单位代码

学 号 ZY2203811

分 类 号



前言语言大模型下游任务性能对比

深度学习与自然语言处理 （NLP） 第五次课后作业

院 （系 ） 名 称 自动化科学与电气工程学院 专 业 名 称 自动化

学 生 姓 名 许铁

2023 年 06 月

1 内容介绍

任意选取3~4个目前前沿语言大模型（不限中文或者英文），通过提示工程的方法来检验和对比不同模型下游任务上的性能，可选择3~5个不同的自然语言下游任务来进行测试。

2 实验原理

2.1 大语言模型

大语言模型（英文：Large Language Model，缩写LLM），也称大型语言模 型，是一种人工智能模型，旨在理解和生成人类语言。它们在大量的文本数据上进行训练，可以执行广泛的任务，包括文本总结、翻译、情感分析等等。LLM的特点是规模庞大，包含数十亿的参数，帮助它们学习语言数据中的复杂模式。这些模型通常基于深度学习架构，如转化器，这有助于它们在各种NLP任务上取得令人印象深刻的表现。

本文选取讯飞星火认知大模型、chatGPT、ChatYuan-large-v2三个LLM进行测试。

（1）ChatYuan-large-v2

ChatYuan-large-v2是阿里巴巴云开发的大型语言生成模型，它可以生成多种语言的高质量响应，包括中文、英文、日文和韩文。

ChatYuan-large-v2使用深度学习技术来分析和理解用户输入的上下文信息，并根据这些信息生成相应的回复。这个模型使用了大规模预训练数据集，并且经过了精细调整和优化，以提高其在各种应用场景下的性能表现。与之前的版本相比，ChatYuan-large-v2具有更高的准确率和更丰富的语言表达能力。它可以生成更加自然流畅的语言输出，并且可以更好地满足用户的需求。

（2）讯飞星火

讯飞星火大语言模型是一种基于T5架构的中文语言模型，主要通过大量无监督中文语料进行训练，采用深度学习算法进行强化学习，具有强大的中文语言理解和生成能力。该模型使用了大量的中文语料进行训练，并且采用了T5架构，在训练过程中只进行少量的无监督学习，以提高模型的训练效率和准确性，同时在模型中采用了了大量的元胞自动机进行微调，以提高模型的鲁棒性和稳定性

（3）chatgpt

ChatGPT是一个由OpenAI开发的大型语言模型，它基于GPT-3.5架构。它通过训练大量的文本数据，具有理解和生成自然语言的能力。ChatGPT可以用于对话系统，它能够理解用户输入的问题或指令，并生成相应的回答或响应。ChatGPT可以处理各种主题和领域的问题，从常见知识到技术性的内容都可以与它进行交流。它还可以提供实时的帮助、建议和解决问题的方法。

2.2 提示工程

提示工程(Prompt Engineering，也称为In-Context Prompting)指在不更新模型参数的前提下、通过输入文本等方法来操控大型语言模型(Large Language Model)以指导其行为、引导其生成我们需要的结果的方法。目前提示工程还处在经验摸索阶段：不同的模型间所需的提升过程方法、以及最终的效果往往会有较大的差异。因此，需要大量实验和启发式的探索

3 实验过程

**3.1** 文本摘要

提示语与结果如下：

（1）

你现在具有生成文字摘要的功能，总结下列文字的主要内容，要求25字之内：

大漠上的风暴呼啸了一夜，直到第二天早晨，才渐渐地平静了下来。霍元龙和陈达海从黄沙之中爬起身来，检点人马，总算损失不大，死了两名伙伴，五匹马。但人人都已熬得筋疲力尽，更糟的是，白马背上的小女孩不知到了何处，十九是葬身在这场大风沙中了。身负武功的粗壮汉子尚且抵不住，何况这样娇嫩的一个小女孩儿。

|  |  |  |
| --- | --- | --- |
| chatgpt | 讯飞星火 | ChatYuan-large-v2 |
| 风暴过后，霍元龙和陈达海检点人马，损失两人五马。小女孩失踪，身负武功的人也无法抵挡。 | 霍元龙和陈达海在大漠风暴中损失了人马，白马背上的小女孩失踪。 | 这段文字描述了两骑马急驰而来，其中一马骑白马，一马骑枣红马。马背上伏著一个高瘦汉子，马背上插著长箭，鲜血从背心流到马背上，又流到地下，滴入黄沙之中。身后有凶悍毒辣的敌人追赶，马被追得筋疲力竭，最后马倒毙，周围尘沙飞扬，有大队敌人追了下来。 |

（2）

你现在具有生成文字摘要的功能，总结下列文字的主要内容，要求25字之内：

我们有一个会计，一个负责购买必需物品，领取教会和警察部门下发物资，兼职车夫的人员，他们都是专业人士，不需要轮换，每周周日休息，剩下的三位文职者，是罗珊，布莱特和老尼尔，他们的工作有：接待来访人员，清理房间，书写案件文档和物品申报清单，以及看守武器、材料和文献库，严格进行出入和领取归还登记，每人每周各自轮休一天，除了周日，另外还有值夜和休息的轮换，都是自己协商

|  |  |  |
| --- | --- | --- |
| chatgpt | 讯飞星火 | ChatYuan-large-v2 |
| 专业人员负责会计、采购和物资管理，兼职车夫。文职者负责接待、清理、文档书写和库房看守。每人每周轮休一天，协商夜班和休息安排。 | 这是一个由专业人士、轮休文职人员和值夜人员组成的团队，负责购买物品、领取物资、接待来访者等任务。 | 总之，这是一个专业人员，负责管理公司的财务和采购，并负责人员聘用和人员培训。 |

结果分析：每一个模型都超出了字数要求，其中针对中文文本生成摘要的任务讯飞星火更加出色。

**3.2** 文本生成

提示语与结果如下：

（1）

你现在是一个诗人，请写一首关于春天的诗歌，该诗歌风格为七律

|  |  |  |
| --- | --- | --- |
| chatgpt | 讯飞星火 | ChatYuan-large-v2 |
| 春风拂面又一年，柳绿花开赏春天。  桃李绽放满园香，春雨润泽万物鲜。  百鸟争鸣枝头醉，游蜂采蜜舞翩跹。  绿意融融满山岭，春光明媚满人间。  山川草木尽苏醒，春水微波涌新情。  明媚天地春光好，满怀希望与梦想。 | 春风拂面柳丝长，  绿意盎然草色香。  桃花盛开红似火，  樱花飘落白如霜。  莺歌燕舞枝头语，  蝶舞蜂飞花间忙。  春雨滋润万物生，  大地回春生机旺。 | 春日阳光明媚，  百花争艳芬芳，  小鸟歌唱欢腾，  春意盎然生机，  万物复苏生机勃勃。  草木萌芽萌发，  花朵绽放绽放，  春风拂面温馨，  万物复苏生机。  春雨滋润万物，  草木生长茁壮，  春意盎然生机，  万物复苏生机。  春风和煦，万物复苏，  春日花开艳丽，  万物复苏生机，  春天，让我们心情愉悦。 |

（2）

帮我写一个假条，请假原因是感冒，假条要包含请假的对象、时间、原因等等，要求150字以内

|  |  |  |
| --- | --- | --- |
| chatgpt | 讯飞星火 | ChatYuan-large-v2 |
| 尊敬的[请假对象姓名]，  我写此假条是希望得到您的谅解和批准，请准予我于[请假起始时间]至[请假结束时间]的请假申请。  我不幸患上了感冒，医生建议我休息并接受治疗以尽快康复。这种疾病严重影响了我的工作效率和集中力，因此我希望能够请假休息，以便能够尽快恢复健康并避免传染给其他同事。  我会确保在请假期间尽快康复，并将在回来后尽快补上因请假而耽误的工作。如果有任何急需处理的事项，我会尽力在请假期间保持通讯畅通。  非常抱歉给您带来不便，请您批准我的请假申请。我将非常感激您的理解和支持。  谢谢！  [您的姓名]。 | 尊敬的XXX老师：  您好！我是您班上的学生XXX。由于我近日感冒发烧，身体状况不佳，无法集中精力学习。因此，我特此向您请假一天，以便及时休息恢复健康。  希望您能够批准我的请假申请。我会尽快补上今天的课程内容，并保证不会影响到我的学习进度。感谢您的理解和支持！  此致  敬礼！  学生 XXX  日期：XXXX年XX月XX日 | 尊敬的XXX领导：  我因感冒症状严重，需要请假一天，敬请批准。请假时间为XXXX年XX月XX日。  请假人：XXX  请假时间：XXXX年XX月XX日  请假事由：感冒  请假期间，我将积极配合治疗，保持身体健康，为公司做出更大的贡献。  谢谢领导的谅解与支持。  此致  敬礼!  XXX  XXXX年XX月XX日情愉悦。 |

结果分析：ChatYuan模型在编写七律诗时与请假条内容出现了紊乱，效果相比其他两个模型较差。

**3.3** 代码生成

提示语与结果如下：

使用python写一个六自由度模块，输入位置、速度、姿态角等等，输出为下一时刻的飞机状态

|  |  |  |
| --- | --- | --- |
| chatgpt | 讯飞星火 | ChatYuan-large-v2 |
| import numpy as np  # 定义飞机模型参数  mass = 1000.0 # 质量 (kg)  inertia = np.diag([1000.0, 2000.0, 1500.0]) # 转动惯量矩阵 (kg·m^2)  g = np.array([0, 0, -9.81]) # 重力加速度 (m/s^2)  # 定义时间步长  dt = 0.01 # 10 ms  def update\_state(position, velocity, attitude, angular\_rate):  # 计算加速度  acceleration = np.dot(attitude, np.array([0, 0, -1])) \* (1 / mass) \* np.array([0, 0, 1]) + g  # 更新速度和位置  velocity += acceleration \* dt  position += velocity \* dt  # 计算转动惯量矩阵  inertia\_tensor = np.dot(np.dot(attitude, inertia), np.linalg.inv(attitude))  # 计算角加速度  angular\_acceleration = np.dot(inertia\_tensor, angular\_rate)  # 更新角速度和姿态角  angular\_rate += angular\_acceleration \* dt  attitude += np.cross(angular\_rate, attitude) \* dt  return position, velocity, attitude, angular\_rate  # 示例用法  initial\_position = np.array([0, 0, 0]) # 初始位置 (m)  initial\_velocity = np.array([0, 0, 0]) # 初始速度 (m/s)  initial\_attitude = np.array([1, 0, 0, 0]) # 初始姿态角 (四元数)  initial\_angular\_rate = np.array([0, 0, 0]) # 初始角速度 (rad/s)  # 循环模拟多个时间步  for \_ in range(10):  # 更新状态  initial\_position, initial\_velocity, initial\_attitude, initial\_angular\_rate = update\_state(  initial\_position, initial\_velocity, initial\_attitude, initial\_angular\_rate  )  # 打印下一时刻的飞机状态  print("Position:", initial\_position)  print("Velocity:", initial\_velocity)  print("Attitude:", initial\_attitude)  print("Angular Rate:", initial\_angular\_rate)  print("-" \* 20) | import numpy as np  class SixDoFModule:  def \_\_init\_\_(self, position, velocity, attitude\_angle):  self.position = np.array(position)  self.velocity = np.array(velocity)  self.attitude\_angle = np.array(attitude\_angle)  def update\_position(self, acceleration, dt):  self.position += self.velocity \* dt + 0.5 \* acceleration \* dt\*\*2  def update\_velocity(self, acceleration, angular\_acceleration, dt):  self.velocity += self.position \* dt + 0.5 \* acceleration \* dt\*\*2  self.attitude\_angle += angular\_acceleration \* dt  def update\_attitude\_angle(self, angular\_acceleration, dt):  self.attitude\_angle += angular\_acceleration \* dt  def get\_next\_state(self):  return {  'position': self.position.tolist(),  'velocity': self.velocity.tolist(),  'attitude\_angle': self.attitude\_angle.tolist()  }  # 示例  position = [1, 2, 3]  velocity = [0, 0, 0]  attitude\_angle = [0, 0, 0]  acceleration = [0, -9.81, 0]  angular\_acceleration = [0, 0, 0]  dt = 0.1  module = SixDoFModule(position, velocity, attitude\_angle)  for \_ in range(10):  module.update\_position(acceleration, dt)  module.update\_velocity(acceleration, angular\_acceleration, dt)  module.update\_attitude\_angle(angular\_acceleration, dt)    next\_state = module.get\_next\_state()  print(next\_state)  ``` | import math  # 初始化六自由度模块 my\_vx, my\_vy, my\_z = 4.0, 4.0, 4.0  # 初始化六自由度坐标系 x\_z = math.cos(my\_vx) y\_z = math.sin(my\_vy) z\_z = math.tan(my\_z)  # 定义六自由度坐标系 x\_z = x\_z \* my\_vx y\_z = y\_z \* my\_vy z\_z = math.tan(x\_z)  # 定义六自由度角坐标系 x\_r = math.sin(my\_vx) y\_r = math.cos(my\_vy) z\_r = math.tan(my\_z), x\_r  # 初始化六自由度坐标系和六自由度角坐标系 x\_v = x\_z \* my\_z y\_v = y\_z \* my\_z z\_v = math.tan(x\_z), y\_z  # 初始化六自由度坐标系的位置 x\_z\_v = math.cos(my\_v) y\_z\_v = math.sin(my\_v) z\_z\_v = math.tan(x)  # 初始化六自由度坐标系的坐标系 x\_v = x\_v \* my\_v y\_v = y\_z \*\* 2 z\_v = math.tan(y), x\_v  # 初始化六自由度坐标系之间的距离 x\_r\_v = math.cos(my\_z) y\_r\_v = math.sin(my邪恶) z\_r\_v = math.tan(y)  # 初始化六自由度坐标法方程 x\_v\_c = math.cos(my\_v)\*x\_z y\_v\_c = math.sin(my邪恶)\*y\_z z\_v\_c = math.tan(y), y\_z  # 初始化的六自由度坐标系和六自由度角坐标生成器 x\_v\_c = math.sin2(x\_z\_v) y\_v\_c = math.cos2(y\_z\_v) z\_v\_c = math.sin3(z\_z\_v)  # 初始化六自由度坐标轴 x\_v\_c = math.tan2(x\_z) y\_v\_c = math.10 z\_v\_c = math.cos(x\_v)  # 初始化六自由坐标系和六自由度角坐标法方程 x\_v\_c = x\_v\_c \* my\_v y\_v\_c = math.tan2xy(z\_v\_c) z\_v\_c = math.10  # 初始化六自由度坐标系与六自由度角坐标系之间的转换 x\_v\_c = math.ceil(x\_v) y\_v\_c = y\_v\_c z\_v\_c = math.ceil2xy(z\_v\_c)  # 初始化六自由度坐标系统 x\_v\_c = my\_v y\_v\_c ≈ x\_z\_v z\_v\_c ≈ y\_z\_v  # 初始化六自由度量 x\_v\_c = math.pi/2 y\_v\_c = y\_z\_v z\_v\_c = math.pi/3  # 初始化六自由度角坐标系 x\_v\_c = math.cancel(x\_v) y\_v\_c=math.cancel(y\_v) z\_v\_c = my\_v  # 初始化六自由度角坐标 x |

结果分析：针对这一代码生成任务，chatgpt与讯飞星火都能成功生成，在满足输入输出要求的同时有详实的注释与命名规范，chatyuan更像是把六自由度词汇相关内容进行了拼凑，无法使用。

**3.4** 情感分析

提示语与结果如下：

判断一段文本的情感倾向：

弱到爆 剧情、氛围烘托、表演、服化道、特效，统统弱到爆！女主和哥哥转折超级无力，给女主默默烫发来表现阶段性转折也是醉了…….，除了小迅猛龙看起来像狗以外，其余真是……..吐槽无力

|  |  |  |
| --- | --- | --- |
| chatgpt | 讯飞星火 | ChatYuan-large-v2 |
| 根据提供的文本，可以判断情感倾向为负面或者消极。以下是对文本情感的分析：  文本中使用了一系列消极的词语和表达，如"弱到爆"、"无力"、"醉了"、"吐槽无力"等，这些词语传达了作者对剧情、氛围烘托、表演、服化道和特效等方面的不满和失望。  提到女主和哥哥的转折，作者认为它们表现得超级无力，女主烫发来表达阶段性转折更加令人困惑。另外，对除了小迅猛龙之外的其他方面也持消极态度，认为它们都很差劲。  综上所述，该文本传达了一种负面或消极的情感倾向，作者对电影或剧集的各个方面感到失望和不满。 | 情感倾向：负面 | 情感倾向：弱到爆 |

结果分析：三者均成功进行了情感判断，但是chatgpt能够合理的进行分析，生成更多信息。

4 总结

针对文本代码生成、情感分析、文本摘要这四个下游任务，三个模型均能成功生成结果，chatgpt与讯飞星火模型的回答更加实用，其中讯飞星火模型比较简洁，无论是从生成速度、准确性都超出ChatYuan大语言模型。

本报告对比了前沿NLP大模型在各种下游任务的性能，总体而言，讯飞星火以及 ChatGPT 模型在大多数任务上具有较高的准确率。然而，在实际应用时，还需考虑因素如计算资源、微调数据量等，并根据项目需求选择合适的模型。