方法说明：

1. Add2Milvus.py

提取视频关键帧，并将其保存在Frames文件夹中。读取视频关键帧并将其转换成特征向量，传入Milvus数据库中存储。

1. Search.py

调用SearchVideoByOnePic方法进行以1张图片搜视频。

1. Add.py

调用add\_videos2milvus方法增量更新视频，将增量视频的关键帧特征向量存储至Milvus数据库中。

1. Delete.py

调用delete\_frame\_by\_v\_id方法根据视频id列表删除Milvus数据库中相应的视频的关键帧的特征向量。

|  |  |  |  |
| --- | --- | --- | --- |
| 方法 | 功能 | 输入参数定义 | 输出参数定义 |
| GetVideoFrames | 获取视频关键帧 | video\_path\_list  (视频路径列表) | videos: { 'video\_id\_list': v\_id\_list (视频id列表), 'video\_path\_list': videos\_path\_list(视频路径列表)  }  frames: {  'frame\_id\_list': frame\_id\_list (关键帧id列表),  'frame\_path\_list': frame\_path\_list (关键帧路径列表),  'frame\_position\_list': frame\_position\_list (关键帧位置信息列表)  } |
| GetFramesFeature | 批量获取视频关键帧特征向量 | frame\_path\_list  (视频关键帧路径列表)  weight\_path  (提取特征向量模型路径) | feature\_list (特征向量列表) |
| SearchVideoByOnePic | 以图搜视频 | Collection  (Milvus中的Collection)  weight\_path  (提取特征向量模型路径)  pic\_path  (图片路径)  Similarity  (相似度阈值)  Number  (前N个视频) | distance\_list (相似度列表),  frame\_id\_list (关键帧id列表) |
| delete\_frame\_by\_v\_id | 删除视频关键帧的特征向量 | Collection  (Milvus中的Collection)  v\_ids  (要删除视频的id列表) | result (被删除的视频关键帧的特征向量的数量) |
| add\_videos2milvus | 对新增视频进行插入操作，返回插入结果 | Collection  (Milvus中的Collection)  video\_path\_list  (新增视频的路径列表)  weight\_path  (提取特征向量模型路径) | result: { "video\_path": video\_path (视频路径),  "isSuccess": (True: 插入成功；False: 插入失败)  } |

Milvus数据库相关说明：

版本：Windows版本

使用方法：

1. 安装docker 的Windows版本,下载网址：<https://www.docker.com/>
2. 安装Attu,下载网址：<https://github.com/zilliztech/attu/blob/main/doc/zh-CN/attu_install-docker.md>
3. 进入milvus 目录后执行以下命令：

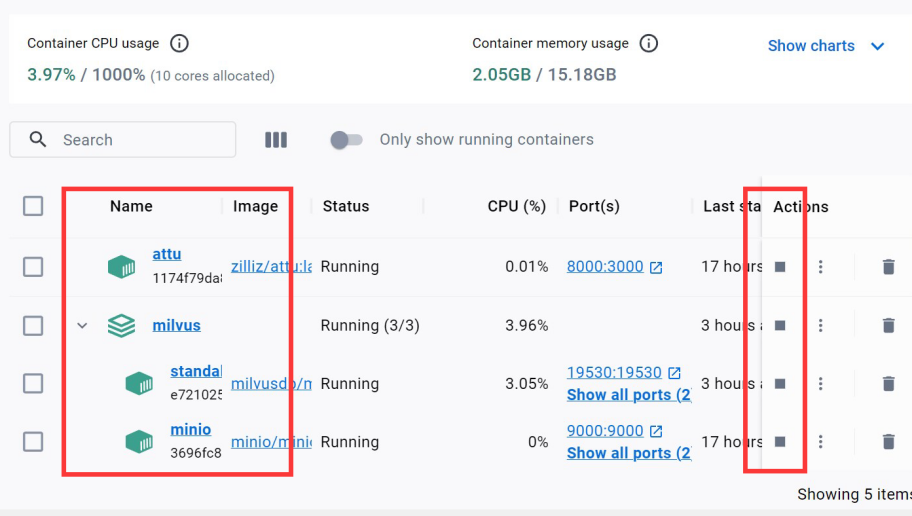
docker-compose up -d

首次启动：docker run --name attu -p 8000:3000 -d -e MILVUS\_URL={your machine IP} zilliz/attu:latest

(红色部分替换为IP地址)（attu为自定义的名字）

后续启动：docker start attu

1. 待处于以下运行状态后，创建并配置好Mysql数据库后，将待提取关键帧的视频放在Crawer\_Video文件夹中，运行Add2Milvus.py，即可将所有视频的关键帧保存至Frames文件夹下（该文件夹会自动生成），并对所有关键帧进行特征向量的提取并存储到Milvus数据库中，然后便可以在Attu中查看(Attu的账号为root,密码为123456)。



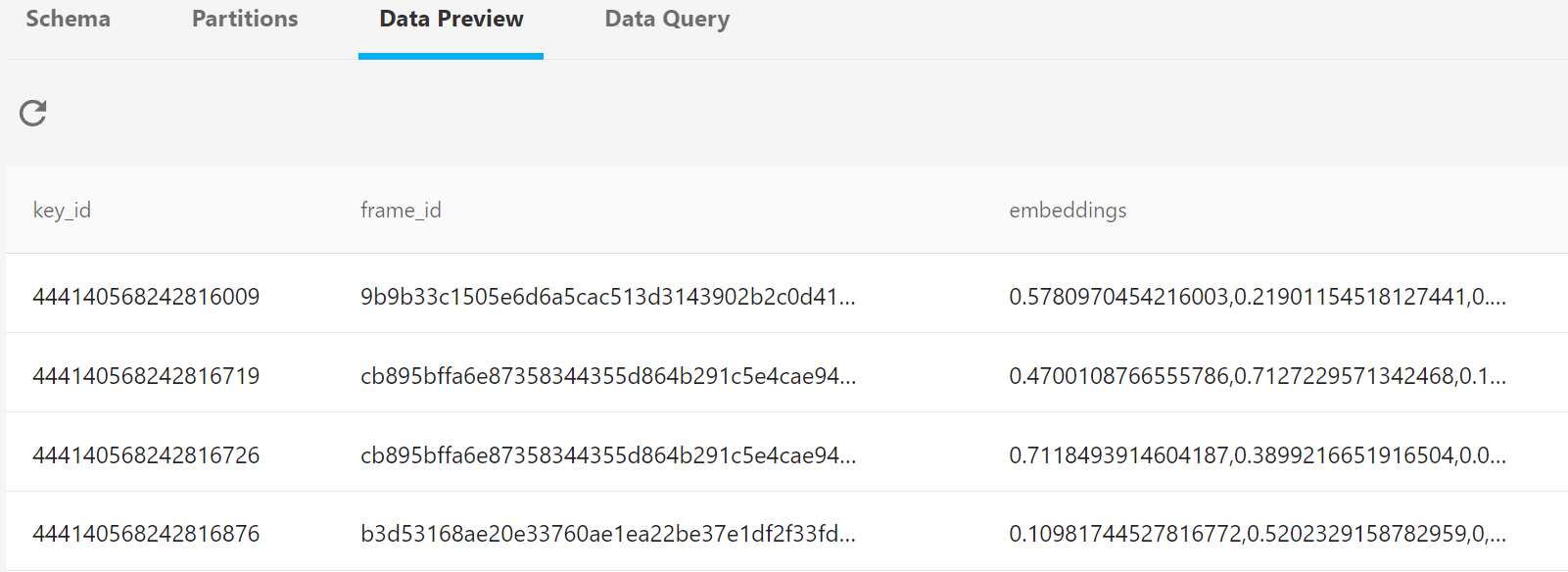


1. 将待检索的的图片放至pics文件夹下，修改图片路径并运行Search.py便可以进行以图搜视频。

数据库结构：

key\_id为Milvus数据库自带的主键,

frame\_id为帧id,（frame\_id由videoMD5和该帧的时间戳信息组成,videoMD5即video的MD5编码，也就是视频id）

embeddings为帧所对应的特征向量。  


Mysql数据库相关说明：

1. 数据库代码部分：mysql\_utils.py：更换mysql\_host、mysql\_db、mysql\_user、mysql\_pwd

文本

中度可信度描述已自动生成

1. 通过可视化数据库软件（如：Navicat）手动建立mysql数据库，videos数据库包括三个表，各表具体设置如下：

文本

描述已自动生成

* 1. frame表



* 1. frame\_video表



* 1. video表

