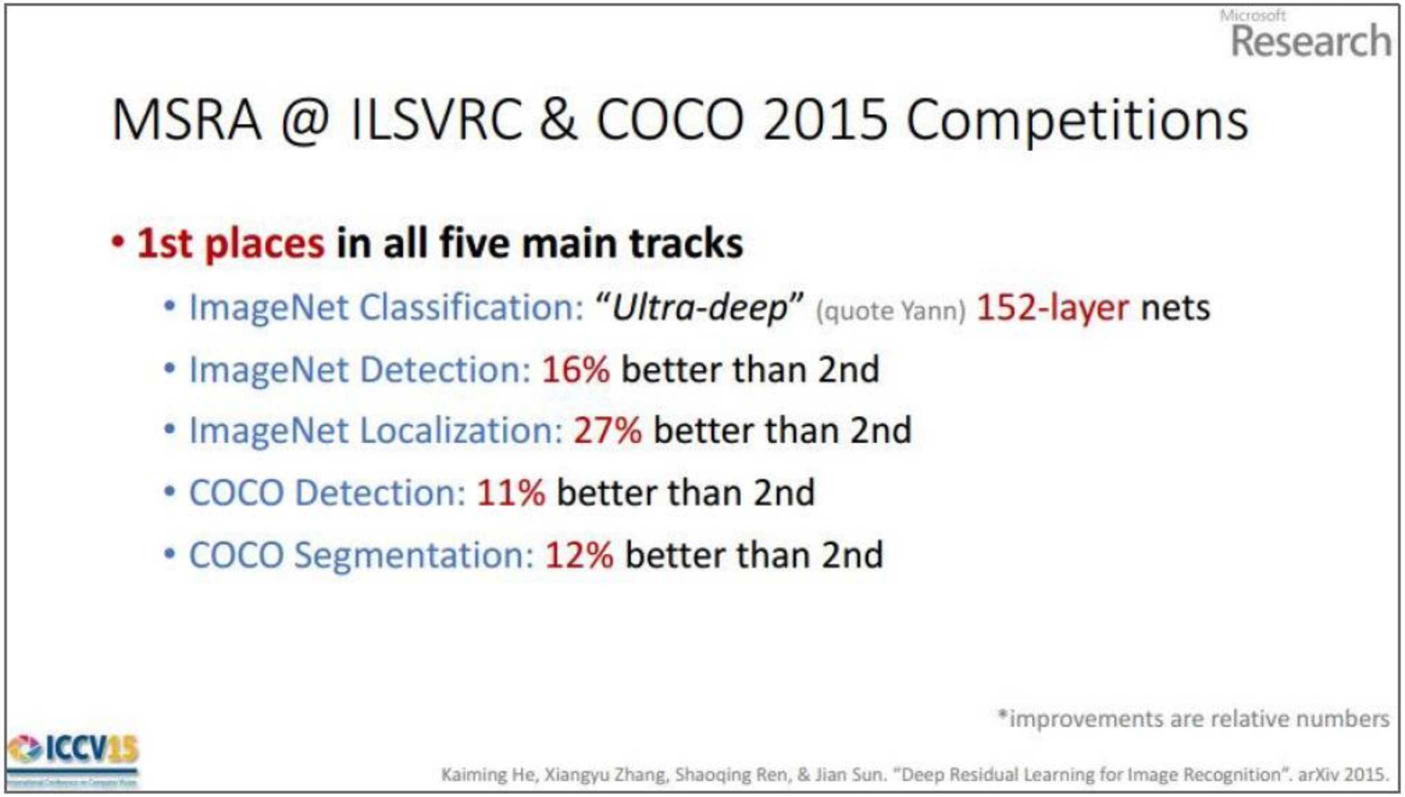
Inception module

ILSVRC 2014 winner (6.7% top 5 error)

CNN for object recognition

- ResNet [He et al., 2015]

ILSVRC 2015 winner (3.6% top 5 error)



Microsoft Research

MSRA @ ILSVRC & COCO 2015 Competitions

- **1st places in all five main tracks**
  - ImageNet Classification: *"Ultra-deep"* (quote Yann) 152-layer nets
  - ImageNet Detection: 16% better than 2nd
  - ImageNet Localization: 27% better than 2nd
  - COCO Detection: 11% better than 2nd
  - COCO Segmentation: 12% better than 2nd

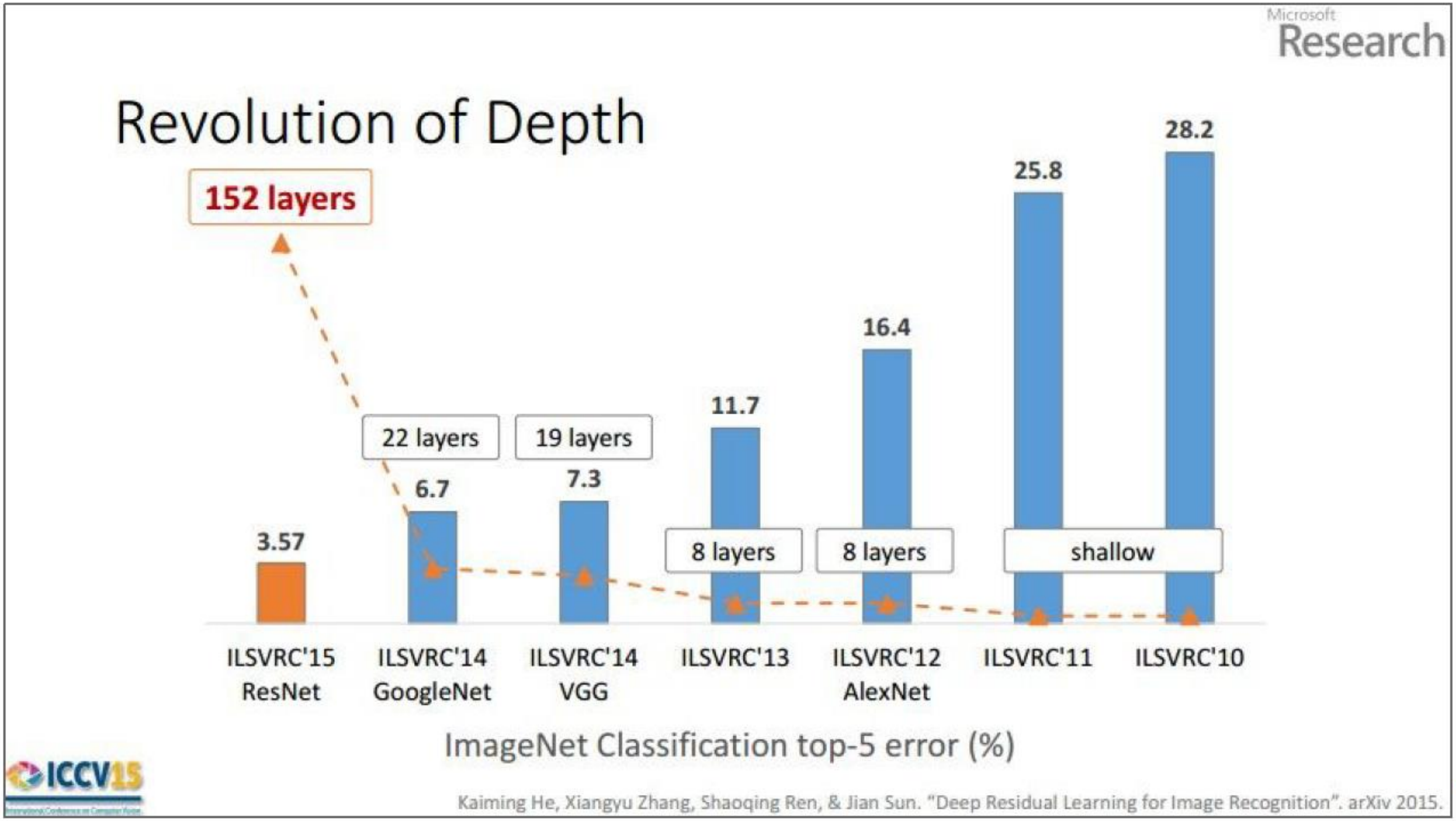
*improvements are relative numbers

ICCV15

Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun. "Deep Residual Learning for Image Recognition". arXiv 2015.

CNN for object recognition

- ResNet [He et al., 2015]

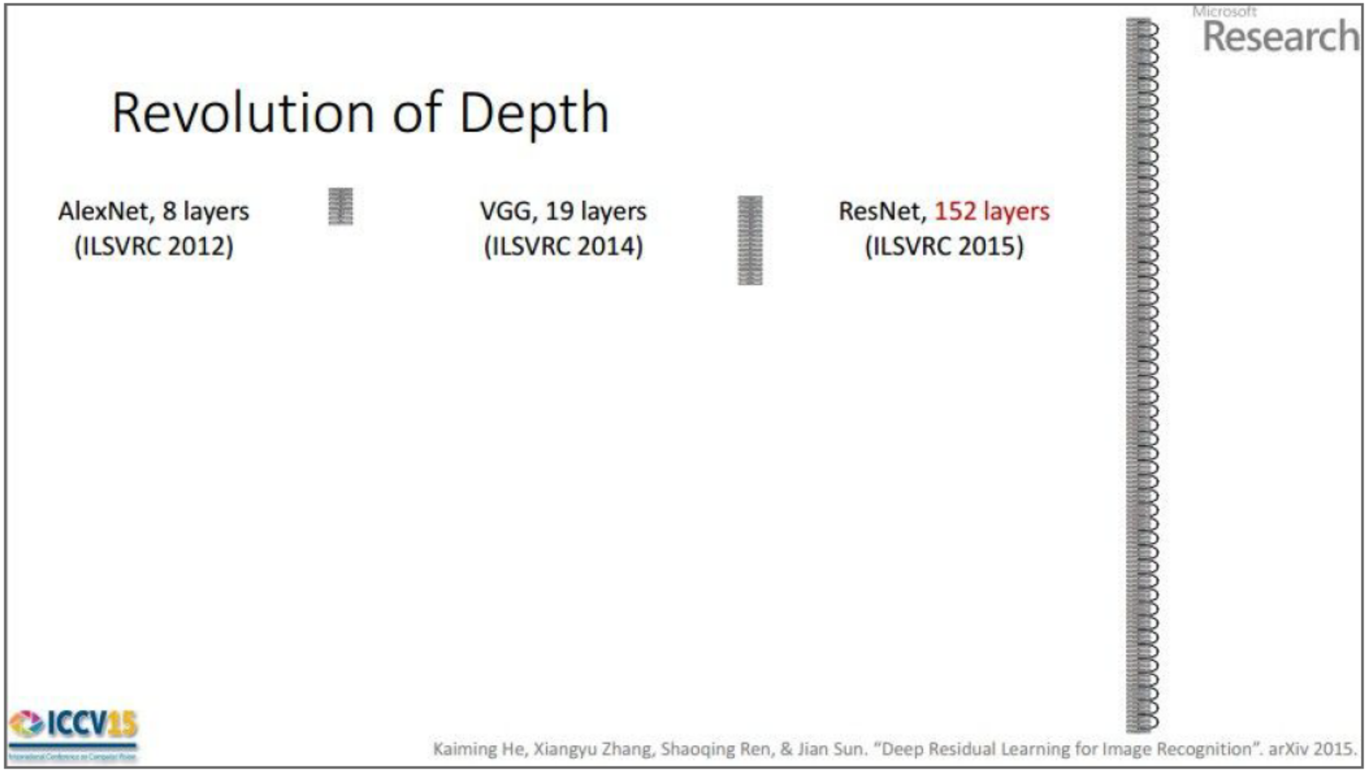


Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun. "Deep Residual Learning for Image Recognition". arXiv 2015.

CNN for object recognition

- ResNet [He et al., 2015]

ILSVRC 2015 winner (3.6% top 5 error)



Microsoft Research

Revolution of Depth

AlexNet, 8 layers
(ILSVRC 2012)

VGG, 19 layers
(ILSVRC 2014)

ResNet, 152 layers
(ILSVRC 2015)

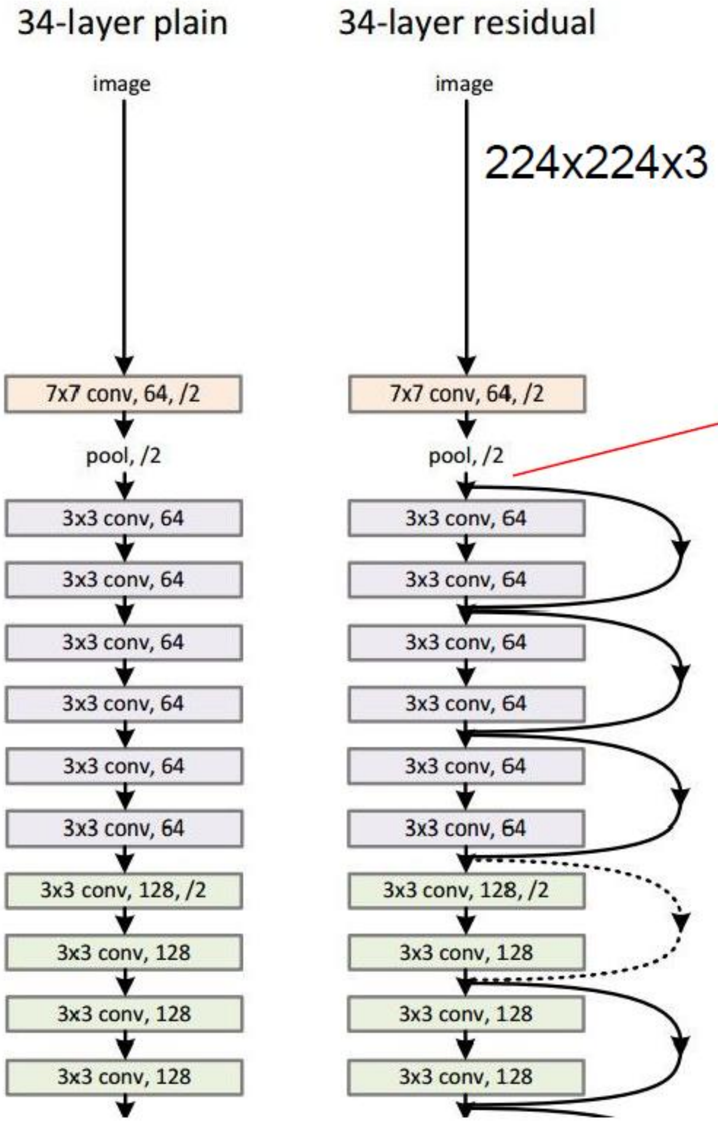
Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun. "Deep Residual Learning for Image Recognition". arXiv 2015.

2-3 weeks of training on 8 GPU machine

at runtime: faster than a VGGNet! (even though it has 8x more layers)

CNN for object recognition

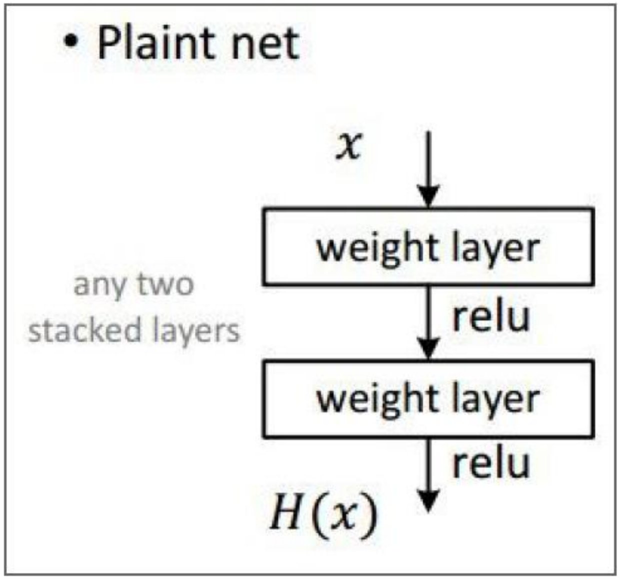
- ResNet [He et al., 2015]

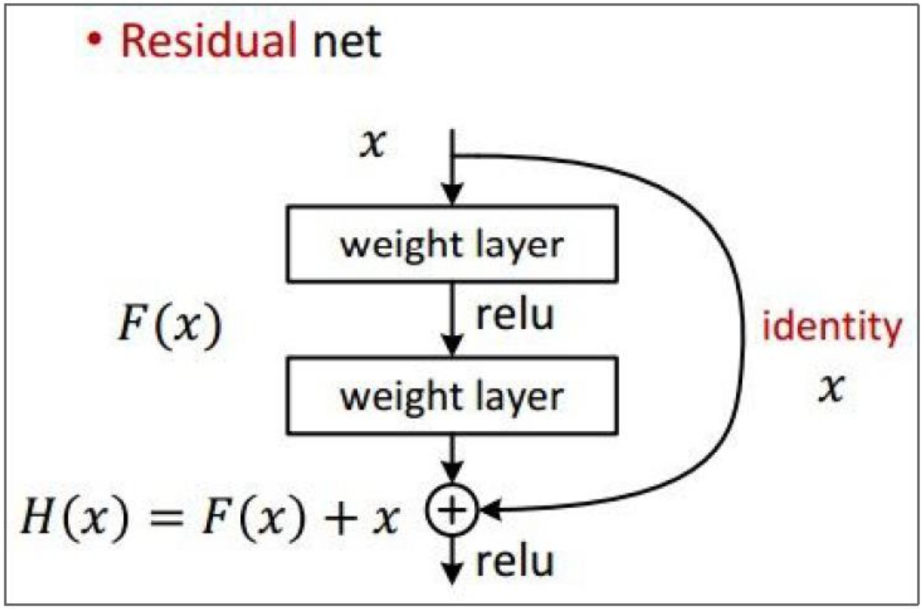


spatial dimension only 56x56!

CNN for object recognition

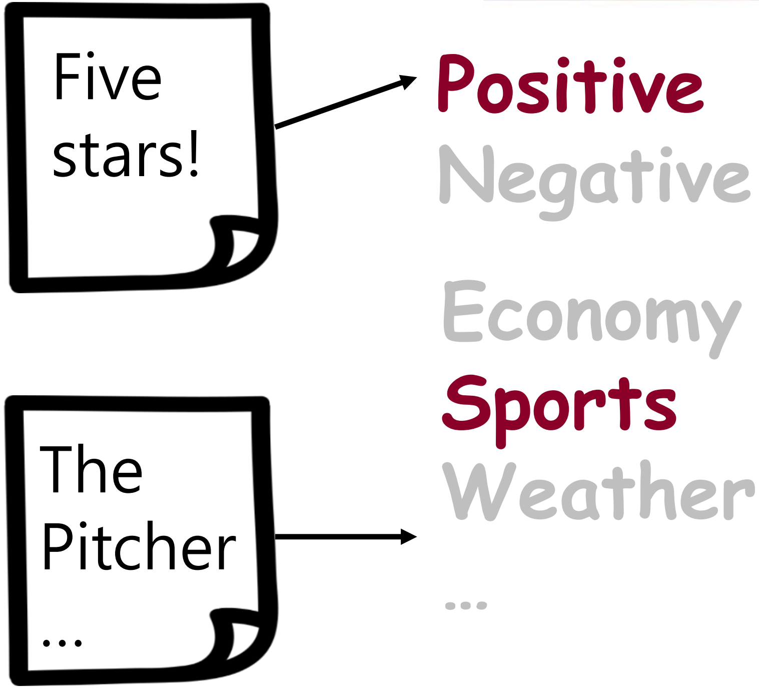
- ResNet [He et al., 2015]





CNN for text classification

- CNN for sentence classification [Kim et al., 2014]

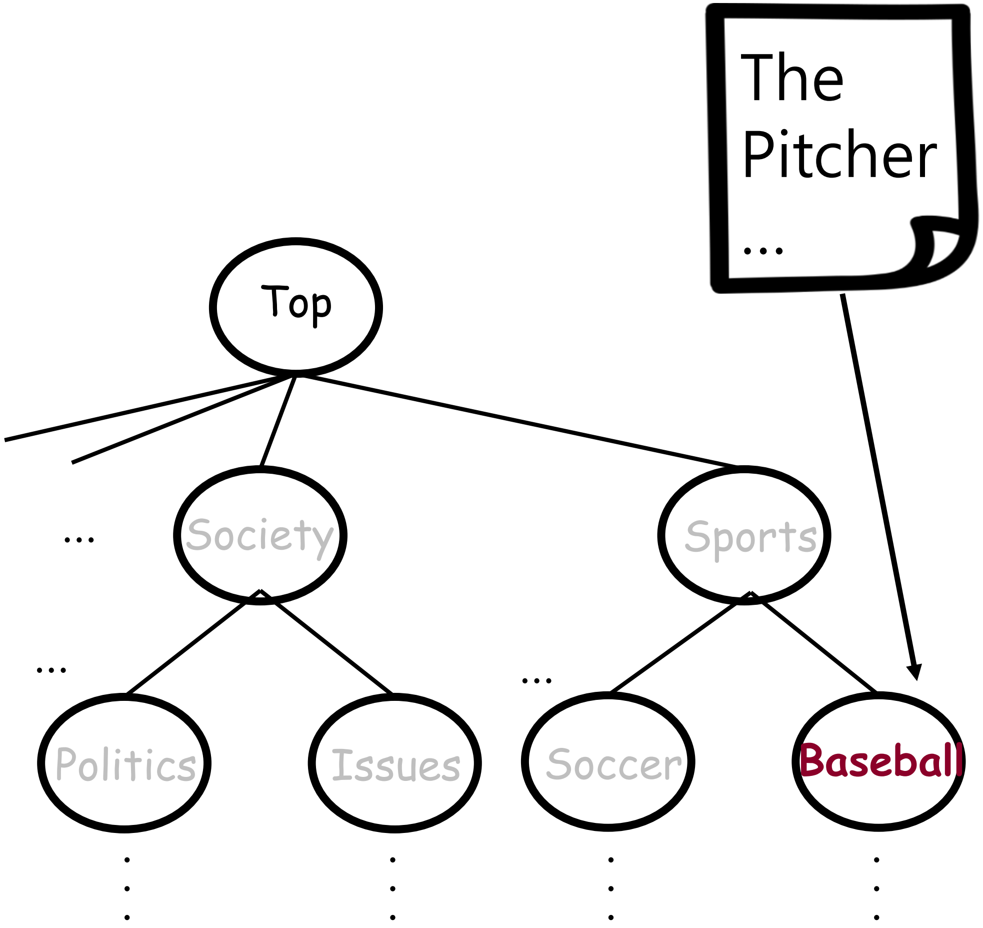


Five stars!

Positive
Negative

The Pitcher …

Economy
Sports
Weather
…



The Pitcher …

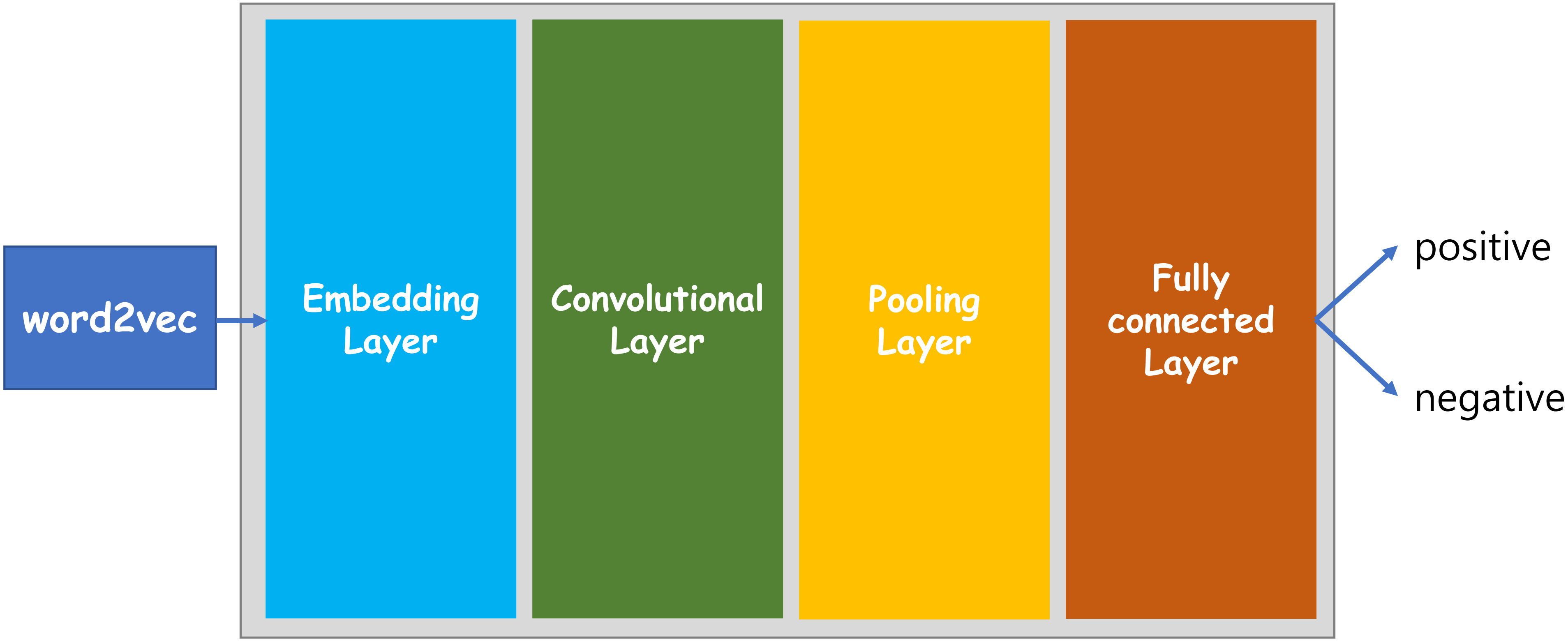
Top

…
Society
Sports

…
Politics
Issues
…
Soccer
Baseball

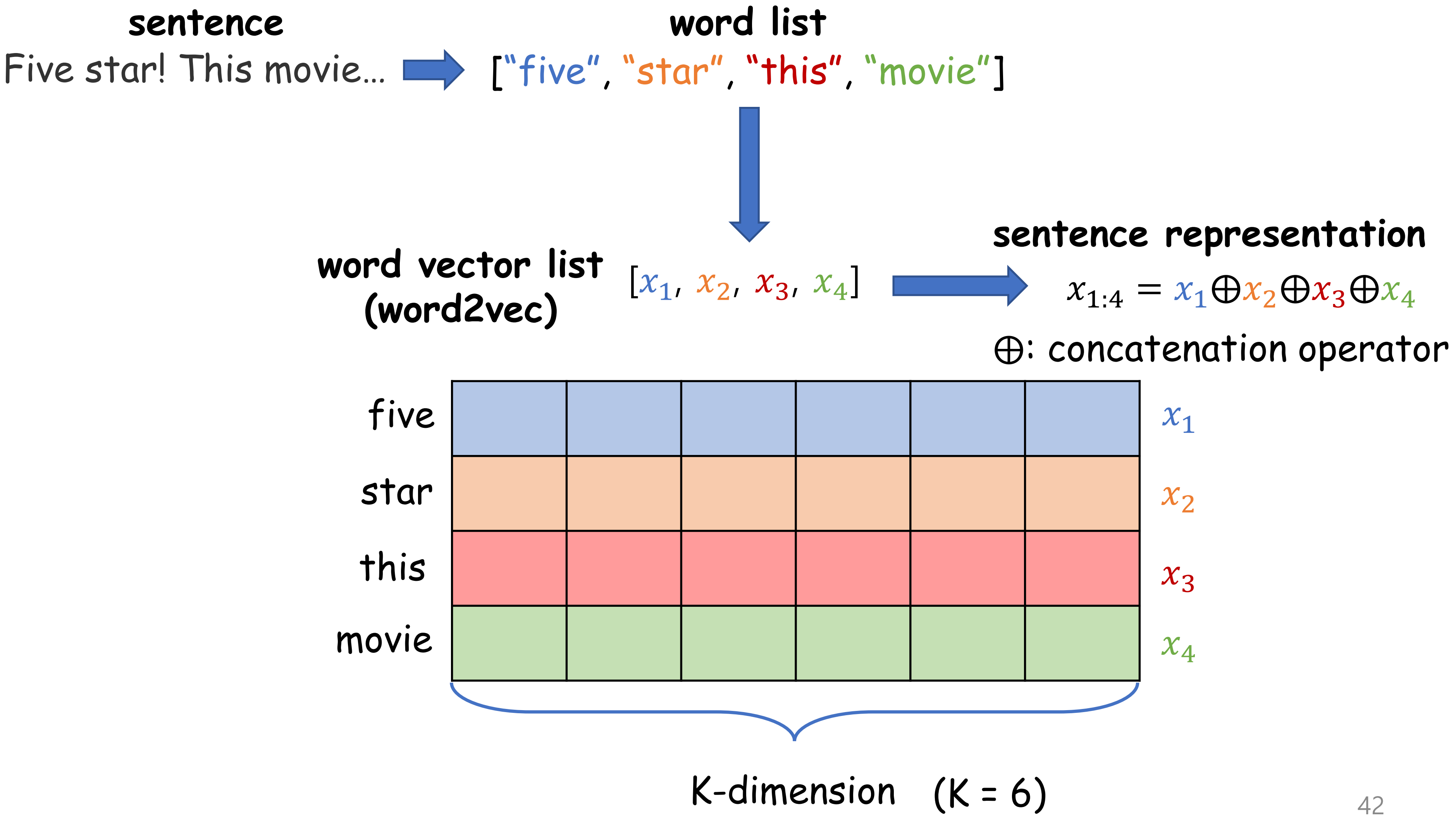
CNN for text classification

- Model overview



CNN for text classification

- Word2vec & Embedding layer



sentence

Five star! This movie…

word list

[“five”, “star”, “this”, “movie”]

word vector list (word2vec)

$[x_1, x_2, x_3, x_4]$

sentence representation

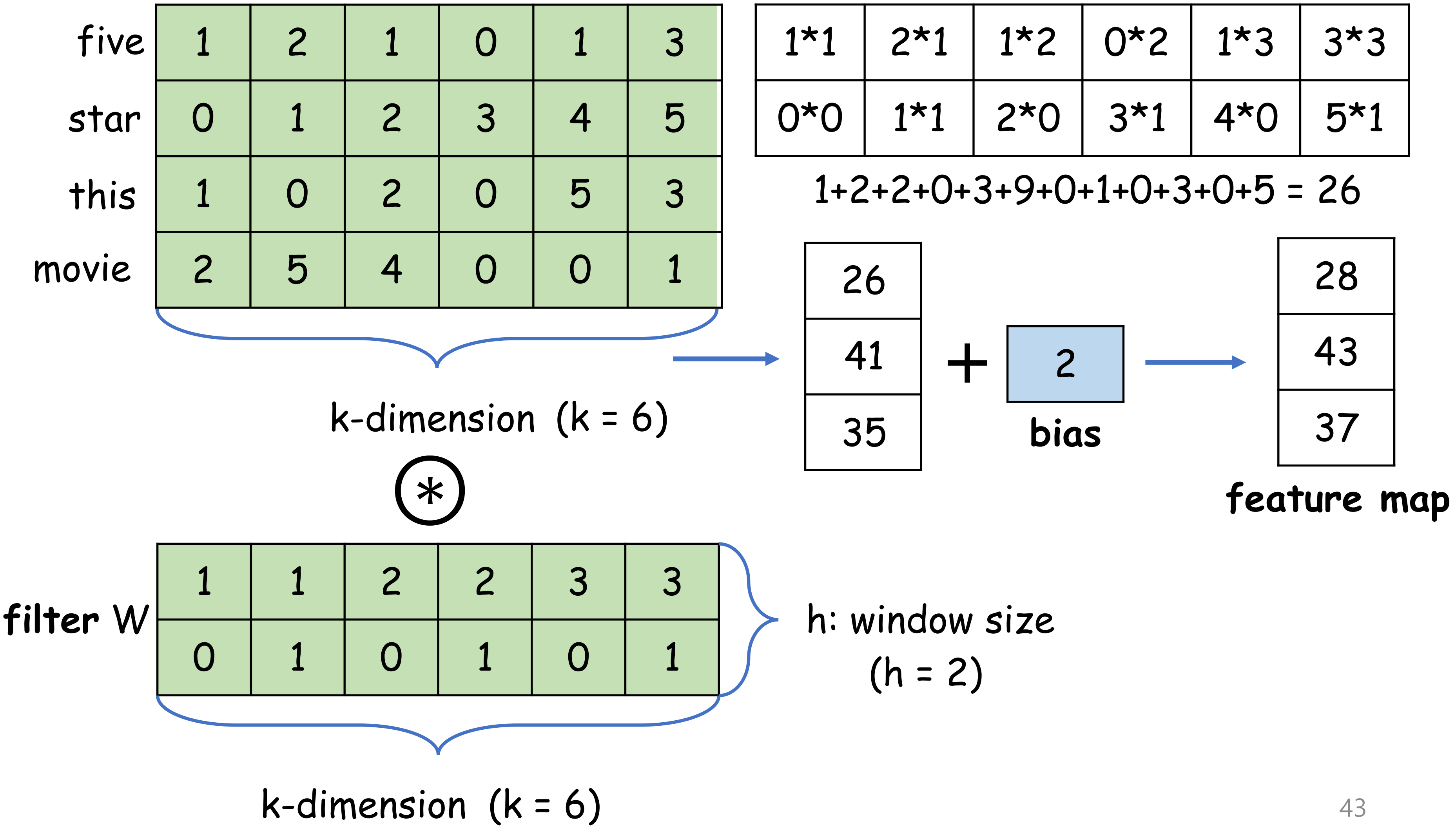
$$x_{1:4} = x_1 \oplus x_2 \oplus x_3 \oplus x_4$$

$\oplus$: concatenation operator

K-dimension (K = 6)

CNN for text classification

- Convolutional layer



five	1	2	1	0	1	3
star	0	1	2	3	4	5
this	1	0	2	0	5	3
movie	2	5	4	0	0	1

k-dimension (k = 6)

⊛

filter W	1	1	2	2	3	3
	0	1	0	1	0	1

h: window size (h = 2)

k-dimension (k = 6)

1*1	2*1	1*2	0*2	1*3	3*3
0*0	1*1	2*0	3*1	4*0	5*1

1+2+2+0+3+9+0+1+0+3+0+5 = 26

26
41
35

+

2

bias

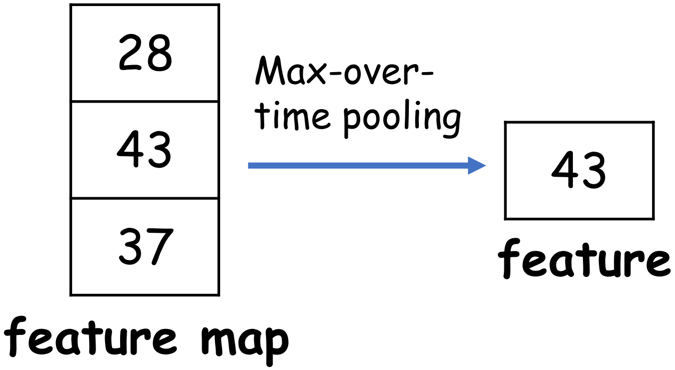
28
43
37

feature map

CNN for text classification

- Pooling layer

five	1	2	1	0	1	3
star	0	1	2	3	4	5
this	1	0	2	0	5	3
movie	2	5	4	0	0	1

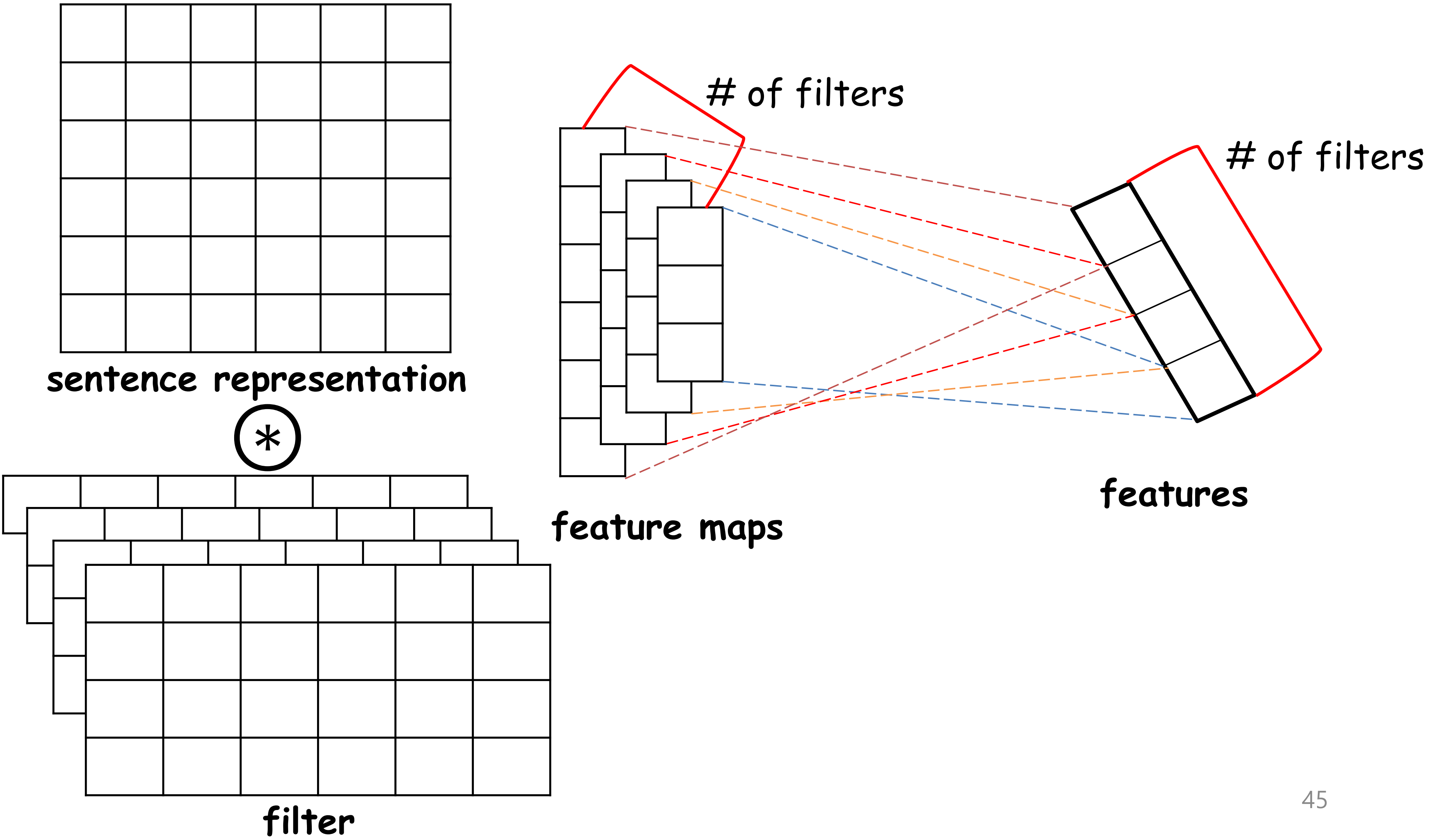


filter W	1	1	2	2	3	3
	0	1	0	1	0	1

One filter -> **One** feature
Multiple filter -> **Multiple** feature

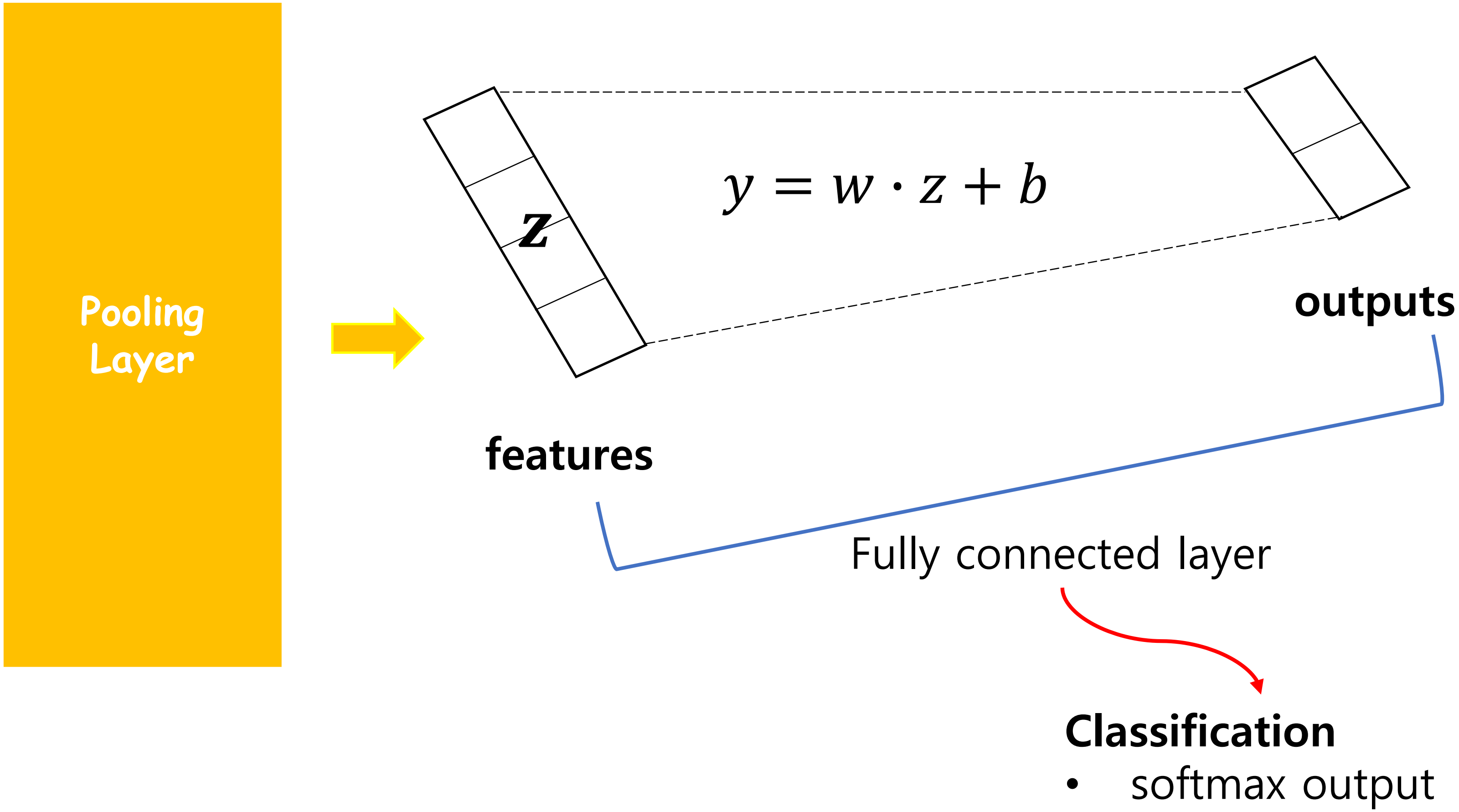
CNN for text classification

- Pooling layer



CNN for text classification

- Fully connected layer



Pooling Layer

z

$$y = w \cdot z + b$$

outputs

features

Fully connected layer

Classification

- softmax output

AI School 6기 5주차

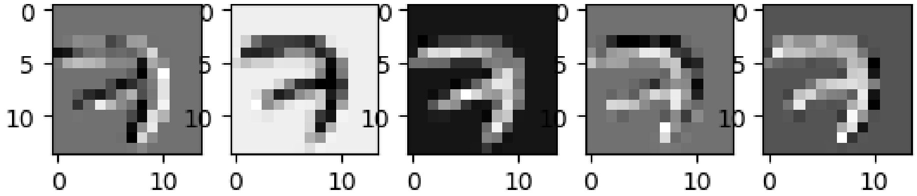
CNN 모델을 활용한 객체 분류

Image Input

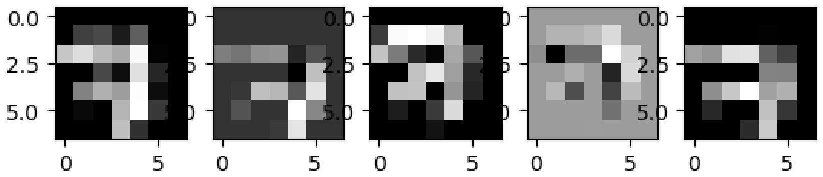
```
X = tf.placeholder(tf.float32, [None, 784] , name="X")
X_img = tf.reshape(X, [-1, 28, 28, 1])   # img 28x28x1 (black/white)
Y = tf.placeholder(tf.float32, [None, 10], name="Y")
keep_prob = tf.placeholder(tf.float32, name="keep_prob")
```

Convolution

```
import matplotlib.pyplot as plt
import numpy as np
import tensorflow as tf

from tensorflow.examples.tutorials.mnist import input_data

mnist = input_data.read_data_sets("MNIST_data/", one_hot=True)
img = mnist.train.images[0].reshape(28,28)
sess = tf.InteractiveSession()

img = img.reshape(-1,28,28,1)
W1 = tf.Variable(tf.random_normal([3, 3, 1, 5], stddev=0.01))
conv2d = tf.nn.conv2d(img, W1, strides=[1, 2, 2, 1], padding='SAME')
print(conv2d)
sess.run(tf.global_variables_initializer())
conv2d_img = conv2d.eval()
conv2d_img = np.swapaxes(conv2d_img, 0, 3)
for i, one_img in enumerate(conv2d_img):
    plt.subplot(1,5,i+1), plt.imshow(one_img.reshape(14,14), cmap='gray')
plt.show()
```



Pooling



```
import matplotlib.pyplot as plt
import numpy as np
import tensorflow as tf
from tensorflow.examples.tutorials.mnist import input_data

mnist = input_data.read_data_sets("MNIST_data/", one_hot=True)
img = mnist.train.images[0].reshape(28,28)
sess = tf.InteractiveSession()

img = img.reshape(-1,28,28,1)
W1 = tf.Variable(tf.random_normal([3, 3, 1, 5], stddev=0.01))
conv2d = tf.nn.conv2d(img, W1, strides=[1, 2, 2, 1], padding='SAME')
pool = tf.nn.max_pool(conv2d, ksize=[1, 2, 2, 1], strides=[1, 2, 2, 1], padding='SAME')
print(pool)
sess.run(tf.global_variables_initializer())
pool_img = pool.eval()
pool_img = np.swapaxes(pool_img, 0, 3)
for i, one_img in enumerate(pool_img):
    plt.subplot(1,5,i+1), plt.imshow(one_img.reshape(7, 7), cmap='gray')
plt.show()
```

Image Input

```
X = tf.placeholder(tf.float32, [None, 784] , name="X")
X_img = tf.reshape(X, [-1, 28, 28, 1])   # img 28x28x1 (black/white)
Y = tf.placeholder(tf.float32, [None, 10], name="Y")
keep_prob = tf.placeholder(tf.float32, name="keep_prob")
```

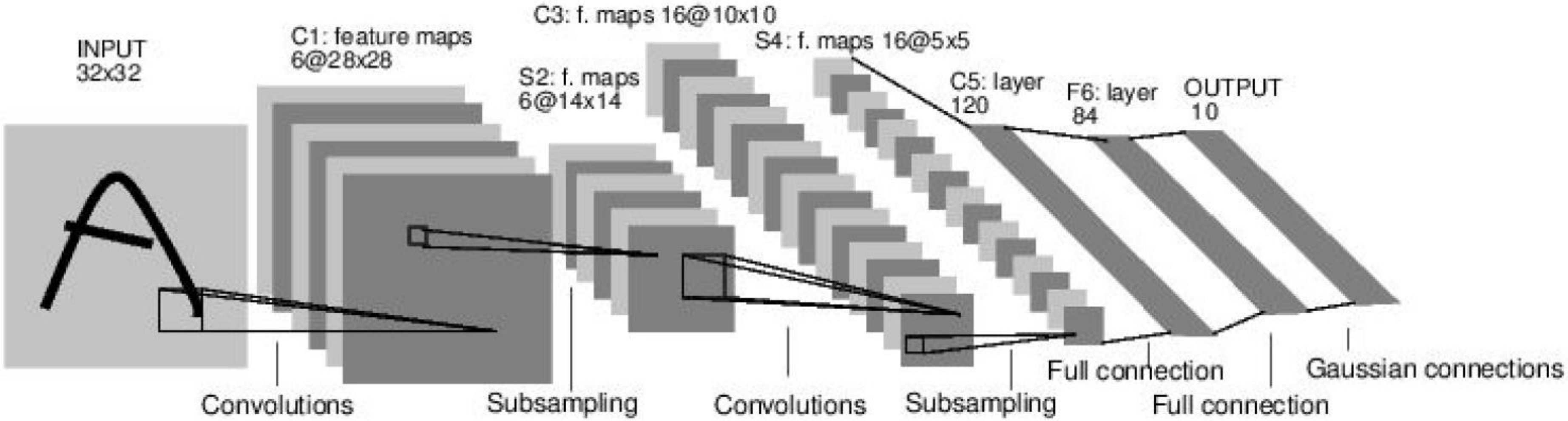
Convolutional & pooling layer 1, 2

```
# L1 ImgIn shape=(?, 28, 28, 1)
W1 = tf.Variable(tf.random_normal([3, 3, 1, 32], stddev=0.01))
#    Conv      -> (?, 28, 28, 32)
#    Pool      -> (?, 14, 14, 32)
L1 = tf.nn.conv2d(X_img, W1, strides=[1, 1, 1, 1], padding='SAME')
L1 = tf.nn.relu(L1)
L1 = tf.nn.max_pool(L1, ksize=[1, 2, 2, 1], strides=[1, 2, 2, 1], padding='SAME')
L1 = tf.nn.dropout(L1, keep_prob=keep_prob)
```

```
# L2 ImgIn shape=(?, 14, 14, 32)
W2 = tf.Variable(tf.random_normal([3, 3, 32, 64], stddev=0.01))
#    Conv       ->(?, 14, 14, 64)
#    Pool       ->(?, 7, 7, 64)
L2 = tf.nn.conv2d(L1, W2, strides=[1, 1, 1, 1], padding='SAME')
L2 = tf.nn.relu(L2)
L2 = tf.nn.max_pool(L2, ksize=[1, 2, 2, 1], strides=[1, 2, 2, 1], padding='SAME’)
L2 = tf.nn.dropout(L2, keep_prob=keep_prob)
L2_flat = tf.reshape(L2, [-1, 7 * 7 * 64])
```

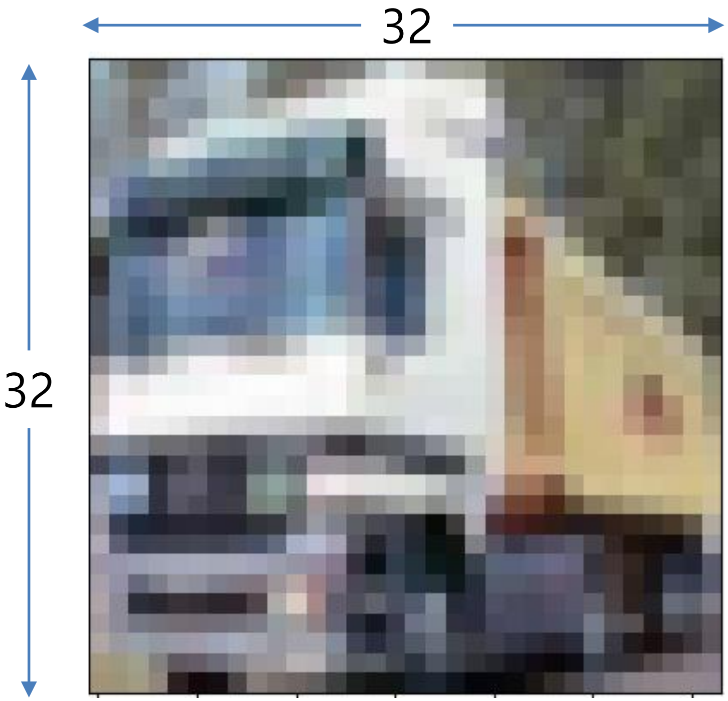
LeNet-5

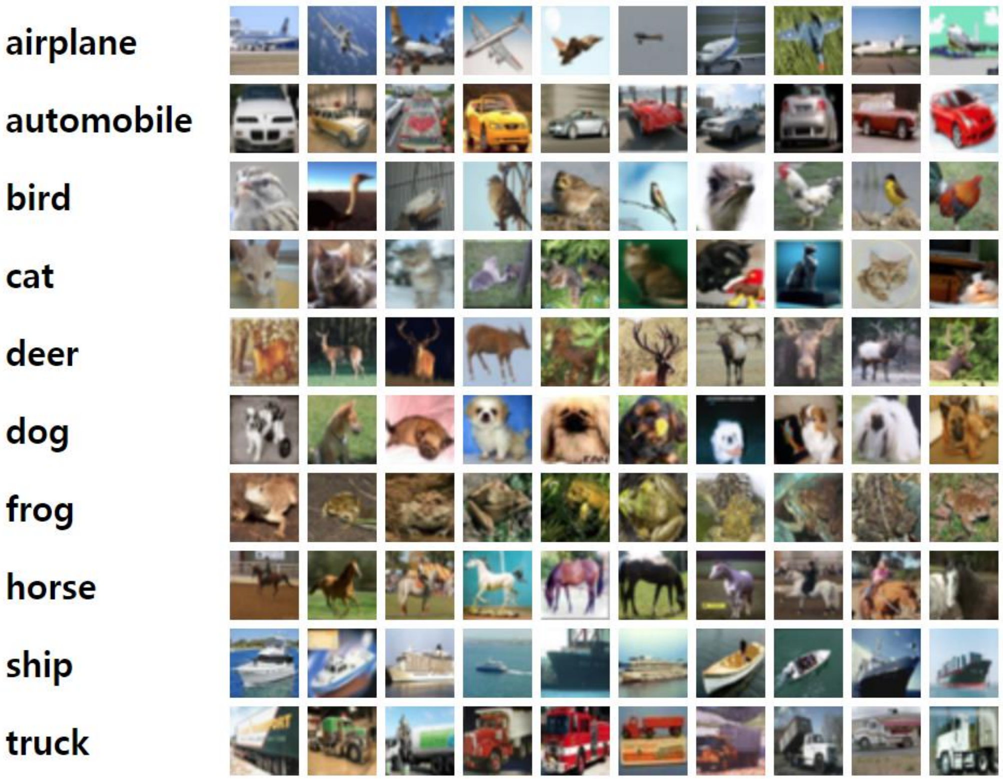
- LeNet [LeCun et al., 1998]



Conv filters were 5x5, applied at stride 1
Subsampling (Pooling) layers were 2x2 applied at stride 2
i.e. architecture is [CONV-POOL-CONV-POOL-CONV-FC]

CIFAR-10 data





```
x = tf.placeholder(tf.float32, shape=[None, 32, 32, 3])
y = tf.placeholder(tf.float32, shape=[None, 10])
```

CIFAR-10 data

```
import numpy as np
import matplotlib.pyplot as plt

from tensorflow.keras.datasets.cifar10 import load_data

(x_train, y_train), (x_test, y_test) = load_data()

print(np.shape(x_train))
print(np.shape(y_train))
print(np.shape(x_test))
print(np.shape(y_test))

# airplane, automobile, bird, cat, deer, dog, frog, horse,
ship, truck
print(y_train[1])
plt.imshow(x_train[1])
plt.show()
```

Image Input

```
X = tf.placeholder(tf.float32, [None, 32, 32, 3], name="X")
Y = tf.placeholder(tf.float32, [None, 10], name="Y")
keep_prob = tf.placeholder(tf.float32, name="keep_prob")
```

Convolutional & pooling layers

```
W1 = tf.Variable(tf.random_normal([3, 3, 3, 32], stddev=0.01))
L1 = tf.nn.conv2d(X, W1, strides=[1, 1, 1, 1], padding='SAME')
L1 = tf.nn.relu(L1)
L1 = tf.nn.max_pool(L1, ksize=[1, 2, 2, 1], strides=[1, 2, 2, 1],
padding='SAME')
L1 = tf.nn.dropout(L1, keep_prob=keep_prob)
.
.
.
W3 = tf.Variable(tf.random_normal([3, 3, 64, 128], stddev=0.01))
L3 = tf.nn.conv2d(L2, W3, strides=[1, 1, 1, 1], padding='SAME')
L3 = tf.nn.relu(L3)
L3 = tf.nn.max_pool(L3, ksize=[1, 2, 2, 1], strides=[1, 2, 2, 1],
padding='SAME')
L3 = tf.nn.dropout(L3, keep_prob=keep_prob)
```

```
L3_flat = tf.reshape(L3, [-1, 4 * 4 * 128])
```

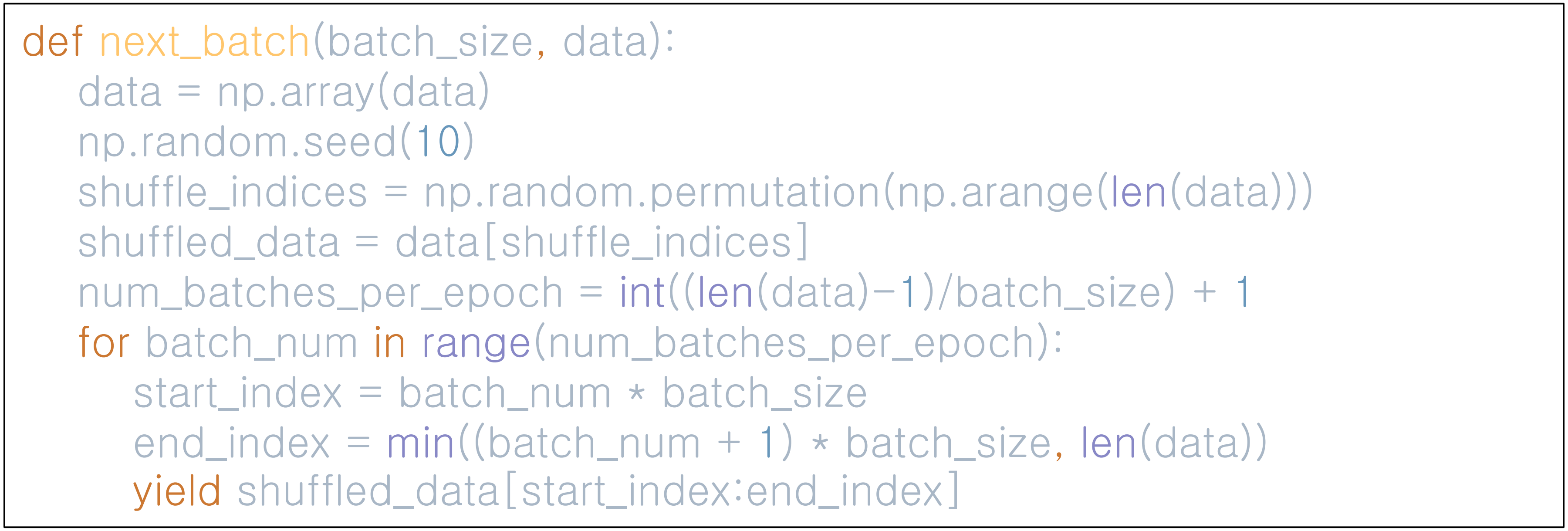
Fully connected layers

```
W4 = tf.get_variable("W4", shape=[4 * 4 * 128, 128],
initializer=tf.initializers.he_normal())
b4 = tf.Variable(tf.random_normal([128]))
FC1 = tf.nn.relu(tf.nn.xw_plus_b(L3_flat, W4, b4))
FC1 = tf.nn.dropout(FC1, keep_prob=keep_prob)

W5 = tf.get_variable("W5", shape=[128, 64],
initializer=tf.initializers.he_normal())
b5 = tf.Variable(tf.random_normal([64]))
FC2 = tf.nn.relu(tf.nn.xw_plus_b(FC1, W5, b5))
FC2 = tf.nn.dropout(FC2, keep_prob=keep_prob)

W6 = tf.get_variable("W6", shape=[64, 10],
initializer=tf.initializers.he_normal())
b6 = tf.Variable(tf.random_normal([10]))
hypothesis = tf.nn.xw_plus_b(FC2, W6, b6, name="hypothesis")
```

preprocessing

```
max = 0
early_stopped = 0
(x_train_val, y_train_val), (x_test, y_test) = load_data()
shuffle_indices = np.random.permutation(np.arange(len(y_train_val)))
shuffled_x = np.asarray(x_train_val[shuffle_indices])
shuffled_y = y_train_val[shuffle_indices]
dev_sample_index = −1 * int(0.1 * float(len(y_train_val)))
x_train, x_val = shuffled_x[:dev_sample_index],
shuffled_x[dev_sample_index:]
y_train, y_val = shuffled_y[:dev_sample_index],
shuffled_y[dev_sample_index:]
x_test = np.asarray(x_test)

y_train_one_hot = np.eye(10)[y_train]
y_train_one_hot = np.squeeze(y_train_one_hot, axis=1)
y_test_one_hot = np.eye(10)[y_test]
y_test_one_hot = np.squeeze(y_test_one_hot, axis=1)
y_val_one_hot = np.eye(10)[y_val]
y_val_one_hot = np.squeeze(y_val_one_hot, axis=1)
```

Next batch



```
def next_batch(batch_size, data):
    data = np.array(data)
    np.random.seed(10)
    shuffle_indices = np.random.permutation(np.arange(len(data)))
    shuffled_data = data[shuffle_indices]
    num_batches_per_epoch = int((len(data)-1)/batch_size) + 1
    for batch_num in range(num_batches_per_epoch):
        start_index = batch_num * batch_size
        end_index = min((batch_num + 1) * batch_size, len(data))
        yield shuffled_data[start_index:end_index]
```

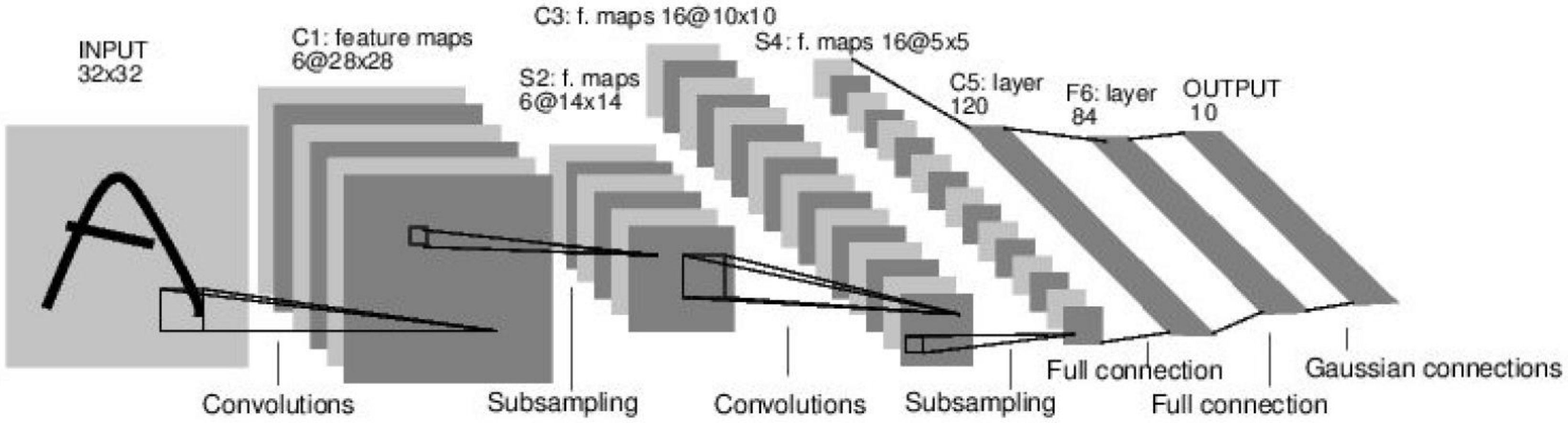
Training

```
for epoch in range(training_epochs):
    avg_cost = 0
    total_batch = int(len(y_train) / batch_size)
    batches = next_batch(batch_size, list(zip(x_train, y_train_one_hot)))
.
.
.
```

Homework

1. Implement Lenet-5!

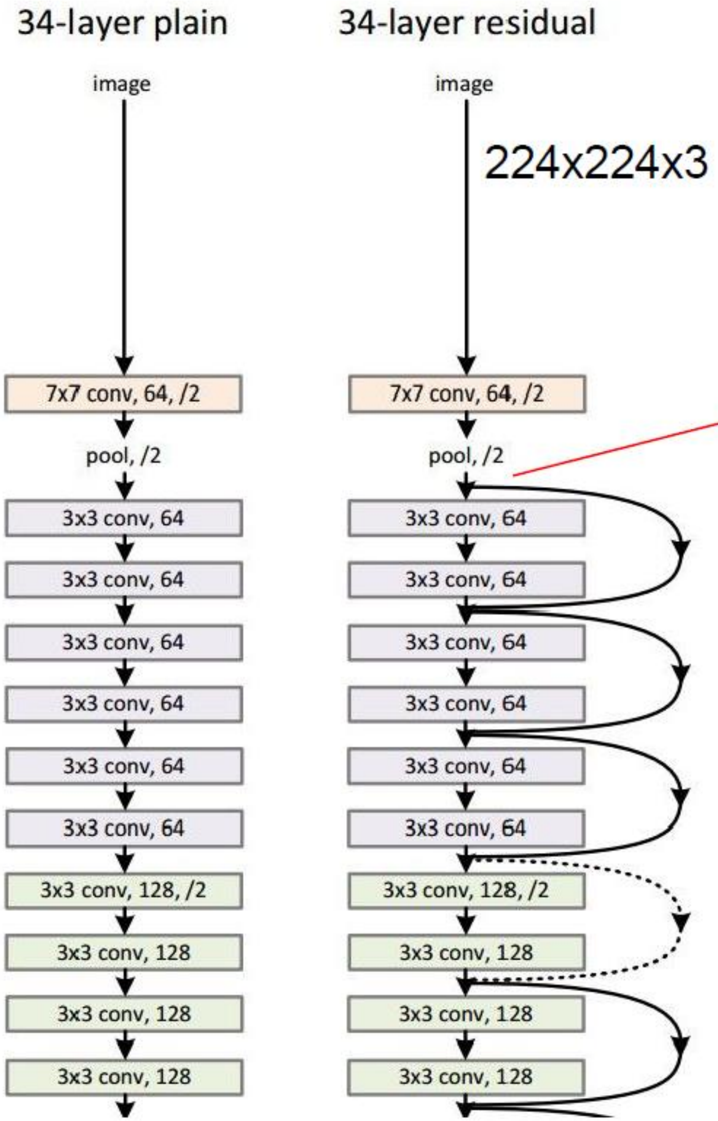
2. Change dropout ratio, initialization (He)



Conv filters were 5x5, applied at stride 1
Subsampling (Pooling) layers were 2x2 applied at stride 2
i.e. architecture is [CONV-POOL-CONV-POOL-CONV-FC]

Next class (1교시)

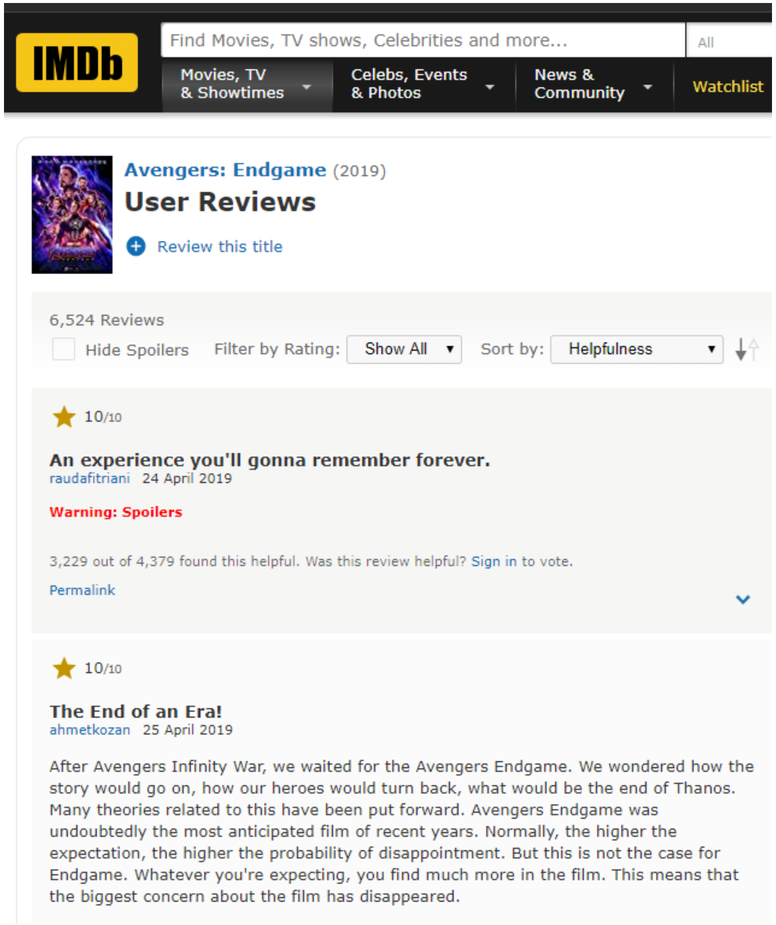
- ResNet [He et al., 2015]



spatial dimension
only 56x56!

Next class (2교시)

- CNN for sentence classification [Kim et al., 2014]



Avengers: Endgame (2019)

User Reviews

Review this title

6,524 Reviews

□ Hide Spoilers Filter by Rating: Show All ▾ Sort by: Helpfulness ▾

★ 10/10

An experience you'll gonna remember forever.

raudafitriani 24 April 2019

Warning: Spoilers

3,229 out of 4,379 found this helpful. Was this review helpful? Sign in to vote.

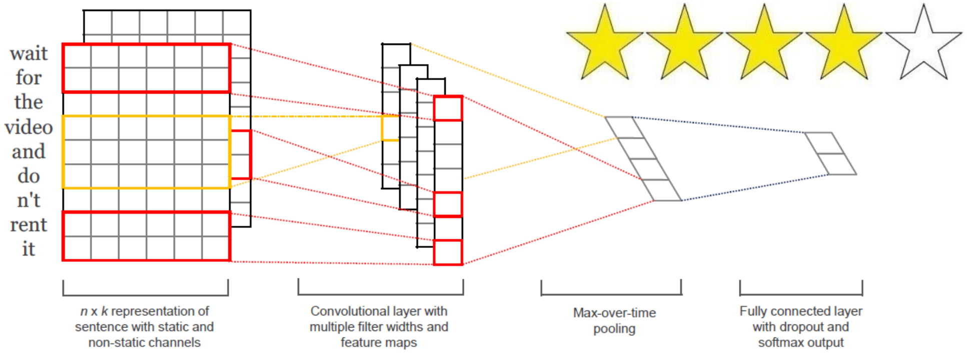
Permalink

★ 10/10

The End of an Era!

ahmetkozan 25 April 2019

After Avengers Infinity War, we waited for the Avengers Endgame. We wondered how the story would go on, how our heroes would turn back, what would be the end of Thanos. Many theories related to this have been put forward. Avengers Endgame was undoubtedly the most anticipated film of recent years. Normally, the higher the expectation, the higher the probability of disappointment. But this is not the case for Endgame. Whatever you're expecting, you find much more in the film. This means that the biggest concern about the film has disappeared.



wait
for
the
video
and
do
n't
rent
it
n x k representation of sentence with static and non-static channels
Convolutional layer with multiple filter widths and feature maps
Max-over-time pooling
Fully connected layer with dropout and softmax output

Q&A