**小工具功能增加**

[TFLR小工具文档集合](https://momenta.feishu.cn/wiki/IaKywKj6Eixx6YkV69HcNAHKnKH?open_in_browser=true)

小工具的需求与开发规划在此查看

小工具是基于streamlit开发的，开发前可先查阅streamlit使用方法，安装好与streamlit相关的包

小工具代码仓库：https://gitlab.momenta.works/fujia.zhao/tflr\_ddld4.0\_mct/-/tree/master/utils

仓库权限请找@Fujia Zhao开启

**代码修改步骤：**

1.在run.py中增加对应的page项，在首页呈现出所需的功能条目



2.在./app路径下新增实现功能的代码，呈现格式参考已有代码即可

3.最后将要求功能完成后，在[TFLR小工具文档集合](https://momenta.feishu.cn/wiki/IaKywKj6Eixx6YkV69HcNAHKnKH?open_in_browser=true)中写好教程，然后在核心群里@大家，让大家知道小工具增加了哪些功能，具体要如何使用

具体教程的编写可以参考[逻辑灯数据转4.0格式](https://momenta.feishu.cn/wiki/YDJkwJKkeiVCY4ksFrXcY4s6nsb?open_in_browser=true)

也可以录成视频的方式呈现

tips：

1.实现具体功能的.py文件名尽量一目了然，让人明白是实现什么功能的文件

2.token要给用户自主上传的接口，避免默认token过期，用户无法使用小工具

3.当流程需要新建文件用于储存中间数据时，文件名可以与时间戳挂钩，避免多个用户同时修改同一个文件

4.公司数据量通常比较大，可以采取线程池的方式实现功能

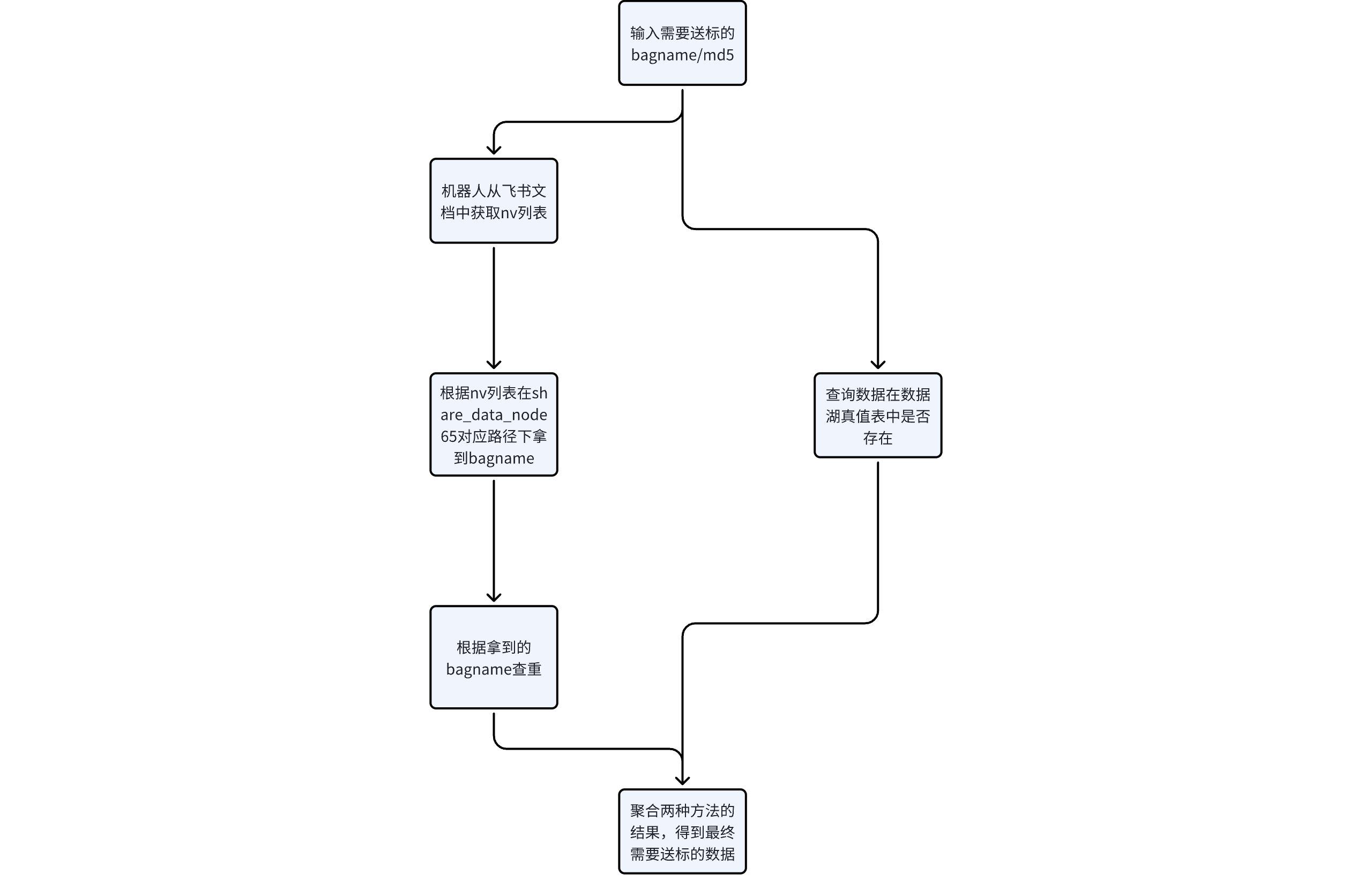
**由我完成部分：**

1.送标查重功能:

送标查重目前根据两个模块，一是数据湖perception\_experiment\_v2中的dd4d\_tflr\_fusion\_gt\_bags表，二是[Sales sheet](https://momenta.feishu.cn/wiki/QGp5wDVbsiqtB2kmKrgcpkdynqb)表。根据nv名去/share\_data\_node65/users/wangqi/tl\_dd\_fusion/fusion\_data/label\_result/v6，

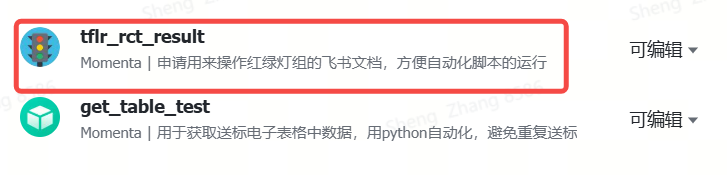
/share\_data\_node65/users/wangqi/tl\_dd\_fusion/fusion\_data/label\_result/v6p1 下取得对应的bag名，来进行查重。

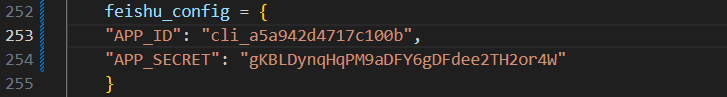
具体查重步骤如下所示：

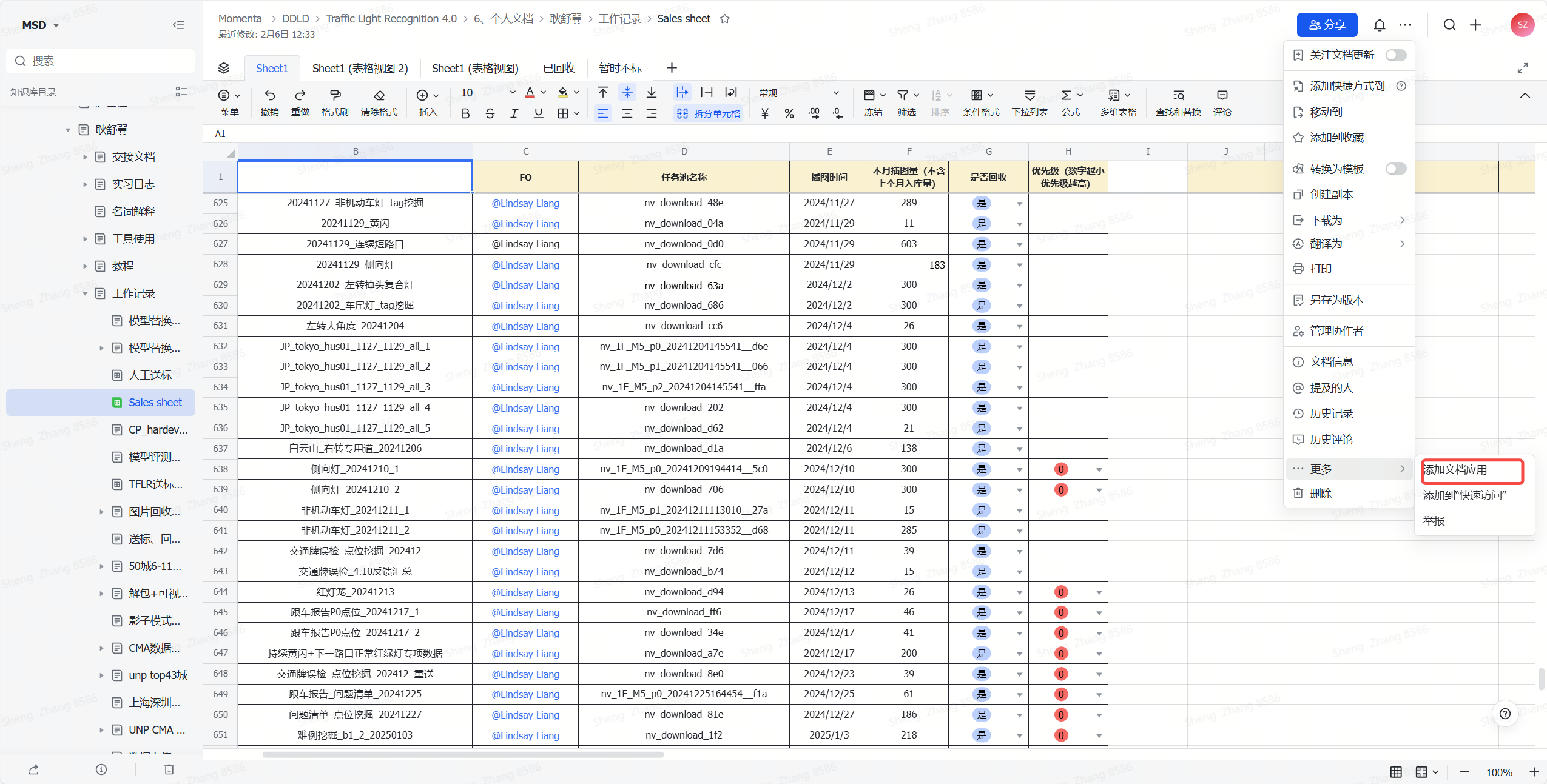


1.创建FeishuHelper对象，使用该对象获得[Sales sheet](https://momenta.feishu.cn/wiki/QGp5wDVbsiqtB2kmKrgcpkdynqb)表中的所有nv名储存在feishu\_data变量中、

飞书表中的数据由飞书机器人获取,对应的机器人是@Fujia Zhao创建的应用







飞书机器人在此添加。首先你要获得文档的管理权限，可以向@Ze Wu申请该文档的管理权限

2.之后使用get\_bag\_names\_multi函数，多线程从/share\_data\_node65/users/wangqi/tl\_dd\_fusion/fusion\_data/label\_result/v6/获取所有已送标数据的bagname。利用获取的bagname与当前需要送标的bagname/md5s对比。把当前需要送标数据分为已经送标与还未送标。

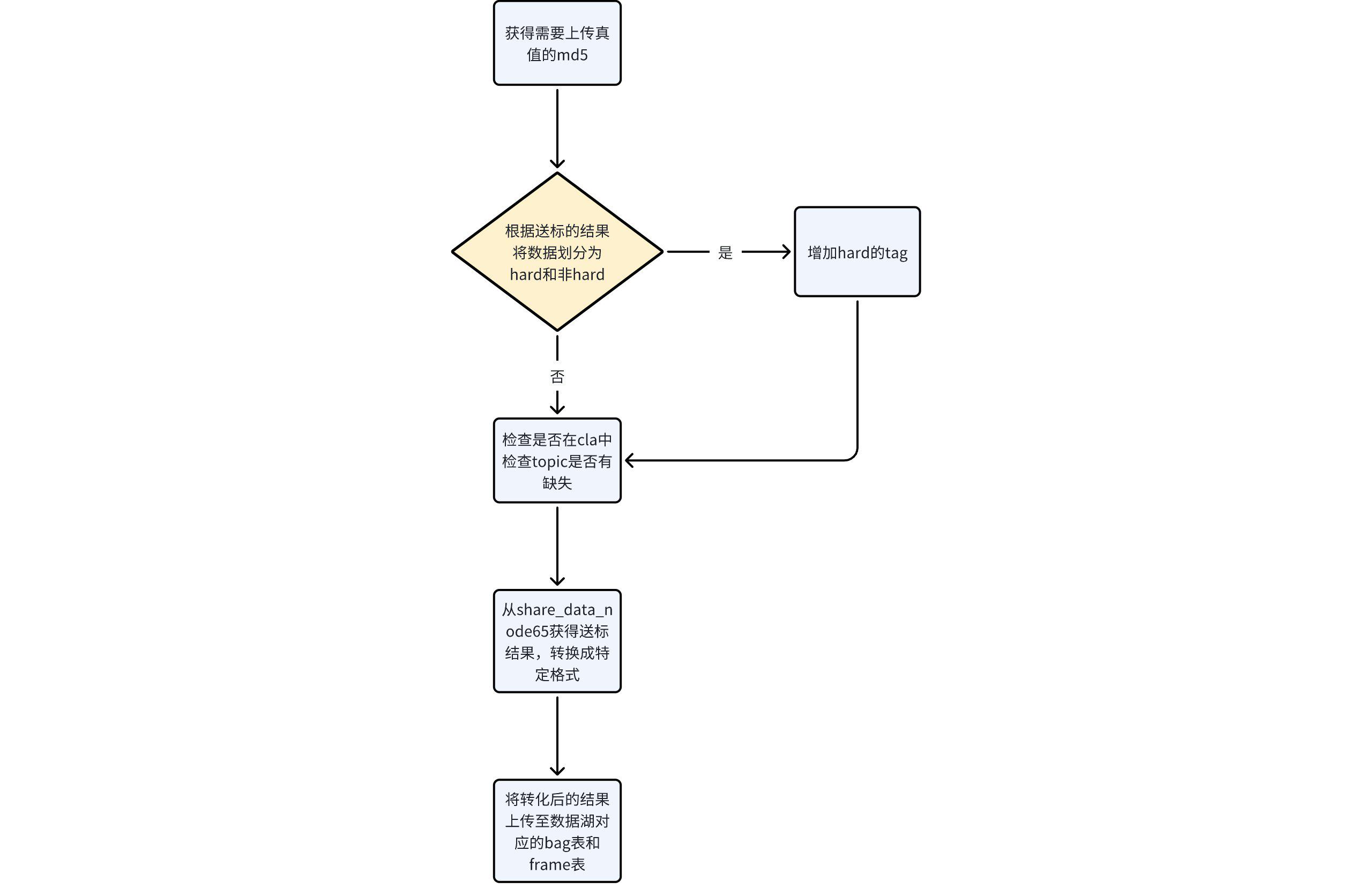
3.从dd4d\_tflr\_fusion\_gt\_bags数据湖中获得所有已上传的md5，查询当前需要送标的数据是否已经在其中存在，把当前需要送标数据分为已经送标与还未送标。

4.聚合第二步与第三步的结果，已经送标取并集，还未送标取交集，返回结果即可

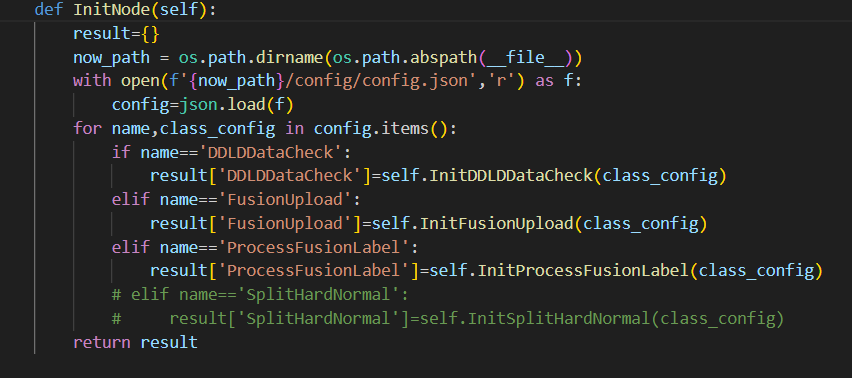
2.fusion真值上传部分

原代码仓库[gitlab.momenta.works](https://gitlab.momenta.works/zhudawei/tflr_dpl_4.0/-/tree/zfj/dev/fusionv3tov4)

上传流程图：



真值上传部分基于该代码重构，把代码中graph部分拆为三个类，



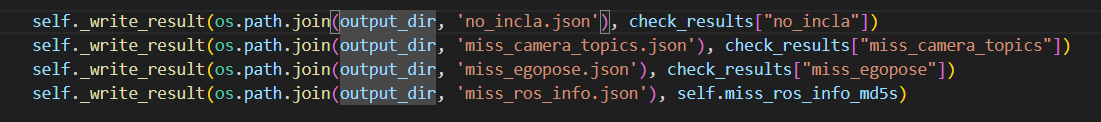
整体介绍：三个类对应原代码中graph的三个步骤，原代码用json文件暂存，运行过程中的中间数据，小工具采取了同样的方法。但是考虑多个用户同时操作的情况，这里使用时间戳为名字建立中间文件。代码中考虑了多种情况，避免新建的中间文件没被删除导致硬盘空间泄漏。但是有极端场景可能未考虑在内，后续如果出现bug需要考虑能否改进。

具体步骤：

1.实例化SplitHardNormal，用实例化后的对象将输入数据划分为hard数据与非hard数据(是否为hard在

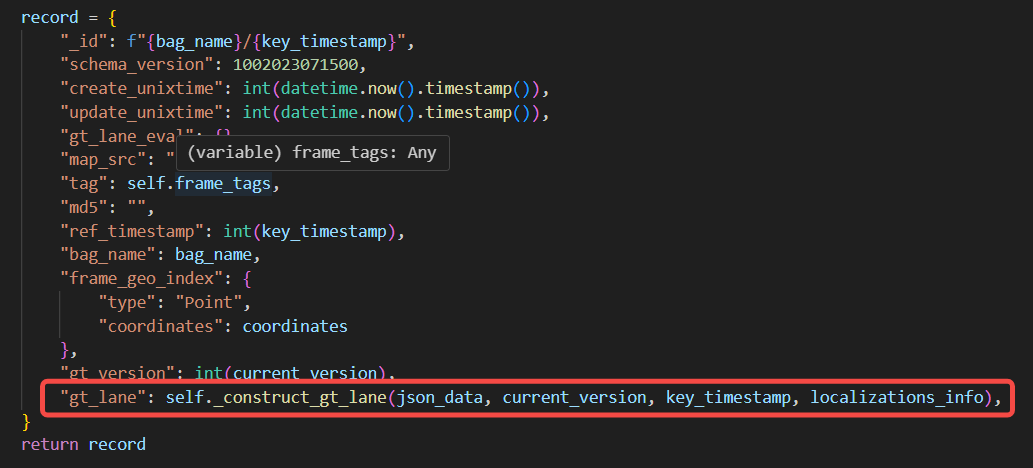
"/share\_data\_node65/users/wangqi/tl\_dd\_fusion/fusion\_data/label\_result/v6"里的人工标注结果中可以查阅到)，hard与非hard数据分别走之后三个流程。

2.DDLDDataCheck用于检查输入数据，先筛选出cla中存在的数据，再对存在的数据进行筛选，检查topic是否有缺失。无法使用的md5储存在如下路径中。



3.把DDLDDataCheck的输出(不存在问题的md5)作为ProcessFusionLabel的输入。ProcessFusionLabel用于将"/share\_data\_node65/users/wangqi/tl\_dd\_fusion/fusion\_data/label\_result/v6"中标注的数据转换为特定格式。使每个时间戳都有与之对应的逻辑灯真值，egopose以及位置信息。最后为每个md5创造一个json文件，用于储存转化后的结果。

4.以ProcessFusionLabel中的输出(每个md5对应的json文件)作为FusionUpload的输入，为frame表中的每个数据添加第三步中获得的真值数据。



使用\_construct\_gt\_lane将真值数据写入对应frame。最后使用数据湖客户端datalake\_client = DatalakeClient(token=self.ppl\_token)，将本地记录的数据推送到数据湖上，完成真值数据的更新。