**项目详细方案**

## 一、项目执行思路介绍

本文选择论文《[Robust High-Resolution Video Matting with Temporal Guidance](https://arxiv.org/pdf/2108.11515.pdf" \t "_blank)》提出的人物分割模型RH-RVM作为算法基础框架。针对高清视频中人物分割问题RH-RVM算法利用帧间信息进行隐式学习，增加了分割结果在时序上的连贯性。但是RH-RVM依然是针对GPU设计的模型，无法在CPU中实现实时处理。因此本文对RH-RVM网络模型进行分析，将RH-RVM中的主干网络从movilenetv3-large更改为movilenetv3-large-0.75,使用数据集进行训练，训练指标达到，略低于论文指标。

部署时选择c++语言和[OpenVINO](https://link.zhihu.com/?target=https%3A//software.intel.com/en-us/openvino-toolkit" \t "_blank)推理框架进行推理工程搭建。[OpenVINO](https://link.zhihu.com/?target=https%3A//software.intel.com/en-us/openvino-toolkit" \t "_blank)是英特尔推出的一款全面的工具套件，最适用于英特尔处理器快速部署应用和解决方案。测试集分割效果如下：

## 二、详细技术方案

### ２.１业务需求

本次任务需要实现一可以支持至少两轮问答，具备大模型的基础问答能力和基于语料库的问答能力，具有交互界面，可通过浏览器进行访问，页面的访问延迟和问答的响应延迟不超过10s的模型。本次数据集选择**三大江南古镇旅游目的地形象数据集，精品旅行服务数据解析，HI GUIDES旅游数据分析**。

**（１）三大江南古镇旅游目的地形象数据集**

数据集包括乌镇、西塘和周庄三个古镇的新闻文本、官方微信公众号文本和网络游记文本。总数据量：11507条。新闻文本：来自中国重要报纸全文数据库，相关新闻272篇；官方微信公众号文本：来自“乌镇景区”、“西塘旅游”和“周庄旅游”三个公众号，相关文章4136篇；网络游记文本：来自马蜂窝游记，相关游记7099篇

**（２）精品旅行服务数据解析**

**精品旅行服务数据解析**通过SQL工具对约5万用户的数据进行了探索，以回答以下几个问题：哪些国家或城市是比较受欢迎的？哪些城市的用户更喜欢境外旅游？在哪些时间段是下单的高峰期？男女生是否对国家或城市有不同的偏好？数据集的特点包括：时间为UTC时间，需要转换提取。使用Excel进行预处理，如删除userComment\_train表中关于commentsKeyWords的内容。数据导入Navicat后，可以看到action\_train表中有超过130万的数据信息。数据集的优点体现在：能够通过数据分析揭示旅游偏好和行为模式。提供了详细的历史订单分析，如订单总数、最受欢迎的城市、VIP服务的选择等。分析了注册用户的性别比例和前往不同国家的订单数分布情况。

**（３）HI GUIDES旅游数据分析**

**HI GUIDES旅游数据分析**主要探讨了2017年中国公民出境旅游的数据。文章通过对大数据的分析，提出了几个关键问题，并使用SQL和其他工具来解答这些问题。以下是文章的主要内容和数据集的特点以及优点的总结：文章主要内容：提出问题：分析了哪些国家城市受游客青睐，游客群体的特征，以及游客出游的时间段。数据整理与导入数据库：使用Navicat Premium和Notepad++等工具进行数据整理和导入。数据分析：通过查询历史订单表，分析了亚太地区旅游目的地的受欢迎程度，以及游客的性别和年龄分布。数据集的特点：数据量大，包含133万条客户行为信息。数据涵盖了2016年11月至2017年9月的订单数据。数据集的优点：消费者画像：揭示了中国出境游游客的画像，包括性别、年龄、客源地城市分布。旅游趋势：分析了出境游的时间趋势，如4-7月和11-12月为出游高峰期。旅游目的地偏好：展示了亚太地区作为主要旅游目的地的情况，以及欧美地区的跟随趋势。

### ２.２模型调研

### ChatGLM2-6B模型推理架构和流程：

### 模型流程

### 可以看到，这个模型在推理阶段主要由两层循环组成。 第一层是while true循环，每循环一次生成一个next token，退出条件是模型生成了<eos>这个token，就是结束符。 第二层循环是固定28次的for循环，对GLMBlock顺序运行28次，根据的attention scores得到最有可能的token id。

### 模型推理流程：

### 1.输入与分词、编码：输入文本首先经过分词和编码处理，转换为模型能够理解的格式。

### 2.嵌入（Embedding）：文本编码后的数据通过嵌入层转换为具有固定特征空间的向量表示。

### 3.GLMBlock\*28：模型包含28个GLMBlock，每个GLMBlock都会对输入数据进行处理，以生成下一个token的预测。

### 4.生成next token：模型通过循环迭代，每次生成一个token，直到生成结束符为止。

### 模型架构概要：

### 1.ChatGLM2-6B是基于GLM大模型的下游对话应用，采用了Prefix Decoder-only架构。

### 2.结合了单项注意力和双向注意力的优点，前者用于生成式任务，后者用于获取更全面的信息。

### 3.模型优化了空白填充预训练，通过二维位置编码技术改进了自然语言理解任务的性能。

### ２.３模型部署

### 计算资源：GTX 3070Ti Laptop

### 部署环境：python 3.11.4，CUDA 1.11.8，CUDNN 1.11 Torch1.12.1

### 项目克隆：

### 

### 安装依赖

### 

### 安装GIT LFS

### 

### 从Hugging Face Hub下载模型

### 

### 模型量化（默认情况下，模型以FP16精度加载，需要约13GB显存。由于现存有限，所以采用模型量化）

### 

### 这里插入一些对话图片

### 6由于要对应自己的前端页面，选择使用gradio+flask的形式重新在本地部署

### 

### 

### CORS支持：为Flask应用添加了CORS支持，这对于前端和后端跨域请求是必要的。模型和分词器的加载：您指定了模型和分词器的路径，并且使用trust\_remote\_code=True参数来加载远程代码。API端点：定义了一个POST方法的/api/predict端点来处理预测请求。处理流程包括获取用户输入、编码输入、生成预测结果以及将结果转换为文本。模型预测：在生成预测结果时，使用了max\_length来限制输出的长度。top\_p和temperature参数，这些参数可以用来控制生成文本的多样性。启动应用：在0.0.0.0的地址和8000端口上运行Flask应用。

### 

### 

### 

### 

模型和分词器的加载：加载了模型和分词器，并将模型放到了GPU上以提高性能。postprocess方法：重写了Gradio Chatbot的postprocess方法，这通常用于自定义输出的格式。预测函数：定义了gradio\_predict函数来处理输入并生成响应。这个函数使用了stream\_chat方法，它是一个自定义的生成响应的方法用户状态的重置：提供了两个函数reset\_user\_input和reset\_state来重置用户输入和状态，这有助于在交云对话中清除历史记录。Gradio界面：使用Gradio Blocks创建了界面，并添加了Markdown、Chatbot、Textbox、Button和Slider组件。这些组件为用户提供了一个友好的交互方式。启动界面：最后，使用demo.launch()来启动Gradio界面。



最后在前端html文件中用javaScript代码处理用户的输入和输出，通过fetch API与后端进行通信。

### 这里插入一些对话图片

### 2.3模型微调

### 基于 P-Tuning v2 的微调

### Tuning v2 将需要微调的参数量减少到原来的 0.1%，再通过模型量化、Gradient Checkpoint 等方法，最低只需要 7GB 显存即可运行。

### 将预处理好的数据集文件放在ptuning 目录下

### PRE\_SEQ\_LEN 和 LR 分别是 soft prompt 长度和训练的学习率，可以进行调节以取得最佳的效果。P-Tuning-v2 方法会冻结全部的模型参数，可通过调整 quantization\_bit 来被原始模型的量化等级，不加此选项则为 FP16 精度加载。

### 在默认配置 quantization\_bit=4、per\_device\_train\_batch\_size=1、gradient\_accumulation\_steps=16 下，INT4 的模型参数被冻结，一次训练迭代会以 1 的批处理大小进行 16 次累加的前后向传播，等效为 16 的总批处理大小，此时最低只需 6.7G 显存。若想在同等批处理大小下提升训练效率，可在二者乘积不变的情况下，加大 per\_device\_train\_batch\_size 的值

### 

### 2.4测评结果

### 我们选取了部分中英文典型数据集进行了评测，以下为 ChatGLM2-6B 模型在 MMLU (英文)、C-Eval（中文）、GSM8K（数学）、BBH（英文） 上的测评结果。在 evaluation 中提供了在 C-Eval 上进行测评的脚本。

### 

### 

### 

### ChatGLM2-6B 使用了 Multi-Query Attention，提高了生成速度

### 

### 

## 三、项目优势与劣势

优势：

模型选择最近2020年由论文《Real-Time High-Resolution Background Matting》提出的RH-RVM模型。主要考虑如下几点：

1）更强大的性能：基于 ChatGLM 初代模型的开发经验，我们全面升级了 ChatGLM2-6B 的基座模型。ChatGLM2-6B 使用了 GLM 的混合目标函数，经过了 1.4T 中英标识符的预训练与人类偏好对齐训练，评测结果显示，相比于初代模型，ChatGLM2-6B 在 MMLU（+23%）、CEval（+33%）、GSM8K（+571%） 、BBH（+60%）等数据集上的性能取得了大幅度的提升，在同尺寸开源模型中具有较强的竞争力。

2）更长的上下文：基于 FlashAttention 技术，我们将基座模型的上下文长度（Context Length）由 ChatGLM-6B 的 2K 扩展到了 32K，并在对话阶段使用 8K 的上下文长度训练。对于更长的上下文，我们发布了 ChatGLM2-6B-32K 模型。LongBench 的测评结果表明，在等量级的开源模型中，ChatGLM2-6B-32K 有着较为明显的竞争优势。

3）更高效的推理：基于 Multi-Query Attention 技术，ChatGLM2-6B 有更高效的推理速度和更低的显存占用：在官方的模型实现下，推理速度相比初代提升了 42%，INT4 量化下，6G 显存支持的对话长度由 1K 提升到了 8K。

4）更开放的协议：ChatGLM2-6B 权重对学术研究完全开放，在填写问卷进行登记后亦允许免费商业使用

劣势：

1）模型结构过于复杂无法在CPU实时处理

2）不利于工程化部署

3）训练源码虽然公开，但是复现时无法达到论文著称精度等

针对上述问题，本文从以下几点进行模型优化：

1. 使用[fastllm](https://github.com/ztxz16/fastllm/): 全平台加速推理方案，单GPU批量推理每秒可达10000+token。纯c++实现，便于跨平台移植，可以在安卓上直接编译，ARM平台支持NEON指令集加速，X86平台支持AVX指令集加速，NVIDIA平台支持CUDA加速，各个平台速度都很快，支持浮点模型（FP32), 半精度模型(FP16), 量化模型(INT8, INT4) 加速，支持多卡部署，支持GPU + CPU混合部署，支持Batch速度优化，支持并发计算时动态拼Batch，支持流式输出，很方便实现打字机效果，支持python调用，前后端分离设计，便于支持新的计算设备
2. 增加数据。除了使用三大江南古镇旅游目的地形象数据集，还增加了DC竞赛中，精品旅行服务数据解析，HI GUIDES旅游数据分析，数据分析告诉你，国庆七天大家都喜欢去哪儿玩？以及SQL数据分析-途牛网情侣5日游四个数据集。
3. 训练中尝试不同参数，例如trust\_remote\_code，pre\_seq\_len以及使用模型生成预测结果时的max\_length，do\_sample，top\_p=以及temperature等参数