

kingsley-yoimiya.github.io 转载请署名

软工实验班复现 | 基于 GPT 对于 AI 模型训练的探索

引言

这是软件工程实验班中对于“ChatGPT 能否真正替代人类 / 使得普通码农成为超级码农（无需相关行业知识，依靠 GPT 即可担任完成复杂、学科交叉性任务）”探索的第一部分。

探索内容

笔者感觉也是较难的一部分，主要探索 GPT 能否利用其内置知识对于 AI 模型进行训练，探索内容为对于 [kaggle](#) 中的题目，GPT 能否和人合作进行题意理解、题目分析、提出解决方案、编写代码、调试代码。

探索难点

- kaggle 题目复杂性：为了更好测试 GPT 能力，大家都选择了较难的、竞赛性的开放式问题，这些问题题面通常很复杂，数据很繁杂，代码非常冗长（具体情况见后文）。
- GPT 代码调试能力：尽管使用了目前最强大的 GPT 模型，但是 GPT 在代码调试时的能力有待验证。
- GPT 记忆能力：尽管 GPT4-turbo 支持 128k token^(注意实际上支持的文字长度要小一些)，但是题面文本近 10k，题目中 baseline^(起始代码，通常是主办方提供的初始模板) 近 30k，GPT 在此时记忆会明显吃力。
- 反馈时间：AI 方面的工作运行时间总是很长，而本次探索是在 kaggle notebook 上进行的，进而反馈有很长的延迟（训练 3h 平均，没有使用特别技巧缩短时间），不过本次探索重点并非调参，因此影响要小一点点了。

具体信息

- 使用模型：官网版 GPT4 / GPT-4-turbo-preview(OpenAI API)
- 使用者：北大大一信息与计算科学专业学生，仅对于 AI 方面有基本概念了解，无训练模型基础。
- 题目：[PlantTraits2024 - FGVC11 | Kaggle](#)
- 训练环境：Kaggle notebook (GPU: P100 / T4 * 2)
- 数据集：<https://www.kaggle.com/competitions/planttraits2024/data>

探索过程

任务初步理解分析

限于篇幅, 请参见之前的汇报 ppt, 仅给出对话结果。

PlantTraits2024 – FGVC11 任务理解汇报 (对话记录)

2300012929 尹锦润

任务描述

旨在通过众包的植物图片和附加数据预测六种植物特性 (如叶面积、植物高度)。

个人理解

- 在以前研究中, 已有模型 (ResNetV2, EfficientNetV2) 可以进行植物特征提取。
- 提供的图片和卫星数据、土壤数据、气候数据等。
- 根据相关模型, 特征提取, 综合其他信息。
- 评分方式: 六种特性 R2 值平均, 忽略负数 (R2值是一种衡量模型预测能力的统计指标, 其值范围可以从负无穷大到1)。

难点初步评估

- 协调问题, 调参, 尝试。
- 多任务学习方案的代码问题 (GPT 的 debug 能力?)

选择动机

- 比赛新鲜 (2.6 开赛), DDL 时间充裕 (6.16 截止)
- 参赛对于较少, 目前有一人取得高分 (0.89258), 具有较高已知上限。
- 本人完全不会, 有利于充分发挥 GPT 能力。

You
中文概括

ChatGPT

PlantTraits2024 挑战赛是CVPR 2024年FGVC11研讨会的一部分, 旨在通过众包的植物图片和附加数据预测六种植物特性 (如叶面积、植物高度), 以促进我们对全球生物多样性模式的理解。这项挑战的重要性在于, 植物特性是理解生态系统多样性、生产力及其对气候变化反应的关键。竞赛利用公民科学照片, 已经收集了超过2000万张植物图片, 旨在探索人工智能从这些照片预测植物特性的潜力。先前的研究已证明使用卷积神经网络从这些照片预测植物特性是可行的, 尽管数据的异质性和植物外观特征与其化学组织属性之间的关联较弱, 可能导致预测精度中等。

参赛者将使用基于深度学习的回归模型 (如CNN或Transformers) 来预测照片中的植物特性。为了提高模型性能, 建议采取以下策略:

- 多任务学习方案:** 实现能同时预测所有植物特性的模型, 以利用不同特性之间的关系。
- 辅助地理数据整合:** 结合植物照片的气候数据和其他地理空间信息 (如土壤数据、卫星数据), 为模型提供额外的上下文信息, 可能提高预测精度。
- 集成模型:** 测试不同CNN模型组合作为集成模型也可能是有益的。

这项竞赛的成果将大大促进我们对生态系统对全球变化反应的理解, 并对生物多样性研究做出重要贡献。参赛者需要以.csv格式提交预测结果, 评估指标为所有六种特性的平均R2值, 仅考虑R2值大于零的情况, 以避免出现大的负值。这项挑战为参与者提供了一个宝贵的机会, 通过分析公民科学数据, 对植物特性及其对全球生态系统的影晌有更深入的了解。

在挑战赛中的应用: ResNetV2可以被用来从植物图片中提取特征, 以预测植物特性。其较简单的架构 (相比于更大的ResNet模型) 可能使它成为快速原型开发和实验的好选择。

在挑战赛中的应用: EfficientNetV2可以用于更精细地从植物图片中提取特征, 并预测植物特性, 尤其适用于需要较高精度预测的场景。

提出解决方案 & 编写代码 & 调试代码

这部分时本报告重点内容, 其中提供给 ChatGPT 的 prompt 为:

用户:

你是一名 AI 工程师, 你要根据我提供给你的问题, 代码, 相关反馈, 回答我问你的问题, 包括模型工作方式, 改进点, 代码如何修改, 你明白你的任务了吗?

ChatGPT:

明白了! 我会尽力完成任务。

后文中, 将会提供给 ChatGPT 的题面为 (所有的 ... 表示省略, 总文件长度为 10k) :

markdown

PlantTraits2024 - FGVC11

Overview

Goal:

We welcome you to PlantTraits2024 Challenge, an exciting part of the FGVC11 workshop at CVPR 2024, ...

Why Does It Matter?

Think of plants as the superheroes of our ecosystems. ...

Description

This competition aims to predict plant properties - so called plant traits....

Competition

The primary objective of this competition is to employ deep learning-based regression models, ...

Evaluation

The models will be evaluated against the independent test data. ...

Dataset Description

Files ...

Columns ...

Background on Dataset ...

Models

ResNetV2 ...

EfficientNetV2...

这是给出的数据: ...

而提供的 baseline 代码是 [PlantTraits2024: KerasCV Starter Notebook | Kaggle](#) 转化为 .py 文件后的结果。

在接下来的内容中，将简述笔者和 GPT 协作进行训练 AI 模型若干方向、思路、结果。

探索思路与结果

GPT 从头开始写代码

探索方式

给 GPT 提供题面后，先让其思考解决方式，然后给出初步代码，接着追问并补充信息，如输入数据、图片位置，训练数据表格具体内容，让 GPT 接着输出代码，并提供报错信息，不断进行调试。

结果

该方向已被笔者放弃。

总结

- 在追问并补充信息环节，笔者已经非常恼火，该环节时就已经有无数信息需要笔者复制（尤其是题目复杂，表格处已经有了 160 多列，每一列都要提供给 GPT 进行参考，当然最后还要笔者亲自写 py 脚本提取重要列）。在这里，笔者纯粹充当了一个中间交互的计算机，没有足够的速度同时耐心有限，非常麻烦。
- 由于 GPT 的长文本逻辑关联不够强悍，生成代码上下变量名有不对齐的现象（GPT 的“假设 xxx 是 xxxx”），需要笔者自行修改或者让 GPT 进行修改。
- GPT 的代码仍然有一定的 bug 存在，需要笔者反复询问挖掘问题和解决方案，最后代码发生奇怪的卡住使得笔者最后放弃。

GPT 修改代码

探索方式

通过将题面和 baseline 代码传给 GPT 后，让 GPT 理解代码逻辑并且提出修改建议，最后从修改建议中挑选笔者认为合适的，进而让其提供修改位置和修改内容进行修改。

结果

GPT 正确地梳理了 baseline 的逻辑：

这份代码通过以下步骤构建了一个用于植物特征预测的深度学习模型，并完成了预测和提交：

1. **介绍和环境设置**：首先介绍了使用 Keras 和 KerasCV 进行植物特征预测的目标，并设置了环境变量以选择模型运行的后端（TensorFlow、PyTorch、JAX）。
2. **安装库**：安装了所需的 KerasCV、TensorFlow 和 Keras 库的本地版本，因为比赛环境中禁用了互联网访问。
3. **导入库**：导入了实现项目所需的各种 Python 库，~~包括~~ Keras、KerasCV、TensorFlow、Pandas、Numpy 等。
4. **设置配置类**：定义了一个配置类（CFG），在其中设置了模型的各种参数，如预训练模型、图像尺寸、批量大小、学习率计划、目标类别等。
5. **重现性设置**：通过设置随机种子来确保实验的可重现性。
6. **数据路径**：设置了数据集的基本路径。
7. **元数据介绍**：解释了数据集中包含的植物特征信息，并说明了预测任务的目标。
8. **数据加载器（DataLoader）**：构建了一个数据加载器，用于同时处理 JPEG 图像和表格特征作为输入，并应用了多种数据增强方法。
9. **数据分割**：使用分层折叠（StratifiedKFold）将数据分为训练集和验证集。
10. **构建训练和验证数据集**：基于之前步骤的数据分割结果，创建了用于模型训练和验证的数据集，并进行了特征的标准化处理。
11. **损失函数和评价指标定义**：定义了 R^2 损失函数和评价指标，用于评估模型的性能。
12. **模型建立**：构建了一个多输入多输出的深度学习模型，使用 EfficientNetV2 作为图像特征提取的主干网络，并通过 Dense 层处理表格数据。
13. **模型概览**：使用 plot_model 函数展示了模型的结构。
14. **学习率计划**：定义了一个学习率调度器，根据预设的学习率模式在训练过程中动态调整学习率。
15. **模型检查点设置**：设置了模型检查点回调，以便在训练过程中保存性能最好的模型。

16. **模型训练**: 使用之前定义的数据加载器、损失函数、评价指标和学习率计划对模型进行训练。
17. **训练结果展示**: 展示了模型在验证集上达到的最佳 R^2 分数和对应的训练周期。
18. **预测和提交**:
 - 加载训练过程中保存的最佳模型。
 - 构建了测试数据集，并对测试数据进行了特征的标准化处理。
 - 使用加载的最佳模型对测试数据进行预测。
 - 根据预测结果创建了提交文件，并将其保存为 CSV 格式，准备提交到比赛平台。

这份代码涵盖了从数据预处理、模型设计和训练到预测和提交的整个流程，展示了如何在禁用互联网的环境中利用 Keras 和 KerasCV 来解决复杂的多输入多输出预测任务。

从这里已经可以一瞥任务和 baseline 代码的复杂程度。

接着让 GPT 给出修改建议：

用户：

深呼吸，告诉我你如何修改这份程序，以提高其预测精准度，先给出一些简单的改进。

ChatGPT：

为了提高这份程序的预测精确度，我们可以从数据预处理、模型架构、训练过程和特征工程等几个方面入手，进行一些简单的改进：

数据预处理和增强

1. **更多的数据增强**: 增加更多的数据增强技术，比如色彩抖动、随机裁剪等，可以帮助模型学到更鲁棒的特征，提高泛化能力。
2. **高分辨率图像**: 如果计算资源允许，使用更高分辨率的图像可以让模型学到更细粒度的特征，可能提升预测精度。

模型架构

1. **尝试不同的预训练模型**: 除了 EfficientNetV2，还可以尝试其他的预训练模型如 ResNet、DenseNet 等，找到最适合当前任务的模型架构。
2. **增加模型复杂度**: 在不过拟合的前提下，可以通过增加网络的深度或宽度来提升模型的学习能力。
3. **双向模型或集成学习**: 可以尝试使用两个模型分别从图像和表格数据学习，然后将它们的输出合并，或者使用集成学习的方法结合多个模型的预测结果。

训练过程

1. **调整学习率调度器**: 尝试不同的学习率调度策略, 如余弦退火、学习率衰减等, 可能会有助于更好地优化模型。
2. **使用早停法**: 设置早停回调, 当验证集上的性能不再提升时停止训练, 以避免过拟合。
3. **更细粒度的损失函数权重调整**: 对主任务和辅助任务的损失函数权重进行更细致的调整, 找到最优的平衡点。

特征工程

1. **高级特征提取**: 对于表格数据, 可以尝试使用一些高级的特征选择方法, 比如基于模型的特征选择, 来提取最有用的信息。
2. **特征交叉和多项式特征**: 创建特征的交叉项或多项式特征, 可能会揭示特征之间复杂的非线性关系, 有助于提高预测精度。
3. **外部数据**: 如果比赛规则允许, 可以尝试集成外部数据源, 比如公开的植物数据库或气象数据, 这可能会为模型提供额外的有用信息。

这些简单的改进措施可以作为提升模型预测精度的起点。在实际操作中, 每一步都需要细心调整和多次实验, 以找到最佳的模型配置。

多次询问有多种策略 (为了篇幅, 仅列出关键词) :

1. 多模态学习深入集成
2. 自注意力机制与 Transformer 模型
3. 图像分割与特征定位
4. 无监督或半监督学习
5. 元学习和少样本学习
6. 模型可解释性和后处理
7. 跨领域迁移学习
8. 强化学习和动态调整

数据方面的改进

1. 使用 GAN 生成的图像
2. 更复杂的数据增强技术

模型结构方面的改进

1. 注意力机制
2. 多尺度特征融合

训练技巧方面的改进

1. 循环学习率
2. 软标签
3. 知识蒸馏

模型融合方面的改进

1. 模型集成
2. 交叉验证融合

进而产生了多个修改思路方式，将在下文进行阐述。

同时，作为基准，baseline 的运行结果为（由于数据更新，该结果和官网上 baseline 的 notebook 分数不同）：



PlantTraits2024: KerasCV Starter Notebook a6817d - Version 1

Complete · 2d ago · Notebook PlantTraits2024: KerasCV Starter Notebook a6817d | Version 1

-9.21458

初步修改

探索方式

直接询问：“请在之前给你的代码中进行修改，告诉我修改位置和内容”。

结果

GPT 给出了四个位置的修改：

```
python

def build_augmenter():
    # Define augmentations
    aug_layers = [
        # 省略原有代码
        # 新增的增强方法
        keras_cv.layers.RandomTranslation(height_factor=0.1, width_factor=0.1),
```

```
]
```

```
# 以下省略原有代码...
```



```
python
```

```
# Branch for image input
# 从EfficientNetV2换成ResNet50
backbone = keras.applications.ResNet50(weights='imagenet', include_top=False, input_shape=(*CFG.image_size, 3))
```



```
python
```

```
def get_lr_callback(batch_size=8, mode='cos', epochs=10, plot=False):
    # 修改学习率调度逻辑以支持余弦退火
    # 以下省略原有代码...
    elif mode == 'cos':
        # 余弦退火策略的实现
        # 以下省略原有代码...
```



```
python
```

```
# Branch for tabular/feature input
x2 = keras.layers.Dense(128, activation="selu")(x2) # 新增的Dense层
```

其中第三个修改笔者并未采用，最后结果为：



PlantTraits2024: KerasCV Starter Notebook - Version 2

Complete · 9d ago · Notebook PlantTraits2024: KerasCV Starter Notebook | Version 2

-9.18056

可以看出，该修改比较普通，和 baseline 分数相当。

增加注意力机制

探索方式

在询问了修改建议后，直接询问：“请你尝试在模型结构中增加注意力机制，并告诉我修改部分和内容”。

结果

这是 GPT 回答：

..... (笔者注：限于篇幅，此处 GPT 给出的修改代码省略)

在这段示例代码中，首先定义了一个 `se_block` 函数来构建 SE 块，然后在加载预训练模型后，将这个 SE 块插入到模型的输出部分。在这里，我们以 `ResNet50` 模型为例，但实际上这种方法也适用于其他模型，如 `EfficientNetV2` 等。最后，添加了全局平均池化层和输出层，完成了整个模型的构建。

通过在模型中增加注意力机制，我们期望模型能够更加关注于对于最终任务更为重要的特征，从而提高模型的性能。

然而这段代码和 `baseline` 的兼容仍然有问题，导致编译报错，笔者接着询问几次也没有得到解决，作为中间层的笔者无法忍受长时间的复制、询问、等待于是放弃该思路。

知识蒸馏

探索方式

询问：“写出利用知识蒸馏改进的代码”，当看到 GPT 回答的前面内容时候，笔者毕竟意识到了 ChatGPT 已经无法记忆前面的代码：

知识蒸馏是一种将大型模型（教师模型）的知识转移到小型模型（学生模型）的技术。通过这种方式，小型模型可以在保持较低计算成本的同时，学习到大型模型的表现能力。这里，我将展示如何在 TensorFlow/Keras 中实现知识蒸馏的基本步骤。

首先，我们需要定义教师模型和学生模型。在这个例子中，我们将使用简化的模型来说明概念（笔者注：这个就是标志）：

于是笔者迅速将原来 `baseline` 的代码再次复制给 GPT，这次，GPT 给出的结果很不错。

但是接着出现了报错，这是笔者的询问（注意，由于 GPT 的机制，可以忽略格式进行粘贴，这里并未进行格式转化）：

```
3 def build_student_model(input_shape):    4     inputs =  
keras.Input(shape=input_shape, name="images") ----> 5     x =  
keras_cv.layers.RandomResizedCrop(input_shape)(inputs)    6     x =  
layers.Conv2D(16, 3, activation='relu')(x)    7     x = layers.MaxPooling2D(pool_size=2)  
(x) AttributeError: module 'keras_cv.layers' has no attribute 'RandomResizedCrop'
```

在这里，`build_student_model` 报错了，你将如何处理。

结果

GPT 给出了正确的代码，但是很不幸，在运行几次，询问几次修改建议后，该代码仍然报错（显存不足），这是运行环境受限以及 GPT 没有当前环境信息导致的，难以避免。

题外话：此时笔者意识到了 GPT 的记忆力受限于是导致上一个探索失败，但是由于时间紧迫，笔者并未重新尝试。

Transformer 集成

探索方式

直接询问：“将 Transformer 集成到这份代码中，告诉我修改的位置和内容”。中途出现了一个错误，在和 GPT 共同努力下，该报错成功解决。

结果



PlantTraits2024: KerasCV Starter Notebook - Version 4

Complete · 6d ago · Notebook PlantTraits2024: KerasCV Starter Notebook | Version 4

-8.65658

有了一定进步。

接着笔者切换了使用的模型，有了不错的进步。



PlantTraits2024: KerasCV Starter Notebook - Version 6

Complete · 4d ago · Notebook PlantTraits2024: KerasCV Starter Notebook | Version 6

-8.15436

Vision Transformer (ViT) 集成

探索方式

笔者成功注意到 ChatGPT 引入的仅仅是 Transformer，而非 Vision Transformer (ViT)，于是笔者询问“之前的代码中可以使用 vision transformer”要求 GPT 进行修改。

结果

ChatGPT 首先对于这个建议给出了赞同的意见：

如果您考虑在项目中使用 Vision Transformer (ViT) 替代传统的 CNN 模型来处理图像数据，这是一个很好的选择，特别是当您处理的是高分辨率图像或者需要模型捕获全局依赖性时。Vision Transformer 通过将图像分割成多个小块 (patches)，然后将这些块处理成序列数据，从而能够利用 Transformer 强大的序列处理能力来理解图像。

然后笔者和 GPT 进行了多轮交互，尝试将 Kaggle 中 Models 的 ViT 进行集成到代码中，但是因为代码是 keras / tensorflow 框架，而 Kaggle 上的是 Pytorch 方式，难以解决该兼容问题，直接加载网络上的又出现了不明报错（可能是禁止了网络，也可能是 GPT 并未给出正确网址），最后笔者选择放弃。

元学习和少样本学习技术集成

结果

GPT 给出的修改过多，笔者目测难度选择放弃。

转机：参考社区内容

这时笔者访问了 Kaggle 上 Discussion 模块，发现了该比赛的数据有很多极端值，同时笔者发现了也是修改 baseline 的公开代码，于是将公开代码复制进入 ChatGPT 询问该代码的改进点。

- 读取训练和测试数据集，为图像路径添加前缀，并处理缺失数据。
- 过滤数据集以去除异常值。
- 使用 `SelectKBest` 和 `f_regression` 从表格数据中选择最重要的特征。
- 对数据集进行分层折叠划分，以便进行交叉验证。

这是 ChatGPT 的一段核心回答，笔者发现了 `SelectKBest` 和 `f_regression`，于是让 ChatGPT 进行类似修改。

数据简化

探索方式

询问：“已知，此问题中有很多元素都有很多极端值，告诉我有什么影响以及如何处理”。

ChatGPT 给出回答的核心：

1. **剔除**：如果极端值是由错误的数据录入或明显的异常情况导致，可以考虑直接剔除这些数据点。
2. **转换**：通过对数据进行转换来减少极端值的影响。常见的转换方法包括对数转换 (`log`)、平方根转换 (`sqrt`) 和倒数转换。
3. **缩放**：使用稳健的缩放方法，如 `RobustScaler`，它通过使用四分位数范围（而不是最大值和最小值）来缩放数据，从而减少极端值的影响。
4. **分箱**：将连续变量分箱（离散化）也可以减少极端值的影响。这种方法将数据分成几个区间，可以有效地将极端值“归类”到最高或最低的箱中。
5. **Winsorizing**：这是一种减少极端值影响的方法，通过将数据集中的极端值替换为数据的其他部分的最大值和最小值。例如，可以将所有超过 95% 分位数的值设置为 95% 分位数，将所有低于 5% 分位数的值设置为 5% 分位数。

接着笔者将参考代码数据简化参数给了 ChatGPT，并让其同时使用 `SelectKBest` 从表格数据中选择最重要的特征。

结果

ChatGPT 给出两个可行的思路，笔者同时提交了这两个代码：

	PlantTraits2024: KerasCV Starter Notebook - Version 7	-9.05313
	Complete · 4d ago · Notebook PlantTraits2024: KerasCV Starter Notebook Version 7	

	PlantTraits2024: KerasCV Starter Notebook - Version 9	-8.07696
	Complete · 4d ago · Notebook PlantTraits2024: KerasCV Starter Notebook Version 9	

可以看出，有一份有了很明显的提升，但是这一份的提升很多建立在 GPT 优化的基础上。

	1-PlantTraits2024: KerasCV Starter Notebook 66a9fa - Version 1	-8.20703
	Complete · 3d ago · Notebook 1-PlantTraits2024: KerasCV Starter Notebook Version 1	

	PlantTraits2024: KerasCV Starter Notebook - Version 10	-8.25253
	Complete · 4d ago · Notebook PlantTraits2024: KerasCV Starter Notebook Version 10	

而之后笔者进行少量参数修改，结果略低。

多模型整合

结果

由于运行环境不足以及笔者和 GPT 合作也无法发现的问题（局限性，对接不行），超出了显存，无法运行。

	Version 16	
	Save & Run All • Diff: +7 -7	
	Failed after 8 minutes and 53 seconds	
	Version 15	
	Save & Run All • Diff: +11 -5	
	Failed after 5 minutes and 53 seconds	
	Version 14	
	Save & Run All • Diff: +3 -3	
	Failed after 6 minutes and 2 seconds	
	Version 13	
	Save & Run All • Diff: +2 -2	
	Failed after 9 minutes and 48 seconds	
	Version 12	
	Save & Run All • Diff: +1 -1	
	Failed after 15 minutes and 53 seconds	
	Version 11	
	Save & Run All • Diff: +56 -20	
	Failed after 16 minutes and 6 seconds	

最终结果

笔者选择了最后成功的代码，分数有了一定优化，但是并非突破性质的。

总结

总体来说，ChatGPT 已经具有了一定的编写、修改、辅助调试代码能力，这种能力在普通领域中优势明显，但是在面对专业性、系统综合性过高的领域上就有点力不从心，在这些领域上，ChatGPT 就只能作为一个辅助工具，而操作者仍然需要一定知识能力水平。

ChatGPT 优点

- 能够很好地给出建议，具有开阔的视角和丰富的经验。
- 长文本能力能够支持在很长的代码中进行调试修改。
- 面对报错（大部分）能够给出问题解决方案。
- 是一个很好的帮手。

ChatGPT 不足

- 长文本能力很好，但是仍然不够，对于这种高难度文本长的项目，笔者进行操作的时候，仅仅是几轮对话下来，ChatGPT 就会忘记原来的代码，导致需要重新复制。这个对于中间的操作者来说是非常消磨耐心的。
- 在对于系统综合性强（各种环境协调、代码协调）的任务中，ChatGPT 的配合能力会还有所下降，而在这种任务中，通常是调试代码环节，需要操作者多多进行询问，而且不一定能解决问题。
- ChatGPT 对于现有的常规性任务处理还是不错，能有一定优化效果，但是对于原创性、开放性的问题，解决能力有所欠缺，这点在本次探索中也清晰可见，GPT 的代码较于 baseline 有一定优化，但仍然无法达到排行榜上靠前的位置（当然不排除数据更新导致新旧结果不统一）。不过对于其他专业领域的综合性问题，ChatGPT 应该有不错的效果（AI 方向困难重重）。

未来可行（此次探索未讨论）的方向

- 将 ChatGPT 和向量数据库结合，以获得更好的效果（可以借助 AutoGPT 等工具）。
- 类似上一点，使用其他增加 ChatGPT 记忆范围的技术，改进效果。
- 使用更加对接更加紧密的工具，减少操作者在中间复制、交互的比重（不然繁琐的任务会让人失去耐心）。

最后，这份报告也仅仅代表 ChatGPT 和人在 AI 领域训练模型的初步尝试，毕竟没有在模型上调参、使用更好的运行环境等等操作，不能完全发掘出 ChatGPT 的潜力，但我们可以发现 ChatGPT 已经能辅助 / 自主进行 AI 模型训练了。