

深度学习 从入坑到放弃

East196

GitHub : <http://github.com/east196>

深度学习简介

是什么？

AI的最前沿分支
深度学习 = 深度神经网络

能做什么？

- 自动驾驶 ~~~ CNN
- 聊天机器人，机器翻译 ~~~ RNN
- 生成文本，图片，语音，视频 ~~~ GAN
- AlphaGo，自动打游戏 ~~~ DQN

机器学习学什么？

- 监督学习
手把手教学
- 无监督学习
丢你本书看，然而并不想理你
- 强化学习
丢你本书看，请你做题，板子伺候

深度学习的强项

监督学习

- 分类
- 回归

分类，就是选择。

为什么用深度学习而不是其他？

简单粗暴效果好！

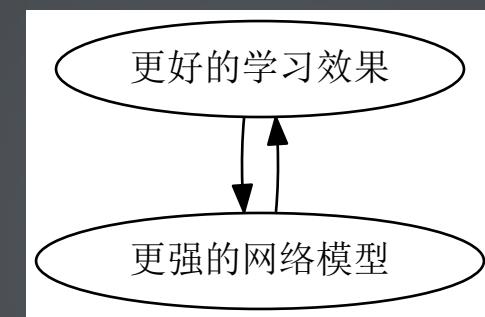
能WORK

- 越来越大的数据
- 越来越快的电脑
- 越来越多的优化技巧

NN打油诗
- East196

机器性能大提升，
海量数据在产生。
群策群力来贡献，
神经网络大又深。

效果好



先驱们的不断开拓优化
CNN , RNN , GAN , DQN , CapsuleNet

看起来公式好难懂~~

只需要**理解**两个概念

- 高数 导数 函数变化的趋势
- 线代 矩阵乘法 维度的对应

初探神经网络

从 $Y=WX+B$ 谈起

$$y = f(x)$$

最简单的函数

二元一次方程 - 普通的线性关系

$$y = wx + b$$

给两组数据：

10, 2
3, 4

构成方程：

$$\begin{cases} 2 = 10w + b \\ 4 = 3w + b \end{cases}$$

怎么解？：)

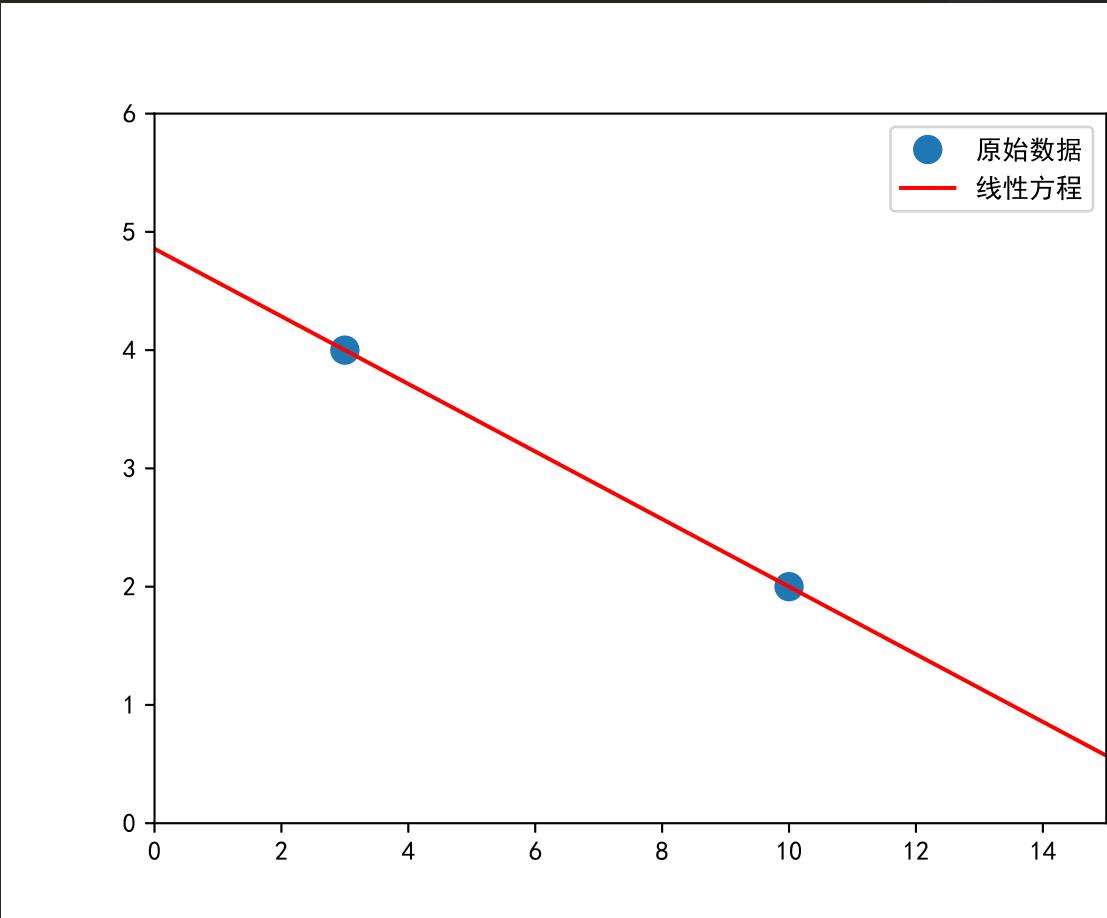
```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.font_manager import *
#指定默认字体
matplotlib.rcParams['font.family']='simhei'
#解决负号'-'显示为方块的问题
matplotlib.rcParams['axes.unicode_minus']=False

#解方程 y = wx + b
x = np.array([10, 3])
y = np.array([2, 4])

A = np.vstack([x, np.ones(len(x))]).T
w, b = np.linalg.lstsq(A, y)[0]
#print(w, b)

# 再来画个图
plt.axis([0, 15, 0 ,6])
plt.plot(x, y, 'o', label=u'原始数据', markersize=10)

t = np.linspace(-10, 20, 10)
plt.plot(t, w*t + b, 'r', label=u'线性方程')
plt.legend()
plt.show()
```



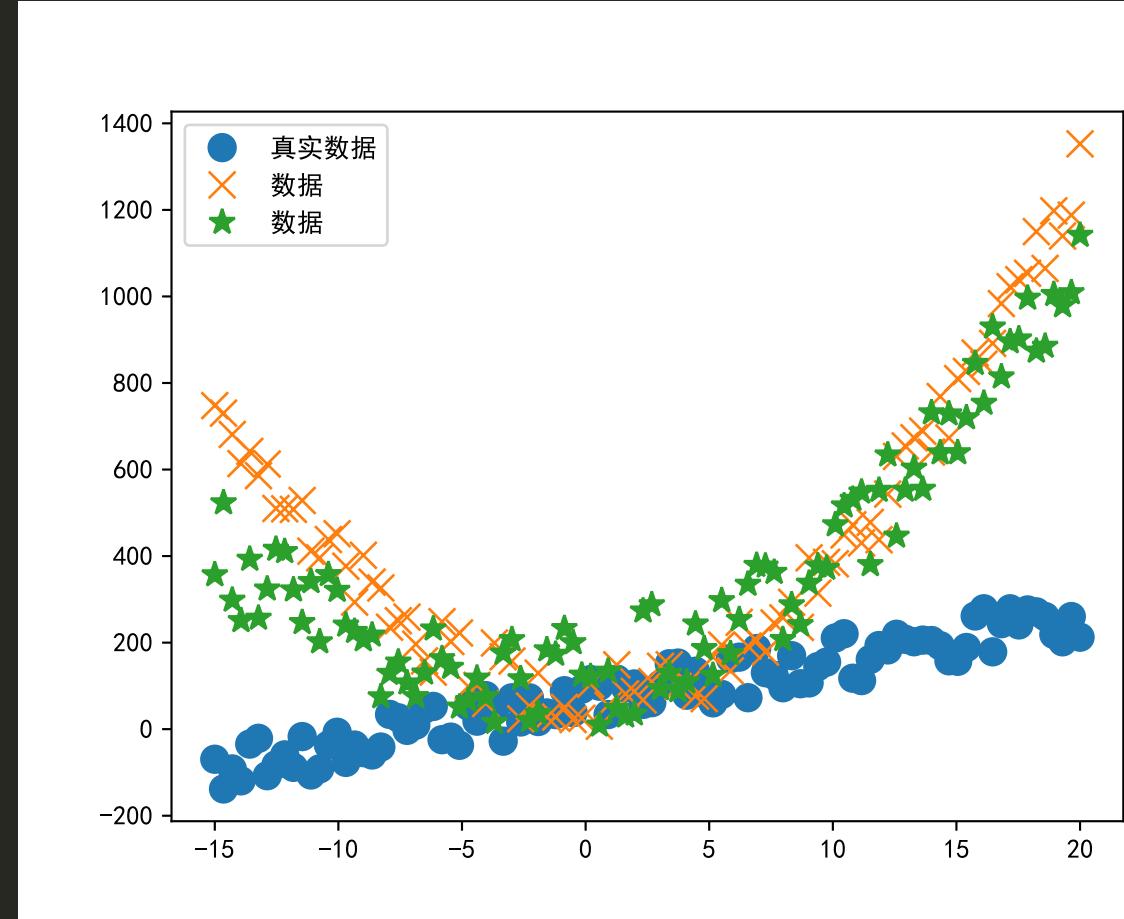
然而，现实是：

```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.font_manager import *
#指定默认字体
matplotlib.rcParams['font.family']='simhei'
#解决负号'-'显示为方块的问题
matplotlib.rcParams['axes.unicode_minus']=False

# 模拟真实数据
x = np.linspace(-15, 20, 100)
y = 10*x +np.random.rand(100)*120
z = 3*x*x +np.random.rand(100)*160
m = 2*x*x +10*x +np.random.rand(100)*250

# 再来画个图
plt.plot(x, y, 'o', label=u'真实数据', markersize=10)
plt.plot(x, z, 'x', label=u'数据', markersize=10)
plt.plot(x, m, '*', label=u'数据', markersize=10)

plt.legend()
plt.show()
```



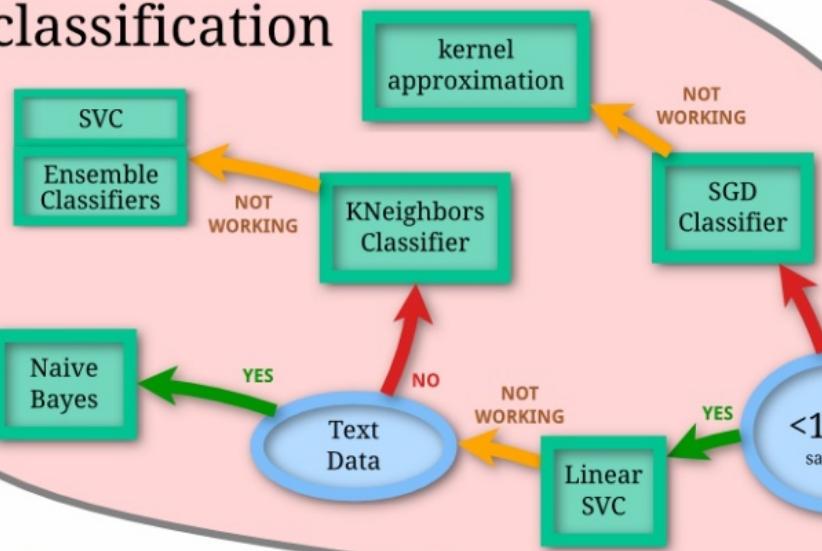
机器学习

use scikit-learn

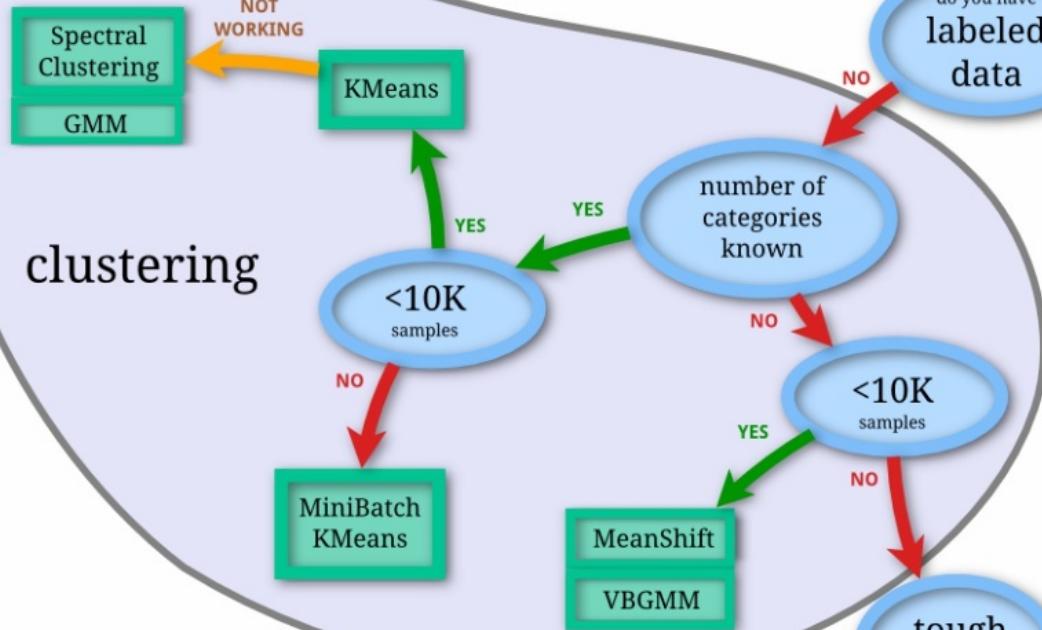
- 监督学习：分类，回归
- 无监督学习：聚类

scikit-learn algorithm cheat-sheet

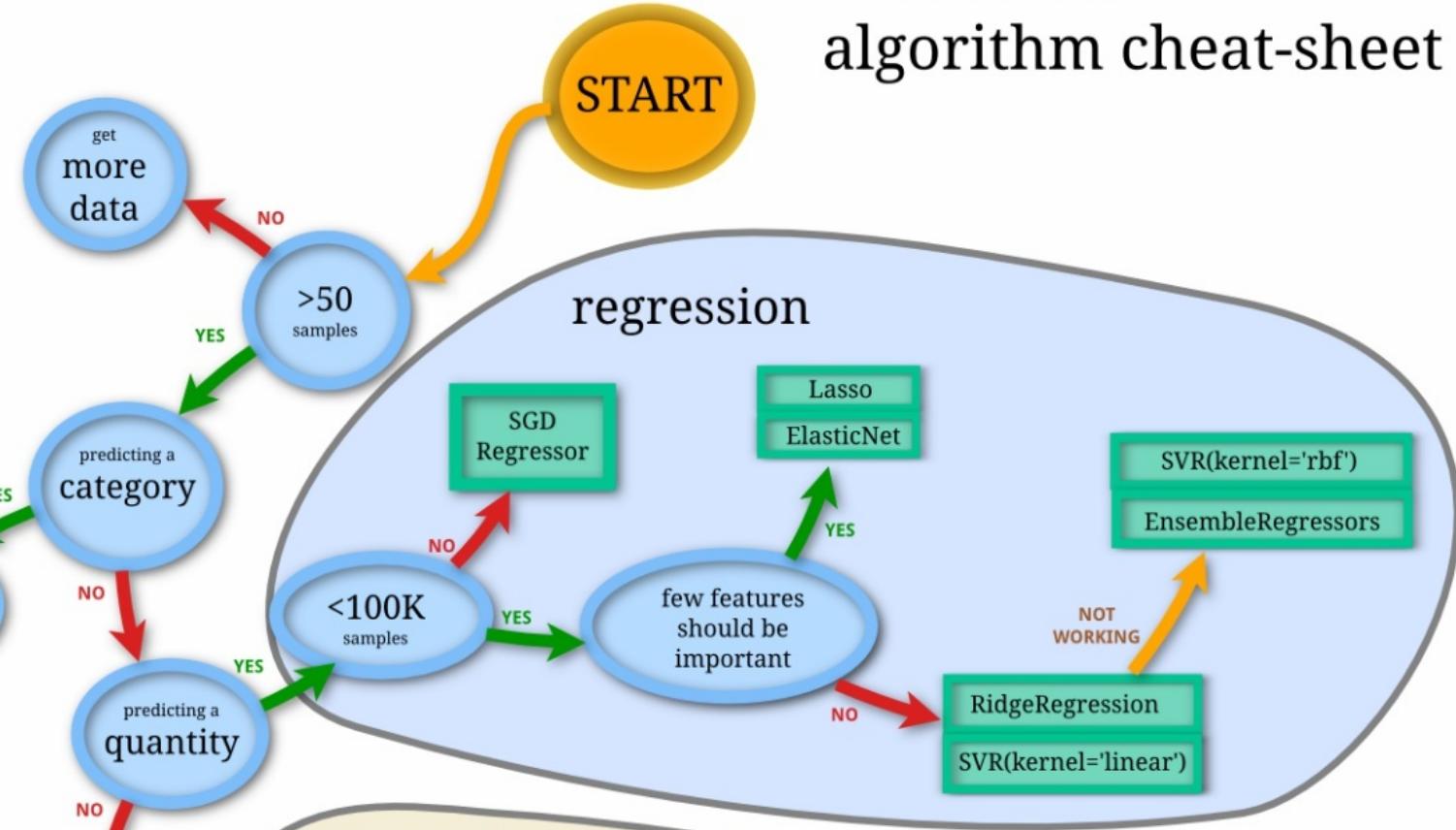
classification



clustering



regression

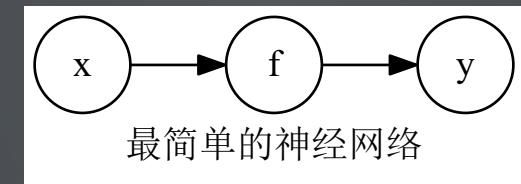


dimensionality reduction

- 基本回归：线性、决策树、SVM、KNN
- 集成方法：随机森林、Adaboost、GradientBoosting、Bagging、ExtraTrees

最简单的神经网络

Neural Network



咦，效果一般啊？
一定是打开的方式不对！

权重WEIGHT与偏置BIASE

$$y = wx + b$$

面熟对不对？

求解线性问题

权重和偏置怎么设置？

我也不知道，那就随机吧...

激活函数

面对现实
非线性世界

激活函数 Sigmoid&Tanh

```
import math
import matplotlib.pyplot as plt
import numpy as np
import matplotlib as mpl
mpl.rcParams['axes.unicode_minus']=False

def sigmoid(x):
    return 1.0 / (1.0 + np.exp(-x))

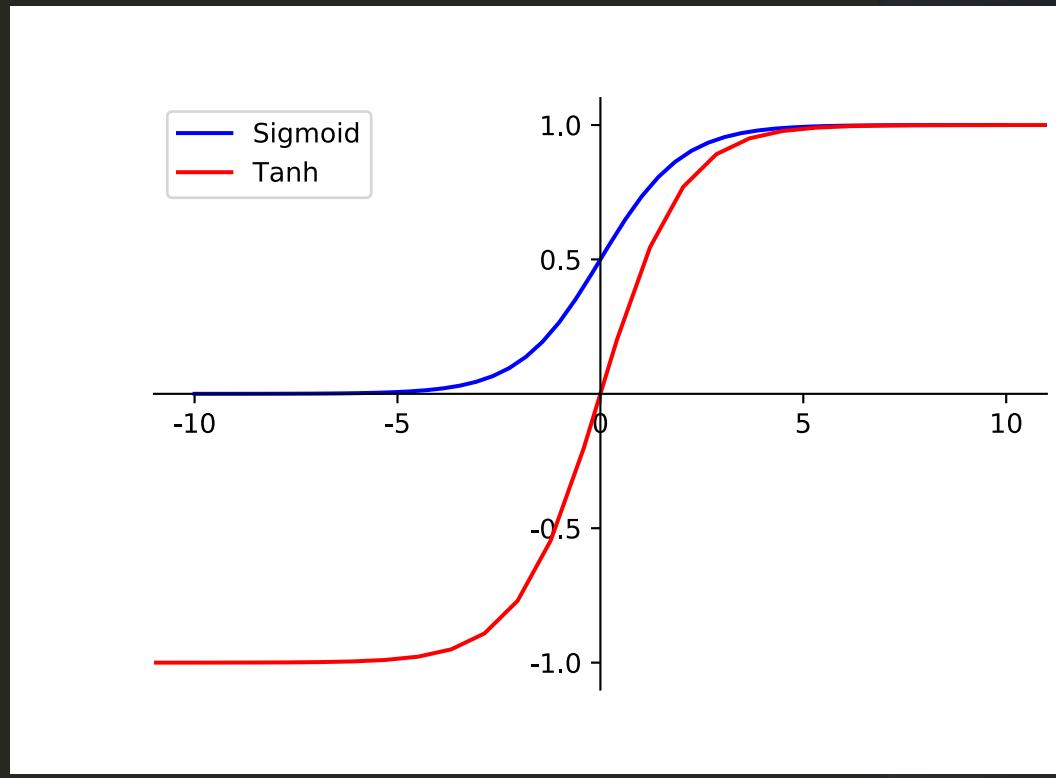
fig = plt.figure(figsize=(6, 4))
ax = fig.add_subplot(111)

x = np.linspace(-10, 10)
y = sigmoid(x)
tanh = 2*sigmoid(2*x) - 1

plt.xlim(-11, 11)
plt.ylim(-1.1, 1.1)

ax.spines['top'].set_color('none')
ax.spines['right'].set_color('none')

ax.xaxis.set_ticks_position('bottom')
ax.spines['bottom'].set_position(('data', 0))
ax.set_xticks([-10, -5, 0, 5, 10])
ax.yaxis.set_ticks_position('left')
```



```
ax.spines['left'].set_position(( 'data' , 0))
ax.set_yticks([-1, -0.5, 0.5, 1])

plt.plot(x, y, label="Sigmoid", color = "blue")
plt.plot(2*x, tanh, label="Tanh", color = "red")
plt.legend()
plt.show()
```

激活函数 ReLU

```
import math
import matplotlib.pyplot as plt
import numpy as np
import matplotlib as mpl
mpl.rcParams['axes.unicode_minus']=False

fig = plt.figure(figsize=(6, 4))
ax = fig.add_subplot(111)

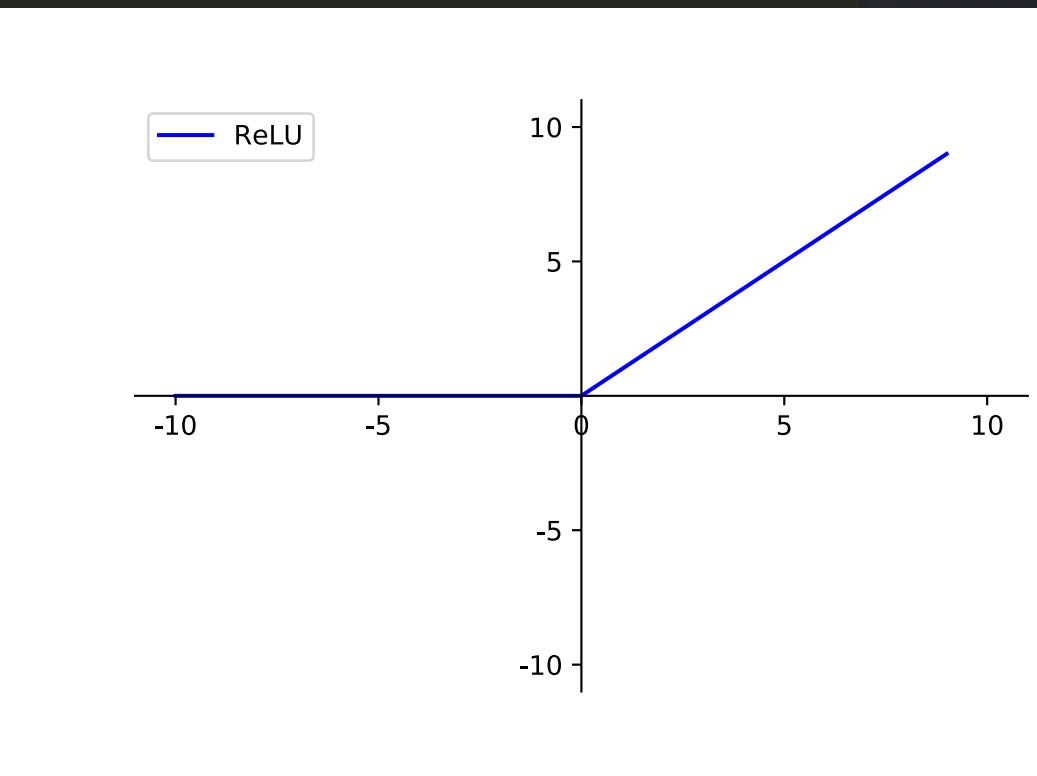
x = np.arange(-10, 10)
y = np.where(x<0, 0, x)

plt.xlim(-11, 11)
plt.ylim(-11, 11)

ax.spines['top'].set_color('none')
ax.spines['right'].set_color('none')

ax.xaxis.set_ticks_position('bottom')
ax.spines['bottom'].set_position(('data', 0))
ax.set_xticks([-10, -5, 0, 5, 10])
ax.yaxis.set_ticks_position('left')
ax.spines['left'].set_position(('data', 0))
ax.set_yticks([-10, -5, 5, 10])

plt.plot(x, y, label="ReLU", color = "blue")
```



```
plt.legend()  
plt.show()
```

BP神经网络

Back-propagation Neural Network

- 相对于Forward Neural Network而言
- 根据导数回头修正参数

然而，*Tensorflow* 默默安排好了一切

可是，我要识别一张图片，我该输入什么？

输入一般称之为特征

特征要靠自己分析，然后选取
比如我要识别车牌上的一个字

画个九宫格，9个特征

上中下，3个特征

整图，1个特征

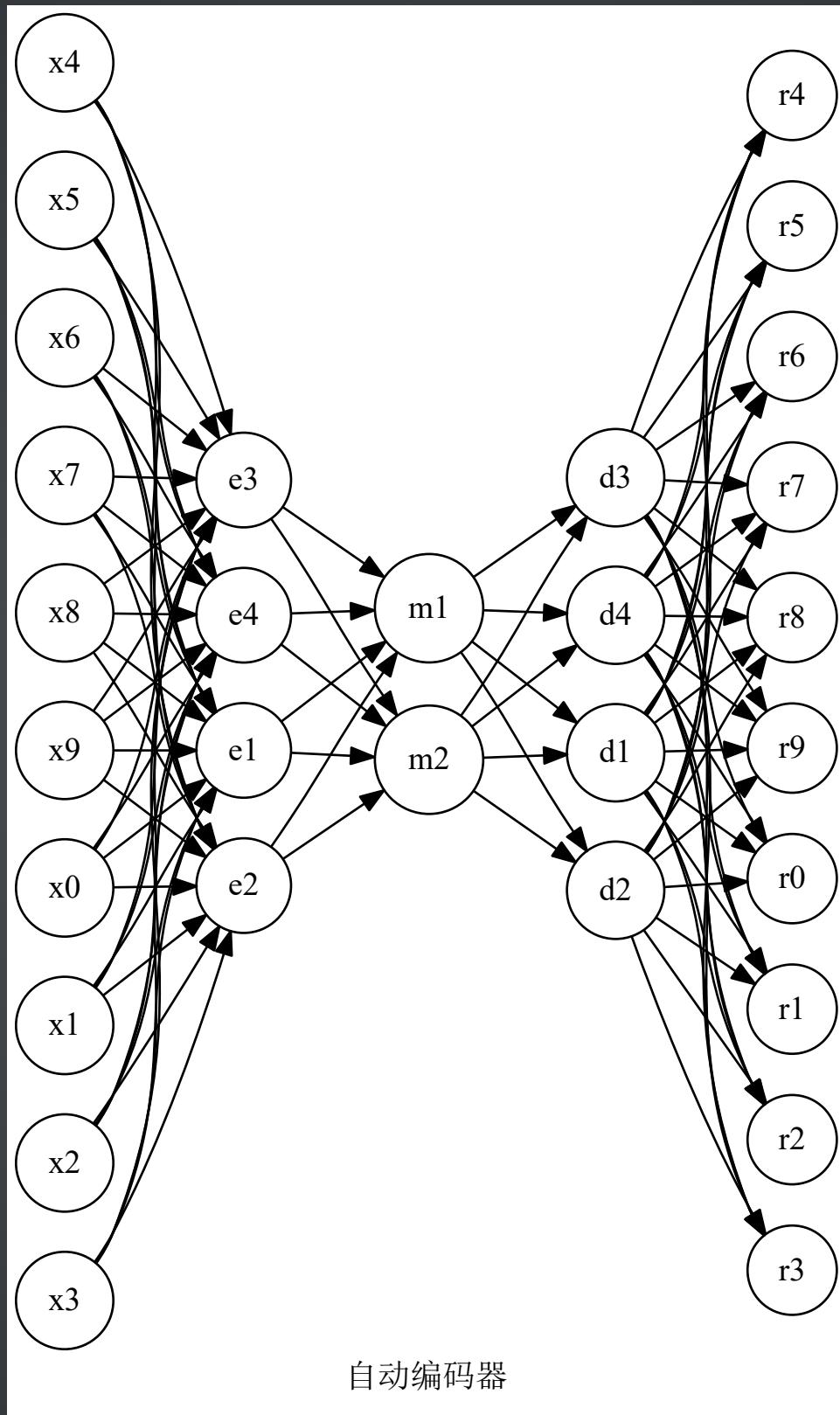
$$9+3+1=13$$

NO !
现在计算机跑这么快了 ,
我要把 宽*高*RGBA 直接扔进去 ! ! !

DNN

Deep Neural Network
更大更深的神经网络

use tensorflow pytorch



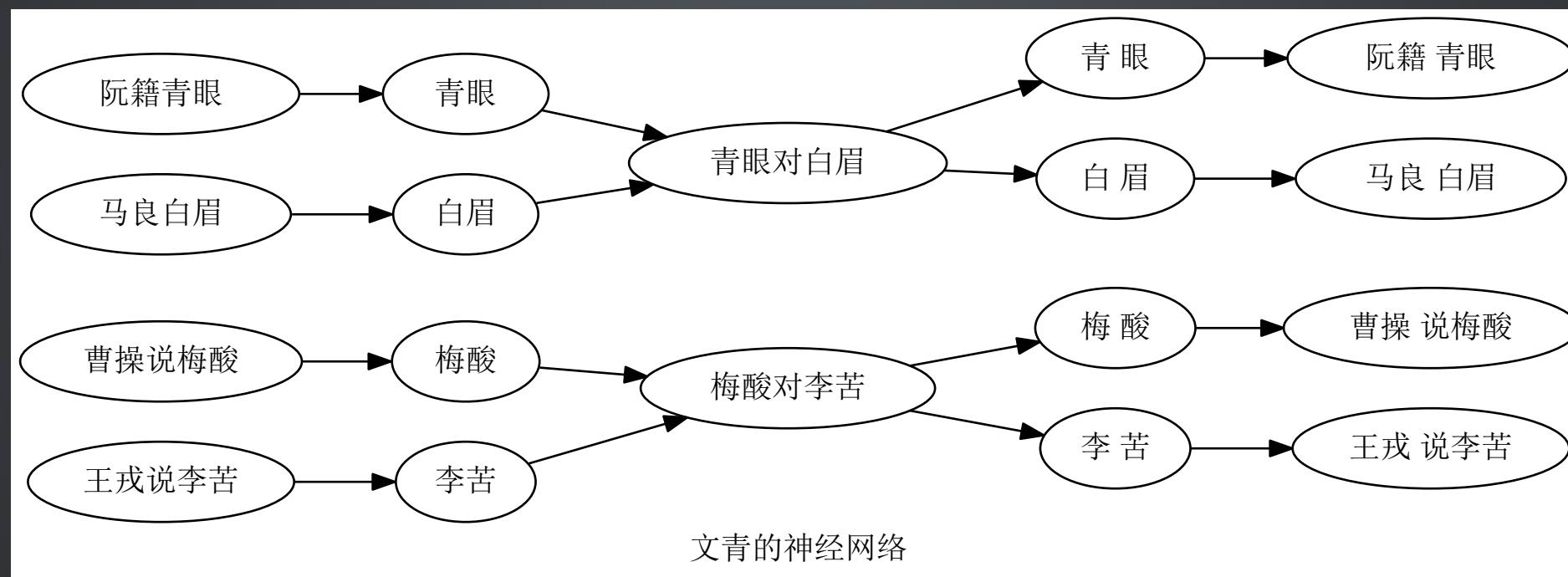
作用

保持输入和输出一致！！！

文青的解釋

声律启蒙

梅酸对李苦，青眼对白眉

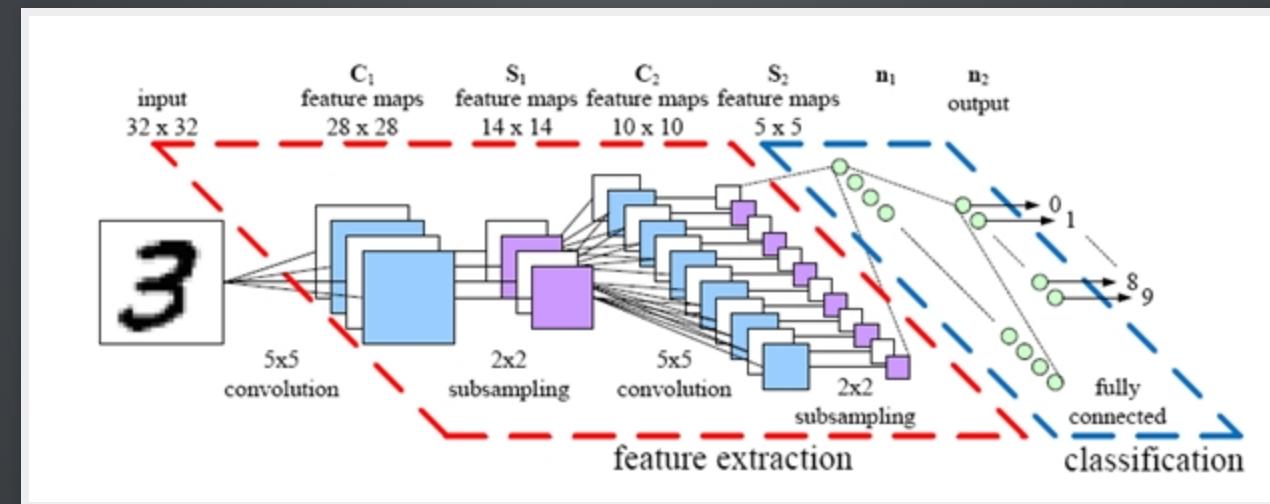


梅酸对李苦，青眼对白眉是能够复原的高度精简过的信息
同样， m_1 、 m_2 代表了全部的输入信息！！！
也就是说自动缩减了特征的维度～
带来了玩法的改变！！！

人类进入了 *End-to-End* 时代

CNN

Convolutional Neural Network 卷积神经网络



卷积：手电筒一块一块过



每次看到手电筒照到的那 **一块地方**

池化：近视眼心更宽



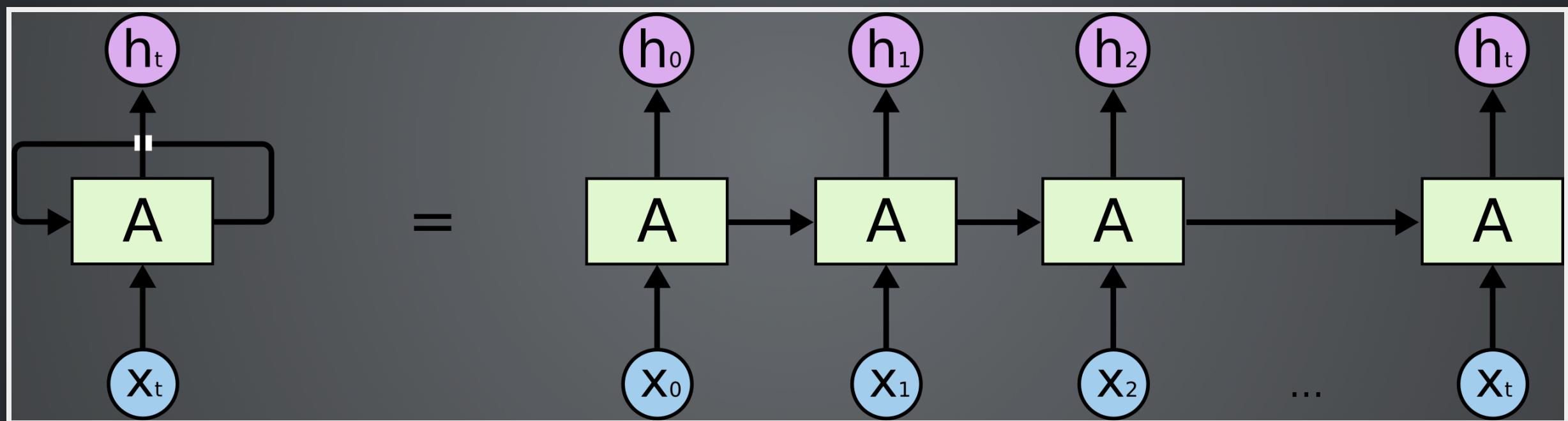
HIMAWARI SAN

CNN应用



RNN

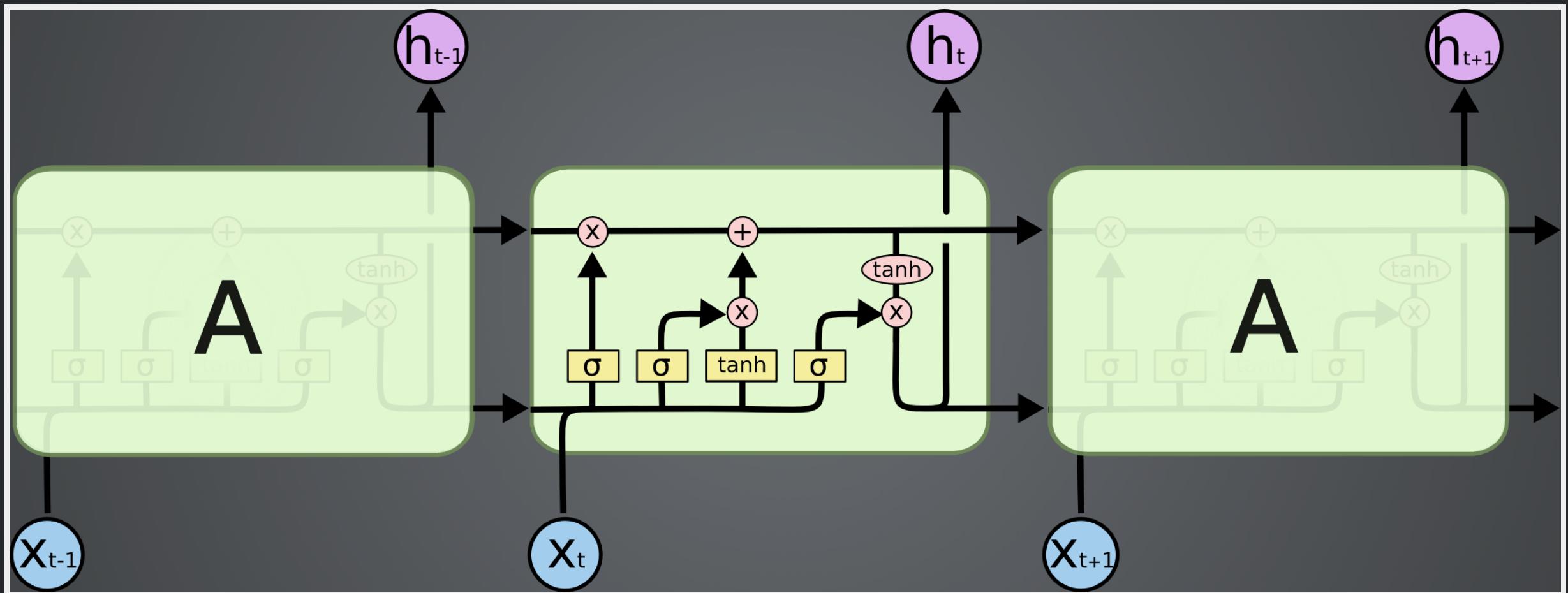
Recurrent Neural Network
循环神经网络



原理 : 状态记忆

LSTM

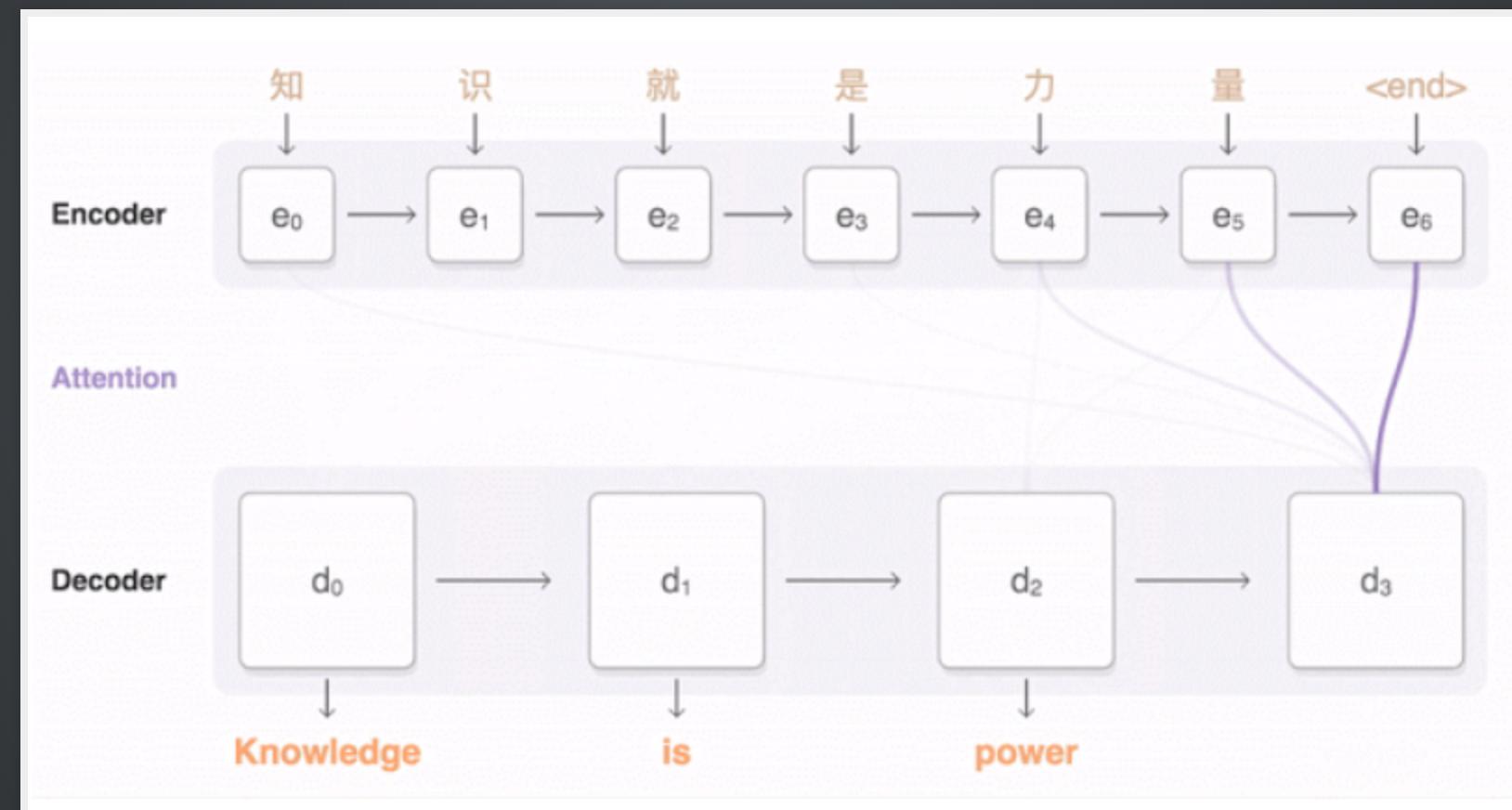
Long-Short Term Memory



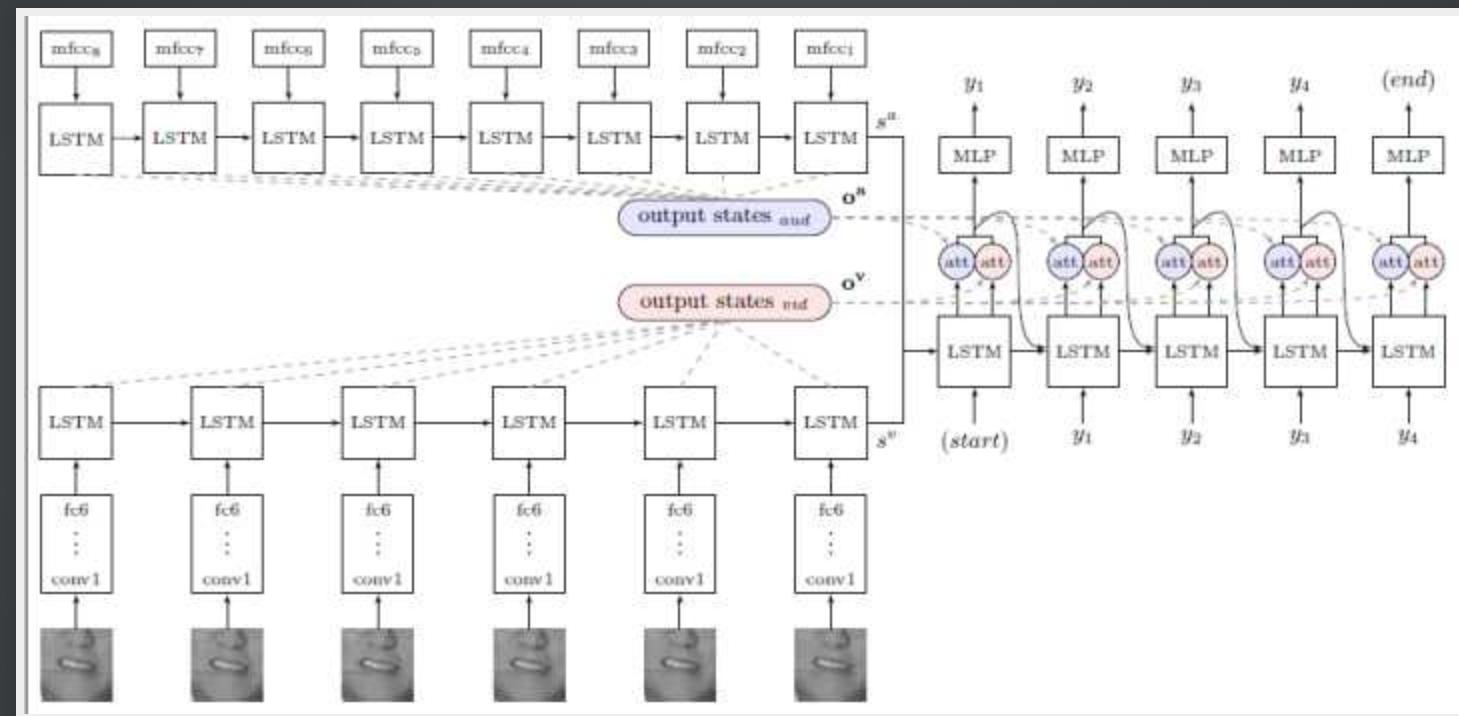
原理：三重门

RNN应用

机器翻译



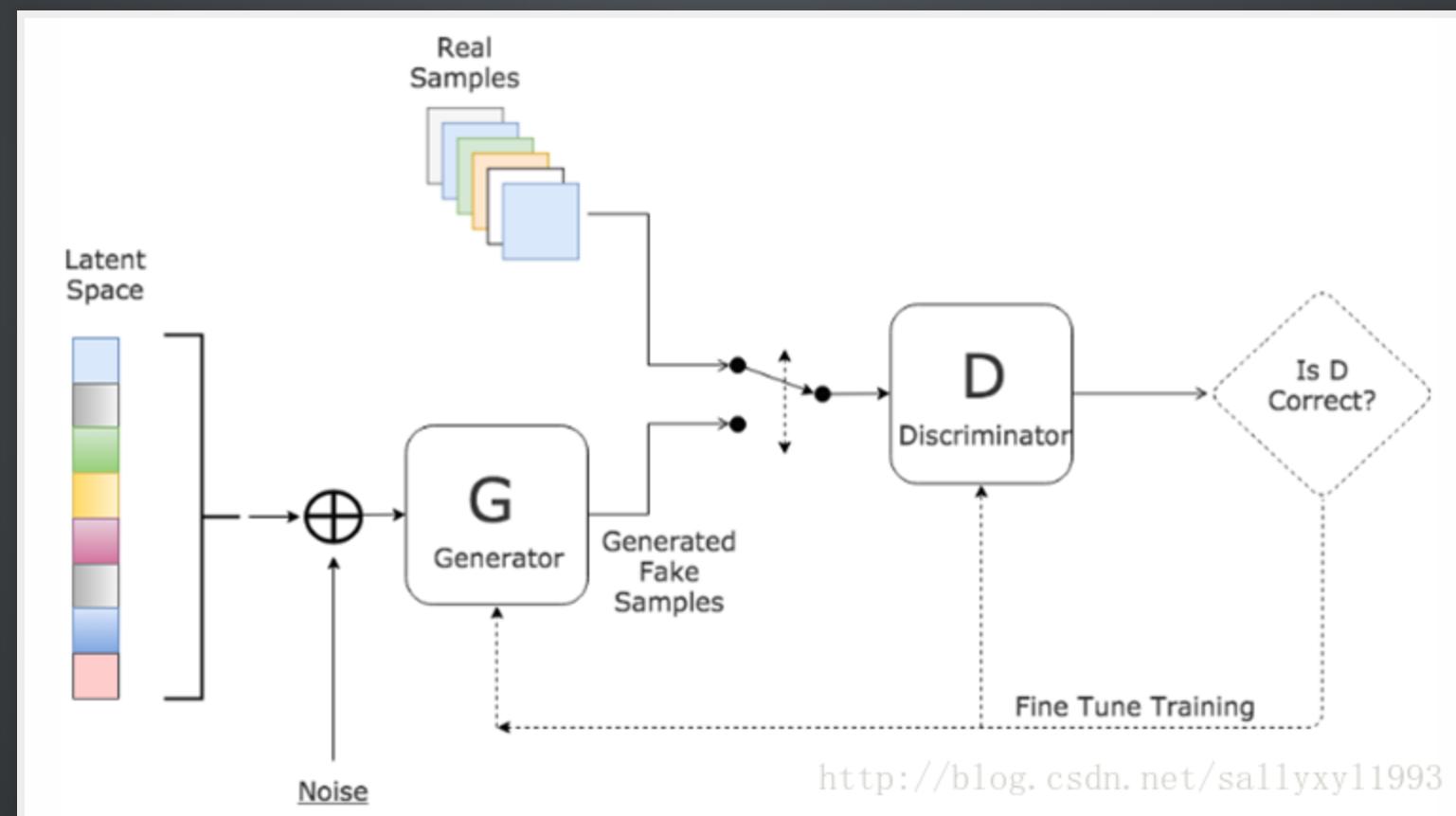
语音识别



前沿技术

GAN

Generative Adversarial Network 生成对抗网络



GAN应用

DCGAN生成女朋友

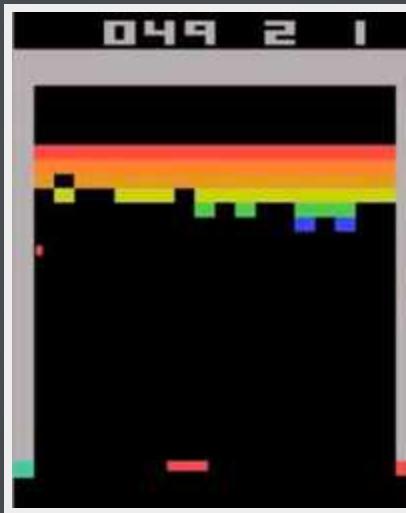
Deep Convolutional Generative Adversarial Network



DRL

Deep Reinforcement Learning

DQN玩游戏



AlphaGo系列



CAPSULE NET

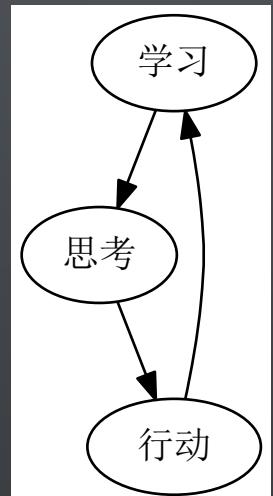
胶囊网络

		capsule	VS.	传统 neuron
来自浅层neuron/capsule的输入		vector(u_i)	scalar(x_i)	
操作	仿射变换	$\hat{u}_{j i} = W_{ij} u_i \quad (\text{Eq. 2})$	—	
	加权	$s_j = \sum_i c_{ij} \hat{u}_{j i} \quad (\text{Eq. 2})$	$a_j = \sum_{i=1}^3 W_i x_i + b$	
	求和			
	非线性激活函数	$v_j = \frac{\ s_j\ ^2}{1 + \ s_j\ ^2} \frac{s_j}{\ s_j\ } \quad (\text{Eq. 1})$	$h_{w,b}(x) = f(a_j)$	
输出		vector(v_j)	scalar(h)	
		$\sum \text{squash}(\cdot) \rightarrow v_j$	 $f(\cdot) : \text{sigmoid, tanh, ReLU, etc.}$	

怎么学？

可能的学习顺序

- 入门：简单易懂
- 经典：全面严谨
- Blog
- Github
- 论文 [arxiv](#)
- 比赛 [kaggle](#) 天池



视频

- Tensorflow教程 by 莫烦
- 网易云课堂的深度学习微专业 by 吴恩达
- 神经网络机器学习课程2012 by Geoffrey Hinton

书籍

实战类

没错，随便买，反正你会去Github上下代码的~~~

专业类

- 《白话深度学习与Tensorflow》 by 高扬、卫峥
- 《深度学习》 by Ian Goodfellow、Yoshua Bengio 和 Aaron Courville

电子版

科普类

- 《终极算法》 by Pedro Domingos

THANKS

- GEOFFREY HINTON的开拓！
- 吴恩达怪蜀黍的布道！
- 吴沫凡小哥哥的小视频！

COME LAST IS BEST •

Thank
you!



