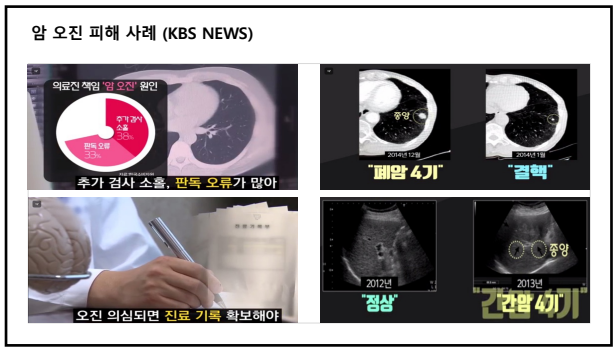

Deep Learning
INDUSTRY
4.0
CLOUD
BIG DATA
VR

1


암 오진 피해 사례 (KBS NEWS)
의료진 책임 '암 오진' 원인
추가 검사 소홀, 판독 오류가 많아
"폐암 4기"
"결핵"
오진 의심되면 진료 기록 확보해야
"정상"
"간암 4기"

2


빅데이터 생각해 보기
인공지능 의사 '왓슨' 이야기
실제로 왓슨을 써보니 ...
왓슨의 시작 만으로 ...

3

빅데이터 생각해 보기

기상 빅데이터를 다루는 왓슨

IBM, 세계 최대 날씨 정보회사 웨더 컴퍼니 인수(2015),

실제로 왓슨을 써보니 ...

2016년 왓슨은
"대기권의 데이터를 받아 날씨를 예측하게 했고 기류에 대해 더 잘 알게 되어, 항공기가 경험하는 난류를 2분의 1로 줄일 수 있었다"

왓슨의 시작 만으로 ...

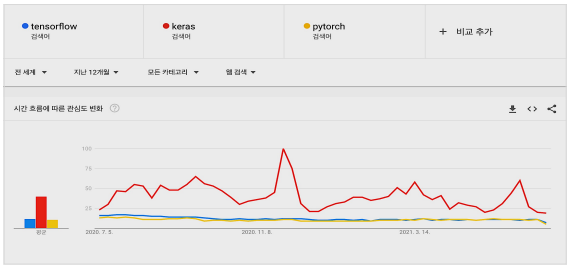
IBM 왓슨의 딥러닝(심층학습) 기술과 웨더컴퍼니가 대기권에 배치한 수십억개의 센서를 통해 15분마다 대기권 정보를 받고 분석하고 있다. 이를 통해 날씨는 물론 미래 기후 변화까지도 예측하고 있다.

왓슨을 통해 "난류의 존재를 파악하고 항공기 조종사가 난류층의 위 혹은 아래로 운항할 수 있도록 하거나, 태풍 경로를 예측하고 이를 스마트폰으로 통보해 즉시 조치를 취하는 것"이가능하다.

- 데이비드 케니 IBM 왓슨 총괄 사장

4

딥러닝 라이브러리



5

Tensorflow

- 2015년 11월 출시
- 구글 브레인 팀에서 심층 신경망을 위해 개발한 강력한 오픈소스 소프트웨어 라이브러리
- 자동 미분, CPU/GPU 옵션 지원, pre-trained model 등 지원
- 파이썬, C++, 자바, R, Go등 주요 언어로 작업 가능
- 케라스가 텐서플로에 통합
  - 케라스는 마이크로소프트 CNTK, 아마존 mxnet, 티아노 등 여러 딥러닝 엔진에 통합될 수 있다.
- 쉬운 모델 배치 및 생산, 시각화 기능, 커뮤니티 활성화 등의 장점
- 2.0 버전부터 케라스 단순 코딩 사용, 직관적 프로그래밍 적용

6

pip를 사용하여 TensorFlow 설치

- **TensorFlow 2 패키지 사용 가능**
- tensorflow - CPU와 GPU 지원이 포함된 안정적인 최신 출시*(Ubuntu 및 Windows)*
- tf-nightly - 미리보기 빌드*(불안정)*. Ubuntu 및 Windows에는 GPU 지원이 포함되어 있습니다.

- **이전 버전의 TensorFlow**
- TensorFlow 1.x의 경우 CPU와 GPU 패키지는 다음과 같이 구분됩니다.
- tensorflow==1.15 - CPU 전용 출시
- tensorflow-gpu==1.15 - GPU 지원이 포함된 출시*(Ubuntu 및 Windows)*

- **시스템 요구사항**
- Python 3.5~3.8
  - Python 3.8 지원에는 TensorFlow 2.2 이상이 필요합니다.
- pip 19.0 이상(manylinux2010 지원 필요)
- Ubuntu 16.04 이상(64비트)
- macOS 10.12.6(Sierra) 이상(64비트)*(GPU 지원 없음)*
- Windows 7 이상(64비트)

  Visual Studio 2015, 2017 및 2019용 Microsoft Visual C++ 재배포 가능 패키지
- Raspbian 9.0 이상
- GPU 지원에는 CUDA® 지원 카드 필요*(Ubuntu 및 Windows)*

7

신경망을 위한 데이터 표현

스칼라 (0D 텐서)

- 넘파이에서는 float32나 float64 타입의 숫자가 스칼라 텐서

- *import numpy as np*
- *x = np.array(12)*
- *x*

- *x.ndim*

8

신경망을 위한 데이터 표현

벡터 (1D 텐서)

- *x = np.array([12,3,6,14,7])*
- *x*

- *x.ndim*

행렬 (2D 텐서)

- *x = np.array([[5, 78, 2, 34, 0],*
- *[6, 79, 3, 35, 1],*
- *[7, 80, 4, 36, 2]])*
- *x*

- *x.ndim*

9

신경망을 위한 데이터 표현

3D 텐서

```
x = np.array([[[5, 78, 2, 34, 0],
               [6, 79, 3, 35, 1],
               [7, 80, 4, 36, 2]],
              [[5, 78, 2, 34, 0],
               [6, 79, 3, 35, 1],
               [7, 80, 4, 36, 2]],
              [[5, 78, 2, 34, 0],
               [6, 79, 3, 35, 1],
               [7, 80, 4, 36, 2]]])
x.ndim
```

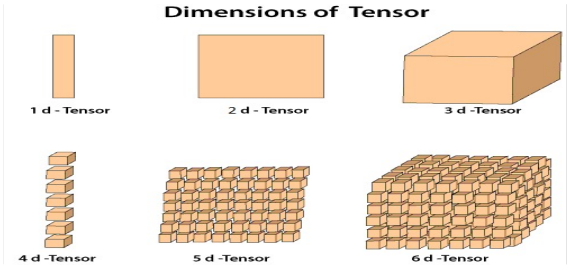
고차원 텐서

```
x = np.array([
[[[5, 78, 2, 34, 0], [6, 79, 3, 35, 1], [7, 80, 4, 36, 2]],
[[5, 78, 2, 34, 0], [6, 79, 3, 35, 1], [7, 80, 4, 36, 2]],
[[5, 78, 2, 34, 0], [6, 79, 3, 35, 1], [7, 80, 4, 36, 2]]],
[[[5, 78, 2, 34, 0], [6, 79, 3, 35, 1], [7, 80, 4, 36, 2]],
[[5, 78, 2, 34, 0], [6, 79, 3, 35, 1], [7, 80, 4, 36, 2]],
[[5, 78, 2, 34, 0], [6, 79, 3, 35, 1], [7, 80, 4, 36, 2]]],
[[[5, 78, 2, 34, 0], [6, 79, 3, 35, 1], [7, 80, 4, 36, 2]],
[[5, 78, 2, 34, 0], [6, 79, 3, 35, 1], [7, 80, 4, 36, 2]],
[[5, 78, 2, 34, 0], [6, 79, 3, 35, 1], [7, 80, 4, 36, 2]]]
])

x.ndim
```

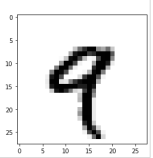
10

Tensor



11

데이터 타입 확인



12

넘파이로 텐서 조작

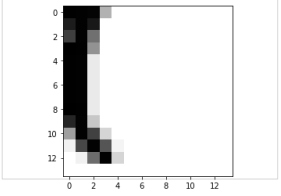
슬라이싱

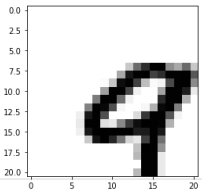
- *문제: 11 번째에서 100 번째까지 숫자를 선택하여 (90, 28, 28 크기의 배열 생성)*

13

넘파이로 텐서 조작

 # 이미지의 오른쪽 아래 14x14 픽셀 선택





14

배치 데이터

- *# MNIST 숫자 데이터에서 크기가 128인 배치 하나는 다음과 같다.*
- *# 그 다음 배치는 다음과 같다.*
- *# 그리고 n 번째 배치는 다음과 같다.*
- *# batch axis or batch dimension*

15

텐서의 실제 사례

1) 벡터 데이터
- : (**samples**, features) 크기의 2D 텐서

2) 시계열 데이터 또는 시퀀스 (sequence) 데이터
- : (**samples**, timesteps, features) 크기의 3D 텐서

3) 이미지
- : (**samples**, height, width, channels) 또는 (**samples**, channels, height, width) 크기의 4D 텐서

4) 동영상
- : (**samples**, frames, height, width, channels) 또는 (**samples**, frames, channels, height, width) 크기의 5D 텐서

16

1) 벡터 데이터

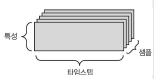
- **사람의 나이, 우편 번호, 소득으로 구성된 인구 통계 데이터**
- 각 사람은 3개의 값을 가진 벡터로 구성되고 10만 명이 포함된 전체 데이터 셋은 (100000, 3)

- **(공통 단어 2만 개로 만든 사전에서) 각 단어가 등장한 횟수로 표현된 텍스트 문서 데이터**
- 각 문서는 2만 개의 원소를 가진 벡터로 인코딩.
- 500개의 문서로 이루어진 전체 데이터셋은 (500, 20000)

17

2) 시계열 데이터 또는 시퀀스 데이터

- **주식 가격 데이터**
- 1분마다 현재 주식 가격, 지난 1분 동안에 최고 가격과 최소 가격을 저장
- 1분마다 데이터는 3D 벡터로 인코딩, 하루 동안의 거래는 (390, 3). * 하루 거래 시간: 390분
- 250일치 데이터는 (250, 390, 3)
- 1일치 데이터가 하나의 샘플



- **트윗 데이터**
- 각 트윗은 128개의 알파벳으로 구성된 280개 문자 시퀀스
- 각 문자가 128개의 크기인 이진 벡터로 인코딩 * 해당 문자의 인덱스만 1이고 나머지는 모두 0
- 각 트윗은 (280, 128)
- 100만개의 트윗은 (1000000, 280, 128)

18

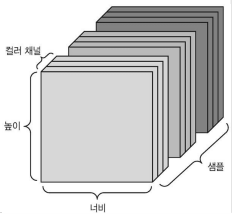
3) 이미지 데이터

- 256 x 256 크기 흑백 이미지에 대한 128개 배치 (128, 256, 256, 1)
- 256 x 256 크기 컬러 이미지에 대한 128개 배치 (128, 256, 256, 3)

- ***Tensorflow***
- 채널 마지막 방식 channel-last
- (samples, height, width, color_depth)
- (128, 256, 256, 1)
- (128, 256, 256, 3)

- ***Pytorch, Theano***
- 채널 우선 방식 channel-first
- (samples, color_depth, height, width)
- (128, 1, 256, 256)
- (128, 3, 256, 256)

- ***Keras:*** channel_last or channel_first



19

4) 비디오 데이터

- 이미지 프레임 (height, width, color_depth)
- 프레임의 연속 (frames, height, width, color_depth)
- 비디오 배치 (samples, frames, height, width, color_depth)

- 60초 짜리 144 x 256 유튜브 비디오 클립을 초당 4 프레임 샘플링하면 240 프레임
- (60, 4, 144, 256, 3)
- 총 106,168,320 개의 값

- Float32: 106,168,320 x 32bit = 405MB

- MPEG, 4K 등 동영상은 압축변환 사용

20

Tensorflow 1.0 vs 2.0

```
Import tensorflow.compat.v1 as tf

In_a = tf.placeholder(dtype=tf.float32, shape=(2))

Def model(x):
  with tf.variable_scope('matmul'):
    w = tf.get_variable('w', initializer=tf.ones(shape=(2,2)))
    b = tf.get_variable('b', initializer=tf.zeros(shape=(2)))
    return x * w + b

Out_a = model(in_a)

With tf.Session() as sess:
  sess.run(tf.global_variables_initializer())
  outs = sess.run([out_a],
                  feed_dict = {in_a:[1,0]}
```

```
Import tensorflow as tf
W = tf.variable(tf.ones(shape=(2,2)), name='W')
B = tf.variable(tf.zeros(shape=(2)), name='b')

@tf.function
Def model(x):
  return w * x + b
Out_a = model([1,0])

Print(out_a)
```

21

주택 가격 예측 회귀 문제 with tensorflow

22

Tensorflow

tensorflow.estimator.LinearRegressor

23

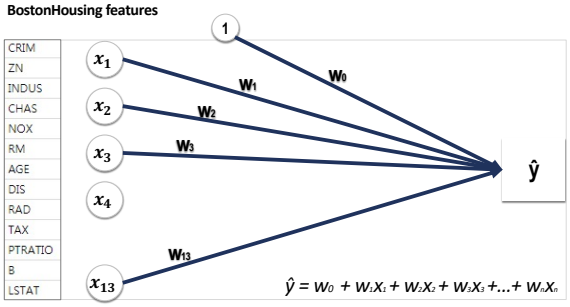
BostonHousing features

BostonHousing 데이터 설명 총 13개의 변수

[01]	CRIM	자치시(town) 별 1인당 범죄율
[02]	ZN	25,000 평방피트를 초과하는 거주지역의 비율
[03]	INDUS	비소매상업지역이 점유하고 있는 토지의 비율
[04]	CHAS	찰스강에 대한 더미변수(강의 경계에 위치한 경우는 1, 아니면 0)
[05]	NOX	10ppm 당 농축 일산화질소
[06]	RM	주택 1가구당 평균 방의 개수
[07]	AGE	1940년 이전에 건축된 소유주택의 비율
[08]	DIS	5개의 보스턴 직업센터까지의 접근성 지수
[09]	RAD	방사형 도로까지의 접근성 지수
[10]	TAX	10,000 달러 당 재산세율
[11]	PTRATIO	자치시(town)별 학생/교사 비율
[12]	B	1000(Bk-0.63)^2, 여기서 Bk는 자치시별 흑인의 비율을 말함.
[13]	LSTAT	모집단의 하위계층의 비율(%)
[14]	MEDV	본인 소유의 주택가격(중앙값) (단위: $1,000)

24

BostonHousing features



25

BostonHousing features

- *# tensorflow 모델 구성*

26

데이터 준비

- *# 보스턴 데이터 준비 및 전처리 진행*

27

pandas

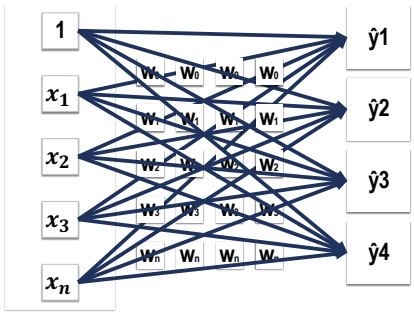
- *# 판다스로 데이터 확인*

28

모델 학습

- *# 모델 컴파일*
- *# 모델 학습*
- *# 모델 평가*

29



30

딥러닝

- 일반선형회귀
- 로지스틱 회귀 (Logistic Regression)
- 신경망 (Neural Network)
- CNN
- RNN

31