人工智能基础（A） CyanHaze

题型

> 判断20分（20题）
>
> 单选20分（20题）
>
> 多选30分（15题）
>
> 计算15分（3题）
>
> 问答15分（3题）

代码考试题型

1、代码注释题：

```
import tensorflow as tf
from tensorflow.keras.models import Sequential
model.fit(x_train, y_train, epochs=10, batch_size=32)
```

2、选择题：

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

3、问答题：输出是什么

比如：计算两个句子“天行健君子以自强不息”、“地势坤君子以厚德载物”的语义相似度

再比如：感知机模型的计算代码

第一章：初识人工智能

- 三大学派：
  - **符号主义**：
    - 又称**逻辑主义**。
    - 认为人类认知和思维的基本单元是符号。认知过程是建立在符号表示基础上的一种逻辑运算。
    - 优点：精准，严谨，可解释性和普适性。

      缺点：知识抽象难度大，知识更新复杂，容易产生语义分歧。
  - **联结主义**：
    - 又称**仿生学派**或**生理学派**。
    - 通过模拟人脑的生物神经网络来解释人的认知功能。
    - 提出了人工神经元、基于神经网络的**机器学习**和**深度学习**。
    - 优点：自动训练可预测，自适应变化，实现高维的学习。

      缺点：可解释性差，训练耗费算力，容易发生过拟合，鲁棒性难以提高，容易产生偏见。
    - 成功案例：人脸识别，机器翻译，图像分类，ChatGPT 等。
  - **行为主义**：
    - 强调模拟人的行为，与外界环境互动决策。
    - 以最大化奖励为目标。
    - 例子：强化学习。
- 发展浪潮
  - 三次浪潮两次低谷
  - 第一次浪潮：符号主义时期
  - 第二次浪潮：联结主义时期
  - 第三次浪潮

第二章：人工智能系统数据基础

- 人工智能三要素：**算法**、**数据**、**算力**
- 人工智能系统技术架构：
  - 基础设施层：系统设施、软件设施、数据设施
  - 智能技术层
  - 智能应用层
- 冯诺依曼体系架构：存储器、运算器、控制器、输入设备、输出设备

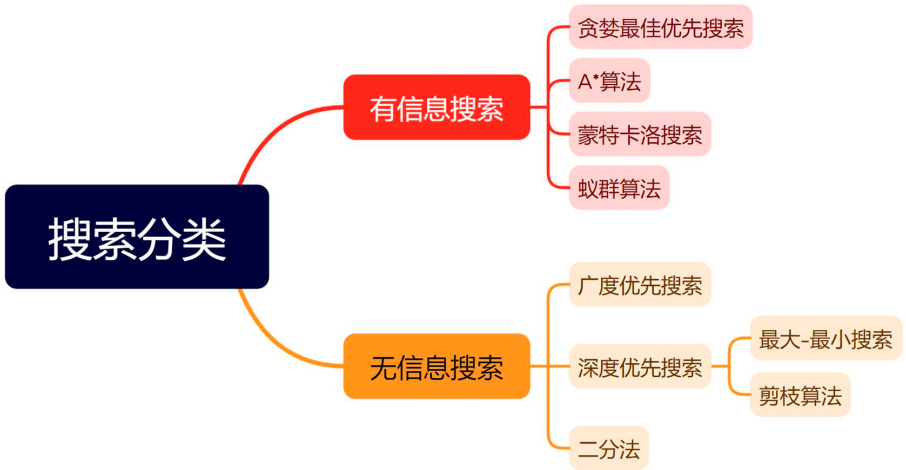
第三章：人工智能的应用开发基础

- Python部分（自学）
- 深度学习框架：PyTorch TensorFlow

第四章：从问题求解到机器学习

> 算法部分基础主要为上学期的C程，如果希望速成可参见：
>
> Week 0 Search - CS50's Introduction to Artificial Intelligence with Python
>
> Week 3 Algorithms - CS50x 2025

- 算法方法学：
  - 贪心法：小方案推广至大方案，选择局部最优解
  - 分治法：Divide and Conquer，多以递归形式呈现
  - 回溯法：也叫穷举搜索法，常用递归形式呈现
  - 动态规划法：以空间换时间，将大问题分解为子问题，子问题的解存放
- 搜索算法：
  - 四要素：
    - 初始状态（Initial State）
    - 目标状态（Goal State）
    - 操作（Operators）
    - 路径（Path）
  - 常见算法：



- 机器学习
  - 核心要素：
    - 数据：输入
    - 模型：输出
    - 算法：建模
  - 过程：
    - 训练：学习算法、训练算法
    - 预测
    - 评估：准确率、精确率、召回率、F1分数
  - 分类：
    - 有监督学习：X、Y都已知
      - 分类（Classification）：针对离散标签进行预测，如图像分类
      - 回归（Regression）：预测连续的值，如房价预测
    - 无监督学习：X已知、Y未知，发现数据的内在结构
      - 聚类：将相似数据分组，如客户细分
      - 降维：减少数据维度便于可视化，如主成分分析（PCA）
    - 半监督学习：部分X、Y已知，结合少量标注数据和大量未标注数据学习
      - 降低获取标注数据的成本
    - 强化学习&深度学习
  - 常见模型：
    - 有监督学习：线性回归、逻辑回归、决策树、随机森林、支持向量机（SVM）、Kl临近算法（K-NN）、神经网络（NN）
    - 无监督学习：K均值聚类（K-Means）、层次聚类、主成分分析（PCA）、异常检测
- 数据处理
  - 数据标准化和归一化：
    - min-max标准化（MinMaxScaler）：映射至$[0,1]$区间

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}}$$

    - Z-score标准化（StandardScaler）：使数据满足标准正态分布

$$x' = \frac{x - \bar{X}}{S}$$

    - 归一化（Normalizer）：L2归一化

$$x' = \frac{x}{\sqrt{\Sigma_j^m x_j^2}}$$

  - 数据二值化：将数据转化为布尔值，Binarizer
  - 编码：
    - 标签编码：将不连续的数值或文本变量转化成有序数值变量
    - 独热编码：将可能值转化成二值化特征

第五章：回归与分类模型

> 具体回归与分类的算法概述不了，请翻阅课本或者相关网课

回归模型

- 整体工作流程
    1. 收集数据
    2. 分析数据
    3. 选择回归类型种类
    4. 训练模型
    5. 评估模型
    6. 使用模型预测
    7. 持续改进
- 回归模型分类
    - 按自变量数量：简单回归、多元回归
    - 按因变量数量：单输出回归、多输出回归
    - 按关系性质：线性回归、非线性回归

分类模型

- 分类模型分类
    - 按类别数量：
        - 二分类问题：逻辑回归、决策树、支持向量机
        - 多分类问题：决策树、随机森林、多类支持向量机
    - 按标签数量：单标签分类、多标签分类
    - 按类别关系：平坦分类、层次分类
- 线性分类器：定义超平面作为决策边界

$$f(x) = w^t x + \beta$$
$$w：权重向量weight$$
$$b：偏置项bias$$
$$x：输入特征向量$$

- 逻辑回归模型：使用$sigmoid$函数将线性函数输出转化为0到1之间概率值

$$Sigmoid：\ \sigma(z) = \frac{1}{1+e^{-z}}$$
$$z = w^t x + b$$

- 决策树分类模型：
    - 构建流程：
        1. 特征选择：选择输入特征分割来进行节点分割数据
        2. 决策树生成：递归过程
        3. 决策树剪枝：解决过拟合问题

第六章：数据的聚类和降维问题

- 数据相似度和距离度量
    - 欧式距离：处理图像分割
    - 曼哈顿距离：对异常值不太敏感
    - 余弦相似度：多用于计算高维稀疏的文本数据

聚类分析技术

- K-means算法
  1. 初始化：选择k个初始簇中心：$\mu_1, \mu 2, \ldots, \mu_k$
  2. 分配过程：计算每一个数据点x_i到每个簇中心的距离

$$d(x_i, \mu_k) = \sqrt{\Sigma_t^{dim}(x_{i,t} - \mu_{k,t})^2}$$

     每朵花被分配至离其最近的簇中心对应的簇
  3. 更新过程：重新计算每个簇的中心，新的簇中心是簇中所有点的算术平均值

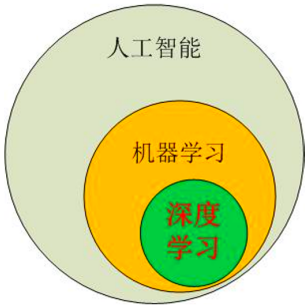
$$\mu_k = \frac{\Sigma_{i=1}^n w_{i,k} x_i}{\Sigma_{i=1}^n w_{i,k}}$$

数据降维技术

- 主成分分析算法（PCA）：寻找数据方差最大的正交方向

第七章：深度网络基础组件

> BP算法、梯度下降法和感知机模型 请自行查阅书本（P152起）和相关网课
>
> 推荐李宏毅机器学习与吴恩达深度学习与机器学习（虽然感觉补天不太可能）



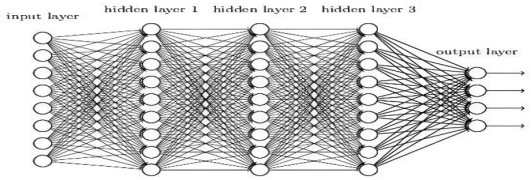
主要模型

1、多层感知机（MLP）
2、卷积神经网络
3、循环神经网络
4、Transformer
5、扩散模型

深度学习（Deep Learning，DL）是机器学习的一个子集，它使用多层人工神经网络来精准完成诸如物体检测、语音识别、语言翻译等任务。其基本思想是通过构建多层网络，对目标进行多层表示，以期通过多层的高层次特征来表示数据的抽象语义信息，从而获得更好的特征鲁棒性。

特征

多层网络结构：由多层神经元组成，包括输入层、隐藏层和输出层。这些层之间通过权重和偏置进行连接，形成复杂的网络结构。



自动特征提取：自动从原始数据中提取有用的特征，无需人工设计特征工程。
非线性激活函数：深度学习模型中通常使用非线性激活函数（如ReLU、sigmoid等），使得模型能够学习复杂的非线性关系。
大规模数据处理能力：借助现代计算技术和大数据资源，深度学习模型能够处理大规模数据集，提高模型的准确性和泛化能力。

- 深度学习基础算法：
  - 拓扑结构：
    - 多个并行的感知机组成网络层
    - 标准深度神经网络由输入层、隐含层和输出层组成
  - 优化器：
    - 梯度下降法是BP算法的核心算法
    - 最初梯度下降法的缺陷有：固定学习率、梯度消失、梯度爆炸、局部最优
  - 预训练-微调
- BP算法：

BP算法的特点

（1）自适应、自主学习：BP算法能够根据预设的参数更新规则，不断地调整神经网络中的参数，以达到最符合期望的输出。
（2）较强的非线性映射能力：由于神经网络中的激活函数都是非线性的，因此BP算法能够处理复杂的非线性问题。
（3）严谨的推导过程：误差的反向传播过程采用的是已经非常成熟的链式法则，其推导过程严谨且科学。
（4）较强的泛化能力：在训练结束后，BP算法可以利用从训练数据中学到的知识解决新的问题。

BP算法的局限性

（1）易陷入局部最小值：由于BP算法采用的是梯度下降法，容易陷入局部极小值而得不到全局最优解。
（2）收敛速度慢：由于神经网络中的参数众多，每次迭代都需要更新大量的权值和阈值，导致收敛速度较慢。
（3）隐节点选取缺乏理论指导：传统方法需要不断地设置隐含层节点数进行试凑，以确定最优的隐含层结构。
（4）学习新样本时可能遗忘旧样本：在训练过程中，学习新样本时可能会遗忘之前已经学习过的旧样本。

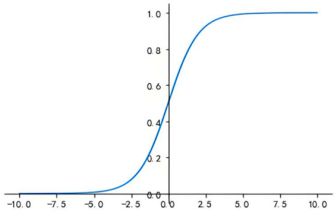
激活函数

- Sigmoid

$$f(x) = \frac{1}{1+e^{-x}}$$

Sigmoid

$f(x) = \frac{1}{1+e^{-x}}$



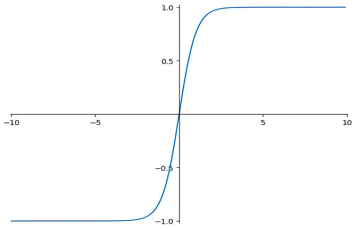
特性：
1）非线性
2）在-3与3之间，优化比较明显
3）在-3与3之外，优化不明显
4）值域在0~1之间，是非对称算法，这意味着下一个神经元只能接受正值的输入

- Tanh

$$f(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$$

Tanh

$f(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$



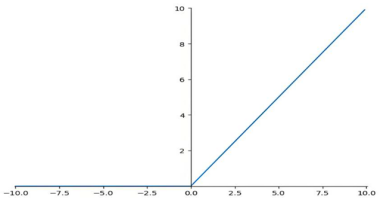
特性：
1）非线性
2）在-3与3之间，优化比较明显
3）在-3与3之外，优化不明显
4）值域在-1~1之间，是对称算法，解决了Sigmoid的局限
5）梯度比Sigmoid陡峭，易形成梯度崩塌

- ReLU

$$f(x) = max(0, x)$$

ReLU

$f(x) = max(0, x)$



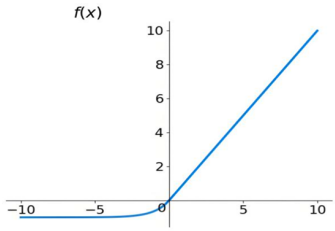
特性：
1）非线性
2）不会同时激活所有神经元
3）计算速度快
4）有趋于0的梯度
ReLU是目前隐含层中最为常用的损失函数

- ELU

$$f_i(a_i) = \begin{cases} x & if\ x > 0 \\ \alpha(e^{x}-1) & if\ x \le 0 \end{cases}$$

ELU

$$f_i(\alpha_i)\begin{cases} x & \text{if } x > 0 \\ \alpha(e^x - 1) & \text{if } x \le 0 \end{cases}$$



特性：
1）非线性
2）计算速度快（特指收敛速度快）
3）参数可调，可控制负数部分的表现
4）与ReLU相比，没有死亡神经元
缺点：负数部分计算较慢

- Softmax

$$S_i = \frac{e^{x_i}}{\Sigma_j e^{x_j}}$$

Softmax

$$S_i = \frac{e^{x_i}}{\sum_j e^{x_j}}$$

Softmax是将一个包含任意实数的K维向量压缩（或称为“归一化”）成另一个K维的实数向量，其中每一个元素的范围都在(0, 1)之间，并且所有元素的和为1。因此，Softmax的输出可以被解释为概率分布，其中每个概率对应着输入向量中对应元素属于某个类别的概率。
用途：应用最为广泛的一个激活函数分类器，一般用于最后一层，输出分类的概率结果，实现神经网络的分类，如：手写数字识别、物体分类、蛋白质结构分类、气象预测等应用

损失函数

> 因为个人习惯问题，此处$\hat{Y}$代表真实值

回归损失函数

- 均方误差（mean_squared_error,MSE）

$$MSE = \frac{1}{N}\Sigma_{i=1}^{N}(Y_i - \hat{Y}_i)^2$$

- 平均绝对误差（mean_absolute_error,MAE）

$$MAE = \frac{1}{N}\Sigma_{i=1}^{N}|Y_i - \hat{Y}_i|$$

- 平均绝对百分比误差（mean_absolute_percentage_error,MAPE）

$$MAPE = \frac{1}{N}\Sigma_{i=1}^{N}|\frac{Y_i - \hat{Y}_i}{\hat{Y}_i} * 100|$$

- 均方对数误差（mean_squared_logarithmic_error,MSLE）

$$MSLE = \frac{1}{N}\Sigma_{i=1}^{N}[log(\hat{Y}_i + 1) - log(Y_i + 1)]^2$$

⚠️：此处教材有误，教材的均方根对数误差为RMSLE，为$\sqrt{MSLE}$

分类损失函数

- 二进制交叉熵（binary_crossentropy）

$$Loss = -\frac{1}{N}\Sigma_{i=1}^{N}[\hat{Y}_i * logY_i + (1 - \hat{Y}_i) * log(1 - Y_i)]$$

⚠️：此处教材有误

- 多分类交叉熵（categorical_crossentropy）

$$Loss = -\frac{1}{N}\Sigma_{i=1}^{N}\hat{Y}_i * logY_i$$

优化器

- 随机梯度下降法（SGD）

随机梯度下降法　Stochastic Gradient Descent, SGD
一次计算全部样本的误差之和→一次只计算一个（批）样本

优点：1 高效
　　2 可并行计算
　　3 可适应新数据变化（预训练思想萌芽）
　　4 有机会全局最优
局限：1不稳定
　　2 没有解决学习率选择问题
　　3 随机最优解
　　4 模型不可控

改进：**小批量梯度下降法（Mini-batch SGD）**
　　动量梯度下降法（Momentum SGD）

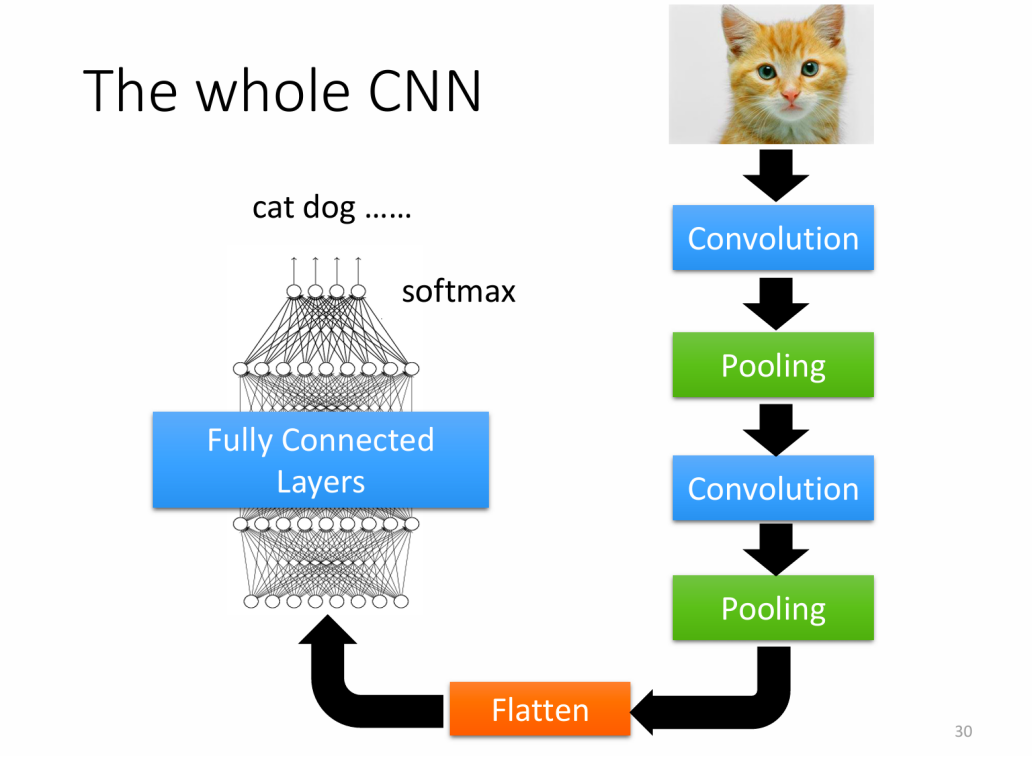
- 自适应梯度算法（AdaGrad）

自适应梯度算法 Adaptive Gradient,，AdaGrad
根据权重的梯度自适应调整学习率

优点：1 自动化调整学习率
　　2 自适应，所有权重“步调一致”
局限：1 梯度消失（训练次数作为分母）
　　2 训练速度慢

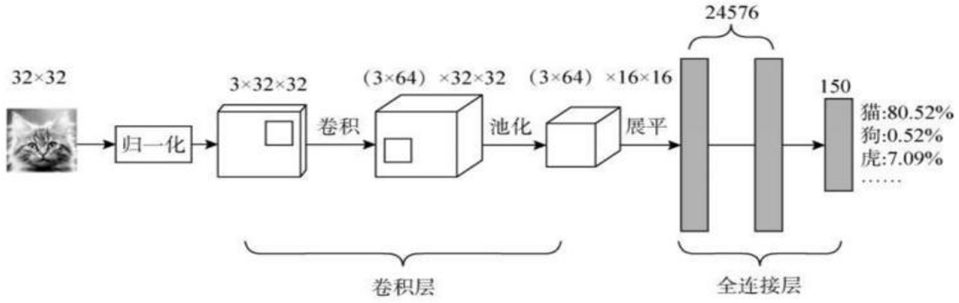
改进：**自适应平方根梯度法（RMSProp）**
　　自适应矩估计法（Adam）　目前最常用

第八章：卷积神经网络（CNN）



一个最典型的卷积神经网络结构是由卷积层、全连接层和输出层组成的前馈神经网络。其中卷积层又包含卷积运算、池化运算和归一化。这种卷积层可以多个堆叠串联。

图8-6是最简单的卷积神经网络：图像经过输入层，数据经过归一化、一次卷积（卷积核=64个）、一次池化（池化核=2×2，步长=2）、MLP全连接层、输出层后，获得150个分类结果。复杂一点的卷积神经网络可以有多个卷积和池化层，如AlexNet，如图8-7所示。



三要件：卷积、池化、展平

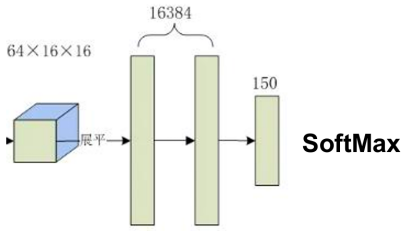
池化（pooling）：池化也称下采样，其作用就是缩小特征图的尺寸，减少计算量。池化的原理是可以用某一图像区域子块的统计信息包含了该子块全局信息。池化操作主要有最大池化、平均池化、随机池化、L2范数池化、K-max池化和全局平均池化等。CNN常用2×2区域进行池化。

例9-2：对如下的特征图进行平均池化计算，池化窗口2×2，

最大池化	池化结果	随机池化	池化结果
125 115 137 120	137	12 11 17 10	Radom(12,11,17,10)
平均池化	**池化结果**	**L2范数池化**	**池化结果**
33 46 57 40	44	19 15 26 23	$\sqrt{19^2+15^2+26^2+23^2}$

23	34	55	32	66	43
15	43	27	30	39	54
33	67	47	28	36	49
56	89	90	35	77	65
54	32	37	48	65	32
63	67	38	43	29	54

A1=(23+34+15+43)/ 4=28.75	A2=(55+32+27+30)/ 4=36	A3=(66+43+39+54)/ 4=50.5
A4=(33+67+56+89)/ 4=61.25	A5=(47+28+90+35)/ 4=50	A6=(36+49+77+65)/ 4=56.75
A7=(54+32+63+67)/ 4=54	A8=(37+48+38+43)/ 4=41.5	A9=(65+32+29+54)/ 4=45



MLP 分类

1	2	3
4	5	6
7	8	9

1 2 3 4 5 6 7 8 9

展平：用于将多维的数据降为一维，构建MLP，完成最后的分类任务。

独热码（One-Hot code）：在分类问题中，我们不能将不同的分类用1,2,3，……这种有序的数字来表示，只能通过**独热码**这样的向量矩阵来进行编码，供计算机识别计算，并用所在索引位置的概率值来预测最终的分类结果

	猫	狗	猪	牛	羊
猫	1	0	0	0	0
狗	0	1	0	0	0

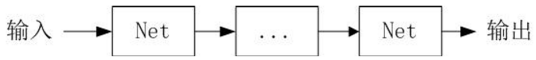
独热码-编码

	猫	狗	猪	牛	羊	求和
猫	0.76	0.18	0.03	0.02	0.01	1
狗	0.05	0.92	0.01	0.01	0.01	1

前馈网络运算结果

第九章：循环神经网络（RNN）

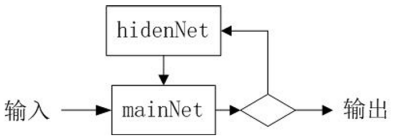
前馈神经网络



多层感知机 MLP

卷积神经网络 CNN

反馈神经网络



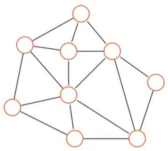
循环神经网络 RNN

长短程记忆网络 LSTM

Hopfield网络

波尔兹曼机

图网络

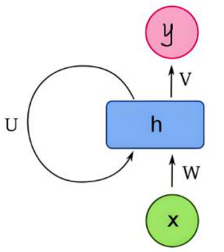


知识图谱

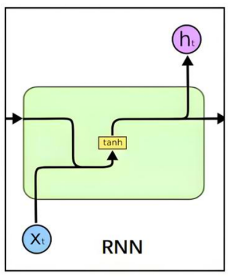
社交网络

城市交通

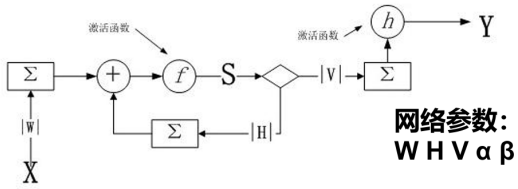
循环神经网络（Recurrent Neural Network，RNN）：是一种特殊类型的反馈神经网络，专门用于处理序列数据。RNN的核心思想在于其循环结构，使得网络能够捕捉和利用序列中的顺序依赖性信息。RNN的基本单元是一个具有循环连接的神经网络层。这个循环连接允许网络在处理每个时间步的数据时，能够利用之前时间步的信息。具体来说，RNN在每个时间步都会接收一个输入，并产生一个输出，同时其内部状态（隐藏状态）会被更新并传递到下一个时间步。



表示法一



表示法二



网络参数：
W H V α β

t时刻的输出：

$$Y_t = h(V \bullet S_t + \alpha)$$

$$S_t = f(W \bullet X_t + H \bullet S_{t-1} + \beta)$$

表示法三

> RNN和CNN不仔细记录了，这种感觉懂的人本来就懂，不懂的人也看不懂（）
>
> 如果原先没学过，我觉得这块复习一个CNN的卷积、池化和展平就好了，具体细节不用深究。

第十章：完整的人工智能应用开发实践

模型评估与选择

预测正、实际正：TP 真正例

预测负、实际正：FN 假反例

预测正、实际负：FP 假正例

预测负、实际负：TN 真反例

- 准确率（Accuracy）

$$准确率 = \frac{预测正确的样本数}{样本总数} * 100\%$$

- 精确率（Precision）

$$精确率 = \frac{TP}{TP + FP} * 100\%$$

- 召回率（Recall）

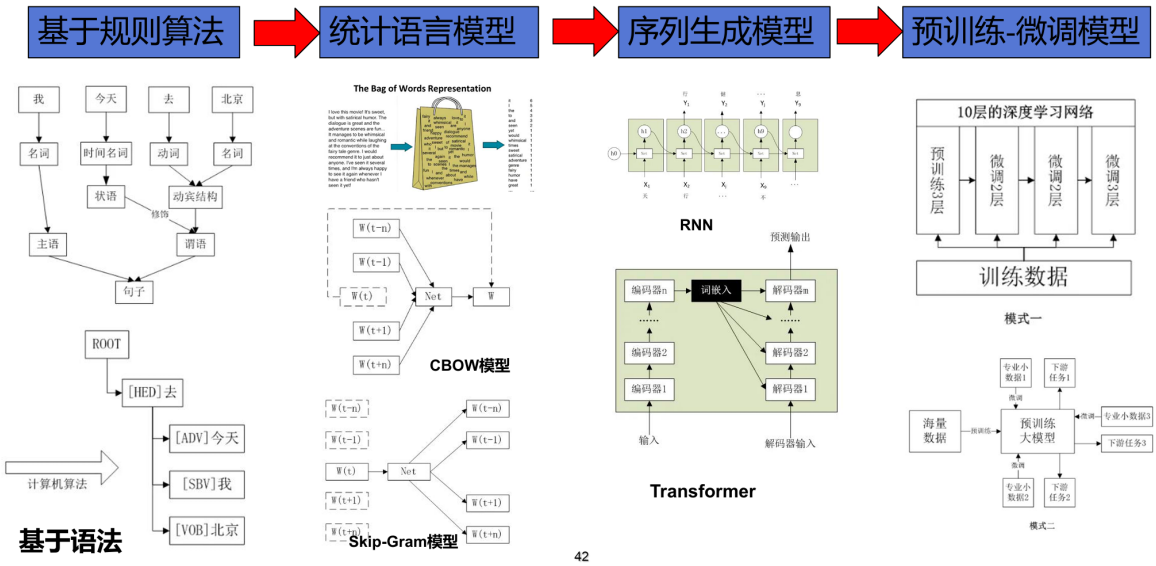
$$召回率 = \frac{TP}{TP + FN} * 100\%$$

- F1值

$$F1 = 2 * \frac{精确率 * 召回率}{精确率 + 召回率}$$

- ROC曲线：面积越大，性能越好

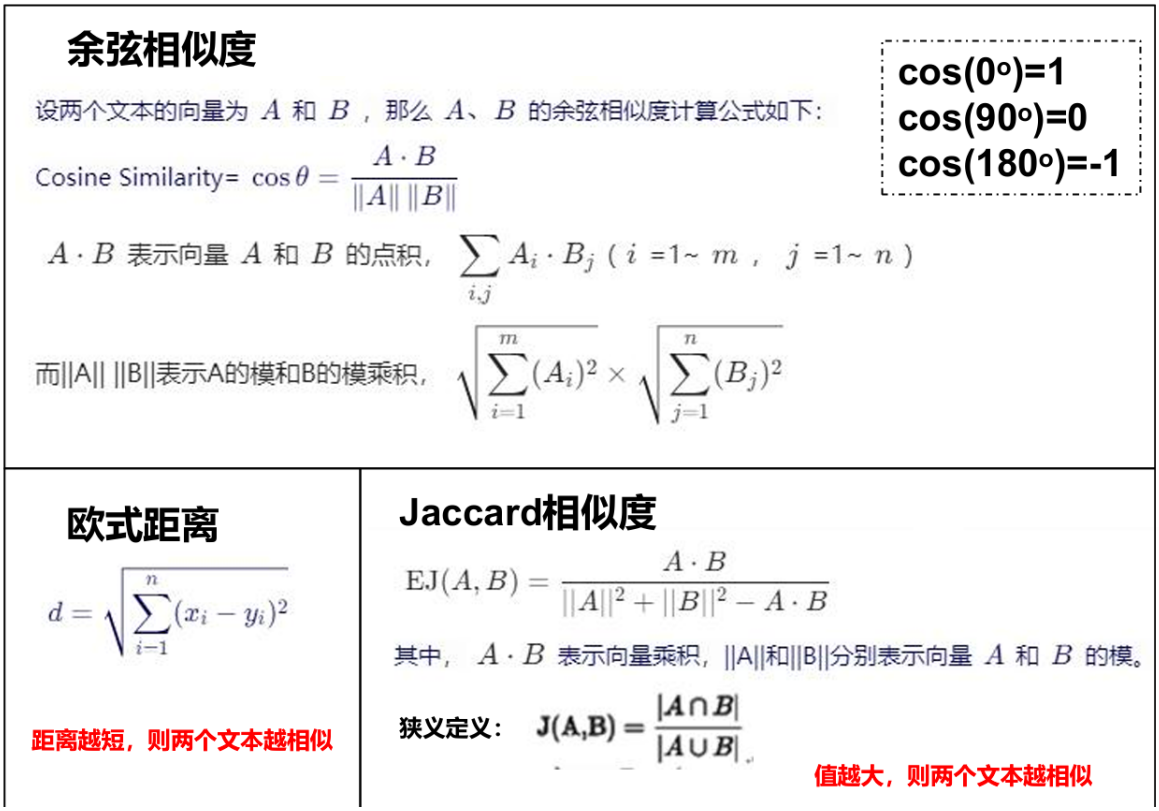
第十一章：自然语言处理中的模型



Token：令牌，可以是一个字，也可以是一个词，或者是一个字母，甚至是一个字节，要看具体的情况。本质上，一个“Token”就是通过分词技术（工具）将一句话分割成的最小单位，是一个特定的自然语言处理模型能处理的最基本元素，至小有内，至大无外。

“黄山落叶松叶落山黄”

- **按字分** “黄-山-落-叶-松-叶-落-山-黄” 词汇表5个token，句子9个token
- **按词分** “黄山-落叶松-叶-落-山-黄” 词汇表6个token，句子6个token



文本相似度

- 余弦相似度

$$CosineSimilarity(\vec{A},\vec{B}) = \frac{\vec{A}\cdot\vec{B}}{\|\vec{A}\|\|\vec{B}\|}$$

- 欧氏距离

$$d = \sqrt{\Sigma_{i=1}^{n}(x_i - y_i)^2}$$

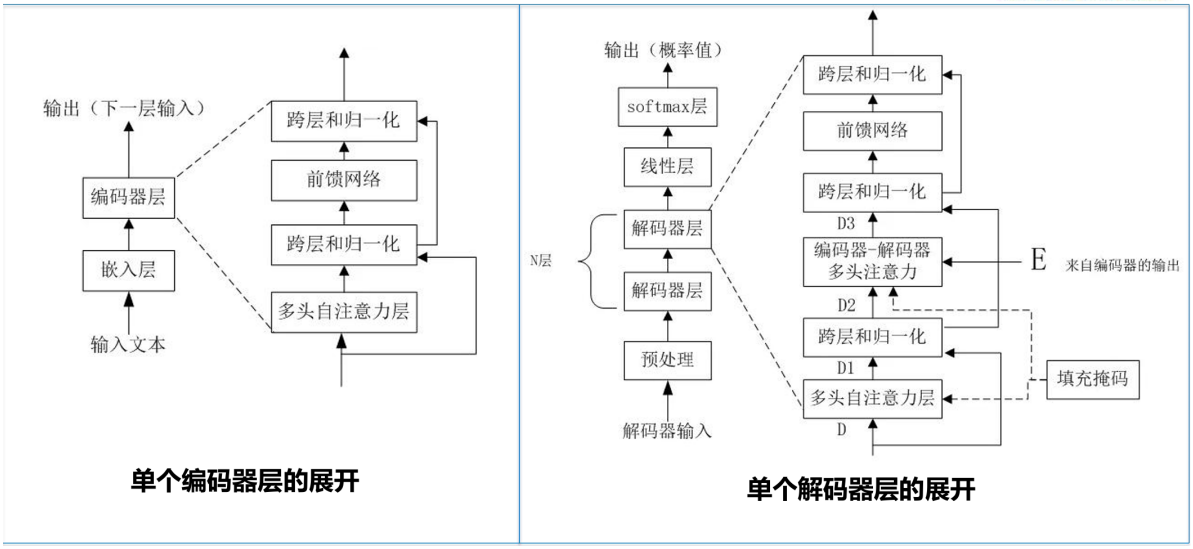
- Jaccard相似度

$$JaccardSimilarity(\vec{A},\vec{B}) = \frac{\vec{A}\cap\vec{B}}{\vec{A}\cup\vec{B}} = \frac{\vec{A}\cdot\vec{B}}{\vec{A}\cdot\vec{A}+\vec{B}\cdot\vec{B}-\vec{A}\cdot\vec{B}}$$

- 曼哈顿距离

$$d = \sqrt{\Sigma_{i=1}^{n}|x_i - y_i|}$$

Transformer



单个编码器层的展开

单个解码器层的展开