001、让你开发微信二维码电脑端登录，你要怎么实现

python 网站实现扫描二维码关注微信公众号，实现自动登陆
微信公众平台接口测试帐号：https://mp.weixin.qq.com/debug/cgi-bin/sandbox?t=sandbox/login
1，微信提供生成带参数的二维码的接口，参数就是唯一值（场景值，我用的时间戳）
参考 微信文档 生成带参数的二维码
2，网站调用微信系统，获取生成的二维码图片
获取到 ret_url 返回给前端 直接显示 二维码图片
用 scene_id 创建一个 websocket 连接
3 、微信公众号配置
设置好微信 回调的地址 httl.../wx/ 用户扫码后会请求此地址
4、处理推送事件 httl.../wx/

001、python 提取视频中的图片

```
# 导入所需要的库
import cv2
import numpy as np

# 定义保存图片函数
# image:要保存的图片名字
# addr；图片地址与相片名字的前部分
# num: 相片，名字的后缀。int 类型
def save_image(image,addr,num):
    address = addr + str(num)+ '.jpg'
    cv2.imwrite(address,image)

# 读取视频文件
videoCapture = cv2.VideoCapture("2.mp4")#文件地址
# 通过摄像头的方式
# videoCapture=cv2.VideoCapture(1)

#读帧
success, frame = videoCapture.read()
i = 0
while success :
```

```
    i = i + 1
    save_image(frame,'./output/image',i)#图片保存地址
    if success:
        print('save image:',i)
    success, frame = videoCapture.read()
```

加入帧数

```
# 导入所需要的库
import cv2
import numpy as np

# 定义保存图片函数
# image:要保存的图片名字
# addr; 图片地址与相片名字的前部分
# num: 相片，名字的后缀。int 类型
def save_image(image,addr,num):
    address = addr + str(num)+ '.jpg'
    cv2.imwrite(address,image)

# 读取视频文件
videoCapture = cv2.VideoCapture("2.mp4")
# 通过摄像头的方式
# videoCapture=cv2.VideoCapture(1)

#读帧
success, frame = videoCapture.read()
i = 0
timeF = 12
j=0
while success :
    i = i + 1
    if (i % timeF == 0):
        j = j + 1
        save_image(frame,'./output/image',j)
        print('save image:',i)
```

```
    success, frame = videoCapture.read()
```

提取 avi 视频

```
from PIL import Image
import cv2

def splitFrames(videoFileName):
    cap = cv2.VideoCapture(videoFileName) # 打开视频文件
    num = 1
    while True:
        # success 表示是否成功，data 是当前帧的图像数据；.read 读取一帧图像，移动到下一帧
        success, data = cap.read()
        if not success:
            break
        im = Image.fromarray(data) # 重建图像
        im.save('result/mold' +str(num)+".jpg") # 保存当前帧的静态图像
        print(num)
        num = num + 1

    cap.release()

splitFrames('2.avi')
```

002、python 画正余弦函数

```
import numpy as np
from matplotlib import pyplot as plt

plt.figure(figsize=(10,6), dpi=80)
x = np.linspace(-np.pi, np.pi, 256,endpoint=True)
C,S = np.cos(x), np.sin(x)

# 设置线的颜色，粗细，和线型
plt.plot(x, C, color="blue", linewidth=2.5, linestyle="-", label=r'$sin(x)$')
plt.plot(x, S, color="red",  linewidth=2.5, linestyle="-", label=r'$cos(x)$')

# 如果觉得线条离边界太近了，可以加大距离
```

```
plt.xlim(x.min()*1.2, x.max()*1.2)
plt.ylim(C.min()*1.2, C.max()*1.2)

# 当前的刻度并不清晰，需要重新设定,并加上更直观的标签
plt.xticks([-np.pi, -np.pi/2, 0, np.pi/2, np.pi],
          [r'$-\pi$', r'$-\pi/2$', r'$0$', r'$+\pi/2$', r'$+\pi$'])
plt.yticks([-1,1],
          [r'$-1$', r'$1$'])

# 添加图例
plt.legend(loc='upper left')

# plt.gca(), 全称是 get current axis
ax = plt.gca()
ax.spines['right'].set_color('none')
ax.spines['top'].set_color('none')

# 由于我们移动的是左边和底部的轴，所以不用设置这两个也可以
ax.xaxis.set_ticks_position('bottom')
ax.yaxis.set_ticks_position('left')

# 指定 data 类型，就是移动到指定数值
# ax.spines['bottom'].set_position('zero')
ax.spines['bottom'].set_position(('data',0))
ax.spines['left'].set_position(('data',0))

t = 2*np.pi/3

# 利用 plt.plot 绘制向下的一条垂直的线，利用 plt.scatter 绘制一个点。
plt.plot([t,t],[0,np.cos(t)], color ='blue', linewidth=2.5, linestyle="--")
plt.scatter([t,],[np.cos(t),], 50, color ='blue')

plt.annotate(r'$sin(\frac{2\pi}{3})=\frac{\sqrt{3}}{2}$',
         xy=(t, np.sin(t)), xycoords='data',
```

```
        xytext=(+10, +30), textcoords='offset points', fontsize=16,
        arrowprops=dict(arrowstyle="->", connectionstyle="arc3,rad=.2"))

# 利用 plt.plot 绘制向上的一条垂直的线，利用 plt.scatter 绘制一个点。
plt.plot([t,t],[0,np.sin(t)], color ='red', linewidth=2.5, linestyle="--")
plt.scatter([t,],[np.sin(t),], 50, color ='red')

plt.annotate(r'$cos(\frac{2\pi}{3})=-\frac{1}{2}$',
        xy=(t, np.cos(t)), xycoords='data',
        xytext=(-90, -50), textcoords='offset points', fontsize=16,
        arrowprops=dict(arrowstyle="->", connectionstyle="arc3,rad=.2"))

plt.show()
```

003、Python 机器学习常用到的几个库（建议收藏）

Python 仅仅是作为语言工具，掌握基础语法即可。重点学习，应该还是机器学习里面涉及到的机器学习的一些库，以及常用的机器学习框架等内容。比如：

- Matplotlib 数据绘图基础课程：主要对 Matplotlib 常用的绘图方法进行简介。并介绍其绘图常用的方法。
- NumPy 数值计算基础课程：讲解 NumPy 强大的高维度数组处理与矩阵运算能力。除此之外，NumPy 还内建了大量的函数，方便你快速构建数学模型。
- SciPy 科学计算基础课程：将带你了解 SciPy 的基础用法。
- Pandas 时间序列数据处理：主要讲解利用 Pandas 对数据集进行快速读取、转换、过滤、分析等一系列操作。同样，Pandas 已经被证明为是非常强大的用于处理时间序列数据的工具。重点介绍所有 Pandas 在时间序列数据上的处理方法。
- PyTorch 深度学习基础课程：我们将学习 PyTorch 的基础语法，了解 Autograd 自动求导机制，并最终利用 PyTorch 构建可用于图像分类任务的人工神经网络。
- TensorFlow 深度学习基础课程：讲解使用 TensorFlow 搭建神经网络常用的低阶 API 和其集成的高阶 API ： Keras。
- TensorFlow 2 深度学习入门与实践：2019 年，TensorFlow 正式推出了 2.0 版本，也意味着 TensorFlow 从 1.x 正式过度到 2.x 时代。

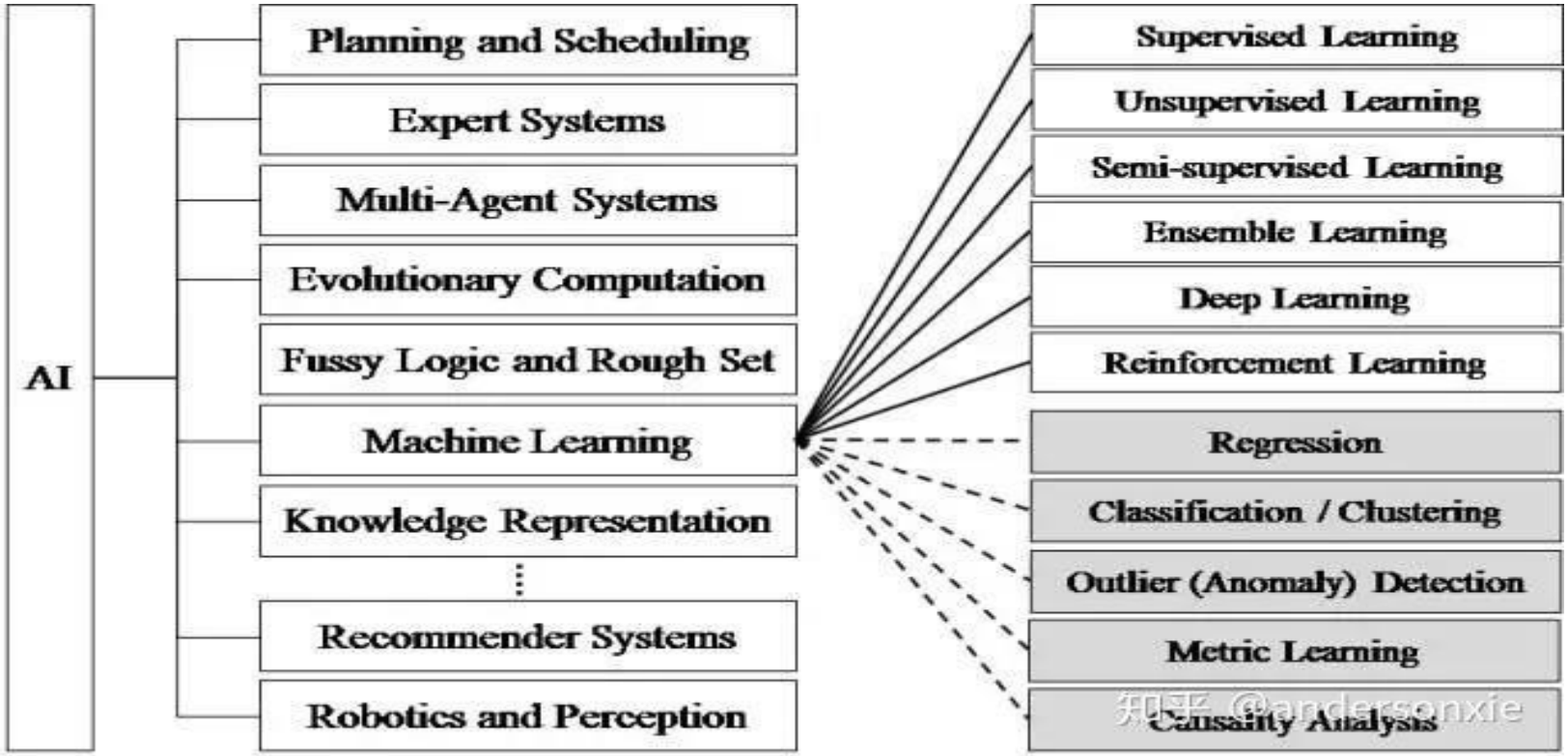
004、机器学习、深度学习、强化学习、迁移学习和人工智能的联系和区别？

人工智能 (Artificial Intelligence, AI)浪潮正在席卷全球，在上一讲中，我们给出了人工智能的定义、话题、四大技术分支、主要应用领域和三种形态：***弱人工智能、强人工智能和超级人工智能***，让大家了解了人工智能这个耳熟能详的概念。其中，我们区别了弱人工智能和强人工智能的概念：前者让机器具备观察和感知的能力，可以做到一定程度的理解和推理；而强人工智能让机器获得自适应能力，解决一些之前没有遇到过的问题。电影里的人工智能多半都是在描绘强人工智能，而这部分在目前的现实世界里难以真正实现；目前的科研工作主要集中在弱人工智能这部分，并且已经取得了一系列的重大突破。

在这一讲中，我们打算理一下人工智能的发展历史，以及各个历史阶段当中侧重的不同算法。

1956 年，几个计算机科学家相聚在***达特茅斯会议***，提出了“人工智能”的概念，梦想着用当时刚刚出现的计算机来构造复杂的、拥有与人类智慧同样本质特性的机器。其后，人工智能就一直萦绕于人们的脑海之中，并在科研实验室中慢慢孵化。之后的几十年，人工智能一直在两极反转，或被称作人类文明耀眼未来的预言，或被当成技术疯子的狂想扔到垃圾堆里。直到 2012 年之前，这两种声音还在同时存在。

2012 年以后，得益于数据量的上涨、运算力的提升和机器学习新算法——***深度学习***的出现，人工智能开始大爆发，研究领域也在不断扩大，下图展示了人工智能研究的各个分支，包括计划调度、专家系统、多智能体系统、进化计算、模糊逻辑、机器学习、知识表示、计算机视觉、自然语言处理、推荐系统、机器感知等等。



诸多媒体流行词汇萦绕在我们耳边，比如人工智能（Artificial Intelligence)、机器学习（Machine Learning)、深度学习（Deep Learning)、强化学习（Reinforcement Learning)、迁移学习（Transfer Learning)，不少人对这些高频词汇的含义及其背后的关系感到困惑；这一讲中，我们会从它们的发展历程、概念、算法种类进行介绍，并且理清它们之间的关系和区别；具体的算法原理留到之后的推送当中详解。

1. 机器学习

弱人工智能是如何实现的，“智能”又从何而来呢？这主要归功于一种实现人工智能的方法——机器学习。

1.1 机器学习的定义一：机器学习定义的第一类答案是 ***IBM 提出的认知计算（Cognitive Computing)***。其目标是构建不需要显式编程的机器（计算机、软件、机器人、网站、移动应用、设备等）。这种机器学习观点可追溯到 ***Arthur Samuel*** 在 1959 年的定义，“机器学习：让计算机无需显式编程也能学习的研究领域”。Arthur Samuel 是机器学习的创始人之一，在 IBM 的时候，他开发了一个程序来学习如何在西洋棋棋艺上超过他。

1.2 机器学习的定义二：Samuel 的定义很好，但可能有点太模糊。1998 年，另一位著名的机器学习研究者 ***Tom Mitchell*** 提出了一个更精确的定义，“正确提出的学习问题：如果计算机程序对于任务 T 的性能度量 P 通过经验 E 得到了提高，则认为此程序对经

验 E 进行了学习”。为了阐述清楚，我们举一个例子：在下棋程序中，经验 E 指的就是程序的上万次的自我联系的经验，任务 T 就是下棋，性能度量 P 指的就是在比赛过程中取胜的概率，有了性能指标后，我们就能告诉系统是否学习该经验。

1.3 机器学习的算法分类：上述定义为机器学习设定了清晰的目标，但是，它们没有告诉我们如何实现该目标，我们应该让定义更明确一些。这就需要第二类定义，这类定义描述了机器学习算法，以下是一些流行的定义。在每种情况下，都会为算法提供一组示例供其学习。

(1) ***监督式学习***：为算法提供训练数据，数据中包含每个示例的“正确答案”；例如，一个检测信用卡欺诈的监督学习算法接受一组记录的交易作为输入，对于每笔交易，训练数据都将包含一个表明它是否存在欺诈的标记。

(2) ***无监督学习***：该算法在训练数据中寻找结构，比如寻找哪些示例彼此类似，并将它们分组到各个集群中。

1.4 机器学习的问题分类：我们希望在机器学习算法分类的基础上更具体一些，一种方法是通过分析机器学习任务能解决的问题类型，对任务进行细化：

(1) ***分类***，一种监督学习问题，其中要学习的答案是有限多个可能值之一；例如，在信用卡示例中，该算法必须学习如何在“欺诈”与“诚信”之间找到正确的答案，在仅有两个可能的值时，我们称之为二元分类问题；用于实现分类的常用算法包括：支持向量机（SVM）、提升（boosted）决策树和袋装（bagged）决策树、k-最近邻、朴素贝叶斯（Naïve Bayes）、判别分析、逻辑回归和神经网络。

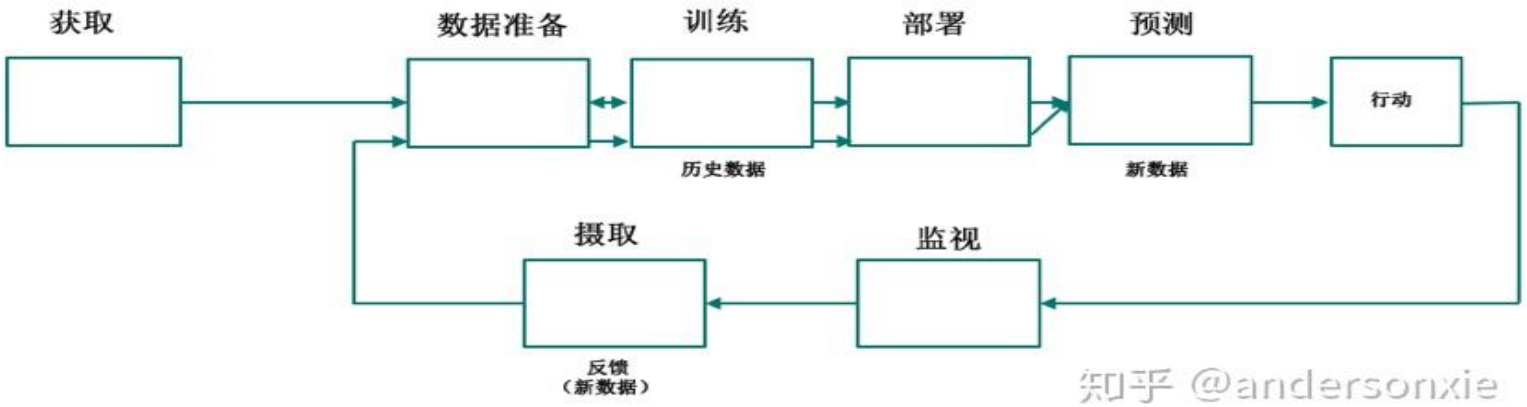
(2) ***回归***，一种监督学习问题，其中要学习的答案是一个连续值。例如，可为算法提供一条房屋销售及其价格的记录，让它学习如何设定房屋价格；常用回归算法包括：线性模型、非线性模型、规则化、逐步回归、提升（boosted）和袋装（bagged）决策树、神经网络和自适应神经模糊学习。

(3) ***细分（聚类）***，一种无监督学习问题，其中要学习的结构是一些类似示例的集群。例如，市场细分旨在将客户分组到有类似购买行

为的人群中；用于执行聚类的常用算法包括：k-均值和 k-中心点（k-medoids）、层次聚类、高斯混合模型、隐马尔可夫模型、自组织映射、模糊 c-均值聚类法和减法聚类。

(4) ***网络分析***，一种无监督学习问题，其中要学习的结构是有关网络中的节点的重要性和作用的信息；例如，网页排名算法会分析网页及其超链接构成的网络，并寻找最重要的网页。谷歌等 Web 搜索引擎使用的就是这种算法，其他网络分析问题包括社交网络分析。

1.5 机器学习工作流及定义三：上述两个定义的问题在于，开发一个机器学习算法并不足以获得一个能学习的系统。诚然，机器学习算法与学习系统之间存在着差距。我给出一个***机器学习工作流***，如下图所示；机器学习算法被用在工作流的“训练”步骤中，然后它的输出（一个经过训练的模型）被用在工作流的“预测”部分中。好的与差的机器算法之间的区别在于，我们在“预测”步骤中获得的预测质量。这就引出了机器学习的另一个定义：“机器学习的目的是从训练数据中学习，以便对新的、未见过的数据做出尽可能好的预测”。

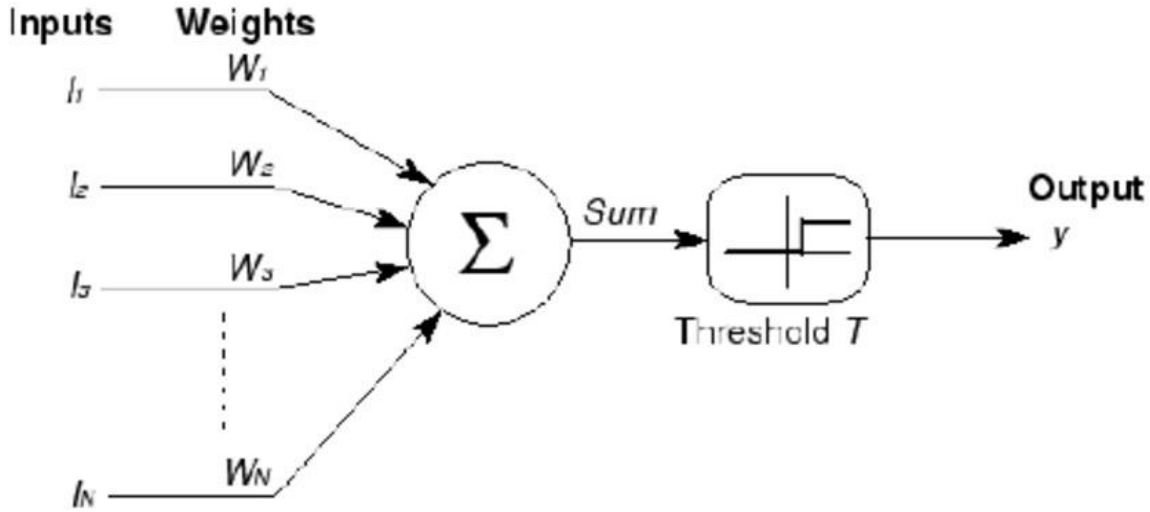


2. 深度学习

我们将从深度学习的发展历程、深度学习的概念、深度神经网络的分类几个方面来阐述。

2.1 深度学习发展历史：

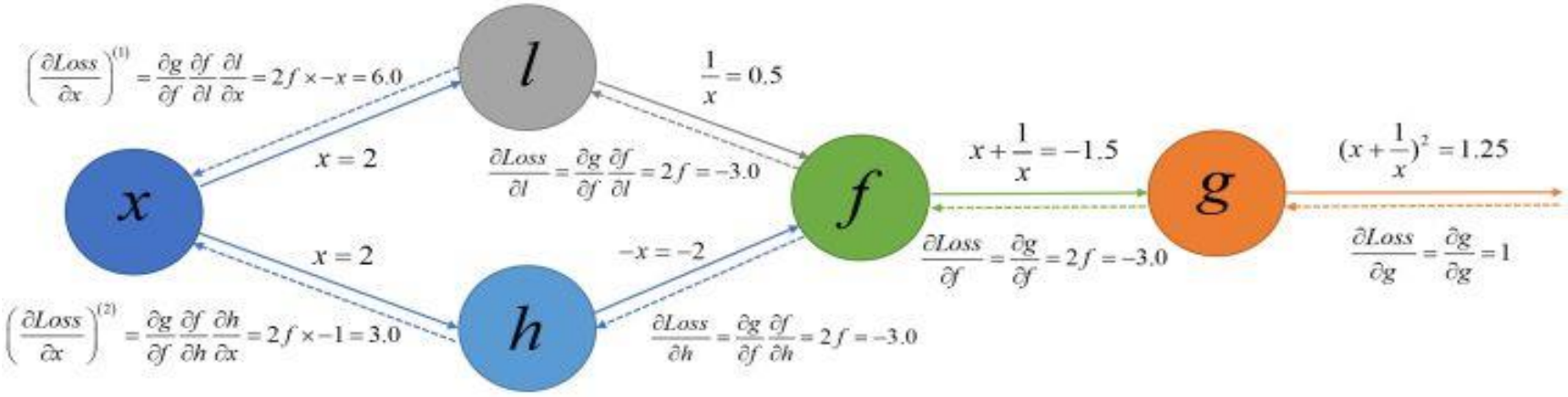
(1) 1943 年，由神经科学家***麦卡洛克*** (W.S.McCilloch) 和数学家***皮兹*** (W.Pitts) 在《数学生物物理学公告》上发表论文《神经活动中内在思想的逻辑演算》(A Logical Calculus of the Ideas Immanent in Nervous Activity)。建立了神经网络和数学模型，称为 ***MCP 模型***。所谓 MCP 模型，其实是按照生物神经元的结构和工作原理构造出来的一个抽象和简化了的模型，也就诞生了所谓的“模拟大脑”，人工神经网络的大门由此开启。MCP 当时是希望能够用计算机来模拟人的神经元反应的过程，该模型将神经元简化为了三个过程：输入信号线性加权，求和，非线性激活（阈值法），如下图所示：



(2) 1958 年，计算机科学家***罗森布拉特*** (Rosenblatt) 提出了两层神经元组成的神经网络，称之为“***感知器***” (Perceptrons)，第一次将 MCP 用于机器学习分类：“感知器”算法算法使用 MCP 模型对输入的多维数据进行二分类，且能够使用梯度下降法从训练样本中自动学习更新权值。1962 年,该方法被证明为能够收敛，理论与实践效果引起第一次神经网络的浪潮。

(3) 1969 年：纵观科学发展史，无疑都是充满曲折的，深度学习也毫不例外。1969 年，美国数学家及人工智能先驱 ***Marvin Minsky*** 在其著作中证明了感知器本质上是一种***线性模型***(Linear Model)，只能处理***线性分类问题***，就连最简单的***亦或*** (XOR) 问题都无法正确分类。这等于直接宣判了感知器的死刑，神经网络的研究也陷入了将近 20 年的停滞。

(4) 1986 年，由***神经网络之父 Geoffrey Hinton*** 在 1986 年发明了适用于***多层感知器*** (MLP)的***反向传播*** BP (Backpropagation) 算法，并采用 ***Sigmoid*** 进行非线性映射，有效解决了***非线性分类和学习***的问题。该方法引起了神经网络的第二次热潮。



Sigmoid 函数是一个在生物学中常见的 S 型的函数（S 型生长曲线）。在信息科学中，由于其单增以及反函数单增等性质，常被用作神经网络的阈值函数，将变量映射到[0, 1]：Sigmoid(x)=1 / (1 + exp(-x))

(5) 90 年代时期，1991 年 BP 算法被指出存在梯度消失问题，也就是说在误差梯度后项传递的过程中，后层梯度以乘性方式叠加到前层，由于 Sigmoid 函数的饱和特性，后层梯度本来就小，误差梯度传到前层时几乎为 0，因此无法对前层进行有效的学习，该问题直接阻碍了深度学习的进一步发展；此外 90 年代中期，支持向量机（SVM）算法诞生等各种浅层机器学习模型被提出，SVM 也是一种有监督的学习模型，应用于模式识别，分类以及回归分析等。支持向量机以统计学为基础，和神经网络有明显的差异，支持向量机等算法的提出再次阻碍了深度学习的发展。

(6) 2006 年——2012 年（发展期），2006 年，加拿大多伦多大学教授、机器学习领域泰斗、神经网络之父——***Geoffrey Hinton*** 和他的学生 ***Ruslan Salakhutdinov*** 在顶尖学术刊物《科学》上发表了一篇文章，该文章提出了深层网络训练中梯度消失问题的解决方案：无监督预训练对权值进行初始化+有监督训练微调。斯坦福大学、纽约大学、加拿大蒙特利尔大学等成为研究深度学习的重镇，至此开启了深度学习在学术界和工业界的浪潮。2011 年，***修正线性单元*** (Rectified Linear Unit, ReLU) 激活函数被提出，该激活函数能够有效的抑制梯度消失问题：ReLU(x)=max(0,x)

2011 年以来，微软首次将深度学习应用在语音识别上，取得了重大突破。微软研究院和 Google 的语音识别研究人员先后采用***深度神经网络*** (DNN) 技术降低语音识别错误率 20%~30%，是语音识别领域十多年来最大的突破性进展。2012 年，DNN 技术在图像识别领域取得惊人的效果，在 ImageNet 评测上将错误率从 26%降低到 15%。在这一年，DNN 还被应用于制药公司的 DrugeActivity 预测问题，并获得世界最好成绩。

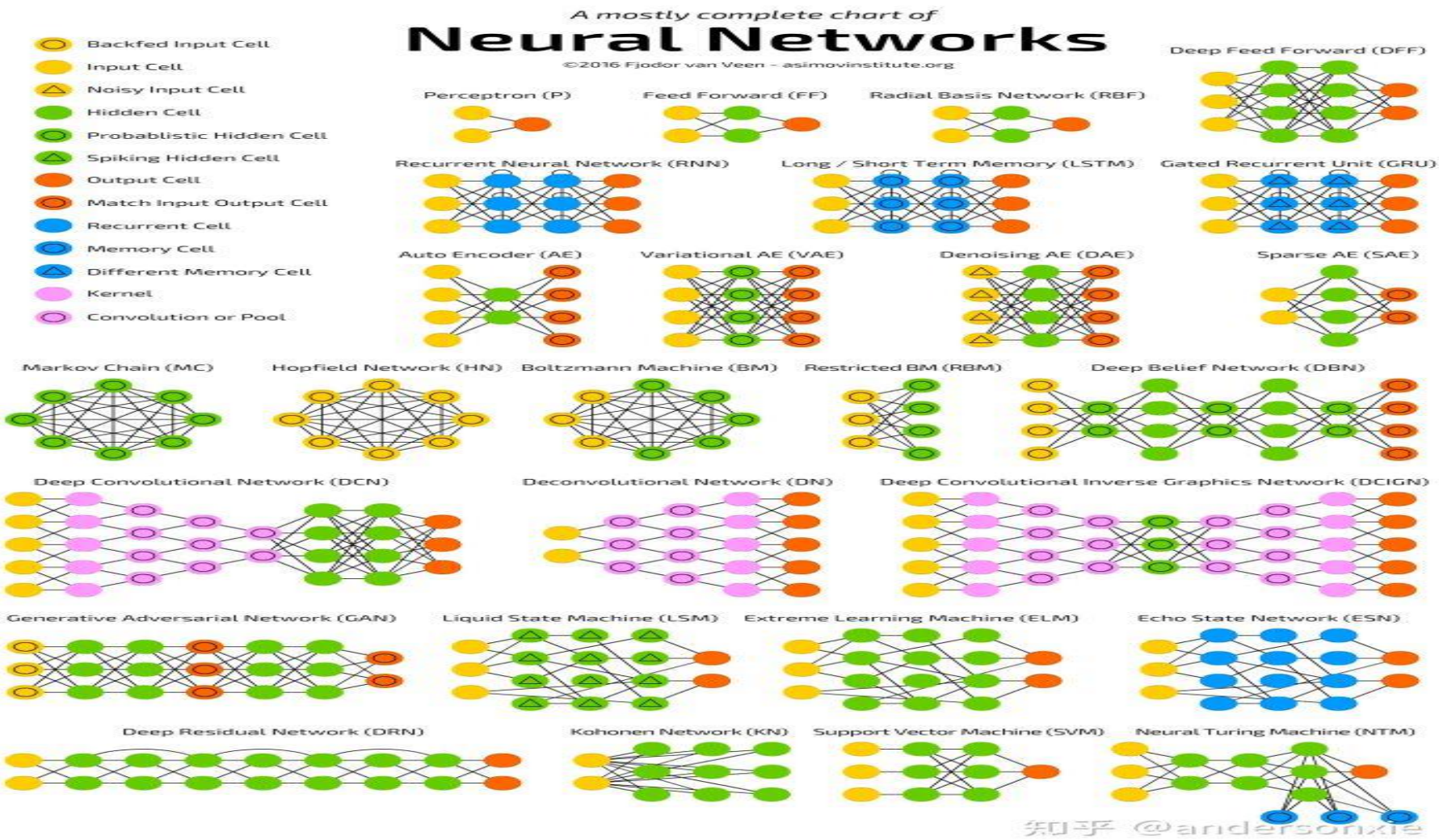
(7) 2012 年——2017 年（爆发期），2012 年，Hinton 课题组为了证明深度学习的潜力，首次参加 ***ImageNet 图像识别比赛***，其通过构建的***卷积神经网络（CNN）AlexNet*** 一举夺得冠军，且碾压第二名（SVM 方法）的分类性能。也正是由于该比赛，CNN 吸引到了众多研究者的注意。

2016 年，随着谷歌（Google）旗下的 ***DeepMind*** 公司基于深度学习开发的 ***AlphaGo*** 以 4：1 的比分战胜了国际顶尖围棋高手***李世石***，深度学习的热度一时无两。后来，AlphaGo 又接连和众多世界级围棋高手过招，均取得了完胜。这也证明了在围棋界，基于深度学习技术的机器人已经超越了人。2016 年末 2017 年初，该程序在中国棋类网站上以“大师”（***Master***）为注册帐号与中日韩数十位围棋高手进行快棋对决，连续 60 局无一败绩；2017 年 5 月，在中国乌镇围棋峰会上，它与排名世界第一的世界围棋冠军***柯洁***对战，以 3 比 0 的总比分获胜。围棋界公认阿尔法围棋的棋力已经超过人类职业围棋顶尖水平；同年，基于强化学习算法的 AlphaGo 升级版 ***AlphaGo Zero*** 横空出世，其采用“从零开始”、“无师自通”的学习模式，以 100:0 的比分轻而易举打败了之前的 AlphaGo。除了围棋，它还精通国际象棋等其它棋类游戏，可以说是真正的棋类“天才”。此外在这一年，深度学习的相关算法在***医疗、金融、艺术、无人驾驶***等多个领域均取得了显著的成果。所以，也有专家把 2017 年看作是深度学习甚至是人工智能发展最为突飞猛进的一年。

2.2 深度学习的定义：深度学习是机器学习中一种基于对数据进行***表征学习***的算法。观测值（例如一幅图像）可以使用多种方式来表示，如每个像素强度值的向量，或者更抽象地表示一系列边、特定形状的区域等。而使用某些特定的表示方法更容易从实例中学习任

务（例如人脸识别、面部表情识别）。深度学习的好处是用非监督式或半监督式的特征学习和分层特征提取高效算法来替代手工获取特征。

2.3 深度神经网络：深度学习的模型有很多，目前开发者最常用的深度学习模型与架构包括卷积神经网络（CNN)、深度置信网络(DBN)、受限玻尔兹曼机（RBM)、递归神经网络（RNN & LSTM & GRU)、递归张量神经网络（RNTN)、自动编码器(AutoEncoder)、生成对抗网络（GAN)等等，更多的模型可以参考下图：

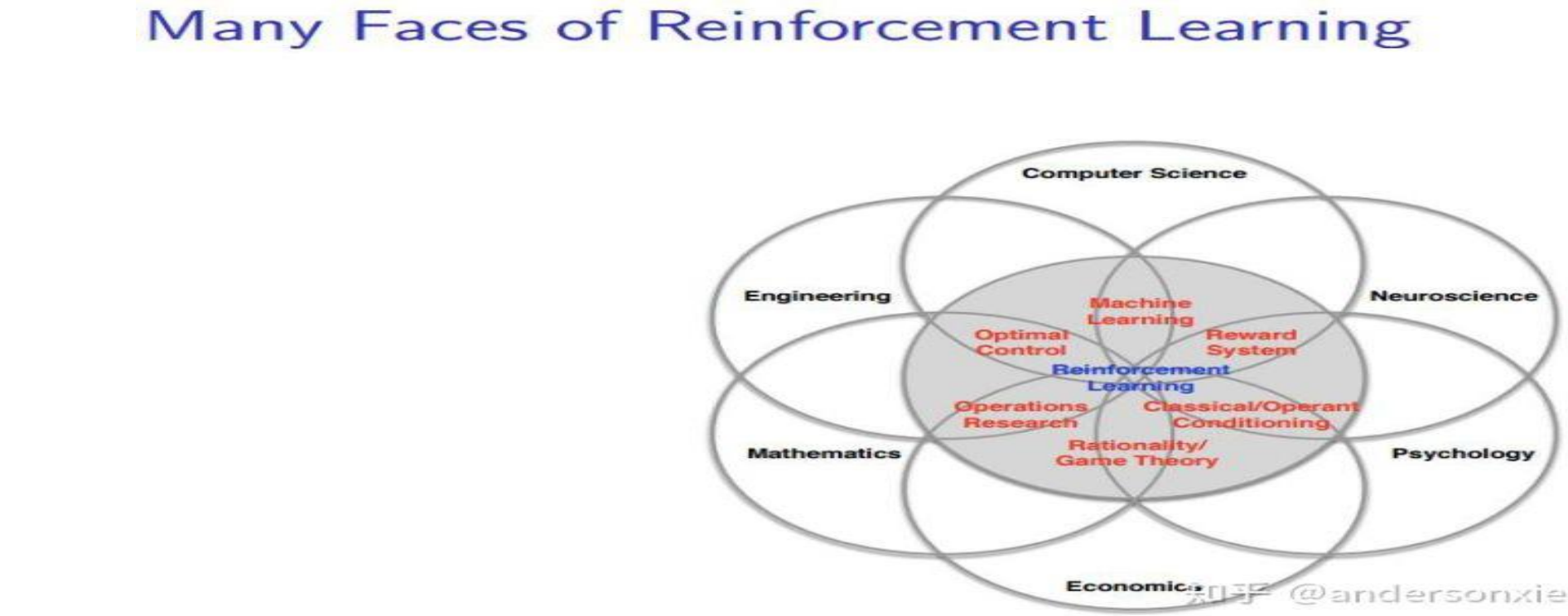


3. 强化学习

强化学习在各个领域当中应用十分广泛，在这里主要给出它的定义、适用范围、组成成分和交互过程。

3.1 强化学习的定义：强化学习（Reinforcement learning, RL）是机器学习中的一个领域，强调如何基于环境而行动，以取得最大化的预期利益；其灵感来源于***心理学中的行为主义理论***，即有机体如何在环境给予的奖励或惩罚的刺激下，逐步形成对刺激的预期，产生能获得最大利益的习惯性行为。

3.2 强化学习适用范围：尽管我们在机器学习社区中广泛使用强化学习，但强化学习不仅仅是一个人工智能术语，它是许多领域中的一个中心思想，如下图（强化学习的多个方面，Many Faces of Reinforcement Learning）所示。事实上，许多这些领域面临着与机器学习相同的问题：如何优化决策以实现最佳结果，这就是***决策科学*** (Science of Decision-Making)；在***神经科学***中，人类研究人脑并发现了一种遵循著名的强化算法的奖励系统；在***心理学***中，人们研究的经典条件反射和操作性条件反射，也可以被认为是一个强化问题；类似的，在***经济学***中我们研究理性博弈论；在***数学***中我们研究运筹学；在***工程学***中我们研究优化控制；所有的这些问题都可以被认为一种强化学习问题——它们研究同一个主题，即为了实现最佳结果而优化决策。

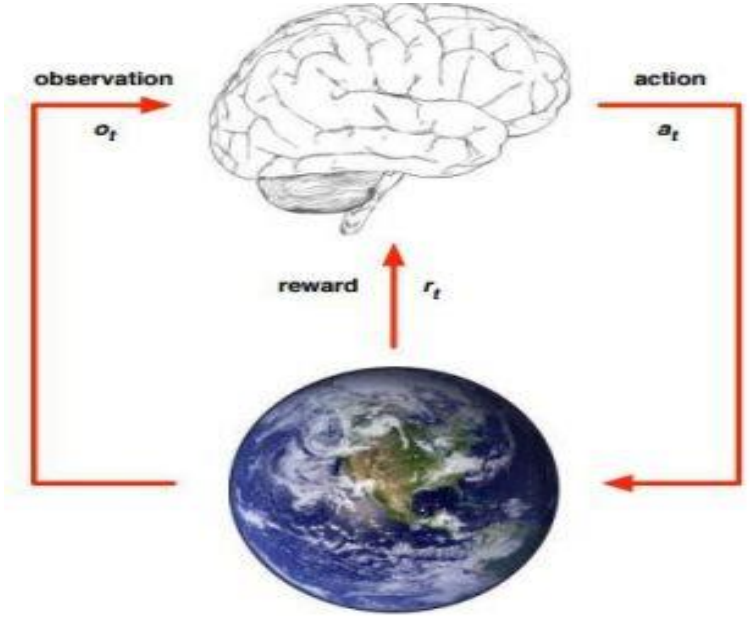


3.3 强化学习基本组成成分：强化学习主要由智能体（Agent)、环境（Environment)、状态（State)、动作（Action)、奖励(Reward)、策略（Policy)、目标（Objective）组成。

(1) ***智能体***：强化学习的本体，作为学习者或者决策者；

(2) ***环境***：强化学习智能体以外的一切，主要由状态集合组成；

(3) ***状态***：一个表示环境的数据，状态集则是环境中所有可能的状态；
(4) ***动作***：智能体可以做出的动作，动作集则是智能体可以做出的所有动作；
(5) ***奖励***：智能体在执行一个动作后，获得的正/负反馈信号，奖励集则是智能体可以获得的所有反馈信息；
(6) ***策略***: 强化学习是从环境状态到动作的映射学习，称该映射关系为策略。通俗的理解，即智能体如何选择动作的思考过程称为策略；
(7) ***目标***：智能体自动寻找在连续时间序列里的最优策略，而最优策略通常指最大化长期累积奖励。在此基础上，智能体和环境通过状态、动作、奖励进行交互的方式为：智能体执行了某个动作后，环境将会转换到一个新的状态，对于该新的状态环境会给出奖励信号（正奖励或者负奖励）。随后，智能体根据新的状态和环境反馈的奖励，按照一定的策略执行新的动作。



- At each step t the agent:
  - Executes action a_t
  - Receives observation o_t
  - Receives scalar reward r_t
- The environment:
  - Receives action a_t
  - Emits observation o_{t+1}
  - Emits scalar reward r_{t+1}

智能体通过强化学习，可以知道自己在什么状态下，应该采取什么样的动作使得自身获得最大奖励；因此，强化学习实际上是智能体在与环境进行交互的过程中，学会最佳决策序列。由于智能体与环境的交互方式与人类与环境的交互方式类似，可以认为强化学习是一套通用的学习框架，可用来解决通用人工智能的问题。因此强化学习也被称为通用人工智能的机器学习方法。

4. 迁移学习

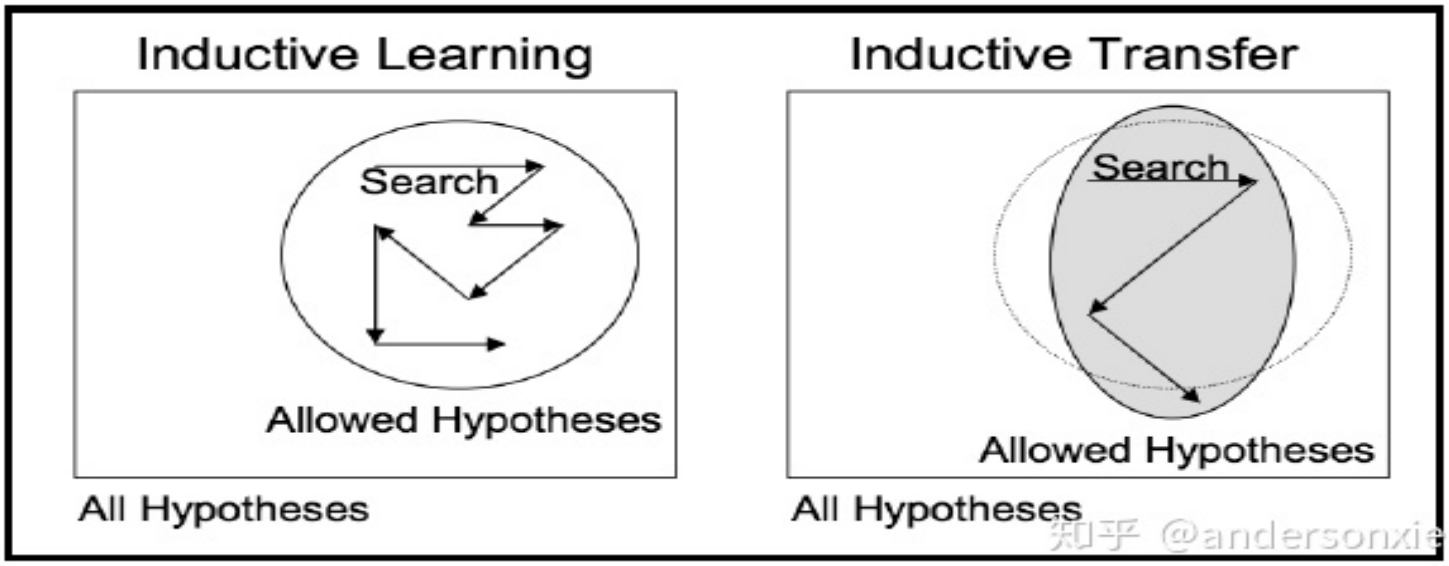
迁移学习（Transfer Learning) 顾名思义就是就是把已学训练好的模型参数迁移到新的模型来帮助新模型训练。考虑到大部分数据或任务是存在相关性的，所以通过迁移学习我们可以将已经学到的模型参数（也可理解为模型学到的知识）通过某种方式来分享给新模型从而加快并优化模型的学习效率不用像大多数网络那样***从零学习（Starting From Scratch)***。

4.1 迁移学习的定义：迁移学习是机器学习技术的一种，其中在一个任务上训练的模型被重新利用在另一个相关的任务上，定义一：“迁移学习和领域自适应指的是将一个任务环境中学到的东西用来提升在另一个任务环境中模型的泛化能力” ——2016 年“Deep

Learning”，526 页；迁移学习也是一种优化方法，可以在对另一个任务建模时提高进展速度或者是模型性能，定义二：“迁移学习就是通过从已学习的相关任务中迁移其知识来对需要学习的新任务进行提高。”——第 11 章：转移学习，机器学习应用研究手册，2009 年；同在 2009 年，Sinno Jialin Pan 和 Qiang Yang 发表了一篇迁移学习的《A Survey on Transfer Learning》，他们给出了迁移学习的数学定义三：

Definition 1 *(Transfer Learning)* Given a source domain $\mathcal{D}_S$ and learning task $\mathcal{T}_S$, a target domain $\mathcal{D}_T$ and learning task $\mathcal{T}_T$, *transfer learning* aims to help improve the learning of the target predictive function $f_T(\cdot)$ in $\mathcal{D}_T$ using the knowledge in $\mathcal{D}_S$ and $\mathcal{T}_S$, where $\mathcal{D}_S \neq \mathcal{D}_T$, or $\mathcal{T}_S \neq \mathcal{T}_T$.

4.2 迁移学习在深度学习中的应用：迁移学习还与***多任务学习***和***概念漂移***等问题有关，它并不完全是深度学习的一个研究领域。尽管如此，由于训练深度学习模型所需耗费巨大资源，包括大量的数据集，迁移学习便成了深度学习是一种很受欢迎的方法。但是，只有当从第一个任务中学到的模型特征是容易泛化的时候，迁移学习才能在深度学习中起到作用。“在迁移学习中，我们首先在基础数据集和任务上训练一个基础网络，然后将学习到的特征重新调整或者迁移到另一个目标网络上，用来训练目标任务的数据集。如果这些特征是容易泛化的，且同时适用于基本任务和目标任务，而不只是特定于基本任务，那迁移学习就能有效进行。”——深度神经网络中的特征如何迁移的？这种用于深度学习的迁移学习形式被称为***推导迁移 (Inductive Transfer)***。就是通过使用合适但不完全相同的相关任务的模型，将模型的范围（模型偏差）以有利的方式缩小。



举个例子，使用图像数据作为输入的预测模型问题中进行迁移学习是很常见的，它可能是一个以照片或视频数据作为输入的预测任务。 对于这些类型的问题，通常会使用预先训练好的深度学习模型来处理大型的和具有挑战性的图像分类任务，例如 ImageNet 1000 级照片分类竞赛，我们可以下载以下模型，并合并到以自己图像数据作为输入的新模型中：牛津的 VGG 模型、谷歌的 Inception 模型、微软的 ResNet 模型。

4.3 迁移学习使用方法：我们可以在自己的预测模型问题上使用迁移学习，通常有两种方法：开发模型方法和预训练模型方法。

对于***开发模型方法***，分为四步：

(1) ***选择源任务***：必须选择一个与大量数据相关的预测模型问题，这个大量的数据需要与输入数据，输出数据和/或从输入到输出数据映射过程中学习的概念之间存在某种关系。

(2) ***开发源模型***：接下来，必须为这个第一项任务开发一个熟练的模型；该模型必须比原始模型更好，以确保一些特征学习已经发挥了其作用。

(3) ***重用模型***：然后可以将适合元任务的模型用作感兴趣的另一个任务模型的起点；这取决于所使用的建模技术，可能涉及到了全部或部分模型。

(4) ***调整模型***：可选项，对感兴趣任务的调整输入—输出配对数据或改进模型。

对于***预训练模型方法***，分为三步：

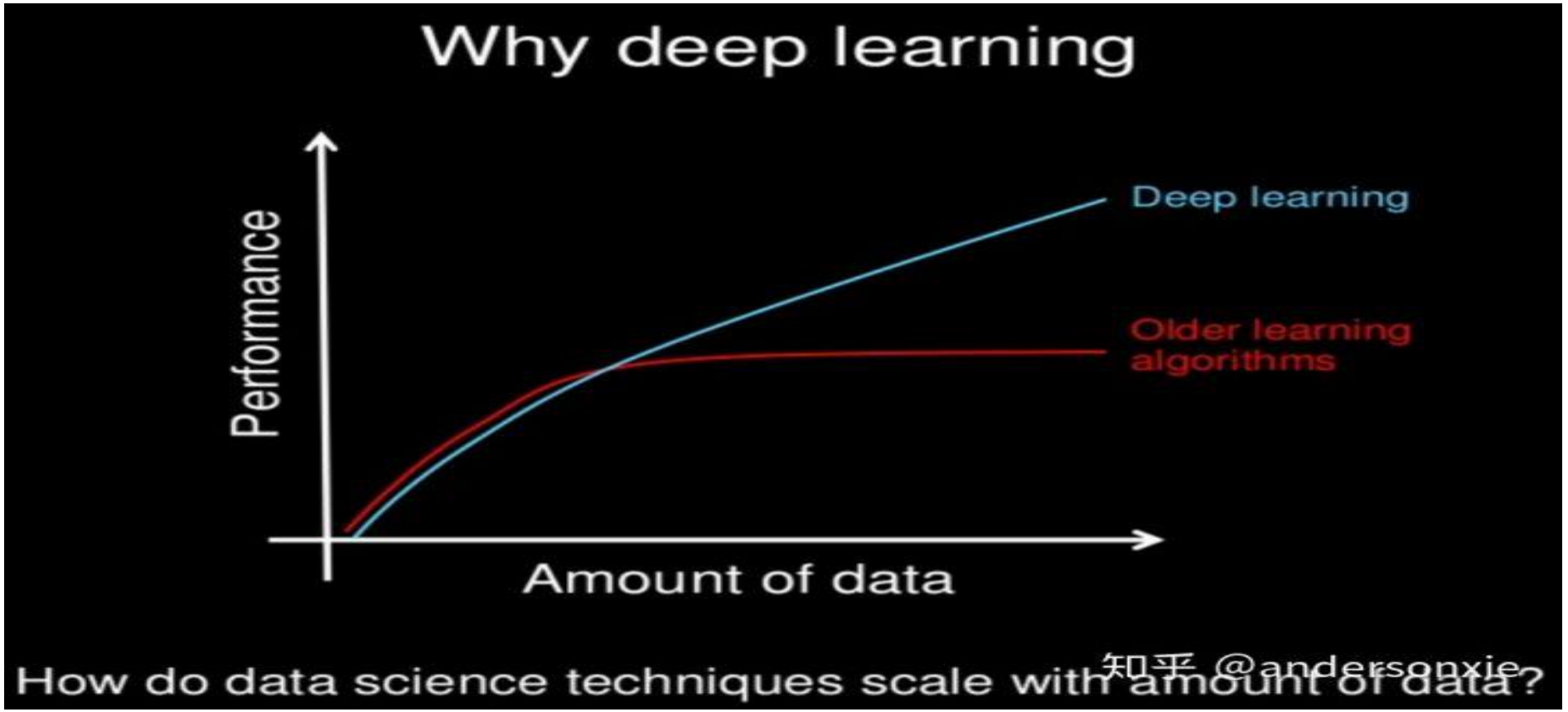
(1) ***选择源任务***：从可用的模型中选择预训练的元模型，许多研究机构会发布已经在大量的且具有挑战性的数据集上训练好的模型，在可用模型的模型池里面也能找到这些模型。

(2) ***重用模型***：然后可以将预训练的模型用作感兴趣的另一个任务模型的起点，这取决于所使用的建模技术，可能涉及使用全部或部分模型。
(3) ***调整模型***：可选项，对感兴趣任务的调整输入—输出配对数据或改进模型。
其中，第二类迁移学习方法在深度学习领域是很常见的。

5. 机器学习 VS 深度学习 VS 强化学习 VS 迁移学习 VS 人工智能?

以上我们分别介绍了机器学习、深度学习、强化学习、迁移学习算法，那么它们之前存在怎样的关系呢？它们和人工智能又存在怎样的关联呢？

5.1 机器学习 VS 深度学习：机器学习是一种实现人工智能的方法，深度学习是一种实现机器学习的技术，目前，学术界的各个人工智能研究方向（计算机视觉、自然语言处理等等），深度学习的效果都远超过传统的机器学习方法，再加上媒体对深度学习进行了大肆夸大的报道，有人甚至认为，“深度学习最终可能会淘汰掉其它所有机器学习算法”。这种看法是正确的吗？
深度学习，作为目前最热的机器学习方法，但并不意味着是机器学习的终点，目前存在以下问题：
(1) 深度学习模型需要大量的训练数据，才能展现出神奇的效果，但现实生活中往往会遇到小样本问题，此时深度学习方法无法入手，传统的机器学习方法就可以处理。
(2) 有些领域，采用传统简单的机器学习方法，可以很好地解决，没必要采用复杂的深度学习方法。
(3) 深度学习的思想，来源于人脑的启发，但绝不是人脑的模拟，举个例子：给一个三四岁的小孩看一辆自行车之后，再见到哪怕外观完全不同的自行车，小孩也十有八九能做出那是一辆自行车的判断，也就是说，人类的学习过程往往不需要大规模的训练数据，目前深度学习方法显然很难做到。
(4) 目前在工业界，基于深度学习的人工智能项目落地非常困难。



5.2 深度学习 VS 强化学习：深度学习和强化学习的主要区别在于：

(1) 深度学习的训练样本是有标签的，强化学习的训练是没有标签的，它是通过环境给出的奖惩来学习。

(2) 深度学习的学习过程是静态的，强化学习的学习过程是动态的；这里静态与动态的区别在于是否会与环境进行交互，深度学习是给什么样本就学什么，而强化学习是要和环境进行交互，再通过环境给出的奖惩来学习。

(3) 深度学习解决的更多是感知问题，强化学习解决的主要是决策问题；有监督学习更像是”五官”，而强化学习更像“大脑”。

5.3 机器学习 VS 深度学习 VS 强化学习：机器学习、深度学习、强化学习之间的关系如下：

(1) *机器学习*：一切通过优化方法挖掘数据中规律的学科，多用于数据挖掘、数据分析和预测等领域。

(2) *深度学习*：一切运用了神经网络作为参数结构进行优化的机器学习算法，广泛地应用于是计算机视觉和自然语言处理领域。

(3) *强化学习*：不仅能利用现有数据，还可以通过对环境的探索获得新数据，并利用新数据循环往复地更新迭代现有模型的机器学习算法；学习是为了更好地对环境进行探索，而探索是为了获取数据进行更好的学习；目前实际应用场景还比较窄，主要包括 AI 游戏（Atari），推荐系统（阿里巴巴），机器人控制（如吴恩达的无人机）。

(4) *深度强化学习*：一切运用了神经网络作为参数结构进行优化的强化学习算法（Google AlphaGo, Master)。

5.4 机器学习 VS 深度学习 VS 迁移学习：当前的机器学习、深度学习存在一些局限性，我们采用迁移学习的方法可以解决这些痛点。

(1) 我们可以在这个数据集上训练一个模型 A，并期望它在同一个任务和域 A 中的未知数据上表现良好；但是，当我们没有足够的来自于我们关心的任务或域的标签数据时（新的标签数据很难获取、费时、昂贵)，传统的监督学习方法会失灵——它往往无法得出一个可靠的模型。

(2) 表达能力的限制：因为一个模型毕竟是一种现实的反映，等于是现实的镜像，它能够描述现实的能力越强就越准确，而机器学习和深度学习都是用变量来描述世界的，它们的变量数是有限的。

(3) 模型复杂度高：随着模型复杂度的提高，其参数个数和所需的数据量也是惊人的。

在 NIPS 2016，吴恩达表示，“在继深度学习之后，迁移学习将引领下一波机器学习技术”。

5.5 总体关系：以上讨论的算法和人工智能的关系如下图所示，可以看出，提及的算法都是属于人工智能的范畴，它们相互交叉却不完全重合：机器学习是人工智能的算法基石，而其它算法都是机器学习的一个分支。

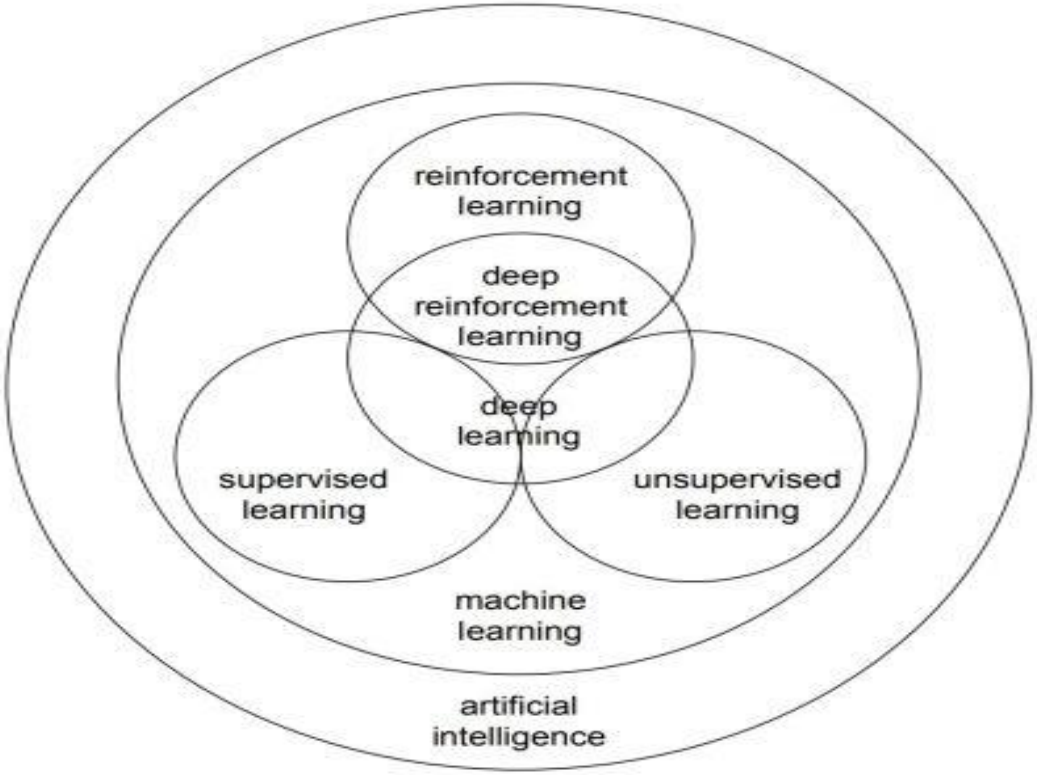


Figure 2: Relationship among deep reinforcement learning, deep learning, reinforcement learning, supervised learning, unsupervised learning, machine learning, and, artificial intelligence. Deep learning and deep reinforcement learning are addressing many classical AI problems.

深度学习大佬 Yoshua Bengio 在 Quora 上有一段话讲得特别好，这里引用一下：Science is NOT a battle, it is a collaboration. We all build on each other’ s ideas. Science is an act of love, not war. Love for the beauty in the world that surrounds us and love to share and build something together. That makes science a highly satisfying activity, emotionally speaking! 这段话的大致意思是：科学不是战争而是合作，任何学科的发展从来都不是一条路走到黑，而是同行之间互相学习、互相借鉴、博采众长、相得益彰，站在巨人的肩膀上不断前行。机器学习的研究也是一样，开放包容才是正道！