Plan

基础：

1. Python
2. Matlab
3. C++

数学：

**高数/数分**

**概率论与数理统计**

**线性代数**

专门针对深度学习还有机器学习

Python

1. numpy 向量，矩阵数学库
2. pandas 处理数据
3. matplotlib 画图

Algorithm and Data Structure

Data Structure:

1. **数组**：固定大小的元素集合。
2. **链表**：元素集合，每个元素指向下一个元素。
3. **栈**：后进先出（LIFO）的数据结构。
4. **队列**：先进先出（FIFO）的数据结构。
5. **哈希表**：通过键值对存储数据，提供快速访问。
6. **树**：层次结构的数据结构，每个节点有零个或多个子节点。
7. **图**：由节点和连接节点的边组成的数据结构。

Algorithm:

1. **排序算法**：
   * 冒泡排序
   * 选择排序
   * 插入排序
   * 快速排序
   * 归并排序
   * 堆排序
2. **搜索算法**：
   * 线性搜索
   * 二分搜索
3. **递归**：算法自我调用解决问题的子问题。
4. **动态规划**：通过将问题分解为子问题来解决复杂问题。
5. **贪心算法**：在每一步选择中都采取在当前状态下最好或最优的选择。

**高级算法**

1. **图算法**：
   * 深度优先搜索（DFS）
   * 广度优先搜索（BFS）
   * Dijkstra算法
   * A\*搜索算法
   * 最短路径问题
   * 最小生成树（MST）
2. **动态规划高级应用**：
   * 背包问题
   * 序列对齐问题
   * 最长公共子序列（LCS）
3. **贪心算法高级应用**：
   * 霍夫曼编码
   * 图的最小生成树
4. **回溯算法**：通过试错来找到问题的解。
5. **分支限界法**：使用界限来剪枝，减少搜索树中节点的搜索。

**算法设计技巧**

1. **分治法**：将问题分解为更小的子问题，递归解决子问题，然后合并结果。
2. **贪心法**：在每一步选择当前看起来最优的选择。
3. **动态规划**：将问题分解为重叠子问题，通过记忆化或表格来避免重复计算。
4. **迭代方法**：使用迭代而非递归来解决问题。

Machine Learning

Machine Learning Framework:

1. Scikit-learn
2. XGBoost
3. LightGBM

Machine Learning Algorithm

**监督学习算法**

1. **线性回归** (Linear Regression)：用于预测连续数值，假设特征和输出变量之间存在线性关系 。
2. **逻辑回归** (Logistic Regression)：用于二分类问题，输出预测结果的概率 。
3. **支持向量机** (Support Vector Machine, SVM)：用于分类和回归问题，找到不同类别之间的最佳边界 。
4. **决策树** (Decision Tree)：用于分类和回归，通过树状图的结构进行决策 。
5. **随机森林** (Random Forest)：集成学习方法，使用多个决策树进行分类或回归 。
6. **K-最近邻** (K-Nearest Neighbors, KNN)：基于实例的学习，通过查找最近的邻居进行分类或回归 。
7. **朴素贝叶斯** (Naive Bayes)：基于贝叶斯定理，适用于文本分类和垃圾邮件检测 。

**无监督学习算法**

1. **K-均值聚类** (K-Means)：将数据点分成K个簇，使得簇内的方差尽可能小 。
2. **层次聚类** (Hierarchical Clustering)：构建一个多层次的嵌套簇树 。
3. **主成分分析** (PCA)：用于降维，通过正交变换将数据转换到新的坐标系

Deep Learning

1. Deep Learning Framework: Pytorch, Tensorflow, Keras

Deep Learning Algorithm

1. **前馈神经网络**：最常见的神经网络类型，信息只在一个方向上流动。
2. **卷积神经网络 (CNN)**：专门用于处理具有网格结构的数据，如图像。
3. **循环神经网络 (RNN)**：适合处理序列数据，如时间序列或自然语言。
4. **长短期记忆网络 (LSTM)**：一种特殊的RNN，能够学习长期依赖信息。
5. **生成对抗网络 (GAN)**：由两个网络组成，一个生成器和一个判别器，通过对抗过程学习生成数据。
6. **深度强化学习**：结合了深度学习和强化学习，用于决策和控制问题。

**核心算法**

1. **反向传播算法**：用于训练神经网络，通过计算损失函数关于网络参数的梯度来更新参数。
2. **梯度下降**：优化算法，用于最小化损失函数。
3. **dropout**：一种正则化技术，用于防止神经网络过拟合。
4. **批量归一化**：加速训练过程，提高模型的稳定性。
5. **激活函数**：如ReLU、sigmoid、tanh等，用于增加网络的非线性。

**优化策略**

1. **学习率衰减**：随着时间减少学习率，以细化模型的权重调整。
2. **早停法**：当验证集上的性能不再提升时停止训练，以避免过拟合。
3. **模型检查点**：在训练过程中保存模型的状态，以便可以从最近的最佳状态恢复。

**应用领域**

1. **计算机视觉**：图像识别、目标检测、图像分割。
2. **自然语言处理**：语言翻译、文本摘要、情感分析。
3. **语音识别**：将语音转换为文本。
4. **推荐系统**：基于用户行为和偏好的推荐算法。
5. **游戏和模拟**：如AlphaGo，用于玩围棋等策略游戏。

**深度学习框架**

1. **TensorFlow**：由Google开发的开源框架。
2. **PyTorch**：由Facebook开发的开源框架，注重动态计算图。
3. **Keras**：用户友好的高级神经网络API，可以运行在TensorFlow或Theano之上。
4. **Caffe**：广泛用于计算机视觉的深度学习框架。
5. **MXNet**：支持灵活和高效的模型训练的深度学习框架。

**重要论文**

1. **AlexNet**：2012年，开启了深度学习在图像识别领域的革命。
2. **VGGNet**：2014年，提出了深度网络结构的简单有效的模型。
3. **GoogLeNet**：2015年，引入了Inception模块，提高了网络的效率。
4. **ResNet**：2015年，引入了残差连接，允许训练非常深的网络。

学习平台:

书籍,网站,视频,在线刷题网站

Python:

[GeeksforGeeks | A computer science portal for geeks](https://www.geeksforgeeks.org/)

[W3Schools Online Web Tutorials](https://www.w3schools.com/)

[Learn Data Science and AI Online | DataCamp](https://www.datacamp.com/)

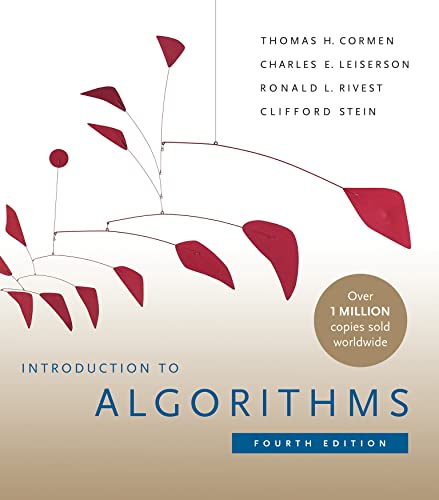
算法

[OI Wiki - OI Wiki (oi-wiki.org)](https://oi-wiki.org/)

算法列表

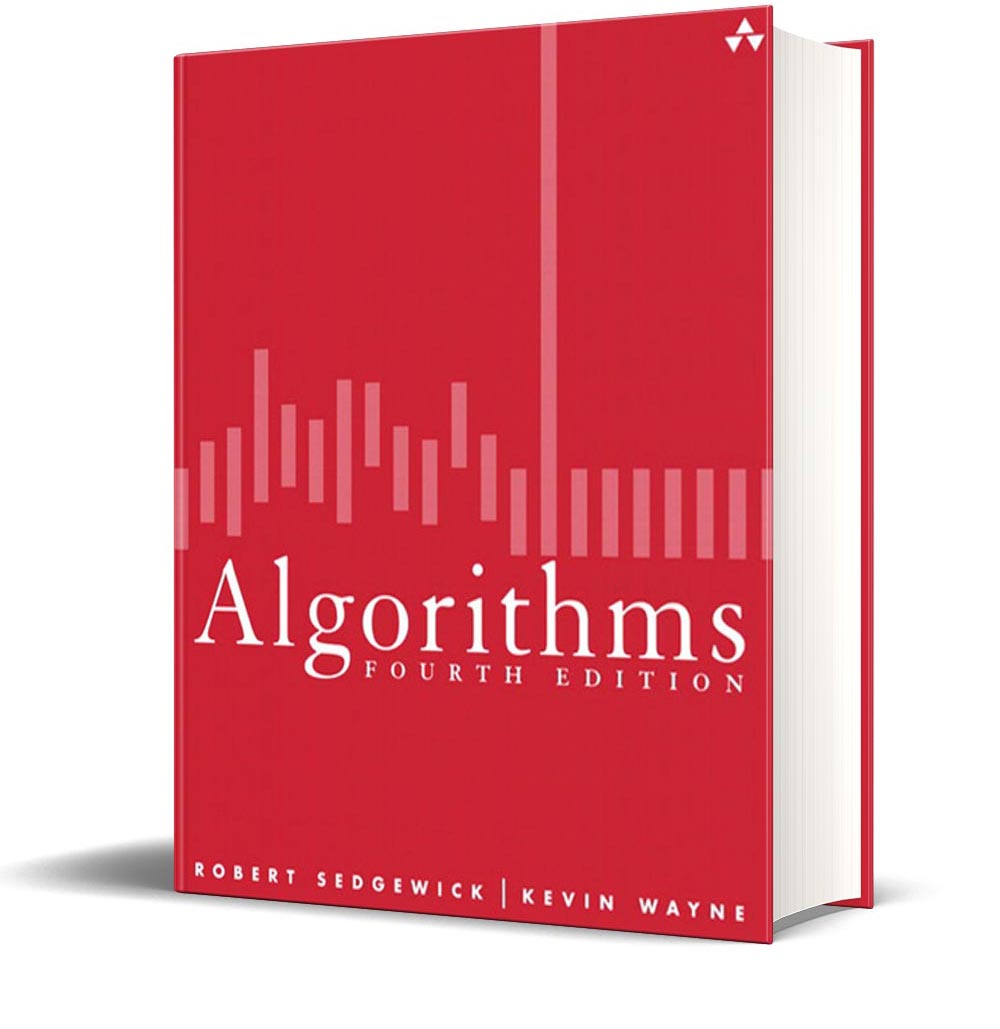
[List of algorithms - Wikipedia](https://en.wikipedia.org/wiki/List_of_algorithms)

书籍

[](https://libgen.gs/ads.php?md5=102effe6da23eb8e62cd121608acc73b)

**Title:** Introduction to Algorithms **Edition:** 4

**Author(s):** Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein

  
**Algorithms, Fourth Edition**

by Robert Sedgewick and Kevin Wayne

[Algorithms, 4th Edition by Robert Sedgewick and Kevin Wayne (princeton.edu)](https://algs4.cs.princeton.edu/home/)

Hello算法

[Hello 算法 (hello-algo.com)](https://www.hello-algo.com/)

**算法**

刷题:leetcode，codeforces CF

概念and 应用场景：看书，看视频，看网站

数学实现：看书，看视频，看网站，用手推一遍

代码实现：用至少一种lang写代码，把代码放到公共的地方Github

复杂度分析：手推

做题：5道