기계학습과 딥러닝 그리고 Pytorch

2.11.2019

Presenter: **KyungTae Lim**

Contents

기계학습의 정의

- 기계 학습이란?
  - 현대적 정의

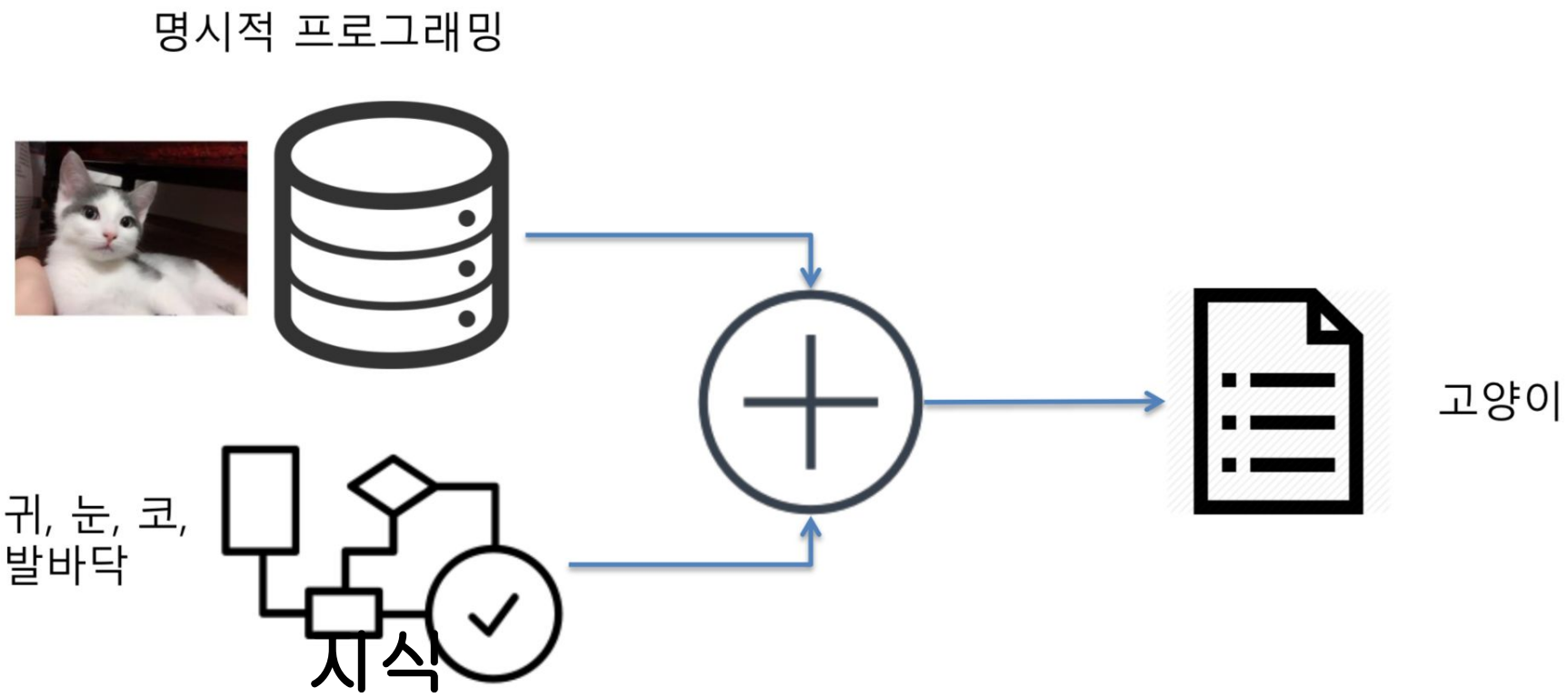
"A computer program is said to learn from experience *E* with respect to some class of tasks *T* and performance measure *P*, if its performance at tasks in *T*, as measured by *P*, improves with experience *E*.
어떤 컴퓨터 프로그램이 *T*라는 작업을 수행한다. 이 프로그램의 성능을 *P*라는 척도로 평가했을 때 경험 *E*를 통해 성능이 개선된다면 이 프로그램은 학습을 한다고 말할 수 있다[Mitchell1997(2쪽)]."

"Programming computers to optimize a performance criterion using example data or past experience
사례 데이터, 즉 과거 경험을 이용하여 성능 기준을 최적화하도록 프로그래밍하는 작업[Alpaydin2010]"

"Computational methods using experience to improve performance or to make accurate predictions
성능을 개선하거나 정확하게 예측하기 위해 경험을 이용하는 계산학 방법들[Mohri2012]"

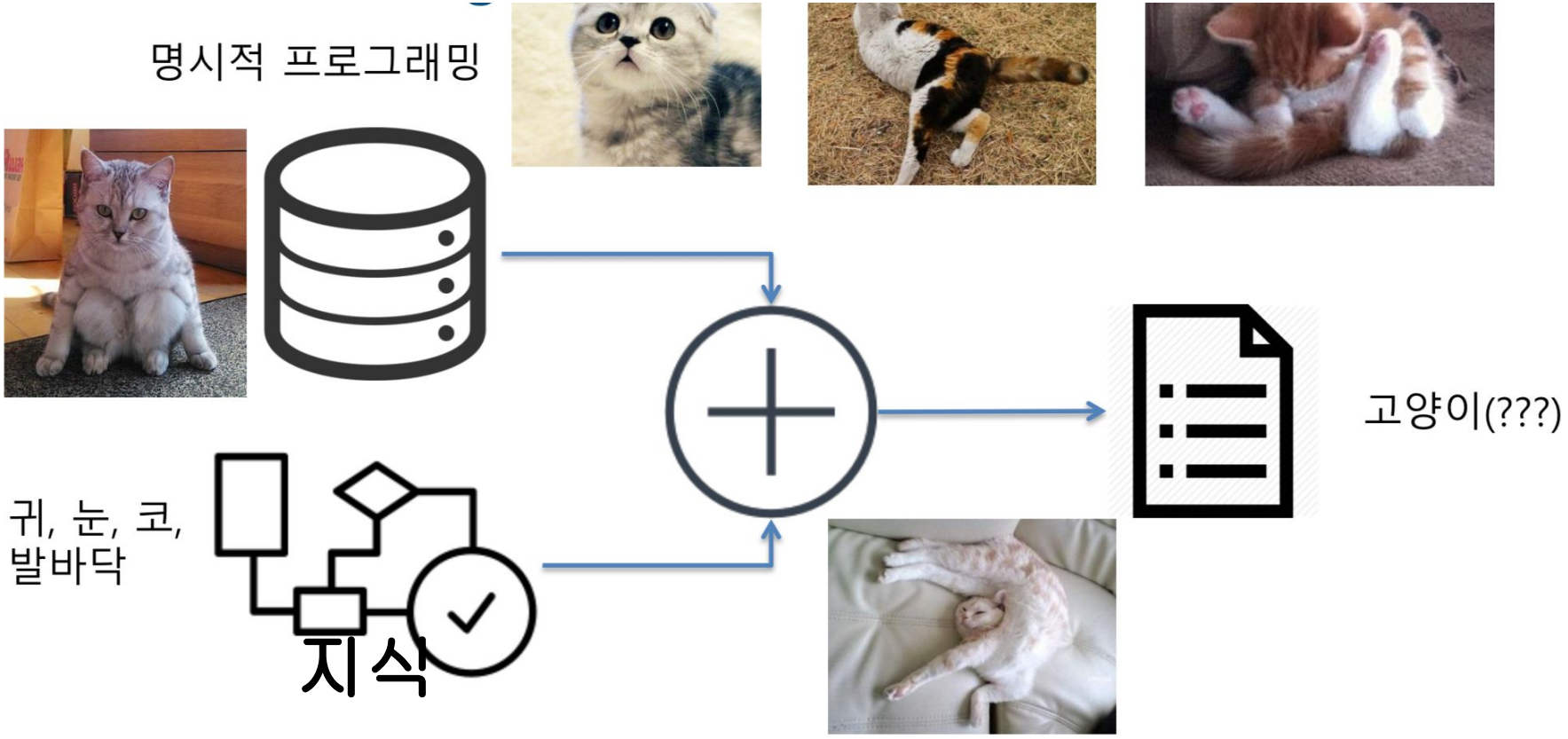
출처: 기계학습 (오원철

지식 기반의 프로그램



출처: PyTorch로 시작하는 딥러닝 입문 (최근

지식 기반의 프로그램



출처: PyTorch로 시작하는 딥러닝 입문 (최근

지식기반 방식에서 기계 학습으로의 대전환

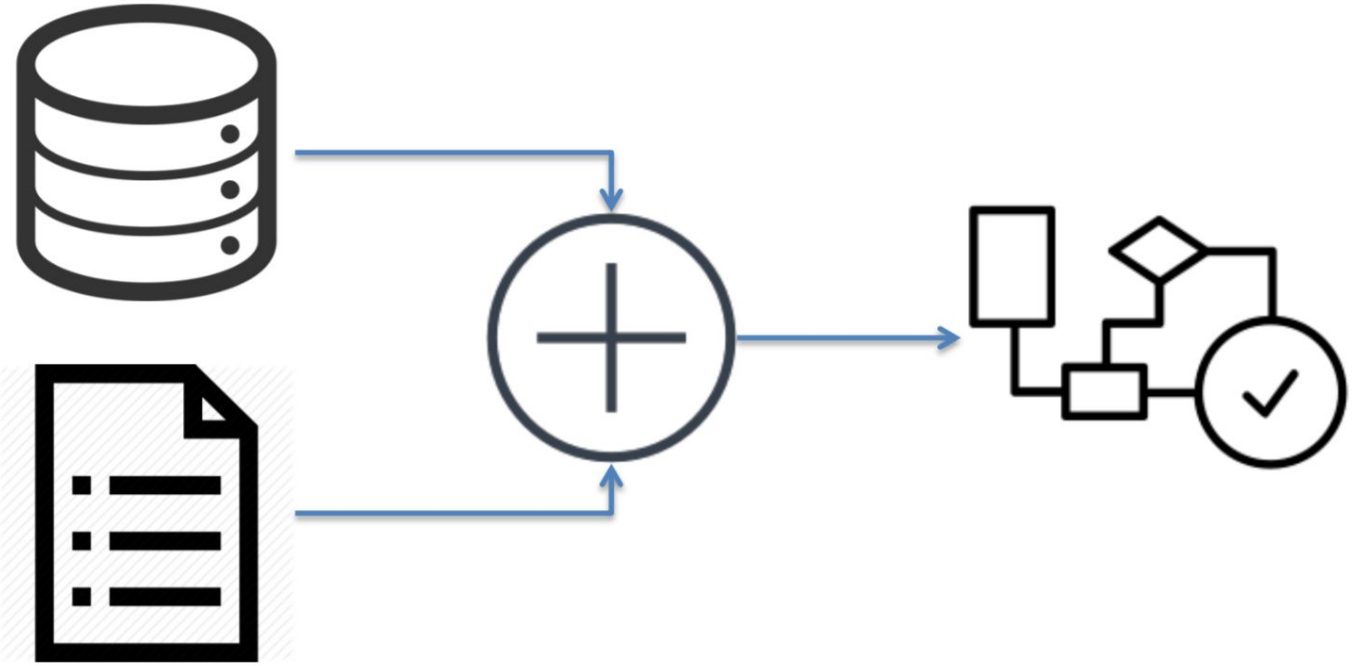
- 큰 깨달음
  - 지식기반의 한계
  - 단추를 “가운데 구멍이 몇 개 있는 물체”라고 규정하면 많은 오류 발생

그림 1-2 인식 시스템이 대처해야 하는 심한 변화 양상(8과 단추라는 패턴을 어떻게 기술할 것인가?)

  - 사람은 변화가 심한 장면을 아주 쉽게 인식하지만, 왜 그렇게 인식하는지 서술하지는 못함

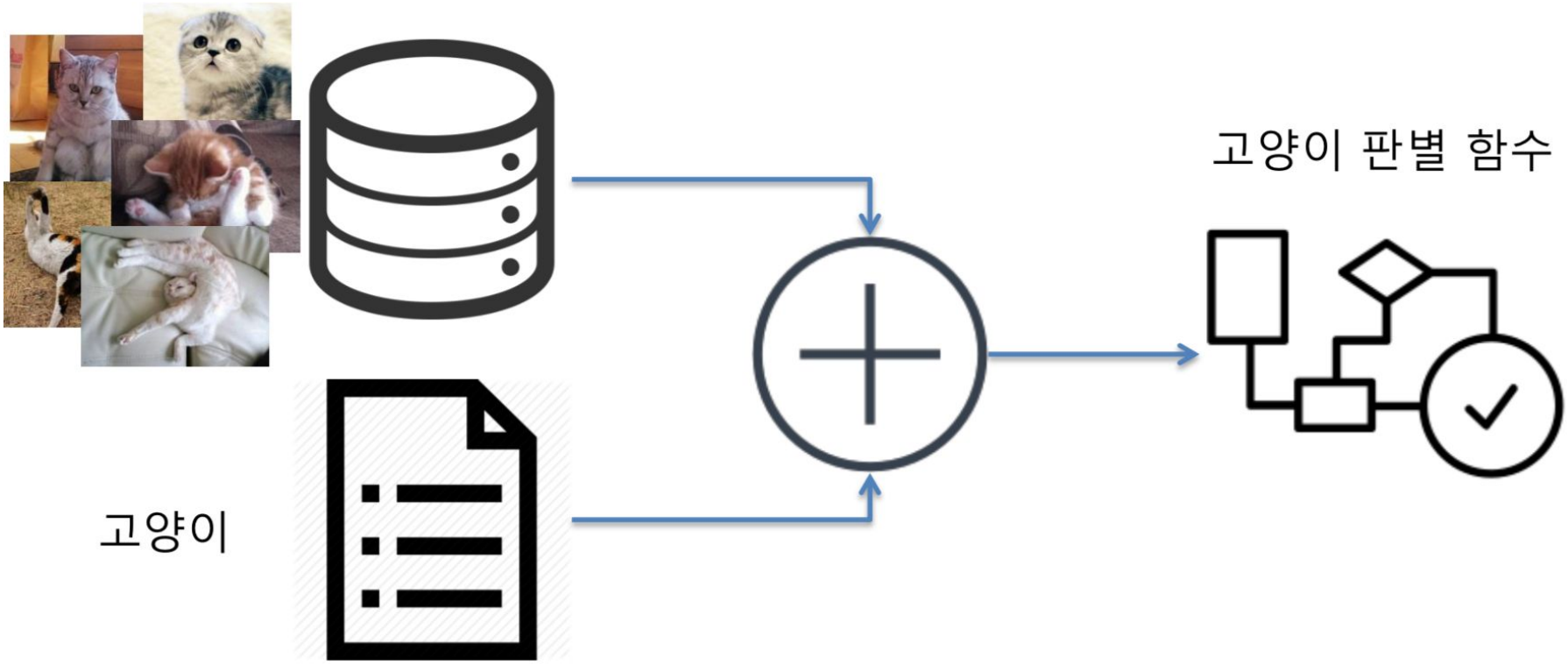
출처: 기계학습 (오원철

기계학습



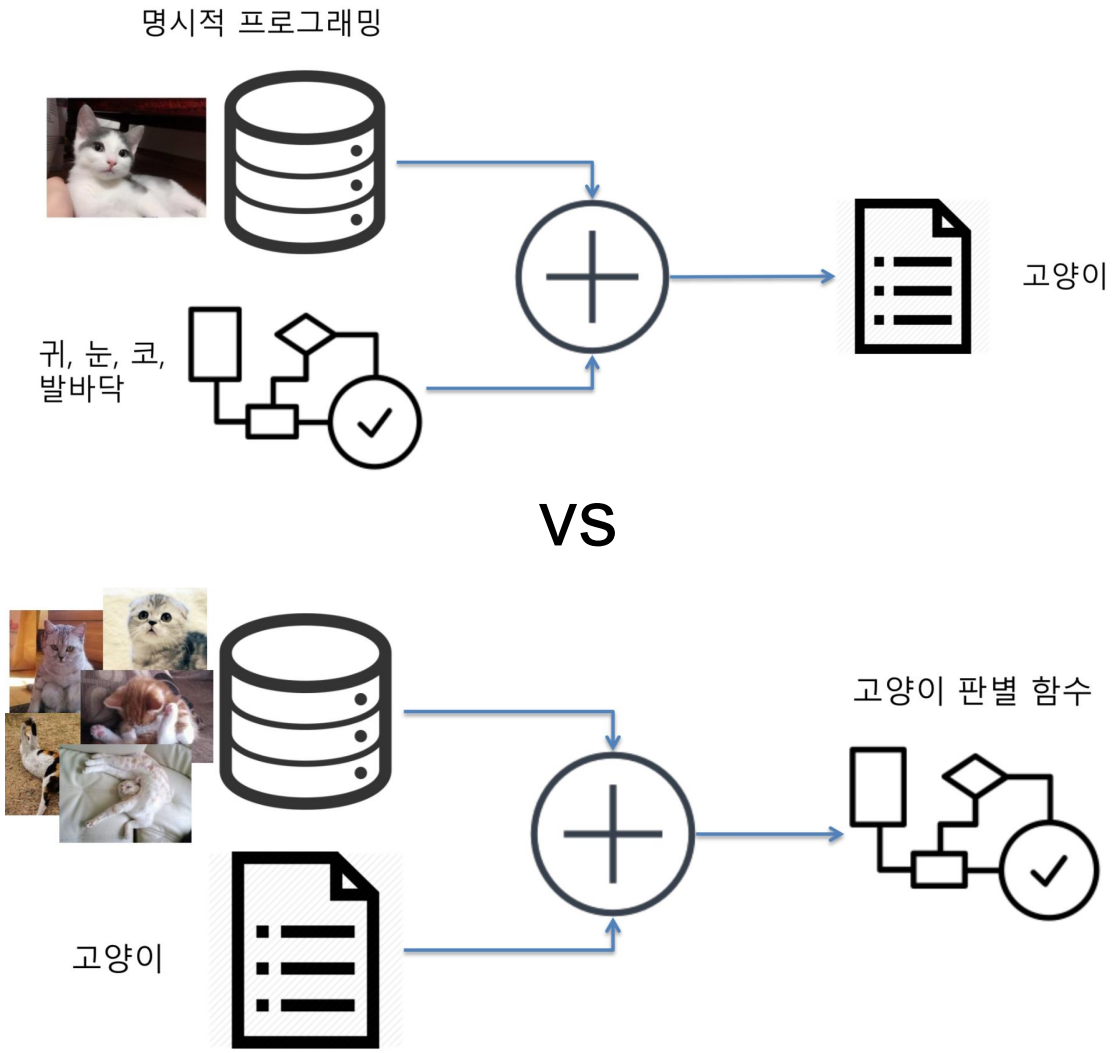
출처: PyTorch로 시작하는 딥러닝 입문 (최근

기계학습



출처: PyTorch로 시작하는 딥러닝 입문 (최근

지식 기반의 프로그램 vs 기계학습



출처: PyTorch로 시작하는 딥러닝 입문 (최근

사람의학습 vs 기계학습

표 1-1 사람의 학습과 기계 학습의 비교

기준	사람의 학습	기계 학습
학습 과정	능동적	수동적
데이터 형식	자연에 존재하는 그대로	일정한 형식에 맞추어 사람이 준비함
동시에 학습 가능한 과업 수	자연스럽게 여러 과업을 학습	하나의 과업만 가능
학습 원리에 대한 지식	매우 제한적으로 알려져 있음	모든 과정이 밝혀져 있음
수학 의존도	매우 낮음	매우 높음
성능 평가	경우에 따라 객관적이거나 주관적	객관적(수치로 평가, 예를 들어 정확률 99.8%)
역사	수백만 년	60년 가량

출처: 기계학습 (오원철

기계학습의 유형

- 지도 학습
  - 특징 벡터 $\mathbb{X}$와 목푯값 $\mathbb{Y}$가 모두 주어진 상황
  - 회귀와 분류 문제로 구분
- 비지도 학습
  - 특징 벡터 $\mathbb{X}$는 주어지는데 목푯값 $\mathbb{Y}$가 주어지지 않는 상황
  - 군집화 과업 (고객 성향에 따른 맞춤 홍보 응용 등)
  - 밀도 추정, 특징 공간 변환 과업

출처: 기계학습 (오원철

기계학습의 유형

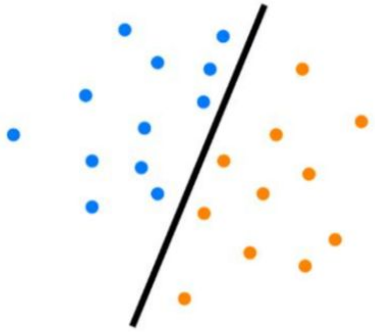
- 강화 학습
  - 목푯값이 주어지는데, 지도 학습과 다른 형태임
  - 예) 바둑
    - 수를 두는 행위가 샘플인데, 게임이 끝나면 목푯값 하나가 부여됨
      - 이기면 1, 패하면 -1을 부여
    - 게임을 구성한 샘플들 각각에 목푯값을 나누어 주어야 함
  - 9장의 주제
- 준지도 학습
  - 일부는 $\mathbb{X}$와 $\mathbb{Y}$를 모두 가지지만, 나머지는 $\mathbb{X}$만 가진 상황
  - 인터넷 덕분으로 $\mathbb{X}$의 수집은 쉽지만, $\mathbb{Y}$는 수작업이 필요하여 최근 중요성 부각

출처: 기계학습 (오원철

기계학습의 유형

Supervised

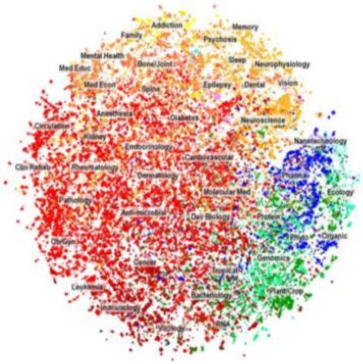
Learning
known
patterns



(위의 고양이판별 예시에 해당)

Unsupervised

Learning
unknown
patterns



Reinforcement

Generating data

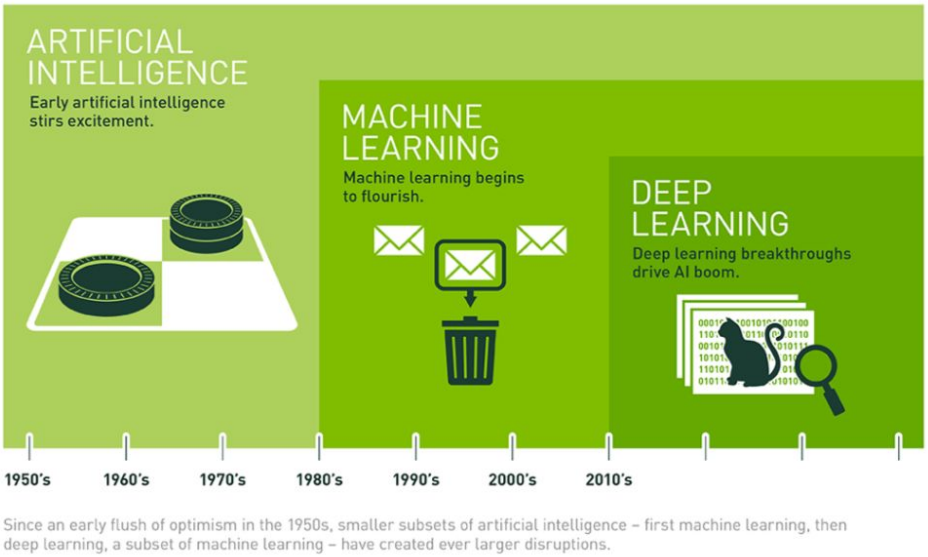
Learning patterns

(출처: http://adgefficiency.com)

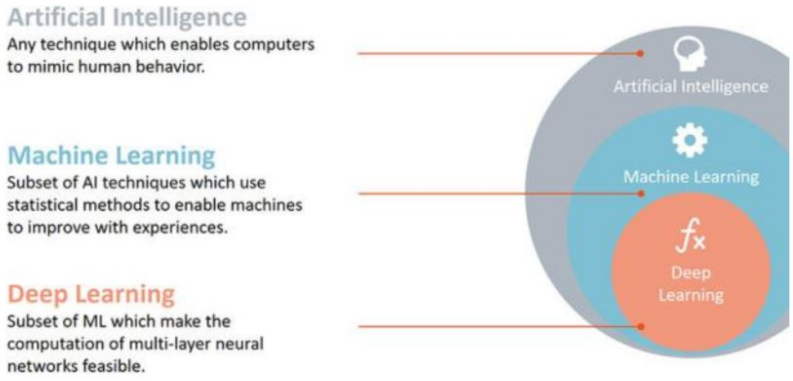
출처: PyTorch로 시작하는 딥러닝 입문 (최근

그럼 딥러닝은 뭔가요?

- 기계학습 기술의 일부분 입니다.



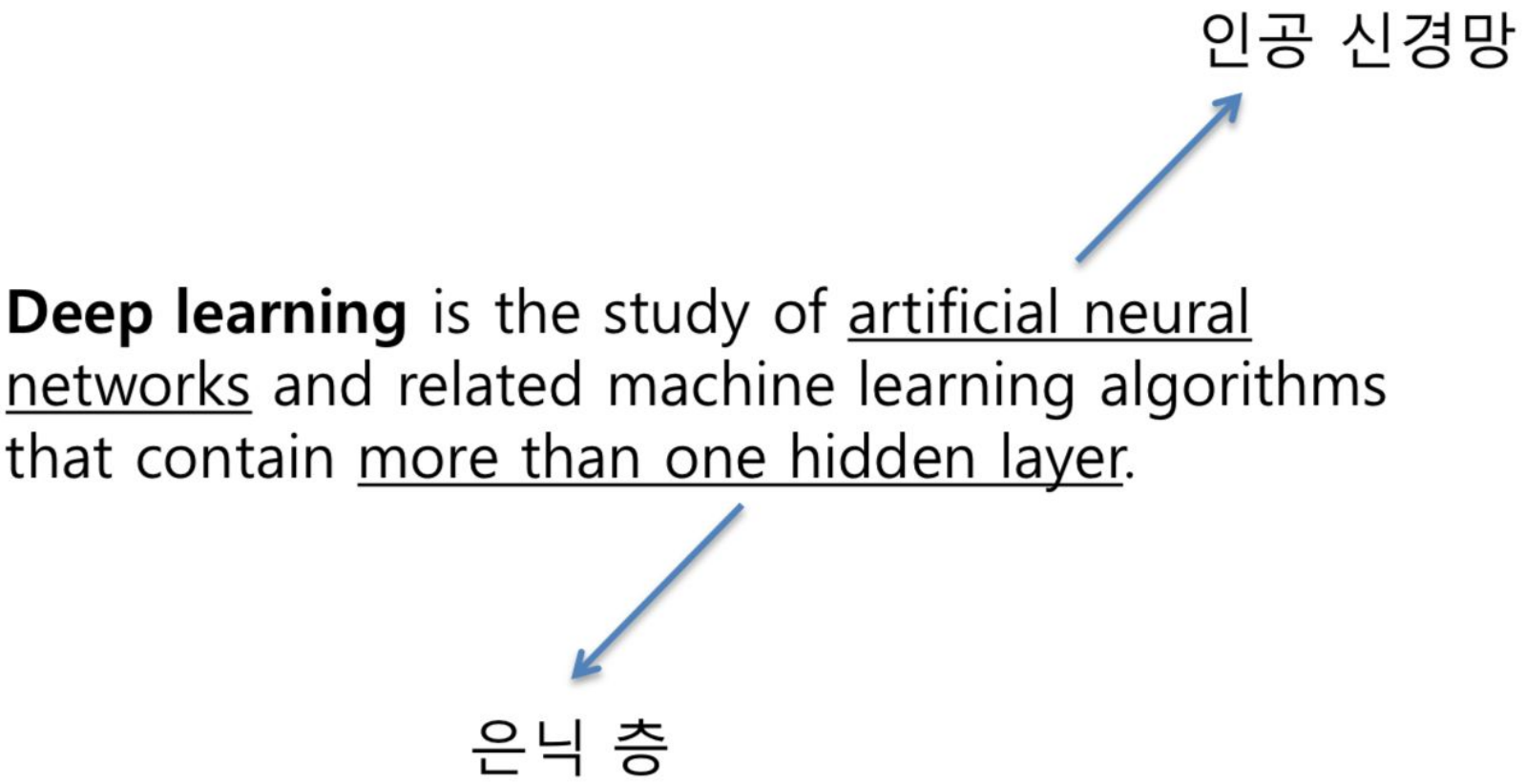
(출처: 엔비디아 블로그)



(출처: rapid miner)

출처: PyTorch로 시작하는 딥러닝 입문 (최근

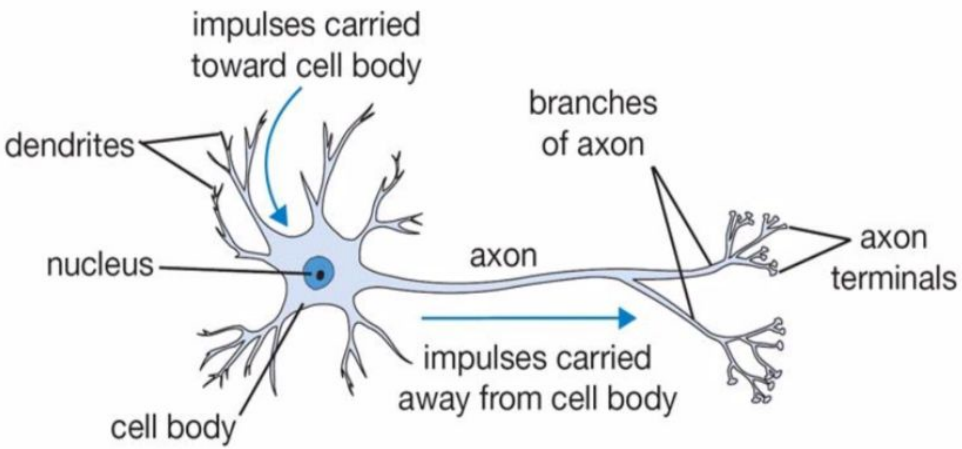
딥러닝

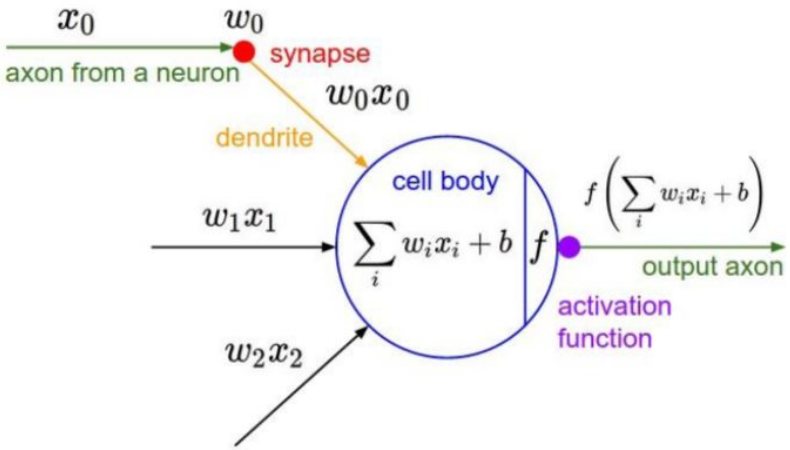


출처: PyTorch로 시작하는 딥러닝 입문 (최근

인공신경망과 생물신경망

- 사람의 뉴런
  - 두뇌의 가장 작은 정보처리 단위
  - 세포체는cell body 간단한 연산, 수상돌기는dendrite 신호 수신, 축삭은axon 처리 결과를 전송
  - 사람은 10^{11}개 정도의 뉴런을 가지며, 뉴런은 1000개 가량 다른 뉴런과 연결되어 있어 10^{14}개 정도의 연결

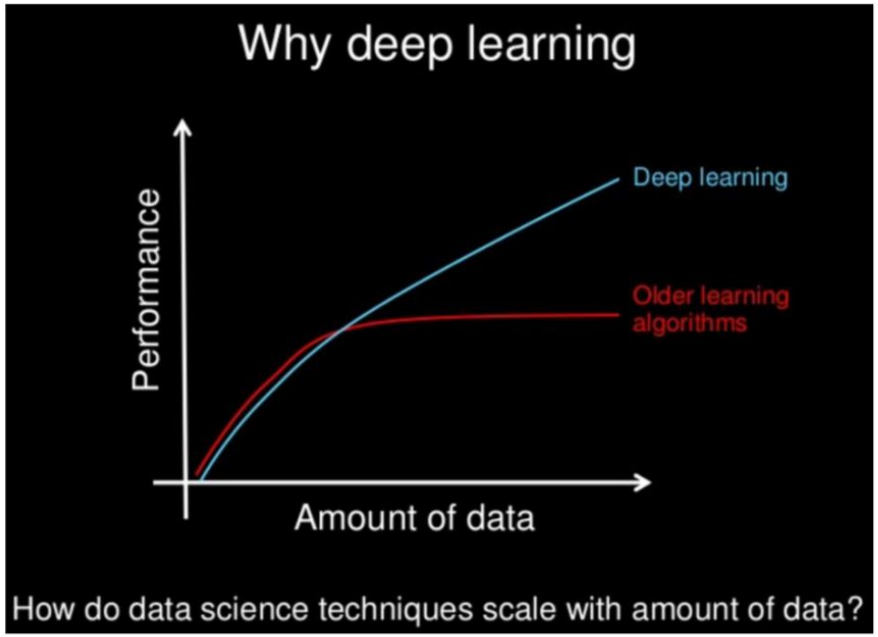




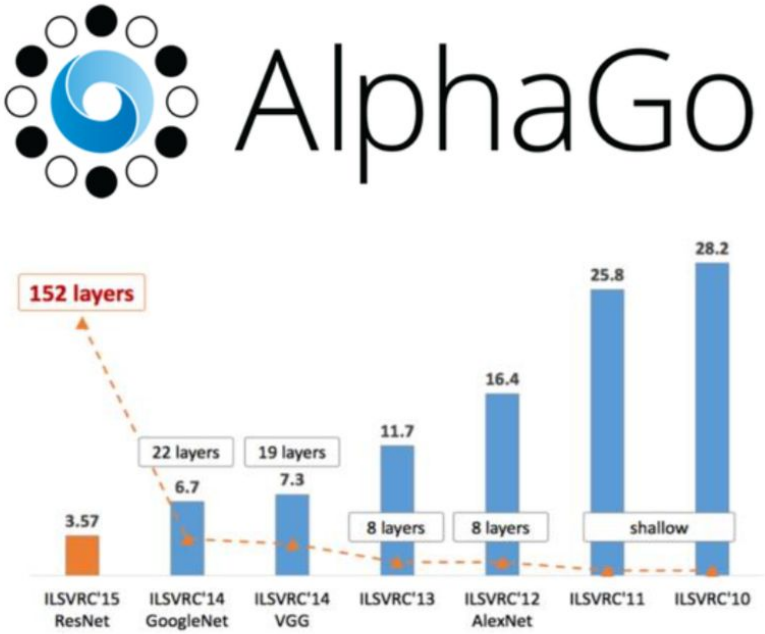
출처: 기계학습 (오원철

근데 딥러닝 왜 써요?

- **비교적 성능이 좋습니다.**



(출처: Andrew Ng)



(출처: Kaming He)

출처: PyTorch로 시작하는 딥러닝 입문 (최근

그럼 전엔 딥러닝 왜 안쓰였나요?

- **컴퓨터가 좋아졌어요. 데이터도**

  뜨게 된 이유

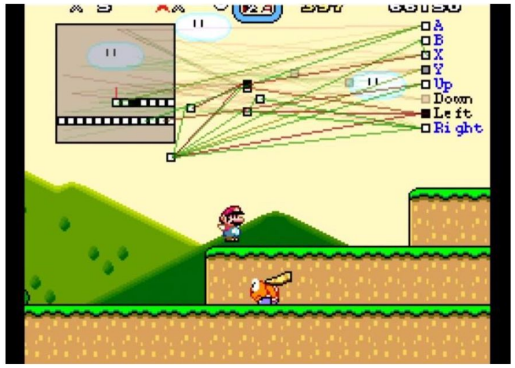
(출처:luluafrikani.wordpress.com)

(출처:http://wccftech.com)

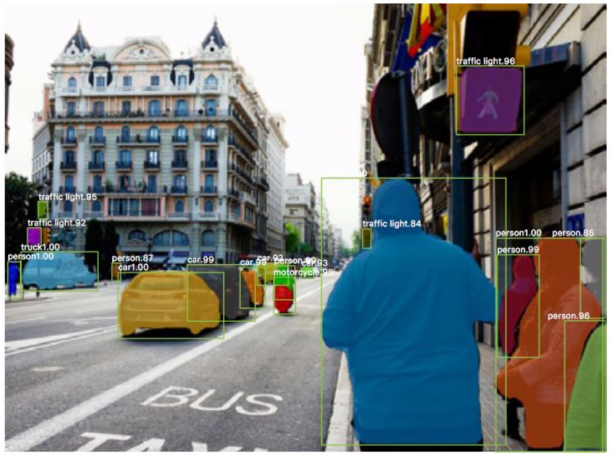
출처: PyTorch로 시작하는 딥러닝 입문 (최근

딥러닝으로 뭐 할 수있어요?

- **할수 있는게 너무 많아요.**



(출처: Marl/O)

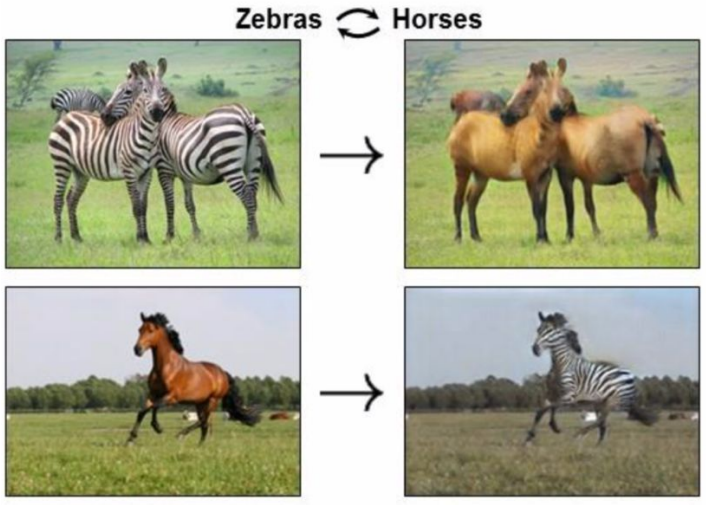


(출처: Mask R-CNN)

출처: PyTorch로 시작하는 딥러닝 입문 (최근

딥러닝으로 뭐 할 수있어요?

- **할수 있는게 너무 많아요.**



(출처: CycleGAN)

(출처: Deep Photo Style Transfer)

출처: PyTorch로 시작하는 딥러닝 입문 (최근

딥러닝은 어떻게 개발해요?

- **Tensorflow, Pytorch, Theano 등으로 개발하는 추세입니다.**

Facebook AI Research

Caffe2

PYTORCH

Yann LeCun

DIRECTOR OF AI RESEARCH

Facebook AI Research (FAIR)

Soumith Chintala

RESEARCH ENGINEER

Facebook AI Research (FAIR)

출처: PyTorch로 시작하는 딥러닝 입문 (최근

왜 Pytorch를 쓰나요?

- 코드가 간결하고 명시적이에요.

vs

PYTORCH

In [4]:
```
import numpy as np
from datetime import datetime
start = datetime.now()

np.random.seed(0)

N,D = 3,4

x = np.random.randn(N,D)
y = np.random.randn(N,D)
z = np.random.randn(N,D)

a = x * y
b = a + z
c = np.sum(b)

grad_c = 1.0
grad_b = grad_c * np.ones((N,D))
grad_a = grad_b.copy()
grad_z = grad_b.copy()
grad_y = grad_a * y
grad_x = grad_a * x

print(grad_x)
print(grad_y)
print(grad_z)
print(datetime.now()-start)
```

```
[[ 1.76405235  0.40015721  0.97873798  2.2408932 ]
 [ 1.86755799 -0.97727788  0.95008842 -0.15135721]
 [-0.10321885  0.4105985   0.14404357  1.45427351]]
[[ 0.76103773  0.12167502  0.44386323  0.33367433]
 [ 1.49407907 -0.20515826  0.3130677  -0.85409574]
 [-2.55298982  0.6536186   0.8644362  -0.74216502]]
[[ 1.  1.  1.  1.]
 [ 1.  1.  1.  1.]
 [ 1.  1.  1.  1.]]
0:00:00.003751
```

In [6]:
```
import torch
from torch.autograd import Variable
from datetime import datetime
start = datetime.now()

N,D = 3,4

x = Variable(torch.randn(N,D).cuda(),requires_grad=True)
y = Variable(torch.randn(N,D).cuda(),requires_grad=True)
z = Variable(torch.randn(N,D).cuda(),requires_grad=True)

a = x * y
b = a + z
c = torch.sum(b)

c.backward(gradient=torch.cuda.FloatTensor([1.0]))

print(x.grad)
print(y.grad)
print(z.grad)
print(datetime.now()-start)
```

```
Variable containing:
-0.6048  0.6640 -1.8035  1.0894
-0.0731 -0.0702 -0.0474 -1.7545
-0.3247  0.6293  2.5135 -0.5967
[torch.cuda.FloatTensor of size 3x4 (GPU 0)]

Variable containing:
 0.1152  1.1809  1.4522  1.9417
 1.0845  0.1587 -1.6526  0.4031
 1.9585 -0.4729 -1.4024 -0.7388
[torch.cuda.FloatTensor of size 3x4 (GPU 0)]

Variable containing:
 1  1  1  1
 1  1  1  1
 1  1  1  1
[torch.cuda.FloatTensor of size 3x4 (GPU 0)]

0:00:00.003434
```

출처: PyTorch로 시작하는 딥러닝 입문 (최근

왜 Pytorch를 쓰나요?

- 딥러닝에 필요한 미분 작업을 자동으로 해줍니다.

vs

Gradient Calculation

```
In [4]: import numpy as np
from datetime import datetime
start = datetime.now()

np.random.seed(0)

N,D = 3,4

x = np.random.randn(N,D)
y = np.random.randn(N,D)
z = np.random.randn(N,D)

a = x * y
b = a + z
c = np.sum(b)

grad_c = 1.0
grad_b = grad_c * np.ones((N,D))
grad_a = grad_b.copy()
grad_z = grad_b.copy()
grad_y = grad_a * y
grad_x = grad_a * x

print(grad_x)
print(grad_y)
print(grad_z)
print(datetime.now()-start)
```

```
[[ 1.76405235  0.40015721  0.97873798  2.2408932 ]
 [ 1.86755799 -0.97727788  0.95008842 -0.15135721]
 [-0.10321885  0.4105985   0.14404357  1.45427351]]
[[ 0.76103773  0.12167502  0.44386323  0.33367433]
 [ 1.49407907 -0.20515826  0.3130677  -0.85409574]
 [-2.55298982  0.6536186   0.8644362  -0.74216502]]
[[ 1.  1.  1.  1.]
 [ 1.  1.  1.  1.]
 [ 1.  1.  1.  1.]]
0:00:00.003751
```

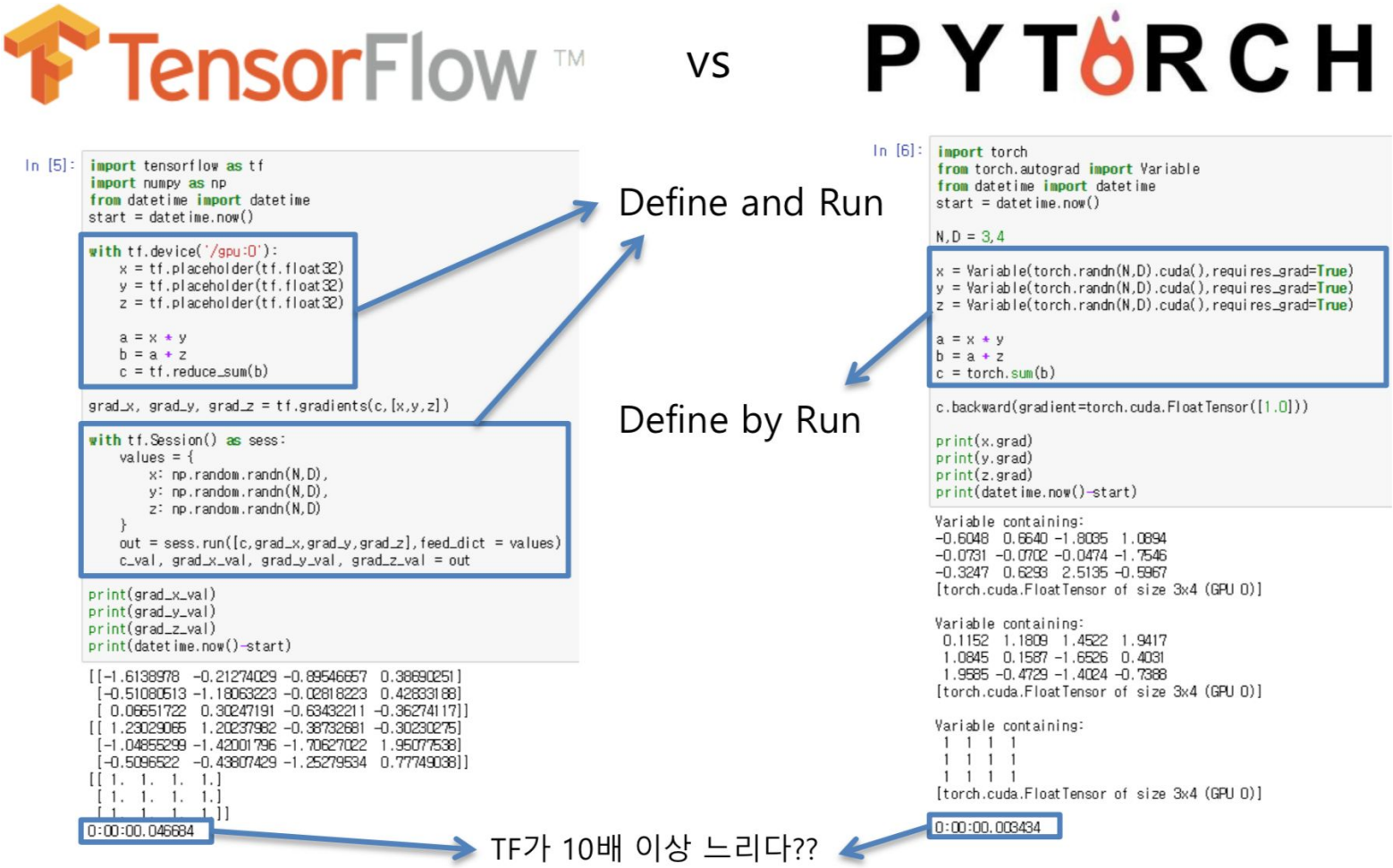
```
In [6]: import torch
from torch.autograd import Variable
from datetime import datetime
start = datetime.now()

N,D = 3,4

x = Variable(torch.randn(N,D).cuda(),requires_grad=True)
y = Variable(torch.randn(N,D).cuda(),requires_grad=True)
z = Variable(torch.randn(N,D).cuda(),requires_grad=True)

a = x * y
b = a + z
c = torch.sum(b)

c.backward(gradient=torch.cuda.FloatTensor([1.0]))

print(x.grad)
print(y.grad)
print(z.grad)
print(datetime.now()-start)
```

```
Variable containing:
-0.6048  0.6640 -1.8035  1.0894
-0.0731 -0.0702 -0.0474 -1.7546
-0.3247  0.6293  2.5135 -0.5967
[torch.cuda.FloatTensor of size 3x4 (GPU 0)]

Variable containing:
 0.1152  1.1809  1.4522  1.9417
 1.0845  0.1587 -1.6526  0.4031
 1.9585 -0.4729 -1.4024 -0.7388
[torch.cuda.FloatTensor of size 3x4 (GPU 0)]

Variable containing:
 1  1  1  1
 1  1  1  1
 1  1  1  1
[torch.cuda.FloatTensor of size 3x4 (GPU 0)]

0:00:00.003434
```

출처: PyTorch로 시작하는 딥러닝 입문 (최근

왜 Pytorch를 쓰나요?

- 속도도 빠릅니다. (극단적 예제이지만 빠름)



출처: PyTorch로 시작하는 딥러닝 입문 (최근

왜 Pytorch를 쓰나요?

vs

- Define and Run
- Static graph
- 사용자 많음
- 자체 운영 포럼 없음
- TF-KR

- Define by Run
- Dynamic graph
- 늘어나는 추세
- 자체적 포럼 운영
- PyTorch-KR

출처: PyTorch로 시작하는 딥러닝 입문 (최근

Pytorch 실습 하러 가요