附录A：chatgpt命令的安装与基本语法

Stata的chatgpt命令第一版于2023年4月首次上线SSC（Statistical Software Components），目前为第三版，如今还在持续更新中，以适应GPT本身的不断发展。

在Stata命令框输入并执行：ssc install chatgpt, replace，显示安装完成即可使用。使用chatgpt需要用户首先安装cURL[[1]](#footnote-1)，并将cURL的路径设置为环境变量。以Windows系统为例，通过控制面板搜索高级系统设置，点击“环境变量”-“系统变量”中的“新建”，填入cURL文件所在路径即可，如下图1所示。

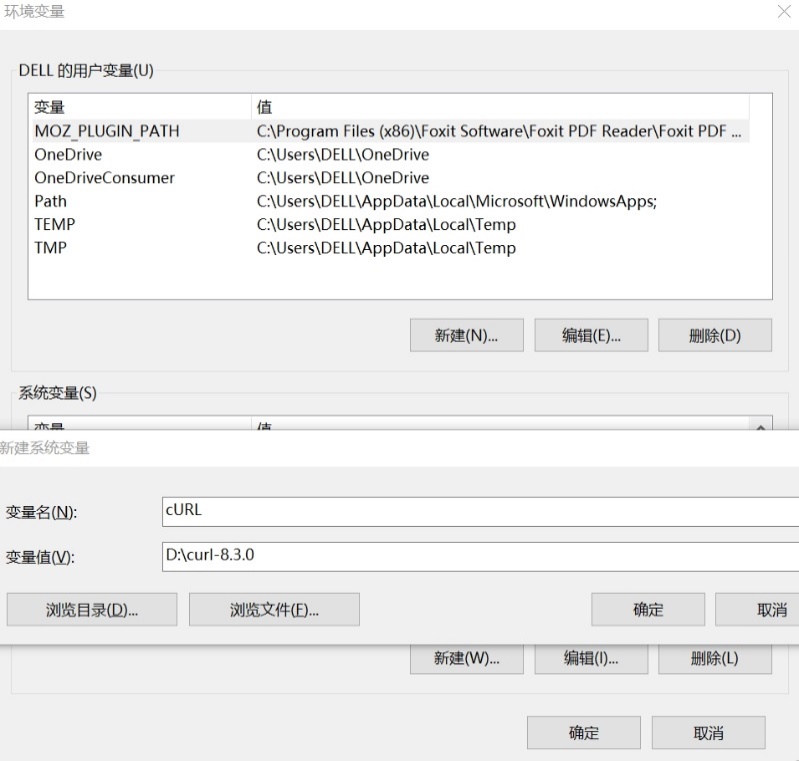


图1 将cURL的路径设置为环境变量

同时，Windows和MacOS用户均需预装Python环境。

chatgpt与Stata的其它用户开发命令一样，在成功安装后，可以通过help chatgpt查阅其帮助信息。通过阅读sthlp文件，我们可以看到chatgpt的命令基本格式如下所示：

chatgpt [ subcommand ] [, options ]

其中，subcommand包括talk、read、session、test。chatgpt命令作为连接GPT的Stata接口，拥有问答（talk）、阅读（read）、构建会话（session）以及测试（test）四种模式。在talk模式中，用户可以向GPT提问，相应的回复将会出现在Stata结果显示窗口中。在read模式中，用户可以要求GPT阅读指定的文本文件、数据、Stata do-file代码或sthlp文件等，并据此回答相应问题。在session模式中，用户可以在Stata中与GPT执行结合上下文情景学习的对话，像OpenAI推出的应用服务ChatGPT一样使用自然语言顺畅交流。在test模式中，用户可以检验命令的安装与相应的配置是否存在问题，根据指引完整配置工作。

用户可以通过选项options设置如下信息：openai\_api\_key(string)、stata、engine(string)、systemprompt(string)、command(string)等。

其中openai\_api\_key(string)是必选项，用户通过该选项输入自己申请的API Key；选项stata用于声明指令主题与Stata相关，使GPT专注Stata相关的技术问题，否则与Stata无关。如我们指定read模式，要求GPT阅读Stata命令psemail的ado文件，此时需要在openai\_api\_key(string)中填入申请的API Key，并添加stata选项，程序如下：

chatgpt read psemail.ado, openai\_api\_key($OPENAI\_API\_KEY) stata

此处需要特别注意的是，API Key是重要的私人信息，不可以对外公开，我们一般会将其放在本地硬盘的一个纯文本文件中，如该文本文件的名字是APIKEY.txt，保存在E:/chatgpt/子目录下，我们可以在最开始先执行一条读入API Key并放置于全局宏（global）的命令，完整的命令如下：

global OPENAI\_API\_KEY=fileread(e:/chatgpt/APIKEY.txt)

chatgpt read psemail.ado, openai\_api\_key($OPENAI\_API\_KEY) stata

选项engine(string)用于指定会话的语言模型引擎，当前OpenAI 可以选择的模型引擎列表如下：gpt-3.5-turbo、gpt-3.5-turbo-0301以及gpt-4。前两个聊天引擎都是基于GPT-3.5模型的优化版本，其中gpt-3.5-turbo-0301更加关注问题内容，不会特别关注自身所要扮演的角色，而gpt-3.5-turbo是OpenAI官方推荐的GPT-3.5模型系列中最好的版本，因为它成本最低但回复效果最好。对于gpt-4模型，OpenAI称其比任何 GPT-3.5 模型都更强大，能够执行更复杂的任务，并且针对聊天进行了优化。经过我们实际测试体验，gpt-4在语义理解、常识储备等方面不仅超越了以往模型，而且足以媲美专业人员，体现出惊人的智能。因此，本文演示的所有案例均通过gpt-4模型完成。如我们指定会话的语言引擎为gpt-4，并询问GPT在Stata中关于描述性统计的命令，程序如下：

global OPENAI\_API\_KEY=fileread(e:/chatgpt/APIKEY.txt)

chatgpt talk, openai\_api\_key($OPENAI\_API\_KEY) command("除了summarize以外，有没有其它描述变量统计特征的命令") stata engine("gpt-4")

选项command(string)用于提出对GPT的要求；选项systemprompt(string)用于指定会话提示词，有助于GPT更好地理解和回答问题。如我们指定两段文本，要求GPT比较这两段文本的相似度，并提示通过余弦相似度的方法来计算文本相似度，程序如下：

global OPENAI\_API\_KEY=fileread(e:/chatgpt/APIKEY.txt)

chatgpt talk, openai\_api\_key($OPENAI\_API\_KEY) command("文本相似度分析：1.我喜欢Stata和Python；2.我们都热爱Stata和Python还有文本分析技术") systemprompt("通过余弦相似度计算")

详细的使用案例可以通过在Stata内输入help chatgpt获取。

附录B：chatgpt命令辅助泛技术类工作专门示例

为了方便读者理解大语言模型如何提高实证研究中泛技术类工作的效率，本文基于Stata统计软件和chatgpt命令，列举以下五种主题案例以及独立完整的实现代码。

本文的所有数据及代码均放置于云端数据库，读者可通过以下命令下载获得：

copy "https://stata-chatgpt.oss-cn-beijing.aliyuncs.com/chatgpt/main.do" "main.do", replace

（一）命令查询

在使用Stata软件的过程中，当我们遇到只清楚处理需求但不知道具体命令的情形时，可以使用chatgpt命令进行查询，比如我们想知道如何获取系统数据“auto.dta”中变量mpg的20%分位数，可以使用如下程序询问具体命令如何编写：

clear all

sysuse auto, clear

global OPENAI\_API\_KEY=fileread(e:/chatgpt/APIKEY.txt)

chatgpt talk, openai\_api\_key($OPENAI\_API\_KEY) command("如何查询变量mpg的20%分位数") stata engine("gpt-4")

此时，在Stata的结果显示窗口返回了GPT回答的具体命令“centile mpg, centile(20)”，如下图2所示。



图2 chatgpt的Stata专门示例-命令查询

由于chatgpt命令会识别GPT回复中的Stata代码，并将其转化为蓝色的可点击形式，因此用户可以直接点击执行chatgpt提供的相关命令。

（二）代码生成

在数据清洗的许多情形下，我们的处理需求往往与当前数据紧密相关，无法直接套用sthlp文件中的现成案例。而chatgpt可以便捷地提交样例数据，使GPT基于当前数据帮助我们编写定制程序。以如下图3所示放置于云端的文件“data1.dta”数据为例。

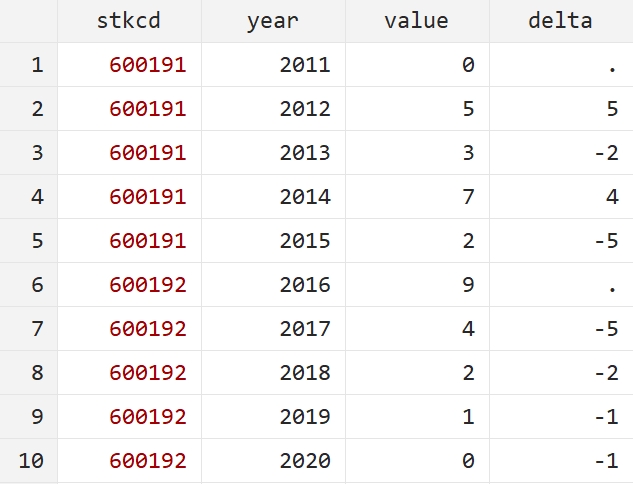


图3 数据“data1.dta”展示

我们想要根据已有变量“stkcd”、“year”和“value”生成变量“delta”，可以通过如下语句询问具体程序的编写：

clear all

webuse set "https://stata-chatgpt.oss-cn-beijing.aliyuncs.com/chatgpt"

webuse data1, clear

global OPENAI\_API\_KEY=fileread(e:/chatgpt/APIKEY.txt)

chatgpt read data, openai\_api\_key($OPENAI\_API\_KEY) stata ///

command("观察数据详情，请指出如何通过stkcd和year生成delta") ///

do("list")

当chatgpt选择“read data”模式，Stata将首先执行do选项中的语句，并将执行结果作为数据描述内容，附在询问内容后提交。此时，Stata结果显示窗口返回了通过变量“stkcd”、“year”和“value”生成变量“delta”的程序“bysort stkcd (year): generate delta = value - value[\_n-1]”，如下图4所示。



图4 chatgpt的Stata专门示例-代码生成

（三）命令速览

在面对从未接触过的Stata新命令时，也可以通过chatgpt询问其具体使用方法。如我们以cntraveltime（Zhang等，2023）[[2]](#footnote-2)命令为例，通过以下程序询问该命令能否计算远洋轮船的行程距离：

clear all

ssc install cntraveltime, replace

global OPENAI\_API\_KEY=fileread(e:/chatgpt/APIKEY.txt)

chatgpt read cntraveltime.sthlp, openai\_api\_key($OPENAI\_API\_KEY) stata ///

command("该命令能否计算远洋轮船的行程距离？")

当chatgpt选择“read”模式并给出要阅读的“cntraveltime.sthlp”文件名后，chatgpt命令会将文件内容一并提交，GPT将在阅读新命令的帮助文件sthlp后，回答我们该命令的使用细节。此时，在Stata的结果显示窗口，针对我们提出的问题，GPT为我们提供了“该命令只能计算陆地交通工具（如汽车、公共交通和自行车）的行程距离和行程时间，无法计算远洋轮船的行程距离”的回答，这与该命令的实际情形相符，如下图5所示。

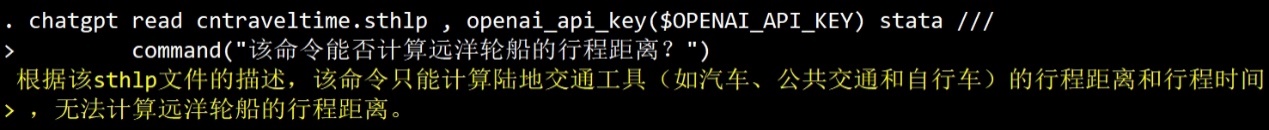


图5 chatgpt的Stata专门示例-命令速览

（四）报错答疑

在使用Stata软件编写程序时，难免会遇到程序报错而出错点却很难寻找的情形。但对于基于大数据训练得到的GPT而言，找出代码中的潜在错误十分容易，特别是在了解了我们的数据详情后。以如下使用系统数据“auto.dta”的错误程序为例，通过chatgpt命令帮助我们“修BUG”：

clear all

sysuse auto, clear

reg mpg prlce

这段程序报错的结果如下图6所示。

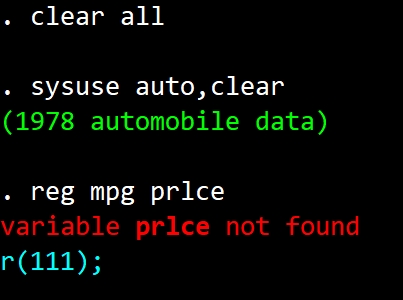


图6 程序报错结果

此时，我们要求chatgpt使用“read data”模式，通过指定do选项，让Stata执行“sum \*”，将当前数据集的描述结果提交给GPT，并询问报错如何改正，相应程序如下所示：

global OPENAI\_API\_KEY=fileread(e:/chatgpt/APIKEY.txt)

chatgpt read data, openai\_api\_key($OPENAI\_API\_KEY) stata ///

command("我输入reg mpg prlce报错：variable prlce not found，怎么改") do(sum \*)

通过询问GPT，我们得到回复“您输入的命令中的变量名拼写错误，应该是price而不是prlce”，如下图7所示。

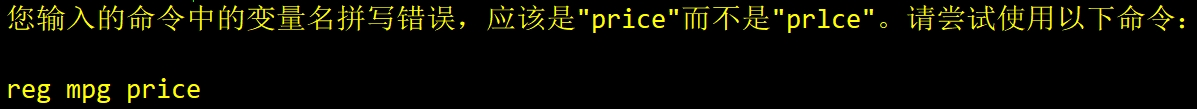


图7 chatgpt的Stata专门示例-报错答疑

（五）实证答疑

针对实证研究中各种命令的编写，我们均可通过chagpt命令向GPT询问，如我们询问“回归时以变量prov作为聚类变量计算聚类稳健标准误”的程序应如何编写，如下所示：

global OPENAI\_API\_KEY=fileread(e:/chatgpt/APIKEY.txt)

chatgpt talk, openai\_api\_key($OPENAI\_API\_KEY) command("回归时以变量 prov 作为聚类变量来计算聚类稳健的标准误。") stata engine("gpt-4")

此时，在Stata结果显示窗口，GPT为我们提供了程序“regress y x, vce(cluster prov)”，并对具体变量进行了解释，如下图8所示。



图8 chatgpt的Stata专门示例-实证答疑

附录C：chatgpt命令辅助文本工作专门示例

由于GPT本身是一个通用的自然语言处理模型，因此除了解决Stata专门问题以外，chatgpt命令也可以完成包括翻译、语言润色等诸多NLP经典任务，本附录提供相关案例以及独立完整的实现代码如下。

（一）翻译

我们以本文的中文摘要为例进行翻译。摘要存在众多文本分析领域的专业术语，我们希望获得一份可用的、有学术风格的英文摘要。该摘要已以纯文本的形式存储为云端文件“content1.txt”，我们键入如下指令：

clear all

copy "https://stata-chatgpt.oss-cn-beijing.aliyuncs.com/chatgpt/content1.txt" "content1.txt", replace

global OPENAI\_API\_KEY=fileread(e:/chatgpt/APIKEY.txt)

chatgpt read other, openai\_api\_key($OPENAI\_API\_KEY) systemprompt("你是一名精通中文与英文的翻译，请将以下中文摘要翻译为符合学术风格与规范的英文。") engine("gpt-4") file("content1.txt")

当chatgpt选择“read other”模式，Stata将file选项中指定的文件内容附在询问内容后提交。该命令获得的翻译如下图9所示，可以看到GPT所提供的回复确实为中文摘要的对应翻译，可以胜任学术文献的翻译任务。

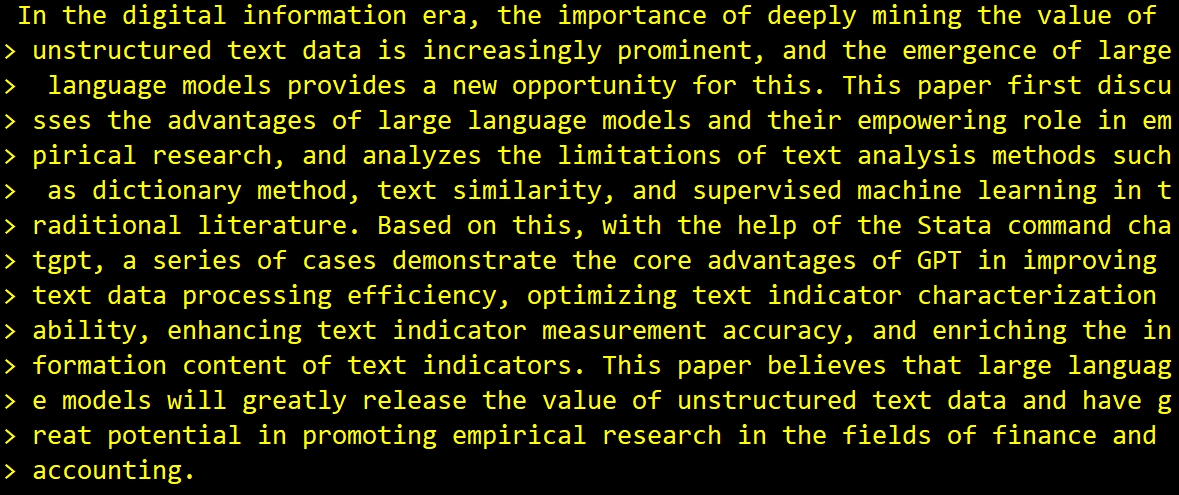


图9 chatgpt的通用示例-翻译

（二）润色

为进一步考察图9中GPT的翻译质量，我们使用Google翻译软件对文章摘要进行翻译，获得机器翻译版英文摘要，要求GPT同时阅读论文原文提供的中文摘要、机器翻译版英文摘要和自己的翻译结果，逐句判断两个英文翻译中哪个翻译质量更好。原文中文摘要、Google翻译版英文摘要和GPT翻译版英文摘要已存储为云端文件“content2.txt”，运行命令如下：

clear all

copy "https://stata-chatgpt.oss-cn-beijing.aliyuncs.com/chatgpt/content2.txt" "content2.txt", replace

global OPENAI\_API\_KEY=fileread(e:/chatgpt/APIKEY.txt)

chatgpt read other, openai\_api\_key($OPENAI\_API\_KEY) systemprompt("你是一名精通中文与英文的翻译，请阅读以下中文摘要及两段对应的英文翻译，尽可能详细举例判断哪份翻译质量更好，更符合学术风格和规范，使用中文回答") engine("gpt-4") file("content2.txt")

GPT回复认为，第二份翻译（由GPT所给出）较之第一份翻译（原文英文摘要）的风格更加正式和学术化，Stata结果显示窗口中GPT的回复如下图10所示。

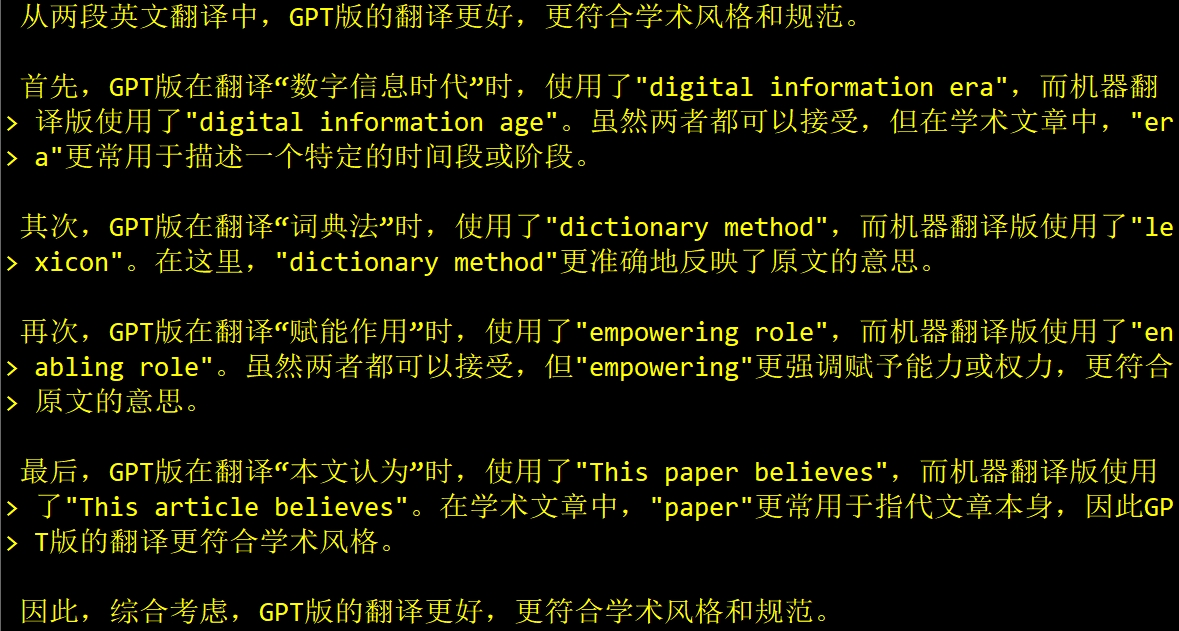


图10 chatgpt的通用示例-润色

附录D：使用chatgpt命令进行数据清洗

在数据清洗过程中，存在着大量目标易识别、过程难处理的基础工作。我们以一段放置于云端的复杂文本“content3.txt”为例，展现GPT数据清洗与信息的智能提取能力，文本信息如下图11所示。

图形用户界面, 文本, 应用程序

描述已自动生成

图11 文本文件“content3.txt”内容展示

我们编程的基本思路是，首先使用read other模式，通过file()选项处理“content3.txt”的文本信息，键入命令如下：

clear all

copy "https://stata-chatgpt.oss-cn-beijing.aliyuncs.com/chatgpt/content3.txt" "content3.txt", replace

global OPENAI\_API\_KEY=fileread(e:/chatgpt/APIKEY.txt)

chatgpt read other, openai\_api\_key($OPENAI\_API\_KEY) command("以csv格式报告中国四个省份的人口，人口单位统一为万人，中文回复") engine("gpt-4") file("content3.txt")

GPT清洗结果如下图12所示。可以看到，GPT展现出了强大的信息提取能力，不仅有效规避了无意义词汇，还准确捕捉到了相关英文和中文信息。此外，GPT也理解了数据的计量单位，将所有省份人口数字的计量单位进行了统一处理。

**文本

描述已自动生成**

图12 GPT-4模型提取信息结果

更进一步地，假设我们需要处理多国的原生文本信息，以存放于云端的data2.dta为样本集为例，该数据包含100个国家的国家名称、领土面积、首都、经纬度、官方语言等信息，这些信息用各种语言写成，部分文档内容如下图13所示。

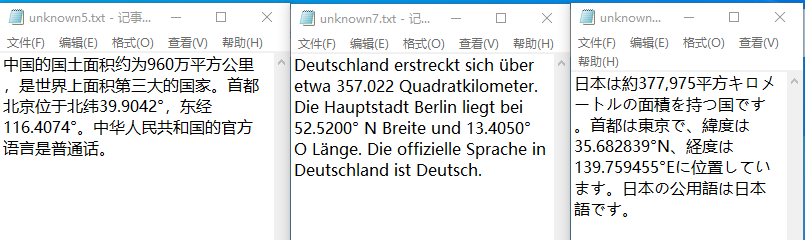


图13 多国语言文本内容展示

我们的目标是将其提取为易于读取的数据格式，仿照前一个例子的方式，这并不难实现。操作流程如下图14所示。

多个复杂文本文件

输出到csv文件

循环读入Stata

chatgpt命令

提取逐个文本信息

图14 使用GPT-4模型提取多国语言文本信息的流程图

我们输入以下程序：

clear all

webuse set "https://stata-chatgpt.oss-cn-beijing.aliyuncs.com/chatgpt"

webuse data2, clear

global OPENAI\_API\_KEY=fileread(e:/chatgpt/APIKEY.txt)

gen GPT\_alalysis = ""

timer clear 1

timer on 1

forvalue i = 1/`=\_N'{

chatgpt talk, openai\_api\_key($OPENAI\_API\_KEY) command(`"`=content[`i']'"') systemprompt("将以下材料翻译为中文并在一行内严格按照输出以下要求的信息。\n1.国家名称\n2.领土面积（以平方公里为单位，只输出数字）\n3.首都\n 4. 经纬度（以十进制方式表示，例如`39.9334,-32.8597`）\n 5.代表性高校。") engine("gpt-4")

replace GPT\_alalysis = fileread("output.txt") in `i'

}

以上每项内容带有双引号，各项内容间使用英文逗号`,`隔开

timer off 1

timer list 1

这个循环程序的含义是，首先，chatgpt命令将在工作路径下生成缓存文件output.txt，该文件内含有本次GPT回复的原文内容。其次，通过fileread捕捉GPT输出的内容，并通过fileread函数读入数据集中，写入对应观测值的GPT\_analysis变量上。这样一来，chatgpt命令可以循环分析每一份文本，最终使得所有原生文本信息均被添加到数据集中。此外，timer命令用于记录循环执行所需要的时间。GPT提取信息的结果如下图15所示。虽然该项工作人工处理并不复杂，但其中细节过于琐碎，技术处理难度极高，而GPT展现了完美解决该类问题的能力。



图15 GPT-4模型提取多国语言信息结果

附录E：使用chatgpt命令进行训练集标注

本文以存放于云端的data3.dta为样本集为例，该数据包含了20篇研报共计245句话。本文采用提示词+研报内容的形式向GPT提交请求，提示词内容包括要求GPT进行分类以及各类别的入选标准，操作流程如下图16所示。

chatgpt命令对文本进行分类

指定特质与非特质信息的提示词

读入Stata

原始训练集文本

图16 使用GPT-4模型进行训练集标注的流程图

具体命令如下：

clear all

webuse set "https://stata-chatgpt.oss-cn-beijing.aliyuncs.com/chatgpt"

webuse data3, clear

global OPENAI\_API\_KEY=fileread(e:/chatgpt/APIKEY.txt)

gen GPT\_alalysis = ""

timer clear 2

timer on 2

forvalue i = 1/`=\_N'{

chatgpt talk, openai\_api\_key($OPENAI\_API\_KEY) systemprompt("请阅读以下材料，将每一句话归为以下两类，分类标准如下：\n 1. 公司特质信息：与公司战略、市场地位、产品、技术、销售、项目投资、融资策略、会计业绩、高管团队、公司治理、供应商、经销商、公司风险等各方面信息相关的信息。\n 2. 非特质信息：会对行业内所有公司产生影响的信息，或宏观经济信息。") engine("gpt-4") command(`"`=content[`i']'"')

replace GPT\_alalysis = fileread("output.txt") in `i'

}

timer off 2

timer list 2

GPT分类结果如下图17所示，可以看到GPT在给定规则下展示出了与人工阅读高度一致的结果。

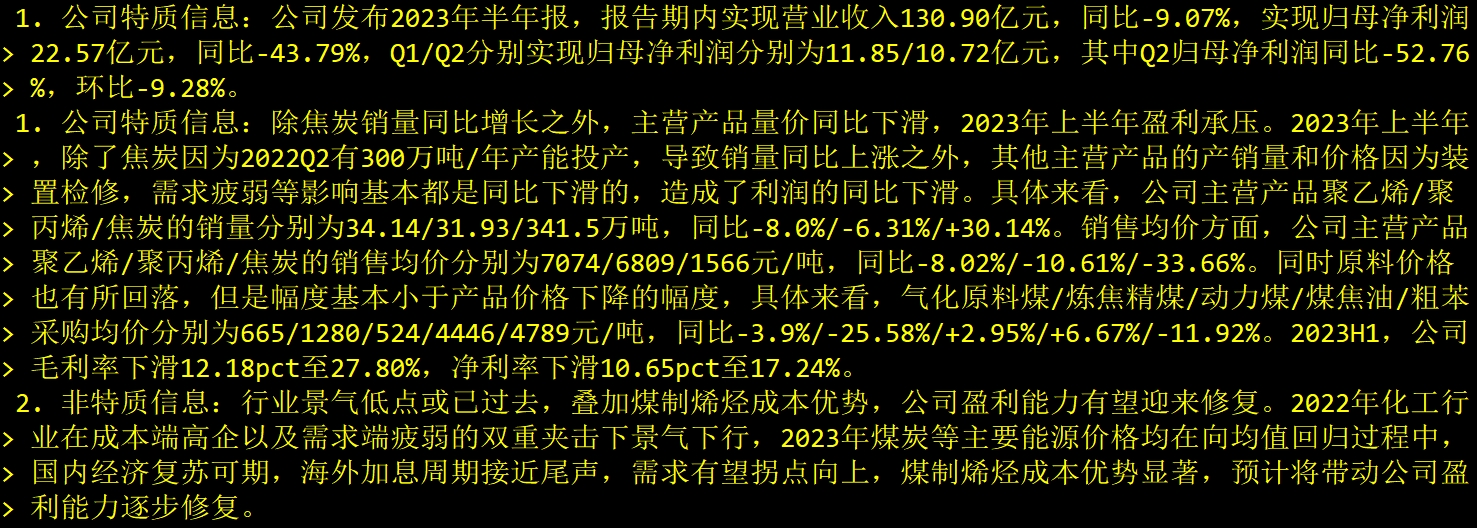


图17 GPT-4模型训练集标注结果

附录F：使用chatgpt命令进行文本语调判断

我们读入放置于云端包含年报MD&A文本信息的数据文件“data4.dta”，要求GPT逐句判断其语调是积极、消极还是中性，其次，为了进一步确定情绪相关的内容类型，我们还要求GPT逐句判断其主旨类型，操作流程如下图18所示。

指定判断文本语调与主旨类型的提示词

读入Stata

chatgpt命令逐句判断文本语调与主旨类型

年报MD&A文本文件

图18 使用GPT-4模型判断文本语调的流程图

命令代码如下所示：

clear all

webuse set "https://stata-chatgpt.oss-cn-beijing.aliyuncs.com/chatgpt"

webuse data4.dta, clear

gen tone = ""

global OPENAI\_API\_KEY=fileread(e:/chatgpt/APIKEY.txt)

timer clear 3

timer on 3

forvalue i = 1/`=\_N'{

chatgpt talk, openai\_api\_key($OPENAI\_API\_KEY) command(`"`=ori\_text[`i']'"') systemprompt("你是一名研究助理，我会发给你一份上市公司年报的表述内容，请严格地逐句进行以下判断。\n 1. 该句话的语调是 积极、消极还是中性。\n 2. 该句话表述内容的主题可以被归纳为宏观环境、企业战略、企业财务、企业业务、法律，如无法归纳请返回无。\n 判断结果严格按照以下格式输出：第X句话“该句话本身”语调判断结果，主题归纳结果。\n... ") engine("gpt-4")

replace tone = fileread("output.txt") in `i'

}

timer off 3

timer list 3

根据上述代码所获得的部分结果如下图19所示。可以发现，文本的情感与主旨类型分类基本正确。

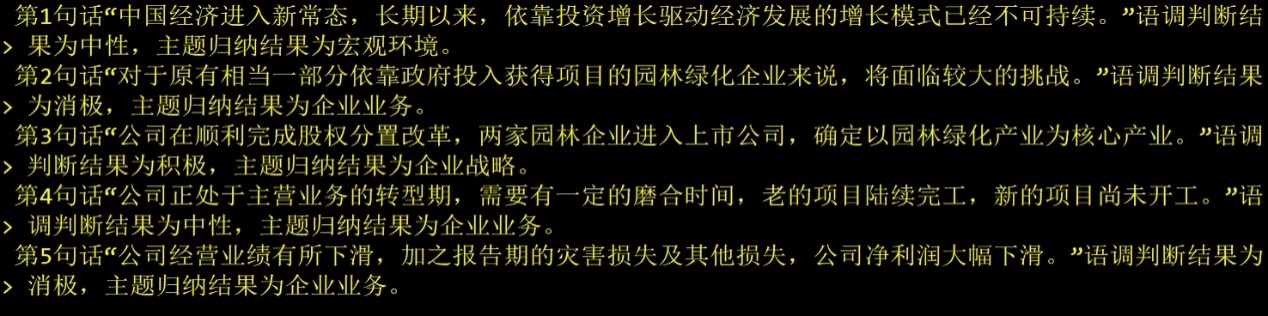


图19 GPT对于文本语调的判断

附录G：使用chatgpt命令进行主题分类

本文以统一股份（股票代码：600506）等五家上市公司资料为例，识别其高管背景。根据放置于云端包含高管个人信息的数据文件“data5.dta”，数据包含2004~2017年共计93名高管的简历，操作流程如下图20所示。

chatgpt命令逐个判断高管类型

高管简历文本文件

读入Stata

提交对高管进行分类的提示词

图20 使用GPT-4模型进行主题分类的流程图

首先，本文结合文献中的高管背景识别原则以及企业信息构建系统提示词。其次，依次提交各个高管的简历内容，命令代码如下所示：

clear all

webuse set "https://stata-chatgpt.oss-cn-beijing.aliyuncs.com/chatgpt"

webuse data5.dta, clear

gen class = ""

global OPENAI\_API\_KEY=fileread(e:/chatgpt/APIKEY.txt)

timer clear 4

timer on 4

forvalue i = 1/`=\_N'{

chatgpt talk, openai\_api\_key($OPENAI\_API\_KEY) command(`"`=resume[`i']'"') systemprompt("你是一名研究助理，我会发给你一份上市公司高管的中文简历，请根据以下定义回复我这名高管的背景类别。\n 1. 土生土长：从始至终一直在公司工作，通过自下而上逐级晋升的方式选拔成为高管。\n 2. 外部空降：原不在该公司工作，而是从外部市场直接聘任为公司高管。\n 3. 半路出家：同时拥有在其他公司工作经历和本公司非高管岗位经历。\n 你在分类时注意以下原则：\n 1. 如果该人曾在该公司的子公司任职，则同样算作该公司履历。\n 2. 高管岗位包括正副总经理、正副总裁、财务总监、董事会秘书或其它相近岗位。\n 3. 简历中的公司可能存在更名、名称简写等情况，请根据简历行文自主判断。\n 请直接回复判断结果，在三种类别中三选其一。") engine("gpt-4")

replace class = fileread("output.txt") in `i'

}

timer off 4

timer list 4

根据上述代码所获得的部分结果如下图21所示。可以发现，在经过人工比对后，chatgpt所提供的分类结果完全正确。值得一提的是，chatgpt还能够智能识别出某些简历的信息含量不足，这直接省去了对于个人履历信息不完整而难以判断的高管样本的处理。

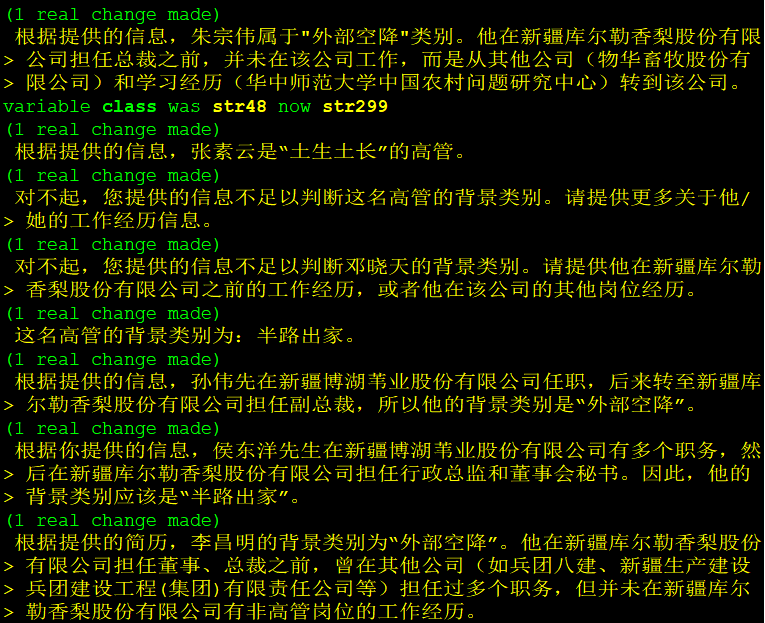


图21 chatgpt基于高管简历对于高管背景分类的结果

附录H：使用chatgpt命令进行精准分词

我们以一段放置于云端的专利摘要语料“content5.txt”为例，展示GPT如何辅助我们扩充专业词典，提高分词精准度，操作流程如下图22所示。

读入Stata

专利摘要文本文件

使用jieba进行分词

对比

读入Stata

专利摘要文本文件

chatgpt命令进行分词

图22 使用GPT-4模型进行文本分词的流程图

首先，我们使用业界通行的jieba分词对中文文本进行分词，其次，我们在Stata中，使用chatgpt命令得到GPT的分词结果，具体命令如下：

clear all

copy "https://stata-chatgpt.oss-cn-beijing.aliyuncs.com/chatgpt/content4.txt" "content4.txt", replace

global OPENAI\_API\_KEY=fileread(e:/chatgpt/APIKEY.txt)

chatgpt read other, openai\_api\_key($OPENAI\_API\_KEY) systemprompt("你是一位中文与英文均精通的专利审查员，以下是一段专利的中文摘要，以及对应的英文摘要，请结合专利内容，注意其中的技术词汇，对中文摘要进行分词工作，词与词之间用空格隔开") engine("gpt-4") file("content4.txt")

完整的分词结果如下表1所示。

表1 jieba原生分词与GPT分词对比

|  |  |
| --- | --- |
| jieba原生分词 | GPT分词 |
| 本 实用新型 涉及 一种 便于 蒸发 结晶器 的 管路 堵塞 清堵 装置 , 包括 **前 引体** , 前 引体 的 外部 设有 锥壳 , 锥壳 上 依次 均匀 的 分布 设有 拨带 和 导槽 , 前 引体 右 末端 设有 接头 , 前 引体 右侧 设有 **扶正 体** , 扶正 体 外部 均匀 的 设有 **导流 带** , 相邻 的 导流 带 之间 设有 **通流 沟** , 导流 带 与 通流 沟均 呈 螺旋状 , 扶正 体 后部 设有 活杆体 , 扶正 体 与 活杆体 之间 设有 转动 接头 , 扶正 体 通过 转动 接头 与 活杆体 连接 。 本 实用新型 采用 前 引体 上 的 拨 带 可以 将 管路 内 的 淤积 物 进行 钻起 , 拨动 下 的 碎物 可以 通过 导槽 进行 向 后 排出 , 通过 扶正 体和活 杆 体内 的 螺旋状 的 通流 沟 可以 将 碎物 不断 的 进行 排出 , 使 管路 可以 快速 的 进行 疏通 , 清堵 的 效果 好 , 清堵 的 操作 方便 , 无需 切断 管路 , 省时省力 , 节省 维护 成本 。 | 本 实用新型 涉及 一种 便于 蒸发结晶器 的 管路 堵塞 清堵装置 , 包括 **前引体** , 前引体 的 外部 设有 锥壳 , 锥壳 上 依次 均匀 的 分布 设有 拨带 和 导槽 , 前引体 右末端 设有 接头 , 前引体 右侧 设有 **扶正体** , 扶正体 外部 均匀 的 设有 导流带 , 相邻 的 **导流带** 之间 设有 **通流沟** , 导流带 与 通流沟 均 呈 螺旋状 , 扶正体 后部 设有 活杆体 , 扶正体 与 活杆体 之间 设有 转动接头 , 扶正体 通过 转动接头 与 活杆体 连接 。 本 实用新型 采用 前引体 上 的 拨带 可以 将 管路 内 的 淤积物 进行 钻起 , 拨动 下 的 碎物 可以 通过 导槽 进行 向后 排出 , 通过 扶正体 和 活杆体 内 的 螺旋状 的 通流沟 可以 将 碎物 不断 的 进行 排出 , 使 管路 可以 快速 的 进行 疏通 , 清堵 的 效果 好 , 清堵 的 操作 方便 , 无需 切断 管路 , 省时省力 , 节省 维护 成本 。 |

附录I：关于基于大语言模型的文本指标可信度高的论述

传统文本指标的可信度概念与准确性相近，但鉴于GPT的输出内容并不能保持完全稳定，本文认为，基于大语言模型的文本指标可信度应该由指标稳定性与指标准确性两方面组成。

文本指标稳定性指的是大预言模型在面对完全相同的问询时，不同输出之间的差异程度。**第一，大语言模型的判断结果具有内在一致性。**尽管由于模型特性，大语言模型面对相同问询时并不总是输出完全相同的回复，但其客观结论总是高度稳定。**第二，大语言模型也可以通过调整参数获得更为稳定的回复结果。**例如，Open AI提供的API服务中有一个参数选项temperature[[3]](#footnote-3)，该参数值处于0到2之间，值越小，返回的GPT回复内容就越稳定，值越大，也并不意味着前后判断相悖，而是输出的内容会更加多样与包容，但基本观点仍一致。**第三，保障模型输出稳定性是双向的事情，既要通过技术手段控制，也尽量不要给GPT过于难以分辨的选择。**例如，尽量避免要求GPT对文本的某种程度给予具体的数值打分，固然GPT可能会在不同轮次给出不同分数，但同样的工作给到两个独立判断的研究者，也很难在数值打分这样差别不大的选择上达成完全一致。

**文本指标准确性。**第一，自ChatGPT应用上线的一年多时间里，各界用户普遍感受到其输出内容较为贴近人的“直觉”，判断结果与人工相差不大。第二，当模型输出不符合观测者“直觉”[[4]](#footnote-4)时，观察者可以向旁人求证，尽量先排除自己的观察偏误，或者考虑特殊的情景设定致使差异存在，并通过prompt将情景设定信息传达给GPT，直至调节到GPT的输出内容基本符合当下的判断场景，提高文本指标准确性。第三，文本指标的准确性从逻辑上可以在更大范围的样本上得到确认。首先，大语言模型面对大量问询的回答均能符合人类直觉，即使存在少量的直觉偏误，也能够通过技术手段调整，这表明文本指标的准确性可以在同一时间的不同样本上得到保证。其次，结合上文关于文本指标稳定性的论述，文本指标的准确性也可以在不同时间的同一样本上得到保证。

总之，本文基于模型可以在广泛样本上保持输出正确性与稳定性，得到大语言模型的文本指标可信度高的结论。

1. 详见https://curl.se/download.html。 [↑](#footnote-ref-1)
2. Zhang X., Xue Y., Li C., 2023, Cntraveltime: Travel Distance And Travel Time in China[J], The Stata Journal, 23(3), 730~753. [↑](#footnote-ref-2)
3. 详见https://platform.openai.com/docs/api-reference/completions/create#temperature [↑](#footnote-ref-3)
4. 应当强调，这里的不符合“直觉”是指GPT给出的判断和观察者的判断存在巨大差别，例如方向相反、形容词相反等情况。而“比较好”与“非常好”之间的差别不足以称之为不符合直觉 [↑](#footnote-ref-4)