Assignment #1: 虚拟机，Shell & 大语言模型

Updated 1745 GMT+8 Sep 8, 2025

2025 fall, Complied by 林奕妃、环境科学与工程学院

作业的各项评分细则及对应的得分

标准	等级	得分
按时提交	完全按时提交：1分 提交有请假说明：0.5分 未提交：0分	1分
源码、耗时（可选）、解题思路（可选）	提交了4个或更多题目且包含所有必要信息：1分 提交了2个或以上题目但不足4个：0.5分 少于2个：0分	1分
AC代码截图	提交了4个或更多题目且包含所有必要信息：1分 提交了2个或以上题目但不足4个：0.5分 少于：0分	1分
清晰头像、PDF文件、MD/DOC附件	包含清晰的Canvas头像、PDF文件以及MD或DOC格式的附件：1分 缺少上述三项中的任意一项：0.5分 缺失两项或以上：0分	1分
学习总结和个人收获	提交了学习总结和个人收获：1分 未提交学习总结或内容不详：0分	1分
总得分： 5	总分满分：5分	

> **说明：**
>
> 1. **解题与记录：**
>
>   对于每一个题目，请提供其解题思路（可选），并附上使用Python或C++编写的源代码（确保已在OpenJudge， Codeforces，LeetCode等平台上获得Accepted）。请将这些信息连同显示“Accepted”的截图一起填写到下方的作业模板中。（推荐使用Typora https://typoraio.cn 进行编辑，当然你也可以选择Word。）无论题目是否已通过，请标明每个题目大致花费的时间。
>
> 2. **课程平台：**课程网站位于Canvas平台（https://pku.instructure.com ）。该平台将在第2周选课结束后正式启用。在平台启用前，请先完成作业并将作业妥善保存。待Canvas平台激活后，再上传你的作业。
>
> 3. **提交安排：**提交时，请首先上传PDF格式的文件，并将.md或.doc格式的文件作为附件上传至右侧的“作业评论”区。确保你的Canvas账户有一个清晰可见的本人头像，提交的文件为PDF格式，并且“作业评论”区包含上传的.md或.doc附件。

> 4. **延迟提交**：**如果你预计无法在截止日期前提交作业，请提前告知具体原因。这有助于我们了解情况并可能为你提供适当的延期或其他帮助。
>
> 请按照上述指导认真准备和提交作业，以保证顺利完成课程要求。

1. 题目

E27653: Fraction类（20分钟）

http://cs101.openjudge.cn/pctbook/E27653/

请练习用OOP方式实现。

思路：面向对象编程（OOP）的方法的四大特征是封装、抽象、多态和继承。针对这个问题采用多态封装的方法，将分子分母作为分数类的实例属性，通过初始化、化简、加法与输出行为实现题目要求。

代码：

```
import math

class Fraction:
    """
    表示分数的类，支持初始化时自动约分和加法运算。
    """
    def __init__(self, top, bottom):
        # 题目保证分母 > 0
        common = math.gcd(top, bottom)
        self.num = top // common
        self.den = bottom // common

    def __str__(self):
        """定义分数的字符串表示形式，如 '3/4'"""
        return f"{self.num}/{self.den}"

    def __add__(self, other_fraction):
        """重载 '+' 运算符，实现分数加法"""
        new_num = self.num * other_fraction.den + self.den * other_fraction.num
        new_den = self.den * other_fraction.den
        # 返回一个新的Fraction对象，其构造函数会自动完成约分
        return Fraction(new_num, new_den)

# --- 主程序 ---

# 读取一行中的四个整数
num1, den1, num2, den2 = map(int, input().split())

# 创建两个分数对象
f1 = Fraction(num1, den1)
f2 = Fraction(num2, den2)

# 计算和并打印结果
# f1 + f2 会调用 f1.__add__(f2)
# print() 会调用结果对象的 __str__() 方法
```

```
print(f1 + f2)
```

代码运行截图 （至少包含有"Accepted"）

M1760.袋子里最少数目的球（30分钟）

binary search, https://leetcode.cn/problems/minimum-limit-of-balls-in-a-bag/

思路：二分查找的难点在于边界条件。最小可能的开销是1个，最大可能的开销是初始时球最多的那个袋子。将求解题变成判断题，即能否在给定最大操作次数内使得所有袋子里的球都不超过k个，其中k由二分查找进行确定。

代码：

```
import math
from typing import List

class Solution:
    def minimumSize(self, nums: List[int], maxOperations: int) -> int:
        # 答案范围在 [1, max(nums)] 之间
        left, right = 1, max(nums)
        ans = right

        while left <= right:
            # mid 是我们猜测的答案，即最终每个袋子球数的最大值
            mid = (left + right) // 2

            # 计算要达成 mid 这个目标需要多少次操作
            # 对于每个球数为 num 的袋子，如果 num > mid，
            # 需要 (num-1)//mid 次操作才能把它拆分到每袋都不超过 mid
            ops_needed = sum([(num - 1) // mid for num in nums])

            if ops_needed <= maxOperations:
                # 操作数足够，说明 mid 是一个可行的目标
                # 记录答案，并尝试更小的目标
                ans = mid
                right = mid - 1
            else:
                # 操作数不够，说明 mid 这个目标太小，不可能实现
```

```
                # 必须尝试更大的目标
                left = mid + 1

        return ans
```

代码运行截图（至少包含有"Accepted"）

M04135: 月度开销（15分钟）

binary search, http://cs101.openjudge.cn/pctbook/M04135/

思路：这题与上一题相似，都是最小化最大值问题，将针对划分月度转化为判断题，即假设每个 fajo月的开销上限为k，我们能否将n天的开销划分成m个或更少的fajo月

代码：

```
import sys

def solve():
    """
    使用二分查找解决农夫约翰的预算问题。
    """
    try:
        # --- 1. 读取输入 ---
        # 使用 sys.stdin.readline() 在处理大量输入时比 input() 更快
        n_str, m_str = sys.stdin.readline().split()
        N = int(n_str)
        M = int(m_str)

        expenses = []
        for _ in range(N):
            expenses.append(int(sys.stdin.readline()))
```

```
    except (IOError, ValueError):
        # 处理可能的空输入或格式错误
        return

    # --- 2. 定义判定函数 check(k) ---
    def check(max_allowed_cost):
        """
        判定函数：在每个月开销不超过 max_allowed_cost 的情况下，
        最少需要多少个月。
        返回所需月份数是否 <= M。
        """
        months_needed = 1
        current_month_cost = 0

        for daily_cost in expenses:
            if current_month_cost + daily_cost <= max_allowed_cost:
                # 如果今天的开销可以加入当前月份
                current_month_cost += daily_cost
            else:
                # 如果不行，则结束当前月份，开启新月份
                months_needed += 1
                current_month_cost = daily_cost

        return months_needed <= M

    # --- 3. 设置二分查找的范围 ---
    # 下界：至少是单日最大开销
    # 上界：所有开销的总和
    left = max(expenses)
    right = sum(expenses)
    ans = right # 初始化答案为最大可能值

    # --- 4. 执行二分查找 ---
    while left <= right:
        mid = (left + right) // 2

        if check(mid):
            # 如果 mid 是一个可行的开销上限
            # 我们记录这个答案，并尝试寻找更小的上限
            ans = mid
            right = mid - 1
        else:
            # 如果 mid 这个上限太小了，不可行
            # 我们必须放宽预算，去右边区间寻找
            left = mid + 1

    # --- 5. 输出结果 ---
    print(ans)

# 调用主函数
solve()
```

代码运行截图 （至少包含有"Accepted"）

#50245985提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
import sys

def solve():
    """
    使用二分查找解决农夫约翰的预算问题。
    """
    try:
```

基本信息

#: 50245985
题目: M04135
提交人: Lin Yifei
内存: 8076kB
时间: 268ms
语言: Python3
提交时间: 2025-10-07 15:19:33

M27300: 模型整理（30分钟）

sortings, AI, http://cs101.openjudge.cn/pctbook/M27300/

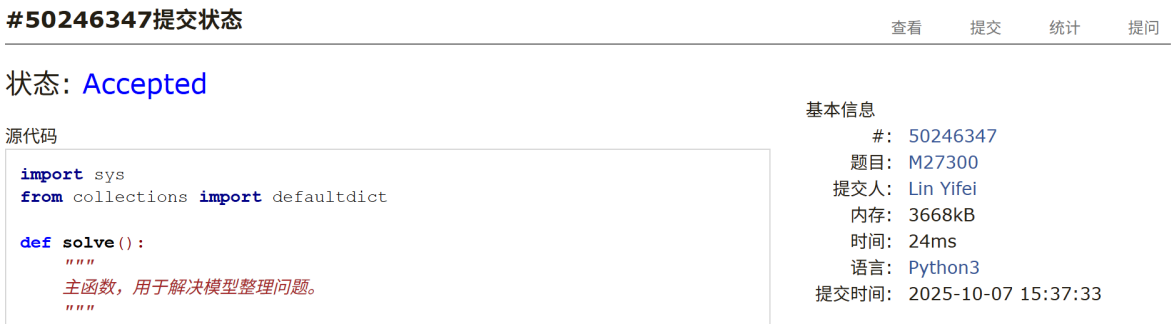
思路：这是字符串处理排序问题，需要将输入的数据进行解析、分组、排序，最后按照指定格式输出。

代码：

```
import sys
from collections import defaultdict

def solve():
    """
    主函数，用于解决模型整理问题。
    """
    try:
        n = int(sys.stdin.readline())
    except (ValueError, IndexError):
        # 处理空输入或格式错误
        return

    # 1. 使用 defaultdict(list) 创建一个字典。
    # 当访问一个不存在的键时，它会自动创建一个空列表作为值。
    models_data = defaultdict(list)

    for _ in range(n):
        line = sys.stdin.readline().strip()
        if not line:
            continue

        # 2. 解析每一行输入
        # 使用 rsplit('-', 1) 从右边第一个 '-' 分割，确保模型名正确
        try:
            model_name, param_str = line.rsplit('-', 1)
        except ValueError:
            # 如果行不包含'-'，跳过此行
            continue

        # 3. 转换参数量为统一的可比较数值 (单位：百万 M)
        unit = param_str[-1]
        number_part = float(param_str[:-1])
```

```
        # 统一转换为 M (百万) 为单位的数值
        comparable_value = 0.0
        if unit == 'M':
            comparable_value = number_part
        elif unit == 'B':
            comparable_value = number_part * 1000

        # 将 (用于排序的数值, 原始参数字符串) 元组存入字典
        models_data[model_name].append((comparable_value, param_str))

    # 4. 排序并输出
    # 首先，按模型名称的字典序排序
    sorted_model_names = sorted(models_data.keys())

    for name in sorted_model_names:
        # 获取该模型的所有参数信息
        params_list = models_data[name]

        # 对参数列表进行排序。Python的 sort 会默认按元组的第一个元素排序。
        params_list.sort()

        # 从排序后的元组列表中提取出原始的参数字符串
        # item[1] 就是原始的 param_str
        original_param_strings = [item[1] for item in params_list]

        # 格式化输出
        print(f"{name}: {', '.join(original_param_strings)}")

# 调用主解决函数
solve()
```

代码运行截图 （至少包含有"Accepted"）



#50246347提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
import sys
from collections import defaultdict

def solve():
    """
    主函数，用于解决模型整理问题。
    """
```

基本信息

#: 50246347
题目: M27300
提交人: Lin Yifei
内存: 3668kB
时间: 24ms
语言: Python3
提交时间: 2025-10-07 15:37:33

Q5. 熟悉云虚拟机Linux环境与大语言模型（LLM）本地部署（1小时）

本项目包括两个任务：

1）通过云虚拟机（如 https://clab.pku.edu.cn/ 提供的资源）熟悉Linux系统操作环境。

2）完成大语言模型（LLM）的本地部署与功能测试。

LLM 部署可选择使用图形化工具（如 LM Studio, https://lmstudio.ai）以简化配置流程，提升部署效率。部署完成后，需对模型进行实际能力测试。

测试内容包括：从主流在线编程评测平台（如 OpenJudge、Codeforces、LeetCode 或洛谷等）选取若干编程题目，提交由本地部署的 LLM 生成的代码解决方案，并确保其能够通过全部测试用例，获得“Accepted”状态。选题时应避免与已知可被 AI 正确解答的题目重复。当前已确认可通过的 AI 解题列表可参考以下 GitHub 仓库：

https://github.com/GMyhf/2025spring-cs201/blob/main/AI_accepted_locally.md

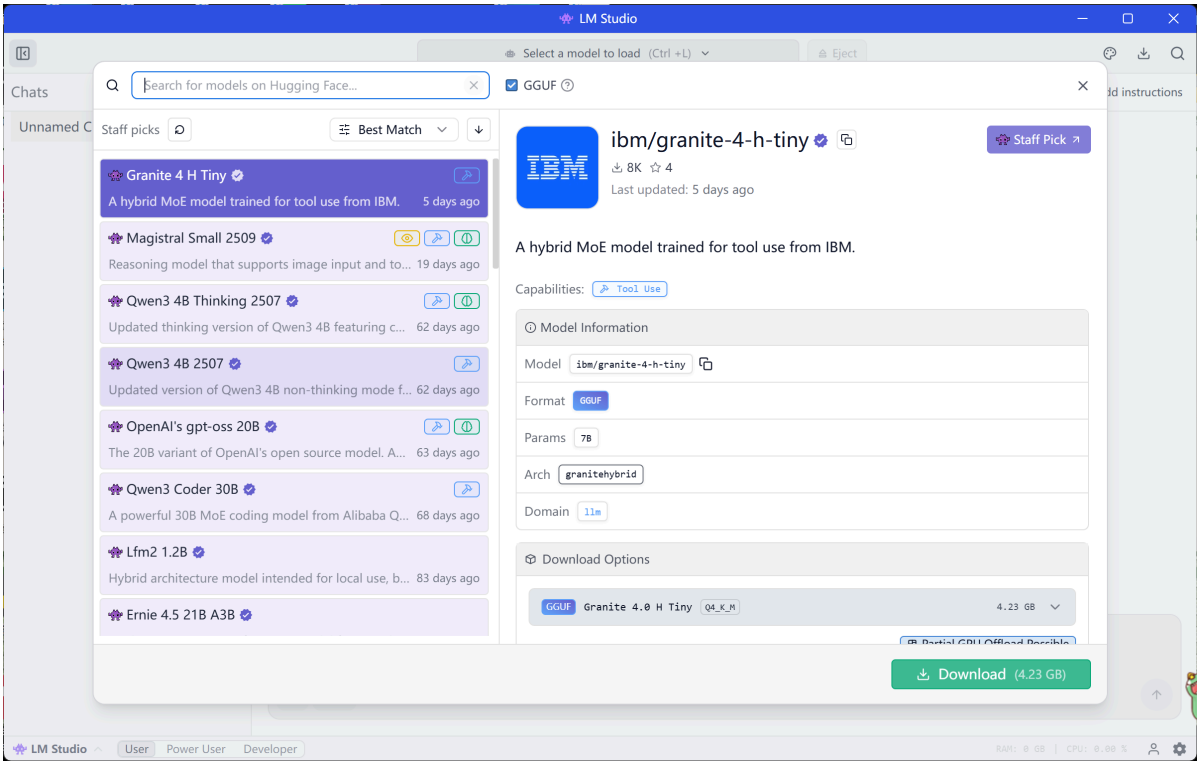
请提供你的项目进展，内容应该包括：关键操作步骤的截图以及遇到的技术问题及相应的解决方法。这将有助于全面掌握项目推进情况，并为后续优化与扩展提供依据。

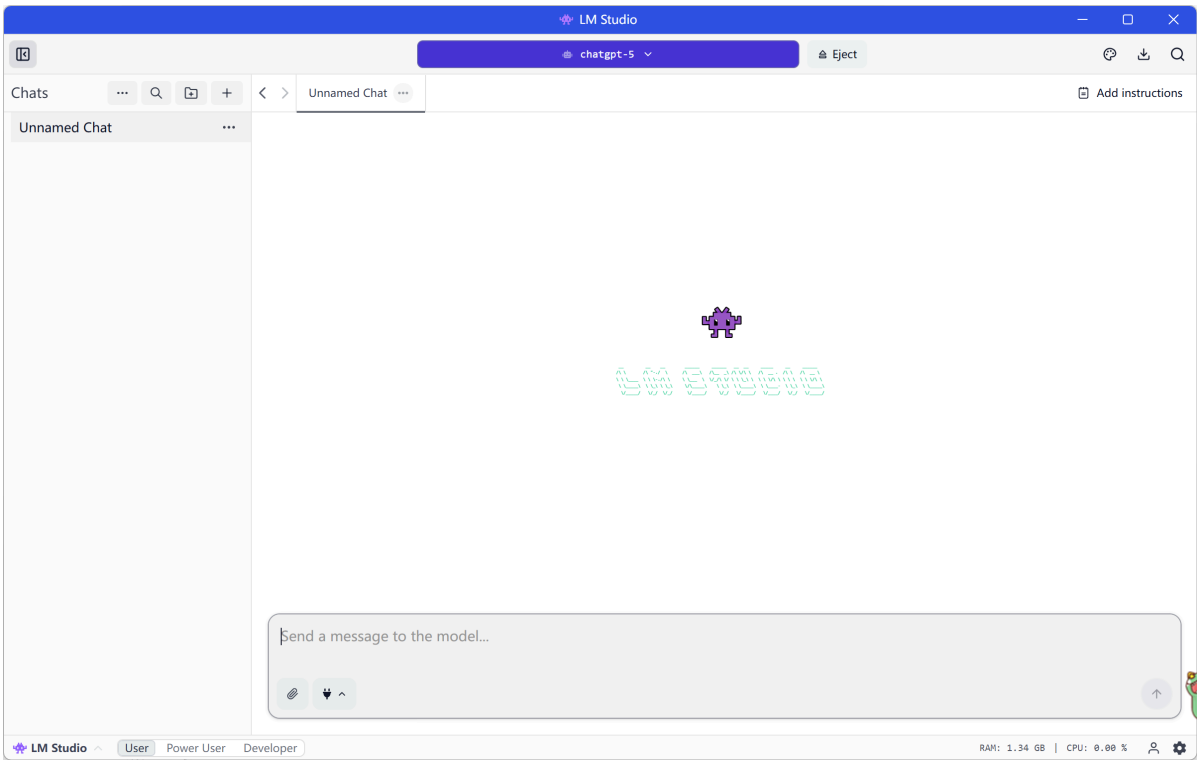
1 创建云主机

2 连接云主机

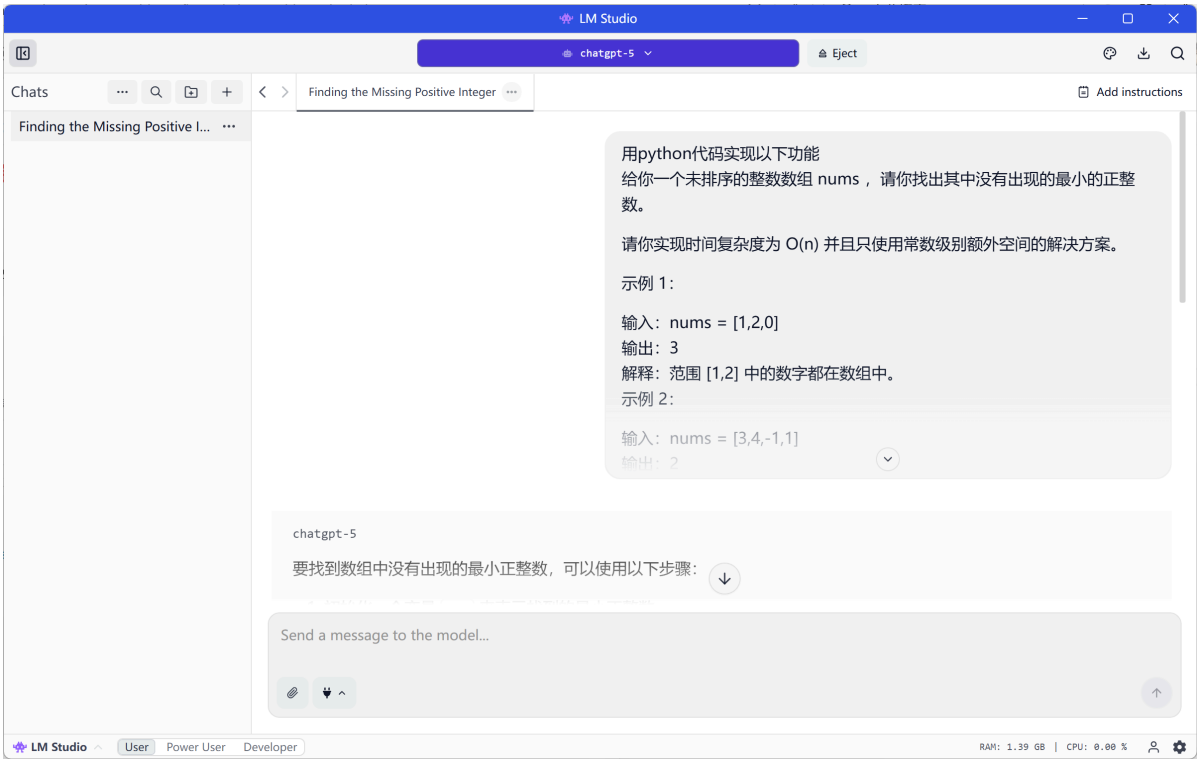
```
PS C:\Users\86188> ssh rocky@10.129.245.250
Last login: Mon Sep  8 13:43:19 2025 from 10.7.9.118
[rocky@linyifei ~]$
```

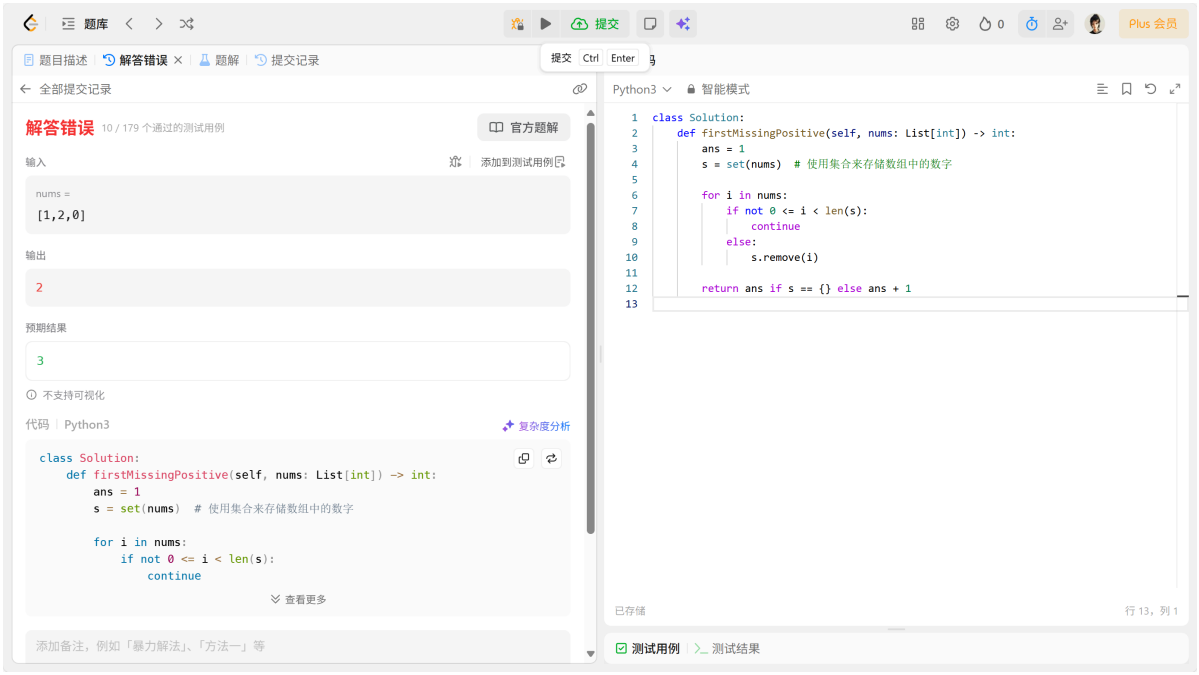
3 部署大语言模型到本地





功能测试初步测试错误，大语言模型提供的循环逻辑是错误的，且不满足时间复杂度要求





Q6. 阅读《Build a Large Language Model (From Scratch)》第一章（20分钟）

作者：Sebastian Raschka

请整理你的学习笔记。这应该包括但不限于对第一章核心概念的理解、关键术语的解析、学习过程中的思考与启发，以及尚存的疑问与反思。通过系统梳理，不仅有助于巩固自身理解，也希望为其他学习者提供有价值的参考。

书中这个图很好地帮我辨析了当下各种说法的区别

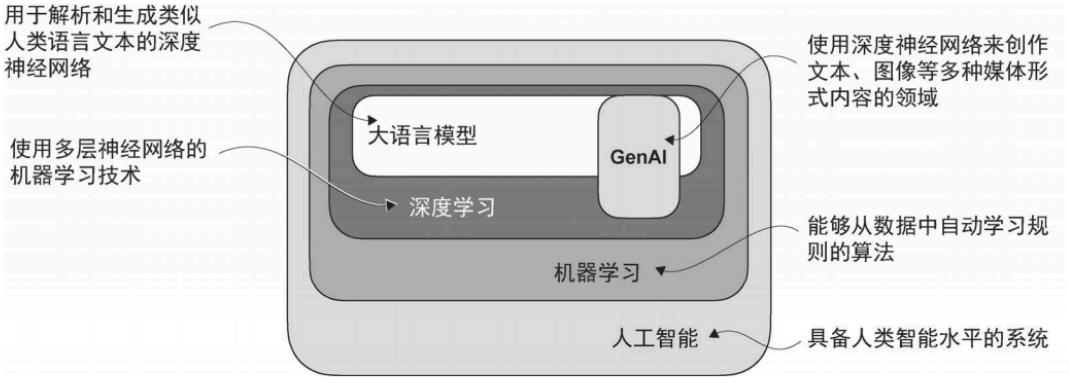


图 1-1 这一层级关系图展示了不同领域之间的关系。大语言模型是深度学习技术的具体应用，能够处理和生成类似人类语言的文本；深度学习是机器学习的一个分支，主要使用多层神经网络；机器学习和深度学习致力于开发算法，使计算机能够从数据中学习，并执行需要人类智能水平的任务

本书的框架

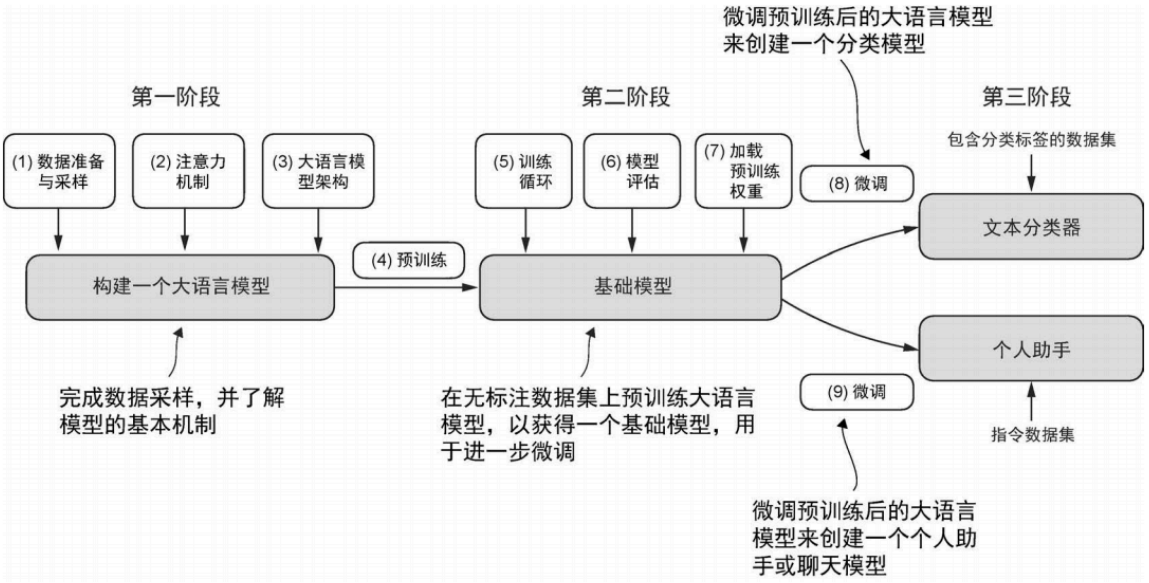


图 1-9　构建大语言模型的 3 个主要阶段：实现模型架构和准备数据（第一阶段）、预训练大语言模型以获得基础模型（第二阶段），以及微调基础模型以得到个人助手或文本分类器（第三阶段）

2. 学习总结和个人收获

距离大一上学期学计概B（C语言）已经过去了三年，我对于编程的基础知识已经有些生疏，且对于python语法不是很熟练，目前想要完成作业代码还是比较困难。针对本次作业我回顾了课堂知识并有针对性的根据老师提供的链接去学习了OOP和二分法，后续可能需要根据老师课上的建议去进行老师计概B课程的python训练。针对Linux系统操作环境，由于专业课需要此前有接触过，也在老师帮助下成功注册了云虚拟机。针对大语言模型，之前由于本研接触过u-net，但是只是针对代码的运用，并没有很好理解其背后的知识。很遗憾由于开学第一个月准备毕业论文开题与相关奖学金答辩与申请等各种事项并没有在课外额外进行训练，我认为这对我进行本课程学习是非常重要的。通过作业也对本课程需要使用的各个编程平台有了尝试，感觉非常方便。尽管本学期有26学分，但我还是会努力多花些时间在本课程的编程训练上。