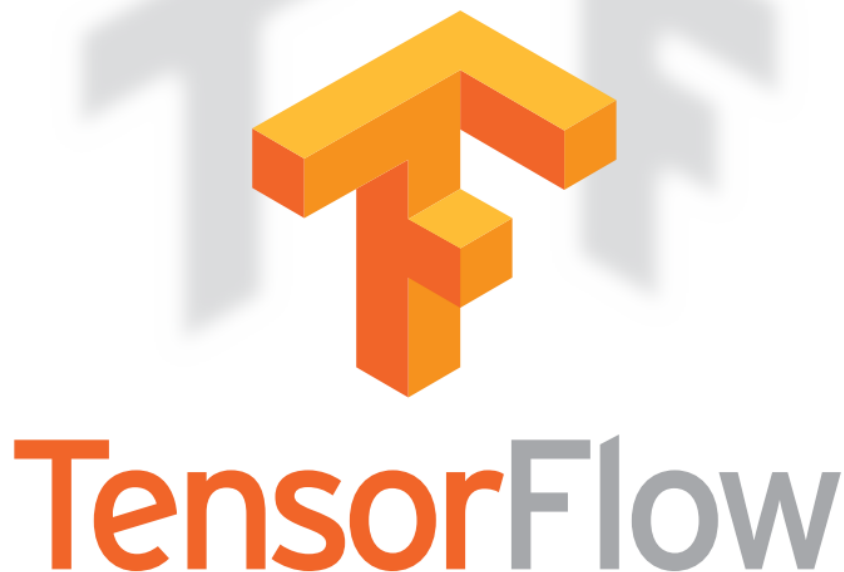


# 딥러닝 구현을 위한 텐서플로우 개발

김성균



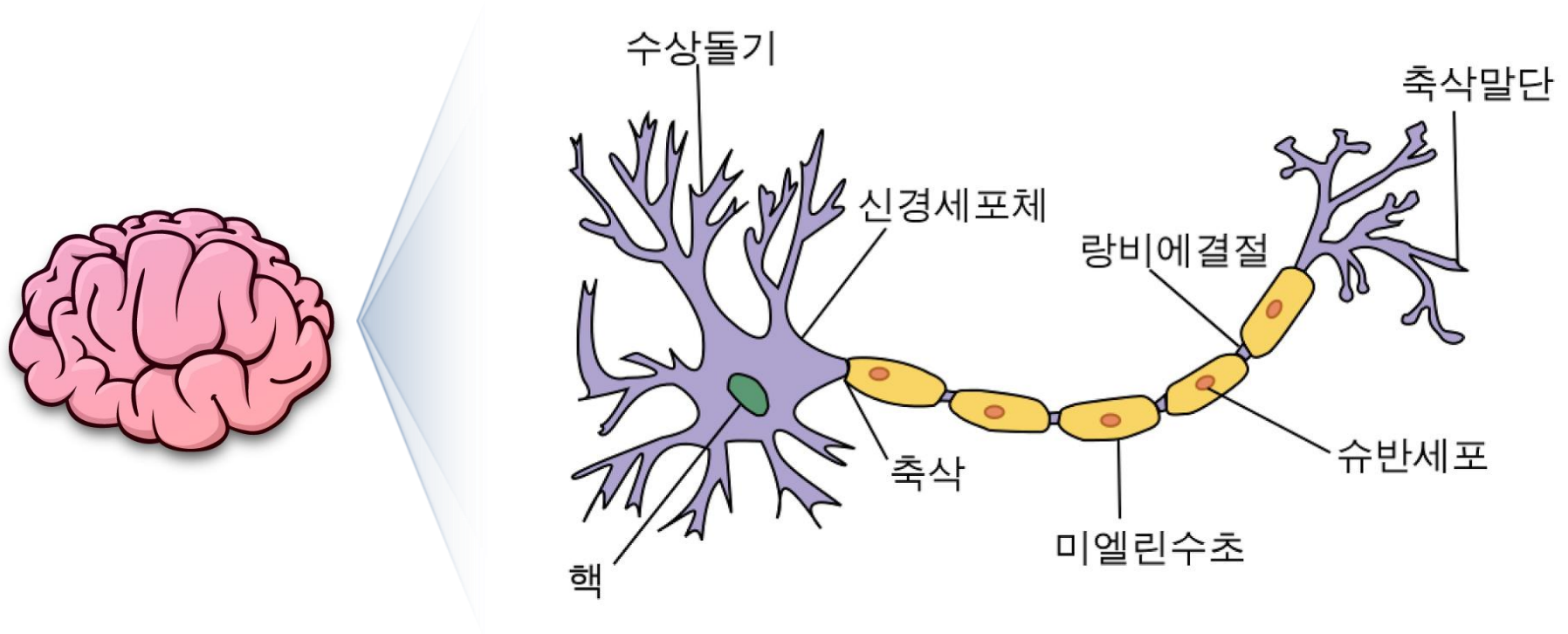
딥러닝 구현을 위한 **텐서플로우** 개발

# 6강

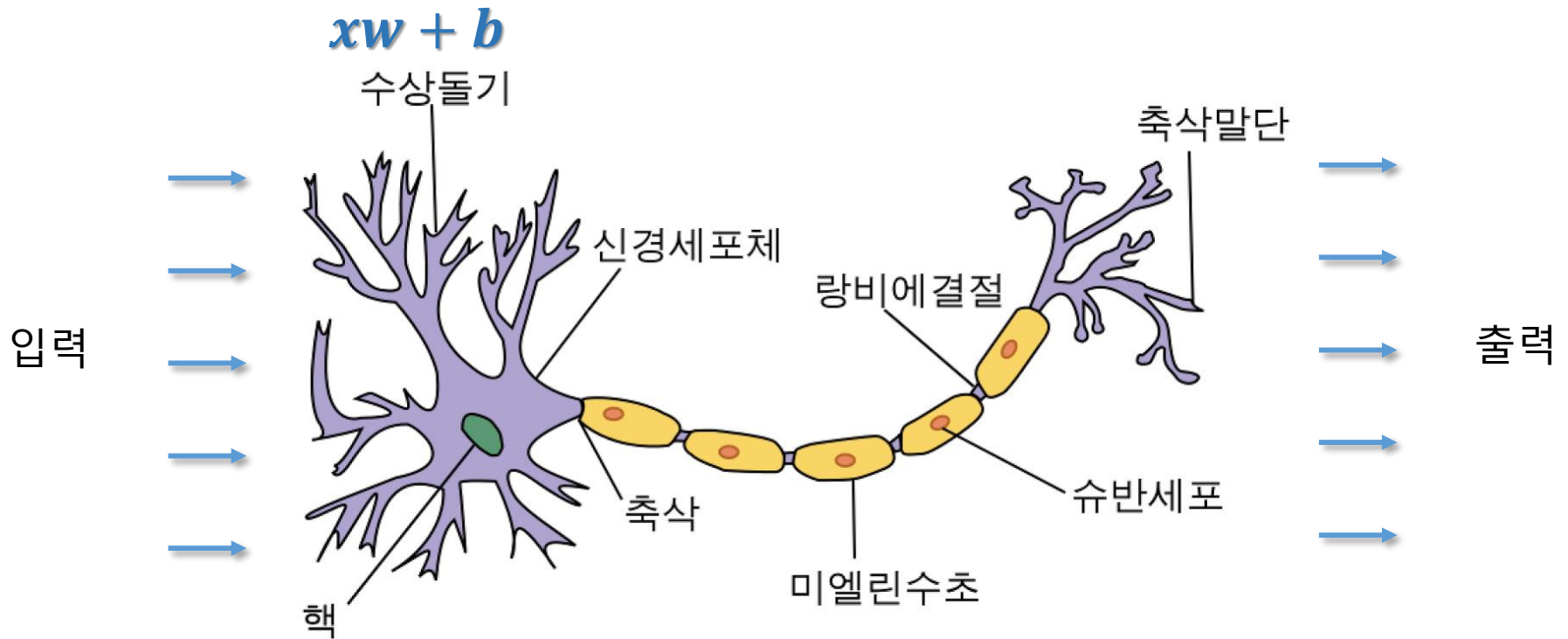
## 딥러닝의 기본개념

# 01.인공신경망(artificial neural network, ANN)

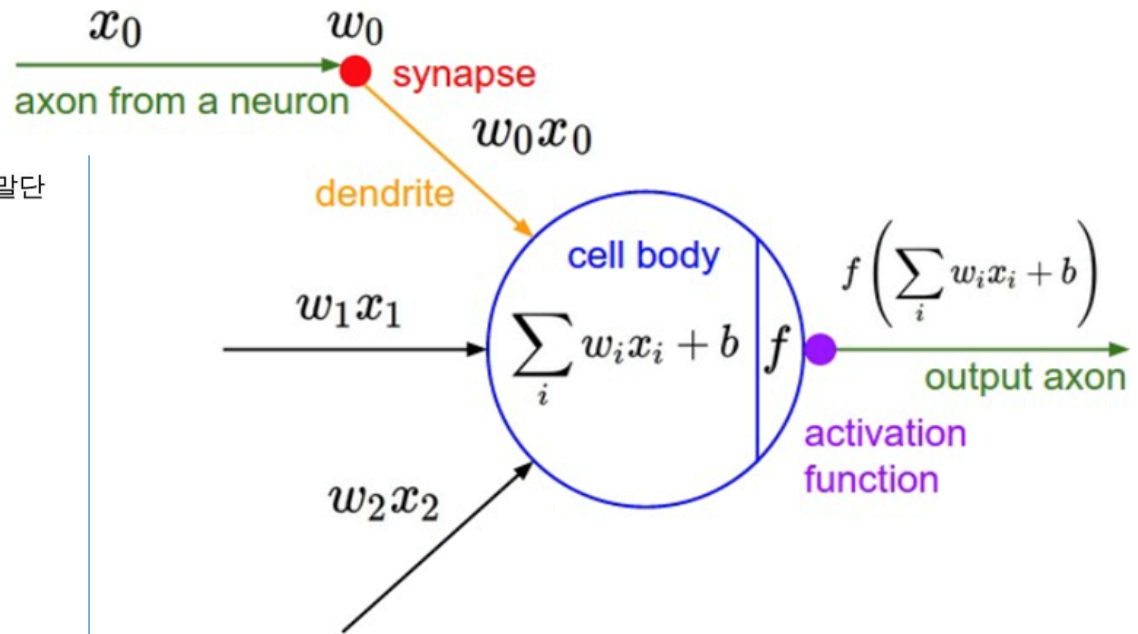
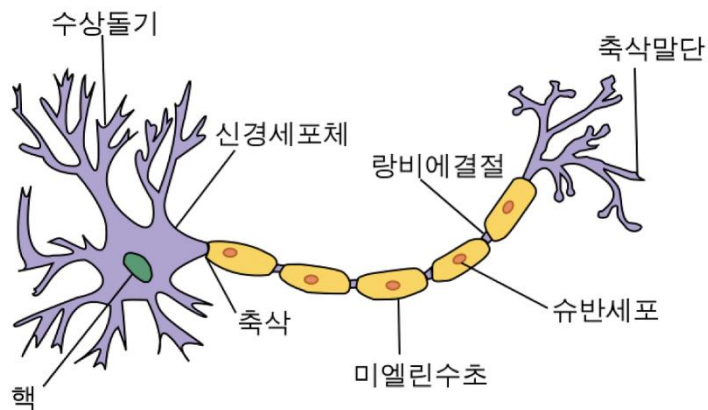
- 생물학의 신경망(동물의 중추신경계 중 특히 뇌)에서 영감을 얻은 통계학적 학습 알고리즘



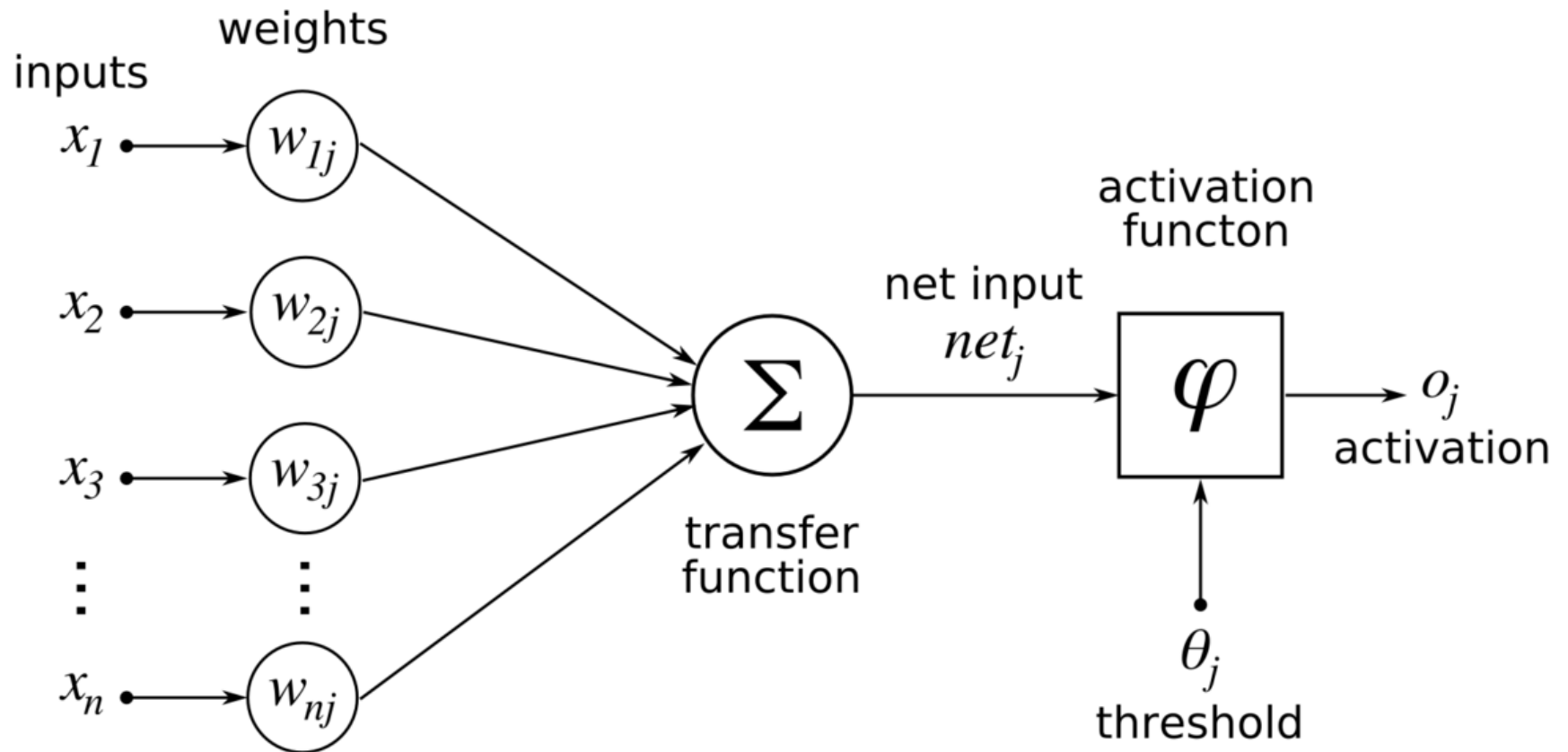
# 01.인공신경망(artificial neural network, ANN)



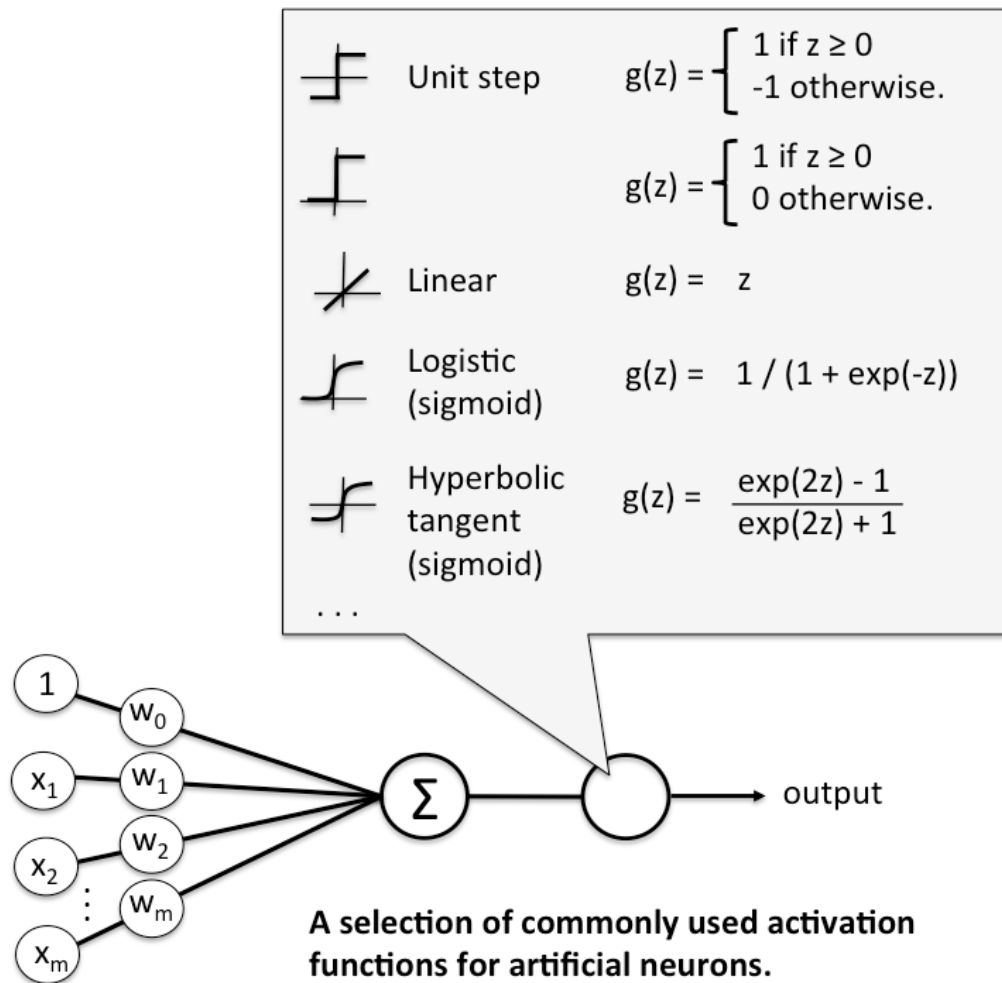
## 02. Activation Functions in Artificial Neural Networks



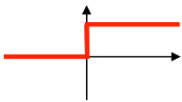
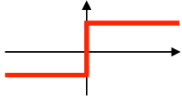
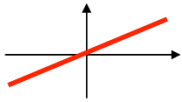
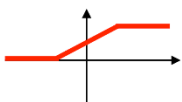
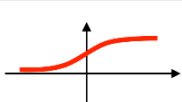
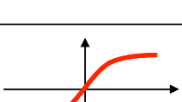
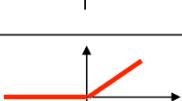
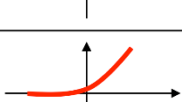
## 02. Activation Functions in Artificial Neural Networks



## 02. Activation Functions in Artificial Neural Networks



## 02. Activation Functions in Artificial Neural Networks

Activation function	Equation	Example	1D Graph
Unit step (Heaviside)	$\phi(z) = \begin{cases} 0, & z < 0, \\ 0.5, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Sign (Signum)	$\phi(z) = \begin{cases} -1, & z < 0, \\ 0, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Linear	$\phi(z) = z$	Adaline, linear regression	
Piece-wise linear	$\phi(z) = \begin{cases} 1, & z \geq \frac{1}{2}, \\ z + \frac{1}{2}, & -\frac{1}{2} < z < \frac{1}{2}, \\ 0, & z \leq -\frac{1}{2}, \end{cases}$	Support vector machine	
Logistic (sigmoid)	$\phi(z) = \frac{1}{1 + e^{-z}}$	Logistic regression, Multi-layer NN	
Hyperbolic tangent	$\phi(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$	Multi-layer Neural Networks	
Rectifier, ReLU (Rectified Linear Unit)	$\phi(z) = \max(0, z)$	Multi-layer Neural Networks	
Rectifier, softplus	$\phi(z) = \ln(1 + e^z)$	Multi-layer Neural Networks	

Copyright © Sebastian Raschka 2016  
(<http://sebastianraschka.com>)



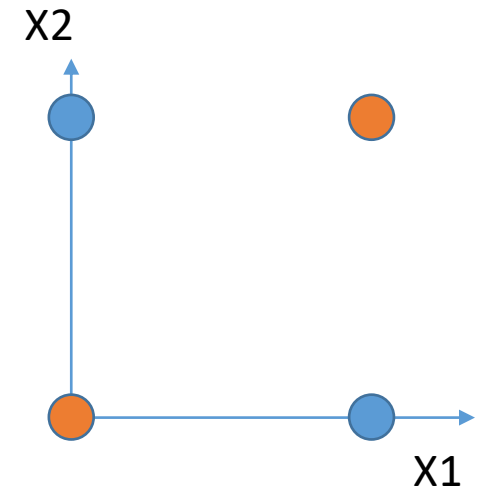
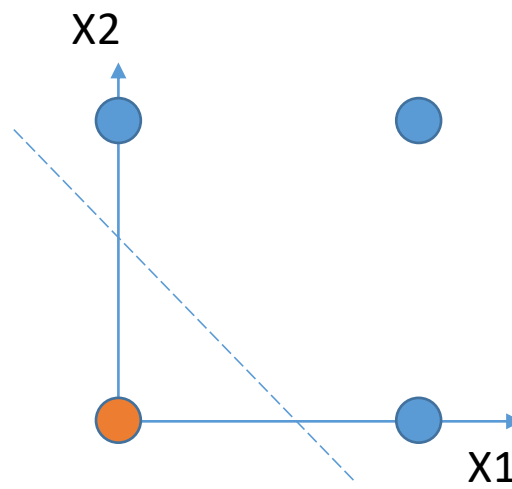
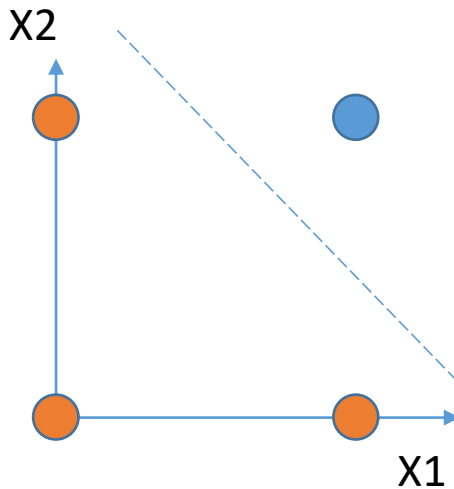
### 03. XOR 문제 대두

- OR과 AND에 대해서는 잘 동작하는데, XOR 문제는 linear 방식으로 풀 수가 없었음.

X1	X2	AND
0	0	0
0	1	0
1	0	0
1	1	1

X1	X2	OR
0	0	0
0	1	1
1	0	1
1	1	1

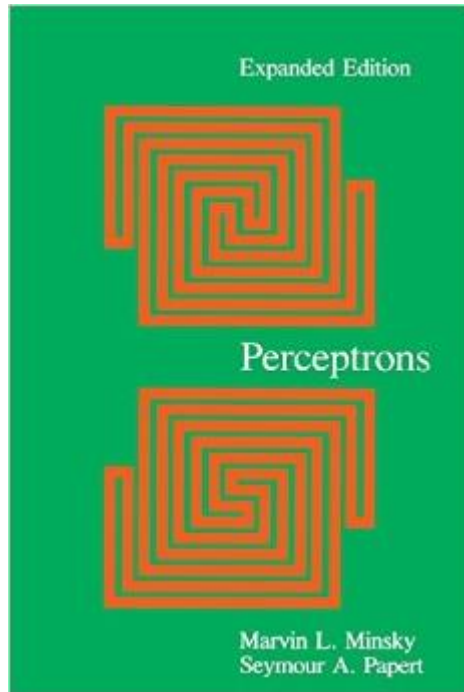
X1	X2	XOR
0	0	0
0	1	1
1	0	1
1	1	0



0: ● 1: ●

## 04. Perceptrons(1969)

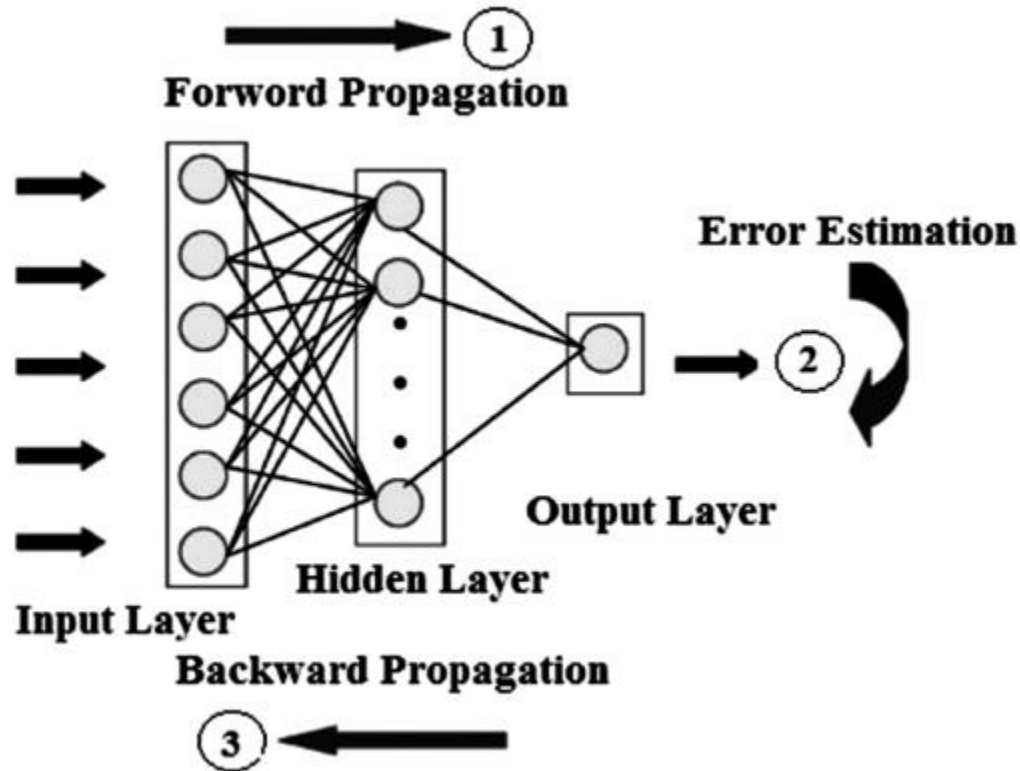
- by Marvin Minsky, founder of the MIT AI Lab



- We need to use MLP, multilayer perceptrons (multilayer neural nets)
- No one on earth had found a viable way to train MLPs good enough to learn such simple functions.
- Marvin Minsky, 1969
  - No one on earth had found a viable way to train –
- layer가 여러 개 있을 때, 각각의 layer에서 사용한  $w$ 와  $b$ 를 조절할 수 없다고 수식으로 증명.  
=> 첫번째 빙하기 도래

## 05. Backpropagation

- 1974, 1982 by Paul Werbos, 1986 by Hinton



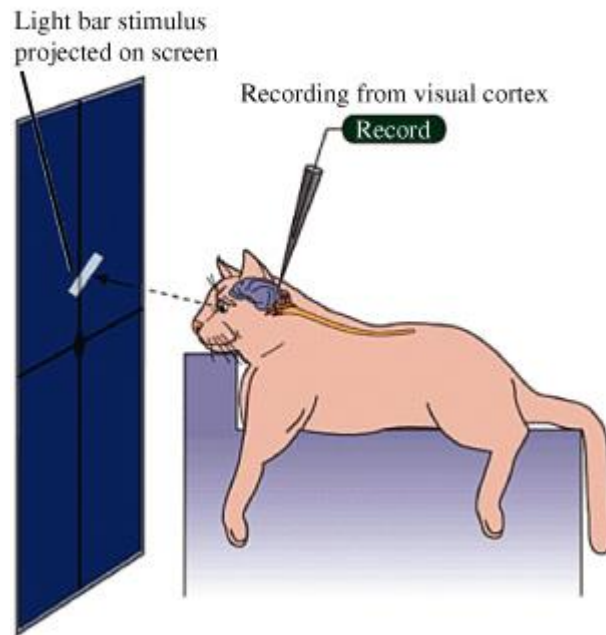
[https://www.researchgate.net/figure/Figure-2-Back-propagation-multilayer-ANN-with-one-hidden-layer\\_241741756\\_fig2](https://www.researchgate.net/figure/Figure-2-Back-propagation-multilayer-ANN-with-one-hidden-layer_241741756_fig2)

## 06. Convolutional Neural Networks

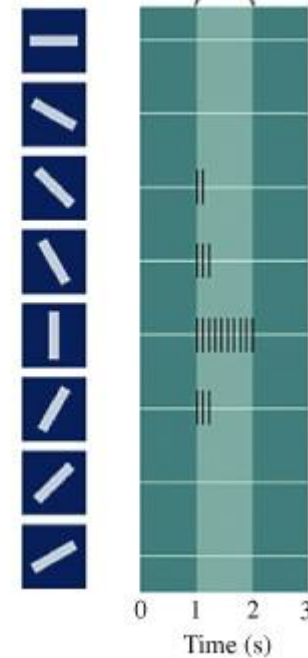
- David Hubel and Torsten Wiesel, 1959



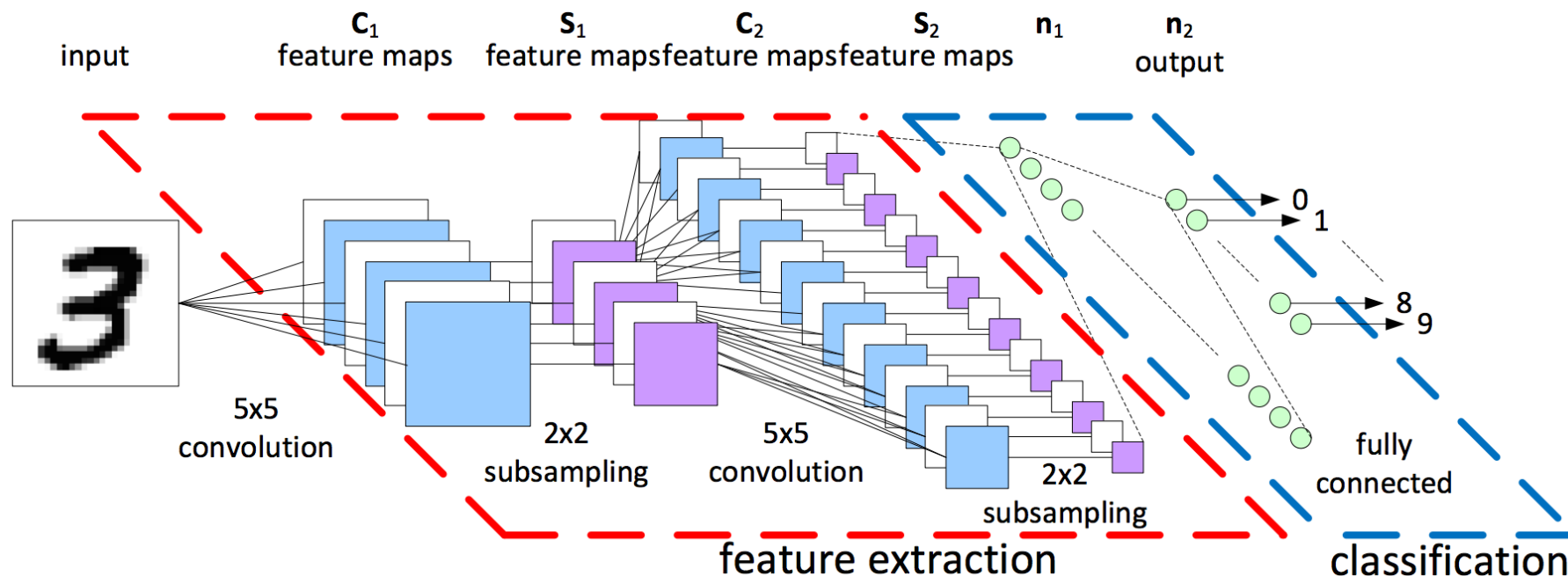
A Experimental setup



B Stimulus orientation

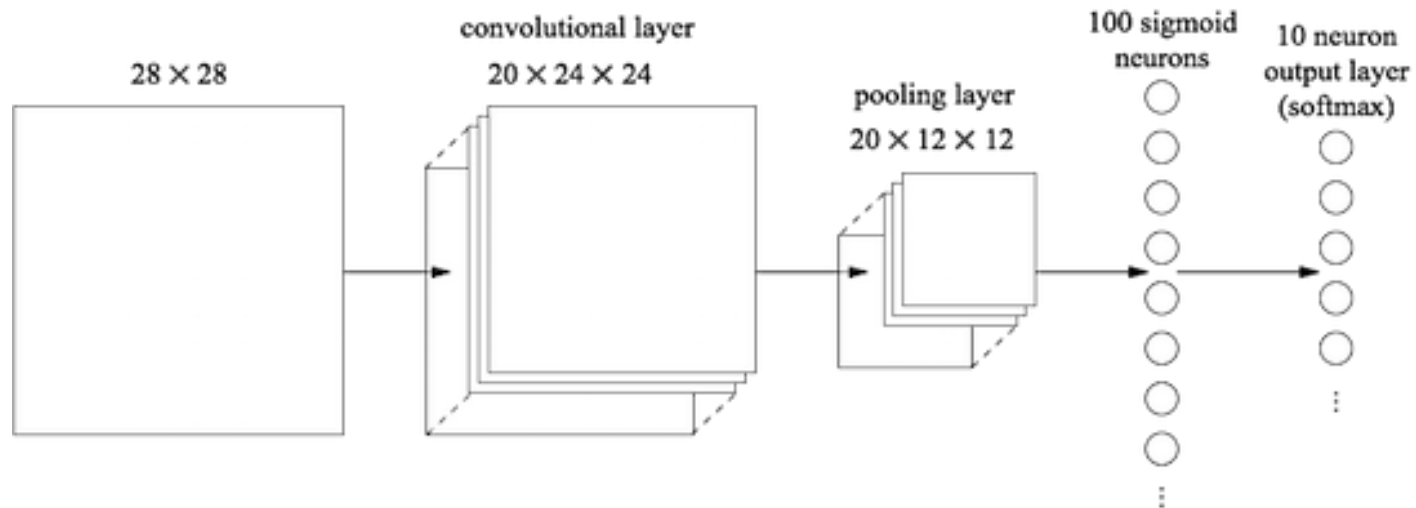
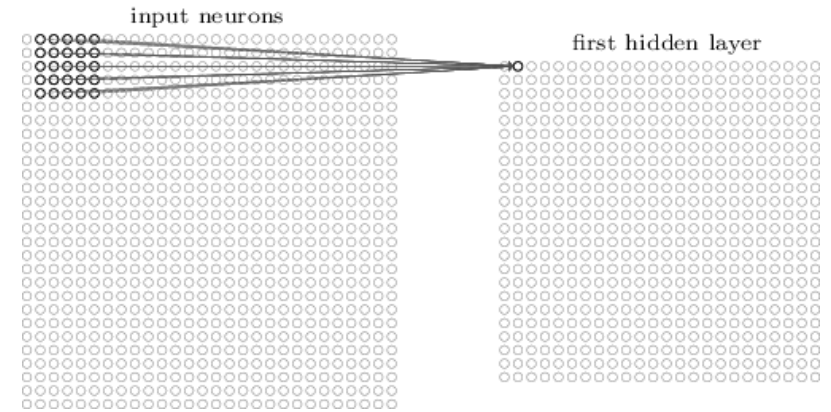
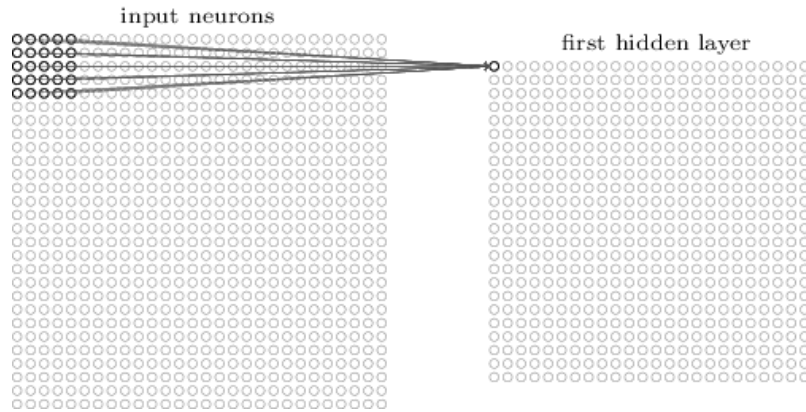


## 06. Convolutional Neural Networks



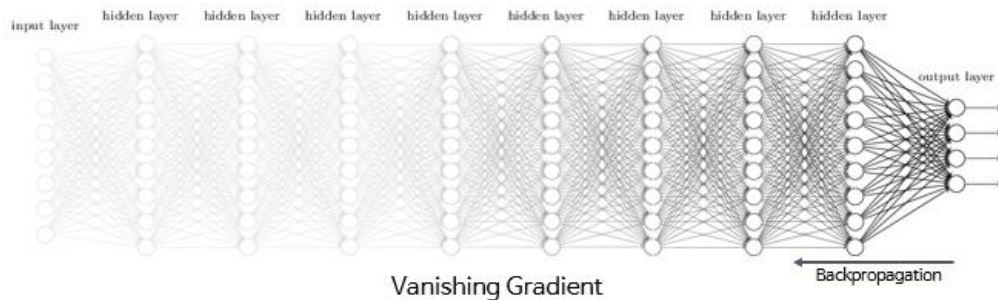
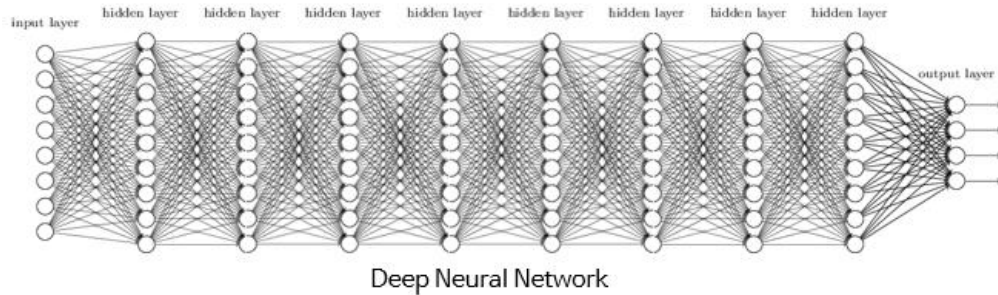
“At some point in the late 1990s,  
one of these systems was reading 10 to 20% of all the checks in the US.”

## 06. Convolutional Neural Networks



## 07. 두 번째 빙하기

- 신경망의 깊이가 깊어질수록 원하는 결과를 얻을 수 없다.



Layer가 일정 개수 이상 늘어나면 기대하는 결과가 나오지 않는 현상 발견

⇒ Backpropagation을 수행할 때 입력층에서 멀리 떨어진 깊은 (deep) layer에 이르게 되면 기울기 (gradient)가 급속히 작아지거나 너무 커져 발산해 버리는 **Vanishing Gradient** 문제가 발생

## 07. 두 번째 빙하기

- 신경망 학습을 위한 파라미터 값의 최적화에 대한 이론적인 근거가 없었다.

⇒ 좋은 성능을 이끌어내기 위한 파라미터들에 대한 노하우는 있었지만 이론적인 근거가 없었다.

- 다른 머신러닝 알고리즘들의 등장

⇒ SVM(Support Vector Machine)이나 RandomForest 등 새로운 기계학습 알고리즘들이 등장

⇒ Neural network보다 새로운 기계학습 알고리즘의 성능이 더 좋음



## 08. Breakthrough

- 2006 년 Hinton, Simon Osindero 외 "깊은 믿음의 빠른 학습 알고리즘"
- 2007 년 Yoshua Bengio 외. "Deep Networks의 Greedy Layer-Wise Training"
- 가중치가 무작위보다는 영리한 방법으로 초기화되는 경우 많은 레이어가 있는 신경망을 실제로 잘 훈련 할 수 있다.
- 깊은 기계 학습 방법은 얇은 방법보다 어려운 문제에 더 효율적
- Rebranding to **Deep Nets, Deep Learning**

## 09. IMAGENET Large Scale Visual Recognition Challenge

### IMAGENET Large Scale Visual Recognition Challenge (ILSVRC) 2017

## ILSVRC Image Classification (CLS) Task

Steel drum



**Output:**  
Scale  
T-shirt  
Steel drum  
Drumstick  
Mud turtle



**Output:**  
Scale  
T-shirt  
Giant panda  
Drumstick  
Mud turtle



## 09. IMAGENET Large Scale Visual Recognition Challenge

- ILSVRC Image Localization (LOC) Task

### Steel drum



## 09. IMAGENET Large Scale Visual Recognition Challenge

Steel drum



Correct



Bad localization

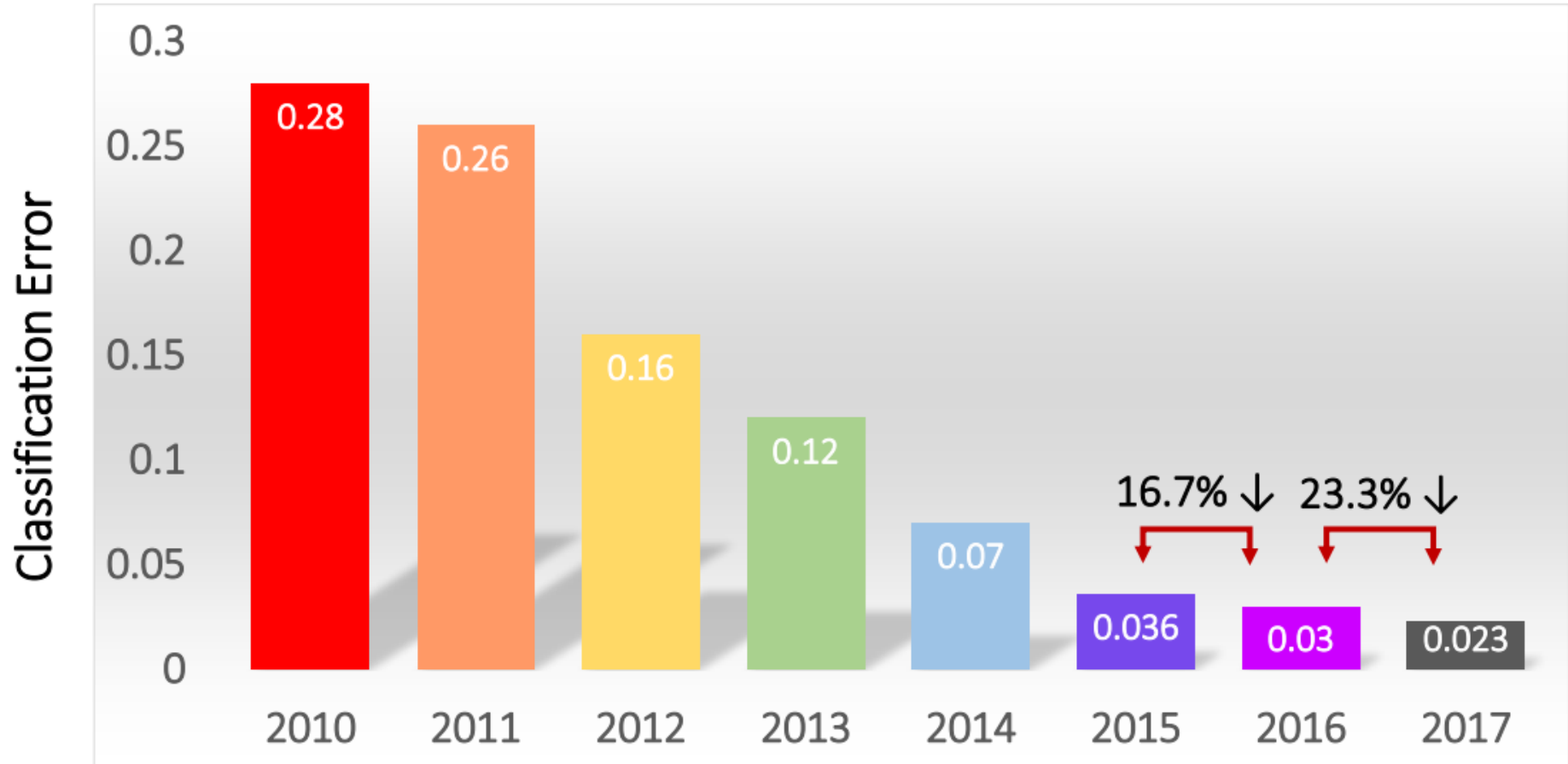


Bad classification



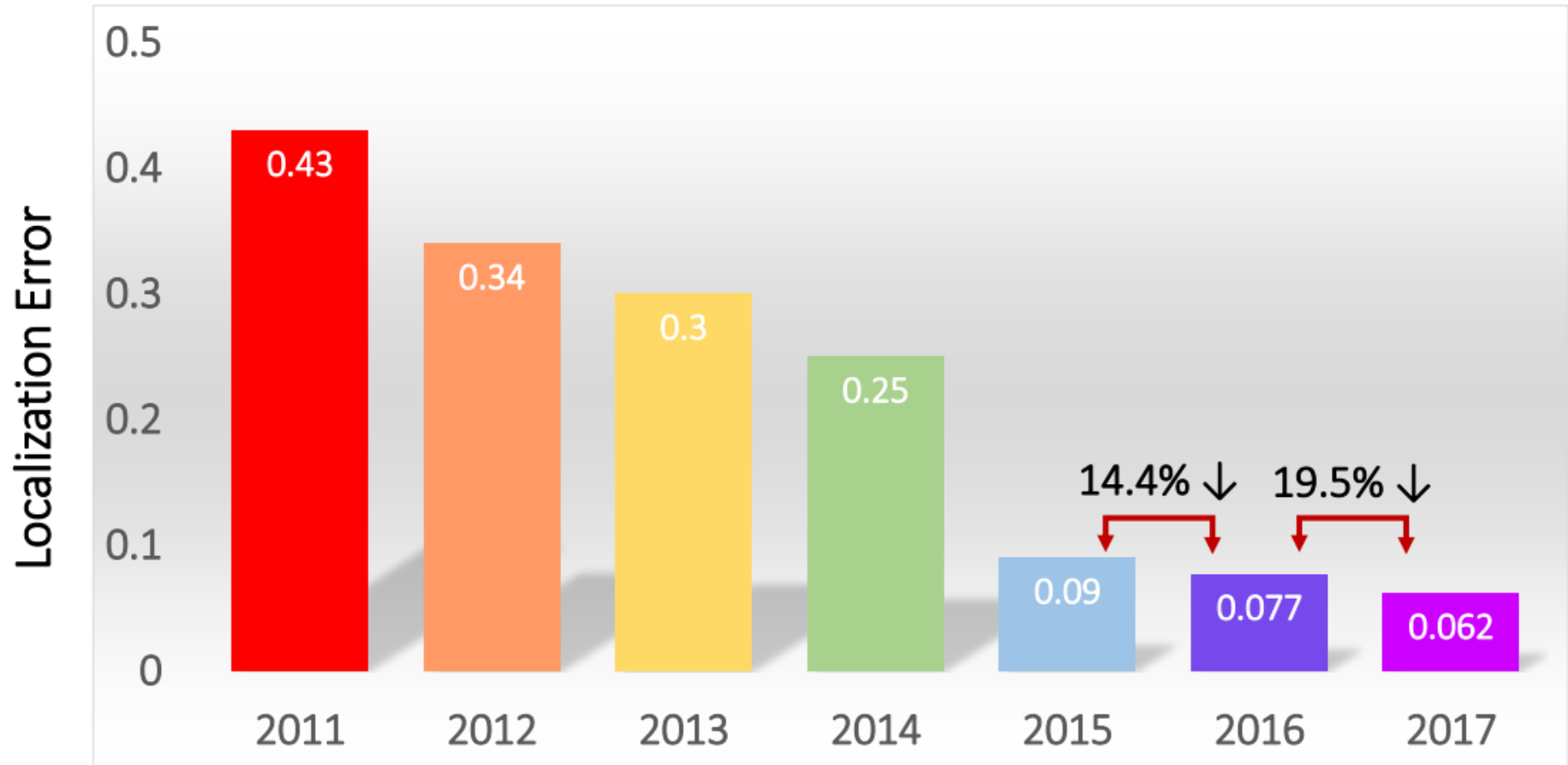
## 09. IMAGENET Large Scale Visual Recognition Challenge

- Classification Results (CLS)



## 09. IMAGENET Large Scale Visual Recognition Challenge

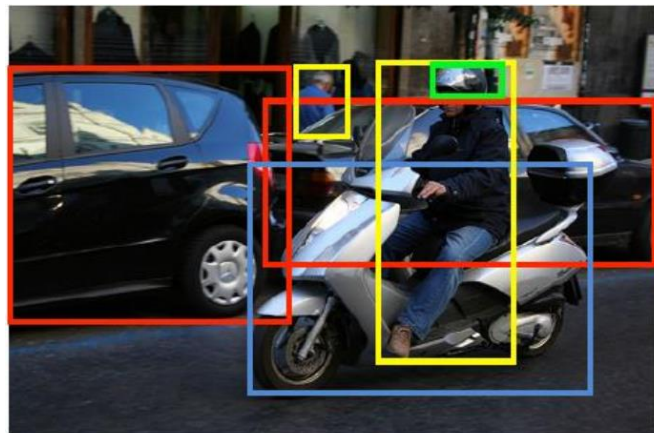
- Localization Results (LOC)





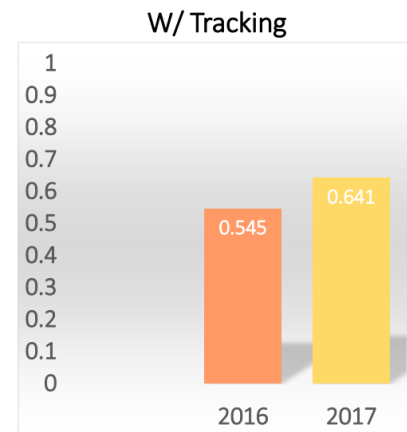
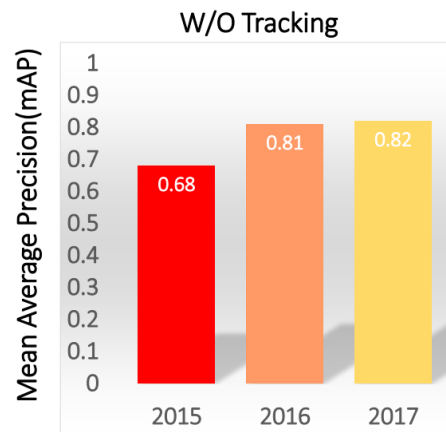
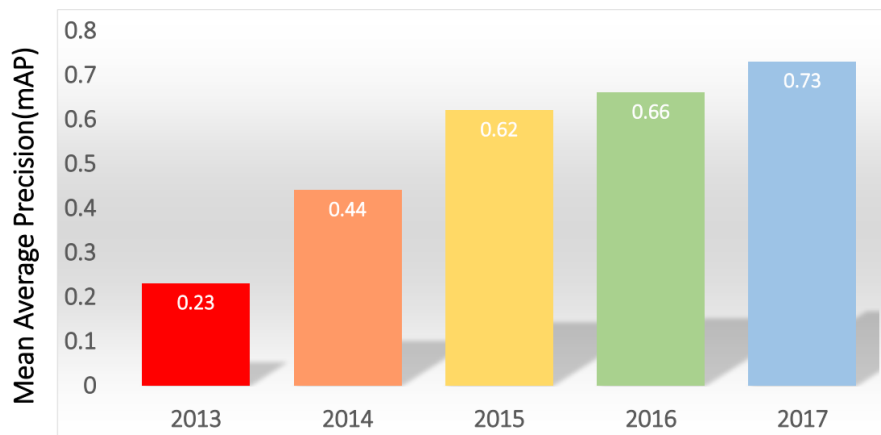
## 09. IMAGENET Large Scale Visual Recognition Challenge

- ILSVRC Object Detection (DET) Task



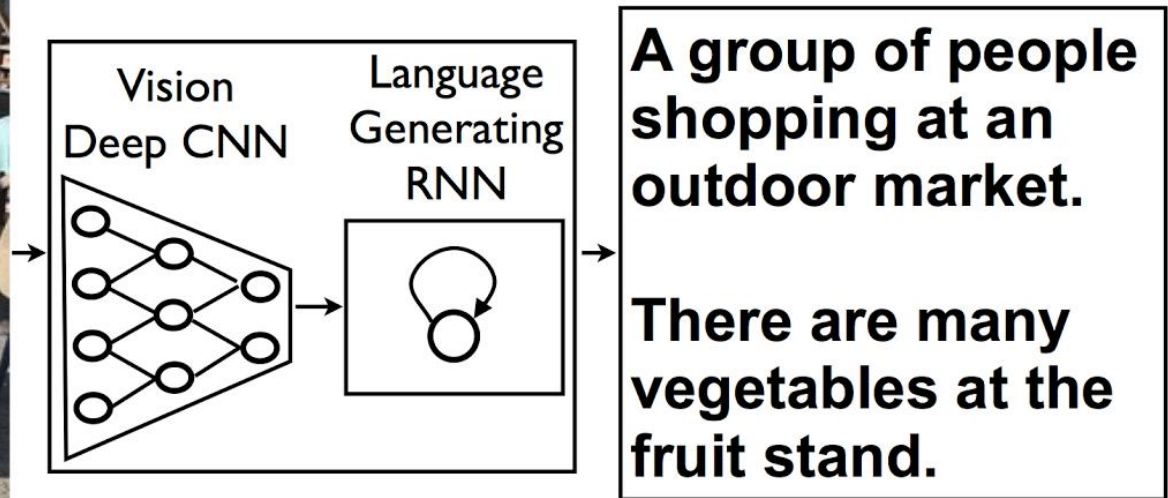
Person  
Car  
Motorcycle  
Helmet

- Object Detection from Video(VID) Task



## 10. neural networks that can explain photos

- Google, Stanford build hybrid neural networks that can explain photos



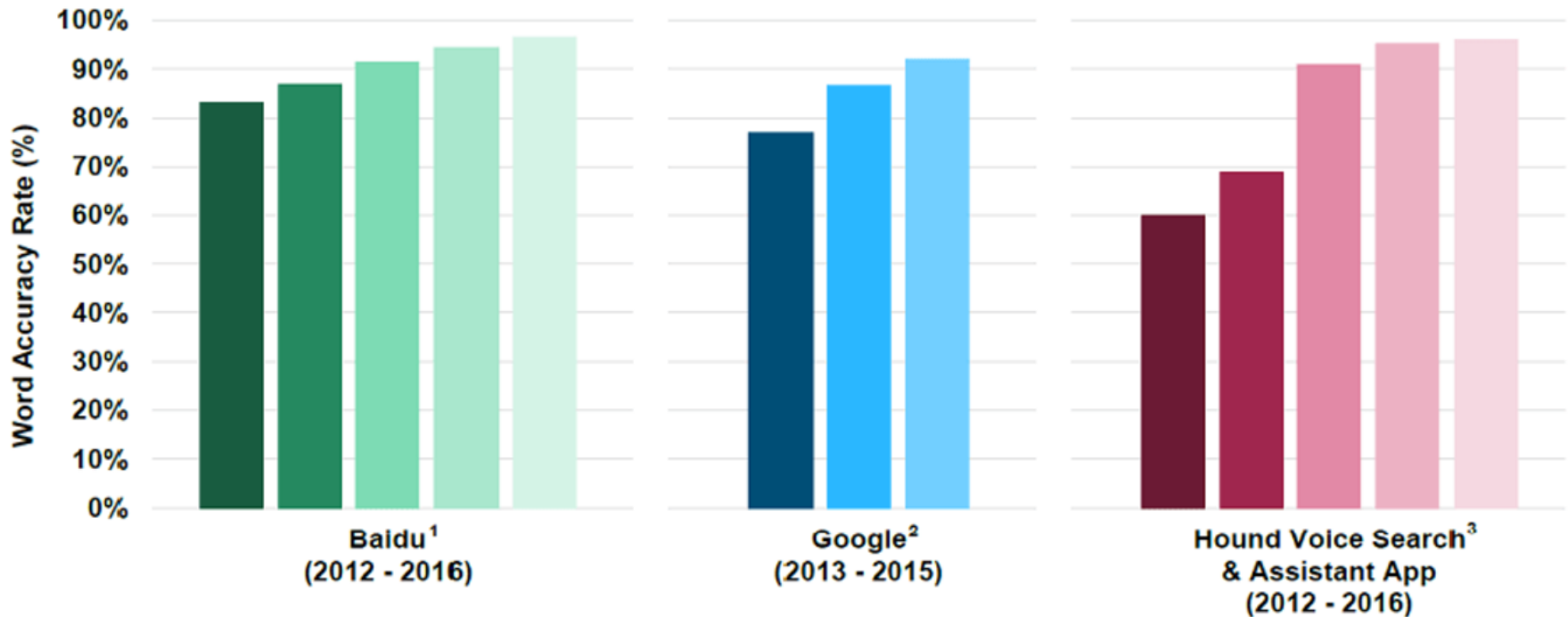
<https://gigaom.com/2014/11/18/google-stanford-build-hybrid-neural-networks-that-can-explain-photos/>



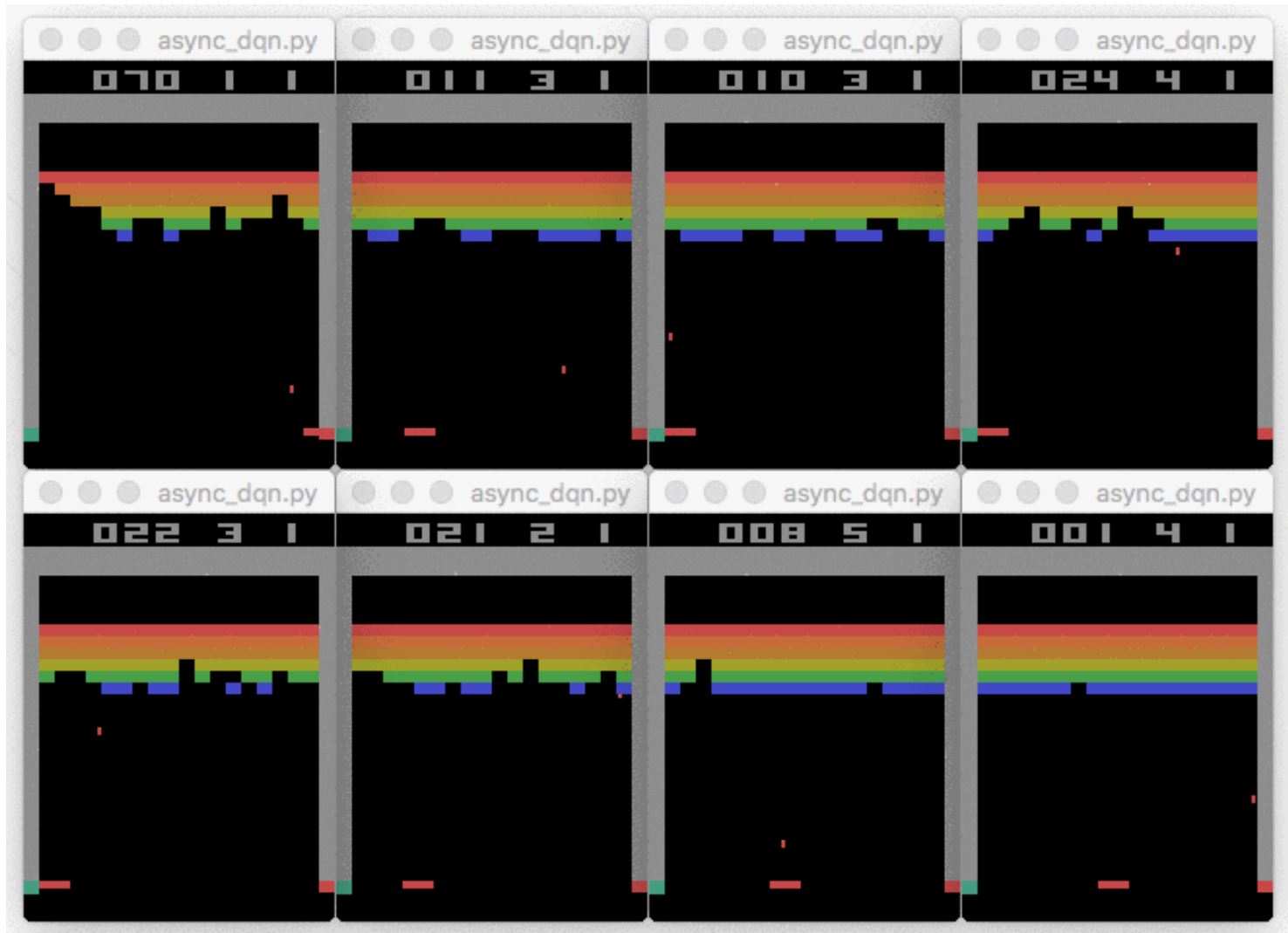
# 11. Speech Recognition

## Word Accuracy Rates by Platform\*, 2012 – 2016

*\*Word accuracy rate definitions are unique to each company...see footnotes for more details*



## 12. 강화학습



## 12. 강화학습



[https://www.youtube.com/watch?v=-hMvDI\\_tLNk](https://www.youtube.com/watch?v=-hMvDI_tLNk)

**BONUSTIME**

Score: 1,755,765

Progress: 283/800

**Jump**

**Slide**

Previous Screen: 0x001700000000

Address	Value	Address	Value	Address	Value
0x001700000000	0x00000000	0x001700000001	0x00000000	0x001700000002	0x00000000
0x001700000003	0x00000000	0x001700000004	0x00000000	0x001700000005	0x00000000
0x001700000006	0x00000000	0x001700000007	0x00000000	0x001700000008	0x00000000
0x001700000009	0x00000000	0x00170000000A	0x00000000	0x00170000000B	0x00000000
0x00170000000C	0x00000000	0x00170000000D	0x00000000	0x00170000000E	0x00000000
0x00170000000F	0x00000000	0x001700000010	0x00000000	0x001700000011	0x00000000
0x001700000012	0x00000000	0x001700000013	0x00000000	0x001700000014	0x00000000
0x001700000015	0x00000000	0x001700000016	0x00000000	0x001700000017	0x00000000
0x001700000018	0x00000000	0x001700000019	0x00000000	0x00170000001A	0x00000000
0x00170000001B	0x00000000	0x00170000001C	0x00000000	0x00170000001D	0x00000000
0x00170000001E	0x00000000	0x00170000001F	0x00000000	0x001700000020	0x00000000
0x001700000021	0x00000000	0x001700000022	0x00000000	0x001700000023	0x00000000
0x001700000024	0x00000000	0x001700000025	0x00000000	0x001700000026	0x00000000
0x001700000027	0x00000000	0x001700000028	0x00000000	0x001700000029	0x00000000
0x00170000002A	0x00000000	0x00170000002B	0x00000000	0x00170000002C	0x00000000
0x00170000002D	0x00000000	0x00170000002E	0x00000000	0x00170000002F	0x00000000
0x001700000030	0x00000000	0x001700000031	0x00000000	0x001700000032	0x00000000
0x001700000033	0x00000000	0x001700000034	0x00000000	0x001700000035	0x00000000
0x001700000036	0x00000000	0x001700000037	0x00000000	0x001700000038	0x00000000
0x001700000039	0x00000000	0x00170000003A	0x00000000	0x00170000003B	0x00000000
0x00170000003C	0x00000000	0x00170000003D	0x00000000	0x00170000003E	0x00000000
0x00170000003F	0x00000000	0x001700000040	0x00000000	0x001700000041	0x00000000
0x001700000042	0x00000000	0x001700000043	0x00000000	0x001700000044	0x00000000
0x001700000045	0x00000000	0x001700000046	0x00000000	0x001700000047	0x00000000
0x001700000048	0x00000000	0x001700000049	0x00000000	0x00170000004A	0x00000000
0x00170000004B	0x00000000	0x00170000004C	0x00000000	0x00170000004D	0x00000000
0x00170000004E	0x00000000	0x00170000004F	0x00000000	0x001700000050	0x00000000
0x001700000051	0x00000000	0x001700000052	0x00000000	0x001700000053	0x00000000
0x001700000054	0x00000000	0x001700000055	0x00000000	0x001700000056	0x00000000
0x001700000057	0x00000000	0x001700000058	0x00000000	0x001700000059	0x00000000
0x00170000005A	0x00000000	0x00170000005B	0x00000000	0x00170000005C	0x00000000
0x00170000005D	0x00000000	0x00170000005E	0x00000000	0x00170000005F	0x00000000
0x001700000060	0x00000000	0x001700000061	0x00000000	0x001700000062	0x00000000
0x001700000063	0x00000000	0x001700000			



## 13. Geoffrey Hinton's 지금까지의 발견요약

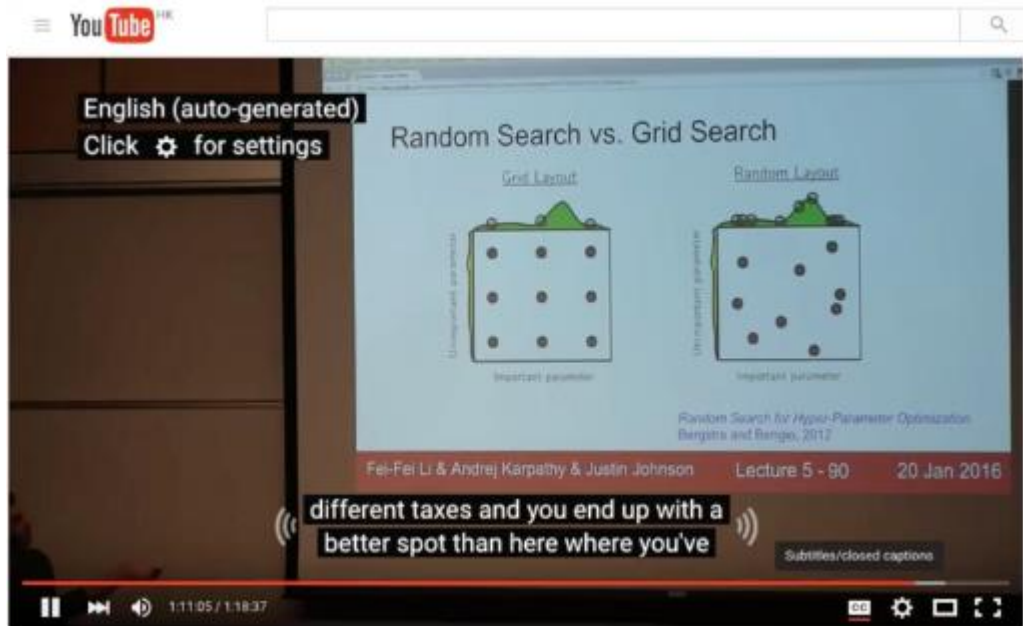
- **labeled** 데이터셋(학습용 데이터셋)이 너무 작았음
- 옛날엔 컴퓨터가 너무 느렸음
- 초기화가 부적절
- non-linearity(sigmoid)를 잘못 사용했다.

## 14. 왜 머신러닝을 공부해야하는가?

- 데이터를 갖고 있다면 누구라도 머신러닝을 활용할 수 있다.
- 연구자나 프로그래머가 아니라도 머신러닝을 활용가능

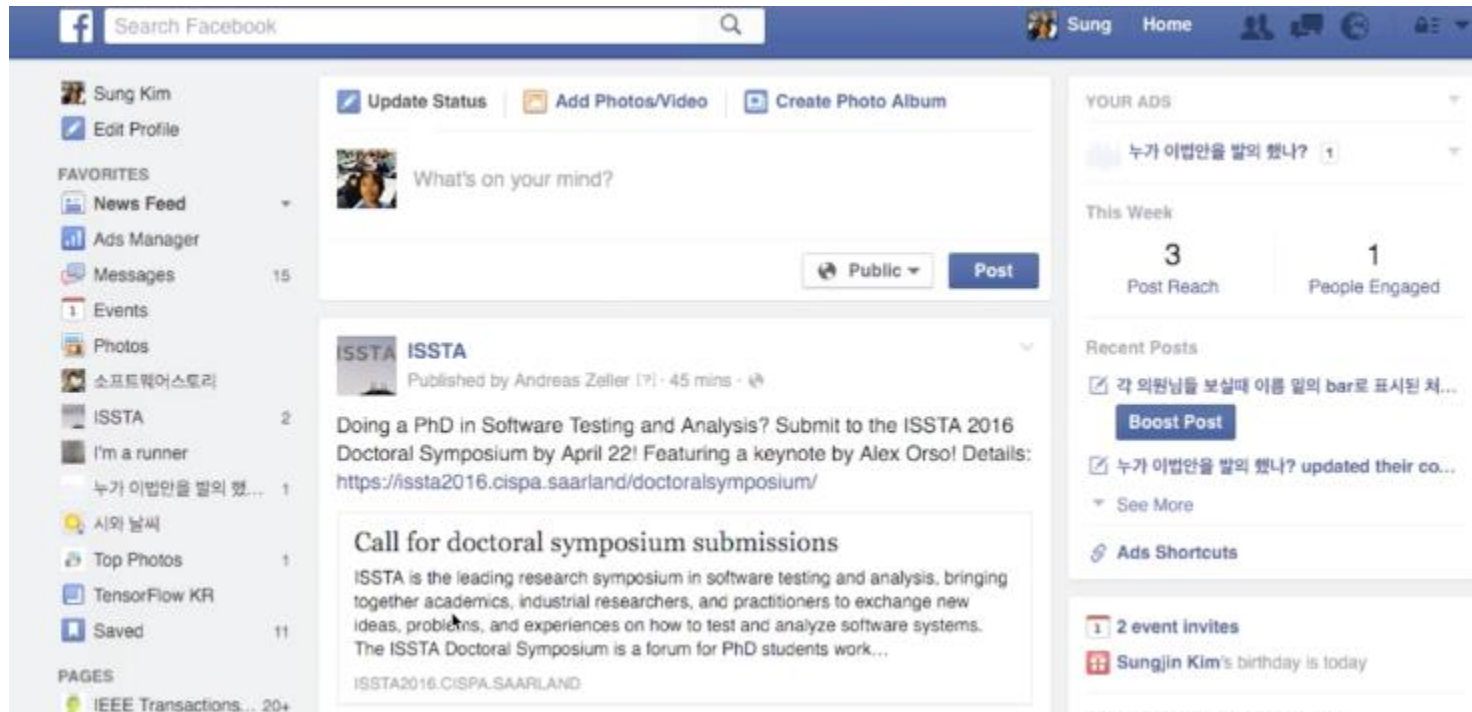
## 29.머신런닝의 활용

- 음성을 인식해서 자동으로 표시되는 유튜브 자막



## 29.머신러닝의 활용

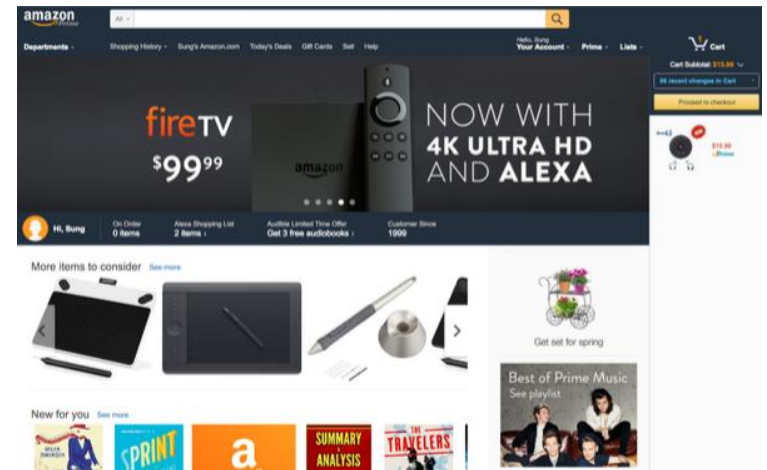
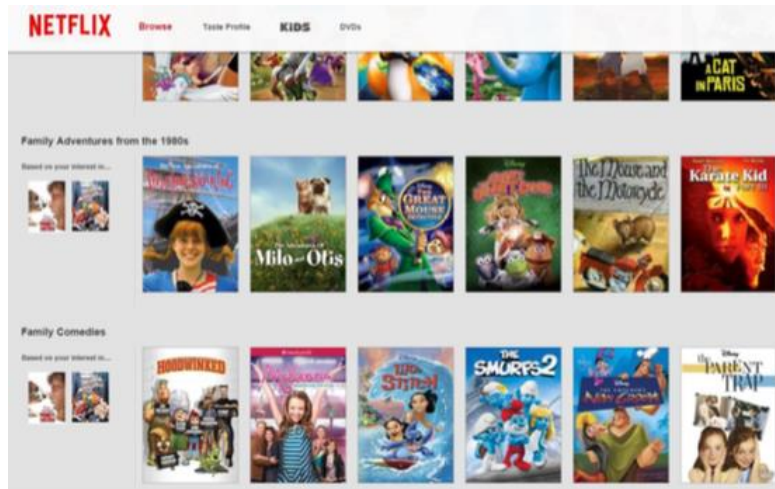
- 친구들이 올린 내용 중에서 내가 좋아할 만한 것만 추천하는 페이스북 뉴스피드 시스템.





## 29.머신러닝의 활용

- 구글 검색 시스템. 내가 찾고 싶은 것을 상위에 표시해 줌
- 내가 보고싶은 영화를 넷플릭스가 알려 줌
- 아마존의 제품추천 시스템



## 30. Why Now?

- Students/Researchers
  - Not too late to be a world expert
  - Not too complicated (mathematically)
- Practitioner
  - Accurate enough to be used in practice
  - many ready-to-use tools such as TensorFlow
  - Many easy/simple programming languages such as Python
- After all, it is fun!