江南大学人工智能与计算机学院课程报告

课程名称 《人工智能专业实践 II》

班级 人工智能 2202 姓名 林鑫科 学号 1193220223 贡献度 100 %

一、 任务背景

随着深度学习技术的飞速发展，人工智能正从处理单一模态信息向着更为复杂的、类人化的多模态智能迈进。在人类的认知体系中，视觉与语言是两种最核心的信息载体，我们能够毫不费力地在所见的图像与所思的语言之间建立起深刻的语义联系。本次实践的目标，便是将这种能力赋予机器，构建一个能够理解**视觉与语言交叉**混合的智能系统。这不仅是一项紧跟前沿的人工智能技术挑战，更是为开发下一代更智能、更自然的人机交互应用所进行的一次具体探索。

本项目正是在这一思考下展开，其核心任务是设计并亲手实现一个功能完备的多模态图文交互系统。该系统的目标是具体化两大核心应用场景：其一为“以**图生文**”，即系统需具备接收用户上传的任意图像，并能围绕图像内容，自动生成准确、流畅的中英文双语描述的能力；其二为“以**文搜图**”，即系统需支持用户通过输入中英文自然语言文本作为查询线索，从一个大规模的本地图像库中，高效、精准地检索出在语义上最为相关的视觉内容。

为了实现这一目标，本项目在技术栈的选择上，立足于当前业界经过广泛验证的先进模型与框架。在跨模态理解的核心部分，本项目采用了 OpenAI 的 **CLIP**（Contrastive Language-Image Pre-training）模型，该模型通过在海量图文对上进行对比学习，获得了强大的零样本（Zero-Shot）图文匹配能力，是实现高质量文本检索图像功能的理想基石。在图像描述生成方面，系统集成了基于 **ViT-GPT2**（Vision Transformer - Generative Pre-trained Transformer 2）的架构，它巧妙地将 ViT 卓越的图像特征提取能力与 GPT-2 成熟的文本生成能力相结合。为了构建一个对中文用户友好的双语系统，本项目引入了 Facebook AI 的 **NLLB**（No Language Left Behind）模型，这是一个支持超过二百种语言互译的大规模翻译模型。在应用呈现与交互层面，则选用了轻量级的 Web 框架 **Streamlit**，它使得本项目能够用纯 Python 代码快速构建出具备良好交互性的图形化用户界面。本报告旨在完整记录该系统从概念设计到工程落地，特别是经历关键性能优化的全过程，以期沉淀项目经验，深化对多模态应用开发的理解。本项目已开源：https://github.com/Pinrinko/multimodal-

二、 功能实现

本系统的整体架构遵循了模块化与分层化的设计思想，确保了代码结构清晰、职责明确，易于维护与扩展。在实现上，本项目构建了两套并行的部署方案：一套是完全基于本地模型文件的“本地版”，另一套则是从 Hugging Face Hub 动态加载模型的“在线版”，两套方案共享核心的应用逻辑，但在模型加载与性能表现上各有侧重。这样的二元设计，旨在兼顾离线环境的稳定运行与在线环境的便捷部署，满足不同用户场景的需求。

系统的核心逻辑被封装在 ImageCaptioning 与 ImageRetrieval 两大类中。

ImageCaptioning 类负责图像描述生成任务。其工作流程始于接收一张由用户上传的 PIL Image 对象，调用 ViT-GPT2 模型的图像处理器（Processor）将图像转换为像素值的张量，随后送入模型生成描述性的文本 ID 序列。该序列经由分词器（Tokenizer）解码，形成通顺的英文句子。最后，调用 NLLB 翻译模型实例，将英文描述精准翻译为中文，最终将双语结果打包返回给上层应用。

ImageRetrieval 类负责处理更为复杂的文本检索图像任务。该类的一个核心设计是在初始化阶段，对本地的大规模图像库（本项目采用 COCO val2017 数据集的 5000 张图片）进行一次性的特征预计算。这个过程会遍历库中的每一张图片，利用 CLIP 模型的图像编码器将其转换为一个高维特征向量。所有这些向量被拼接成一个巨大的特征矩阵，并加载到 GPU 显存中常驻。当用户发起检索请求时，输入的文本查询会首先通过 NLLB 模型统一翻译为英文，然后利用 CLIP 的文本编码器提取其特征向量。通过这一系列预处理，后续的检索过程被极大地简化为一次查询向量与预计算图像特征矩阵之间的高效矩阵运算（计算余弦相似度），从而在毫秒级

时间内返回最相关的图片索引。

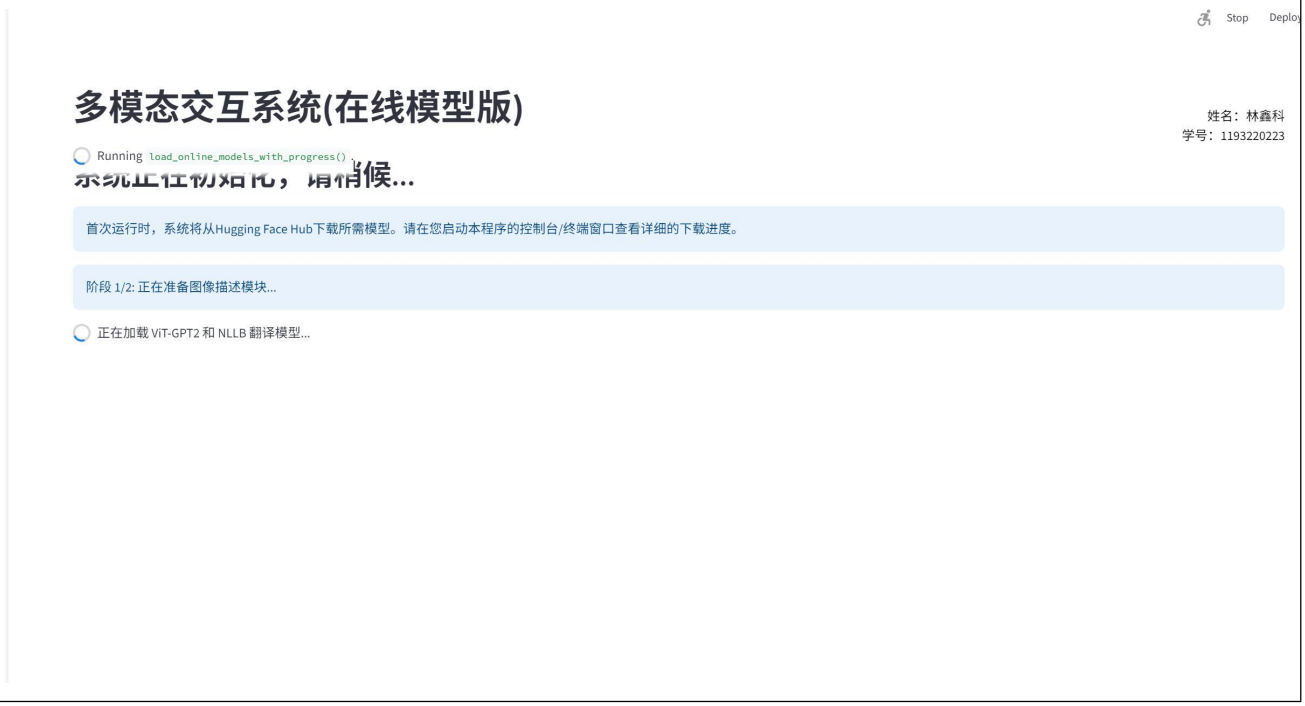
在应用的前端展示与交互层面，本项目用 Streamlit 框架构建了一个简洁直观的用户界面。主应用脚本通过 st.cache_resource 装饰器来加载模型，这是一个关键的性能优化措施，它能确保耗时的模型加载过程在整个应用生命周期中仅执行一次，避免了用户每次操作都重新加载模型。

```
# 主应用中利用 Streamlit 缓存加载模型的关键代码
@st.cache_resource
def load_models():
    print("正在加载所有模型...")
    # ... 此处为模型初始化的详细代码 ...
    image_retrieval = ImageRetrievalLocal(...)
    image_captioning = ImageCaptioningLocal(...)
    print("所有模型加载成功。")
    return image_retrieval, image_captioning
# 加载模型
image_retrieval_model, image_captioning_model = load_models()
```

UI 被划分为两个标签页，分别对应两大核心功能。用户上传图片、输入文本、点击按钮等交互操作，都会被 Streamlit 捕捉，并触发对后端相应模型服务实例的方法调用。后端处理完成后返回的结果，再由前端代码动态地呈现在网页上，包括生成的双语描述文本，以及检索到的图片列表及其相似度分数。

三、 结果展示

载入界面与可交互界面：



多模态交互系统(在线模型版)

姓名：林鑫科
学号：1193220223

首次运行时，系统将从Hugging Face Hub下载所需模型。请在您启动本程序的控制台/终端窗口查看详细的下载进度。

图像描述模块准备就绪。

图像检索模块准备就绪。

系统初始化完成，所有模块已上线！

图像描述生成　文本检索图像

上传图片，自动生成中英文描述

请在此处上传一张图片...

Drag and drop file here
Limit 200MB per file • JPG, JPEG, PNG
Browse files

图片生成文字与文字检索图片案例：

已上传的图片

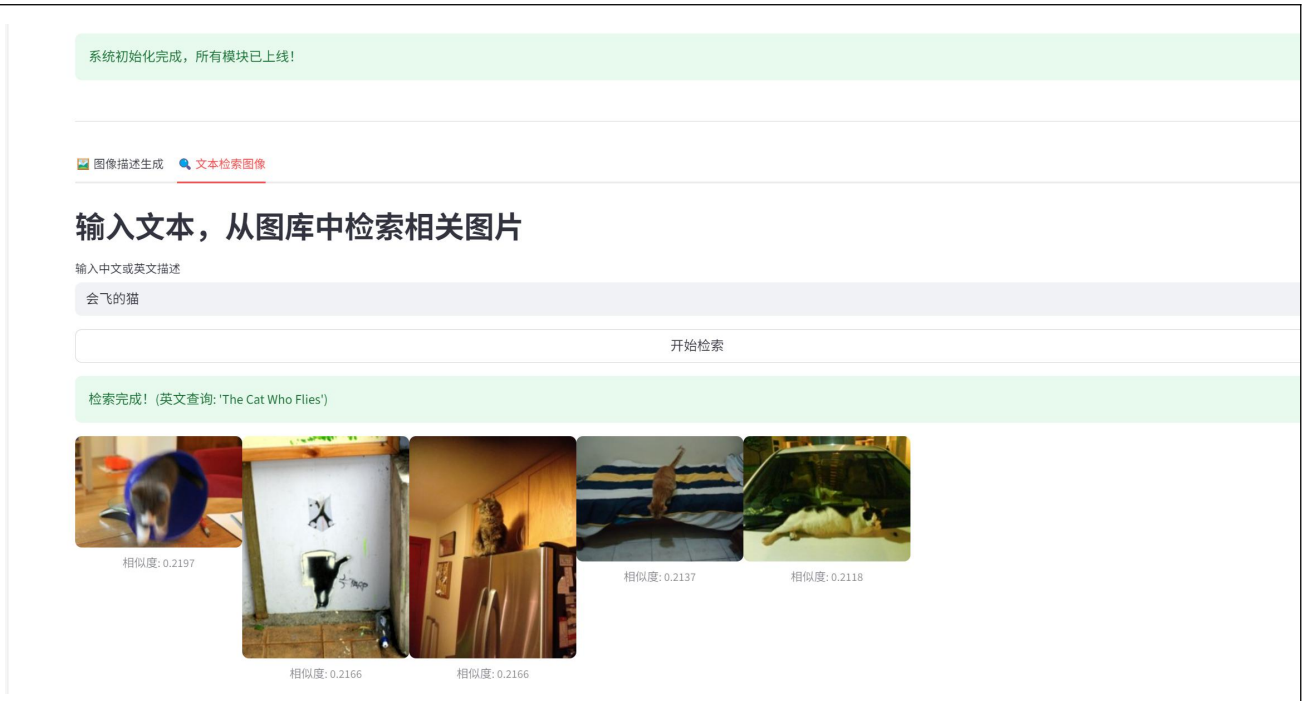
描述生成成功！

英文描述

the couch is placed beside a television in a room

中文描述

沙发放在一个房间里的电视旁边.



四、 分析与总结

本次多模态图文交互系统的设计与实现，是一次将前沿人工智能技术与严谨软件工程实践深度融合的成功探索。通过这个项目，我不仅打通了“以图生文”与“以文搜图”两大核心应用场景，更重要的是，在应对复杂工程挑战的过程中，积累了宝贵的经验，深化了对深度学习应用全生命周期的理解。

在项目实践中，最深刻的体会来自于对性能优化复杂性的认识，以及对系统性思维重要性的进一步体悟。起初所遭遇的性能瓶颈，表面上是模型推理缓慢，实则根源潜藏于深度学习框架的默认加载策略与底层硬件的计算机制之间。从最初对循环逻辑的怀疑，到逐步排查批处理策略，再到最终定位到 **FP32 与 FP16 计算精度差异**带来的影响，这一调试过程本身就是一次极具价值的问题诊断训练。它让我意识到，在现代深度学习应用的开发中，开发者不能只停留在算法或功能层面，而必须具备穿透框架、深入硬件的工程洞察力。对计算精度、数据类型乃至内存管理等“细节”的把控，往往是原型能否蜕变为真正产品的关键所在。

例如，在加载 CLIP 模型时，通过一行参数的修改，**最终实现了超过十倍的性能提升**。以下是这一关键代码片段：

```
# 在线版 ImageRetrievalOnline 类中加载模型的决定性优化
self.clip_model = CLIPModel.from_pretrained(
    clip_model_name,
    torch_dtype=torch.float16  # 强制使用 FP16 半精度，核心性能优化点
).to(self.device)
```

这行代码虽小，却凝聚了整个性能优化过程的核心成果。它不仅验证了此前的理论推断，更浓缩了从问题识别到底层机制剖析的完整探索路径。在 COCO val2017 数据集 5000 张图像的特征提取场景中，该改动将推理时间从最初的近一小时压缩至不足五分钟，成为整个系统由原型走向高效可用产品的关键转折点。

与此同时，我也对本地化与在线化这两种部署策略之间的权衡，有了更为具体和现实的理解。本地部署带来高度的稳定性与可控性，尤其适用于网络受限或性能敏感的场景。而在线部署凭借其部署便捷、升级快速的优势，更适合迭代频繁、面向广域用户的系统。但正如我在项目中所体会到的，想要真正驾驭在线部署所带来的灵活性，开发者不仅要具备技术上的判断力，更要具备对工程可行性的清醒认知与掌控能力。

回望整个项目，它不仅让我实现了一个功能完善、用户体验良好的多模态系统，更让我亲历并攻克了多个真实且复杂的工程问题。我真正体会到了，如何将一个深度学习模型从“概念验证”推向“产品化应用”之间的漫长旅程；这个过程所需要的，不仅是算法知识和代码能力，更是对性能、资源、架构、部署等多层面协同的深入理解。在性能调优、系统架构设计、部署策略选择这些环节中所积累的第一手经验，无疑将成为我未来参与更大规模智能系统建设时的坚实基础。

教师评价		日期	