一、项目概述

1.1 项目背景与目标

随着人工智能技术的发展，单一模态的语言或视觉模型已难以满足复杂场景下的智能交互需求。视觉问答（Visual Question Answering, VQA）作为跨模态智能的核心任务，要求模型同时理解图像内容与文本问题，并生成精准的文本回答，在智能客服、自动驾驶、人机交互等领域具有广泛应用前景。本项目作为 LLM 课程大作业，旨在基于现有预训练模型，构建一套端到端的多模态 VQA 系统，实现图文信息的深度融合与高效推理，具备良好的问答准确性与泛化能力。

1.2 核心技术路线

本项目采用“视觉特征提取→特征维度对齐→图文融合建模→自回归文本生成”的端到端技术路线，整合 SigLIP 视觉模型与 Qwen 语言模型的优势，通过模块化设计实现各环节的高效协同。整体流程可概括为：首先通过视觉编码器提取图像的语义与细节特征，再通过投影层完成视觉特征与文本特征的维度对齐，随后将对齐后的视觉特征注入语言模型实现图文融合，最终由语言模型以自回归方式生成符合问题需求的回答。

二、数据集处理

2.1 数据集选型

为确保模型训练数据的多样性与场景覆盖度，本项目选用两类主流公开数据集，兼顾数据格式的灵活性与标注的规范性，具体信息如下表所示：

数据集类别	具体名称	数据特点	标注格式
阿里系数据集	AI Challenger	字段格式灵活，图文配对丰富，覆盖多场景日常问答	多字段标注，支持 caption 与 conversations 双格式
COCO 数据集	COCO 2017（子数据集：complex_reasoning_77k.json、	图像质量高，标注精准，涵盖复杂推理与细节描述类问题	固定对话式标注，含明确问题与答案配对

	detail_23k.json）		

2.2 数据预处理流程

由于两类数据集的标注格式存在差异，需通过标准化预处理转化为统一格式，为后续模型训练提供一致输入。预处理核心脚本为`process_image.py`，整体流程分为配置读取、分数据集处理、数据合并保存三个步骤，具体实现如下：

2.2.1 配置与数据读取

首先通过 YAML 解析工具读取`config.yaml`配置文件，获取各类数据集的存储路径、输出文件路径及核心参数，确保程序可配置性。随后调用封装的`read_json`函数，分别读取 AI Challenger 数据集与两个 COCO 子数据集的原始 JSON 标注数据，同时加载对应图像文件路径信息，为后续处理奠定基础。

2.2.2 分数据集标准化处理

针对不同数据集的格式特点，设计专属处理函数，实现标注信息的标准化提取与格式统一：

（1）AI Challenger 数据集处理（`process_ali`函数）：考虑到该数据集字段格式灵活，支持`image_id/image`、`caption/conversations`等多字段组合，处理时从多标注列表中随机选取一句标注，以 50%概率切换两种预设提问模板，生成统一格式的问答对。同时建立图像名与图像信息（全局唯一 ID、存储路径）的映射关系，图像 ID 从 0 开始递增，确保全局唯一性。

（2）COCO 数据集处理（`process_coco`函数）：针对其固定对话式标注格式，直接提取问题文本与答案文本，将问题中的``占位符统一替换为`<<|extra_0|>`，使图像占位符与后续语言模型的输入规范对齐；答案部分封装为列表格式，便于模型训练时的标签匹配。为避免与 AI Challenger 数据集的图像 ID 冲突，基于前序数据集的图像数量设置`start_ID`，保证两类数据集图像 ID 不重复。2.2.3 数据合并与保存通过 Python 字典解包语法，合并两类数据集的图像信息映射表与问答标注数据，统计并打印各数据集规模及总数据量，便于数据分布分析与训练参数调整。最终以 UTF-8 编码、格式化输出方式，将合并后的图像信息数据与问答标注数据分别保存为新的 JSON 文件，生成可直接用于模型训练的规整数据。

三、核心模块设计与实现

本项目采用模块化设计思想，将 VQA 系统拆解为视觉特征提取模块、图文融合建模模块、模型训练模块三大核心组件，各模块独立实现且接口标准化，便于维护与迭代优化。

3.1 视觉特征提取模块

3.1.1 模块实现基础

该模块负责从输入图像中提取兼具局部细节与全局语义的视觉特征，为图文融合提供高质量视觉表征。模块实现文件为`./visual/SIGLIP_VIT.py`，核心基于预训练的SigLIP 模型，选用其视觉分支作为特征提取器——SigLIP 模型在图像语义理解任务上表现优异，且预训练权重具备良好的泛化能力，无需从零训练即可提取高质量特征。

3.1.2 特征提取逻辑

模块通过继承 SigLIP 模型并重构特征输出逻辑，简化后续处理流程，具体步骤如下：

1. 输入接收：接收预处理后的图像像素张量（`torch.Tensor`格式，维度为`(batch_size, 3, H, W)`），符合 RGB 图像输入规范；

2. 前向传播：调用 SigLIP 模型的视觉分支（`self.vision_model`）执行前向计算，获取模型全量输出；

3. 特征筛选：舍弃模型输出中的辅助信息，仅提取`last_hidden_state`（最后一层隐藏状态序列）作为核心视觉特征，该特征可同时捕捉图像的局部 patch 细节与全局语义信息，是跨模态融合的最优视觉表征；

4. 格式优化：重构输出逻辑，直接返回张量格式的隐藏状态，替代父类的字典格式输出，减少后续模块的特征解析成本，提升整体效率。

3.2 图文融合建模模块

图文融合是 VQA 系统的核心环节，需解决视觉特征与文本特征的维度不匹配、信息异构等问题。本模块基于 Hugging Face Transformers 框架，扩展 Qwen 语言模型以支持图文输入，同时封装端到端模型与数据集加载逻辑，实现多模态信息的深度融合。

3.2.1 多模态语言模型扩展（`./qwen/Mqwen.py`）

为使 Qwen 语言模型具备图像理解能力，对其进行轻量化扩展，核心思路是将视觉特征注入文本序列，实现图文信息的协同建模，具体实现如下：

1. 文本嵌入：通过 Qwen 原始词嵌入层（`self.wte`），将 token 化后的文本`input_ids`转换为文本嵌入`hidden_states`，维度与语言模型隐藏层维度一致；

2. 图文融合注入：在文本序列中预设的图像占位 token（`<<|extra_0|>`）位置，插入经过维度对齐后的视觉特征（`images[b_idx]`），形成“文本-视觉-文本”的混合嵌入序列，使语言模型能够精准关联文本问题与图像内容；

3. 生成逻辑兼容：保持模型自回归生成文本的核心逻辑不变，确保扩展后的模型与原

始 Qwen 模型的训练、推理流程兼容，降低后续开发成本。

3.2.2 端到端模型封装（`./model/model.py`）

实现`MMultiModal`类，将视觉编码器、特征投影层与扩展后的语言模型有机整合，构建端到端多模态模型，核心设计如下：

1. 组件集成：集成前文实现的 SIGLIP-ViT 视觉编码器、自定义特征投影层与扩展后的 Qwen 语言模型，各组件通过标准化接口对接；

2. 特征投影层：核心解决视觉特征与文本特征的维度不匹配问题，通过双层线性网络（`Linear + GELU + Linear`）将视觉特征维度映射至与语言模型隐藏层维度一致，同时采用正态初始化（权重`N(0,0.01)`，偏置置零）保证训练稳定性；

3. 梯度控制：在训练阶段，通过`with torch.no_grad()`与`detach()`操作冻结视觉编码器权重，仅训练投影层与语言模型的 LoRA 层，平衡训练效率与模型性能。

3.2.3 数据集加载（`./dataset/image_caption_dataset.py`）

实现`ImageCaptionDataset`类，负责加载图文配对数据并完成格式转换，为模型训练提供批量输入，核心功能包括：

1. 数据加载：同步加载图像文件与对应的问答标注数据，建立一一对应关系；

2. 格式对齐：对图像执行 Resize、Normalize 等预处理，使其符合 SIGLIP-ViT 编码器的输入要求；对文本执行 token 化、截断、填充等操作，生成符合 Qwen 模型输入规范的`input_ids`与`labels`；

3. 批量拼接：自定义`collate`函数，处理不同长度的图文数据，生成统一维度的批量数据，适配模型训练的批量计算需求。

3.3 模型训练模块（`./train.py`）

3.3.1 训练策略设计

考虑到多模态模型参数量大、训练成本高的问题，本项目采用“冻结主干+微调局部”的高效训练策略，在保证模型性能的同时降低显存占用与训练耗时：

1. 权重冻结：冻结 SIGLIP-ViT 视觉模型与 Qwen 语言模型的主干权重，仅解冻特征投影层与语言模型的 LoRA（Low-Rank Adaptation）层，减少训练参数量；

2. 显存优化：启用梯度检查点（`gradient_checkpointing_enable`），通过牺牲少量计算速度换取显存占用降低，支持更大批量训练；

3. 精度配置：统一使用`torch.bfloat16`混合精度训练，在避免精度损失的同时提升训练速度，适配主流 GPU 硬件。

3.3.2 训练流程实现

模型训练采用端到端方式，全程自动化执行数据加载、前向计算、损失优化等流程，具体步骤如下：

1. 初始化配置：加载训练参数、模型权重、优化器（选用 AdamW）与学习率调度器，完成训练环境初始化；

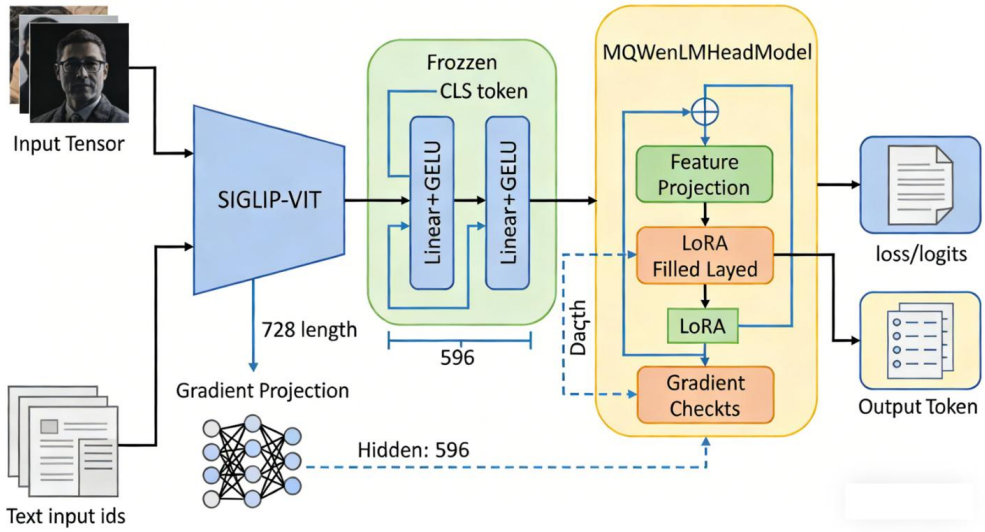
2. 数据加载：通过`ImageCaptionDataset`与`DataLoader`加载预处理后的批量图文数据，传入模型训练；

3. 前向传播：将图像像素张量与文本`input_ids`输入模型，依次完成视觉特征提取、维度投影、图文融合与语言模型计算，输出模型预测`logits`；

4. 损失计算与优化：基于预测`logits`与监督标签`labels`计算交叉熵损失，仅对解冻层参数执行反向传播与梯度更新，冻结层参数不参与梯度计算；

5. 日志与保存：定期记录训练损失、学习率等指标，按迭代步数保存模型 checkpoint，便于后续测试与断点续训。

四、模型架构



4.1 整体架构设计

本项目构建的 VQA 模型为分层式多模态架构，从上至下分为输入层、特征提取层、特征对齐层、融合建模层与输出层，各层职责清晰、协同工作，具体架构如下：

4.1.1 输入层

支持双模态输入，格式严格规范以适配后续模块处理：

- 视觉输入：`torch.Tensor`格式，维度为`(batch_size, 3, H, W)`，代表批量 RGB 图像像素数据，其中 H、W 需与 SIGLIP-ViT 模型输入要求一致（默认 224×224）；

- 文本输入：`torch.LongTensor`格式，维度为`(batch_size, seq_len)`，包含 token 化后的`input_ids`（问题文本+占位符）与对应的`labels`（答案文本），seq_len 为文本序列最大长度。

4.1.2 特征提取层

仅负责视觉特征提取，核心组件为 SIGLIP-ViT 视觉编码器，输入为视觉输入层的像素张量，输出为`(batch_size, image_context_length, Vhidden_dim)`维度的视觉特征序列，其中`image_context_length=728`（固定配置），`Vhidden_dim`为视觉模型隐藏层维度。

4.1.3 特征对齐层

核心为特征投影网络`self.feature_proj`，输入为视觉特征序列，通过双层线性网络将视觉特征维度从`Vhidden_dim`映射至`Lhidden_dim`（语言模型隐藏层维度），输出维度为`(batch_size, 728, Lhidden_dim)`，与文本嵌入维度完全对齐，为图文融合提供基础。

4.1.4 融合建模层

核心为扩展后的 Qwen 多模态语言模型，输入为对齐后的视觉特征与文本嵌入，通过将视觉特征注入文本序列实现图文融合，基于自回归因果语言模型架构完成跨模态推理，输出为`(batch_size, seq_len, vocab_size)`维度的预测`logits`。

4.1.5 输出层

根据场景输出对应结果：训练阶段输出`CausalLMOutputWithPast`对象，包含损失、`logits`与历史键值对，用于模型优化；推理阶段通过`generate`函数输出自回归生成的文本序列，截取有效部分作为最终问答结果。

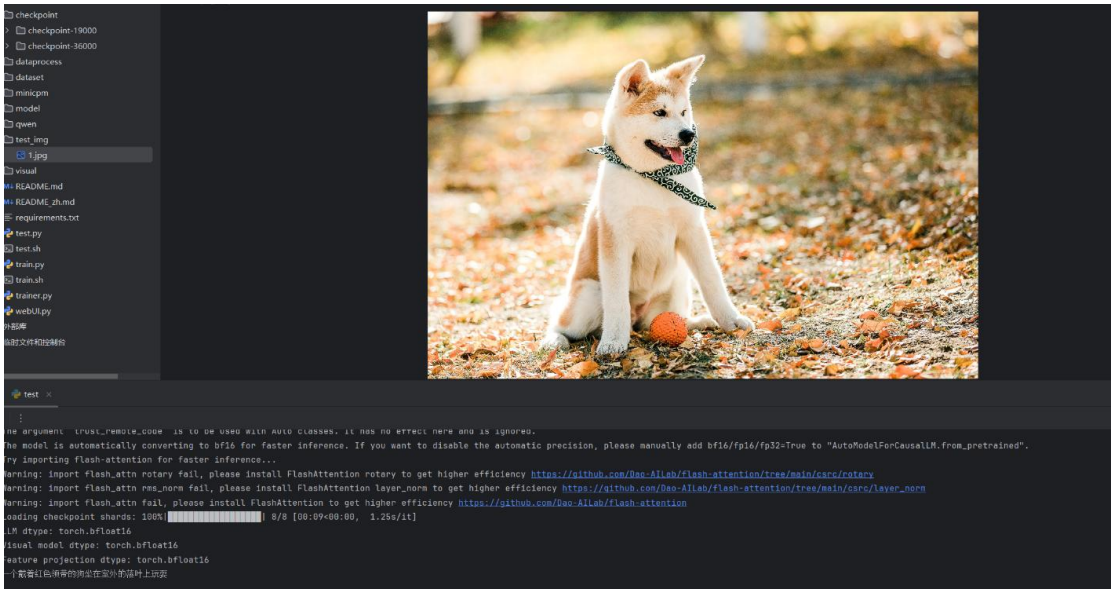
4.2 核心数据流向

模型整体数据流向清晰，实现从原始图文输入到最终问答结果的端到端转化，具体流向如下：

原始图像 → 预处理（Resize/Normalize）→ SIGLIP-ViT 视觉编码器 → 提取`last_hidden_state`视觉特征 → 丢弃`<CLS>` token 特征 → 特征投影层维度对齐 → 注入扩展版 Qwen 模型（文本序列占位符位置）→ 图文融合自回归推理 → 生成问答结果

其中，视觉特征提取阶段通过`with torch.no_grad()`冻结梯度，仅特征投影层与 Qwen 模型的 LoRA 层参与训练更新，确保训练效率与模型稳定性。

五、模型测试与验证



5.1 测试准备

测试权重获取

本项目训练完成的模型权重文件为`checkpoint-36000.zip`，已上传至百度网盘供测试使用，获取方式如下：

链接：https://pan.baidu.com/s/1YVyuO2A4YgwciVn9rAxuYA?pwd=fr7n，提取码：fr7n。下载后解压至`./checkpoints`目录（可通过配置文件修改路径），模型将自动加载权重进行测试。

六、总结与展望

6.1 项目总结

本项目成功构建了一套完整的端到端多模态视觉问答模型，实现了从数据预处理、模块开发、模型训练到测试验证的全流程落地，核心成果与亮点如下：

1. 数据层面：完成多源异构数据集的标准化预处理，解决了不同数据集格式不一致的问题，生成统一、高质量的训练数据，为模型性能提供基础保障；

2. 模块层面：基于现有预训练模型进行轻量化扩展与集成，实现了高效的视觉特征提取、特征对齐与图文融合，模块接口标准化，具备良好的可扩展性；

3. 训练层面：设计“冻结主干+微调局部”的高效训练策略，结合混合精度训练与梯度检查点优化，在普通 GPU 硬件上即可完成训练，降低了部署成本；

4. 应用层面：提供 GUI 可视化测试工具，简化模型验证流程，使非技术人员也可快速上手，为模型的实际应用奠定基础。

同时，项目也存在一定局限性：模型对复杂场景（如遮挡、模糊图像）的问答准确性有待提升，图文融合的深度与效率仍有优化空间，训练数据的场景覆盖度可进一步拓展。

6.2 未来展望

基于本项目的基础架构，未来可从数据集、模型、训练策略、功能拓展四个维度进行优化迭代，提升模型性能与应用价值：

1. 数据集扩展：引入医疗、工业、遥感等专业领域的 VQA 数据集，丰富训练数据的场景与领域覆盖度，提升模型的领域泛化能力；同时可考虑加入噪声数据，增强模型的鲁棒性；

2. 模型结构优化：探索更先进的图文融合策略，如 Cross-Attention 增强、模态注意力机制等，提升跨模态信息交互效率；尝试替换更高效的视觉编码器（如 ViT-G、EVA），进一步提升视觉特征提取质量；

3. 训练策略改进：探索全参数微调与 LoRA 微调的混合策略，或采用 QLoRA 等更高效的参数高效微调方法，在控制训练成本的同时提升模型性能；引入对比学习机制，增强图文模态的关联性建模；

4. 功能与性能拓展：扩展模型功能，支持多轮图文对话、图像区域问答、图文生成等复杂任务；优化推理速度，通过模型量化、剪枝等技术降低显存占用，实现模型的移动端或边缘端部署；

5. 实际应用落地：结合具体场景需求（如智能导购、辅助诊断），优化模型交互逻辑与问答准确性，推动模型从实验阶段走向实际应用，发挥跨模态智能的实用价值。