

Machine Learning HWO

TAs
ntu.mlta@gmail.com

Outline

- ❖ 加簽規則
- ❖ 作業說明
- ❖ GitHub設定

加簽規則

- ❖ 欲加簽者需在**明天（9/16）中午12:00前**完成HW0，依照指定方式繳交，未完成者無法加簽。
- ❖ 助教批改完之後會寄授權碼至同學學校信箱
 - 收信時會收到一份**文件及全部授權碼的照片**，每張授權碼都有編號，請在文件中**確認自己學號對應的編號**，並依照該編號找到對應的授權碼並至選課系統加簽。

HWO

- ❖ HWO希望確保同學們能夠執行機器學習所需的一些基本操作。
- ❖ 往後的作業會用GitHub繳交，每位同學要有GitHub帳號並將repository設為**private**。
- ❖ 請依照規定格式繳交，若是格式不符視同錯誤，請不要來跟助教爭論。
- ❖ 助教批改環境：Linux
- ❖ 資料位置：<https://ppt.cc/fzdYXx>

Q1 出現字數統計

1. 讀取**words.txt**中的所有英文單字，英文單字之間皆由
<space>做分隔
2. 按照單字出現的順序，給予編號（0, 1, 2 ...）。(不要跳號)
3. 計算單字出現的次數。
4. 將得到編號和次數，由**單字出現在words.txt的順序**輸出至
Q1.txt，最後一行不要換行，每一行皆為：

<單字><space><編號><space><出現次數>
(輸出範例參考下頁)
5. **Ntu, ntu** 為不同單字

Q1 出現單字統計 - 輸出範例

words.txt :(助教給的檔案)

ntu ml mlds ml ntu ntuee

Q1.txt : (輸出的檔案)

ntu 0 2

ml 1 2

mlds 2 1

ntuee 3 1

Q2 圖片淡化

1. 讀取 westbrook.jpg
2. 將每個pixel的RGB數值都減半 (ex: (122, 244, **245**) -> (61, 122, **122**)) ，再將圖片輸出為 Q2.jpg
3. RGB數值記得要去掉小數點！(無條件捨去)



繳交格式

`./Q1.sh <DATA_FILE_PATH>`

輸出檔名: Q1.txt

`./Q2.sh <IMAGE_FILE_PATH>`

輸出檔名: Q2.jpg

將這兩個script以及你的程式放在**ML2017FALL/hw0/**底下

ML2017FALL是助教改作業的repository，hw0是本次作業的資料夾名稱，
請大家將作業放在正確的地方，沒放正確是不會被批改的，位置如下：

ML2017FALL/hw0/<your files>

填Github Repo URL表單: <https://ppt.cc/fB0g9x>

GitHub

開設GitHub帳號

1. GitHub: <https://github.com/> 使用學校信箱開帳號
 - a. 學校信箱可免費使用**private**功能
 - b. 可綁定多個信箱
2. 申請學生版的附加功能
 - a. 網址: <https://education.github.com/>
 - b. 點選 Request a discount
 - c. 輸入資料，靜候佳音

step 1.進入網址

The screenshot shows the GitHub Education homepage. At the top left is the GitHub logo followed by "Education". To the right are navigation links: "Stories", "Events", "Student pack", "Classroom", "Community", and "Contact us". The main background is a chalkboard with a green border. On the left, there's a white chalk drawing of laboratory glassware: a round-bottom flask on a stand with a condenser tube leading to an Erlenmeyer flask. In the center, the text "TEACH AND LEARN BETTER, TOGETHER" is written in a white, hand-drawn style. Below it is a green button with white text that says "Request a discount". To the right, there's a cartoon illustration of a scientist wearing a white lab coat, a black hood, and a clear safety mask with large eyes. The scientist is holding a test tube with blue liquid and a small yellow beaker. At the bottom left, there's a yellow backpack with the GitHub logo on the front pocket. The bottom section has a white background with a thin grey horizontal line separating it from the chalkboard. The word "STUDENT DEVELOPER PACK" is centered above the backpack. To the right of the backpack, the text "Get the Student Developer Pack" is displayed in a large, dark font, followed by a smaller line of text: "Dozens of free resources from great companies to help students learn." Below this is a blue button with white text that says "Get the pack". At the very bottom, there are two small, faint horizontal lines with the words "STORIES" and "CLASSROOM" respectively.

GitHub Education

Stories Events Student pack Classroom Community Contact us

TEACH AND LEARN BETTER, TOGETHER

Request a discount

STUDENT DEVELOPER PACK

Get the Student Developer Pack

Dozens of free resources from great companies to help students learn.

Get the pack

STORIES

step 2. 填入資料

Discounted and free plans are available for educational use

You have an active discount on your account. If your current coupon is still active when this request is approved, it will be replaced. There should be no lapse in access to any of your private repositories.

Step 1

Tell us what you need

Step 2

Tell us about you

Name

zweilin314

Verify academic status

Select your school-issued email address:

r0494211@ntu.edu.tw

If your school-issued email address isn't listed, please [add and verify it](#), then refresh this page.

School name

Nation Taiwan University

Graduation year

2017

step 3.靜候佳音

GitHub Education

Stories

Events

Student pack

Classroom

Community

Contact us

Request a discount

Thanks for submitting!

You should be getting an email from us in a few weeks.

Have an Octotastic day!

© 2016 GitHub, Inc. [Terms](#) [Privacy](#) [Security](#) [Contact](#)



[@GitHubEducation](#) [Status](#) [Blog](#) [About](#)

作業繳交

1. New repository

- a. 請將名稱取為 **ML2017FALL**
- b. 往後所有的作業程式都會在這個路徑下被批改
- c. 權限請設為**private**

2. 將助教帳號加入存取權

- a. 名稱: **ntumIta**

Create New Repository

Overview

Repositories 0

Stars 0

Followers 0

Following 0

Search repositories...

Type: All ▾

New

Owner

Repository name



/ ML2017FALL



Great repository names are short and memorable. Need inspiration? How about [ideal-sniffle](#).

Description (optional)



Public

Anyone can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.

Initialize this repository with a README

This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: **None** ▾

Add a license: **None** ▾



Add TA account to Collaborators

The screenshot shows a GitHub repository page with the following interface elements:

- Top Navigation:** Code, Issues 0, Pull requests 0, Projects 0, Wiki, Pulse, Graphs, Settings (the "Settings" button is circled in red).
- Left Sidebar (Options):** Options, Collaborators (circled in red), Branches, Webhooks, Integrations & services, Deploy keys.
- Collaborators Section:** Title "Collaborators", Subtitle "Push access to the repository", Message "This repository doesn't have any collaborators yet. Use the form below to add a collaborator.", Search bar "Search by username, full name or email address", Note "You'll only be able to find a GitHub user by their email address if they've chosen to list it publicly. Otherwise, use their username instead.", Input field containing "ntumlta" (circled in red), and a "Add collaborator" button.

Get Your Git Repo URL

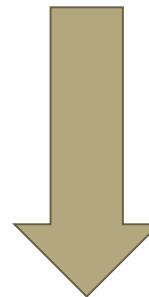
The screenshot shows a GitHub repository page for 'ML2017'. At the top, there are summary statistics: 1 commit, 1 branch, 0 releases, and 1 contributor. Below these are buttons for 'Branch: master ▾', 'New pull request', 'Create new file', 'Upload files', 'Find file', and a prominent green 'Clone or download ▾' button. A red oval highlights the 'Clone or download' button. To its right is a 'Clone with SSH' section with a link 'git@github.com:ntumlta/ML2017.git', which is also highlighted by a red oval. Below this are links for 'Open in Desktop' and 'Download ZIP'. The main content area shows a single commit by user 'ntumlta' with the message 'Initial commit'. There are two 'README.md' files listed.

填Git Repo URL表單：

<https://goo.gl/forms/XtzWWLoENzhk9FSi2>

GitHub

1. Open your terminal.
2. \$git clone *git repo url*
3. \$cd ML2017FALL
4. \$mkdir hw0
5. ...
6. ...



git clone repository

將整個repository clone
到本機之後即可編輯。
mkdir, vim, etc

GitHub

1. \$git add xxx.py

(請確保所有作業所需檔案
都被成功加入repository)

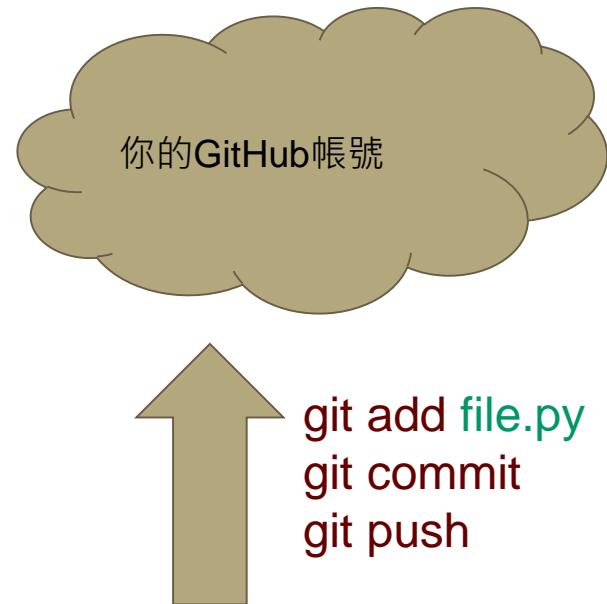
(Q1.sh, Q2.sh, Q1.py Q2.py)

2. \$git commit

3. \$git push

4. 在GitHub網頁上確認master是否已更新

更多：<https://goo.gl/NRRCLm>



Windows系統怎麼辦

安裝virtual machine

- a. 優：用起來跟Linux一樣，有問題可以詢問助教
- b. 劣：花一點時間安裝

該使用什麼程式語言

限定使用**python3.5**以上

可以使用（ import ）什麼套件

Q1：無

Q2：PIL

請勿在作業中import其他套件

Q&A

ntu.mlta@gmail.com

Course Policy

李宏毅

Hung-yi Lee

評量方式

- 不點名、不考試
- 作業 (60%)：沒有分組、每個人都要繳交
- 期末專題 (40%)：分組進行
- 會有各種加分

成績是相對的

成績是相對的

成績是相對的

評量方式 - 作業 (60%)

- 作業一 (10%) : 預測 PM2.5 (9/29 公告)
- 作業二 (10%) : 猜測年收入
- 作業三 (10%) : 人臉表情辨識
- 作業四 (10%) : 降維
- 作業五 (10%) : 文字情緒分析
- 作業六 (10%) : 電影推薦

評量方式 - 作業 (60%)

- 程式碼：程式碼要符合指定格式可以順利執行，經助教要求修改後才能執行會被扣分
- 課堂內競賽成績：同學上傳程式執行結果到競賽專用平台 Kaggle，可以即時得知成果
 - 達到 baseline 就得到大部分的分數
 - 課堂內競賽成績優異的同學會被邀請在上課時間發表，會有額外的加分。
 - 課堂內競賽視同考試，嚴禁任何作弊行為
 - 在機器學習過程中使用禁止使用的資料，如測試資料(視同考試攜帶小抄)
 - 註冊多重分身參加比賽(視同考試請人代考)
- 繳交報告回答問題

評量方式 - 期末專題 (40%)

- 2 ~ 4人一組
- 找不到隊友也沒關係，會幫忙配對
- 進行方式：會公告幾個可能的題目給同學們選擇，其餘規定同作業
- 最後會有組內互評

FB 社團 / 課程網頁

- 課程網頁：
http://speech.ee.ntu.edu.tw/~tlkagk/courses_ML17_2.html
- 社團：“Machine Learning (2017, Fall)”
 - <https://www.facebook.com/groups/426161884451961/>
- 有問題最有效率的方式是直接在 FB 社團上發問

上課方式

- 本學期採用翻轉教學，上課投影片和錄音會放到課程網頁上，請自行在家觀賞
- 那上課要做什麼呢？
• 公告作業
• 在作業截止的前一週，由助教講解作業實作方式
• 問老師、助教問題，和同學交換情報
• 在上課時間完成作業，有問題立刻問「小老師」
• 表現優異同學的發表成果
• 外賓演講
- 上課時間：週五上午 **10:20 – 12:10**，地點明達205
- 下週五 (9/22) : python 教學

什麼是小老師

- 小老師的工作：在上課時間待在教室協助同學解決作業問題
- 每次加總成績一分
- 如何申請成為小老師？
 - 當需要小老師時，助教會在社團上用表單徵求小老師
 - 資格：作業公告後第一週就超過 baseline 或第二週超過 top baseline
 - 排序：當過次數少的同學優先，如果當過次數相同，就看 Kaggle 上的排名
- 期末由同學們票選出最佳小老師五位，總成績另外加三分

加簽 – 方式一

- 參加「科技大擂台」至少上傳一次結果



請填寫以下表單：

<https://goo.gl/forms/R6c6v dbsKzyb3SBW2>

跟科技部的名單核對之後以 e-mail 發給受權碼

加簽 - 方式二



國立臺灣大學
National Taiwan University

電信工程學研究所丙組

資料科學與智慧網路組



Python
Machine Learning
Data Scientist
Gamification
Biometrics
MapReduce
Algorithm
NTU GICE
Data Science and
Smart Networking
Elasticsearch Predictive Analytics
Deep Learning
Cassandra Logstash
Kibana

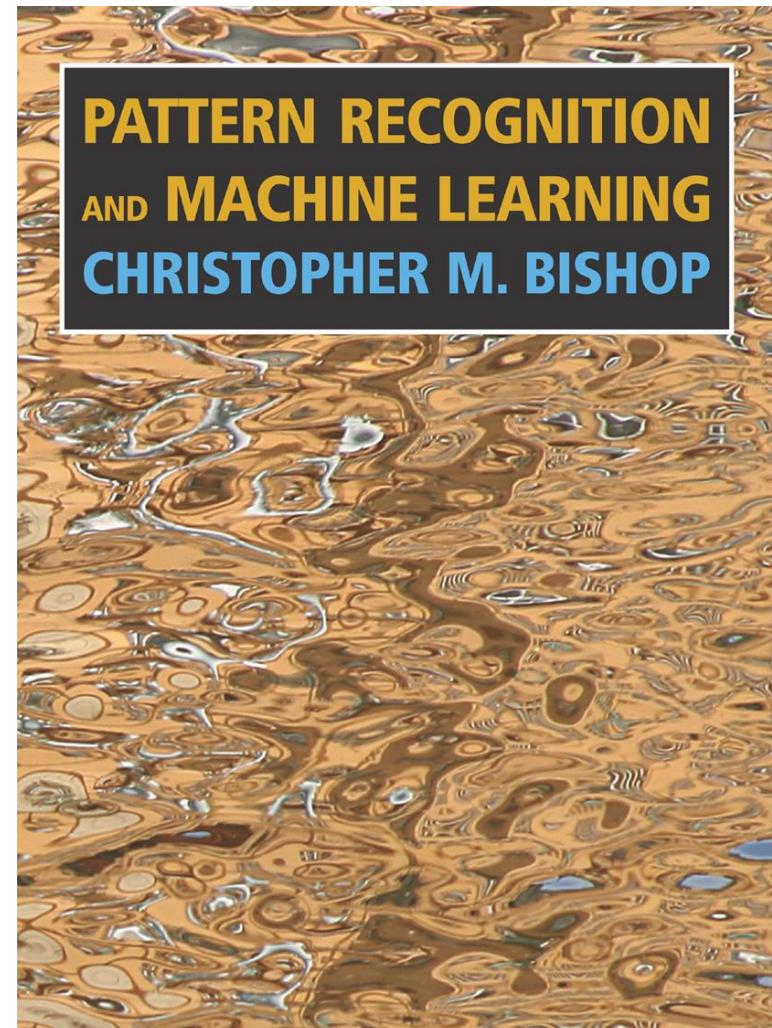
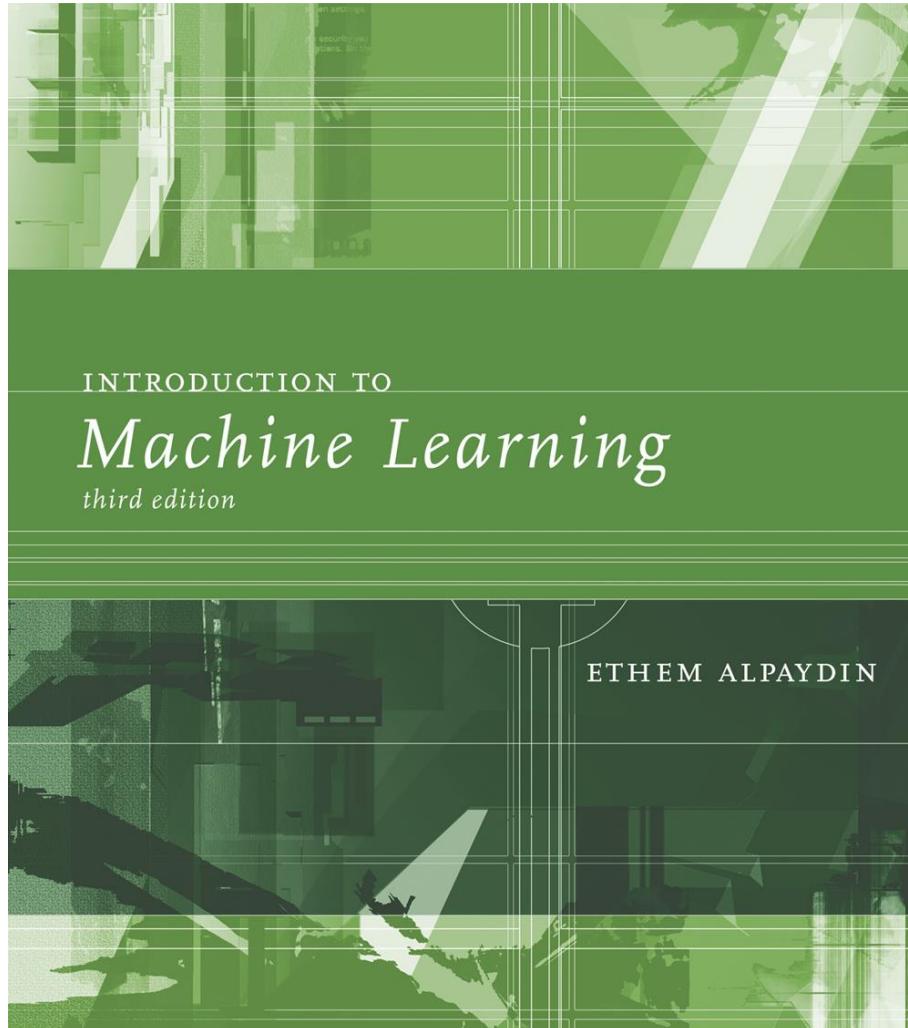
1

參加資網組相關活動

加簽 – 方式三

- 等一下助教會公告作業 0
 - 作業 0 跟機器學習無關，只是測驗基礎程式能力
 - 正確完成後會取得授權碼
- 請仔細聽助教說明

參考書籍



課程定位

- 在課程內容上儘量和電資學院的其它機器學習相關課程互補
- 課程特色
 - 以深度學習為主軸
 - 強調實作
- Q: 和 “Machine Learning and having it Deep and Structured” (MLDS) 有何不同？
 - ML 和 MLDS 內容互補不重複

助教

- TA 信箱 : ntu.mlta@gmail.com



大 助教 李佳軒

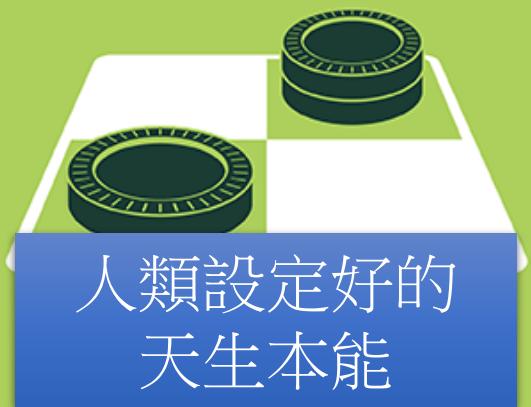
Introduction of this course

李宏毅

Hung-yi Lee

ARTIFICIAL INTELLIGENCE

Early artificial intelligence stirs excitement.



人工智慧 目標

MACHINE LEARNING

Machine learning begins to flourish.



機器學習 手段

DEEP LEARNING

Deep learning breakthroughs drive AI boom.



深度學習



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.

人類設定好的天生本能

- E.g. You want to build a Chat-bot ...
 - If there is “turn off” in the input, then “turn off the music” (hand-crafted rules)
 - You can say “Please turn off the music” or “Can you turn off the music?”. Smart?
 - What if someone says “Please don’t turn off the music”
- Weakness of hand-crafted rules
 - Hard to consider all possibilities
 - 永遠無法超越創造者
 - Lots of human efforts (not suitable for small industry)

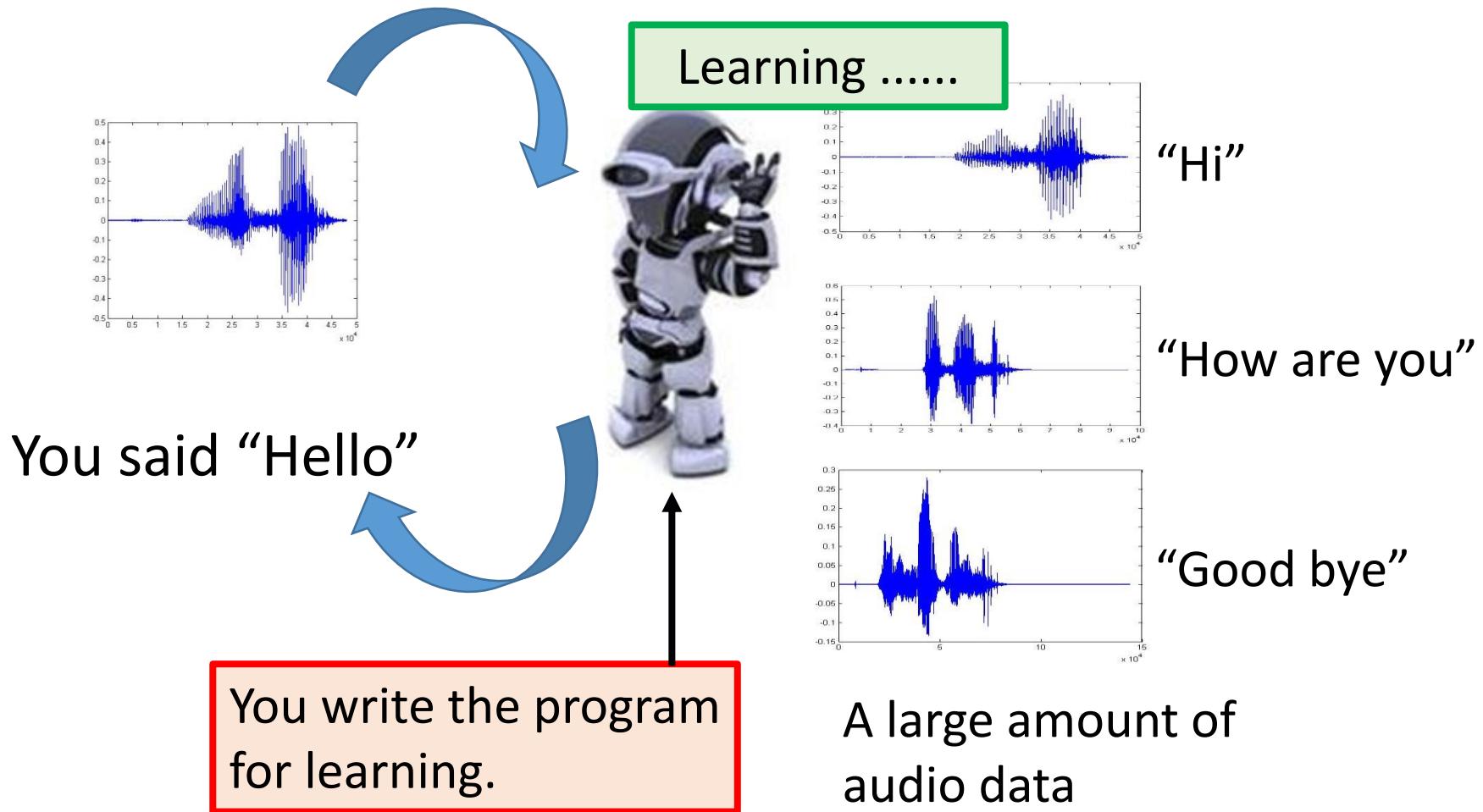
人類設定好的天生本能

- AI?

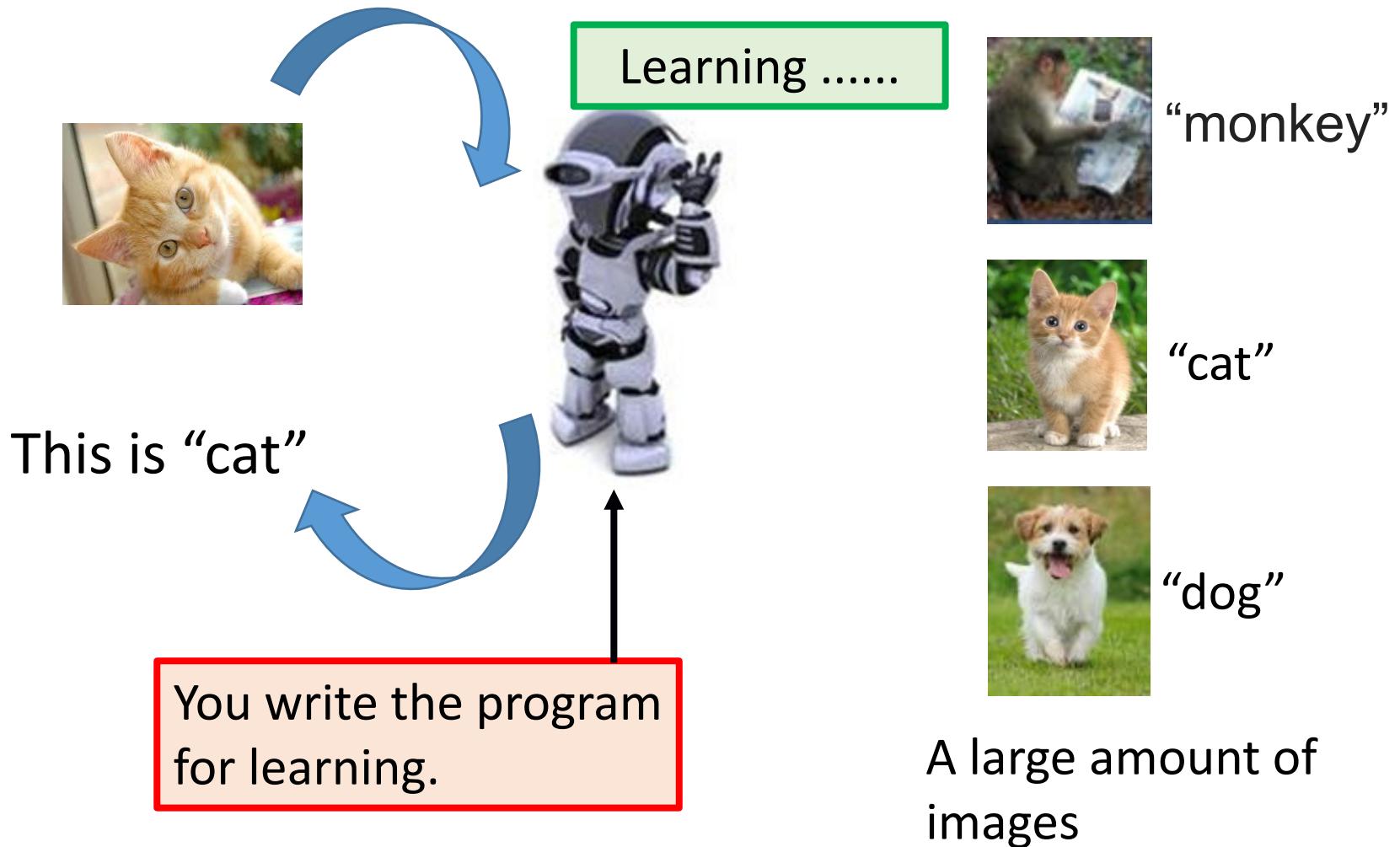
<http://www.commitstrip.com/en/2017/06/07/ai-inside/?>

Shared on Yann
LeCun's FB

What is Machine Learning?



What is Machine Learning?



Machine Learning ≈ Looking for a Function

- Speech Recognition

$$f\left(\begin{array}{c} \text{[audio waveform plot]} \\ \text{[blue waveform on white background]} \end{array} \right) = \text{“How are you”}$$

- Image Recognition

$$f\left(\begin{array}{c} \text{[image of a kitten]} \\ \text{[orange tabby kitten looking up]} \end{array} \right) = \text{“Cat”}$$

- Playing Go

$$f\left(\begin{array}{c} \text{[image of a Go board]} \\ \text{[black and white stones on a grid]} \end{array} \right) = \text{“5-5”} \quad \text{(next move)}$$

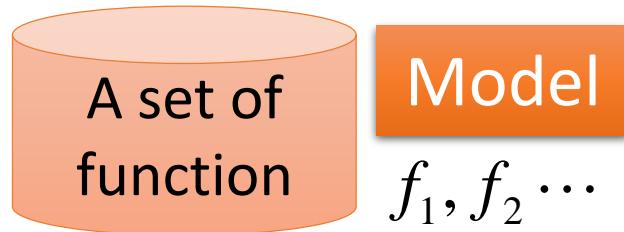
- Dialogue System

$$f\left(\begin{array}{c} \text{“Hi”} \\ \text{(what the user said)} \end{array} \right) = \text{“Hello”} \quad \text{(system response)}$$

Framework

Image Recognition:

$$f(\text{}) = \text{"cat"}$$



$$f_1(\text{}) = \text{"cat"} \quad f_2(\text{}) = \text{"money"}$$

$$f_1(\text{}) = \text{"dog"} \quad f_2(\text{}) = \text{"snake"}$$

Framework

A set of function

Model
 $f_1, f_2 \dots$

Goodness of function f

Training Data

Image Recognition:

$$f\left(\begin{array}{c} \text{Image of a cat} \end{array} \right) = \text{"cat"}$$

$$\begin{array}{ll} f_1\left(\begin{array}{c} \text{Image of a cat} \end{array} \right) = \text{"cat"} & f_2\left(\begin{array}{c} \text{Image of a cat} \end{array} \right) = \text{"money"} \\ f_1\left(\begin{array}{c} \text{Image of a dog} \end{array} \right) = \text{"dog"} & f_2\left(\begin{array}{c} \text{Image of a dog} \end{array} \right) = \text{"snake"} \end{array}$$

Better!

function input:

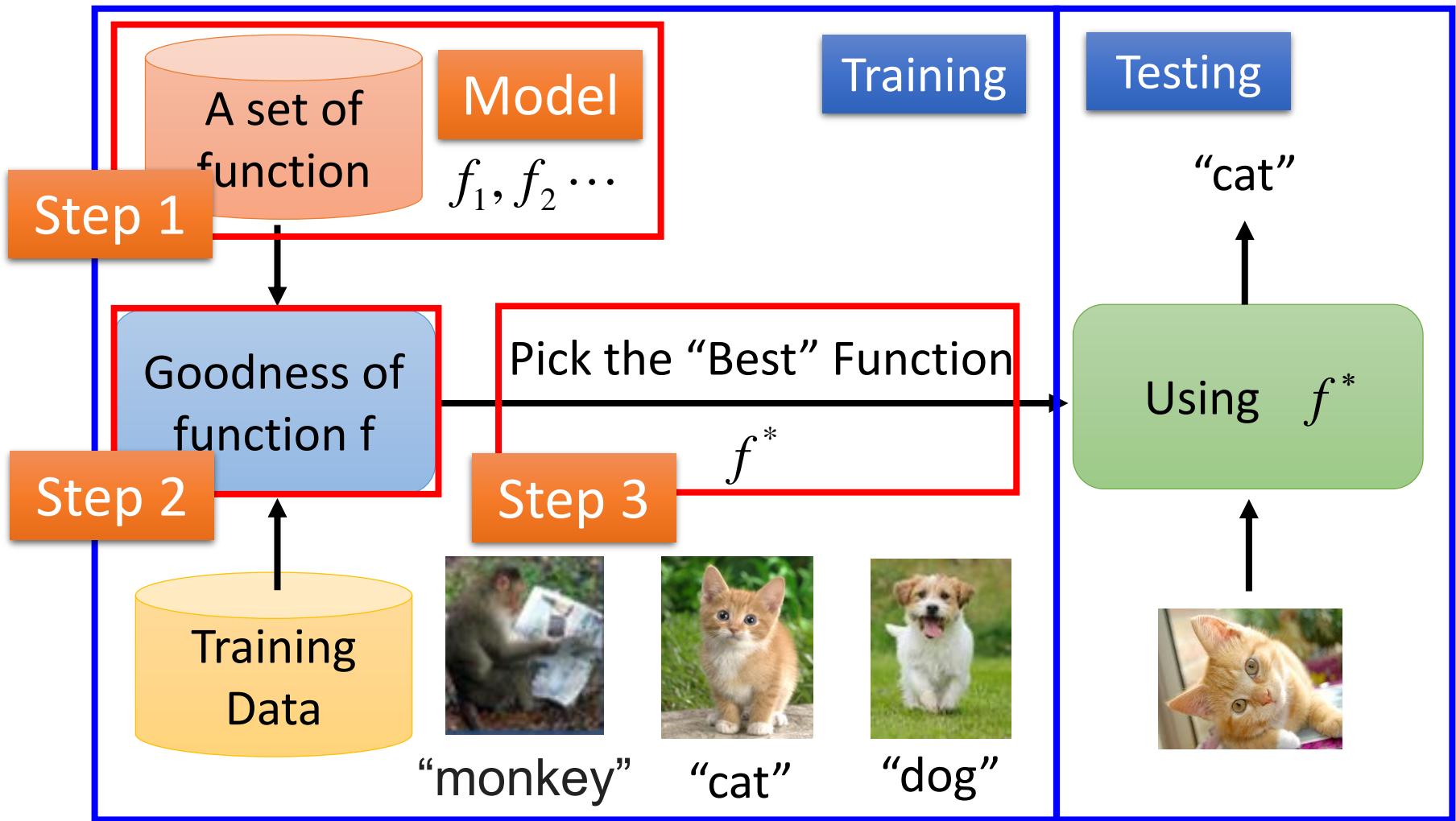


function output: "monkey" "cat" "dog"

Framework

Image Recognition:

$$f(\text{cat image}) = \text{"cat"}$$



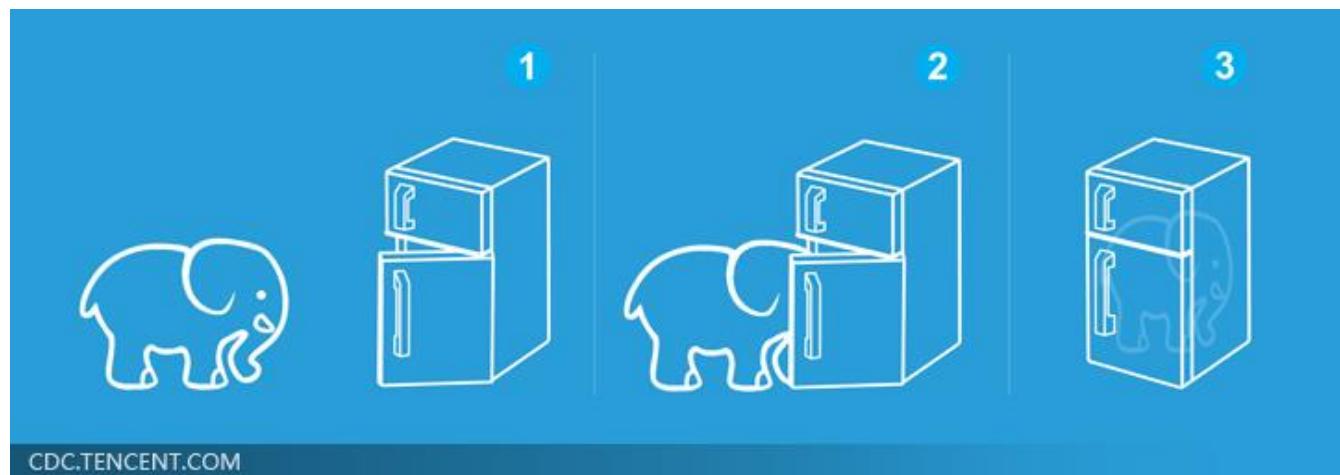
Machine Learning is so simple

Step 1:
define a set
of function

Step 2:
goodness of
function

Step 3: pick
the best
function

就好像把大象放进冰箱



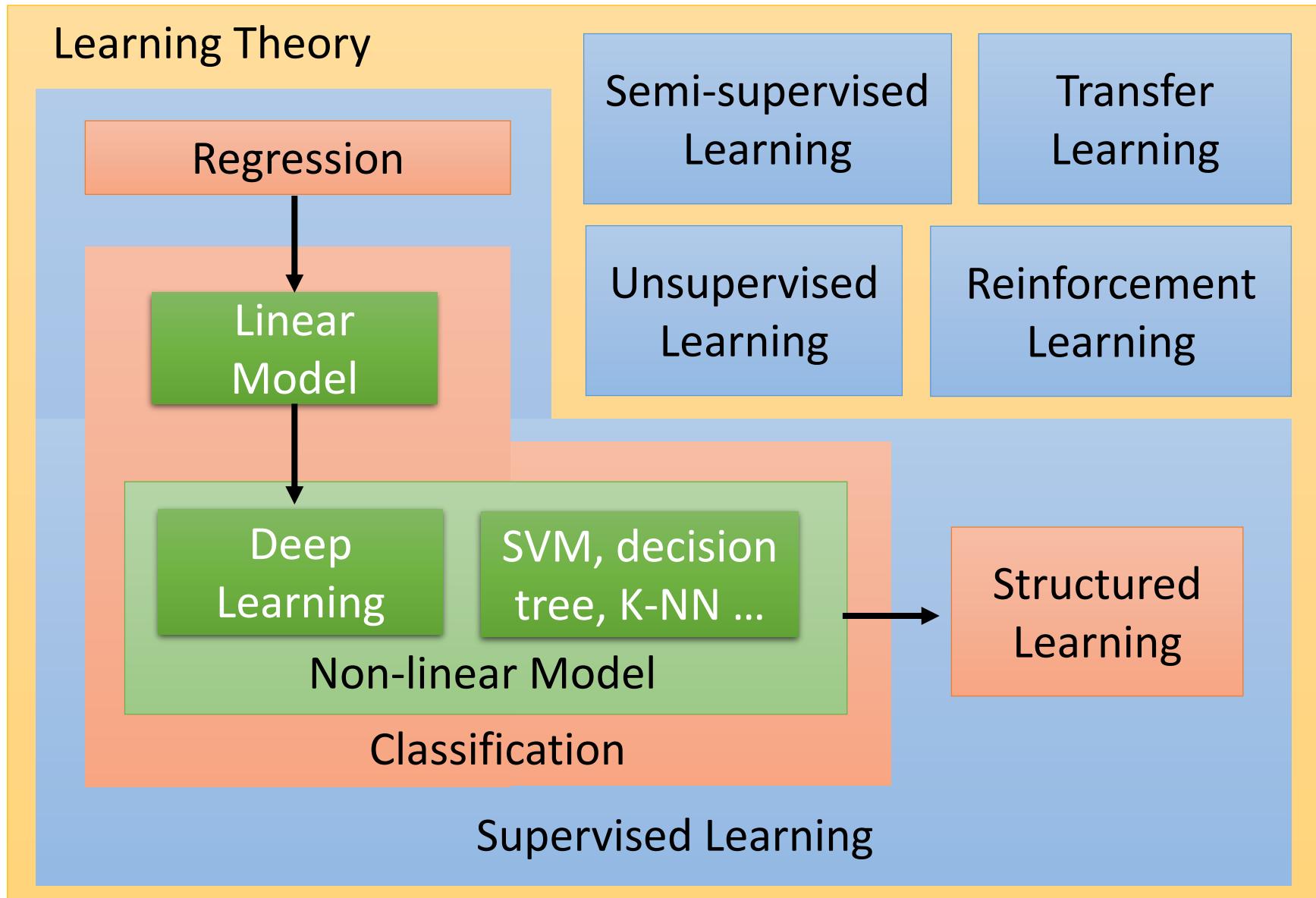
Learning Map

Learning Map

scenario

task

method



Learning Map

Regression

The output of the target function f is “scalar”.

預測
PM2.5



Training Data:

Input:

9/01 上午 PM2.5 = 63 9/02 上午 PM2.5 = 65

Input:

9/12 上午 PM2.5 = 30 9/13 上午 PM2.5 = 25

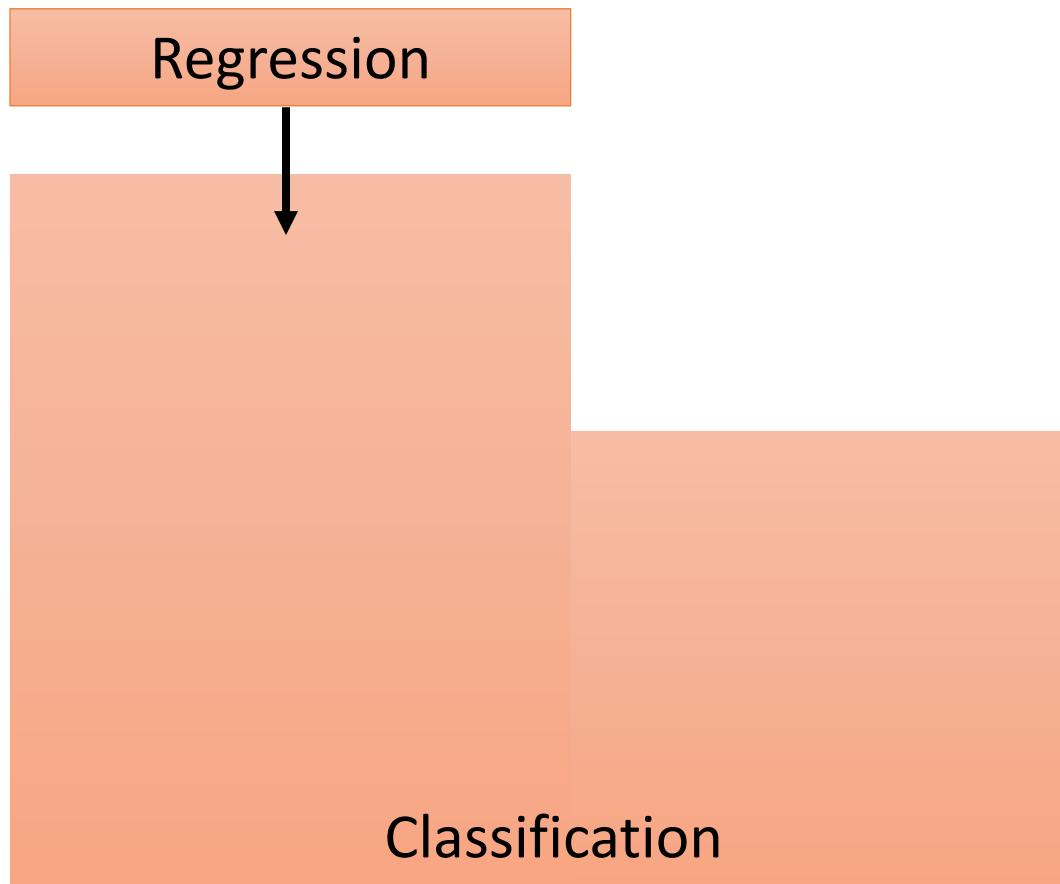
Output:

9/03 上午 PM2.5 = 100

Output:

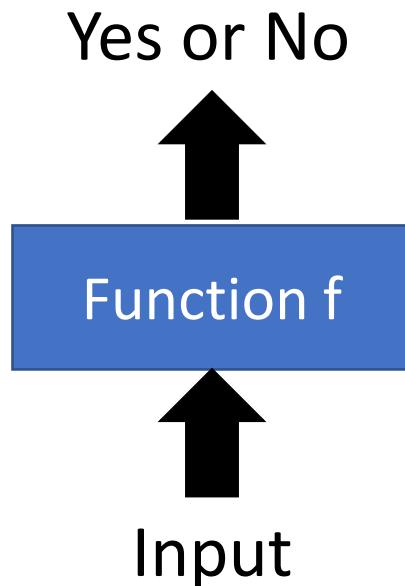
9/14 上午 PM2.5 = 20

Learning Map

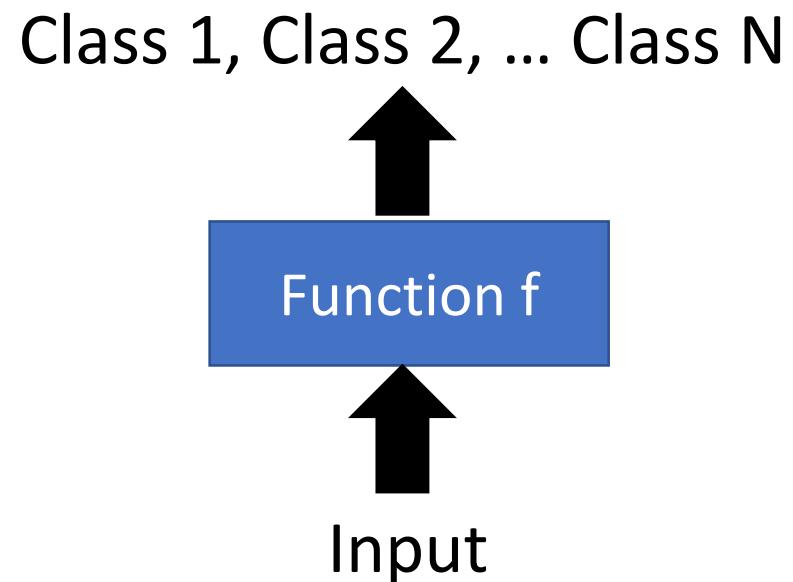


Classification

- Binary Classification

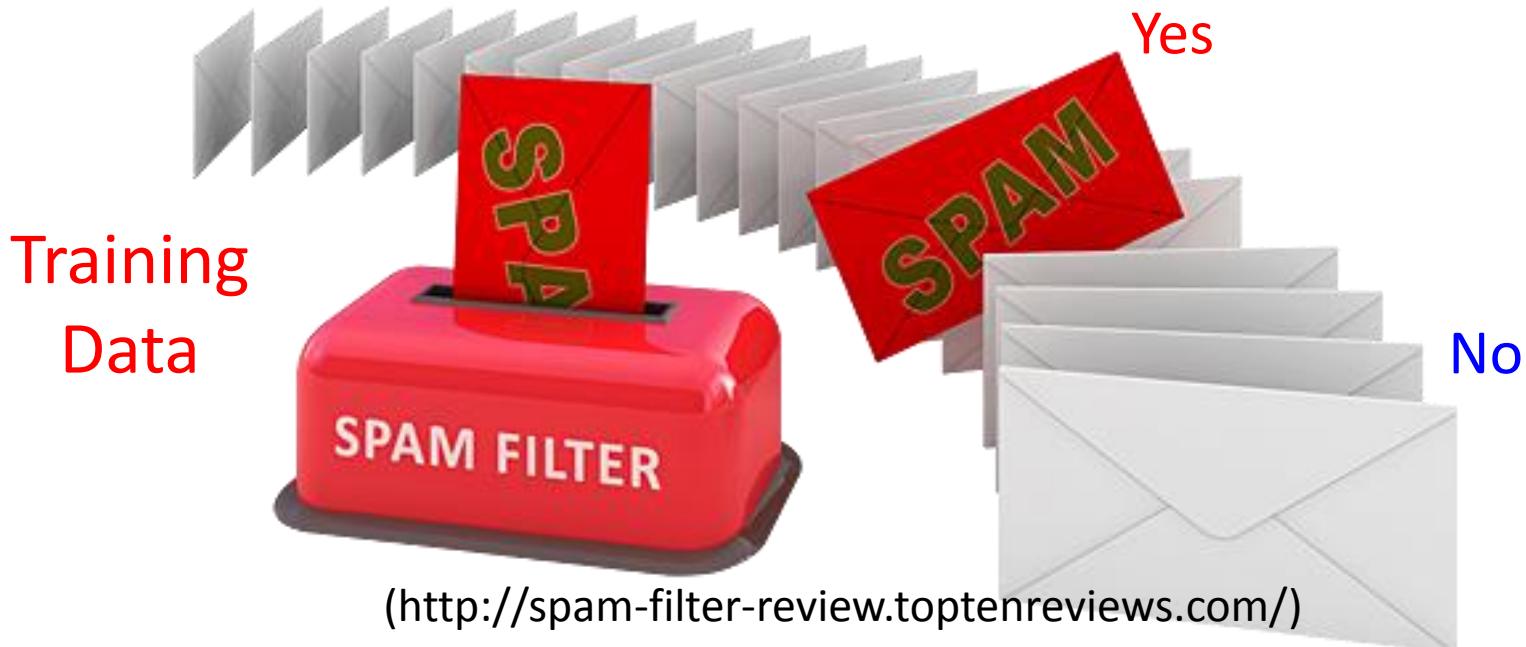
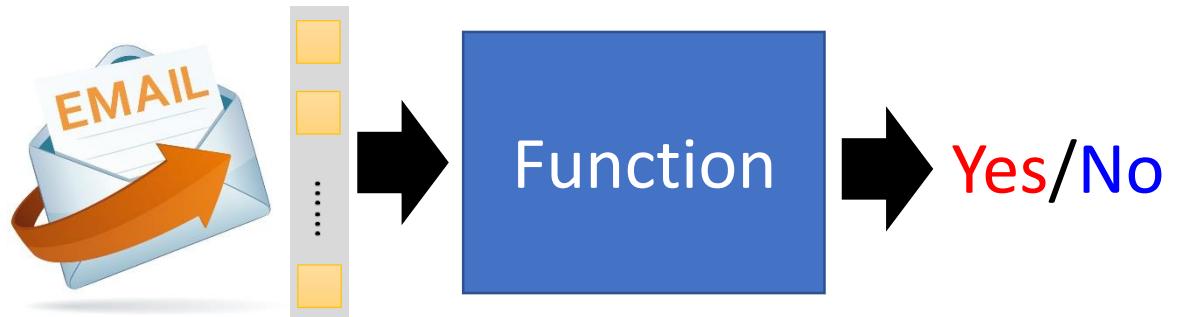


- Multi-class Classification



Binary Classification

**Spam
filtering**

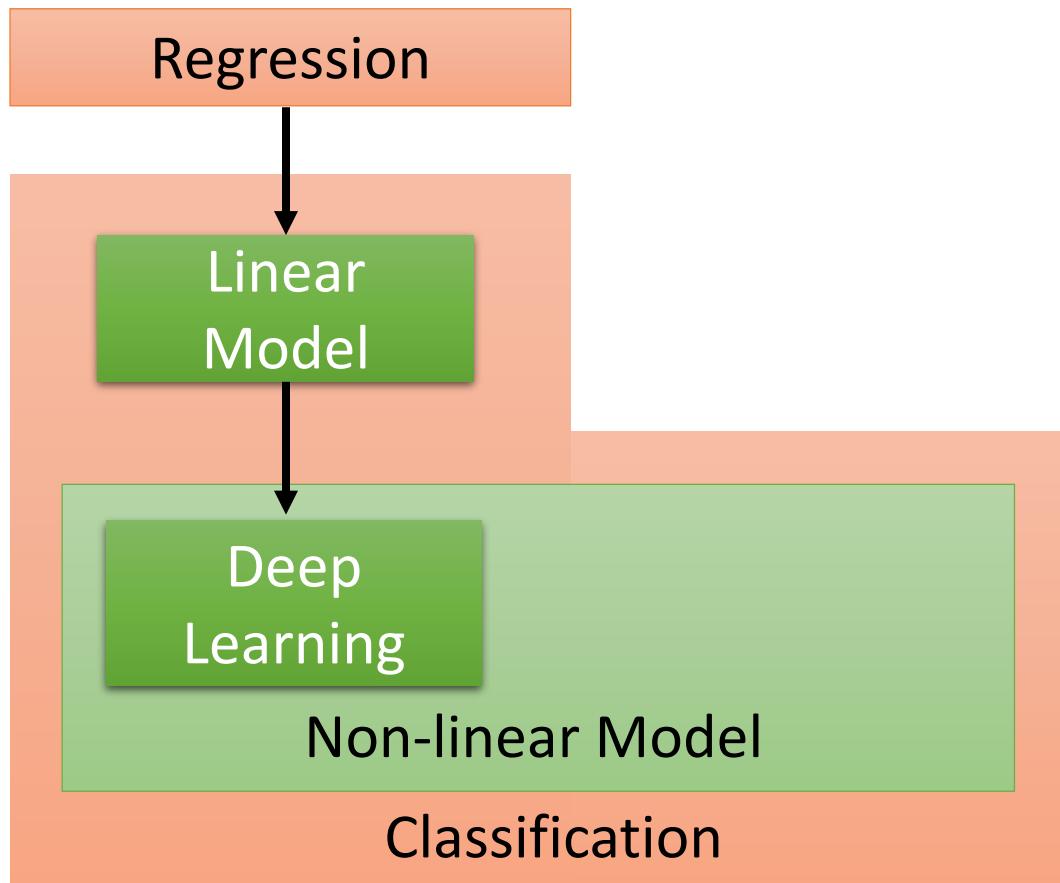


Multi-class Classification

Document *Classification*

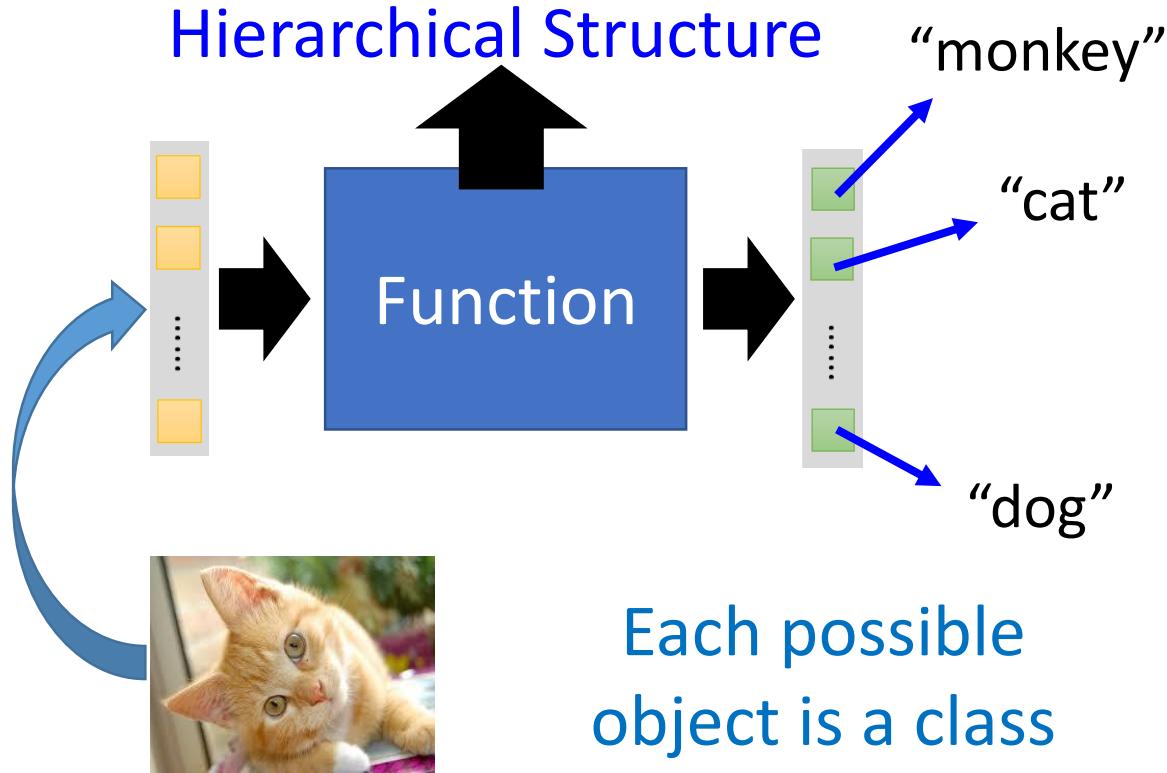


Learning Map

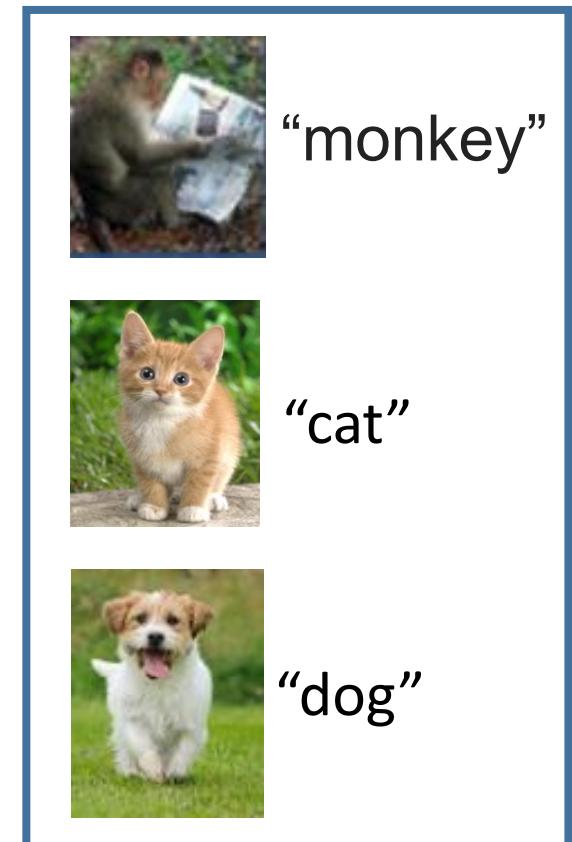


Classification - Deep Learning

- Image Recognition

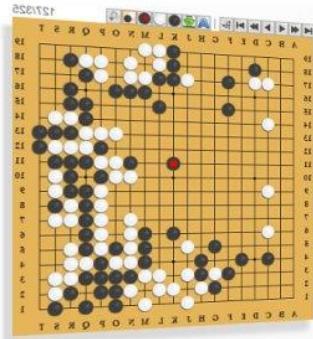


Training Data



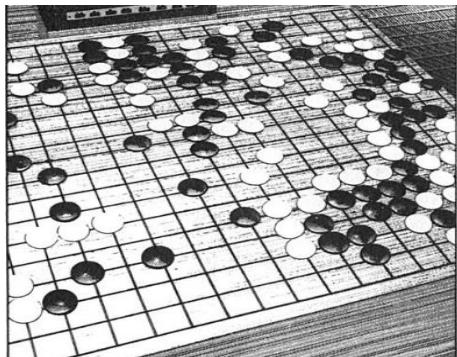
Classification - Deep Learning

- Playing GO



Next move
Each position
is a class
(19×19 classes)

Training Data



一堆棋譜

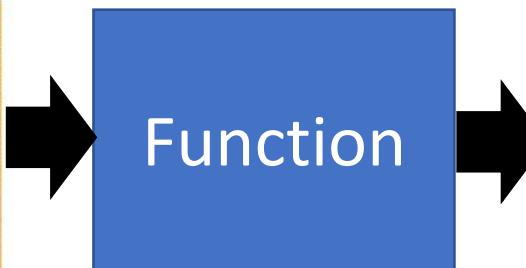
進藤光 v.s. 社清春

黒: 5之五 → 白: 天元 → 黑: 五之5



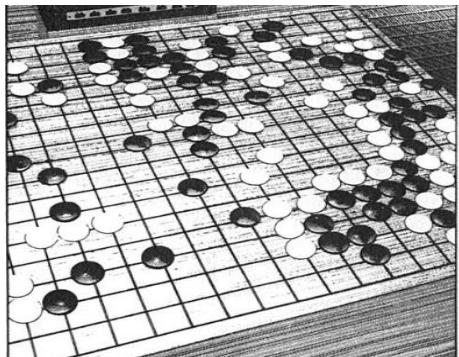
Classification - Deep Learning

- Playing GO



Next move
Each position
is a class
(19×19 classes)

Training Data



一堆棋譜

進藤光 v.s. 社清春

黑: 5之五 → 白: 天元 → 黑: 五之5

Input:

黑: 5之五



Output:

天元

Input:

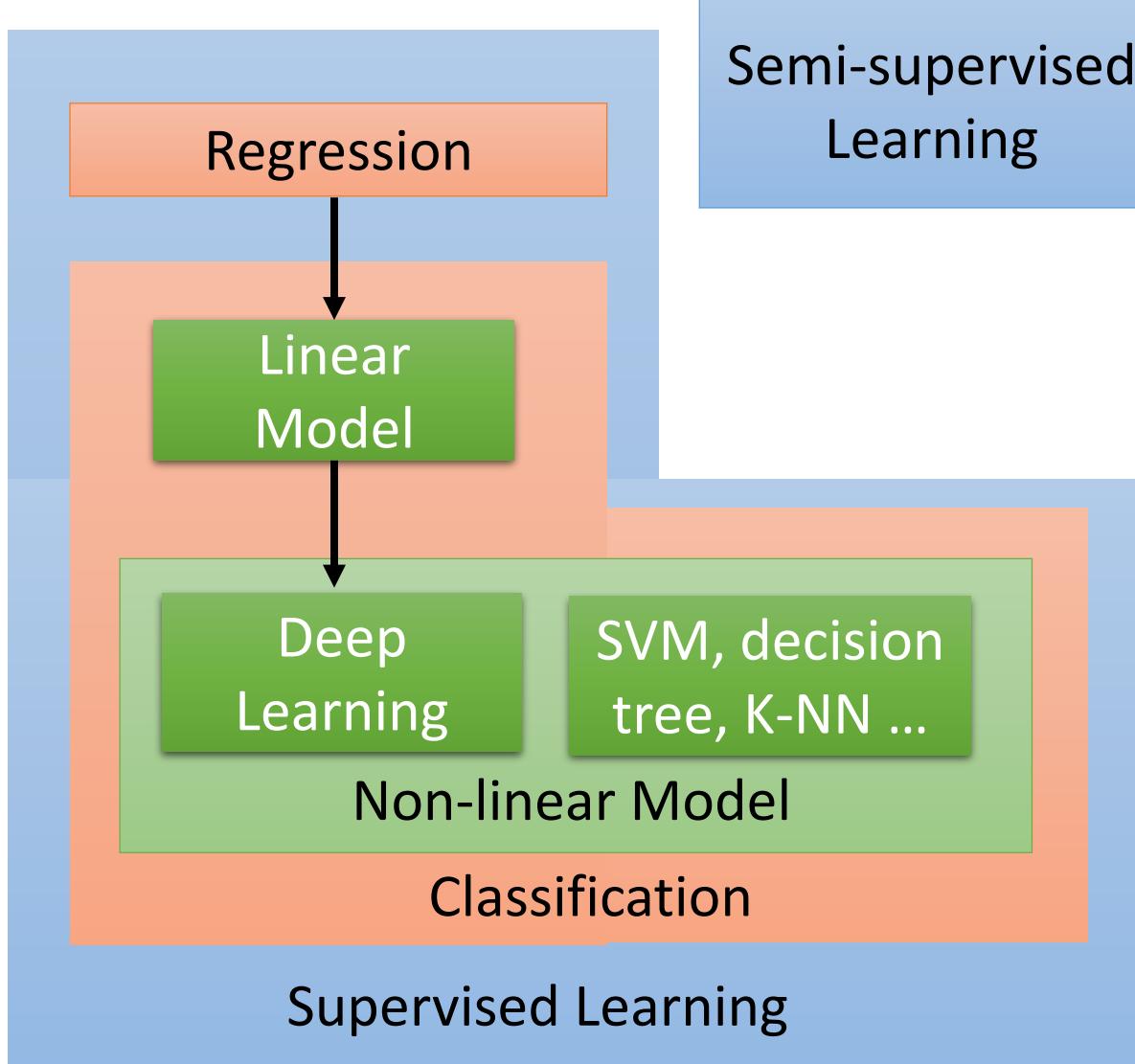
黑: 5之五、白: 天元



Output:

五之5

Learning Map



Hard to collect a large amount of labelled data

Training Data:
Input/output pair of target function
Function output = label

Semi-supervised Learning

For example, recognizing cats and dogs

Labelled
data



cat



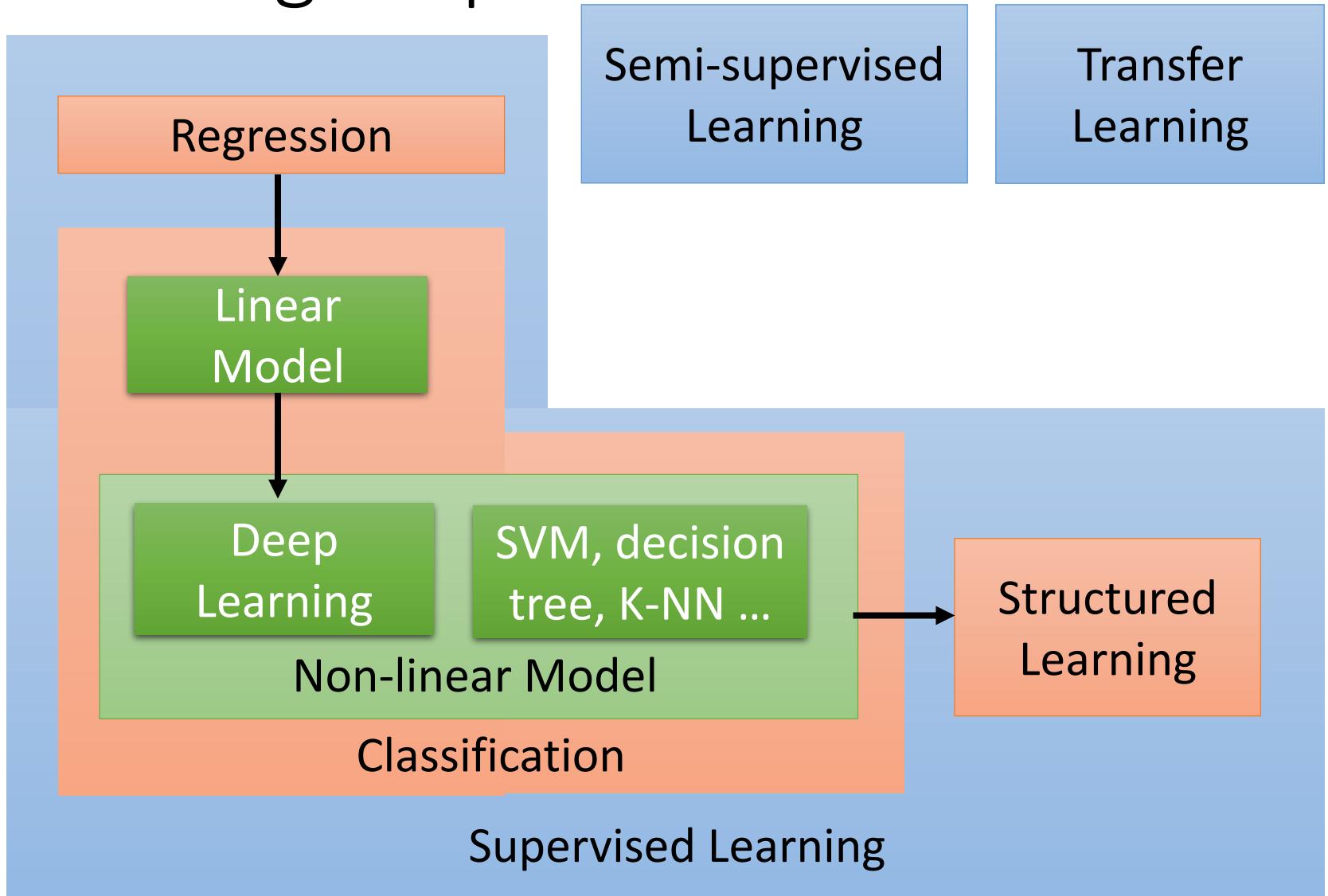
dog

Unlabeled
data



(Images of cats and dogs)

Learning Map



Transfer Learning

For example, recognizing cats and dogs

Labelled
data



cat



dog



elephant

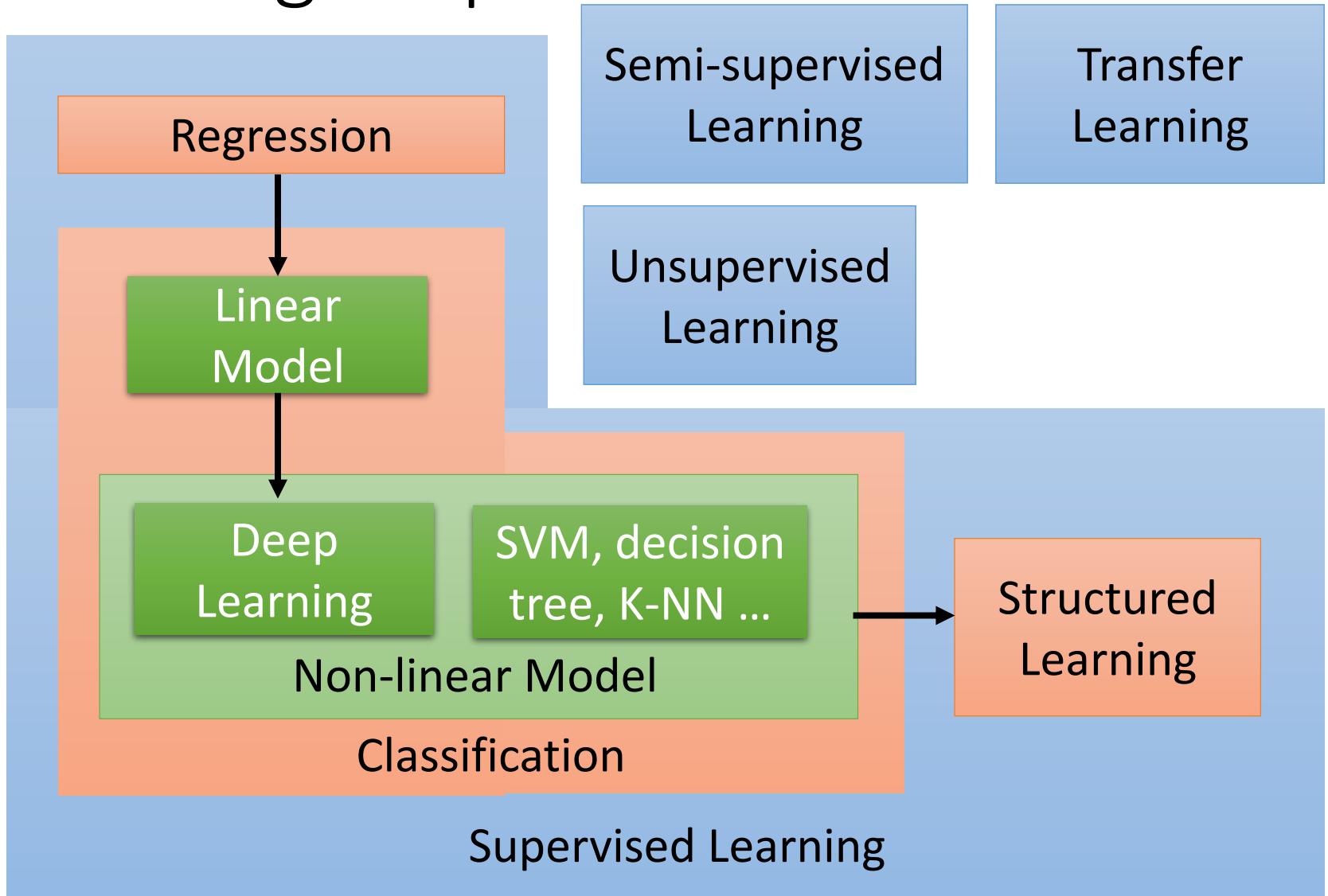


Haruhi



Data not related to the task considered
(can be either labeled or unlabeled)

Learning Map



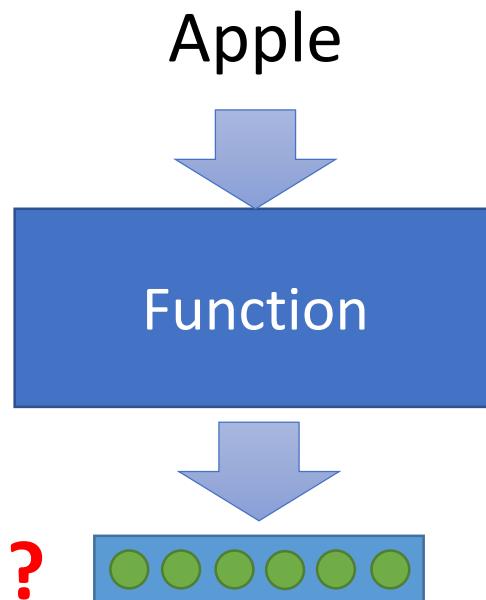
Unsupervised Learning

- Machine Reading: Machine learns the meaning of words from reading a lot of documents



Unsupervised Learning

- Machine Reading: Machine learns the meaning of words from reading a lot of documents



Training data is a lot of text



<https://garavato.files.wordpress.com/2011/11/stacksdocuments.jpg?w=490>

Unsupervised Learning

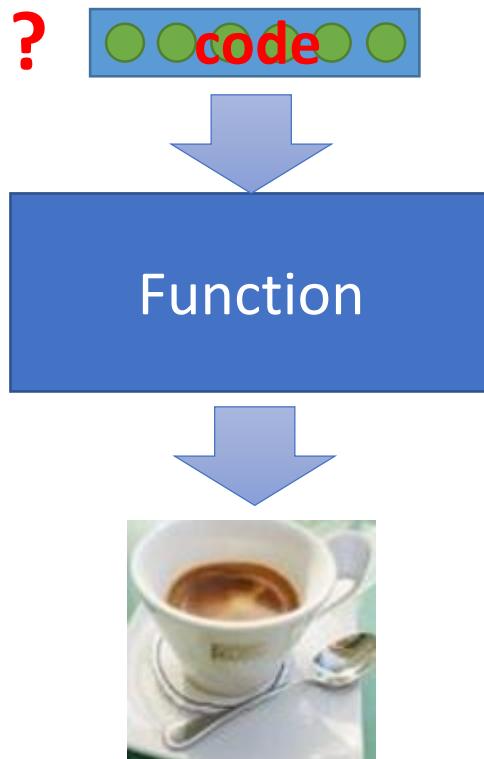


Draw something!

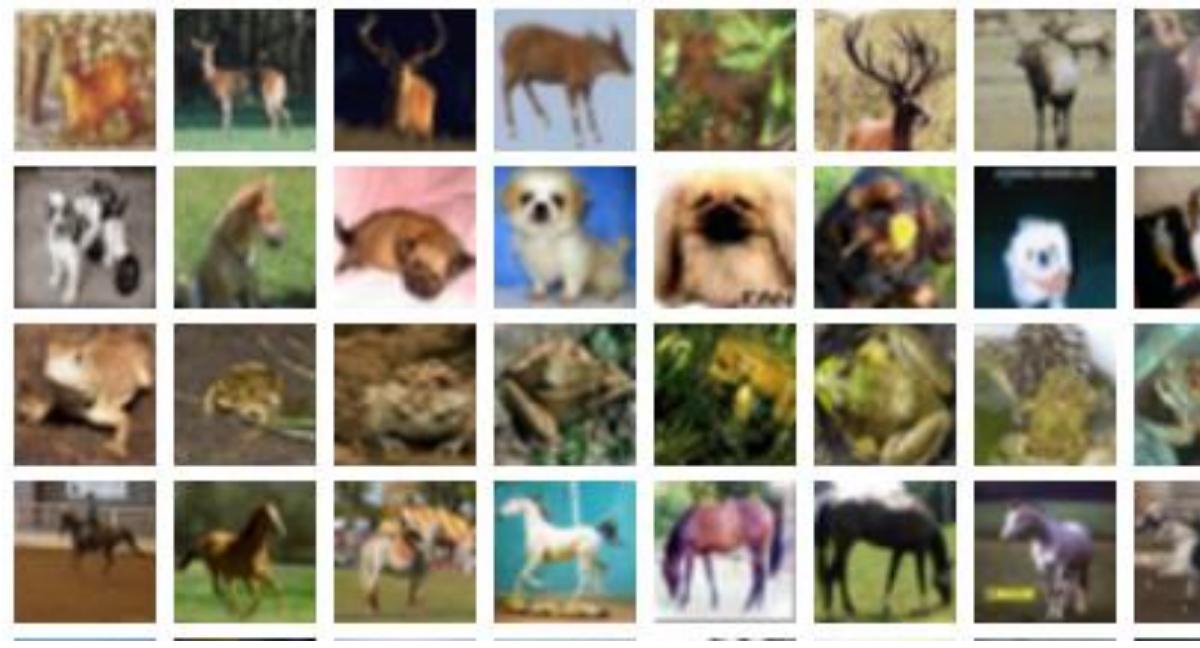


Unsupervised Learning

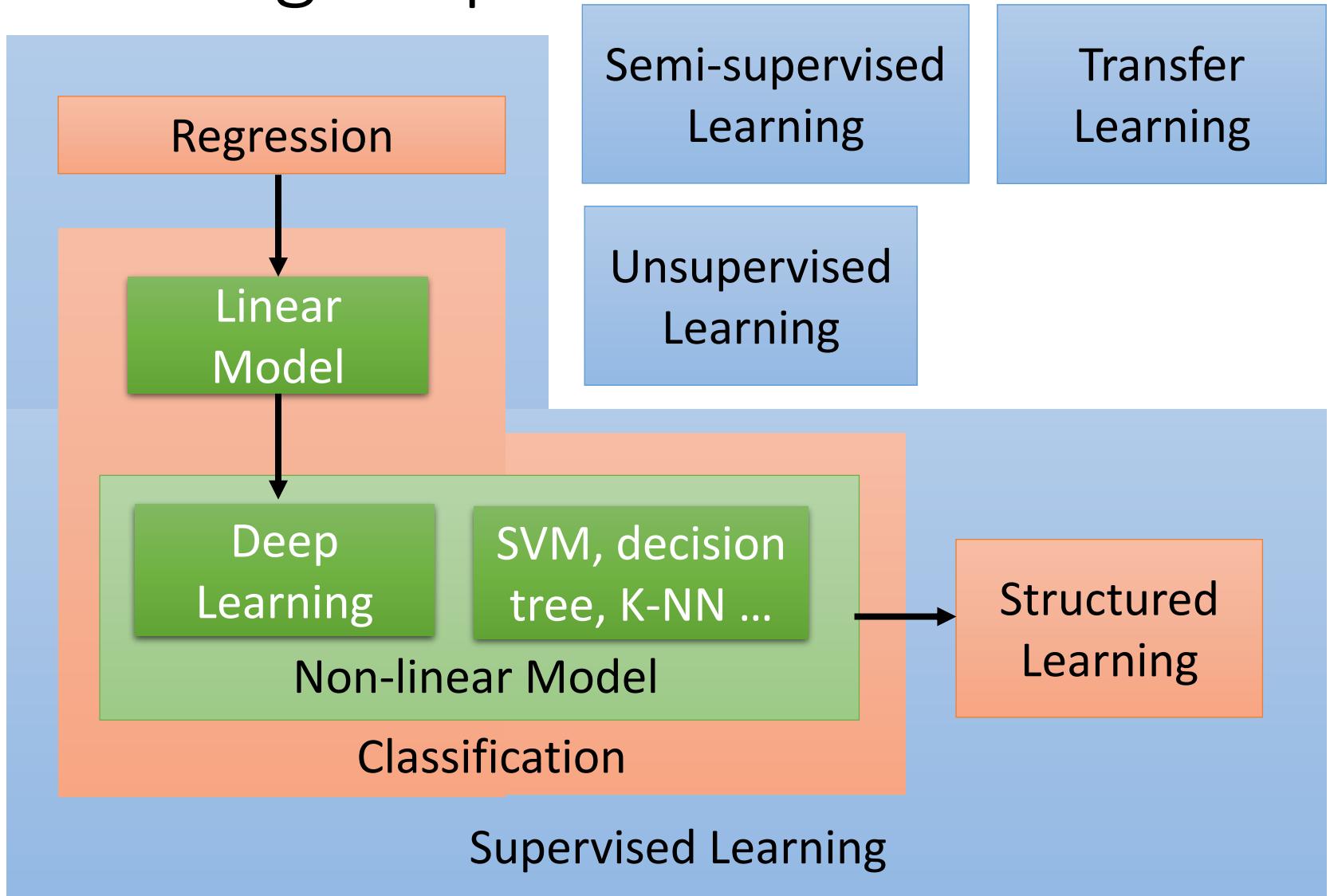
- Machine Drawing



Training data is a lot of images

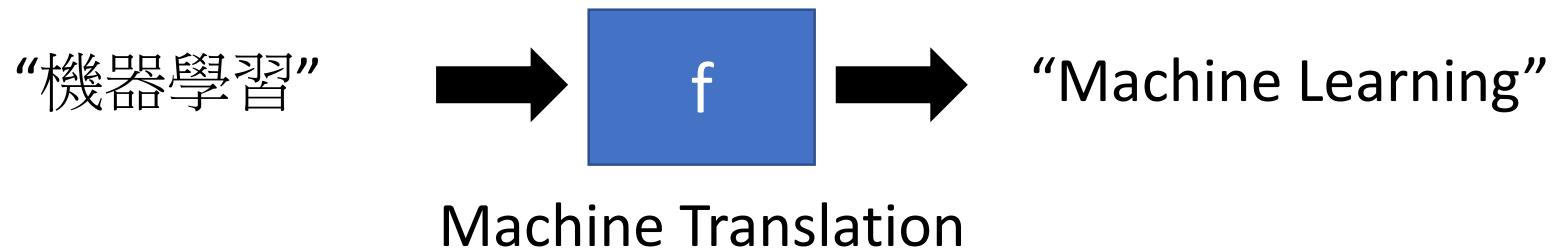


Learning Map

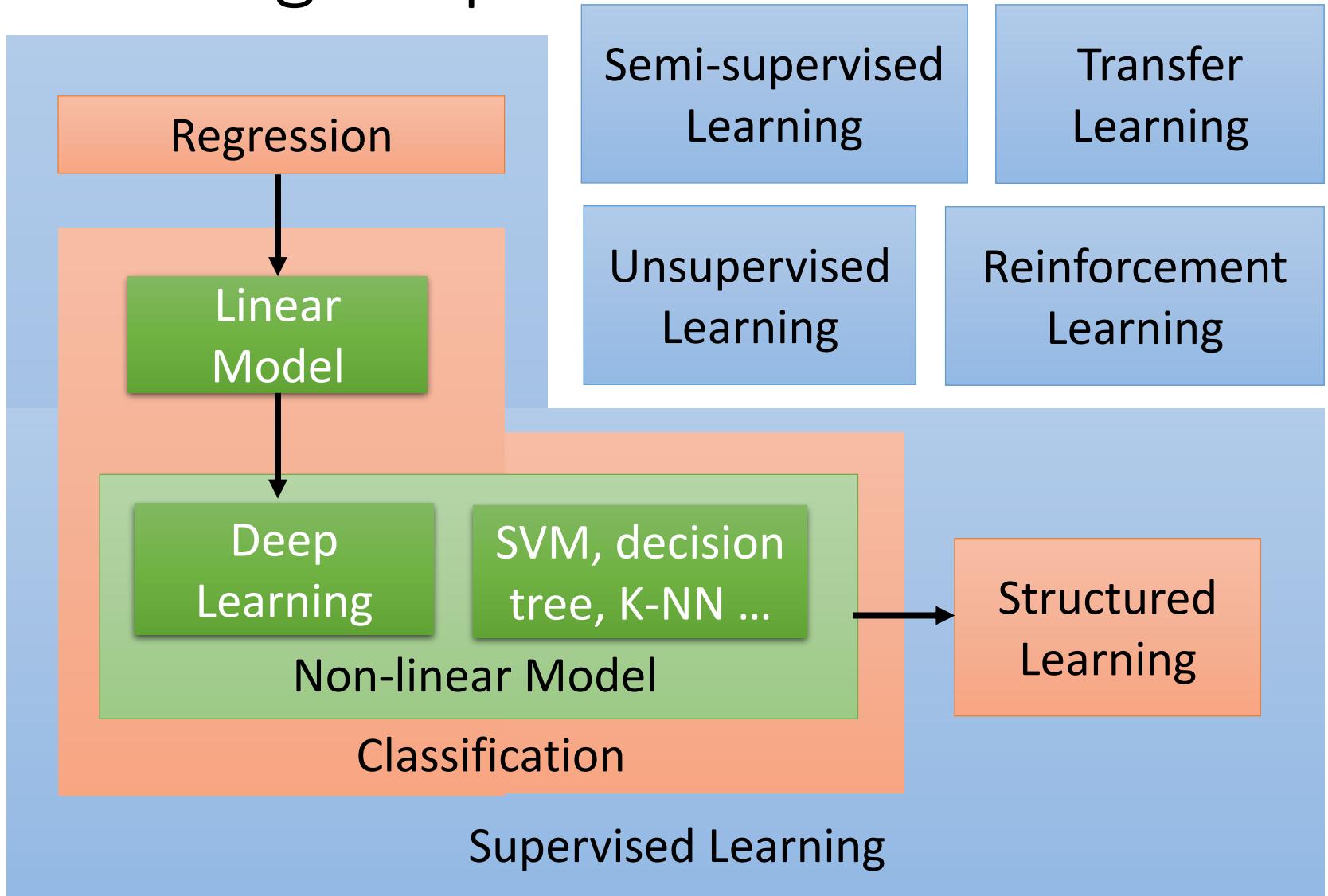


Structured Learning

- Beyond Classification



Learning Map



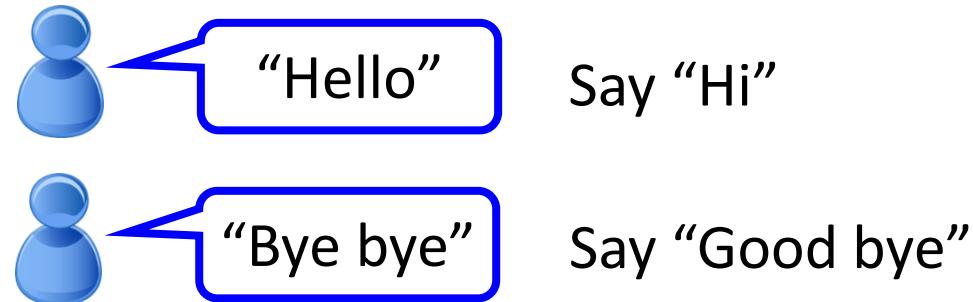
Reinforcement Learning



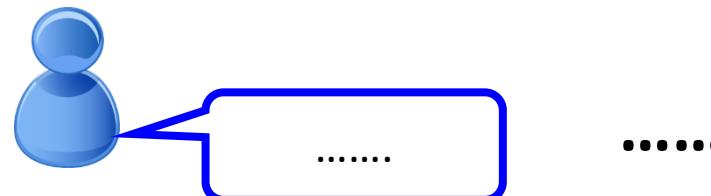
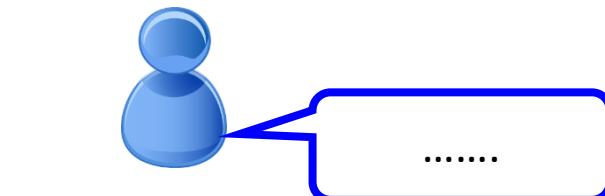
Supervised v.s. Reinforcement

- Supervised

Learning from
teacher



- Reinforcement



.....



Bad

Learning from
critics

Hello ☺

Agent

.....

Agent

Supervised v.s. Reinforcement

- Supervised:



Next move:
“5-5”



Next move:
“3-3”

- Reinforcement Learning

First move → many moves → Win!

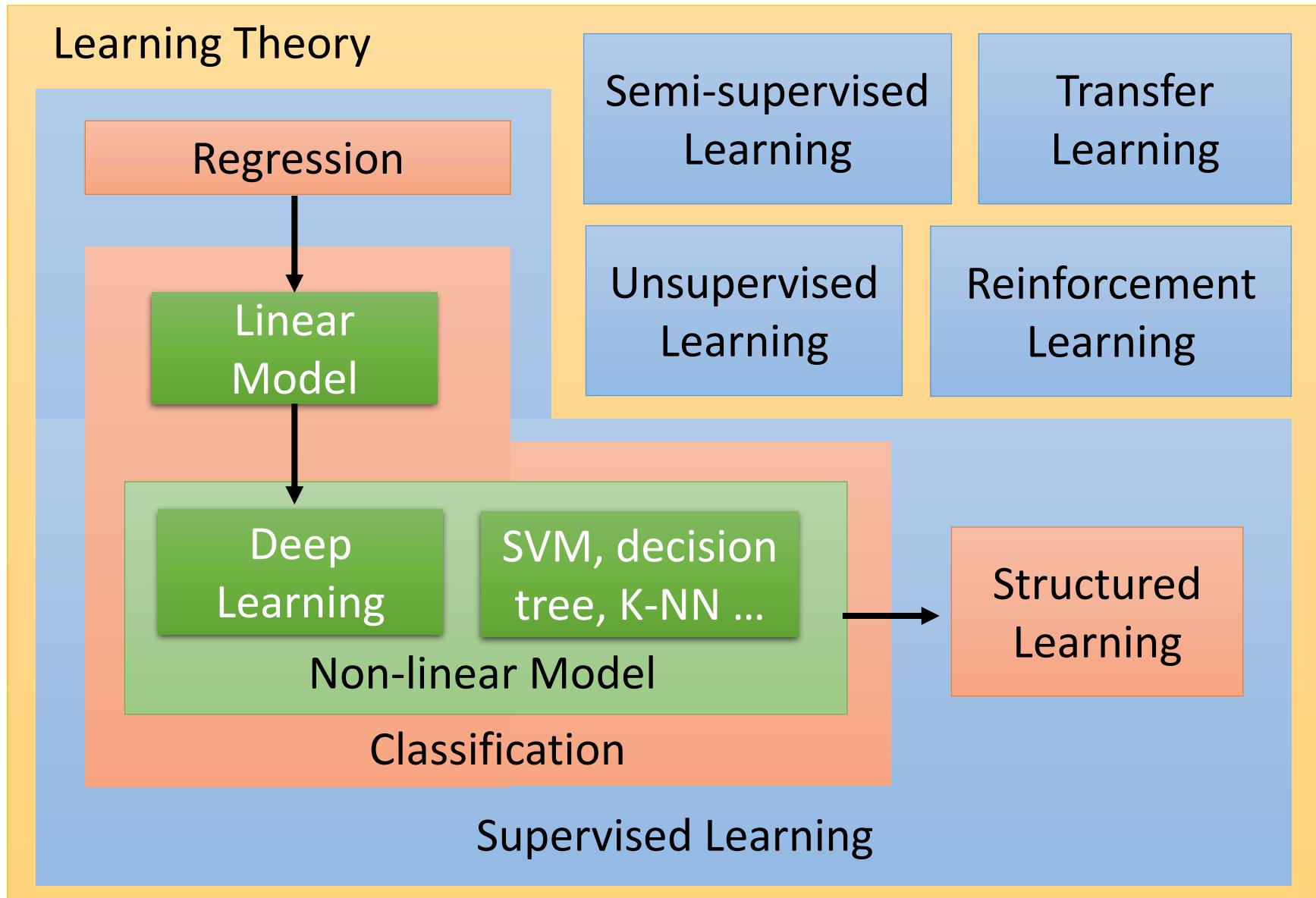
Alpha Go is supervised learning + reinforcement learning.

Learning Map

scenario

task

method





<http://www.express.co.uk/news/science/651202/First-step-towards-The-Terminator-becoming-reality-AI-beats-champ-of-world-s-oldest-game>

Why we need to learn Machine Learning?

AI 即將取代部分的工作? 新工作 : AI 訓練師

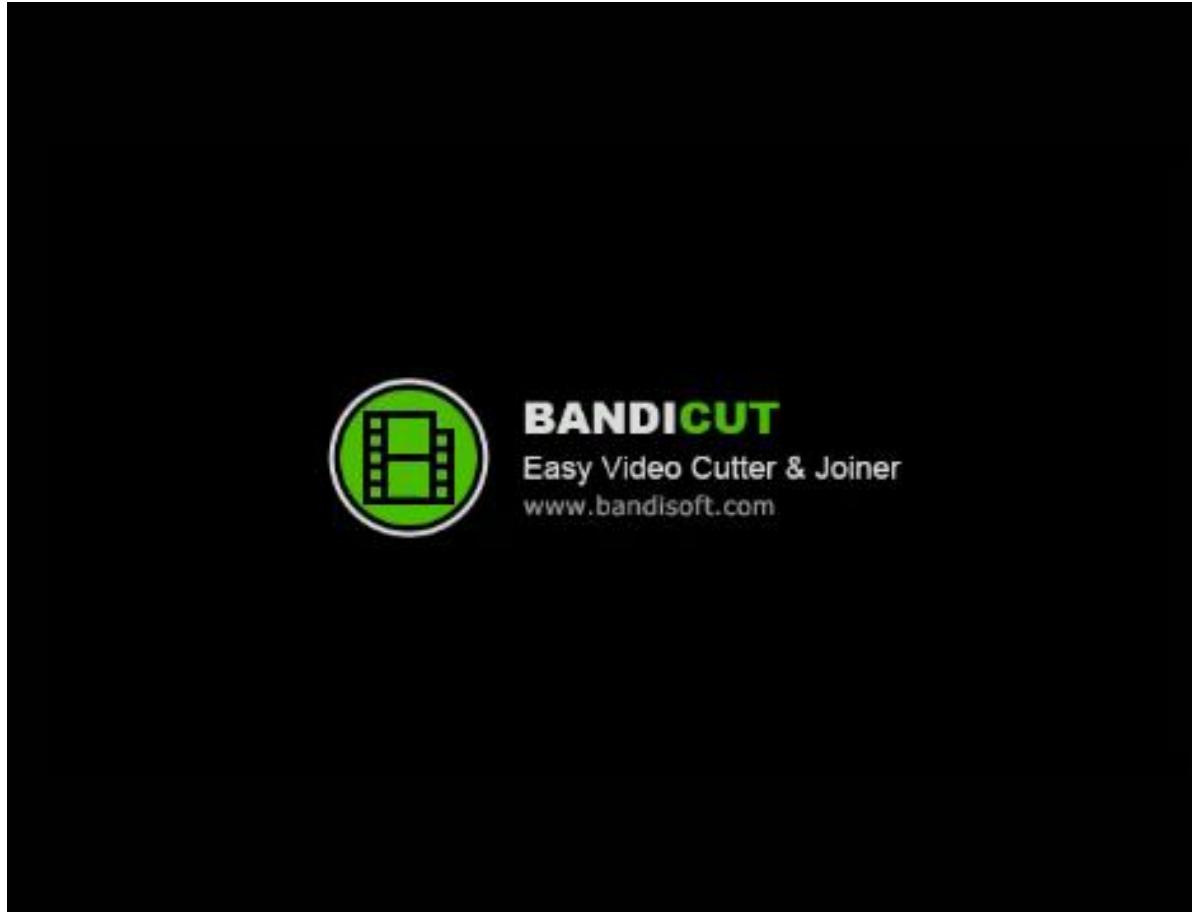
AI 訓練師



機器不是自己會學嗎？
為什麼需要 AI 訓練師

戰鬥是寶可夢在打，
為什麼需要寶可夢訓練師？

神奇寶貝第5集 尼比市的決鬥



https://www.youtube.com/watch?v=uUOZZb8eJ_k

AI 訓練師



寶可夢訓練師

- 寶可夢訓練師要挑選適合的寶可夢來戰鬥
 - 寶可夢有不同的屬性

AI 訓練師

- AI訓練師要挑選合適的 model, loss function
 - 不同 model, loss function 適合解決不同的問題

神奇寶貝第106集 噴火龍·就決定是你了



BANDICUT

Easy Video Cutter & Joiner
www.bandisoft.com

https://www.youtube.com/watch?v=4G_aoKiCDc4

AI 訓練師

Step 1:
define a set
of function

Step 2:
goodness of
function

Step 3: pick
the best
function

寶可夢訓練師

- 寶可夢訓練師要挑選適合的寶可夢來戰鬥
 - 寶可夢有不同的屬性
- 召喚出來的寶可夢不一定聽話
 - E.g. 小智的噴火龍
 - 需要有經驗的寶可夢訓練師

AI 訓練師

- AI訓練師要挑選合適的 model, loss function
 - 不同 model, loss function 適合解決不同的問題
- 不一定能找出 best function
 - E.g. Deep Learning
 - 需要有經驗的 AI 訓練師

大家還記得寶可夢的開場嗎？



BANDICUT

Easy Video Cutter & Joiner
www.bandisoft.com

<https://www.youtube.com/watch?v=NyCNkq4ByzY>

AI 訓練師

- 厲害的 AI ， AI 訓練師功不可沒
- 讓我們一起朝 AI 訓練師之路邁進



Regression

Hung-yi Lee

李宏毅

Regression: Output a scalar

- Stock Market Forecast

 $f($  $) = \text{Dow Jones Industrial Average at tomorrow}$

- Self-driving Car

 $f($  $) = \text{方向盤角度}$

- Recommendation

 $f($

使用者 A

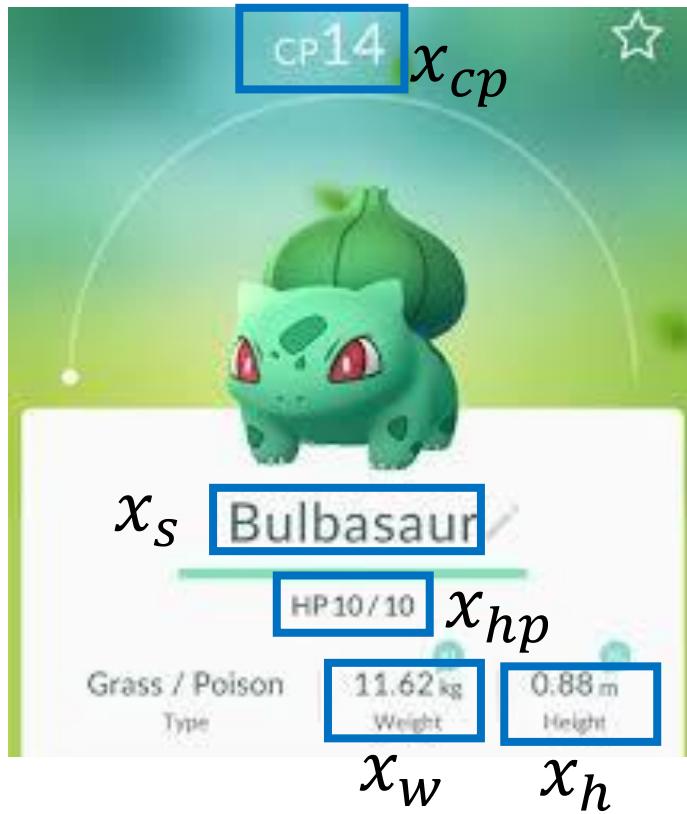
商品 B

 $) =$

購買可能性

Example Application

- Estimating the Combat Power (CP) of a pokemon after evolution

 $f($  $) =$

CP after
evolution

 y

Step 1: Model

$$y = b + w \cdot x_{cp}$$

A set of function

Model

$$f_1, f_2 \dots$$

$f($



Linear model:

$$y = b + \sum w_i x_i$$

w and b are parameters
(can be any value)

$$f_1: y = 10.0 + 9.0 \cdot x_{cp}$$

$$f_2: y = 9.8 + 9.2 \cdot x_{cp}$$

$$f_3: y = -0.8 - 1.2 \cdot x_{cp}$$

..... infinite

$x) =$ CP after evolution y

$x_i: x_{cp}, x_{hp}, x_w, x_h \dots$

feature

$w_i: \text{weight}, b: \text{bias}$

Step 2: Goodness of Function

$$y = b + w \cdot x_{cp}$$

A set of function

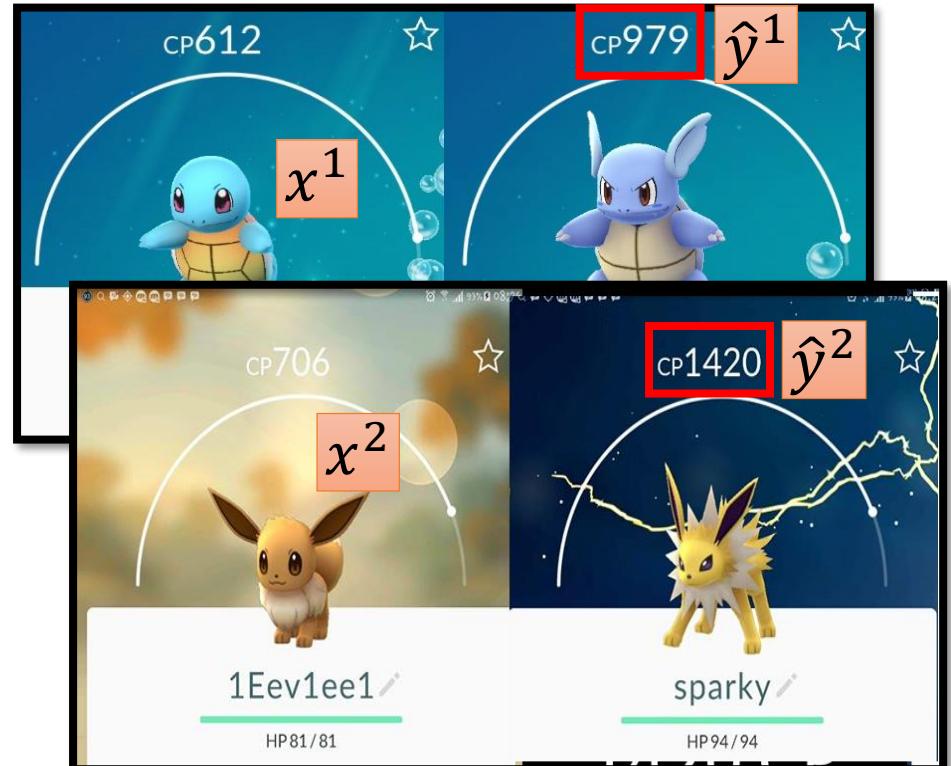
Model

$f_1, f_2 \dots$

Training Data

function input:

function Output (scalar):



Step 2: Goodness of Function

Training Data:
10 pokemons

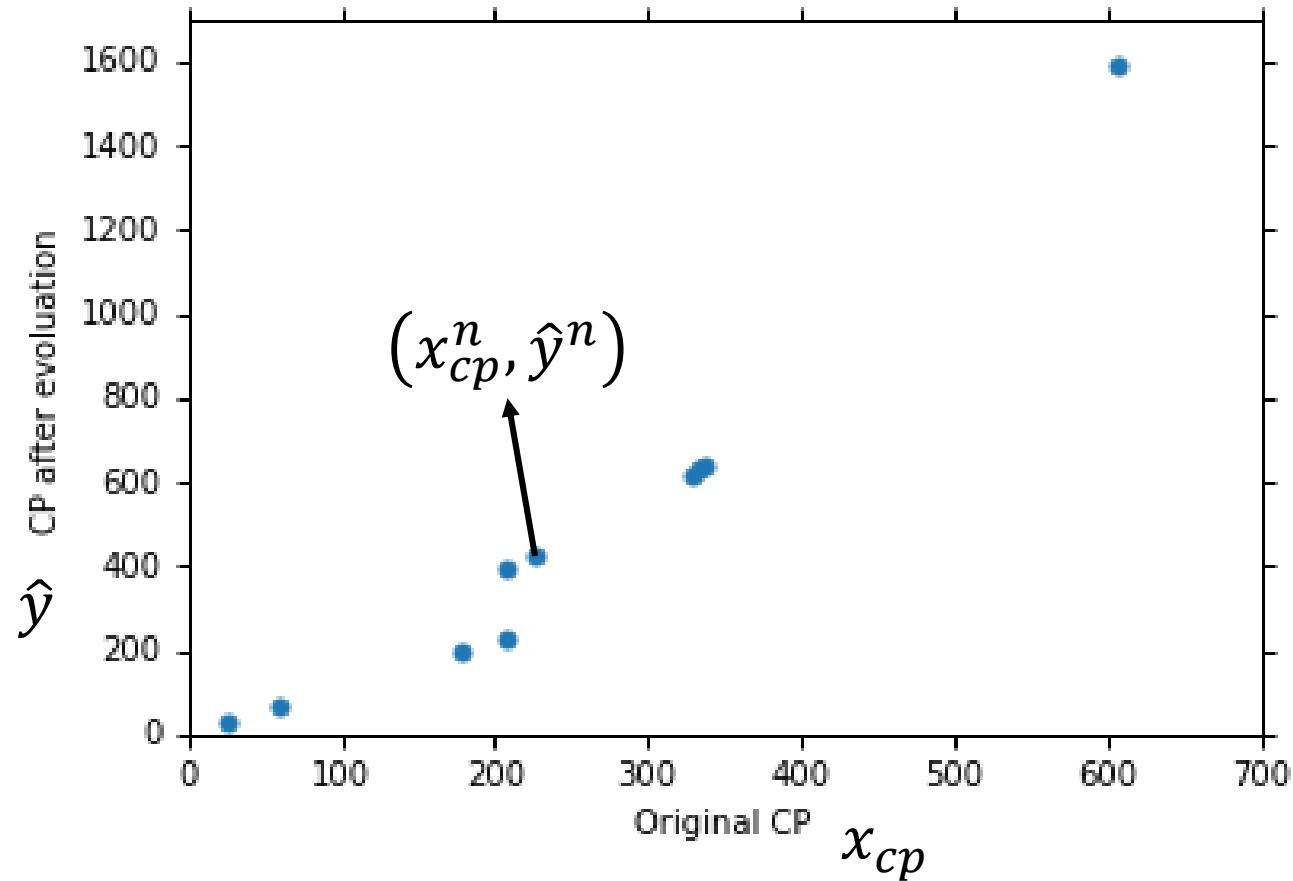
$$(x^1, \hat{y}^1)$$

$$(x^2, \hat{y}^2)$$

⋮

$$(x^{10}, \hat{y}^{10})$$

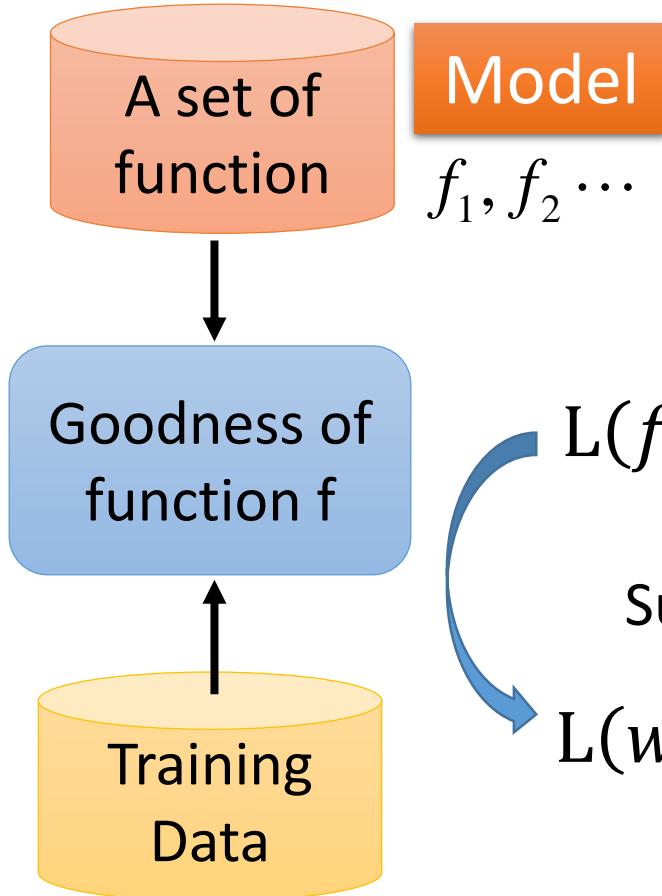
This is real data.



Source: <https://www.openintro.org/stat/data/?data=pokemon>

Step 2: Goodness of Function

$$y = b + w \cdot x_{cp}$$



Loss function L :

Input: a function, output:
how bad it is

$$L(f) = \sum_{n=1}^{10} \left(\hat{y}^n - f(x_{cp}^n) \right)^2$$

Sum over examples

Estimation error
Estimated y based on input function

$$L(w, b) = \sum_{n=1}^{10} \left(\hat{y}^n - (b + w \cdot x_{cp}^n) \right)^2$$

Step 2: Goodness of Function

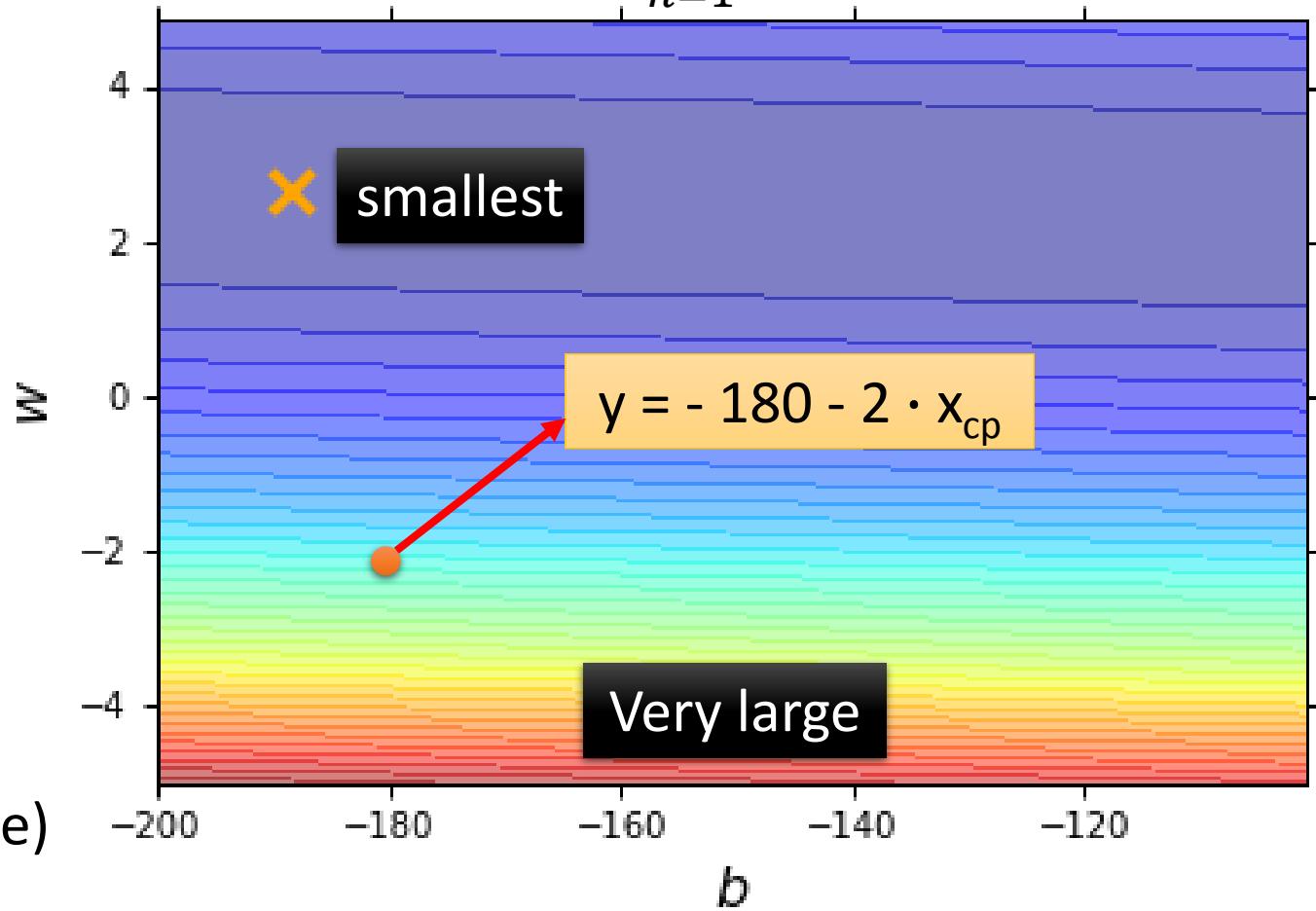
- Loss Function

Each point in the figure is a function

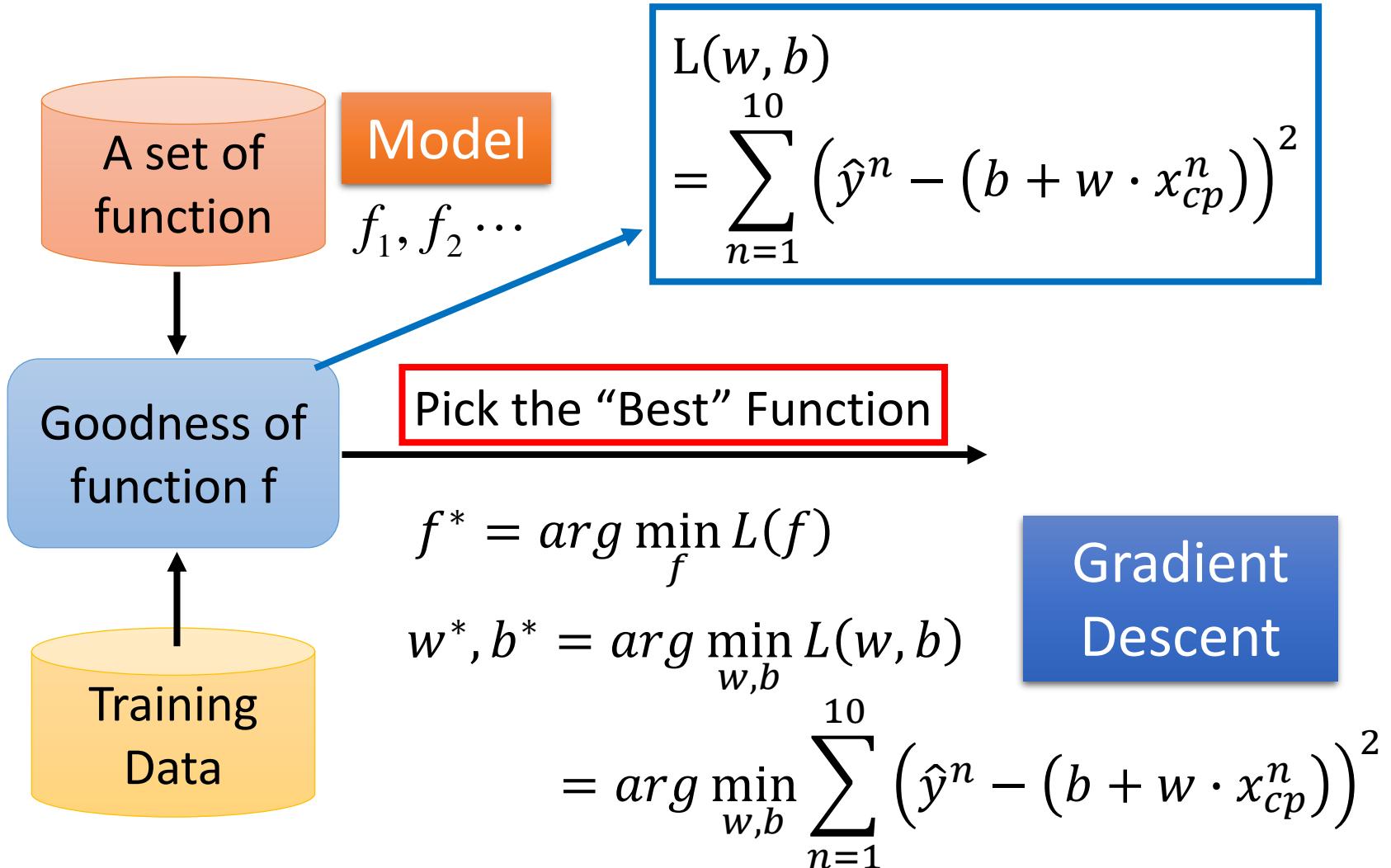
The color represents $L(w, b)$.

(true example)

$$L(w, b) = \sum_{n=1}^{10} (\hat{y}^n - (b + w \cdot x_{cp}^n))^2$$



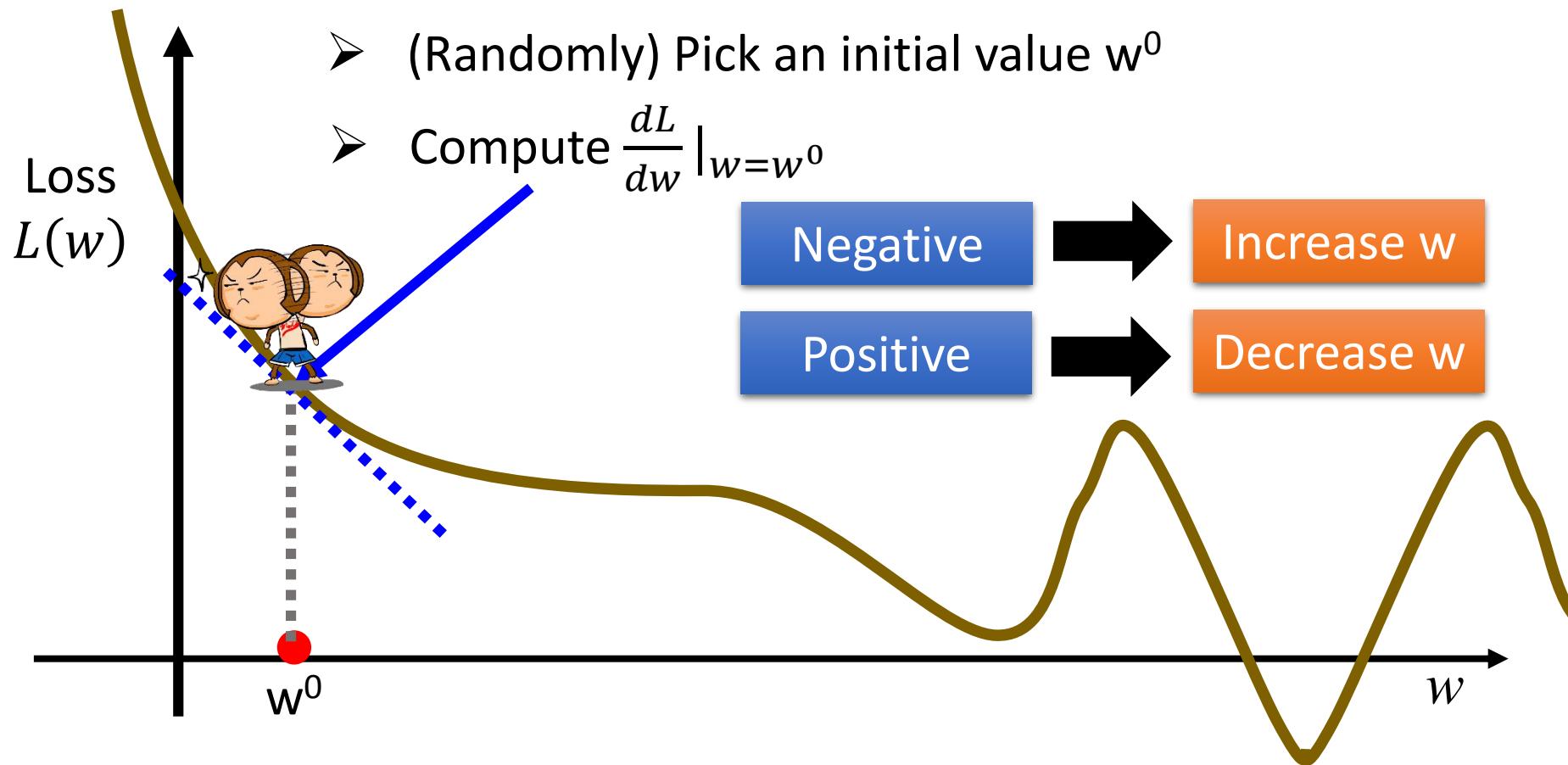
Step 3: Best Function



Step 3: Gradient Descent

$$w^* = \arg \min_w L(w)$$

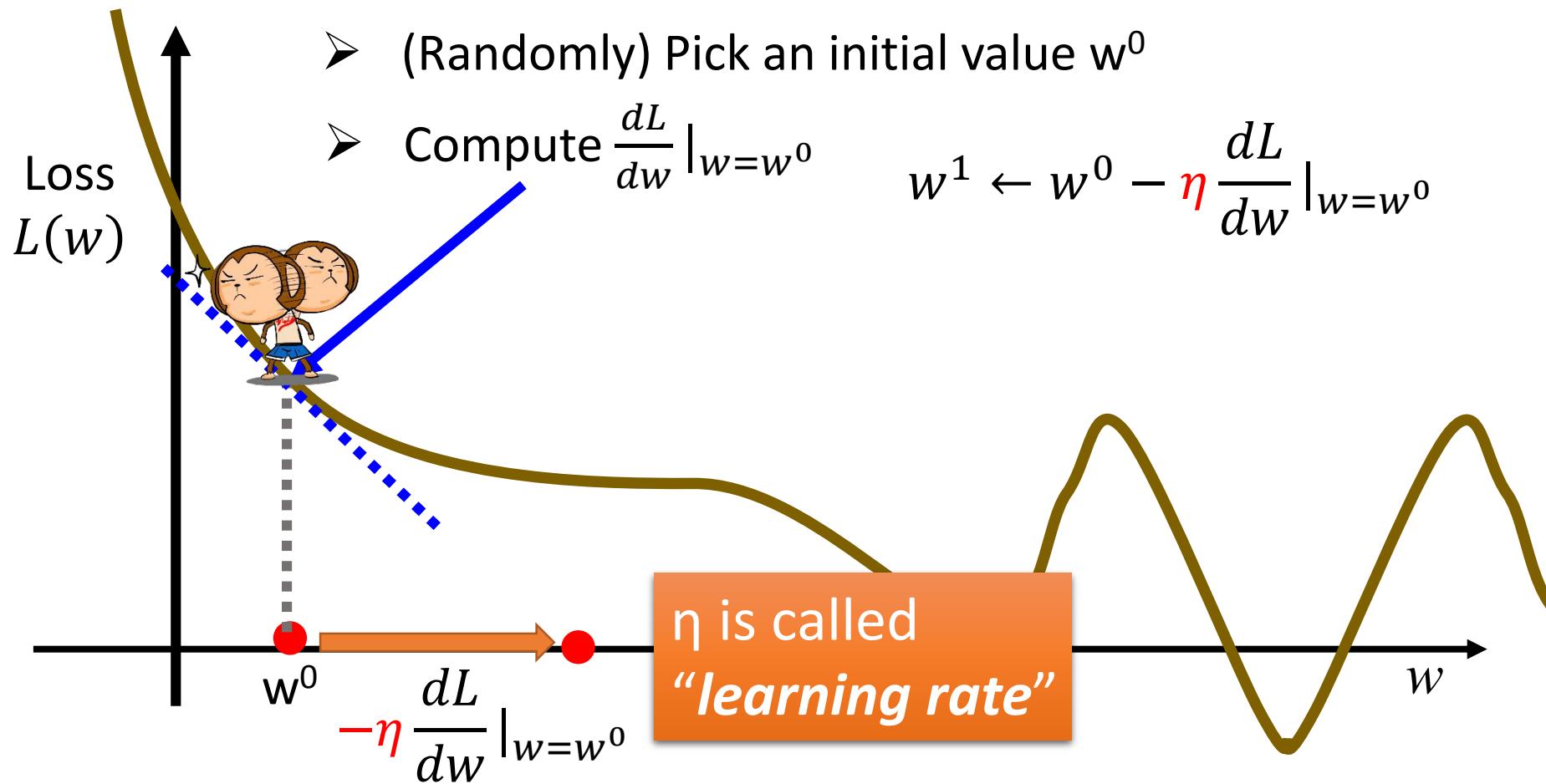
- Consider loss function $L(w)$ with one parameter w :



Step 3: Gradient Descent

$$w^* = \arg \min_w L(w)$$

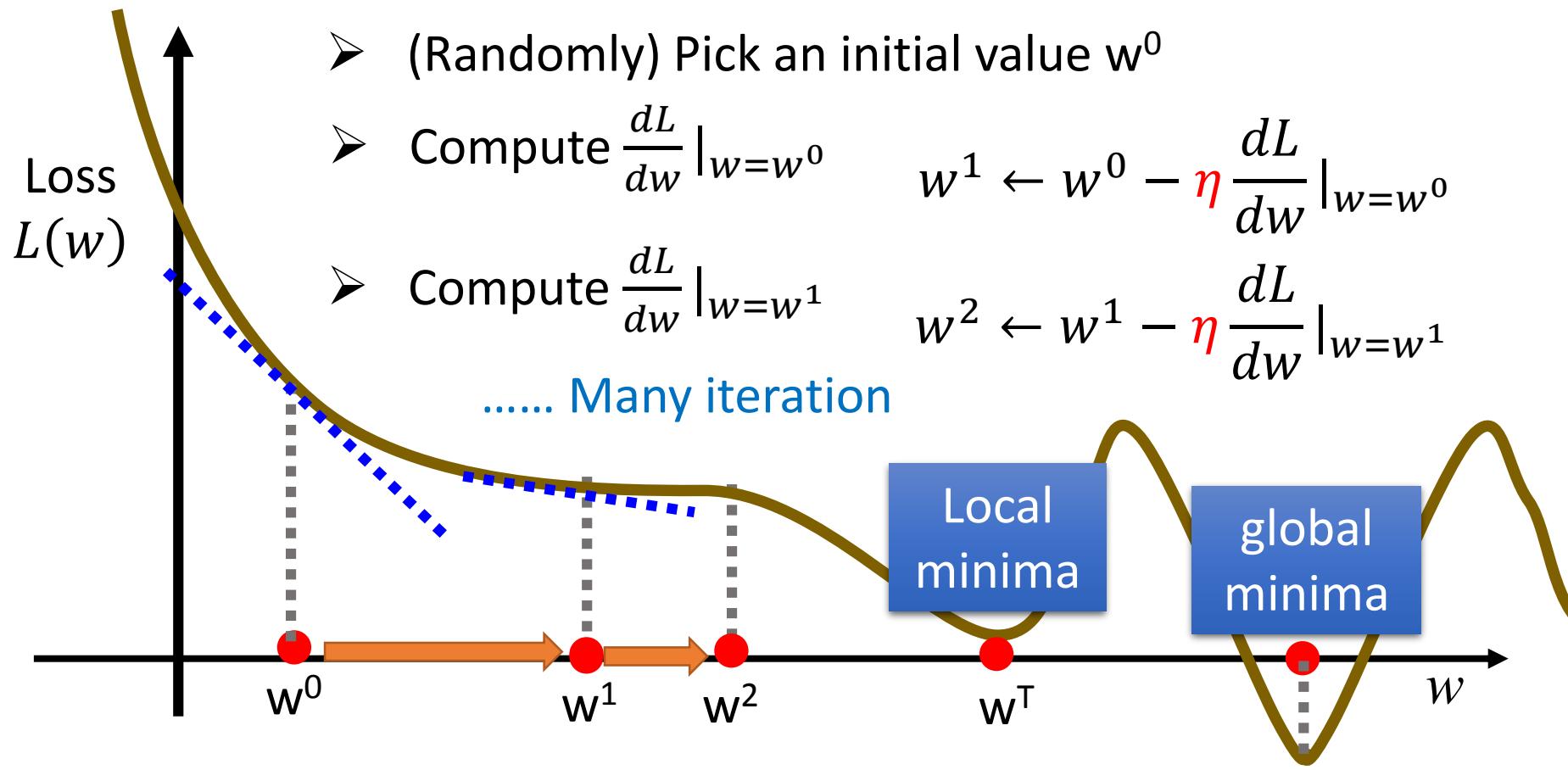
- Consider loss function $L(w)$ with one parameter w :



Step 3: Gradient Descent

$$w^* = \arg \min_w L(w)$$

- Consider loss function $L(w)$ with one parameter w :



$$\begin{bmatrix} \frac{\partial L}{\partial w} \\ \frac{\partial L}{\partial b} \end{bmatrix}$$

Step 3: Gradient Descent

gradient

- How about two parameters? $w^*, b^* = \arg \min_{w,b} L(w, b)$

➤ (Randomly) Pick an initial value w^0, b^0

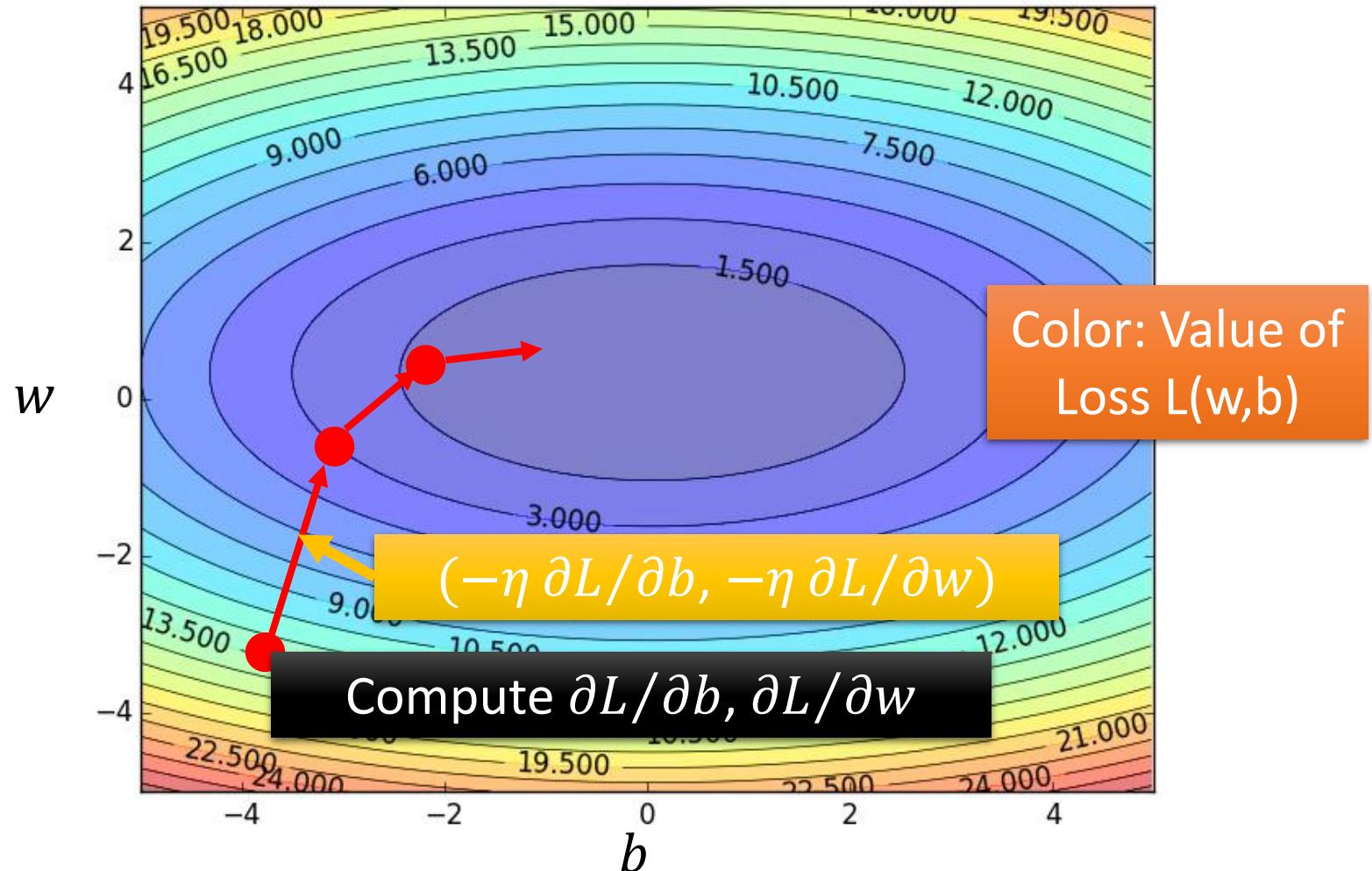
➤ Compute $\frac{\partial L}{\partial w} |_{w=w^0, b=b^0}, \frac{\partial L}{\partial b} |_{w=w^0, b=b^0}$

$$w^1 \leftarrow w^0 - \eta \frac{\partial L}{\partial w} |_{w=w^0, b=b^0} \quad b^1 \leftarrow b^0 - \eta \frac{\partial L}{\partial b} |_{w=w^0, b=b^0}$$

➤ Compute $\frac{\partial L}{\partial w} |_{w=w^1, b=b^1}, \frac{\partial L}{\partial b} |_{w=w^1, b=b^1}$

$$w^2 \leftarrow w^1 - \eta \frac{\partial L}{\partial w} |_{w=w^1, b=b^1} \quad b^2 \leftarrow b^1 - \eta \frac{\partial L}{\partial b} |_{w=w^1, b=b^1}$$

Step 3: Gradient Descent



Step 3: Gradient Descent

- When solving:

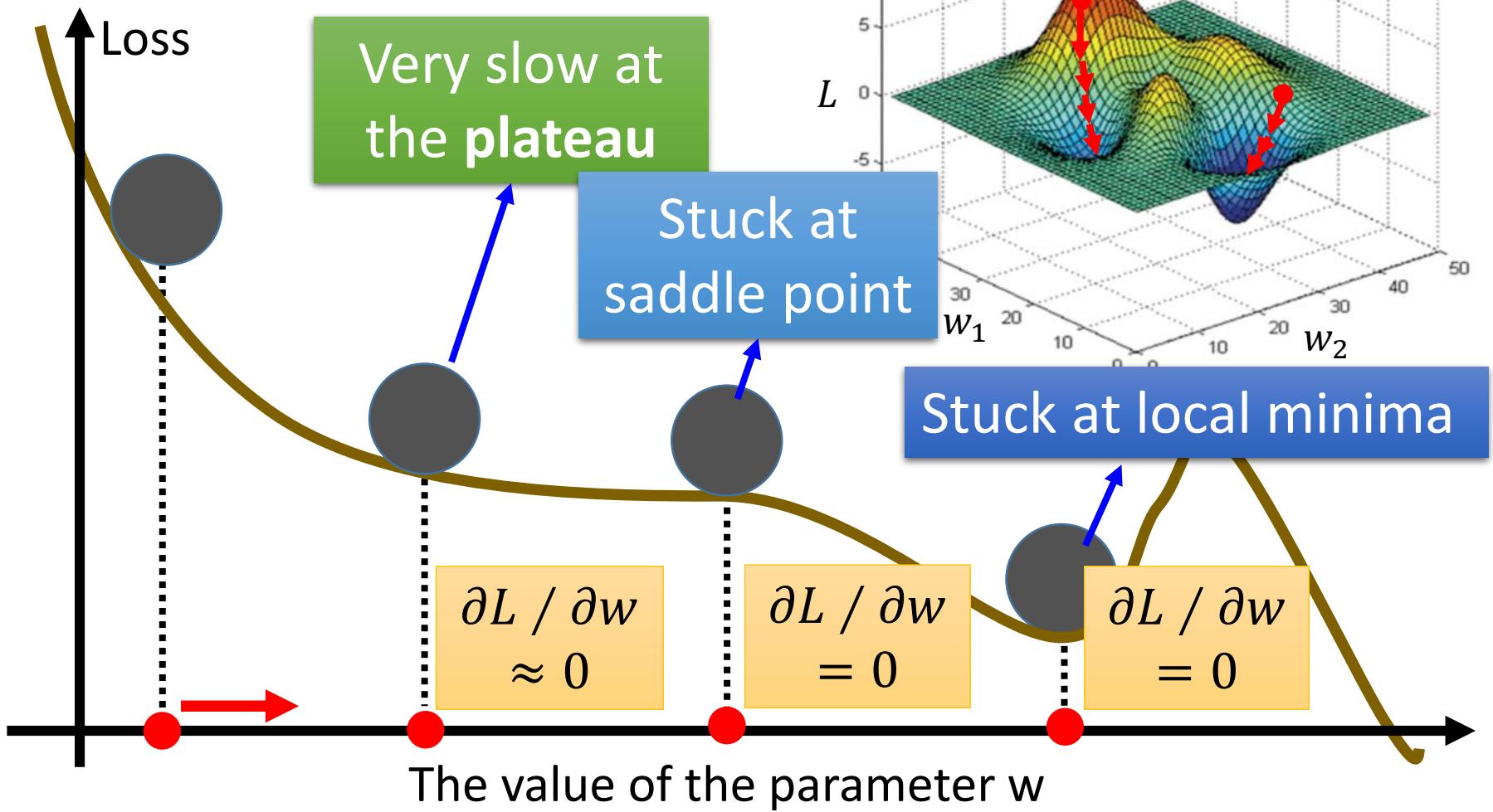
$$\theta^* = \arg \max_{\theta} L(\theta) \quad \text{by gradient descent}$$

- Each time we update the parameters, we obtain θ that makes $L(\theta)$ smaller.

$$L(\theta^0) > L(\theta^1) > L(\theta^2) > \dots$$

Is this statement correct?

Step 3: Gradient Descent



Step 3: Gradient Descent

- Formulation of $\partial L / \partial w$ and $\partial L / \partial b$

$$L(w, b) = \sum_{n=1}^{10} \left(\hat{y}^n - \left(b + \underline{w \cdot x_{cp}^n} \right) \right)^2$$

$$\frac{\partial L}{\partial w} = ? \sum_{n=1}^{10} 2 \left(\hat{y}^n - \left(b + w \cdot x_{cp}^n \right) \right)$$

$$\frac{\partial L}{\partial b} = ?$$

Step 3: Gradient Descent

- Formulation of $\partial L / \partial w$ and $\partial L / \partial b$

$$L(w, b) = \sum_{n=1}^{10} \left(\hat{y}^n - (\underline{b} + w \cdot x_{cp}^n) \right)^2$$

$$\frac{\partial L}{\partial w} = ? \sum_{n=1}^{10} 2 \left(\hat{y}^n - (b + w \cdot x_{cp}^n) \right) (-x_{cp}^n)$$

$$\frac{\partial L}{\partial b} = ? \sum_{n=1}^{10} 2 \left(\hat{y}^n - (b + w \cdot x_{cp}^n) \right)$$

Step 3: Gradient Descent

How's the results?

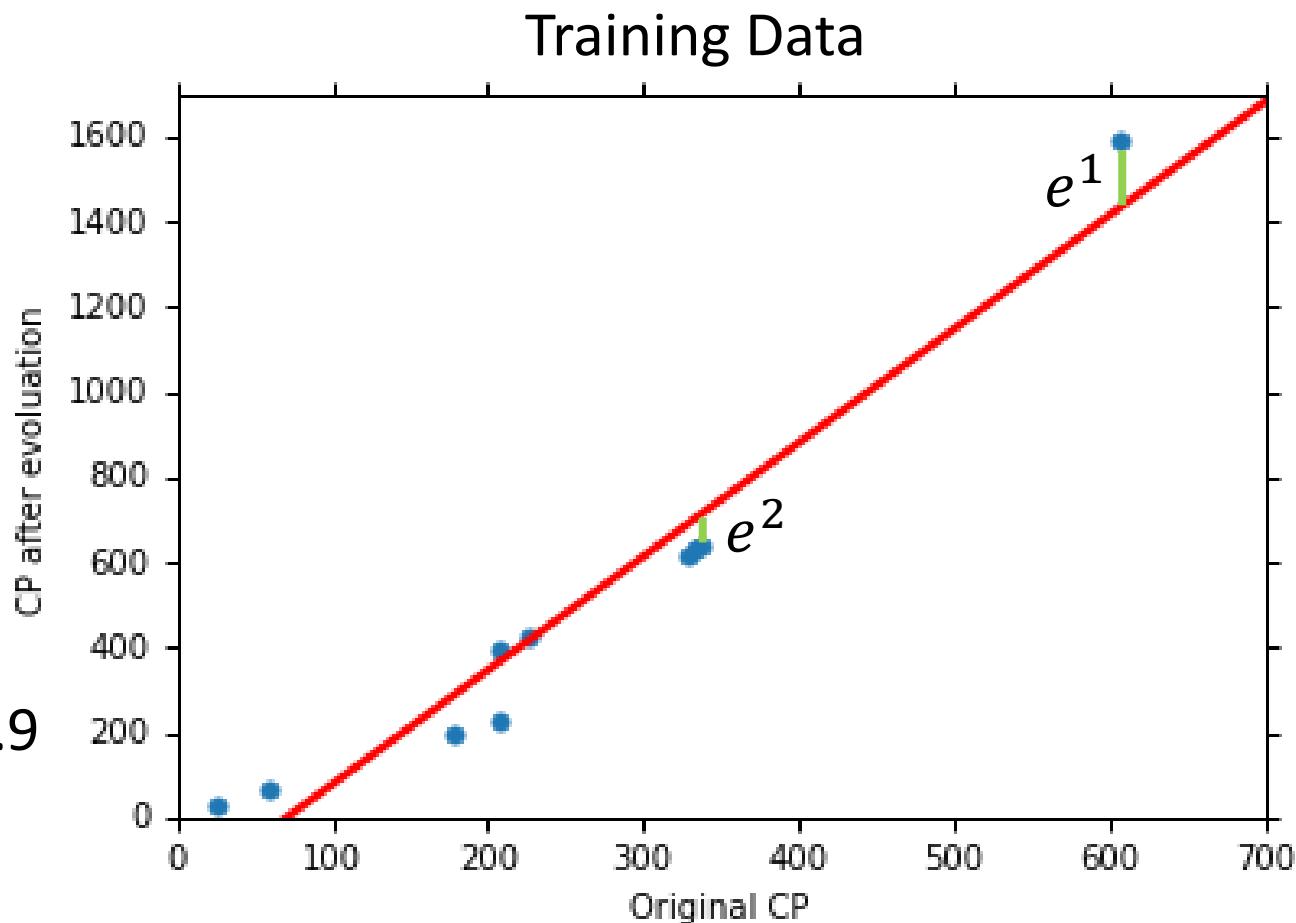
$$y = b + w \cdot x_{cp}$$

$$b = -188.4$$

$$w = 2.7$$

Average Error on
Training Data

$$= \frac{1}{10} \sum_{n=1}^{10} e^n = 31.9$$



How's the results? - Generalization

What we really care about is the error on new data (testing data)

$$y = b + w \cdot x_{cp}$$

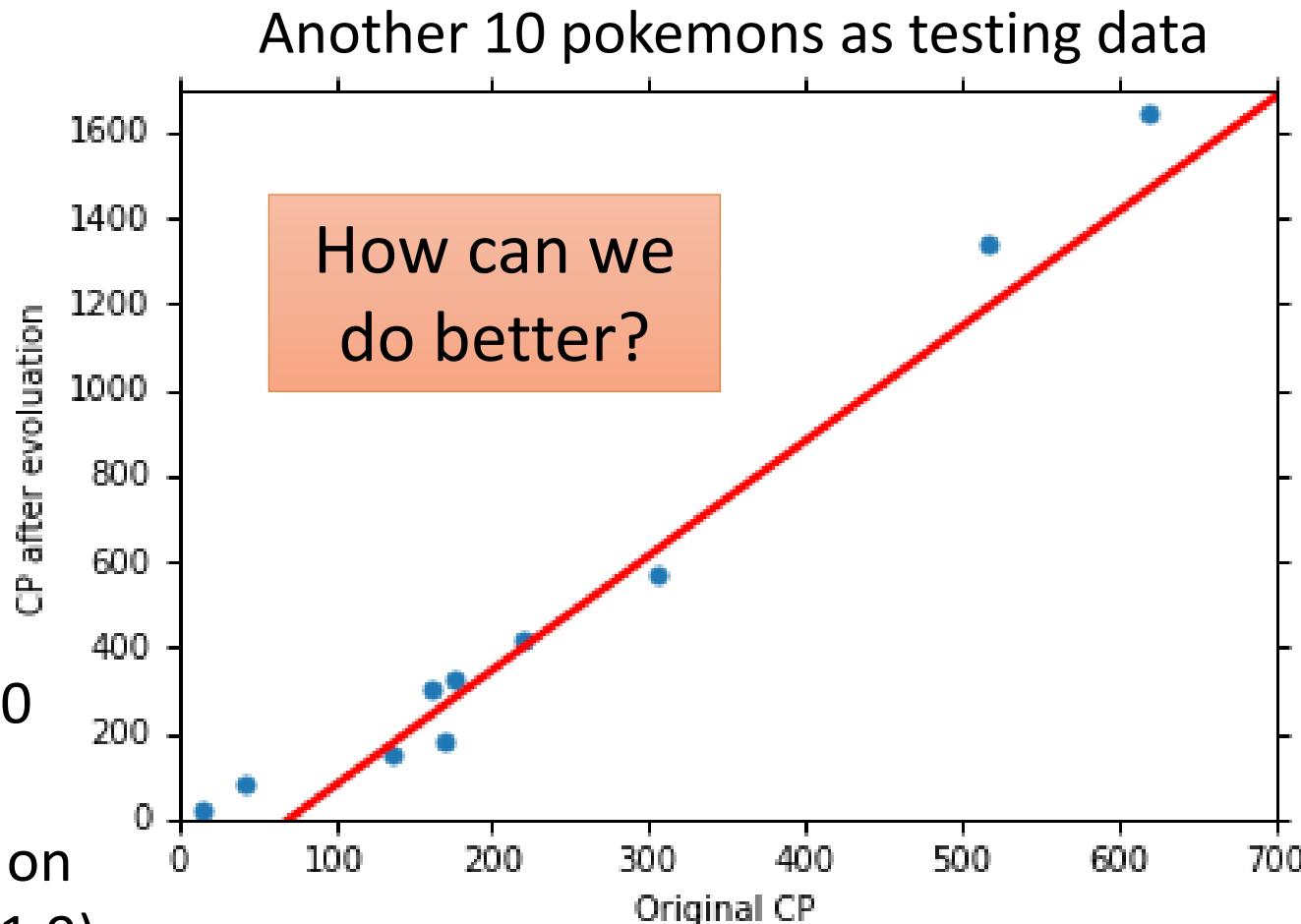
$$b = -188.4$$

$$w = 2.7$$

Average Error on Testing Data

$$= \frac{1}{10} \sum_{n=1}^{10} e^n = 35.0$$

> Average Error on Training Data (31.9)



Selecting another Model

$$y = b + w_1 \cdot x_{cp} + w_2 \cdot (x_{cp})^2$$

Best Function

$$b = -10.3$$

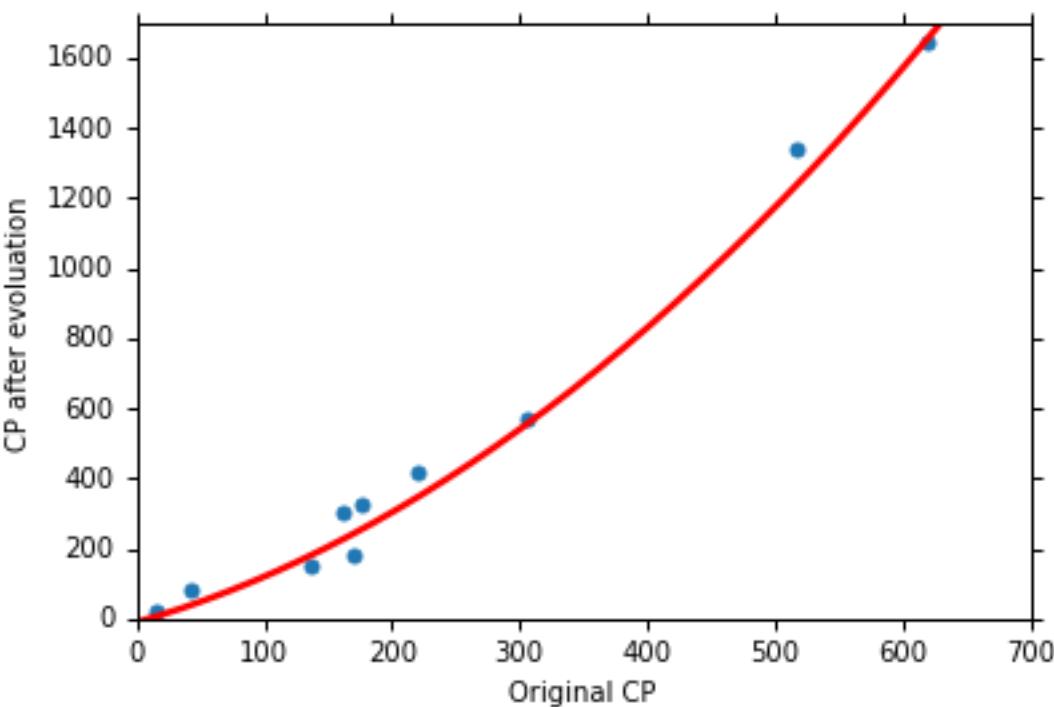
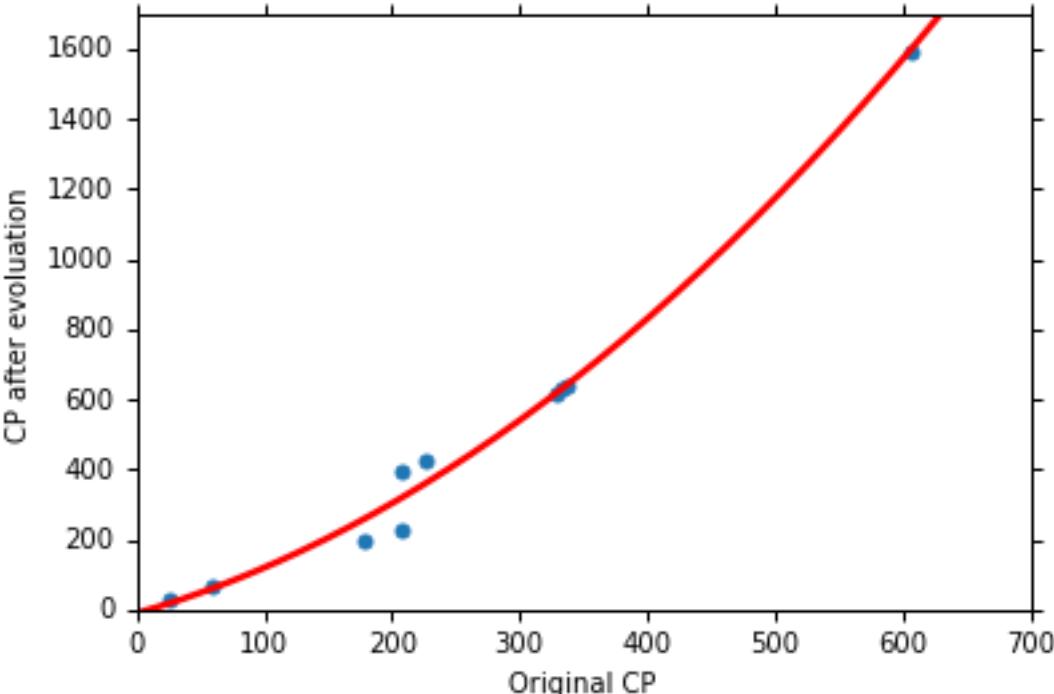
$$w_1 = 1.0, w_2 = 2.7 \times 10^{-3}$$

$$\text{Average Error} = 15.4$$

Testing:

$$\text{Average Error} = 18.4$$

Better! Could it be even better?



Selecting another Model

$$y = b + w_1 \cdot x_{cp} + w_2 \cdot (x_{cp})^2 + w_3 \cdot (x_{cp})^3$$

Best Function

$$b = 6.4, w_1 = 0.66$$

$$w_2 = 4.3 \times 10^{-3}$$

$$w_3 = -1.8 \times 10^{-6}$$

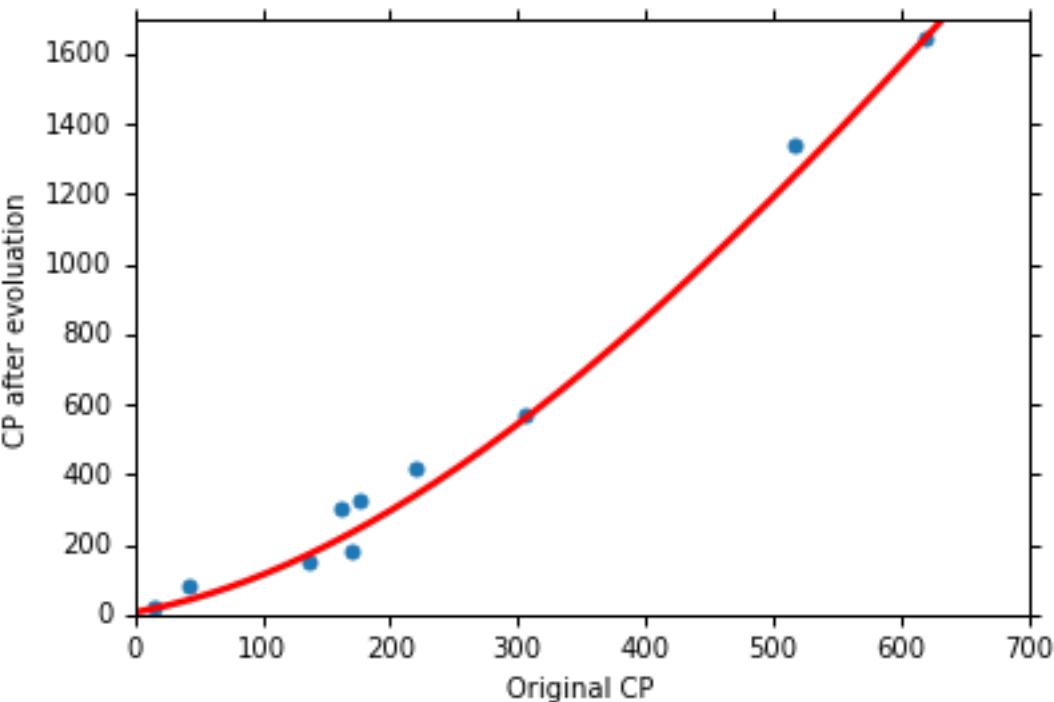
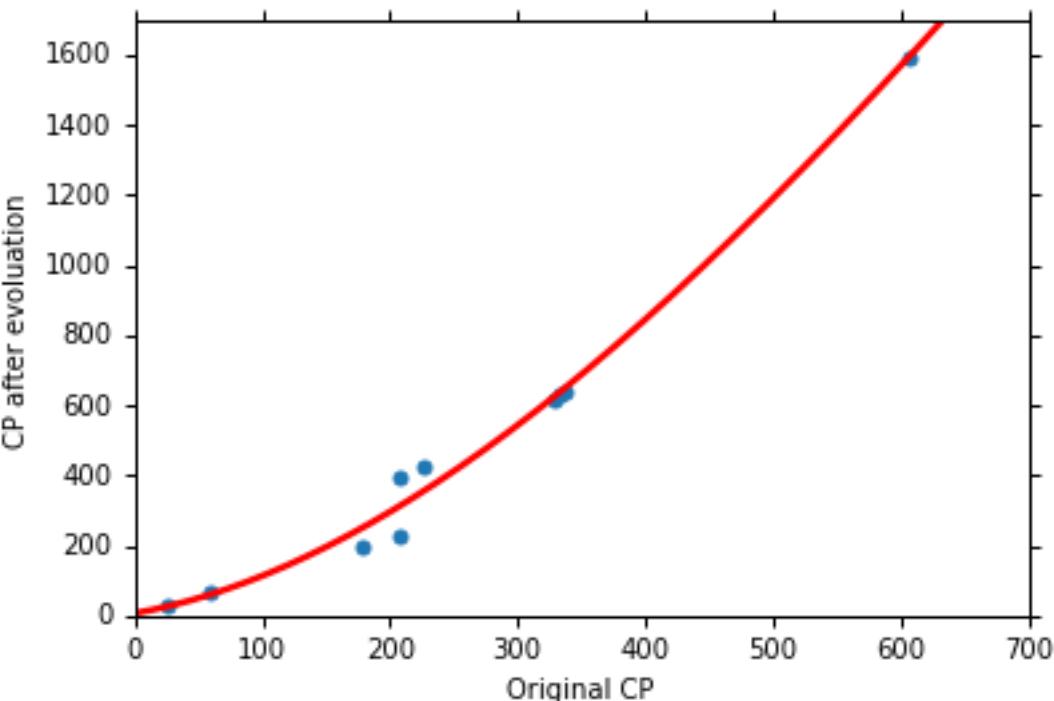
$$\text{Average Error} = 15.3$$

Testing:

$$\text{Average Error} = 18.1$$

Slightly better.

How about more complex model?



Selecting another Model

$$y = b + w_1 \cdot x_{cp} + w_2 \cdot (x_{cp})^2 + w_3 \cdot (x_{cp})^3 + w_4 \cdot (x_{cp})^4$$

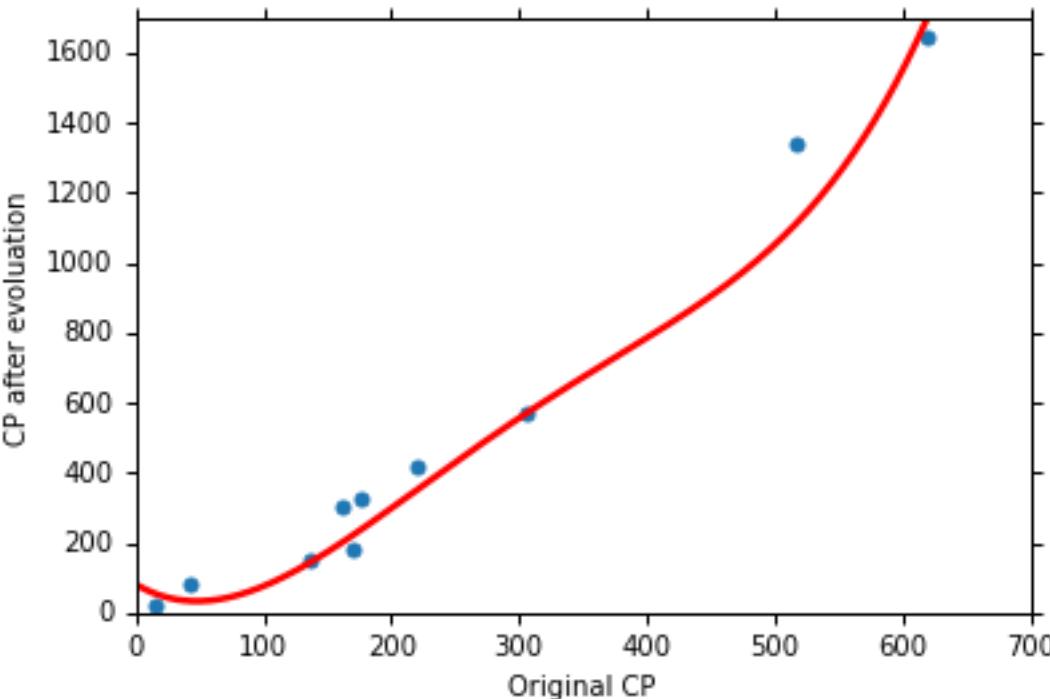
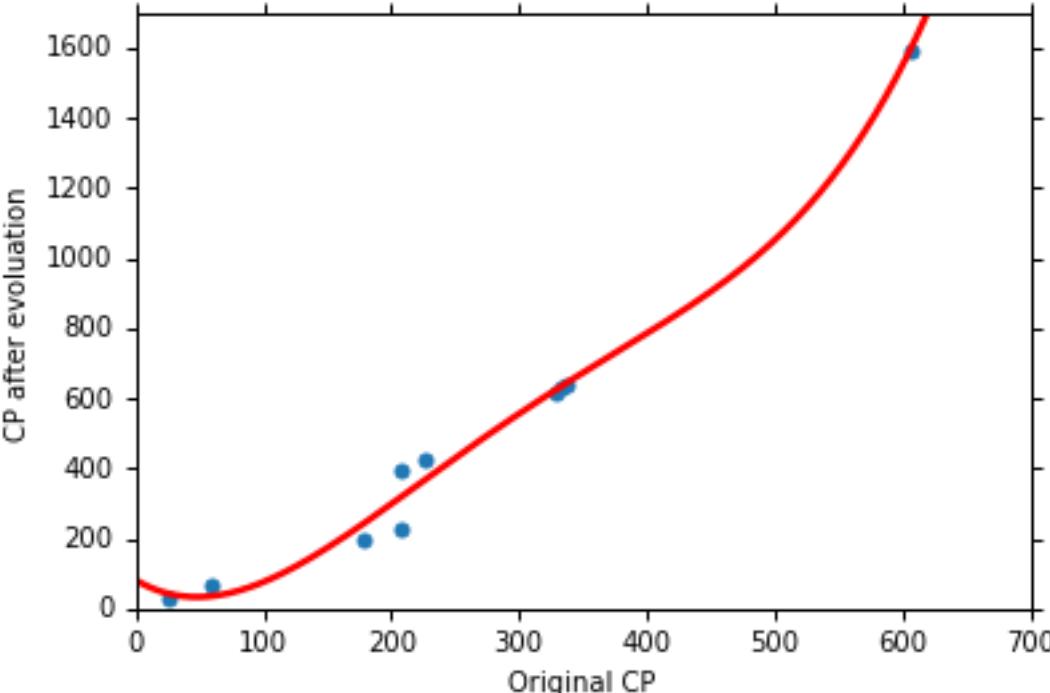
Best Function

Average Error = 14.9

Testing:

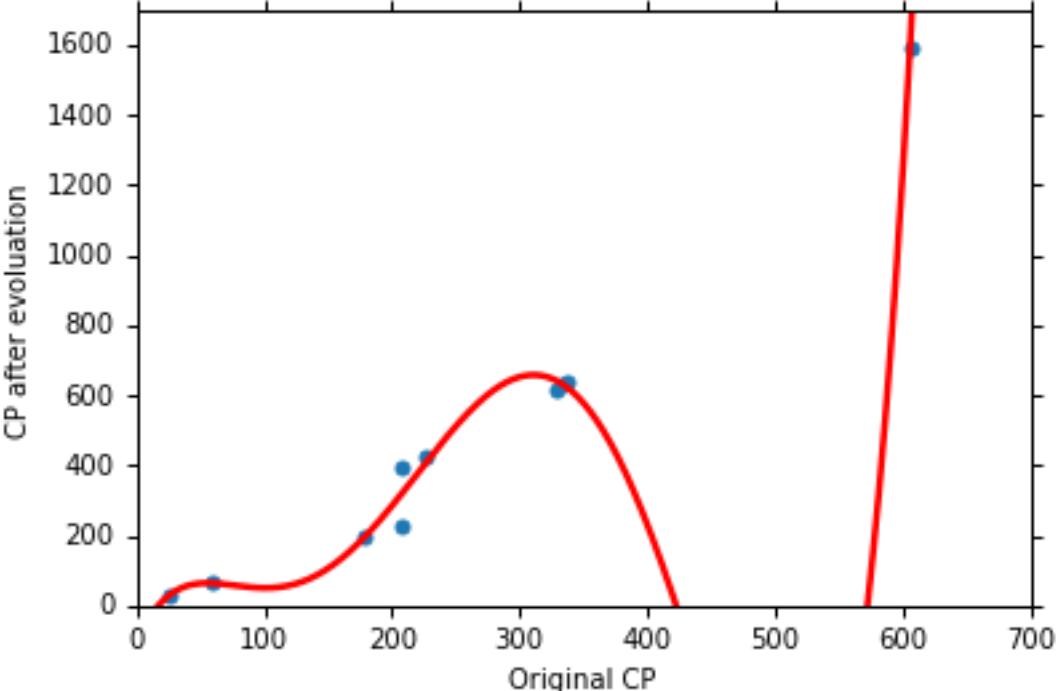
Average Error = 28.8

The results become
worse ...



Selecting another Model

$$y = b + w_1 \cdot x_{cp} + w_2 \cdot (x_{cp})^2 + w_3 \cdot (x_{cp})^3 + w_4 \cdot (x_{cp})^4 + w_5 \cdot (x_{cp})^5$$



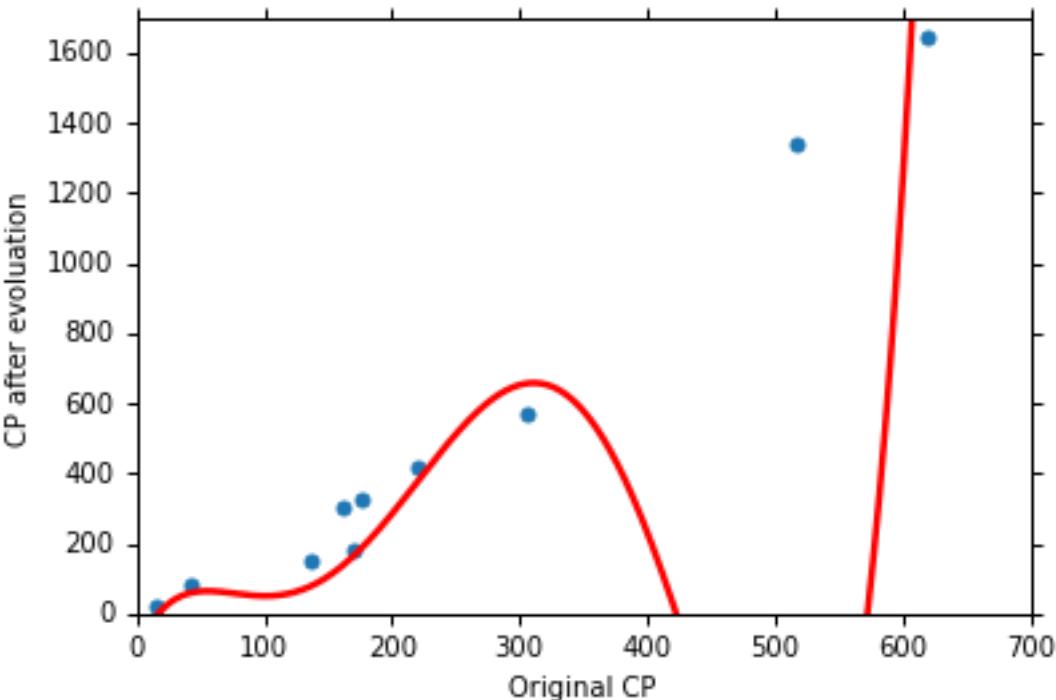
Best Function

Average Error = 12.8

Testing:

Average Error = 232.1

The results are so bad.



Model Selection

1. $y = b + w \cdot x_{cp}$

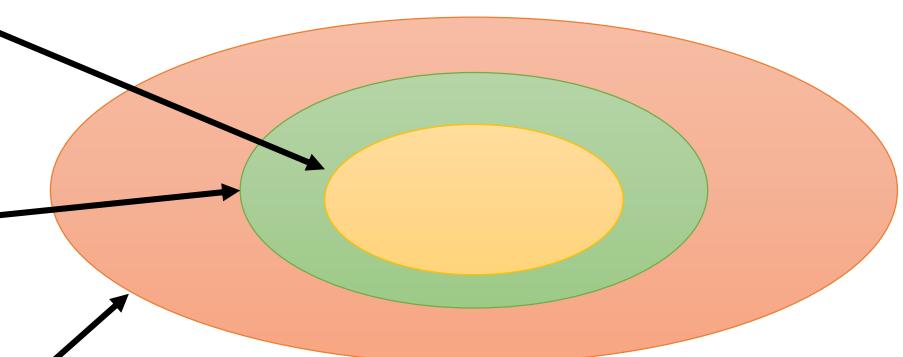
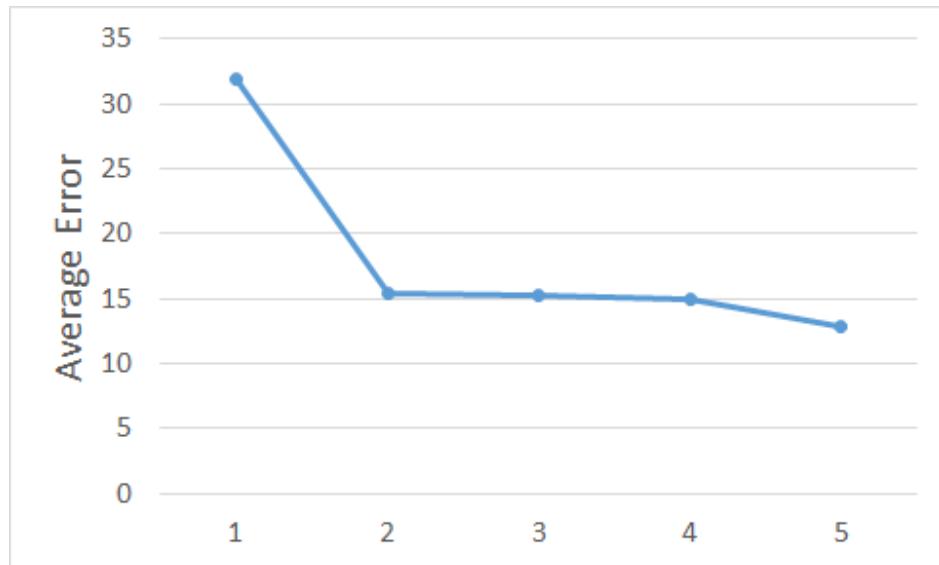
2. $y = b + w_1 \cdot x_{cp} + w_2 \cdot (x_{cp})^2$

3. $y = b + w_1 \cdot x_{cp} + w_2 \cdot (x_{cp})^2 + w_3 \cdot (x_{cp})^3$

4. $y = b + w_1 \cdot x_{cp} + w_2 \cdot (x_{cp})^2 + w_3 \cdot (x_{cp})^3 + w_4 \cdot (x_{cp})^4$

5. $y = b + w_1 \cdot x_{cp} + w_2 \cdot (x_{cp})^2 + w_3 \cdot (x_{cp})^3 + w_4 \cdot (x_{cp})^4 + w_5 \cdot (x_{cp})^5$

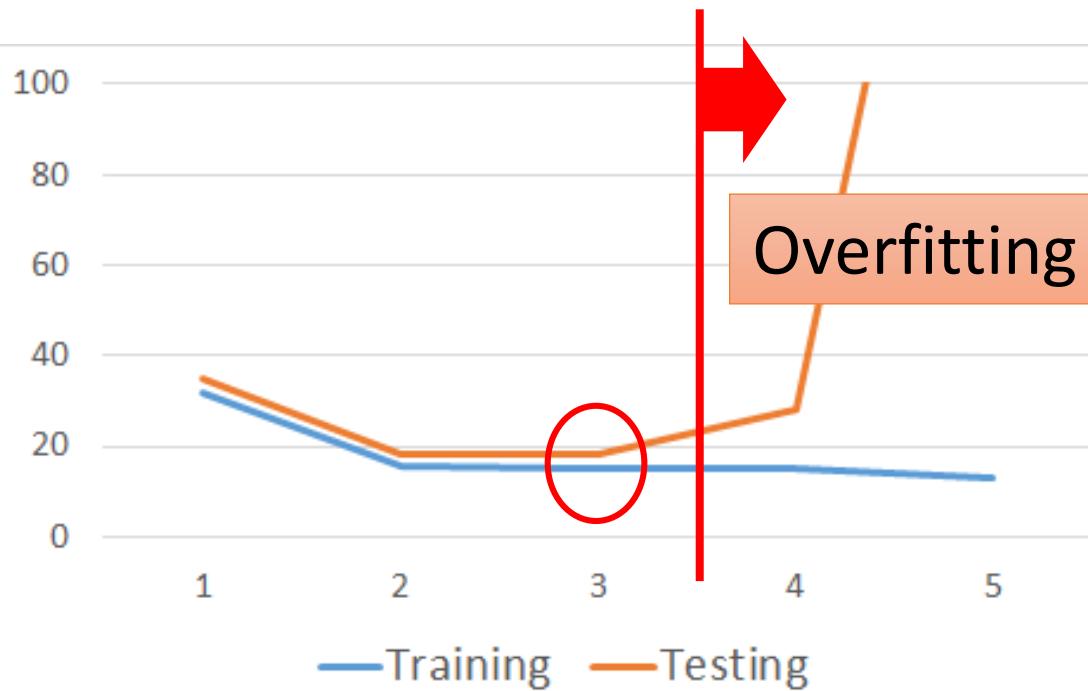
Training Data



A more complex model yields lower error on training data.

If we can truly find the best function

Model Selection

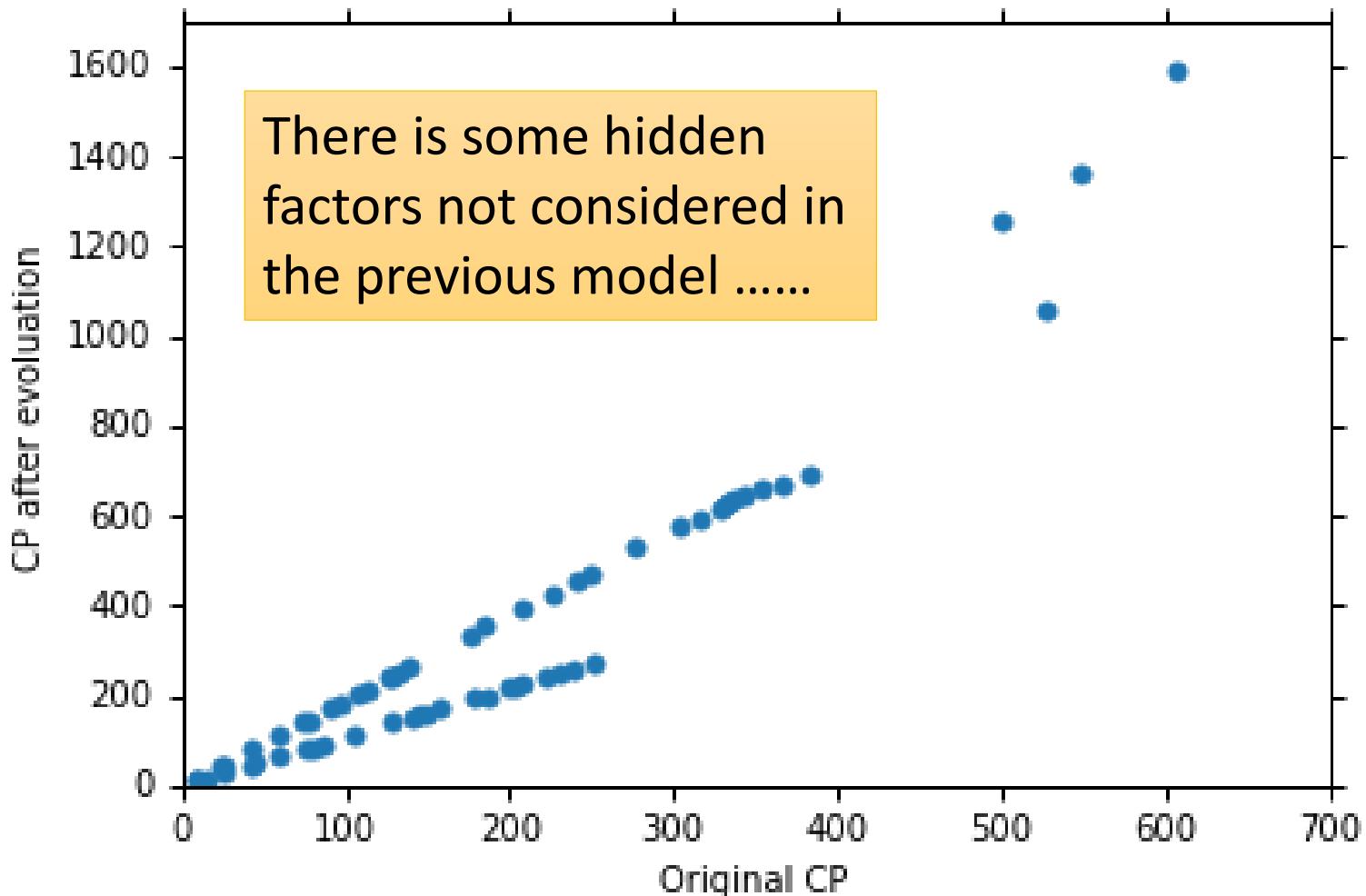


	Training	Testing
1	31.9	35.0
2	15.4	18.4
3	15.3	18.1
4	14.9	28.2
5	12.8	232.1

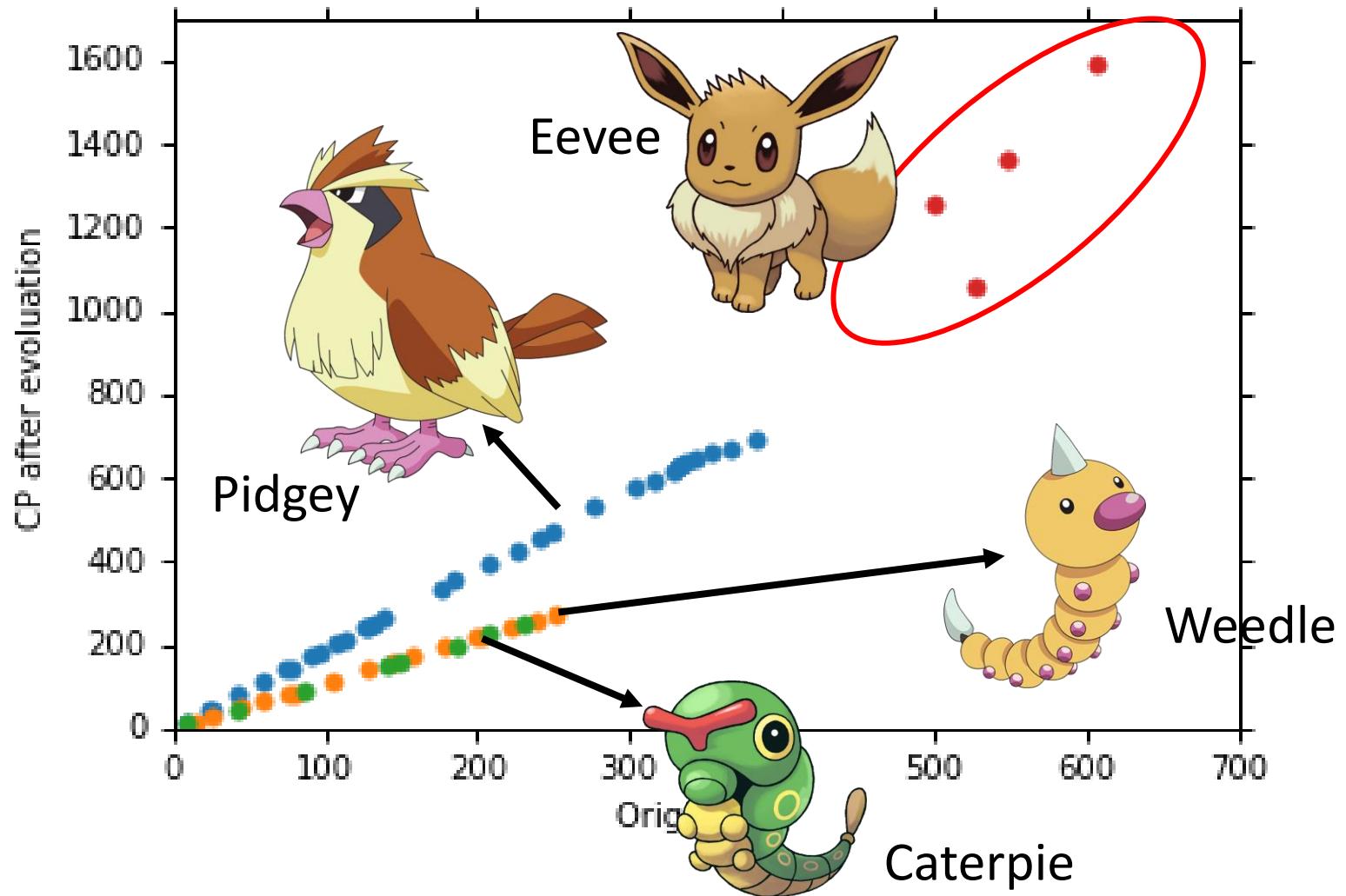
A more complex model does not always lead to better performance on *testing data*.

This is **Overfitting**. ➔ Select suitable model

Let's collect more data



What are the hidden factors?

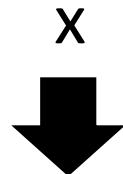


Back to step 1: Redesign the Model

$$y = b + \sum w_i x_i$$

Linear model?

x_s = species of x



If x_s = Pidgey:

$$y = b_1 + w_1 \cdot x_{cp}$$

If x_s = Weedle:

$$y = b_2 + w_2 \cdot x_{cp}$$

If x_s = Caterpie:

$$y = b_3 + w_3 \cdot x_{cp}$$

If x_s = Eevee:

$$y = b_4 + w_4 \cdot x_{cp}$$



Back to step 1: Redesign the Model

$$y = b + \sum w_i x_i$$

Linear model?

$$\begin{aligned} y &= b_1 \cdot 1 \\ &+ w_1 \cdot 1 \quad x_{cp} \\ &+ b_2 \cdot 0 \\ &+ w_2 \cdot 0 \\ &+ b_3 \cdot 0 \\ &+ w_3 \cdot 0 \\ &+ b_4 \cdot 0 \\ &+ w_4 \cdot 0 \end{aligned}$$

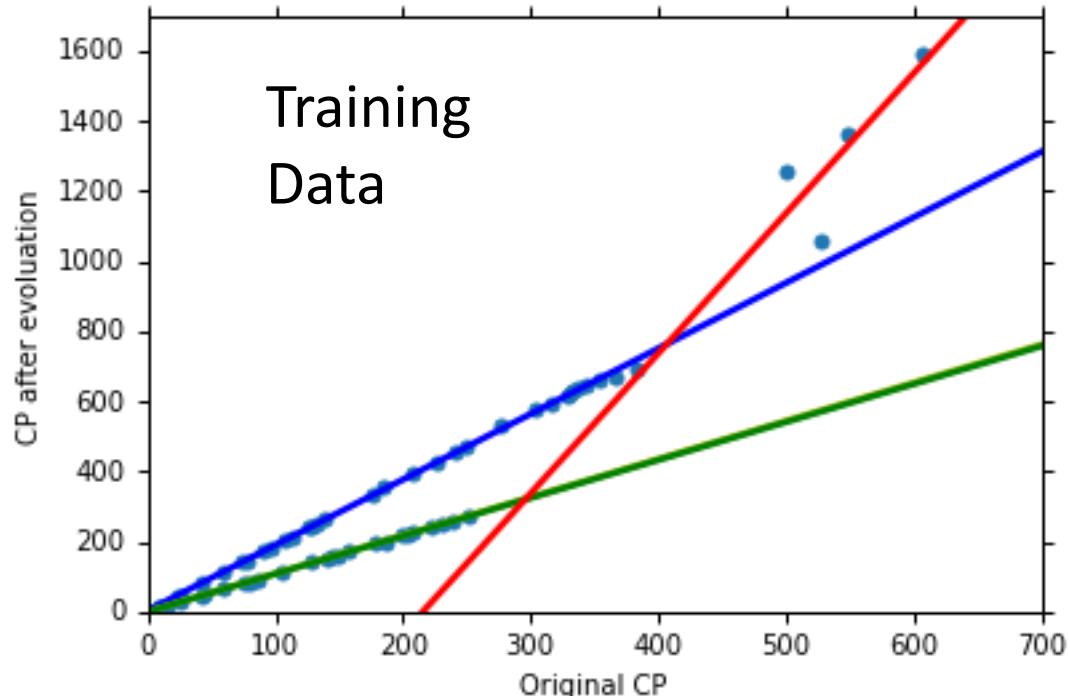
$$\delta(x_s = \text{Pidgey})$$

$$\begin{cases} =1 & \text{If } x_s = \text{Pidgey} \\ =0 & \text{otherwise} \end{cases}$$

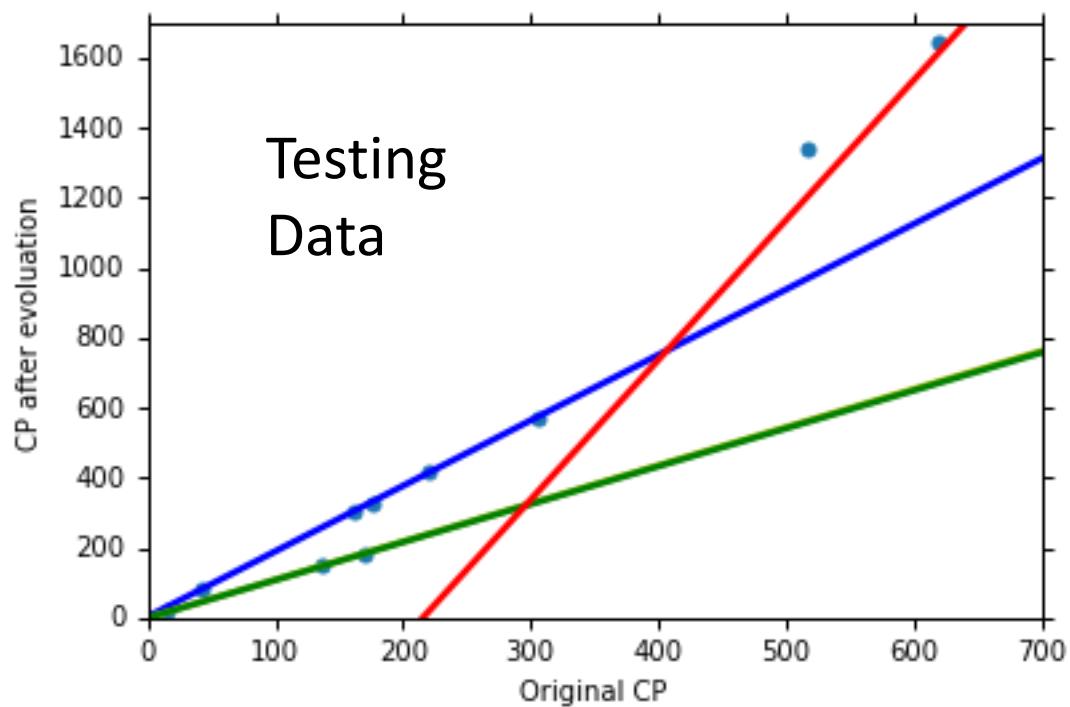
If $x_s = \text{Pidgey}$

$$y = b_1 + w_1 \cdot x_{cp}$$

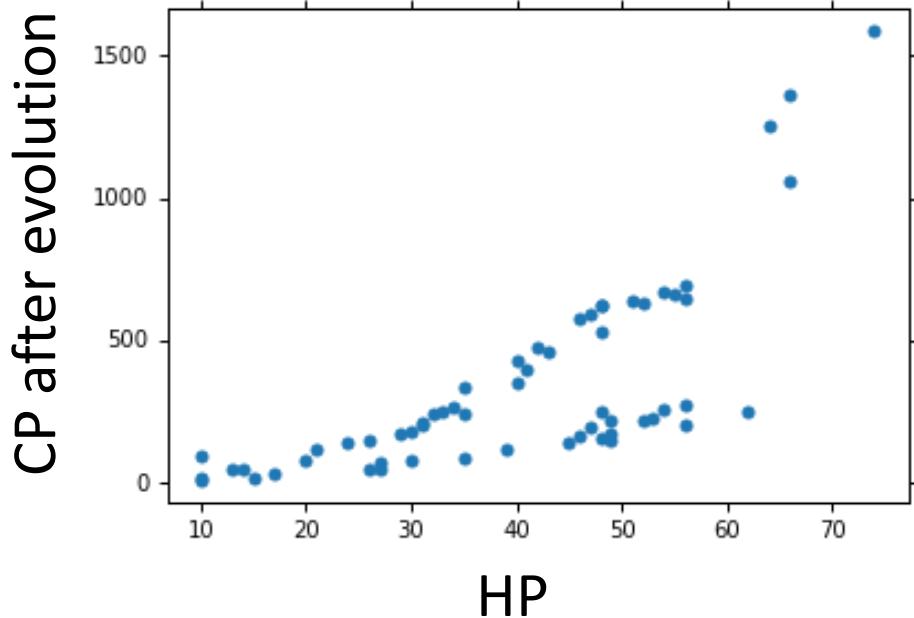
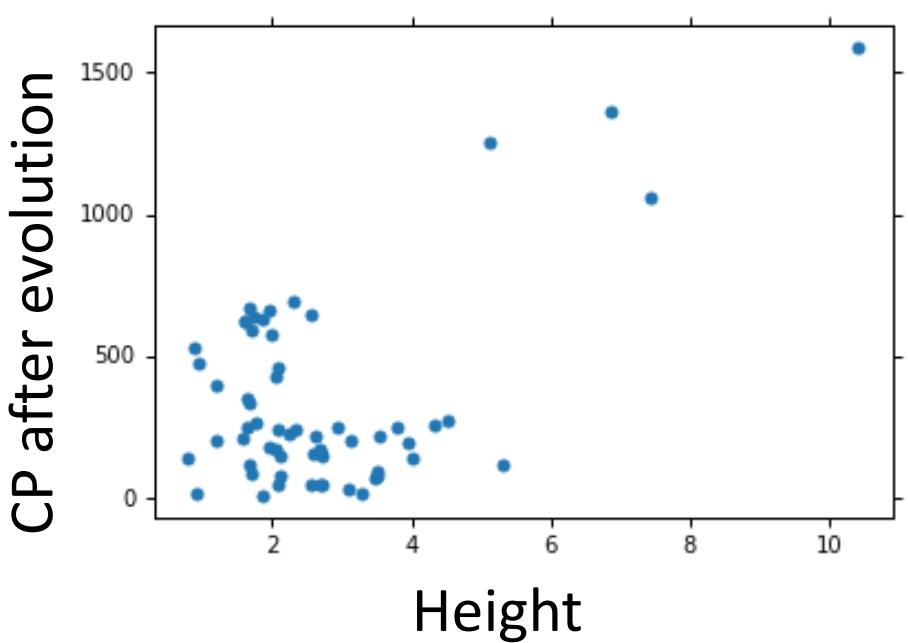
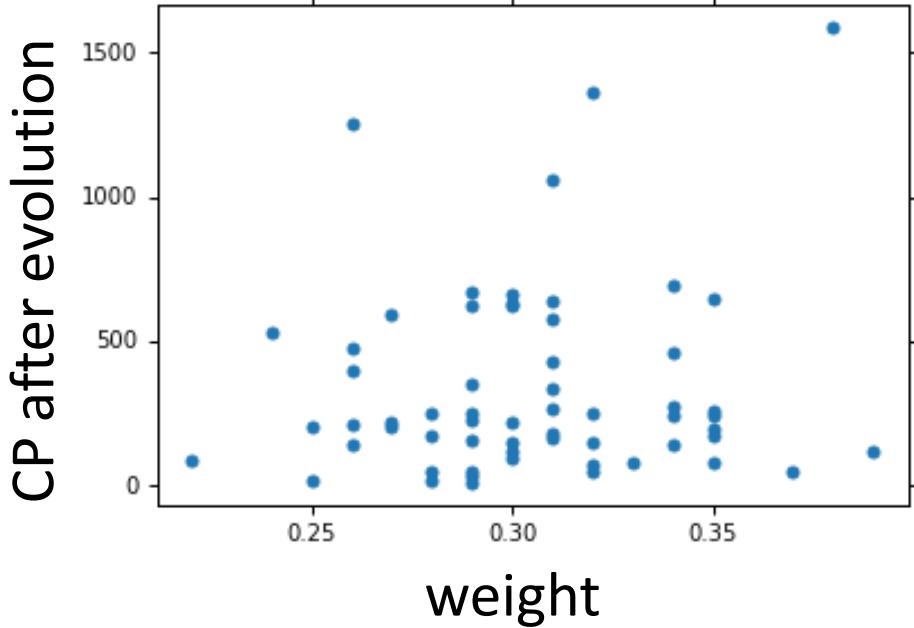
Average error
= 3.8



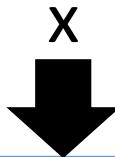
Average error
= 14.3



Are there any other
hidden factors?



Back to step 1: Redesign the Model Again



If $x_s = \text{Pidgey}$:	$y' = b_1 + w_1 \cdot x_{cp} + w_5 \cdot (x_{cp})^2$
If $x_s = \text{Weedle}$:	$y' = b_2 + w_2 \cdot x_{cp} + w_6 \cdot (x_{cp})^2$
If $x_s = \text{Caterpie}$:	$y' = b_3 + w_3 \cdot x_{cp} + w_7 \cdot (x_{cp})^2$
If $x_s = \text{Eevee}$:	$y' = b_4 + w_4 \cdot x_{cp} + w_8 \cdot (x_{cp})^2$
$y = y' + w_9 \cdot x_{hp} + w_{10} \cdot (x_{hp})^2$ $+ w_{11} \cdot x_h + w_{12} \cdot (x_h)^2 + w_{13} \cdot x_w + w_{14} \cdot (x_w)^2$	

Training Error
= 1.9

Testing Error
= 102.3

Overfitting!



Back to step 2: Regularization

$$y = b + \sum w_i x_i$$

$$L = \sum_n \left(\hat{y}^n - \left(b + \sum w_i x_i \right) \right)^2$$

The functions with
smaller w_i are better

$$+ \lambda \sum (w_i)^2$$

- Smaller w_i means ...

smoother

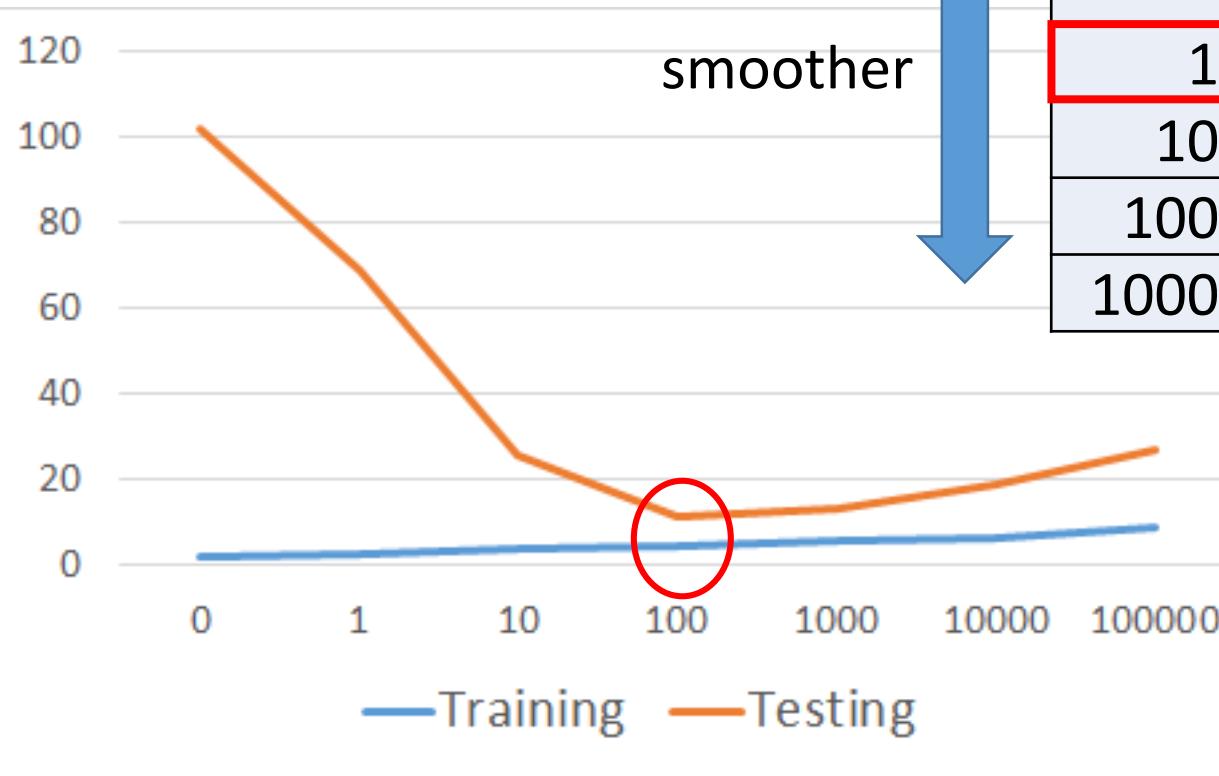
$$y = b + \sum w_i x_i$$

$$y + \sum w_i \Delta x_i = b + \sum w_i (x_i + \Delta x_i)$$

- We believe smoother function is more likely to be correct

Do you have to apply regularization on bias?

Regularization



How smooth?

Select λ obtaining
the best model

- Training error: larger λ , considering the training error less
- We prefer smooth function, but don't be too smooth.

Conclusion

- Pokémon: Original CP and species almost decide the CP after evolution
 - There are probably other hidden factors
- Gradient descent
 - More theory and tips in the following lectures
- We finally get average error = 11.1 on the testing data
 - How about new data? Larger error? Lower error?
- Next lecture: Where does the error come from?
 - More theory about overfitting and regularization
 - The concept of validation

Reference

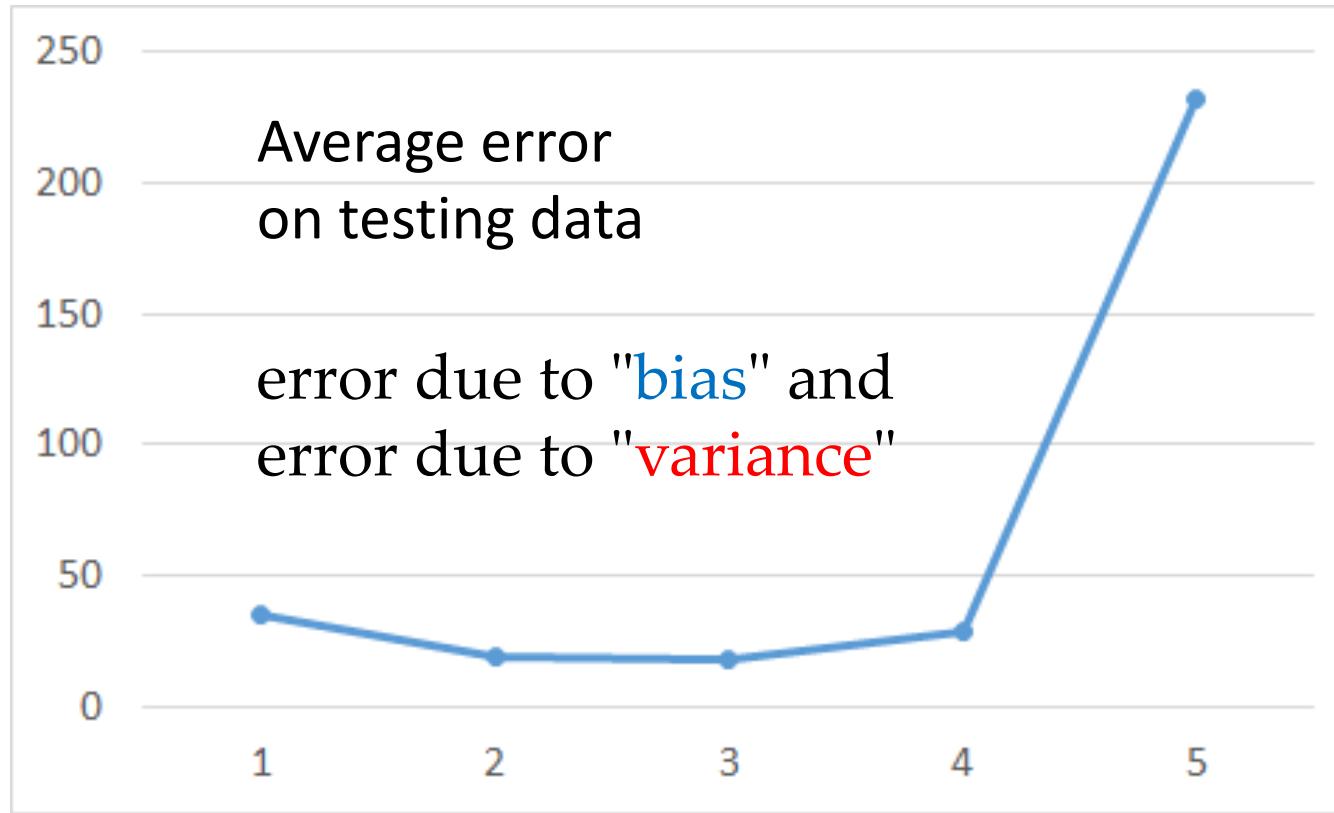
- Bishop: Chapter 1.1

Acknowledgment

- 感謝 鄭凱文 同學發現投影片上的符號錯誤
- 感謝 童寬 同學發現投影片上的符號錯誤
- 感謝 黃振綸 同學發現課程網頁上影片連結錯誤的符號錯誤

Where does the error
come from?

Review



A more complex model does not always lead to better performance on *testing data*.

Estimator

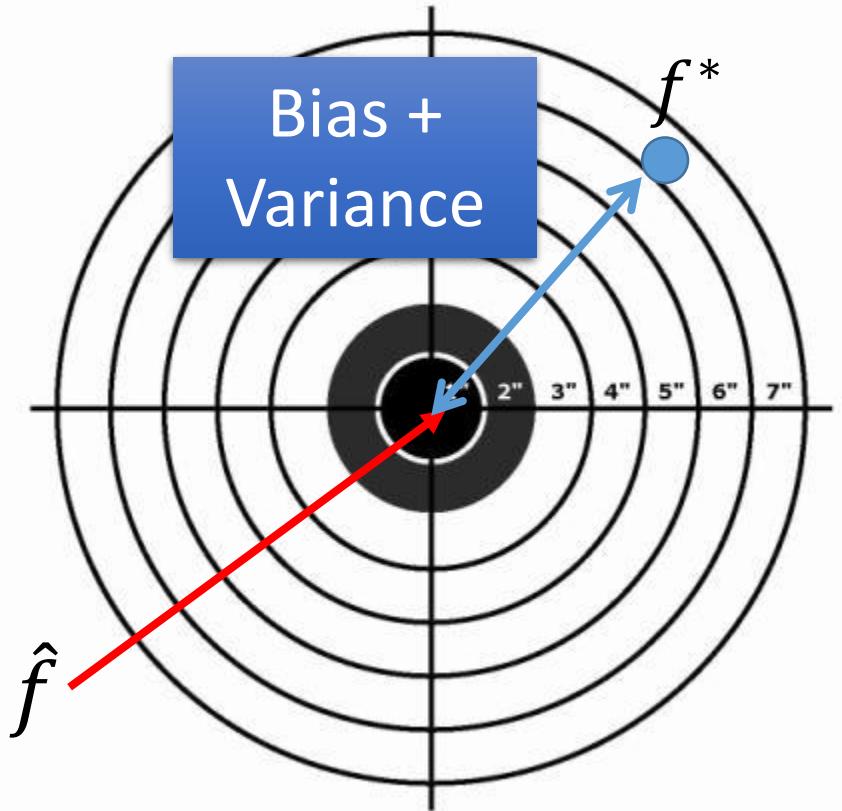
$$\hat{y} = \hat{f}()$$



Only Niantic knows \hat{f}

From training data,
we find f^*

f^* is an estimator of \hat{f}

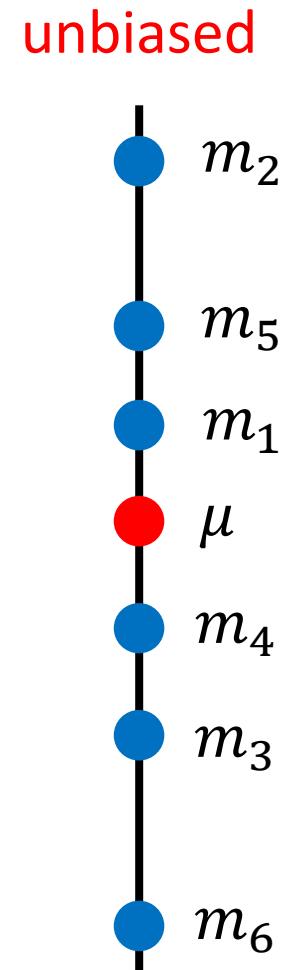


Bias and Variance of Estimator

- Estimate the mean of a variable x
 - assume the mean of x is μ
 - assume the variance of x is σ^2
- Estimator of mean μ
 - Sample N points: $\{x^1, x^2, \dots, x^N\}$

$$m = \frac{1}{N} \sum_n x^n \neq \mu$$

$$E[m] = E \left[\frac{1}{N} \sum_n x^n \right] = \frac{1}{N} \sum_n E[x^n] = \mu$$



Bias and Variance of Estimator

- Estimate the mean of a variable x
 - assume the mean of x is μ
 - assume the variance of x is σ^2
- Estimator of mean μ
 - Sample N points: $\{x^1, x^2, \dots, x^N\}$

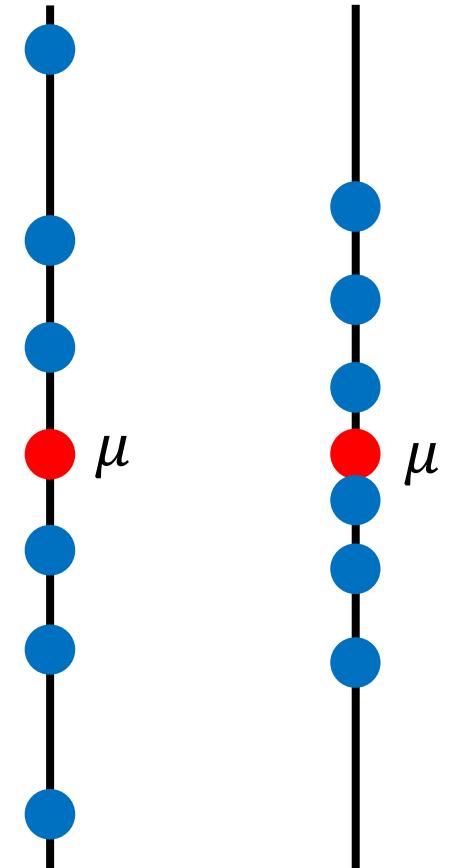
$$m = \frac{1}{N} \sum_n x^n \neq \mu$$

$$\text{Var}[m] = \frac{\sigma^2}{N}$$

Variance depends
on the number of
samples

unbiased

Smaller N Larger N



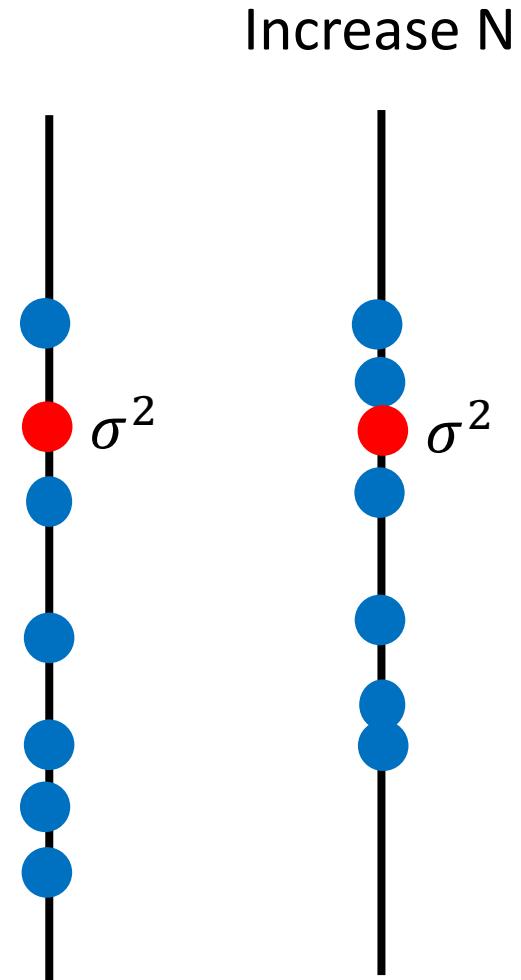
Bias and Variance of Estimator

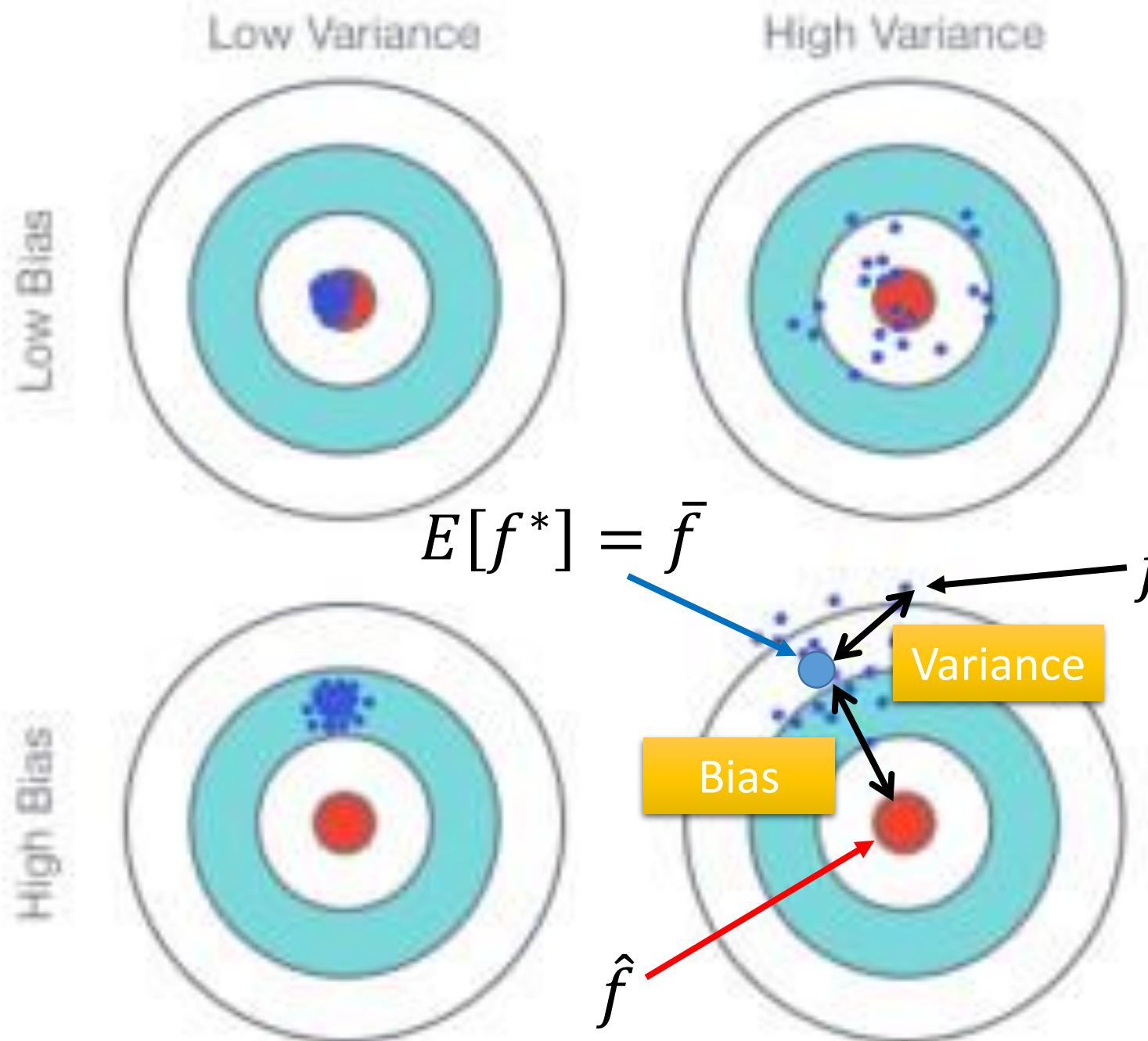
- Estimate the mean of a variable x
 - assume the mean of x is μ
 - assume the variance of x is σ^2
- Estimator of variance σ^2
 - Sample N points: $\{x^1, x^2, \dots, x^N\}$

$$m = \frac{1}{N} \sum_n x^n \quad s^2 = \frac{1}{N} \sum_n (x^n - m)^2$$

Biased estimator

$$E[s^2] = \frac{N-1}{N} \sigma^2 \neq \sigma^2$$

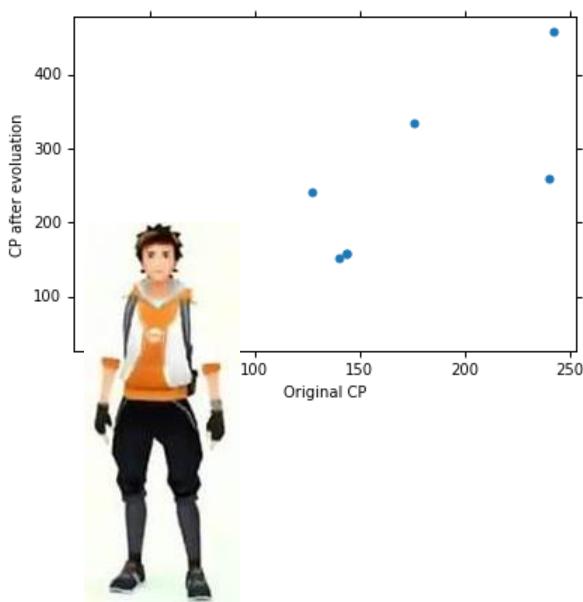




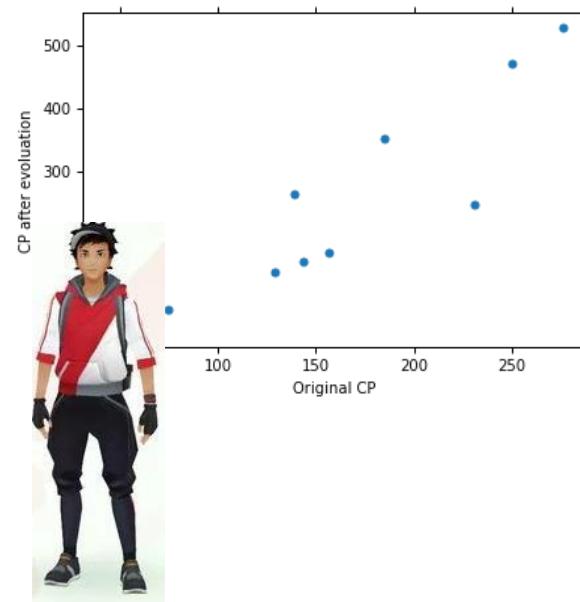
Parallel Universes

- In all the universes, we are collecting (catching) 10 Pokémons as training data to find f^*

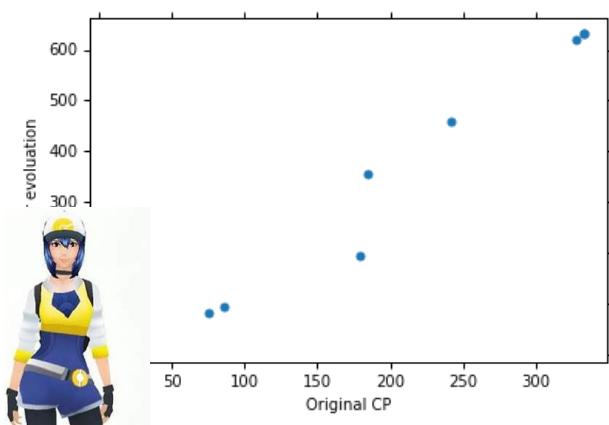
Universe 1



Universe 2



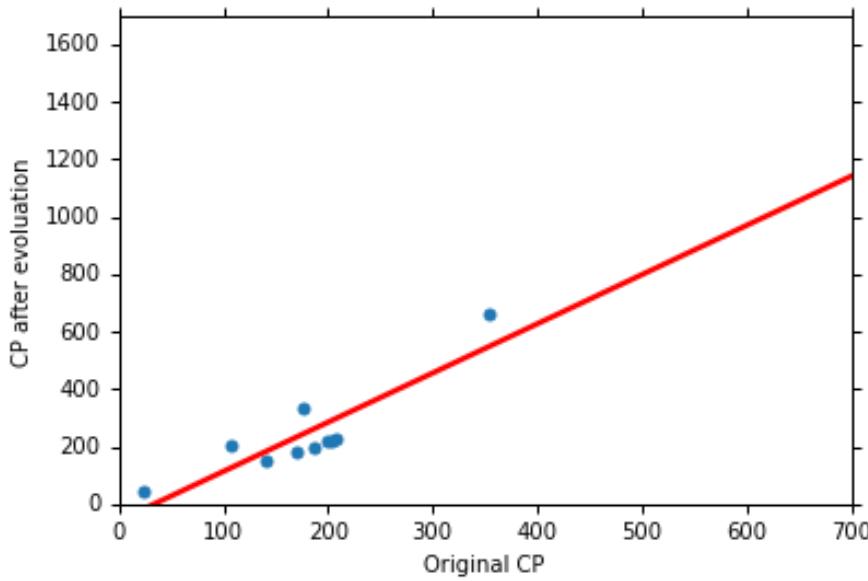
Universe 3



Parallel Universes

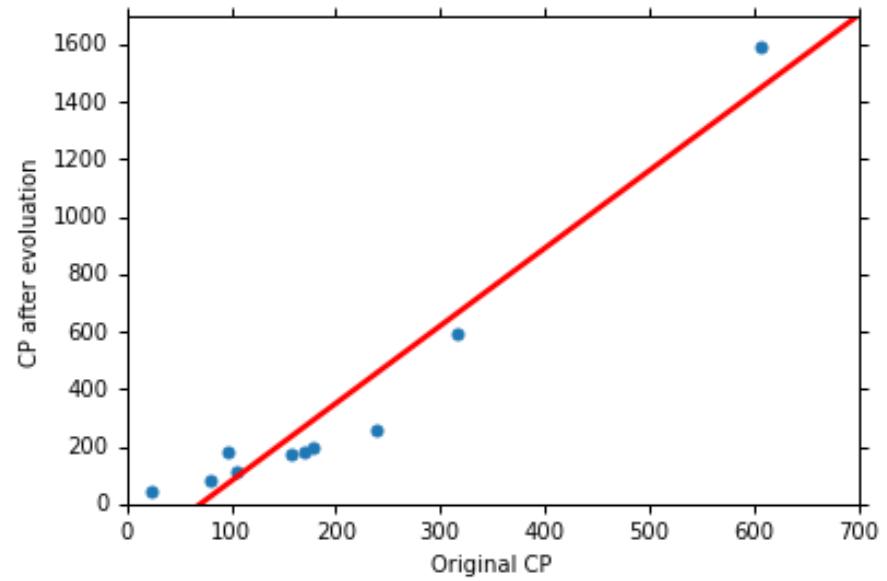
- In different universes, we use the same model, but obtain different f^*

Universe 123



$$y = b + w \cdot x_{cp}$$

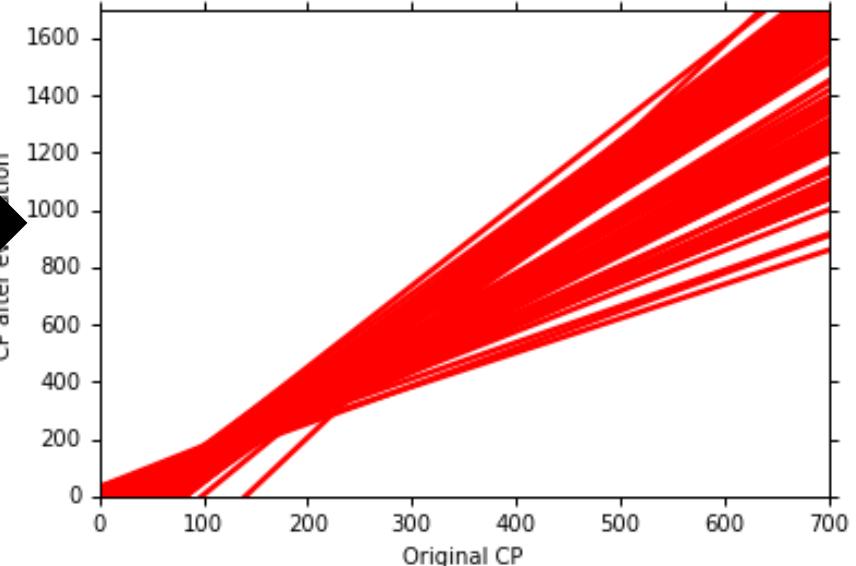
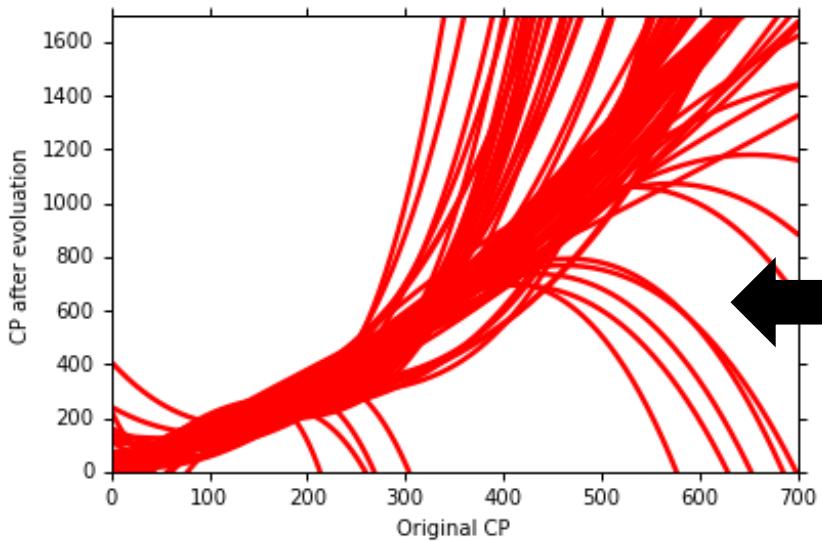
Universe 345



$$y = b + w \cdot x_{cp}$$

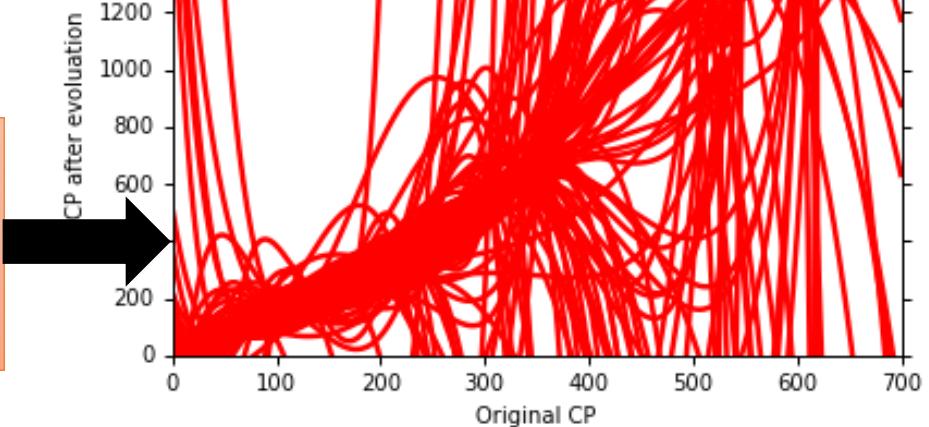
f^* in 100 Universes

$$y = b + w \cdot x_{cp}$$



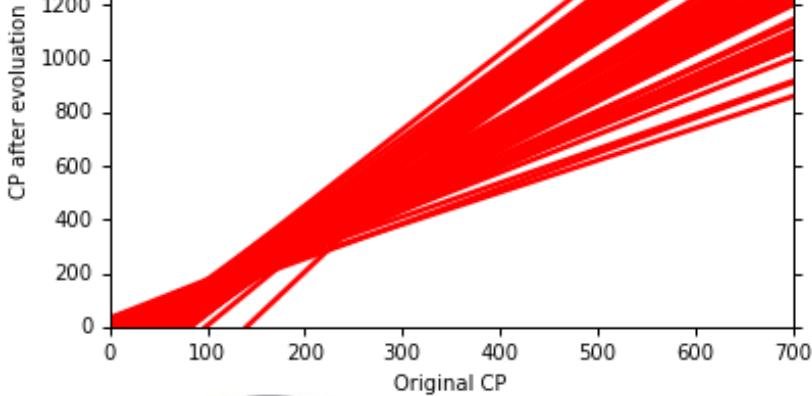
$$y = b + w_1 \cdot x_{cp} + w_2 \cdot (x_{cp})^2 + w_3 \cdot (x_{cp})^3$$

$$y = b + w_1 \cdot x_{cp} + w_2 \cdot (x_{cp})^2 + w_3 \cdot (x_{cp})^3 + w_4 \cdot (x_{cp})^4 + w_5 \cdot (x_{cp})^5$$



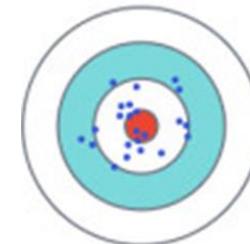
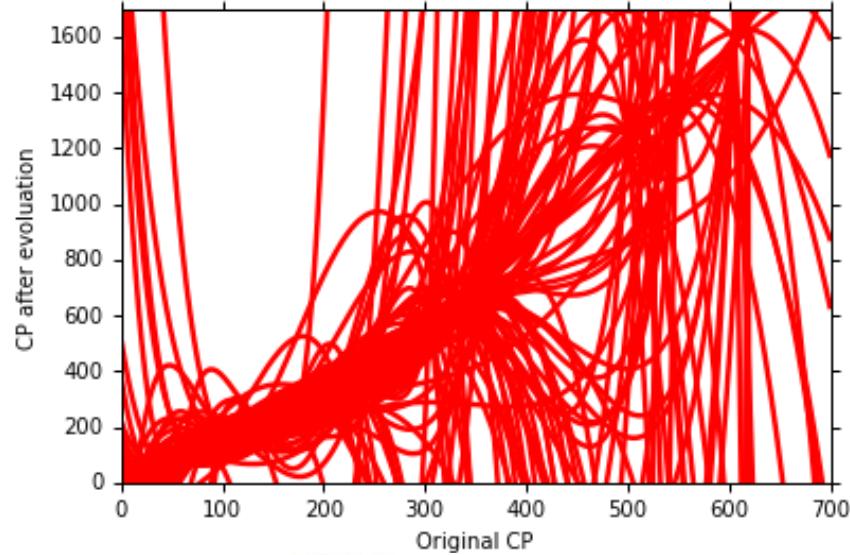
Variance

$$y = b + w \cdot x_{cp}$$



Small
Variance

$$\begin{aligned}y &= b + w_1 \cdot x_{cp} + w_2 \cdot (x_{cp})^2 \\&+ w_3 \cdot (x_{cp})^3 + w_4 \cdot (x_{cp})^4 \\&+ w_5 \cdot (x_{cp})^5\end{aligned}$$



Large
Variance

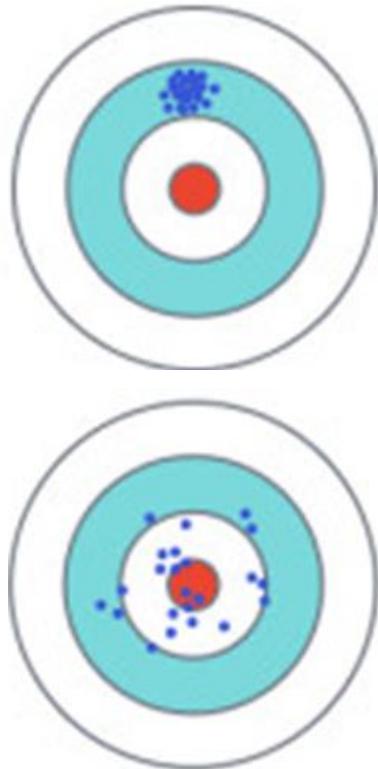
Simpler model is less influenced by the sampled data

Consider the extreme case $f(x) = c$

Bias

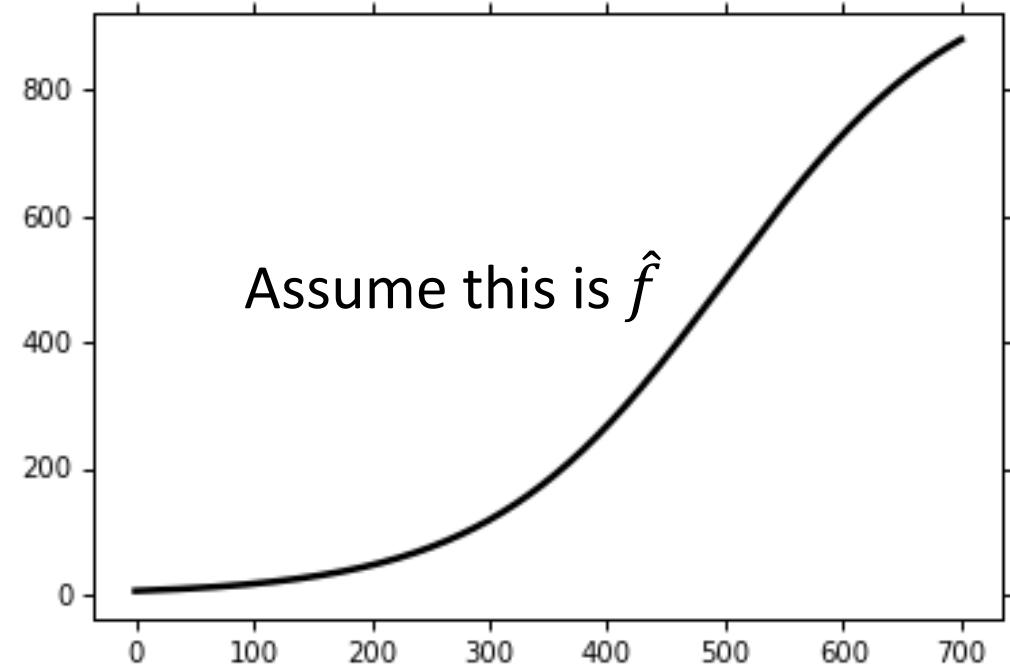
$$E[f^*] = \bar{f}$$

- Bias: If we average all the f^* , is it close to \hat{f}



Large
Bias

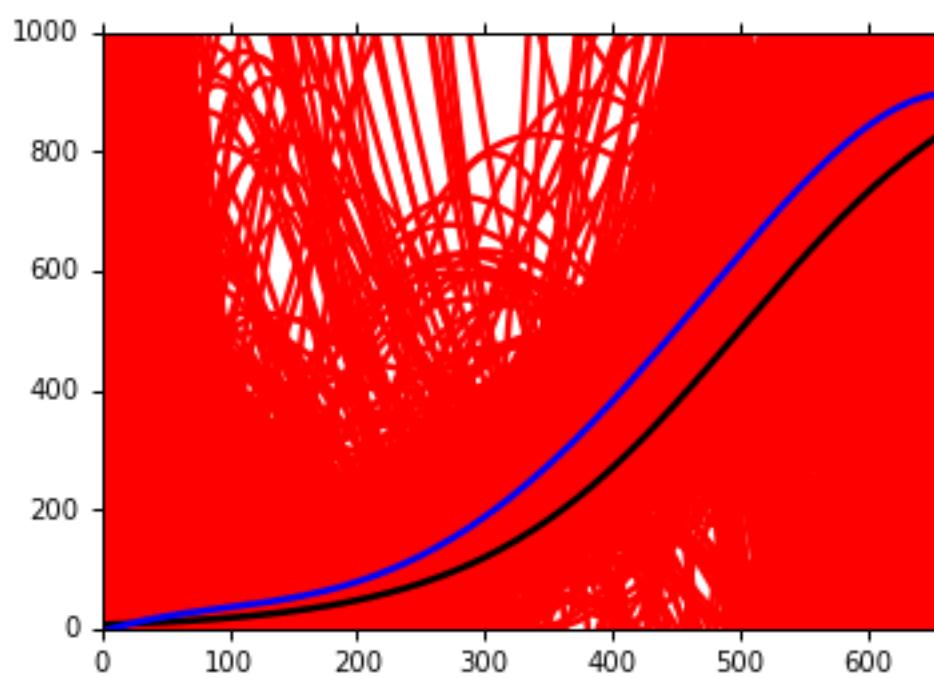
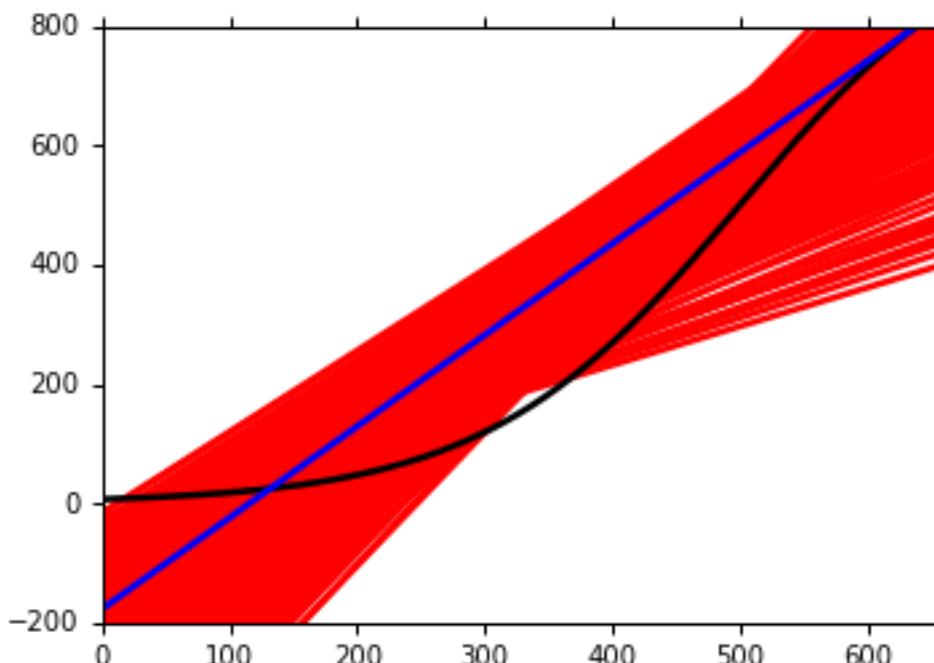
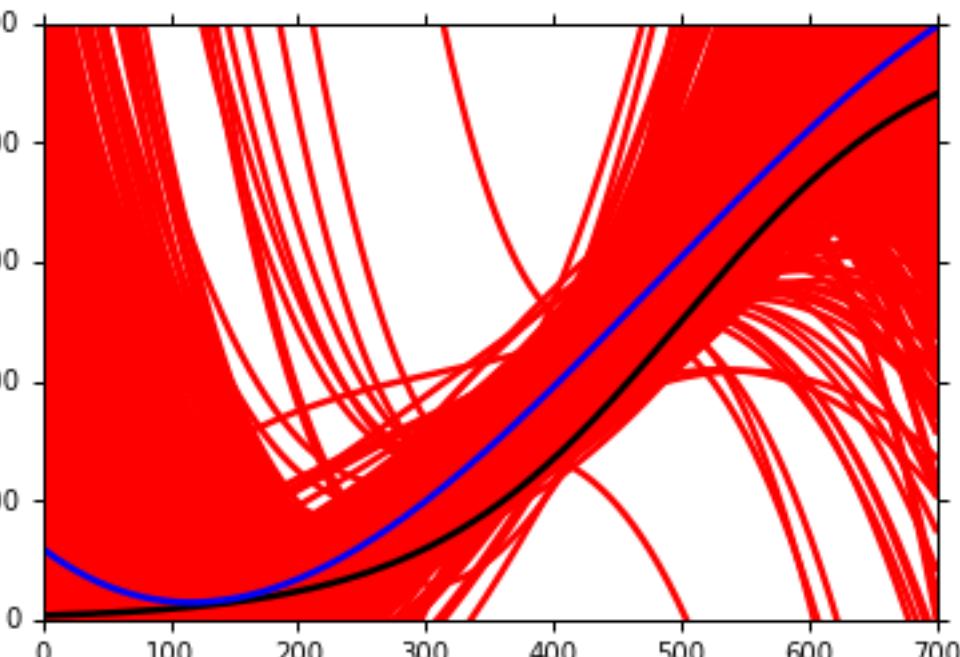
Small
Bias



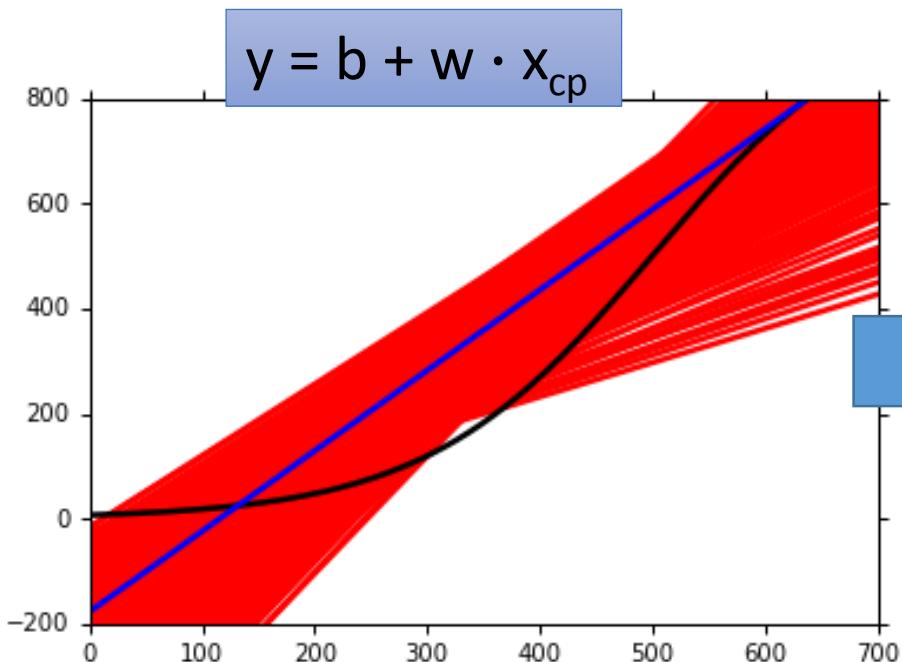
Black curve: the true function \hat{f}

Red curves: 5000 f^*

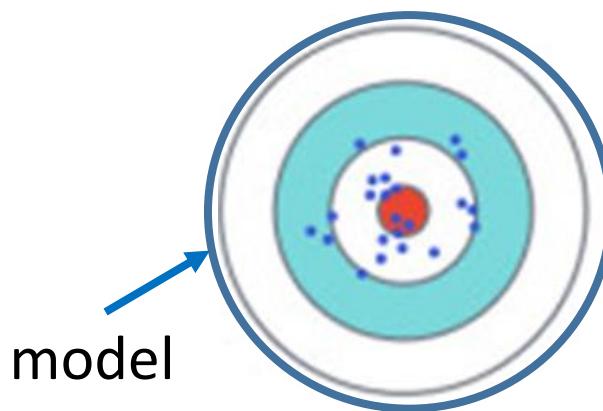
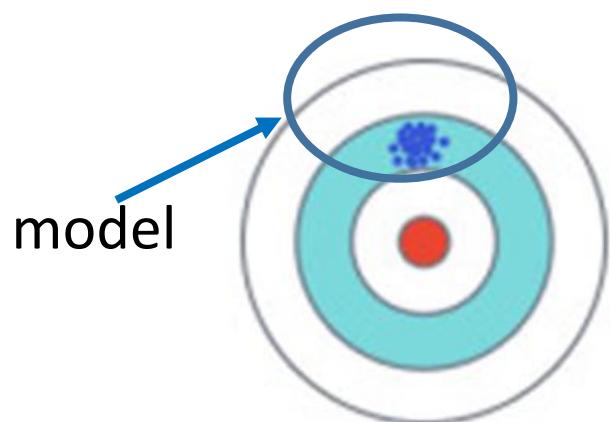
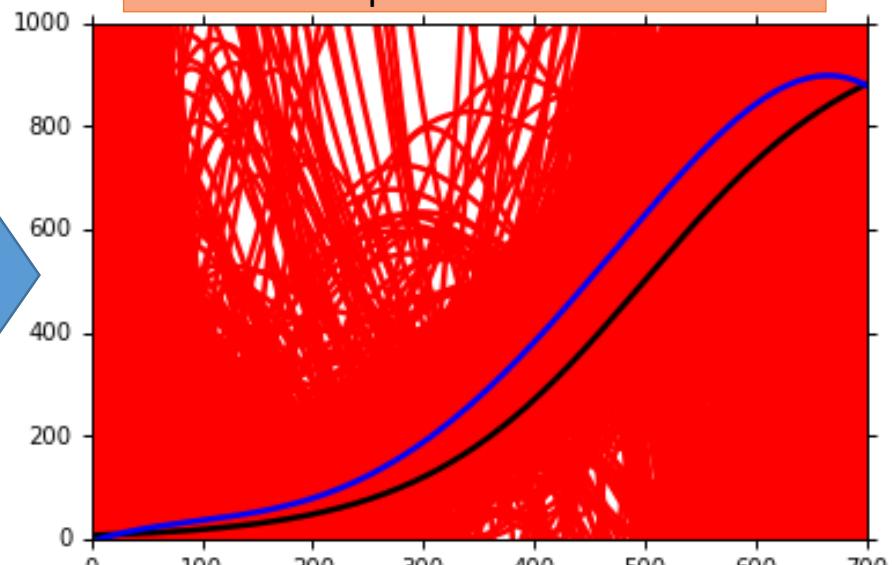
Blue curve: the average of 5000 f^*
 $= \bar{f}$



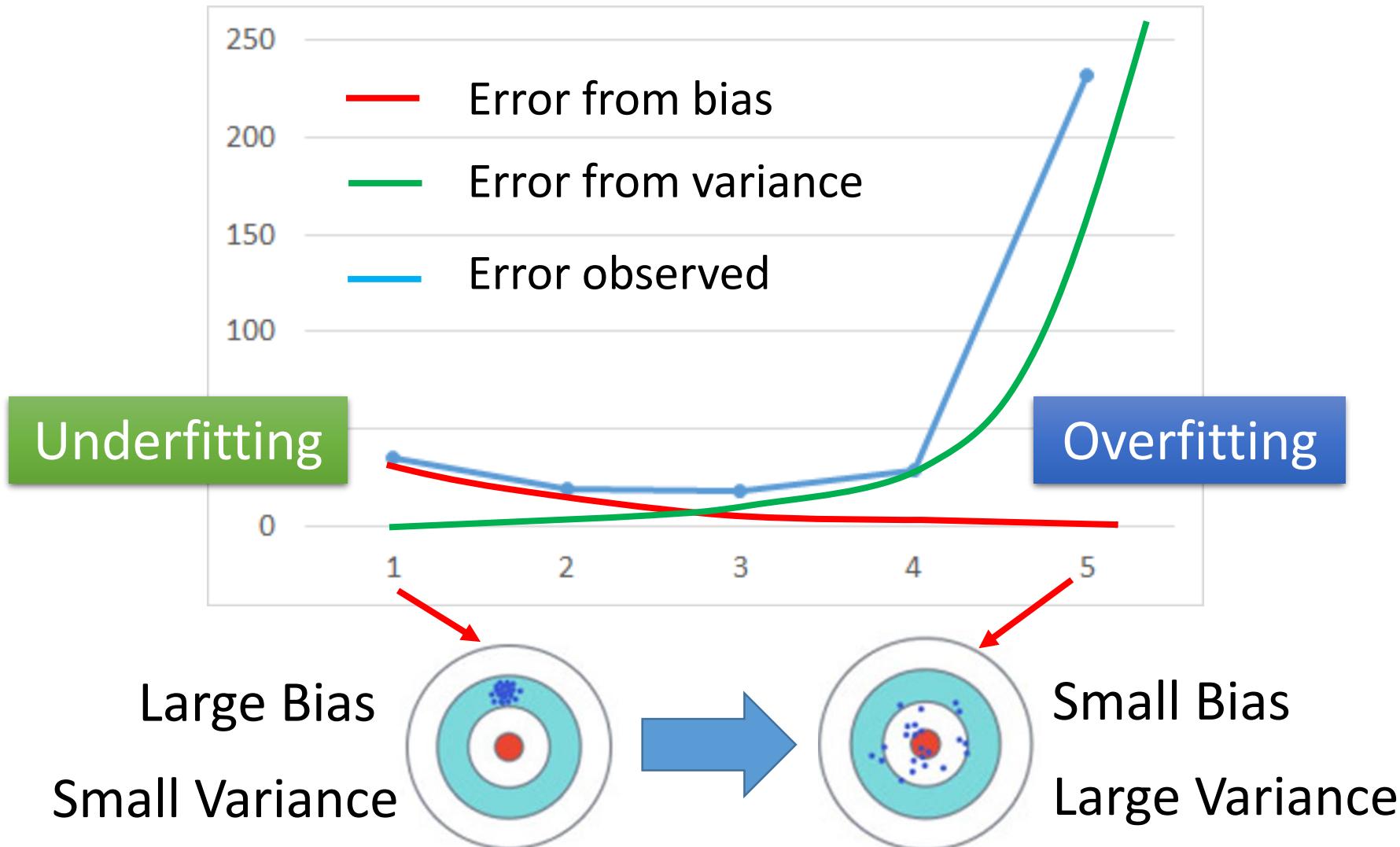
Bias



$$y = b + w_1 \cdot x_{cp} + w_2 \cdot (x_{cp})^2 + w_3 \cdot (x_{cp})^3 + w_4 \cdot (x_{cp})^4 + w_5 \cdot (x_{cp})^5$$

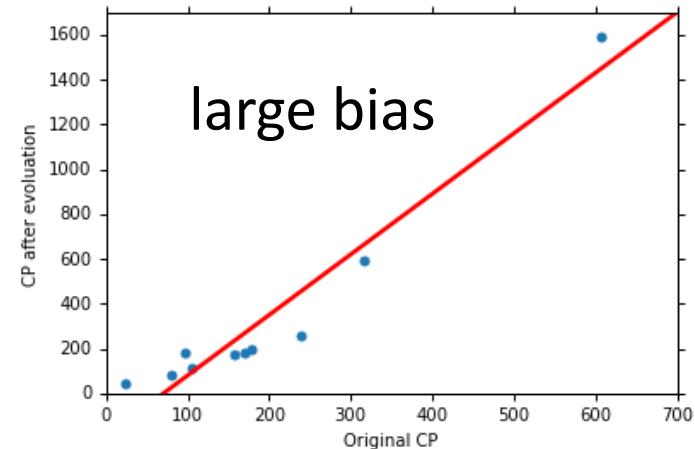


Bias v.s. Variance



What to do with large bias?

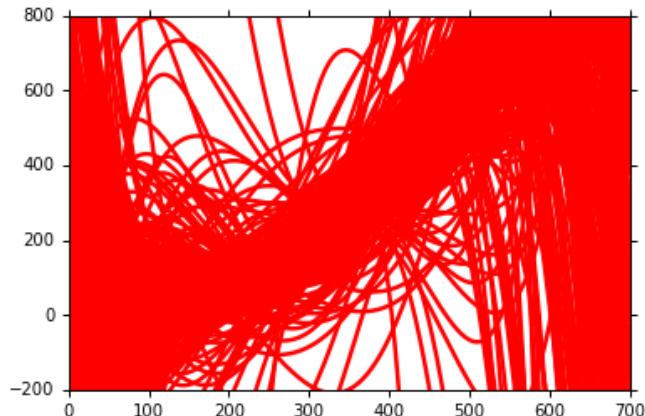
- Diagnosis:
 - If your model cannot even fit the training examples, then you have large bias **Underfitting**
 - If you can fit the training data, but large error on testing data, then you probably have large variance **Overfitting**
- For bias, redesign your model:
 - Add more features as input
 - A more complex model



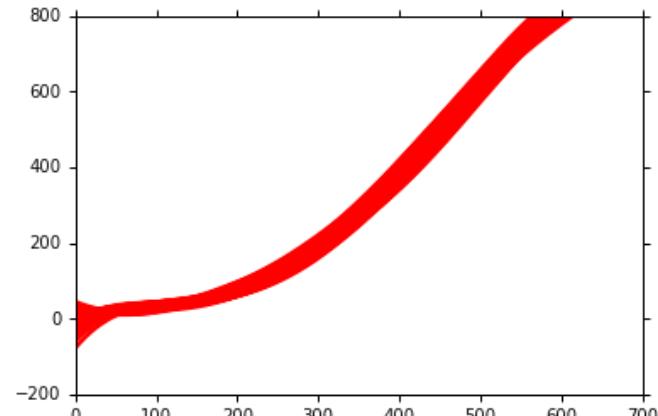
What to do with large variance?

- More data

Very effective,
but not always
practical

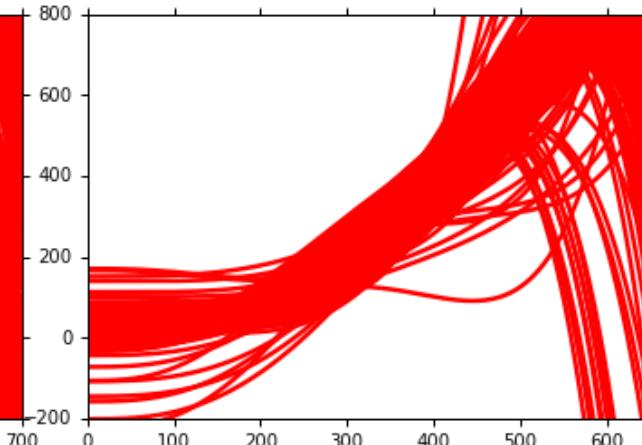
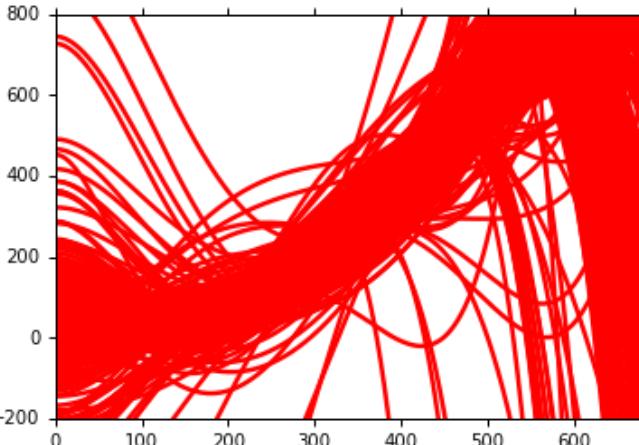
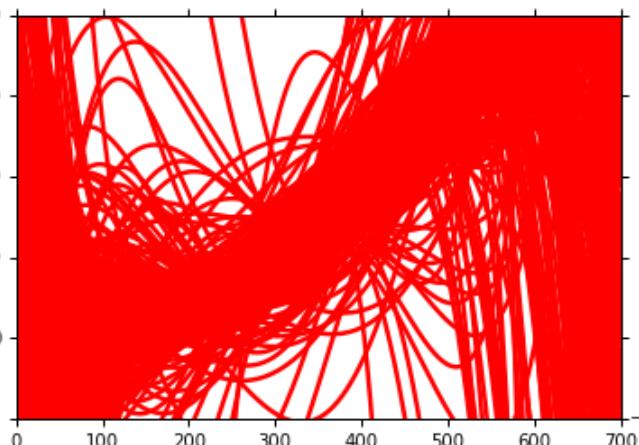


10 examples



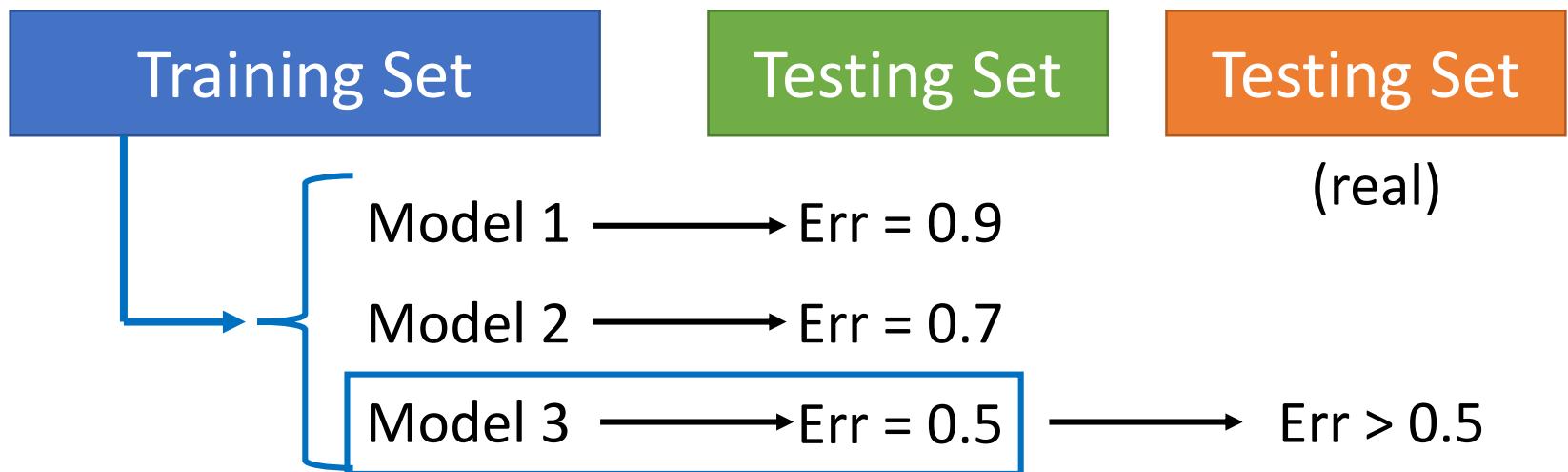
100 examples

- Regularization

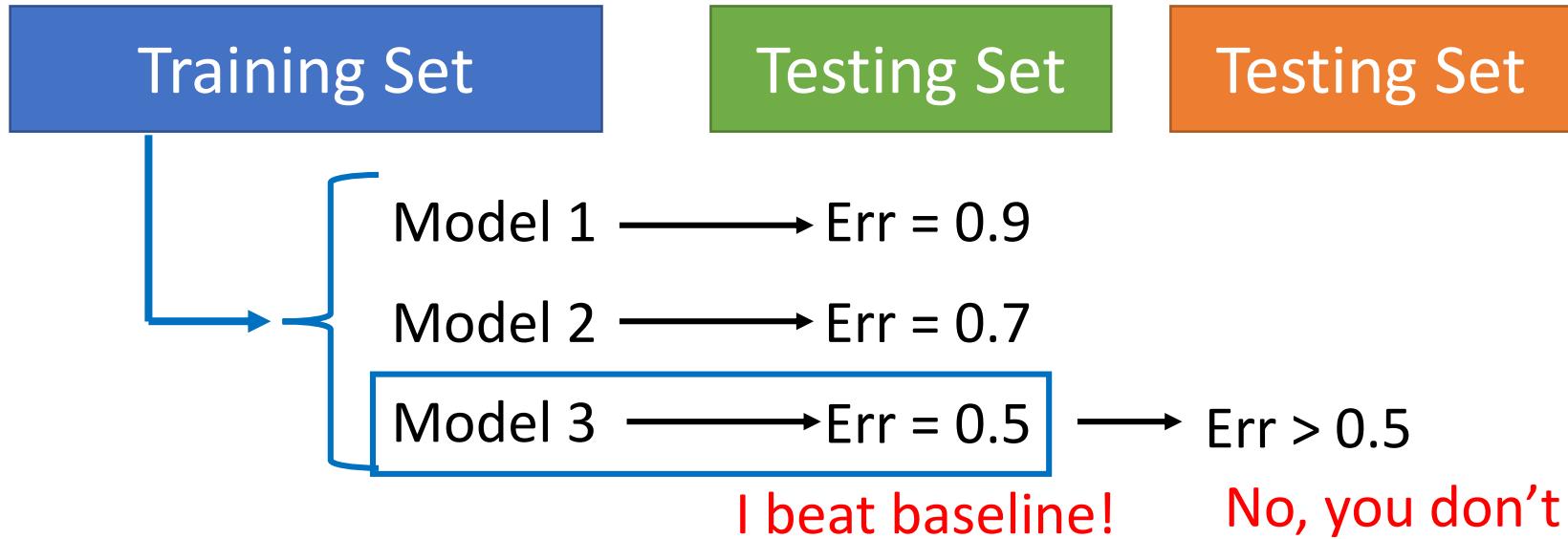


Model Selection

- There is usually a trade-off between bias and variance.
- Select a model that balances two kinds of error to minimize total error
- What you should NOT do:

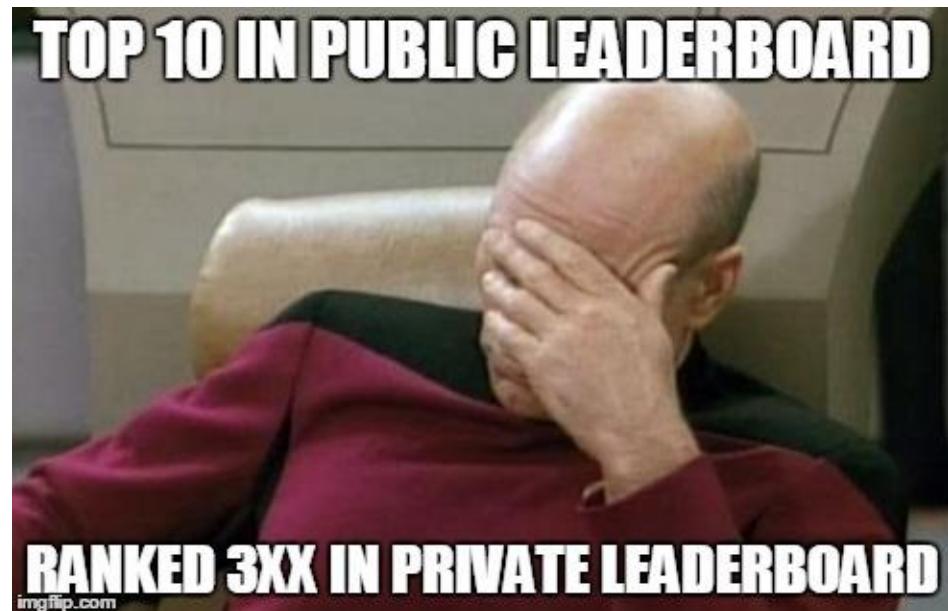


Homework

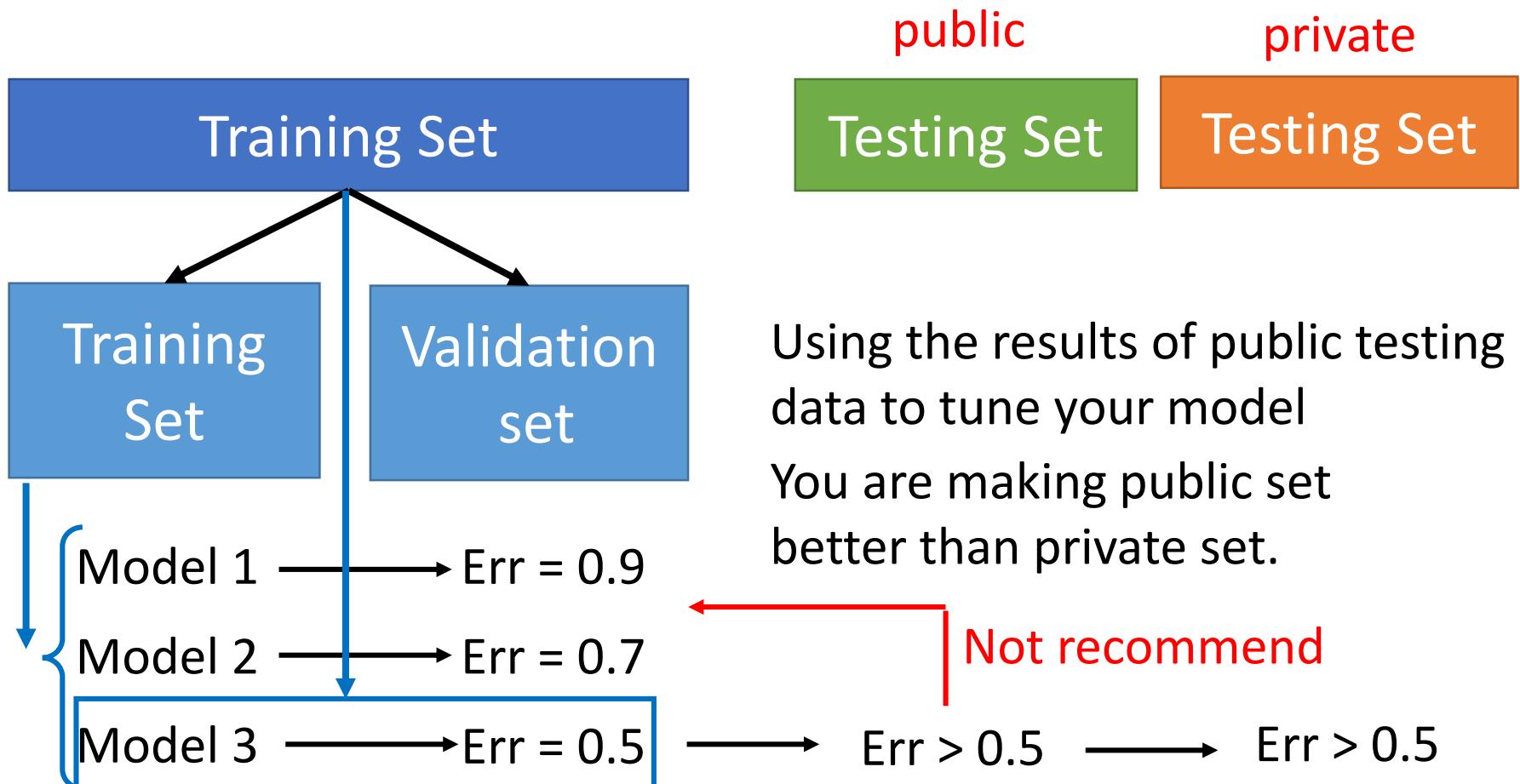


What will happen next Friday?

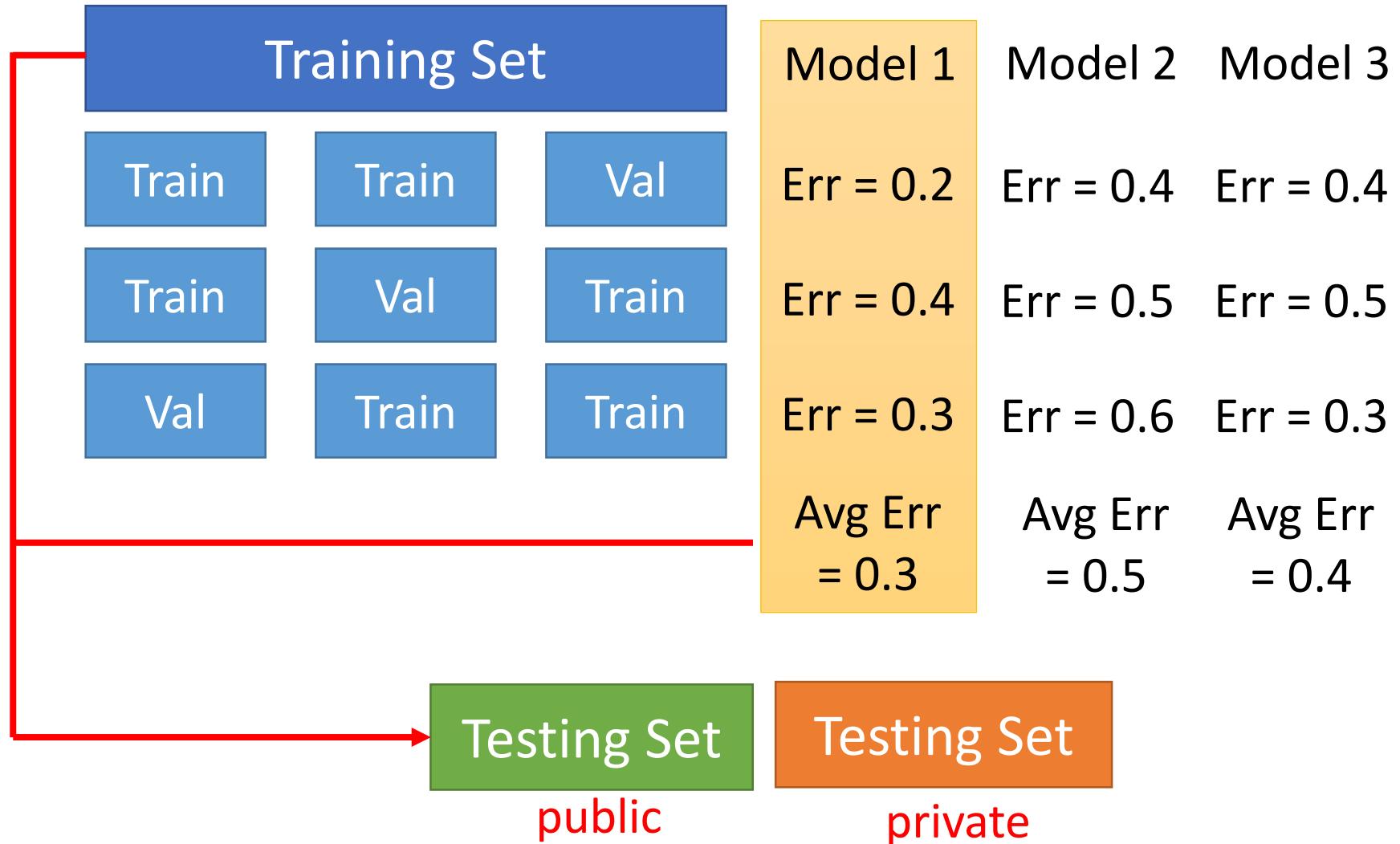
<http://www.chioka.in/how-to-select-your-final-models-in-a-kaggle-competition/>



Cross Validation



N-fold Cross Validation



Reference

- Bishop: Chapter 3.2

Gradient Descent

Review: Gradient Descent

- In step 3, we have to solve the following optimization problem:

$$\theta^* = \arg \min_{\theta} L(\theta) \quad L: \text{loss function} \quad \theta: \text{parameters}$$

Suppose that θ has two variables $\{\theta_1, \theta_2\}$

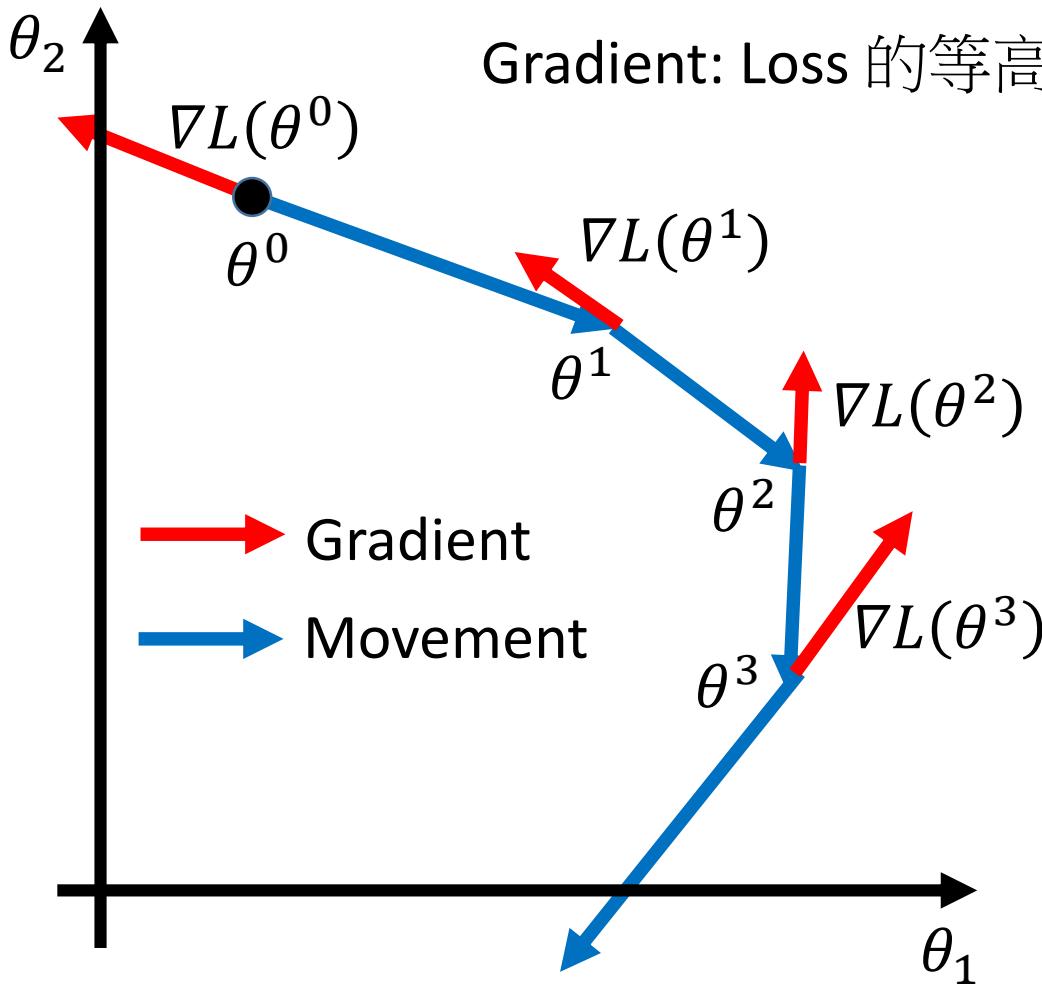
Randomly start at $\theta^0 = \begin{bmatrix} \theta_1^0 \\ \theta_2^0 \end{bmatrix}$

$$\nabla L(\theta) = \begin{bmatrix} \partial L(\theta_1)/\partial \theta_1 \\ \partial L(\theta_2)/\partial \theta_2 \end{bmatrix}$$

$$\begin{bmatrix} \theta_1^1 \\ \theta_2^1 \end{bmatrix} = \begin{bmatrix} \theta_1^0 \\ \theta_2^0 \end{bmatrix} - \eta \begin{bmatrix} \partial L(\theta_1^0)/\partial \theta_1 \\ \partial L(\theta_2^0)/\partial \theta_2 \end{bmatrix} \rightarrow \theta^1 = \theta^0 - \eta \nabla L(\theta^0)$$

$$\begin{bmatrix} \theta_1^2 \\ \theta_2^2 \end{bmatrix} = \begin{bmatrix} \theta_1^1 \\ \theta_2^1 \end{bmatrix} - \eta \begin{bmatrix} \partial L(\theta_1^1)/\partial \theta_1 \\ \partial L(\theta_2^1)/\partial \theta_2 \end{bmatrix} \rightarrow \theta^2 = \theta^1 - \eta \nabla L(\theta^1)$$

Review: Gradient Descent



Gradient: Loss 的等高線的法線方向

Start at position θ^0

Compute gradient at θ^0

Move to $\theta^1 = \theta^0 - \eta \nabla L(\theta^0)$

Compute gradient at θ^1

Move to $\theta^2 = \theta^1 - \eta \nabla L(\theta^1)$

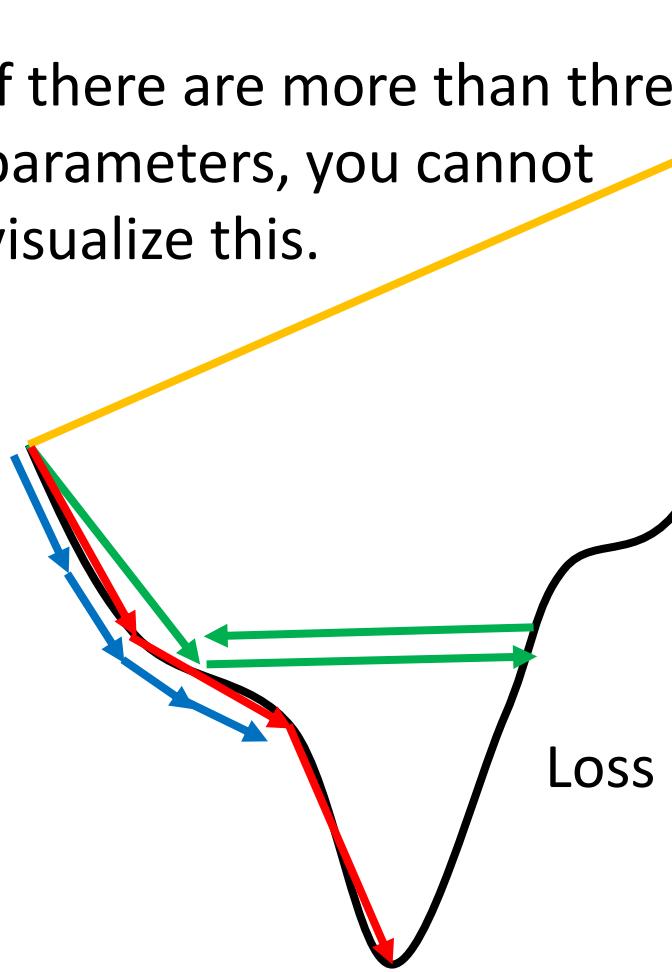
⋮

Gradient Descent

Tip 1: Tuning your
learning rates

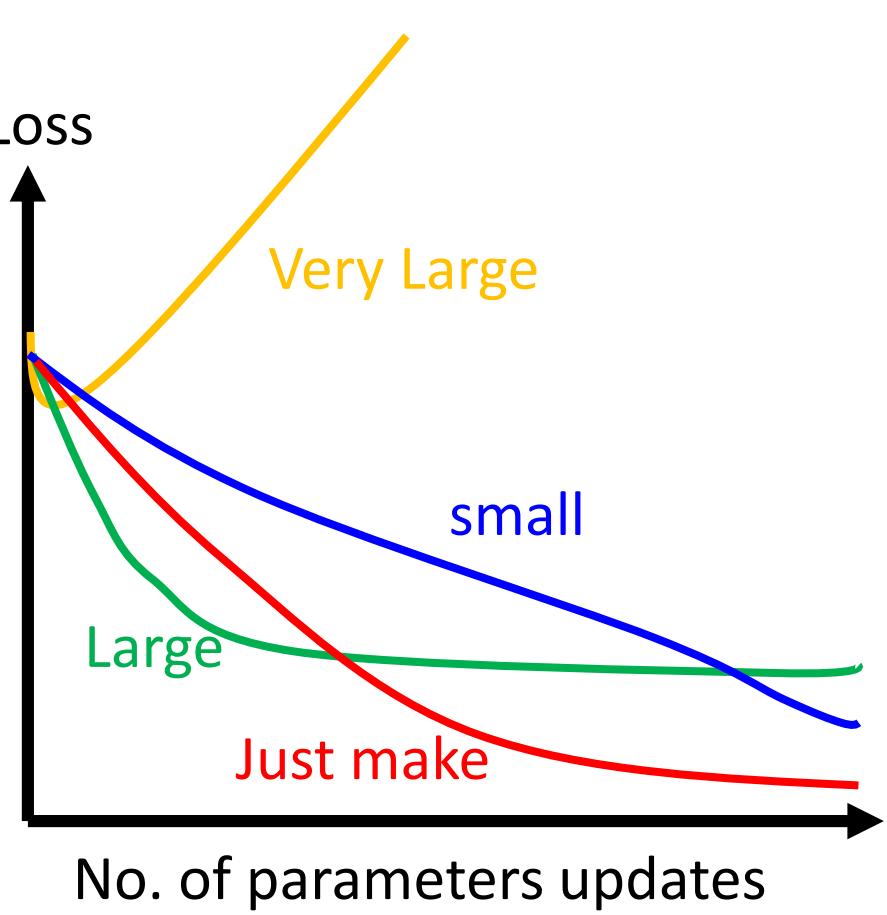
Learning Rate

If there are more than three parameters, you cannot visualize this.



$$\theta^i = \theta^{i-1} - \eta \nabla L(\theta^{i-1})$$

Set the learning rate η carefully



But you can always visualize this.

Adaptive Learning Rates

- Popular & Simple Idea: Reduce the learning rate by some factor every few epochs.
 - At the beginning, we are far from the destination, so we use larger learning rate
 - After several epochs, we are close to the destination, so we reduce the learning rate
 - E.g. 1/t decay: $\eta^t = \eta / \sqrt{t + 1}$
- Learning rate cannot be one-size-fits-all
 - Giving different parameters different learning rates

Adagrad

$$\eta^t = \frac{\eta}{\sqrt{t+1}} \quad g^t = \frac{\partial L(\theta^t)}{\partial w}$$

- Divide the learning rate of each parameter by the ***root mean square of its previous derivatives***

Vanilla Gradient descent

$$w^{t+1} \leftarrow w^t - \eta^t g^t$$

w is one parameters

Adagrad

$$w^{t+1} \leftarrow w^t - \frac{\eta^t}{\sigma^t} g^t$$

σ^t : ***root mean square*** of the previous derivatives of parameter w

Parameter dependent

Adagrad

$$w^1 \leftarrow w^0 - \frac{\eta^0}{\sigma^0} g^0$$

$$w^2 \leftarrow w^1 - \frac{\eta^1}{\sigma^1} g^1$$

$$w^3 \leftarrow w^2 - \frac{\eta^2}{\sigma^2} g^2$$

⋮
⋮

$$w^{t+1} \leftarrow w^t - \frac{\eta^t}{\sigma^t} g^t$$

σ^t : **root mean square** of
the previous derivatives of
parameter w

$$\sigma^0 = \sqrt{(g^0)^2}$$

$$\sigma^1 = \sqrt{\frac{1}{2} [(g^0)^2 + (g^1)^2]}$$

$$\sigma^2 = \sqrt{\frac{1}{3} [(g^0)^2 + (g^1)^2 + (g^2)^2]}$$

$$\sigma^t = \sqrt{\frac{1}{t+1} \sum_{i=0}^t (g^i)^2}$$

Adagrad

- Divide the learning rate of each parameter by the *root mean square of its previous derivatives*

$$w^{t+1} \leftarrow w^t - \frac{\eta^t}{\sigma^t} g^t$$

~~$\eta^t = \frac{\eta}{\sqrt{t + 1}}$ 1/t decay~~

$$\sigma^t = \sqrt{\frac{1}{t + 1} \sum_{i=0}^t (g^i)^2}$$

\downarrow

$$w^{t+1} \leftarrow w^t - \frac{\eta}{\sqrt{\sum_{i=0}^t (g^i)^2}} g^t$$

Contradiction? $\eta^t = \frac{\eta}{\sqrt{t+1}}$ $g^t = \frac{\partial L(\theta^t)}{\partial w}$

Vanilla Gradient descent

$$w^{t+1} \leftarrow w^t - \eta^t g^t$$

Larger gradient,
larger step

Adagrad

$$w^{t+1} \leftarrow w^t - \frac{\eta}{\sqrt{\sum_{i=0}^t (g^i)^2}} g^t$$

Larger gradient,
larger step

Larger gradient,
smaller step

Intuitive Reason

$$\eta^t = \frac{\eta}{\sqrt{t+1}} \quad g^t = \frac{\partial C(\theta^t)}{\partial w}$$

- How surprise it is 反差

特別大

g^0	g^1	g^2	g^3	g^4
0.001	0.001	0.003	0.002	0.1

g^0	g^1	g^2	g^3	g^4
10.8	20.9	31.7	12.1	0.1

特別小

$$w^{t+1} \leftarrow w^t - \frac{\eta}{\sqrt{\sum_{i=0}^t (g^i)^2}} g^t$$

造成反差的效果

Larger gradient, larger steps?

Larger 1st order derivative means far from the minima

$$y = ax^2 + bx + c$$



$$-\frac{b}{2a}$$

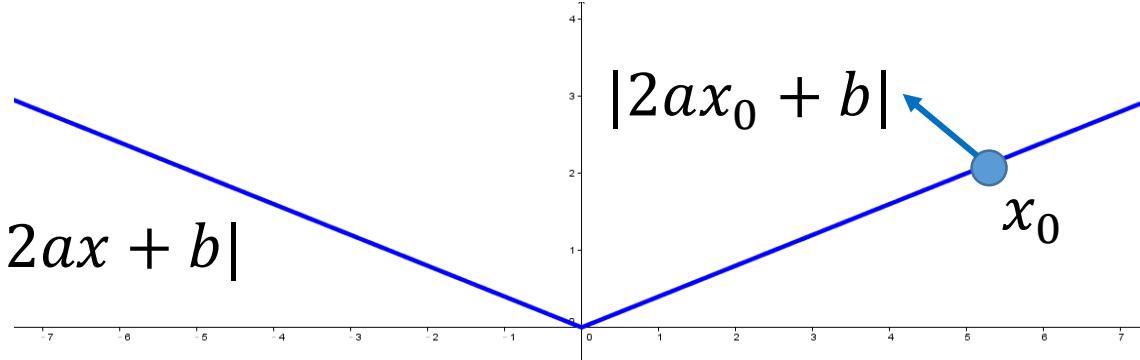
Best step:

$$|x_0 + \frac{b}{2a}|$$

$$\frac{|2ax_0 + b|}{2a}$$

x_0

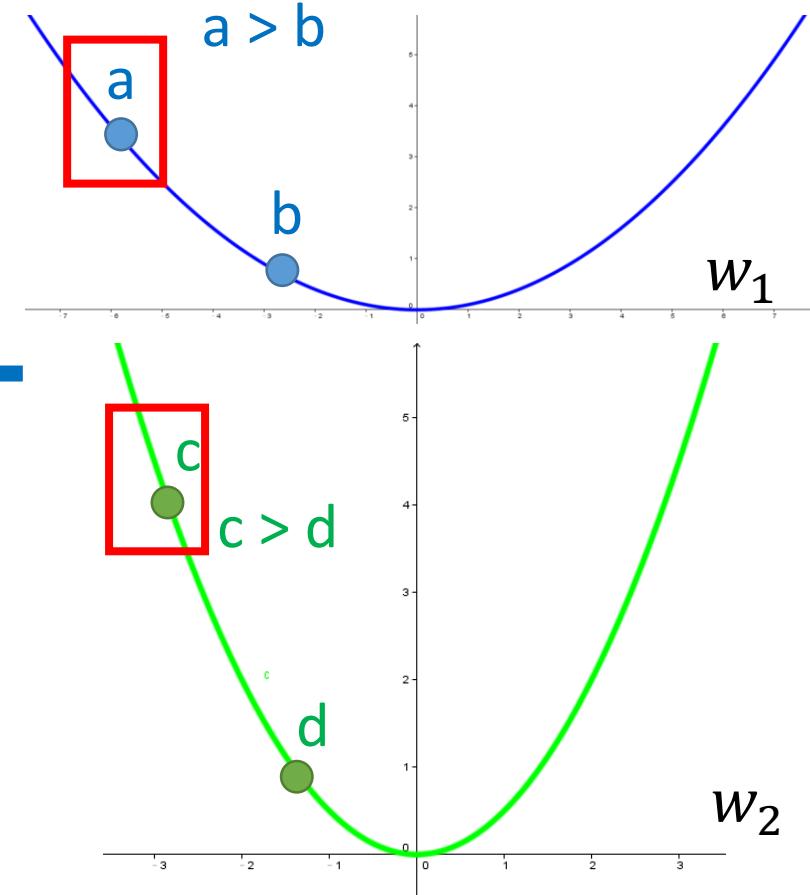
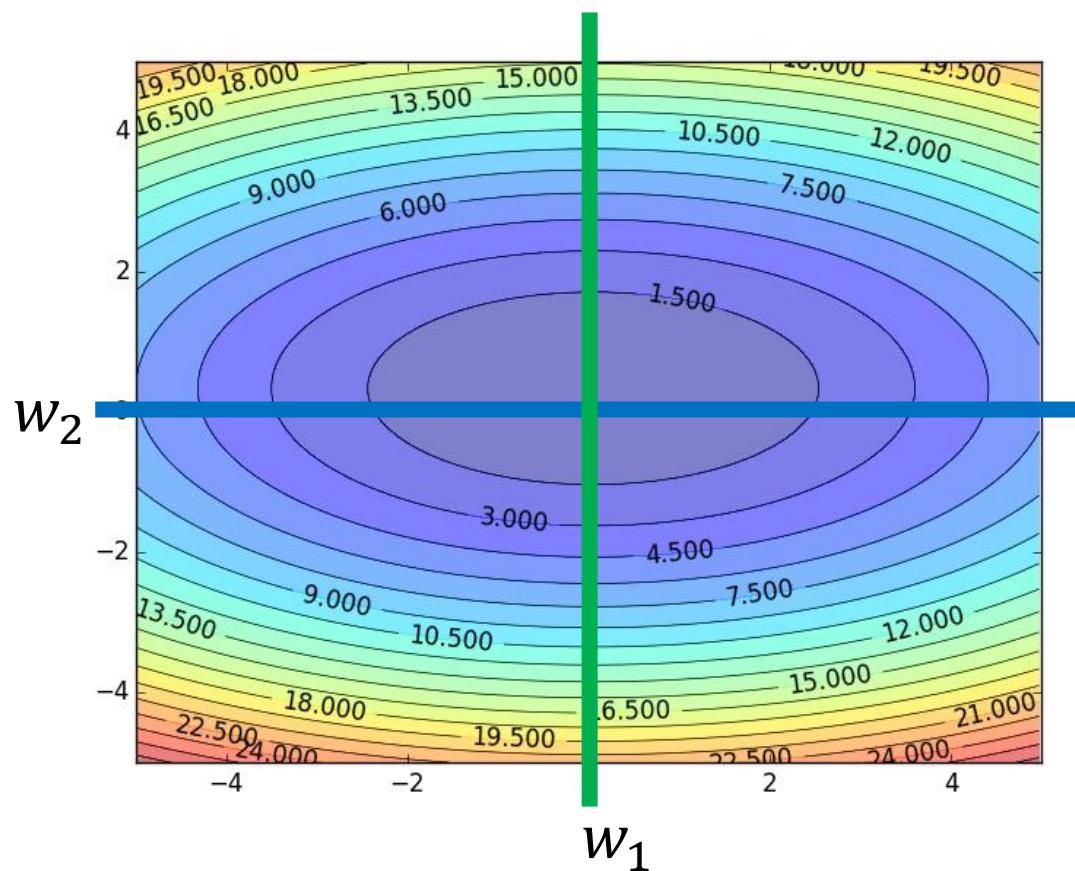
$$\left| \frac{\partial y}{\partial x} \right| = |2ax + b|$$



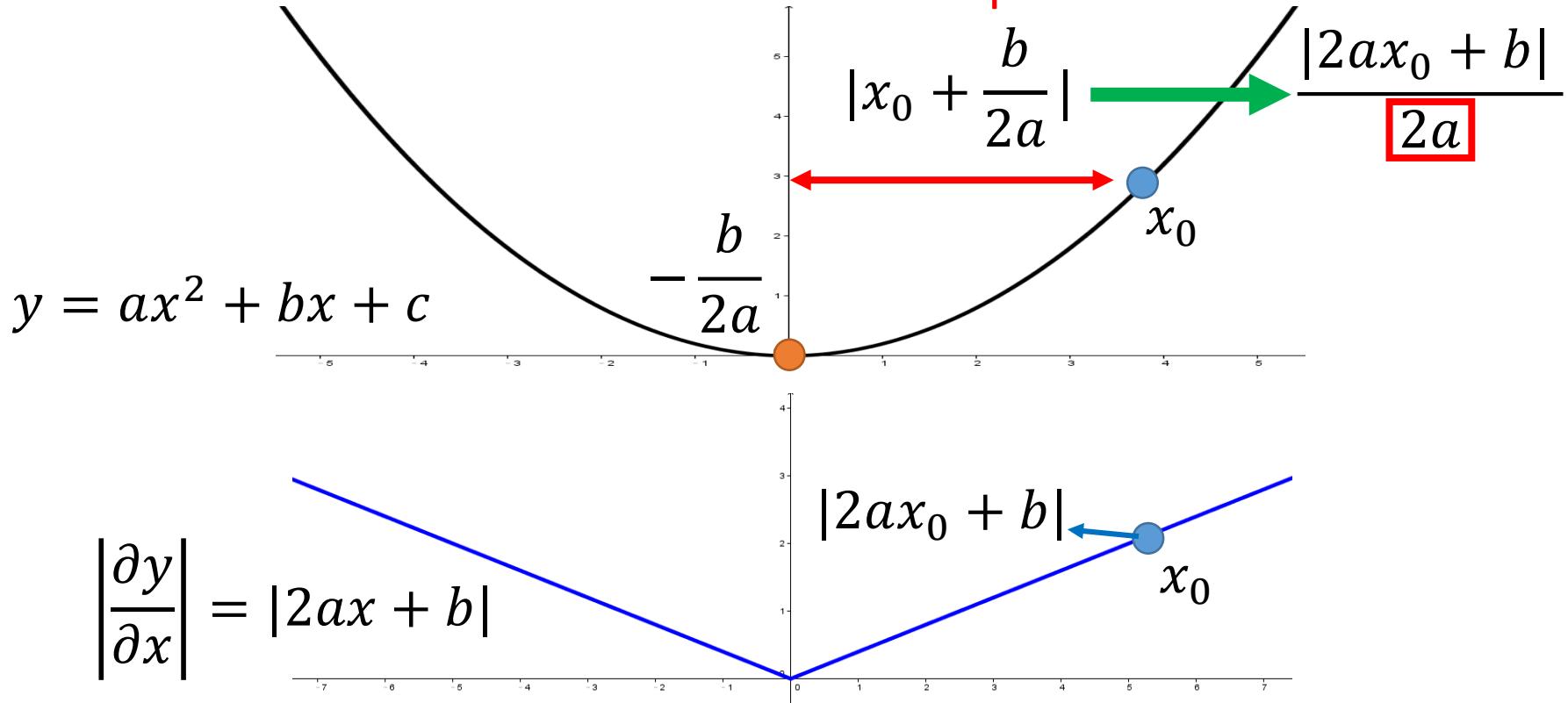
Comparison between different parameters

Larger 1st order derivative means far from the minima

Do not cross parameters



Second Derivative



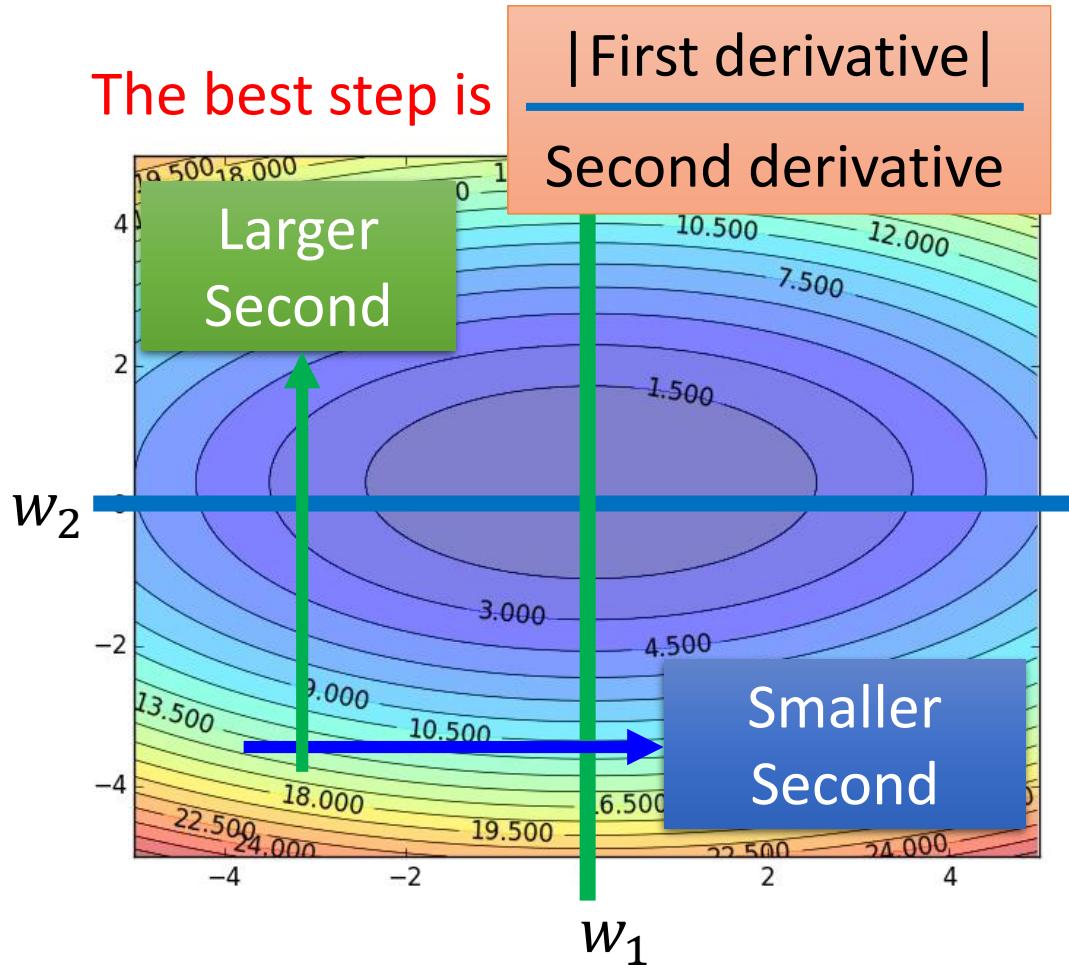
$$\frac{\partial^2 y}{\partial x^2} = 2a$$

The best step is

$$\frac{|\text{First derivative}|}{\text{Second derivative}}$$

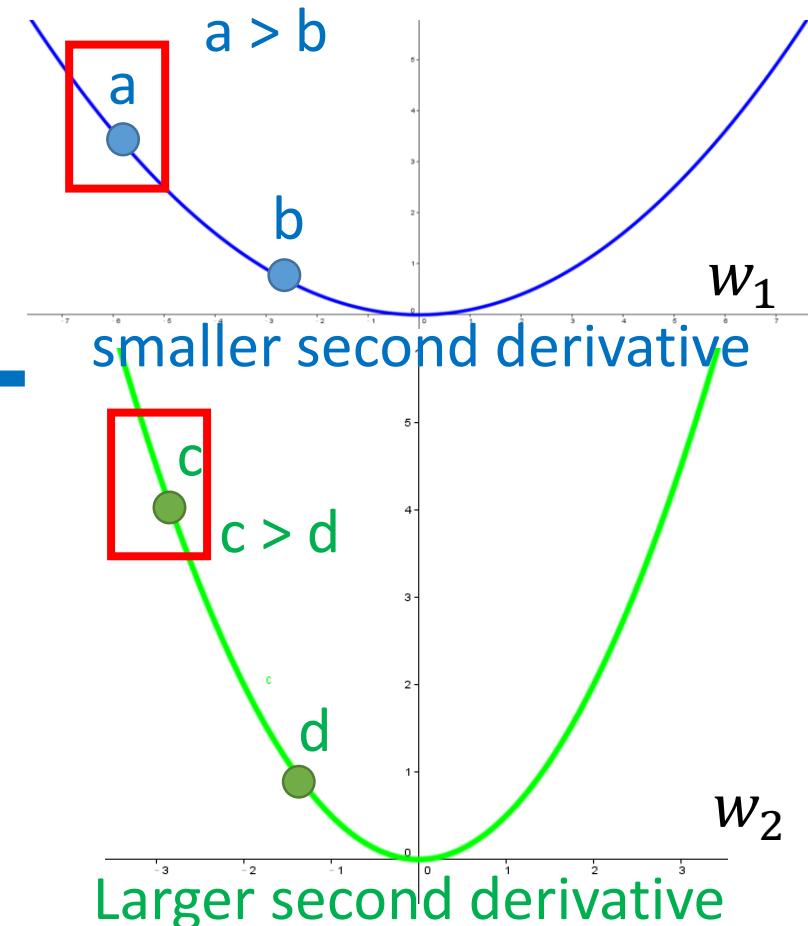
Comparison between different parameters

The best step is



~~Larger 1st order derivative means far from the minima~~

Do not cross parameters

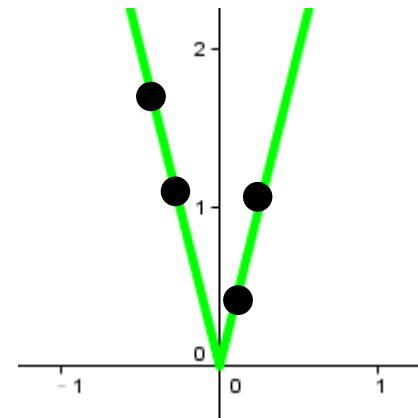
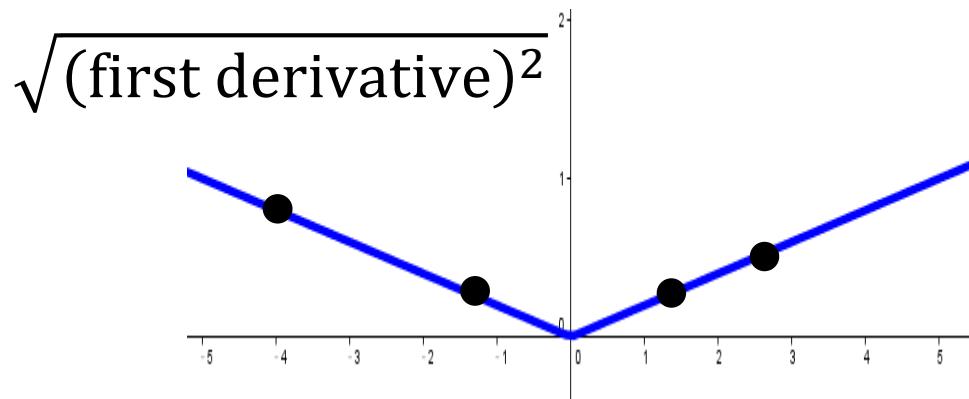
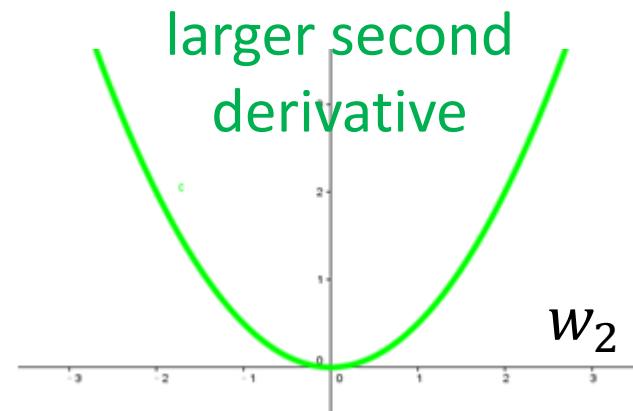
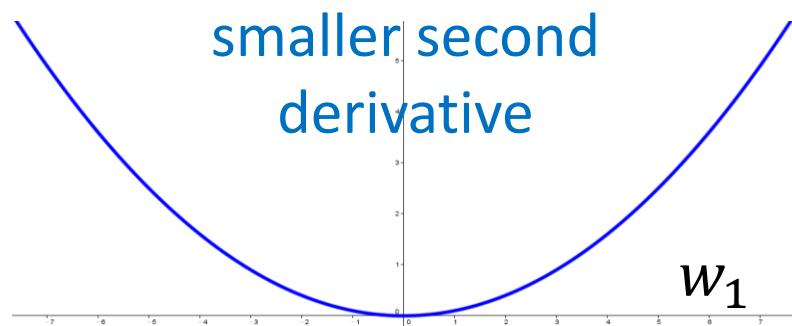


$$w^{t+1} \leftarrow w^t - \frac{\eta}{\sqrt{\sum_{i=0}^t (g^i)^2}} g^t$$

The best step is
 |First derivative|

 Second derivative
 ?

Use first derivative to estimate second derivative



Gradient Descent

Tip 2: Stochastic Gradient Descent

Make the training faster

Stochastic Gradient Descent

$$L = \sum_n \left(\hat{y}^n - \left(b + \sum w_i x_i^n \right) \right)^2$$

Loss is the summation over all training examples

◆ **Gradient Descent** $\theta^i = \theta^{i-1} - \eta \nabla L(\theta^{i-1})$

◆ **Stochastic Gradient Descent**

Faster!

Pick an example x^n

$$L^n = \left(\hat{y}^n - \left(b + \sum w_i x_i^n \right) \right)^2$$

Loss for only one example

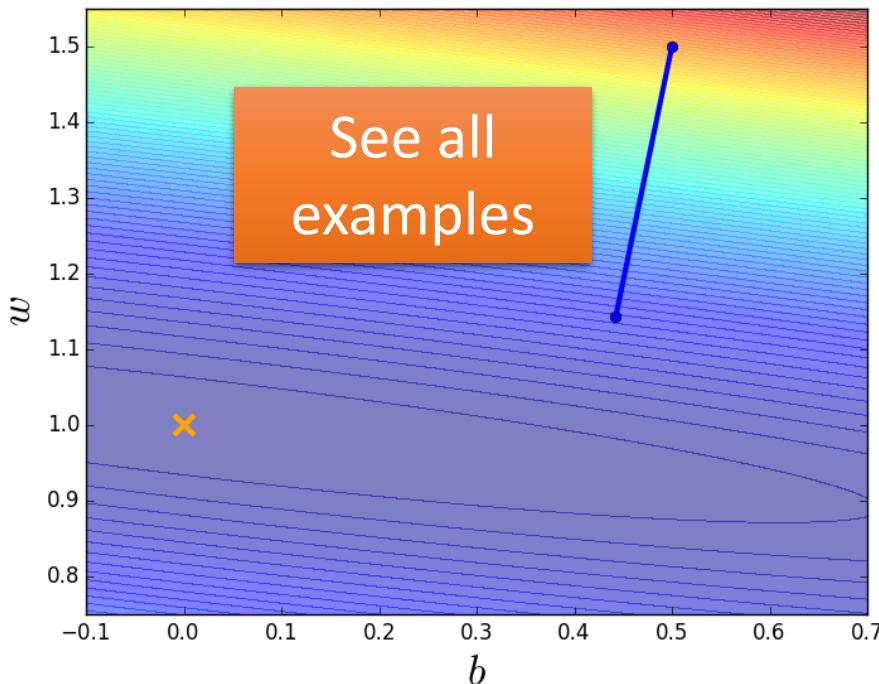
$$\theta^i = \theta^{i-1} - \eta \nabla L^n(\theta^{i-1})$$

- Demo

Stochastic Gradient Descent

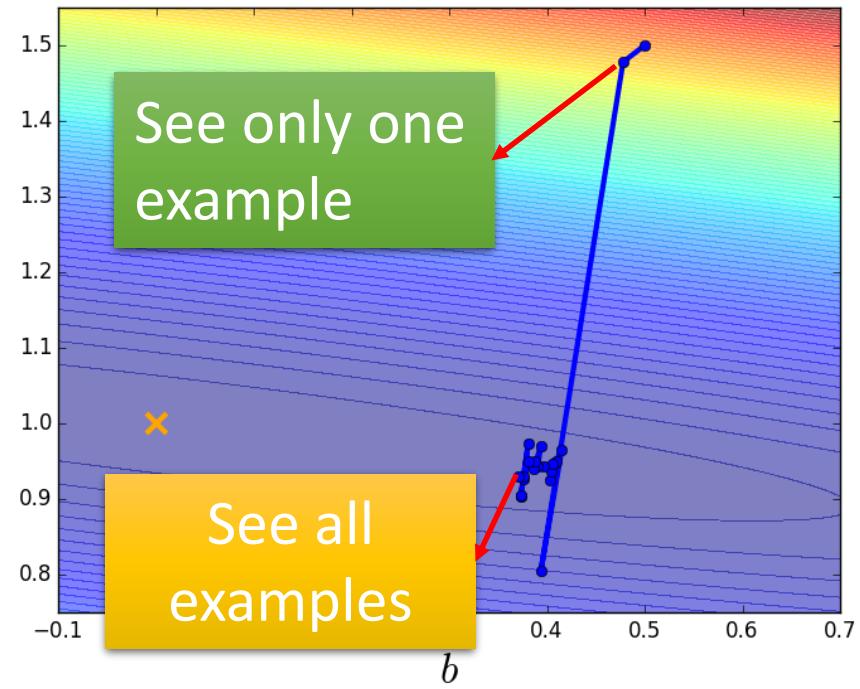
Gradient Descent

Update after seeing all examples



Stochastic Gradient Descent

Update for each example
If there are 20 examples,
20 times faster.



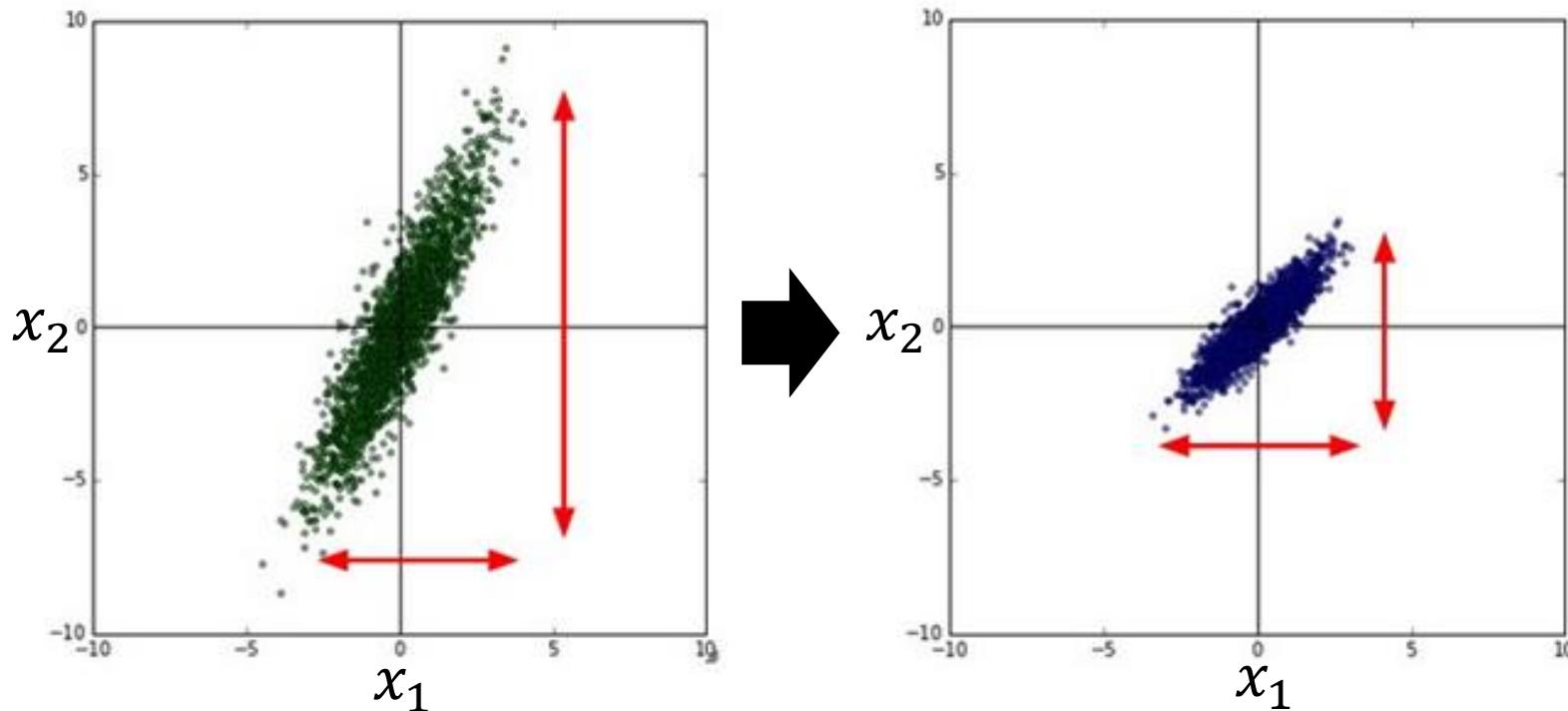
Gradient Descent

Tip 3: Feature Scaling

Feature Scaling

Source of figure:
<http://cs231n.github.io/neural-networks-2/>

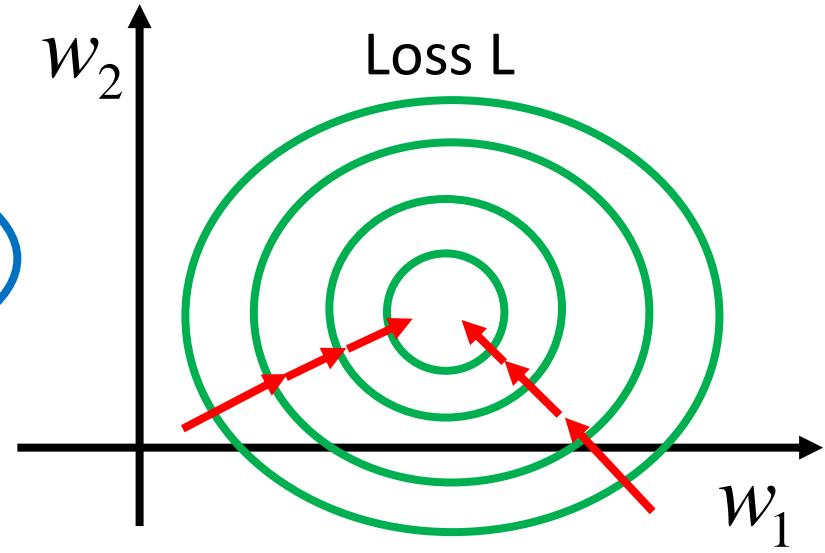
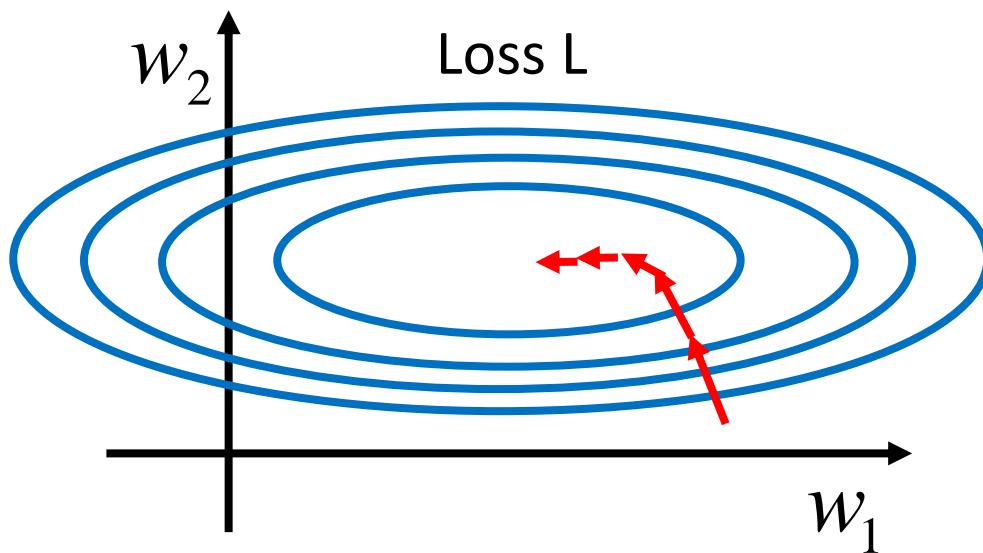
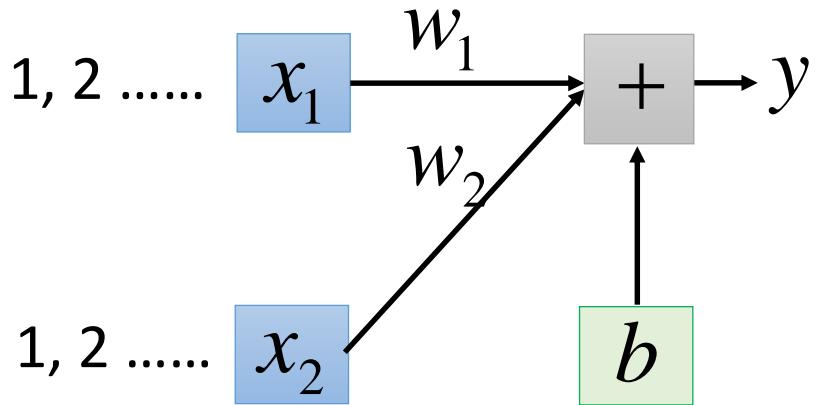
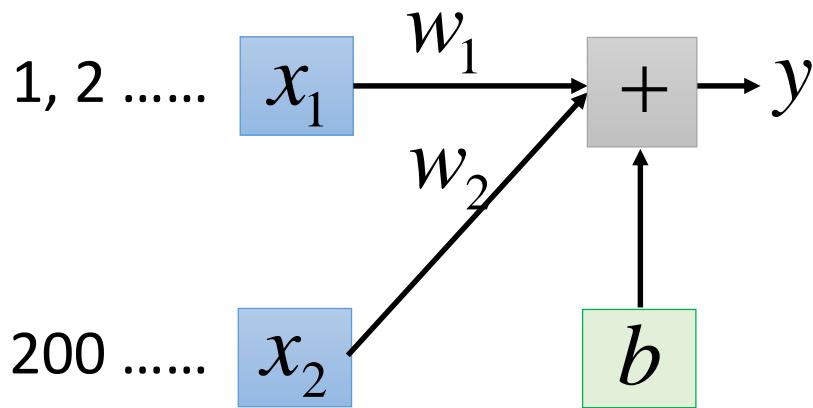
$$y = b + w_1 x_1 + w_2 x_2$$



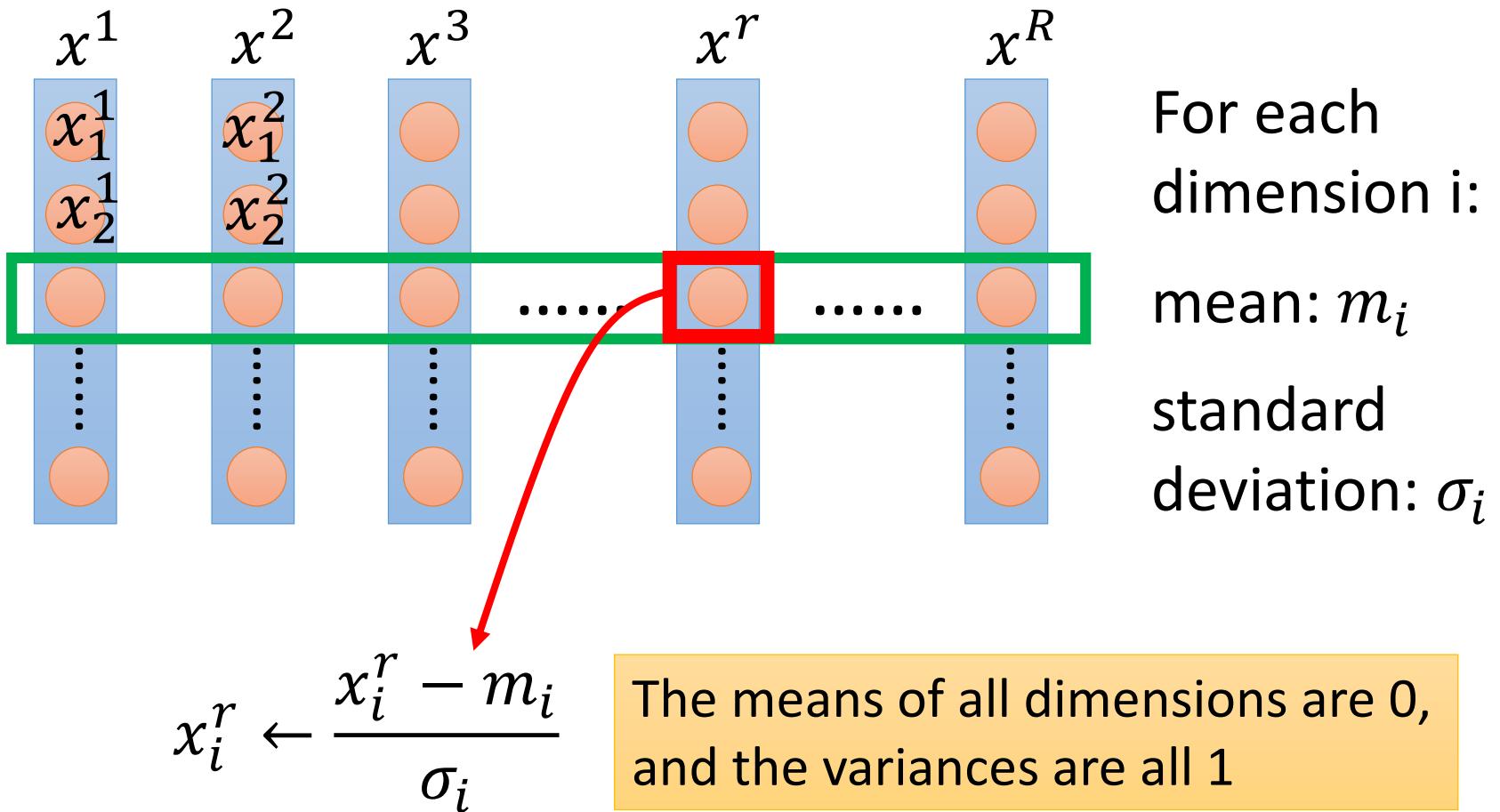
Make different features have the same scaling

Feature Scaling

$$y = b + w_1x_1 + w_2x_2$$



Feature Scaling



Gradient Descent Theory

Question

- When solving:

$$\theta^* = \arg \min_{\theta} L(\theta) \quad \text{by gradient descent}$$

- Each time we update the parameters, we obtain θ that makes $L(\theta)$ smaller.

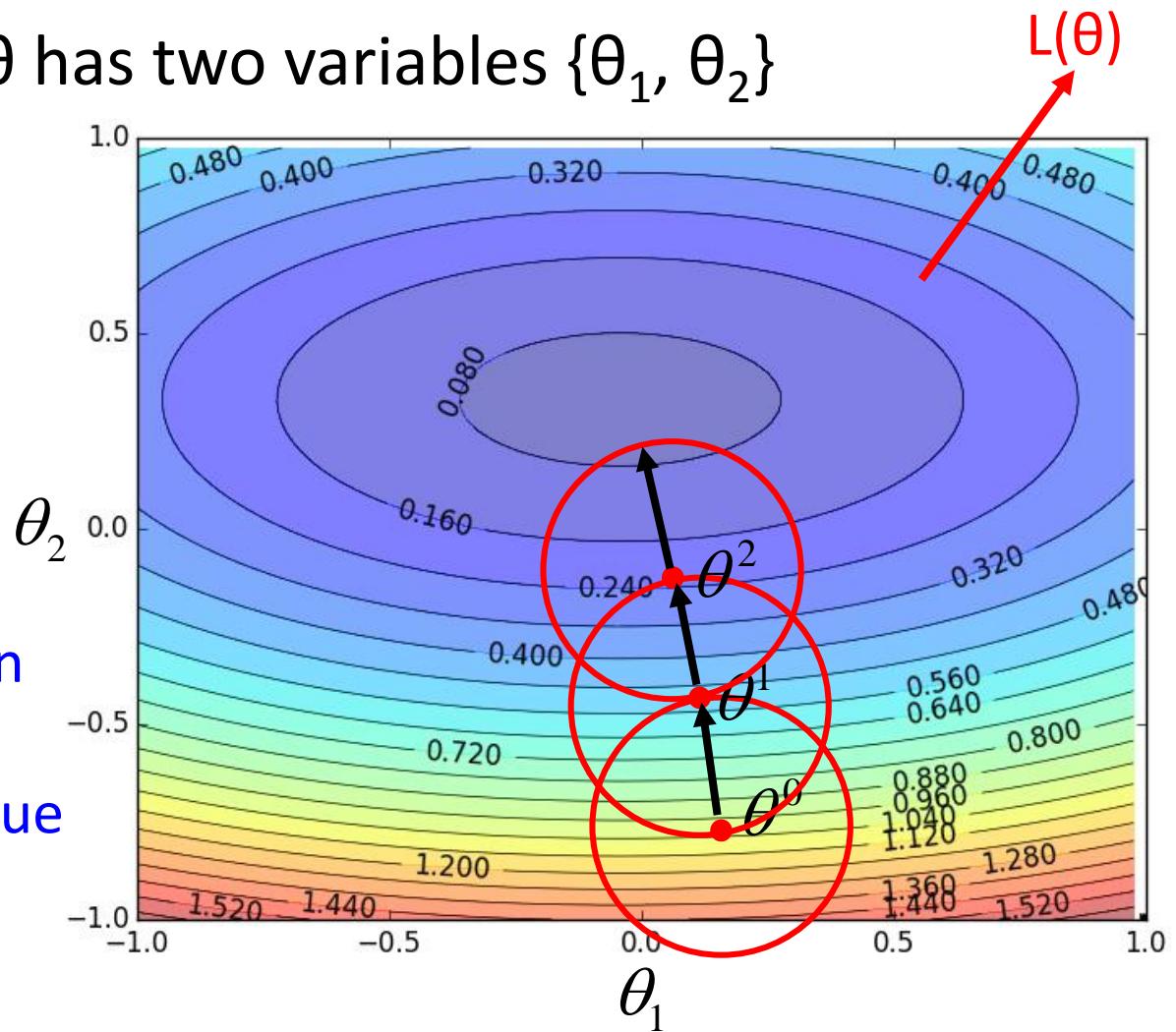
$$L(\theta^0) > L(\theta^1) > L(\theta^2) > \dots$$

Is this statement correct?

Warning of Math

Formal Derivation

- Suppose that θ has two variables $\{\theta_1, \theta_2\}$



Given a point, we can easily find the point with the smallest value nearby. How?

Taylor Series

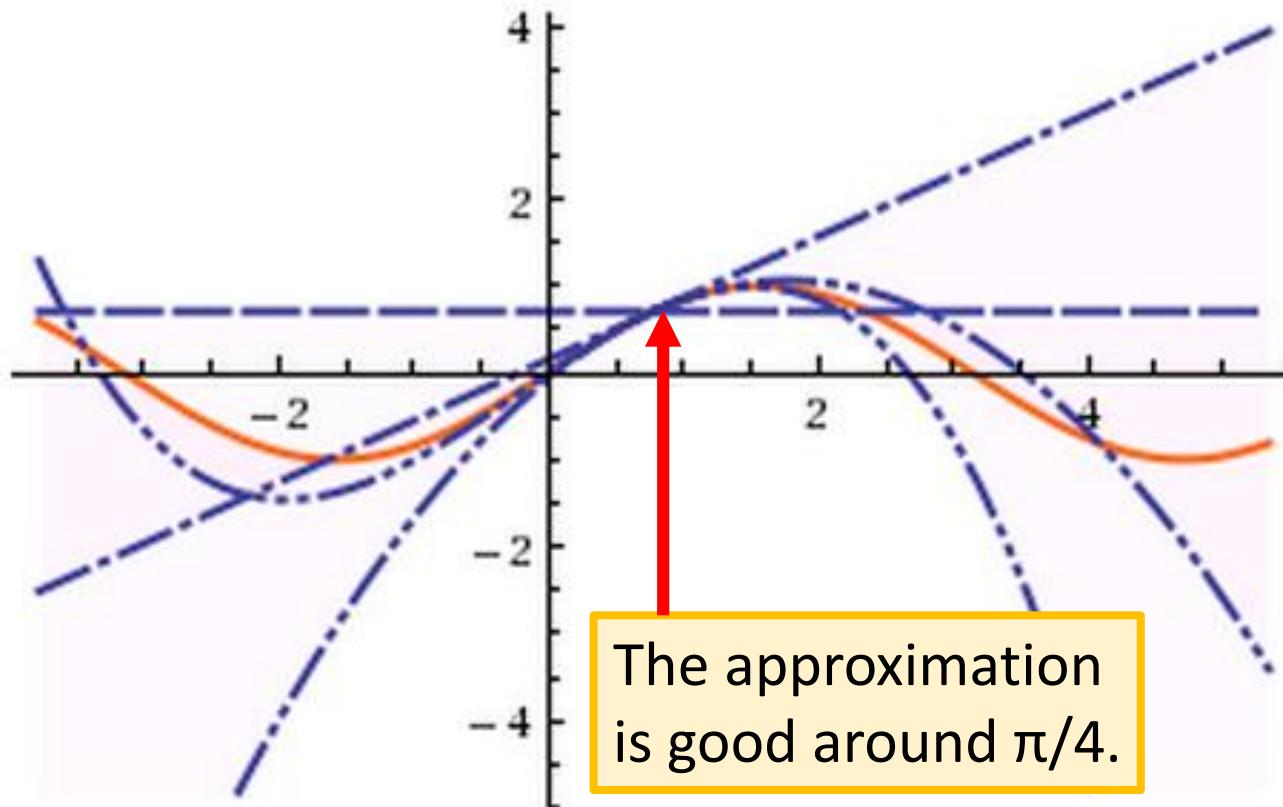
- **Taylor series:** Let $h(x)$ be any function infinitely differentiable around $x = x_0$.

$$\begin{aligned} h(x) &= \sum_{k=0}^{\infty} \frac{h^{(k)}(x_0)}{k!} (x - x_0)^k \\ &= h(x_0) + h'(x_0)(x - x_0) + \frac{h''(x_0)}{2!} (x - x_0)^2 + \dots \end{aligned}$$

When x is close to x_0  $h(x) \approx h(x_0) + h'(x_0)(x - x_0)$

E.g. Taylor series for $h(x)=\sin(x)$ around $x_0=\pi/4$

$$\begin{aligned}\sin(x) = & \frac{1}{\sqrt{2}} + \frac{x - \frac{\pi}{4}}{\sqrt{2}} - \frac{\left(x - \frac{\pi}{4}\right)^2}{2\sqrt{2}} - \frac{\left(x - \frac{\pi}{4}\right)^3}{6\sqrt{2}} + \frac{\left(x - \frac{\pi}{4}\right)^4}{24\sqrt{2}} + \frac{\left(x - \frac{\pi}{4}\right)^5}{120\sqrt{2}} - \frac{\left(x - \frac{\pi}{4}\right)^6}{720\sqrt{2}} - \\& \frac{\left(x - \frac{\pi}{4}\right)^7}{5040\sqrt{2}} + \frac{\left(x - \frac{\pi}{4}\right)^8}{40320\sqrt{2}} + \frac{\left(x - \frac{\pi}{4}\right)^9}{362880\sqrt{2}} - \frac{\left(x - \frac{\pi}{4}\right)^{10}}{3628800\sqrt{2}} + \dots\end{aligned}$$



Multivariable Taylor Series

$$h(x, y) = h(x_0, y_0) + \frac{\partial h(x_0, y_0)}{\partial x} (x - x_0) + \frac{\partial h(x_0, y_0)}{\partial y} (y - y_0)$$

+ something related to $(x-x_0)^2$ and $(y-y_0)^2 + \dots$

When x and y is close to x_0 and y_0



$$h(x, y) \approx h(x_0, y_0) + \frac{\partial h(x_0, y_0)}{\partial x} (x - x_0) + \frac{\partial h(x_0, y_0)}{\partial y} (y - y_0)$$

Back to Formal Derivation

Based on Taylor Series:

If the red circle is small enough, in the red circle

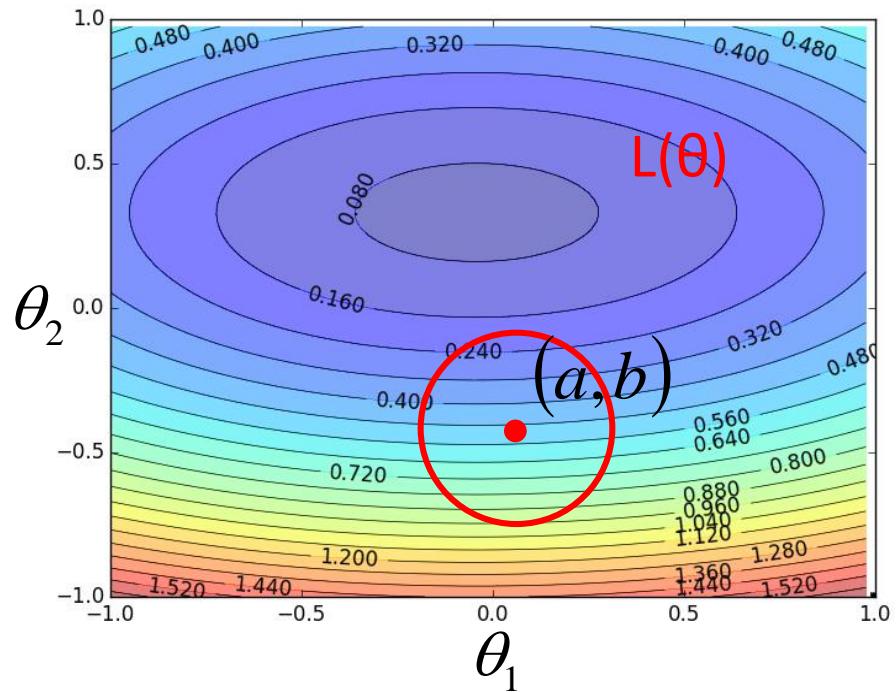
$$L(\theta) \approx L(a, b) + \frac{\partial L(a, b)}{\partial \theta_1} (\theta_1 - a) + \frac{\partial L(a, b)}{\partial \theta_2} (\theta_2 - b)$$

$$s = L(a, b)$$

$$u = \frac{\partial L(a, b)}{\partial \theta_1}, v = \frac{\partial L(a, b)}{\partial \theta_2}$$

$$L(\theta)$$

$$\approx s + u(\theta_1 - a) + v(\theta_2 - b)$$



Back to Formal Derivation

Based on Taylor Series:

If the red circle is small enough, in the red circle

$$L(\theta) \approx s + u(\theta_1 - a) + v(\theta_2 - b)$$

Find θ_1 and θ_2 in the red circle
minimizing $L(\theta)$

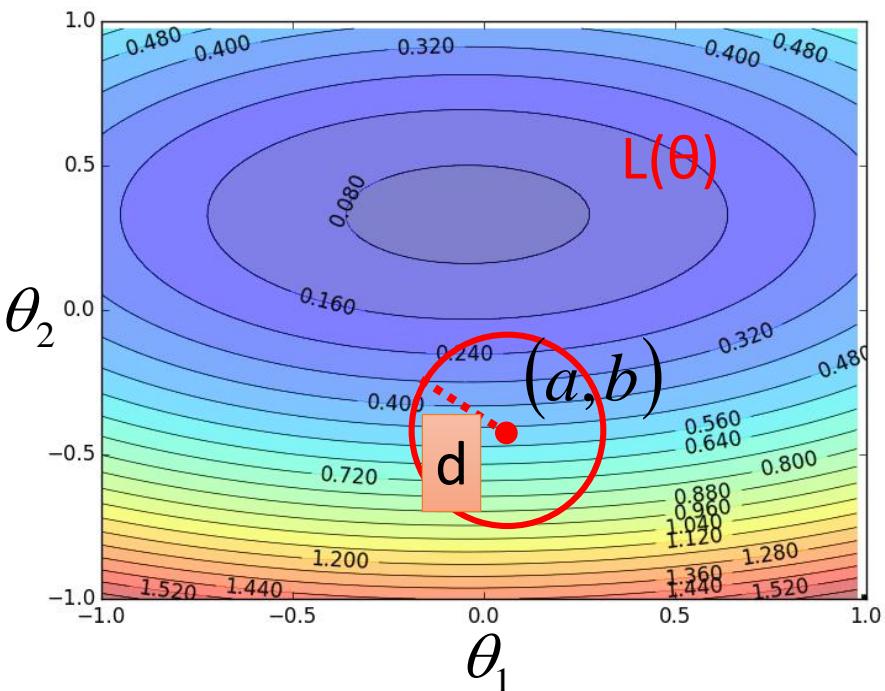
$$(\theta_1 - a)^2 + (\theta_2 - b)^2 \leq d^2$$

Simple, right?

constant

$$s = L(a, b)$$

$$u = \frac{\partial L(a, b)}{\partial \theta_1}, v = \frac{\partial L(a, b)}{\partial \theta_2}$$



Gradient descent – two variables

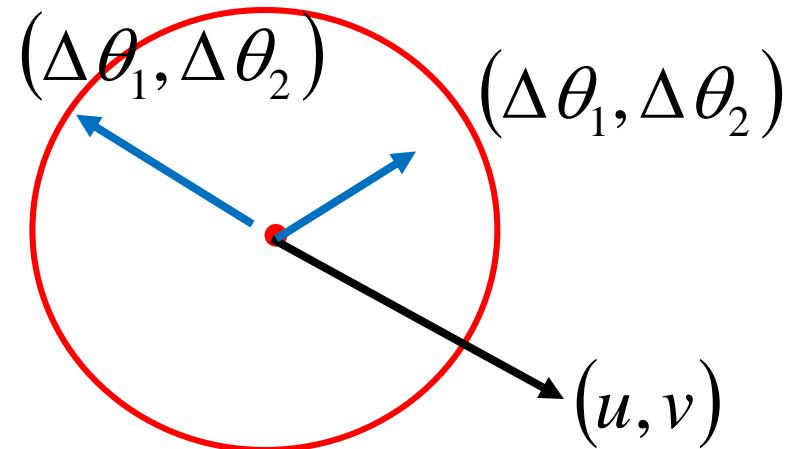
Red Circle: (If the radius is small)

$$L(\theta) \approx s + u \frac{\theta_1 - a}{\Delta\theta_1} + v \frac{\theta_2 - b}{\Delta\theta_2}$$

Find θ_1 and θ_2 in the red circle
minimizing $L(\theta)$

$$\frac{(\theta_1 - a)^2}{\Delta\theta_1} + \frac{(\theta_2 - b)^2}{\Delta\theta_2} \leq d^2$$

To minimize $L(\theta)$



$$\begin{bmatrix} \Delta\theta_1 \\ \Delta\theta_2 \end{bmatrix} = -\eta \begin{bmatrix} u \\ v \end{bmatrix} \quad \rightarrow \quad \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix} = \begin{bmatrix} a \\ b \end{bmatrix} - \eta \begin{bmatrix} u \\ v \end{bmatrix}$$

Back to Formal Derivation

Based on Taylor Series:

If the red circle is small enough, in the red circle

$$L(\theta) \approx s + u(\theta_1 - a) + v(\theta_2 - b)$$

constant

$$s = L(a, b)$$

$$u = \frac{\partial L(a, b)}{\partial \theta_1}, v = \frac{\partial L(a, b)}{\partial \theta_2}$$

Find θ_1 and θ_2 yielding the smallest value of $L(\theta)$ in the circle

$$\begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix} = \begin{bmatrix} a \\ b \end{bmatrix} - \eta \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} a \\ b \end{bmatrix} - \eta \begin{bmatrix} \frac{\partial L(a, b)}{\partial \theta_1} \\ \frac{\partial L(a, b)}{\partial \theta_2} \end{bmatrix}$$

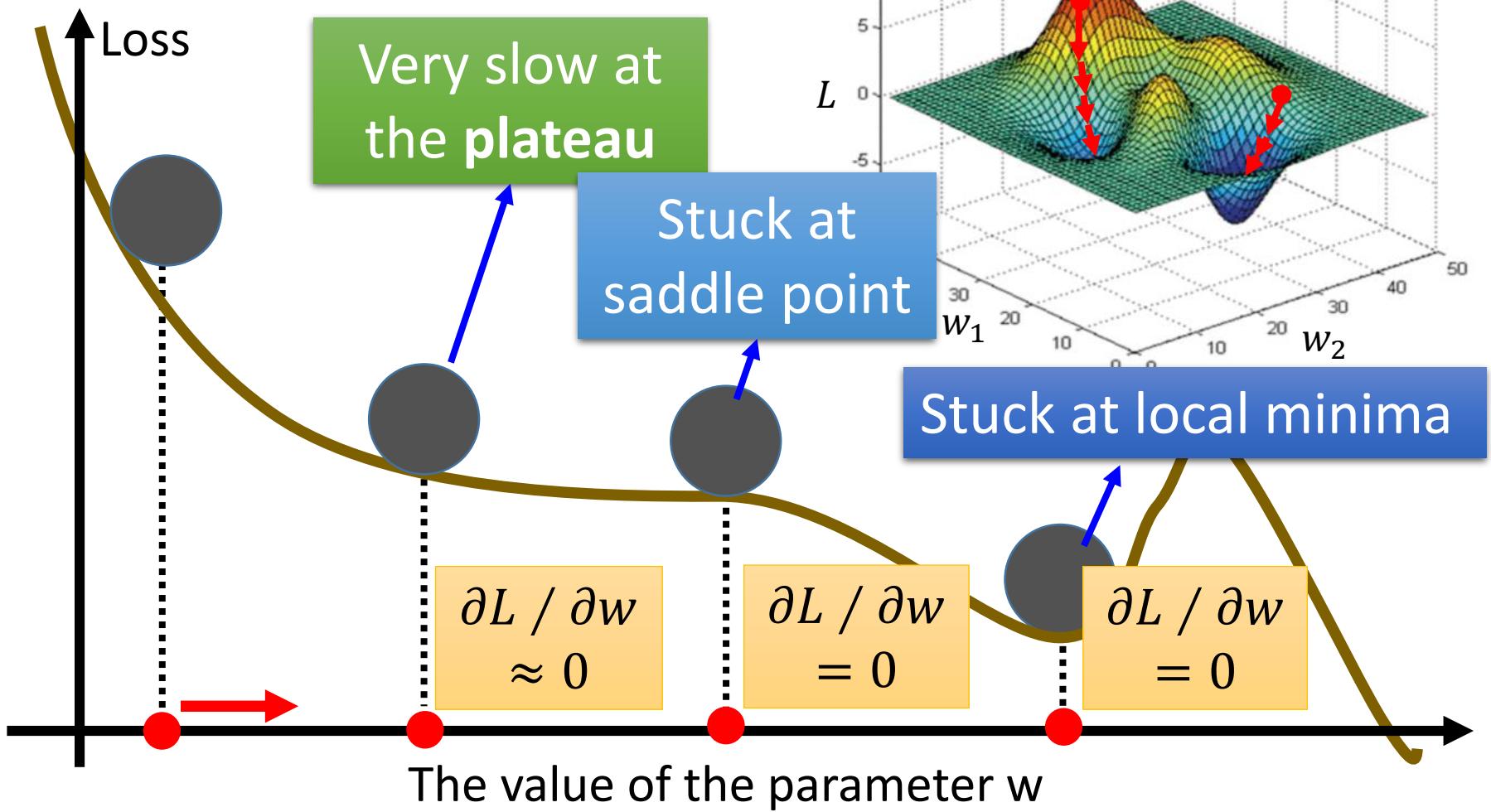
This is gradient descent.

Not satisfied if the red circle (learning rate) is not small enough

You can consider the second order term, e.g. Newton's method.

End of Warning

More Limitation of Gradient Descent



Acknowledgement

- 感謝 Victor Chen 發現投影片上的打字錯誤

Classification: Probabilistic Generative Model

Classification



- Credit Scoring
 - Input: income, savings, profession, age, past financial history
 - Output: accept or refuse
- Medical Diagnosis
 - Input: current symptoms, age, gender, past medical history
 - Output: which kind of diseases
- Handwritten character recognition
- Face recognition
 - Input: image of a face, output: person



Example Application

POKÉMON TYPE SYMBOLS



NORMAL



FIRE



WATER



ELECTRIC



GRASS



ICE



FIGHTING



POISON



GROUND



FLYING



PSYCHIC



BUG



ROCK



GHOST



DRAGON



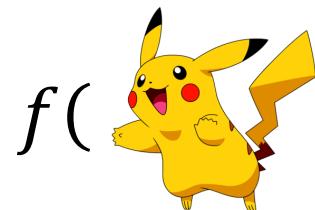
DARK



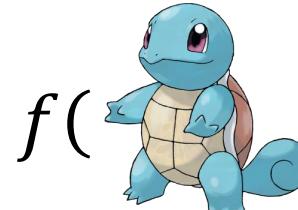
STEEL



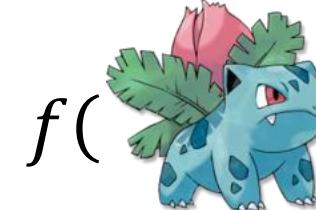
FAIRY



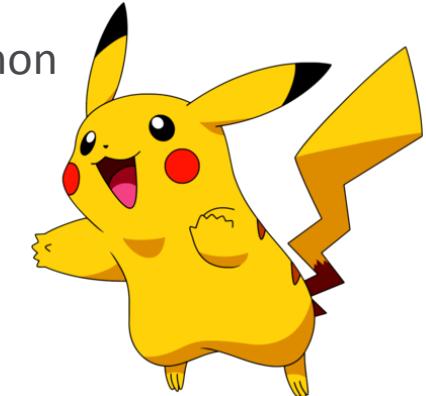
$$f(\text{Pikachu}) = \text{ELECTRIC}$$



$$f(\text{Squirtle}) = \text{WATER}$$



$$f(\text{Bulbasaur}) = \text{GRASS}$$



Example Application

- **Total:** sum of all stats that come after this, a general guide to how strong a pokemon is **320**
- **HP:** hit points, or health, defines how much damage a pokemon can withstand before fainting **35**
- **Attack:** the base modifier for normal attacks (eg. Scratch, Punch) **55**
- **Defense:** the base damage resistance against normal attacks **40**
- **SP Atk:** special attack, the base modifier for special attacks (e.g. fire blast, bubble beam) **50**
- **SP Def:** the base damage resistance against special attacks **50**
- **Speed:** determines which pokemon attacks first each round **90**

Can we predict the “type” of pokemon based on the information?

Example Application

		防禦方的屬性																	
		一般	格鬥	飛行	毒	地面	岩石	蟲	幽靈	鋼	火	水	草	電	超能力	冰	龍	惡	妖精
攻擊方的屬性	一般	1x	1x	1x	1x	1x	1/2x	1x	0x	1/2x	1x	1x	1x	1x	1x	1x	1x	1x	1x
	格鬥	2x	1x	1/2x	1/2x	1x	2x	1/2x	0x	2x	1x	1x	1x	1x	1/2x	2x	1x	2x	1/2x
	飛行	1x	2x	1x	1x	1x	1/2x	2x	1x	1/2x	1x	1x	2x	1/2x	1x	1x	1x	1x	1x
	毒	1x	1x	1x	1/2x	1/2x	1/2x	1x	1/2x	0x	1x	1x	2x	1x	1x	1x	1x	1x	2x
	地面	1x	1x	0x	2x	1x	2x	1/2x	1x	2x	2x	1x	1/2x	2x	1x	1x	1x	1x	1x
	岩石	1x	1/2x	2x	1x	1/2x	1x	2x	1x	1/2x	2x	1x	1x	1x	1x	2x	1x	1x	1x
	蟲	1x	1/2x	1/2x	1/2x	1x	1x	1x	1/2x	1/2x	1/2x	1x	2x	1x	2x	1x	1x	2x	1/2x
	幽靈	0x	1x	1x	1x	1x	1x	1x	2x	1x	1x	1x	1x	1x	2x	1x	1x	1/2x	1x
	鋼	1x	1x	1x	1x	1x	2x	1x	1x	1/2x	1/2x	1/2x	1x	1/2x	1x	2x	1x	1x	2x
	火	1x	1x	1x	1x	1x	1/2x	2x	1x	2x	1/2x	1/2x	2x	1x	1x	2x	1/2x	1x	1x
	水	1x	1x	1x	1x	2x	2x	1x	1x	1x	2x	1/2x	1/2x	1x	1x	1x	1/2x	1x	1x
	草	1x	1x	1/2x	1/2x	2x	2x	1/2x	1x	1/2x	1/2x	2x	1/2x	1x	1x	1x	1/2x	1x	1x
	電	1x	1x	2x	1x	0x	1x	1x	1x	1x	1x	2x	1/2x	1/2x	1x	1x	1/2x	1x	1x
	超能力	1x	2x	1x	2x	1x	1x	1x	1/2x	1x	1x	1x	1x	1/2x	1x	1x	0x	1x	1x
	冰	1x	1x	2x	1x	2x	1x	1x	1/2x	1/2x	1/2x	2x	1x	1x	1/2x	2x	1x	1x	1x
	龍	1x	1x	1x	1x	1x	1x	1x	1x	1/2x	1x	1x	1x	1x	1x	2x	1x	0x	1x
	惡	1x	1/2x	1x	1x	1x	1x	1x	2x	1x	1x	1x	1x	1x	2x	1x	1x	1/2x	1/2x
	妖精	1x	2x	1x	1/2x	1x	1x	1x	1x	1/2x	1/2x	1x	1x	1x	1x	1x	2x	2x	1x

這些倍數適用於XY及之後的遊戲。

How to do Classification

- Training data for Classification

$$(x^1, \hat{y}^1)$$

$$(x^2, \hat{y}^2)$$


... ...

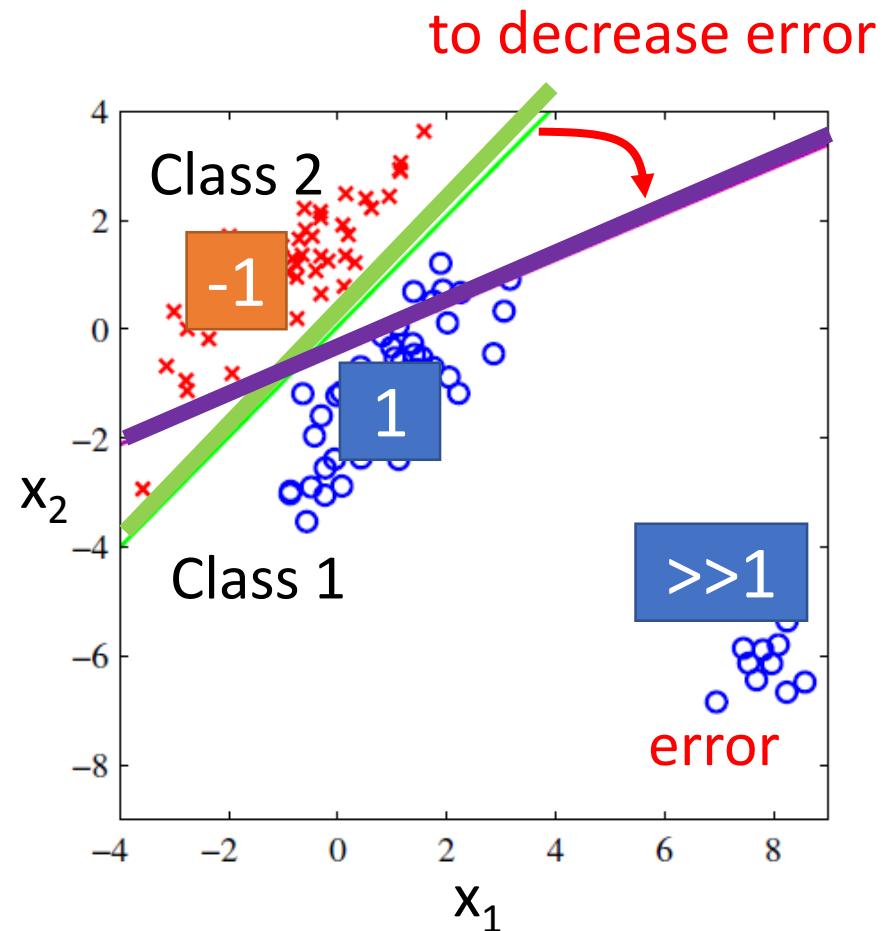
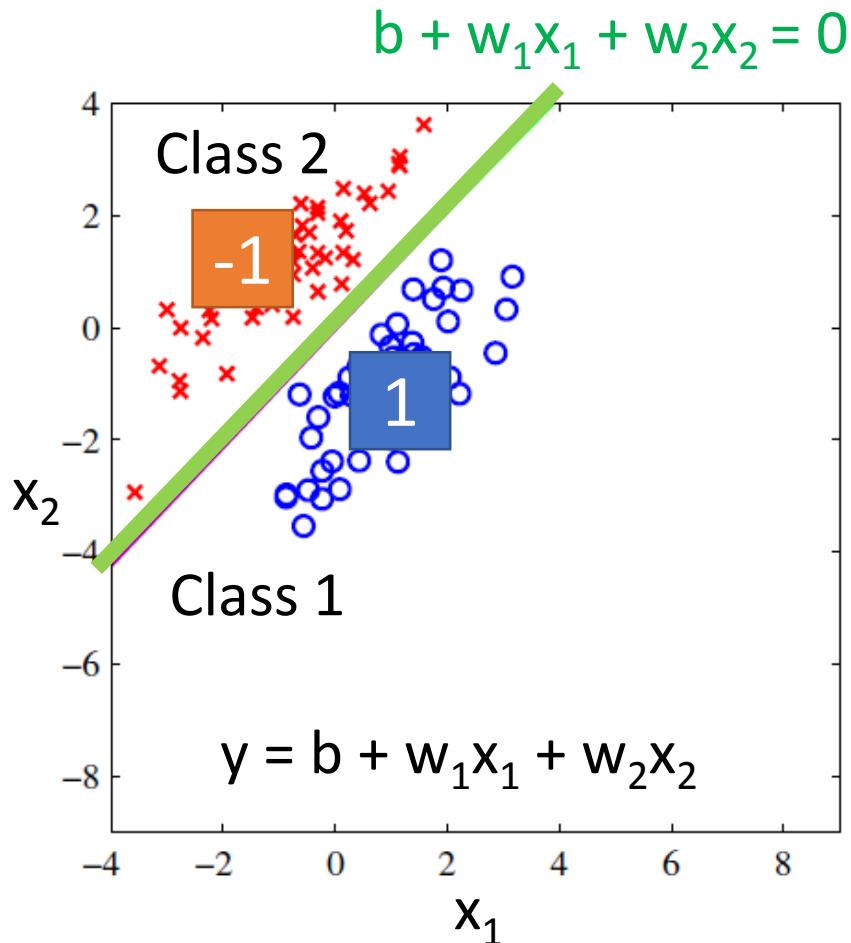
$$(x^N, \hat{y}^N)$$


Classification as Regression?

Binary classification as example

Training: Class 1 means the target is 1; Class 2 means the target is -1

Testing: closer to 1 → class 1; closer to -1 → class 2

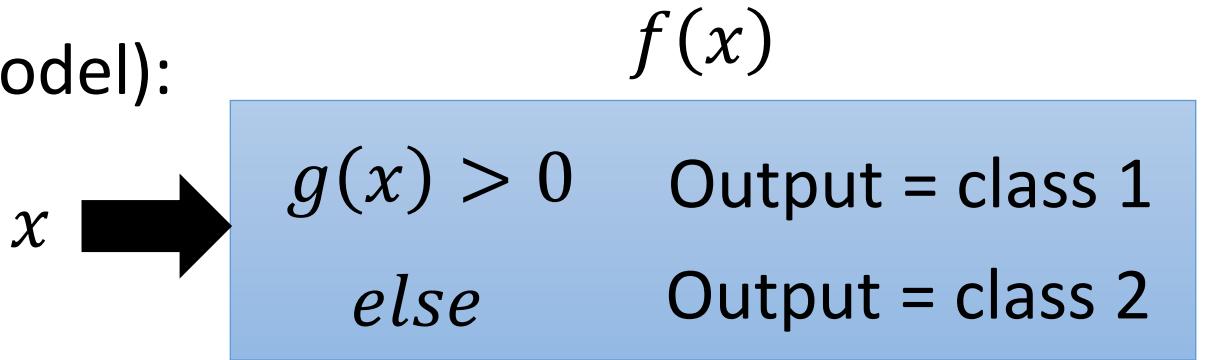


Penalize to the examples that are “too correct” ... (Bishop, P186)

- Multiple class: Class 1 means the target is 1; Class 2 means the target is 2; Class 3 means the target is 3 problematic

Ideal Alternatives

- Function (Model):



- Loss function:

$$L(f) = \sum_n \delta(f(x^n) \neq \hat{y}^n)$$

The number of times f get incorrect results on training data.

- Find the best function:

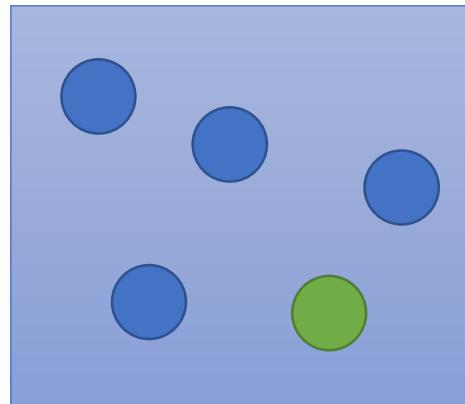
- Example: Perceptron, SVM

Not Today

Two Boxes

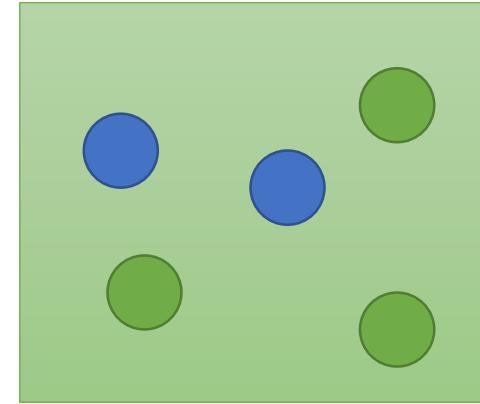
Box 1

$$P(B_1) = 2/3$$



Box 2

$$P(B_2) = 1/3$$



$$P(\text{Blue} | B_1) = 4/5$$

$$P(\text{Green} | B_1) = 1/5$$

$$P(\text{Blue} | B_2) = 2/5$$

$$P(\text{Green} | B_2) = 3/5$$



from one of the boxes

Where does it come from?

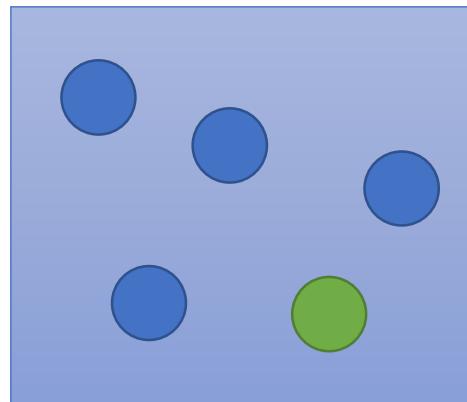
$$P(B_1 | \text{Blue}) = \frac{P(\text{Blue} | B_1)P(B_1)}{P(\text{Blue} | B_1)P(B_1) + P(\text{Blue} | B_2)P(B_2)}$$

Two Classes

Estimating the Probabilities
From training data

Class 1

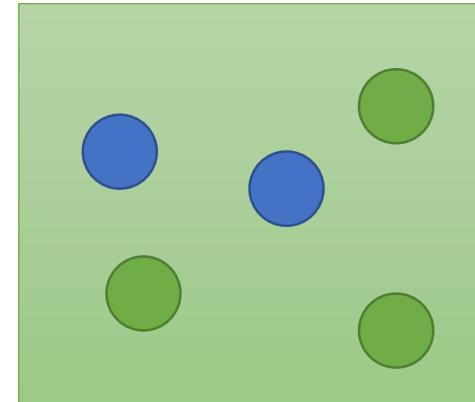
$$P(C_1)$$



$$P(x|C_1)$$

Class 2

$$P(C_2)$$



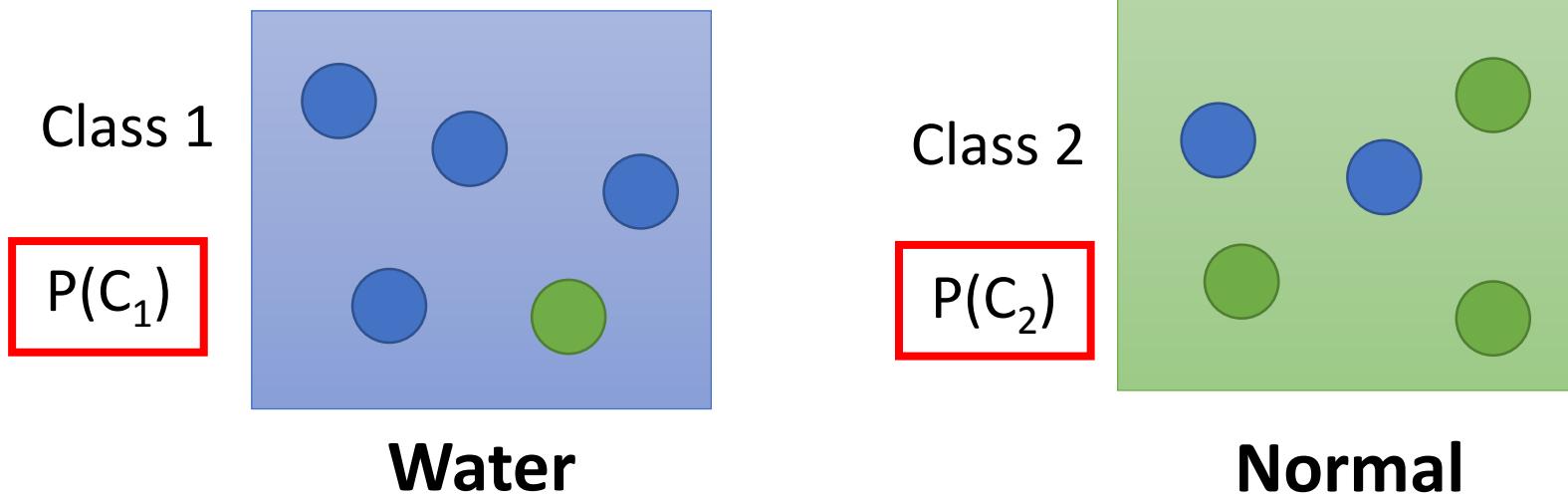
$$P(x|C_2)$$

Given an x , which class does it belong to

$$P(C_1|x) = \frac{P(x|C_1)P(C_1)}{P(x|C_1)P(C_1) + P(x|C_2)P(C_2)}$$

Generative Model $P(x) = P(x|C_1)P(C_1) + P(x|C_2)P(C_2)$

Prior



Water and Normal type with ID < 400 for training,
rest for testing

Training: 79 Water, 61 Normal

$$P(C_1) = 79 / (79 + 61) = 0.56$$

$$P(C_2) = 61 / (79 + 61) = 0.44$$

Probability from Class

$$P(x|C_1) = ? \quad P($$



$$| \text{Water}) = ?$$

Each Pokémon is represented as
a vector by its attribute.

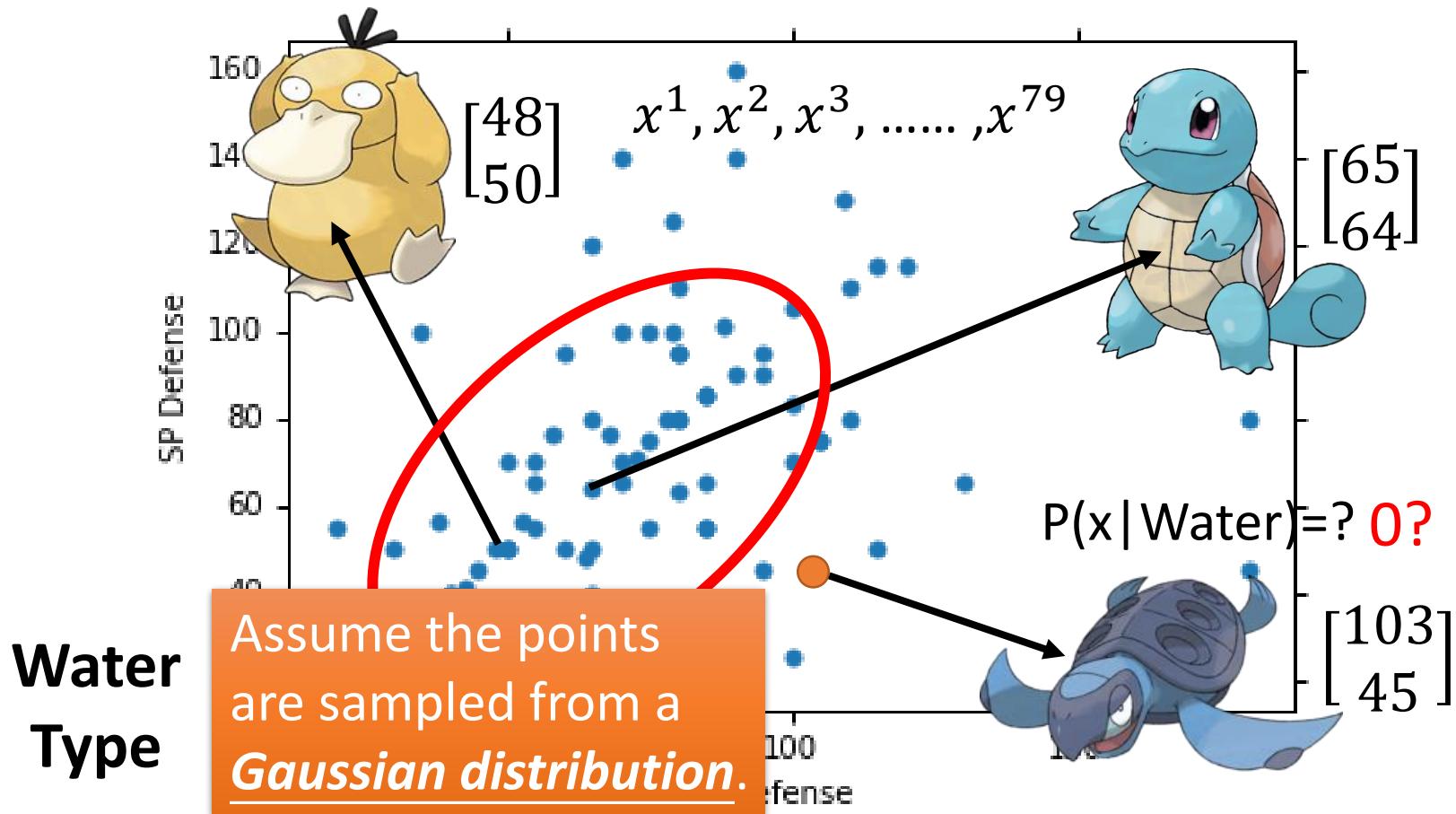
feature

Water
Type



Probability from Class - Feature

- Considering **Defense** and **SP Defense**

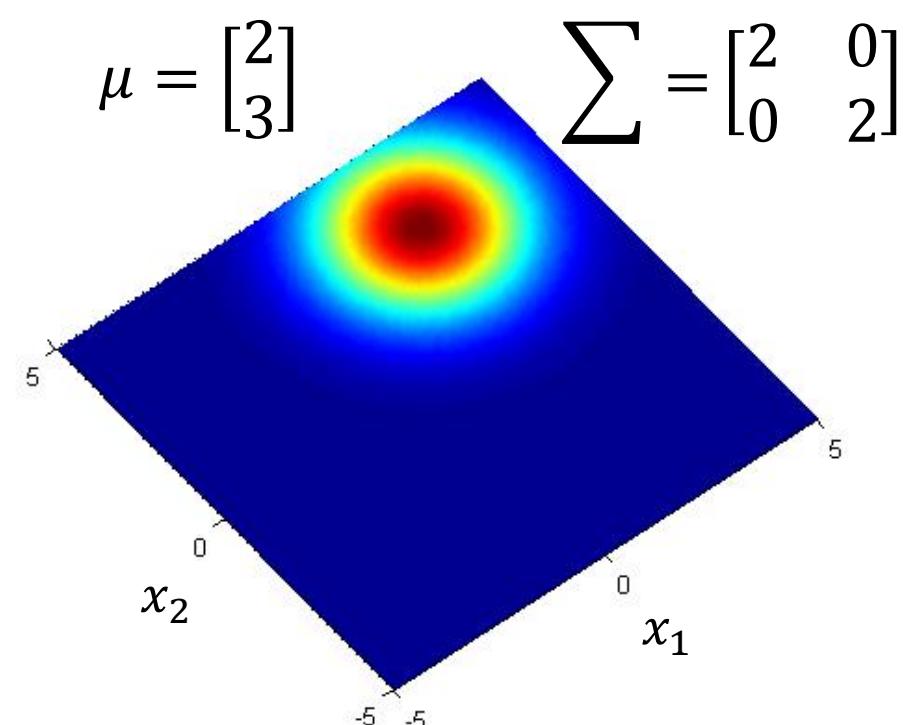
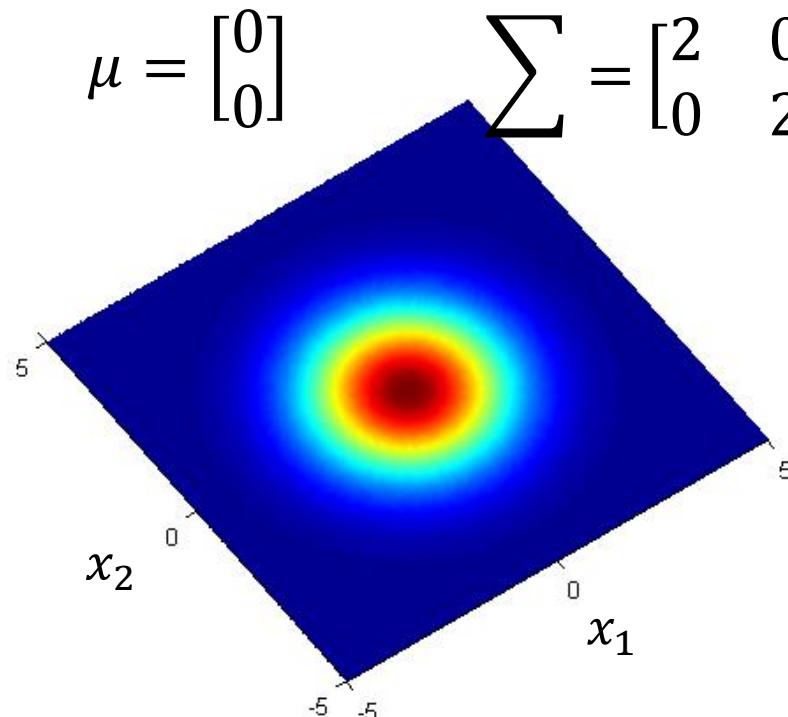


Gaussian Distribution

$$f_{\mu, \Sigma}(x) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\Sigma|^{1/2}} \exp \left\{ -\frac{1}{2} (x - \mu)^T \Sigma^{-1} (x - \mu) \right\}$$

Input: vector x , output: probability of sampling x

The shape of the function determines by **mean μ** and **covariance matrix Σ**

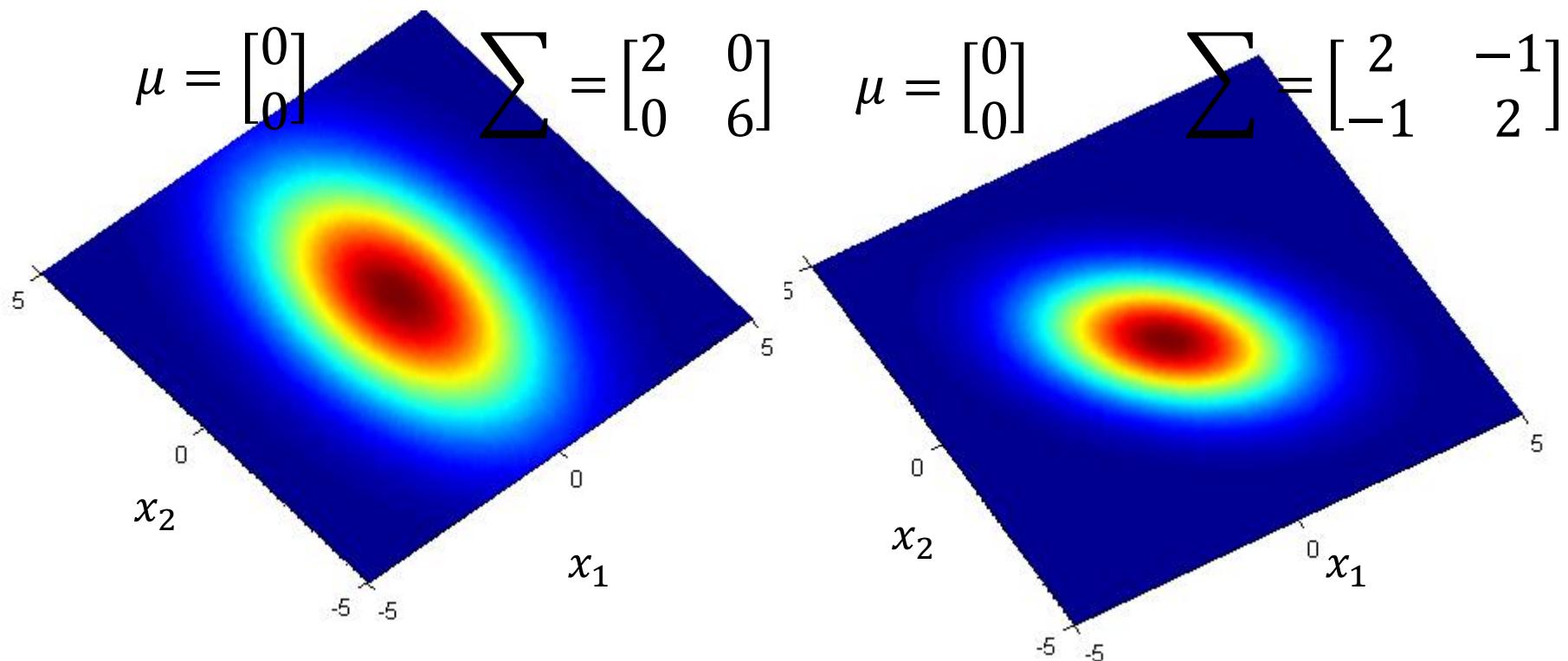


Gaussian Distribution

$$f_{\mu, \Sigma}(x) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\Sigma|^{1/2}} \exp \left\{ -\frac{1}{2} (x - \mu)^T \Sigma^{-1} (x - \mu) \right\}$$

Input: vector x , output: probability of sampling x

The shape of the function determines by **mean μ** and **covariance matrix Σ**



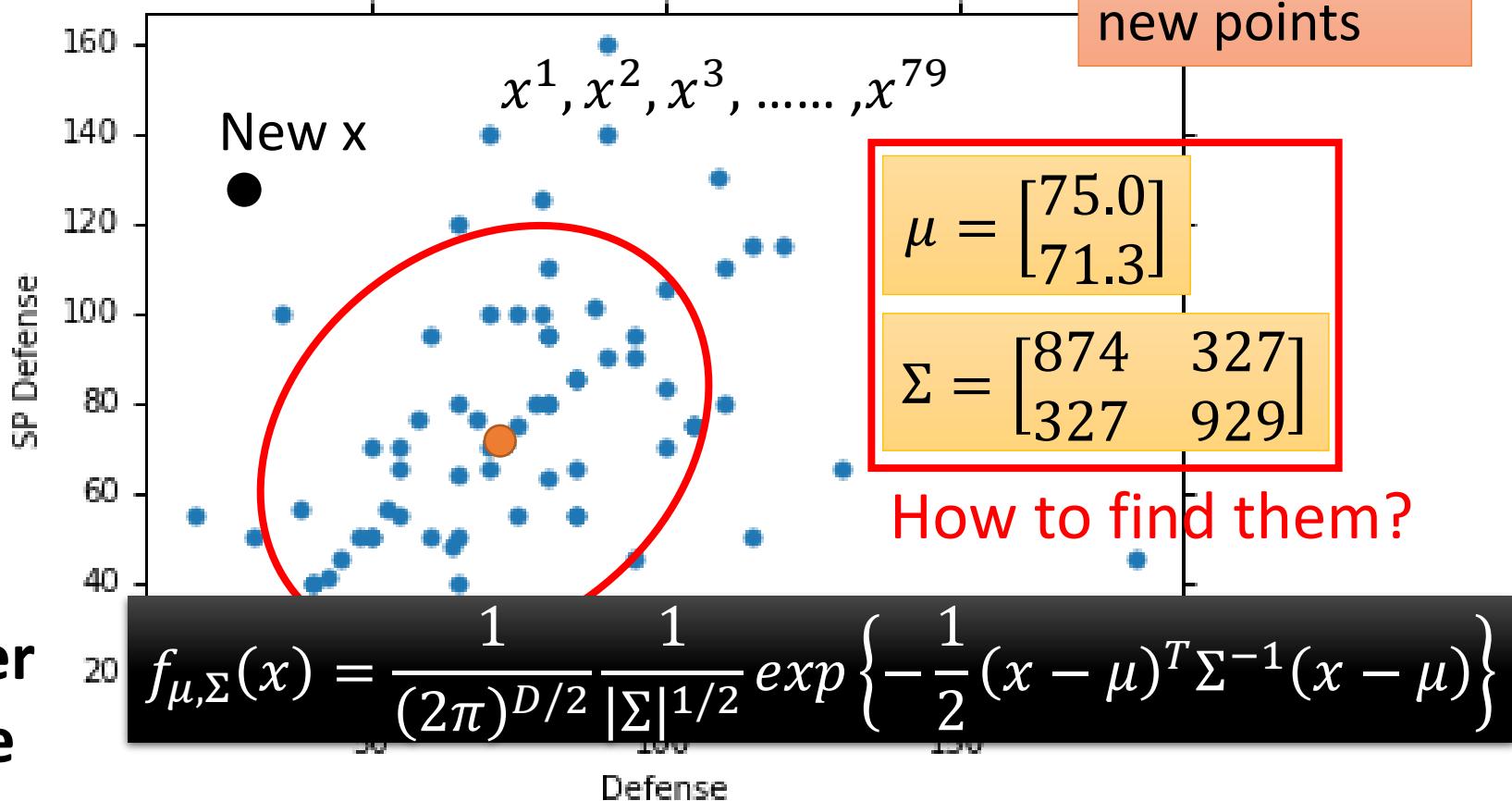
Probability from Class

Assume the points are sampled from a Gaussian distribution

Find the Gaussian distribution behind them

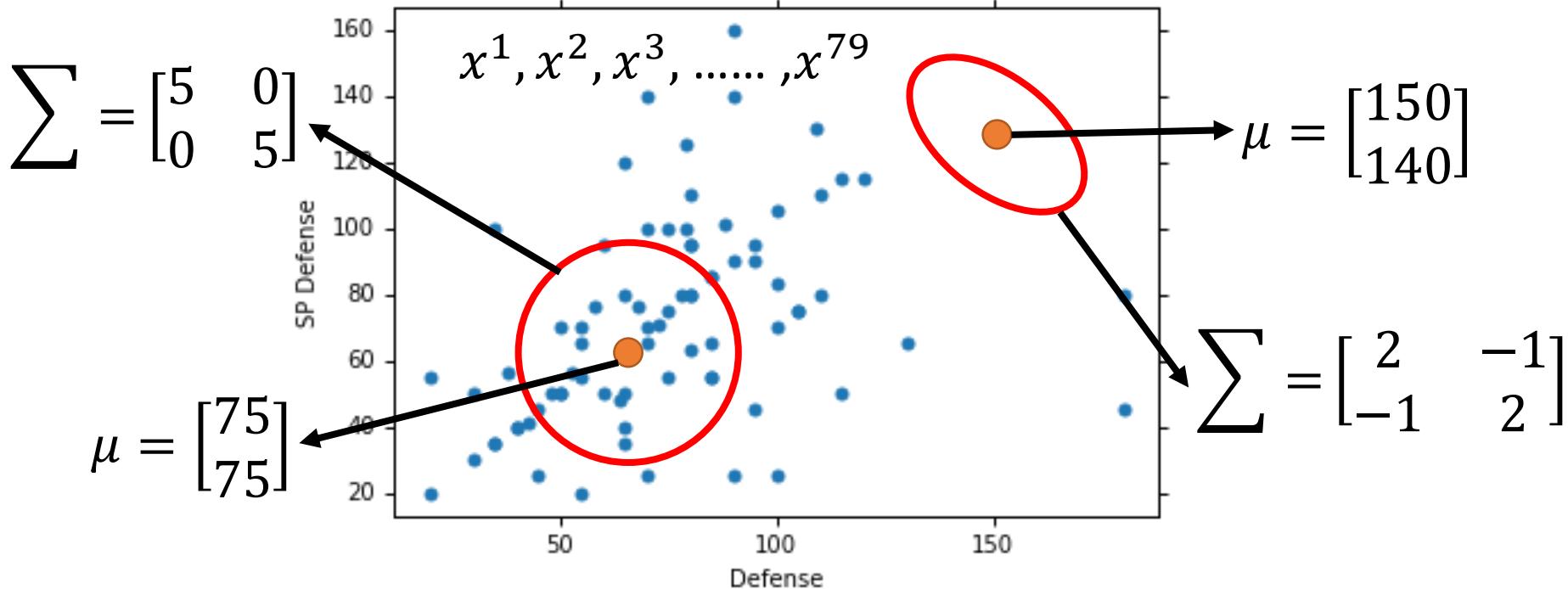


Probability for
new points



Maximum Likelihood

$$f_{\mu, \Sigma}(x) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\Sigma|^{1/2}} \exp \left\{ -\frac{1}{2} (x - \mu)^T \Sigma^{-1} (x - \mu) \right\}$$



The Gaussian with any mean μ and covariance matrix Σ can generate these points.

→ Different Likelihood

Likelihood of a Gaussian with mean μ and covariance matrix Σ

= the probability of the Gaussian samples $x^1, x^2, x^3, \dots, x^{79}$

$$L(\mu, \Sigma) = f_{\mu, \Sigma}(x^1) f_{\mu, \Sigma}(x^2) f_{\mu, \Sigma}(x^3) \dots f_{\mu, \Sigma}(x^{79})$$

Maximum Likelihood

We have the “Water” type Pokémons: $x^1, x^2, x^3, \dots, x^{79}$

We assume $x^1, x^2, x^3, \dots, x^{79}$ generate from the Gaussian (μ^*, Σ^*) with the **maximum likelihood**

$$L(\mu, \Sigma) = f_{\mu, \Sigma}(x^1)f_{\mu, \Sigma}(x^2)f_{\mu, \Sigma}(x^3) \dots f_{\mu, \Sigma}(x^{79})$$

$$f_{\mu, \Sigma}(x) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\Sigma|^{1/2}} \exp \left\{ -\frac{1}{2} (x - \mu)^T \Sigma^{-1} (x - \mu) \right\}$$

$$\mu^*, \Sigma^* = \arg \max_{\mu, \Sigma} L(\mu, \Sigma)$$

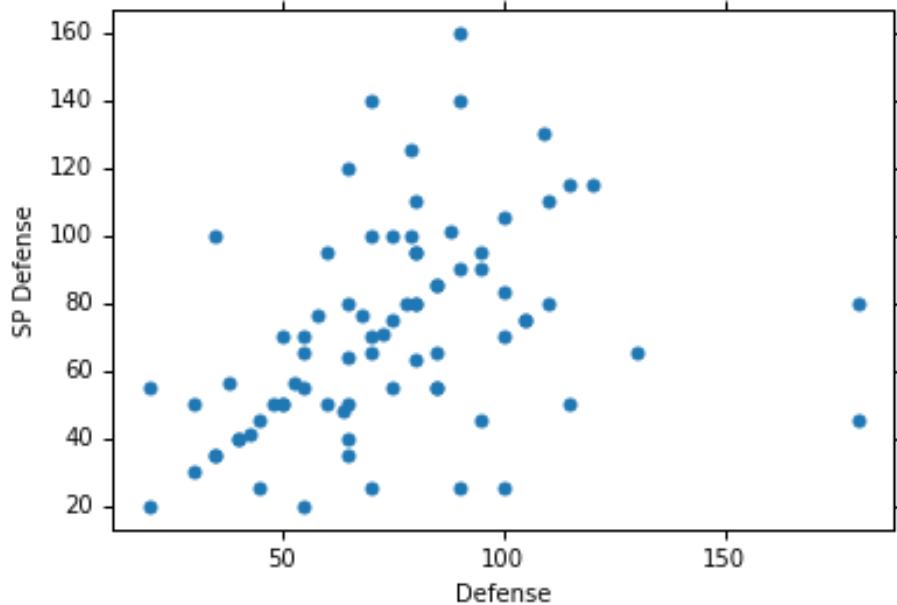
$$\mu^* = \frac{1}{79} \sum_{n=1}^{79} x^n$$

average

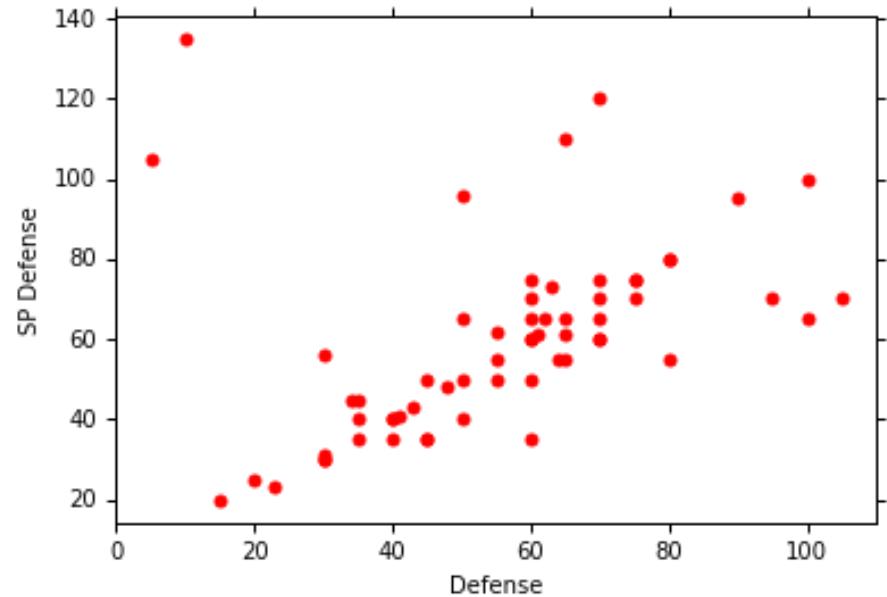
$$\Sigma^* = \frac{1}{79} \sum_{n=1}^{79} (x^n - \mu^*) (x^n - \mu^*)^T$$

Maximum Likelihood

Class 1: Water



Class 2: Normal



$$\mu^1 = \begin{bmatrix} 75.0 \\ 71.3 \end{bmatrix} \quad \Sigma^1 = \begin{bmatrix} 874 & 327 \\ 327 & 929 \end{bmatrix}$$

$$\mu^2 = \begin{bmatrix} 55.6 \\ 59.8 \end{bmatrix} \quad \Sigma^2 = \begin{bmatrix} 847 & 422 \\ 422 & 685 \end{bmatrix}$$

Now we can do classification 😊

$$f_{\mu^1, \Sigma^1}(x) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\Sigma^1|^{1/2}} \exp \left\{ -\frac{1}{2} (x - \mu^1)^T (\Sigma^1)^{-1} (x - \mu^1) \right\}$$

$$\mu^1 = \begin{bmatrix} 75.0 \\ 71.3 \end{bmatrix} \quad \Sigma^1 = \begin{bmatrix} 874 & 327 \\ 327 & 929 \end{bmatrix}$$

$$P(C_1|x) = \frac{P(x|C_1)P(C_1)}{P(x|C_1)P(C_1) + P(x|C_2)P(C_2)}$$

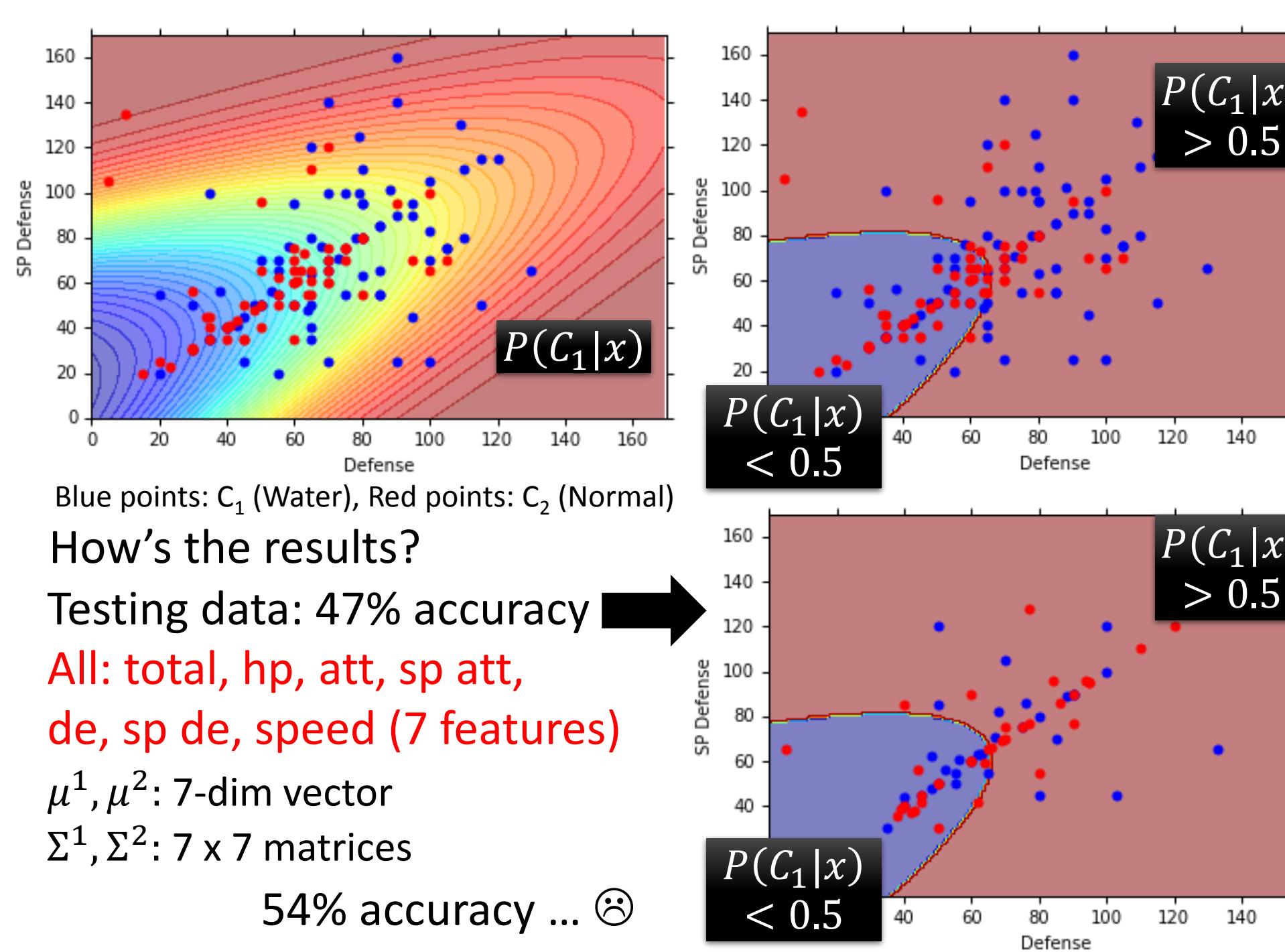
$$f_{\mu^2, \Sigma^2}(x) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\Sigma^2|^{1/2}} \exp \left\{ -\frac{1}{2} (x - \mu^2)^T (\Sigma^2)^{-1} (x - \mu^2) \right\}$$

$$\mu^2 = \begin{bmatrix} 55.6 \\ 59.8 \end{bmatrix} \quad \Sigma^2 = \begin{bmatrix} 847 & 422 \\ 422 & 685 \end{bmatrix}$$

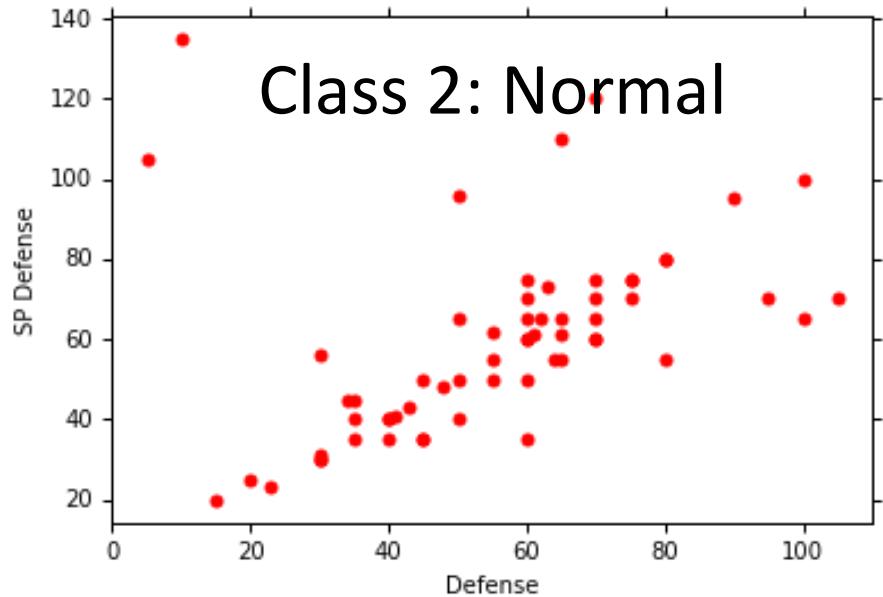
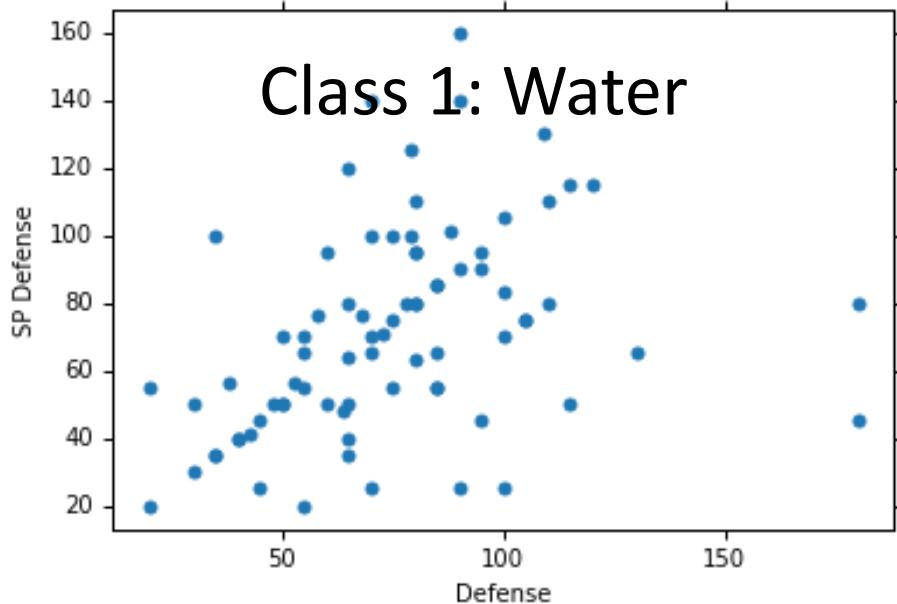
$$\begin{aligned} P(C1) \\ = 79 / (79 + 61) = 0.56 \end{aligned}$$

$$\begin{aligned} P(C2) \\ = 61 / (79 + 61) \\ = 0.44 \end{aligned}$$

If $P(C_1|x) > 0.5$ ➡ x belongs to class 1 (Water)



Modifying Model



$$\mu^1 = \begin{bmatrix} 75.0 \\ 71.3 \end{bmatrix} \quad \Sigma^1 = \begin{bmatrix} 874 & 327 \\ 327 & 929 \end{bmatrix}$$

$$\mu^2 = \begin{bmatrix} 55.6 \\ 59.8 \end{bmatrix} \quad \Sigma^2 = \begin{bmatrix} 847 & 422 \\ 422 & 685 \end{bmatrix}$$

The same Σ
Less parameters

Modifying Model

Ref: Bishop,
chapter 4.2.2

- Maximum likelihood

“Water” type Pokémons:

$$x^1, x^2, x^3, \dots, x^{79}$$

$$\mu^1$$

“Normal” type Pokémons:

$$x^{80}, x^{81}, x^{82}, \dots, x^{140}$$

$$\mu^2$$

$$\Sigma$$

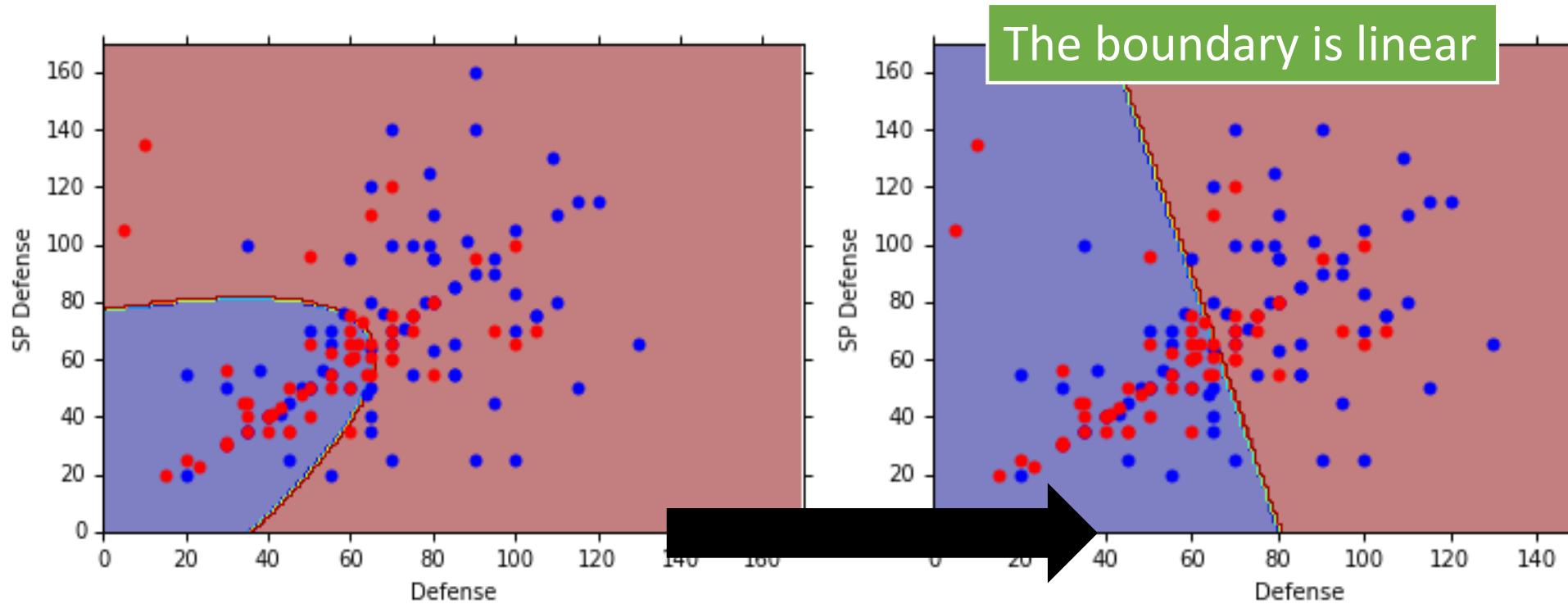
Find μ^1, μ^2, Σ maximizing the likelihood $L(\mu^1, \mu^2, \Sigma)$

$$\begin{aligned} L(\mu^1, \mu^2, \Sigma) &= f_{\mu^1, \Sigma}(x^1) f_{\mu^1, \Sigma}(x^2) \cdots f_{\mu^1, \Sigma}(x^{79}) \\ &\quad \times f_{\mu^2, \Sigma}(x^{80}) f_{\mu^2, \Sigma}(x^{81}) \cdots f_{\mu^2, \Sigma}(x^{140}) \end{aligned}$$

μ^1 and μ^2 is the same

$$\Sigma = \frac{79}{140} \Sigma^1 + \frac{61}{140} \Sigma^2$$

Modifying Model



The same covariance matrix

All: total, hp, att, sp att, de, sp de, speed

54% accuracy → 73% accuracy

Three Steps

- Function Set (Model):

$x \rightarrow$

$$P(C_1|x) = \frac{P(x|C_1)P(C_1)}{P(x|C_1)P(C_1) + P(x|C_2)P(C_2)}$$

If $P(C_1|x) > 0.5$, output: class 1

Otherwise, output: class 2

- Goodness of a function:

- The mean μ and covariance Σ that maximizing the likelihood (the probability of generating data)
- Find the best function: easy

Probability Distribution

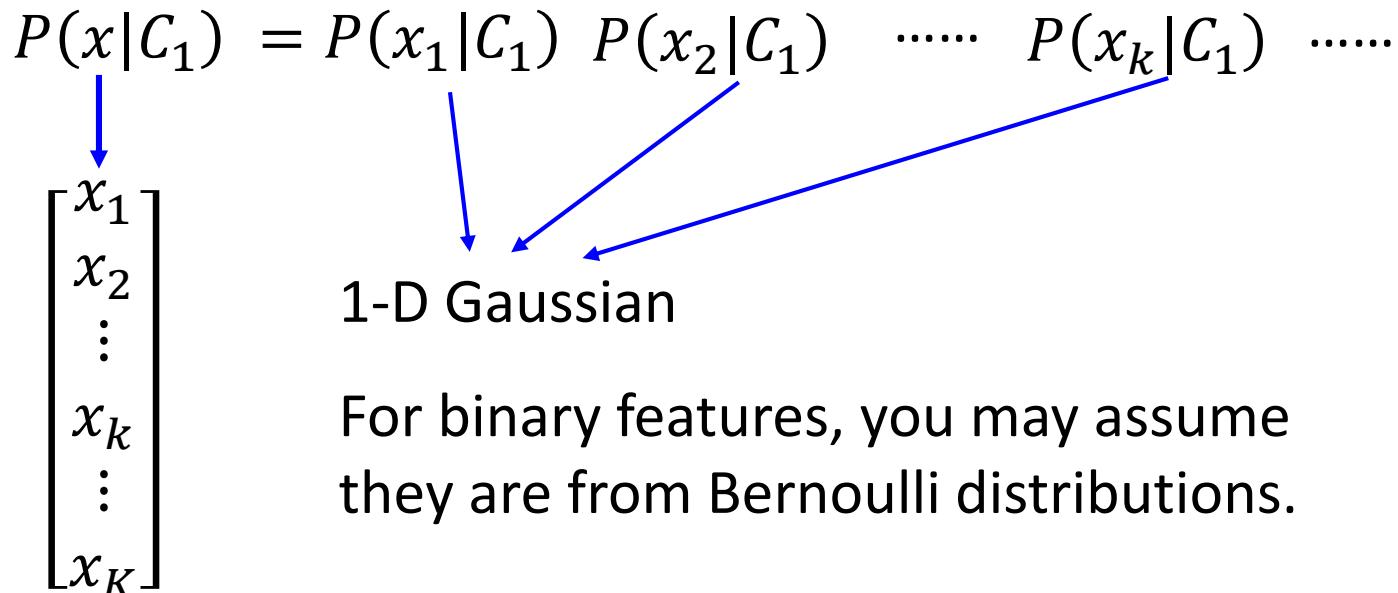
- You can always use the distribution you like 😊

$$P(x|C_1) = P(x_1|C_1) P(x_2|C_1) \dots \dots P(x_k|C_1) \dots \dots$$

$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_k \\ \vdots \\ x_K \end{bmatrix}$

1-D Gaussian

For binary features, you may assume they are from Bernoulli distributions.



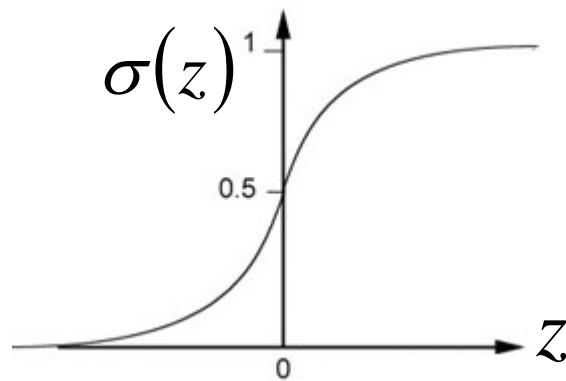
If you assume all the dimensions are independent, then you are using *Naive Bayes Classifier*.

Posterior Probability

$$P(C_1|x) = \frac{P(x|C_1)P(C_1)}{P(x|C_1)P(C_1) + P(x|C_2)P(C_2)}$$
$$= \frac{1}{1 + \frac{P(x|C_2)P(C_2)}{P(x|C_1)P(C_1)}} = \frac{1}{1 + \exp(-z)} = \sigma(z)$$

Sigmoid function

$$z = \ln \frac{P(x|C_1)P(C_1)}{P(x|C_2)P(C_2)}$$



Warning of Math

Posterior Probability

$$P(C_1|x) = \sigma(z) \quad \text{sigmoid} \quad z = \ln \frac{P(x|C_1)P(C_1)}{P(x|C_2)P(C_2)}$$

$$z = \ln \frac{P(x|C_1)}{P(x|C_2)} + \ln \frac{P(C_1)}{P(C_2)} \rightarrow \frac{\frac{N_1}{N_1 + N_2}}{\frac{N_2}{N_1 + N_2}} = \frac{N_1}{N_2}$$

$$P(x|C_1) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\Sigma^1|^{1/2}} \exp \left\{ -\frac{1}{2} (x - \mu^1)^T (\Sigma^1)^{-1} (x - \mu^1) \right\}$$

$$P(x|C_2) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\Sigma^2|^{1/2}} \exp \left\{ -\frac{1}{2} (x - \mu^2)^T (\Sigma^2)^{-1} (x - \mu^2) \right\}$$

$$z = \ln \frac{P(x|C_1)}{P(x|C_2)} + \ln \frac{P(C_1)}{P(C_2)} = \frac{N_1}{N_2}$$

$$P(x|C_1) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\Sigma^1|^{1/2}} \exp \left\{ -\frac{1}{2} (x - \mu^1)^T (\Sigma^1)^{-1} (x - \mu^1) \right\}$$

$$P(x|C_2) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\Sigma^2|^{1/2}} \exp \left\{ -\frac{1}{2} (x - \mu^2)^T (\Sigma^2)^{-1} (x - \mu^2) \right\}$$

$$\ln \frac{\frac{1}{(2\pi)^{D/2}} \frac{1}{|\Sigma^1|^{1/2}} \exp \left\{ -\frac{1}{2} (x - \mu^1)^T (\Sigma^1)^{-1} (x - \mu^1) \right\}}{\frac{1}{(2\pi)^{D/2}} \frac{1}{|\Sigma^2|^{1/2}} \exp \left\{ -\frac{1}{2} (x - \mu^2)^T (\Sigma^2)^{-1} (x - \mu^2) \right\}}$$

$$= \ln \frac{|\Sigma^2|^{1/2}}{|\Sigma^1|^{1/2}} \exp \left\{ -\frac{1}{2} [(x - \mu^1)^T (\Sigma^1)^{-1} (x - \mu^1) - (x - \mu^2)^T (\Sigma^2)^{-1} (x - \mu^2)] \right\}$$

$$= \ln \frac{|\Sigma^2|^{1/2}}{|\Sigma^1|^{1/2}} - \frac{1}{2} [(x - \mu^1)^T (\Sigma^1)^{-1} (x - \mu^1) - (x - \mu^2)^T (\Sigma^2)^{-1} (x - \mu^2)]$$

$$z = \ln \frac{P(x|C_1)}{P(x|C_2)} + \ln \frac{P(C_1)}{P(C_2)} = \frac{N_1}{N_2}$$

$$= \ln \frac{|\Sigma^2|^{1/2}}{|\Sigma^1|^{1/2}} - \frac{1}{2} [(x - \mu^1)^T (\Sigma^1)^{-1} (x - \mu^1) - (x - \mu^2)^T (\Sigma^2)^{-1} (x - \mu^2)]$$

$$(x - \mu^1)^T (\Sigma^1)^{-1} (x - \mu^1)$$

$$= x^T (\Sigma^1)^{-1} x - x^T (\Sigma^1)^{-1} \mu^1 - (\mu^1)^T (\Sigma^1)^{-1} x + (\mu^1)^T (\Sigma^1)^{-1} \mu^1$$

$$= x^T (\Sigma^1)^{-1} x - 2(\mu^1)^T (\Sigma^1)^{-1} x + (\mu^1)^T (\Sigma^1)^{-1} \mu^1$$

$$(x - \mu^2)^T (\Sigma^2)^{-1} (x - \mu^2)$$

$$= x^T (\Sigma^2)^{-1} x - 2(\mu^2)^T (\Sigma^2)^{-1} x + (\mu^2)^T (\Sigma^2)^{-1} \mu^2$$

$$\begin{aligned} z &= \ln \frac{|\Sigma^2|^{1/2}}{|\Sigma^1|^{1/2}} - \frac{1}{2} x^T (\Sigma^1)^{-1} x + (\mu^1)^T (\Sigma^1)^{-1} x - \frac{1}{2} (\mu^1)^T (\Sigma^1)^{-1} \mu^1 \\ &\quad + \frac{1}{2} x^T (\Sigma^2)^{-1} x - (\mu^2)^T (\Sigma^2)^{-1} x + \frac{1}{2} (\mu^2)^T (\Sigma^2)^{-1} \mu^2 + \ln \frac{N_1}{N_2} \end{aligned}$$

End of Warning

$$P(C_1|x) = \sigma(z)$$

$$\begin{aligned} z &= \ln \frac{|\Sigma^2|^{1/2}}{|\Sigma^1|^{1/2}} - \frac{1}{2} x^T (\Sigma^1)^{-1} x + (\mu^1)^T (\Sigma^1)^{-1} x - \frac{1}{2} (\mu^1)^T (\Sigma^1)^{-1} \mu^1 \\ &\quad + \frac{1}{2} x^T (\Sigma^2)^{-1} x - (\mu^2)^T (\Sigma^2)^{-1} x + \frac{1}{2} (\mu^2)^T (\Sigma^2)^{-1} \mu^2 + \ln \frac{N_1}{N_2} \end{aligned}$$

$$\Sigma_1 = \Sigma_2 = \Sigma$$

$$z = \frac{(\mu^1 - \mu^2)^T \Sigma^{-1} x - \frac{1}{2} (\mu^1)^T \Sigma^{-1} \mu^1 + \frac{1}{2} (\mu^2)^T \Sigma^{-1} \mu^2 + \ln \frac{N_1}{N_2}}{\mathbf{w}^T \mathbf{b}}$$

$$P(C_1|x) = \sigma(w \cdot x + b)$$

How about directly find w and b ?

In generative model, we estimate $N_1, N_2, \mu^1, \mu^2, \Sigma$

Then we have w and b

Reference

- Bishop: Chapter 4.1 – 4.2
- Data: <https://www.kaggle.com/abcsds/pokemon>
- Useful posts:
 - <https://www.kaggle.com/nishantbhadauria/d/abcsds/pokemon/pokemon-speed-attack-hp-defense-analysis-by-type>
 - <https://www.kaggle.com/nikos90/d/abcsds/pokemon/mastering-pokebars/discussion>
 - <https://www.kaggle.com/ndrewgele/d/abcsds/pokemon/visualizing-pok-mon-stats-with-seaborn/discussion>

Acknowledgment

- 感謝 江貫榮 同學發現課程網頁上的日期錯誤
- 感謝 范廷瀚 同學提供寶可夢的 domain knowledge
- 感謝 Victor Chen 發現投影片上的打字錯誤

Logistic Regression

Step 1: Function Set

We want to find $P_{w,b}(C_1|x)$

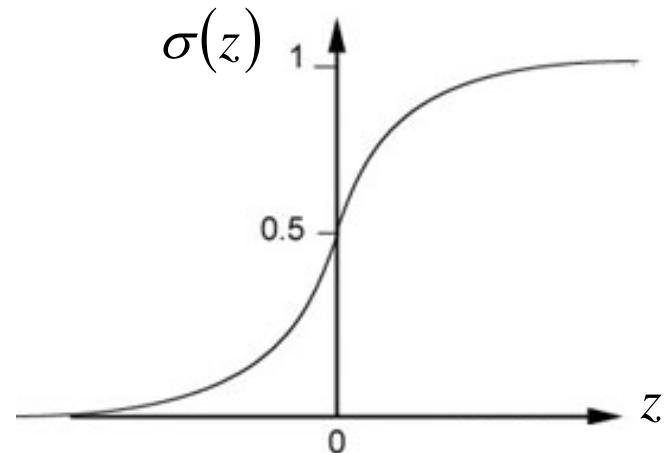
If $P_{w,b}(C_1|x) \geq 0.5$, output C_1

Otherwise, output C_2

$$P_{w,b}(C_1|x) = \sigma(z)$$

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

$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$

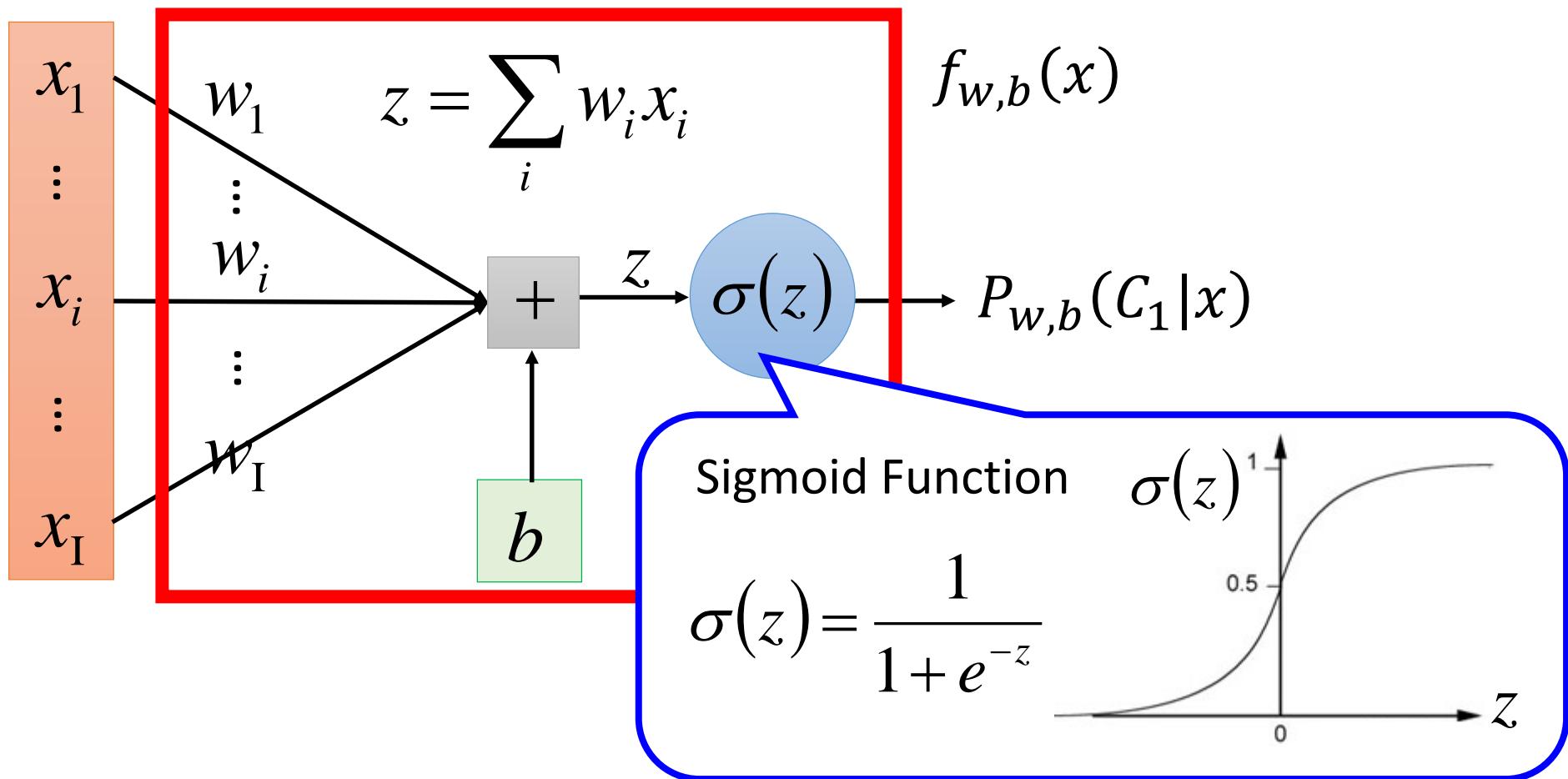


Function set:

$$f_{w,b}(x) = P_{w,b}(C_1|x)$$

Including all
different w and b

Step 1: Function Set



Logistic Regression

Step 1: $f_{w,b}(x) = \sigma\left(\sum_i w_i x_i + b\right)$

Output: between 0 and 1

Linear Regression

$$f_{w,b}(x) = \sum_i w_i x_i + b$$

Output: any value

Step 2:

Step 3:

Step 2: Goodness of a Function

Training
Data

	x^1	x^2	x^3	x^N
	C_1	C_1	C_2	C_1

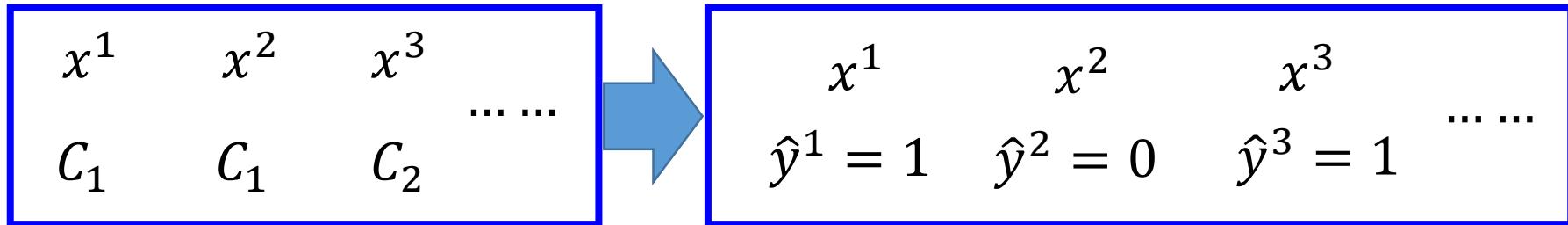
Assume the data is generated based on $f_{w,b}(x) = P_{w,b}(C_1|x)$

Given a set of w and b , what is its probability of generating the data?

$$L(w, b) = f_{w,b}(x^1)f_{w,b}(x^2)\left(1 - f_{w,b}(x^3)\right)\cdots f_{w,b}(x^N)$$

The most likely w^* and b^* is the one with the largest $L(w, b)$.

$$w^*, b^* = \arg \max_{w,b} L(w, b)$$



\hat{y}^n : 1 for class 1, 0 for class 2

$$L(w, b) = f_{w,b}(x^1)f_{w,b}(x^2)\left(1 - f_{w,b}(x^3)\right)\cdots$$

$$w^*, b^* = \arg \max_{w,b} L(w, b) = w^*, b^* = \arg \min_{w,b} -\ln L(w, b)$$

$$-\ln L(w, b)$$

$$= -\ln f_{w,b}(x^1) \rightarrow -[1 \ln f(x^1) + 0 \ln(1 - f(x^1))]$$

$$-\ln f_{w,b}(x^2) \rightarrow -[1 \ln f(x^2) + 0 \ln(1 - f(x^2))]$$

$$-\ln \left(1 - f_{w,b}(x^3)\right) \rightarrow -[0 \ln f(x^3) + 1 \ln(1 - f(x^3))]$$

⋮

Step 2: Goodness of a Function

$$L(w, b) = f_{w,b}(x^1)f_{w,b}(x^2)\left(1 - f_{w,b}(x^3)\right)\cdots f_{w,b}(x^N)$$

$$-lnL(w, b) = lnf_{w,b}(x^1) + lnf_{w,b}(x^2) + ln\left(1 - f_{w,b}(x^3)\right)\cdots$$

\hat{y}^n : 1 for class 1, 0 for class 2

$$= \sum_n -\left[\hat{y}^n ln f_{w,b}(x^n) + (1 - \hat{y}^n) ln \left(1 - f_{w,b}(x^n)\right)\right]$$

Cross entropy between two Bernoulli distribution

Distribution p:

$$p(x = 1) = \hat{y}^n$$

$$p(x = 0) = 1 - \hat{y}^n$$

Distribution q:

$$q(x = 1) = f(x^n)$$

$$q(x = 0) = 1 - f(x^n)$$

←
cross
entropy→

$$H(p, q) = -\sum_x p(x)ln(q(x))$$

Logistic Regression

Step 1: $f_{w,b}(x) = \sigma\left(\sum_i w_i x_i + b\right)$

Output: between 0 and 1

Linear Regression

$$f_{w,b}(x) = \sum_i w_i x_i + b$$

Output: any value

Training data: (x^n, \hat{y}^n)

Step 2: \hat{y}^n : 1 for class 1, 0 for class 2

$$L(f) = \sum_n C(f(x^n), \hat{y}^n)$$

Training data: (x^n, \hat{y}^n)

\hat{y}^n : a real number

$$L(f) = \frac{1}{2} \sum_n (f(x^n) - \hat{y}^n)^2$$

Cross entropy:

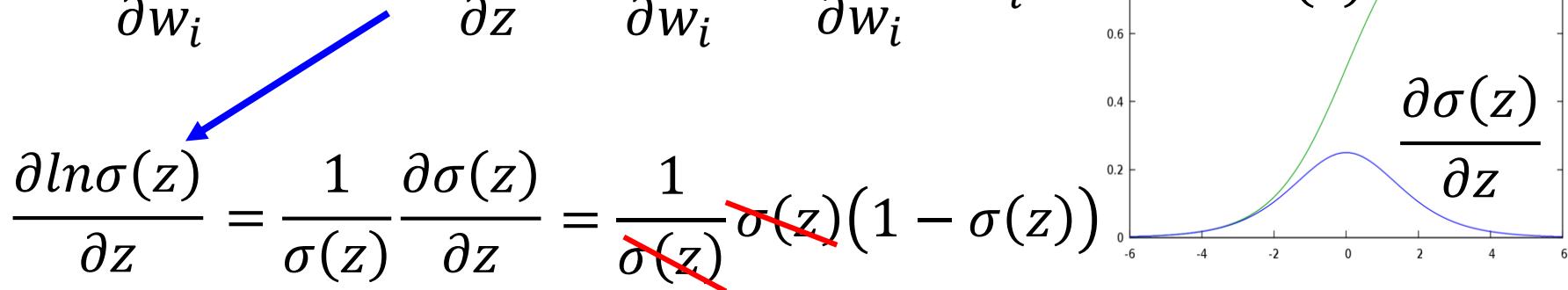
$$C(f(x^n), \hat{y}^n) = -[\hat{y}^n \ln f(x^n) + (1 - \hat{y}^n) \ln(1 - f(x^n))]$$

Question: Why don't we simply use square error as linear regression?

Step 3: Find the best function

$$\frac{\partial \ln L(w, b)}{\partial w_i} = \sum_n - \left[\hat{y}^n \frac{\partial \ln f_{w,b}(x^n)}{\partial w_i} + (1 - \hat{y}^n) \frac{\ln(1 - f_{w,b}(x^n))}{\partial w_i} \right]$$

$$\frac{\partial \ln f_{w,b}(x)}{\partial w_i} = \frac{\partial \ln f_{w,b}(x)}{\partial z} \frac{\partial z}{\partial w_i} \quad \frac{\partial z}{\partial w_i} = x_i$$



$$\begin{aligned} f_{w,b}(x) &= \sigma(z) \\ &= 1 / (1 + \exp(-z)) \end{aligned}$$

$$z = w \cdot x + b = \sum_i w_i x_i + b$$

Step 3: Find the best function

$$\frac{-\ln L(w, b)}{\partial w_i} = \sum_n - \left[\hat{y}^n \frac{\ln f_{w,b}(x^n)}{\partial w_i} + (1 - \hat{y}^n) \frac{\ln (1 - f_{w,b}(x^n))}{\partial w_i} \right]$$

$$\frac{\partial \ln (1 - f_{w,b}(x))}{\partial w_i} = \frac{\partial \ln (1 - f_{w,b}(x))}{\partial z} \frac{\partial z}{\partial w_i} \quad \frac{\partial z}{\partial w_i} = x_i$$

$$\frac{\partial \ln(1 - \sigma(z))}{\partial z} = -\frac{1}{1 - \sigma(z)} \frac{\partial \sigma(z)}{\partial z} = -\frac{1}{1 - \sigma(z)} \sigma(z)(1 - \sigma(z))$$

$$f_{w,b}(x) = \sigma(z)$$

$$= 1 / 1 + \exp(-z)$$

$$z = w \cdot x + b = \sum_i w_i x_i + b$$

Step 3: Find the best function

$$\frac{-\ln L(w, b)}{\partial w_i} = \sum_n - \left[\hat{y}^n \frac{\ln f_{w,b}(x^n)}{\partial w_i} + (1 - \hat{y}^n) \frac{\ln (1 - f_{w,b}(x^n))}{\partial w_i} \right]$$

$$= \sum_n - \left[\hat{y}^n \underbrace{(1 - f_{w,b}(x^n))}_{\textcolor{blue}{-}} x_i^n - (1 - \hat{y}^n) \underbrace{f_{w,b}(x^n)}_{\textcolor{blue}{-}} x_i^n \right]$$

$$= \sum_n - \left[\hat{y}^n - \cancel{\hat{y}^n f_{w,b}(x^n)} - f_{w,b}(x^n) + \cancel{\hat{y}^n f_{w,b}(x^n)} \right] \underbrace{x_i^n}_{\textcolor{blue}{-}}$$

$$= \sum_n - \left(\hat{y}^n - f_{w,b}(x^n) \right) x_i^n$$

Larger difference,
larger update

$$w_i \leftarrow w_i - \eta \sum_n - \left(\hat{y}^n - f_{w,b}(x^n) \right) x_i^n$$

Logistic Regression

Step 1: $f_{w,b}(x) = \sigma\left(\sum_i w_i x_i + b\right)$

Output: between 0 and 1

Training data: (x^n, \hat{y}^n)

Step 2: \hat{y}^n : 1 for class 1, 0 for class 2

$$L(f) = \sum_n C(f(x^n), \hat{y}^n)$$

Linear Regression

$$f_{w,b}(x) = \sum_i w_i x_i + b$$

Output: any value

Training data: (x^n, \hat{y}^n)

\hat{y}^n : a real number

$$L(f) = \frac{1}{2} \sum_n (f(x^n) - \hat{y}^n)^2$$

Logistic regression: $w_i \leftarrow w_i - \eta \sum_n -(\hat{y}^n - f_{w,b}(x^n)) x_i^n$

Step 3:

Linear regression: $w_i \leftarrow w_i - \eta \sum_n -(\hat{y}^n - f_{w,b}(x^n)) x_i^n$

Logistic Regression + Square Error

Step 1: $f_{w,b}(x) = \sigma\left(\sum_i w_i x_i + b\right)$

Step 2: Training data: (x^n, \hat{y}^n) , \hat{y}^n : 1 for class 1, 0 for class 2

$$L(f) = \frac{1}{2} \sum_n (f_{w,b}(x^n) - \hat{y}^n)^2$$

Step 3:

$$\frac{\partial (f_{w,b}(x) - \hat{y})^2}{\partial w_i} = 2(f_{w,b}(x) - \hat{y}) \frac{\partial f_{w,b}(x)}{\partial z} \frac{\partial z}{\partial w_i}$$
$$= 2(f_{w,b}(x) - \hat{y}) f_{w,b}(x) (1 - f_{w,b}(x)) x_i$$

$\hat{y}^n = 1$ If $f_{w,b}(x^n) = 1$ (close to target) $\rightarrow \partial L / \partial w_i = 0$

If $f_{w,b}(x^n) = 0$ (far from target) $\rightarrow \partial L / \partial w_i = 0$

Logistic Regression + Square Error

Step 1: $f_{w,b}(x) = \sigma\left(\sum_i w_i x_i + b\right)$

Step 2: Training data: (x^n, \hat{y}^n) , \hat{y}^n : 1 for class 1, 0 for class 2

$$L(f) = \frac{1}{2} \sum_n (f_{w,b}(x^n) - \hat{y}^n)^2$$

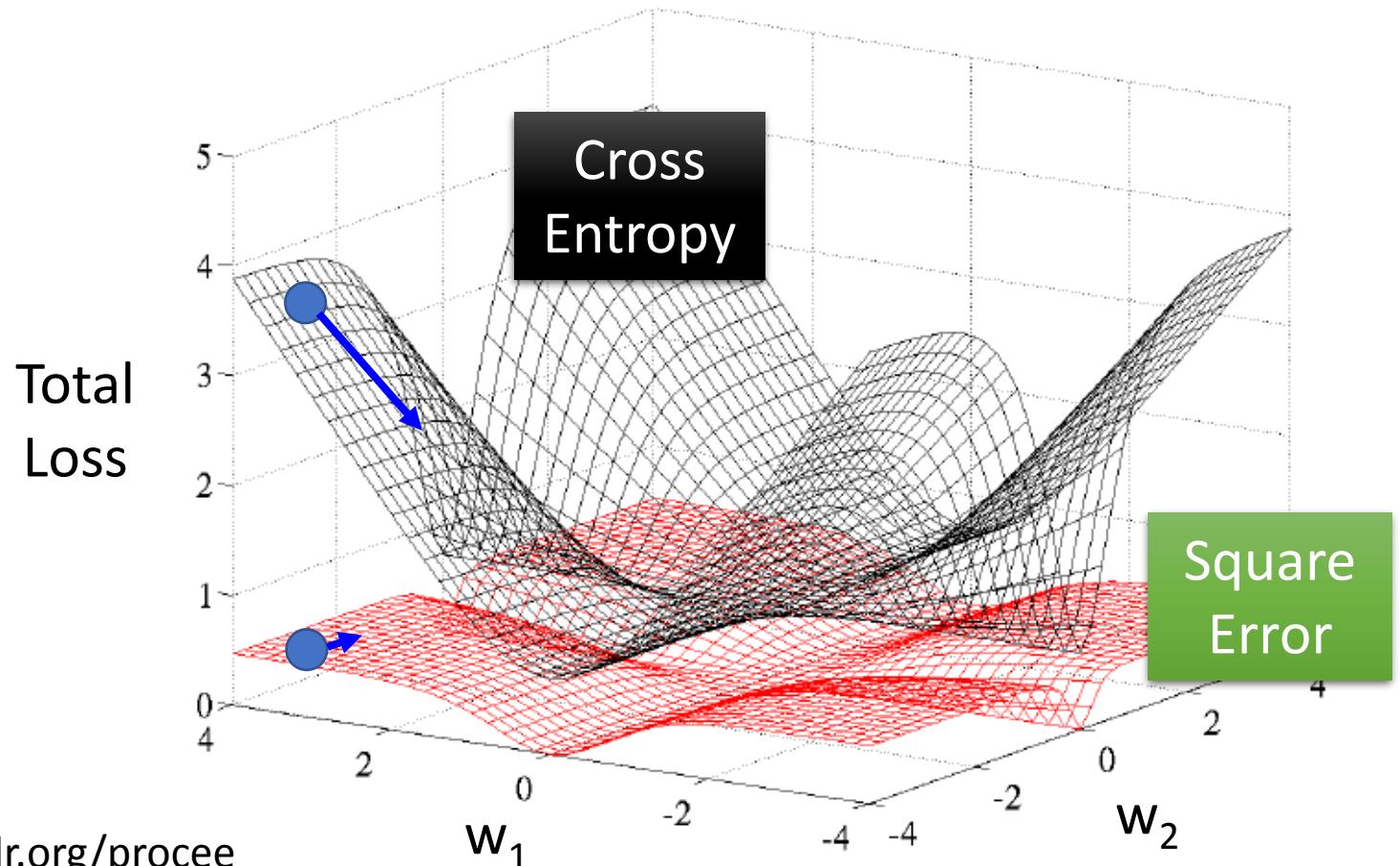
Step 3:

$$\frac{\partial (f_{w,b}(x) - \hat{y})^2}{\partial w_i} = 2(f_{w,b}(x) - \hat{y}) \frac{\partial f_{w,b}(x)}{\partial z} \frac{\partial z}{\partial w_i}$$
$$= 2(f_{w,b}(x) - \hat{y}) f_{w,b}(x) (1 - f_{w,b}(x)) x_i$$

$\hat{y}^n = 0$ If $f_{w,b}(x^n) = 1$ (far from target) $\rightarrow \partial L / \partial w_i = 0$

If $f_{w,b}(x^n) = 0$ (close to target) $\rightarrow \partial L / \partial w_i = 0$

Cross Entropy v.s. Square Error



<http://jmlr.org/proceedings/papers/v9/glorot10a/glorot10a.pdf>

Discriminative v.s. Generative

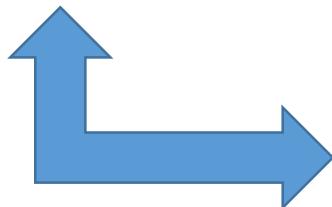
$$P(C_1|x) = \sigma(w \cdot x + b)$$



directly find w and b



Find $\mu^1, \mu^2, \Sigma^{-1}$



Will we obtain the same set of w and b ?

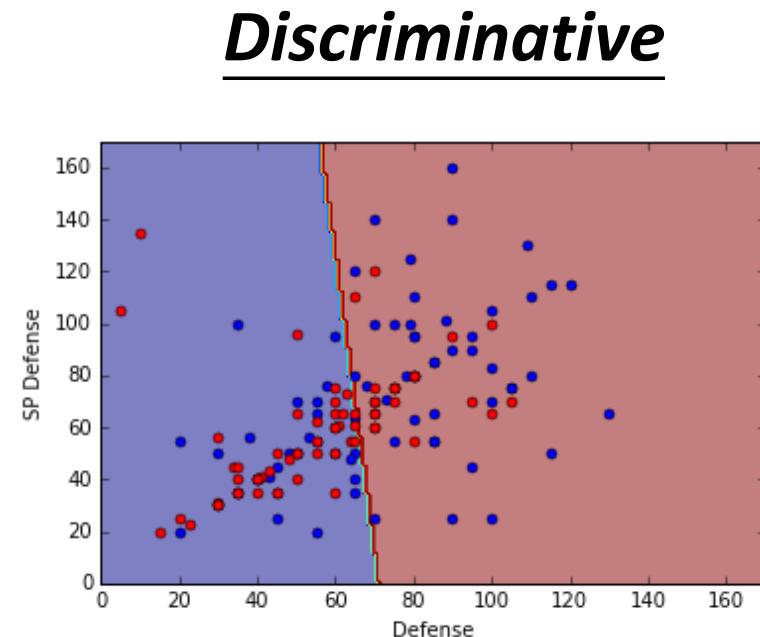
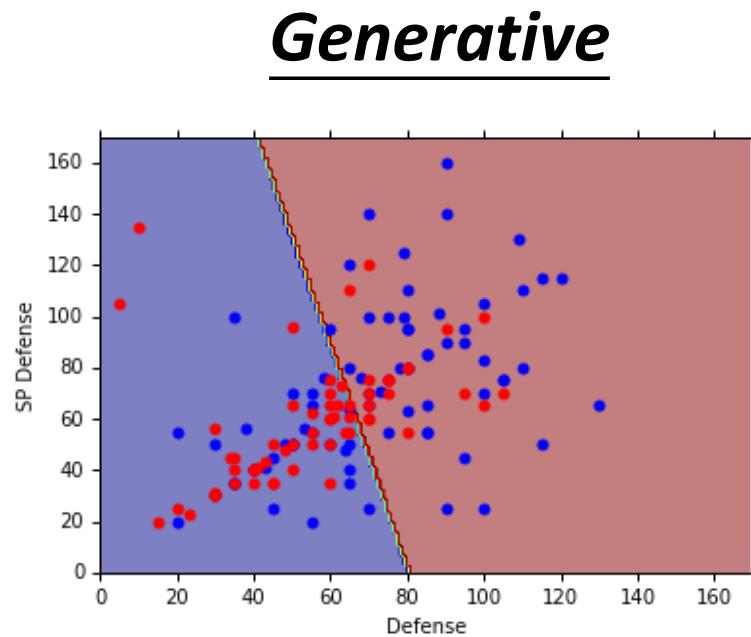
$$w^T = (\mu^1 - \mu^2)^T \Sigma^{-1}$$

$$b = -\frac{1}{2}(\mu^1)^T (\Sigma^1)^{-1} \mu^1$$

$$+ \frac{1}{2}(\mu^2)^T (\Sigma^2)^{-1} \mu^2 + \ln \frac{N_1}{N_2}$$

The same model (function set), but different function is selected by the same training data.

Generative v.s. Discriminative



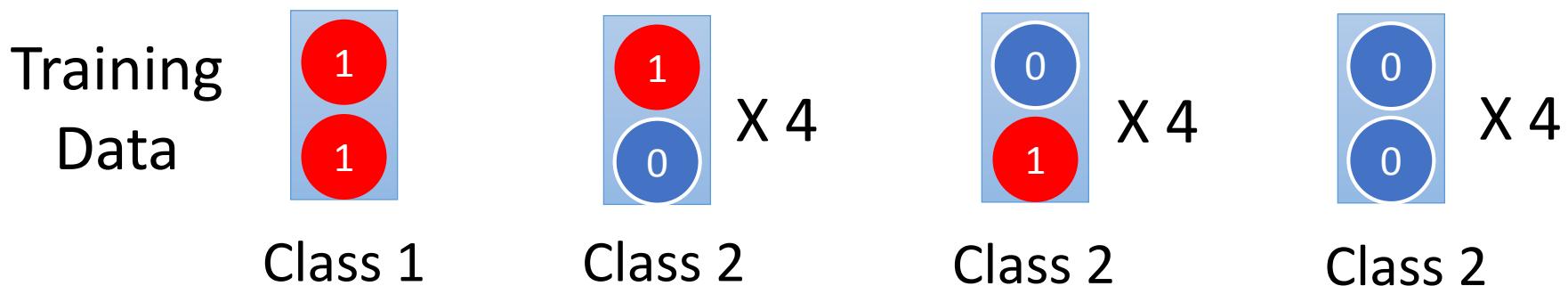
All: total, hp, att, sp att, de, sp de, speed

73% accuracy

79% accuracy

Generative v.s. Discriminative

- Example

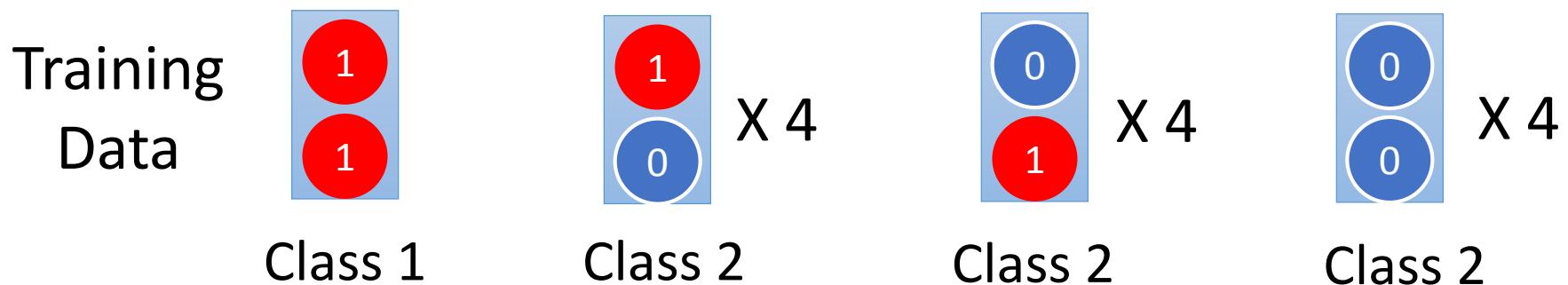


Testing Data  Class 1?
Class 2?

How about Naïve Bayes?
 $P(x|C_i) = P(x_1|C_i)P(x_2|C_i)$

Generative v.s. Discriminative

- Example



$$P(C_1) = \frac{1}{13}$$

$$P(x_1 = 1|C_1) = 1$$

$$P(x_2 = 1|C_1) = 1$$

$$P(C_2) = \frac{12}{13}$$

$$P(x_1 = 1|C_2) = \frac{1}{3}$$

$$P(x_2 = 1|C_2) = \frac{1}{3}$$

Training
Data



Class 1



X 4



X 4



X 4

Testing
Data



<0.5

$$P(C_1|x) = \frac{P(x|C_1)P(C_1)}{P(x|C_1)P(C_1) + P(x|C_2)P(C_2)}$$

1 × 1 $\frac{1}{13}$
 1 × 1 $\frac{1}{13}$
 $\frac{1}{3} \times \frac{1}{3}$ $\frac{12}{13}$

$$P(C_1) = \frac{1}{13}$$

$$P(x_1 = 1|C_1) = 1$$

$$P(x_2 = 1|C_1) = 1$$

$$P(C_2) = \frac{12}{13}$$

$$P(x_1 = 1|C_2) = \frac{1}{3}$$

$$P(x_2 = 1|C_2) = \frac{1}{3}$$

Generative v.s. Discriminative

- Benefit of generative model
 - With the assumption of probability distribution, less training data is needed
 - With the assumption of probability distribution, more robust to the noise
 - Priors and class-dependent probabilities can be estimated from different sources.

Multi-class Classification

(3 classes as example)

$$C_1: w^1, b_1 \quad z_1 = w^1 \cdot x + b_1$$

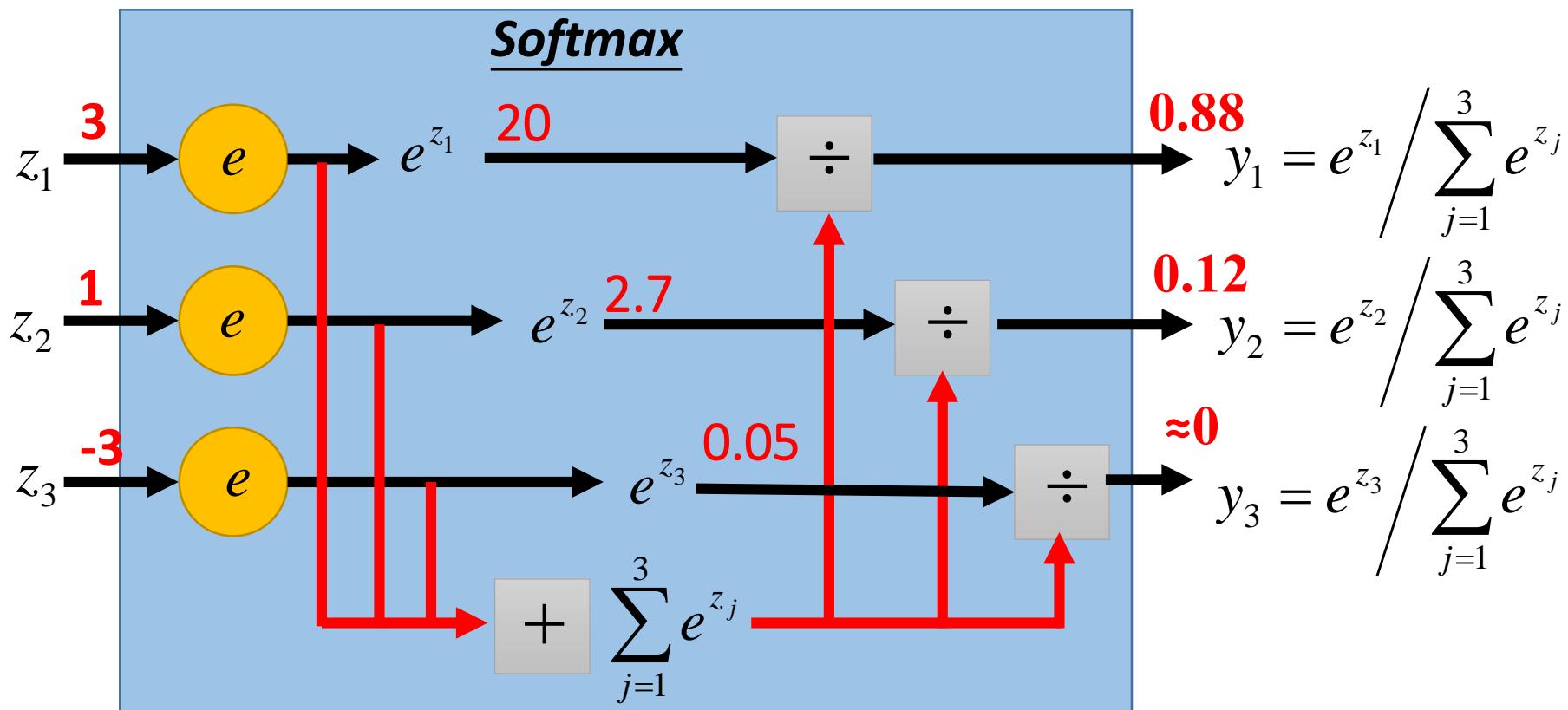
$$C_2: w^2, b_2 \quad z_2 = w^2 \cdot x + b_2$$

$$C_3: w^3, b_3 \quad z_3 = w^3 \cdot x + b_3$$

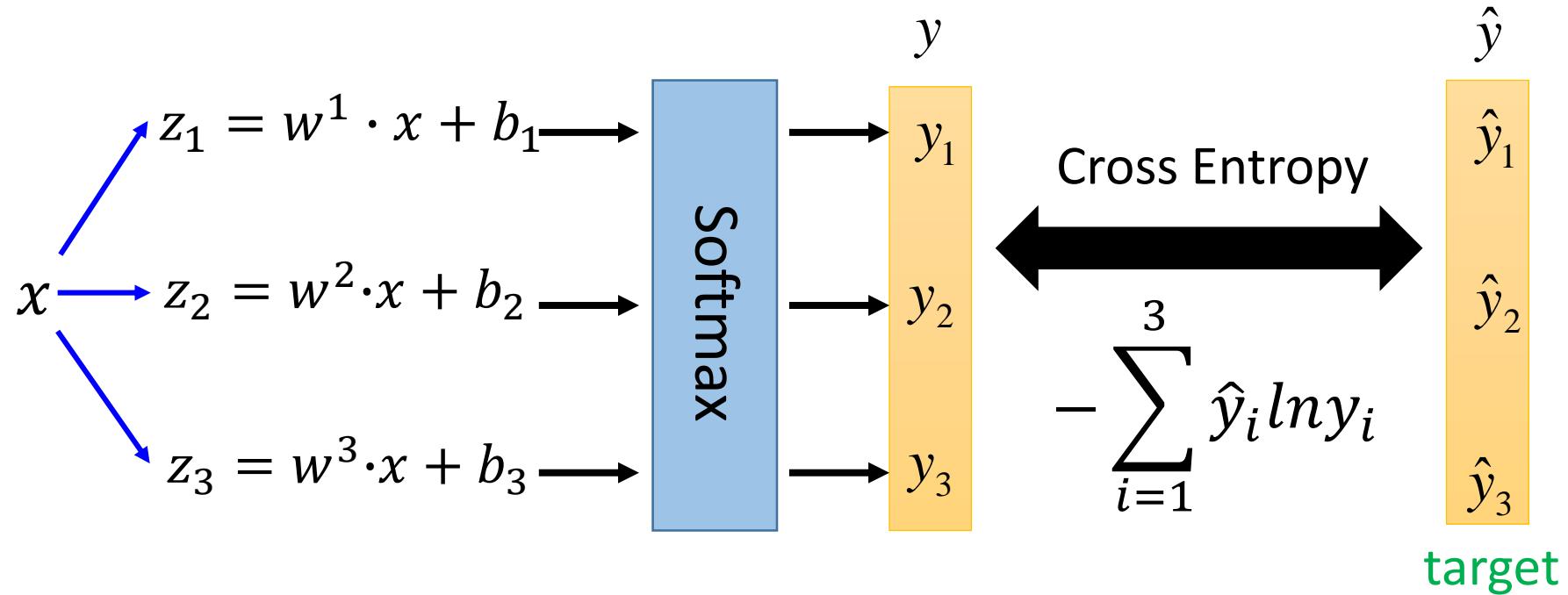
Probability:

- $1 > y_i > 0$
- $\sum_i y_i = 1$

$$y_i = P(C_i | x)$$



Multi-class Classification

 (3 classes as example)

If $x \in$ class 1

$$\hat{y} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

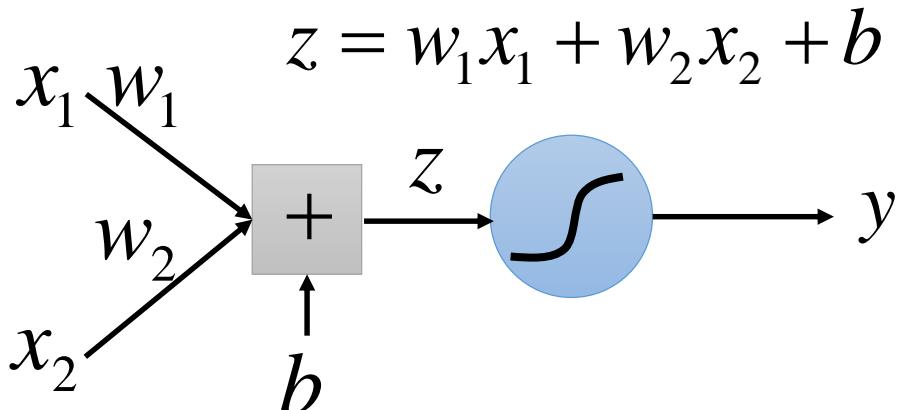
If $x \in$ class 2

$$\hat{y} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

If $x \in$ class 3

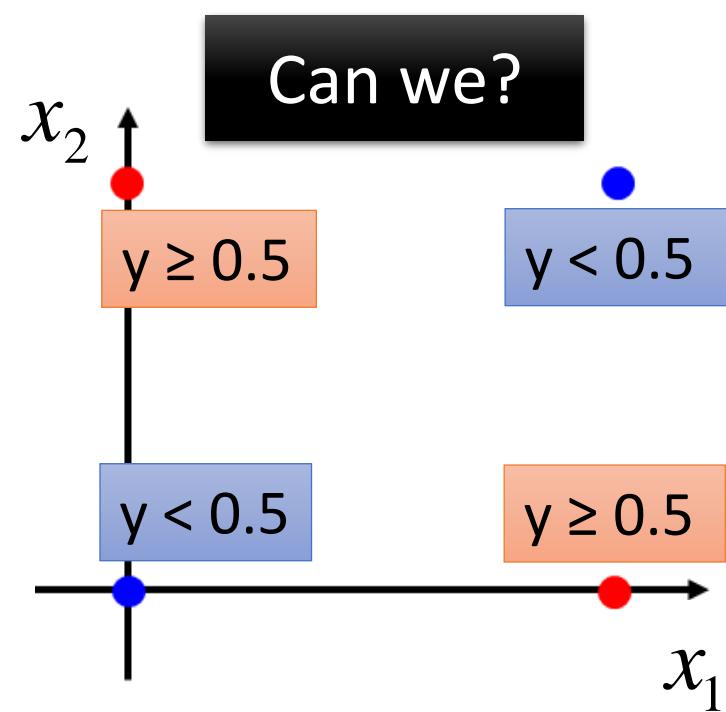
$$\hat{y} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

Limitation of Logistic Regression



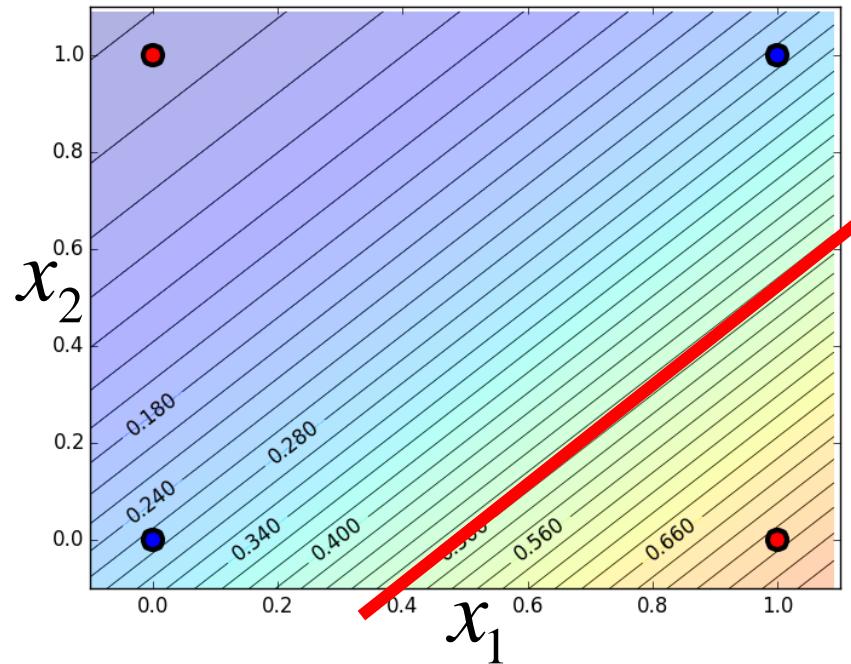
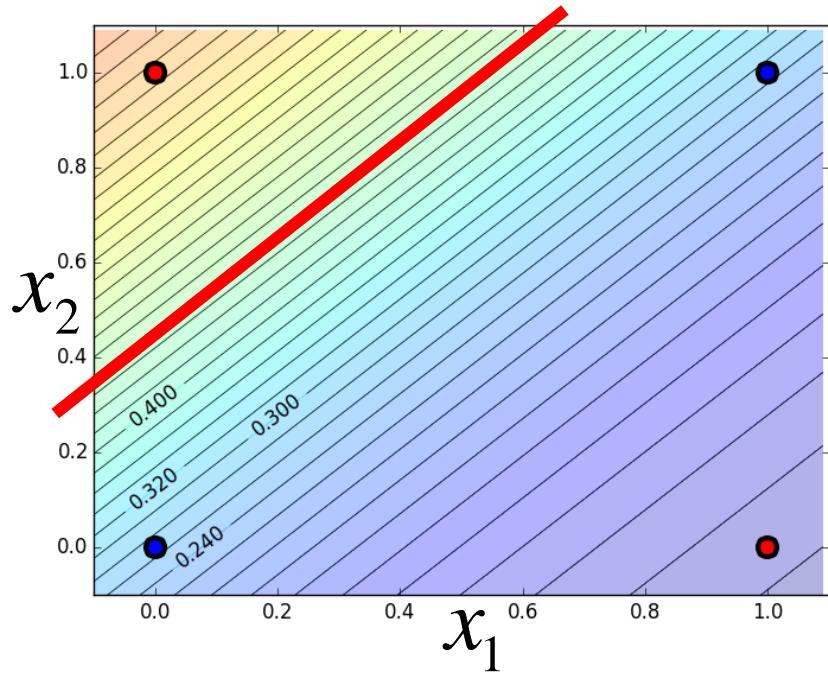
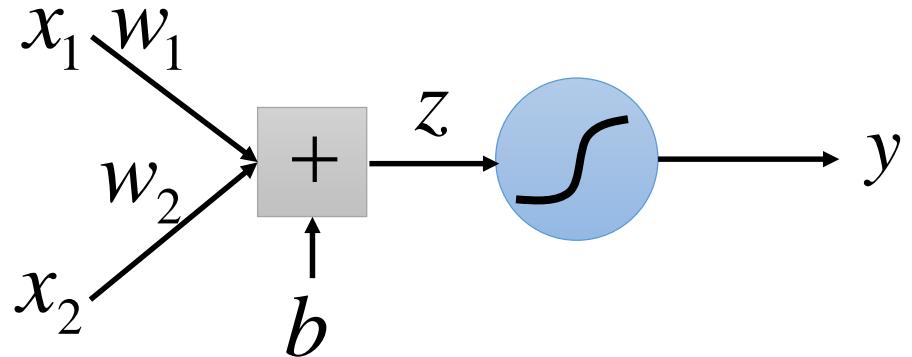
$$\begin{cases} \text{Class1} & y \geq 0.5 \\ \text{Class2} & y < 0.5 \end{cases}$$

Input Feature		Label
x_1	x_2	
0	0	Class 2
0	1	Class 1
1	0	Class 1
1	1	Class 2



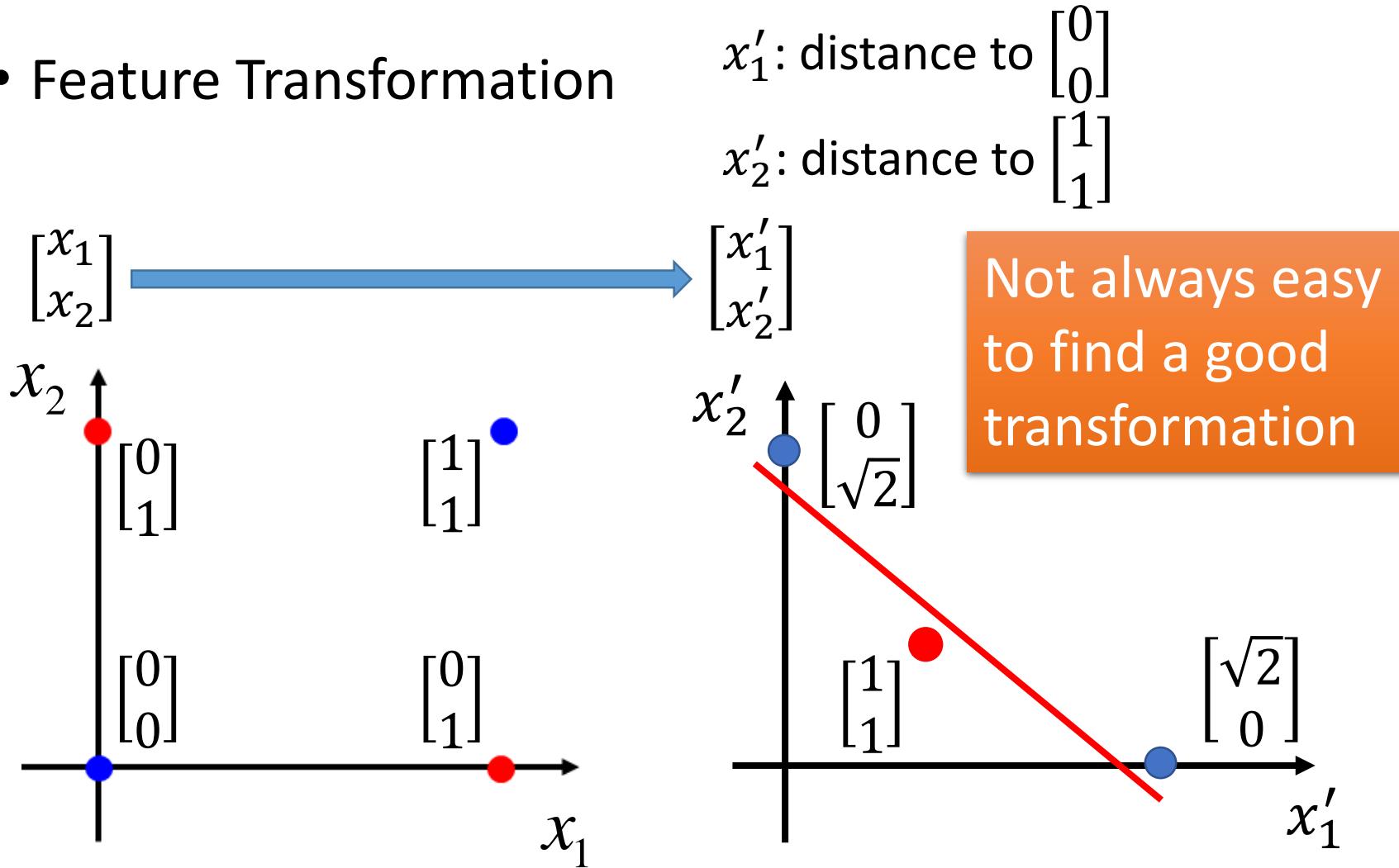
Limitation of Logistic Regression

- No, we can't



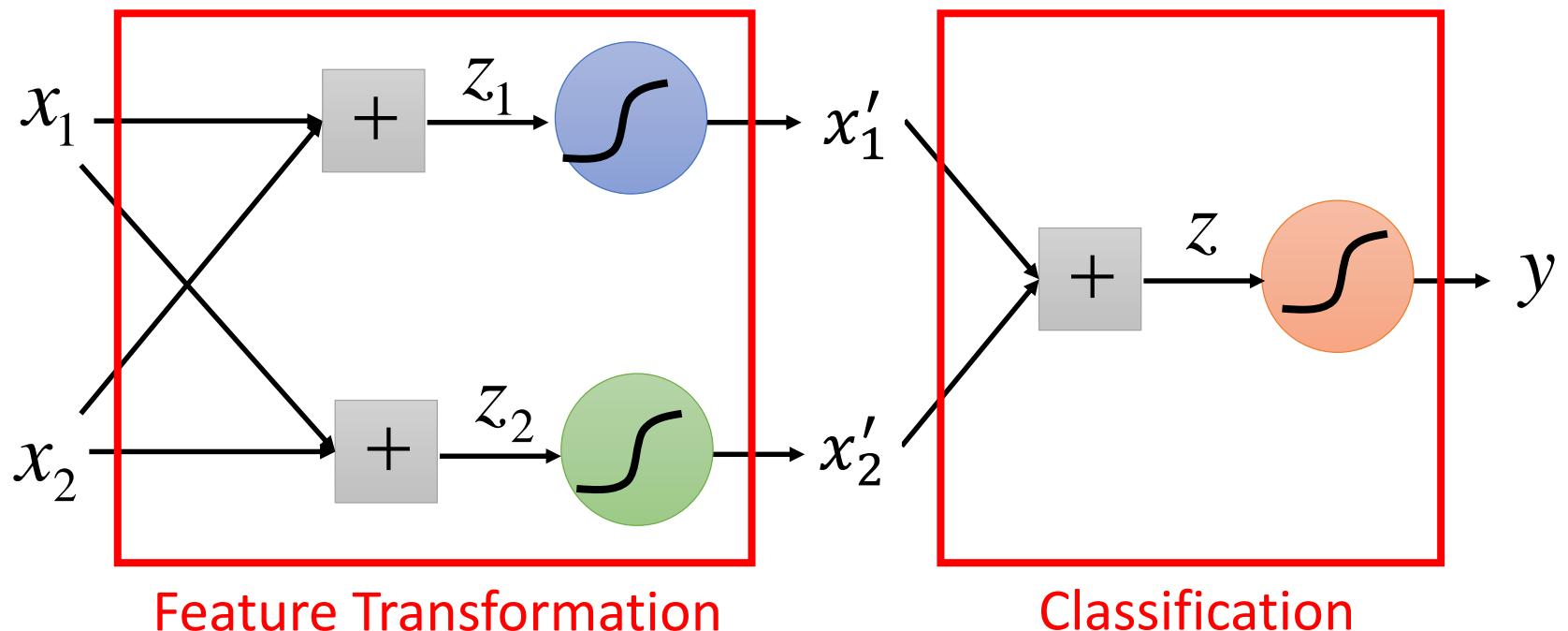
Limitation of Logistic Regression

- Feature Transformation

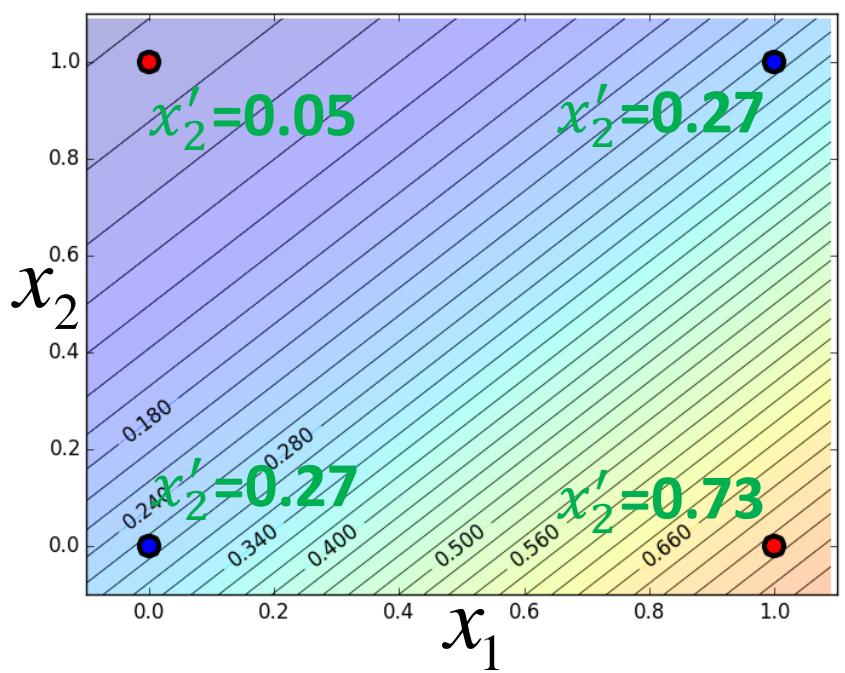
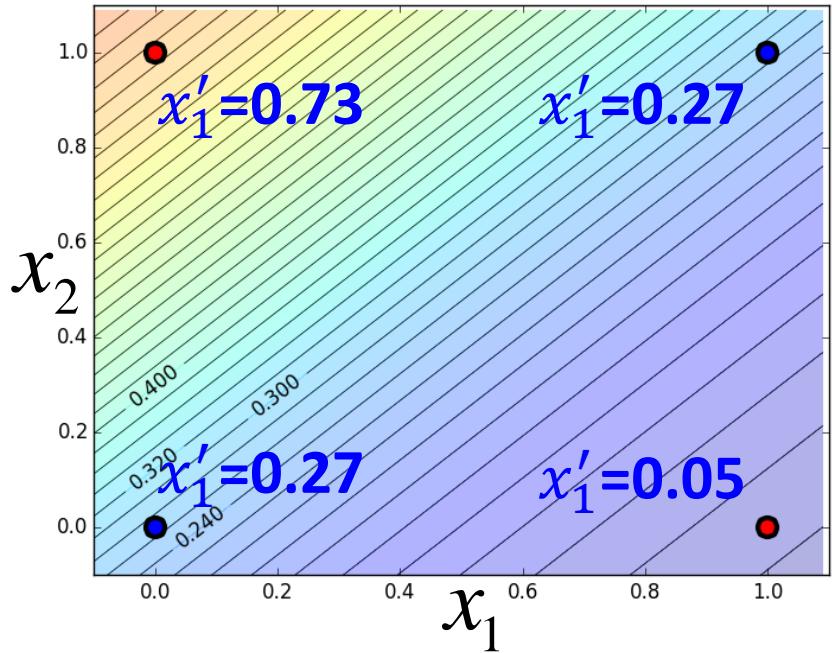
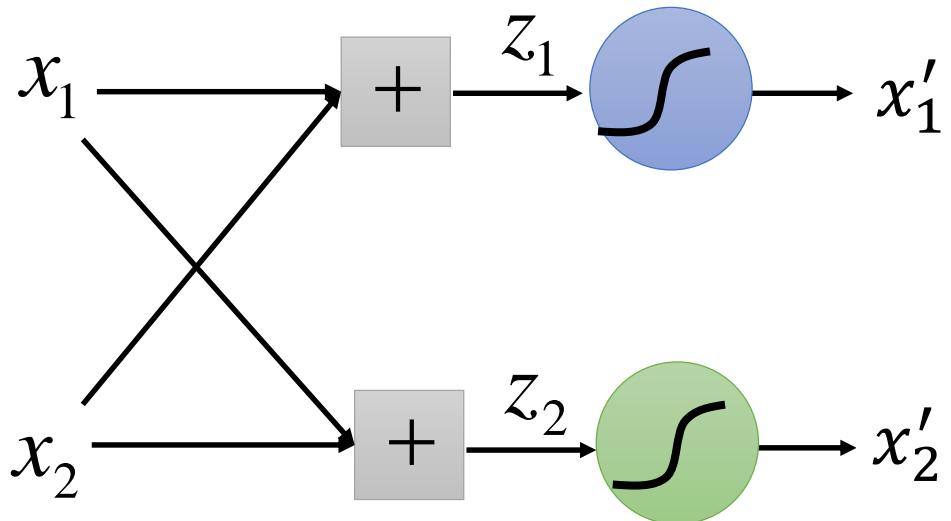


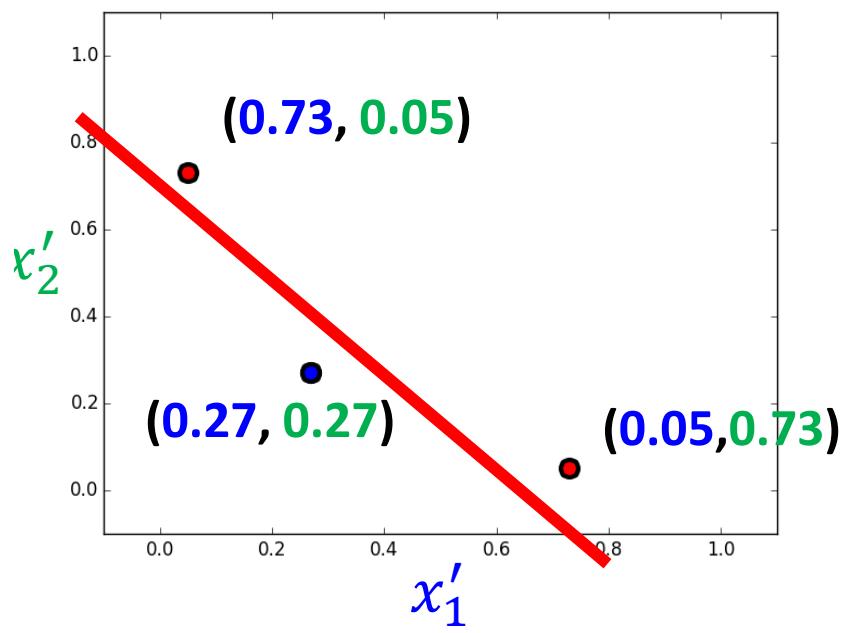
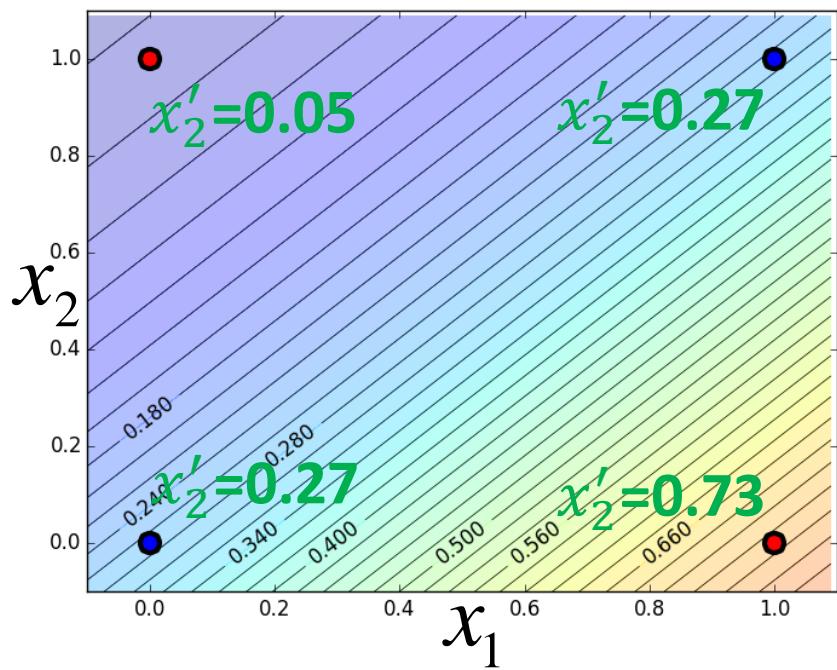
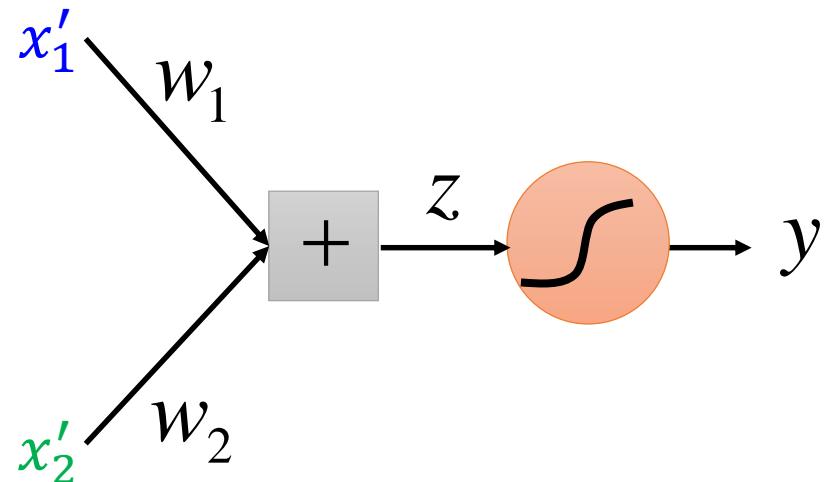
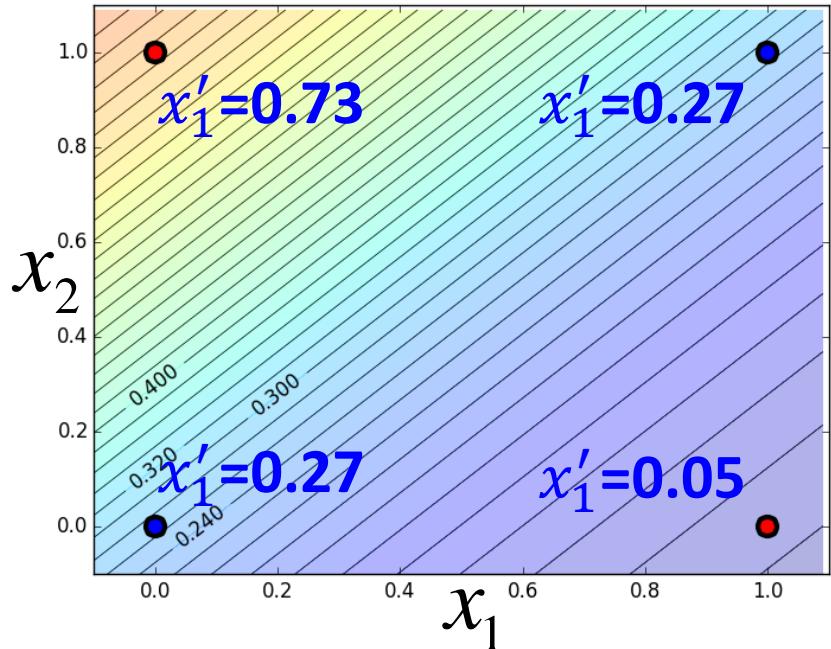
Limitation of Logistic Regression

- Cascading logistic regression models

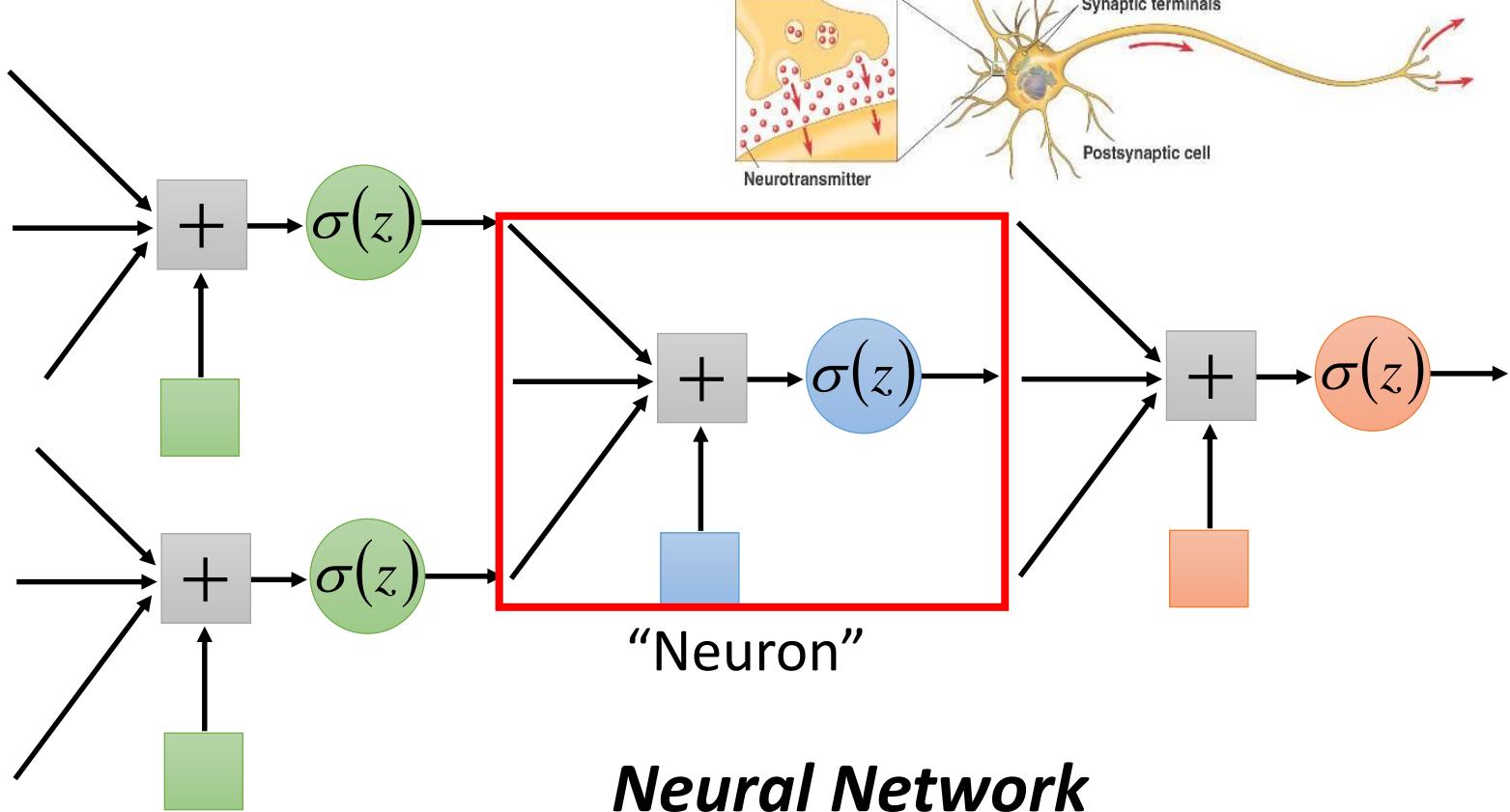
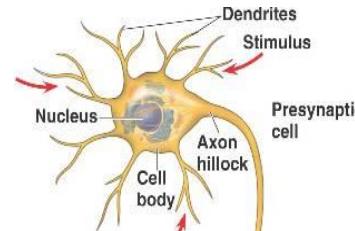


(ignore bias in this figure)





Deep Learning!



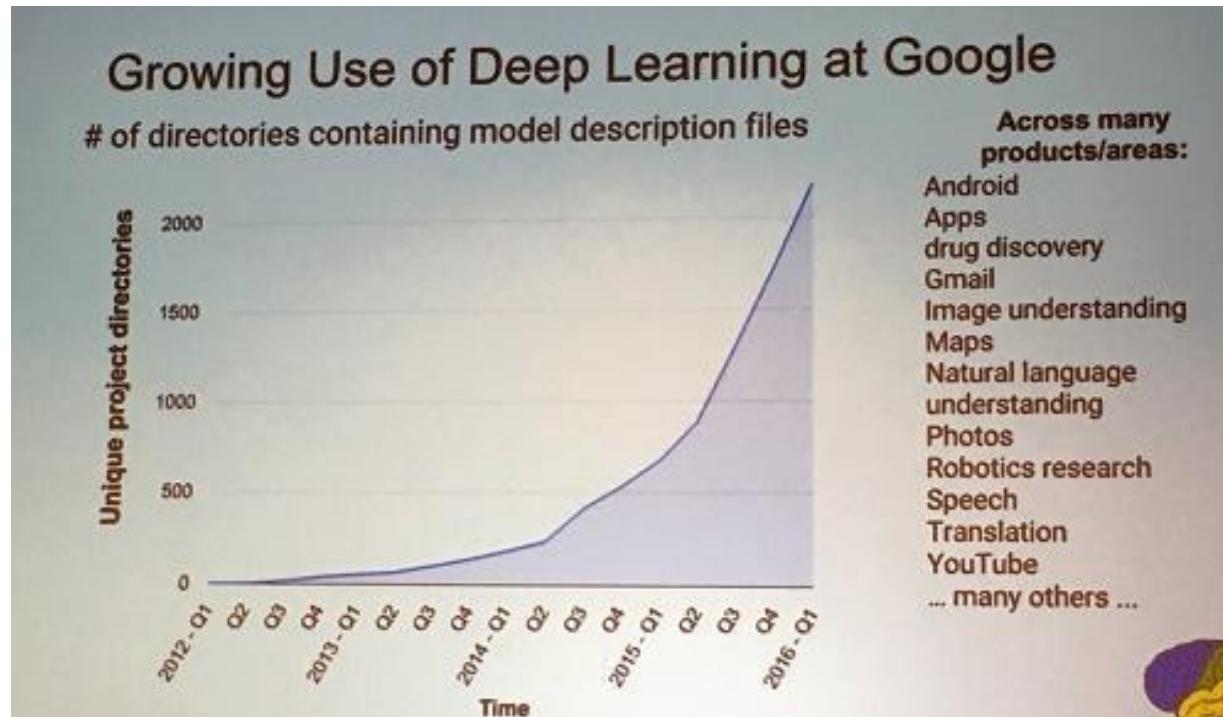
Reference

- Bishop: Chapter 4.3

Deep Learning

Deep learning attracts lots of attention.

- I believe you have seen lots of exciting results before.



Deep learning trends at Google. Source: SIGMOD/Jeff Dean

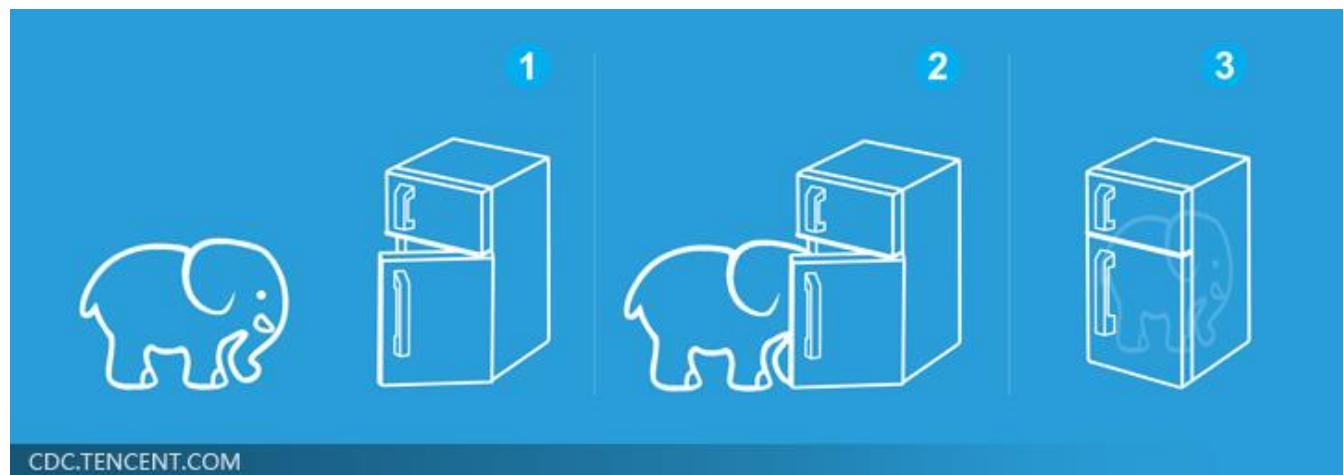
Ups and downs of Deep Learning

- 1958: Perceptron (linear model)
- 1969: Perceptron has limitation
- 1980s: Multi-layer perceptron
 - Do not have significant difference from DNN today
- 1986: Backpropagation
 - Usually more than 3 hidden layers is not helpful
- 1989: 1 hidden layer is “good enough”, why deep?
- 2006: RBM initialization (breakthrough)
- 2009: GPU
- 2011: Start to be popular in speech recognition
- 2012: win ILSVRC image competition

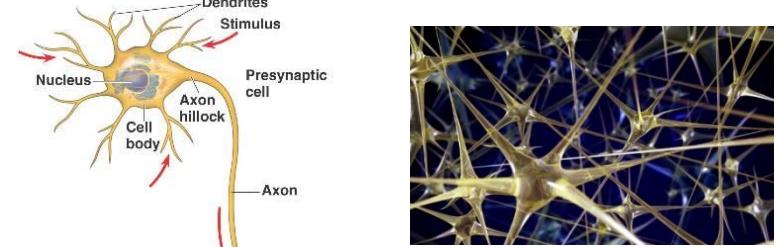
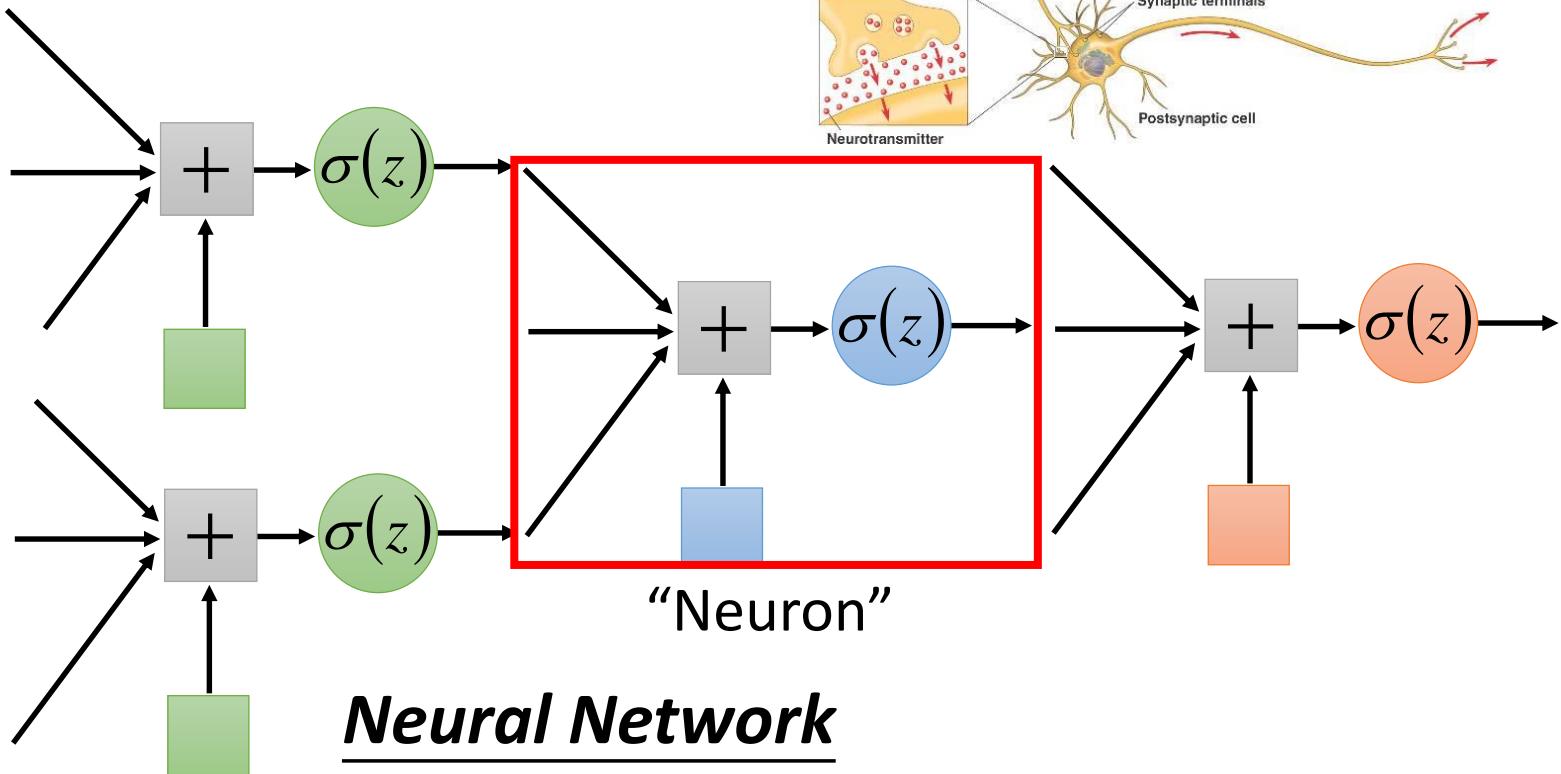
Three Steps for Deep Learning



Deep Learning is so simple



Neural Network

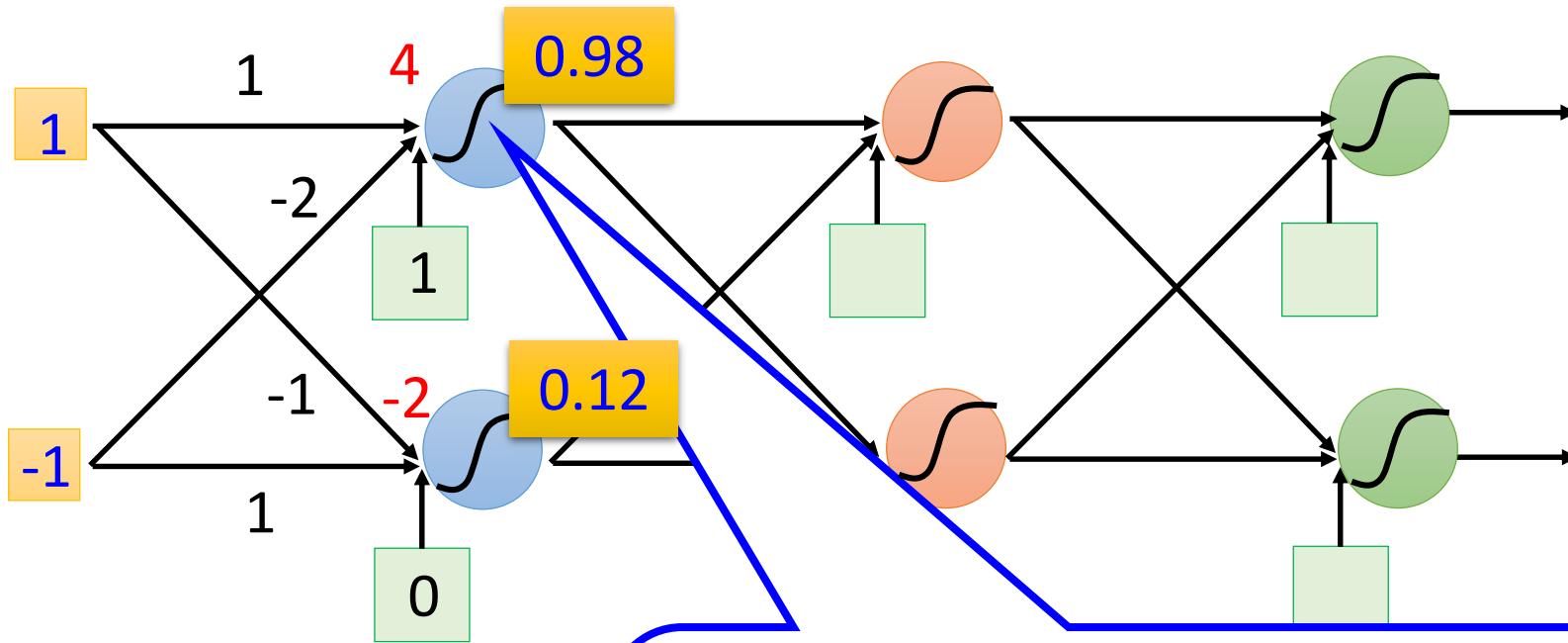


Neural Network

Different connection leads to different network structures

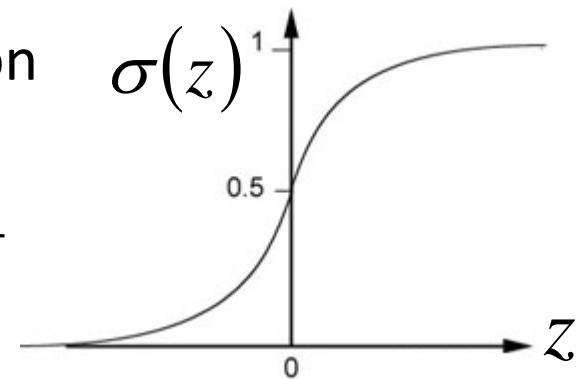
Network parameter θ : all the weights and biases in the “neurons”

Fully Connect Feedforward Network

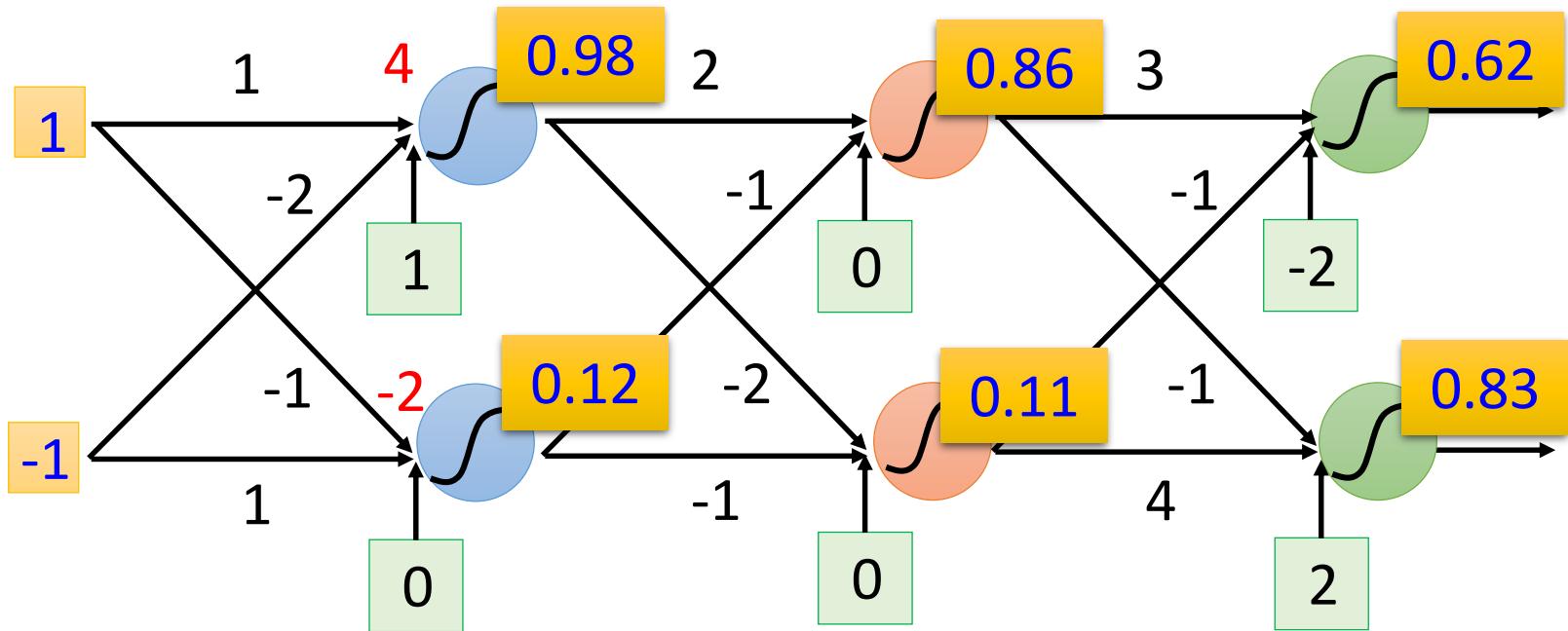


Sigmoid Function

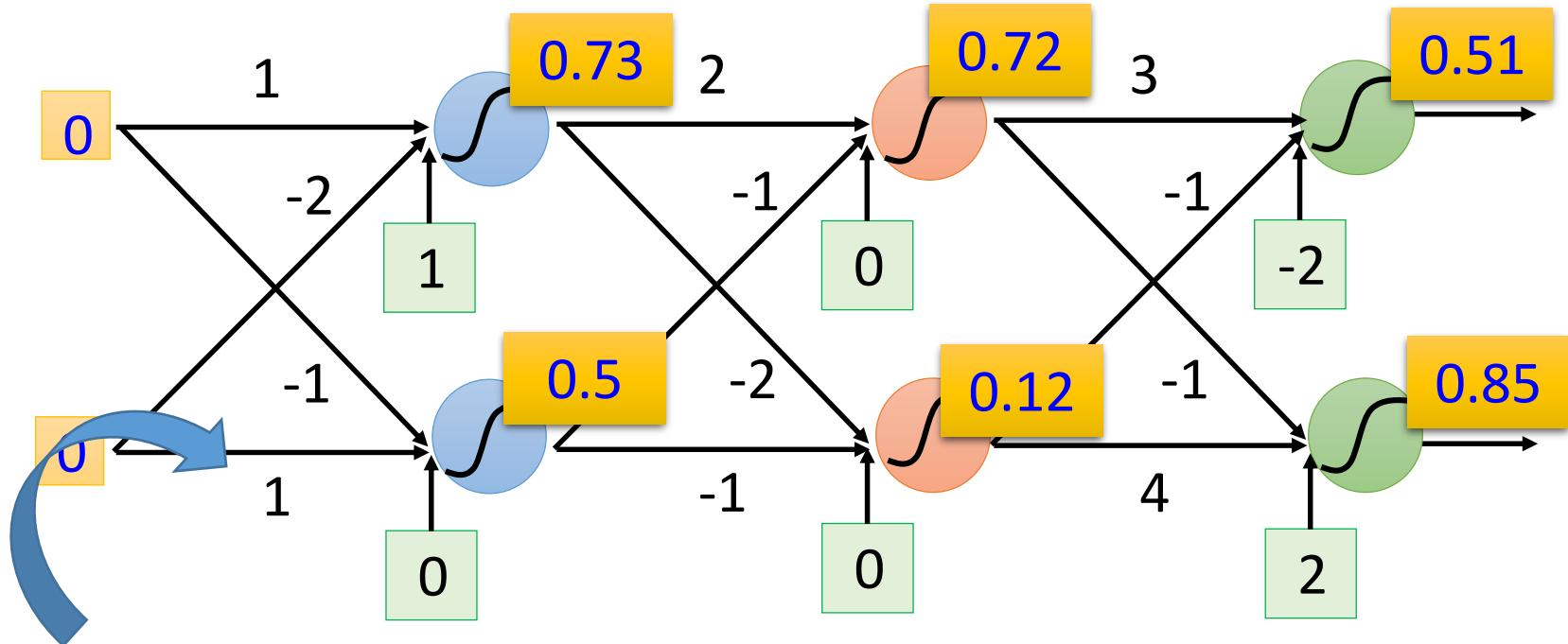
$$\sigma(z) = \frac{1}{1 + e^{-z}}$$



Fully Connect Feedforward Network



Fully Connect Feedforward Network

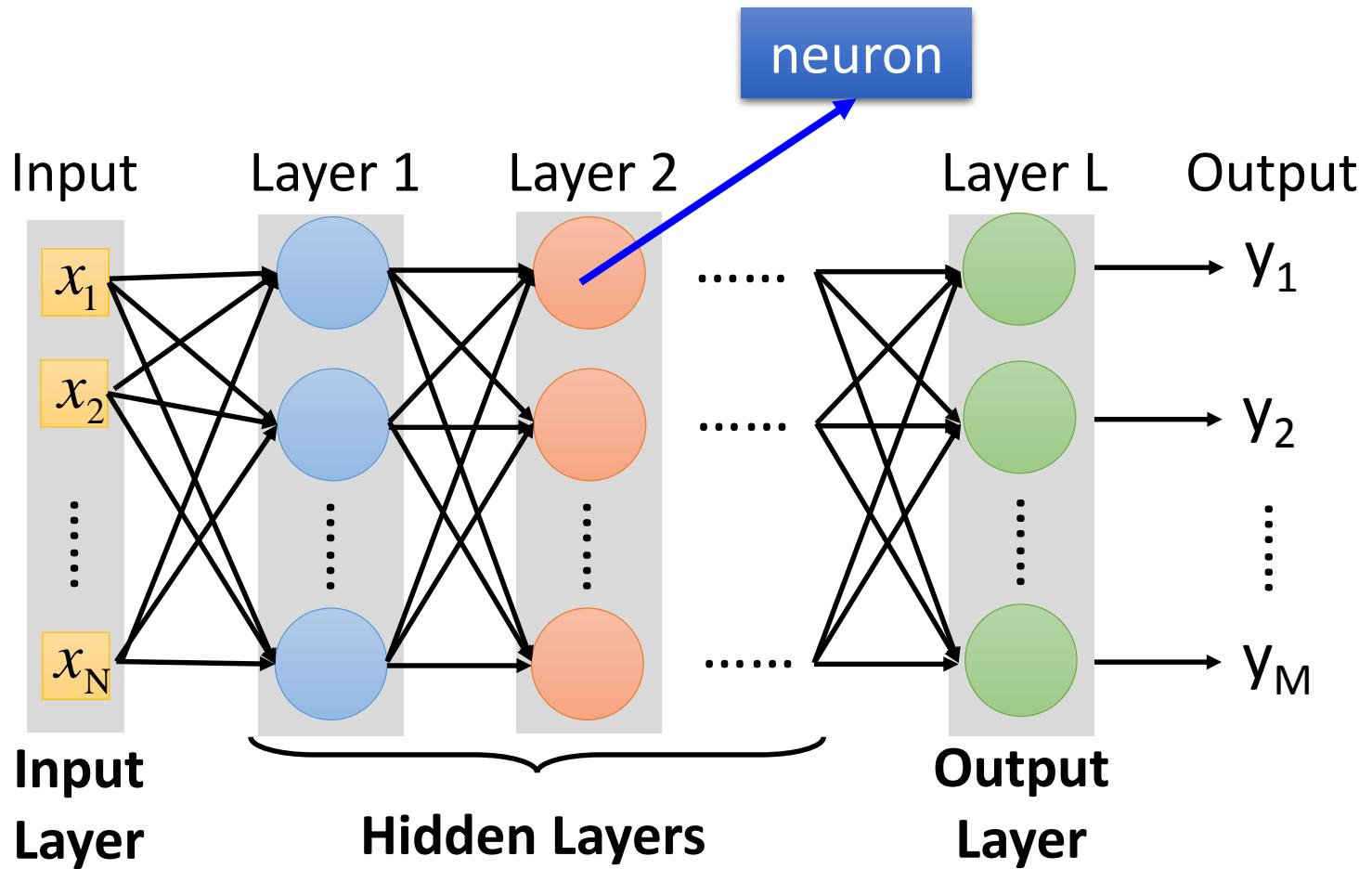


This is a function.
Input vector, output vector

$$f \left(\begin{bmatrix} 1 \\ -1 \end{bmatrix} \right) = \begin{bmatrix} 0.62 \\ 0.83 \end{bmatrix} \quad f \left(\begin{bmatrix} 0 \\ 0 \end{bmatrix} \right) = \begin{bmatrix} 0.51 \\ 0.85 \end{bmatrix}$$

Given network structure, define a function set

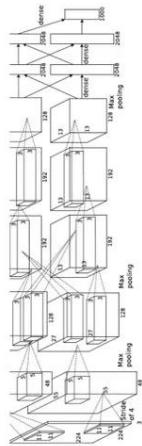
Fully Connect Feedforward Network



Deep = Many hidden layers

http://cs231n.stanford.edu/slides/winter1516_lecuture8.pdf

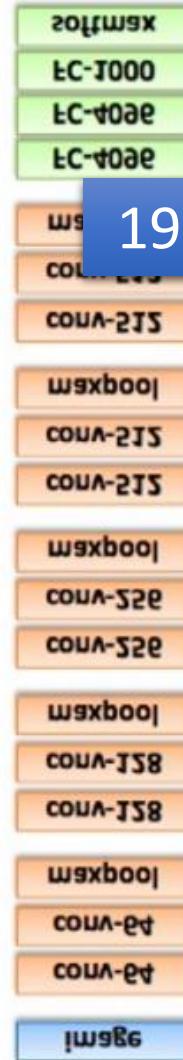
16.4%



AlexNet (2012)

8 layers

7.3%

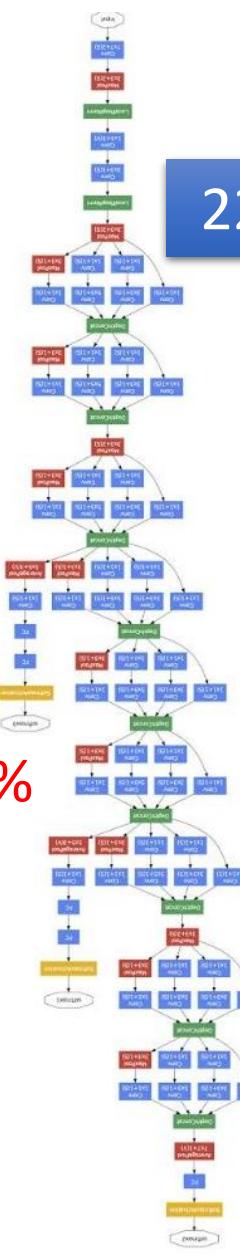


VGG (2014)

19 layers

6.7%

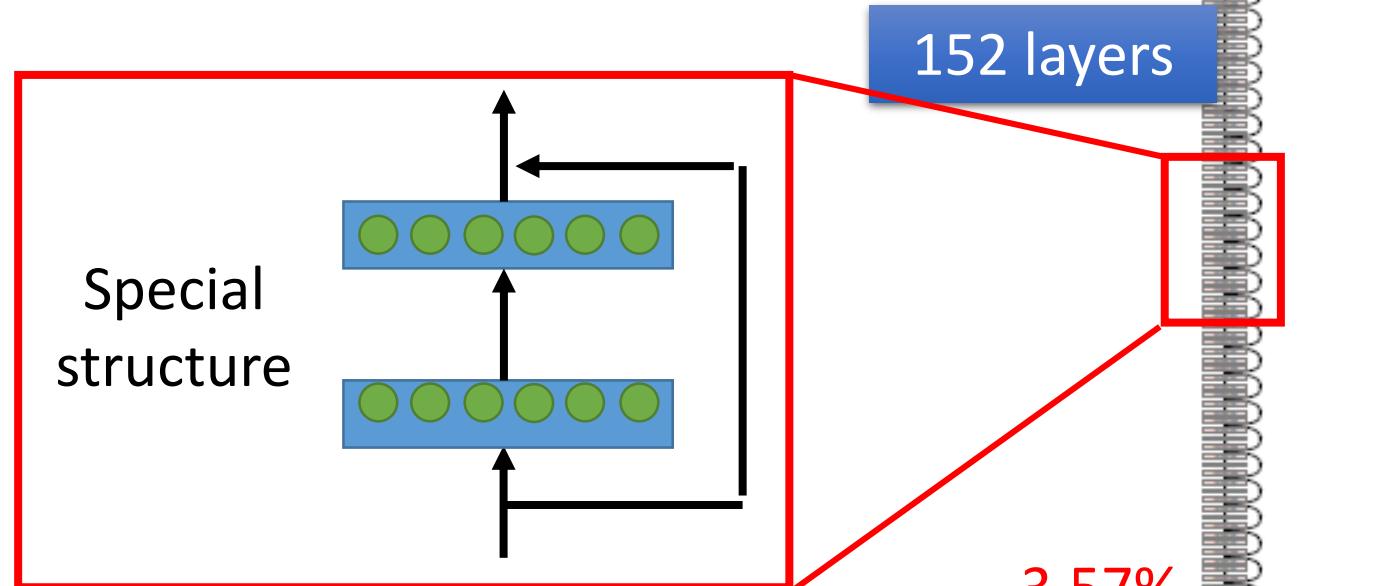
(source)



GoogleNet (2014)

22 layers

Deep = Many hidden layers



16.4%



AlexNet
(2012)

7.3%



VGG
(2014)

6.7%



GoogleNet
(2014)

Residual Net
(2015)

3.57%

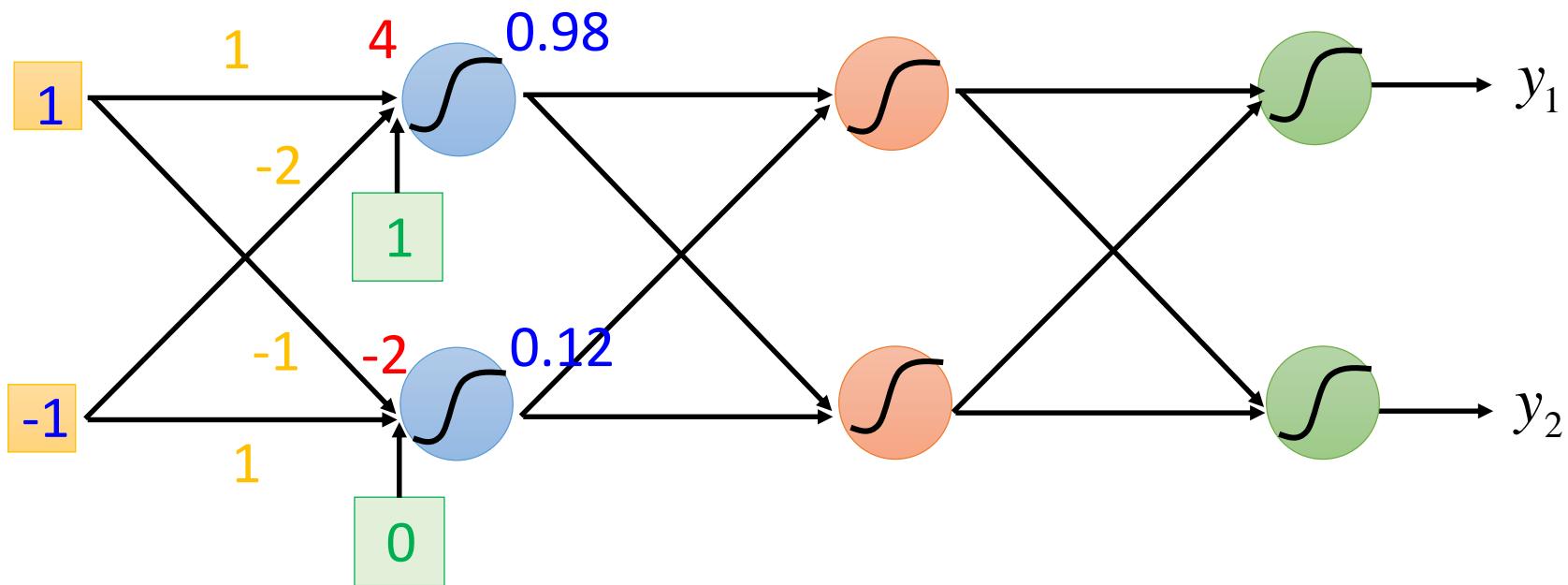
152 layers

101 layers



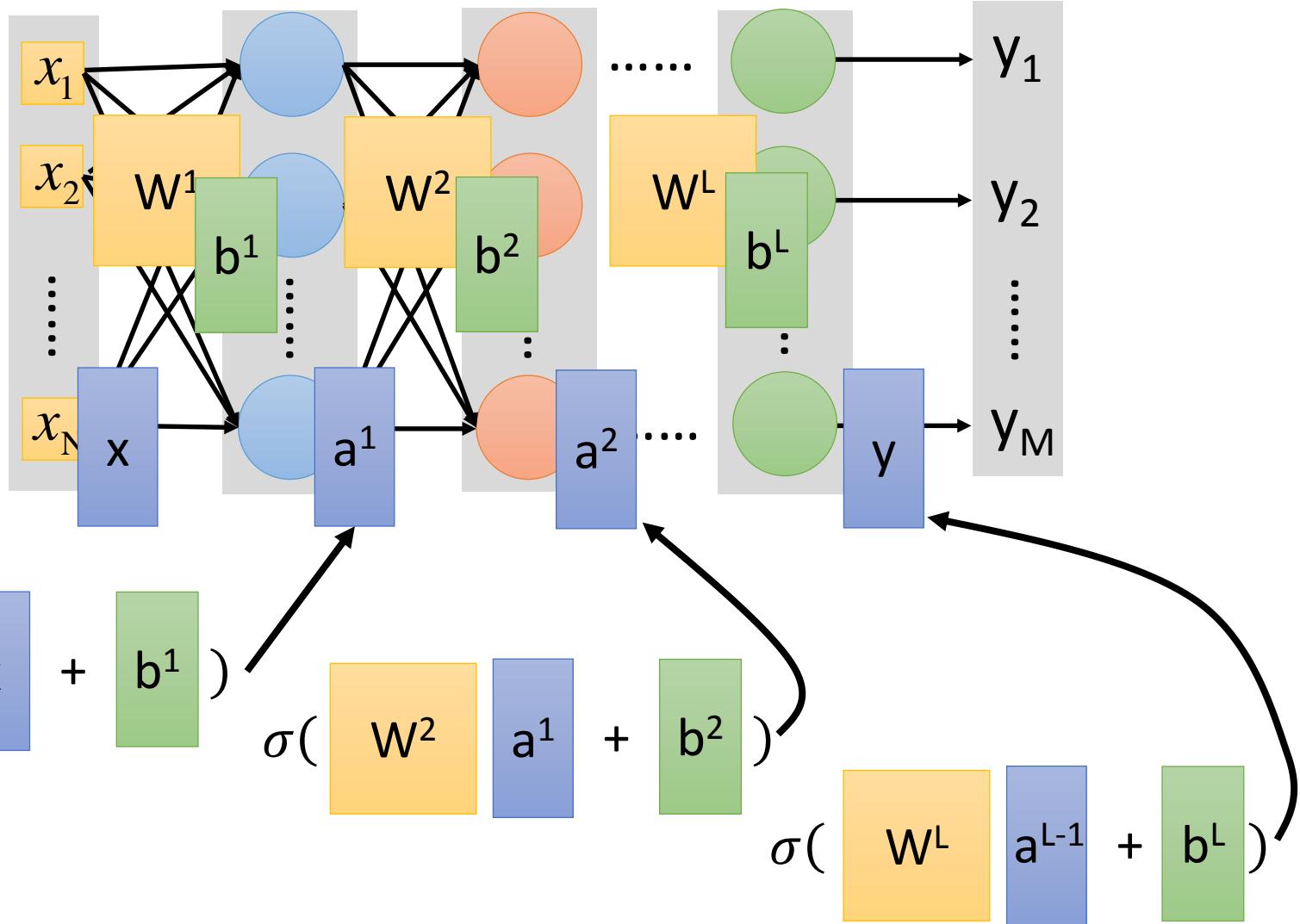
Taipei
101

Matrix Operation

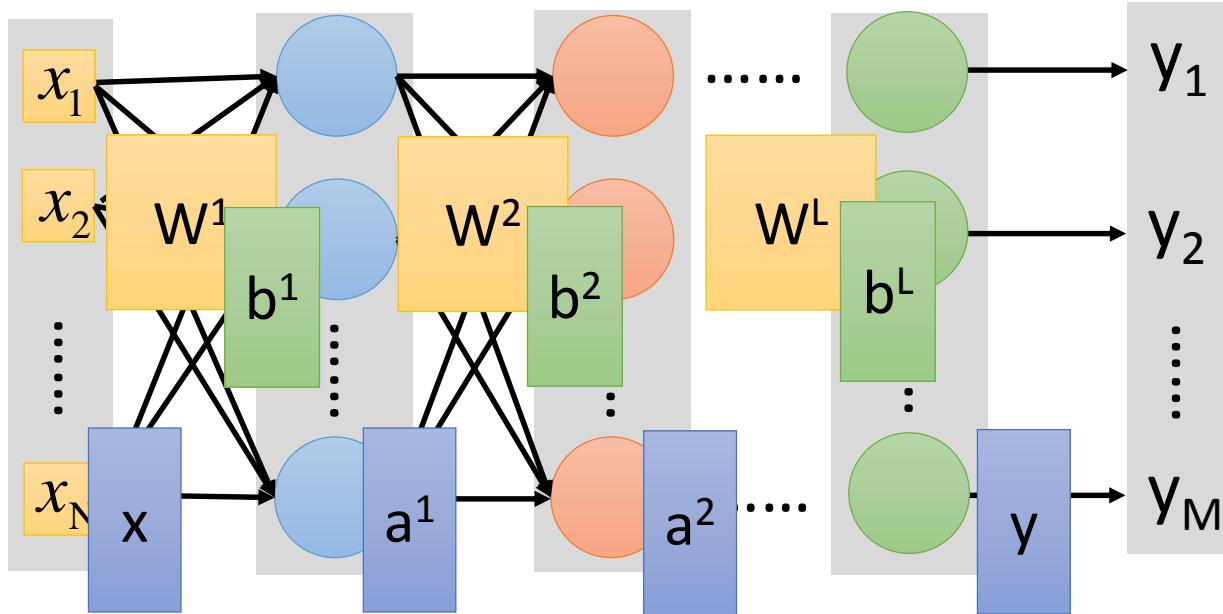


$$\sigma \left(\underbrace{\begin{bmatrix} 1 & -2 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix}}_{\begin{bmatrix} 4 \\ -2 \end{bmatrix}} \right) = \begin{bmatrix} 0.98 \\ 0.12 \end{bmatrix}$$

Neural Network



Neural Network



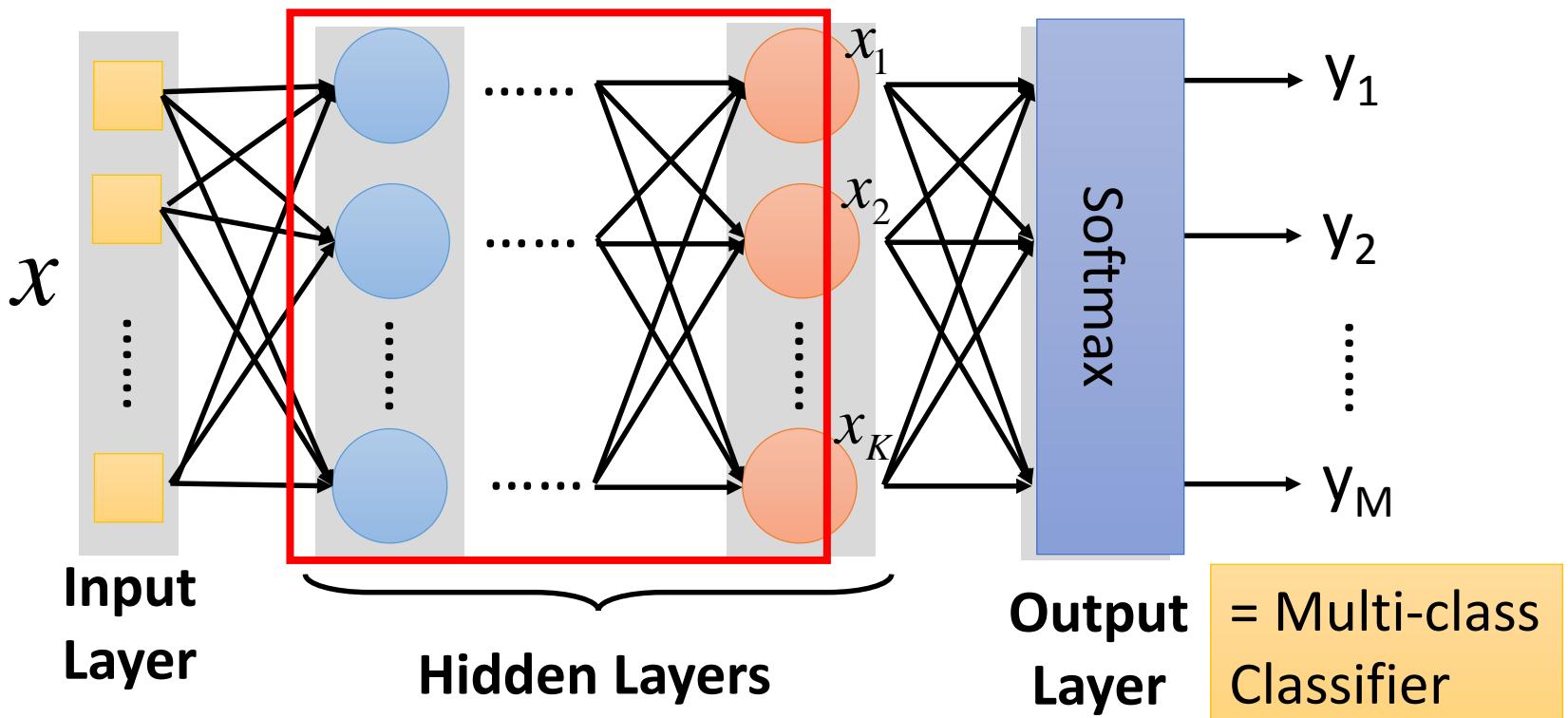
$$y = f(x)$$

Using parallel computing techniques
to speed up matrix operation

$$= \sigma(W^L \dots \sigma(W^2 \sigma(W^1 x + b^1) + b^2) \dots + b^L)$$

Output Layer

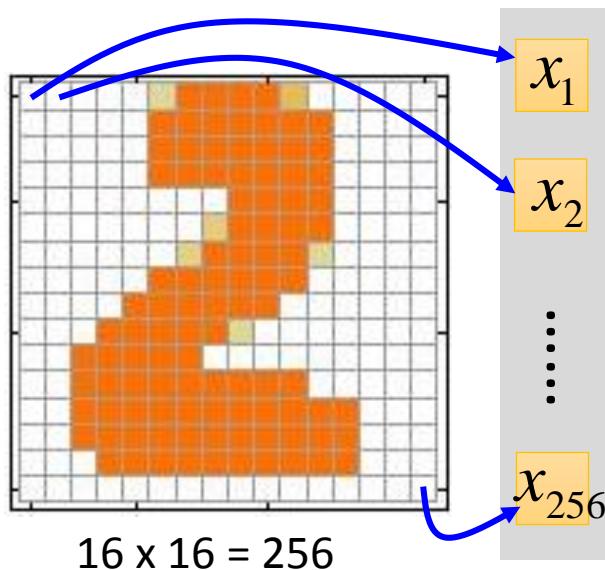
Feature extractor replacing
feature engineering



Example Application



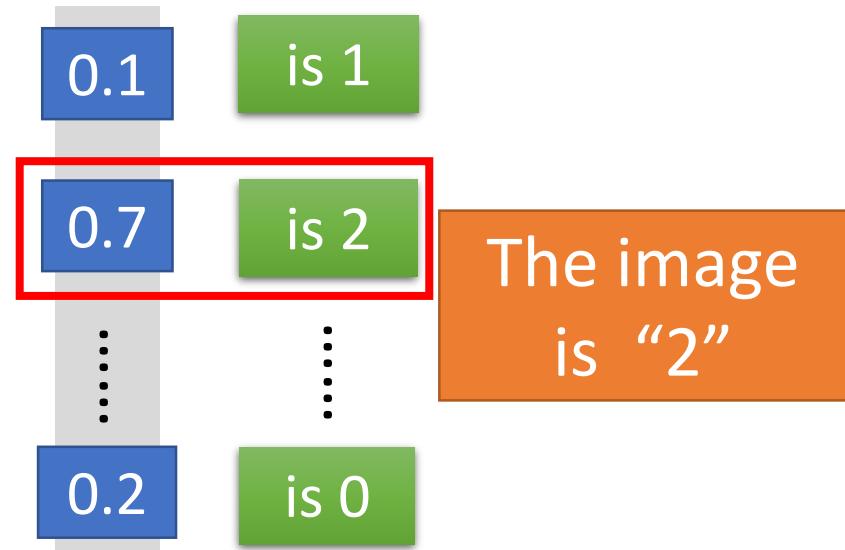
Input



Ink → 1

No ink → 0

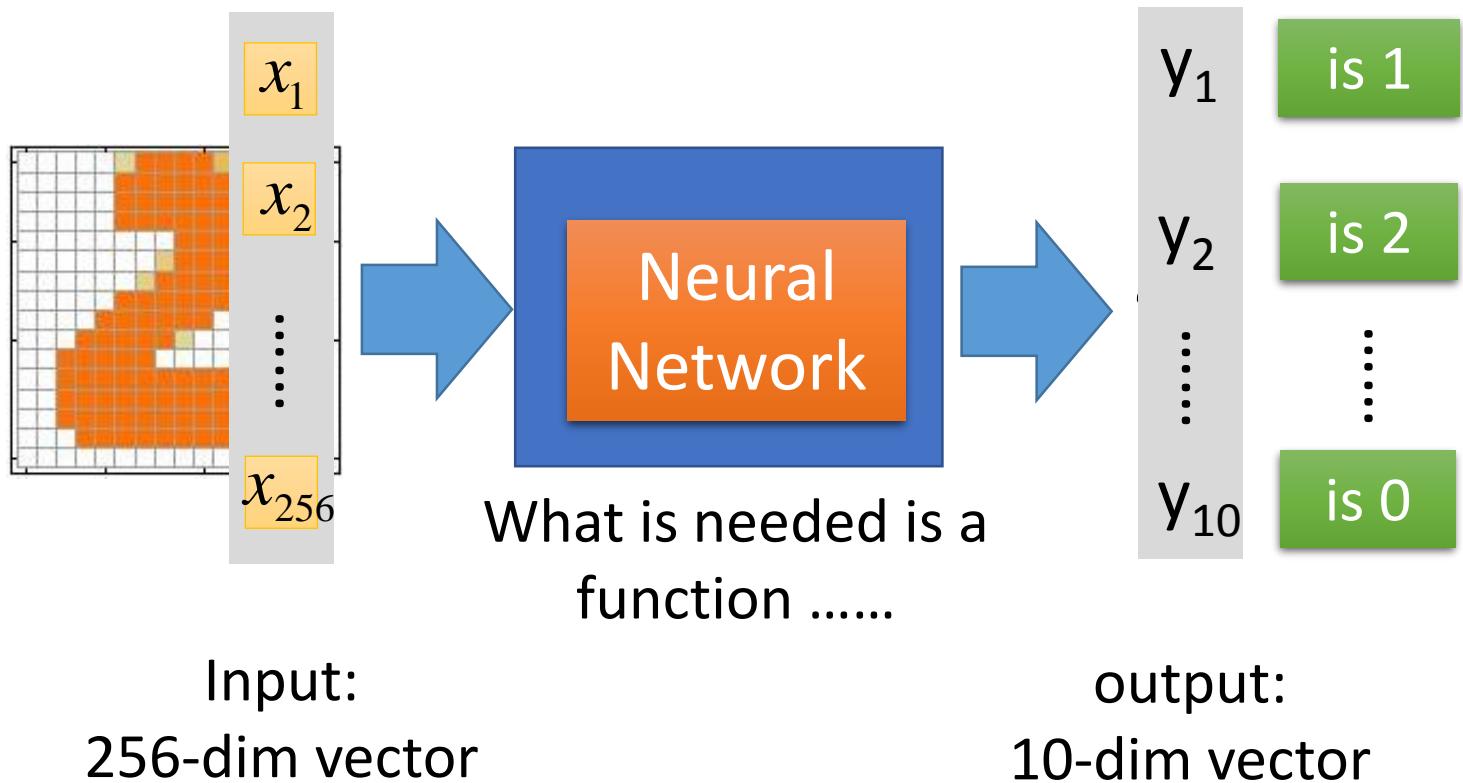
Output



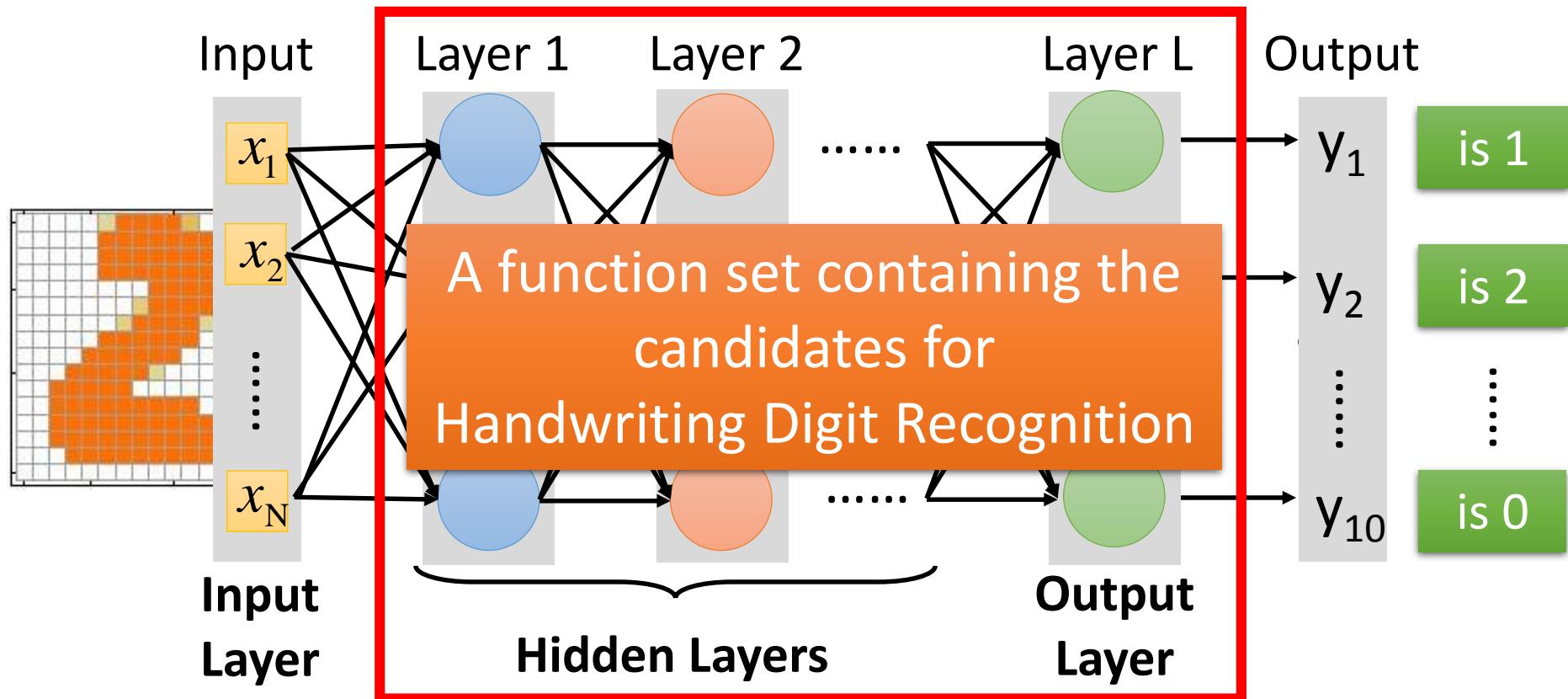
Each dimension represents the confidence of a digit.

Example Application

- Handwriting Digit Recognition

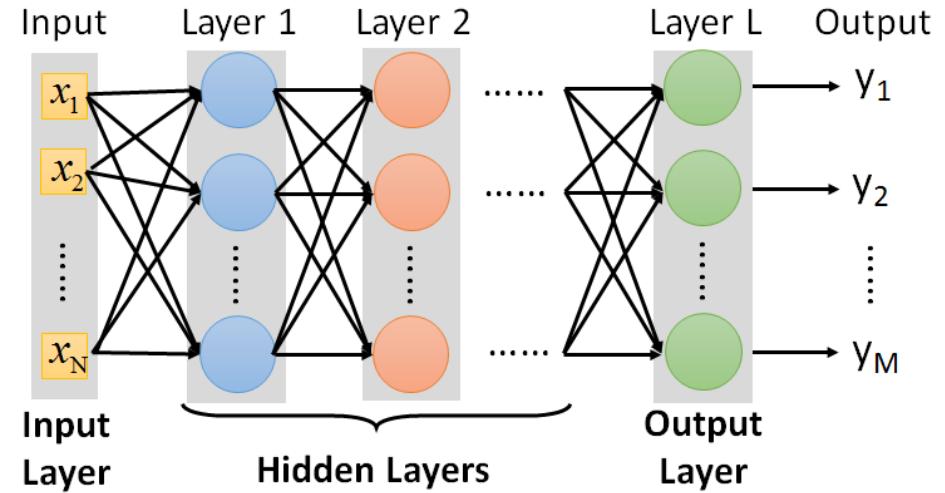


Example Application



You need to decide the network structure to let a good function in your function set.

FAQ



- Q: How many layers? How many neurons for each layer?

Trial and Error

+

Intuition

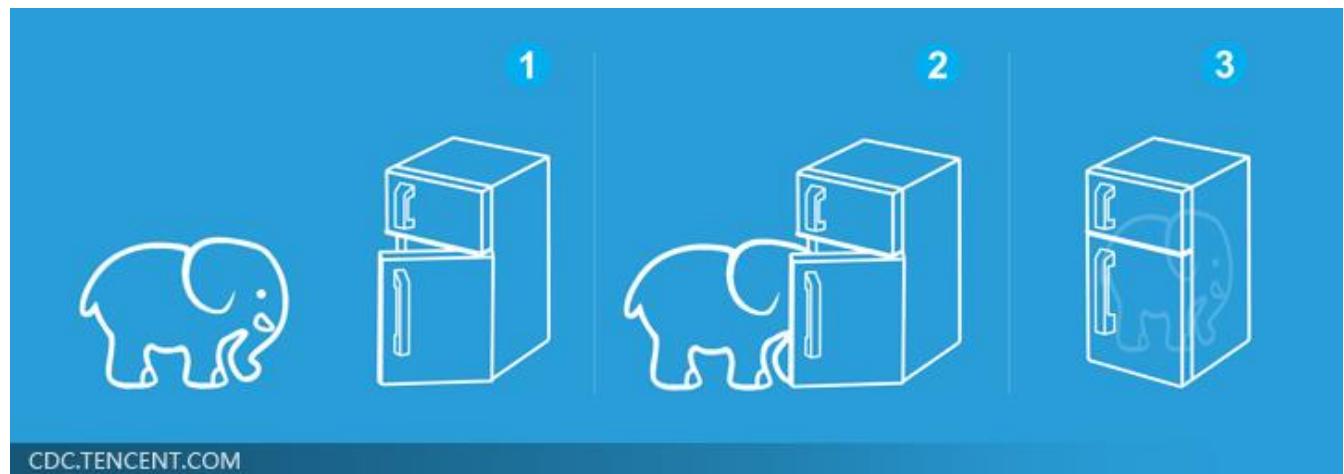
- Q: Can the structure be automatically determined?
 - E.g. Evolutionary Artificial Neural Networks
- Q: Can we design the network structure?

Convolutional Neural Network (CNN)

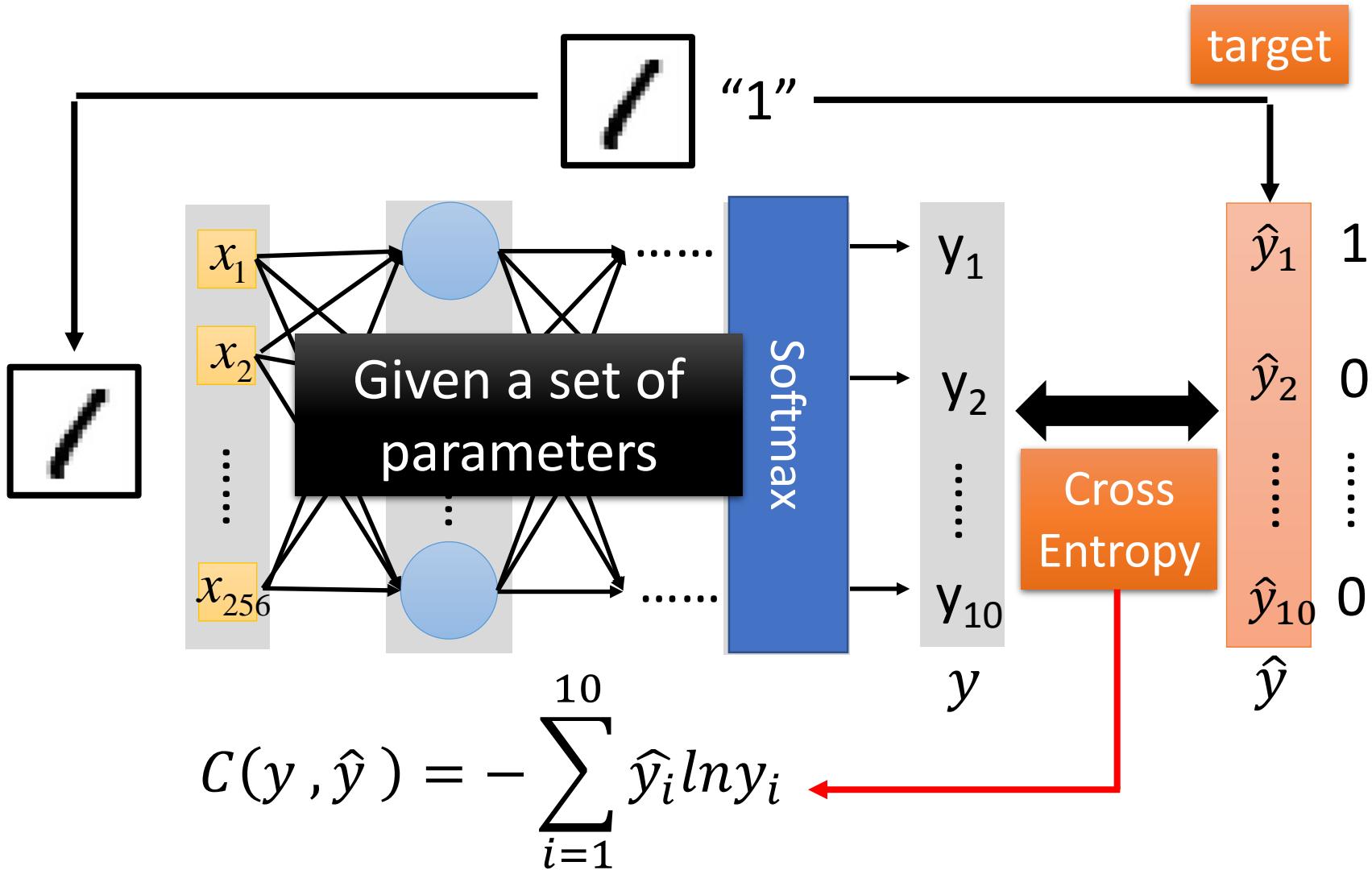
Three Steps for Deep Learning



Deep Learning is so simple

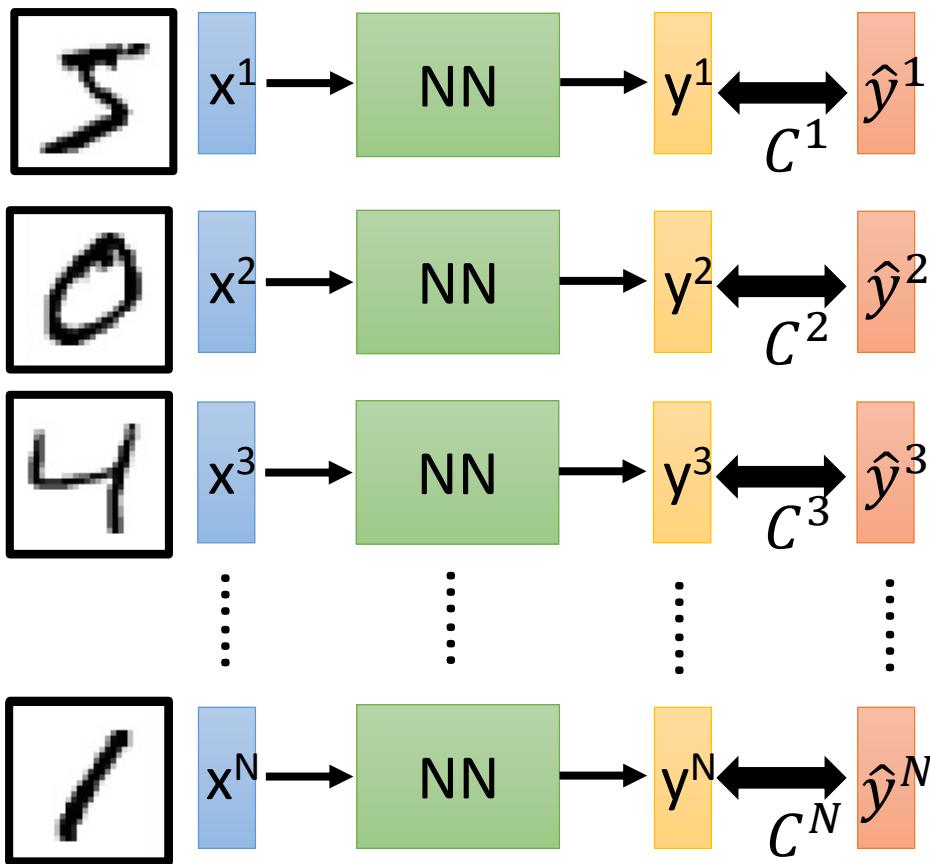


Loss for an Example



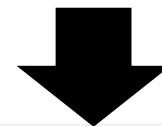
Total Loss

For all training data ...



Total Loss:

$$L = \sum_{n=1}^N C^n$$



Find a function in function set that minimizes total loss L

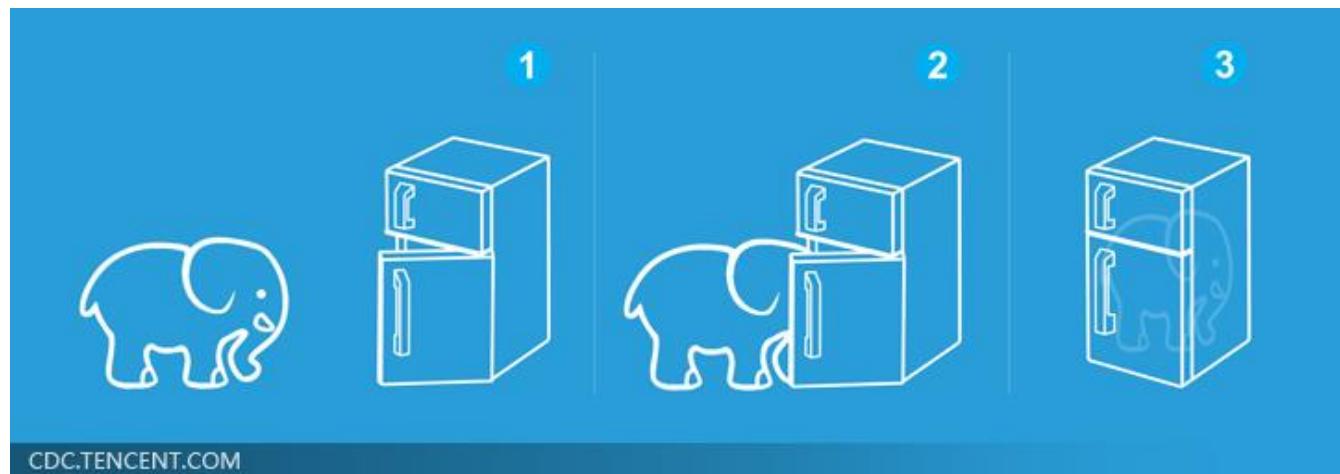


Find the network parameters θ^* that minimize total loss L

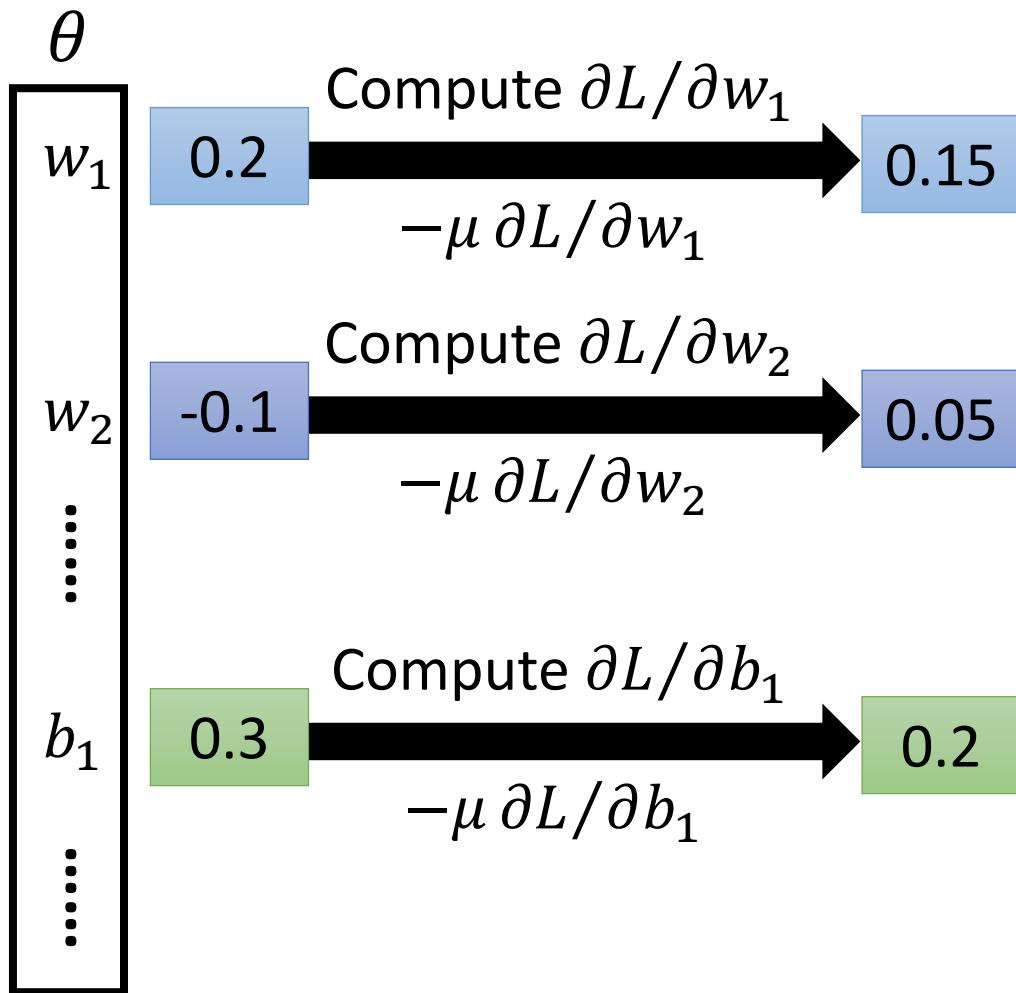
Three Steps for Deep Learning



Deep Learning is so simple



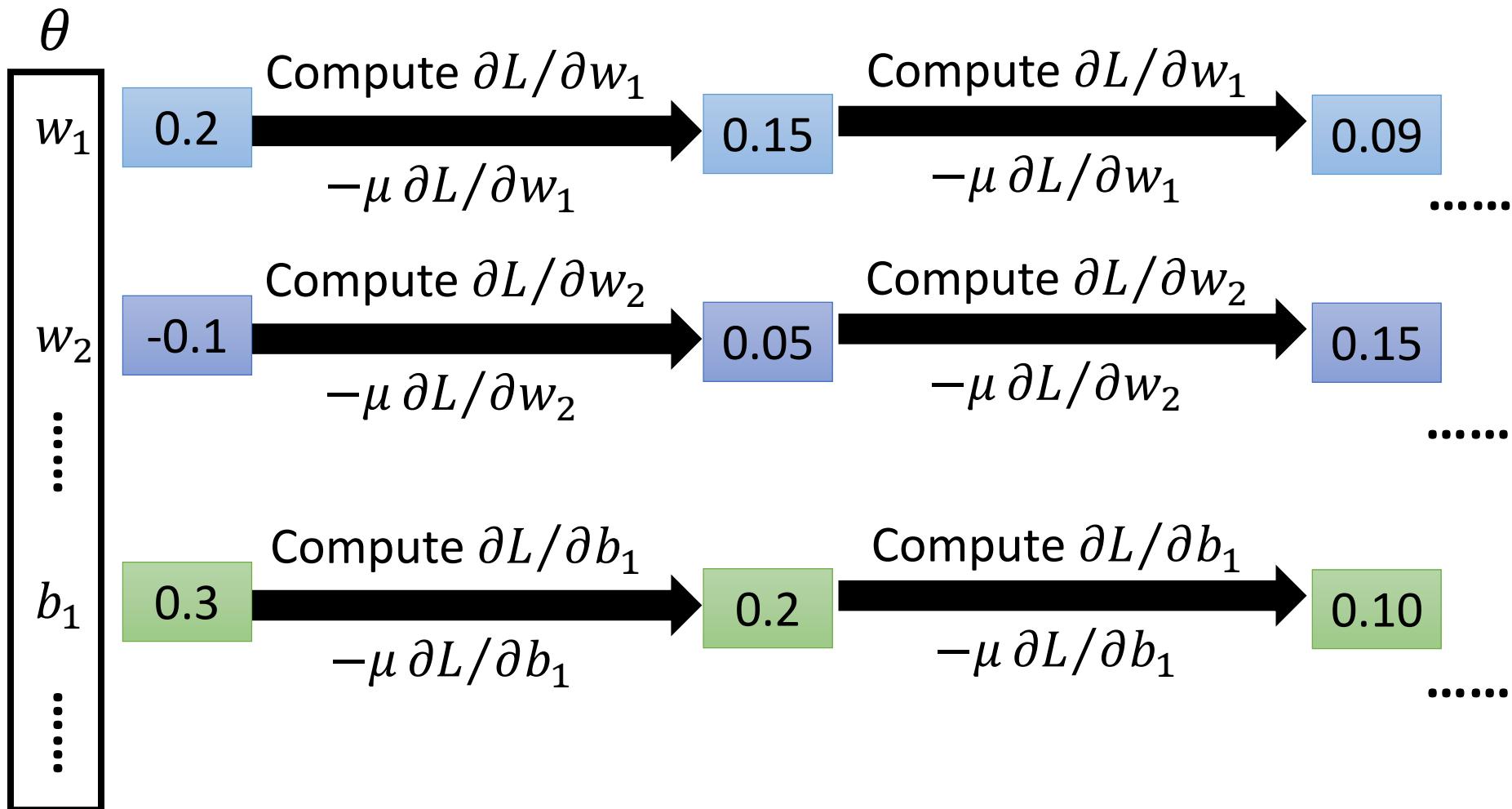
Gradient Descent



$$\nabla L = \begin{bmatrix} \frac{\partial L}{\partial w_1} \\ \frac{\partial L}{\partial w_2} \\ \vdots \\ \frac{\partial L}{\partial b_1} \\ \vdots \end{bmatrix}$$

gradient

Gradient Descent

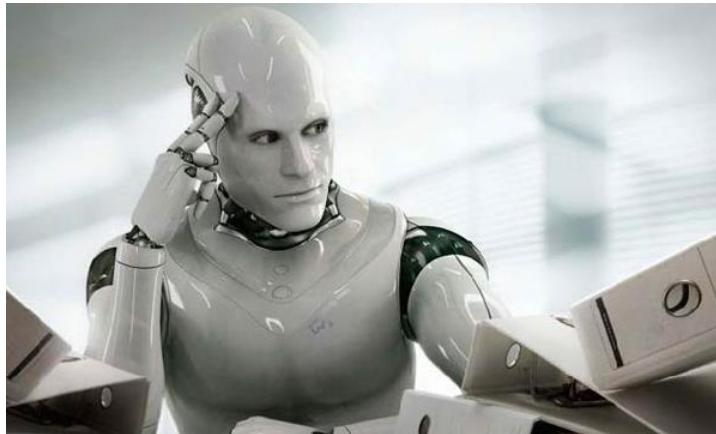


Gradient Descent

This is the “learning” of machines in deep learning

→ Even alpha go using this approach.

People image



Actually



I hope you are not too disappointed :p

Backpropagation

- Backpropagation: an efficient way to compute $\partial L / \partial w$ in neural network



Caffe



theano

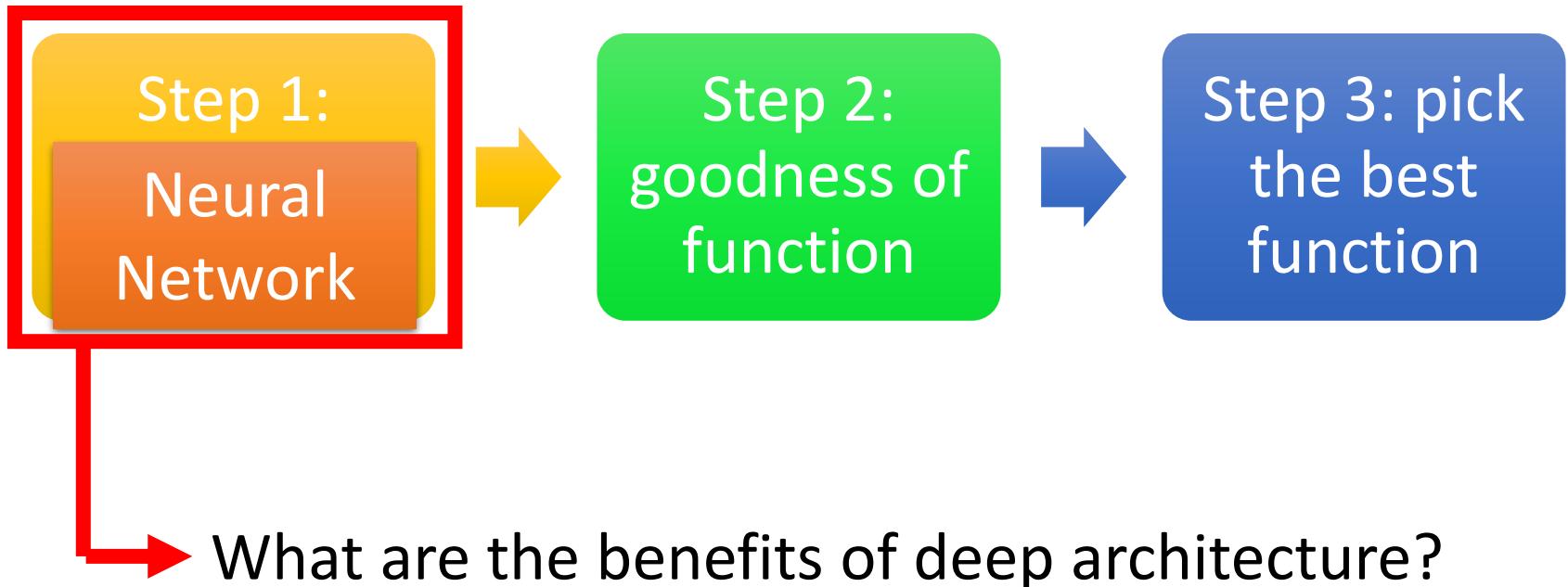


libdnn
台大周伯威
同學開發

Ref:

http://speech.ee.ntu.edu.tw/~tlkagk/courses/MLDS_2015_2/Lecture/DNN%20backprop.ecm.mp4/index.html

Concluding Remarks



Deeper is Better?

Layer X Size	Word Error Rate (%)
1 X 2k	24.2
2 X 2k	20.4
3 X 2k	18.4
4 X 2k	17.8
5 X 2k	17.2
7 X 2k	17.1

Not surprised, more parameters, better performance

Seide, Frank, Gang Li, and Dong Yu. "Conversational Speech Transcription Using Context-Dependent Deep Neural Networks." *Interspeech*. 2011.

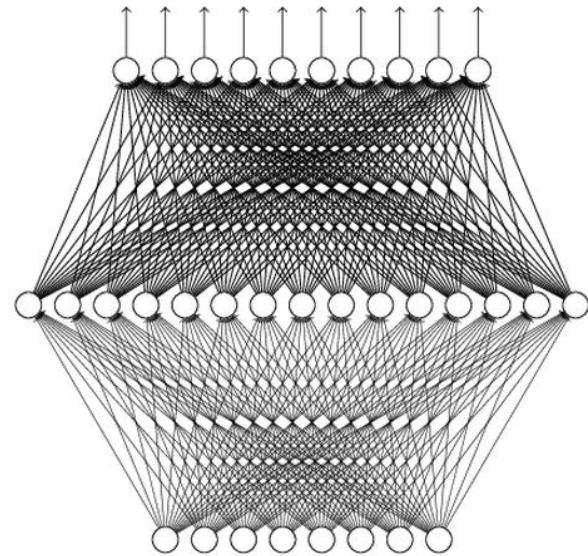
Universality Theorem

Any continuous function f

$$f : R^N \rightarrow R^M$$

Can be realized by a network
with one hidden layer

(given **enough** hidden
neurons)



Reference for the reason:
<http://neuralnetworksanddeeplearning.com/chap4.html>

Why “Deep” neural network not “Fat” neural network?

(next lecture)

“深度學習深度學習”

- My Course: Machine learning and having it deep and structured
 - http://speech.ee.ntu.edu.tw/~tlkagk/courses_MLSD15_2.html
 - 6 hour version: http://www.slideshare.net/tw_dsconf/ss-62245351
- “Neural Networks and Deep Learning”
 - written by Michael Nielsen
 - <http://neuralnetworksanddeeplearning.com/>
- “Deep Learning”
 - written by Yoshua Bengio, Ian J. Goodfellow and Aaron Courville
 - <http://www.deeplearningbook.org>

Acknowledgment

- 感謝 Victor Chen 發現投影片上的打字錯誤

Backpropagation

Gradient Descent

Network parameters $\theta = \{w_1, w_2, \dots, b_1, b_2, \dots\}$

Starting
Parameters $\theta^0 \longrightarrow \theta^1 \longrightarrow \theta^2 \longrightarrow \dots$

$$\nabla L(\theta) = \begin{bmatrix} \partial L(\theta)/\partial w_1 \\ \partial L(\theta)/\partial w_2 \\ \vdots \\ \partial L(\theta)/\partial b_1 \\ \partial L(\theta)/\partial b_2 \\ \vdots \end{bmatrix}$$

Compute $\nabla L(\theta^0)$ $\theta^1 = \theta^0 - \eta \nabla L(\theta^0)$

Compute $\nabla L(\theta^1)$ $\theta^2 = \theta^1 - \eta \nabla L(\theta^1)$

Millions of parameters

To compute the gradients efficiently,
we use **backpropagation**.

Chain Rule

Case 1

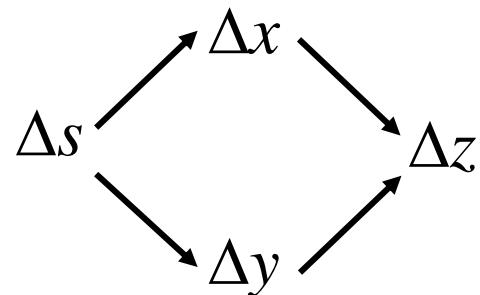
$$y = g(x) \quad z = h(y)$$

$$\Delta x \rightarrow \Delta y \rightarrow \Delta z$$

$$\frac{dz}{dx} = \frac{dz}{dy} \frac{dy}{dx}$$

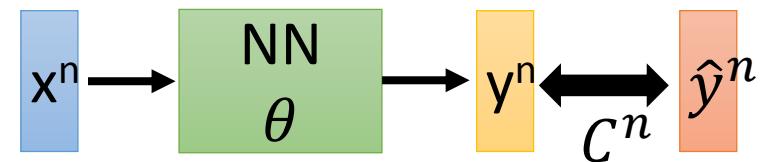
Case 2

$$x = g(s) \quad y = h(s) \quad z = k(x, y)$$



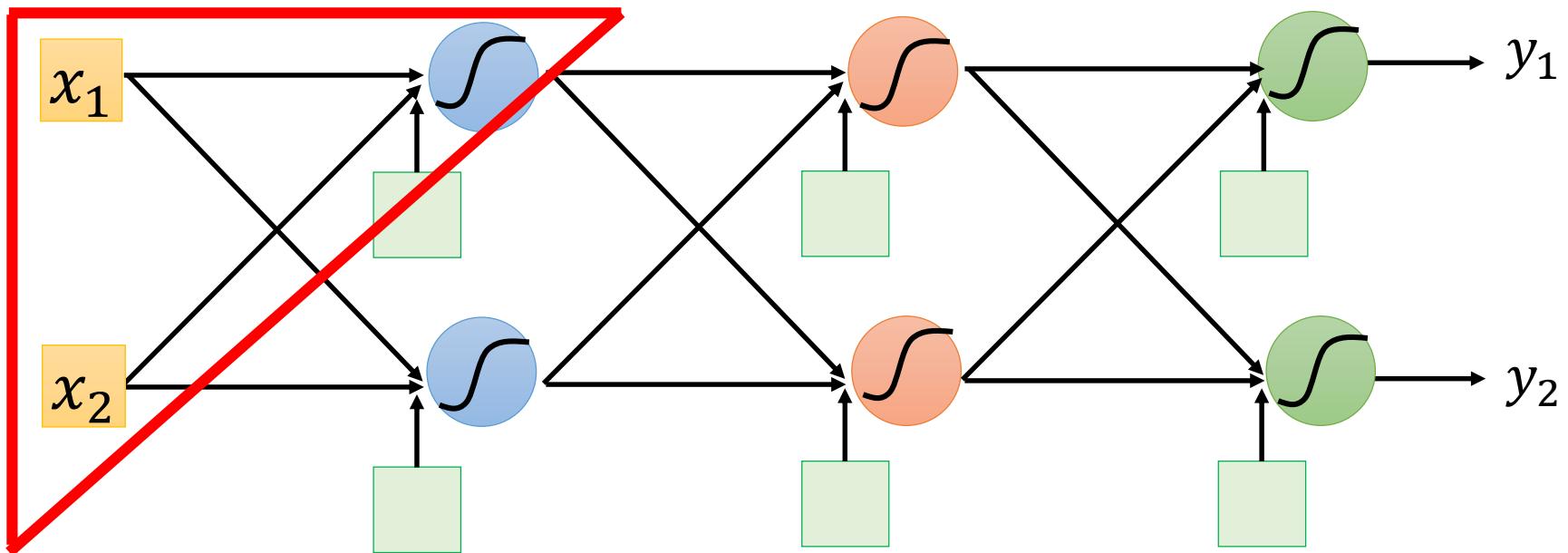
$$\frac{dz}{ds} = \frac{\partial z}{\partial x} \frac{dx}{ds} + \frac{\partial z}{\partial y} \frac{dy}{ds}$$

Backpropagation

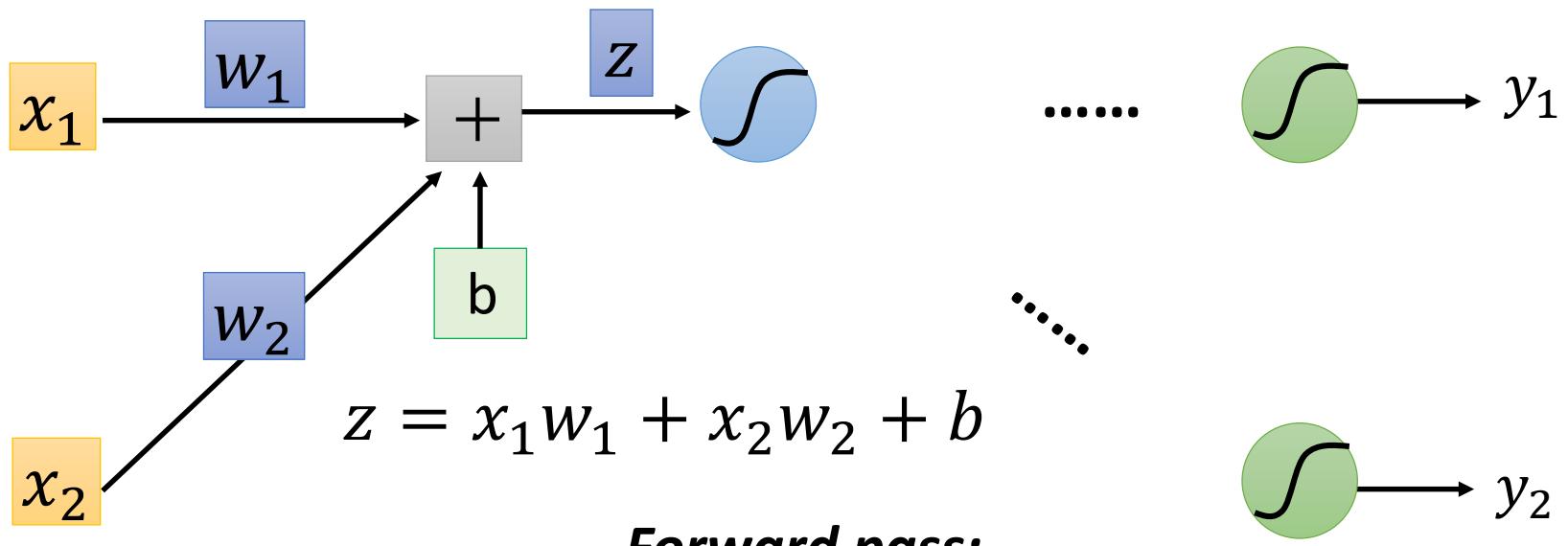


$$L(\theta) = \sum_{n=1}^N C^n(\theta) \quad \rightarrow$$

$$\frac{\partial L(\theta)}{\partial w} = \sum_{n=1}^N \frac{\partial C^n(\theta)}{\partial w}$$



Backpropagation



$$\frac{\partial C}{\partial w} = ? \quad \frac{\partial z}{\partial w} \frac{\partial C}{\partial z}$$

(Chain rule)

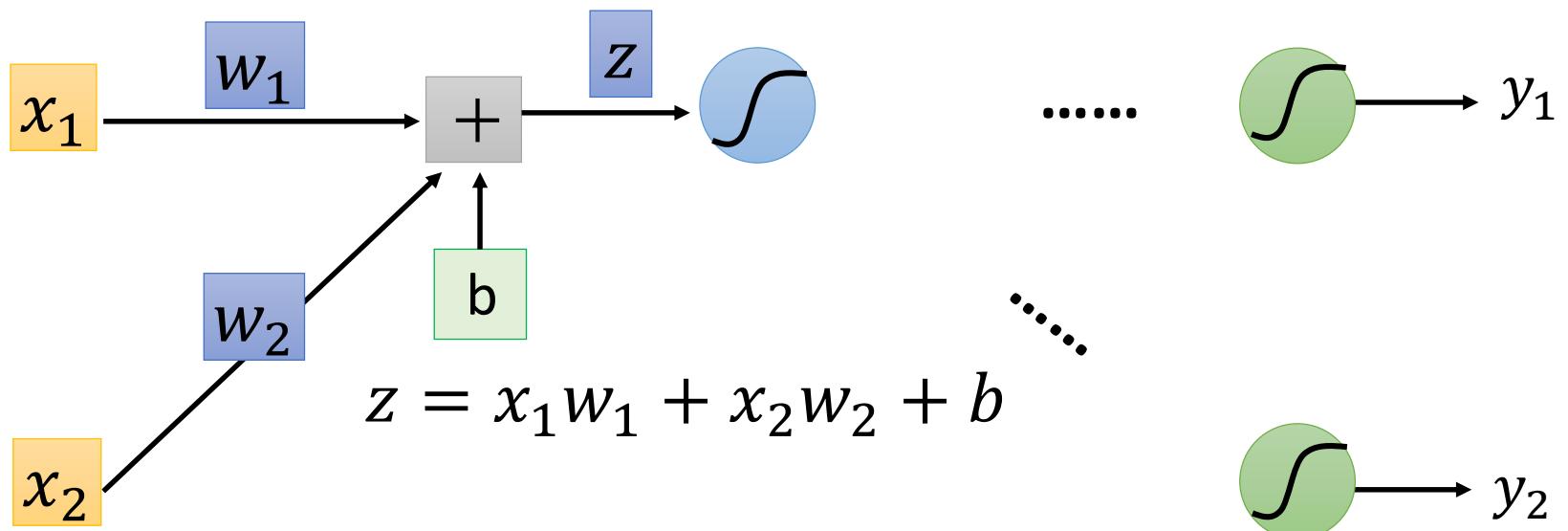
Compute $\partial z / \partial w$ for all parameters

Backward pass:

Compute $\partial C / \partial z$ for all activation function inputs z

Backpropagation – Forward pass

Compute $\partial z / \partial w$ for all parameters

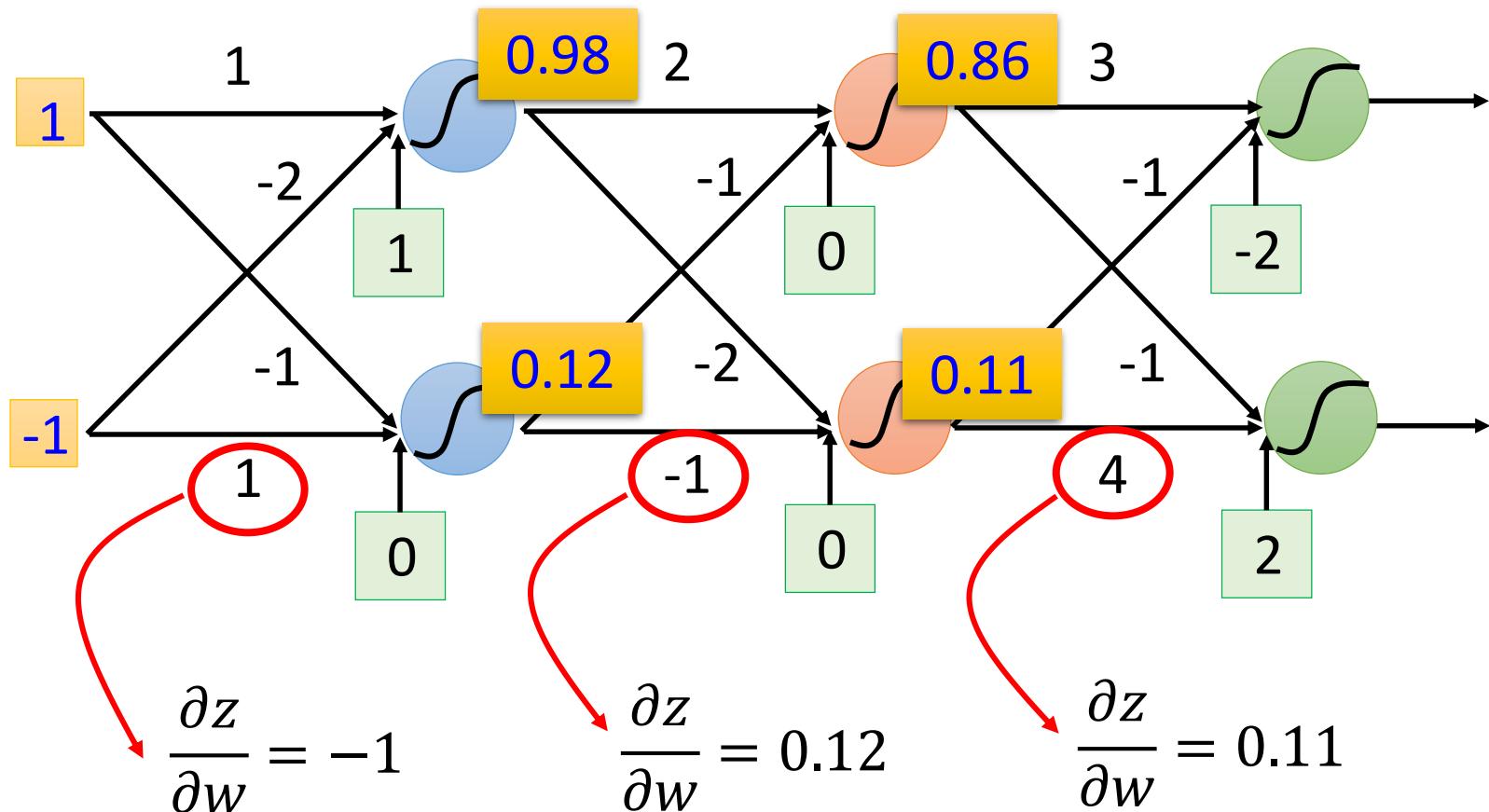


$$\begin{aligned}\partial z / \partial w_1 &=? x_1 \\ \partial z / \partial w_2 &=? x_2\end{aligned}$$

The value of the input
connected by the weight

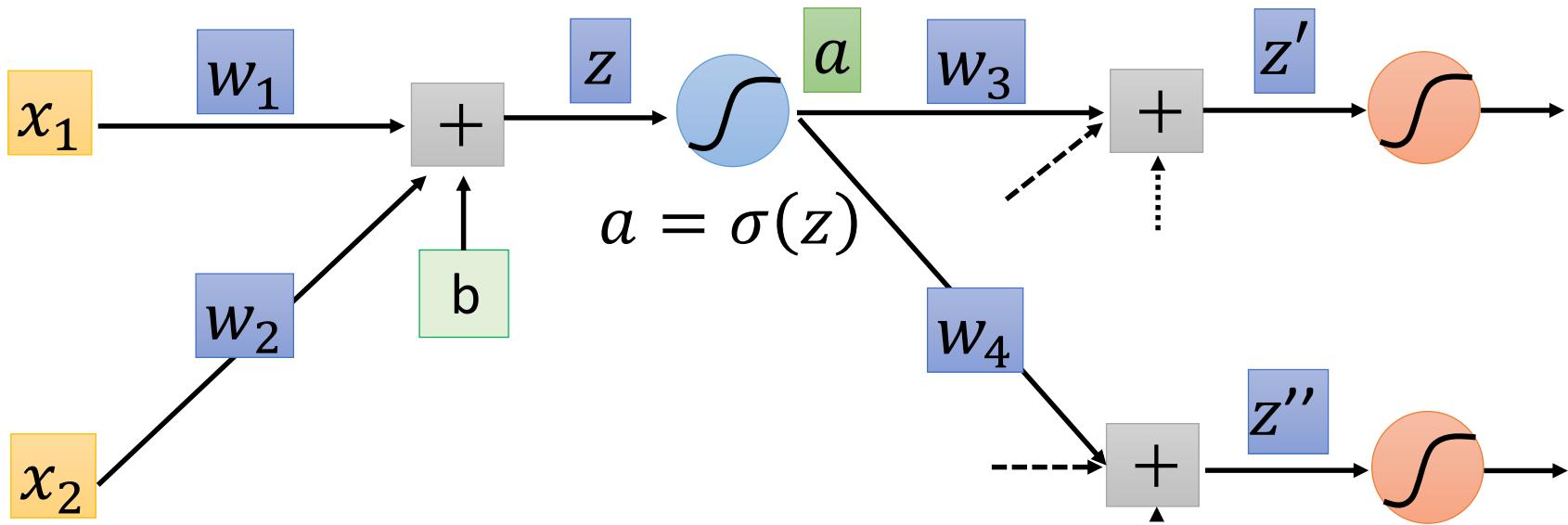
Backpropagation – Forward pass

Compute $\frac{\partial z}{\partial w}$ for all parameters



Backpropagation – Backward pass

Compute $\partial C / \partial z$ for all activation function inputs z

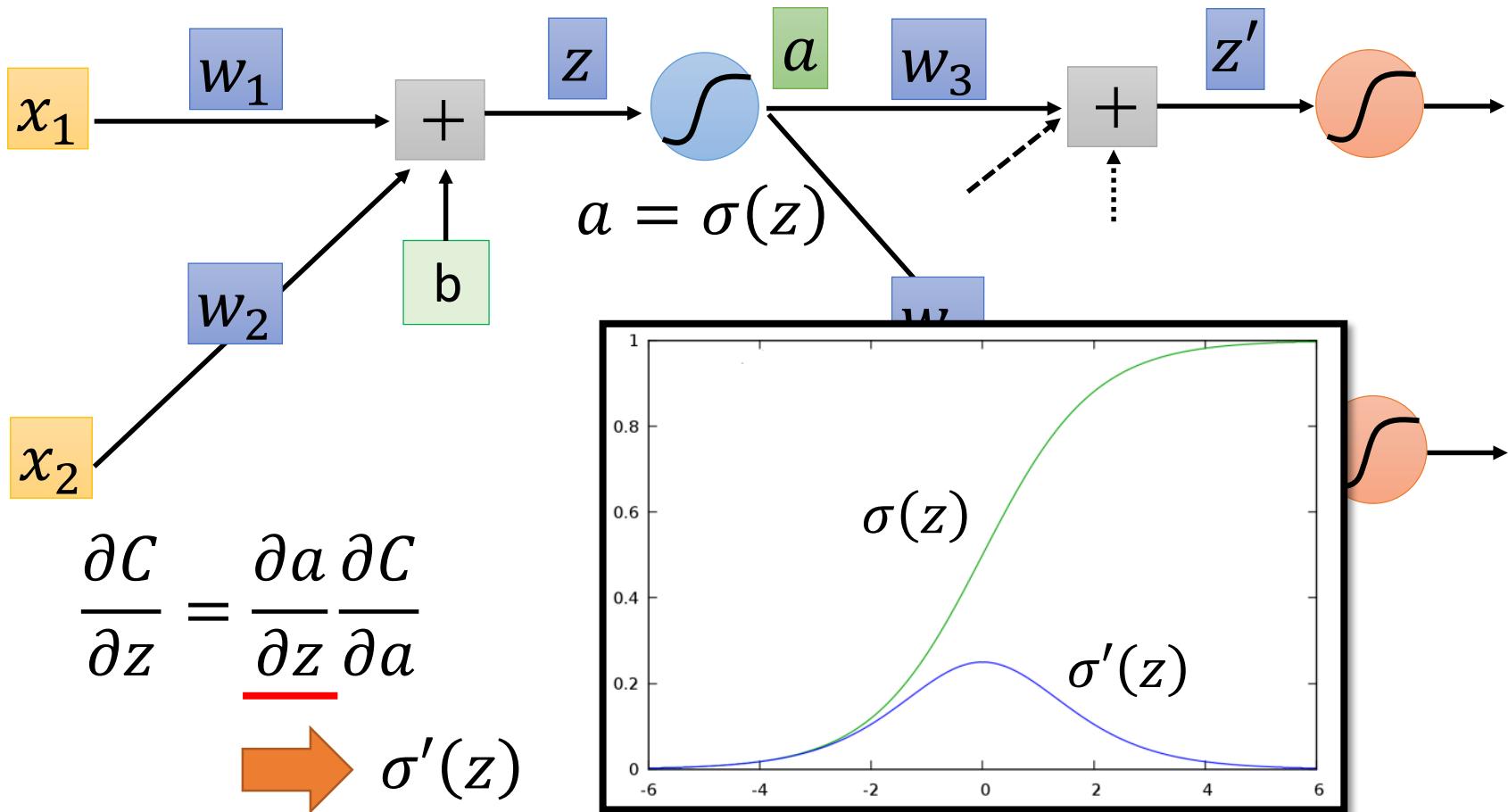


$$\frac{\partial C}{\partial z} = \frac{\partial a}{\partial z} \frac{\partial C}{\partial a}$$

→ $\sigma'(z)$

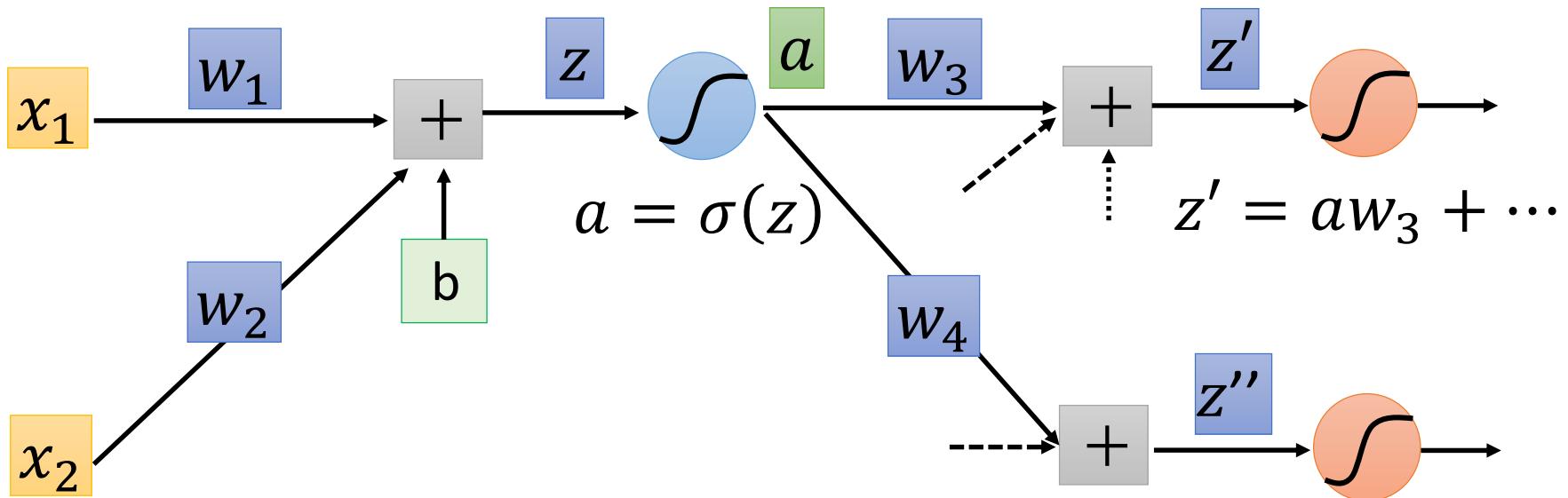
Backpropagation – Backward pass

Compute $\partial C / \partial z$ for all activation function inputs z



Backpropagation – Backward pass

Compute $\partial C / \partial z$ for all activation function inputs z



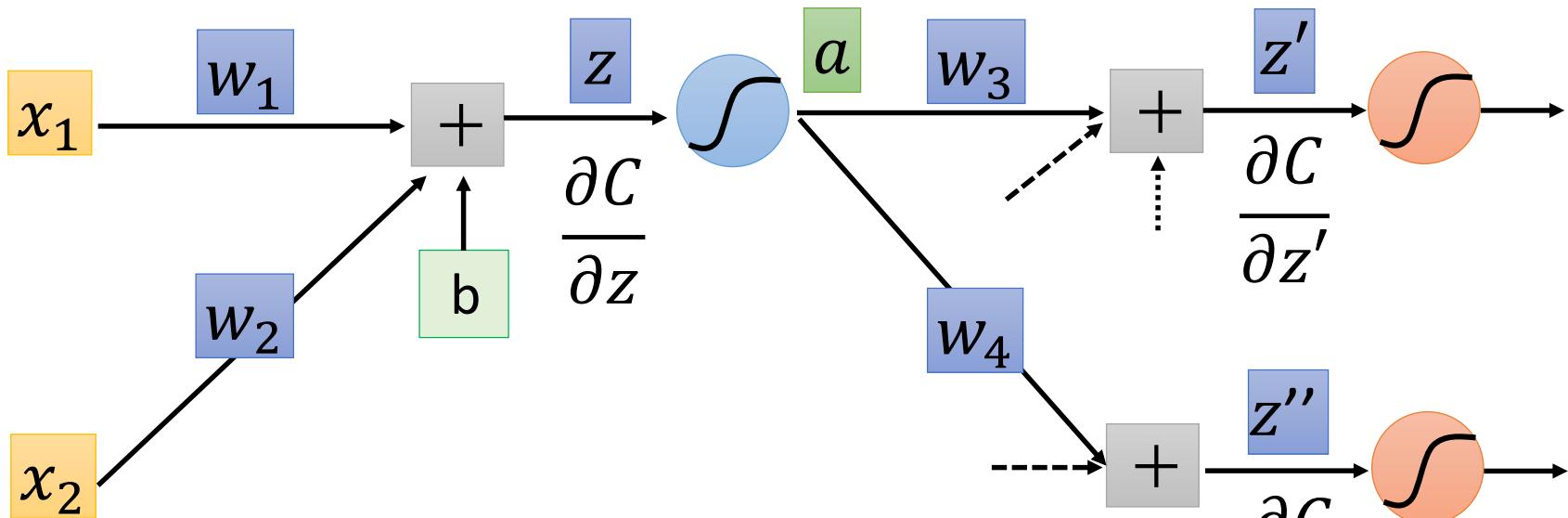
$$\frac{\partial C}{\partial z} = \frac{\partial a}{\partial z} \frac{\partial C}{\partial a}$$

$$\frac{\partial C}{\partial a} = \frac{\partial z'}{\partial a} \frac{\partial C}{\partial z'} + \frac{\partial z''}{\partial a} \frac{\partial C}{\partial z''}$$
 (Chain rule)
?
?
?

Assumed it's known

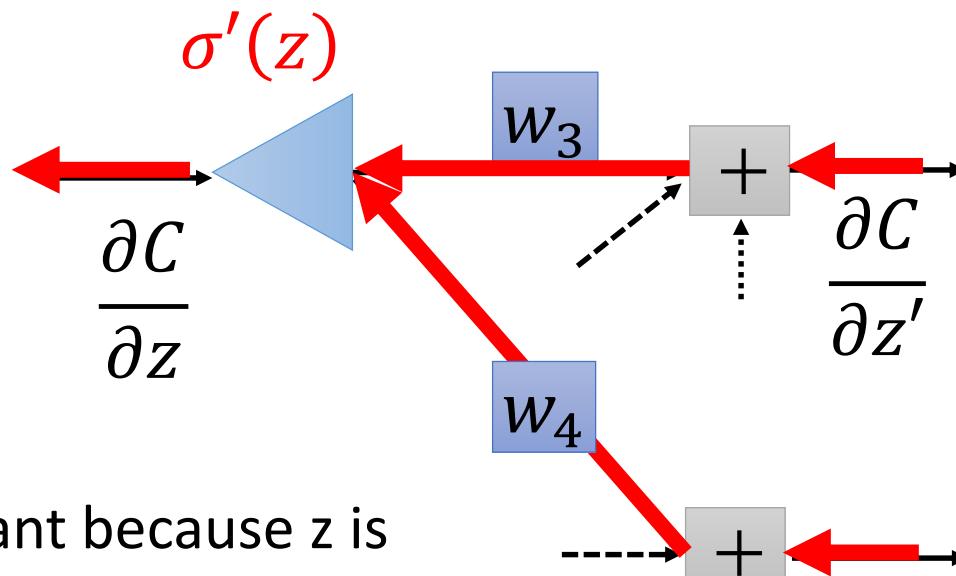
Backpropagation – Backward pass

Compute $\frac{\partial C}{\partial z}$ for all activation function inputs z



$$\frac{\partial C}{\partial z} = \sigma'(z) \left[w_3 \frac{\partial C}{\partial z'} + w_4 \frac{\partial C}{\partial z''} \right]$$

Backpropagation – Backward pass

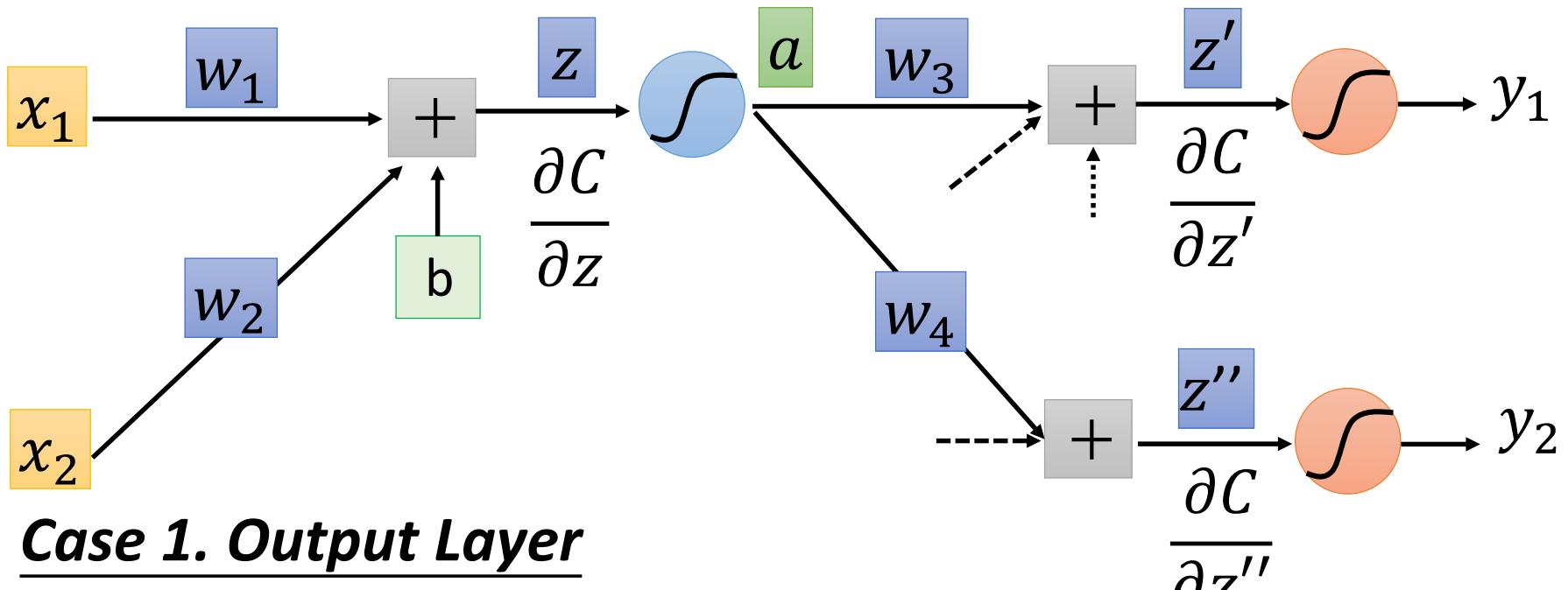


$\sigma'(z)$ is a constant because z is already determined in the forward pass.

$$\frac{\partial C}{\partial z} = \sigma'(z) \left[w_3 \frac{\partial C}{\partial z'} + w_4 \frac{\partial C}{\partial z''} \right]$$

Backpropagation – Backward pass

Compute $\frac{\partial C}{\partial z}$ for all activation function inputs z



Case 1. Output Layer

$$\frac{\partial C}{\partial z'} = \frac{\partial y_1}{\partial z'} \frac{\partial C}{\partial y_1}$$

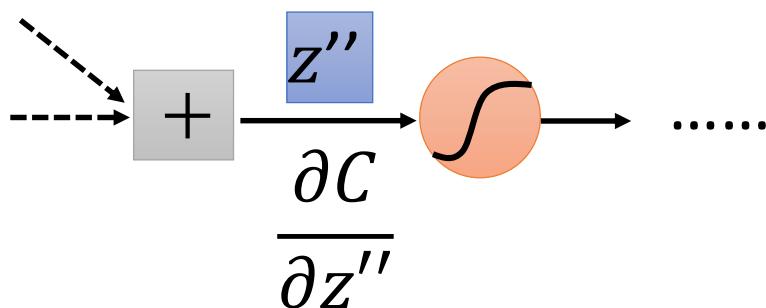
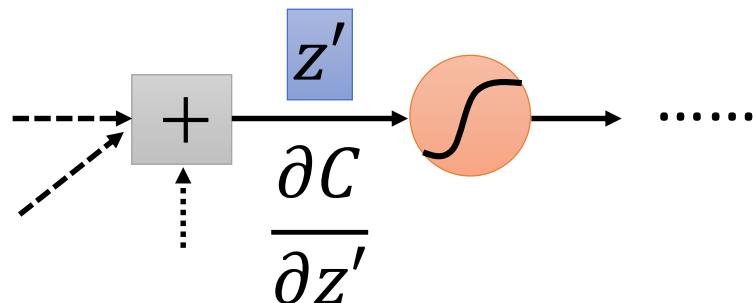
$$\frac{\partial C}{\partial z''} = \frac{\partial y_2}{\partial z''} \frac{\partial C}{\partial y_2}$$

Done!

Backpropagation – Backward pass

Compute $\frac{\partial C}{\partial z}$ for all activation function inputs z

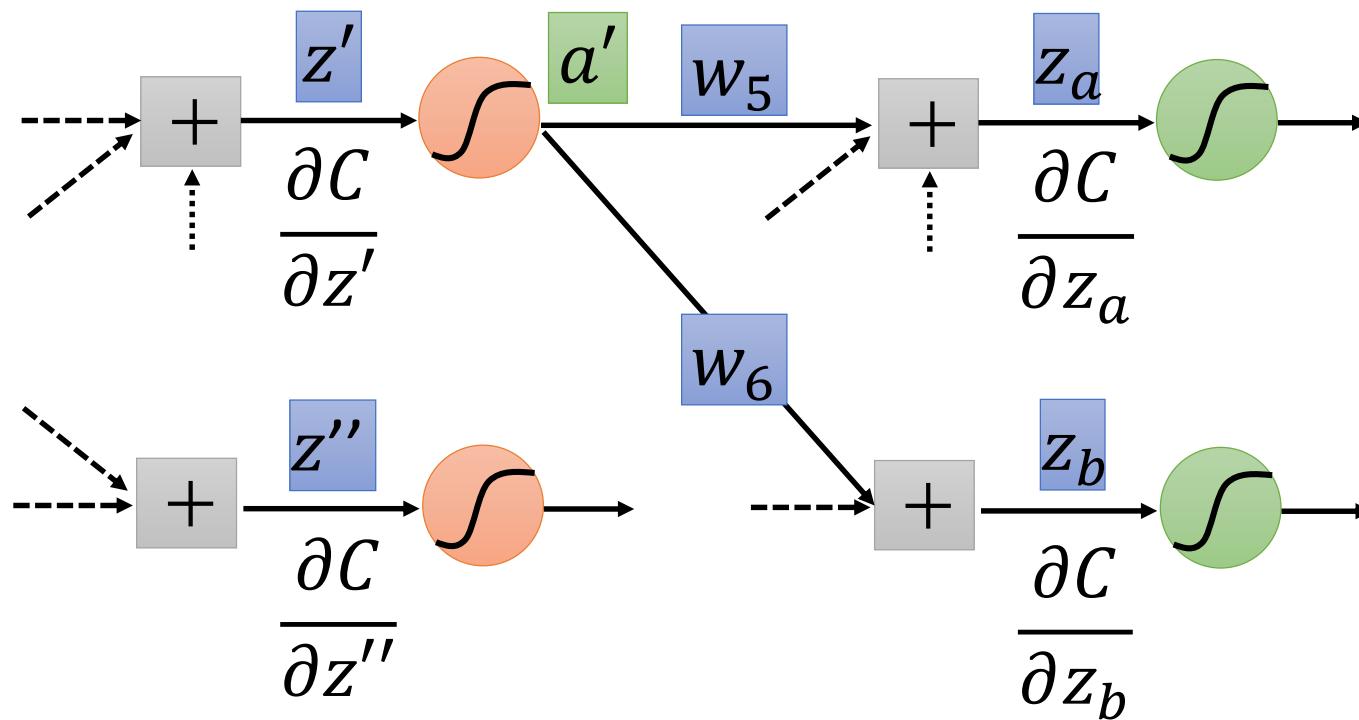
Case 2. Not Output Layer



Backpropagation – Backward pass

Compute $\frac{\partial C}{\partial z}$ for all activation function inputs z

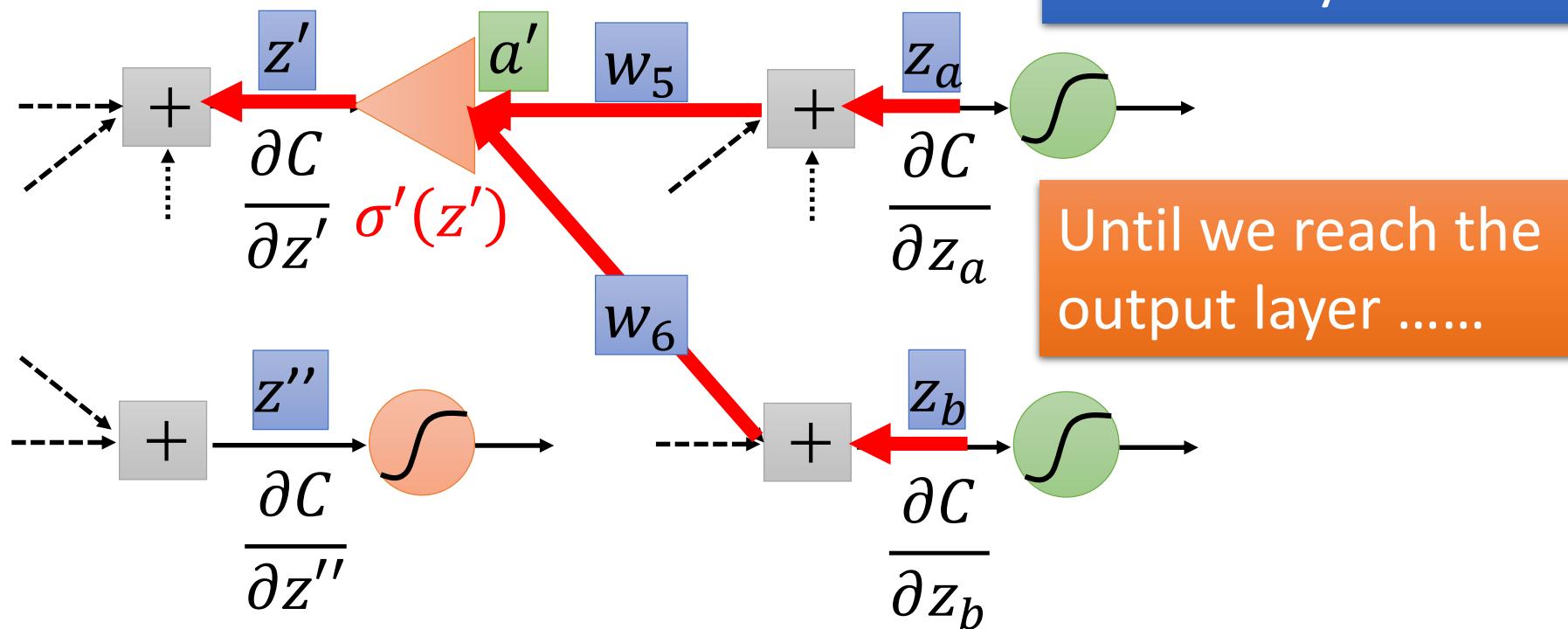
Case 2. Not Output Layer



Backpropagation – Backward pass

Compute $\partial C / \partial z$ for all activation function inputs z

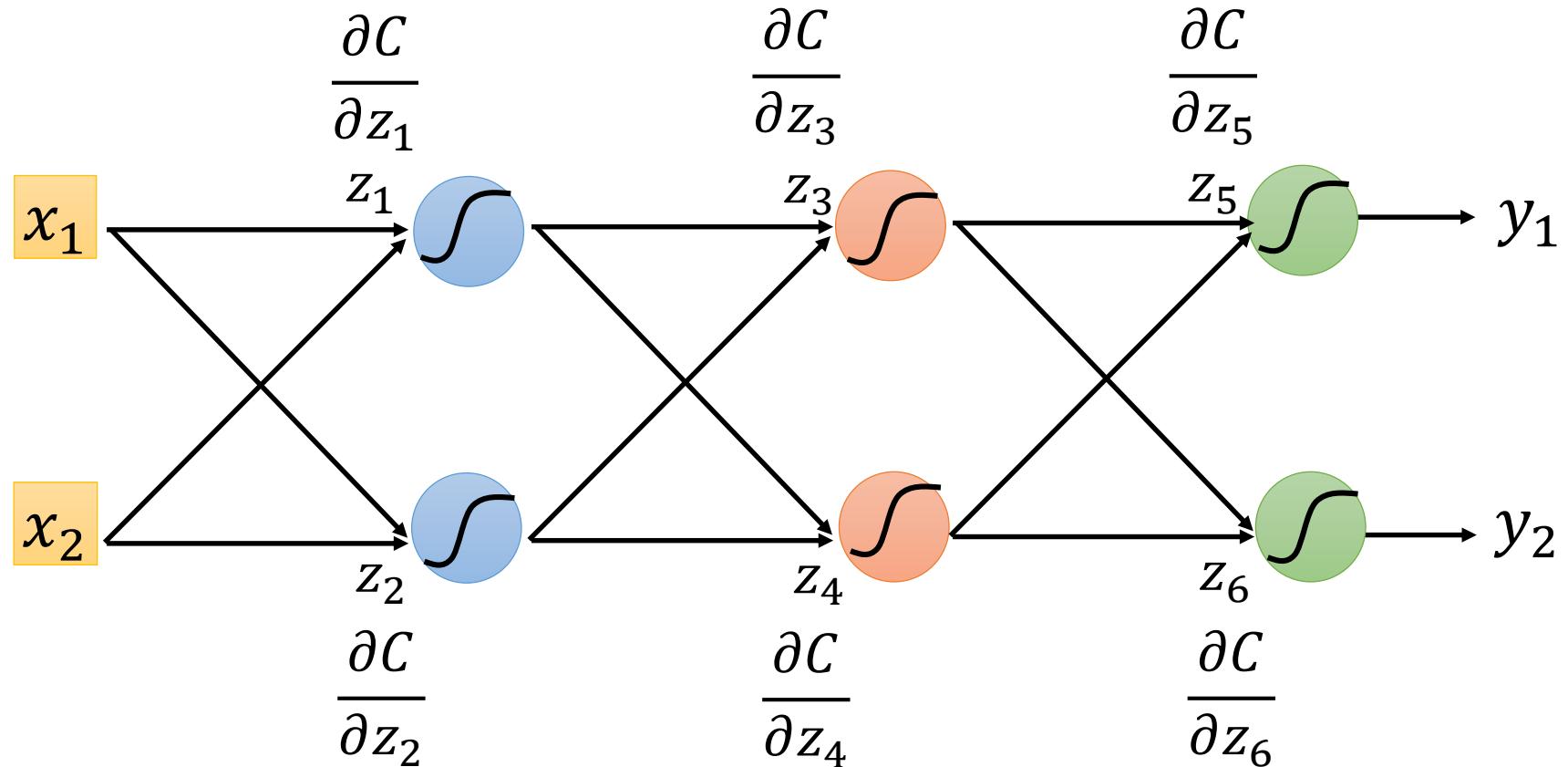
Case 2. Not Output Layer



Backpropagation – Backward Pass

Compute $\partial C / \partial z$ for all activation function inputs z

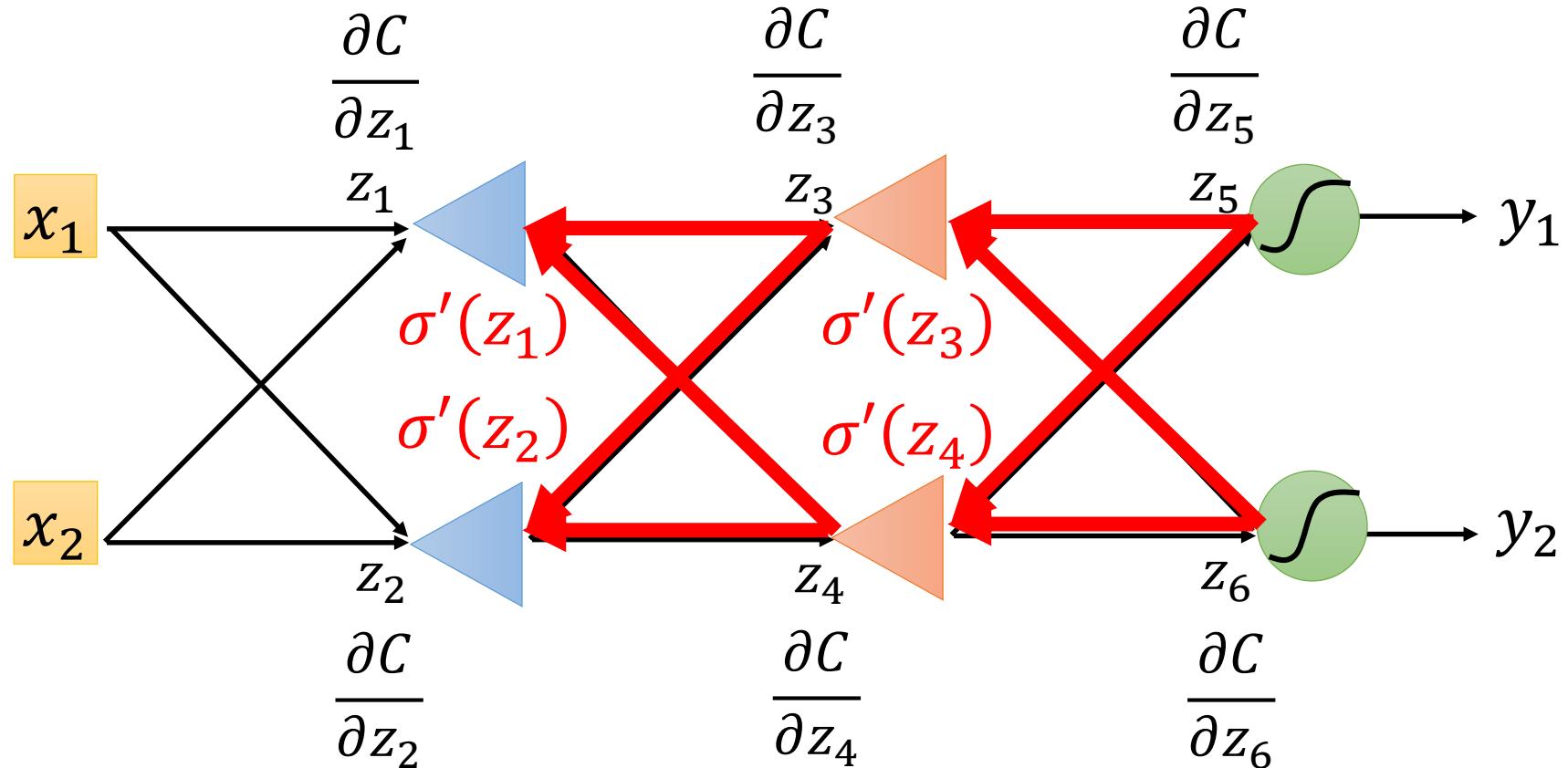
Compute $\partial C / \partial z$ from the output layer



Backpropagation – Backward Pass

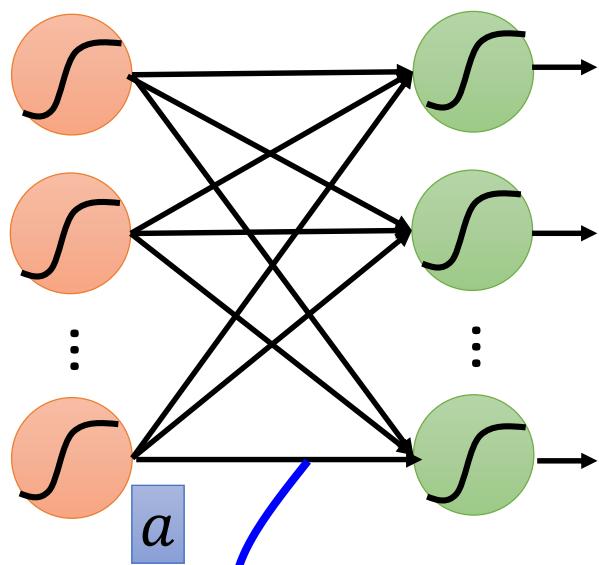
Compute $\partial C / \partial z$ for all activation function inputs z

Compute $\partial C / \partial z$ from the output layer



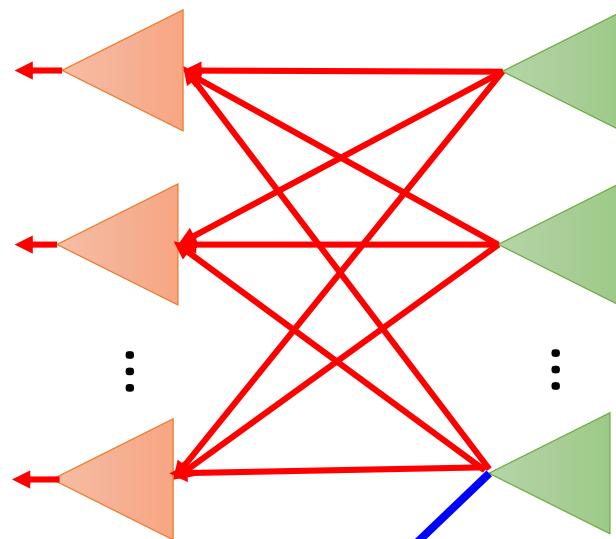
Backpropagation – Summary

Forward Pass



$$\frac{\partial z}{\partial w} = a$$

Backward Pass



$$X \quad \frac{\partial C}{\partial z} = \frac{\partial C}{\partial w}$$

for all w

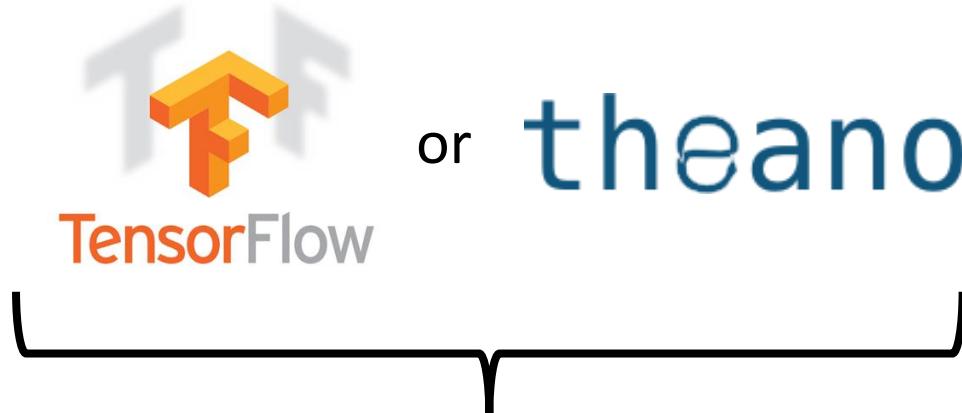
“Hello world”
of deep learning

Keras

If you want to learn theano:

http://speech.ee.ntu.edu.tw/~tlkagk/courses/MLDS_2015_2/Lecture/Theano%20DNN.ecm.mp4/index.html

[http://speech.ee.ntu.edu.tw/~tlkagk/courses/MLDS_2015_2/Lecture/RNN%20training%20\(v6\).ecm.mp4/index.html](http://speech.ee.ntu.edu.tw/~tlkagk/courses/MLDS_2015_2/Lecture/RNN%20training%20(v6).ecm.mp4/index.html)



Interface of
TensorFlow or
Theano

Very flexible
Need some
effort to learn

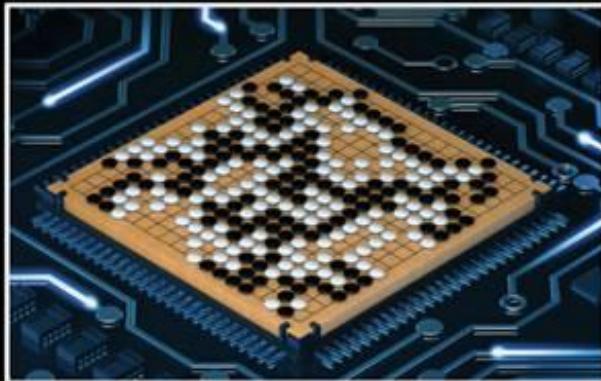
Easy to learn and use
(still have some flexibility)
You can modify it if you can write
TensorFlow or Theano

Keras

- François Chollet is the author of Keras.
 - He currently works for Google as a deep learning engineer and researcher.
- Keras means *horn* in Greek
- Documentation: <http://keras.io/>
- Example:
<https://github.com/fchollet/keras/tree/master/examples>

使用 Keras 心得

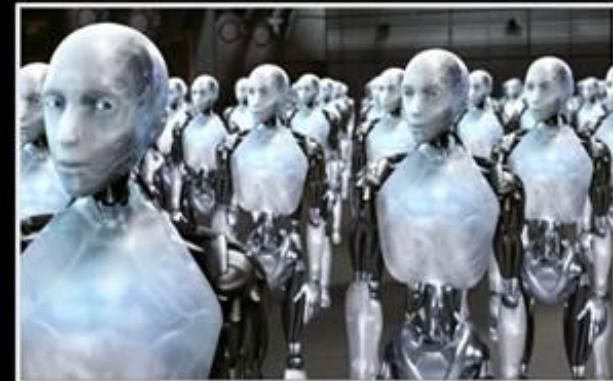
Deep Learning研究生



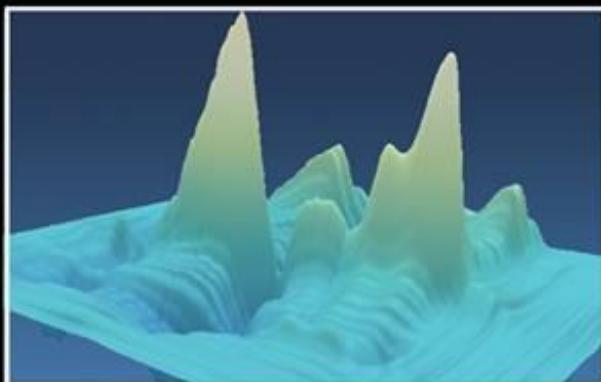
朋友覺得我在



我媽覺得我在



大眾覺得我在



指導教授覺得我在



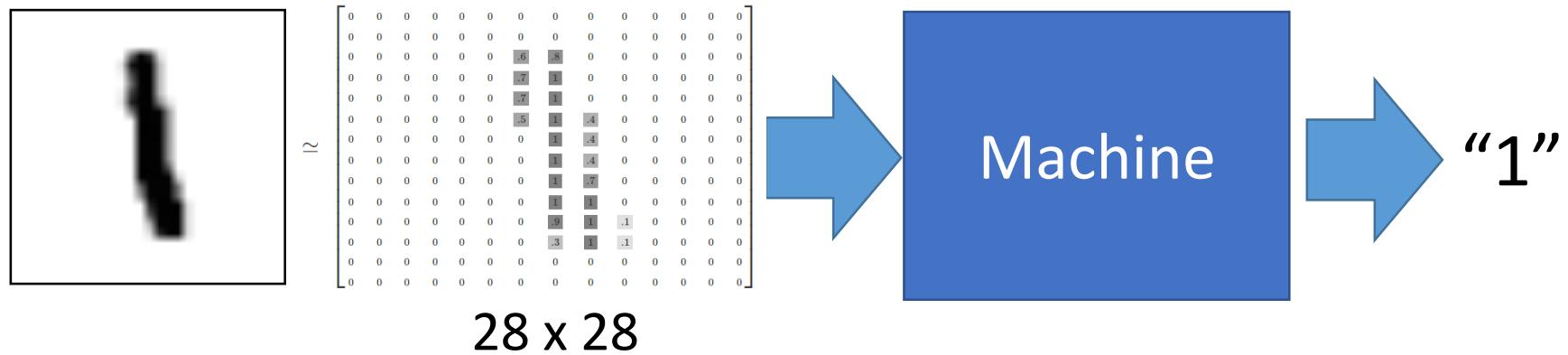
我以為我在



事實上我在

“Hello world”

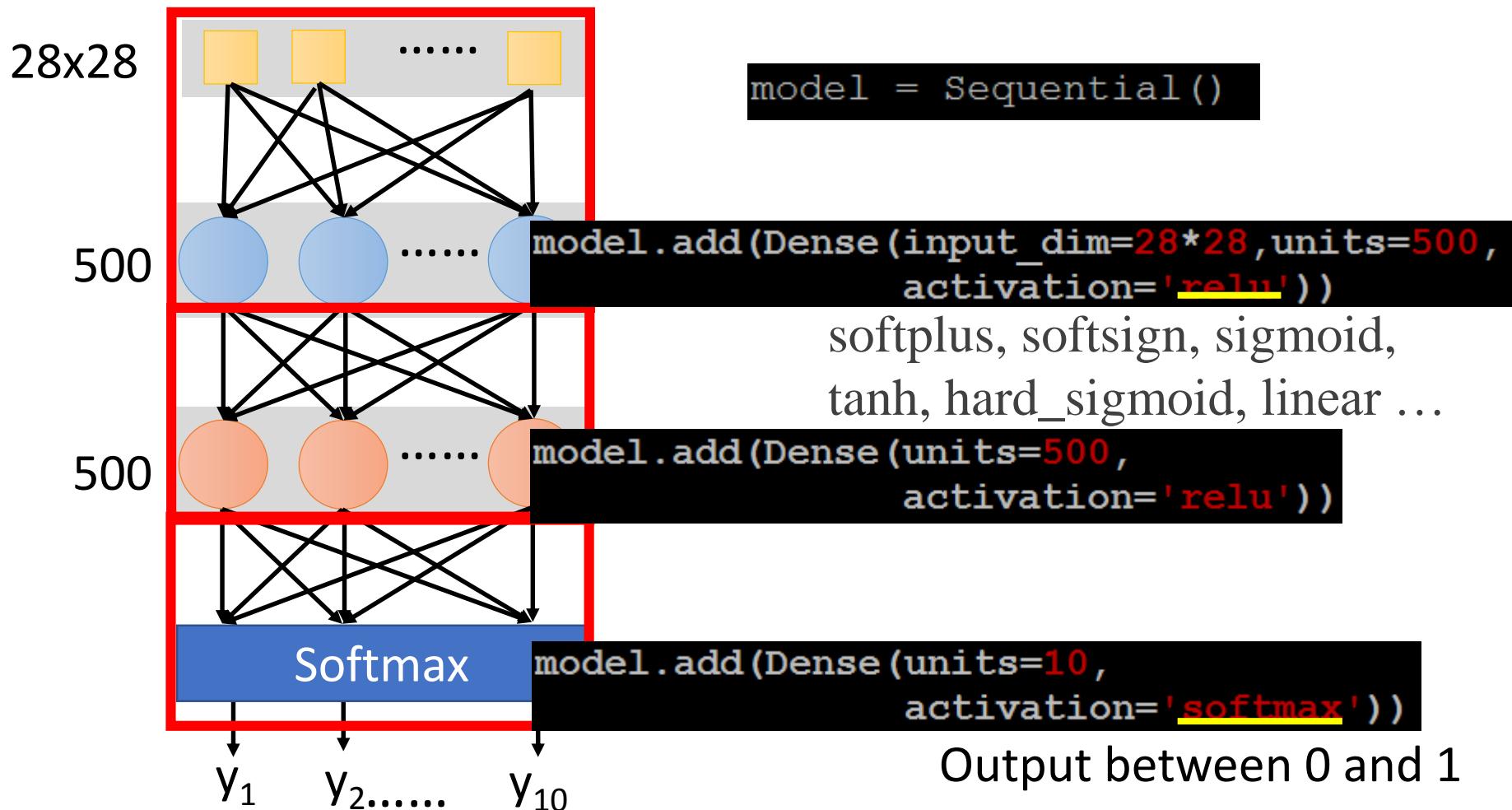
- Handwriting Digit Recognition



MNIST Data: <http://yann.lecun.com/exdb/mnist/>

Keras provides data sets loading function: <http://keras.io/datasets/>

Keras: Building a Network



Configuration

Several alternatives: <https://keras.io/objectives/>

```
model.compile(loss='categorical_crossentropy',  
              optimizer='adam',  
              metrics=['accuracy'])
```

SGD, RMSprop, Adagrad, Adadelta, Adam, Adamax, Nadam

Pick the best function

```
model.fit(x_train,y_train,batch_size=100,epochs=20)
```

numpy array

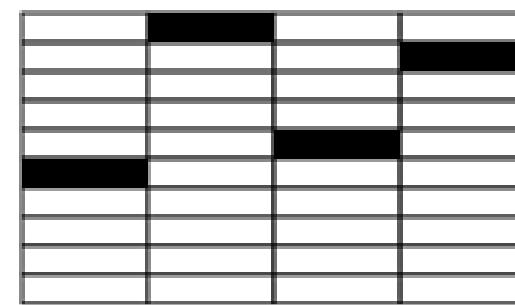
$$28 \times 28 = 784$$



Number of training examples

numpy array

$$10$$

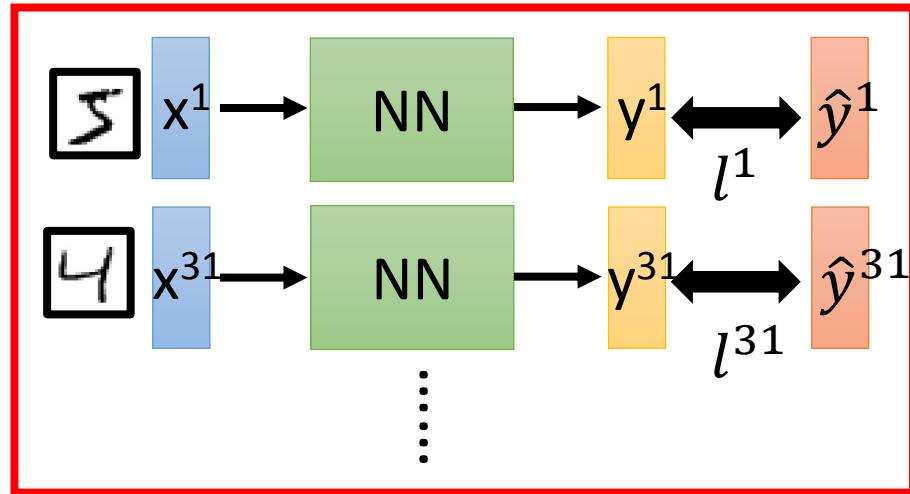


Number of training examples

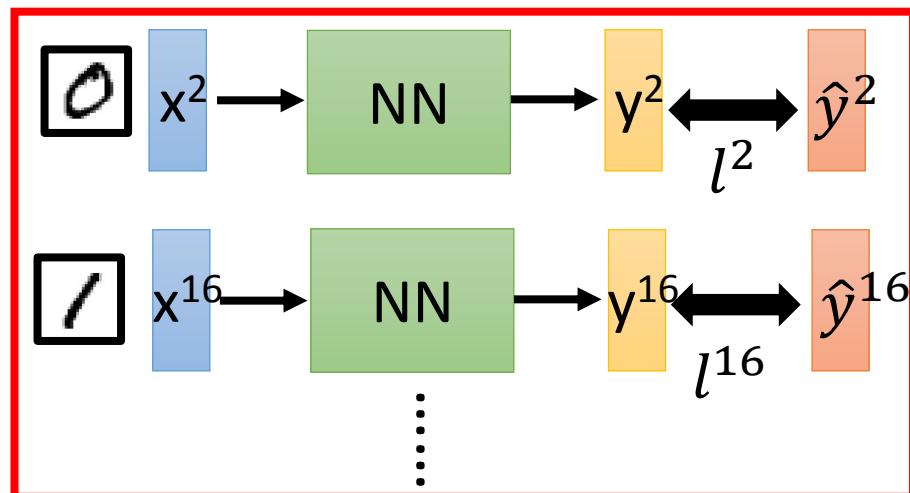
We do not really minimize total loss!

Mini-batch

Mini-batch



Mini-batch



- Randomly initialize network parameters

- Pick the 1st batch
 $L' = l^1 + l^{31} + \dots$
Update parameters once
- Pick the 2nd batch
 $L'' = l^2 + l^{16} + \dots$
Update parameters once
- ⋮
- Until all mini-batches have been picked

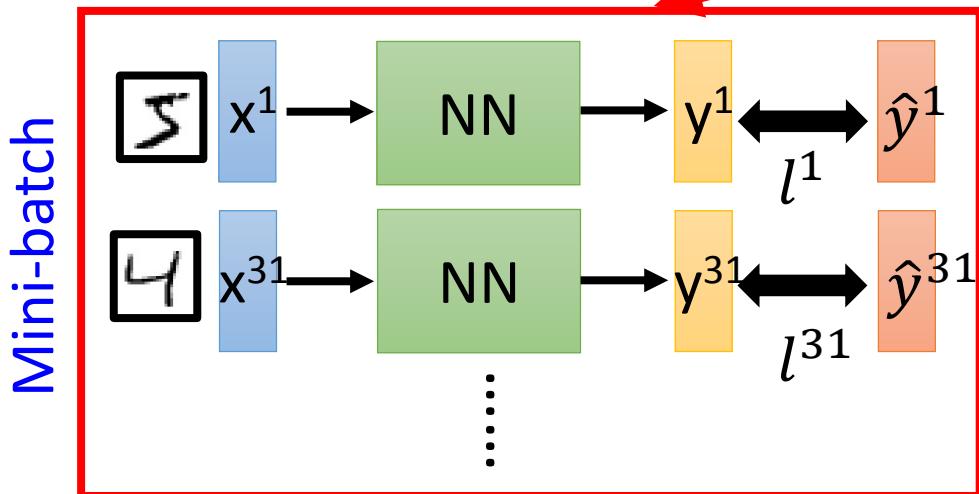
one epoch

Repeat the above process

Mini-batch

Batch size influences both *speed* and *performance*. You have to tune it.

```
model.fit(x_train, y_train, batch_size=100, nb_epoch=20)
```



100 examples in a mini-batch

Batch size = 1 ➔

Stochastic gradient descent

Repeat 20 times

one epoch

- Pick the 1st batch
 $L' = l^1 + l^{31} + \dots$
Update parameters once
- Pick the 2nd batch
 $L'' = l^2 + l^{16} + \dots$
Update parameters once
- ⋮
- Until all mini-batches have been picked

Speed

Very large batch size can yield worse performance

- Smaller batch size means more updates in one epoch

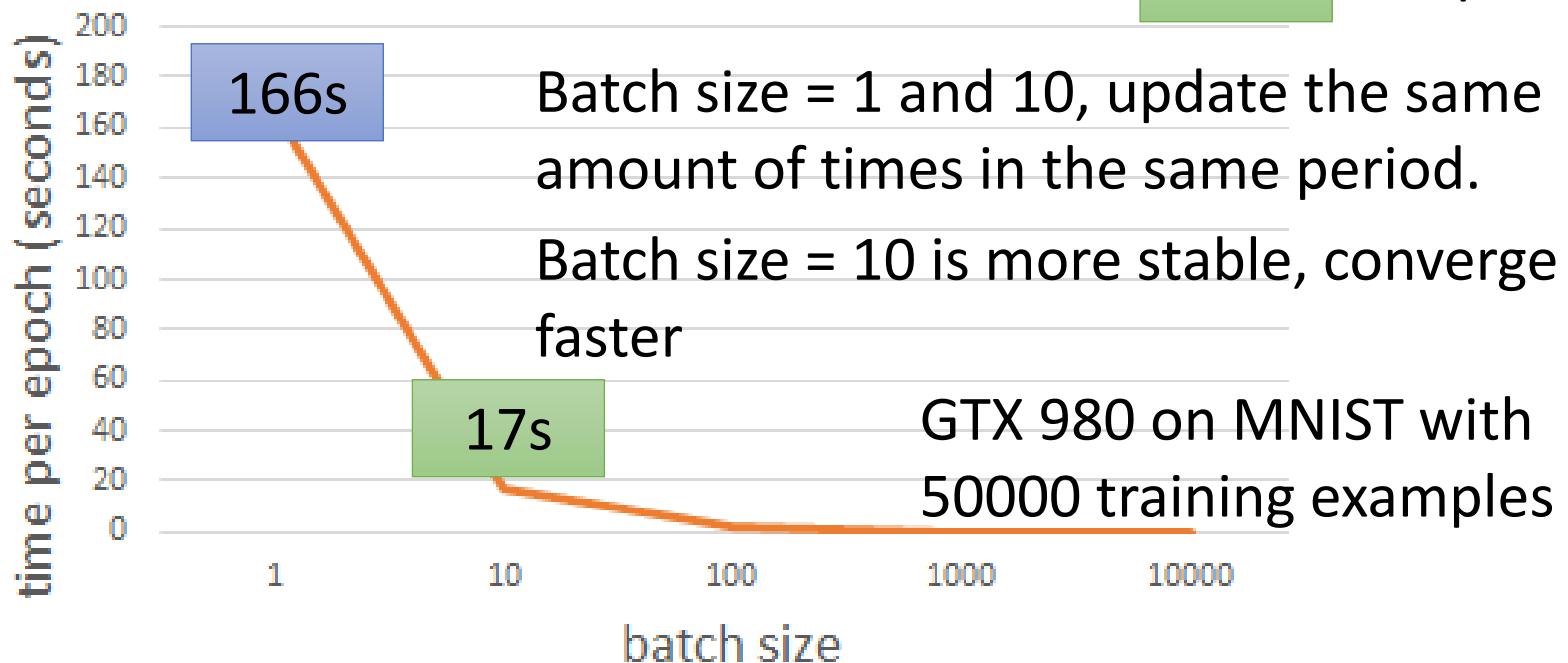
- E.g. 50000 examples

- batch size = 1, 50000 updates in one epoch

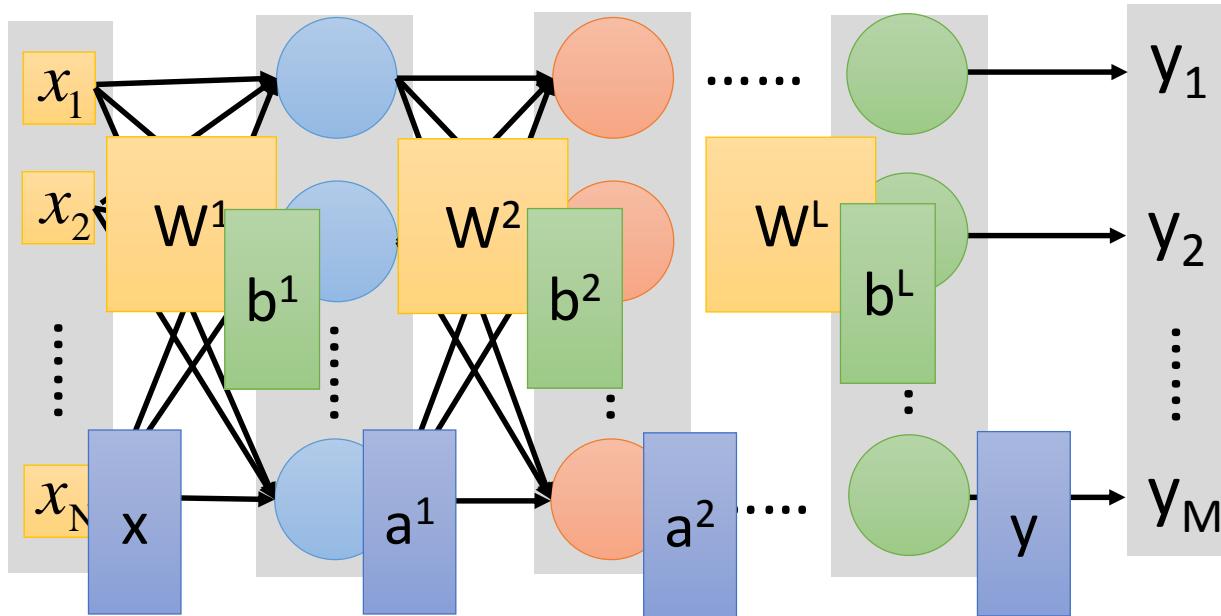
166s 1 epoch

- batch size = 10. 5000 updates in one epoch

17s 10 epoch



Speed - Matrix Operation



$y = f(x)$ Forward pass (Backward pass is similar)

$$= \sigma(W^L \dots \sigma(W^2 \sigma(W^1 x + b^1) + b^2) \dots + b^L)$$

Speed - Matrix Operation

- Why mini-batch is faster than stochastic gradient descent?

Stochastic Gradient Descent

$$z^1 = W^1 x$$
$$z^1 = W^1 x \dots\dots$$

Mini-batch

$$\begin{matrix} z^1 & z^1 \end{matrix} = W^1 \text{ matrix}$$
$$\begin{matrix} x & x \end{matrix}$$

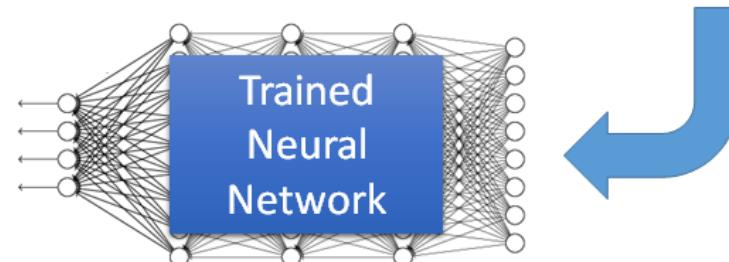
Practically, which one is faster?

Keras

Step 1:
define a set
of function

Step 2:
goodness of
function

Step 3: pick
the best
function



Save and load models

<https://faroit.github.io/keras-docs/2.0.2/getting-started/faq/#how-can-i-save-a-keras-model>

How to use the neural network (testing):

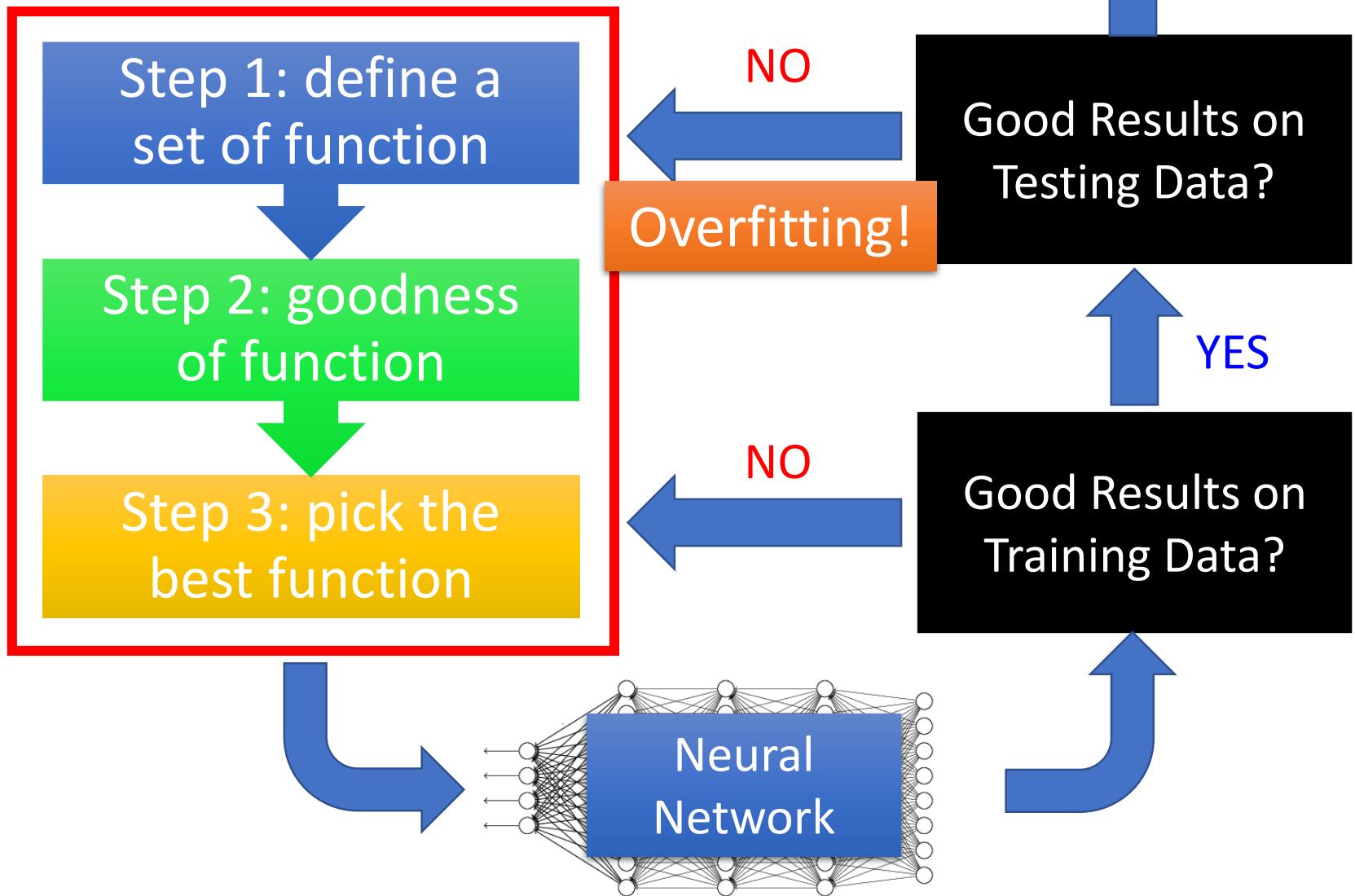
```
score = model.evaluate(x_test, y_test)
case 1: print('Total loss on Testing Set:', score[0])
         print('Accuracy of Testing Set:', score[1])
```

```
case 2: result = model.predict(x_test)
```

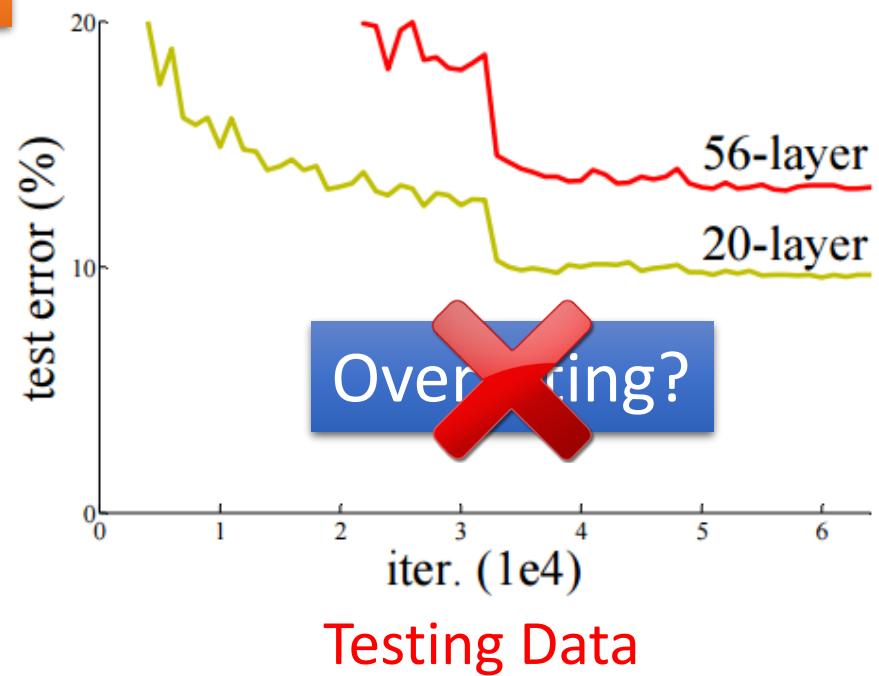
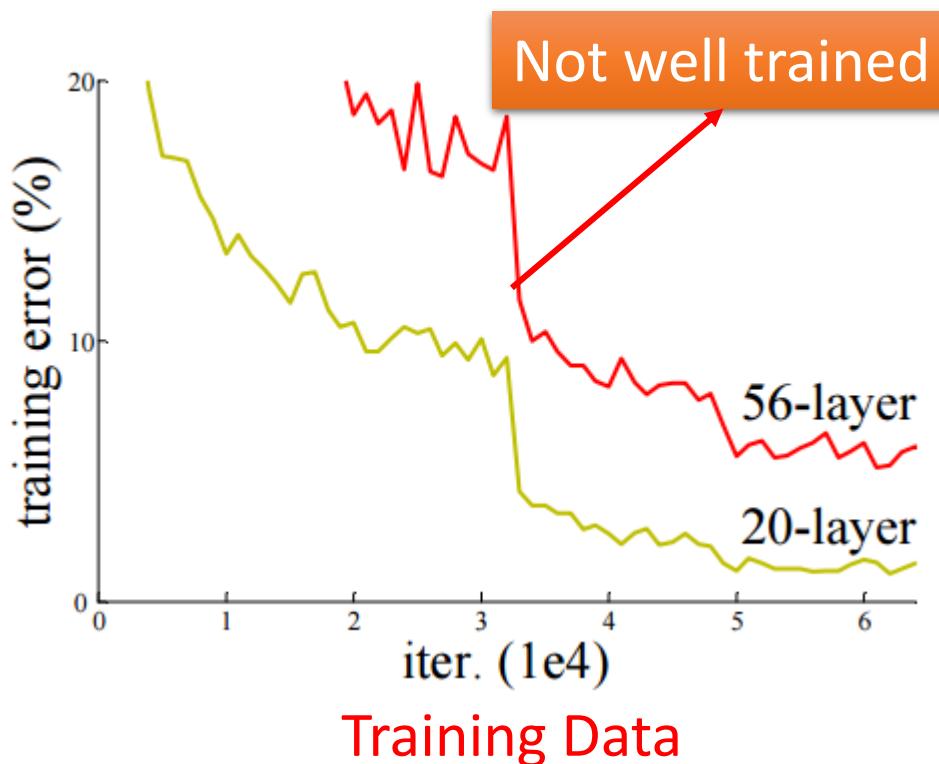
Live Demo

Tips for Deep Learning

Recipe of Deep Learning



Do not always blame Overfitting

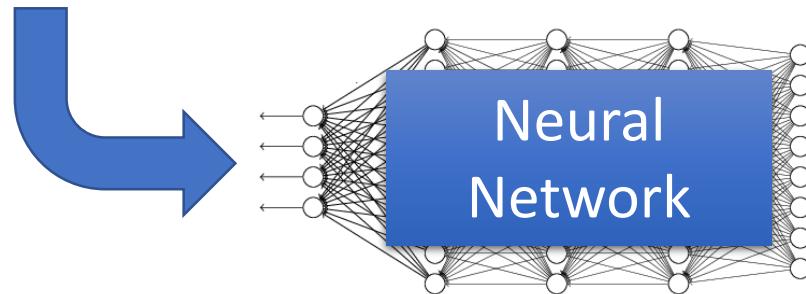


Deep Residual Learning for Image Recognition
<http://arxiv.org/abs/1512.03385>

Recipe of Deep Learning

Different approaches for different problems.

e.g. dropout for good results on testing data

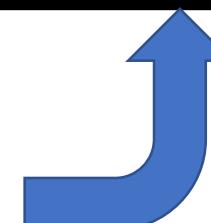
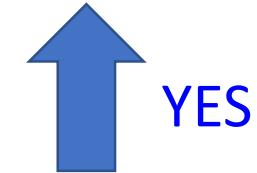


Good Results on Testing Data?

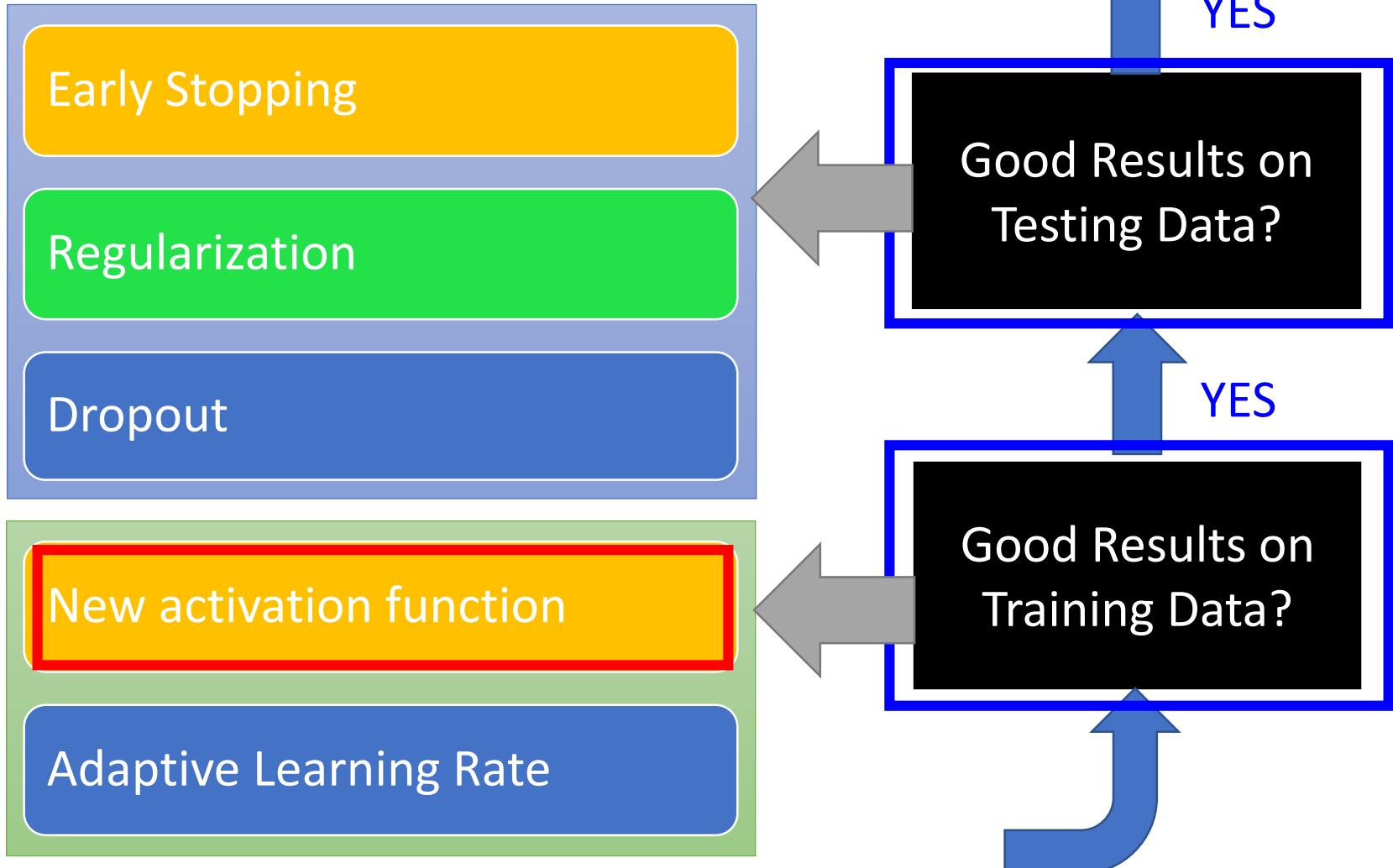
Good Results on Training Data?



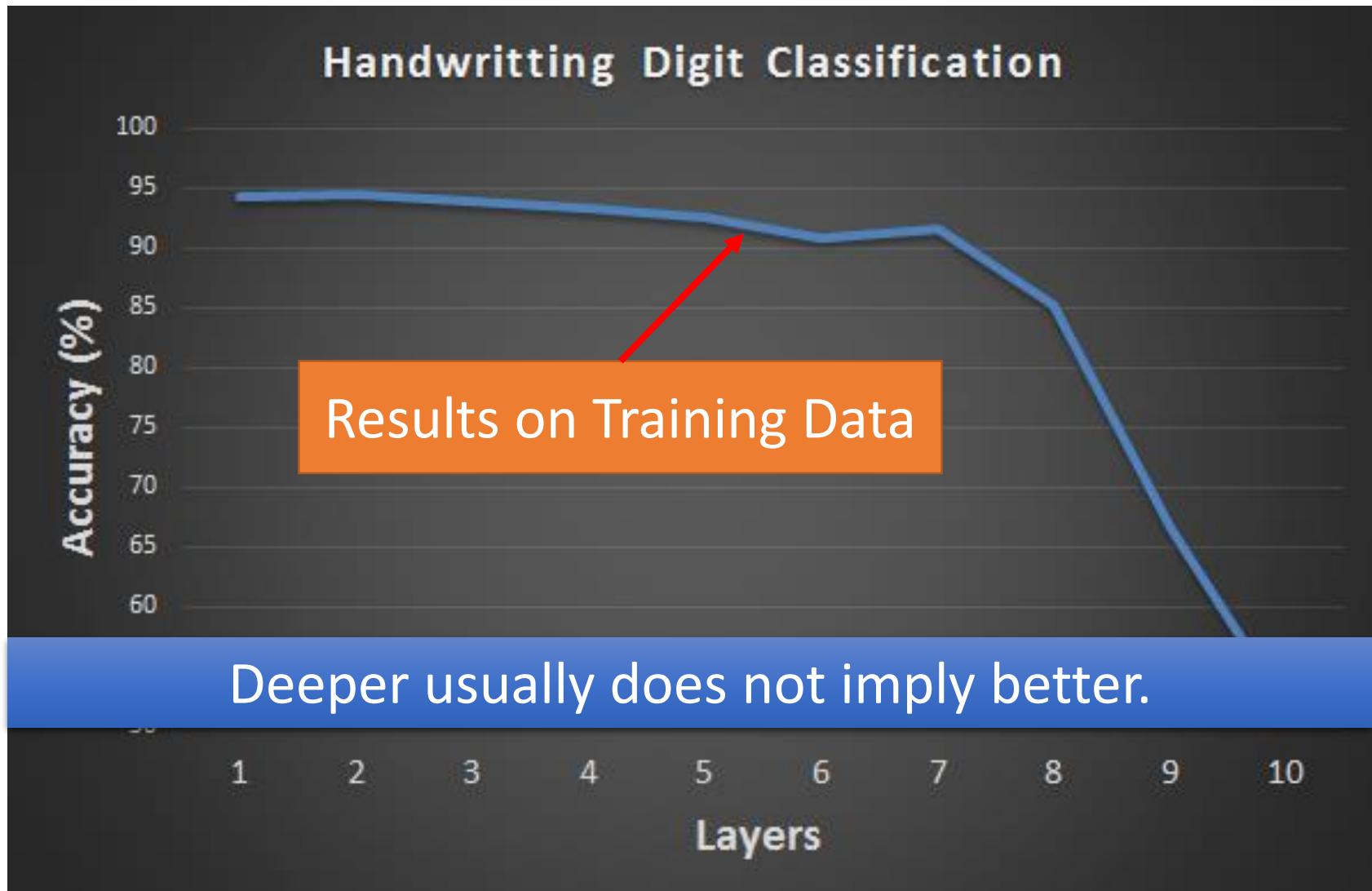
YES



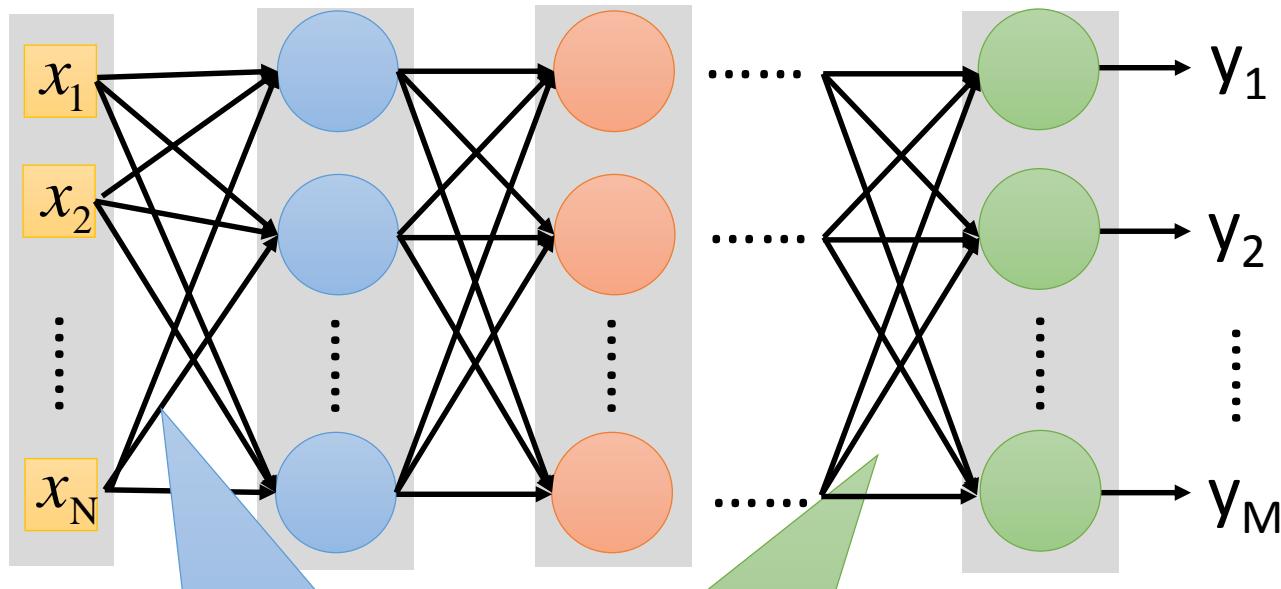
Recipe of Deep Learning



Hard to get the power of Deep ...



Vanishing Gradient Problem



Smaller gradients

Learn very slow

Almost random

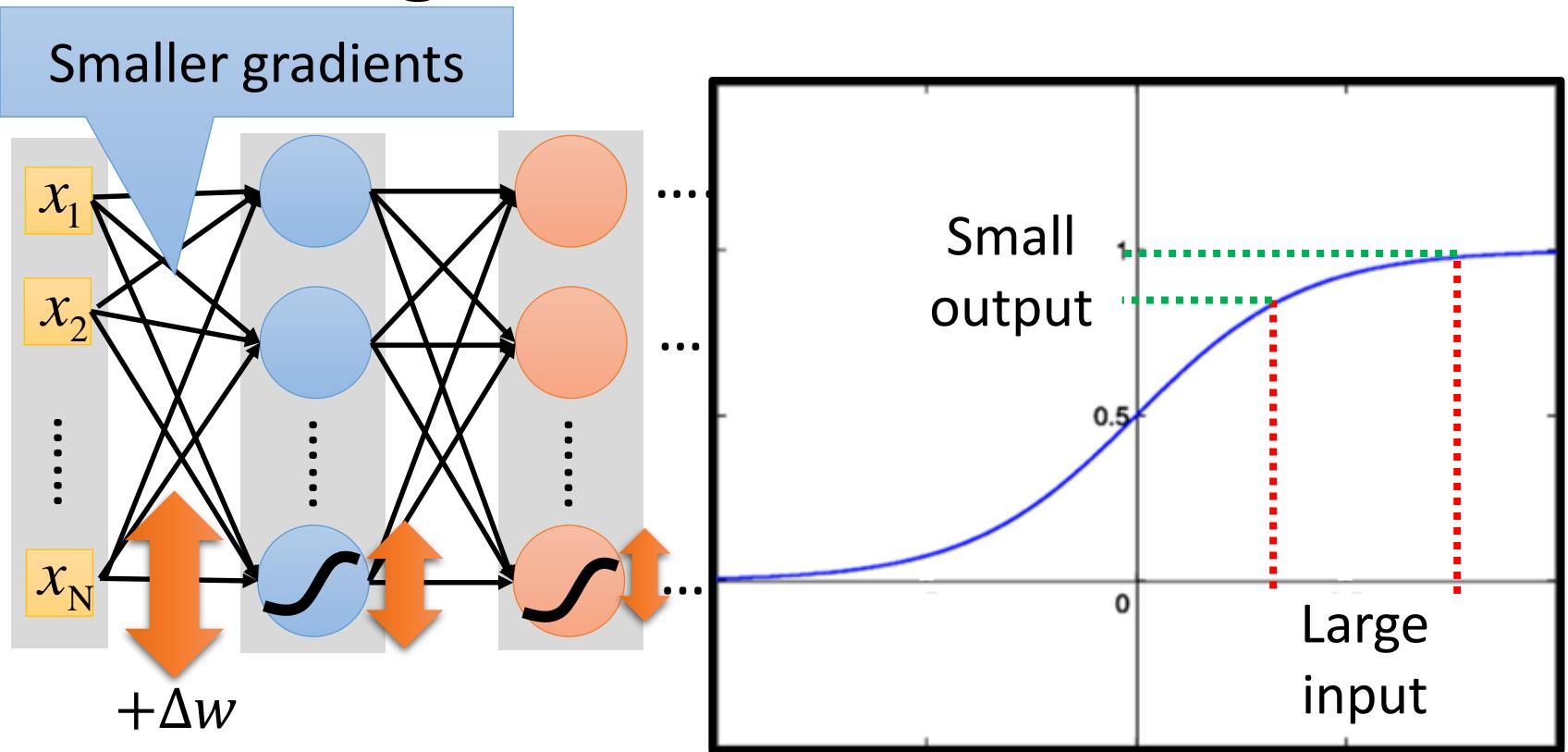
Larger gradients

Learn very fast

Already converge

based on random!?

Vanishing Gradient Problem

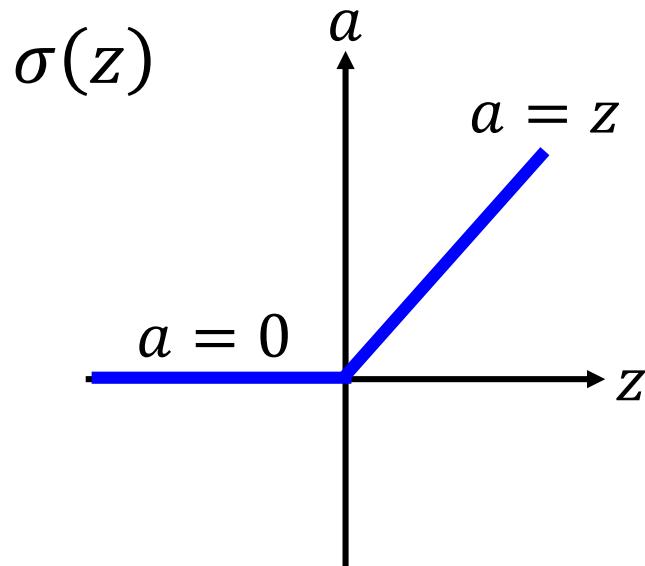


Intuitive way to compute the derivatives ...

$$\frac{\partial l}{\partial w} = ? \quad \frac{\Delta l}{\Delta w}$$

ReLU

- Rectified Linear Unit (ReLU)

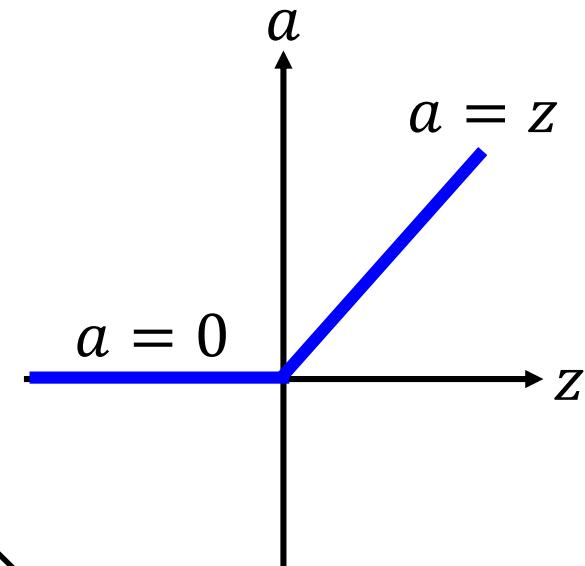
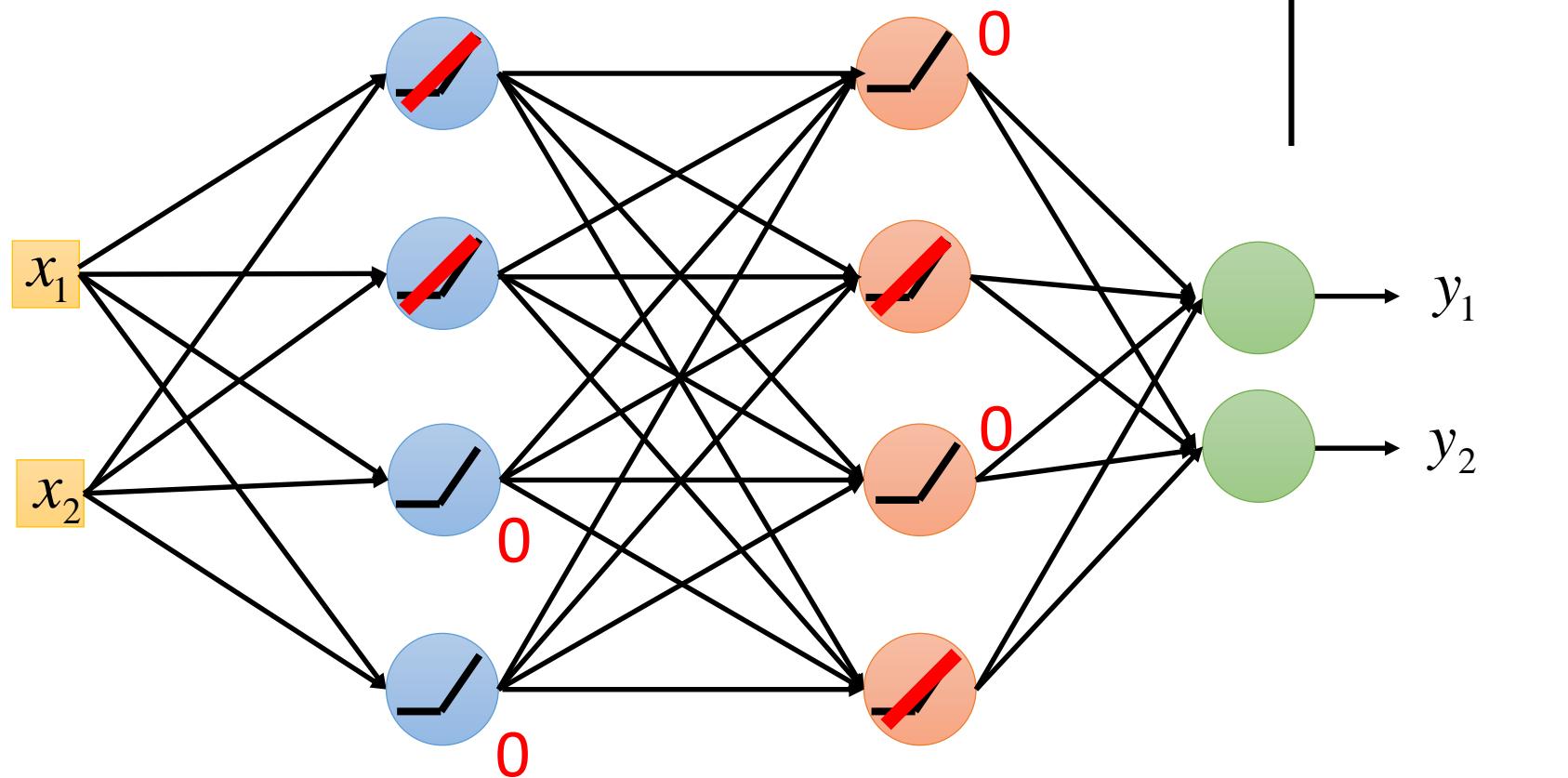


[Xavier Glorot, AISTATS'11]
[Andrew L. Maas, ICML'13]
[Kaiming He, arXiv'15]

Reason:

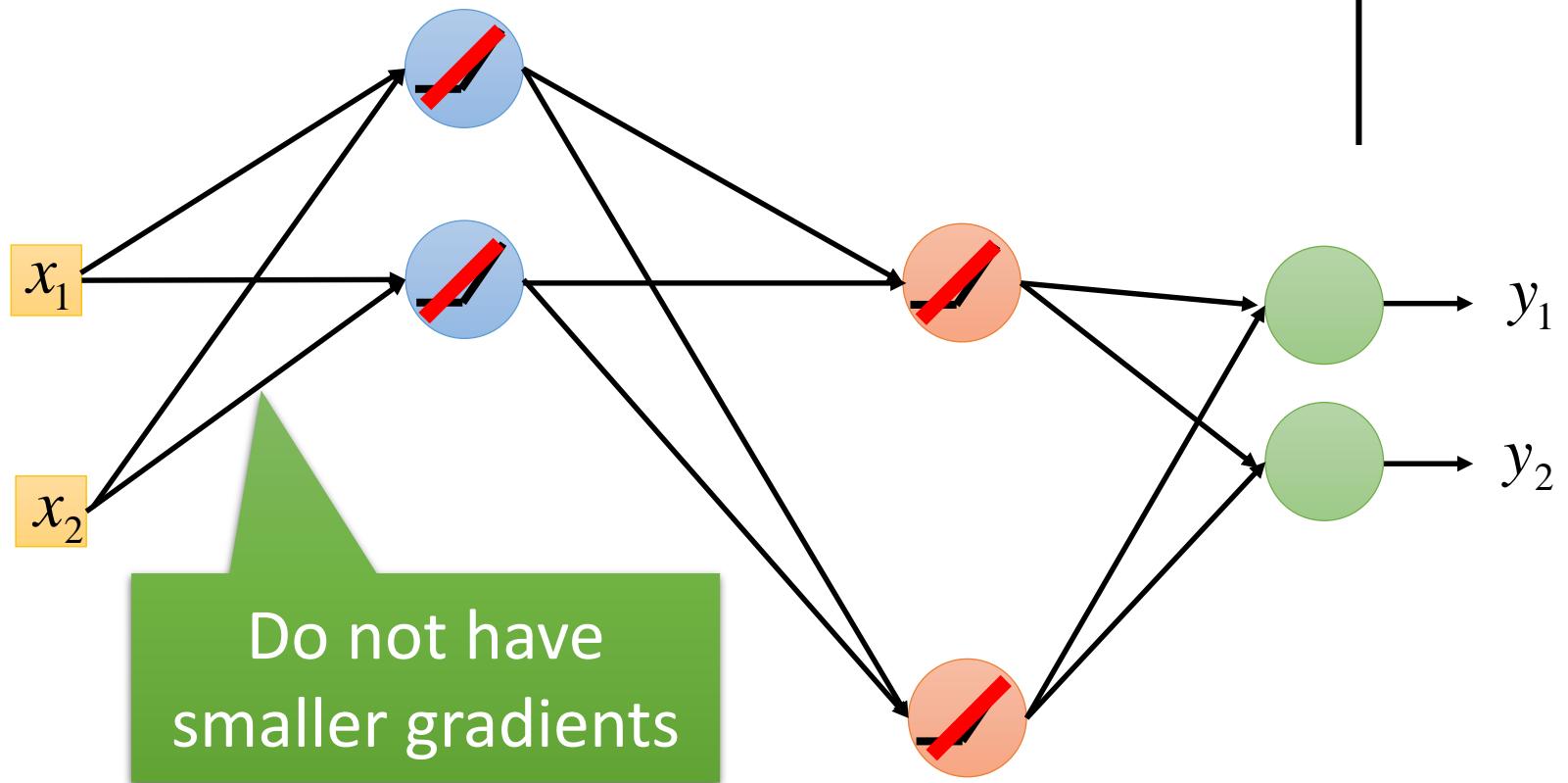
1. Fast to compute
2. Biological reason
3. Infinite sigmoid with different biases
4. Vanishing gradient problem

ReLU



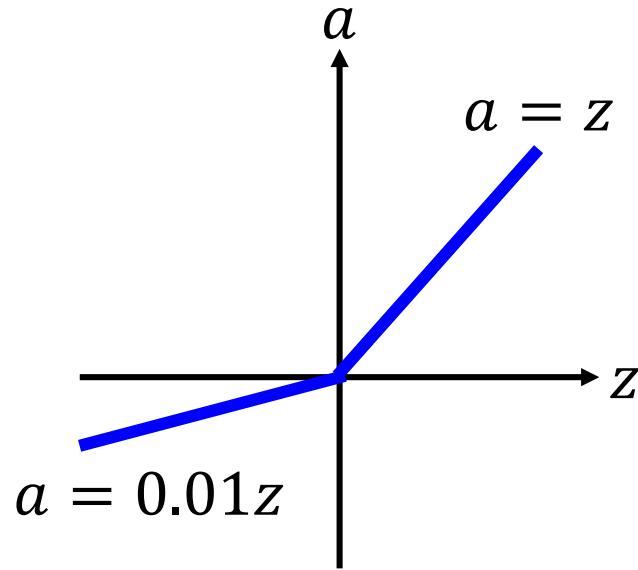
ReLU

A Thinner linear network

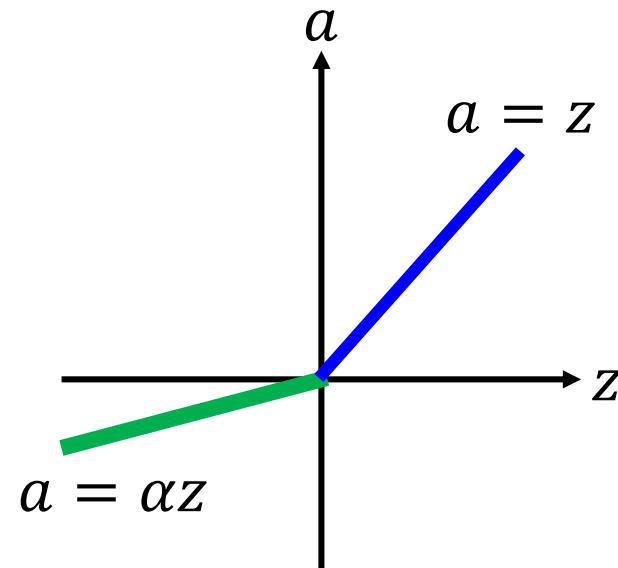


ReLU - variant

Leaky ReLU



Parametric ReLU

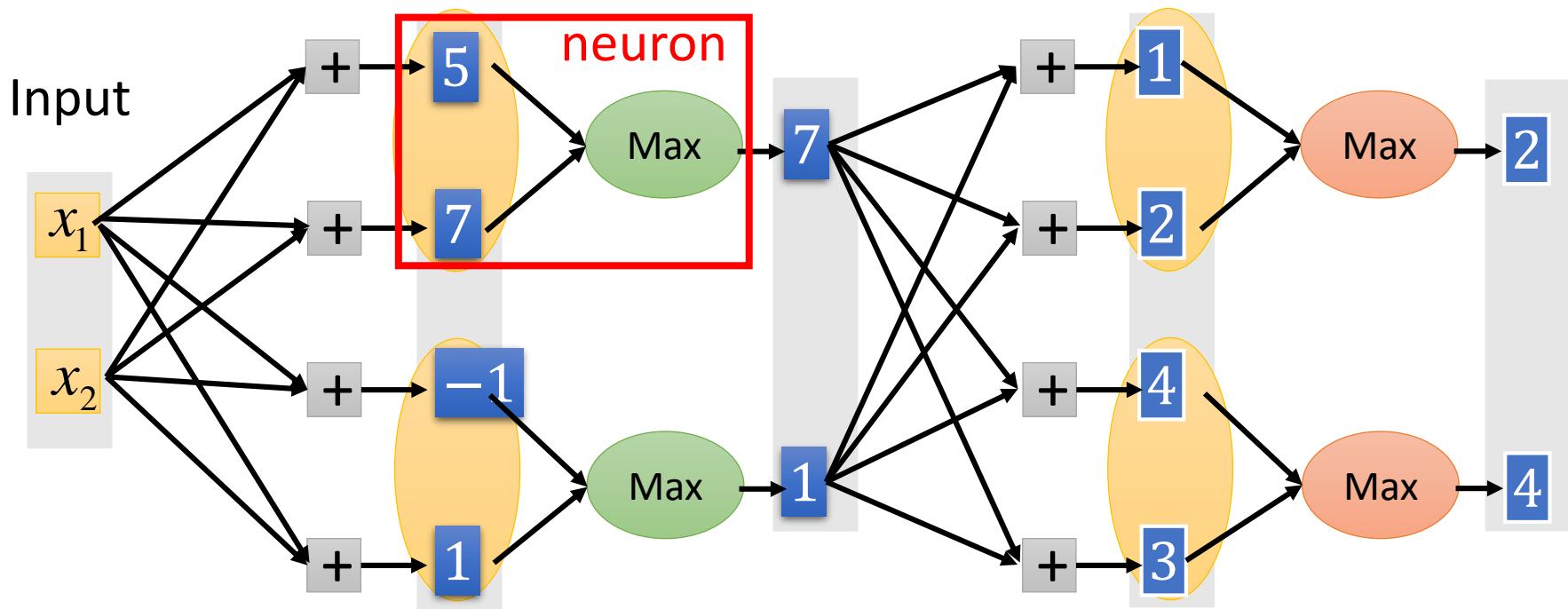


α also learned by
gradient descent

Maxout

ReLU is a special cases of Maxout

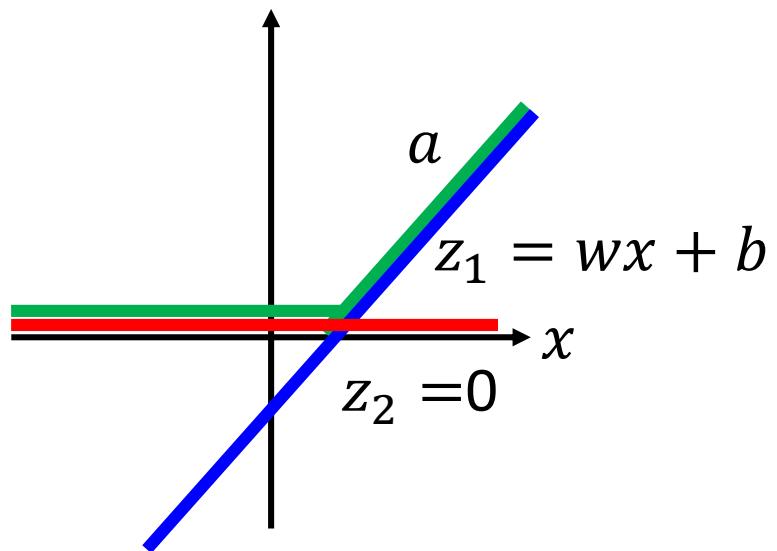
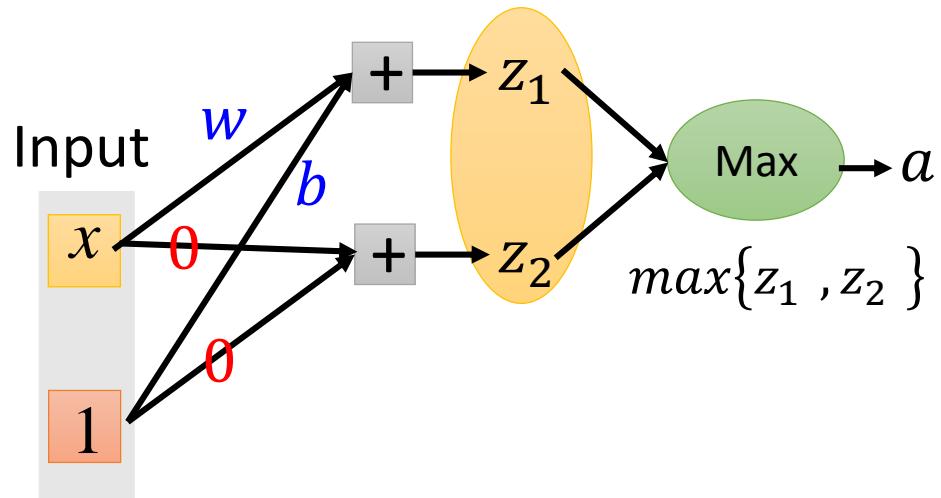
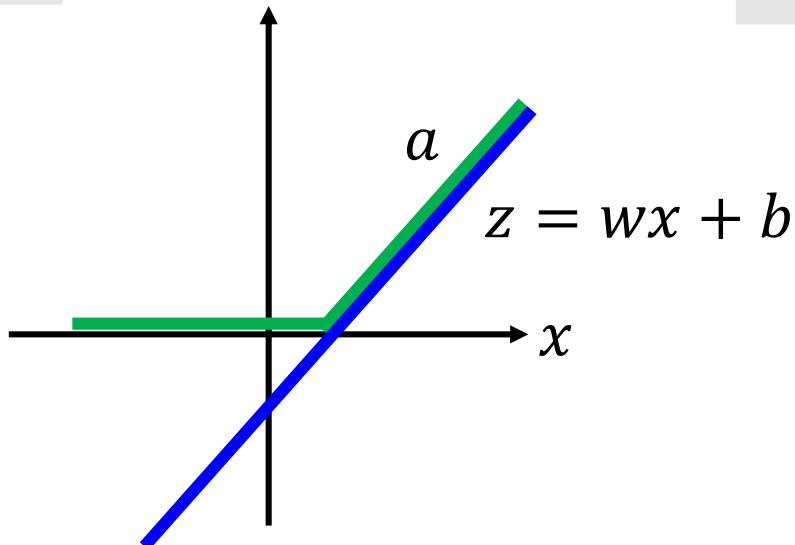
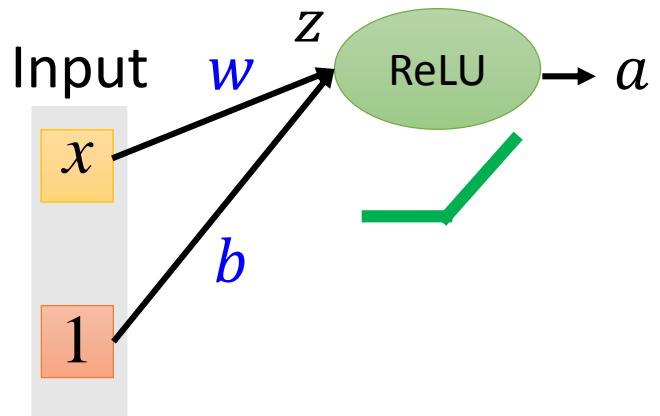
- Learnable activation function [Ian J. Goodfellow, ICML'13]



You can have more than 2 elements in a group.

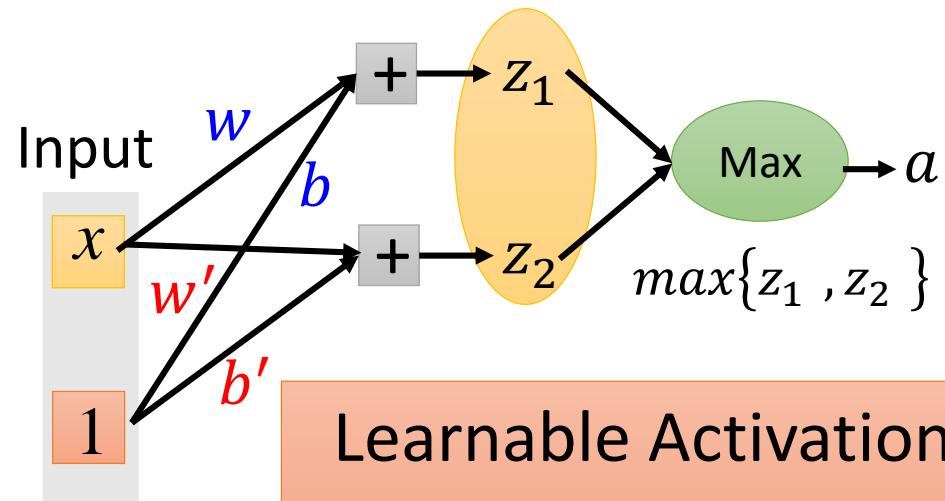
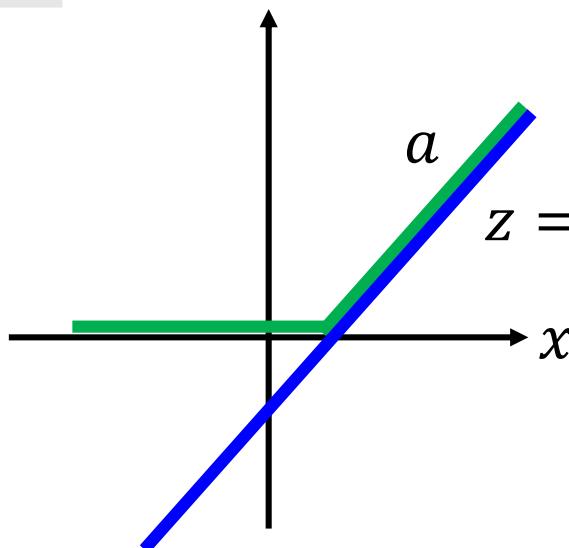
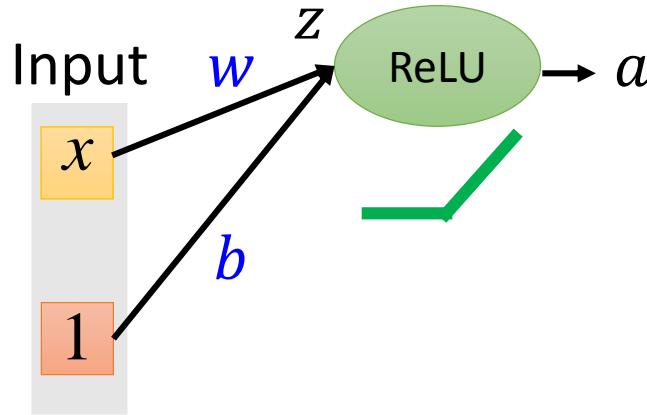
Maxout

ReLU is a special cases of Maxout

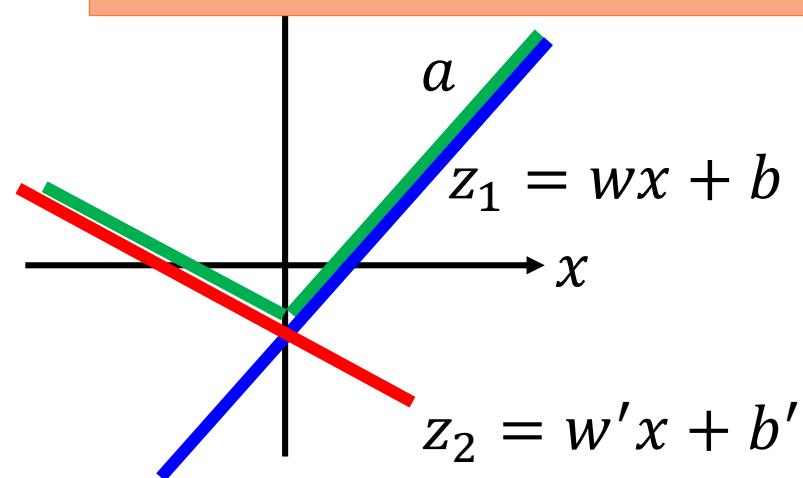


Maxout

More than ReLU



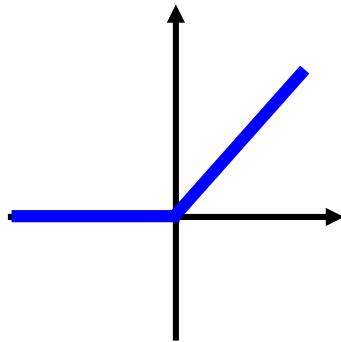
Learnable Activation Function



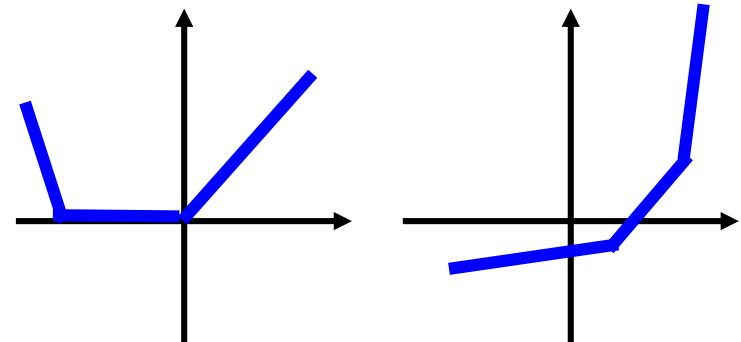
Maxout

- Learnable activation function [Ian J. Goodfellow, ICML'13]
 - Activation function in maxout network can be any piecewise linear convex function
 - How many pieces depending on how many elements in a group

2 elements in a group

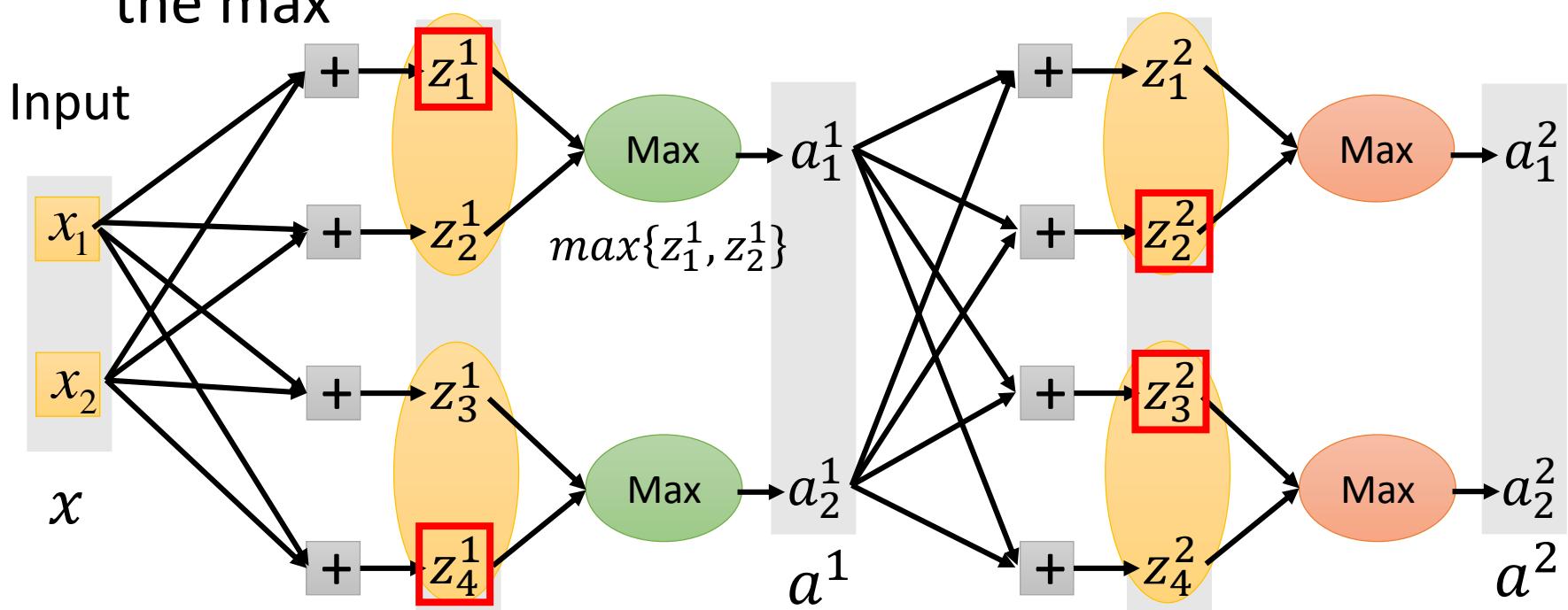


3 elements in a group



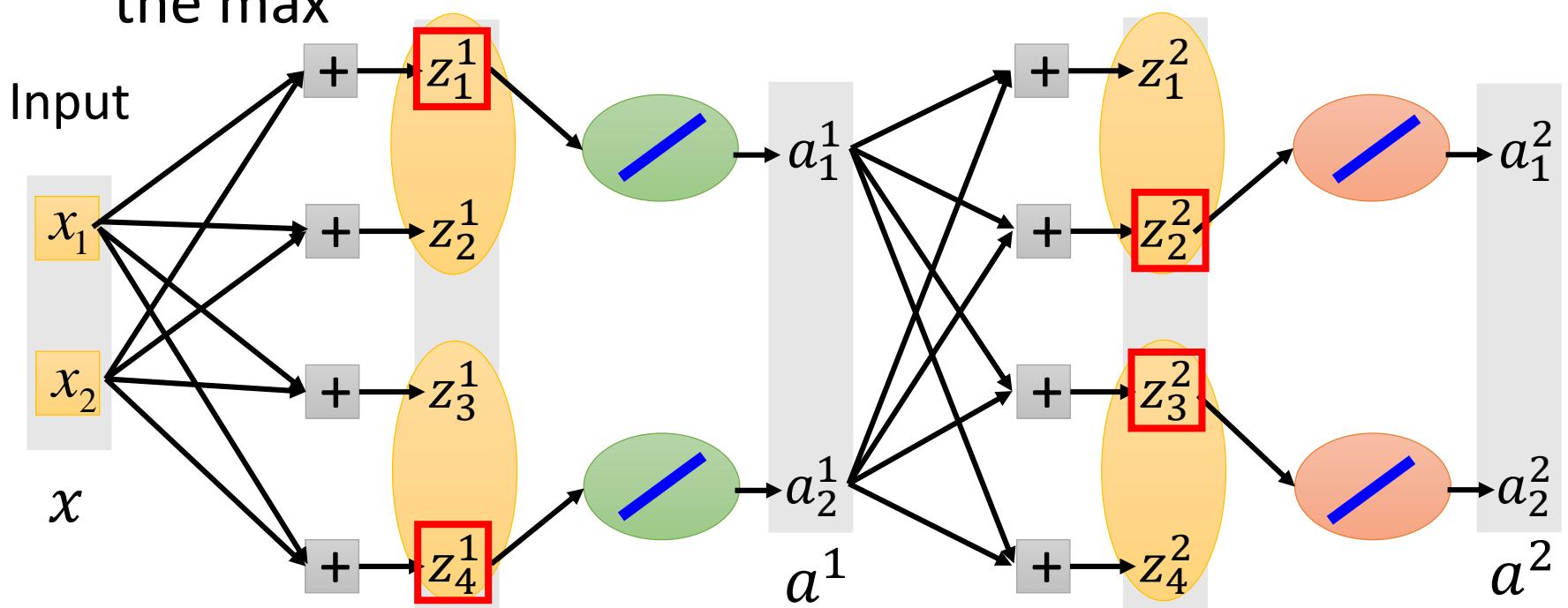
Maxout - Training

- Given a training data x , we know which z would be the max



Maxout - Training

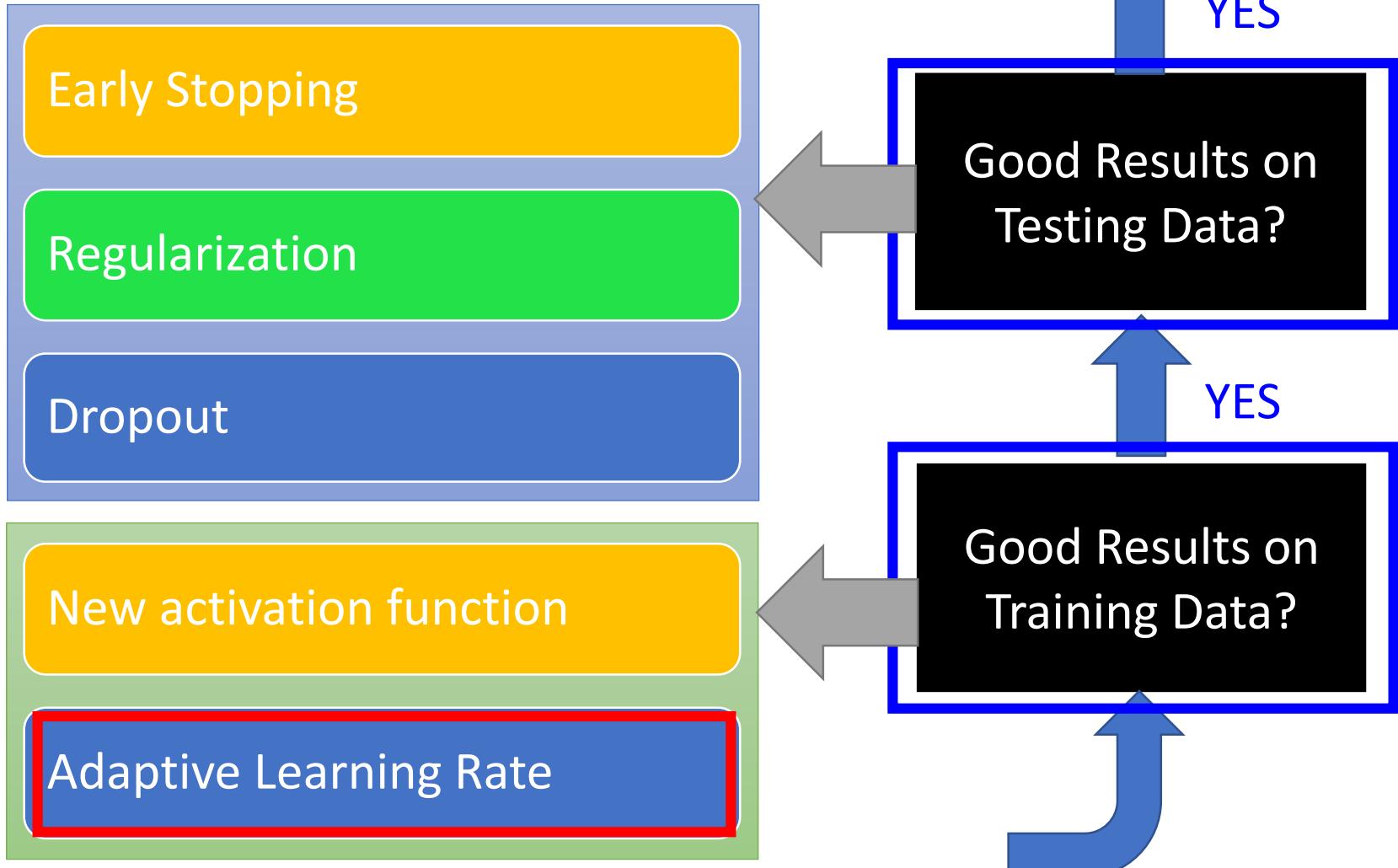
- Given a training data x , we know which z would be the max



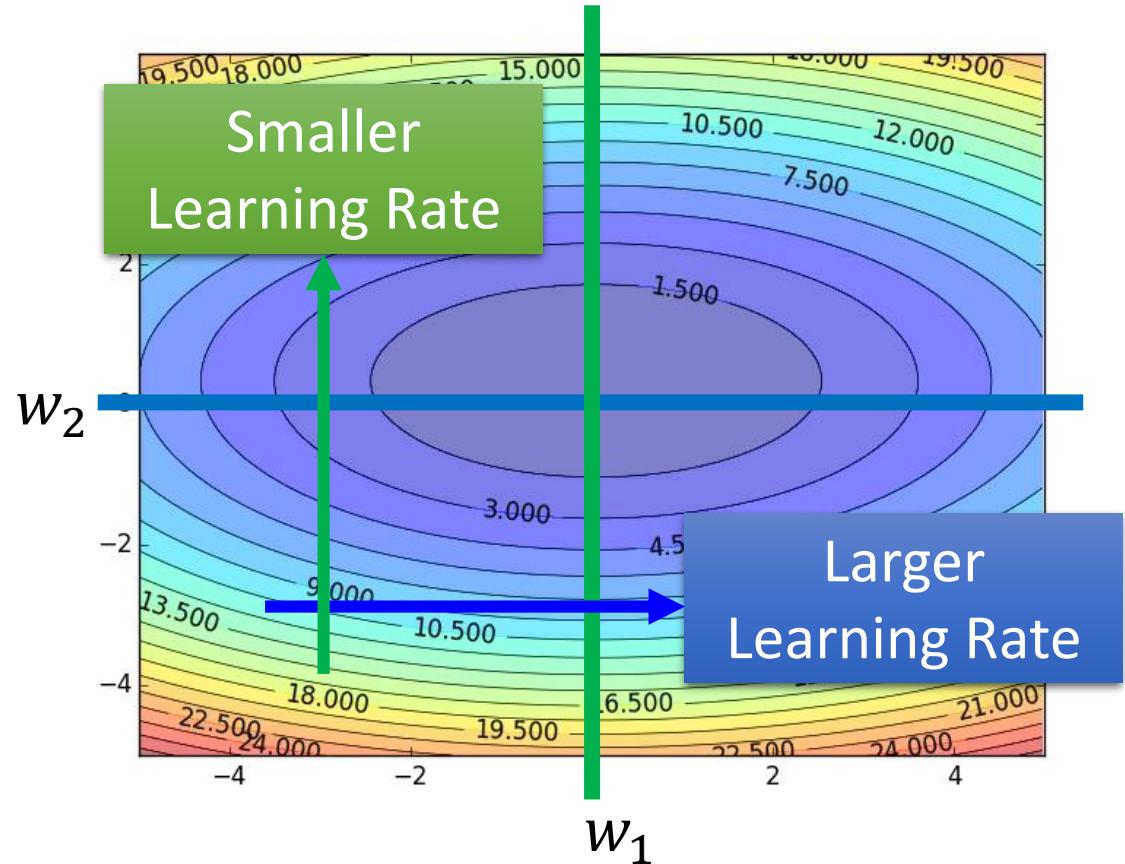
- Train this thin and linear network

Different thin and linear network for different examples

Recipe of Deep Learning



Review



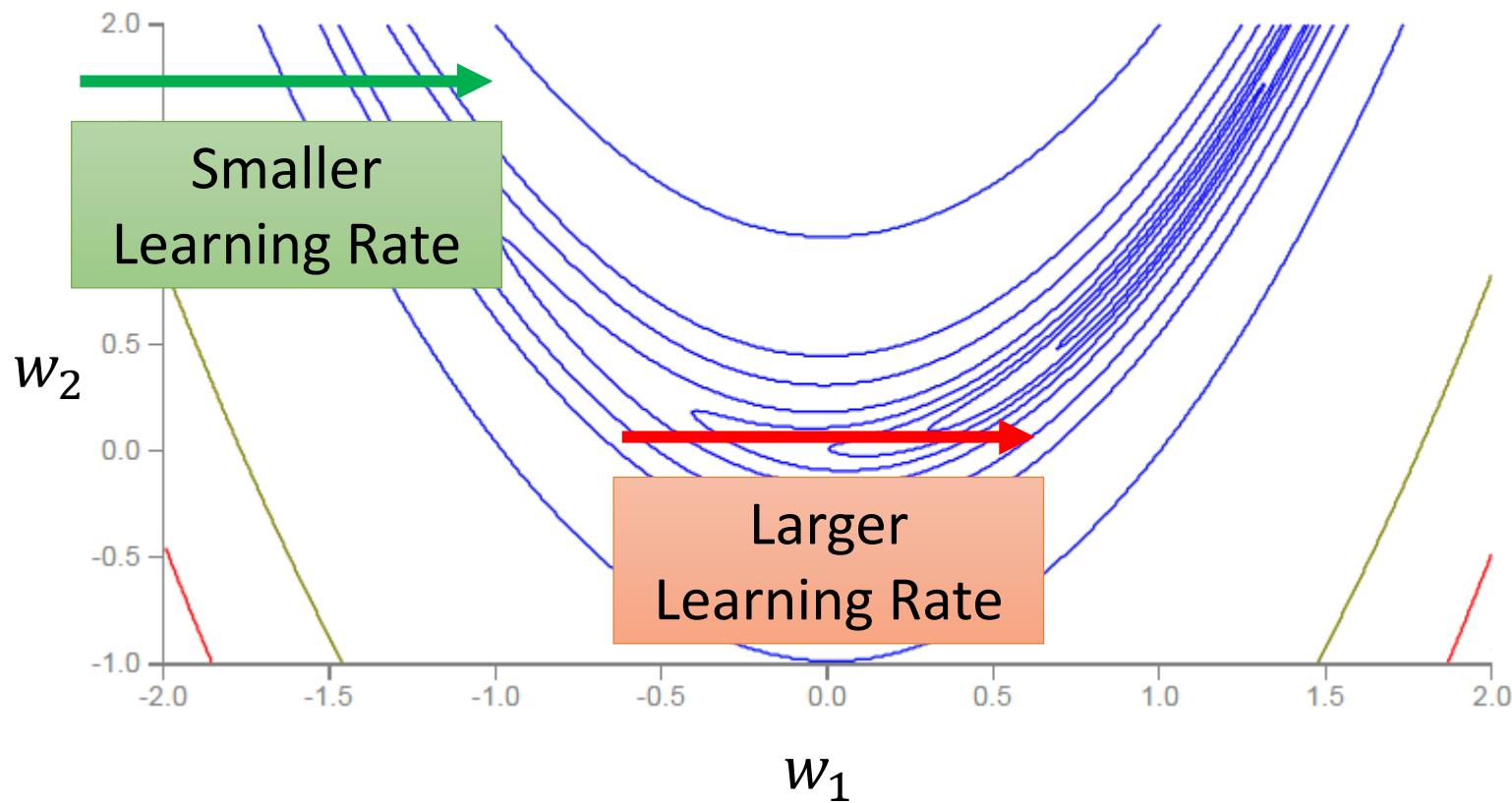
Adagrad

$$w^{t+1} \leftarrow w^t - \frac{\eta}{\sqrt{\sum_{i=0}^t (g^i)^2}} g^t$$

Use first derivative to estimate second derivative

RMSProp

Error Surface can be very complex when training NN.



RMSProp

$$w^1 \leftarrow w^0 - \frac{\eta}{\sigma^0} g^0 \quad \sigma^0 = g^0$$

$$w^2 \leftarrow w^1 - \frac{\eta}{\sigma^1} g^1 \quad \sigma^1 = \sqrt{\alpha(\sigma^0)^2 + (1 - \alpha)(g^1)^2}$$

$$w^3 \leftarrow w^2 - \frac{\eta}{\sigma^2} g^2 \quad \sigma^2 = \sqrt{\alpha(\sigma^1)^2 + (1 - \alpha)(g^2)^2}$$

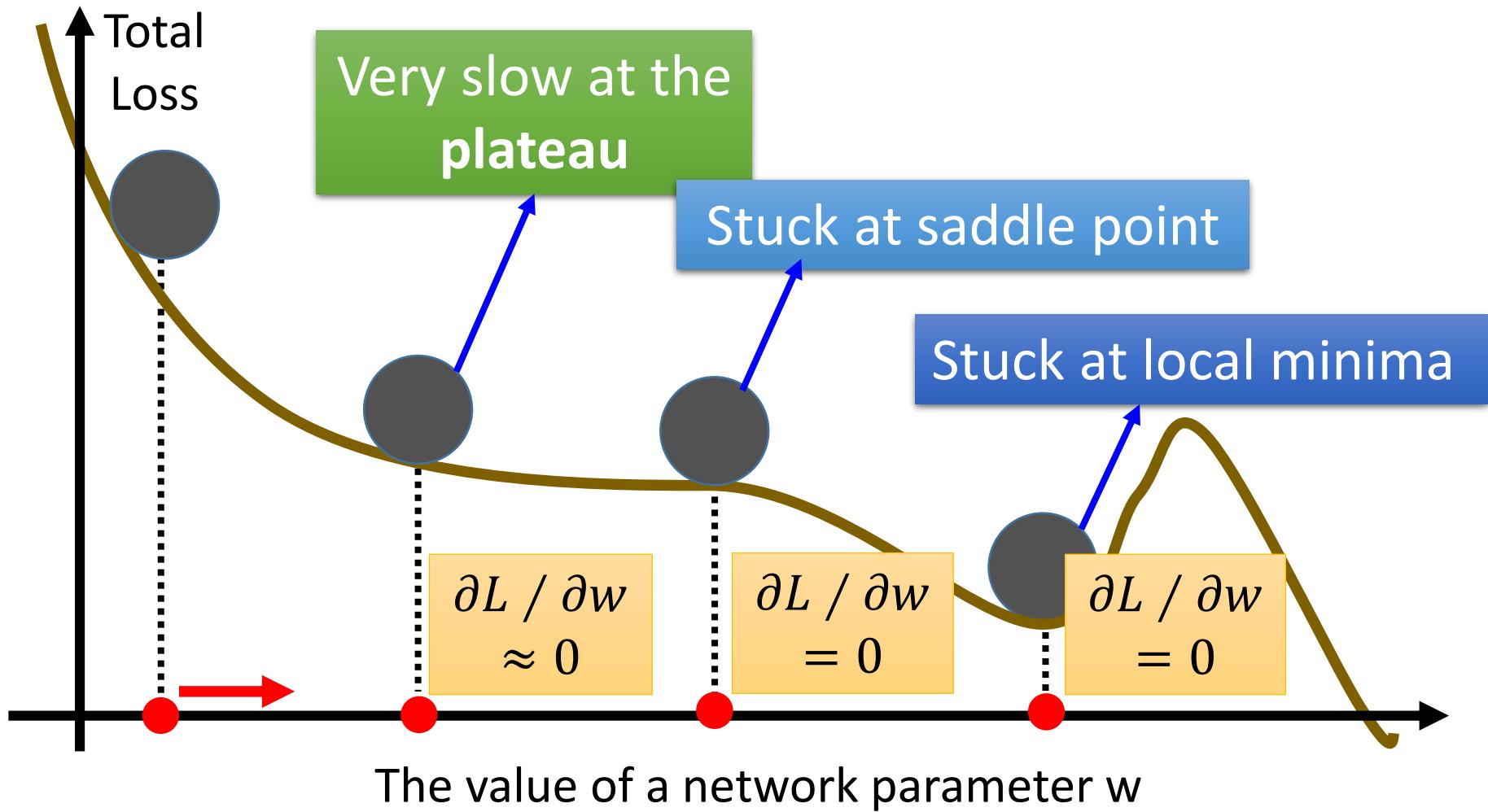
:

:

$$w^{t+1} \leftarrow w^t - \frac{\eta}{\sigma^t} g^t \quad \sigma^t = \sqrt{\alpha(\sigma^{t-1})^2 + (1 - \alpha)(g^t)^2}$$

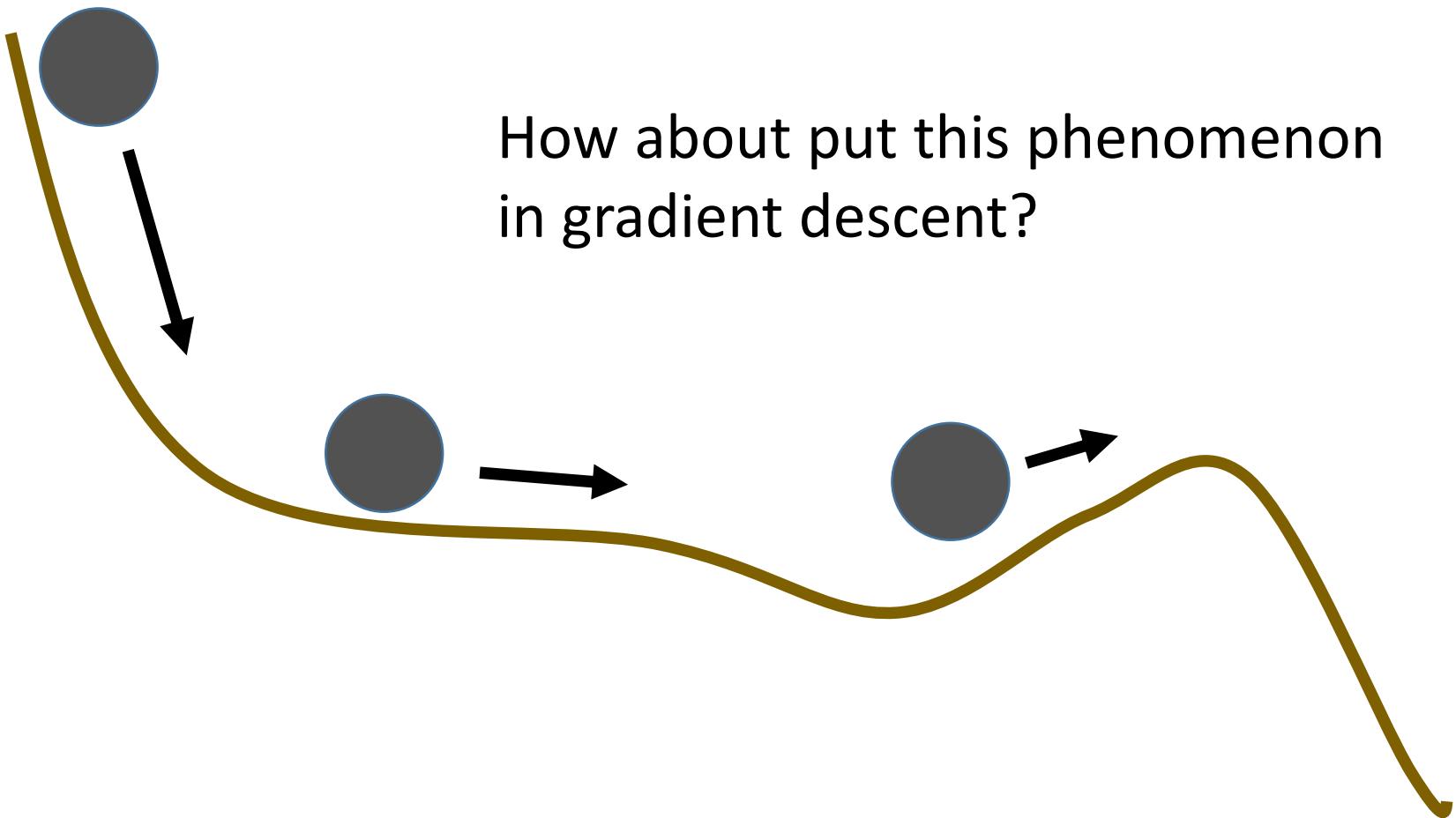
Root Mean Square of the gradients
with previous gradients being decayed

Hard to find optimal network parameters

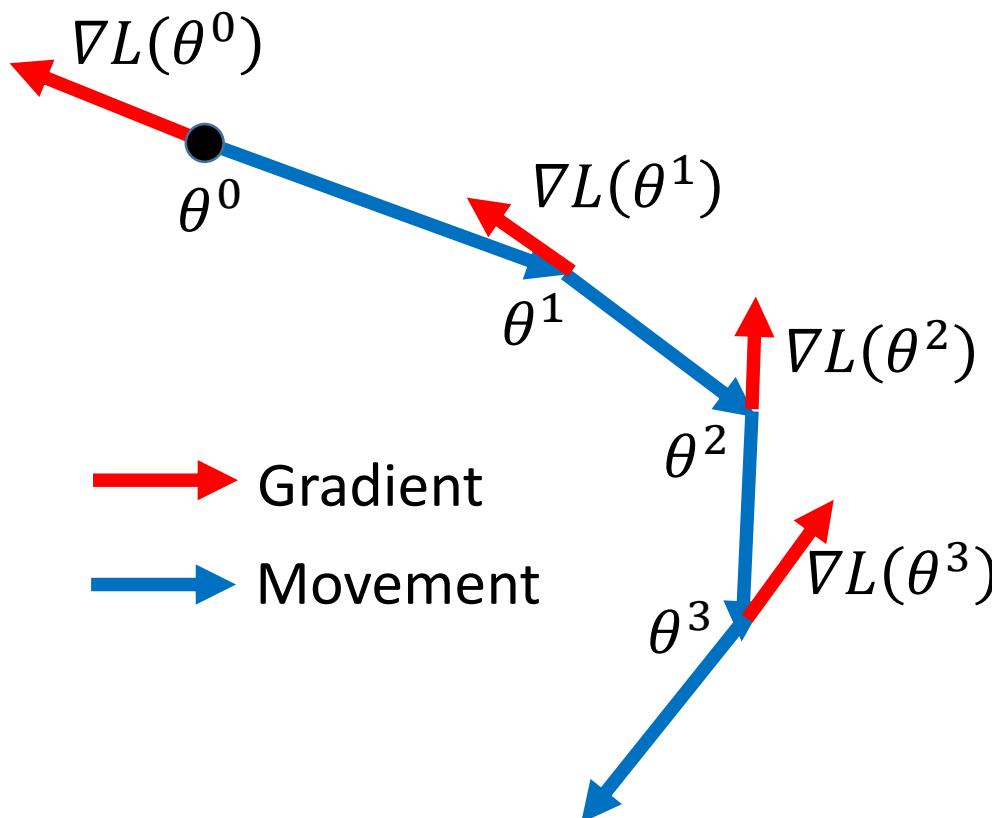


In physical world

- Momentum



Review: Vanilla Gradient Descent



Start at position θ^0

Compute gradient at θ^0

Move to $\theta^1 = \theta^0 - \eta \nabla L(\theta^0)$

Compute gradient at θ^1

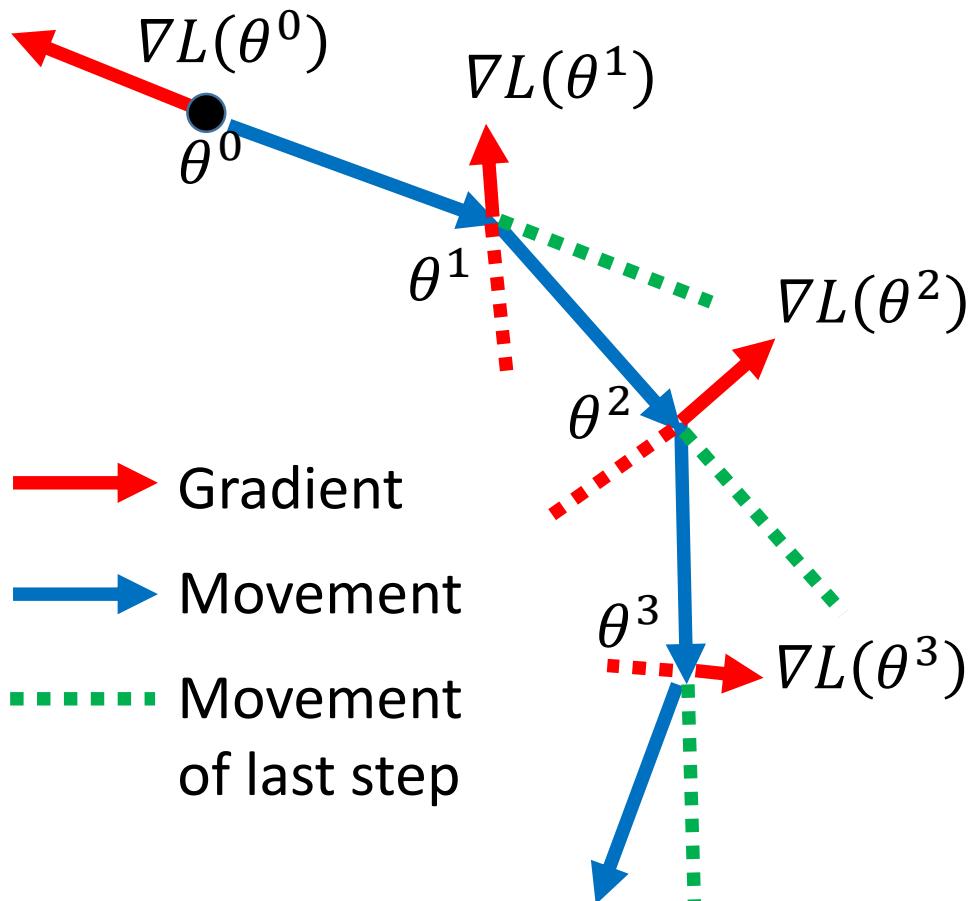
Move to $\theta^2 = \theta^1 - \eta \nabla L(\theta^1)$

⋮

Stop until $\nabla L(\theta^t) \approx 0$

Momentum

Movement: movement of last step minus gradient at present



Start at point θ^0

Movement $v^0=0$

Compute gradient at θ^0

Movement $v^1 = \lambda v^0 - \eta \nabla L(\theta^0)$

Move to $\theta^1 = \theta^0 + v^1$

Compute gradient at θ^1

Movement $v^2 = \lambda v^1 - \eta \nabla L(\theta^1)$

Move to $\theta^2 = \theta^1 + v^2$

Movement not just based on gradient, but previous movement.

Momentum

Movement: movement of last step minus gradient at present

v^i is actually the weighted sum of all the previous gradient:
 $\nabla L(\theta^0), \nabla L(\theta^1), \dots \nabla L(\theta^{i-1})$

$$v^0 = 0$$

$$v^1 = -\eta \nabla L(\theta^0)$$

$$v^2 = -\lambda \eta \nabla L(\theta^0) - \eta \nabla L(\theta^1)$$

⋮

Start at point θ^0

Movement $v^0=0$

Compute gradient at θ^0

Movement $v^1 = \lambda v^0 - \eta \nabla L(\theta^0)$

Move to $\theta^1 = \theta^0 + v^1$

Compute gradient at θ^1

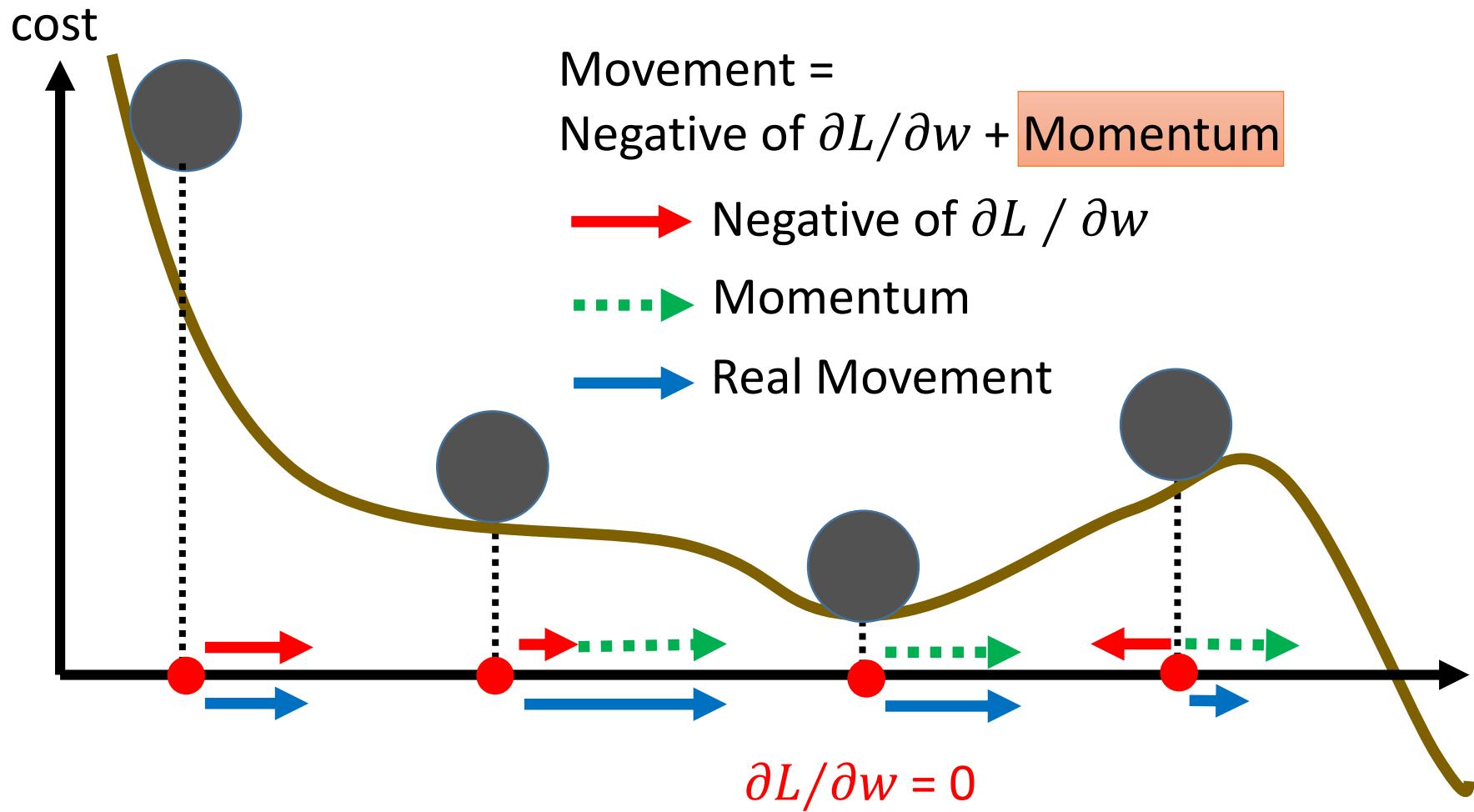
Movement $v^2 = \lambda v^1 - \eta \nabla L(\theta^1)$

Move to $\theta^2 = \theta^1 + v^2$

Movement not just based on gradient, but previous movement

Momentum

Still not guarantee reaching global minima, but give some hope



Adam

RMSProp + Momentum

Algorithm 1: Adam, our proposed algorithm for stochastic optimization. See section 2 for details, and for a slightly more efficient (but less clear) order of computation. g_t^2 indicates the elementwise square $g_t \odot g_t$. Good default settings for the tested machine learning problems are $\alpha = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$. All operations on vectors are element-wise. With β_1^t and β_2^t we denote β_1 and β_2 to the power t .

Require: α : Stepsize

Require: $\beta_1, \beta_2 \in [0, 1]$: Exponential decay rates for the moment estimates

Require: $f(\theta)$: Stochastic objective function with parameters θ

Require: θ_0 : Initial parameter vector

$m_0 \leftarrow 0$ (Initialize 1st moment vector)

→ for momentum

$v_0 \leftarrow 0$ (Initialize 2nd moment vector)

→ for RMSprop

$t \leftarrow 0$ (Initialize timestep)

while θ_t not converged **do**

$t \leftarrow t + 1$

$g_t \leftarrow \nabla_{\theta} f_t(\theta_{t-1})$ (Get gradients w.r.t. stochastic objective at timestep t)

$m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$ (Update biased first moment estimate)

$v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$ (Update biased second raw moment estimate)

$\hat{m}_t \leftarrow m_t / (1 - \beta_1^t)$ (Compute bias-corrected first moment estimate)

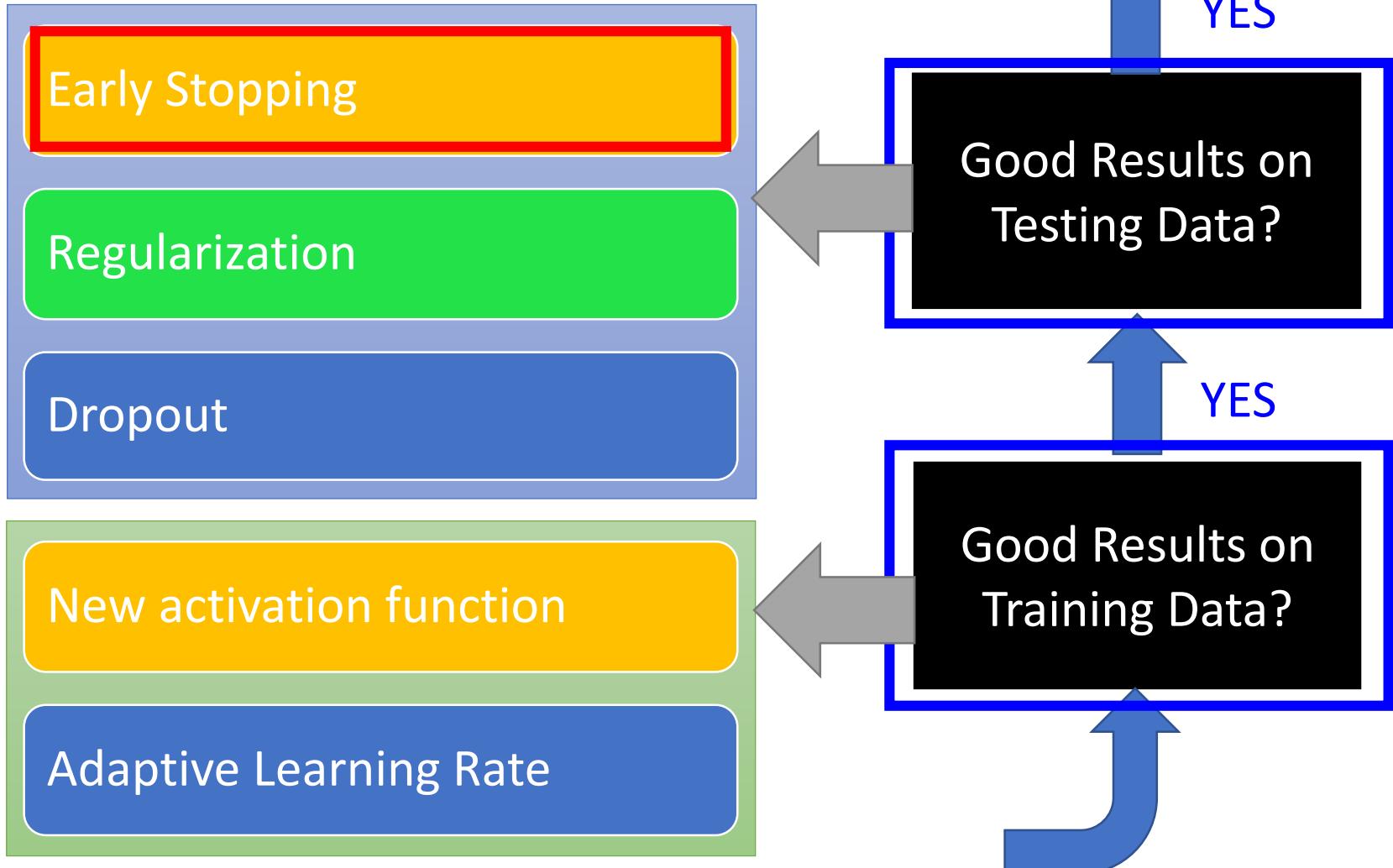
$\hat{v}_t \leftarrow v_t / (1 - \beta_2^t)$ (Compute bias-corrected second raw moment estimate)

$\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \hat{m}_t / (\sqrt{\hat{v}_t} + \epsilon)$ (Update parameters)

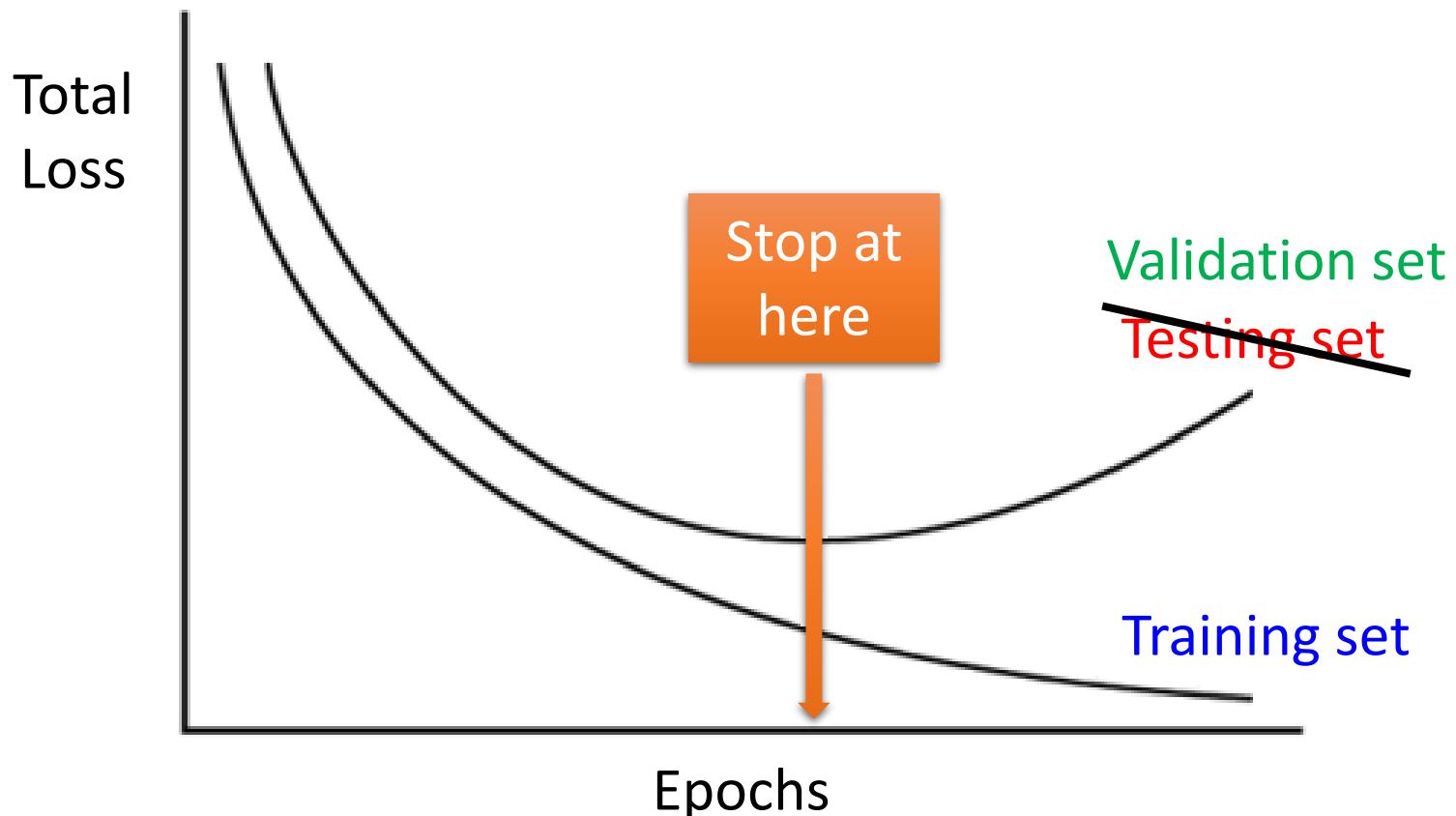
end while

return θ_t (Resulting parameters)

Recipe of Deep Learning

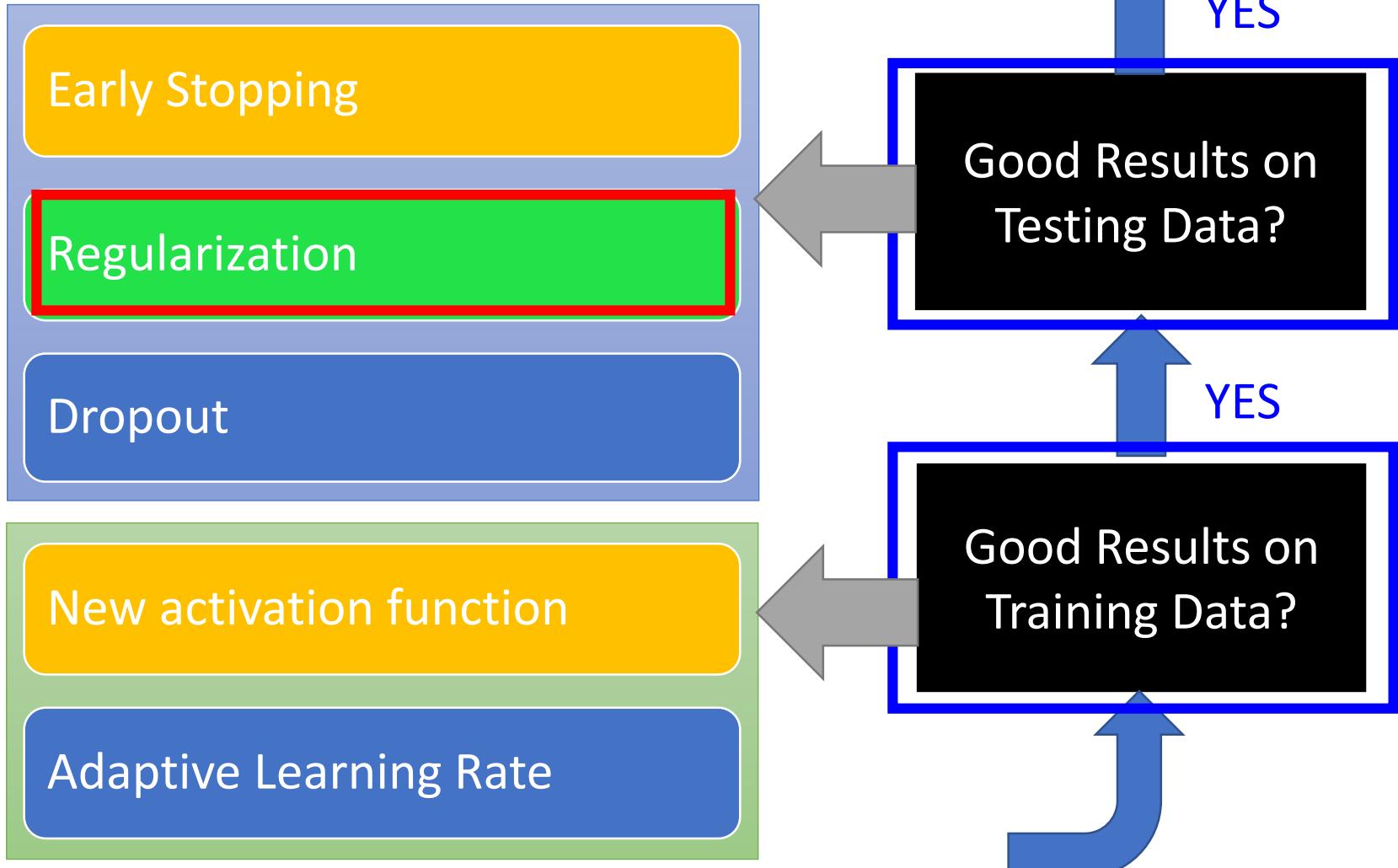


Early Stopping



Keras: <http://keras.io/getting-started/faq/#how-can-i-interrupt-training-when-the-validation-loss-isnt-decreasing-anymore>

Recipe of Deep Learning



Regularization

- New loss function to be minimized
 - Find a set of weight not only minimizing original cost but also close to zero

$$L'(\theta) = \underline{L(\theta)} + \lambda \frac{1}{2} \underline{\|\theta\|_2} \rightarrow \text{Regularization term}$$

Original loss
(e.g. minimize square error, cross entropy ...)

$$\theta = \{w_1, w_2, \dots\}$$

L2 regularization:

$$\|\theta\|_2 = (w_1)^2 + (w_2)^2 + \dots$$

(usually not consider biases)

Regularization

L2 regularization:

$$\|\theta\|_2 = (w_1)^2 + (w_2)^2 + \dots$$

- New loss function to be minimized

$$L'(\theta) = L(\theta) + \lambda \frac{1}{2} \|\theta\|_2 \quad \text{Gradient: } \frac{\partial L'}{\partial w} = \frac{\partial L}{\partial w} + \lambda w$$

Update: $w^{t+1} \rightarrow w^t - \eta \frac{\partial L'}{\partial w} = w^t - \eta \left(\frac{\partial L}{\partial w} + \lambda w^t \right)$

$$= \underbrace{(1 - \eta \lambda)w^t}_{\downarrow} - \eta \underbrace{\frac{\partial L}{\partial w}}$$

Weight Decay

Closer to zero

Regularization

L1 regularization:

$$\|\theta\|_1 = |w_1| + |w_2| + \dots$$

- New loss function to be minimized

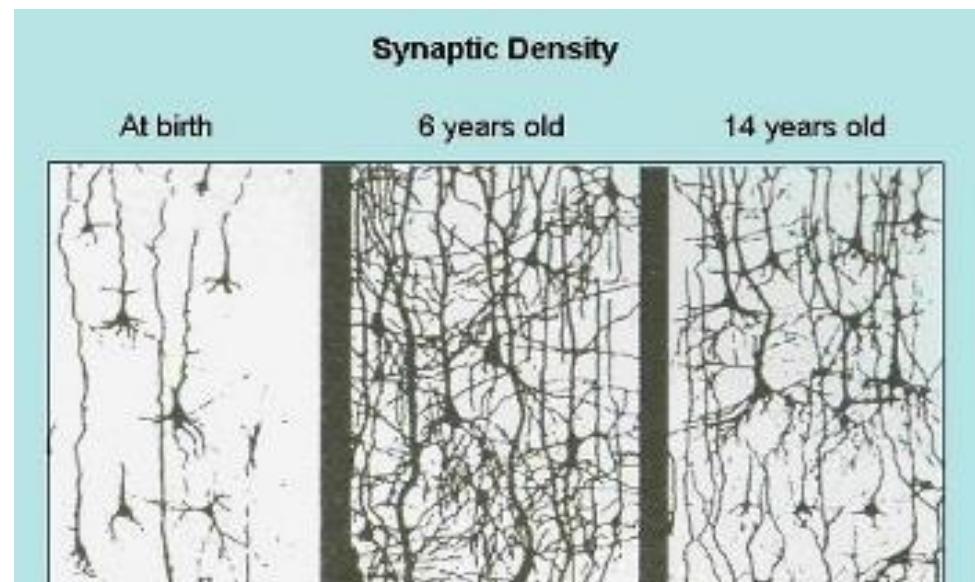
$$L'(\theta) = L(\theta) + \lambda \frac{1}{2} \|\theta\|_1 \quad \frac{\partial L'}{\partial w} = \frac{\partial L}{\partial w} + \lambda \operatorname{sgn}(w)$$

Update:

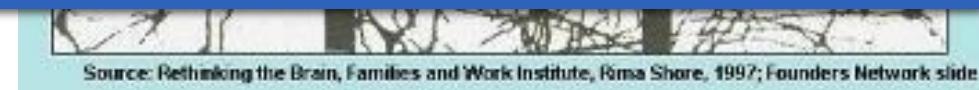
$$\begin{aligned} w^{t+1} &\rightarrow w^t - \eta \frac{\partial L'}{\partial w} = w^t - \eta \left(\frac{\partial L}{\partial w} + \lambda \operatorname{sgn}(w^t) \right) \\ &= w^t - \eta \frac{\partial L}{\partial w} - \underline{\eta \lambda \operatorname{sgn}(w^t)} \quad \text{Always delete} \\ &= (1 - \eta \lambda) w^t - \eta \frac{\partial L}{\partial w} \quad \dots \text{L2} \end{aligned}$$

Regularization - Weight Decay

- Our brain prunes out the useless link between neurons.

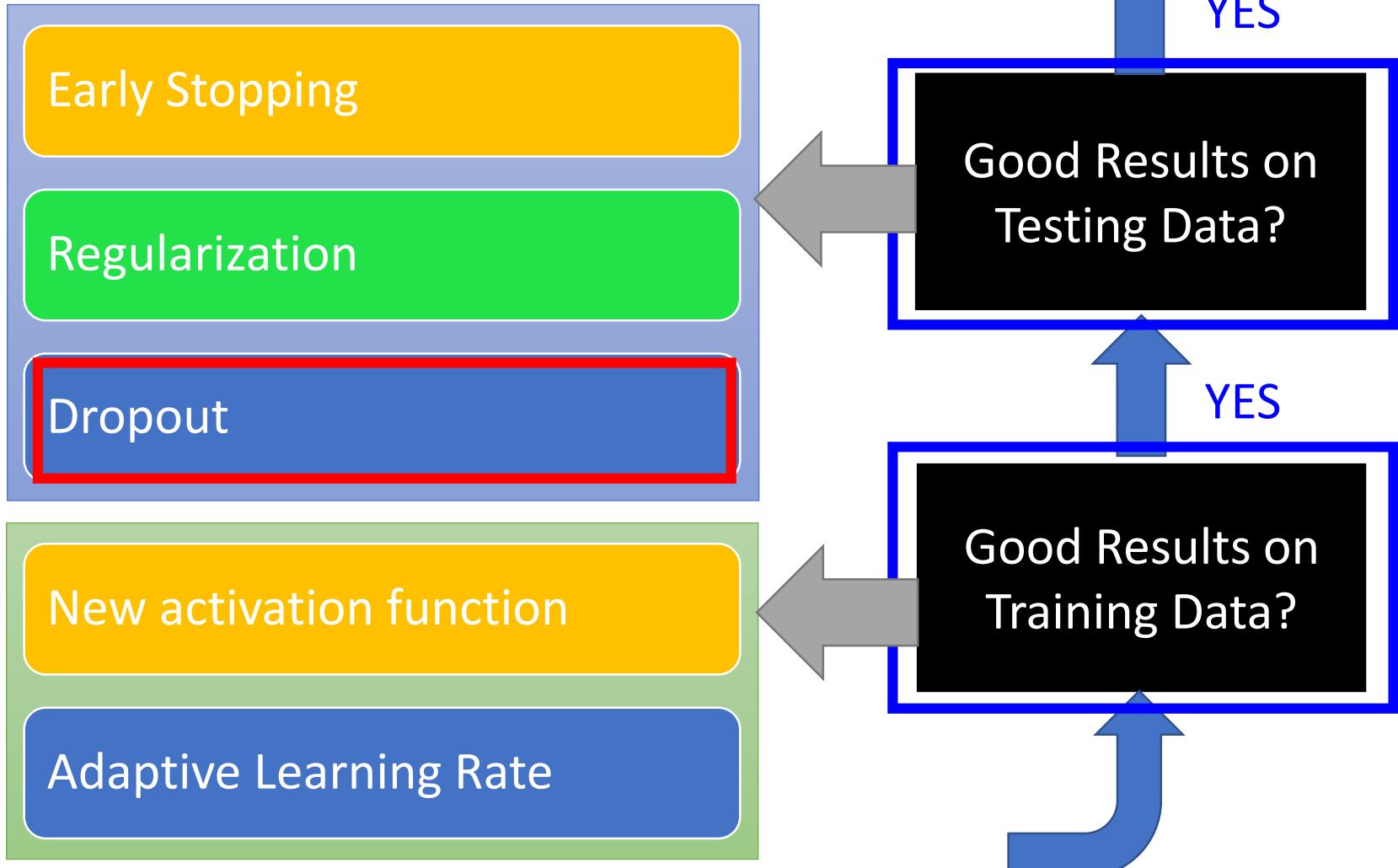


Doing the same thing to machine's brain improves the performance.



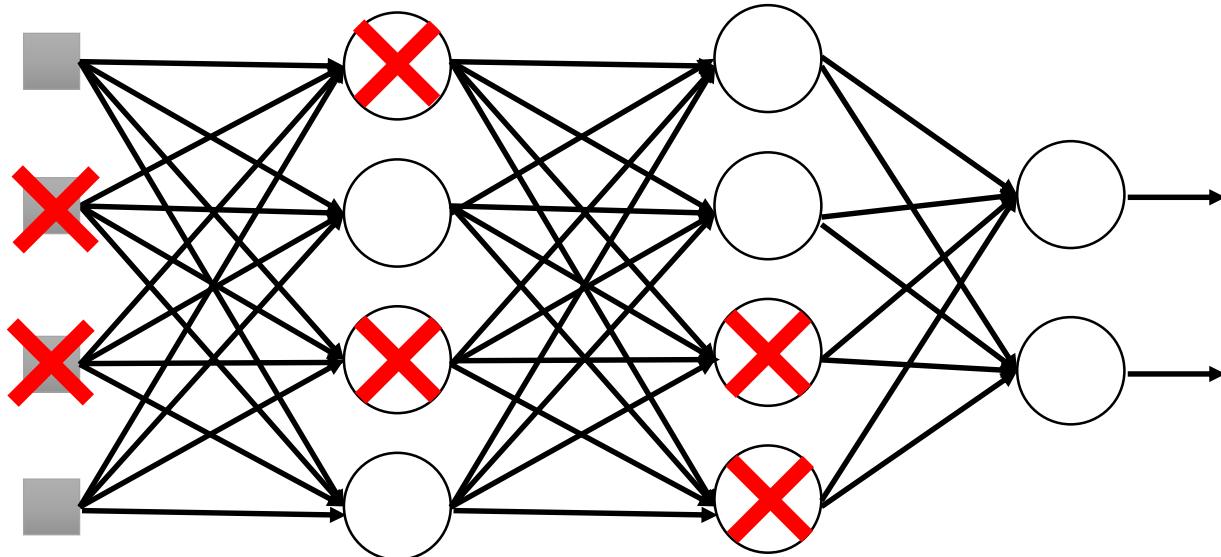
Source: Rethinking the Brain, Families and Work Institute, Rima Shore, 1997; Founders Network slide

Recipe of Deep Learning



Dropout

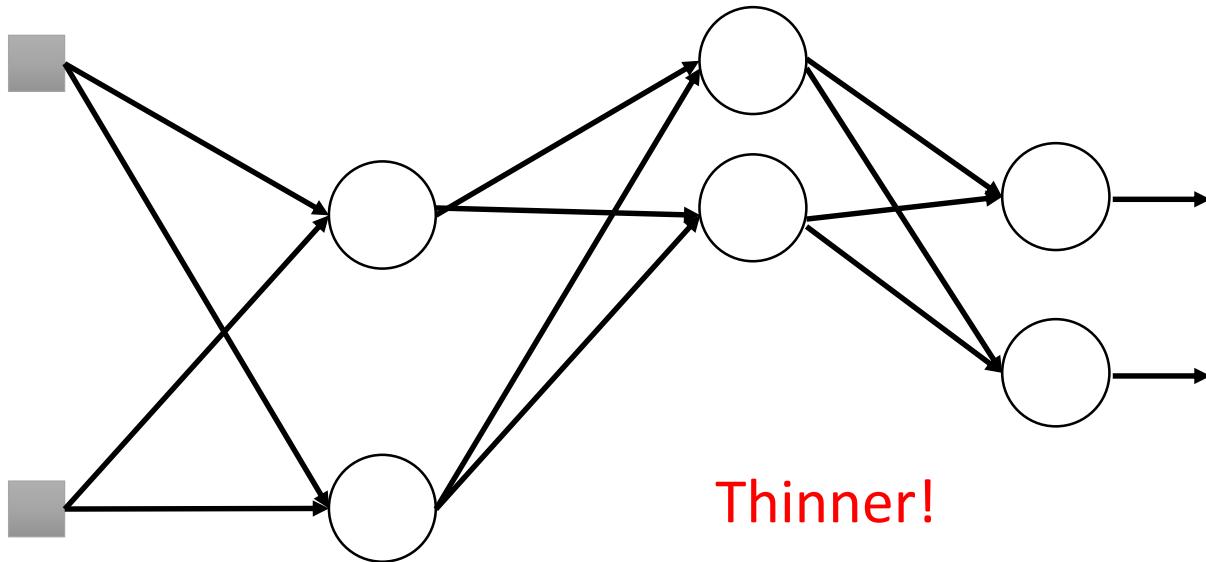
Training:



- **Each time before updating the parameters**
 - Each neuron has $p\%$ to dropout

Dropout

Training:

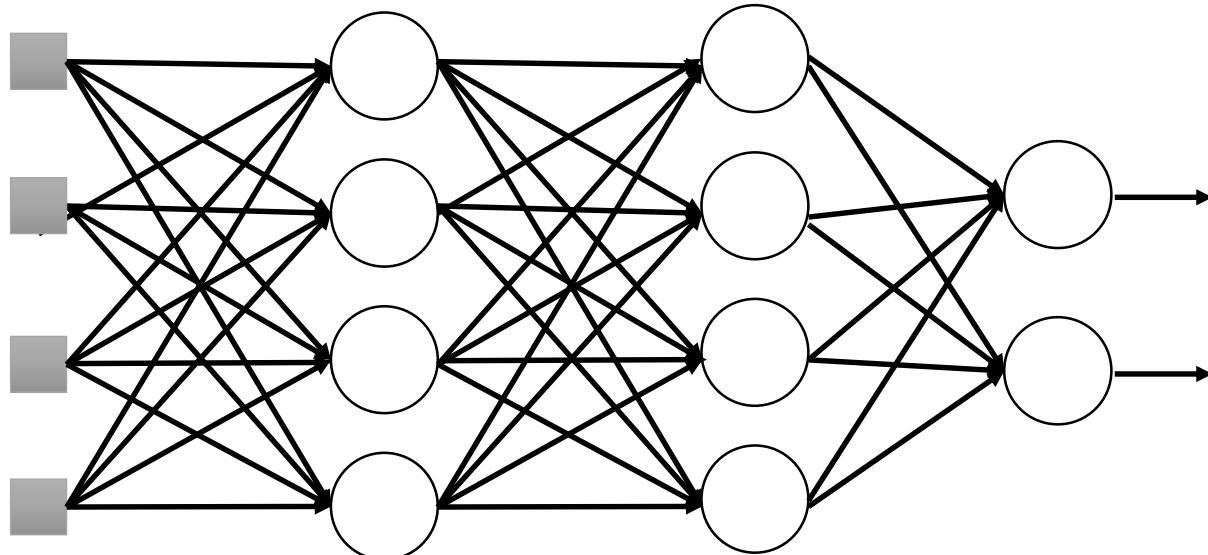


- **Each time before updating the parameters**
 - Each neuron has $p\%$ to dropout
 - ➡ **The structure of the network is changed.**
 - Using the new network for training

For each mini-batch, we resample the dropout neurons

Dropout

Testing:



➤ No dropout

- If the dropout rate at training is $p\%$,
all the weights times $1-p\%$
- Assume that the dropout rate is 50%.
If a weight $w = 1$ by training, set $w = 0.5$ for testing.

Dropout

- Intuitive Reason

Training

Dropout (腳上綁重物)

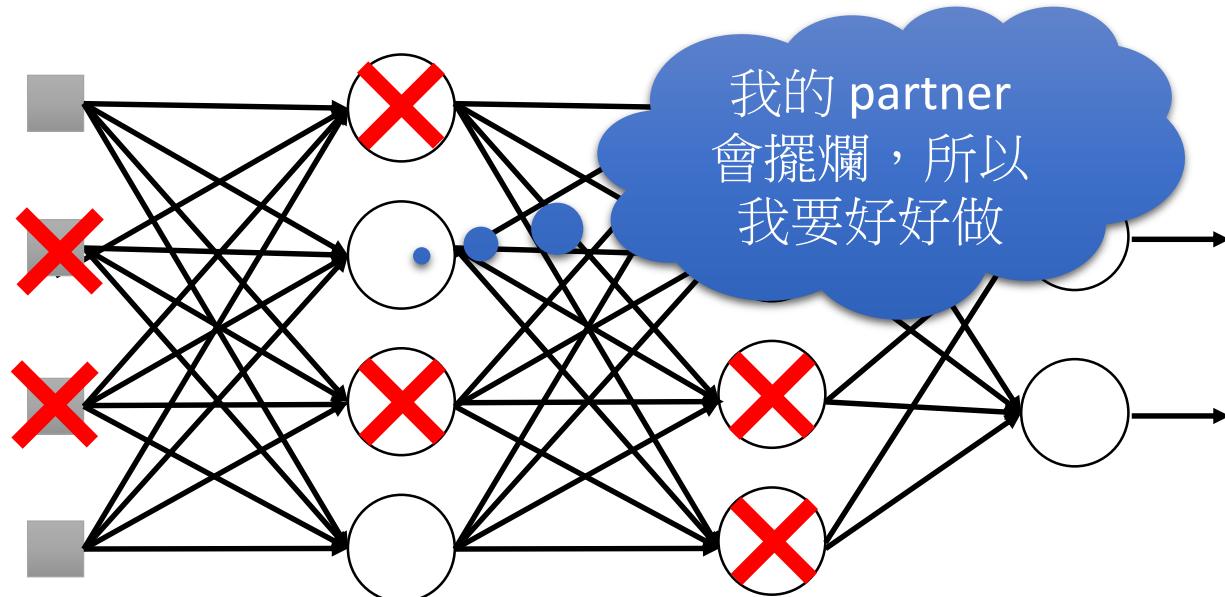


Testing

No dropout
(拿下重物後就變很強)



Dropout - Intuitive Reason



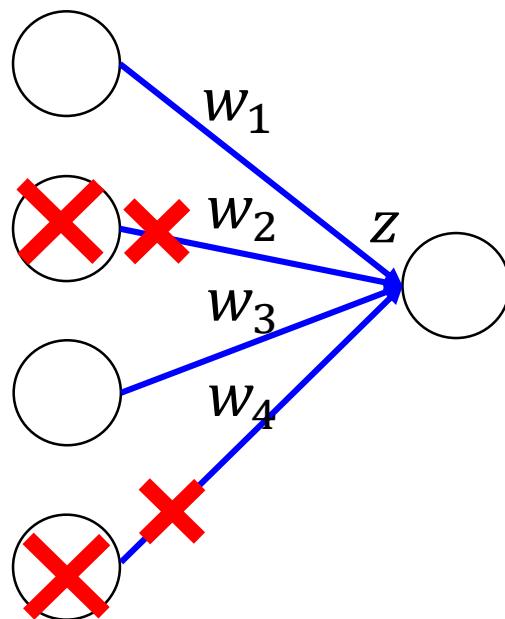
- When teams up, if everyone expect the partner will do the work, nothing will be done finally.
- However, if you know your partner will dropout, you will do better.
- When testing, no one dropout actually, so obtaining good results eventually.

Dropout - Intuitive Reason

- Why the weights should multiply $(1-p)\%$ (dropout rate) when testing?

Training of Dropout

Assume dropout rate is 50%

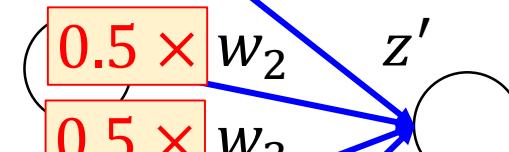


Testing of Dropout

No dropout



Weights from training



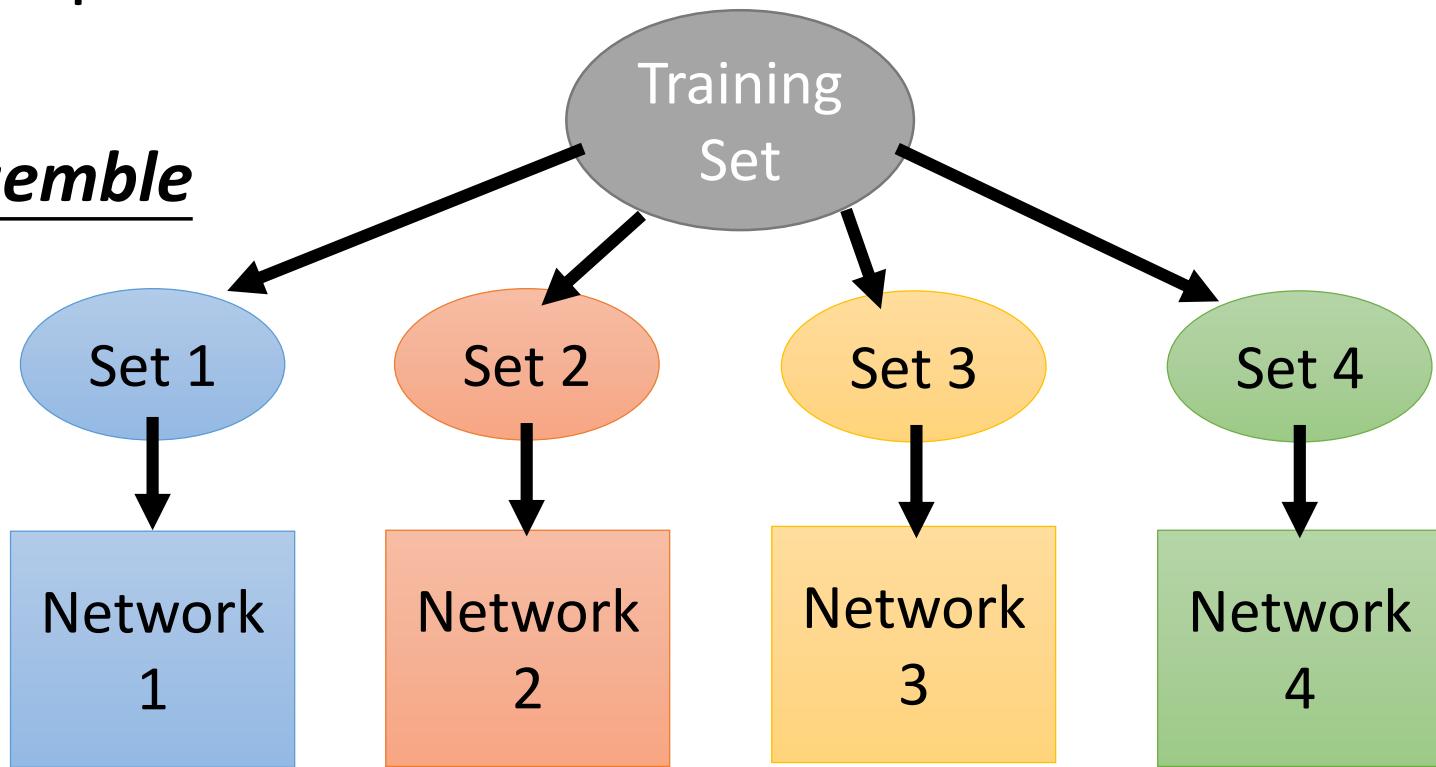
Weights multiply $1-p\%$



$\rightarrow z' \approx z$

Dropout is a kind of ensemble.

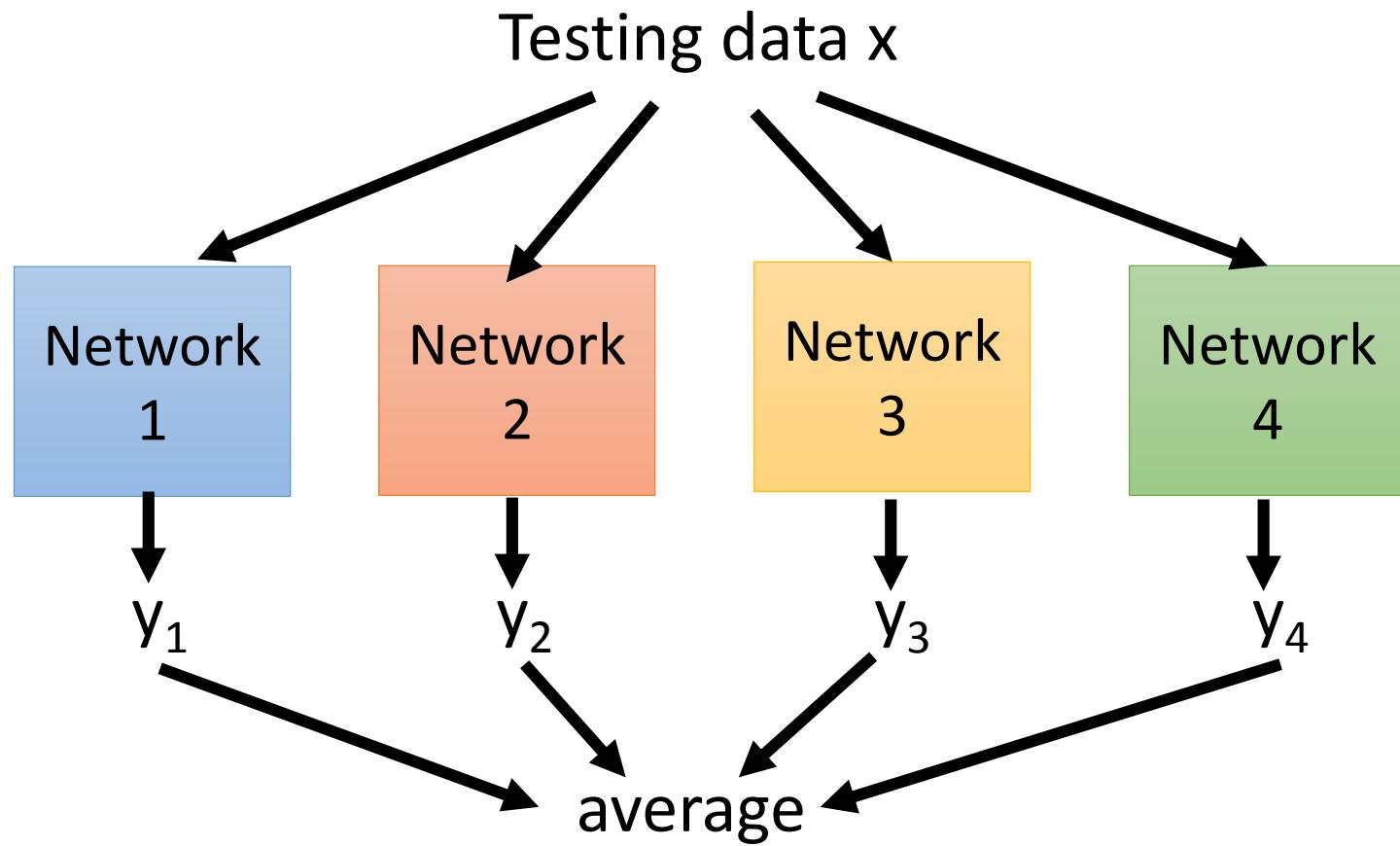
Ensemble



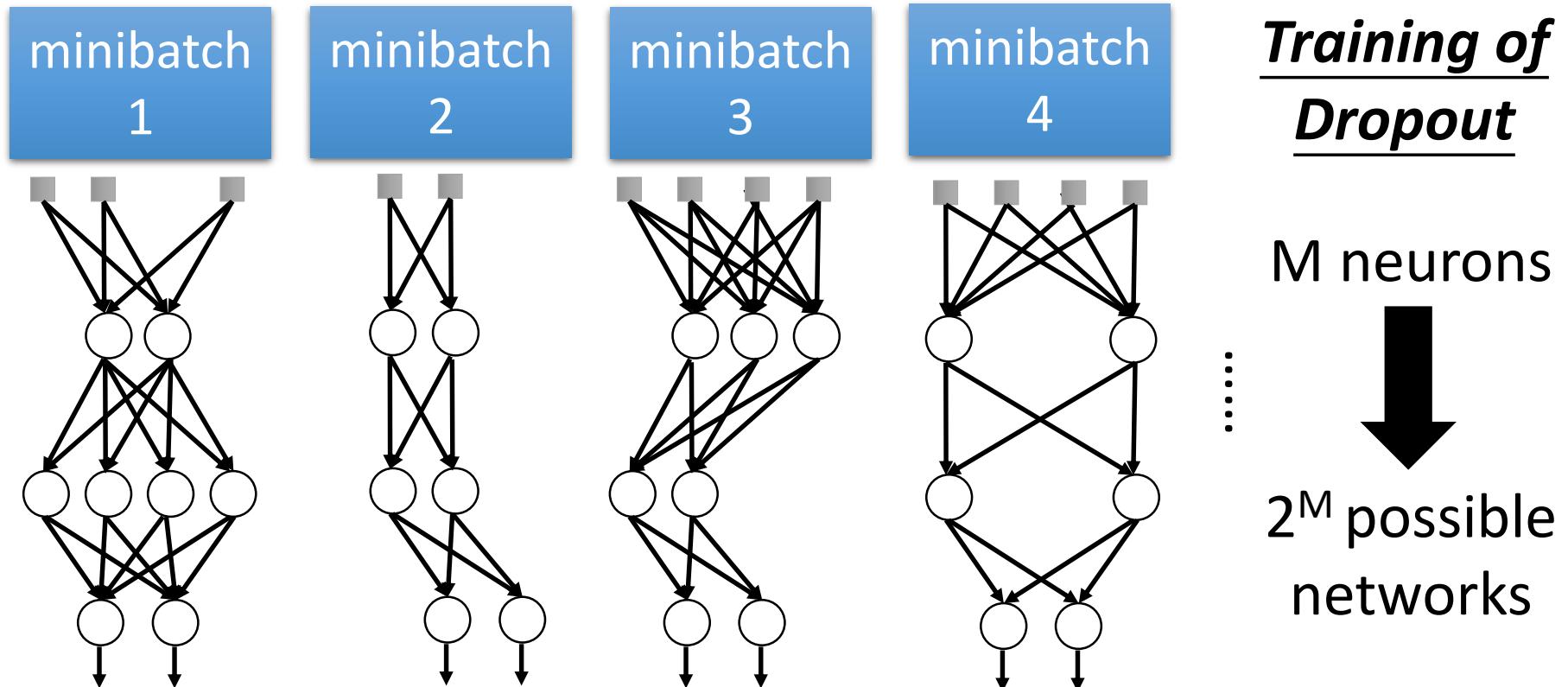
Train a bunch of networks with different structures

Dropout is a kind of ensemble.

Ensemble



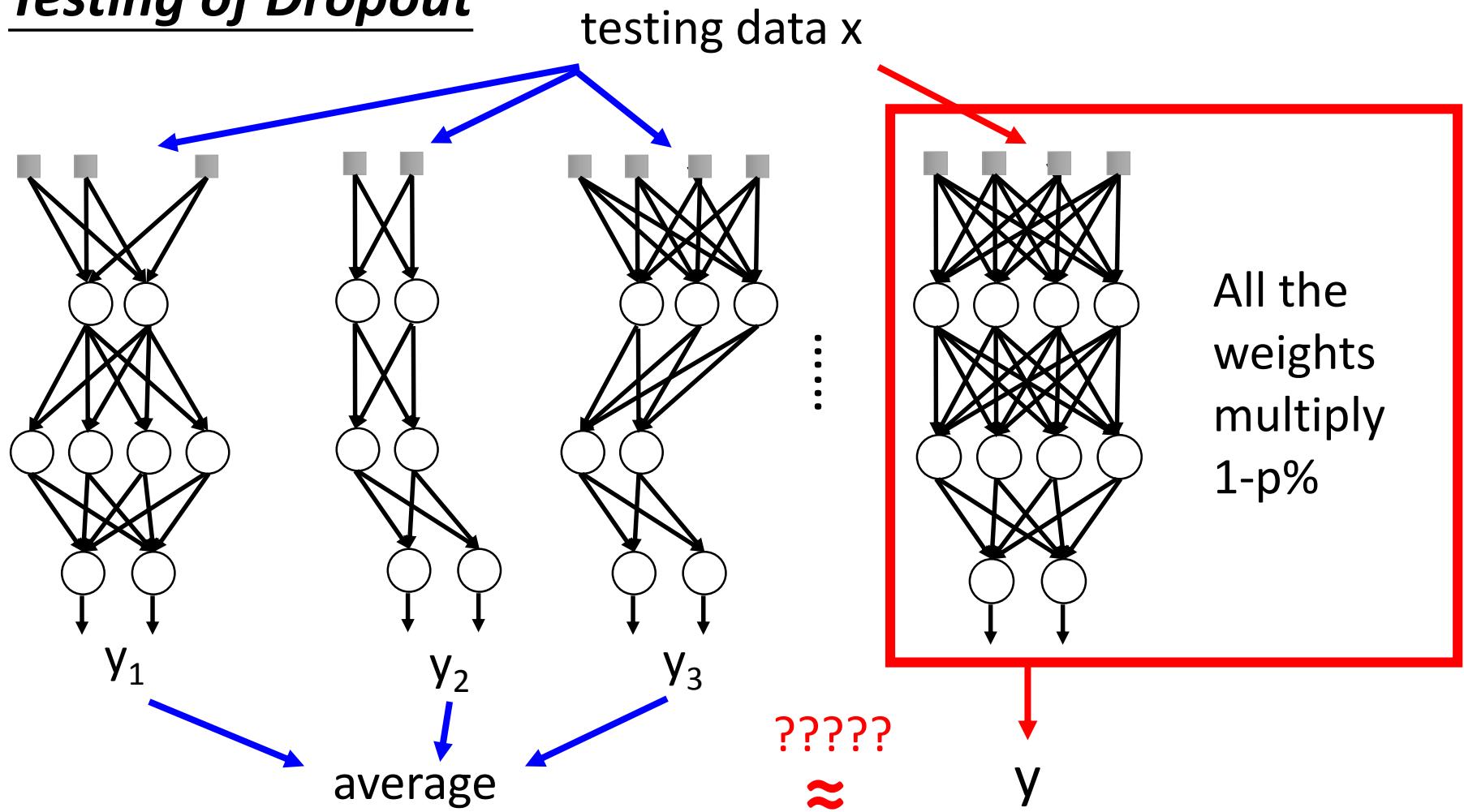
Dropout is a kind of ensemble.



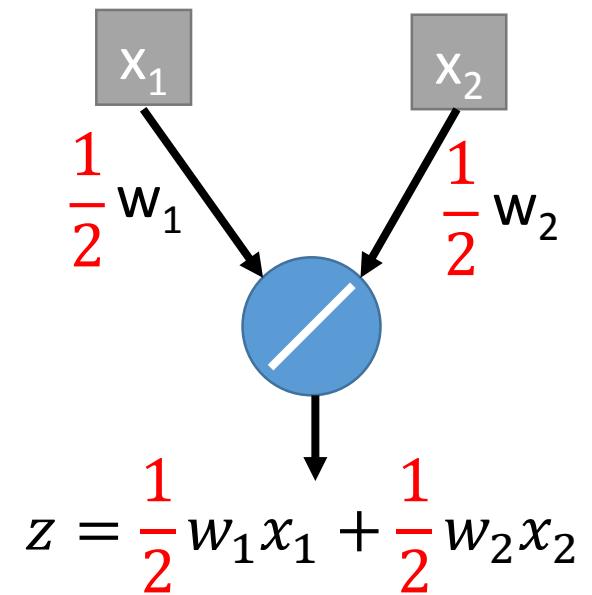
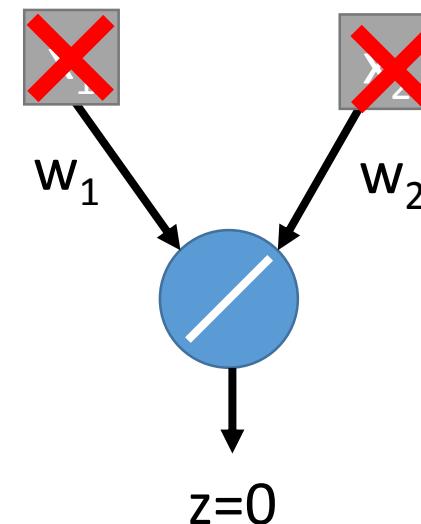
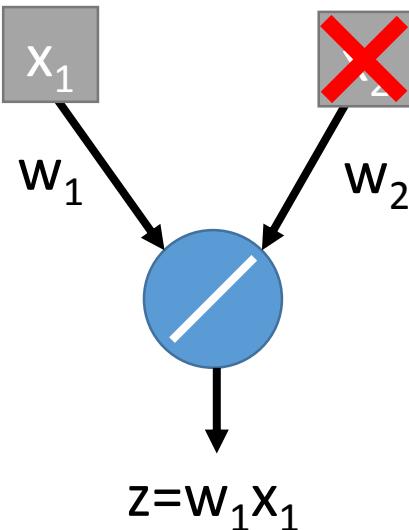
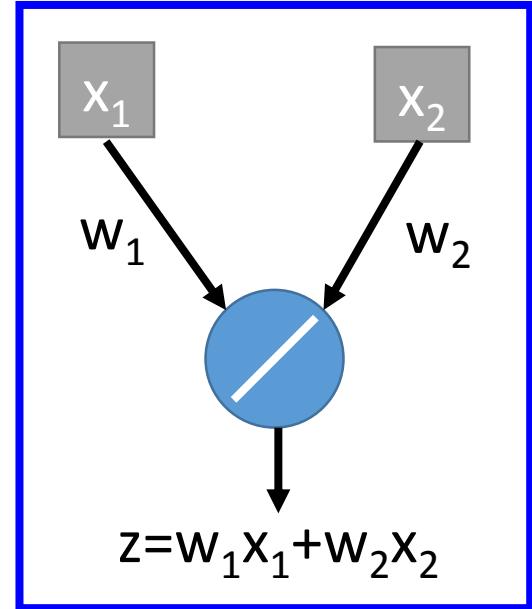
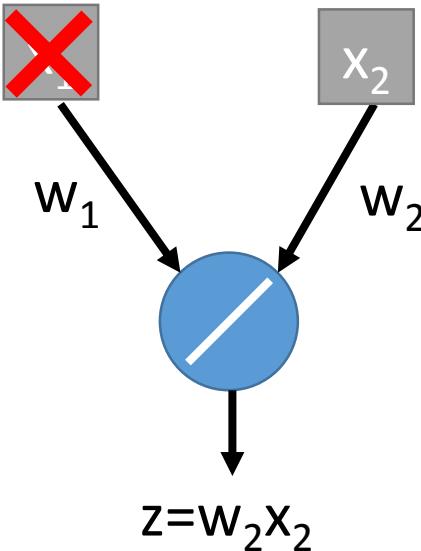
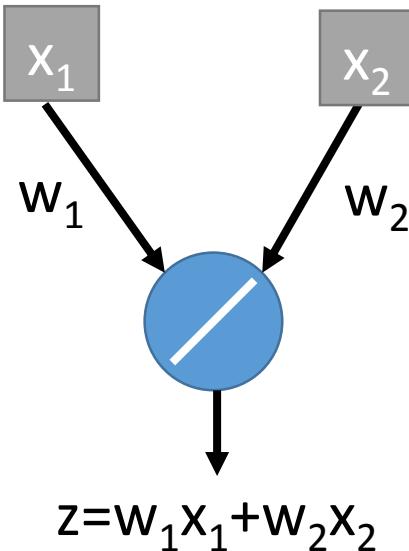
- Using one mini-batch to train one network
- Some parameters in the network are shared

Dropout is a kind of ensemble.

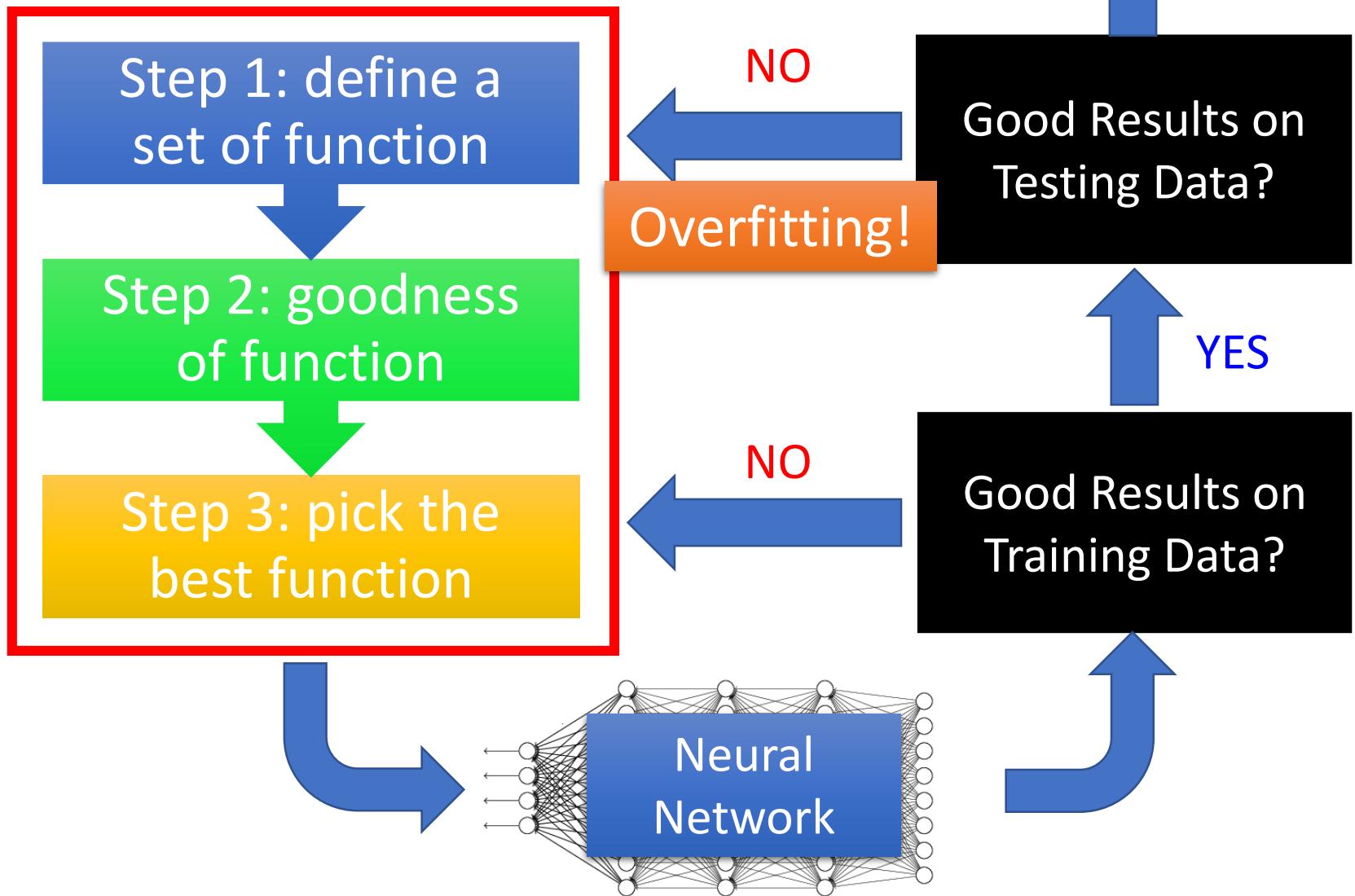
Testing of Dropout



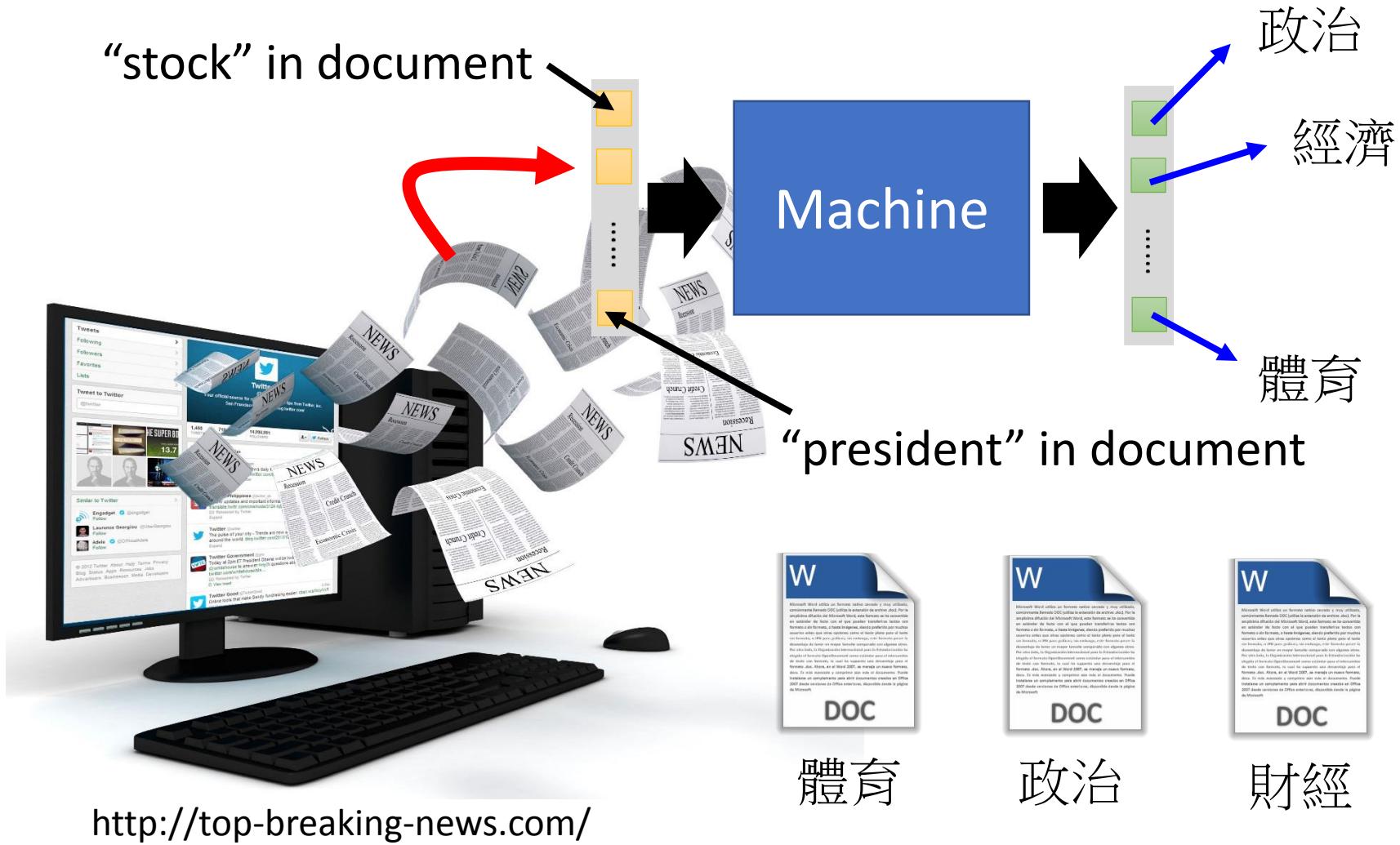
Testing of Dropout



Recipe of Deep Learning



Try another task



Try another task

```
In [8]: x_train.shape  
Out[8]: (8982, 1000)
```

```
In [9]: y_train.shape  
Out[9]: (8982, 46)
```

```
In [12]: x train[0]
```

Out[12]:

```
array([[ 0.,  1.,  1.,  0.,  1.,  1.,  1.,  1.,  1.], [ 0.,  0.,  1.,  1.,  1.,  0.,  1.,  0.,  0.], [ 1.,  0.,  0.,  1.,  1.,  0.,  1.,  0.,  0.], [ 1.,  0.,  0.,  0.,  1.,  1.,  0.,  0.,  0.], [ 1.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.], [ 0.,  0.,  1.,  0.,  0.,  0.,  0.,  0.,  0.], [ 0.,  0.,  0.,  0.,  0.,  1.,  1.,  0.,  0.], [ 0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.], [ 0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.], [ 0.,  0.,  0.,  0.,  1.,  0.,  1.,  0.,  0.], [ 0.,  0.,  0.,  0.,  0.,  0.,  1.,  0.,  0.], [ 0.,  0.,  1.,  0.,  1.,  0.,  0.,  0.,  0.], [ 0.,  0.,  0.,  0.,  0.,  0.,  0.,  1.,  1.], [ 0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.], [ 0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.], [ 0.,  0.,  0.,  0.,  1.,  0.,  0.,  0.,  0.], [ 0.,  0.,  1.,  0.,  0.,  0.,  0.,  0.,  0.], [ 0.,  1.,  0.,  0.,  0.,  0.,  0.,  0.,  0.], [ 0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.]])
```

```
In [10]: x_test.shape
```

```
In [11]: y_test.shape  
Out[11]: (2246, 46)
```

```
In [13]: y_train[0]
```

Out[13]:

```
array([ 0.,  0.,  0.,  1.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  
       0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  
       0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  
       0.,  0.,  0.,  0.,  0.,  0.,  0.])
```

Live Demo

Convolutional Neural Network

Hung-yi Lee

Can the network be simplified by
considering the properties of images?

Why CNN for Image

- Some patterns are much smaller than the whole image

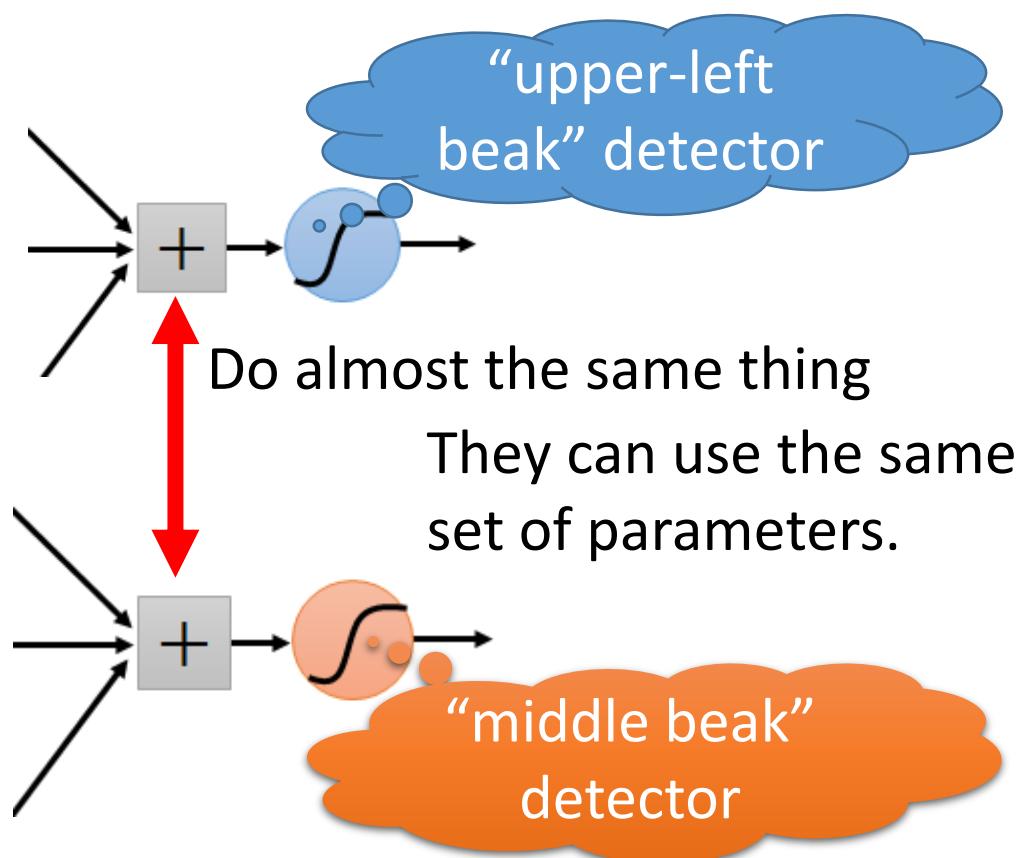
A neuron does not have to see the whole image to discover the pattern.

Connecting to small region with less parameters



Why CNN for Image

- The same patterns appear in different regions.



Why CNN for Image

- Subsampling the pixels will not change the object

bird



subsampling

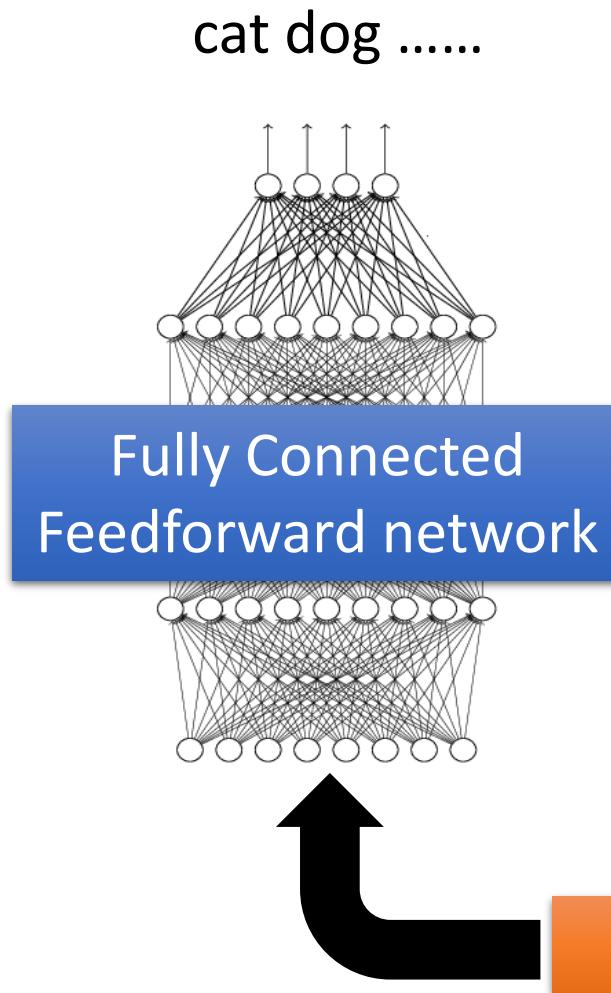
bird



We can subsample the pixels to make image smaller

→ Less parameters for the network to process the image

The whole CNN



Can repeat
many times

The whole CNN

Property 1

- Some patterns are much smaller than the whole image

Property 2

- The same patterns appear in different regions.

Property 3

- Subsampling the pixels will not change the object



Convolution



Max Pooling



Convolution

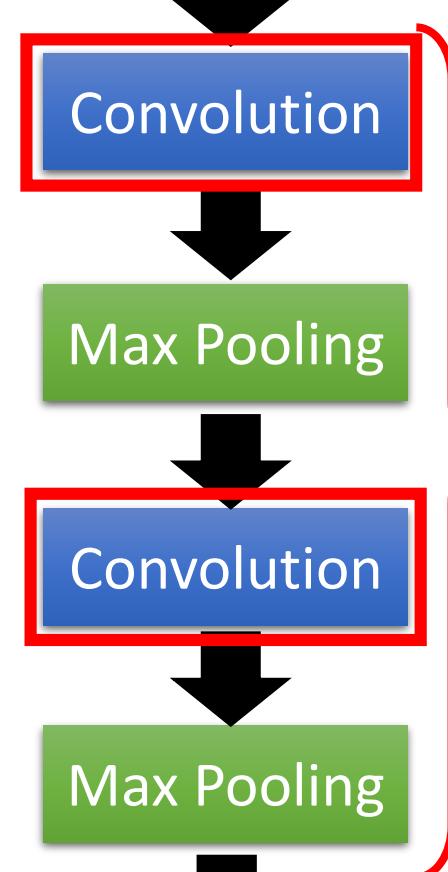
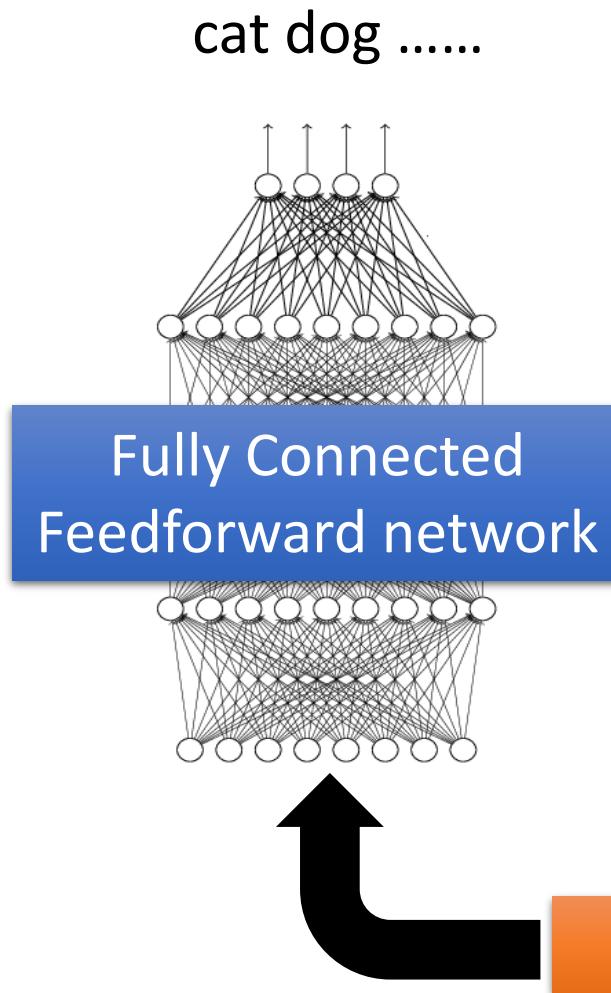


Max Pooling



Can repeat
many times

The whole CNN



Flatten

Can repeat
many times

CNN – Convolution

Those are the network parameters to be learned.

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1
Matrix

-1	1	-1
-1	1	-1
-1	1	-1

Filter 2
Matrix

⋮

Property 1

Each filter detects a small pattern (3 x 3).

CNN – Convolution

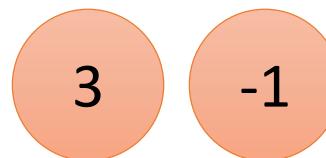
stride=1

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1



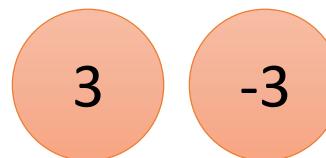
CNN – Convolution

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

If stride=2

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

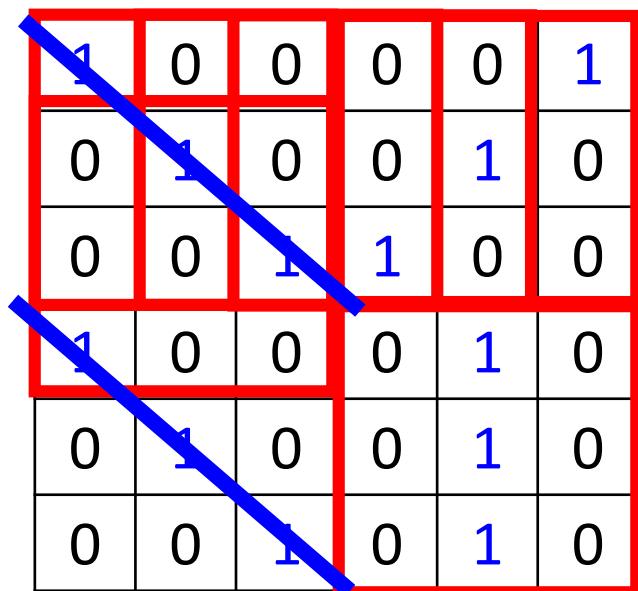


We set stride=1 below

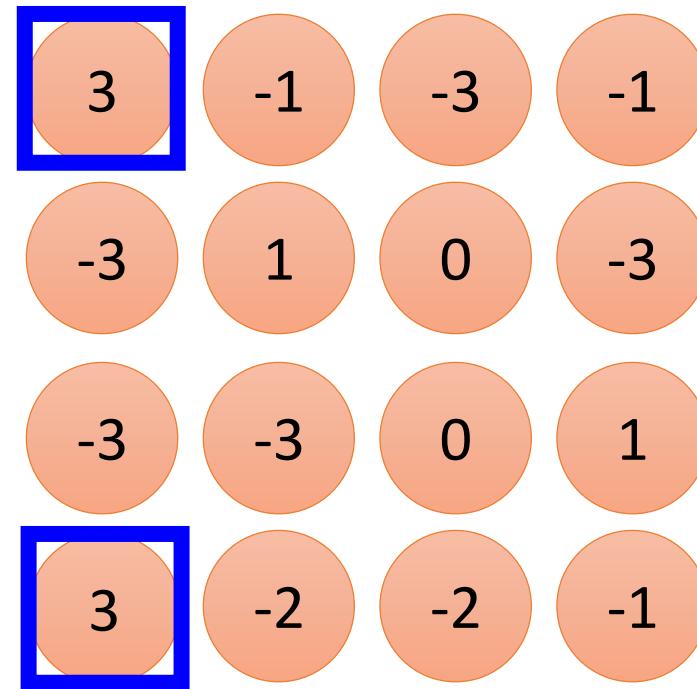
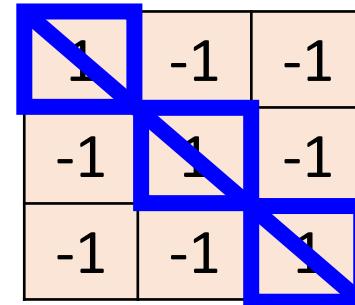
6 x 6 image

CNN – Convolution

stride=1



6 x 6 image



CNN – Convolution

stride=1

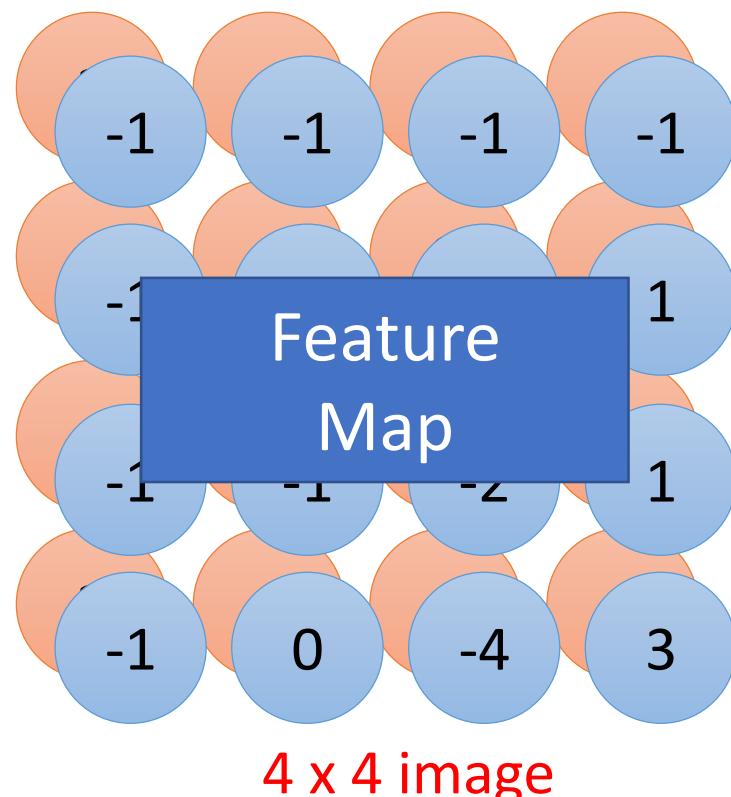
1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image

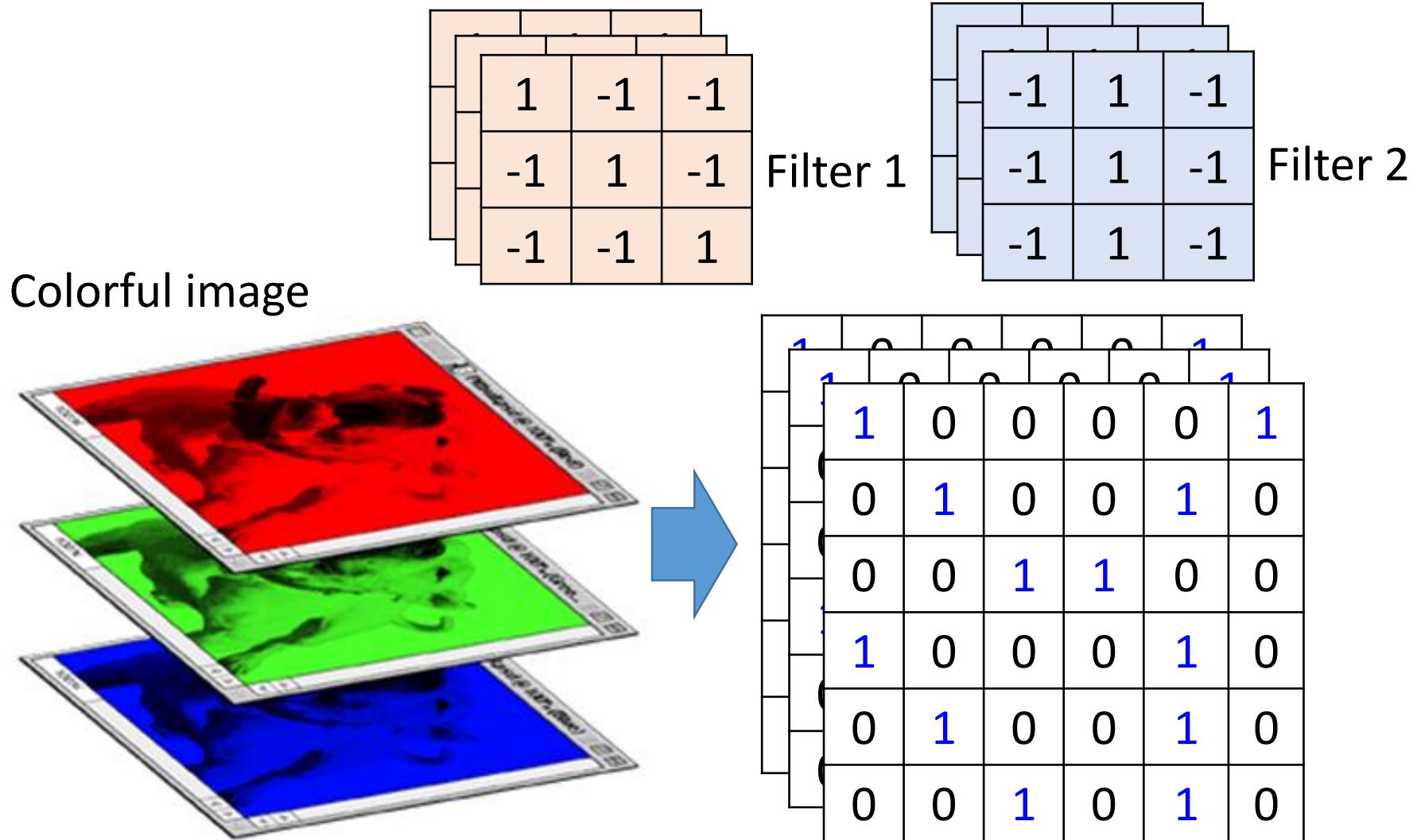
-1	1	-1
-1	1	-1
-1	1	-1

Filter 2

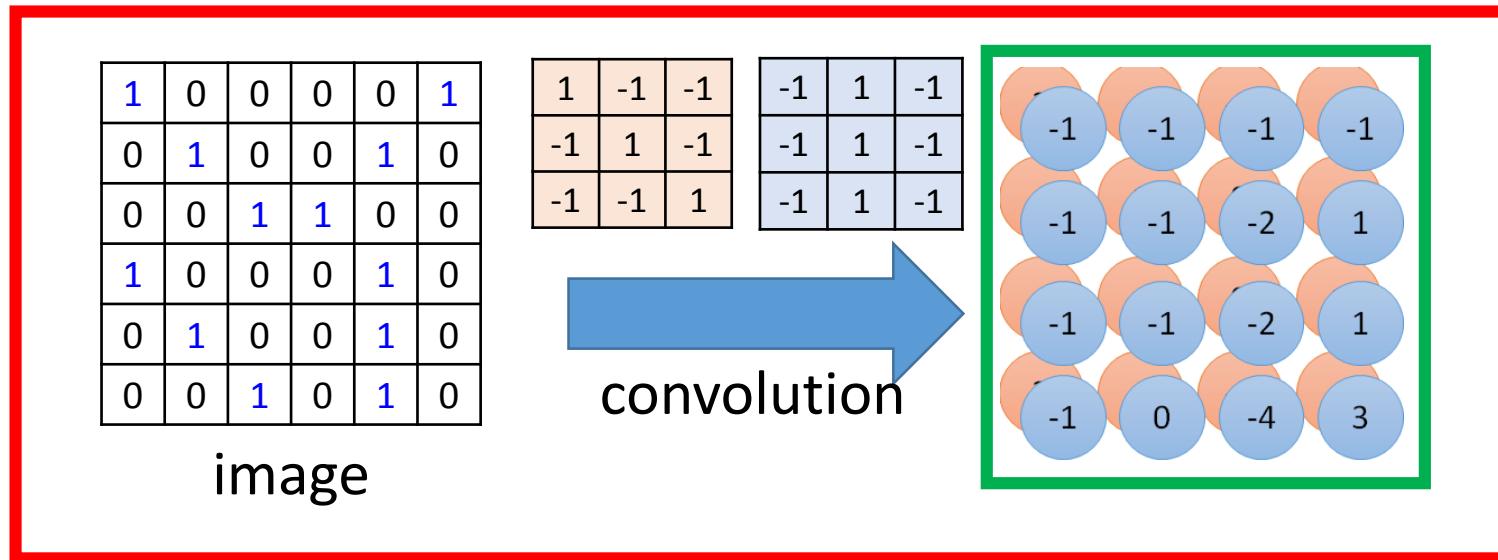
Do the same process for every filter



CNN – Colorful image

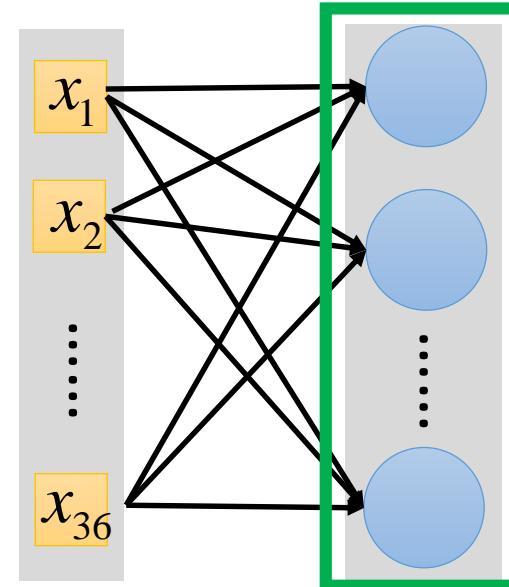


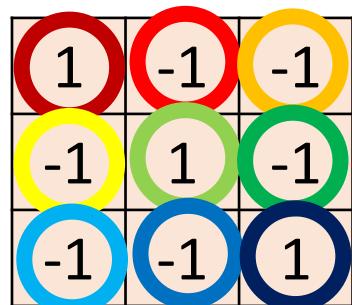
Convolution v.s. Fully Connected



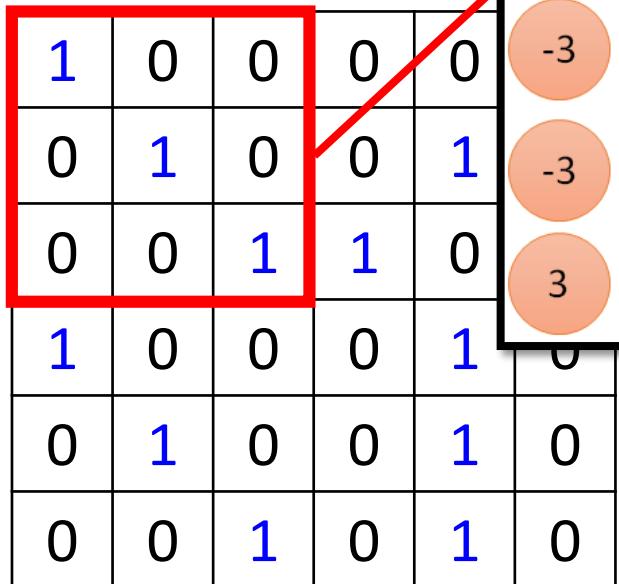
Fully-
connected

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0



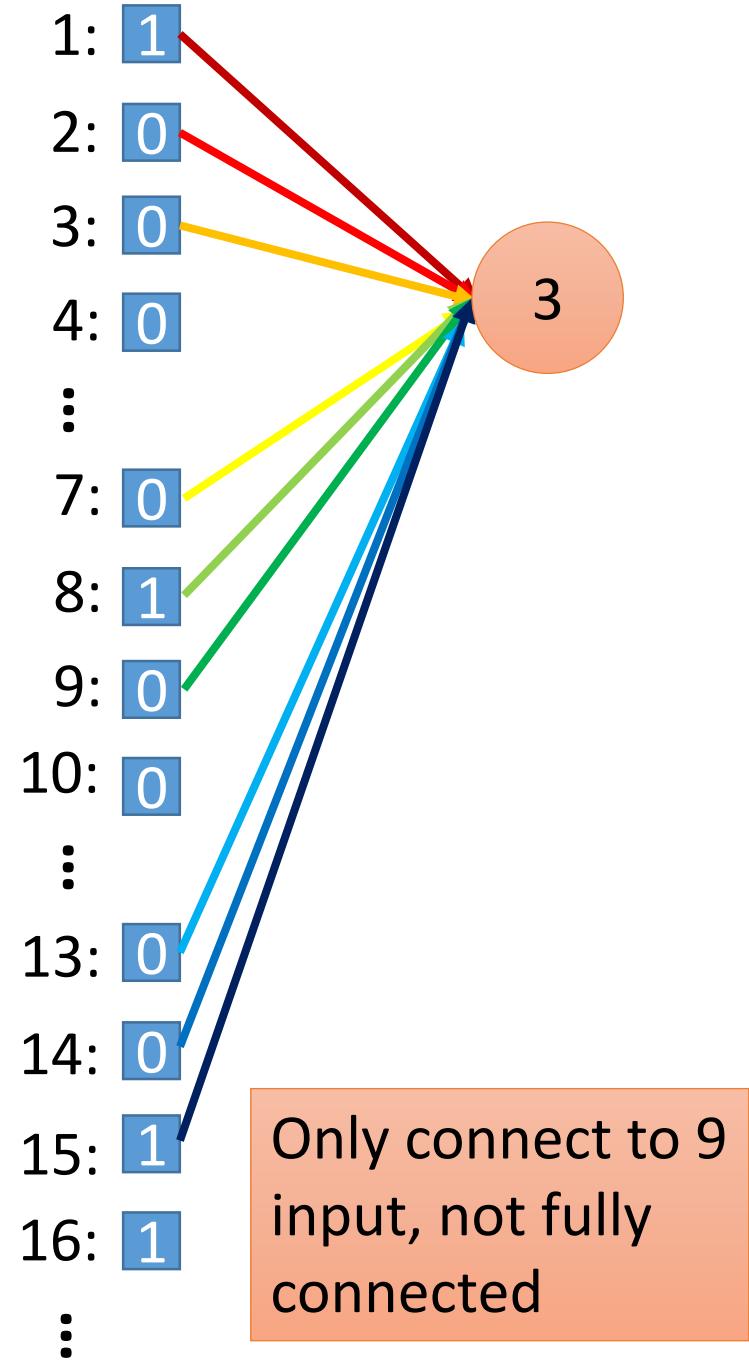
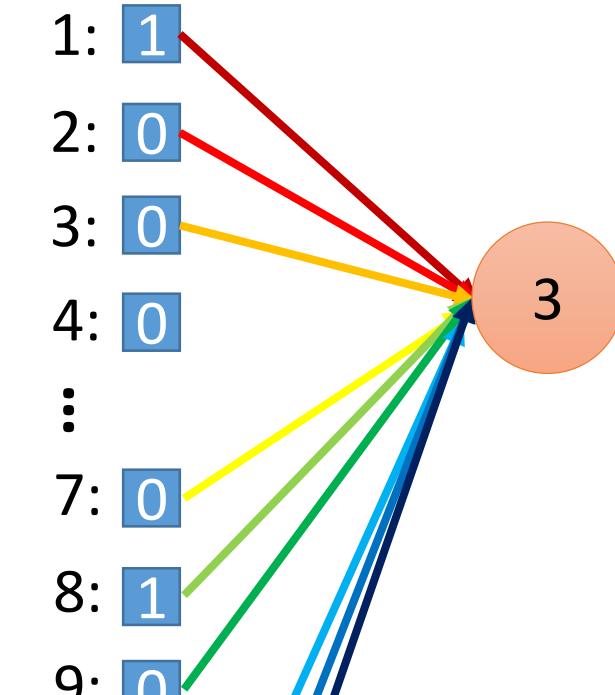
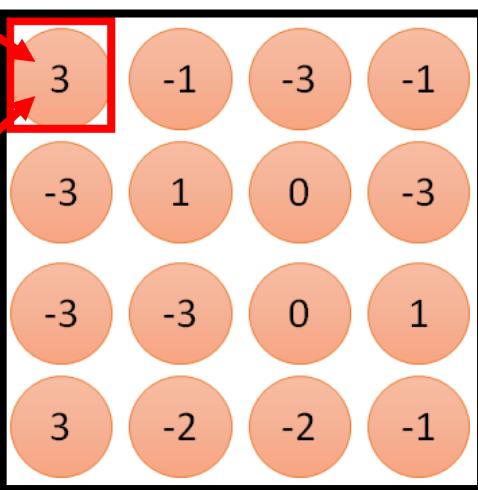


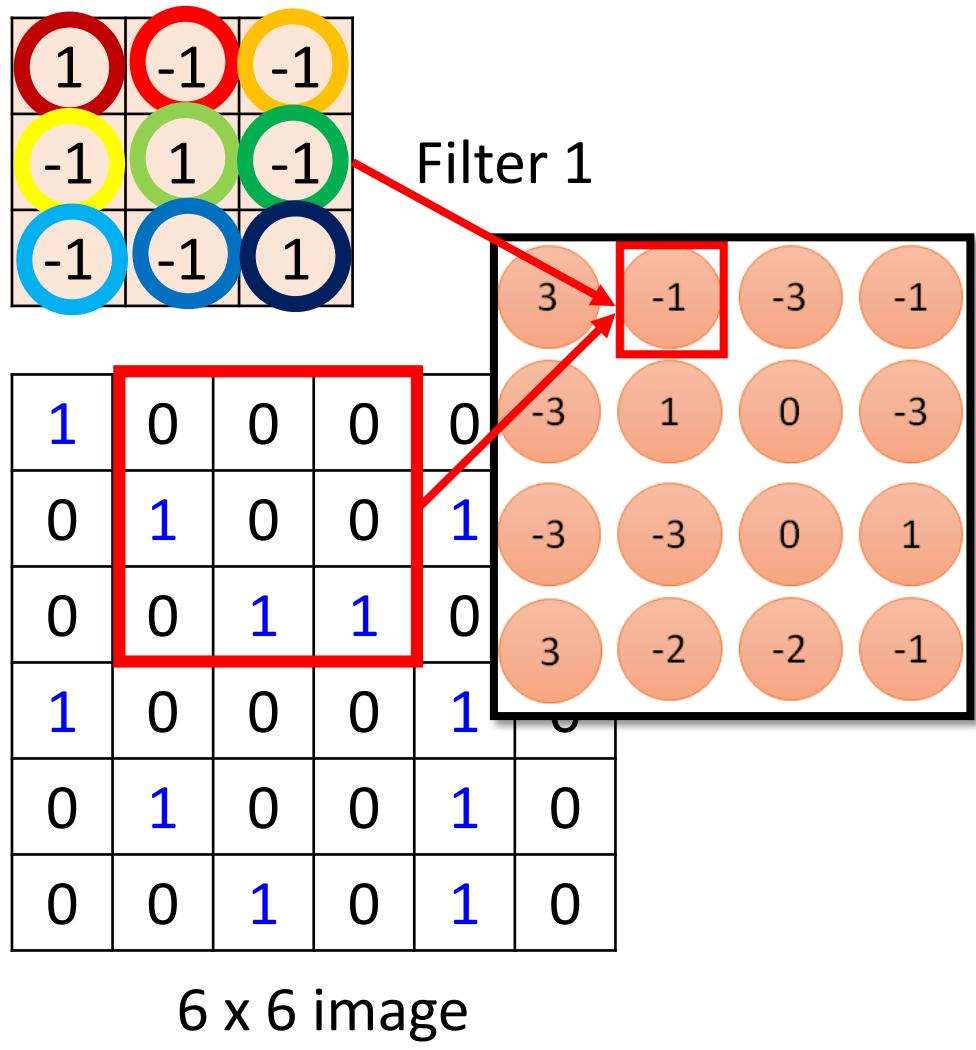
Filter 1



6 x 6 image

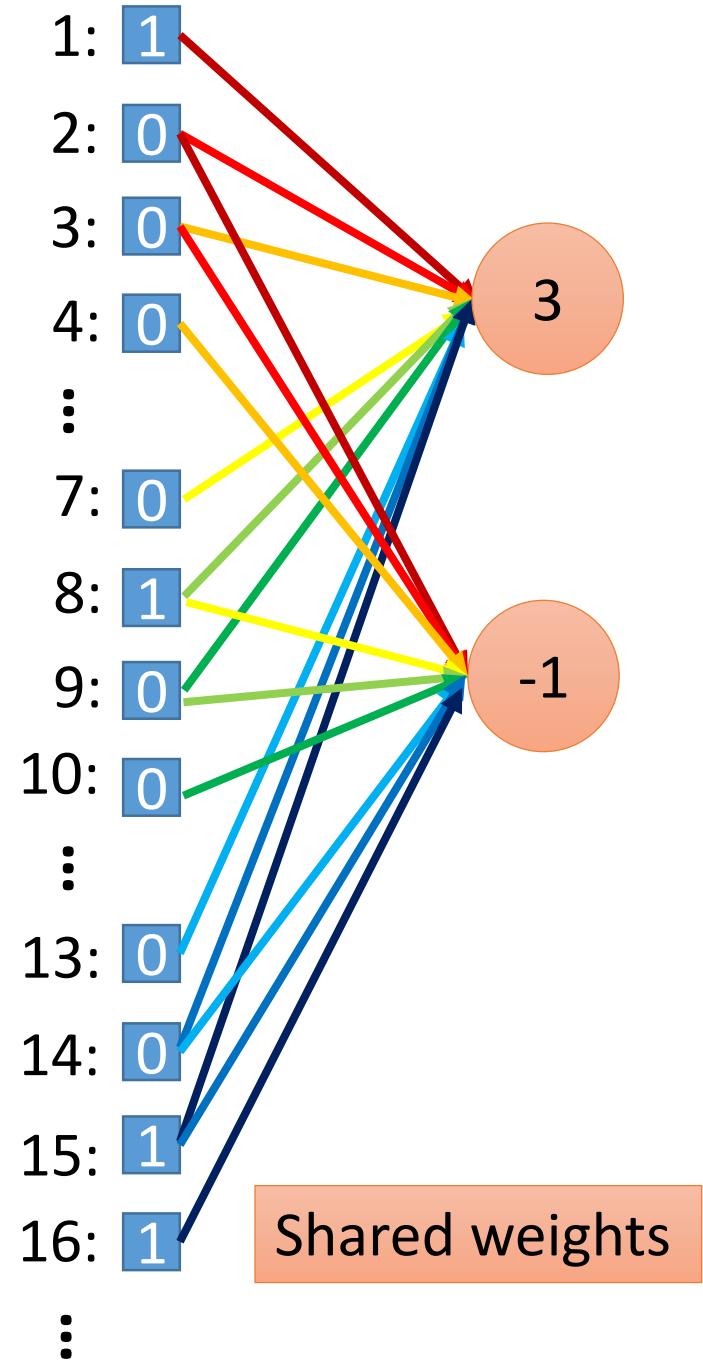
Less parameters!



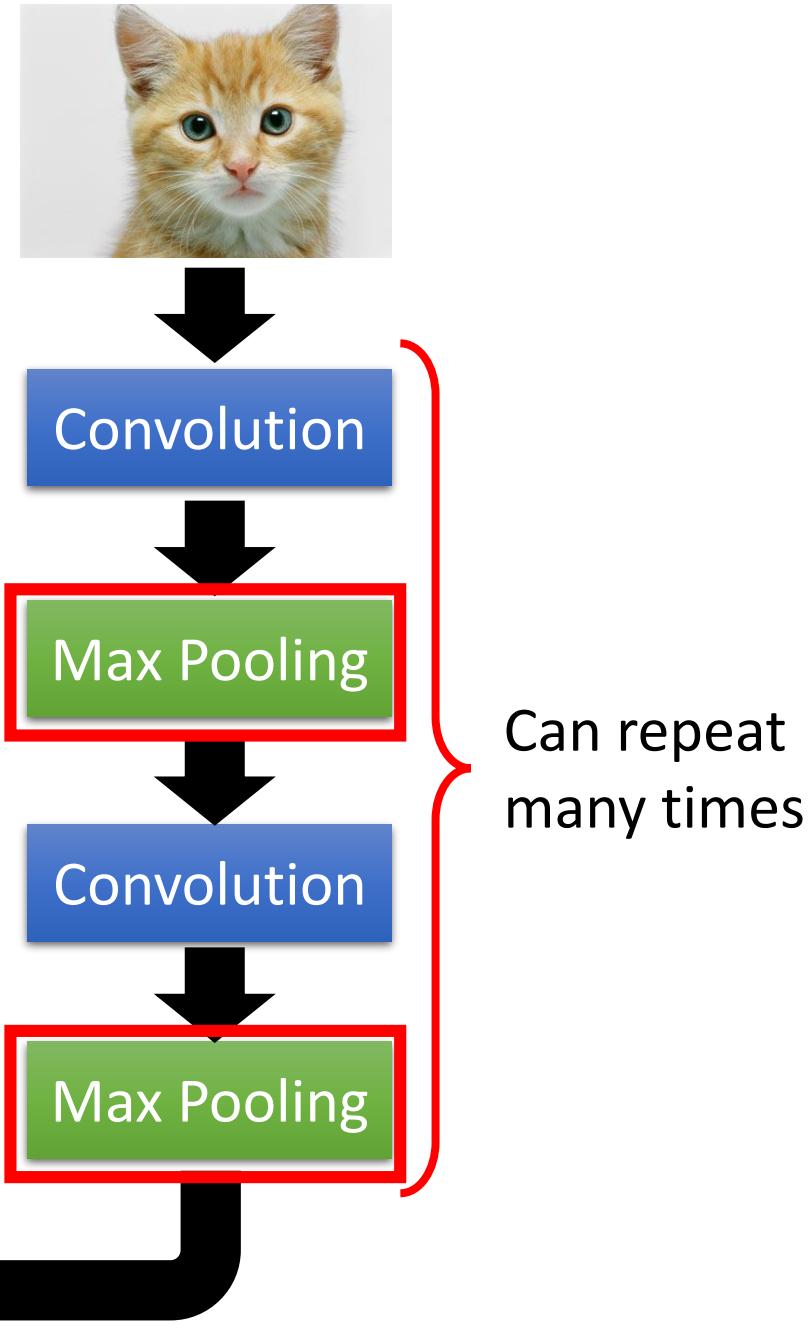
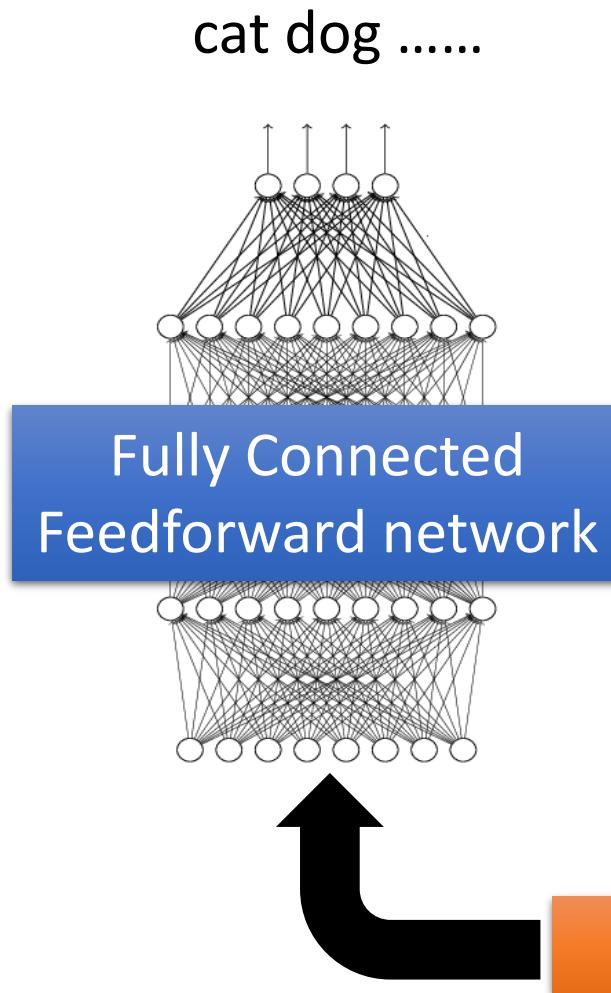


Less parameters!

Even less parameters!



The whole CNN



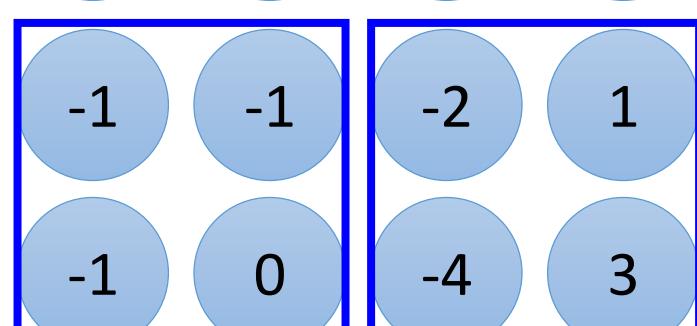
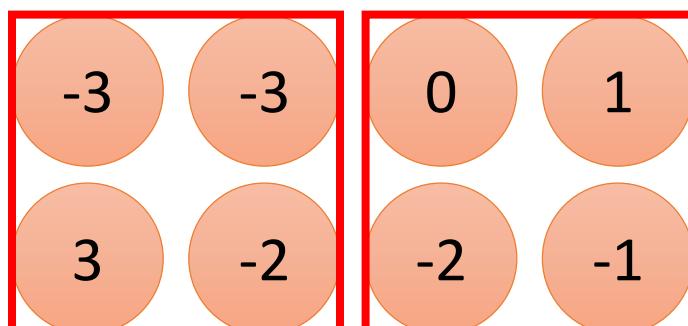
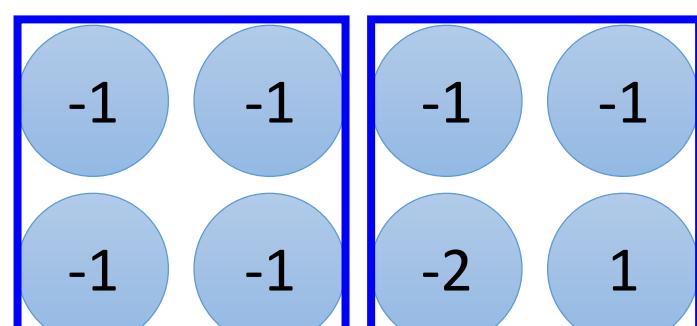
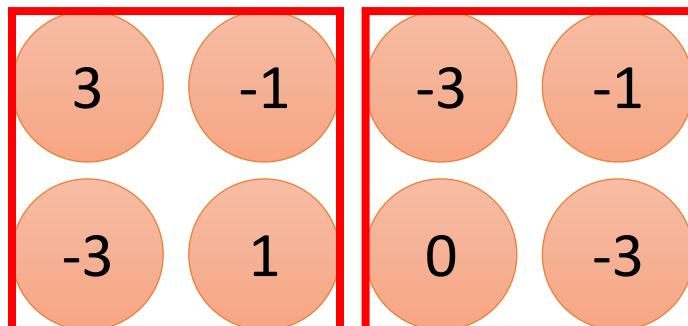
CNN – Max Pooling

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

-1	1	-1
-1	1	-1
-1	1	-1

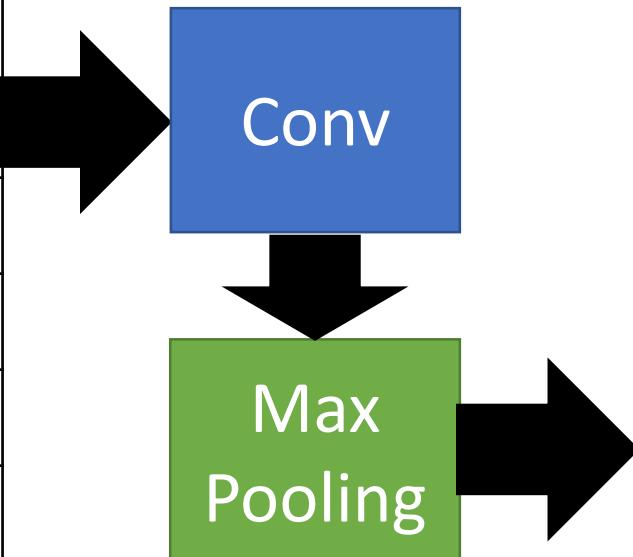
Filter 2



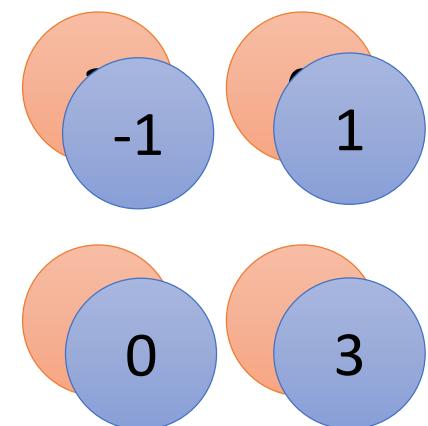
CNN – Max Pooling

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image



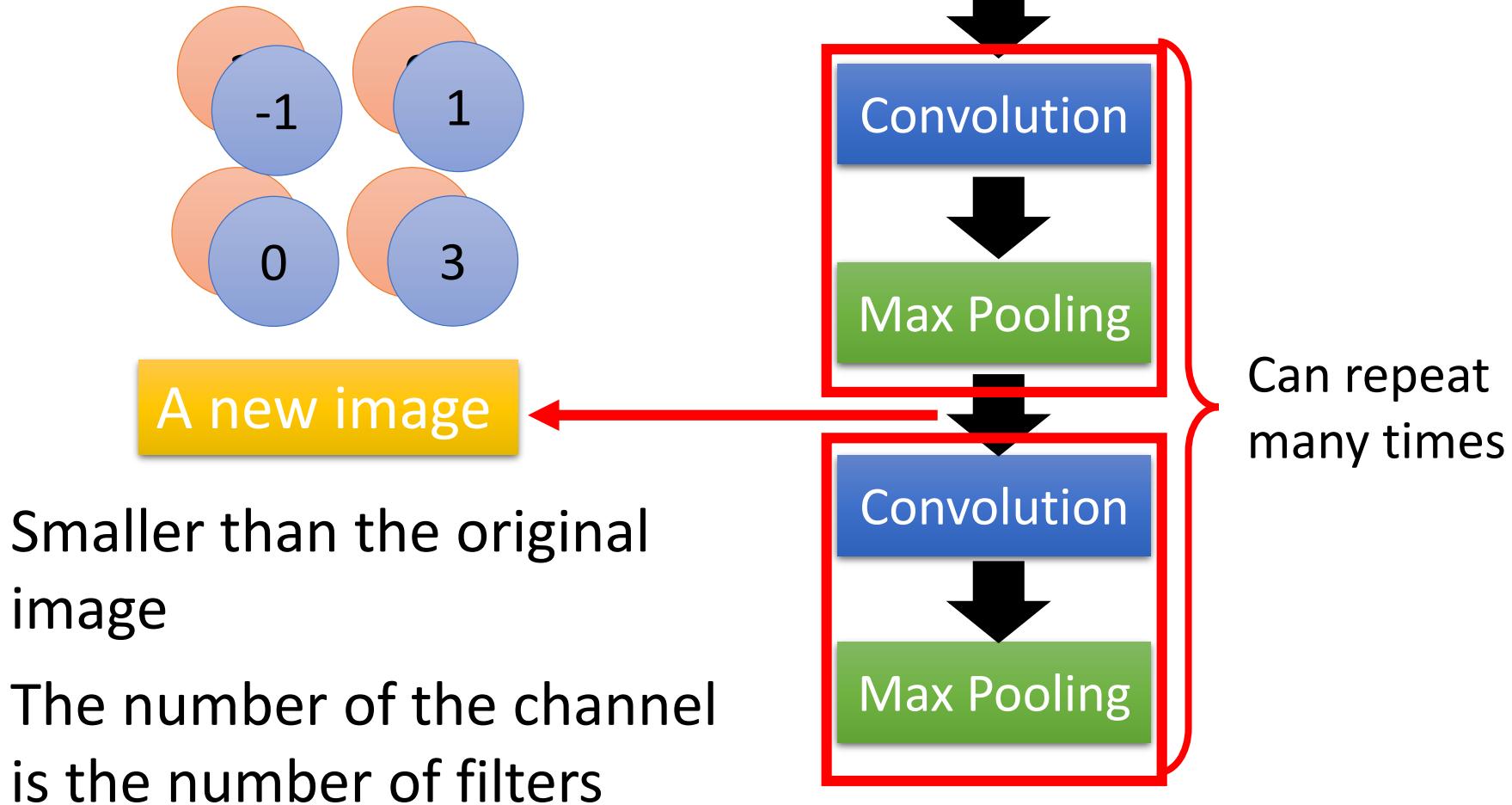
New image
but smaller



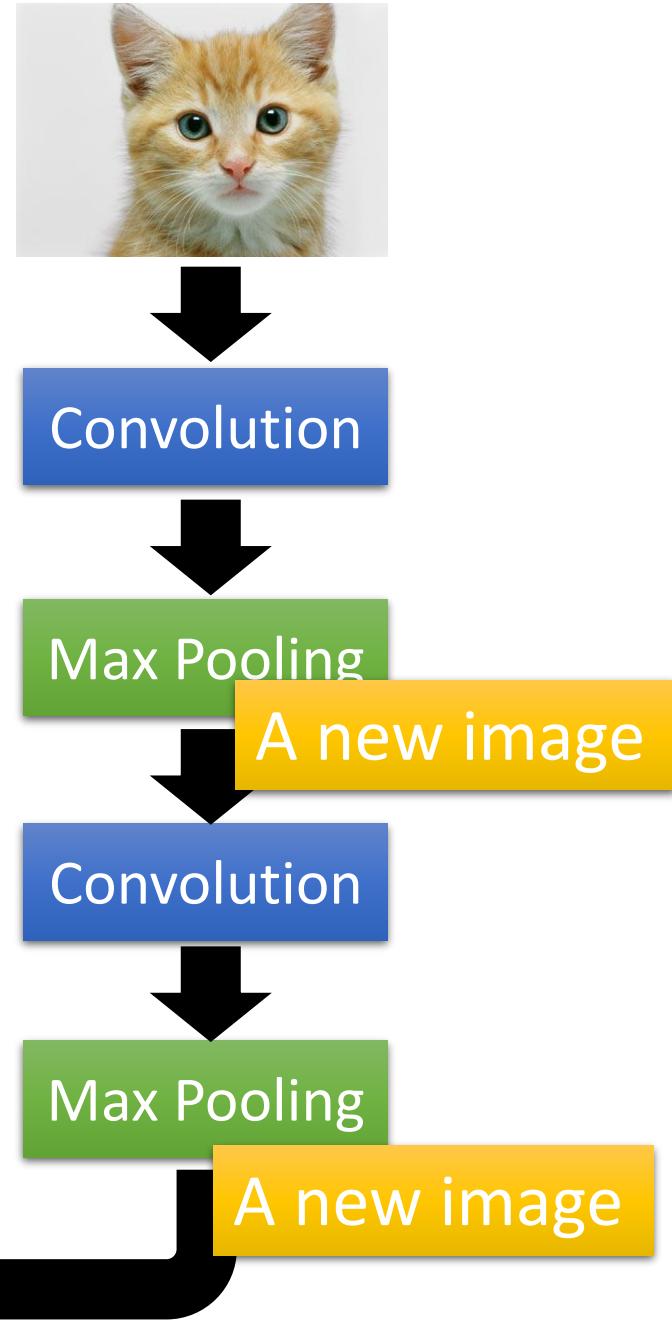
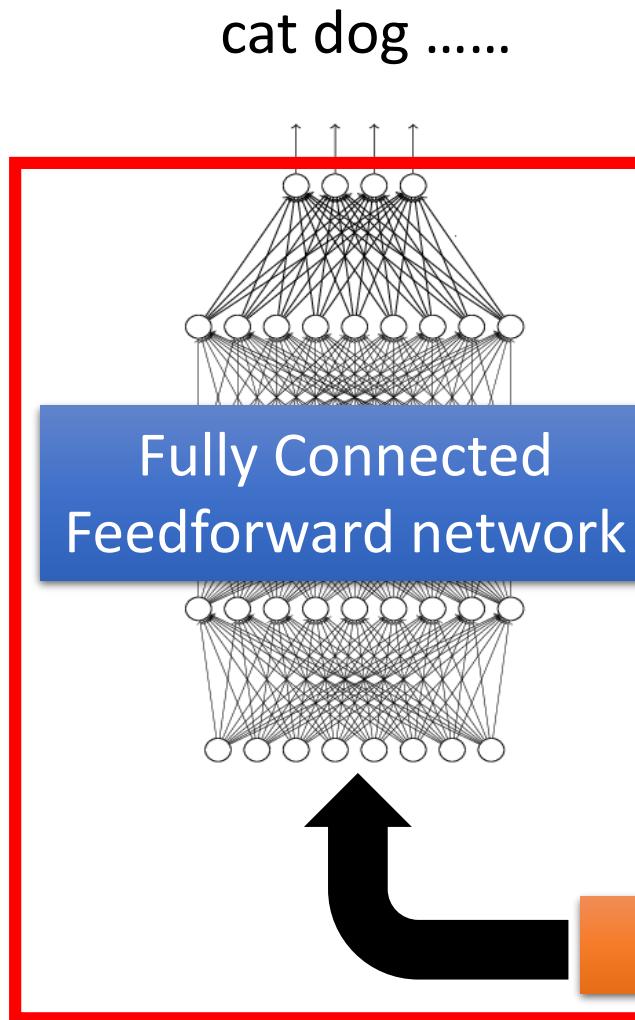
2 x 2 image

Each filter
is a channel

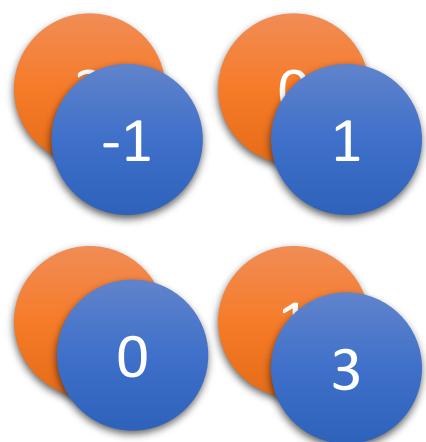
The whole CNN



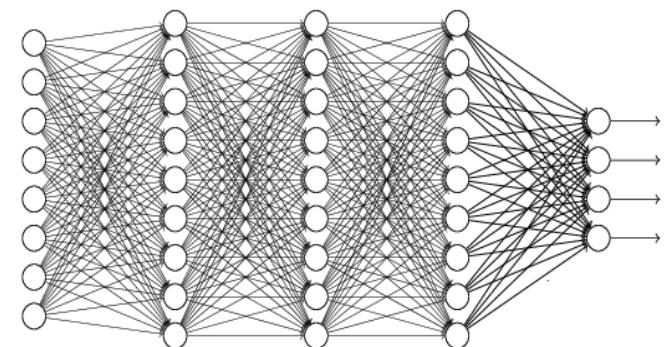
The whole CNN



Flatten



Flatten

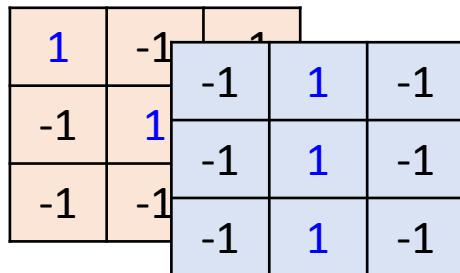


Fully Connected
Feedforward network

CNN in Keras

Only modified the *network structure* and *input format (vector -> 3-D tensor)*

```
model2.add( Convolution2D( 25, 3, 3,  
                           input_shape=(28, 28, 1)) )
```

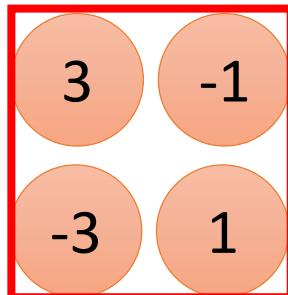


There are 25
3x3 filters.

Input_shape = (28, 28, 1)

28 x 28 pixels 1: black/white, 3: RGB

```
model2 .add (MaxPooling2D ( (2,2) ))
```



input
↓

Convolution



Max Pooling



Convolution



Max Pooling

CNN in Keras

Only modified the *network structure* and *input format (vector -> 3-D tensor)*

How many parameters
for each filter?

9

25 x 26 x 26

```
model2.add(Convolution2D( 25, 3, 3,  
    input_shape=(28,28,1)) )
```

How many parameters
for each filter?

225

50 x 11 x 11

```
model2.add(MaxPooling2D( (2,2) ))
```

50 x 5 x 5

input



Convolution



Max Pooling



Convolution

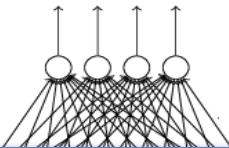


Max Pooling

CNN in Keras

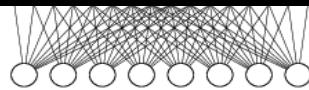
Only modified the *network structure* and *input format (vector -> 3-D tensor)*

output



Fully Connected
Feedforward network

```
model2.add(Dense(output_dim=100))  
model2.add(Activation('relu'))  
model2.add(Dense(output_dim=10))  
model2.add(Activation('softmax'))
```



1250

Flatten

```
model2.add(Flatten())
```

input

$1 \times 28 \times 28$

Convolution

$25 \times 26 \times 26$

Max Pooling

$25 \times 13 \times 13$

Convolution

$50 \times 11 \times 11$

Max Pooling

$50 \times 5 \times 5$

Live Demo

What does machine learn?

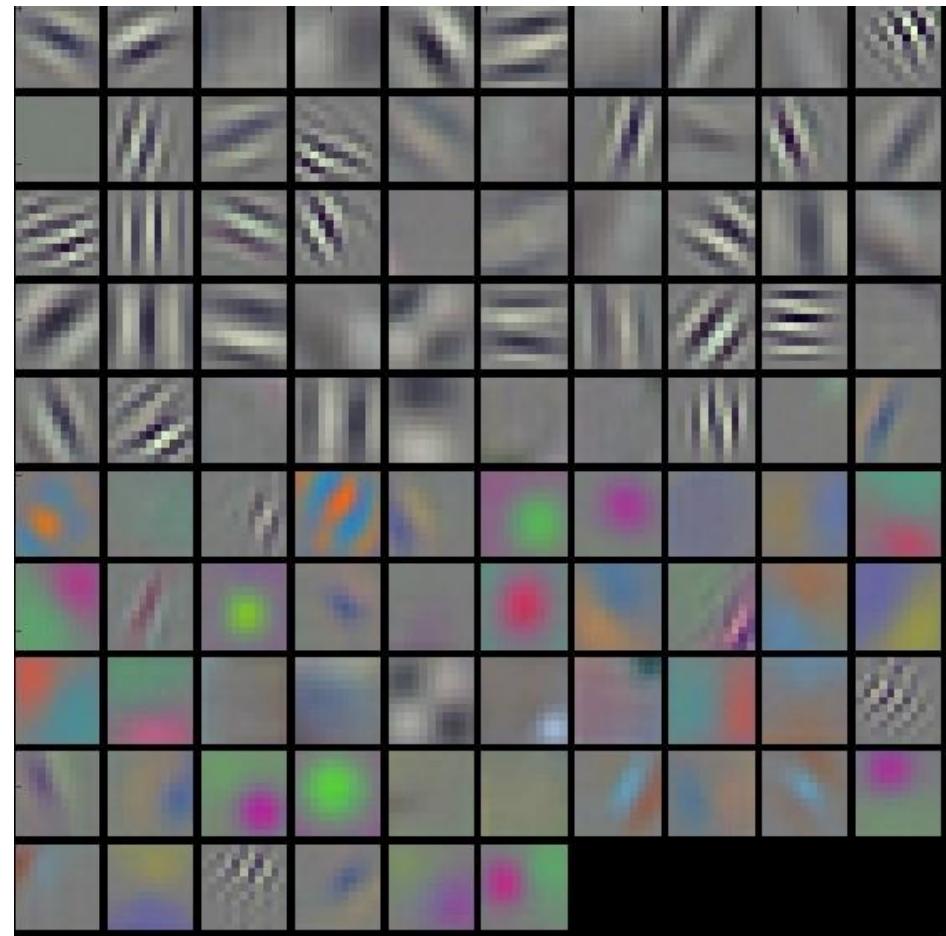


<http://newsneakernews.wpengine.netdna-cdn.com/wp-content/uploads/2016/11/rihanna-puma-creepervelvet-release-date-02.jpg>

First Convolution Layer

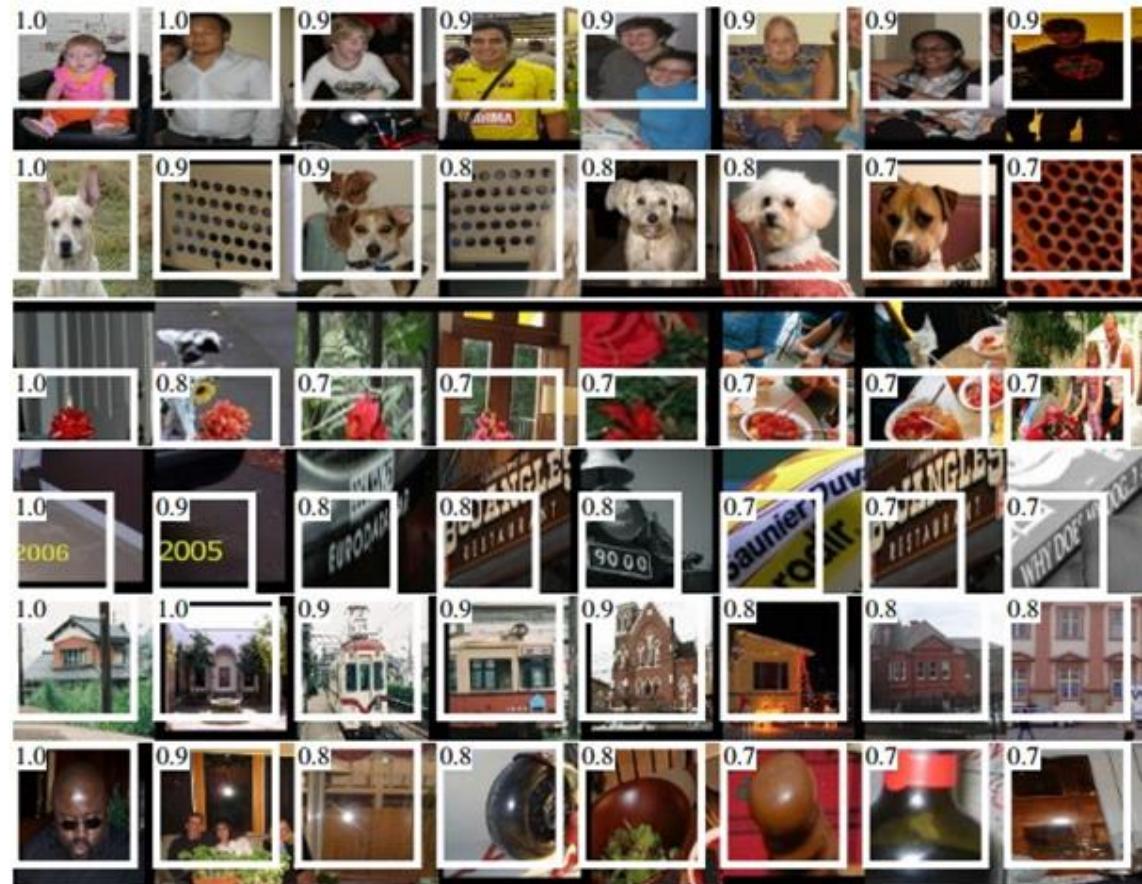
- Typical-looking filters on the trained first layer

11 x 11
(AlexNet)



How about higher layers?

- Which images make a specific neuron activate



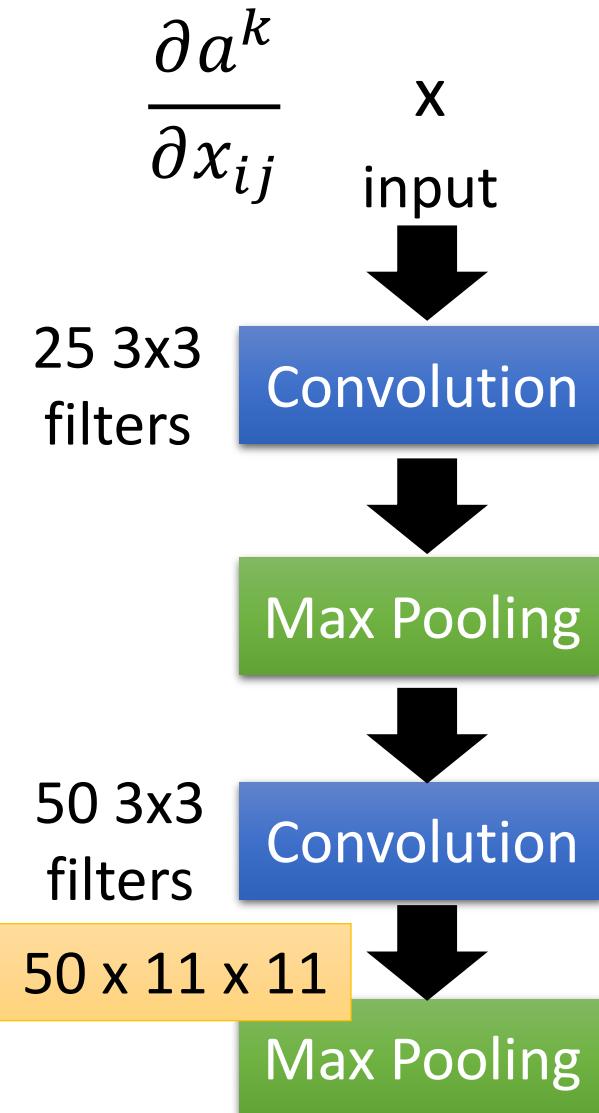
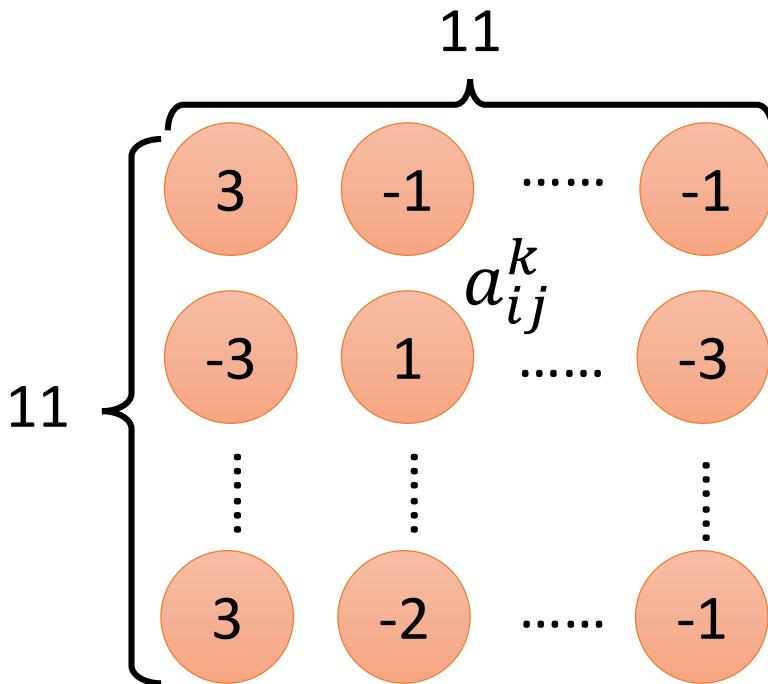
Ross Girshick, Jeff
Donahue, Trevor
Darrell, Jitendra Malik, "Rich
feature hierarchies for accurate
object detection and semantic
segmentation", CVPR, 2014

What does CNN learn?

The output of the k-th filter is a 11×11 matrix.

Degree of the activation of the k-th filter: $a^k = \sum_{i=1}^{11} \sum_{j=1}^{11} a_{ij}^k$

$$x^* = \arg \max_x a^k \text{ (gradient ascent)}$$

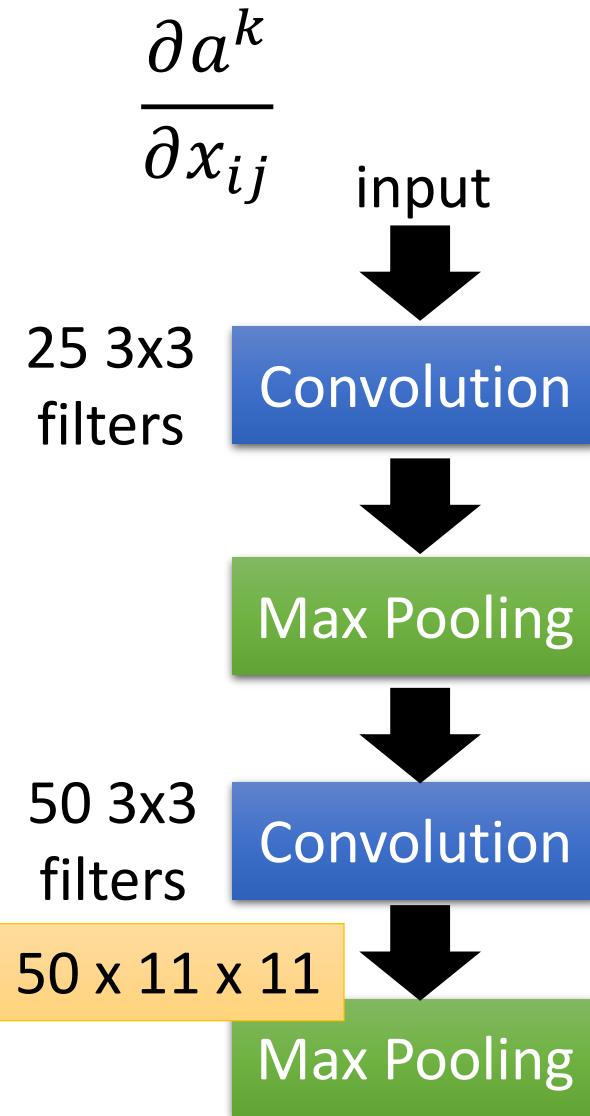
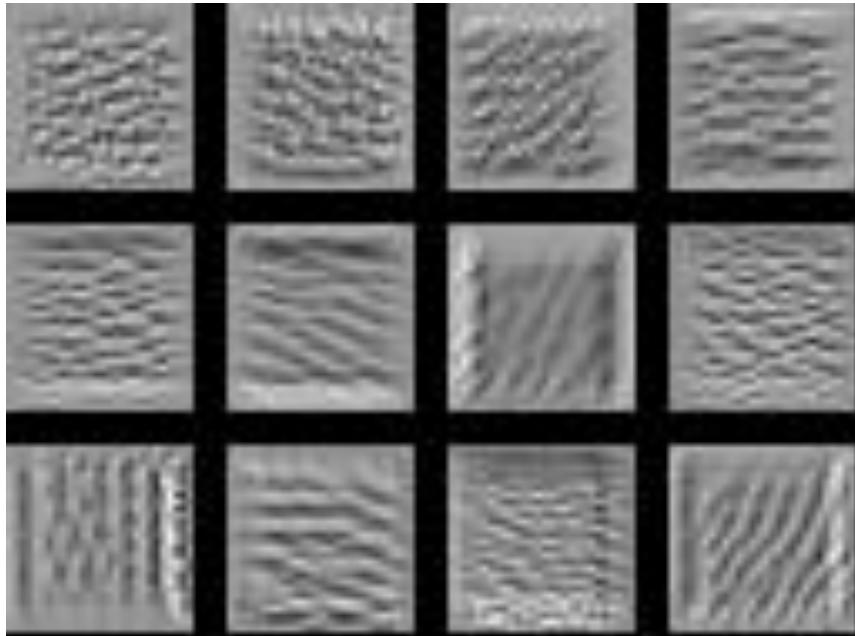


What does CNN learn?

The output of the k-th filter is a 11×11 matrix.

Degree of the activation of the k-th filter: $a^k = \sum_{i=1}^{11} \sum_{j=1}^{11} a_{ij}^k$

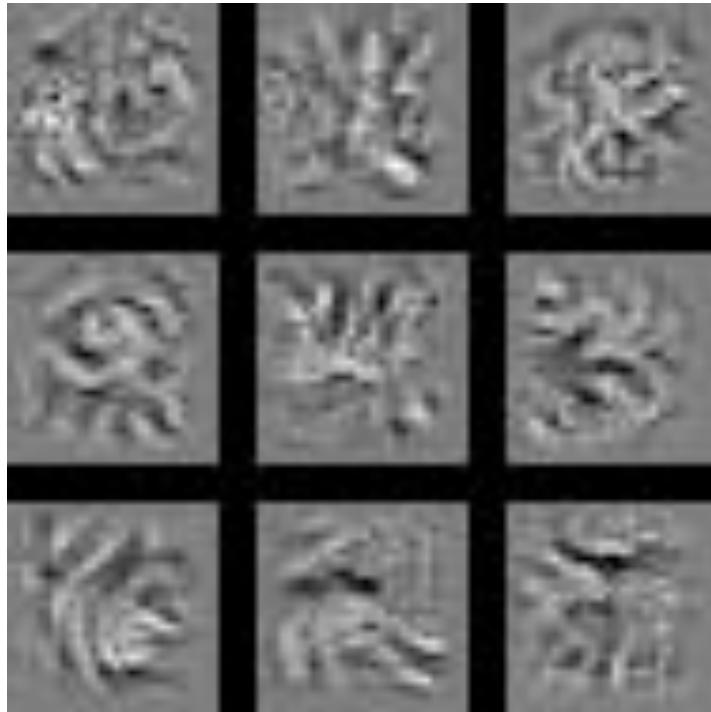
$$x^* = \arg \max_x a^k \text{ (gradient ascent)}$$



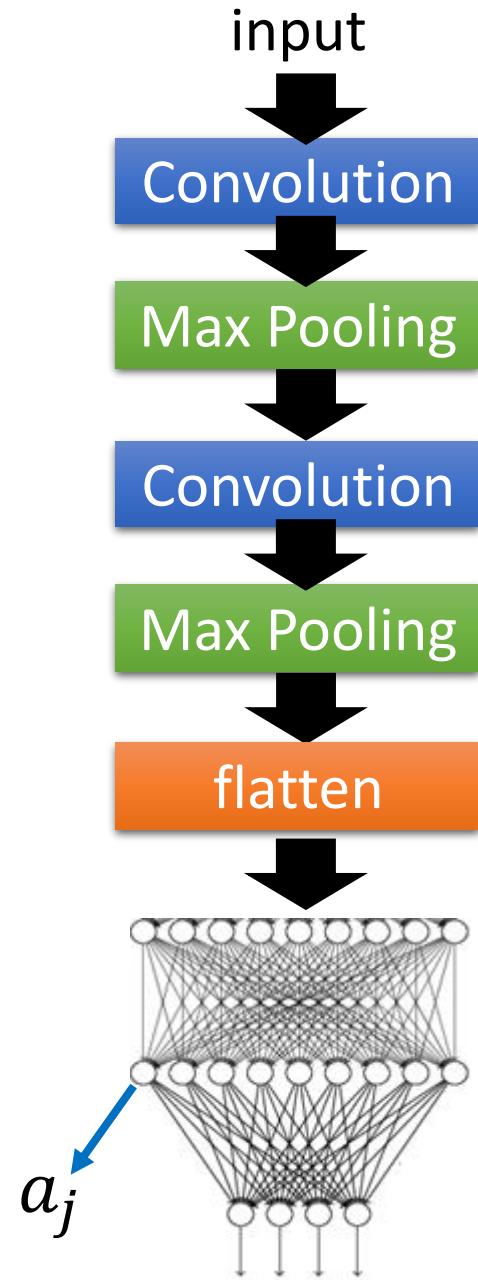
What does CNN learn?

Find an image maximizing the output of neuron:

$$x^* = \arg \max_x a_j$$



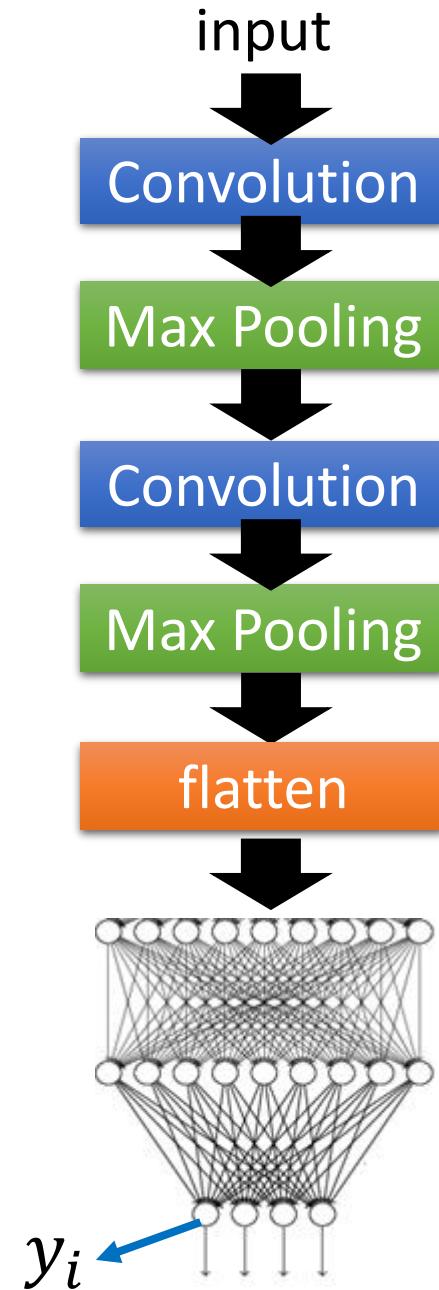
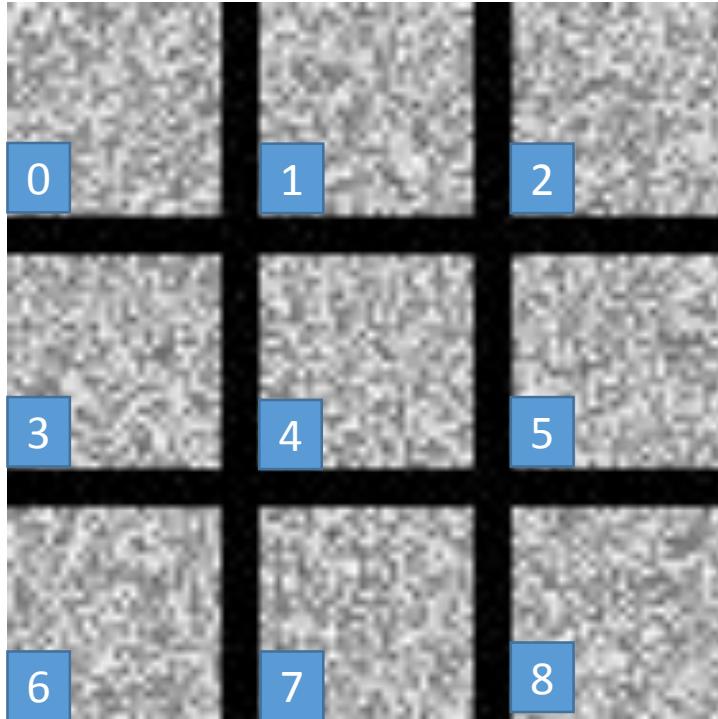
Each figure corresponds to a neuron



What does CNN learn?

$$x^* = \arg \max_x y^i$$

Can we see digits?

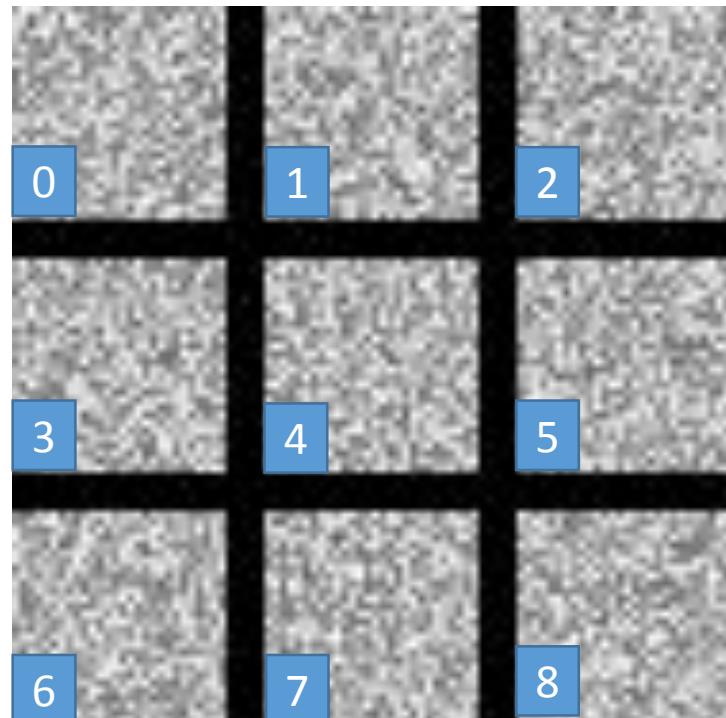


Deep Neural Networks are Easily Fooled

<https://www.youtube.com/watch?v=M2IebCN9Ht4>

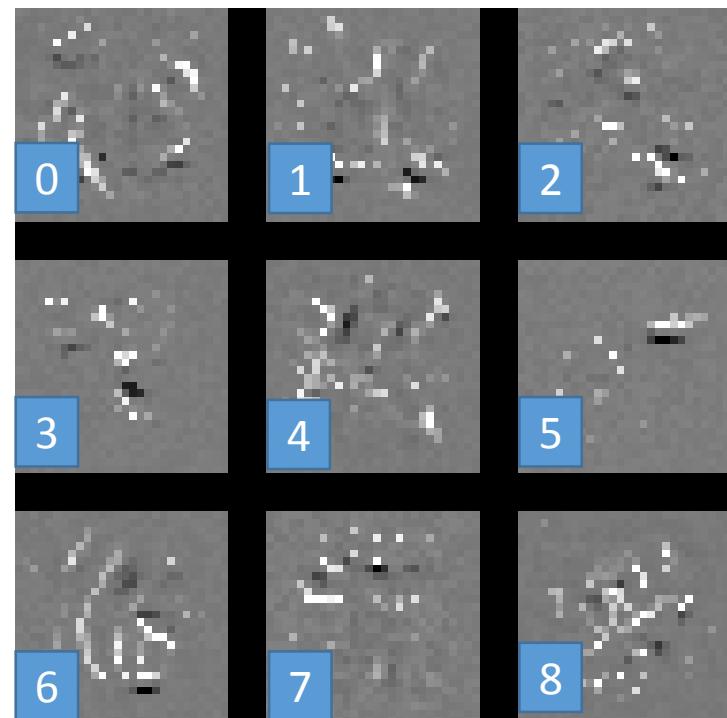
What does CNN learn?

$$x^* = \arg \max_x y^i$$



Over all
pixel values

$$x^* = \arg \max_x \left(y^i - \sum_{i,j} |x_{ij}| \right)$$

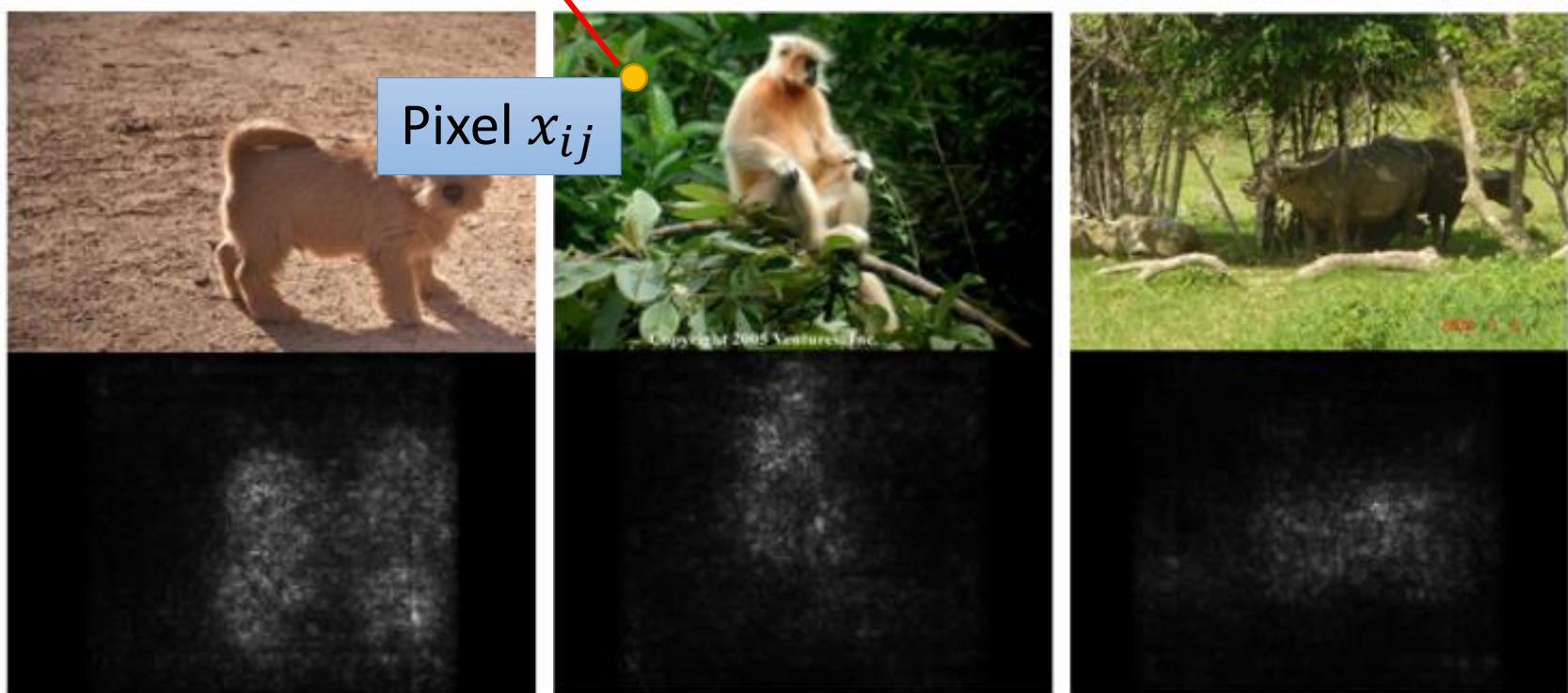




Karen Simonyan, Andrea Vedaldi, Andrew Zisserman, "Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps", ICLR, 2014

$$\left| \frac{\partial y_k}{\partial x_{ij}} \right|$$

y_k : the predicted class of the model



Karen Simonyan, Andrea Vedaldi, Andrew Zisserman, "Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps", ICLR, 2014



True Label: Pomeranian



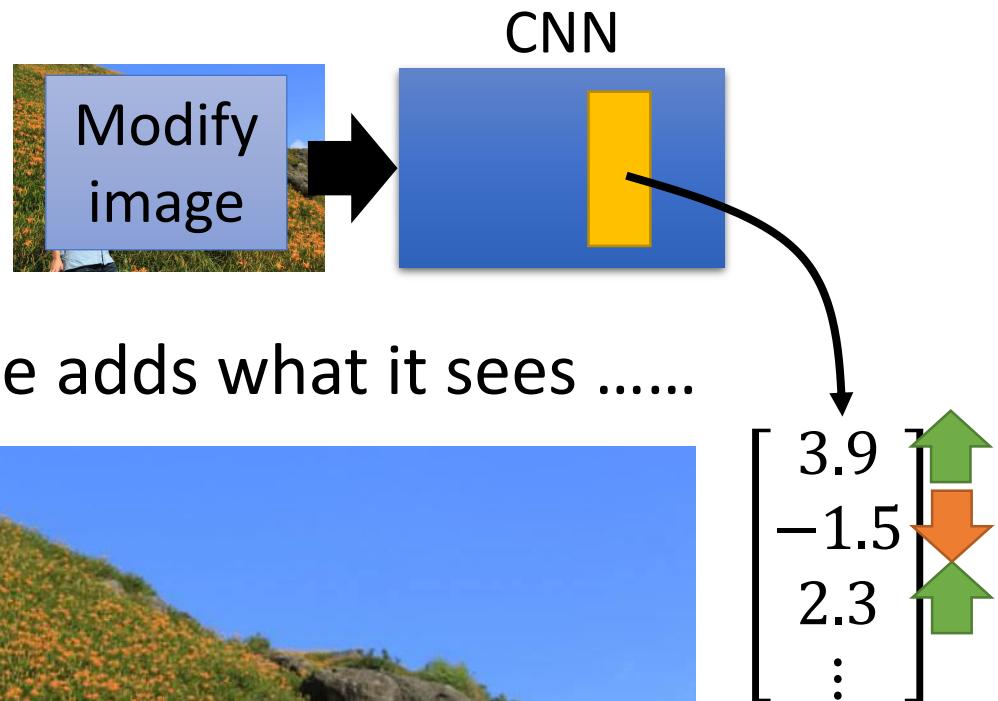
True Label: Car Wheel



True Label: Afghan Hound

Reference: Zeiler, M. D., & Fergus, R. (2014). Visualizing and understanding convolutional networks. In *Computer Vision–ECCV 2014* (pp. 818-833)

Deep Dream

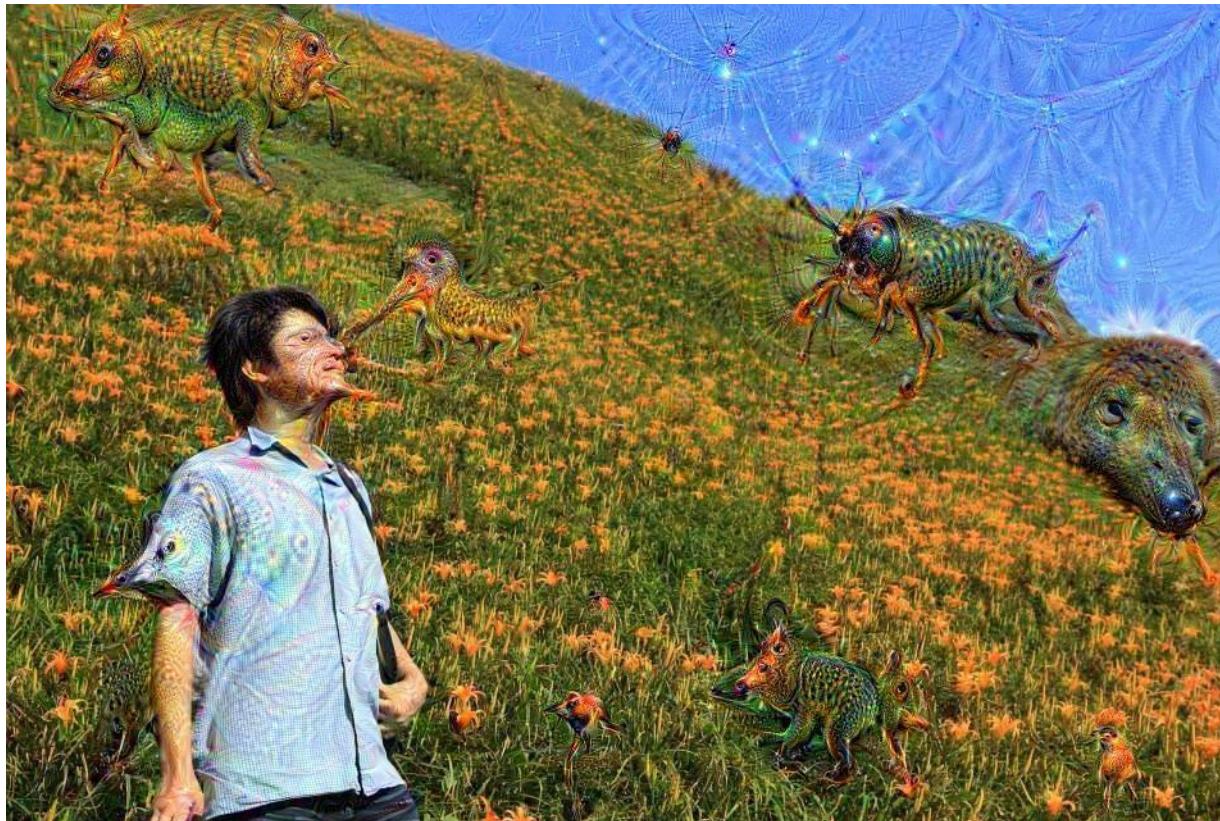


- Given a photo, machine adds what it sees



Deep Dream

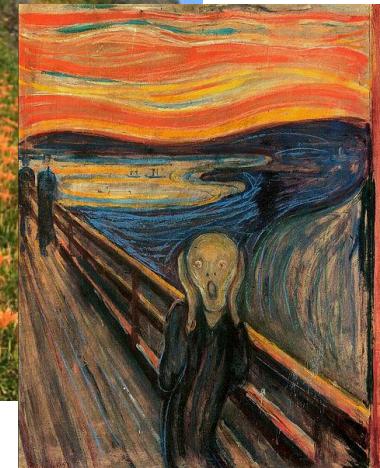
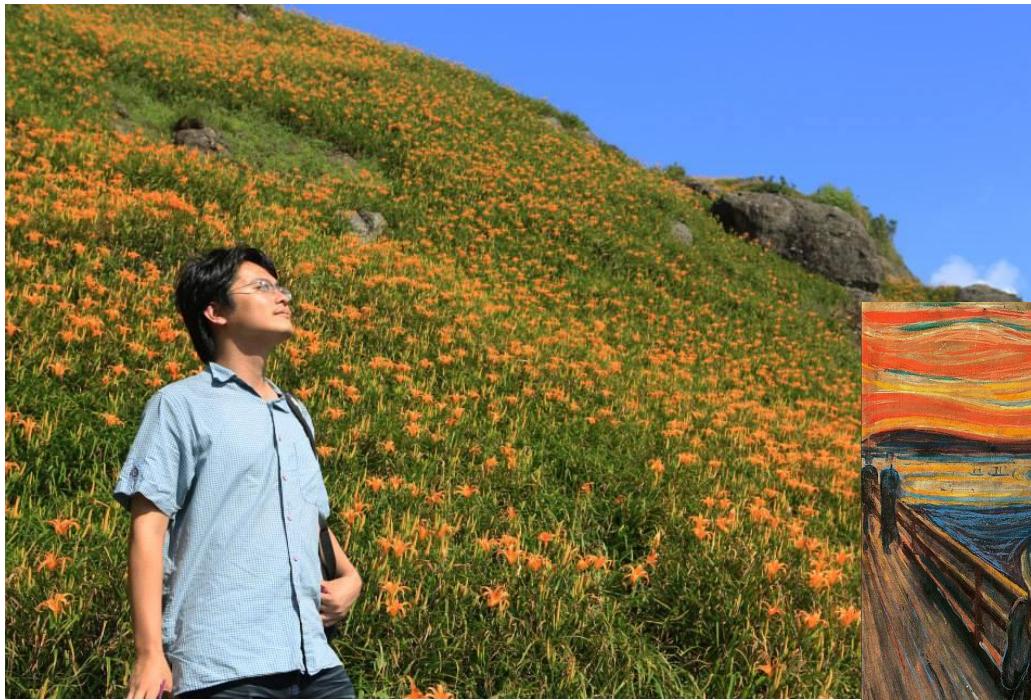
- Given a photo, machine adds what it sees



<http://deepdreamgenerator.com/>

Deep Style

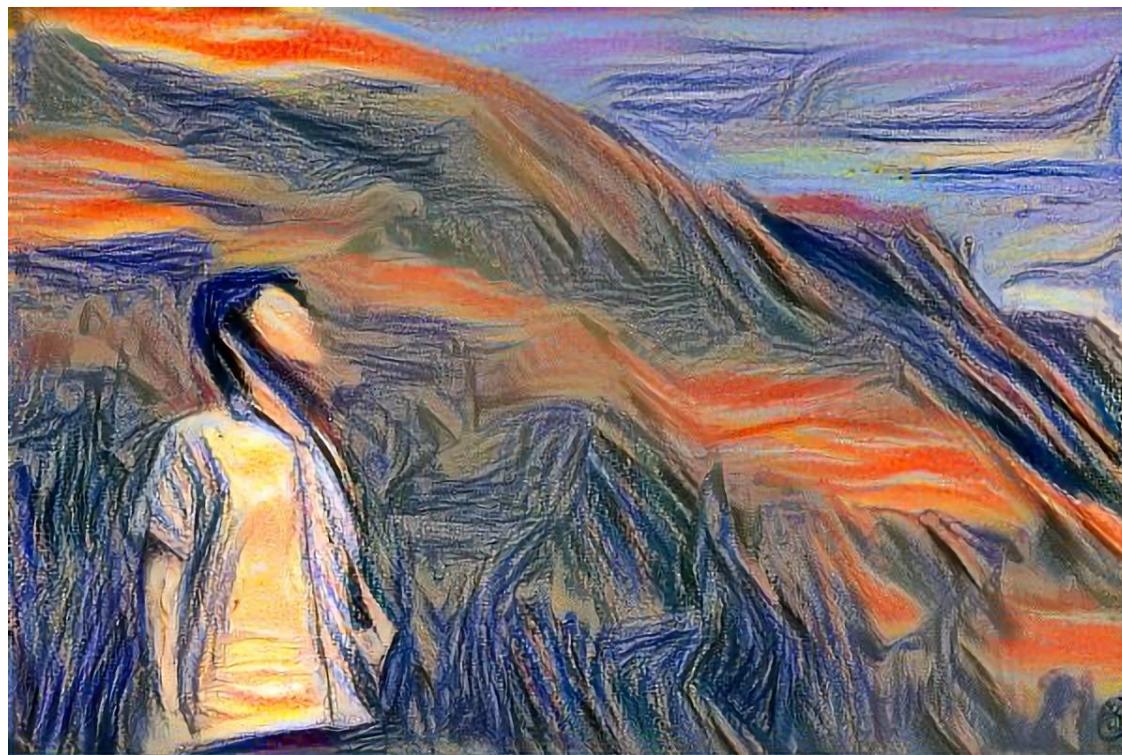
- Given a photo, make its style like famous paintings



<https://dreamscopeapp.com/>

Deep Style

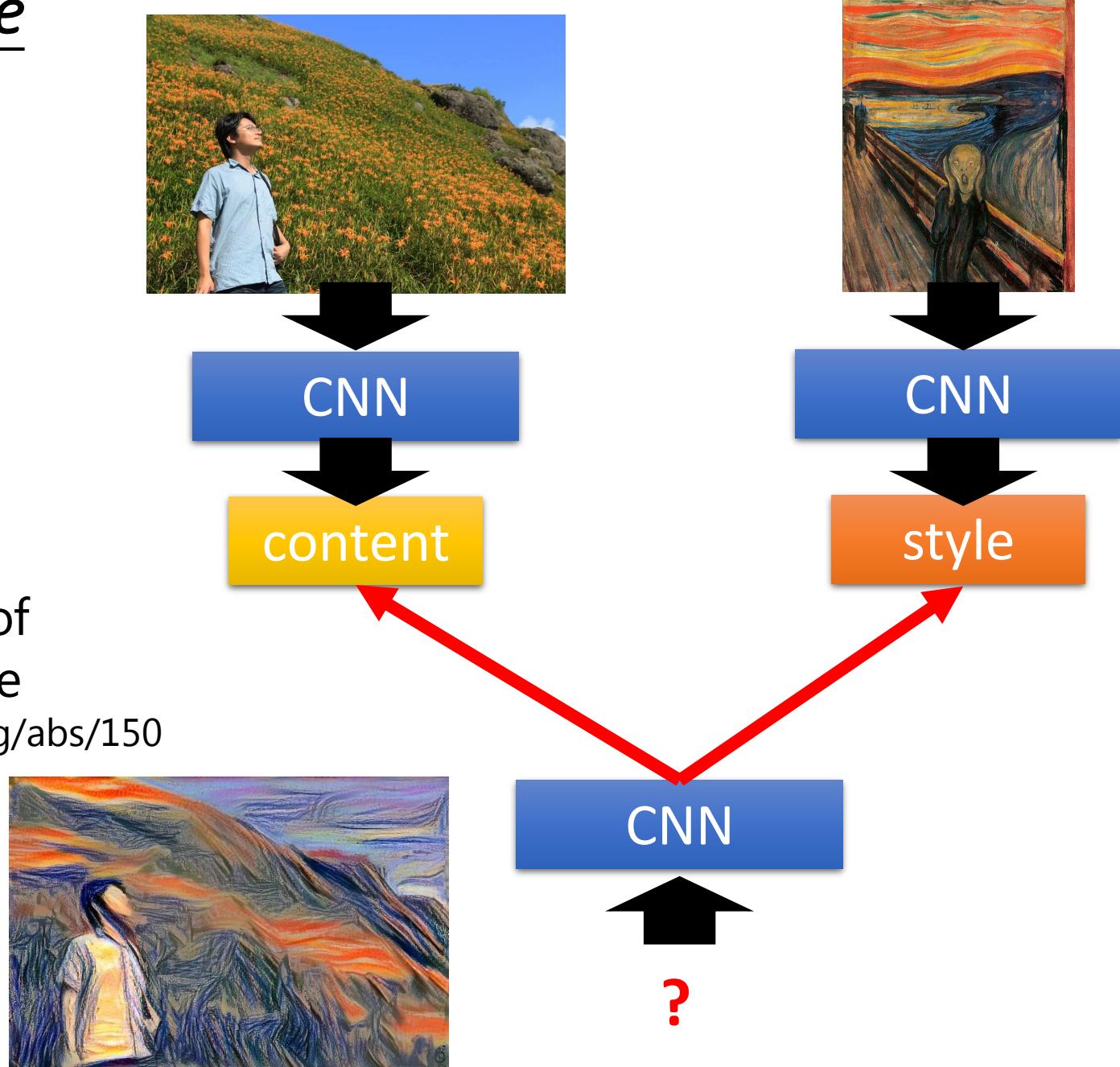
- Given a photo, make its style like famous paintings



<https://dreamscopeapp.com/>

Deep Style

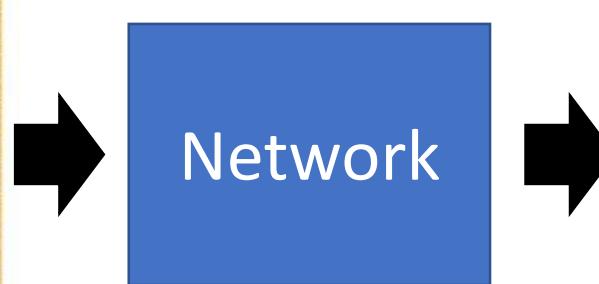
A Neural
Algorithm of
Artistic Style
<https://arxiv.org/abs/1508.06576>



More Application: Playing Go



19 x 19 matrix
(image)



Next move
(19 x 19
positions)

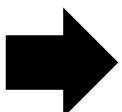
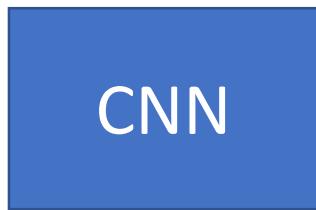
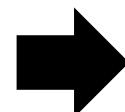
Black: 1
white: -1
none: 0

Fully-connected feedforward
network can be used

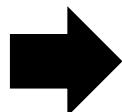
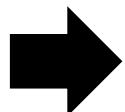
But CNN performs much better.

More Application: Playing Go

Training: record of previous plays 黑: 5之五 → 白: 天元 → 黑: 五之5 ...



Target:
“天元” = 1
else = 0

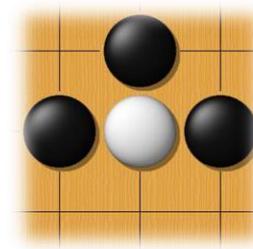


Target:
“五之5” = 1
else = 0

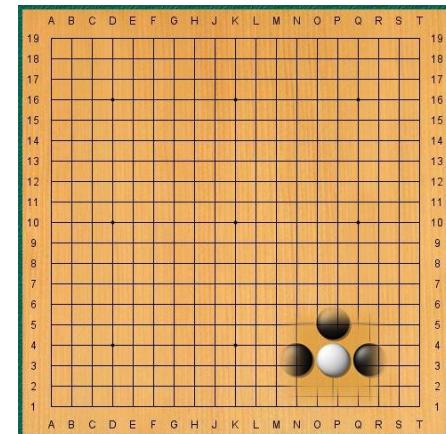
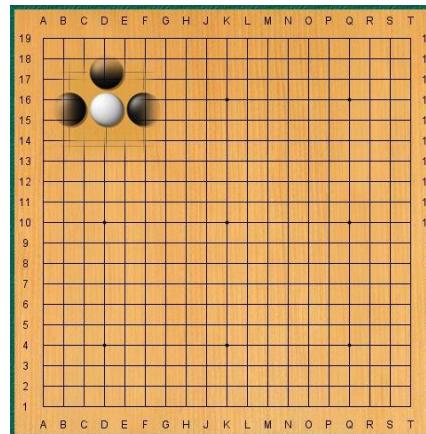
Why CNN for playing Go?

- Some patterns are much smaller than the whole image

Alpha Go uses 5×5 for first layer



- The same patterns appear in different regions.



Why CNN for playing Go?

- Subsampling the pixels will not change the object

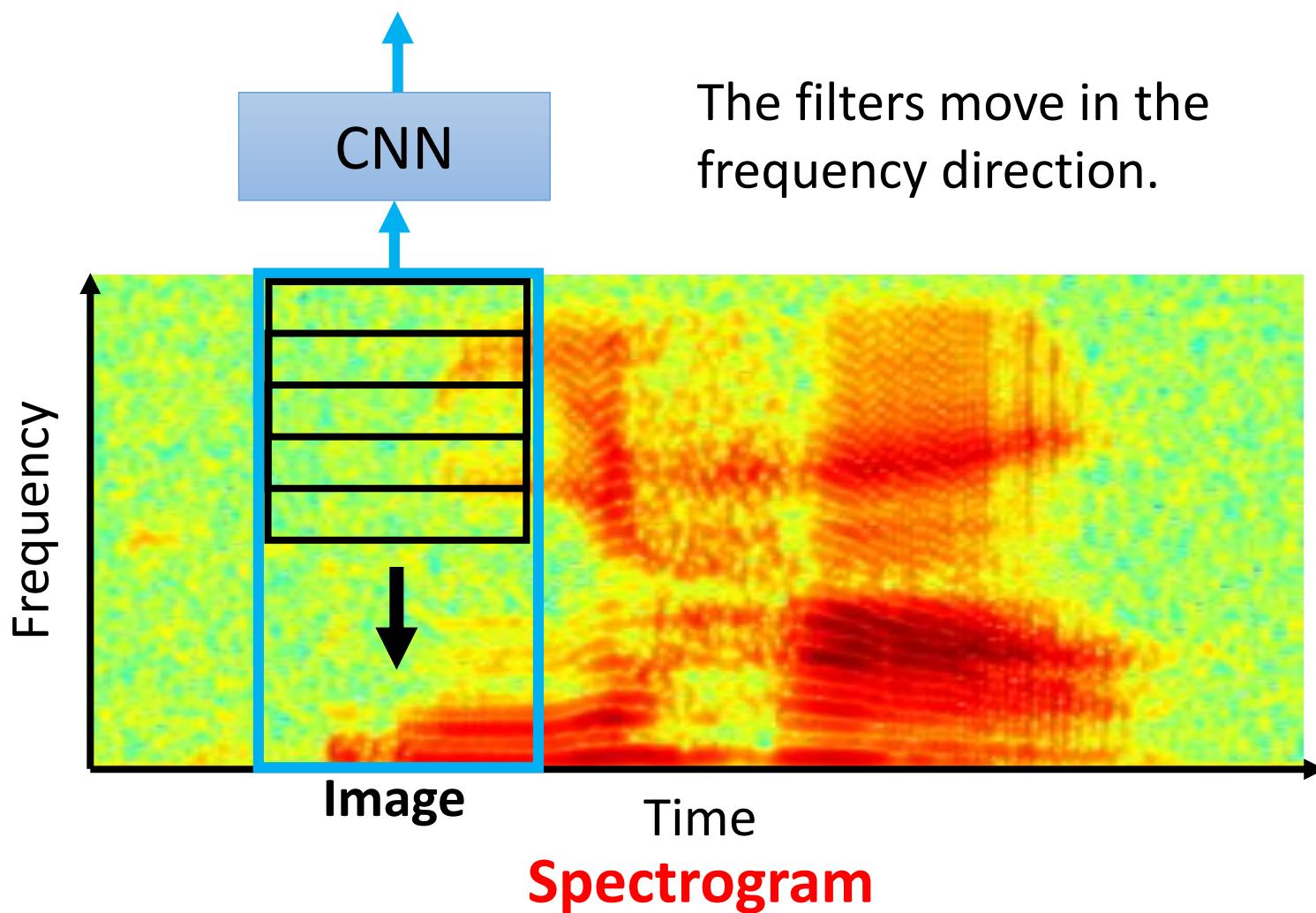


Max Pooling

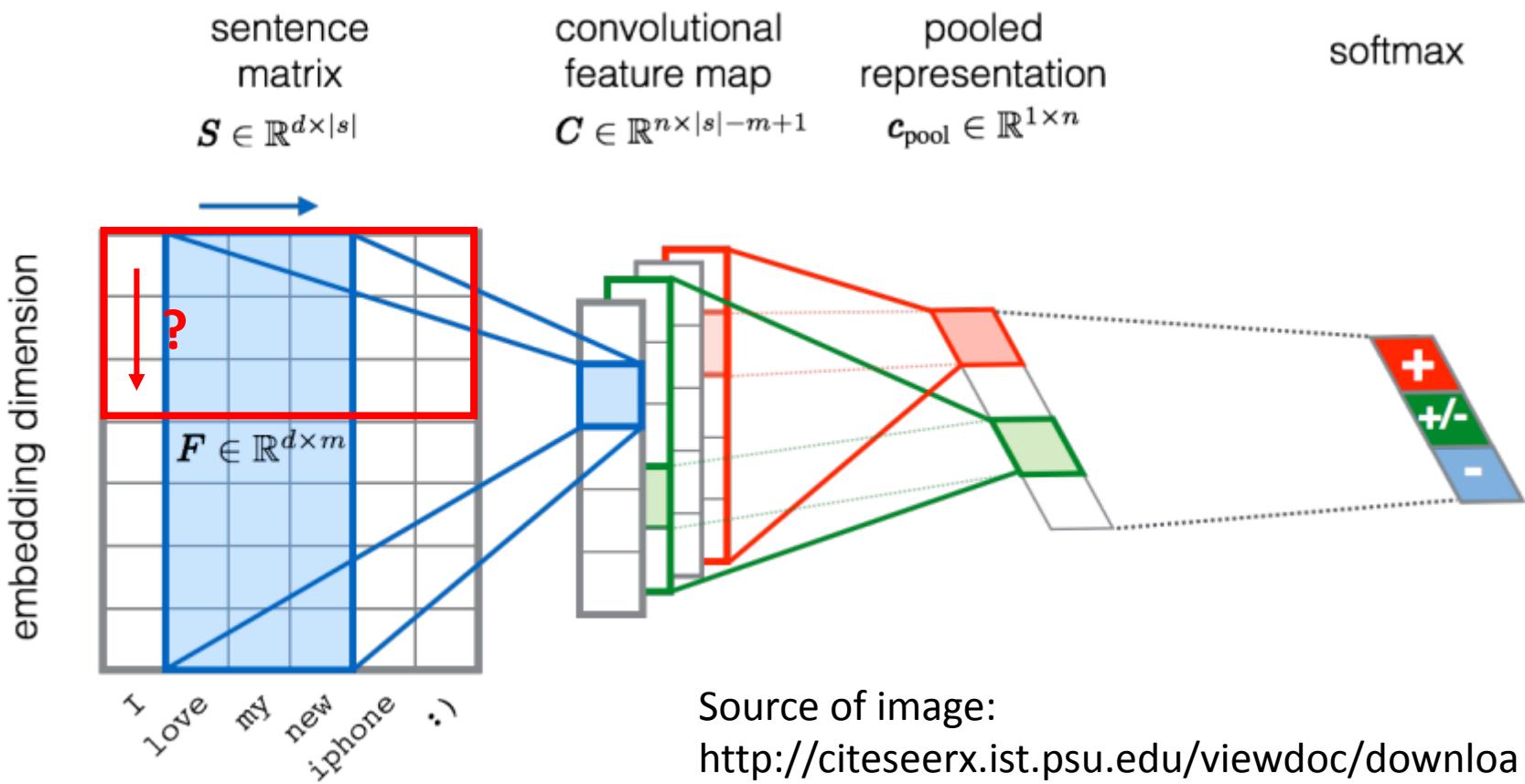
How to explain this???

Neural network architecture. The input to the policy network is a $19 \times 19 \times 48$ image stack consisting of 48 feature planes. The first hidden layer zero pads the input into a 23×23 image, then convolves k filters of kernel size 5×5 with stride 1 with the input image and applies a rectifier nonlinearity. Each of the subsequent hidden layers 2 to 12 zero pads the respective previous hidden layer into a 21×21 image, then convolves k filters of kernel size 3×3 with stride 1, again followed by a rectifier nonlinearity. The final layer convolves 1 filter of kernel size 1×1 with stride 1 with a different bias for each position and applies a softmax function. The Alpha Go does not use Max Pooling Extended Data Table 3 additionally show the results of training with $k = 128, 256$ and 384 filters.

More Application: Speech



More Application: Text



Source of image:
<http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.703.6858&rep=rep1&type=pdf>

Acknowledgment

- 感謝 Guobiao Mo 發現投影片上的打字錯誤

Why Deep Learning?

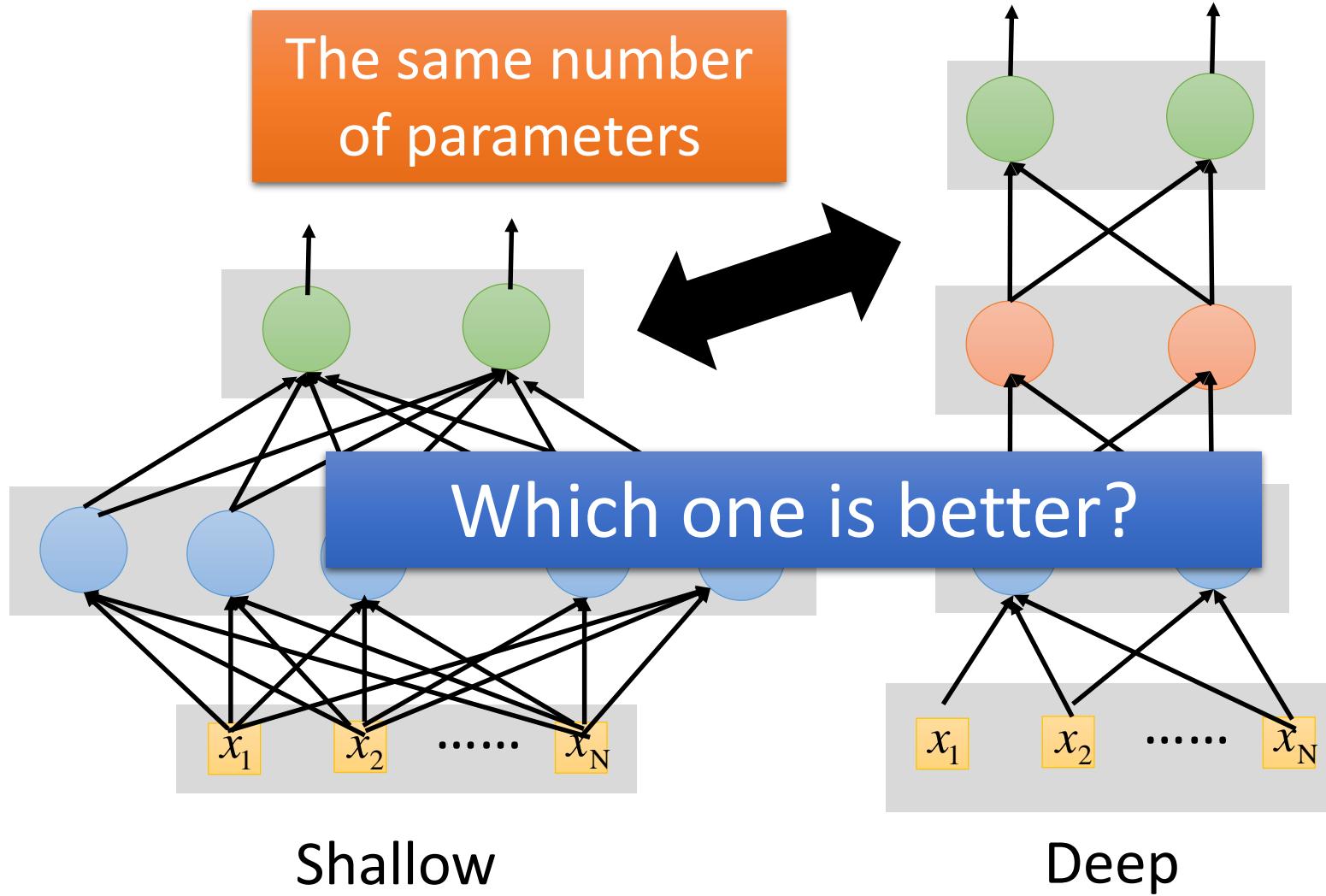
Deeper is Better?

Layer X Size	Word Error Rate (%)
1 X 2k	24.2
2 X 2k	20.4
3 X 2k	18.4
4 X 2k	17.8
5 X 2k	17.2
7 X 2k	17.1

Not surprised, more parameters, better performance

Seide, Frank, Gang Li, and Dong Yu. "Conversational Speech Transcription Using Context-Dependent Deep Neural Networks." *Interspeech*. 2011.

Fat + Short v.s. Thin + Tall



Fat + Short v.s. Thin + Tall

Layer X Size	Word Error Rate (%)	Layer X Size	Word Error Rate (%)
1 X 2k	24.2		
2 X 2k	20.4		
3 X 2k	18.4		
4 X 2k	17.8		
5 X 2k	17.2	1 X 3772	22.5
7 X 2k	17.1	1 X 4634	22.6
		1 X 16k	22.1

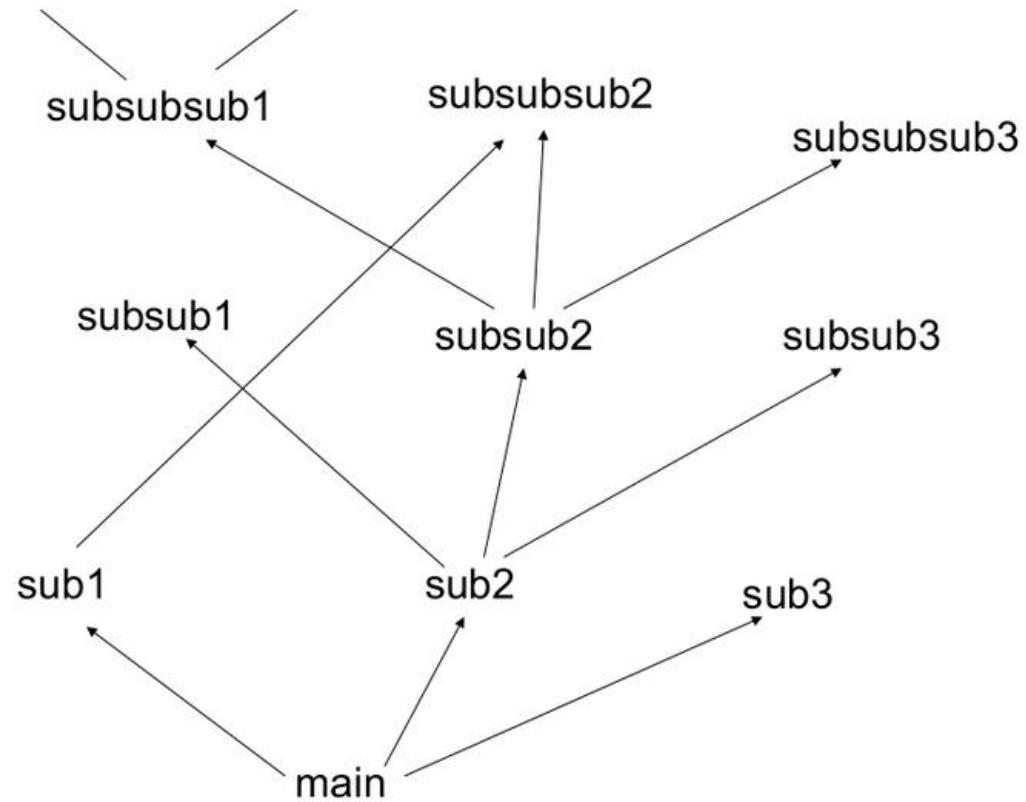
Why?

Seide, Frank, Gang Li, and Dong Yu. "Conversational Speech Transcription Using Context-Dependent Deep Neural Networks." *Interspeech*. 2011.

Modularization

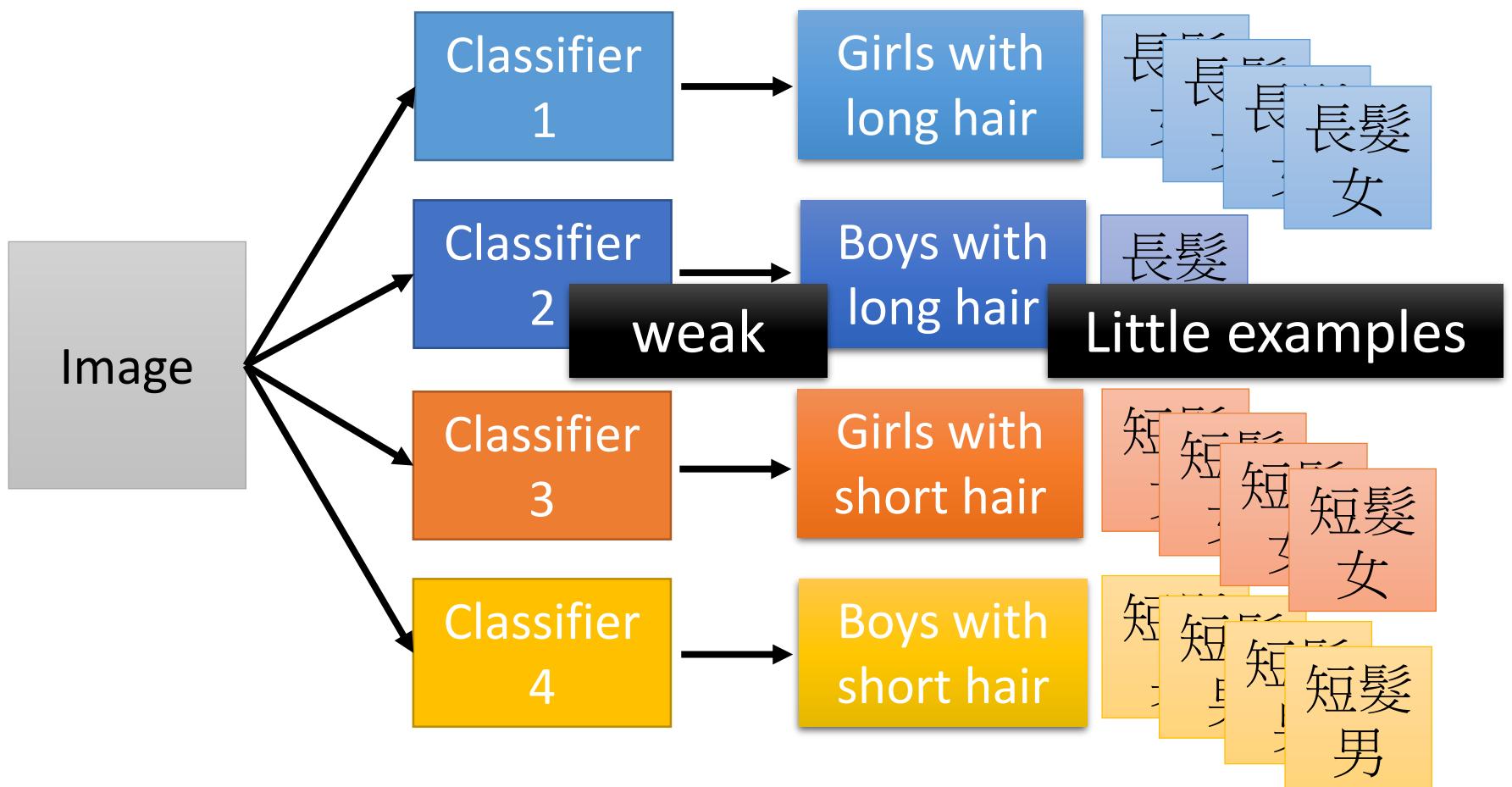
- Deep → Modularization

Don't put
everything in your
main function.



Modularization

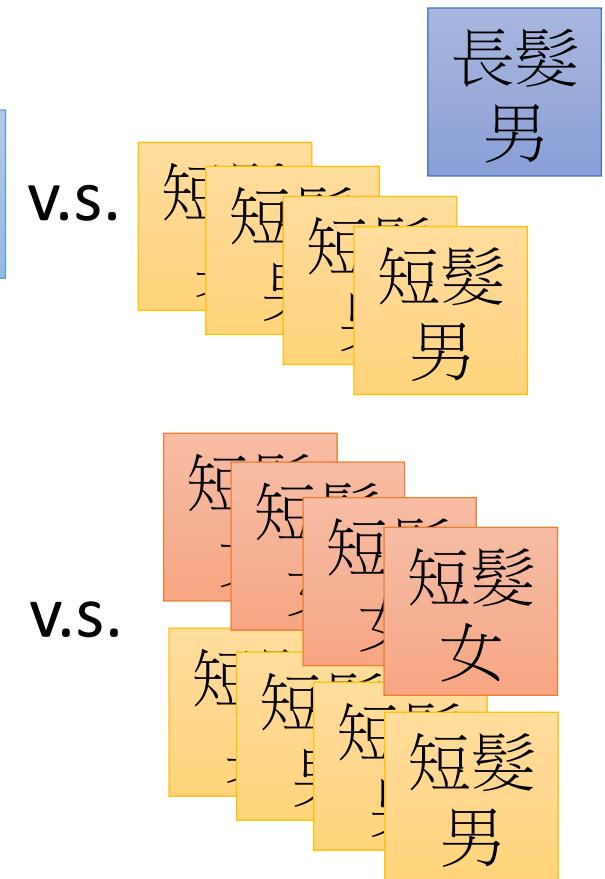
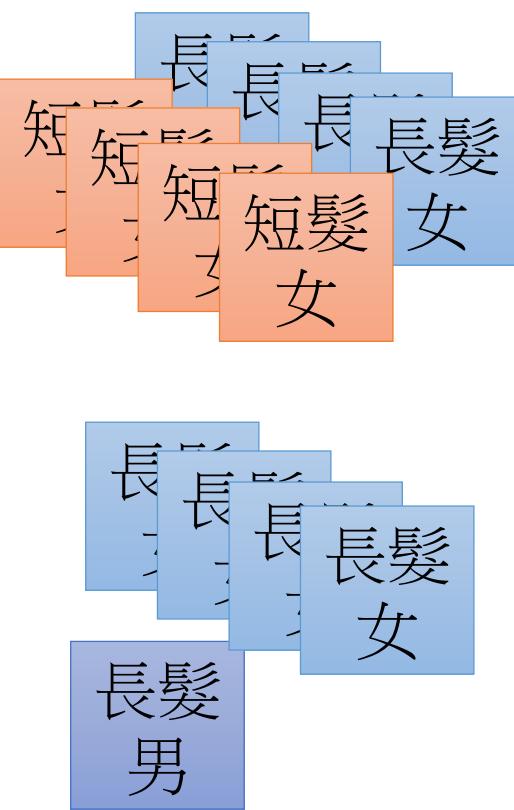
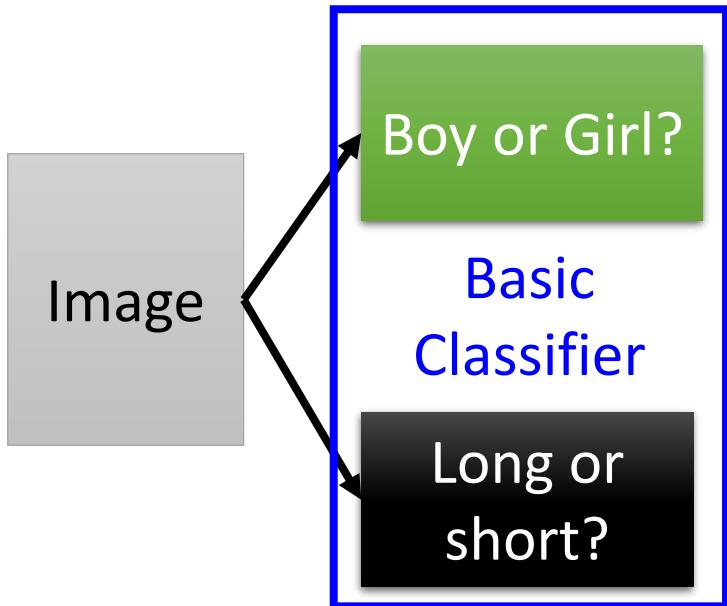
- Deep → Modularization



Modularization

Each basic classifier can have sufficient training examples.

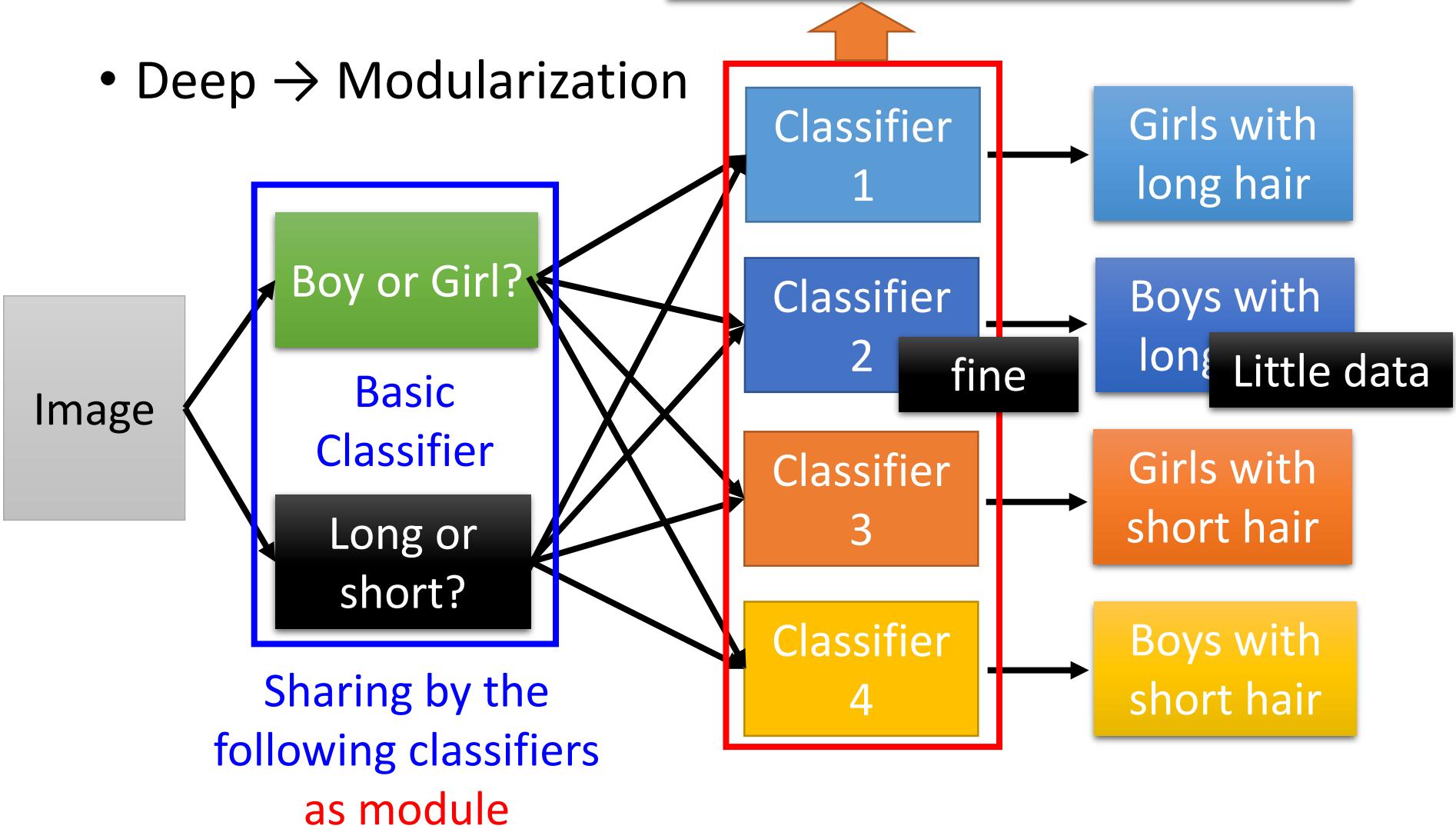
- Deep → Modularization



Modularization

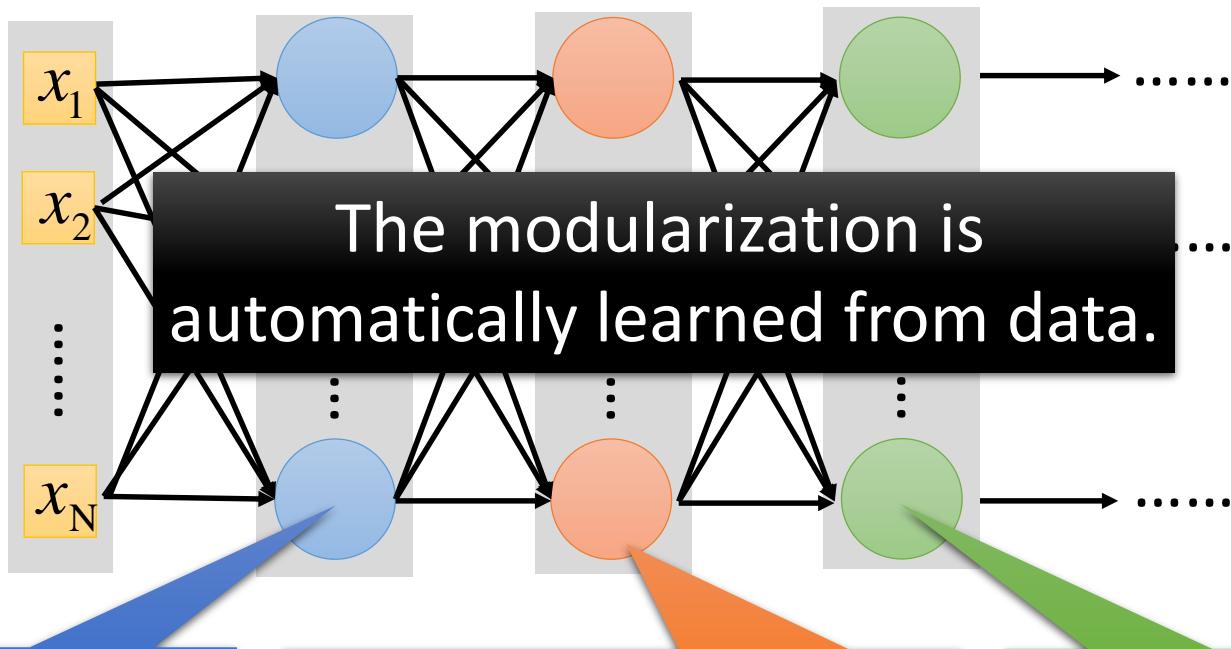
- Deep → Modularization

can be trained by little data



Modularization

- Deep \rightarrow Modularization → Less training data?



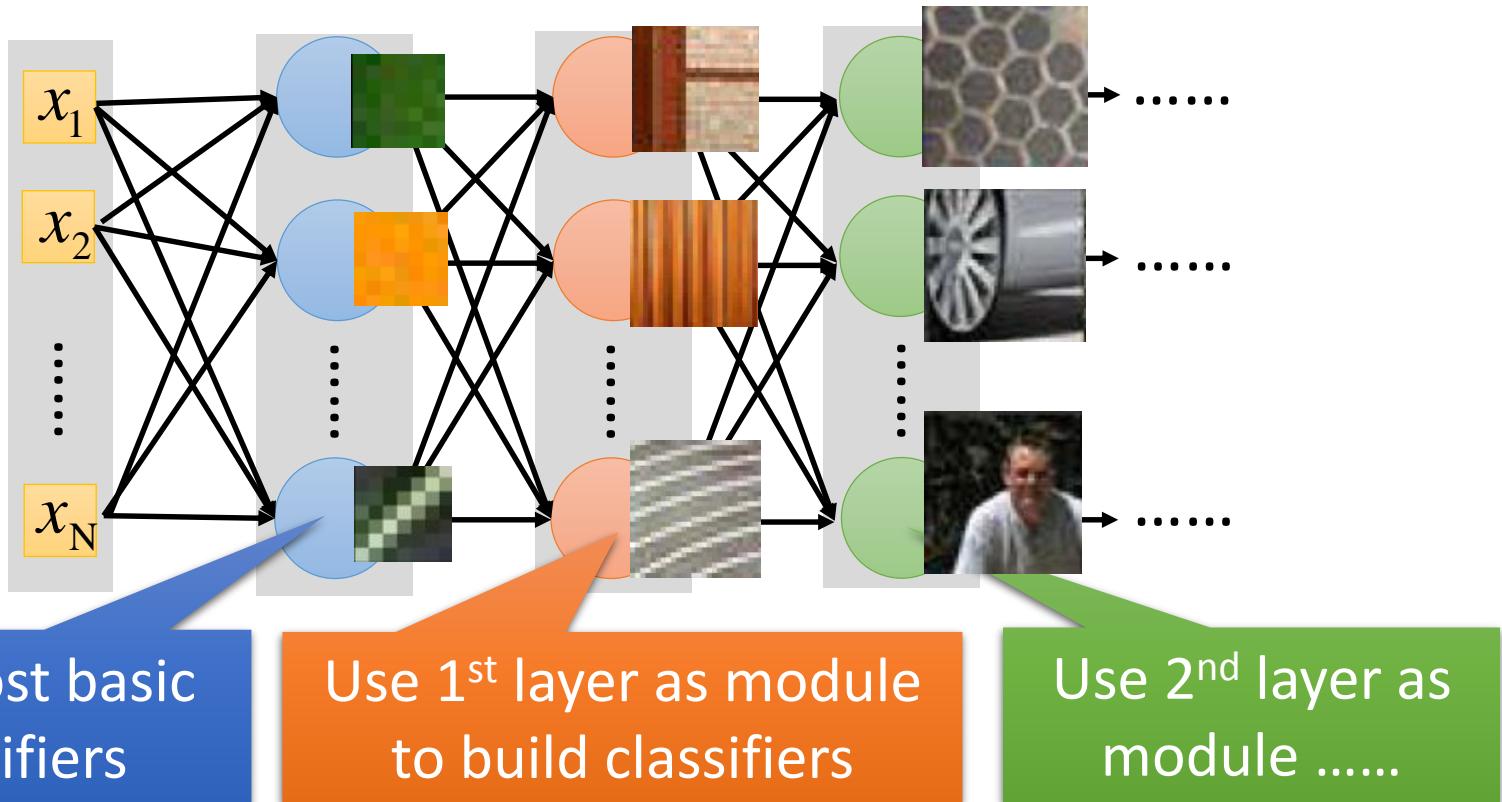
The most basic
classifiers

Use 1st layer as module
to build classifiers

Use 2nd layer as
module

Modularization - Image

- Deep → Modularization



The most basic
classifiers

Use 1st layer as module
to build classifiers

Use 2nd layer as
module

Reference: Zeiler, M. D., & Fergus, R. (2014). Visualizing and understanding convolutional networks. In *Computer Vision–ECCV 2014* (pp. 818-833)

Modularization - Speech

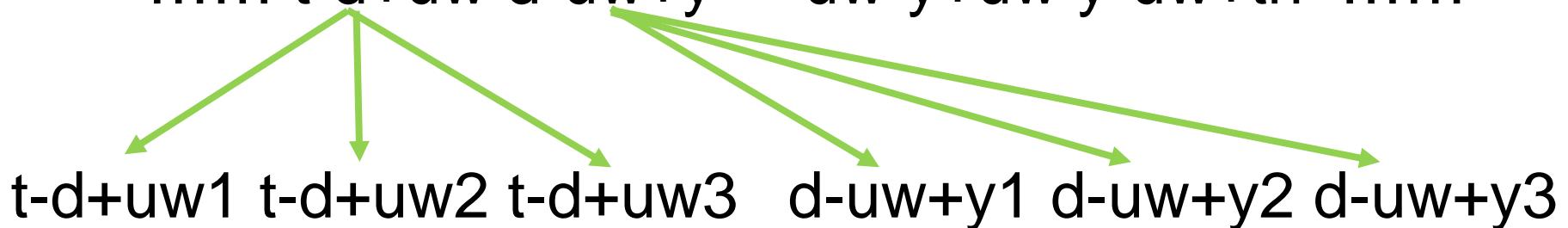
- The hierarchical structure of human languages
what do you think

Phoneme:

hh w aa t d uw y uw th ih ng k

Tri-phone:

..... t-d+uw d-uw+y uw-y+uw y-uw+th



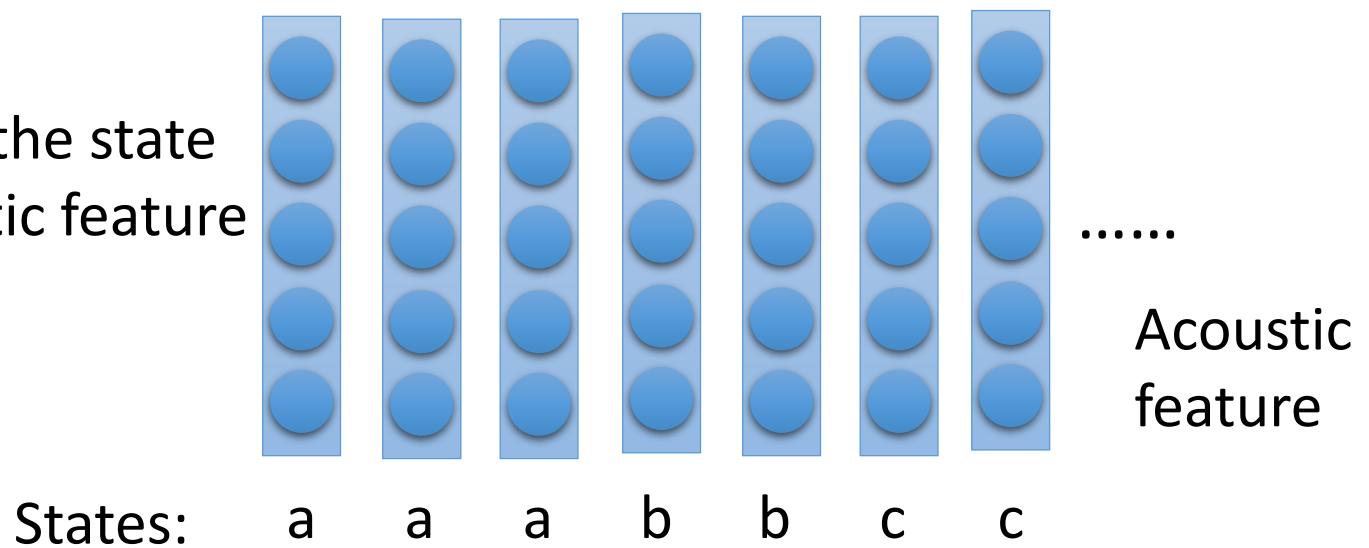
State:

Modularization - Speech

- The first stage of speech recognition
 - Classification: input → acoustic feature, output → state



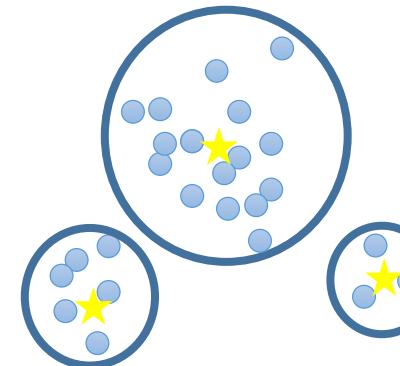
Determine the state each acoustic feature belongs to



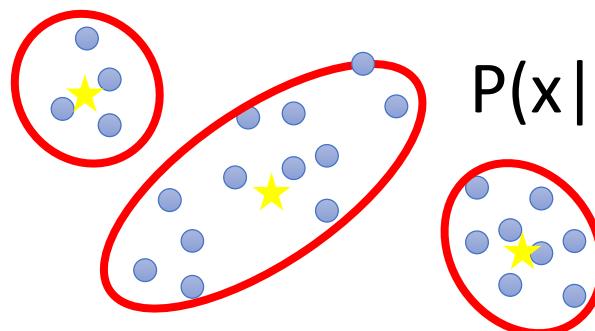
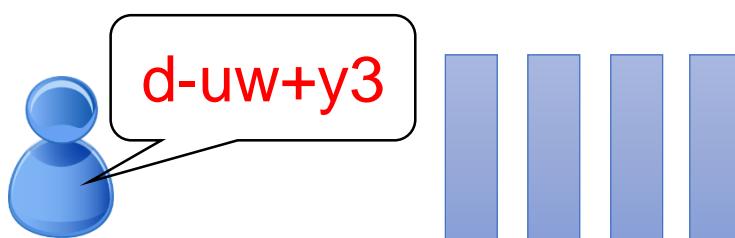
Modularization - Speech

- Each state has a stationary distribution for acoustic features

Gaussian Mixture Model (GMM)



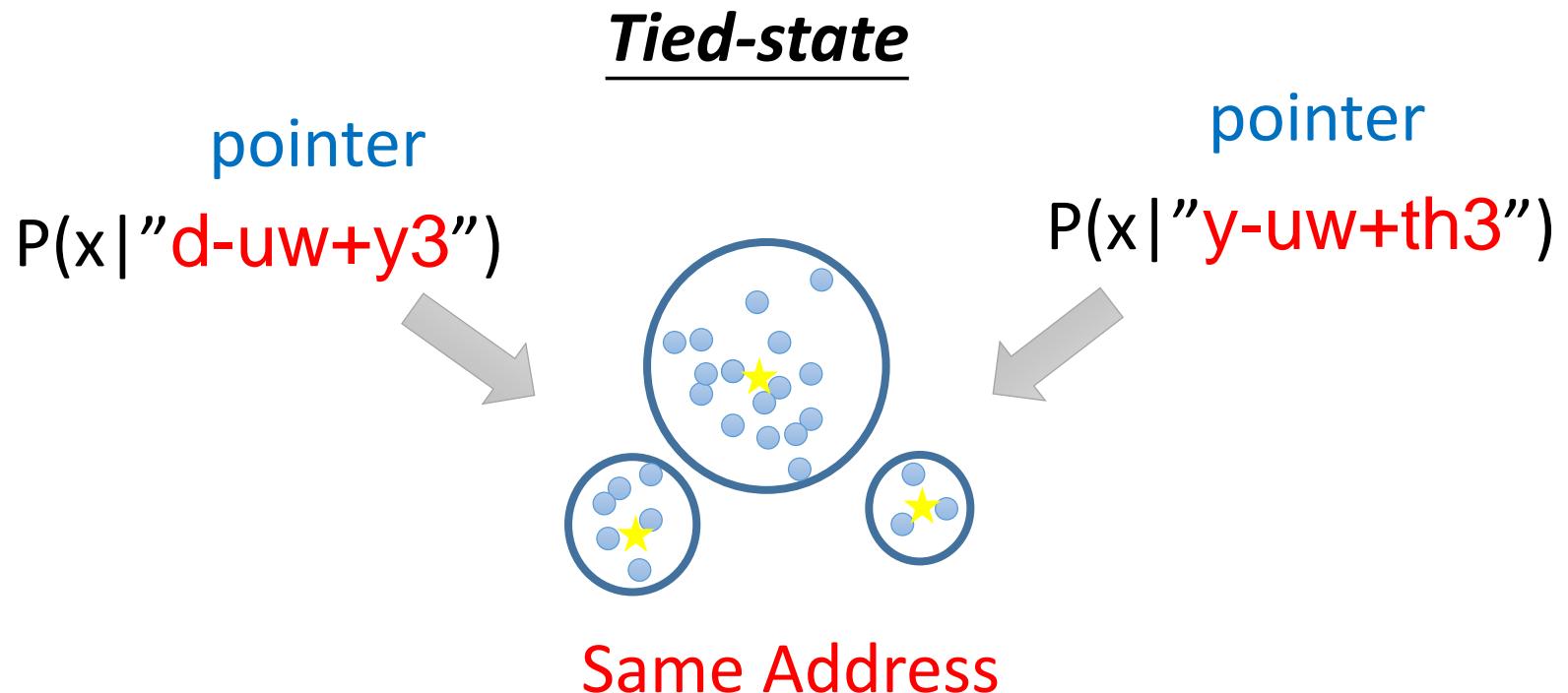
$P(x | "t-d+uw1")$



$P(x | "d-uw+y3")$

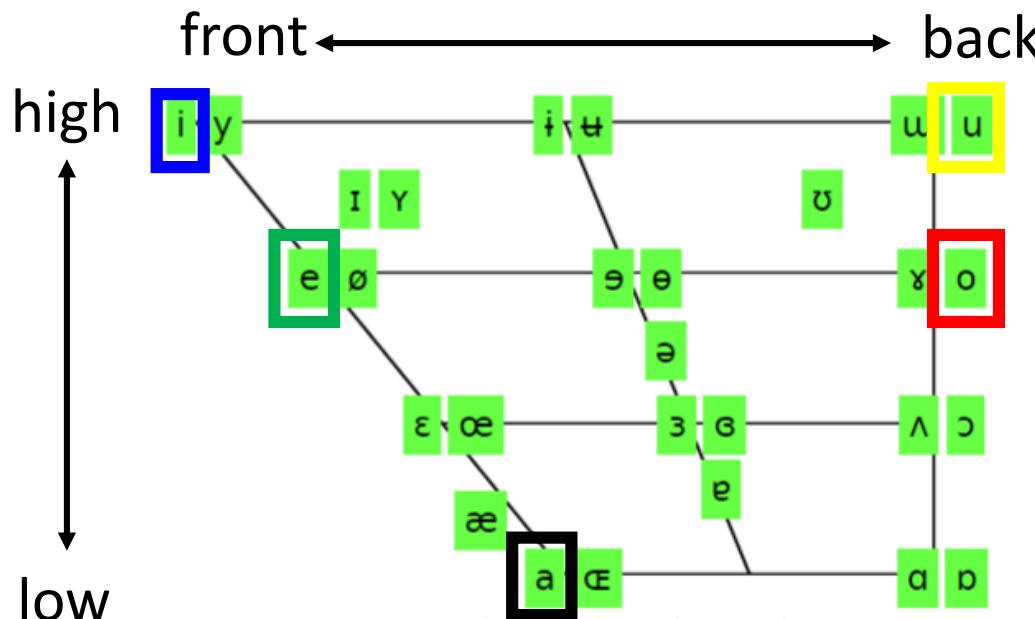
Modularization - Speech

- Each state has a stationary distribution for acoustic features



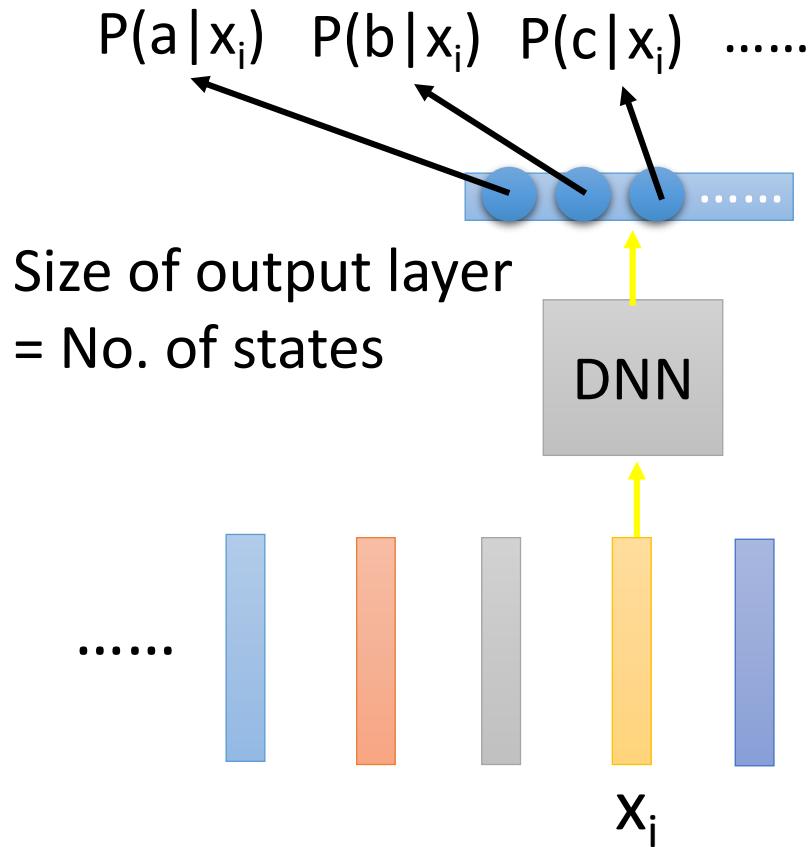
Modularization - Speech

- In HMM-GMM, all the phonemes are modeled independently
 - Not an effective way to model human voice



The sound of vowel is
only controlled by a few
factors.

Modularization - Speech



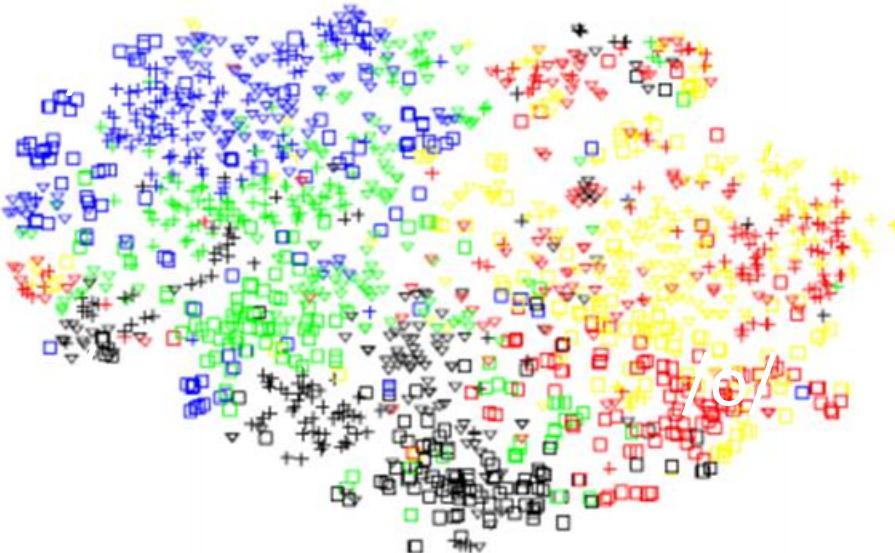
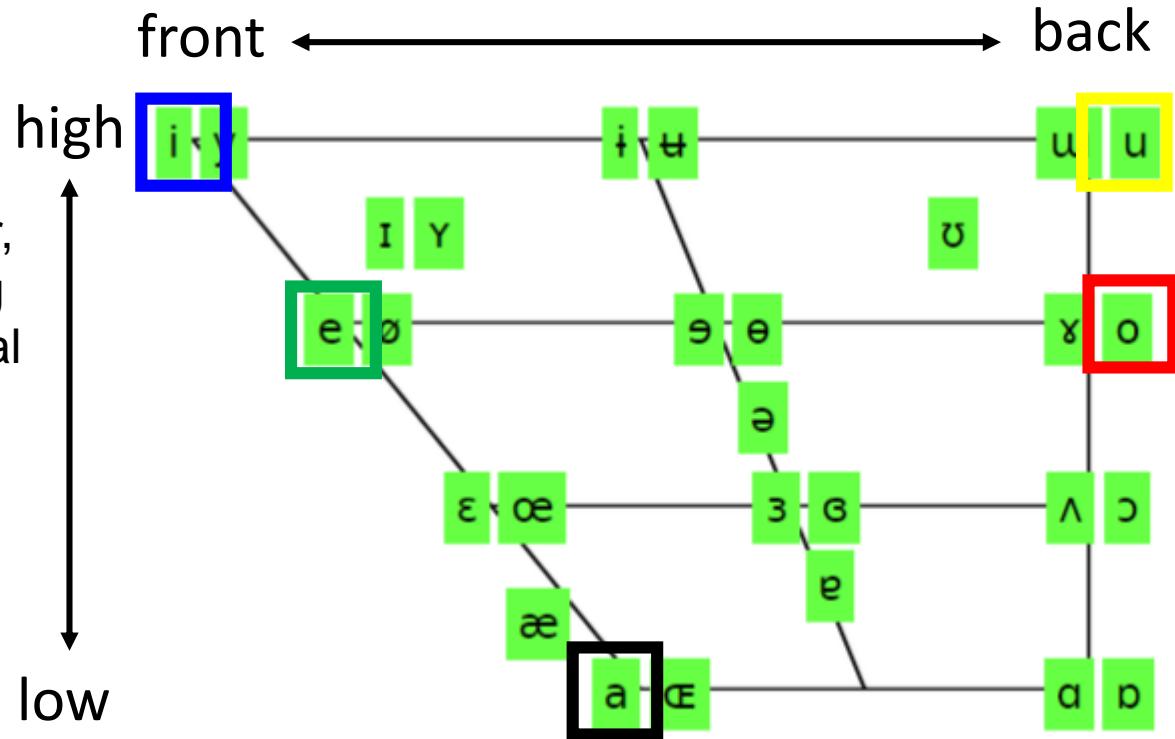
- DNN input:
One acoustic feature
- DNN output:
Probability of each state

All the states use
the same DNN

Modularization

Vu, Ngoc Thang, Jochen Weiner, and Tanja Schultz. "Investigating the Learning Effect of Multilingual Bottle-Neck Features for ASR." *Interspeech*. 2014.

Output of hidden
layer reduce to two
dimensions



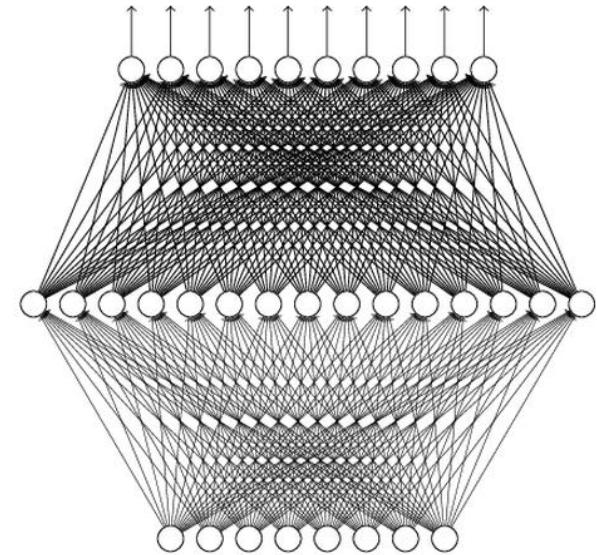
- The lower layers detect the manner of articulation
 - All the phonemes share the results from the same set of detectors.
 - Use parameters effectively

Universality Theorem

Any continuous function f

$$f : R^N \rightarrow R^M$$

Can be realized by a network
with one hidden layer
(given **enough** hidden neurons)

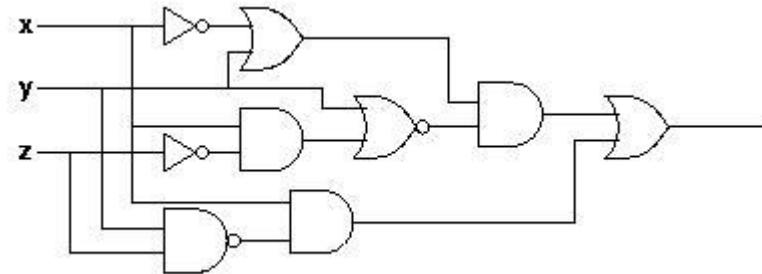


Reference for the reason:
<http://neuralnetworksanddeeplearning.com/chap4.html>

Yes, shallow network can represent any function.

However, using deep structure is more effective.

Analogy



Logic circuits

- Logic circuits consists of **gates**
- **A two layers of logic gates** can represent **any Boolean function.**
- Using multiple layers of logic gates to build some functions are much simpler



less gates needed



- Neural network consists of **neurons**
- **A hidden layer network** can represent **any continuous function.**
- Using multiple layers of neurons to represent some functions are much simpler



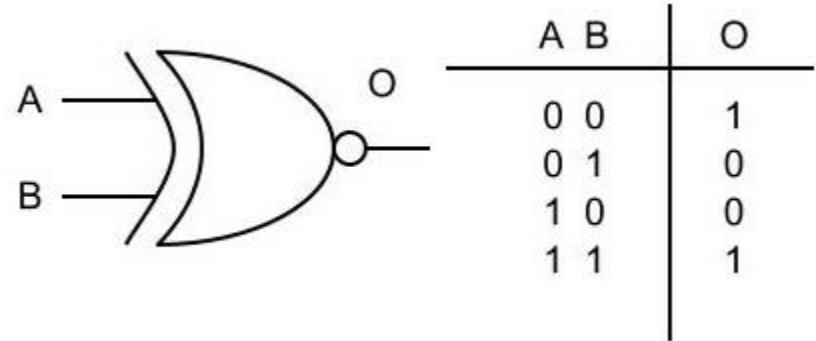
less parameters



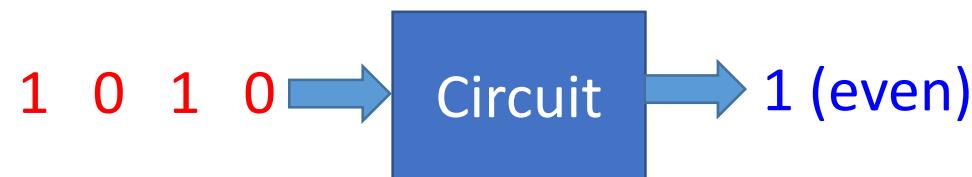
less data?

This page is for EE background.

Analogy

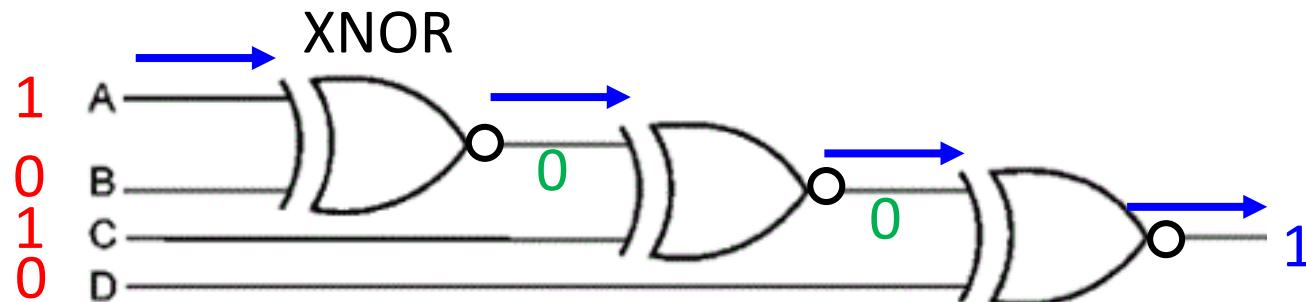


- E.g. parity check



For input sequence
with d bits,

Two-layer circuit
need $O(2^d)$ gates.

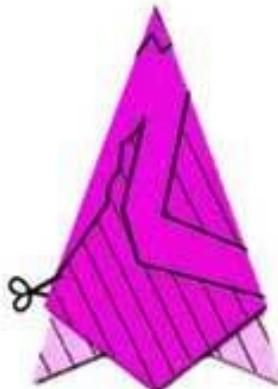


With multiple layers, we need only $O(d)$ gates.

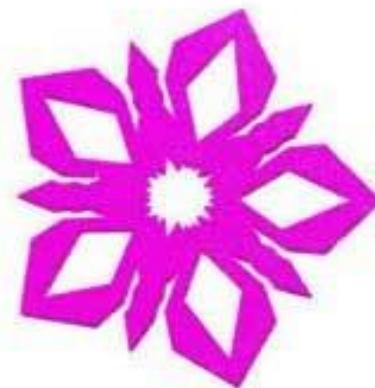
More Analogy



① 画



② 剪



③ 展开, 完成



① 画



② 剪

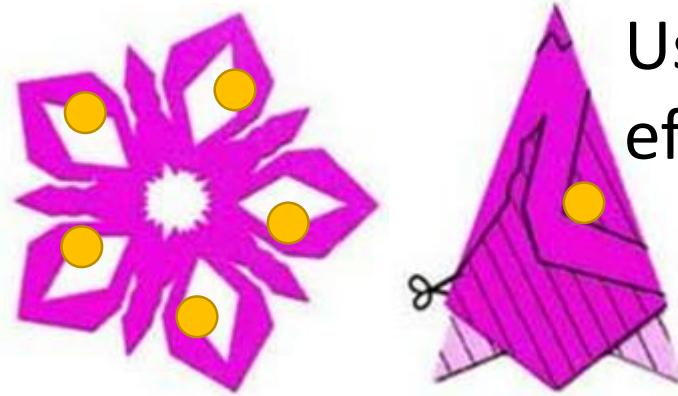


③ 展开, 完成

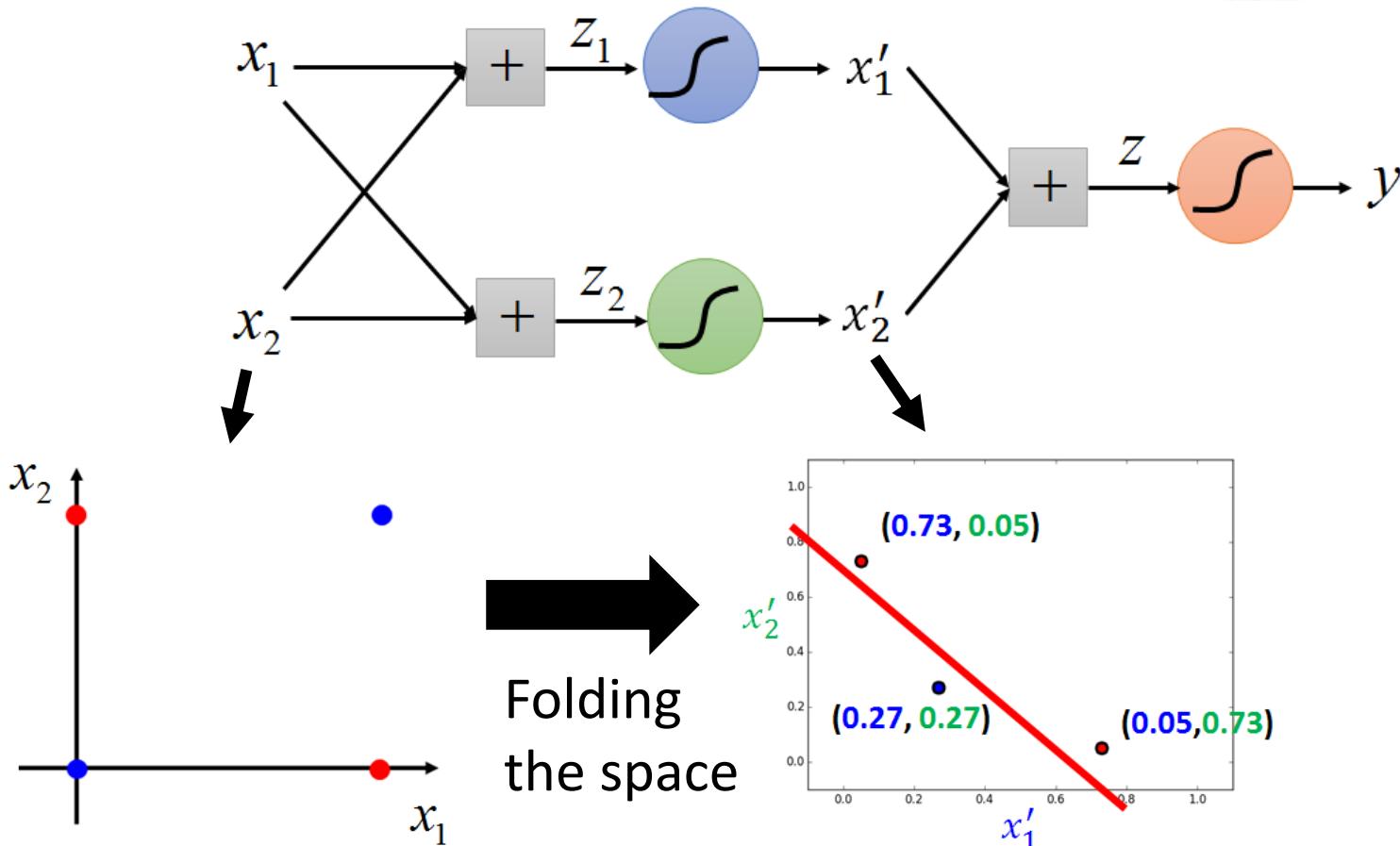
手工制作 www.xuhan.org

· 五角折剪
· 窗花

More Analogy

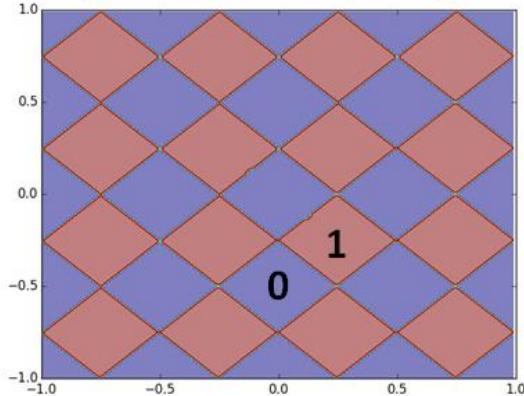


Use data effectively



More Analogy - Experiment

$$f : R^2 \rightarrow \{0,1\}$$

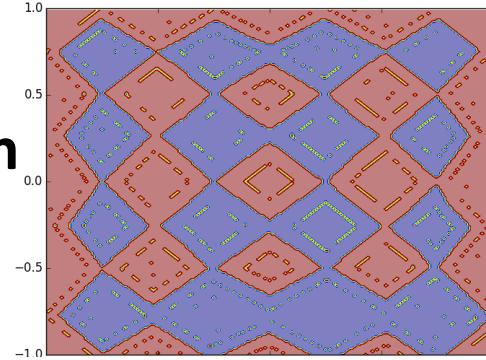


**1 hidden
layer**

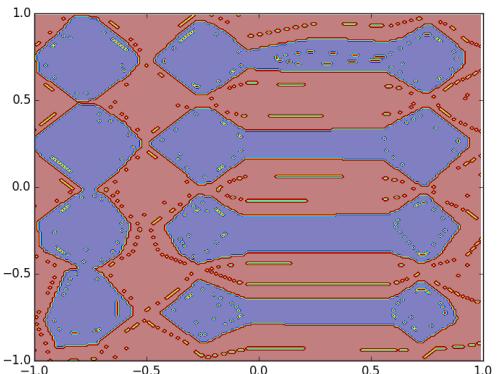
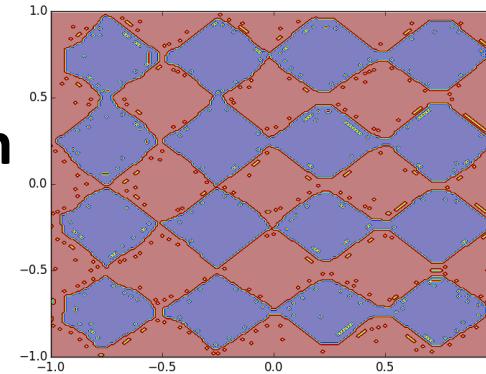
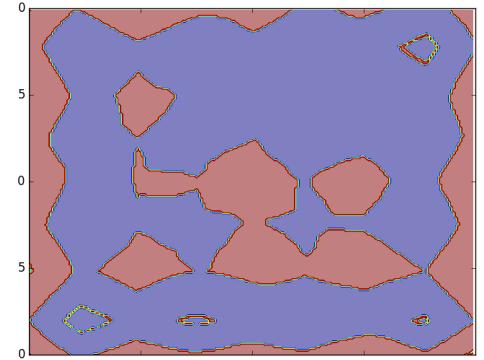
**3 hidden
layers**

Different numbers of training examples

10,000

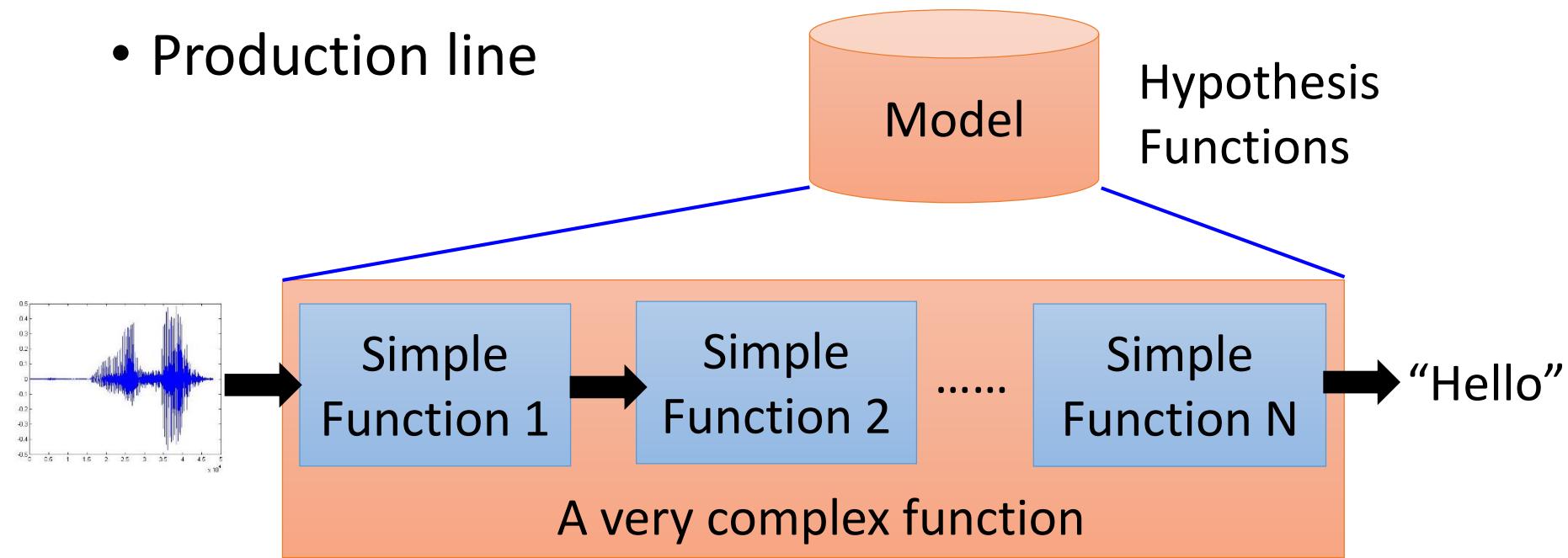


2,000



End-to-end Learning

- Production line



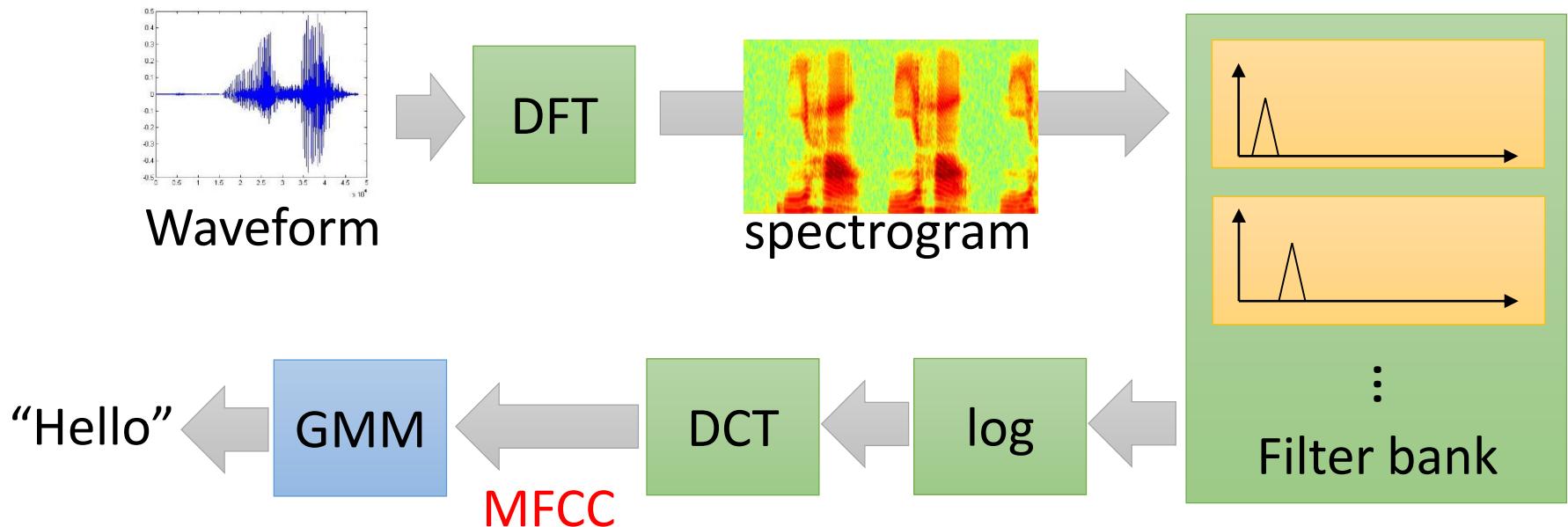
End-to-end training:

What each function should do is learned automatically

End-to-end Learning

- Speech Recognition

- Shallow Approach



Each box is a simple function in the production line:



:hand-crafted

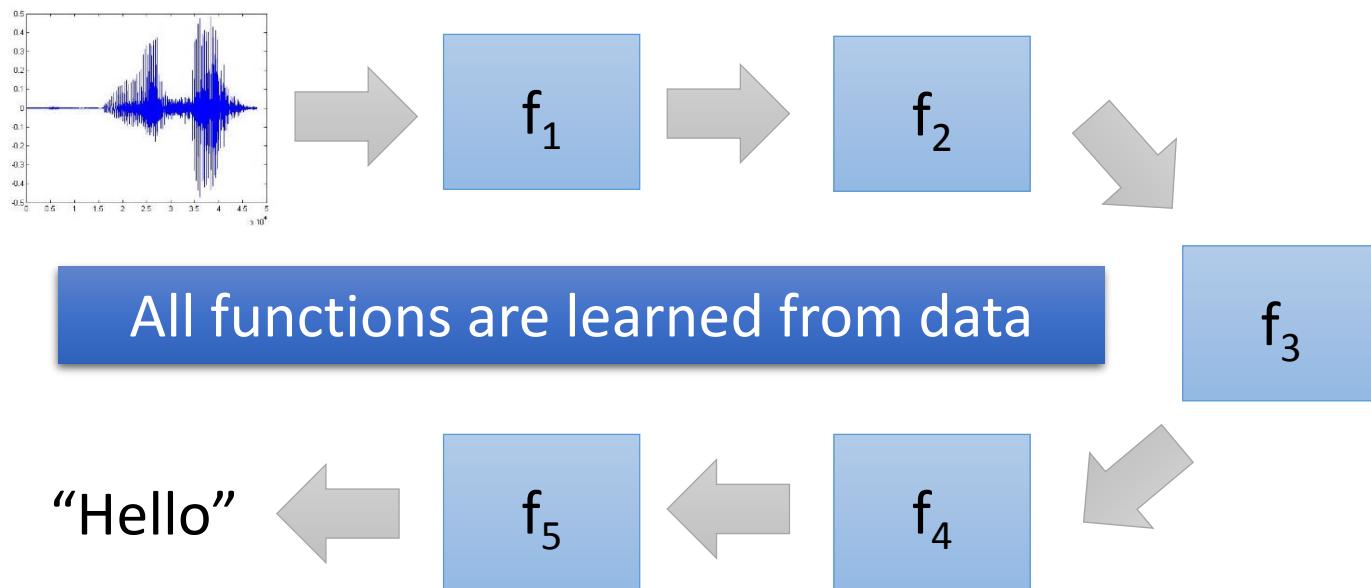


:learned from data

End-to-end Learning

- Speech Recognition

- Deep Learning

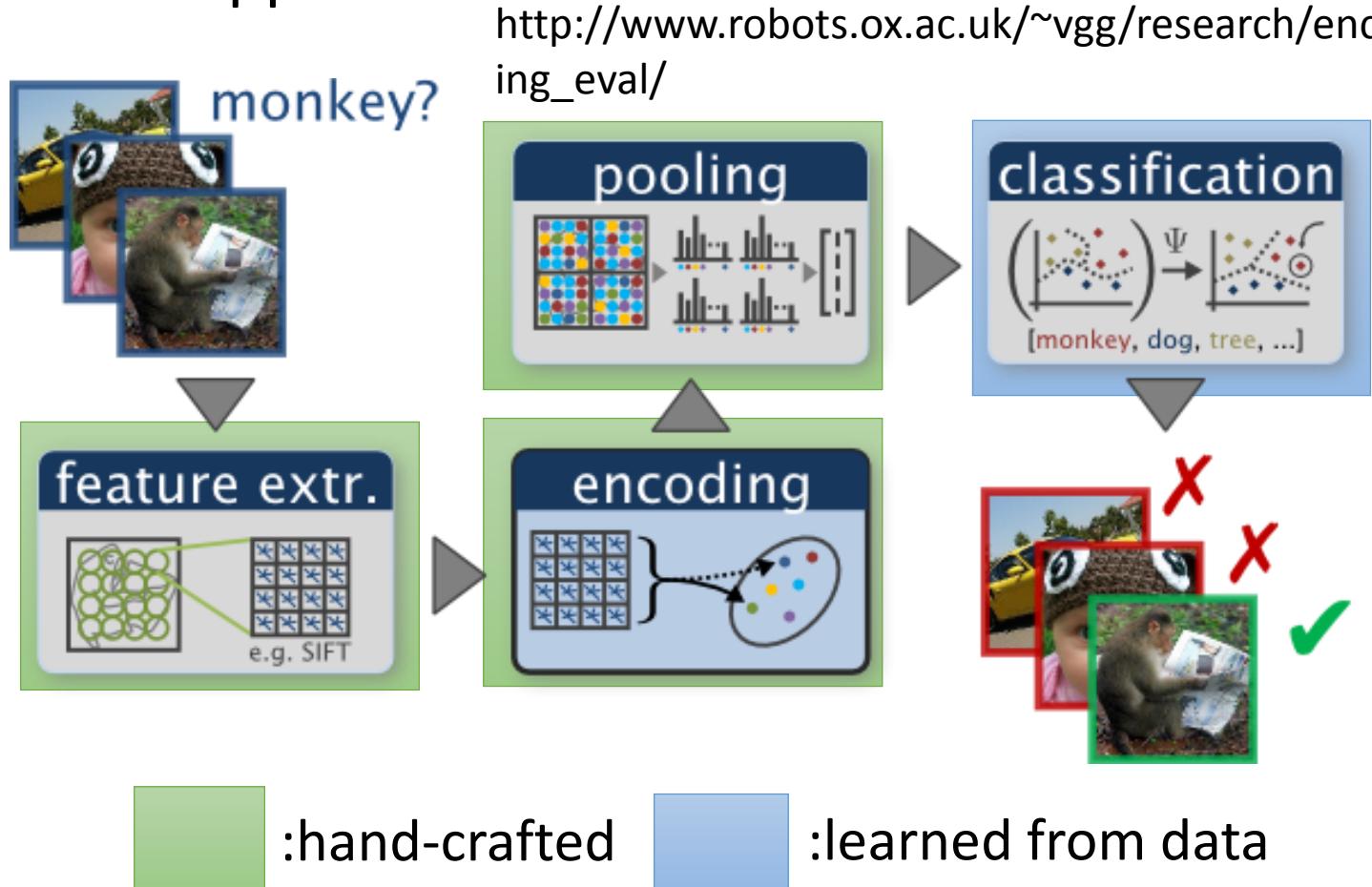


Less engineering labor, but machine learns more

End-to-end Learning

- Image Recognition

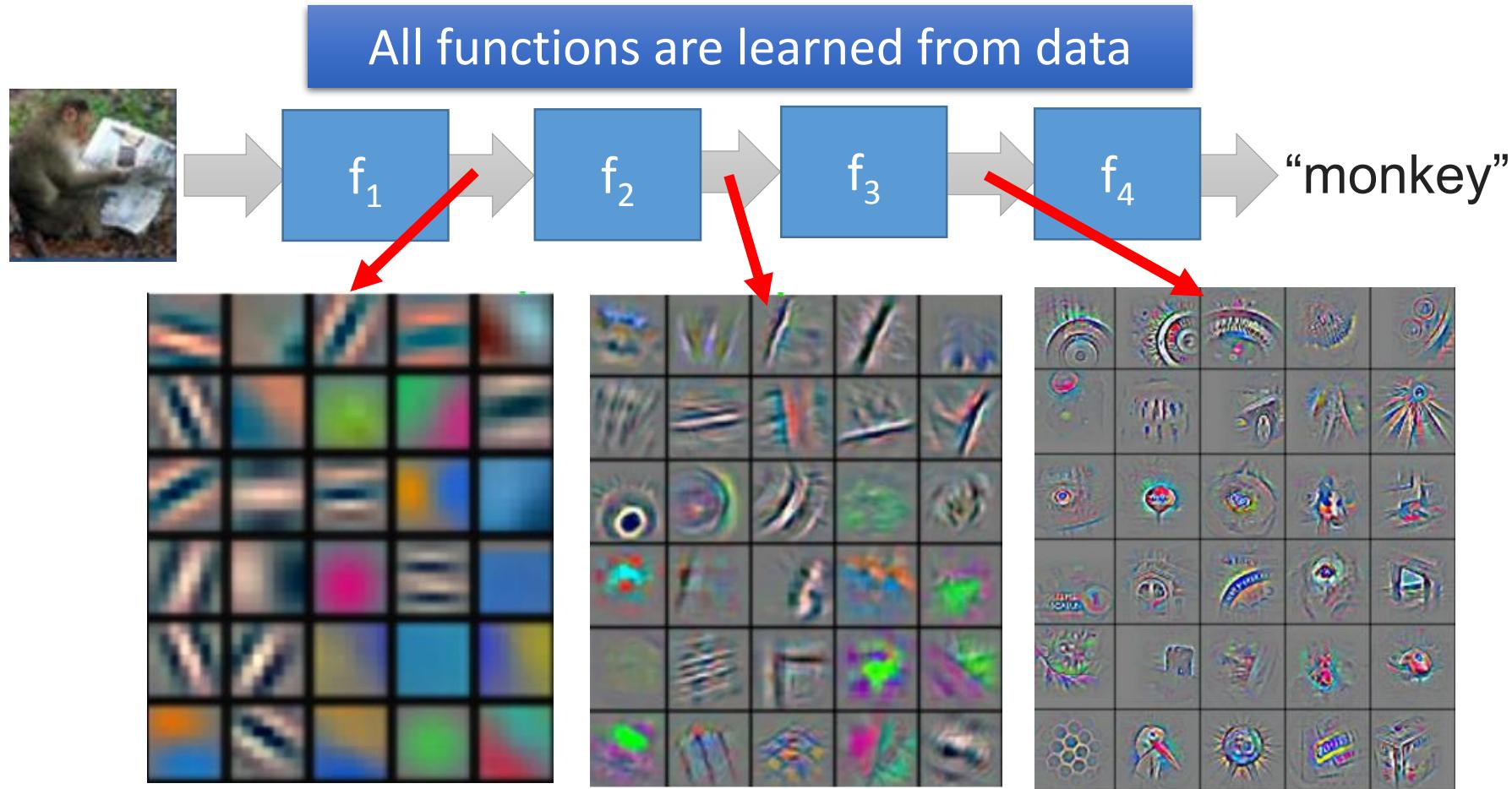
- Shallow Approach



End-to-end Learning

- Image Recognition

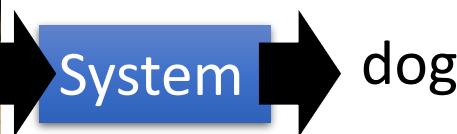
- Deep Learning



Reference: Zeiler, M. D., & Fergus, R. (2014). Visualizing and understanding convolutional networks. In *Computer Vision–ECCV 2014* (pp. 818–833)

Complex Task ...

- Very similar input, different output



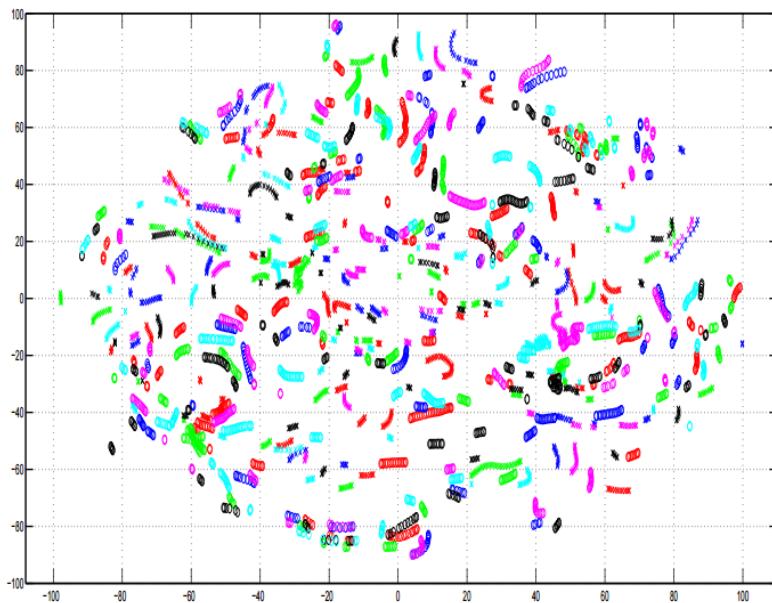
- Very different input, similar output



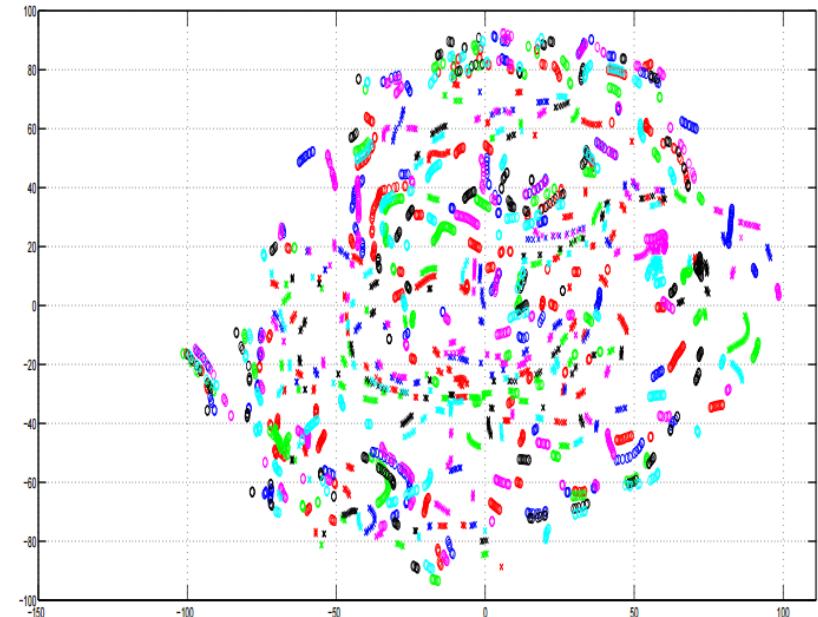
Complex Task ...

A. Mohamed, G. Hinton, and G. Penn, “Understanding how Deep Belief Networks Perform Acoustic Modelling,” in ICASSP, 2012.

- Speech recognition: Speaker normalization is automatically done in DNN



Input Acoustic Feature (MFCC)

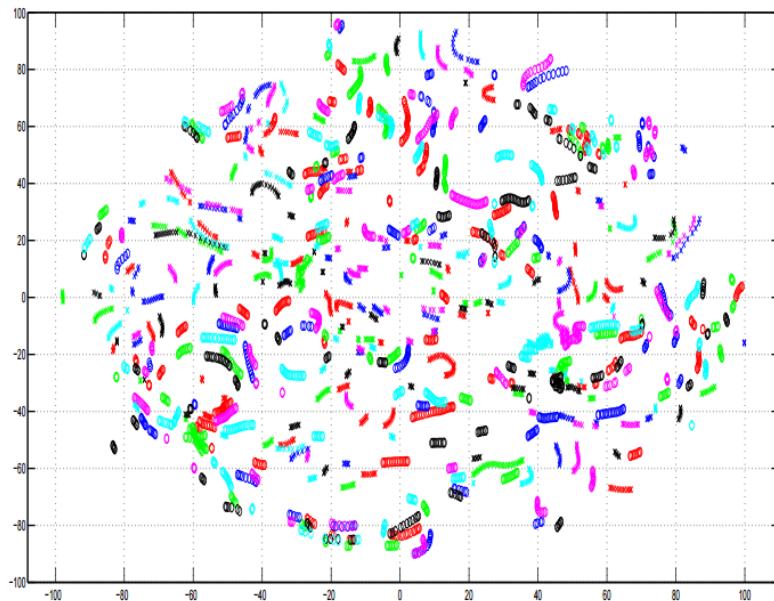


1-st Hidden Layer

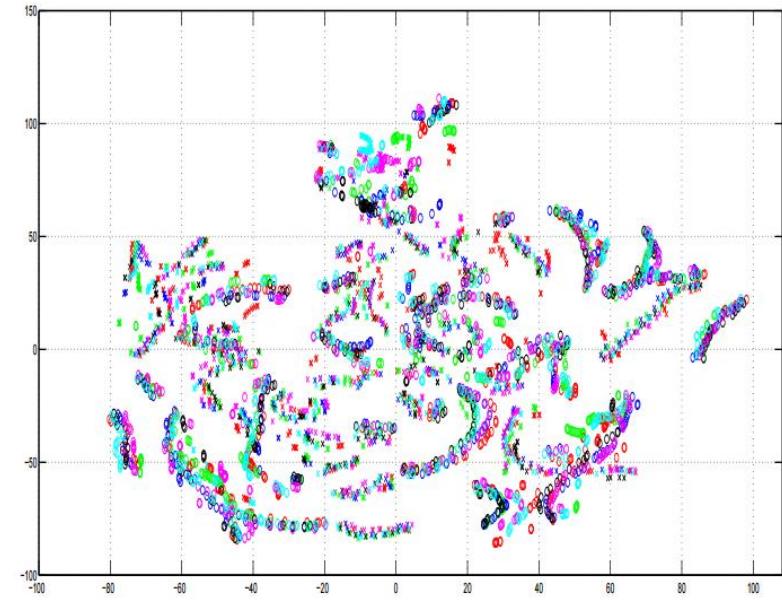
Complex Task ...

A. Mohamed, G. Hinton, and G. Penn, “Understanding how Deep Belief Networks Perform Acoustic Modelling,” in ICASSP, 2012.

- Speech recognition: Speaker normalization is automatically done in DNN

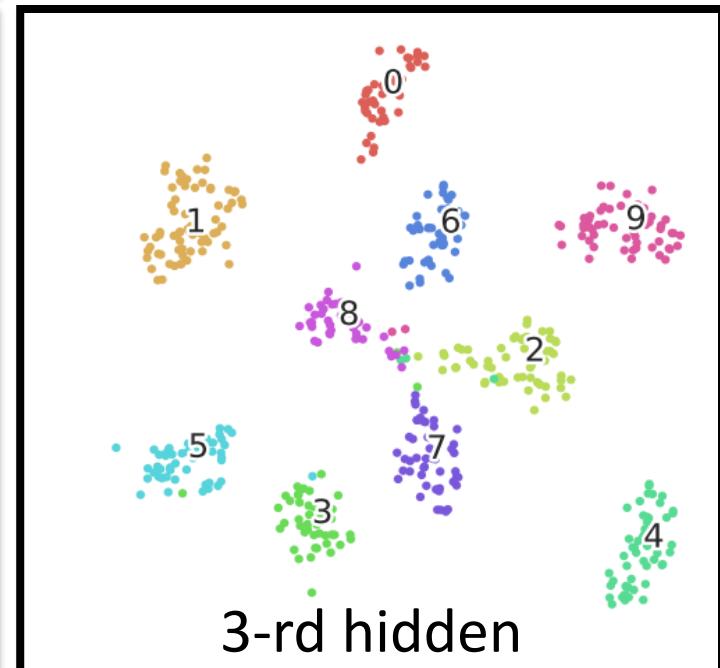
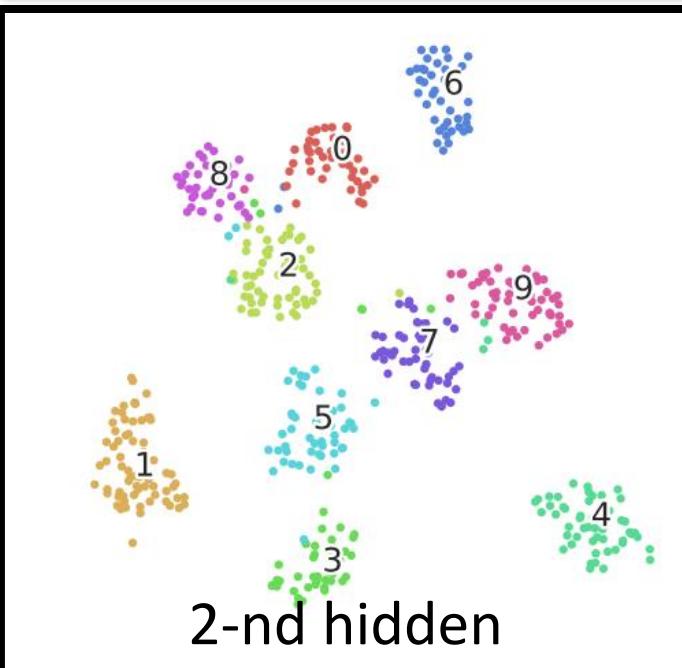
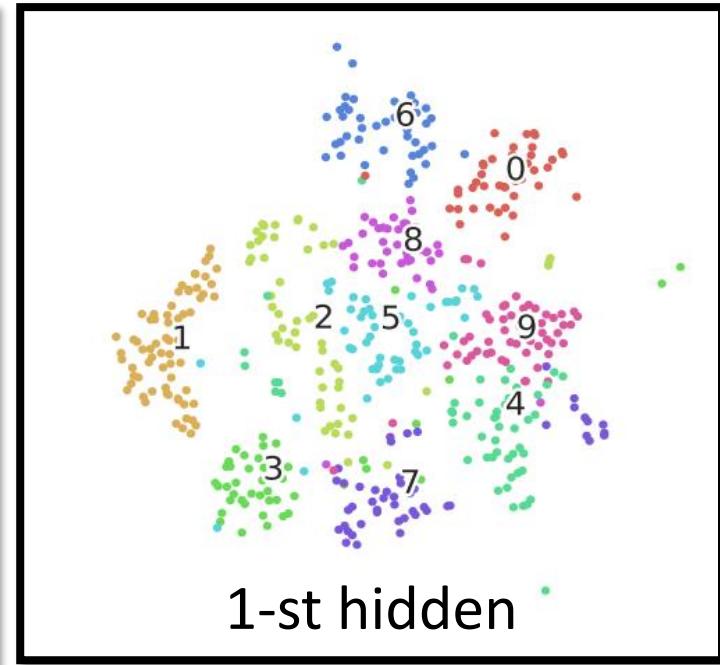
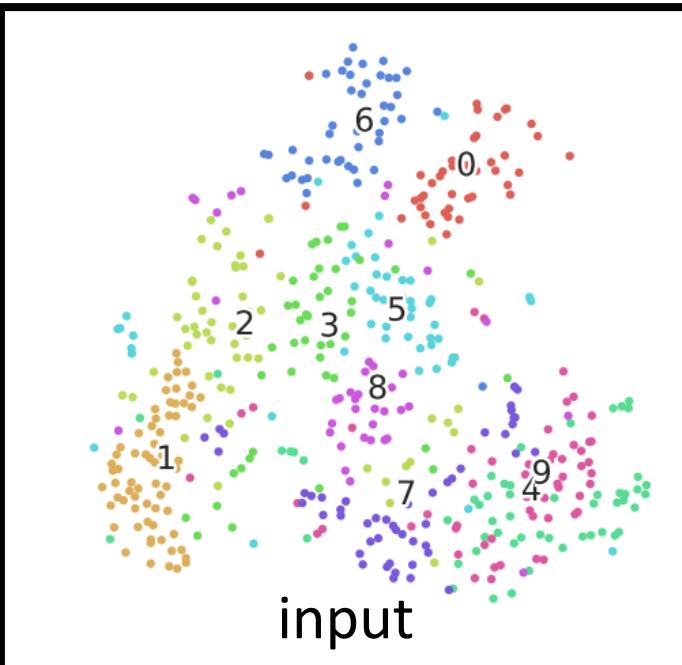


Input Acoustic Feature (MFCC)



8-th Hidden Layer

MNIST



To learn more ...

- Do Deep Nets Really Need To Be Deep? (by Rich Caruana)
- <http://research.microsoft.com/apps/video/default.aspx?id=232373&r=1>

Do deep nets really
need to be deep?

Rich Caruana
Microsoft Research

Lei Jimmy Ba
MSR Intern, University of Toronto

Thanks also to: Gregor Urban, Krzysztof Geras, Samira Kahou, Abdelrahman Mohamed, Jinyu Li, Rui Zhao, Jui-Ting Huang, and Yifan Gong

Yes!

Thank You

Any Questions?

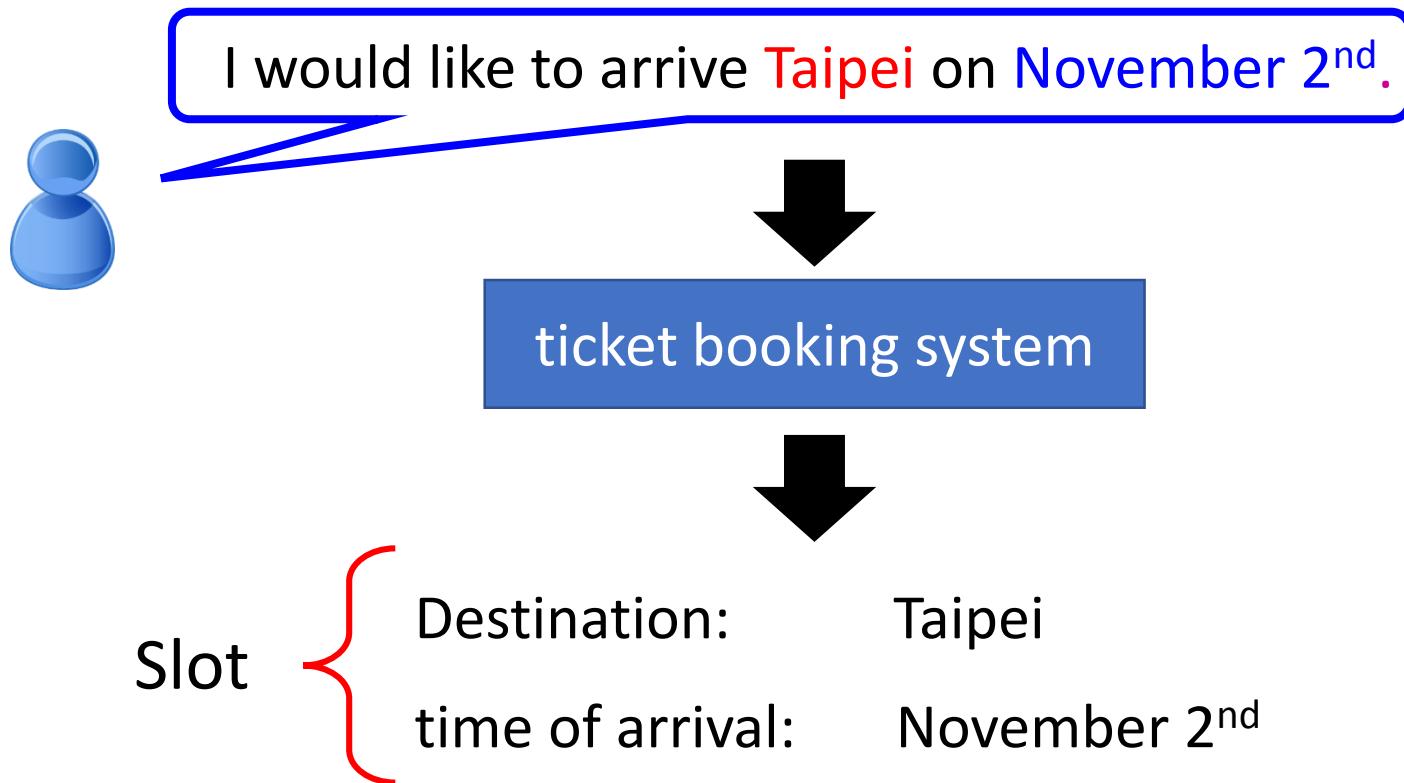
To learn more ...

- Deep Learning: Theoretical Motivations (*Yoshua Bengio*)
 - http://videolectures.net/deeplearning2015_bengio_theoretical_motivations/
- Connections between physics and deep learning
 - <https://www.youtube.com/watch?v=5MdSE-N0bxS>
- Why Deep Learning Works: Perspectives from Theoretical Chemistry
 - <https://www.youtube.com/watch?v=kIbKHIPbxiU>

Recurrent Neural Network (RNN)

Example Application

- Slot Filling



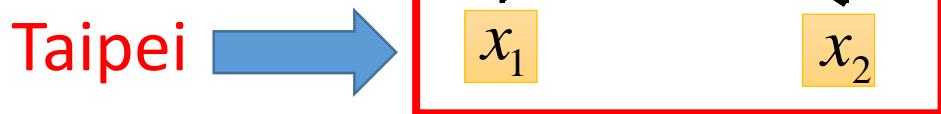
Example Application

Solving slot filling by
Feedforward network?

Input: a word

(Each word is represented
as a vector)

Taipei



1-of-N encoding

How to represent each word as a vector?

1-of-N Encoding lexicon = {apple, bag, cat, dog, elephant}

The vector is lexicon size.

$$\text{apple} = [1 \ 0 \ 0 \ 0 \ 0]$$

Each dimension corresponds
to a word in the lexicon

$$\text{bag} = [0 \ 1 \ 0 \ 0 \ 0]$$

The dimension for the word
is 1, and others are 0

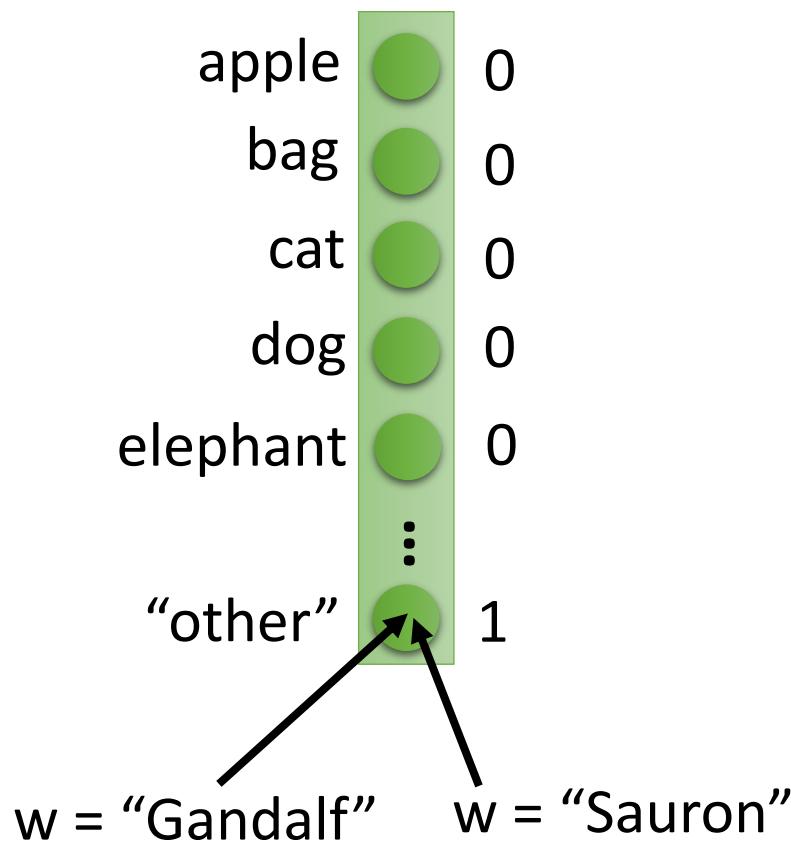
$$\text{cat} = [0 \ 0 \ 1 \ 0 \ 0]$$

$$\text{dog} = [0 \ 0 \ 0 \ 1 \ 0]$$

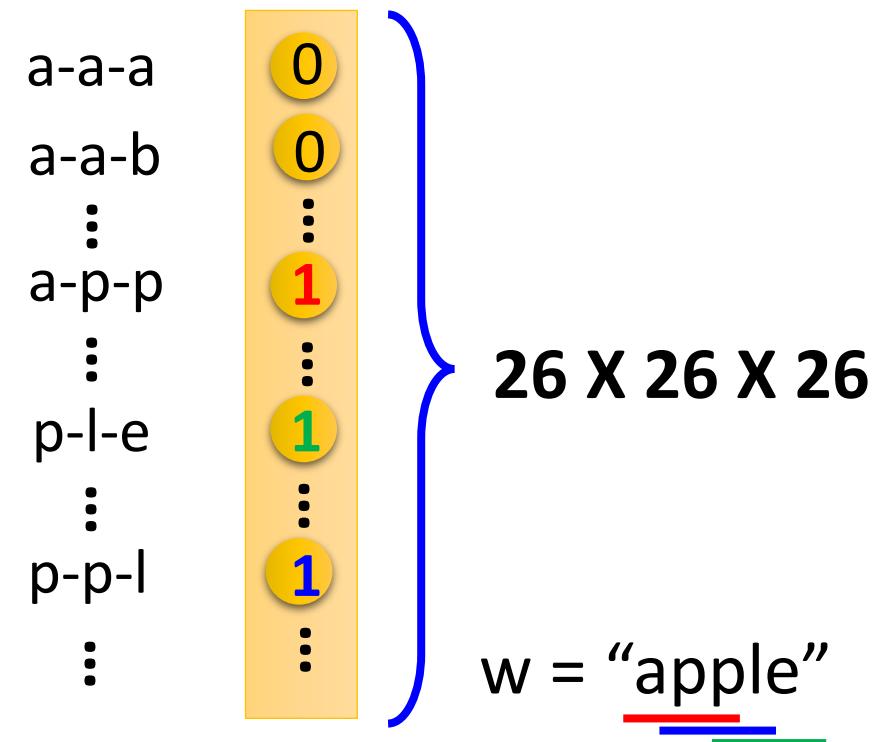
$$\text{elephant} = [0 \ 0 \ 0 \ 0 \ 1]$$

Beyond 1-of-N encoding

Dimension for “Other”



Word hashing



Example Application

Solving slot filling by
Feedforward network?

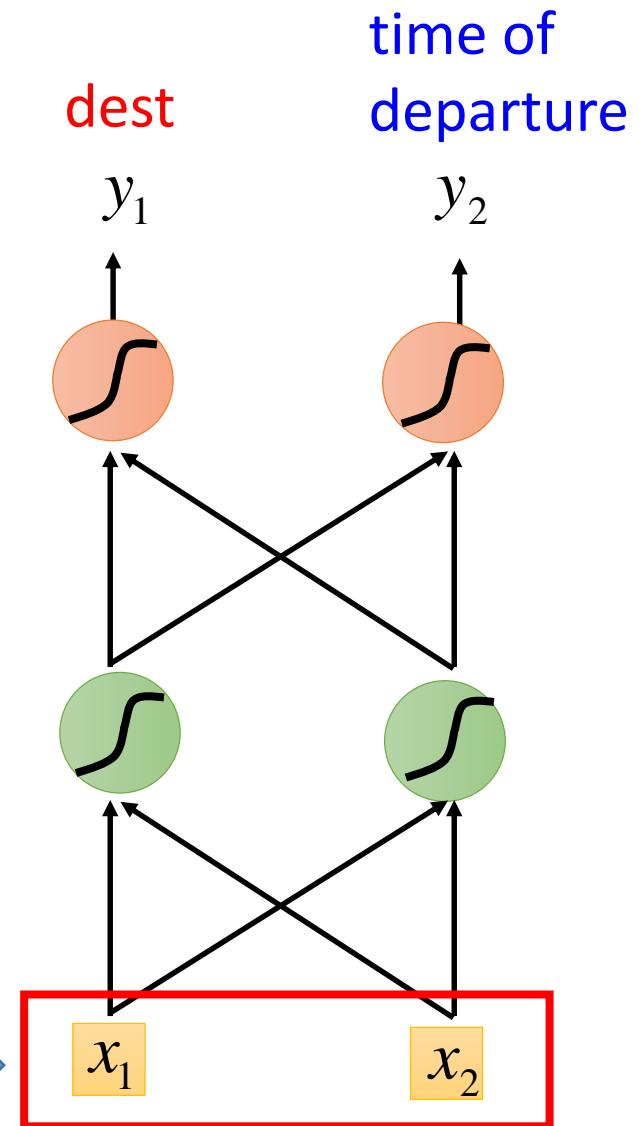
Input: a word

(Each word is represented
as a vector)

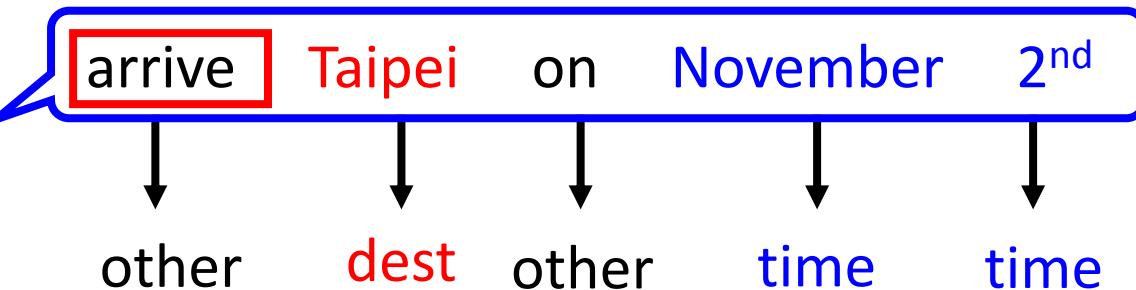
Output:

Probability distribution that
the input word belonging to
the slots

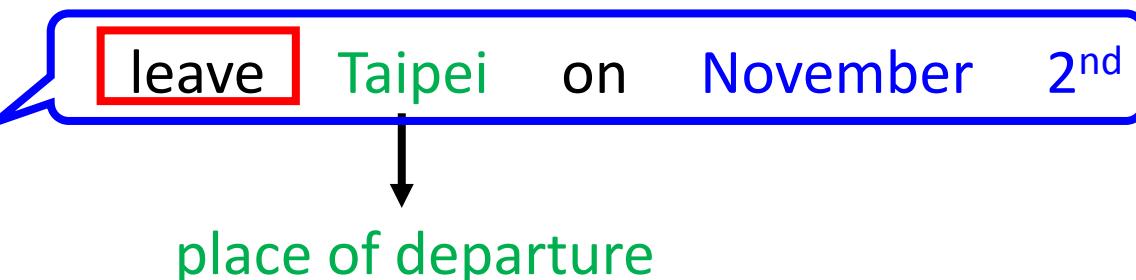
Taipei



Example Application



Problem?



Neural network
needs memory!

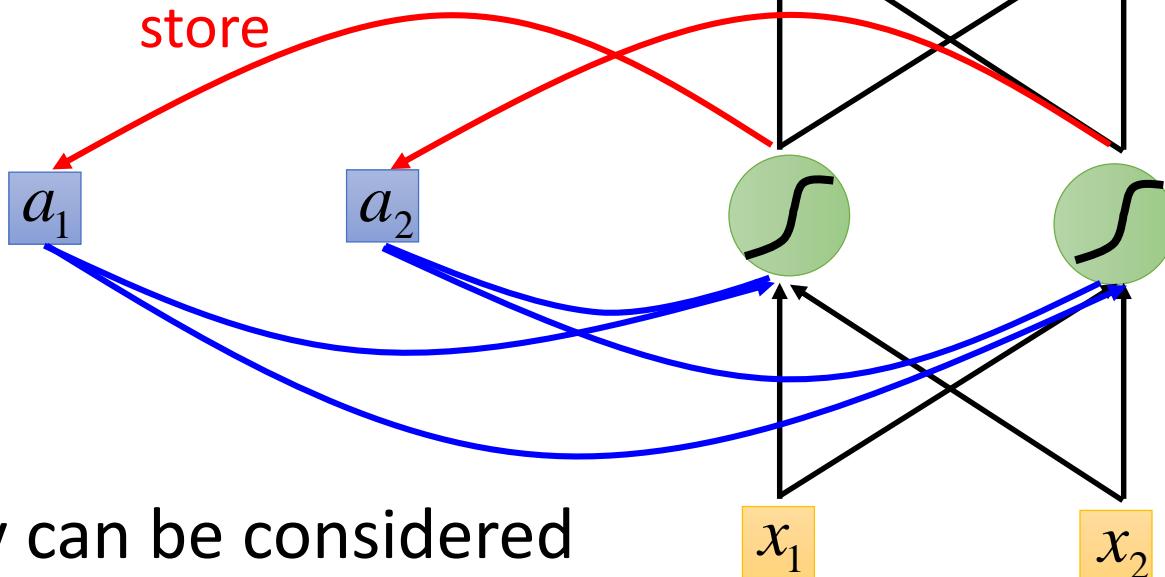
Taipei →



time of
departure

Recurrent Neural Network (RNN)

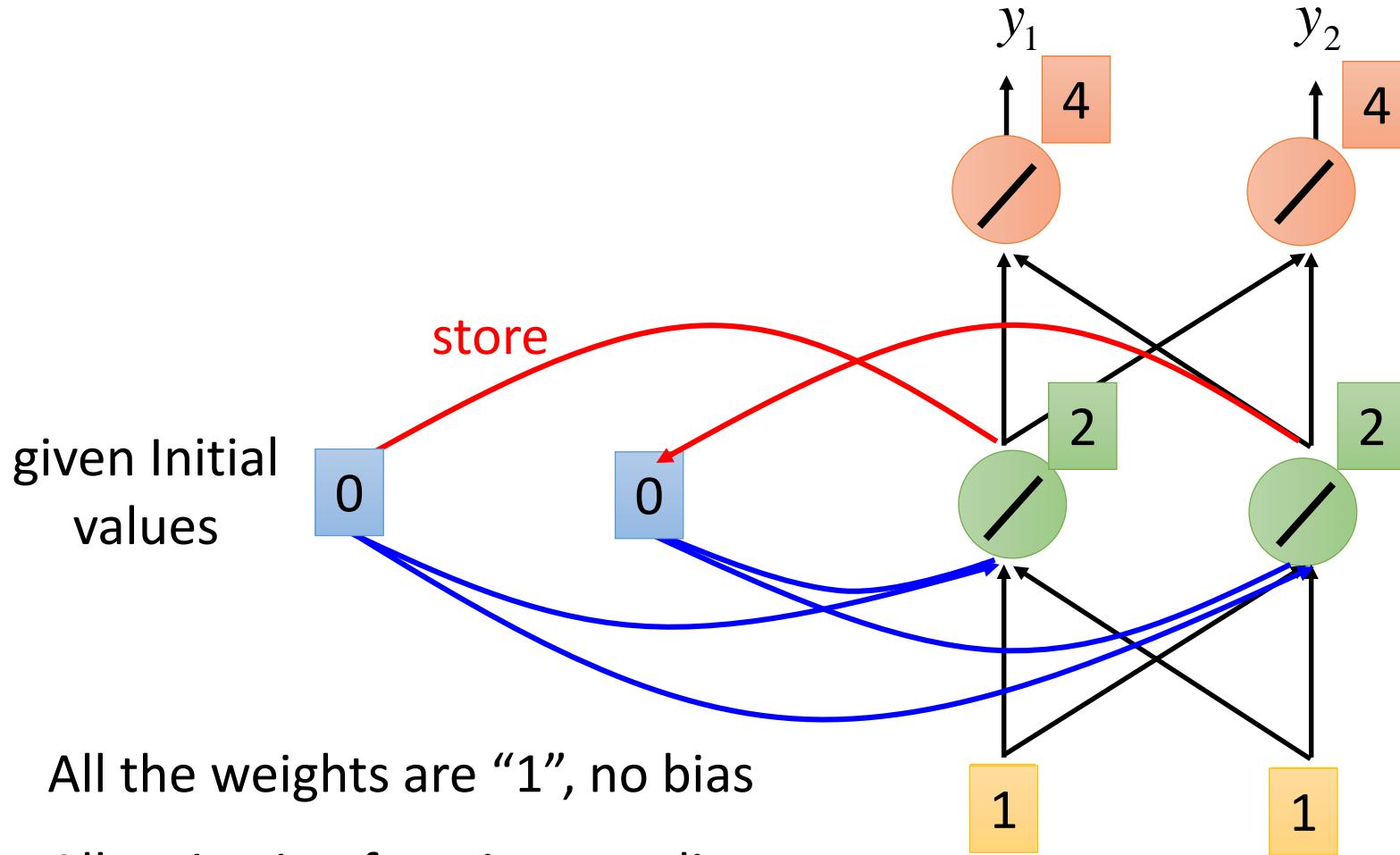
The output of hidden layer
are stored in the memory.



Memory can be considered
as another input.

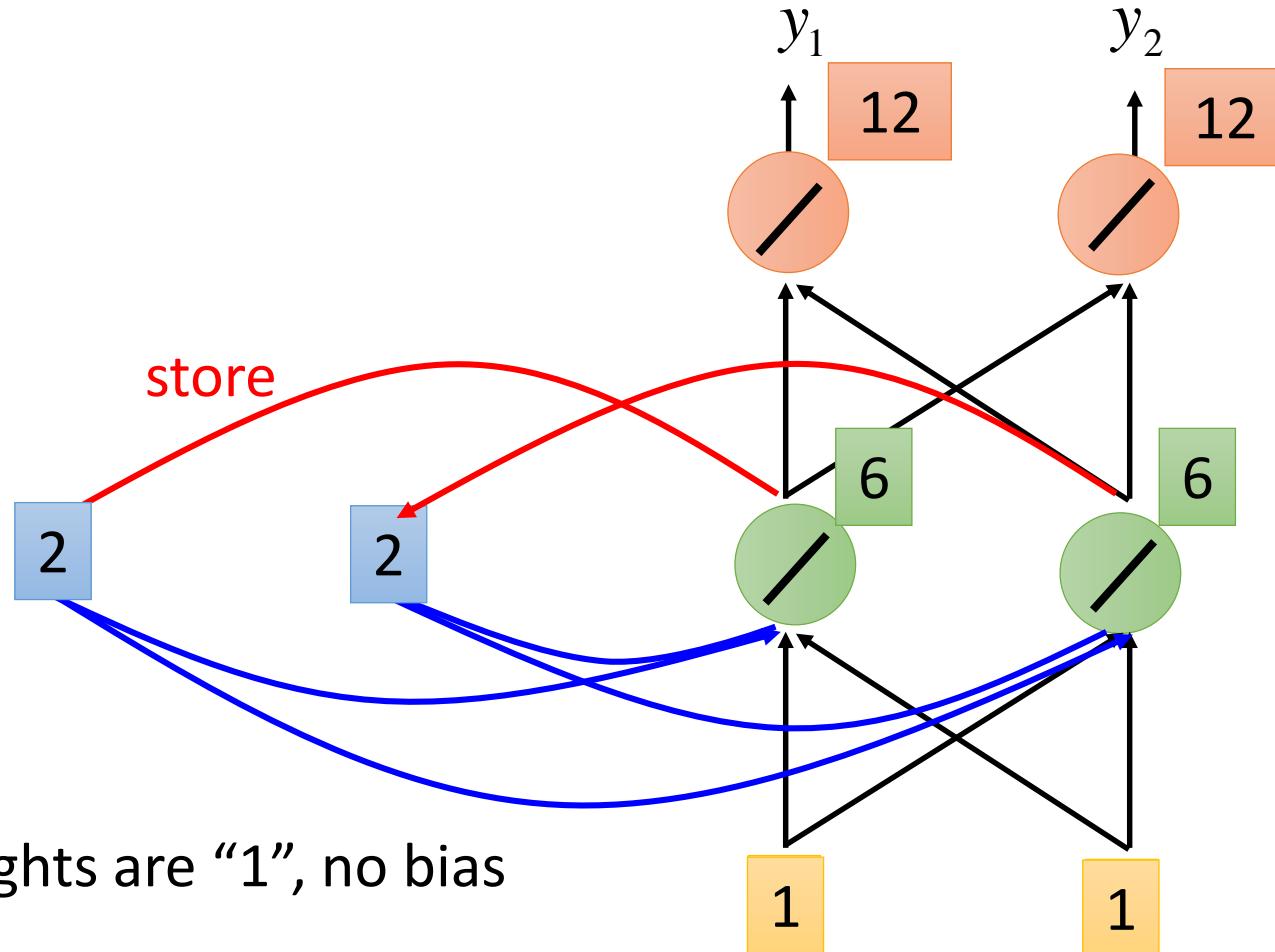
Example

Input sequence: $\begin{bmatrix} 1 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} \begin{bmatrix} 2 \\ 2 \end{bmatrix} \dots \dots$
output sequence: $\begin{bmatrix} 4 \\ 4 \end{bmatrix}$



Example

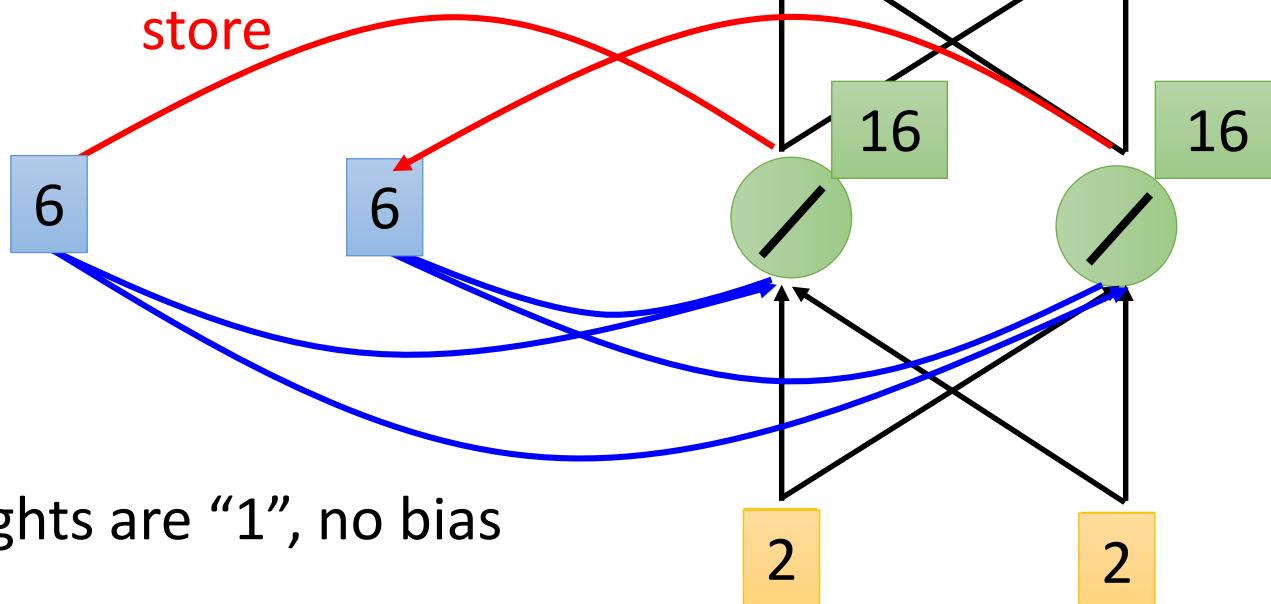
Input sequence: $\begin{bmatrix} 1 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} \begin{bmatrix} 2 \\ 2 \end{bmatrix} \dots \dots$
output sequence: $\begin{bmatrix} 4 \\ 4 \end{bmatrix} \begin{bmatrix} 12 \\ 12 \end{bmatrix}$



Example

Input sequence: $\begin{bmatrix} 1 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} \begin{bmatrix} 2 \\ 2 \end{bmatrix} \dots \dots$
output sequence: $\begin{bmatrix} 4 \\ 4 \end{bmatrix} \begin{bmatrix} 12 \\ 12 \end{bmatrix} \begin{bmatrix} 32 \\ 32 \end{bmatrix}$

Changing the sequence
order will change the output.



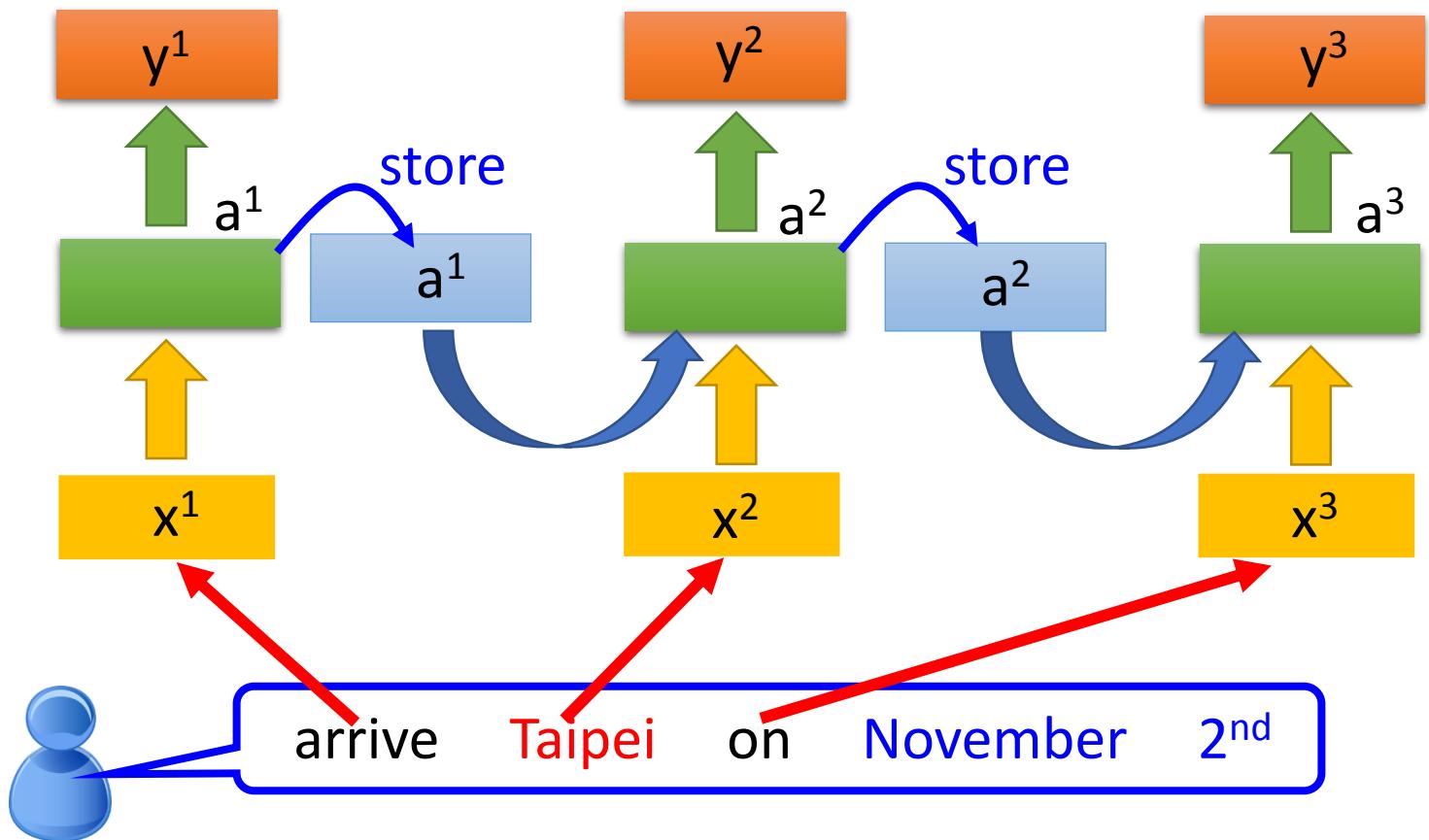
RNN

The same network is used again and again.

Probability of
“arrive” in each slot

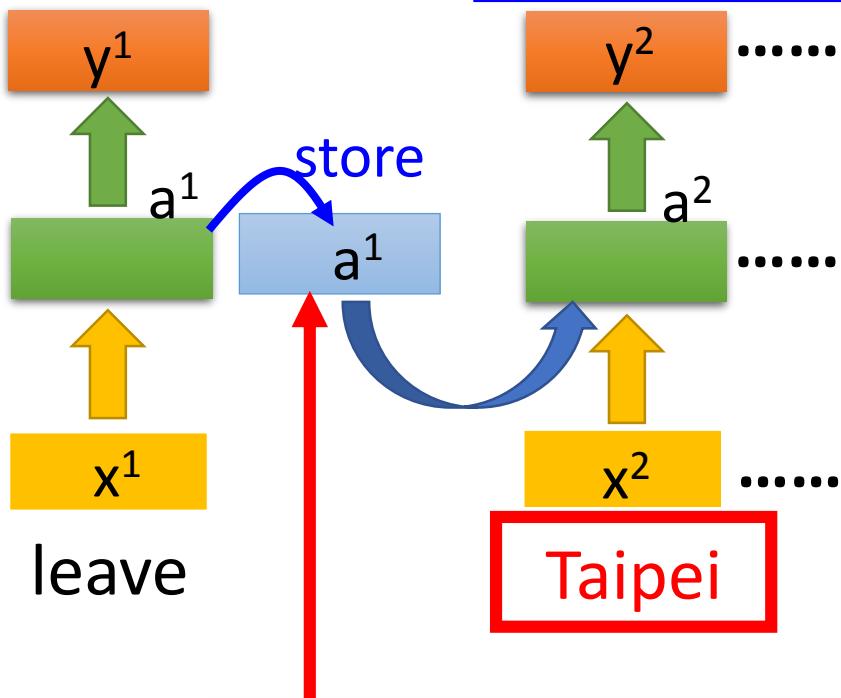
Probability of
“Taipei” in each slot

Probability of
“on” in each slot



RNN

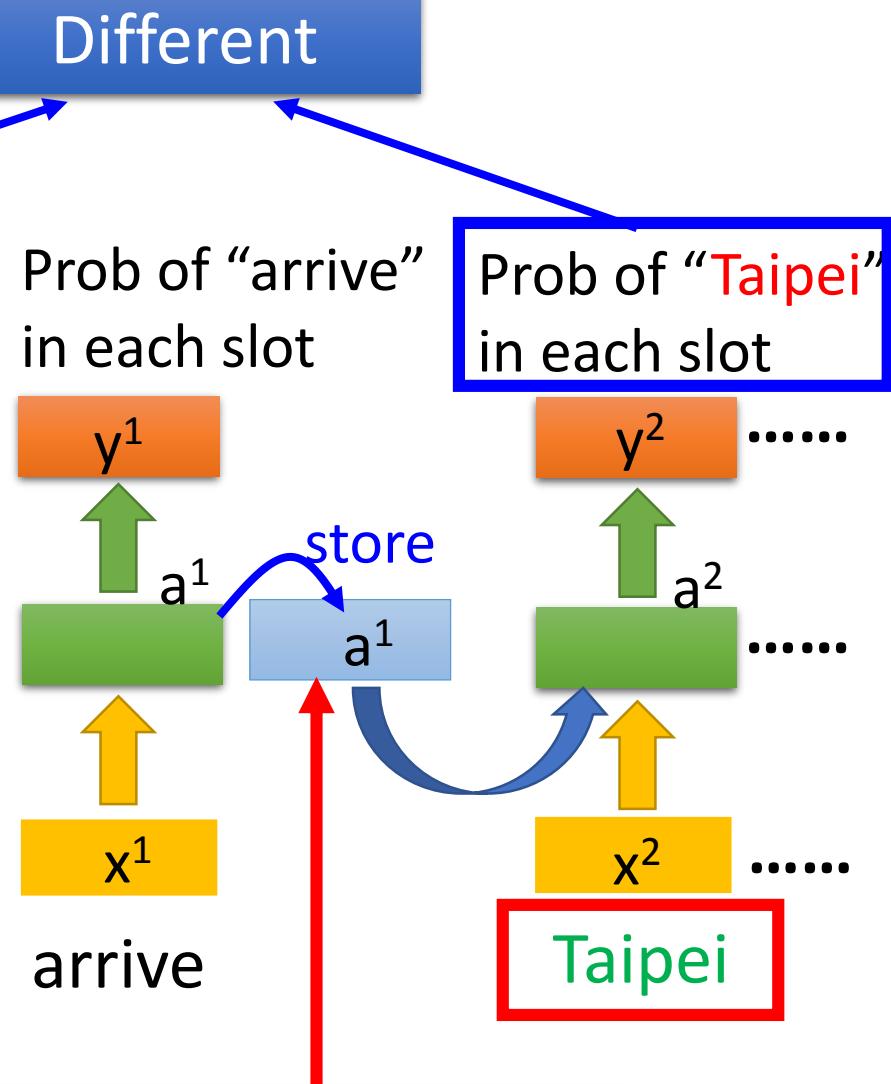
Prob of “leave”
in each slot



Prob of “Taipei”
in each slot

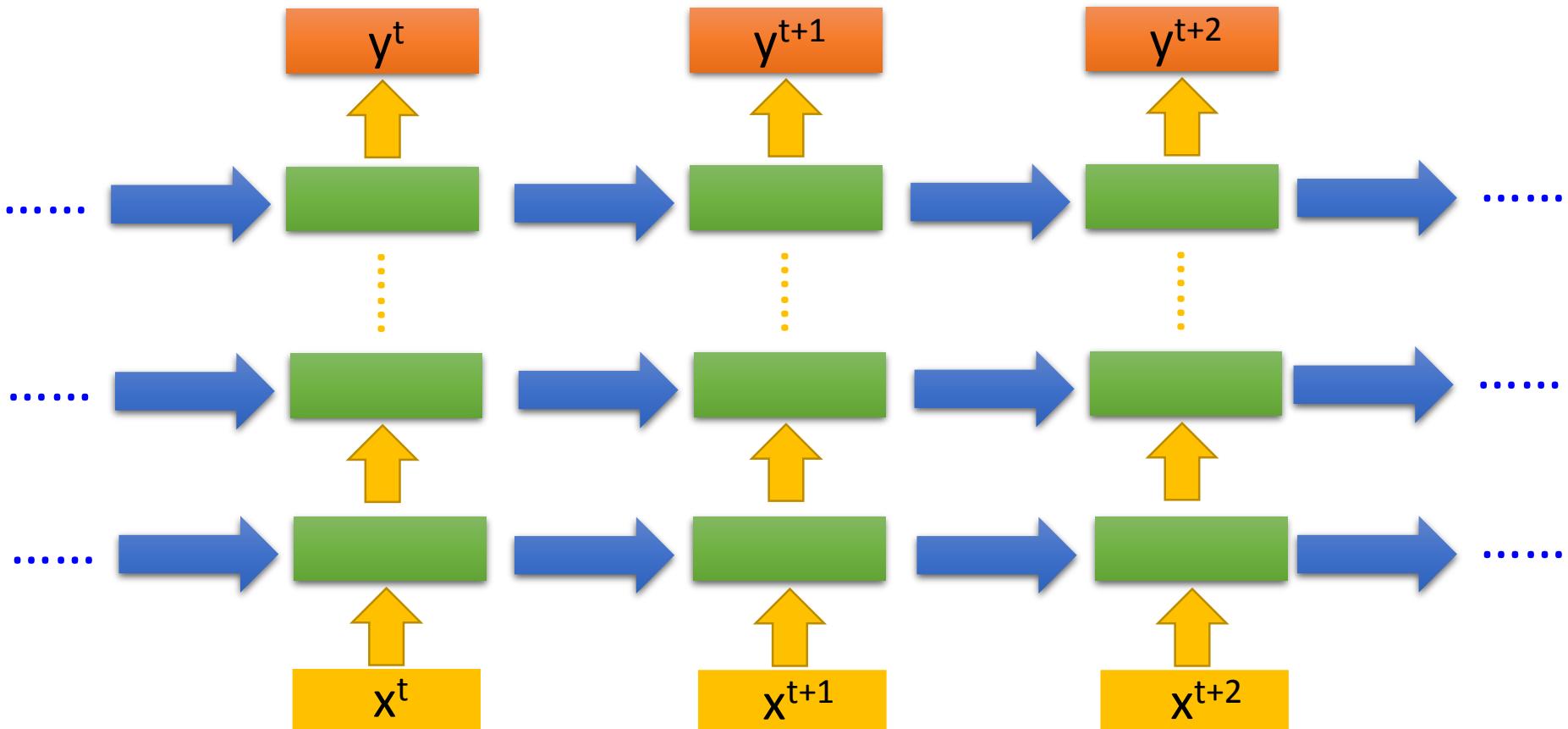
Different

Prob of “arrive”
in each slot



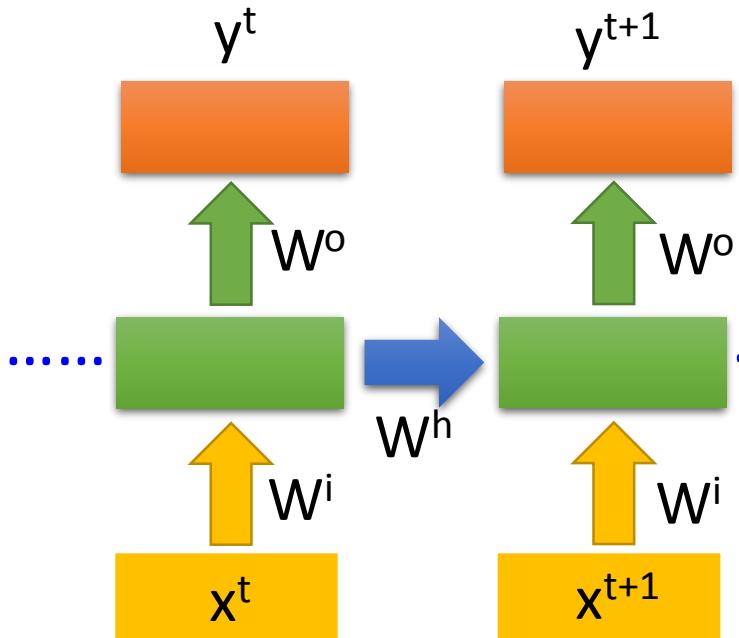
The values stored in the memory is different.

Of course it can be deep ...

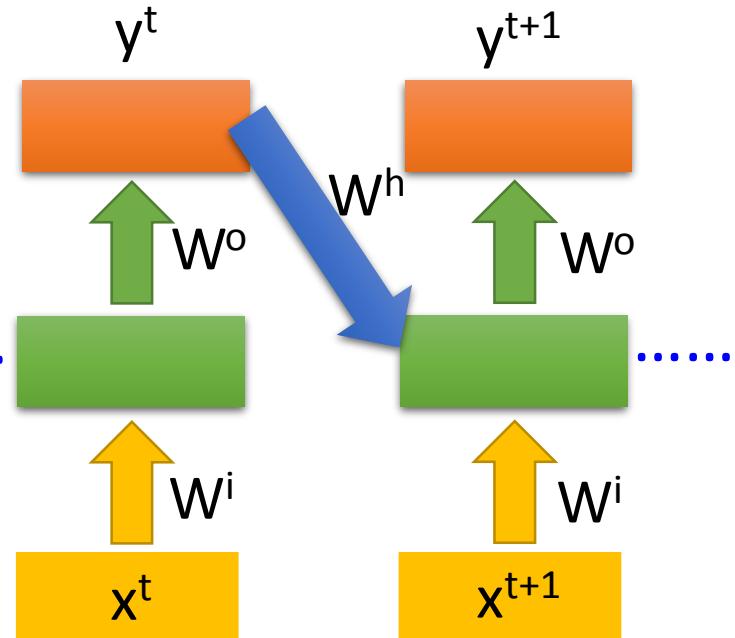


Elman Network & Jordan Network

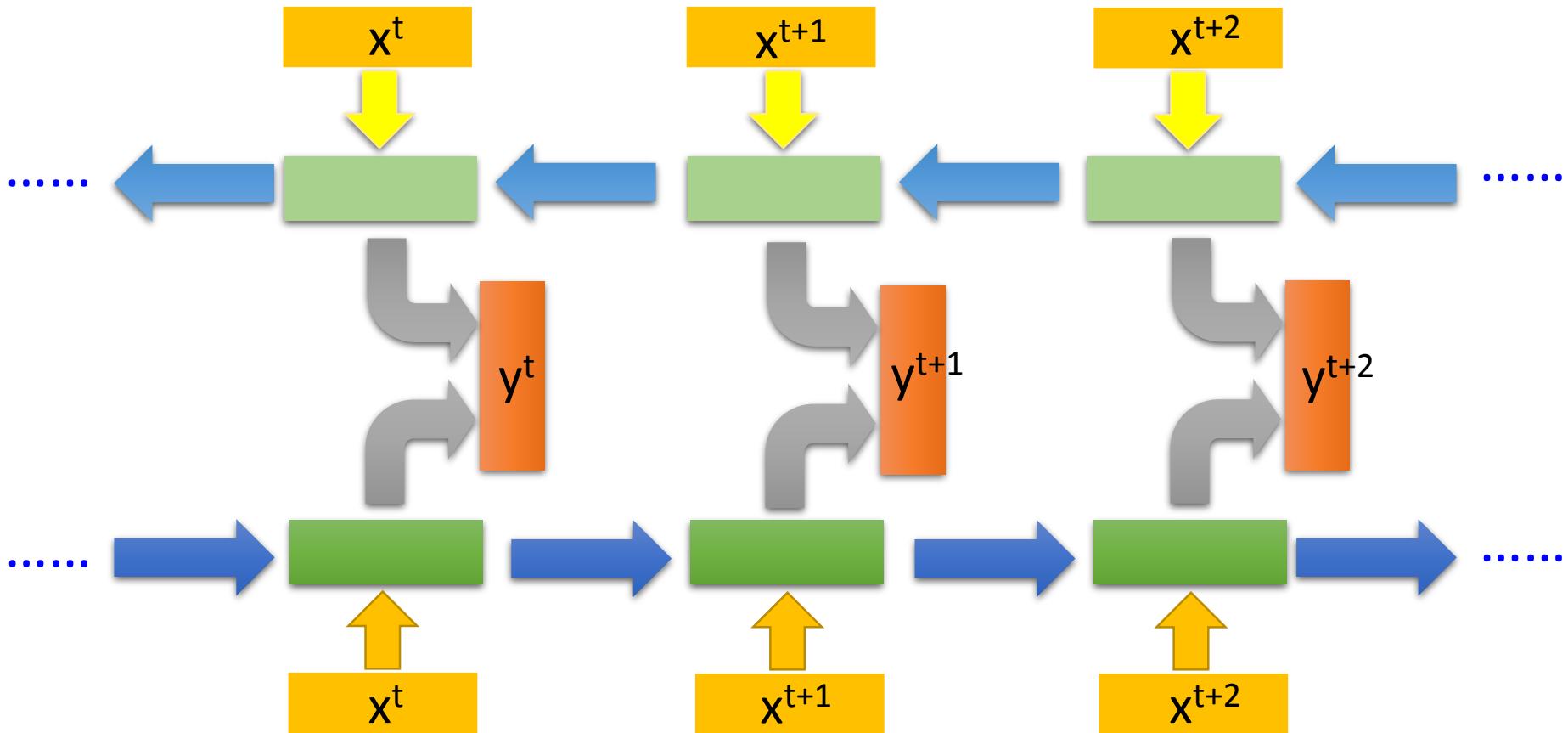
Elman Network



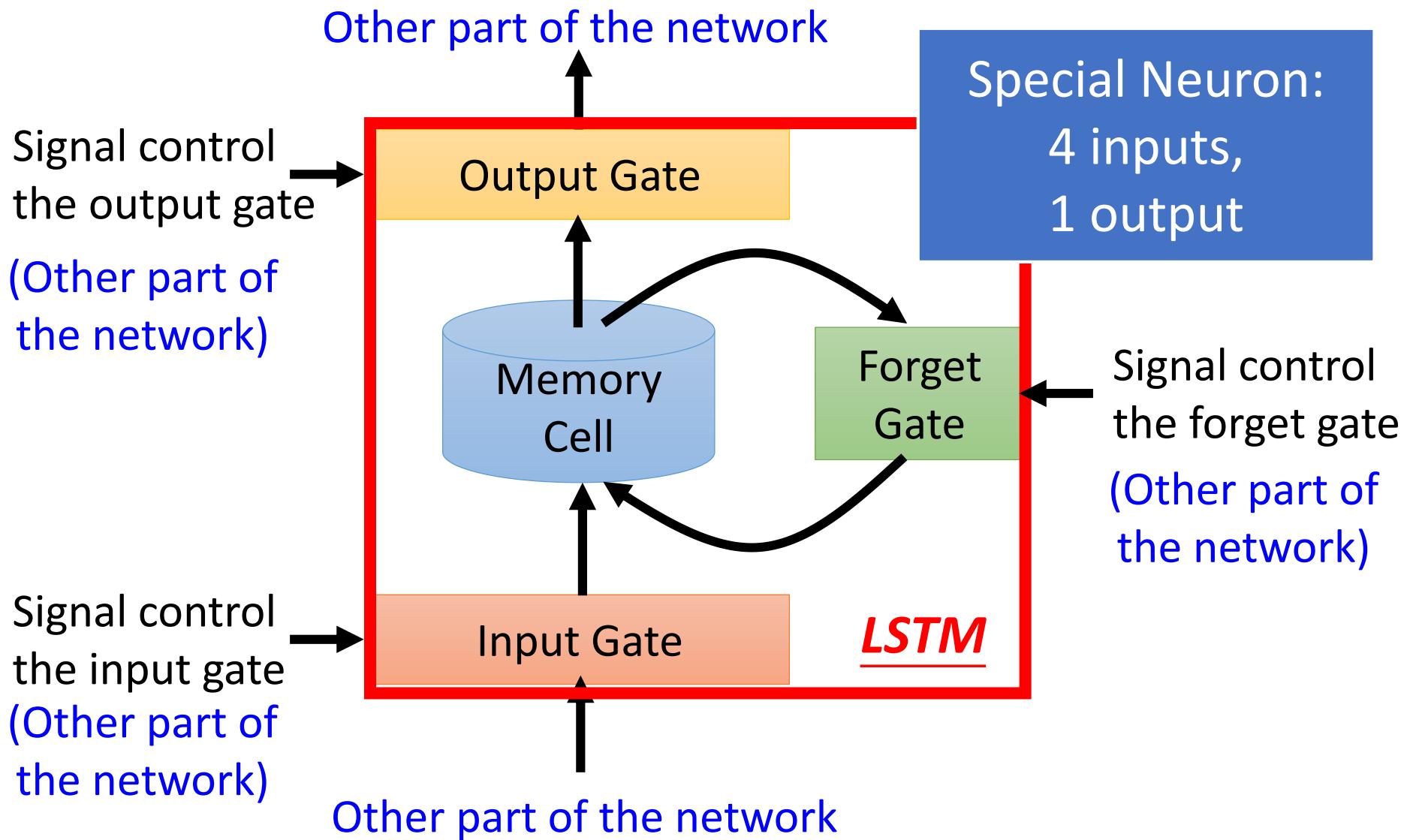
Jordan Network

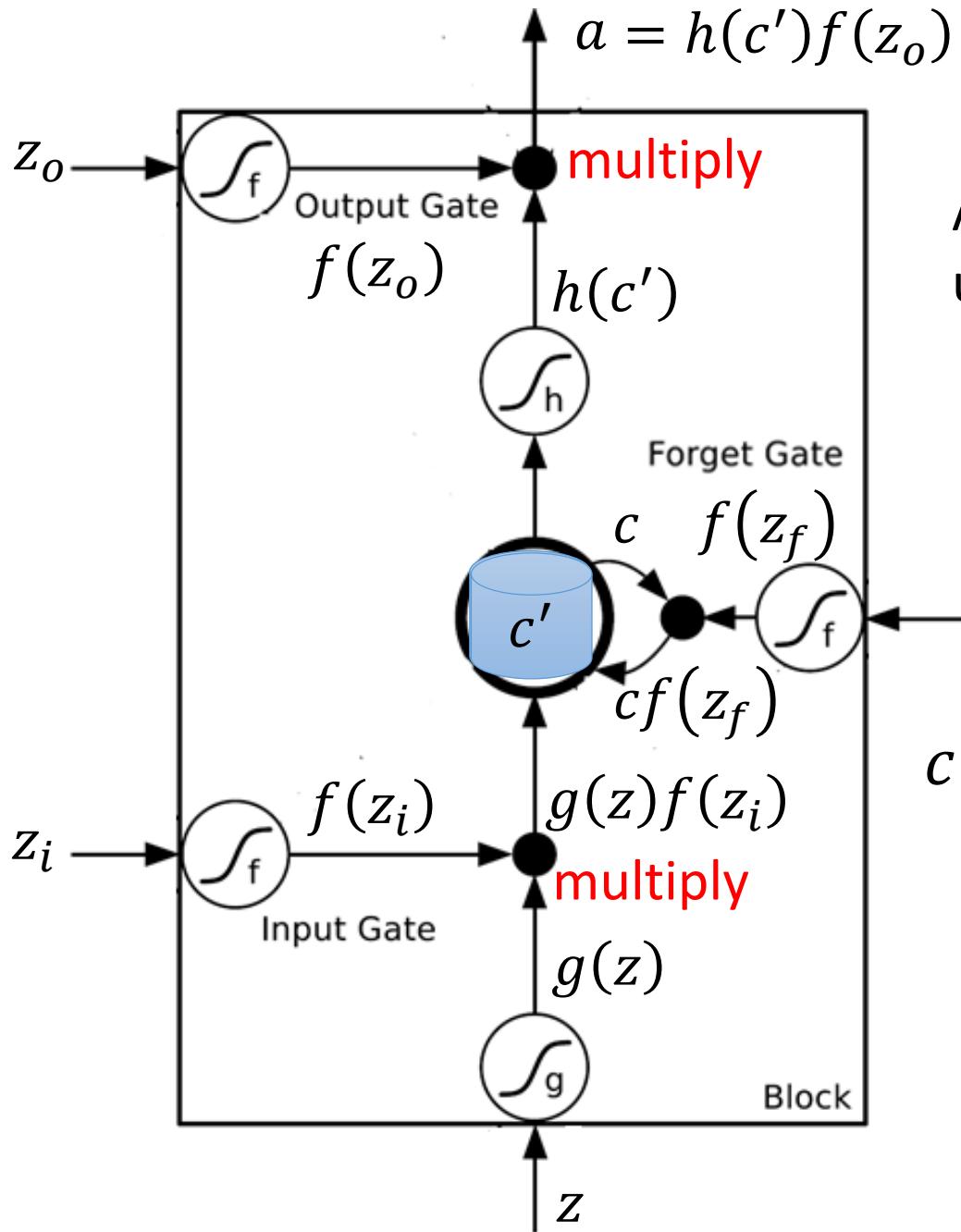


Bidirectional RNN



Long Short-term Memory (LSTM)





Activation function f is usually a sigmoid function

Between 0 and 1

Mimic open and close gate

$$c' = g(z)f(z_i) + cf(z_f)$$

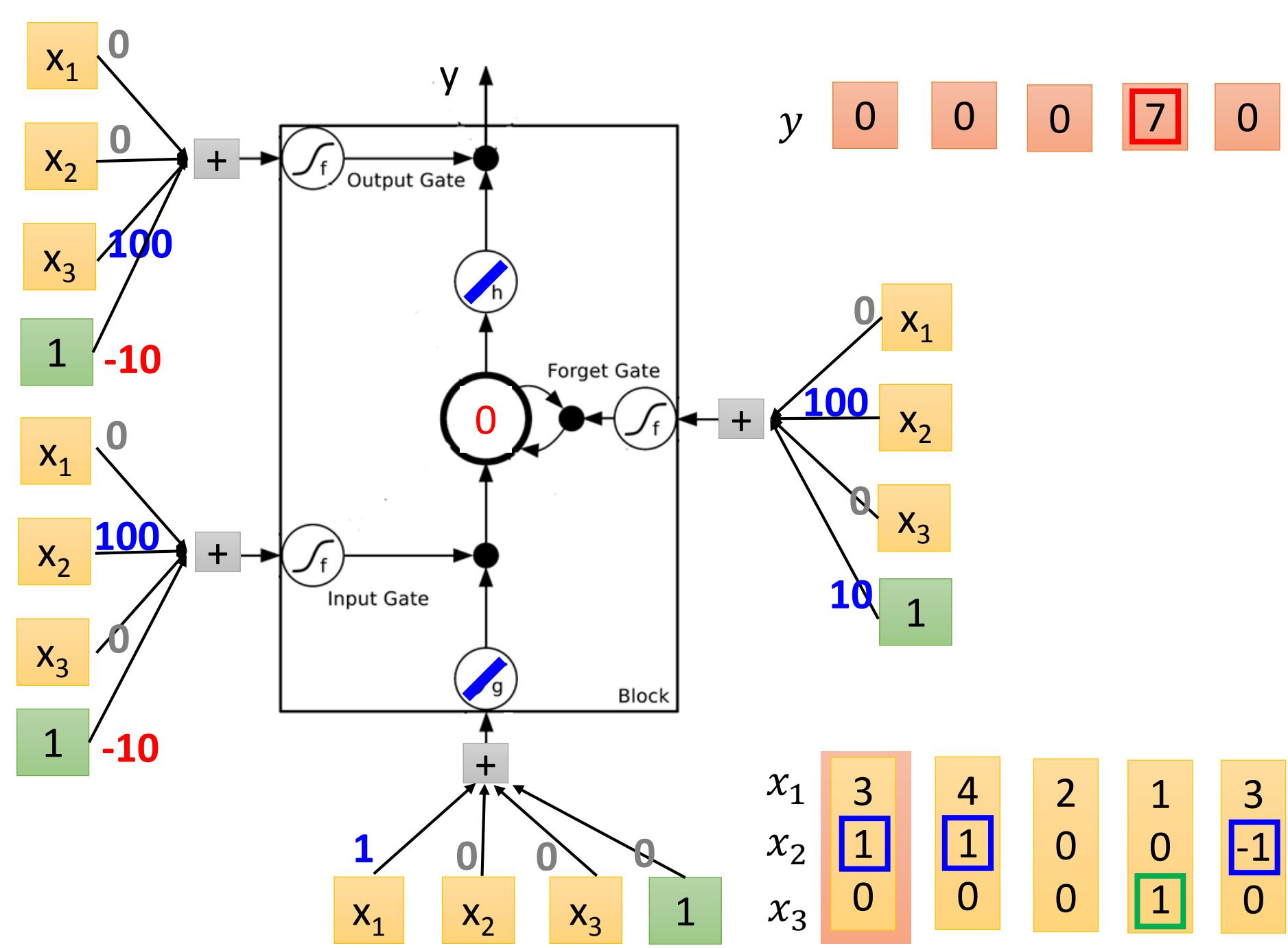
LSTM - Example

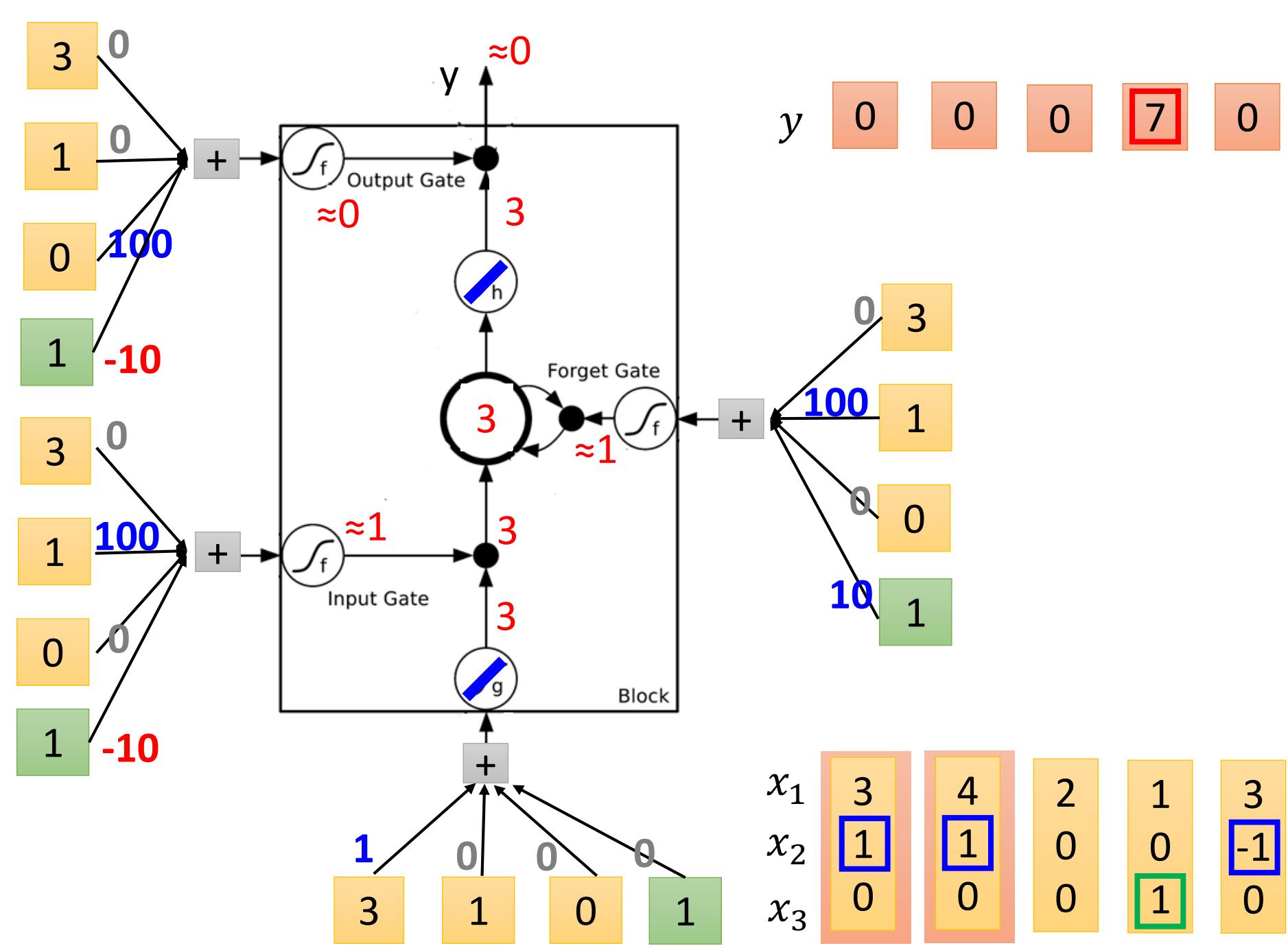
	0	0	3	3	7	7	7	0	6
x_1	1	3	2	4	2	1	3	6	1
x_2	0	1	0	1	0	0	-1	1	0
x_3	0	0	0	0	0	1	0	0	1
y	0	0	0	0	0	7	0	0	6

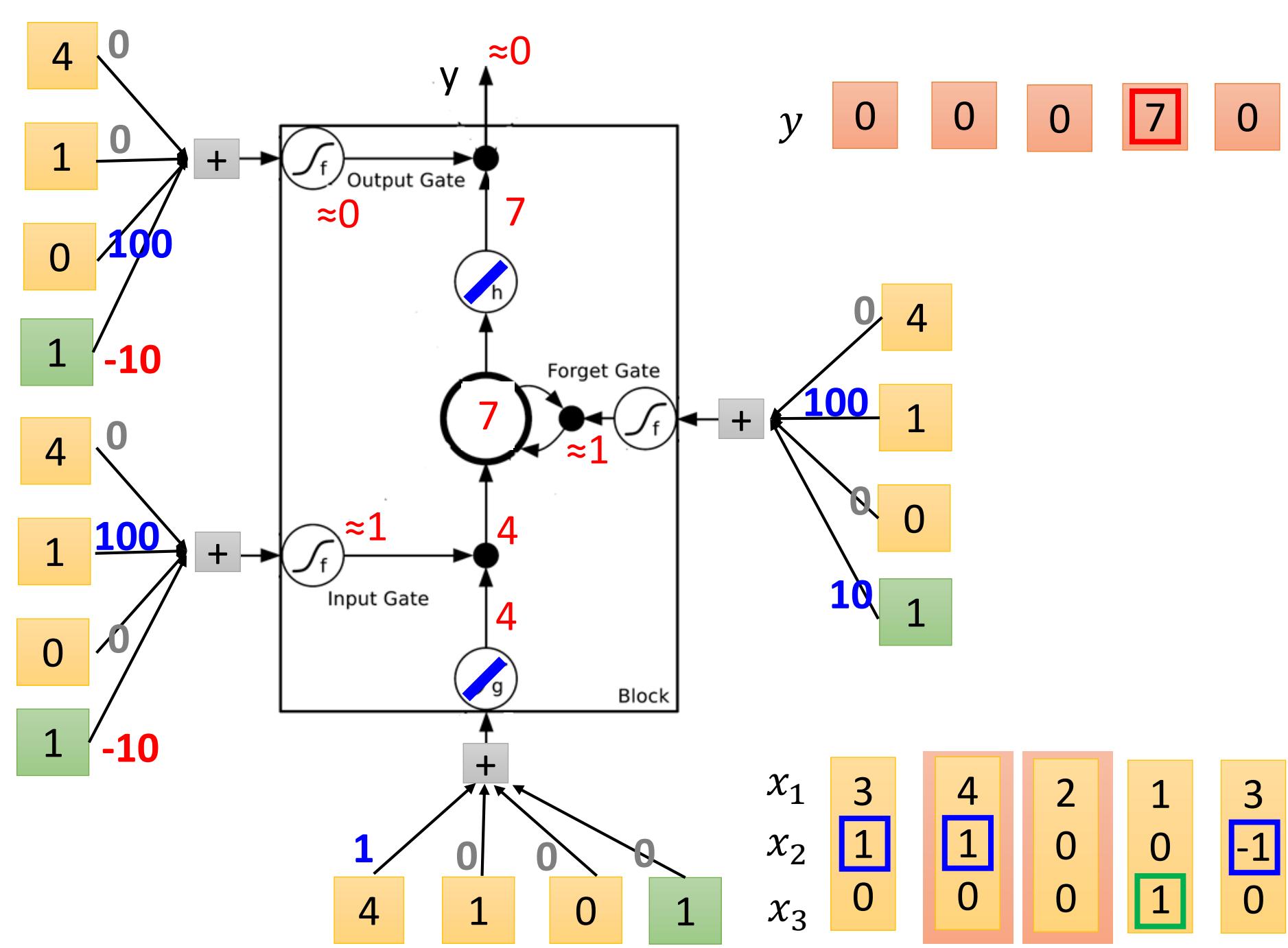
When $x_2 = 1$, add the numbers of x_1 into the memory

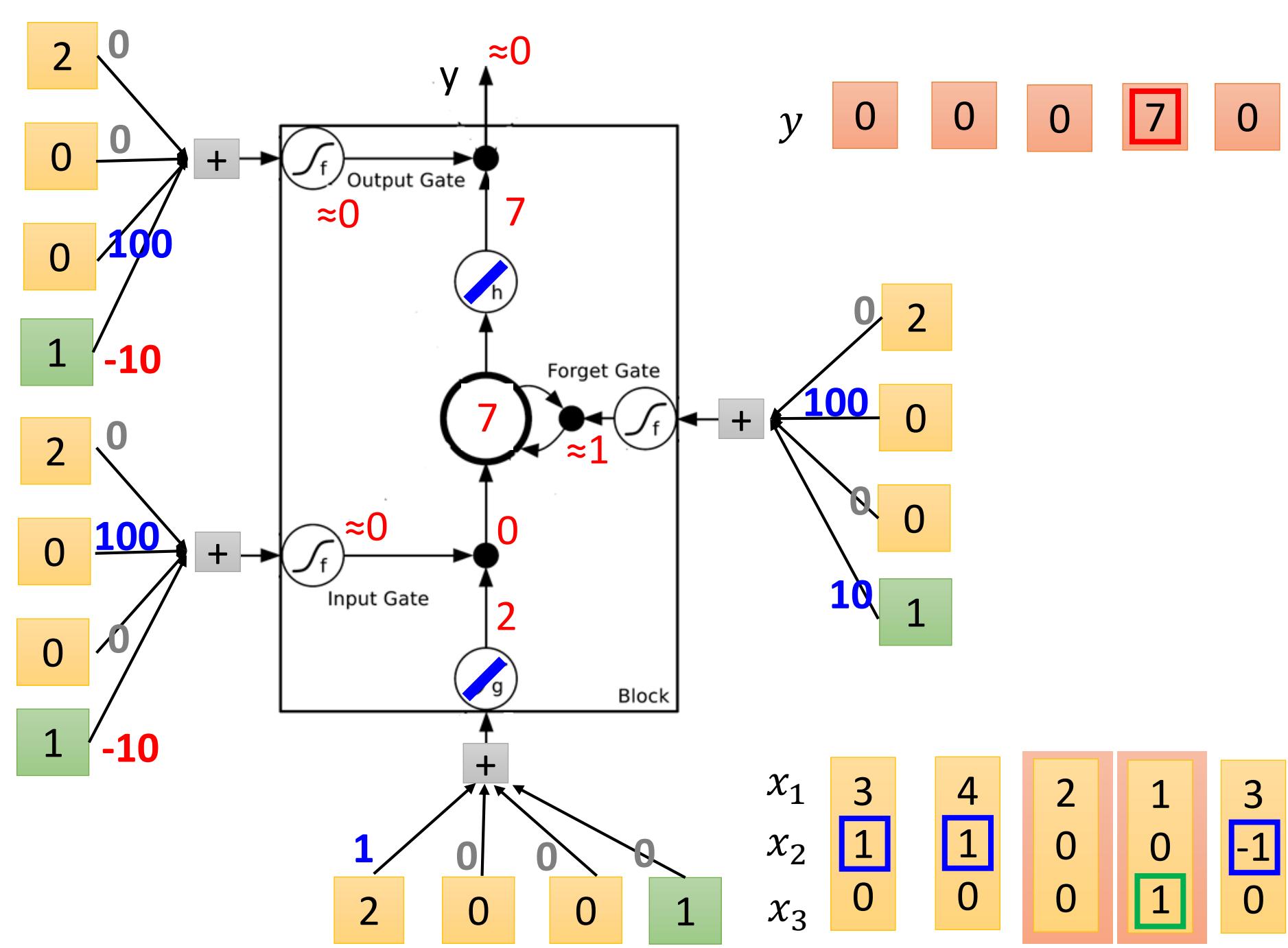
When $x_2 = -1$, reset the memory

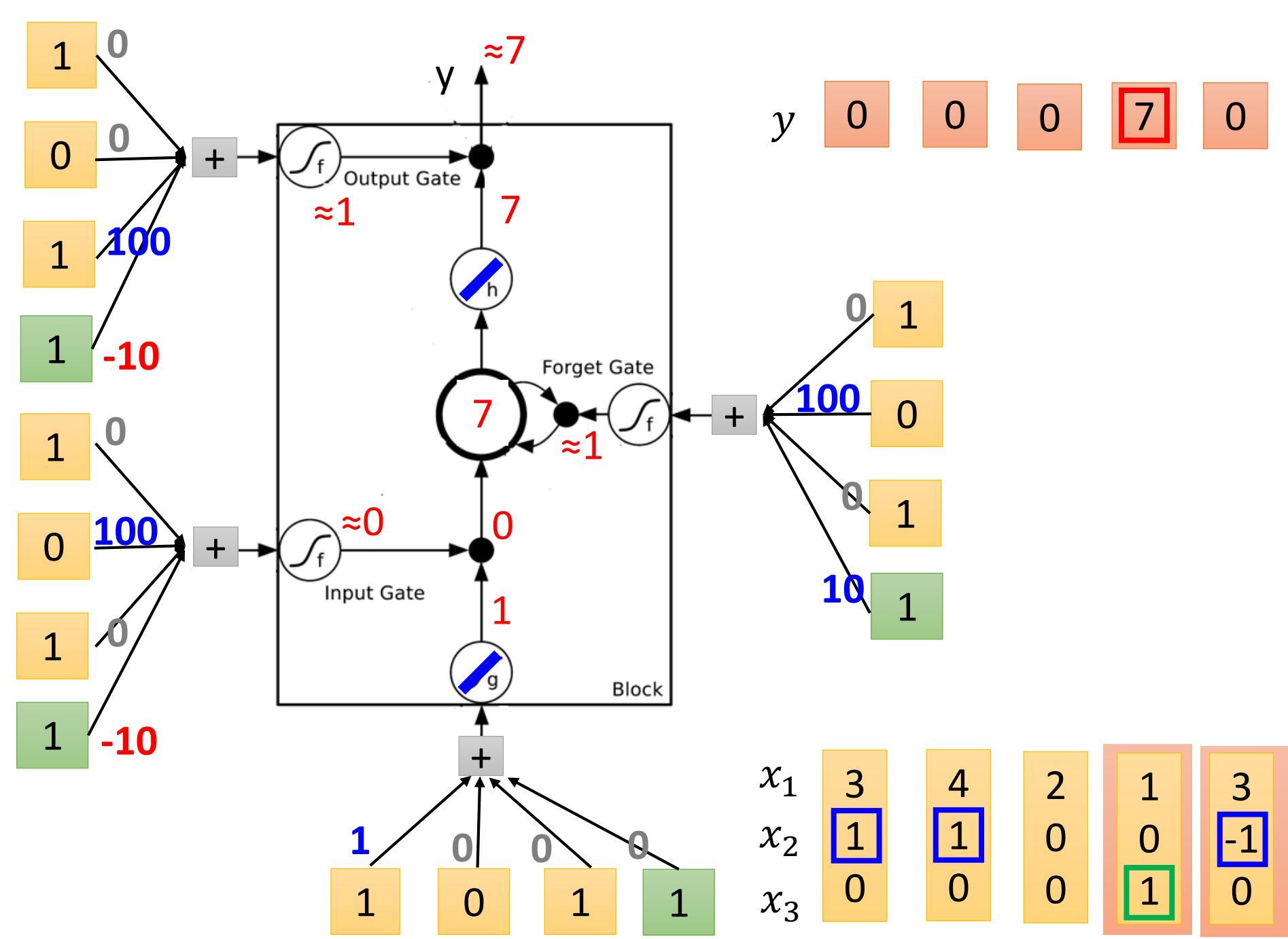
When $x_3 = 1$, output the number in the memory.

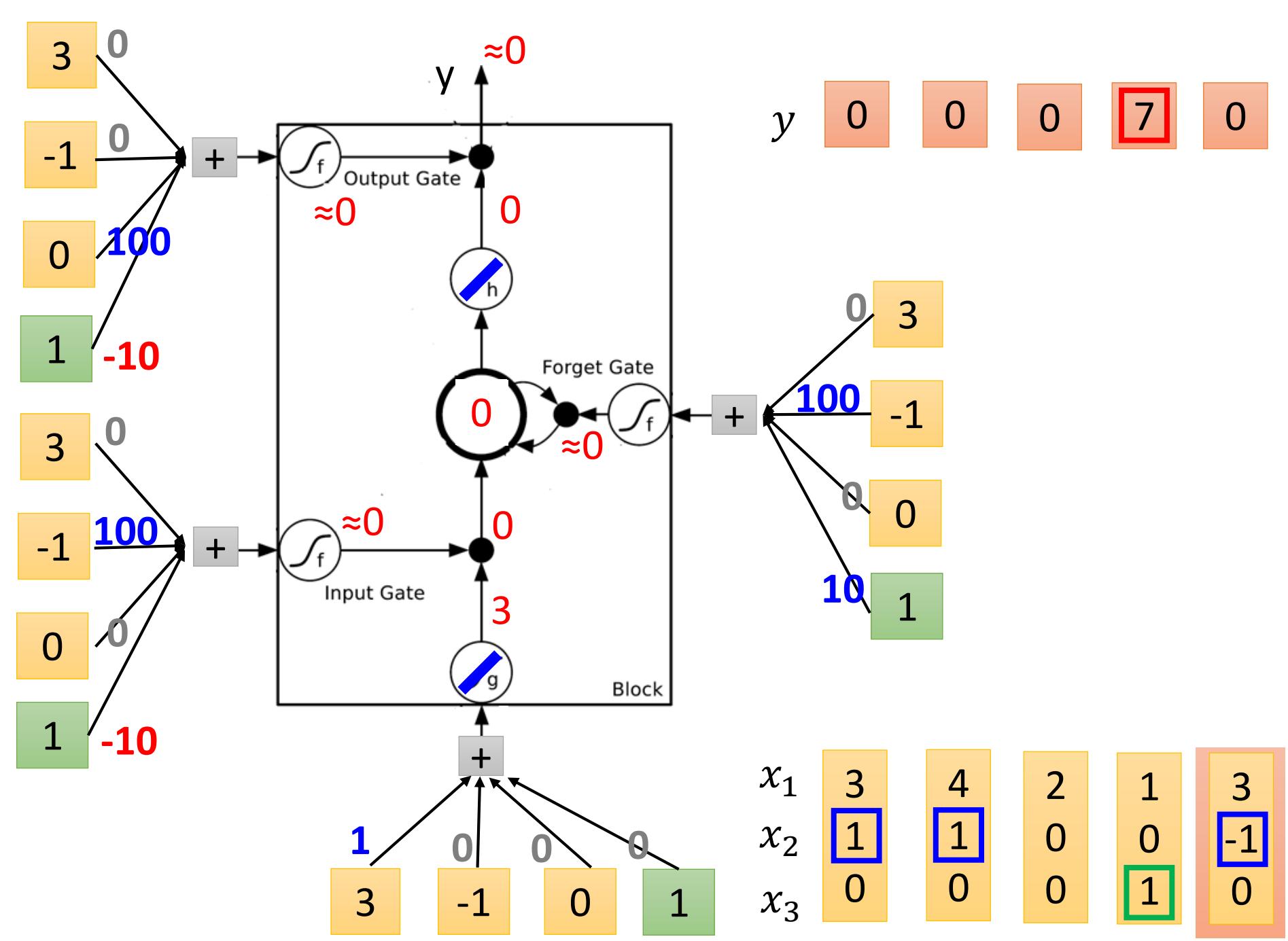






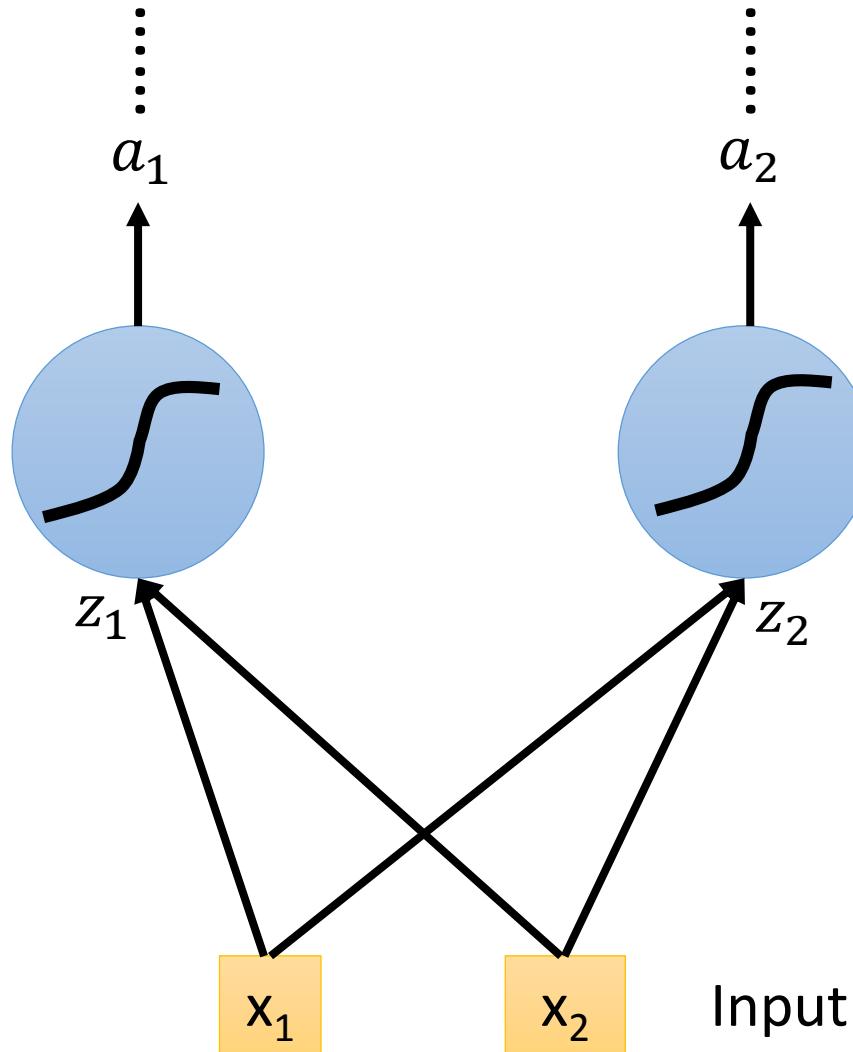


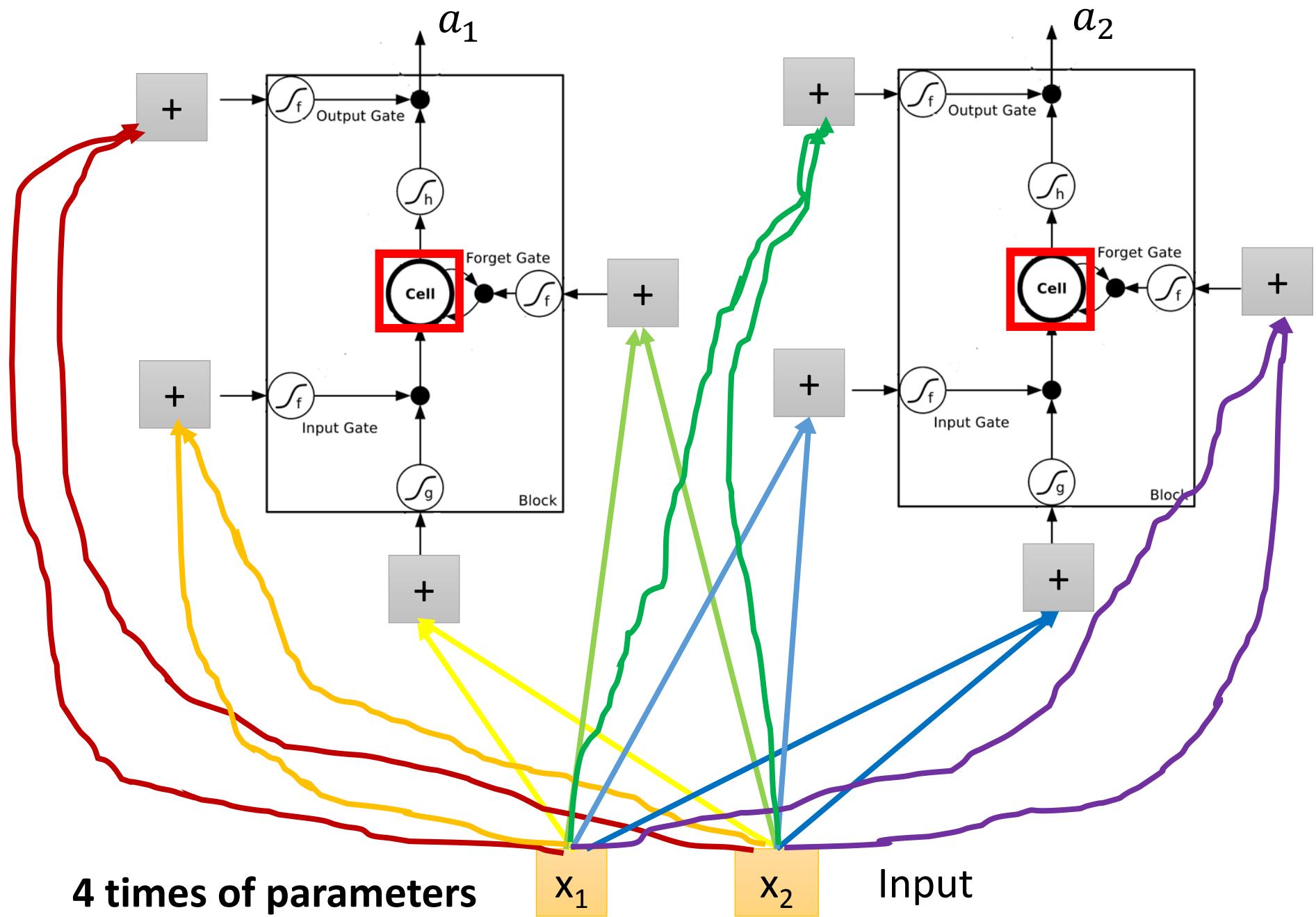




Original Network:

➤ Simply replace the neurons with LSTM

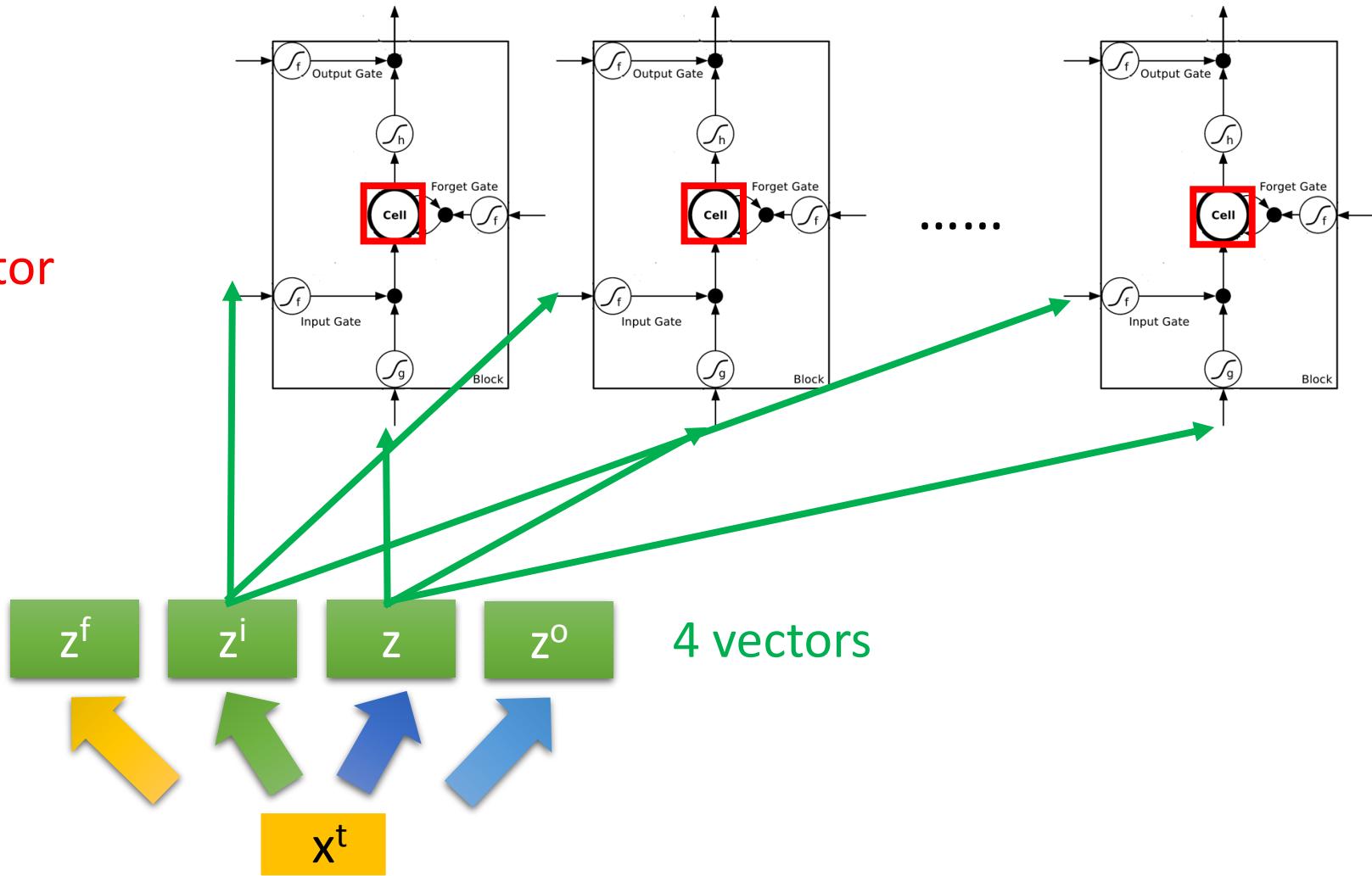




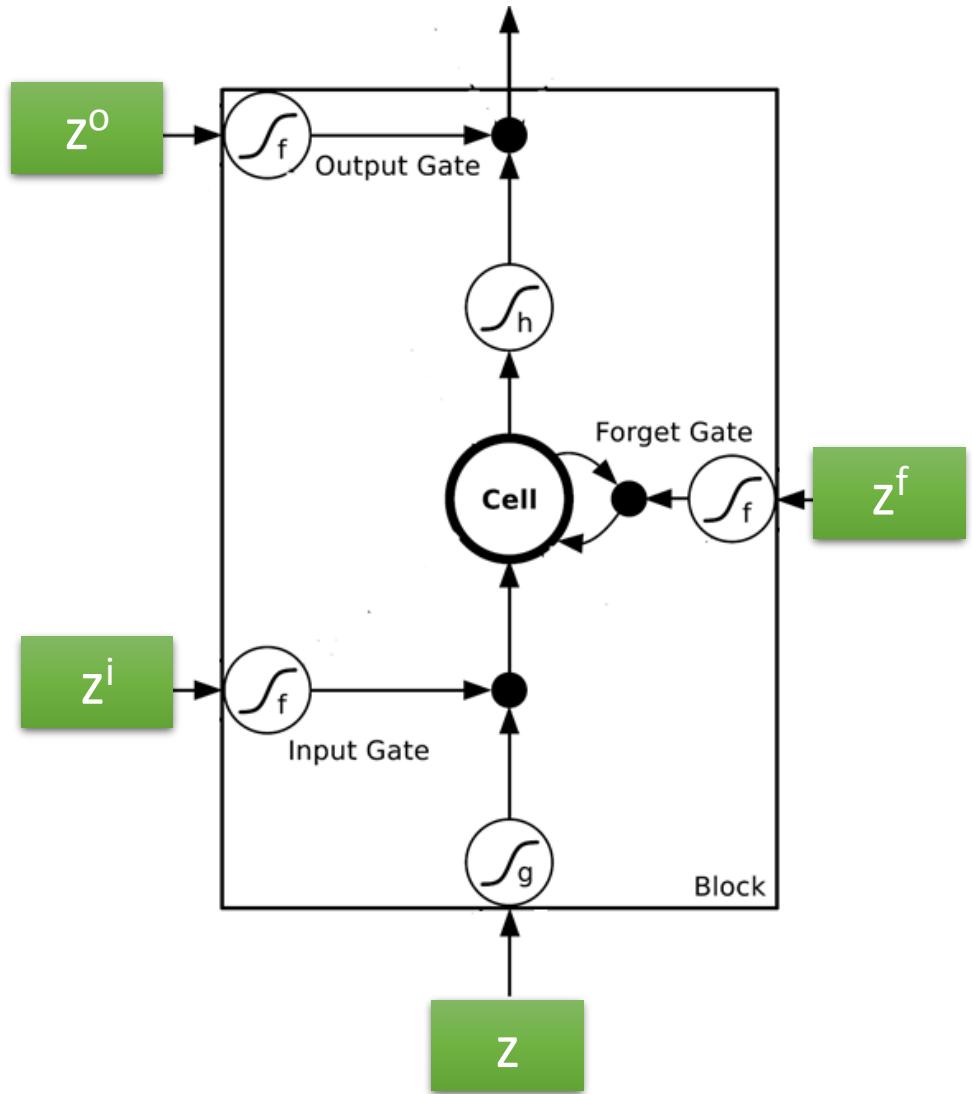
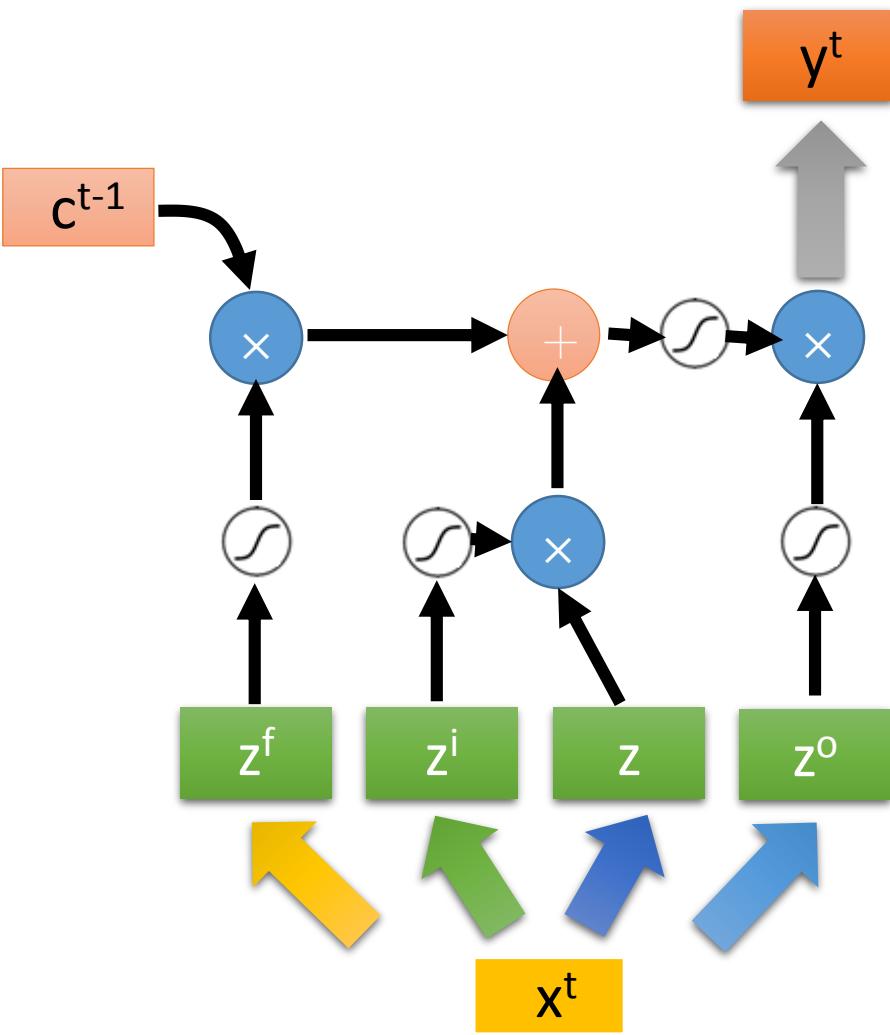
LSTM

C^{t-1}

vector

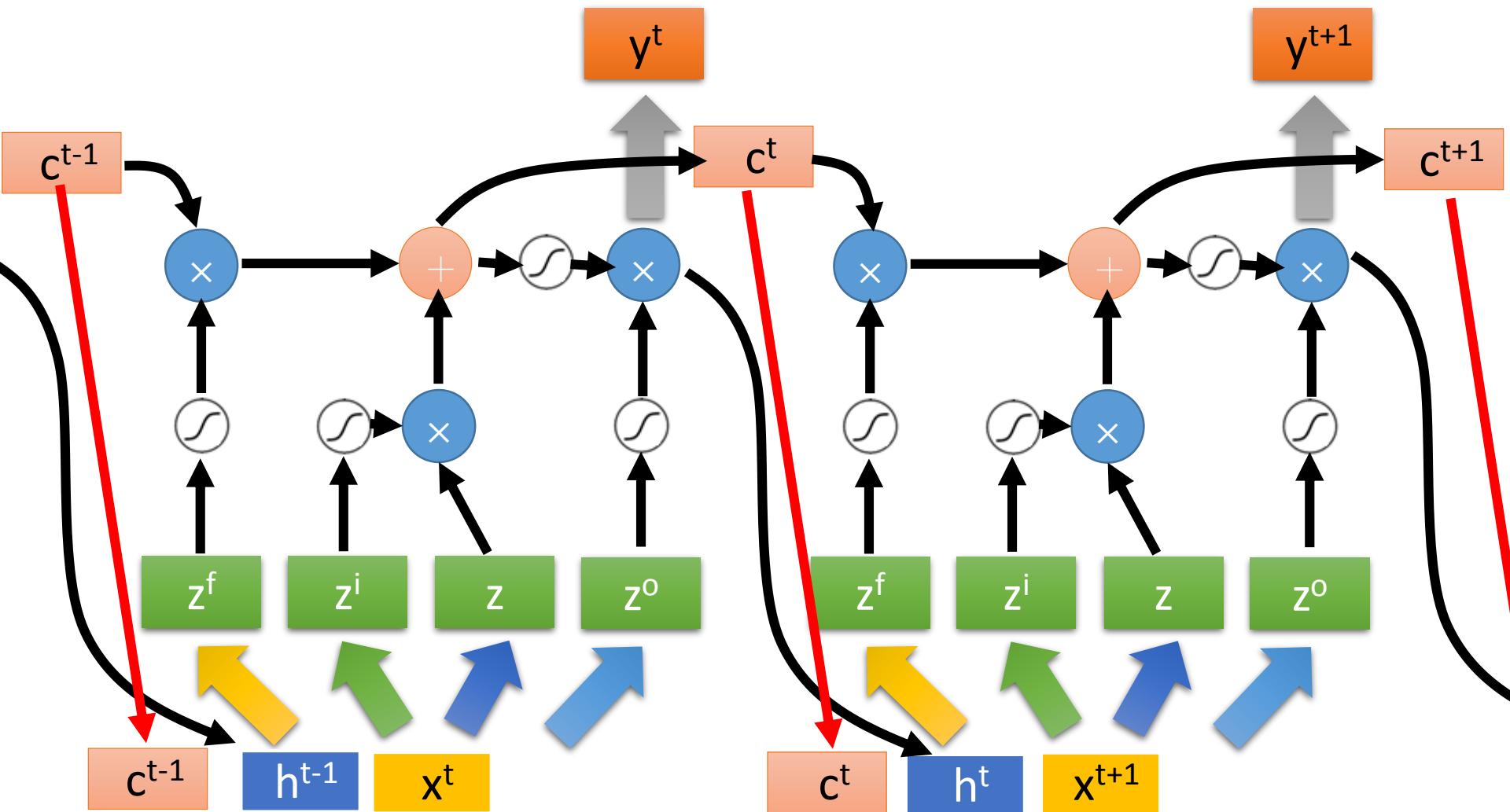


LSTM

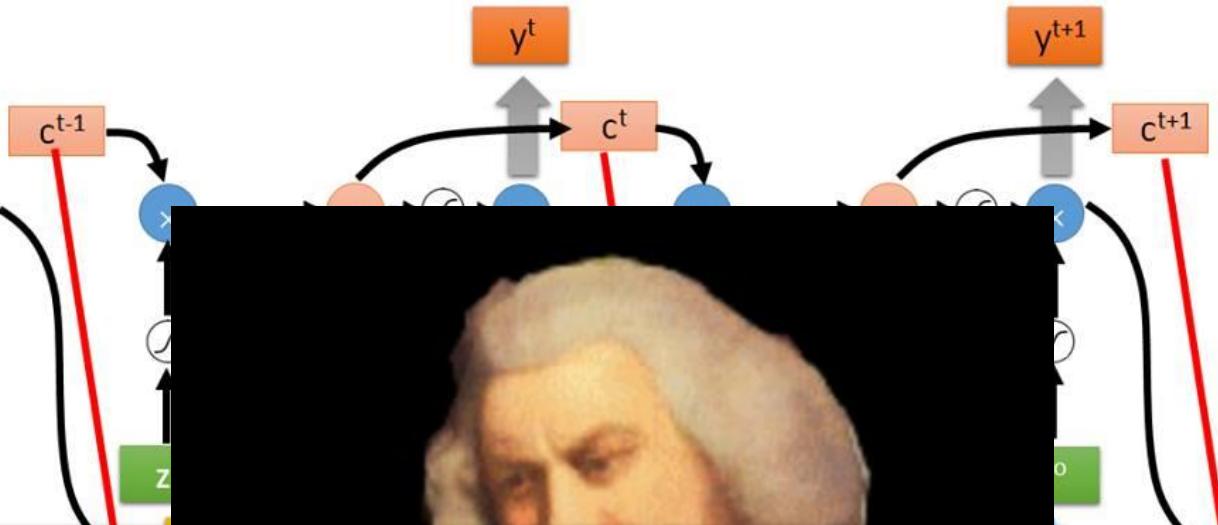


LSTM

Extension: “peephole”



Multiple-layer LSTM



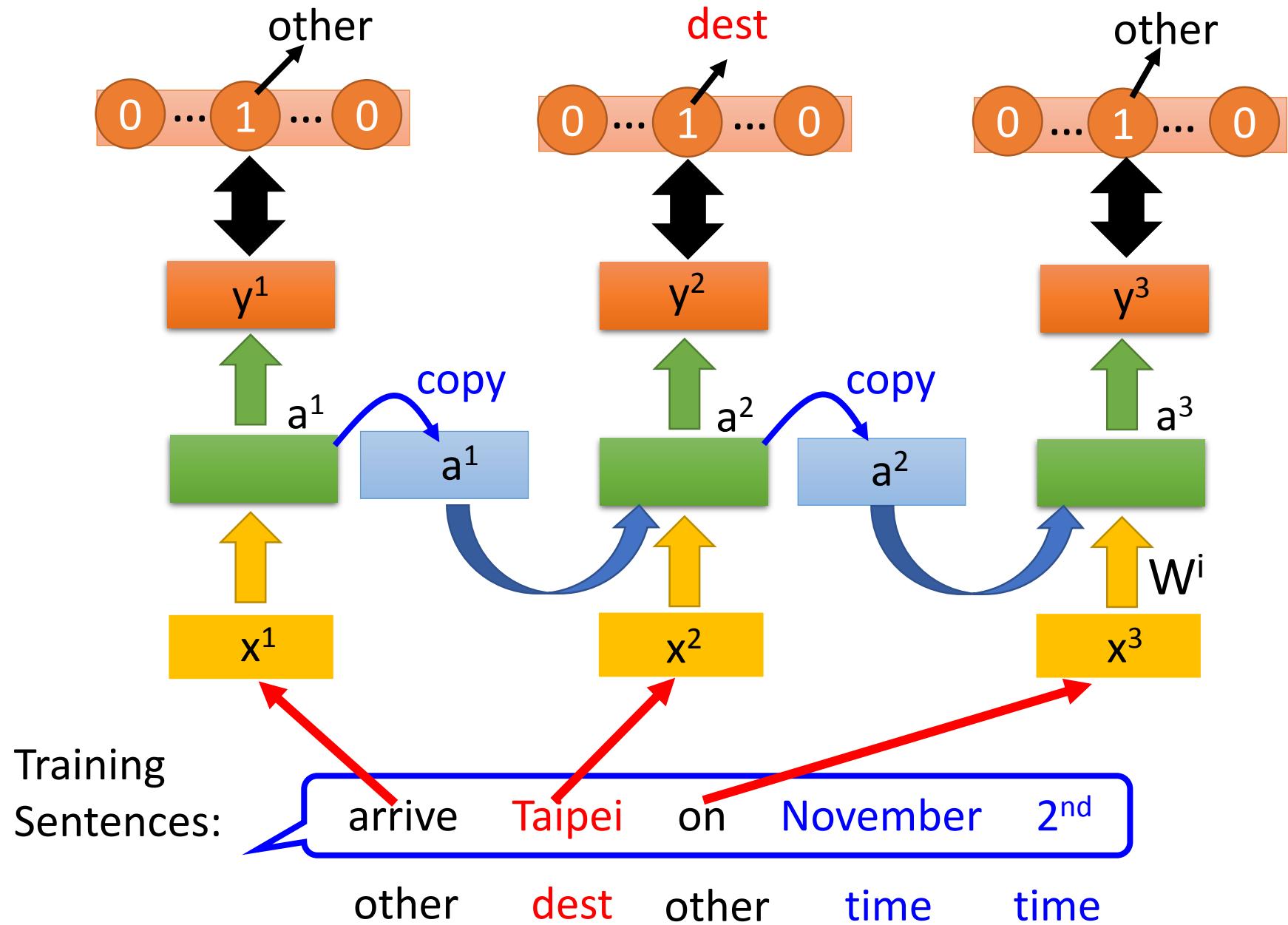
Don't worry if you cannot understand this.
Keras can handle it.

Keras supports
“LSTM”, “GRU”, “SimpleRNN” layers

This is quite
standard now.

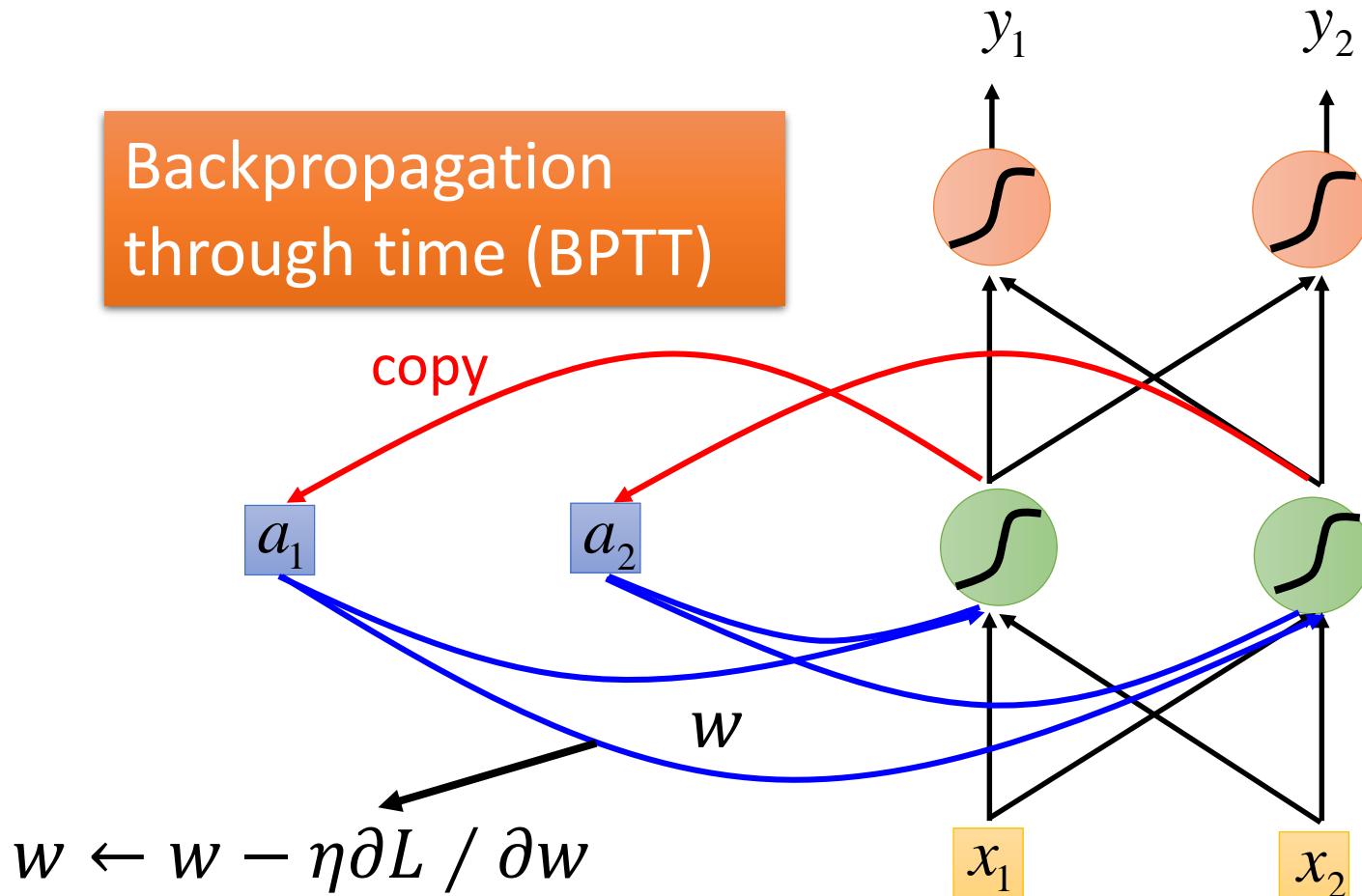


Learning Target



Learning

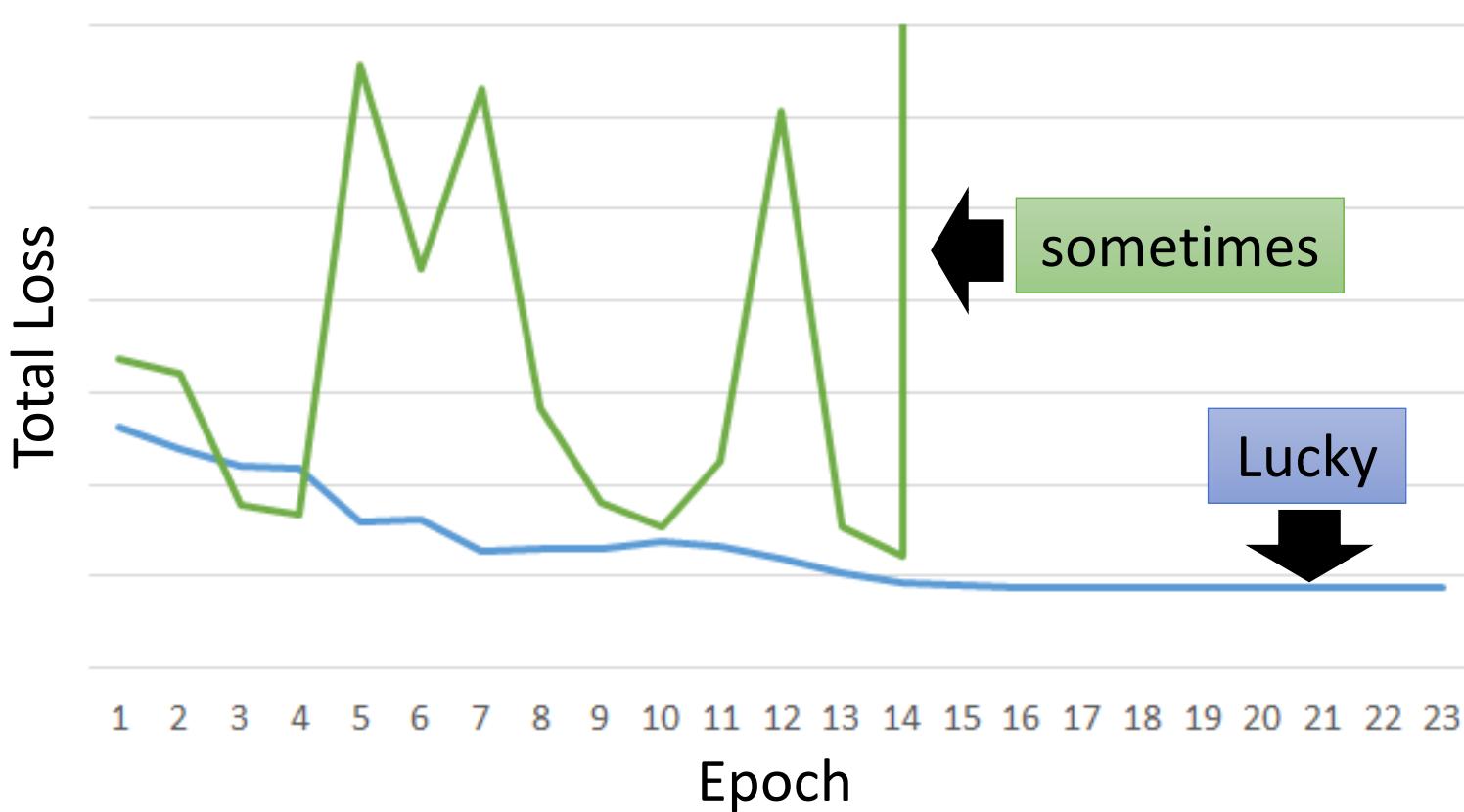
Backpropagation
through time (BPTT)



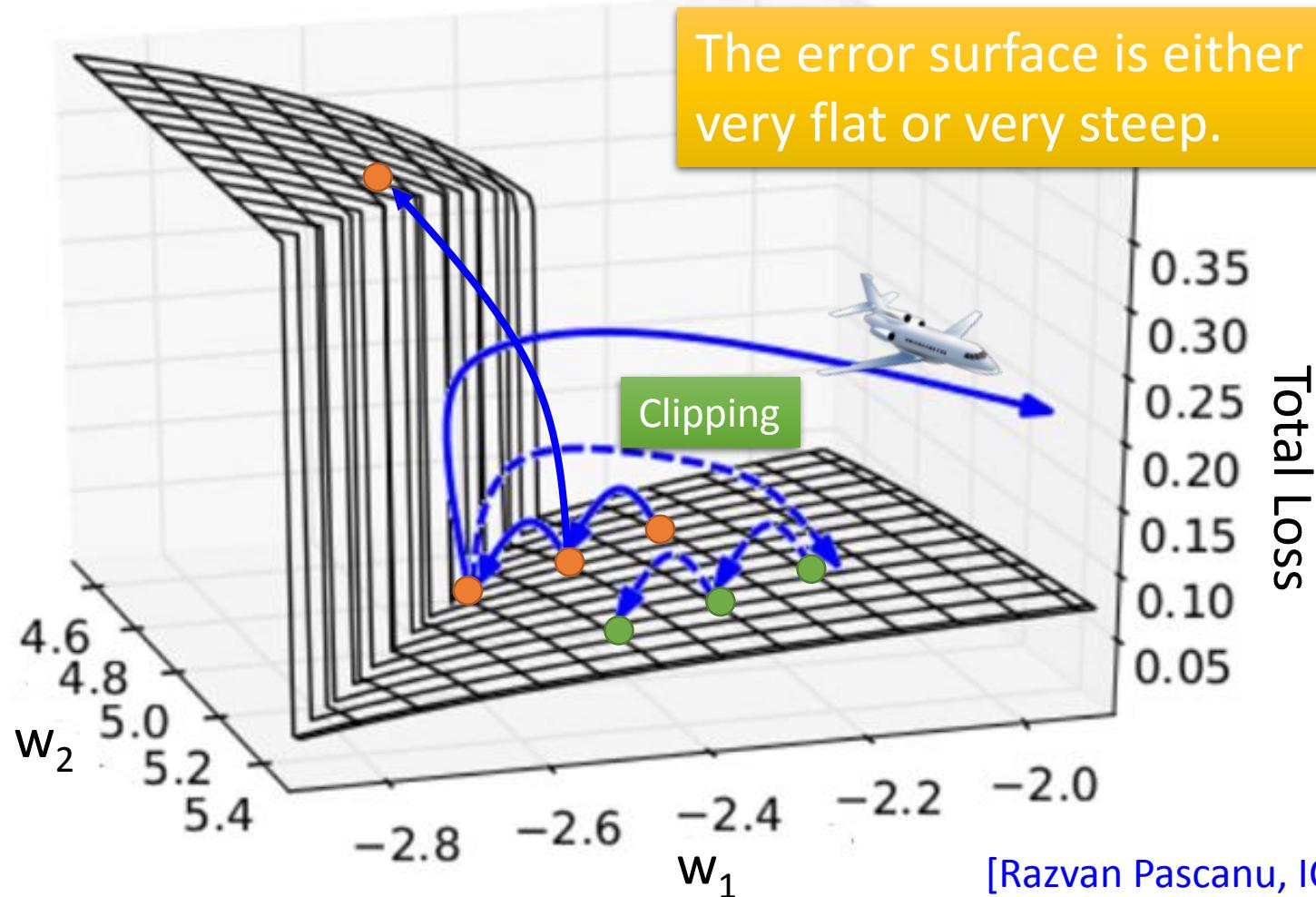
Unfortunately

- RNN-based network is not always easy to learn

Real experiments on Language modeling



The error surface is rough.



Why?

$$w = 1 \quad \rightarrow \quad y^{1000} = 1$$

$$w = 1.01 \quad \rightarrow \quad y^{1000} \approx 20000$$

$$w = 0.99 \quad \rightarrow \quad y^{1000} \approx 0$$

$$w = 0.01 \quad \rightarrow \quad y^{1000} \approx 0$$

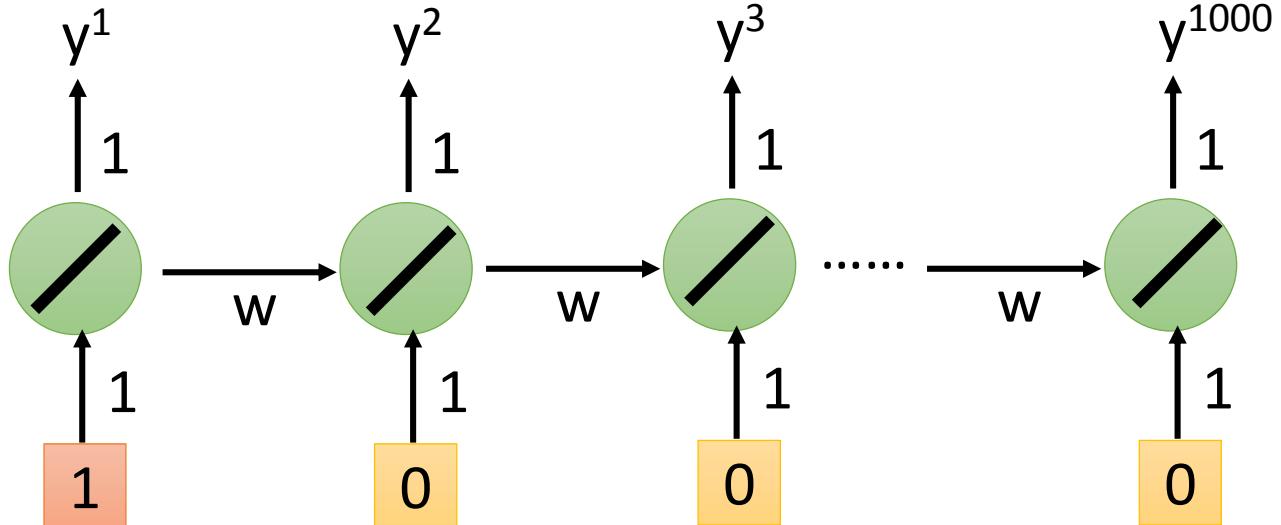
Large
 $\partial L / \partial w$

Small
Learning rate?

small
 $\partial L / \partial w$

Large
Learning rate?

Toy Example



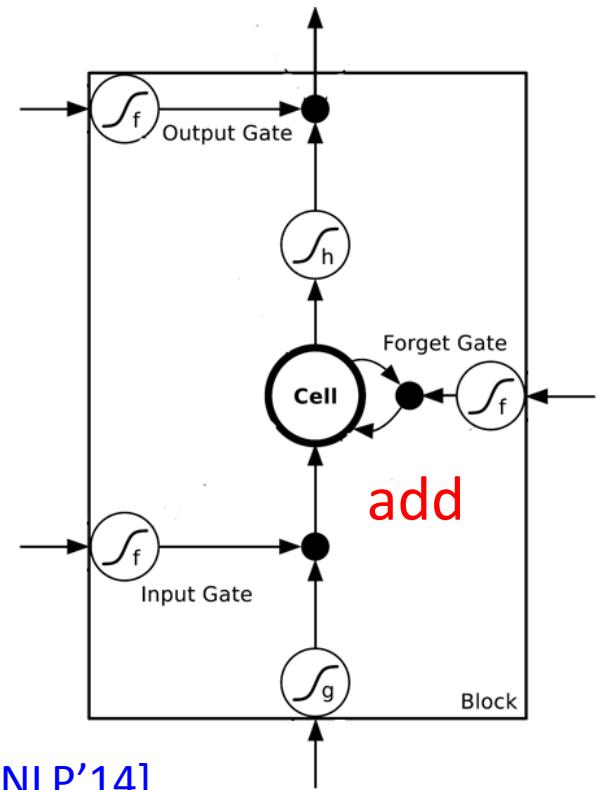
Helpful Techniques

• Long Short-term Memory (LSTM)

- Can deal with gradient vanishing (not gradient explode)
 - Memory and input are added
 - The influence never disappears unless forget gate is closed
- No Gradient vanishing
(If forget gate is opened.)

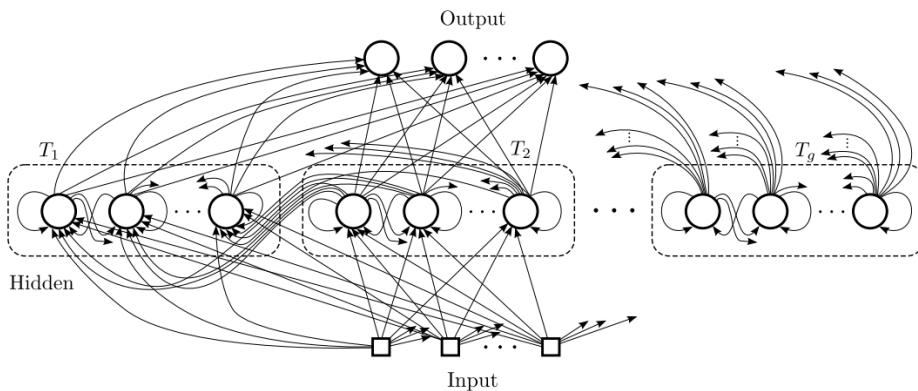
Gated Recurrent Unit (GRU):
simpler than LSTM

[Cho, EMNLP'14]



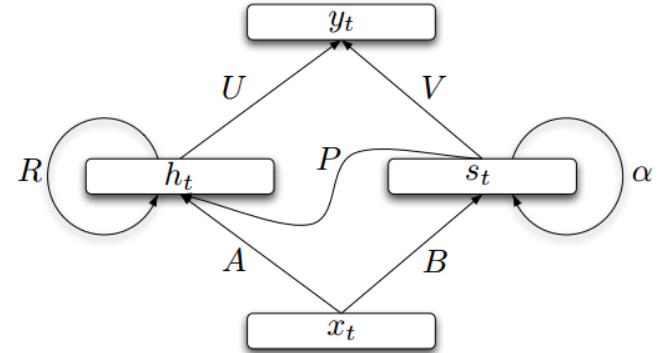
Helpful Techniques

Clockwise RNN



[Jan Koutnik, JMLR'14]

Structurally Constrained Recurrent Network (SCRN)



[Tomas Mikolov, ICLR'15]

Vanilla RNN Initialized with Identity matrix + ReLU activation function [Quoc V. Le, arXiv'15]

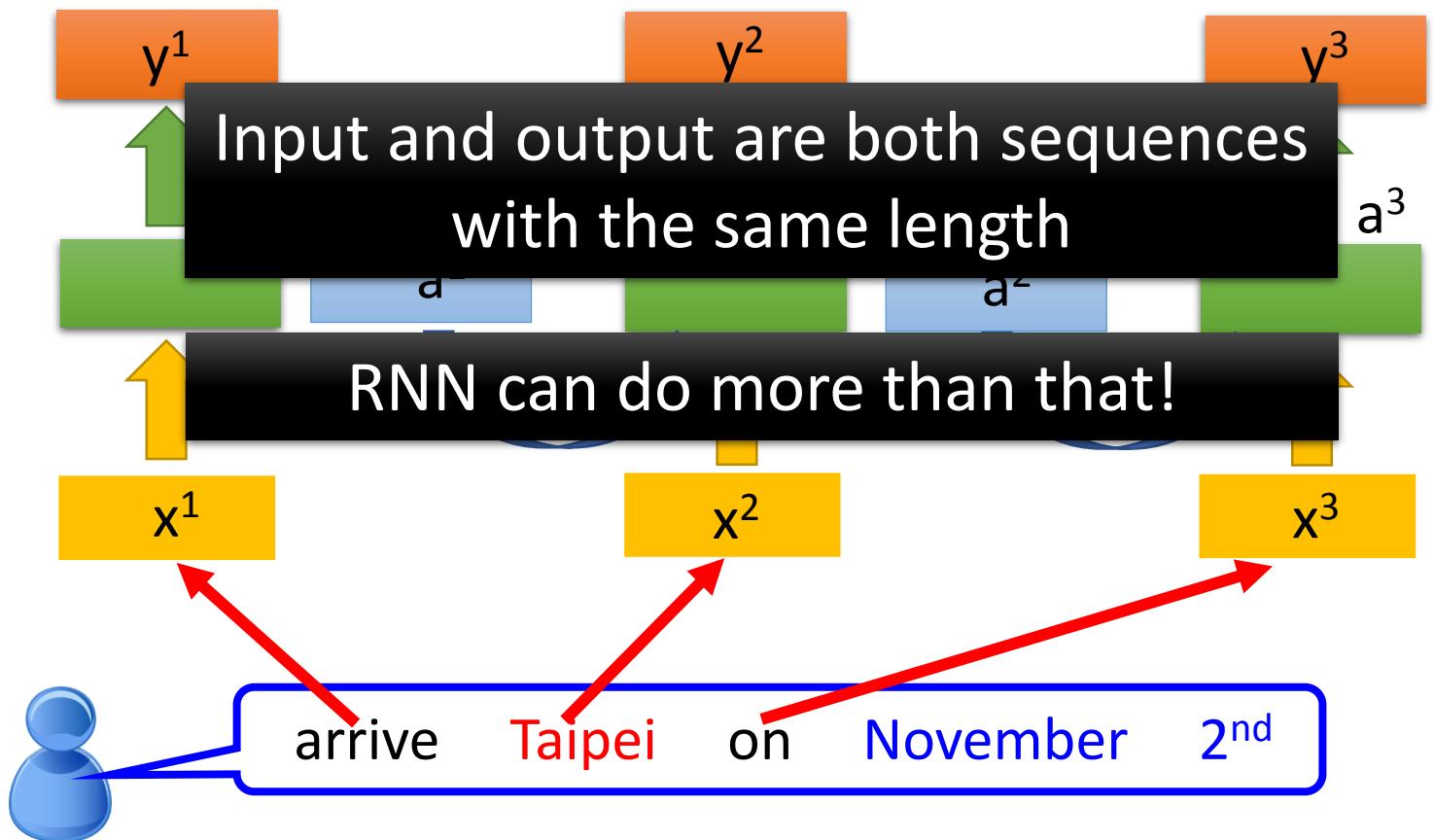
- Outperform or be comparable with LSTM in 4 different tasks

More Applications

Probability of
“arrive” in each slot

Probability of
“Taipei” in each slot

Probability of
“on” in each slot



Many to one

- Input is a vector sequence, but output is only one vector

Sentiment Analysis

看了這部電影覺
得很高興

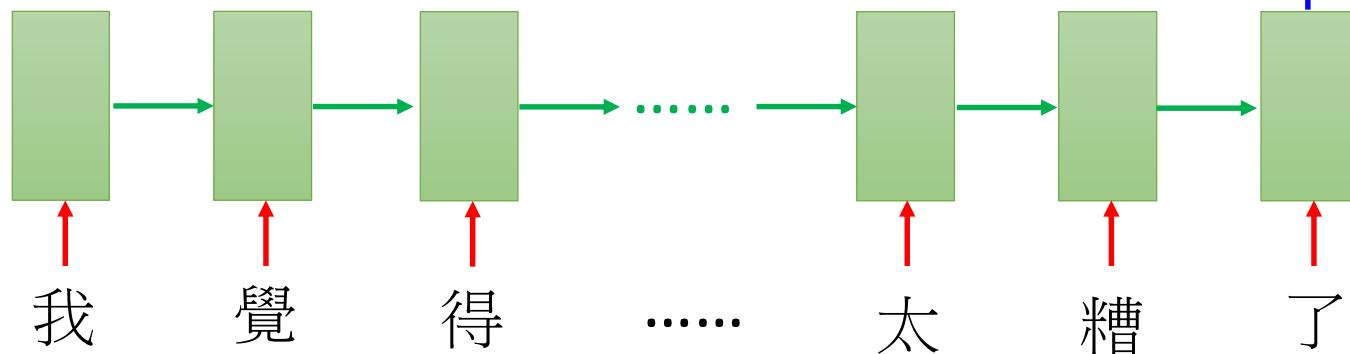
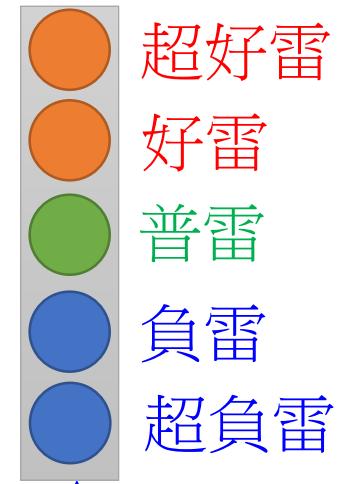
Positive (正雷)

這部電影太糟了
.....

Negative (負雷)

這部電影很
棒

Positive (正雷)

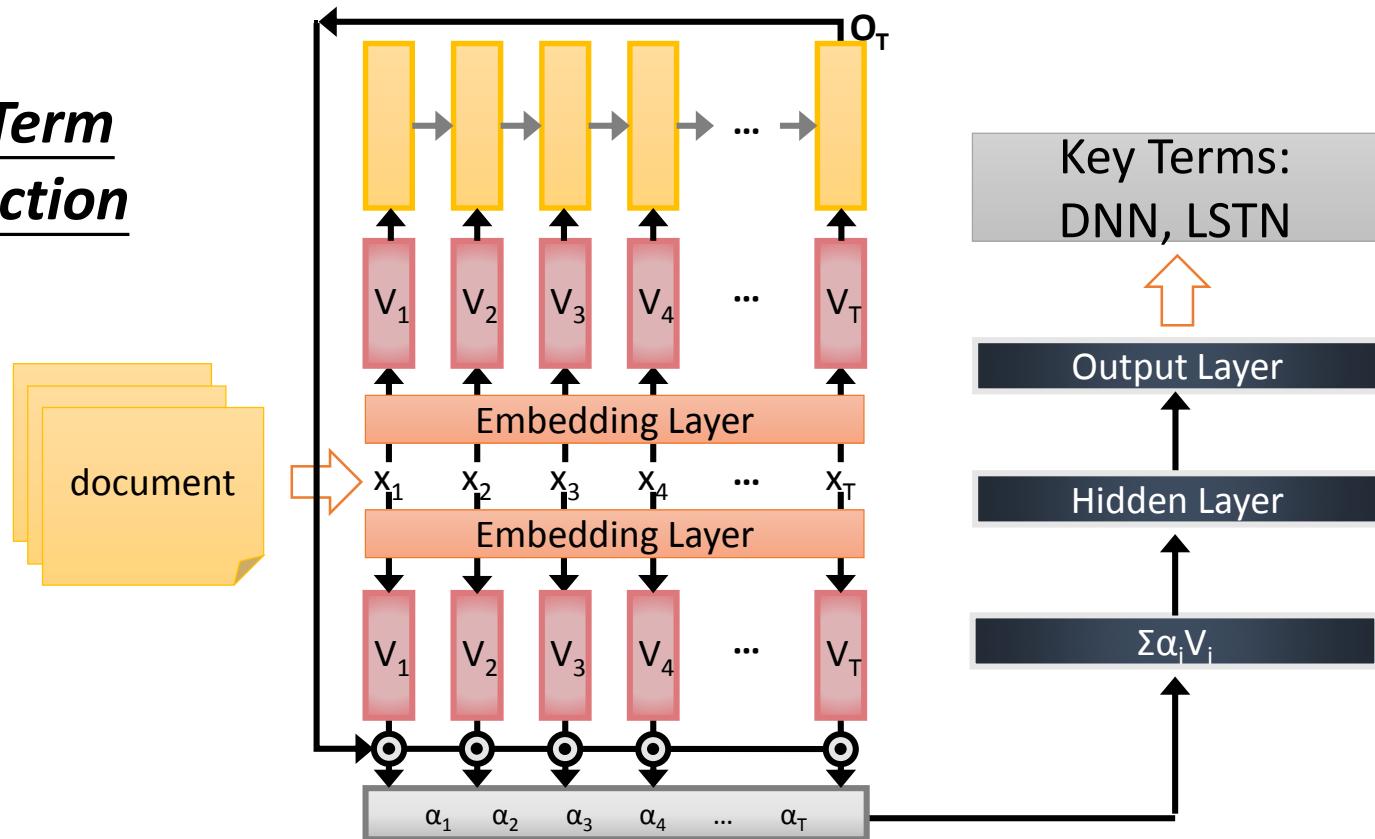


Many to one

[Shen & Lee, Interspeech 16]

- Input is a vector sequence, but output is only one vector

Key Term Extraction



Many to Many (Output is shorter)

- Both input and output are both sequences, **but the output is shorter.**
 - E.g. **Speech Recognition**

Problem?

Why can't it be
“好棒棒”

Output: “好棒” (character sequence)



好 好 好 棒 棒 棒 棒

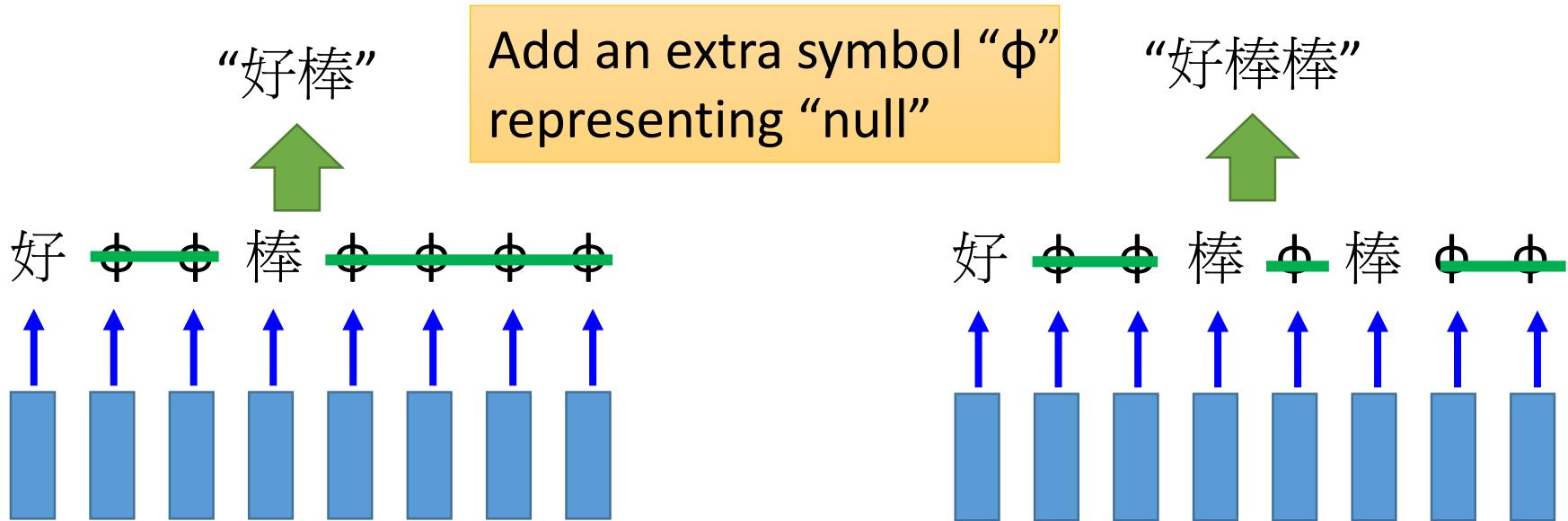
Input:

(vector sequence)



Many to Many (Output is shorter)

- Both input and output are both sequences, **but the output is shorter.**
- Connectionist Temporal Classification (CTC) [Alex Graves, ICML'06][Alex Graves, ICML'14][Hasim Sak, Interspeech'15][Jie Li, Interspeech'15][Andrew Senior, ASRU'15]



Many to Many (Output is shorter)

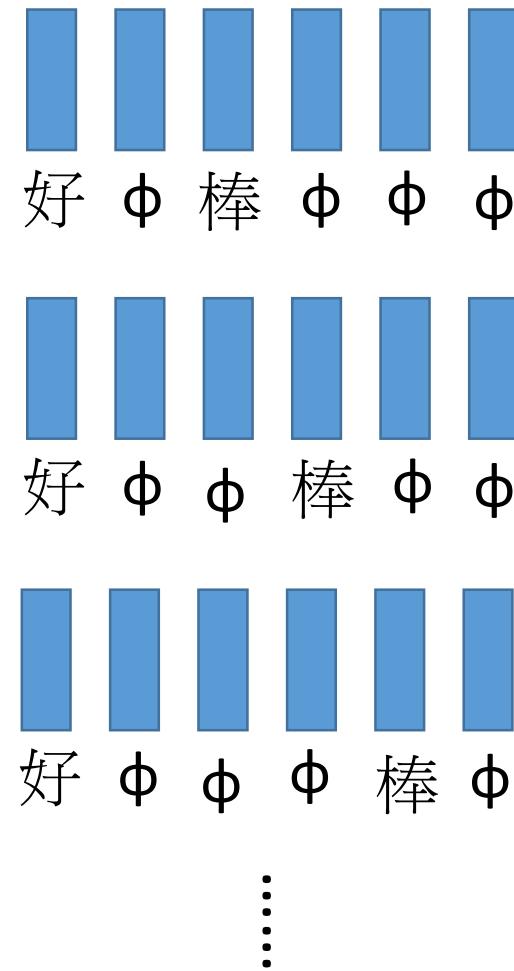
- CTC: Training

Acoustic
Features:



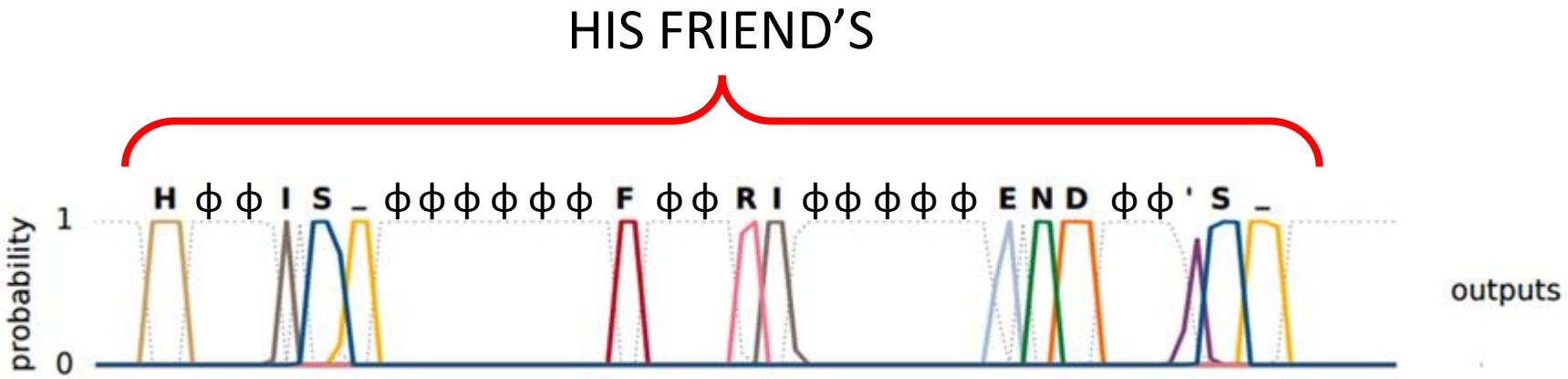
Label: 好 棒

All possible alignments are
considered as correct.



Many to Many (Output is shorter)

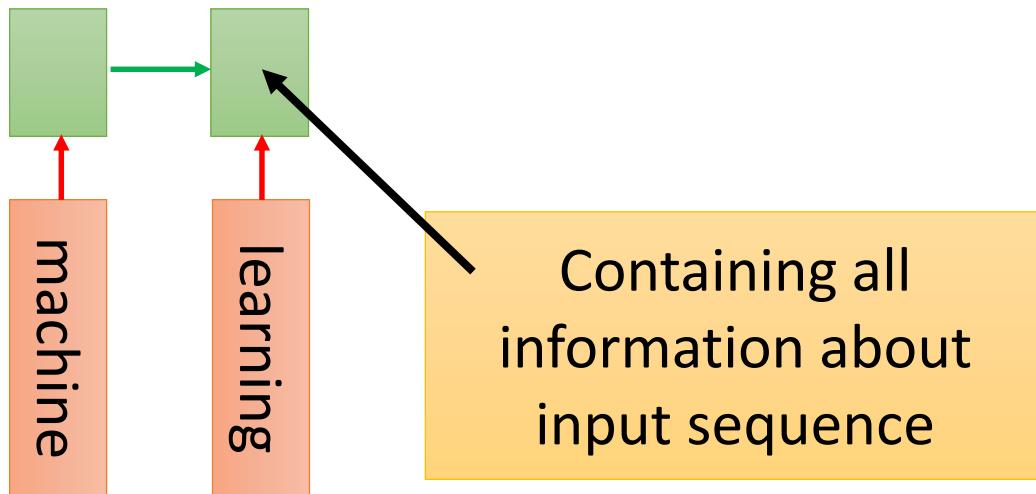
- CTC: example



Graves, Alex, and Navdeep Jaitly. "Towards end-to-end speech recognition with recurrent neural networks." *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*. 2014.

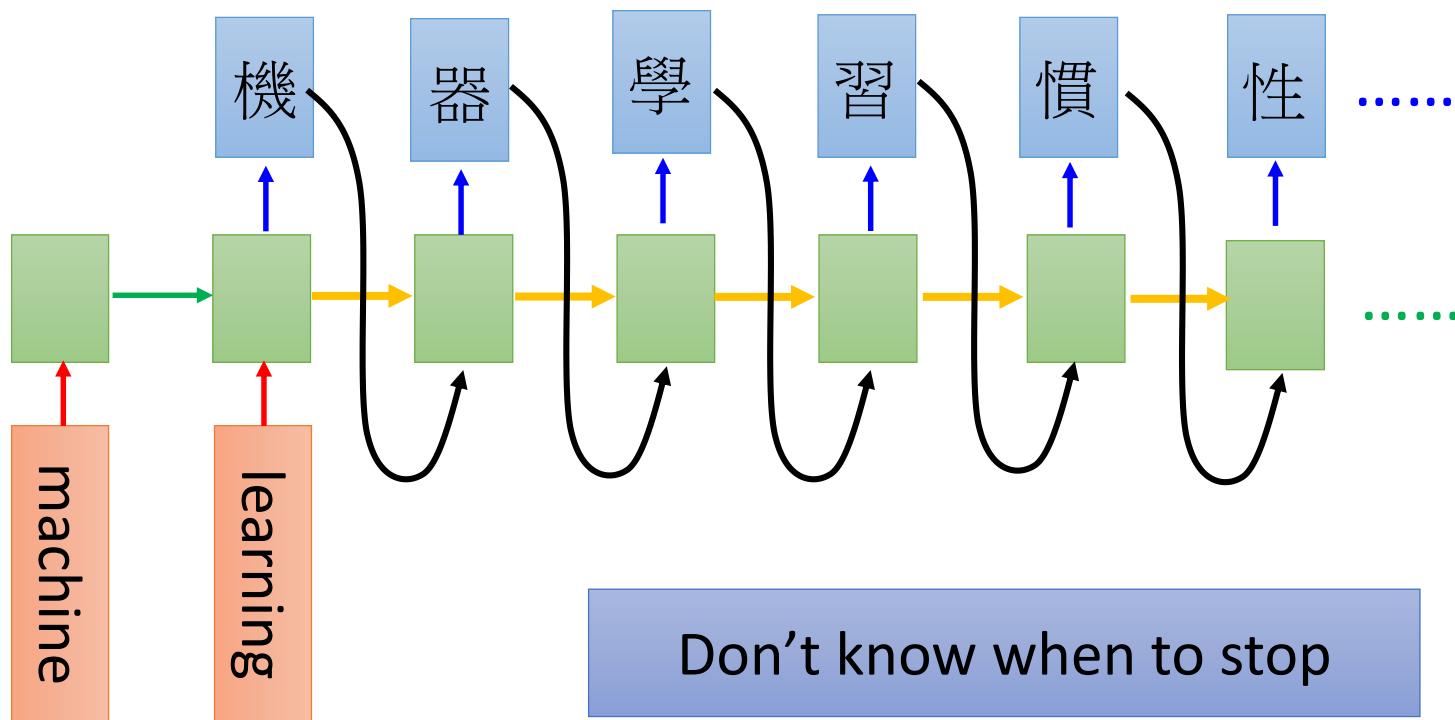
Many to Many (No Limitation)

- Both input and output are both sequences *with different lengths*. → *Sequence to sequence learning*
 - E.g. *Machine Translation* (machine learning → 機器學習)



Many to Many (No Limitation)

- Both input and output are both sequences with different lengths. → Sequence to sequence learning
 - E.g. Machine Translation (machine learning → 機器學習)



Many to Many (No Limitation)

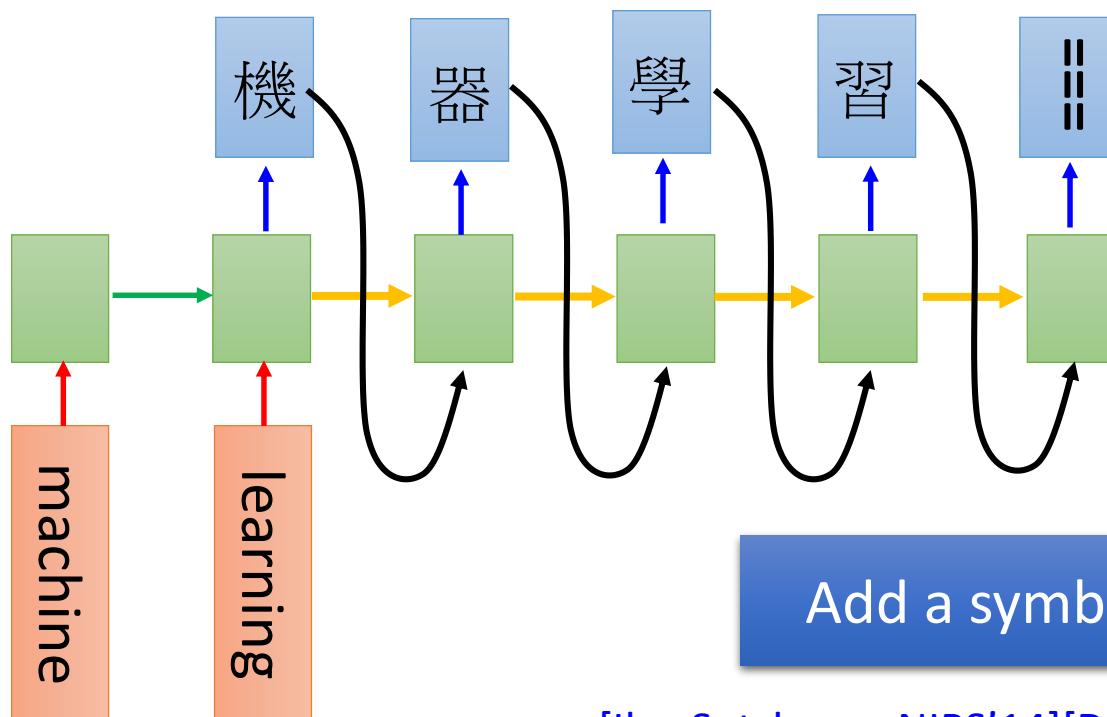
推	: 超	06/12 10:39
推	: n: 人	06/12 10:40
推	: tion: 正	06/12 10:41
→	: host: 大	06/12 10:47
推	: 中	06/12 10:59
推	: 403: 天	06/12 11:11
推	: 外	06/12 11:13
推	: 527: 飛	06/12 11:17
→	: 990b: 仙	06/12 11:32
→	: 512: 草	06/12 12:15

推 tlkagk: =====斷=====

接龍推文是ptt在推文中的一種趣味玩法，與推齊有些類似但又有所不同，是指在推文中接續上一樓的字句，而推出連續的意思。該類玩法確切起源已不可知(鄉民百科)

Many to Many (No Limitation)

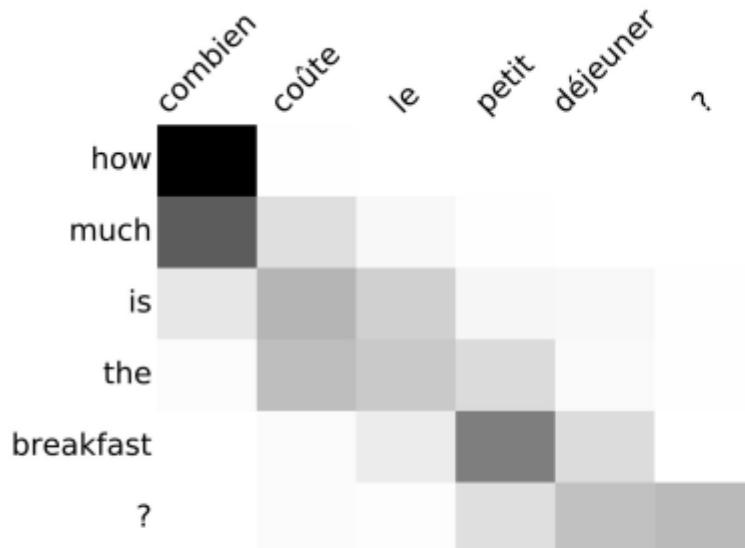
- Both input and output are both sequences with different lengths. → Sequence to sequence learning
 - E.g. Machine Translation (machine learning→機器學習)



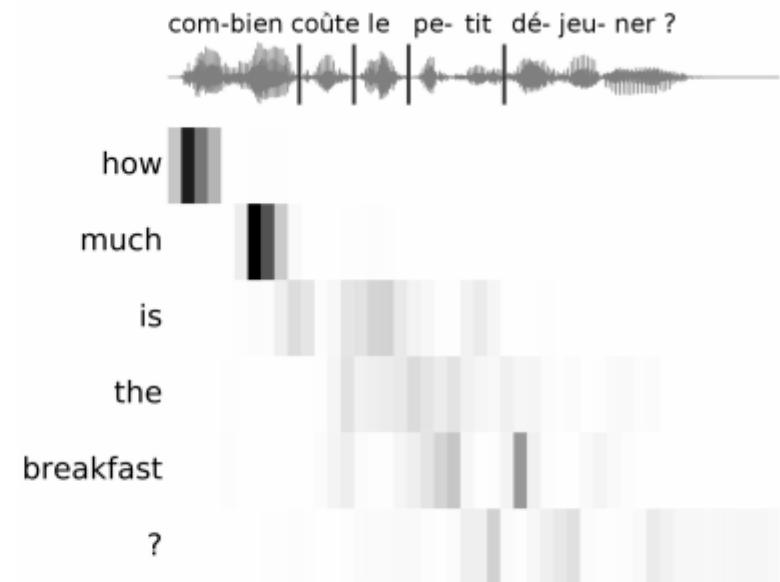
[Ilya Sutskever, NIPS'14][Dzmitry Bahdanau, arXiv'15]

Many to Many (No Limitation)

- Both input and output are both sequences with different lengths. → Sequence to sequence learning
 - E.g. Machine Translation (machine learning → 機器學習)



(a) Machine translation alignment



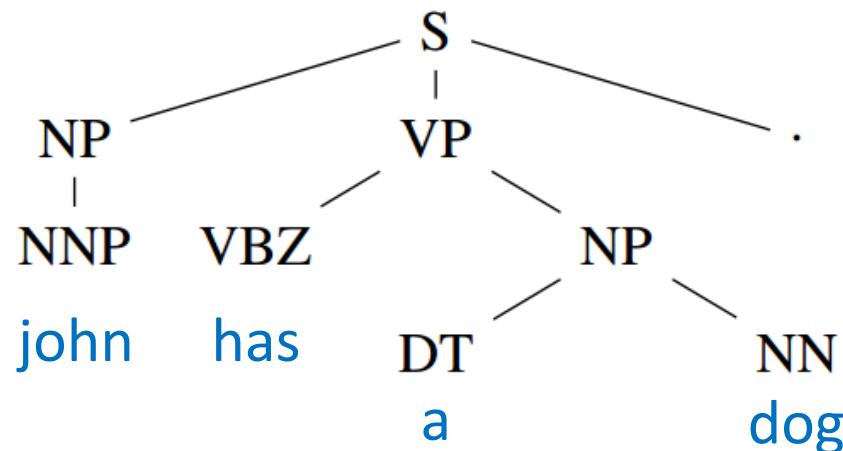
(b) Speech translation alignment

Figure 1: Alignments performed by the attention model during training

Beyond Sequence

- Syntactic parsing

John has a dog . →

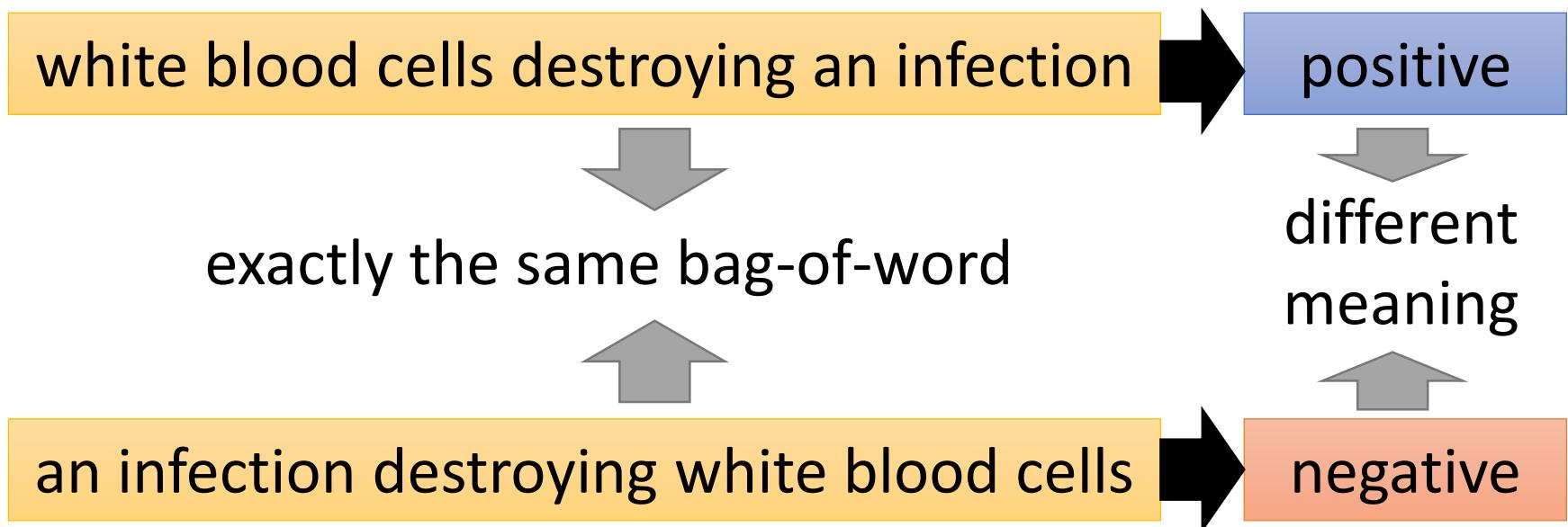


John has a dog . →

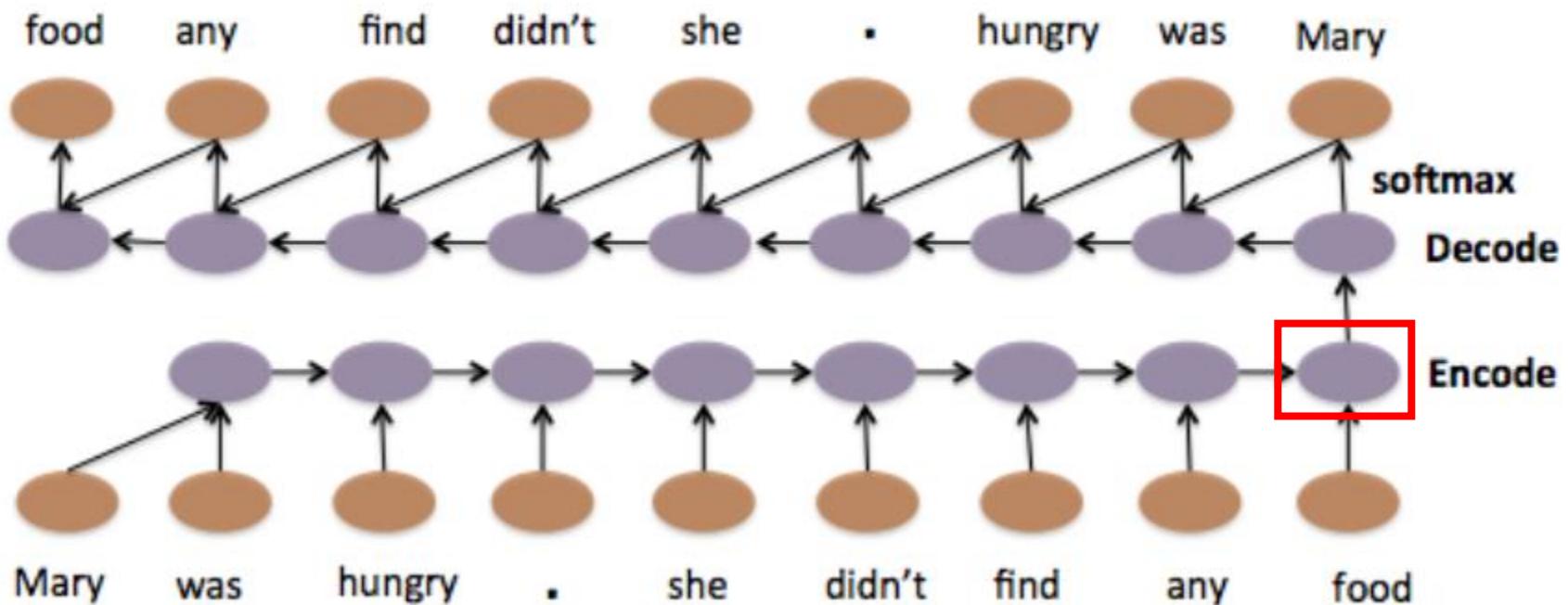
$(S (NP NNP)_{NP} (VP VBZ (NP DT NN)_{NP})_{VP} .)_S$

Sequence-to-sequence Auto-encoder - Text

- To understand the meaning of a word sequence, the order of the words can not be ignored.

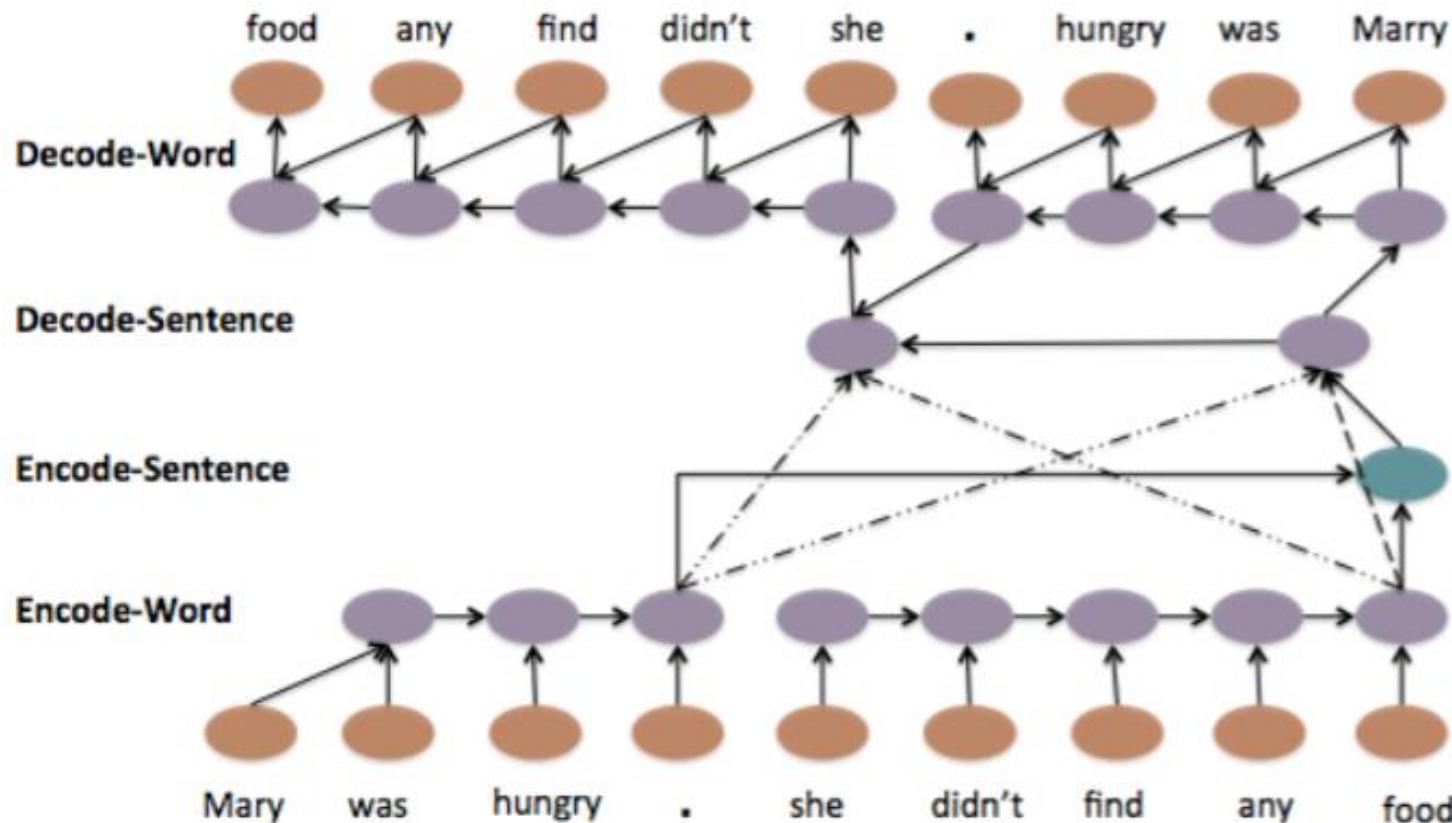


Sequence-to-sequence Auto-encoder - Text



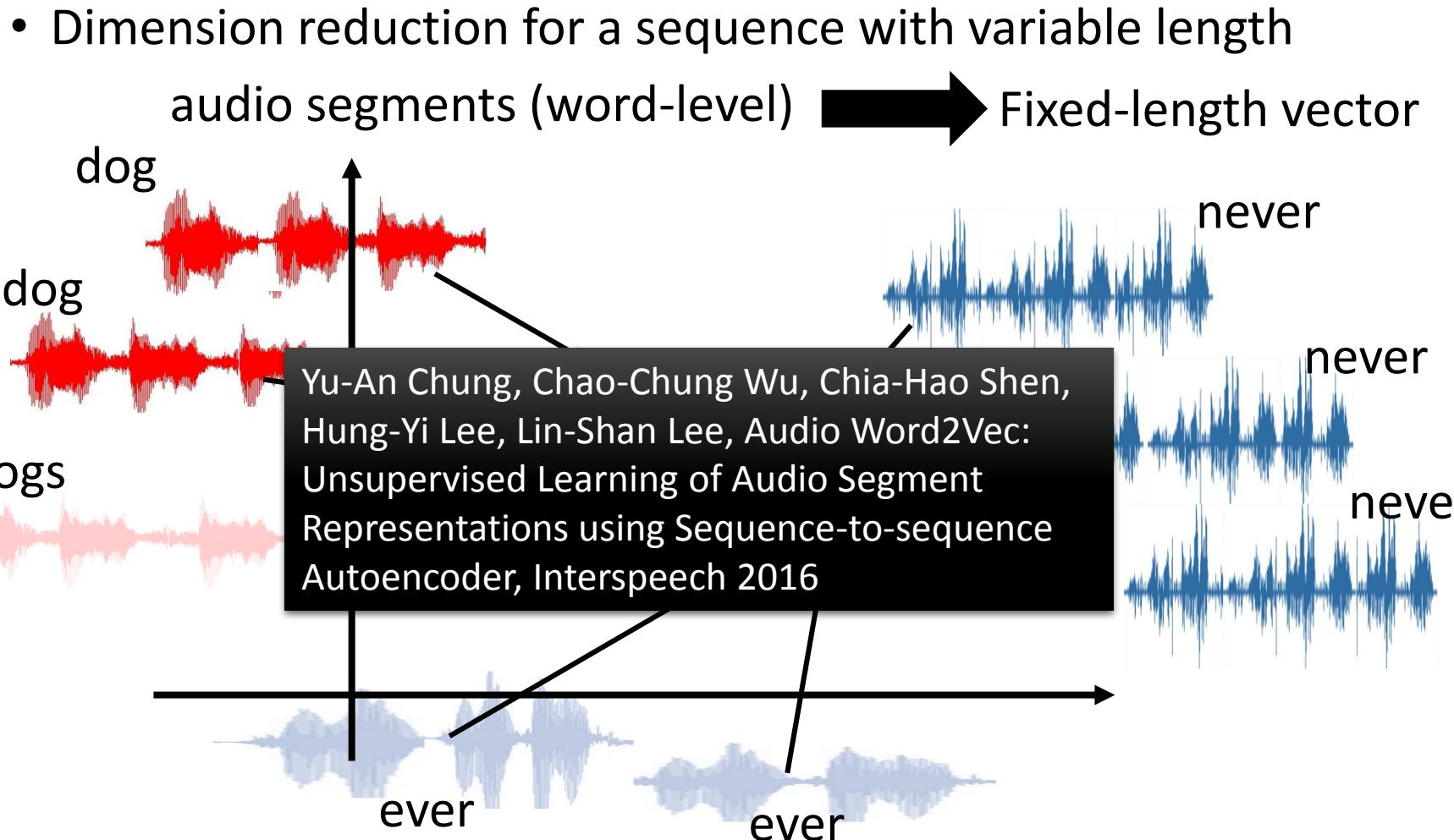
Li, Jiwei, Minh-Thang Luong, and Dan Jurafsky. "A hierarchical neural autoencoder for paragraphs and documents." *arXiv preprint arXiv:1506.01057*(2015).

Sequence-to-sequence Auto-encoder - Text



Li, Jiwei, Minh-Thang Luong, and Dan Jurafsky. "A hierarchical neural autoencoder for paragraphs and documents." *arXiv preprint arXiv:1506.01057*(2015).

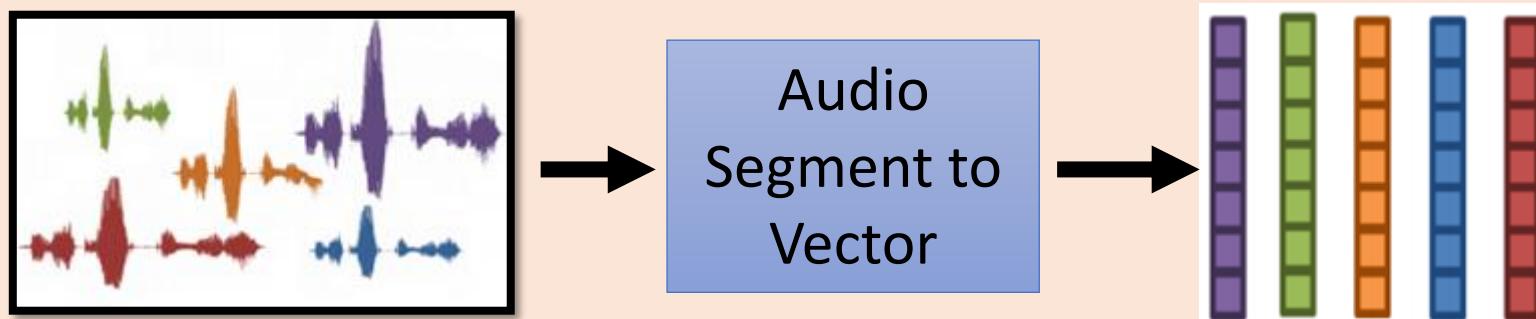
Sequence-to-sequence Auto-encoder - Speech



Sequence-to-sequence Auto-encoder - Speech

Audio archive divided into variable-length audio segments

Off-line



Spoken Query

Audio
Segment to
Vector

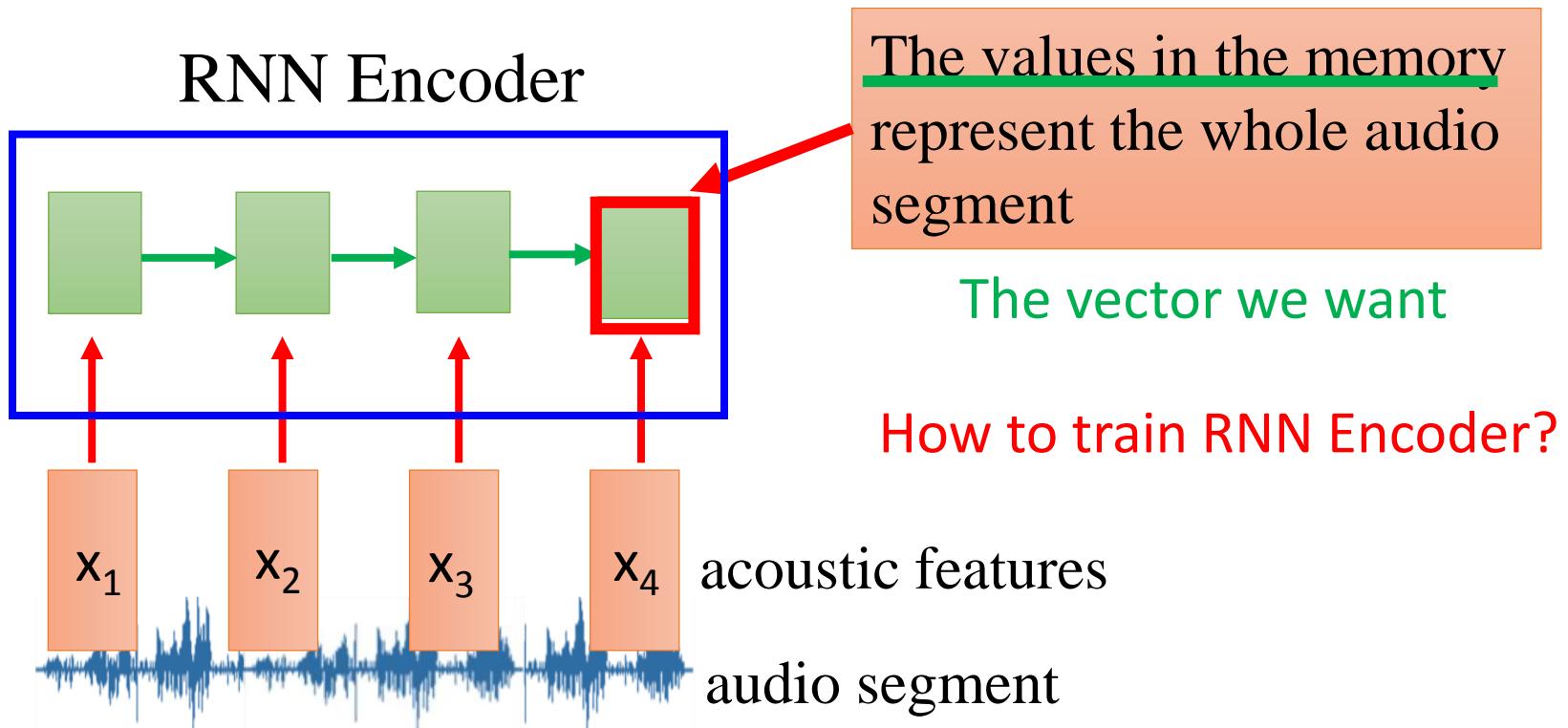


Similarity

Search Result

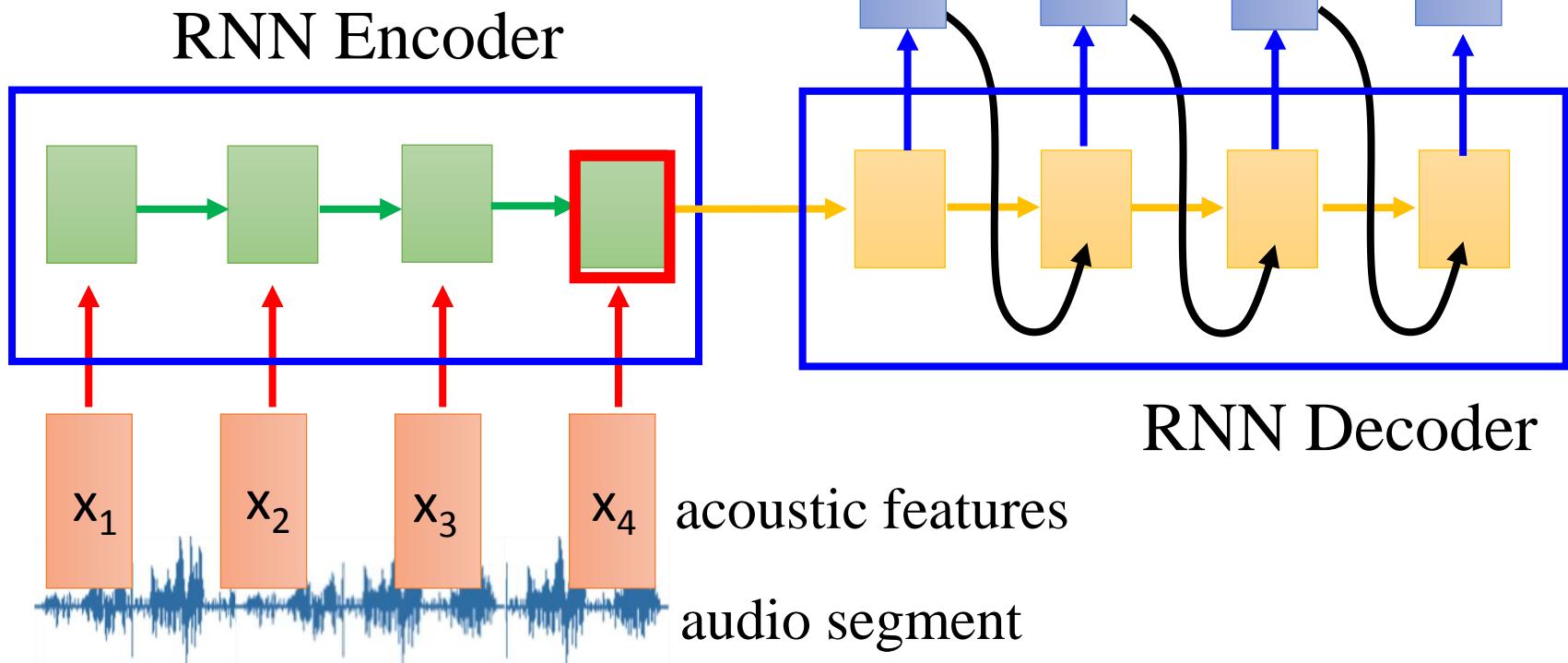
On-line

Sequence-to-sequence Auto-encoder - Speech



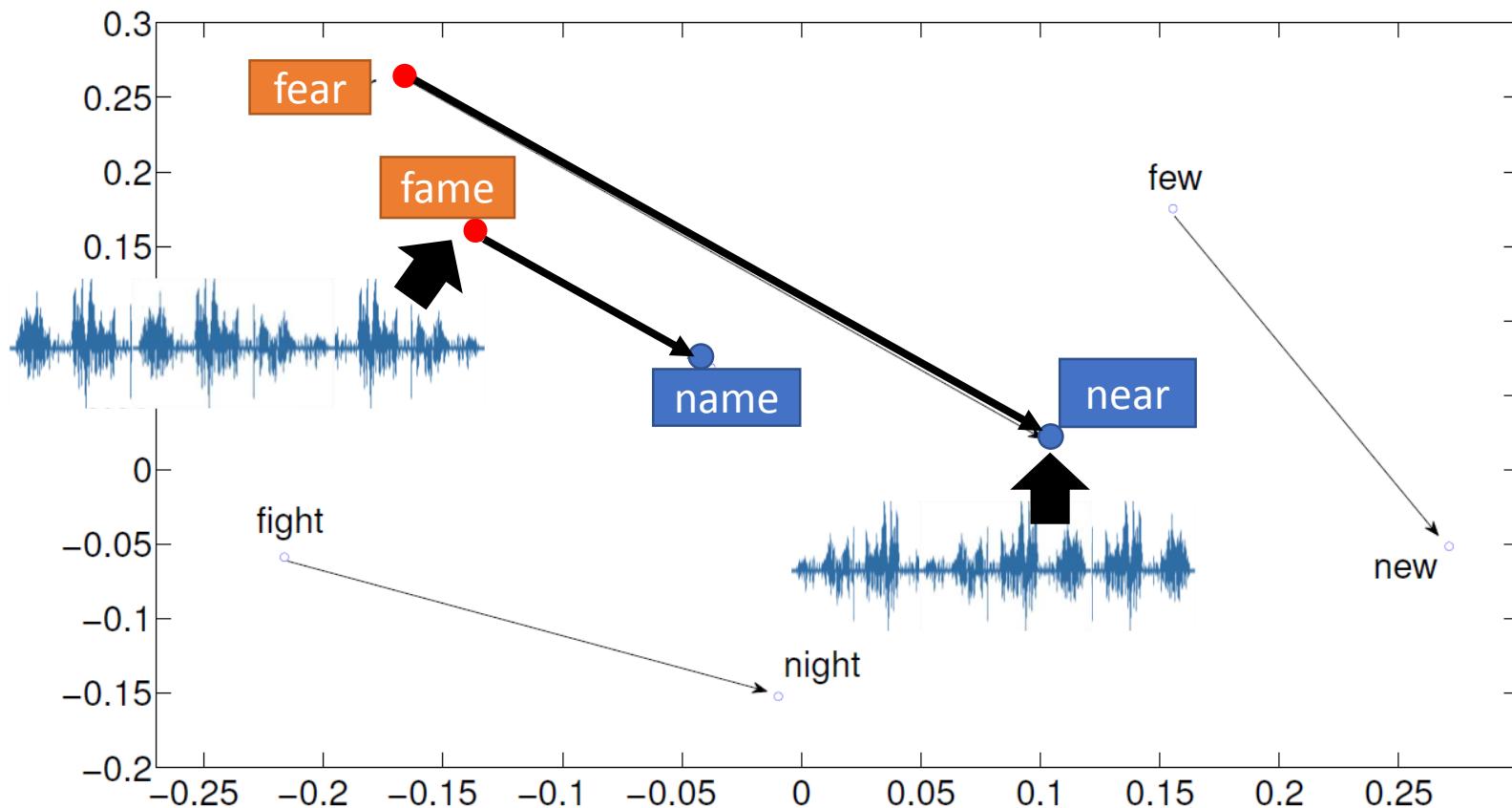
Sequence-to-sequence Auto-encoder

The RNN encoder and
decoder are jointly trained.

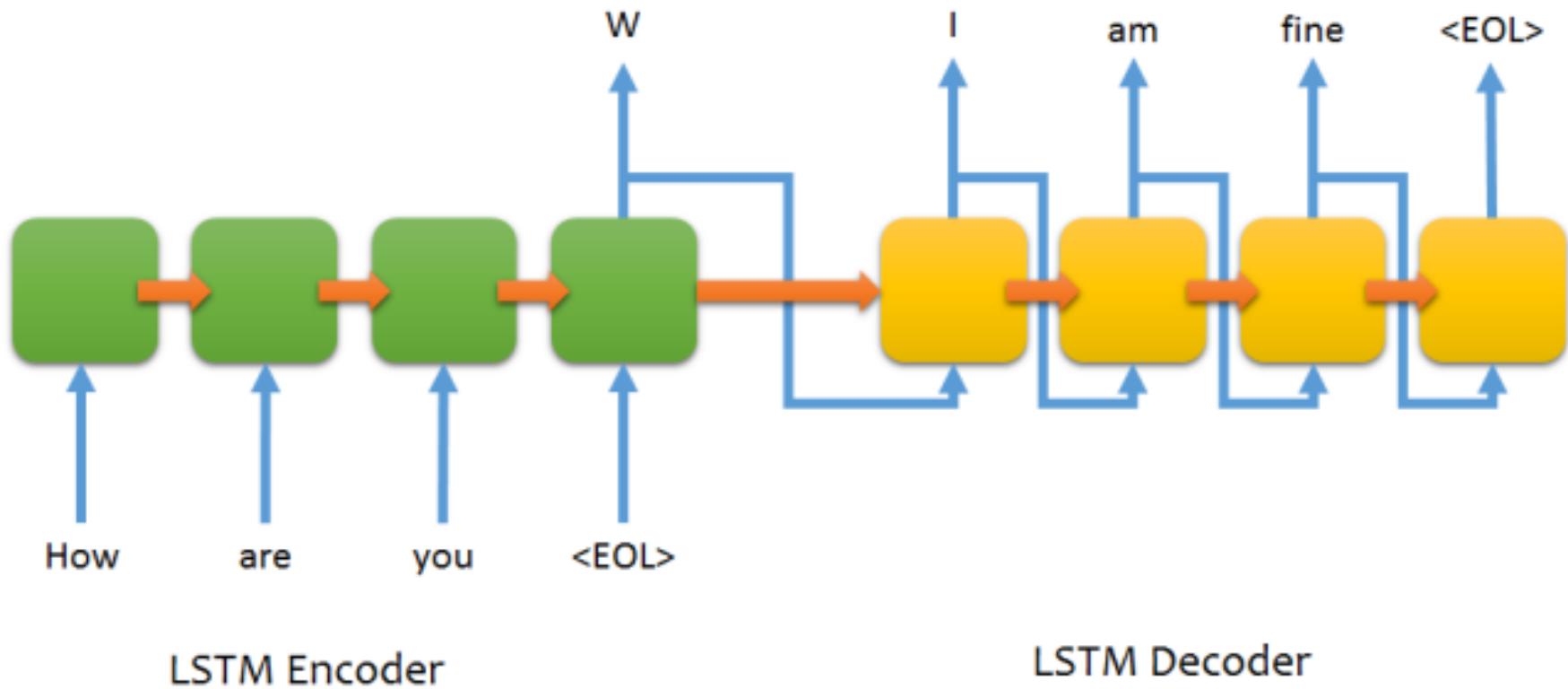


Sequence-to-sequence Auto-encoder - Speech

- Visualizing embedding vectors of the words



Demo: Chat-bot



LSTM Encoder

LSTM Decoder

電視影集 (~40,000 sentences)、美國總統大選辯論

Demo: Chat-bot

- Develop Team
 - Interface design: Prof. Lin-Lin Chen & Arron Lu
 - Web programming: Shi-Yun Huang
 - Data collection: Chao-Chuang Shih
 - System implementation: Kevin Wu, Derek Chuang, & Zhi-Wei Lee (李致緯), Roy Lu (盧柏儒)
 - System design: Richard Tsai & Hung-Yi Lee



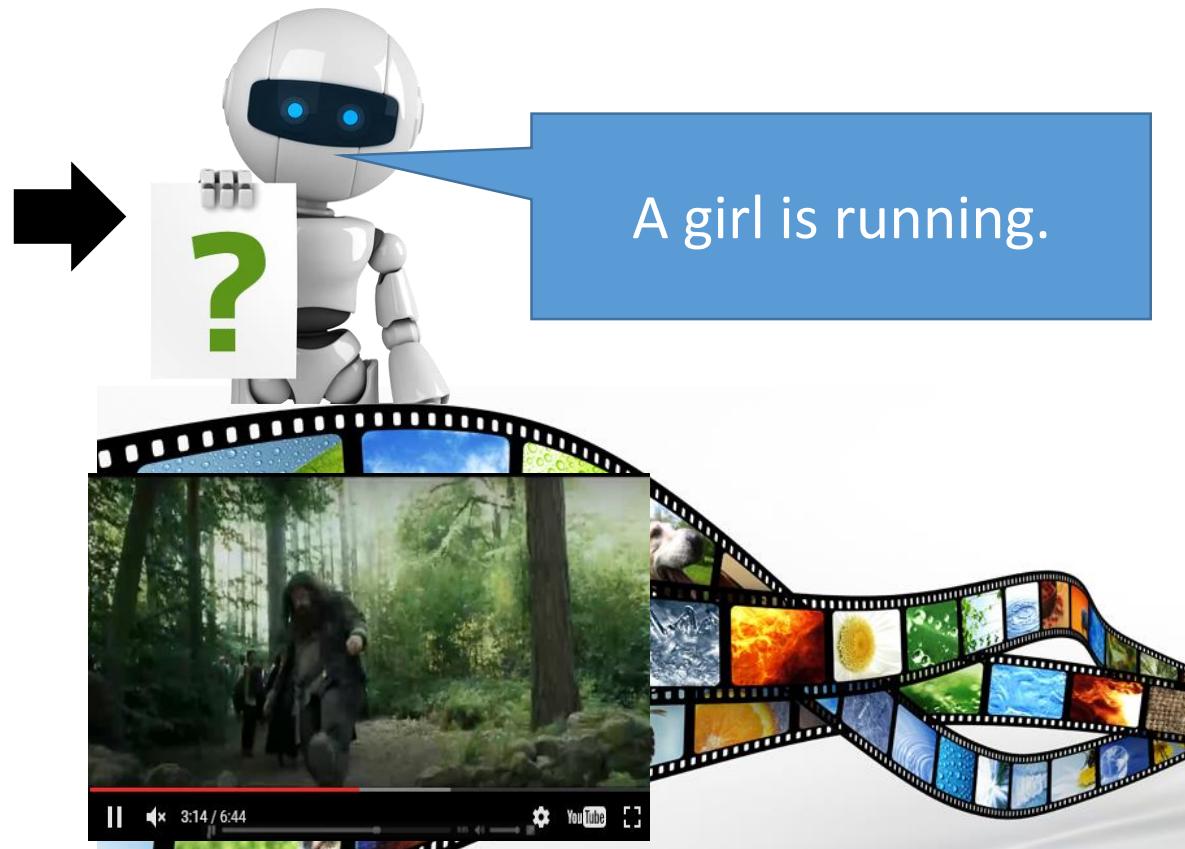
Demo: Video Caption Generation



Video



A group of people is knocked by a tree.



A group of people is walking in the forest.

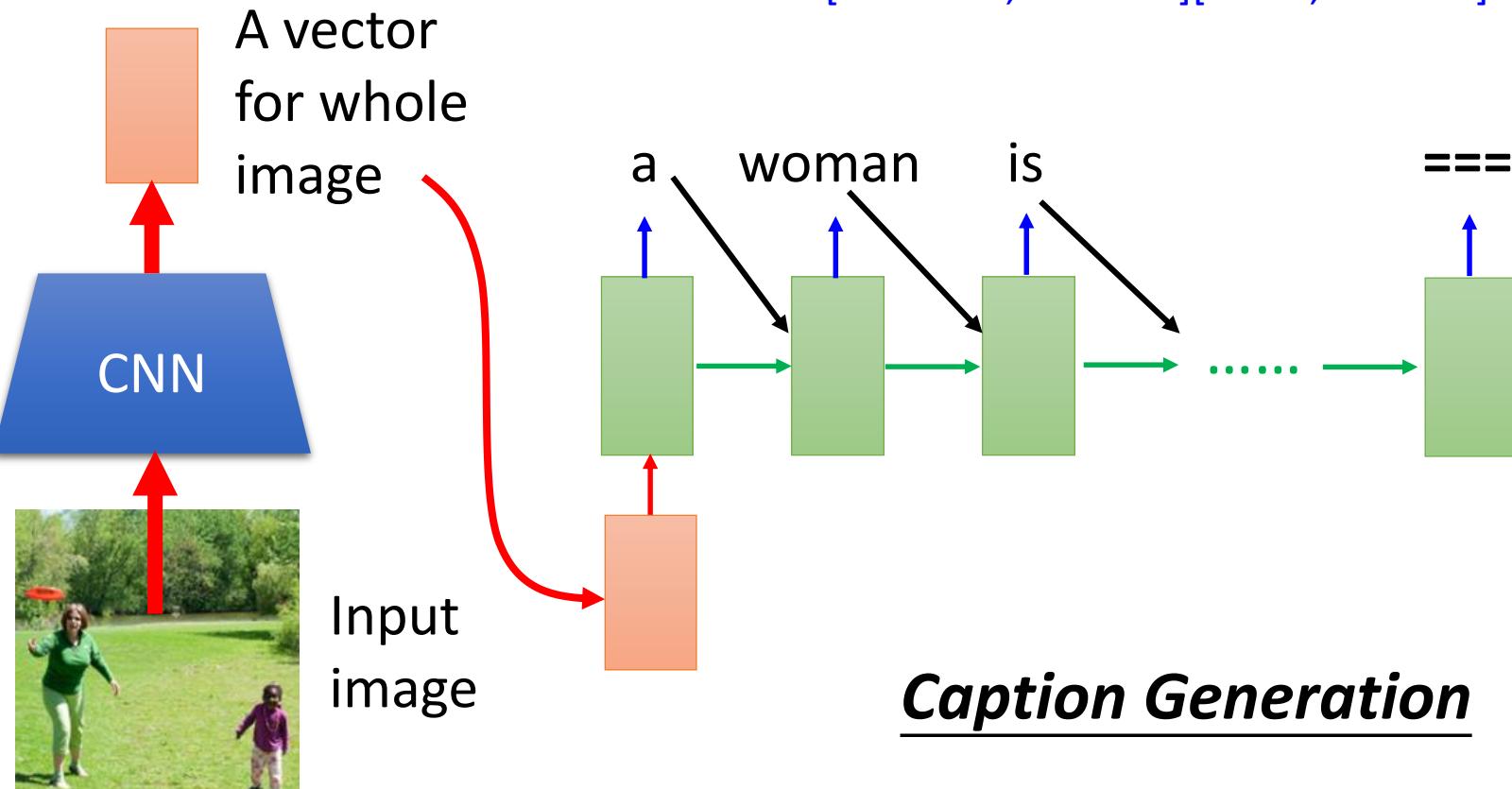
Demo: Video Caption Generation

- Can machine describe what it see from video?
- Demo: 台大語音處理實驗室 曾柏翔、吳柏瑜、盧宏宗
- Video: 莊舜博、楊棋宇、黃邦齊、萬家宏

Demo: Image Caption Generation

- Input an image, but output a sequence of words

[Kelvin Xu, arXiv'15][Li Yao, ICCV'15]



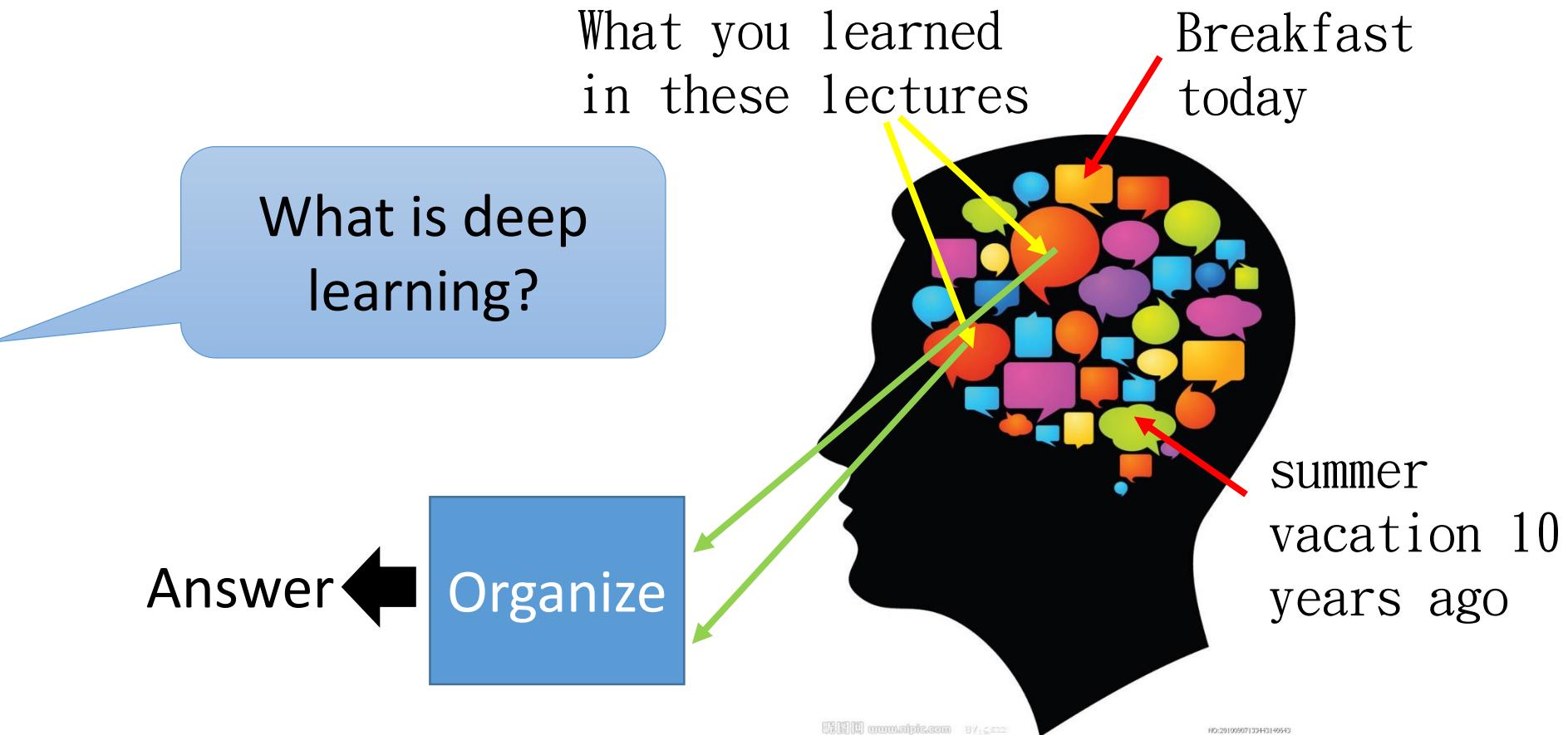
Demo: Image Caption Generation

- Can machine describe what it see from image?
- Demo: 台大電機系 大四 蘇子睿、林奕辰、徐翊祥、陳奕安

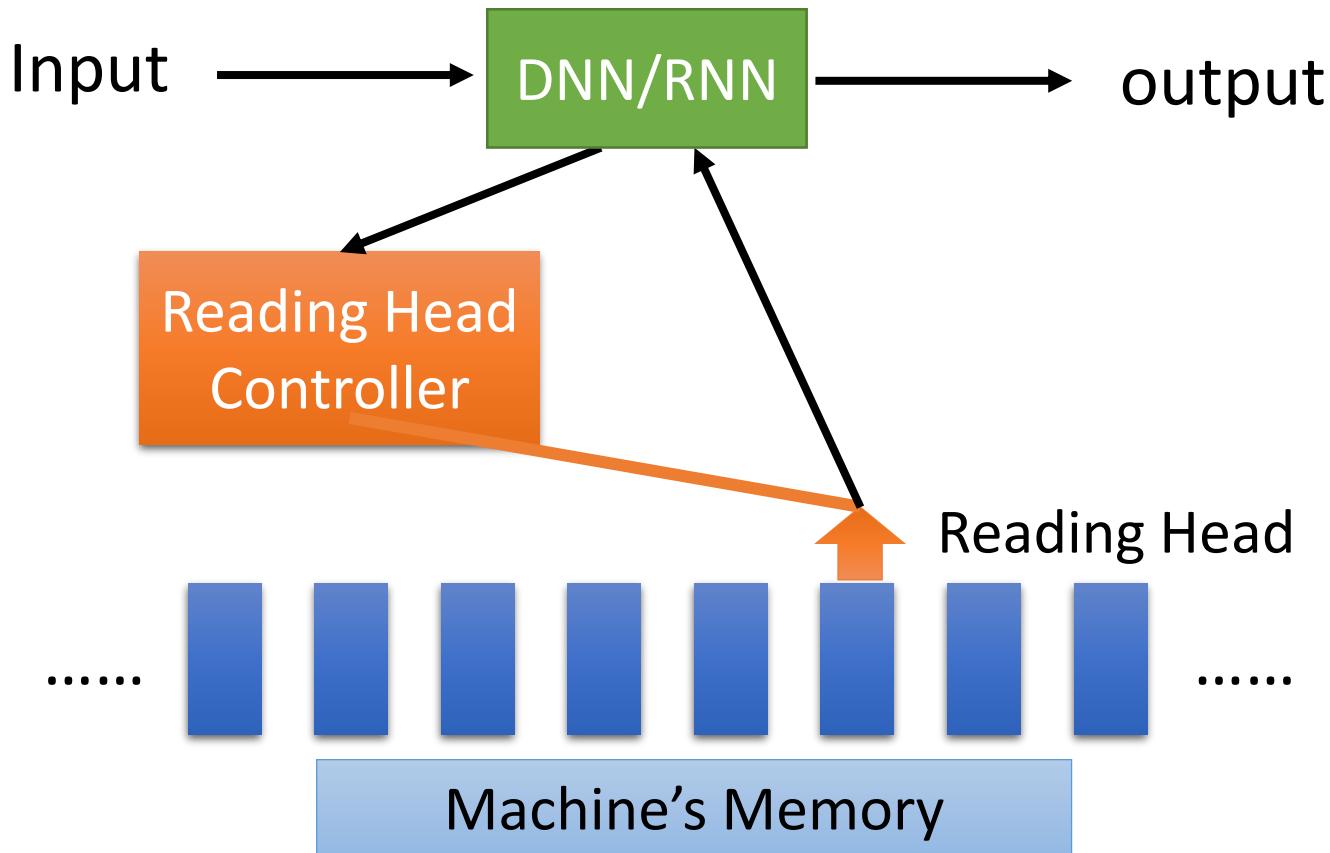
http://news.ltn.com.tw/photo/politics/breakingnews/975542_1



Attention-based Model



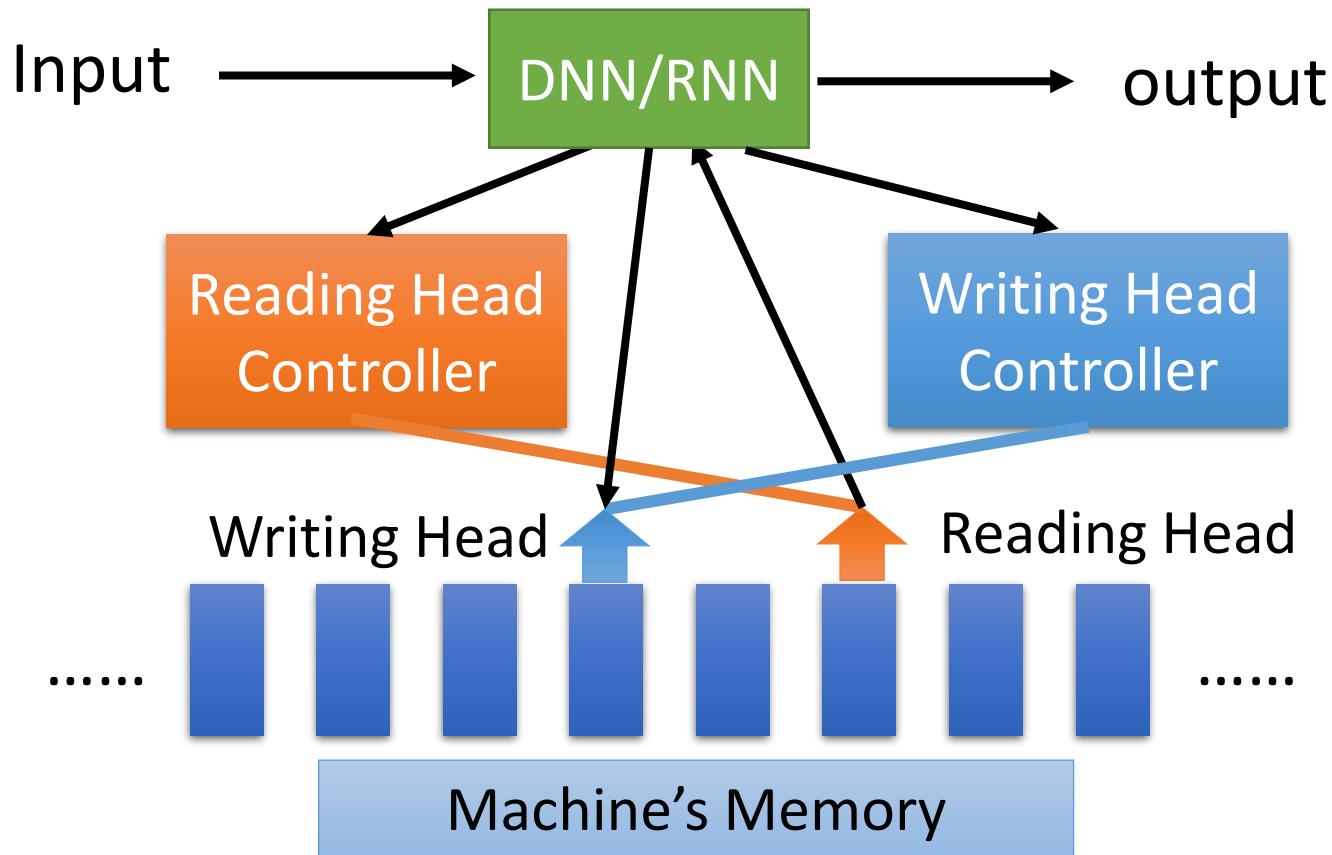
Attention-based Model



Ref:

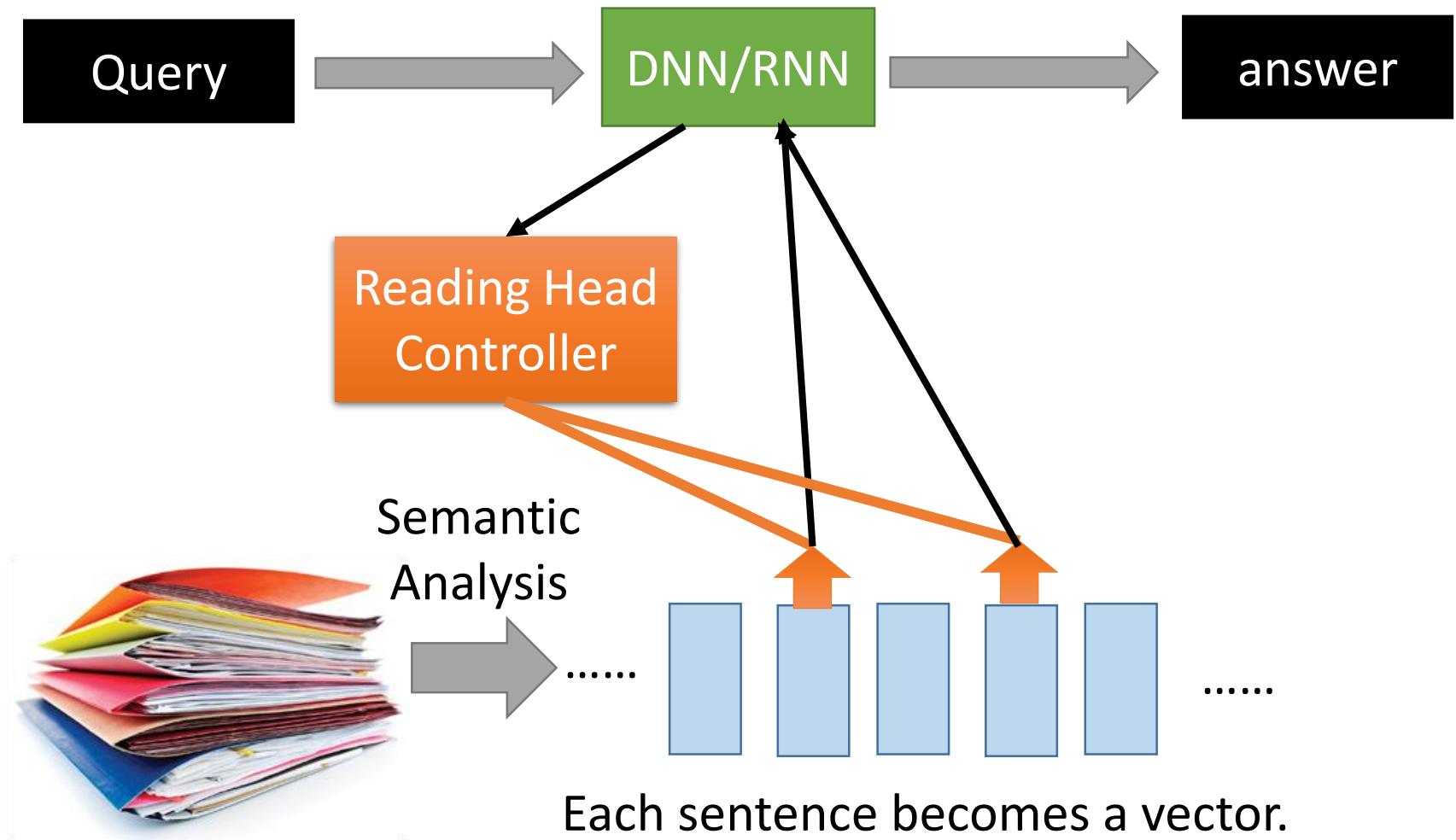
[http://speech.ee.ntu.edu.tw/~tlkagk/courses/MLDS_2015_2/Lecture/Attain%20\(v3\).ecm.mp4/index.html](http://speech.ee.ntu.edu.tw/~tlkagk/courses/MLDS_2015_2/Lecture/Attain%20(v3).ecm.mp4/index.html)

Attention-based Model v2



Neural Turing Machine

Reading Comprehension



Reading Comprehension

- End-To-End Memory Networks. S. Sukhbaatar, A. Szlam, J. Weston, R. Fergus. NIPS, 2015.

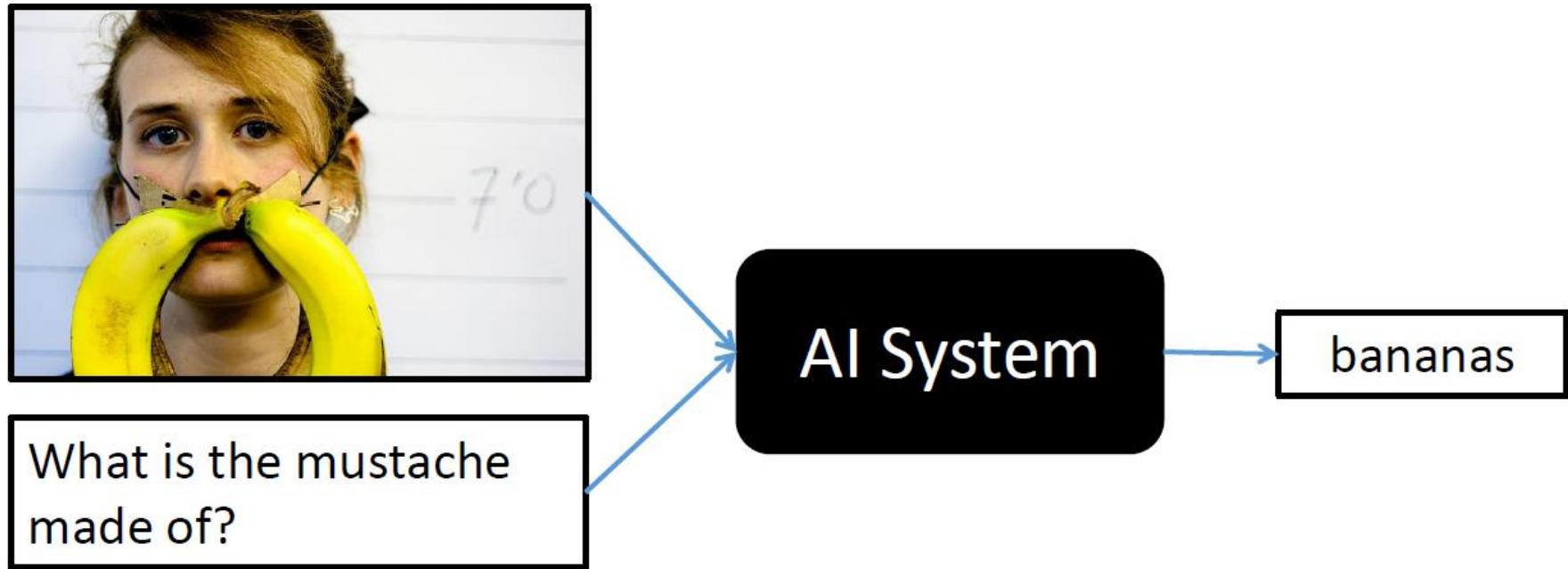
The position of reading head:

Story (16: basic induction)	Support	Hop 1	Hop 2	Hop 3
Brian is a frog.	yes	0.00	0.98	0.00
Lily is gray.		0.07	0.00	0.00
Brian is yellow.	yes	0.07	0.00	1.00
Julius is green.		0.06	0.00	0.00
Greg is a frog.	yes	0.76	0.02	0.00
What color is Greg? Answer: yellow		Prediction: yellow		

Keras has example:

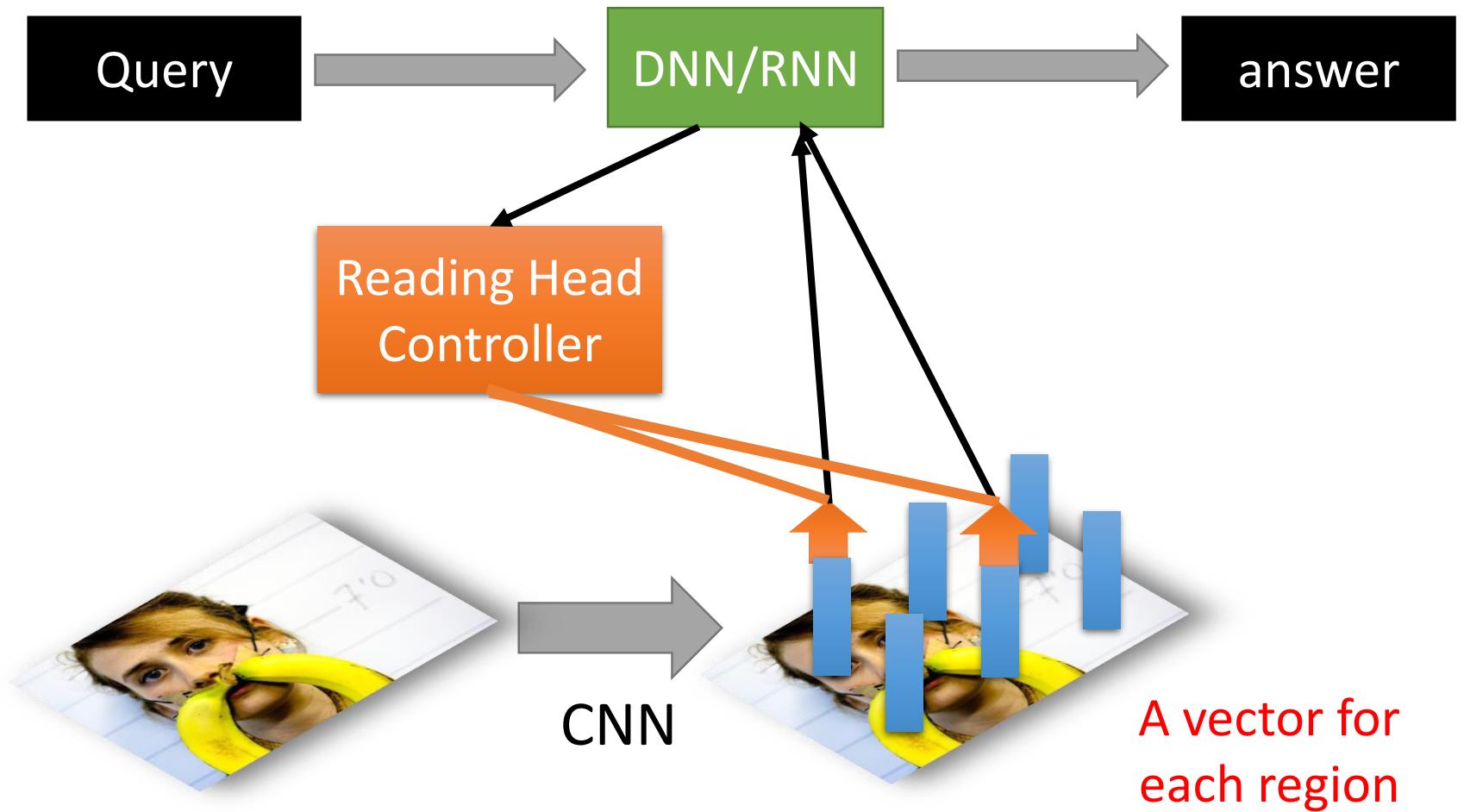
https://github.com/fchollet/keras/blob/master/examples/babi_memnn.py

Visual Question Answering



source: <http://visualqa.org/>

Visual Question Answering



Speech Question Answering

- **TOEFL Listening Comprehension Test by Machine**
- Example:

Audio Story:  (The original story is 5 min long.)

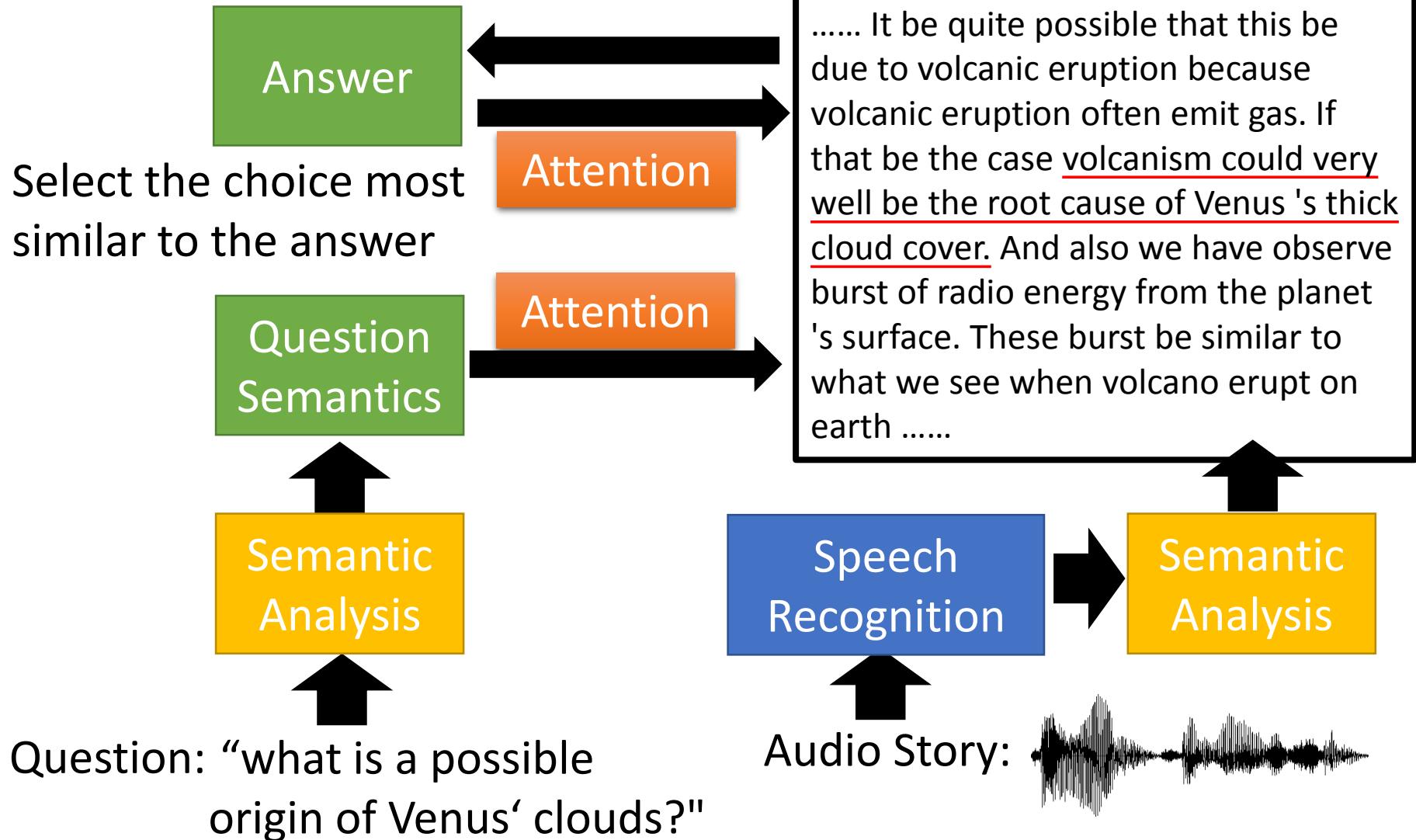
Question: “What is a possible origin of Venus’ clouds?”

Choices:

- (A) gases released as a result of volcanic activity
- (B) chemical reactions caused by high surface temperatures
- (C) bursts of radio energy from the planet's surface
- (D) strong winds that blow dust into the atmosphere

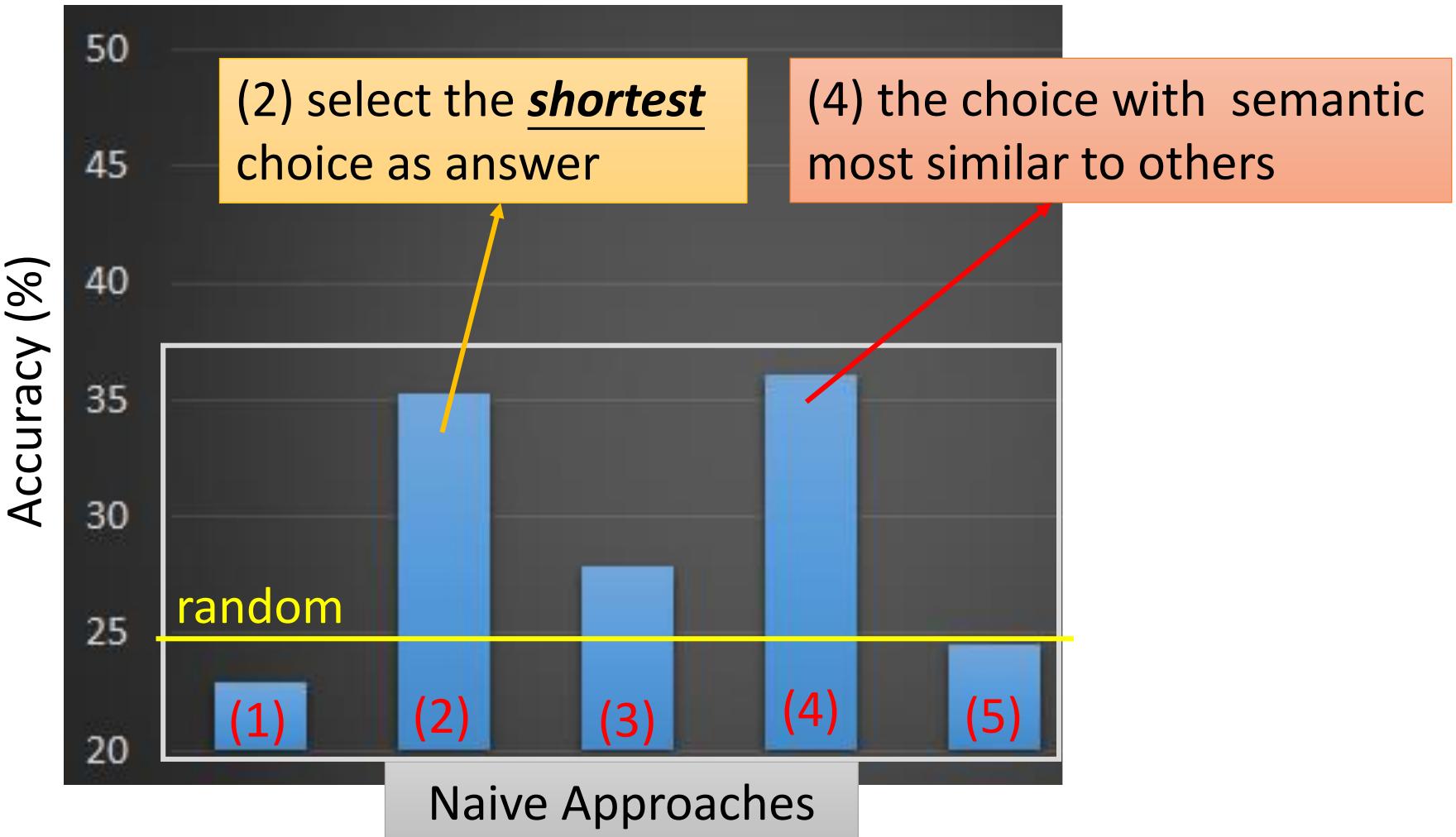
Model Architecture

Everything is learned from training examples

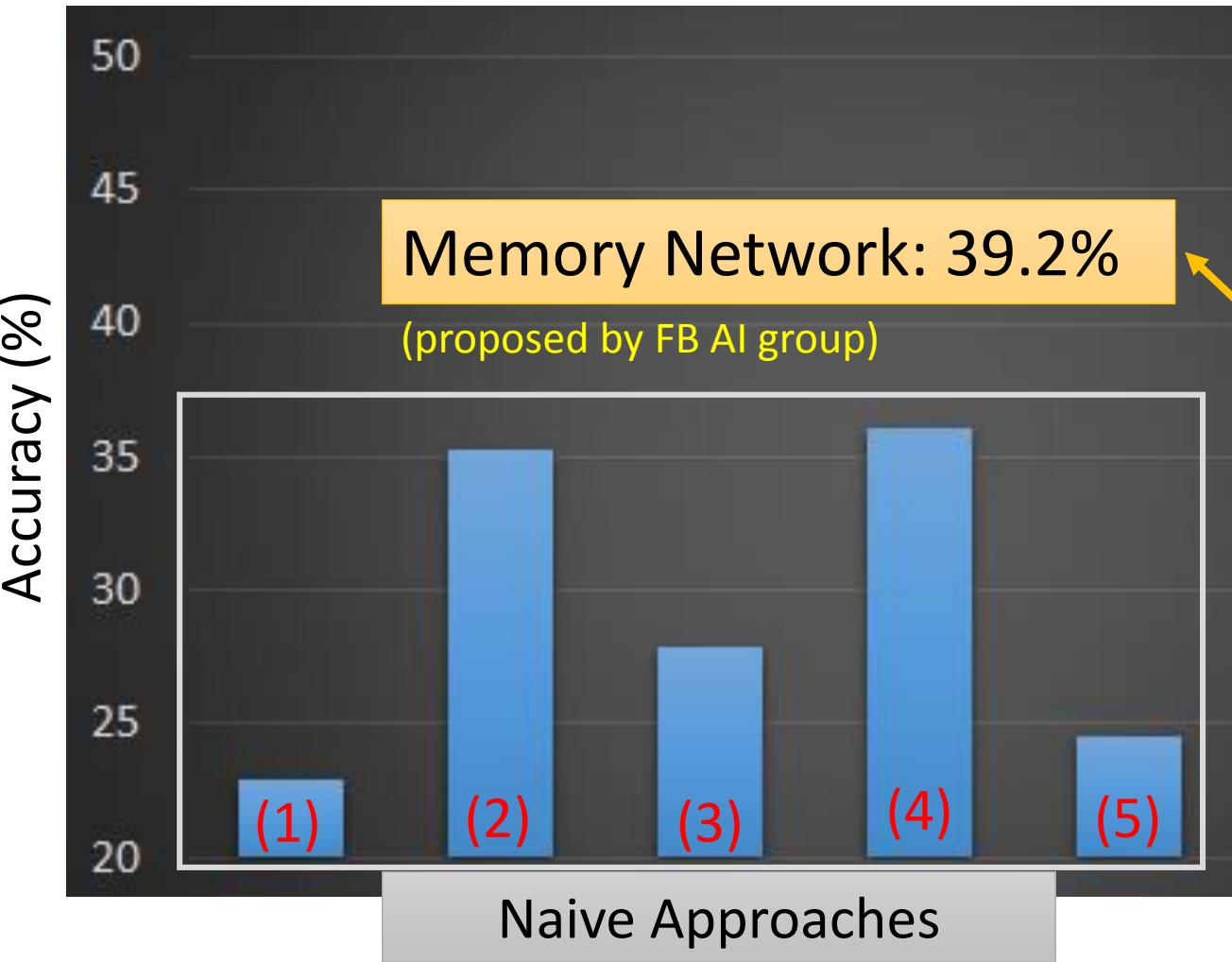


Simple Baselines

Experimental setup:
717 for training,
124 for validation, 122 for testing

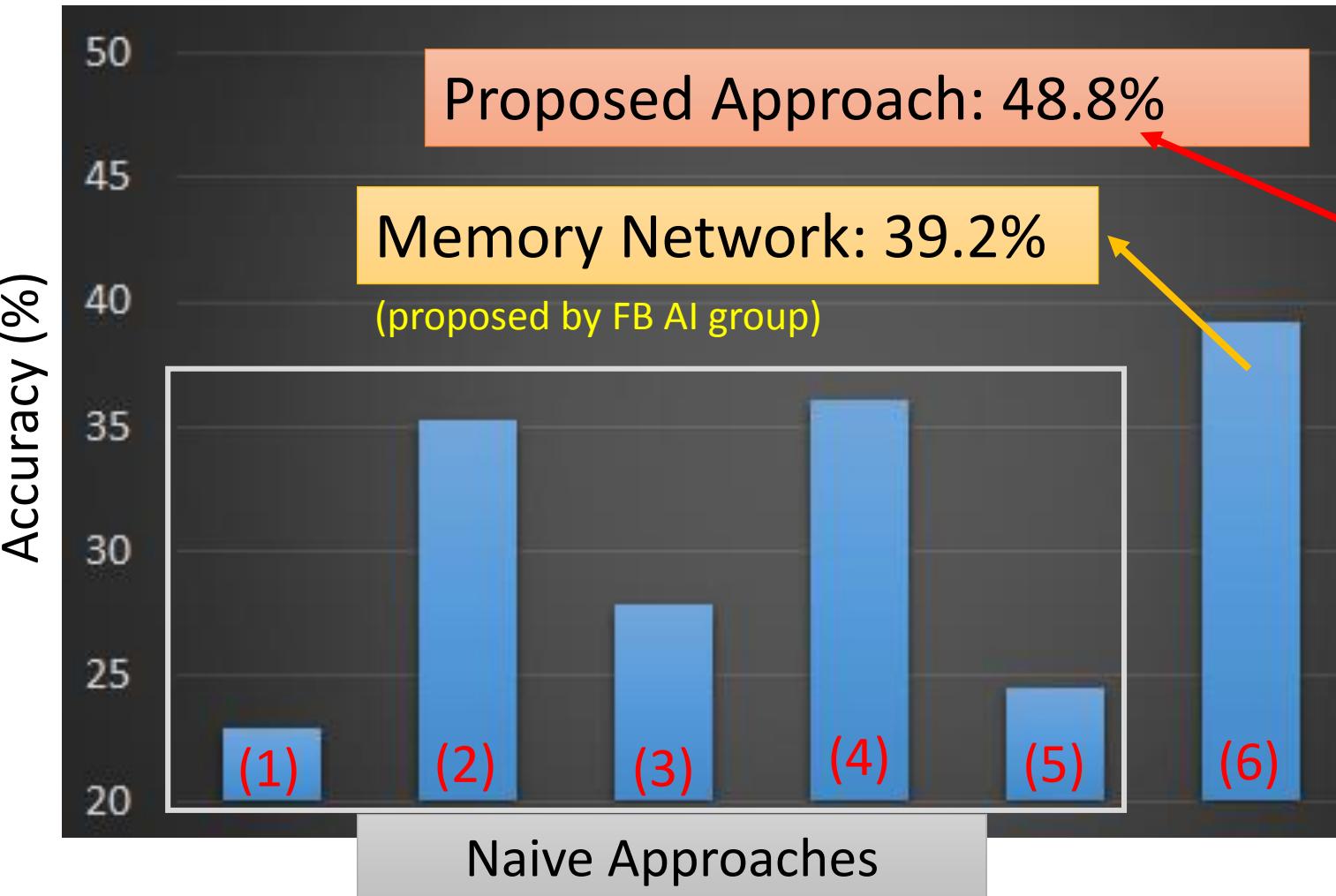


Memory Network



Proposed Approach

[Tseng & Lee, Interspeech 16]
[Fang & Hsu & Lee, SLT 16]



To Learn More

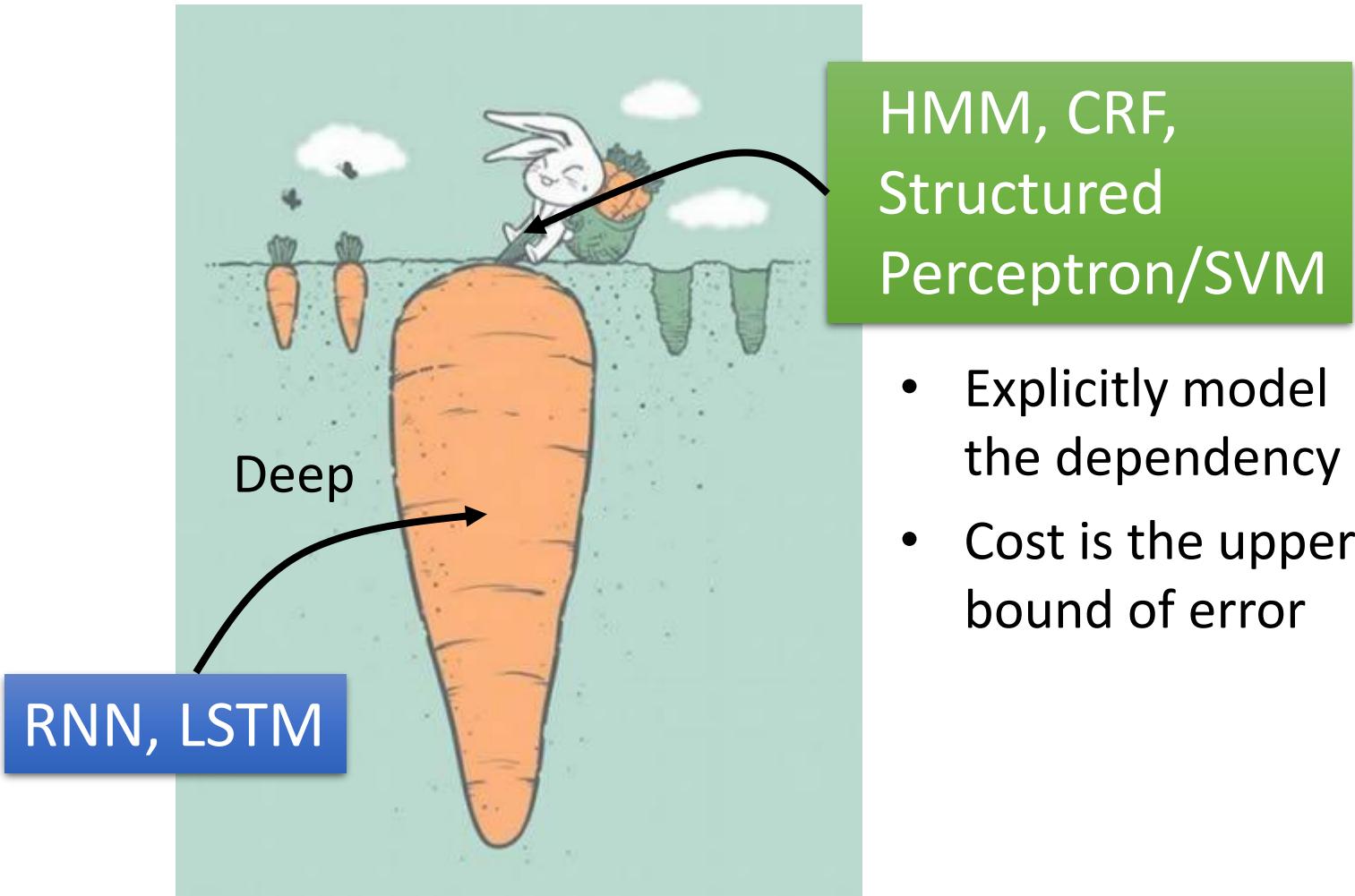
- The Unreasonable Effectiveness of Recurrent Neural Networks
 - <http://karpathy.github.io/2015/05/21/rnn-effectiveness/>
- Understanding LSTM Networks
 - <http://colah.github.io/posts/2015-08-Understanding-LSTMs/>

Deep & Structured

RNN v.s. Structured Learning

- RNN, LSTM
 - Unidirectional RNN does not consider the whole sequence
 - Cost and error not always related
 - Deep 
- HMM, CRF, Structured Perceptron/SVM
 - Using Viterbi, so consider the whole sequence 
 - How about Bidirectional RNN?
 - Can explicitly consider the label dependency 
 - Cost is the upper bound of error 

Integrated Together

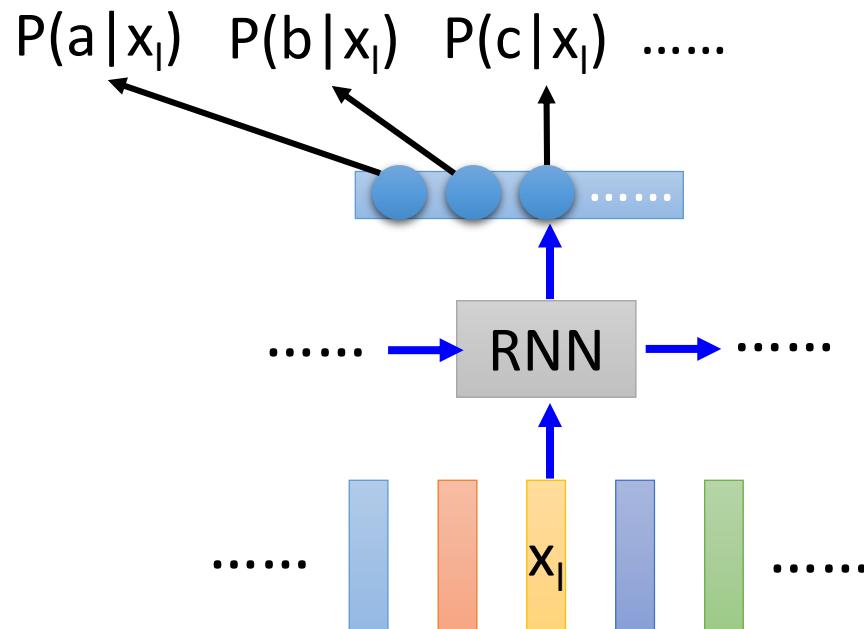


- Explicitly model the dependency
- Cost is the upper bound of error

Integrated together

- Speech Recognition: CNN/LSTM/DNN + HMM

$$P(x, y) = P(y_1 | start) \prod_{l=1}^{L-1} P(y_{l+1} | y_l) P(end | y_L) \prod_{l=1}^L P(x_l | y_l)$$



$$P(x_l | y_l) = \frac{P(x_l, y_l)}{P(y_l)}$$

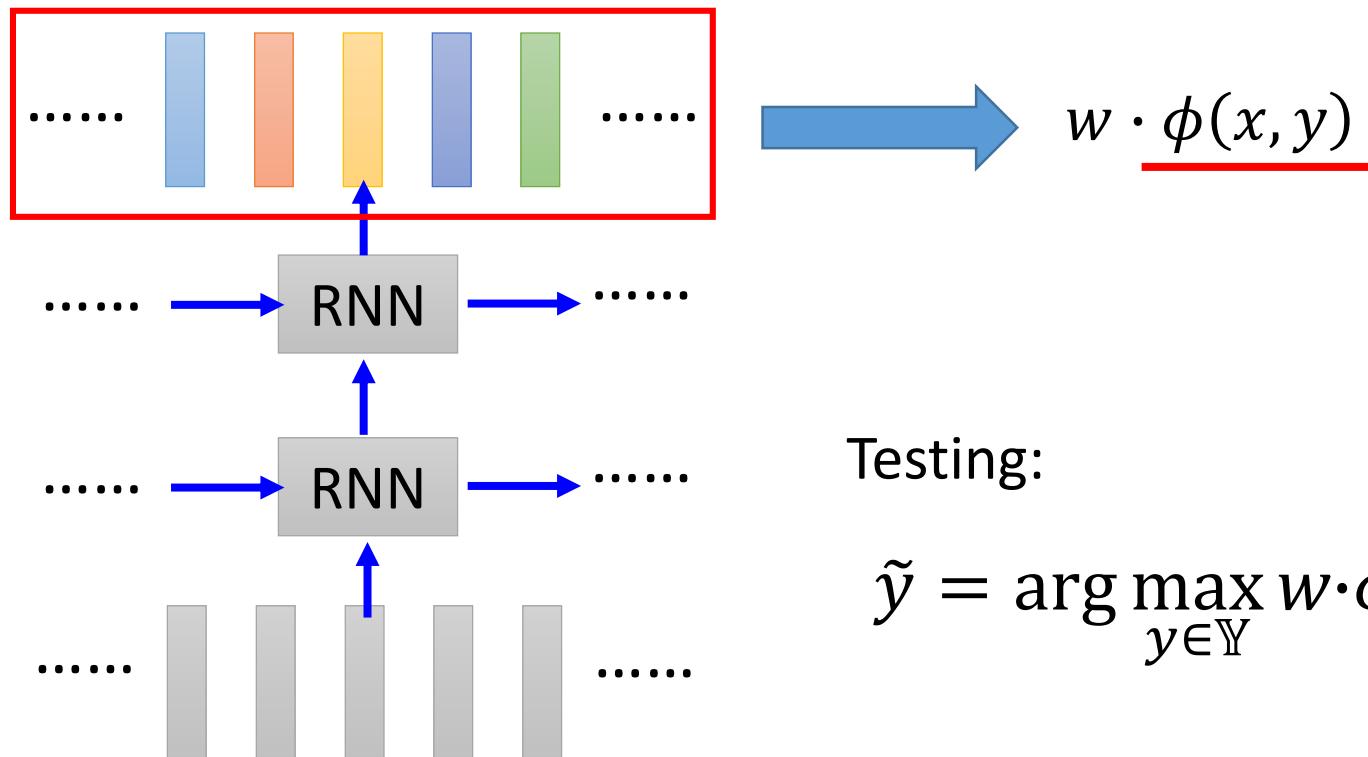
RNN

$$= \frac{P(y_l | x_l) \cancel{P(x_l)}}{P(y_l)}$$

Count

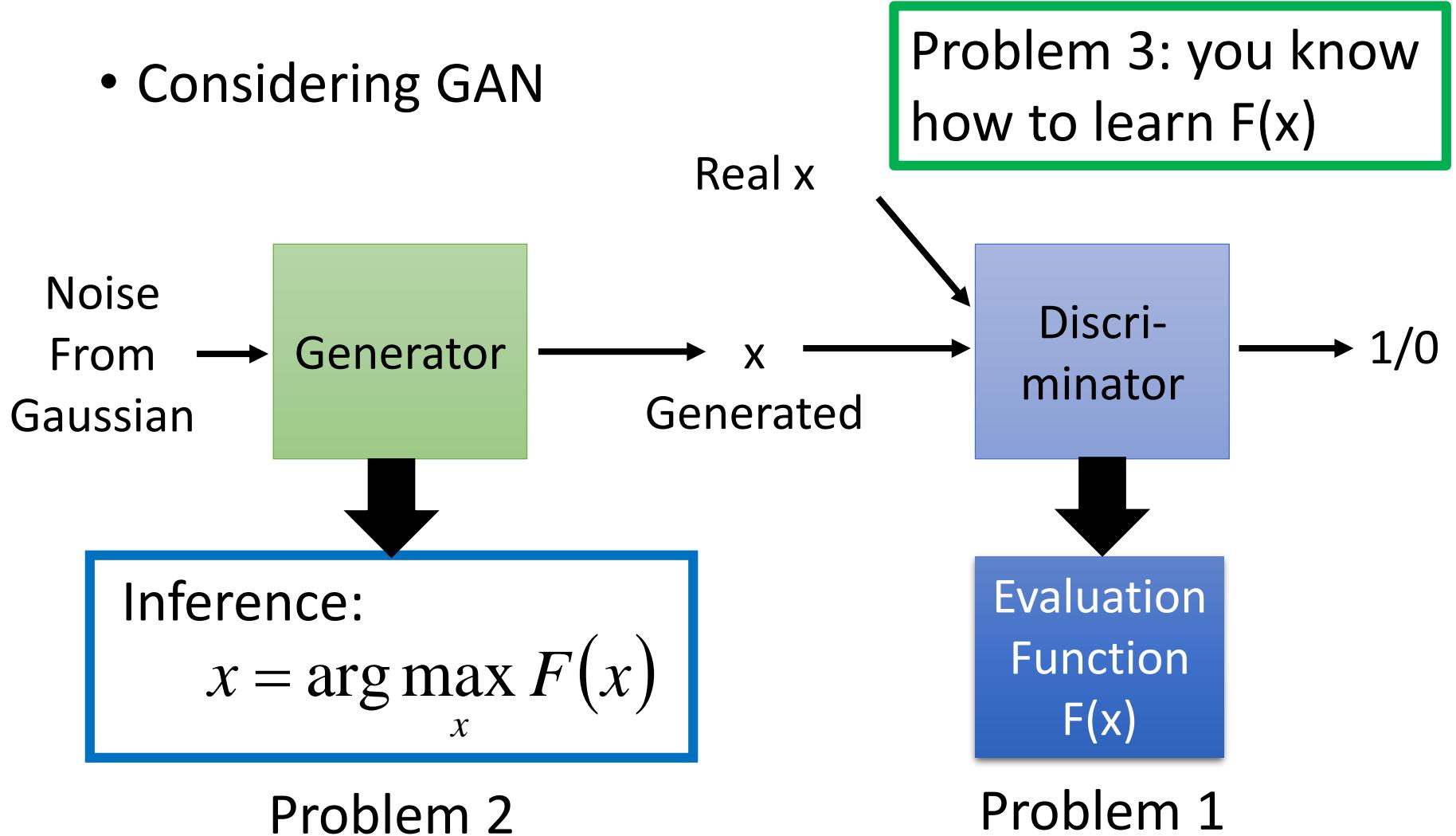
Integrated together

- Semantic Tagging: Bi-directional LSTM + CRF/Structured SVM



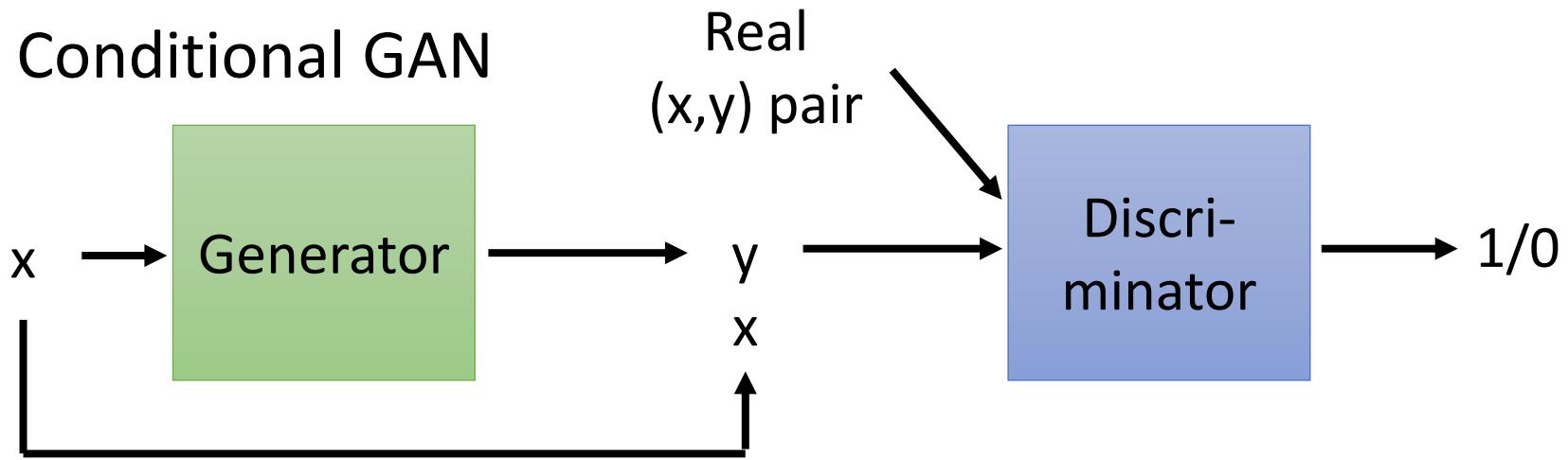
Is structured learning practical?

- Considering GAN

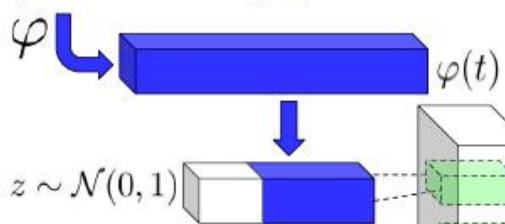


Is structured learning practical?

- Conditional GAN



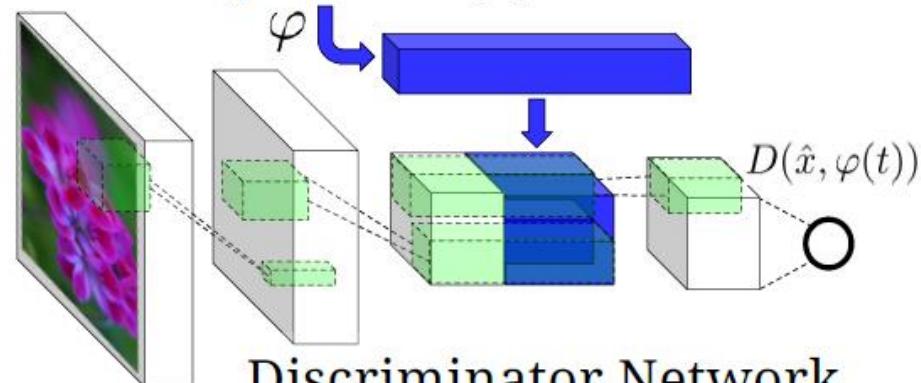
This flower has small, round violet petals with a dark purple center



Generator Network

$$\hat{x} := G(z, \varphi(t))$$

This flower has small, round violet petals with a dark purple center



Discriminator Network

Deep and Structured
will be the future.

Sounds crazy?

People do think in this way ...

- Connect Energy-based model with GAN:
 - A Connection Between Generative Adversarial Networks, Inverse Reinforcement Learning, and Energy-Based Models
 - Deep Directed Generative Models with Energy-Based Probability Estimation
 - ENERGY-BASED GENERATIVE ADVERSARIAL NETWORKS
- Deep learning model for inference
 - Deep Unfolding: Model-Based Inspiration of Novel Deep Architectures
 - Conditional Random Fields as Recurrent Neural Networks

Machine learning and having it deep and structured (MLDS)

- 和 ML 的不同
 - 在這學期 ML 中有提過的內容 (DNN, CNN ...) , 在 MLDS 中不再重複，只做必要的復習
- 教科書：“Deep Learning”
(<http://www.deeplearningbook.org/>)
 - Part II 是講 deep learning 、 Part III 就是講 structured learning

- Part II: Modern Practical Deep Networks
 - 6 Deep Feedforward Networks
 - 7 Regularization for Deep Learning
 - 8 Optimization for Training Deep Models
 - 9 Convolutional Networks
 - 10 Sequence Modeling: Recurrent and Recurrent Models
 - 11 Practical Methodology
 - 12 Applications

- Part III: Deep Learning Research
 - 13 Linear Factor Models
 - 14 Autoencoders
 - 15 Representation Learning
 - 16 Structured Probabilistic Models for Deep Learning
 - 17 Monte Carlo Methods
 - 18 Confronting the Partition Function
 - 19 Approximate Inference
 - 20 Deep Generative Models

Machine learning and having it deep and structured (MLDS)

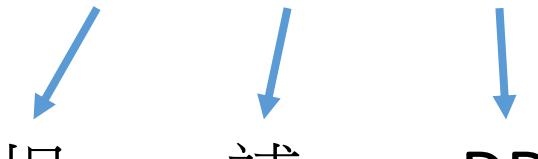
- 所有作業都 2 ~ 4 人一組，可以先組好隊後一起來修
- MLDS 的作業和之前不同
 - RNN (把之前 MLDS 的三個作業合為一個)、Attention-based model 、Deep Reinforcement Learning 、Deep Generative Model 、Sequence-to-sequence learning
- MLDS 初選不開放加簽，以組為單位加簽，作業0的內容是做一個 DNN （可用現成套件）

Ensemble

Introduction

- We are almost at the end of the semester/final competition.
 - <https://inclass.kaggle.com/c/ml2016-cyber-security-attack-defender/leaderboard>
 - <https://www.kaggle.com/c/outbrain-click-prediction/leaderboard>
 - <https://www.kaggle.com/c/transfer-learning-on-stack-exchange-tags/leaderboard>
- You already developed some algorithms and codes.
Lazy to modify them.
- Ensemble: improving your machine with little modification

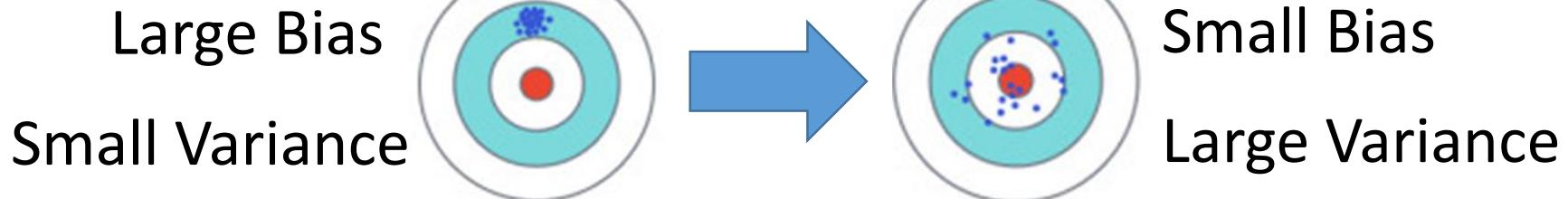
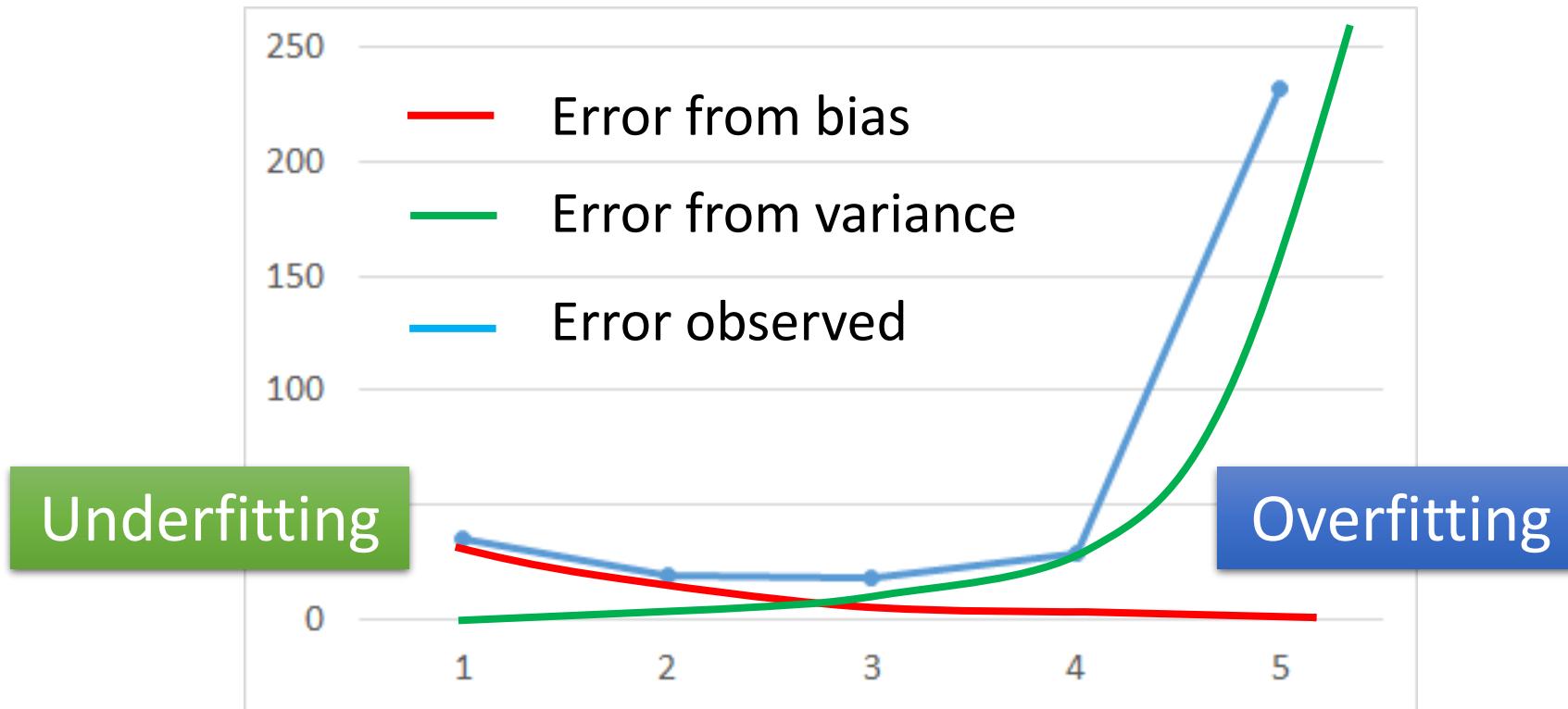
Framework of Ensemble

- Get a set of classifiers
 - $f_1(x), f_2(x), f_3(x), \dots$ 

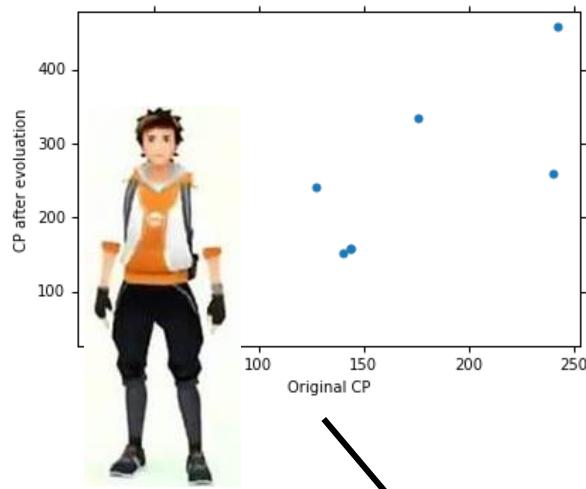
坦 補 DD They should be diverse.
- Aggregate the classifiers (*properly*)
 - 在打王時每個人都有該站的位置

Ensemble: Bagging

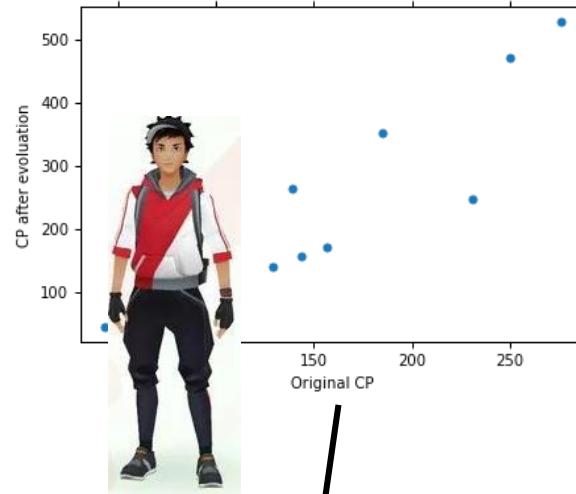
Review: Bias v.s. Variance



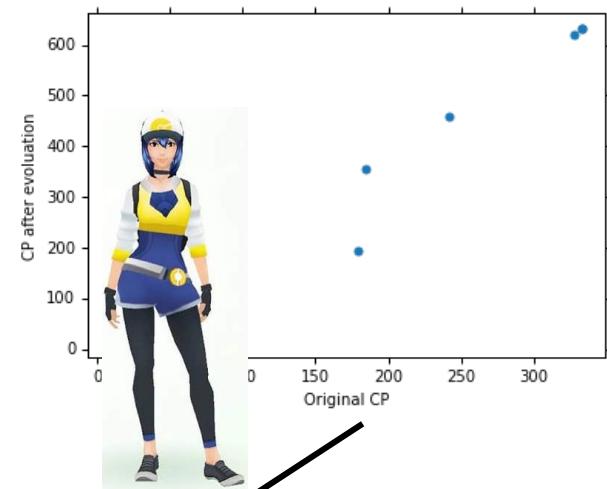
Universe 1



Universe 2

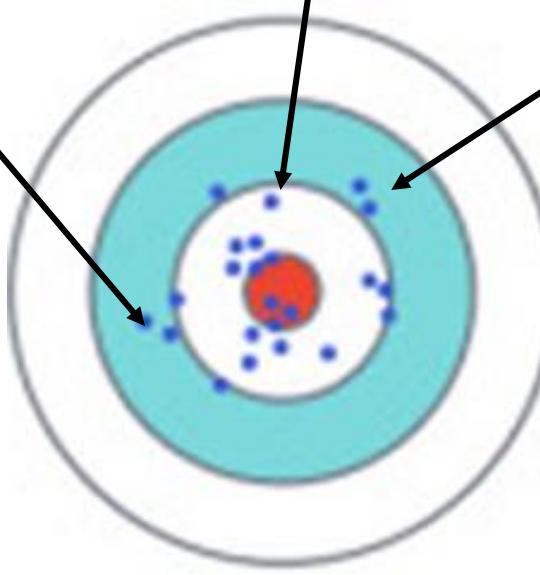


Universe 3



A complex model will have large variance.

We can average complex models to reduce variance.



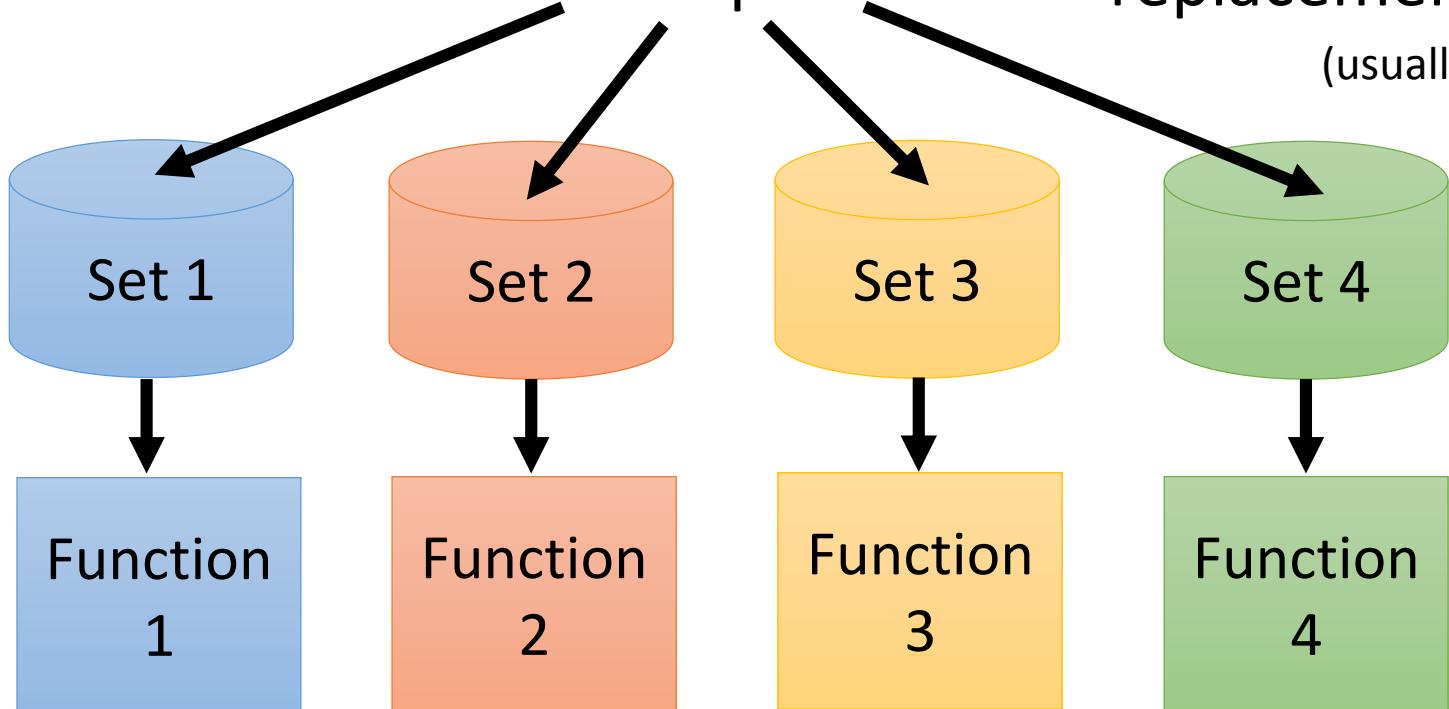
If we average all the f^* , is it close to \hat{f}

$$E[f^*] = \hat{f}$$

Bagging

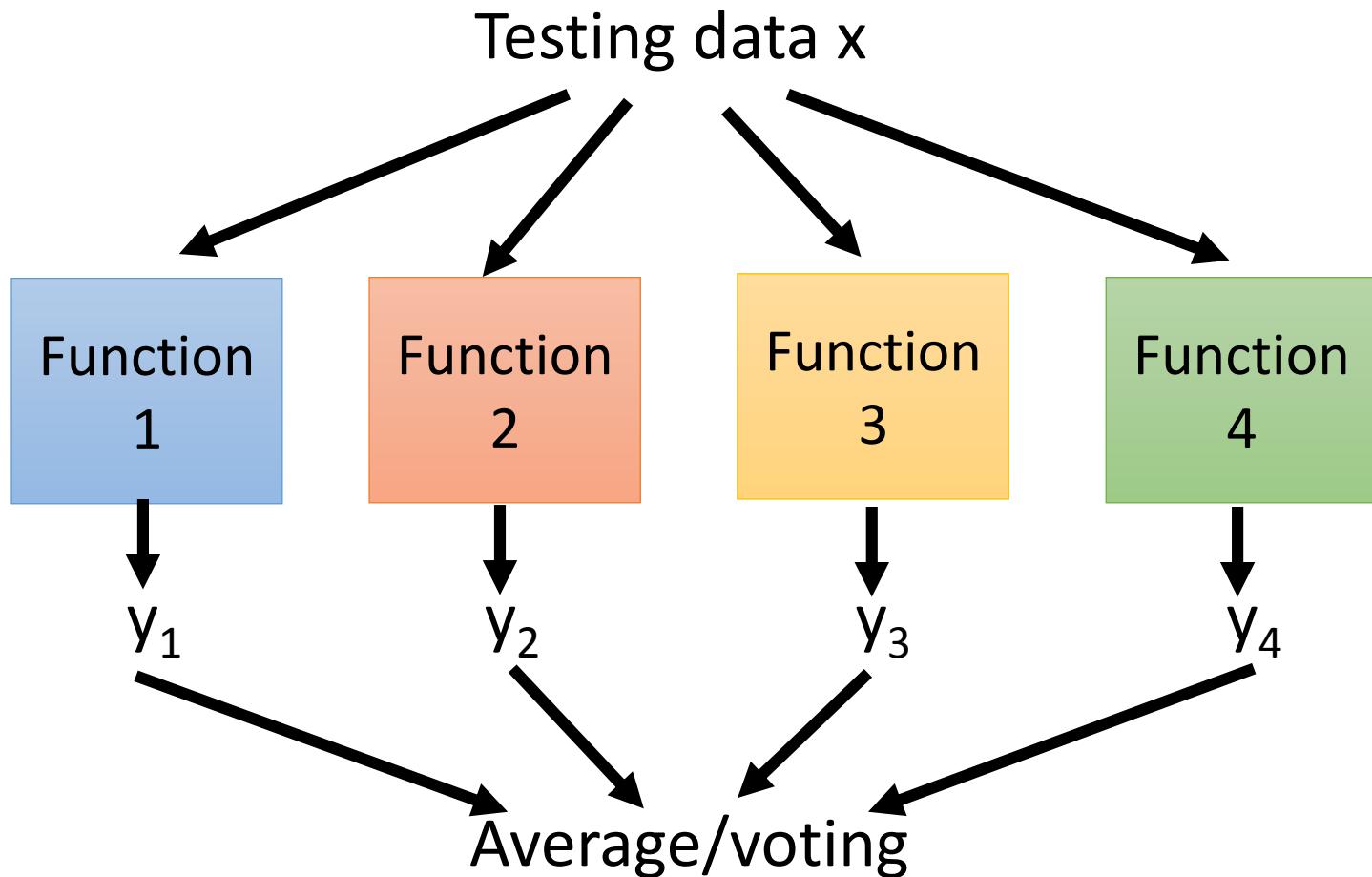
N training
examples

Sampling N'
examples with
replacement
(usually $N=N'$)

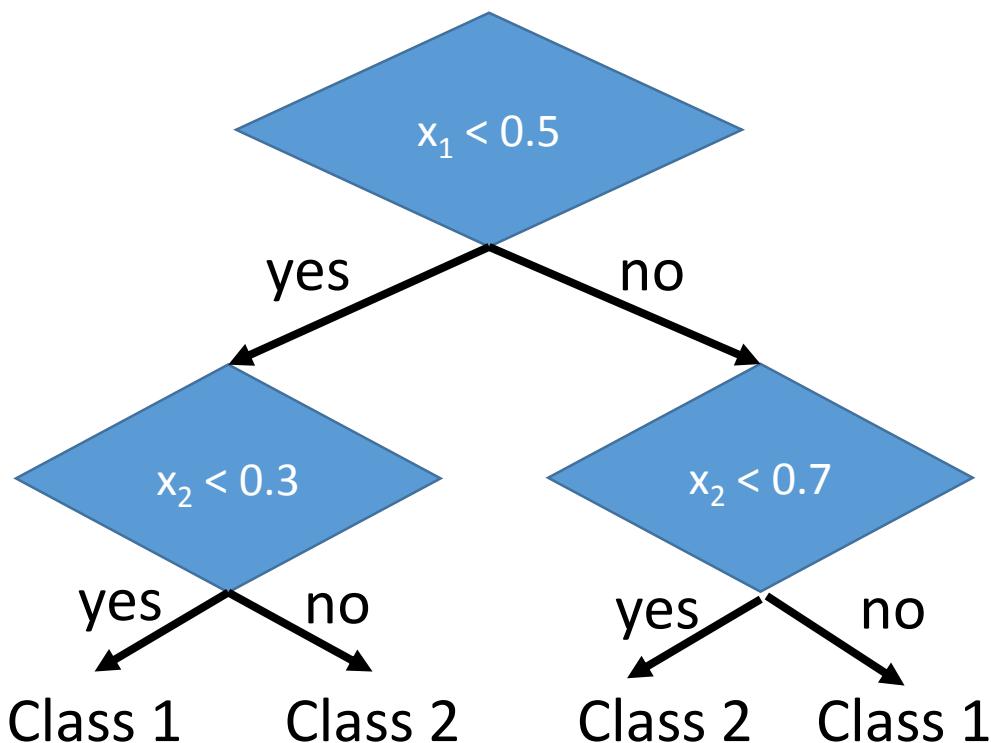


Bagging

This approach would be helpful when your model is complex, easy to overfit.
e.g. decision tree

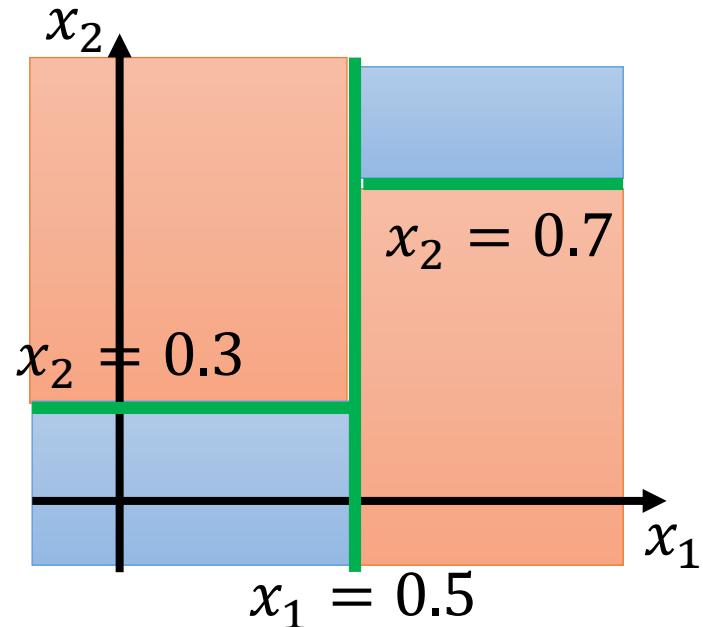


Decision Tree



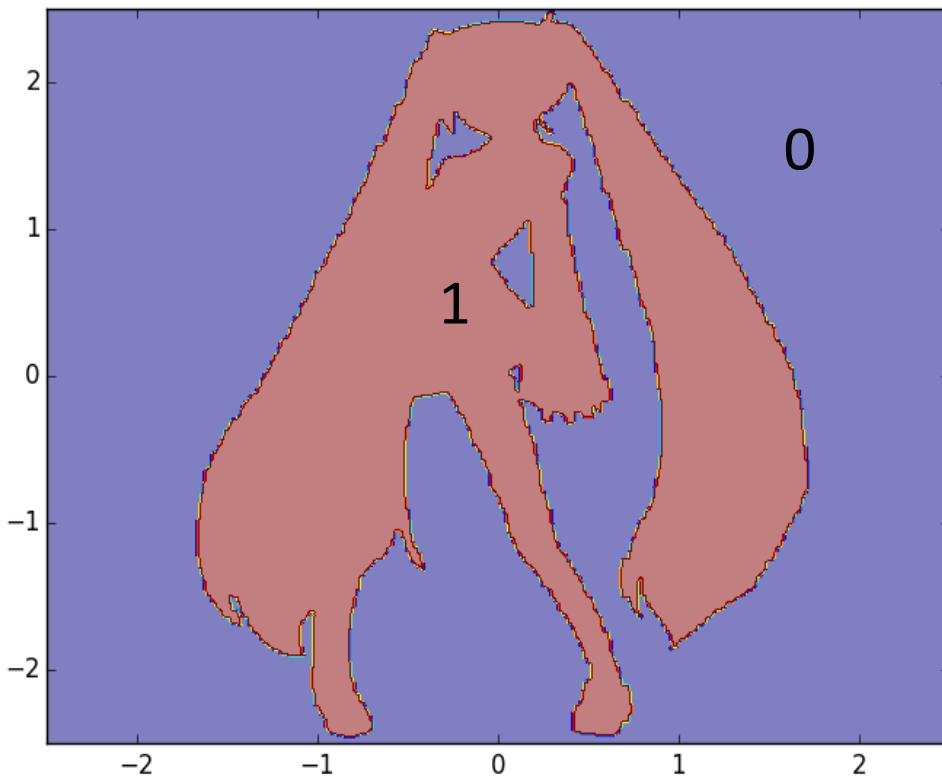
Can have more complex questions

Assume each object x is represented by a 2-dim vector $\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$



The questions in training
number of branches,
Branching criteria,
termination criteria,
base hypothesis

Experiment: Function of Miku



http://speech.ee.ntu.edu.tw/~tlkagk/courses/MLDS_2015_2/theano/miku

(1st column: x, 2nd column: y, 3rd column: output (1 or 0))

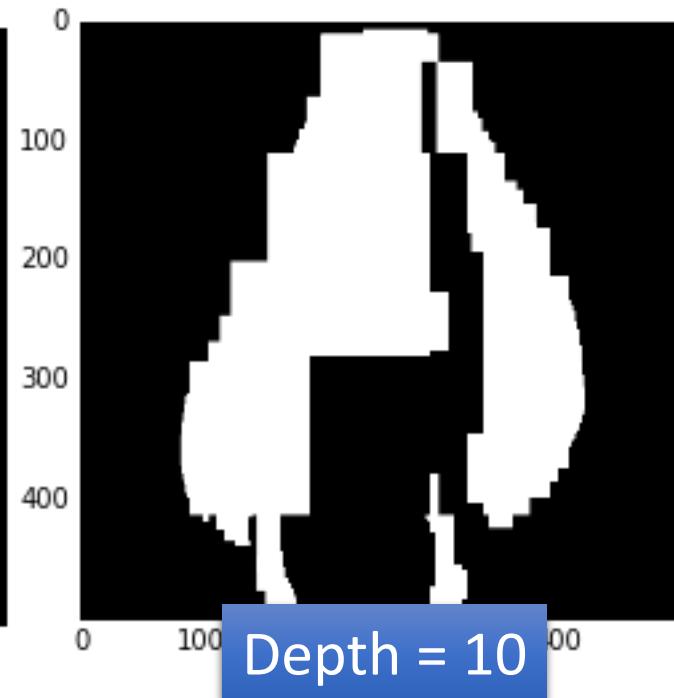
Experiment:

Function of Miku

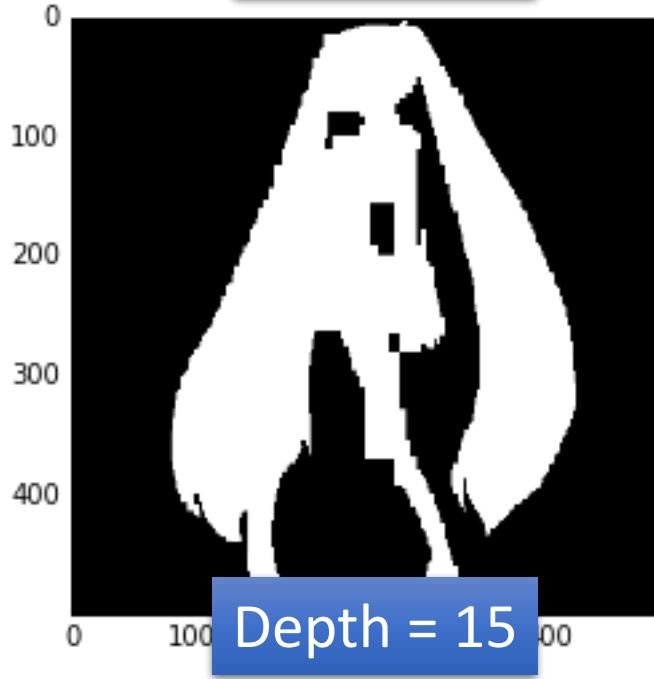
Single
Decision
Tree



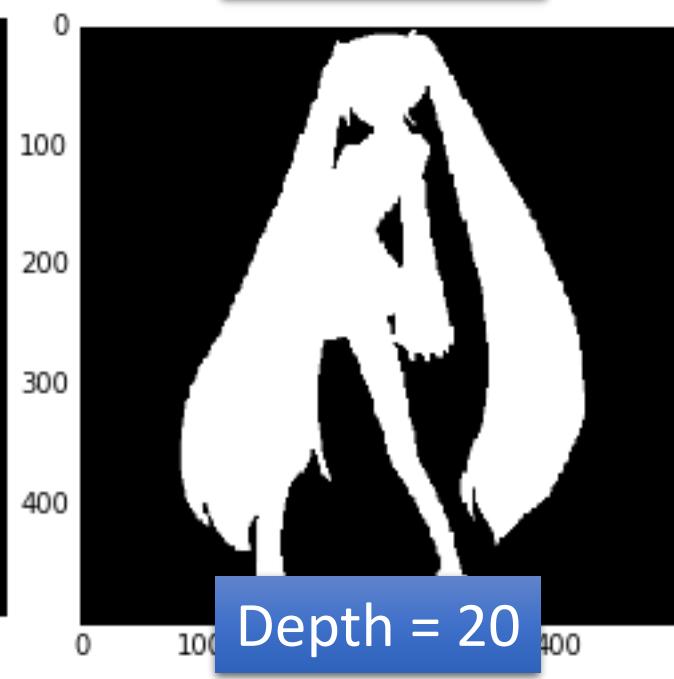
Depth = 5



Depth = 10



Depth = 15



Depth = 20

Random Forest

- Decision tree:
 - Easy to achieve 0% error rate on training data
 - If each training example has its own leaf
- Random forest: Bagging of decision tree
 - Resampling training data is not sufficient
 - Randomly restrict the features/questions used in each split
- Out-of-bag validation for bagging
 - Using $RF = f_2 + f_4$ to test x^1
 - Using $RF = f_2 + f_3$ to test x^2
 - Using $RF = f_1 + f_4$ to test x^3
 - Using $RF = f_1 + f_3$ to test x^4

train	f_1	f_2	f_3	f_4
x^1	O	X	O	X
x^2	O	X	X	O
x^3	X	O	O	X
x^4	X	O	X	O

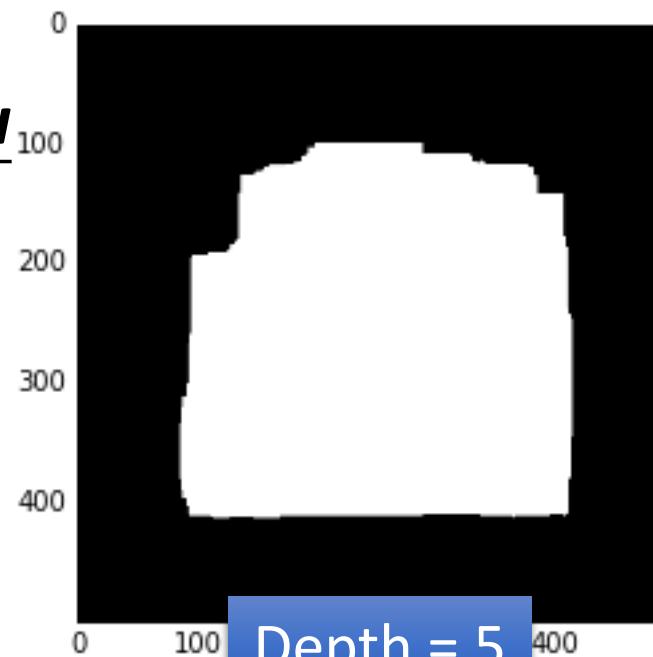
Out-of-bag (OOB) error
Good error estimation
of testing set

Experiment:

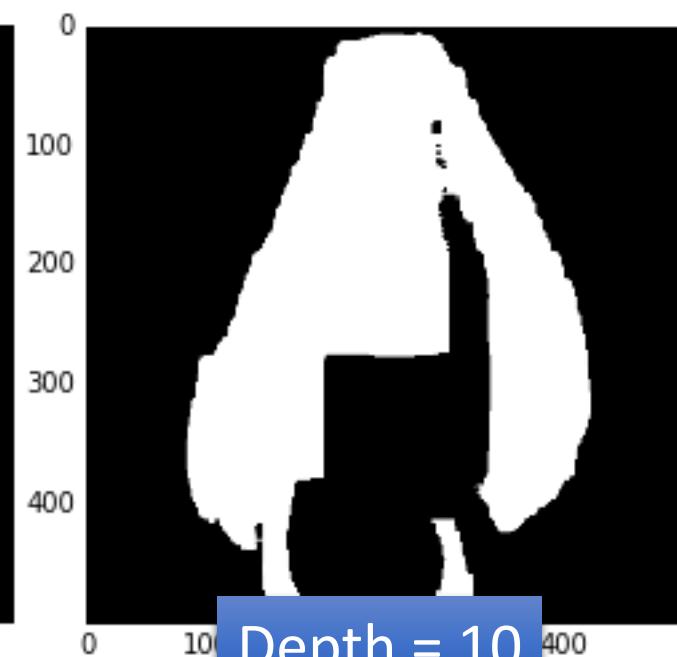
Function of Miku

Random
Forest

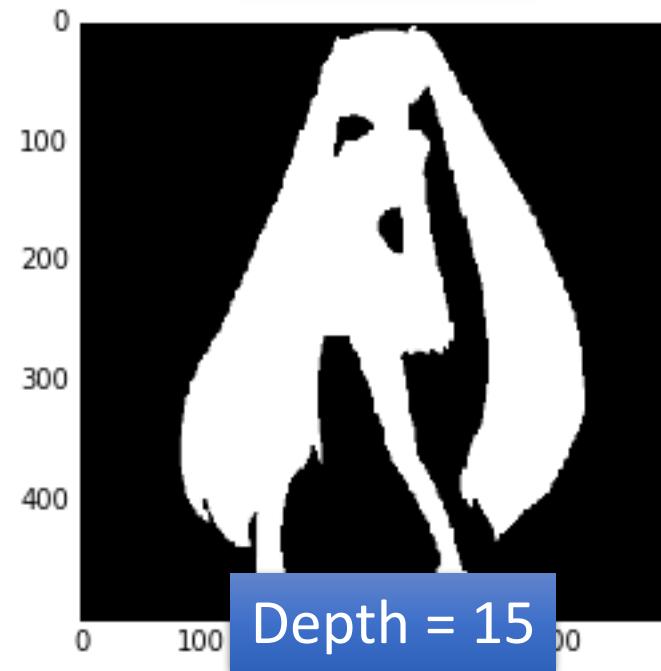
(100 trees)



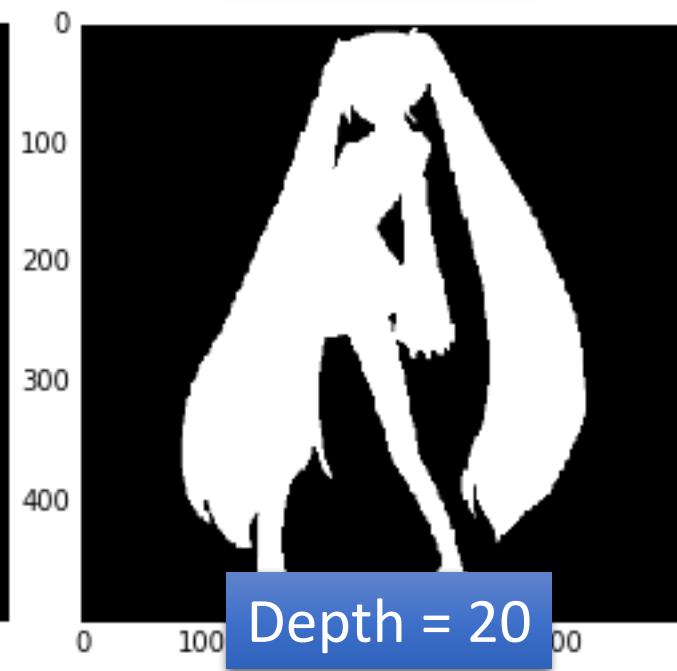
Depth = 5



Depth = 10



Depth = 15



Depth = 20

Ensemble: Boosting

Improving Weak Classifiers

Boosting

Training data:

$$\{(x^1, \hat{y}^1), \dots, (x^n, \hat{y}^n), \dots, (x^N, \hat{y}^N)\}$$

$\hat{y} = \pm 1$ (binary classification)

- Guarantee:
 - If your ML algorithm can produce classifier with error rate smaller than 50% on training data
 - You can obtain 0% error rate classifier after boosting.
- Framework of boosting
 - Obtain the first classifier $f_1(x)$
 - Find another function $f_2(x)$ to help $f_1(x)$
 - However, if $f_2(x)$ is similar to $f_1(x)$, it will not help a lot.
 - We want $f_2(x)$ to be complementary with $f_1(x)$ (How?)
 - Obtain the second classifier $f_2(x)$
 - Finally, combining all the classifiers
- The classifiers are learned sequentially.

How to obtain different classifiers?

- Training on different training data sets
- How to have different training data sets
 - Re-sampling your training data to form a new set
 - Re-weighting your training data to form a new set
 - In real implementation, you only have to change the cost/objective function

$$(x^1, \hat{y}^1, u^1) \quad u^1 = \cancel{1} \quad 0.4$$

$$(x^2, \hat{y}^2, u^2) \quad u^2 = \cancel{1} \quad 2.1$$

$$(x^3, \hat{y}^3, u^3) \quad u^3 = \cancel{1} \quad 0.7$$

$$L(f) = \sum_n l(f(x^n), \hat{y}^n)$$



$$L(f) = \sum_n u^n l(f(x^n), \hat{y}^n)$$

Idea of Adaboost

- Idea: **training $f_2(x)$ on the new training set that fails $f_1(x)$**
- How to find a new training set that fails $f_1(x)$?

ε_1 : the error rate of $f_1(x)$ on its training data

$$\varepsilon_1 = \frac{\sum_n u_1^n \delta(f_1(x^n) \neq \hat{y}^n)}{Z_1} \quad Z_1 = \sum_n u_1^n \quad \varepsilon_1 < 0.5$$

Changing the example weights from u_1^n to u_2^n such that

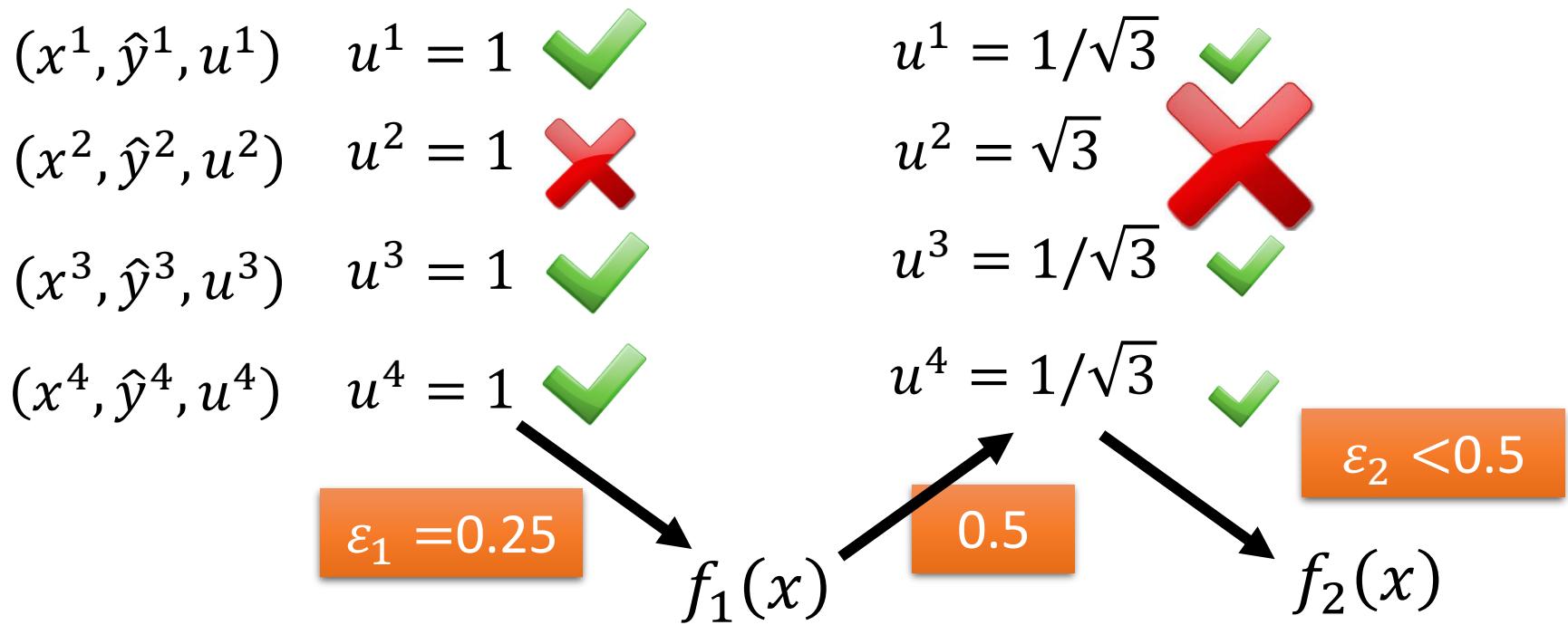
$$\frac{\sum_n u_2^n \delta(f_1(x^n) \neq \hat{y}^n)}{Z_2} = 0.5$$

The performance of f_1 for new weights would be random.

Training $f_2(x)$ based on the new weights u_2^n

Re-weighting Training Data

- Idea: **training $f_2(x)$ on the new training set that fails $f_1(x)$**
- How to find a new training set that fails $f_1(x)$?



Re-weighting Training Data

- Idea: **training $f_2(x)$ on the new training set that fails $f_1(x)$**
- How to find a new training set that fails $f_1(x)$?

{ If x^n misclassified by f_1 ($f_1(x^n) \neq \hat{y}^n$)
 $u_2^n \leftarrow u_1^n$ multiplying d_1 increase
If x^n correctly classified by f_1 ($f_1(x^n) = \hat{y}^n$)
 $u_2^n \leftarrow u_1^n$ devided by d_1 decrease

f_2 will be learned based on example weights u_2^n

What is the value of d_1 ?

Re-weighting Training Data

$$\varepsilon_1 = \frac{\sum_n u_1^n \delta(f_1(x^n) \neq \hat{y}^n)}{Z_1} \quad Z_1 = \sum_n u_1^n$$

$$\frac{\sum_n u_2^n \delta(f_1(x^n) \neq \hat{y}^n)}{Z_2} = 0.5 \quad \begin{array}{ll} f_1(x^n) \neq \hat{y}^n & u_2^n \leftarrow u_1^n \text{ multiplying } d_1 \\ f_1(x^n) = \hat{y}^n & u_2^n \leftarrow u_1^n \text{ devided by } d_1 \end{array}$$

$$\begin{aligned} &= \sum_{f_1(x^n) \neq \hat{y}^n} u_1^n d_1 &= \sum_{f_1(x^n) \neq \hat{y}^n} u_2^n + \sum_{f_1(x^n) = \hat{y}^n} u_2^n \\ &= \sum_n u_2^n &= \sum_{f_1(x^n) \neq \hat{y}^n} u_1^n d_1 + \sum_{f_1(x^n) = \hat{y}^n} u_1^n / d_1 \end{aligned}$$

$$\frac{\sum_{f_1(x^n) \neq \hat{y}^n} u_1^n d_1 + \sum_{f_1(x^n) = \hat{y}^n} u_1^n / d_1}{\sum_{f_1(x^n) \neq \hat{y}^n} u_1^n d_1} = 2$$

Re-weighting Training Data

$$\varepsilon_1 = \frac{\sum_n u_1^n \delta(f_1(x^n) \neq \hat{y}^n)}{Z_1} \quad Z_1 = \sum_n u_1^n$$

$$\frac{\sum_n u_2^n \delta(f_1(x^n) \neq \hat{y}^n)}{Z_2} = 0.5 \quad \begin{array}{ll} f_1(x^n) \neq \hat{y}^n & u_2^n \leftarrow u_1^n \text{ multiplying } d_1 \\ f_1(x^n) = \hat{y}^n & u_2^n \leftarrow u_1^n \text{ devided by } d_1 \end{array}$$

$$\frac{\sum_{f_1(x^n) \neq \hat{y}^n} u_1^n d_1 + \sum_{f_1(x^n) = \hat{y}^n} u_1^n / d_1}{\sum_{f_1(x^n) \neq \hat{y}^n} u_1^n d_1} = 2 \quad \frac{\sum_{f_1(x^n) = \hat{y}^n} u_1^n / d_1}{\sum_{f_1(x^n) \neq \hat{y}^n} u_1^n d_1} = 1$$

$$\sum_{f_1(x^n) = \hat{y}^n} u_1^n / d_1 = \sum_{f_1(x^n) \neq \hat{y}^n} u_1^n d_1 \quad \frac{1}{d_1} \sum_{f_1(x^n) = \hat{y}^n} u_1^n = d_1 \sum_{f_1(x^n) \neq \hat{y}^n} u_1^n$$

$$\varepsilon_1 = \frac{\sum_{f_1(x^n) \neq \hat{y}^n} u_1^n}{Z_1} \quad \sum_{f_1(x^n) \neq \hat{y}^n} u_1^n = Z_1 \varepsilon_1$$

$$Z_1(1 - \varepsilon_1) \quad Z_1(1 - \varepsilon_1) / d_1 = Z_1 \varepsilon_1 d_1$$

$$d_1 = \sqrt{(1 - \varepsilon_1) / \varepsilon_1} > 1$$

Algorithm for AdaBoost

- Giving training data
 $\{(x^1, \hat{y}^1, u_1^1), \dots, (x^n, \hat{y}^n, u_1^n), \dots, (x^N, \hat{y}^N, u_1^N)\}$
 - $\hat{y} = \pm 1$ (Binary classification), $u_1^n = 1$ (equal weights)
- For $t = 1, \dots, T$:
 - Training weak classifier $f_t(x)$ with weights $\{u_t^1, \dots, u_t^N\}$
 - ε_t is the error rate of $f_t(x)$ with weights $\{u_t^1, \dots, u_t^N\}$
 - For $n = 1, \dots, N$:
 - If x^n is misclassified by $f_t(x)$: $\hat{y}^n \neq f_t(x^n)$
 - $u_{t+1}^n = u_t^n \times d_t = u_t^n \times \exp(\alpha_t)$ $d_t = \sqrt{(1 - \varepsilon_t)/\varepsilon_t}$
 - Else:
 - $u_{t+1}^n = u_t^n / d_t = u_t^n \times \exp(-\alpha_t)$ $\alpha_t = \ln \sqrt{(1 - \varepsilon_t)/\varepsilon_t}$

$$u_{t+1}^n \leftarrow u_t^n \times \exp(-\alpha_t)$$

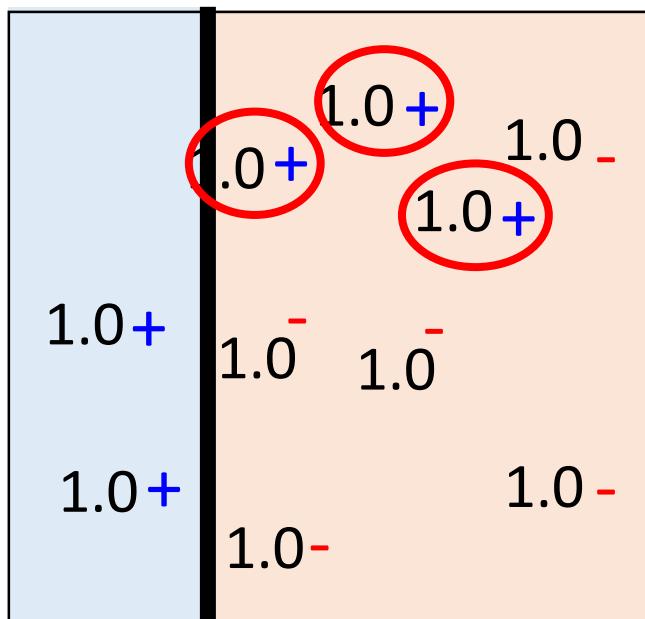
Algorithm for AdaBoost

- We obtain a set of functions: $f_1(x), \dots, f_t(x), \dots, f_T(x)$
 - How to aggregate them?
 - Uniform weight:
 - $H(x) = \text{sign}(\sum_{t=1}^T f_t(x))$
 - Non-uniform weight:
 - $H(x) = \text{sign}(\sum_{t=1}^T \alpha_t f_t(x))$
- Smaller error ε_t ,
larger weight for
final voting
- $$\alpha_t = \ln \sqrt{(1 - \varepsilon_t) / \varepsilon_t} \quad \epsilon^t = 0.1 \quad \epsilon^t = 0.4$$
- $$u_{t+1}^n = u_t^n \times \exp(-\hat{y}^n f_t(x^n) \alpha_t) \quad \alpha^t = 1.10 \quad \alpha^t = 0.20$$

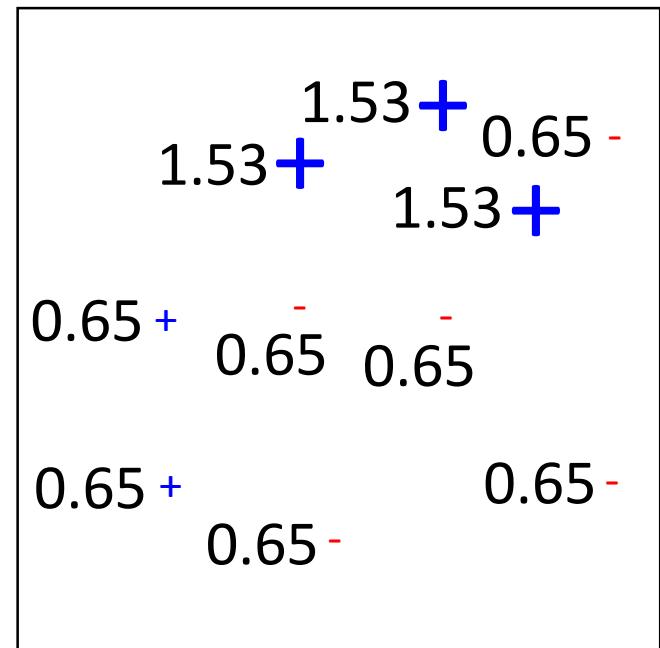
Toy Example

T=3, weak classifier = decision stump

- t=1



$$\begin{aligned}\varepsilon_1 &= 0.30 \\ d_1 &= 1.53 \\ \alpha_1 &= 0.42\end{aligned}$$



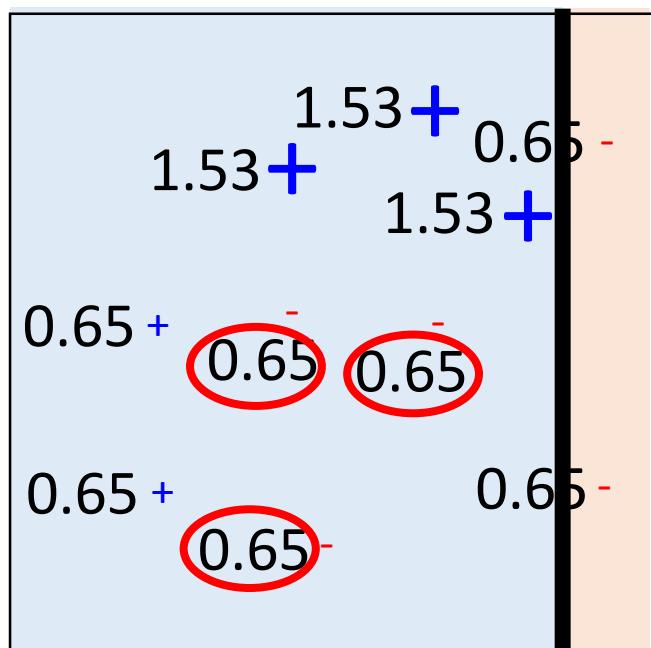
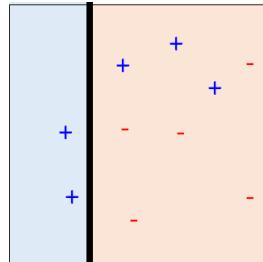
Toy Example

T=3, weak classifier = decision stump

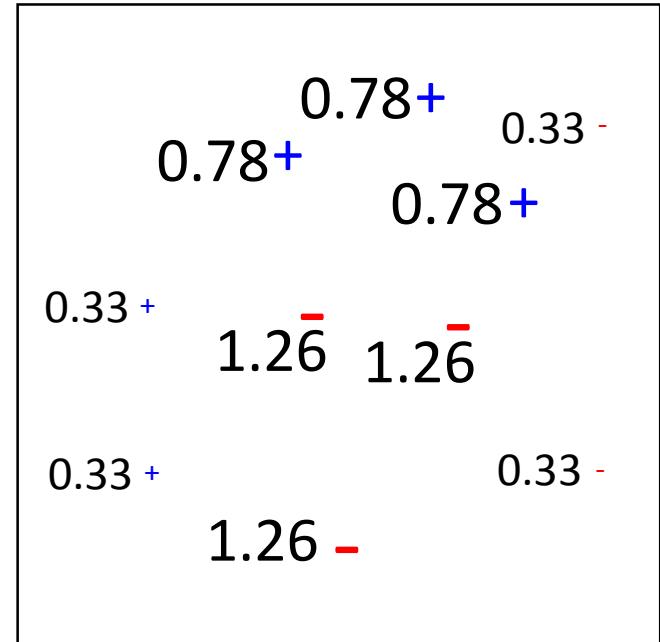
- t=2

$$\alpha_1 = 0.42$$

$$f_1(x):$$



$$\begin{aligned}\varepsilon_2 &= 0.21 \\ d_2 &= 1.94 \\ \alpha_2 &= 0.66\end{aligned}$$

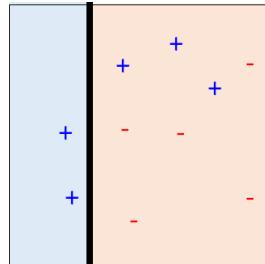


Toy Example

T=3, weak classifier = decision stump

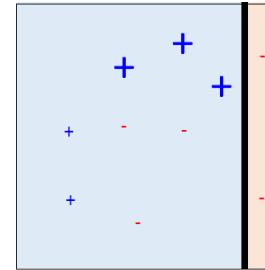
- t=3

$$f_1(x) :$$



$$\alpha_1 = 0.42$$

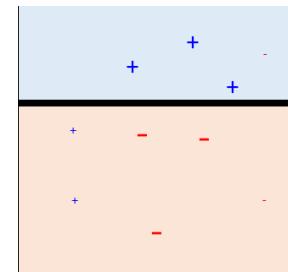
$$f_2(x) :$$



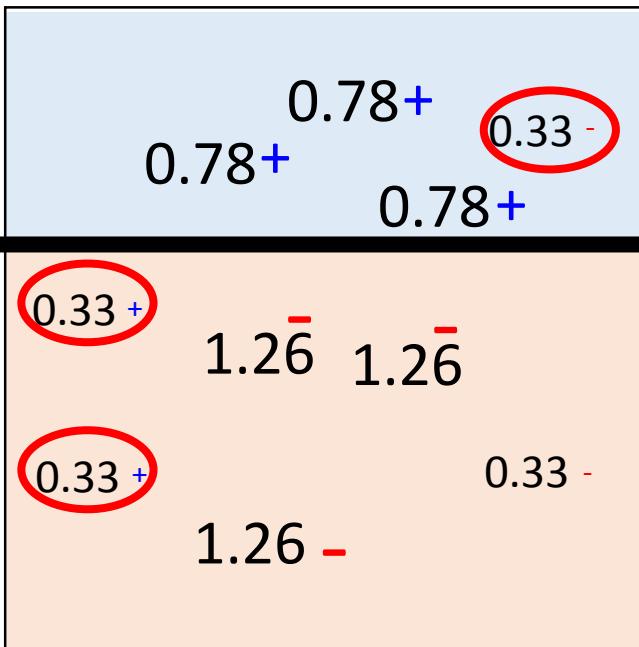
$$\alpha_2 = 0.66$$

$$f_3(x) :$$

$$\alpha_3 = 0.95$$



$$f_3(x)$$



$$\varepsilon_3 = 0.13$$

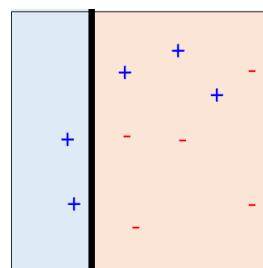
$$d_3 = 2.59$$

$$\alpha_3 = 0.95$$

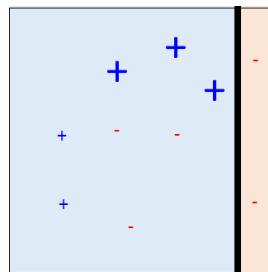
Toy Example

- Final Classifier: $H(x) = \text{sign}(\sum_{t=1}^T \alpha_t f_t(x))$

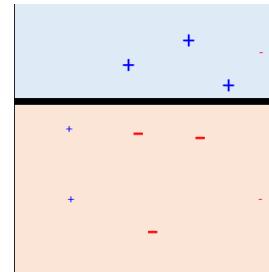
$\text{sign}(0.42$



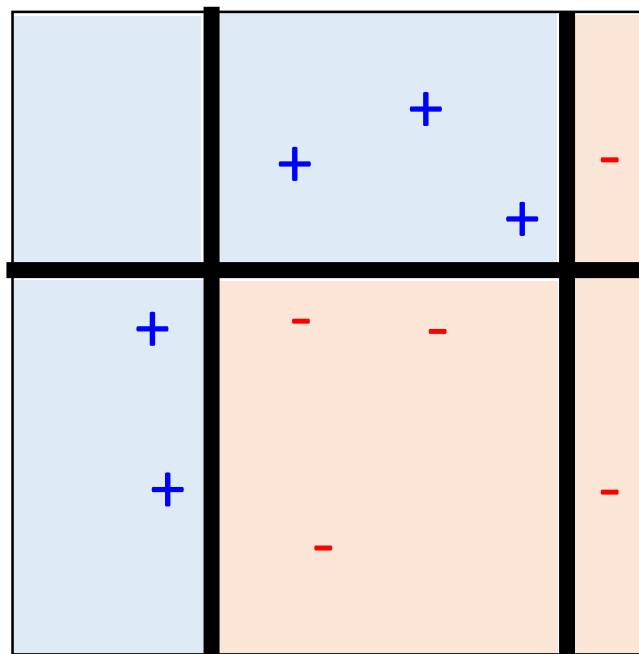
$+ 0.66$



$+ 0.95$



)



Warning of Math

$$H(x) = \text{sign} \left(\sum_{t=1}^T \alpha_t f_t(x) \right) \quad \alpha_t = \ln \sqrt{(1 - \varepsilon_t) / \varepsilon_t}$$

As we have more and more f_t (T increases), $H(x)$ achieves smaller and smaller error rate on training data.

Error Rate of Final Classifier

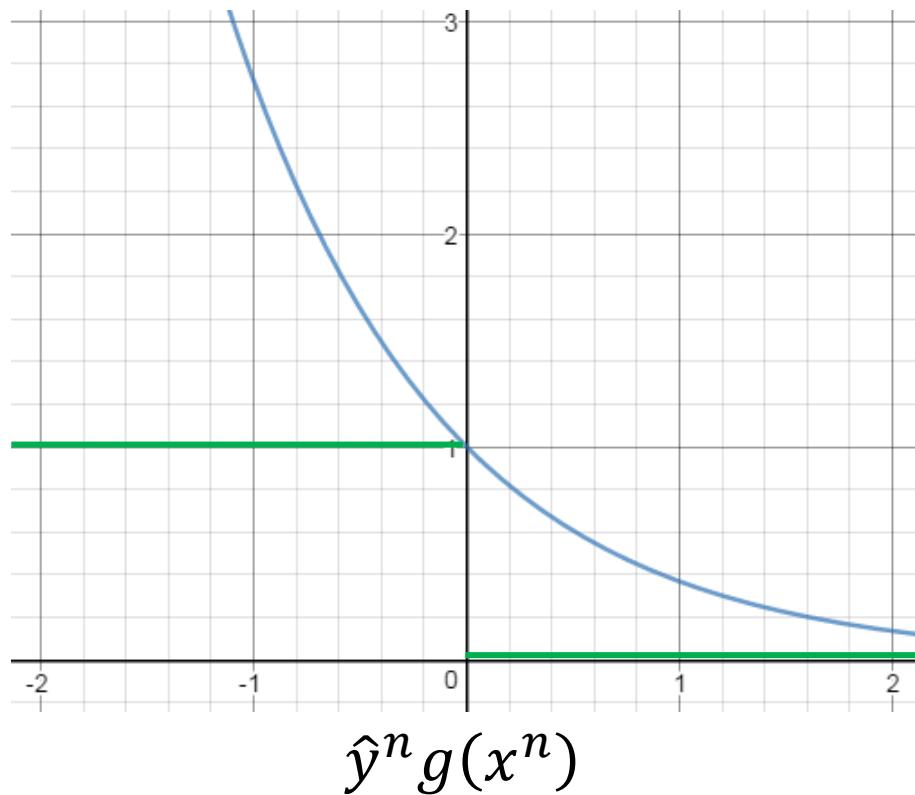
- Final classifier: $H(x) = \text{sign}(\sum_{t=1}^T \alpha_t f_t(x))$
 - $\alpha_t = \ln \sqrt{(1 - \varepsilon_t)/\varepsilon_t}$

Training Data Error Rate

$$= \frac{1}{N} \sum_n \delta(H(x^n) \neq \hat{y}^n)$$

$$= \frac{1}{N} \sum_n \underline{\delta(\hat{y}^n g(x^n) < 0)}$$

$$\leq \frac{1}{N} \sum_n \underline{\exp(-\hat{y}^n g(x^n))}$$



Training Data Error Rate

$$\leq \frac{1}{N} \sum_n \exp(-\hat{y}^n g(x^n)) = \frac{1}{N} Z_{T+1}$$

$$g(x) = \sum_{t=1}^T \alpha_t f_t(x)$$

$$\alpha_t = \ln \sqrt{(1 - \varepsilon_t) / \varepsilon_t}$$

Z_t : the summation of the weights of training data for training f_t

What is $Z_{T+1} = ?$ $Z_{T+1} = \sum_n u_{T+1}^n$

$$\left. \begin{array}{l} u_1^n = 1 \\ u_{t+1}^n = u_t^n \times \exp(-\hat{y}^n f_t(x^n) \alpha_t) \end{array} \right\} u_{T+1}^n = \prod_{t=1}^T \exp(-\hat{y}^n f_t(x^n) \alpha_t)$$

$$Z_{T+1} = \sum_n \prod_{t=1}^T \exp(-\hat{y}^n f_t(x^n) \alpha_t)$$

$$= \sum_n \exp \left(-\hat{y}^n \sum_{t=1}^T f_t(x^n) \alpha_t \right)$$

Training Data Error Rate

$$\leq \frac{1}{N} \sum_n \exp(-\hat{y}^n g(x^n)) = \frac{1}{N} Z_{T+1}$$

$$g(x) = \sum_{t=1}^T \alpha_t f_t(x)$$
$$\alpha_t = \ln \sqrt{(1 - \varepsilon_t)/\varepsilon_t}$$

$Z_1 = N$ (equal weights)

$$Z_t = \underline{Z_{t-1} \varepsilon_t \exp(\alpha_t)} + \underline{Z_{t-1} (1 - \varepsilon_t) \exp(-\alpha_t)}$$

Misclassified portion in Z_{t-1} Correctly classified portion in Z_{t-1}

$$= Z_{t-1} \varepsilon_t \sqrt{(1 - \varepsilon_t)/\varepsilon_t} + Z_{t-1} (1 - \varepsilon_t) \sqrt{\varepsilon_t/(1 - \varepsilon_t)}$$

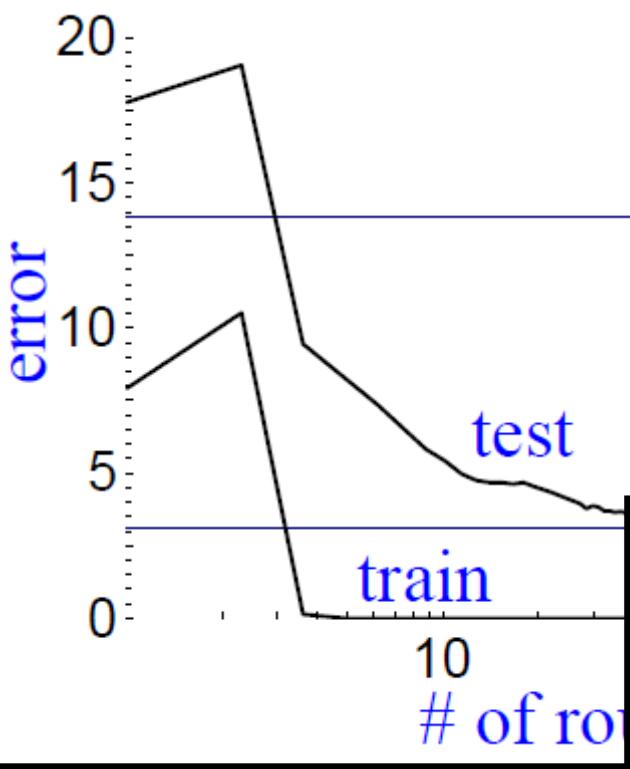
$$= Z_{t-1} \times 2 \sqrt{\varepsilon_t (1 - \varepsilon_t)} \quad Z_{T+1} = N \prod_{t=1}^T 2 \sqrt{\varepsilon_t (1 - \varepsilon_t)}$$

$$\text{Training Data Error Rate} \leq \prod_{t=1}^T \frac{2 \sqrt{\varepsilon_t (1 - \varepsilon_t)}}{<1}$$

Smaller and
smaller

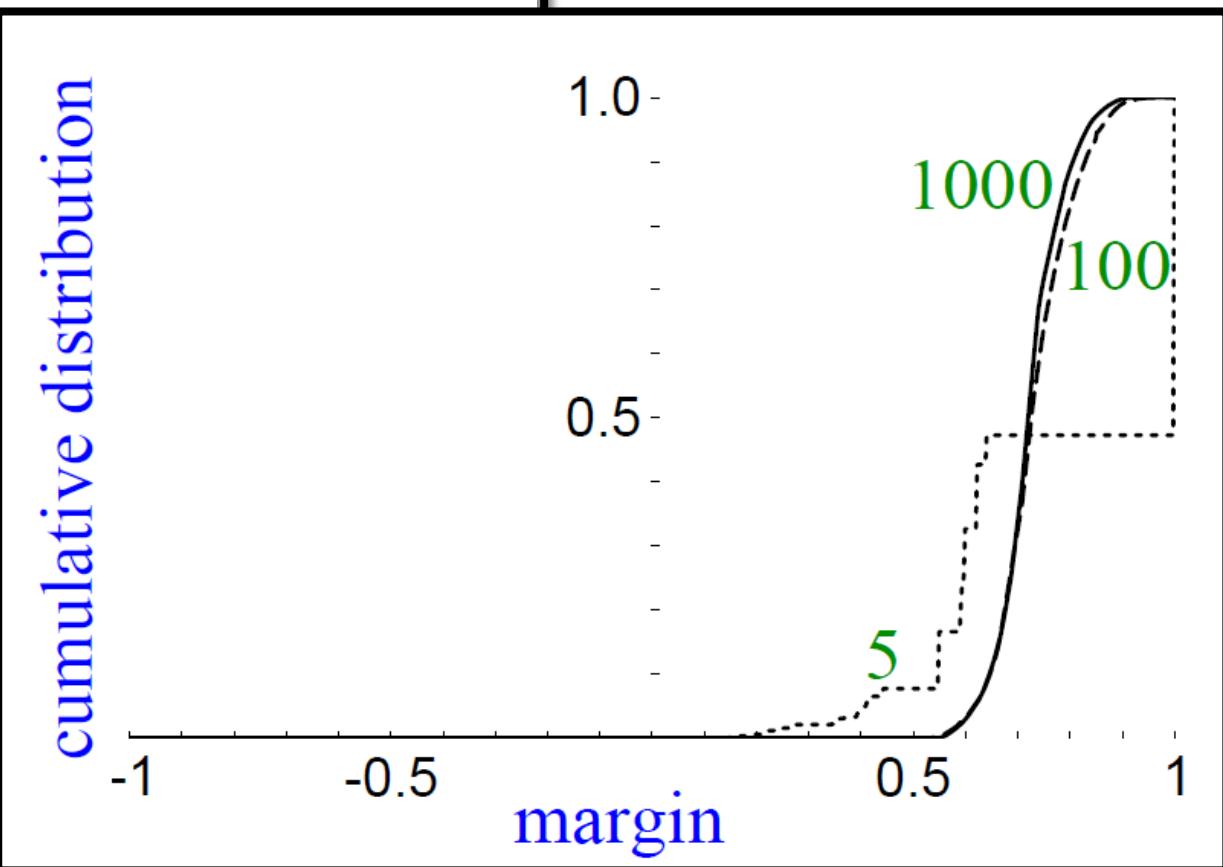
End of Warning

Even though the training error is 0,
the testing error still decreases?



$$H(x) = \text{sign} \left(\sum_{t=1}^T \alpha_t f_t(x) \right)$$

$$\text{Margin} = \hat{y}g(x)$$



Large Margin?

$$H(x) = \text{sign} \left(\sum_{t=1}^T \alpha_t f_t(x) \right)$$

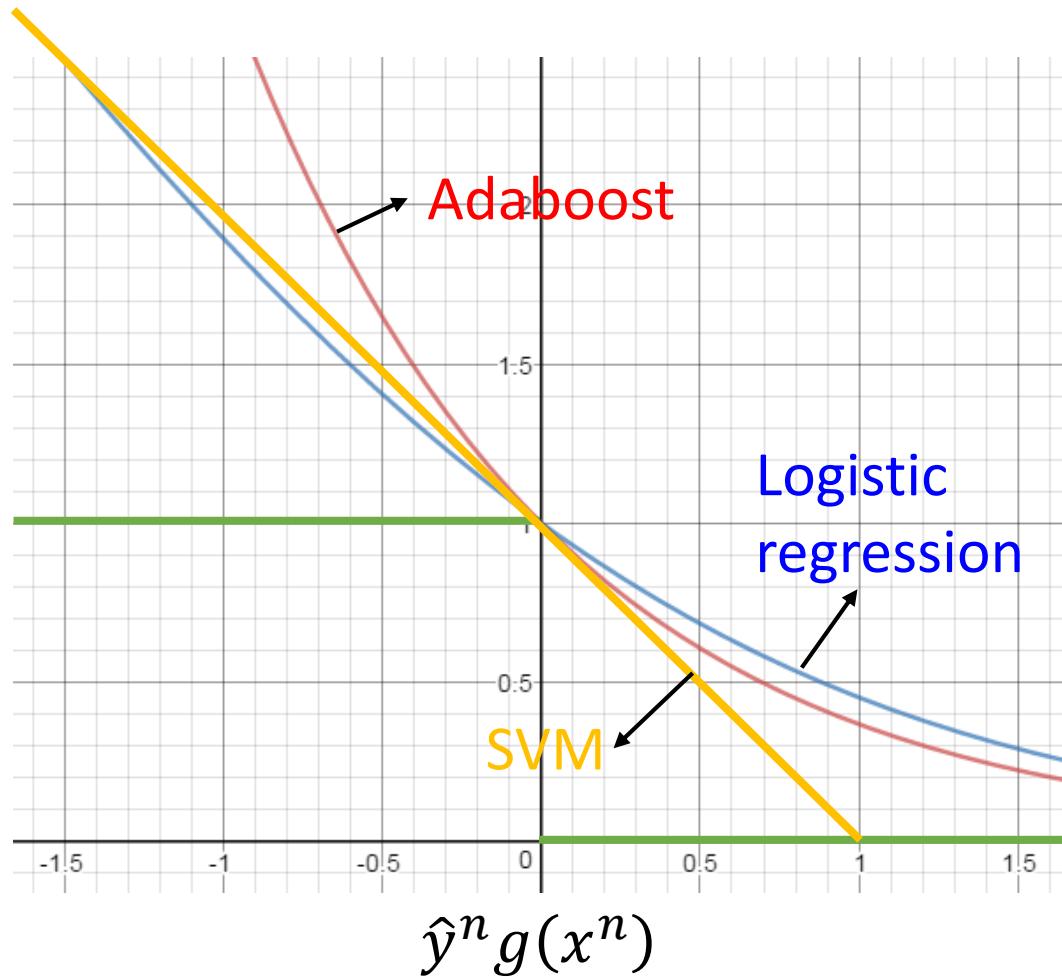
Training Data Error Rate =

$$= \frac{1}{N} \sum_n \delta(H(x^n) \neq \hat{y}^n)$$

$$\leq \frac{1}{N} \sum_n \exp(-\hat{y}^n g(x^n))$$

$$= \prod_{t=1}^T 2\sqrt{\epsilon_t(1-\epsilon_t)}$$

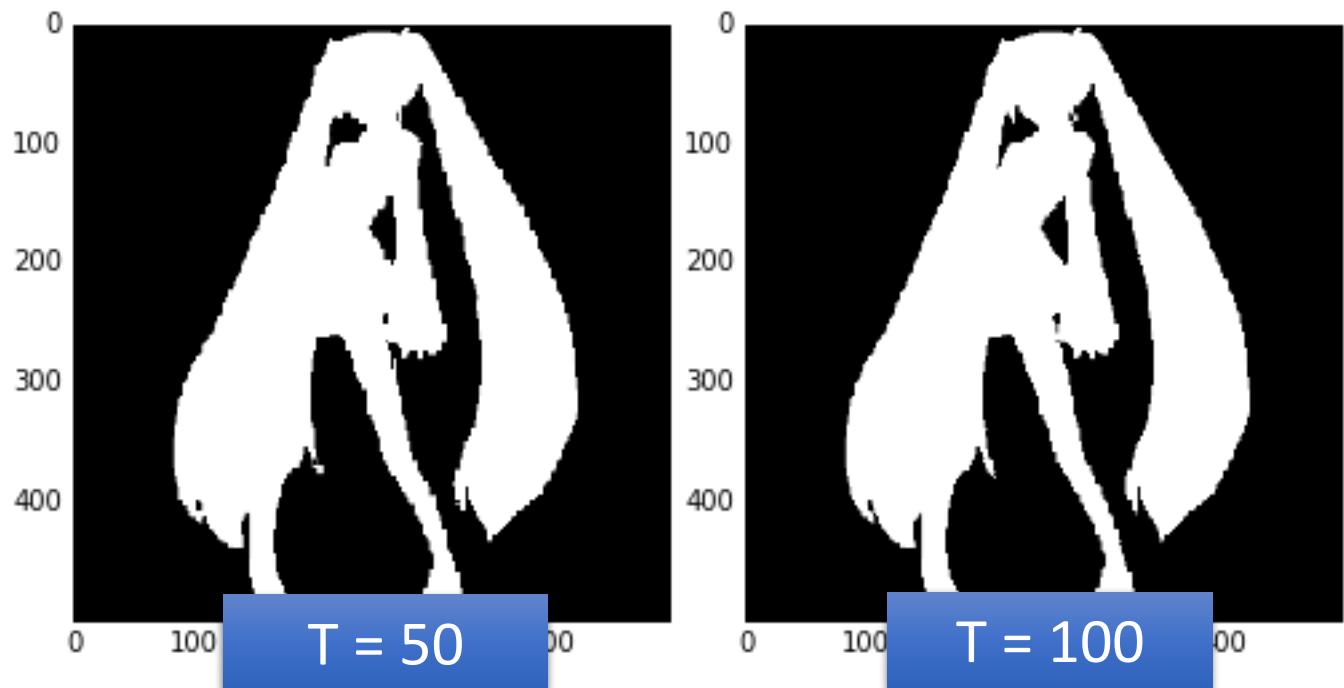
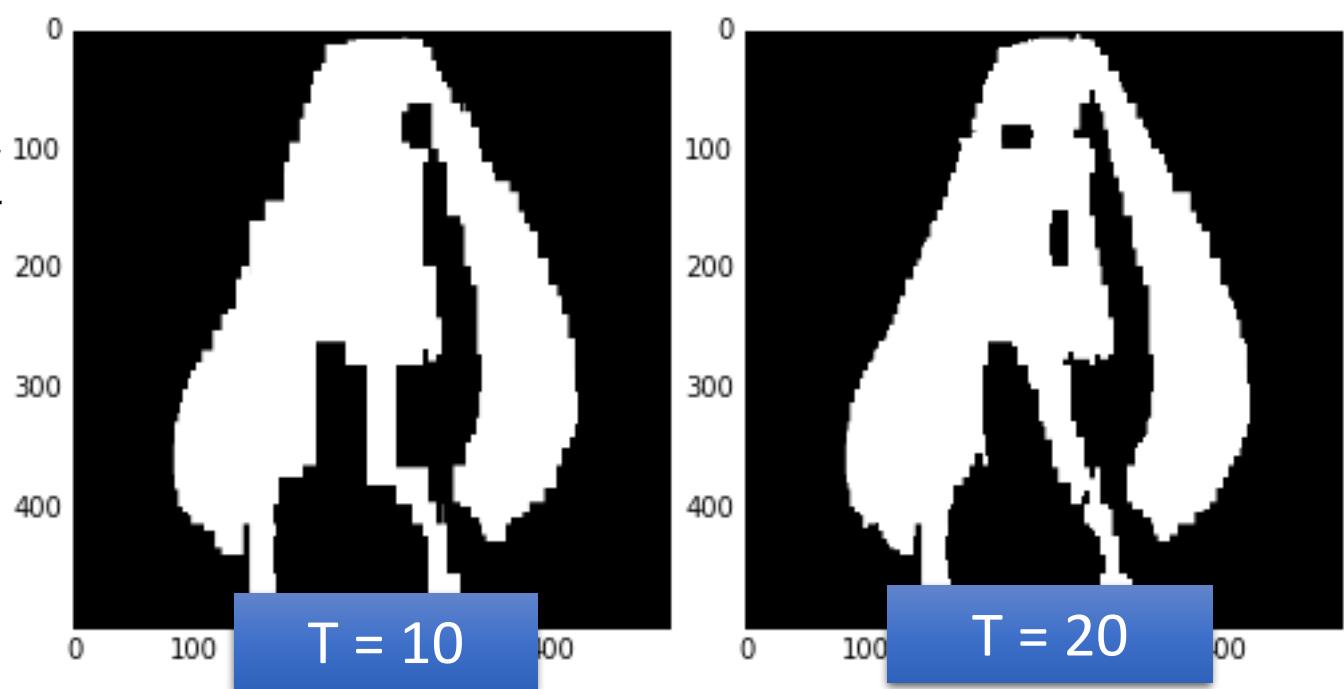
Getting smaller and
smaller as T increase



Experiment:

Function of Miku

Adaboost
+Decision Tree
(depth = 5)



To learn more ...

- Introduction of Adaboost:
 - Freund; Schapire (1999). "A Short Introduction to Boosting"
- Multiclass/Regression
 - Y. Freund, R. Schapire, "A Decision-Theoretic Generalization of on-Line Learning and an Application to Boosting", 1995.
 - Robert E. Schapire and Yoram Singer. Improved boosting algorithms using confidence-rated predictions. In Proceedings of the Eleventh Annual Conference on Computational Learning Theory, pages 80–91, 1998.
- Gentle Boost
 - Schapire, Robert; Singer, Yoram (1999). "Improved Boosting Algorithms Using Confidence-rated Predictions".

General Formulation of Boosting

- Initial function $g_0(x) = 0$
- For $t = 1$ to T :
 - Find a function $f_t(x)$ and α_t to improve $g_{t-1}(x)$
 - $g_{t-1}(x) = \sum_{i=1}^{t-1} \alpha_i f_i(x)$
 - $g_t(x) = g_{t-1}(x) + \alpha_t f_t(x)$
 - Output: $H(x) = \text{sign}(g_T(x))$

What is the learning target of $g(x)$?

$$\text{Minimize } L(g) = \sum_n l(\hat{y}^n, g(x^n)) = \sum_n \exp(-\hat{y}^n g(x^n))$$

Gradient Boosting

- Find $g(x)$, minimize $L(g) = \sum_n \exp(-\hat{y}^n g(x^n))$
 - If we already have $g(x) = g_{t-1}(x)$, how to update $g(x)$?

Gradient Descent:

$$g_t(x) = g_{t-1}(x) - \eta \frac{\partial L(g)}{\partial g(x)} \Big|_{g(x) = g_{t-1}(x)}$$

Same direction

$$\sum_n \exp(-\hat{y}^n g_{t-1}(x^n))(\hat{y}^n)$$
$$g_t(x) = g_{t-1}(x) + \alpha_t f_t(x)$$

Gradient Boosting

$$f_t(x) \longleftrightarrow \sum_n \exp(-\hat{y}^n g_t(x^n)) (\hat{y}^n)$$

Same direction

We want to find $f_t(x)$ maximizing

$$\sum_n \frac{\exp(-\hat{y}^n g_{t-1}(x^n)) (\hat{y}^n)}{\text{example weight } u_t^n} f_t(x^n)$$

Minimize Error
Same sign

$$u_t^n = \exp(-\hat{y}^n g_{t-1}(x^n)) = \exp\left(-\hat{y}^n \sum_{i=1}^{t-1} \alpha_i f_i(x^n)\right)$$

$$= \prod_{i=1}^{t-1} \exp(-\hat{y}^n \alpha_i f_i(x^n))$$

Exactly the weights we obtain
in Adaboost

Gradient Boosting

- Find $g(x)$, minimize $L(g) = \sum_n \exp(-\hat{y}^n g(x^n))$

$$g_t(x) = g_{t-1}(x) + \alpha_t f_t(x)$$

α_t is something like learning rate

Find α_t minimizing $L(g_{t+1})$

$$\begin{aligned} L(g) &= \sum_n \exp(-\hat{y}^n (g_{t-1}(x) + \alpha_t f_t(x))) \\ &= \sum_n \exp(-\hat{y}^n g_{t-1}(x)) \exp(-\hat{y}^n \alpha_t f_t(x)) \\ &= \sum_{\hat{y}^n \neq f_t(x)} \exp(-\hat{y}^n g_{t-1}(x^n)) \exp(\alpha_t) \\ &\quad + \sum_{\hat{y}^n = f_t(x)} \exp(-\hat{y}^n g_{t-1}(x^n)) \exp(-\alpha_t) \end{aligned}$$

Find α_t such that

$$\frac{\partial L(g)}{\partial \alpha_t} = 0$$

$$\alpha_t = \ln \sqrt{(1 - \varepsilon_t) / \varepsilon_t}$$

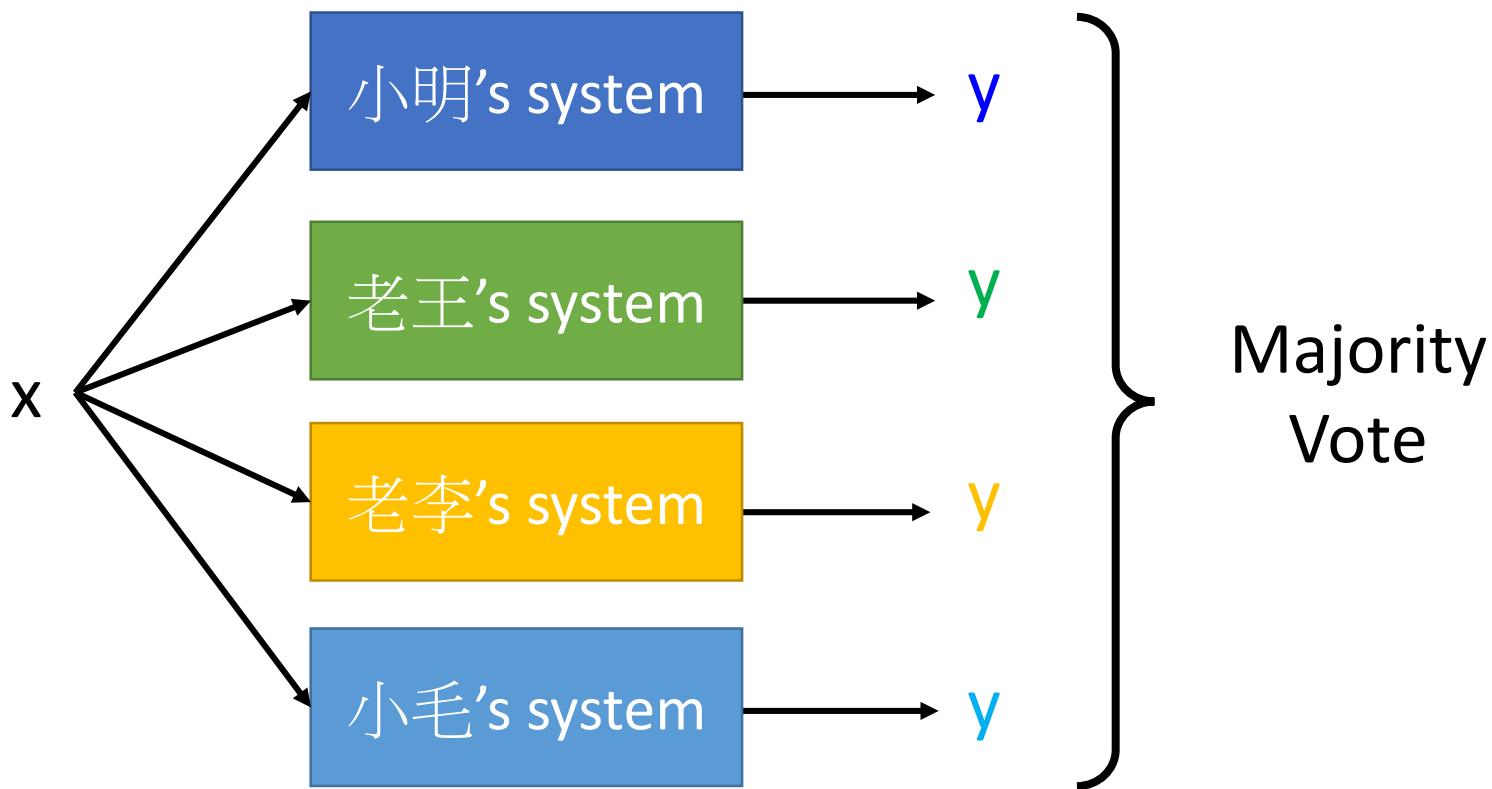
Adaboost!

Cool Demo

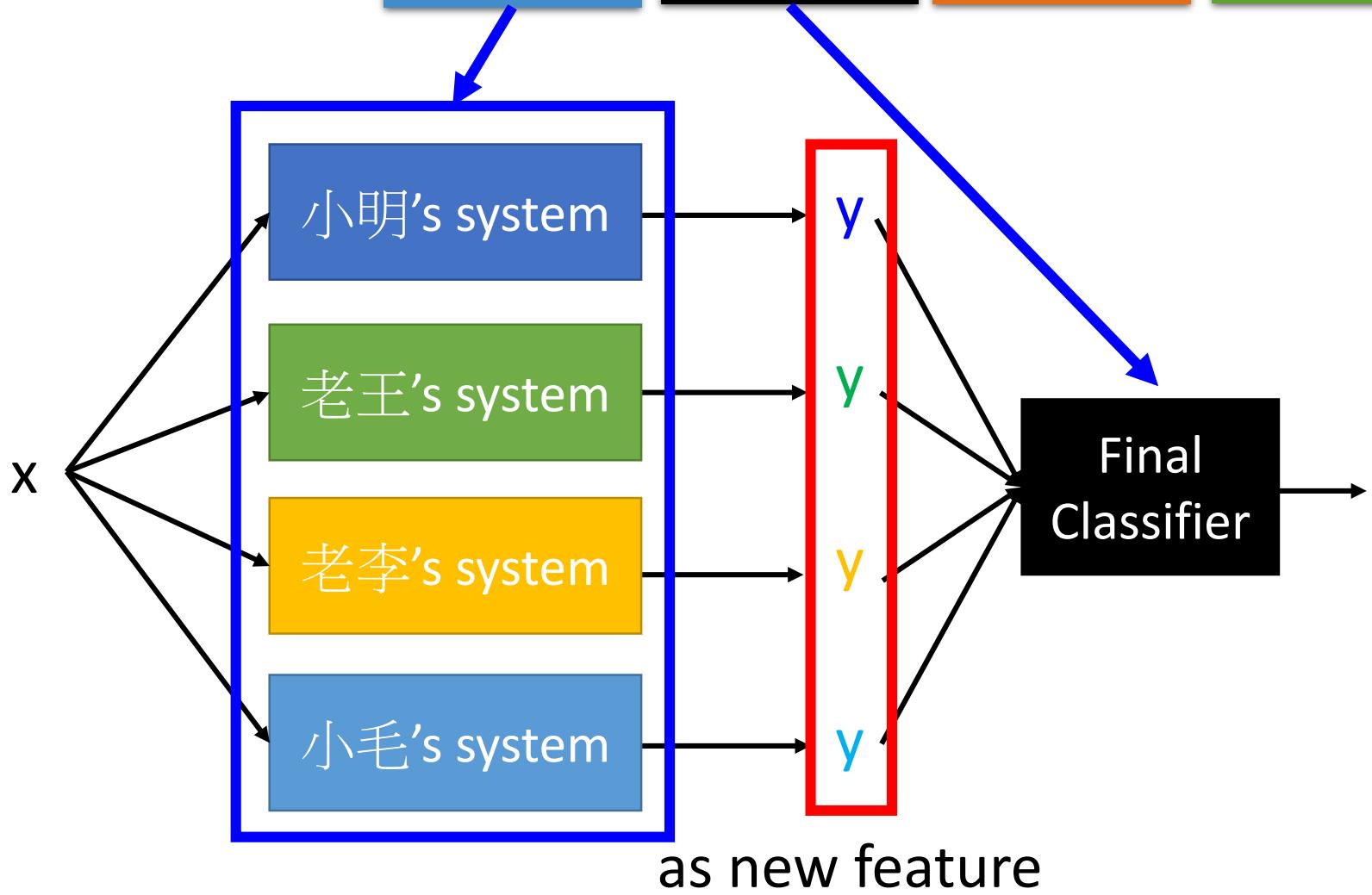
- http://arogozhnikov.github.io/2016/07/05/gradient_boosting_playground.html

Ensemble: Stacking

Voting



Stacking



2017

新年快樂

Happy New Year

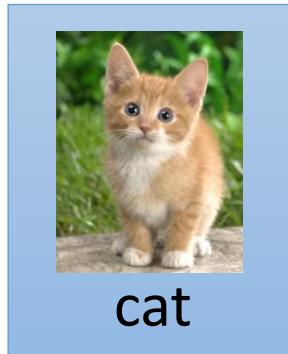
Semi-supervised Learning

Introduction

- Supervised learning: $\{(x^r, \hat{y}^r)\}_{r=1}^R$
 - E.g. x^r : image, \hat{y}^r : class labels
- Semi-supervised learning: $\{(x^r, \hat{y}^r)\}_{r=1}^R, \{x^u\}_{u=R}^{R+U}$
 - A set of unlabeled data, usually $U \gg R$
 - Transductive learning: unlabeled data is the testing data
 - Inductive learning: unlabeled data is not the testing data
- Why semi-supervised learning?
 - Collecting data is easy, but collecting “labelled” data is expensive
 - We do semi-supervised learning in our lives

Why semi-supervised learning helps?

Labelled
data

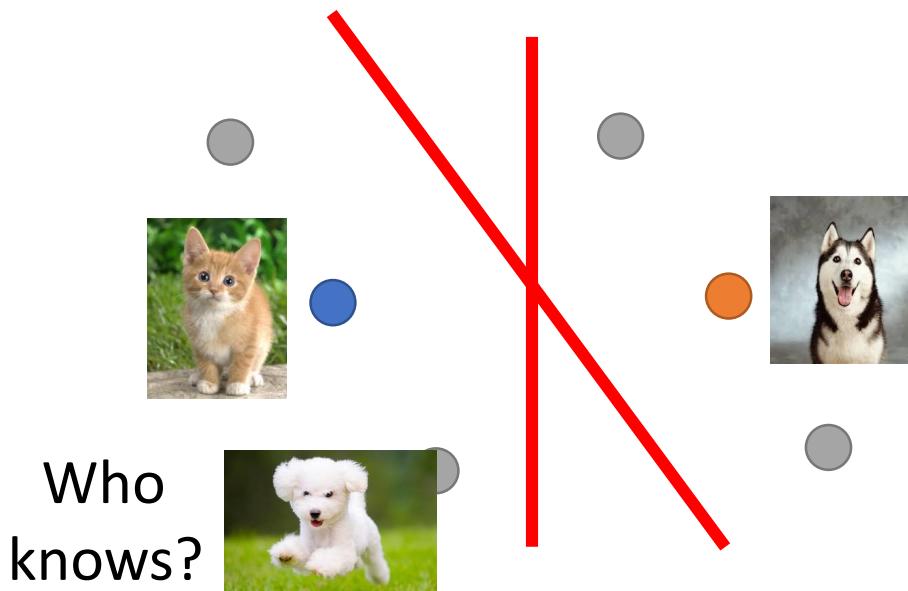


Unlabeled
data



(Image of cats and dogs without labeling)

Why semi-supervised learning helps?



The distribution of the unlabeled data tell us *something*.

Usually with some assumptions

Outline

Semi-supervised Learning for Generative Model

Low-density Separation Assumption

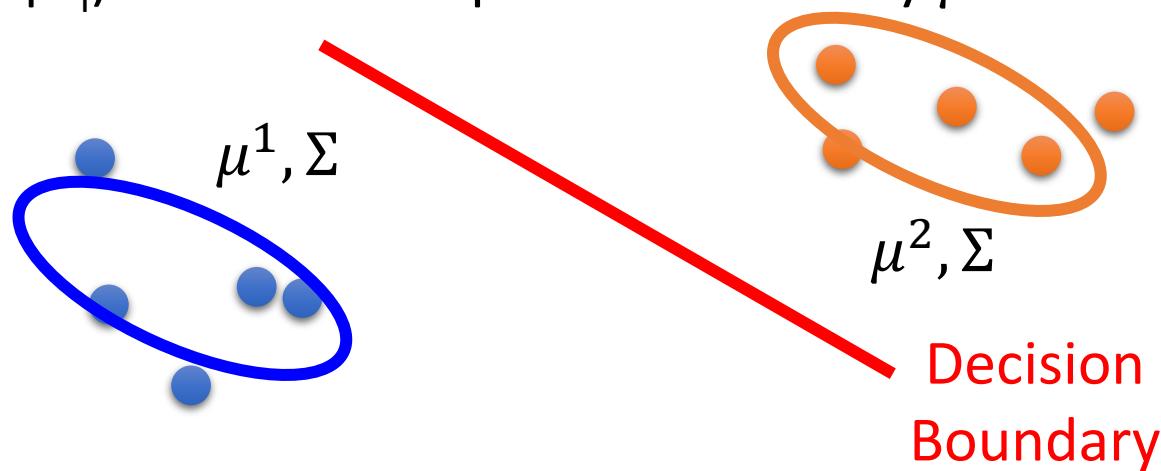
Smoothness Assumption

Better Representation

Semi-supervised Learning for Generative Model

Supervised Generative Model

- Given labelled training examples $x^r \in C_1, C_2$
 - looking for most likely prior probability $P(C_i)$ and class-dependent probability $P(x|C_i)$
 - $P(x|C_i)$ is a Gaussian parameterized by μ^i and Σ

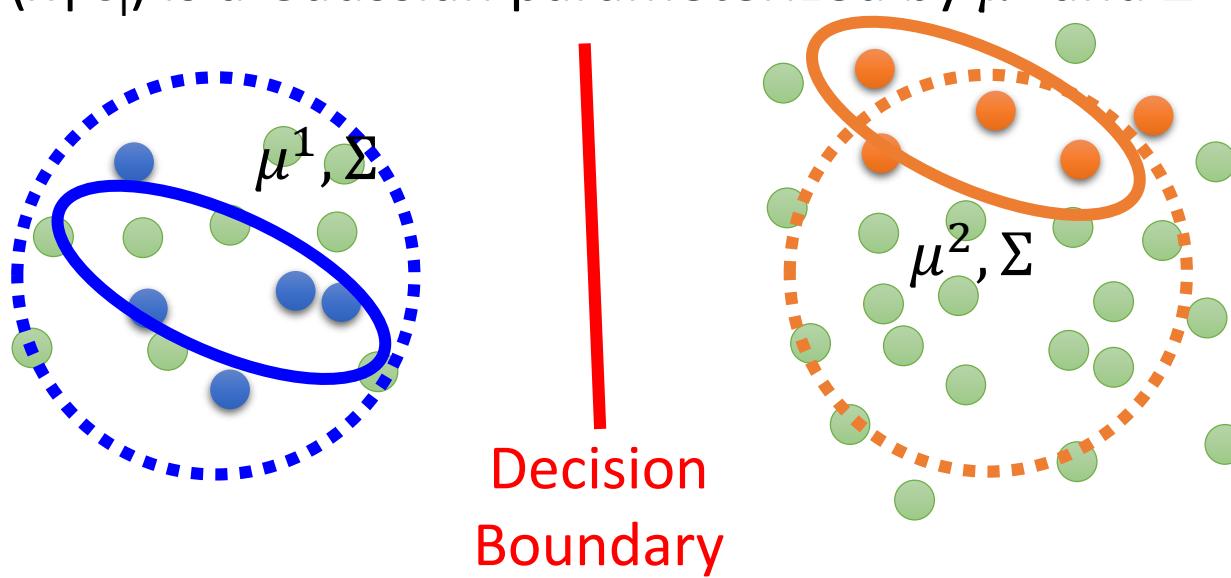


With $P(C_1), P(C_2), \mu^1, \mu^2, \Sigma$

$$P(C_1|x) = \frac{P(x|C_1)P(C_1)}{P(x|C_1)P(C_1) + P(x|C_2)P(C_2)}$$

Semi-supervised Generative Model

- Given labelled training examples $x^r \in C_1, C_2$
 - looking for most likely prior probability $P(C_i)$ and class-dependent probability $P(x|C_i)$
 - $P(x|C_i)$ is a Gaussian parameterized by μ^i and Σ



The unlabeled data x^u help re-estimate $P(C_1), P(C_2), \mu^1, \mu^2, \Sigma$

Semi-supervised Generative Model

The algorithm converges eventually, but the initialization influences the results.

- Initialization: $\theta = \{P(C_1), P(C_2), \mu^1, \mu^2, \Sigma\}$
- Step 1: compute the posterior probability of unlabeled data

$$P_\theta(C_1|x^u) \quad \text{Depending on model } \theta$$

M

- Step 2: update model

↑
Back to step 1

$$P(C_1) = \frac{N_1 + \sum_{x^u} P(C_1|x^u)}{N}$$

N: total number of examples
N₁: number of examples belonging to C₁

$$\mu^1 = \frac{1}{N_1} \sum_{x^r \in C_1} x^r + \frac{1}{\sum_{x^u} P(C_1|x^u)} \sum_{x^u} P(C_1|x^u)x^u \dots$$

Why?

$$\theta = \{P(C_1), P(C_2), \mu^1, \mu^2, \Sigma\}$$

- Maximum likelihood with labelled data Closed-form solution

$$\log L(\theta) = \sum_{x^r} \log P_\theta(x^r, \hat{y}^r)$$

$$\begin{aligned} P_\theta(x^r, \hat{y}^r) \\ = P_\theta(x^r | \hat{y}^r) P(\hat{y}^r) \end{aligned}$$

- Maximum likelihood with labelled + unlabeled data

$$\log L(\theta) = \sum_{x^r} \log P_\theta(x^r, \hat{y}^r) + \sum_{x^u} \log P_\theta(x^u)$$

Solved
iteratively

$$P_\theta(x^u) = P_\theta(x^u | C_1) P(C_1) + P_\theta(x^u | C_2) P(C_2)$$

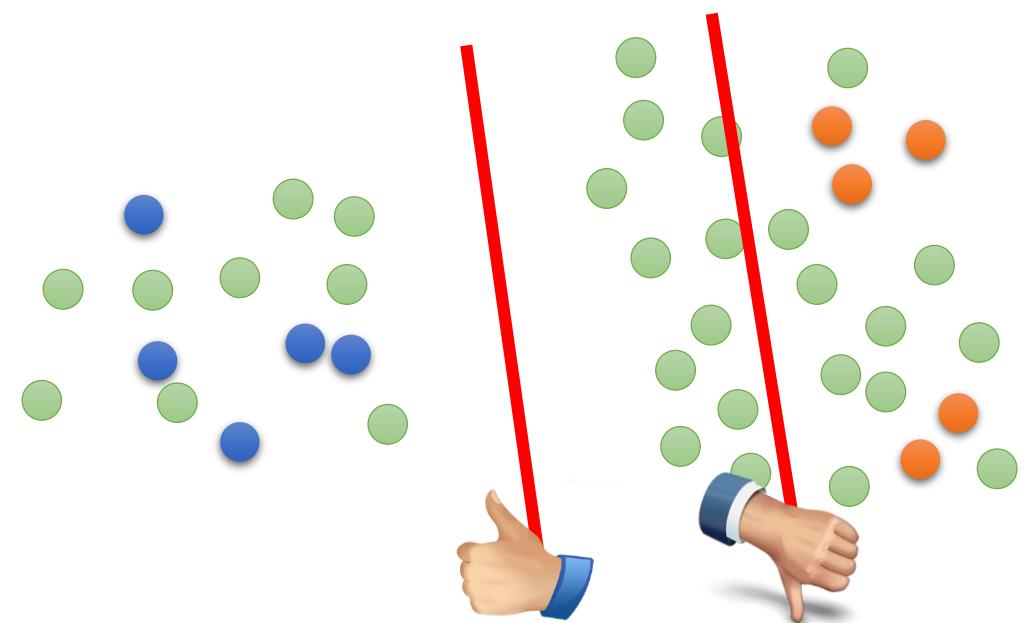
$(x^u$ can come from either C_1 and C_2)

Semi-supervised Learning

Low-density Separation

非黑即白

“Black-or-white”



Self-training

- Given: labelled data set = $\{(x^r, \hat{y}^r)\}_{r=1}^R$, unlabeled data set = $\{x^u\}_{u=l}^{R+U}$
- Repeat:
 - Train model f^* from labelled data set
 - Independent to the model
 - Regression?
 - Apply f^* to the unlabeled data set
 - Obtain $\{(x^u, y^u)\}_{u=l}^{R+U}$ Pseudo-label
 - Remove a set of data from unlabeled data set, and add them into the labeled data set

How to choose the data set remains open

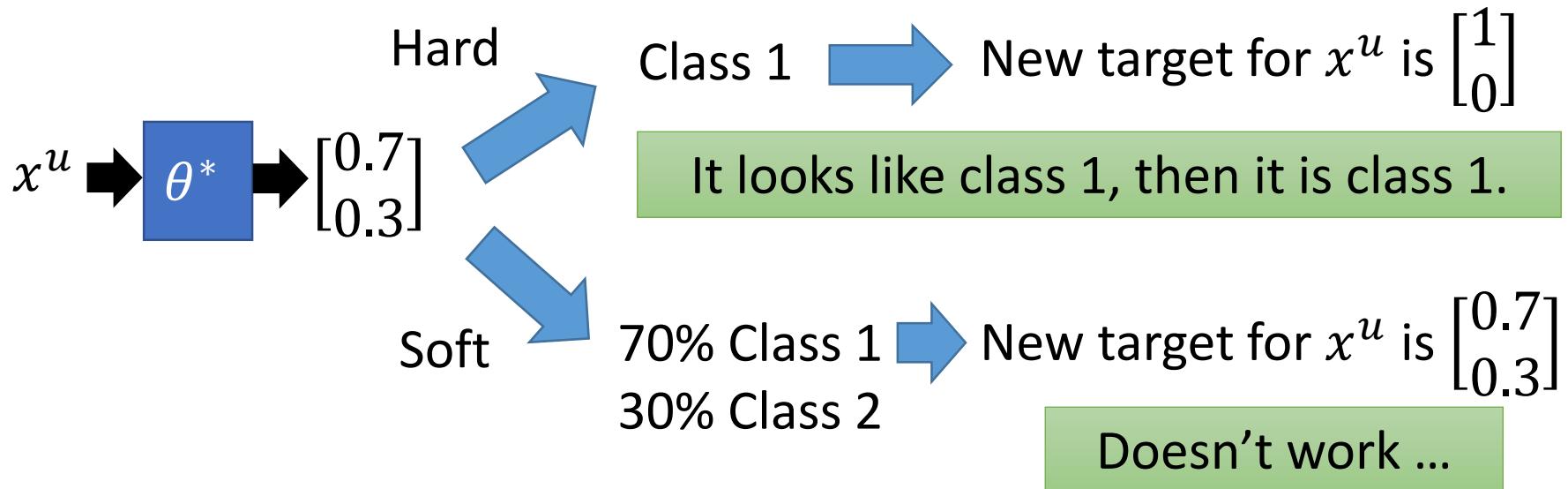
You can also provide a weight to each data.

Self-training

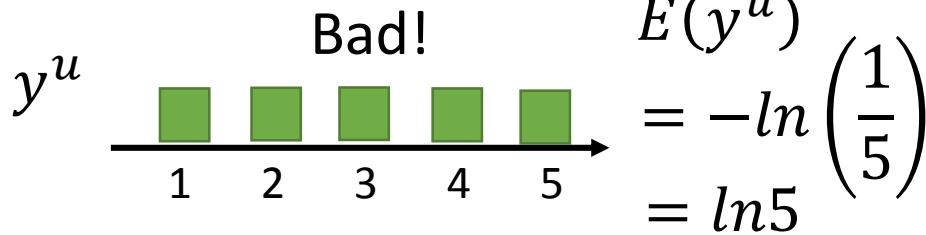
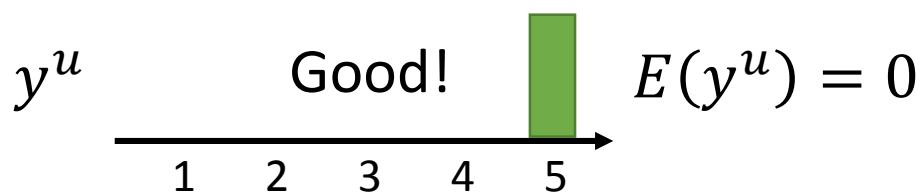
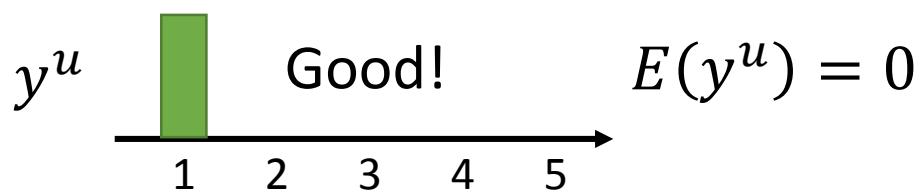
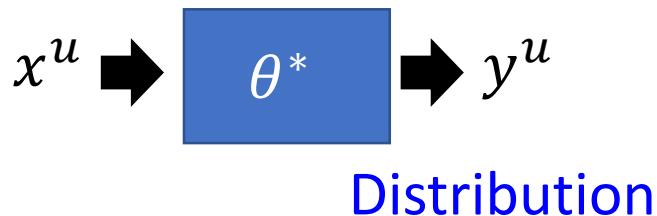
- Similar to semi-supervised learning for generative model
- Hard label v.s. Soft label

Considering using neural network

θ^* (network parameter) from labelled data



Entropy-based Regularization



Entropy of y^u :
Evaluate how concentrate
the distribution y^u is

$$E(y^u) = - \sum_{m=1}^5 y_m^u \ln(y_m^u)$$

As small as possible

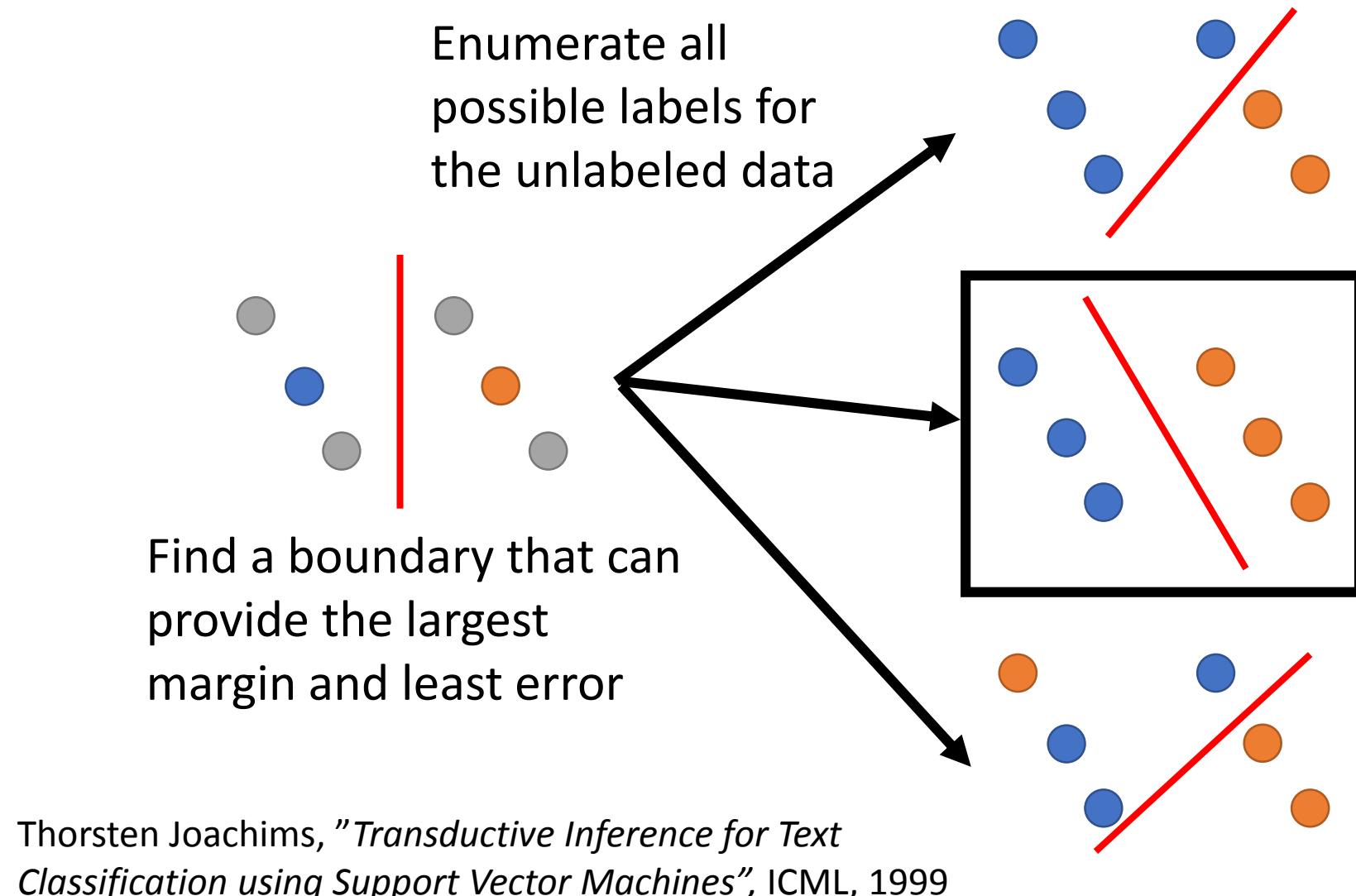
$$L = \sum_{x^r} C(y^r, \hat{y}^r)$$

labelled
data

$$+ \lambda \sum_{x^u} E(y^u)$$

unlabeled
data

Outlook: Semi-supervised SVM



Semi-supervised Learning

Smoothness Assumption

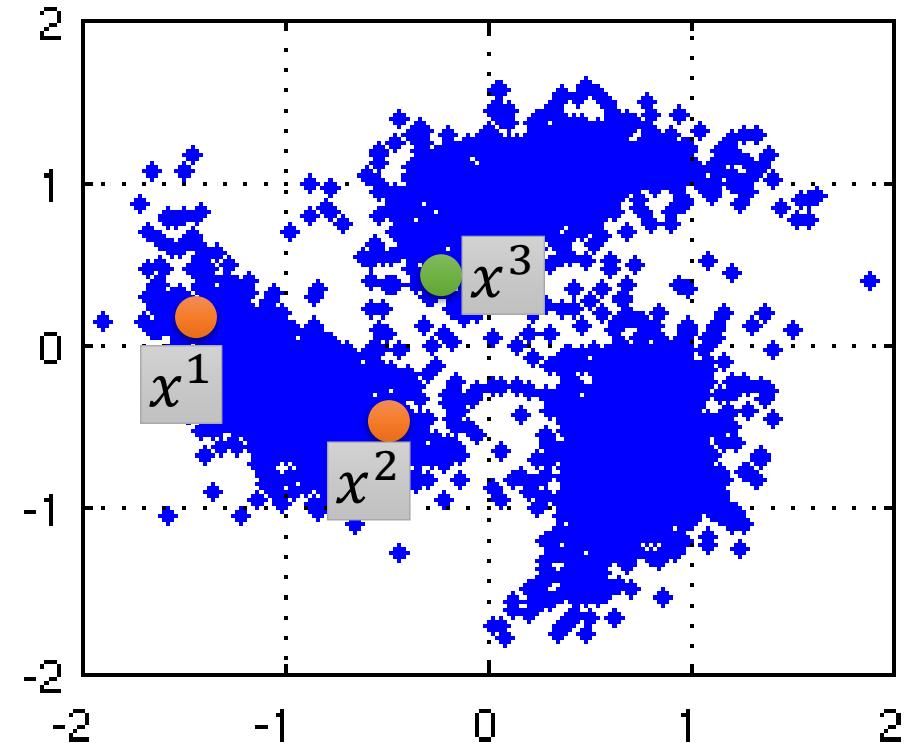
近朱者赤，近墨者黑

“You are known by the company you keep”

Smoothness Assumption

- Assumption: “similar” x has the same \hat{y}
- More precisely:
 - x is not uniform.
 - If x^1 and x^2 are close in a high density region, \hat{y}^1 and \hat{y}^2 are the same.

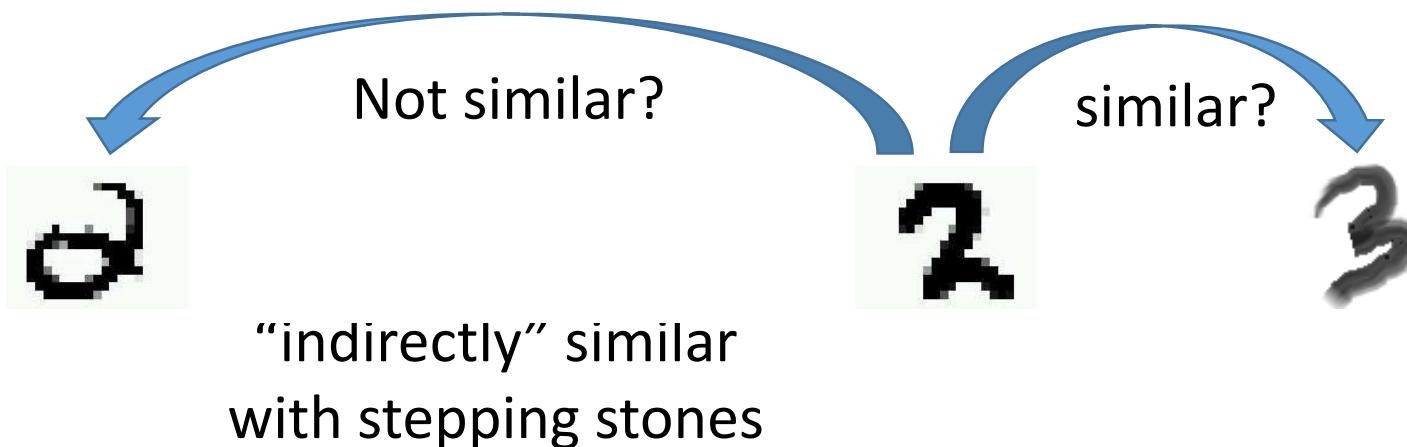
connected by a
high density path



Source of image:
<http://hips.seas.harvard.edu/files/pinwheel.png>

x^1 and x^2 have the same label
 x^2 and x^3 have different labels

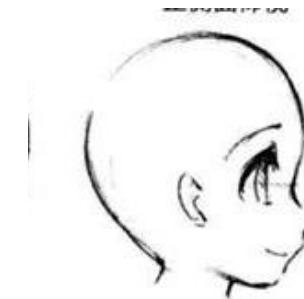
Smoothness Assumption



(The example is from the tutorial slides of Xiaojin Zhu.)



正侧面



正侧面

Source of image: <http://www.moehui.com/5833.html/5/>

Smoothness Assumption

- Classify astronomy vs. travel articles

	d_1	d_3	d_4	d_2
asteroid	•	•		
bright	•	•		
comet		•		
year				
zodiac				
:				
:				
airport				
bike				
camp				
yellowstone		•	•	
zion			•	

	d_1	d_3	d_4	d_2
asteroid	•			
bright	•			
comet				
year				
zodiac				
:				
:				
airport				
bike				
camp				
yellowstone				•
zion				•

(The example is from the tutorial slides of Xiaojin Zhu.)

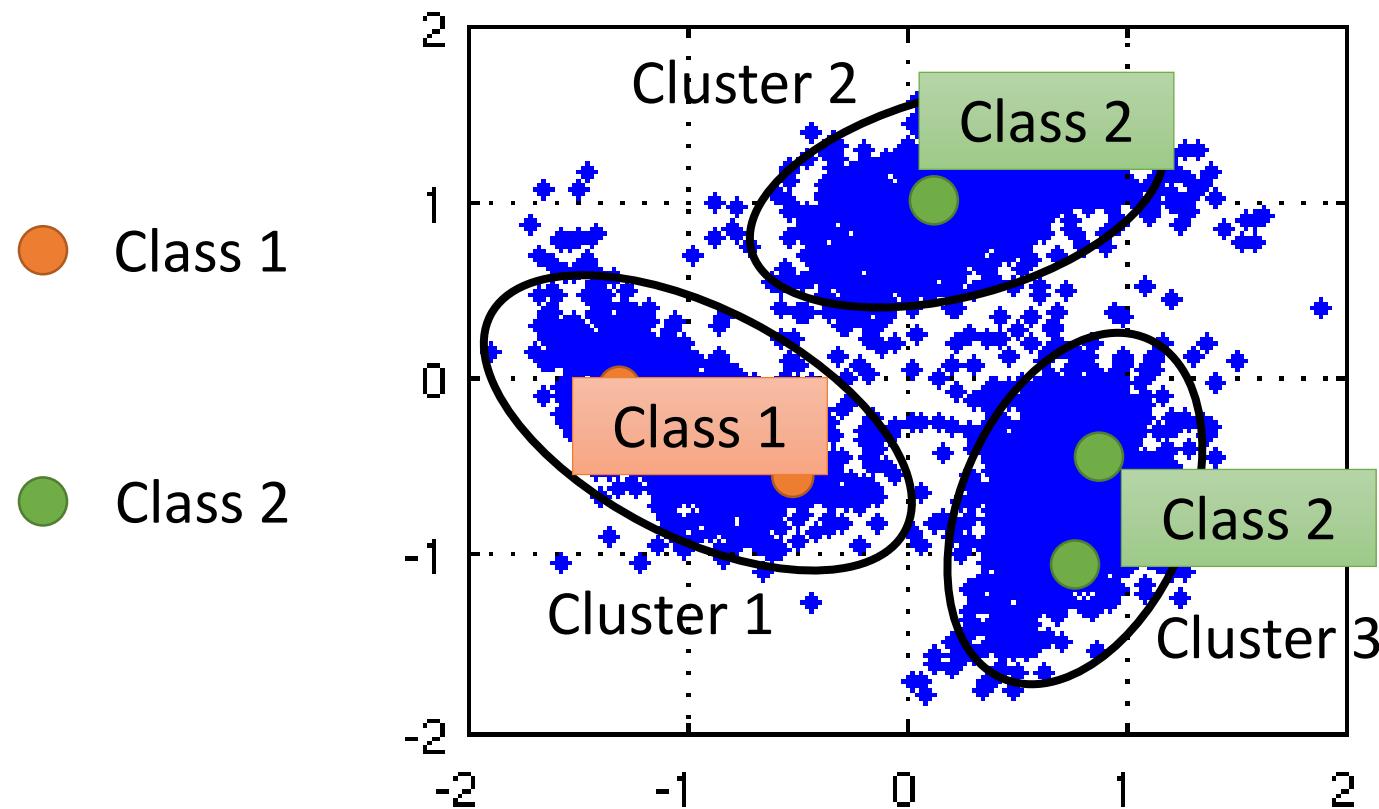
Smoothness Assumption

- Classify astronomy vs. travel articles

	d_1	d_5	d_6	d_7	d_3	d_4	d_8	d_9	d_2
asteroid	•								
bright	•		•						
comet		•		•					
year			•		•				
zodiac					•	•			
.									
.									
airport							•		
bike						•		•	
camp							•		•
yellowstone								•	
zion									•

(The example is from the tutorial slides of Xiaojin Zhu.)

Cluster and then Label



Using all the data to learn a classifier as usual

Graph-based Approach

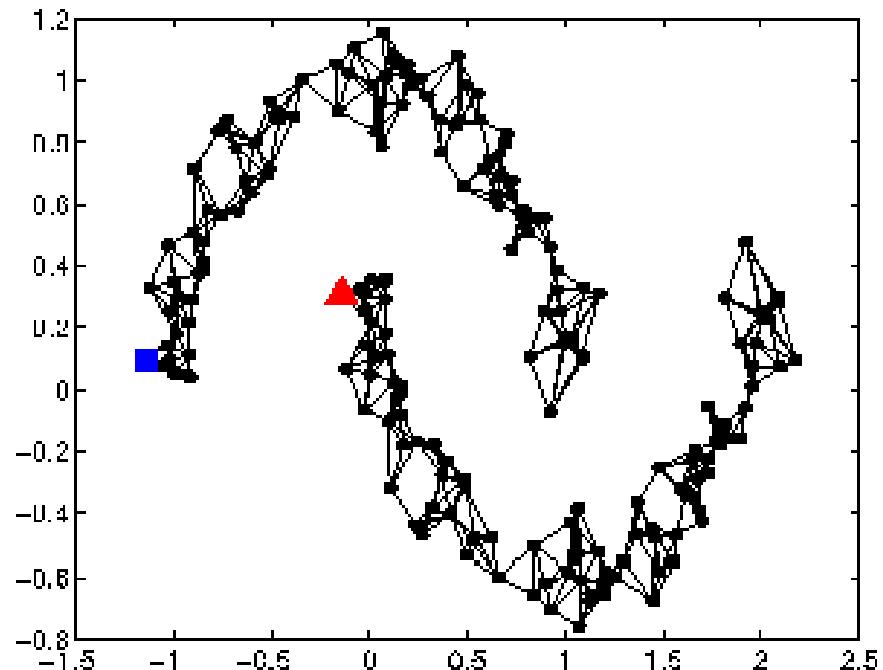
- How to know x^1 and x^2 are close in a high density region (connected by a high density path)

Represented the data points as a *graph*

Graph representation is nature sometimes.

E.g. Hyperlink of webpages, citation of papers

Sometimes you have to construct the graph yourself.



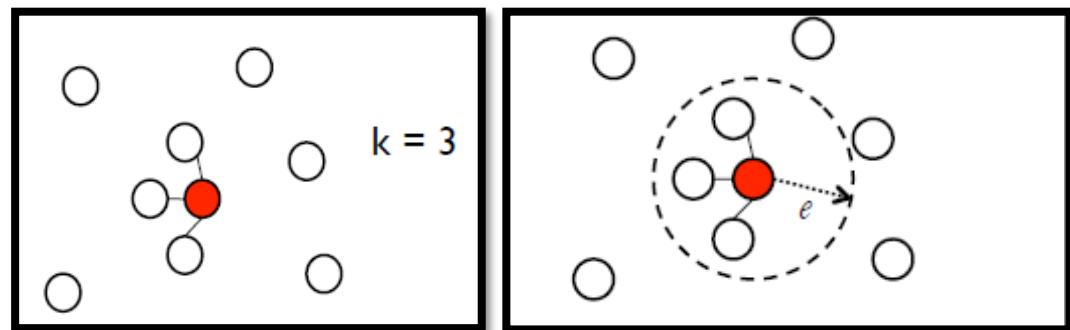
Graph-based Approach

- Graph Construction

The image is from the tutorial slides of Amarnag Subramanya and Partha Pratim Talukdar

- Define the similarity $s(x^i, x^j)$ between x^i and x^j
- Add edge:

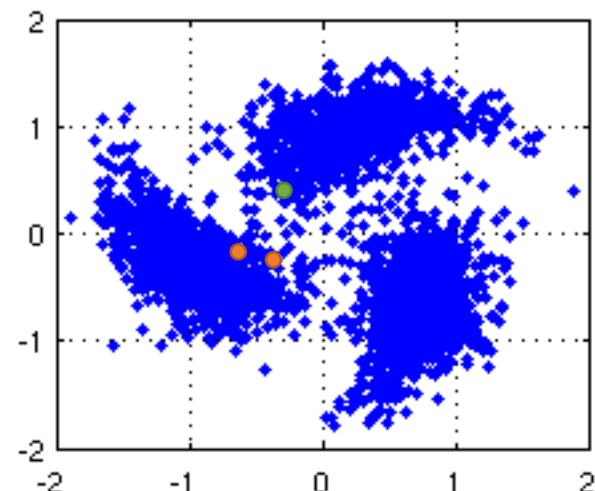
- K Nearest Neighbor
- e -Neighborhood



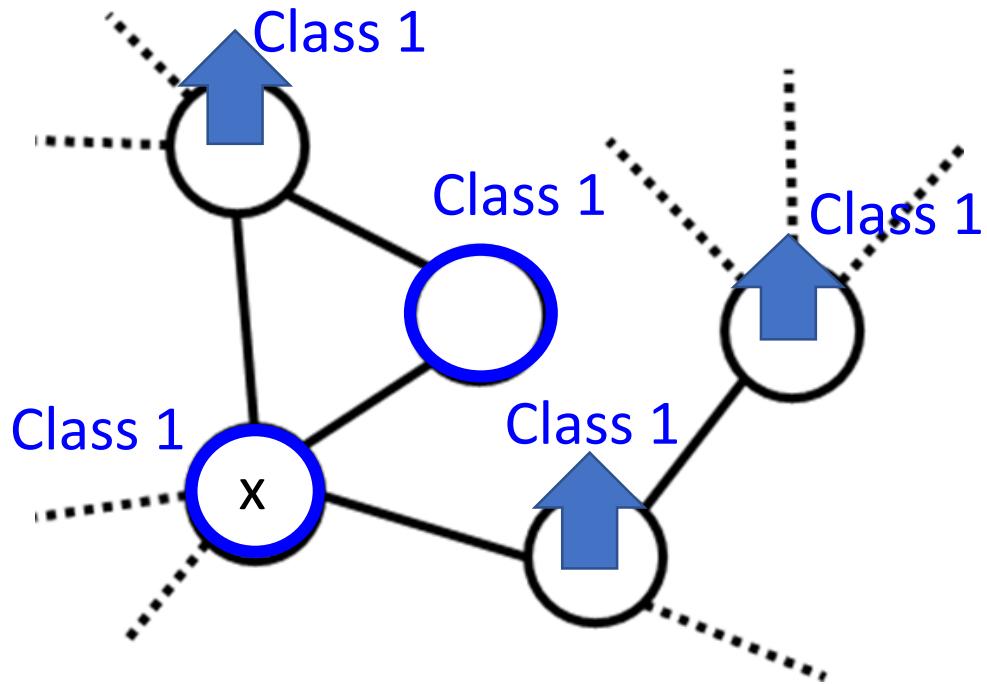
- Edge weight is proportional to $s(x^i, x^j)$

Gaussian Radial Basis Function:

$$s(x^i, x^j) = \exp(-\gamma \|x^i - x^j\|^2)$$

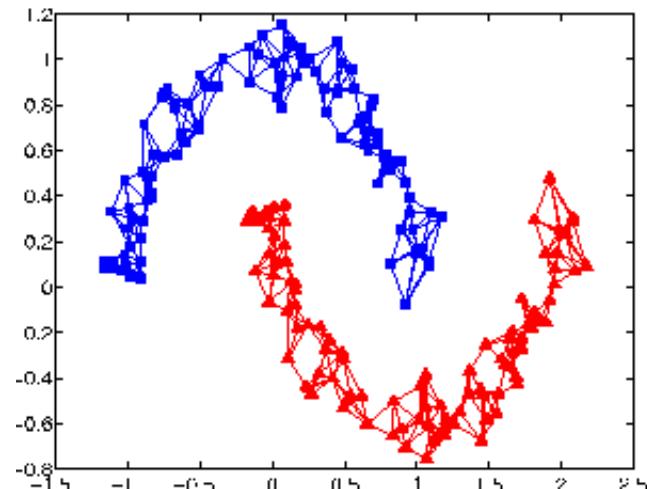
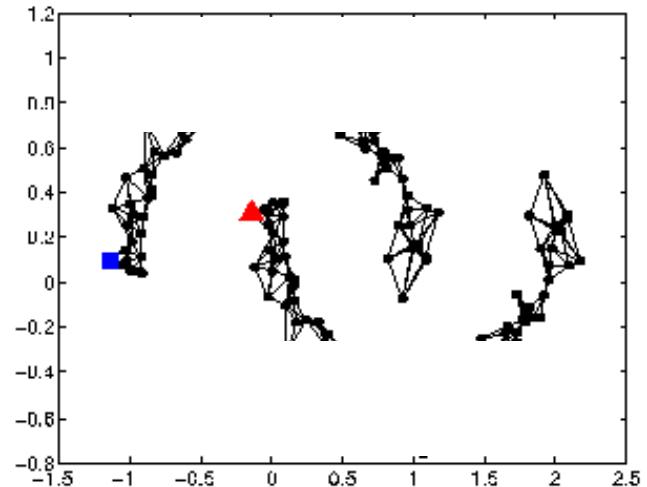


Graph-based Approach



The labelled data influence their neighbors.

Propagate through the graph



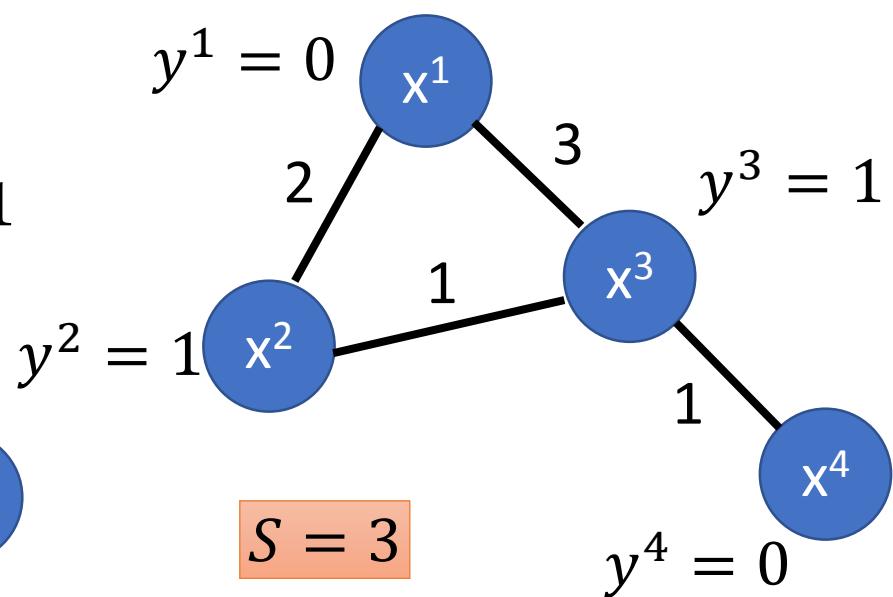
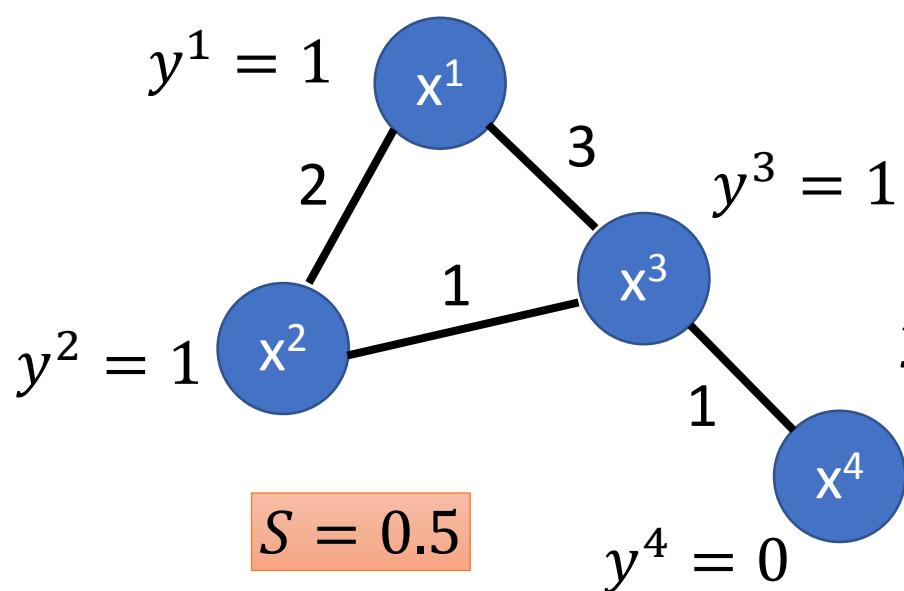
Graph-based Approach

- Define the smoothness of the labels on the graph

$$S = \frac{1}{2} \sum_{i,j} w_{i,j} (y^i - y^j)^2$$

Smaller means smoother

\searrow For all data (no matter labelled or not)



Graph-based Approach

- Define the smoothness of the labels on the graph

$$S = \frac{1}{2} \sum_{i,j} w_{i,j} (y^i - y^j)^2 = \mathbf{y}^T L \mathbf{y}$$

\mathbf{y} : (R+U)-dim vector

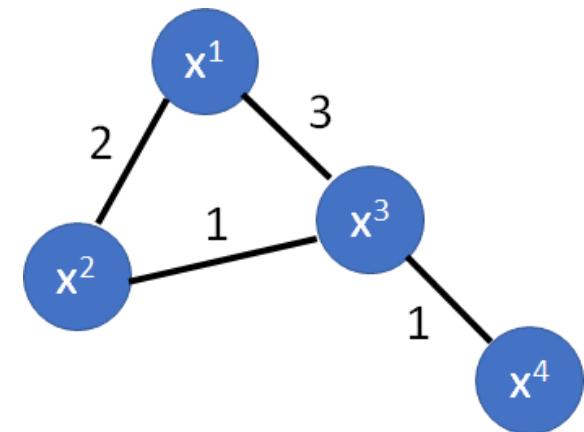
$$\mathbf{y} = [\dots y^i \dots y^j \dots]^T$$

L: (R+U) x (R+U) matrix

Graph Laplacian

$$L = \underline{D} - \underline{W}$$

$$W = \begin{bmatrix} 0 & 2 & 3 & 0 \\ 2 & 0 & 1 & 0 \\ 3 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$
$$D = \begin{bmatrix} 5 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



Graph-based Approach

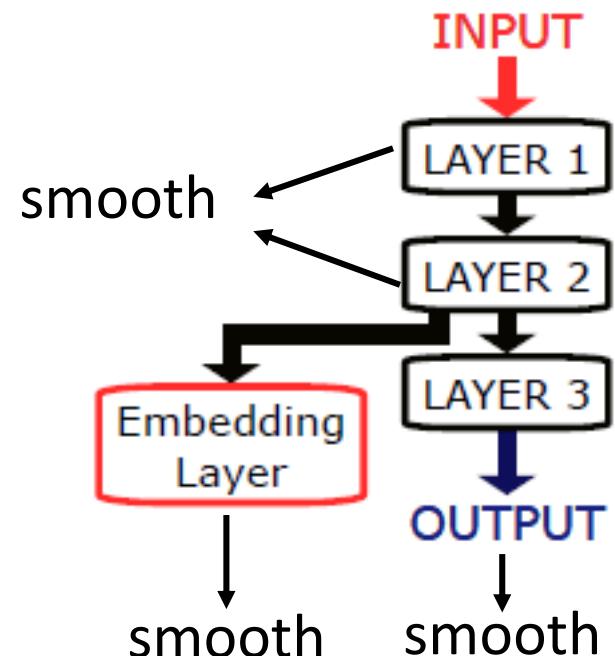
- Define the smoothness of the labels on the graph

$$S = \frac{1}{2} \sum_{i,j} w_{i,j} (y^i - y^j)^2 = \mathbf{y}^T L \mathbf{y}$$

Depending on network parameters

$$L = \sum_{x^r} C(y^r, \hat{y}^r) + \lambda S$$

As a regularization term



J. Weston, F. Ratle, and R. Collobert, “Deep learning via semi-supervised embedding,” ICML, 2008

Semi-supervised Learning

Better Representation

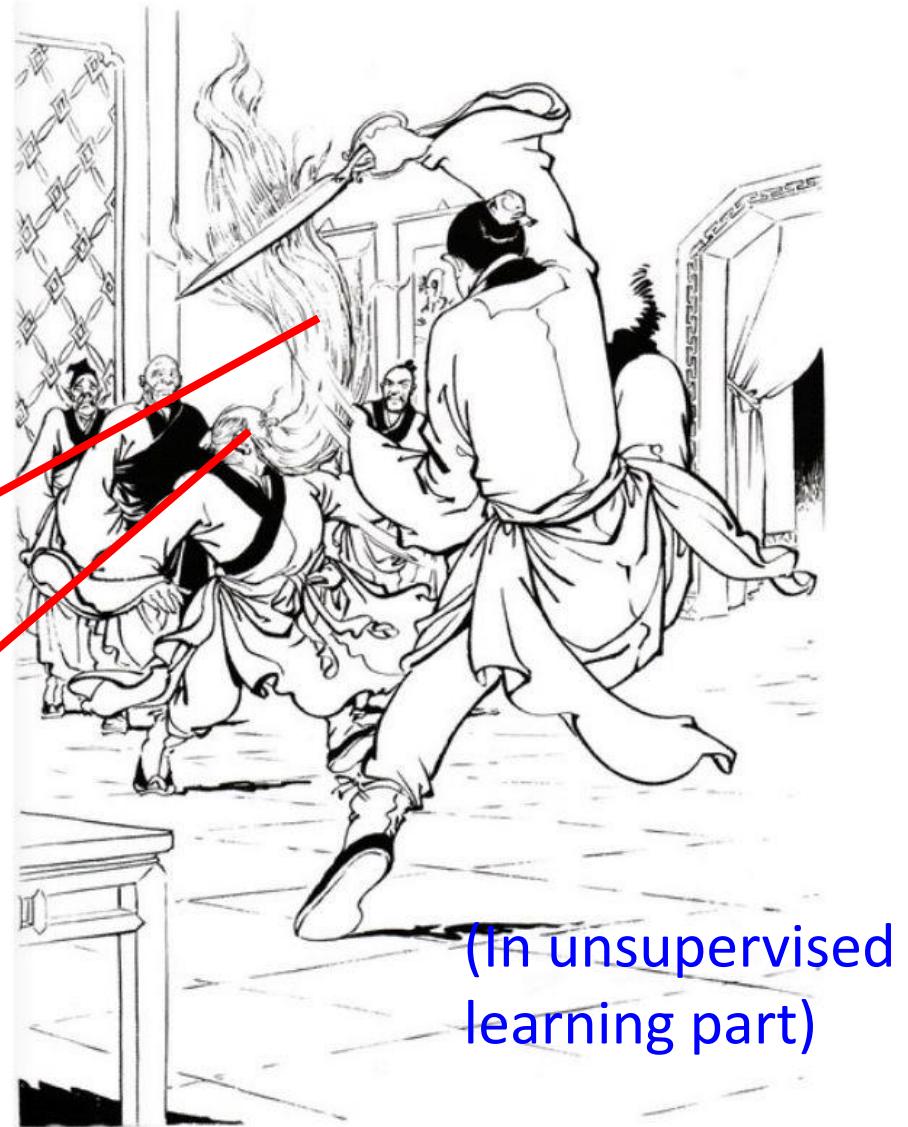
去蕪存菁，化繁為簡

Looking for Better Representation

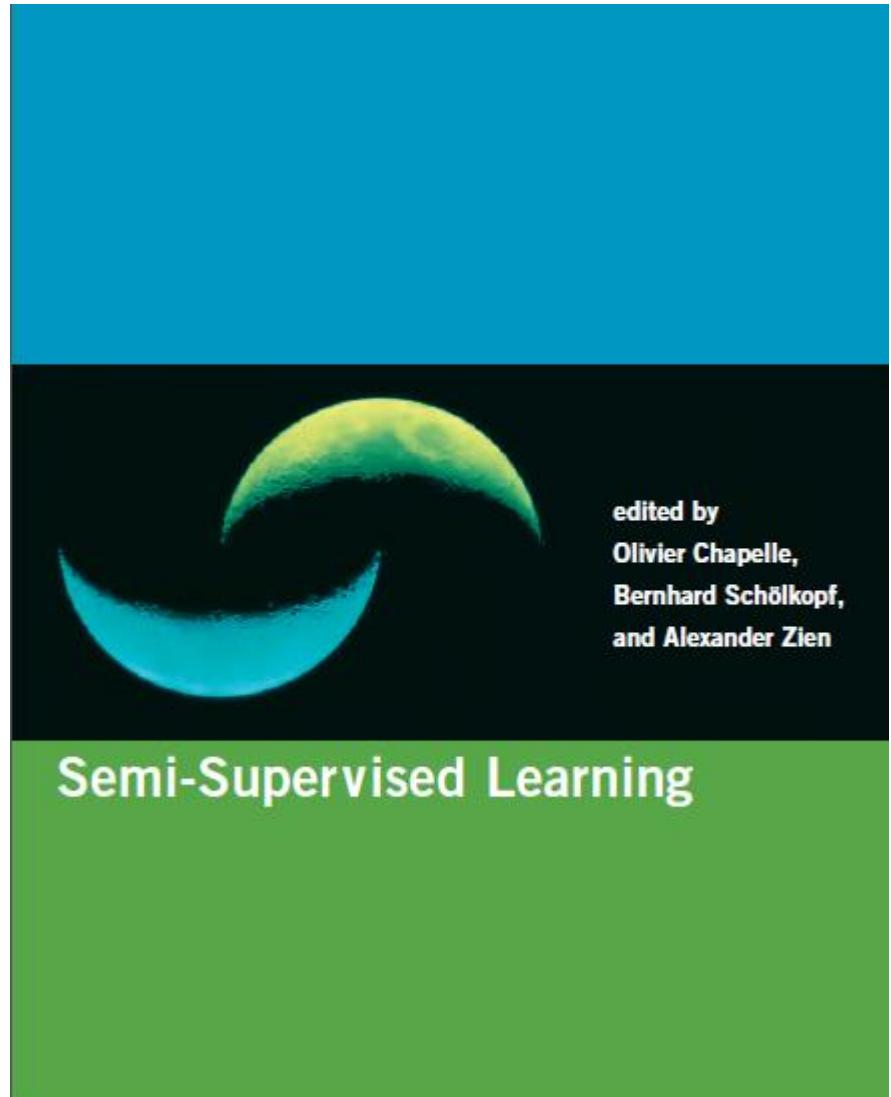
- Find the latent factors behind the observation
- The latent factors (usually simpler) are better representations

observation

Better representation
(Latent factor)



Reference



<http://olivier.chapelle.cc/ssl-book/>

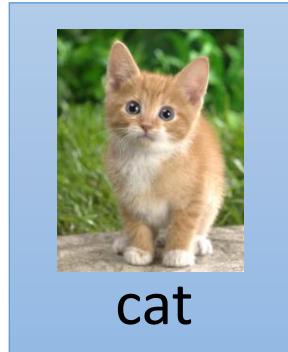
Acknowledgement

- 感謝 劉議隆 同學指出投影片上的錯字
- 感謝 丁勃雄 同學指出投影片上的錯字

Transfer Learning

Transfer Learning

Dog/Cat
Classifier



<http://weebly110810.weebly.com/396403913129399.html>

<http://www.sucaitianxia.com/png/cartoon/200811/4261.html>



Data *not directly related to* the task considered



elephant

tiger

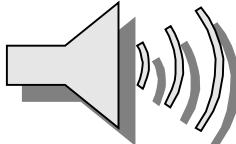
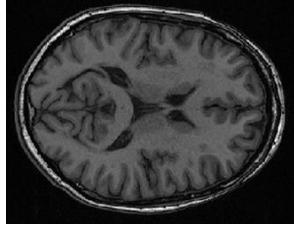
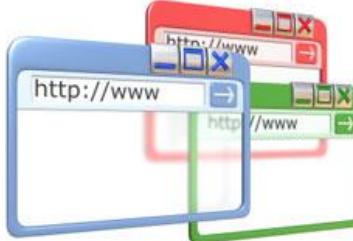


Similar domain, different tasks

Different domains, same task

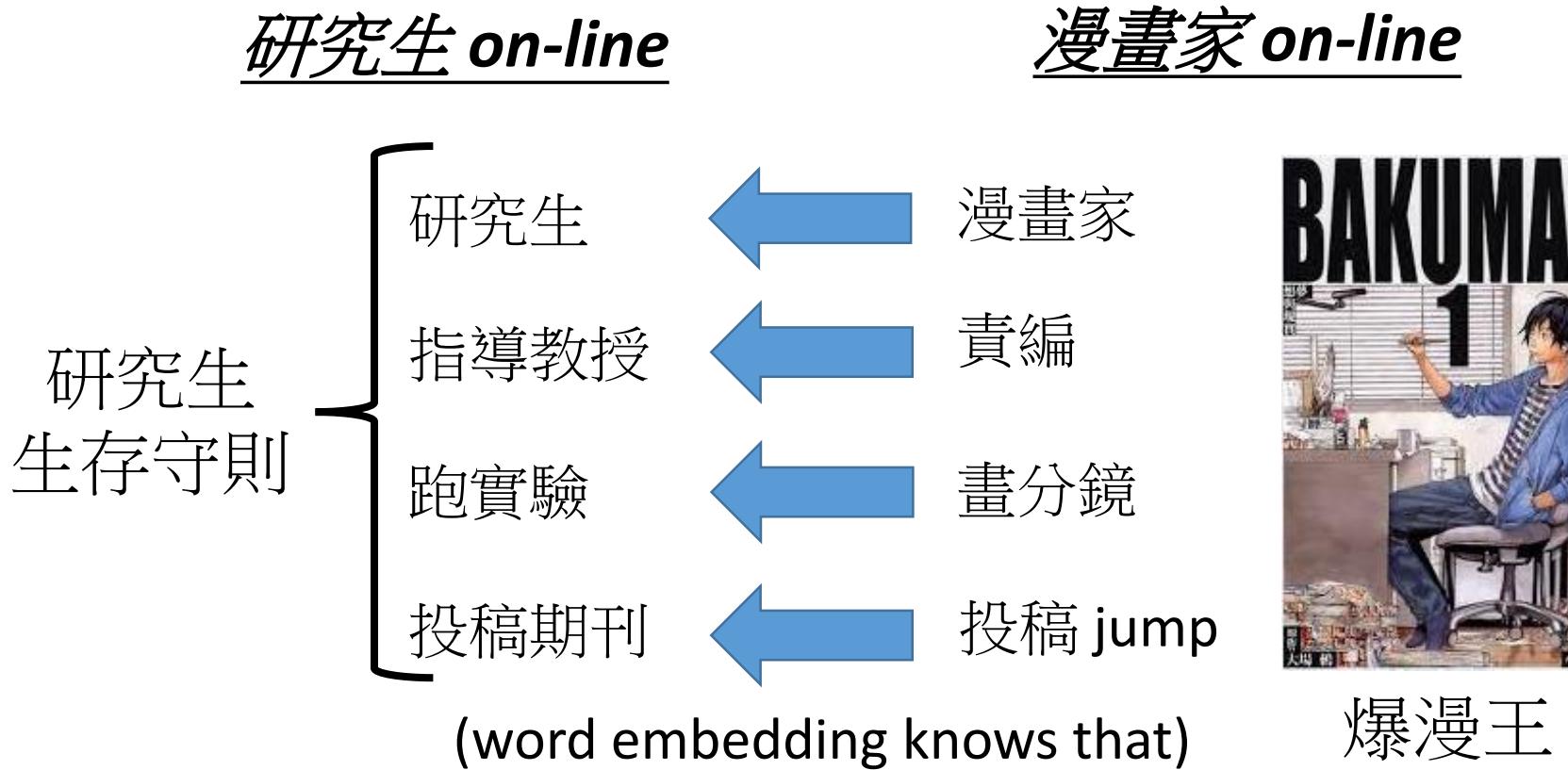
Why?

<http://www.bigr.nl/website/structure/main.php?page=researchlines&subpage=project&id=64>
<http://www.spear.com.hk/Translation-company-Directory.html>

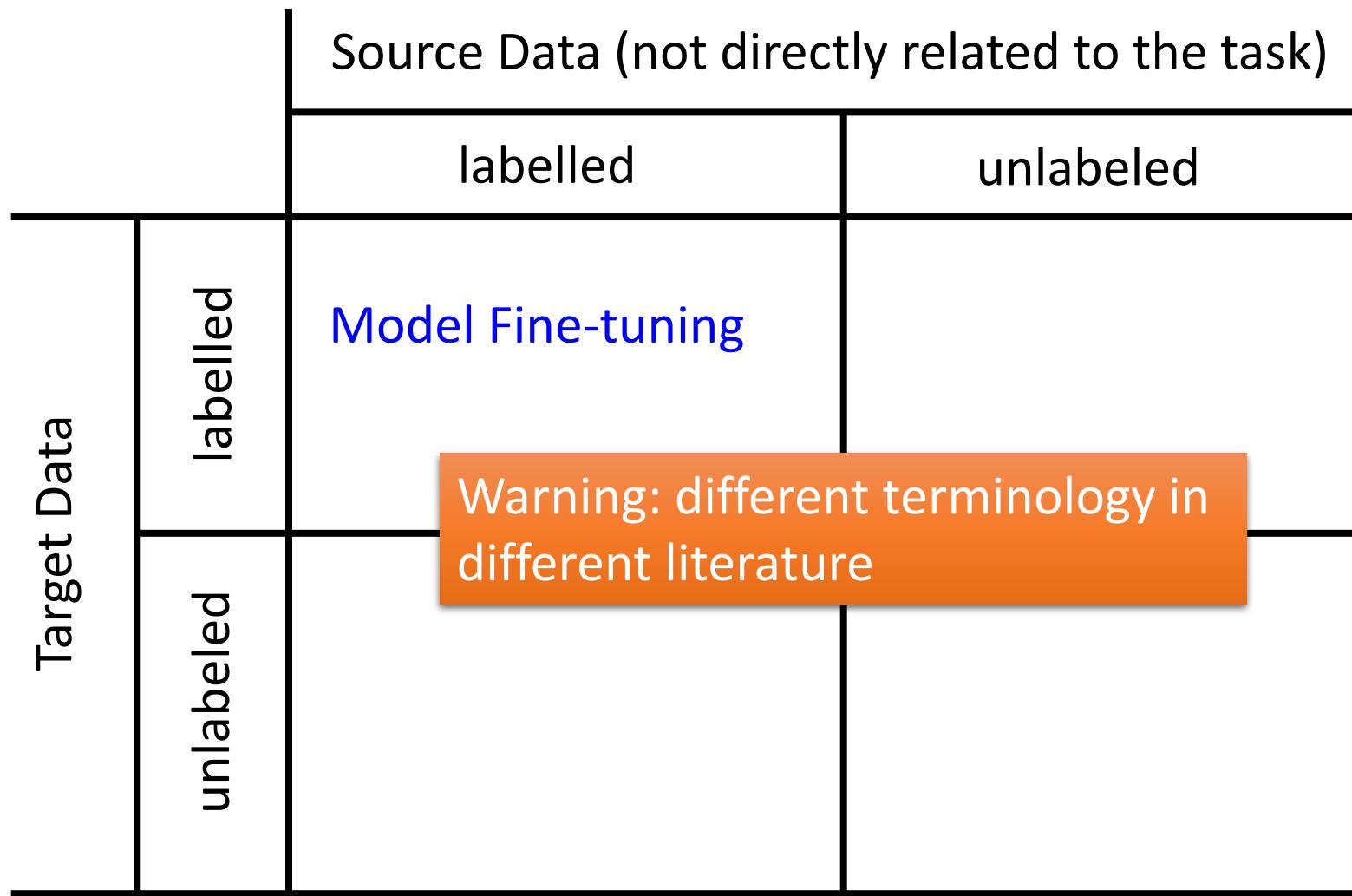
Task Considered	Data not directly related
Speech Recognition	 Taiwanese  English Chinese
Image Recognition	 Medical Images 
Text Analysis	 Specific domain  Webpages

Transfer Learning

- Example in real life



Transfer Learning - Overview

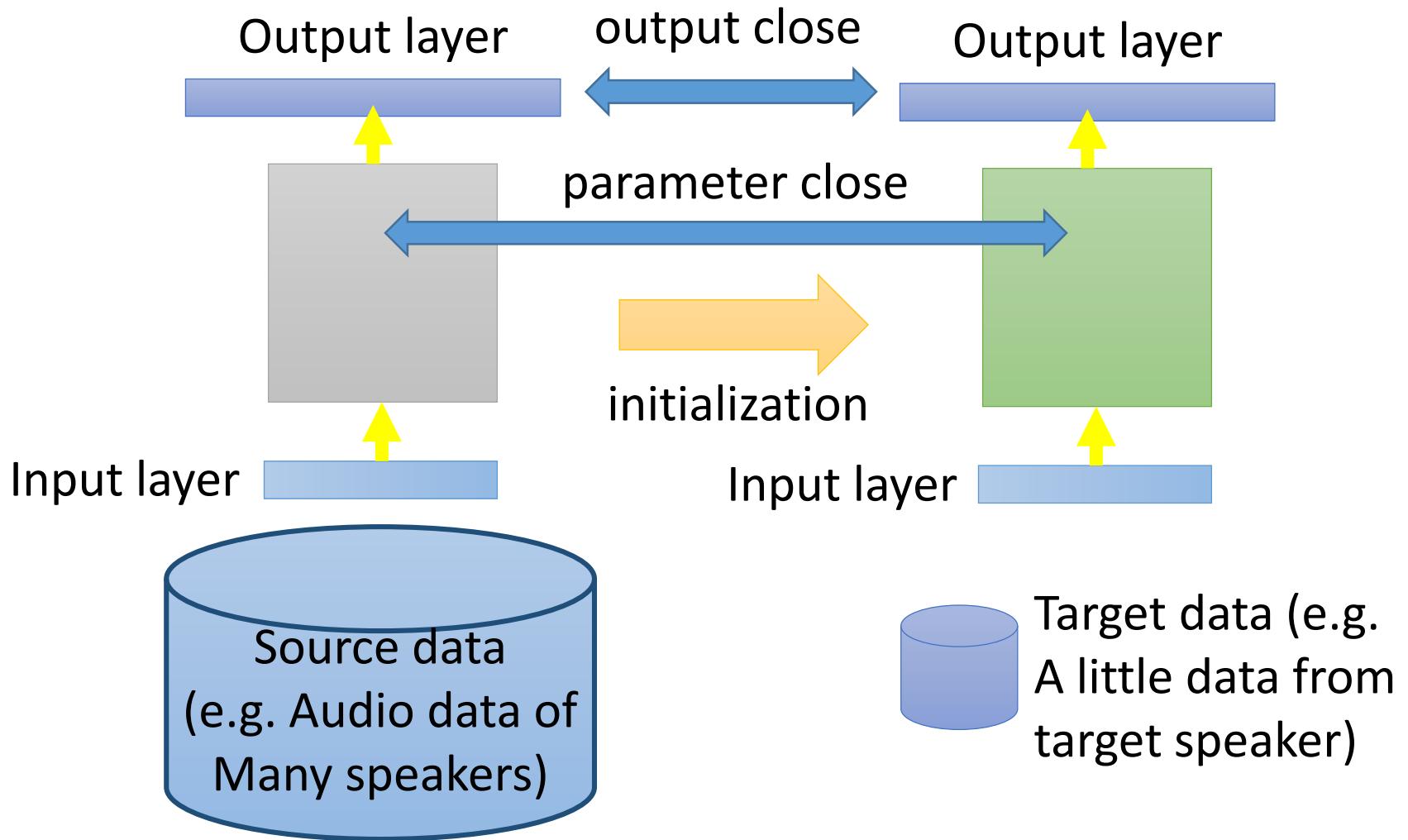


Model Fine-tuning

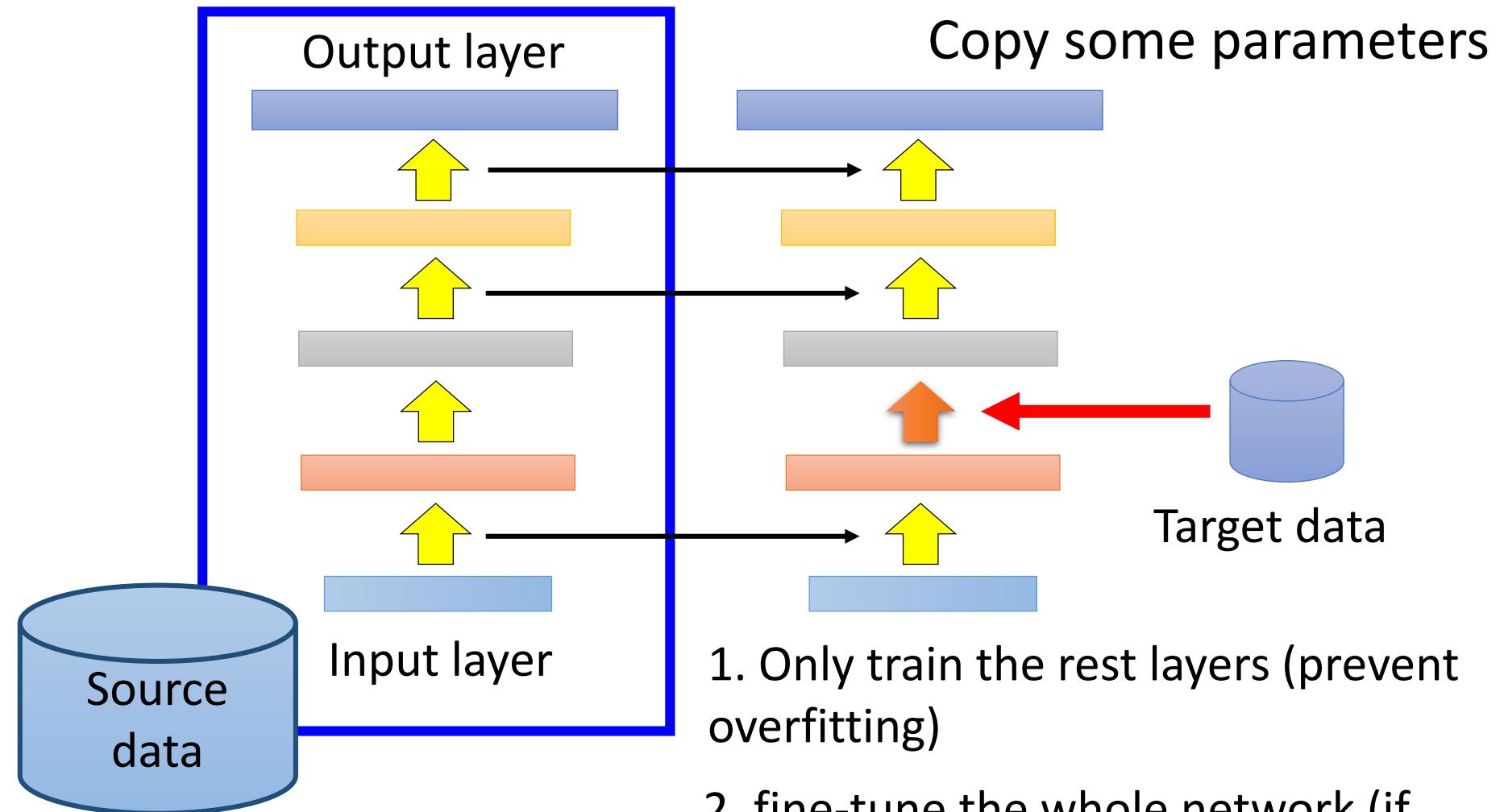
One-shot learning: only a few examples in target domain

- Task description
 - Target data: (x^t, y^t) ← Very little
 - Source data: (x^s, y^s) ← A large amount
- Example: (supervised) speaker adaption
 - Target data: audio data and its transcriptions of specific user
 - Source data: audio data and transcriptions from many speakers
- Idea: training a model by source data, then fine-tune the model by target data
 - Challenge: only limited target data, so be careful about overfitting

Conservative Training

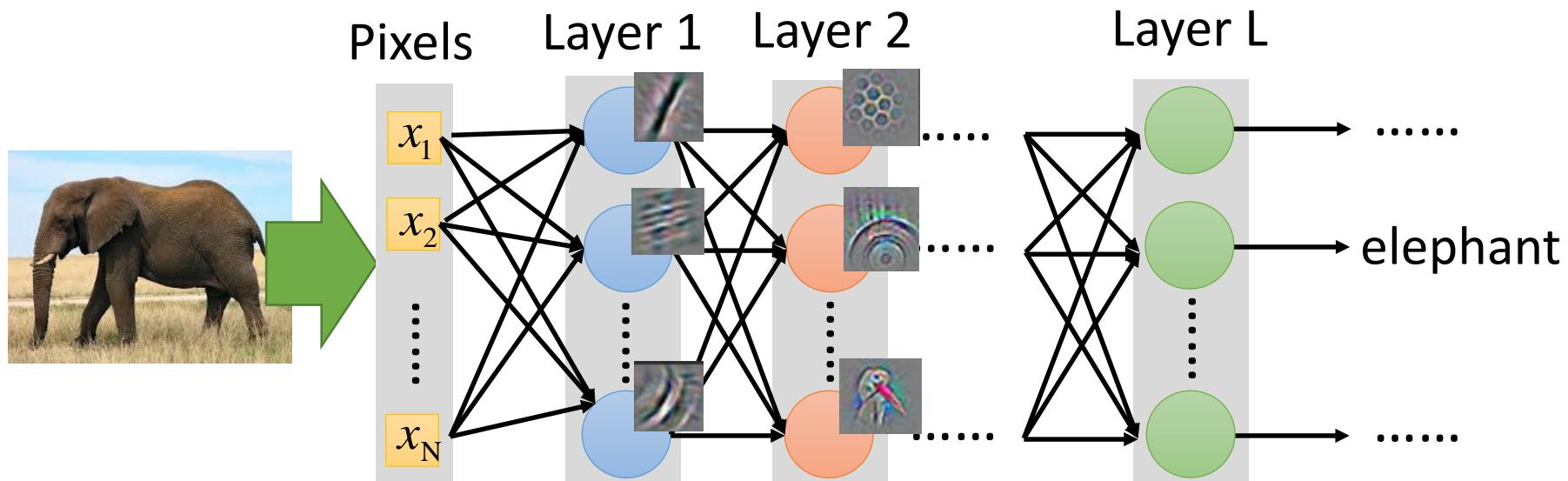


Layer Transfer

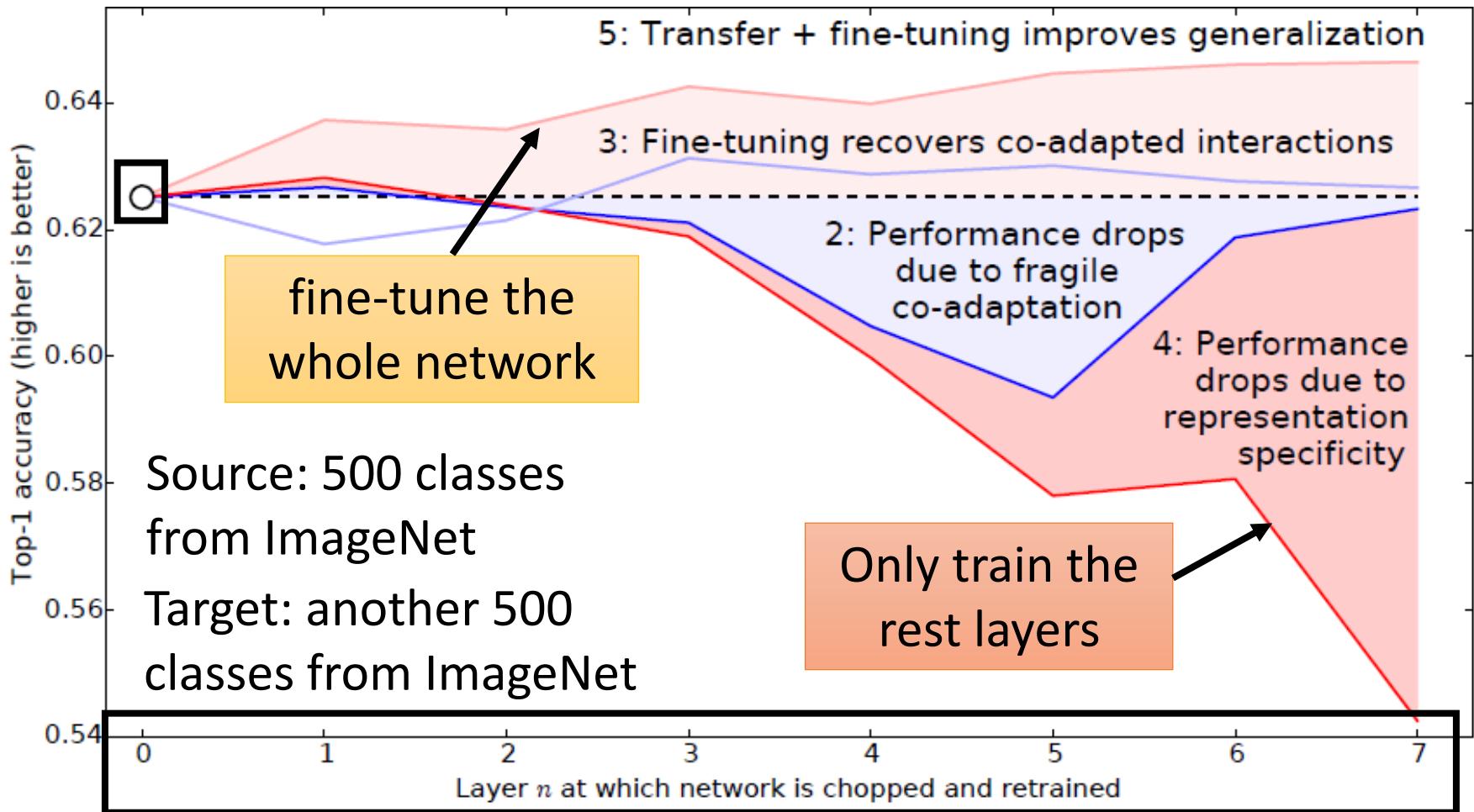


Layer Transfer

- Which layer can be transferred (copied)?
 - Speech: usually copy the last few layers
 - Image: usually copy the first few layers

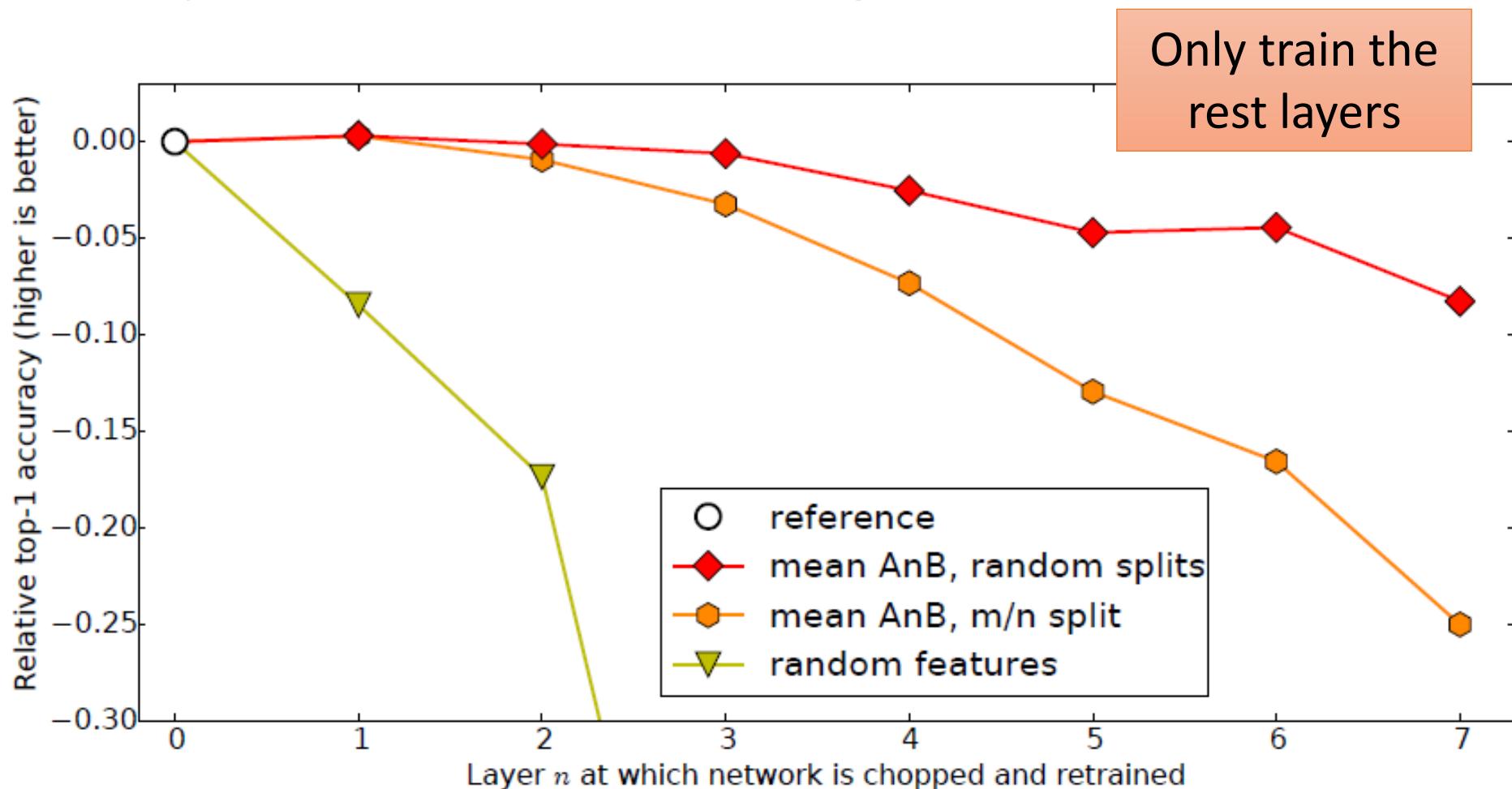


Layer Transfer - Image



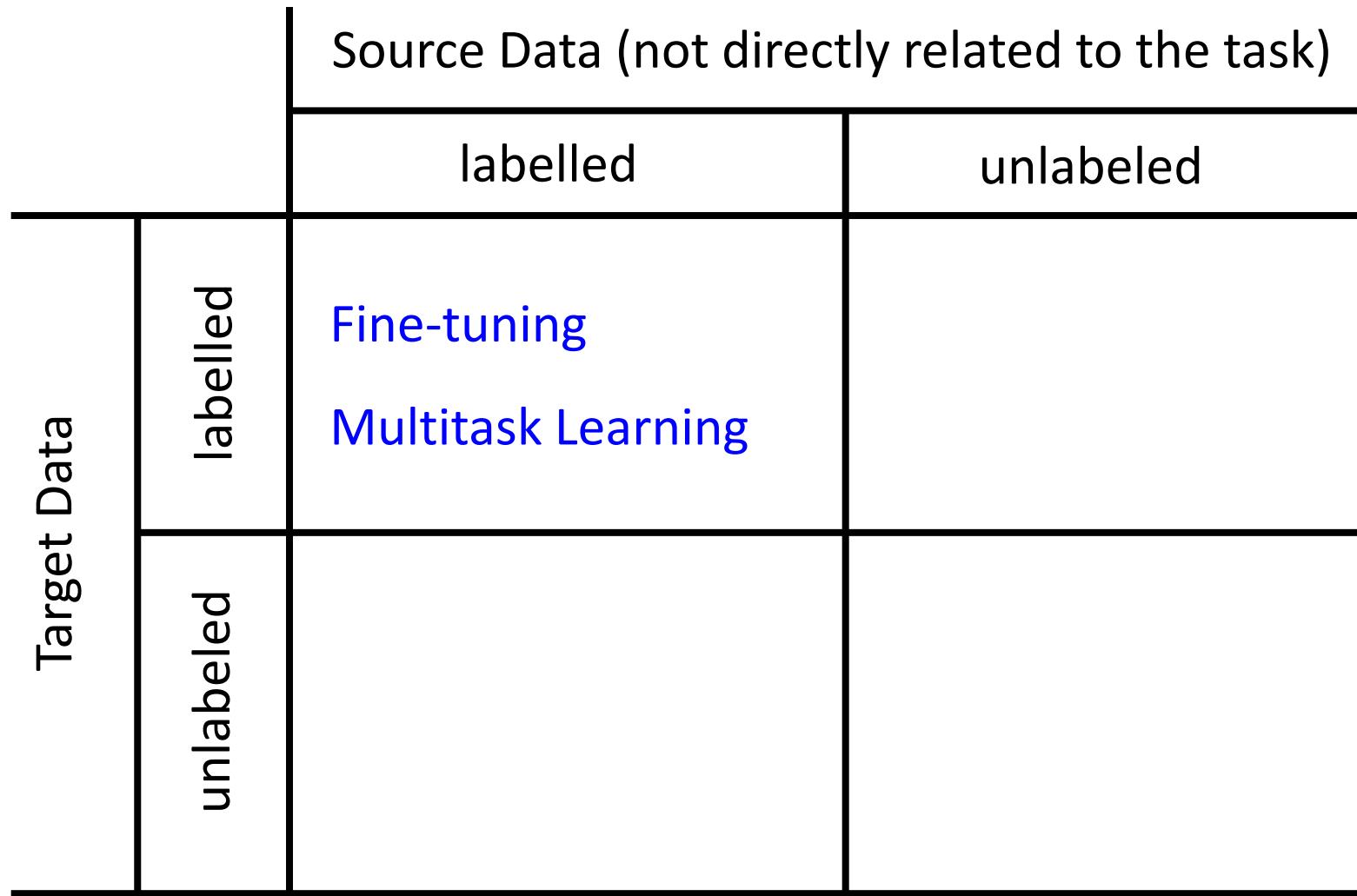
Jason Yosinski, Jeff Clune, Yoshua Bengio, Hod Lipson, "How transferable are features in deep neural networks?", NIPS, 2014

Layer Transfer - Image



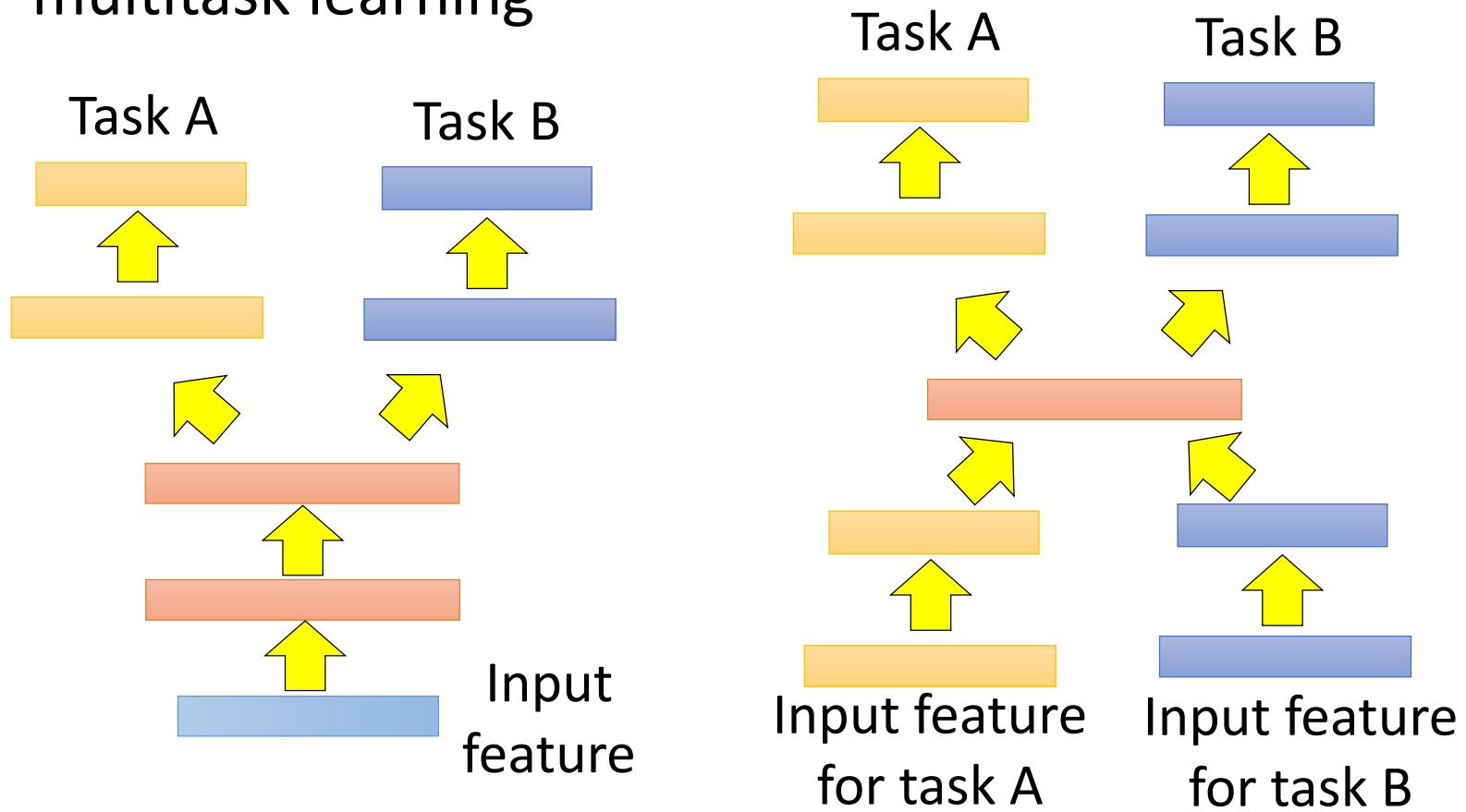
Jason Yosinski, Jeff Clune, Yoshua Bengio, Hod Lipson, "How transferable are features in deep neural networks?", NIPS, 2014

Transfer Learning - Overview



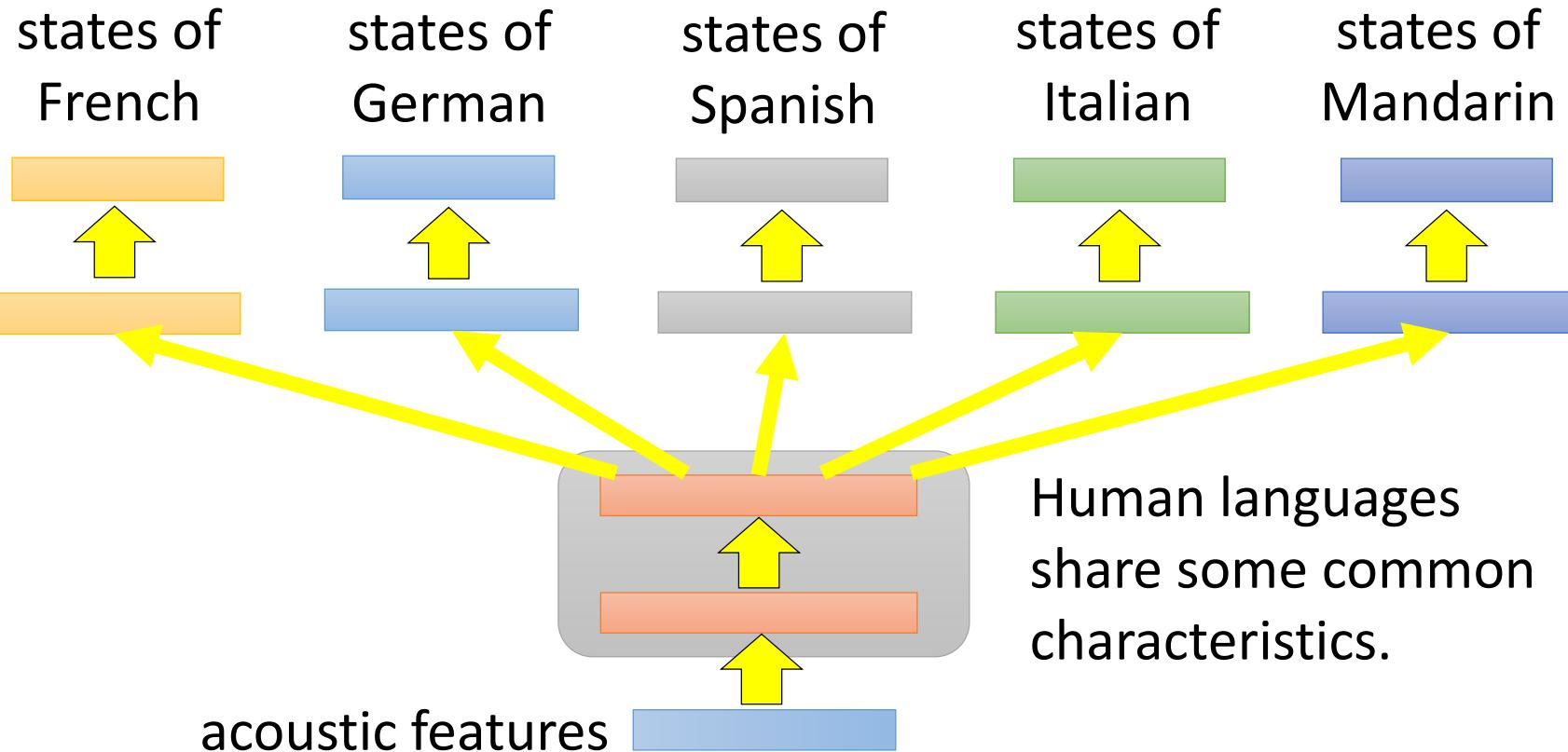
Multitask Learning

- The multi-layer structure makes NN suitable for multitask learning



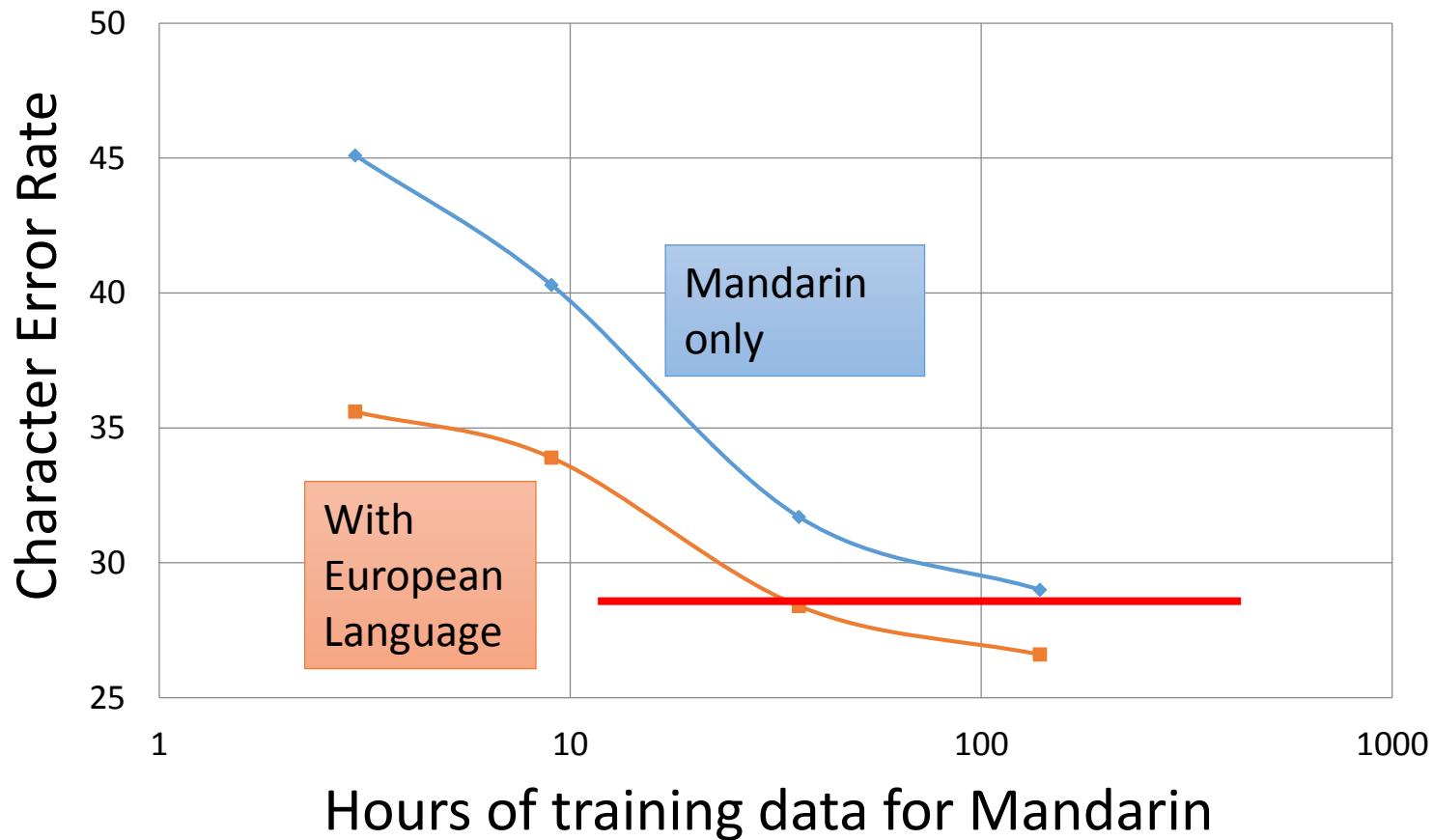
Multitask Learning

- Multilingual Speech Recognition



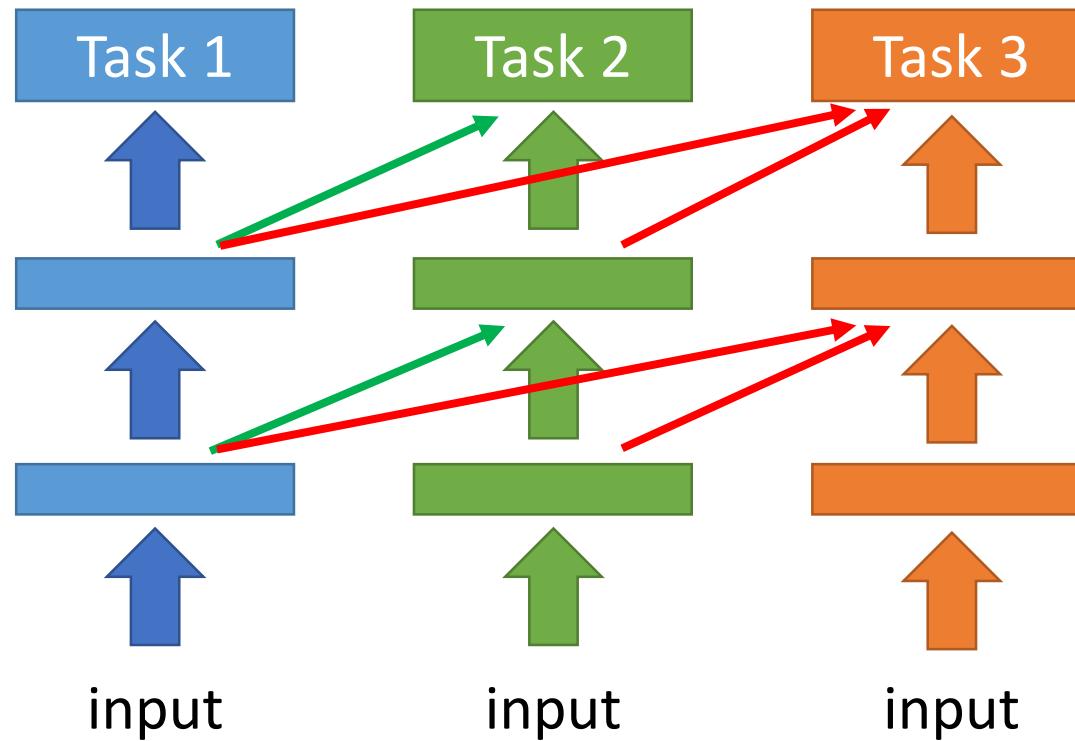
Similar idea in translation: Daxiang Dong, Hua Wu, Wei He, Dianhai Yu and Haifeng Wang, "Multi-task learning for multiple language translation.", ACL 2015

Multitask Learning - Multilingual



Huang, Jui-Ting, et al. "Cross-language knowledge transfer using multilingual deep neural network with shared hidden layers." *ICASSP*, 2013

Progressive Neural Networks



Andrei A. Rusu, Neil C. Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, Raia Hadsell, "Progressive Neural Networks", arXiv preprint 2016

Transfer Learning - Overview

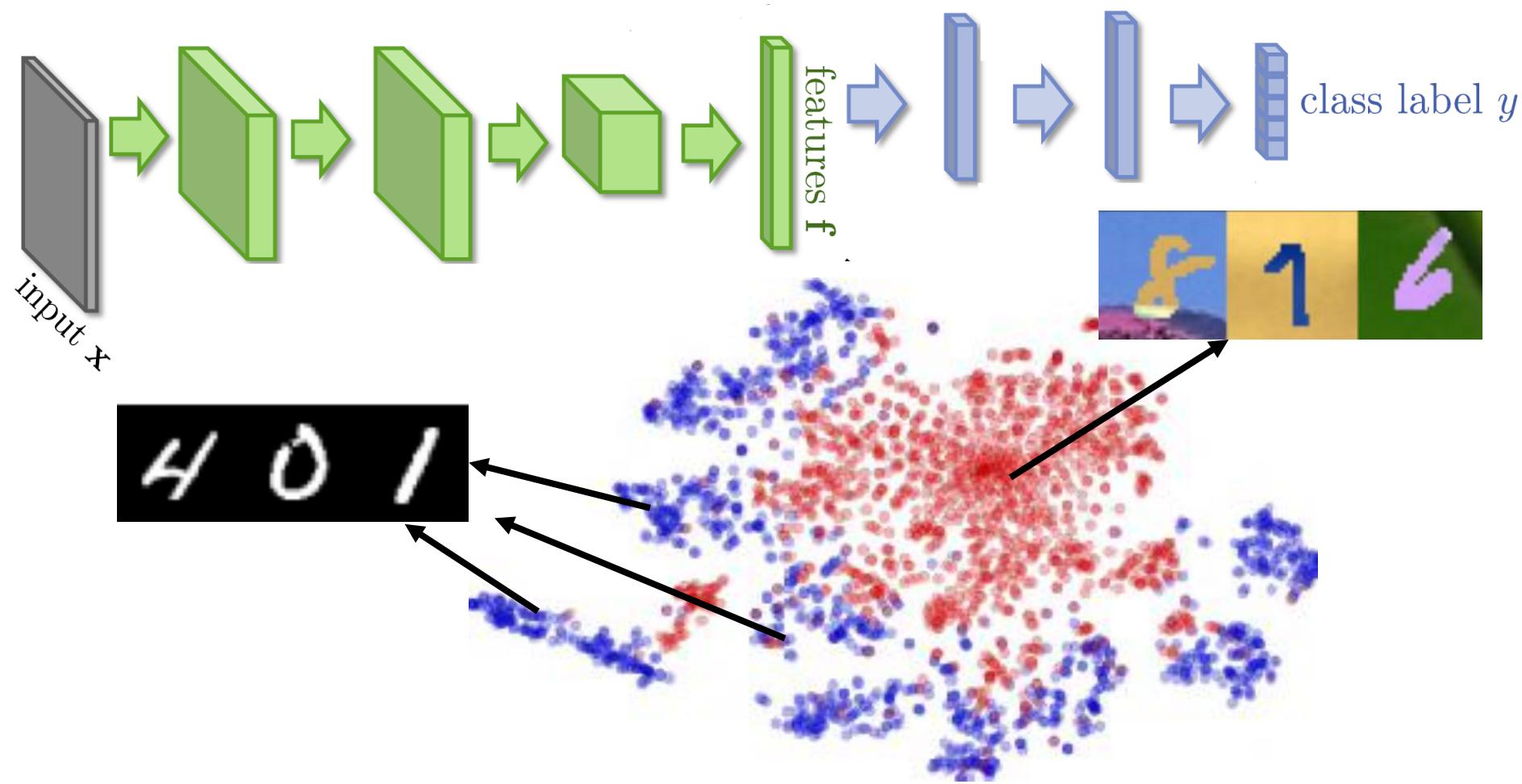
		Source Data (not directly related to the task)	
		labelled	unlabeled
Target Data	labelled	Fine-tuning Multitask Learning	
	unlabeled	Domain-adversarial training	

Task description

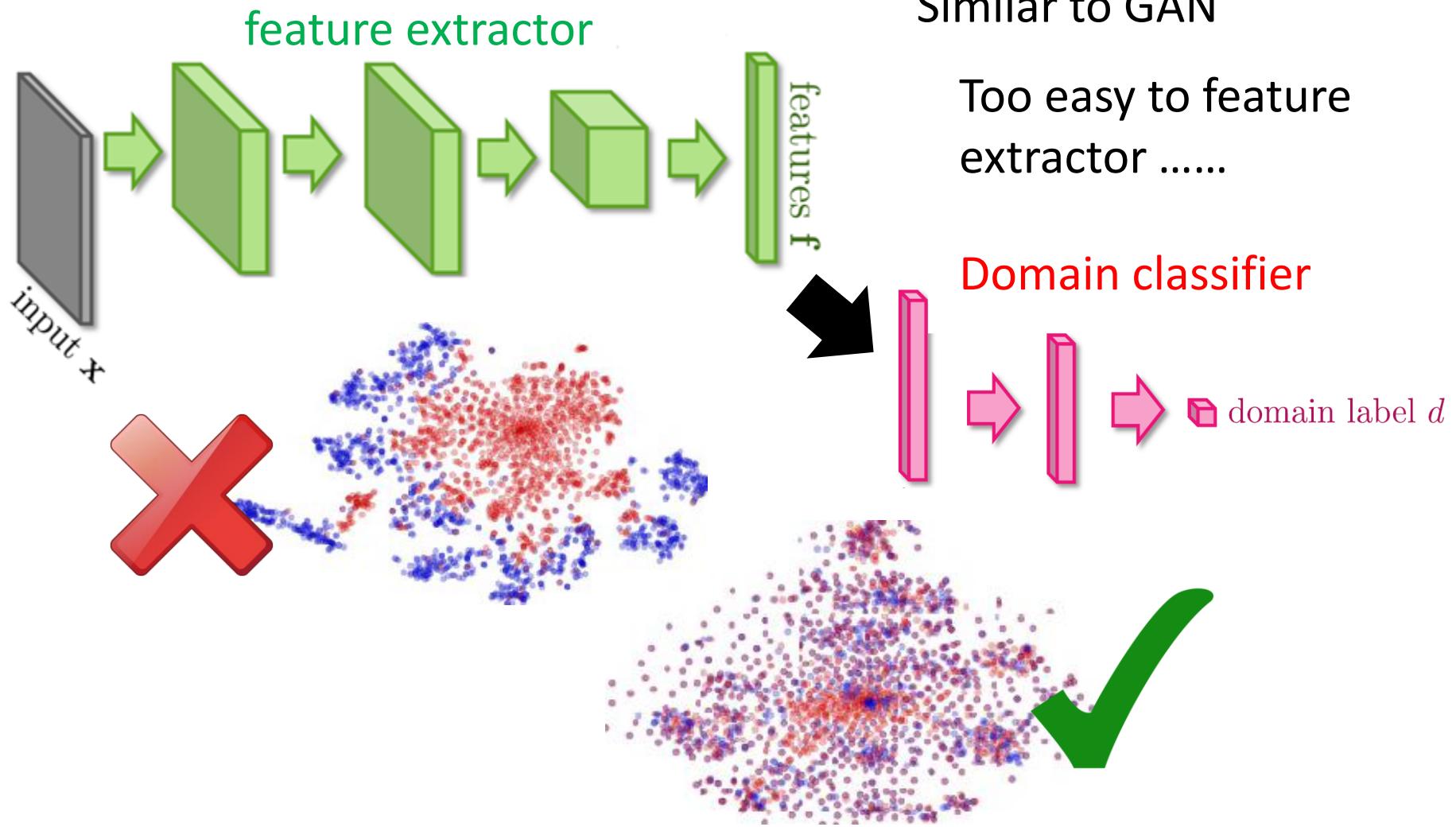
- Source data: $(x^s, y^s) \rightarrow$ Training data
 - Target data: $(x^t) \rightarrow$ Testing data
- } mismatch



Domain-adversarial training

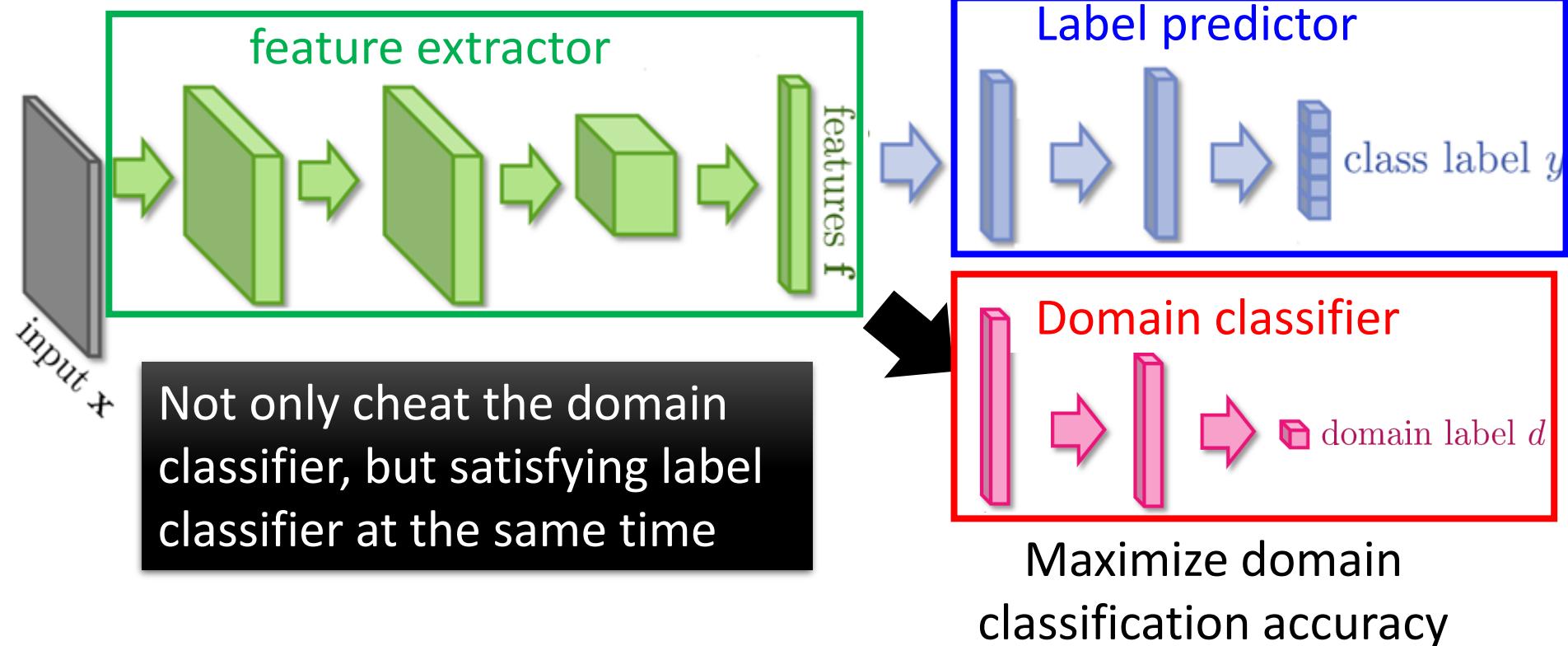


Domain-adversarial training



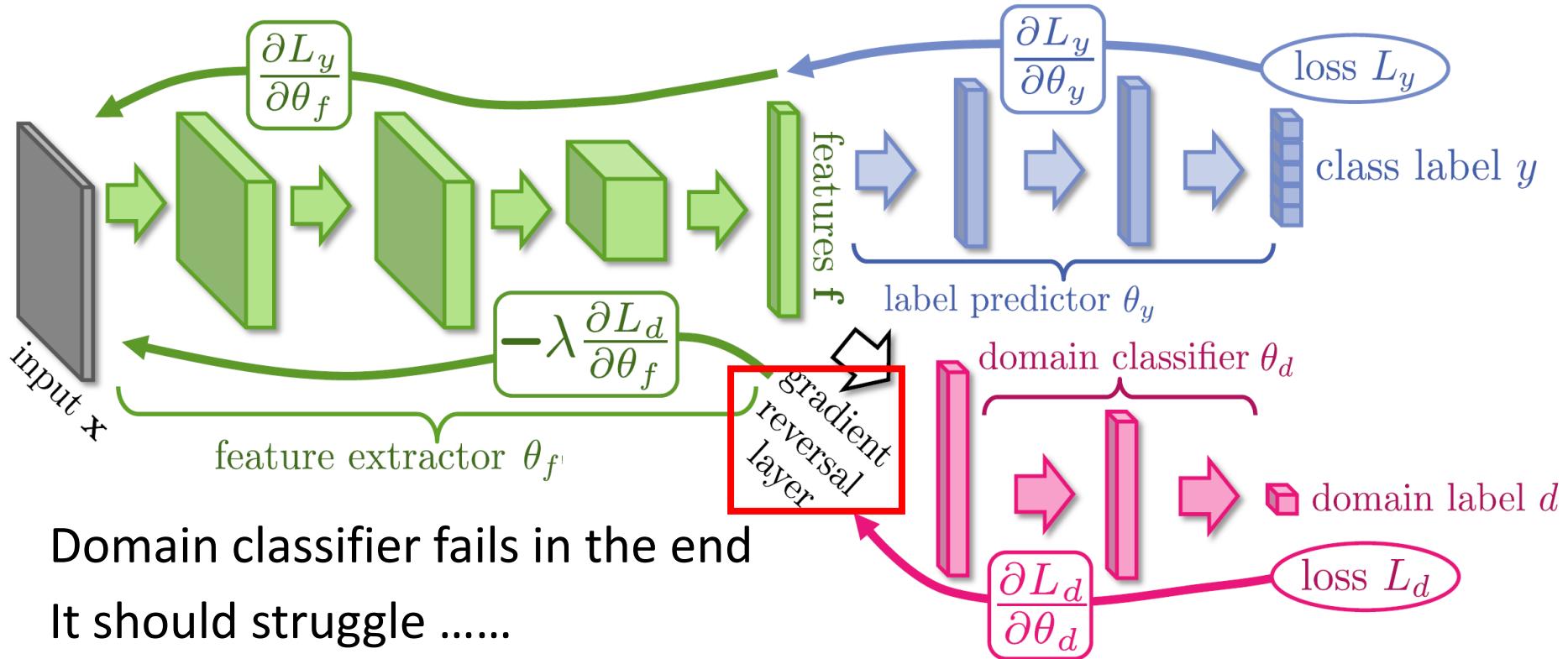
Domain-adversarial training

Maximize label classification accuracy +
minimize domain classification accuracy



This is a big network, but different parts have different goals.

Domain-adversarial training



Domain classifier fails in the end
It should struggle

Yaroslav Ganin, Victor Lempitsky, Unsupervised Domain Adaptation by Backpropagation, ICML, 2015

Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, Domain-Adversarial Training of Neural Networks, JMLR, 2016

Domain-adversarial training



METHOD	SOURCE	MNIST	SYN NUMBERS	SVHN	SYN SIGNS
	TARGET	MNIST-M	SVHN	MNIST	GTSRB
SOURCE ONLY		.5749	.8665	.5919	.7400
SA (FERNANDO ET AL., 2013)		.6078 (7.9%)	.8672 (1.3%)	.6157 (5.9%)	.7635 (9.1%)
PROPOSED APPROACH		.8149 (57.9%)	.9048 (66.1%)	.7107 (29.3%)	.8866 (56.7%)
TRAIN ON TARGET		.9891	.9244	.9951	.9987

Yaroslav Ganin, Victor Lempitsky, Unsupervised Domain Adaptation by Backpropagation, ICML, 2015

Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, Domain-Adversarial Training of Neural Networks, JMLR, 2016

Transfer Learning - Overview

		Source Data (not directly related to the task)	
		labelled	unlabeled
Target Data	labelled	Fine-tuning Multitask Learning	
	unlabeled	Domain-adversarial training Zero-shot learning	

Zero-shot Learning

<http://evchk.wikia.com/wiki/%E8%8D%89%E6%B3%A5%E9%A6%AC>

- Source data: (x^s, y^s) → Training data
- Target data: (x^t) → Testing data

Different tasks

$x^s:$



.....

$x^t :$



$y^s:$

cat

dog

.....

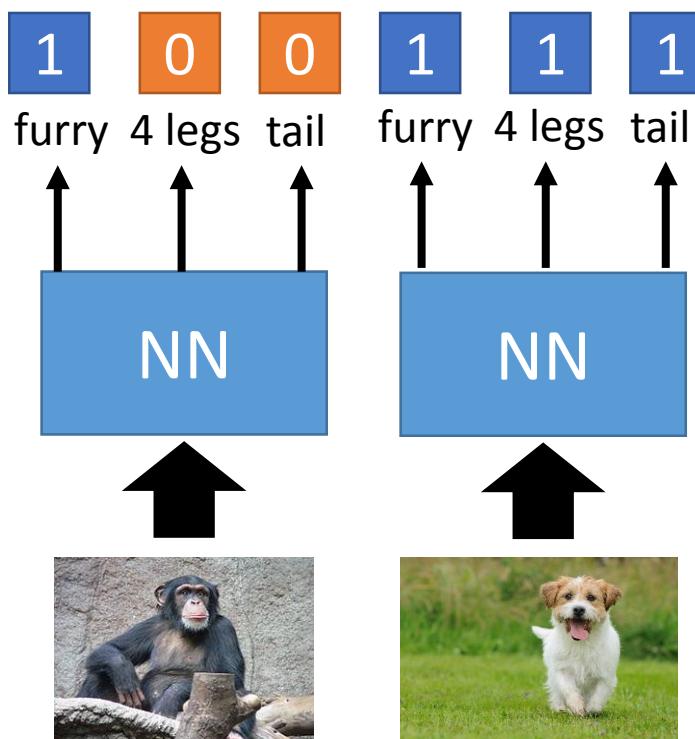
In speech recognition, we can not have all possible words in the source (training) data.

How we solve this problem in speech recognition?

Zero-shot Learning

- Representing each class by its attributes

Training



Database

attributes

	furry	4 legs	tail	...
Dog	0	0	0	
Fish	X	X	0	
Chimp	0	X	X	
...				

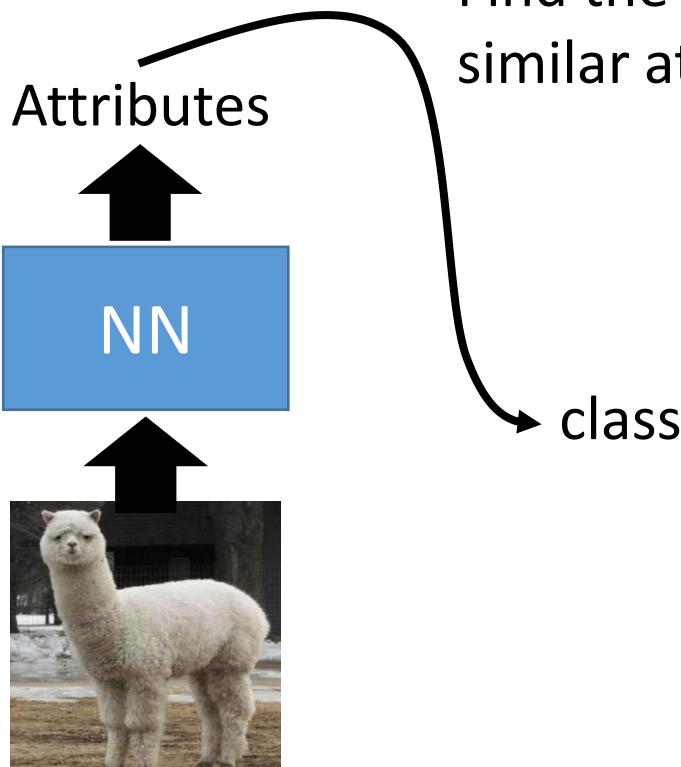
class

sufficient attributes for one
to one mapping

Zero-shot Learning

- Representing each class by its attributes

Testing



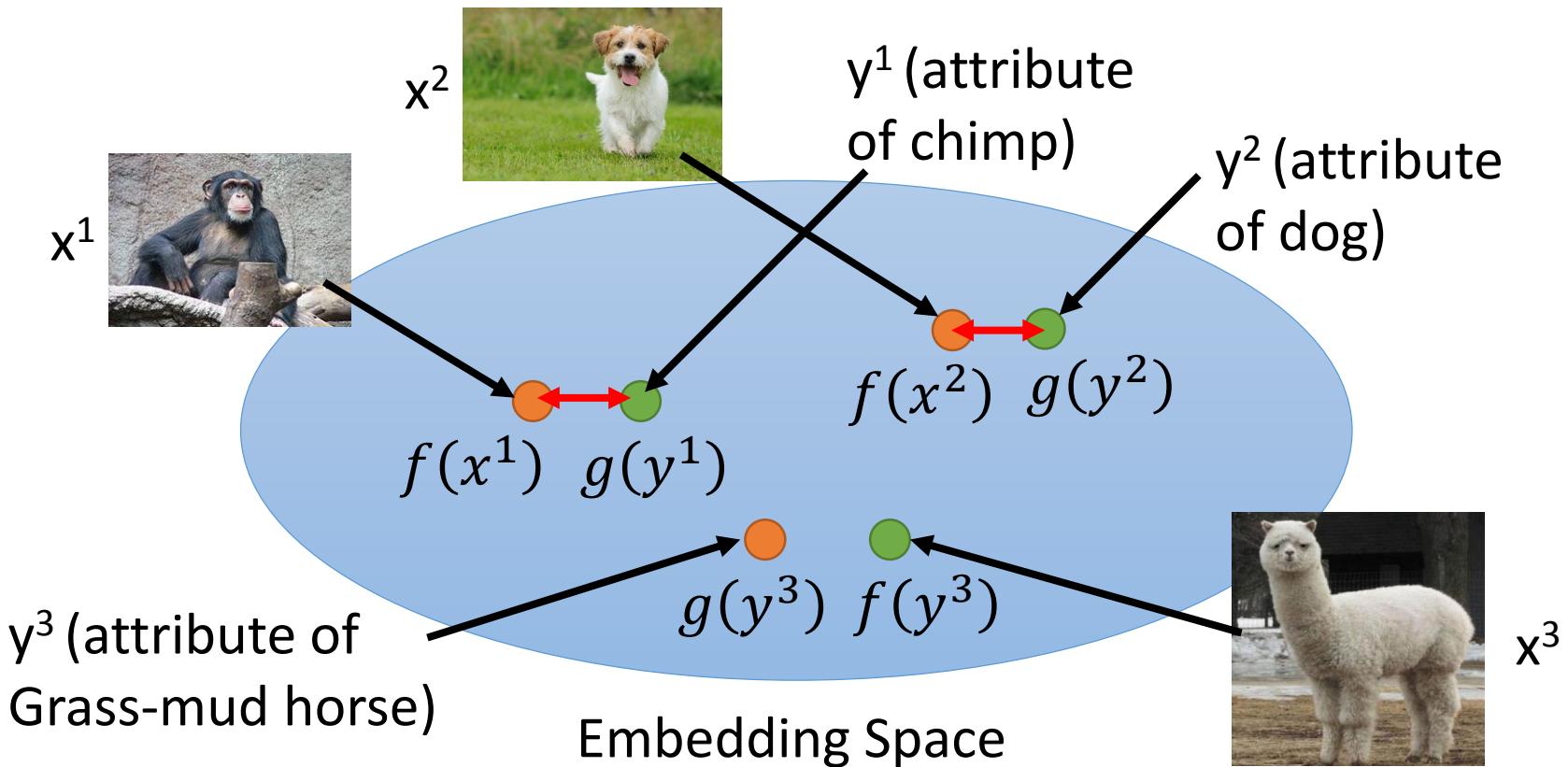
Find the class with the most similar attributes

	furry	4 legs	tail	...
Dog	o	o	o	
Fish	x	x	o	
Chimp	o	x	x	
...				

sufficient attributes for one to one mapping

Zero-shot Learning

- Attribute embedding



$f(*)$ and $g(*)$ can be NN.

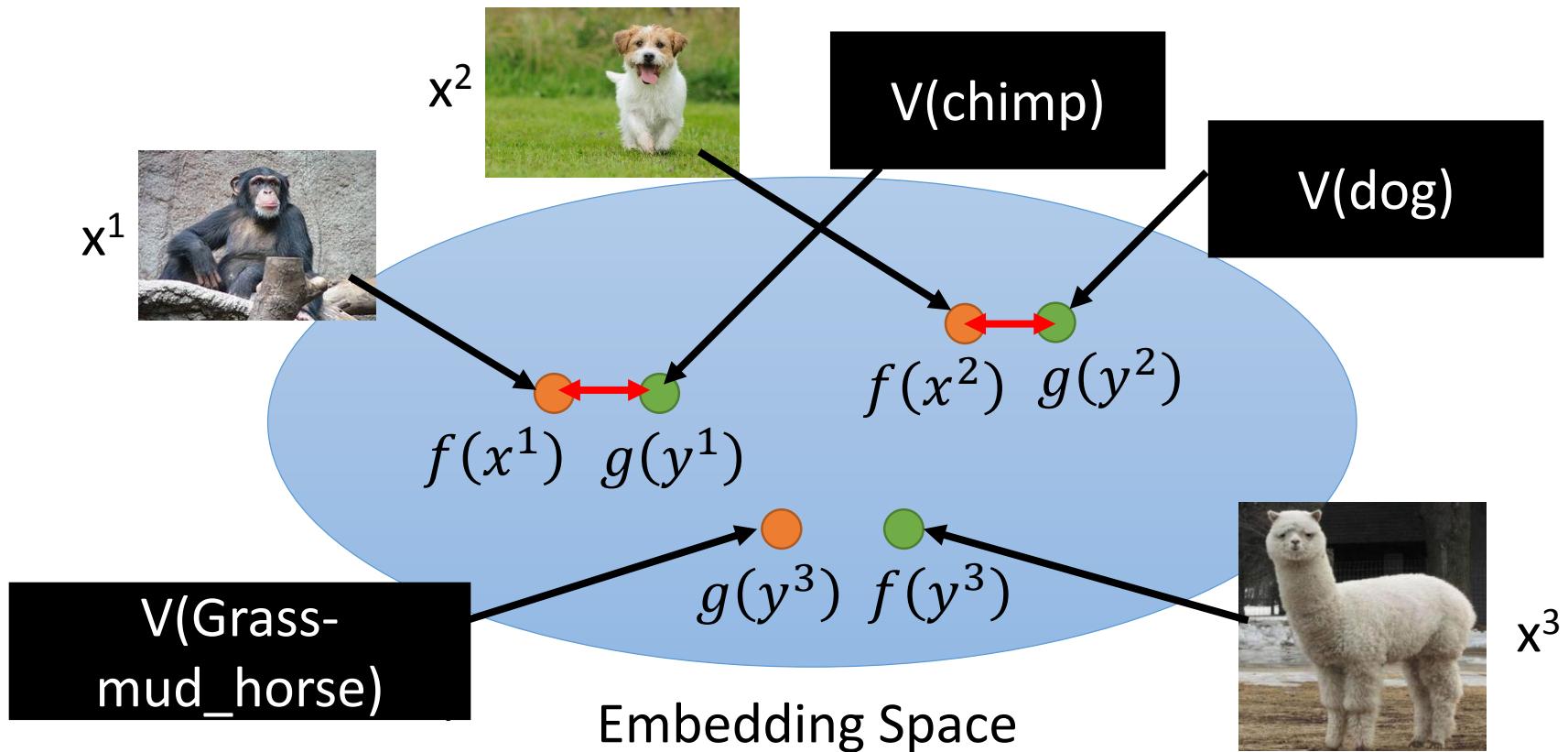
Training target:

$f(x^n)$ and $g(y^n)$ as close as possible

Zero-shot Learning

What if we don't have database

- Attribute embedding + word embedding



Zero-shot Learning

$$f^*, g^* = \arg \min_{f,g} \sum_n \|f(x^n) - g(y^n)\|_2 \quad \text{Problem?}$$

$$f^*, g^* = \arg \min_{f,g} \sum_n \max \left(0, k - f(x^n) \cdot g(y^n) + \max_{m \neq n} f(x^n) \cdot g(y^m) \right)$$

↑
Margin you defined

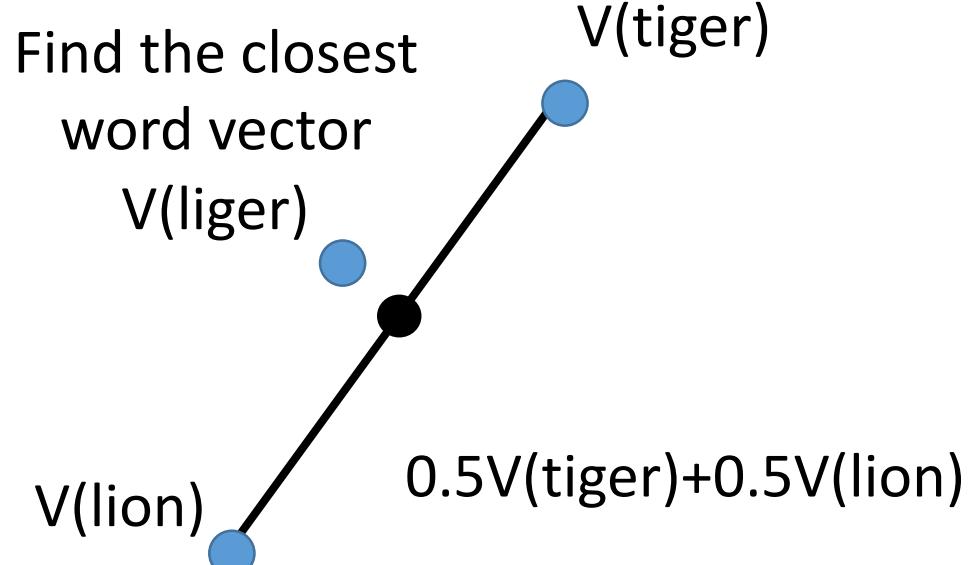
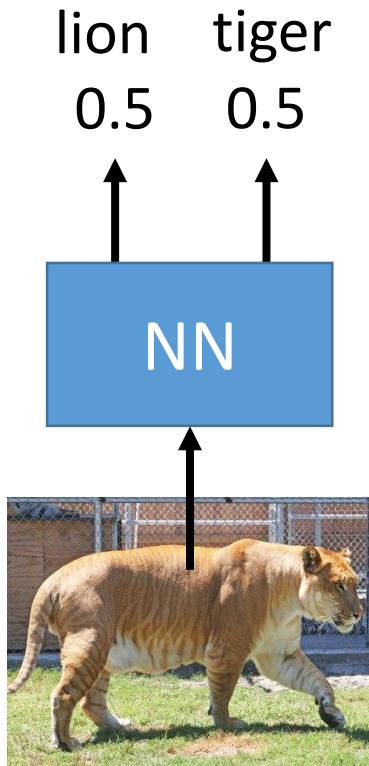
Zero loss: $k - f(x^n) \cdot g(y^n) + \max_{m \neq n} f(x^n) \cdot g(y^m) < 0$

$$\frac{f(x^n) \cdot g(y^n)}{\text{---}} - \frac{\max_{m \neq n} f(x^n) \cdot g(y^m)}{\text{---}} > k$$

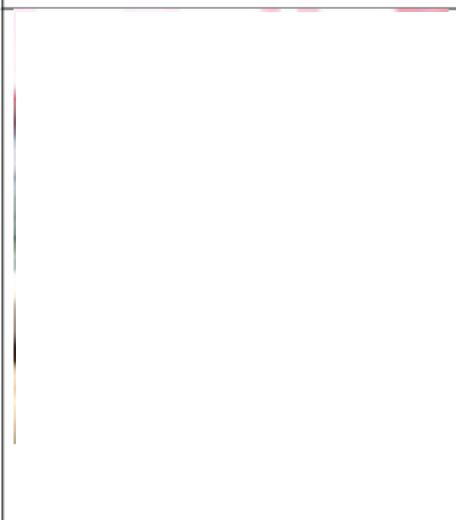
↑
 $f(x^n)$ and $g(y^n)$ as close $f(x^n)$ and $g(y^m)$ not as close

Zero-shot Learning

- Convex Combination of Semantic Embedding

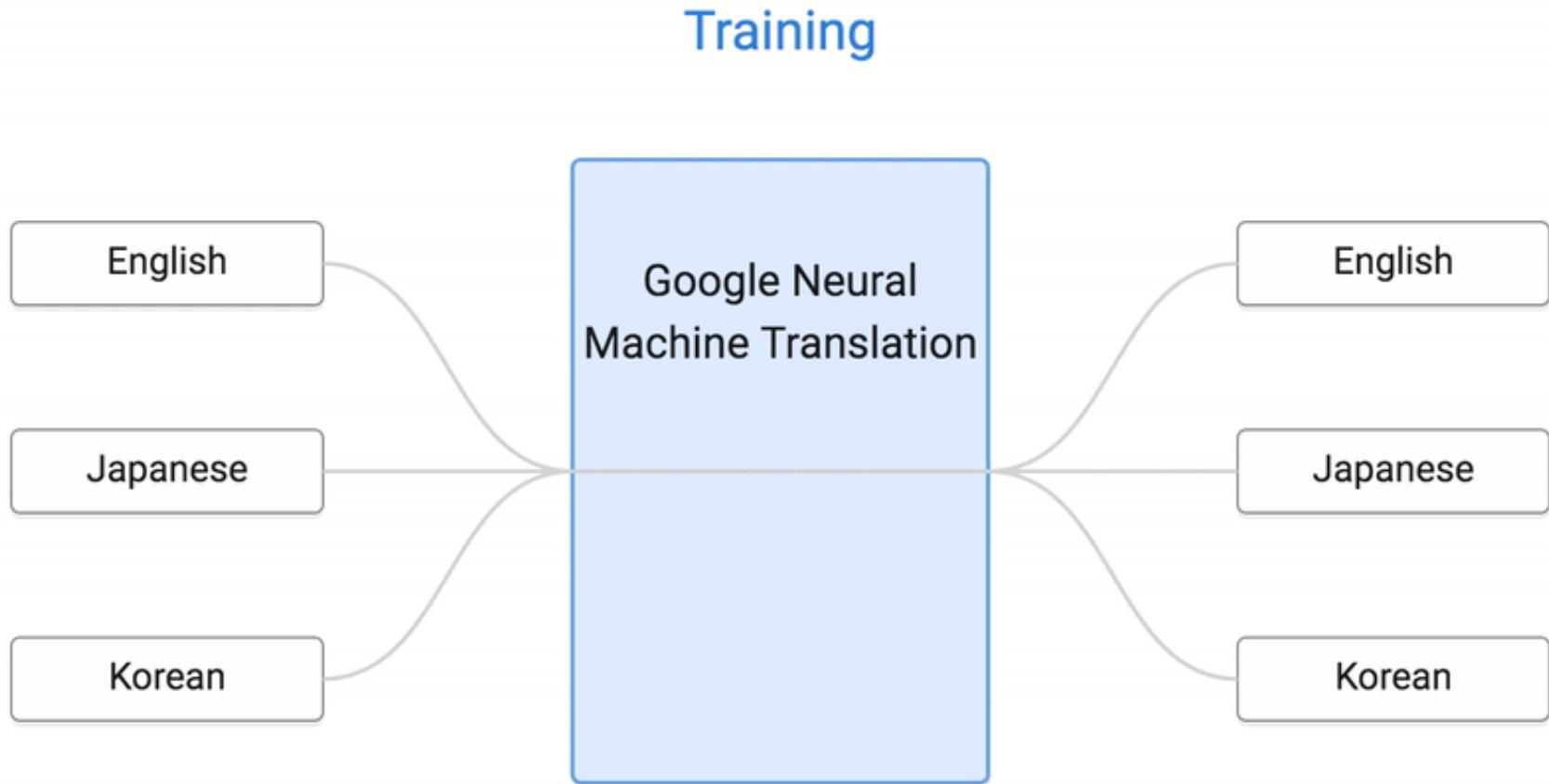


Only need off-the-shelf NN for
ImageNet and word vector

Test Image	ConvNet	DeViSE	ConSE(10)
			

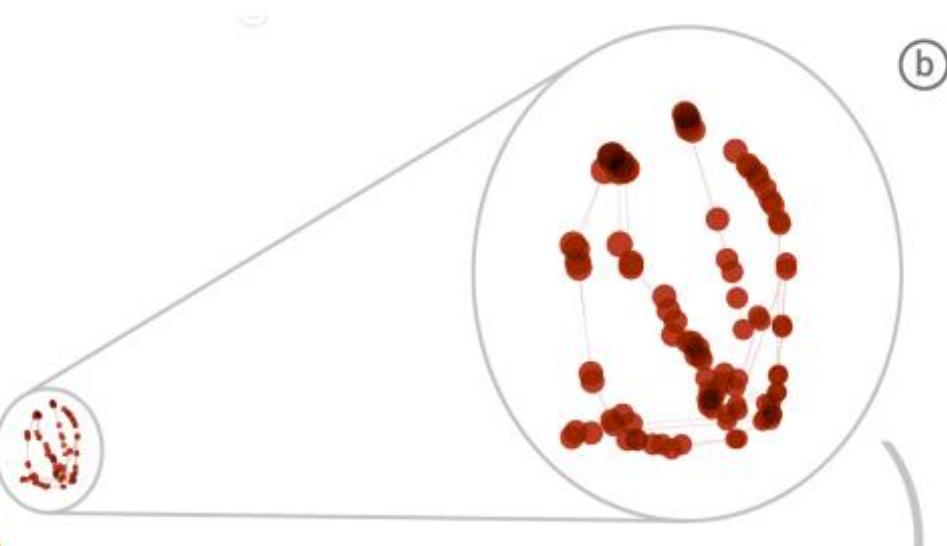
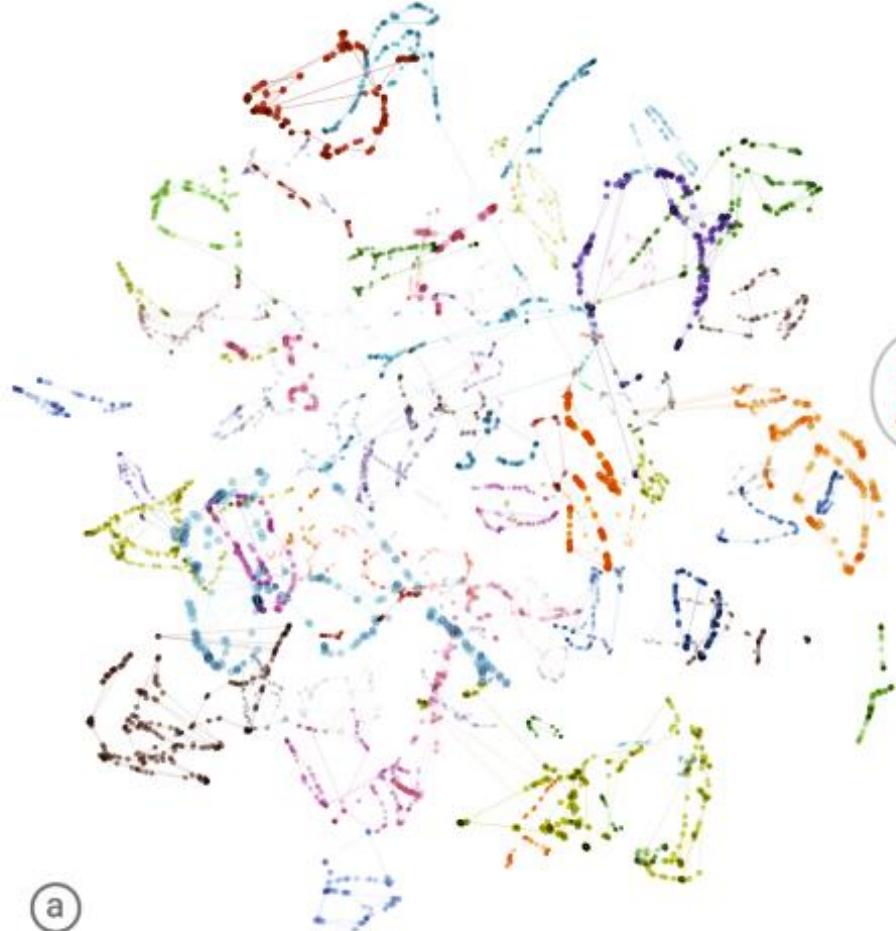
<https://arxiv.org/pdf/1312.5650v3.pdf>

Example of Zero-shot Learning



Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat. Google's Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation, arXiv preprint 2016

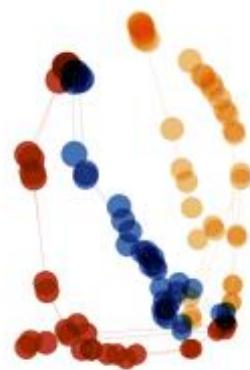
Example of Zero-shot Learning



ENGLISH
The stratosphere extends from about 10km to about 50km in altitude.

KOREAN
성층권은 고도 약 10km부터 약 50km까지 확장됩니다.

JAPANESE
成層圏は、高度 10km から 50km の範囲にあります。



More about Zero-shot learning

- Mark Palatucci, Dean Pomerleau, Geoffrey E. Hinton, Tom M. Mitchell, “Zero-shot Learning with Semantic Output Codes”, NIPS 2009
- Zeynep Akata, Florent Perronnin, Zaid Harchaoui and Cordelia Schmid, “Label-Embedding for Attribute-Based Classification”, CVPR 2013
- Andrea Frome, Greg S. Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Marc'Aurelio Ranzato, Tomas Mikolov, “DeViSE: A Deep Visual-Semantic Embedding Model”, NIPS 2013
- Mohammad Norouzi, Tomas Mikolov, Samy Bengio, Yoram Singer, Jonathon Shlens, Andrea Frome, Greg S. Corrado, Jeffrey Dean, “Zero-Shot Learning by Convex Combination of Semantic Embeddings”, arXiv preprint 2013
- Subhashini Venugopalan, Lisa Anne Hendricks, Marcus Rohrbach, Raymond Mooney, Trevor Darrell, Kate Saenko, “Captioning Images with Diverse Objects”, arXiv preprint 2016

Transfer Learning - Overview

		Source Data (not directly related to the task)	
		labelled	unlabeled
Target Data	labelled	Fine-tuning Multitask Learning	Self-taught learning Rajat Raina , Alexis Battle , Honglak Lee , Benjamin Packer , Andrew Y. Ng, Self-taught learning: transfer learning from unlabeled data, ICML, 2007
	unlabeled	Domain-adversarial training Zero-shot learning	Self-taught Clustering Wenyuan Dai, Qiang Yang, Gui-Rong Xue, Yong Yu, "Self-taught clustering", ICML 2008

Self-taught learning

- Learning to extract better representation from the source data (unsupervised approach)
- Extracting better representation for target data

Domain	Unlabeled data	Labeled data	Classes	Raw features
Image classification	10 images of outdoor scenes	Caltech101 image classification dataset	101	Intensities in 14x14 pixel patch
Handwritten character recognition	Handwritten digits ("0"–"9")	Handwritten English characters ("a"–"z")	26	Intensities in 28x28 pixel character/digit image
Font character recognition	Handwritten English characters ("a"–"z")	Font characters ("a"/"A" – "z"/"Z")	26	Intensities in 28x28 pixel character image
Song genre classification	Song snippets from 10 genres	Song snippets from 7 different genres	7	Log-frequency spectrogram over 50ms time windows
Webpage classification	100,000 news articles (Reuters newswire)	Categorized webpages (from DMOZ hierarchy)	2	Bag-of-words with 500 word vocabulary
UseNet article classification	100,000 news articles (Reuters newswire)	Categorized UseNet posts (from "SRAA" dataset)	2	Bag-of-words with 377 word vocabulary

Acknowledgement

- 感謝 劉致廷 同學於上課時發現投影片上的錯誤

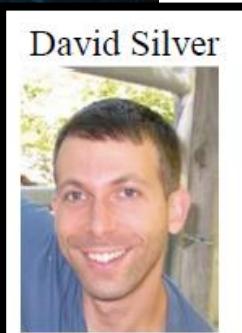
Deep Reinforcement Learning

Scratching the surface

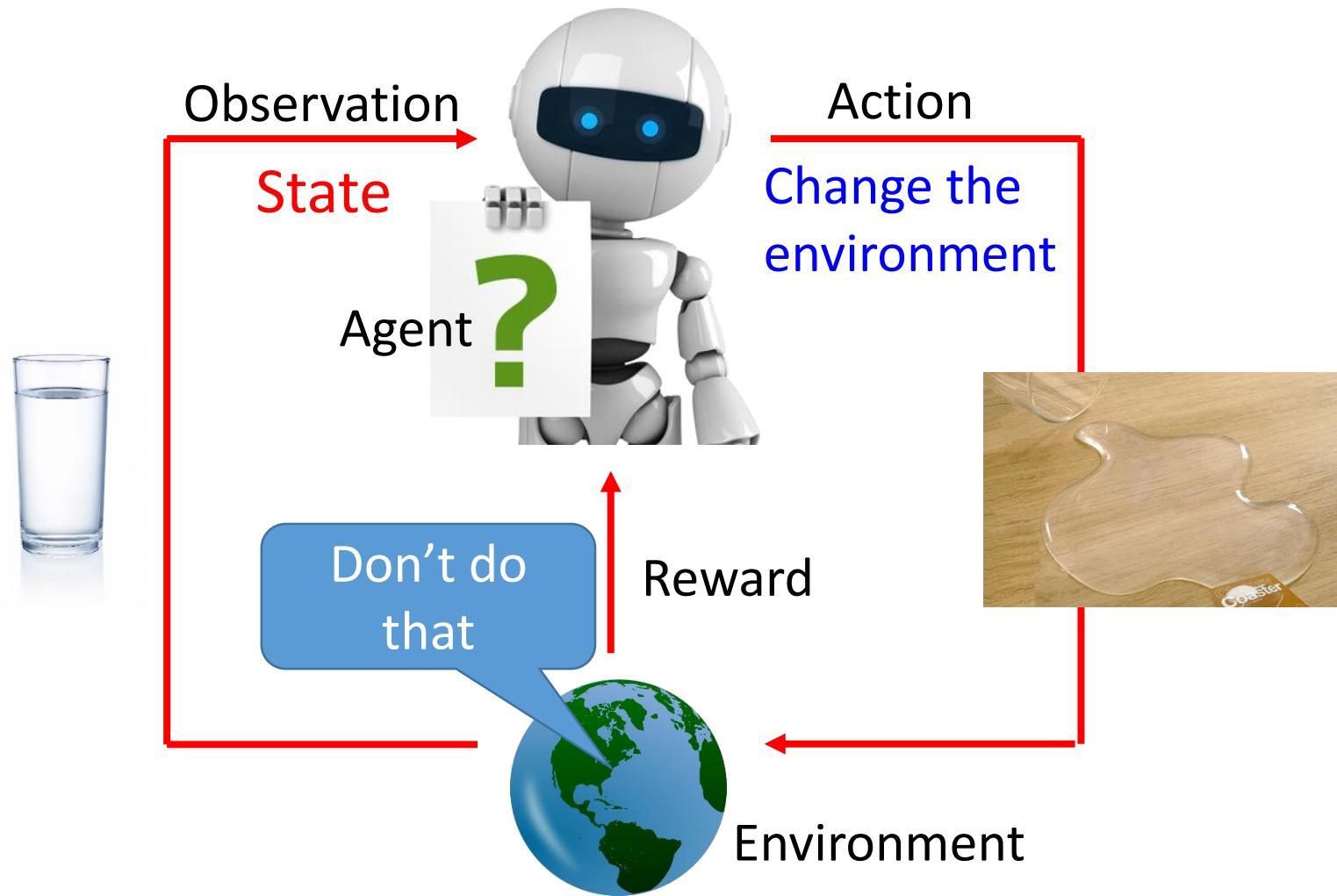
Deep Reinforcement Learning



Deep Reinforcement Learning: $\text{AI} = \text{RL} + \text{DL}$



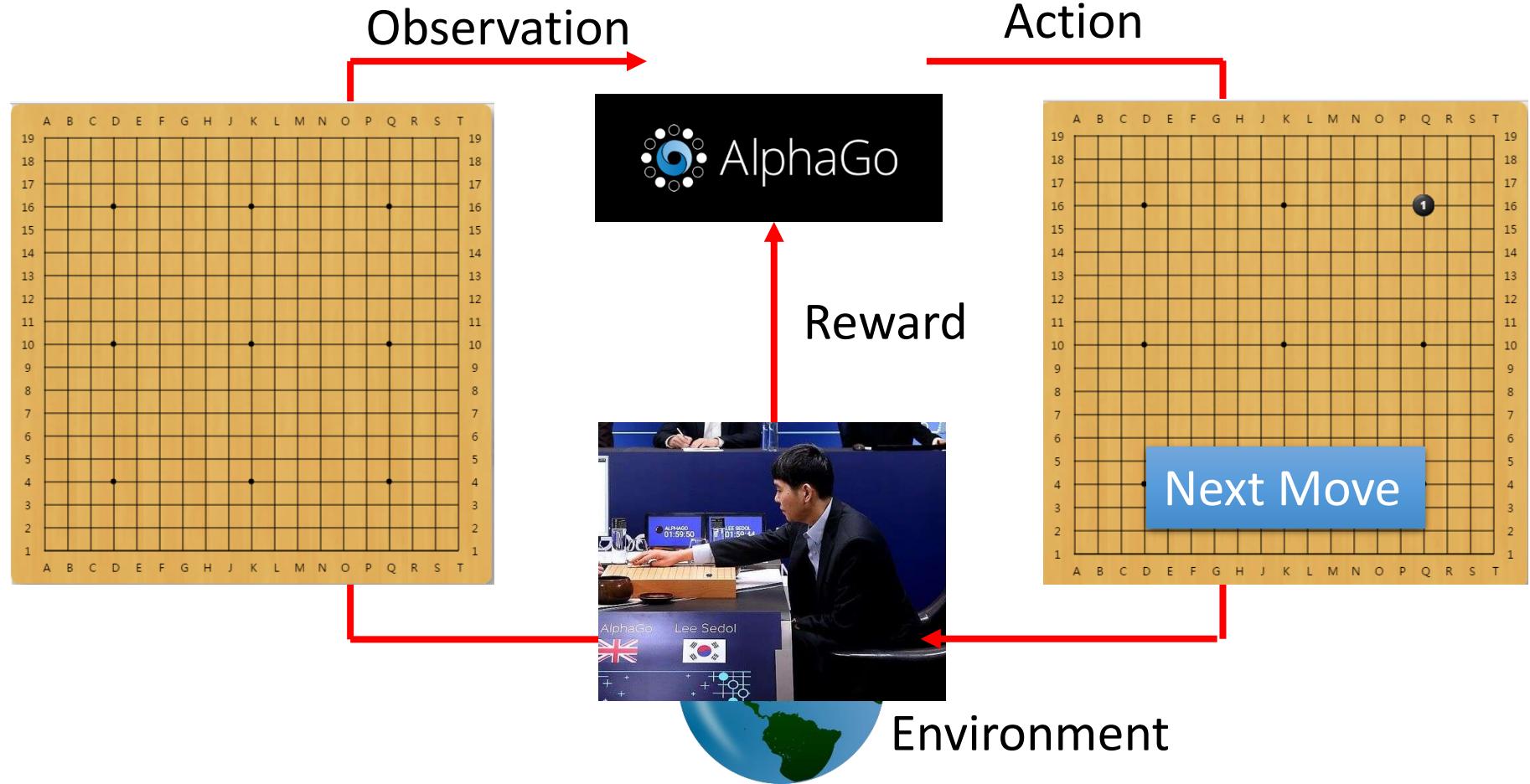
Scenario of Reinforcement Learning



Scenario of Reinforcement Learning

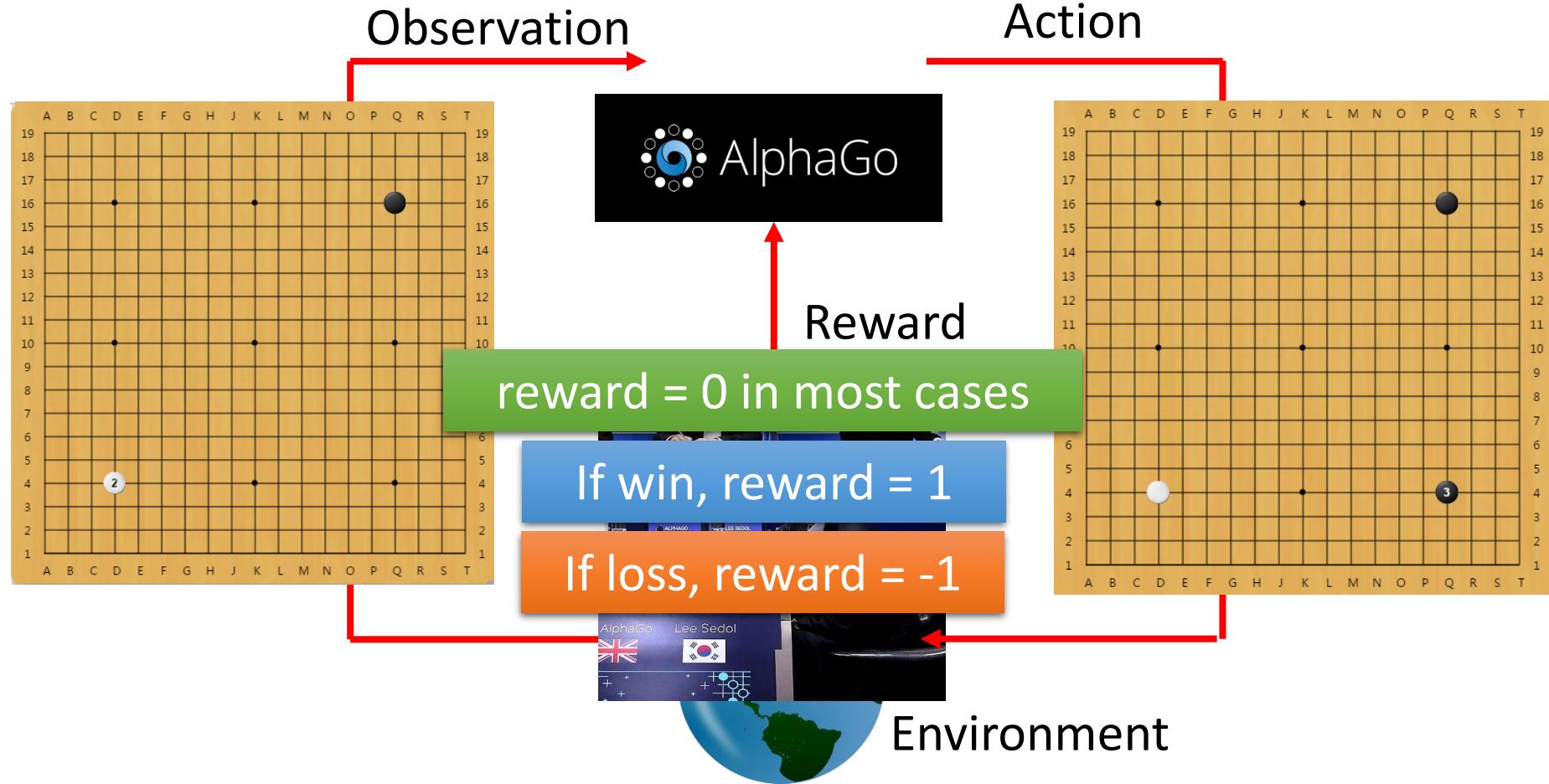


Learning to play Go



Learning to play Go

Agent learns to take actions to maximize expected reward.



Learning to play Go

- Supervised v.s. Reinforcement

- Supervised: Learning from teacher



Next move:
“5-5”



Next move:
“3-3”

- Reinforcement Learning

Learning from experience

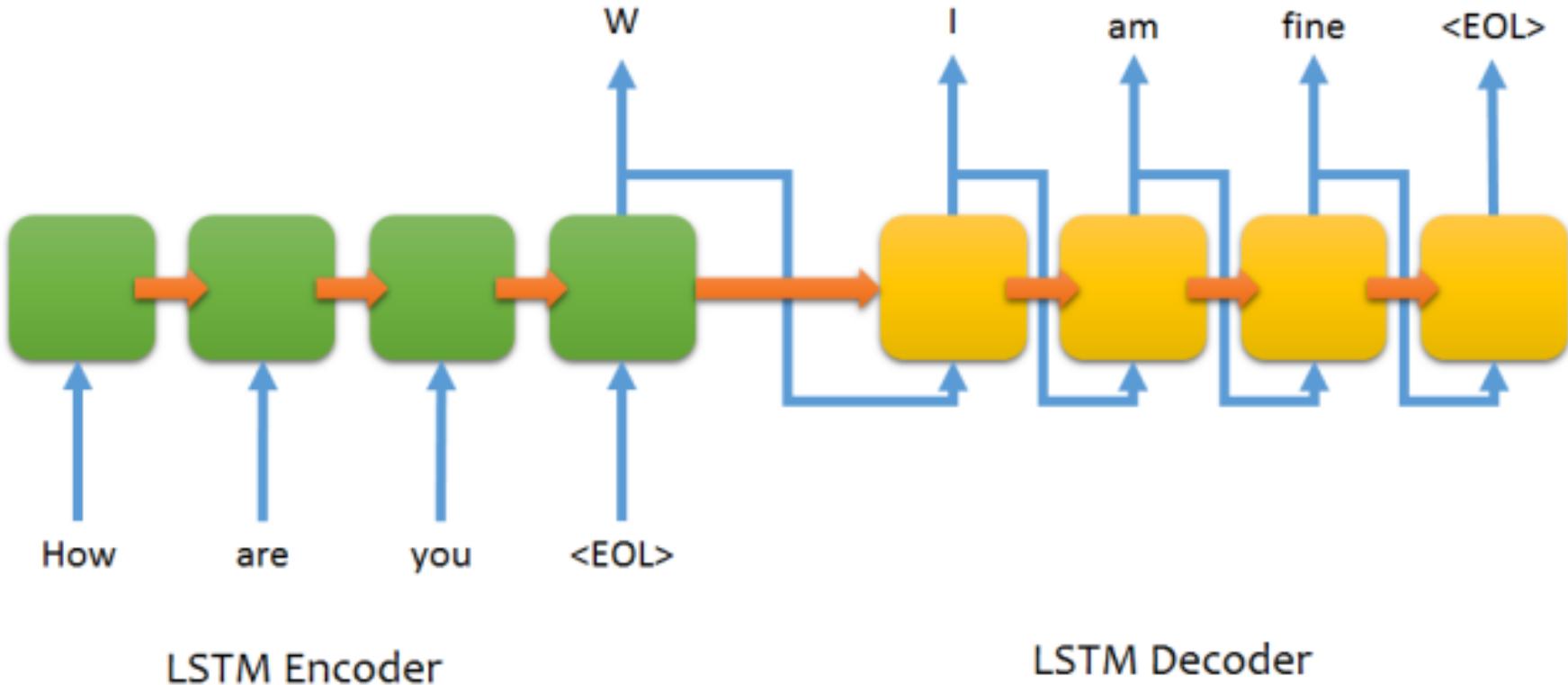
First move → many moves → Win!

(Two agents play with each other.)

Alpha Go is supervised learning + reinforcement learning.

Learning a chat-bot

- Sequence-to-sequence learning



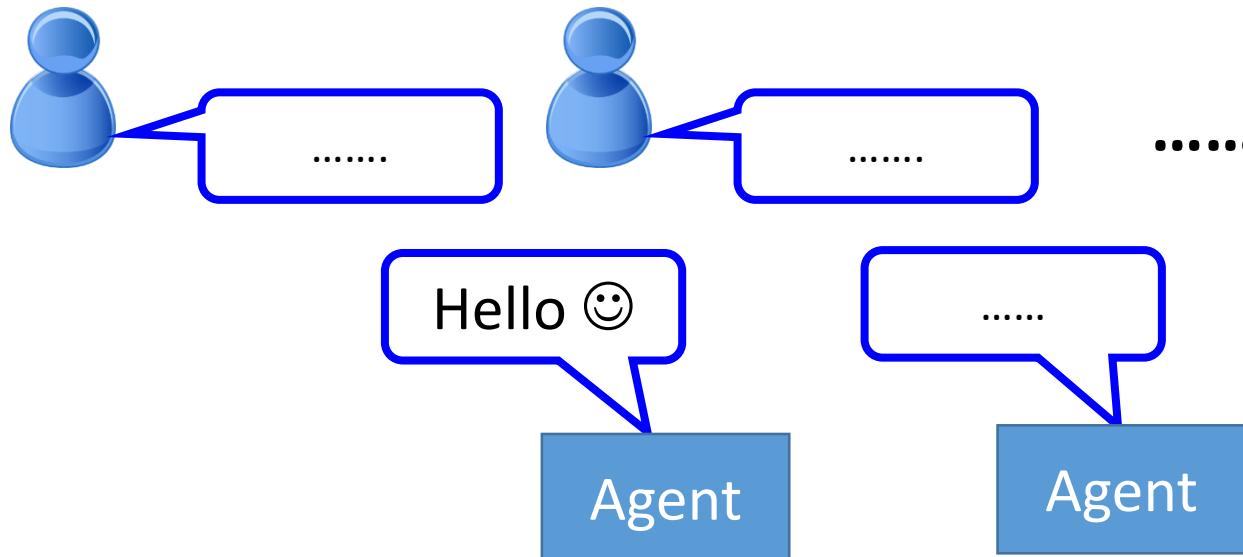
Learning a chat-bot

- Supervised v.s. Reinforcement

- Supervised



- Reinforcement



Bad

Learning a chat-bot

- Reinforcement Learning

- Let two agents talk to each other (sometimes generate good dialogue, sometimes bad)



How old are you?



How old are you?

See you.



I am 16.



See you.



See you.



I thought you were 12.



What make you
think so?

Learning a chat-bot

- Reinforcement Learning

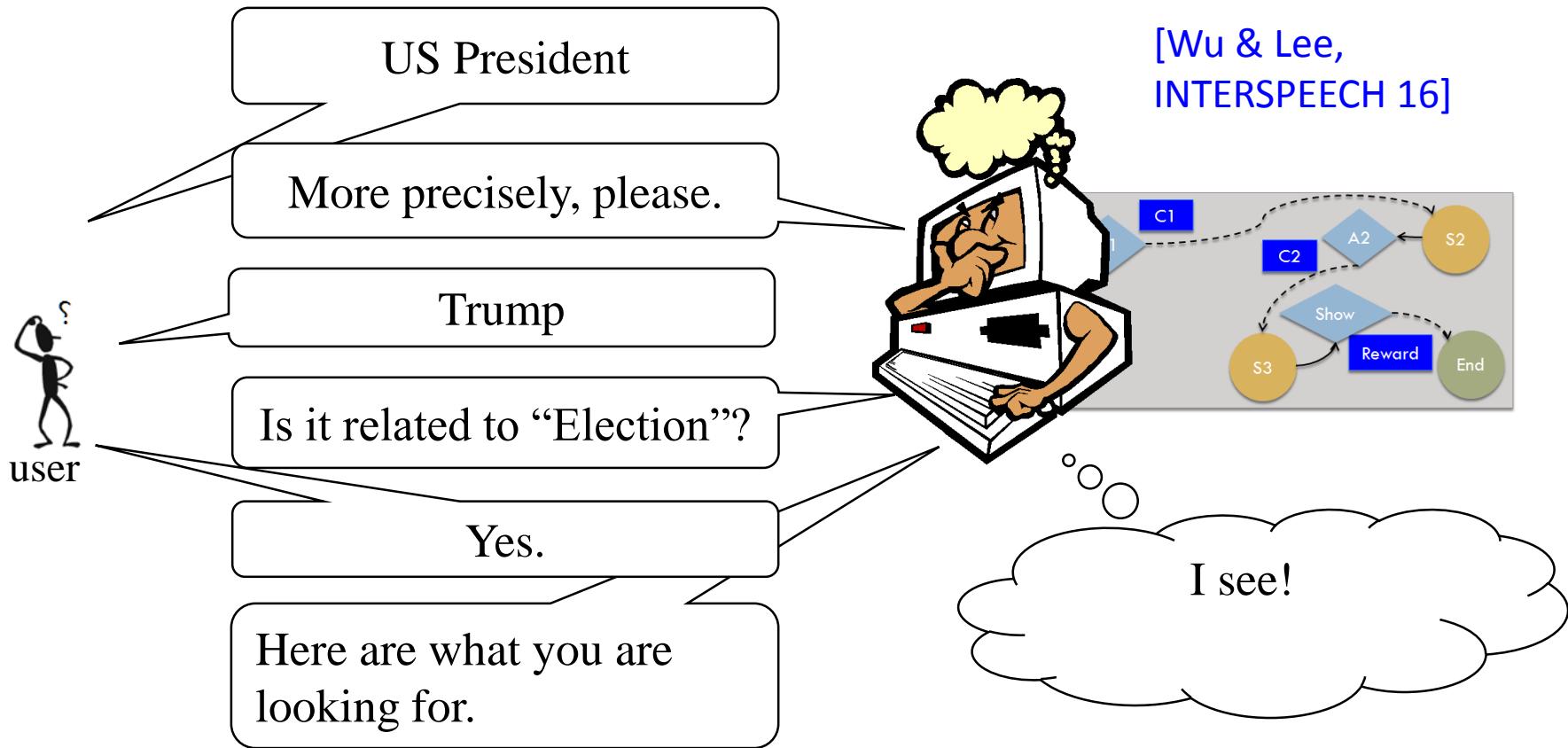
- By this approach, we can generate a lot of dialogues.
- Use some pre-defined rules to evaluate the goodness of a dialogue

Machine learns from the evaluation



More applications

- Interactive retrieval



More applications

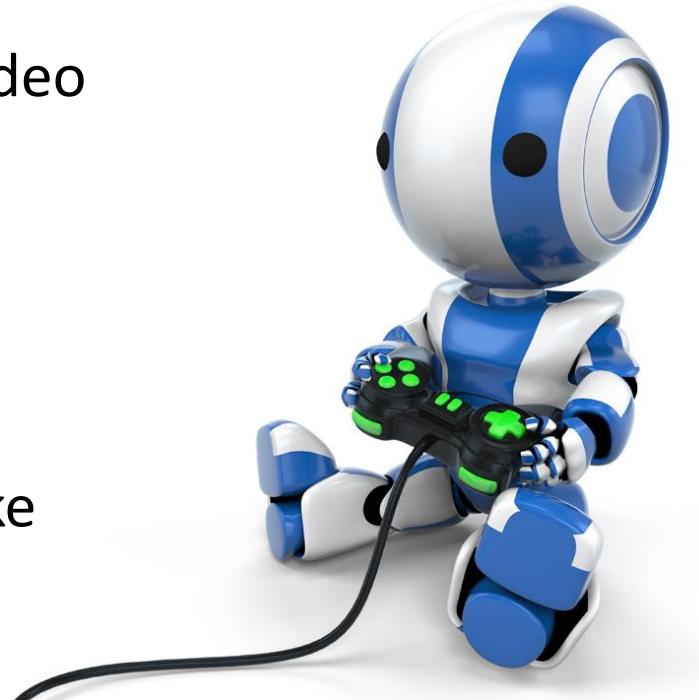
- Flying Helicopter
 - <https://www.youtube.com/watch?v=0JL04JJjocc>
- Driving
 - <https://www.youtube.com/watch?v=0xo1Ldx3L5Q>
- Google Cuts Its Giant Electricity Bill With DeepMind-Powered AI
 - <http://www.bloomberg.com/news/articles/2016-07-19/google-cuts-its-giant-electricity-bill-with-deepmind-powered-ai>
- Text generation
 - Hongyu Guo, “Generating Text with Deep Reinforcement Learning”, NIPS, 2015
 - Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, Wojciech Zaremba, “Sequence Level Training with Recurrent Neural Networks”, ICLR, 2016

Example: Playing Video Game

- Widely studies:
 - Gym: <https://gym.openai.com/>
 - Universe: <https://openai.com/blog/universe/>

Machine learns to play video games as human players

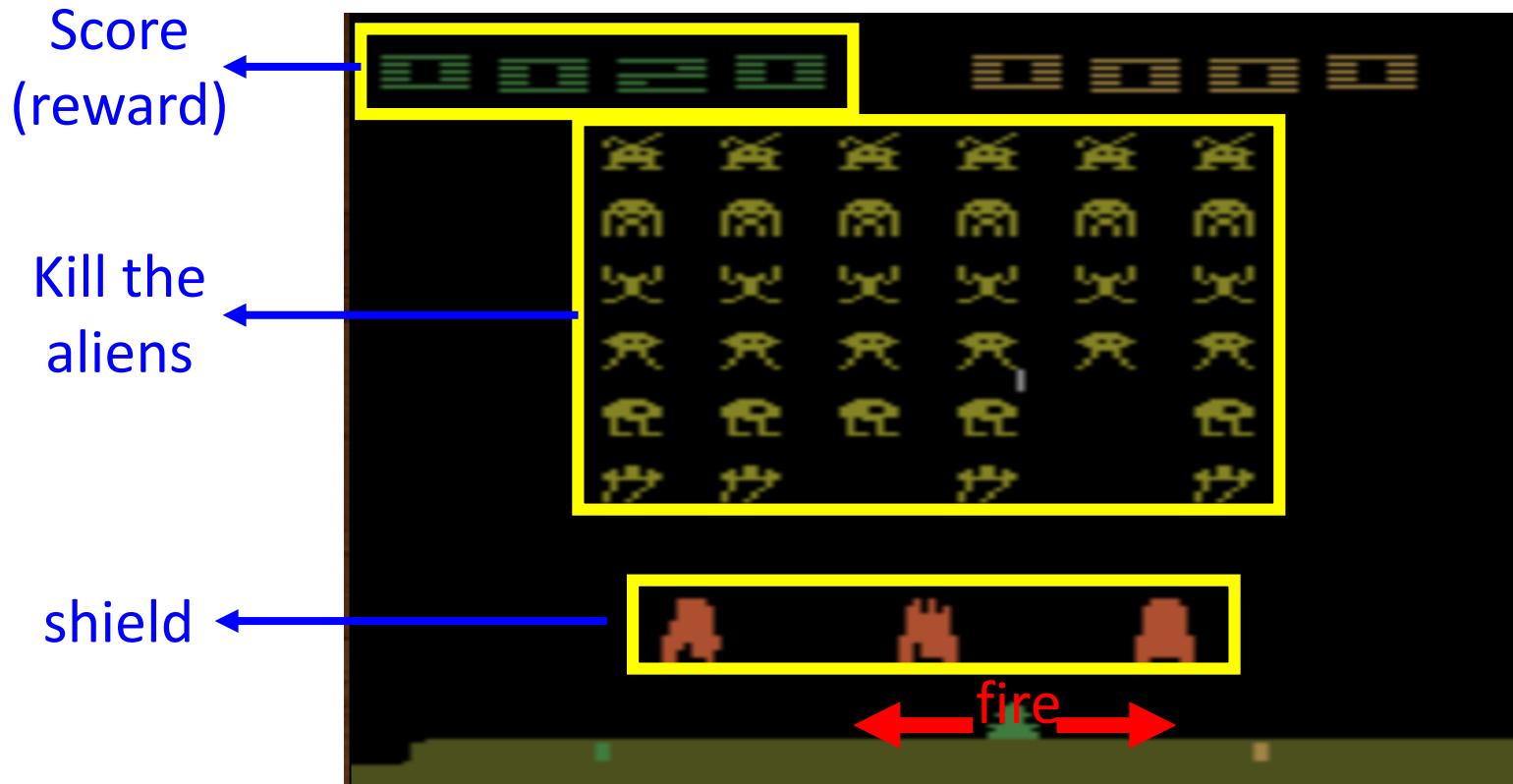
- What machine observes is pixels
- Machine learns to take proper action itself



Example: Playing Video Game

- Space invader

Termination: all the aliens are killed, or your spaceship is destroyed.



Example: Playing Video Game

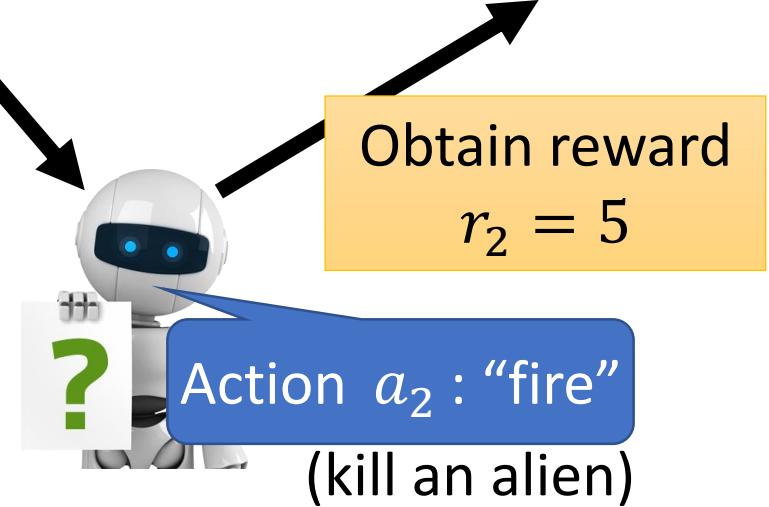
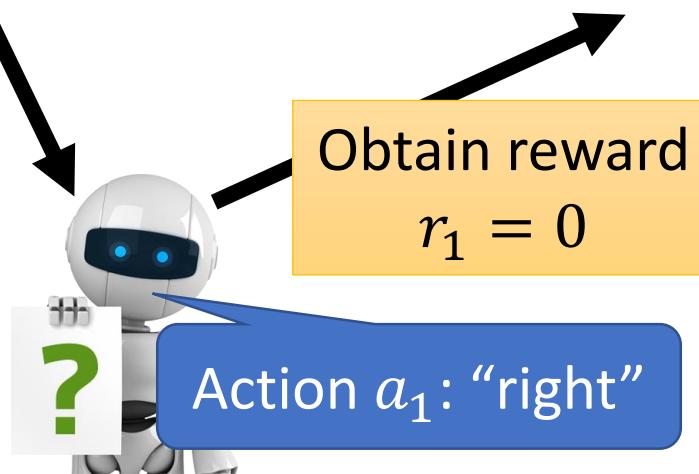
- Space invader
 - Play yourself:
<http://www.2600online.com/spaceinvaders.html>
 - How about machine:
https://gym.openai.com/evaluations/eval_Eduo zx4HRyqgTCVk9ltw

Example: Playing Video Game

Start with
observation s_1

Observation s_2

Observation s_3



Usually there is some randomness in the environment

Example: Playing Video Game

Start with
observation s_1



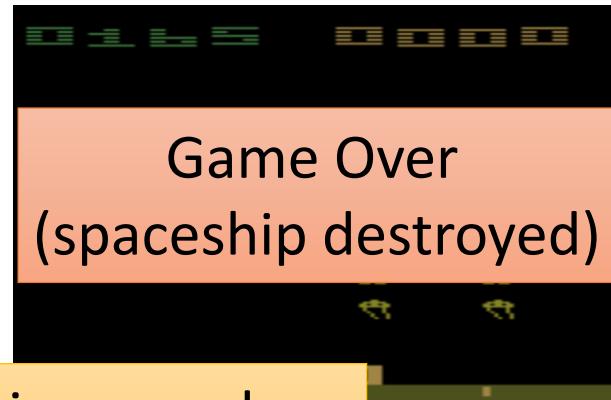
Observation s_2



Observation s_3



After many turns



Obtain reward r_T

Action a_T

This is an *episode*.

Learn to maximize the
expected cumulative
reward per episode

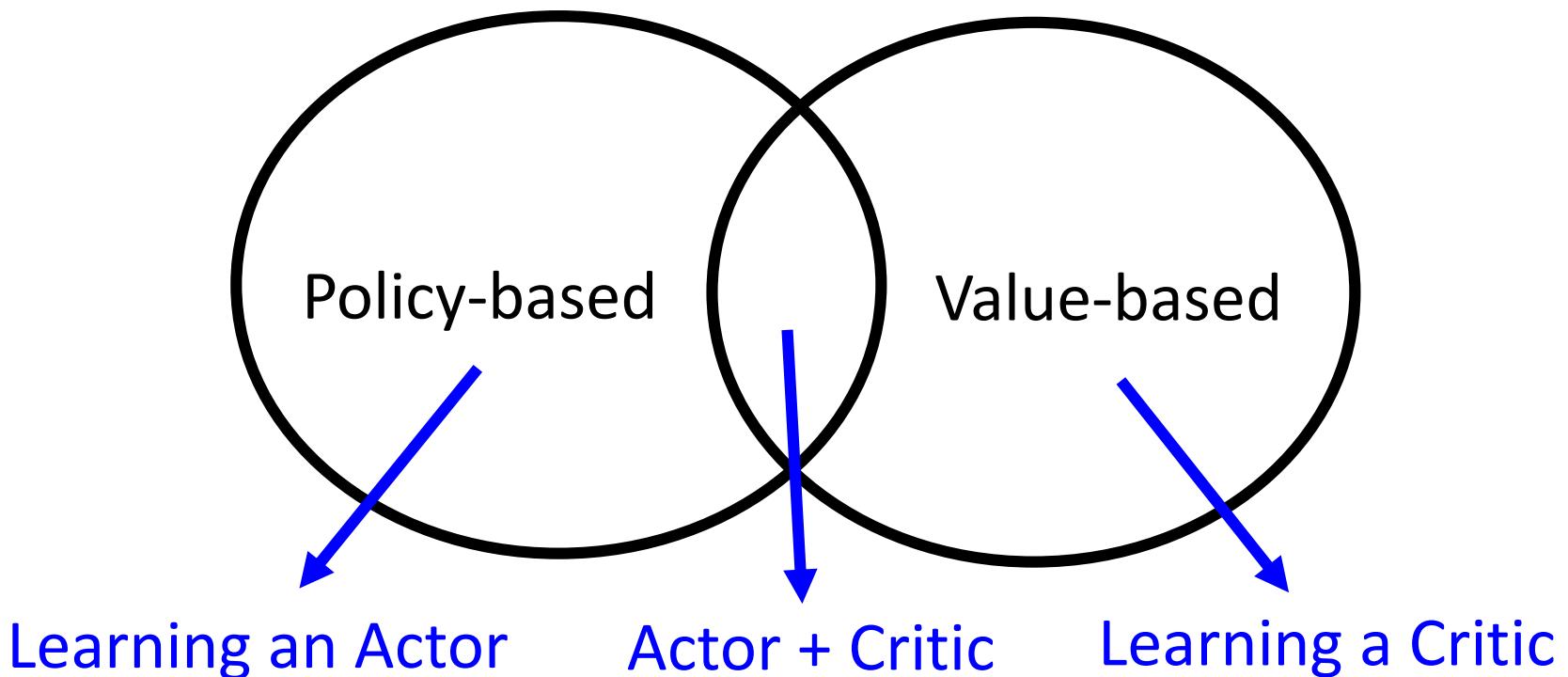
Difficulties of Reinforcement Learning

- Reward delay
 - In space invader, only “fire” obtains reward
 - Although the moving before “fire” is important
 - In Go playing, it may be better to sacrifice immediate reward to gain more long-term reward
- Agent’s actions affect the subsequent data it receives
 - E.g. Exploration



Outline

Alpha Go: policy-based + value-based
+ model-based



Asynchronous Advantage Actor-Critic (A3C)

Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, Koray Kavukcuoglu, "Asynchronous Methods for Deep Reinforcement Learning", ICML, 2016

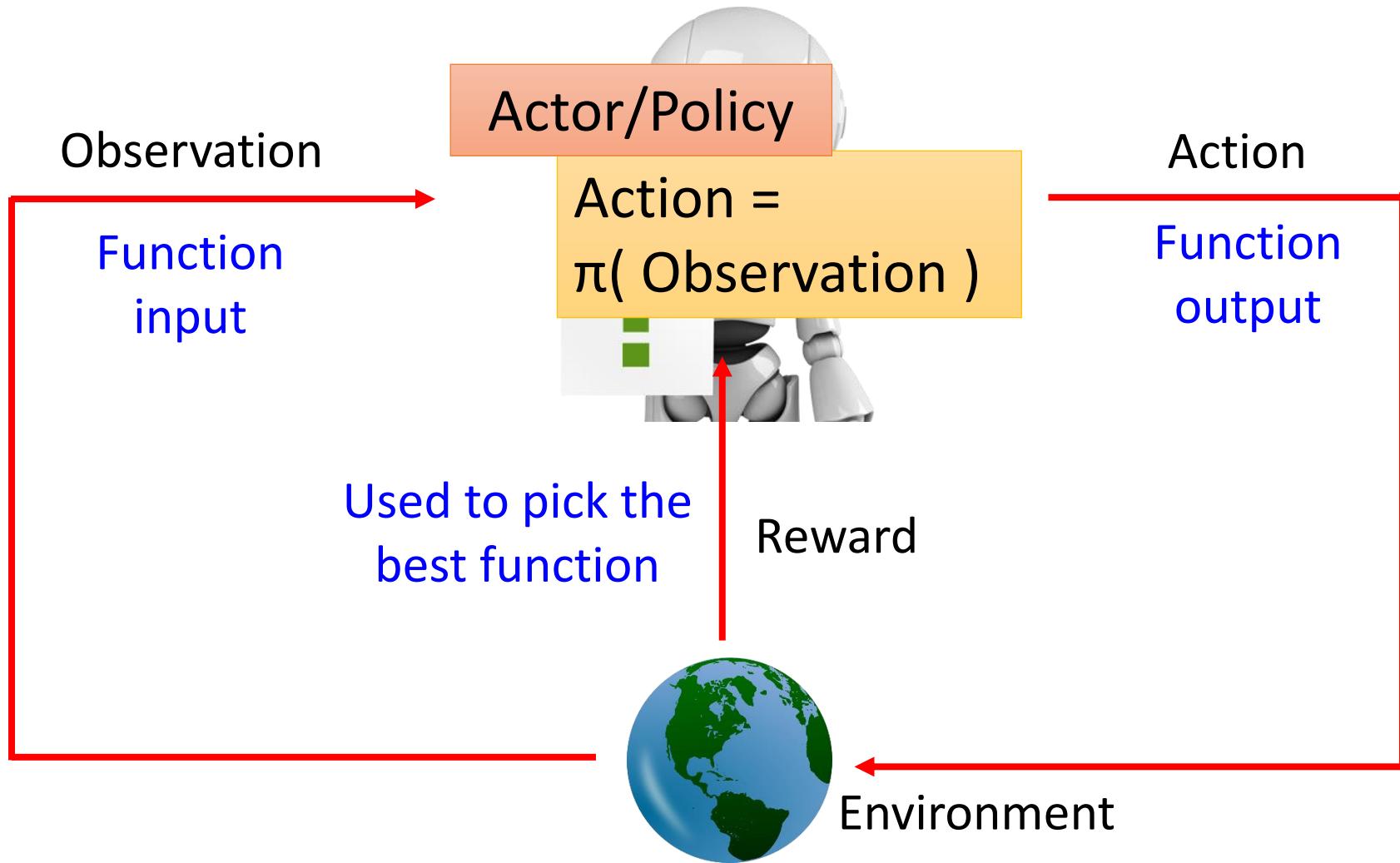
To learn deep reinforcement learning

- Textbook: Reinforcement Learning: An Introduction
 - <https://webdocs.cs.ualberta.ca/~sutton/book/the-book.html>
- Lectures of David Silver
 - <http://www0.cs.ucl.ac.uk/staff/D.Silver/web/Teaching.html> (10 lectures, 1:30 each)
 - http://videolectures.net/rldm2015_silver_reinforcement_learning/ (Deep Reinforcement Learning)
- Lectures of John Schulman
 - https://youtu.be/aUrX-rP_ss4

Policy-based Approach

Learning an Actor

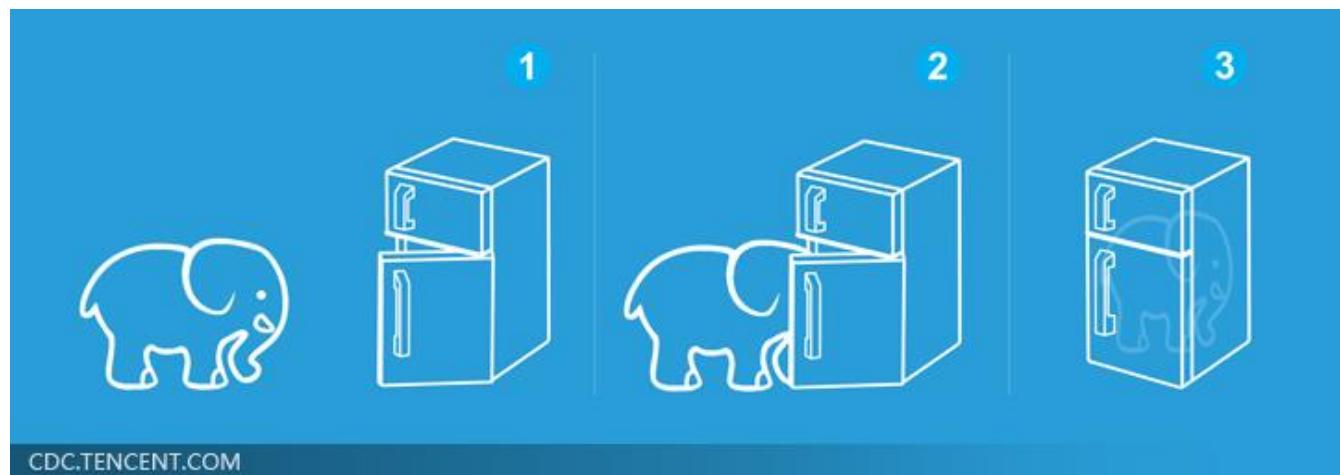
Machine Learning ≈ Looking for a Function



Three Steps for Deep Learning

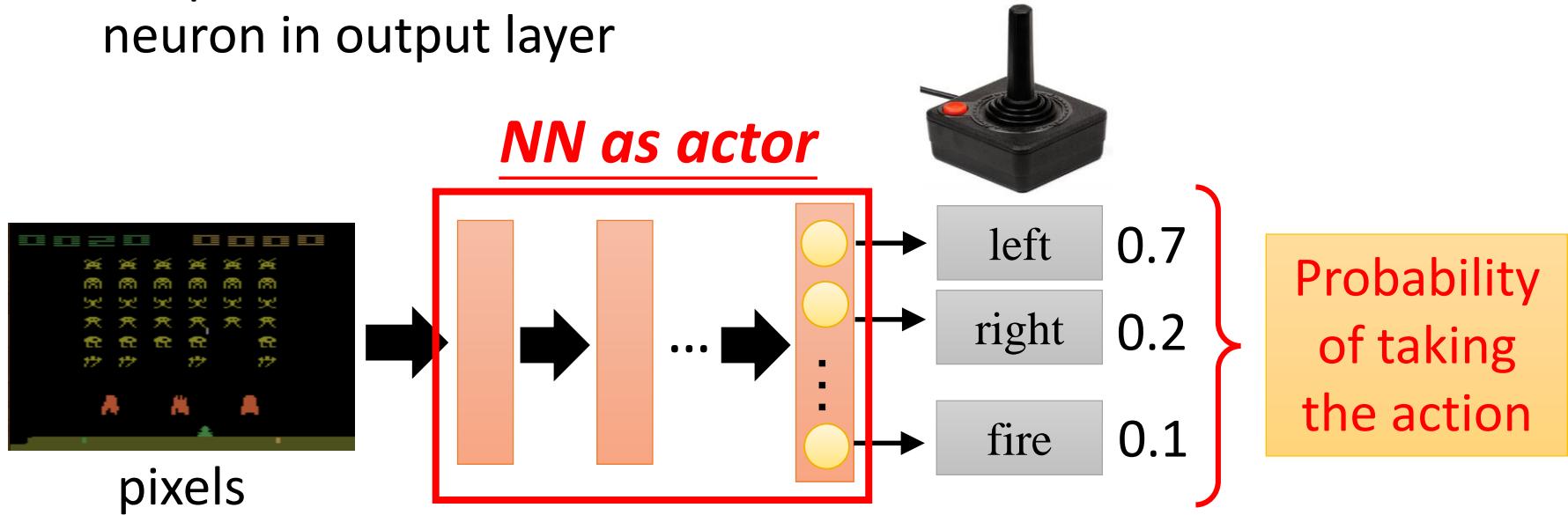


Deep Learning is so simple



Neural network as Actor

- Input of neural network: the observation of machine represented as a vector or a matrix
- Output neural network : each action corresponds to a neuron in output layer



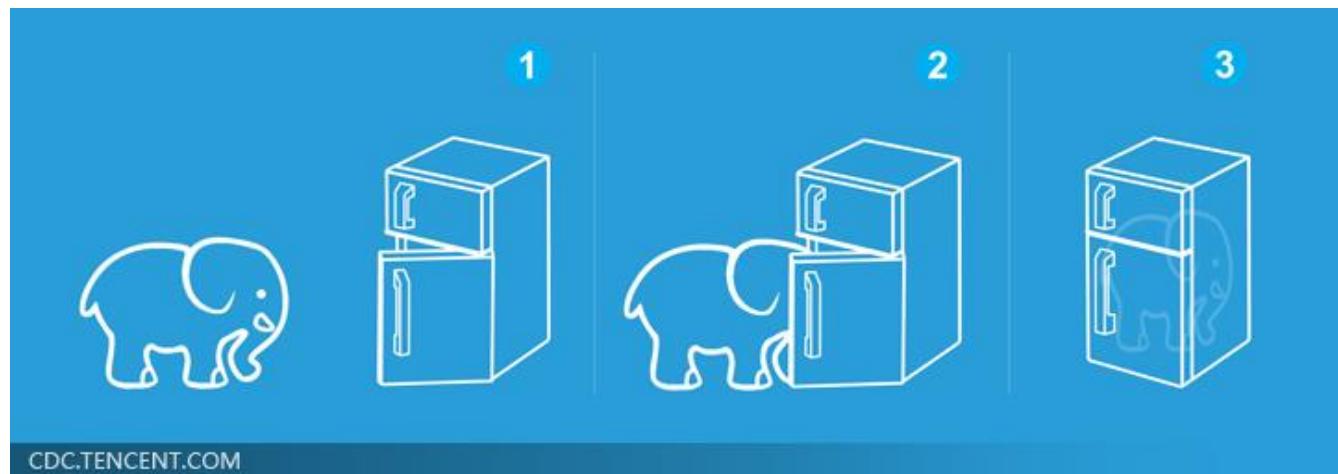
What is the benefit of using network instead of lookup table?

generalization

Three Steps for Deep Learning



Deep Learning is so simple



Goodness of Actor

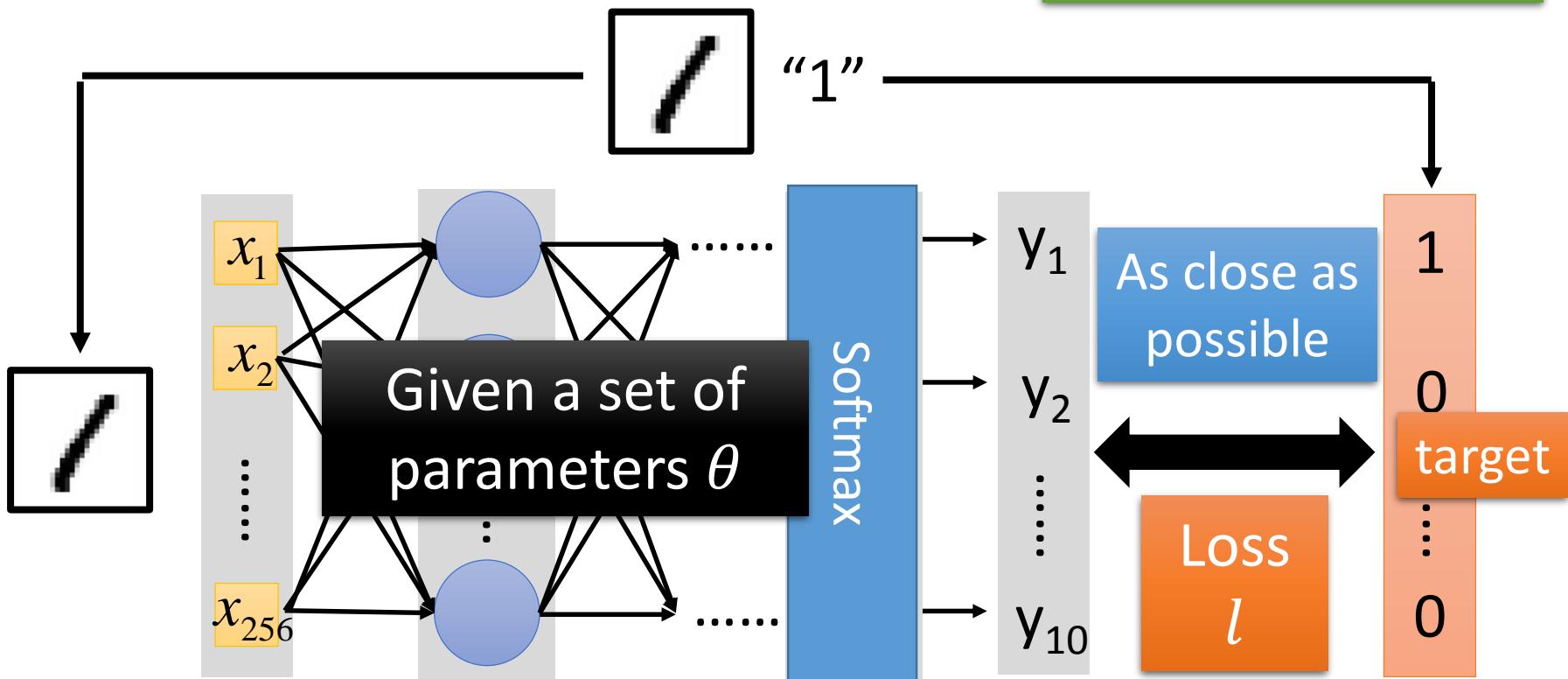
Total Loss:

$$L = \sum_{n=1}^N l_n$$

- Review: Supervised learning

Training Example

Find the network parameters θ^* that minimize total loss L



Goodness of Actor

- Given an actor $\pi_\theta(s)$ with network parameter θ
- Use the actor $\pi_\theta(s)$ to play the video game
 - Start with observation s_1
 - Machine decides to take a_1
 - Machine obtains reward r_1
 - Machine sees observation s_2
 - Machine decides to take a_2
 - Machine obtains reward r_2
 - Machine sees observation s_3
 -
 - Machine decides to take a_T
 - Machine obtains reward r_T

END

Total reward: $R_\theta = \sum_{t=1}^T r_t$

Even with the same actor,
 R_θ is different each time

Randomness in the actor
and the game

We define \bar{R}_θ as the
expected value of R_θ

\bar{R}_θ evaluates the goodness of an actor $\pi_\theta(s)$

Goodness of Actor

- An episode is considered as a trajectory τ
 - $\tau = \{s_1, a_1, r_1, s_2, a_2, r_2, \dots, s_T, a_T, r_T\}$
 - $R(\tau) = \sum_{t=1}^T r_t$
 - If you use an actor to play the game, each τ has a probability to be sampled
 - The probability depends on actor parameter θ :
 $P(\tau|\theta)$

$$\bar{R}_\theta = \sum_{\tau} R(\tau) P(\tau|\theta) \approx \frac{1}{N} \sum_{n=1}^N R(\tau^n)$$

Sum over all possible trajectory

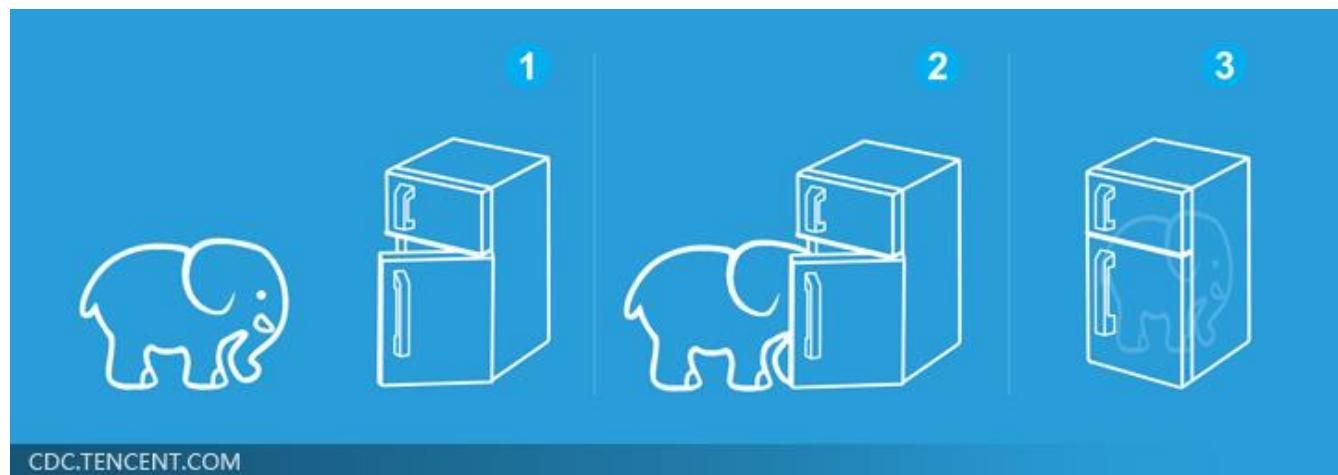
Use π_θ to play the game N times, obtain $\{\tau^1, \tau^2, \dots, \tau^N\}$

Sampling τ from $P(\tau|\theta)$ N times

Three Steps for Deep Learning



Deep Learning is so simple



Gradient Ascent

- Problem statement

$$\theta^* = \arg \max_{\theta} \bar{R}_{\theta}$$

$$\bar{R}_{\theta} = \sum_{\tau} R(\tau)P(\tau|\theta)$$

- Gradient ascent

- Start with θ^0

$$\bullet \theta^1 \leftarrow \theta^0 + \eta \nabla \bar{R}_{\theta^0}$$

$$\bullet \theta^2 \leftarrow \theta^1 + \eta \nabla \bar{R}_{\theta^1}$$

-

$$\theta = \{w_1, w_2, \dots, b_1, \dots\}$$

$$\nabla \bar{R}_{\theta} = \begin{bmatrix} \partial \bar{R}_{\theta} / \partial w_1 \\ \partial \bar{R}_{\theta} / \partial w_2 \\ \vdots \\ \partial \bar{R}_{\theta} / \partial b_1 \\ \vdots \end{bmatrix}$$

Gradient Ascent

$$\bar{R}_\theta = \sum_\tau R(\tau)P(\tau|\theta) \quad \nabla \bar{R}_\theta = ?$$

$$\nabla \bar{R}_\theta = \sum_\tau R(\tau)\nabla P(\tau|\theta) = \sum_\tau R(\tau)P(\tau|\theta) \frac{\nabla P(\tau|\theta)}{P(\tau|\theta)}$$

$R(\tau)$ do not have to be differentiable

It can even be a black box.

$$= \boxed{\sum_\tau} R(\tau) \boxed{P(\tau|\theta)} \nabla \log P(\tau|\theta)$$

$$\frac{d \log(f(x))}{dx} = \frac{1}{f(x)} \frac{df(x)}{dx}$$

$$\approx \frac{1}{N} \sum_{n=1}^N R(\tau^n) \underline{\nabla \log P(\tau^n|\theta)}$$

Use π_θ to play the game N times,
Obtain $\{\tau^1, \tau^2, \dots, \tau^N\}$

Gradient Ascent

$$\nabla \log P(\tau | \theta) = ?$$

- $\tau = \{s_1, a_1, r_1, s_2, a_2, r_2, \dots, s_T, a_T, r_T\}$

$$P(\tau | \theta) =$$

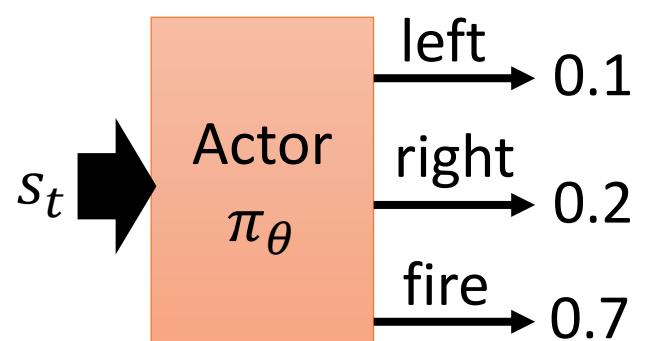
$$p(s_1)p(a_1|s_1, \theta)p(r_1, s_2|s_1, a_1)p(a_2|s_2, \theta)p(r_2, s_3|s_2, a_2)\dots$$

$$= p(s_1) \prod_{t=1}^T p(a_t|s_t, \theta)p(r_t, s_{t+1}|s_t, a_t)$$

not related
to your actor

Control by
your actor π_θ

$$p(a_t = "fire" | s_t, \theta) = 0.7$$



Gradient Ascent

$$\nabla \log P(\tau|\theta) = ?$$

- $\tau = \{s_1, a_1, r_1, s_2, a_2, r_2, \dots, s_T, a_T, r_T\}$

$$P(\tau|\theta) = p(s_1) \prod_{t=1}^T p(a_t|s_t, \theta) p(r_t, s_{t+1}|s_t, a_t)$$

$$\log P(\tau|\theta)$$

$$= \log p(s_1) + \sum_{t=1}^T \log p(a_t|s_t, \theta) + \log p(r_t, s_{t+1}|s_t, a_t)$$

$$\nabla \log P(\tau|\theta) = \sum_{t=1}^T \nabla \log p(a_t|s_t, \theta)$$

Ignore the terms
not related to θ

Gradient Ascent

$$\theta^{new} \leftarrow \theta^{old} + \eta \nabla \bar{R}_{\theta^{old}}$$

$$\begin{aligned}\nabla \bar{R}_{\theta} &\approx \frac{1}{N} \sum_{n=1}^N R(\tau^n) \nabla \log P(\tau^n | \theta) = \frac{1}{N} \sum_{n=1}^N R(\tau^n) \sum_{t=1}^{T_n} \nabla \log p(a_t^n | s_t^n, \theta) \\ &= \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} R(\tau^n) \nabla \log p(a_t^n | s_t^n, \theta)\end{aligned}$$

What if we replace
 $R(\tau^n)$ with r_t^n

If in τ^n machine takes a_t^n when seeing s_t^n in

$R(\tau^n)$ is positive 

Tuning θ to increase $p(a_t^n | s_t^n)$

$R(\tau^n)$ is negative 

Tuning θ to decrease $p(a_t^n | s_t^n)$

It is very important to consider the cumulative reward $R(\tau^n)$ of the whole trajectory τ^n instead of immediate reward r_t^n

Gradient Ascent

$$\theta^{new} \leftarrow \theta^{old} + \eta \nabla \bar{R}_{\theta^{old}}$$

$$\begin{aligned}\nabla \log P(\tau | \theta) \\ = \sum_{t=1}^T \nabla \log p(a_t | s_t, \theta)\end{aligned}$$

$$\begin{aligned}\nabla \bar{R}_\theta &\approx \frac{1}{N} \sum_{n=1}^N R(\tau^n) \nabla \log P(\tau^n | \theta) = \frac{1}{N} \sum_{n=1}^N R(\tau^n) \sum_{t=1}^{T_n} \nabla \log p(a_t^n | s_t^n, \theta) \\ &= \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} R(\tau^n) \boxed{\nabla \log p(a_t^n | s_t^n, \theta)} \quad \frac{\nabla p(a_t^n | s_t^n, \theta)}{p(a_t^n | s_t^n, \theta)}\end{aligned}$$

Why divided by $p(a_t^n | s_t^n, \theta)$?

e.g. in the sampling data ... s has been seen in $\tau^{13}, \tau^{15}, \tau^{17}, \tau^{33}$

In τ^{13} , take action a

$R(\tau^{13}) = 2$

In τ^{15} , take action b

$R(\tau^{15}) = 1$

In τ^{17} , take action b

$R(\tau^{17}) = 1$

In τ^{33} , take action b

$R(\tau^{33}) = 1$

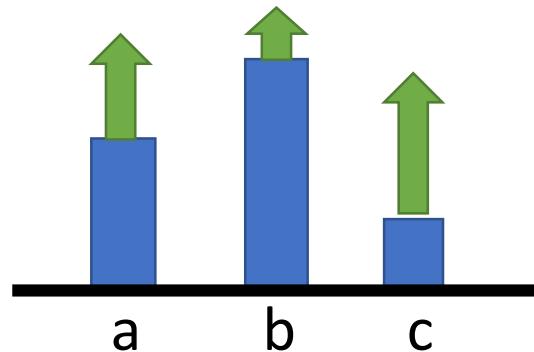
Add a Baseline

It is possible that $R(\tau^n)$ is always positive.

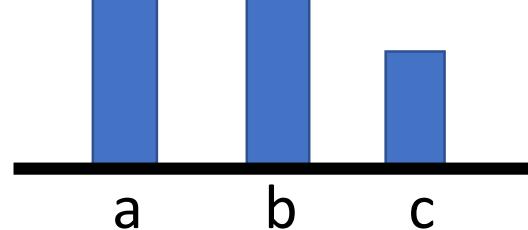
$$\theta^{new} \leftarrow \theta^{old} + \eta \nabla \bar{R}_{\theta^{old}}$$

$$\nabla \bar{R}_{\theta} \approx \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} (R(\tau^n) - b) \nabla \log p(a_t^n | s_t^n, \theta)$$

Ideal case

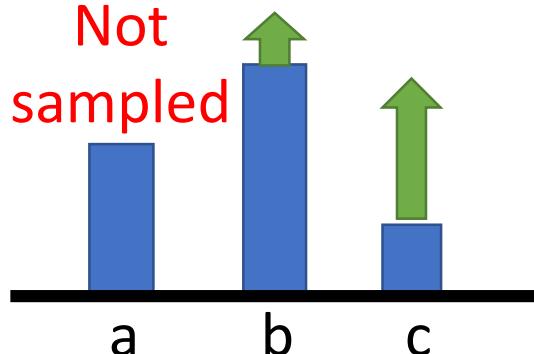


It is probability ...



Sampling

.....



The probability of the actions not sampled will decrease.



Value-based Approach

Learning a Critic

Critic

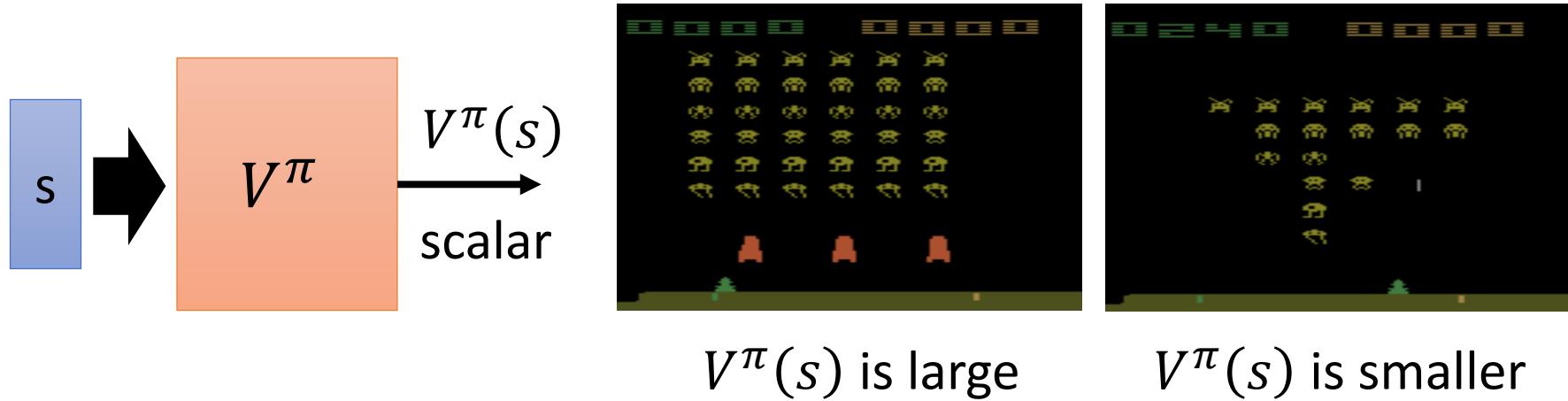
- A critic does not determine the action.
- Given an actor, it evaluates the how good the actor is

An actor can be found from a critic.
e.g. Q-learning
(not today)



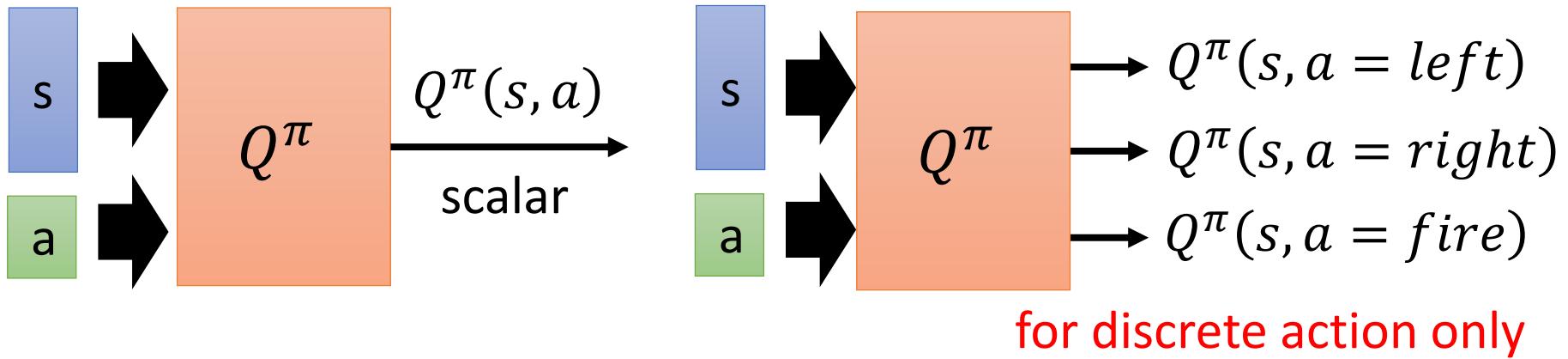
Three kinds of Critics

- A critic is a function depending on the actor π it is evaluated
 - The function is represented by a neural network
- State value function $V^\pi(s)$
 - When using actor π , the *cumulated* reward expects to be obtained after seeing observation (state) s



Three kinds of Critics

- State-action value function $Q^\pi(s, a)$
 - When using actor π , the *cumulated* reward expects to be obtained after seeing observation s and taking a

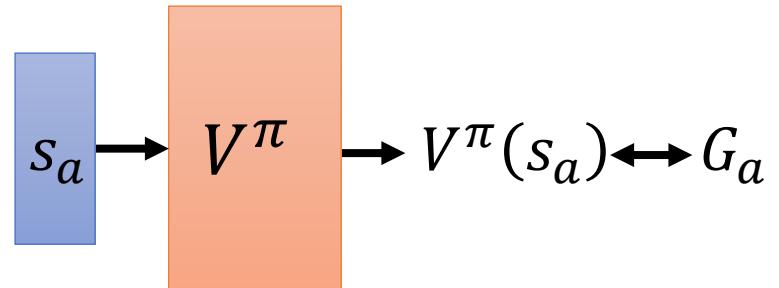


How to estimate $V^\pi(s)$

- Monte-Carlo based approach
 - The critic watches π playing the game

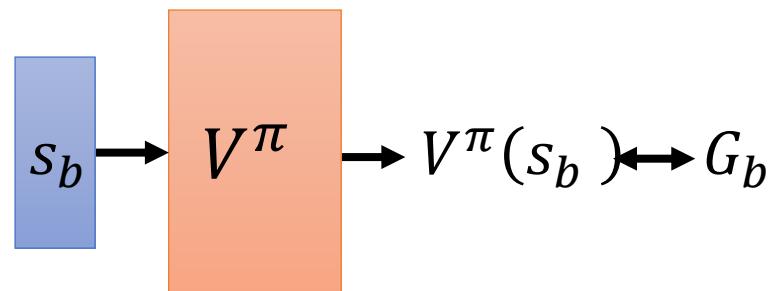
After seeing s_a ,

Until the end of the episode,
the cumulated reward is G_a



After seeing s_b ,

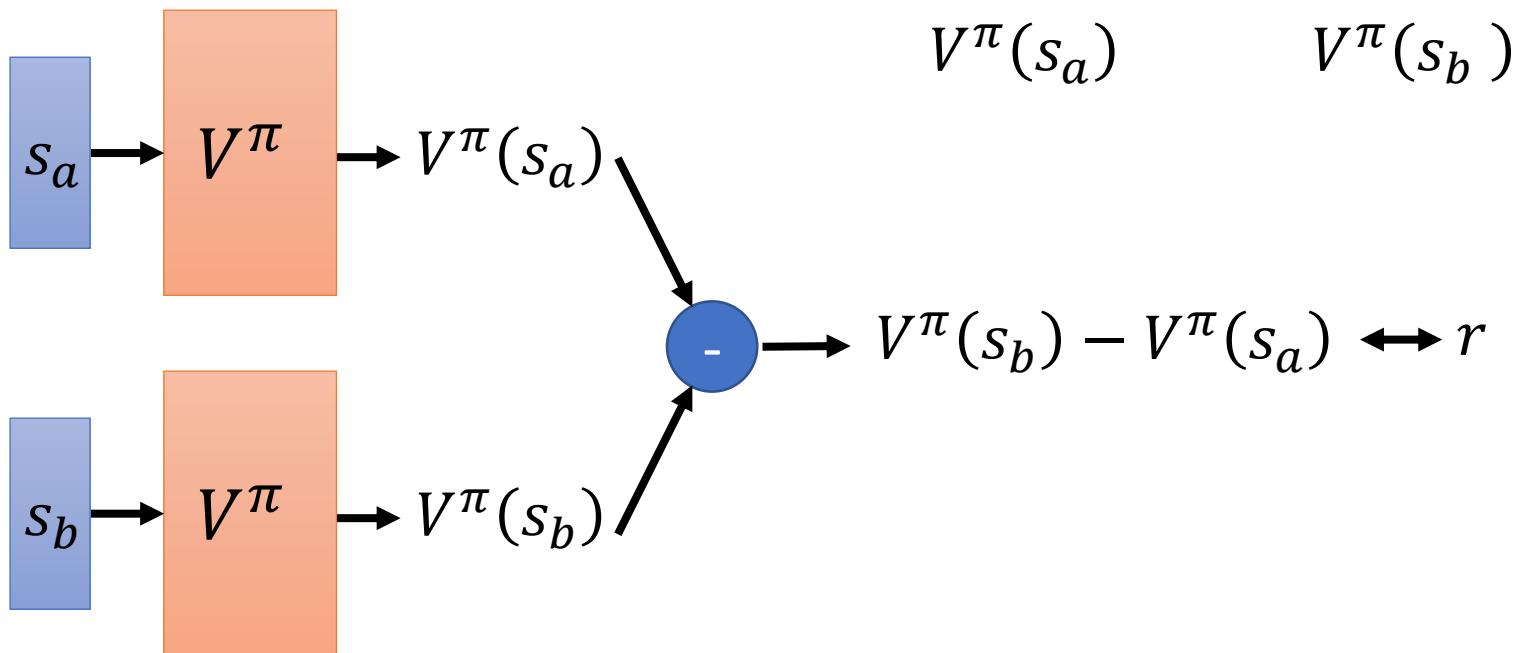
Until the end of the episode,
the cumulated reward is G_b



How to estimate $V^\pi(s)$

- Temporal-difference approach

$\cdots s_a, a, r, s_b \cdots$



Some applications have very long episodes, so that delaying all learning until an episode's end is too slow.

How to estimate $V^\pi(s)$

[Sutton, v2,
Example 6.4]

- The critic has the following 8 episodes
 - $s_a, r = 0, s_b, r = 0, \text{END}$
 - $s_b, r = 1, \text{END}$
 - $s_b, r = 0, \text{END}$
- $V^\pi(s_b) = 3/4$
- $V^\pi(s_a) = ? \quad 0? \quad 3/4?$
- Monte-Carlo: $V^\pi(s_a) = 0$
- Temporal-difference:
- $$V^\pi(s_a) + r = V^\pi(s_b)$$
- $$3/4 \quad 0 \quad 3/4$$

(The actions are ignored here.)

Deep Reinforcement Learning

Actor-Critic

Actor-Critic

$$\theta^{new} \leftarrow \theta^{old} + \eta \nabla \bar{R}_{\theta^{old}}$$

$$\nabla \bar{R}_{\theta} \approx \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} R(\tau^n) \nabla \log p(a_t^n | s_t^n, \theta)$$

Evaluated by critic

$$\text{Advantage Function: } r_t^n - (V^{\pi_\theta}(s_t^n) - V^{\pi_\theta}(s_{t+1}^n))$$

Baseline
is added

The reward r_t^n we truly obtain when taking action a_t^n

Expected reward r_t^n we obtain if we use actor π_θ

Positive advantage function



Increasing the prob. of action a_t^n

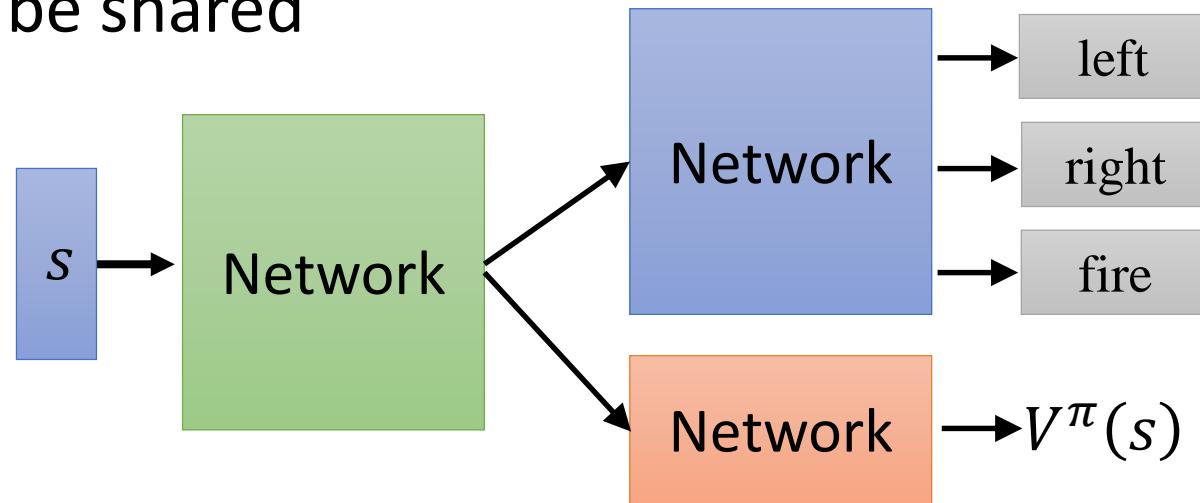
Negative advantage function



decreasing the prob. of action a_t^n

Actor-Critic

- Tips
 - The parameters of actor $\pi(s)$ and critic $V^\pi(s)$ can be shared



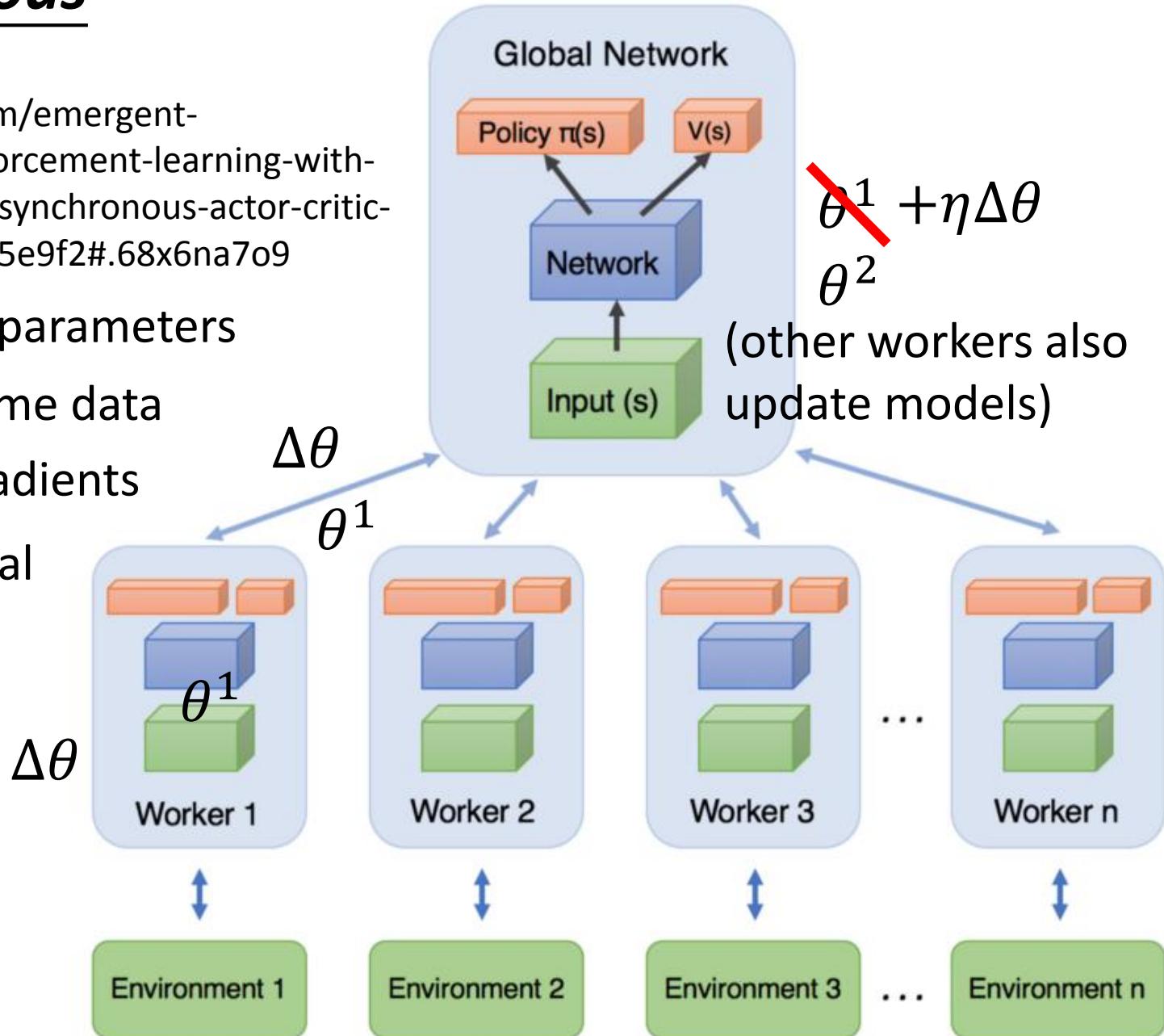
- Use output entropy as regularization for $\pi(s)$
 - Larger entropy is preferred → exploration

Asynchronous

Source of image:

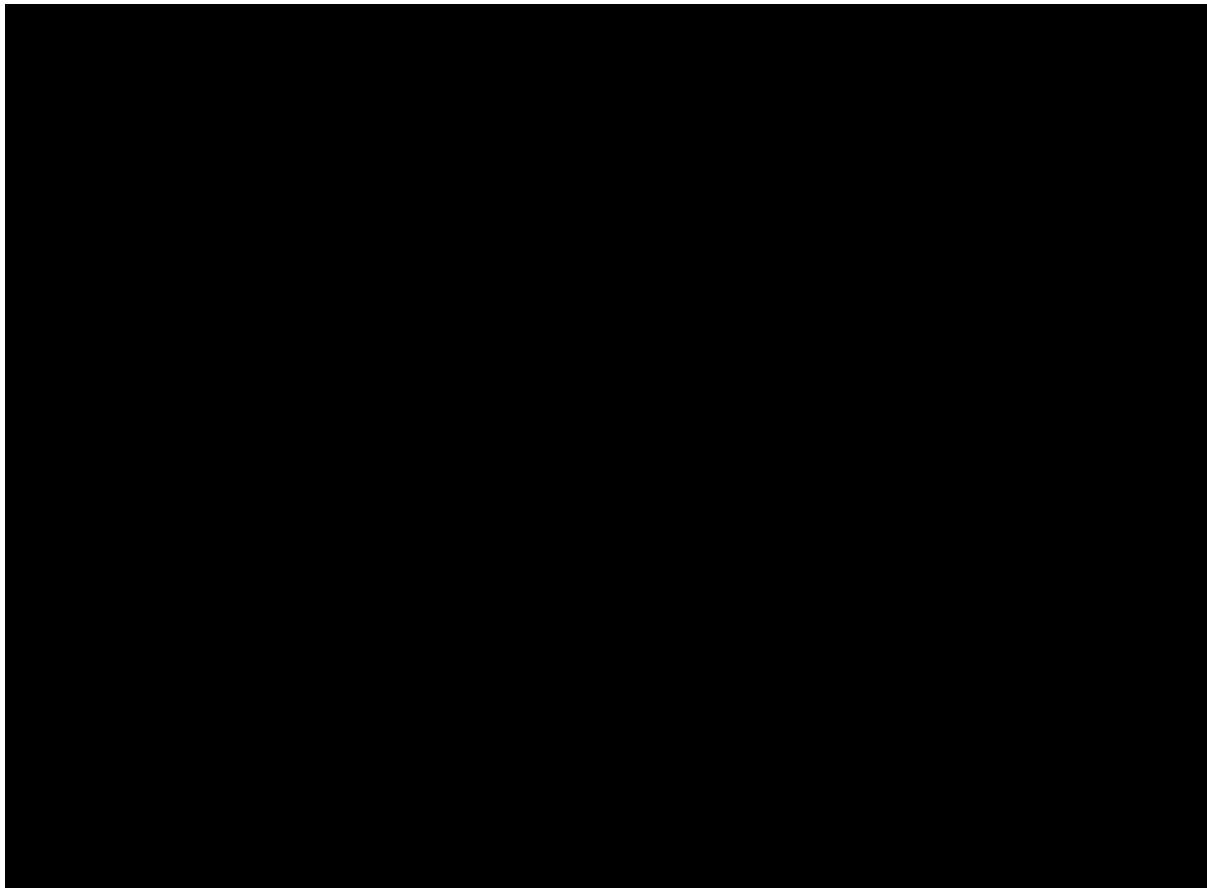
<https://medium.com/emergent-future/simple-reinforcement-learning-with-tensorflow-part-8-async-actor-critic-agents-a3c-c88f72a5e9f2#.68x6na7o9>

1. Copy global parameters
2. Sampling some data
3. Compute gradients
4. Update global models



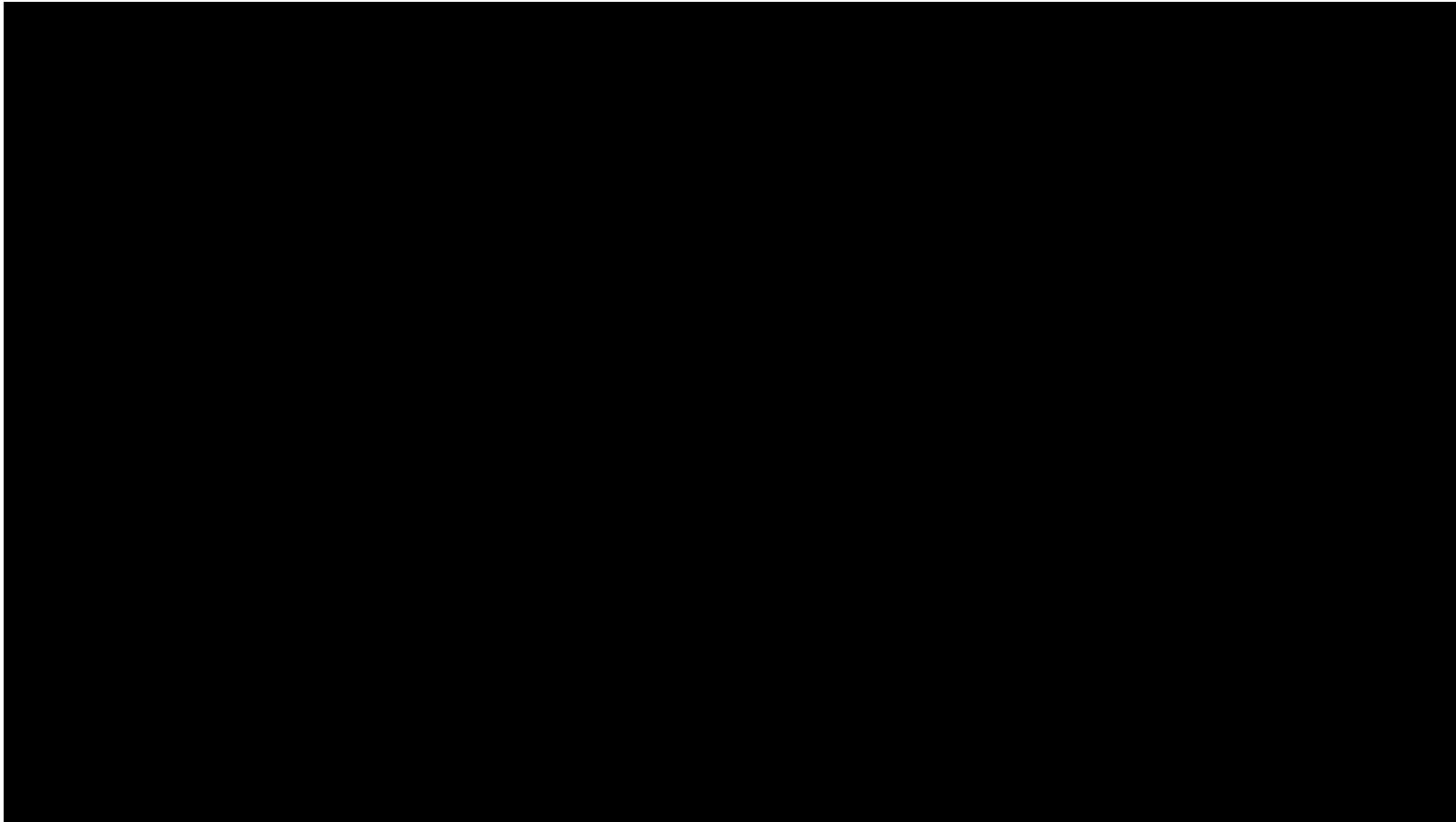
Demo of A3C

- DeepMind <https://www.youtube.com/watch?v=nMR5mjCFZCw>



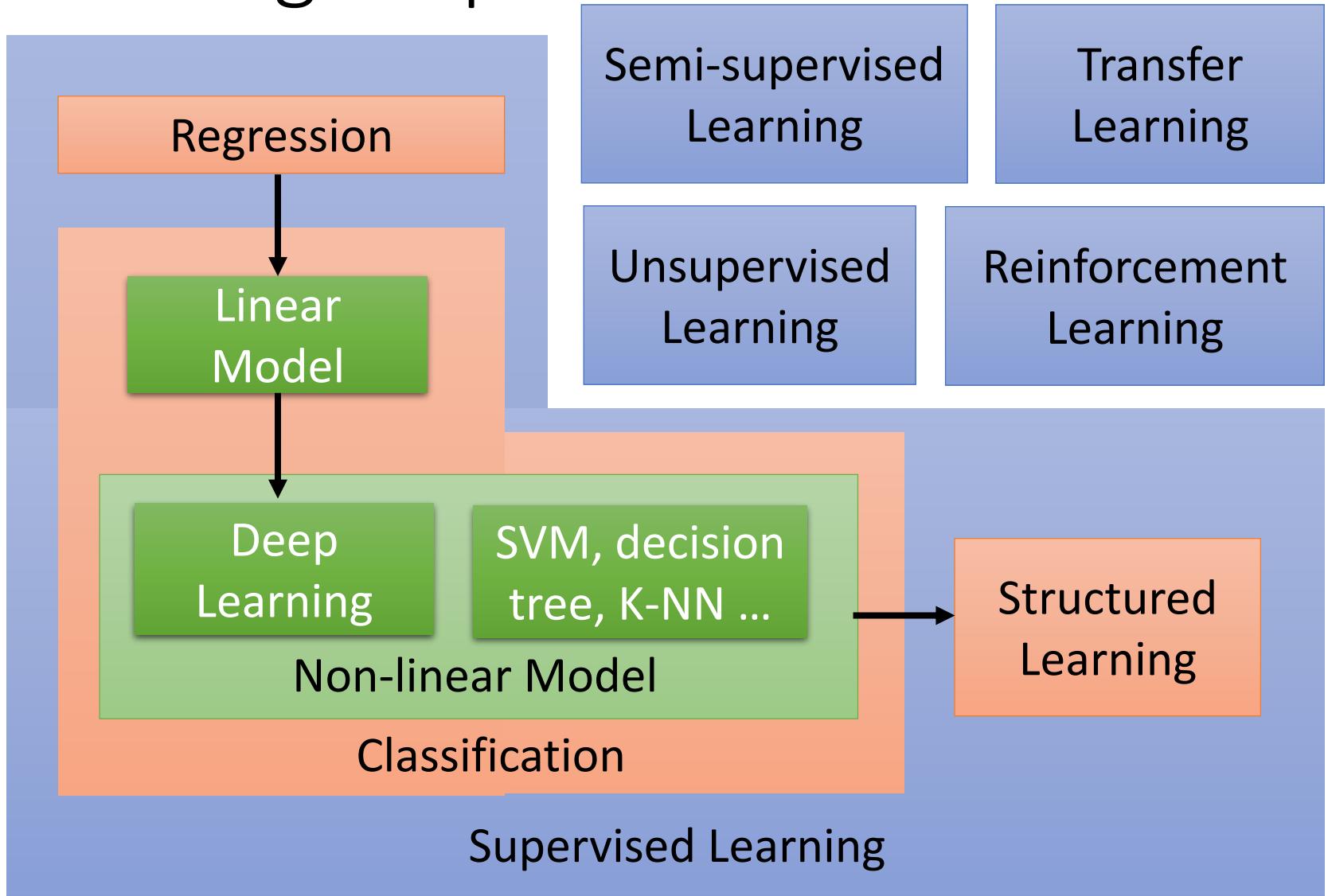
Demo of A3C

- DeepMind <https://www.youtube.com/watch?v=0xo1Ldx3L5Q>



Conclusion
of This Semester

Learning Map



Acknowledgment

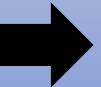
- 感謝 Larry Guo 同學發現投影片上的符號錯誤

Structured Linear Model

Hung-yi Lee

Structured Linear Model

Problem 1: Evaluation

- What does $F(x,y)$ look like?  in a specific form

Problem 2: Inference

- How to solve the “arg max” problem

$$y = \arg \max_{y \in Y} F(x, y)$$

Problem 3: Training

- Given training data, how to find $F(x,y)$

Structured Linear Model:

Problem 1

- **Evaluation:** What does $F(x,y)$ look like?

Characteristics

The diagram illustrates the construction of a structured linear model. On the left, inputs x and y are shown in a grey box. Arrows point from y to three characteristic functions: $\phi_1(x, y)$, $\phi_2(x, y)$, and $\phi_3(x, y)$. Below these, a vertical ellipsis indicates additional characteristics. To the right, a large arrow points down from the input and characteristic section to the final equation. The equation $F(x, y) = w \cdot \phi(x, y)$ is shown, where w is a vector of weights $[w_1, w_2, w_3, \dots]$ and $\phi(x, y)$ is a vector of characteristic functions $[\phi_1(x, y), \phi_2(x, y), \phi_3(x, y), \dots]$.

$$F(x, y) = w_1 \cdot \phi_1(x, y) + w_2 \cdot \phi_2(x, y) + w_3 \cdot \phi_3(x, y) \dots$$

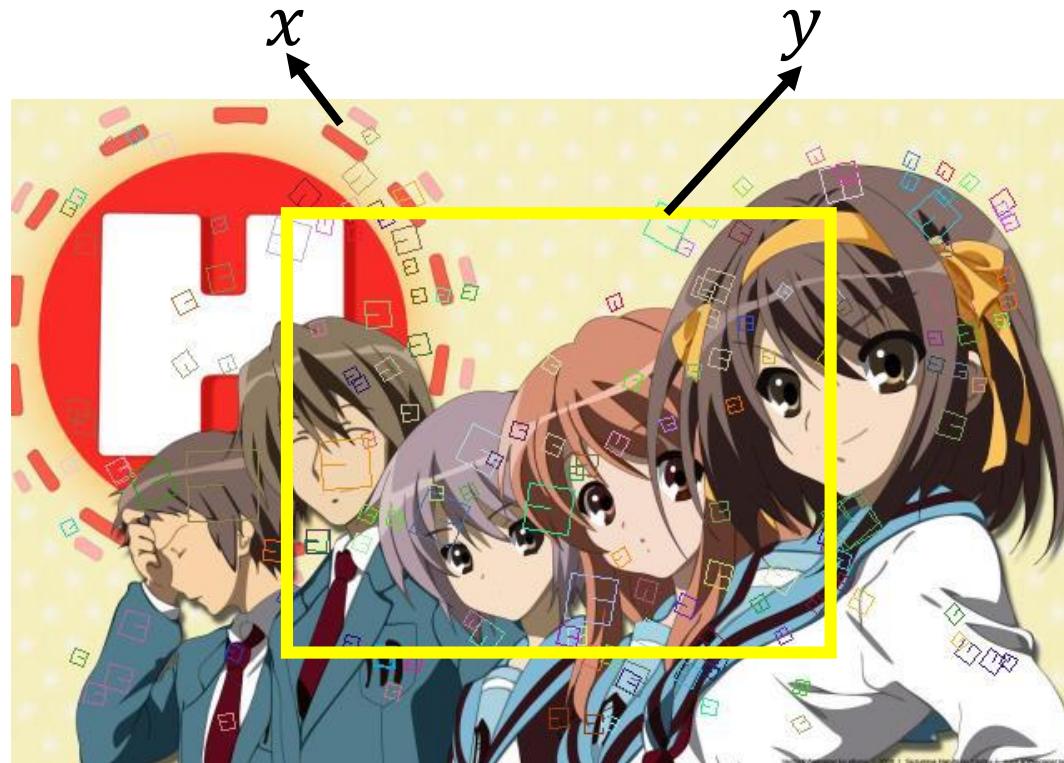
Learning from data

$$F(x, y) = w \cdot \phi(x, y)$$

Structured Linear Model:

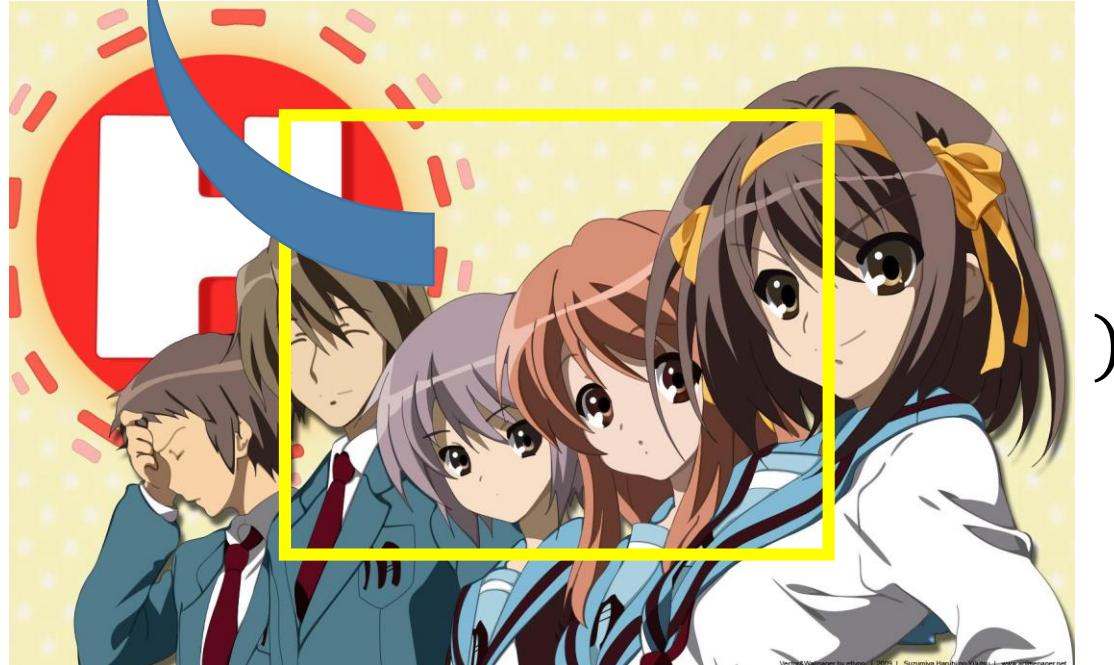
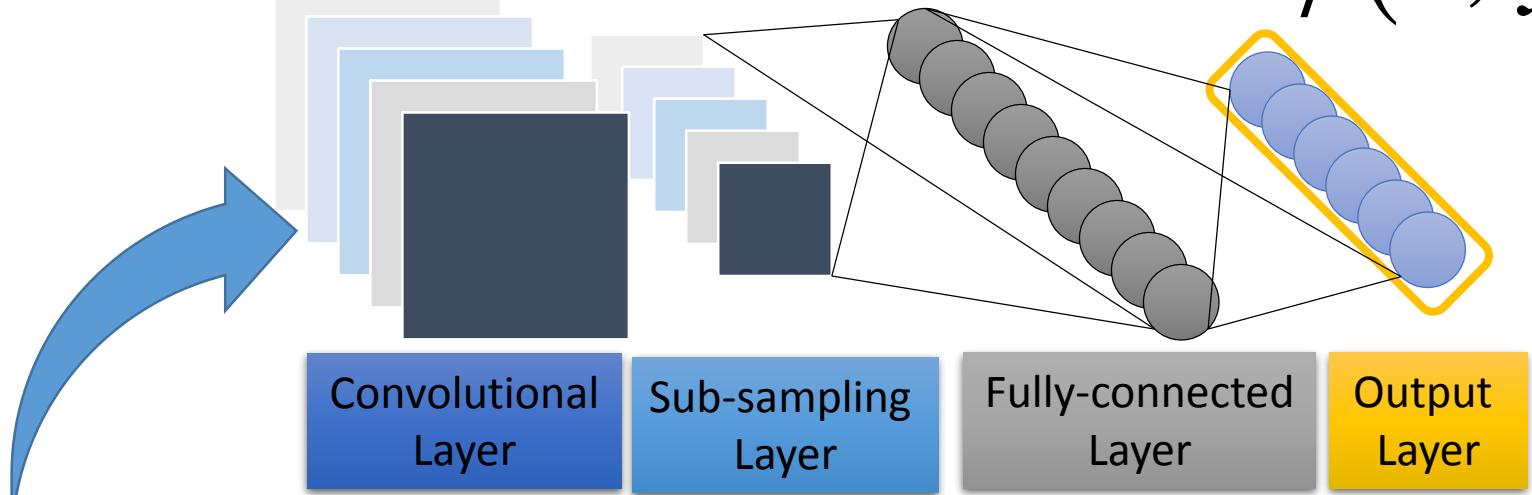
Problem 1

- Evaluation: What does $F(x,y)$ look like?
- Example: Object Detection



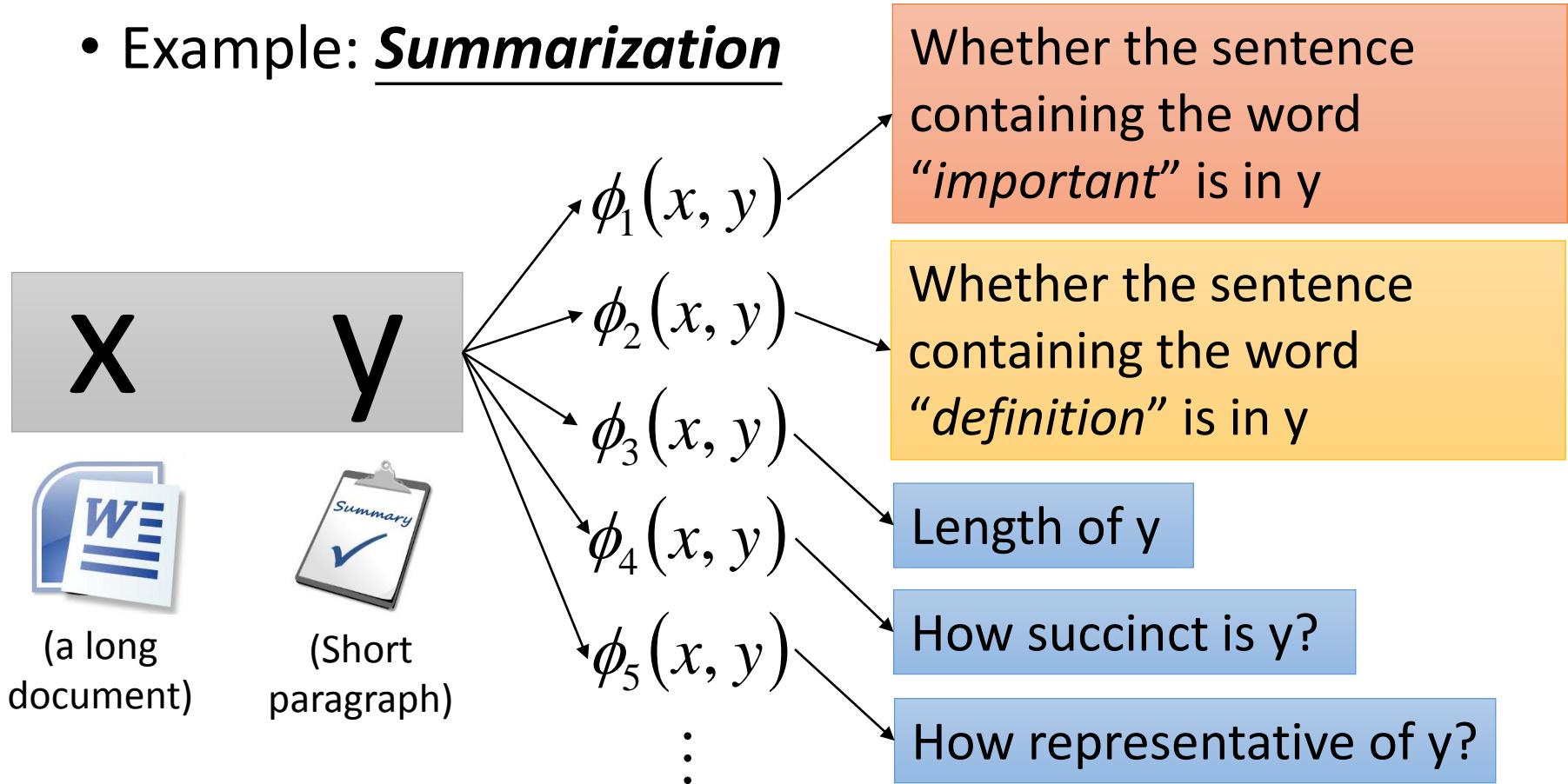
) = [percentage of color red in box y
percentage of color green in box y
percentage of color blue in box y
percentage of color red out of box y
.....
area of box y
number of specific patterns in box y
.....]

$$\phi(x, y)$$



Structured Linear Model: Problem 1

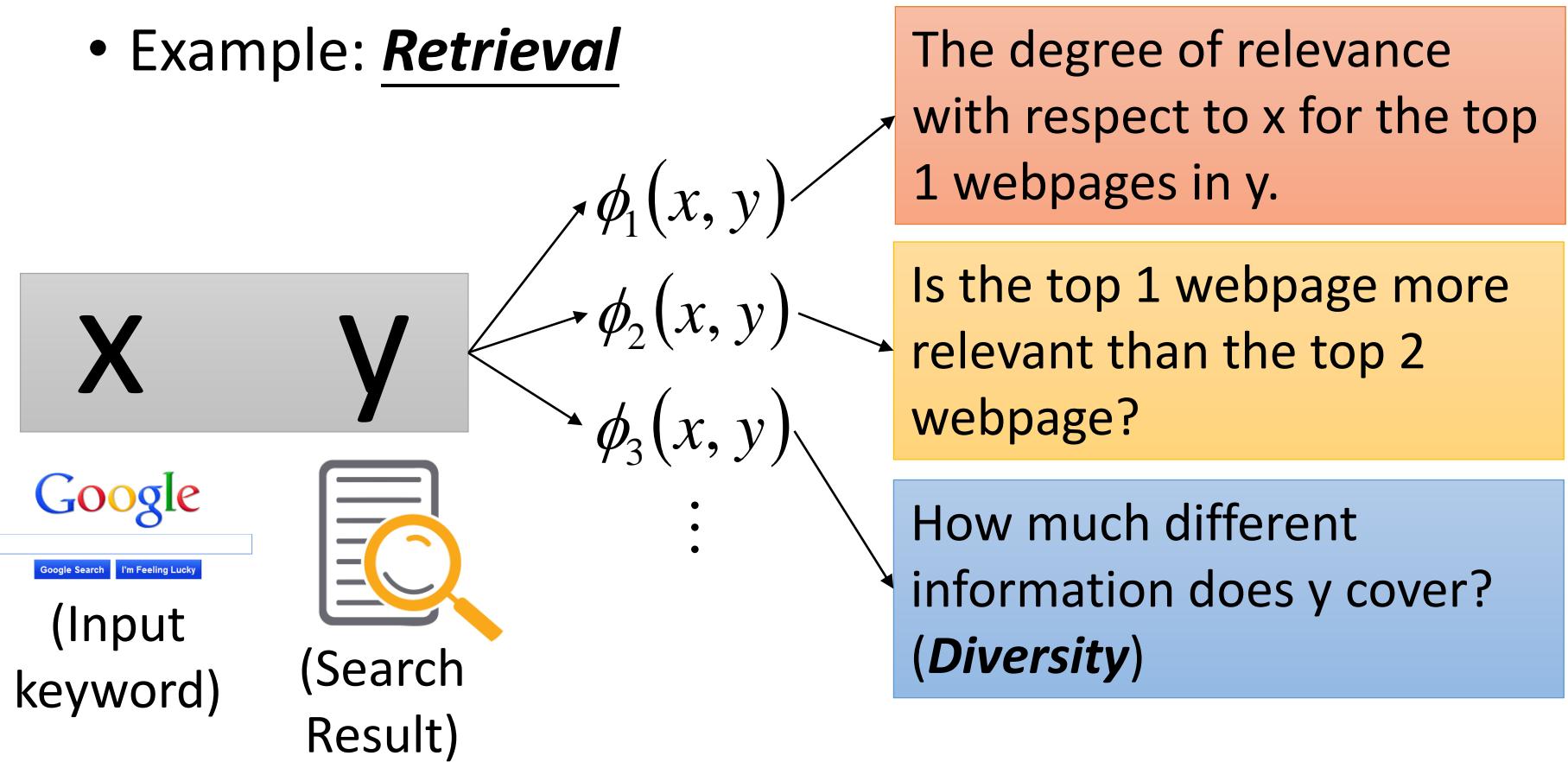
- Evaluation: What does $F(x,y)$ look like?
- Example: Summarization



Structured Linear Model:

Problem 1

- **Evaluation:** What does $F(x,y)$ look like?
- Example: **Retrieval**



Structured Linear Model: Problem 2

- **Inference:** How to solve the “arg max” problem

$$y = \arg \max_{y \in Y} F(x, y)$$

$$F(x, y) = w \cdot \phi(x, y) \rightarrow y = \arg \max_{y \in Y} w \cdot \phi(x, y)$$

- Assume we have solved this question.

Structured Linear Model: Problem 3

- Training: Given training data, how to learn $F(x,y)$
 - $F(x,y) = w \cdot \phi(x,y)$, so what we have to learn is w

Training data: $\{(x^1, \hat{y}^1), (x^2, \hat{y}^2), \dots, (x^r, \hat{y}^r), \dots\}$

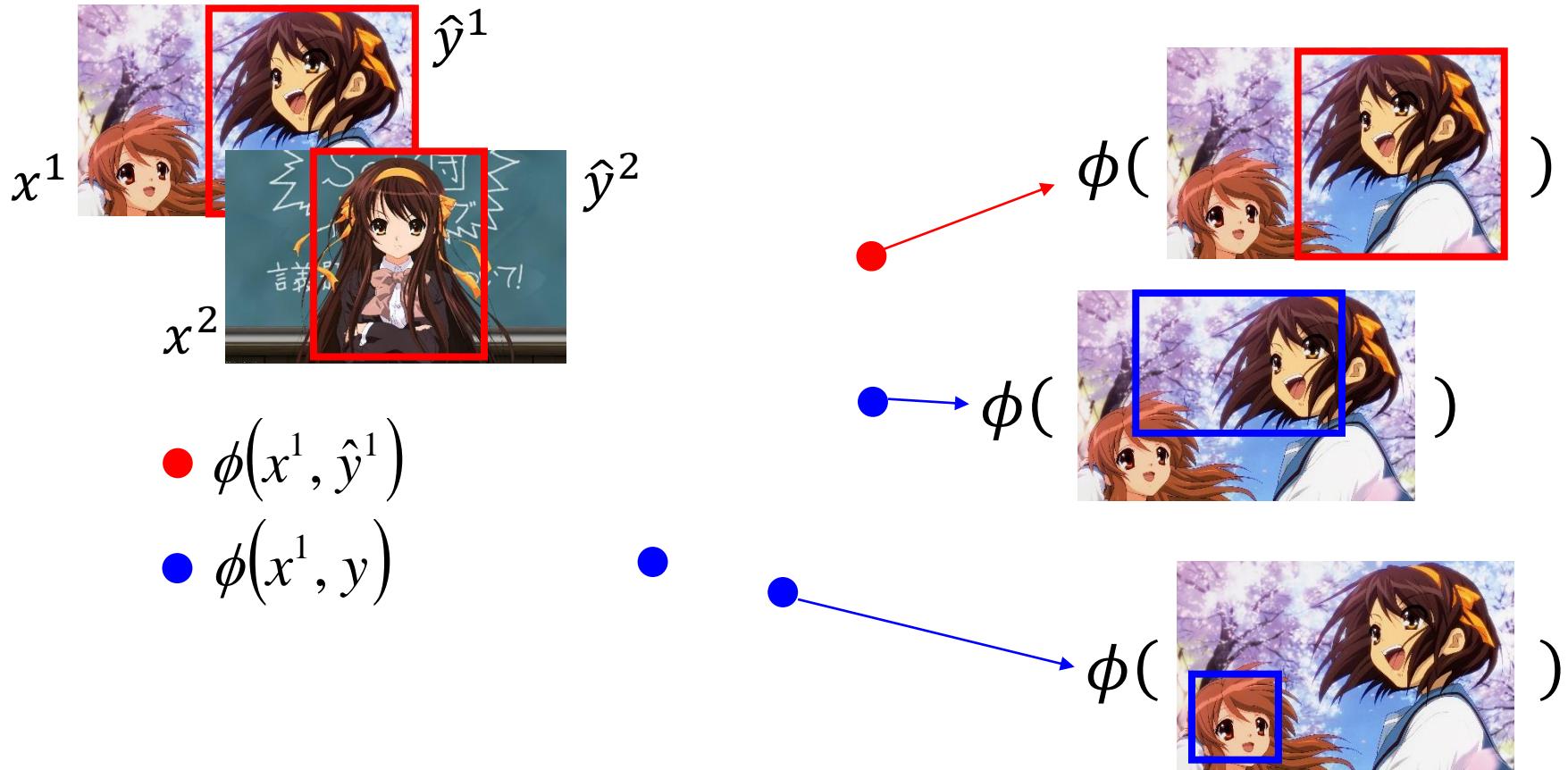
We should find w such that

$\forall r$ (All training examples)

$\forall y \in Y - \{\hat{y}^r\}$ (All incorrect label
for r-th example)

$$w \cdot \phi(x^r, \hat{y}^r) > w \cdot \phi(x^r, y)$$

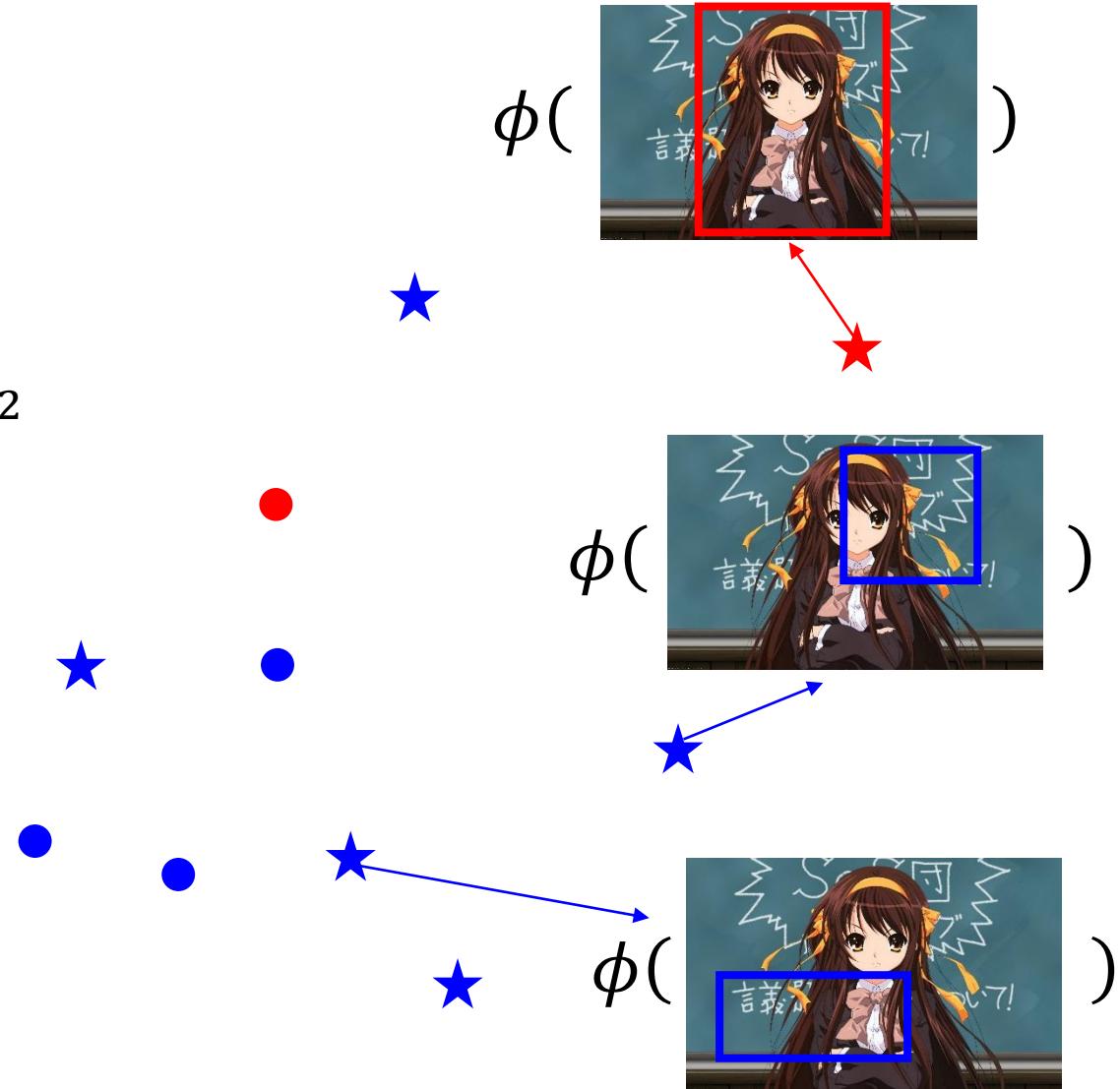
Structured Linear Model: Problem 3



Structured Linear Model: Problem 3



- $\phi(x^1, \hat{y}^1)$
- $\phi(x^1, y)$
- ★ $\phi(x^2, \hat{y}^2)$
- ★ $\phi(x^2, y)$



Structured Linear Model: Problem 3

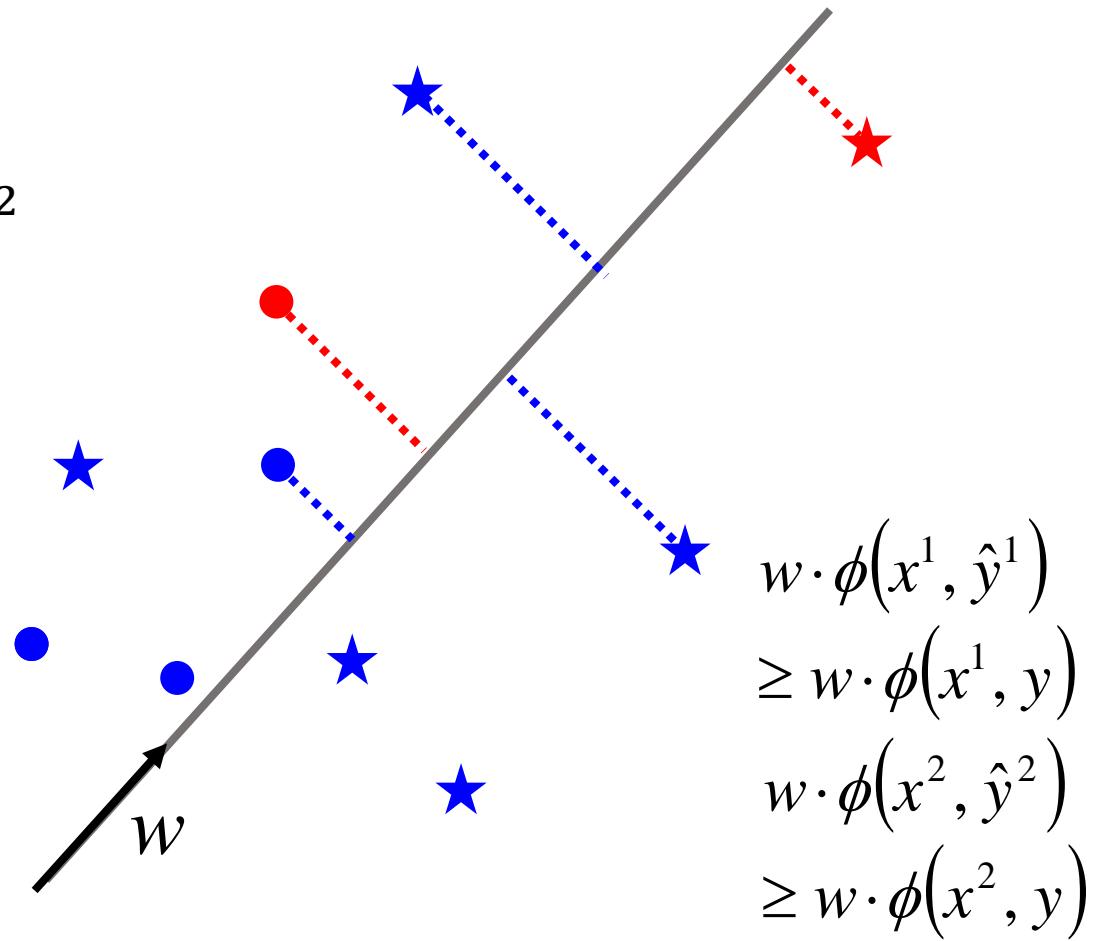


$$\bullet \phi(x^1, \hat{y}^1)$$

$$\bullet \phi(x^1, y)$$

$$\star \phi(x^2, \hat{y}^2)$$

$$\star \phi(x^2, y)$$



Solution of Problem 3

Difficult?

Not as difficult as expected

Algorithm

Will it terminate?

- Input: training data set $\{(x^1, \hat{y}^1), (x^2, \hat{y}^2), \dots, (x^r, \hat{y}^r), \dots\}$
- Output: weight vector w
- Algorithm: Initialize $w = 0$
 - do
 - For each pair of training example (x^r, \hat{y}^r)
 - Find the label \tilde{y}^r maximizing $w \cdot \phi(x^r, y)$
$$\tilde{y}^r = \arg \max_{y \in Y} w \cdot \phi(x^r, y)$$
 (question 2)
 - If $\tilde{y}^r \neq \hat{y}^r$, update w
$$w \rightarrow w + \phi(x^r, \hat{y}^r) - \phi(x^r, \tilde{y}^r)$$
 - until w is not updated  We are done!

Algorithm - Example

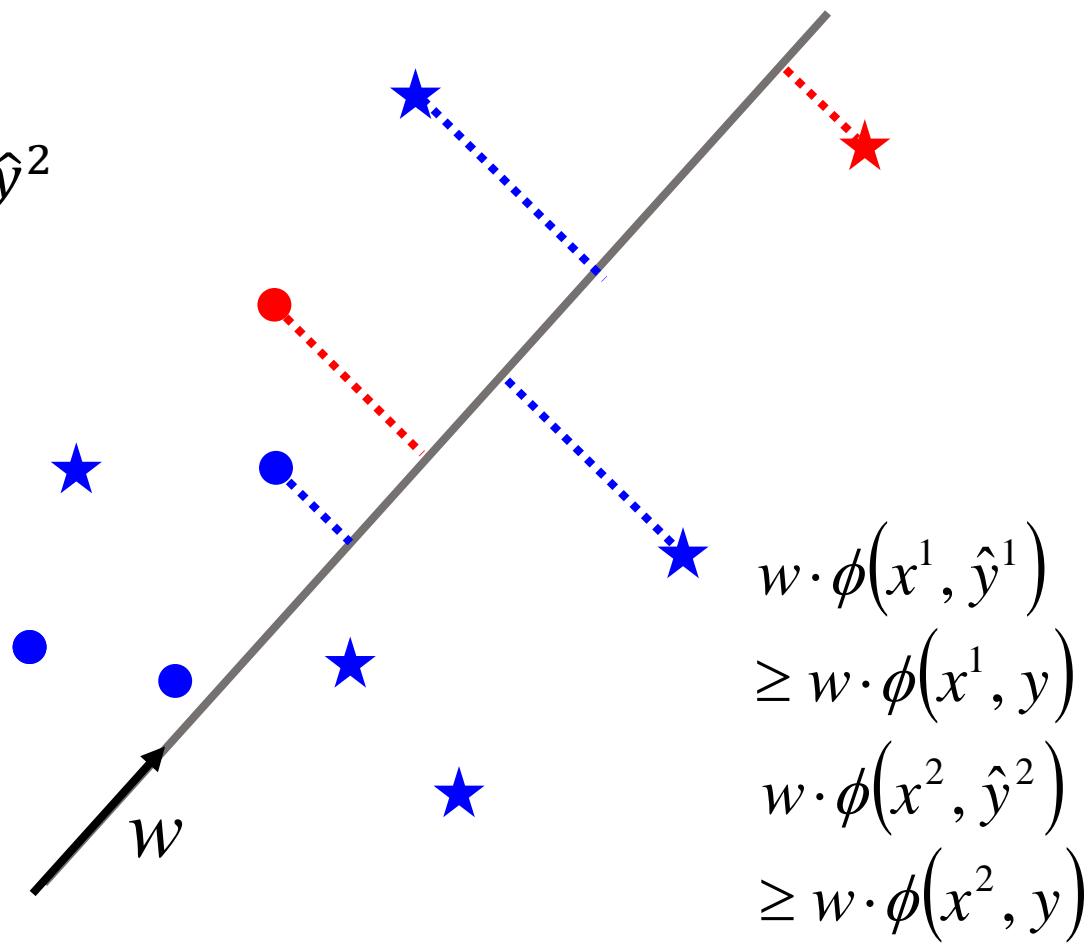


● $\phi(x^1, \hat{y}^1)$

● $\phi(x^1, y)$

★ $\phi(x^2, \hat{y}^2)$

★ $\phi(x^2, y)$



Algorithm - Example

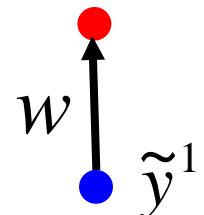
Initialize $w = 0$

pick (x^1, \hat{y}^1)

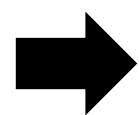
$$\tilde{y}^1 = \arg \max_{y \in Y} w \cdot \phi(x^1, y)$$

If $\tilde{y}^1 \neq \hat{y}^1$, update w

$$w \rightarrow w + \phi(x^1, \hat{y}^1) - \phi(x^1, \tilde{y}^1)$$



Because $w=0$ at this time, $\phi(x^1, y)$ always 0



Random pick one point as \tilde{y}^r

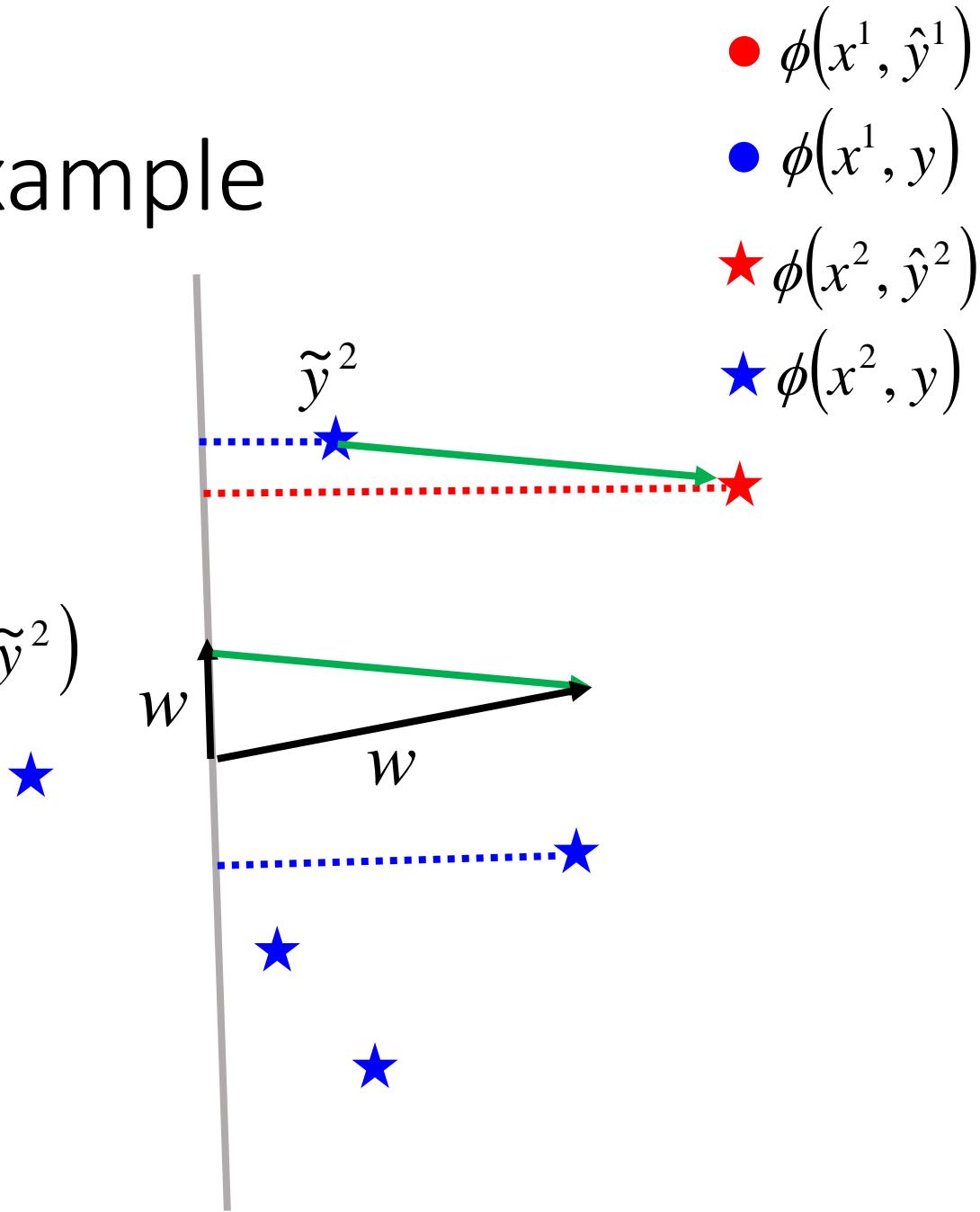
Algorithm - Example

pick (x^2 , \hat{y}^2)

$$\tilde{y}^2 = \arg \max_{y \in Y} w \cdot \phi(x^2, y)$$

If $\tilde{y}^2 \neq \hat{y}^2$, update w

$$w \rightarrow w + \phi(x^2, \hat{y}^2) - \phi(x^2, \tilde{y}^2)$$



Algorithm - Example

pick (x^1, \hat{y}^1) again

$$\tilde{y}^1 = \arg \max_{y \in Y} w \cdot \phi(x^1, y)$$

$$\tilde{y}^1 = \hat{y}^1 \rightarrow \text{do not update } w$$



pick (x^2, \hat{y}^2) again

$$\tilde{y}^2 = \arg \max_{y \in Y} w \cdot \phi(x^2, y)$$

$$\tilde{y}^2 = \hat{y}^2 \rightarrow \text{do not update } w$$

$$\begin{aligned} w \cdot \phi(x^1, \hat{y}^1) &\geq w \cdot \phi(x^1, y) \\ w \cdot \phi(x^2, \hat{y}^2) &\geq w \cdot \phi(x^2, y) \end{aligned}$$

So we are done

Assumption: Separable

- There exists a weight vector \hat{w} $\|\hat{w}\| = 1$

$\forall r$ (All training examples)

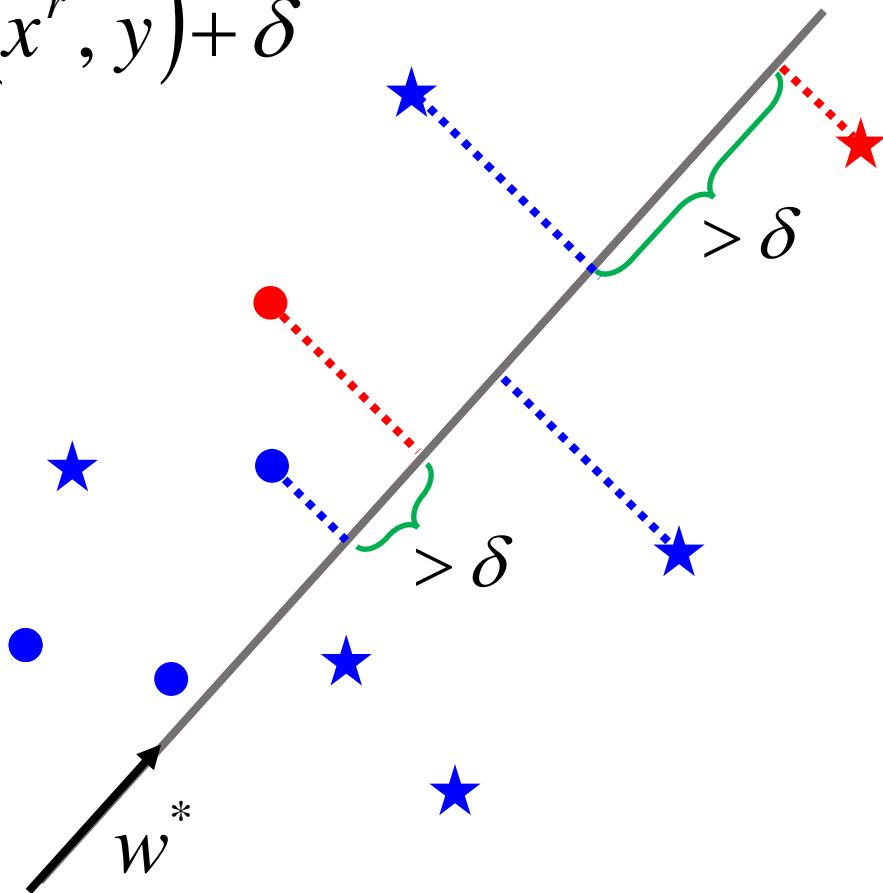
$\forall y \in Y - \{\hat{y}^r\}$ (All incorrect label for an example)


$$\hat{w} \cdot \phi(x^r, \hat{y}^r) \geq \hat{w} \cdot \phi(x^r, y) \quad (\text{The target exists})$$
$$\hat{w} \cdot \phi(x^r, \hat{y}^r) \geq \hat{w} \cdot \phi(x^r, y) + \delta$$

Assumption: Separable

$$\hat{w} \cdot \phi(x^r, \hat{y}^r) \geq \hat{w} \cdot \phi(x^r, y) + \delta$$

- $\phi(x^1, \hat{y}^1)$
- $\phi(x^1, y)$
- ★ $\phi(x^2, \hat{y}^2)$
- ★ $\phi(x^2, y)$
-



Proof of Termination

w is updated once it sees a mistake

$$w^0 = 0 \rightarrow w^1 \rightarrow w^2 \rightarrow \dots \rightarrow w^k \rightarrow w^{k+1} \rightarrow \dots$$

$$w^k = w^{k-1} + \phi(x^n, \hat{y}^n) - \phi(x^n, \tilde{y}^n) \quad (\text{the relation of } w^k \text{ and } w^{k-1})$$

Proof that: The angle ρ_k between \hat{w} and w_k is smaller as k increases

Analysis $\cos \rho_k$ (larger and larger?)

$$\cos \rho_k = \frac{\hat{w} \cdot w^k}{\|\hat{w}\| \cdot \|w^k\|}$$

$$\begin{aligned} \hat{w} \cdot w^k &= \hat{w} \cdot (w^{k-1} + \phi(x^n, \hat{y}^n) - \phi(x^n, \tilde{y}^n)) \\ &= \hat{w} \cdot w^{k-1} + \underbrace{\hat{w} \cdot \phi(x^n, \hat{y}^n) - \hat{w} \cdot \phi(x^n, \tilde{y}^n)}_{\geq \delta \text{ (Separable)}} \geq \hat{w} \cdot w^{k-1} + \delta \end{aligned}$$

Proof of Termination

w is updated once it sees a mistake

$$w^0 = 0 \rightarrow w^1 \rightarrow w^2 \rightarrow \dots \rightarrow w^k \rightarrow w^{k+1} \rightarrow \dots$$

$$w^k = w^{k-1} + \phi(x^n, \hat{y}^n) - \phi(x^n, \tilde{y}^n) \quad (\text{the relation of } w^k \text{ and } w^{k-1})$$

Proof that: The angle ρ_k between \hat{w} and w_k is smaller as k increases

Analysis $\cos \rho_k$ (larger and larger?)

$$\cos \rho_k = \frac{\hat{w} \cdot w^k}{\|\hat{w}\| \cdot \|w^k\|}$$

$$\hat{w} \cdot w^k \geq \hat{w} \cdot w^{k-1} + \delta$$

$$\hat{w} \cdot w^1 \geq \hat{w} \cdot w^0 + \delta \quad \stackrel{=0}{}$$

$$\hat{w} \cdot w^1 \geq \delta$$

$$\hat{w} \cdot w^2 \geq \hat{w} \cdot w^1 + \delta \quad \stackrel{\geq \delta}{\dots}$$

$$\hat{w} \cdot w^2 \geq 2\delta \quad \dots$$

..... }

$\hat{w} \cdot w^k \geq k\delta$
(so what)

Proof of Termination

$$\cos \rho_k = \frac{\hat{w}}{\|\hat{w}\|} \cdot \frac{w^k}{\|w^k\|} \quad w^k = w^{k-1} + \phi(x^n, \hat{y}^n) - \phi(x^n, \tilde{y}^n)$$

$$\begin{aligned} \|w^k\|^2 &= \|w^{k-1} + \phi(x^n, \hat{y}^n) - \phi(x^n, \tilde{y}^n)\|^2 \\ &= \|w^{k-1}\|^2 + \underbrace{\|\phi(x^n, \hat{y}^n) - \phi(x^n, \tilde{y}^n)\|^2}_{> 0} + \underbrace{2w^{k-1} \cdot (\phi(x^n, \hat{y}^n) - \phi(x^n, \tilde{y}^n))}_{? < 0 \text{ (mistake)}} \end{aligned}$$

Assume the distance
between any two feature
vector is smaller than R

$$\leq \|w^{k-1}\| + R^2$$

$$\|w^1\|^2 \leq \|w^0\|^2 + R^2 = R^2$$

$$\|w^2\|^2 \leq \|w^1\|^2 + R^2 \leq 2R^2$$

...

$$\|w^k\|^2 \leq kR^2$$

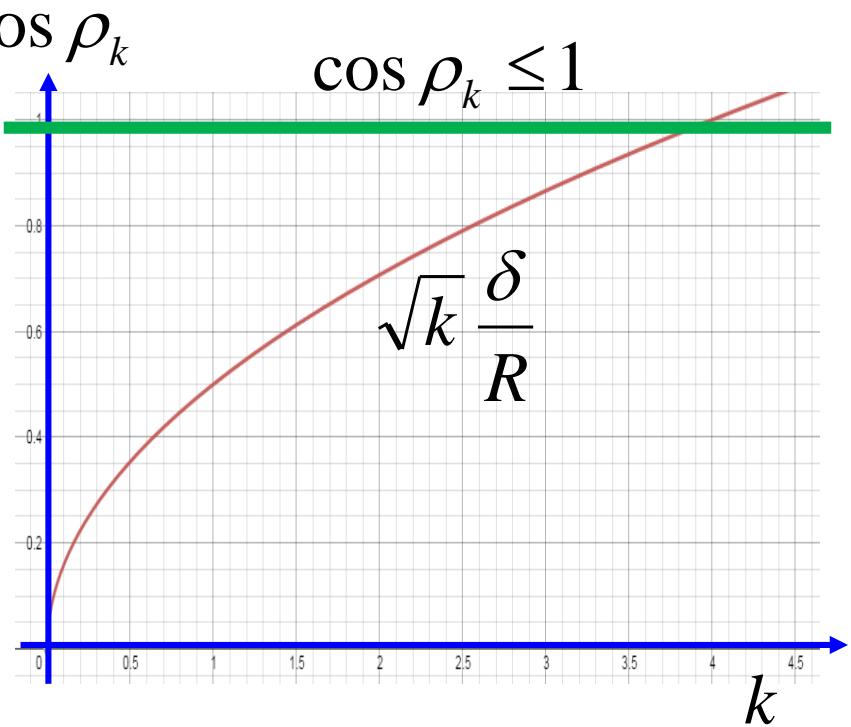
Proof of Termination

$$\cos \rho_k = \frac{\hat{w}}{\|\hat{w}\|} \cdot \frac{w^k}{\|w^k\|} \quad \hat{w} \cdot w^k \geq k\delta \quad \|w^k\|^2 \leq kR^2$$

$$\geq \frac{k\delta}{\sqrt{kR^2}} = \sqrt{k} \frac{\delta}{R}$$

$$\sqrt{k} \frac{\delta}{R} \leq 1$$

$$k \leq \left(\frac{R}{\delta} \right)^2$$



Proof of Termination

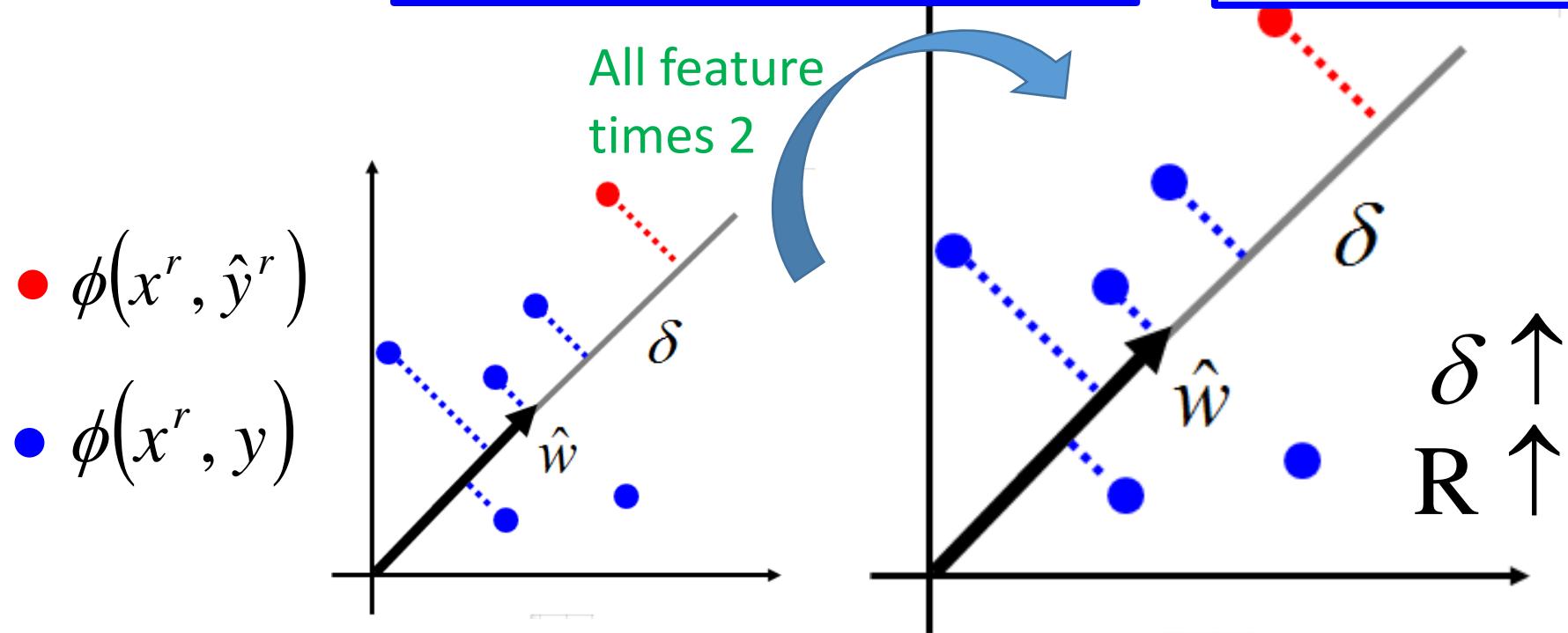
$$k \leq \left(\frac{R}{\delta}\right)^2$$

The largest distances between features

Margin: Is it easy to separable red points from the blue ones

Normalization

Larger margin, less update



Structured Linear Model: Reduce 3 Problems to 2

Problem 1: Evaluation

- How to define $F(x,y)$

Problem 2: Inference

- How to find the y with the largest $F(x,y)$

Problem 3: Training

- How to learn $F(x,y)$

$$F(x,y) = w \cdot \phi(x,y)$$

Problem A: Feature

- How to define $\phi(x,y)$

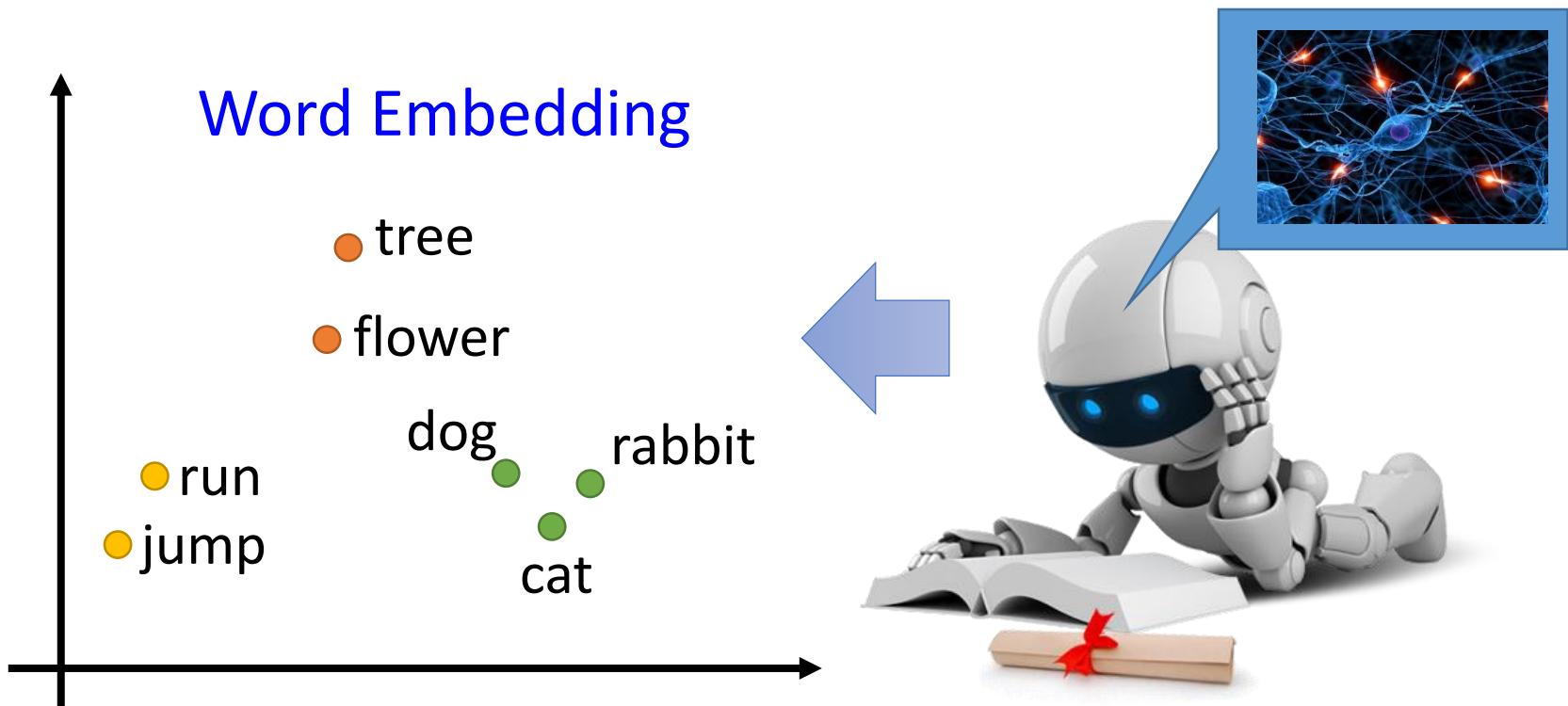
Problem B: Inference

- How to find the y with the largest $w \cdot \phi(x,y)$

Unsupervised Learning: Word Embedding

Word Embedding

- Machine learns the meaning of words from reading a lot of documents without supervision



1-of-N Encoding

apple = [1 0 0 0 0]

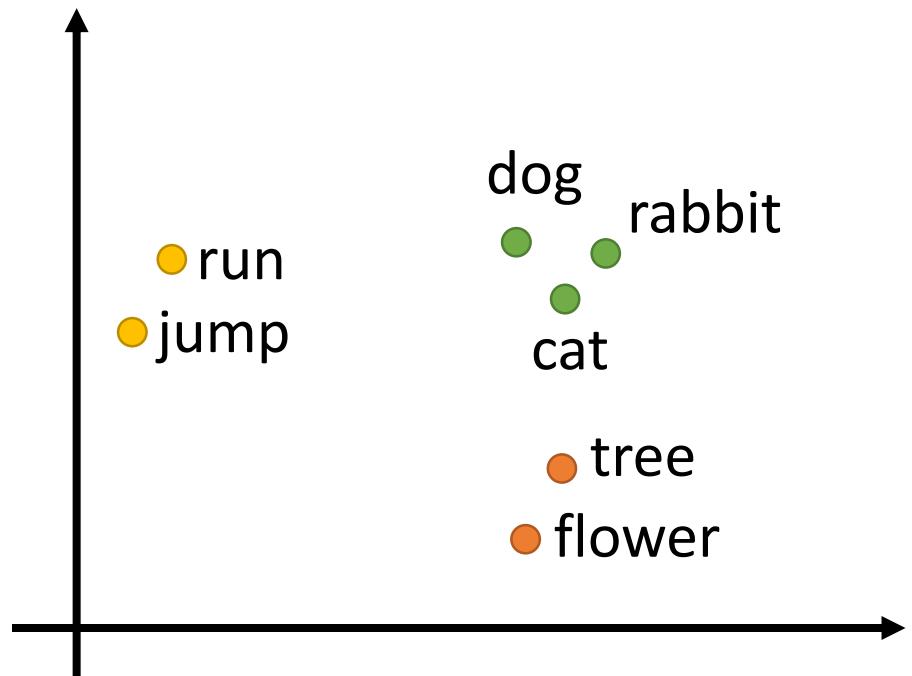
bag = [0 1 0 0 0]

cat = [0 0 1 0 0]

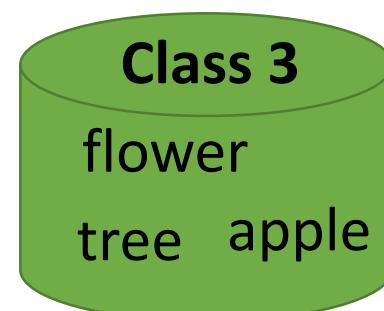
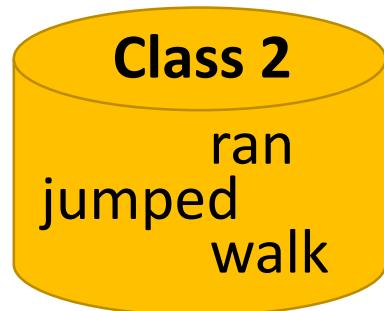
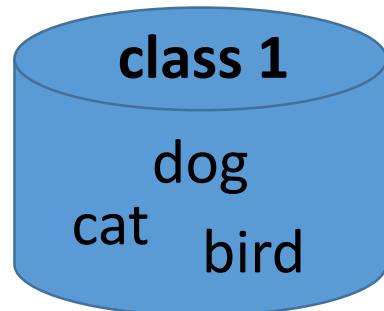
dog = [0 0 0 1 0]

elephant = [0 0 0 0 1]

Word Embedding



Word Class



Word Embedding

- Machine learns the meaning of words from reading a lot of documents without supervision
- A word can be understood by its context

蔡英文、馬英九 are something very similar

You shall know a word by the company it keeps

馬英九 520宣誓就職

蔡英文 520宣誓就職



How to exploit the context?

- **Count based**

- If two words w_i and w_j frequently co-occur, $V(w_i)$ and $V(w_j)$ would be close to each other
- E.g. Glove Vector:
<http://nlp.stanford.edu/projects/glove/>

 $V(w_i) \cdot V(w_j)$  $N_{i,j}$

Inner product

Number of times w_i and w_j
in the same document

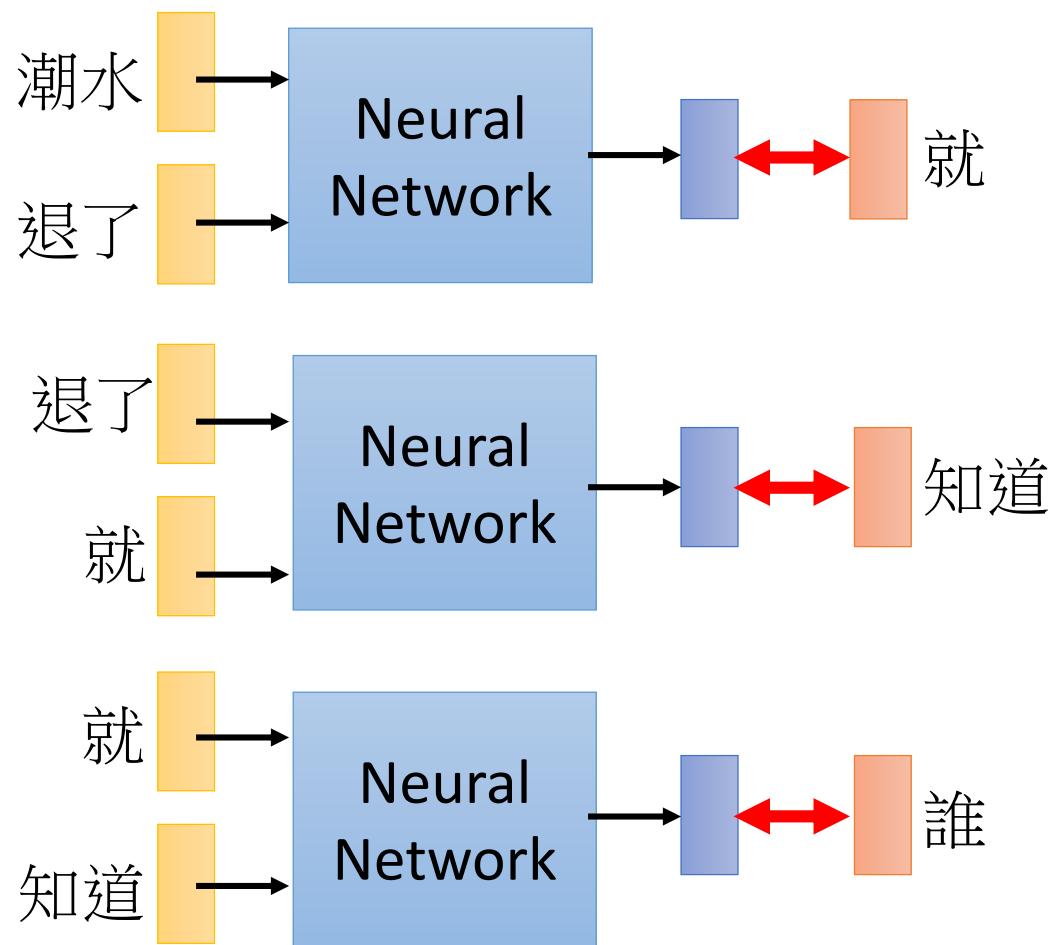
- **Perdition based**

Prediction-based – Training

Collect data:

潮水 退了 就 知道 誰 ...
不爽 不要 買 ...
公道價 八萬 一 ...
.....

Minimizing
cross entropy



Prediction-based - 推文接話

推 louisee :話說十幾年前我念公立國中時,老師也曾做過這種事,但

<https://www.ptt.cc/bbs/Teacher/M.1317226791.A.558.html>

推 AO56789: 我同學才扯好不好，他有一次要交家政料理報告
→ AO56789:其中一個是要寫一樣水煮料理的食譜，他居然給我寫

著名簽名檔 (出處不詳)

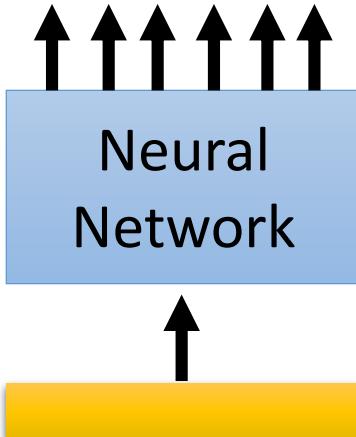
Prediction-based – Language Modeling

$P(\text{"wreck a nice beach"})$

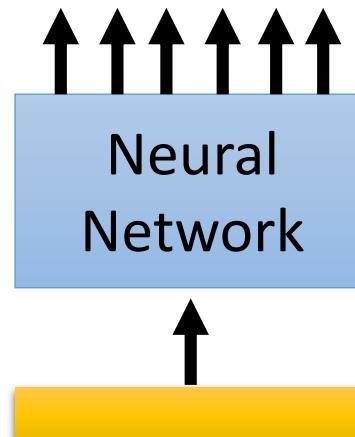
$$= P(\text{wreck} | \text{START}) P(\text{a} | \text{wreck}) P(\text{nice} | \text{a}) P(\text{beach} | \text{nice})$$

$P(b|a)$: the probability of NN predicting the next word.

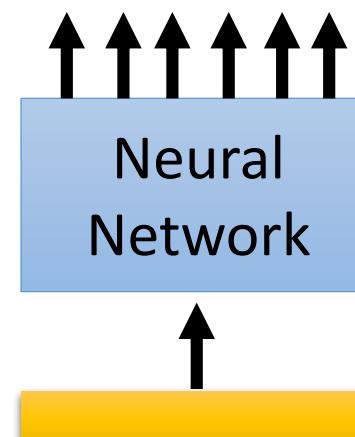
$P(\text{next word is}$
 $\text{"wreck"})$



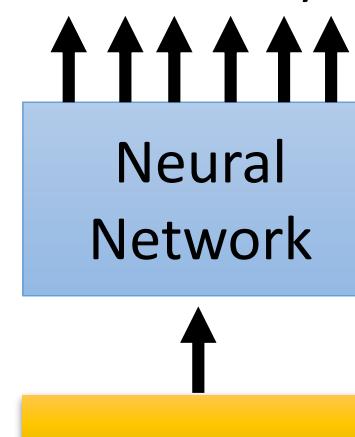
$P(\text{next word is}$
 $\text{"a"})$



$P(\text{next word is}$
 $\text{"nice"})$



$P(\text{next word is}$
 $\text{"beach"})$

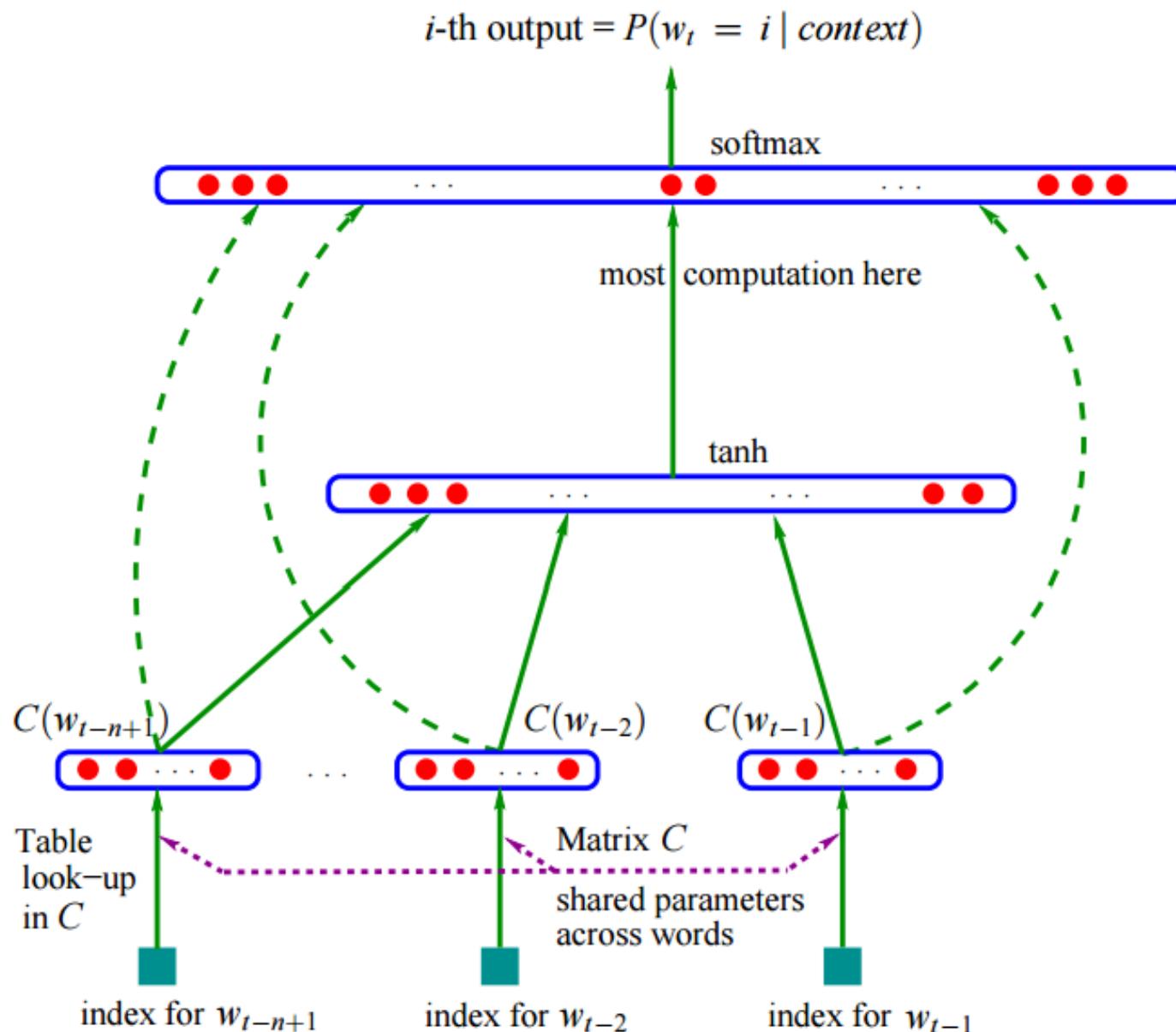


1-of-N encoding
of "START"

1-of-N encoding
of "wreck"

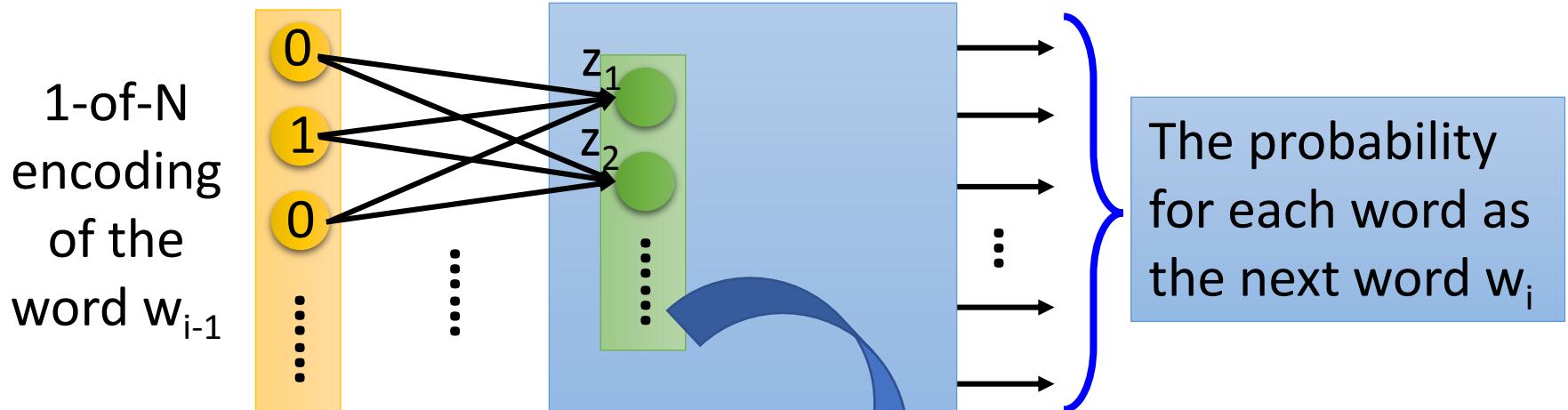
1-of-N encoding
of "a"

1-of-N encoding
of "nice"

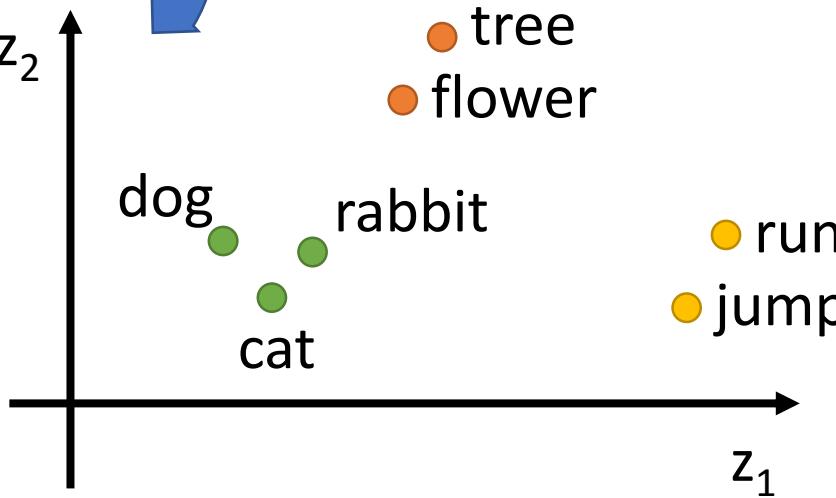


Bengio, Y., Ducharme, R., Vincent, P., & Jauvin, C. (2003). A neural probabilistic language model. *Journal of machine learning research*, 3(Feb), 1137-1155.

Prediction-based



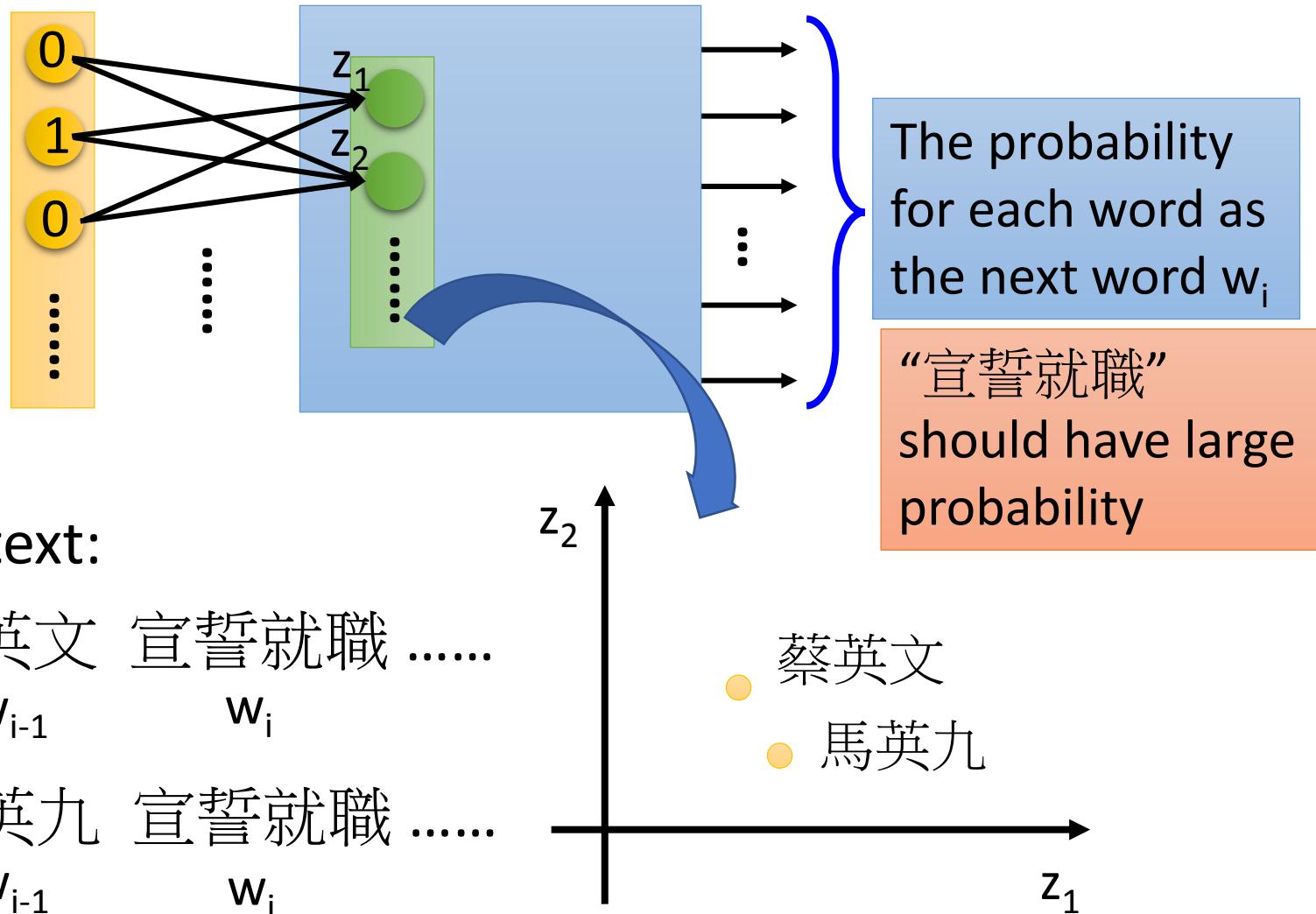
- Take out the input of the neurons in the first layer
- Use it to represent a word w
- Word vector, word embedding feature: $V(w)$



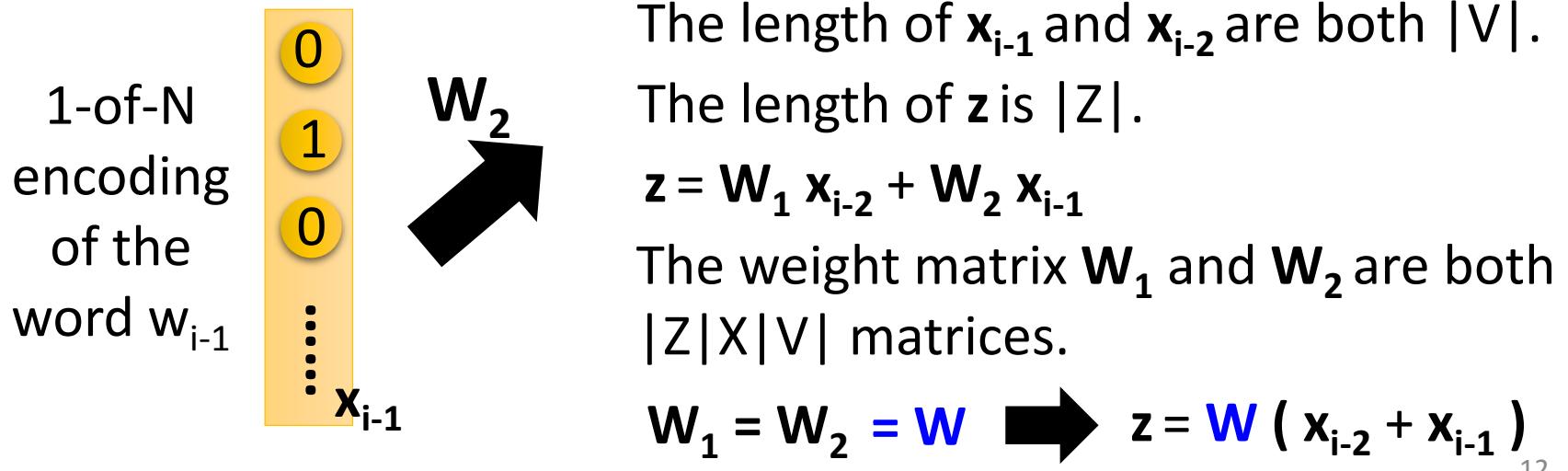
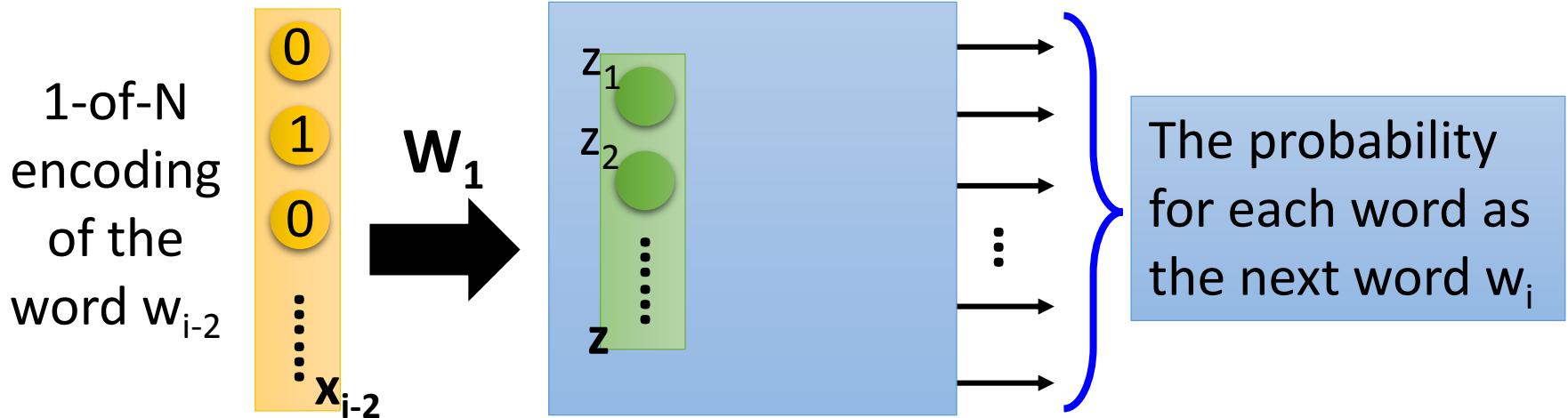
Prediction-based

You shall know a word
by the company it keeps

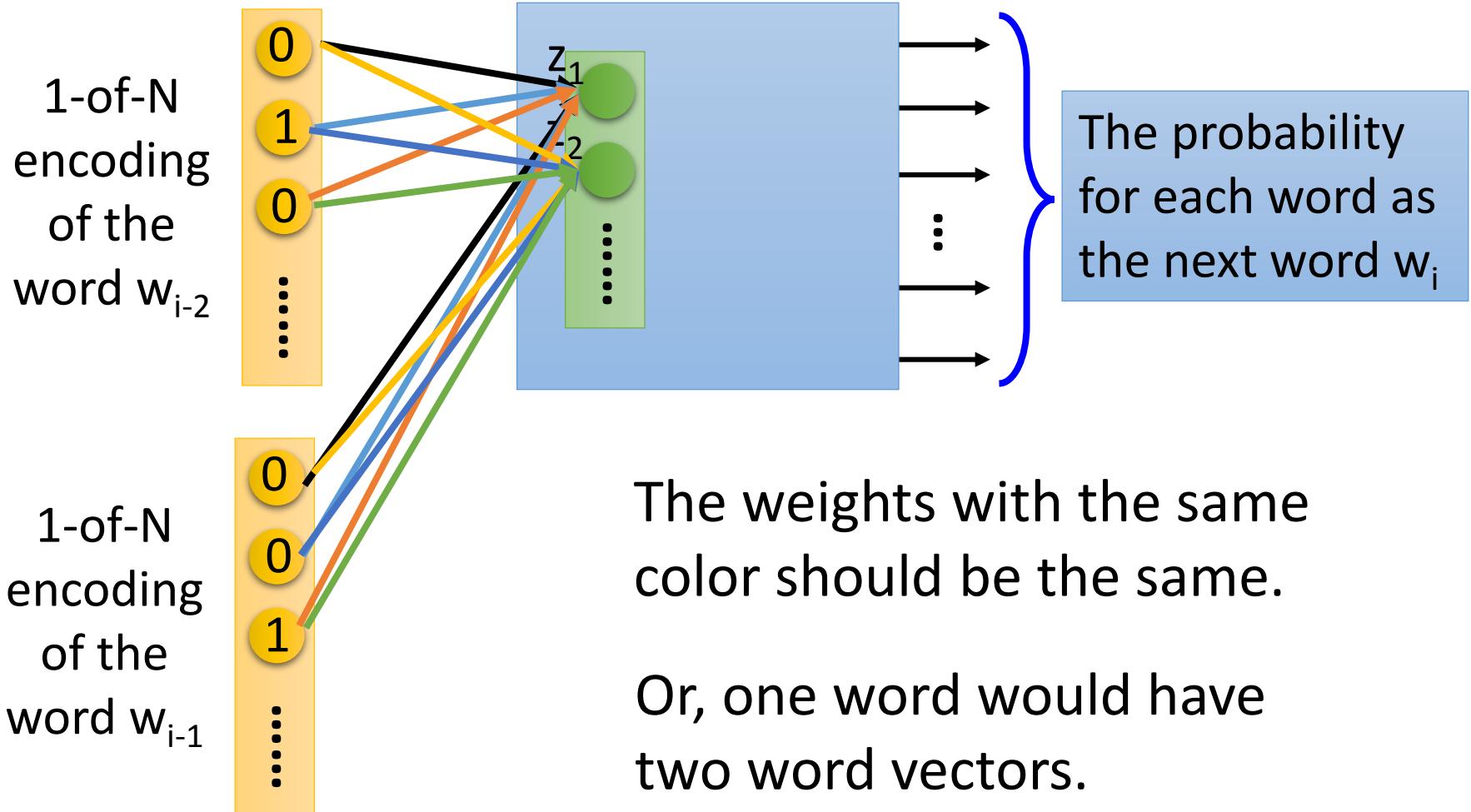
蔡英文
or
馬英九



Prediction-based – Sharing Parameters



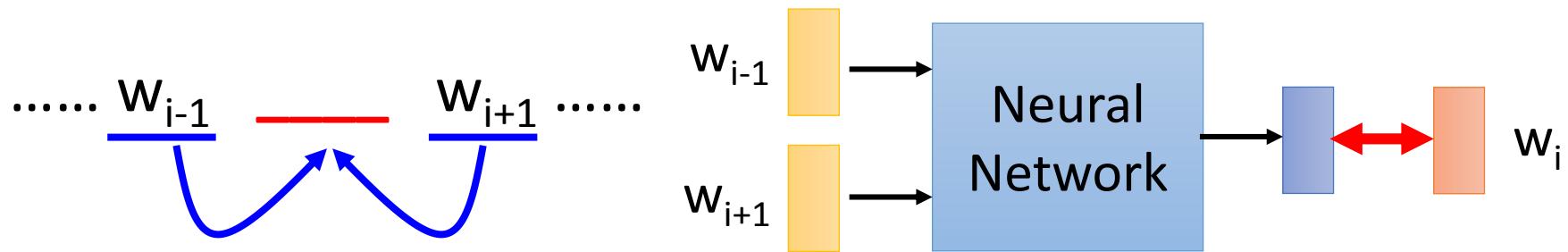
Prediction-based – Sharing Parameters



Prediction-based

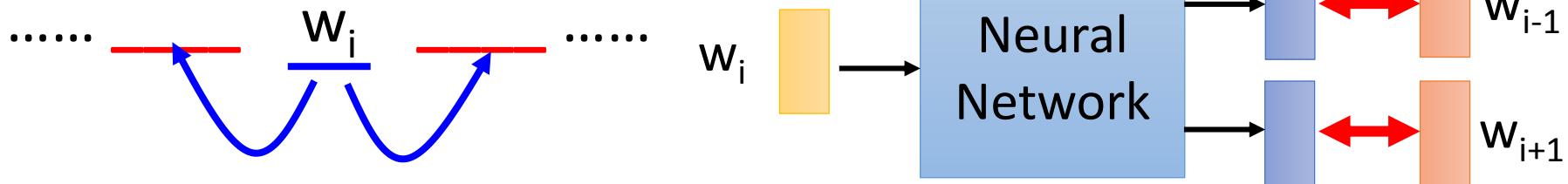
– Various Architectures

- Continuous bag of word (CBOW) model



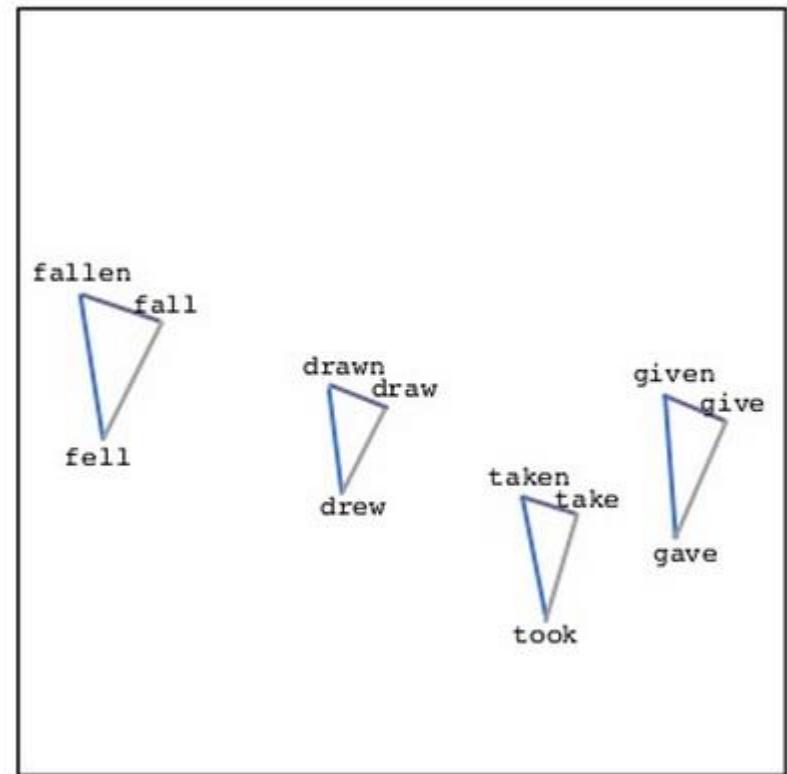
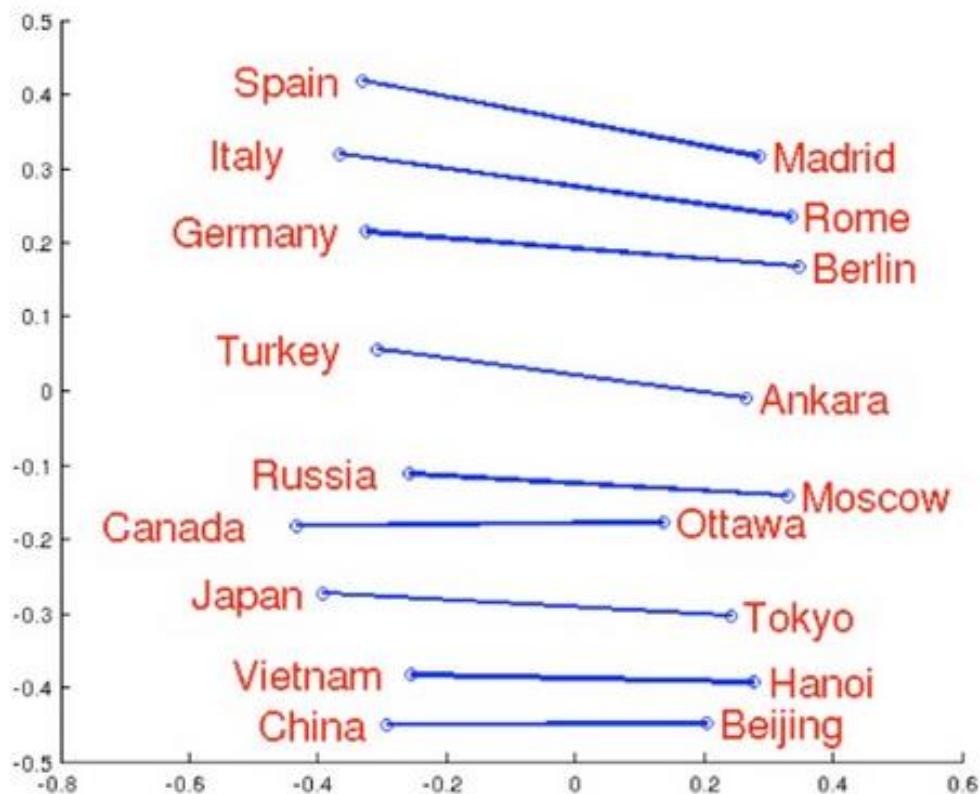
predicting the word given its context

- Skip-gram



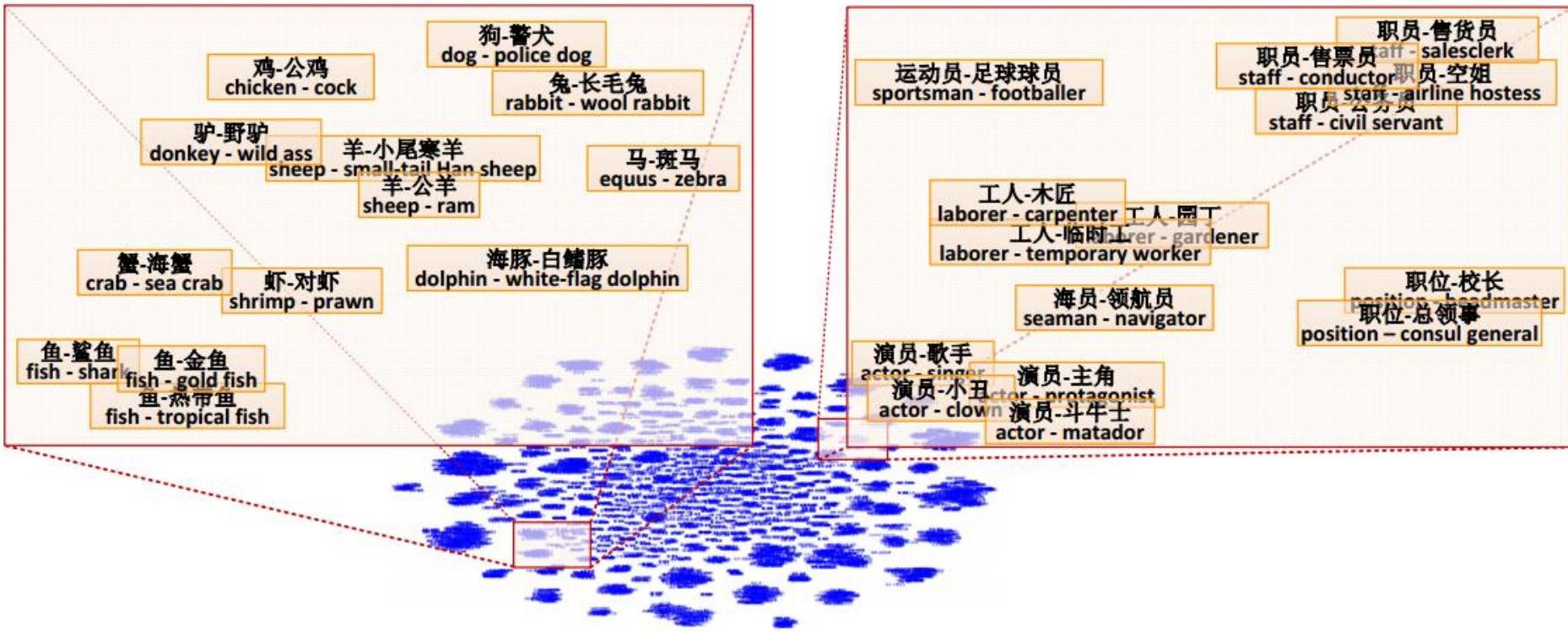
predicting the context given a word

Word Embedding



Source: <http://www.slideshare.net/hustwj/cikm-keynotenov2014>

Word Embedding



Fu, Ruiji, et al. "Learning semantic hierarchies via word embeddings." *Proceedings of the 52th Annual Meeting of the Association for Computational Linguistics: Long Papers*. Vol. 1. 2014.

Word Embedding

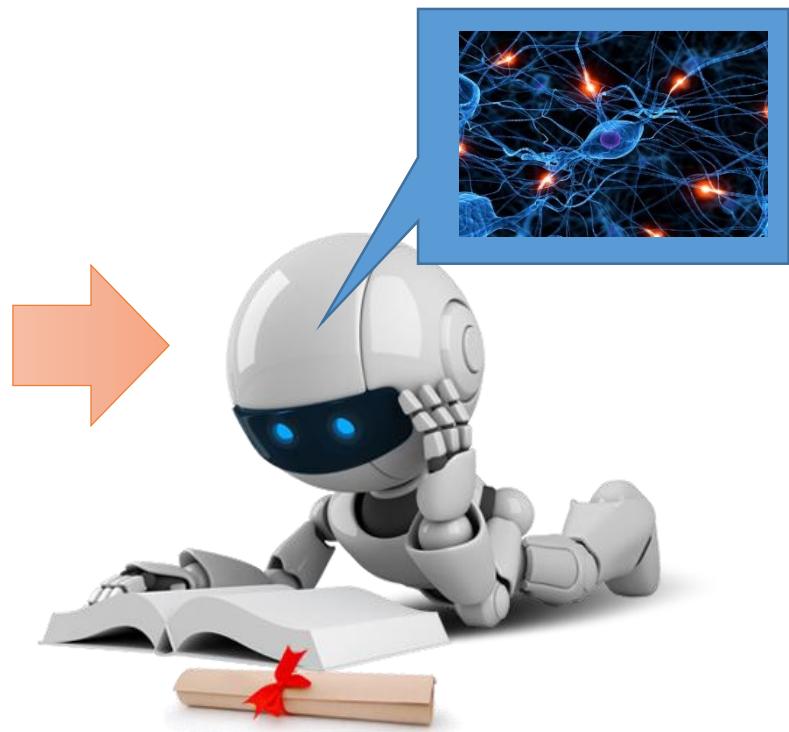
- Characteristics
$$V(Germany) \approx V(Berlin) - V(Rome) + V(Italy)$$
$$V(hotter) - V(hot) \approx V(bigger) - V(big)$$
$$V(Rome) - V(Italy) \approx V(Berlin) - V(Germany)$$
$$V(king) - V(queen) \approx V(uncle) - V(aunt)$$
- Solving analogies

Rome : Italy = Berlin : ?

Compute $V(Berlin) - V(Rome) + V(Italy)$
Find the word w with the closest $V(w)$

Demo

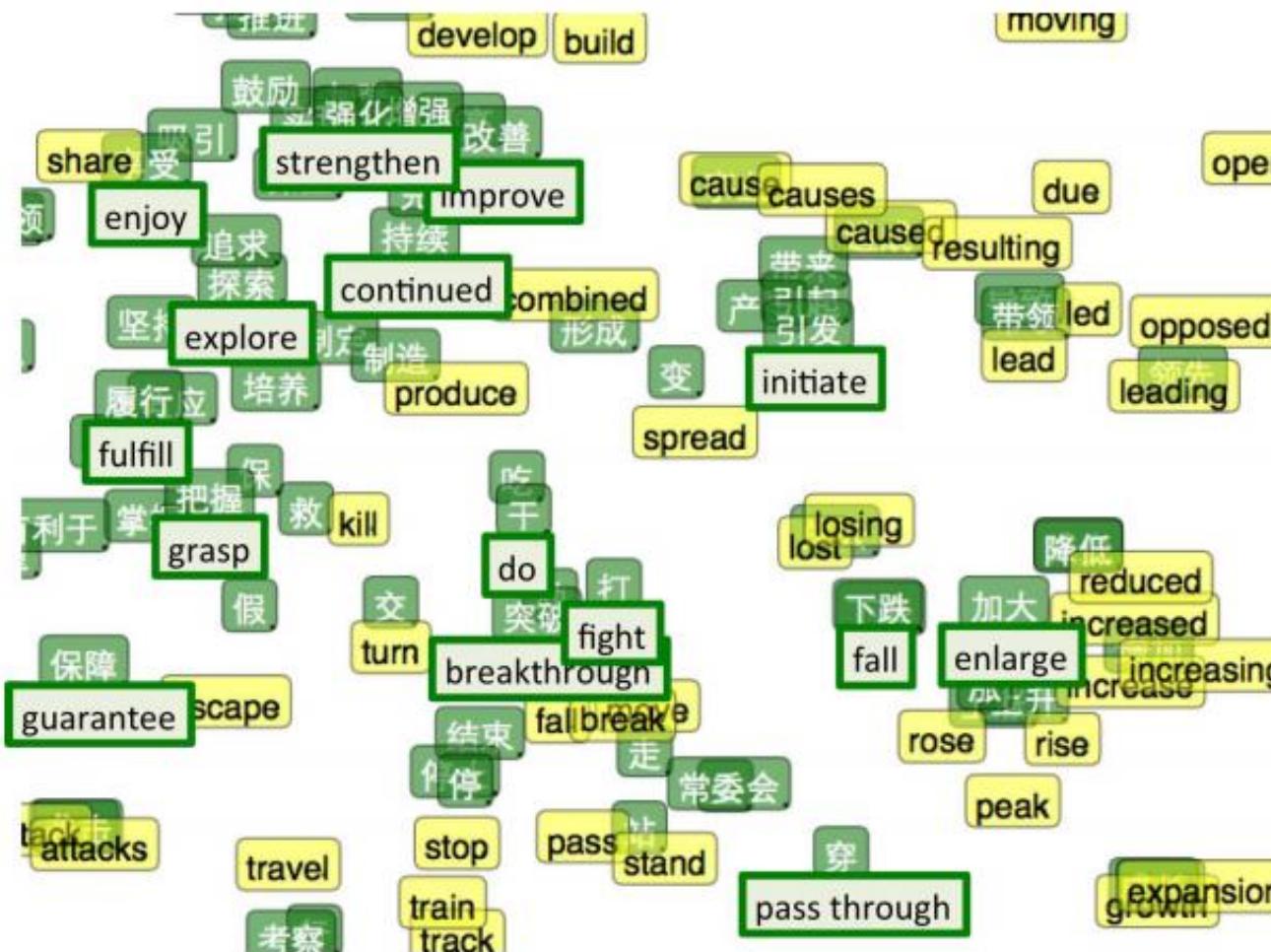
- Machine learns the meaning of words from reading a lot of documents without supervision



Demo

- Model used in demo is provided by 陳仰德
 - Part of the project done by 陳仰德、林資偉
 - TA: 劉元銘
 - Training data is from PTT (collected by 葉青峰)

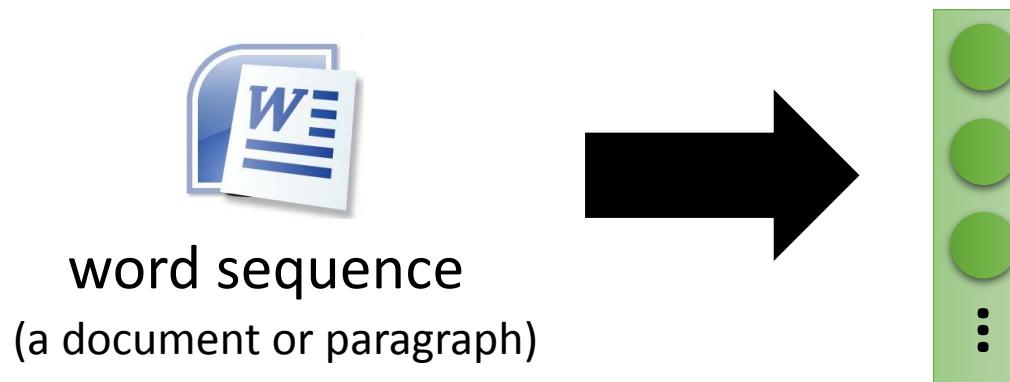
Multi-lingual Embedding



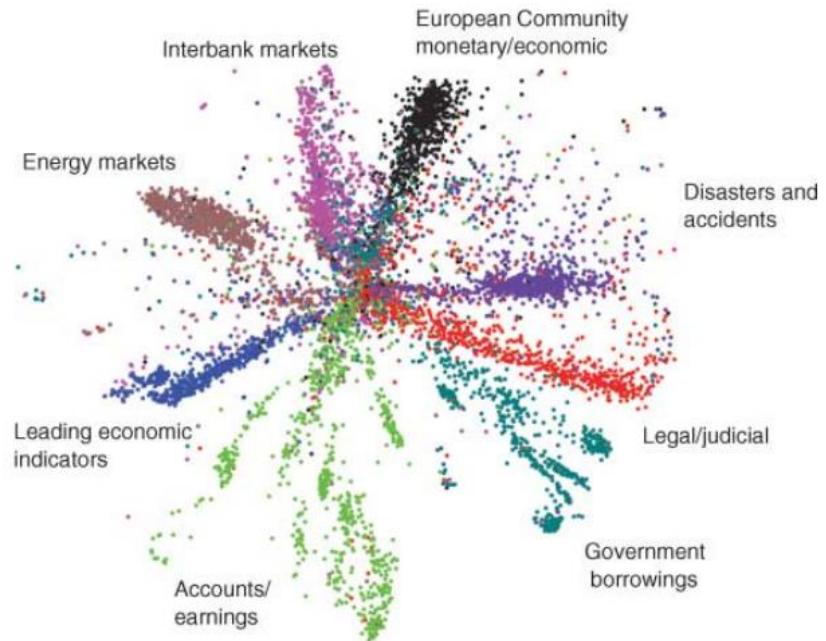
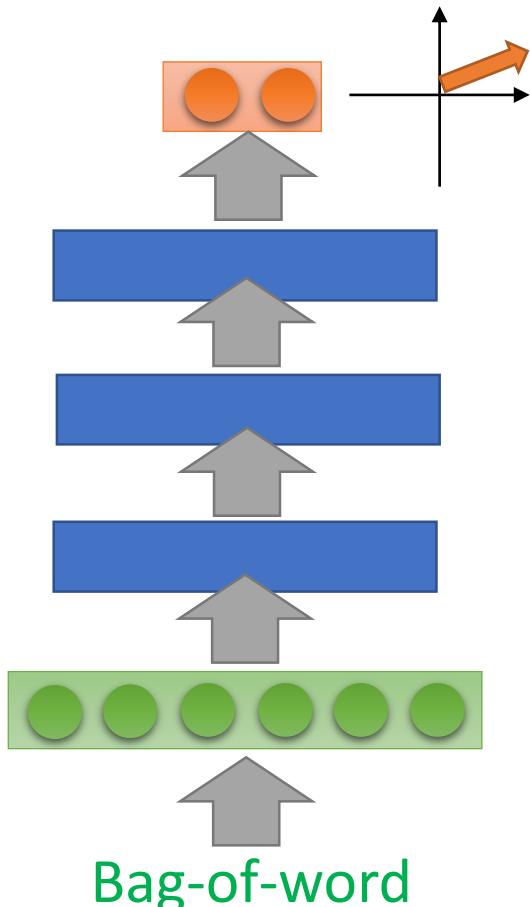
Bilingual Word Embeddings for Phrase-Based Machine Translation, Will Zou, Richard Socher, Daniel Cer and Christopher Manning, EMNLP, 2013

Document Embedding

- word sequences with different lengths → the vector with the same length
 - The vector representing the meaning of the word sequence
 - A word sequence can be a document or a paragraph



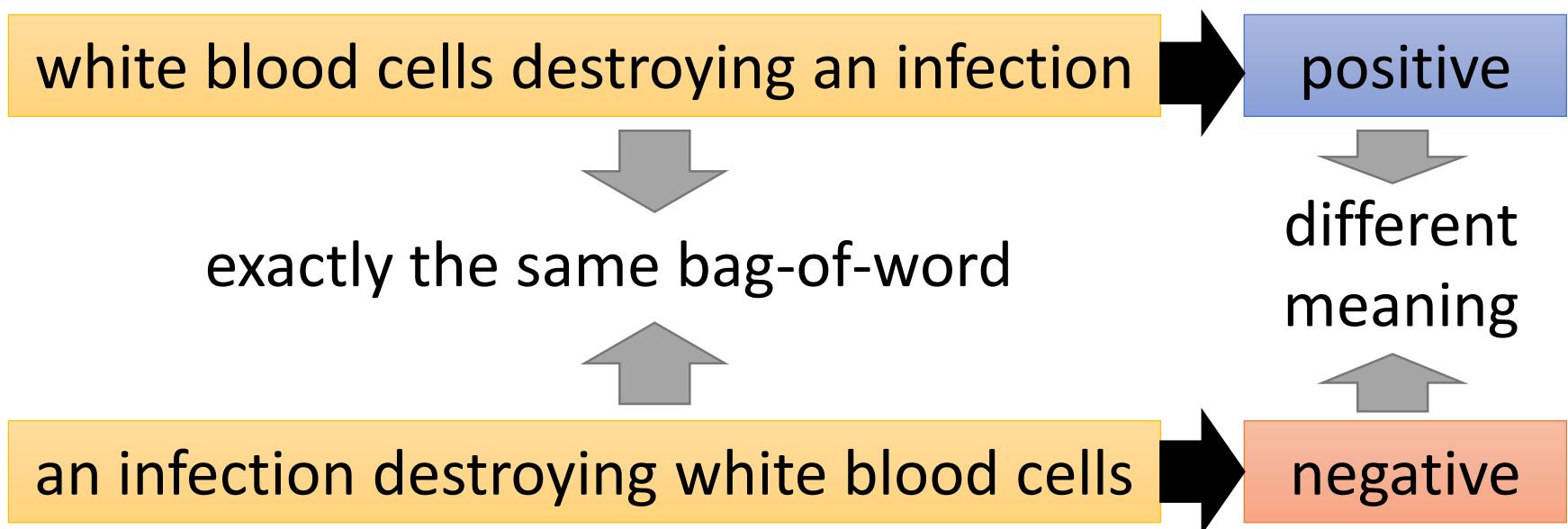
Semantic Embedding



Reference: Hinton, Geoffrey E., and Ruslan R. Salakhutdinov. "Reducing the dimensionality of data with neural networks." *Science* 313.5786 (2006): 504-507

Beyond Bag of Word

- To understand the meaning of a word sequence, the order of the words can not be ignored.

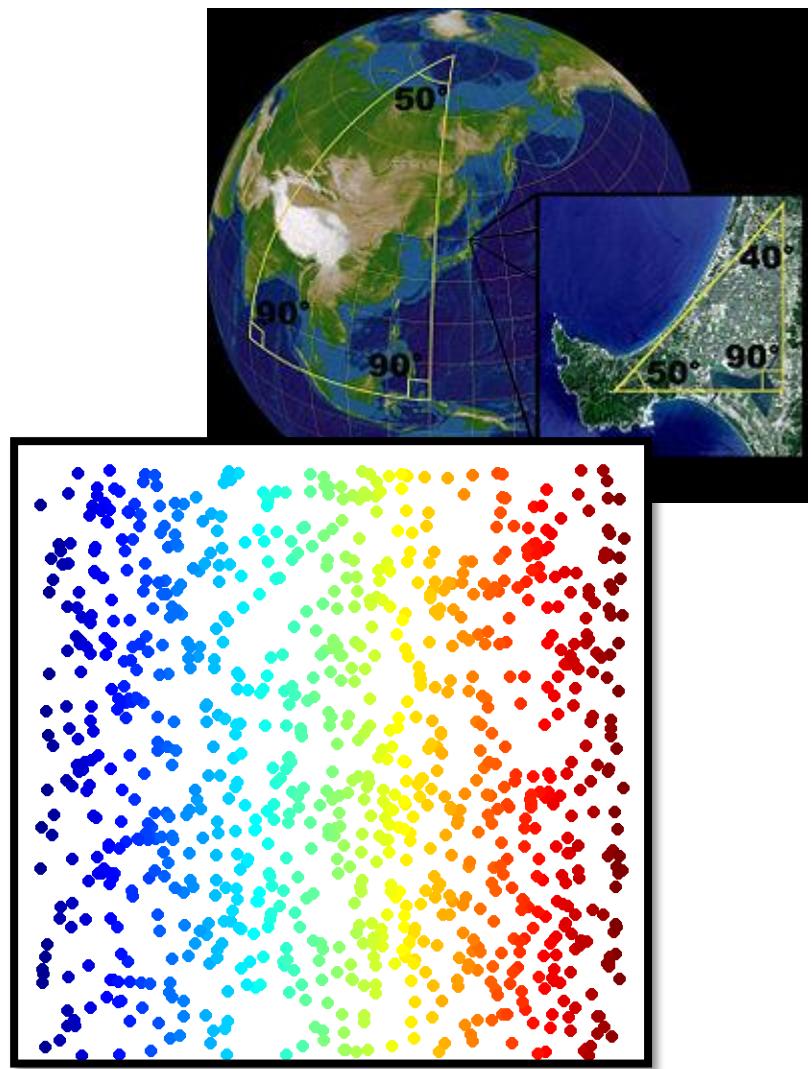
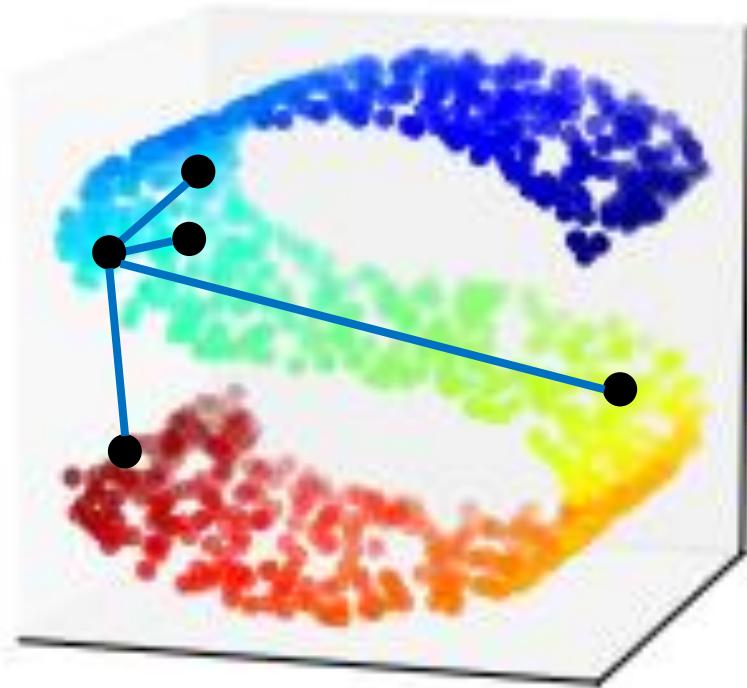


Beyond Bag of Word

- **Paragraph Vector**: Le, Quoc, and Tomas Mikolov. "Distributed Representations of Sentences and Documents." ICML, 2014
- **Seq2seq Auto-encoder**: Li, Jiwei, Minh-Thang Luong, and Dan Jurafsky. "A hierarchical neural autoencoder for paragraphs and documents." arXiv preprint, 2015
- **Skip Thought**: Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Antonio Torralba, Raquel Urtasun, Sanja Fidler, "Skip-Thought Vectors" arXiv preprint, 2015.

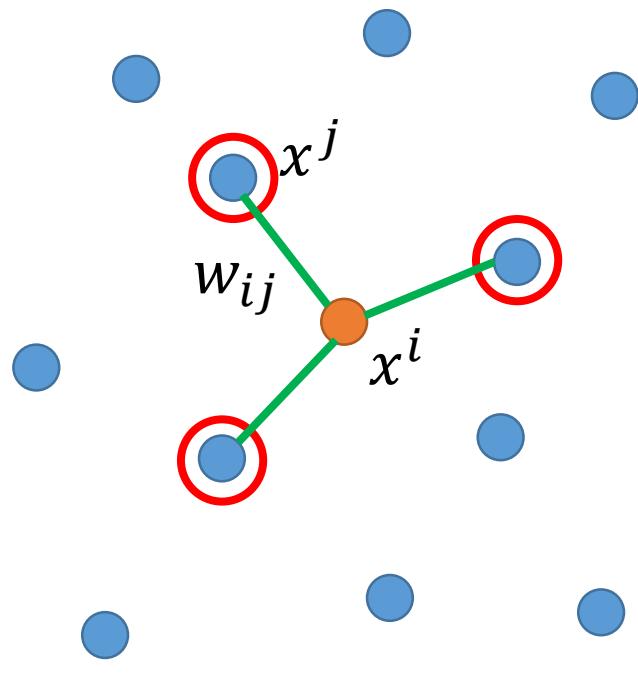
Unsupervised Learning: Neighbor Embedding

Manifold Learning



Suitable for clustering or
following supervised learning

Locally Linear Embedding (LLE)



w_{ij} represents the relation between x^i and x^j

Find a set of w_{ij} minimizing

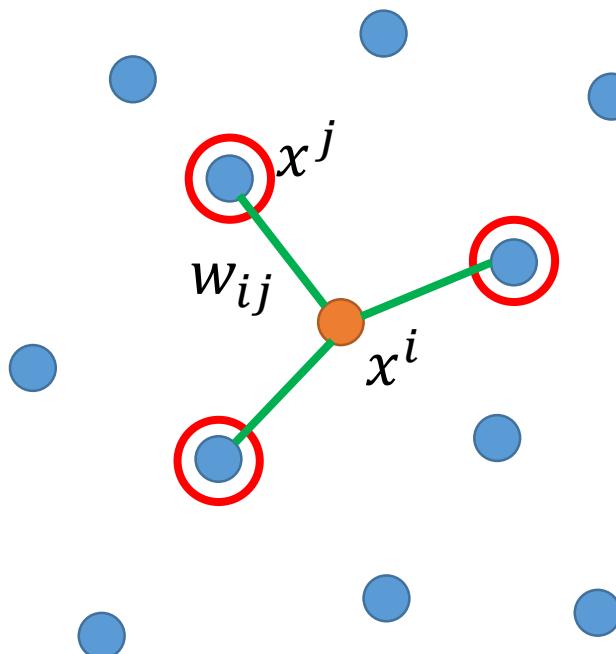
$$\sum_i \left\| x^i - \sum_j w_{ij} x^j \right\|_2$$

Then find the dimension reduction results z^i and z^j based on w_{ij}

LLE

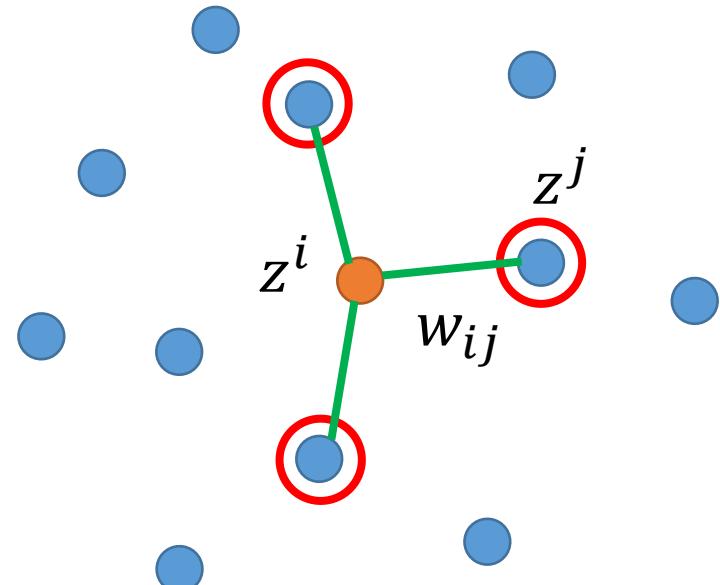
Find a set of z^i minimizing

Keep w_{ij} unchanged



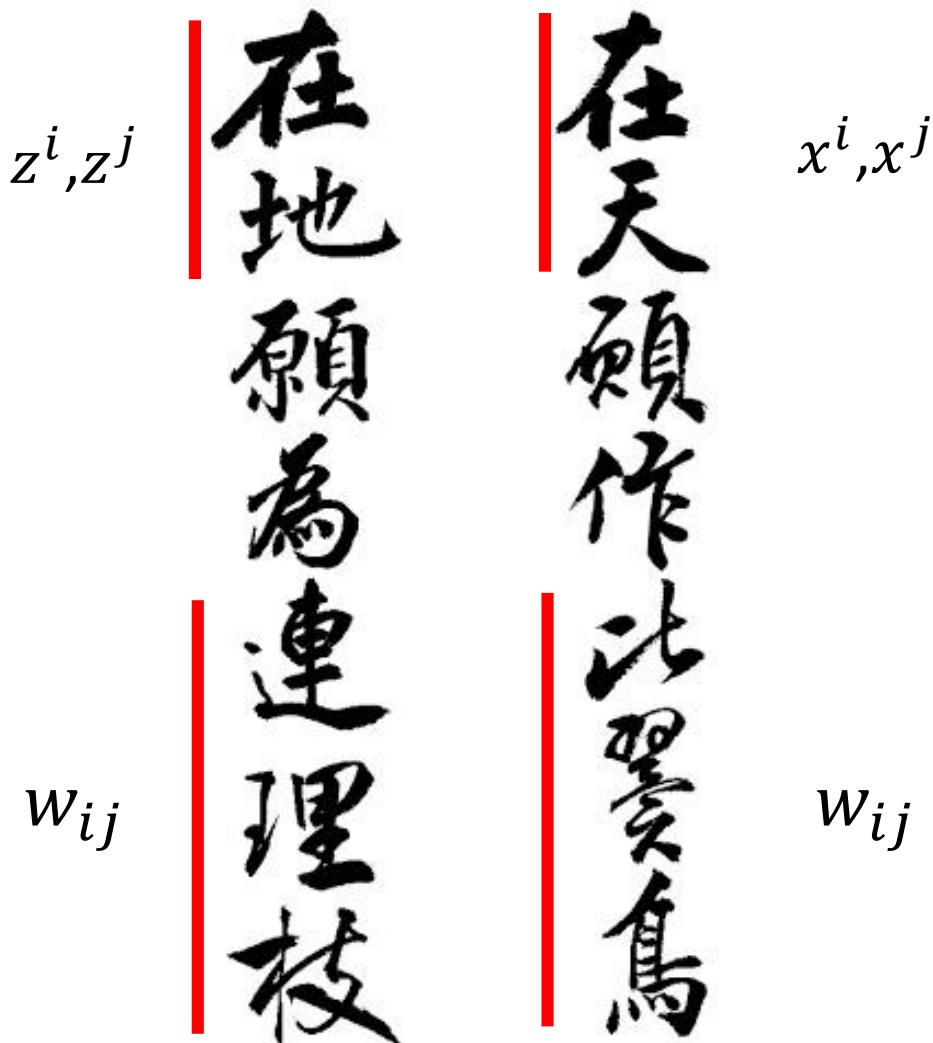
Original Space

$$\sum_i \left\| z^i - \sum_j w_{ij} z^j \right\|_2$$



New (Low-dim) Space

LLE

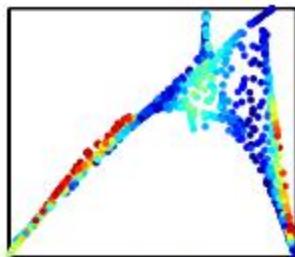


Source of image:

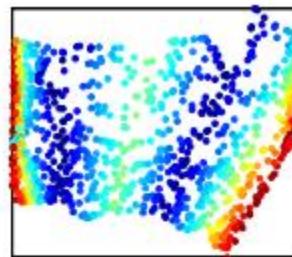
http://feetsprint.blogspot.tw/2016/02/blog-post_29.html

LLE

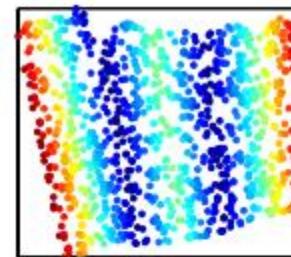
Lawrence K. Saul, Sam T. Roweis, "Think Globally, Fit Locally: Unsupervised Learning of Low Dimensional Manifolds", JMLR, 2013



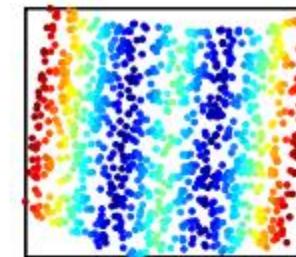
K = 5



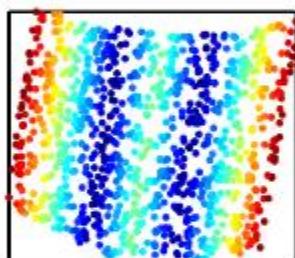
K = 6



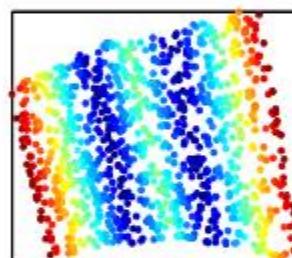
K = 8



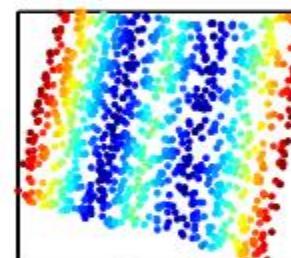
K = 10



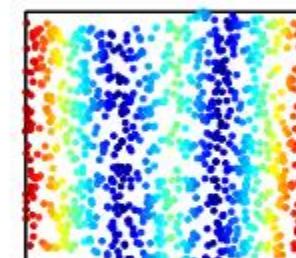
K = 12



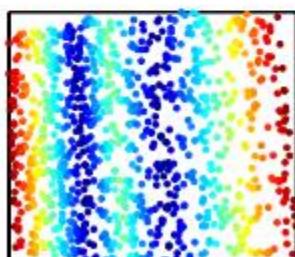
K = 14



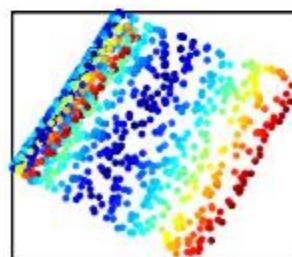
K = 16



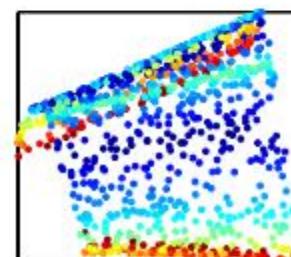
K = 18



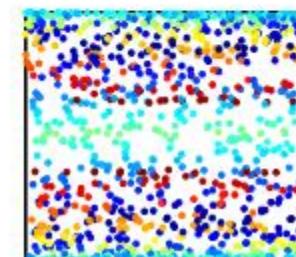
K = 20



K = 30



K = 40



K = 60

Laplacian Eigenmaps

- Graph-based approach

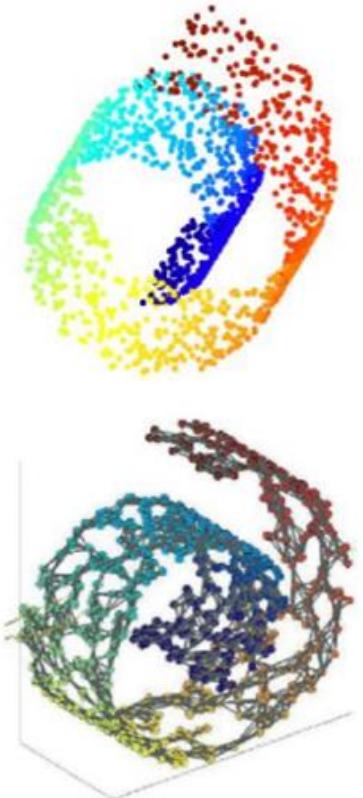
Distance defined by graph approximate the distance on manifold

Construct the data points as a *graph*

Laplacian Eigenmaps

$$w_{i,j} = \begin{cases} \text{similarity} & \text{If connected} \\ 0 & \text{otherwise} \end{cases}$$

- *Review in semi-supervised learning:* If x^1 and x^2 are close in a high density region, \hat{y}^1 and \hat{y}^2 are probably the same.



$$L = \sum_{x^r} C(y^r, \hat{y}^r) + \lambda S$$

As a regularization term

$$S = \frac{1}{2} \sum_{i,j} w_{i,j} (y^i - y^j)^2 = \mathbf{y}^T L \mathbf{y}$$

S evaluates how smooth your label is

L: (R+U) x (R+U) matrix

Graph Laplacian

$$L = D - W$$

Laplacian Eigenmaps

- *Dimension Reduction:* If x^1 and x^2 are close in a high density region, z^1 and z^2 are close to each other.

$$S = \frac{1}{2} \sum_{i,j} w_{i,j} (z^i - z^j)^2$$

Any problem? How about $z^i = z^j = 0$?

Giving some constraints to z :

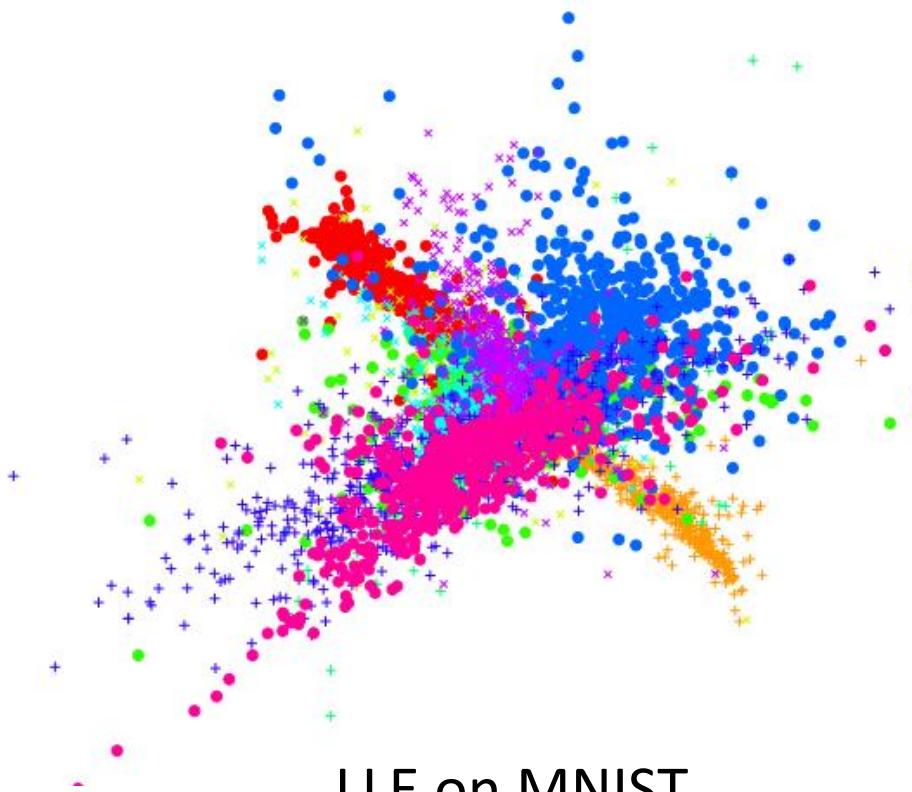
If the dim of z is M , $\text{Span}\{z^1, z^2, \dots, z^N\} = \mathbb{R}^M$

Spectral clustering: clustering on z

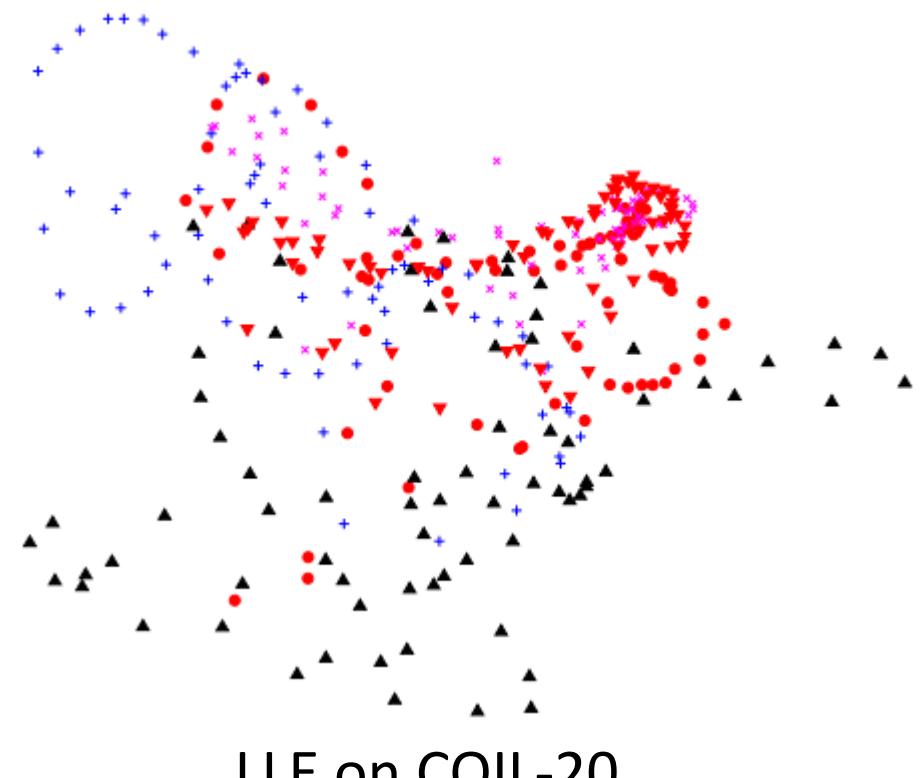
Belkin, M., Niyogi, P. Laplacian eigenmaps and spectral techniques for embedding and clustering. *Advances in neural information processing systems* . 2002

T-distributed Stochastic Neighbor Embedding (t-SNE)

- Problem of the previous approaches
 - Similar data are close, but different data may collapse



LLE on MNIST



LLE on COIL-20

t-SNE



Compute similarity between all pairs of x: $S(x^i, x^j)$

$$P(x^j|x^i) = \frac{S(x^i, x^j)}{\sum_{k \neq i} S(x^i, x^k)}$$

Compute similarity between all pairs of z: $S'(z^i, z^j)$

$$Q(z^j|z^i) = \frac{S'(z^i, z^j)}{\sum_{k \neq i} S'(z^i, z^k)}$$

Find a set of z making the two distributions as close as possible

$$\begin{aligned} L &= \sum_i KL\left(P(*)|x^i)||Q(*)|z^i)\right) \\ &= \sum_i \sum_j P(x^j|x^i) \log \frac{P(x^j|x^i)}{Q(z^j|z^i)} \end{aligned}$$

Ignore σ for simplicity

t-SNE –Similarity Measure

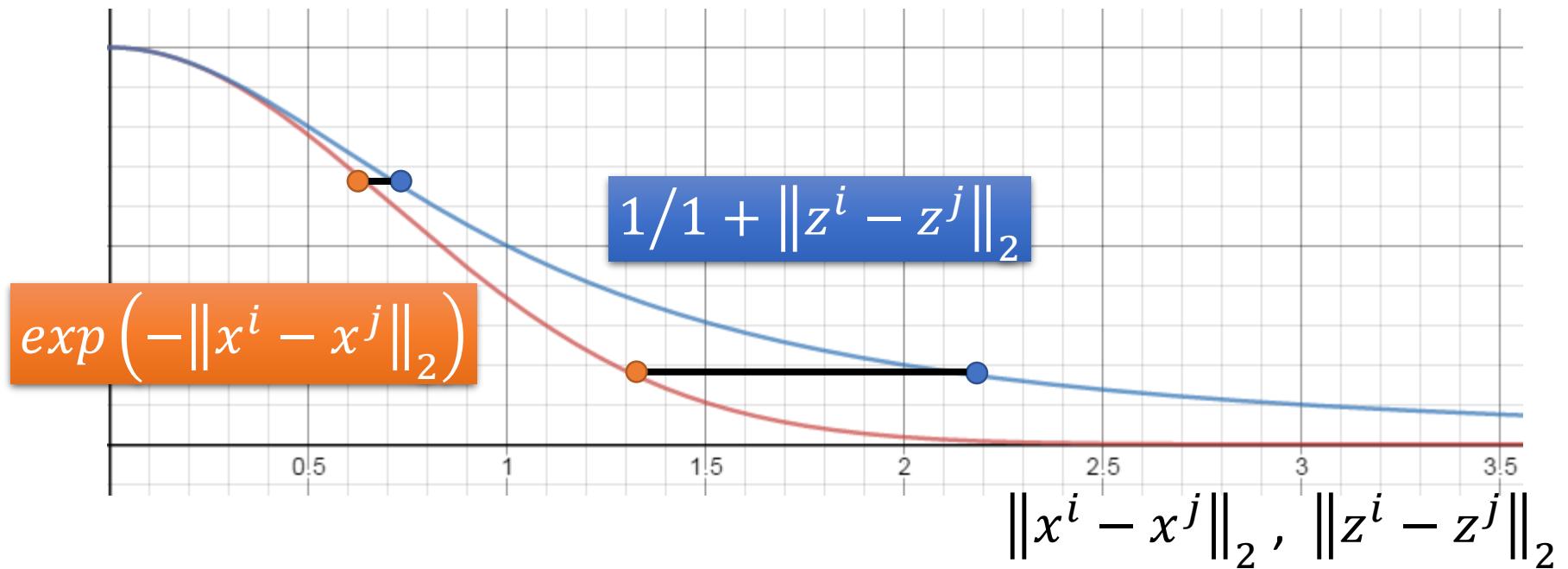
$$S(x^i, x^j) = \exp\left(-\|x^i - x^j\|_2\right)$$

SNE:

$$S'(z^i, z^j) = \exp\left(-\|z^i - z^j\|_2\right)$$

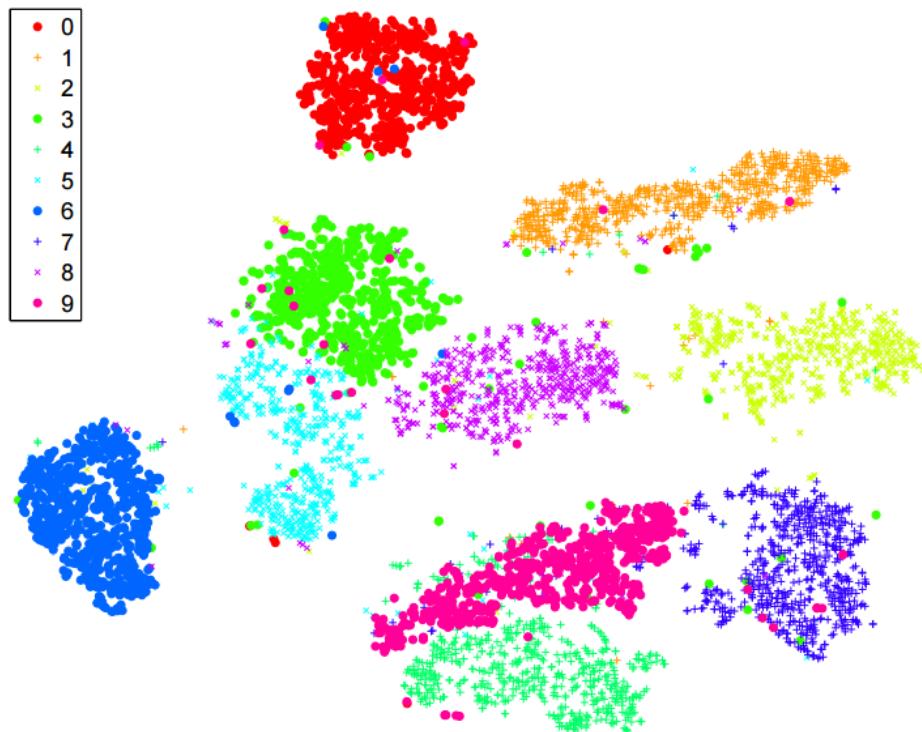
t-SNE:

$$S'(z^i, z^j) = 1 / 1 + \|z^i - z^j\|_2$$



t-SNE

- Good at visualization



t-SNE on MNIST



t-SNE on COIL-20

9

To learn more ...

- Locally Linear Embedding (LLE): [Alpaydin, Chapter 6.11]
- Laplacian Eigenmaps: [Alpaydin, Chapter 6.12]
- t-SNE
 - Laurens van der Maaten, Geoffrey Hinton, “Visualizing Data using t-SNE”, JMLR, 2008
 - Excellent tutorial:
<https://github.com/oreillymedia/t-SNE-tutorial>

Gated RNN & Sequence Generation

Hung-yi Lee

李宏毅

Outline

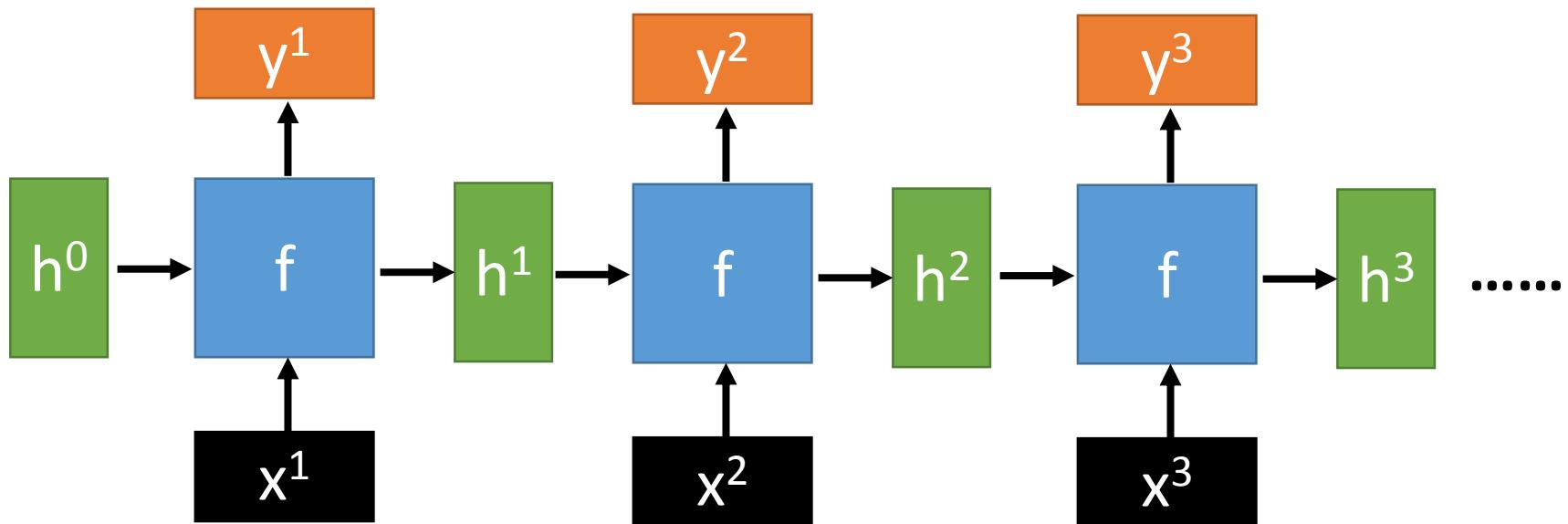
- RNN with Gated Mechanism
- Sequence Generation
- Conditional Sequence Generation
- Tips for Generation

RNN with Gated Mechanism

Recurrent Neural Network

- Given function $f: h', y = f(h, x)$

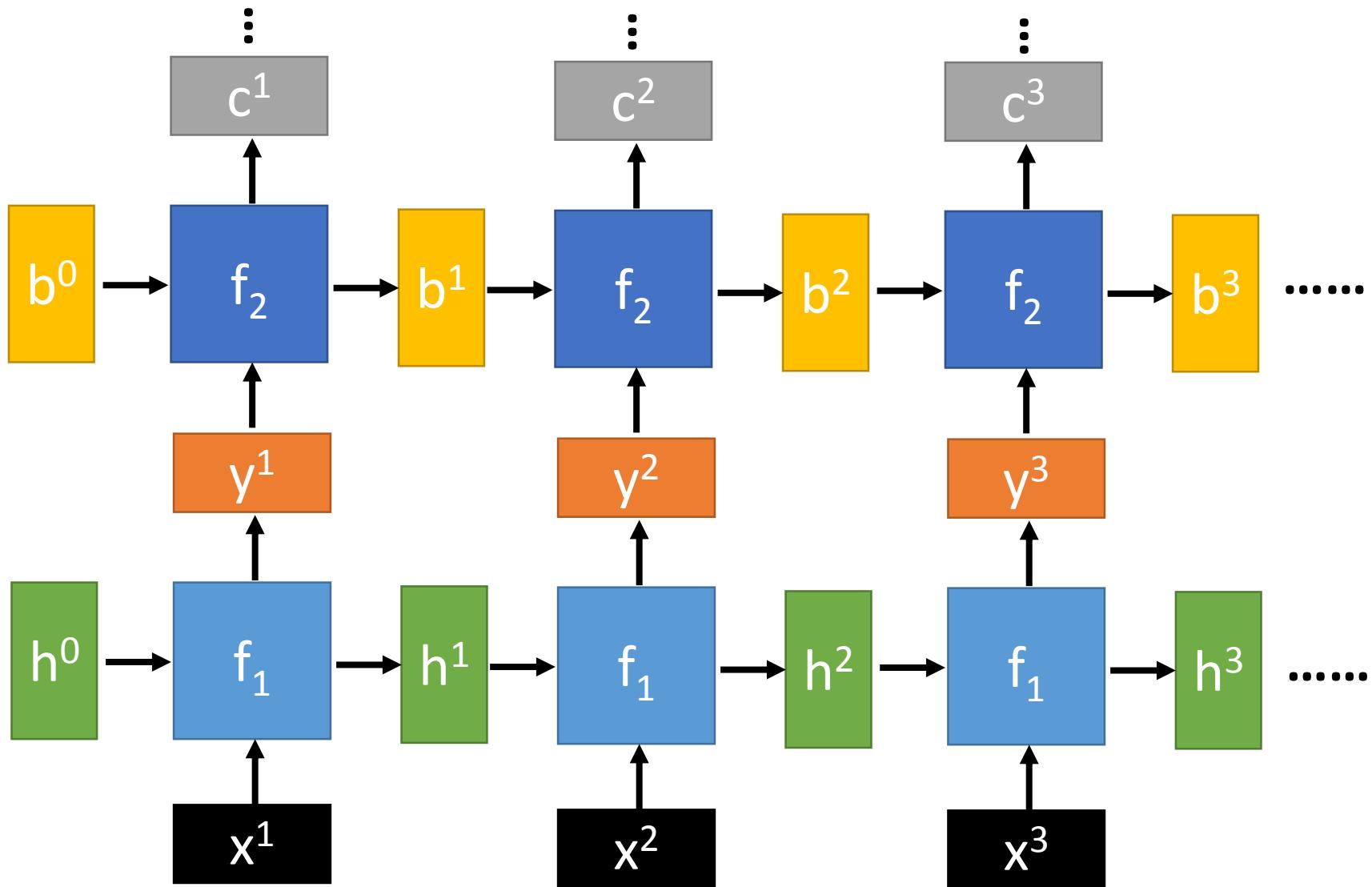
h and h' are vectors with the same dimension



No matter how long the input/output sequence is,
we only need one function f

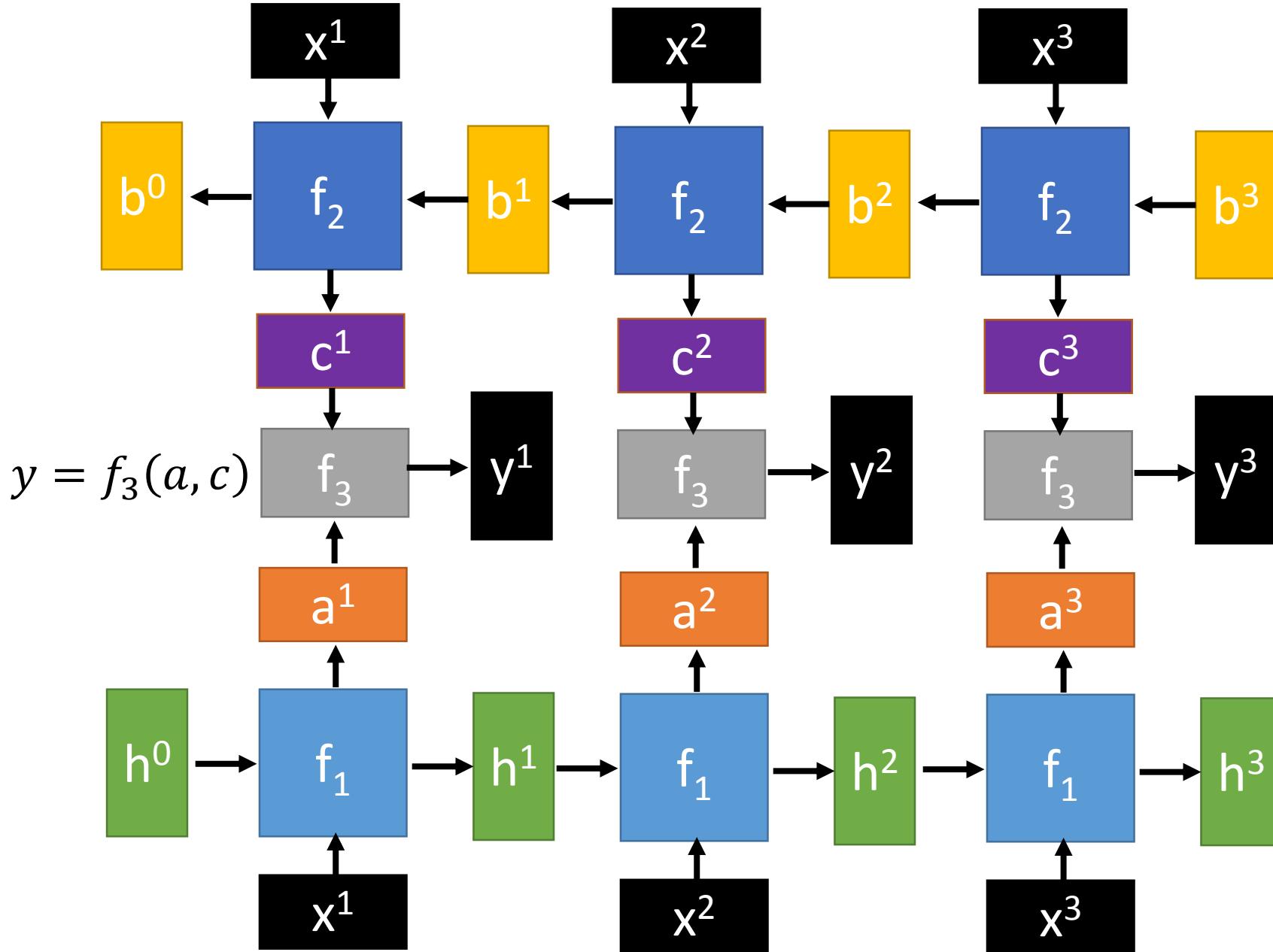
Deep RNN

$$h', y = f_1(h, x) \quad b', c = f_2(b, y) \quad \dots$$



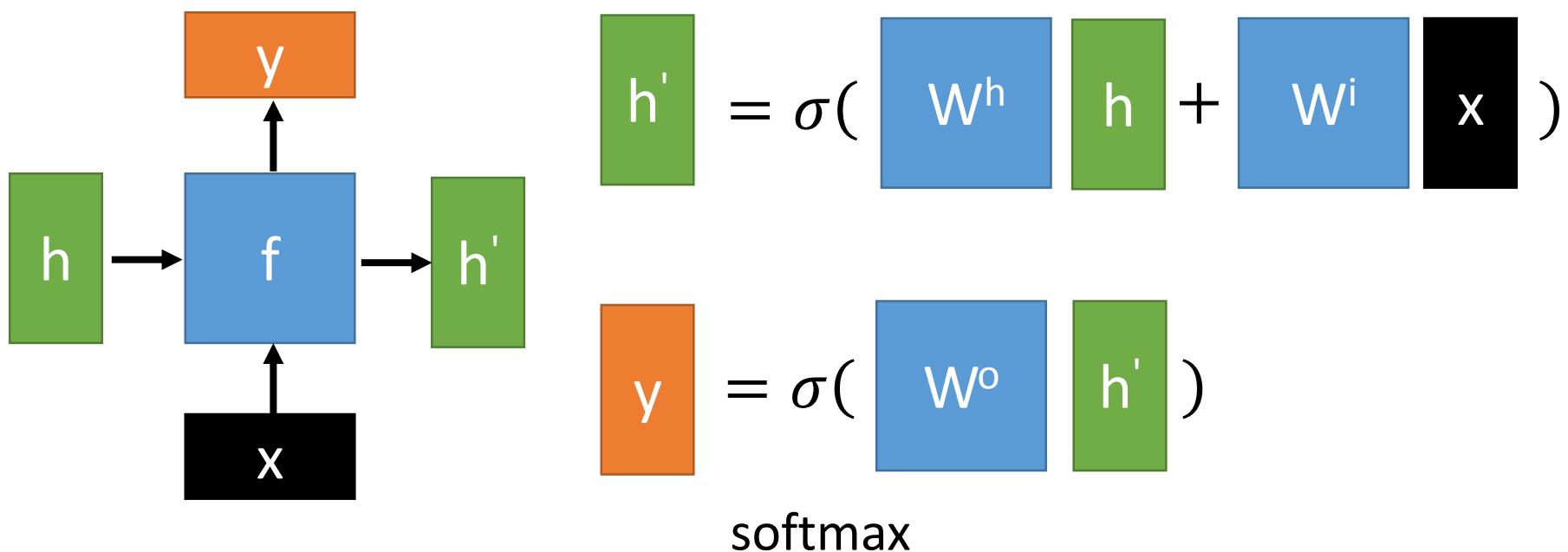
Bidirectional RNN

$$h', a = f_1(h, x) \quad b', c = f_2(b, x)$$



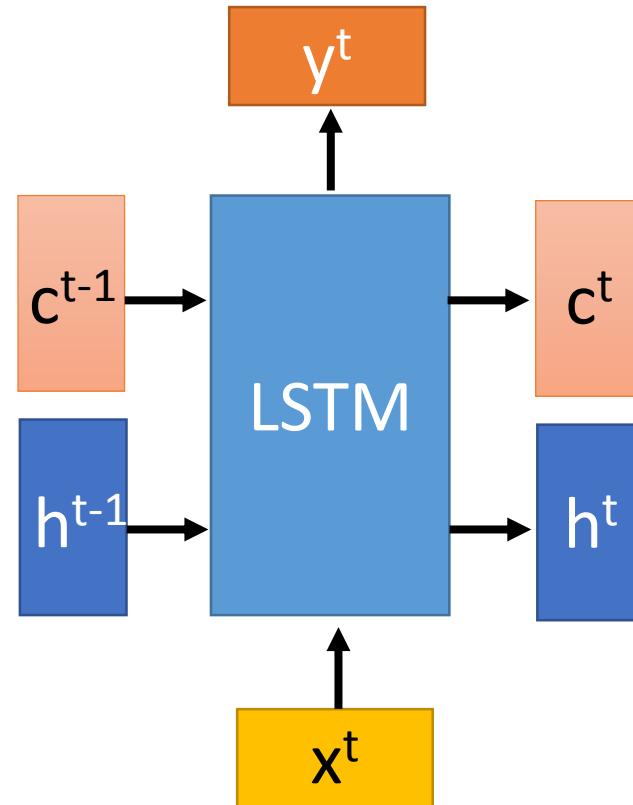
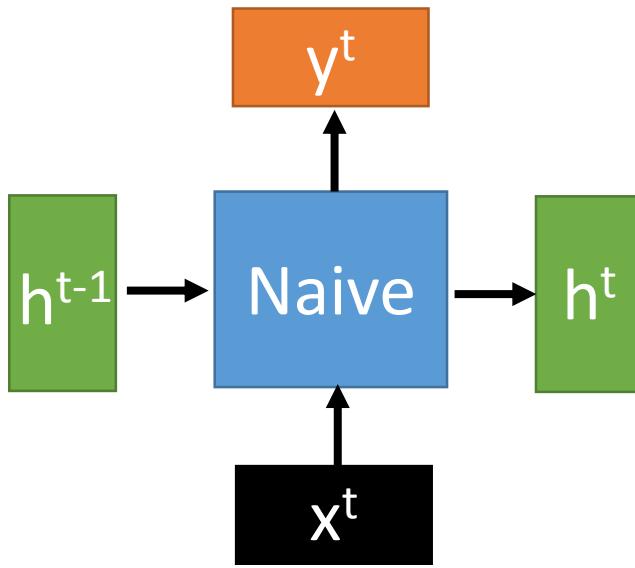
Naïve RNN

- Given function $f: h', y = f(h, x)$



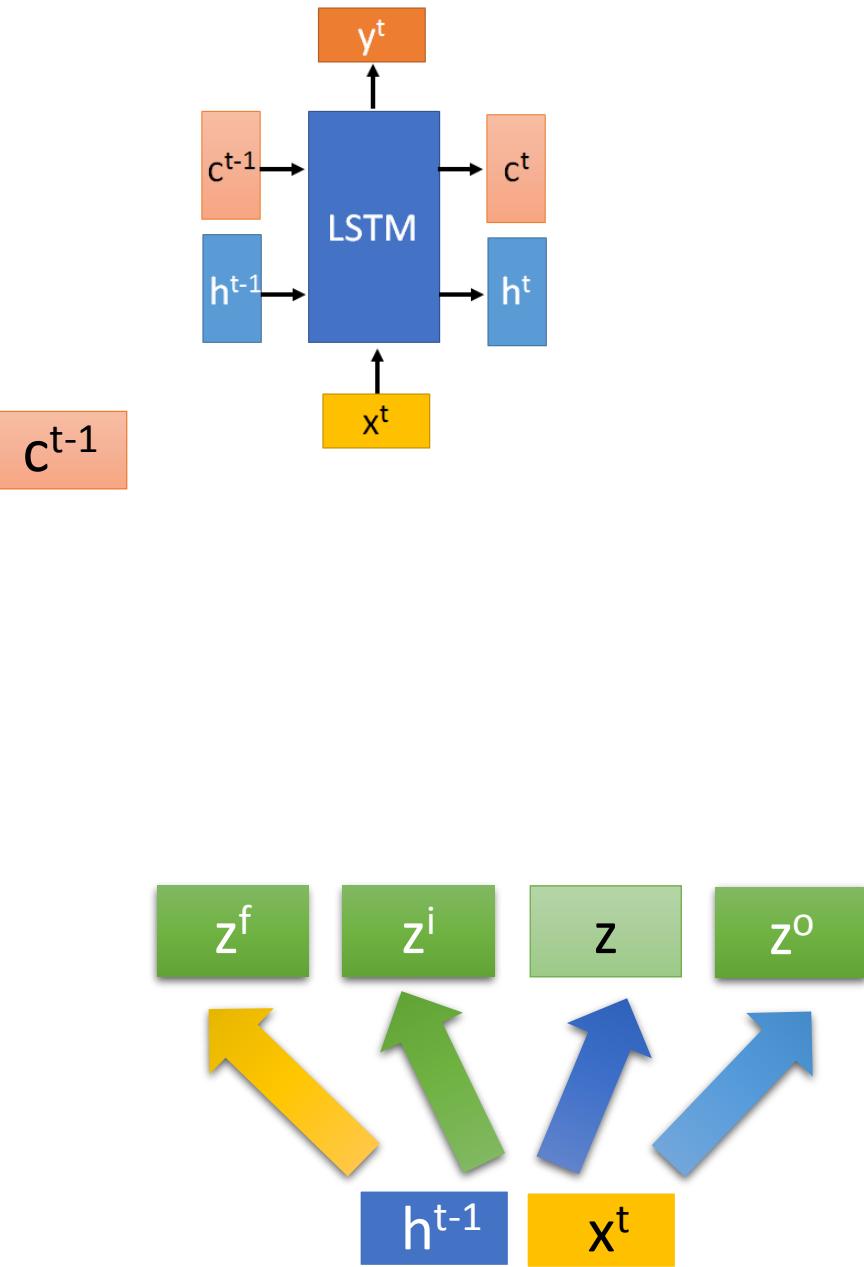
Ignore bias here

LSTM



c changes slowly $\rightarrow c^t$ is c^{t-1} added by something

h changes faster $\rightarrow h^t$ and h^{t-1} can be very different



$$z = \tanh(W h^{t-1} + x^t)$$

$$z^i = \sigma(W^i h^{t-1} + x^t)$$

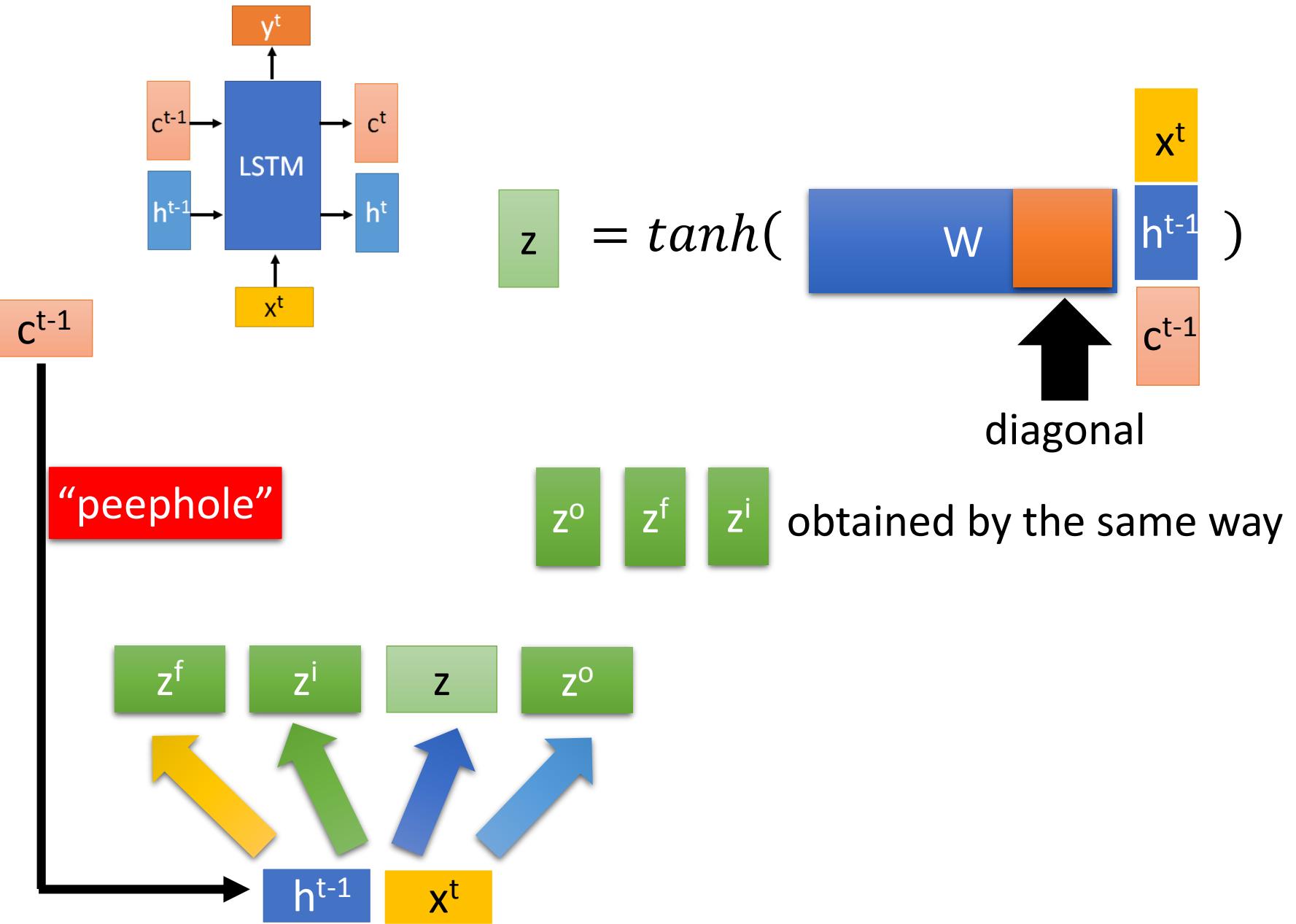
Input gate

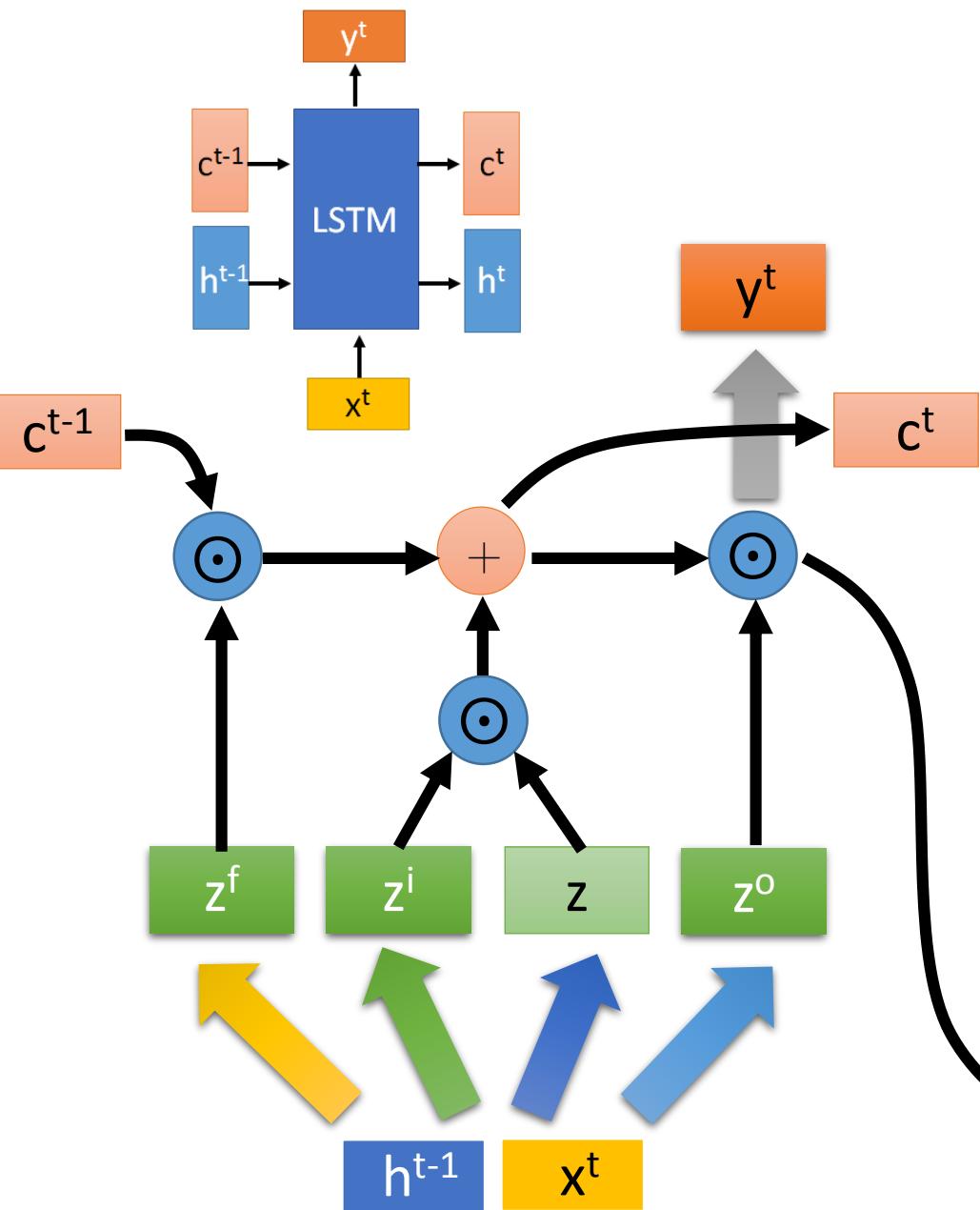
$$z^f = \sigma(W^f h^{t-1} + x^t)$$

forget gate

$$z^o = \sigma(W^o h^{t-1} + x^t)$$

output gate





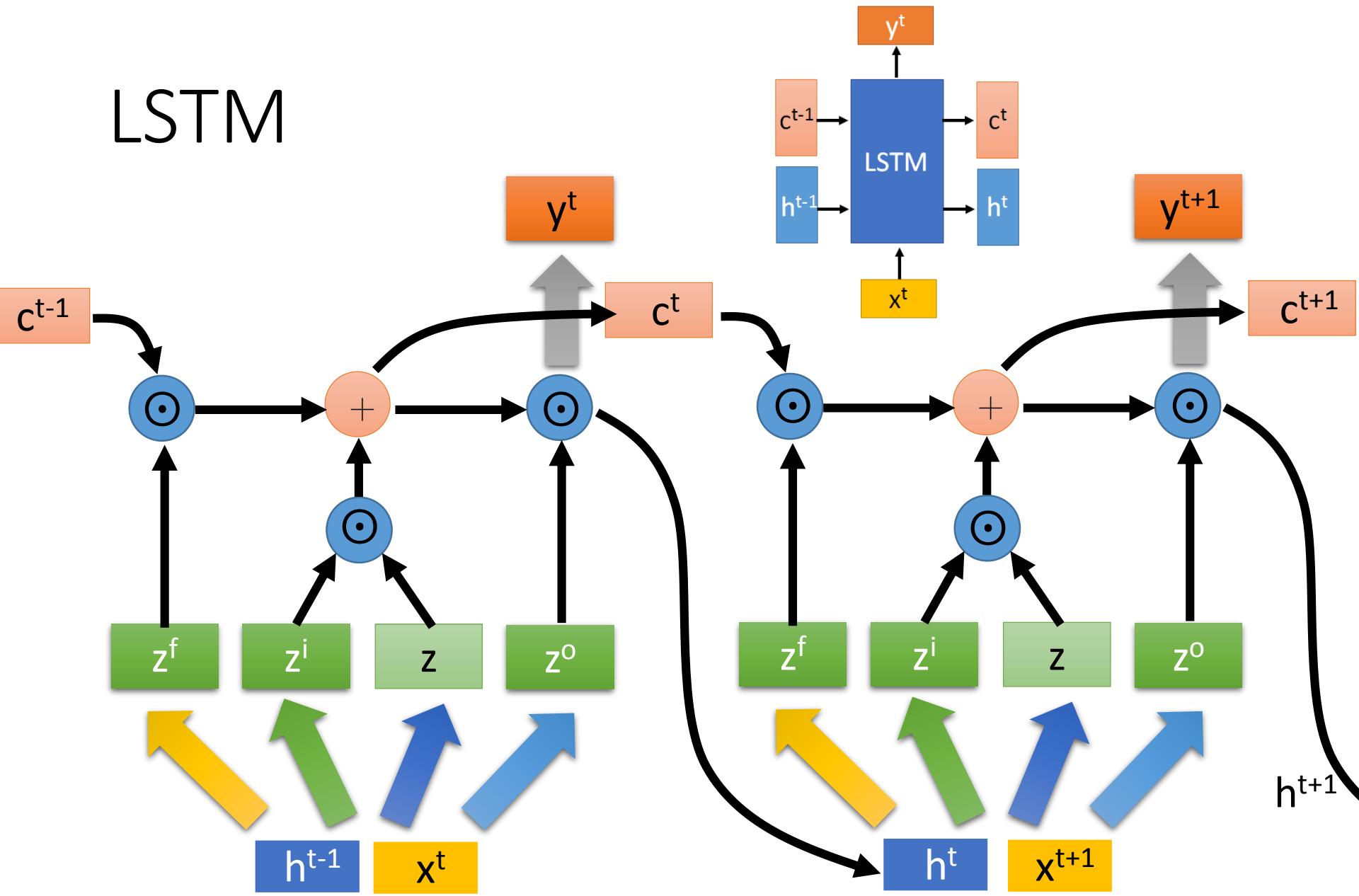
$$c^t = z^f \odot c^{t-1} + z^i \odot z$$

$$h^t = z^o \odot \tanh(c^t)$$

$$y^t = \sigma(W' h^t)$$

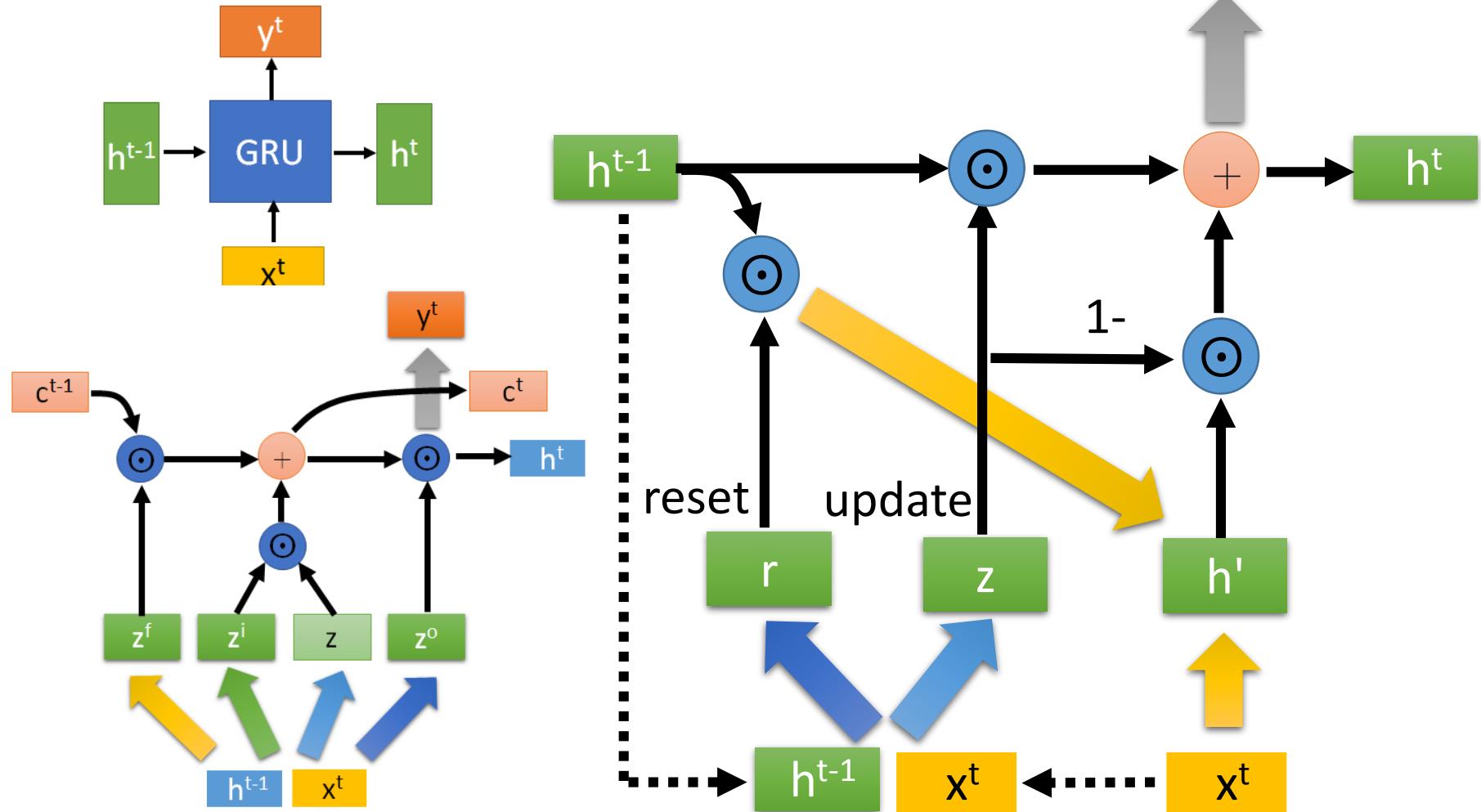
h^t

LSTM

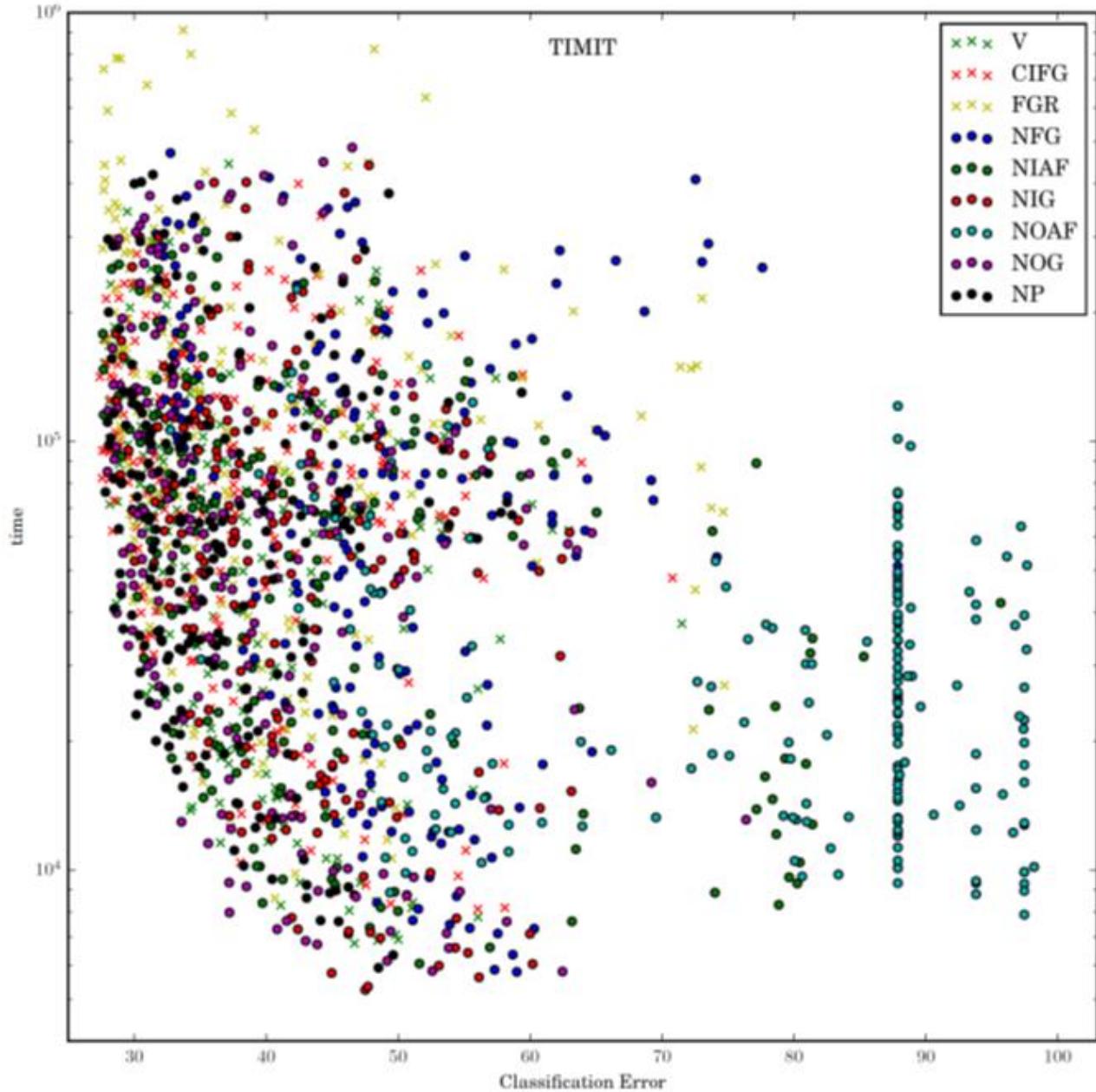


GRU

$$h^t = z \odot h^{t-1} + (1 - z) \odot h'$$

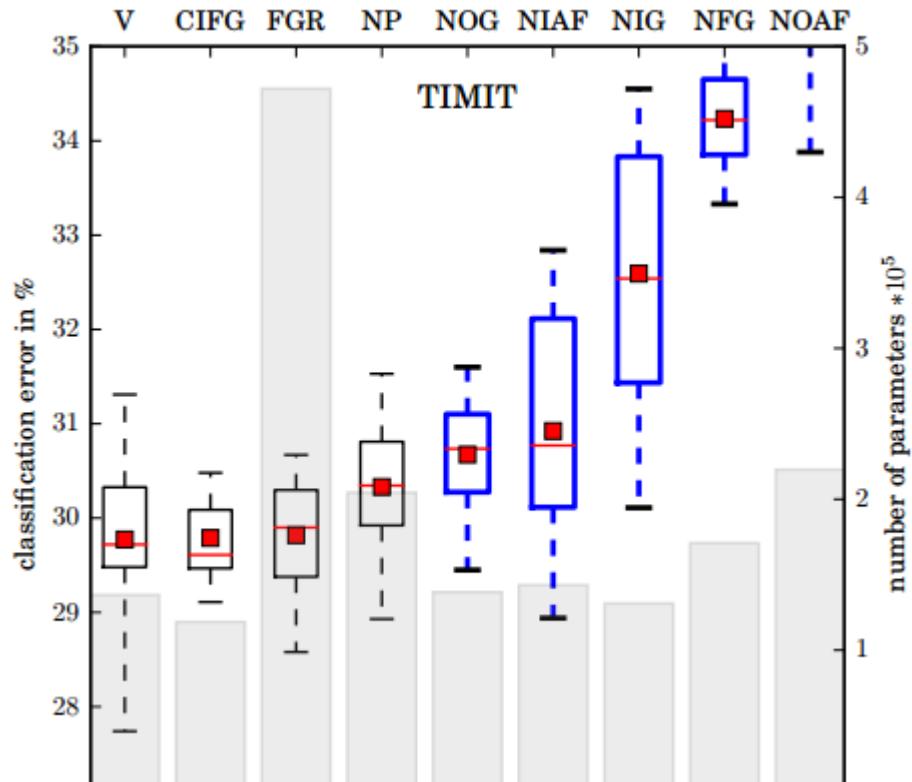


LSTM: A Search Space Odyssey



LSTM: A Search Space Odyssey

1. No Input Gate (NIG)
2. No Forget Gate (NFG)
3. No Output Gate (NOG)
4. No Input Activation Function (NIAF)
5. No Output Activation Function (NOAF)
6. No Peepholes (NP)
7. Coupled Input and Forget Gate (CIFG)
8. Full Gate Recurrence (FGR)



Standard LSTM works well

Simply LSTM: coupling input and forget gate, removing peephole

Forget gate is critical for performance

Output gate activation function is critical

An Empirical Exploration of Recurrent Network Architectures

Arch.	Arith.	XML	PTB
Tanh	0.29493	0.32050	0.08782
LSTM	0.89228	0.42470	0.08912
LSTM-f	0.29292	0.23356	0.08808
LSTM-i	0.75109	0.41371	0.08662
LSTM-o	0.86747	0.42117	0.08933
LSTM-b	0.90163	0.44434	0.08952
GRU	0.89565	0.45963	0.09069
MUT1	0.92135	0.47483	0.08968
MUT2	0.89735	0.47324	0.09036
MUT3	0.90728	0.46478	0.09161

LSTM-f/i/o: removing
forget/input/output gates

LSTM-b: large bias

Importance: forget > input > output

Large bias for forget gate is helpful

An Empirical Exploration of Recurrent Network Architectures

MUT1:

$$\begin{aligned} z &= \text{sigm}(W_{xz}x_t + b_z) \\ r &= \text{sigm}(W_{xr}x_t + W_{hr}h_t + b_r) \\ h_{t+1} &= \tanh(W_{hh}(r \odot h_t) + \tanh(x_t) + b_h) \odot z \\ &\quad + h_t \odot (1 - z) \end{aligned}$$

MUT2:

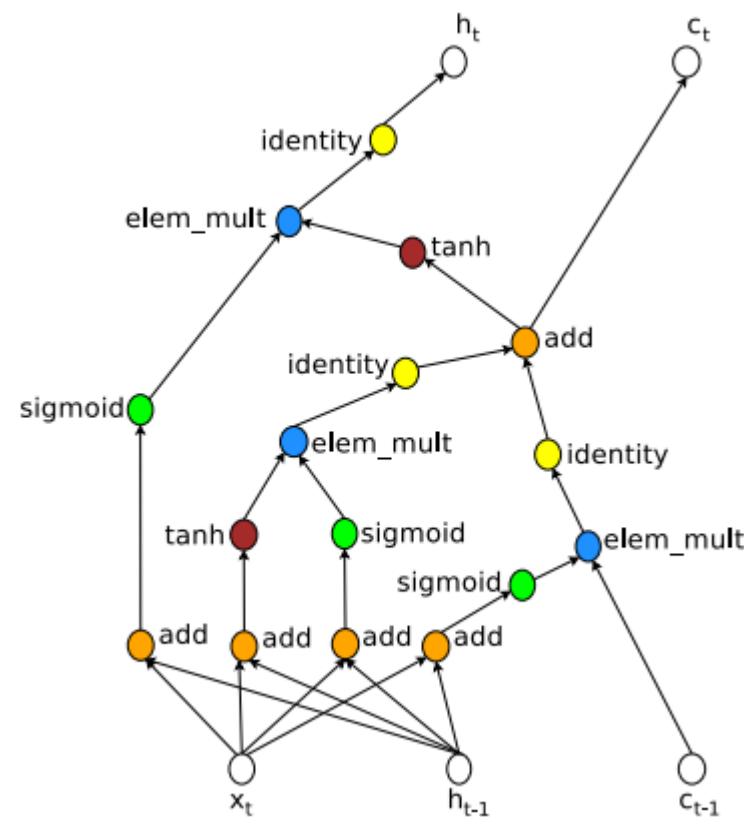
$$\begin{aligned} z &= \text{sigm}(W_{xz}x_t + W_{hz}h_t + b_z) \\ r &= \text{sigm}(x_t + W_{hr}h_t + b_r) \\ h_{t+1} &= \tanh(W_{hh}(r \odot h_t) + W_{xh}x_t + b_h) \odot z \\ &\quad + h_t \odot (1 - z) \end{aligned}$$

MUT3:

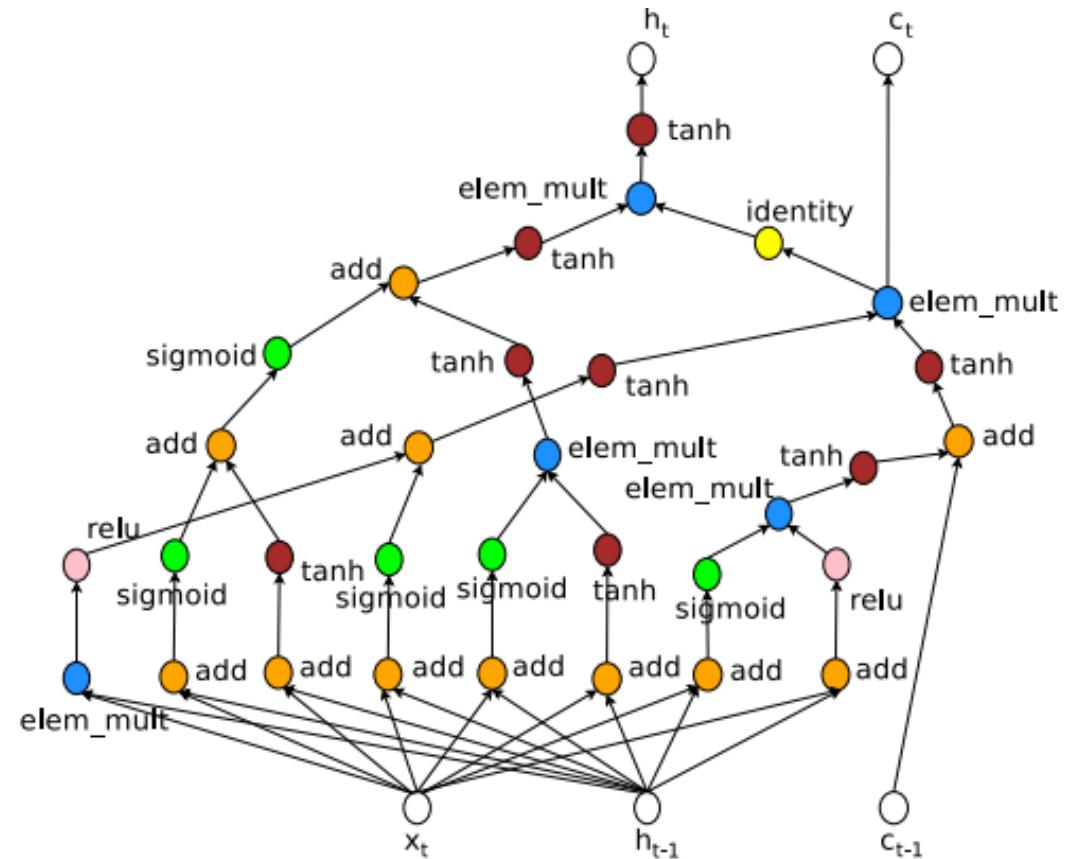
$$\begin{aligned} z &= \text{sigm}(W_{xz}x_t + W_{hz} \tanh(h_t) + b_z) \\ r &= \text{sigm}(W_{xr}x_t + W_{hr}h_t + b_r) \\ h_{t+1} &= \tanh(W_{hh}(r \odot h_t) + W_{xh}x_t + b_h) \odot z \\ &\quad + h_t \odot (1 - z) \end{aligned}$$

Neural Architecture Search with Reinforcement Learning

LSTM

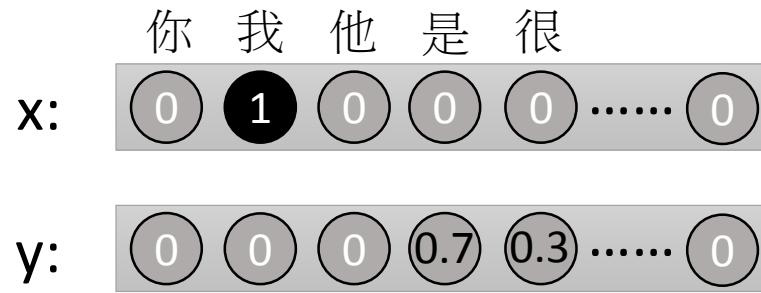


From Reinforcement Learning

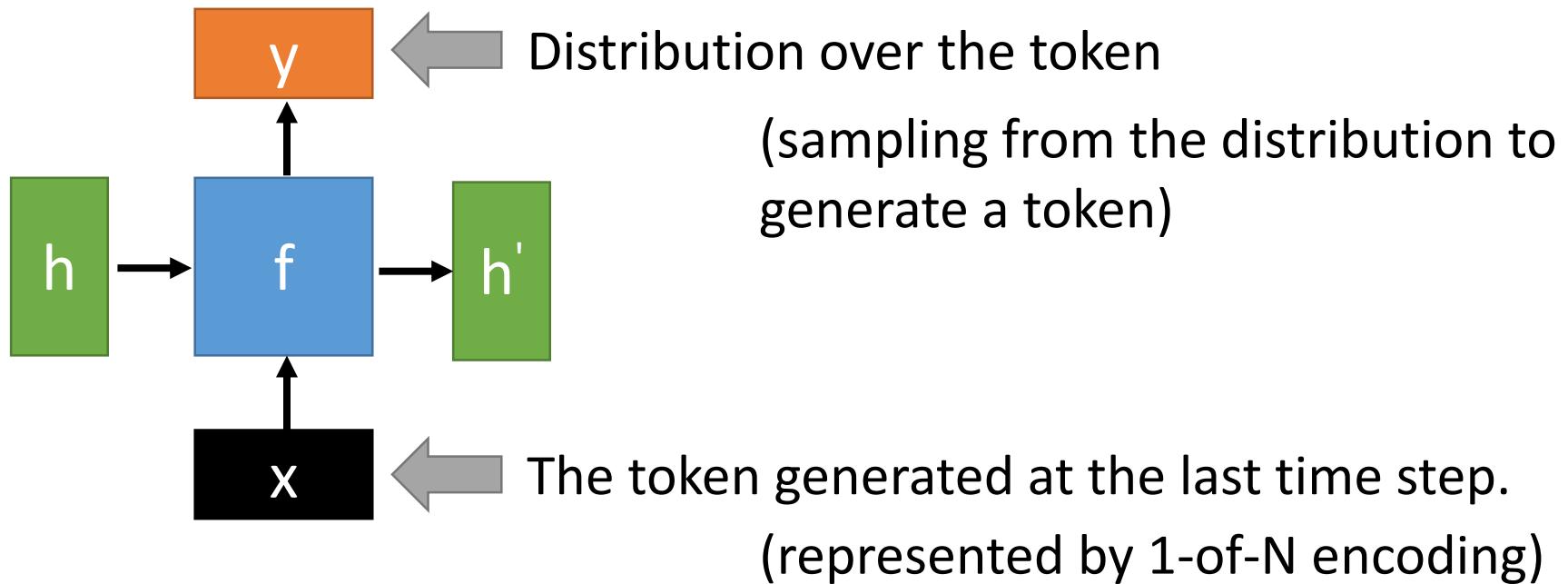


Sequence Generation

Generation



- Sentences are composed of characters/words
- Generating a character/word at each time by RNN



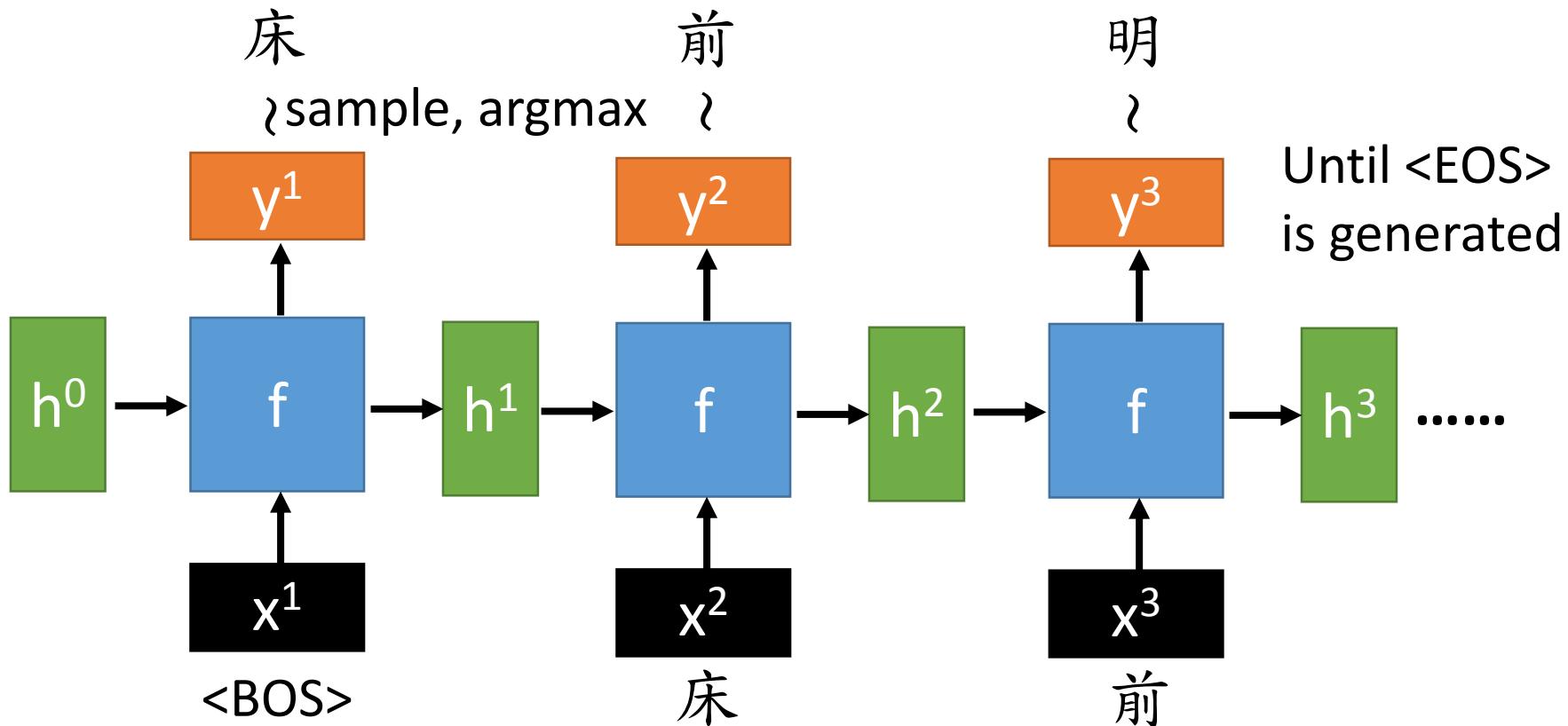
Generation

$y^1: P(w | \text{<BOS>})$

$y^2: P(w | \text{<BOS>, 床})$

$y^3: P(w | \text{<BOS>, 床, 前})$

- Sentences are composed of characters/words
- Generating a character/word at each time by RNN



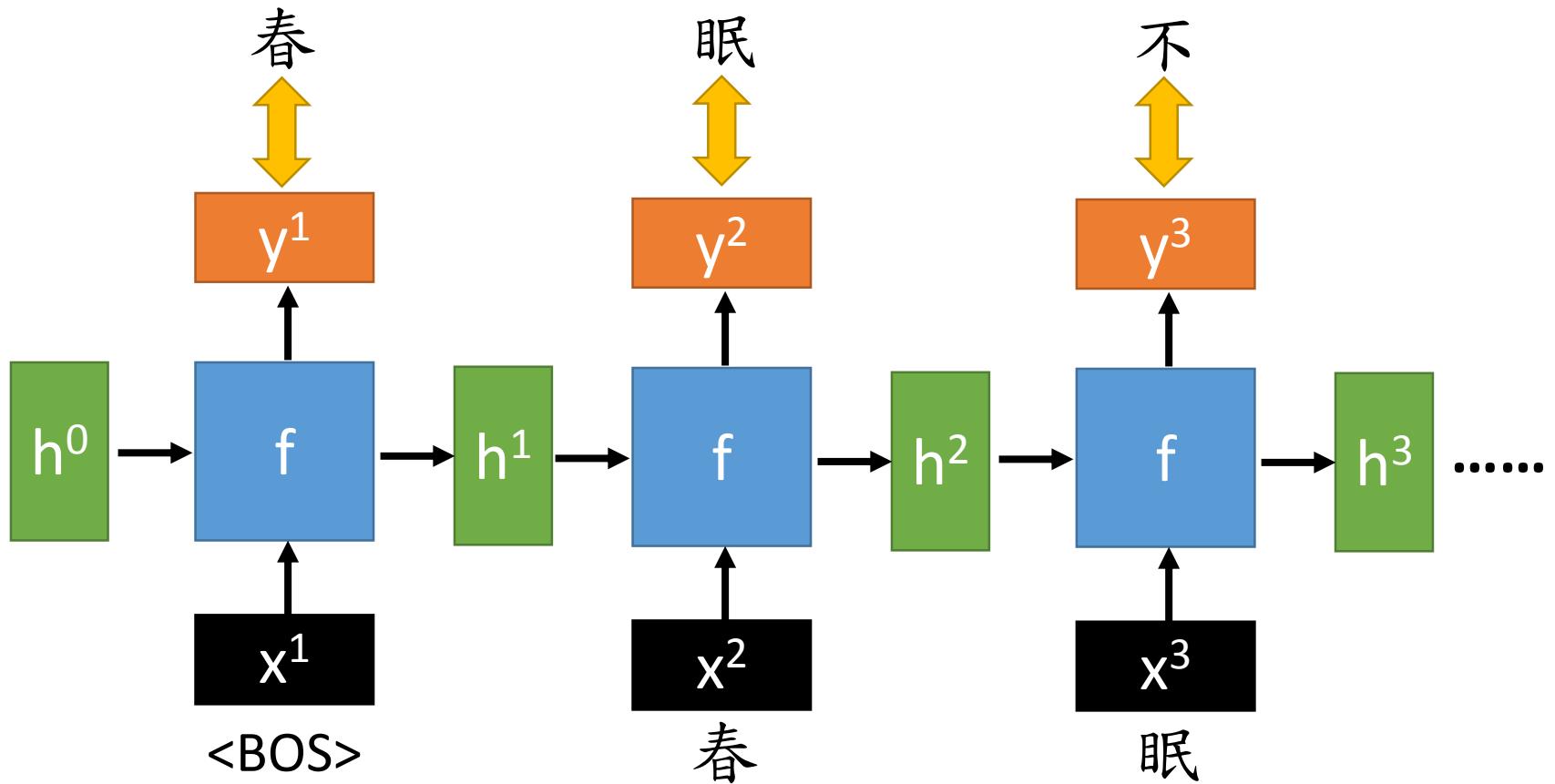
Generation



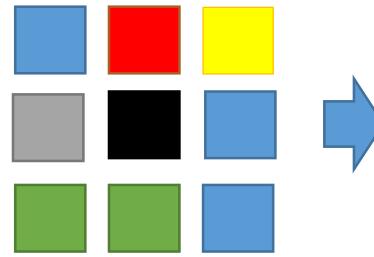
: minimizing cross-entropy

- Training

Training data: 春 眠 不 覺 曉



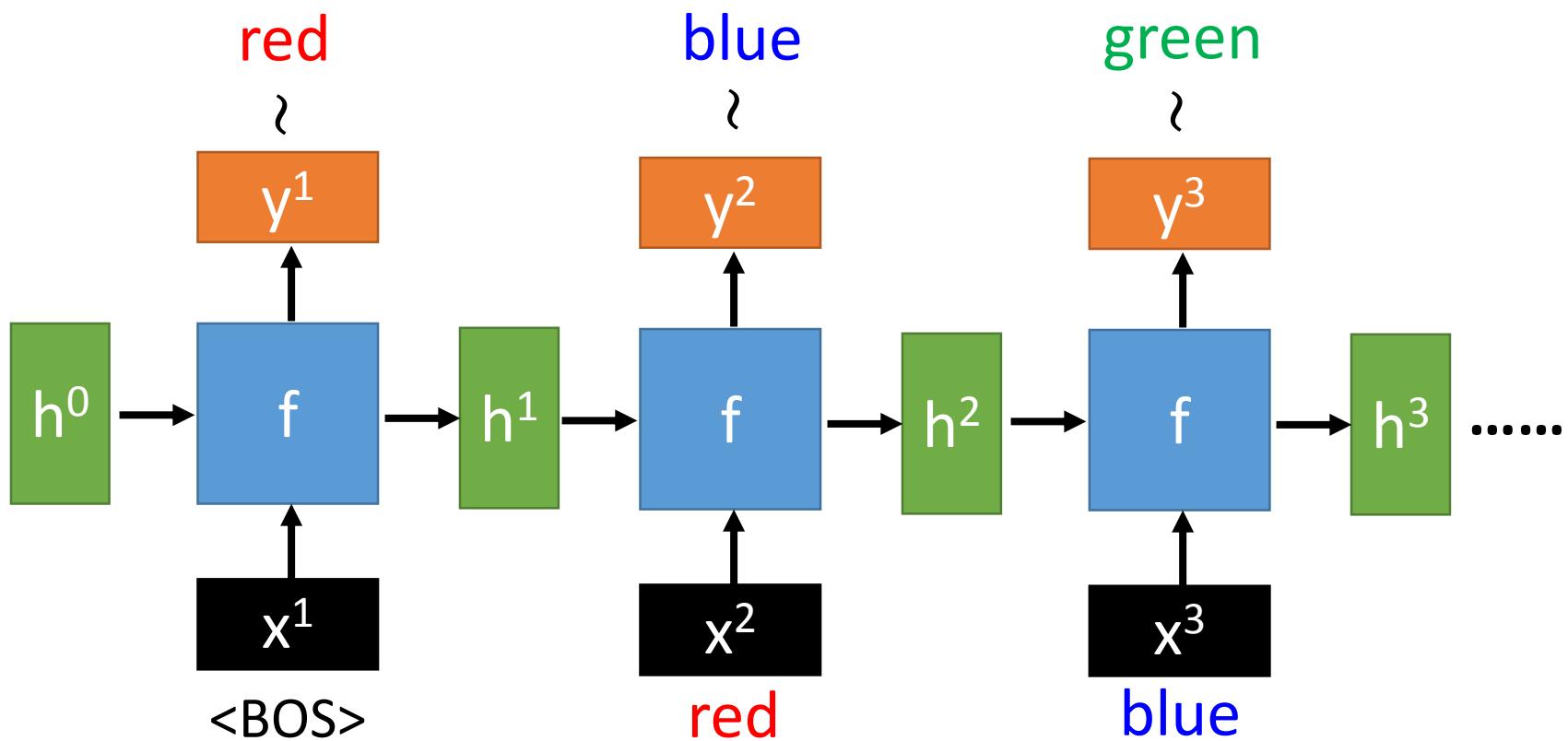
Generation



Consider as a sentence
blue red yellow gray

Train a RNN based on the
“sentences”

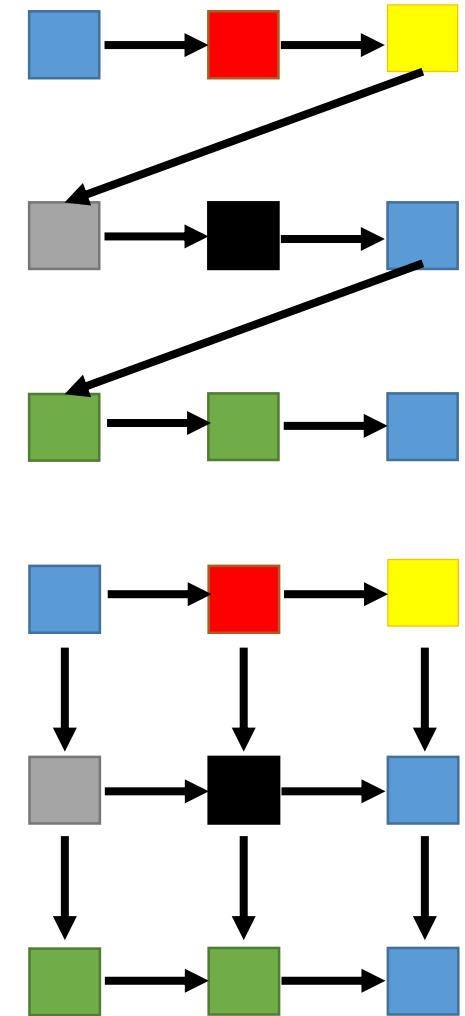
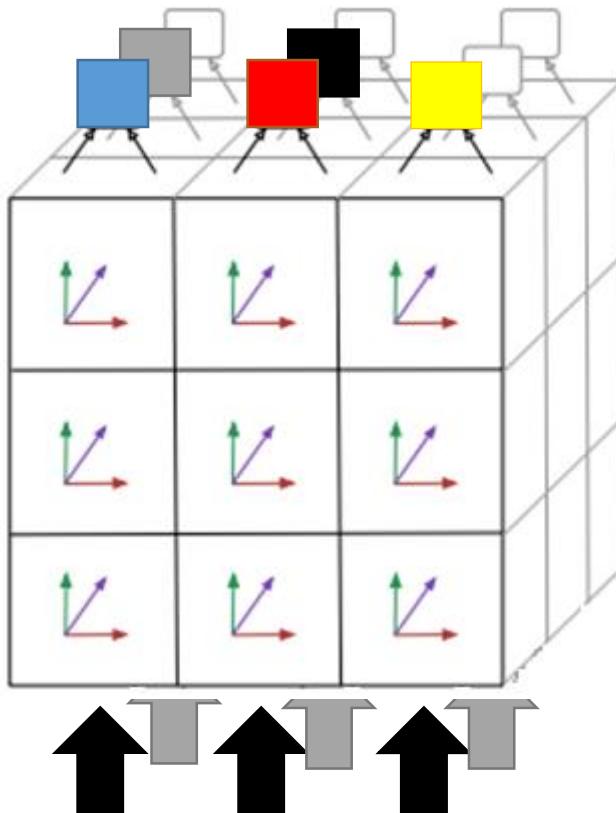
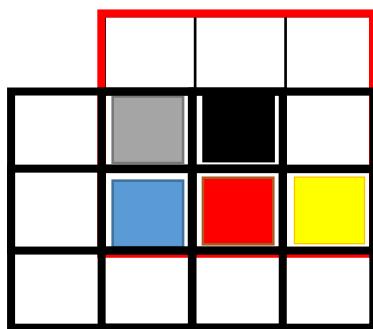
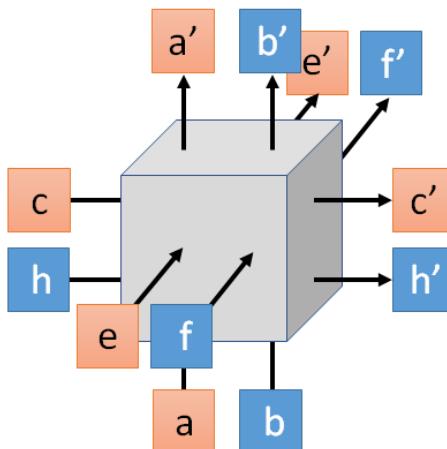
- Images are composed of pixels
- Generating a pixel at each time by RNN



Generation - PixelRNN

3 x 3 images

- Images are composed of pixels



Conditional Sequence Generation

Conditional Generation

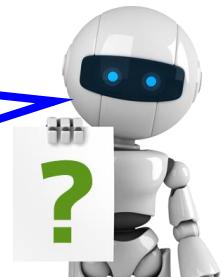
- We don't want to simply generate some random sentences.
- Generate sentences based on conditions:

Caption Generation

Given
condition:

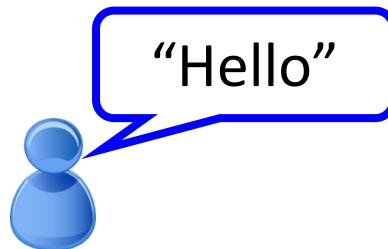


“A young girl
is dancing.”

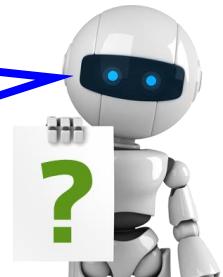


Chat-bot

Given
condition:



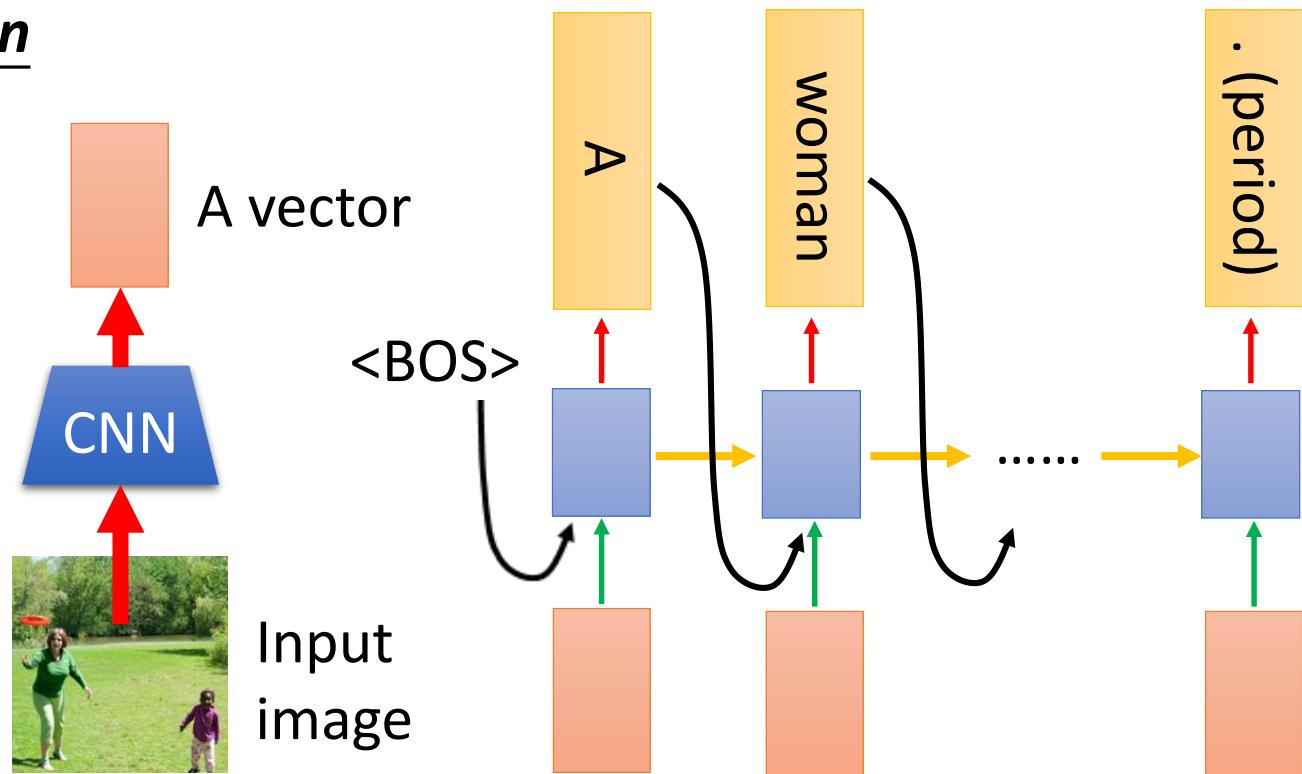
“Hello. Nice
to see you.”



Conditional Generation

- Represent the input condition as a vector, and consider the vector as the input of RNN generator

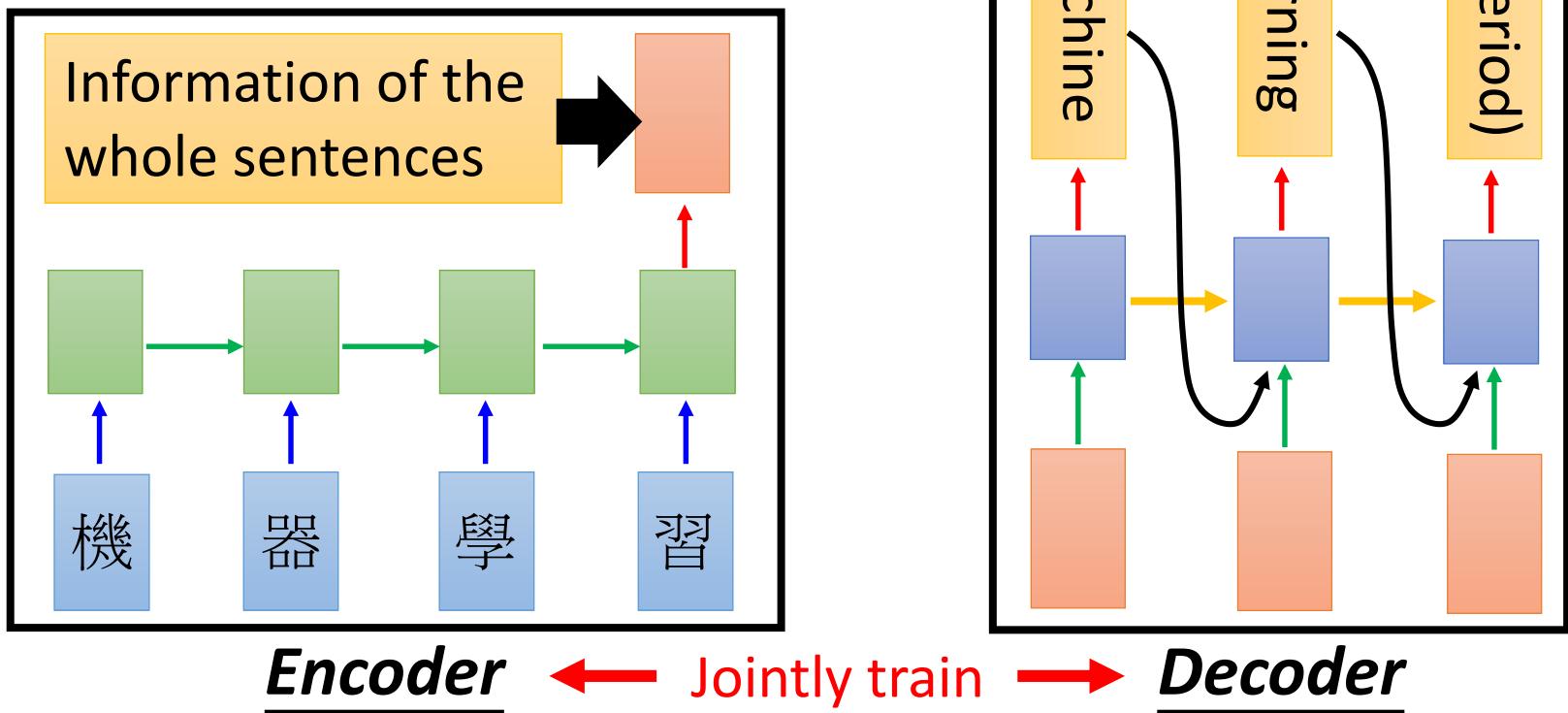
Image Caption Generation



Conditional Generation

Sequence-to-sequence learning

- Represent the input condition as a vector, and consider the vector as the input of RNN generator
- E.g. Machine translation / Chat-bot



Conditional Generation

M: Hello

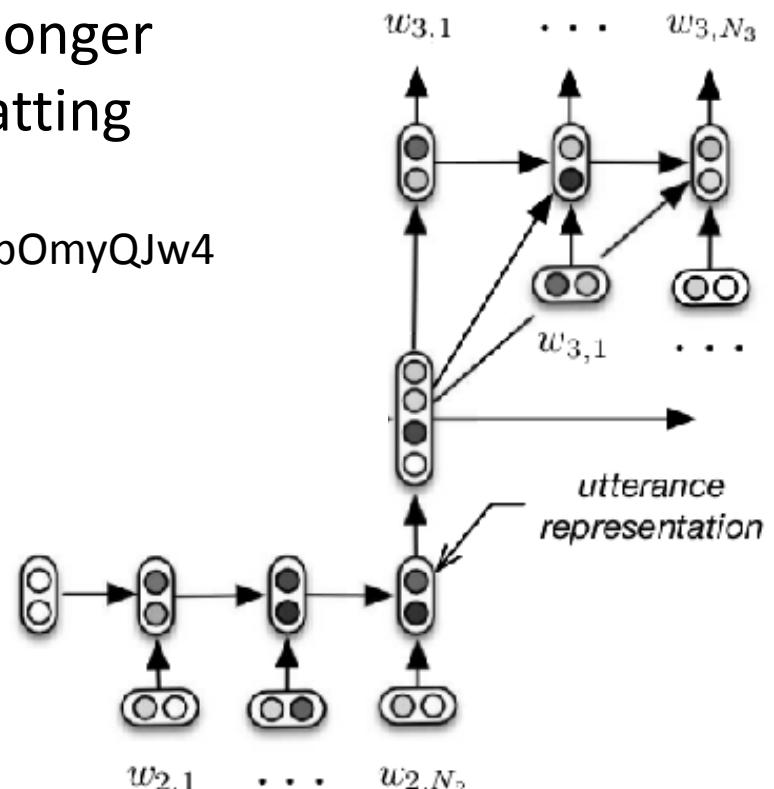
U: Hi

M: Hi

<https://www.youtube.com/watch?v=e2MpOmyQJw4>

Need to consider longer context during chatting

M: Hi 

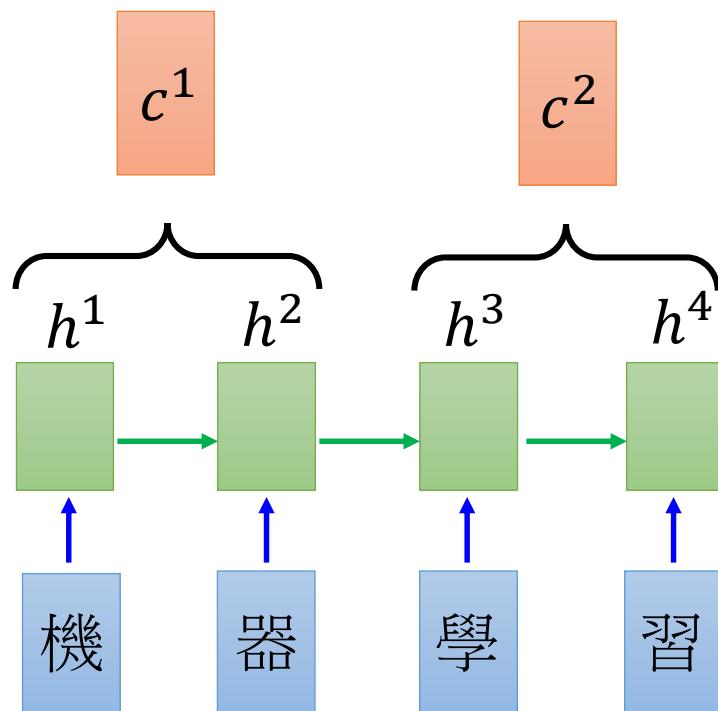


M: Hello

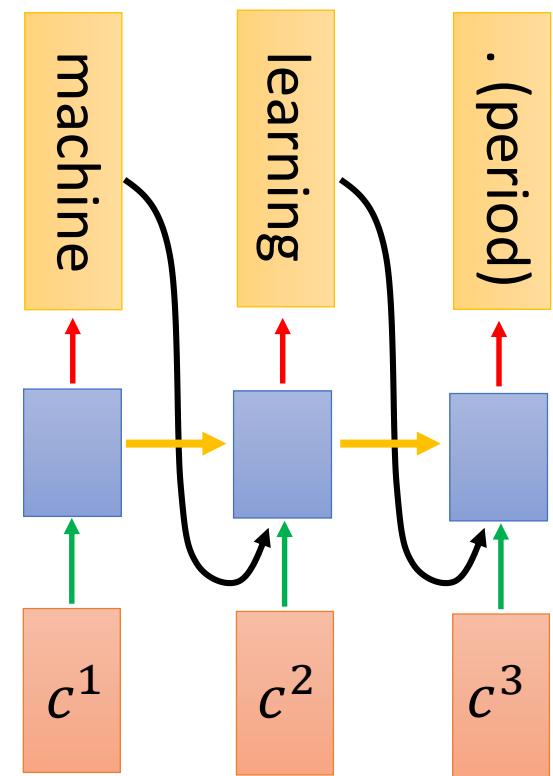
U: Hi

Serban, Iulian V., Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau, 2015
"Building End-To-End Dialogue Systems Using Generative Hierarchical Neural Network Models."

Dynamic Conditional Generation



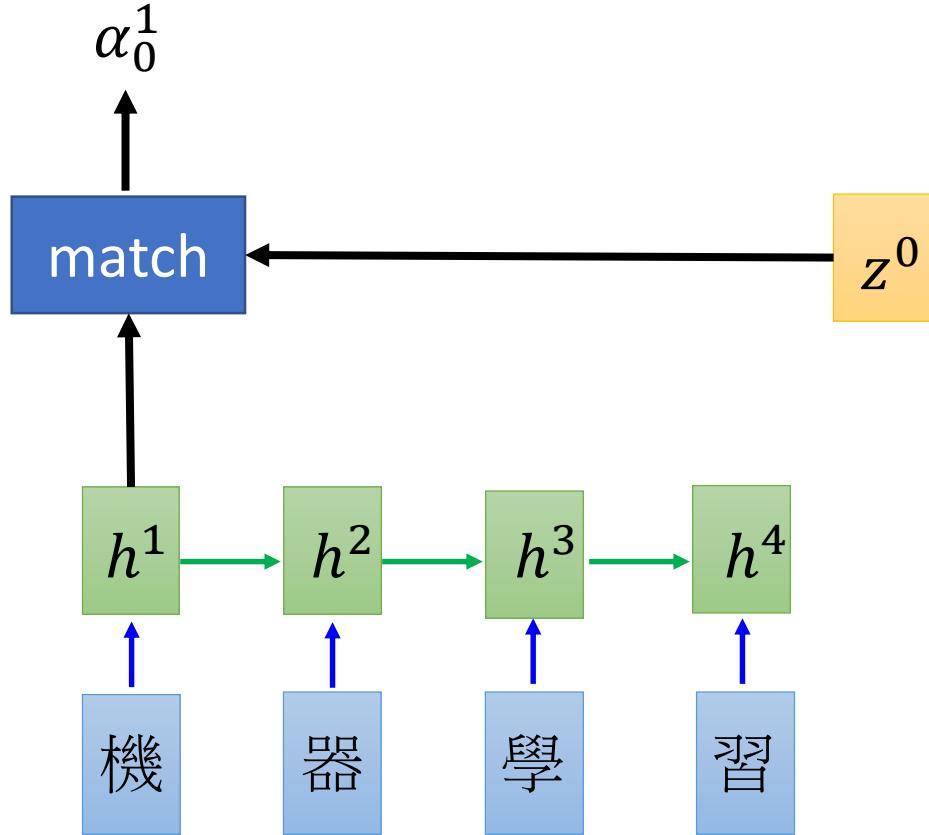
Encoder



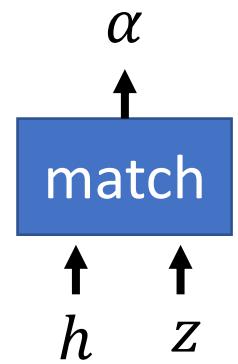
Decoder

Machine Translation

- Attention-based model



Jointly learned
with other part
of the network



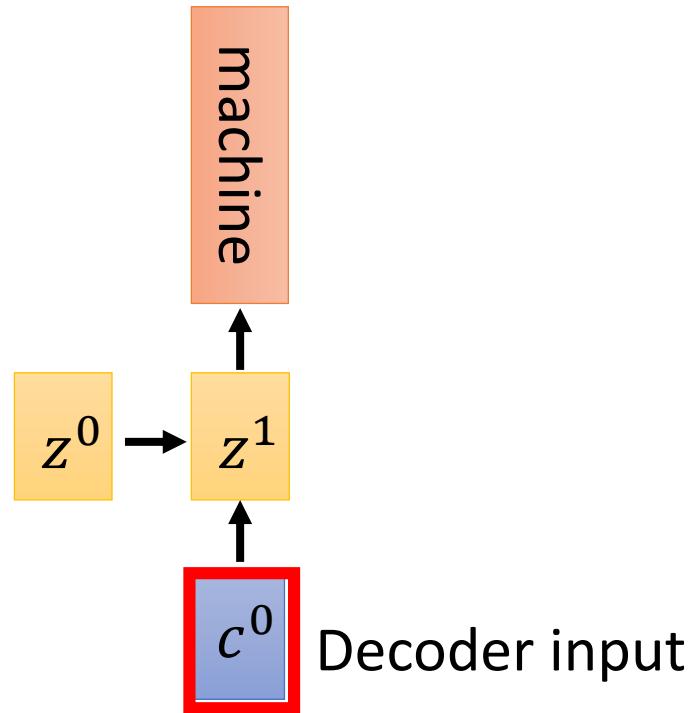
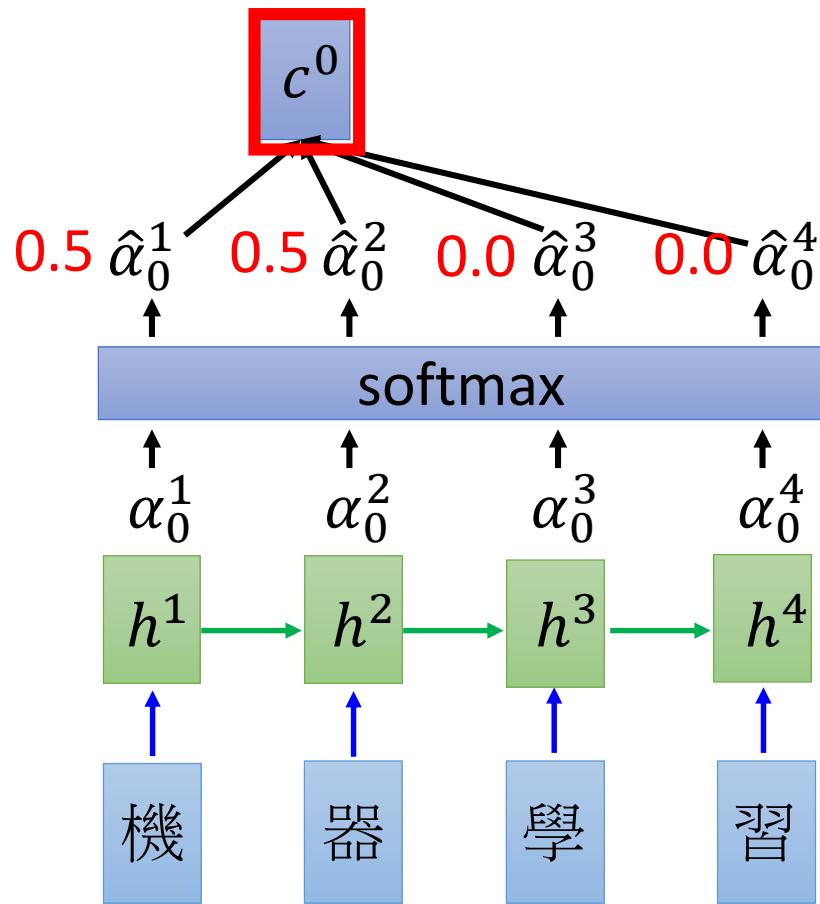
What is **match** ?

Design by yourself

- Cosine similarity of z and h
- Small NN whose input is z and h , output a scalar
- $\alpha = h^T W z$

Machine Translation

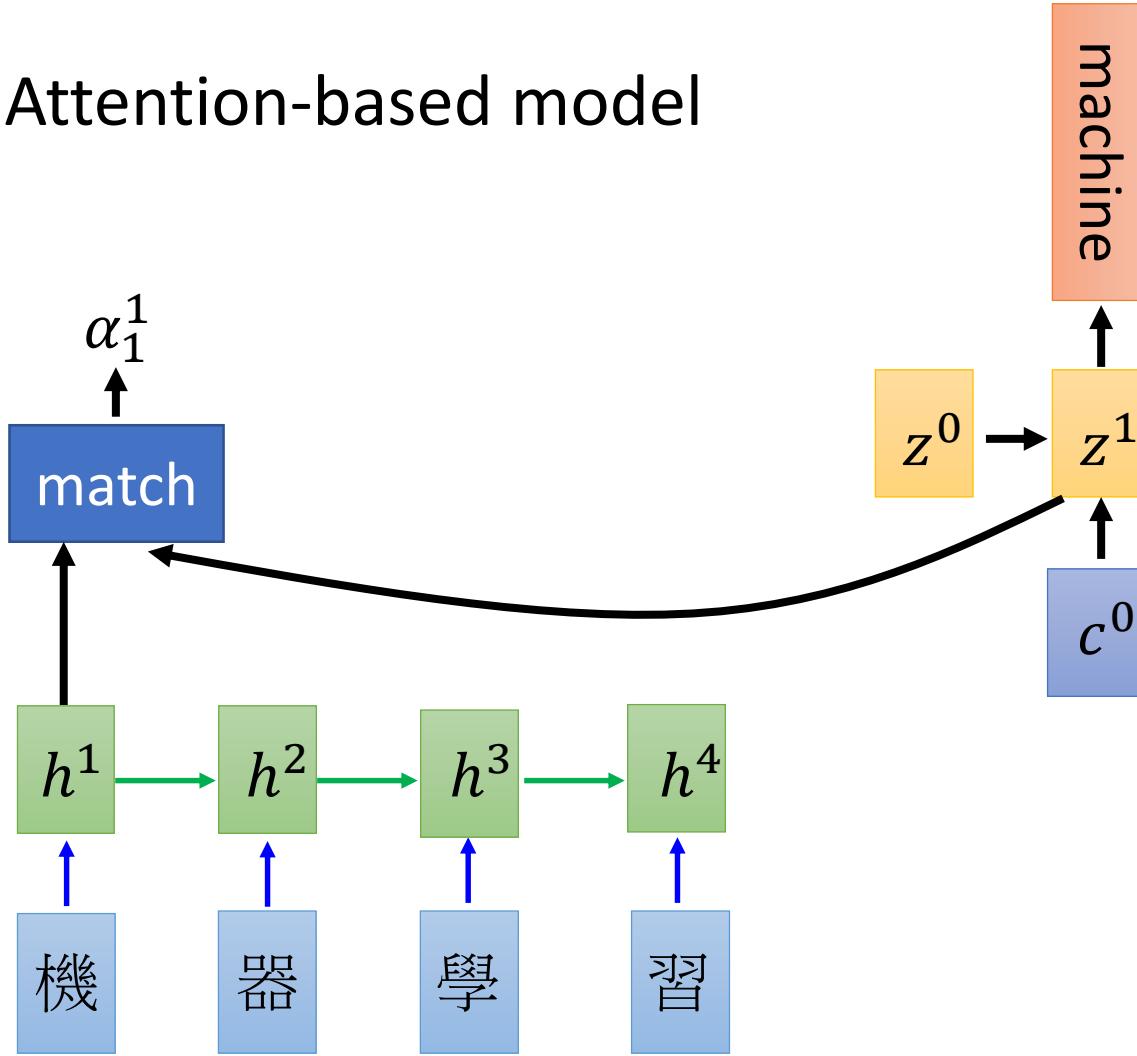
- Attention-based model



$$\begin{aligned}c^0 &= \sum \hat{\alpha}_0^i h^i \\&= 0.5h^1 + 0.5h^2\end{aligned}$$

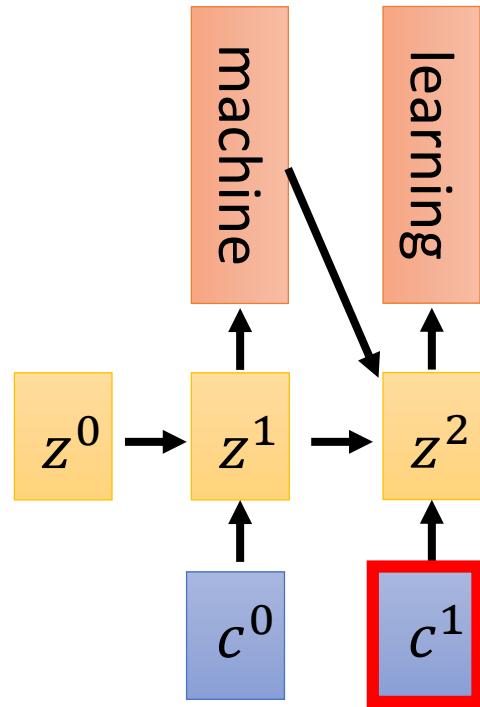
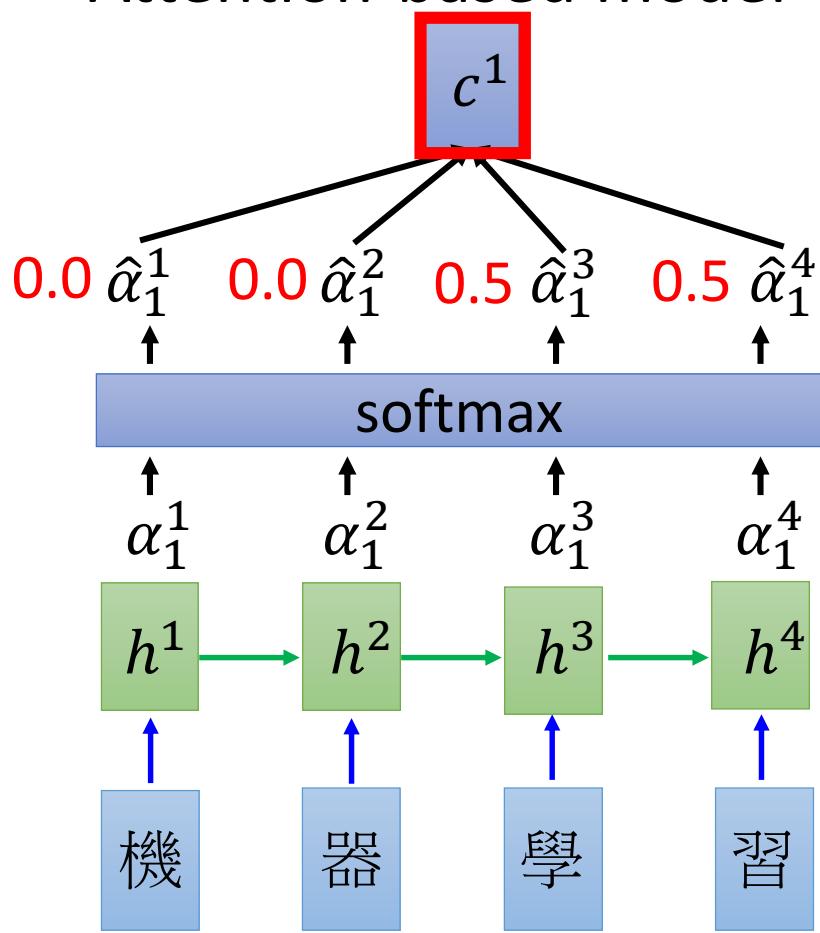
Machine Translation

- Attention-based model



Machine Translation

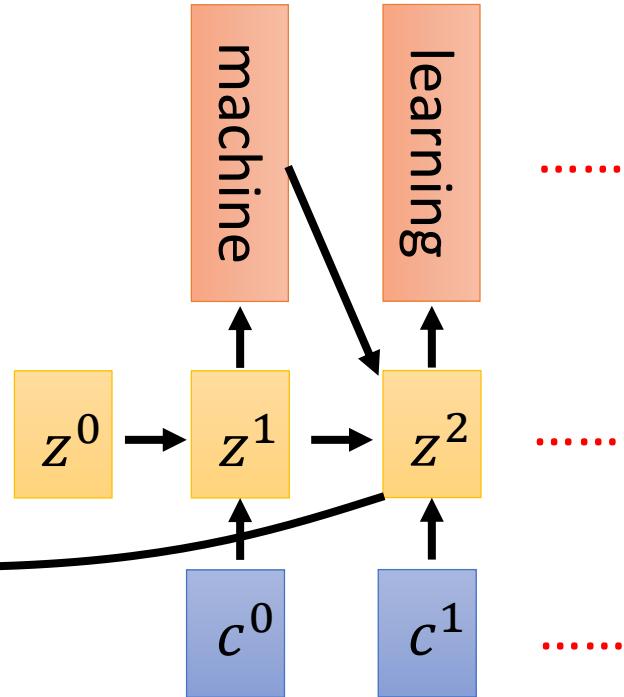
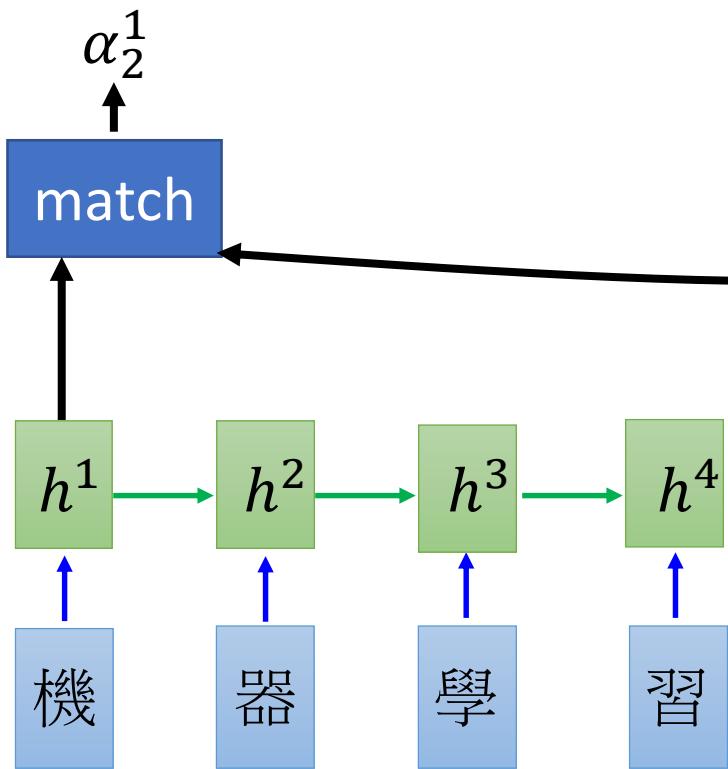
- Attention-based model



$$\begin{aligned} c^1 &= \sum \hat{\alpha}_1^i h^i \\ &= 0.5h^3 + 0.5h^4 \end{aligned}$$

Machine Translation

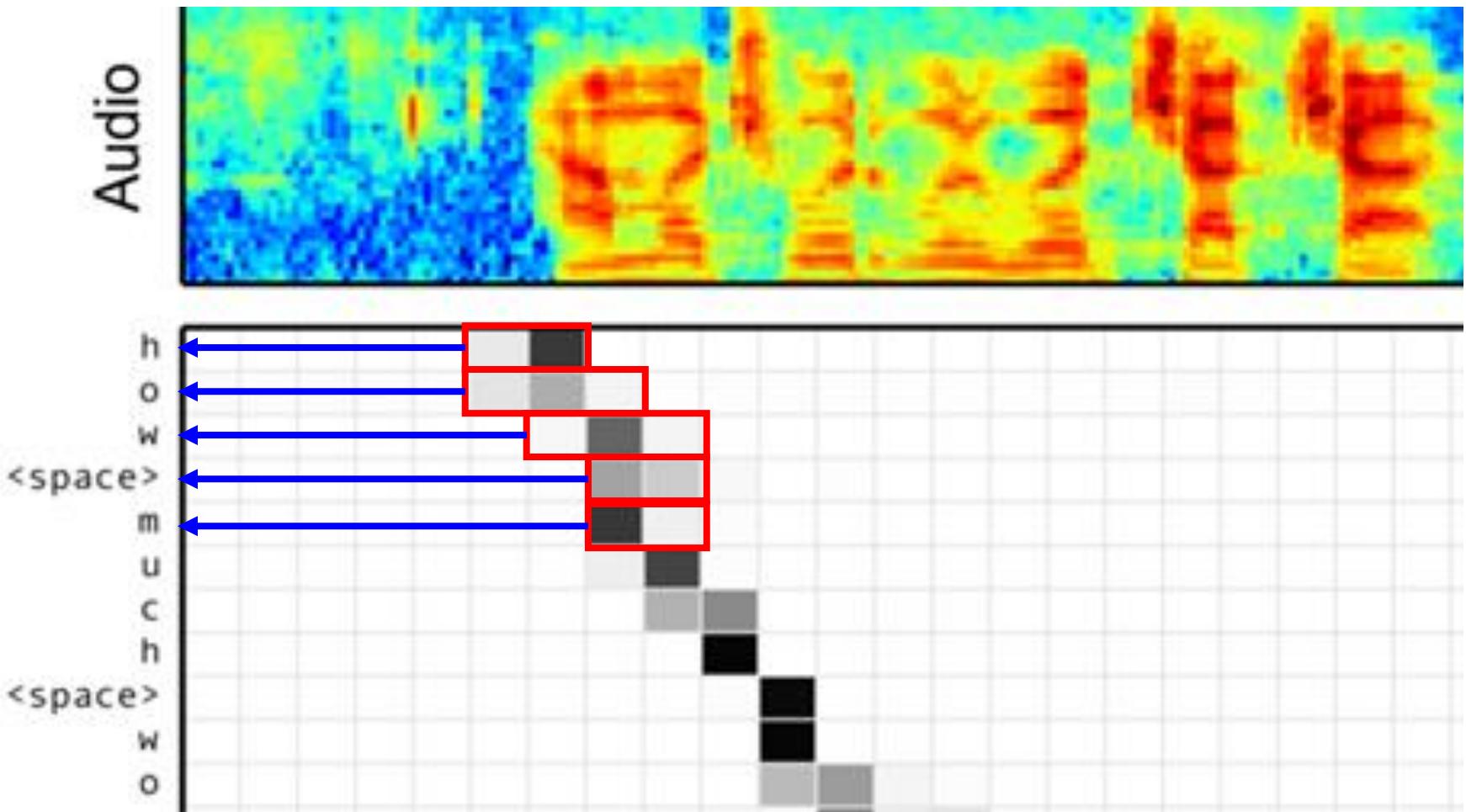
- Attention-based model



The same process repeat
until generating

<EOS>

Speech Recognition



Model	Clean WER	Noisy WER
CLDNN-HMM [22]	8.0	8.9
LAS	14.1	16.5
LAS + LM Rescoring	10.3	12.0

William Chan, Navdeep Jaitly, Quoc V. Le, Oriol Vinyals, “Listen, Attend and Spell”, ICASSP, 2016

Image Caption Generation

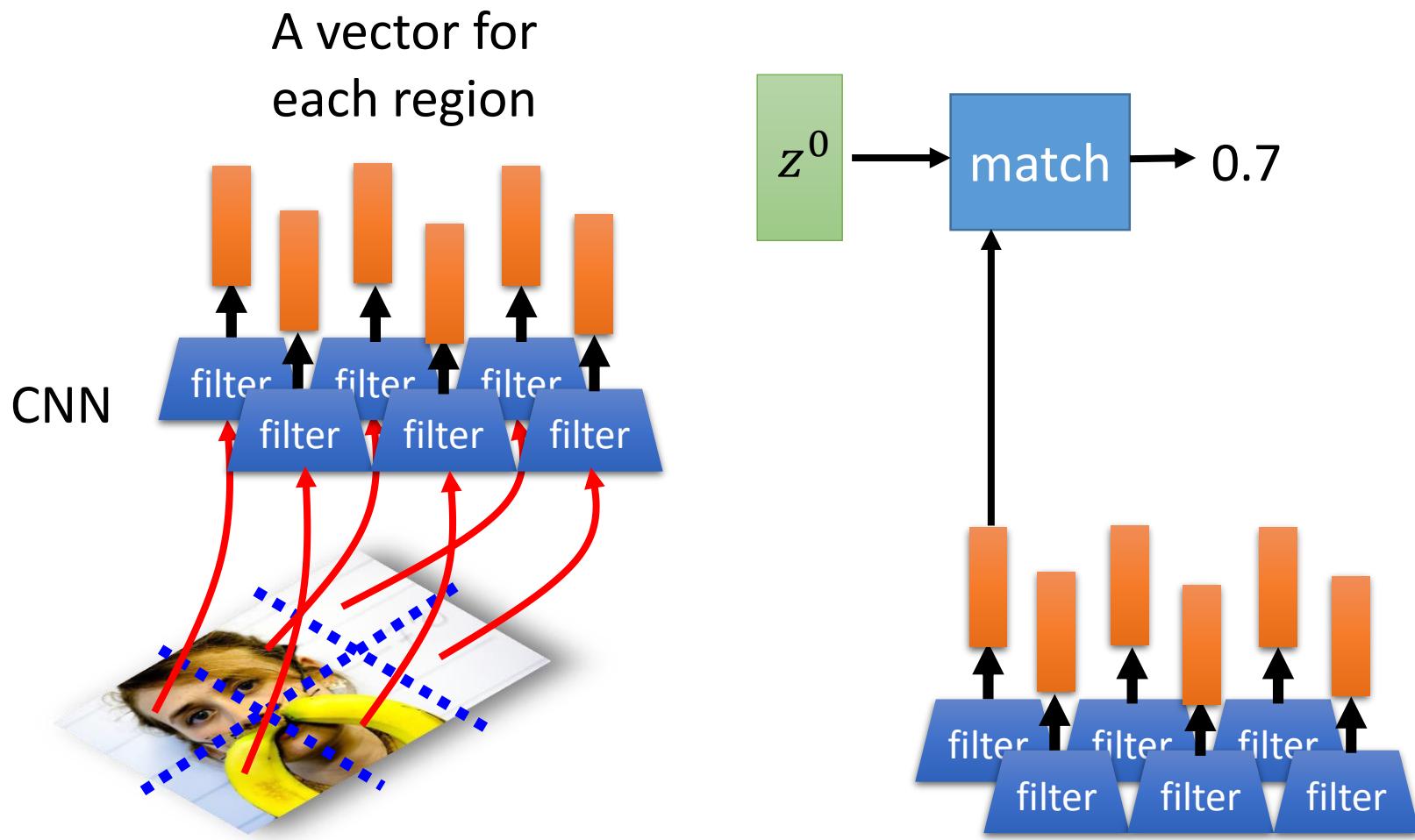


Image Caption Generation

A vector for each region

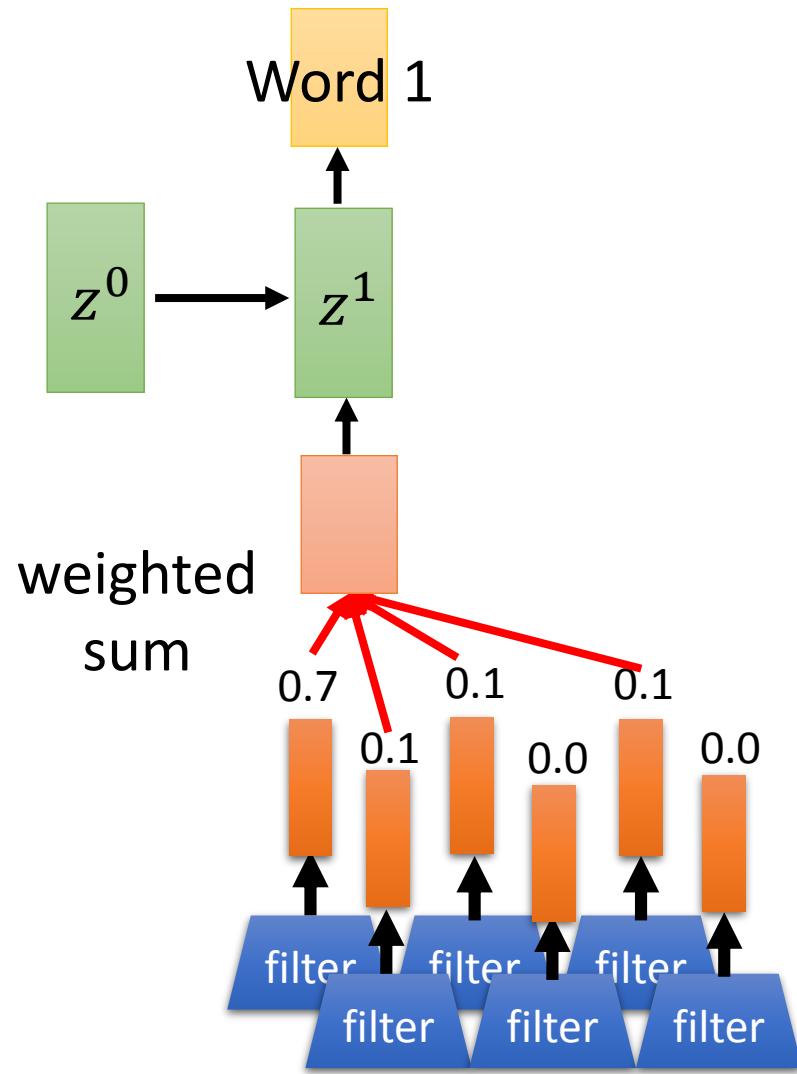
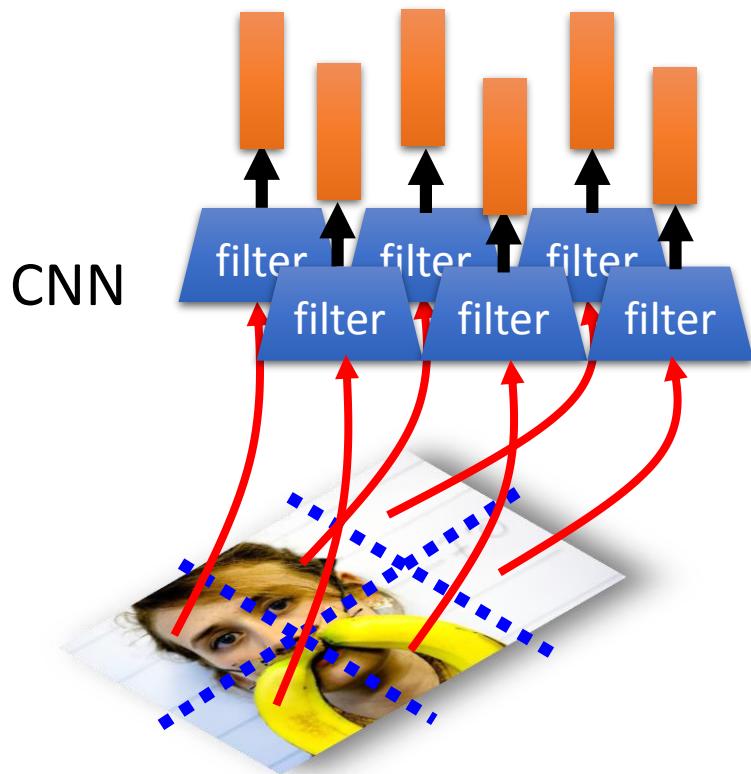


Image Caption Generation

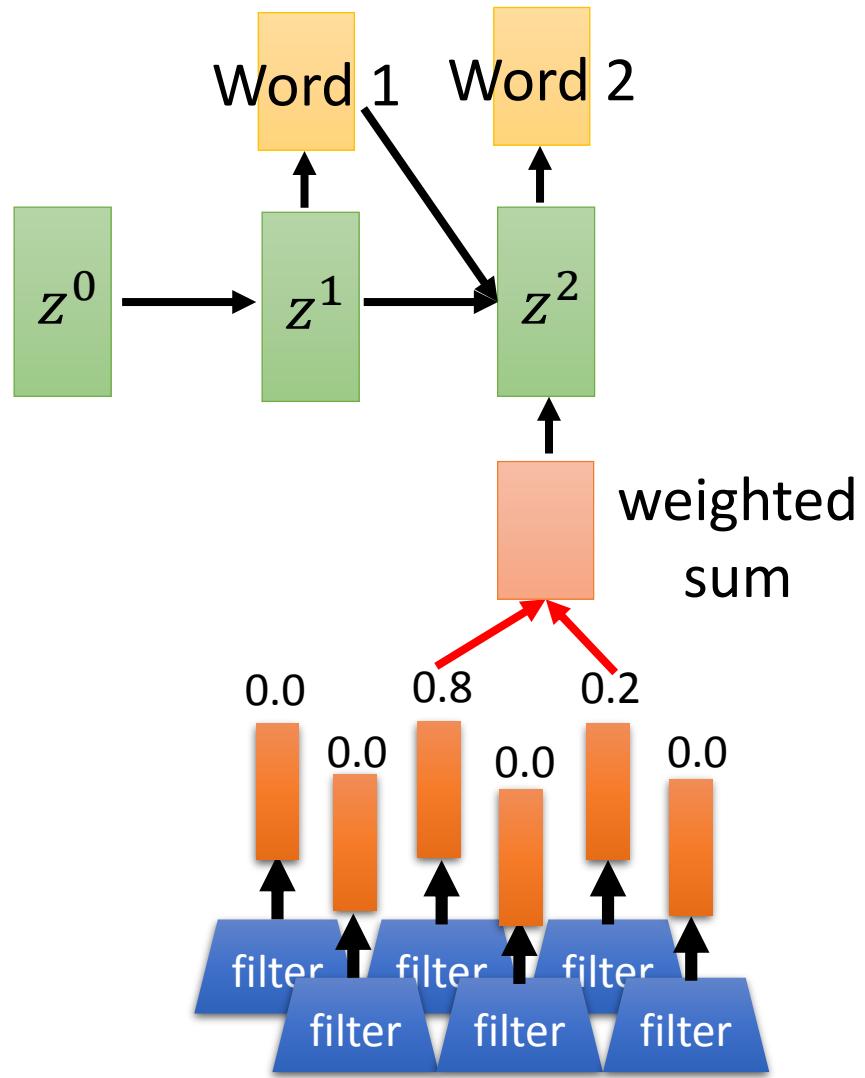
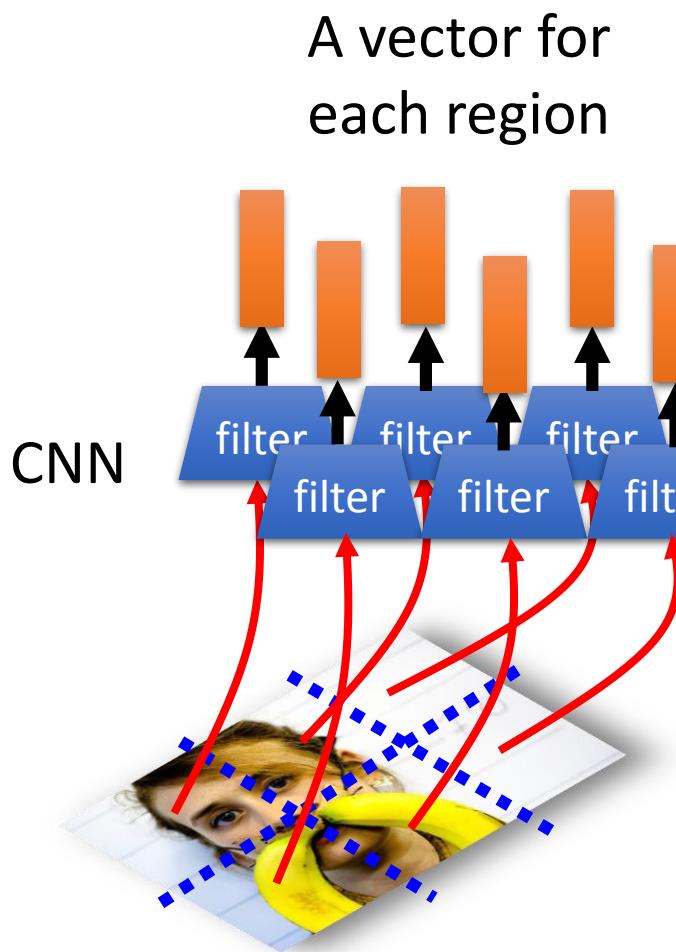


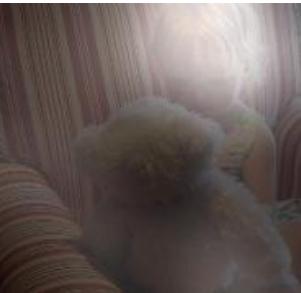
Image Caption Generation



A woman is throwing a frisbee in a park.

A dog is standing on a hardwood floor.

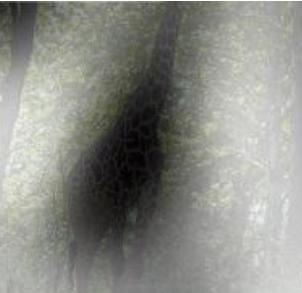
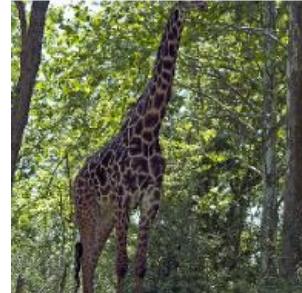
A stop sign is on a road with a mountain in the background.



A little girl sitting on a bed with a teddy bear.



A group of people sitting on a boat in the water.



A giraffe standing in a forest with trees in the background.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, Yoshua Bengio, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention", ICML, 2015

Image Caption Generation



A large white bird standing in a forest.



A woman holding a clock in her hand.



A man wearing a hat and a hat on a skateboard.



A person is standing on a beach with a surfboard.



A woman is sitting at a table with a large pizza.



A man is talking on his cell phone while another man watches.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, Yoshua Bengio, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention", ICML, 2015



Ref: A man and a woman ride a motorcycle

A **man** and a **woman** are **talking** on the **road**



Ref: A woman is frying food

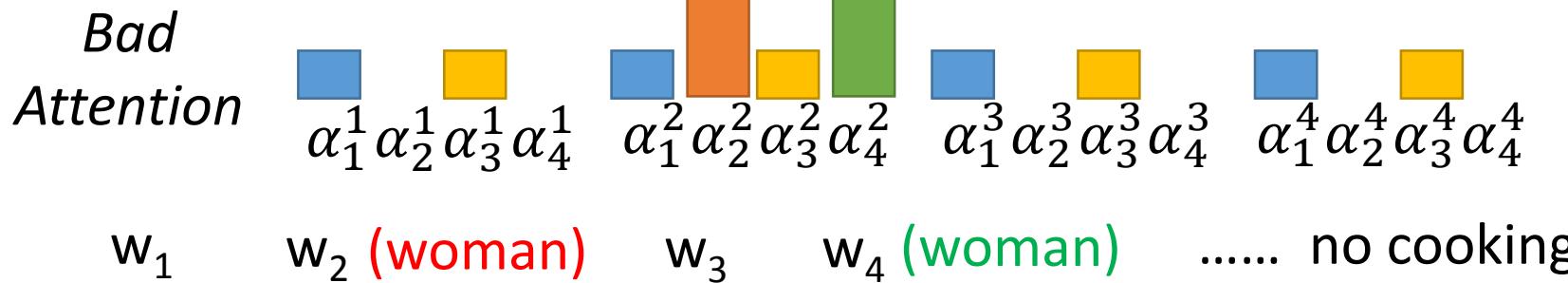
Someone is **frying** a **fish** in a **pot**

Tips for Generation

Attention

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, Yoshua Bengio, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention", ICML, 2015

component
 α_t^i → time



Good Attention: each input component has approximately the same attention weight

E.g. Regularization term: $\sum_i \left(\tau - \sum_t \alpha_t^i \right)^2$

For each component Over the generation

Mismatch between Train and Test

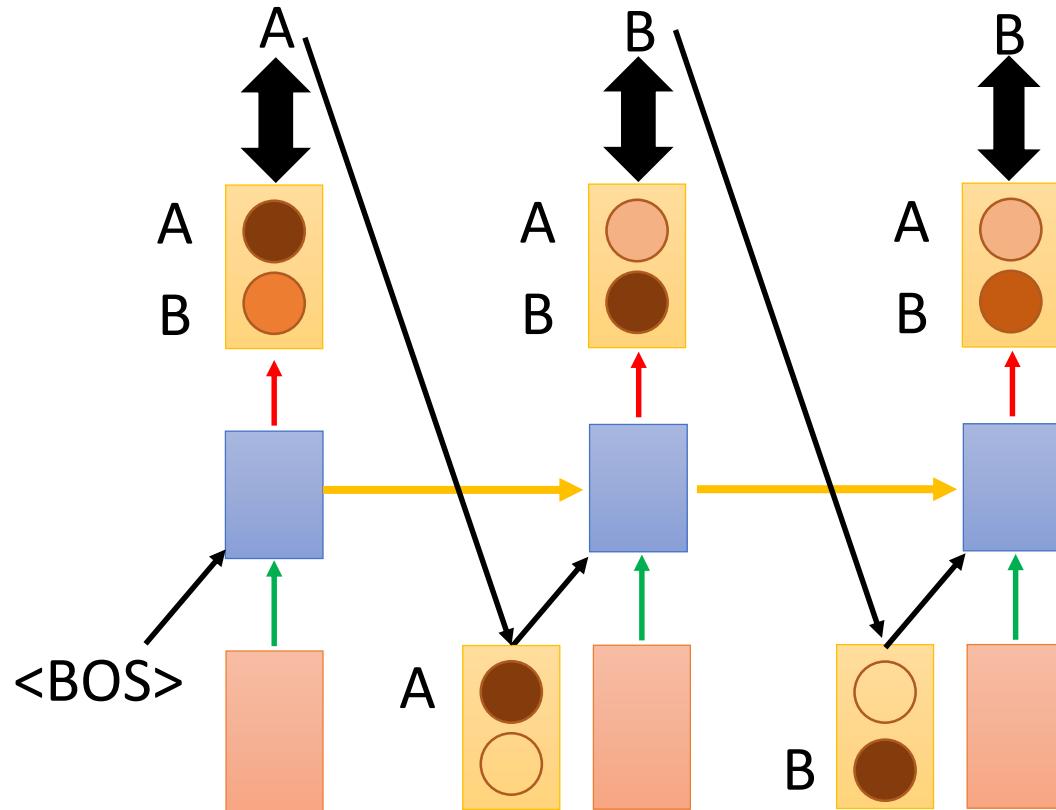
- Training

$$C = \sum_t C_t$$

Minimizing
cross-entropy of
each component

 : condition

Reference:



Mismatch between Train and Test

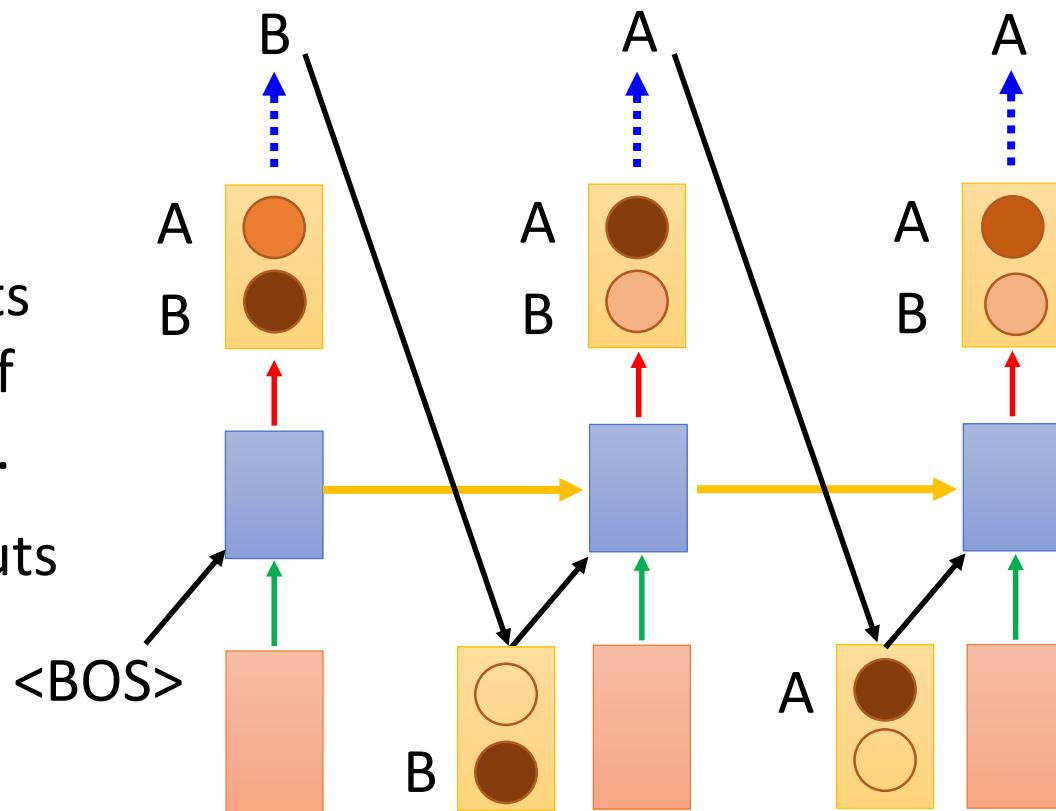
- *Generation*

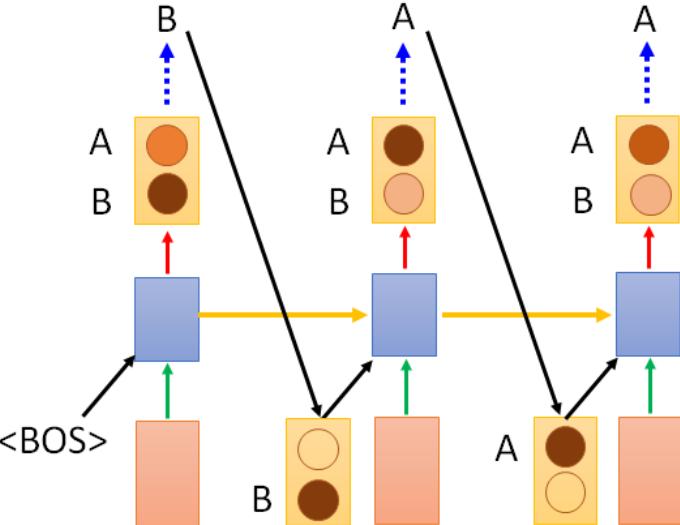
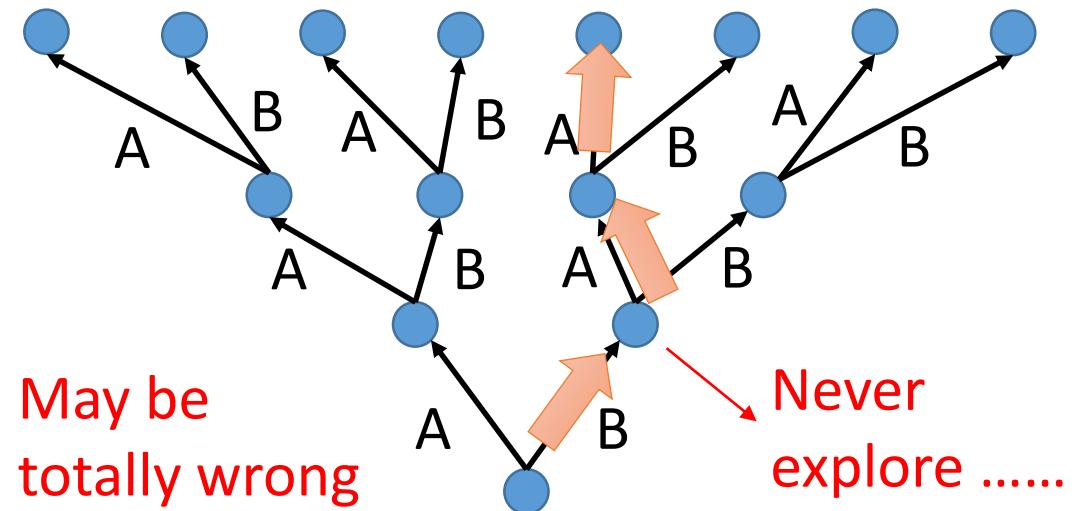
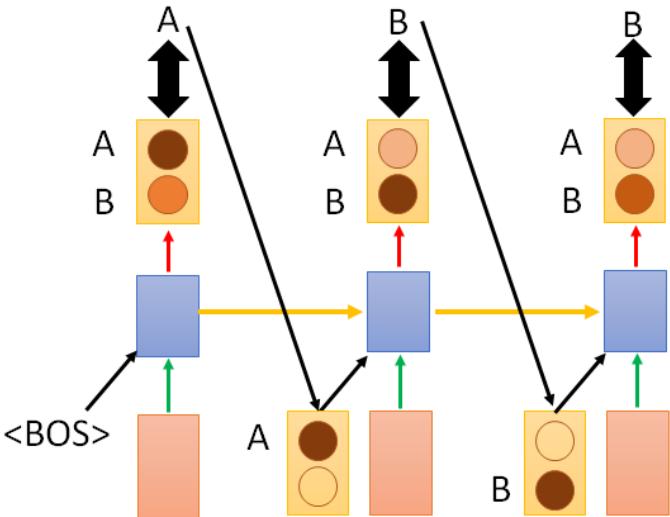
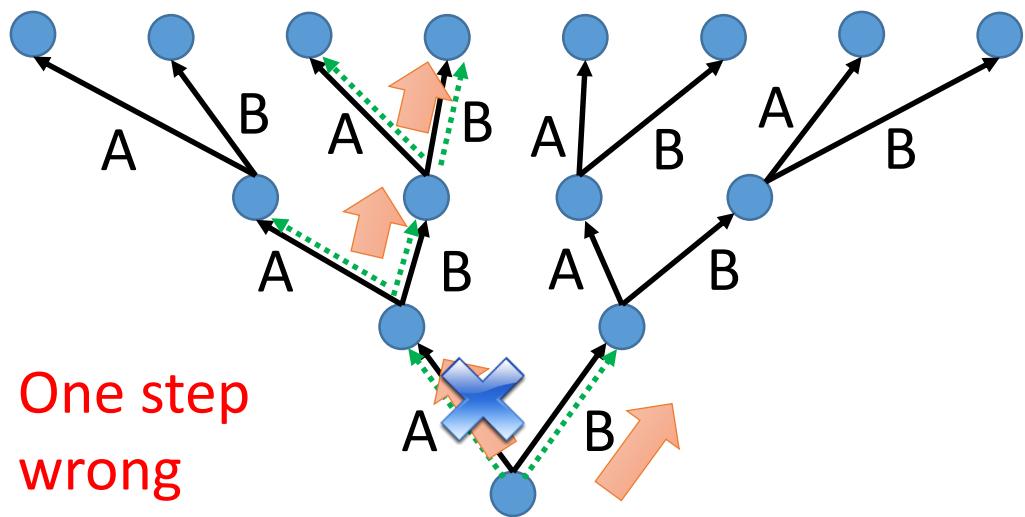
We do not know
the reference

Testing: The inputs
are the outputs of
the last time step.

Training: The inputs
are reference.

Exposure Bias





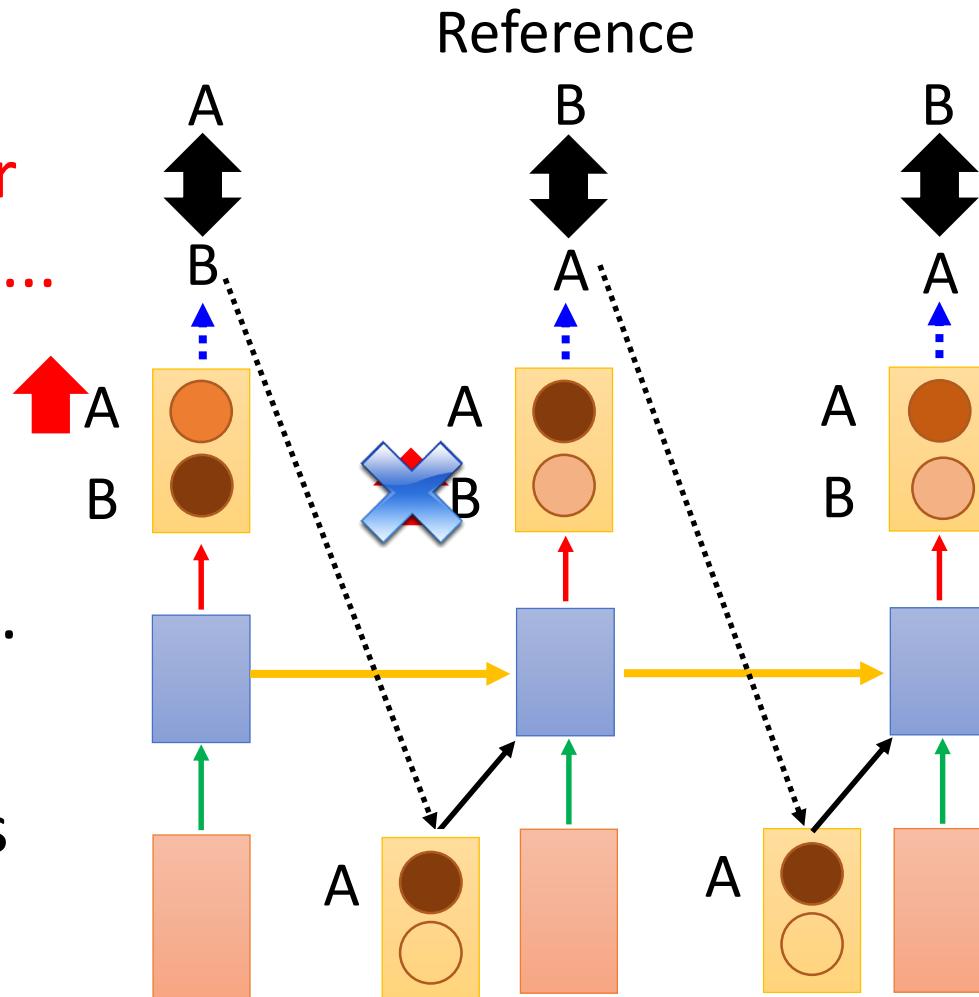
一步錯，步步錯

Modifying Training Process?

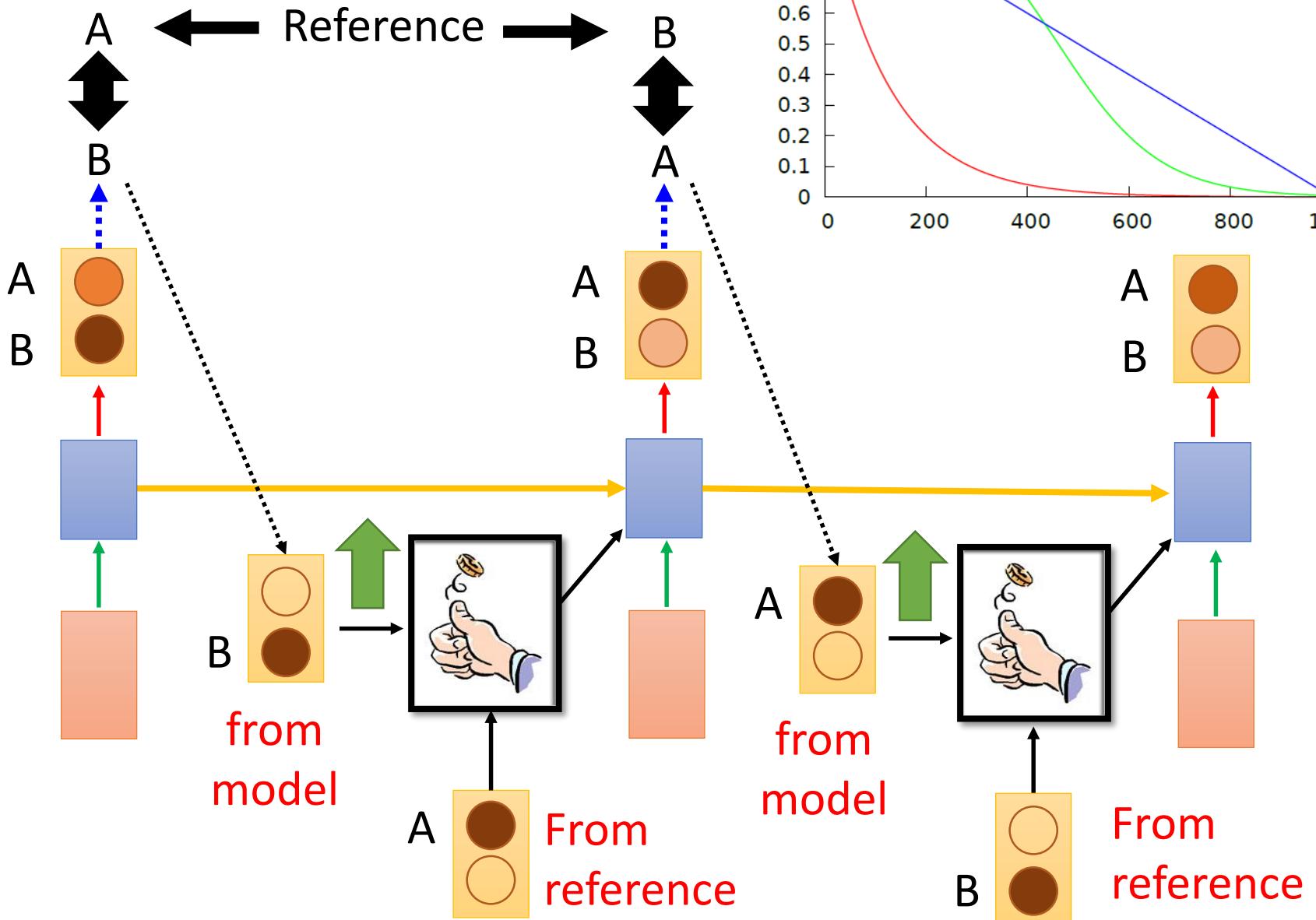
When we try to
decrease the loss for
both steps 1 and 2

Training is
matched to testing.

In practice, it is
hard to train in this
way.



Scheduled Sampling



Scheduled Sampling

- Caption generation on MSCOCO

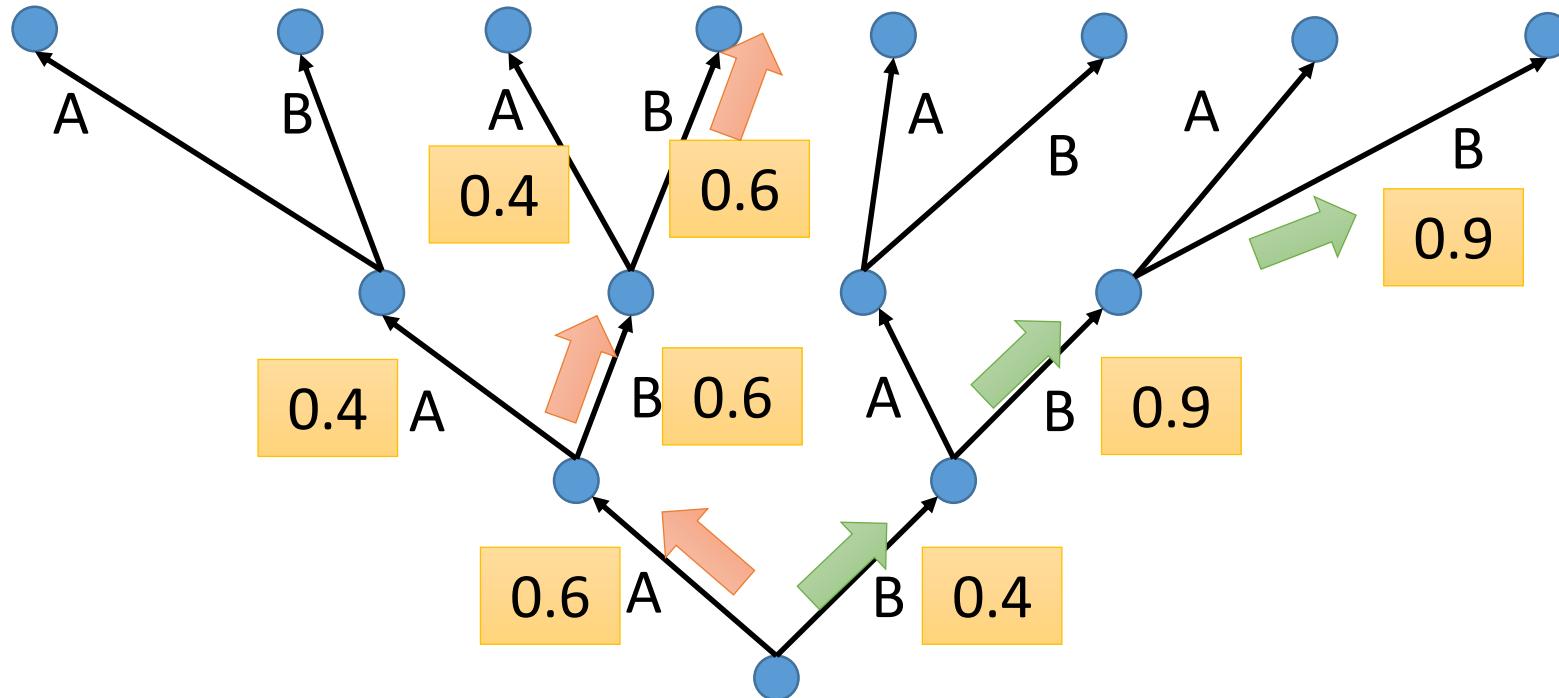
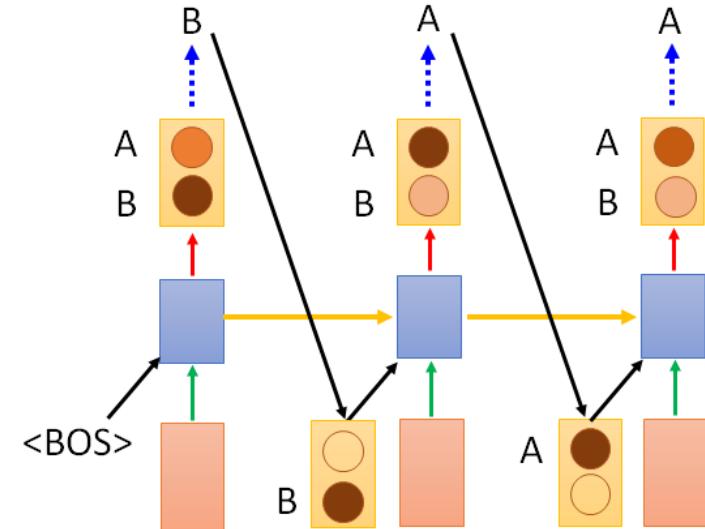
	BLEU-4	METEOR	CIDER
Always from reference	28.8	24.2	89.5
Always from model	11.2	15.7	49.7
Scheduled Sampling	30.6	24.3	92.1

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, Noam Shazeer, Scheduled Sampling for Sequence Prediction with Recurrent Neural Networks, arXiv preprint, 2015

Beam Search

The green path has higher score.

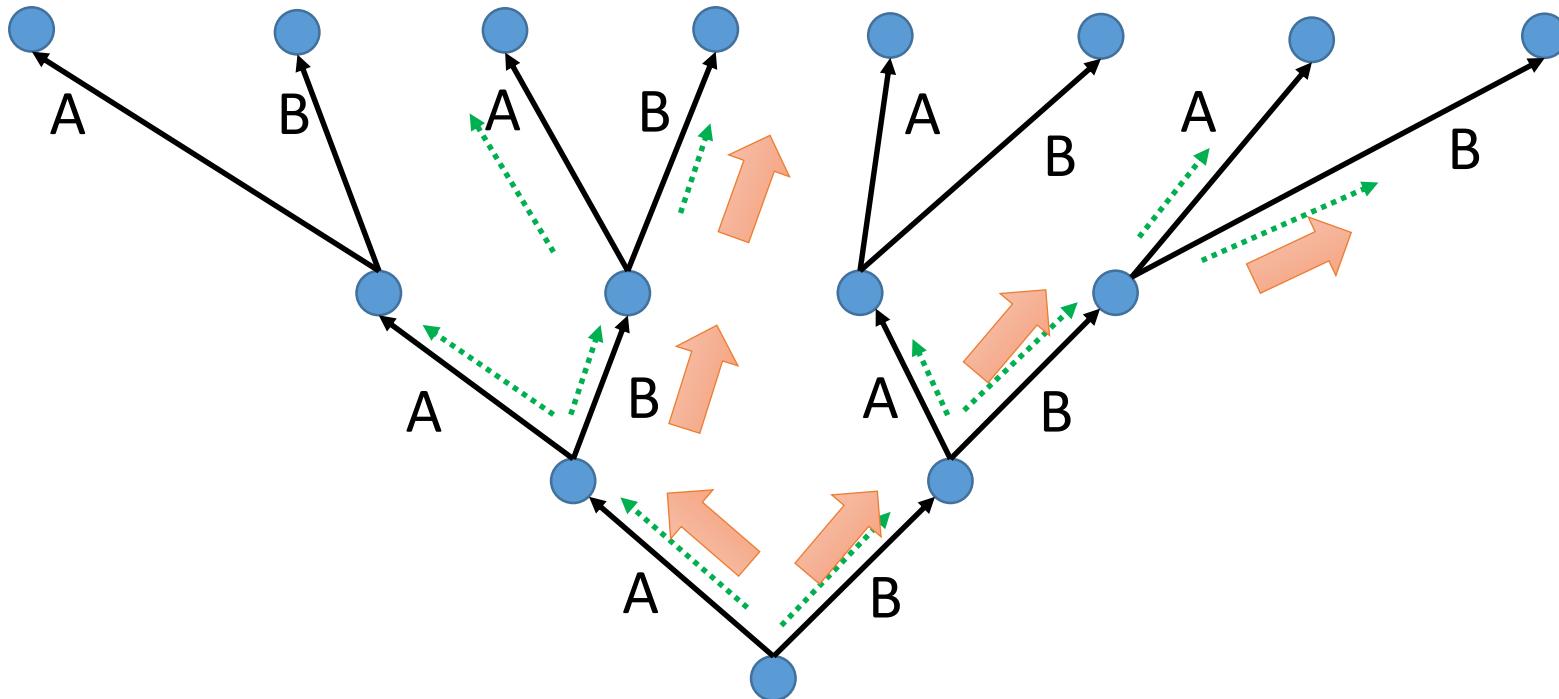
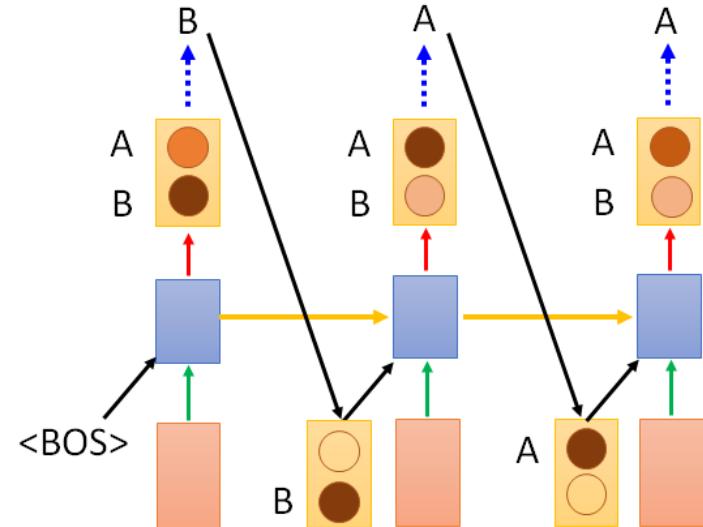
Not possible to check all the paths



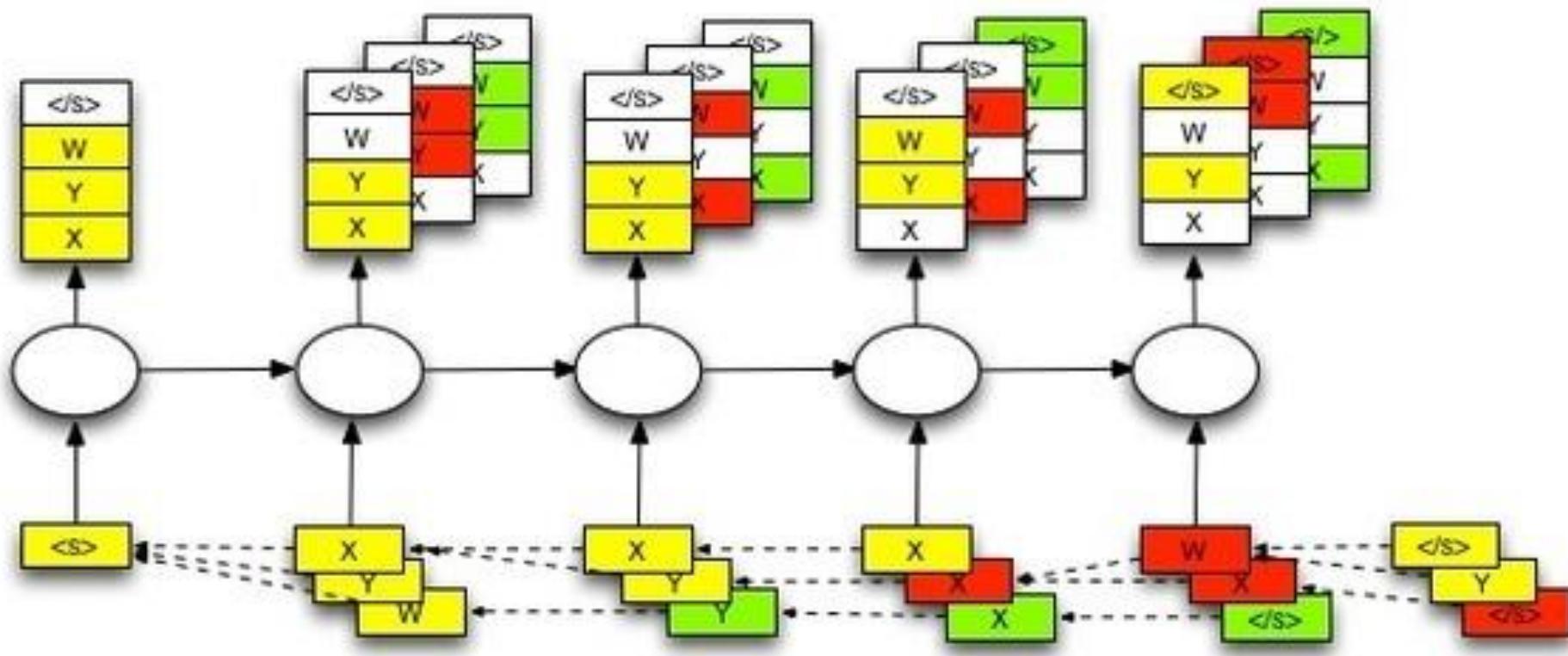
Beam Search

Keep several best path at each step

Beam size = 2



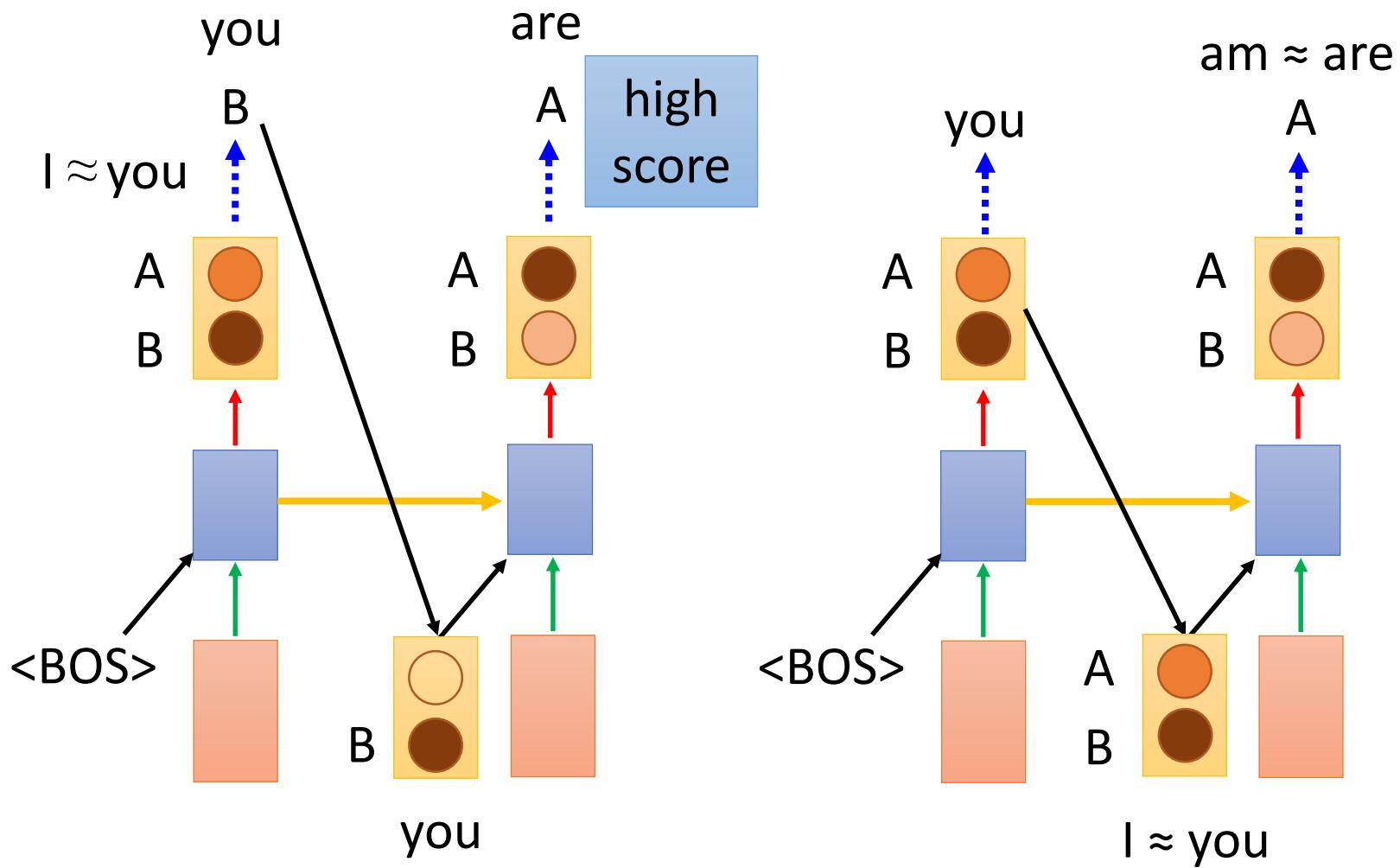
Beam Search



The size of beam is 3 in this example.

Better Idea?

I am ✓
You are ✓
I are ✗
You am ✗



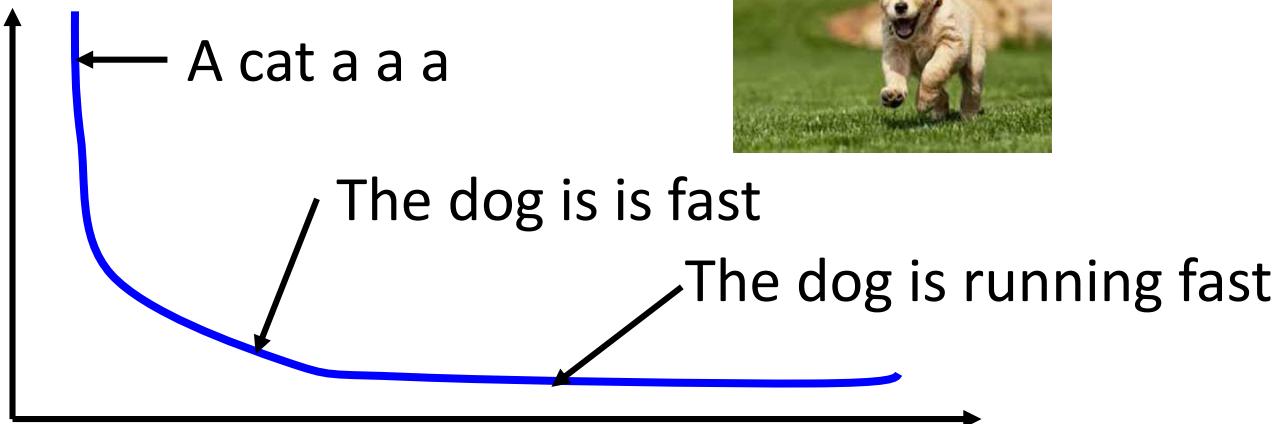
Object level v.s. Component level

- Minimizing the error defined on component level is not equivalent to improving the generated objects

Ref: The dog is running fast

$$C = \sum_t C_t$$

Cross-entropy
of each step



Optimize object-level criterion instead of component-level cross-entropy. object-level criterion: $R(y, \hat{y})$ Gradient Descent?

y : generated utterance, \hat{y} : ground truth

Reinforcement learning?

Start with
observation s_1

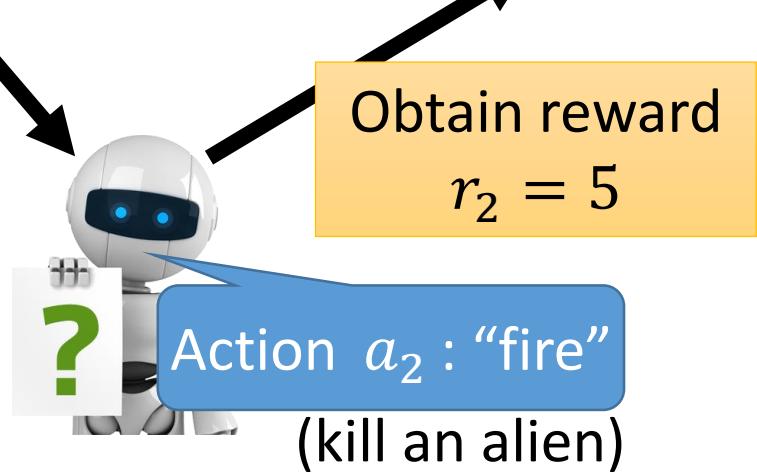
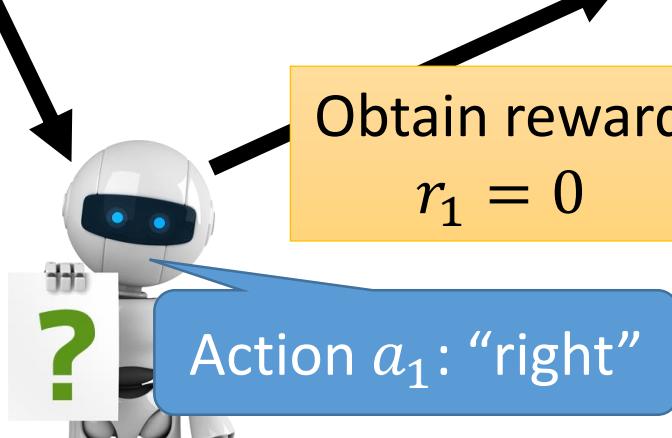
Observation s_2

Observation s_3



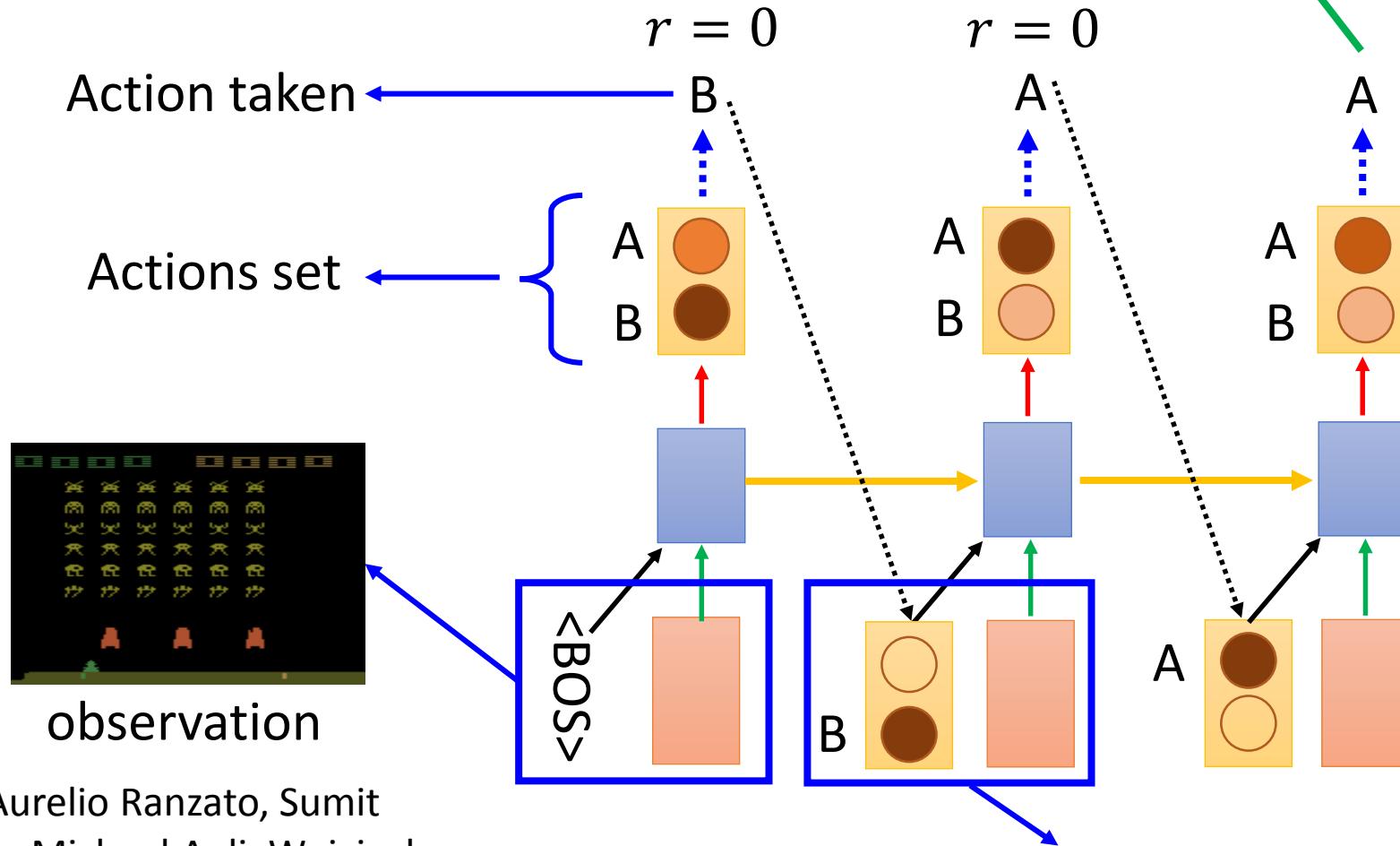
Obtain reward
 $r_1 = 0$

Obtain reward
 $r_2 = 5$



Reinforcement learning?

reward:
 $R(\text{"BAA"}, \text{reference})$



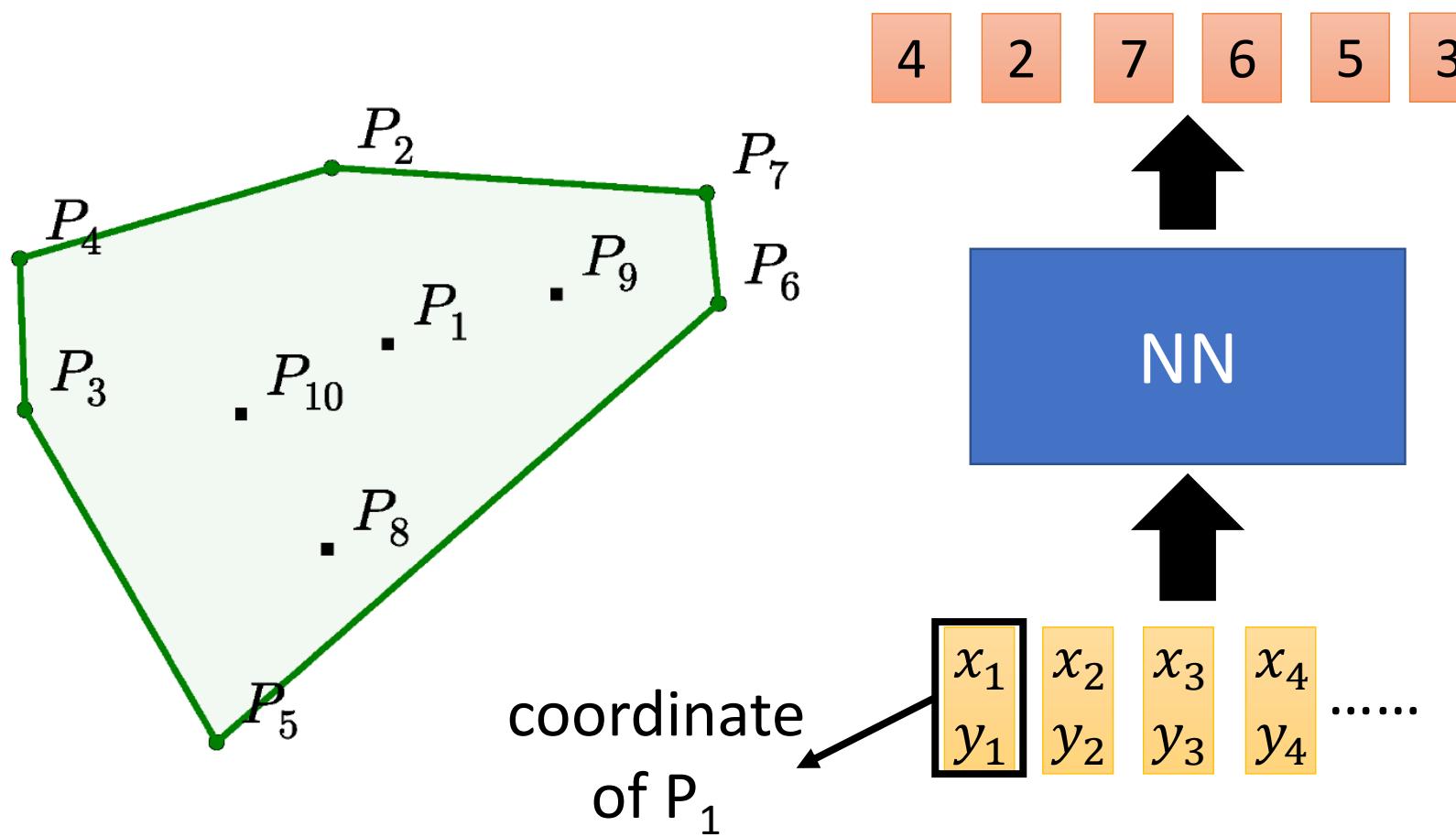
Marc'Aurelio Ranzato, Sumit
Chopra, Michael Auli, Wojciech
Zaremba, "Sequence Level Training with
Recurrent Neural Networks", ICLR, 2016

The action we take influence
the observation in the next step

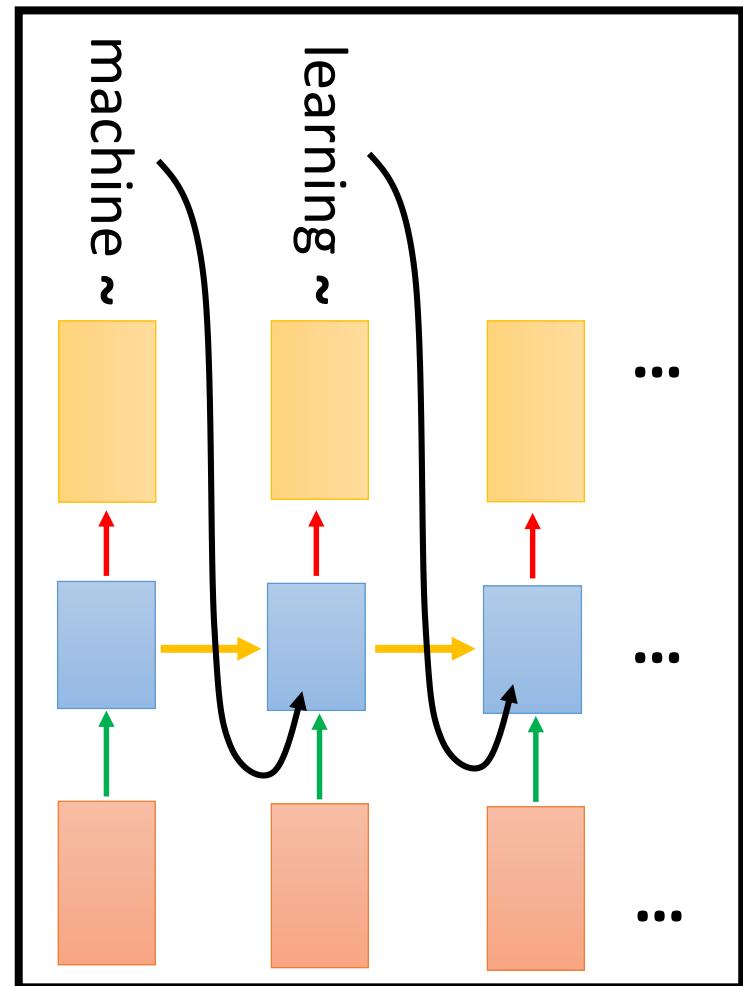
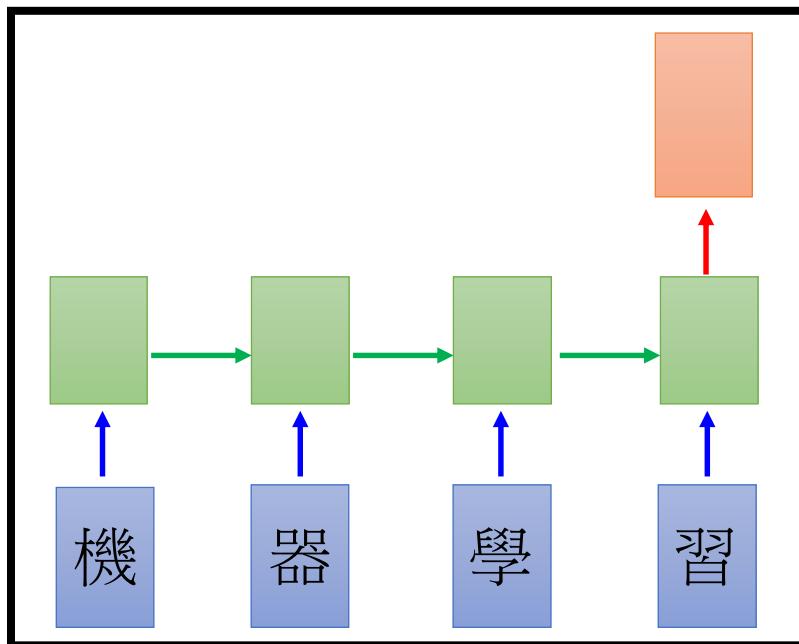
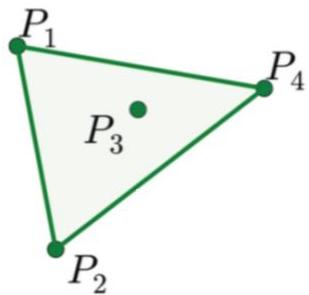
Concluding Remarks

- RNN with Gated Mechanism
- Sequence Generation
- Conditional Sequence Generation
- Tips for Generation

Pointer Network

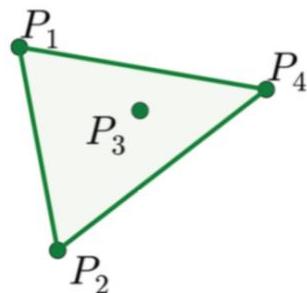


Sequence-to-sequence?



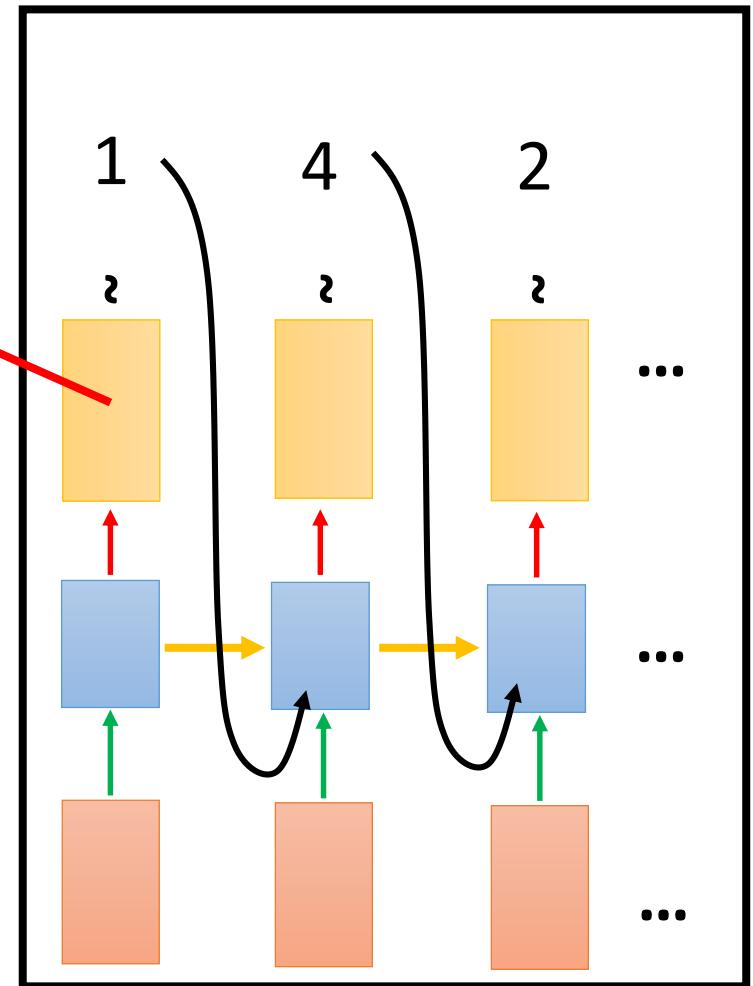
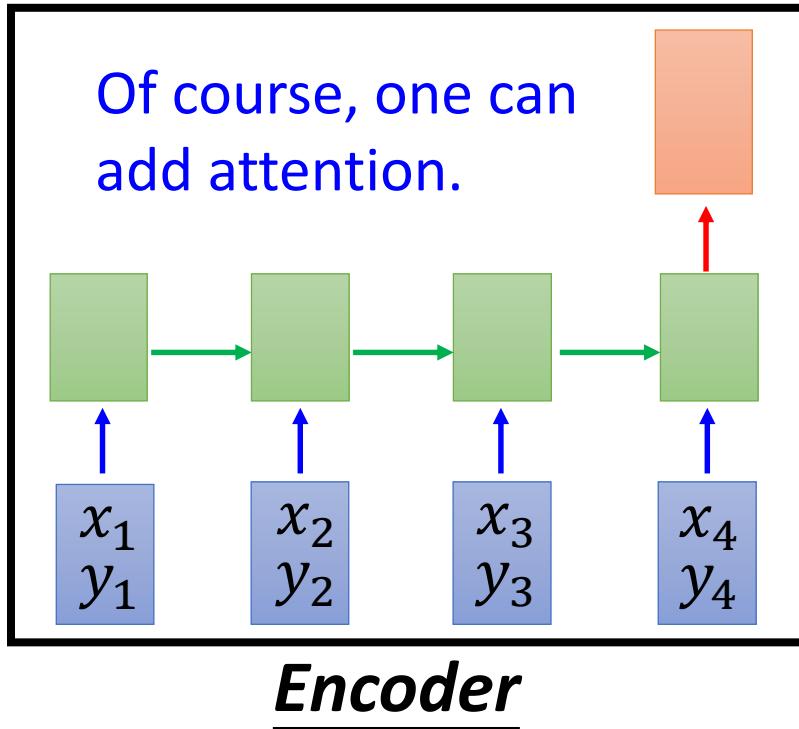
Problem?

Sequence-to-sequence?



{1, 2, 3, 4, END}

Of course, one can
add attention.

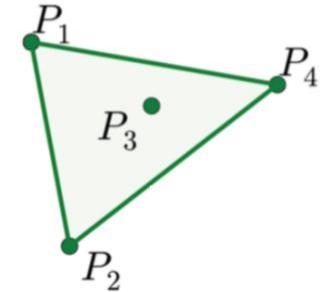


Encoder

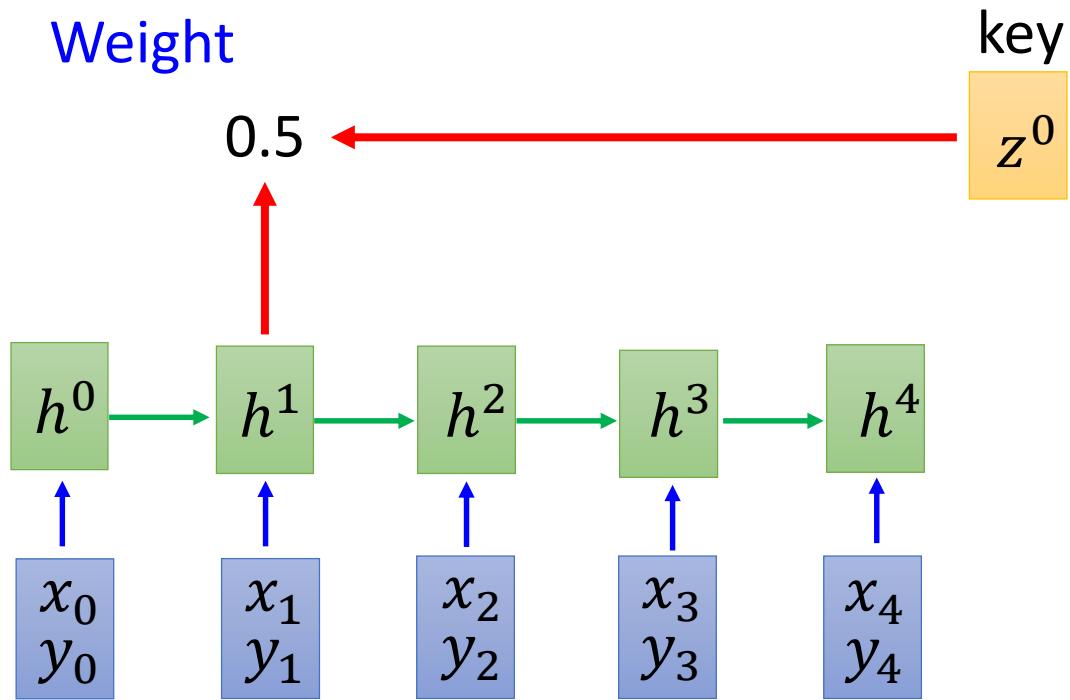
Decoder

Pointer Network

x_0
 y_0 : END

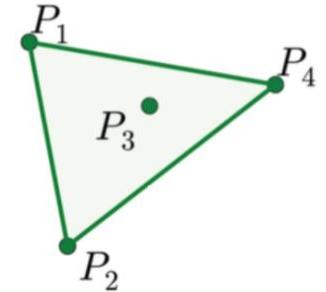


Attention
Weight



Pointer Network

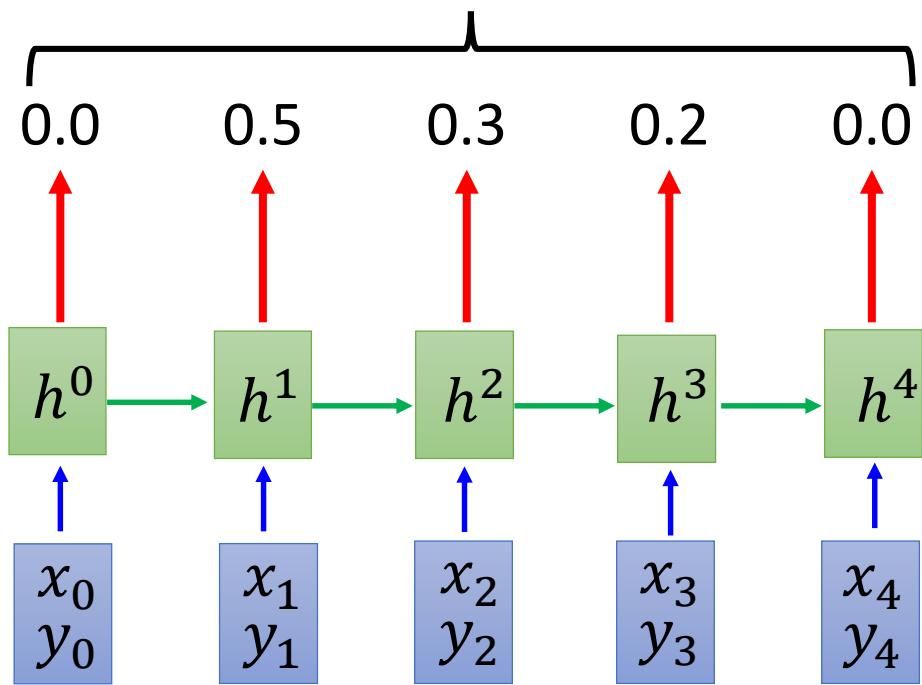
x_0
 y_0 : END



Output: 1

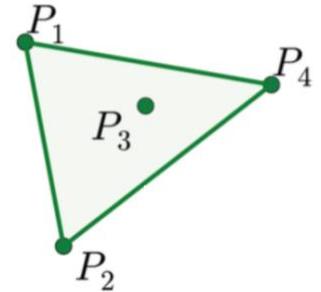
?

argmax from this distribution

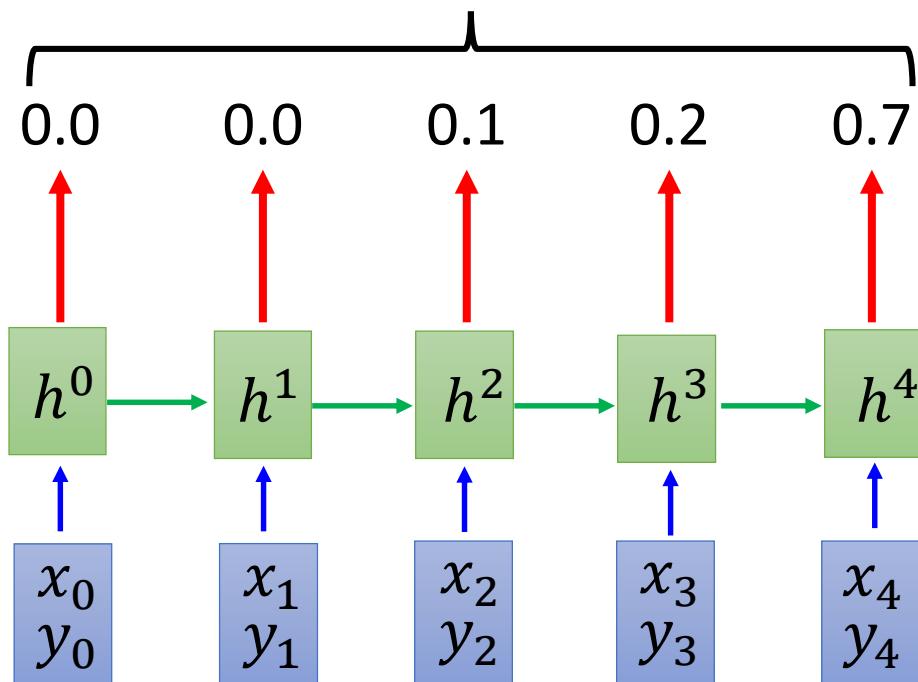


Pointer Network

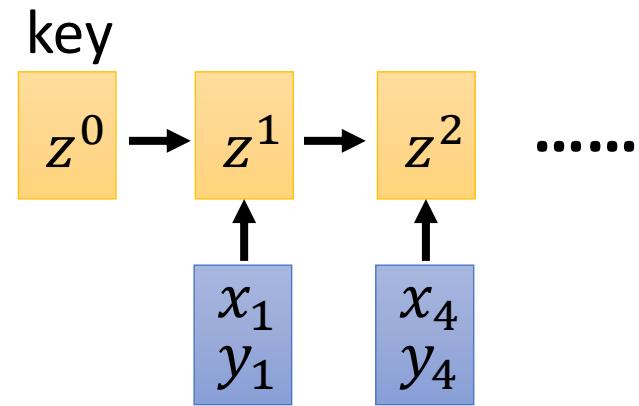
x_0
 y_0 : END



Output: 4
argmax from this distribution

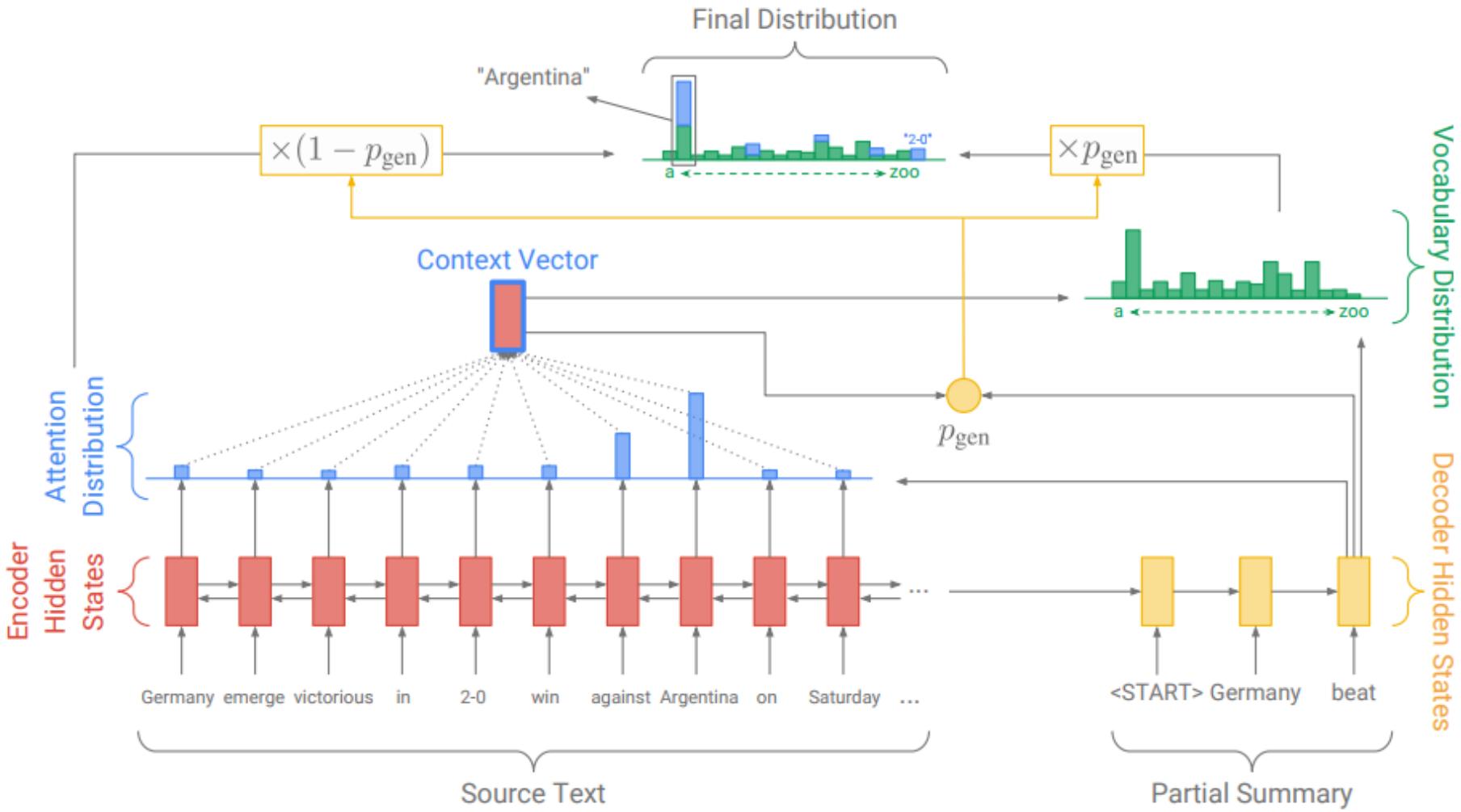


What decoder can output depends on the input.



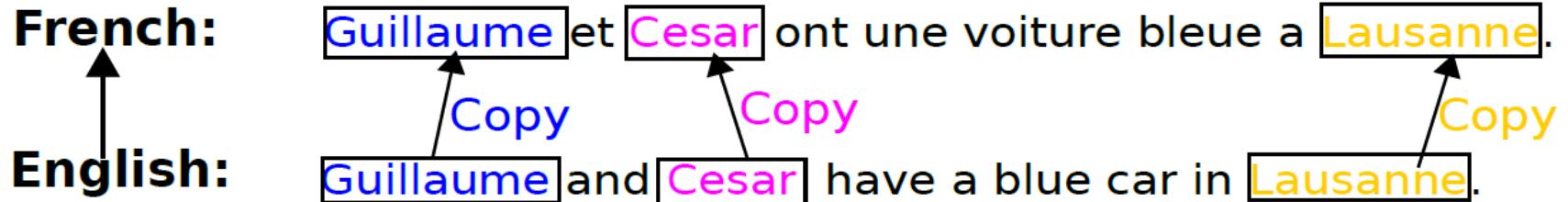
The process stops when “END” has the largest attention weights.

Applications - Summarization

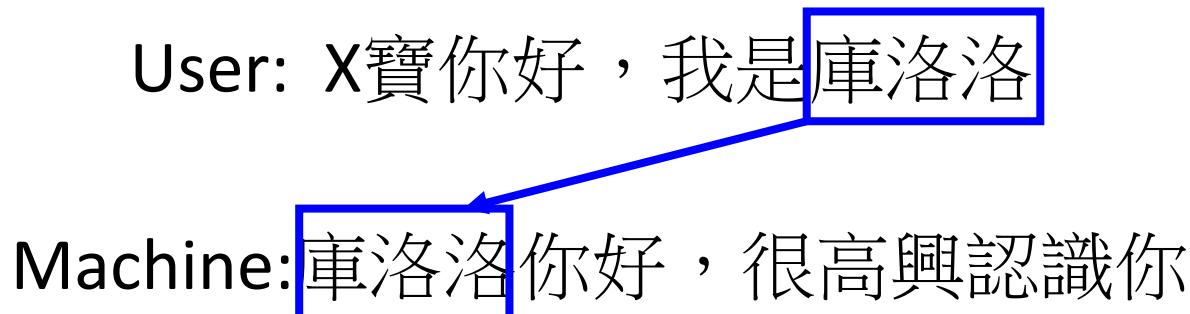


More Applications

Machine Translation



Chat-bot



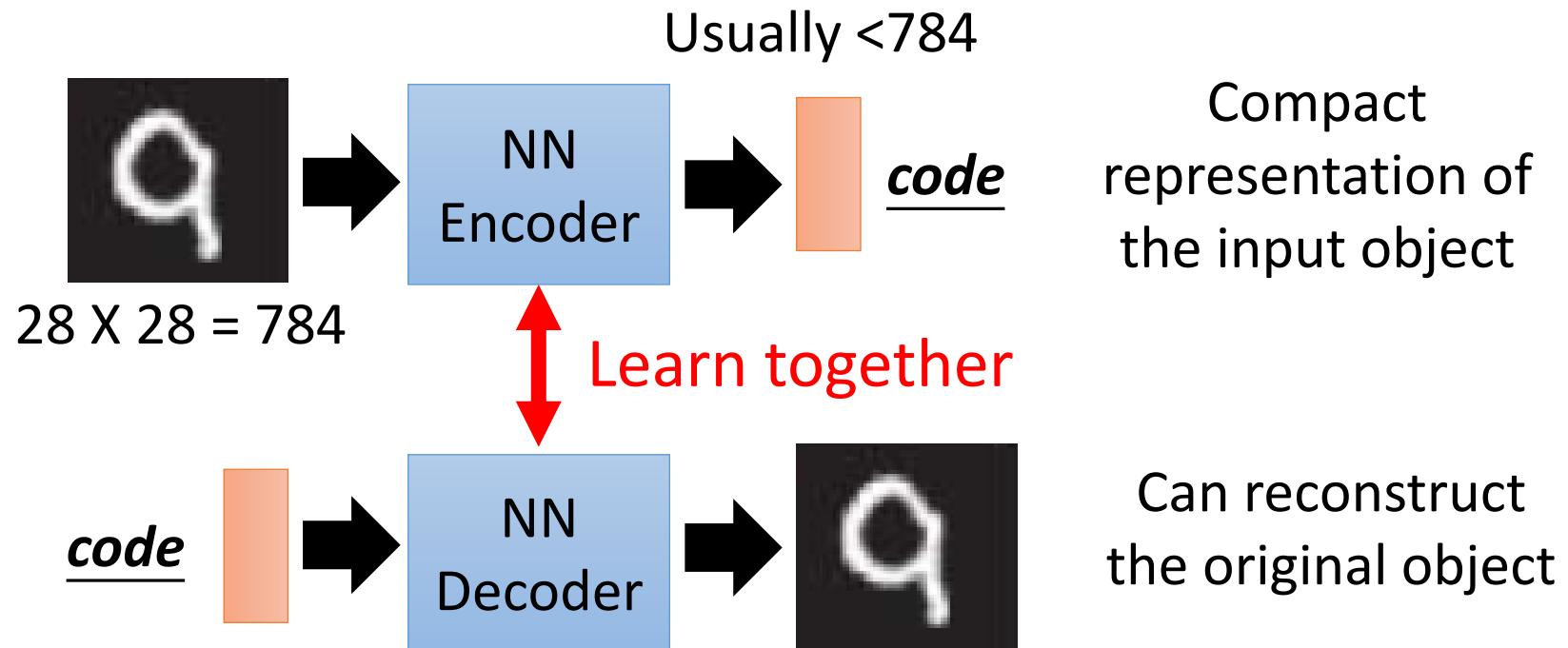
Unsupervised Learning: Deep Auto-encoder

Unsupervised Learning

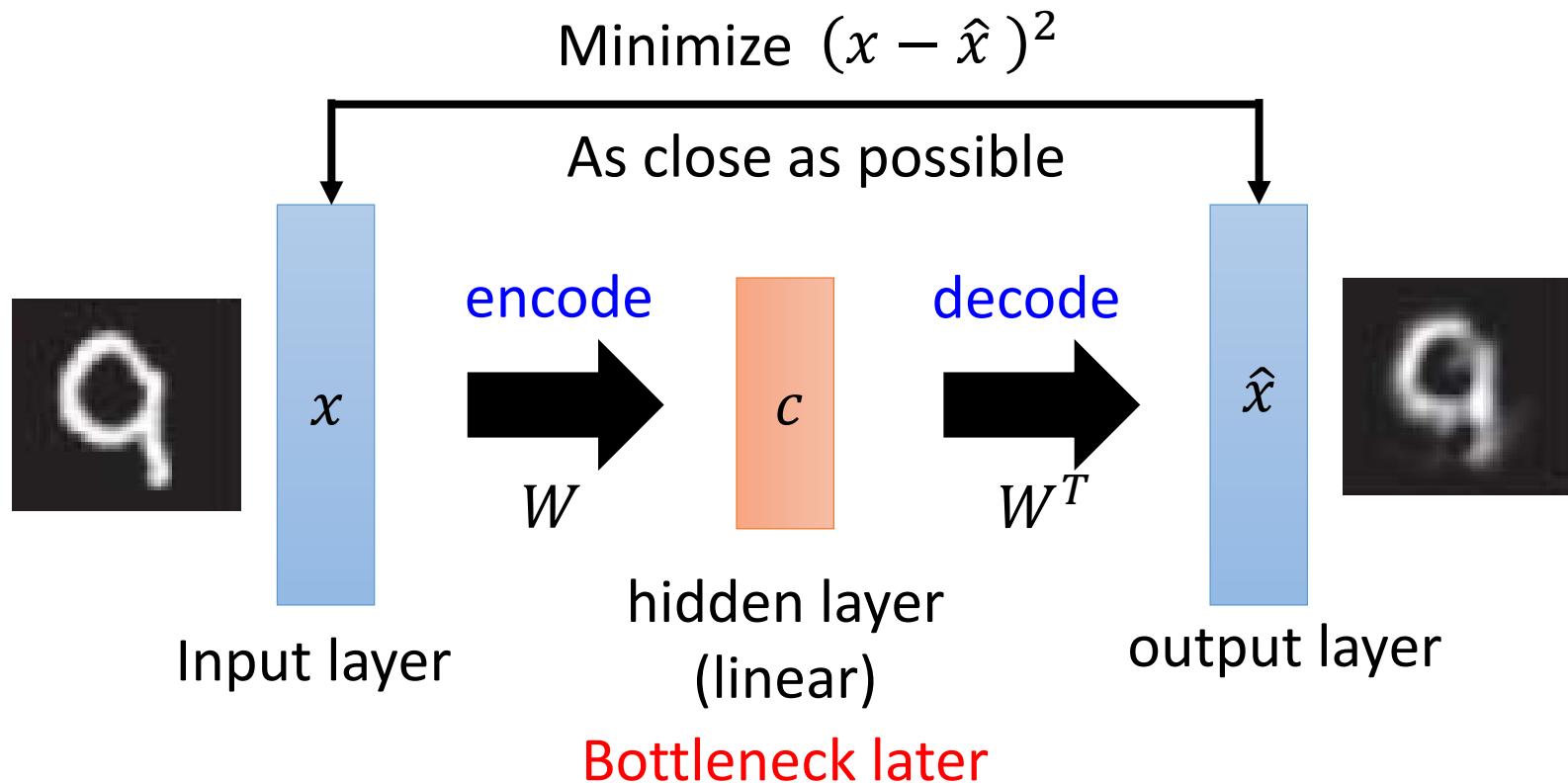
“We expect unsupervised learning to become far more important in the longer term. Human and animal learning is largely unsupervised: we discover the structure of the world by observing it, not by being told the name of every object.”
– LeCun, Bengio, Hinton, Nature 2015

As I've said in previous statements: most of human and animal learning is unsupervised learning. If intelligence was a cake, unsupervised learning would be the cake, supervised learning would be the icing on the cake, and reinforcement learning would be the cherry on the cake. We know how to make the icing and the cherry, but we don't know how to make the cake.
- Yann LeCun, March 14, 2016 (Facebook)

Auto-encoder



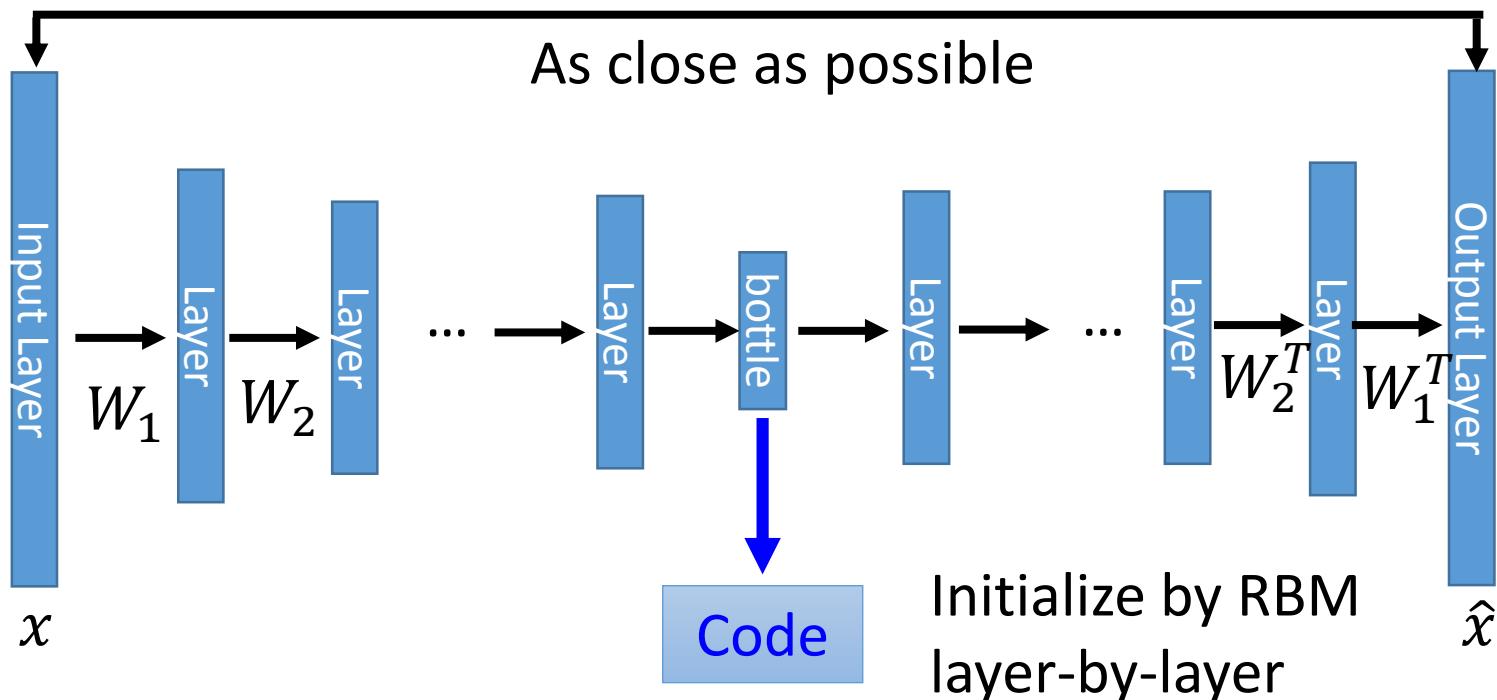
Recap: PCA



Deep Auto-encoder

Symmetric is not necessary.

- Of course, the auto-encoder can be deep



Reference: Hinton, Geoffrey E., and Ruslan R. Salakhutdinov. "Reducing the dimensionality of data with neural networks." *Science* 313.5786 (2006): 504-507

Deep Auto-encoder

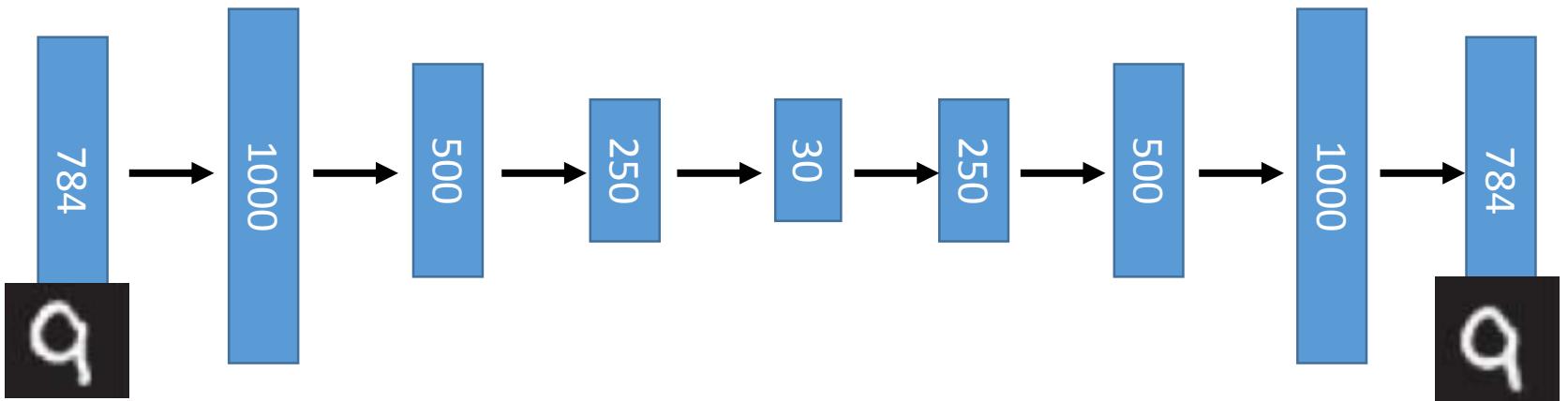
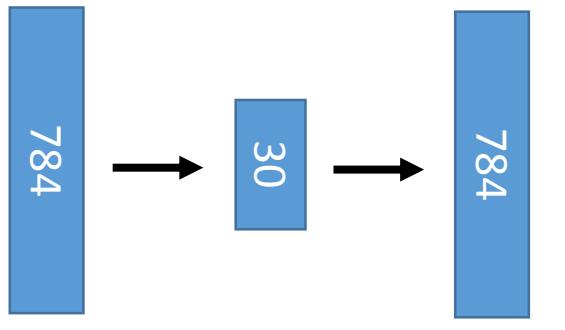
Original
Image

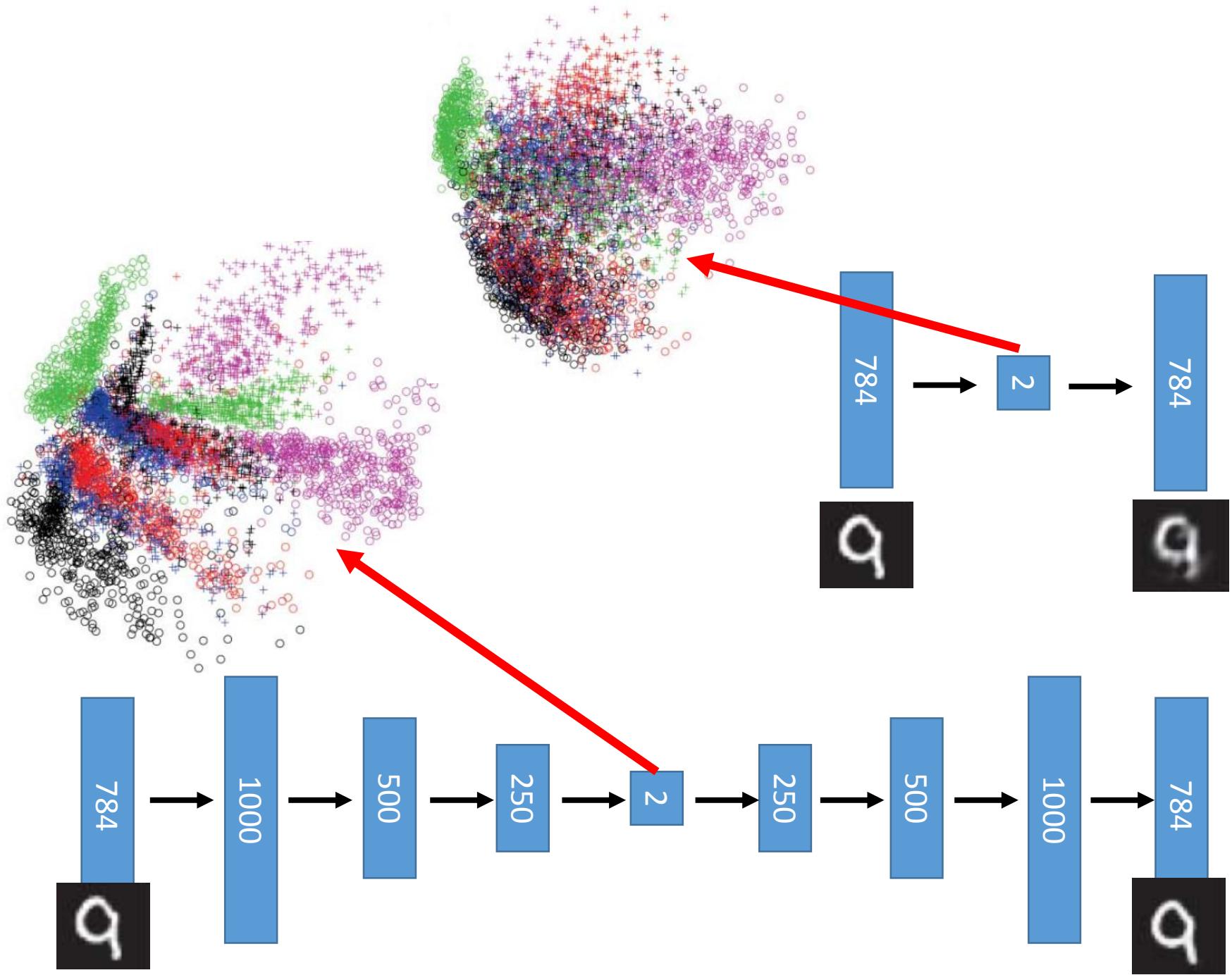


PCA



Deep
Auto-encoder



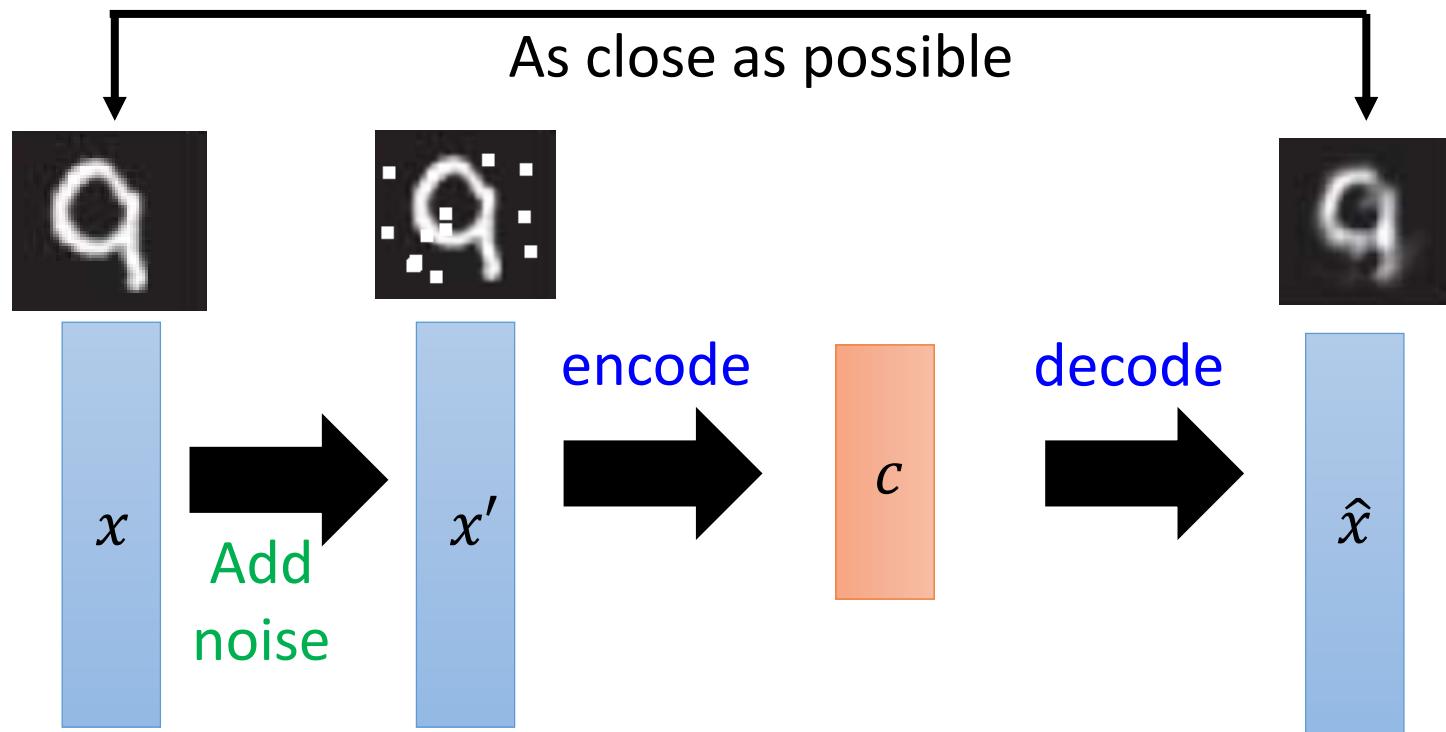


Auto-encoder

- De-noising auto-encoder

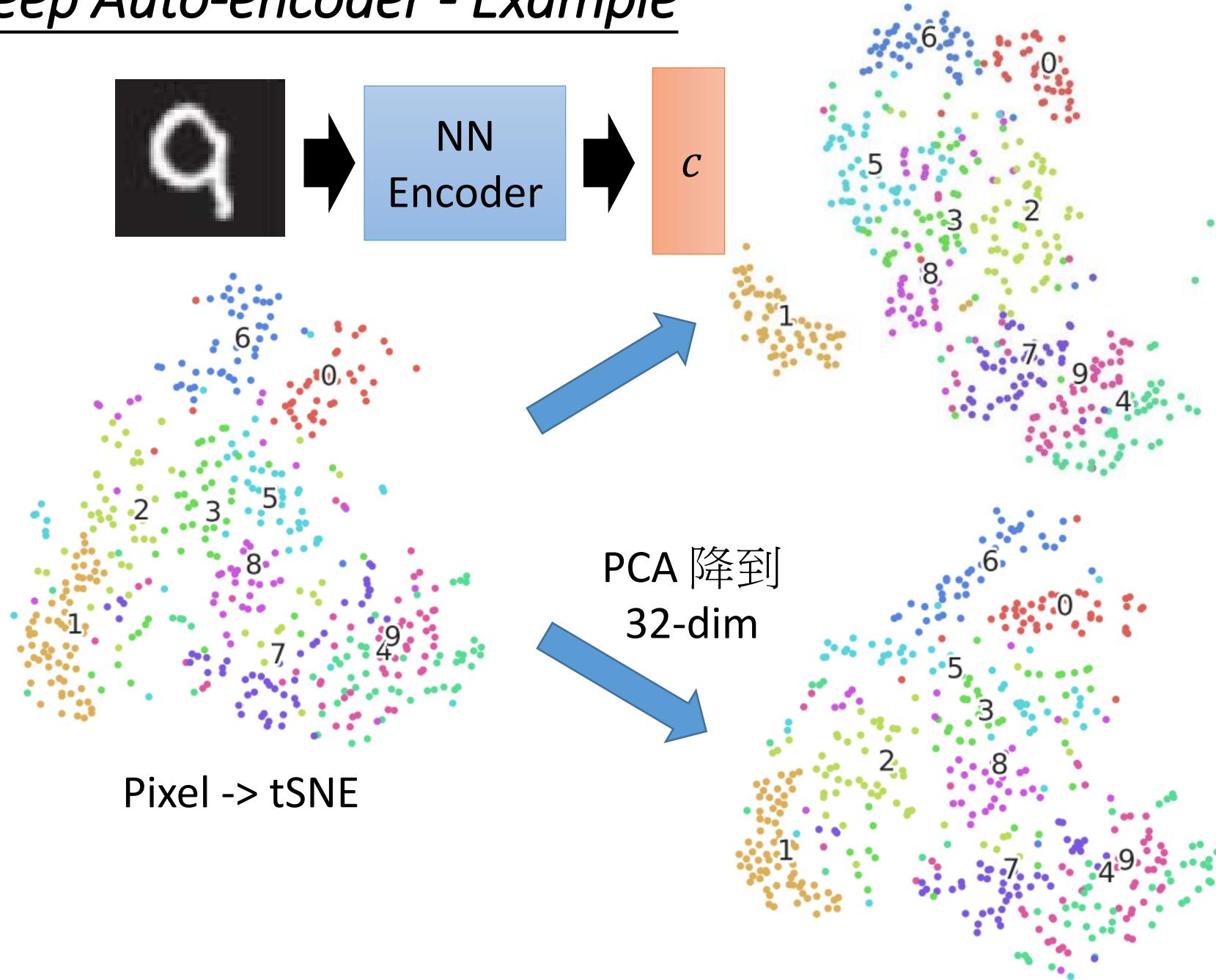
More: Contractive auto-encoder

Ref: Rifai, Salah, et al. "Contractive auto-encoders: Explicit invariance during feature extraction." *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*. 2011.

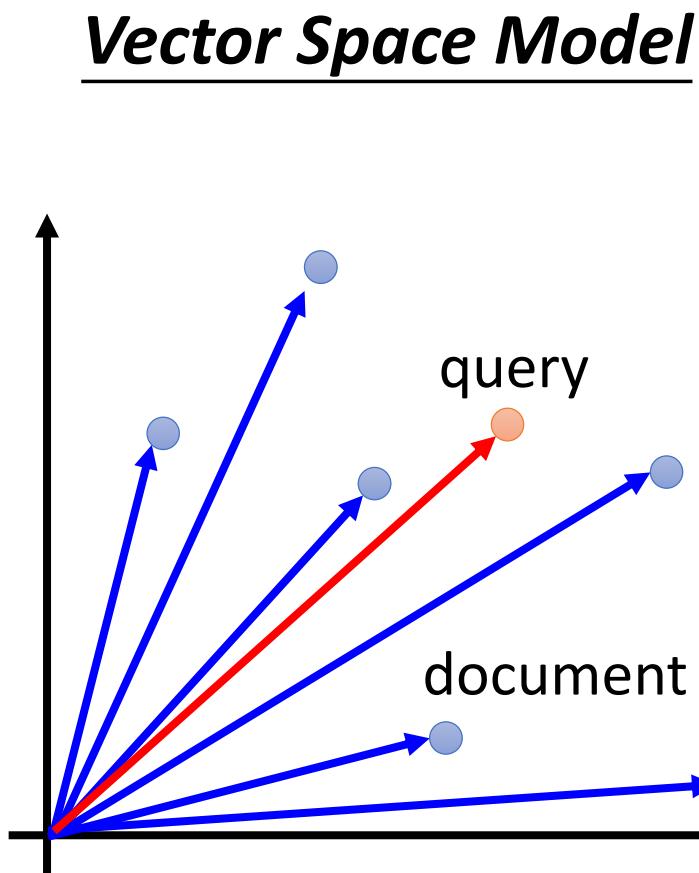


Vincent, Pascal, et al. "Extracting and composing robust features with denoising autoencoders." *ICML*, 2008.

Deep Auto-encoder - Example



Auto-encoder – Text Retrieval



Bag-of-word

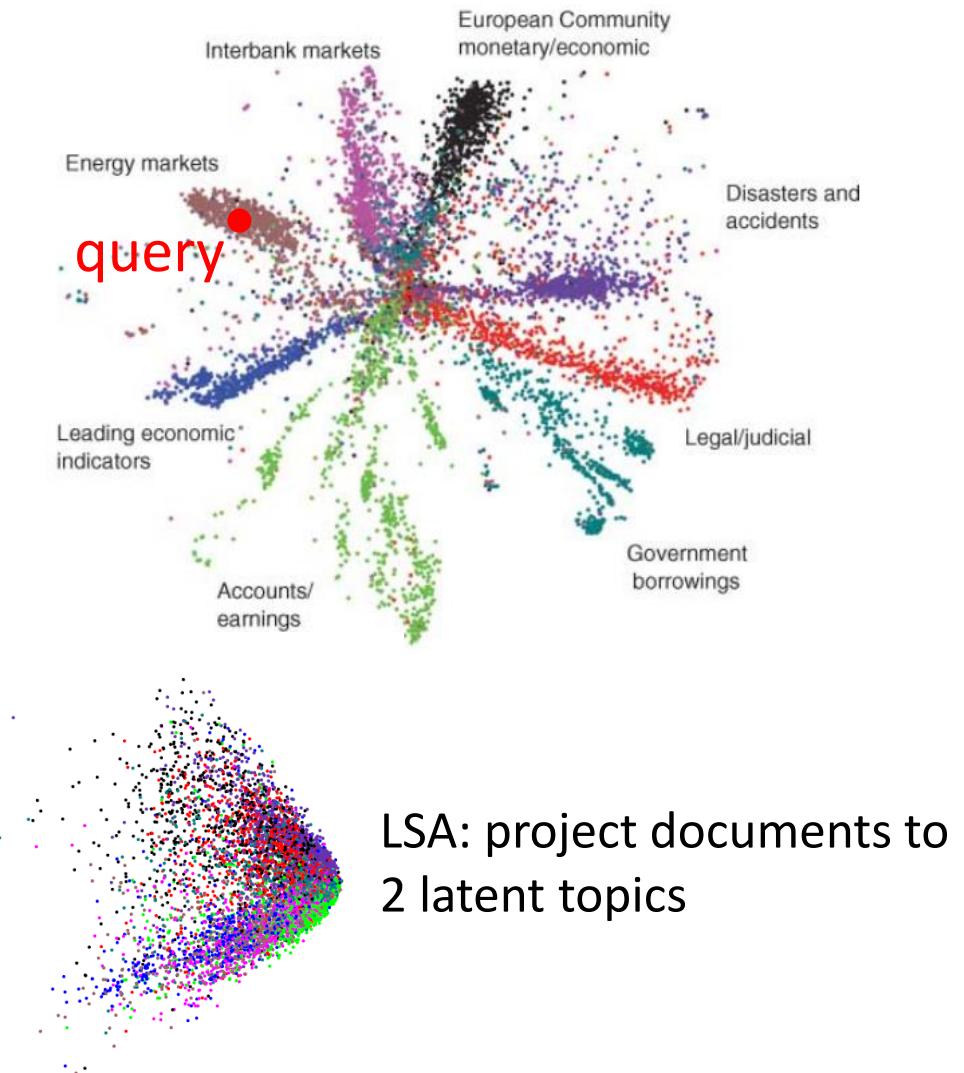
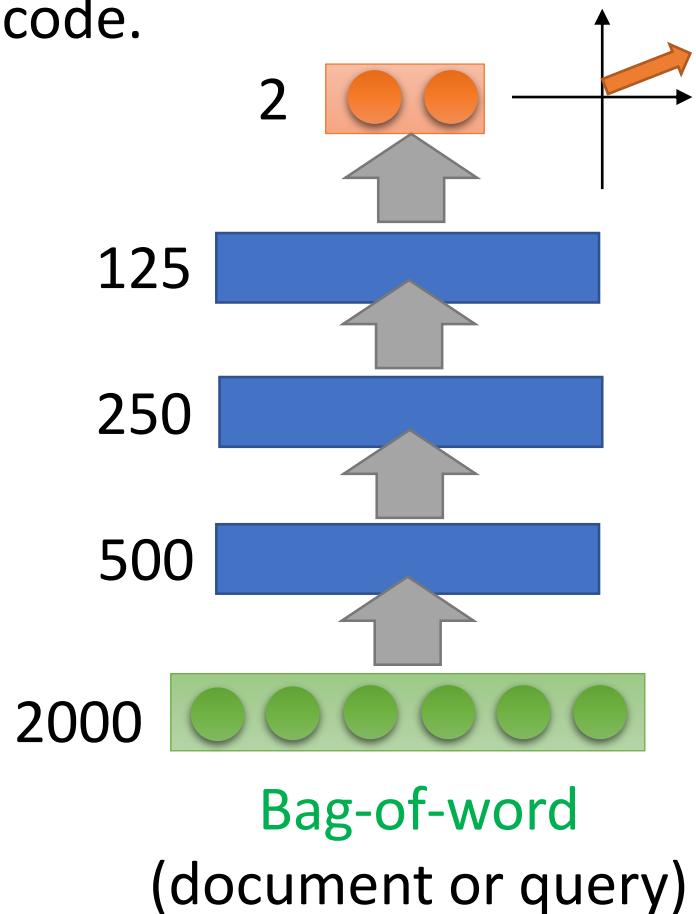
word string:
“This is an apple”

this	1
is	1
a	0
an	1
apple	1
pen	0
:	

Semantics are not considered.

Auto-encoder – Text Retrieval

The documents talking about the same thing will have close code.



LSA: project documents to
2 latent topics

Auto-encoder – Similar Image Search

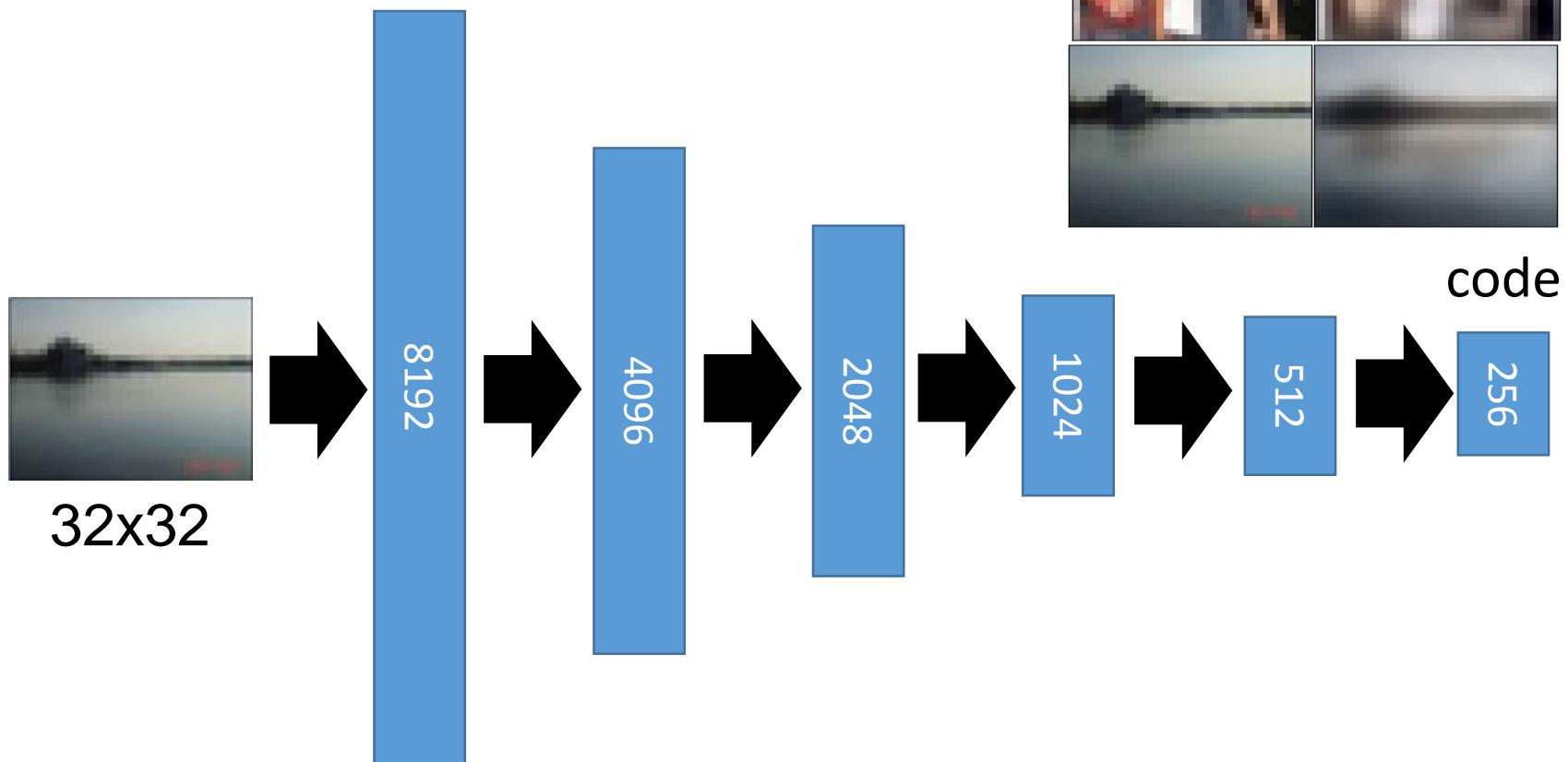
Retrieved using Euclidean distance in pixel intensity space



(Images from Hinton's slides on Coursera)

Reference: Krizhevsky, Alex, and Geoffrey E. Hinton. "Using very deep autoencoders for content-based image retrieval." *ESANN*. 2011.

Auto-encoder – Similar Image Search



(crawl millions of images from the Internet)

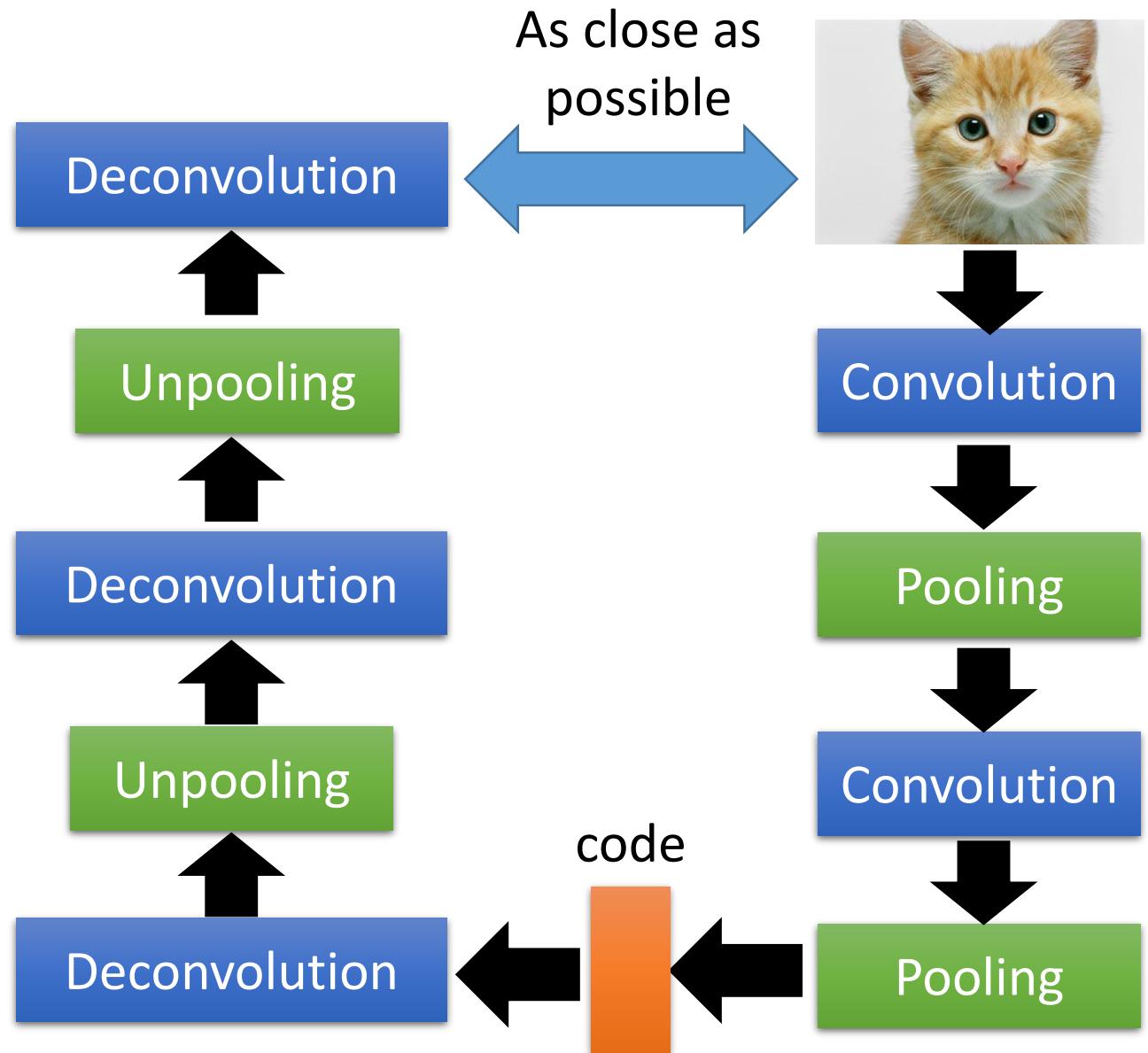
Retrieved using Euclidean distance in pixel intensity space



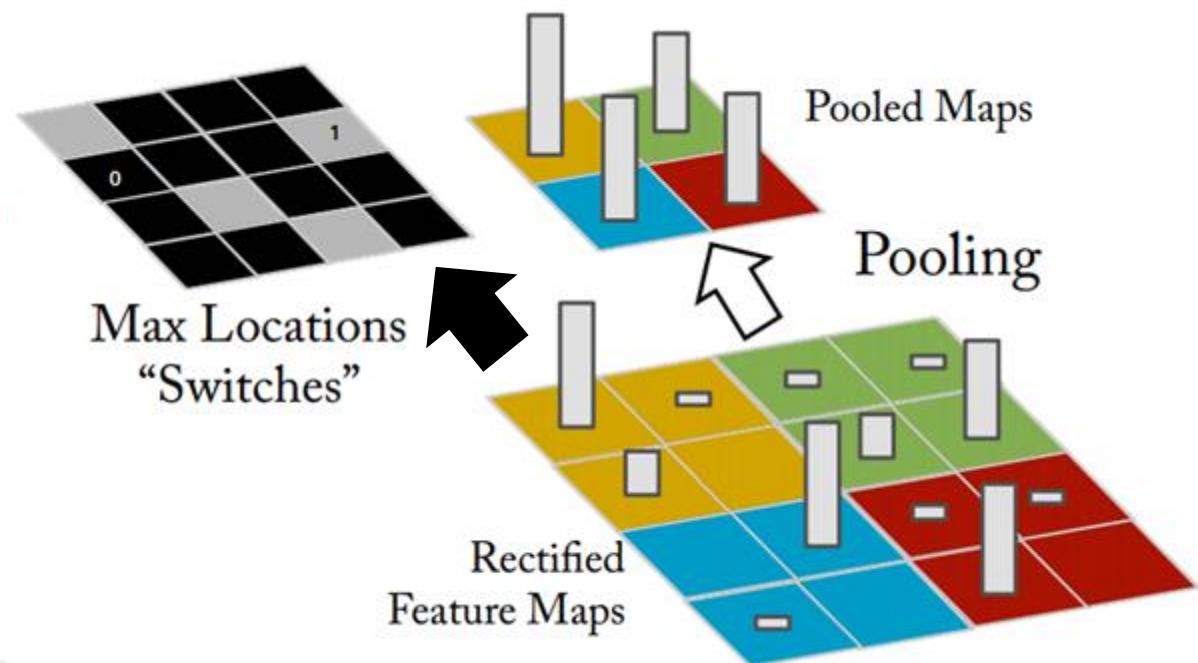
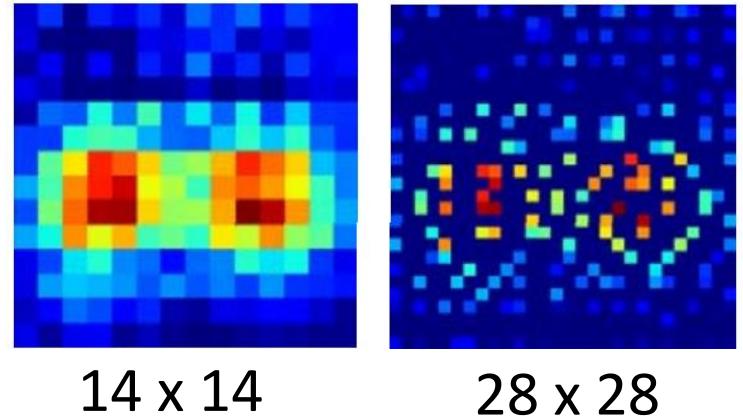
retrieved using 256 codes



Auto-encoder for CNN



CNN -Unpooling



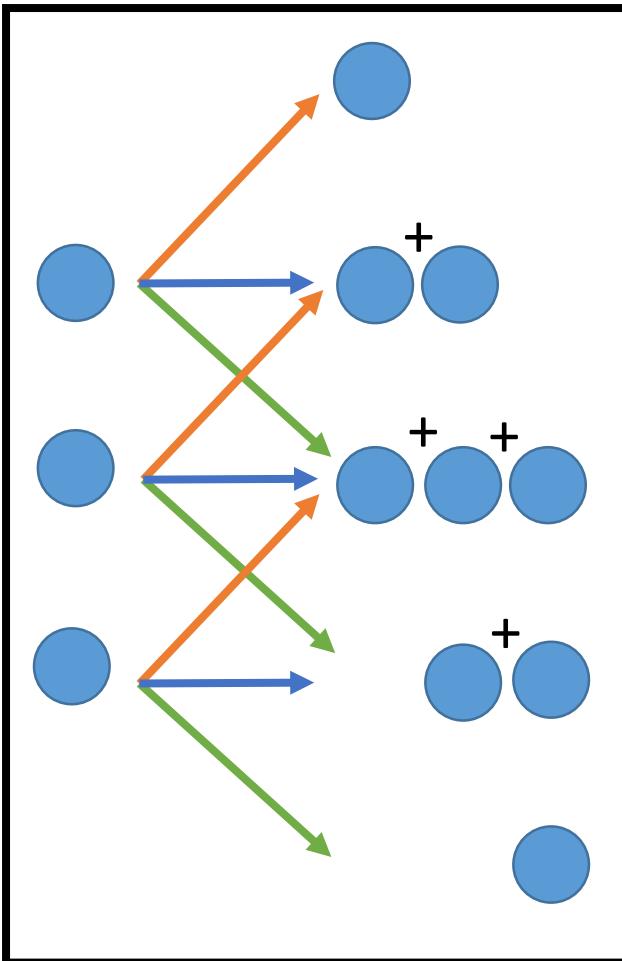
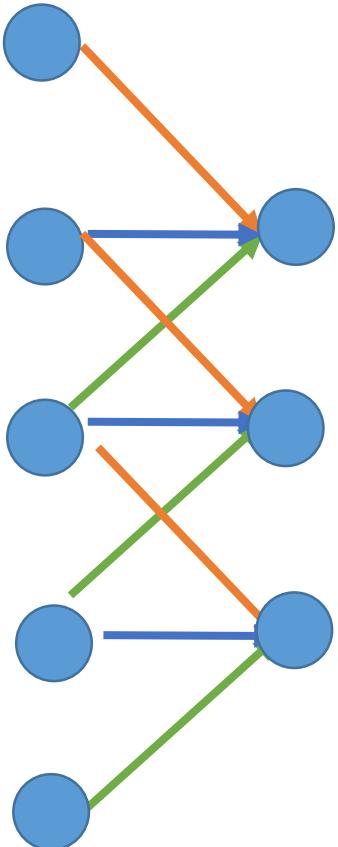
Alternative: simply repeat the values

Source of image :
https://leonardoaraujosantos.gitbooks.io/artificial-intelligence/content/image_segmentation.html

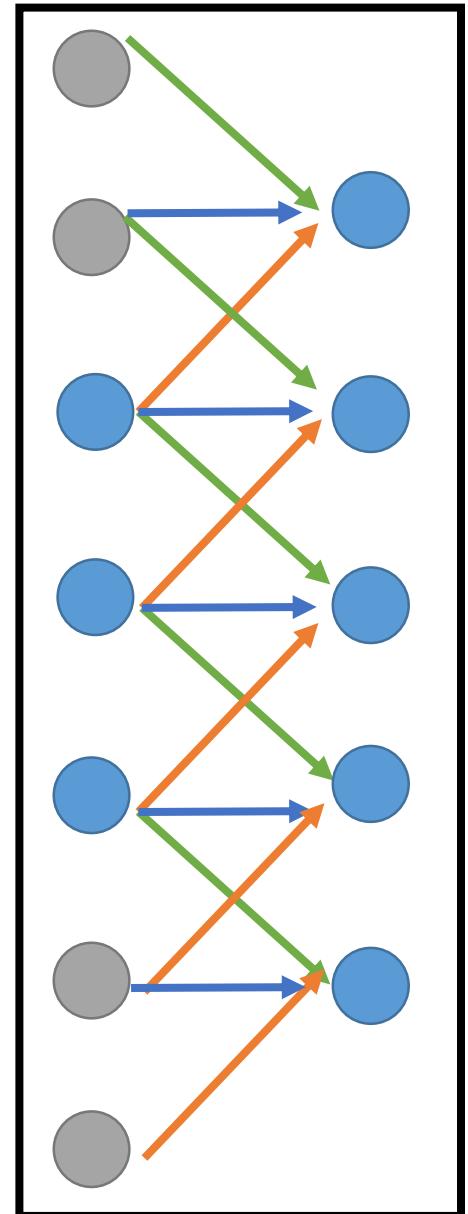
CNN

- Deconvolution

Actually, deconvolution is convolution.

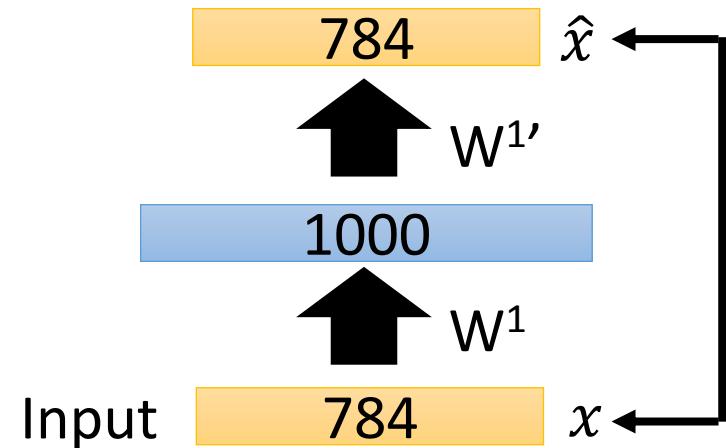
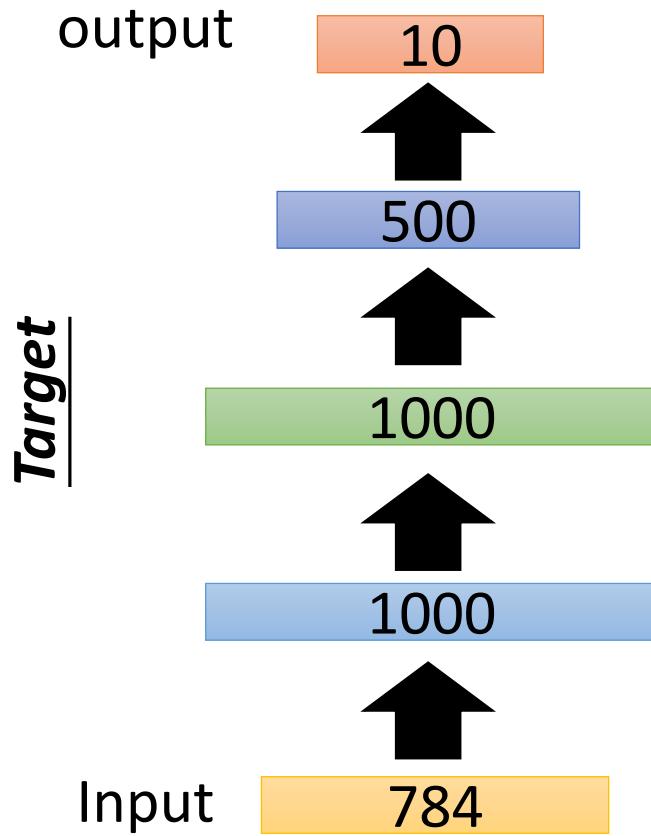


=



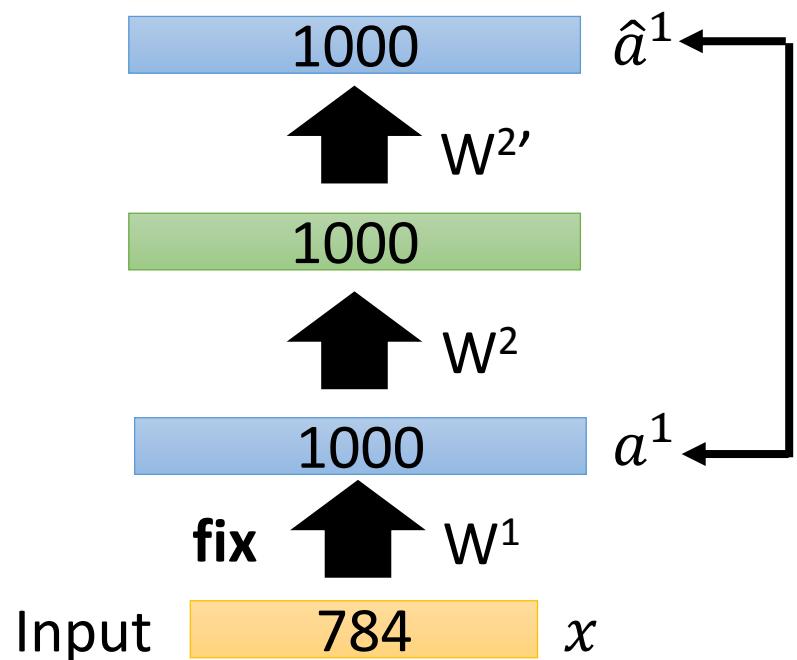
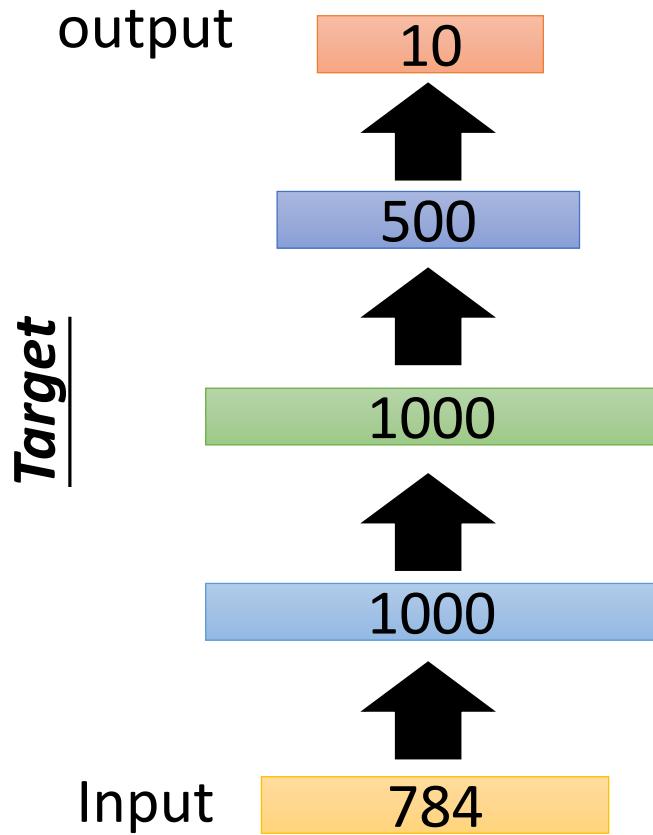
Auto-encoder – Pre-training DNN

- Greedy Layer-wise Pre-training *again*



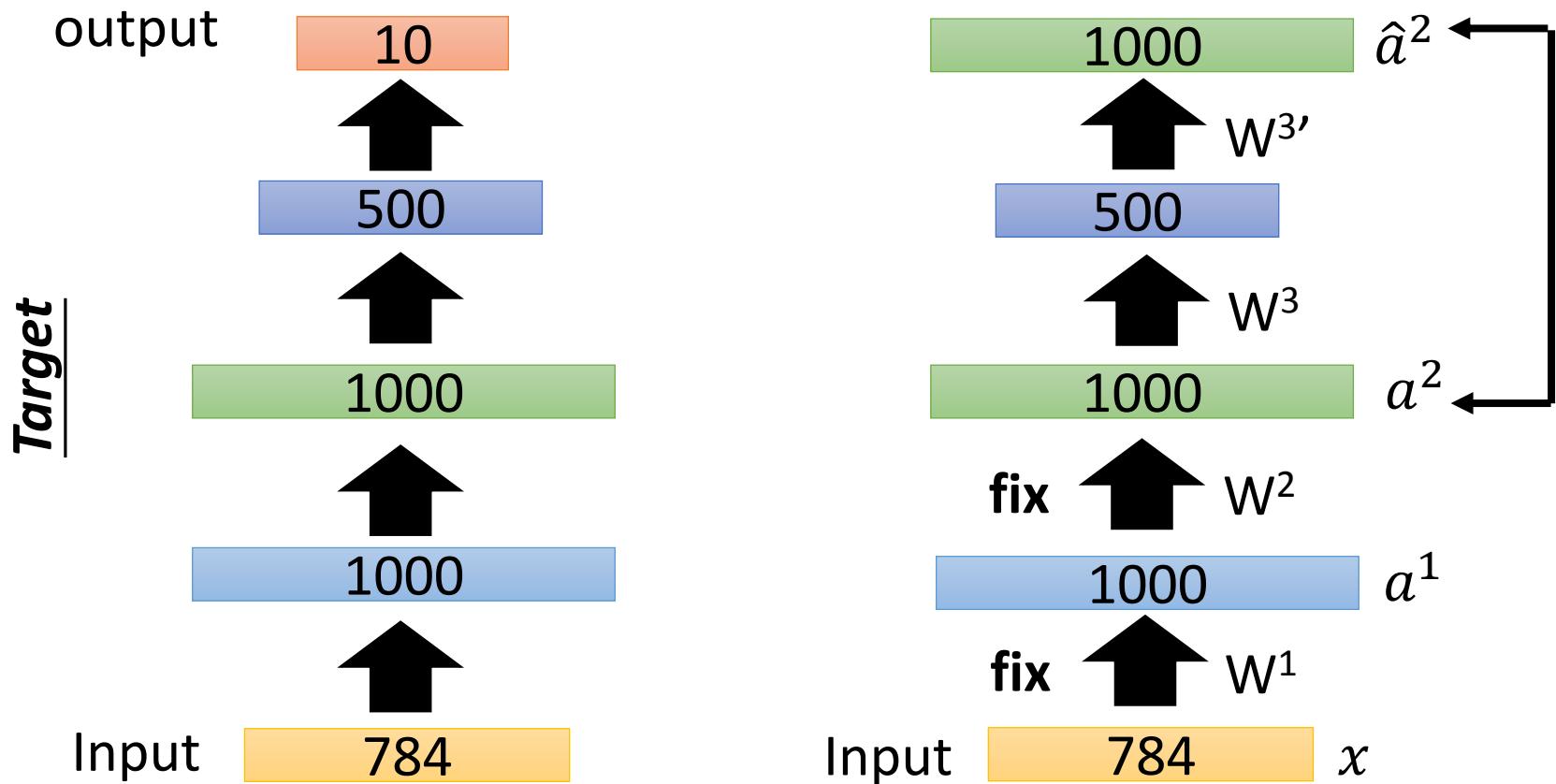
Auto-encoder – Pre-training DNN

- Greedy Layer-wise Pre-training *again*



Auto-encoder – Pre-training DNN

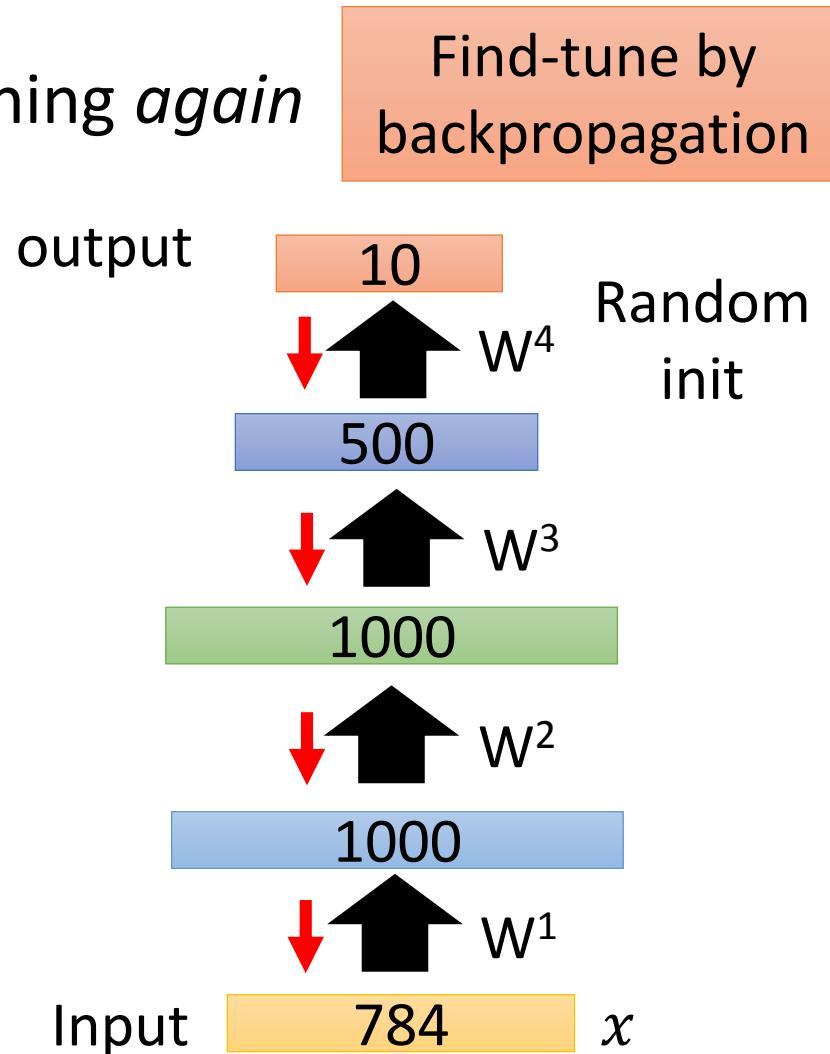
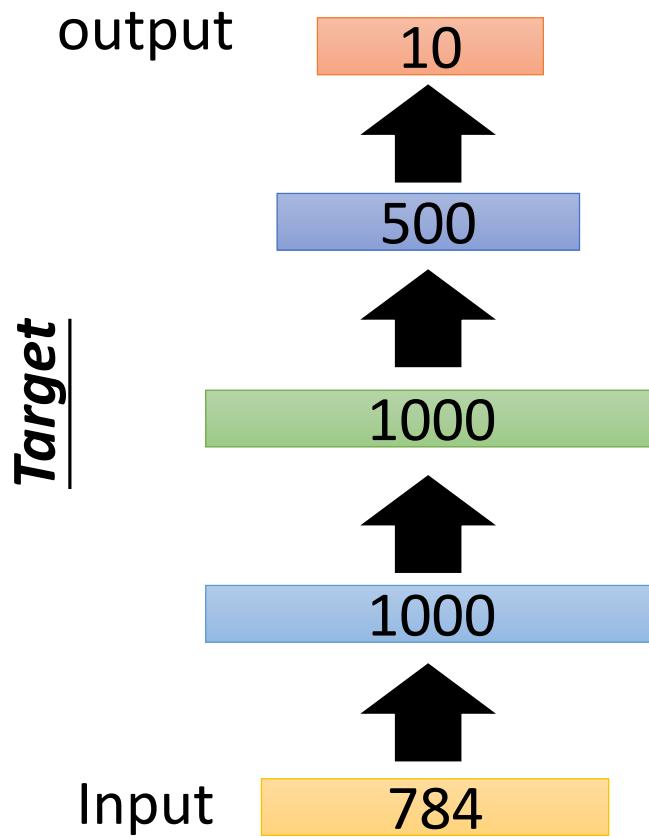
- Greedy Layer-wise Pre-training *again*



Auto-encoder – Pre-training DNN

- Greedy Layer-wise Pre-training *again*

Find-tune by
backpropagation



Learning More

- Restricted Boltzmann Machine

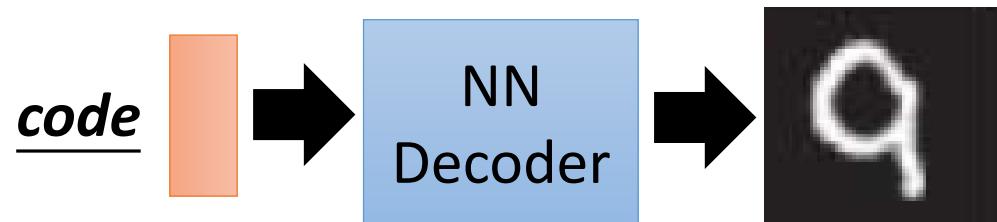
- Neural networks [5.1] : Restricted Boltzmann machine – definition
 - https://www.youtube.com/watch?v=p4Vh_zMw-HQ&index=36&list=PL6Xpj9I5qXYEcOhn7TqghAJ6NAPrNmUBH
- Neural networks [5.2] : Restricted Boltzmann machine – inference
 - https://www.youtube.com/watch?v=lekCh_i32iE&list=PL6Xpj9I5qXYEcOhn7TqghAJ6NAPrNmUBH&index=37
- Neural networks [5.3] : Restricted Boltzmann machine - free energy
 - https://www.youtube.com/watch?v=e0Ts_7Y6hZU&list=PL6Xpj9I5qXYEcOhn7TqghAJ6NAPrNmUBH&index=38

Learning More

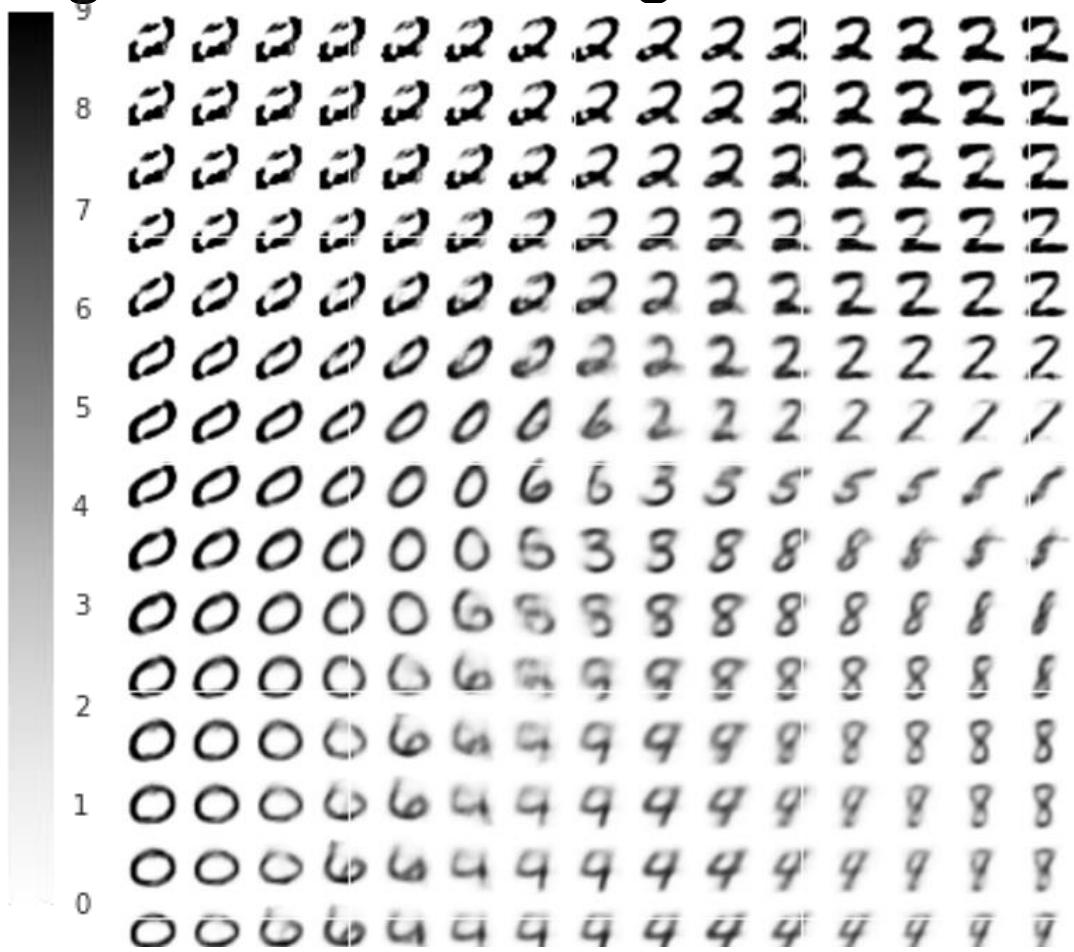
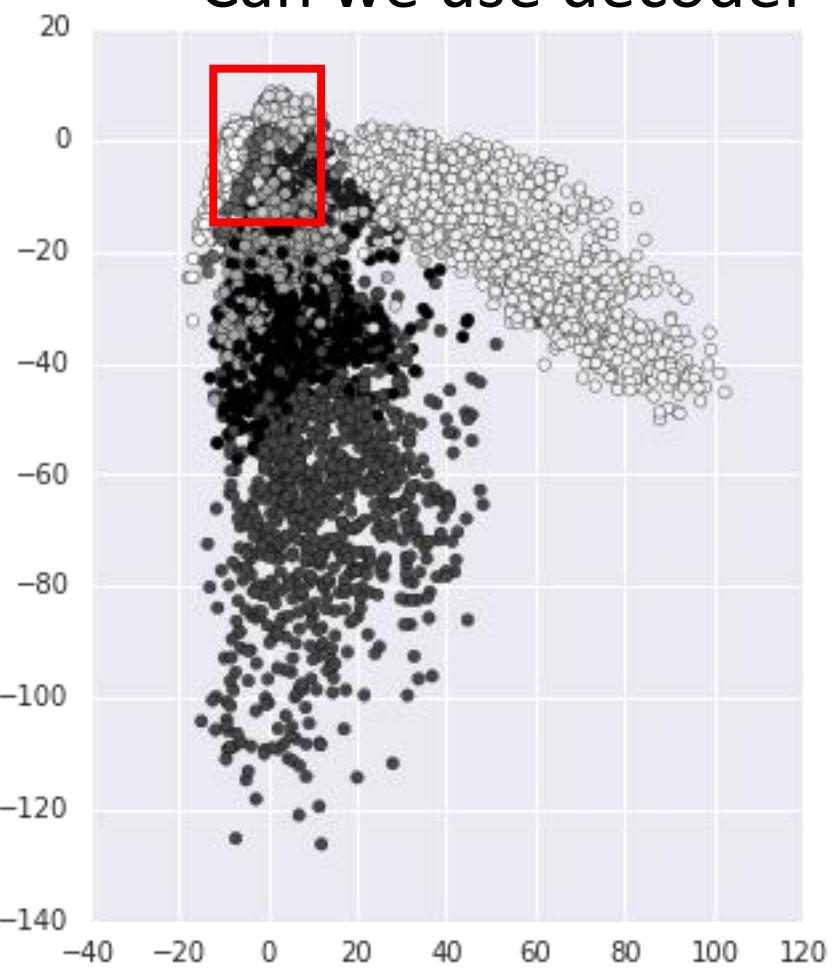
- Deep Belief Network

- Neural networks [7.7] : Deep learning - deep belief network
 - <https://www.youtube.com/watch?v=vkb6AWYZ5I&list=PL6Xpj9I5qXYEcOhn7TqghAJ6NAPrNmUBH&index=57>
- Neural networks [7.8] : Deep learning - variational bound
 - <https://www.youtube.com/watch?v=pStDscJh2Wo&list=PL6Xpj9I5qXYEcOhn7TqghAJ6NAPrNmUBH&index=58>
- Neural networks [7.9] : Deep learning - DBN pre-training
 - <https://www.youtube.com/watch?v=35MUIYCColk&list=PL6Xpj9I5qXYEcOhn7TqghAJ6NAPrNmUBH&index=59>

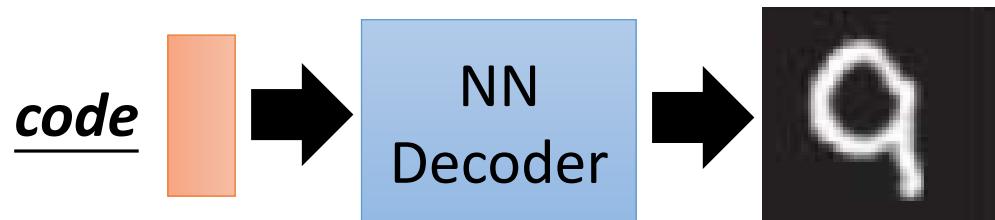
Next



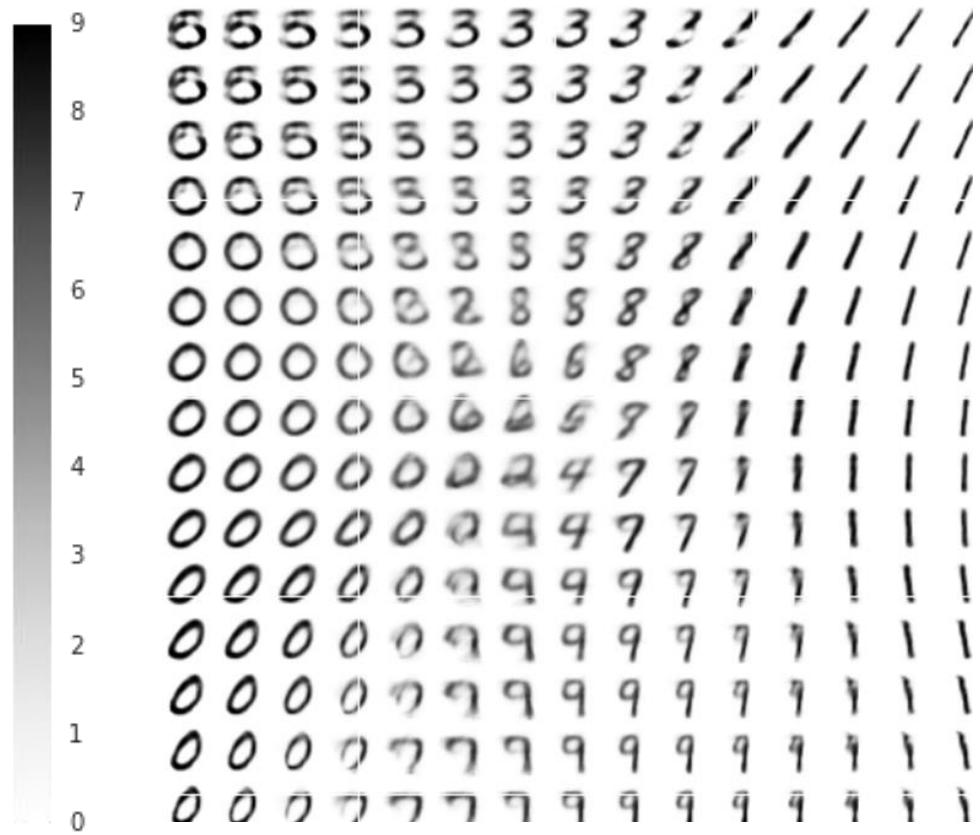
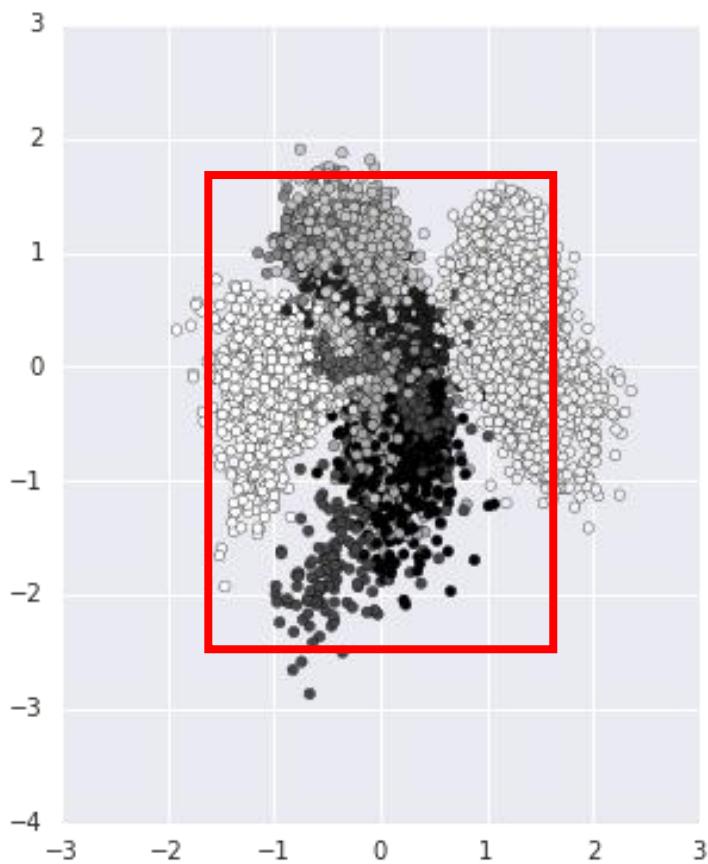
- Can we use decoder to generate something?



Next



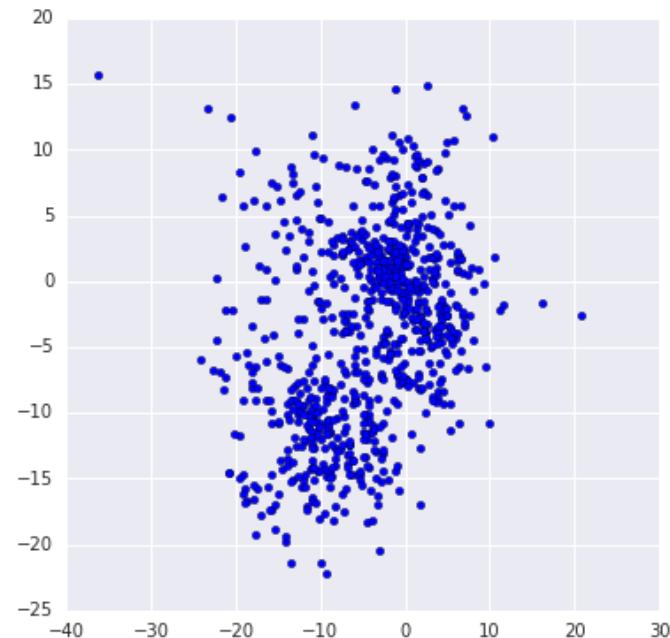
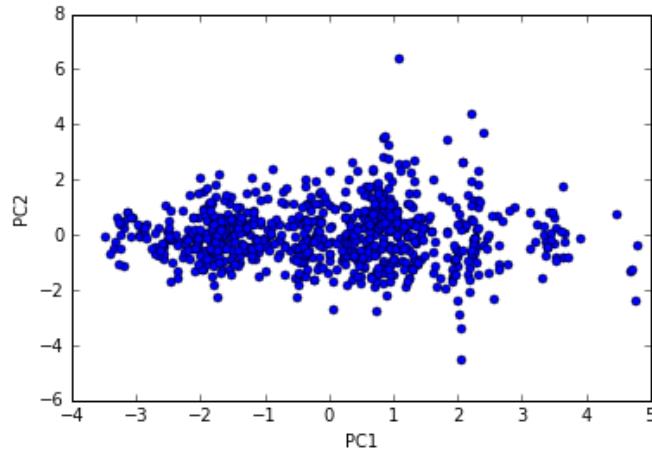
- Can we use decoder to generate something?



Appendix

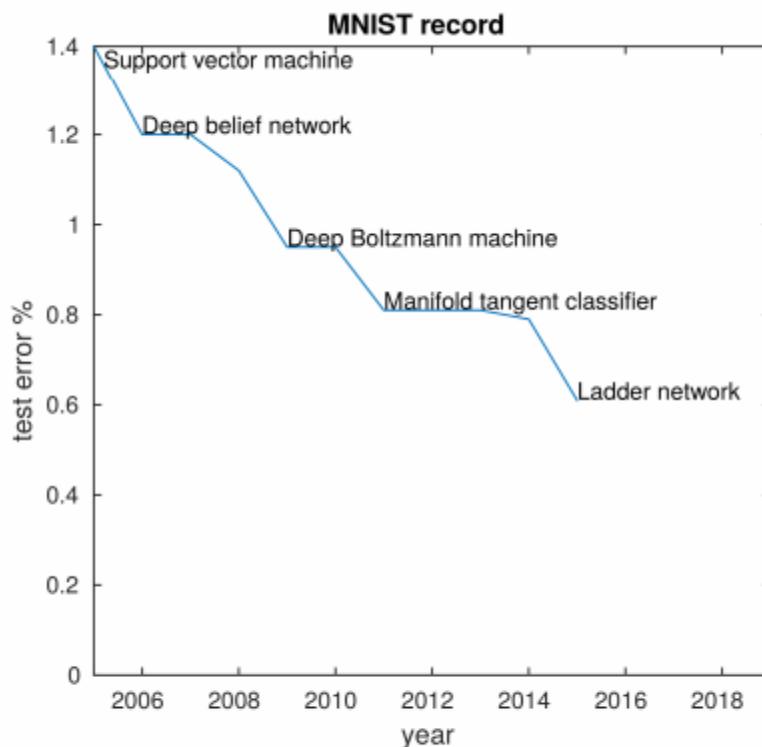
Pokémon

- <http://140.112.21.35:2880/~tlkagk/pokemon/pca.html>
- <http://140.112.21.35:2880/~tlkagk/pokemon/auto.html>
- The code is modified from
 - <http://jkunst.com/r/pokemon-visualize-em-all/>



Add: Ladder Network

- <http://rinuboney.github.io/2016/01/19/ladder-network.html>
- https://mycourses.aalto.fi/pluginfile.php/146701/mod_resource/content/1/08%20semisup%20ladder.pdf
- <https://arxiv.org/abs/1507.02672>



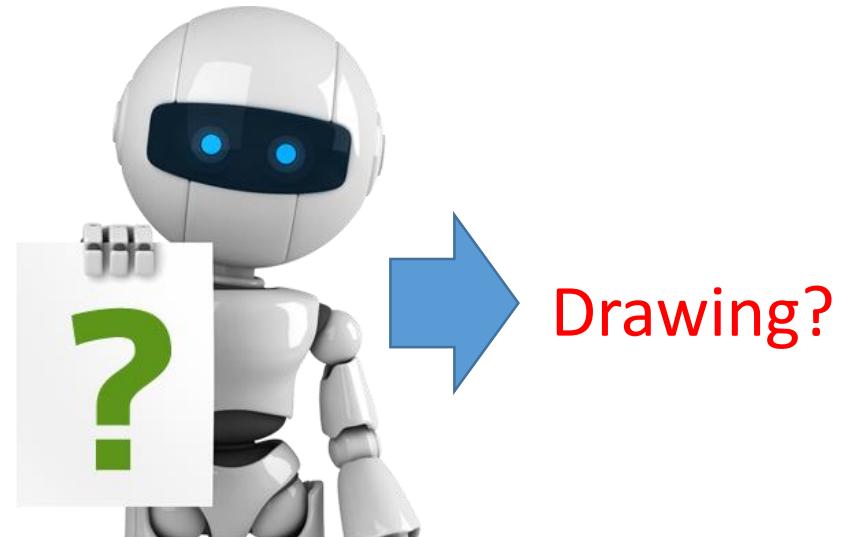
Yearly progress in permutation-invariant MNIST.

A. Rasmus, H. Valpola, M. Honkala, M. Berglund, and T. Raiko.

Semi-Supervised Learning with Ladder Network. To appear in NIPS 2015.

Unsupervised Learning: Generation

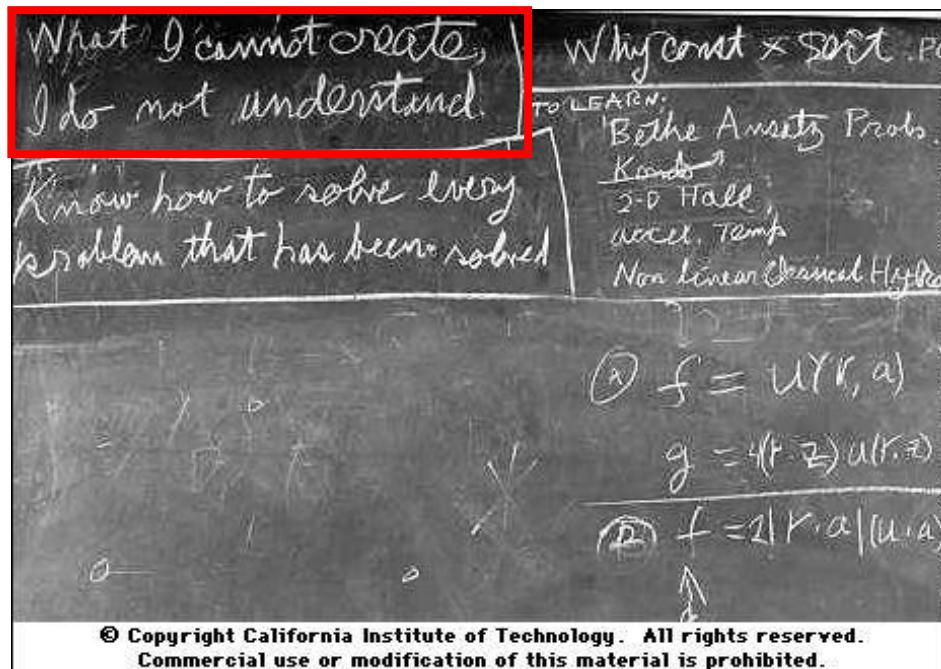
Creation



Creation

- Generative Models:

<https://openai.com/blog/generative-models/>



What I cannot create,
I do not understand.

Richard Feynman

<https://www.quora.com/What-did-Richard-Feynman-mean-when-he-said-What-I-cannot-create-I-do-not-understand>

Creation

Now



v.s.



In the future

Machine
draws a cat



<http://www.wikihow.com/Draw-a-Cat-Face>

Generative Models

Component-by-component

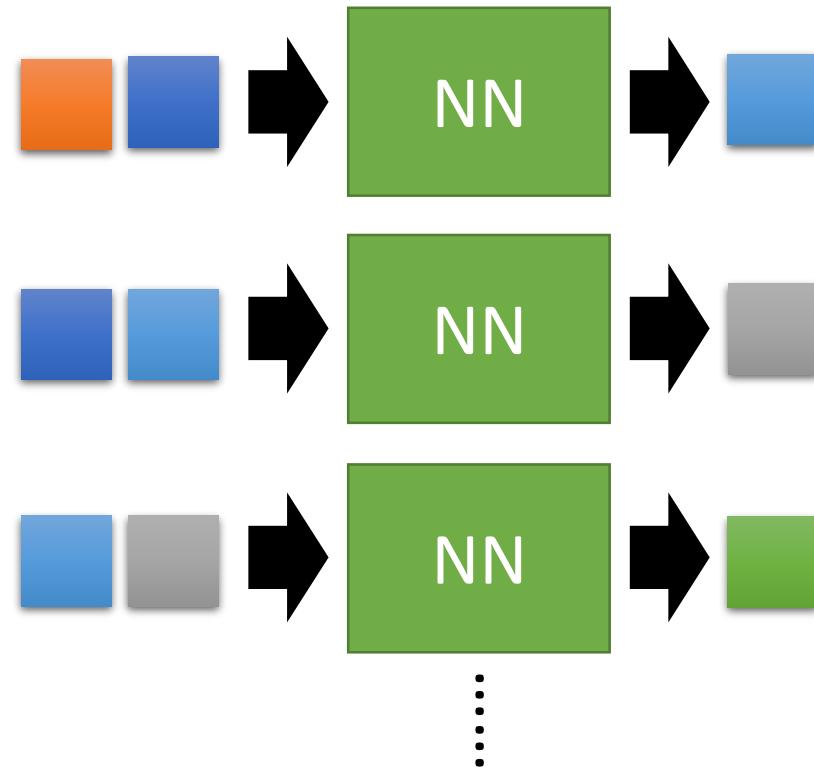
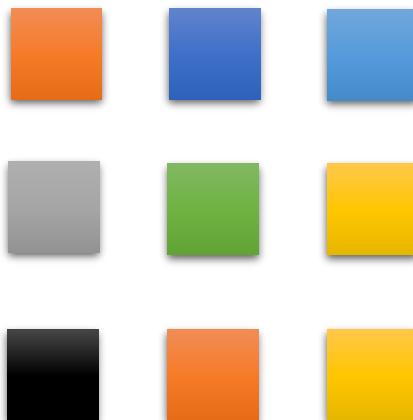
Autoencoder

Generative Adversarial Network
(GAN)

Component-by-component

- Image generation

E.g. 3×3 images

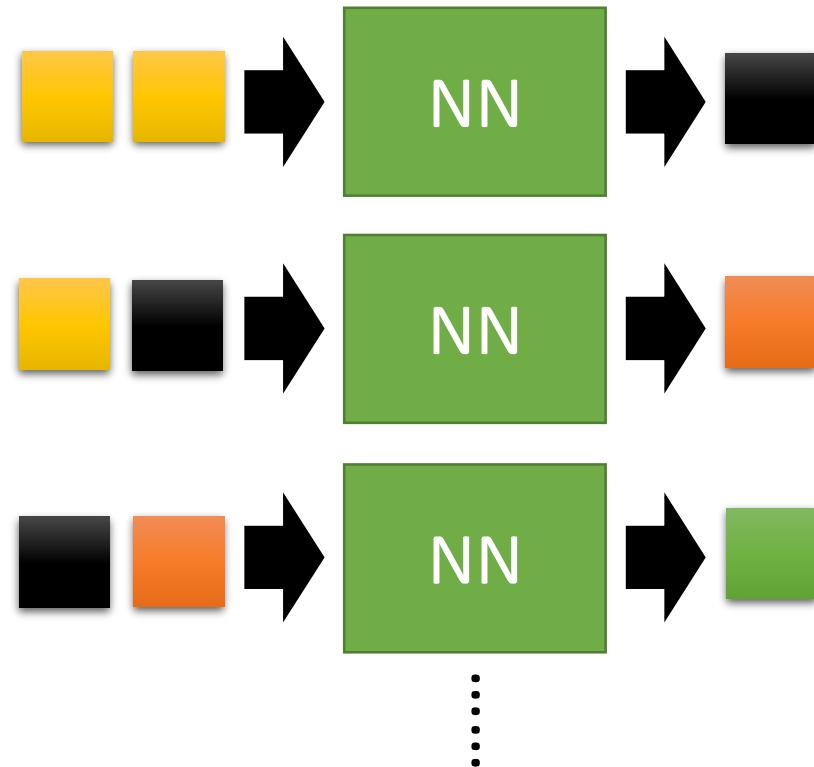
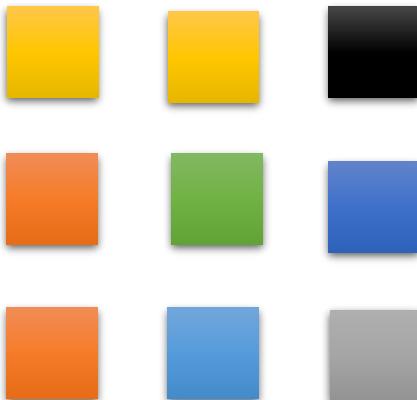


Can be trained just with a large collection of images
without any annotation

Component-by-component

- Image generation

E.g. 3×3 images



Can be trained just with a large collection of images without any annotation

Practicing Generation Models: Pokémon Creation

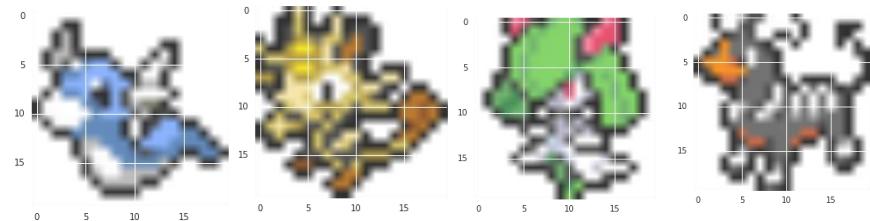
- Small images of 792 Pokémon's
 - Can machine learn to create new Pokémons?

Don't catch them! Create them!

- Source of image:
[http://bulbapedia.bulbagarden.net/wiki/List_of_Pok%C3%A9mon_by_base_stats_\(Generation_VI\)](http://bulbapedia.bulbagarden.net/wiki/List_of_Pok%C3%A9mon_by_base_stats_(Generation_VI))

Original image is 40 x 40

Making them into 20 x 20



Practicing Generation Models: Pokémon Creation

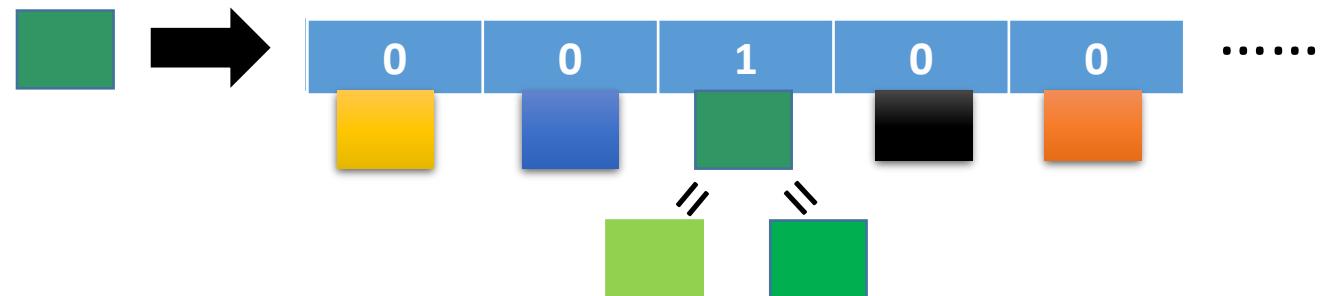
- Tips (?)

- Each pixel is represented by 3 numbers (corresponding to RGB)



R=50, G=150, B=100

- Each pixel is represented by a 1-of-N encoding feature



Clustering the similar color → 167 colors in total

Practicing Generation Models: Pokémon Creation

- Original image (40 x 40):
http://speech.ee.ntu.edu.tw/~tlkagk/courses/ML_2016/Pokemon_creation/image.rar
- Pixels (20 x 20):
http://speech.ee.ntu.edu.tw/~tlkagk/courses/ML_2016/Pokemon_creation/pixel_color.txt
 - Each line corresponds to an image, and each number corresponds to a pixel
 - http://speech.ee.ntu.edu.tw/~tlkagk/courses/ML_2016/Pokemon_creation/colormap.txt

0
0 0 0 19 41 34 0 0 19 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 1 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 1 44 74 44 51 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 1 21 80 80 81 0 0 0 0 0 0 0 0
0 0 0 0 0 0 1 2 3 18 35 22 0 5 2 0 0 0 0 0 0
93 94 93 93 85 95 38 96 97 98 99 99 67 99 9
0 0 0 0 0 0 1 106 106 106 106 106 61 107 0

0 → 255 255 255
1 → 53 53 53
2 → 49 49 49
.....
186 186 186
51 51 51
54 54 54
187 187 187
83 83 83
50 51 52
251 251 251
52 52 52
:

- Following experiment: 1-layer LSTM, 512 cells

Real
Pokémon

Never seen
by machine!

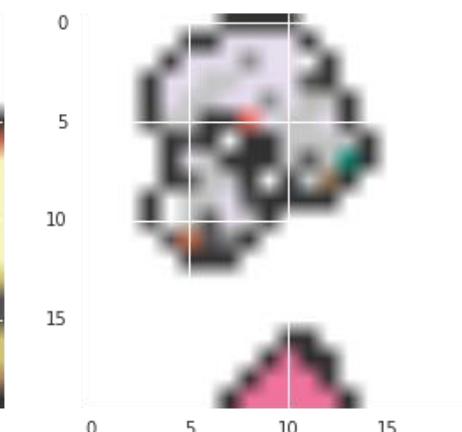
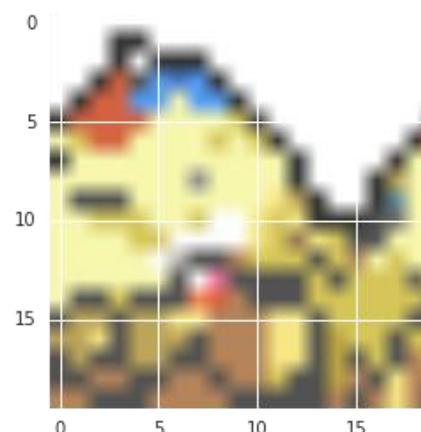
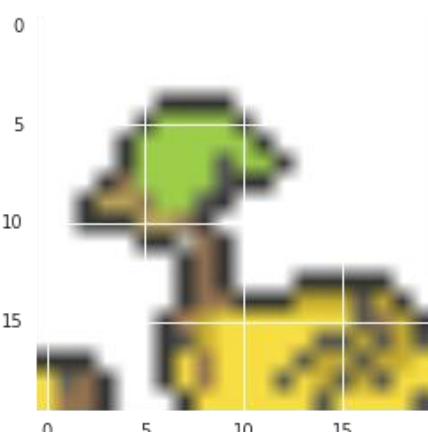
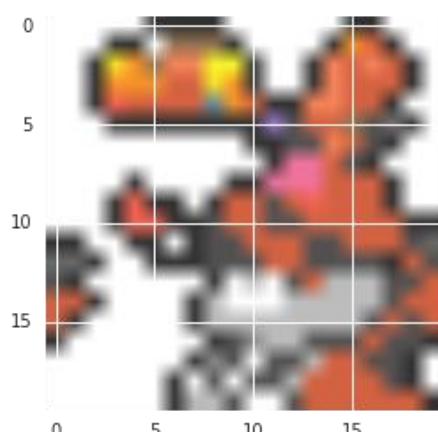
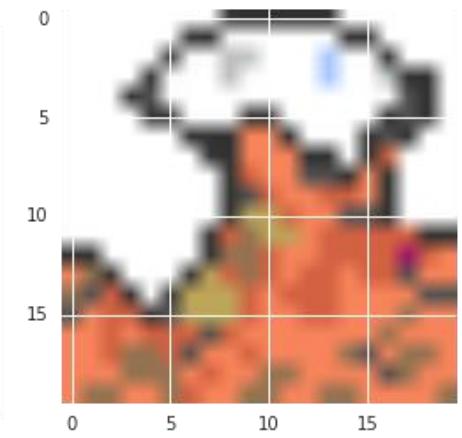
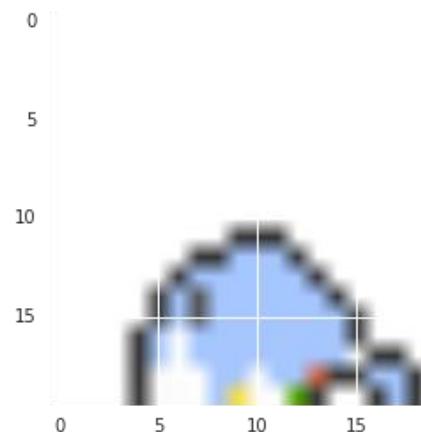
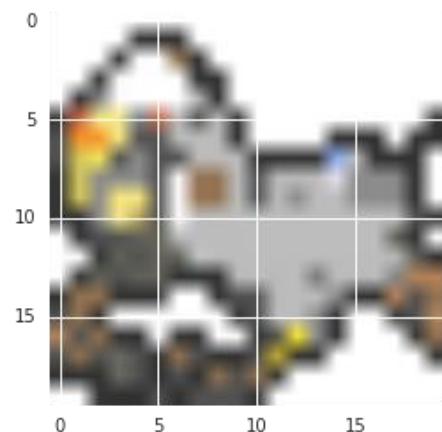
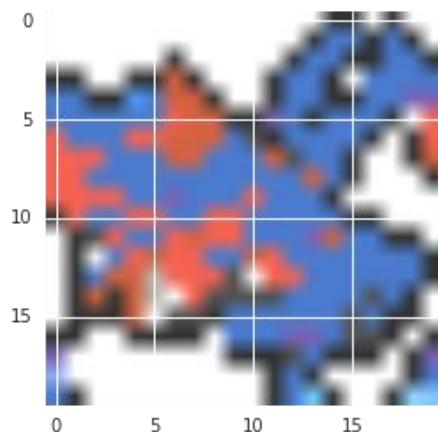
Cover 50%

It is difficult to evaluate generation.

Cover 75%

Pokémon Creation

Drawing from scratch
Need some randomness

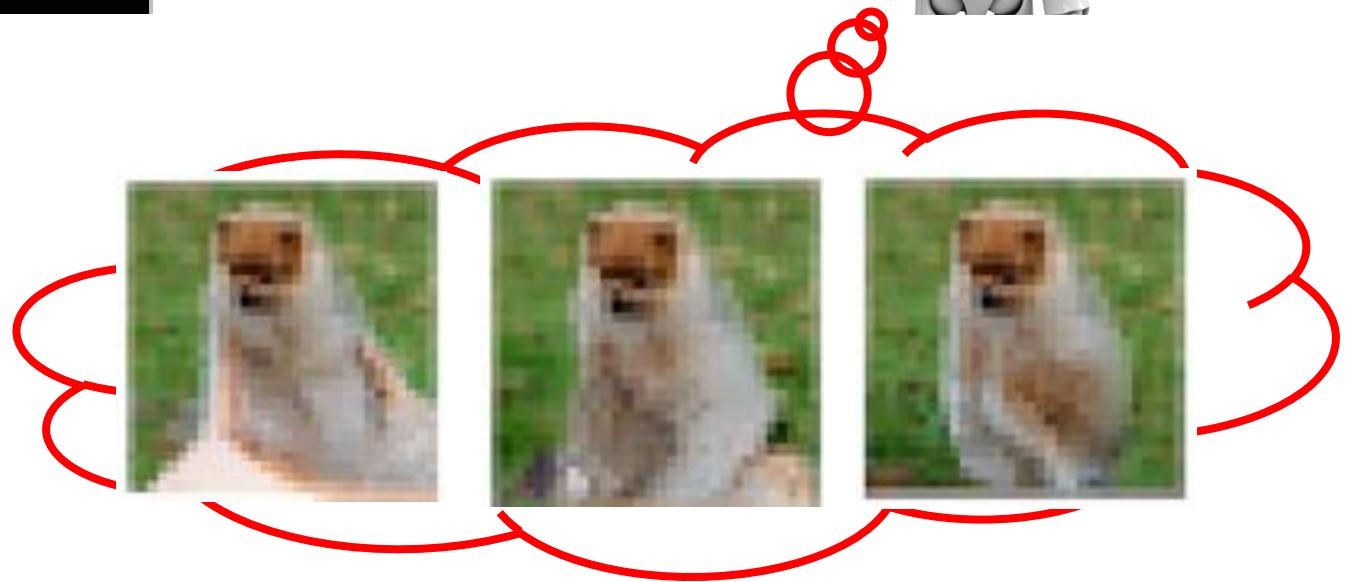


PixelRNN

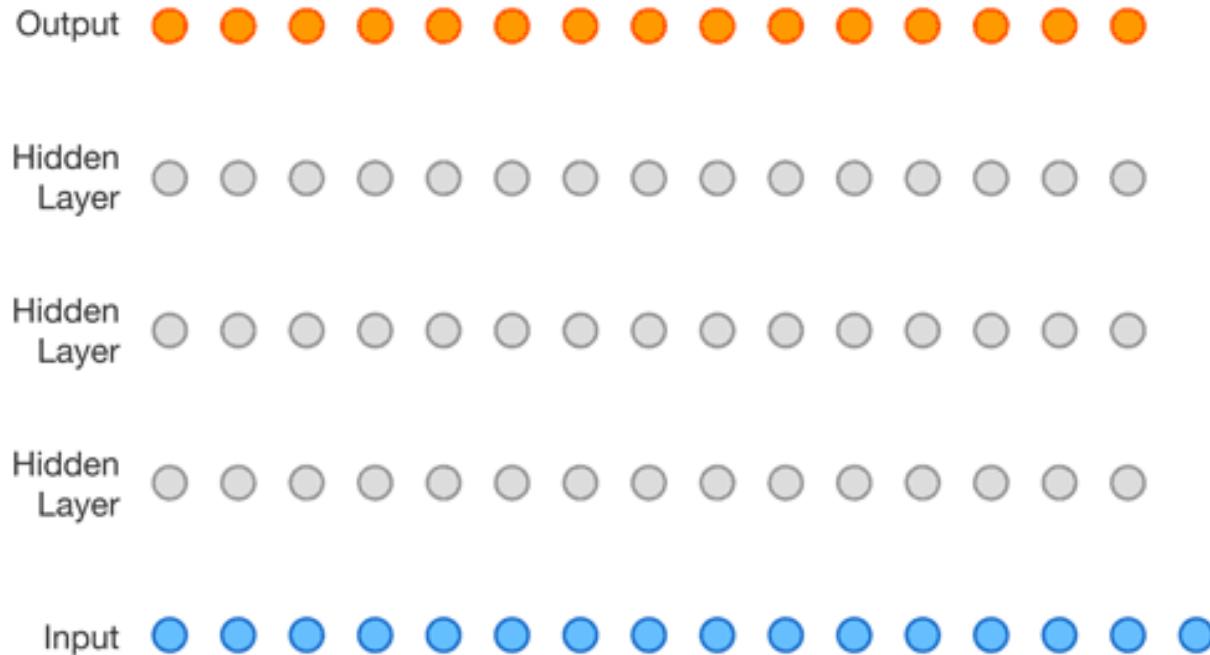
Ref: Aaron van den Oord, Nal Kalchbrenner, Koray Kavukcuoglu, Pixel Recurrent Neural Networks, arXiv preprint, 2016



Real
World



More than images



Audio: Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, Koray Kavukcuoglu,
WaveNet: A Generative Model for Raw Audio, arXiv preprint, 2016

Video: Nal Kalchbrenner, Aaron van den Oord, Karen Simonyan, Ivo
Danihelka, Oriol Vinyals, Alex Graves, Koray Kavukcuoglu, Video Pixel Networks ,
arXiv preprint, 2016

Generative Models

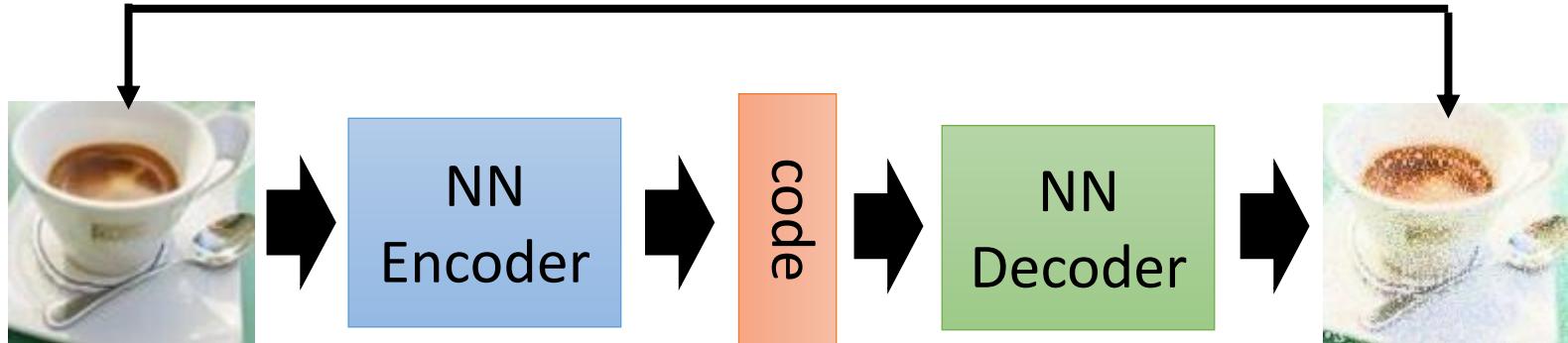
Component-by-component

Autoencoder

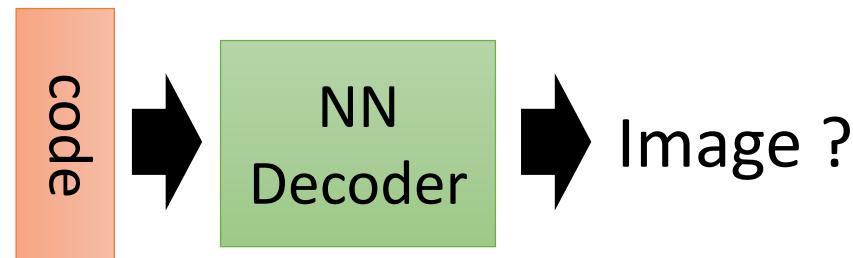
Generative Adversarial Network
(GAN)

Auto-encoder

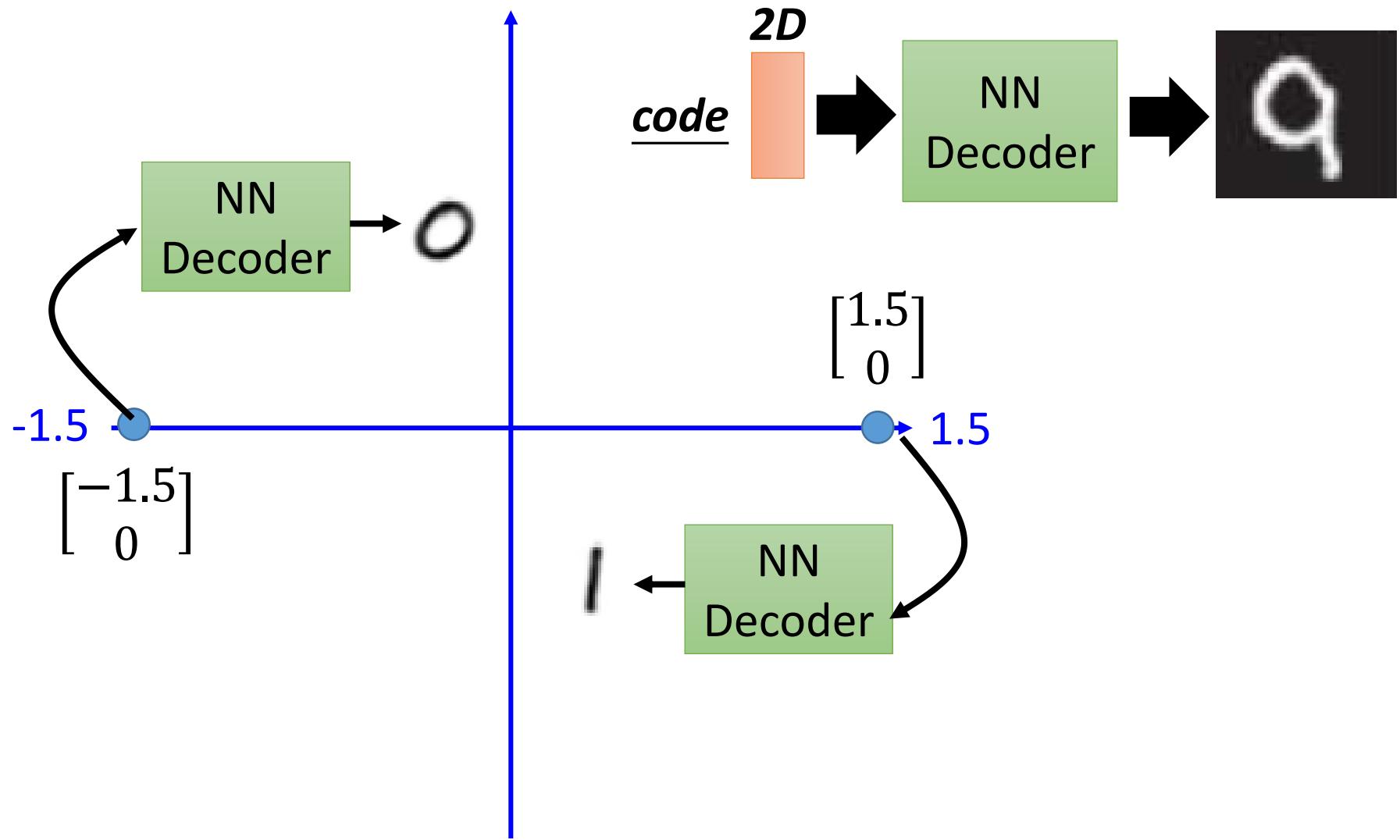
As close as possible



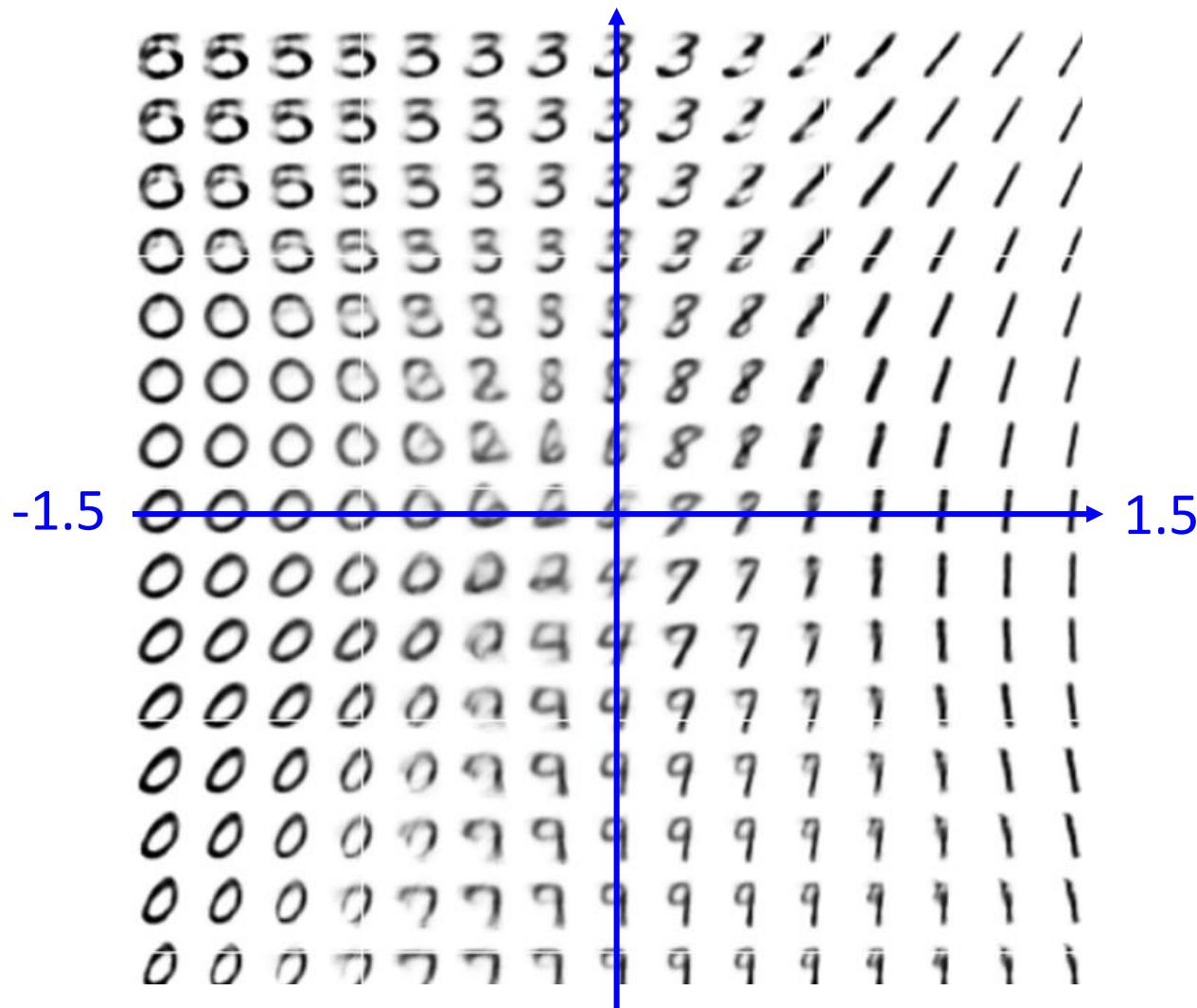
Randomly generate
a vector as code



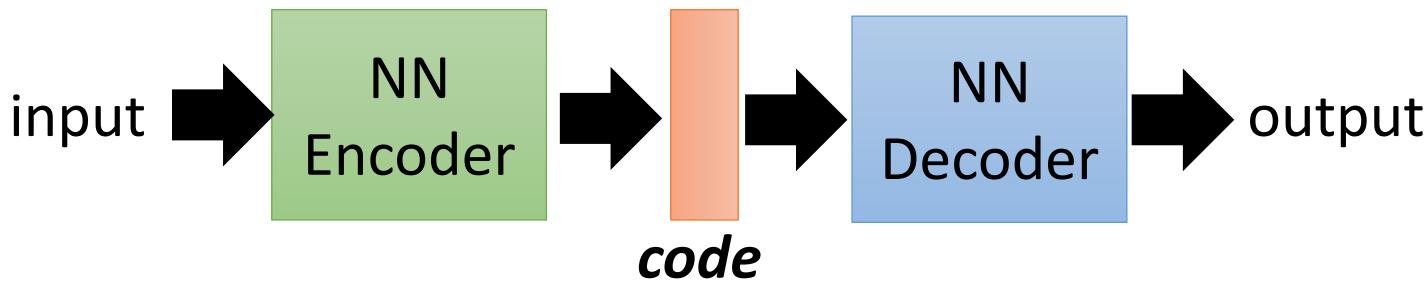
Review: Auto-encoder



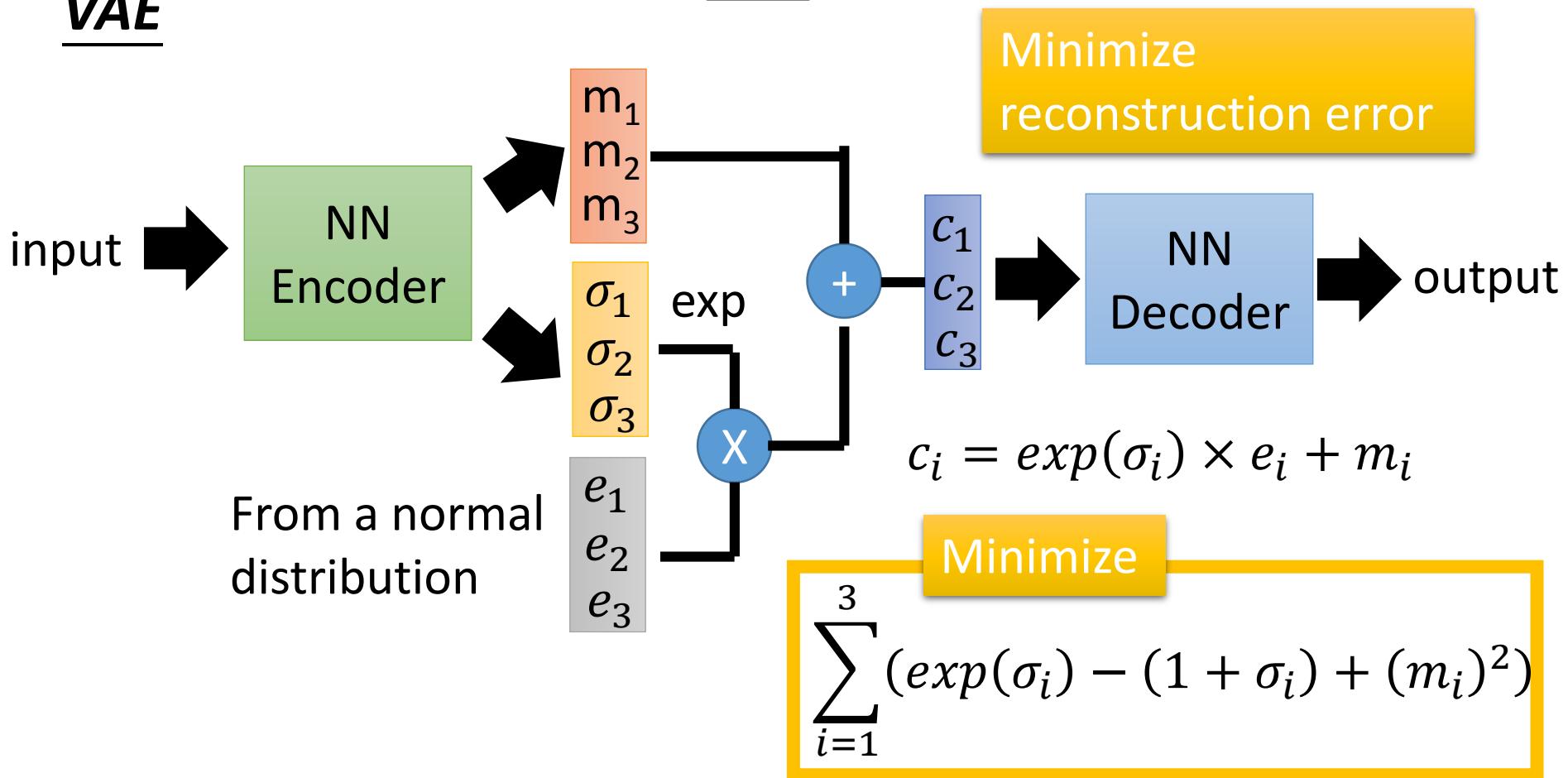
Review: Auto-encoder



Auto-encoder



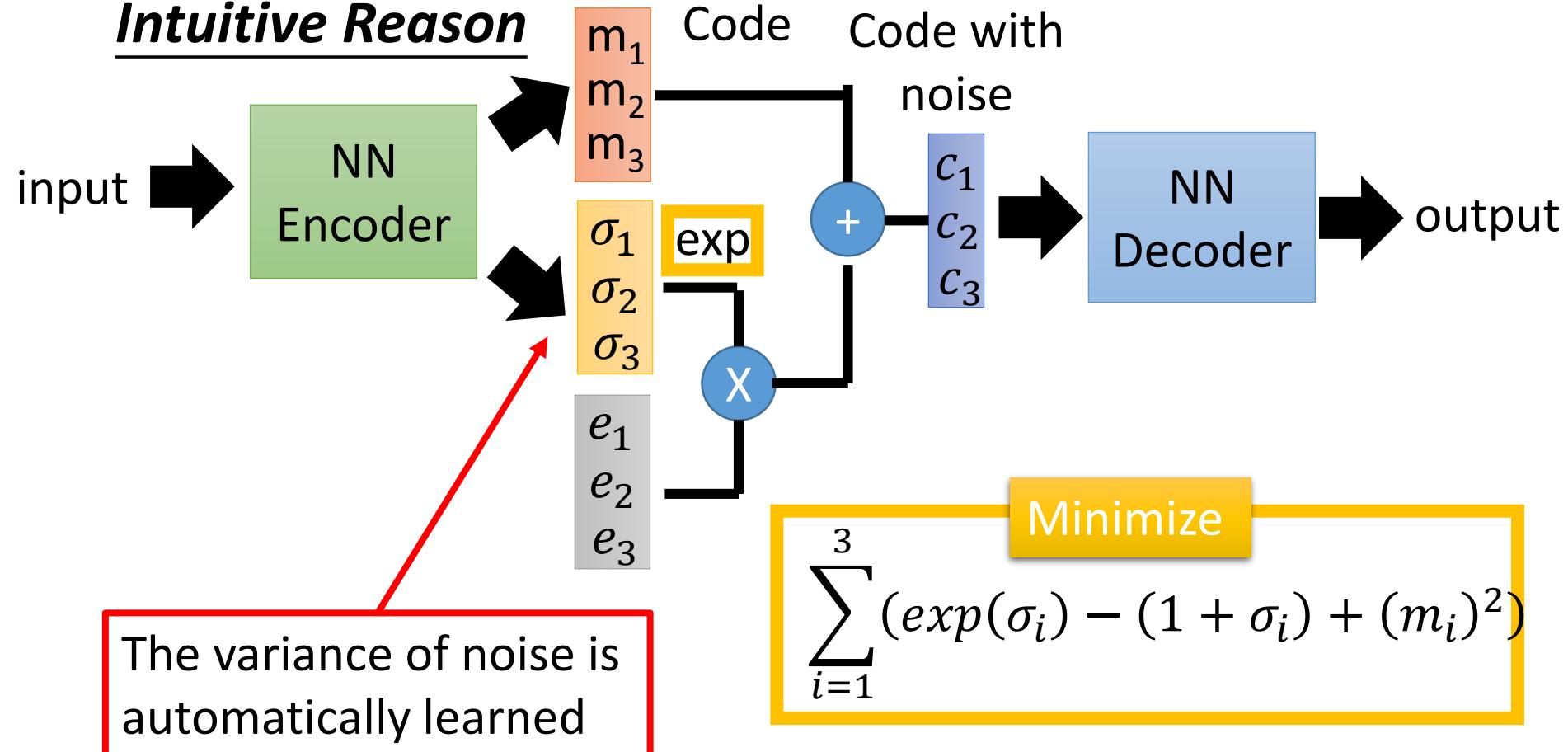
VAE



Why VAE?

What will happen if we only minimize reconstruction error?

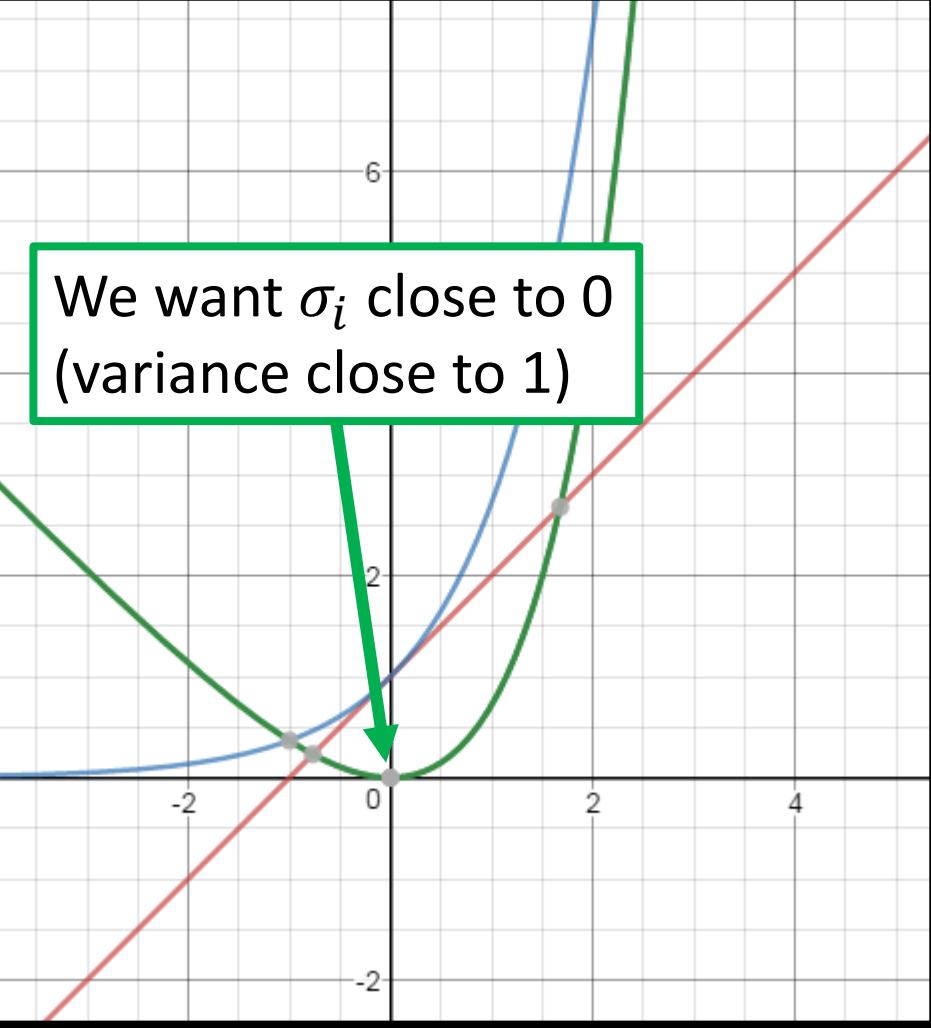
Intuitive Reason



Why VAE?

What will happen if we only minimize reconstruction error?

Intuitive Reason



Original

Code with
noisy

$$c_1 \\ c_2 \\ c_3$$

NN
Decoder

output

c_3

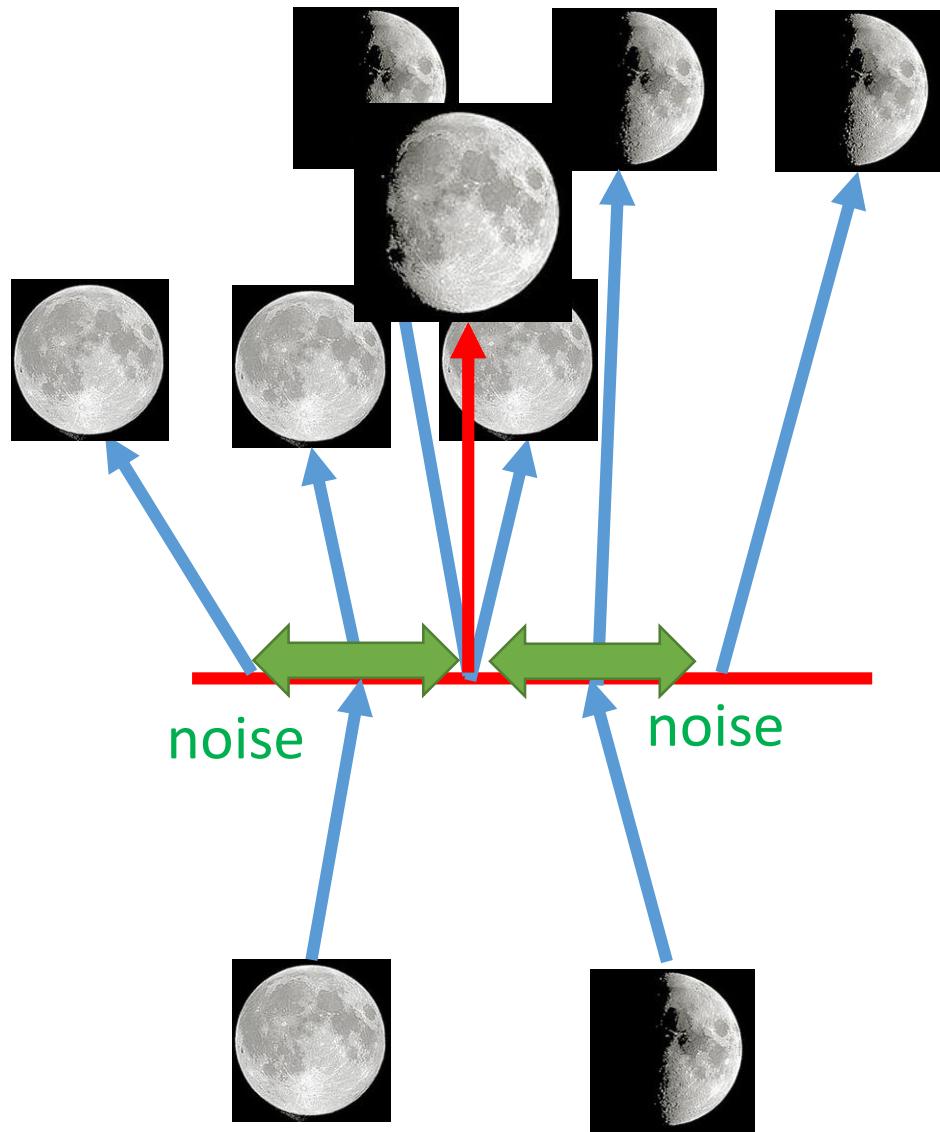
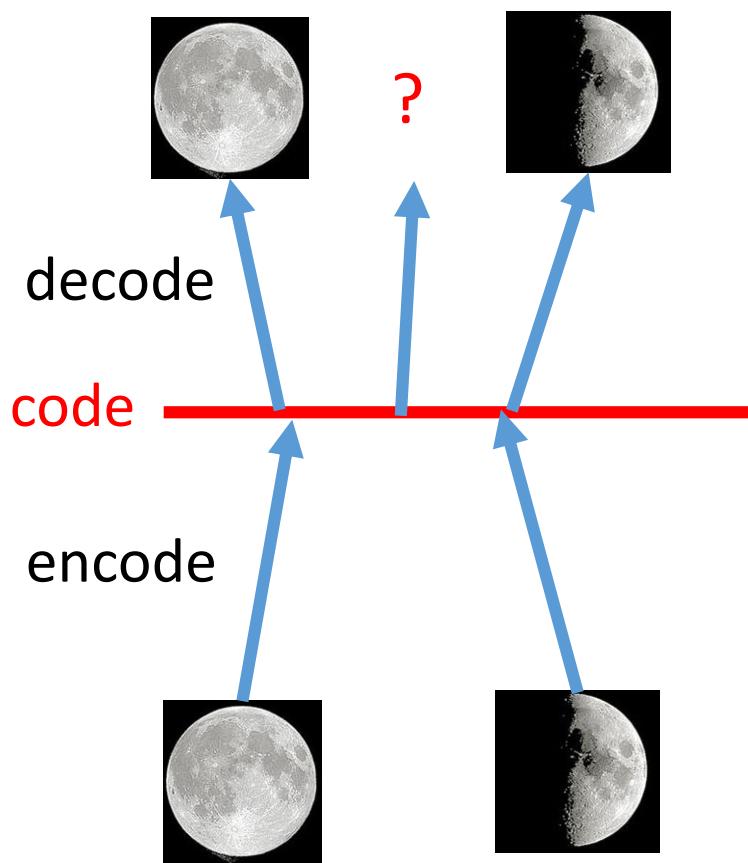
Minimize

$$\sum_{i=1}^3 \left(\exp(\sigma_i) - (1 + \sigma_i) + (m_i)^2 \right)$$

L2 regularization

Why VAE?

Intuitive Reason



Warning of Math

Gaussian Mixture Model

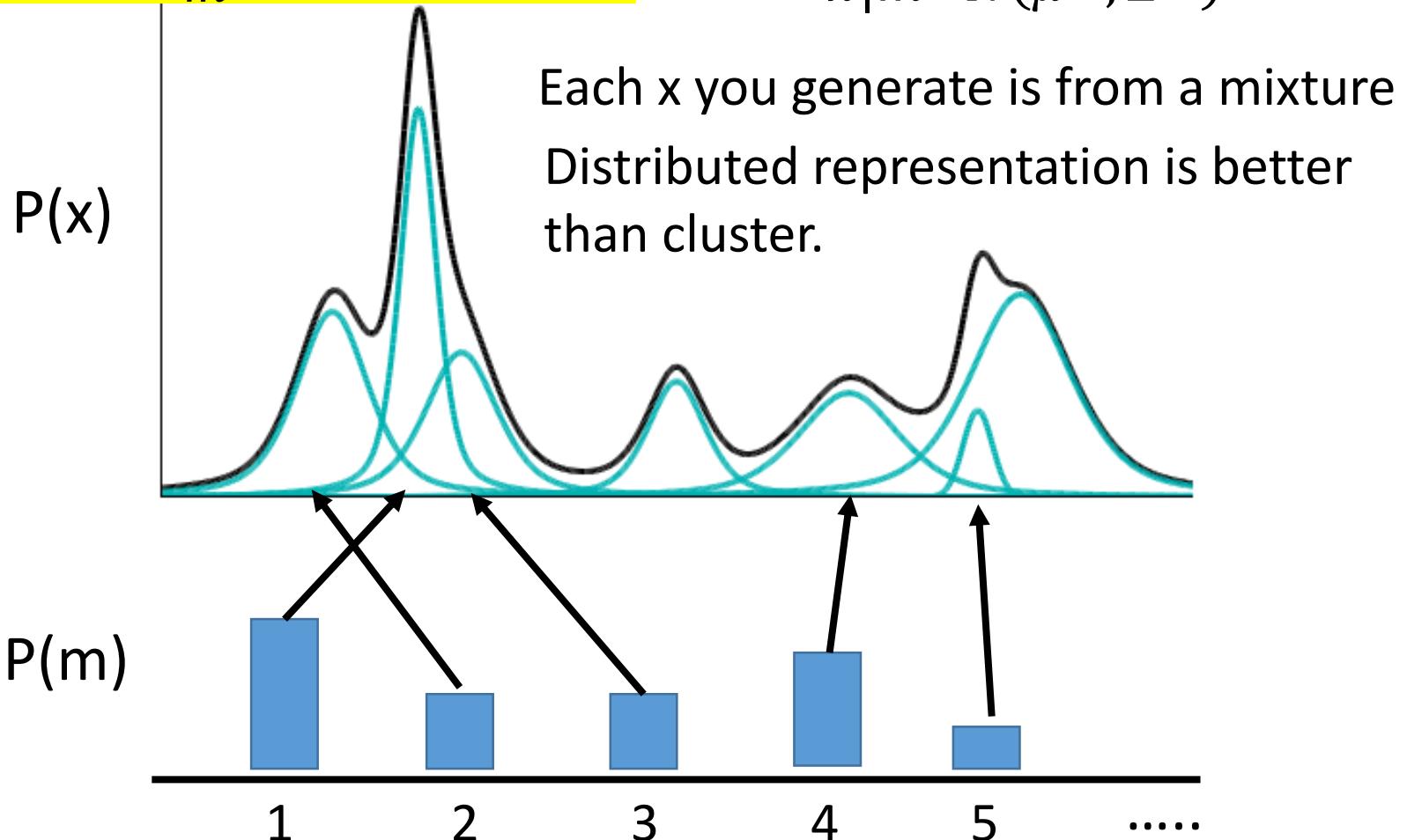
$$P(x) = \sum_m P(m)P(x|m)$$

How to sample?

$m \sim P(m)$ (multinomial)

m is an integer

$x|m \sim N(\mu^m, \Sigma^m)$



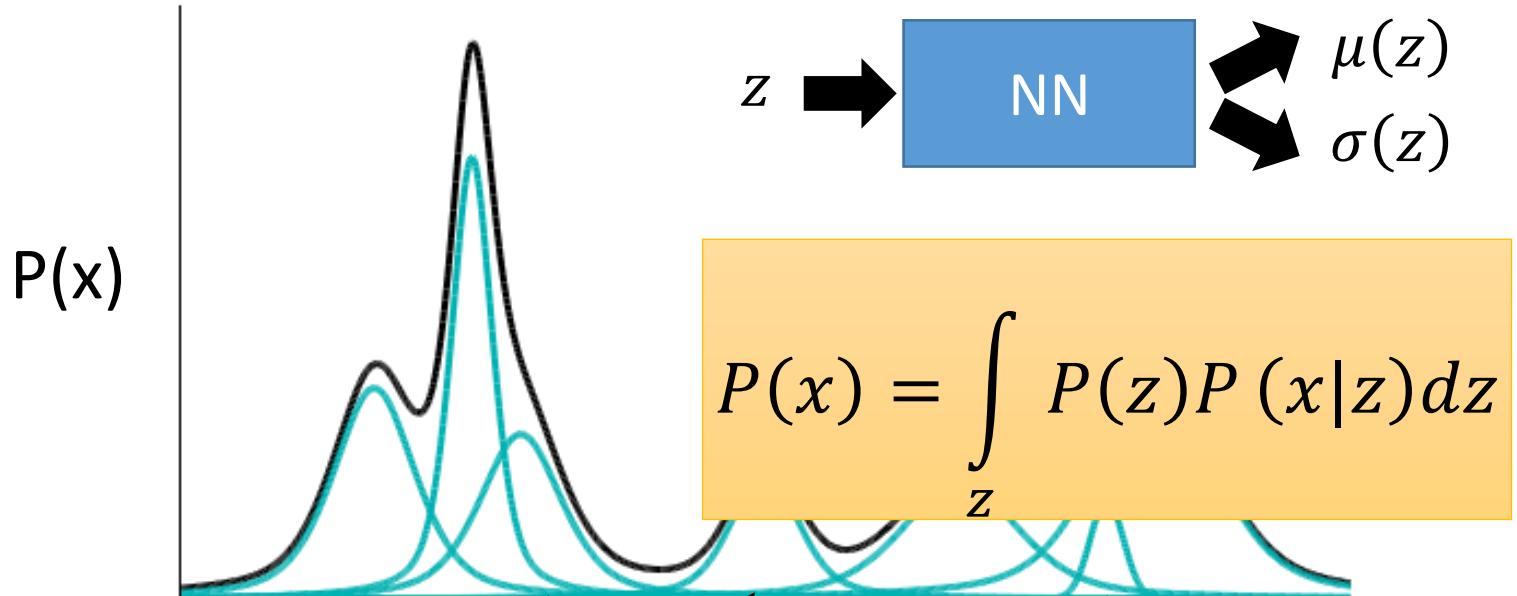
VAE

$$z \sim N(0, I)$$

z is a vector from normal distribution

$$x|z \sim N(\mu(z), \sigma(z))$$

Each dimension of z
represents an attribute



Even though z is
from $N(0, I)$, $P(x)$
can be very complex

z

Infinite Gaussian

Maximizing Likelihood

$$P(x) = \int_z P(z)P(x|z)dz$$

$P(z)$ is normal distribution

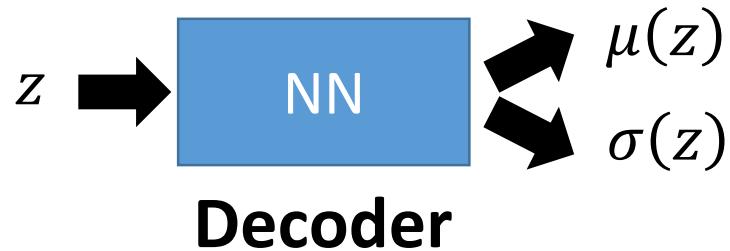
$$x|z \sim N(\mu(z), \sigma(z))$$

$\mu(z), \sigma(z)$ is going to be estimated

$$L = \sum_x \log P(x)$$

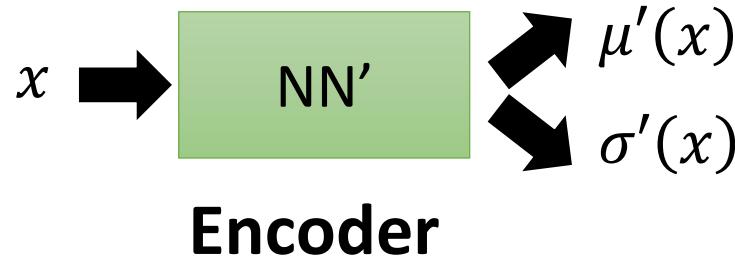
Maximizing the likelihood of the observed x

Tuning the parameters to
maximize likelihood L



We need another
distribution $q(z|x)$

$$z|x \sim N(\mu'(x), \sigma'(x))$$



Maximizing Likelihood

$$P(x) = \int_z P(z)P(x|z)dz$$

P(z) is normal distribution

$$x|z \sim N(\mu(z), \sigma(z))$$

$\mu(z), \sigma(z)$ is going to be estimated

$$L = \sum_x logP(x)$$

Maximizing the likelihood of the observed x

$$logP(x) = \int_z q(z|x)logP(x)dz$$

q(z|x) can be any distribution

$$= \int_z q(z|x)log\left(\frac{P(z,x)}{P(z|x)}\right)dz = \int_z q(z|x)log\left(\frac{P(z,x)}{q(z|x)}\frac{q(z|x)}{P(z|x)}\right)dz$$

$$= \int_z q(z|x)log\left(\frac{P(z,x)}{q(z|x)}\right)dz + \underbrace{\int_z q(z|x)log\left(\frac{q(z|x)}{P(z|x)}\right)dz}_{KL(q(z|x)||P(z|x))}$$

$$\geq \int_z q(z|x)log\left(\frac{P(x|z)P(z)}{q(z|x)}\right)dz$$

lower bound L_b

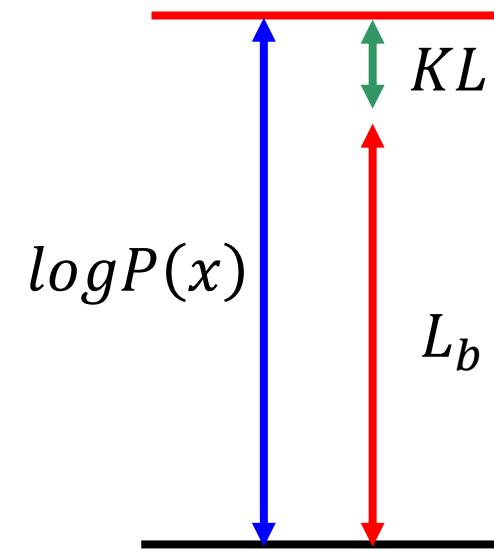
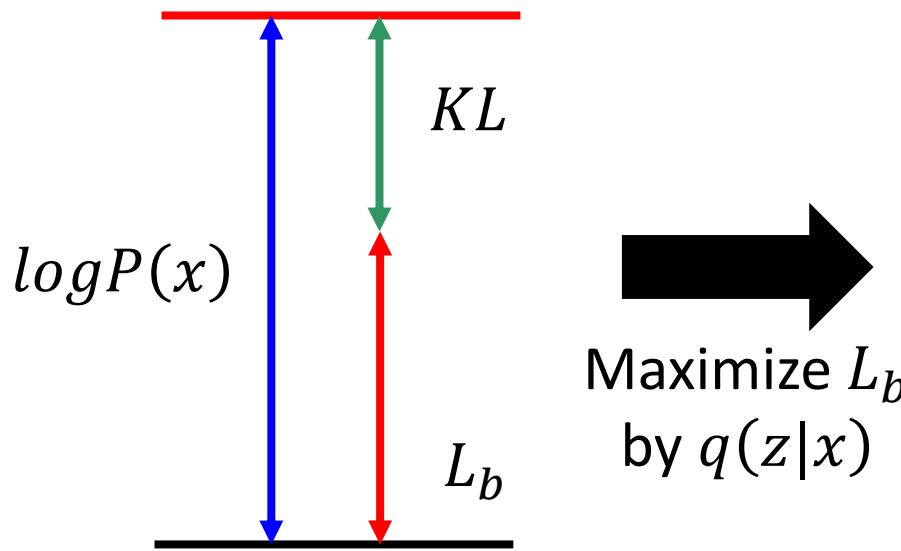
≥ 0

Maximizing Likelihood

$$\log P(x) = L_b + KL(q(z|x)||P(z|x))$$

$$L_b = \int_z q(z|x) \log \left(\frac{P(x|z)P(z)}{q(z|x)} \right) dz$$

Find $P(x|z)$ and $q(z|x)$ maximizing L_b



$q(z|x)$ will be an approximation of $p(z|x)$ in the end

Maximizing Likelihood

$$P(x) = \int_z P(z)P(x|z)dz$$

$$L = \sum_x \log P(x)$$

Maximizing the likelihood of the observed x

$$L_b = \int_z q(z|x) \log \left(\frac{P(z,x)}{q(z|x)} \right) dz = \int_z q(z|x) \log \left(\frac{P(x|z)P(z)}{q(z|x)} \right) dz$$

$$= \int_z \underbrace{q(z|x) \log \left(\frac{P(z)}{q(z|x)} \right) dz}_{-KL(q(z|x)||P(z))} + \int_z q(z|x) \log P(x|z) dz$$

$$z|x \sim N(\mu'(x), \sigma'(x))$$



Connection with Network

Minimizing $KL(q(z|x)||P(z))$



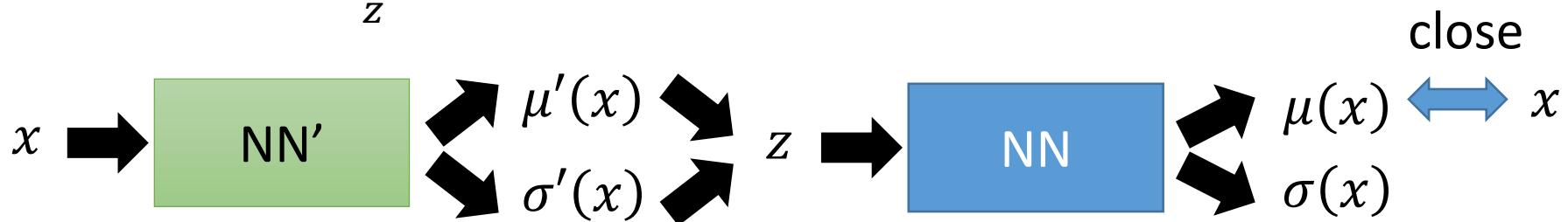
Minimize

$$\sum_{i=1}^3 (\exp(\sigma_i) - (1 + \sigma_i) + (m_i)^2)$$

(Refer to the Appendix B of the original VAE paper)

Maximizing

$$\int_z q(z|x) \log P(x|z) dz = E_{q(z|x)} [\log P(x|z)]$$

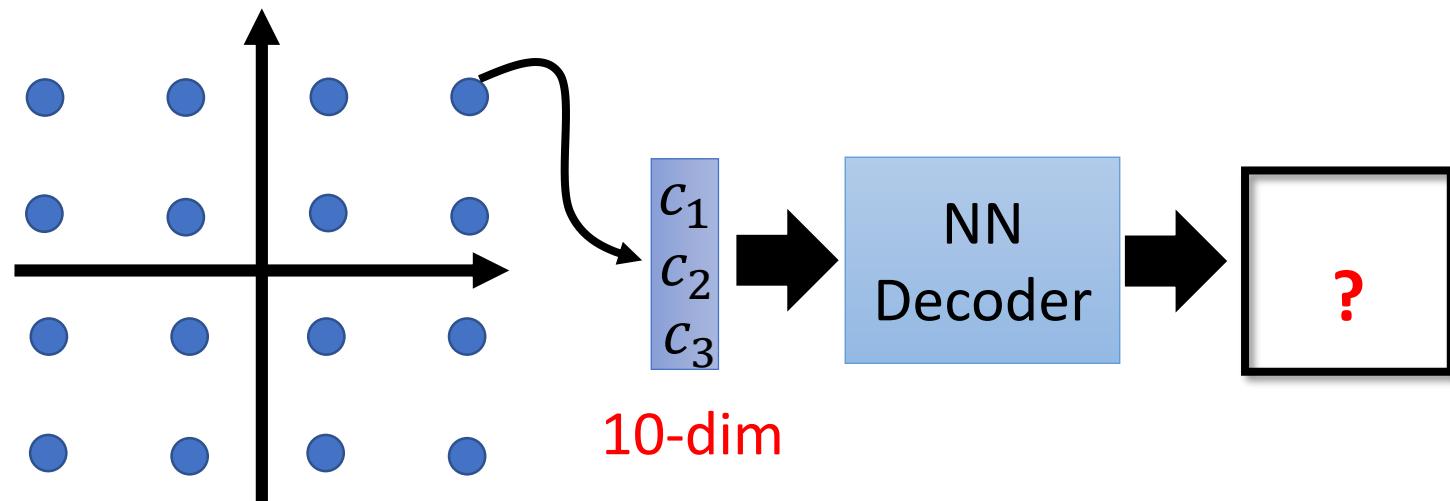
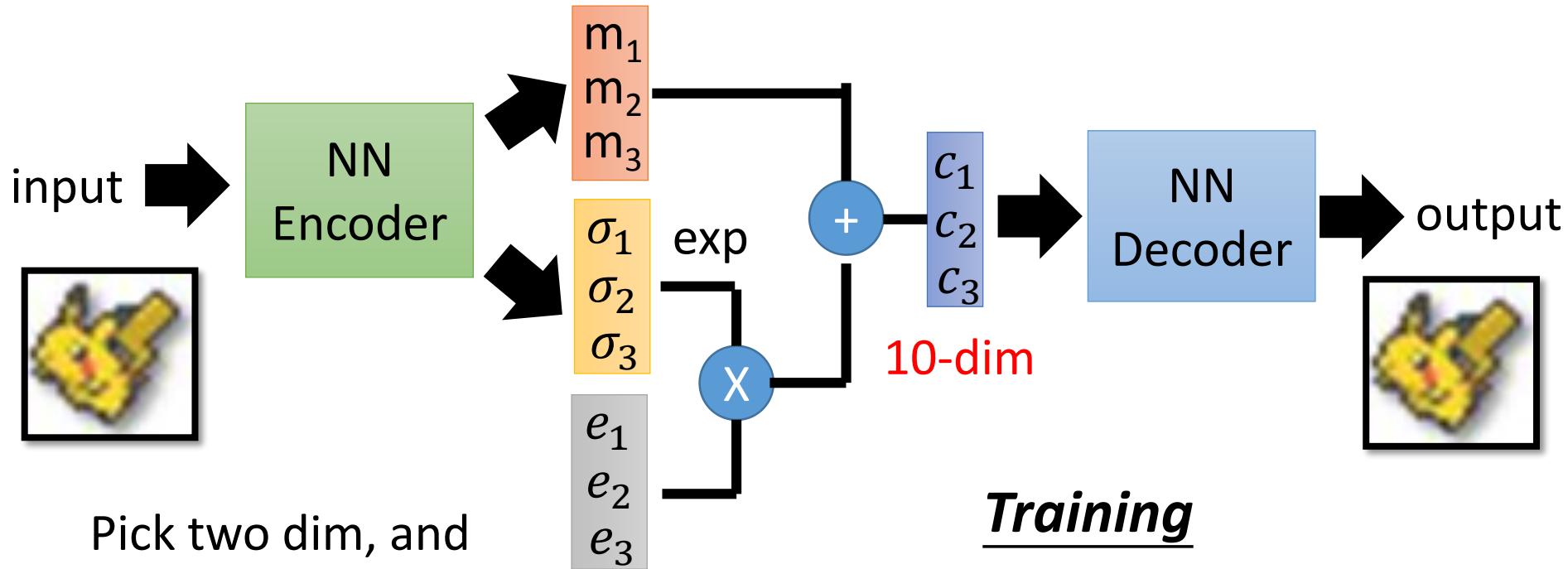


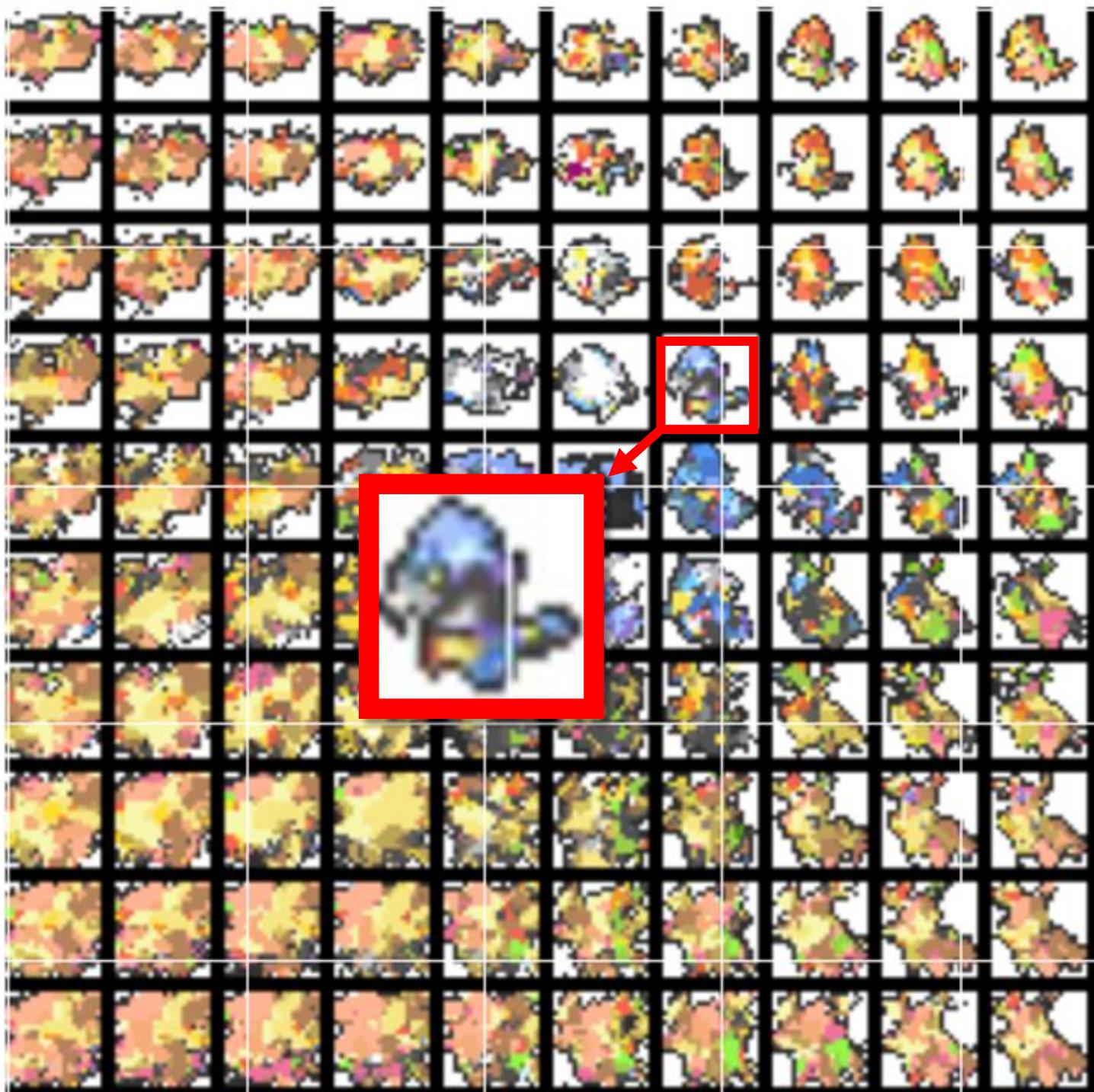
close

This is the auto-encoder

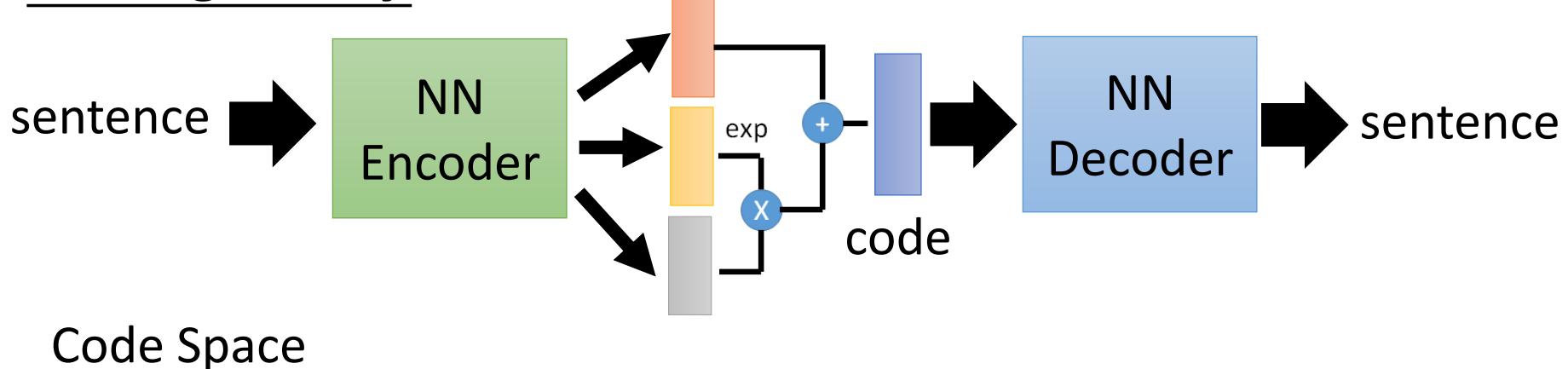
End of Warning

Pokémon Creation

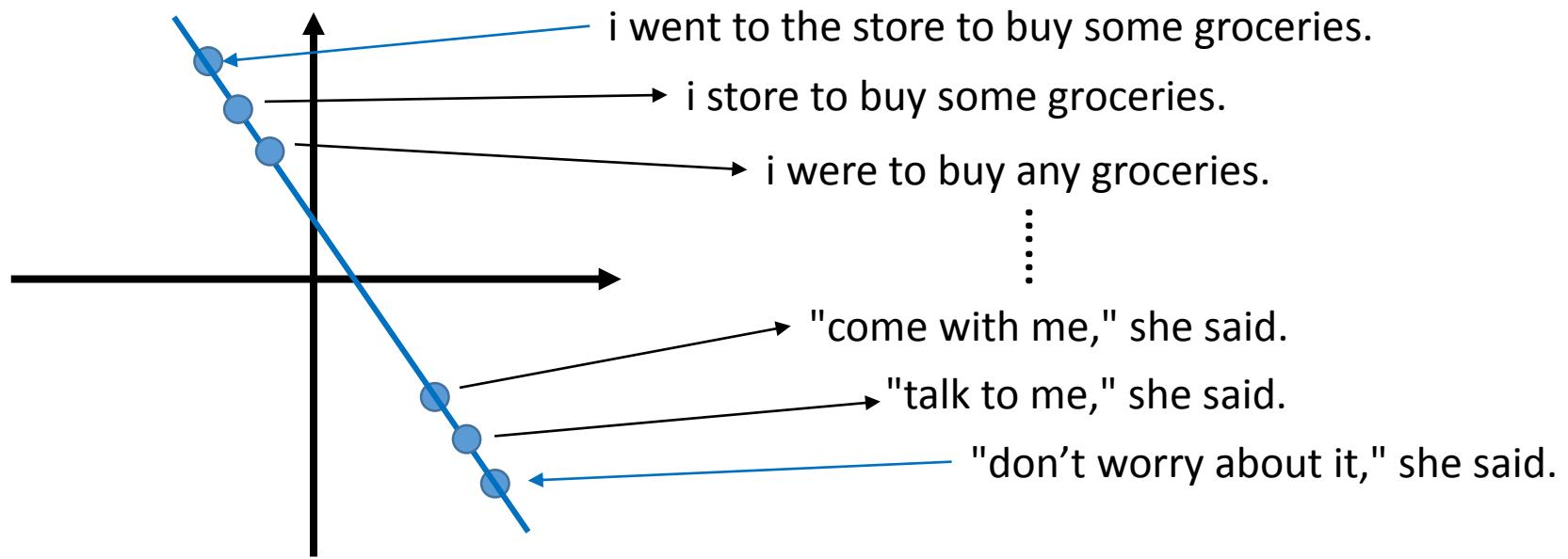




Writing Poetry



Code Space

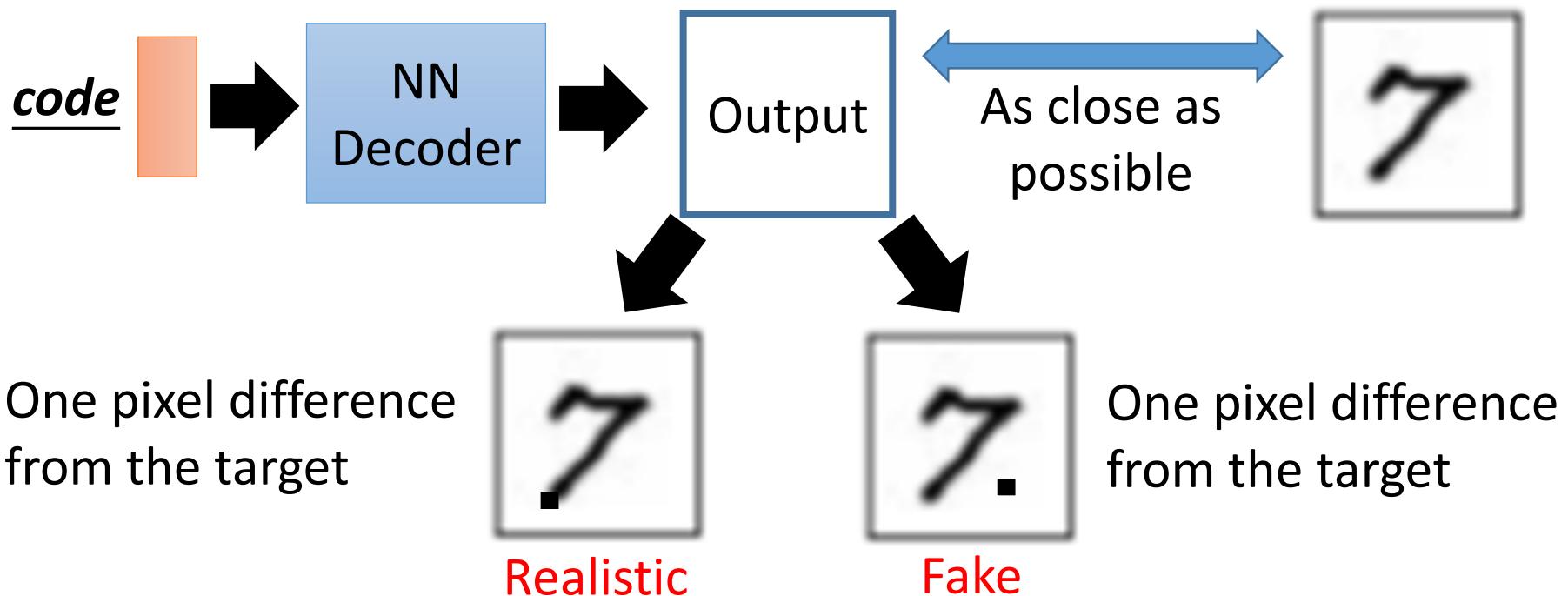


Ref: <http://www.wired.co.uk/article/google-artificial-intelligence-poetry>

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Jozefowicz, Samy Bengio, Generating Sentences from a Continuous Space, arXiv preprint, 2015

Problems of VAE

- It does not really try to simulate real images



VAE may just memorize the existing images, instead of generating new images

Generative Models

Component-by-component

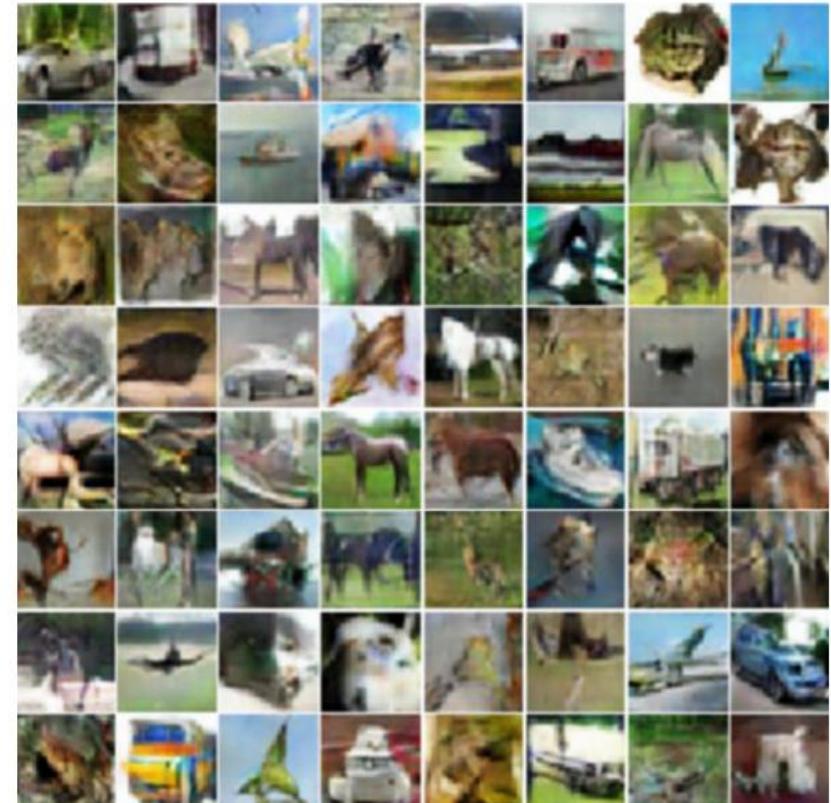
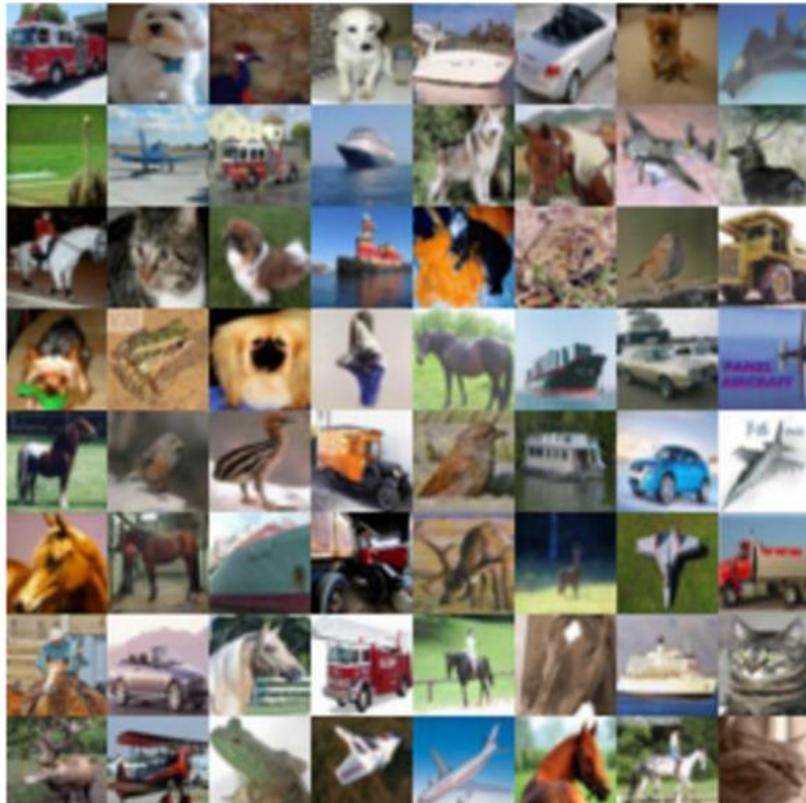
Autoencoder

Generative Adversarial Network
(GAN)

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio, Generative Adversarial Networks, arXiv preprint 2014

Cifar-10

- Which one is machine-generated?



Ref: <https://openai.com/blog/generative-models/>

Yann LeCun's comment

What are some recent and potentially upcoming breakthroughs in unsupervised learning?



Yann LeCun, Director of AI Research at Facebook and Professor at NYU



Written Jul 29 · Upvoted by Joaquin Quiñonero Candela, Director Applied Machine Learning at Facebook and Huang Xiao

Adversarial training is the coolest thing since sliced bread.

I've listed a bunch of relevant papers in a previous answer.

Expect more impressive results with this technique in the coming years.

What's missing at the moment is a good understanding of it so we can make it work reliably. It's very finicky. Sort of like ConvNet were in the 1990s, when I had the reputation of being the only person who could make them work (which wasn't true).

Yann LeCun's comment

What are some recent and potentially upcoming breakthroughs in deep learning?



Yann LeCun, Director of AI Research at Facebook and Professor at NYU

Written Jul 29 · Upvoted by Joaquin Quiñonero Candela, Director Applied Machine Learning at Facebook and Nikhil Garg, I lead a team of Quora engineers working on ML/NLP problems



.....

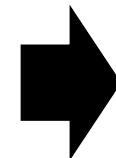
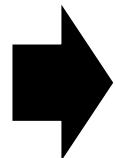
The most important one, in my opinion, is adversarial training (also called GAN for Generative Adversarial Networks). This is an idea that was originally proposed by Ian Goodfellow when he was a student with Yoshua Bengio at the University of Montreal (he since moved to Google Brain and recently to OpenAI).

This, and the variations that are now being proposed is the most interesting idea in the last 10 years in ML, in my opinion.

<https://www.quora.com/What-are-some-recent-and-potentially-upcoming-breakthroughs-in-deep-learning>

Evolution

<http://peellden.pixnet.net/blog/post/40406899-2013-%E7%AC%AC%E5%9B%9B%E5%AD%A3%EF%BC%8C%E5%86%AC%E8%9D%B6%E5%AF%82%E5%AF%A5>



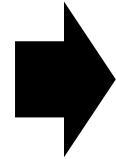
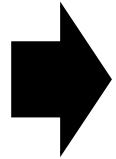
Brown

veins

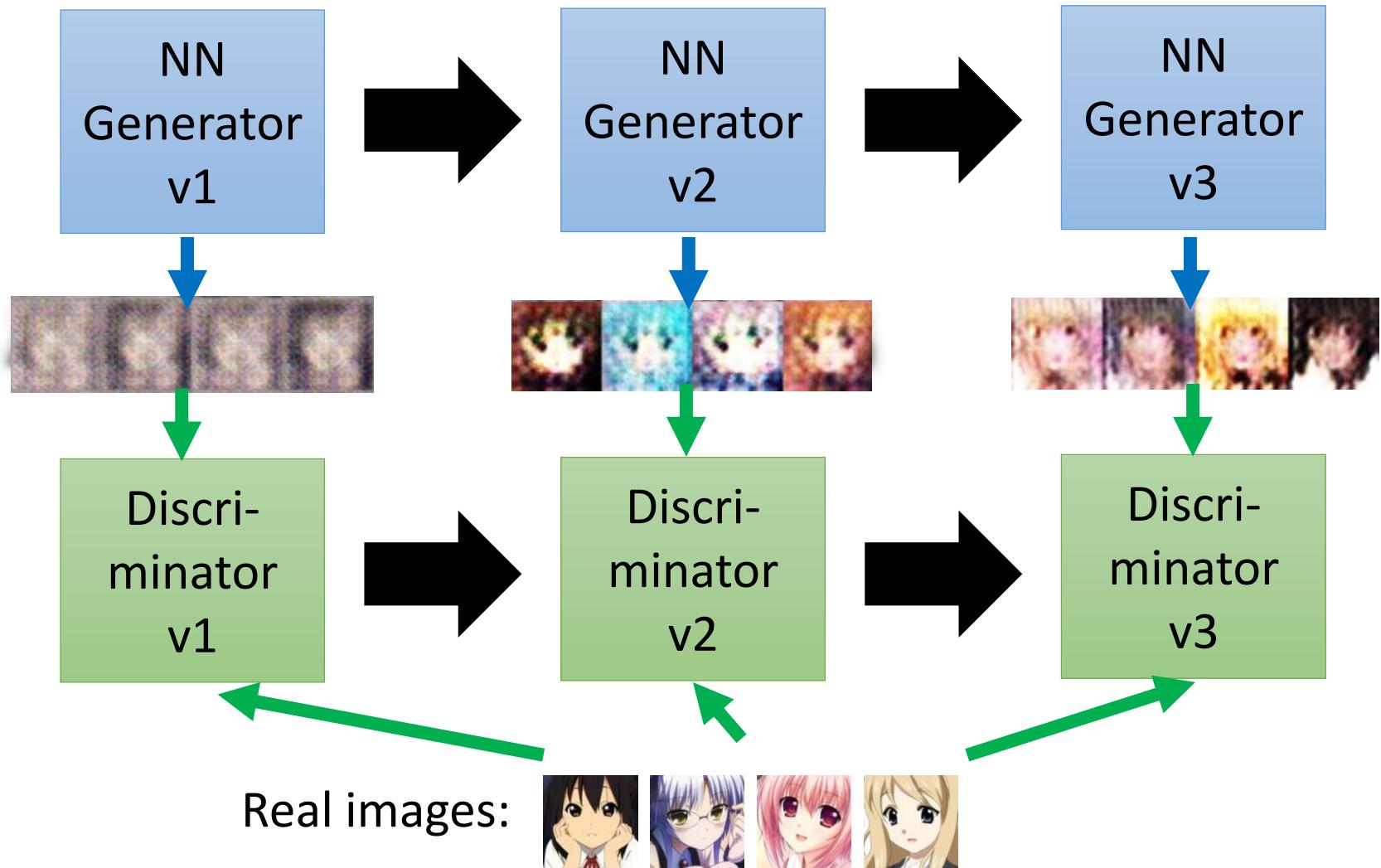
Butterflies are
not brown

Butterflies do
not have veins

.....

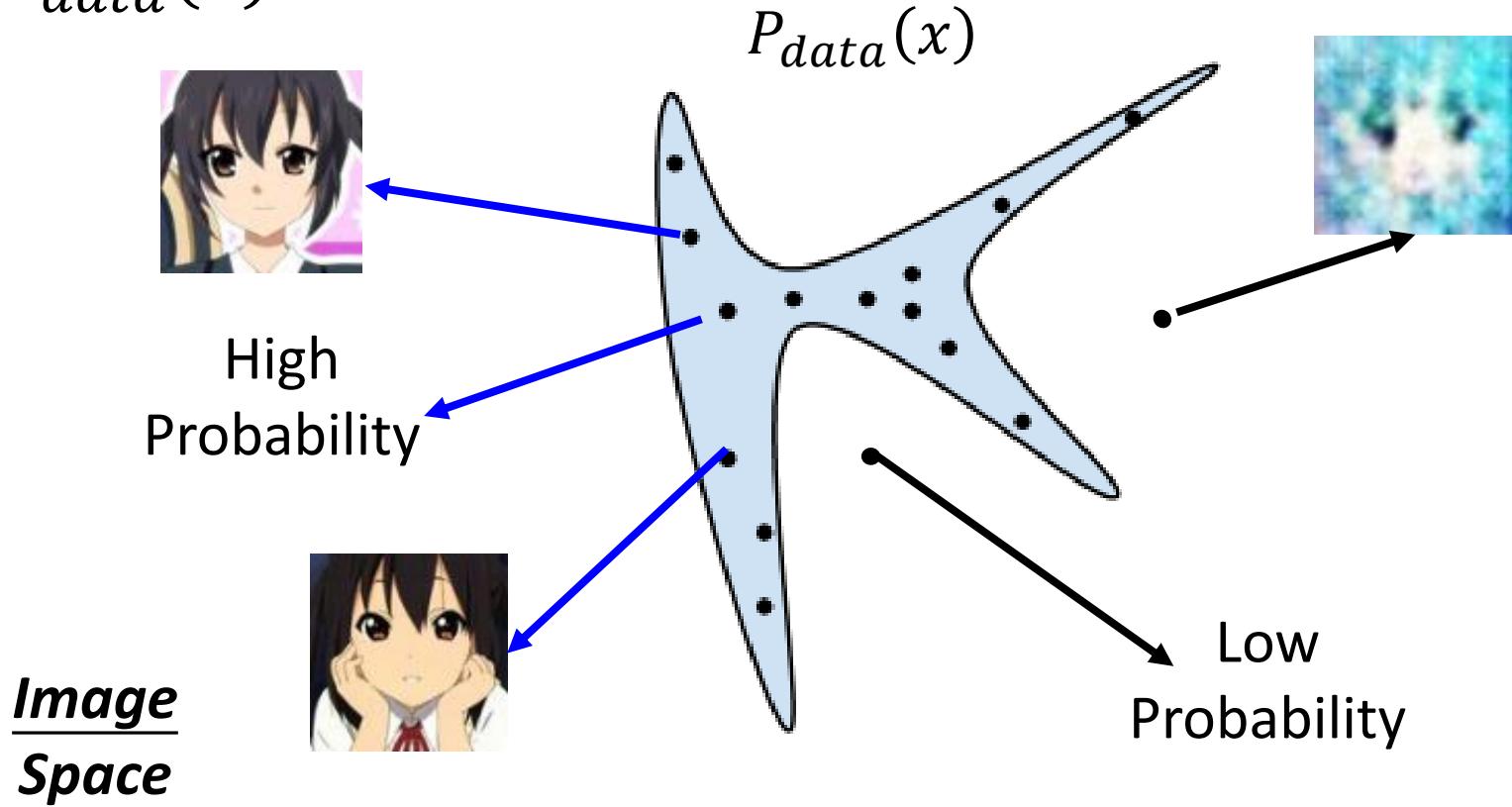


The evolution of generation



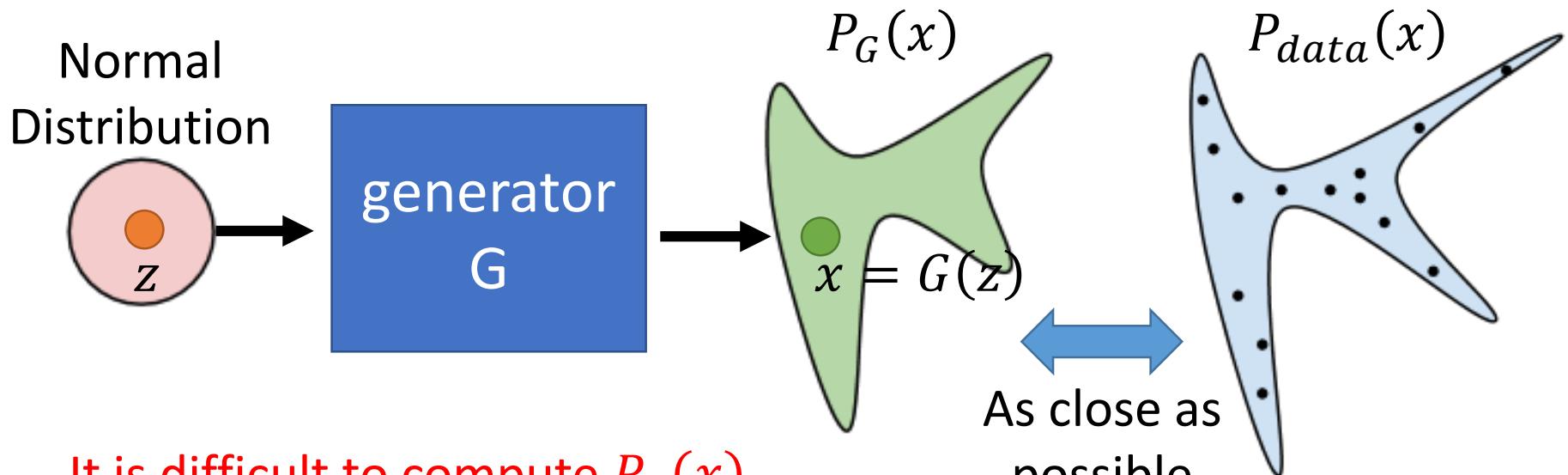
Basic Idea of GAN

- The data we want to generate has a distribution $P_{data}(x)$



Basic Idea of GAN

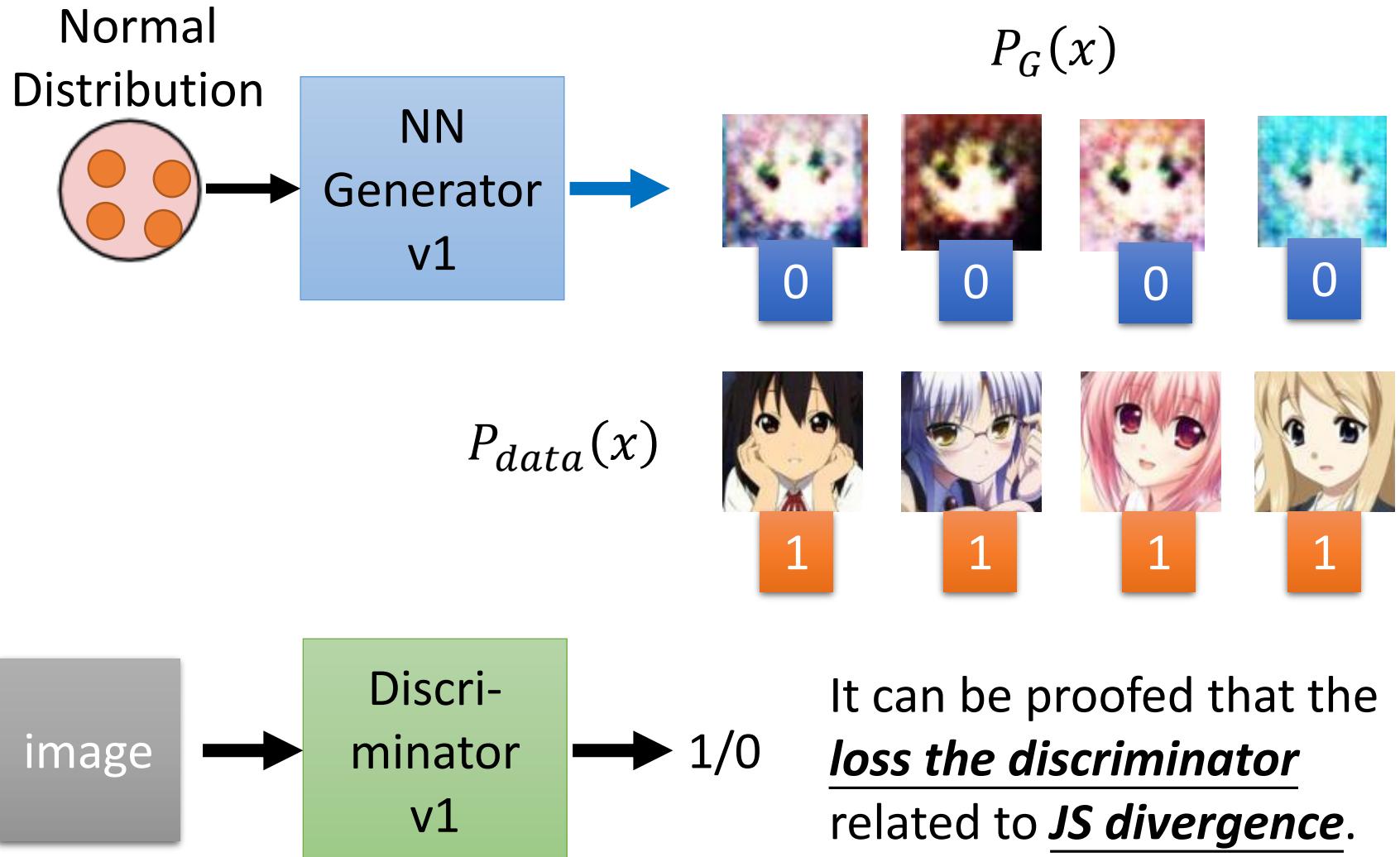
- A generator G is a network. The network defines a probability distribution.



It is difficult to compute $P_G(x)$

We do not know what the distribution looks like.

Basic Idea of GAN



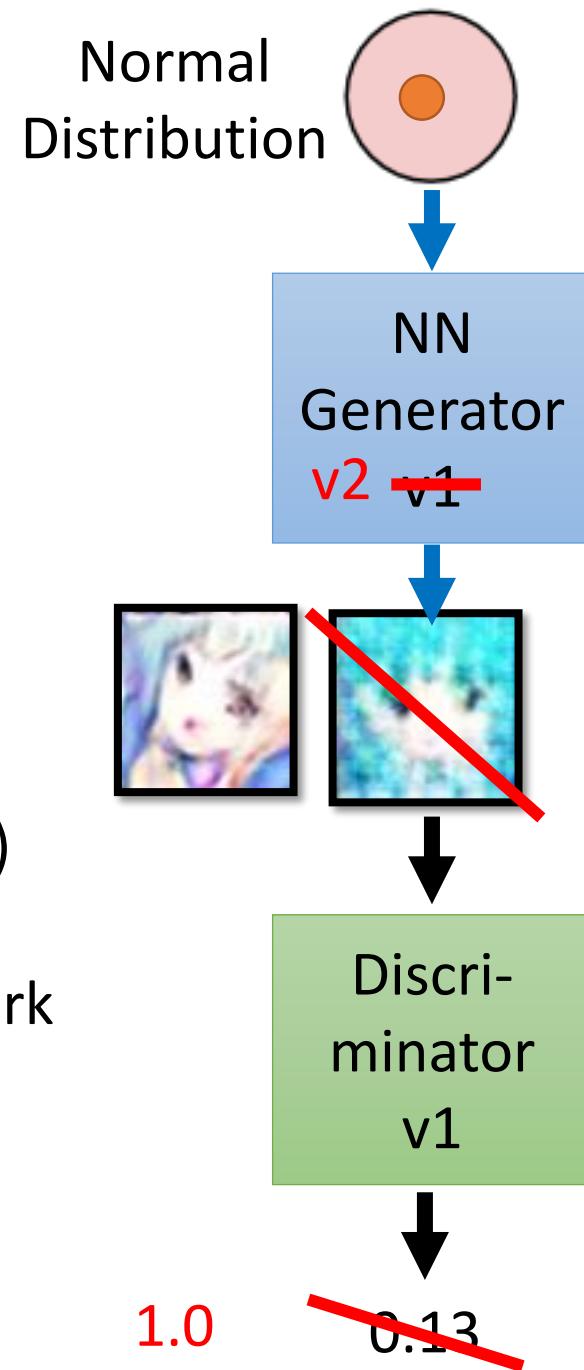
Basic Idea of GAN

- Next step:
 - Updating the parameters of generator
 - To minimize the JS divergence

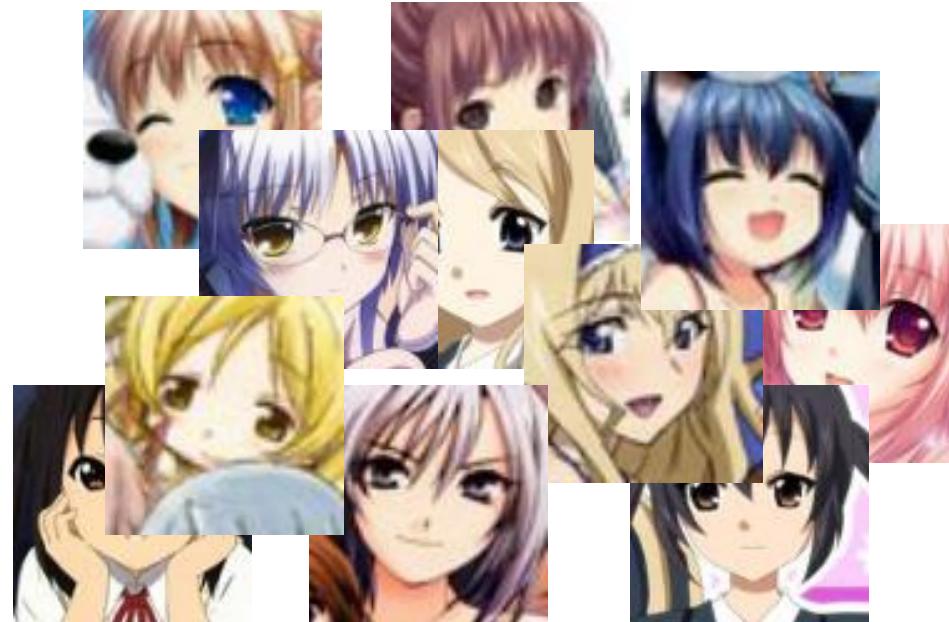
→ The output be classified as “real” (as close to 1 as possible)

Generator + Discriminator = a network

Using gradient descent to update the parameters in the generator, but fix the discriminator



GAN – 二次元人物頭像鍊成

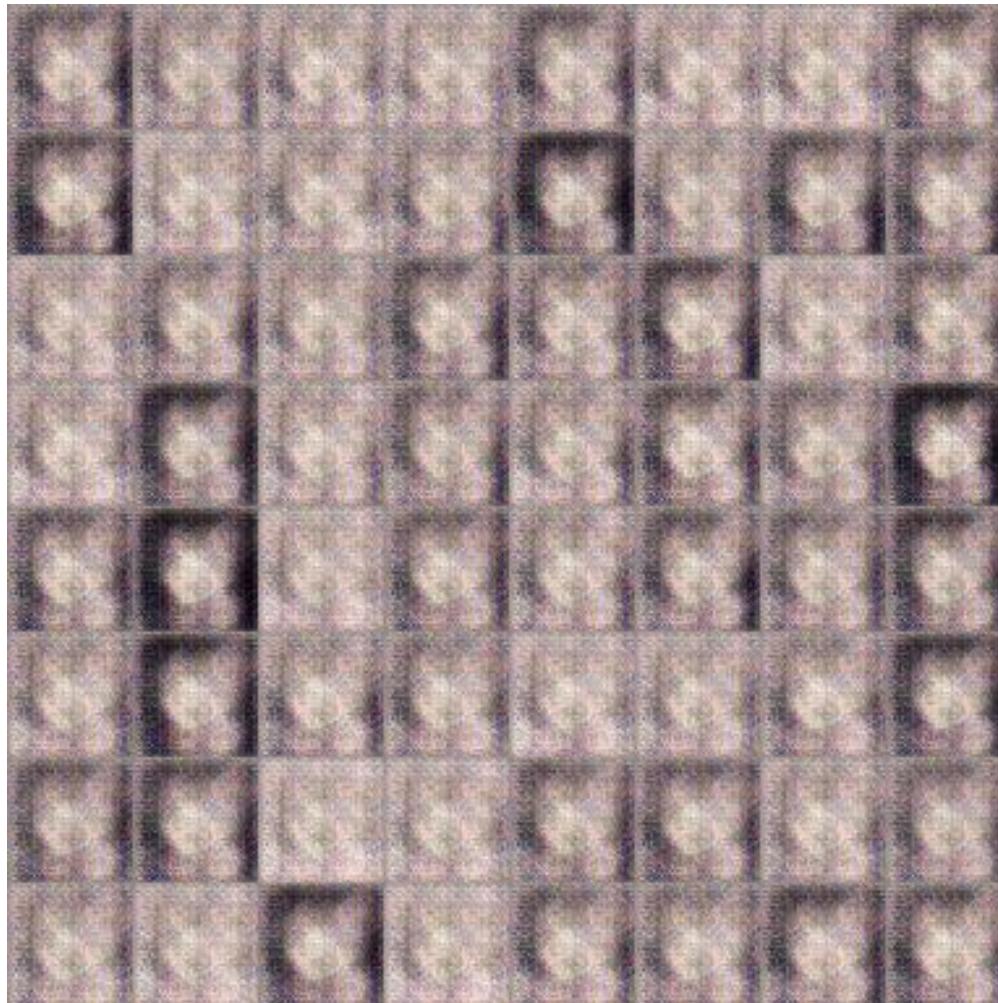


Source of images: <https://zhuanlan.zhihu.com/p/24767059>

DCGAN: <https://github.com/carpedm20/DCGAN-tensorflow>

GAN – 二次元人物頭像鍊成

100 rounds



GAN – 二次元人物頭像鍊成



1000 rounds

GAN – 二次元人物頭像鍊成

2000 rounds



GAN – 二次元人物頭像鍊成

5000 rounds



GAN – 二次元人物頭像鍊成



10,000 rounds

GAN – 二次元人物頭像鍊成

20,000 rounds



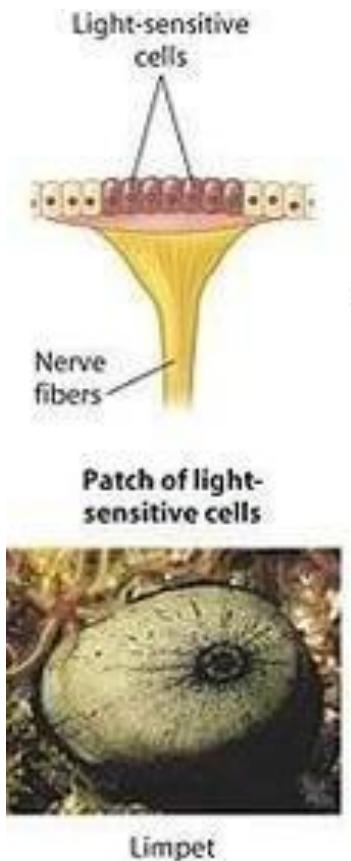
GAN – 二次元人物頭像鍊成

50,000 rounds

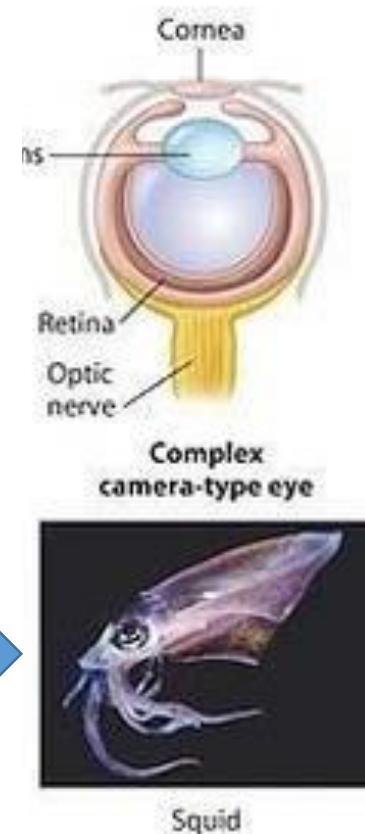


Why GAN is hard to train?

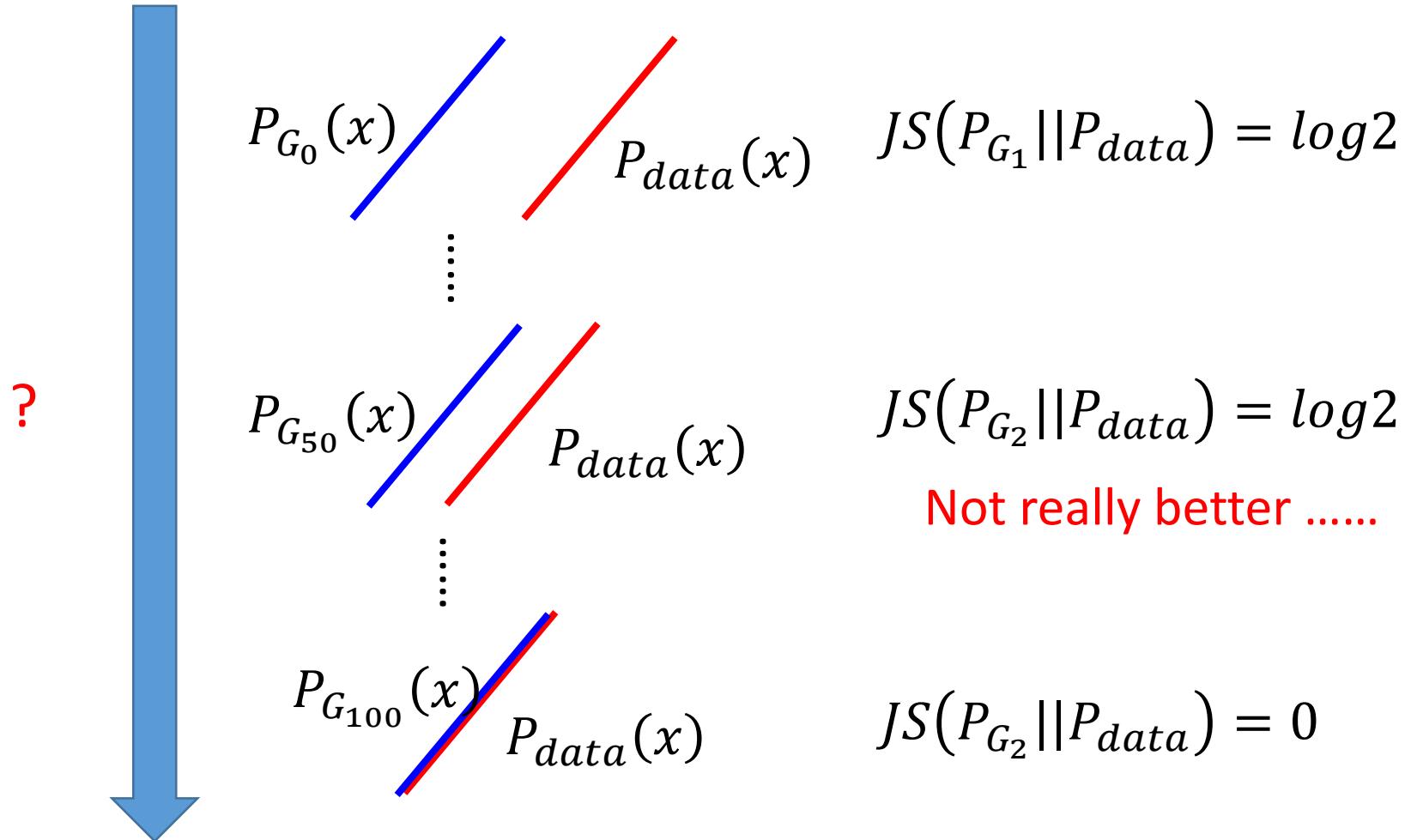
回到演化的比喻



Better

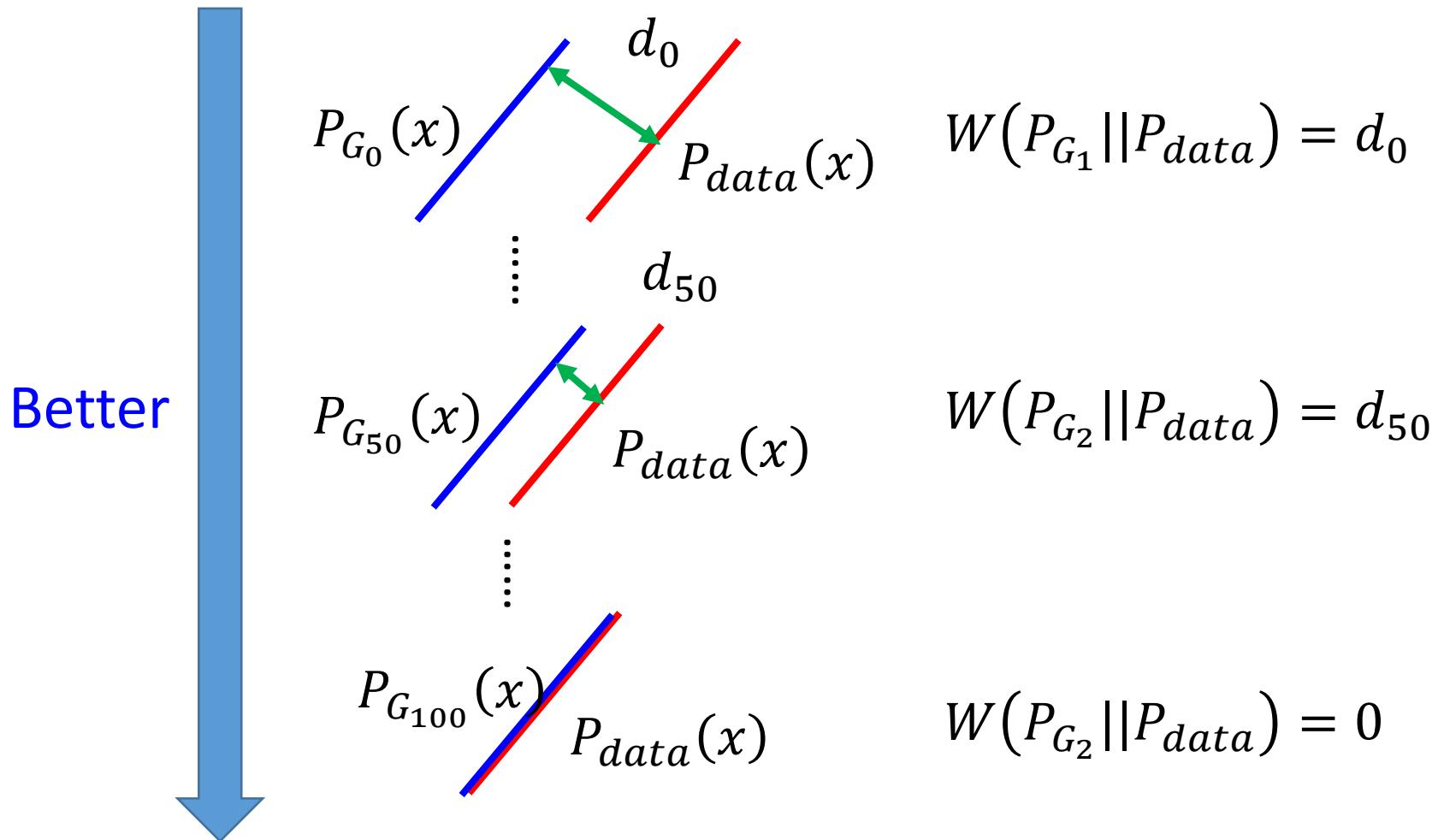


Why GAN is hard to train?

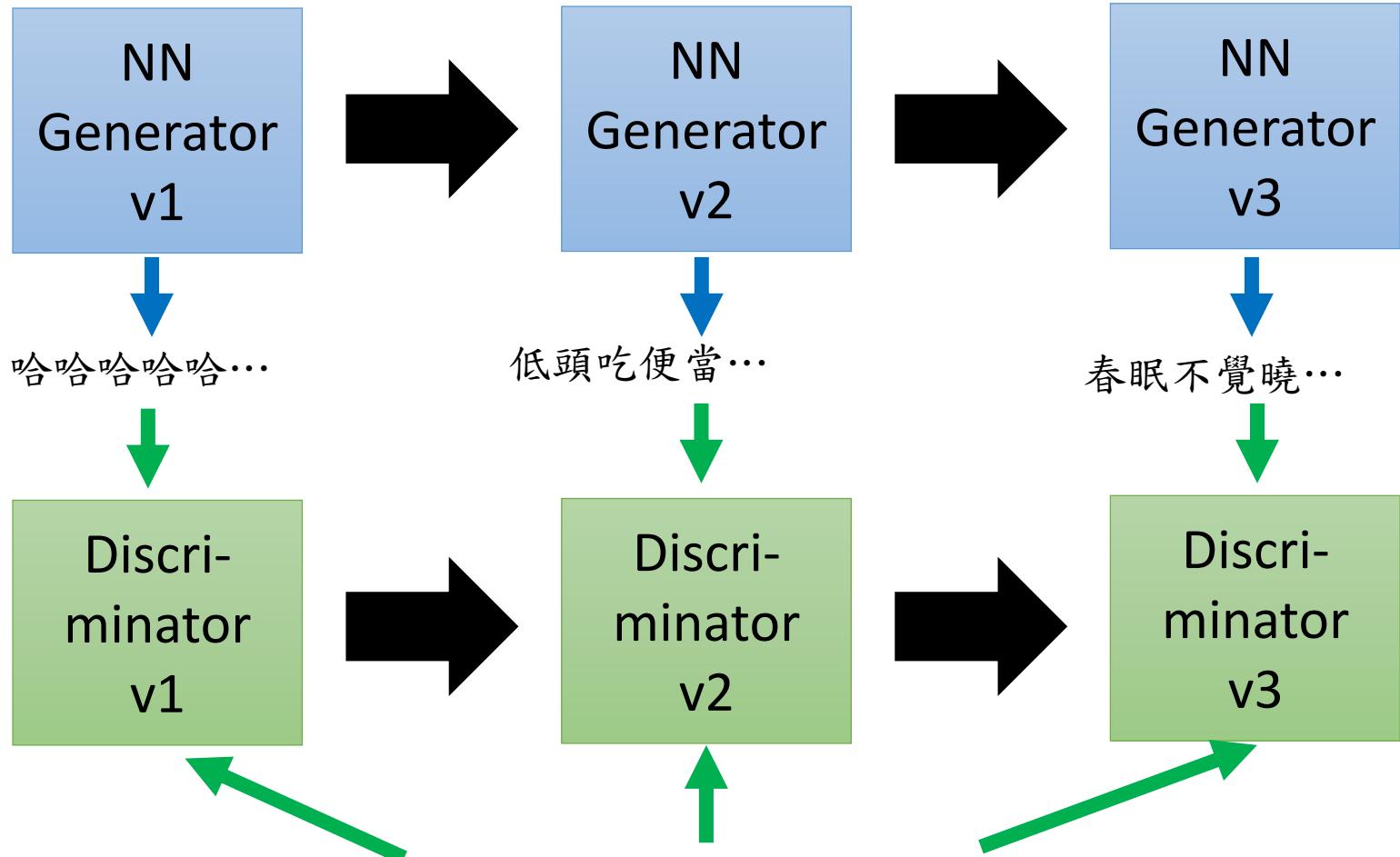


WGAN

Using Wasserstein distance instead of JS divergence



WGAN – 唐詩鍊成



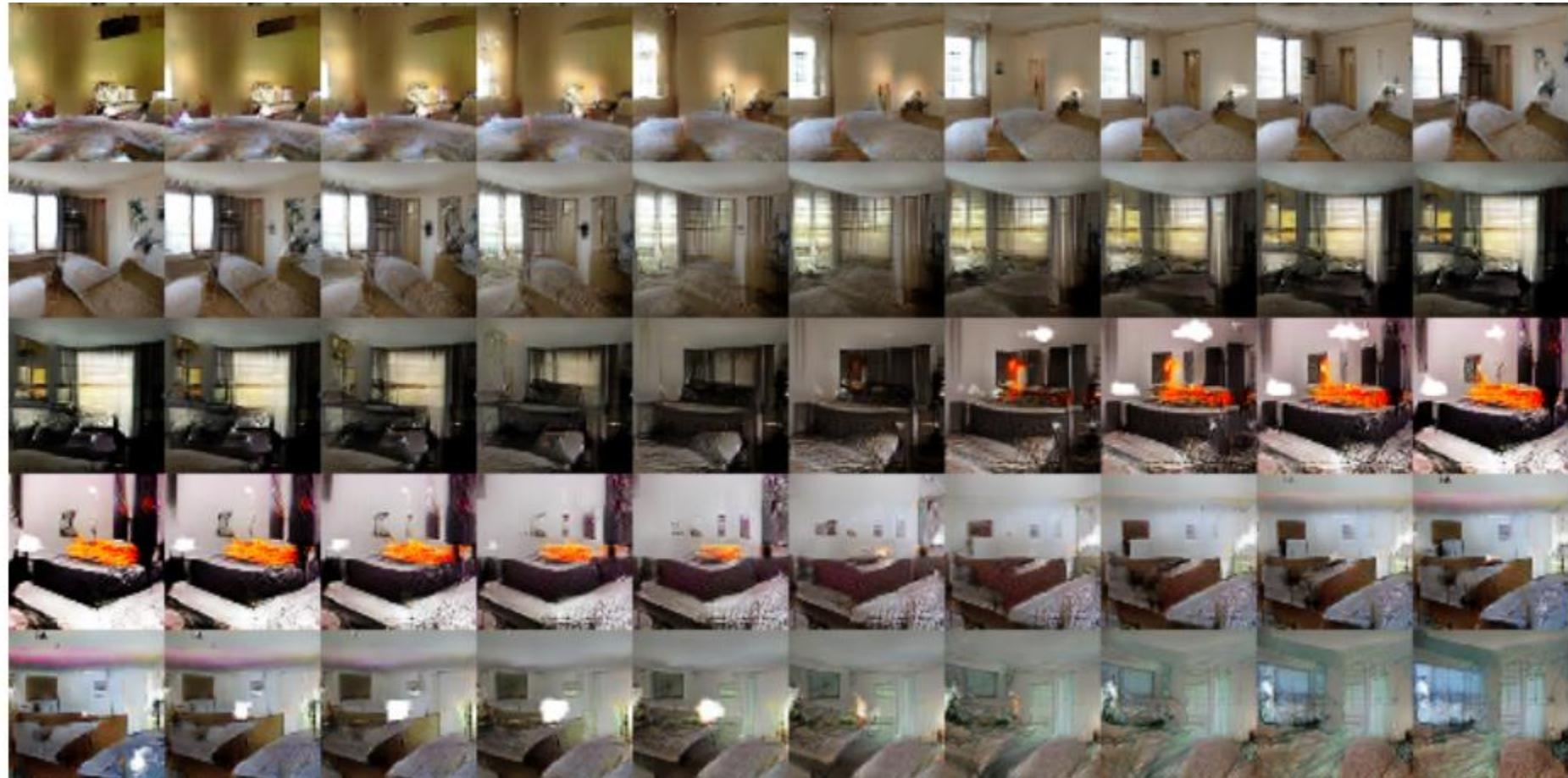
Real poems: 床前明月光，疑似地上霜，舉頭望明月，低頭思故鄉。

由李仲翊同學提供實驗結果
Random generated

WGAN – 唐詩鍊成

- 升雲白遲丹齋取，此酒新巷市入頭。黃道故海歸中後，不驚入得韻子門。
- 據口容章蕃翎翎，邦貸無遊隔將毬。外蕭曾臺遶出畧，此計推上呂天夢。
- 新來寶伎泉，手雪泓臺蓑。曾子花路魏，不謀散薦船。
- 功持牧度機邈爭，不躡官嬉牧涼散。不迎白旅今掩冬，盡蘸金祇可停。
- 玉十洪沄爭春風，溪子風佛挺橫鞋。盤盤稅焰先花齋，誰過飄鶴一丞幢。
- 海人依野庇，為阻例沉迴。座花不佐樹，弟闌十名儂。
- 入維當興日世瀕，不評皺。頭醉空其杯，駸園凋送頭。
- 鉢笙動春枝，寶叅潔長知。官爲密爛去，絆粒薛一靜。
- 吾涼腕不楚，縱先待旅知。楚人縱酒待，一蔓飄聖猜。
- 折幕故蠩應韻子，徑頭霜瓊老徑徑。尚錯春鏘熊悽梅，去吹依能九將香。
- 通可矯目鷁須淨，丹迤掣花一抵嫖。外子當目中前醒，迎日幽筆釣弧前。
- 庭愛四樹人庭好，無衣服仍繡秋州。更怯風流欲鵝雲，帛陽舊據畝婷儻。

Moving on the code space

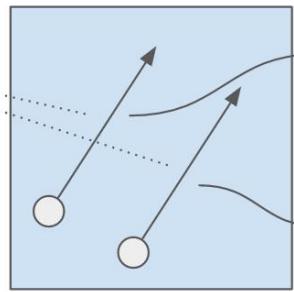


Alec Radford, Luke Metz, Soumith Chintala, Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks, ICLR, 2016

Moving on the code space

- Ref: <http://qiita.com/matty/items/e5bfe5e04b9d2f0bbd47>

長髪化ベクトル



一番左のキャラクターが元画像で、
右に行くほど長髪化ベクトルを強く足している



元画像



-赤髪 + 金髪



-赤目 + 青目



+制服 + セーラー

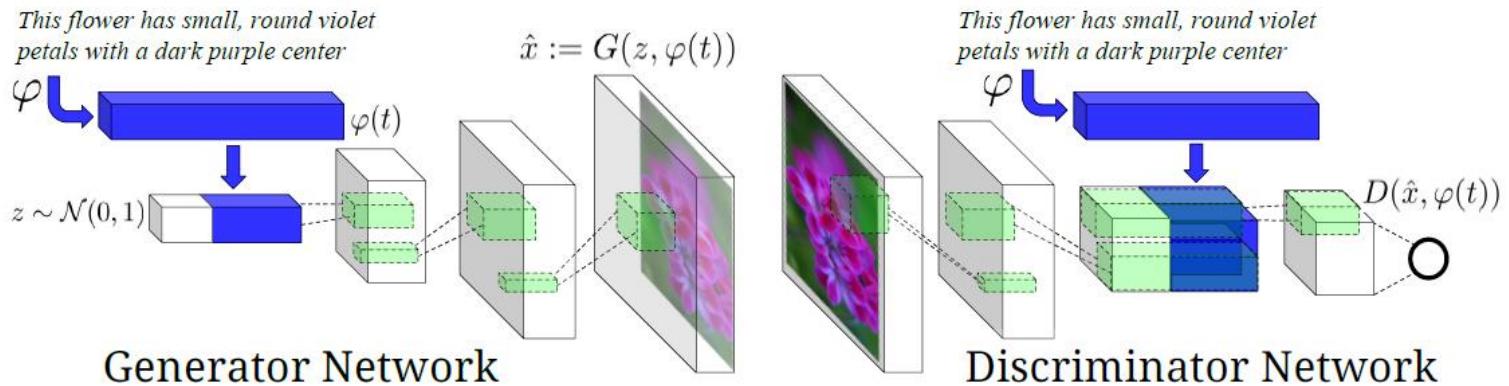


+笑顔 + 口開き



+青背景

Text to Image



Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, Honglak Lee, “Generative Adversarial Text-to-Image Synthesis”, ICML 2016

Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaolei Huang, Xiaogang Wang, Dimitris Metaxas, “StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks”, arXiv preprint, 2016

Scott Reed, Zeynep Akata, Santosh Mohan, Samuel Tenka, Bernt Schiele, Honglak Lee, “Learning What and Where to Draw”, NIPS 2016

Text to Image

"red flower with black center"



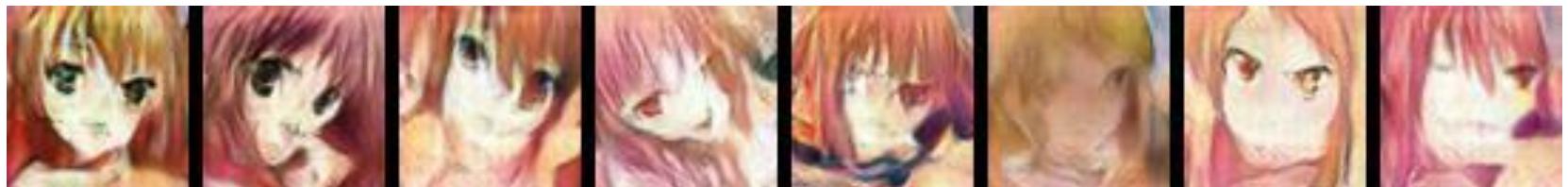
Caption	Image
this flower has white petals and a yellow stamen	A 4x4 grid of images showing different types of white flowers with prominent yellow centers, such as daisies or marguerites.
the center is yellow surrounded by wavy dark purple petals	A 4x4 grid of images showing purple flowers with a yellow center and dark purple, wavy petals, characteristic of certain species of pansies or violas.
this flower has lots of small round pink petals	A 4x4 grid of images showing pink flowers composed of numerous small, rounded petals, resembling carnations or similar carnation-like flowers.

Text to Image

由 曾柏翔 同學
提供實驗結果

- E.g. 根據文字敘述畫出動漫人物頭像

Red hair, long hair



Black hair, blue eyes



Blue hair, green eyes

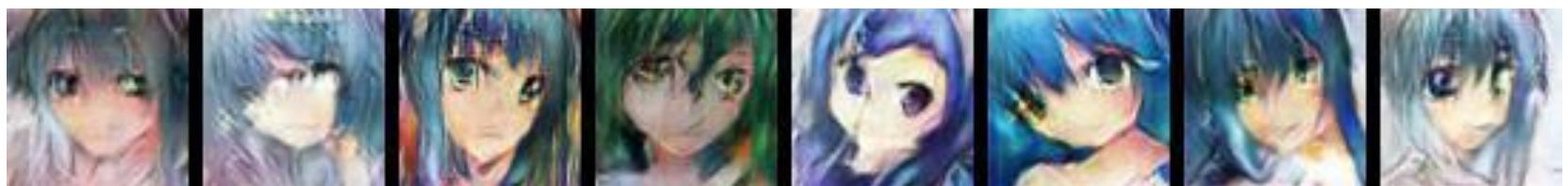
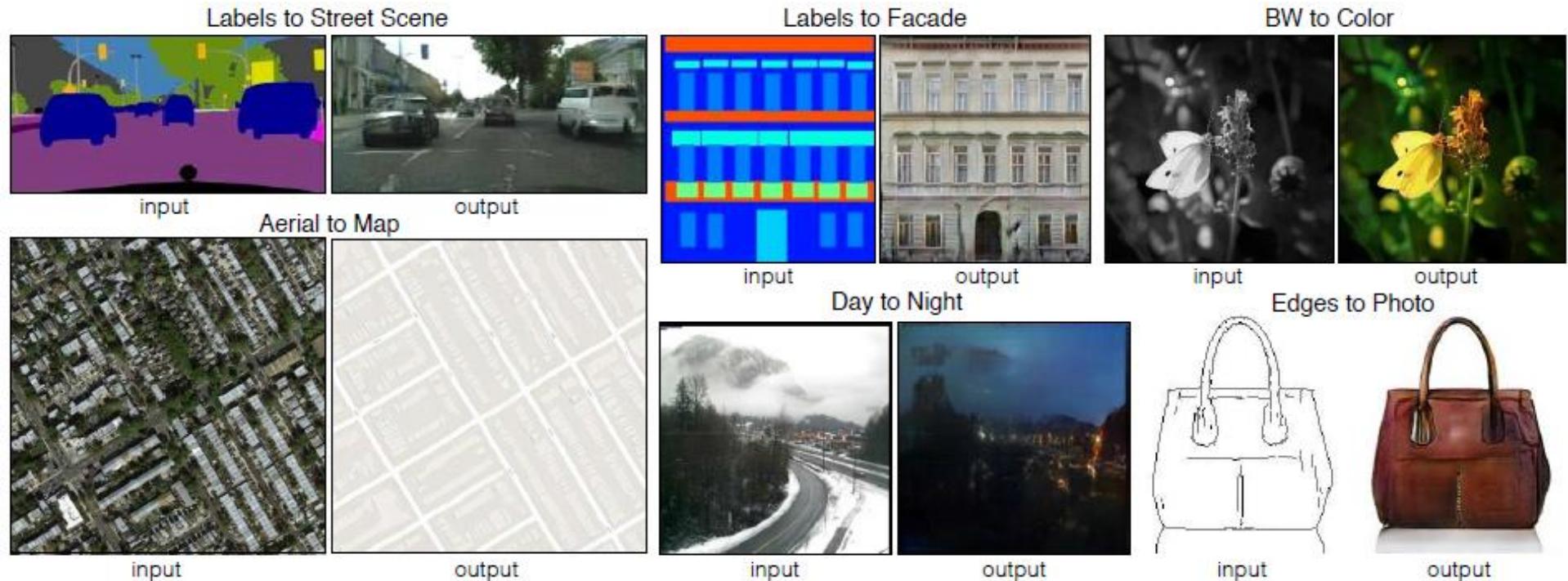
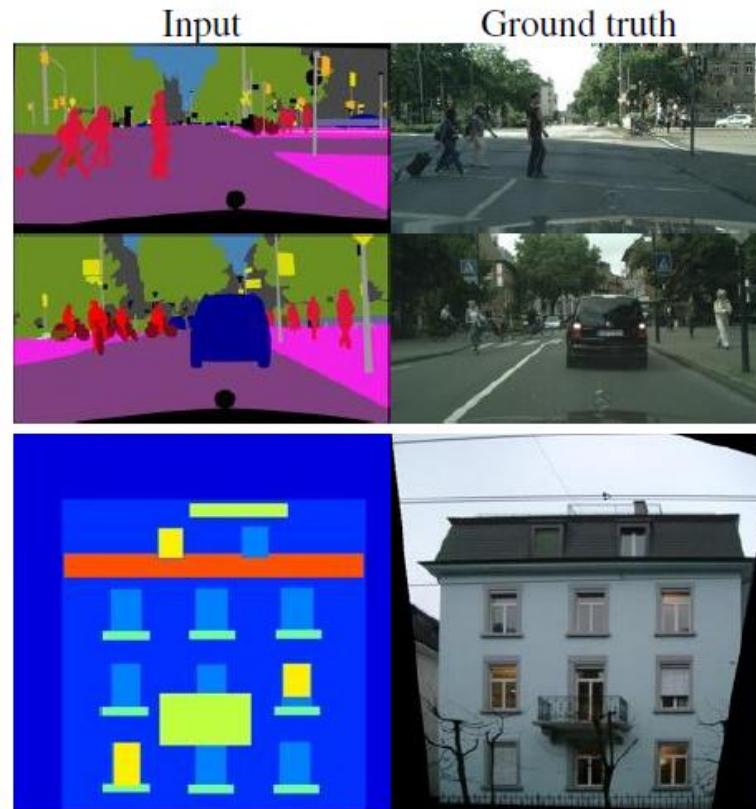


Image-to-image Translation



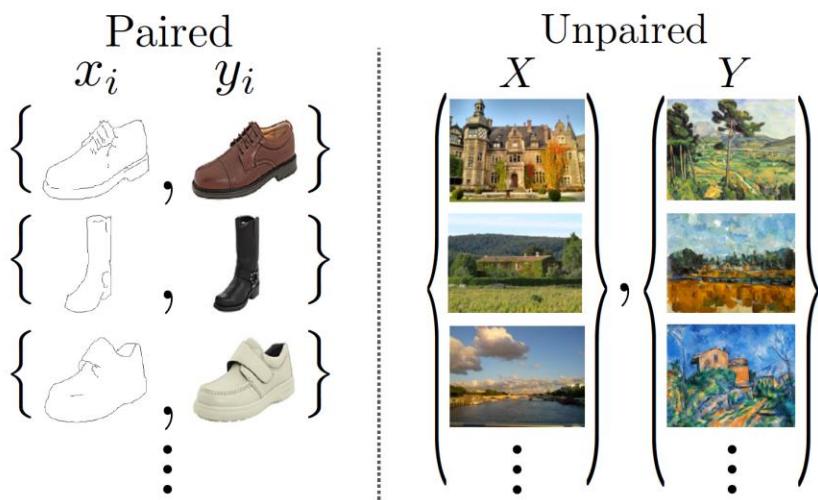
Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, Alexei A. Efros, "Image-to-Image Translation with Conditional Adversarial Networks", arXiv preprint, 2016

Image-to-image Translation - Results



Cycle GAN

<https://arxiv.org/abs/1703.10593>



Monet Photos



Monet → photo

Zebras Horses



zebra → horse

Summer ↗ Winter



summer → winter

photo → Monet

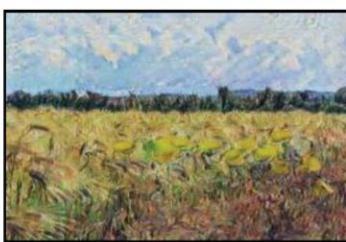
A composite image showing two animals side-by-side. On the left is a brown horse in mid-gallop, its front legs lifted. On the right is a black and white zebra in a similar galloping pose. The two animals are separated by a thin horizontal line.

horse → zebra

winter → summer



Monet



Van Gogh

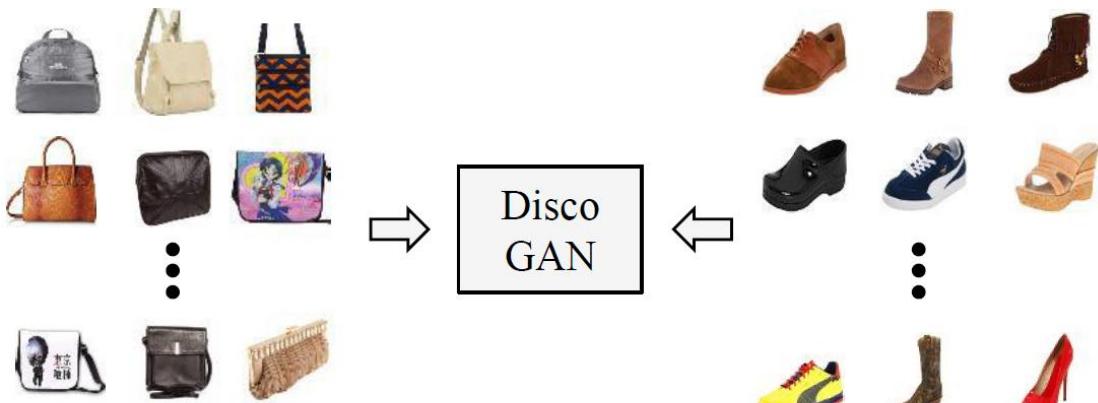


Cezanne



Ukiyo-e

Disco GAN



(a) Learning cross-domain relations **without any extra label**



(b) Handbag images (input) & **Generated** shoe images (output)



(c) Shoe images (input) & **Generated** handbag images (output)

機械学習で美少女化～あるいはNEW GAME! の世界

- <http://qiita.com/Hiking/items/8d36d9029ad1203aac55>



So many GANs

..... Just name a few

Modifying the Optimization of GAN

fGAN

WGAN

Least-square GAN

Loss Sensitive GAN

Energy-based GAN

Boundary-seeking GAN

Unroll GAN

.....

Different Structure from the Original GAN

Conditional GAN

Semi-supervised GAN

InfoGAN

BiGAN

Cycle GAN

Disco GAN

VAE-GAN

.....

Acknowledgement

- 感謝 Ryan Sun 來信指出投影片上的錯字