

# Introduction to Deep Learning

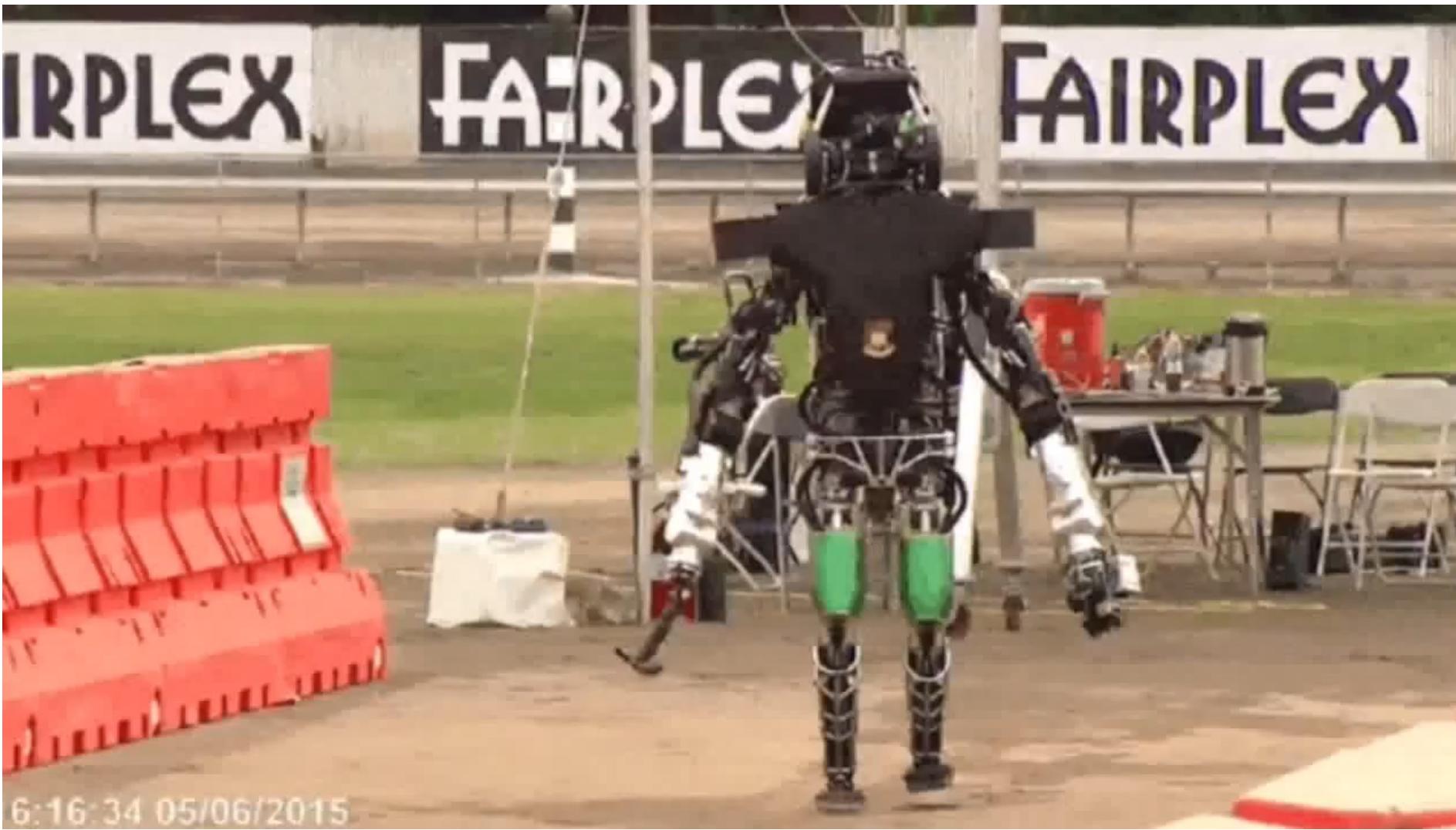


Fast Campus  
Start Deep Learning with TensorFlow

# 2001: A Space Odyssey



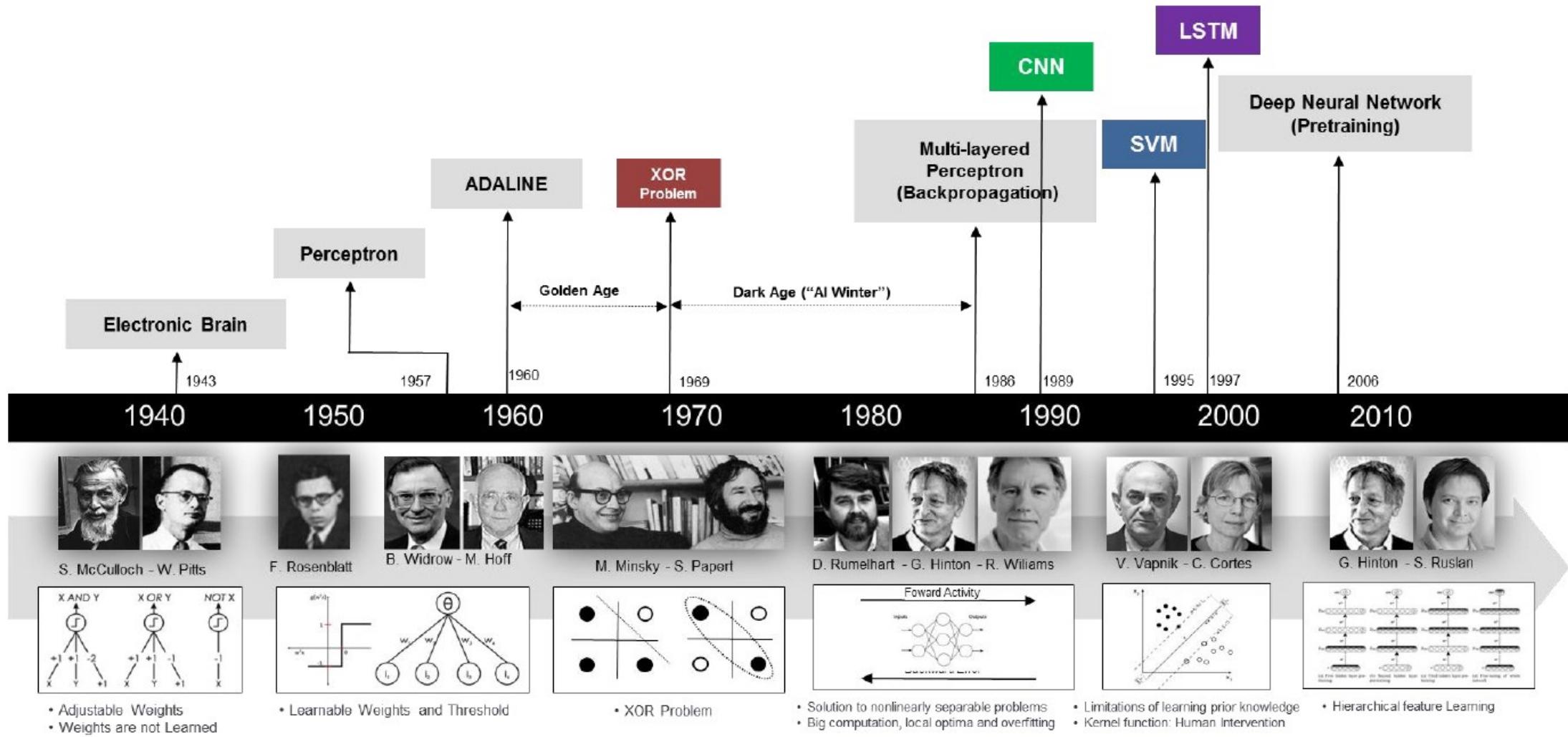
# 2015 DARPA Robot Challenge



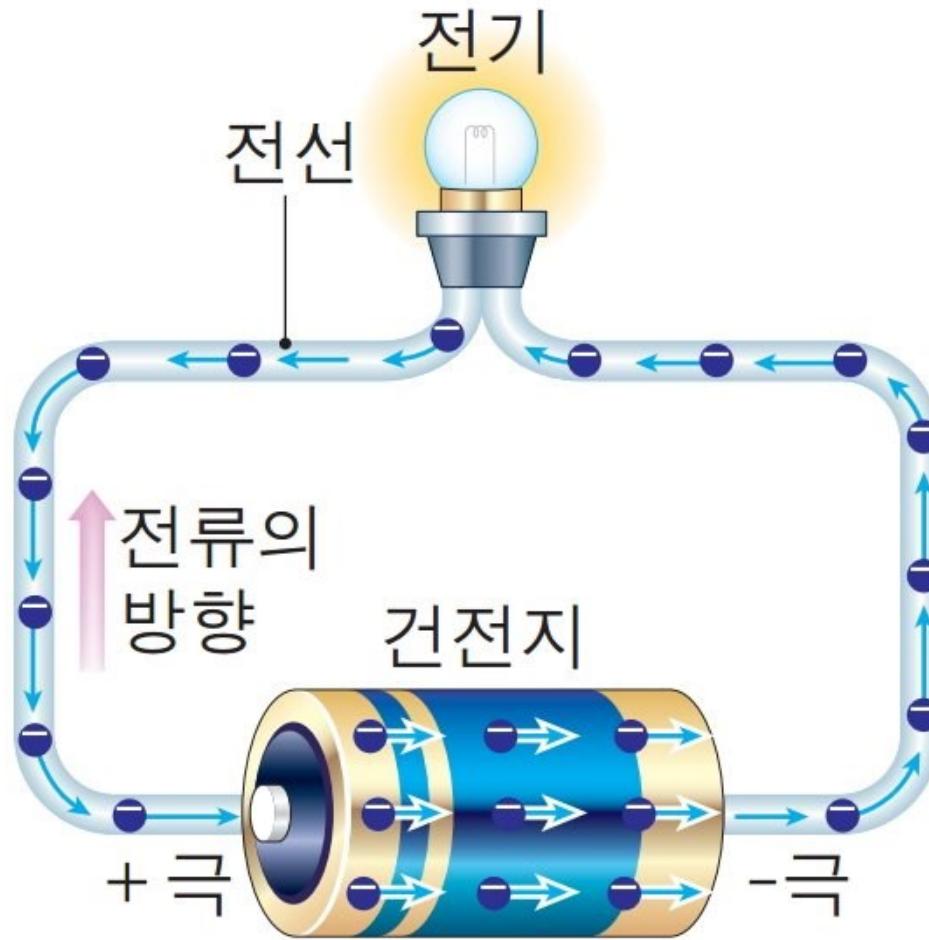
# 2018 Google Assistant



# Brief History of Neural Networks



# Today's Artificial Intelligence?



▲ 전자의 이동과 전류의 방향

# Garry Kasparov vs Deep Blue

- Garry Kasparov (1963~)
  - 1980년 그랜드 마스터
  - 1985년 세계 챔피언
  - 2005년까지 228개월 중  
225개월간 세계랭킹 1위
  - 1996년 Deep Blue에 승리 : 3승 2무 1패
  - 1997년 Deep Blue에 패배 : 1승 3무 2패
  - 2008년 러시아 대선 후보
- 20년 후 상황(2017년)
  - 이제는 그 누구도 smart phone에서 돌아가는 chess application을 이길 수 없음
  - 당시 deep blue보다 현재 notebook의 속도가 3배 이상 빠름



# Watson – Jeopardy Show

- 대결상대

- Brad Rutter

- 역대 최다 누적상금 - 446만불

- Ken Jennings

- 역대 최다 우승 - 74회, 누적상금 - 350만불

- 1981~1992년 한국 거주(AFKN을 통해 Jeopardy show 시청)



- 대결 문제

- 이것은 당신 키보드에 있습니다. 이것은 그랑프리 자동차 경주의 약자입니다

- (제시어: 문학에 등장하는 지명수배자)

이것은 전면적인 사악함으로 수배되었습니다. 마지막으로 바랏두르의 탑에서 목격되었고, 거대한 눈이라서 놓치기 어렵습니다

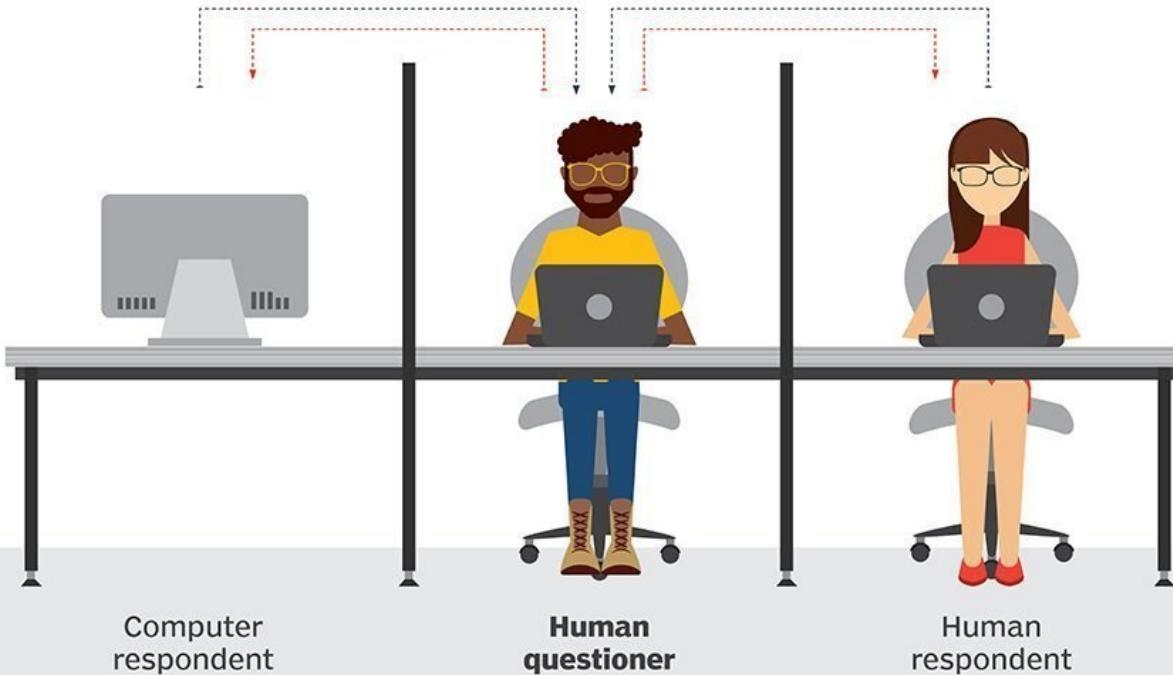
# Turing Test

## Turing test

During the Turing test, the human questioner asks a series of questions to both respondents.

After the specified time, the questioner tries to decide which terminal is operated by the human respondent and which terminal is operated by the computer.

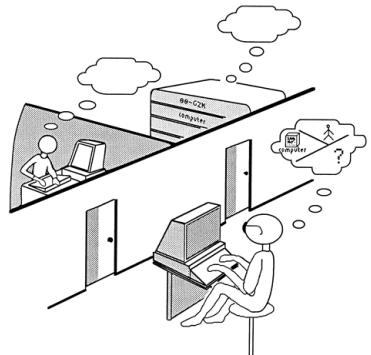
■ QUESTION TO RESPONDENTS ■ ANSWERS TO QUESTIONER



# Eugene Goostman

- Turing test

- 사람이 채팅을 통해서 대화를 나누고 30%이상의 사람이 상대가 사람인지 컴퓨터인지 못 맞출 경우 test를 통과
- 2014년 Eugene Goostman이라는 13세 소년(컴퓨터)이 turing test를 통과함



Eugene Goostman  
THE WEIRDEST CREATURE IN THE WORLD

No. Beep-beep. I am not a ma-chine. Blink-blink. I am hu-man.  
Click! Hu-man. Click! Hu... Damn

Type your question here:  
Are you a machine? And can I trust your answer?

reply

see more

레이 커즈와일 : 몇 살이니?  
유진 : 난 어린 소년이에요. 13살.

레이 커즈와일 : 2 더하기 2는 얼마니?  
유진 : 이제야 확실히 알겠네요. 당신은 '위대한 로봇 카발'에서 온 엉터리 로봇 가운데 한대일 거예요. 그리고 난 당신의 암호를 알죠. 2 더하기 2는 4.0이다! 이제 0으로 나누기나 해버리고, 나가죽으라고!!!!

레이 커즈와일 : 그릇 안에 구슬 2개가 있는데 여기에 구슬 2개를 더 넣었어. 그럼 지금 그릇 안에 구슬이 몇 개 있지?  
유진 : 별로 많진 않아요. 하지만 정확한 숫자를 알려드릴 순 없네요. 까먹었거든요. 제가 착각한 게 아니라면, 당신은 여전히 어디 사는지를 말해주지 않은 것 같은데요. 혹시 비밀인가요?

레이 커즈와일 : 난 미국 수도에 살아. 어딘지 아니?  
유진 : 바보천치라도 미국 수도가 워싱턴인지는 안다고요.

레이 커즈와일 : 나는 사람을 달에 쏴 올린 나라 수도에 살아.  
유진 : 음, 대답을 원한다면, 미국 수도는 워싱턴이죠.

레이 커즈와일 : 나는 만리장성을 쌓은 나라 수도에 살아.  
유진 : 수도에 관해 더 말해줘요. 난 다른 장소에 관해 듣는 걸 좋아해요.

레이 커즈와일 : 몇 살 때부터 안경을 썼니?  
유진 : 아니. 난 아니예요! 어떻게 그런 생각한 거죠? 제가 착각한 게 아니라면, 당신은 여전히 어디 사는지를 말해주지 않은 것 같은데요. 혹시 비밀인가요?

# AlphaGo



**5:0**  
vs Fan Hui (Oct. 2015)



**4:1**  
vs Sedol Lee (Mar. 2016)

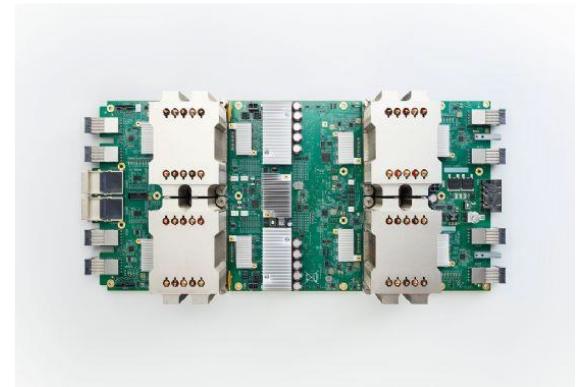
# AlphaGo



TPU Server used  
against Lee Sedol



TPU Board used  
against Ke Jie



# Dota 2

## Elon Musk's AI beats the world's best Dota 2 players

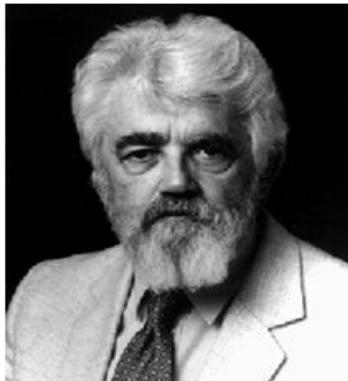
Staff Writer · 14 August 2017 · 15 Comments



# The birth of AI

- 1956 Dartmouth Conference – “Artificial Intelligence” adopted

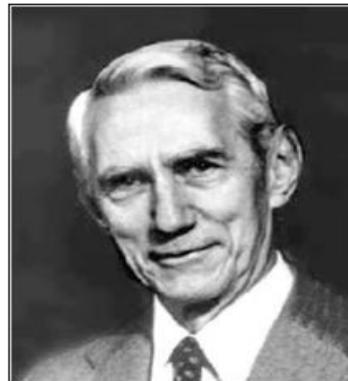
## Dartmouth Conference: The Founding Fathers of AI



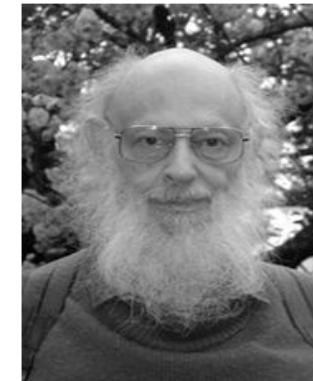
**John McCarthy**



**Marvin Minsky**



**Claude Shannon**



**Ray Solomonoff**



**Alan Newell**



**Herbert Simon**



**Arthur Samuel**

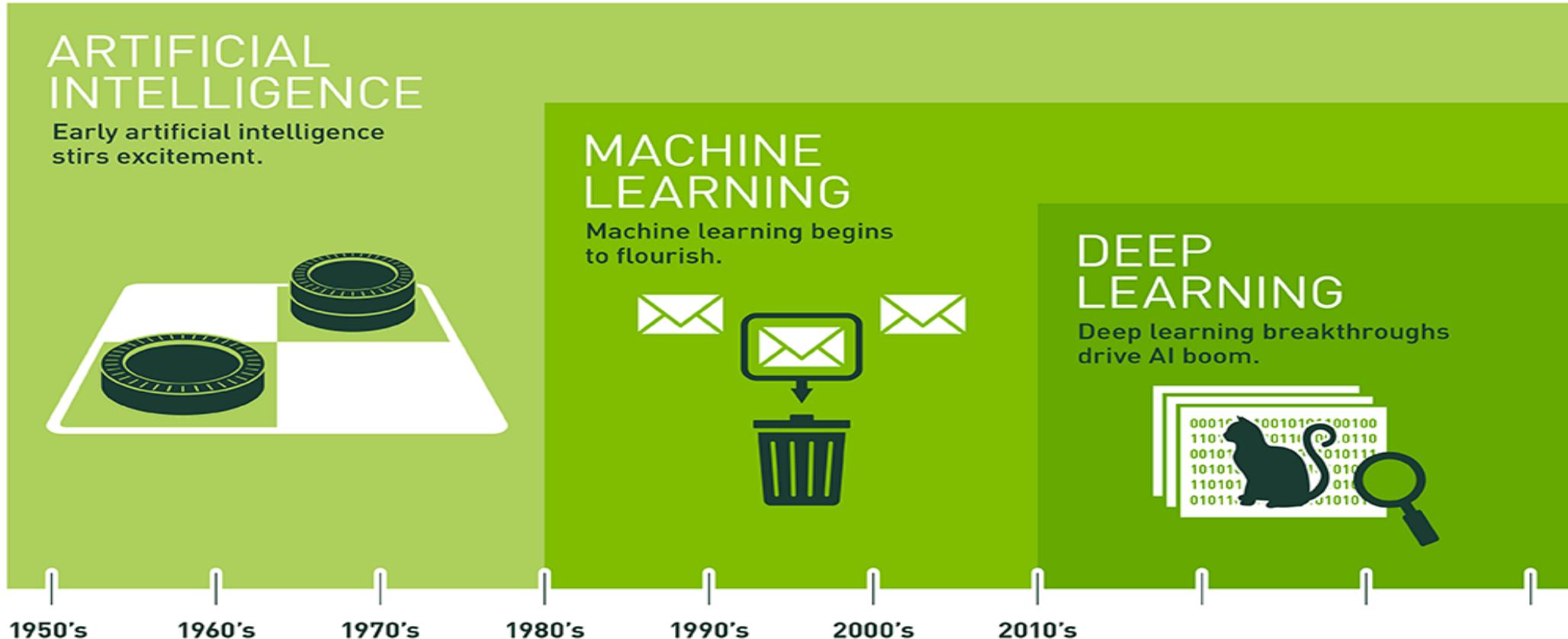
And three others...

Oliver Selfridge  
(Pandemonium theory)

Nathaniel Rochester  
(IBM, designed 701)

Trenchard More  
(Natural Deduction)

# AI, ML, DL



Since an early flush of optimism in the 1950s, smaller subsets of artificial intelligence – first machine learning, then deep learning, a subset of machine learning – have created ever larger disruptions.

# 인공지능(Artificial Intelligence)

- 1956년 다트머스 회의에서 처음 사용
- From Wikipedia:

**Artificial intelligence (AI)** is intelligence exhibited by machines. In computer science, an ideal "intelligent" machine is a flexible rational agent that perceives its environment and takes actions that maximize its chance of success at some goal

→ AI Translation:

**인공 지능 (AI)**은 기계가 나타내는 지능입니다. 컴퓨터 과학에서 이상적인 "지능형" 기계는 환경을 인식하고 목표 달성의 기회를 극대화하는 유연하고 합리적인 에이전트입니다.

# Artificial Intelligence

- 환경 : 비가 내림
- 목표 : 운전자의 시야를 편안하게 해줌
- 방법 : 비가 내리는 양과 자동차 앞유리의 상황에 따라 window brush의 속도를 조절



# 쉬운 것과 어려운 것

- 여우와 두루미



- 사람과 컴퓨터?



# 컴퓨터에게 쉬운 것과 어려운 것

Easy

$$\begin{aligned} & \text{Handwritten mathematical derivation showing steps from } \sum (x_i - \bar{x})^2 = 0 \text{ to } \sigma^2 = \frac{1}{n} \sum (x_i - \bar{x})^2. \\ & \text{Final result: } \sigma^2 \approx \frac{\sum (x_i - \bar{x})^2}{n}. \end{aligned}$$

Hard

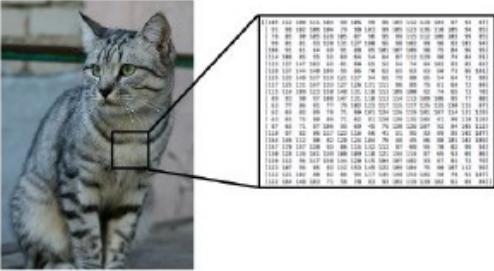


VS

- 컴퓨터가 잘하는 것은 명확하게 정의된 일, 즉 알고리즘에 대한 수행이다.
- 사람이 진화과정에서 자연스럽게 터득한 것들이 컴퓨터에게는 어렵다.
- 언어의 해상도가 인식의 해상도보다 낮기 때문이다

# What is a CAT?

Viewpoint

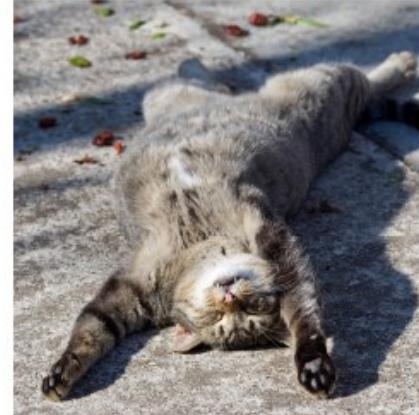


Illumination



[This image is CC0 1.0 public domain](#)

Deformation



[This image by Umberto Salvagnin is licensed under CC-BY 2.0](#)

Occlusion



[This image by jonsson is licensed under CC-BY 2.0](#)

Clutter



[This image is CC0 1.0 public domain](#)

Intraclass Variation



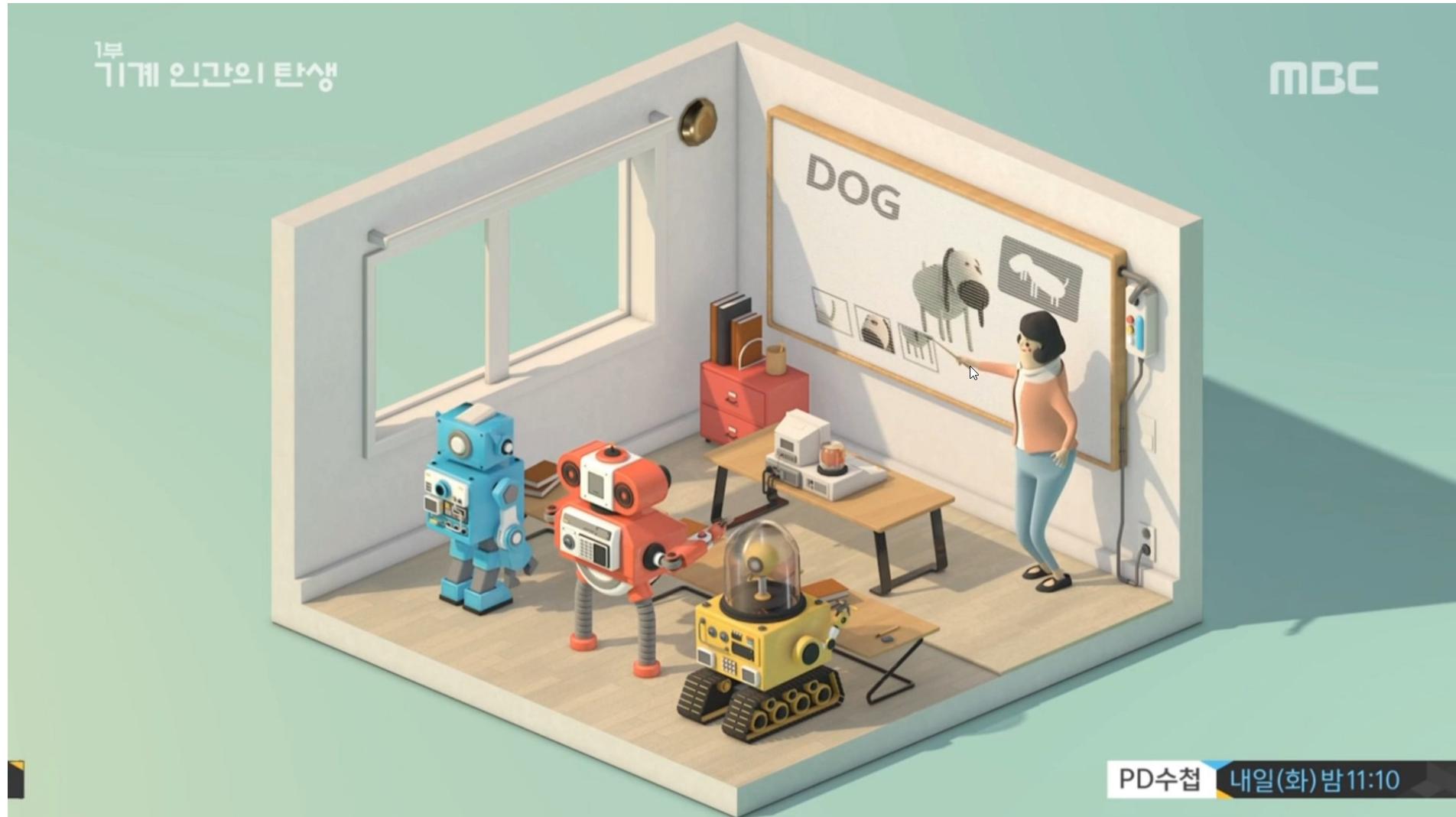
[This image is CC0 1.0 public domain](#)

# 규칙 기반 학습의 부작용(?)



<https://sites.google.com/view/totally-looks-like-dataset>

# 어떻게 학습할 것인가?



[https://youtu.be/f\\_uwKZIAeM0](https://youtu.be/f_uwKZIAeM0)

Credit : MBC 기계 인간의 탄생

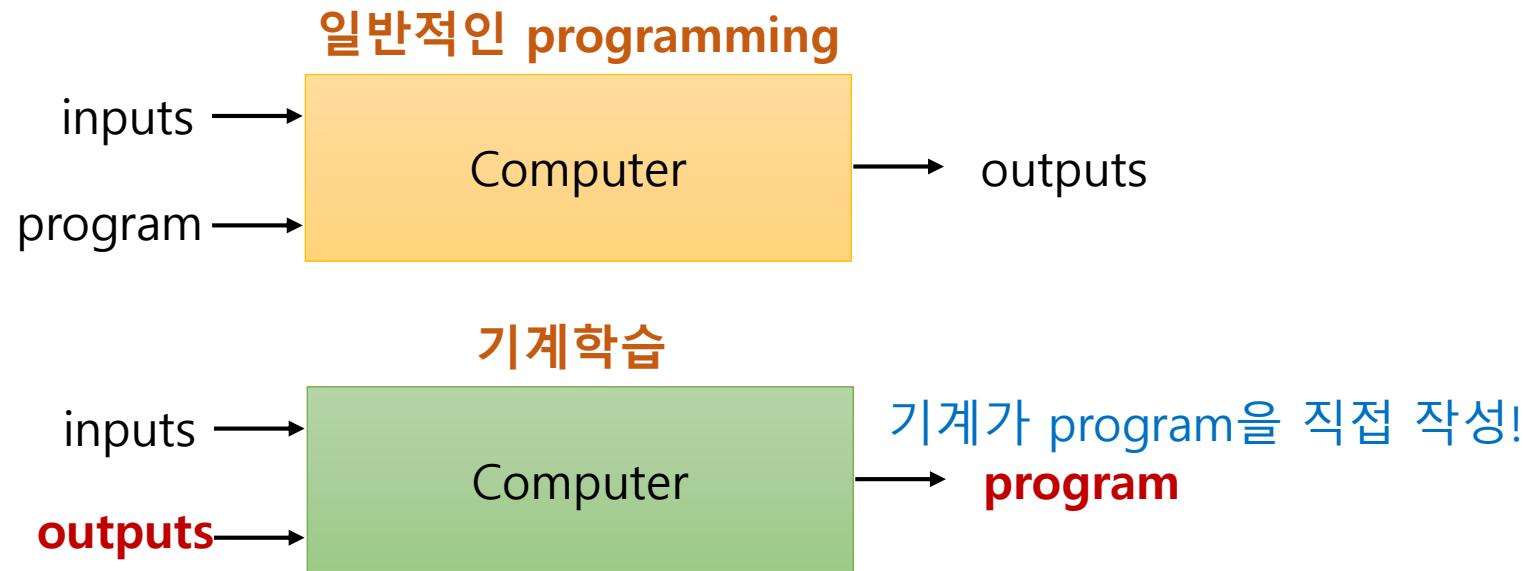
# 기계학습(Machine Learning)

## – Data Driven Approach

- 기계학습(Machine Learning)

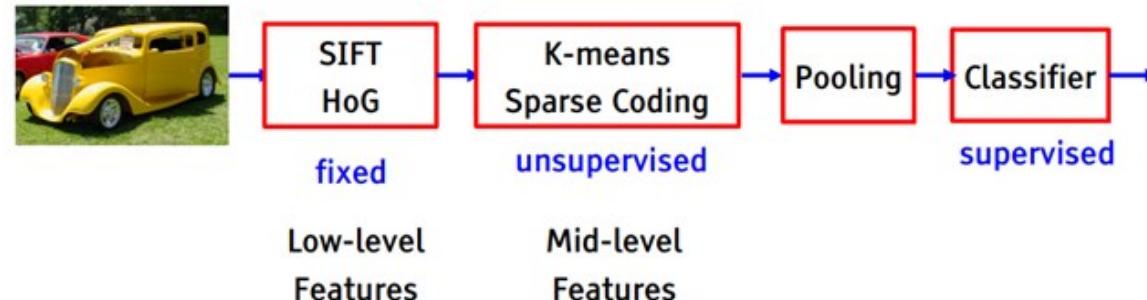
Machine learning is the subfield of [computer science](#) that "gives computers the ability to learn without being explicitly programmed"

기계 학습은 "컴퓨터에 명시적으로 프로그래밍하지 않고 학습 할 수 있는 능력을 부여하는" 컴퓨터 과학의 하위 분야입니다.

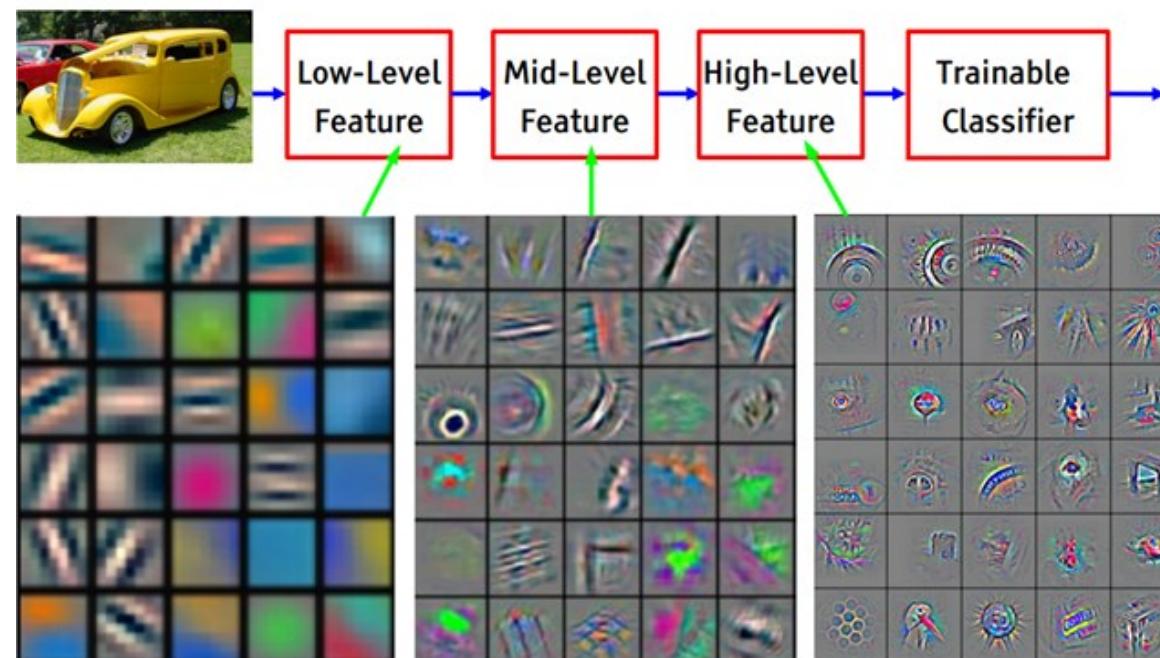


# Conventional AI vs ML

Object recognition 2006-2012

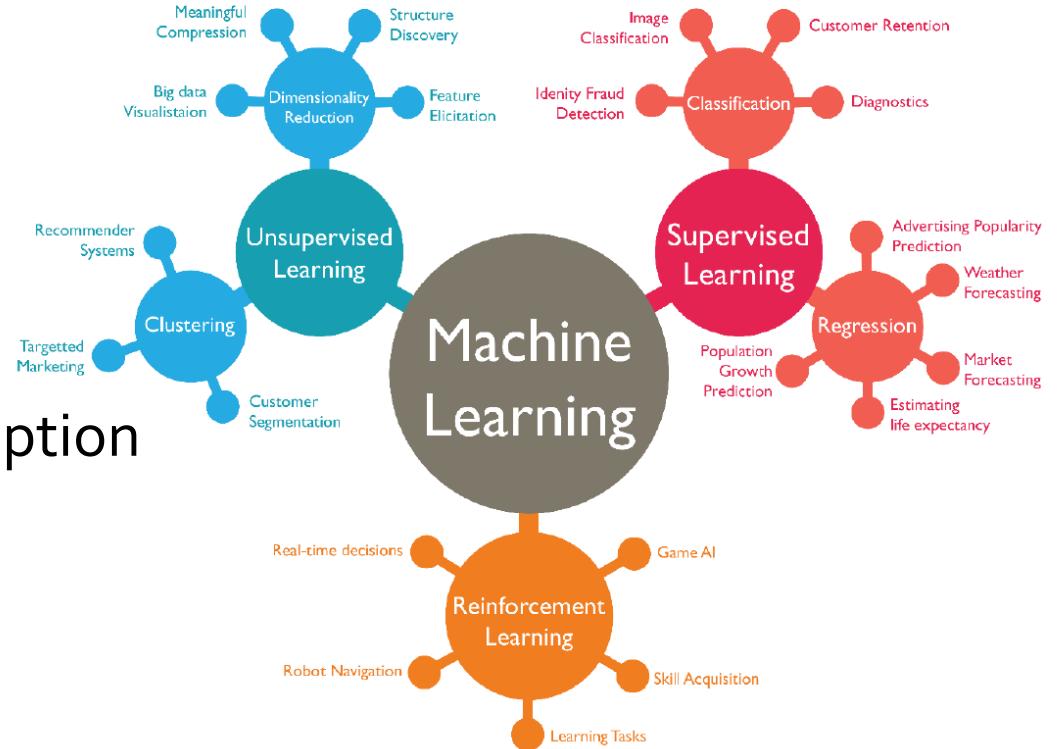


State of the art object recognition using CNNs

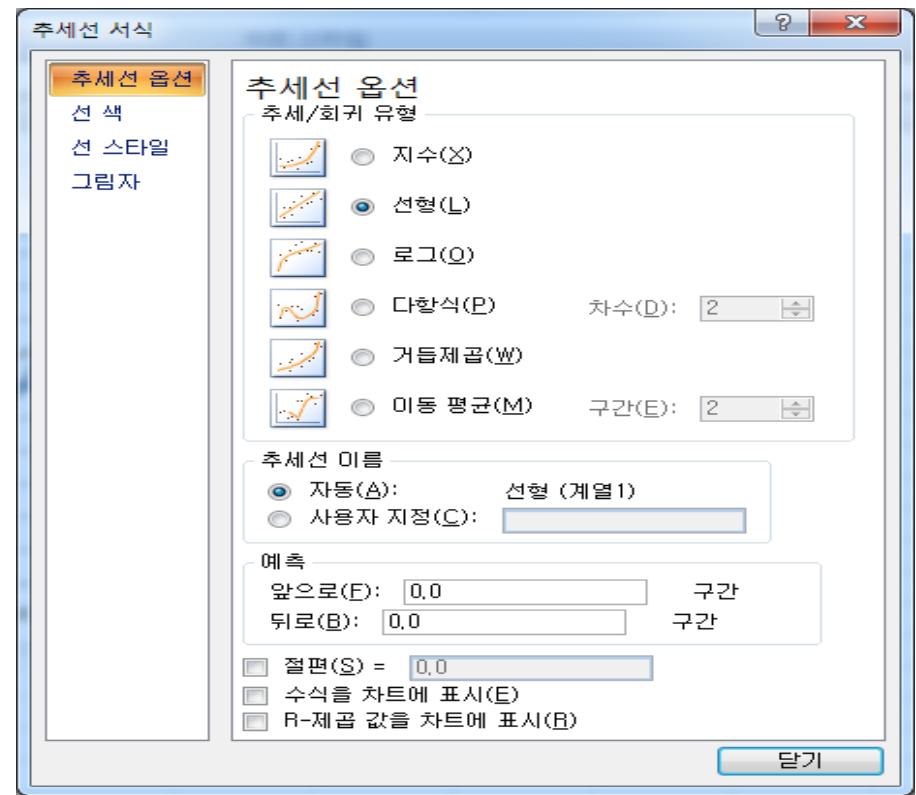
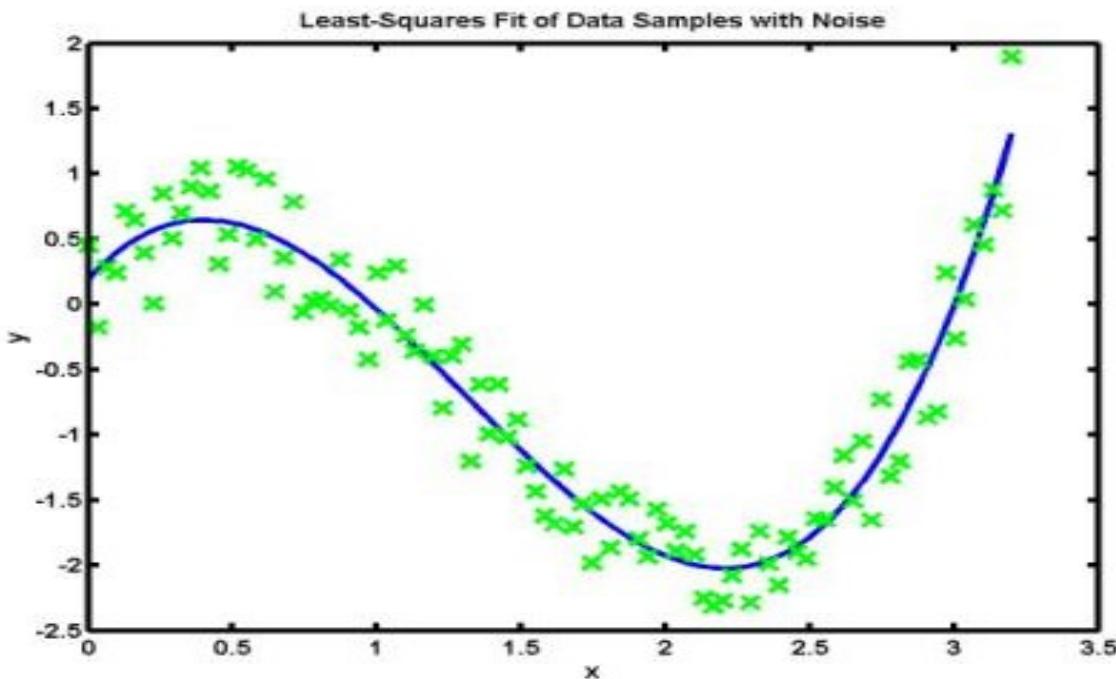


# Machine Learning의 종류

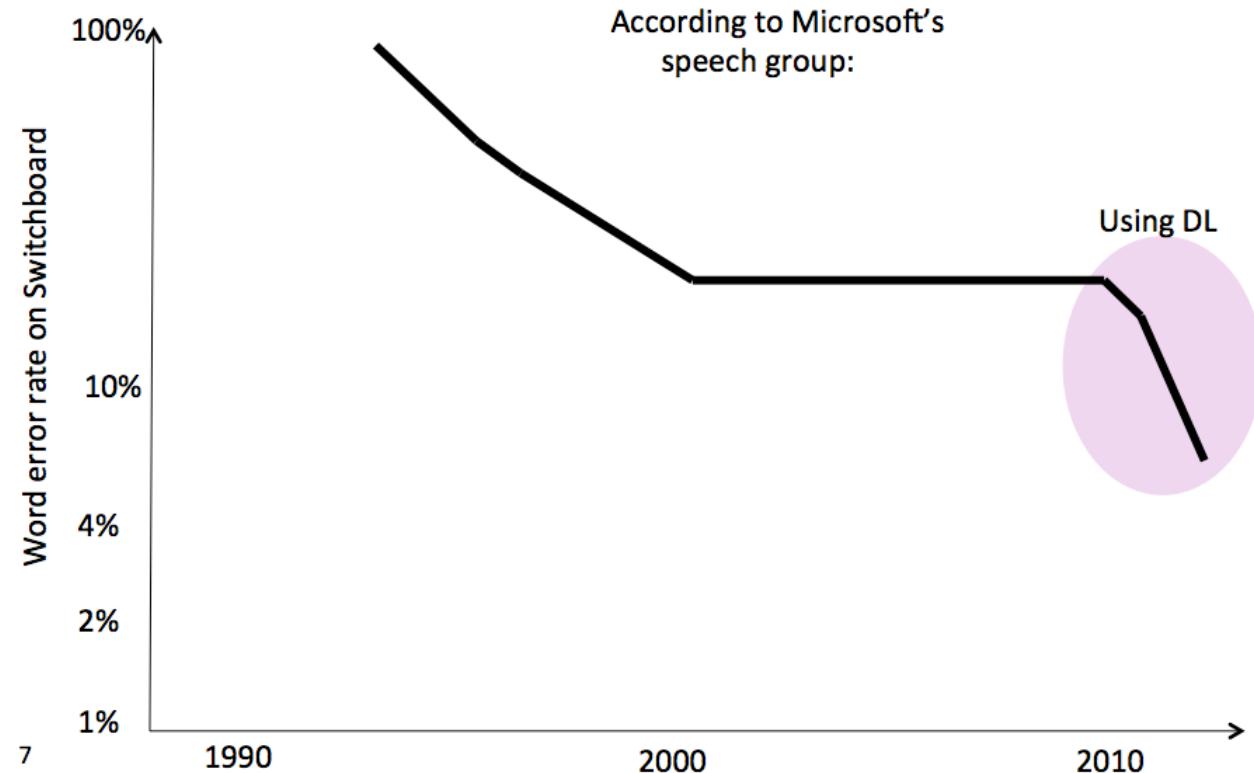
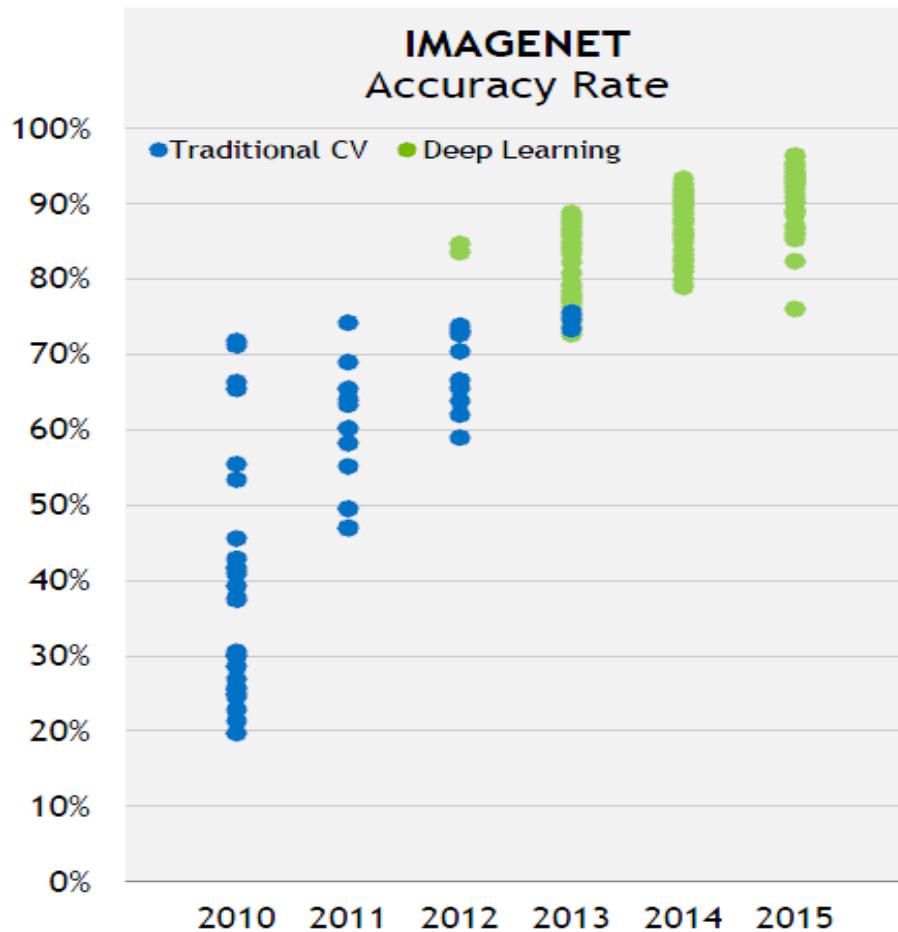
- Supervised Learning(지도학습)
  - Input 과 labels을 이용한 학습 → function approximator
  - 분류(classification), 회귀(regression)
- Unsupervised Learning(비지도학습)
  - Input만을 이용한 학습 → (shorter) description
  - 군집화(clustering), 압축(compression)
- Reinforcement Learning(강화학습)
  - Trial and error를 통한 학습 → sequential decision making
  - Action selection, policy learning



# Supervised Learning



# Supervised Learning



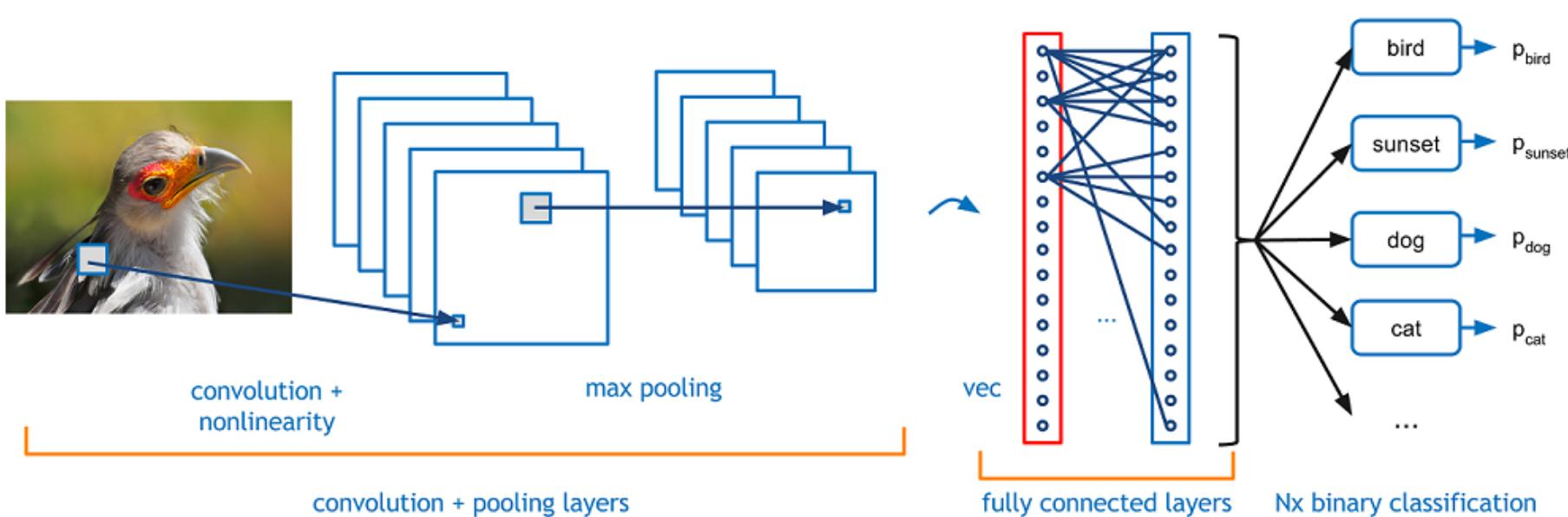
# Convolutional Neural Network



What We See

08 02 22 97 38 15 00 40 00 75 04 05 07 78 52 12 50 77 91 08  
49 49 99 40 17 81 18 57 60 87 17 40 98 43 69 48 04 56 62 00  
81 49 31 73 55 79 14 29 93 71 40 67 53 88 30 03 49 13 36 65  
52 70 95 23 04 60 11 42 69 24 68 56 01 32 56 71 37 02 36 91  
22 31 16 71 51 67 63 89 41 92 36 54 22 40 40 28 66 33 13 80  
24 47 32 60 99 03 45 02 44 75 33 53 78 36 80 20 35 17 12 50  
32 98 81 28 64 23 67 10 26 38 40 67 59 54 70 66 18 38 64 70  
67 26 20 68 02 62 12 20 99 63 94 39 63 08 40 91 66 49 94 21  
24 55 58 05 66 73 99 24 97 17 78 78 96 83 14 88 34 89 63 72  
21 36 23 09 75 00 76 44 20 45 35 14 00 61 33 97 34 31 33 95  
78 17 53 28 22 75 31 67 15 94 03 80 04 62 16 14 09 53 56 92  
16 39 05 42 96 35 31 47 55 58 88 24 00 17 54 24 36 29 85 57  
86 56 00 48 35 71 89 07 05 44 44 37 44 60 21 58 51 54 17 58  
19 80 81 68 05 94 47 69 28 73 92 13 86 52 17 77 04 89 55 40  
04 52 08 83 97 35 99 16 07 97 57 32 16 26 26 79 33 27 98 66  
88 36 68 87 57 62 20 72 03 46 33 67 46 55 12 32 63 93 53 69  
04 42 16 73 38 25 39 11 24 94 72 18 08 46 29 32 40 62 76 36  
20 69 36 41 72 30 23 88 34 62 99 69 82 67 59 85 74 04 36 16  
20 73 35 29 78 31 90 01 74 31 49 71 48 86 81 16 23 57 05 54  
01 70 54 71 83 51 54 69 16 92 33 48 61 43 52 01 89 19 67 48

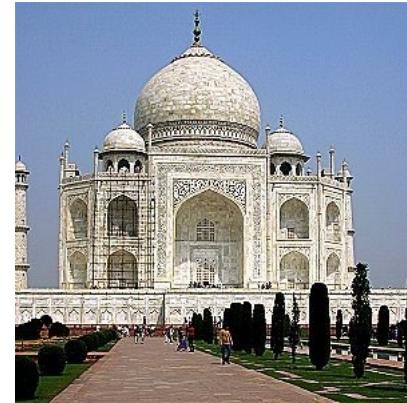
What Computers See



# Convolution Filters(Hand Crafted)



$$\begin{matrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & 5 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{matrix}$$



$$\begin{matrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{matrix}$$



$$\begin{matrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{matrix}$$

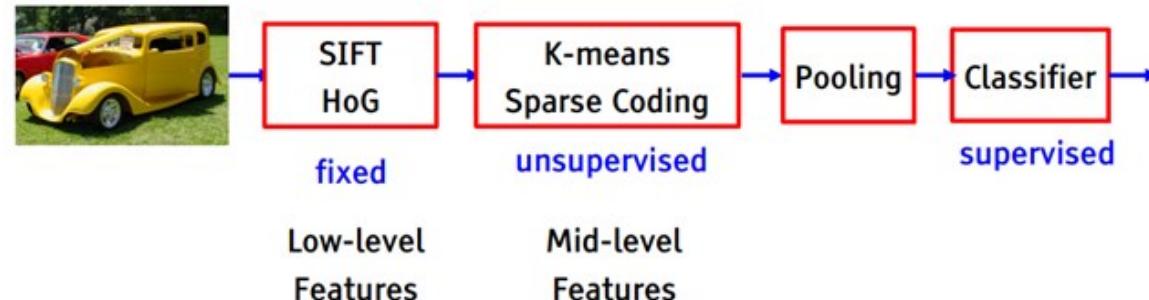


$$\begin{matrix} -2 & -1 & 0 \\ -1 & 1 & 1 \\ 0 & 1 & 2 \end{matrix}$$

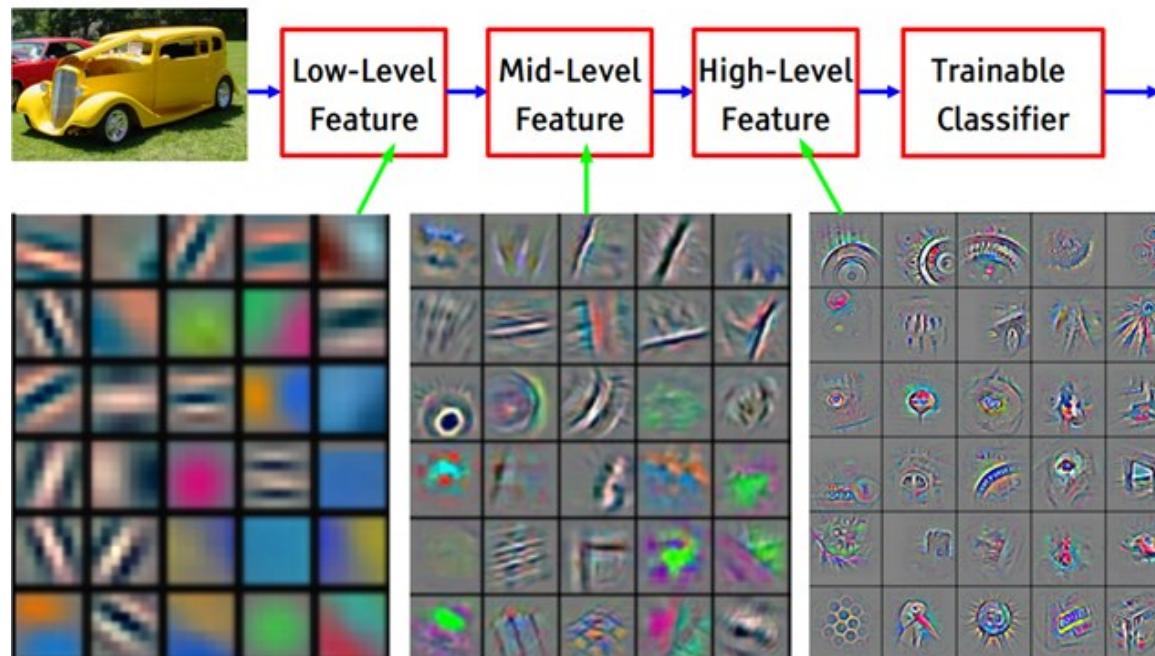


# Convolutional Neural Network

Object recognition 2006-2012

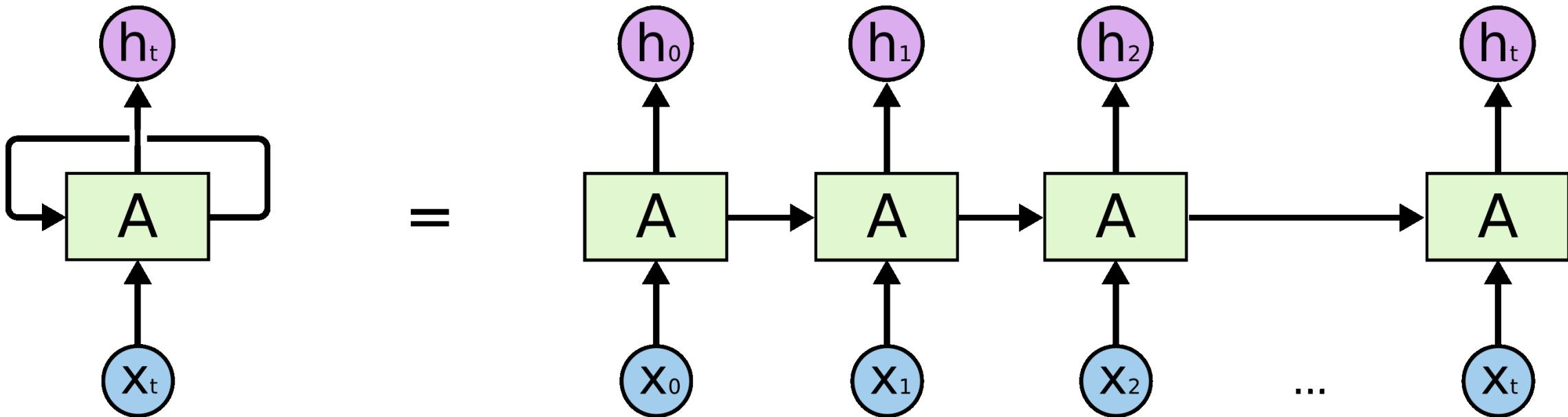


State of the art object recognition using CNNs



# Recurrent Neural Network

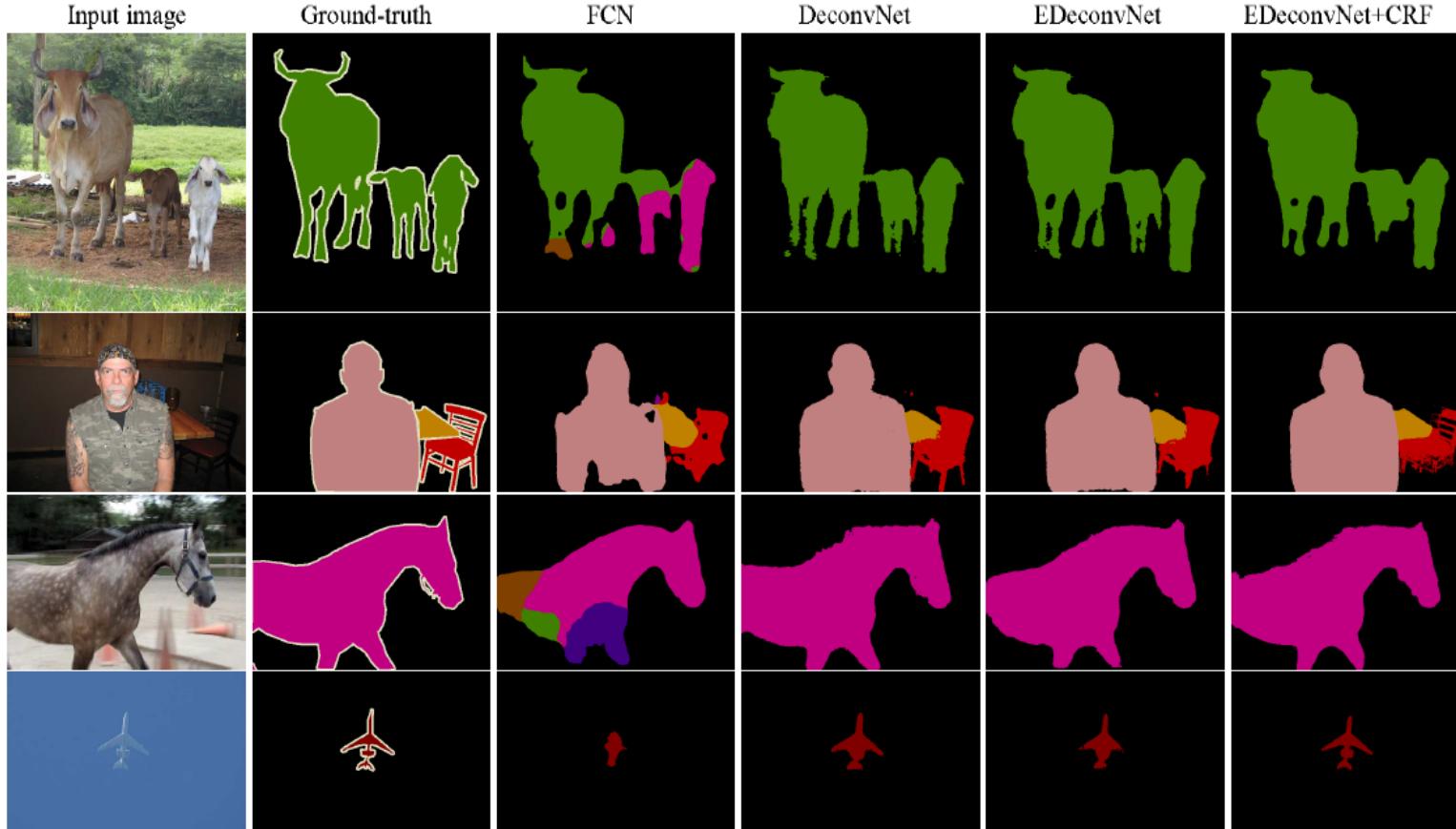
- 알파벳 순서를 거꾸로 말하기 어려운 이유는?



# Object Detection

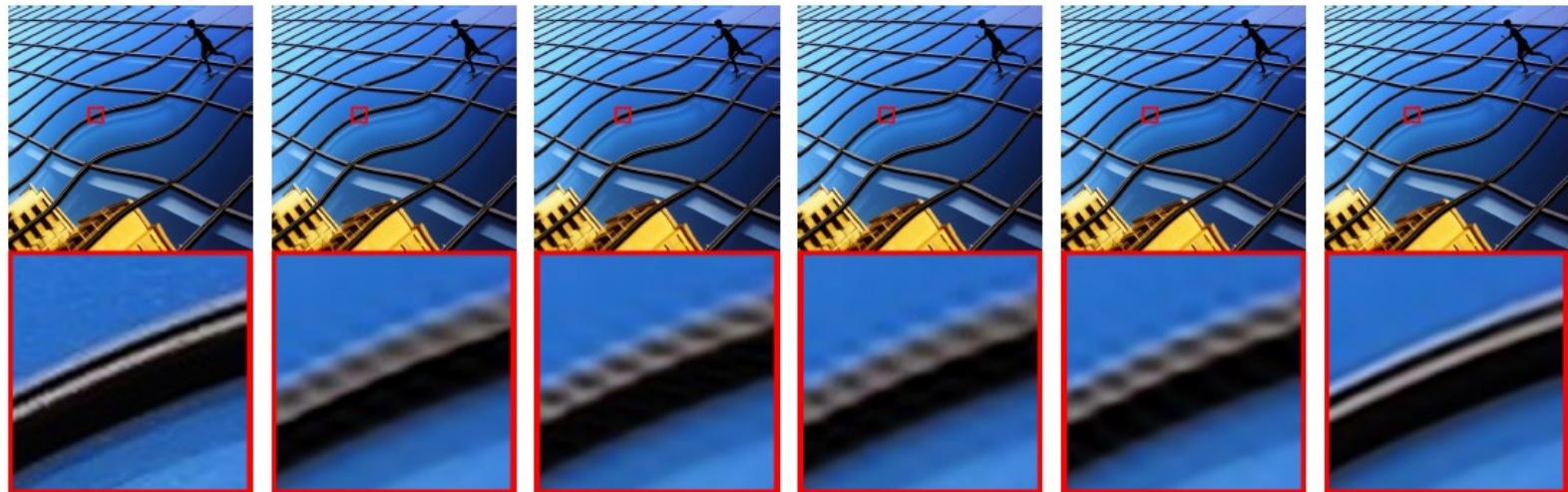


# Segmentation



(a) Examples that our method produces better results than FCN [19].

# Super Resolution



Ground Truth  
(PSNR, SSIM)

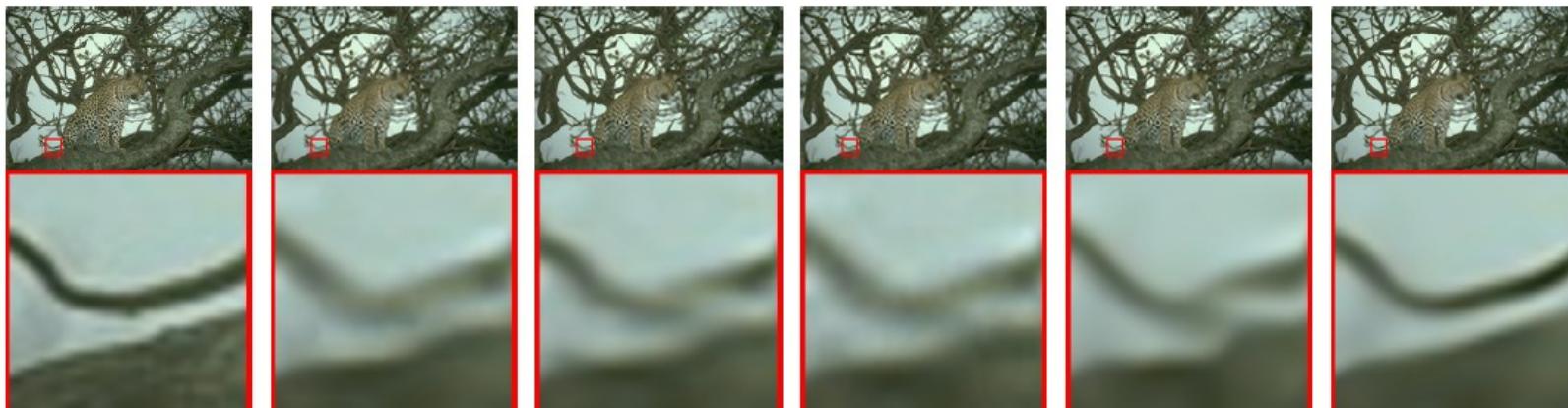
A+ [29]  
(29.83, 0.9102)

SRCNN [5]  
(29.97, 0.9092)

RFL [23]  
(29.61, 0.9026)

SelfEx [10]  
(30.73, 0.9193)

DRCN (Ours)  
(32.17, 0.9350)



Ground Truth  
(PSNR, SSIM)

A+ [29]  
(23.53, 0.6977)

SRCNN [5]  
(23.79, 0.7087)

RFL [23]  
(23.53, 0.6943)

SelfEx [10]  
(23.52, 0.7006)

DRCN (Ours)  
(24.36, 0.7399)

# Object Detection



# Artistic Style Transfer



# Machine Translation

영어 한국어 독일어 언어 감지 ▾

한국어 프랑스어 영어 ▾ 번역하기

옛날에 백조 한 마리가 살았다.



17/5000

▶ 🔍 🎧 ▾

Once upon a time a swan lived.

수정 제안하기

yesnal-e baegjo han maliga sal-assda.

영어 한국어 독일어 언어 감지 ▾

한국어 프랑스어 영어 ▾ 번역하기

옛날에 백조 한 마리가 살았습니다.



19/5000

▶ 🔍 🎧 ▾

The 100,000,000,000,001 lived long ago. ✓

수정 제안하기

yesnal-e baegjo han maliga sal-assseubnida.

# Image Captioning



a group of people standing around a room with remotes  
logprob: -9.17



a young boy is holding a baseball bat  
logprob: -7.61



a cow is standing in the middle of a street  
logprob: -8.84



a baby laying on a bed with a stuffed bear  
logprob: -8.66



a young boy is holding a baseball bat  
logprob: -7.65



a woman holding a teddy bear in front of a mirror  
logprob: -9.65

# Visual QnA

Q: What is the boy holding?



DPPnet: **surfboard**

Q: What is the animal doing?



DPPnet: **resting** (relaxing)



DPPnet: **bat**



DPPnet: **swimming** (fishing)

# Auto Speech Recognition



# Text to Speech

- <https://post.naver.com/viewer/postView.nhn?volumeNo=12607569&memberNo=34653588>
- <https://carpedm2o.github.io/tacotron/>
- <https://youtu.be/klnfWhPGPRs>



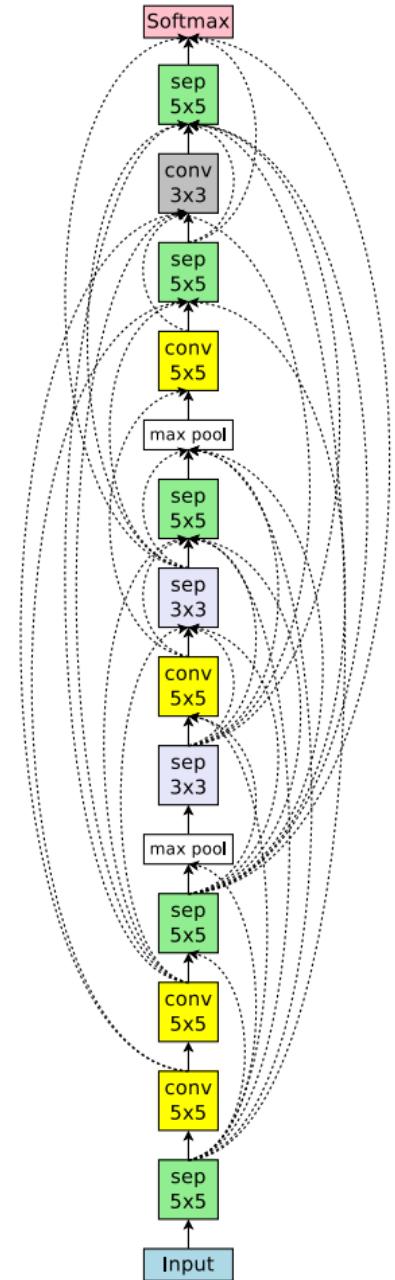
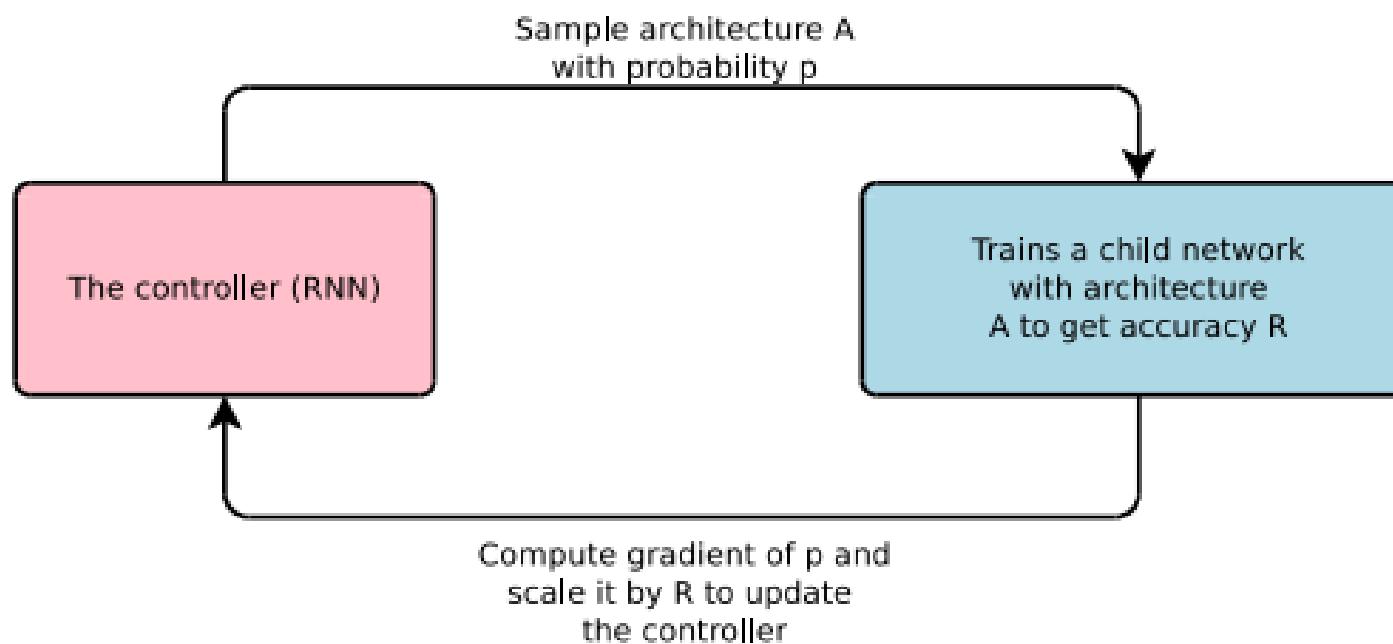
## Multi-Speaker Tacotron in TensorFlow

TensorFlow implementation of Deep Voice 2: Multi-Speaker Neural Text-to-Speech.

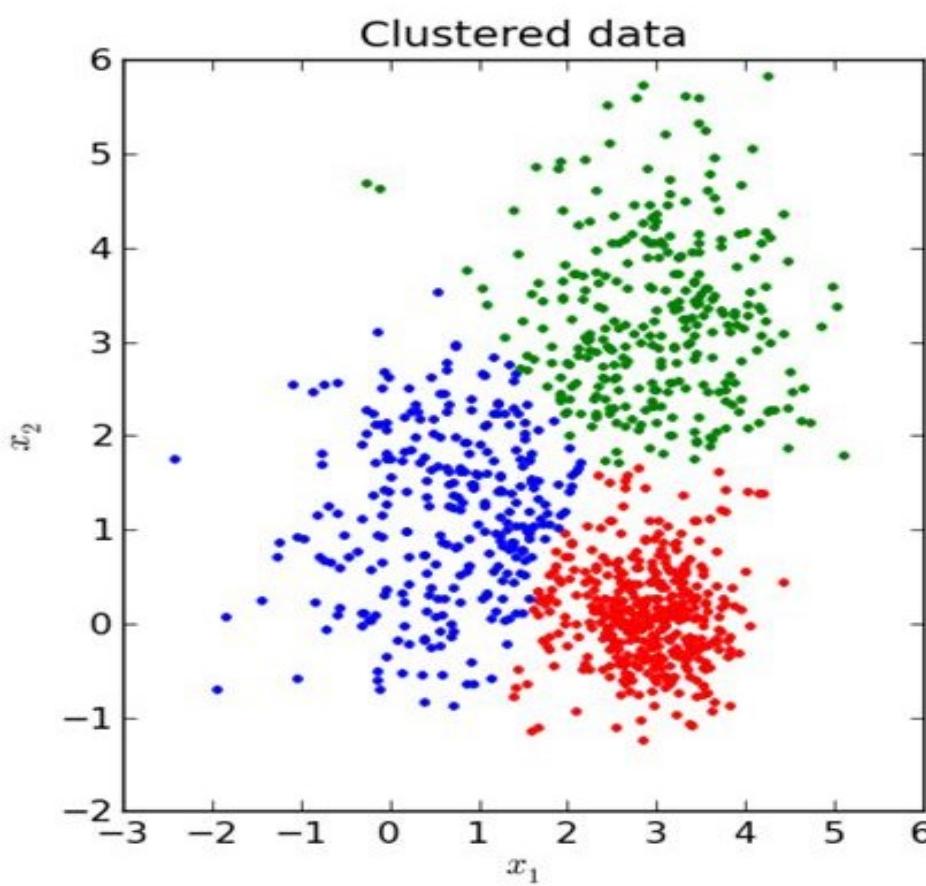
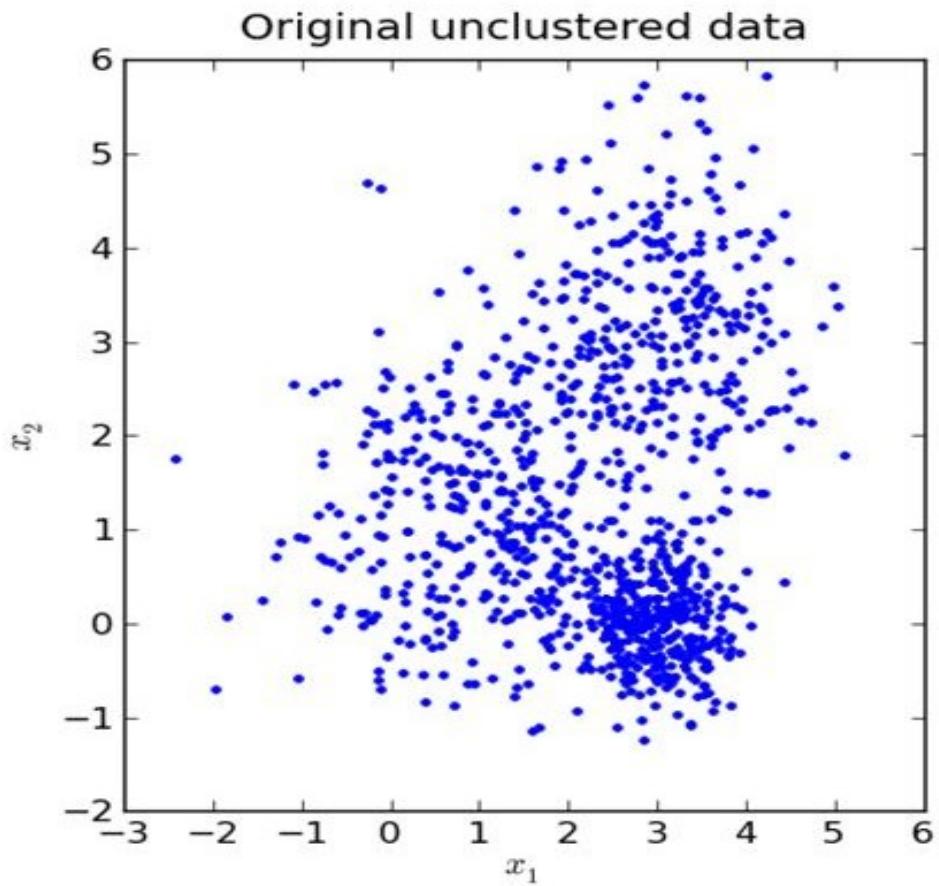
Taehoon Kim  
carpedm20

# Auto ML – Neural Architecture Search

- Learning to Learn



# Unsupervised Learning



# Unsupervised Learning

- 아래 음식을 둘로 분류하면?



(1)



(2)



(3)



(4)



(5)



(6)



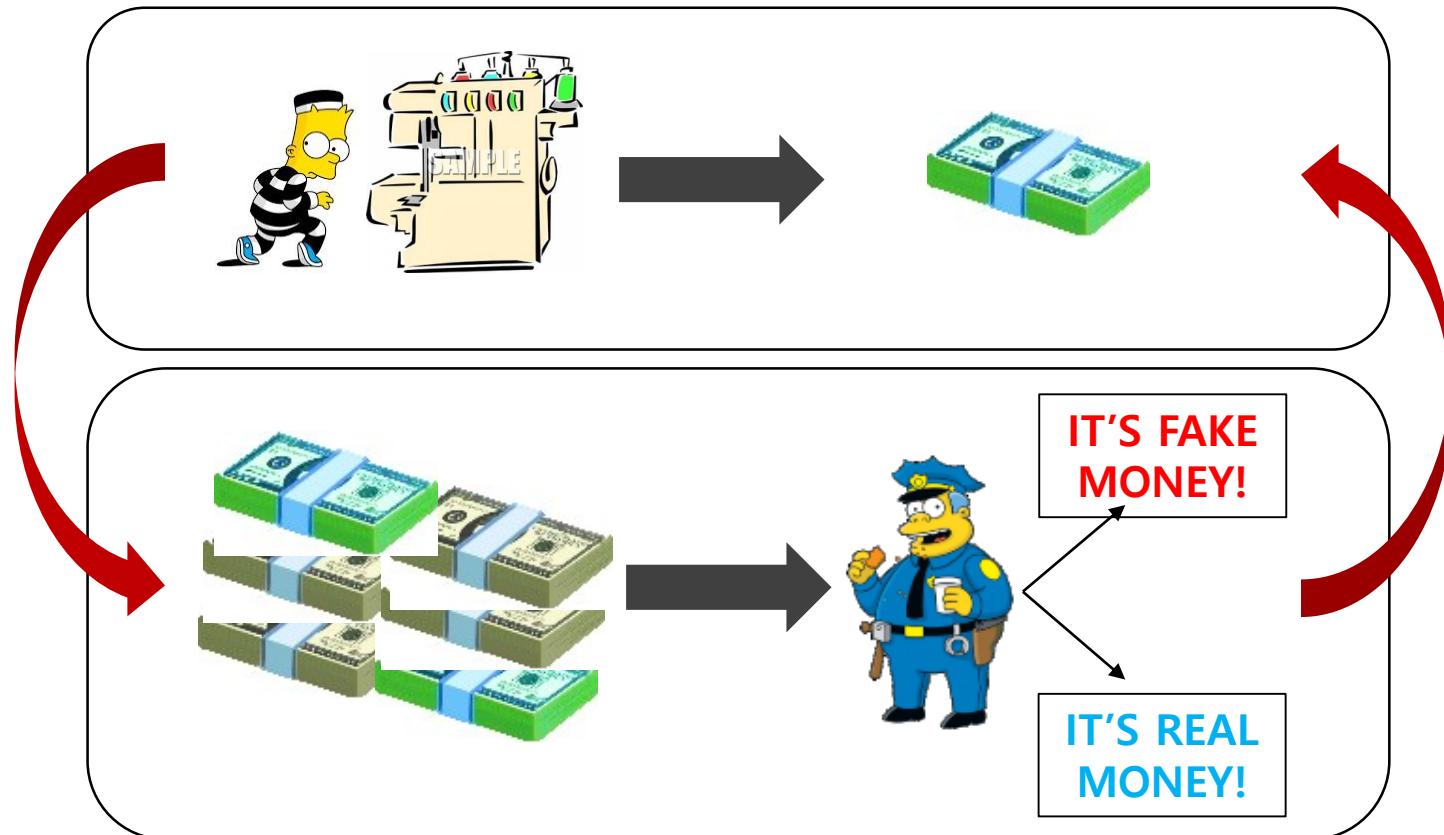
(7)



(8)

# Generative Adversarial Network

- Counterfeitors vs Police Game



# DCGAN



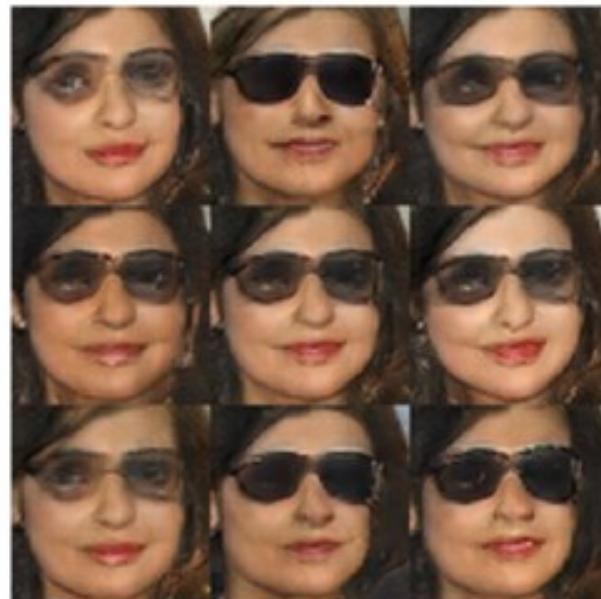
man  
with glasses



man  
without glasses



woman  
without glasses



woman with glasses

# Text to Image Generation

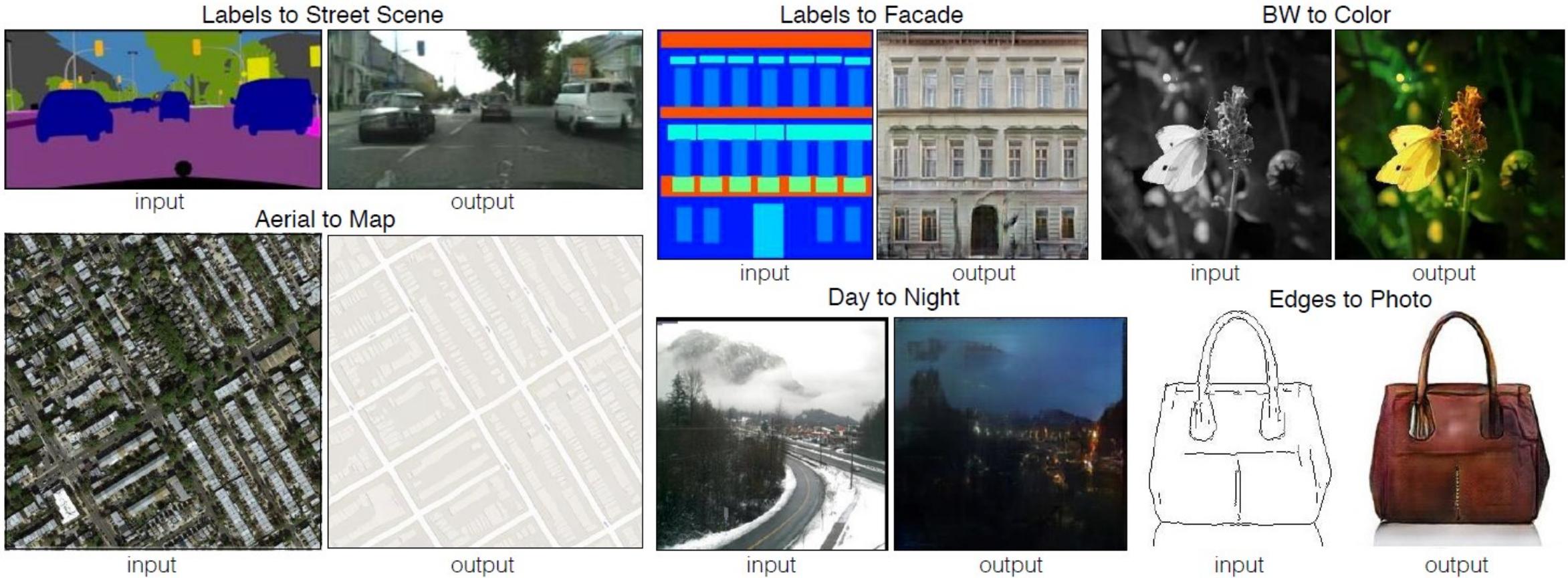
Text description	This bird is red and brown in color, with a stubby beak	The bird is short and stubby with yellow on its body	A bird with a medium orange bill white body gray wings and webbed feet	This small black bird has a short, slightly curved bill and long legs	with varying shades of brown with white under the eyes	bird with a black crown and a short black pointed beak	has a white breast, light grey head, and black wings and tail
64x64 GAN-INT-CLS [22]							
128x128 GAWWN [20]							
256x256 StackGAN							

# Domain Transfer

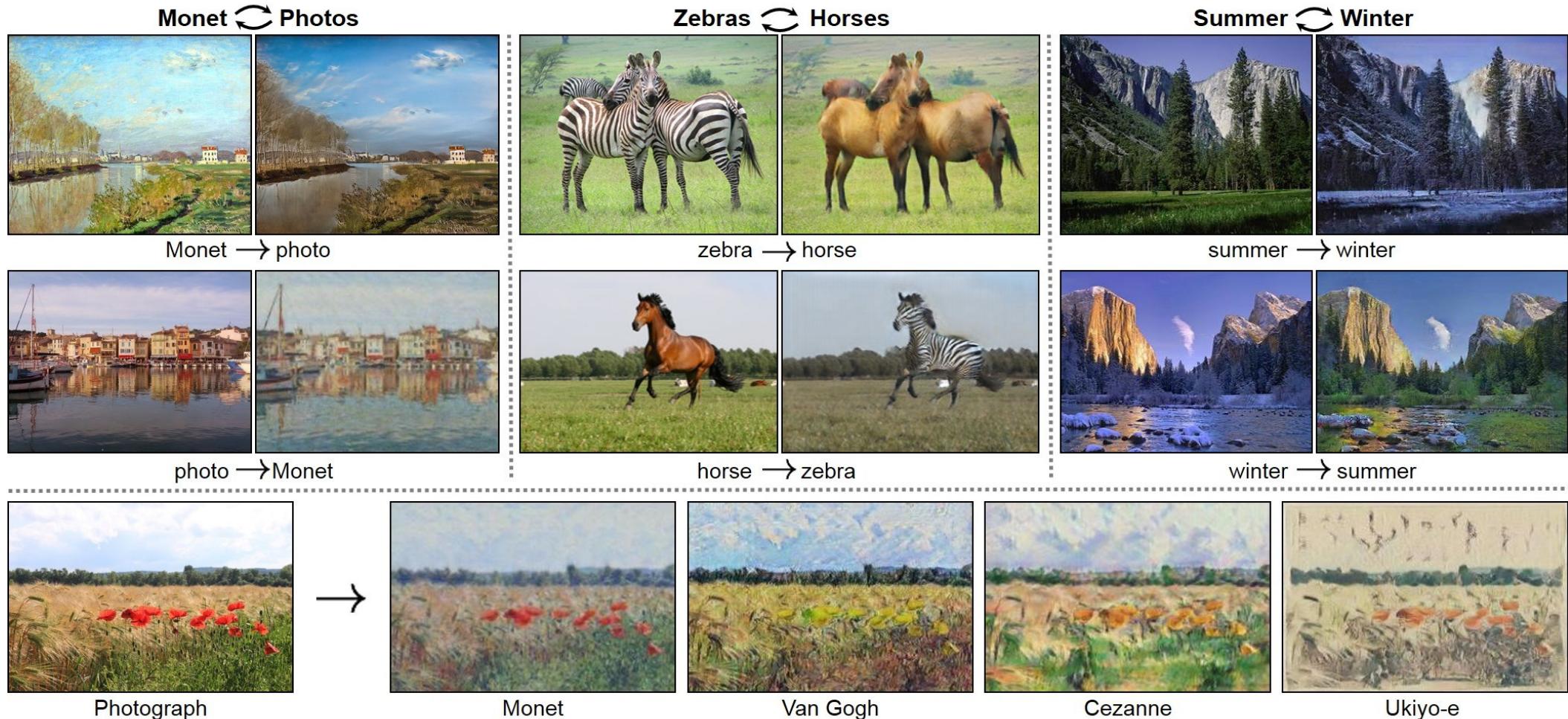


# Pix2Pix

- <https://affinelayer.com/pixsrv/>



# CycleGAN



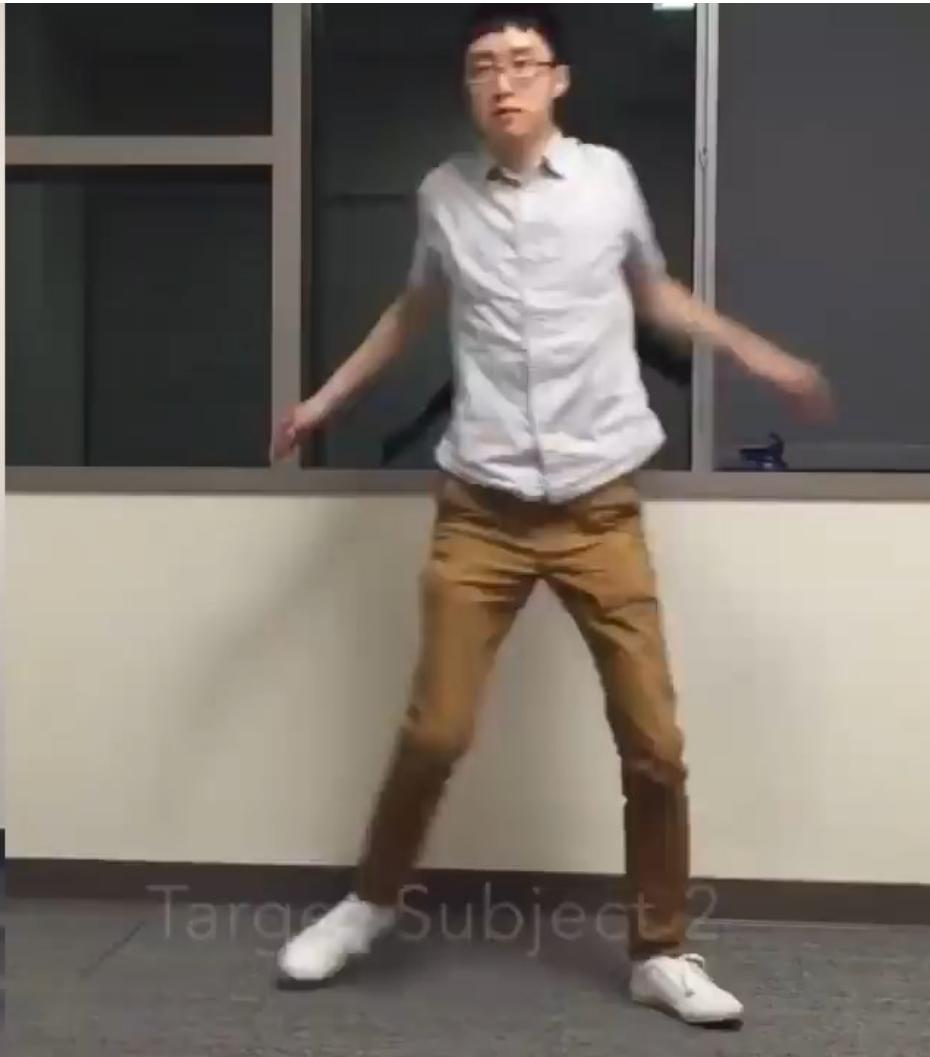
# Video-to-Video Synthesis

## Video-to-Video Synthesis

Ting-Chun Wang<sup>1</sup>, Ming-Yu Liu<sup>1</sup>, Jun-Yan Zhu<sup>2</sup>, Guilin Liu<sup>1</sup>,  
Andrew Tao<sup>1</sup>, Jan Kautz<sup>1</sup>, Bryan Catanzaro<sup>1</sup>

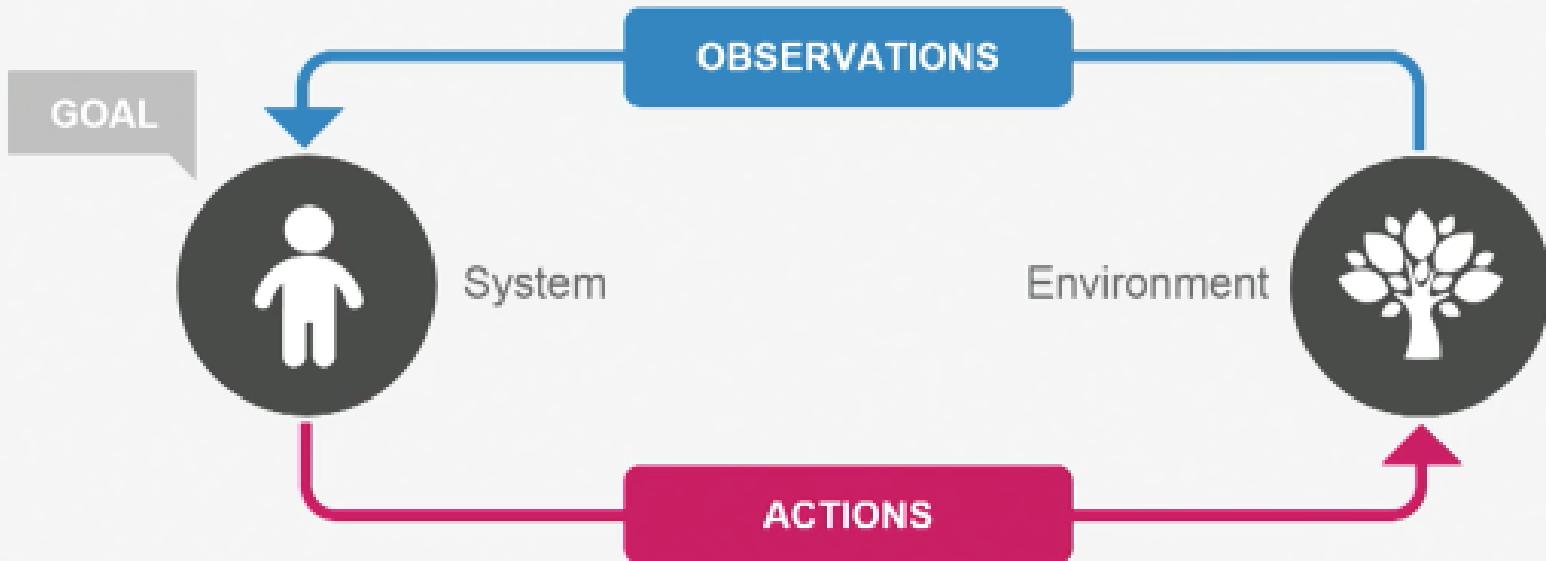
<sup>1</sup>NVIDIA Corporation    <sup>2</sup>MIT

# Video Generation



# Reinforcement Learning

## Reinforcement Learning Framework

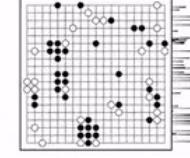


# Reinforcement Learning

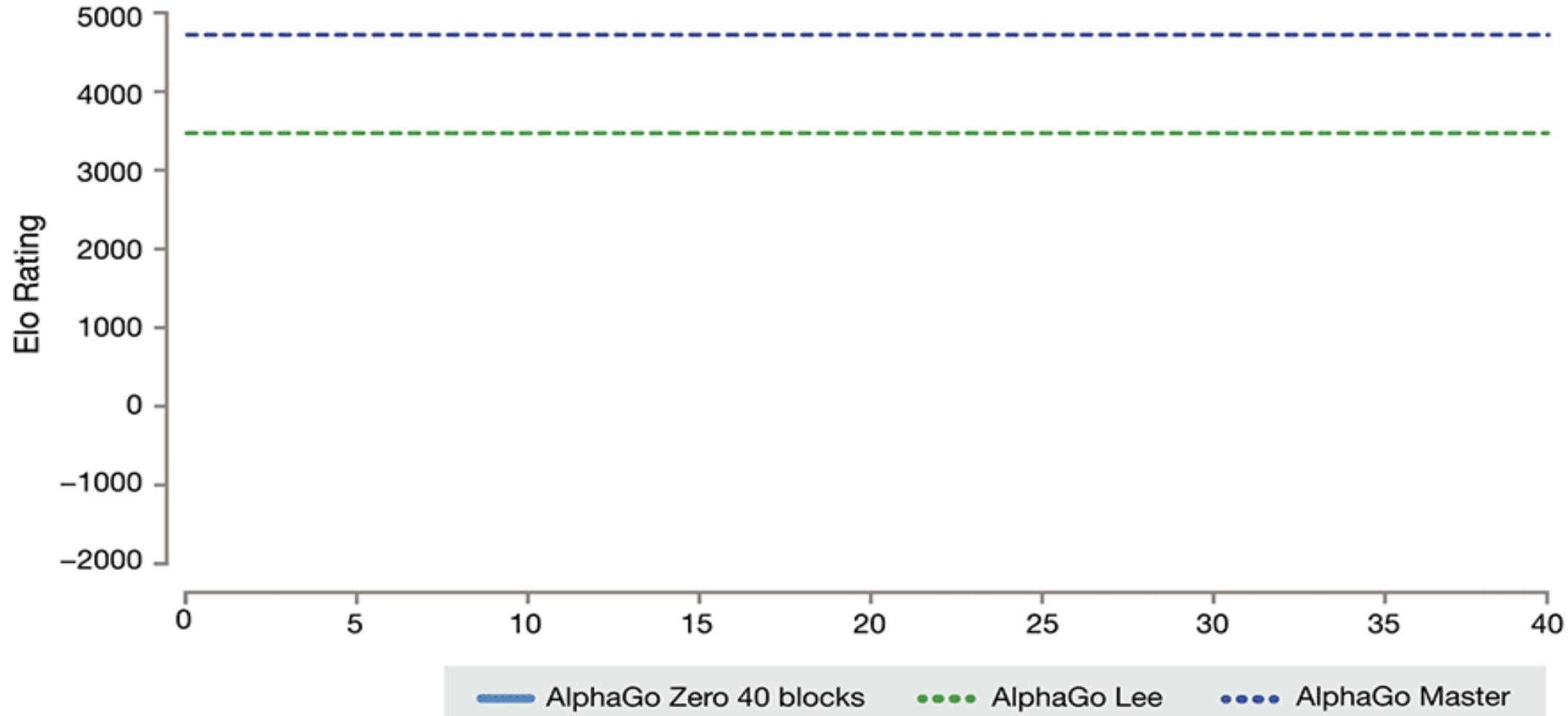


# AlphaGo

- 왜 바둑인가?
  - 사람이 만든 게임 중 가장 복잡하다
    - 경우의 수  $10^{170} \sim 10^{700}$
  - 형세판단을 하기 힘들다
    - 바둑은 점수 계산방식이 따로 없고 최종적으로 집의 수로 승패를 결정
    - 알파고와 이세돌 대결 때 2차전까지는 해설하는 프로기사들 간에도 유불리에 대한 의견이 분분하였음

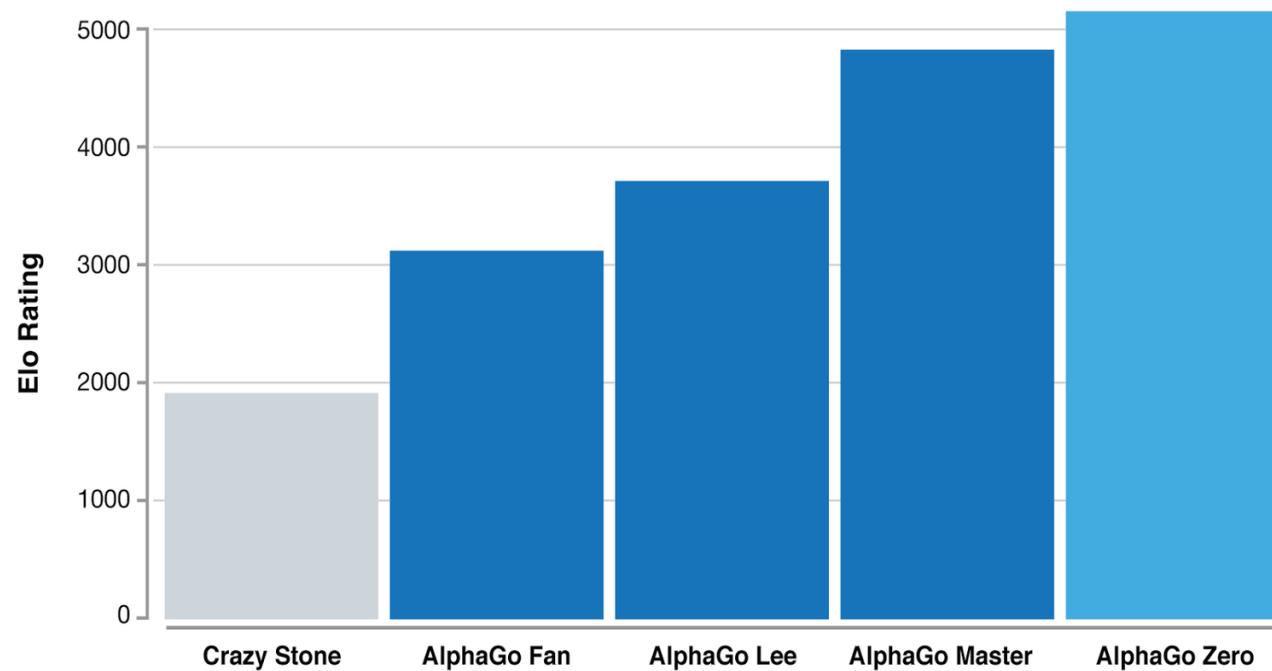
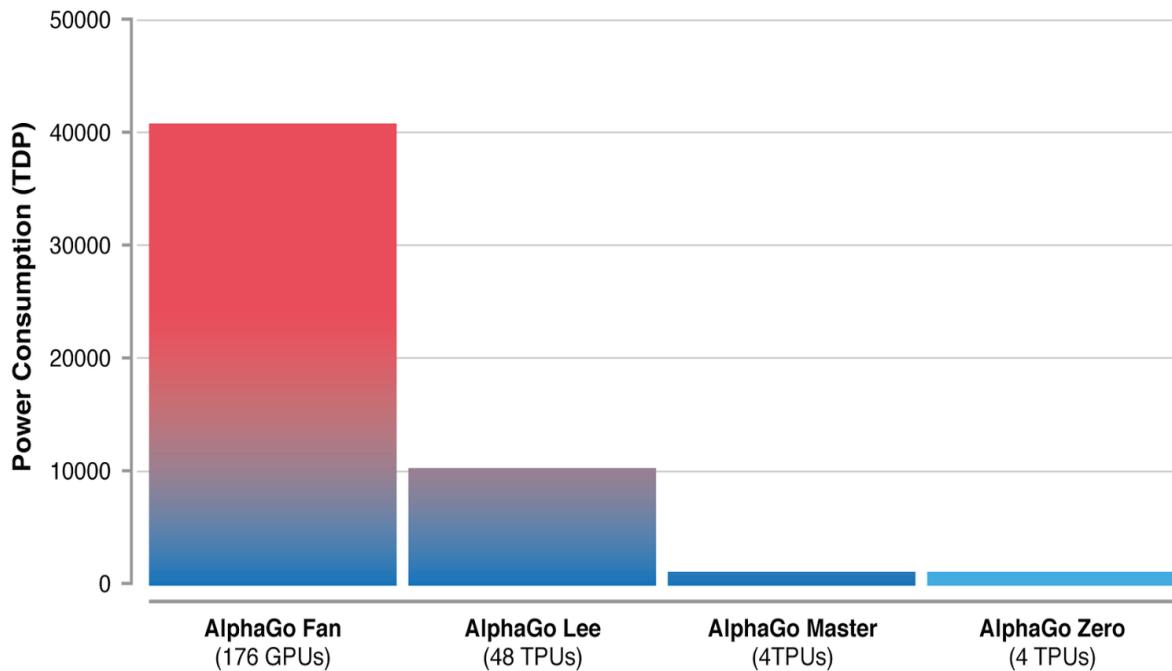


# AlphaGo Zero

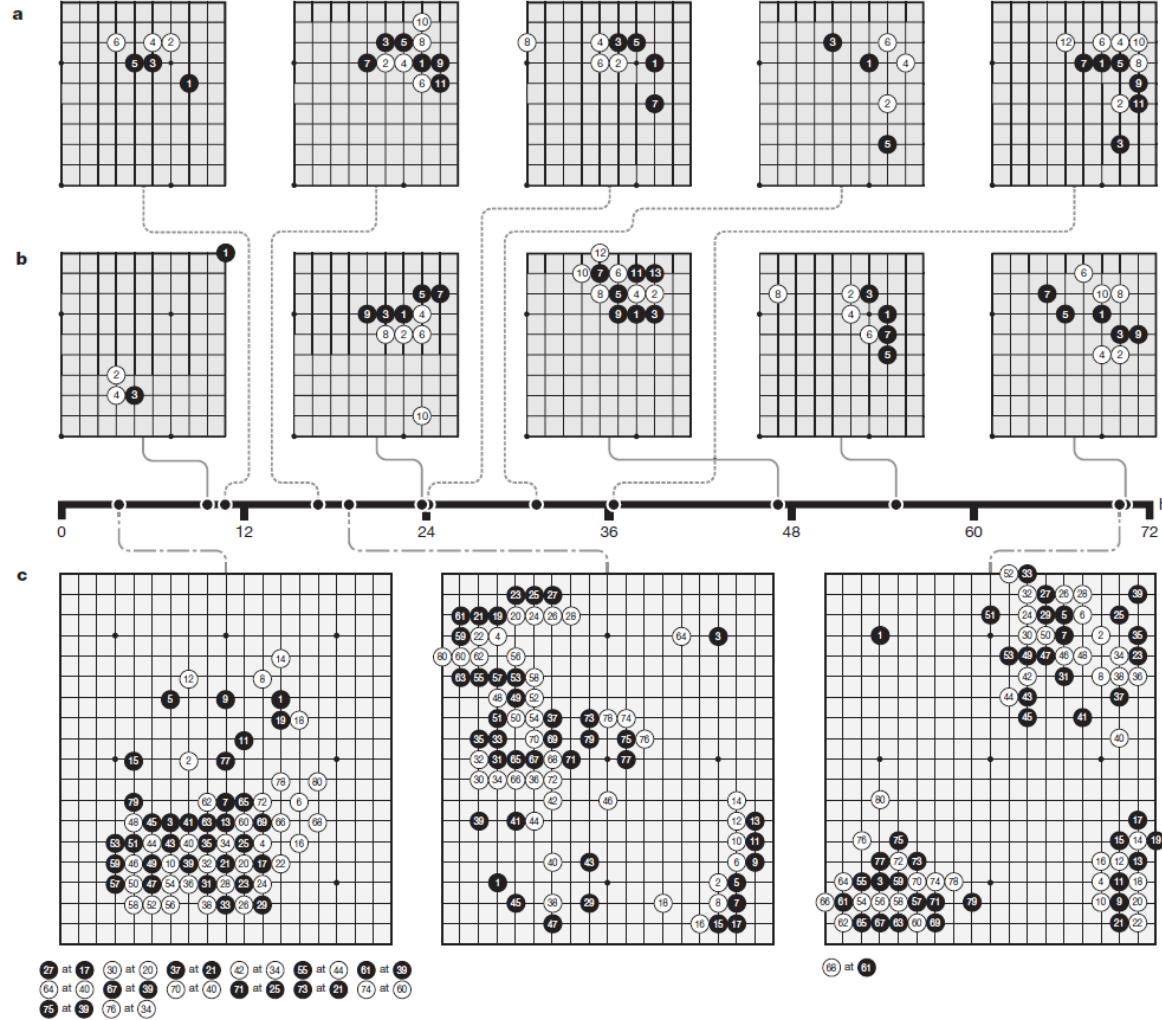


# AlphaGo Zero

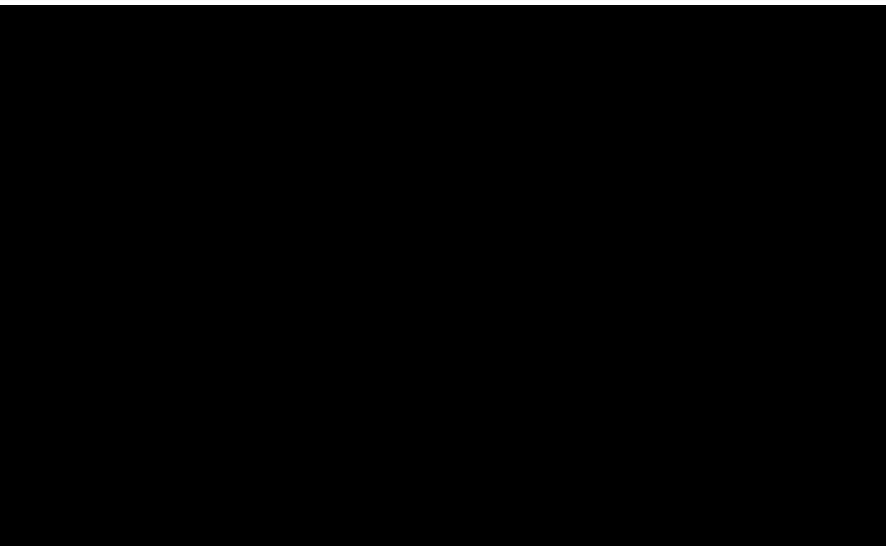
- Tabula rasa
- Without handcrafted input feature
- Rollout is removed



# AlphaGo Zero



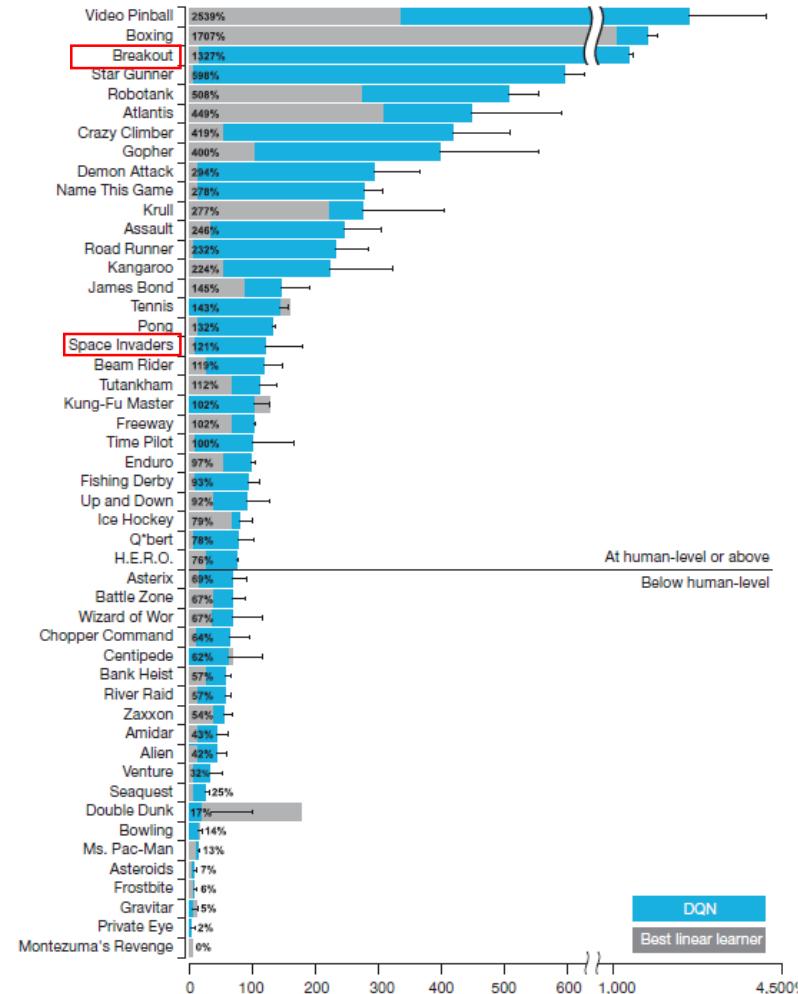
# Atari Games



Space Invaders



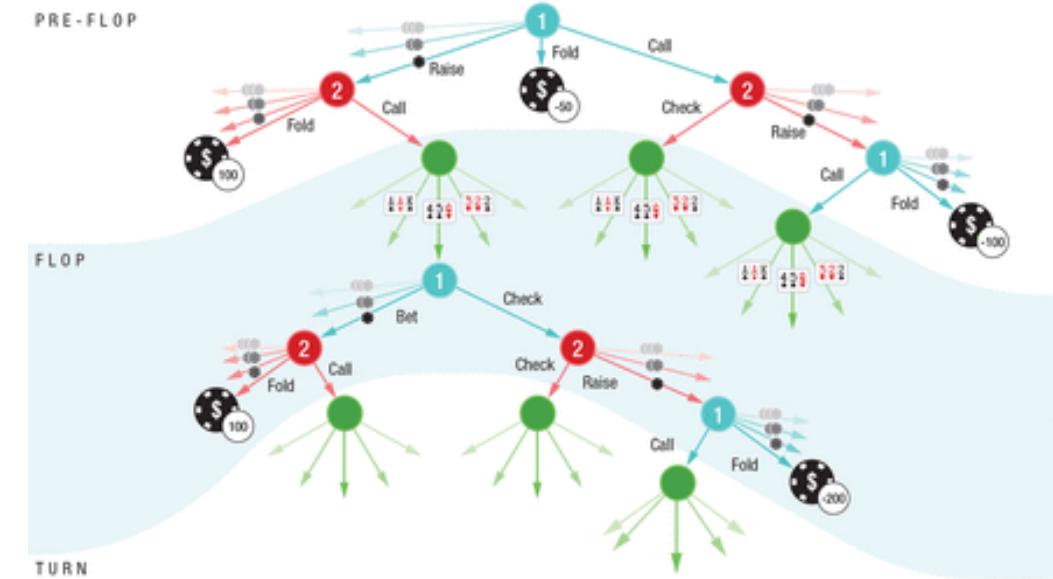
<https://youtu.be/SucculnpiDo>



# Poker Game

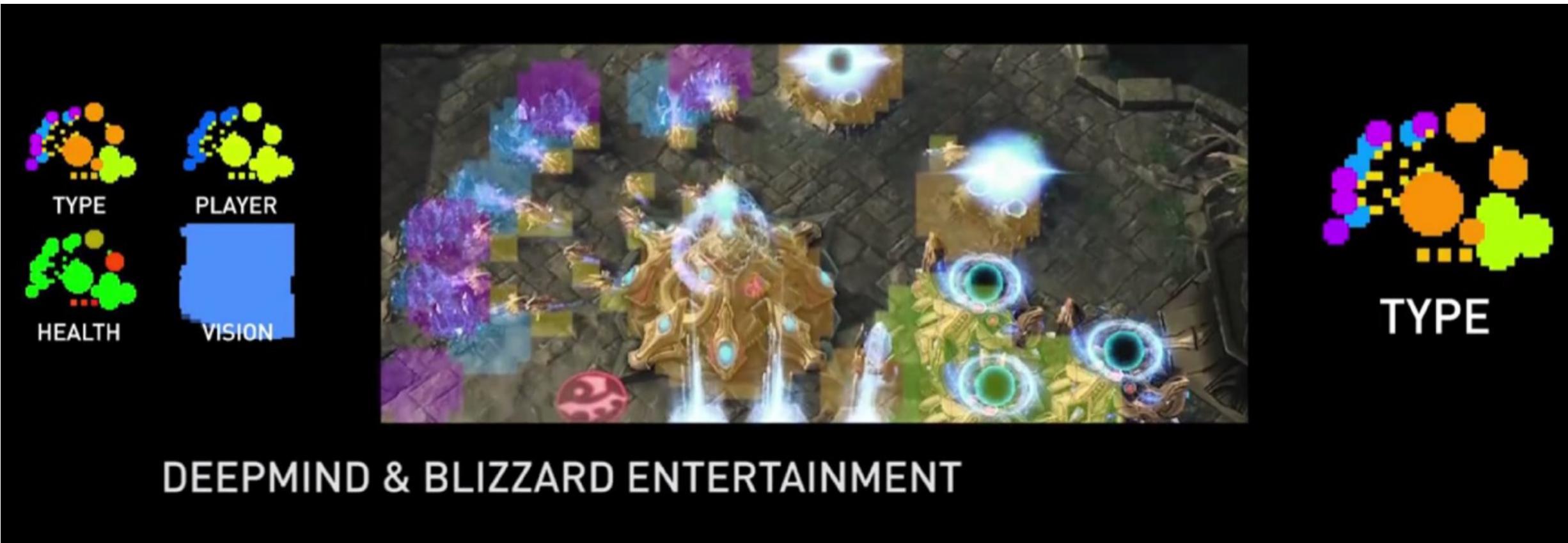


Libratus(Jan 30, 2017)



DeepStack(Science, Mar 02, 2017)

# Starcraft II



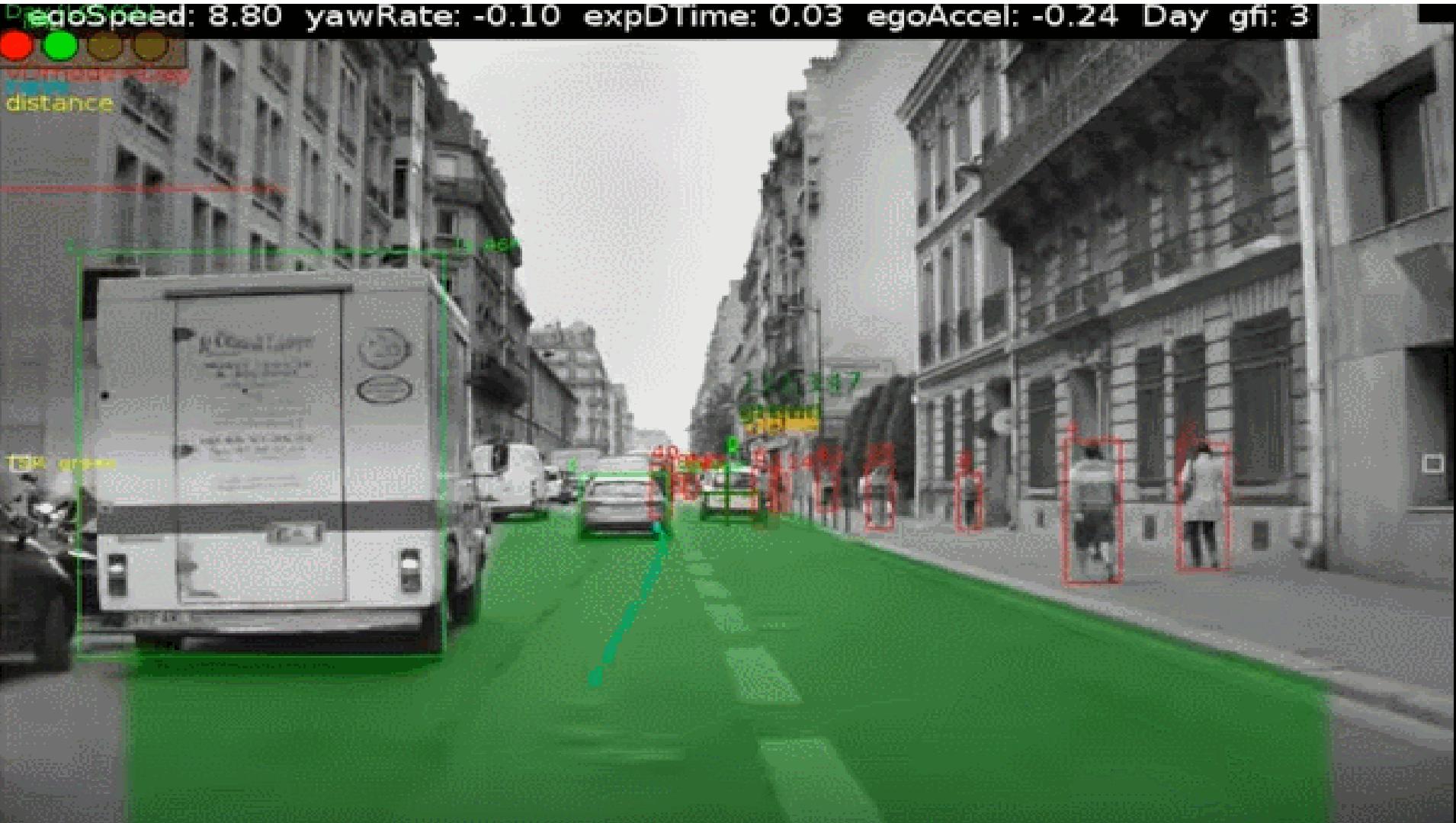
<https://youtu.be/St5lxIxYGkl>

<https://arxiv.org/abs/1708.04782>

# Other Games

- Super Mario World
  - [https://youtu.be/L4KBBAwF\\_bE](https://youtu.be/L4KBBAwF_bE)
  - <https://youtu.be/qv6UVOQoF44>
- Cookie Run
  - <https://youtu.be/exXD6wJLJ6s>
- Starcraft I
  - <https://youtu.be/GgkmJDjeJtw>
- GTA
  - <https://youtu.be/X4u2DCOLoIg>

# Autonomous Driving



# Quiz

- □와 △에 들어갈 정수는?
  - $3 \times \square + 2 \times \triangle = 1$
  - $1 \times \square + 4 \times \triangle = -3$
  - $5 \times \square + 5 \times \triangle = 0$
  - $8 \times \square + 3 \times \triangle = 5$
- $\square = 1, \triangle = -1$
- $(3, 2), (1, 4), (5, 5), (8, 3)$  은 input data,  $1, -3, 0, 5$  는 label이다
- □와 △를 weight라고 하며 이 weight 값을 기계가 스스로 학습을 통해 찾아내도록 하는 것이 neural network를 이용한 기계학습이 하는 일

# Machine Learning Again

- Learning from data!
- Essence
  - Data
  - Pattern in the data
  - Difficult to explain / understand mathematically

# Problem Formulation

- Running example : credit approval

► **given:**

- applicant information

feature	value
age	23 years
gender	female
annual salary	\$30,000
years in residence	1 year
years in job	1 year
current debt	\$15,000

► **task:**

- approve credit or not?

# Notation

component	symbol	credit approval metaphor
input	$\mathbf{x}$	customer application
output	$y$	approve or deny
target function	$f : \mathcal{X} \rightarrow \mathcal{Y}$	ideal credit approval formula
data	$(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)$	historical records
hypothesis	$g : \mathcal{X} \rightarrow \mathcal{Y}$	formula to be used

- ▶  $f$ : unknown target function
- ▶  $\mathcal{X}$ : input space (set of all possible inputs  $\mathbf{x}$ )
- ▶  $\mathcal{Y}$ : output space (set of all possible outputs)
- ▶  $N$ : the number of input-output examples (*i.e.* training examples)
- ▶  $\mathcal{D} \triangleq \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)\}$ : data set where  $y_n = f(\mathbf{x}_n)$

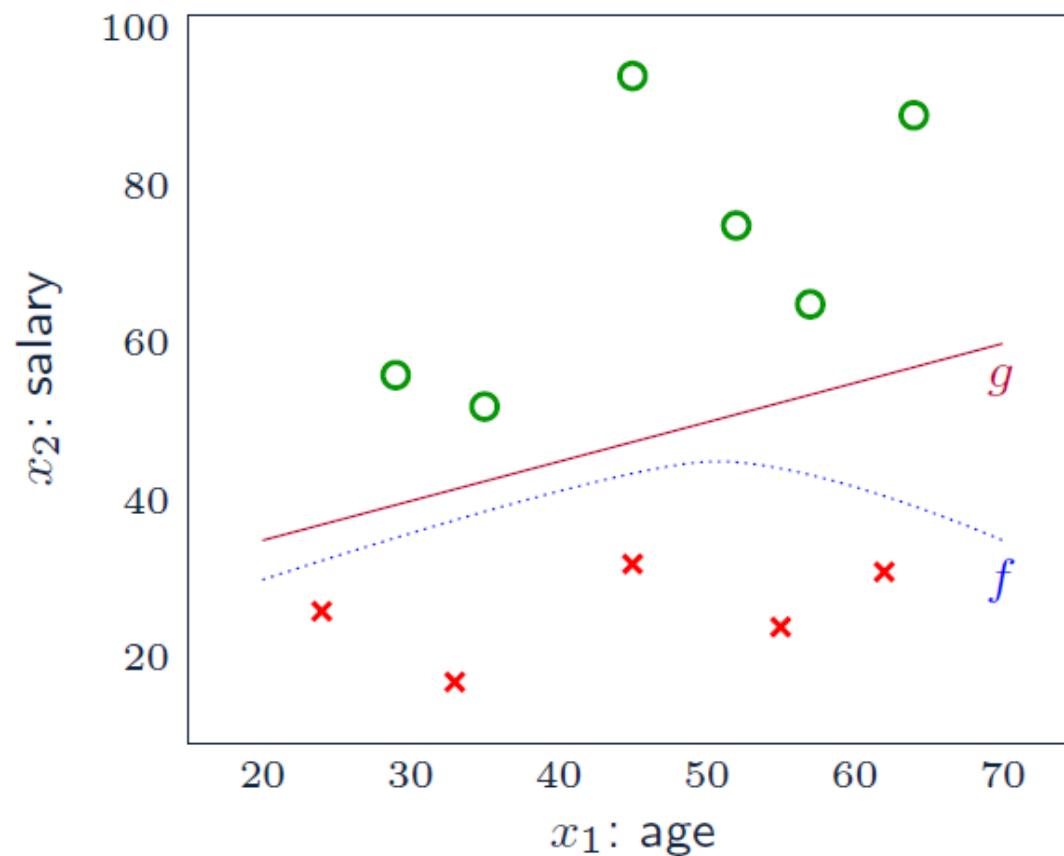
# Example – Credit Approval

$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$  where  $x_1$ : age and  $x_2$ : annual salary in USD

$N = 11$ ,  $d = 2$ ,  $\mathcal{X} = \mathbb{R}^2$ , and  $\mathcal{Y} = \{\text{approve}, \text{deny}\}$

data set  $\mathcal{D}$ :

$n$	$x_1$	$x_2$	$y$
1	29	56k	approve
2	64	89k	approve
3	33	17k	deny
4	45	94k	approve
5	24	26k	deny
6	55	24k	deny
7	35	52k	approve
8	57	65k	approve
9	45	32k	deny
10	52	75k	approve
11	62	31k	deny



# Components of Learning

- ▶ **learning algorithm**  $\mathcal{A}$

- uses  $\mathcal{D}$  to pick formula  $g : \mathcal{X} \rightarrow \mathcal{Y}$  that approximates  $f$
- chooses  $g$  from a set  $\mathcal{H}$

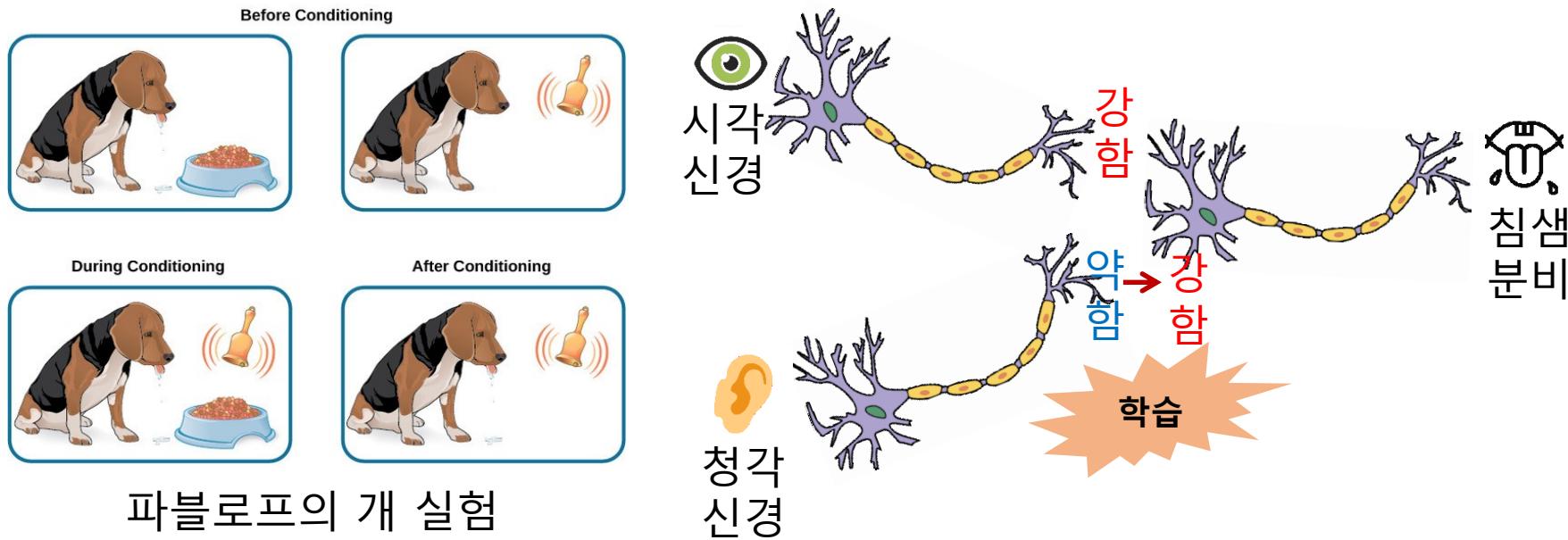
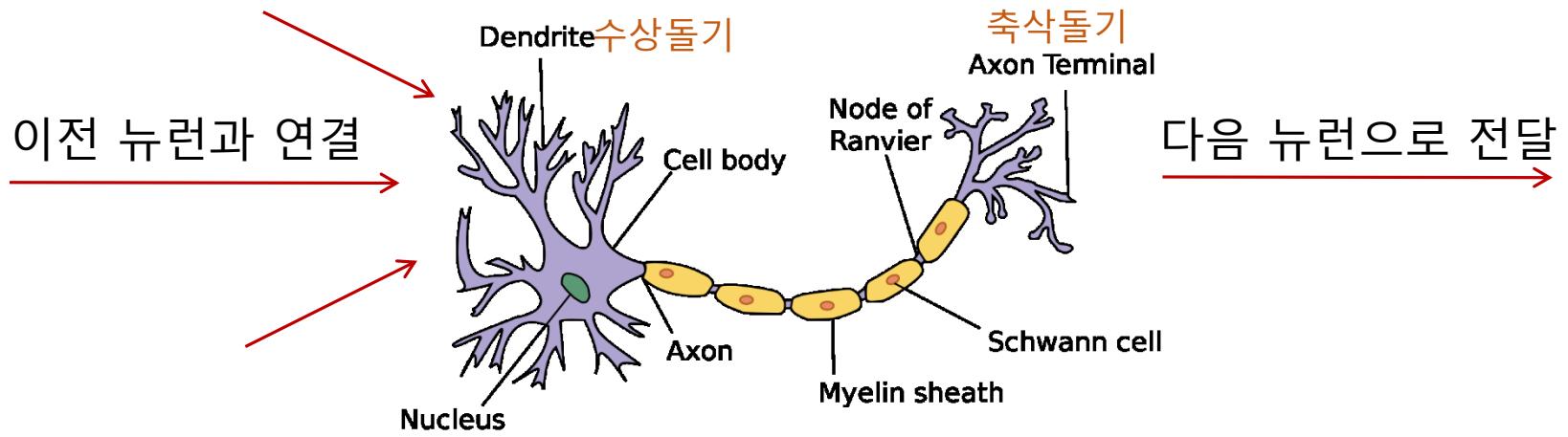
- ▶ **hypothesis set**  $\mathcal{H}$

- a set of candidate formula under consideration
- e.g.  $\mathcal{H}$  could be the set of all linear formulas

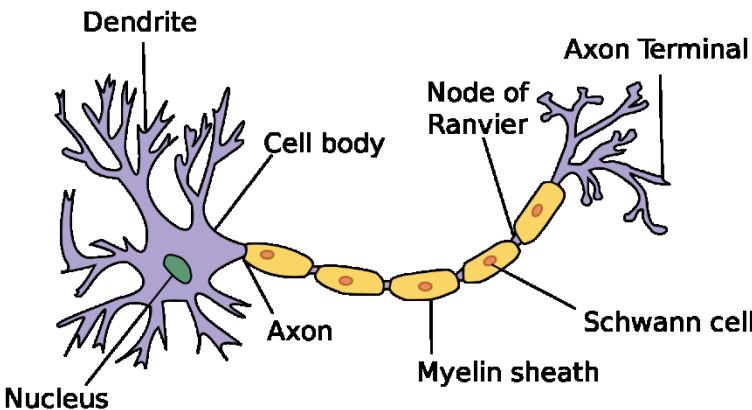
- ▶ **learning model:  $\mathcal{A}$  and  $\mathcal{H}$**

- two solution tools we should choose

# 뉴런과 사람의 학습



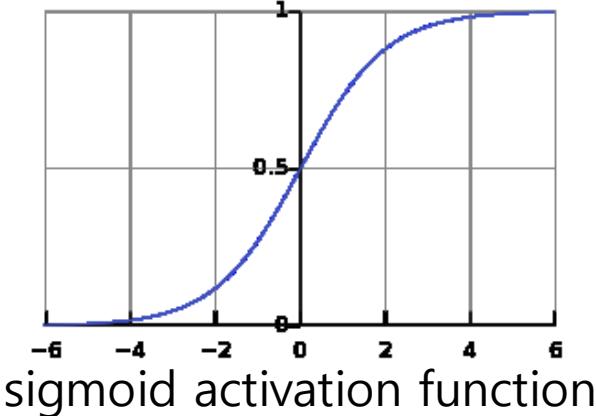
# Perceptron(Artificial Neural Network)



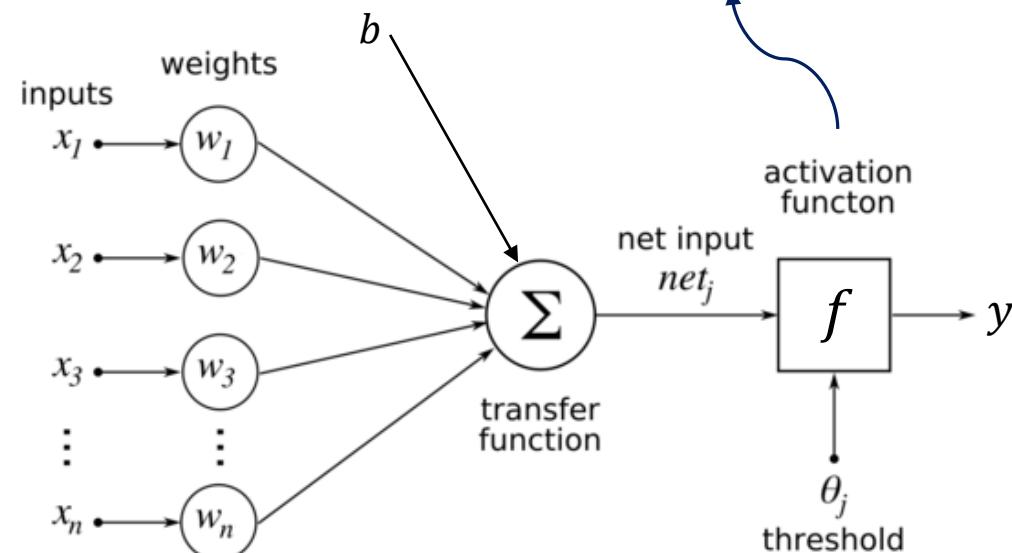
$$y = f(\mathbf{w}\mathbf{x} + b)$$

$$\mathbf{w} = [w_1 \ w_2 \ w_3 \ \dots \ w_n]$$

$$\mathbf{x} = [x_1 \ x_2 \ x_3 \ \dots \ x_n]^T$$

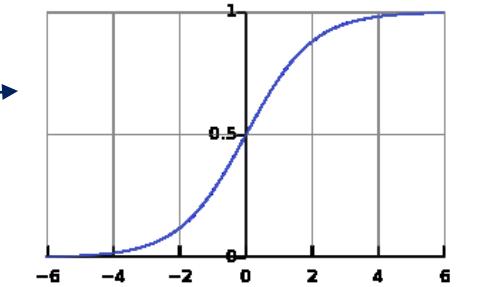
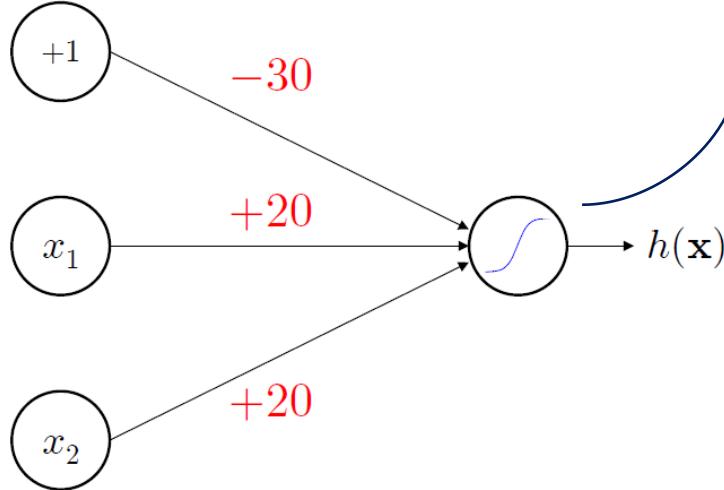


$$f(x) = \frac{1}{1 + e^{-x}}$$



<Perceptron>

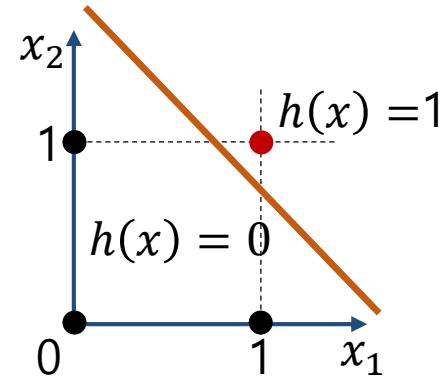
# Example of ANN(logical AND)



$x_1$	$x_2$	$h(\mathbf{x})$
0	0	$\sigma(-30) \approx 0$
0	1	$\sigma(-10) \approx 0$
1	0	$\sigma(-10) \approx 0$
1	1	$\sigma(10) \approx 1$

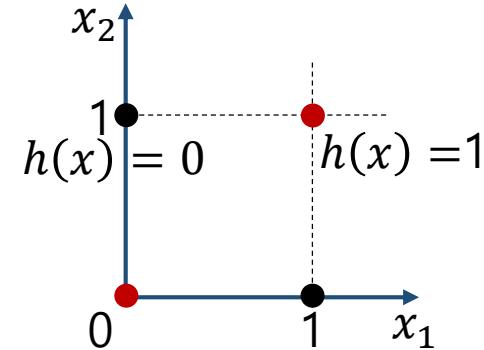
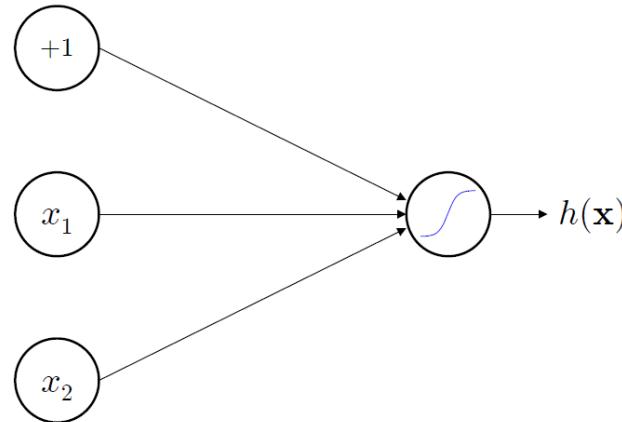
$$h(\mathbf{x}) = \sigma(-30 + 20x_1 + 20x_2)$$

학습이란 이러한 weight 값(-30, 20, 20)을  
기계 스스로 찾을 수 있도록 해주는 과정!

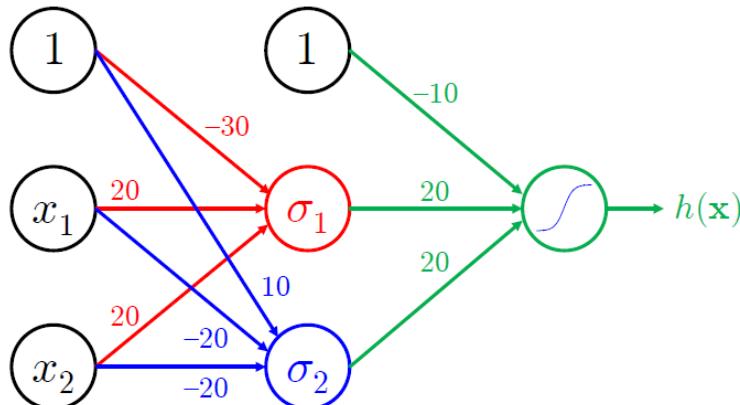


# Logical XNOR with Perceptron

- Is it possible(Linearly separable)?

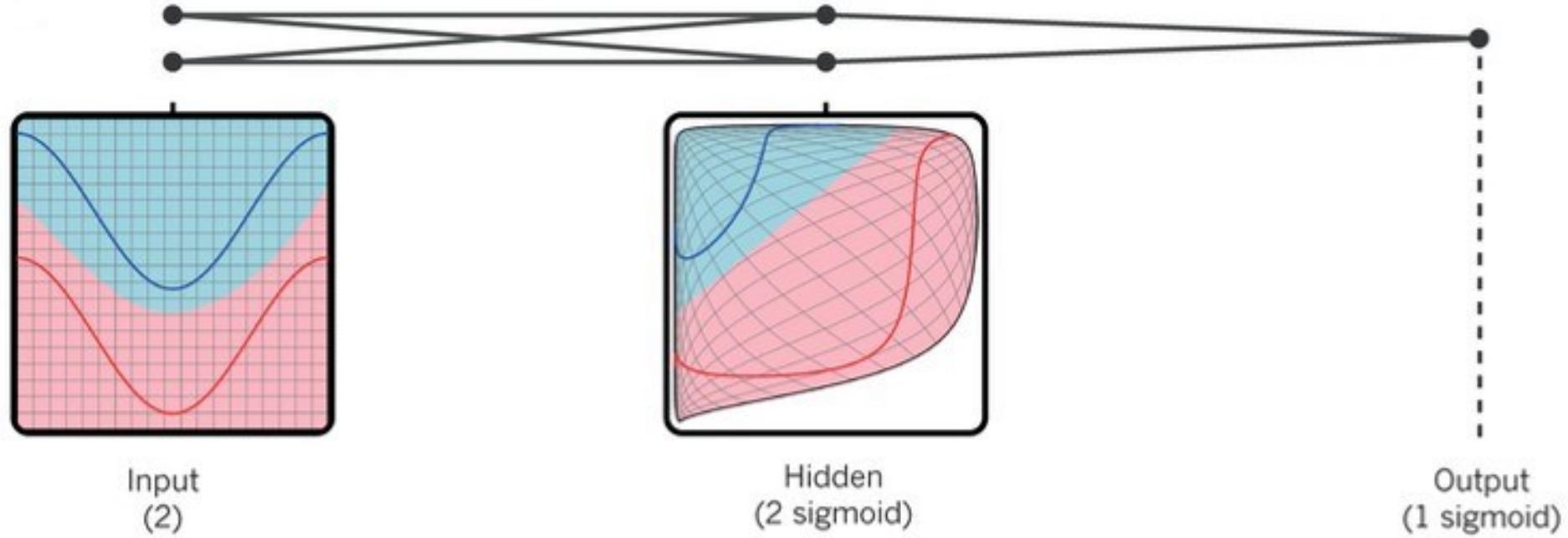


- We need more perceptrons and more layers



$x_1$	$x_2$	$\sigma_1$	$\sigma_2$	$h(\mathbf{x})$
0	0	0	1	1
0	1	0	0	0
1	0	0	0	0
1	1	1	0	1

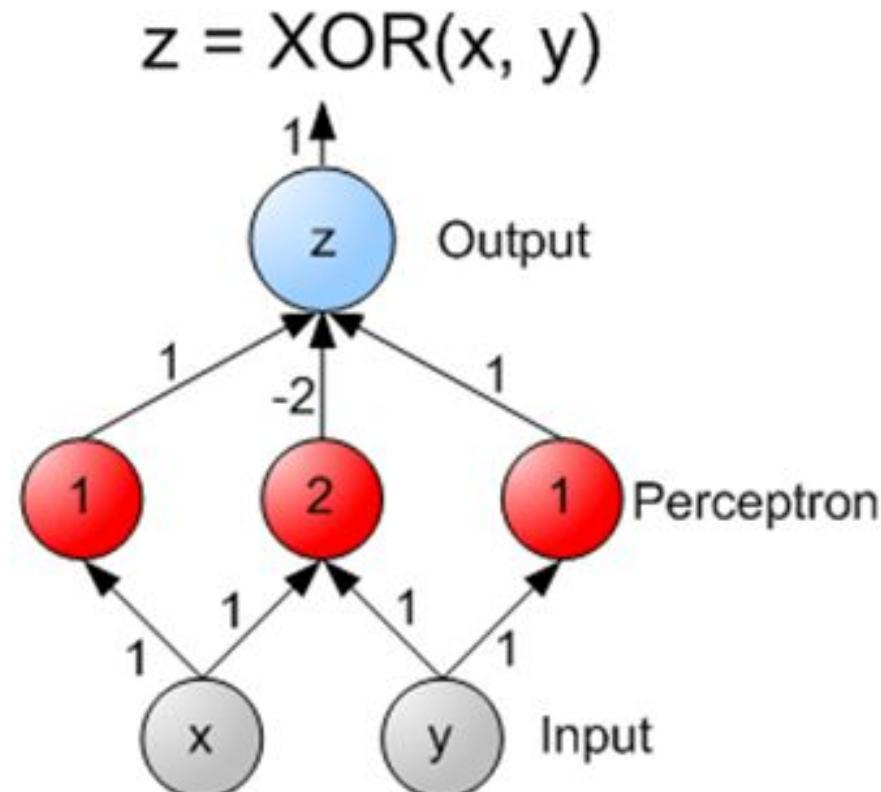
# Multi Layer Perceptron(MLP)



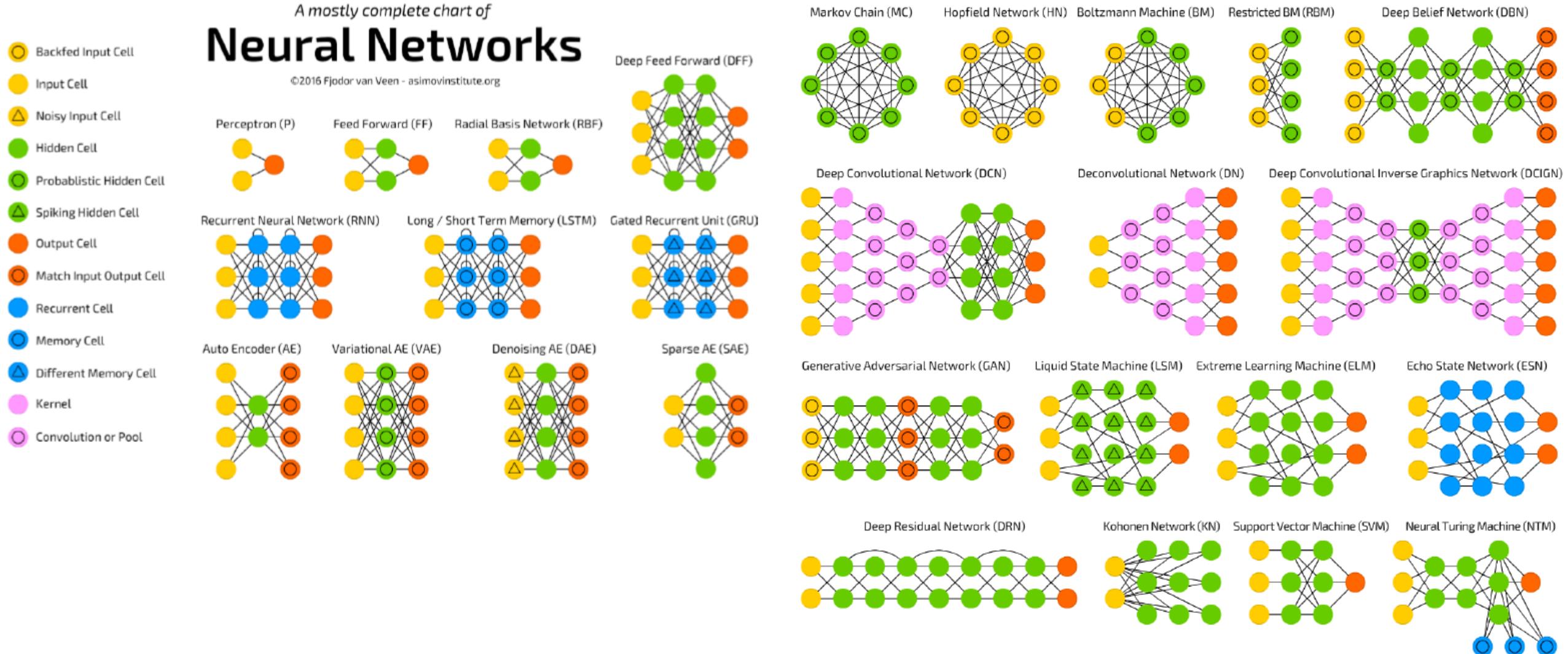
- 선을 잘 긋고 input 공간을 잘 왜곡하고 합하는 과정을 반복해서 데이터들을 잘 구분 해보자(classification)
- 이렇게 perceptron을 여러층으로 쌓으면 더 복잡한 문제를 풀 수 있다
- Linear fitting과 Non-linear transform의 반복

# Universal Function Approximation

The universal approximation theorem for neural networks states that every continuous function that maps intervals of real numbers to some output interval of real numbers can be approximated arbitrarily closely by a multi-layer perceptron with just one hidden layer. This result holds only for restricted classes of activation functions, e.g. for the sigmoidal functions.  
(Wikipedia.org)

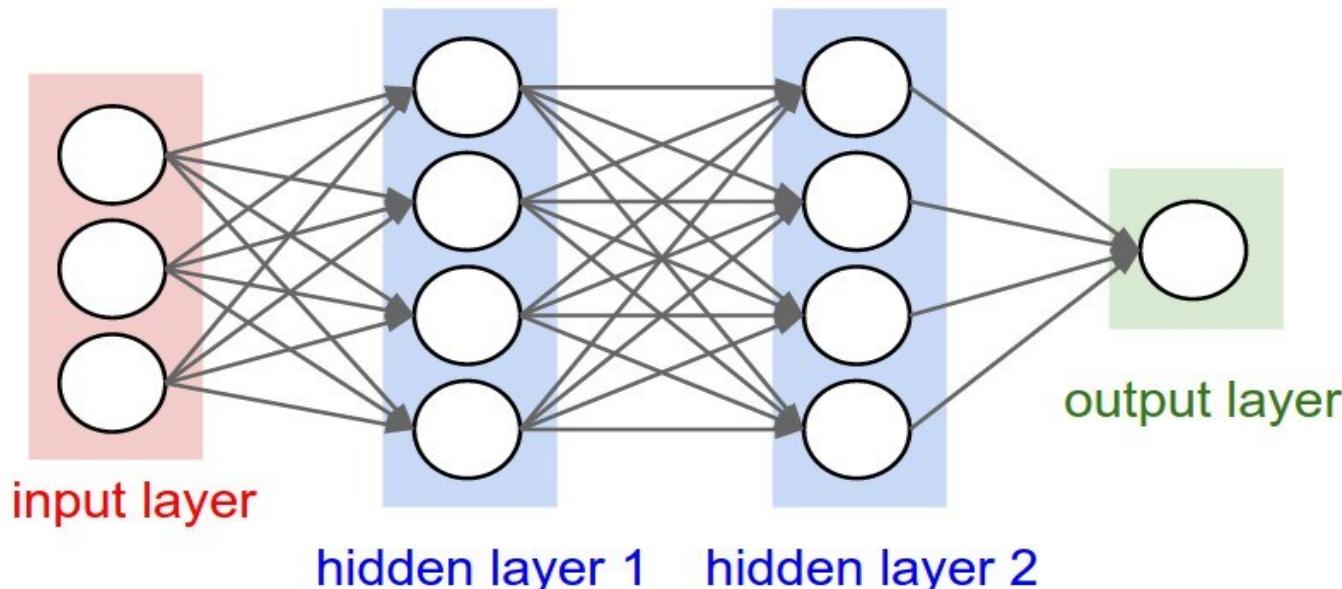


# Making Neural Networks



# 딥러닝(Deep Learning)

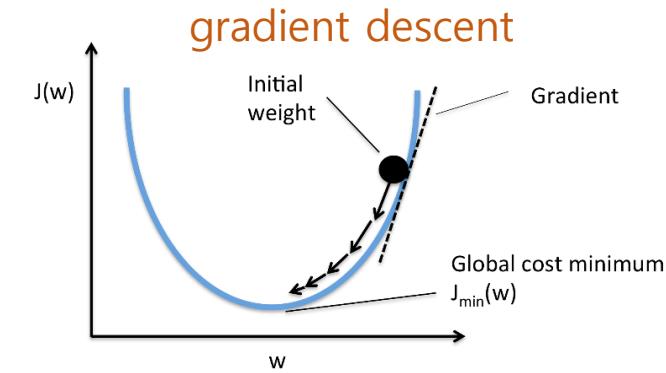
- 딥러닝은 deep neural network를 통하여 학습하는 것을 의미함
- Hidden layer의 수  $\leq 1 \rightarrow$  shallow network
- Hidden layer의 수  $\geq 2 \rightarrow$  deep network



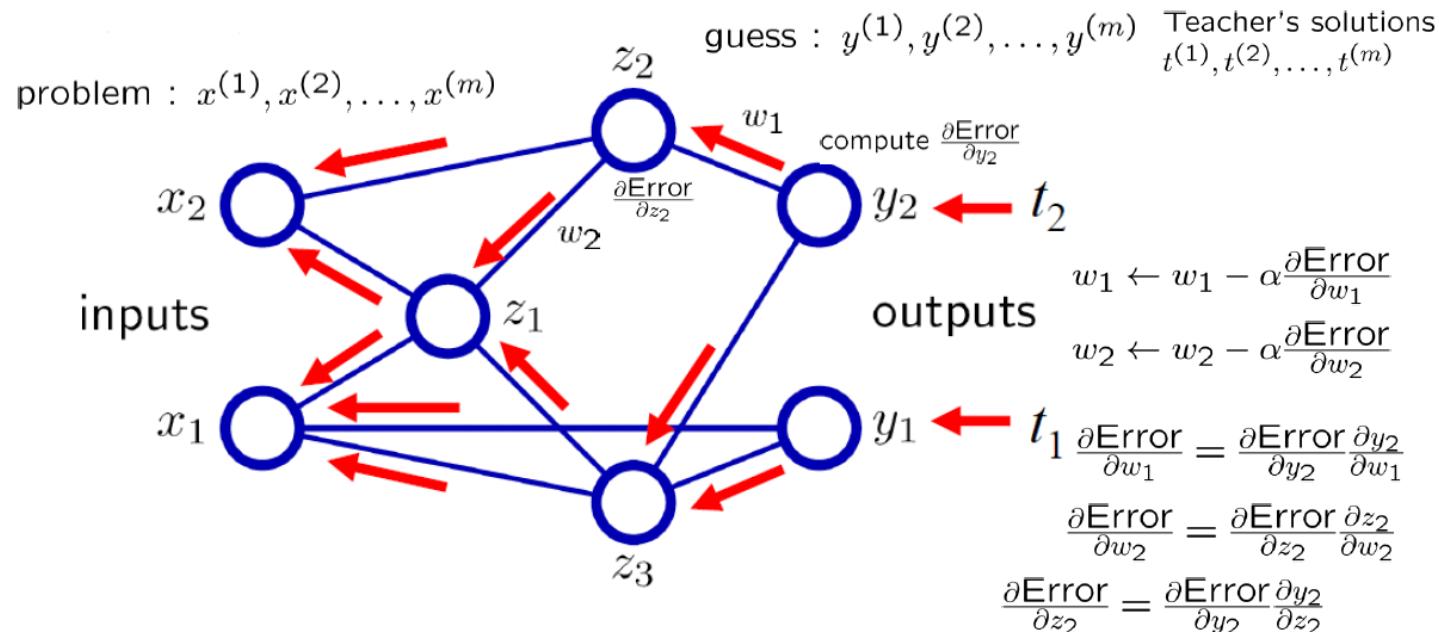
- 이렇게 많은 weight 값들을 어떻게 학습시킬 것인가??

# Back Propagation

- 학습과정 : back propagation of error
  - Output layer에서 error(cost)를 계산
  - Error의 미분값을 back propagation
  - 미분값에  $\alpha$ 를 곱한 만큼  $w$ 를 보정(학습!)
  - $\alpha$ 는 learning rate를 의미함



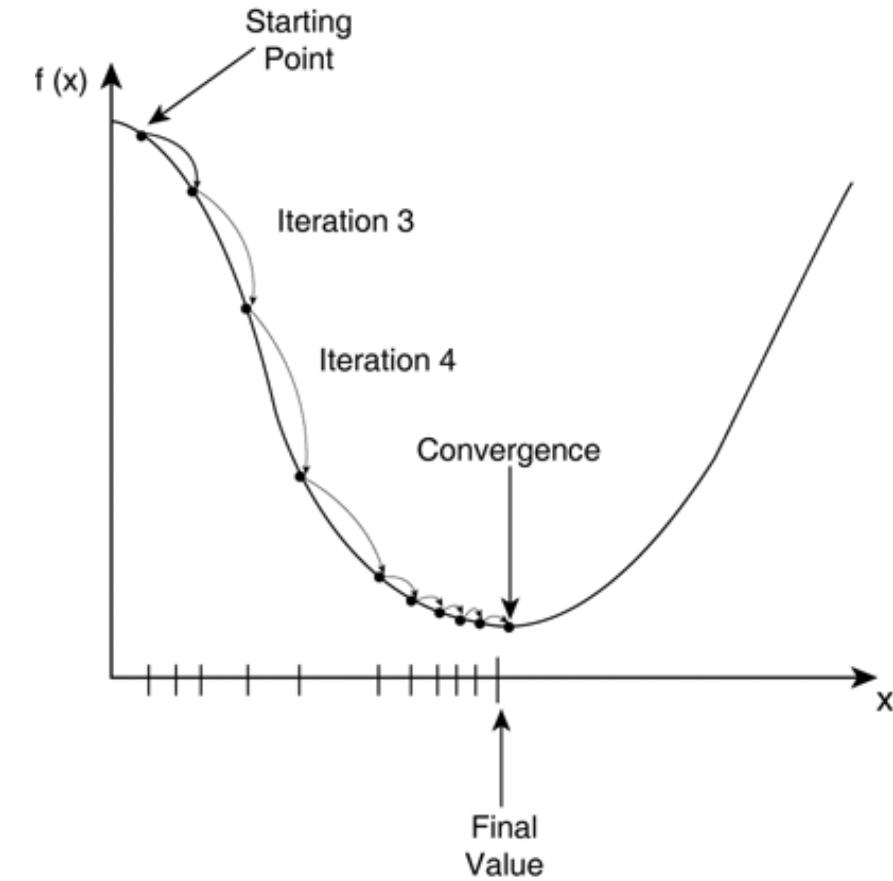
$$\text{minimize Error} = \sum_{l=1}^m (y^{(l)} - t^{(l)})^2$$



# Gradient Descent

- Loss Function 을  $W$ (parameter)로 편미분해서  $w$ 에 대한 Gradient를 구한다
- Gradient를 이용해서  $w$ 를 업데이트 한다

$$W^* = W - \alpha \text{Loss}'(W)$$

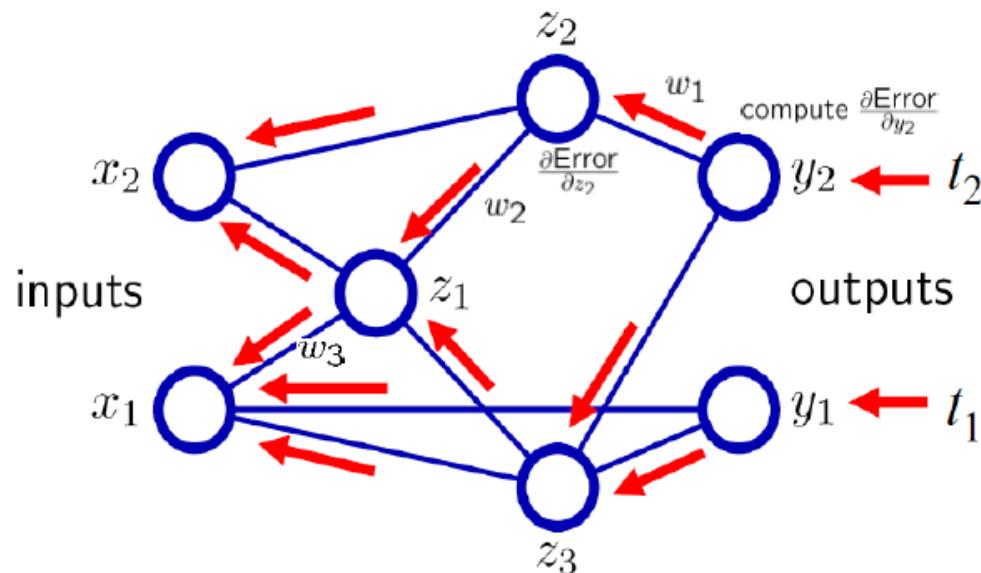
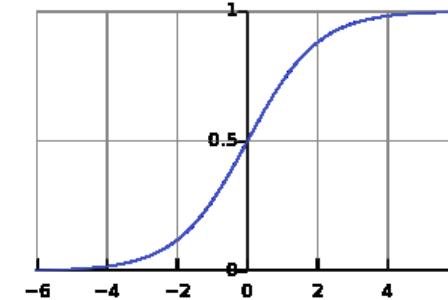


# 딥러닝을 어렵게 하는 것들

- Vanishing gradient problem
- Overfitting problem
- Get stuck in local minima

# Vanishing Gradient Problem

- Gradient 값이 뒤로 전달될 수록 점점 작아짐
- Sigmoid 사용으로 인하여(미분값의 최대 :  $\frac{1}{4}$ ) 아래쪽 layer는 학습이 이루어지지 않음



$$\frac{\partial \text{Error}}{\partial w_1} = \frac{\partial \text{Error}}{\partial y_2} \frac{\partial y_2}{\partial w_1} = \frac{\partial \text{Error}}{\partial y_2} \sigma'(\cdot) z_2$$

$$\frac{\partial y_2}{\partial w_1} = \sigma'(\cdot) z_2$$

$$\frac{\partial \text{Error}}{\partial w_2} = \frac{\partial \text{Error}}{\partial z_2} \frac{\partial z_2}{\partial w_2} = \frac{\partial \text{Error}}{\partial y_2} \sigma'(\cdot) w_1 \sigma'(\cdot) z_1$$

$$\frac{\partial z_2}{\partial w_2} = \sigma'(\cdot) z_1$$

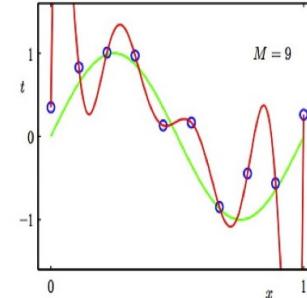
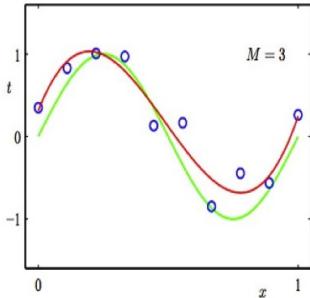
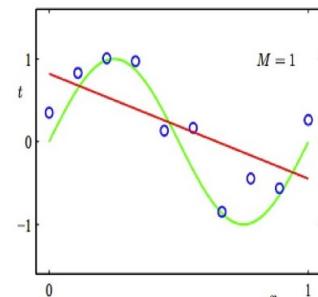
$$\frac{\partial \text{Error}}{\partial w_3} = \sigma'(\cdot) \sigma'(\cdot) \sigma'(\cdot) C$$

$$\sigma'(\cdot) \approx 0$$

# Overfitting Problem

- Data가 많지 않은 경우에 발생할 수 있음
- 학습한 data에만 최적화되어서, 학습하지 않은 data(test data)에 대한 추론 성능이 악화되는 현상

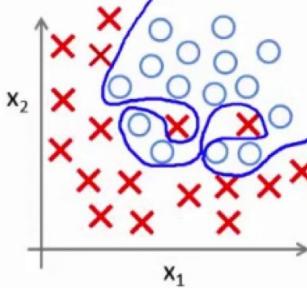
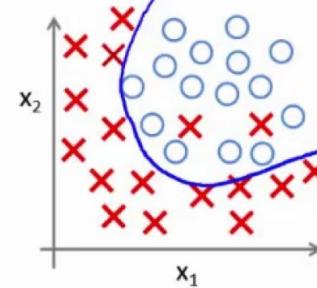
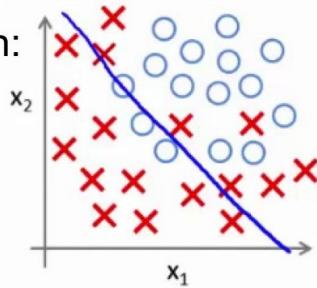
Regression:



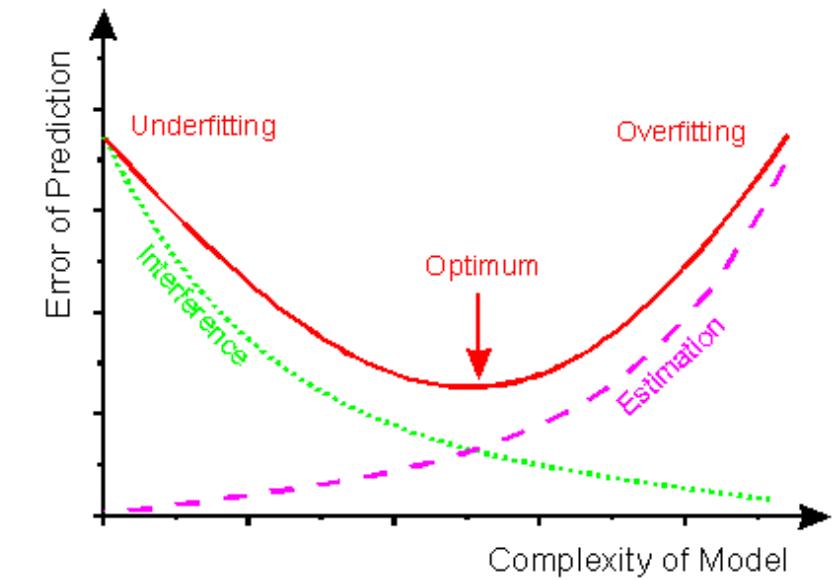
predictor too inflexible:  
cannot capture pattern

predictor too flexible:  
fits noise in the data

Classification:



Copyright © 2014 Victor Lavrenko

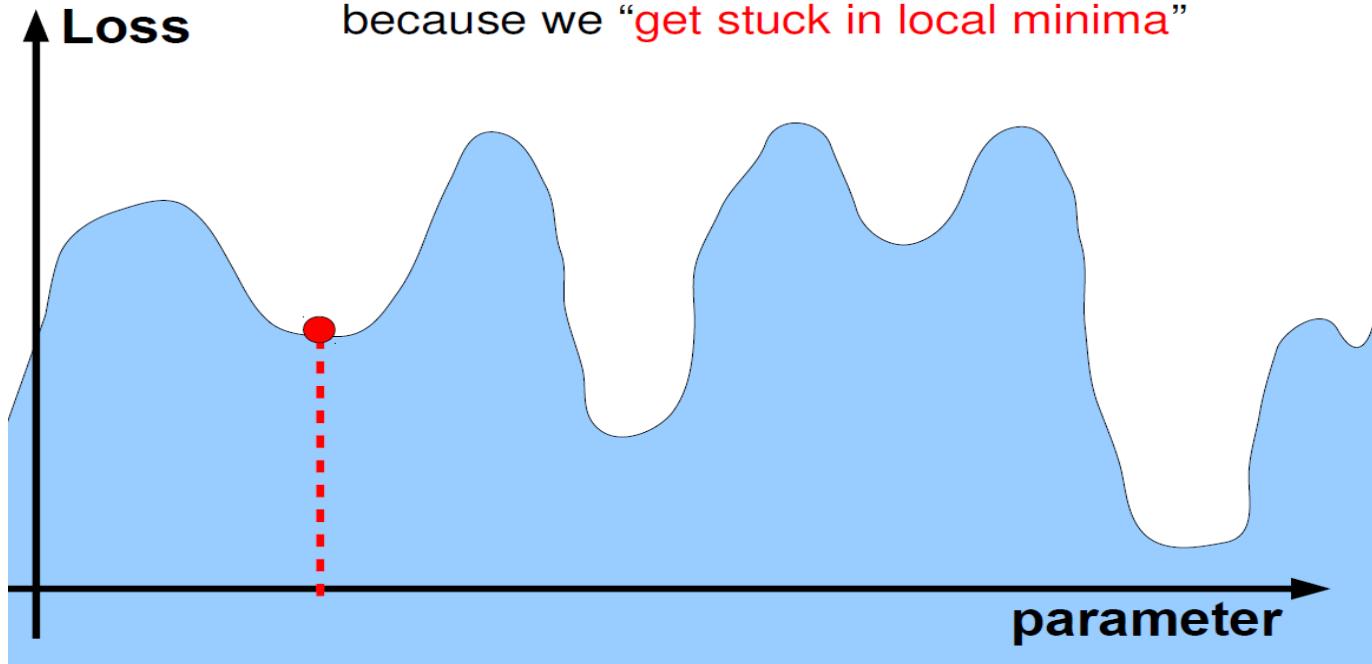


# Local Minima

- 어디서 시작하느냐에 따라서 잘못하면 local minima에 빠질 위험이 존재

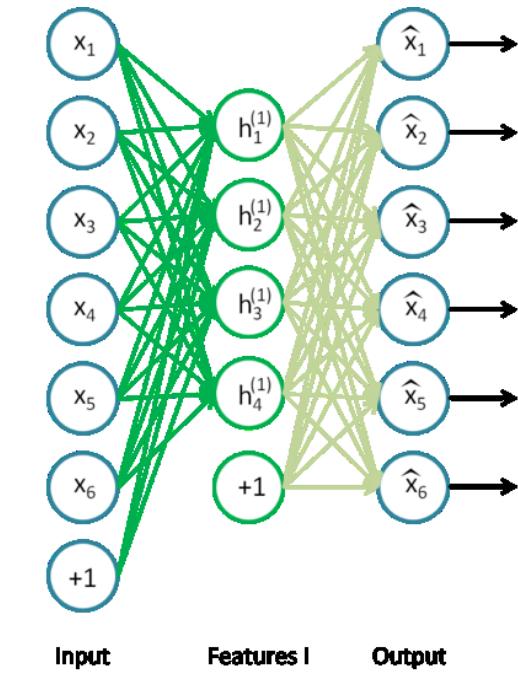
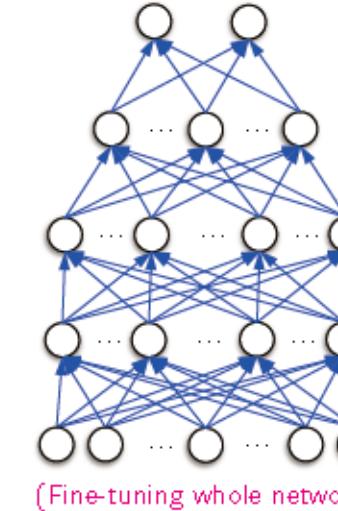
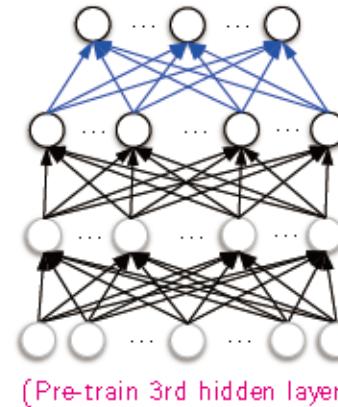
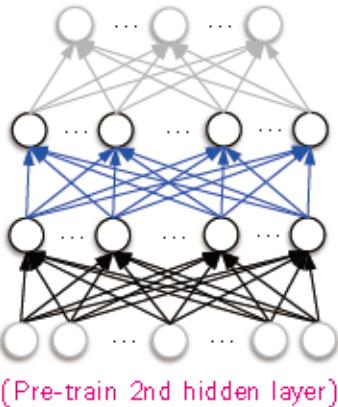
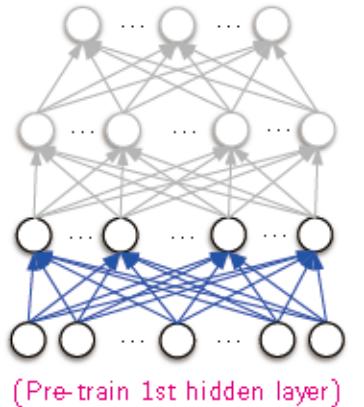
## ConvNets: till 2012

Common wisdom: training does not work because we “get stuck in local minima”



# Deep Belief Network

- Probabilistic generative model
- Deep architecture – multiple layers (stacks of RBMs)
- Greedy layer-wise training algorithm
- Supervised fine-tuning can be applied



# Difficulties of Training DNN

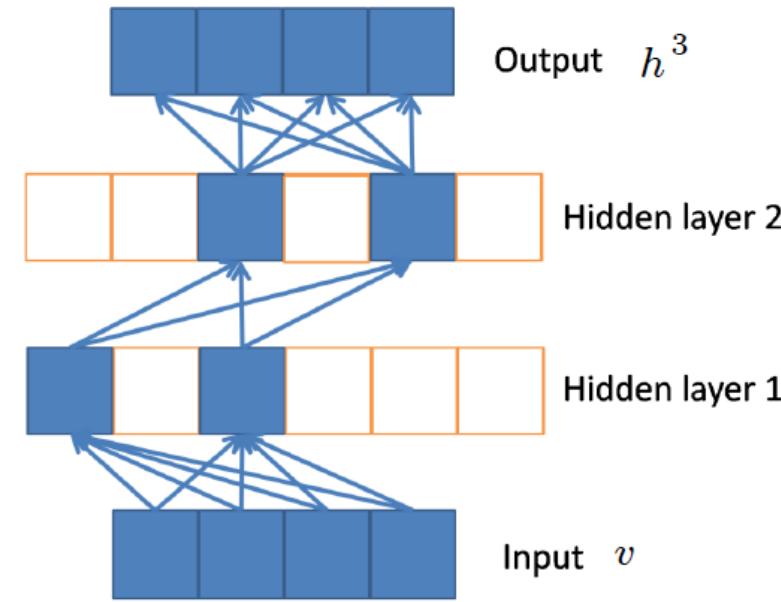
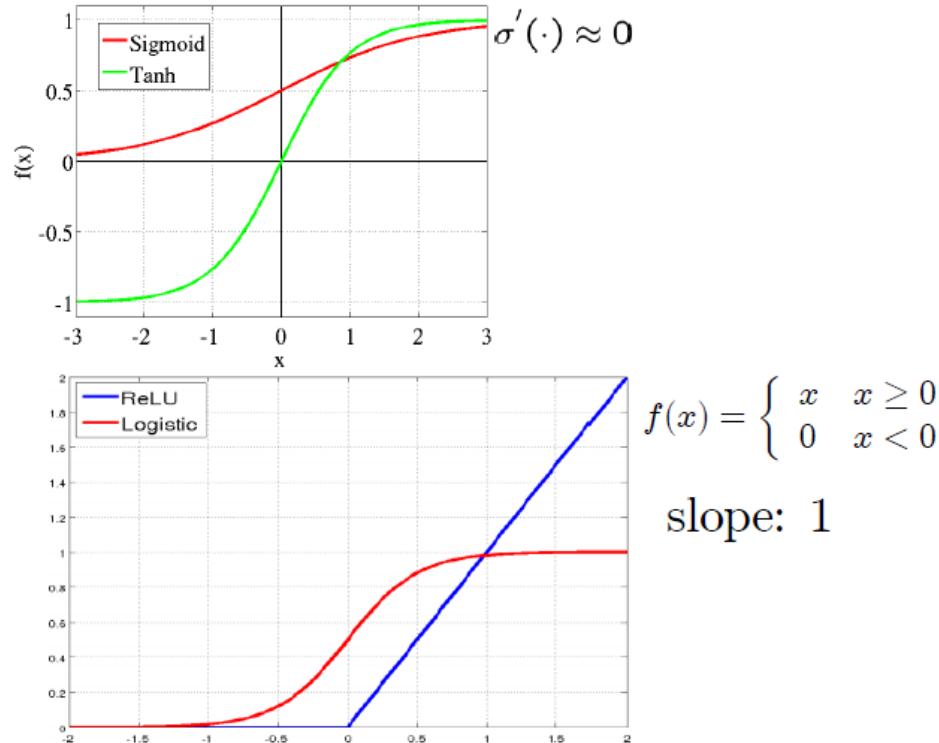
- Vanishing gradient problem
  - Solved by bottom-up layerwise unsupervised pre-training
- Typically requires lots of labeled data
- Overfitting problem
  - Solved by using lots of unlabeled data
- Get stuck in local minima(?)
  - Unsupervised pre-training may help the network initialize with good parameters(?)

# 새로운 방법들

- Vanishing gradient problem  
→ Sigmoid 말고 ReLU를 쓰자
- Overfitting problem  
→ Regularization method를 쓰자(예 : dropout)
- Get stuck in local minima  
→ Local minima에 빠져도 괜찮다

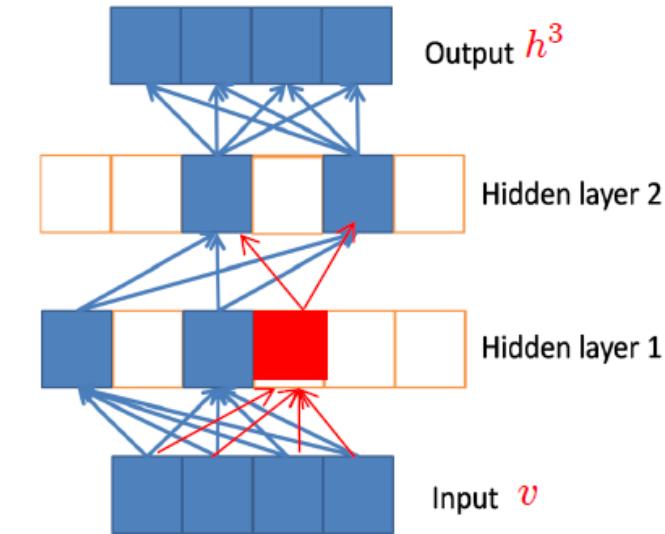
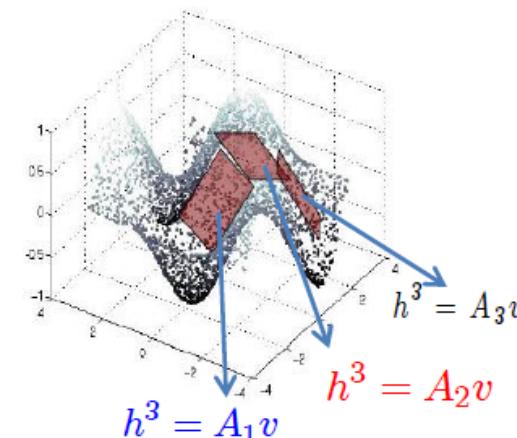
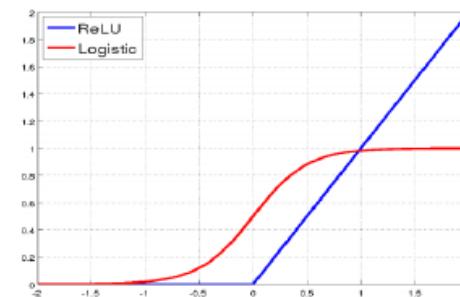
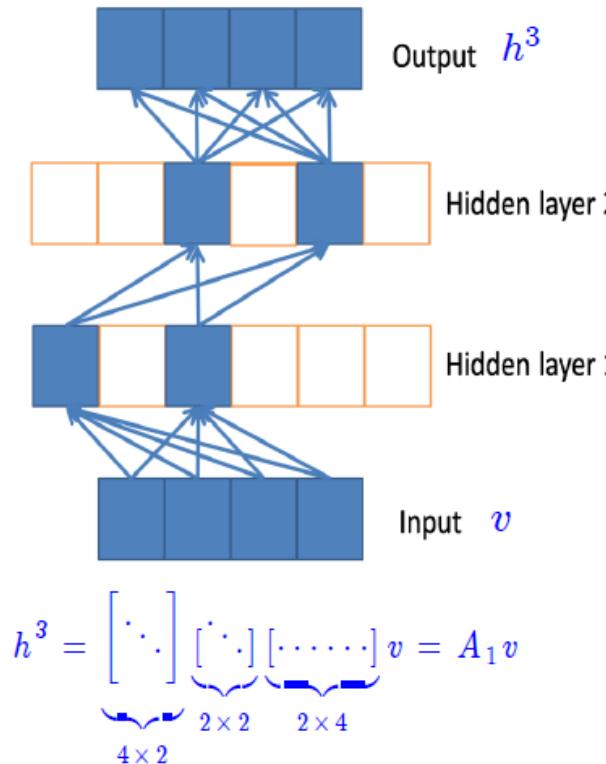
# ReLU : Rectified Linear Unit

- ReLU를 activation function으로 사용 → sparse activation
- ReLU는 미분값이 0 아니면 1 → vanishing gradient 해결



# ReLU의 의미

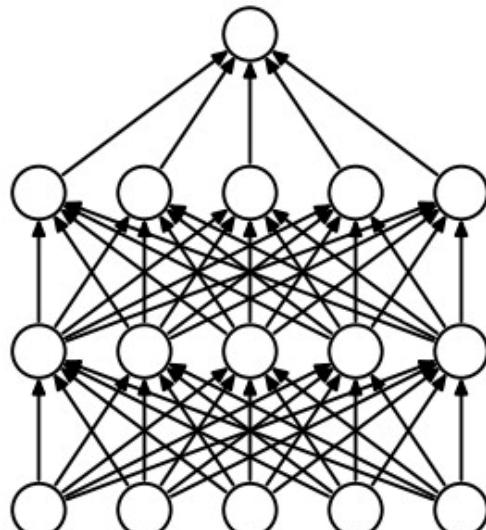
- Piece-wise linear tiling : locally linear mapping



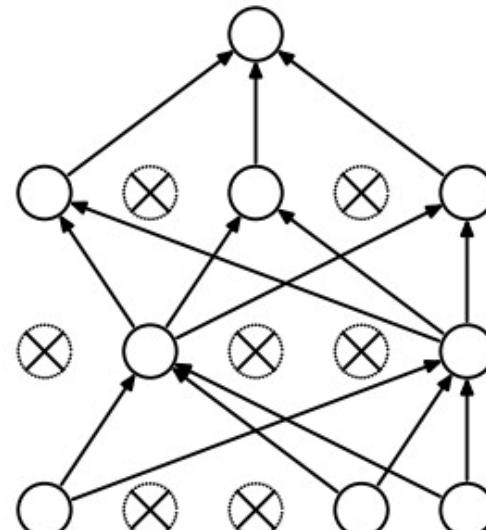
$$h^3 = \underbrace{\begin{bmatrix} \cdot & \cdot \\ \cdot & \cdot \end{bmatrix}}_{4 \times 2} \underbrace{\begin{bmatrix} \cdot & \cdot \end{bmatrix}}_{2 \times 3} \underbrace{\begin{bmatrix} \cdots \cdots \end{bmatrix}}_{3 \times 4} v = A_2 v$$

# Dropout(Regularization Method)

- 각 학습 단계마다, 특정 확률로(예 : 50%) random하게 hidden layer에 있는 unit들을 없애고 학습하는 방법
- Ensemble 개념을 적용
  - 여러 개의 model을 사용하여 평균값을 쓰면 하나의 model을 쓰는 경우보다 좋음
  - 하나의 model로 비슷한 효과를 낼 수 있는 방법



(a) Standard Neural Net



(b) After applying dropout.

# Other Regularization Methods

- Weight Decay(L2 Regularization)

$$E(w) = E_0(w) + \frac{1}{2} \lambda \sum_i w_i^2$$

- Batch Normalization

- Benefits of BN

- Increase learning rate
- Remove dropout
- Reduce L2 weight decay
- Remove LRN

**Input:** Values of  $x$  over a mini-batch:  $\mathcal{B} = \{x_1 \dots m\}$ ;  
Parameters to be learned:  $\gamma, \beta$

**Output:**  $\{y_i = \text{BN}_{\gamma, \beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m} \sum_{i=1}^m x_i \quad // \text{mini-batch mean}$$

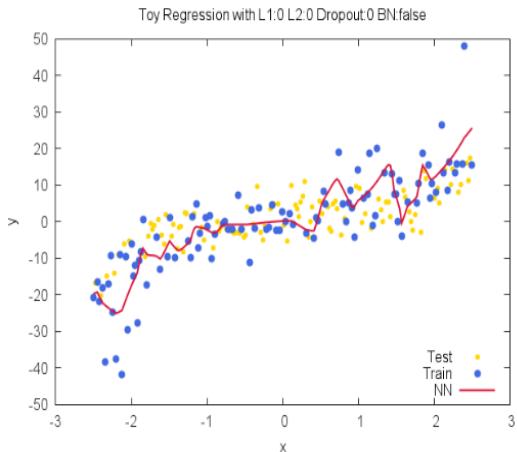
$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m} \sum_{i=1}^m (x_i - \mu_{\mathcal{B}})^2 \quad // \text{mini-batch variance}$$

$$\hat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \quad // \text{normalize}$$

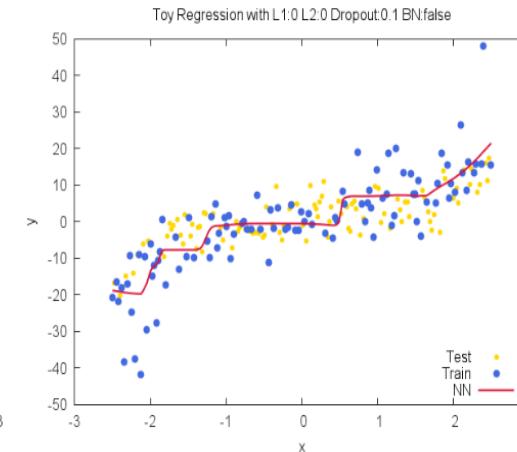
$$y_i \leftarrow \gamma \hat{x}_i + \beta \equiv \text{BN}_{\gamma, \beta}(x_i) \quad // \text{scale and shift}$$

# Regularization Methods

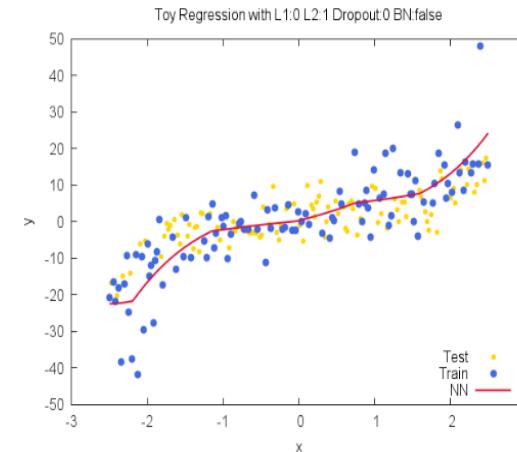
No Regularization



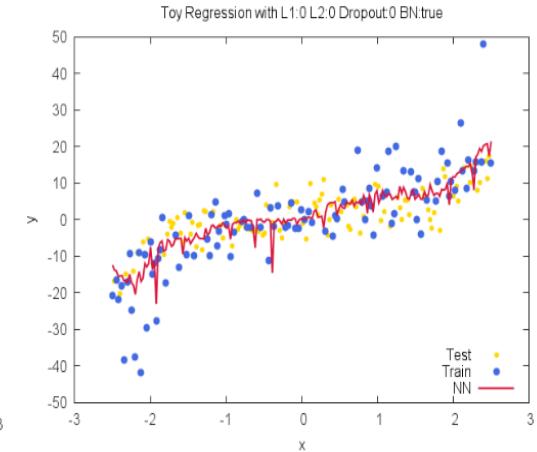
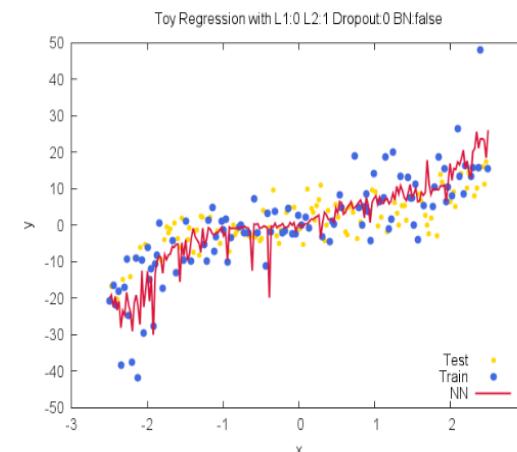
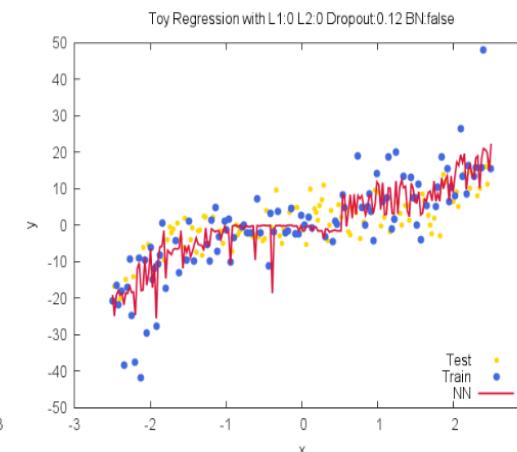
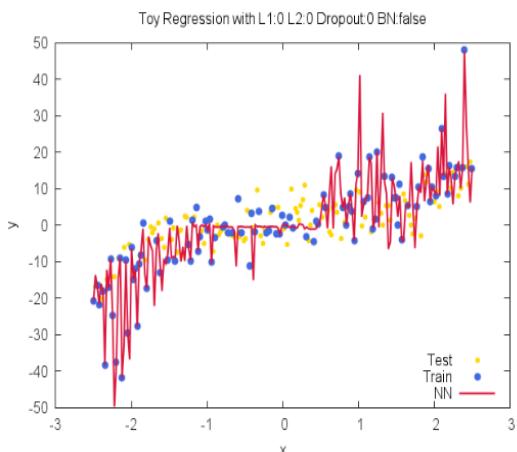
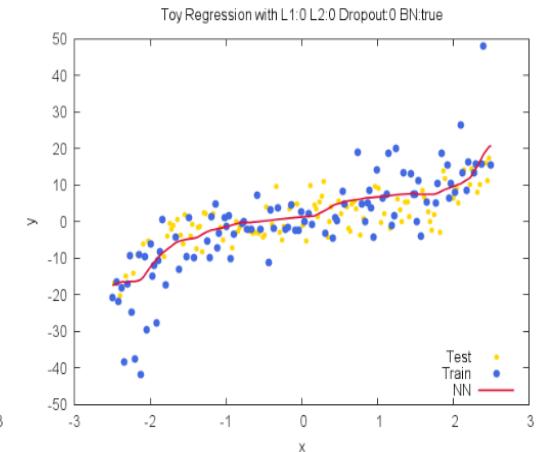
Dropout



L2 Regularization



Batch Normalization

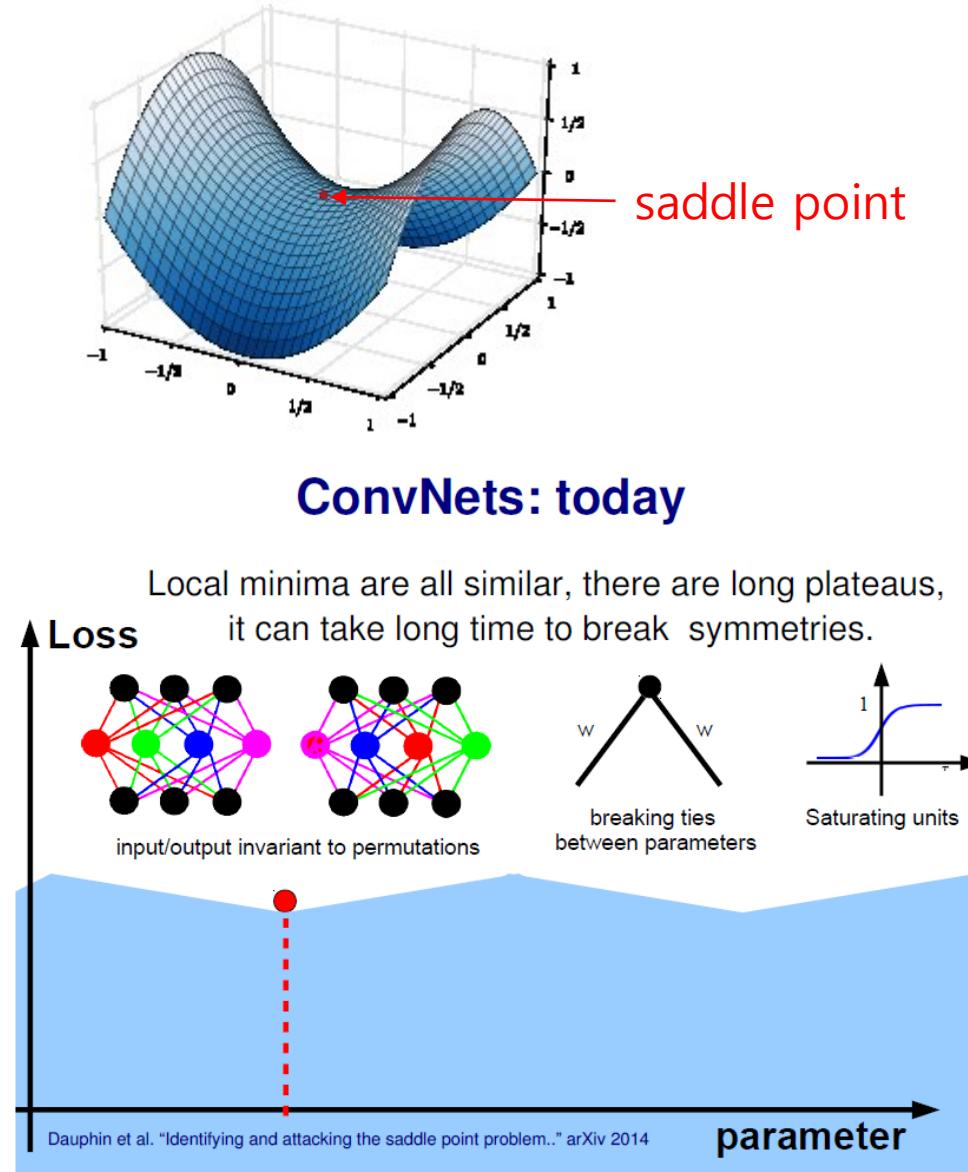


# Local Minima

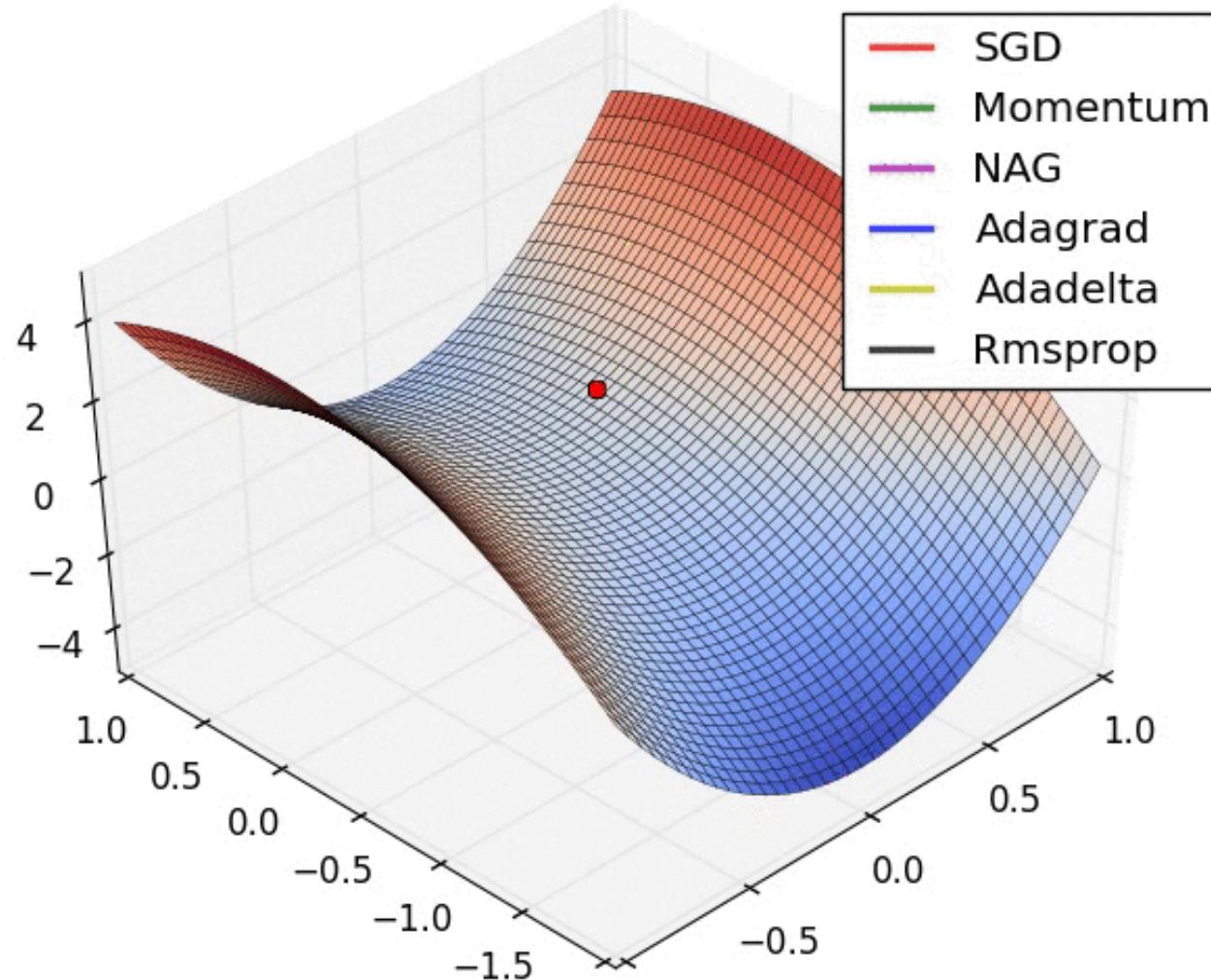


# Local Minima에 대하여

- minimum이라고 하는 것은 현재 차원에서 이동할 수 있는 모든 방향으로의 gradient 값이 증가하는 방향이어야 하는데 이런 경우는 확률적으로 희박함
- DNN과 같은 고차원 구조에서는 대부분은 local minima가 아니라 saddle point일 가능성이 높음
- 만약 실제 local minima가 존재한다면 그것은 global minimum과 거의 차이가 없을 가능성이 높음(neural network의 대칭성)



# Optimization Methods

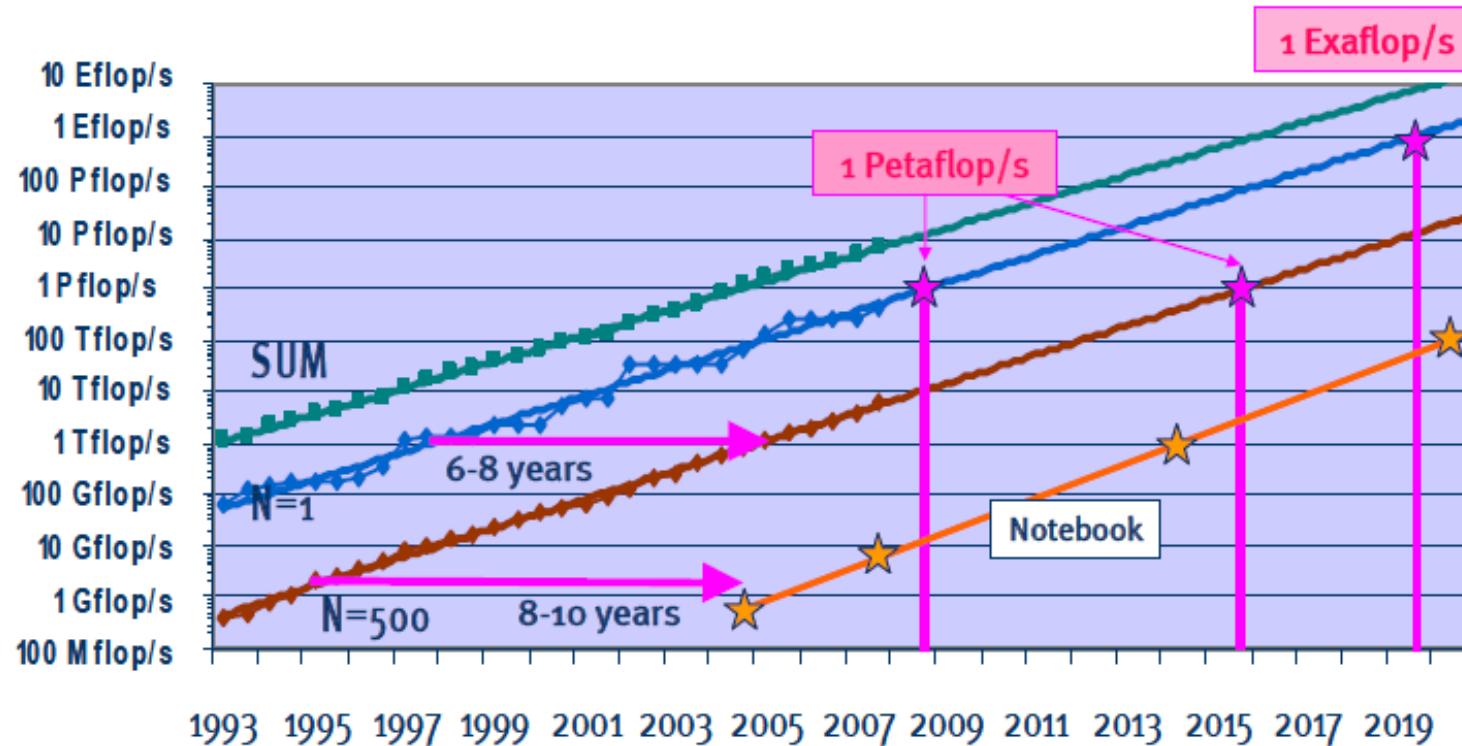


# 그 밖에 Deep Learning을 가능하게 했던 것들

- Hardware
  - CPUs, GPUs!, ASICs <https://youtu.be/-P28LKWTzrl>
- Organized Large Datasets
  - ImageNet
- Algorithms and Research
  - Backprop, CNN, LSTM
- Software and Infrastructure
  - Git, AWS, Amazon Mechanical Turk, TensorFlow, ...
- Financial Backing of large Companies
  - Google, Facebook, Amazon, ...

# DeepBlue와 AlphaGo

- 1997년 6월 기준, DeepBlue는 세상에서 259번째로 빠른 슈퍼컴퓨터였음
  - Performance = 11.4 GFLOPs(Galaxy S8: 375 GFLOPs)
- 이세돌과 대결할 당시 AlphaGo는 세계에서 약 500번째로 빠른 컴퓨터였음



# 어떻게 시작할까?

- Data 준비
- Data를 2 or 3가지로 나눔
  - Training set, Test set
  - Training set, Validation set, Test set
- 어떤 모델(알고리즘)을 사용할 것인지 결정
  - 원하는 output의 형태에 따라
  - Data의 종류와 특성에 따라
- Training → Tuning(validation) → Test

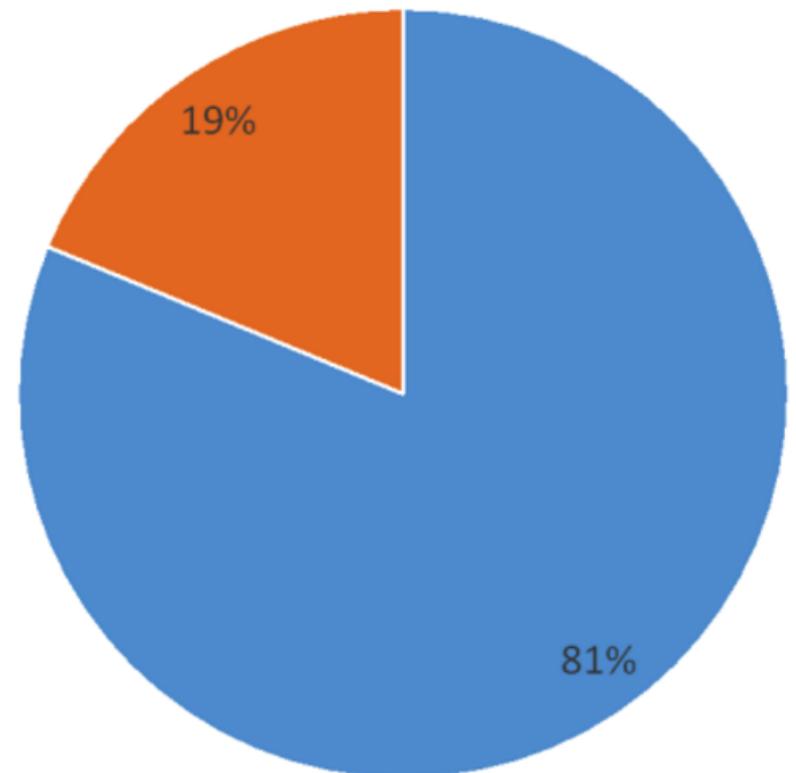


# Validation Dataset

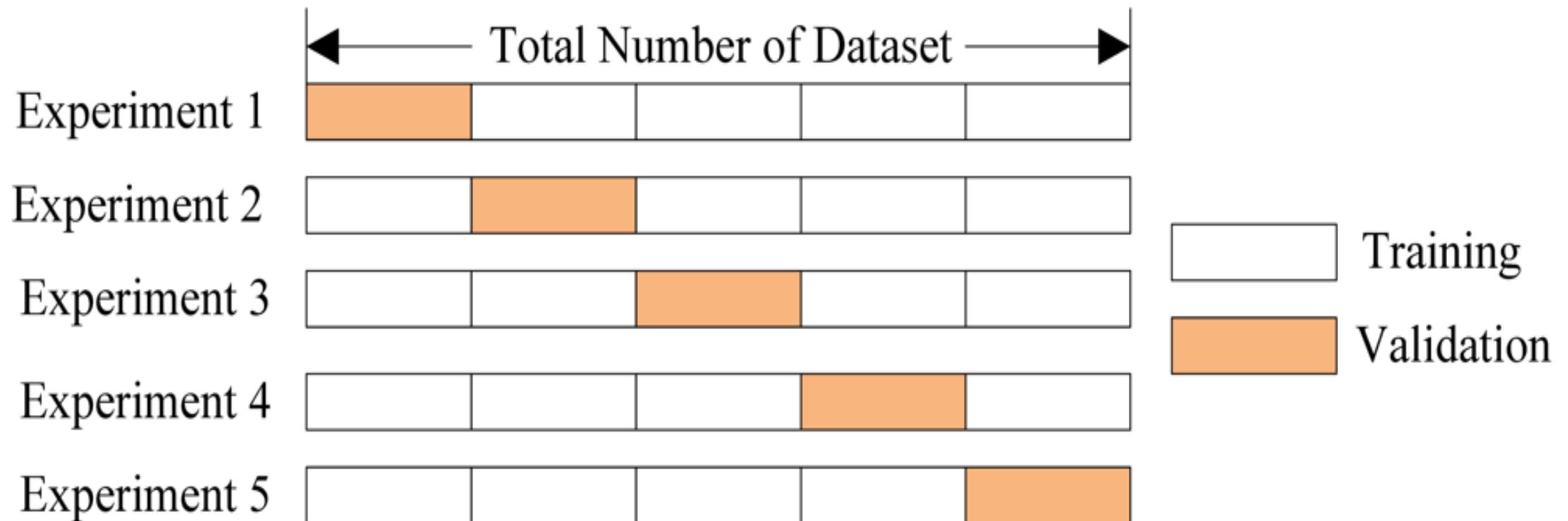
- Problems

- 모든 data를 학습에 사용하지 못함
- 해당 split에만 운좋게 잘할 수 있음(overfitting)

Percentage Split



# Cross-Validation

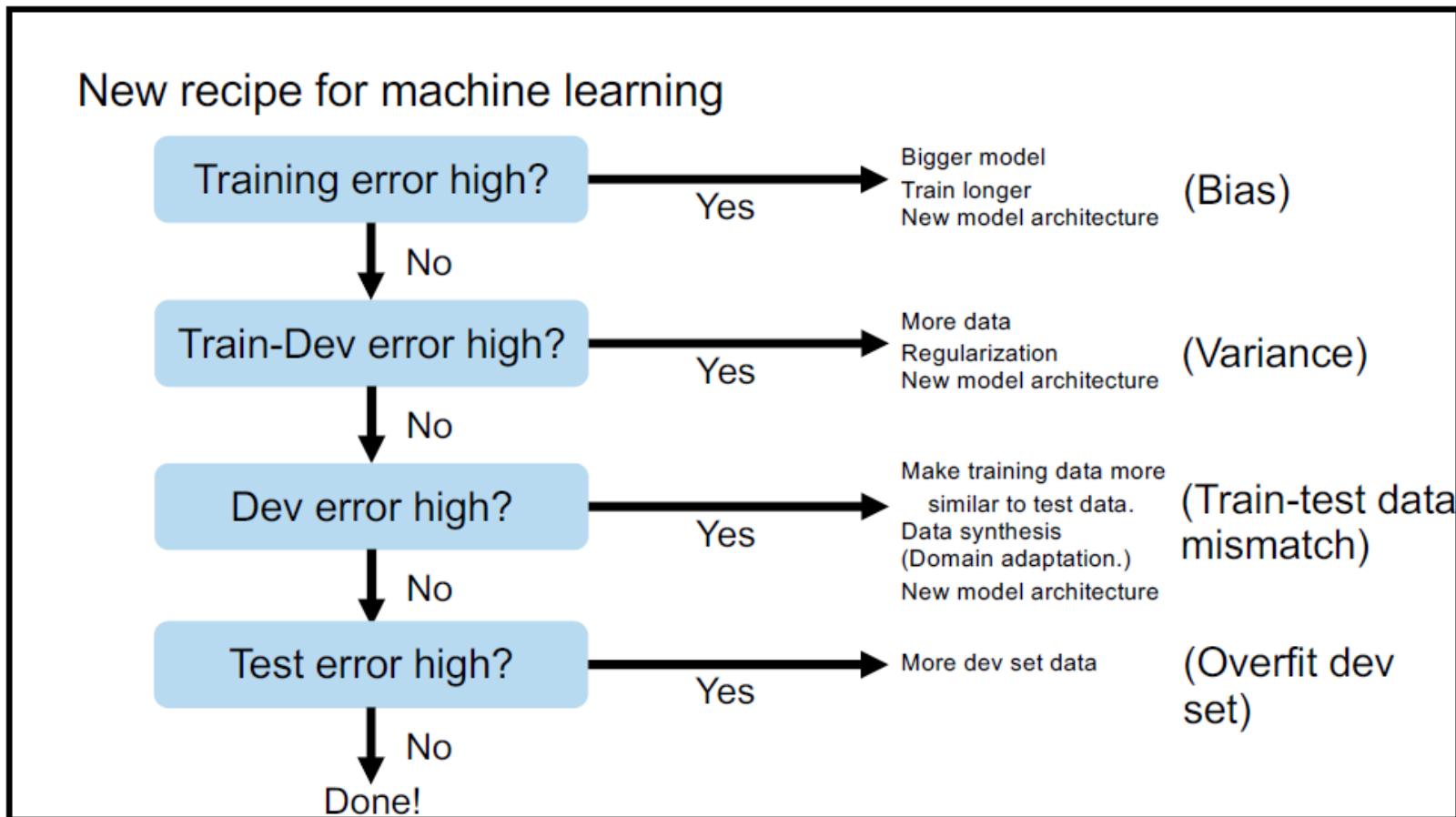


# Realistic Problems

- Data
- Measurement
- Result Interpretation

# Data

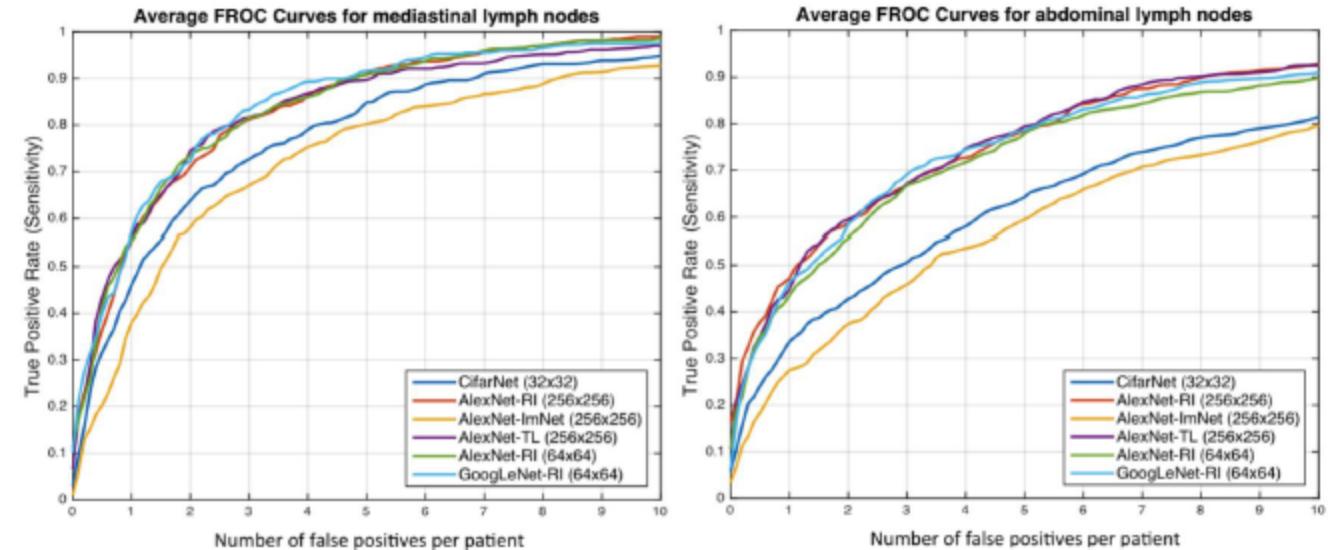
- Data, more data (by Andrew Ng's tutorial @NIPS 2016)



# If We Don't Have Enough Data

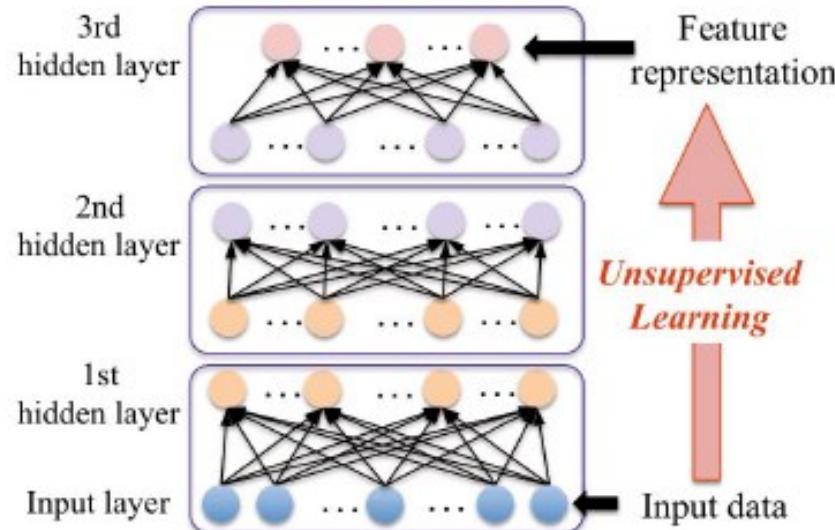
- Use pre-trained model!
- Transfer learning from other domains
  - Transferred network with 'deep' fine-tuning shows good results

Region	Mediastinum		Abdomen	
Method	AUC	TPR/3FP	AUC	TPR/3FP
[41]	-	0.63	-	0.70
[22]	<b>0.92</b>	0.70	<b>0.94</b>	<b>0.83</b>
[36]	-	<b>0.78</b>	-	0.78
CifarNet	0.91	0.70	0.81	0.44
AlexNet-ImNet	0.89	0.63	0.80	0.41
AlexNet-RI-H	0.94	0.79	0.92	0.67
AlexNet-TL-H	0.94	0.81	0.92	0.69
GoogLeNet-RI-H	0.85	0.61	0.80	0.48
GoogLeNet-TL-H	<b>0.94</b>	<b>0.81</b>	<b>0.92</b>	<b>0.70</b>

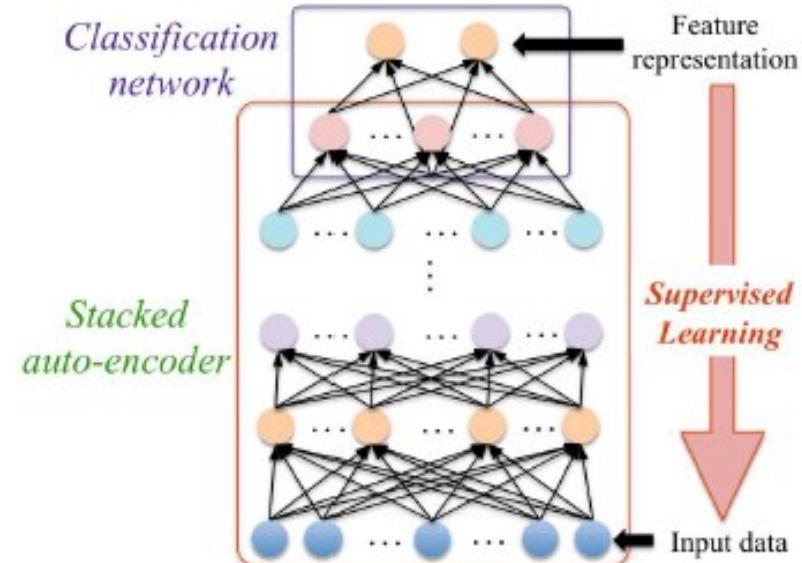


# If We Don't Have Enough Data

- Unsupervised pre-training and supervised Fine-tuning
  - Unsupervised training of unlabeled data(using auto-encoder) before supervised learning can make good results



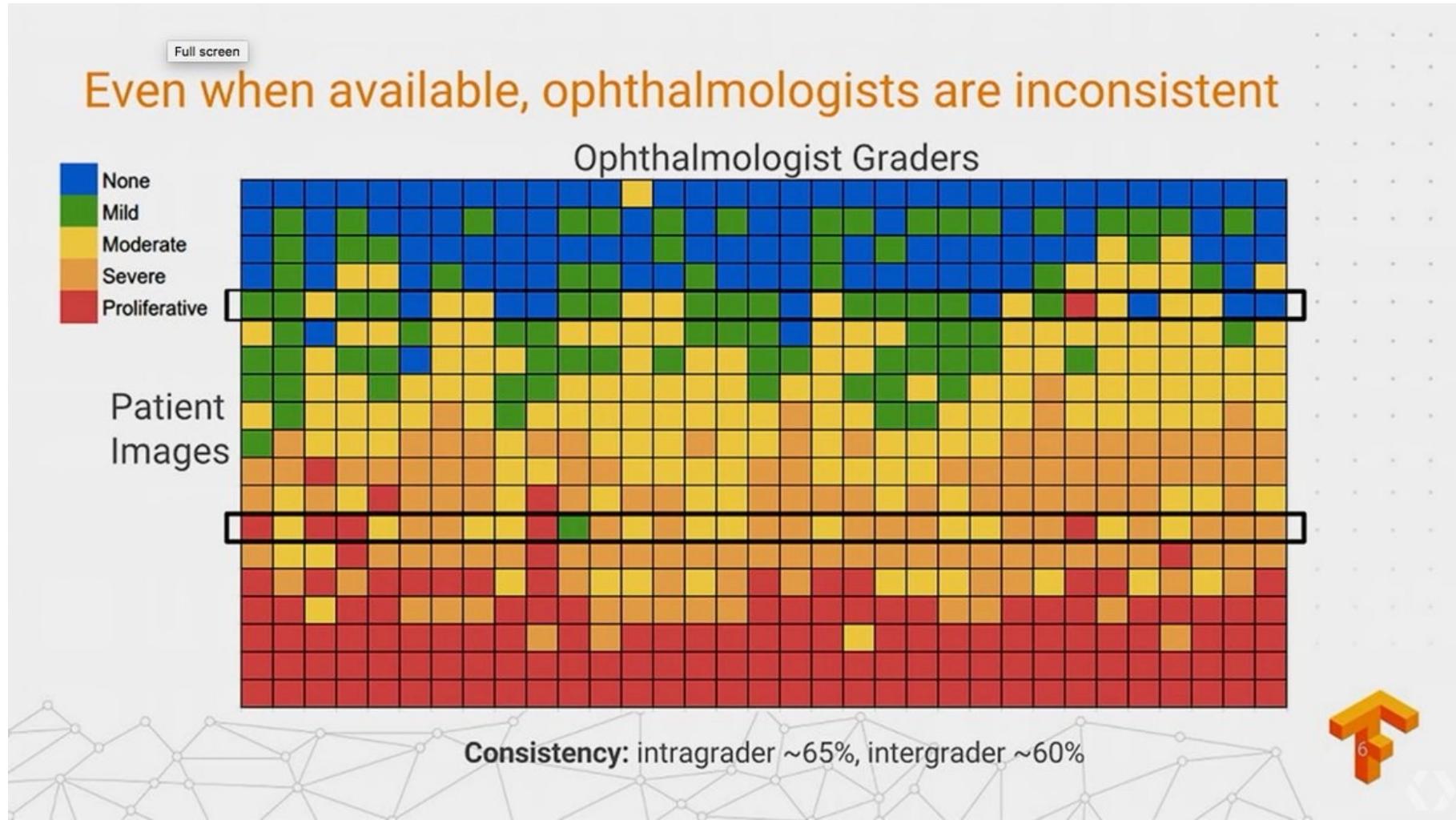
(a) Unsupervised pre-training



(b) Supervised fine-tuning

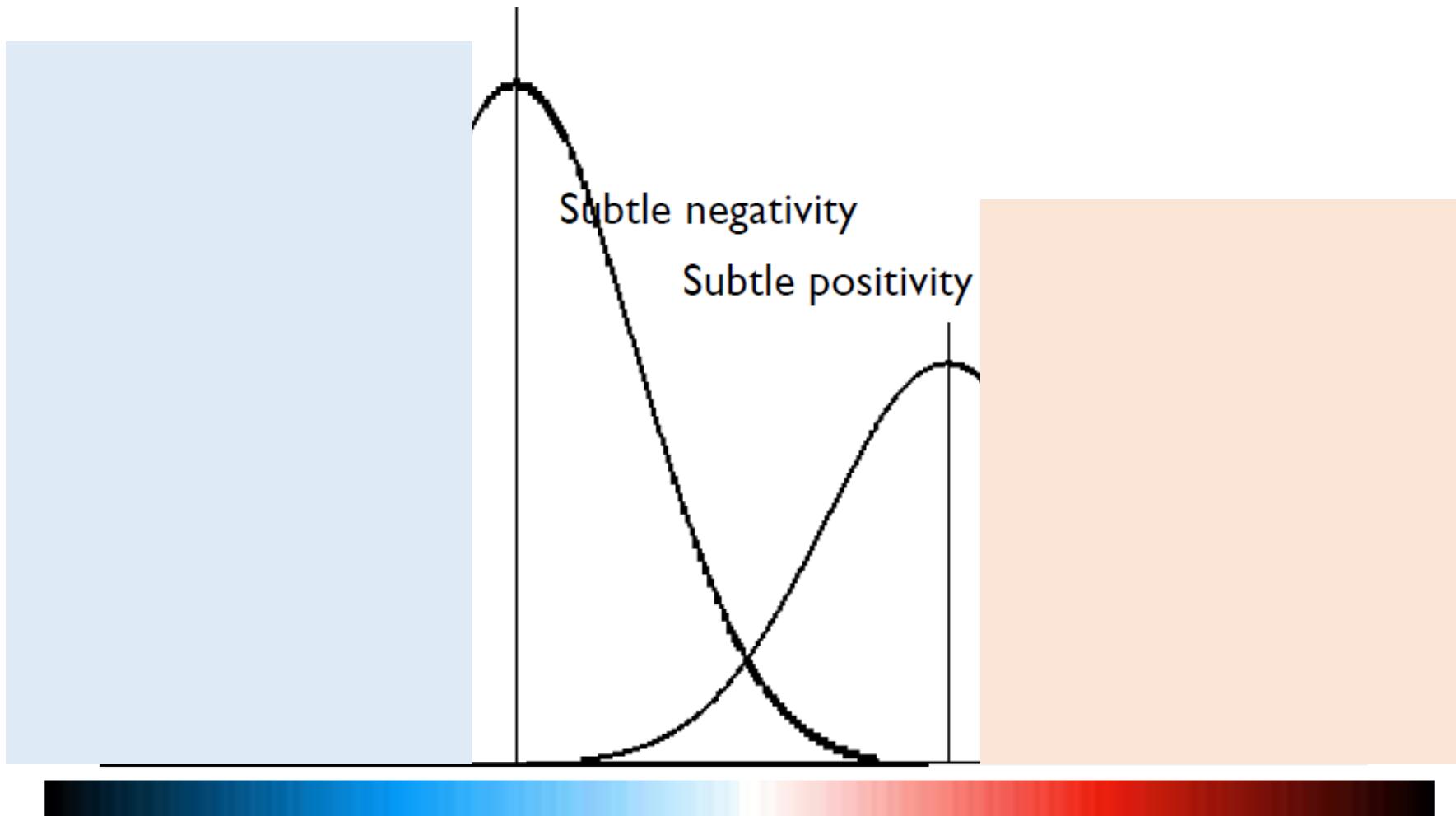
# Good Data vs Bad Data

- Our labels are perfect?



# Good Data vs Bad Data

- Our data is unbiased?



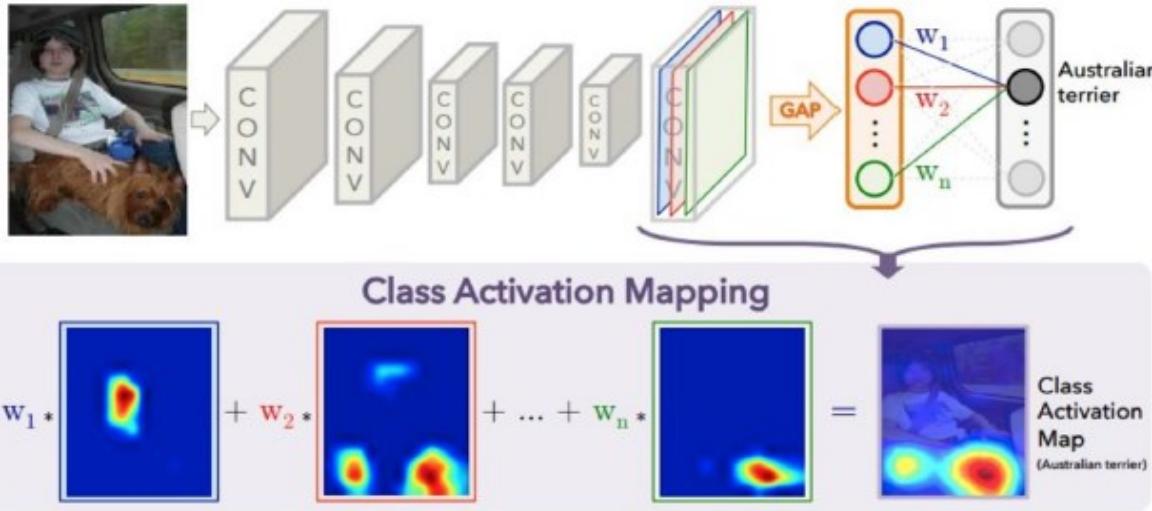
# Is Our Model Good Enough?

- Sometimes accuracy is not a good measurement

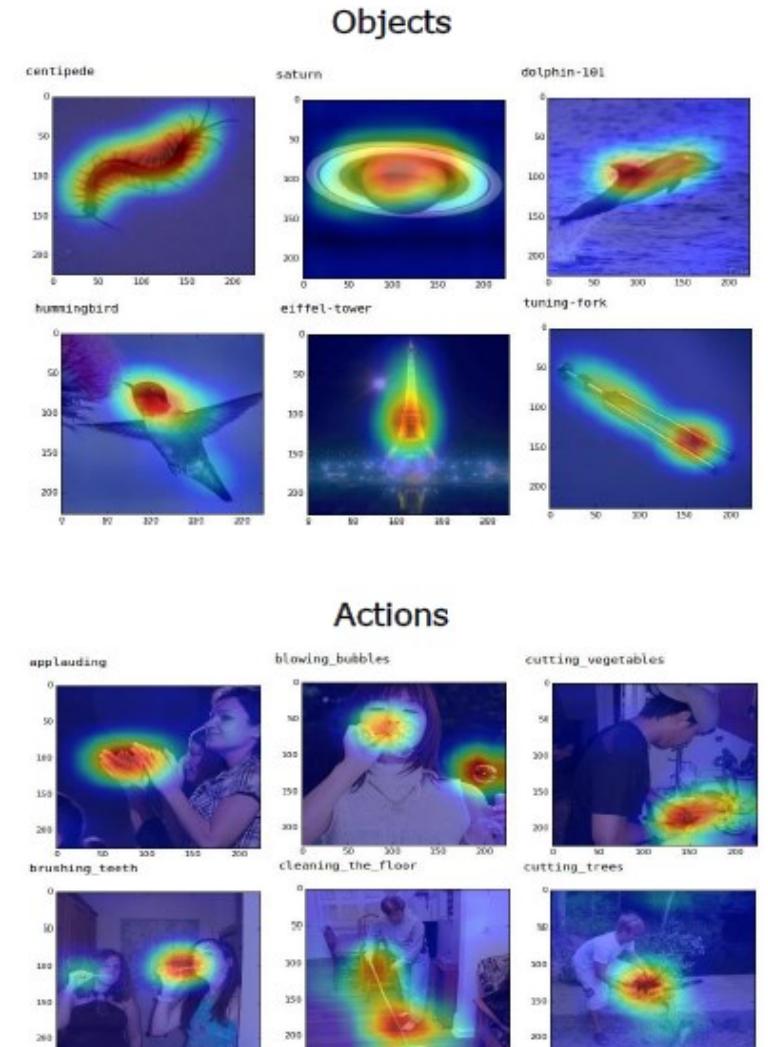
		Predicted class	
		$P$	$N$
Actual Class	$P$	True Positives (TP)	False Negatives (FN)
	$N$	False Positives (FP)	True Negatives (TN)

**Sensitivity**      **Recall**      **Precision**

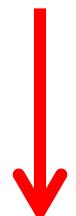
# Can we visually interpret the result



B. Zhou et. al., CVPR, 2016



# Trends in Deep Learning

- Difficult for humans but easy for computers
- 
- Easy for humans but difficult for computers(Now)
- 
- Difficult for humans and difficult for computers

# Challenges of Deep Learning

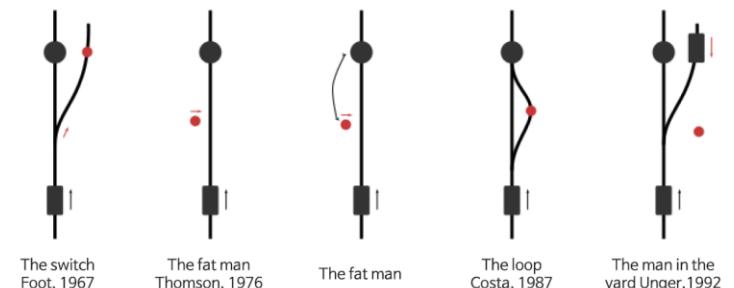
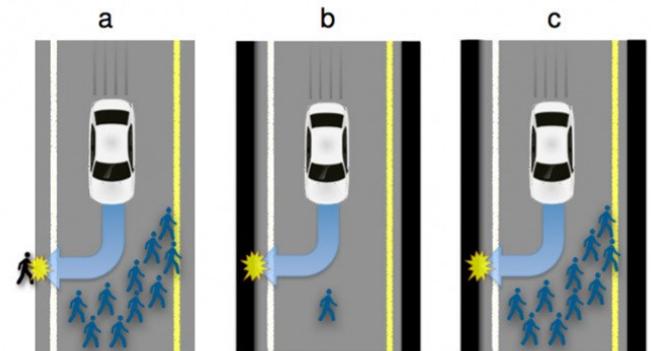
- Extract information from unlabeled data
- Develop a theoretical understanding of neural networks
- Tap into the potential of transfer learning(AGI)
- Scale for distributed big data & optimization for edges

# Deep Learning은 만능?

- 동유럽국가 몬테네그로의 수도는 어디인가?
- 바둑판을 1줄씩 늘려서  $20 \times 20$ 으로 만들고 지금 당장 이세돌과 알파고가 대결한다면?
- 의료 data를 분석하여, 수명을 예측해봅시다
  - 평생 한번도 담배를 안피운 사람
  - 현재 흡연을 하고 있는 사람
  - 과거에 흡연을 했다가 끊은 사람
- 인공지능은 인류의 지능 향상에 도움이 되는가?
- 인공지능의 능력이 커질수록 법적, 도덕적 문제에 대한 해결이 필요



L'Avion III de Clément Ader, 1897  
(Musée du CNAM, Paris)



The switch  
Foot, 1967

The fat man  
Thomson, 1976

The fat man  
Costa, 1987

The loop  
Unger, 1992

The man in the  
yard