期末复习总结

2020.11.18

考核方式

- 最终成绩由以下组成：（末尾淘汰8%）
  - 作业：10%
    - 晚1天交，成绩9折；晚3天交，成绩8折；超过3天，不接受。
  - 4次实验：20%（Python, Tensorflow, Keras）
  - Project（组队）：10%
    - 3人一组
  - 期中考试：机器学习/深度学习项目的开发（比如，Kaggle入门竞赛题），2-3人一组，20%
  - 期末考试：半开卷，40%
  - 加分：
    - 小助教：实验课上帮助学生完成编程，额外奖励+1分/次
    - 优秀实验/作业的分享（1分）
  - 扣分：随机抽查考勤，缺勤一次扣一分。

期末考试题型

- 单选题（20分：10×2’）

- 简答题/计算题（45分）
  - 计算题：梯度的计算，以及神经网络中参数量的预估。
  - 简答题：涵盖范围较广
- 综合分析题（35分）
  - 给定项目案例，针对具体问题，进行相应解答。

机器学习中的关键术语

- 机器学习中模型的种类：(从应用场景的角度划分)
  - 监督学习、非监督学习、半监督学习、强化学习、迁移学习
- 数据：
  - 3类数据集：
    - 训练集：利用训练集上的误差来训练模型
    - 验证集：利用验证集上的误差 来挑出备选模型，确定各个备选模型的最佳超参数，并从中挑选出最佳模型
    - 测试集：用于评估模型的表现
  - 样本：（输入，输出），一条数据为一个样本
  - 标签/标注：样本数据中的输出项
- 2个计算过程：
  - 学习/训练/优化：目标是找到最佳函数来表达输入输出之间的关系
    - 理想的最佳函数，应在优化和泛化的平衡点处，应刚好在欠拟合和过拟合的界线上
  - 推理/预测/泛化：将找到的最佳函数应用到测试数据/真实数据上，做出推理/预测。

机器学习中的关键术语

- 两类函数：
  - 最佳函数：（模型训练的终极目标是找到这个函数）
  - 成本/代价/损失函数：用来定义训练过程中找到的函数的好坏程度
- 两类参数：
  - 参数：模型训练过程中自动找到
  - 超参数：程序员来选择/指定
    - 半自动：网格搜索、随机搜索
    - 手动：程序员设定
- 训练过程中，模型的状态：
  - 欠拟合
  - 过拟合
  - 正好

机器学习中的关键术语

- 两个评估指标：
  - 模型评估标准（在训练数据集和验证数据集上）：优化指标，即损失函数。通常用于模型的训练中。
    - 损失函数的取值 越小，则越好。
  - 模型测试标准（在测试集上）：验收指标。用于测试模型的泛化性能
    - 比如：样本平衡的分类问题中常用的验收指标 **ROC AUC**
    - 但是， **ROC AUC**不能直接被优化，故不能作为损失函数，而是将**ROC AUC**的替代指标 交叉熵作为优化指标，即损失函数。
- 评估方法：
  - 留出验证集
  - 交叉验证**(Cross Validation)**

- 机器学习分为两个阶段：
  - 训练：“三步曲” on training set
    - 定义Model
    - 定义Loss/cost/error/objective function
    - 如何找到Model中的最佳function：利用梯度下降来迭代调整参数，以使得损失函数达到最小值
      - 反向传播
  - 预测：on Dev set and testing set
    - 前向传播：预测输出，计算Loss

回顾：机器学习项目的通用工作流程

1 定义问题：软件架构设计、确定评价指标

2 获取数据：自动化方式

3 研究数据：可视化方式，相关性研究等

4 准备数据：数据清理、特征选择及处理

5 研究模型：确定验证集上的评估方法、列出可能的模型并训练，选择最有希望的3~5个模型

6 微调模型：寻找最佳超参数，模型融合，评估泛化性能

7 展示解决方案：将工作进行文档化总结展示

8 启动、监视、维护系统：投入使用

线性回归

- 什么是机器学习？
  - 对于某类任务T和性能度量P，一个计算机程序被认为可以从经验E中学习是指，通过经验E改进后，它在任务T上由性能度量P衡量的性能有所提升
  - T, P, E 是什么？

- 线性回归model形如：$Y = W^T X + b$

如何训练模型？——利用梯度下降

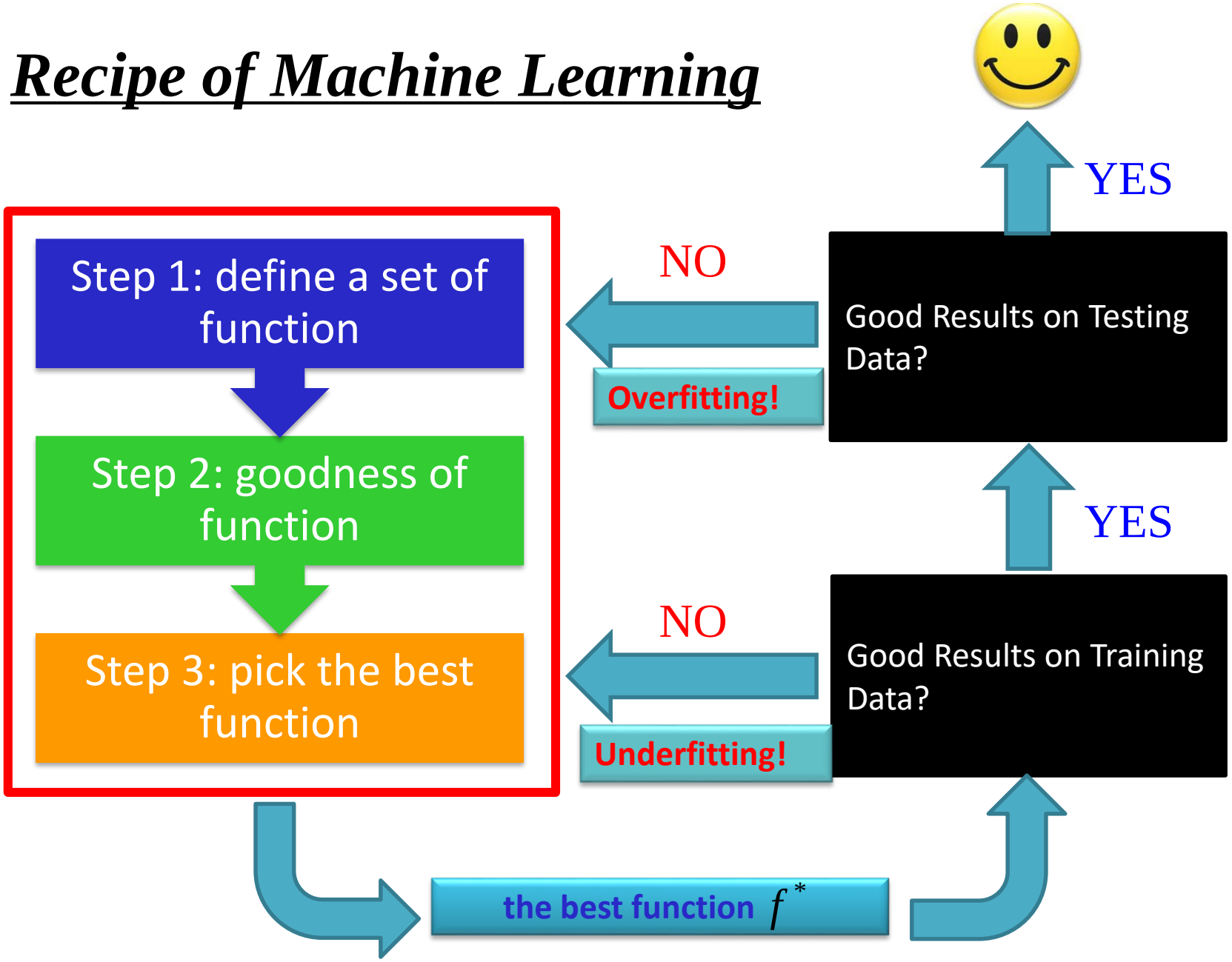
- 梯度下降(Gradient Descent)是一种非常通用的优化算法，适合在特征个数非常多，训练实例非常多，内存无法满足要求的时候使用。
- 如何利用梯度下降来更新参数，以便找到最佳参数？
- 梯度下降的Tips

Gradient Descent Tips

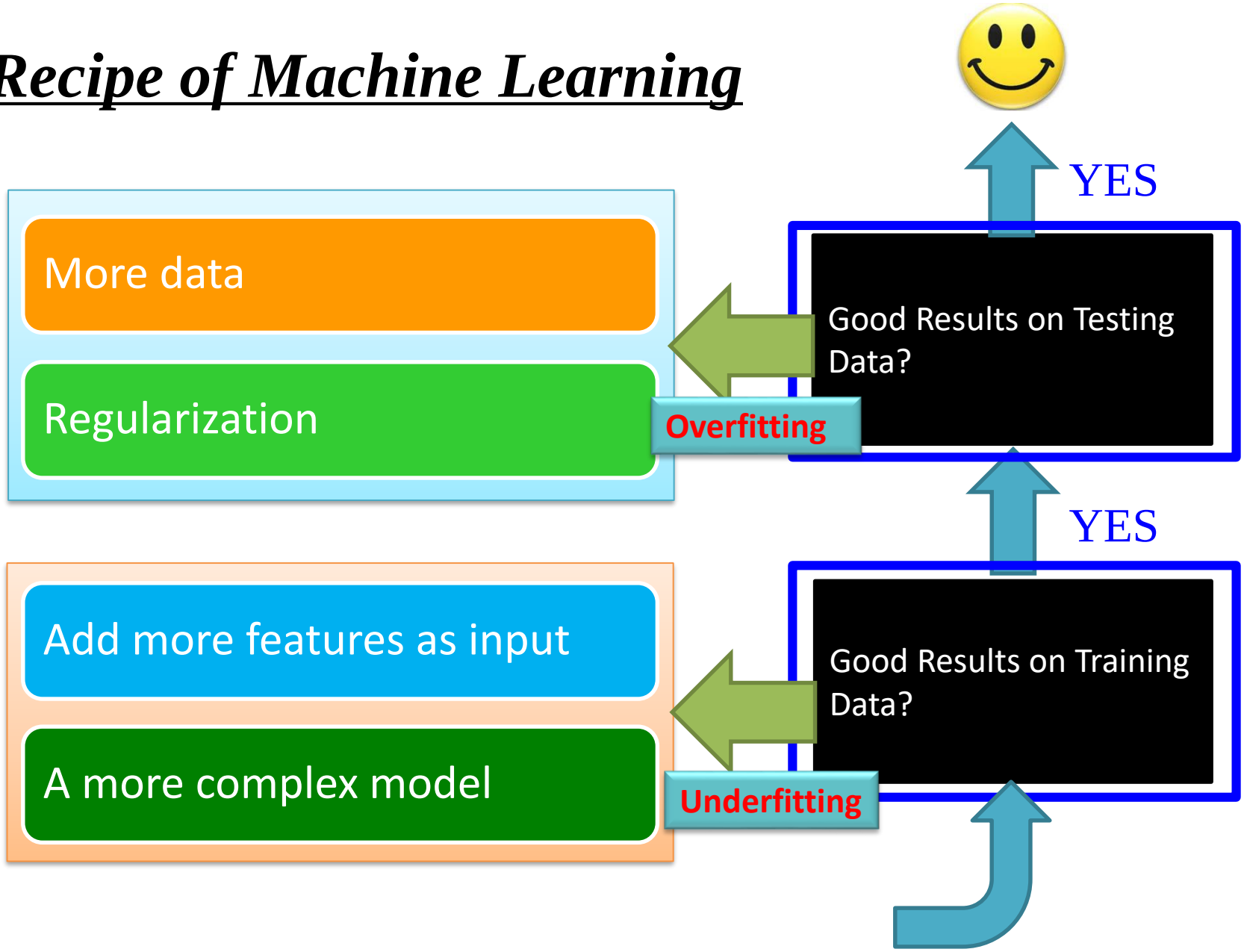
- Tip 1: Tuning your learning rates
  - 给定实验结果，如何判断学习率是过大，还是过小？
  - 学习率过大或过小，将带来什么影响？
- Tip 2: Feature Scaling/Normalization
  - Make the training faster
- Tip 3: Variants of Gradient Descent
  - Stochastic Gradient Descent, Mini-batch Gradient Descent
  - 不同的变体，训练效果是不同的。

- 欠拟合问题和过拟合问题（本质：误差分析）
  1. 什么是欠拟合？什么是过拟合？
  2. 如何判断欠拟合和过拟合？
     - 利用训练误差和验证误差
     - 利用训练集和验证集上的学习曲线
  3. 如何解决？

Recipe of Machine Learning

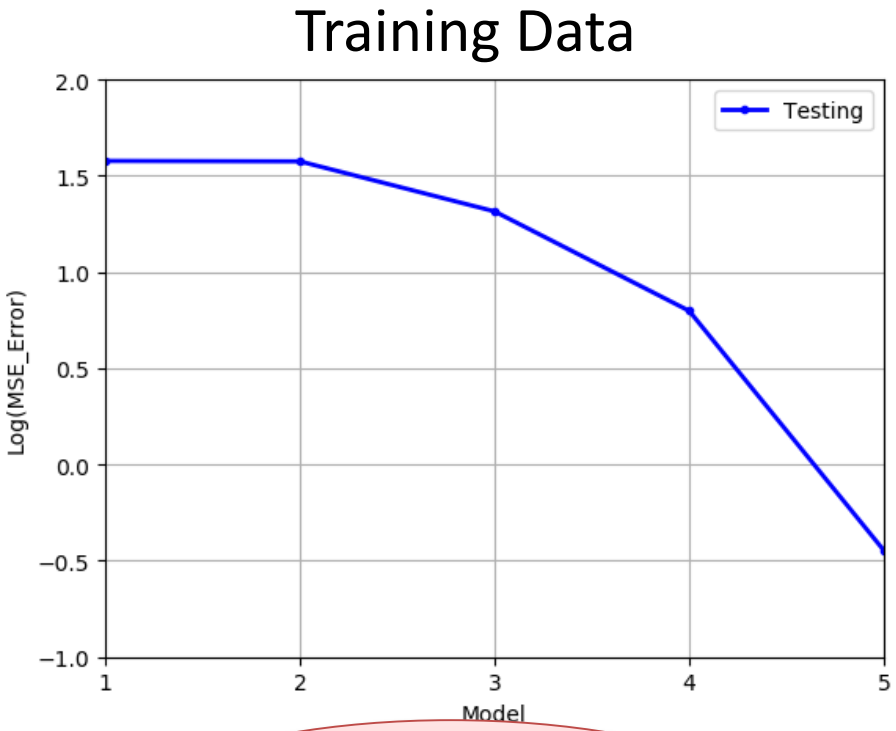


Recipe of Machine Learning



Model Selection

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



A more complex model yields lower error on training data.

If we can truly find the best function

正则化(Regularization)

- 正则化技术
  - 是解决过拟合问题的通用技术。
  - 正则化一个模型，是指对该模型增加限制，使得该模型自由度减少
  - 比如，正则化一个多项式模型，一个简单的方法就是减少多项式的阶数。
- 如何实现正则化？
  - 在损失函数中，增加正则项，以约束模型中参数的权重。
    - L’(θ) = L(θ) + 正则项

$$\|\theta\|_1 = |w_1| + |w_2| + \mathrm{K}$$

$$\|\theta\|_2 = (w_1)^2 + (w_2)^2 + \mathrm{K}$$

正则化(Regularization)

- L’(θ) = L(θ) + 正则项

$$\|\theta\|_1 = |w_1| + |w_2| + \mathrm{K}$$

$$\|\theta\|_2 = (w_1)^2 + (w_2)^2 + \mathrm{K}$$

- 寻找最佳参数θ，使得新的损失函数L’(θ) 最小化
  - Find **a set of weight** not only minimizing original cost L(θ) but also close to zero

- 注意：
  - 正则项只有在训练过程中才会被加到损失函数，即训练过程中使用L’(θ) 来计算梯度，以更新参数。
  - 一旦训练完成，应该使用没有正则化的损失函数L(θ)去测量评价模型的表现
  - 偏置项（即常数项）没有正则化

正则化(Regularization)

- 三种正则化方法：
  - 岭（Ridge）回归（ **L2-norm** ）：Weight Decay

$$L'(\theta) = L(\theta) + \lambda \cdot \|\theta\|_2 \quad , \quad \|\theta\|_2 = (w_1)^2 + (w_2)^2 + \mathrm{K}$$

  - Lasso回归（ **L1-norm** ）

$$L'(\theta) = L(\theta) + \lambda \cdot \|\theta\|_1 \quad , \quad \|\theta\|_1 = |w_1| + |w_2| + \mathrm{K}$$

  - ElasticNet（ **L1 + L2** ）

$$L'(\theta) = L(\theta) + \lambda \cdot [\rho \cdot \|\theta\|_1 + (1 - \rho) \cdot \|\theta\|_2]$$

超参数：正则化因子λ

误差来源分析

- Testing Error = Bias error + Variance Error.
  - Bias error ≈ 训练集上的错误率 （非正式地, By 吴恩达）
    - 训练集上的错误率 = avoidable error + unavoidable error
  - Variance error ≈ 开发集（或测试集）上的表现比训练集上差多少 （非正式地, By 吴恩达）
    - 用“开发误差 - 训练误差”来衡量

- 欠拟合： 高bias error, 低 variance error
- 过拟合： 低bias error, 高variance error
- Good： 低bias error, 低variance error

Classification

- 为什么不直接利用线性回归模型来解决分类问题？
- 二分类问题中，逻辑回归模型训练“三步曲”
  - 分类问题中，为什么使用交叉熵（cross entropy）而不是MSE作为损失函数？
- 多分类问题中，使用Softmax回归模型
  - 对比 逻辑回归模型，比较异同
- 分类器的性能评估
    - 混淆矩阵，精度和召回率，PRC， ROC
- 利用混淆矩阵进行错误分析

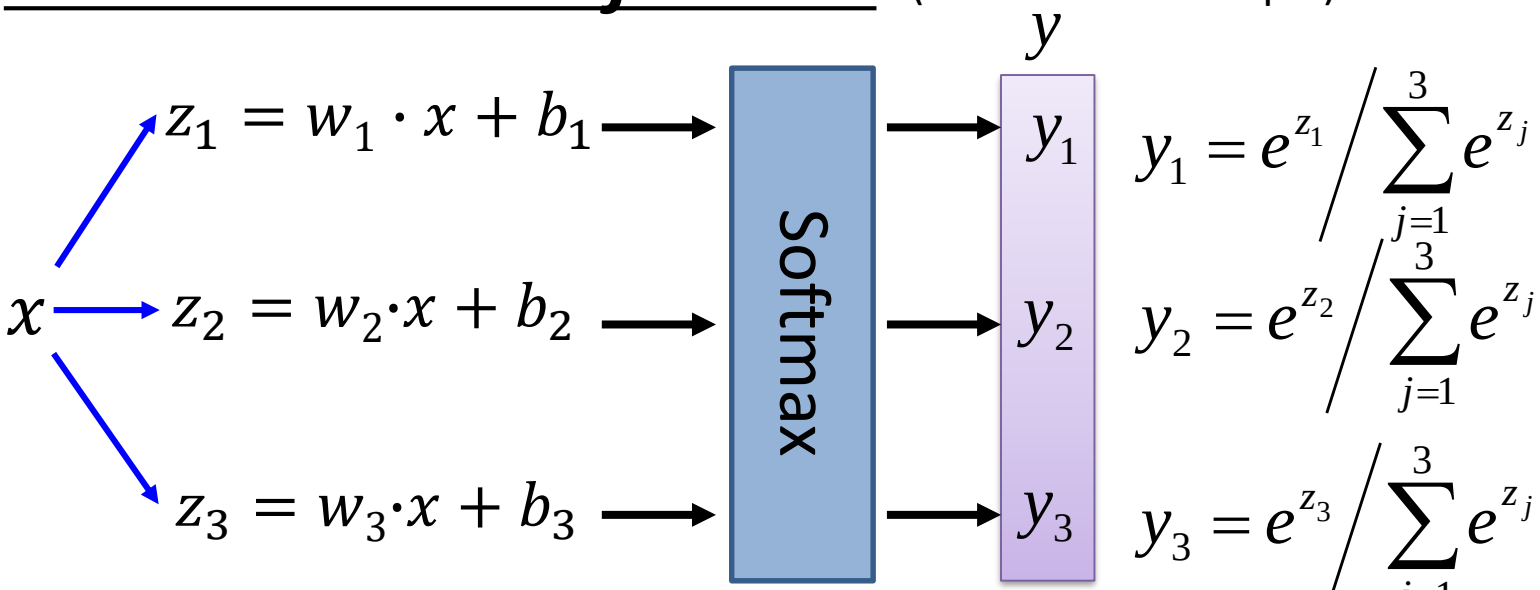
	Logistic Regression	***Linear Regression***
Step 1:	$f_{w,b}(x) = \sigma\left(\sum_i w_i x_i + b\right)$	$f_{w,b}(x) = \sum_i w_i x_i + b$
	Output: between 0 and 1	Output: any value
Step 2:	Training data: $\left(x^{(n)}, \hat{y}^{(n)}\right)$ $\hat{y}^{(n)}$: 1 for class 1, 0 for class 2 $L(f) = \sum_n l\left(f\left(x^{(n)}\right), \hat{y}^{(n)}\right)$	Training data: $\left(x^{(n)}, \hat{y}^{(n)}\right)$ $\hat{y}^{(n)}$: a real number $L(f) = \frac{1}{2}\sum_n \left(f\left(x^{(n)}\right) - \hat{y}^{(n)}\right)^2$

Cross entropy:

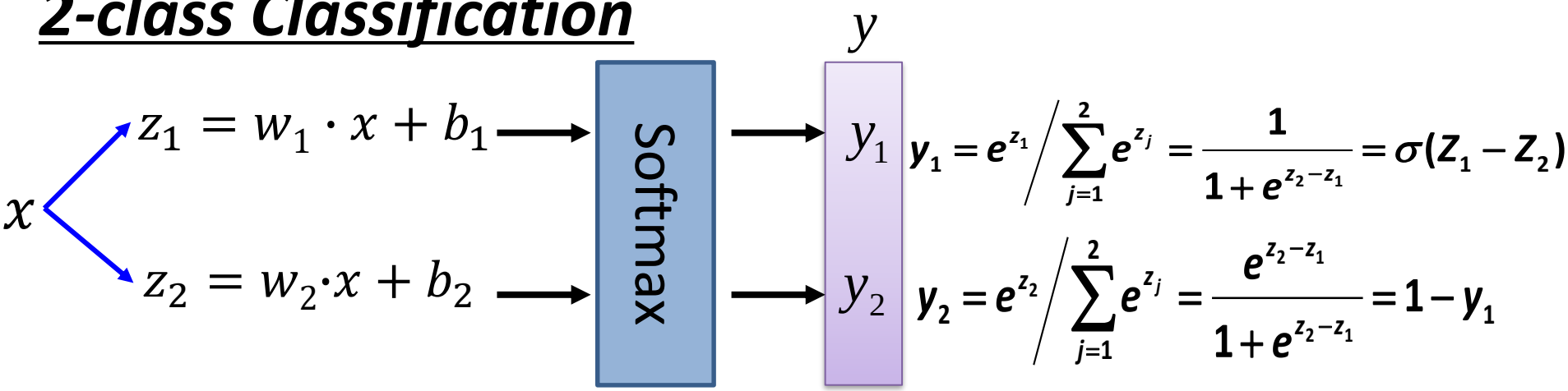
$$l\left(f\left(x^{(n)}\right), \hat{y}^{(n)}\right) = -\left[\hat{y}^{(n)} ln f\left(x^{(n)}\right) + \left(1 - \hat{y}^{(n)}\right) ln\left(1 - f\left(x^{(n)}\right)\right)\right]$$

[Bishop, P209-210]

Multi-class Classification (3 classes as example)



2-class Classification



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

模型融合/集成学习

- 什么是模型融合/集成学习？
  - 模型融合的目标；聚合策略；模型融合方法的分类
- 模型融合/集成学习肯定有效吗？
- 模型融合的常用方案有哪些？
  - Voting
  - Stacking

  基预测器可以为不同种类
  - Bagging
    - 随机森林
  - Boosting
    - AdaBoost
    - Gradient Boosting: GBRT, XGBoost, LightGBM

  基预测器都是同一类
- Kaggle比赛案例展示
  - 帮助快速熟悉支持向量机、决策树、随机森林的代码实现
  - 理解模型融合所带来的效果

深度学习

1 什么是深度学习

2 神经网络的数学基础

- 张量，张量的运算，梯度计算

3 神经网络入门

- 如何训练神经网络？

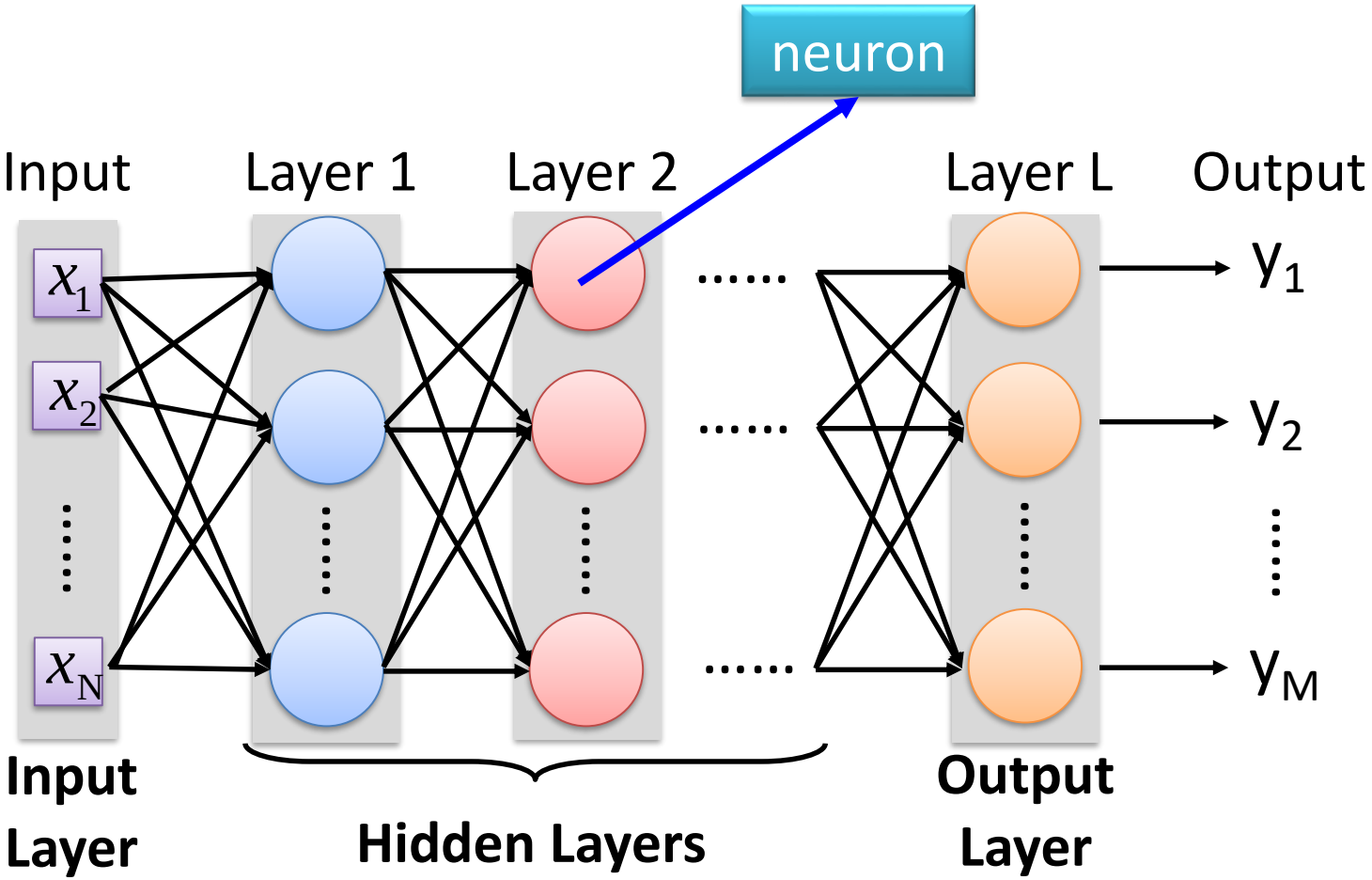
4 神经网络的通用工作流程

- 对比 机器学习的通用工作流程

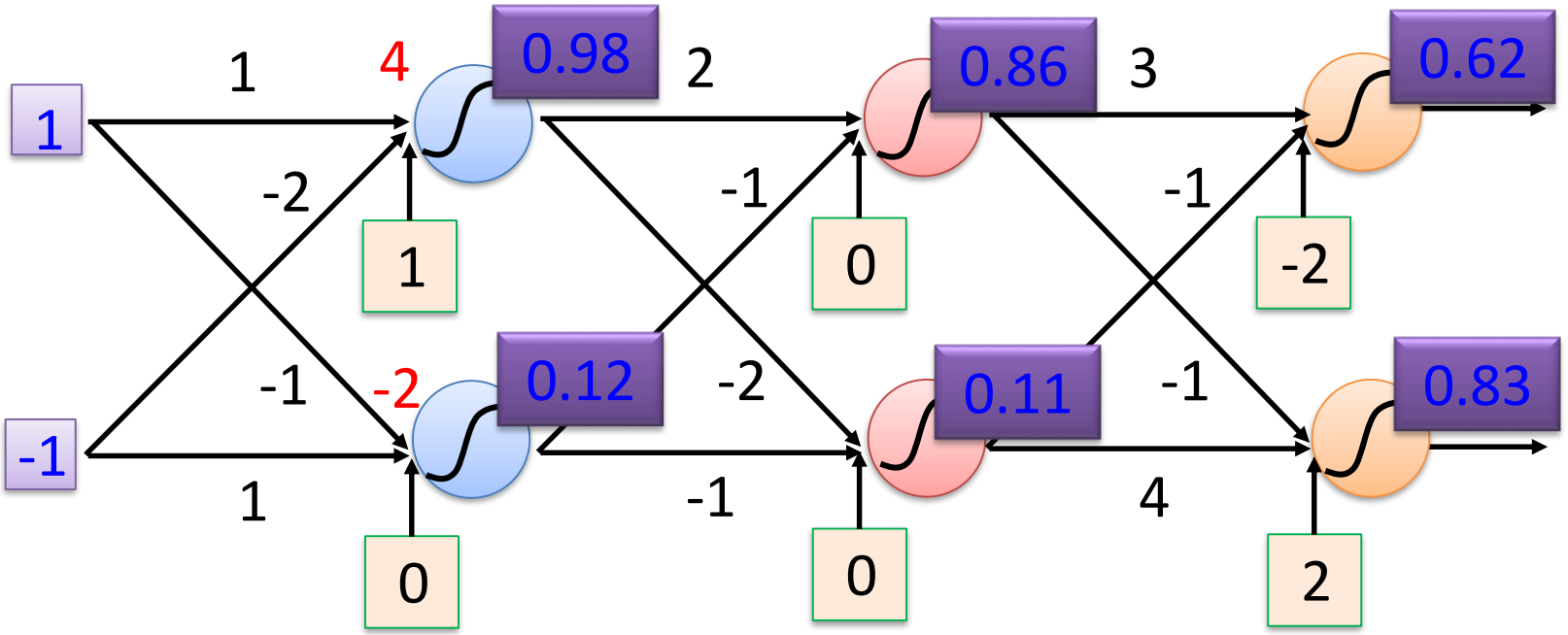
Deep learning

- 训练过程中的“三部曲”，与机器学习中的“三部曲”进行对照比较异同
  - Model定义：定义一个network structure
- 反向传播：用于更新模型参数
  - 会计算梯度（即偏导）
- 前向传播：用于预测输出，计算Loss

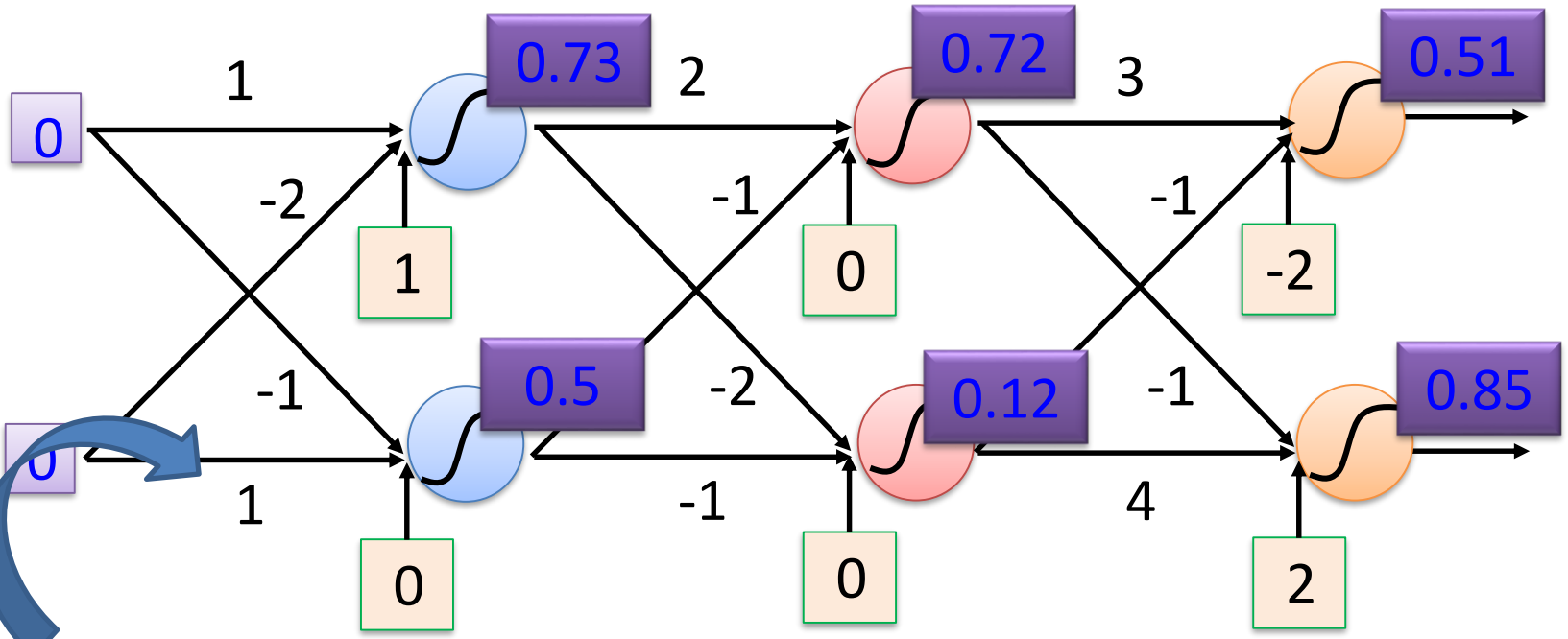
Fully Connect Feedforward Network



Fully Connect Feedforward Network



Fully Connect Feedforward Network

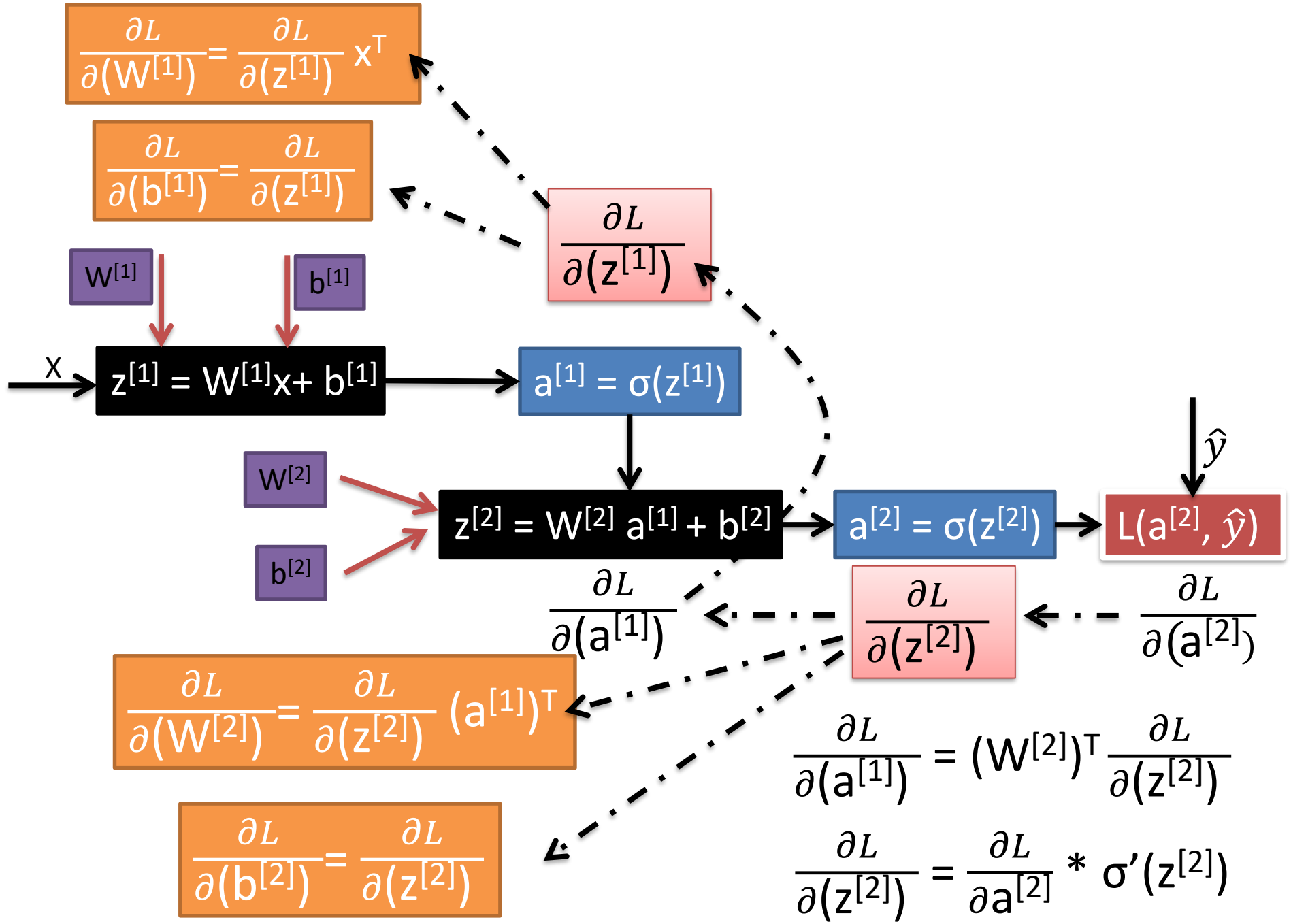


This is a function.

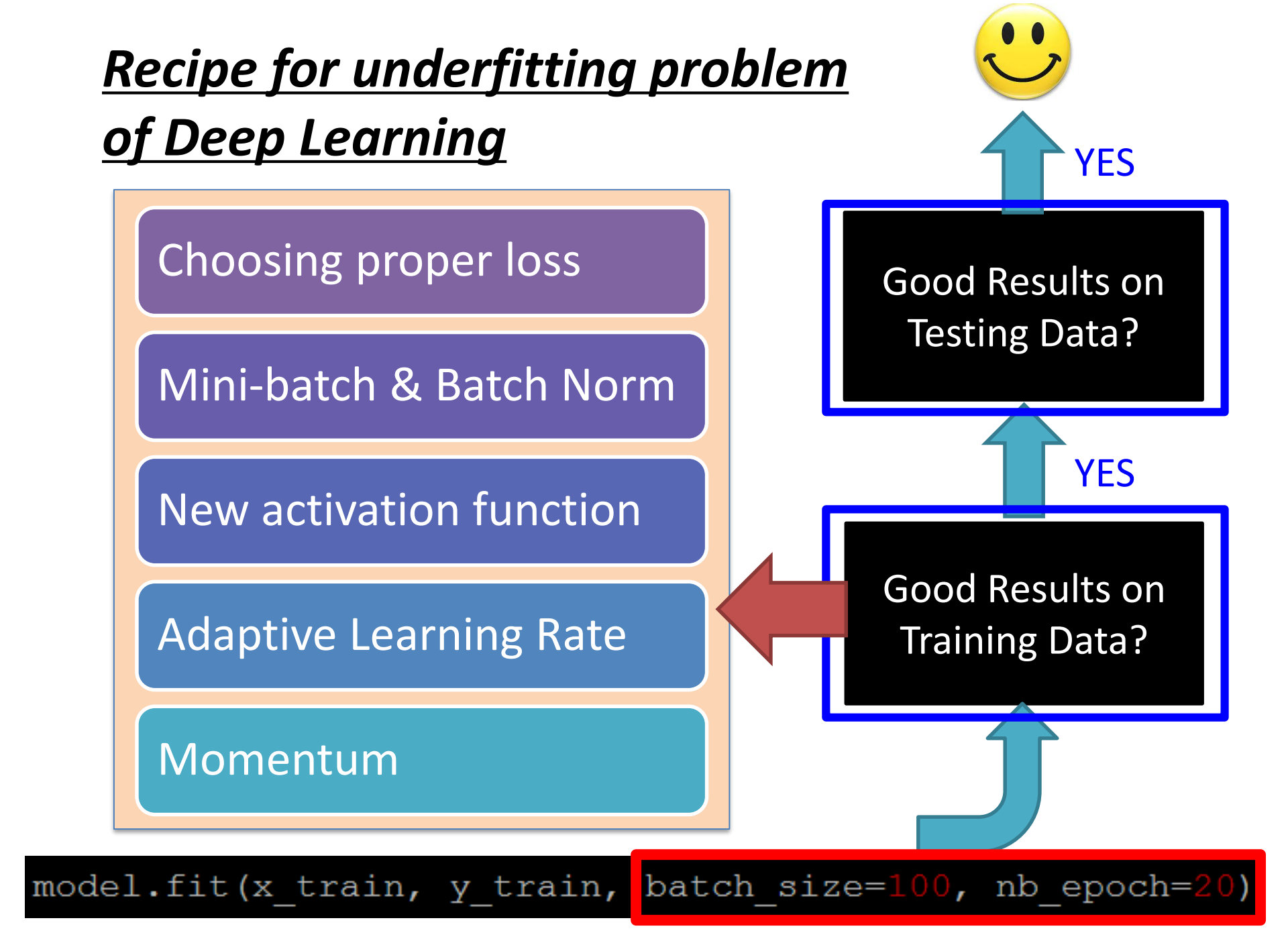
Input vector, output vector

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

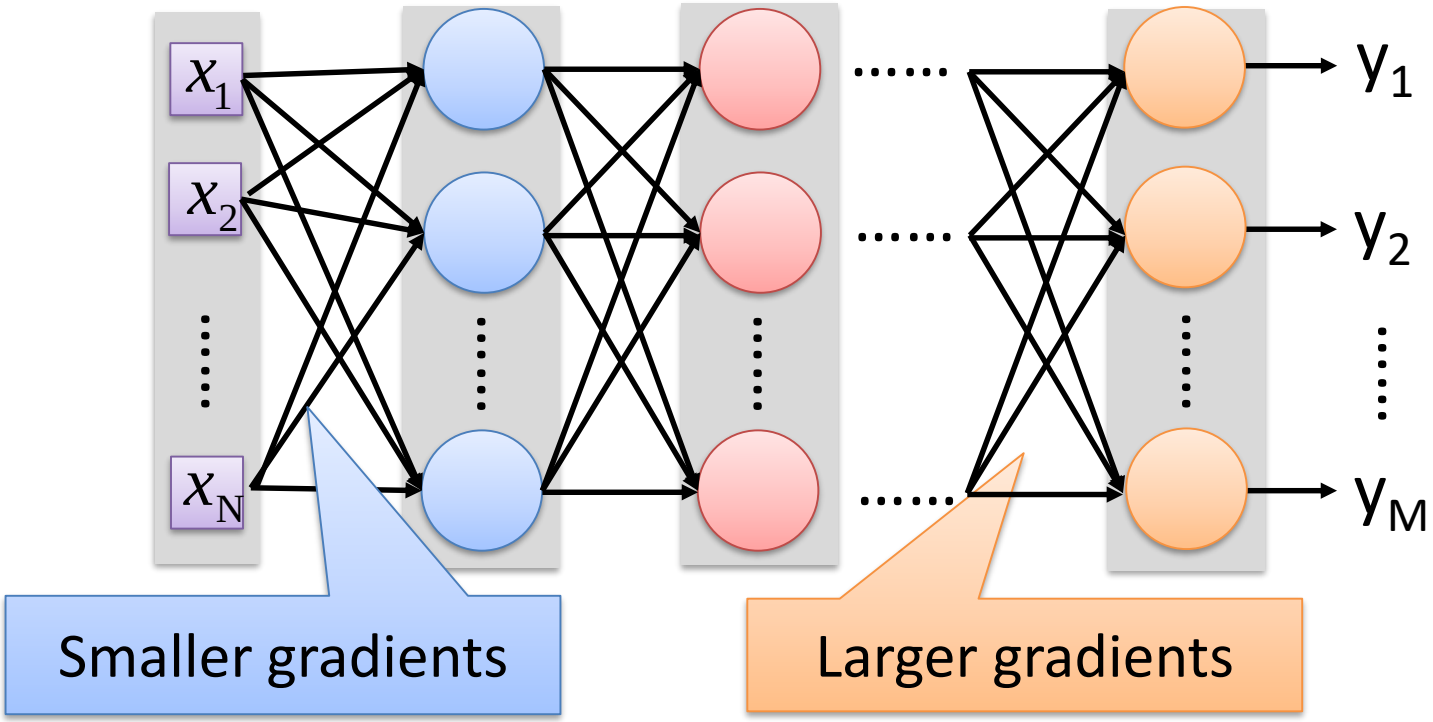
Given network structure, define ***a function set***


$\frac{\partial L}{\partial (W^{[1]})} = \frac{\partial L}{\partial (z^{[1]})} x^T$
$\frac{\partial L}{\partial (b^{[1]})} = \frac{\partial L}{\partial (z^{[1]})}$
$\frac{\partial L}{\partial (z^{[1]})}$
$W^{[1]}$
$b^{[1]}$
x
$z^{[1]} = W^{[1]}x + b^{[1]}$
$a^{[1]} = \sigma(z^{[1]})$
$W^{[2]}$
$b^{[2]}$
$z^{[2]} = W^{[2]} a^{[1]} + b^{[2]}$
$a^{[2]} = \sigma(z^{[2]})$
$\hat{y}$
$L(a^{[2]}, \hat{y})$
$\frac{\partial L}{\partial (a^{[1]})}$
$\frac{\partial L}{\partial (z^{[2]})}$
$\frac{\partial L}{\partial (a^{[2]})}$
$\frac{\partial L}{\partial (W^{[2]})} = \frac{\partial L}{\partial (z^{[2]})} (a^{[1]})^T$
$\frac{\partial L}{\partial (b^{[2]})} = \frac{\partial L}{\partial (z^{[2]})}$
$\frac{\partial L}{\partial (a^{[1]})} = (W^{[2]})^T \frac{\partial L}{\partial (z^{[2]})}$
$\frac{\partial L}{\partial (z^{[2]})} = \frac{\partial L}{\partial a^{[2]}} * \sigma'(z^{[2]})$

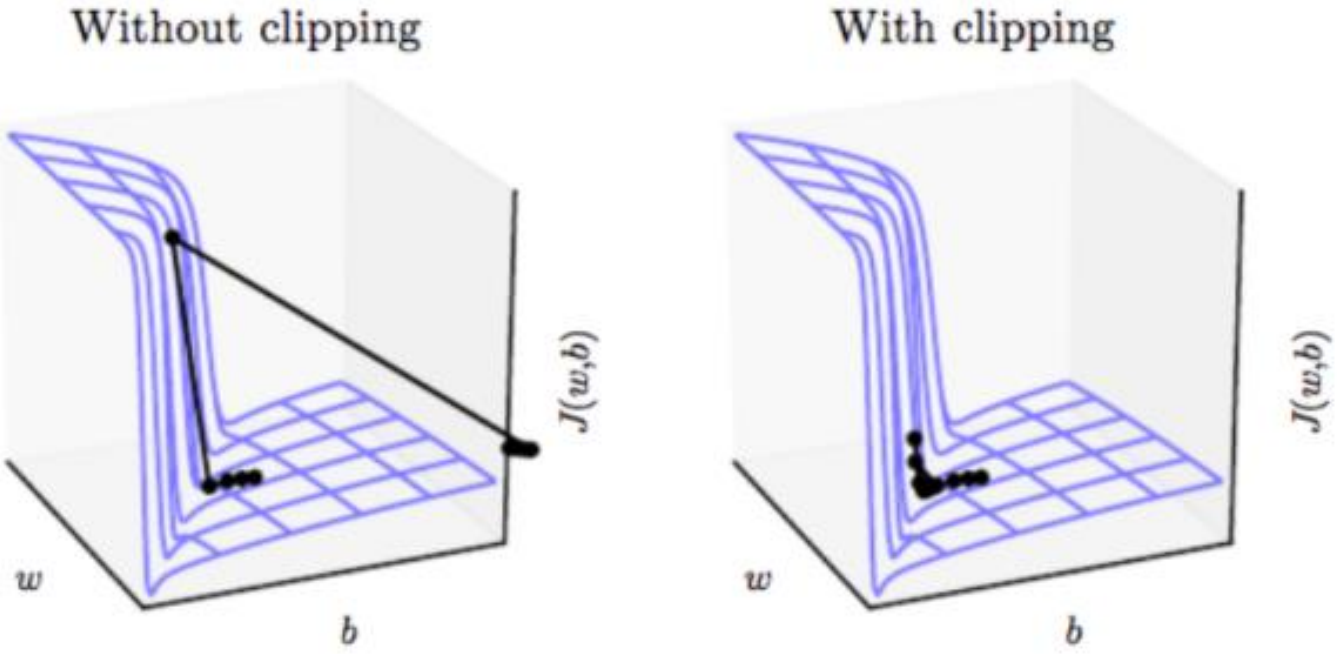
Recipe for underfitting problem of Deep Learning



Vanishing Gradient Problem



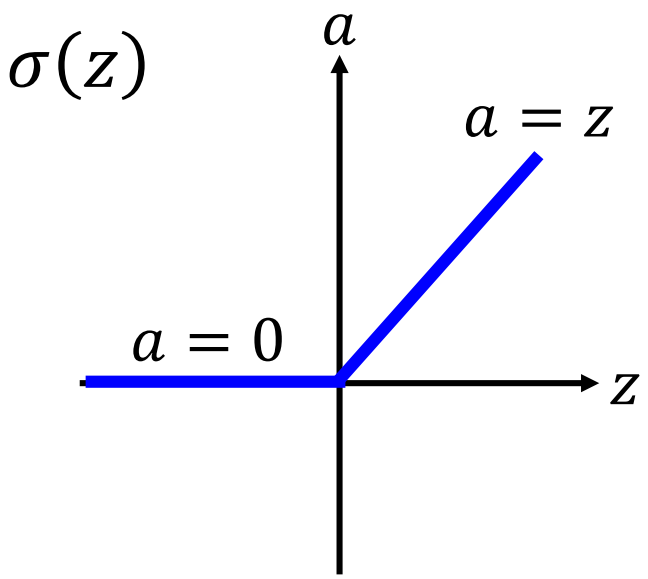
Exploding Gradient



— Goodfellow et al., *Deep Learning*

ReLU

- Rectified Linear Unit (ReLU)



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

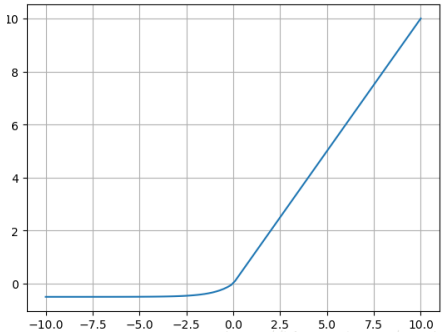
Reason:

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

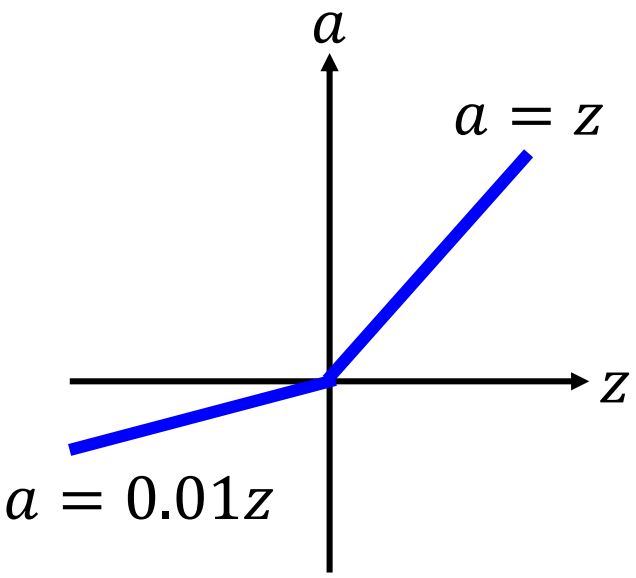
ReLU - variant

Exponential Linear Unit (ELU)

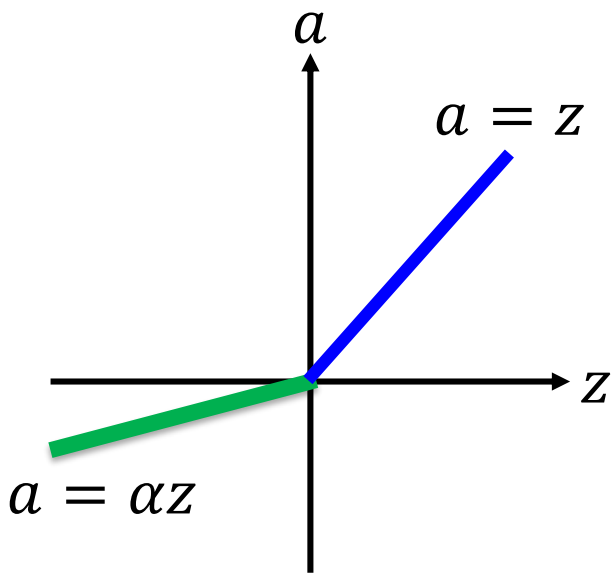
[FAST AND ACCURATE DEEP NETWORK LEARNING BY EXPONENTIAL LINEAR UNITS (ELUS)（2016，Djork-Arn´e Clevert, Thomas Unterthiner & Sepp Hochreiter）]



Leaky ReLU

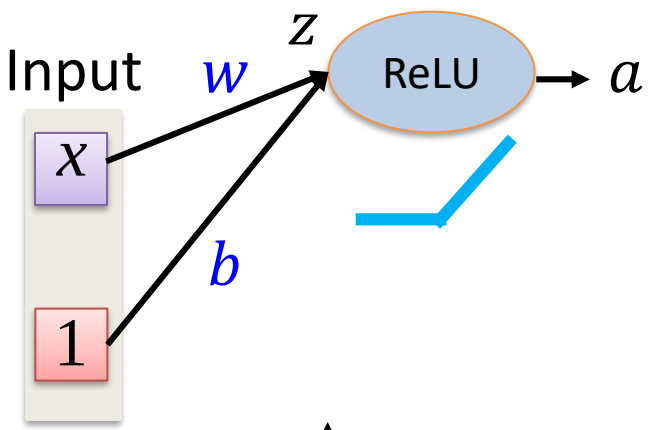


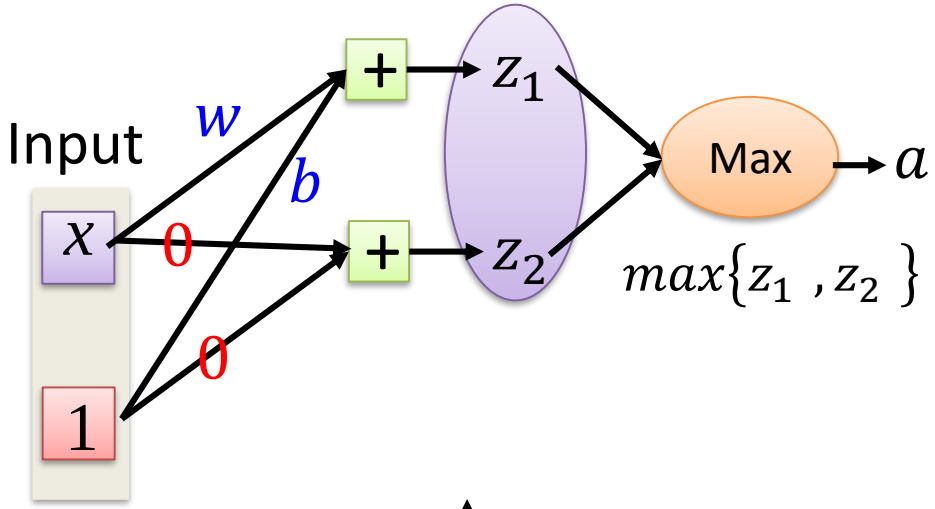
Parametric ReLU

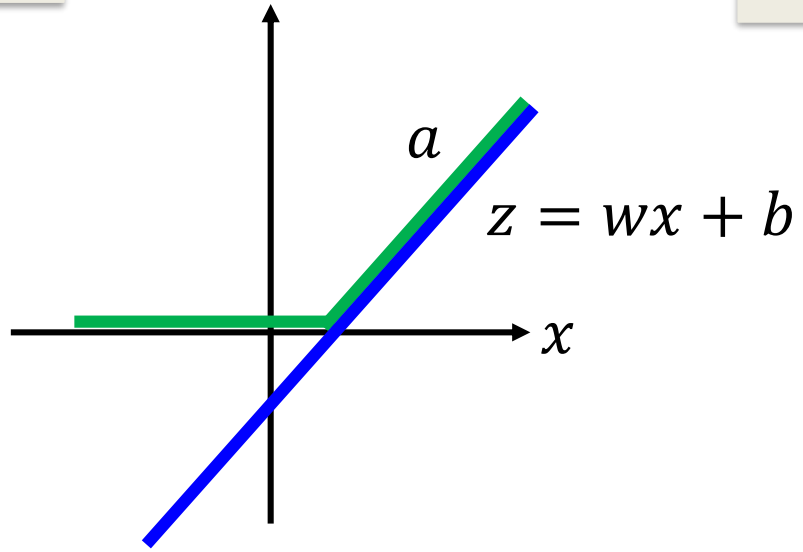


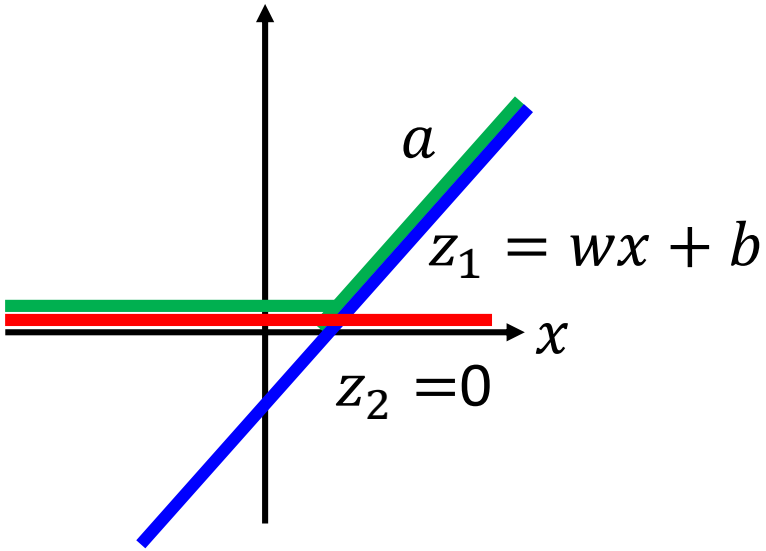
α also learned by gradient descent

ReLU is a special cases of Maxout



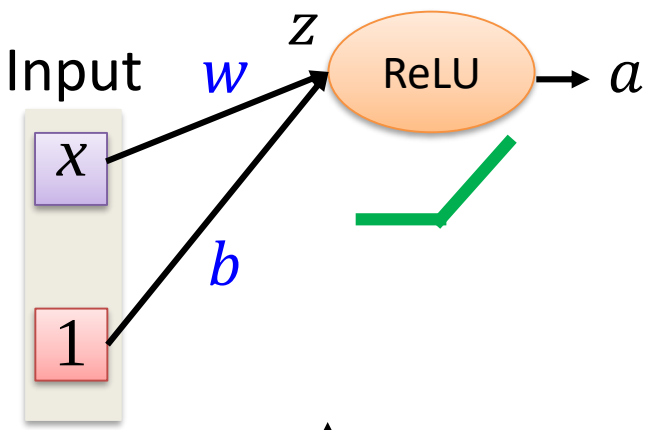


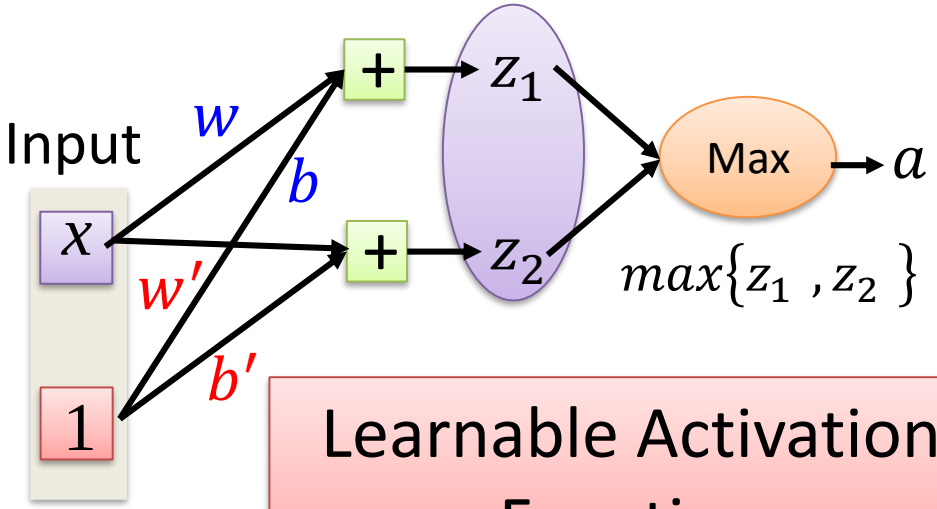




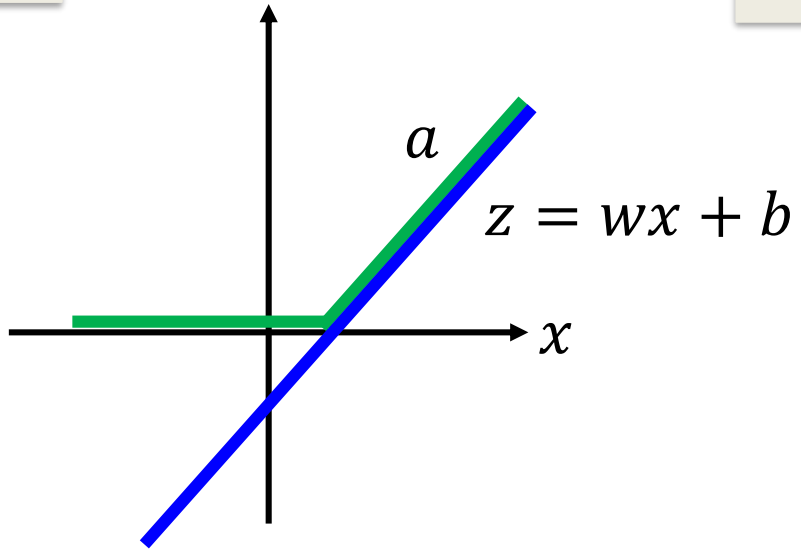
Maxout

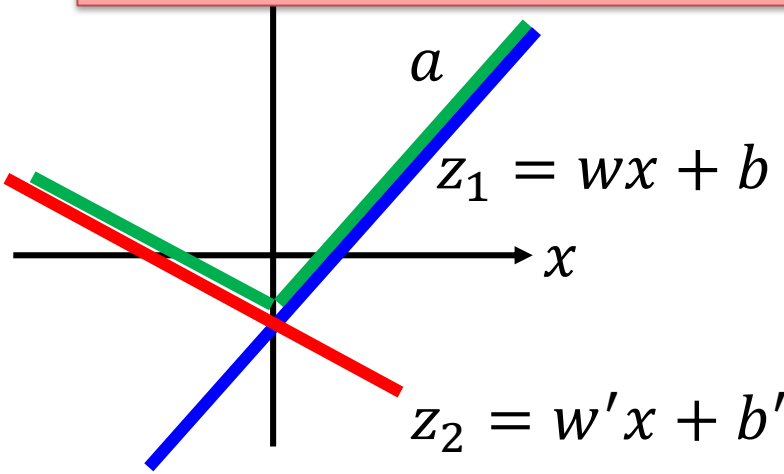
More than ReLU





Learnable Activation Function

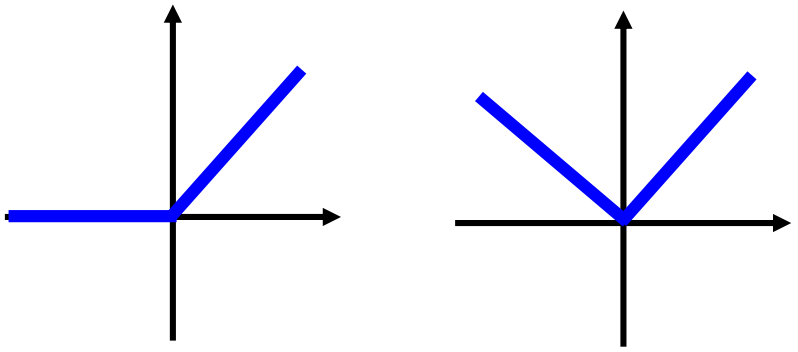




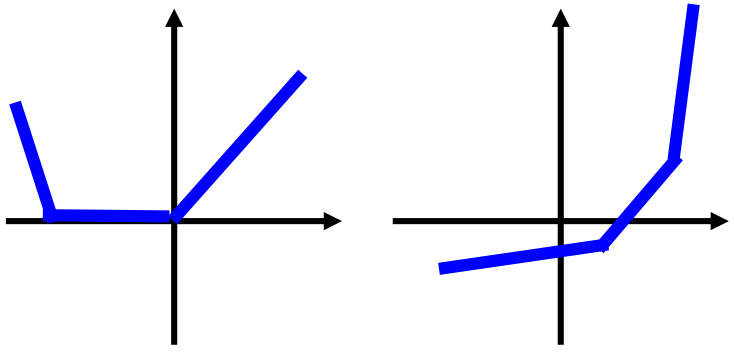
Maxout

- 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



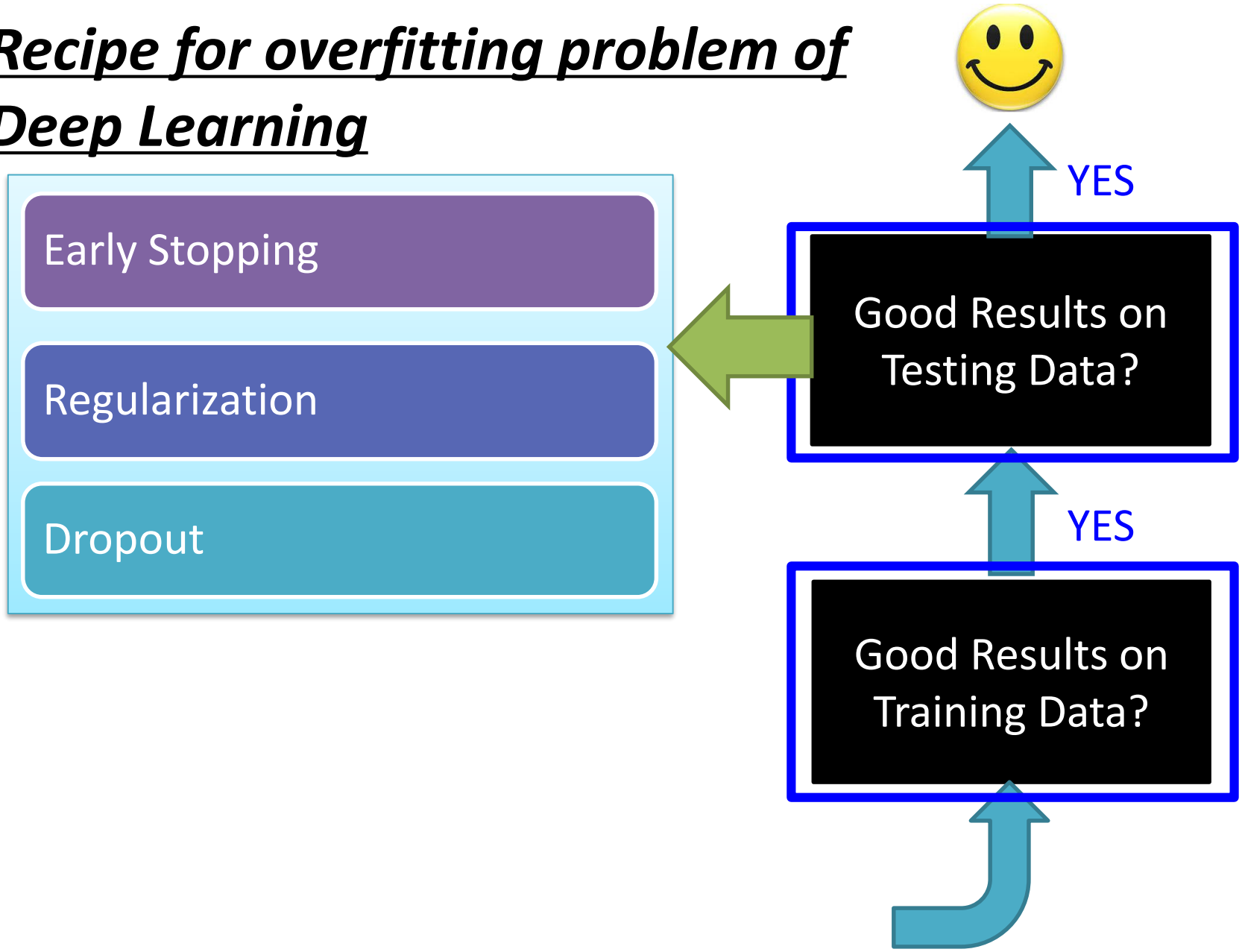
$g^{<t>} = \frac{\partial L(\theta^{<t>})}{\partial \theta}$

Gradient Descent Method	$\Delta\theta(= \theta^{<t+1>} - \theta^{<t>}$
SGD, BGD, MBGD	$-\eta g^{<t>}$
Adagrad	$-\frac{\eta}{\sqrt{\sum_{i=0}^{t}(g^{<i>})^2}} g^{<t>}$
RMSProp	$-\frac{\eta}{\sigma^{<t>}} g^{<t>}$ $(\sigma^{<0>} = g^{<0>},$ $\sigma^{<t>} = \sqrt{\alpha(\sigma^{<t-1>})^2 + (1-\alpha)(g^{<t>})^2})$
SGD with Momentum (SGDM)	$v^{<t+1>} = \lambda v^{<t>} - \eta g^{<t>}$ $v^{<0>}=0$
SGD with Nesterov (NAG)	$v^{<t+1>} = \lambda v^{<t>} - \eta \frac{\partial L^n(\theta^{<t>}+v^{<t>})}{\partial \theta}$
Adam(RMSProp + Momentum)	
Nadam (Nesterov+Adam)	

引入二阶动量：加速度

引入一阶动量：惯性速度

Recipe for overfitting problem of Deep Learning



Panacea(万能药) for Overfitting

- Have more training data
- ***Create*** more training data (?)

Handwriting recognition:

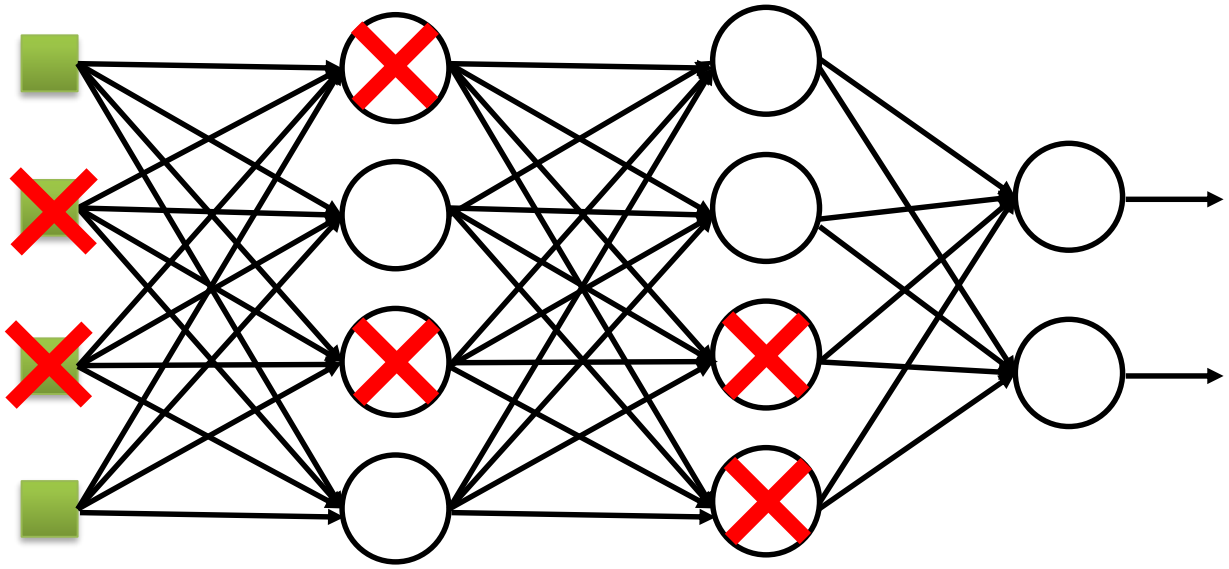
Original Training Data:

Created Training Data:

Shift 15°

Dropout(随机失活)

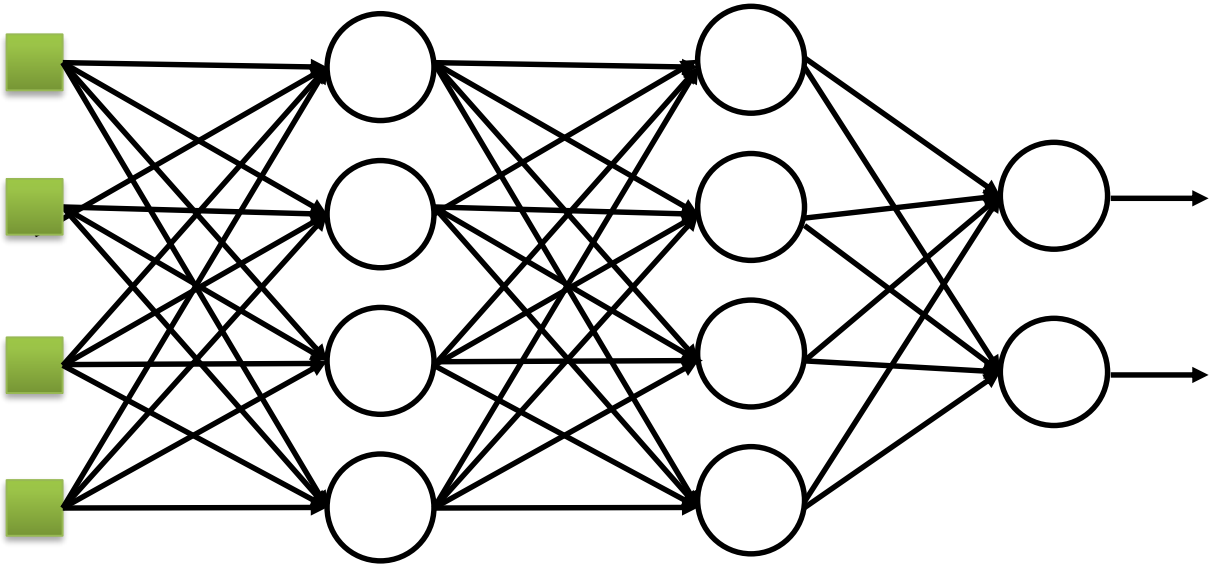
Training:



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

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.

神经网络的通用工作流程

1. 定义问题，收集数据集
2. 选择衡量成功的指标
   - 要在验证数据上监控哪些指标？
3. 确定评估方法
   - 留出验证？K折验证？你应该将哪一部分数据用于验证？
4. 准备数据
5. 开发比基准更好的模型
6. 扩大模型规模：开发过拟合的模型
7. 模型正则化与调节超参数：基于模型在验证数据上的性能 （目前被过度关注）

神经网络的通用工作流程

1. 定义问题，收集数据集
2. 选择衡量成功的指标
   - 要在验证数据上监控哪些指标？
3. 确定评估方法
   - 留出验证？K折验证？你应该将哪一部分数据用于验证？
4. 准备数据
5. 开发比基准更好的模型
6. 扩大模型规模：开发过拟合的模型
7. 模型正则化与调节超参数：基于模型在验证数据上的性能 （目前被过度关注）

1 定义问题：软件架构设计、确定评价指标

2 获取数据：自动化方式

3 研究数据：可视化方式，相关性研究等

4 准备数据：数据清理、特征选择及处理

5 研究模型：确定验证集上的评估方法、列出可能的模型并训练，选择最有希望的3~5个模型

6 微调模型：寻找最佳超参数，模型融合，评估泛化性能

7 展示解决方案：将工作进行文档化总结展示

8 启动、监视、维护系统：投入使用

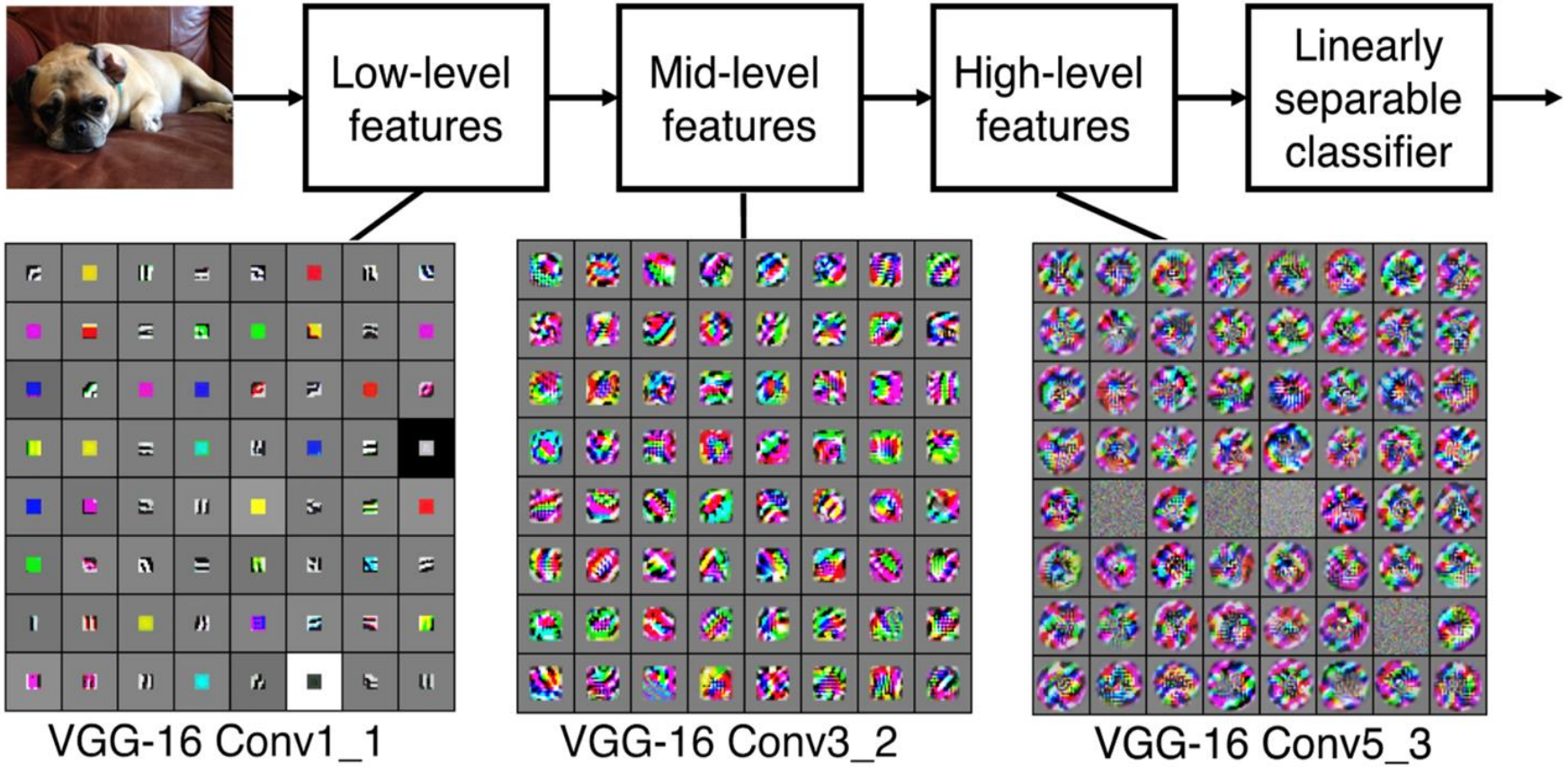
两种特定应用场景中的NN

- Convolutional Neural Network
  - 输入为图像
  - 关注：机器自动学习到图像特征
- Recurrent Neural Network (RNN)
  - 输入或输出为序列数据

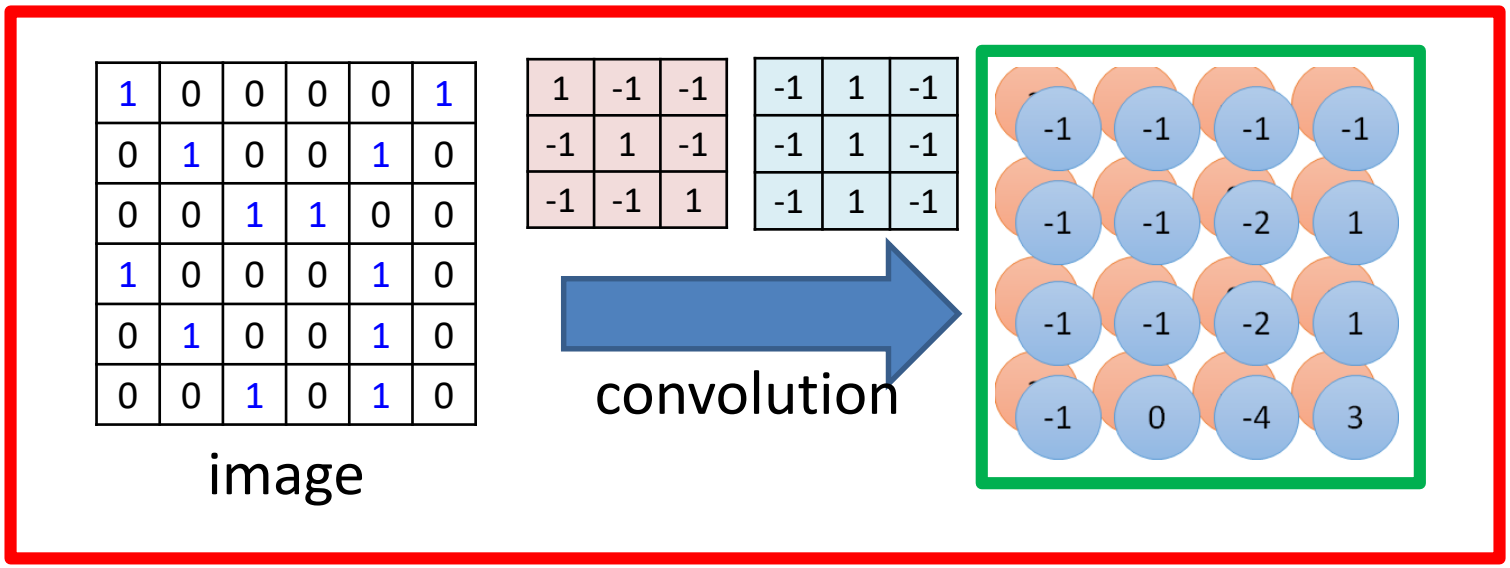
Preview

[Zeiler and Fergus 2013]

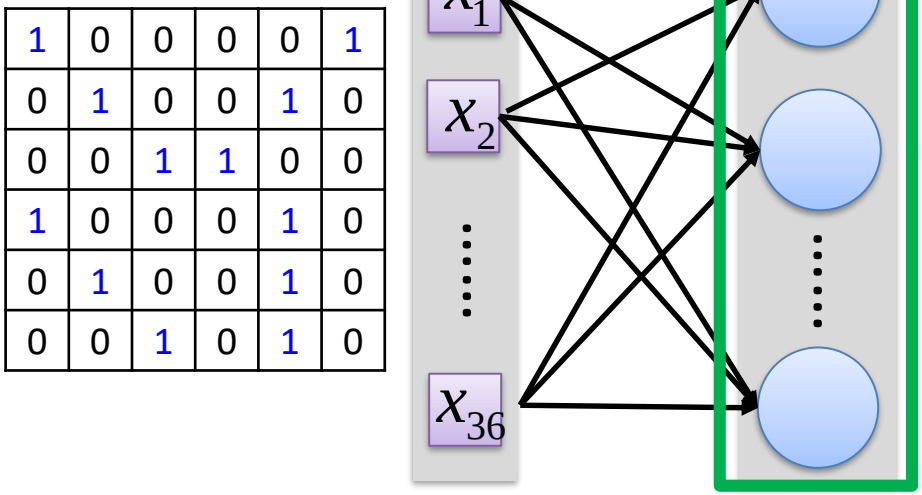
Visualization of VGG-16 by Lane McIntosh. VGG-16 architecture from [Simonyan and Zisserman 2014].

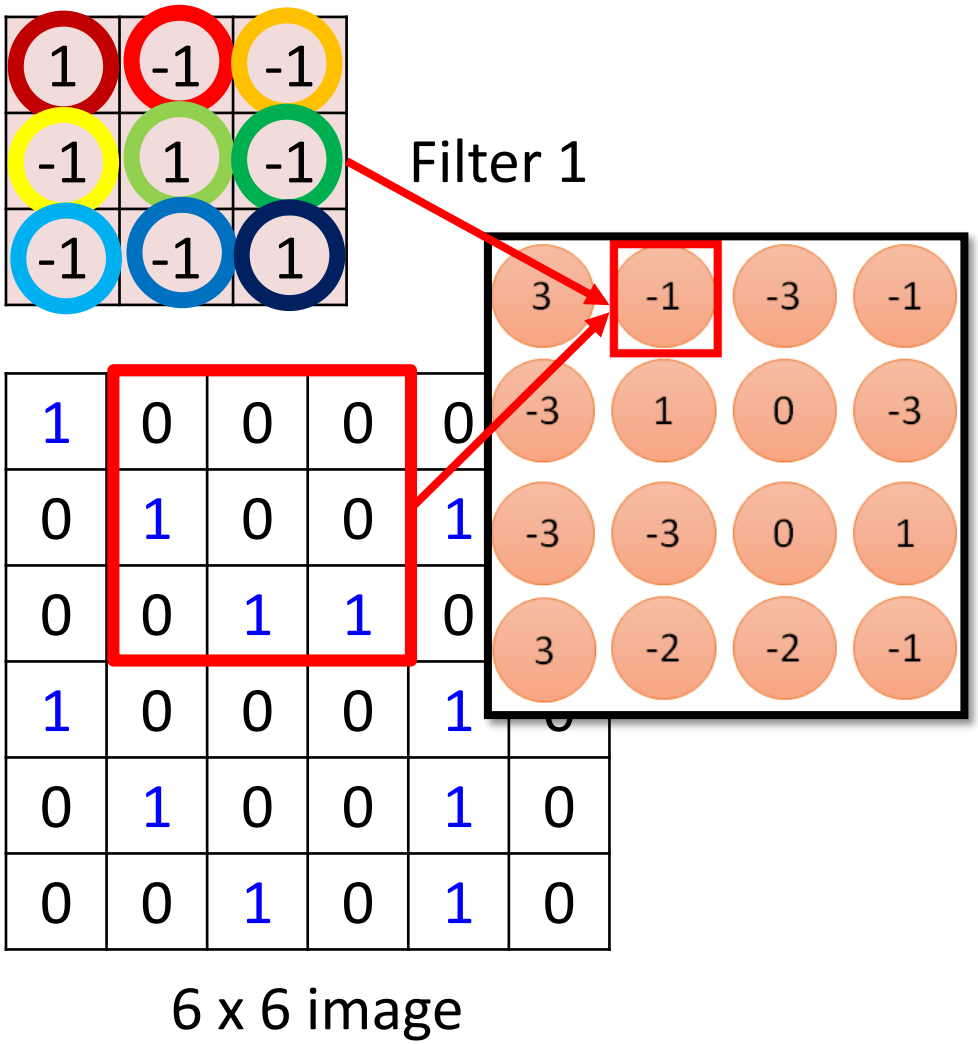


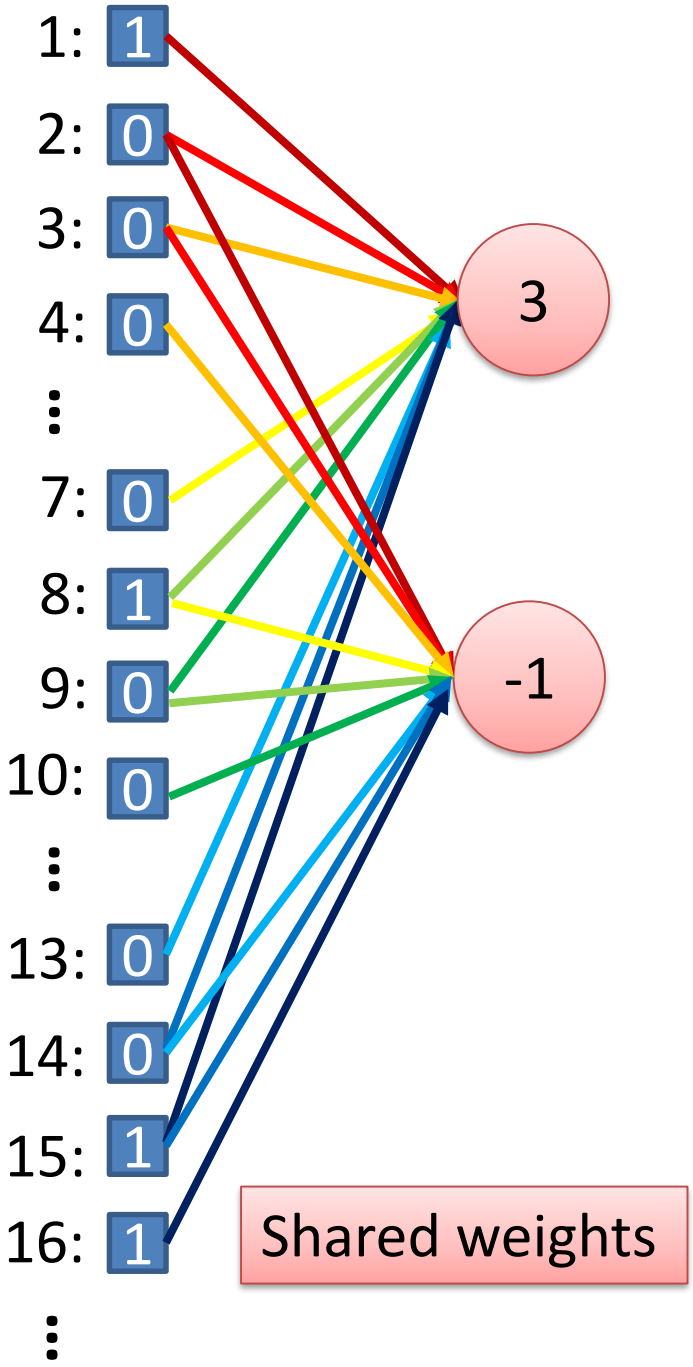
Convolution v.s. Fully Connected



Fully-connected







Sparsity of connections: Less parameters!

Parameter sharing: Even less parameters!

The whole CNN

Property 1

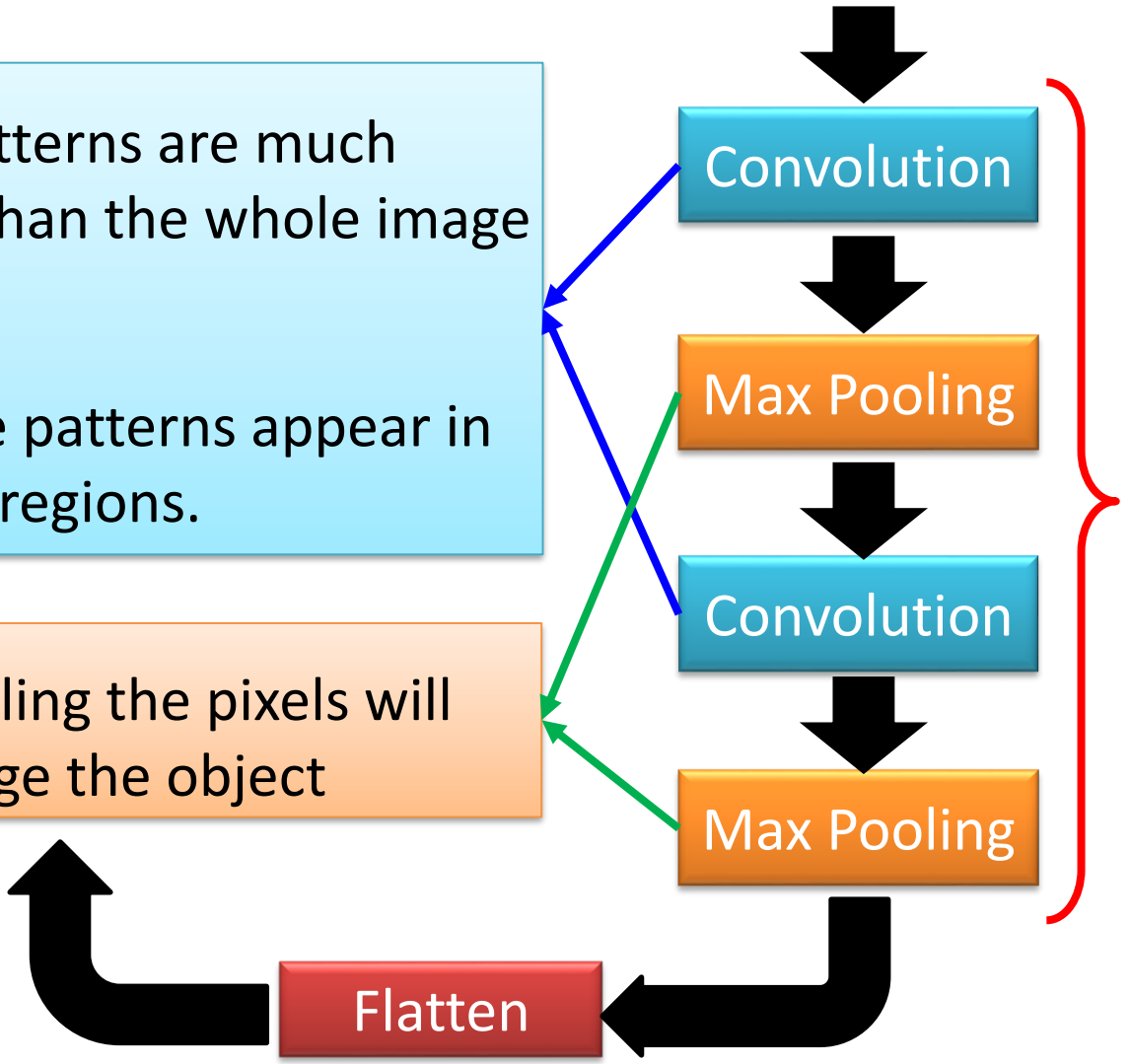
- Some patterns are much smaller than the whole image

Property 2

- The same patterns appear in different regions.

Property 3

- Subsampling the pixels will not change the object



The whole CNN

Convolution

Max Pooling

Convolution

Max Pooling

Can repeat many times

Flatten

Fully Connected Feedforward network

cat dog ……

More details in Convolution Layer

- Kernel/Filter
- Stride
- Padding

- Hyperparamters for the convolution layer L:
  - f[L]: Filter Size
  - p[L]: Padding
  - s[L]: Stride
  - N_f : number of filters

Each filter is: f[L] * f[L] * N_c [L-1]

Input: $N_H[L-1] * N_W[L-1] * N_c[L-1]$

Output: $N_H[L] * N_W[L] * N_c[L]$

$$N_H[L] = \left\lfloor \frac{N_H[L-1] + 2*p[L] - f[L]}{s[L]} + 1 \right\rfloor$$

$$N_W[L] = \left\lfloor \frac{N_W[L-1] + 2*p[L] - f[L]}{s[L]} + 1 \right\rfloor$$

$$N_c[L] = N_f$$

- 利用数据增强（data augmentation）来解决过拟合！

代码清单 5-11　利用 ImageDataGenerator 来设置数据增强

```
datagen = ImageDataGenerator(
      rotation_range=40,
      width_shift_range=0.2,
      height_shift_range=0.2,
      shear_range=0.2,
      zoom_range=0.2,
      horizontal_flip=True,
      fill_mode='nearest')
```

注意，不能增强验证集！

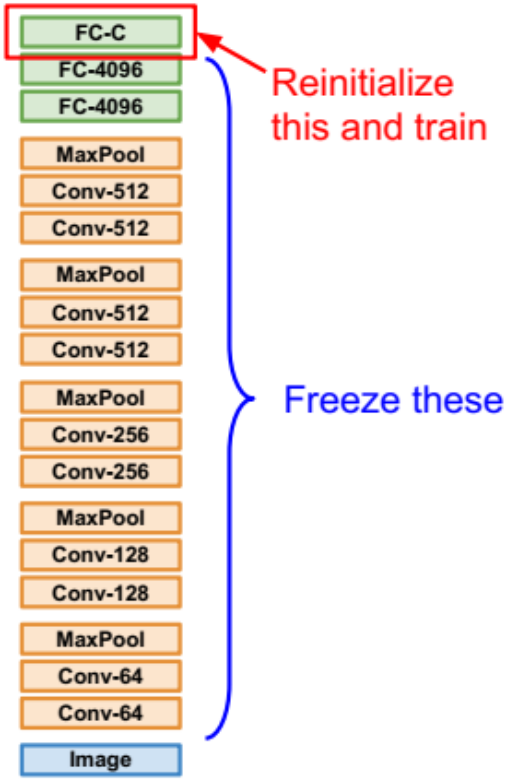
Rotation_range是角度值（在0~180范围内），表示图像随机旋转的角度范围。
width_shift 和 **height_shift**是图像在水平或垂直方向上平移的范围（相对于总宽度或总高度的比例）。
shear_range是随机错切变换的角度。
zoom_range是图像随机缩放的范围。
horizontal_flip是随机将一半图像水平翻转。如果没有水平不对称的假设（比如真实世界的图像），这种做法是有意义的。
fill_mode是用于填充新创建像素的方法，这些新像素可能来自于旋转或宽度/高度平移。

Transfer Learning with CNNs

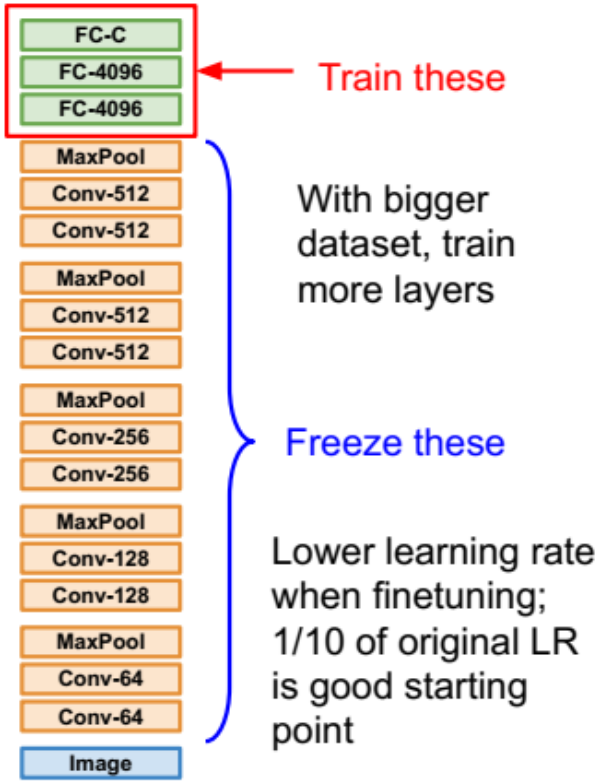
Donahue et al, "DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition", ICML 2014
Razavian et al, "CNN Features Off-the-Shelf: An Astounding Baseline for Recognition", CVPR Workshops 2014

1. Train on Imagenet

2. Small Dataset (C classes)



3. Bigger dataset



Classification based on CNN

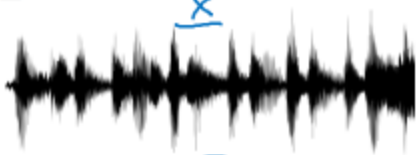
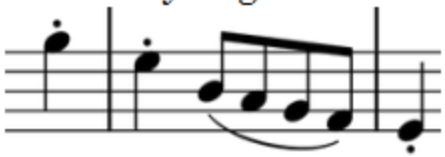
- LeNet-5 [LeCun et al., 1998]
- AlexNet [Krizhevsky et al., 2012]
  - 使用CNN解决图像分类问题，在ILSVRC'12中大大获胜并大大提高了State-of-art的准确率，促进了CNN在图像分类领域的快速发展
- VGG
- Inception系列

从以下三个方面把握：
1）网络整体结构是怎样的？
2）创新点是什么？（包含网络结构的创新和比较新颖的激活函数等方法）
2）创新点可以带来什么好效果？为什么？

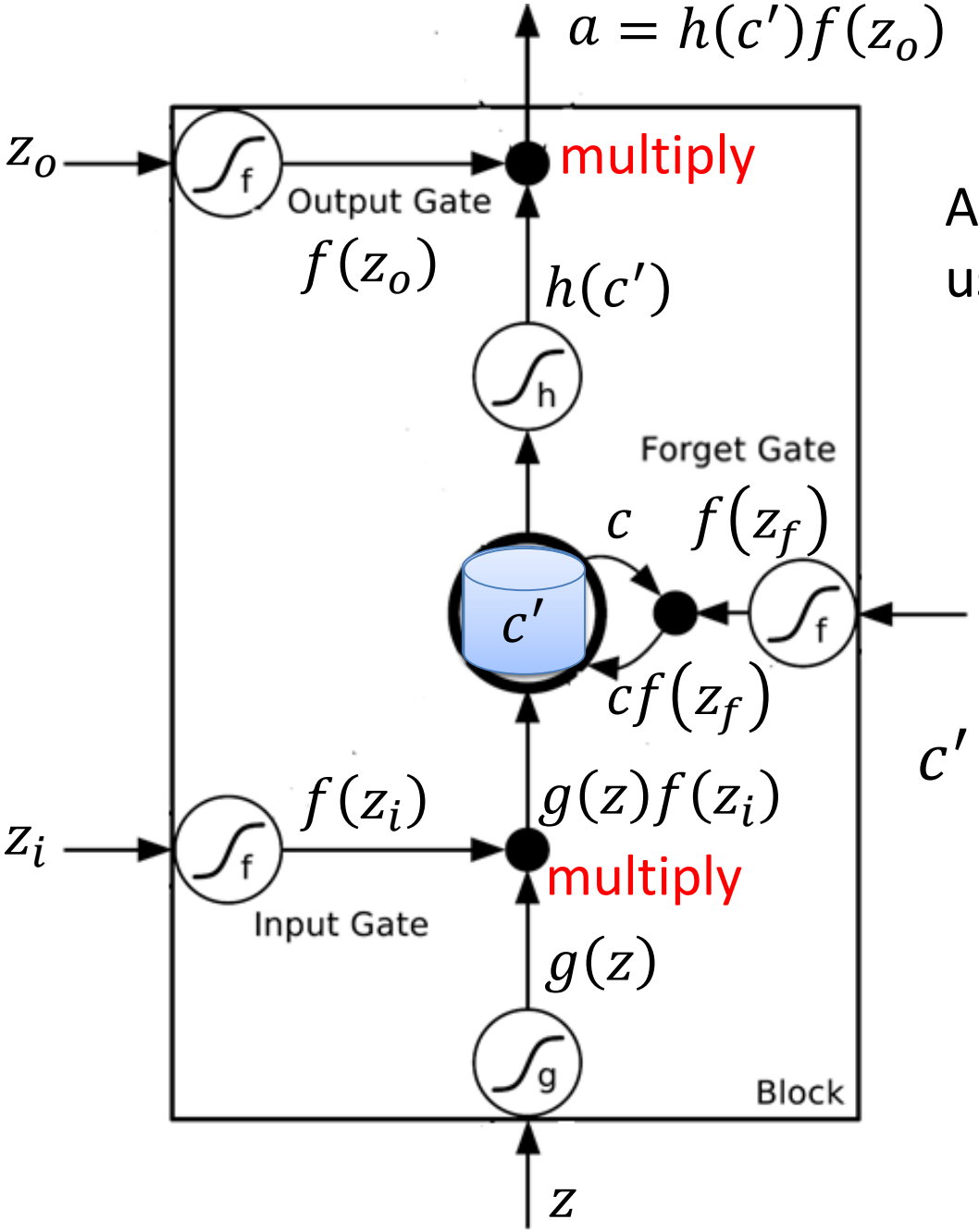
Recurrent Neural Network (RNN)

- 什么是序列数据？会举例说明。
- 对于给定问题，能判断出是否该使用RNN模型
- 即使只有一层的RNN模型，仍可能出现梯度消失和梯度爆炸，为什么？
- LSTM中一个神经元的基本结构是？
- LSTM，与一般的RNN相比，优势在哪？

Examples of sequence data

Speech recognition		"The quick brown fox jumped over the lazy dog."
Music generation		
Sentiment classification	"There is nothing to like in this movie."	
DNA sequence analysis	AGCCCCTGTGAGGAACTAG	AGCCCCTGTGAGGAACTAG
Machine translation	Voulez-vous chanter avec moi?	Do you want to sing with me?
Video activity recognition		Running
Name entity recognition	Yesterday, Harry Potter met Hermione Granger.	Yesterday, Harry Potter met Hermione Granger.

Andrew Ng



Activation function f is usually a sigmoid function

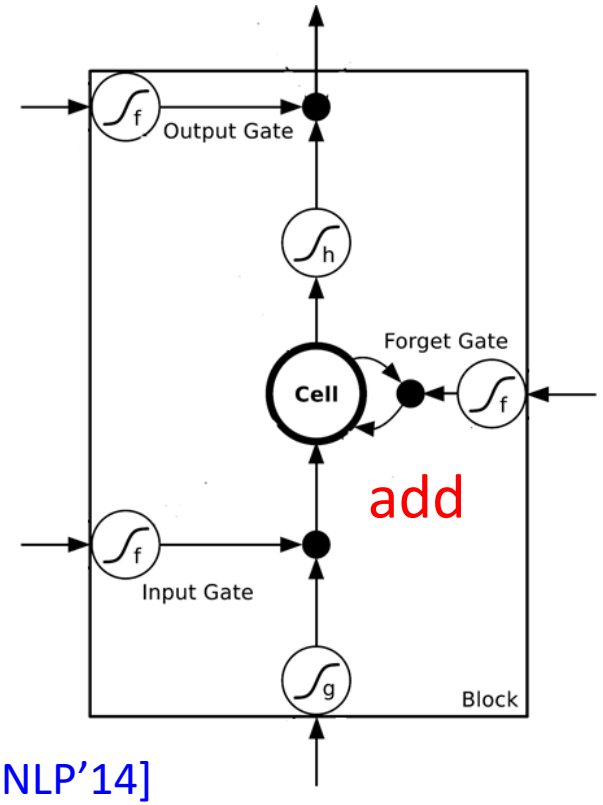
Between 0 and 1

Mimic(模仿) open and close gate

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

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]

预祝各位期末取得满意的成绩，谢谢！