# 《数据分析与可视化实践》课程设计任务指导书

## 前置准备

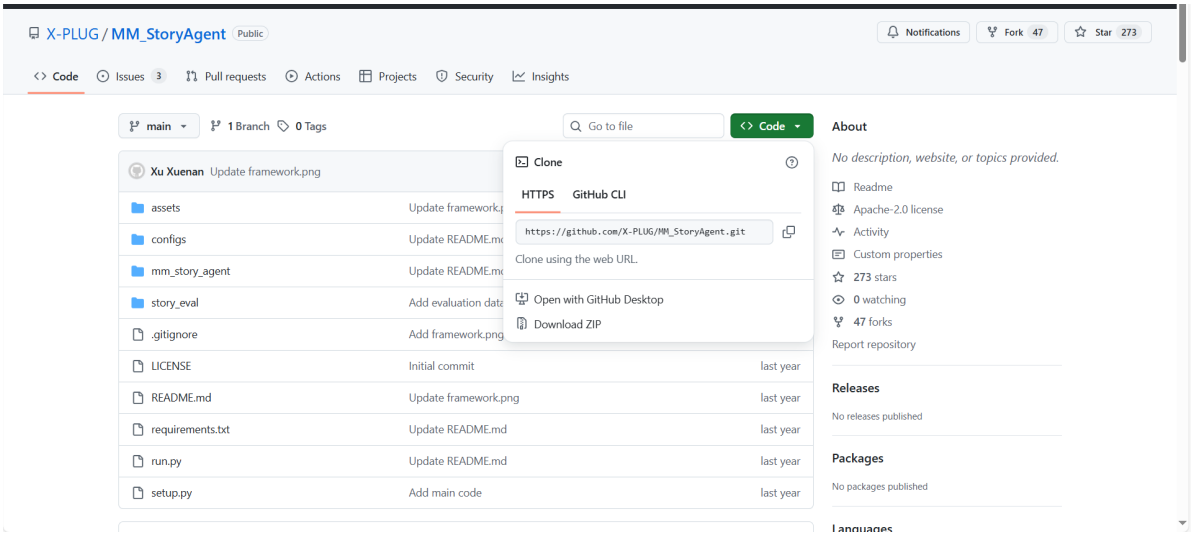
git工具

python环境（3.6以上）

## 一.核心功能复现

### 1.克隆代码和环境准备

git仓库地址：https://github.com/X-PLUG/MM_StoryAgent




在项目文件夹中使用git工具克隆项目代码

```
git init
git clone https://github.com/X-PLUG/MM_StoryAgent.git
```

在项目终端执行命令 安装依赖和故事生成模块

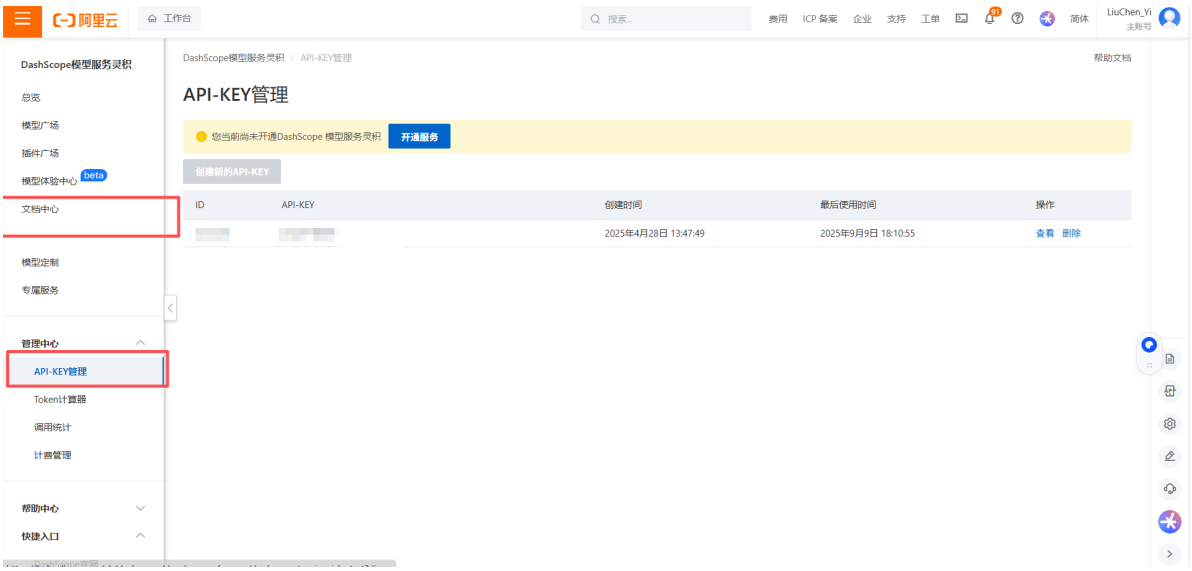
```
pip install -r requirements.txt
pip install -e .
```

### 2. 注册api服务并配置环境变量

根据代码注册：

#### 1.大语言模型服务

原代码中使用的是dashscope（详见MM_StoryAgent\mm_story_agent\modality_agents\llm.py文件）需要配置DASHSCOPE_API_KEY




创建apikey并根据模型id进行调用

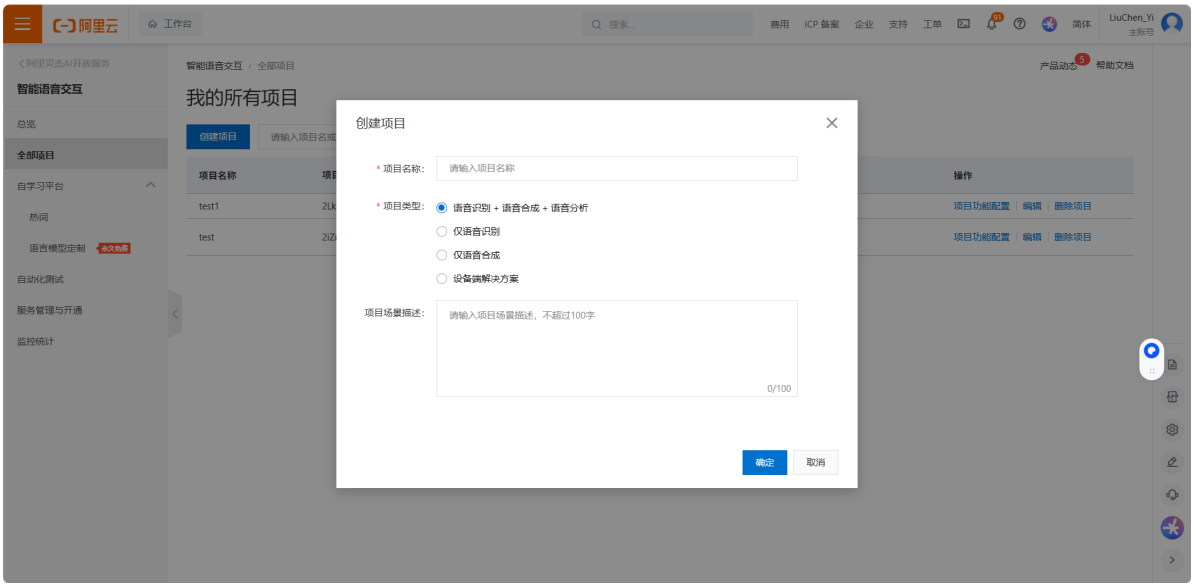
地址：[模型服务灵积-模型广场]

如需要使用更先进的大语言模型 根据开发文档自行配置变量并修改llm.py的封装调用

### 2.长文本语音合成服务

代码中使用的是阿里云的智能语音交互

创建项目：




开通服务后配置项目并获取语音项目的ALIYUN_APP_KEY配置环境变量

文档地址：https://help.aliyun.com/zh/isi/getting-started/start-here

### 3.文生图服务

根据自己需要调用

配置环境变量 在项目根目录下配置环境文件setup_env.sh

示例（根据实际调用服务配置）

```
#!/bin/bash

# StoryAgent 环境变量设置脚本
# 使用方法: source setup_env.sh 或 . setup_env.sh

echo "正在设置 StoryAgent 环境变量..."

# 阿里云 API 密钥
export ALIYUN_APP_KEY=********

# 阿里云访问密钥
export ALIYUN_ACCESS_KEY_ID=********
export ALIYUN_ACCESS_KEY_SECRET=********

# DashScope API 密钥 (通义千问)
export DASHSCOPE_API_KEY=********


echo "✅ 环境变量设置完成！"
echo ""
echo "已设置的环境变量："
echo "- ALIYUN_APP_KEY: $ALIYUN_APP_KEY"
echo "- ALIYUN_ACCESS_KEY_ID: $ALIYUN_ACCESS_KEY_ID"
echo "- ALIYUN_ACCESS_KEY_SECRET: [已设置]"
echo "- DASHSCOPE_API_KEY: [已设置]"
echo ""
echo "现在可以运行 StoryAgent 了："
echo "python run.py -c configs/mm_story_agent_api_only.yaml"
```

# 3.修改视频生成架构

### 1.去除背景音乐和音效模块

熟悉代码 去除MM_StoryAgent\mm_story_agent\mm_story_agent.py和MM_StoryAgent\mm_story_agent\video_compose_agent.py中涉及音效、背景音乐生成的部分

MM_StoryAgent / mm_story_agent / mm_story_agent.py ↑ Top

Code Blame 89 lines (71 loc) · 3.03 KB Raw

```python
from .base import init_tool_instance


class MMStoryAgent:

    def __init__(self) -> None:
        self.modalities = ["image", "sound", "speech", "music"]

    def call_modality_agent(self, modality, agent, params, return_dict):
        result = agent.call(params)
        return_dict[modality] = result

    def write_story(self, config):
        cfg = config["story_writer"]
        story_writer = init_tool_instance(cfg)
        pages = story_writer.call(cfg["params"])
        return pages

    def generate_modality_assets(self, config, pages):
        script_data = {"pages": [{"story": page} for page in pages]}
        story_dir = Path(config["story_dir"])

        for sub_dir in self.modalities:
            (story_dir / sub_dir).mkdir(exist_ok=True, parents=True)

        agents = {}
        params = {}
        for modality in self.modalities:
            agents[modality] = init_tool_instance(config[modality + "_generation"])
            params[modality] = config[modality + "_generation"]["params"].copy()
            params[modality].update({
                "pages": pages,
```

MM_StoryAgent / mm_story_agent / video_compose_agent.py ↑ Top

Code Blame 419 lines (351 loc) · 16.6 KB Raw

```python
        ).set_start(sum(durations[:-1]) - slide_duration * (len(clips) - 1))
        videos.append(last_clip)

        video = CompositeVideoClip(videos)
        return video


def compose_video(story_dir: Union[str, Path],
                  save_path: Union[str, Path],
                  captions: List,
                  music_path: Union[str, Path],
                  num_pages: int,
                  fps: int = 10,
                  audio_sample_rate: int = 16000,
                  audio_codec: str = "mp3",
                  caption_config: dict = {},
                  fade_duration: float = 1.0,
                  slide_duration: float = 0.4,
                  zoom_speed: float = 0.5,
                  move_ratio: float = 0.95,
                  sound_volume: float = 0.2,
                  music_volume: float = 0.2,
                  bg_speech_ratio: float = 0.4):
    if not isinstance(story_dir, Path):
        story_dir = Path(story_dir)

    sound_dir = story_dir / "sound"
    image_dir = story_dir / "image"
    speech_dir = story_dir / "speech"

    video_clips = []
    # audio_durations = []
```

## 2.修改生图模块为api调用方式

修改MM_StoryAgent\mm_story_agent\modality_agents\image_agent.py

将原先的stable diffusion模型调整为api调用方式 并响应代码架构 调试生图对应的prompt

具体需要修改的生图调用方法在MM_StoryAgent\mm_story_agent\modality_agents\image_agent.py的story_diffusion_t2i工具的call方法中

toryAgent/blob/main/mm_story_agent/modality_agents/image_agent.py

MM_StoryAgent / mm_story_agent / modality_agents / image_agent.py ↑ Top

Code Blame 693 lines (595 loc) · 30.3 KB Raw

```
574
575
576     @register_tool("story_diffusion_t2i")
577     class StoryDiffusionAgent:
578
579         def __init__(self, cfg) -> None:
580             self.cfg = cfg
581
582         def call(self, params: Dict):
583             pages: List = params["pages"]
584             save_path: str = params["save_path"]
585             role_dict = self.extract_role_from_story(pages)
586             image_prompts = self.generate_image_prompt_from_story(pages)
587             image_prompts_with_role_desc = []
588             for image_prompt in image_prompts:
589                 for role, role_desc in role_dict.items():
590                     if role in image_prompt:
591                         image_prompt = image_prompt.replace(role, role_desc)
592                 image_prompts_with_role_desc.append(image_prompt)
593             generation_agent = StoryDiffusionSynthesizer(
594                 num_pages=len(pages),
595                 height=self.cfg.get("height", 512),
596                 width=self.cfg.get("width", 512),
597                 model_name=self.cfg.get("model_name", "stabilityai/stable-diffusion-xl-base-1.0"),
598                 id_length=self.cfg.get("id_length", 4),
599                 num_steps=self.cfg.get("num_steps", 50)
600             )
601             images = generation_agent.call(
602                 image_prompts_with_role_desc,
603                 style_name=params.get("style_name", "Storybook"),
604                 guidance_scale=params.get("guidance_scale", 5.0),
```

# 二.加入长文本处理扩展模块

修改MM_StoryAgent\mm_story_agent\modality_agents\story_agent.py

将原先的story topic -> outline -> story pages的架构修改为data -> story pages，并调试生成的prompt

**生成代码原本的架构：**

输入：MM_StoryAgent/configs/mm_story_agent.yaml中的storyWriter.params

```
story_writer:
    tool: qa_outline_story_writer
    cfg:
        max_conv_turns: 3
        num_outline: 4
        temperature: 0.5
    params:
        story_topic: "Time Management: A child learning how to manage their time effectively."
        main_role: "(no main role specified)"
        scene: "(no scene specified)"
```

处理:MM_StoryAgent/mm_story_agent/modality_agents/story_agent.py 对storytopi进行大纲扩写并生成相应的故事页面。

```
136             # print(all_pages)
137             return all_pages
138
139         def call(self, params):
140             outline = self.generate_outline(params)
141             pages = self.generate_story_from_outline(outline)
142             return pages
```

由于数据叙事需要更复杂的真实数据驱动，由扩写带来的幻觉是我们生成视频不需要的，因此需要修改MM_StoryAgent/configs/mm_story_agent.yaml中的storyWriter.params配置输入为长文本，并相应地调整MM_StoryAgent/mm_story_agent/modality_agents/story_agent.py 中generate_outline和generate_story_from_outline函数以及对应的prompt

# 三.系统基础功能

进入项目根目录

配置环境变量

```
source setup_env.sh
```

执行代码生成视频

```
python run.py -c configs/mm_story_agent.yaml
```