景行人工智能平台

用户手册 V5.4

目　录

目　录 1
关键词 6
第一章 概述与架构 7
第二章 主要功能 8
2.1. 数据集 8
2.1.1. 数值数据 8
2.1.1.1. “数值数据-数据文件”数据集 8
2.1.1.2. “数值数据-数据库”数据集 11
2.1.1.3. “数值数据-HDFS”数据集 13
2.1.2. 图像数据 16
2.1.2.1. “图像数据-数据文件”数据集 16
2.1.2.2. “图像数据-HDFS”数据集 18
2.1.3. 其他 20
2.1.4. 共享数据集 22
2.1.4.1. 将我的数据集共享至共享数据集 23
2.1.4.2. 新建共享数据集 26
2.1.5. 图像标注 29
2.1.5.1. 编辑标签 30
2.1.5.2. 导入图像 30
2.1.5.3. 图像识别 32
2.1.5.4. 目标检测 33
2.2. 可视化建模 40
2.2.1. 方案设计 40
2.2.1.1. 构建方案设计流程 41
2.2.1.1.1. 新建 41
2.2.1.1.2. 设计 42
2.2.1.2. 操作 49

2.2.1.2.1. 方案保存 49
2.2.1.2.2. 方案另存为 49
2.2.1.2.3. 方案运行 50
2.2.1.3. 管理 52
2.2.1.3.1. 查看描述 53
2.2.1.3.2. 删除 53
2.2.2. 方案实例 54
2.2.3. 模型库 57
2.2.3.1. 新建模型库 58
2.2.3.2. 模型训练历史列表 59
2.2.3.3. 模型卡片 63
2.2.3.4. 共享模型 64
2.3. 案例中心 66
2.3.1. 案例详情 67
2.3.2. 案例使用 68
2.4. 开发中心 70
2.4.1. 新建开发环境 71
2.4.2. 使用开发环境 74
2.4.2.1. 应用 Jupyter 服务 74
2.4.2.2. 应用 VSCode 服务 77
2.4.2.3. 应用 CentOs 桌面/Ubuntu 桌面 82
2.5. AI 作业提交 86
2.5.1. Tensorflow 86
2.5.1.1. 单机模式 86
2.5.1.2. PS 并行模式 89
2.5.1.3. Horovod 并行模式 90
2.5.2. Pytorch 90
2.5.2.1. 单机模式 90
2.5.2.2. Horovod 并行模式 92

2.5.3. Mxnet 93
2.5.3.1. 单机模式 93
2.5.3.2. Horovod 并行模式 94
2.5.4. PaddlePaddle 95
2.5.4.1. 单机单卡 95
2.5.4.2. 单机多卡 97
2.6. 实验管理 97
2.6.1. 准备训练程序 98
2.6.2. 执行训练程序 99
2.6.2.1. 提交 AI 作业 99
2.6.2.2. 运行模型结构的方案 100
2.6.2.3. 从开发中心运行 101
2.7. 服务管理 104
2.7.1. 添加服务 105
2.7.1.1. “终端容器”类型的服务 105
2.7.1.2. “Tensorboard”类型的服务 106
2.7.1.3. “Tensorflow Serving”类型的服务 107
2.7.1.4. “Pytorch Serving”类型的服务 109
2.7.2. 重新添加服务 110
2.7.3. 查看服务信息 112
2.7.4. 应用服务 113
2.7.4.1. 访问服务 113
2.7.4.2. 保存为镜像 114
2.7.5. 删除服务 115
附录一：数据加载处理组件属性说明 117
（一） 数据提取 117
（二） 数据加载 117
（三） 数据预处理 119
1. 数值数据集预处理 120

2. 图像数据集预处理 123
（1） 图像预处理 123
（2） 分类图像数据增强 126
（3） 目标检测数据集增强 134
附录二：模型设计组件属性说明 136
（一） 深度学习 139
（二） 基础网络模型 178
（三） 模型结果可视化 188
附录三：算法库组件属性说明 190
（一） 数值算法 190
（二） 视觉算法 217
（三） 仿真回归算法 228
附录四：模型预测组件属性说明 252
附录五：镜像说明以及相关 python 包说明 254
（一） jhinno/tensorflow:py37-2.5 254
（二） jhinno/pytorch:py37-1.10.0 268
（三） jhinno/mxnet:py37-1.9.1 281
（四） jhinno/paddle:py37-2.3.2 291
（五） jhinno/webide:cuda11.2.0 302
（六） jhinno/tensorboard:v5.4 321
（七） jhinno/tf-serving:py37-2.5 325
（八） jhinno/torch-serving:py3-1.10 327
（九） jhinno/ubuntu-desktop:18.04-cudagl11 328
（十） jhinno/centos-desktop:7-cudagl11 342
（十一） jhinno/training-center:v5.4 355
（十二） jhinno/makesense:v1.0 355
（十三） jhinno/portainer:v5.4 355
（十四） jhinno/traefik:v2.4.8 355
附录六：libaiflow API 356

（一） 实验管理相关 API 356
（二） 指标和超参数相关 API 360
（三） 文件和可视化相关 API 362
（四） 模型相关 API 367
（五） 数据集相关 API 367
附录七：实验管理 SDK 代码样例 377
附录八：常见问题及解决办法 381

关键词

${APPFORM_TOP}或$APPFORM_TOP：景行应用门户软件安装后的顶级目录，如：/apps/appform；

${APPFORM_CONFDIR}或$APPFORM_CONFDIR：景行应用门户软件安装后的配置文件目录，如：/apps/appform/conf；

${JH_SPOOLER_BASE}或$JH_SPOOLER_BASE：仿真计算软件的数据目录。

第一章 概述与架构

景行人工智能软件提供了丰富的算法和建模框架，不仅提供了用户图形化、拖拽式的建模方式，支持用户“零编码建模”；也提供了开放式接口，支持高级用户自定义算法和模型。用户创建的模型可以用来创建自己的 AI 场景解决方案，也可以共享给其他用户使用。人工智能软件提供了数据集、模型共享功能，用户可以把自己的数据集和模型共享出去，帮助团队简单、快速地构建模型。人工智能软件还提供了常用的 VSCode、Jupyter 等 Web IDE，灵活扩展用户 AI 实践过程。

系统的整体功能架构如下图所示：

系统功能架构图

整个系统主要包括 Web 应用门户、数据集管理、模型管理、方案管理、图像标注等模块。

第二章 主要功能

2.1. 数据集

数据集中包含“我的数据集”和“共享数据集”，在“我的数据集”下可以创建“数值数据”、“图像数据”和“其他”数据集。数据集为用户数据的集合，目前支持“数值数据”数据集、“图像数据”数据集和“其他”数据集的管理。同时支持数据集的共享，让不同用户之间可以相互共享数据。

方案设计中支持以可视化方式引用数据集，可以使用已有的数据集进行数据处理和模型训练。

2.1.1. 数值数据

2.1.1.1. “数值数据-数据文件”数据集

新建“数值数据-数据文件”数据集，操作步骤如下：依次点击“我的数据集”-“数值数据”-“数据文件”，然后在右侧的工作区，点击“新建数据集”卡片按钮，此时会弹出“新建数据集”窗口，如下图所示：

新建“数值数据-数据文件”数据集

图中每个参数的具体含义如下：

名称：数据集名称，输入任意符合命名规则的名称即可。

密级：管理员开启密级功能后，显示此选项，默认为用户密级，用于数据安全、保密。

选择文件：点击蓝色的“文件夹图标”按钮，然后在弹出的“选择文件”窗口中选择文件即可。目前仅支持 csv 格式的数据文件，支持选择多个文件，需要保证表头相同，无表头的数据文件需要保证数据列数相同，否则会导致新建数据集失败。

字符集：选择文件的编码格式，支持以下字符集：

- UTF-8
- GB2312
- GBK
- ISO-8859-1
- UTF-16
- Big5
- UTF-32
- US-ASCII

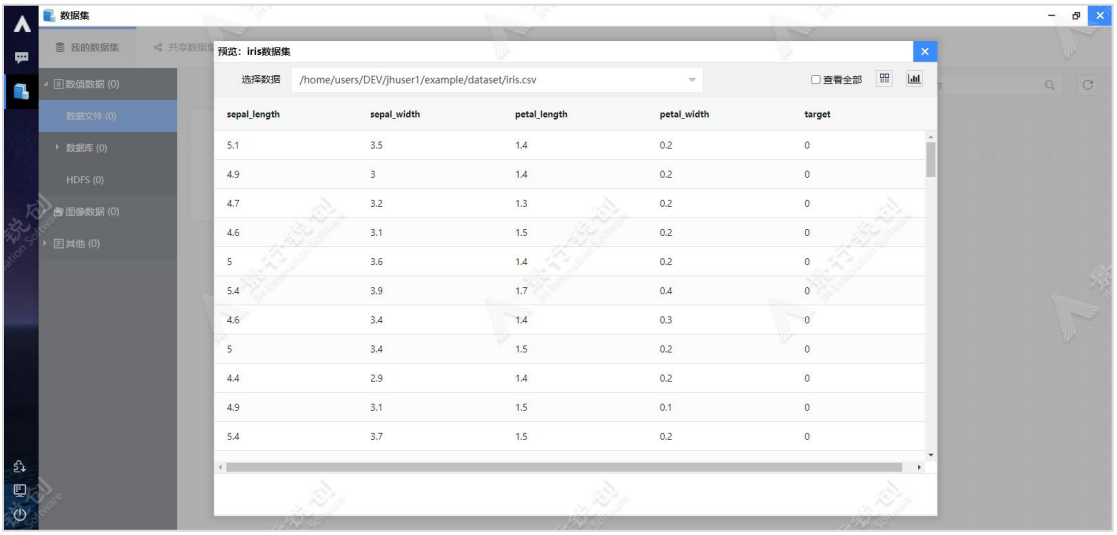
分隔符：选择文件中使用的分隔符，支持以下分隔符：

- 逗号
- 空格
- 制表符
- 下划线
- 中划线
- 分号

表头：选择文件是否包含表头。

描述：输入该数据集的描述信息，可选。

填写完成后，点击“预览”按钮，即可预览数据集的数据文件内容，如下图所示：

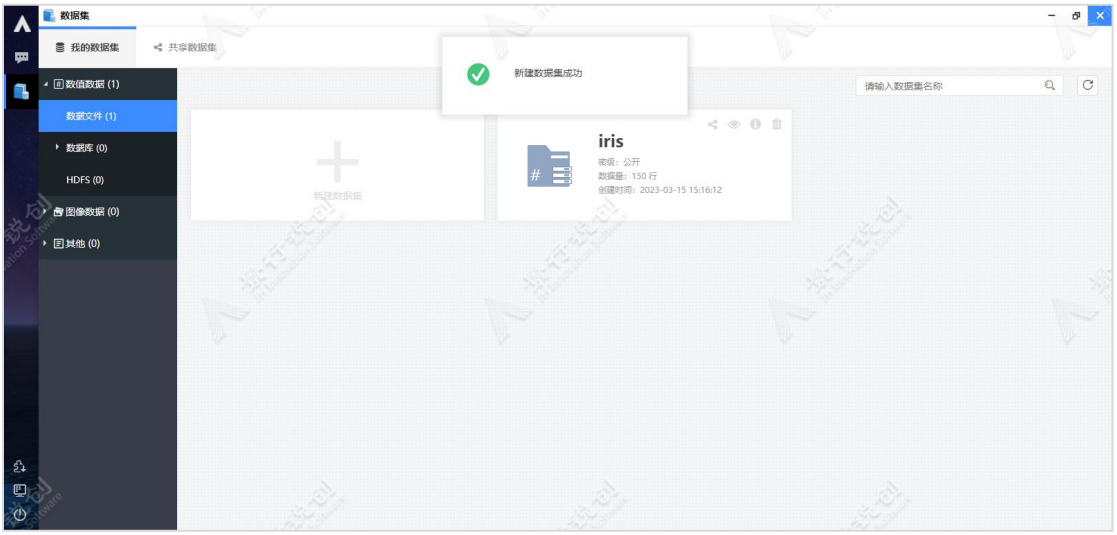


预览“数值数据-数据文件”数据集

点击右上角的“数据可视化”图标按钮，将预览界面切换至数据可视化界面，如下图所示：

以可视化形式预览数据

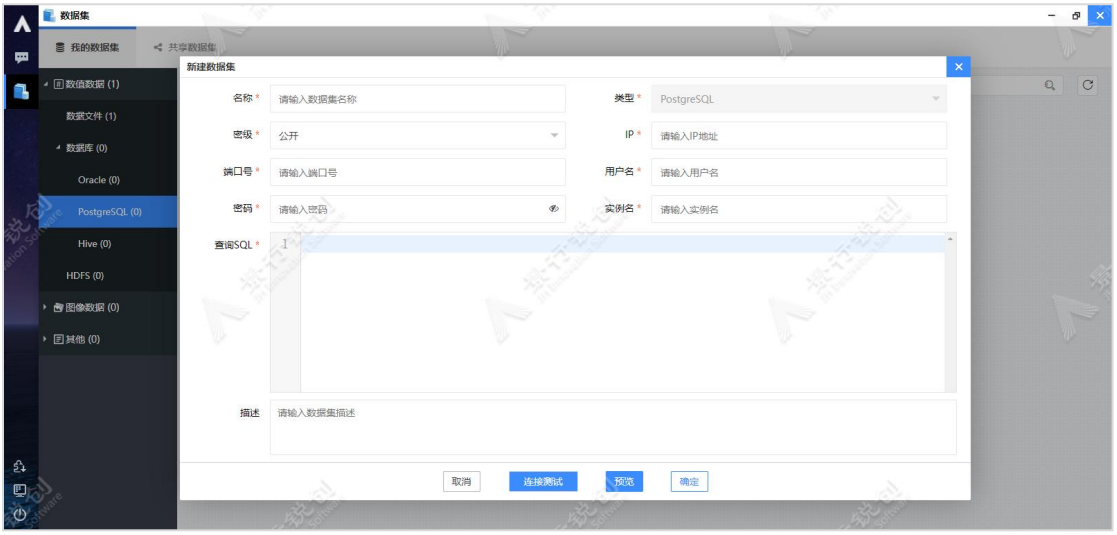
点击“确定”按钮，新建数据集，如下图所示：



新建“数值数据-数据文件”数据集成功示意图

2.1.1.2. “数值数据-数据库”数据集

新建“PostgreSQL 数据库”数据集，操作步骤如下：依次点击“我的数据集”-“数值数据”-“数据库”-“PostgreSQL”，然后在右侧的工作区，点击“新建数据集”卡片按钮，此时会弹出“新建数据集”窗口，如下图所示：



新建“数值数据-PostgreSQL 数据库”数据集

图中每个参数的具体含义如下：

名称：数据集的名称，输入任意符合命名规则的名称即可。

类型：选择数据库的类型。

密级：管理员开启密级功能后，显示此选项，默认为用户密级，用于数据安全、保密。

IP：输入数据库所在机器的 IP。

端口号：输入数据库使用的端口号。

用户名：输入数据库的用户名。

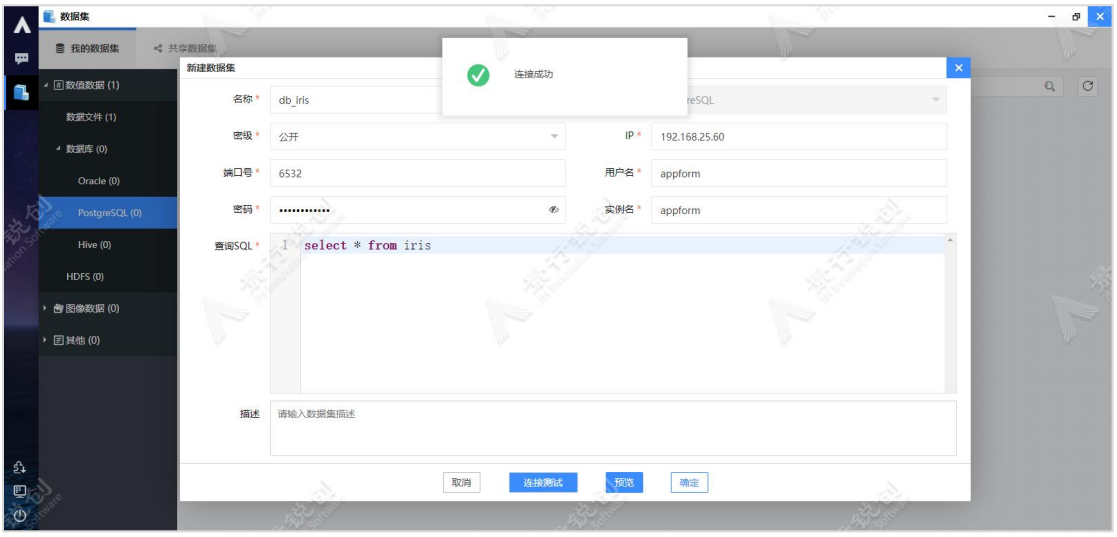
密码：输入数据库用户对应的密码。

实例名：输入数据库名。

查询 SQL：输入 SQL 查询语句（注意：不要在查询语句后面加分号，eg：select * from tbl）。

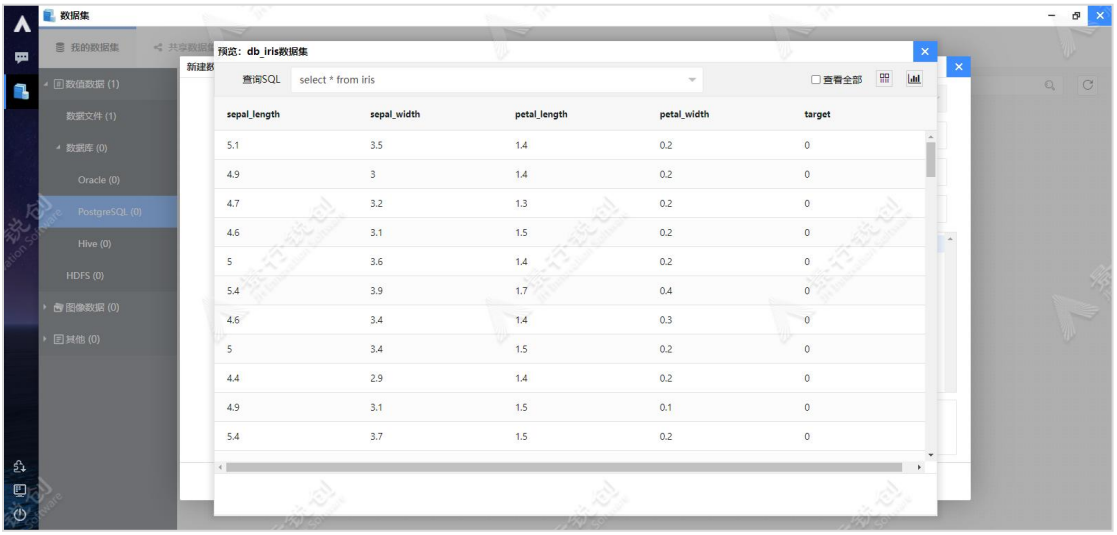
描述：输入该数据集的描述信息，可选。

填写完对应信息后，点击“连接测试”按钮，即可测试数据库的连通性，会有成功和失败提示。连接测试效果如下图所示：



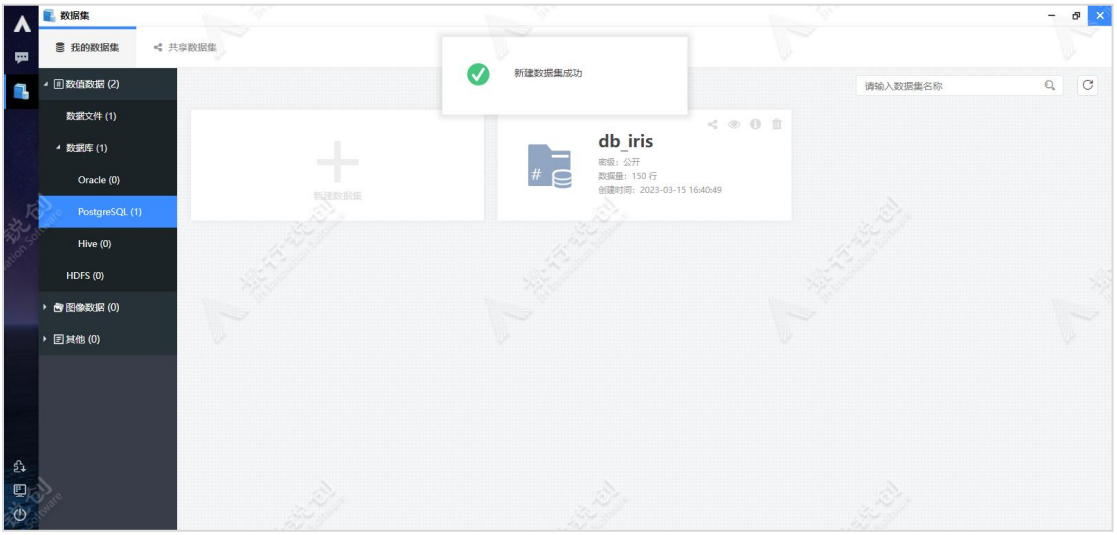
数据库连接测试

点击“预览”按钮，即可查看“查询 SQL”的结果。如下图所示：



预览“数值数据-PostgreSQL 数据库”数据集

点击“确定”按钮，新建数据集。如下图所示：



新建“数值数据-PostgreSQL 数据库”数据集成功示意图

2.1.1.3. “数值数据-HDFS”数据集

新建“数值数据-HDFS”数据集，操作步骤如下：依次点击“我的数据集”-“数值数据”-“HDFS”，然后在右侧的工作区，点击“新建数据集”卡片按钮，此时会弹出“新建数据集”窗口，如下图所示：

新建“数值数据-HDFS”数据集

图中每个参数的具体含义如下：

名称：数据集的名称，输入任意符合命名规则的名称即可。

密级：管理员开启密级功能后，显示此选项，默认为用户密级，用于数据安全、保密。

IP：输入 HDFS 服务节点的 IP。

端口号：输入 HDFS 服务使用的端口号。

用户名：输入 HDFS 服务的用户名。

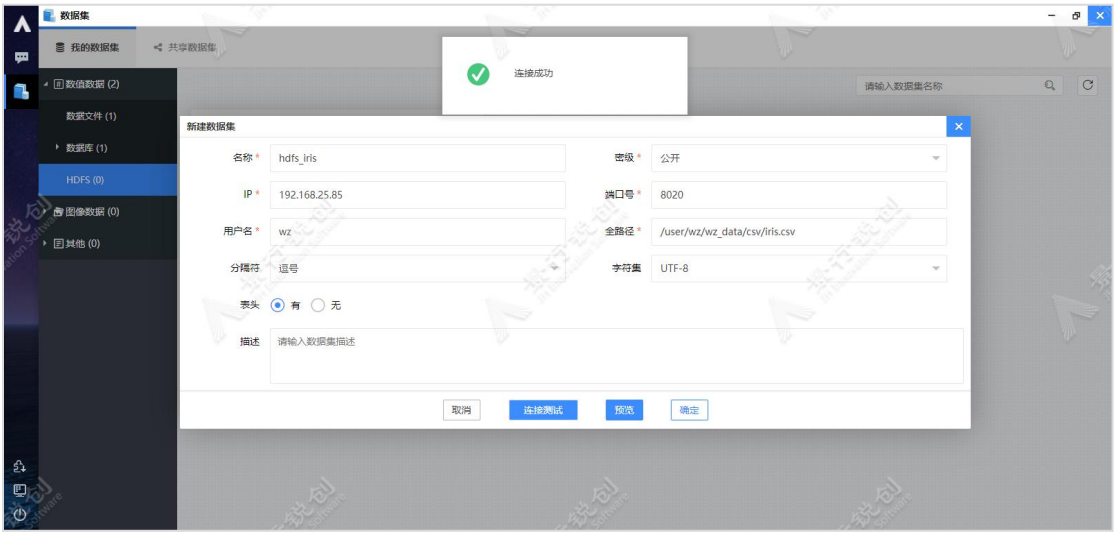
全路径：输入数据文件的绝对路径，目前仅支持 csv 格式的数据文件。

字符集：选择文件的编码格式。

分隔符：选择文件中使用的分隔符。

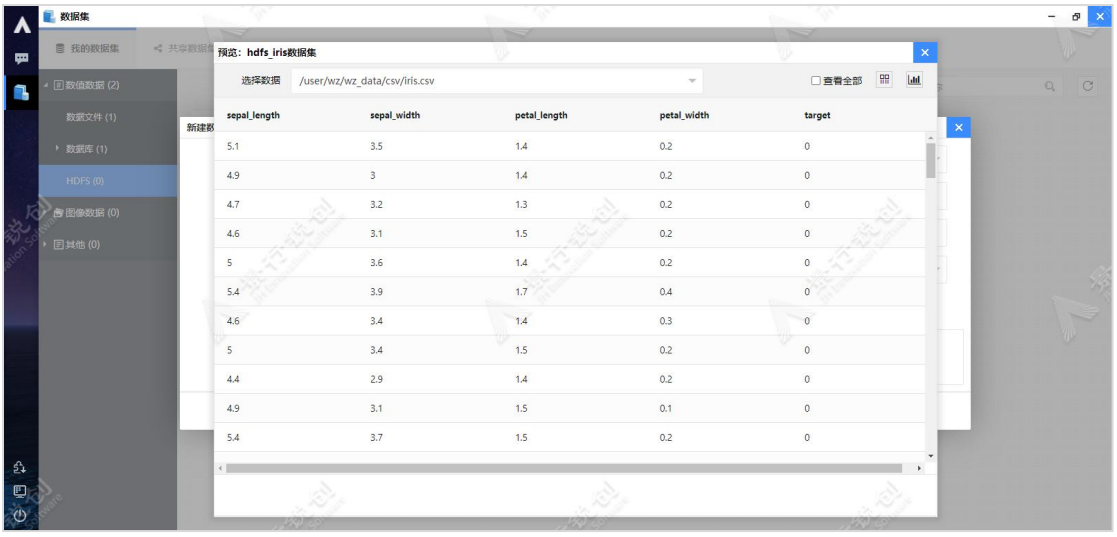
描述：输入该数据集的描述信息，可选。

填写完对应信息后，点击“连接测试”按钮，即可测试 HDFS 服务的连通性，会有成功和失败提示。连接测试效果如下图所示：



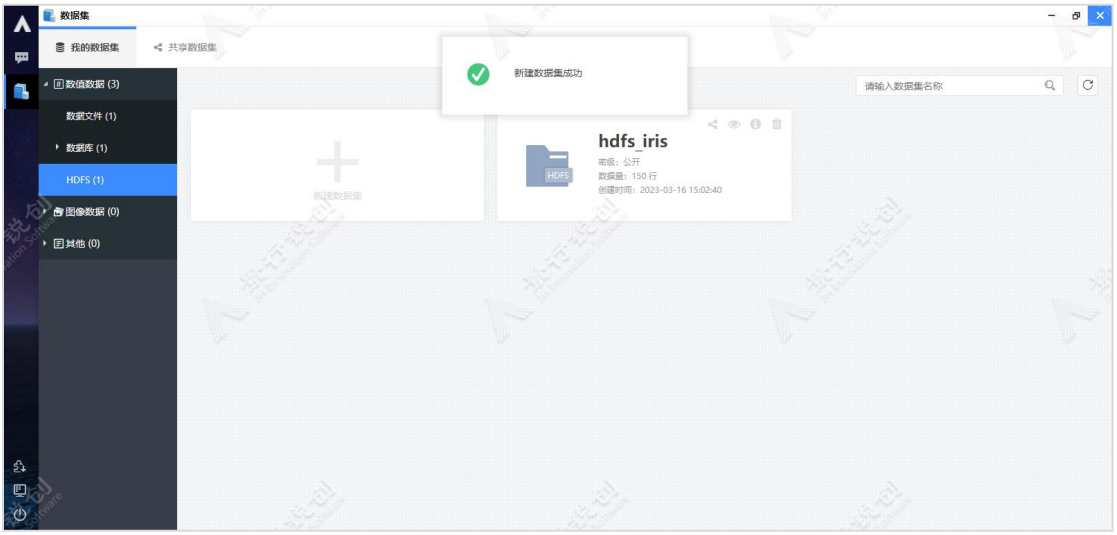
连接测试示意图

点击“预览”按钮，即可查看“查询 SQL”的结果。如下图所示：



预览“数值数据”示意图

点击“确定”按钮，新建数据集。如下图所示：

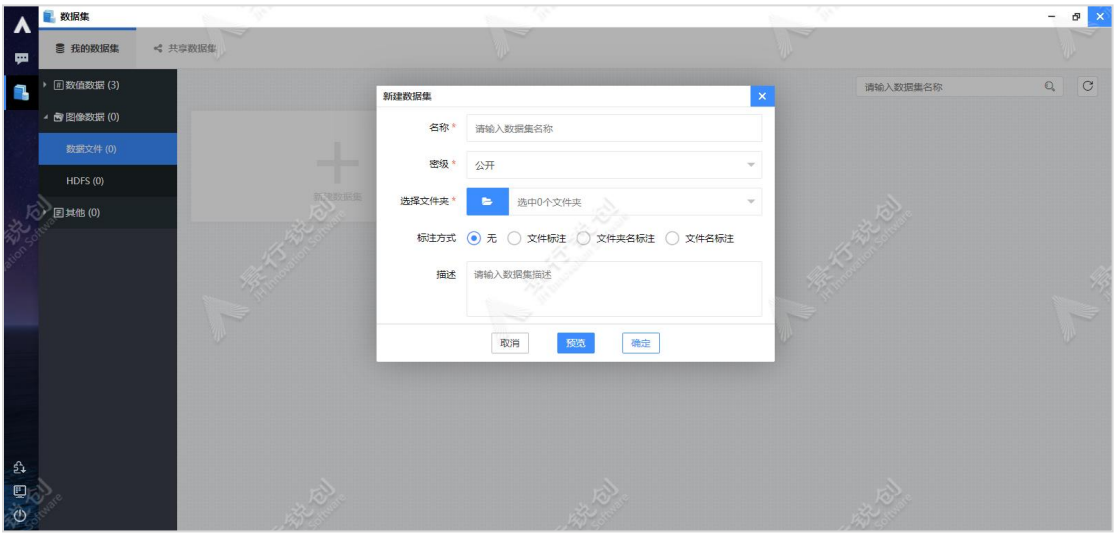


新建“数值数据-HDFS”数据集成功示意图

2.1.2. 图像数据

2.1.2.1. “图像数据-数据文件”数据集

新建“图像数据-数据文件”数据集，操作步骤如下：依次点击“我的数据集”-“图像数据”-“数据文件”，然后在右侧的工作区，点击“新建数据集”卡片按钮，此时会弹出“新建数据集”窗口，如下图所示：



新建“图像数据-数据文件”数据集

图中每个参数的具体含义如下：

名称：数据集的名称，输入任意符合命名规则的名称即可。

密级：管理员开启密级功能后，显示此选项，默认为用户密级，用于数据安全、保密。

选择文件夹：点击蓝色的“文件夹图标”按钮，然后在弹出的“选择文件夹”窗口中，选择图像文件夹即可。支持选择多个文件夹，程序仅支持 1 层目录，更多子文件夹内容自动忽略。

标注方式：有 4 种标注方式，如下所示：

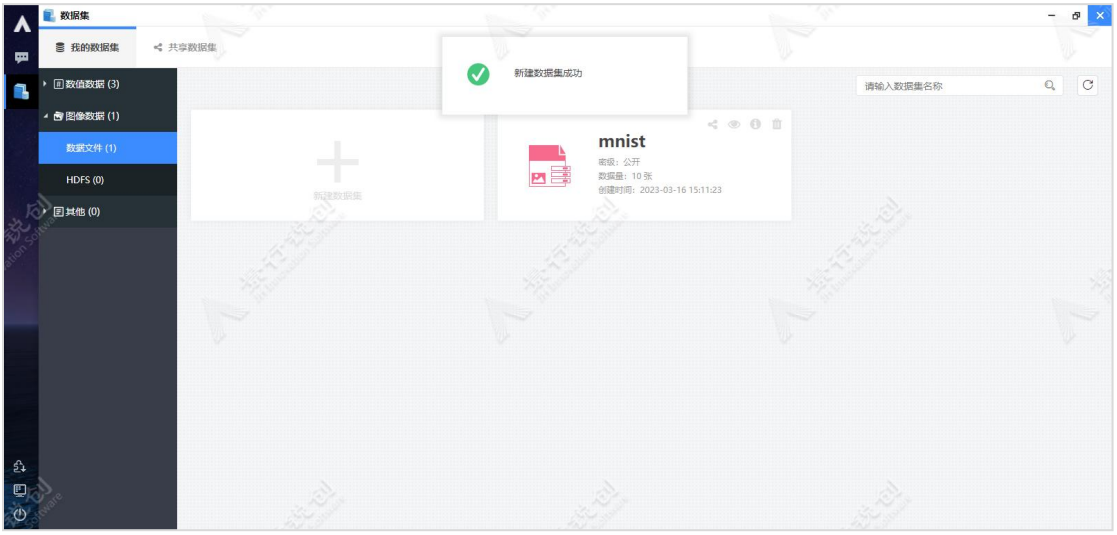
1) “无”：表示没有使用任何标注方式，例如：测试集；
2) “文件标注”：标签文件项选择与选择文件项对应的标签文件，目前仅支持“csv”、“json”和“xml”格式的标签文件；
3) “文件夹名标注”：每个文件夹就是一个类别的集合，并且通过文件夹名来标注类别；
4) “文件名标注”：图像的类别通过图像文件名称标识，例如：1-pic.png，即该图像中的内容是 1，此时使用了“标注信息-文件名”类别的标注方式；

描述：输入该数据集的描述信息，可选。

填写完对应的信息后，点击“预览”按钮，即可查看数据集的数据文件数据。预览效果如下图所示：

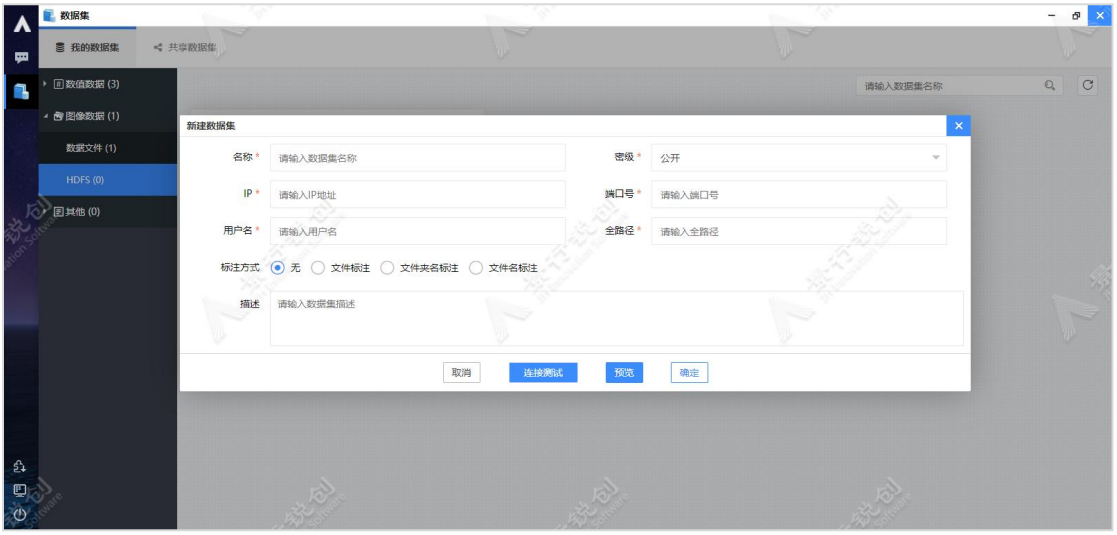
预览“图像数据-数据文件”数据集

点击“确定”按钮，新建数据集。如下图所示：



2.1.2.2. “图像数据-HDFS”数据集

新建“图像数据-HDFS”数据集，操作步骤如下：依次点击“我的数据集”-“图像数据”-“数据文件”，然后在右侧的工作区，点击“新建数据集”卡片按钮，此时会弹出“新建数据集”窗口，如下图所示：



新建“图像数据-HDFS”数据集

图中每个参数的具体含义如下：

名称：数据集的名称，输入任意符合命名规则的名称即可。

密级：管理员开启密级功能后，显示此选项，默认为用户密级，用于数据安

全、保密。

IP：输入 HDFS 服务节点的 IP。

端口号：输入 HDFS 服务使用的端口号。

用户名：输入 HDFS 服务的用户名。

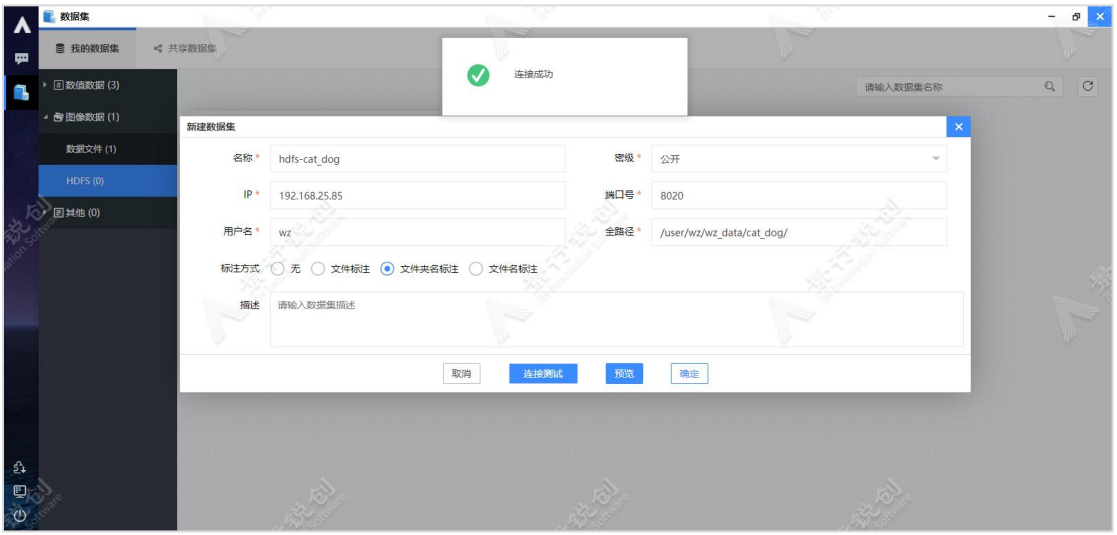
全路径：输入数据文件目录的绝对路径。

标注方式：有 4 种标注方式，如下所示：

- “无”：表示没有使用任何标注方式，例如：测试集；
- “文件标注”：标签文件项选择与选择文件项对应的标签文件，目前仅支持“csv”、“json”和“xml”格式的标签文件；
- “文件夹名标注”：每个文件夹就是一个类别的集合，并且通过文件夹名来标注类别；
- “文件名标注”：图像的类别通过图像文件名称标识，例如：1-pic.png，即该图像中的内容是 1，此时使用了“标注信息-文件名”类别的标注方式；

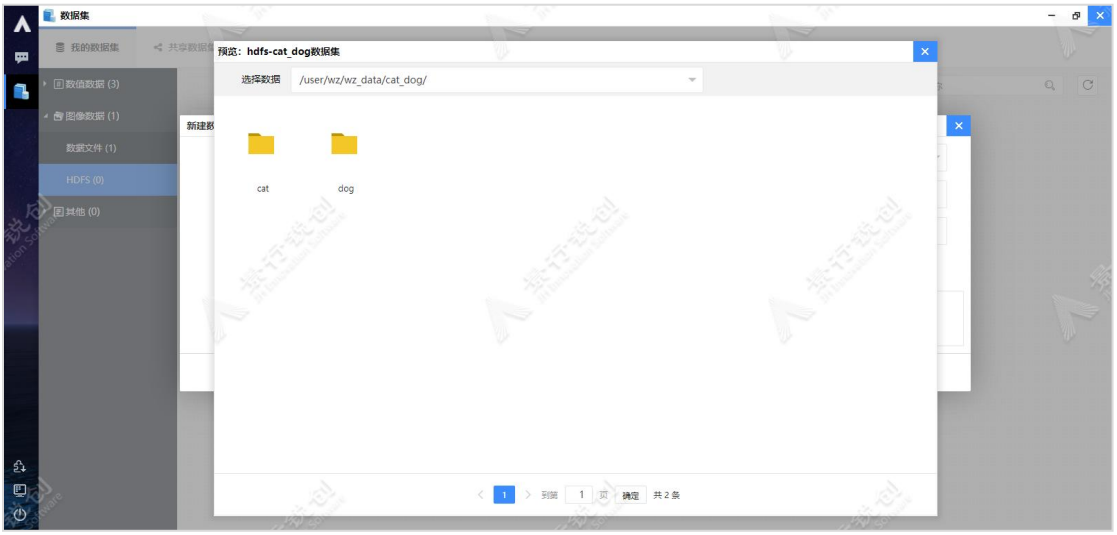
描述：输入该数据集的描述信息，可选。

填写完对应信息后，点击“连接测试”按钮，即可测试 HDFS 服务的连通性，会有成功和失败提示。连接测试效果如下图所示：



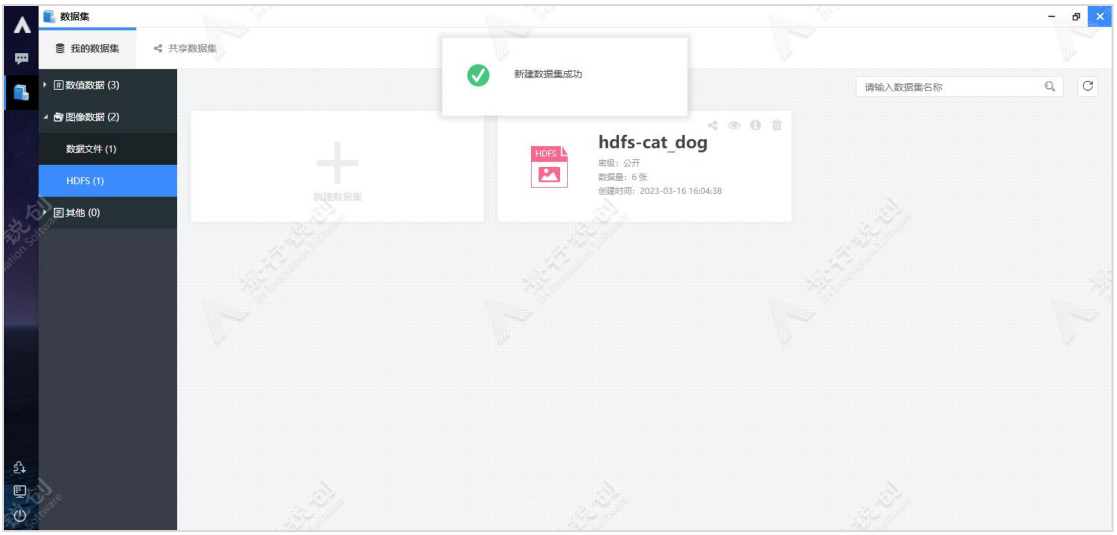
连接测试示意图

点击“预览”按钮，即可查看数据集的数据文件数据。如下图所示：



预览“图像数据-HDFS”数据集示意图

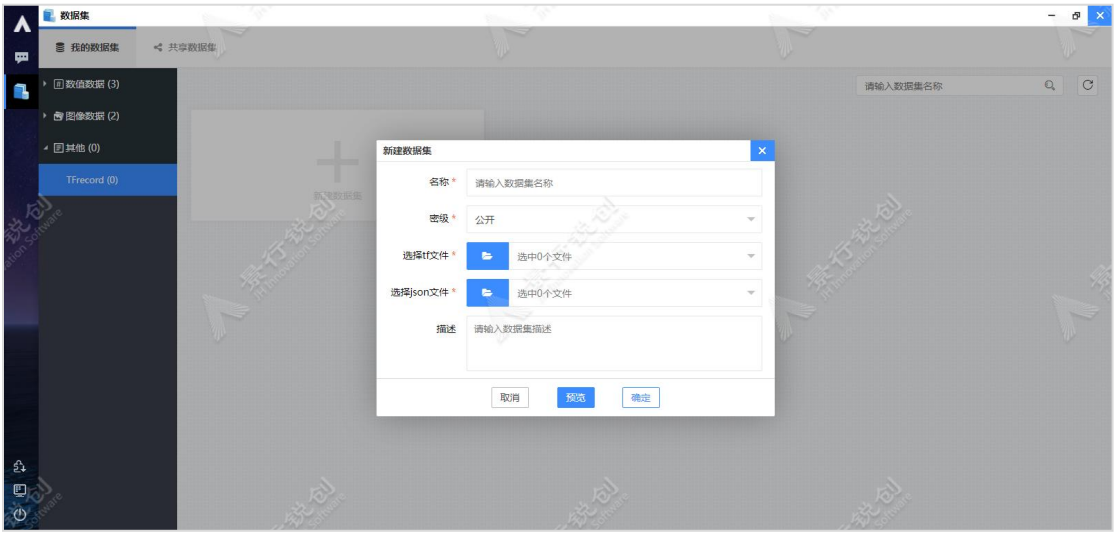
点击“确定”按钮，新建数据集。如下图所示：



新建“图像数据-数据文件”数据集成功示意图

2.1.3. 其他

新建“其他-TFrecord”数据集，操作步骤如下：依次点击“我的数据集”-“其他”-“TFrecord”，然后在右侧的工作区，点击“新建数据集”卡片按钮，此时会弹出“新建数据集”窗口，如下图所示：



新建“其他-TFrecord”数据集示意图

图中每个参数的具体含义如下：

名称：数据集的名称，输入任意符合命名规则的名称即可。

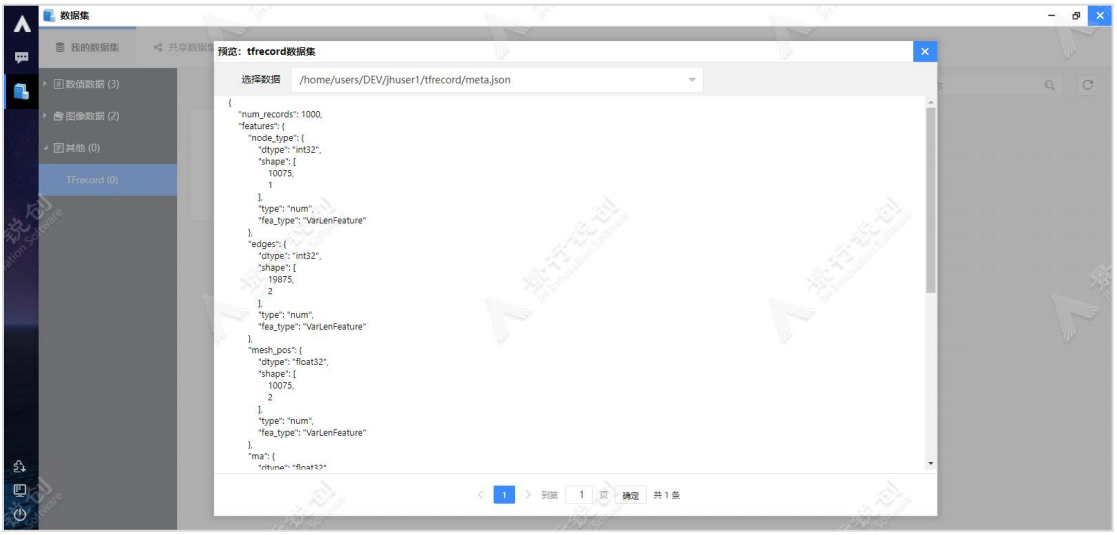
密级：管理员开启密级功能后，显示此选项，默认为用户密级，用于数据安全、保密。

选择 tf 文件：点击蓝色的“文件图标”按钮，然后在弹出的“选择文件”窗口中，选择 tfrecord 文件即可。只允许支持选择一个 tfrecord 文件。

选择 json 文件：点击蓝色的“文件图标”按钮，然后在弹出的“选择文件”窗口中，选择 json 文件即可。只允许支持选择一个 json 文件。

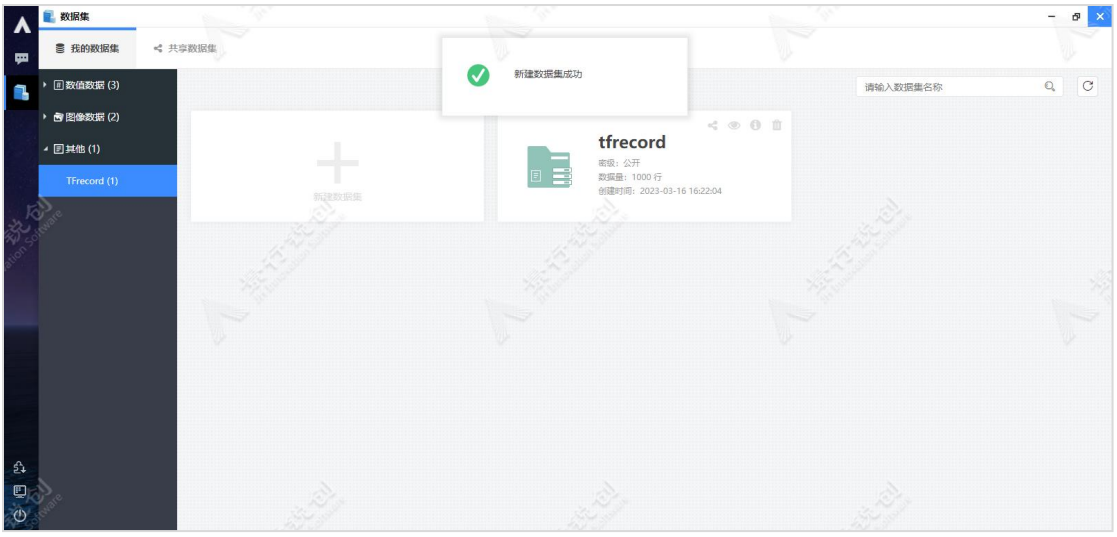
描述：输入该数据集的描述信息，可选。

点击“预览”按钮，即可查看数据集的数据文件数据。如下图所示：



预览“其他-TFrecord”数据集示意图

点击“确定”按钮，新建数据集。如下图所示：



新建“其他-TFrecord”数据集成功示意图

2.1.4. 共享数据集

共享数据集的产生有 2 种方式：

- 在我的数据集中，将我的数据集共享至共享数据集；
- 在共享数据集中，应用共享组中的数据文件，新建共享数据集；

各种角色对共享数据集的操作权限：

共享者：对共享数据集有修改、预览、使用和删除的权限；

共享组用户：对共享数据集有预览、使用的权限；

管理员：对共享数据集仅有删除权限；

其他用户：看不到共享数据集；

用途：共享组成员可以在方案设计和 AI 作业中使用共享数据集。

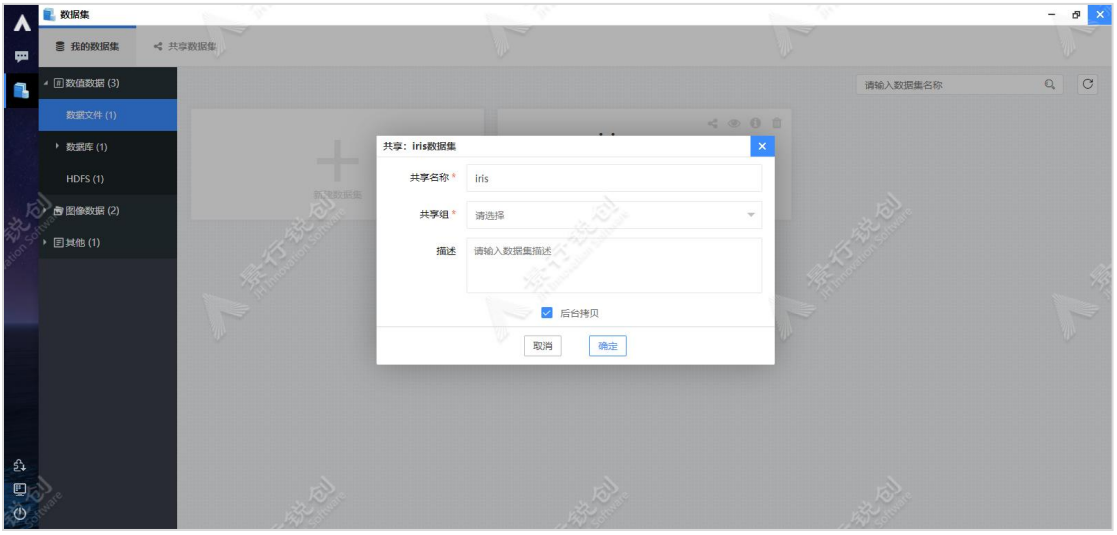
注意：进行共享操作前，需要先创建共享组。

2.1.4.1. 将我的数据集共享至共享数据集

共享数据文件类的数据集，如下所示：

- “数值数据-数据文件”数据集
- “图像数据-数据文件”数据集
- “其他-TFrecord”数据集

以共享“数值数据-数据文件”数据集为例，步骤如下：依次点击“我的数据集”-“数值数据”-“数据文件”，然后在右侧的工作区，点击数据集卡片上的“共享”图标按钮，此时会弹出“共享”窗口，如下图所示：



数据文件共享至共享数值数据集

图中每个参数的具体含义如下：

共享名称：共享后的数据集名称，输入任意符合命名规则的名称即可。

共享组：选择可见的共享组，选项从我的数据->共享数据区获取。

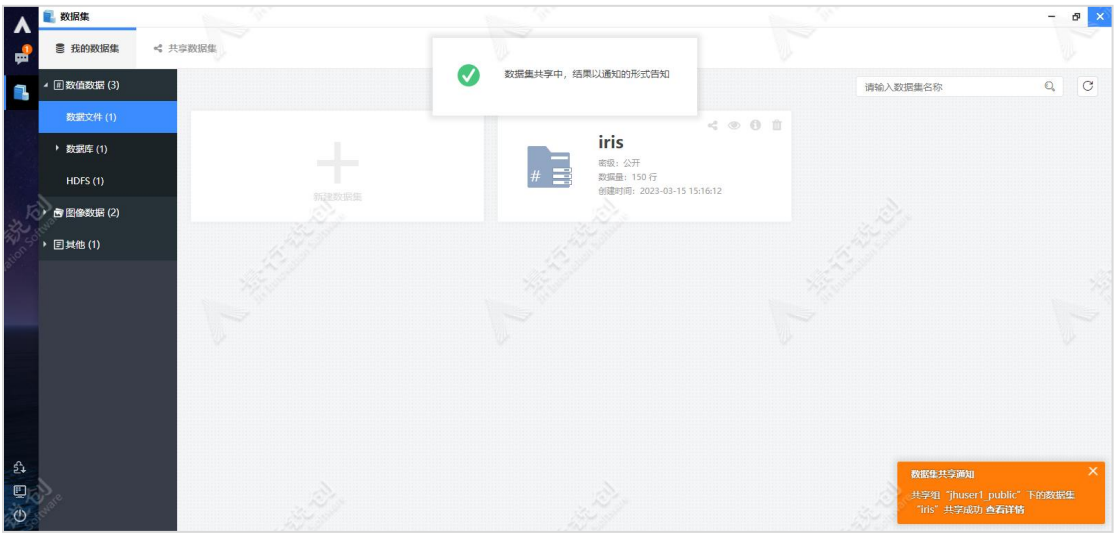
共享用户：选择共享组之后显示，显示为选择的共享组成员。

共享目录：选择共享组之后显示，只能在所选共享组的共享数据区选择文件夹，点击蓝色的“文件夹图标”按钮，然后在弹出的“选择文件夹”窗口中选择文件夹即可，用作存放源数据文件目录，只允许选择一个文件夹。

描述：共享后的数据集描述信息，可选。

后台拷贝：默认勾选，采用后台形式拷贝源数据集数据文件至共享目录，当前共享页面退出；不勾选则采用前台拷贝方式，当前共享页面直至共享结束退出；

填写完成后，点击“确定”按钮，进行数据集共享，如下图所示：



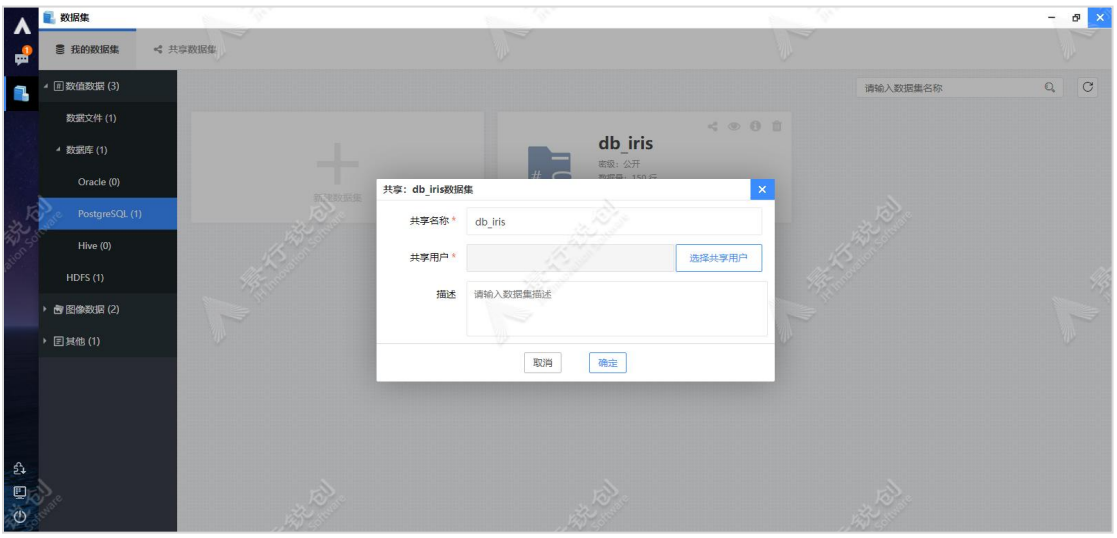
共享数据集示意图

共享非“数据文件”类的数据集，如下所示：

- “数值数据-数据库”数据集
- “数值数据-HDFS”数据集
- “图像数据-HDFS”数据集

以共享“数值数据-数据库-PostgreSQL”数据集为例，操作步骤如下：依次点击“我的数据集”-“数值数据”-“数据库”-“PostgreSQL”，然后在右侧

的工作区，点击数据集卡片上的“共享”按钮，此时会弹出“共享”窗口，如下图所示：



PostgreSQL 共享至共享数值数据集

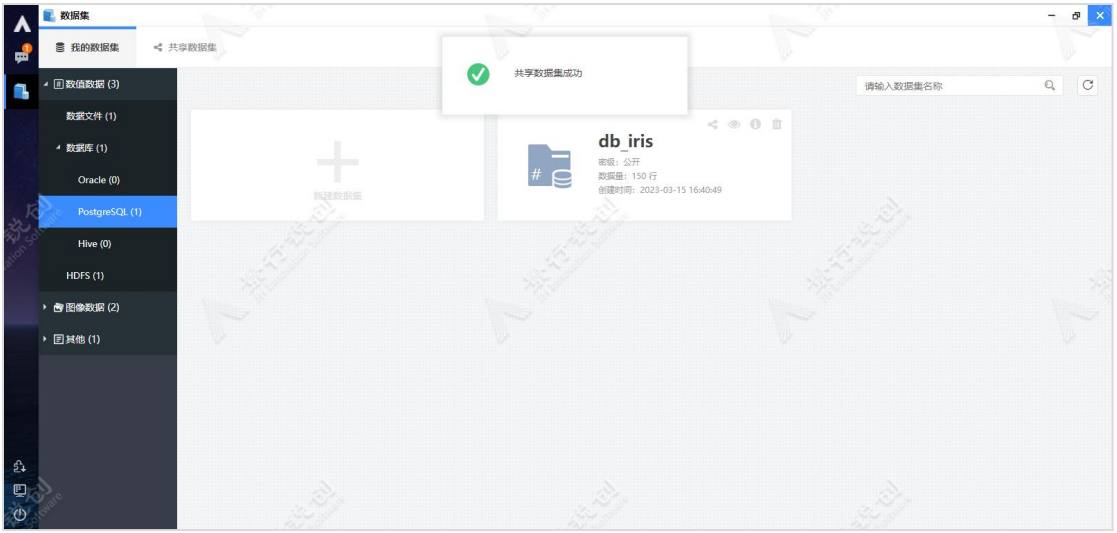
图中每个参数的具体含义如下：

共享名称：共享后的数据集名称，输入任意符合命名规则的名称即可。

共享用户：选择共享用户。

描述：共享后的数据集描述信息，可选。

填写完成后，点击“确定”按钮，共享数据集，如下图所示：



2.1.4.2. 新建共享数据集

新建数据文件类的共享数据集，如下所示：

- 数值数据->数据文件
- 图像数据->数据文件
- 其他->TFrecord

以新建“数值数据-数据文件”共享数据集为例，操作步骤如下：依次点击“共享数据集”-“数值数据”-“数据文件”，然后在右侧的工作区，点击“新建共享数据集”卡片按钮，此时会弹出“新建共享数据集”窗口，如下图所示：

使用数据文件添加共享数值数据集

图中每个参数的具体含义如下：

名称：同我的数据集中数值数据->数据文件。

密级：同我的数据集中数值数据->数据文件。

共享组：选择可见的共享组，选项从我的数据->共享数据区获取。

共享用户：选择共享组之后显示，显示为选择的共享组成员。

选择文件：选择共享组之后显示，只能在所选共享组的共享数据区选择文件，同我的数据集中数值数据->数据文件。

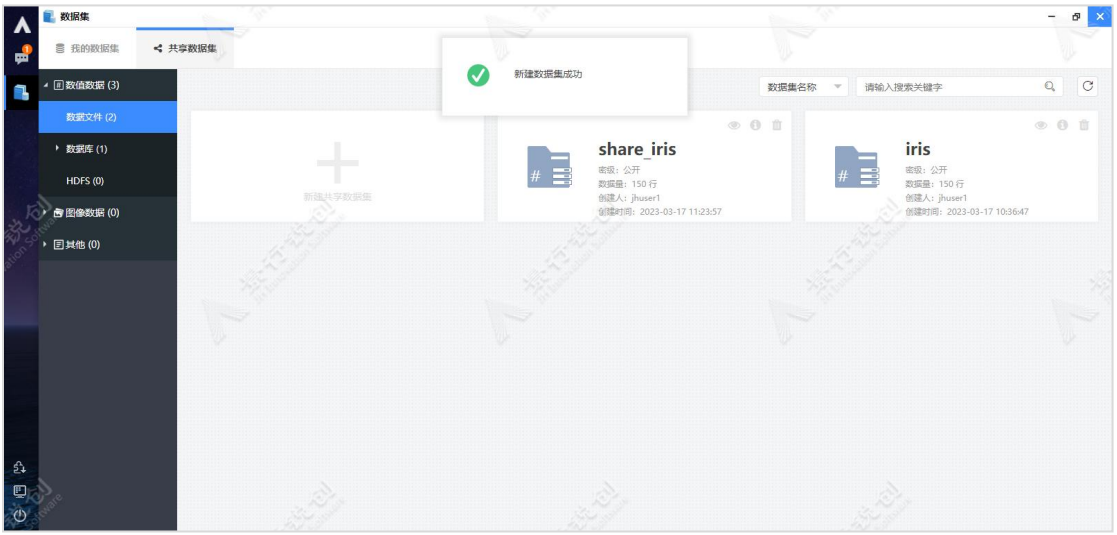
字符集：同我的数据集中数值数据->数据文件。

分隔符：同我的数据集中数值数据->数据文件。

表头：同我的数据集中数值数据->数据文件。

描述：同我的数据集中数值数据->数据文件。

填写完成后，点击“确定”按钮，新建共享数据集。如下图所示：

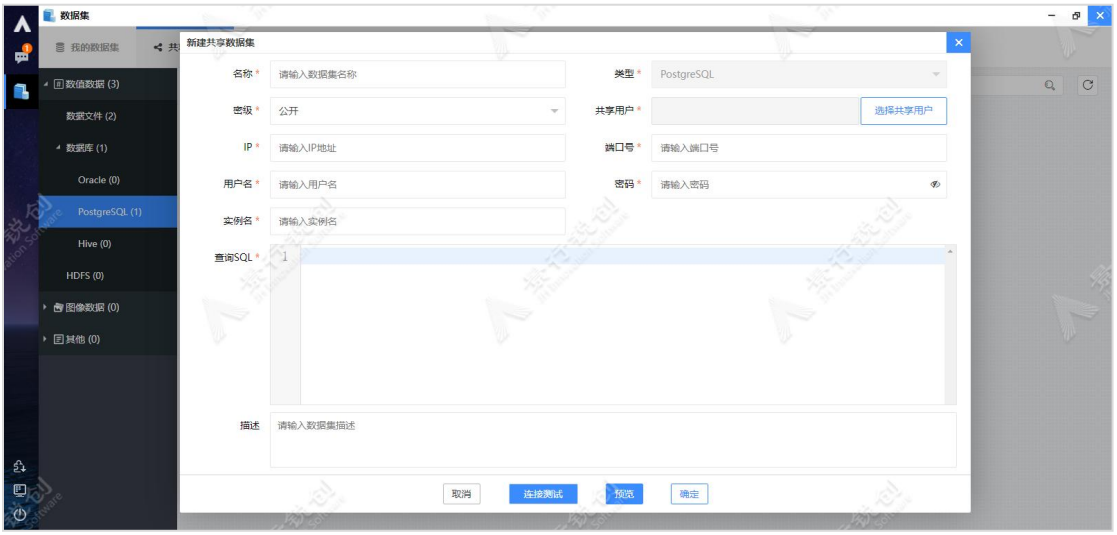


新建共享数据集示意图

共享非“数据文件”类的数据集，如下所示：

- “数值数据-数据库”数据集
- “数值数据-HDFS”数据集
- “图像数据-HDFS”数据集

以新建“数值数据-数据库-PostgreSQL”共享数据集为例，操作步骤如下：依次点击“共享数据集”-“数值数据”-“数据库”-“PostgreSQL”，然后在右侧的工作区，点击“新建共享数据集”卡片按钮，此时会弹出“新建共享数据集”窗口，如下图所示：



使用数据库添加数值数据集

图中每个参数的具体含义如下：

名称：同我的数据集中数值数据->数据库->PostgreSQL。

类型：同我的数据集中数值数据->数据库->PostgreSQL。

密级：同我的数据集中数值数据->数据库->PostgreSQL。

共享用户：选择共享用户。

IP：同我的数据集中数值数据->数据库->PostgreSQL。

端口号：同我的数据集中数值数据->数据库->PostgreSQL。

用户名：同我的数据集中数值数据->数据库->PostgreSQL。

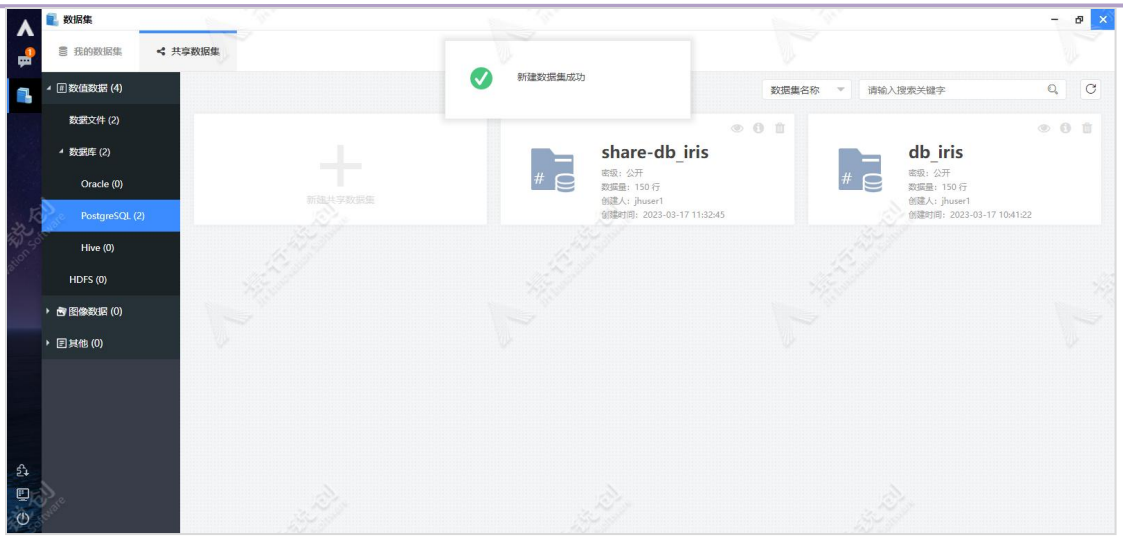
密码：同我的数据集中数值数据->数据库->PostgreSQL。

实例名：同我的数据集中数值数据->数据库->PostgreSQL。

查询 SQL：同我的数据集中数值数据->数据库->PostgreSQL。

描述：同我的数据集中数值数据->数据库->PostgreSQL。

填写完成后，点击“确定”按钮，新建共享数据集。如下图所示：



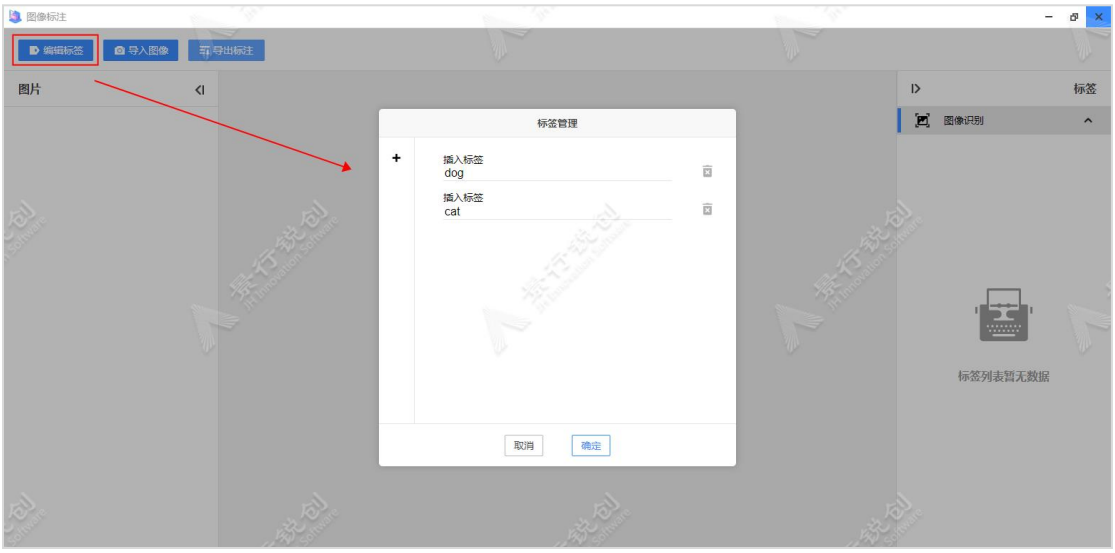
2.1.5. 图像标注

图像标注用于对图像进行手动标注，支持图像识别和目标检测两种类型的标注。目标检测中支持矩形标注、点标注、线标注以及多边形标注，标注结果可以导出为多种格式的标注文件。

进入图像标注主界面，默认应用图像识别，如下图所示：

2.1.5.1. 编辑标签

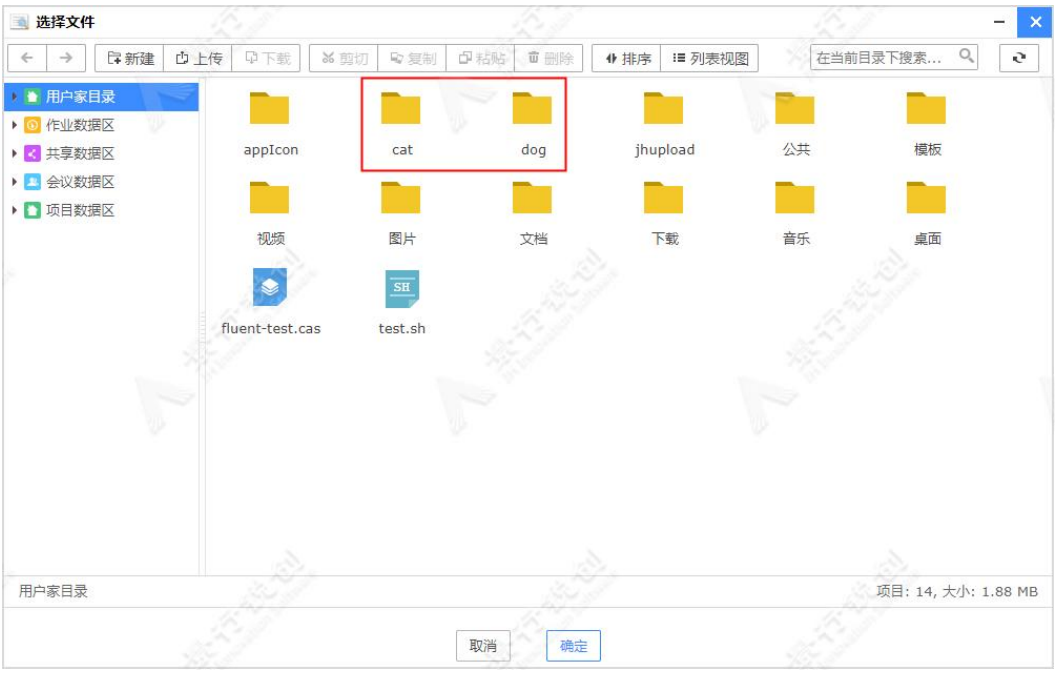
点击编辑标签按钮，弹出标签管理弹窗。点击加号（“+”）按钮，便可对图像数据标签进行添加，标签添加成功后，可在图像标注主界面的右侧区域查看标签列表。以添加 cat 和 dog 两个标签为例，具体操作如下图所示：



2.1.5.2. 导入图像

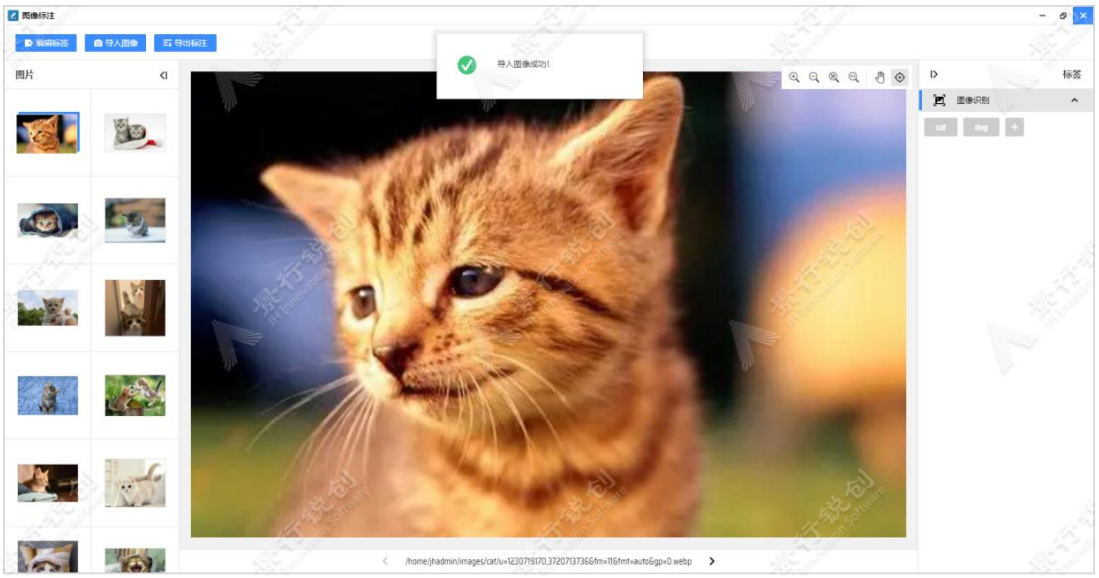
点击导入图像按钮，打开导入图像弹窗，用于告知导入图像的格式要求，如下图所示：

点击导入按钮，会弹出选择文件窗口。在选择文件弹窗中，可以选择图像文件或文件夹，选择一个包含各种格式的文件夹，会自动忽略非图像格式的文件。以导入用户家目录下的 cat 和 dog 目录为例，具体操作如下图所示：



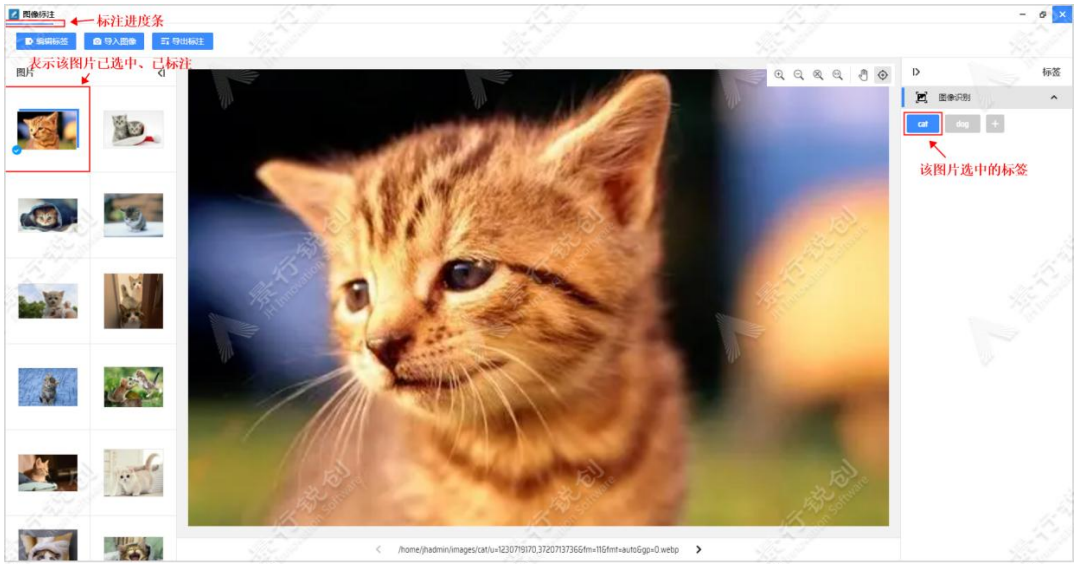
注：按住 ctrl 键，选择需要的文件或目录，便可完成多选。

导入图像成功后，会在左侧的图片区域加载图片列表，并且在画布区域默认加载第一张图片，如下图所示：

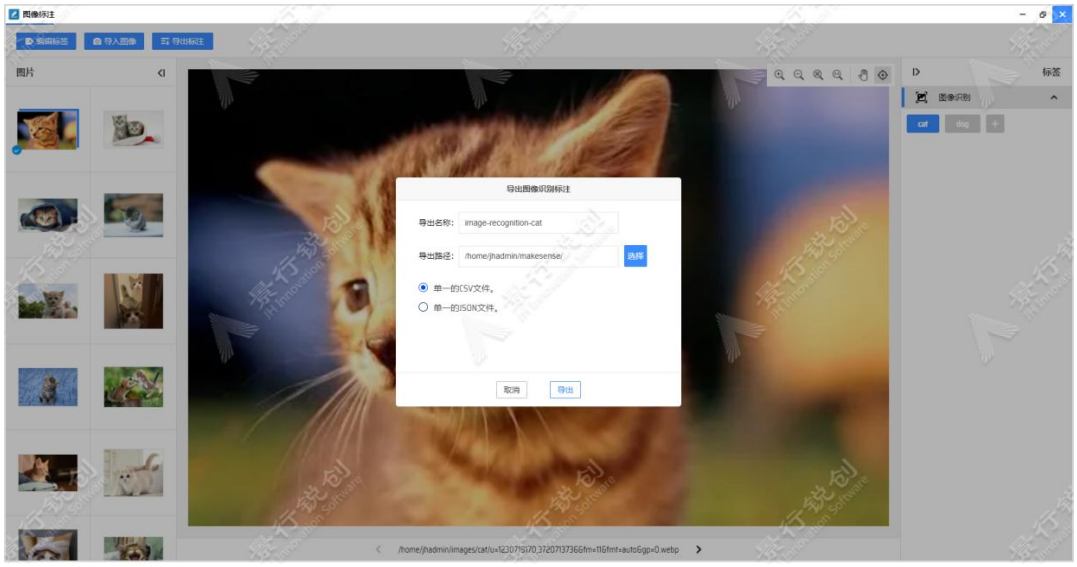


2.1.5.3. 图像识别

图像标注工具支持图像识别，标注图像数据文件属于某一分类或多个分类，点击右侧标签列表中的标签对当前选中的图片进行标注，如下图所示：



点击左上角的“导出标注”按钮，会弹出“导出图像识别标注”窗口，如下图所示：



图中每个参数的具体含义如下所示：

导出名称：默认为 my-project-name，可修改。

导出路径：选择标注文件的存储路径。

导出文件类型：支持导出为 csv 和 json 格式的标注文件。

在上图中填写完对应信息后，点击导出按钮，将标注结果导出。在选择的存储路径中查看标注文件内容，如下图所示：

```
pic_path,label
/home/jhadmin/images/cat/u=1230719170,3720713736&fm=11&fmt=auto&gp=0.webp,"[cat]"
```

数据详情：

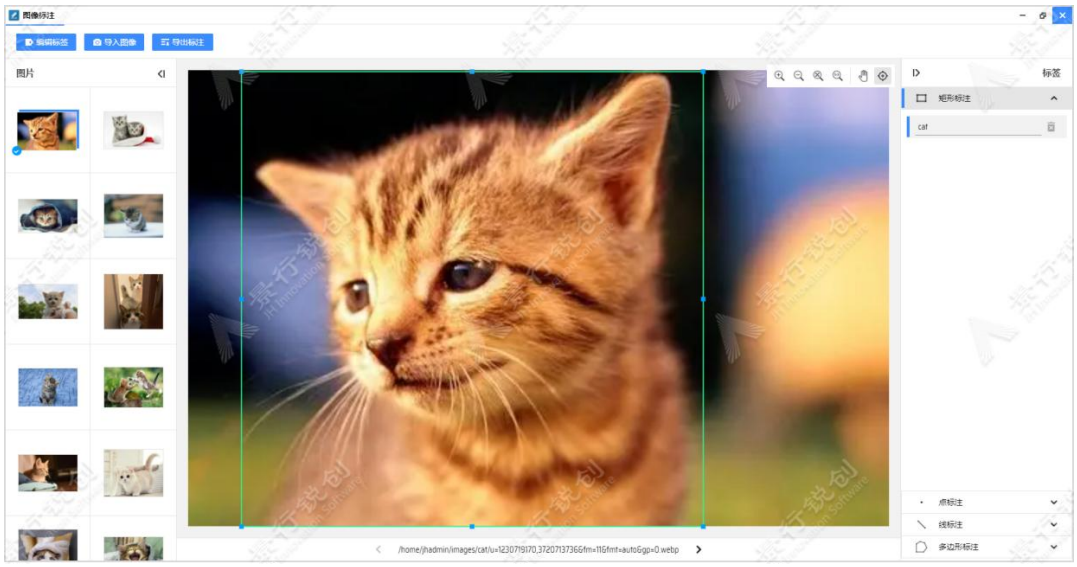
- pic_path：第一列表示该图片的路径。
- label：第二列表示该图片选中的标签集合

2.1.5.4. 目标检测

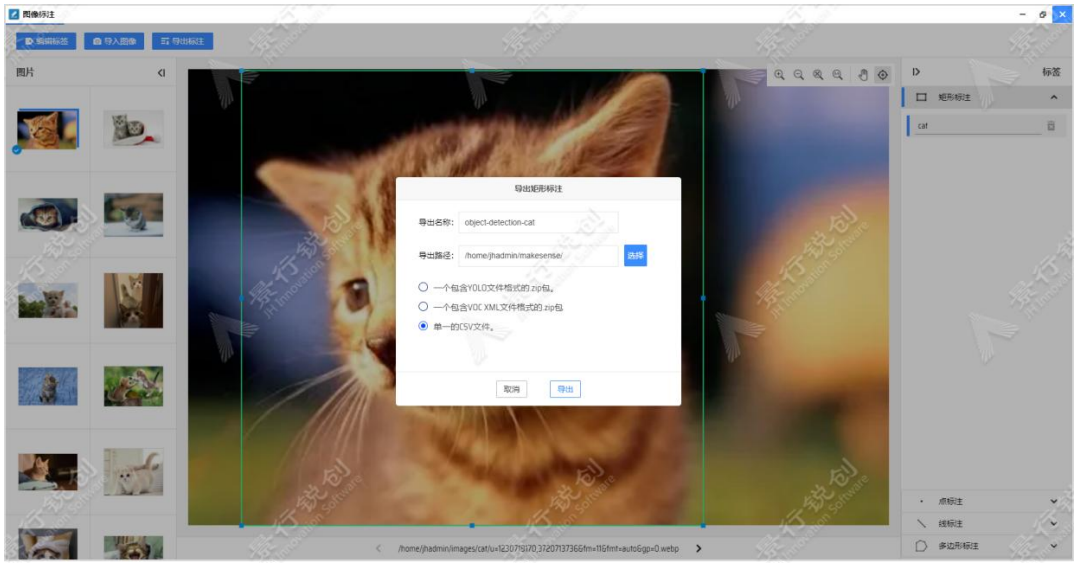
选中目标检测单选项，进入目标检测主页：

(1) 矩形标注

默认使用矩形标注，对图像文件中的矩形区域进行标注，使用矩形框将所要标注的局部图像框起来，然后选择相应的标签，如下图所示：



点击左上角的“导出标注”按钮，会弹出“导出矩形标注”窗口，如下图所示：



图中每个参数的具体含义如下：

导出名称：默认为 my-project-name，可修改。

导出路径：选择标注文件的存储路径。

导出文件类型：支持导出为 YOLO、VOC XML 和 CSV 格式的标注文件。

在上图中填写完对应信息后，点击导出按钮，将标注结果导出。在选择的存储路径中查看标注文件内容，如下图所示：

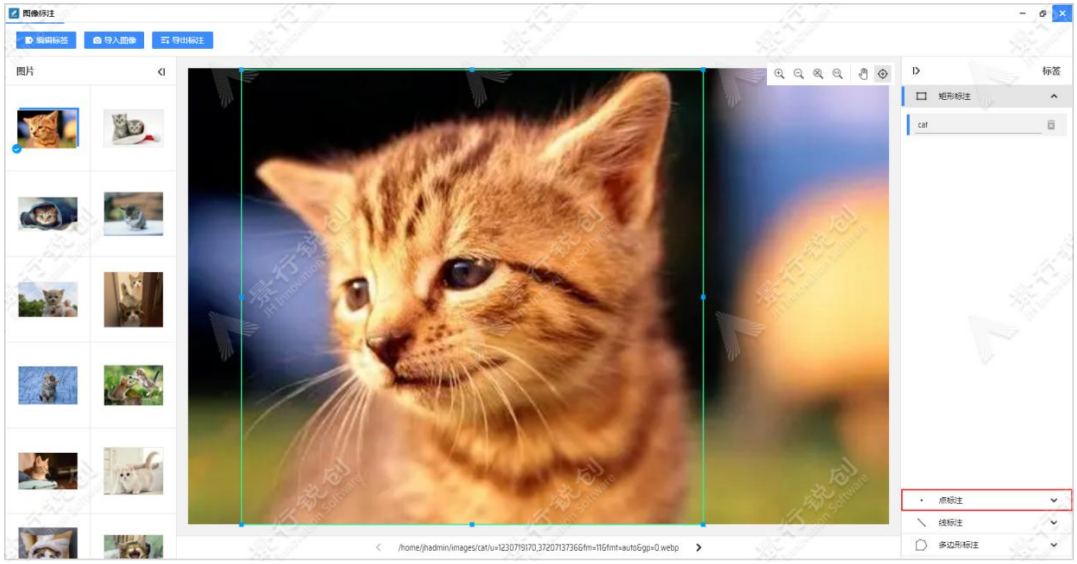
```
label, xmin, ymin, boxw, boxh, pic_path, width, height
cat, 37, 2, 325, 323, /home/jhadmin/images/cat/u=1230719170, 3720713736&fm=11&fmt=auto&gp=0.webp, 500, 325
```

标注详情：

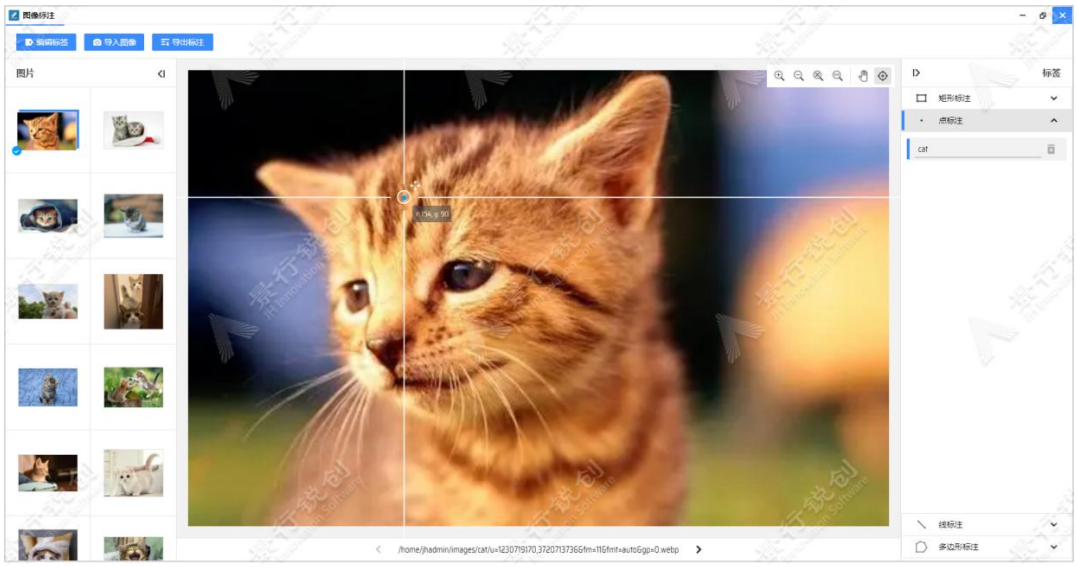
- label：第一列表示对该图片局部区域的图像选中的标签
- xmin：第二列表示矩形框左上角的 x 轴坐标
- ymin：第三列表示矩形框左上角的 y 轴坐标
- boxw：第四列表示矩形框的宽度
- boxh：第五列表示矩形框的高度
- pic_path：第六列表示该图片的路径
- width：第七列表示该图片的宽度
- height：第八列表示该图片的高度

(2) 点标注

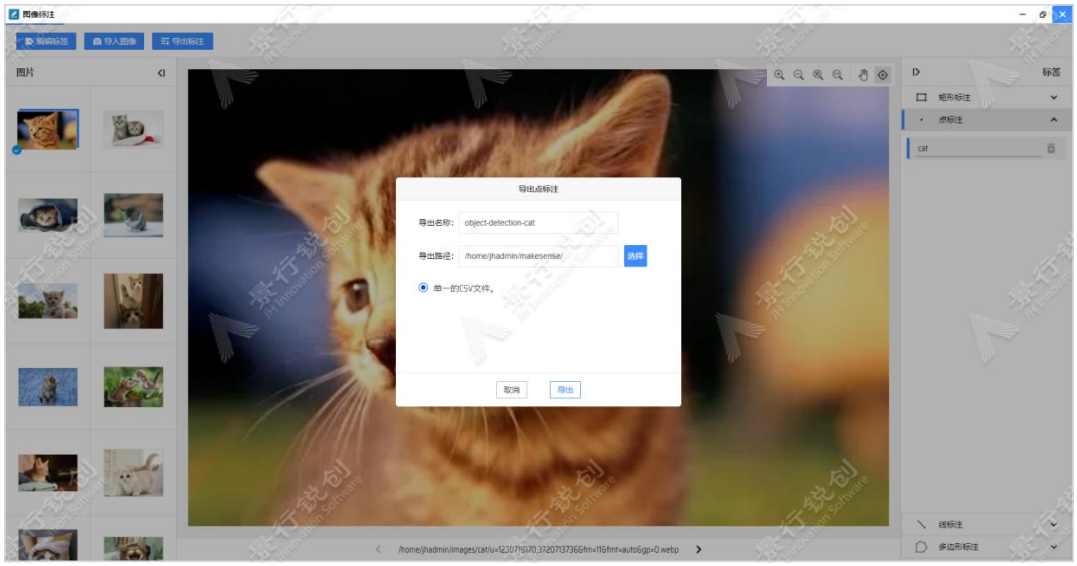
点击点标注切换至点标注模式，如下图所示：



对图像文件中的点域进行标注，在画布图片上进行点击，便可进行点标注，如下图所示：



点击左上角“导出标注”按钮，会弹出“导出点标注”窗口，如下图所示：



图中每个参数的具体含义如下：

导出名称：默认为 my-project-name，可修改。

导出路径：选择标注文件存储的文件夹。

导出文件类型：仅支持导出 csv 格式的标注文件。

在上图中填写完对应信息后，点击导出按钮，将标注结果导出。在选择的存储路径中查看标注文件内容，如下图所示：

```
label,x,y,pic_path,width,height
cat,180,93,/home/jhadmin/images/cat/u=1230719170,3720713736&fm=11&fmt=auto&gp=0.webp,500,325
```

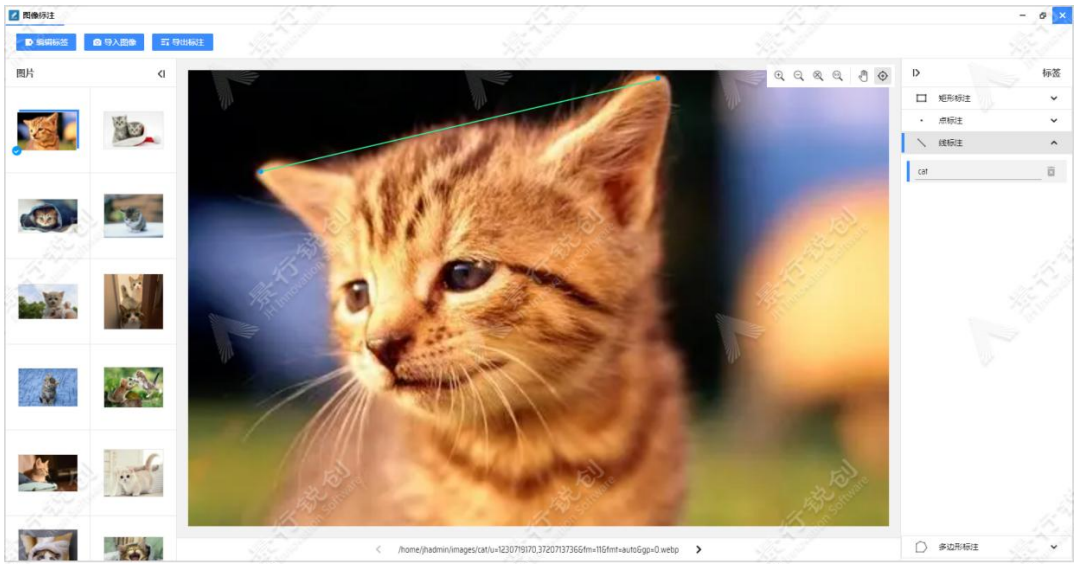
标注详情：

- label：第一列表示对该点标注选中的标签
- x：第二列表示该点标注的 x 轴坐标
- y：第三列表示该点标注的 y 轴坐标
- pic_path：第四列表示该图片的路径。
- width：第五列表示该图片的宽度
- height：第六列表示该图片的高度

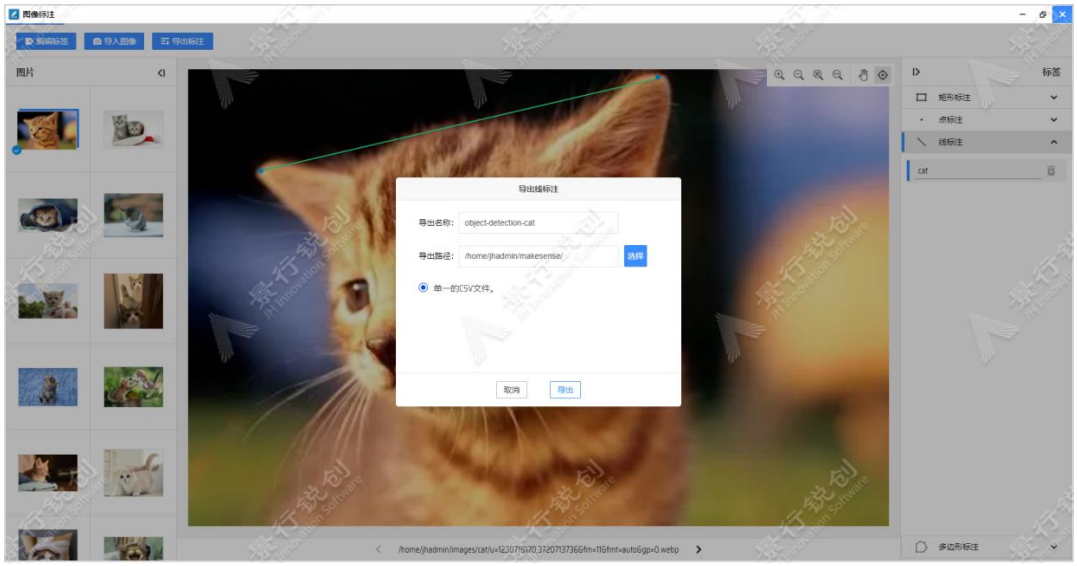
(3) 线标注

点击线标注切换至线标注模式，如下图所示：

通过鼠标在画布区域中的图片中点击两次，两点一线，便可进行线标注，如下图所示：



点击左上角的“导出标注”按钮，会弹出“导出线标注”窗口，如下图所示：



图中每个参数的具体含义如下：

导出名称：默认为 my-project-name，可修改。

导出路径：选择标注文件存储的文件夹。

导出文件类型：仅支持导出 csv 格式的标注文件。

在上图中填写完对应信息后，点击导出按钮，将标注结果导出。在选择的存储路径中查看标注文件内容，如下图所示：

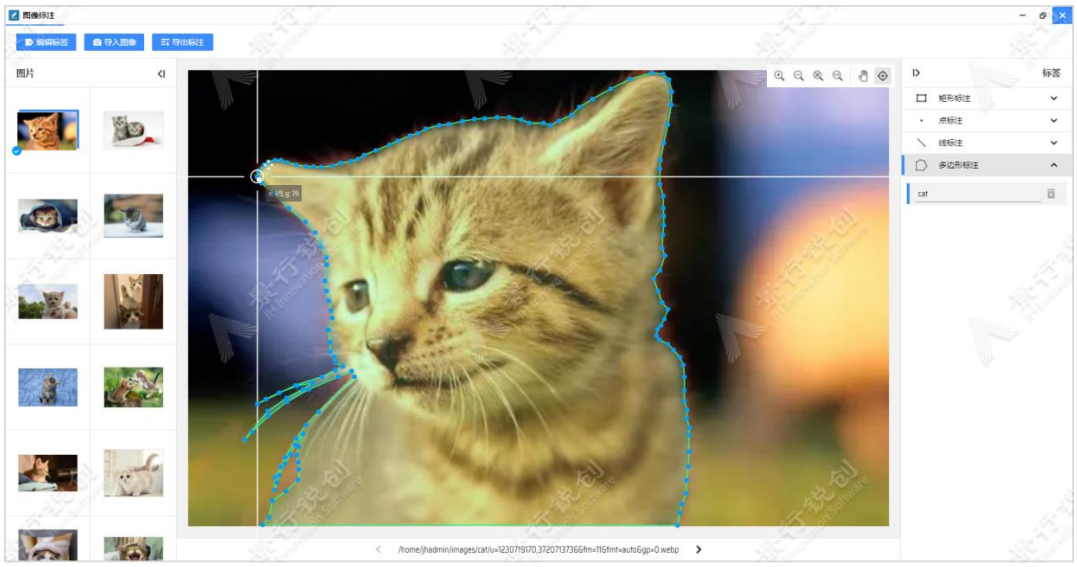
```
label,xmin,ymin,xmax,ymax,pic_path,width,height
cat,49,73,338,5,/home/jhadmin/images/cat/u=1230719170,3720713736&fm=11&fmt=auto&gp=0.webp,500,325
```

标注详情：

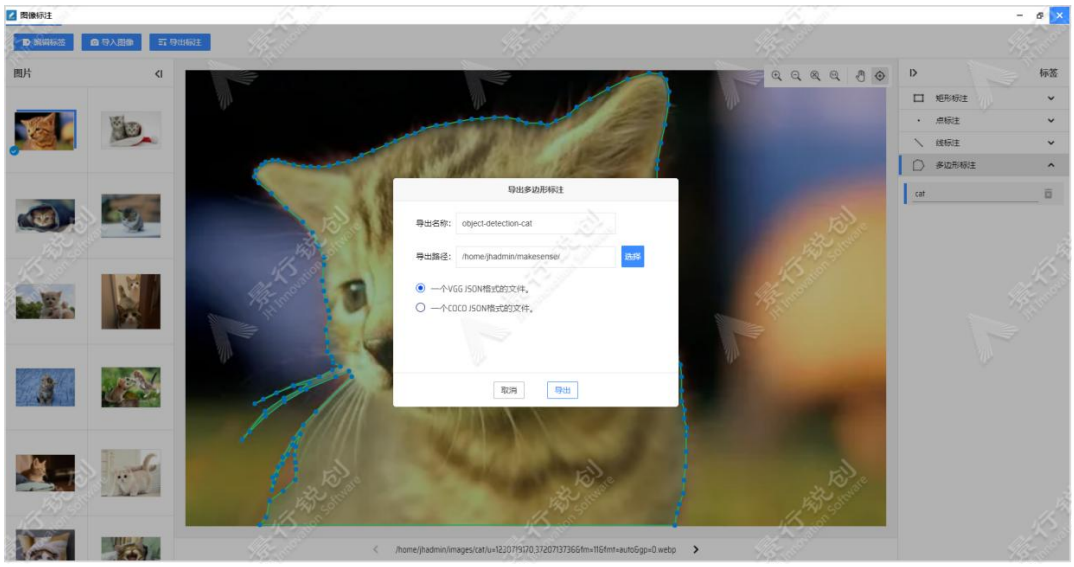
- label：第一列表示对该线标注选中的标签
- xmin：第二列表示起始点的 x 轴坐标
- ymin：第三列表示起始点的 y 轴坐标
- xmax：第四列表示终止点的 x 轴坐标
- ymax：第五列表示终止点的 y 轴坐标
- pic_path：第六列表示该图片的路径。
- width：第七列表示该图片的宽度
- height：第八列表示该图片的高度

(4) 多边形标注

点击多边形标注切换至多边形标注模式，通过鼠标在画布区域的图片上，多次点击产生的点连成一条线且首尾呼应，圈出一个不规则的区域，从而进行更加精确的标注，便可进行多边形标注，如下图所示：



点击左上角的“导出标注”按钮，会弹出“导出多边形标注”窗口，如下图所示：



图中每个参数的具体含义如下：

导出名称：默认为 my-project-name，可修改。

导出路径：选择标注文件存储的文件夹。

导出文件类型：支持导出为 VGG JSON 和 COCO JSON 格式的标注文件。

在上图中填写完对应信息后，点击导出按钮，将标注结果导出。在选择的存储路径中查看标注文件内容，如下图所示：

{"/home/jhadmin/images/cat/u=1230719170,3720713736&fm=11&fmt=auto&gp=0.webp":{"fileref":"","size":13634,"filename":"/home/jhadmin/images/cat/u=1230719170,3720713736&fm=11&fmt=auto&gp=0.webp","base64_img_data":"","file_attributes":{},"regions":{"0":{"shape_attributes":{"name":"polygon","all_points_x":[48.8095238052384,53.69674185463662,62.24937343358398,72.83834586466168,85.05639097744363,94.42355889724313,100.9398496240601 7,113.56516290726819,123.33959899749375,132.70676691729324,142.88847117794487,145.33208020050128,155.5137844611529,168.54636591478697,183.20802005012533,198.6842105263158,212.93859649122808,228.00751879699249,241.04010025062658,254.07268170426065,258.55263157894734,266.2907268170426,276.8796992481203,286.65413533834584,307.4248120300752,323.7155388471178,332.67543859649123,338.7844611528822,342.04260651629073,343.671679197995,343.671679197995,342.4498746867168,338.7844611528822,337.562656641604,337.562656641604,337.562656641604,337.562656641604,334.3045112781955,335.1190476190476,339.5989974937343,335.93358395989975,340.4135338345865,344.07894736842104,350.5952380952381,355.0751879699248,357.5187969924812,358.74060150375936,356.70426065162906,357.5187969924812,352.6315789473684,347.33709273182956,341.88596491228073,55.32581453634088,63.87844611528824,63.06390977443611,72.02380952380955,87.9072681704261,107.4561403508772,117.63784461152883,116.00877192982458,101.75438596491 23,65.5075187969925,40.25689223057647,46.365914786967444,56.95488721804514,86.68546365914789,61.84210526315792,46.365914786967444,111.93609022556393,102.97619047619047,89.12907268170429,104.19799498746869,83.0200501253133,102.9761904761905,100.53258145363411,100.93984962406017,102.56892230576443,97.68170426065164,99.31077694235591,98.90350877192985,94.01629072681706,88.72180451127822,75.28195488721806,61.434837092731854,53.69674185463662,48.80952380952384],"all_points_y":[72.08646616541353,67.19924812030074,64.34837092731829,64.34837092731829,68.42105263157895,68.82832080200501,66.79197994987467,63.12656641604009,62.71929824561403,59.05388471177944,53.352130325814 53,50.501253132832076,48.057644110275 68,43.98496240601503,38.69047619047618,37.875939849624054,35.0250626566416,35.0250626566416,36.65413533834586,37.875939849624054,37.061403508771924,29.73057644110275,27.69423558897243,19.95614035087719,8.959899749373433,1.2218045112781954,1.2218045112781954,3.665413533834586,6.109022556390976,11.810776942355888,20.363408521303256,32.17418546365914,53.75939849624059,68.42105263157895,88.7844611528822,112.40601503759397,132.36215538847117,145.39473684210523,158.42731829573933,164.94360902255636,178.79072681704258,189.78696741854634,197.52506265664158,208.11403508771926,221.55388471177943,243.13909774436087,264.3170426065163,285.49498746867164,300.9711779448621,311.1528822055137,324.99999999999994,324.99999999999994,307.0802005012531,298.5275689223057,288.3458646616541,267.1679197994987,247.2117794486215,226.4411027568922,217.88847117794484,215.44486215538845,220.7393483709273,238.65914786967414,262.2807017543859,252.91353383458645,241.9172932330827,222.36842105263156,231.32832080200498,239.47368421052627,209.74310776942355,204.0413533834586,205.26315789473682,199.96867167919797,199.15413533834584,194.67418546365911,189.37969924812203,178.38345864661653,169.01629072681703,161.68546365914784,151.50375939849621,138.8784461152882,125.43859649122805,111.99874686716791,101.40977443609022,88.7844611528822,79.82456140350877,72.08646616541353]],"region_attributes":{"label":"cat"}}}}}

2.2. 可视化建模

2.2.1. 方案设计

方案设计提供人工智能的流程可视化设计，方案提供了数据接入、数据处理、

机器学习和深度学习等组件，通过拖拽组件和连线的方式可以实现数据处理和人工智能建模、训练、评估等复杂流程。

方案设计页面

2.2.1.1. 构建方案设计流程

2.2.1.1.1. 新建

点击“新建方案”卡片，弹出“新建方案”窗口，如下图所示：

新建方案

再点击“新建空白方案”卡片后，页面跳转至“方案设计”页面，如下图所示：

新建方案页面

图中每个参数的具体含义如下：

方案名称：用户自定义，但是不能与现有方案名称一致。

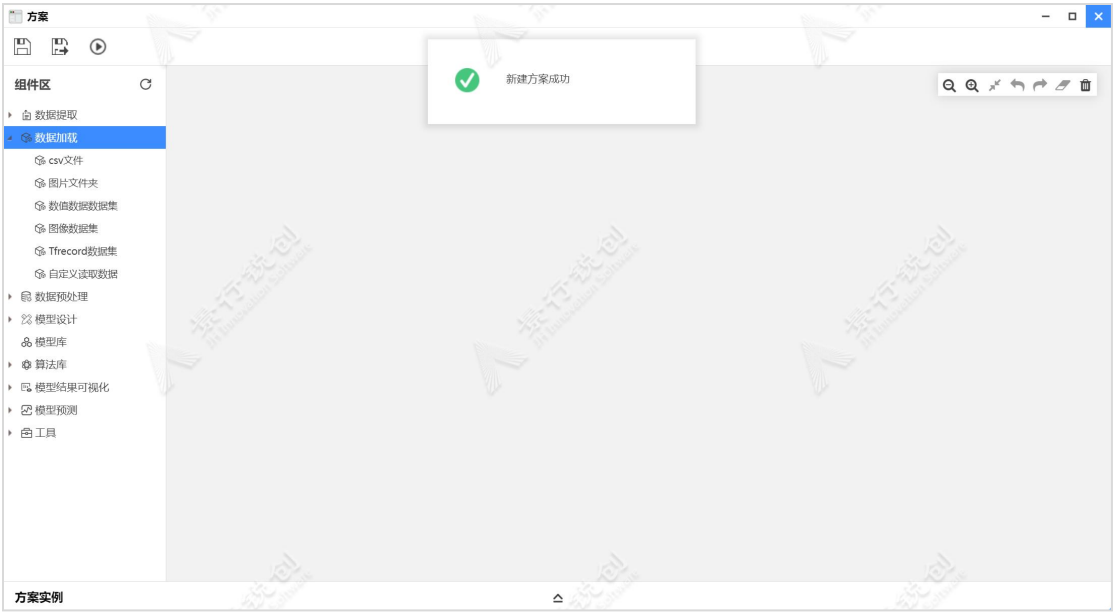
密级：管理员开启密级功能后，显示此选项，默认为用户密级，用于数据安全、保密。

方案类型：默认为已选的方案类型，也可以自定义方案类型。

方案描述：用于描述方案设计的相关内容，选填。

2.2.1.1.2. 设计

填写完成新建方案的表单后，点击“确定”按钮，进入方案设计页面。



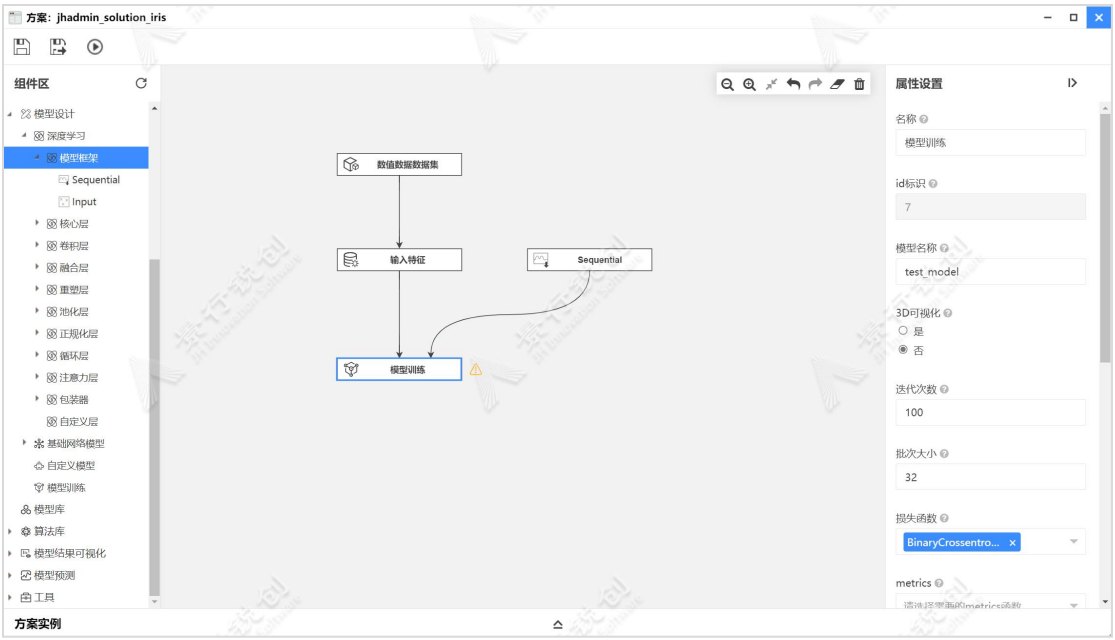
方案设计页面

“方案设计”左侧区域为组件区，组件区又根据组件属性分为“数据提取”，“数据加载”，“数据预处理”，“模型设计”，“模型库”，“算法库”，“模型结果可视化”，“模型预测”和“工具”。每一个组件都是一个模块的封装。

注：各组件对应的属性说明见附录一（数据加载处理组件属性说明）。

“方案设计”右侧区域为属性区，组件拖拽到画布区域后，属性设置面板就会从画布右侧滑出，用户可自行配置参数。当属性设置有误或未设置属性时，设计区的组件右侧会显示黄色叹号的图标，当鼠标移至此图标上时会显示出错误信息，必须纠正错误后才能够开始运行此方案。

画布右上角的工具栏可对画布区域和画布中的组件进行快捷操作，功能包含缩小、放大、实际比例、后退、前进、删除和清空。其中“删除”按钮仅会删除画布中选中的任意组件，“清空”则会删除整个画布中的内容，长按鼠标右键可拖动整个画布。



画布工具及组件属性配置

自定义函数管理

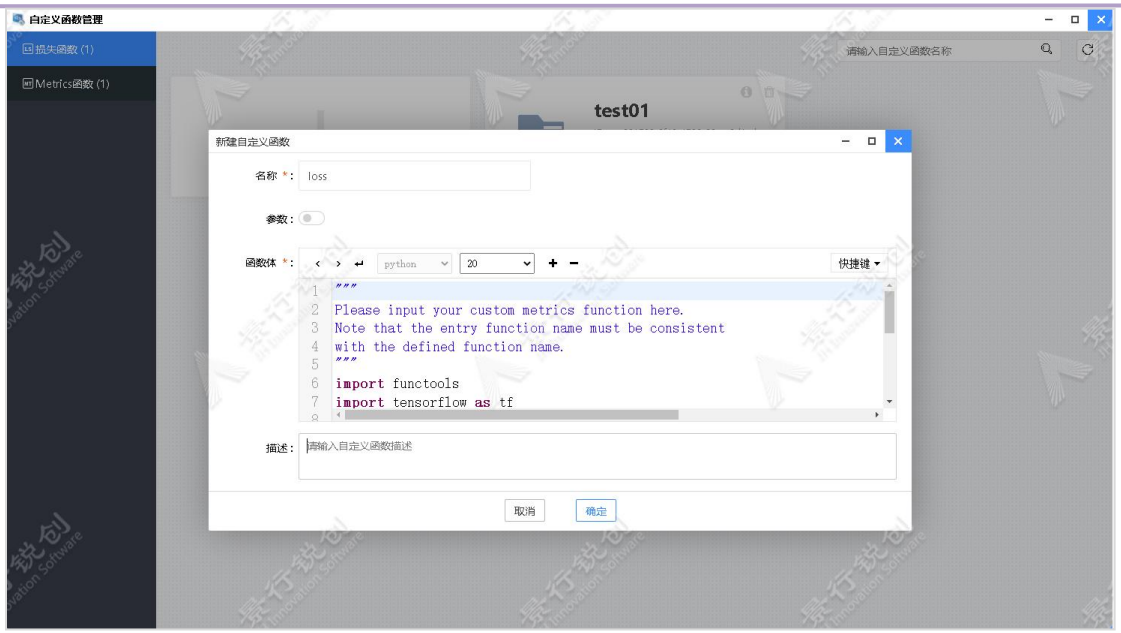
在自定义函数管理中可以创建和管理“损失函数”和“Metrics 函数”，选择相应的分类并点击新建卡片即可新建自定义函数。

创建成功的自定义函数会以卡片的形式显示在右侧区域中，卡片中的信息包括函数名称、ID 和创建时间，当鼠标停留在卡片右上角的“描述”按钮处时，会显示此函数的描述信息，点击卡片右上角的删除按钮即可删除当前自定义函数，删除操作不可恢复。

注意：在此创建的自定义函数可在方案设计中被引用。

- 损失函数

进入“自定义函数管理”页面，选择左侧“损失函数”，点击“新建自定义函数”卡片，弹出新建窗口，如下图所示：



自定义损失函数

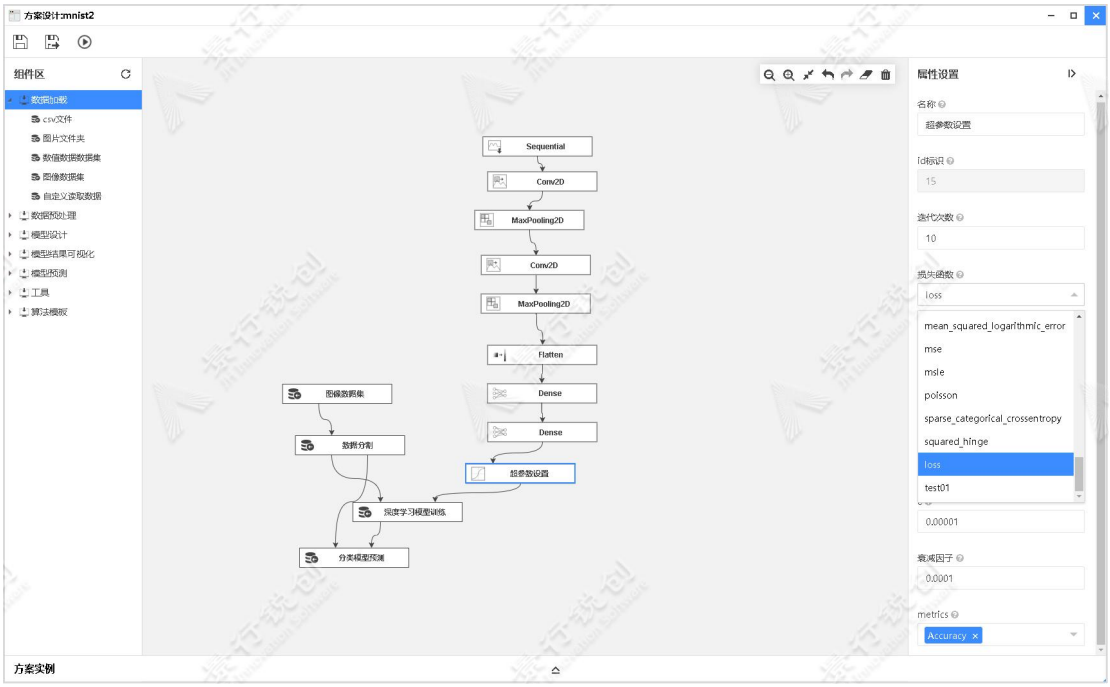
输入名称，并根据函数是否包含参数选择是否打开“参数”开关，在函数体部分输入自定义函数的内容，在描述部分输入对函数的描述或者备注信息，点击“确定”按钮，即可完成自定义损失函数的创建。

注意：函数“名称”的命名需要和函数入口名称一致。

损失函数内容如下：

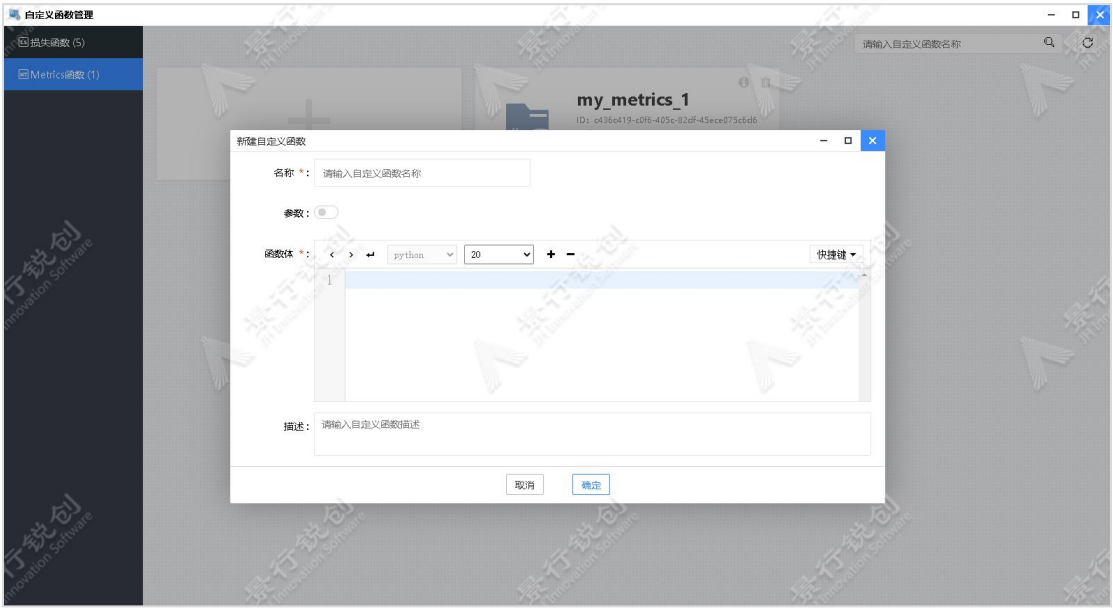
```
"""
Please input your custom metrics function here.
Note that the entry function name must be consistent
with the defined function name.
"""
import functools
import tensorflow as tf
from tensorflow.python.keras.losses import LossFunctionWrapper

def scaled_mean_square_error(y_true, y_pred, mean=0., std=1.):
    y_pred = (tf.convert_to_tensor(y_pred) - mean) / std
    y_true = (tf.cast(y_true, y_pred.dtype) - mean) / std
    return tf.keras.metrics.mean_squared_error(y_true, y_pred)

class my_loss_1(LossFunctionWrapper):
    def __init__(self, mean=0., std=1.,
name='scaled_mean_square_error'):
        super(my_loss_1,
self).__init__(functools.partial(scaled_mean_square_error,
mean=mean, std=std), name=name)
```

损失函数创建完成后在方案设计中训练深度学习方案结构时可以引用此函数。



- Metrics 函数

选择左侧“Metrics 函数”，点击“新建自定义函数”卡片，弹出新建窗口，如下图所示：



自定义 Metrics 函数

输入名称，根据函数是否包含参数选择是否打开“参数”开关，在函数体部分输入自定义函数的内容，在描述部分输入对函数的描述或者备注信息，点击“确

定”按钮，即可完成自定义 metrics 函数的创建。

注意：函数“名称”的命名需要和函数入口名称一致。

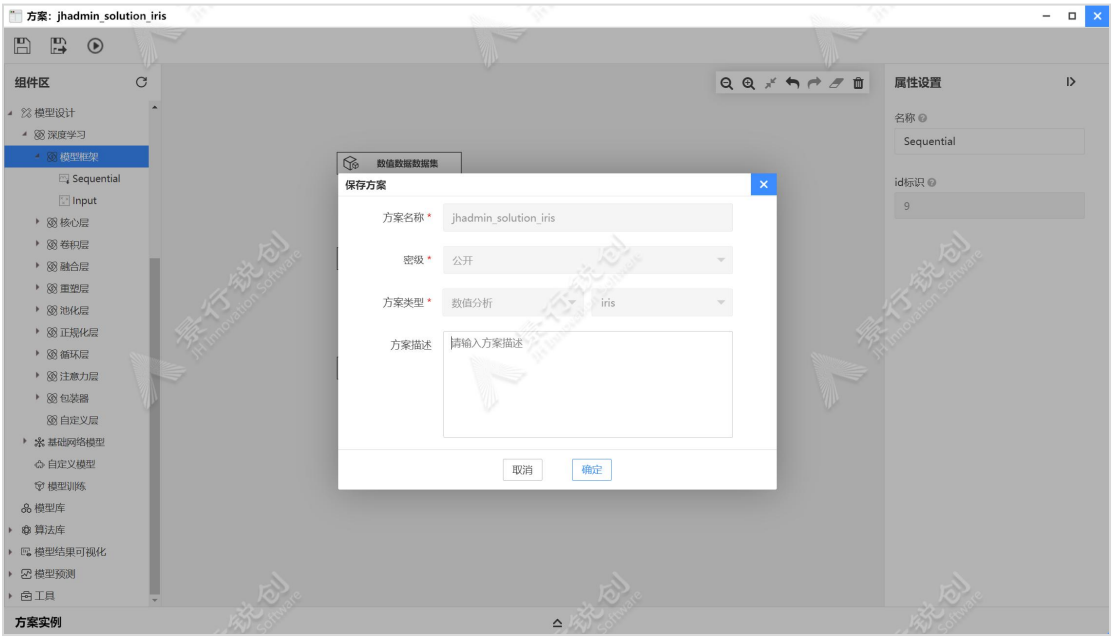
metrics 函数内容如下：

```
"""
Please input your custom metrics function here.
Note that the entry function name must be consistent
with the defined function name.
"""
import functools
import tensorflow as tf
from tensorflow.python.keras.metrics import MeanMetricWrapper

def mean_scaled_relative_error(y_true, y_pred, epsilon=1.):
    y_pred = tf.convert_to_tensor(y_pred)
    y_true = tf.cast(y_true, y_pred.dtype)
    return tf.keras.backend.mean(tf.abs(y_pred - y_true) /
(tf.abs(y_true) + epsilon), axis=-1)

class my_metrics_1(MeanMetricWrapper):
    def __init__(self, epsilon=0.1,
name='mean_scaled_relative_error', dtype=None):
        super(my_metrics_1,
self).__init__(functools.partial(mean_scaled_relative_error,
epsilon=epsilon), name, dtype=dtype)
```

2.2.1.2. 操作

2.2.1.2.1. 方案保存

点击工具栏中的“保存”按钮，即可将方案描述和画布上的方案设计保存至数据库，当再次打开该方案时，画布上会显示最近一次保存的方案。“保存方案”页面，如下图所示：



保存方案页面

图中每个参数的具体含义如下：

模型名称：任何状态下均不可修改。

密级：任何状态下均不可修改。

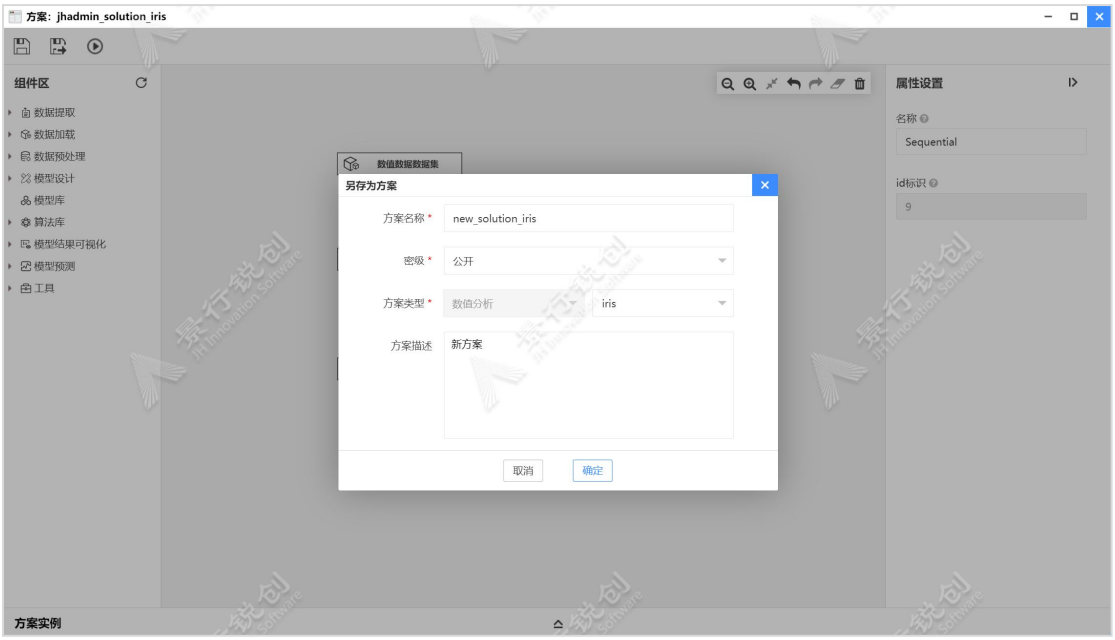
模型类型：任何状态下均不可修改。

模型描述：用于描述方案的相关内容，选填。

2.2.1.2.2. 方案另存为

点击工具栏中的“另存为”按钮，即可将方案另存为一个新的方案，方案可

以是已保存的和未保存的。“另存为方案”页面。如下图所示：



另存为方案页面

图中每个参数的具体含义如下：

方案名称：用户自定义，但是不能与现有的模型名称一致，必填。

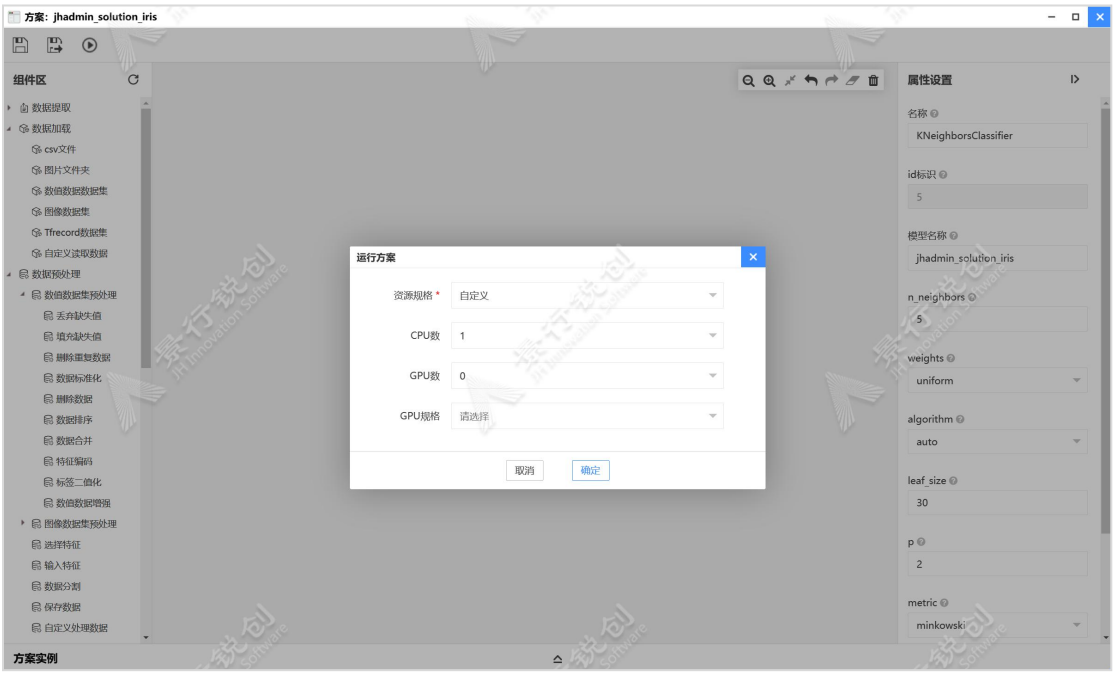
密级：对于另存的方案可根据需要修改密级。

方案类型：默认为当前方案的方案类型，可以选择，也可以自定义方案类型，必填。

方案描述：用于描述方案的相关内容，选填。

2.2.1.2.3. 方案运行

方案设计完成后，点击“运行”按钮，弹出“运行方案”窗口，选择训练方案需要使用的资源，资源选择后，点击“确定”按钮，即可开始训练。如下图所示：



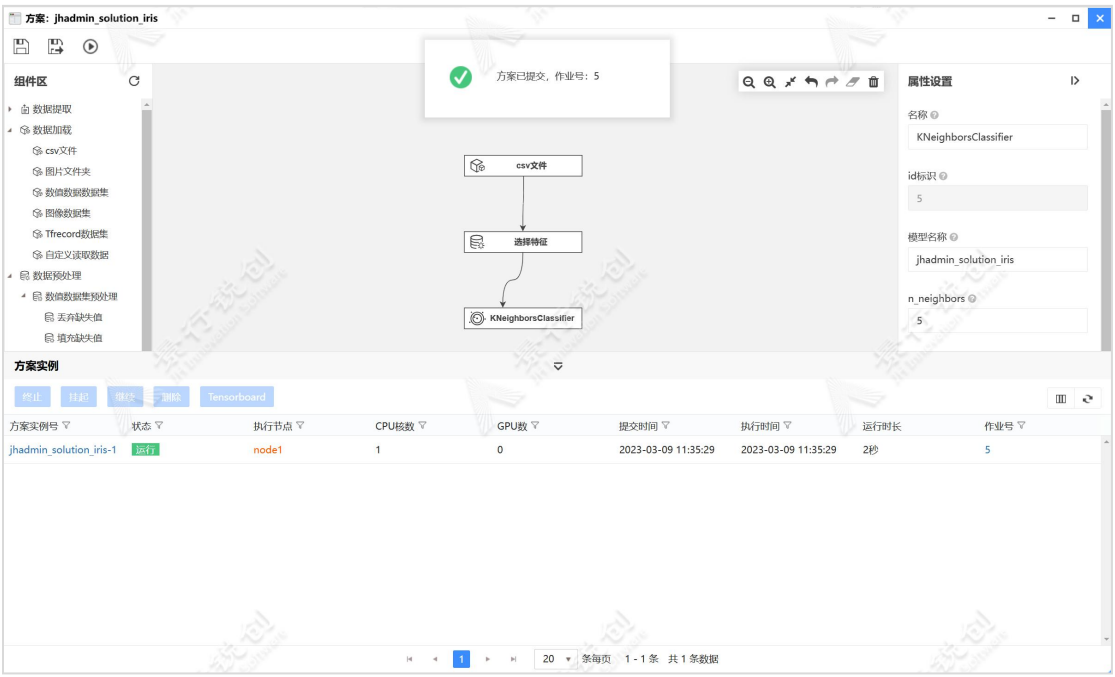
资源配置页面

点击运行按钮，打开选择资源页面，同时进行方案保存。选择资源后，点击确定按钮，即可开始训练方案。

注意：在方案设计页面中，点击“运行”按钮时，错误和问题对应如下：

- 画布中不存在组件时，提示“组件不能为空”。
- 组件的锚点没有连线、必填项属性为空时，提示“请检查异常组件并清除异常”。

开始训练方案后，会自动打开底部方案实例滑块，如下图所示：



底部方案实例滑块除了只能显示该方案的所有实例外，功能操作与方案实例一致，具体操作详情请看“方案实例”章节。

2.2.1.3. 管理

方案新建成功后，在方案设计应用中会自动生成了一个方案卡片，卡片内容包含方案名称、方案 ID、最后更新时间，点击方案卡片的右上角功能图标，支持对方案进行“发布”，“查看描述”和“删除”操作。

2.2.1.3.1. 查看描述

鼠标移至方案卡片上的“描述”按钮时，可以查看方案的描述信息，如下图所示：

点击“描述”的图标按钮，可在弹出窗口中修改方案的描述信息，如下图所示：

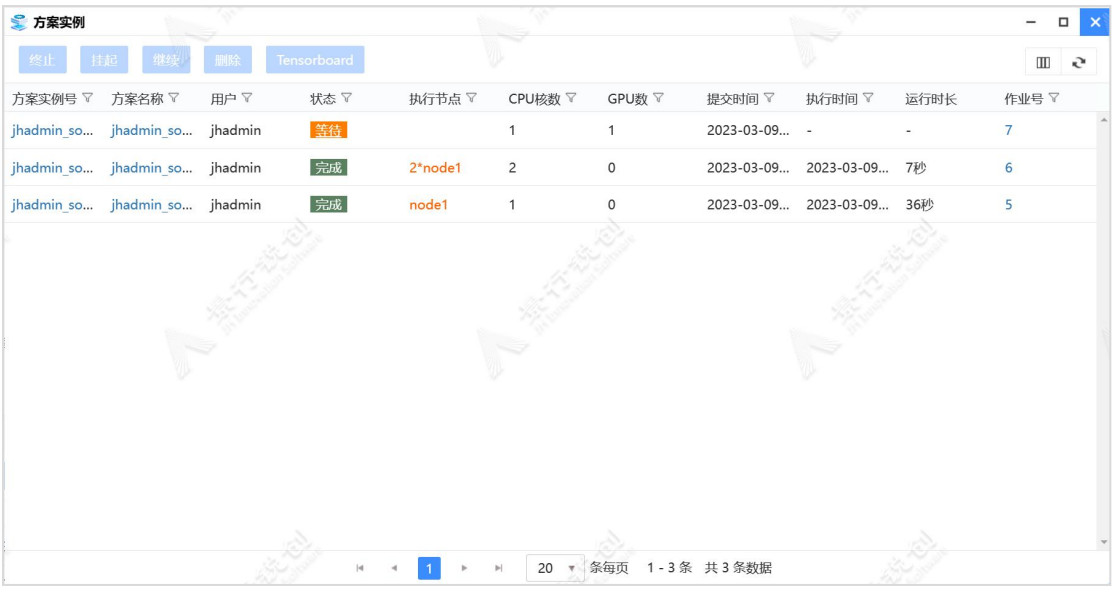
2.2.1.3.2. 删除

点击“删除”按钮，可以把该方案卡片删除。

2.2.2. 方案实例

方案实例中以列表的形式记录了所有方案的历史运行记录，用户可以通过实例号、名称等筛选功能快速过滤实例。

普通用户可在方案实例应用中查看自己的所有实例，或在具体的方案设计页面中，通过点击画布下方的方案实例滑块，在展开的面板中查看当前方案实例的信息，亦可对其进行终止、挂起等操作。管理员可在方案实例应用中查看所有用户的方案实例并对其进行操作。

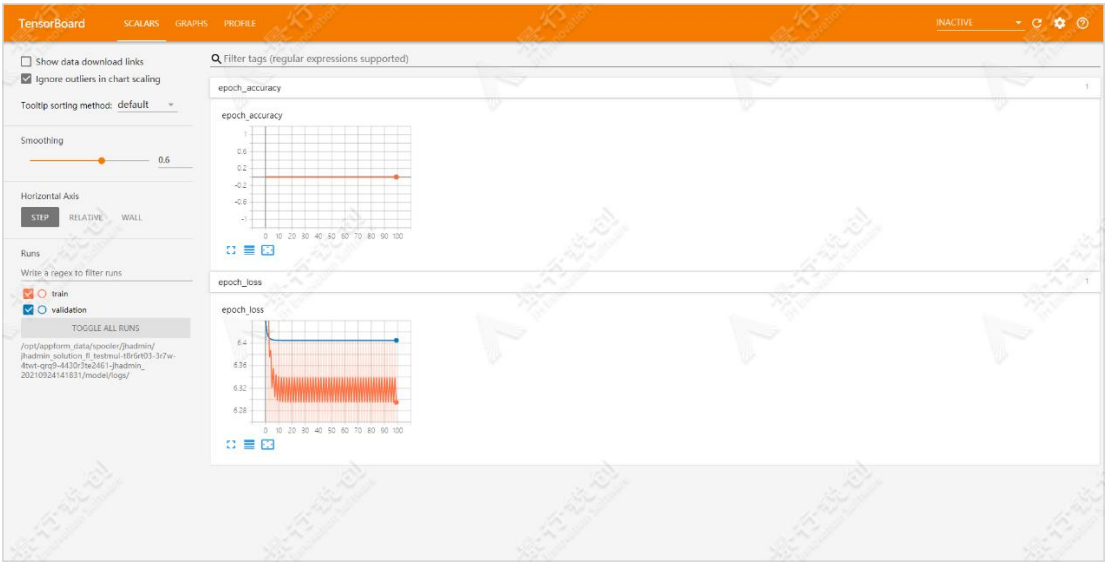


- 功能按钮：

方案实例支持对方案实例进行“终止”、“挂起”、“继续”、“删除”以

及打开 Tensorboard 操作。

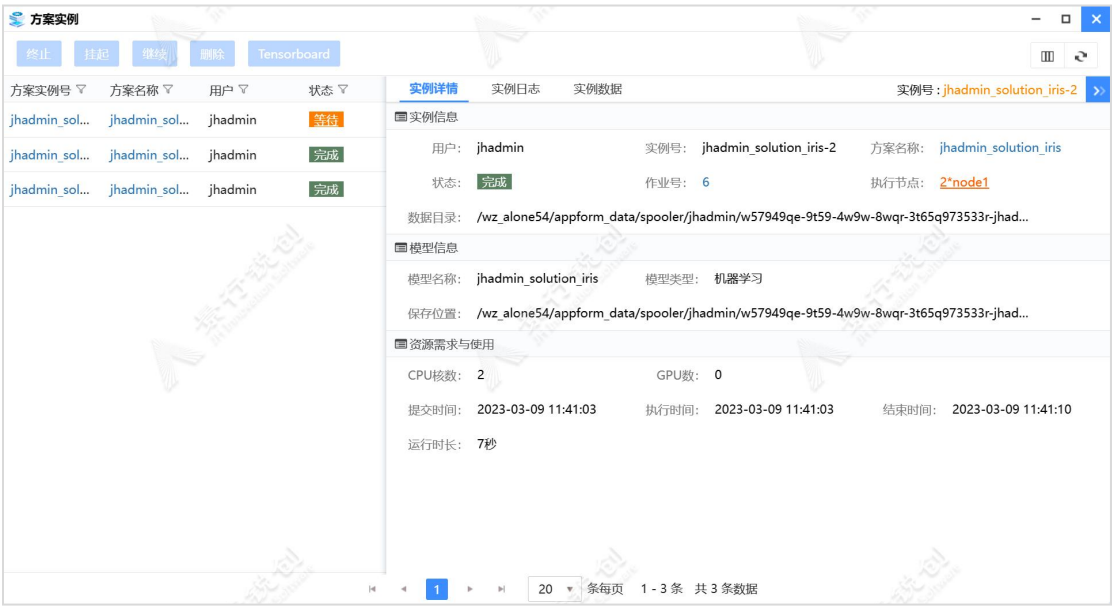
当深度学习的方案运行完成后，点击“Tensorboard”按钮可以以图表形式查看当前实例运行结果。



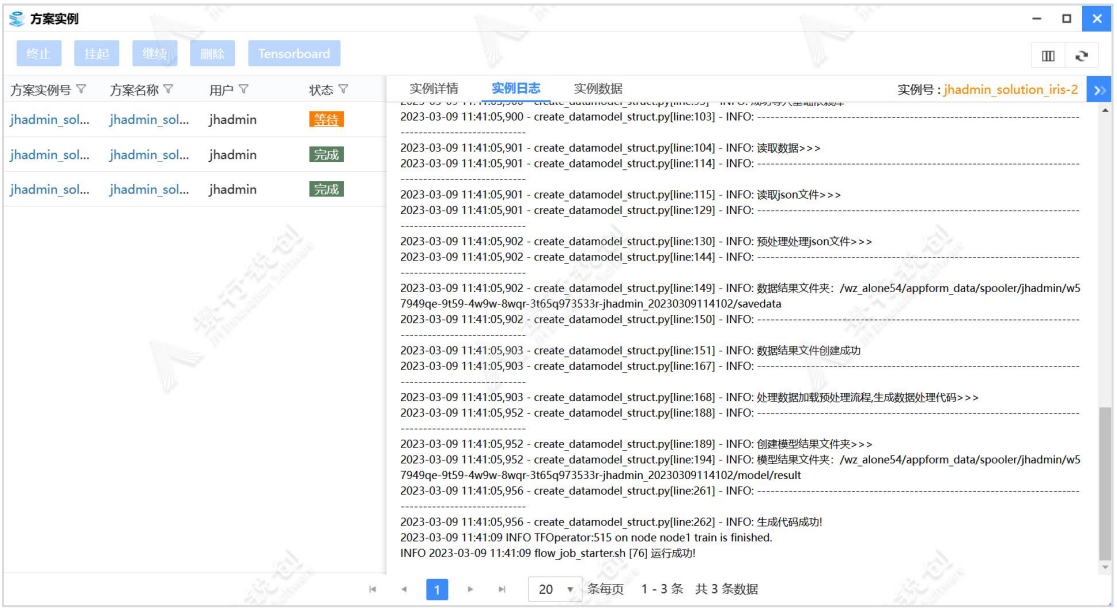
- 方案实例滑块：

点击“方案实例号”或者双击方案实例列表中的某条记录，会打开右侧滑块。滑块有三个标签页，分别为实例详情、实例日志和实例数据。

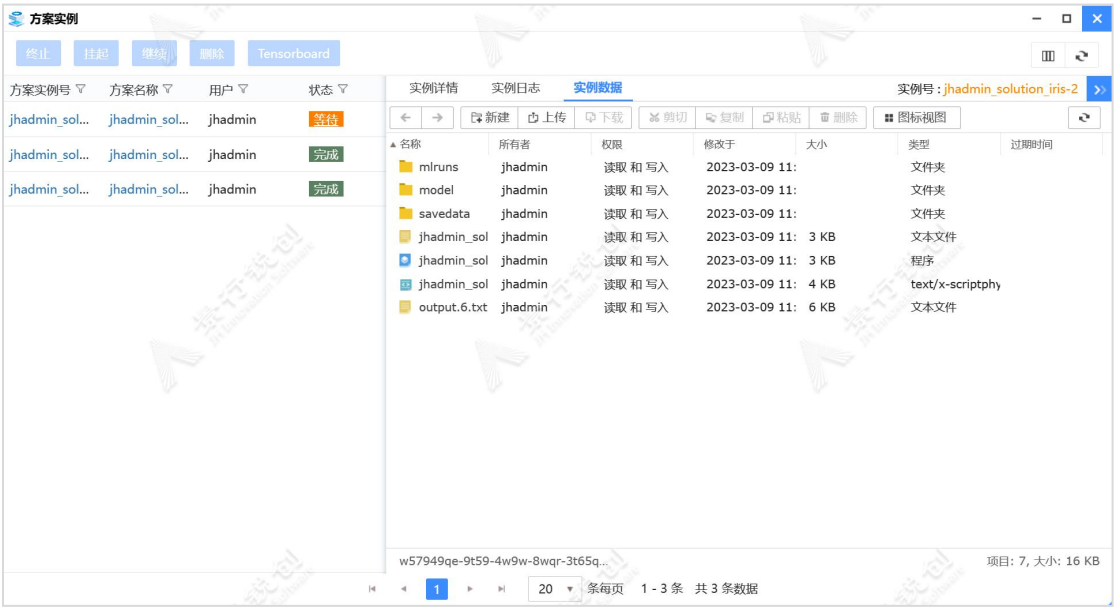
实例详情：显示方案实例的基本信息、资源信息。点击作业号，可以查看该实例关联的作业信息。实例详情中包含方案实例关联的数据集信息，包括数据集名称、数据集类型、字符集、分隔符以及数据保存目录。实例详情中包含方案实例关联的模型信息，实例详情中会显示模型信息，包括模型名称、模型类型、超参数(仅深度学习模型)。



实例日志：点击实例日志按钮，打开实例日志界面。方案实例的状态处于正在运行时，该实例日志会实时输出。方案实例的状态处于完成时，会显示该方案实例的所有日志。

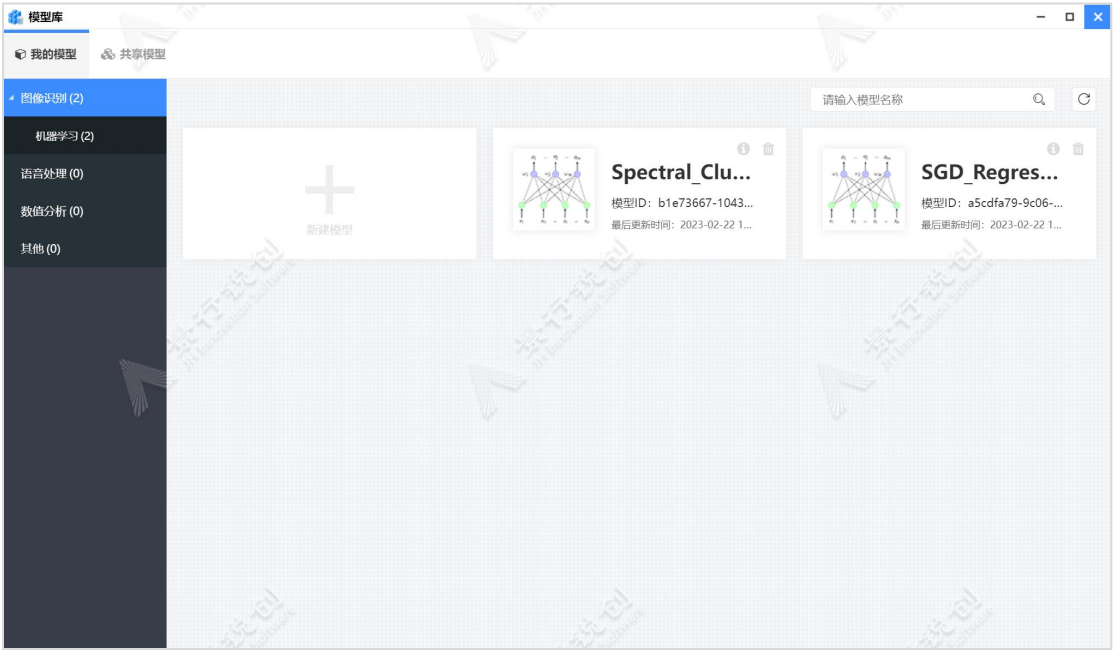


实例数据：点击实例数据按钮，打开实例数据界面，显示该方案实例包含的所有相关文件。



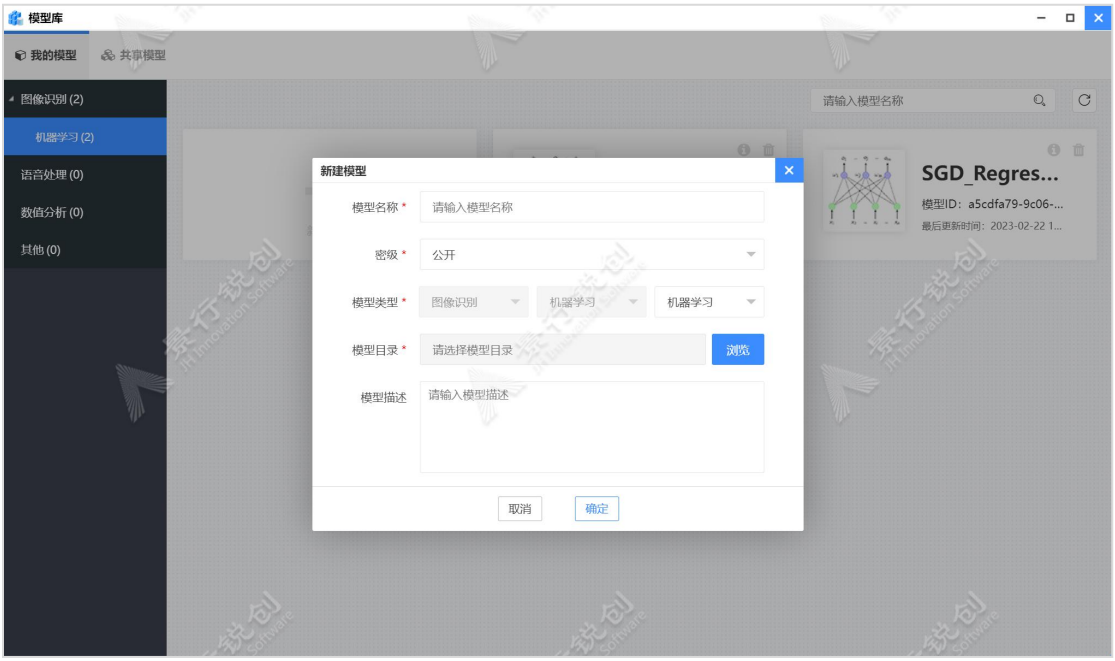
2.2.3. 模型库

模型库方便用户管理所有程序训练产生的模型，通过新建模型卡片来新建自定义模型。点击桌面上的“模型库”桌面图标，进入模型库界面，如下图所示：



2.2.3.1. 新建模型库

用户可以通过“新建模型”卡片，导入代码训练产生的 AI 模型。点击“新建模型”卡片，弹出“新建模型”窗口，如下图所示：



新建模型示意图

模型名称：用户自定义，但是不能与现有模型名称一致，必填项。

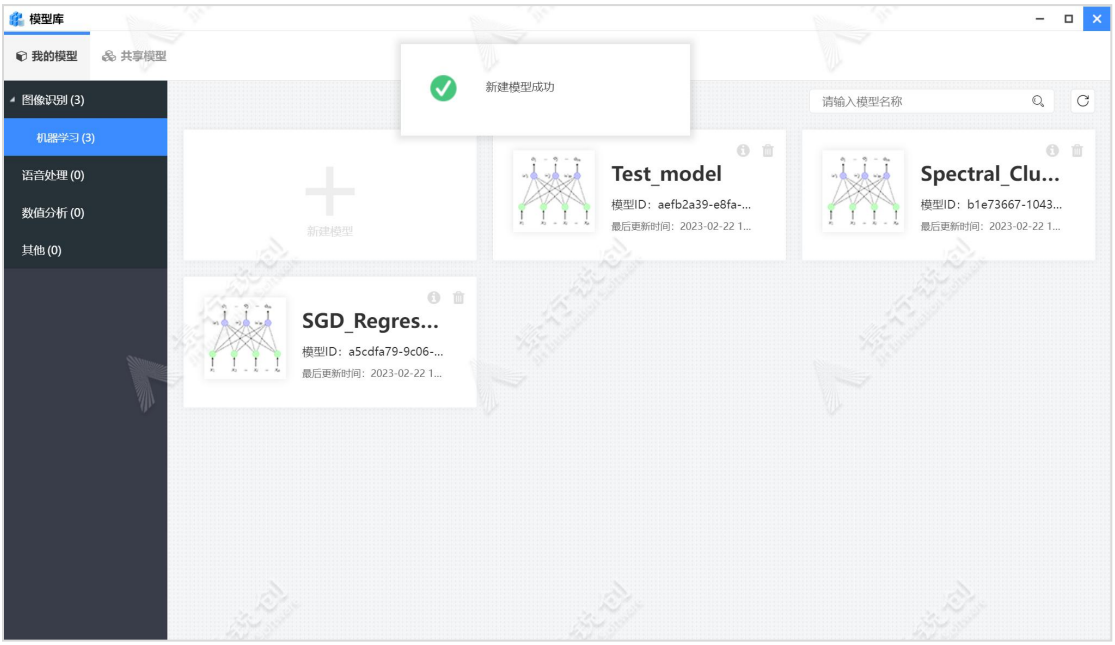
密级：管理员开启密级功能后，显示此选项，默认为用户密级，用于数据安全、保密。

模型类型：默认为已选的模型类型，也可以自定义模型类型，必填项。

模型目录：选择模型文件所在的目录，必填项。

模型描述：用于描述模型结果的相关内容，选填项。

点击“确定”按钮后，创建模型卡片。模型卡片创建成功后，会提示“新建模型成功”，如下图所示：



新建模型示例图

2.2.3.2. 模型训练历史列表

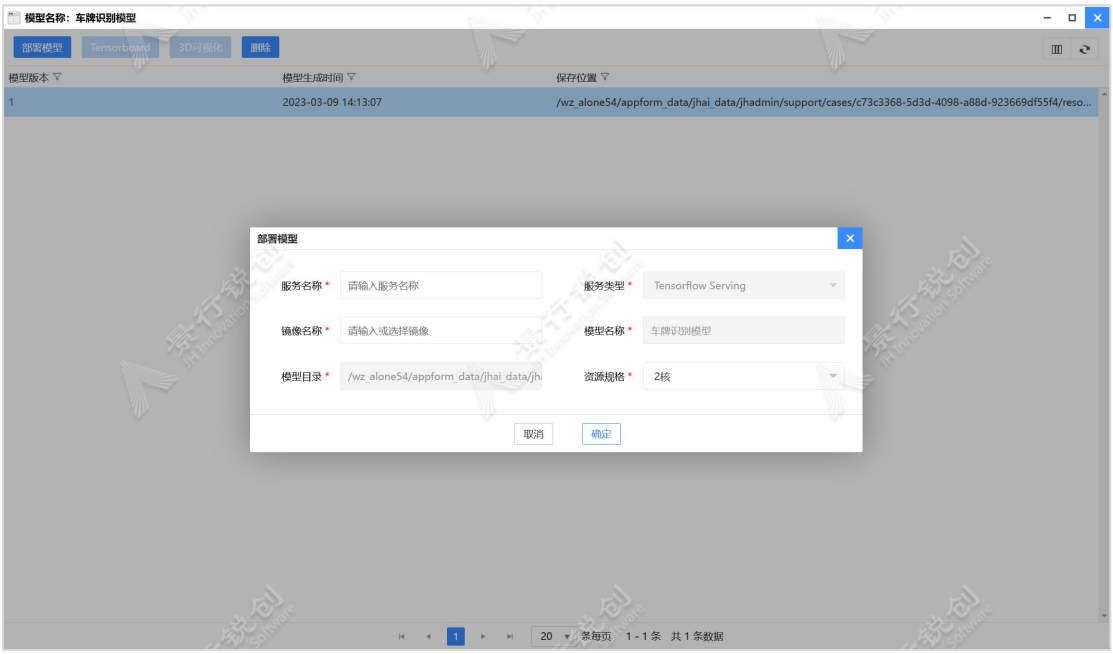
点击模型卡片，打开模型训练历史列表，方案运行实例产生的模型结果都存储于模型训练历史列表中，点击模型版本号或双击模型所在行则会展开滑块，滑块中包含“模型详情”和“模型结果”两个页签，在对应的板块中可以查看模型相关的具体信息，如下图所示：

模型训练历史列表中可以进行部署模型、Tensorboard、3D 可视化、删除和

共享操作。

1. 模型部署：

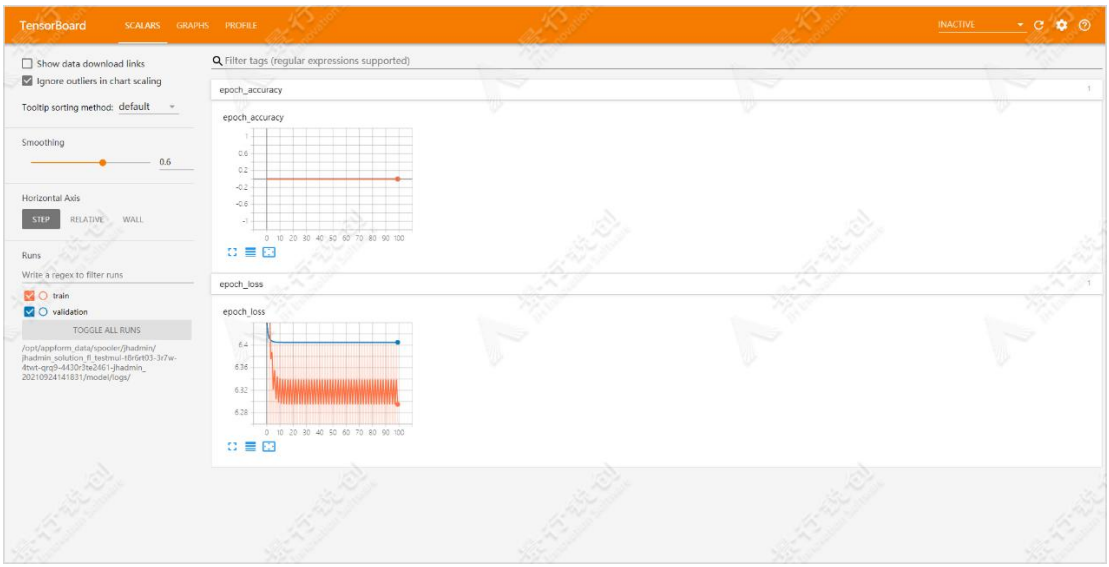
模型部署将训练好的 AI 模型部署为 HTTP 服务，可以供外部程序调用实现推理功能，目前模型部署仅支持深度学习模型。点击模型部署按钮，会打开部署模型窗口，如下图所示：



填写服务名称、选择资源。点击确定按钮，提交模型服务的作业。

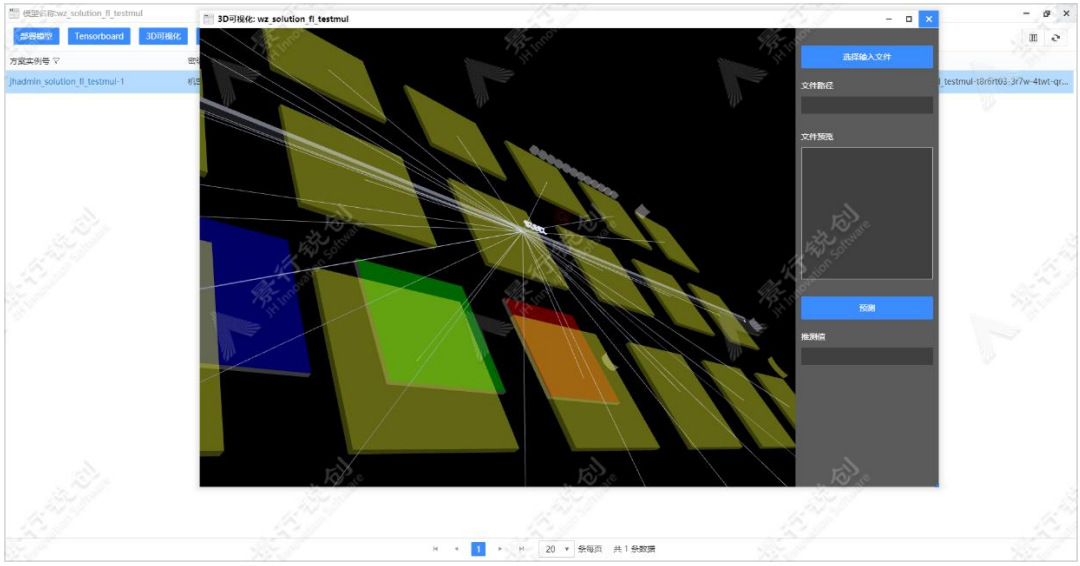
2. Tensorboard

打开 Tensorboard 查看模型的指标和结构，目前仅支持深度学习模型。选中训练历史中某条记录，点击 Tensorboard 按钮，即可打开 Tensorboard 服务页面。



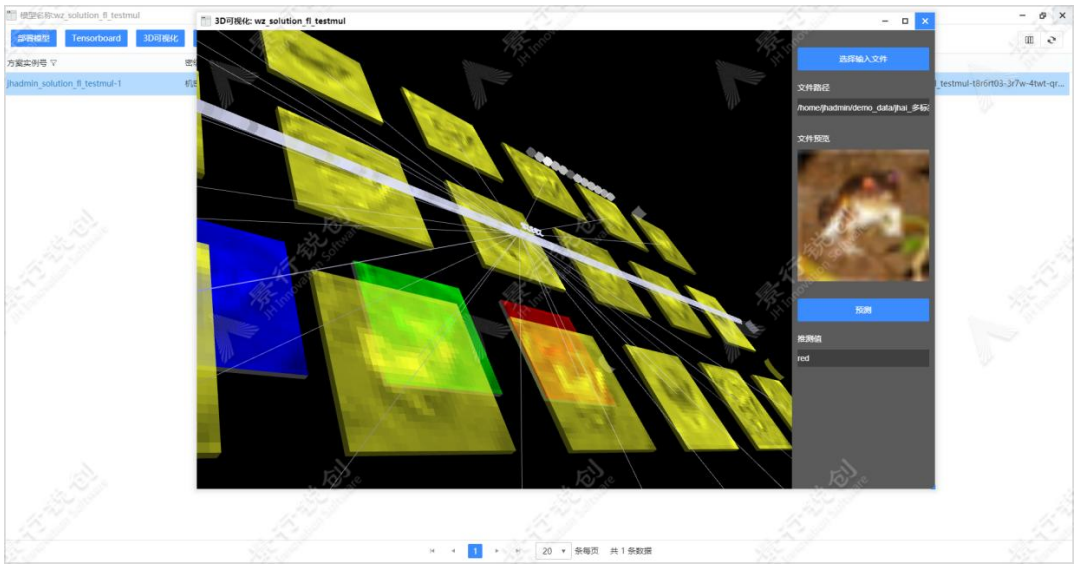
3. 3D 可视化

3D 可视化功能支持在 Web 浏览器中以 3D 的形式显示模型结构，目前仅支持部分深度学习模型。点击 3D 可视化按钮，弹出该模型结果的 3D 可视化页面，点击窗口的彩色块，可查看模型结构和配置参数等。



深度学习模型-3D 可视化

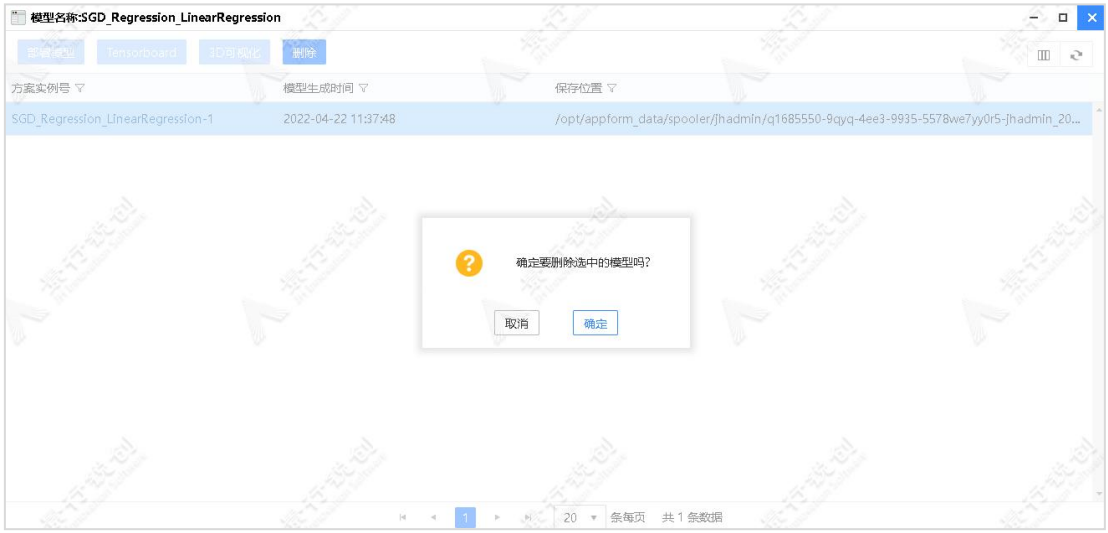
页面右侧，点击“选择输入文件”选择需要预测的数据，接着点击“预测”，可实现模型预测，在“推理值”位置显示预测的具体值。可视化页面如下：



3D 可视化模型预测

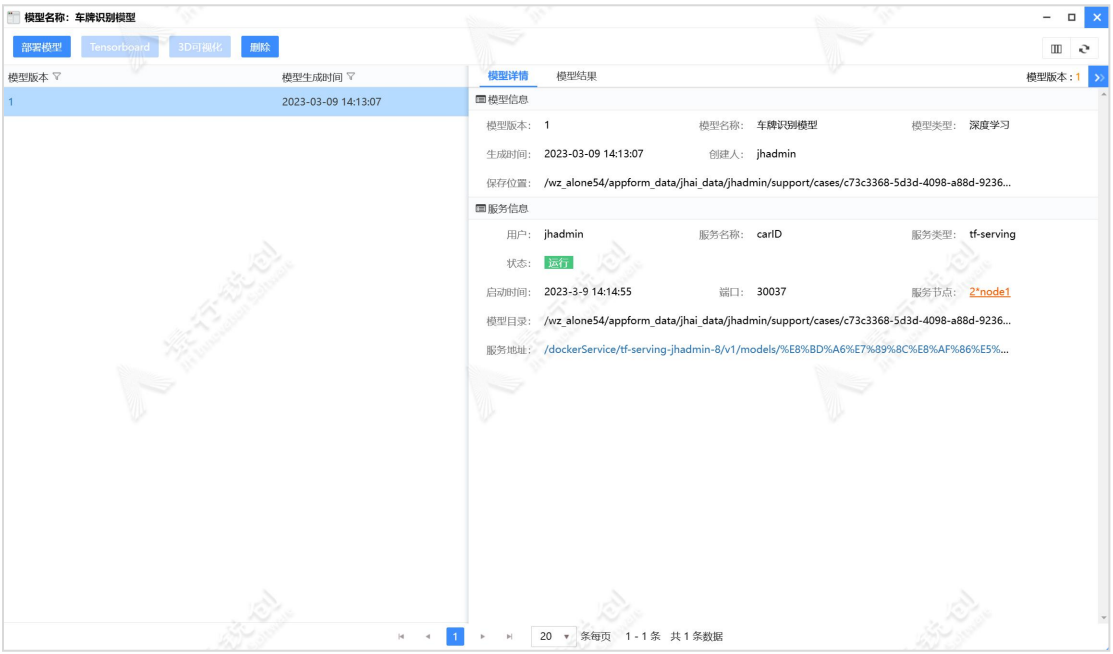
4. **删除：**

点击删除按钮，可以删除该模型训练历史记录。

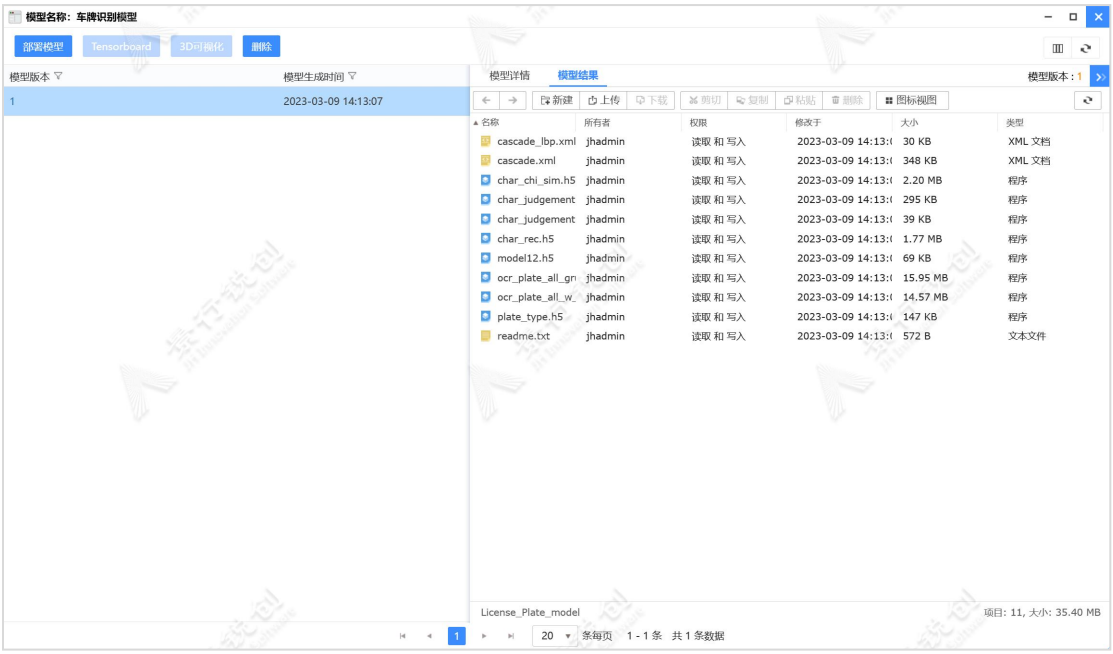


点击训练历史列表中某条记录的方案实例号或者双击某条记录，会打开右侧滑块。滑块中有两个标签页，分别为“模型详情”和“模型结果”。

- **模型详情：**显示模型的基本信息和关联的服务信息。如下图所示：



➢ **模型结果：**点击模型结果按钮，打开模型结果界面，显示模型的结果文件与日志文件，如下图所示：



2.2.3.3. 模型卡片

方案训练成功后，模型库自动生成模型卡片，模型卡片包含模型名称、模型ID、最后更新时间等信息，模型支持“修改描述”和“删除”操作。

2.2.3.4. 共享模型

共享模型用以给共享组成员或指定成员在方案设计和服务发布中使用。

我的模型共享至共享模型

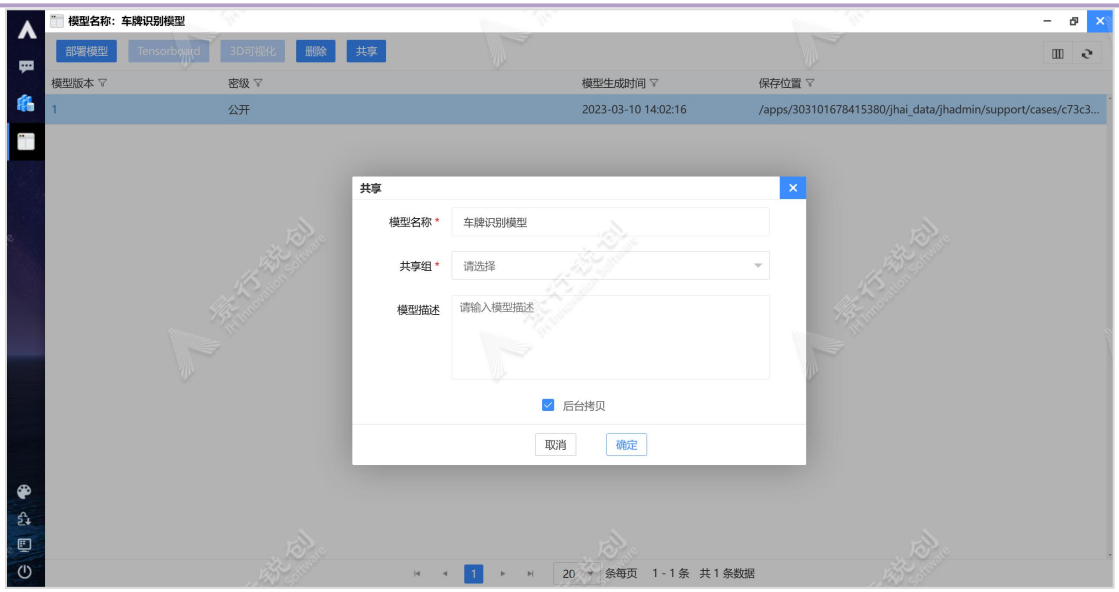
注意：添加共享模型之前需要先创建共享组。

以车牌识别模型为例，步骤如下：

依次点击“我的模型”-“图像识别”-“车牌识别”，然后在右侧的工作区，点击“车牌识别模型”卡片，打开模型版本列表，如下图所示：

模型版本

选中某一个模型版本，点击顶部“共享”按钮，弹出共享窗口，如下图所示：



共享模型

图中每个参数的具体含义如下：

模型名称：共享后的模型名称，输入任意符合命名规则的名称即可。

共享组：选择可见的共享组，选项从我的数据->共享数据区获取。

共享用户：选择共享组之后显示，显示为选择的共享组成员。

共享目录：选择共享组之后显示，只能在所选共享组的共享数据区选择文件夹，点击蓝色的“文件夹图标”按钮，然后在弹出的“选择文件夹”窗口中选择文件夹即可，用作存放源数据文件目录，只允许选择一个文件夹。

模型描述：共享后的数据集描述信息。

后台拷贝：勾选后采用后台形式拷贝源模型文件至共享目录，当前共享页面退出；不勾选则采用前台拷贝方式，当前共享页面不退出。

在共享窗口中填写完成后，点击“确定”按钮，可以保存此次共享的模型。

2.3. 案例中心

案例中心是将目前在景行人工智能平台上已经成功实现并且较为成熟的案例集中进行展示的地方。用户不仅可以查看现有案例，了解在平台中训练模型和推理的具体流程，还可以使用案例中心的案例，训练自己的模型，推理数据。

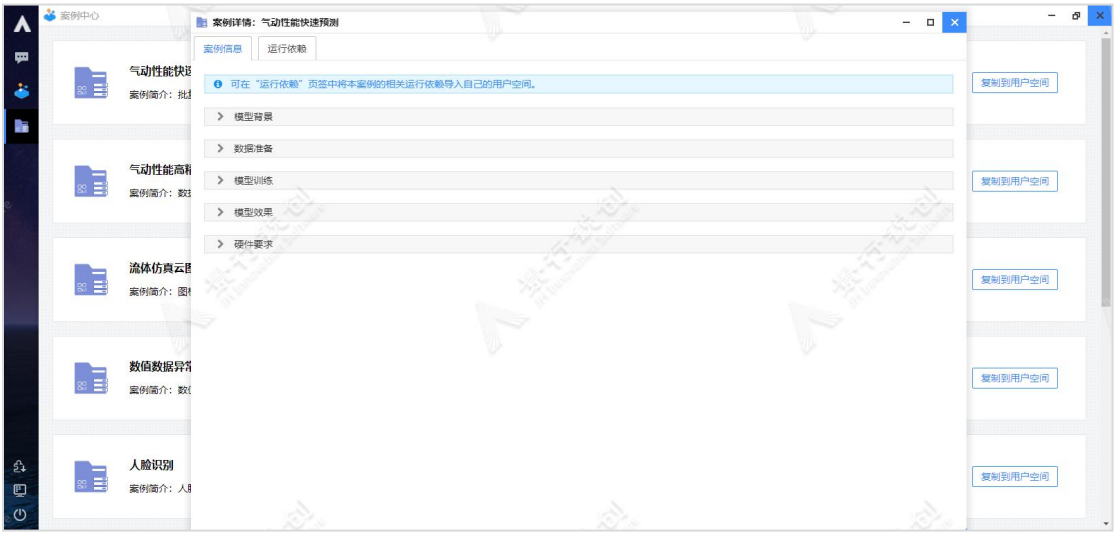
双击“案例中心”桌面图标，打开“案例中心”界面，如下图所示：

案例中心管理示意图

2.3.1. 案例详情

目前案例中心中集成了 9 个案例，包括：气动性能快速预测、气动性能高精度预测、流体仿真云图预测、数值数据异常检测、人脸识别、车牌识别、火灾检测、行人轮廓检测和人体关键点检测。

案例详情展示了案例的详细介绍和使用方法。以“气动性能快速预测”案例为例，操作步骤如下所示：点击“气动性能快速预测”案例卡片右边上的“案例详情”按钮，弹出“案例详情”窗口，如下图所示：



案例信息界面详细描述了案例的模型背景、数据准备、模型训练、模型效果

和硬件要求，用户可以按照该步骤训练自己的模型和推理数据。每一项标题都可以展开和收起，方便查看。

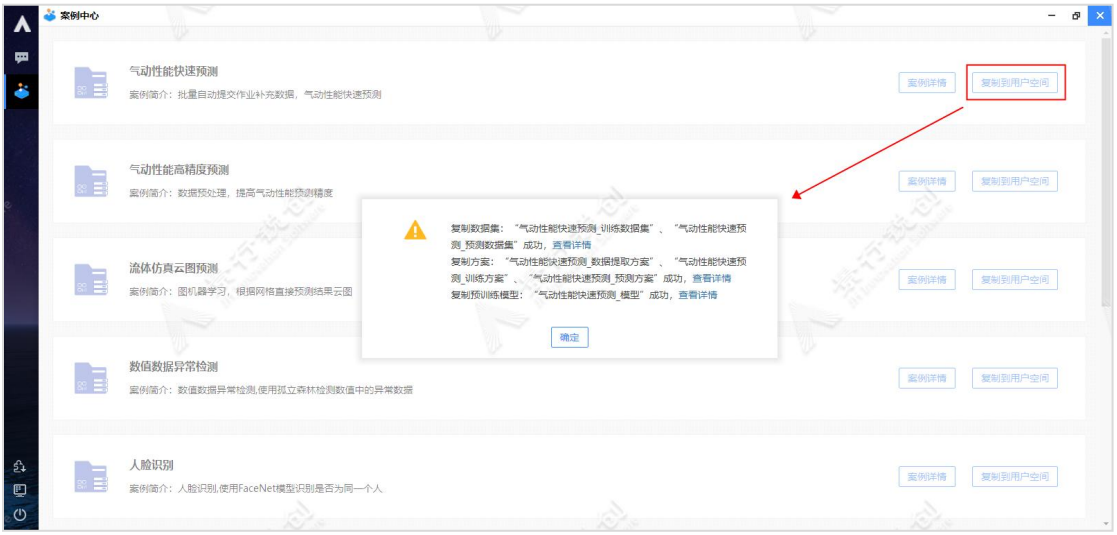
点击“运行依赖”按钮，切换至“运行依赖”标签页，用户可以查看案例所需要的数据、方案、模型等依赖。如下图所示：

运行依赖页面示意图

其中，“预览”按钮，可以查看数据集的数据或者方案的方案结构；“其他文件”框中提供了推理文件，支持下载。

2.3.2. 案例使用

点击“复制到用户空间”按钮，可以将案例依赖的所有数据文件复制到用户自己的环境中。以“气动性能快速预测”案例为例，操作步骤如下所示：点击“气动性能快速预测”案例卡片右边上的“复制到用户空间”按钮，将案例复制到自己的空间，如下图所示：



复制到用户空间示意图

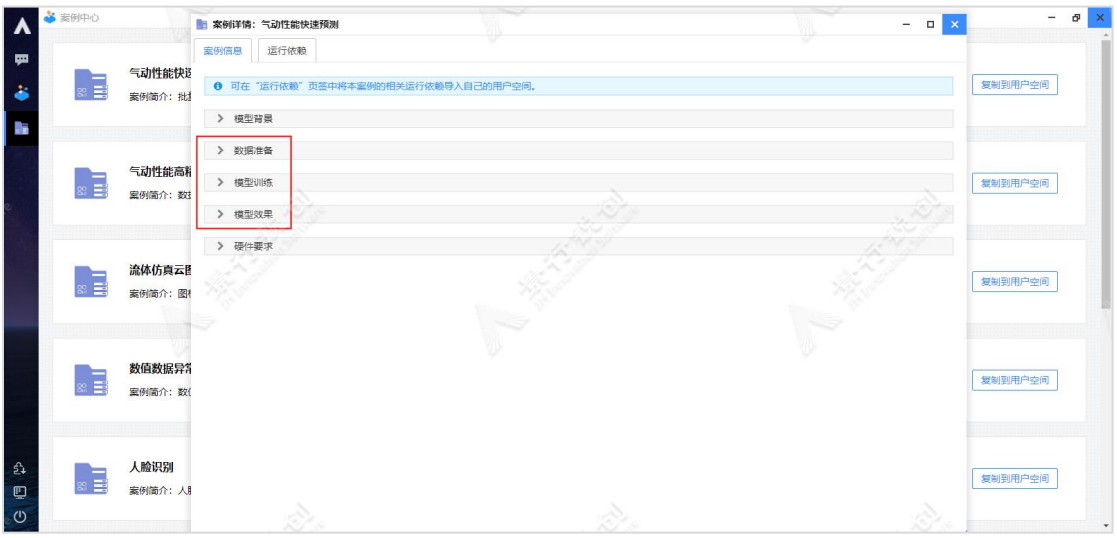
注意：案例的每次复制都会覆盖用户本地的同名数据集、方案和模型。因此用户可以将修改过的方案进行另存为，防止被覆盖导致的丢失。

可以点击提示中的“查看详情”跳转链接，查看复制过来的数据集、方案或者模型。以查看复制过来的数据集为例，操作步骤如下所示：点击复制数据集项的“查看详情”跳转链接，可以在“数据集管理”-“我的数据集”中，查看数据集信息，如下图所示：

查看数据集信息

根据“案例详情”中的步骤：“数据准备”-“模型训练”-“模型效果”，

依次准备数据、训练数据、部署服务、推理数据即可，推理文件在“运行依赖”中下载，如下图所示：



查看案例信息

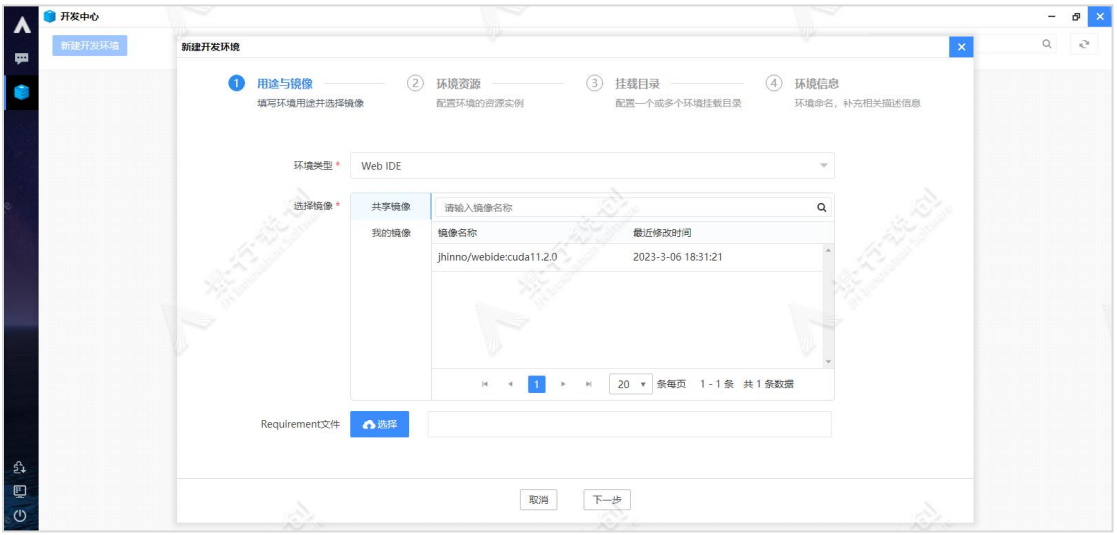
推理文件下载

2.4. 开发中心

开发中心支持人工智能模块在线安全编程环境，通过 web 方式调试和编写 AI 代码，集成了“Web IDE”、“CentOS 桌面”、“Ubuntu 桌面”和“SSH”四种环境类型。满足用户在开发中心编写和调试模型训练代码，然后基于该代码进行模型的训练。

2.4.1. 新建开发环境

以新建“Web IDE”类型的开发环境为例，操作步骤如下：点击“新建开发环境”按钮，弹出“新建开发环境”窗口，如下图所示：



新建开发环境

图中每个参数的具体含义如下：

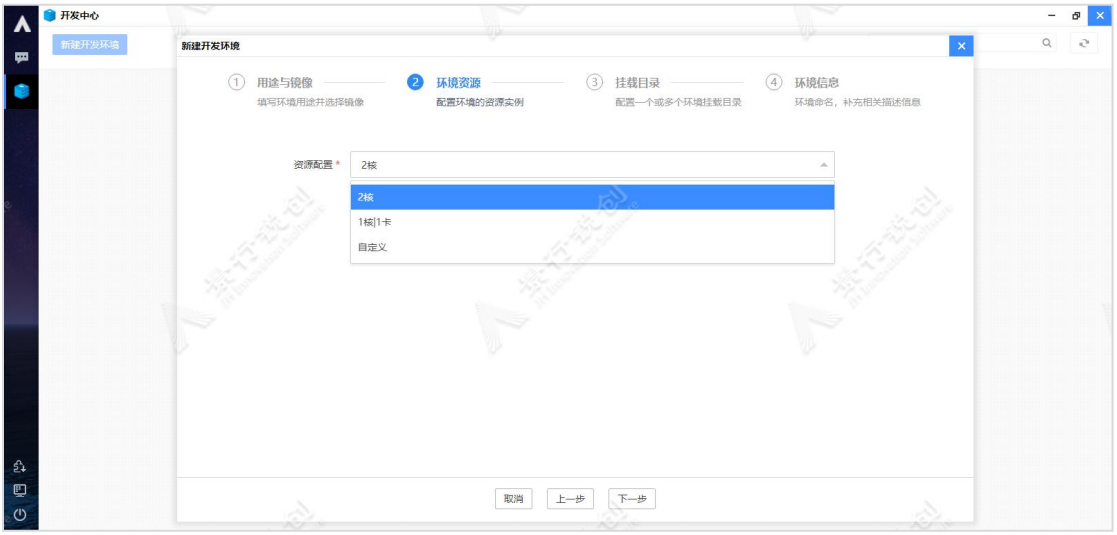
环境类型：选择新建环境的类型，支持四种类型的环境，如下所示：

- Web IDE：该环境集成了 Jupyter 和 VScode 开发工具。
- CentOS 桌面：提供带 GUI 的操作系统环境。
- Ubuntu 桌面：提供带 GUI 的操作系统环境。
- SSH：仅提供 WEB 终端。

选择镜像：选择创建环境实例所用的模板镜像。

Requirement 文件：根据选择的配置文件，快速配置环境实例，可选。

选择完环境类型和镜像后，点击“下一步”按钮进入环境资源配置页面：

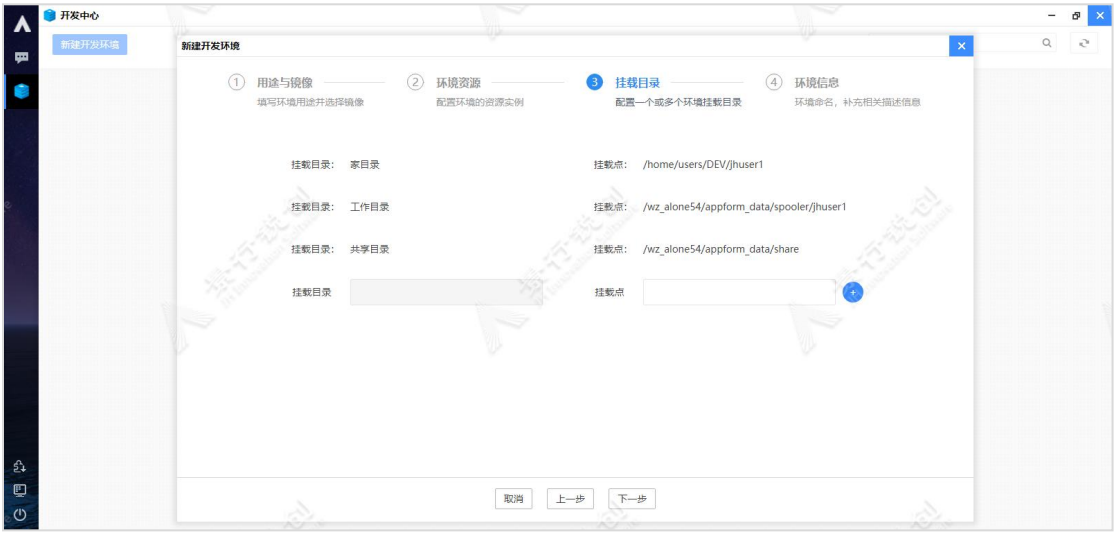


环境资源

图中每个参数的具体含义如下：

资源配置：默认为 2 核，选择新建环境实例所需要的资源配置。

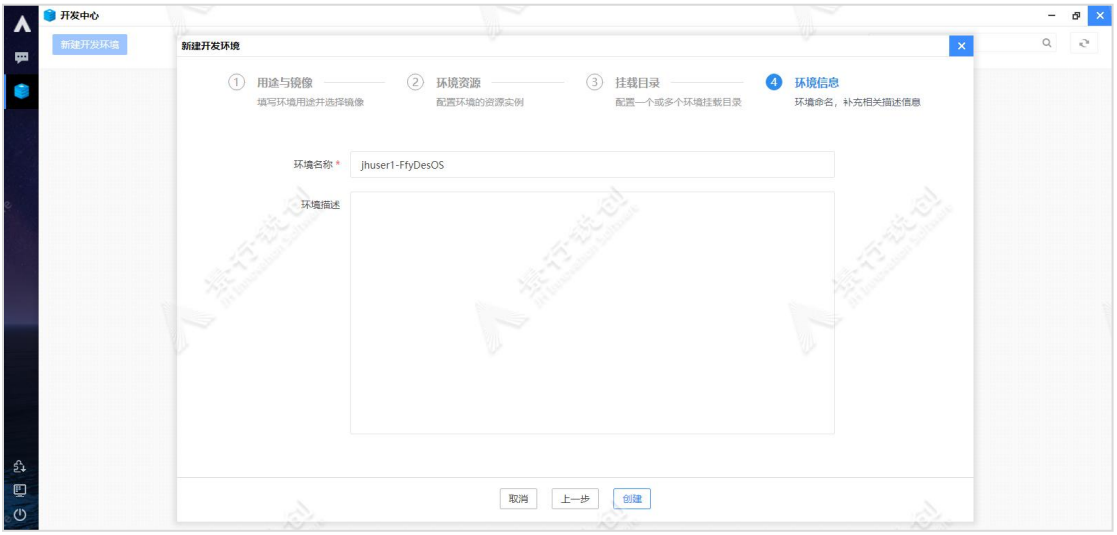
选择合适资源规格后，点击“下一步”按钮，进入挂载目录页面：



挂载目录

默认会把“家目录”、“工作目录”、“共享目录”挂载到容器中，也可以额外指定目录，挂载到开发环境容器的中。

完成挂载目录配置，点击“下一步”按钮配置环境名称和描述，如下图所示：



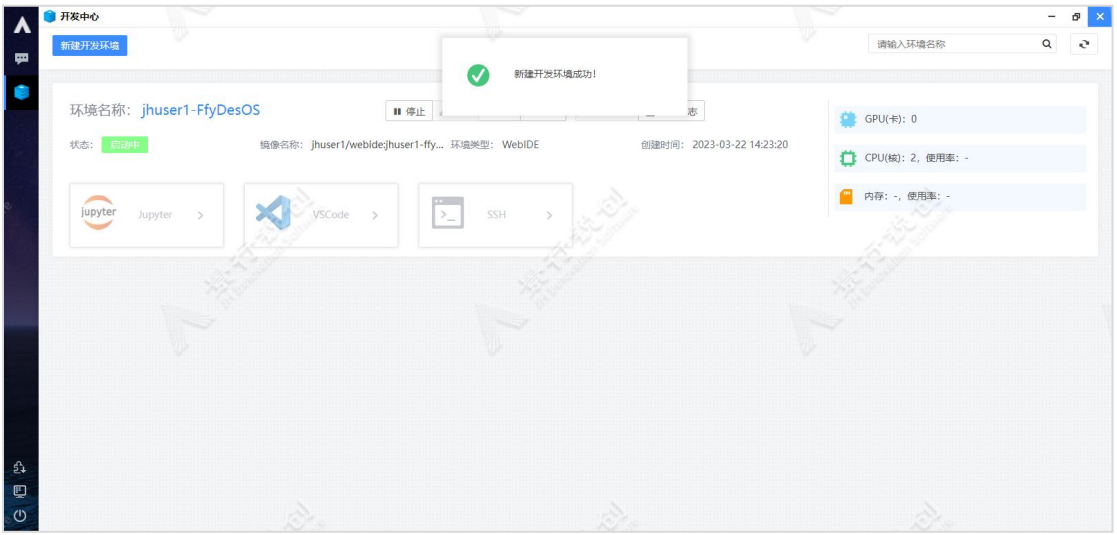
环境信息

图中每个参数的具体含义如下：

环境名称：默认系统会回填一个可用的环境名称。

环境描述：输入环境描述信息，可选。

点击“创建”按钮，即可完成开发环境的创建，如下图所示：



开发环境列表

2.4.2. 使用开发环境

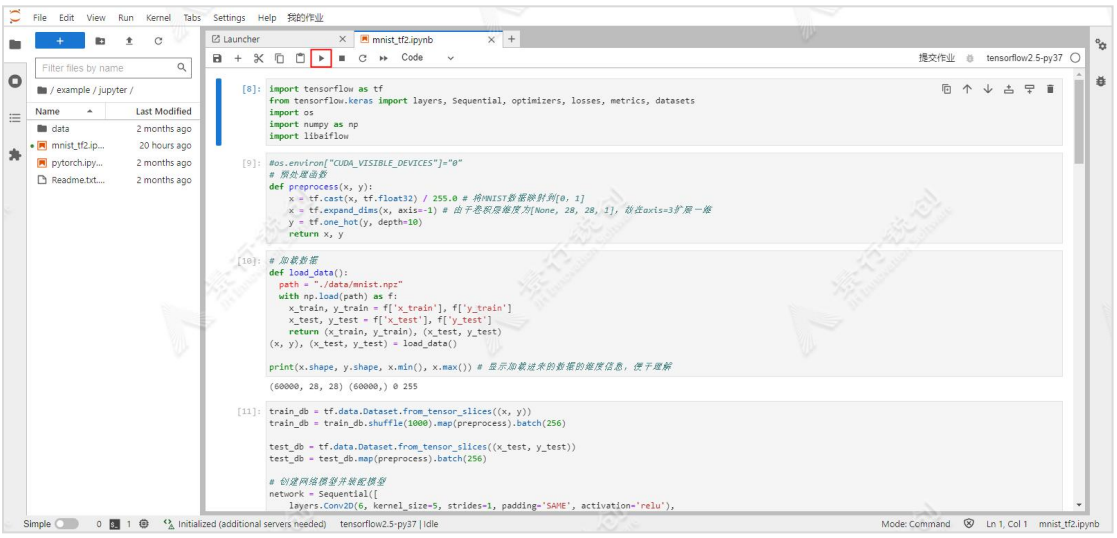
2.4.2.1. 应用 Jupyter 服务

点击“Web IDE”类型的开发环境实例上的“Jupyter”卡片按钮，访问 Jupyter 服务，运行程序的操作方式有 2 种：

- 直接运行。
- 以提交作业的方式运行程序。

（一）直接运行程序

点击右上角的“运行”按钮，运行程序，如下图所示：



（二）以提交作业的方式运行程序

选择“ipynb”格式的训练程序后，点击“提交作业”按钮，弹出“提交作业”窗口，如下图所示：

提交 AI 作业示意图

作业提交参数：

项目：指定项目，默认为：default。

作业名：指定作业名。

环境镜像：必选项，选择程序运行环境的镜像，需要根据运行程序选择合适的镜像。

工程目录：指定代码的工程目录，默认回填运行程序的父目录。

创建作业执行目录：默认不勾选，直接在工程目录中运行程序。手动勾选，会在“我的作业-作业数据区”中创建临时执行目录，将工程目录中的数据复制至其中后，进行运行程序操作。

运行命令：必填项，指定运行程序的入口文件和程序参数，例如：train.py --data-path=/data --batch=54。

Tensorboard：选择是否允许访问 tensorboard 服务。若启动，作业详情页面，点击“访问 Tensorboard”按钮，即可查看模型训练情况。

密级：管理员开启密级功能后，显示此选项，默认为用户密级，用于数据安全、保密。

运行方式：选择作业运行方式，如下所示：

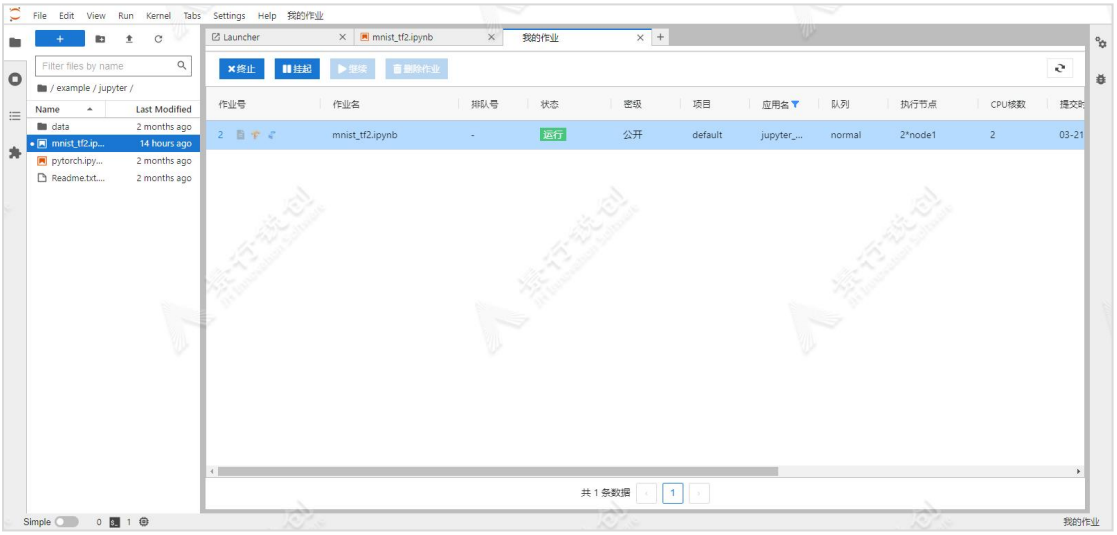
- **单机：**默认选择单机，仅单节点上运行训练程序。
- **PS 并行：**提交以 Parameter Server 方式实现的训练代码，支持跨节点，需填写并行相关参数，如下所示：

- **ParamServer 数：**指定参数服务的数量。
- **Worker 数：**指定训练服务数量。
- **实例规格：**分别指定每个 ParamServer 和 Worker 使用的资源。

- **Horovod 并行：**提交以 Horovod 实现的并行 AI 训练程序， 需要指定 Horovod 相关参数。Horovod 并行模式需要模型训练代码以 Horovod 库的并行方式进行编码，如下所示：
  - **并行实例数：**指定并行训练的节点数量。
  - **实例规格：**指定每个并行节点使用的资源。

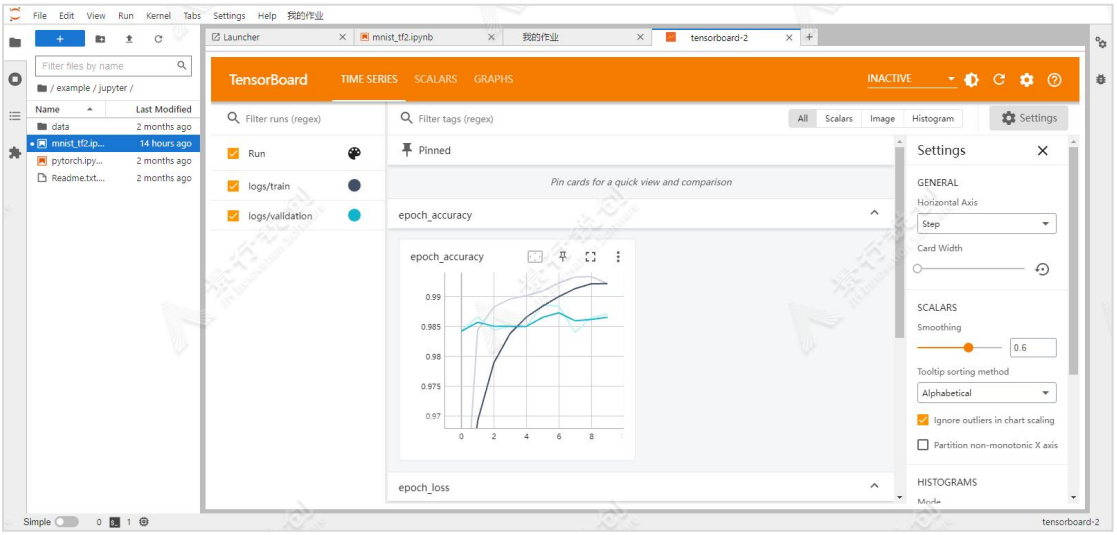
环境变量：指定程序运行需要的额外环境变量。

点击“我的作业”按钮，打开我的作业 tab 页，如下图所示：



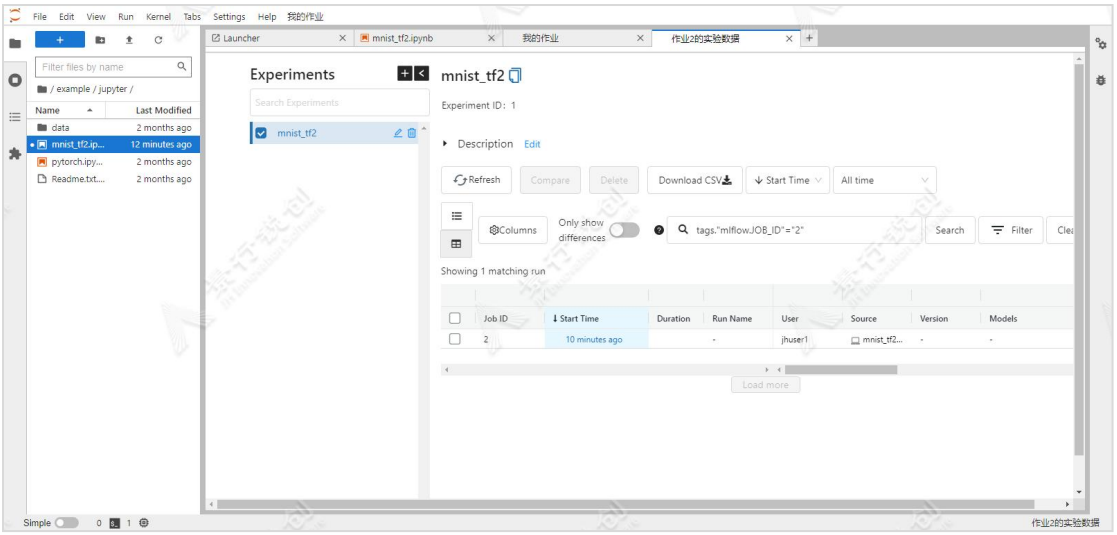
我的作业示意图

点击作业实例上的“Tensorboard”图标按钮，访问 Tensorboard 服务查看模型数据，如下图所示：



访问 Tensorboard 服务示意图

点击作业实例上的“实验管理”图标按钮，查看实验数据，如下图所示：



查看实验数据示意图

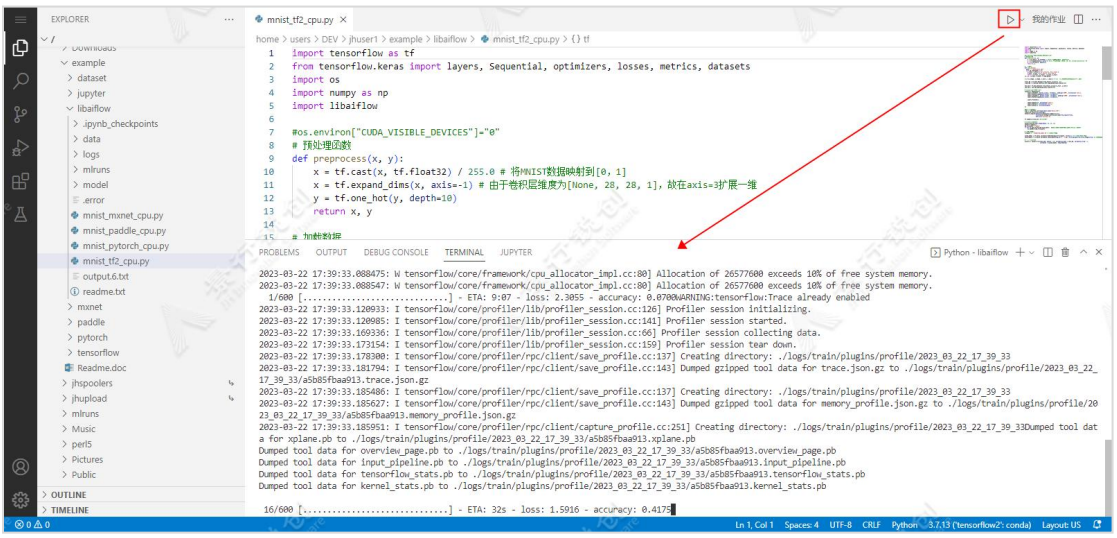
2.4.2.2. 应用 VSCode 服务

点击“Web IDE”类型的开发环境实例上的“VSCode”卡片按钮，访问 VSCode 服务，运行程序的操作方式有 2 种：

- 直接运行。
- 以提交作业的方式运行程序。

（一）直接运行程序

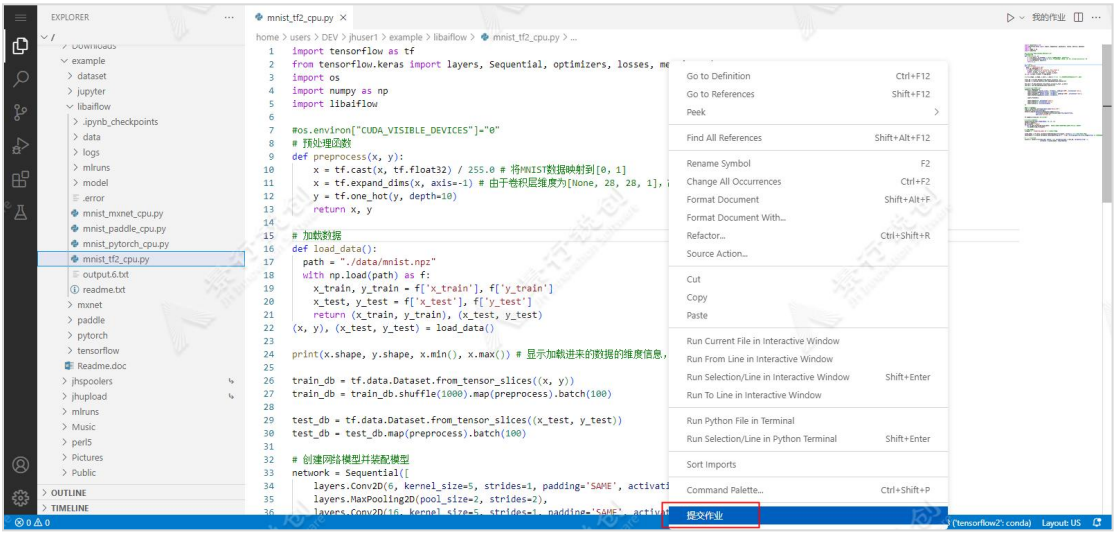
打开执行程序后，点击右上角的“运行”按钮，运行程序，如下图所示：



运行程序示意图

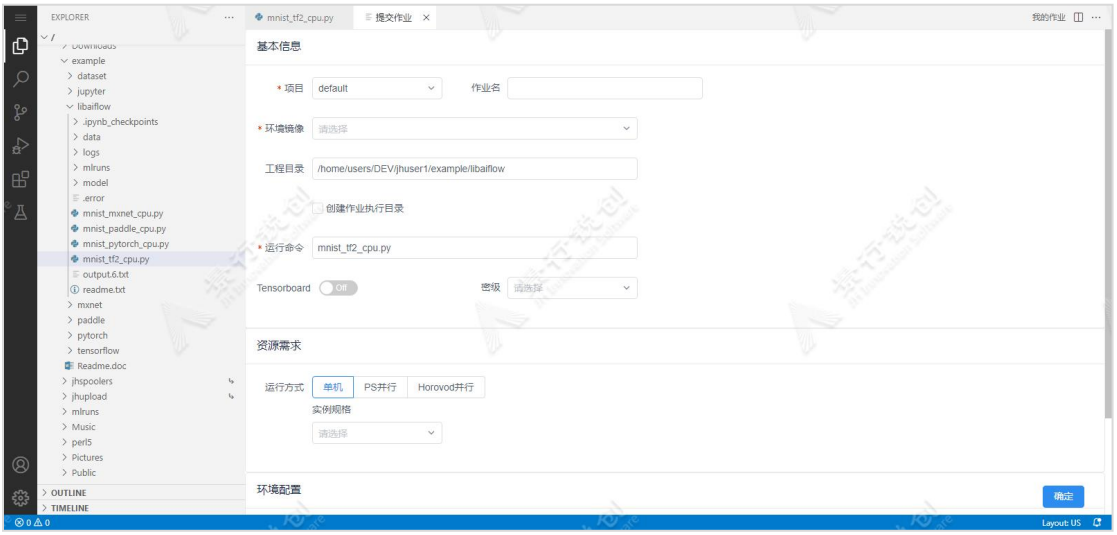
（二）以提交作业的方式运行程序

Explorer 中选中文件或者打开文件，右键打开右键菜单，如下图所示：



右键菜单示意图

点击“提交作业”选项，弹出“提交作业”弹窗，如下图所示：



作业提交页面示意图

作业提交参数：

项目：指定项目，默认为：default。

作业名：指定作业名。

环境镜像：必选项，选择程序运行环境的镜像，需要根据运行程序选择合适的镜像。

工程目录：指定代码的工程目录，默认回填运行程序的父目录。

创建作业执行目录：默认不勾选，直接在工程目录中运行程序。手动勾选，会在“我的作业-作业数据区”中创建临时执行目录，将工程目录中的数据复制至其中后，进行运行程序操作。

运行命令：必填项，指定运行程序的入口文件和程序参数，例如：train.py --data-path=/data --batch=54。

Tensorboard：选择是否允许访问 tensorboard 服务。若启动，作业详情页面，点击“访问 Tensorboard”按钮，即可查看模型训练情况。

密级：管理员开启密级功能后，显示此选项，默认为用户密级，用于数据安全、保密。

运行方式：选择作业运行方式，如下所示：

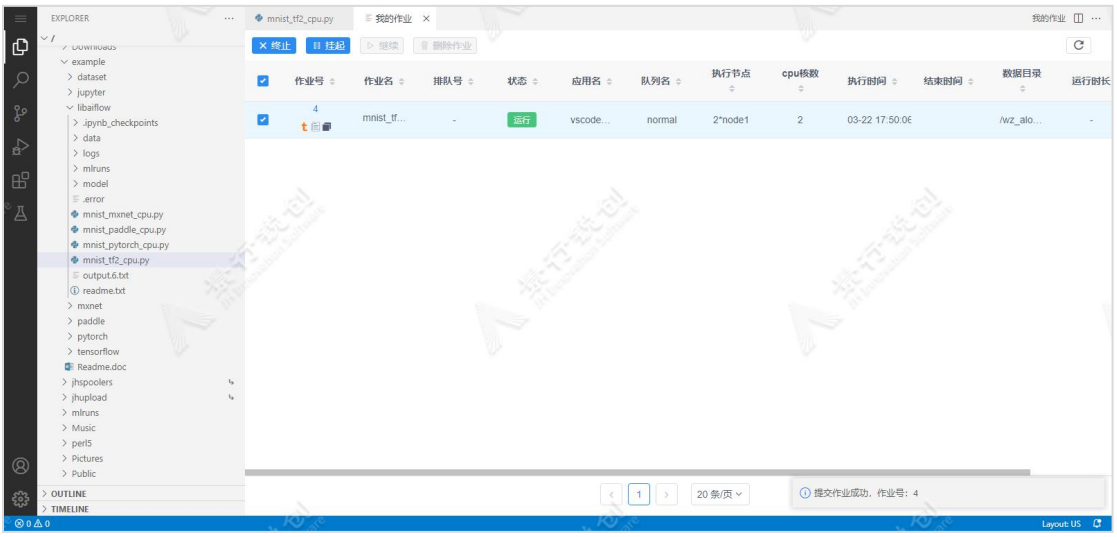
- **单机：**默认选择单机，仅单节点上运行训练程序。
- **PS 并行：**提交以 Parameter Server 方式实现的训练代码，支持跨节点，需填写并行相关参数，如下所示：

- **ParamServer 数：**指定参数服务的数量。
- **Worker 数：**指定训练服务数量。
- **实例规格：**分别指定每个 ParamServer 和 Worker 使用的资源。

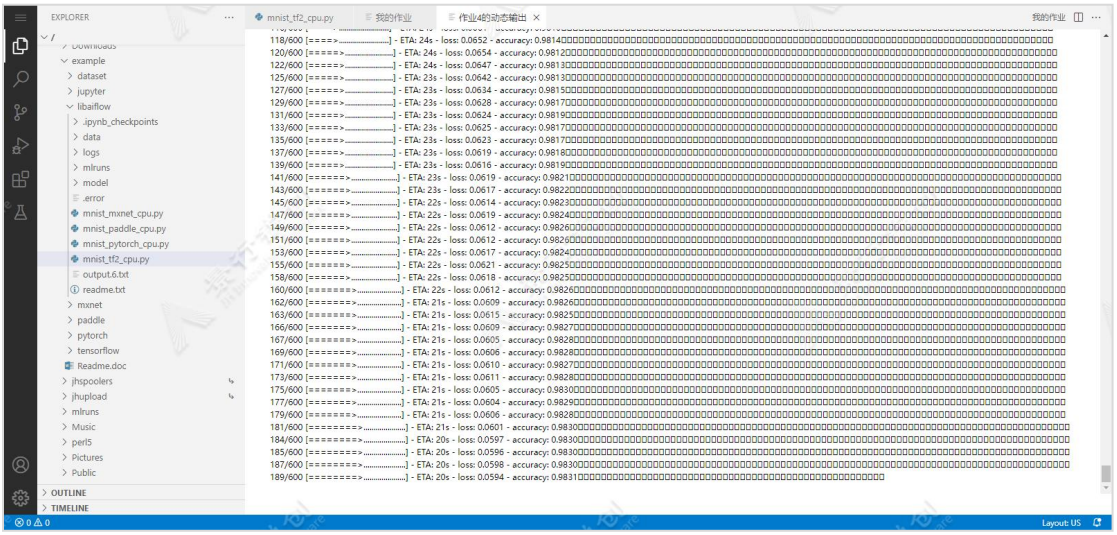
- **Horovod 并行：**提交以 Horovod 实现的并行 AI 训练程序， 需要指定 Horovod 相关参数。Horovod 并行模式需要模型训练代码以 Horovod 库的并行方式进行编码，如下所示：
  - **并行实例数：**指定并行训练的节点数量。
  - **实例规格：**指定每个并行节点使用的资源。

环境变量：指定程序运行需要的额外环境变量。

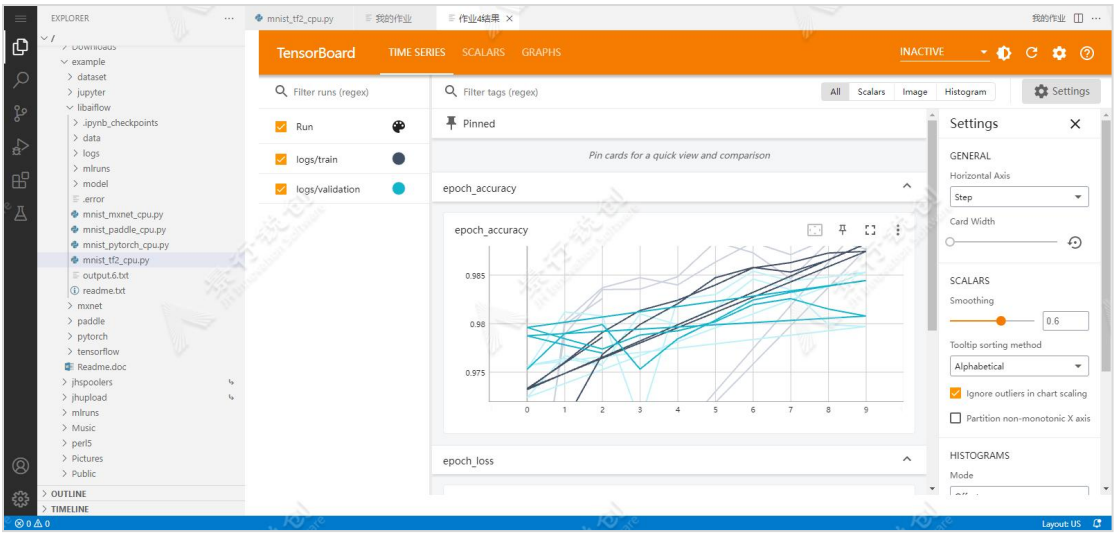
作业提交后，会自动打开“我的作业”页面，方便用户在此查看提交作业的信息，如下图所示：



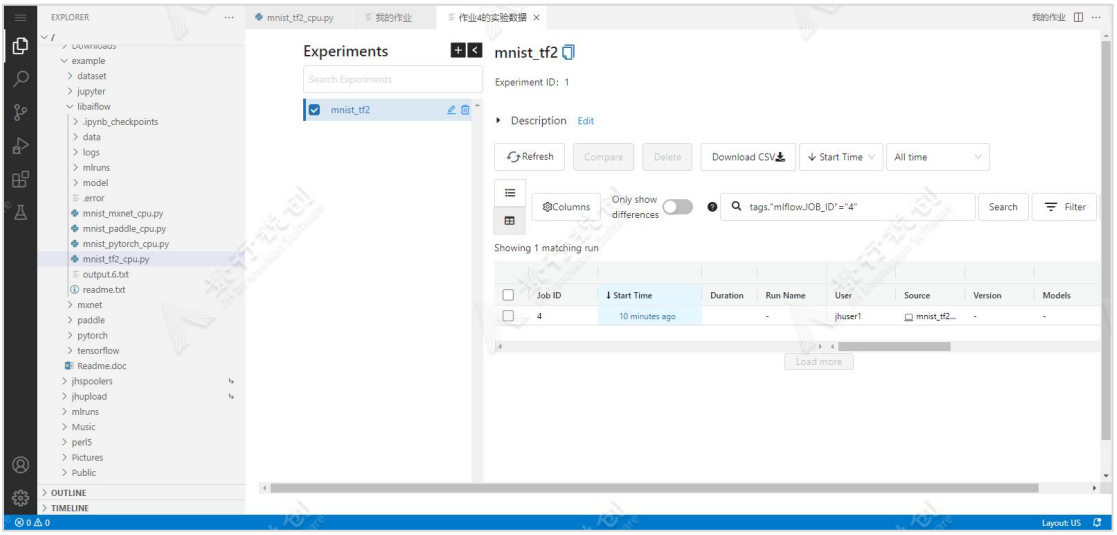
点击作业实例上的“动态输出”图标按钮，查看作业日志的动态输出，如下图所示：



点击作业实例上的“Tensorboard”图标按钮，访问 Tensorboard 服务查看模型数据，如下图所示：



点击作业实例上的“实验管理”图标按钮，查看实验数据，如下图所示：



查看实验数据示意图

2.4.2.3. 应用 CentOs 桌面/Ubuntu 桌面

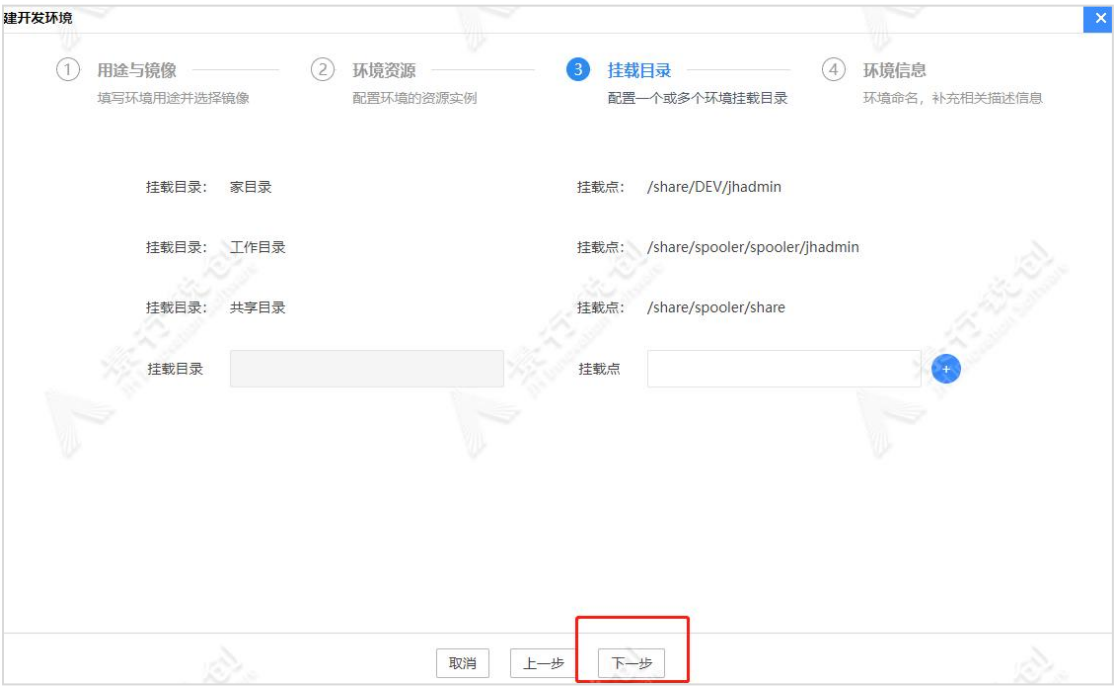
创建 CentOs 桌面/Ubuntu 桌面开发环境，以 CentOs 为例，具体步骤如下：

新建开发环境

在“资源配置”中选择合适的资源规格，然后点击“下一步”：

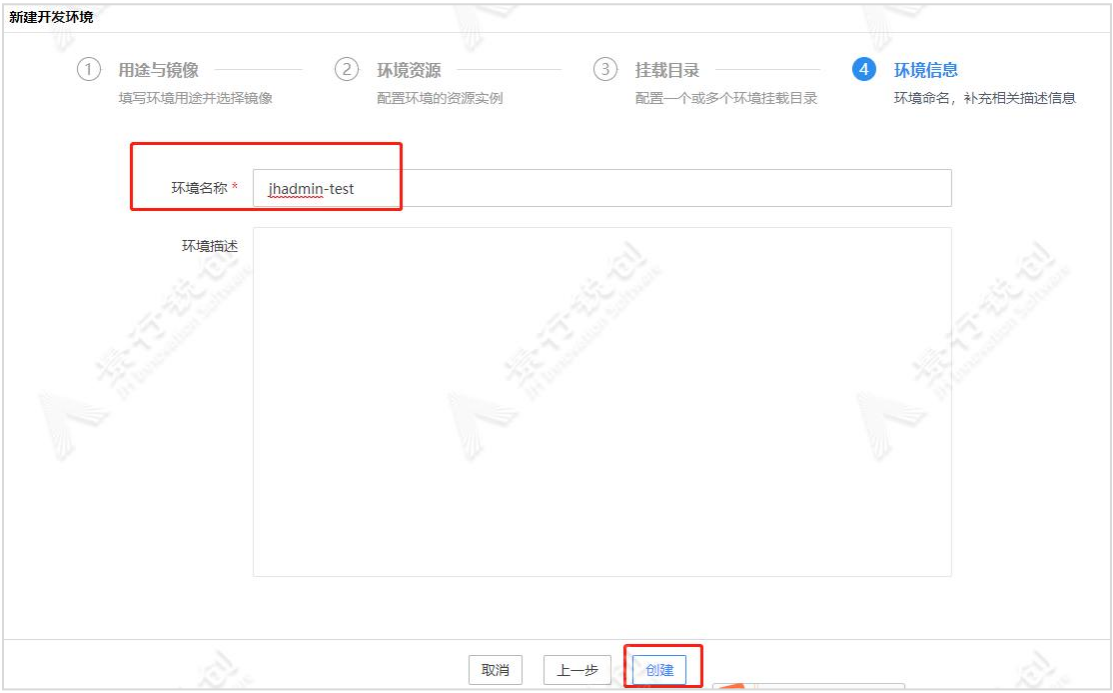
环境资源

在挂载目录页面选择需要挂载的目录，正常情况下使用默认挂载即可，点击“下一步”：



挂载目录

输入环境名称，点击“创建”。

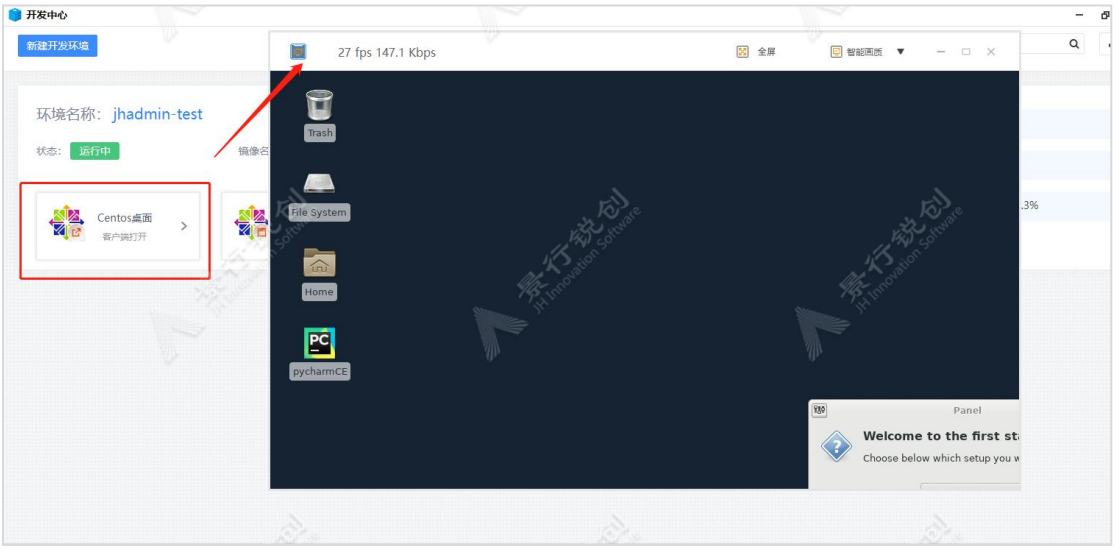


环境信息

此时，在开发中心下可以看到 CentOs 桌面开发环境卡片：

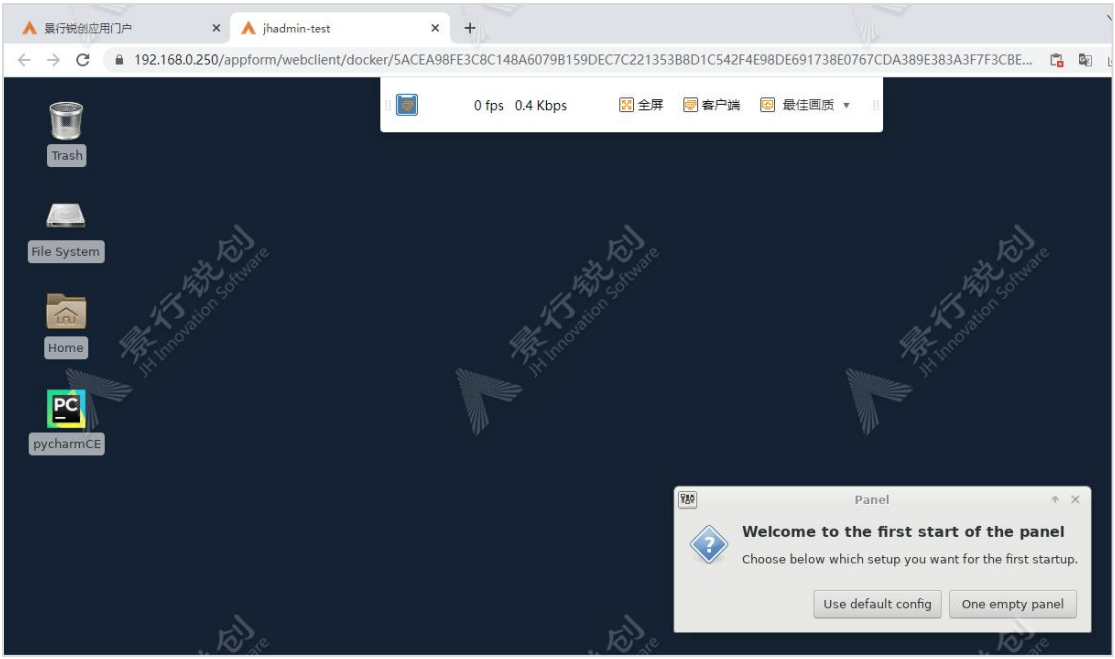
CentOs 桌面卡片

点击“CentOs 桌面 客户端打开”可以用客户端形式打开 CentOs 桌面，前提：机器上需要安装 JH_Client_V5.4_Win_x86_r3****.exe 软件包：



CentOs 桌面客户端界面

点击“Centos 桌面 WEB 打开”，此时会在 WEB 上打开 CentOs 桌面：



CentOs 桌面 WEB 页面

在打开的 Centos 桌面中正常使用即可。

注意：此 CentOs 桌面和普通的 CentOs 物理机桌面稍有不同，例如：该 CentOs 桌面不支持 mount 文件，不支持粘贴文件、图片等。

2.5. AI 作业提交

景行 AI 平台集成了各类主流的深度学习框架，包括 Tensorflow、Pytorch、Mxnet 和 PaddlePaddle，并提供相应的 Web 页面以提交深度学习作业。

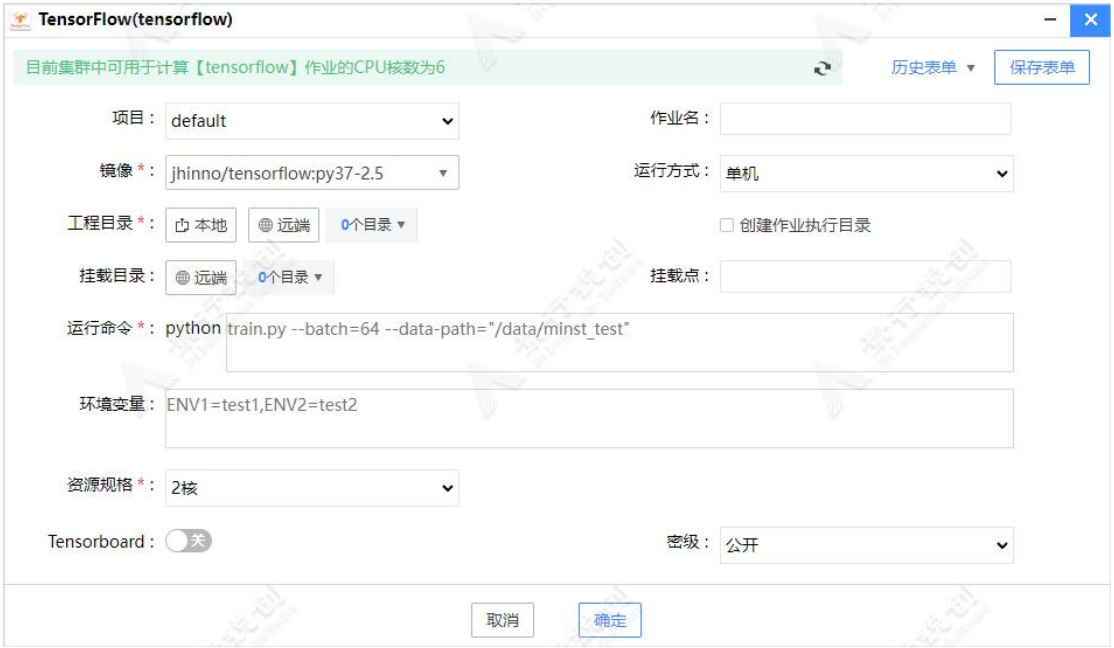
提供 Tensorflow，Pytorch、Mxnet 和 Paddle 等框架编写模型的 GPU 或 CPU 并行训练功能，支持单机多 GPU，多机多 GPU，多机多 CPU 的训练方式。

支持 Tensorflow、Pytorch、Mxnet 和 Paddle 等机器学习作业的容器封装，用户可在 Web 门户中提交容器封装的训练作业，提交时可指定训练作业所需的 CPU、GPU 个数以及内存数量，支持从 web 门户打开该训练作业的屏幕输出、日志和 Tensorboard 界面以查看训练进展，并支持作业的插队、重新提交、暂停和终止等控制功能。

2.5.1. Tensorflow

2.5.1.1. 单机模式

点击“Tensorflow”图标，进入作业提交参数设置页面，如下图所示：



Tensorflow 单机模式作业提交页面

作业提交参数：

项目：指定项目，默认为：default。

作业名：指定作业名，默认为：tensorflow。

镜像：必选项，选择程序运行环境的镜像，需要根据运行程序选择合适的镜像，下拉列表中仅显示包含“tensorflow”关键字的镜像。

运行方式：默认选择单机，仅单节点上运行训练程序。

工程目录：指定代码的工程目录，可以从本地上传或选择服务端已存在的目录。

创建作业执行目录：默认不勾选，直接在工程目录中运行程序。手动勾选，会在“我的作业-作业数据区”中创建临时执行目录，将工程目录中的数据复制到临时目录后，进行运行程序操作。

挂载目录：指定挂载额外的目录到运行容器中，例如：可以选择挂载数据集。

挂载点：挂载目录在容器中对应的目录，未填写时挂载点和挂载目录路径保持一致。

运行命令：必填项，指定运行程序的入口文件和程序参数，例如：train.py --data-path=/data --batch=54。

环境变量：指定程序运行需要的额外环境变量。

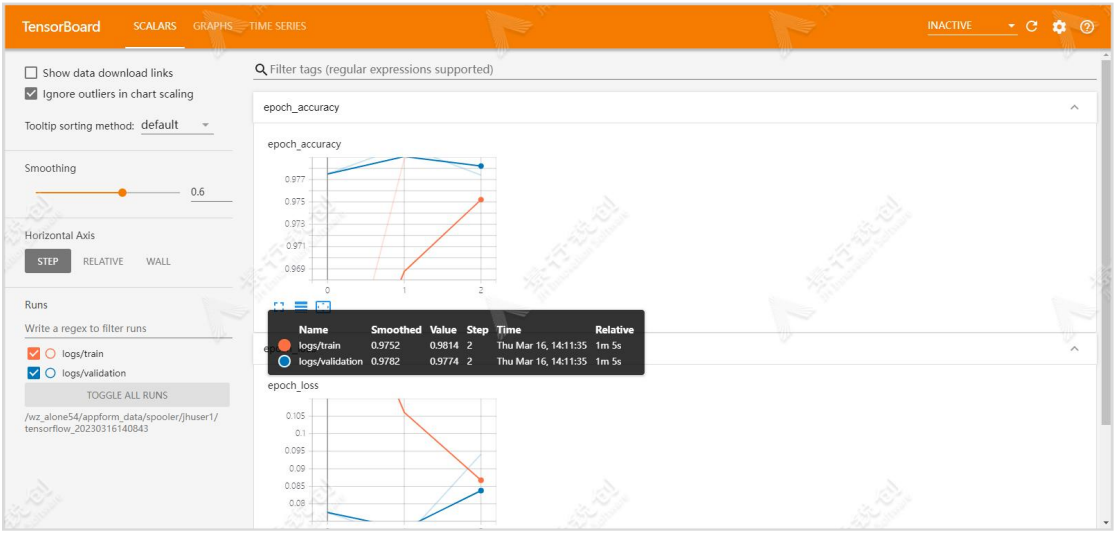
资源规格：默认为 2 核，选择运行程序所需要的资源配置。

Tensorboard：选择是否允许访问 tensorboard 服务。若启动，作业详情页面，点击“访问 Tensorboard”按钮，即可查看模型训练情况。

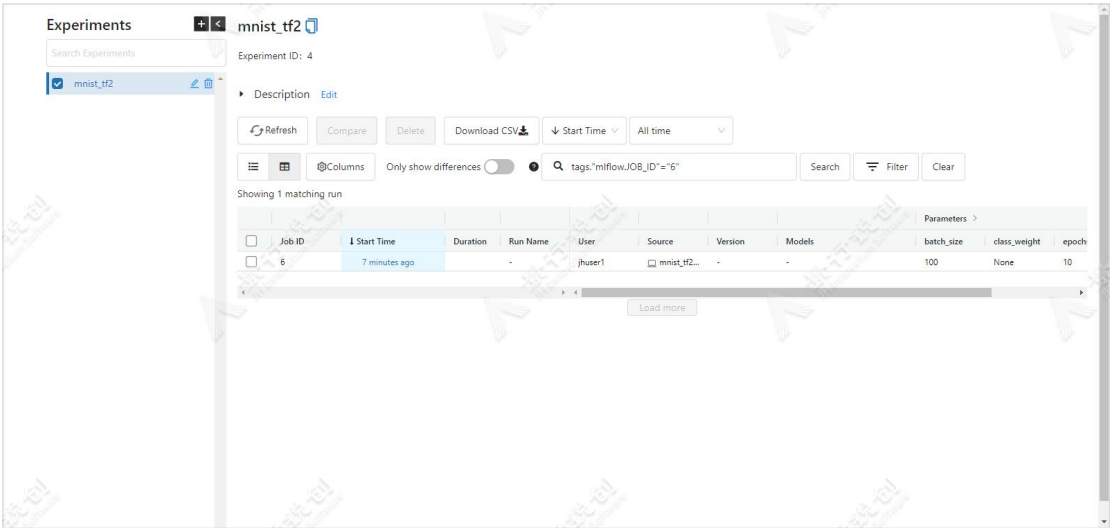
密级：管理员开启密级功能后，显示此选项，默认为用户密级，用于数据安全、保密。

Tensorflow 作业提交后，会自动打开“我的作业”页面，方便用户在此查看提交作业的信息，如下图所示：

点击“访问 tensorboard”按钮，可以查看训练情况，如下图所示：



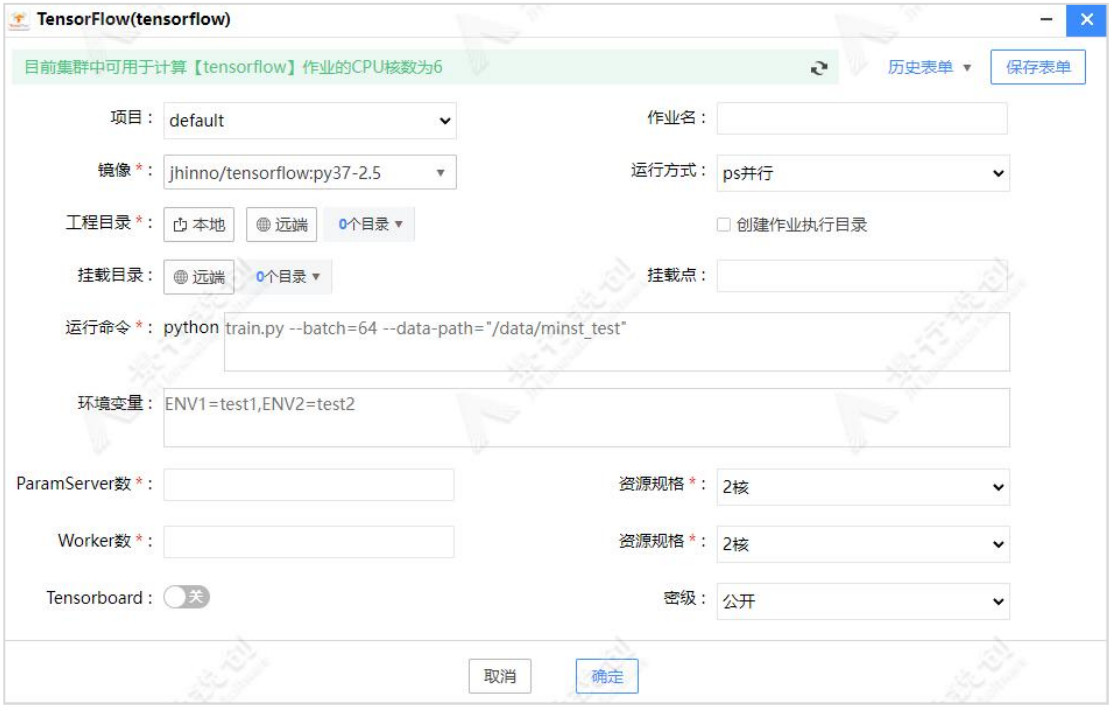
运行程序中集成了 libaiflow api，点击“查看实验数据”按钮，可以查看实验数据，如下图所示：



2.5.1.2. PS 并行模式

提交以 Parameter Server 方式实现的训练代码，支持跨节点，需填写并行相关参数，如下所示：

- **ParamServer 数：**指定参数服务的数量。
- **Worker 数：**指定训练服务数量。
- **资源规格：**分别指定每个 ParamServer 和 Worker 使用的资源。



Tensorflow PS 并行模式作业提交页面

2.5.1.3. Horovod 并行模式

提交以 Horovod 实现的并行 AI 训练程序， 需要指定 Horovod 相关参数。Horovod 并行模式需要模型训练代码以 Horovod 库的并行方式进行编码，如下所示：

- **并行实例数：**指定并行训练的节点数量。
- **资源规格：**指定每个并行节点使用的资源。

Tensorflow Horovod 并行模式作业提交页面

2.5.2. Pytorch

2.5.2.1. 单机模式

点击“Pytorch”图标，进入作业提交参数设置页面，如下图所示：

Pytorch 单机模式作业提交页面

作业提交参数：

项目：指定项目，默认为：default。

作业名：指定作业名，默认为：pytorch。

镜像：必选项，选择程序运行环境的镜像，需要根据运行程序选择合适的镜像，下拉列表中仅显示包含“pytorch”关键字的镜像。

运行方式：默认选择单机，仅单节点上运行训练程序。

工程目录：指定代码的工程目录，可以从本地上传或选择服务端已存在的目录。

创建作业执行目录：默认不勾选，直接在工程目录中运行程序；手动勾选，会在“我的作业-作业数据区”中创建临时执行目录，将工程目录中的数据复制至其中后，进行运行程序操作。

挂载目录：指定挂载额外的目录到运行容器中，例如：可以选择挂载数据集。

挂载点：挂载目录在容器中对应的目录，未填写时挂载点和挂载目录路径保持一致。

运行命令：必填项，指定运行程序的入口文件和程序参数，例如：train.py --data-path=/data --batch=54。

环境变量：指定程序运行需要的额外环境变量。

资源规格：默认为 2 核，选择运行程序所需要的资源配置。

Tensorboard：选择是否启动 tensorboard 服务。若启动，作业详情页面，点击“访问 Tensorboard”按钮，即可查看模型训练情况。

密级：管理员开启密级功能后，显示此选项，默认为用户密级，用于数据安全、保密。

2.5.2.2. Horovod 并行模式

提交以 Horovod 实现的并行 AI 训练程序， 需要指定 Horovod 相关参数。Horovod 并行模式需要模型训练代码以 Horovod 库的并行方式进行编码，如下所示：

- **并行节点数：**指定并行训练的节点数量。
- **资源规格：**指定每个并行节点使用的资源。

Pytorch Horovod 并行模式作业提交页面

2.5.3. Mxnet

Mxnet 是开源深度学习框架，允许用户在多种设备上定义、训练和部署深度神经网络。该框架具备高度可扩展性，可以进行快速的模型训练，并支持灵活的编程模型。

2.5.3.1. 单机模式

点击“Mxnet”图标，进入作业提交参数设置页面，如下图所示：

Mxnet 单机模式作业提交页面

作业提交参数：

项目：指定项目，默认为：default。

作业名：指定作业名，默认为：mxnet。

镜像：必选项，选择程序运行环境的镜像，需要根据运行程序选择合适的镜像，下拉列表中仅显示包含“mxnet”关键字的镜像。

运行方式：默认选择单机，仅单节点上运行训练程序。

工程目录：指定代码的工程目录，可以从本地上传或选择服务端已存在的目

录。

创建作业执行目录：默认不勾选，直接在工程目录中运行程序；手动勾选，会在“我的作业-作业数据区”中创建临时执行目录，将工程目录中的数据复制至其中后，进行运行程序操作。

挂载目录：指定挂载额外的目录到运行容器中，例如：可以选择挂载数据集。

挂载点：挂载目录在容器中对应的目录，未填写时挂载点和挂载目录路径保持一致。

运行命令：必填项，指定运行程序的入口文件和程序参数，例如：train.py --data-path=/data --batch=54。

环境变量：指定程序运行需要的额外环境变量。

资源规格：默认为 2 核，选择运行程序所需要的资源配置。

密级：管理员开启密级功能后，显示此选项，默认为用户密级，用于数据安全、保密。

2.5.3.2. Horovod 并行模式

提交以 Horovod 实现的并行 AI 训练程序， 需要指定 Horovod 相关参数。Horovod 并行模式需要模型训练代码以 Horovod 库的并行方式进行编码，如下所示：

- **并行节点数：**指定并行训练的节点数量。
- **资源规格：**指定每个并行节点使用的资源。

Mxnet Horovod 并行模式作业提交页面

2.5.4. PaddlePaddle

飞桨(PaddlePaddle)是集深度学习核心框架、工具组件和服务平台为一体的技术先进、功能完备的开源深度学习平台。

2.5.4.1. 单机单卡

点击“PaddlePaddle”图标，进入作业提交参数设置页面，如下图所示：

PaddlePaddle 单机模式作业提交页面

作业提交参数：

项目：指定项目，默认为：default。

作业名：指定作业名，默认为：paddle。

镜像：必选项，选择程序运行环境的镜像，需要根据运行程序选择合适的镜像，下拉列表中仅显示包含“paddle”关键字的镜像。

运行方式：默认选择单机，仅单节点上运行训练程序。

工程目录：指定代码的工程目录，可以从本地上传或选择服务端已存在的目录。

创建作业执行目录：默认不勾选，直接在工程目录中运行程序；手动勾选，会在“我的作业-作业数据区”中创建临时执行目录，将工程目录中的数据复制到临时目录后，进行运行程序操作。

挂载目录：指定挂载额外的目录到运行容器中，例如：可以选择挂载数据集。

挂载点：挂载目录在容器中对应的目录，未填写时挂载点和挂载目录路径保持一致。

运行命令：必填项，指定运行程序的入口文件和程序参数，例如：train.py --data-path=/data --batch=54。

环境变量：指定程序运行需要的额外环境变量。

资源规格：默认为 2 核，选择运行程序所需要的资源配置。

密级：管理员开启密级功能后，显示此选项，默认为用户密级，用于数据安全、保密。

2.5.4.2. 单机多卡

提交单机多卡 PaddlePaddle 作业，需要在运行命令处额外配置“-m paddle.distributed.launch”参数，具体操作详情如下图所示：

PaddlePaddle 单机多卡模式作业提交页面

2.6. 实验管理

实验管理为景行人工智能软件平台提供跟踪、比较和记录模型训练指标的工具和平台，是保存每个运行实验所关注的所有实验相关信息的过程。实验训练过程中的参数，超参数，指标，数字，图像，文件等信息会记录到数据库中，通过可视化界面供用户查看。

涉及的一些关键字：

实验（experiment）：相当于项目名，所有的调参都是针对这个实验。

运行（Run）：每次调参的运行记录。

参数(parameter)：每次调参所记录的参数。

指标(metric)：记录每次调参之后的指标结果。

文件(artifact)：记录文件。

2.6.1. 准备训练程序

准备实验训练程序，并在实验训练程序中集成 libaiflow API。

方式一：通过 libaiflow.tensorflow.autolog 和 libaiflow.pytorch.autolog 提供的 Callback 机制支持快速简易的收集 tensorflow 和 pytorch 实验的参数，指标等多种实验训练数据。下面仅展示 autolog 自动收集数据的关键代码，（完整样例代码参考附录七样例一）。

```
import libaiflow
expt = libaiflow.init(experiment_name="mnist_tf2_cpu")
libaiflow.tensorflow.autolog()
```

方式二：通过指标和超参数相关 api 记录和收集数据，（完整样例代码参考附录七样例二）。

```
import libaiflow
expt = libaiflow.init(experiment_name="pytorch-sample")
......
with libaiflow.start_run(experiment_id=expt.experiment_id):
    lr = ElasticNet(alpha=alpha, l1_ratio=l1_ratio, random_state=42)
    lr.fit(train_x, train_y)
    predicted_qualities = lr.predict(test_x)
    (rmse, mae, r2) = eval_metrics(test_y, predicted_qualities)
    print("Elasticnet model (alpha=%f, l1_ratio=%f):" % (alpha,
```

```
l1_ratio))
    print("  RMSE: %s" % rmse)
    print("  MAE: %s" % mae)
    print("  R2: %s" % r2)
    libaiflow.log_param("alpha", alpha)
    libaiflow.log_param("l1_ratio", l1_ratio)
    libaiflow.log_metric("rmse", rmse)
    libaiflow.log_metric("r2", r2)
    libaiflow.log_metric("mae", mae)
```

2.6.2. 执行训练程序

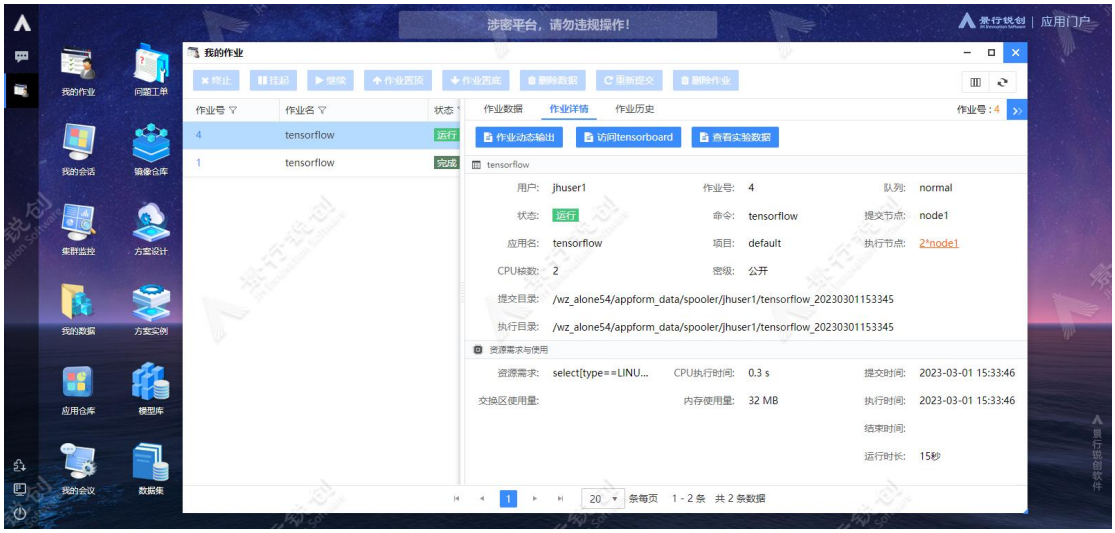
景行人工智能平台的多个功能已集成 libaiflow 库，如：提交 AI 作业、运行模型结构的方案、Jupyter 和 VSCode 等。

2.6.2.1. 提交 AI 作业

AI 作业默认集成 libaiflow 库，应用集成 libaiflow API 的程序，提交 AI 作业，进行模型训练。以提交 Tensorflow 作业为例，如下图所示：

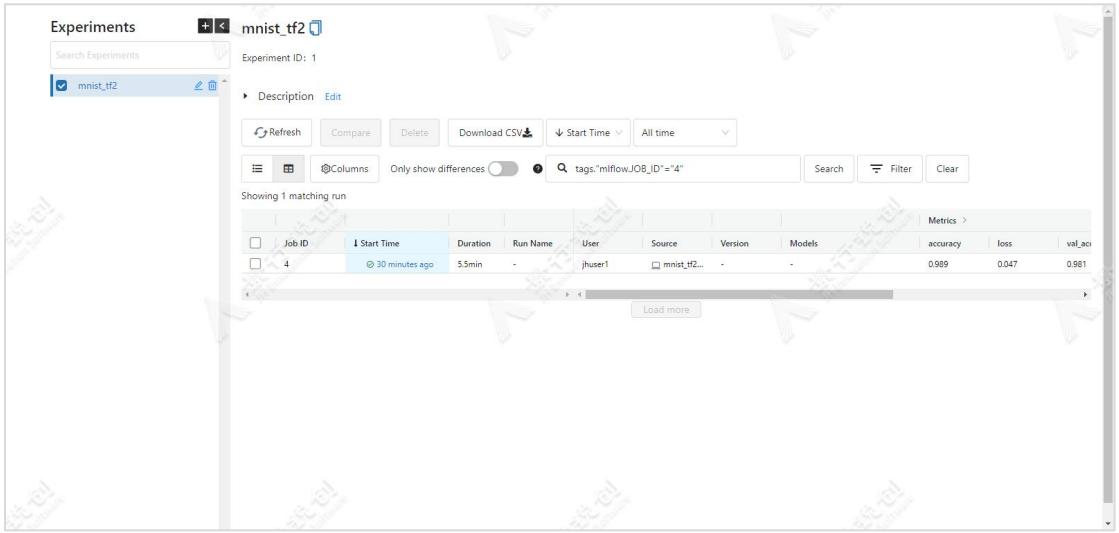
提交 AI 作业示意图

Tensorflow 作业提交后，会自动打开“我的作业”页面，方便用户在此查看提交作业的信息，如下图所示：



注意：选择集成 libaiflow api 的训练程序，才能查看实验数据。

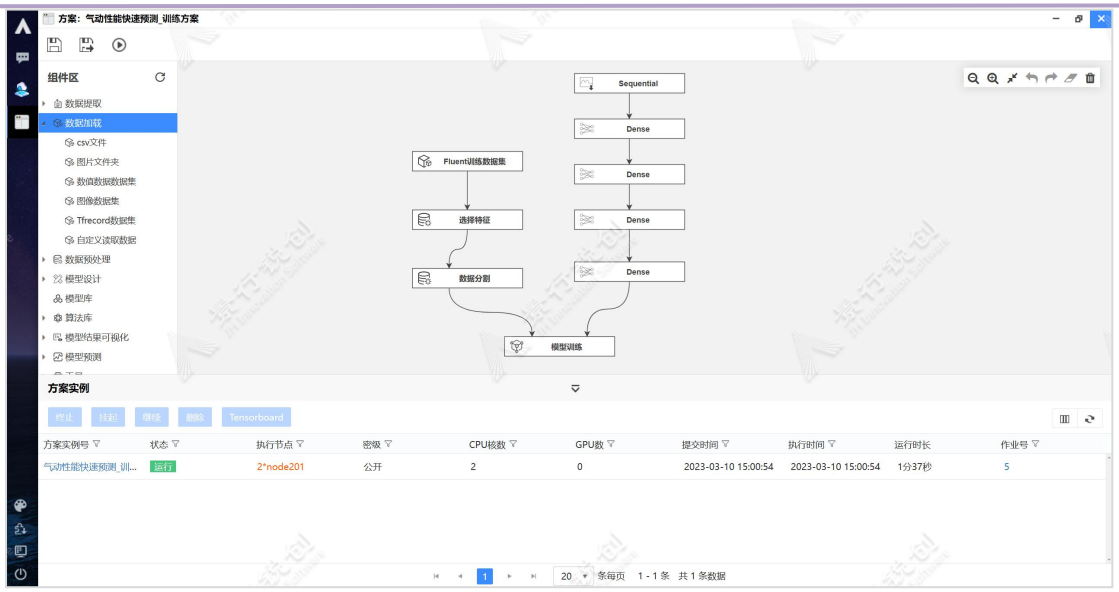
点击“查看实验数据”按钮，可以查看实验数据，如下图所示：



查看实验数据示意图

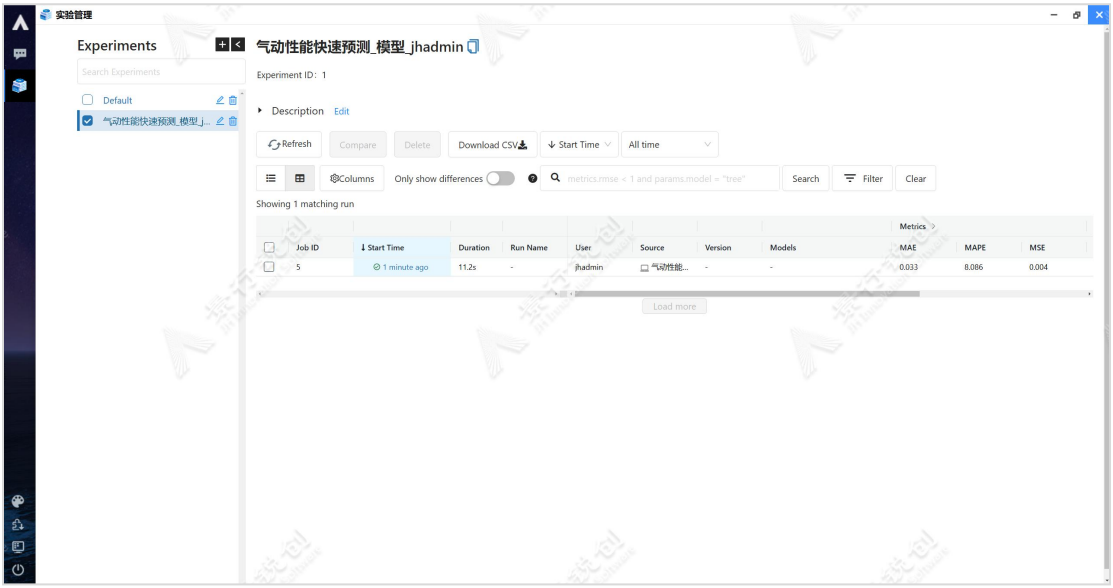
2.6.2.2. 运行模型结构的方案

打开方案设计管理，进入方案设计器界面，构建模型结构流程、训练模型，如下图所示：



运行方案示意图

在门户桌面，双击“实验管理”桌面图标，可以查看实验数据，如下图所示：



查看实验数据

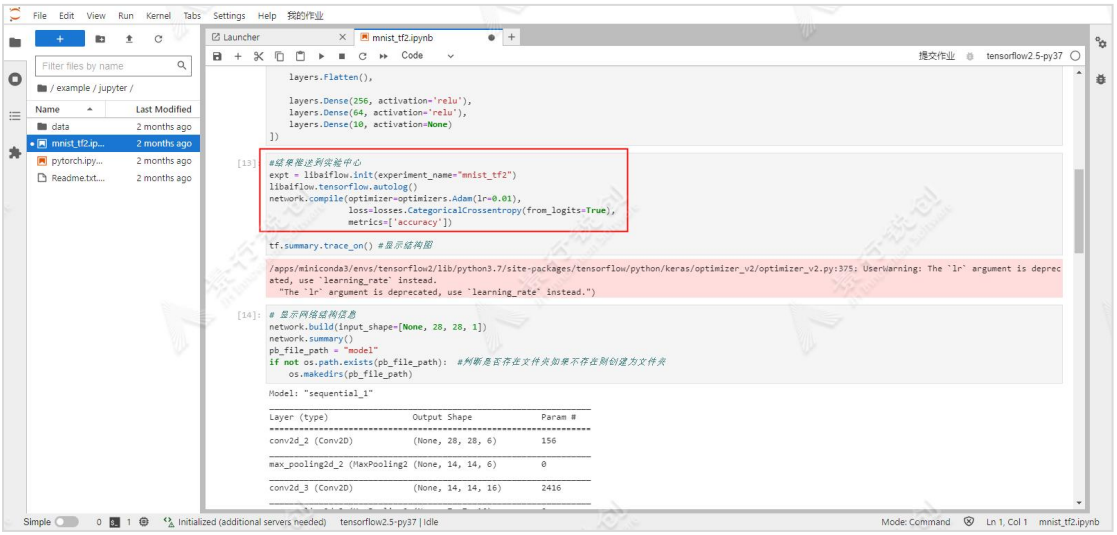
2.6.2.3. 从开发中心运行

WebIDE 类型的开发环境默认集成 libaiflow 库，访问“Jupyter”和“VSCode”服务训练程序或者提交 AI 作业，进行模型训练。以通过 Jupyter 服务应用 libaiflow API 为例，操作步骤如下：双击“开发中心”桌面图标，打开“开发

中心管理”界面，如下图所示：

开发中心管理示意图

点击“Jupyter”卡片按钮，访问 Jupyter 服务，选择集成 libaiflow API 的训练程序，执行训练程序，或通过提交作业方式训练程序。如下图所示：

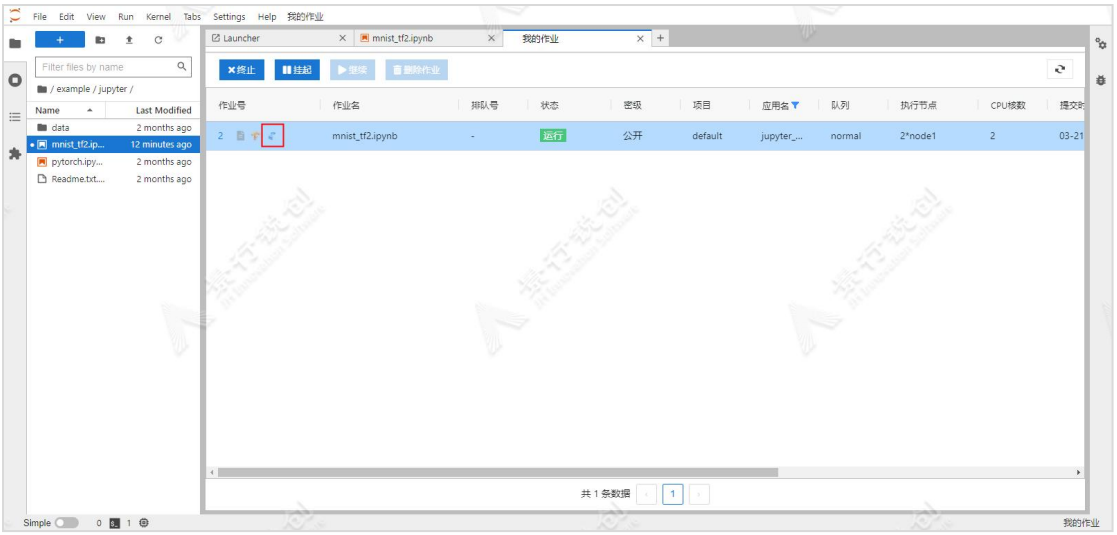


访问“Jupyter”服务示意图

点击“提交作业”按钮，提交作业，如下图所示：

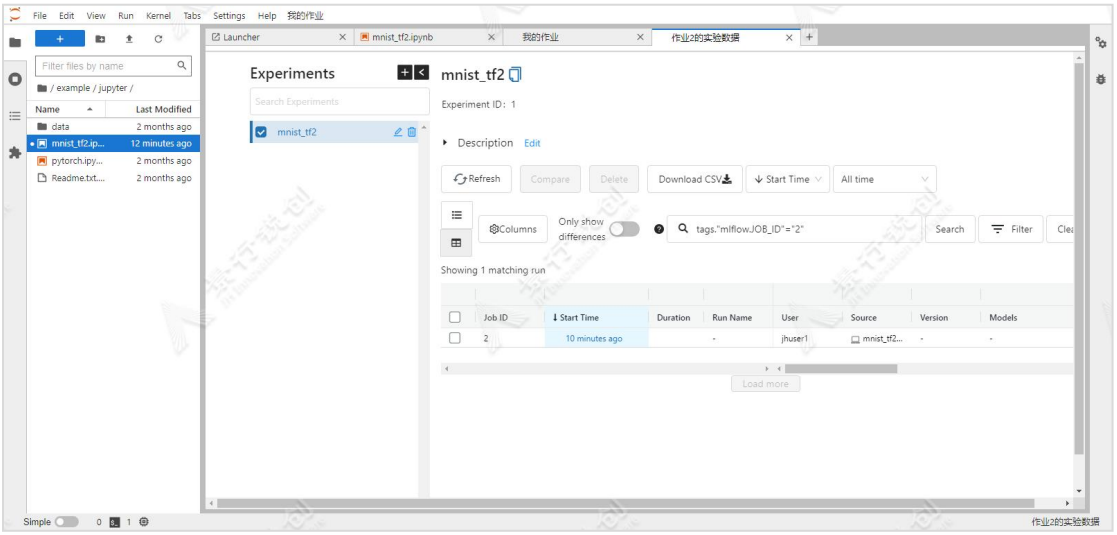
提交 AI 作业示意图

点击“我的作业”按钮，打开我的作业 tab 页，如下图所示：



我的作业示意图

点击作业实例上的“实验管理”图标按钮，查看实验数据，如下图所示：



查看实验数据示意图

2.7. 服务管理

服务管理支持“容器终端”、“Tensorboard”、“Tensorflow Serving”和“Torch Serving”类型的服务。如下所示：

- **容器终端：**仅启动一个 web ssh 服务。
- **Tensorboard：**用于可视化查看模型数据。
- **Tensorflow Serving：**将训练的 Tensorflow 模型发布为 Web 服务。
- **Torch Serving：**将训练的 Pytorch 模型发布为 Web 服务。

各个角色对服务实例的操作权限：

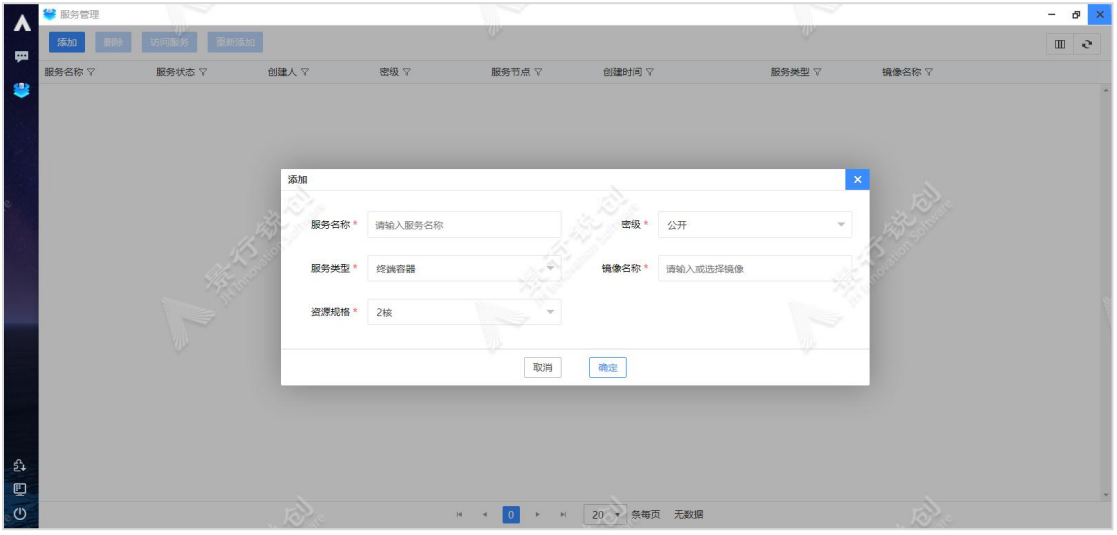
添加者：对服务实例具有添加、重新添加、访问服务以及删除服务的权限；

管理员：对其他人添加的服务实例仅有查看服务实例信息和删除服务的权限。

2.7.1. 添加服务

2.7.1.1. “终端容器”类型的服务

点击“添加”按钮，弹出“添加”窗口，选择启动的服务类型和填写相关参数，如下图所示：



添加“终端容器”类型的服务配置示意图

图中每个参数的具体含义如下：

服务名称：用户自定义。

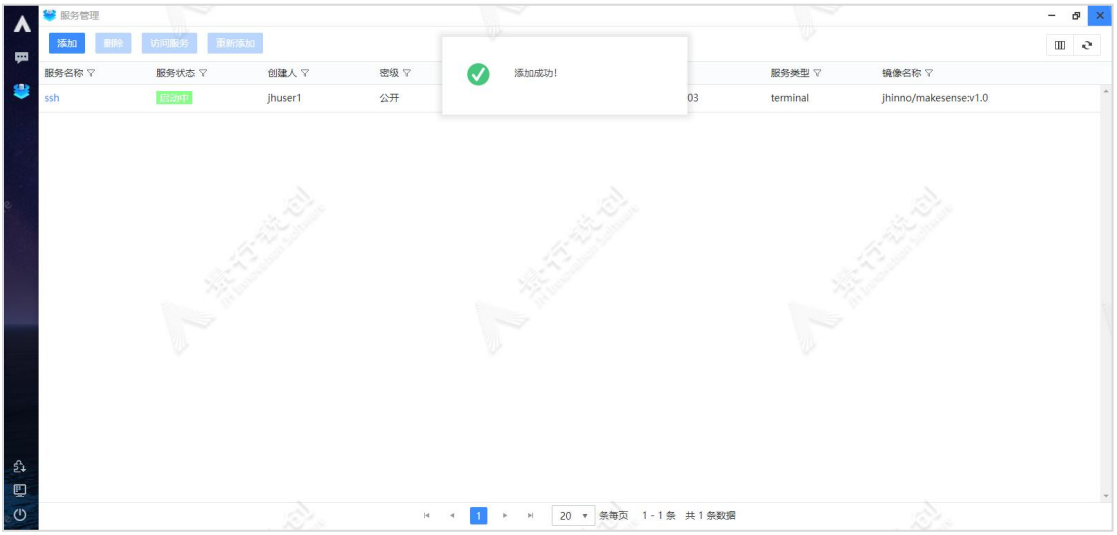
密级：管理员开启密级功能后，显示此选项，默认为用户密级，用于数据安全、保密。

服务类型：选择启动的服务类型，默认为“终端容器”。

镜像名称：通过下拉列表选择需要的镜像。

资源规格：选择启动服务所需要的资源配置，默认为 2 核。

填写完成后，点击“确定”按钮，添加服务实例，如下图所示：



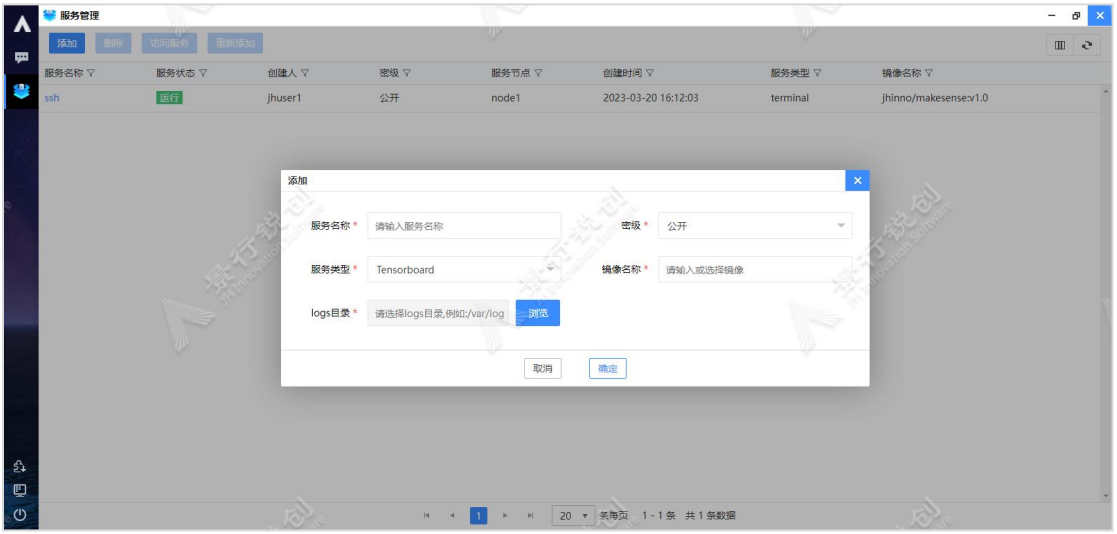
添加“终端容器”类型的服务示意图

2.7.1.2. “Tensorboard”类型的服务

添加“Tensorboard”类型的服务有 2 种方式，如下所示：

- 当服务管理中没有“Tensorboard”类型的服务时，在我的作业、方案设计和模型库中，可以通过点击“访问 Tensorboard”按钮，添加“Tensorboard”类型的服务。
- 进入服务管理，手动添加“Tensorboard”类型的服务。

以手动添加“Tensorboard”类型的服务为例，操作步骤如下：点击“添加”按钮，弹出“添加”窗口，选择启动的服务类型和填写相关参数，如下图所示：



添加“Tensorboard”类型的服务配置示意图

图中每个参数的具体含义如下：

服务名称：用户自定义。

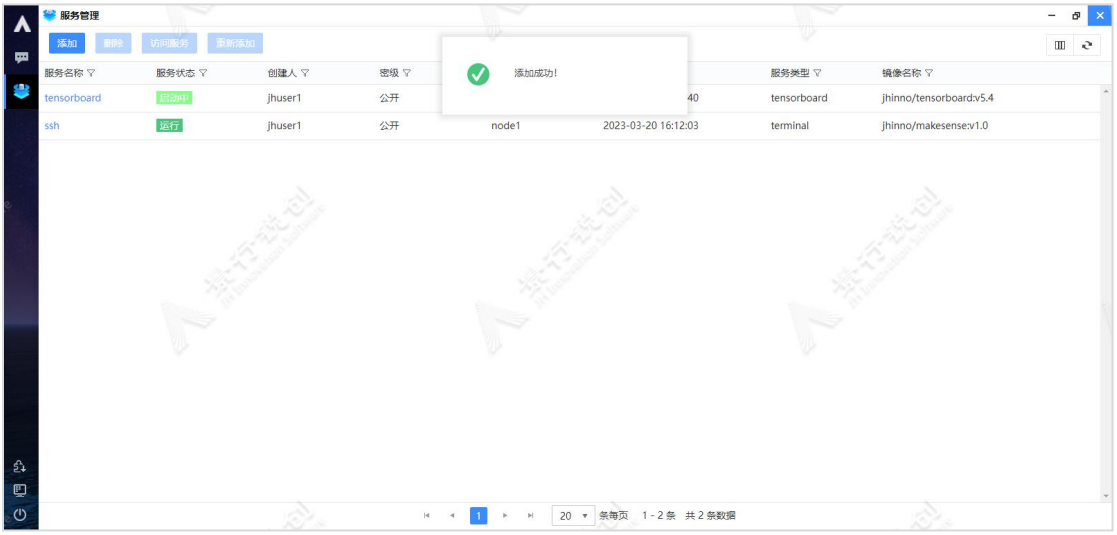
密级：管理员开启密级功能后，显示此选项，默认为用户密级，用于数据安全、保密。

服务类型：选择“Tensorboard”。

镜像名称：通过下拉列表选择镜像。

Logs 地址：单击浏览按钮选择路径。

填写完成后，点击“确定”按钮，添加服务实例，如下图所示：



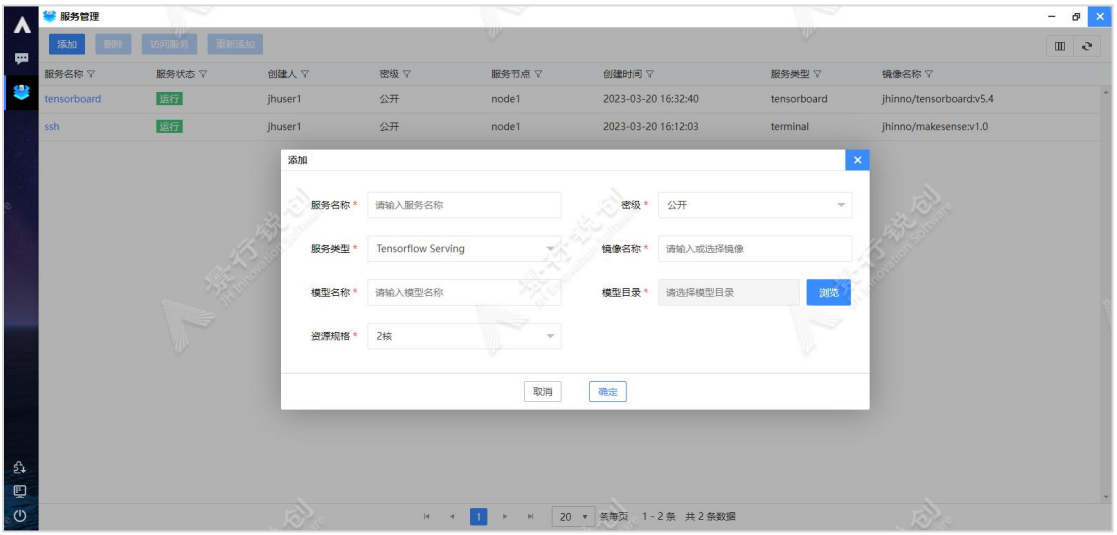
添加“Tensorboard”类型的服务示意图

2.7.1.3. “Tensorflow Serving”类型的服务

添加“Tensorflow Serving”类型的服务有 2 种方式，如下所示：

- 进入模型库，点击模型卡片，进入模型实例列表，选中一个模型实例后，点击“模型部署”按钮，添加“Tensorflow Serving”类型的服务。
- 进入服务管理，手动添加“Tensorflow Serving”类型的服务。

以手动添加“Tensorflow Serving”类型的服务为例，操作步骤如下：点击“添加”按钮，弹出“添加”窗口，选择启动的服务类型和填写相关参数，如下图所示：



添加“Tensorflow Serving”类型的服务配置示意图

图中每个参数的具体含义如下：

服务名称：用户自定义。

密级：管理员开启密级功能后，显示此选项，默认为用户密级，用于数据安全、保密。

服务类型：选择“Tensorflow Serving”。

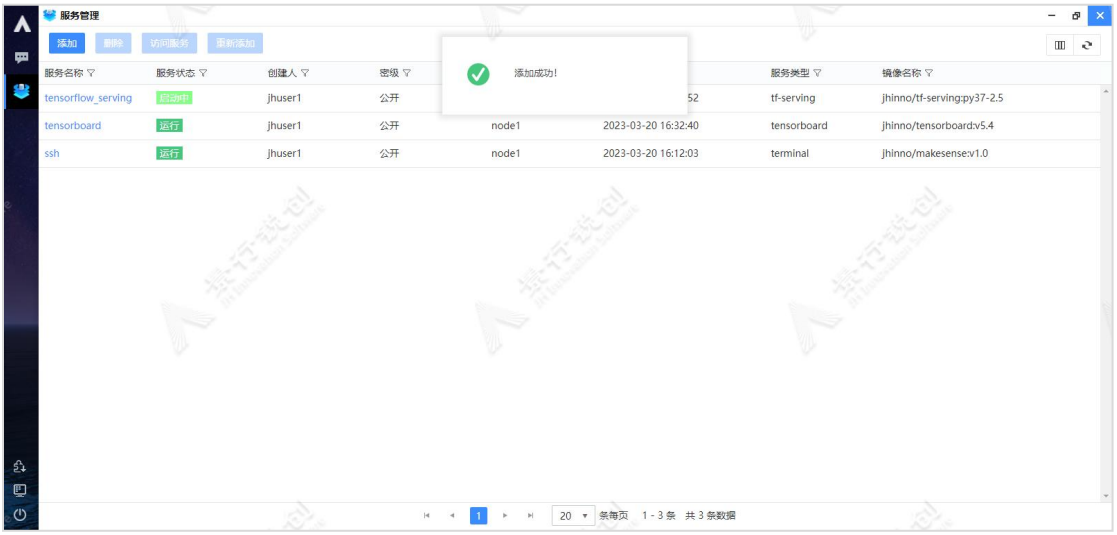
镜像名称：通过下拉列表选择镜像。

模型名称：用户自定义。

模型目录：选择模型结果所在的目录。

资源规格：选择启动服务所需要的资源配置，默认为 2 核。

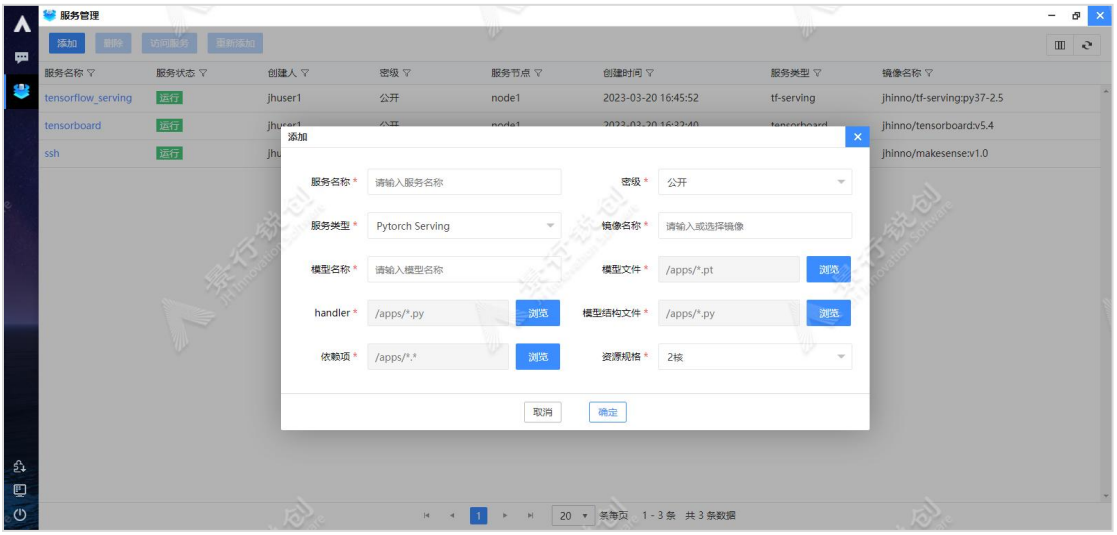
填写完成后，点击“确定”按钮，添加服务实例，如下图所示：



添加“Tensorflow Serving”类型的服务示意图

2.7.1.4. “Pytorch Serving”类型的服务

点击“添加”按钮，弹出“添加”窗口，选择启动的服务类型和填写相关参数，如下图所示：



添加“Pytorch Serving”类型的服务配置示意图

图中每个参数的具体含义如下：

服务名称：用户自定义。

密级：管理员开启密级功能后，显示此选项，默认为用户密级，用于数据安全、保密。

服务类型：选择“Pytorch Serving”。

镜像名称：通过下拉列表选择镜像。

模型名称：用户自定义。

模型文件：选择*.pt 模型文件。

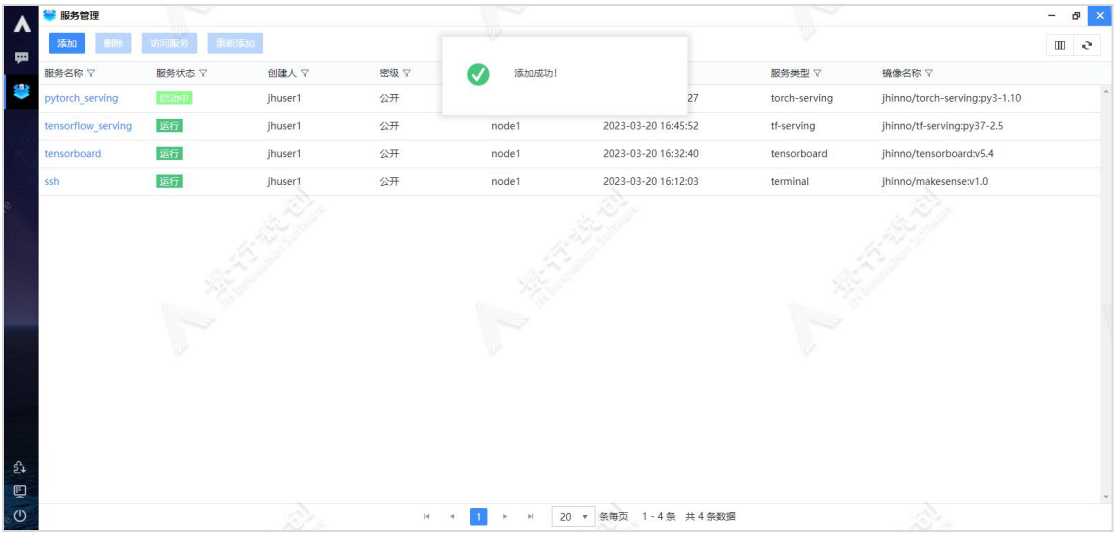
handler：选择*.py 文件。

模型结构文件：选择*.py 模型结构文件。

依赖项：选择依赖项。

资源规格：选择启动服务所需要的资源配置，默认为 2 核。

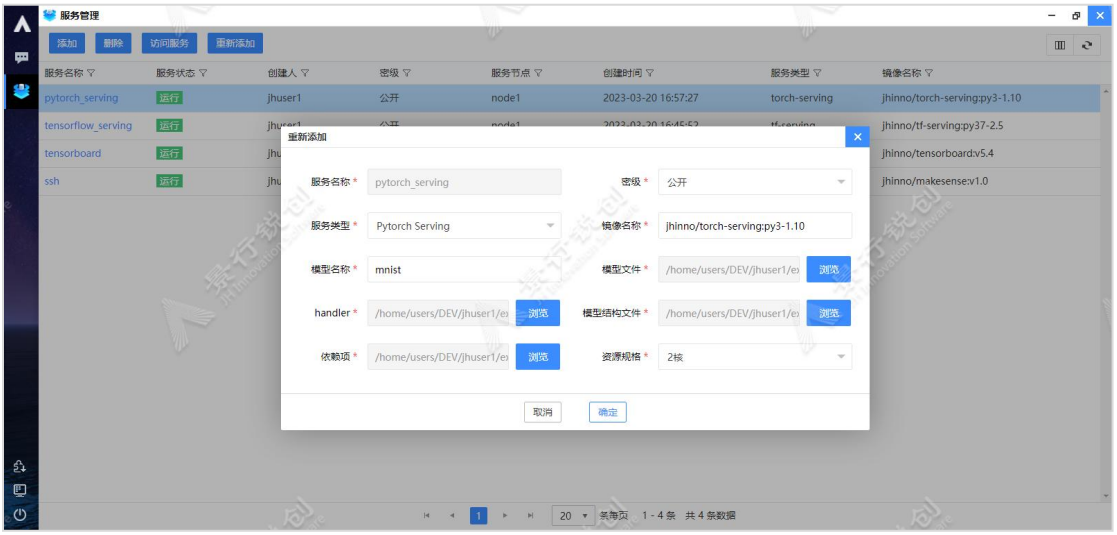
填写完成后，点击“确定”按钮，添加服务实例，如下图所示：



添加“Pytorch Serving”类型的服务示意图

2.7.2. 重新添加服务

以重新添加“Tensorflow Serving”类型的服务为例，操作步骤如下：选中一条服务实例后，点击“重新添加”按钮，弹出“重新添加”窗口，如下图所示：



重新添加“Pytorch Serving”类型的服务配置示意图

图中每个参数的具体含义如下：

服务名称：服务名称只读，默认回填该服务实例的服务名称。

密级：管理员开启密级功能后，显示此选项，默认为该服务实例的密级。

服务类型：Pytorch Serving。

镜像名称：通过下拉列表选择镜像。

模型名称：用户自定义。

模型文件：选择*.pt 模型文件。

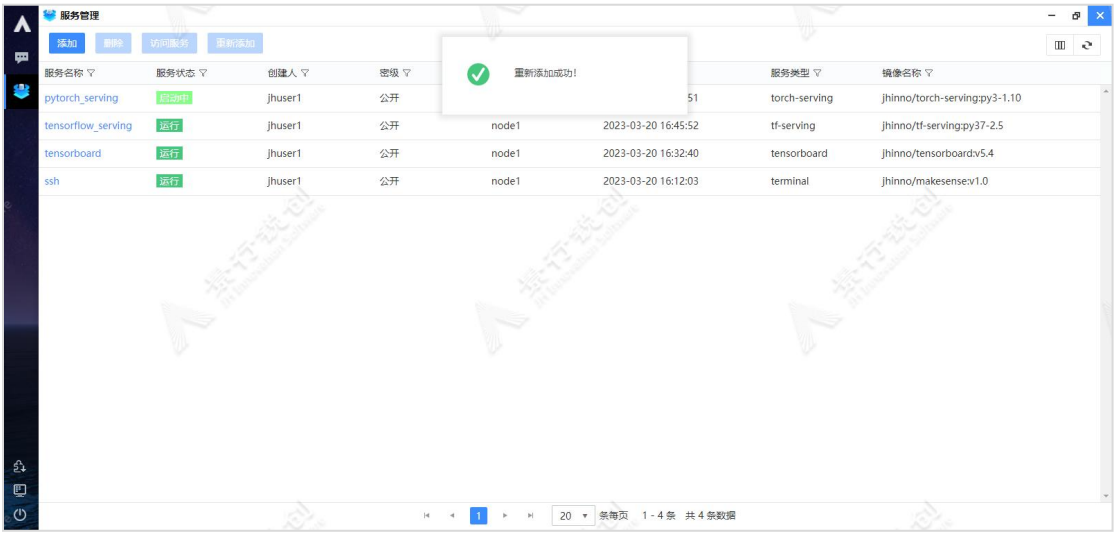
handler：选择*.py 文件。

模型结构文件：选择*.py 模型结构文件。

依赖项：选择依赖项。

资源规格：选择启动服务所需要的资源配置，默认为 2 核。

填写完成后，点击“确定”按钮，重新添加服务实例，如下图所示：

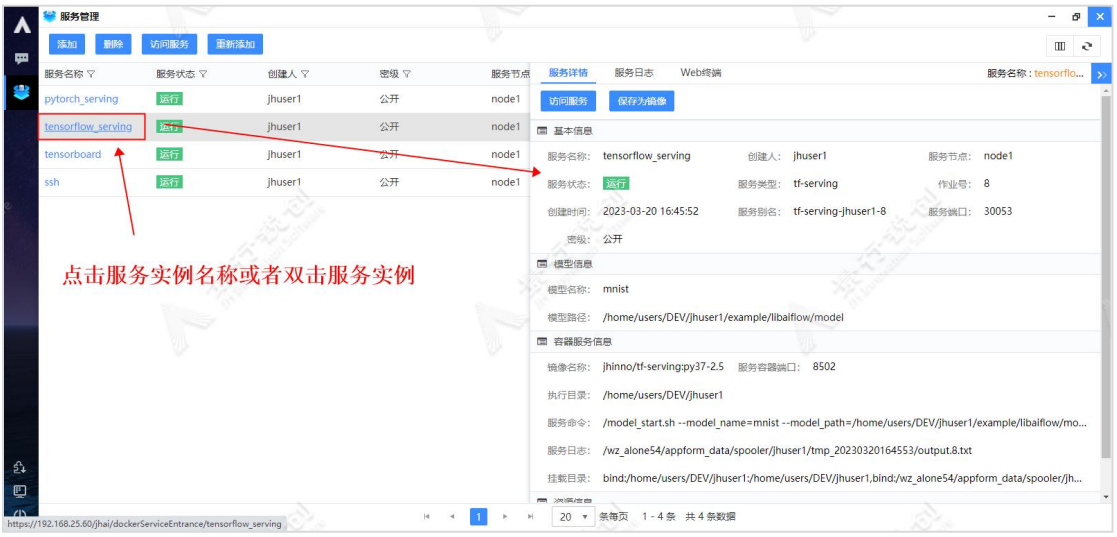


重新添加“Pytorch Serving”类型的服务示意图

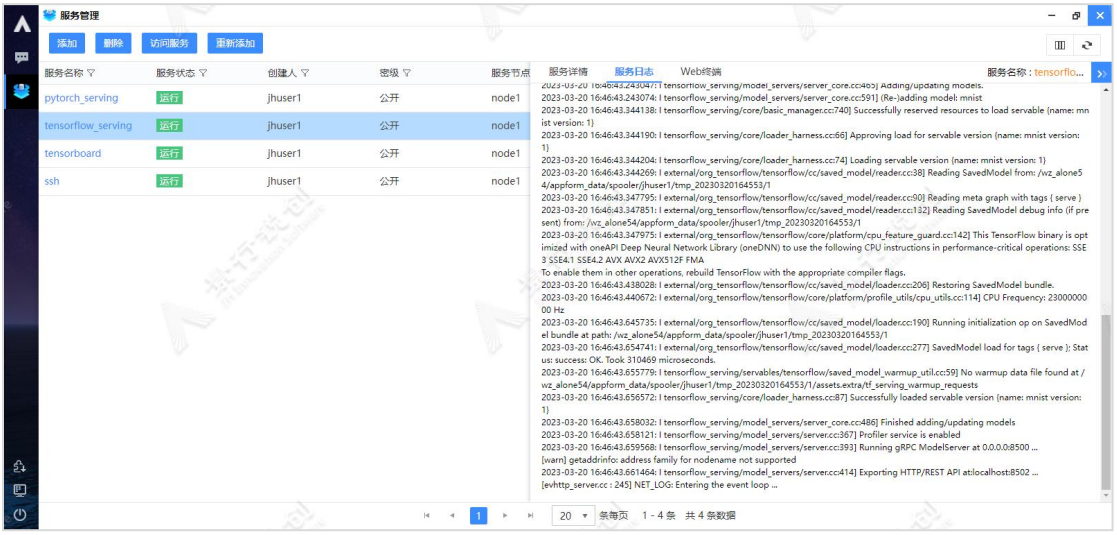
2.7.3. 查看服务信息

点击服务列表中的某一条服务实例的服务名称或者双击某一条服务实例，会打开右侧滑块。滑块中有三个标签页：服务详情、服务日志以及 Web 终端。

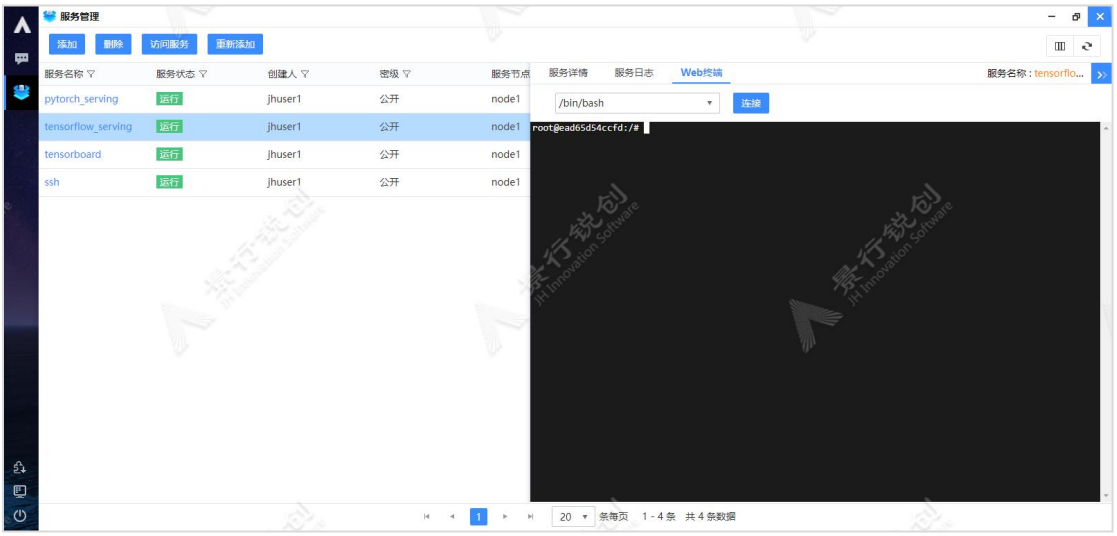
服务详情：显示服务基本信息、模型信息、容器服务信息和资源信息，如下图所示：



服务日志：点击“服务日志”按钮，切换至服务日志页面，查看服务日志。当该服务的状态处于“启动中”和“运行”状态时，会动态输出该服务的日志，否则会静态输出所有日志，具体详情如下图所示：



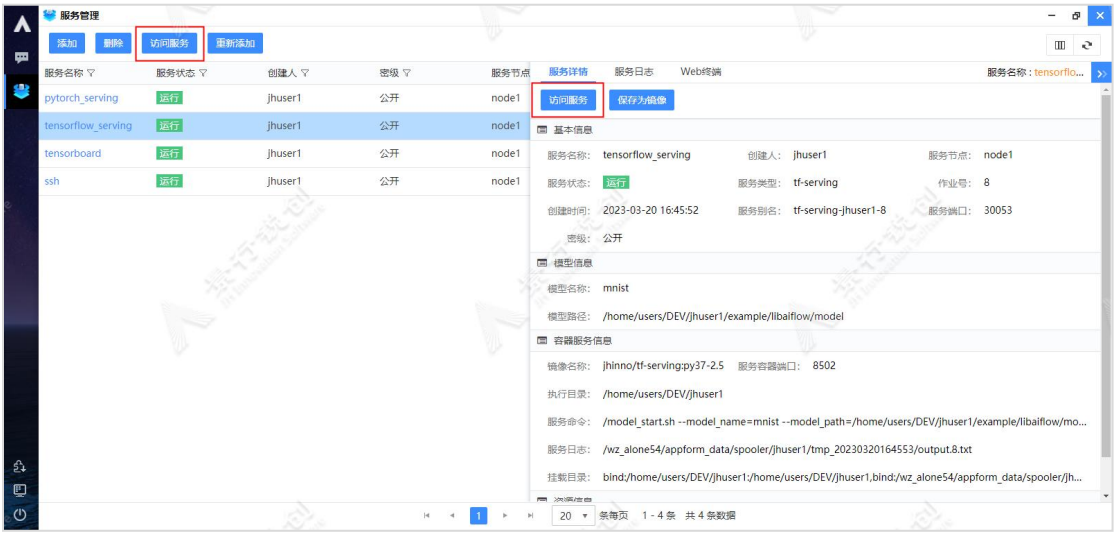
Web 终端：点击“web 终端”按钮，切换至 Web 终端页面，可以在页面模拟 ssh 终端访问操作容器内系统，如下图所示：



2.7.4. 应用服务

2.7.4.1. 访问服务

点击选中服务列表中的某一条服务实例后，点击“访问服务”按钮，会在一个新的窗口打开服务地址，如下图所示：

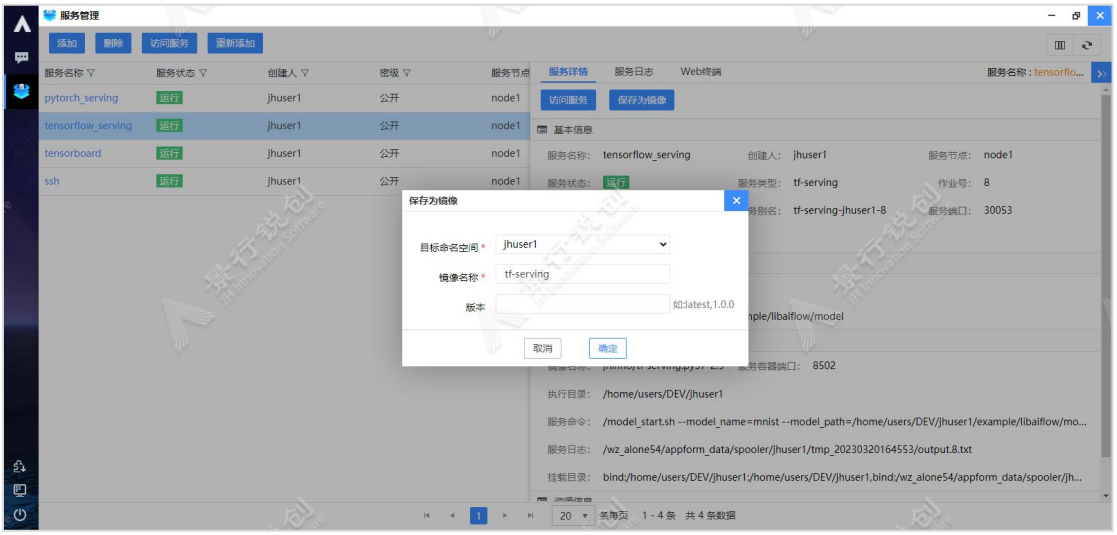


注意：访问服务功能仅在服务实例的状态为“运行”状态时可用。


```
{
 "model_version_status": [
  {
   "version": "1",
   "state": "AVAILABLE",
   "status": {
    "error_code": "OK",
    "error_message": ""
   }
  }
 ]
}
```


2.7.4.2. 保存为镜像

点击服务列表中的某一条服务实例的服务名称或者双击某一条服务实例，会打开右侧滑块。点击“保存为镜像”按钮，弹出“保存为镜像”窗口，如下图所示：



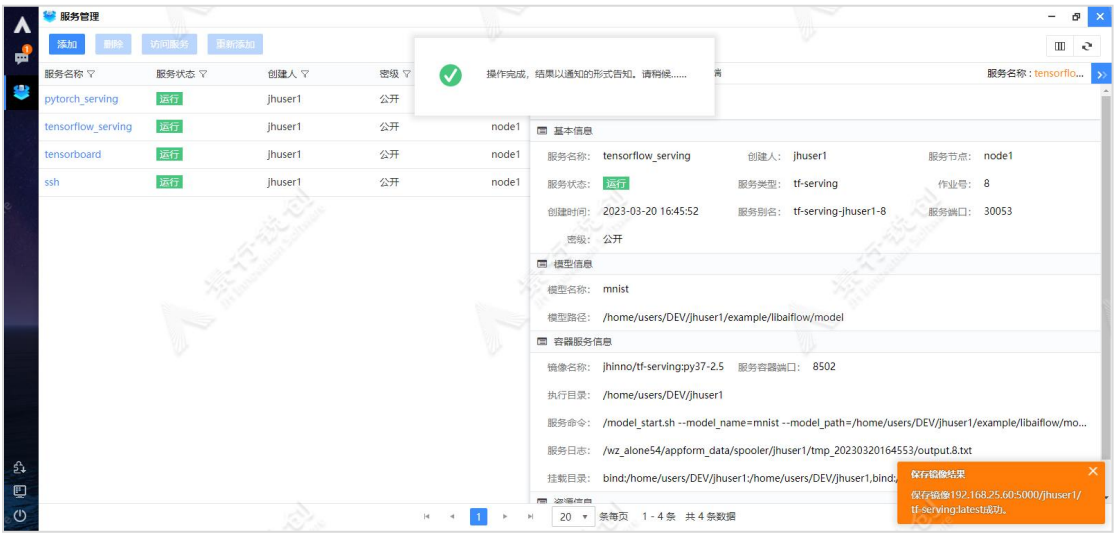
图中每个参数的具体含义如下：

目标命名空间：选择镜像的保存位置。

镜像名称：输入保存后的镜像名称。

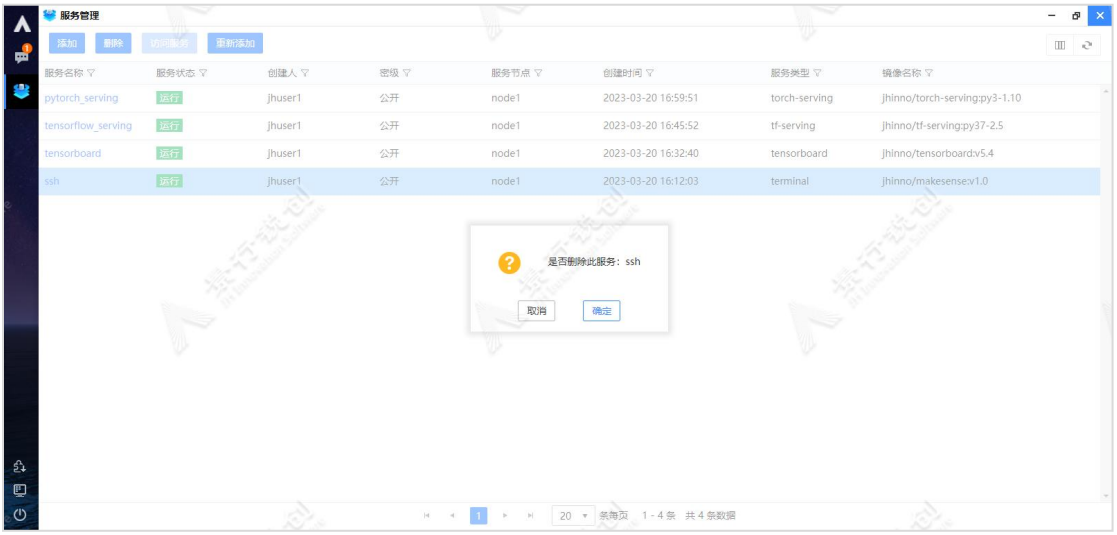
版本：输入镜像版本，默认为 latest。

点击“确定”按钮，将服务保存为镜像，如下图所示：

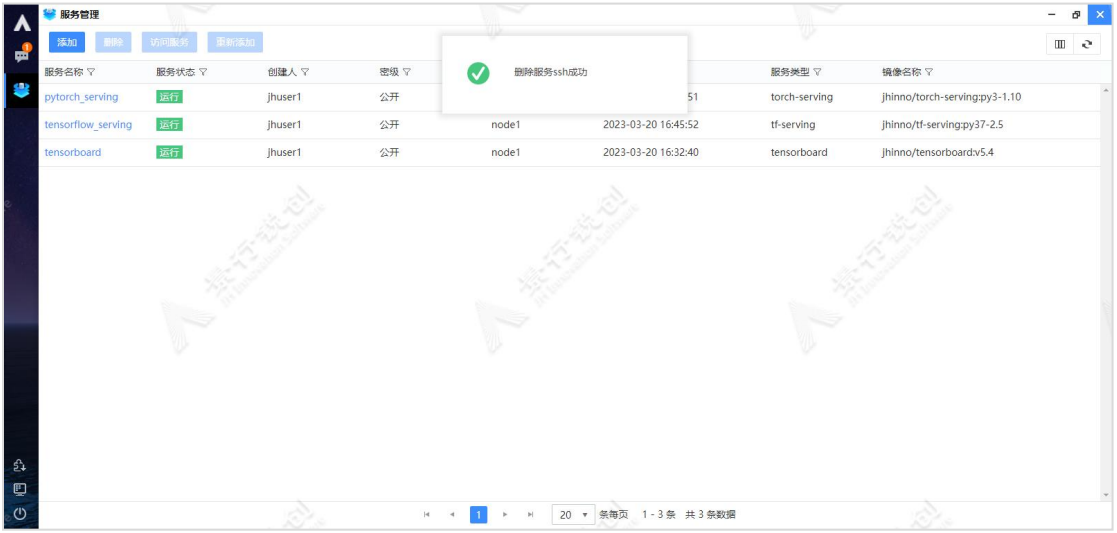


2.7.5. 删除服务

选中服务列表中某一条服务实例后，点击“删除”按钮，弹出“是否删除此服务”提示窗口，如下图所示：



点击“确定”按钮，删除该条服务实例，如下图所示：



附录一：数据加载处理组件属性说明

（一）数据提取

- Fluent 数据提取
  - Fluent 数据提取：Fluent 数据提取。
  - 选择规则文件：选择 rul 格式的数据文件。
  - 待提取数据：选择待提取数据。

- CFDpp 数据提取
  - 选择规则文件：选择 rul 格式的数据文件。
  - 待提取数据：选择待提取数据。

- 导入到 csv 数据集
  - 导入到 csv 数据集：导入到 csv 数据集。
  - 生成数据集的名称：生成数据集名称。

（二）数据加载

- csv 文件
  - csv 文件：csvImport，csv 格式的数据导入。
  - 样本特征：选择样本特征作为模型输入。
  - 选择数据文件：选择 csv 格式的数据文件。

- 图片文件夹
  - 图片文件夹：图像数据导入。
  - 样本特征：选择样本特征作为模型输入。

- 图像文件：选择图像文件夹，图像需保存在同一个文件夹中。

- 数值数据数据集
  - 数值数据数据集：数值数据数据集。
  - 选择数据集：选择数值数据数据集。
  - 样本特征：选择样本特征作为模型输入。

- 图像数据集
  - 图像数据集：图像数据集。
  - 选择数据集：选择图像数据集。
  - 样本特征：选择样本特征作为模型输入。

- Tfrecord 数据集
  - Tfrecord 数据集：Tfrecord 数据集。
  - 选择数据集：选择 Tfrecord 数据集。
  - 样本特征：选择样本特征作为模型输入。
  - 自动获取样本数：布尔值，指定是否遍历 tfrecord 文件来获取样本个数，并写入 meta.json 文件中，如果 meta.json 中已经存在样本数记录则忽略此项。

- 自定义读取数据
  - 自定义读取数据：自定义读取数据。
  - 选择数据文件：选择 csv 格式的数据文件。
  - 自定义函数名称：填写自定义读取数据的函数名称。
  - train_size：训练集大小，当该组件直连模型训练时，参数生效，默认为 100。
  - val_size：验证集大小，当该组件直连模型训练时，参数生效，默认为 100。
  - 自定义读取数据脚本：按照模板添加自定义读取数据脚本。

（三）数据预处理

- 选择特征
  - 选择特征：选择样本特征和标签特征。
  - 样本特征：选择样本特征作为模型输入。
  - 样本标签：选择样本标签作为模型输出。

**********FeatureInput**********

- 输入特征
  - 输入特征：输入样本特征和标签特征。
  - 样本特征：样本特征列作为模型输入，如[1,2,3]选择第二列，第三列和第四列或者[1-4]选择第二列到第四列或者[1]表示选择第二列，当参数为 all 时，表示选择所有列，默认 all。
  - 标签特征：样本标签列作为模型标签输入，如[1,2,3]选择第二列，第三列和第四列或者[1:4]选择第二列到第四列或者[1]表示选择第二列，当参数为 all 时，表示选择所有列，None 表示不选，默认 None。

- 数据分割
  - 数据分割：实现训练集、测试集和验证集的随机分割。
  - 训练/验证集/测试：数据集划分，按所选比例划分为训练集、测试集和验证集。

- 保存数据
  - 保存数据：保存生成的数据。
  - 生成数据集的名称：生成数据集名称。
  - 数据集类型：保存数据集的类型。

- 自定义处理数据
  - 自定义数据处理：自定义处理数据。
  - 自定义处理数据脚本：按照模板添加自定义处理数据脚本。
  - 自定义函数名称：填写自定义处理数据的函数名称。

1．数值数据集预处理

- 丢弃缺失值
  - 丢弃缺失值：丢弃缺失值所在的行或者列。
  - column index：输入行列表，如[0,1]，默认为None，当axis设置为1时该参数生效，否则无效，默认为None。
  - axis：默认axis=0。0为按行删除，1为按列删除。
  - how：选择删除规则，默认any.'any'： 只要含有NA，就舍去该行/列;'all'： 只有该行/列均为NA时才舍去。
  - thresh：指定行/列具有非NA的数目，即至少具有thresh个非NA值时才进行保留。

- 填充缺失值
  - 填充缺失值：填充缺失值。
  - value：填充值。
  - method：填充缺失值的方法，当为'ffill'，表示用前面的值填充，当'bfill'表示用后面的值填充。
  - axis：非负整数，连接轴，不包括批处理轴，默认0.0 表示'index', 1表示'columns'。
  - limit：如果指定了方法，则这是连续的NaN值的前向/后向填充的最大数量。换句话说，如果连续NaN数量超过这个数字，它将只被部分填充。如果未指定方法，则这是沿着整个轴的最大数量，其中NaN将被填充。如果不是无，则必须大于0。
  - downcast：尝试向下转换为适当的相等类型的字符串。

- 删除重复数据
  - 删除重复数据：删除重复的数据。
  - keep：删除重复数据的规则。first：保留第一次出现的重复行，删除后面的重复行。last： 删除重复项，除了最后一次出现。False：删除所有重复项。

- 数据标准化
  - 数据标准化：数据标准化。
  - method：选择数据标准化方法。

- 删除数据
  - 删除数据：删除指定行或列数据。
  - labels:输入删除的行列的名字,输入列表或者 None,列表可以是[1,2]或者['a','b']。
  - axis：指定删除行或者列，0 指删除行，1 表示删除列，默认 0。
  - index：输入行列表或者行号，如 0 或者[0,1]或者 None，默认为 None，当 axis 设置为 1 时该参数生效，否则无效。
  - level：针对多重索引时的优先级，默认为 None。

- 数据排序
  - 数据排序：根据指定列数据也可根据指定行的数据排序。
  - column index：输入行列表或者行号，如 0 或者[0,1]，默认为 0，当 axis 设置为 1 时该参数生效，否则无效。
  - axis：按照行或者列排序，0 指按照指定列排序，1 指按照指定行排序，默认 0。
  - ascending：是否按指定列的数组升序排列，默认为 True，即升序排列。

- na_position：设定缺失值的显示位置。

- 数据合并
  - 数据合并：根据列索引或者索引列表对数据合并。
  - how：数据的拼接方式，inner。
  - left_on：左侧 DataFrame 中的列或索引级别用作键。可以是列名，索引级名称，也可以是长度等于 DataFrame 长度的数组。
  - right_on：右侧 DataFrame 中的列或索引级别用作键。可以是列名，索引级名称，也可以是长度等于 DataFrame 长度的数组。
  - sort：按字典顺序通过连接键对结果 DataFrame 进行排序，默认为 True。
  - suffixes：用于重叠列的字符串后缀元组。 默认为('x','y')。
  - indicator：将一列添加到名为_merge 的输出 DataFrame，其中包含有关每行源的信息。

- 特征编码
  - 特征编码：对选择的数据进行编码，分为哑编码和独热编码。
  - how：OrdinalEncoder 为哑编码，能够将分类特征转换为分类数值。OneHotEncoder 为独热编码,将每一个分类特征变量的 m 个可能的取值转变成 m 个二值特征，对于每一条数据这 m 个值中仅有一个特征值为 1，其他的都为 0；LabelEncoder，可以将数据(类别型或者数值型都可以)转换为一个整数值。

- 标签二值化
  - 标签二值化：对选择的标签数据进行二值化，分单标签和多标签。
  - how：LabelBinarizer，多类别单标签数据二值化；MultiLabelBinarizer，多类别多标签数值二值化。

- 通过复制原始数据并添加高斯分布的噪音来进行数据增强

- 数值数据增强：自定义组件别名。
- 是否包含原始数据：数据增强后是否包含原始数据，默认为 True。
- 增强倍数：数据增强的倍数。
- 标准差：高斯噪音的标准差。
- 是否根据每列范围调整：是否根据当列的数据范围调整当列的噪音的标准差，默认为 False。

2. 图像数据集预处理

（1）图像预处理

- 图像缩放
  - 图像缩放：图像缩放。
  - dsize：非负整数组成的元组，例如，shape=(32，32)表示图像缩放到 32*32，默认(32,32)。
  - fx：输入非负浮点数，沿水平轴缩放的比例因子，默认 0。
  - fy：输入非负浮点数，沿垂直轴缩放的比例因子，默认 0。
  - interpolation：设置为 resize 的插值方式，默认为双线性插值。

- 均值滤波
  - 均值滤波：均值滤波。
  - ksize：正整数组成的元组，例如，shape=(3，3)表示滤波器核大小为 3*3，默认(3,3)。
  - anchor：锚点，被平滑的那个点，如果这个点坐标是负值的话，就表示取核的中心为锚点，所以默认值 Point(-1,-1)表示这个锚点在核的中心，默认值 Point(-1,-1)。
  - borderType：用于推断图像外部像素的某种边界模式，默认值 BORDER_DEFAULT。

- 方框滤波
  - 方框滤波：方框滤波。
  - ddepth：输入整数，输出图像深度，默认-1。
  - ksize：正整数组成的元组，例如，shape=(3，3)表示滤波器核大小为 3*3，默认(3,3)。
  - anchor：锚点，被平滑的那个点,如果这个点坐标是负值的话，就表示取核的中心为锚点，所以默认值 Point(-1,-1)表示这个锚点在核的中心,默认值 Point(-1,-1)。
  - normalize：内核是否按其面积进行标准化。
  - borderType：用于推断图像外部像素的某种边界模式，默认值 BORDER_DEFAULT。

- 高斯滤波
  - 高斯滤波：高斯滤波。
  - ksize：正整数组成的元组，例如，shape=(3，3)表示滤波器核大小为 3*3，默认(3,3)。
  - sigmaX：X 方向上的高斯核标准偏差,默认值 0。
  - sigmaY：Y 方向上的高斯核标准偏差,默认值 0。
  - borderType：用于推断图像外部像素的某种边界模式，默认值 BORDER_DEFAULT。

- 中值滤波
  - 中值滤波：中值滤波。
  - ksize：线性直径大小,例如：3, 5, 7，默认 3。

- 双边滤波
  - 双边滤波：双边滤波。
  - d：过滤过程中每个像素领域的直径范围。

- sigmaColor：颜色空间过滤器的 sigma 值,浮点数。
- sigmaSpace：坐标空间中滤波器的 sigma 值,浮点数。
- borderType：用于推断图像外部像素的某种边界模式，默认值 BORDER_DEFAULT。

- 2D 卷积
  - 2D 卷积：2D 卷积。
  - ddepth：输入整数，输出图像深度，默认-1。
  - kernel：正整数组成的元组，例如，shape=(3，3)表示滤波器核大小为 3*3，默认(3,3)。
  - anchor：锚点，被平滑的那个点,如果这个点坐标是负值的话，就表示取核的中心为锚点，所以默认值 Point(-1,-1)表示这个锚点在核的中心,默认值 Point(-1,-1)。
  - delta：输入浮点数，选填,滤波后的像素添加固定参数，默认 0。
  - borderType：用于推断图像外部像素的某种边界模式，默认值 BORDER_DEFAULT。

- 图像裁剪
  - 图像裁剪：图像裁剪。
  - rect：输入字符串，字符串由','分开，分开后的字符串格式应为 a-b 型，a 与 b 均为非负整数，例如，'0-32,0-32'表示裁剪区域为[0-32,0-32]也就是[y0-y1,x0-x1]。

- 全局二值化
  - 全局二值化：全局二值化。
  - thresh：输入浮点数，指用来对像素值进行分类的阈值，默认 0。
  - maxval：输入浮点数，指当像素值高于（有时是小于）阈值时应该被赋予的新的像素值，默认 0。
  - thresholdType：设置为阈值方法，默认为 cv.INTER_NEAREST(线性

插值)。

- 局部二值化
  - 局部二值化：局部二值化。
  - maxValue：输入浮点数，阈值的最大值，默认 0。
  - adaptiveMethod：在一个邻域内计算阈值所采用的算法，有两个取值，分别为 ADAPTIVE_THRESH_MEAN_C 和 ADAPTIVE_THRESH_GAUSSIAN_C。
  - thresholdType：设置为阈值方法，默认为 cv.INTER_NEAREST(线性插值)。
  - blockSize：adaptiveThreshold 的计算单位是像素的邻域块，这是局部邻域大小，3、5、7 等，非负整数，默认 3。
  - C：一个偏移值调整量，用均值和高斯计算阈值后，再减或加这个值就是最终阈值，浮点数，默认 0。

- 颜色空间转换
  - 颜色空间转换：图像颜色空间转换。
  - code：指定颜色空间转换类型，默认为 cv2.COLOR_BGR2RGB。

- 图像颜色空间分离
  - 图像颜色空间分离：图像颜色空间分离。

（2）分类图像数据增强

- 文件名标注数据集
  - 文件名标注数据集：文件名标注数据集。
  - featurewise_center：如果为 False，将输入数据的均值设置为 0，逐特征进行，默认'False'。
  - samplewise_center：如果为 False，将每个样本的均值设置为 0，

逐特征进行，默认'False'。

- featurewise_std_normalization：如果为False，将输入除以数据标准差，逐特征进行，默认'False'。
- samplewise_std_normalization：如果为False，将每个输入除以其标准差，逐特征进行，默认'False'。
- zca_epsilon：ZCA 白化的 epsilon 值，默认为 1e-6。
- zca_whitening：是否应用 ZCA 白化，默认'False'。
- rotation_range：整数，随机旋转的度数范围，默认 0。
- width_shift_range：整数，随机旋转的度数范围，默认 0;float：如果 <1，则是除以总宽度的值，或者如果 >=1，则为像素值;1-D 数组：数组中的随机元素;int： 来自间隔 (-width_shift_range, +width_shift_range) 之间的整数个像素;width_shift_range=2 时，可能值是整数 [-1, 0, +1]，与 width_shift_range=[-1, 0, +1] 相同；而 width_shift_range=1.0 时，可能值是 [-1.0, +1.0) 之间的浮点数。
- height_shift_range：整数，随机旋转的度数范围，默认 0;float：如果 <1，则是除以总宽度的值，或者如果 >=1，则为像素值；1-D 数组： 数组中的随机元素；int： 来自间隔 (-width_shift_range, +width_shift_range) 之间的整数个像素；width_shift_range=2 时，可能值是整数 [-1, 0, +1]，与 width_shift_range=[-1, 0, +1] 相同；而 width_shift_range=1.0 时，可能值是 [-1.0, +1.0) 之间的浮点数。
- shear_range：浮点数。剪切强度（以弧度逆时针方向剪切角度），默认 0.0。
- zoom_range：浮点数 或 [lower, upper]。随机缩放范围。如果是浮点数，[lower, upper] = [1-zoom_range, 1+zoom_range]，默认 0.0。
- channel_shift_range：浮点数。随机通道转换的范围，默认 0.0。
- fill_mode：'constant', 'nearest', 'reflect' or 'wrap' 之一。

默认为 'nearest'。输入边界以外的点根据给定的模式填充。

- cval：浮点数或整数。用于边界之外的点的值，当 fill_mode ='constant' 时，默认 0.0。
- horizontal_flip：是否随机水平翻转，默认'False'。
- vertical_flip：是否随机垂直翻转，默认'False'。
- rescale：如果是 None 或 0，不进行缩放，否则将数据乘以所提供的值（在应用任何其他转换之前），默认 None。
- preprocessing_function：应用于每个输入的函数。这个函数会在任何其他改变之前运行。这个函数需要一个参数：一张图像（秩为 3 的 Numpy 张量），并且应该输出一个同尺寸的 Numpy 张量，默认 None。
- Data Format：选择数据格式，定义输入数据的维度顺序，channels_last 对应输入形状为(batch, height, weight, ..., channels)，channels_first 对应输入形状为（batch, channels, height, weight, ...），默认 channels_last。
- validation_split:浮点数。Float. 保留用于验证的图像的比例(严格在 0 和 1 之间），默认 0.0。
- dtype：生成数组使用的数据类型，默认 None。
- batch_size：正整数，默认 32。
- shuffle：是否随机打乱数据，默认为 True。
- seed：整数或者 None，默认 None。
- save_to_dir：None 或 字符串（默认 None）。这使你可以最佳地指定正在生成的增强图片要保存的目录，默认 None。
- save_prefix：字符串。 保存图片的文件名前缀（仅当 save_to_dir 设置时可用）。
- save_format：'png'或'jpeg'之一，指定保存图片的数据格式，默认'jpeg'。

- 文件标注数据集

- 文件标注数据集：文件标注数据集，输入 dataframe 和目录的路径，并生成批量的增强/标准化的数据。
- featurewise_center：如果为 False，将输入数据的均值设置为 0，逐特征进行，默认'False'。
- samplewise_center：如果为 False，将每个样本的均值设置为 0，逐特征进行，默认'False'。
- featurewise_std_normalization：如果为 False，将输入除以数据标准差，逐特征进行，默认'False'。
- samplewise_std_normalization：如果为 False，将每个输入除以其标准差，逐特征进行，默认'False'。
- zca_epsilon：ZCA 白化的 epsilon 值，默认为 1e-6。
- zca_whitening：是否应用 ZCA 白化，默认'False'。
- rotation_range：整数，随机旋转的度数范围，默认 0。
- width_shift_range： 整数，随机旋转的度数范围，默认 0;float：如果 <1，则是除以总宽度的值，或者如果 >=1，则为像素值；1-D 数组： 数组中的随机元素；int： 来自间隔 (-width_shift_range, +width_shift_range) 之间的整数个像素；width_shift_range=2 时，可能值是整数 [-1, 0, +1]，与 width_shift_range=[-1, 0, +1] 相同；而 width_shift_range=1.0 时，可能值是 [-1.0, +1.0) 之间的浮点数。
- height_shift_range： 整数，随机旋转的度数范围，默认 0;float：如果 <1，则是除以总宽度的值，或者如果 >=1，则为像素值；1-D 数组： 数组中的随机元素；int： 来自间隔 (-width_shift_range, +width_shift_range) 之间的整数个像素；width_shift_range=2 时，可能值是整数 [-1, 0, +1]，与 width_shift_range=[-1, 0, +1] 相同；而 width_shift_range=1.0 时，可能值是 [-1.0, +1.0) 之间的浮点数。
- shear_range：浮点数。剪切强度（以弧度逆时针方向剪切角度），默认 0.0。

- zoom_range：浮点数 或 [lower, upper]。随机缩放范围。如果是浮点数，[lower, upper] = [1-zoom_range, 1+zoom_range]，默认 0.0。
- channel_shift_range：浮点数。随机通道转换的范围，默认 0.0。
- fill_mode：'constant', 'nearest', 'reflect' or 'wrap' 之一。默认为 'nearest'。输入边界以外的点根据给定的模式填充。
- cval：浮点数或整数。用于边界之外的点的值，当 fill_mode ='constant' 时，默认 0.0。
- horizontal_flip：是否随机水平翻转，默认'False'。
- vertical_flip：是否随机垂直翻转，默认'False'。
- rescale：如果是 None 或 0，不进行缩放，否则将数据乘以所提供的值（在应用任何其他转换之前），默认 None。
- preprocessing_function：应用于每个输入的函数。这个函数会在任何其他改变之前运行。这个函数需要一个参数：一张图像（秩为 3 的 Numpy 张量），并且应该输出一个同尺寸的 Numpy 张量，默认 None。
- Data Format：选择数据格式，定义输入数据的维度顺序，channels_last 对应输入形状为(batch, height, weight, ..., channels)，channels_first 对应输入形状为（batch, channels, height, weight, ...），默认 channels_last。
- validation_split:浮点数。Float. 保留用于验证的图像的比例(严格在 0 和 1 之间），默认 0.0。
- dtype：生成数组使用的数据类型，默认 None。
- dtype：数据所在目录，生成脚本中自动回填以组件名称命名的文件夹，默认 CsvLabel。
- y_col：字符串或字符串列表，dataframe 中将作为目标数据的列，默认'class'。
- weight_col：dataframe 中包含样本权重的列。默认 None。
- target_size：整数元组 (height, width)，所有找到的图都会调整

到这个维度，默认为 (256, 256)。

- color_mode：'grayscale', 'rbg' 之一，图像是否转换为 1 个或 3 个颜色通道，默认：'rgb'。
- classes：可选的类别列表 (例如，['dogs', 'cats'])。默认：None。 如未提供，类比列表将自动从 y_col 中推理出来，y_col 将会被映射为类别索引）。包含从类名到类索引的映射的字典可以通过属性 class_indices 获得。
- class_mode： 'categorical', 'binary', 'sparse', 'input', 'other' or None 之一。 默认：'categorical'。决定返回标签数组的类型：'categorical' 将是 2D one-hot 编码标签；'binary'将是 1D 二进制标签；'sparse'将是 1D 整数标签；'input' 将是与输入图像相同的图像（主要用于与自动编码器一起使用）;'other'将是 y_col 数据的 numpy 数组；None, 不返回任何标签。
- batch_size：整数，默认 32。
- shuffle：是否随机打乱数据，默认为 True。
- seed：整数或者 None，可选的混洗和转换的随机种子，默认 None。
- save_to_dir：None 或 字符串（默认 None）。这使你可以最佳地指定正在生成的增强图片要保存的目录，默认 None。
- save_prefix：字符串。 保存图片的文件名前缀（仅当 save_to_dir 设置时可用）。
- save_format：'png'或'jpeg'之一，指定保存图片的数据格式，默认'jpeg'。
- subset：如果在 ImageDataGenerator 中设置了 validation_split，则为 training 或者 validation 的子集，默认 None。
- interpolation：在目标大小与加载图像的大小不同时，用于重新采样图像的插值方法，支持的方法有'nearest', 'bilinear', and 'bicubic'，默认情况下，使用 'nearest'。

- 文件夹名标注数据集

- 文件夹名标注数据集：文件夹名标注数据集，每个类应该包含一个子目录。任何在子目录树下的 PNG, JPG, BMP, PPM 或 TIF 图像，都将被包含在生成器中。
- featurewise_center：如果为 False，将输入数据的均值设置为 0，逐特征进行，默认'False'。
- samplewise_center：如果为 False，将每个样本的均值设置为 0，逐特征进行，默认'False'。
- featurewise_std_normalization：如果为 False，将输入除以数据标准差，逐特征进行，默认'False'。
- samplewise_std_normalization：如果为 False，将每个输入除以其标准差，逐特征进行，默认'False'。
- zca_epsilon：ZCA 白化的 epsilon 值，默认为 1e-6。
- zca_whitening：是否应用 ZCA 白化，默认'False'。
- rotation_range：整数，随机旋转的度数范围，默认 0。
- width_shift_range： 整数，随机旋转的度数范围，默认 0;float：如果 <1，则是除以总宽度的值，或者如果 >=1，则为像素值；1-D 数组： 数组中的随机元素；int： 来自间隔 (-width_shift_range, +width_shift_range) 之间的整数个像素；width_shift_range=2 时，可能值是整数 [-1, 0, +1]，与 width_shift_range=[-1, 0, +1] 相同；而 width_shift_range=1.0 时，可能值是 [-1.0, +1.0) 之间的浮点数。
- height_shift_range： 整数，随机旋转的度数范围，默认 0;float：如果 <1，则是除以总宽度的值，或者如果 >=1，则为像素值；1-D 数组： 数组中的随机元素；int： 来自间隔 (-width_shift_range, +width_shift_range) 之间的整数个像素；width_shift_range=2 时，可能值是整数 [-1, 0, +1]，与 width_shift_range=[-1, 0, +1] 相同；而 width_shift_range=1.0 时，可能值是 [-1.0, +1.0) 之间的浮点数。
- shear_range：浮点数。剪切强度（以弧度逆时针方向剪切角度），

默认 0.0。

- zoom_range：浮点数 或 [lower, upper]。随机缩放范围。如果是浮点数，[lower, upper] = [1-zoom_range, 1+zoom_range]，默认 0.0。
- channel_shift_range：浮点数。随机通道转换的范围，默认 0.0。
- fill_mode：'constant', 'nearest', 'reflect' or 'wrap' 之一。默认为 'nearest'。输入边界以外的点根据给定的模式填充。
- cval：浮点数或整数。用于边界之外的点的值，当 fill_mode ='constant' 时，默认 0.0。
- horizontal_flip：是否随机水平翻转，默认'False'。
- vertical_flip：是否随机垂直翻转，默认'False'。
- rescale：如果是 None 或 0，不进行缩放，否则将数据乘以所提供的值（在应用任何其他转换之前），默认 None。
- preprocessing_function：应用于每个输入的函数。这个函数会在任何其他改变之前运行。这个函数需要一个参数：一张图像（秩为 3 的 Numpy 张量），并且应该输出一个同尺寸的 Numpy 张量，默认 None。
- Data Format：选择数据格式，定义输入数据的维度顺序，channels_last 对应输入形状为(batch, height, weight, ..., channels)，channels_first 对应输入形状为（batch, channels, height, weight, ...），默认 channels_last。
- validation_split:浮点数。Float. 保留用于验证的图像的比例(严格在 0 和 1 之间），默认 0.0。
- dtype：生成数组使用的数据类型，默认 None。
- target_size：整数元组 (height, width)，所有找到的图都会调整到这个维度，默认为 (256, 256)。
- color_mode：'grayscale', 'rbg' 之一，图像是否转换为 1 个或 3 个颜色通道，默认：'rgb'。
- classes：可选的类别列表 (例如， ['dogs', 'cats'])。默认：No

ne。 如未提供，类比列表将自动从 y_col 中推理出来，y_col 将会被映射为类别索引）。 包含从类名到类索引的映射的字典可以通过属性 class_indices 获得。

- class_mode：'categorical', 'binary', 'sparse', 'input', 'other' or None 之一。 默认：'categorical'。决定返回标签数组的类型：'categorical' 将是 2D one-hot 编码标签；'binary'将是 1D 二进制标签；'sparse'将是 1D 整数标签；'input' 将是与输入图像相同的图像（主要用于与自动编码器一起使用）;'other'将是 y_col 数据的 numpy 数组；None, 不返回任何标签。
- batch_size：非负整数, 一批数据的大小，默认 32。
- shuffle：是否随机打乱数据，默认为 True。
- seed：整数或者 None，可选的混洗和转换的随机种子，默认 None;
- save_to_dir：None 或 字符串（默认 None）。这使你可以最佳地指定正在生成的增强图片要保存的目录，默认 None。
- save_prefix：字符串。 保存图片的文件名前缀（仅当 save_to_dir 设置时可用）。
- save_format：'png'或'jpeg'之一，指定保存图片的数据格式，默认'png'。
- subset：如果在 ImageDataGenerator 中设置了 validation_split，则为 training 或者 validation 的子集，默认 None。
- interpolation：在目标大小与加载图像的大小不同时，用于重新采样图像的插值方法，支持的方法有'nearest', 'bilinear', and 'bicubic'，默认情况下，使用 'nearest'。

（3）目标检测数据集增强

- 随机反转图像
  - 随机反转图像：在水平方向上随机反转图像，使用概率 p 对图像数据进行反转。

- p：输入浮点数，图像翻转概率，默认 0.5。

- 随机缩放图像
  - 随机缩放图像：在保持纵横比不变的情况下，缩放图像。丢弃变换图像中剩余面积小于 25% 的边界框。 分辨率保持不变，用黑色填充剩余区域。
  - scale：图像缩放因子，从范围（1-scale，1+scale）中随机抽取图像的缩放因子，默认 0.2。

- 随机平移图像
  - 随机平移图像：随机平移图像，删除变换图像中剩余区域中面积小于 25%的边界框，保持分辨率不变，如果存在剩余区域则用黑色填充。
  - translate：输入 0 到 1 之间的整数或者浮点数，图像从范围（1-translate，1+translate）中随机抽取的因子进行平移，默认 0.2。

- 随机旋转图像
  - 随机旋转图像：随机旋转图像。
  - angle：输入正整数，图像旋转范围从（-angle，angle）中随机抽取的因子进行旋转，默认 10。

- 随机裁剪图像
  - 随机裁剪图像：在水平方向上，随机裁剪图像。
  - shear_factor：输入 0 到 1 之间的整数或者浮点数，定义随机水平裁剪图像因子，默认 0.2。

附录二：模型设计组件属性说明

- 自定义模型
  - 自定义模型：自定义模型结构。
  - 自定义模型脚本：按照模板添加自定义模型脚本。

- 模型训练
  - 模型训练：开始训练模型，包括深度学习组件拖拽生成的模型，自定义模型和深度学习模型增量训练。
  - 模型名称：保存模型的名称。
  - 3D 可视化：是否开启 3D 可视化，3D 可视化仅支持深度学习模型且为 Sequential 式模型，其他类型的模型暂不支持。
  - 迭代次数：训练模型的迭代次数，默认为 100。
  - 批次大小：一次训练所选取的样本数。
  - 损失函数：模型训练的损失函数，暂时不支持 Reduction 函数。
  - metrics：metrics 函数，暂时不支持 MeanRelativeError，MeanIoU，PrecisionAtRecall，RecallAtPrecision，SensitivityAtSpecificity 和 SpecificityAtSensitivity。
  - 优化器：设置模型优化器。
  - 学习率调整策略：设置学习率调整策略：None 表示不设置任何学习策略，学习率为固定值；CosineDecay，余弦衰减策略，与该策略相关的参数有：initial_learning_rate(初始学习率),decay_steps(衰减步数),alpha(最小学习率值作为 initial_learning_rate 的一部分);CosineDecayRestarts，包含重新启动的余弦衰减策略，与该策略相关的参数有：initial_learning_rate(初始学习率),first_decay_steps(要衰减的步数),t_mul(用于导出 i-th 周期内的迭代次数),m_mul(用于导出 i-th 周期的初始学习率),alpha(最小学习率值作为 initial_learning_rate 的一部分);ExponentialDecay，指数

衰减策略，与该策略相关的参数有：initial_learning_rate(初始学习率),decay_steps(衰减步数),decay_rate(衰减率),staircase(如果 True 以离散间隔衰减学习率);InverseTimeDecay，反时间衰减策略，与该策略相关的参数有：initial_learning_rate(初始学习率),decay_steps(衰减步数),decay_rate(衰减率),staircase(是否在离散梯度中使用衰减，而不是在连续梯度);PiecewiseConstantDecay，分段常熟衰减策略，与该策略相关的参数有：boundaries(Tensors 或 ints 或 floats 的列表具有严格增加的条目，并且所有元素具有与优化器步骤相同的类型),values(Tensor 或 float 或 int 列表，指定由 boundaries 定义的间隔的值，它应该比 boundaries 多一个元素，并且所有元素都应该具有相同的类型);PolynomialDecay，多项式学习能力衰减策略，与该策略相关的参数有：initial_learning_rate(初始学习率),decay_steps(衰减步数),end_learning_rate(最小的最终学习率),power(多项式的幂),cycle(是否应该循环超过 decay_steps)。

- learning_rate：设置固定学习率，0 到 1 之间的浮点数，默认 0.001。
- initial_learning_rate：初始学习率，0 到 1 之间的浮点数，默认 0.1。
- decay_steps：衰减步数，正整数，默认 1。
- alpha：最小学习率值作为 initial_learning_rate 的一部分，0 到 1 之间的浮点数，默认 0.0。
- first_decay_steps：衰减步数，正整数，默认 1。
- t_mul：用于导出 i-th 周期内的迭代次数，非负浮点数，默认 2.0。
- m_mul：用于导出 i-th 周期的初始学习率，非负浮点数，默认 1.0。
- decay_rate：衰减率，非负浮点数，默认 0.5。
- staircase：是否在离散梯度中使用衰减，而不是在连续梯度，默认'False'。
- boundaries：严格递增的由浮点型或者整型组成的列表，默认[1000

00,110000]。

- values：指定由 boundaries 定义的间隔值，它应该比 boundaries 多一个元素，并且所有元素都应该具有相同的类型，默认[1.0,0.5,0.1]。
- end_learning_rate：最小的最终学习率，非负浮点数，默认 0.5。
- power：多项式的幂，非负浮点数，默认 1.0。
- cycle：是否应该循环超过 decay_steps，默认'False'。
- rho：Adadelta 梯度平方移动均值的衰减率，非负浮点数，默认 0.95。
- epsilon：epsilon，防止除 0 错误，0 到 1 之间的浮点数或者科学计数法，默认 1e-07。
- initial_accumulator_value：初始梯度累加和，非负浮点数，默认 0.1。
- beta_1：0 到 1 之间，非常接近 1 的浮点数，默认 0.9。
- beta_2：0 到 1 之间，非常接近 1 的浮点数，默认 0.999。
- amsgrad：是否应用此算法的 AMSGrad 变种，默认'False'。
- centered：如果 True，则通过梯度的估计方差对梯度进行归一化；如果为 False，则通过非居中的第二时刻。将此设置为 True 可能有助于训练，但在计算和内存方面稍微贵一些，默认'False'。
- momentum：用于加速 SGD 在相关方向上前进，并抑制震荡，非负浮点数，默认 0.0。
- nesterov：是否使用 Nesterov 动量，默认'False'。
- beta_2：浮点值，必须小于或等于零，控制在训练期间学习率如何降低，使用零表示固定的学习率，默认-0.5。
- l1_regularization_strength：浮点值，必须大于或等于零。默认为 0.0。
- l2_regularization_strength：浮点值，必须大于或等于零。默认为 0.0。
- l2_shrinkage_regularization_strength：浮点值，必须大于或等

于零。这与上面的 L2 不同，因为上面的 L2 是一个稳定惩罚，而这个 L2 收缩是一个幅度惩罚。当输入稀疏时，收缩只会发生在活动权重上。默认为 0.0。

- beta：浮点值，beta 值。默认为 0.0。

（一）深度学习

- Sequential
  - Sequential：指定模型框架，只能创建顺序结构的模型。

- Input
  - Input：指定模型框架，利用函数 API 创建模型，可以创建复杂模型，Input 用于实例化 keras 张量。
  - shape：正整数组成的元组或 None，例如，shape=(32，)表示预期的输入将是一批 32 维向量；“None”表示形状未知的维度，默认 None。
  - batch_size：正整数，可选的静态批处理大小，默认 None。
  - name：字符串，层的可选名称。在模型中应该是唯一的(不要重复使用相同的名称)。如果没有提供，它将自动生成，默认 None。
  - dtype：输入所期望的数据类型，默认 None。
  - sparse：布尔值，指定要创建的占位符是否稀疏，默认 False。
  - tensor：可选的将现有张量封装到输入层中。如果设置，该层将不会创建占位张量。默认 None。
  - ragged：布尔值，指定要创建的占位符是否不规则，默认 False。

- Dense
  - Dense：Dense 层，又称全连接层，作为第一层时需要指定输入形状，以对接输入数据的形状。
  - Units：正整数，定义该层神经元数，即输出维度，如 32。

- Activation：选择激活函数进行非线性转换，如'relu'。若选择'None'，默认使用'linear'。
- Input Shape：正整数组成的元组或None。当Dense作为模型第一层时，需要指定输入维度，如(12,)表示输入维度为12，非第一层时，输入None，默认为None。
- Use Bias：选择是否使用偏置向量，默认'True'，即包含偏置。
- Kernel Initializer：选择权值初始化方法，默认'glorot_uniform'。
- Bias Initializer：选择偏置向量的初始化方法，默认'zeros'。
- Kernel Regularizer：定义权值矩阵的正则化方法，如l1(0.01)、l2(0.01)、l1_l2(l1=0.01, l2=0.01)或None，默认None。
- Bias Regularizer：定义偏置向量的正则化函数，如l1(0.01)、l2(0.01)、l1_l2(l1=0.01, l2=0.01)或None，默认None。
- Activity Regularizer：定义该层输出的正则化函数，如l1(0.01)、l2(0.01)、l1_l2(l1=0.01, l2=0.01)或None，默认None。
- Kernel Constraint：选择权值矩阵的约束（限制）函数，默认None。
- Bias Constraint：选择偏置向量的约束（限制）函数，默认None。

- Activation
  - Activation：Activation层，指定激活函数，并将激活函数应用于输出。
  - Activation：选择激活函数，默认relu。

- Dropout
  - Dropout：Dropout层，作用于输入，在训练期间每次更新时将输入单元的分数率随机设置为0，有助于防止过拟合。
  - Dropout Rate：输入0到1之间的整数或者浮点数，定义输入数据需要丢弃的比例，默认0.0。

- noise_shape：1D 整数张量， 表示将与输入相乘的二进制 dropout 掩层的形状，默认为 None。
- seed：一个作为随机种子的 Python 整数，默认为 None。

- Reshape
  - Reshape：Reshape 层，将输出重新塑造为特定的形状。
  - Target Shape：输入整数元组，将数据转换为特定的形状，默认(3, 4)。
  - Input Shape：正整数组成的元组或 None。当 Reshape 作为模型第一层时，需要指定输入维度，如（12,），表示输入维度为 12，非第一层时，输入 None，默认为 None。

- LeakyReLU
  - LeakyReLU：LeakyReLU 激活函数层，作用是给所有负值赋予一个非零斜率。
  - alpha：非负的浮点数，定义负斜率系数，默认 0.3。

- Embedding
  - Embedding：Embedding 层，又称嵌入层，将正整数(索引)转换为固定大小的稠密向量。
  - Input Dim：正整数，输入文本词汇的大小，如 1000。
  - Output Dim：正整数，指定嵌入层的维度，如 64。
  - Embeddings Initializer：选择权值初始化方法，默认'glorot_uniform'。
  - Embeddings Regularizer：定义权值矩阵的正则化方法，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01, l2=0.01)或 None，默认 None。
  - Activity Regularizer：定义该层输出的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01, l2=0.01)或 None，默认 None。
  - Embeddings Constraint：选择权值矩阵的约束（限制）函数，默认

None，暂时只支持 None。

- Mask Zero：输入值 0 是否是一个应该被屏蔽的特殊“填充”值。
- Input Length：正整数，输入序列的长度，如果要连接 Flatten 或 Dense 层则需要设置该参数为常数，如 100。

- Masking
  - Masking：Masking 层，对于输入张量的每一个时间步（张量的第一个维度），如果所有时间步中输入张量的值与 mask_value 相等，那么这个时间步将在所有下游层被覆盖(跳过)（只要它们支持覆盖）。
  - Target Shape：输入浮点数，mask value，默认 0.0。
  - Input Shape：正整数组成的元组或 None。当 Reshape 作为模型第一层时，需要指定输入维度，如（12,），表示输入维度为 12，非第一层时，输入 None，默认为 None。

- Lambda
  - Lambda：Lambda 层，将任意表达式封装为 Layer 对象。
  - function：需要封装的函数，将输入张量作为第一个参数。
  - output_shape：预期的函数输出尺寸，只在使用 Theano 时有意义，可以是元组或者函数。 如果是元组，它只指定第一个维度；样本维度假设与输入相同：output_shape = (input_shape[0],) + output_shape 或者，输入是 None 且样本维度也是 None： output_shape = (None,) + output_shape 如果是函数，它指定整个尺寸为输入尺寸的一个函数： output_shape = f(input_shape)。
  - arguments：可选的，需要传递给函数的关键字参数。
  - 函数定义：添加函数定义。

- Conv1D
  - Conv1D：Conv1D 层，一维卷积层或时间卷积层，该层创建了一个卷

积核，与输入层在单个空间(或时间)维度上进行卷积，产生一个输出张量。

- Filters：正整数，定义卷积核的数目，即输出的维度，默认 32。
- Kernel Size：整数或由单个整数构成的列表或元组，定义卷积核的空域或时域窗口的长度，默认 2。
- Strides：整数或由单个整数构成的列表或元组，定义卷积的步长，默认 1。
- Padding：选择填充的补 0 策略，valid 表示只进行有效的卷积，即对边界数据不处理；same 表示保留边界处的卷积结果，使输出形状和输入形状相同；causal 表示将产生因果（膨胀的）卷积，即 output[t]不依赖于 input[t+1:]，当对不能违反时间顺序的时序信号建模时有用；默认 valid。
- Data Format：选择数据格式，定义输入数据的维度顺序，channels_last 对应输入形状为（batch, steps, channels），channels_first 对应输入形状为（batch, channels, steps），默认 channels_last。
- Dilation Rate：整数或由单个整数构成的列表或元组，定义膨胀卷积的膨胀率，默认 1。
- Activation：选择激活函数进行非线性转换，如'relu'。若选择'None'，默认使用'linear'。
- Input Shape：整数元组或 None，定义输入数据的维度，该层作为 Sequential 层下的第一层时，需指定输入维度，例如(10,128)代表一个长为 10 的序列，序列中每个信号为 128 向量，而(None, 128)代表变长的 128 维向量序列，非第一层时输入 None，默认 None。
- Use Bias：选择是否使用偏置向量，默认'True'，即包含偏置。
- Kernel Initializer：选择权值初始化方法，默认'glorot_uniform'。
- Bias Initializer：选择偏置向量的初始化方法，默认'zeros'。
- Kernel Regularizer：定义权值矩阵的正则化方法，如 l1(0.01)、l

2(0.01)、l1_l2(l1=0.01, l2=0.01)或 None，默认 None。

- Bias Regularizer：定义偏置向量的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01, l2=0.01)或 None，默认 None。
- Activity Regularizer：定义该层输出的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01, l2=0.01)或 None，默认 None。
- Kernel Constraint：选择权值矩阵的约束（限制）函数，默认 None。
- Bias Constraint：选择偏置向量的约束（限制）函数，默认 None。

- Conv2D
  - Conv2D：Conv2D 层，二维卷积层或图像空间卷积层，该层创建了一个卷积核，与输入层进行卷积，产生一个输出张量。
  - Filters：正整数，定义卷积核的数目，即输出的维度，默认 32。
  - Kernel Size：单个整数或由 2 个整数构成的列表或元组，指定二维卷积窗口的高度和宽度，可以是单个整数，以便为所有空间维度指定相同的值，默认(2, 2)。
  - Strides：单个整数或由两个整数构成的列表或元组，指定沿高度和宽度的卷积步长。可以是单个整数，以便为所有空间维度指定相同的值，默认（1,1）。
  - Padding：选择填充的补 0 策略，valid 表示只进行有效的卷积，即对边界数据不处理；same 表示保留边界处的卷积结果，使输出形状和输入形状相同；默认 valid。
  - Data Format：选择数据格式，定义输入数据的维度顺序，channels_last 对应输入形状为（batch, height, width, channels），channels_first 对应输入形状为(batch, channels, height, width)，默认 channels_last。
  - Dilation Rate：单个整数或两个整数的元组/列表，指定用于扩展卷积的扩展速率。可以是单个整数，以便为所有空间维度指定相同的值，默认(1, 1)。

- Activation：选择激活函数进行非线性转换，如'relu'。若选择'None'，默认使用'linear'。
- Input Shape：None 或整数组成的元组，定义输入数据的维度，该层作为 Sequential 层下的第一层时，需指定输入维度，例如 input_shape = (128,128,3)代表 128*128 的彩色 RGB 图像（data_format='channels_last'），非第一层时输入 None，默认 None。
- Use Bias：选择是否使用偏置向量，默认'True'，即包含偏置。
- Kernel Initializer：选择权值初始化方法，默认'glorot_uniform'。
- Bias Initializer：选择偏置向量的初始化方法，默认'zeros'。
- Kernel Regularizer：定义权值矩阵的正则化方法，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
- Bias Regularizer：定义偏置向量的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
- Activity Regularizer：定义该层输出的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
- Kernel Constraint：选择权值矩阵的约束（限制）函数，默认 None。
- Bias Constraint：选择偏置向量的约束（限制）函数，默认 None。

- Conv2DTranspose
  - Conv2DTranspose：Conv2DTranspose 层，二维转置卷积层或去卷积层，可以使具有某些卷积输出形状的东西到具有其输入形状的东西，同时保持与上述卷积兼容的连接模式。
  - Filters：正整数，定义卷积核的数目，即输出的维度，默认 32。
  - Kernel Size：单个整数或由 2 个整数构成的列表或元组，指定二维卷积窗口的高度和宽度，可以是单个整数，以便为所有空间维度指定相同的值，默认(2,?2)。
  - Strides：单个整数或由两个整数构成的列表或元组，指定沿高度和

宽度的卷积步长。可以是单个整数，以便为所有空间维度指定相同的值，默认（1,1）。

- Padding：选择填充的补 0 策略，valid 表示只进行有效的卷积，即对边界数据不处理；same 表示保留边界处的卷积结果，使输出形状和输入形状相同；默认 valid。
- Output Padding：None 或单个整数或 2 个整数的元组/列表，指定沿着输出张量的高度和宽度的填充量。可以是单个整数，以便为所有空间维度指定相同的值。沿给定维度的输出填充量必须小于沿同一维度的步幅。默认 None，推断输出形状。
- Data Format：选择数据格式，定义输入数据的维度顺序，channels_last 对应输入形状为（batch,?height,?width,?channels），channels_first 对应输入形状为(batch,?channels,?height,?width)，默认 channels_last。
- Dilation Rate：单个整数或两个整数的元组/列表，指定用于扩展卷积的扩展速率。可以是单个整数，以便为所有空间维度指定相同的值，默认(1,?1)。
- Activation：选择激活函数进行非线性转换，如'relu'。若选择'None'，默认使用'linear'。
- Input Shape：None 或整数组成的元组，定义输入数据的维度，该层作为 Sequential 层下的第一层时，需指定输入维度，例如 input_shape = (128,128,3)代表 128*128 的彩色 RGB 图像（data_format='channels_last'），非第一层时输入 None，默认 None。
- Use Bias：选择是否使用偏置向量，默认'True'，即包含偏置。
- Kernel Initializer：选择权值初始化方法，默认'glorot_uniform'。
- Bias Initializer：选择偏置向量的初始化方法，默认'zeros'。
- Kernel Regularizer：定义权值矩阵的正则化方法，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
- Bias Regularizer：定义偏置向量的正则化函数，如 l1(0.01)、l2

(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。

- Activity Regularizer：定义该层输出的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
- Kernel Constraint：选择权值矩阵的约束（限制）函数，默认 None。
- Bias Constraint：选择偏置向量的约束（限制）函数，默认 None。

- DepthwiseConv2D
  - DepthwiseConv2D：DepthwiseConv2D 层，仅执行深度空间卷积中的第一步，分别作用于每个输入通道。
  - Kernel Size：整数或由 2 个整数构成的列表或元组，指定二维卷积窗口的高度和宽度，可以是单个整数，以便为所有空间维度指定相同的值，默认(2, 2)。
  - Strides：单个整数或由两个整数构成的列表或元组，指定沿高度和宽度的卷积步长，该 keras 版本仅支持元组数值一致。可以是单个整数，以便为所有空间维度指定相同的值，默认（1,1）。
  - Padding：选择填充的补 0 策略，valid 表示只进行有效的卷积，即对边界数据不处理；same 表示保留边界处的卷积结果，使输出形状和输入形状相同；默认 valid。
  - Data Format：选择数据格式，定义输入数据的维度顺序，channels_last 对应输入形状为（batch, height, width, channels），channels_first 对应输入形状为(batch, channels, height, width)，默认 channels_last。
  - depth_multiplier：每个输入通道的深度方向卷积输出通道的数量，默认 1。
  - Activation：选择激活函数进行非线性转换，如'relu'。若选择'None'，默认使用'linear'。
  - Use Bias：选择是否使用偏置向量，默认'True'，即包含偏置。
  - depthwise Initializer：选择深度方向核矩阵的初始化方法，默认

'glorot_uniform'。

- Bias Initializer：选择偏置向量的初始化方法，默认'zeros'。
- depthwise Regularizer：定义深度方向核矩阵的正则化方法，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
- Bias Regularizer：定义偏置向量的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
- Activity Regularizer：定义该层输出的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
- depthwise Constraint：选择深度方向核矩阵的约束（限制）函数，默认 None。
- Bias Constraint：选择偏置向量的约束（限制）函数，默认 None。

- SeparableConv2D
  - SeparableConv2D：深度方向的可分离 2D 卷积，可分离的卷积的操作包括，首先执行深度方向的空间卷积 （分别作用于每个输入通道），紧接一个将所得输出通道 混合在一起的逐点卷积。
  - Filters：正整数，定义卷积核的数目，即输出的维度，默认 32。
  - Kernel Size：单个整数或由 2 个整数构成的列表或元组，指定二维卷积窗口的高度和宽度，可以是单个整数，以便为所有空间维度指定相同的值，默认(2, 2)。
  - Strides：单个整数或由两个整数构成的列表或元组，指定沿高度和宽度的卷积步长。可以是单个整数，以便为所有空间维度指定相同的值，默认（1,1）。
  - Padding：选择填充的补 0 策略，valid 表示只进行有效的卷积，即对边界数据不处理；same 表示保留边界处的卷积结果，使输出形状和输入形状相同；默认 valid。
  - Data Format：选择数据格式，定义输入数据的维度顺序，channels_last 对应输入形状为（batch, height, width, channels），channels_first 对应输入形状为(batch, channels, height, width),

默认 channels_last。

- Dilation Rate：单个整数或两个整数的元组/列表，指定用于扩展卷积的扩展速率。可以是单个整数，以便为所有空间维度指定相同的值，默认(1, 1)。
- depth_multiplier：每个输入通道的深度方向卷积输出通道的数量，默认 1。
- Activation：选择激活函数进行非线性转换，如'relu'。若选择'None'，默认使用'linear'。
- Use Bias：选择是否使用偏置向量，默认'True'，即包含偏置。
- depthwise Initializer：运用到深度方向的核矩阵的初始化器，默认'glorot_uniform'。
- Pointwise Initializer：运用到逐点核矩阵的初始化器，默认'glorot_uniform'。
- depthwise Initializer：偏置向量的初始化器，默认'glorot_uniform'。
- Depthwise Regularizer：运用到深度方向的核矩阵的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
- Bias Regularizer：运用到偏置向量的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
- Activity Regularizer：运用到层输出（它的激活值）的正则化函数 ，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
- depthwise constraint：运用到深度方向的核矩阵的约束函数，默认 None。
- Bias Constraint：选择偏置向量的约束（限制）函数，默认 None。
- Pointwise Constraint：选择偏置向量的约束（限制）函数，默认 None。

- Conv3D
  - Conv3D：Conv3D 层，三维卷积层或体积空间卷积层，该层创建了一个卷积核，与输入层进行卷积，产生一个输出张量。
  - Filters：正整数，定义卷积核的数目，即输出的维度，默认 32。
  - Kernel Size：一个整数或 3 个整数的元组/列表，指定三维卷积窗口的深度、高度和宽度。可以是单个整数，以便为所有空间维度指定相同的值，默认(2, 2, 2)。
  - Strides：一个整数或 3 个整数的元组/列表，指定卷积在每个空间维度上的步长。是否可以使用单个整数为所有空间维度指定相同的值，默认(1, 1, 1)。
  - Padding：选择填充的补 0 策略，valid 表示只进行有效的卷积，即对边界数据不处理；same 表示保留边界处的卷积结果，使输出形状和输入形状相同；默认 valid。
  - Data Format：选择数据格式，定义输入数据的维度顺序，channels_last 对应输入形状(batch, spatial_dim1, spatial_dim2, spatial_dim3, channels)，channels_first 对应输入形状(batch, channels, spatial_dim1, spatial_dim2, spatial_dim3)，默认 channels_last。
  - Dilation Rate：一个整数或 3 个整数的元组/列表，指定用于扩展卷积的扩展速率。可以是单个整数，以便为所有空间维度指定相同的值，默认(1, 1, 1)。
  - Activation：选择激活函数进行非线性转换，如'relu'。若选择'None'，默认使用'linear'。
  - Input Shape：None 或整数组成的元组，定义输入数据的维度，该层作为 Sequential 层下的第一层时，需指定输入维度，例如，input_shape=(128, 128, 128, 1)表示只有一个通道的 128x128x128 的图像数据（data_format='channels_last'），非第一层时输入 None，默认 None。
  - Use Bias：选择是否使用偏置向量，默认'True'，即包含偏置。

- Kernel Initializer：选择权值初始化方法，默认'glorot_uniform'。
- Bias Initializer：选择偏置向量的初始化方法，默认'zeros'。
- Kernel Regularizer：定义权值矩阵的正则化方法，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
- Bias Regularizer：定义偏置向量的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
- Activity Regularizer：定义该层输出的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
- Kernel Constraint：选择权值矩阵的约束（限制）函数，默认 None。
- Bias Constraint：选择偏置向量的约束（限制）函数，默认 None。

- Conv3DTranspose
  - Conv3DTranspose：Conv3DTranspose 层，三维转置卷积层或去卷积层，可以使具有某些卷积输出形状的东西到具有其输入形状的东西，同时保持与上述卷积兼容的连接模式。
  - Filters：正整数，定义卷积核的数目，即输出的维度，默认 32。
  - Kernel Size：一个整数或 3 个整数的元组/列表，指定三维卷积窗口的深度、高度和宽度。可以是单个整数，以便为所有空间维度指定相同的值，默认(2, 2, 2)。
  - Strides：一个整数或 3 个整数的元组/列表，指定卷积在每个空间维度上的步长。是否可以使用单个整数为所有空间维度指定相同的值，默认(1, 1, 1)。
  - Padding：选择填充的补 0 策略，valid 表示只进行有效的卷积，即对边界数据不处理；same 表示保留边界处的卷积结果，使输出形状和输入形状相同；默认 valid。
  - Output Padding：单个整数或 3 个整数的元组/列表，指定沿着输出张量的深度、高度和宽度的填充量。可以是单个整数，以便为所有

空间维度指定相同的值。沿给定维度的输出填充量必须小于沿同一维度的步幅。默认 None，推断输出形状。

- Data Format：选择数据格式，定义输入数据的维度顺序，channels_last 对应输入形状(batch, depth, height, width, channels)，channels_first 对应输入形状(batch, channels, depth, height, width)，默认 channels_last。
- Dilation Rate：一个整数或 3 个整数的元组/列表，指定用于扩展卷积的扩展速率。可以是单个整数，以便为所有空间维度指定相同的值，默认(1, 1, 1)。
- Activation：选择激活函数进行非线性转换，如'relu'。若选择'None'，默认使用'linear'。
- Input Shape：整数元组，定义输入数据的维度，该层作为 Sequential 层下的第一层时，需指定输入维度，例如，input_shape=(128, 128, 128, 3)表示有 3 个通道的 128x128x128 的图像数据（data_format='channels_last'），非第一层时输入 None，默认 None。
- Use Bias：选择是否使用偏置向量，默认'True'，即包含偏置。
- Kernel Initializer：选择权值初始化方法，默认'glorot_uniform'。
- Bias Initializer：选择偏置向量的初始化方法，默认'zeros'。
- Kernel Regularizer：定义权值矩阵的正则化方法，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
- Bias Regularizer：定义偏置向量的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
- Activity Regularizer：定义该层输出的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
- Kernel Constraint：选择权值矩阵的约束（限制）函数，默认 None。
- Bias Constraint：选择偏置向量的约束（限制）函数，默认 None。

- Concatenate
  - Concatenate：concatenate 层，对输入级联处理。
  - axis：选择级联方式，数据按照指定的维度进行级联，Axis 对应的值为需要级联的数据维度，如-1，0，1，默认-1。

- Add
  - Add：add 层，又称合并层，添加输入列表的层，接受一组形状相同的张量作为输入，然后返回一个形状相同的张量。

- Multiply
  - Multiply：Multiply 层，计算输入张量列表的（逐元素间的）乘积。

- Flatten
  - Flatten：Flatten 层，将输入扁平化处理，不影响批次大小。
  - Data Format：选择数据格式，定义输入数据的维度顺序，channels_last 对应输入形状为(batch, height, weight, ..., channels)，channels_first 对应输入形状为（batch, channels, height, weight, ...），默认 channels_last。

- RepeatVector
  - RepeatVector：RepeatVector 层，将输入重复 n 次。
  - Number of Repeat：整数，指定输入数据的重复次数，默认 0。

- ZeroPadding1D
  - ZeroPadding1D：ZeroPadding1D 层，一维输入的零填充层。
  - Padding：整数或 2 个整数的元组，定义填充大小，默认 1。

- ZeroPadding2D
  - ZeroPadding2D：ZeroPadding2D 层，二维输入的零填充层，可以在

图像张量的顶部、底部、左侧和右侧添加由 0 组成的行和列。

- Padding：整数或 2 个整数的元组或 2 个整数的元组，定义填充大小，默认(1, 1)。
- Data Format：选择数据格式，定义输入数据的维度顺序，channels_last 对应输入形状为(batch, spatial_dim1, spatial_dim2, spatial_dim3, channels)，channels_first 对应输入形状为(batch, channels, spatial_dim1, spatial_dim2, spatial_dim3)，默认 channels_last。

- Cropping1D
  - Cropping1D：Cropping1D 层，一维输入的裁剪层(如时间序列)，其沿着时间维度(轴 1)生长。
  - Cropping：一维输入(如时序)的裁剪层，沿着时间维度生长，输入整数或长为 2 的整数元组，指定在序列的首尾要裁剪掉多少个元素，默认（1, 1）。

- Cropping2D
  - Cropping2D：Cropping2D 层，二维输入的剪裁层（如图像），其沿着空间维度生长，即高度和宽度。
  - Cropping：二维输入(例如图片)的裁剪层输入，沿着空间维度（高度和宽度）生长。输入整数或 2 个整数的元组或 2 个整数的 2 个元组的元组，如果为整数（如 1），表示相同的对称裁剪应用于高度和宽度；如果为 2 个整数的元组（如（1,1）），表示对高度和宽度的两个不同的对称裁剪值：(symmetric_height_crop, symmetric_width_crop)；如果为 2 个整数的 2 个元组的元组：解释为?((top_crop, bottom_crop), (left_crop, right_crop))，默认((0,0),(0,0))。
  - Data Format：选择数据格式，定义输入数据的维度顺序，channels_last 对应输入形状为（batch,?height,?width,?channels），cha

nnels_first 对应输入形状为(batch,?channels,?height,?width)，默认 channels_last。

- Cropping3D
  - Cropping3D：Cropping3D 层，三维数据的剪裁层（如空间或时空）。
  - Cropping：三维数据(如空间或时空)的裁剪层，输入整数或 3 个整数的元组或 2 个整数的 3 个元组的元组，如果为整数，将对深度、宽度和高度应用相同的对称裁剪；如果为 3 个整数的元组：解释为对深度、高度和宽度的 3 个不同的对称裁剪值：(symmetric_dim1_crop, symmetric_dim2_crop, symmetric_dim3_crop)；如果为 2 个整数的 3 个元组的元组：解释为((left_dim1_crop, right_dim1_crop), (left_dim2_crop, right_dim2_crop), (left_dim3_crop, right_dim3_crop))，默认((1,1),(1,1),(1,1))。
  - Data Format：选择数据格式，定义输入数据的维度顺序，channels_last 对应输入形状为(batch, spatial_dim1, spatial_dim2, spatial_dim3, channels)，channels_first 对应输入形状为(batch, channels, spatial_dim1, spatial_dim2, spatial_dim3)，默认 channels_last。

- UpSampling1D
  - UpSampling1D：UpSampling1D 层，一维输入的上采样层，沿时间轴重复每个时间步长 n 次。
  - UpSampling Size：整数，定义上采样因子，指沿着时间轴重复每个时间步的次数，默认 2。

- UpSampling2D
  - UpSampling2D：UpSampling2D 层，二维输入的上采样层，按参数大小分别重复数据的行和列。
  - UpSampling Size：整数或 2 个整数的元组，定义行和列的上采样因

子，指沿着数据的行和列分别重复的次数，默认（2,2）。

- Data Format：选择数据格式，定义输入数据的维度顺序，channels_last 对应输入形状为（batch,?height,?width,?channels），channels_first 对应输入形状为(batch,?channels,?height,?width)，默认 channels_last。
- Interpolation：选择插值方法，nearest 或 bilinear。

- UpSampling3D
  - UpSampling3D：UpSampling3D 层，三维输入的上采样层，按参数大小分别重复数据的第 1、2、3 维。
  - UpSampling Size：整数或 3 个整数的元组，定义 3 个维度的上采样因子，指沿着数据的 3 个维度分别重复的次数，默认(2,2,2)。
  - Data Format：选择数据格式，定义输入数据的维度顺序，channels_last 对应输入形状为(batch, spatial_dim1, spatial_dim2, spatial_dim3, channels)，channels_first 对应输入形状为(batch, channels, spatial_dim1, spatial_dim2, spatial_dim3)，默认 channels_last。

- MaxPooling1D
  - MaxPooling1D：MaxPooling1D 层，一维最大池化层，对一维时序数据进行最大池化操作。
  - Pool Size：整数，定义最大池化的窗口大小，默认 2。
  - Strides：整数或 None，表示下采样因子，例如 2 表示将使得输出形状为输入的一半，None 表示默认使用 pool_size 的值，默认 None。
  - Padding：选择填充的补 0 策略，valid 表示只进行有效的卷积，即对边界数据不处理；same 表示保留边界处的卷积结果，使输出形状和输入形状相同；默认 valid。
  - Data Format：选择数据格式，定义输入数据的维度顺序，channels_last 对应输入形状为（batch, steps, features），channels_fi

rst 对应输入形状为（batch, features, steps），默认 channels_last。

- MaxPooling2D
  - MaxPooling2D：MaxPooling2D 层，二维最大池化层，对二维空间数据进行最大池化操作。
  - Pool Size：整数，或者 2 个整数表示的元组，沿(垂直，水平)方向缩小比例的因数。(2,2)会把输入张量的两个维度都缩小一半。如果只使用一个整数，那么两个维度都会使用同样的窗口长度，默认(2,2)。
  - Strides：整数或长为 2 的整数元组或 None，定义步长大小，如果为 None，表示 pool_size 的值，默认 None。
  - Padding：选择填充的补 0 策略，valid 表示只进行有效的卷积，即对边界数据不处理；same 表示保留边界处的卷积结果，使输出形状和输入形状相同；默认 valid。
  - Data Format：选择数据格式，定义输入数据的维度顺序，channels_last 对应输入形状为（batch,?height,?width,?channels），channels_first 对应输入形状为(batch,?channels,?height,?width)，默认 channels_last。

- MaxPooling3D
  - MaxPooling3D：MaxPooling3D 层，三维最大池化层，对三维数据（空间或时空）进行最大池化操作。
  - Pool Size： 3 个整数表示的元组，缩小(dim1,dim2,dim3)比例的因数。(2,2,2)会把 3D 输入张量的每个维度缩小一半，默认(2,2,2)。
  - Strides：长为 3 的整数元组或 None，定义步长大小，如果为 None，表示 pool_size 的值，默认 None。
  - Padding：选择填充的补 0 策略，valid 表示只进行有效的卷积，即对边界数据不处理；same 表示保留边界处的卷积结果，使输出形状

和输入形状相同；默认 valid。

- Data Format：选择数据格式，定义输入数据的维度顺序，channels_last 对应输入形状为（batch, spatial_dim1, spatial_dim2, spatial_dim3, channels），channels_first 对应输入形状为（batch, channels, spatial_dim1, spatial_dim2, spatial_dim3），默认 channels_last。

- AveragePooling1D
  - AveragePooling1D：AveragePooling1D 层，一维平均池化层，对时序数据进行平均池化操作。
  - Pool Size：整数，定义平均池化的窗口大小，默认 2。
  - Strides：整数或 None，表示下采样因子，例如 2 表示将使得输出 shape 为输入的一半，None 表示默认使用 pool_size 的值，默认 None。
  - Padding：选择填充的补 0 策略，valid 表示只进行有效的卷积，即对边界数据不处理；same 表示保留边界处的卷积结果，使输出形状和输入形状相同；默认 valid。
  - Data Format：选择数据格式，定义输入数据的维度顺序，channels_last 对应输入形状为（batch, steps, features），channels_first 对应输入形状为（batch, features, steps），默认 channels_last。

- AveragePooling2D
  - AveragePooling2D：AveragePooling2D 层，二维平均池化层，对二维空间数据进行平均池化操作。
  - Pool Size：整数或者 2 个整数表示的元组，沿(垂直，水平)方向缩小比例的因数。(2,2)会把输入张量的两个维度都缩小一半。如果只使用一个整数，那么两个维度都会使用同样的窗口长度，默认(2,2)。

- Strides：整数或长为 2 的整数元组或 None，定义步长大小，如果为 None，表示 pool_size 的值，默认 None。
- Padding：选择填充的补 0 策略，valid 表示只进行有效的卷积，即对边界数据不处理；same 表示保留边界处的卷积结果，使输出形状和输入形状相同；默认 valid。
- Data Format：选择数据格式，定义输入数据的维度顺序，channels_last 对应输入形状为（batch,?height,?width,?channels），channels_first 对应输入形状为(batch,?channels,?height,?width)，默认 channels_last。

- AveragePooling3D
  - AveragePooling3D：AveragePooling3D 层，三维平均池化层，对三维数据（空间或时空）进行平均池化操作。
  - Pool Size：3 个整数表示的元组，缩小(dim1,dim2,dim3)比例的因数。(2,2,2)会把 3D 输入张量的每个维度缩小一半。
  - Strides：长为 3 的整数元组或 None，定义步长大小，如果为 None，表示 pool_size 的值，默认 None。
  - Padding：选择填充的补 0 策略，valid 表示只进行有效的卷积，即对边界数据不处理；same 表示保留边界处的卷积结果，使输出形状和输入形状相同；默认 valid。
  - Data Format：选择数据格式，定义输入数据的维度顺序，channels_last 对应输入形状为（batch, spatial_dim1, spatial_dim2, spatial_dim3, channels），channels_first 对应输入形状为（batch, channels, spatial_dim1, spatial_dim2, spatial_dim3），默认 channels_last。

- GlobalAveragePooling1D
  - GlobalAveragePooling1D：GlobalAveragePooling1D 层，对于时序数据的全局平均池化。

- Data Format：选择数据格式，定义输入数据的维度顺序，channels_last 对应输入形状为（batch, steps, features），channels_first 对应输入形状为（batch, features, steps），默认 channels_last。

- GlobalAveragePooling2D
  - GlobalAveragePooling2D：GlobalAveragePooling2D 层，对于空域数据的全局平均池化。
  - Data Format：选择数据格式，定义输入数据的维度顺序，channels_last 对应输入形状为（batch, steps, features），channels_first 对应输入形状为（batch, features, steps），默认 channels_last。

- GlobalAveragePooling3D
  - GlobalAveragePooling3D：GlobalAveragePooling3D 层，对于 3D 数据的全局平均池化。
  - Data Format：选择数据格式，定义输入数据的维度顺序，channels_last 对应输入形状为（batch, steps, features），channels_first 对应输入形状为（batch, features, steps），默认 channels_last。

- GlobalMaxPooling1D
  - GlobalMaxPooling1D：GlobalMaxPooling3D 层，对于时序数据的全局最大池化。
  - Data Format：选择数据格式，定义输入数据的维度顺序，channels_last 对应输入形状为（batch, steps, features），channels_first 对应输入形状为（batch, features, steps），默认 channels_last。

- GlobalMaxPooling2D
  - GlobalMaxPooling2D：GlobalMaxPooling2D 层，对于空域数据的全局最大池化。
  - Data Format：选择数据格式，定义输入数据的维度顺序，channels_last 对应输入形状为（batch, steps, features），channels_first 对应输入形状为（batch, features, steps），默认 channels_last。

- GlobalMaxPooling3D
  - GlobalMaxPooling3D：GlobalMaxPooling3D 层，对于 3D 数据的全局最大池化。
  - Data Format：选择数据格式，定义输入数据的维度顺序，channels_last 对应输入形状为（batch, steps, features），channels_first 对应输入形状为（batch, features, steps），默认 channels_last。

- GlobalMaxPooling3D
  - GlobalMaxPooling3D：GlobalMaxPooling3D 层，对于 3D 数据的全局最大池化。
  - Data Format：选择数据格式，定义输入数据的维度顺序，channels_last 对应输入形状为（batch, steps, features），channels_first 对应输入形状为（batch, features, steps），默认 channels_last。

- BatchNormalization
  - BatchNormalization：BatchNormalization 层，对每个批处理中正则化前一步激活函数的结果。
  - Axis：整数，指定需要标准化的轴，例如，在 data_format='channels_first'的 Conv2D 层之后，在 BatchNormalization 中设置 axis

=1。默认-1。

- Momentum：浮点数，指定移动均值和移动方差的动量，默认 0.99。
- Epsilon：浮点数，为方差增加一个小的浮点数，避免被除数为零，默认 0.001。
- Center：布尔值，如果为 True，表示把偏移量加到标准化张量上，如果为 False，则忽略，默认 True。
- Scale：布尔值，如果为 True，则乘以 gamma，如果为 False，则不使用 gamma。
- Beta Initializer：指定 beta 权重的初始化方法，默认 zeros。
- Gamma Initializer：指定 gamma 权重的初始化方法，默认 ones。
- Moving Mean Initializer：指定移动平均的初始化方法，默认 zeros。
- Moving Variance Initializer：指定移动方差的初始化方法，默认 ones。
- Beta Regularizer：为 beta 权重指定正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01, l2=0.01)或 None，默认 None。
- Gamma Regularizer：为 gamma 权重指定正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01, l2=0.01)或 None，默认 None。
- Beta Constraint：指定 beta 权值的约束函数，默认 None。
- Gamma Constraint：指定 gamma 权值的约束函数，默认 None。
- renorm：是否使用批量再归一化。
- renorm_clipping：一种字典，可以将关键字“rmax”，“rmin”，“dmax”映射到用于剪裁 renorm 校正的标量张量。校正（r，d）用作 corrected_value = normalized_value * r + d，其中 r 被剪裁为[rmin，rmax]，d 被剪裁为[-dmax，dmax]。缺少的 rmax、rmin 和 dmax 分别设置为 inf、0 和 inf，默认 None。
- renorm_momentum：用 renorm 更新滑动方式和标准偏差的动量。与动量不同，会影响训练，既不应太小（会增加噪音），也不应太大（会给出过时的估计），默认 0.99。

- fused：如果是 True，使用更快的融合实现，在没有可用的快速融合实现将抛出异常。如果为 False，则不使用；如果是 None，会使用可以使用的快速融合。
- trainable：如果为 True，还将变量添加到图形集合。
- virtual_batch_size：一个 int。默认情况下，virtual_batch_size 为 None，这意味着在整个批次中执行批次规范化。当 virtual_batch_size 不是 None 时，改为执行“Ghost Batch Normalization”，创建每个单独规范化的虚拟子批（使用共享 gamma、beta 和滑动统计）。必须在执行期间划分实际批大小。
- adjustment：仅在训练期间，采用包含输入张量（动态）形状的张量并返回一对（scale、bias）以应用于标准化值（γ 和 β 之前）的函数。例如，如果 axis=-1，adjustment = lambda shape: (tf.random_uniform(shape[-1:], 0.93, 1.07)，tf.random_uniform(shape[-1:], -0.1, 0.1))将标准化值向上或向下缩放 7%，然后将结果向上滑动 0.1（每个功能都有独立的缩放和偏移，但在所有示例中都有共享），最后应用 gamma 和/或 beta。如果没有，则不应用调整。如果指定了 virtual_batch_size，则无法指定。
- name：字符串，层的名称。

- SimpleRNN
  - SimpleRNN：SimpleRNN 层，全连接的 RNN 层，该层的输出可以反馈到输入。
  - Units：整数，定义输出空间的维度，默认 8。
  - Activation：选择激活函数进行非线性转换，如'relu'。若选择'None'，默认使用'linear'。
  - Use Bias：选择是否使用偏置向量，默认'True'，即包含偏置。
  - Kernel Initializer：选择权值初始化方法，默认'glorot_uniform'。
  - Recurrent Initializer：选择递归核权值矩阵初始化方法，用于递

归状态的线性变换，默认'orthogonal'。

- Bias Initializer：选择偏置向量的初始化方法，默认'zeros'。
- Kernel Regularizer：定义权值矩阵的正则化方法，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
- Recurrent Regularizer：定义递归和权值矩阵的正则化方法，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
- Bias Regularizer：定义偏置向量的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
- Activity Regularizer：定义该层输出的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
- Kernel Constraint：选择权值矩阵的约束（限制）函数，默认 None。
- Recurrent Constraint：选择递归和权值矩阵的约束（限制）函数，默认 None。
- Bias Constraint：选择偏置向量的约束（限制）函数，默认 None。
- Dropout：0 到 1 之间的整数或者浮点数，定义输入的线性变换的单位下降的分数，默认 0.0。
- Recurrent Dropout：0 到 1 之间的整数或者浮点数，定义递归状态的线性变换中单位下降的部分，默认 0.0。
- Return Sequences：布尔值，指定是返回输出序列中的最后一个输出，还是返回完整序列，默认 False。
- Return State：布尔值，指定是否返回输出之外的最后一个状态，默认 False。
- Go Backwards：布尔值，指定是否反向处理输入序列并返回相反的序列。如果为真，则反向处理输入序列并返回相反的序列，默认 False。
- Stateful：布尔值，如果为真，则批处理中索引 i 处的每个样本的最后一个状态将用作下一批处理中索引 i 的样本的初始状态，默认 False。

- Unroll：布尔值，如果为真，则网络将展开，否则将使用符号循环。展开可以加快 RNN 的速度，尽管它往往需要更多的内存。展开只适用于短序列，默认 False。
- Input Shape：正整数组成的元组或 None。检索图层的输入形状,默认为 None。

- GRU
  - GRU：GRU 层，门控循环单元，作为 LSTM 的一种变体，将忘记门和输入门合成了一个单一的更新门。
  - Units：整数，定义输出空间的维度，默认 8。
  - Activation：选择激活函数进行非线性转换，如'relu'。若选择'None'，默认使用'linear'。
  - Recurrent Activation：选择循环时间步的激活函数，如'relu'。若选择'None'，默认使用'linear'。
  - Use Bias：选择是否使用偏置向量，默认'True'，即包含偏置。
  - Kernel Initializer：选择权值初始化方法，默认'glorot_uniform'。
  - Recurrent Initializer：选择递归核权值矩阵初始化方法，用于递归状态的线性变换，默认'orthogonal'。
  - Bias Initializer：选择偏置向量的初始化方法，默认'zeros'。
  - Kernel Regularizer：定义权值矩阵的正则化方法，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
  - Recurrent Regularizer：定义递归和权值矩阵的正则化方法，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
  - Bias Regularizer：定义偏置向量的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
  - Activity Regularizer：定义该层输出的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
  - Kernel Constraint：选择权值矩阵的约束（限制）函数，默认 Non

e。

- Recurrent Constraint：选择递归和权值矩阵的约束（限制）函数，默认 None。
- Bias Constraint：选择偏置向量的约束（限制）函数，默认 None。
- Dropout：0 到 1 之间的整数或者浮点数，定义输入的线性变换的单位下降的分数，默认 0.0。
- Recurrent Dropout：0 到 1 之间的整数或者浮点数，定义递归状态的线性变换中单位下降的部分，默认 0.0。
- Implementation：执行模式，1 或 2。模式 1 将其操作结构化为大量的小点积和加法，而模式 2 将其批量处理成更少、更大的操作。这些模式在不同的硬件和不同的应用程序上具有不同的性能，默认 2。
- Return Sequences：布尔值，指定是返回输出序列中的最后一个输出，还是返回完整序列，默认 False。
- Return State：布尔值，指定是否返回输出之外的最后一个状态，默认 False。
- Go Backwards：布尔值，指定是否反向处理输入序列并返回相反的序列。如果为真，则反向处理输入序列并返回相反的序列，默认 False。
- Stateful：布尔值，如果为真，则批处理中索引 i 处的每个样本的最后一个状态将用作下一批处理中索引 i 的样本的初始状态，默认 False。
- Unroll：布尔值，如果为真，则网络将展开，否则将使用符号循环。展开可以加快 RNN 的速度，尽管它往往需要更多的内存。展开只适用于短序列，默认 False。
- Unroll：布尔值，指定是否在矩阵乘法之前或之后使用重置门，默认 False，表示之前，True 表示之后。
- Input Shape：正整数组成的元组或 None。检索图层的输入形状，默认为 None。

- LSTM
  - LSTM：LSTM 层，长短期记忆网络层，可以将以往学习的结果应用到当前学习的模型。
  - Units：整数，定义输出空间的维度，默认 8。
  - Activation：选择激活函数进行非线性转换，如'relu'。若选择'None'，默认使用'linear'。
  - Recurrent Activation：选择循环时间步的激活函数，如'relu'。若选择'None'，默认使用'linear'。
  - Use Bias：选择是否使用偏置向量，默认'True'，即包含偏置。
  - Kernel Initializer：选择权值初始化方法，默认'glorot_uniform'。
  - Recurrent Initializer：选择递归核权值矩阵初始化方法，用于递归状态的线性变换，默认'orthogonal'。
  - Bias Initializer：选择偏置向量的初始化方法，默认'zeros'。
  - Unit Forget Bias：布尔值。如果为真，则在初始化时对忘记门的偏差加 1。将其设置为 true 也将强制 bias_initializer='zeros'. 默认 True。
  - Kernel Regularizer：定义权值矩阵的正则化方法，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
  - Recurrent Regularizer：定义递归和权值矩阵的正则化方法，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
  - Bias Regularizer：定义偏置向量的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
  - Activity Regularizer：定义该层输出的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
  - Kernel Constraint：选择权值矩阵的约束（限制）函数，默认 None。
  - Recurrent Constraint：选择递归和权值矩阵的约束（限制）函数，默认 None。

- Bias Constraint：选择偏置向量的约束（限制）函数，默认 None。
- Dropout：0 到 1 之间的整数或者浮点数，定义输入的线性变换的单位下降的分数，默认 0.0。
- Recurrent Dropout：0 到 1 之间的整数或者浮点数，定义递归状态的线性变换中单位下降的部分，默认 0.0。
- Implementation：执行模式，1 或 2。模式 1 将其操作结构化为大量的小点积和加法，而模式 2 将其批量处理成更少、更大的操作。这些模式在不同的硬件和不同的应用程序上具有不同的性能，默认 2。
- Return Sequences：布尔值，指定是返回输出序列中的最后一个输出，还是返回完整序列，默认 False。
- Return State：布尔值，指定是否返回输出之外的最后一个状态，默认 False。
- Go Backwards：布尔值，指定是否反向处理输入序列并返回相反的序列。如果为真，则反向处理输入序列并返回相反的序列，默认 False。
- Stateful：布尔值，如果为真，则批处理中索引 i 处的每个样本的最后一个状态将用作下一批处理中索引 i 的样本的初始状态，默认 False。
- Unroll：布尔值，如果为真，则网络将展开，否则将使用符号循环。展开可以加快 RNN 的速度，尽管它往往需要更多的内存。展开只适用于短序列，默认 False。
- Input Shape：正整数组成的元组或 None。检索图层的输入形状，默认为 None。

- ConvLSTM2D
  - convLSTM2D：ConvLSTM2D 层，卷积 LSTM 层，类似于 LSTM 层，其输入转换和递归转换都是卷积的。
  - Filters：正整数，定义卷积核的数目，即输出的维度，默认 32。
  - Kernel Size：整数或由 2 个整数构成的列表或元组，指定二维卷积

窗口的高度和宽度，可以是单个整数，以便为所有空间维度指定相同的值，默认(2,?2)。

- Strides：单个整数或由两个整数构成的列表或元组，指定沿高度和宽度的卷积步长。可以是单个整数，以便为所有空间维度指定相同的值，默认（1,1）。
- Padding：选择填充的补 0 策略，valid 表示只进行有效的卷积，即对边界数据不处理；same 表示保留边界处的卷积结果，使输出形状和输入形状相同；默认 valid。
- Data Format：选择数据格式，定义输入数据的维度顺序，channels_last 对应输入形状为（batch, time, ... , channels），channels_first 对应输入形状为(batch, time, channels, ...)，默认 channels_last。
- Dilation Rate：一个整数或 n 个整数的元组/列表，指定用于扩展卷积的扩展速率。可以是单个整数，以便为所有空间维度指定相同的值，默认(1,?1)。
- Activation：选择激活函数进行非线性转换，如'relu'。若选择'None'，默认使用'linear'。
- Recurrent Activation：选择循环时间步的激活函数，如'relu'。若选择'None'，默认使用'linear'。
- Use Bias：选择是否使用偏置向量，默认'True'，即包含偏置。
- Kernel Initializer：选择权值初始化方法，默认'glorot_uniform'。
- Recurrent Initializer：选择递归核权值矩阵初始化方法，用于递归状态的线性变换，默认'orthogonal'。
- Bias Initializer：选择偏置向量的初始化方法，默认'zeros'。
- Unit Forget Bias：布尔值。如果为真，则在初始化时对忘记门的偏差加 1。将其设置为 true 也将强制 bias_initializer='zeros'. 默认 True。
- Kernel Regularizer：定义权值矩阵的正则化方法，如 l1(0.01)、l

2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。

- Bias Regularizer：定义偏置向量的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
- Activity Regularizer：定义该层输出的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
- Kernel Constraint：选择权值矩阵的约束（限制）函数，默认 None。
- Bias Constraint：选择偏置向量的约束（限制）函数，默认 None。
- Return Sequences：布尔值，指定是返回输出序列中的最后一个输出，还是返回完整序列，默认 False。
- Go Backwards：布尔值，指定是否反向处理输入序列并返回相反的序列。如果为真，则反向处理输入序列并返回相反的序列，默认 False。
- Stateful：布尔值，如果为真，则批处理中索引 i 处的每个样本的最后一个状态将用作下一批处理中索引 i 的样本的初始状态，默认 False。
- Dropout：0 到 1 之间的整数或者浮点数，定义输入的线性变换的单位下降的分数，默认 0.0。
- Recurrent Dropout：0 到 1 之间的整数或者浮点数，定义递归状态的线性变换中单位下降的部分，默认 0.0。

- SimpleRNNCell
  - SimpleRNNCell：SimpleRNNCell 层，全连接 RNN 单元格。
  - Units：整数，定义输出空间的维度，默认 16。
  - Activation：选择激活函数进行非线性转换，如'relu'。若选择'None'，默认使用'linear'。
  - Use Bias：选择是否使用偏置向量，默认'True'，即包含偏置。
  - Kernel Initializer：选择权值初始化方法，默认'glorot_uniform'。

- Recurrent Initializer：选择递归核权值矩阵初始化方法，用于递归状态的线性变换，默认'orthogonal'。
- Bias Initializer：选择偏置向量的初始化方法，默认'zeros'。
- Kernel Regularizer：定义权值矩阵的正则化方法，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
- Recurrent Regularizer：定义递归和权值矩阵的正则化方法，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
- Bias Regularizer：定义偏置向量的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
- Kernel Constraint：选择权值矩阵的约束（限制）函数，默认 None。
- Recurrent Constraint：选择递归和权值矩阵的约束（限制）函数，默认 None。
- Bias Constraint：选择偏置向量的约束（限制）函数，默认 None。
- Dropout：0 到 1 之间的整数或者浮点数，定义输入的线性变换的单位下降的分数，默认 0.0。
- Recurrent Dropout：0 到 1 之间的整数或者浮点数，定义递归状态的线性变换中单位下降的部分，默认 0.0。

- GRUCell
  - GRUCell：GRUCell 层，GRU 单元格。
  - Units：整数，定义输出空间的维度，默认 16。
  - Activation：选择激活函数进行非线性转换，如'relu'。若选择'None'，默认使用'linear'。
  - Recurrent Activation：选择循环时间步的激活函数，如'relu'。若选择'None'，默认使用'linear'。
  - Use Bias：选择是否使用偏置向量，默认'True'，即包含偏置。
  - Kernel Initializer：选择权值初始化方法，默认'glorot_uniform'。

- Recurrent Initializer：选择递归核权值矩阵初始化方法，用于递归状态的线性变换，默认'orthogonal'。
- Bias Initializer：选择偏置向量的初始化方法，默认'zeros'。
- Kernel Regularizer：定义权值矩阵的正则化方法，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
- Recurrent Regularizer：定义递归和权值矩阵的正则化方法，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
- Bias Regularizer：定义偏置向量的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
- Kernel Constraint：选择权值矩阵的约束（限制）函数，默认 None。
- Recurrent Constraint：选择递归和权值矩阵的约束（限制）函数，默认 None。
- Bias Constraint：选择偏置向量的约束（限制）函数，默认 None。
- Dropout：0 到 1 之间的整数或者浮点数，定义输入的线性变换的单位下降的分数，默认 0.0。
- Recurrent Dropout：0 到 1 之间的整数或者浮点数，定义递归状态的线性变换中单位下降的部分，默认 0.0。
- Implementation：执行模式，1 或 2。模式 1 将其操作结构化为大量的小点积和加法，而模式 2 将其批量处理成更少、更大的操作。这些模式在不同的硬件和不同的应用程序上具有不同的性能，默认 2。
- Unroll：布尔值，指定是否在矩阵乘法之前或之后使用重置门，默认 False，表示之前，True 表示之后。

- LSTMCell
  - LSTMCell：LSTMCell 层，LSTM 单元格。
  - Units：整数，定义输出空间的维度，默认 8。
  - Activation：选择激活函数进行非线性转换，如'relu'。若选择'None'，默认使用'linear'。

- Recurrent Activation：选择循环时间步的激活函数，如'relu'。若选择'None'，默认使用'linear'。
- Use Bias：选择是否使用偏置向量，默认'True'，即包含偏置。
- Kernel Initializer：选择权值初始化方法，默认'glorot_uniform'。
- Recurrent Initializer：选择递归核权值矩阵初始化方法，用于递归状态的线性变换，默认'orthogonal'。
- Bias Initializer：选择偏置向量的初始化方法，默认'zeros'。
- Unit Forget Bias：布尔值。如果为真，则在初始化时对忘记门的偏差加 1。将其设置为 true 也将强制 bias_initializer='zeros'. 默认 True。
- Kernel Regularizer：定义权值矩阵的正则化方法，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
- Recurrent Regularizer：定义递归和权值矩阵的正则化方法，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
- Bias Regularizer：定义偏置向量的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
- Kernel Constraint：选择权值矩阵的约束（限制）函数，默认 None。
- Recurrent Constraint：选择递归和权值矩阵的约束（限制）函数，默认 None。
- Bias Constraint：选择偏置向量的约束（限制）函数，默认 None。
- Dropout：0 到 1 之间的整数或者浮点数，定义输入的线性变换的单位下降的分数，默认 0.0。
- Recurrent Dropout：0 到 1 之间的整数或者浮点数，定义递归状态的线性变换中单位下降的部分，默认 0.0。
- Implementation：执行模式，1 或 2。模式 1 将其操作结构化为大量的小点积和加法，而模式 2 将其批量处理成更少、更大的操作。这些模式在不同的硬件和不同的应用程序上具有不同的性能，默认 2。

- CuDNNGRU
  - CuDNNGRU：CuDNNGRU 层，由 CuDNN 支持的快速 GRU 实现。
  - Units：整数，定义输出空间的维度，默认 8。
  - Kernel Initializer：选择权值初始化方法，默认'glorot_uniform'。
  - Recurrent Initializer：选择递归核权值矩阵初始化方法，用于递归状态的线性变换，默认'orthogonal'。
  - Bias Initializer：选择偏置向量的初始化方法，默认'zeros'。
  - Kernel Regularizer：定义权值矩阵的正则化方法，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
  - Recurrent Regularizer：定义递归和权值矩阵的正则化方法，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
  - Bias Regularizer：定义偏置向量的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
  - Activity Regularizer：定义该层输出的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
  - Kernel Constraint：选择权值矩阵的约束（限制）函数，默认 None。
  - Recurrent Constraint：选择递归和权值矩阵的约束（限制）函数，默认 None。
  - Bias Constraint：选择偏置向量的约束（限制）函数，默认 None。
  - Return Sequences：布尔值，指定是返回输出序列中的最后一个输出，还是返回完整序列，默认 False。
  - Return State：布尔值，指定是否返回输出之外的最后一个状态，默认 False。
  - Stateful：布尔值，如果为真，则批处理中索引 i 处的每个样本的最后一个状态将用作下一批处理中索引 i 的样本的初始状态，默认 False。

- CuDNNLSTM
  - CuDNNLSTM：快速 LSTM 层，由 CuDNN 支持的快速 LSTM 实现。
  - Units：整数，定义输出空间的维度，默认 8。
  - Kernel Initializer：选择权值初始化方法，默认'glorot_uniform'。
  - Recurrent Initializer：选择递归核权值矩阵初始化方法，用于递归状态的线性变换，默认'orthogonal'。
  - Bias Initializer：选择偏置向量的初始化方法，默认'zeros'。
  - Unit Forget Bias：布尔值。如果为真，则在初始化时对忘记门的偏差加 1。将其设置为 true 也将强制 bias_initializer='zeros'. 默认 True。
  - Kernel Regularizer：定义权值矩阵的正则化方法，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
  - Recurrent Regularizer：定义递归和权值矩阵的正则化方法，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
  - Bias Regularizer：定义偏置向量的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
  - Activity Regularizer：定义该层输出的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01,?l2=0.01)或 None，默认 None。
  - Kernel Constraint：选择权值矩阵的约束（限制）函数，默认 None。
  - Recurrent Constraint：选择递归和权值矩阵的约束（限制）函数，默认 None。
  - Bias Constraint：选择偏置向量的约束（限制）函数，默认 None。
  - Return Sequences：布尔值，指定是返回输出序列中的最后一个输出，还是返回完整序列，默认 False。
  - Return State：布尔值，指定是否返回输出之外的最后一个状态，默认 False。

- Stateful：布尔值，如果为真，则批处理中索引 i 处的每个样本的最后一个状态将用作下一批处理中索引 i 的样本的初始状态，默认 False。

- Attention
  - Attention：点乘注意力层，Luong Attention。
  - use_scale：如果为 True，将会创建一个标量的变量对注意力分数进行缩放，默认'True'。
  - causal：设置为 True 可使解码器 self-attention。添加一个罩，使位置 i 无法参与位置 j> i。这样可以防止信息流从未来传递到过去，默认'True'。
  - dropout：输入 0 到 1 之间的整数或者浮点数，attention scores 下降的百分比，默认 0.0。

- AdditiveAttention
  - AdditiveAttention：加法注意力层，Bahdanau Attention。
  - use_scale：如果为 True，将会创建一个标量的变量对注意力分数进行缩放，默认'True'。
  - causal：设置为 True 可使解码器 self-attention。添加一个罩，使位置 i 无法参与位置 j> i。这样可以防止信息流从未来传递到过去，默认'True'。
  - dropout：输入 0 到 1 之间的整数或者浮点数，attention scores 下降的百分比，默认 0.0。

- MultiHeadAttention
  - MultiHeadAttention：多头注意力层，多头自注意力是将一个查询（query）到一系列（键 key-值 value）对的线性映射 h 次，再进行

h 次自注意力操作，得到 h 个自注意结果进行拼接状。

- num_heads：正整数，注意力头的数量，如 2。
- key_dim：正整数，每个注意力头 query 和 key 的大小，如 2。
- value_dim：正整数或者 None，每个注意力头的价值大小，默认为 None。
- dropout：输入 0 到 1 之间的整数或者浮点数，丢弃率，默认 0.0。
- Use Bias：选择是否使用偏置向量，默认'True'，即包含偏置。
- output_shape：输出张量的预期形状，除了批次和序列维度。如果未指定，则投影回关键函数维度。默认设置为 None。
- attention_axes：应用注意力的轴。None 表示对所有轴的注意力，但批处理、头部和特征。默认 None。
- Kernel Initializer：选择权值初始化方法，默认'glorot_uniform'。
- Bias Initializer：选择偏置向量的初始化方法，默认'zeros'。
- Kernel Regularizer：定义权值矩阵的正则化方法，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01, l2=0.01)或 None，默认 None。
- Bias Regularizer：定义偏置向量的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01, l2=0.01)或 None，默认 None。
- Bias Regularizer：定义激活函数的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01, l2=0.01)或 None，默认 None。
- Kernel Constraint：选择权值矩阵的约束（限制）函数，默认 None。
- Bias Constraint：选择偏置向量的约束（限制）函数，默认 None。

- TimeDistributed
  - TimeDistributed：TimeDistributed 包装器，可以把一个层应用到输入的每一个时间步上。
  - Input Shape：正整数组成的元组或 None。检索图层的输入形状，默认为 None。

- Bidirectional
  - Bidirectional：双向 RNN 包装器。
  - merge_mode：前向和后向 RNN 输出的结合方式，为 sum,mul,concat, ave 和 None 之一，若设为 None，则返回值不结合，而是以列表的形式返回。
  - Input Shape：正整数组成的元组或 None。检索图层的输入形状，默认为 None。

- 自定义层
  - 自定义层：自定义层。
  - 自定义层名称：填写自定义层函数的名称。
  - 自定义层脚本：按照模板添加自定义层脚本。

（二）基础网络模型

- densenet
  - densenet：DenseNet 模型，稠密卷积网络模型结构。
  - version：DenseNet 网络模型的版本，默认情况下，使用 DenseNet121。
  - include_top：是否在网络顶部包括全连接层。默认情况下，使用 True。
  - weights：None 代表随机初始化，'imagenet' 代表加载在 ImageNet 上预训练的权值。
  - input_tensor：可选，Keras tensor 作为模型的输入（即 layers.Input() 输出的 tensor），默认为 None。
  - input_shape：可选，输入尺寸元组，仅当 include_top=False 时有效，否则输入形状必须是 (244, 244, 3)（对于 channels_last 数据格式），或者 (3, 244, 244)（对于 channels_first 数据格

式）。它必须拥有 3 个输入通道，且宽高必须不小于 32。例如（200, 200, 3）是一个合法的输入尺寸，默认为 None。

- pooling：当 include_top 为 False 时，该参数指定了特征提取时的池化方式。None 代表不池化，直接输出最后一层卷积层的输出，该输出是一个四维张量；'avg' 代表全局平均池化（GlobalAveragePooling2D），相当于在最后一层卷积层后面再加一层全局平均池化层，输出是一个二维张量。
- classes：正整数，图片分类的类别数，仅当 include_top 为 True 并且不加载预训练权值时可用，默认 1000。

- efficientnet
  - efficientnet：efficientnet 模型。
  - version：efficientnet 网络模型的版本，默认情况下，使用 EfficientNetB0。
  - include_top：是否在网络顶部包括全连接层。默认情况下，使用 True。
  - weights：None 代表随机初始化，'imagenet' 代表加载在 ImageNet 上预训练的权值。
  - input_tensor：可选，Keras tensor 作为模型的输入（即 layers.Input() 输出的 tensor），默认为 None。
  - input_shape：可选，输入尺寸元组，仅当 include_top=False 时有效，它必须拥有 3 个输入通道，默认为 None。
  - pooling：当 include_top 为 False 时，该参数指定了特征提取时的池化方式。None 代表不池化，直接输出最后一层卷积层的输出，该输出是一个四维张量；'avg' 代表全局平均池化（GlobalAveragePooling2D），相当于在最后一层卷积层后面再加一层全局平均池化层，输出是一个二维张量。
  - classes：正整数，图片分类的类别数，仅当 include_top 为 True 并且不加载预训练权值时可用，默认 1000。

- classifier_activation：选择激活函数进行非线性转换，如'softmax'，默认使用'softmax'。

- InceptionResNetV2
  - InceptionResNetV2：Inception-ResNet V2 模型，权值由 ImageNet 训练而来。
  - include_top：是否在网络顶部包括全连接层。默认情况下，使用 True。
  - weights：None 代表随机初始化，'imagenet' 代表加载在 ImageNet 上预训练的权值。
  - input_tensor：可选，Keras tensor 作为模型的输入（即 layers.Input() 输出的 tensor），默认为 None。
  - input_shape：输入尺寸元组，仅当 include_top=False 时有效，否则输入形状必须是 (299, 299, 3)（对于 channels_last 数据格式），或者 (3, 299, 299)（对于 channels_first 数据格式）。它必须拥有 3 个输入通道，且宽高必须不小于 139。例如 (150, 150, 3) 是一个合法的输入尺寸，默认为 None。
  - pooling：当 include_top 为 False 时，该参数指定了特征提取时的池化方式。None 代表不池化，直接输出最后一层卷积层的输出，该输出是一个四维张量；'avg' 代表全局平均池化（GlobalAveragePooling2D），相当于在最后一层卷积层后面再加一层全局平均池化层，输出是一个二维张量。
  - classes：正整数，图片分类的类别数，仅当 include_top 为 True 并且不加载预训练权值时可用，默认 1000。
  - classifier_activation：选择激活函数进行非线性转换，如'softmax'，默认使用'softmax'。

- InceptionV3
  - InceptionV3：InceptionV3 模型，权值由 ImageNet 训练而来。

- include_top：是否在网络顶部包括全连接层。默认情况下，使用 True。
- weights：None 代表随机初始化，'imagenet' 代表加载在 ImageNet 上预训练的权值。
- input_tensor：可选，Keras tensor 作为模型的输入（即 layers.Input() 输出的 tensor），默认为 None。
- input_shape：输入尺寸元组，仅当 include_top=False 时有效，否则输入形状必须是 (299, 299, 3)（对于 channels_last 数据格式），或者 (3, 299, 299)（对于 channels_first 数据格式）。它必须拥有 3 个输入通道，且宽高必须不小于 139。例如 (150, 150, 3) 是一个合法的输入尺寸，默认为 None。
- pooling：当 include_top 为 False 时，该参数指定了特征提取时的池化方式。None 代表不池化，直接输出最后一层卷积层的输出，该输出是一个四维张量；'avg' 代表全局平均池化（GlobalAveragePooling2D），相当于在最后一层卷积层后面再加一层全局平均池化层，输出是一个二维张量。
- classes：正整数，图片分类的类别数，仅当 include_top 为 True 并且不加载预训练权值时可用，默认 1000。
- classifier_activation：选择激活函数进行非线性转换，如'softmax'，默认使用'softmax'。

- MobileNet
  - MobileNet：MobileNet 模型，权值由 ImageNet 训练而来。
  - include_top：是否在网络顶部包括全连接层。默认情况下，使用 True。
  - alpha：非负的浮点数，控制网络的宽度.如果 alpha < 1.0，则同比例减少每层的滤波器个数；如果 alpha > 1.0，则同比例增加每层的滤波器个数；如果 alpha = 1，使用论文默认的滤波器个数；默认 1.0。

- depth_multiplier：depthwise 卷积的深度乘子，也称为（分辨率乘子），默认 1。
- Dropout：浮点数或科学计数法，丢弃率，默认 1e-3。
- weights：None 代表随机初始化，'imagenet' 代表加载在 ImageNet 上预训练的权值。
- input_tensor：可选，Keras tensor 作为模型的输入（即 layers.Input() 输出的 tensor），默认为 None。
- input_shape：可选，输入尺寸元组，仅当 include_top=False 时有效，否则输入形状必须是 (224, 224, 3)（对于 channels_last 数据格式），或者 (3, 224, 224)（对于 channels_first 数据格式）。它必须拥有 3 个输入通道，且宽高必须不小于 32。例如 (200, 200, 3) 是一个合法的输入尺寸，默认为 None。
- pooling：当 include_top 为 False 时，该参数指定了特征提取时的池化方式。None 代表不池化，直接输出最后一层卷积层的输出，该输出是一个四维张量；'avg' 代表全局平均池化（GlobalAveragePooling2D），相当于在最后一层卷积层后面再加一层全局平均池化层，输出是一个二维张量。
- classes：正整数，图片分类的类别数，仅当 include_top 为 True 并且不加载预训练权值时可用，默认 1000。
- classifier_activation：选择激活函数进行非线性转换，如'softmax'，默认使用'softmax'。

- MobileNetV2
  - MobileNetV2：MobileNetV2 模型，权值由 ImageNet 训练而来。
  - include_top：是否在网络顶部包括全连接层。默认情况下，使用 True。
  - alpha：非负的浮点数，控制网络的宽度.如果 alpha < 1.0，则同比例减少每层的滤波器个数；如果 alpha > 1.0，则同比例增加每层的滤波器个数；如果 alpha = 1，使用论文默认的滤波器个数；

默认 1.0。

- weights：None 代表随机初始化，'imagenet' 代表加载在 ImageNet 上预训练的权值。
- input_tensor：可选，Keras tensor 作为模型的输入（即 layers.Input() 输出的 tensor），默认为 None。
- input_shape：可选形状元组，如果要使用输入 img 分辨率不是（224，224，3）的模型，则需要指定，应是 3 个通道（224,224,3）。如果要从输入张量推断输入形状，也可以省略此选项。如果您选择同时包含 input_tensor 和 input_shape，如果它们匹配，则将使用 input_shape，如果形状不匹配，则我们将抛出一个错误。如（160，160，3）将是一个有效值。默认为 None。
- pooling：当 include_top 为 False 时，该参数指定了特征提取时的池化方式。None 代表不池化，直接输出最后一层卷积层的输出，该输出是一个四维张量；'avg' 代表全局平均池化（GlobalAveragePooling2D），相当于在最后一层卷积层后面再加一层全局平均池化层，输出是一个二维张量。
- classes：正整数，图片分类的类别数，仅当 include_top 为 True 并且不加载预训练权值时可用，默认 1000。
- classifier_activation：选择激活函数进行非线性转换，如'softmax'，默认使用'softmax'。

- nasnet
  - nasnet：NASNet 模型，神经结构搜索网络模型，权值由 ImageNet 训练而来。
  - version：nasnet 网络模型的版本，默认情况下，使用 NASNetMobile。
  - include_top：是否在网络顶部包括全连接层。默认情况下，使用 True。
  - weights：None 代表随机初始化，'imagenet' 代表加载在 ImageNe

t 上预训练的权值。

- input_tensor：可选，Keras tensor 作为模型的输入（即 layers.Input() 输出的 tensor），默认为 None。
- input_shape：输入尺寸元组，仅当 include_top=False 时有效，对于 NASNetMobile 模型来说，输入形状必须是 (224, 224, 3)（channels_last 格式）或 (3, 224, 224)（channels_first 格式），对于 NASNetLarge 模型来说，输入形状必须是 (331, 331, 3)（channels_last 格式）或 (3, 331, 331)（channels_first 格式）。它必须为 3 个输入通道，且宽高必须不小于 32，比如 (200, 200, 3) 是一个合法的输入尺寸，默认为 None。
- pooling：当 include_top 为 False 时，该参数指定了特征提取时的池化方式。None 代表不池化，直接输出最后一层卷积层的输出，该输出是一个四维张量；'avg' 代表全局平均池化（GlobalAveragePooling2D），相当于在最后一层卷积层后面再加一层全局平均池化层，输出是一个二维张量。
- classes：正整数，图片分类的类别数，仅当 include_top 为 True 并且不加载预训练权值时可用，默认 1000。

- resnet
  - resnet：resnet 模型，权值由 ImageNet 训练而来。
  - version：ResNet 网络模型的版本，默认情况下，使用 ResNet101。
  - include_top：是否在网络顶部包括全连接层。默认情况下，使用 True。
  - weights：None 代表随机初始化，'imagenet' 代表加载在 ImageNet 上预训练的权值。
  - input_tensor：可选，Keras tensor 作为模型的输入（即 layers.Input() 输出的 tensor），默认为 None。
  - input_shape：输入尺寸元组，仅当 include_top=False 时有效，否则输入形状必须是 (244, 244, 3)（对于 channels_last 数据格

式），或者（3, 244, 244）（对于 channels_first 数据格式）。它必须拥有 3 个输入通道，且宽高必须不小于 32。例如（200, 200, 3）是一个合法的输入尺寸，默认为 None。

- pooling：当 include_top 为 False 时，该参数指定了特征提取时的池化方式。None 代表不池化，直接输出最后一层卷积层的输出，该输出是一个四维张量；'avg'代表全局平均池化（GlobalAveragePooling2D），相当于在最后一层卷积层后面再加一层全局平均池化层，输出是一个二维张量。
- classes：正整数，图片分类的类别数，仅当 include_top 为 True 并且不加载预训练权值时可用，默认 1000。

- resnet_v2
  - resnet_v2：resnet_v2 模型，权值由 ImageNet 训练而来。
  - version：ResNetV2 网络模型的版本，默认情况下，使用 ResNet101V2。
  - include_top：是否在网络顶部包括全连接层。默认情况下，使用 True。
  - weights：None 代表随机初始化，'imagenet'代表加载在 ImageNet 上预训练的权值。
  - input_tensor：可选，Keras tensor 作为模型的输入（即 layers.Input() 输出的 tensor），默认为 None。
  - input_shape：输入尺寸元组，仅当 include_top=False 时有效，否则输入形状必须是（244, 244, 3）（对于 channels_last 数据格式），或者（3, 244, 244）（对于 channels_first 数据格式）。它必须拥有 3 个输入通道，且宽高必须不小于 32。例如（200, 200, 3）是一个合法的输入尺寸，默认为 None。
  - pooling：当 include_top 为 False 时，该参数指定了特征提取时的池化方式。None 代表不池化，直接输出最后一层卷积层的输出，该输出是一个四维张量；'avg'代表全局平均池化（GlobalAverage

Pooling2D），相当于在最后一层卷积层后面再加一层全局平均池化层，输出是一个二维张量。

- classes：正整数，图片分类的类别数，仅当 include_top 为 True 并且不加载预训练权值时可用，默认 1000。
- classifier_activation：选择激活函数进行非线性转换，如'softmax'，默认使用'softmax'。

- VGG16
  - VGG16：VGG16 模型，权值由 ImageNet 训练而来。
  - include_top：是否在网络顶部包括全连接层。默认情况下，使用 True。
  - weights：None 代表随机初始化，'imagenet' 代表加载在 ImageNet 上预训练的权值。
  - input_tensor：可选，Keras tensor 作为模型的输入（即 layers.Input() 输出的 tensor），默认为 None。
  - input_shape：输入尺寸元组，仅当 include_top=False 时有效，否则输入形状必须是 (224, 224, 3)（对于 channels_last 数据格式），或者 (3, 224, 224)（对于 channels_first 数据格式）。它必须拥有 3 个输入通道，且宽高必须不小于 32。例如 (200, 200, 3) 是一个合法的输入尺寸，默认为 None。
  - pooling：当 include_top 为 False 时，该参数指定了特征提取时的池化方式。None 代表不池化，直接输出最后一层卷积层的输出，该输出是一个四维张量；'avg' 代表全局平均池化（GlobalAveragePooling2D），相当于在最后一层卷积层后面再加一层全局平均池化层，输出是一个二维张量。
  - classes：输入 None 或正整数，图片分类的类别数，仅当 include_top 为 True 并且不加载预训练权值时可用，默认 1000。
  - classifier_activation：选择激活函数进行非线性转换，如'softmax'，默认使用'softmax'。

- VGG19
  - VGG19：VGG19 模型，权值由 ImageNet 训练而来。
  - include_top：是否在网络顶部包括全连接层。默认情况下，使用 True。
  - weights：None 代表随机初始化，'imagenet' 代表加载在 ImageNet 上预训练的权值。
  - input_tensor：可选，Keras tensor 作为模型的输入（即 layers.Input() 输出的 tensor），默认为 None。
  - input_shape：输入尺寸元组，仅当 include_top=False 时有效，否则输入形状必须是 (244, 244, 3)（对于 channels_last 数据格式），或者 (3, 244, 244)（对于 channels_first 数据格式）。它必须拥有 3 个输入通道，且宽高必须不小于 32。例如 (200, 200, 3) 是一个合法的输入尺寸，默认为 None。
  - pooling：当 include_top 为 False 时，该参数指定了特征提取时的池化方式。None 代表不池化，直接输出最后一层卷积层的输出，该输出是一个四维张量；'avg' 代表全局平均池化（GlobalAveragePooling2D），相当于在最后一层卷积层后面再加一层全局平均池化层，输出是一个二维张量。
  - classes：正整数，图片分类的类别数，仅当 include_top 为 True 并且不加载预训练权值时可用，默认 1000。
  - classifier_activation：选择激活函数进行非线性转换，如'softmax'，默认使用'softmax'。

- Xception
  - Xception：Xception 模型，权值由 ImageNet 训练而来。
  - include_top：是否在网络顶部包括全连接层。默认情况下，使用 True。
  - weights：None 代表随机初始化，'imagenet' 代表加载在 ImageNe

t 上预训练的权值。

- input_tensor：可选，Keras tensor 作为模型的输入（即 layers.Input() 输出的 tensor），默认为 None。
- input_shape：输入尺寸元组，仅当 include_top=False 时有效（否则输入形状必须是 (299, 299, 3)，因为预训练模型是以这个大小训练的）。它必须拥有 3 个输入通道，且宽高必须不小于 71。例如 (150, 150, 3) 是一个合法的输入尺寸，默认为 None。
- pooling：当 include_top 为 False 时，该参数指定了特征提取时的池化方式。None 代表不池化，直接输出最后一层卷积层的输出，该输出是一个四维张量；'avg' 代表全局平均池化（GlobalAveragePooling2D），相当于在最后一层卷积层后面再加一层全局平均池化层，输出是一个二维张量。
- classes：正整数，图片分类的类别数，仅当 include_top 为 True 并且不加载预训练权值时可用，默认 1000。
- classifier_activation：选择激活函数进行非线性转换，如'softmax'，默认使用'softmax'。

- 模型库
  - 模型库：训练完成的模型。
  - 模型名称：现有模型的名称。
  - 模型版本：选择具体使用哪个模型。

（三）模型结果可视化

- TensorBoard
  - TensorBoard：配置 TensorBoard 参数。
  - histogram_freq：对于模型中各个层计算激活值和模型权重直方图的频率（训练轮数中）。 如果设置成 0 ，直方图不会被计算。对于直方图可视化的验证数据（或分离数据）一定要明确的指出。

- write_graph：是否在 TensorBoard 中可视化图像。
- write_grads：是否在 TensorBoard 中可视化梯度值直方图， histogram_freq 必须大于 0。
- batch_size：用以直方图计算的传入神经元网络输入批的大小。
- write_images：是否在 TensorBoard 中将模型权重以图片可视化。
- embeddings_freq：被选中的嵌入层会被保存的频率（在训练轮中）。
- embeddings_layer_names：一个列表，会被监测层的名字。 如果是 None 或空列表，那么所有的嵌入层都会被监测。
- embeddings_metadata：一个字典，对应层的名字到保存有这个嵌入层元数据文件的名字。
- embeddings_data：要嵌入在 embeddings_layer_names 指定的层的数据。 Numpy 数组（如果模型有单个输入）或 Numpy 数组列表（如果模型有多个输入）。
- update_freq：'batch' 或 'epoch' 或 整数。当使用 'batch' 时，在每个 batch 之后将损失和评估值写入到 TensorBoard 中。同样的情况应用到 'epoch' 中。如果使用整数，例如 10000，这个回调会在每 10000 个样本之后将损失和评估值写入到 TensorBoard 中。

- 评估结果
  - 评估结果：显示默认设置的损失函数和 metrics 函数。

附录三：算法库组件属性说明

（一）数值算法

- KNeighborsClassifier
  - KNeighborsClassifier:KNeighborsClassifier,K 近邻算法分类器，基于每个查询点的指定的最近邻数量进行学习分类。
  - 模型名称：保存模型的名称。
  - n_neighbors：整数，查询的默认邻居的数量，默认 5。
  - weights：选择用于预测的权重函数，包括 uniform 和 distance。其中 uniform 表示统一的权重，即在每一个邻居区域里的点的权重都是一样的，distance 表示权重点等于他们距离的倒数，使用此函数，更近的邻居对于所预测的点的影响更大。默认 uniform。
  - algorithm：选择最近邻所用的算法，包括 auto、ball_tree、kd_tree、brute，其中 auto 表示会根据传递给 fit 的数据自动决定使用最合适的算法，ball_tree 表示使用 BallTree 算法，kd_tree 表示使用 KDTree 算法，brute 表示使用暴力搜索。默认 auto。
  - leaf_size：正整数，指定叶子数量。叶子大小传递给 BallTree 或 KDTree，这可能会影响构造和查询的速度，以及存储树所需的内存，最优值取决于问题的性质。默认 30。
  - p：整数，闵科夫斯基距离指数或超参数。p=1 等价于使用曼哈顿距离（l1），p=2 等价于欧式距离(l2)，对于任何 p ，使用的是闵科夫斯基距离（l_p）。默认 2。
  - metric：选择用于树的距离度量，如果和 p=2 一起使用相当于使用标准欧几里德距离度量，默认 minkowski，目前仅支持实数值向量空间的度量。
  - metric_params：字典或 None，度量函数的附加关键字参数。默认 None。
  - n_jobs：整数或 None 或-1，表示要为邻居搜索运行的并行作业的数

量，-1 表示判断是否可以在这里设置并行，None 表示 1，-1 表示所有，默认 None。

- RadiusNeighborsClassifier
  - RadiusNeighborsClassifier：RadiusNeighborsClassifier，半径近邻分类算法,根据在给定半径内的邻域占比实现分类。
  - 模型名称：保存模型的名称。
  - Radius：浮点数，radius_neighbors 查询默认使用的参数空间范围，默认 1.0。
  - weights：选择用于预测的权重函数，包括 uniform 和 distance。其中 uniform 表示统一的权重，即在每一个邻居区域里的点的权重都是一样的，distance 表示权重点等于他们距离的倒数，使用此函数，更近的邻居对于所预测的点的影响更大。默认 uniform。
  - Algorithm：选择最近邻所用的算法，包括 auto、ball_tree、kd_tree、brute，其中 auto 表示会根据传递给 fit 的数据自动决定使用最合适的算法，ball_tree 表示使用 BallTree 算法，kd_tree 表示使用 KDTree 算法，brute 表示使用暴力搜索。默认 auto。
  - leaf_size：整数，指定叶子数量。叶子大小传递给 BallTree 或 KDTree，这可能会影响构造和查询的速度，以及存储树所需的内存，最优值取决于问题的性质。默认 30。
  - p：整数，闵科夫斯基距离指数或超参数。p=1 等价于使用曼哈顿距离（l1），p=2 等价于欧式距离(l2)，对于任何 p ，使用的是闵科夫斯基距离（l_p）。默认 2。
  - metric：选择用于树的距离度量，如果和 p=2 一起使用相当于使用标准欧几里德距离度量，默认 minkowski，目前仅支持实数值向量空间的度量。
  - outlier_label：用于离群样本(给定半径内没有邻居的样本)的标记。支持字符或整数标签(应与 y 相同类型)或列表的手动标签（如果多输出使用），也可输入“most_frequent”将最频繁出现的 y 标

记分配给离群值。默认 None，当检测到任何异常值时，将引发 ValueError 异常。

- metric_params：字典或 None，度量函数的附加关键字参数。默认 None。
- n_jobs：整数或 None 或-1，表示要为邻居搜索运行的并行作业的数量，None 表示 1，-1 表示所有，默认 None。

- NearestCentroid
  - NearestCentroid：NearestCentroid，最小中值距离分类算法，每个类由它的质心表示，测试样本被分类为具有最近的质心的类。
  - 模型名称：保存模型的名称。
  - Metric：选择在计算特性数组中实例之间的距离时使用的度量方法，默认 euclidean。
  - shrink_threshold：浮点数或 None，缩小中心体以去除特征的阈值，默认 None。

- LinearSVC
  - LinearSVC：LinearSVC，线性支持向量分类算法，与参数 kernel='linear'的 SVC 类似，使用 liblinear 实现，在惩罚和损失函数的选择上更灵活。
  - 模型名称：保存模型的名称。
  - penalty：指定处罚中使用的规范，包含 l1 和 l2，其中 l2 是 SVC 中使用的标准，l1 会导致系数向量稀疏化，默认 l2。
  - loss：指定损失函数，包含 hinge 和 squared_hinge，其中 hinge 是标准 SVM 的损失函数，squared_hinge 是 hinge 损失函数的平方，默认 squared_hinge。
  - Dual：选择求解对偶优化问题或原始优化问题的算法，当样本数大于特征数时，dual=False，默认 True。
  - tol：指定停止判据的公差，默认 1e-4。

- C：正则化的强度与 C 成反比，必须严格为正，默认 1.0。
- multi_class：如果 y 包含两个以上的类，则需要确定多类策略。包含 ovr 和 crammer_singer，其中 ovr 训练 n 类一对多分类器，crammer_singer 优化所有类的联合目标，默认 ovr。
- fit_intercept：指定是否计算该模型的截距，如果设置为 false，则不会在计算中使用截距(即数据应该已经居中)，默认 True。
- intercept_scaling：当 self.fit_intercept 为 True 时，实例向量 x 变为[x，self.intercept_scaling]，即具有等于 intercept_scaling 的常量值的“合成”特征被附加到实例向量。 截距变为 intercept_scaling *合成特征权重注意！ 合成特征权重与所有其他特征一样经受 l1 / l2 正则化。 为了减小正则化对合成特征权重（并因此对截距）的影响，必须增加 intercept_scaling。默认 1。
- class_weight：设置类的参数，包含 dict 和 balance，其中 balance 模式使用 y 的值自动调整与输入数据中的类频率成反比的权重，如果设置为 None，所有类均使用 1 为权重，默认 None。
- Verbose：指定是否启用详细输出，0 表示不启用，其他正整数表示启用，默认 0。
- random_state：正整数或随机数实例或 None，伪随机数生成器的种子，用于对数据进行变换以实现对偶坐标下降。如果是 int，则 random_state 是随机数生成器使用的种子， 如果是 RandomState 实例，则 random_state 是随机数生成器， 如果为 None，则随机数生成器是 np.random 使用的 RandomState 实例，默认 None。
- max_iter：要运行的最大迭代次数，默认 1000。

- SVC
  - SVC：SVC，C 支持向量分类算法，利用 libsvm 实现的支持向量机分类器。
  - 模型名称：保存模型的名称。
  - C：大于 0 的浮点数，惩罚参数，误差项的惩罚参数 C。C 值越小，

对误分类的惩罚减小，允许容错，泛化能力较强，C 值越大，效果相反，默认 1.0。

- Kernel：选择算法使用的核函数，包含 linear、poly、rbf、sigmoid、precomputed，其中 linear 表示线性函数，poly 表示多项式函数，rbf 表示 RBF 高斯核函数，sigmoid 表示 S 形曲线函数，默认 rbf。
- Degree：整数，多项式 poly 函数的维度，选择其他核函数时会被忽略，默认 3。
- gamma：auto 或浮点数或 scale，poly、rbf、sigmoid 核函数参数，默认 auto。
- coef0：浮点数，核函数的常数项，针对 poly 和 sigmoid 核函数有效，默认 0.0。
- Shrinking：是否采用 shrinking heuristic 方法，默认 True。
- Probability：是否采用概率估计，默认 False。
- tol：0 到 1 之间的小数，停止训练的误差值大小，默认 1e-3。
- cache_size：整数，核函数缓存大小，默认 200。
- class_weight：选择类别的权重设置方式，balanced 模式使用 y 值自动调整输入数据中与类频率成反比的权重，None 表示所有的类都使用权重 1，默认 None。
- Verbose：选择是否启用详细输出，默认 False。
- max_iter：整数，指定在求解器中迭代的硬限制，-1 表示无限制，默认-1。
- decision_function_shape：选择框，选择返回的决策函数的形状，ovr 表示一对多，ovo 表示一对一，默认 ovr。
- random_state：整数，随机数的种子值，默认 None。

- NuSVC
  - NuSVC：NuSVC，Nu-支持向量分类算法，类似于 SVC，但是使用一个参数来控制支持向量的数量。

- 模型名称：保存模型的名称。
- nu:浮点数,训练误差分数的上界和支持向量分数的下界,在区间(0,1.0]内，默认 0.5。
- Kernel：选择算法使用的核函数，包含 linear、poly、rbf、sigmoid、precomputed，其中 linear 表示线性函数，poly 表示多项式函数，rbf 表示 RBF 高斯核函数，sigmoid 表示 S 形曲线函数，默认 rbf。
- Degree：整数，多项式 poly 函数的维度，选择其他核函数时会被忽略，默认 3。
- gamma：auto 或浮点数或 scale，poly、rbf、sigmoid 核函数参数，默认 auto。
- coef0：浮点数，核函数的常数项，针对 poly 和 sigmoid 核函数有效，默认 0.0。
- shrinking：是否采用 shrinking heuristic 方法，默认 True。
- Probability：是否采用概率估计，默认 False。
- Tolerance：0 到 1 之间的小数，停止训练的误差值大小，默认 1e-3。
- cache_size：整数，核函数缓存大小，默认 200。
- class_weight：选择类别的权重设置方式，balanced 模式使用 y 值自动调整输入数据中与类频率成反比的权重，None 表示所有的类都使用权重 1，默认 None。
- Verbose：选择是否启用详细输出，默认 False。
- max_iter：整整，指定在求解器中迭代的硬限制，-1 表示无限制，默认-1。
- decision_function_shape：选择框，选择返回的决策函数的形状，ovr 表示一对多，ovo 表示一对一，默认 ovr。
- random_state：整数或 None，随机数的种子值，默认 None。

- DecisionTreeClassifier
  - DecisionTreeClassifier：DecisionTreeClassifier，决策树分类

算法。

- 模型名称：保存模型的名称。
- Criterion：选择测量分割质量的函数，包含 gini 和 entropy，gini 表示基尼系数，entropy 表示信息熵，默认 gini。
- Splitter：选择每个节点的分割策略，包含 best 和 random，best 表示在所有特征中找最好的切分点，适合样本量不大的时候；random 表示在部分特征中找切分点，适合样本数据量非常大的时候，默认 best。
- max_depth：正整数或 None，设置决策随机森林中的决策树的最大深度，深度越大，越容易过拟合，推荐树的深度为 5-20 之间，默认 None，None 表示不对决策树的最大深度作约束，直到每个叶子结点上的样本均属于同一类，或者少于 min_samples_leaf 参数指定的叶子结点上的样本个数。
- min_samples_split：整数，设置分割内部节点所需的最小样本数，默认 2,参数为整数时应不小于 2,参数为浮点数参数范围为(0,1.0]。
- min_samples_leaf：整数或者浮点数，设置叶节点所需的最小样本数,默认 1,参数为整数时应不小于 1,参数为浮点数时参数范围为(0,0.5]。
- min_weight_fraction_leaf：浮点数，叶节点所需的(所有输入样本的)总权值的最小加权分数，参数范围为[0.0,0.5]。当不提供 sample_weight 时，样本的权重相等，默认 0.0。
- max_features：整数或 None 或 auto 或 sqrt 或 log2，指定寻找最佳分割时要考虑的特性的数量，None 表示将所有特征作为 max_features,auto 表示 max_features=sqrt(n_features)，sqrt 表示 max_features=sqrt(n_features)，log2 表示 max_features=log2(n_features)，默认 None。
- random_state：整数或 None，随机数的种子值，如果为 None，随机数生成器就是 np.random 使用的 RandomState 实例，默认 None。
- max_leaf_nodes：正整数或 None，最大叶子节点数，通过限制该值

可以防止过拟合，None 表示不限制最大的叶子节点数，默认 None。

- min_impurity_decrease：浮点数，设置节点分裂阈值，如果节点分裂导致杂质的减少大于或等于该值，该节点将被分裂，默认 0.0。
- class_weight：字典或字典组成的列表或 None，指定样本各类别的权重，为了防止训练集某些类别的样本过多导致训练的决策树过于偏向这些类别，默认 None。

- BernoulliNB
  - BernoulliNB：BernoulliNB，伯努利朴素贝叶斯分类算法，用于处理二项分布问题。
  - 模型名称：保存模型的名称。
  - alpha：浮点数，拉普拉斯或利德斯通平滑的参数，如果设置为 0 则表示完全没有平滑，默认 1.0。
  - binarize：浮点数或 None，用于对样本特征进行二值化(映射到布尔值)的阈值。如果没有，则假定输入已经由二进制向量组成，默认 0.0。
  - fit_prior：选择是否学习类先验概率，False 表示使用统一的先验，默认 True。
  - class_prior：类数组结构（形状为(n_classes,)）或 None，类的先验概率，如果指定，则不根据数据调整先验，None 表示自动根据数据来进行计算，默认 None。

- LogisticRegression
  - LogisticRegression：LogisticRegression，逻辑回归分类算法。
  - 模型名称：保存模型的名称。
  - penalty：指定处罚中使用的规范，包含 l1 和 l2，newton-cg、sag 和 lbfgs 求解器仅支持 l2，elasticnet 仅支持 saga 求解器，none 表示不适用求解器，默认 l2。
  - Dual：选择求解对偶优化问题或原始优化问题的算法，只适用于 li

blinear 求解器且 penalty 为 l2 的情况，当样本数大于特征数时，dual=False，默认 False。

- tol：0 到 1 之间的小数，停止训练的误差值大小，默认 1e-4。
- C：正则化的强度与 C 成反比，必须严格为正，默认 1.0。
- fit_intercept：选择是否应将常数(偏差或截距)添加到决策函数，默认 True。
- intercept_scaling：当 self.fit_intercept 为 True 且使用 liblinear 求解器时，实例向量 x 变为[x，self.intercept_scaling]，即具有等于 intercept_scaling 的常量值的“合成”特征被附加到实例向量。 为了减小正则化对合成特征权重（并因此对截距）的影响，必须增加 intercept_scaling。默认 1.0。
- class_weight：设置类的参数，包含 dict 和 balance 和 None，其中 balance 模式使用 y 的值自动调整与输入数据中的类频率成反比的权重，如果设置为 None，所有类均使用 1 为权重，默认 None。
- random_state：整数或 None，随机数的种子值，如果为 None，随机数生成器就是 np.random 使用的 RandomState 实例，默认 None。
- solver：选择用于优化问题的算法，包含 liblinear、newton-cg、lbfgs、sag，对于小的数据集，建议选择 liblinear；而对于大的数据集，sag 和 saga 更快；对于多类问题，newton-cg、sag、saga 和 lbfgs 可以处理多项损失；newton-cg、lbfgs 和 sag 只处理 L2 处罚，而 liblinear 和 saga 处理 L1 处罚，默认 liblinear。
- max_iter：整数，指定求解器收敛所需的最大迭代次数，只适用于 newton-cg, sag 和 lbfgs 算法。
- multi_class：ovr 表示每个标签都适合一个二进制问题，multinomial 表示即使数据是二进制的，损失最小化也是整个概率分布的多项式损失，当 solver 选择 liblinear 时，multinomial 不可用，默认 ovr。
- verbose：整数，对于 liblinear 和 lbfgs 求解器，将 verbose 设置为任何正数均表示启用详细输出，默认 0。

- warm_start：True 表示重用前一个调用的解决方案以适应初始化，False 表示擦除前一个解决方案。liblinear 求解器不可用，默认 False。
- n_jobs：当 multi_class=ovr，表示在类之间并行化时使用的 CPU 核数。当求解器被设置为 liblinear 时，不管是否指定了 multi_class，此参数都将被忽略。None 表示 1，-1 表示使用所有可用的求解器，默认 None。

- ExtraTreeClassifier
  - ExtraTreeClassifier：ExtraTreeClassifier，随机树分类算法。
  - 模型名称：保存模型的名称。
  - Criterion：选择测量分割质量的函数，包含 gini 和 entropy，gini 表示基尼系数，entropy 表示信息熵，默认 gini。
  - Splitter：选择每个节点的分割策略，包含 best 和 random，best 表示在所有特征中找最好的切分点，适合样本量不大的时候；random 表示在部分特征中找切分点，适合样本数据量非常大的时候，默认 random。
  - max_depth：正整数或 None，设置决策随机森林中的决策树的最大深度，深度越大，越容易过拟合，推荐树的深度为 5-20 之间，默认 None。
  - min_samples_split：整数，设置分割内部节点所需的最小样本数，默认 2,参数为整数时不小于 2,参数为浮点数时参数范围为(0,1.0]。
  - min_samples_leaf：整数，设置叶节点所需的最小样本数，默认 1，参数为整数时应不小于 1，参数为浮点数时参数范围为(0,0.5]。
  - min_weight_fraction_leaf：浮点数，叶节点所需的(所有输入样本的)总权值的最小加权分数。当不提供 sample_weight 时，样本的权重相等，默认 0.0，参数范围为[0.0,0.5]。
  - max_features：整数或 None 或 auto 或 sqrt 或 log2，指定寻找最佳分割时要考虑的特性的数量，None 表示将所有特征作为 max_featur

es,auto 表示 max_features=sqrt(n_features)，sqrt 表示 max_features=sqrt(n_features)，log2 表示 max_features=log2(n_features)，默认 None。

- random_state：整数或 None，随机数的种子值，如果为 None，随机数生成器就是 np.random 使用的 RandomState 实例，默认 None。
- max_leaf_nodes：正整数或 None，最大叶子节点数，通过限制该值可以防止过拟合，None 表示不限制最大的叶子节点数，默认 None。
- min_impurity_decrease：浮点数，设置节点分裂阈值，如果节点分裂导致杂质的减少大于或等于该值，该节点将被分裂，默认 0.0。
- ccp_alpha：非负浮点数，最小剪枝系数，该参数用来限制树过拟合的剪枝参数，ccp_alpha=0 时，决策树不剪枝；ccp_alpha 越大，越多的节点被剪枝，默认 0.0。
- class_weight：字典或字典组成的列表或 None，指定样本各类别的权重，为了防止训练集某些类别的样本过多导致训练的决策树过于偏向这些类别，默认 None。

- GaussianNB
  - GaussianNB：GaussianNB，高斯朴素贝叶斯分类算法。
  - 模型名称：保存模型的名称。
  - prior：类数组结构（形状为(n_classes,)）或 None，类的先验概率，如果指定，则不根据数据调整先验，None 表示自动根据数据来进行计算，默认 None。
  - var_smoothing：浮点数，所有特征的最大方差的一部分，添加到方差中以保持计算稳定性，默认 1e-9。

- KNeighborsRegressor
  - KNeighborsRegressor：KNeighborsRegressor，K 近邻回归。
  - 模型名称：保存模型的名称。
  - n_neighbors：正整数，kneighbors 查询默认使用的邻居数，默认 5。

- weights：选择用于预测的权重函数，包括 uniform 和 distance。其中 uniform 表示统一的权重，即在每一个邻居区域里的点的权重都是一样的，distance 表示权重点等于他们距离的倒数，使用此函数，更近的邻居对于所预测的点的影响更大。默认 uniform。
- Algorithm：选择最近邻所用的算法，包括 auto、ball_tree、kd_tree、brute，其中 auto 表示会根据传递给 fit 的数据自动决定使用最合适的算法，ball_tree 表示使用 BallTree 算法，kd_tree 表示使用 KDTree 算法，brute 表示使用暴力搜索。默认 auto。
- leaf_size：正整数，指定叶子数量。叶子大小传递给 BallTree 或 KDTree，这可能会影响构造和查询的速度，以及存储树所需的内存，最优值取决于问题的性质。默认 30。
- p：整数，闵科夫斯基距离指数或超参数。p=1 等价于使用曼哈顿距离（l1），p=2 等价于欧式距离(l2)，对于任何 p ，使用的是闵科夫斯基距离（l_p）。默认 2。
- metric：选择用于树的距离度量，如果和 p=2 一起使用相当于使用标准欧几里德距离度量，默认 minkowski，目前仅支持实数值向量空间的度量。
- metric_params：字典或 None，度量函数的附加关键字参数。默认 None。
- n_jobs：整数或 None 或-1，表示要为邻居搜索运行的并行作业的数量，None 表示 1，-1 表示所有搜索到的作业数，默认 None。

- RadiusNeighborsRegressor
  - RadiusNeighborsRegressor：RadiusNeighborsRegressor，基于固定半径内邻居的回归。
  - 模型名称：保存模型的名称。
  - Radius：浮点数，radius_neighbors 查询默认使用的参数空间范围，默认 1.0。
  - Weight：选择用于预测的权重函数，包括 uniform 和 distance。其

中 uniform 表示统一的权重，即在每一个邻居区域里的点的权重都是一样的，distance 表示权重点等于他们距离的倒数，使用此函数，更近的邻居对于所预测的点的影响更大。默认 uniform。

- Algorithm：选择最近邻所用的算法，包括 auto、ball_tree、kd_tree、brute，其中 auto 表示会根据传递给 fit 的数据自动决定使用最合适的算法，ball_tree 表示使用 BallTree 算法，kd_tree 表示使用 KDTree 算法，brute 表示使用暴力搜索。默认 auto。
- leaf_size：整数，指定叶子数量。叶子大小传递给 BallTree 或 KDTree，这可能会影响构造和查询的速度，以及存储树所需的内存，最优值取决于问题的性质。默认 30。
- p：整数，闵科夫斯基距离指数或超参数。p=1 等价于使用曼哈顿距离（l1），p=2 等价于欧式距离(l2)，对于任何 p ，使用的是闵科夫斯基距离（l_p）。默认 2。
- metric：选择用于树的距离度量，如果和 p=2 一起使用相当于使用标准欧几里德距离度量，默认 minkowski，目前仅支持实数值向量空间的度量。
- metric_params：字典或 None，度量函数的附加关键字参数。默认 None。
- n_jobs：整数或 None 或-1，表示要为邻居搜索运行的并行作业的数量，None 表示 1，-1 表示所有，默认 None。

- LinearSVR
  - LinearSVR：LinearSVR，线性支持向量回归。
  - 模型名称：保存模型的名称。
  - Epsilon：浮点数，损失函数选择 epsilon-insensitive 时的参数，该参数依赖于目标变量 y 的大小，如果不确定则设置为 0.0，默认 0.1。
  - tol：0 到 1 之间的小数，停止训练的误差值大小，默认 1e-4。
  - C：大于 0 的浮点数，误差项的惩罚参数 C。惩罚是 12 的平方，该值

越大，表示使用的正则化越少，默认 1.0。

- Loss：选择损失函数，epsilon_insensitive（标准 SVR）是 l1 损失，squared_epsilon_insensitive 是 l2 损失，默认 epsilon_insensitive。
- fit_intercept：是否计算该模型的截距，False 表示不会在计算中使用截距，默认 True。
- intercept_scaling：当 self.fit_intercept 为 True 且使用 liblinear 求解器时，实例向量 x 变为[x，self.intercept_scaling]，即具有等于 intercept_scaling 的常量值的“合成”特征被附加到实例向量。 为了减小正则化对合成特征权重（并因此对截距）的影响，必须增加 intercept_scaling。默认 1.0。
- Dual：选择求解对偶优化问题或原始优化问题的算法，True 表示求解对偶优化问题，当样本数大于特征数时，dual=False，默认 True。
- Verbose：整数，是否启用详细输出，默认 0。
- random_state：整数或 None，随机数的种子值，如果为 None，随机数生成器就是 np.random 使用的 RandomState 实例，默认 None。
- max_iter：整数，要运行的最大迭代次数，默认 1000。

- SVR
  - SVR：SVR，Epsilon-支持向量回归，模型中的自由参数为 C 和 epsilon。
  - 模型名称：保存模型的名称。
  - Kernel：选择算法使用的核函数，包含 linear、poly、rbf、sigmoid、precomputed，其中 linear 表示线性函数，poly 表示多项式函数，rbf 表示 RBF 高斯核函数，sigmoid 表示 S 形曲线函数，默认 rbf。
  - Degree：整数，多项式 poly 函数的维度，选择其他核函数时会被忽略，默认 3。
  - gamma：auto 或浮点数或 scale，poly、rbf、sigmoid 核函数参数，

默认 auto。

- coef0：浮点数，核函数的常数项，针对 poly 和 sigmoid 核函数有效，默认 0.0。
- tol：0 到 1 之间的小数，停止训练的误差值大小，默认 1e-3。
- C：大于 0 的浮点数，误差项的惩罚参数 C，默认 1.0。
- epsilon：浮点数，惩罚参数，默认 0.1。
- Shrinking：是否采用 shrinking heuristic 方法，默认 True。
- cache_size：整数，核函数缓存大小（以 MB 为单位），默认 200。
- Verbose：是否启用详细输出，默认 False。
- max_iter：整数，指定在求解器中迭代的硬限制，-1 表示无限制，默认-1。

- NuSVR
  - NuSVR：NuSVR，Nu-支持向量回归，使用参数 nu 代替 SVR 的 epsilon 参数控制支持向量的数量。
  - 模型名称：保存模型的名称。
  - nu:浮点数,训练误差分数的上界和支持向量分数的下界,在区间(0,1.0]内，默认 0.5。
  - C：大于 0 的浮点数，误差项的惩罚参数 C，默认 1.0。
  - Kernel：选择算法使用的核函数，包含 linear、poly、rbf、sigmoid、precomputed，其中 linear 表示线性函数，poly 表示多项式函数，rbf 表示 RBF 高斯核函数，sigmoid 表示 S 形曲线函数，默认 rbf。
  - Degree：正整数，多项式 poly 函数的维度，选择其他核函数时会被忽略，默认 3。
  - gamma：auto 或浮点数或 scale，poly、rbf、sigmoid 核函数参数，默认 auto。
  - coef0：浮点数，核函数的常数项，针对 poly 和 sigmoid 核函数有效，默认 0.0。

- Shrinking：是否采用 shrinking heuristic 方法，默认 True。
- tol：0 到 1 之间的浮点数或科学计数法，停止训练的误差值大小，默认 1e-3。
- cache_size：正整数，核函数缓存大小（以 MB 为单位），默认 200。
- Verbose：是否启用详细输出，默认 False。
- max_iter：整数，指定在求解器中迭代的硬限制，-1 表示无限制，默认-1。

- DecisionTreeRegressor
  - DecisionTreeRegressor：DecisionTreeRegressor，决策树回归。
  - 模型名称：保存模型的名称。
  - Criterion：选择测量分割质量的函数，支持的标准有：均方误差的'squared_error'，方差缩减作为特征选择标准，使用每个终端节点的平均值最小化 L2 损失，'friedman_mse'，表示费尔德曼均方误差算法，'absolute_error'的平均绝对误差，使用每个终端节点的中位数最小化 L1 损失，'poisson'，表示泊松偏差算法。默认'squared_error'。
  - Splitter：选择每个节点的分割策略，包含 best 和 random，best 表示选择最好的特征进行分割;random 表示随机选择特征进行分割，默认 best。
  - max_depth：正整数或 None，设置决策树的最大深度，None 表示展开节点直到所有叶子都是纯的,或者直到所有叶子包含的样本小于 min_samples_split，默认 None。
  - min_samples_split：正整数，设置分割内部节点所需的最小样本数，默认 2，当入参为整数时不小于 2，入参为浮点数时参数范围为(0,1.0]。
  - min_samples_leaf：正整数，设置叶节点所需的最小样本数，默认 1，参数为整数时应不小于 1，参数为浮点数时参数范围为(0,0.5]。
  - min_weight_fraction_leaf：浮点数，叶节点所需的(所有输入样本

的)总权值的最小加权分数。当不提供 sample_weight 时，样本的权重相等，默认 0.0，参数范围为[0.0，0.5]。

- max_features：整数或者浮点数或 None 或 auto 或 sqrt 或 log2，指定寻找最佳分割时要考虑的特性的数量，None 表示将所有特征作为 max_features,auto 表示 max_features=sqrt(n_features)，sqrt 表示 max_features=sqrt(n_features)，log2 表示 max_features=log2(n_features)，默认 None。
- random_state：正整数或 None，随机数的种子值，如果为 None，随机数生成器就是 np.random 使用的 RandomState 实例，默认 None。
- max_leaf_nodes：正整数或 None，最大叶子节点数，通过限制该值可以防止过拟合，None 表示不限制最大的叶子节点数，默认 None。
- min_impurity_decrease：浮点数，设置节点分裂阈值，如果节点分裂导致杂质的减少大于或等于该值，该节点将被分裂，默认 0.0。
- ccp_alpha：非负浮点数，最小剪枝系数，该参数用来限制树过拟合的剪枝参数，ccp_alpha=0 时，决策树不剪枝；ccp_alpha 越大，越多的节点被剪枝，默认 0.0。

- ExtraTreeRegressor
  - ExtraTreeRegressor：ExtraTreeRegressor，随机树回归。
  - 模型名称：保存模型的名称。
  - Criterion：选择测量分割质量的函数，包含 squared_error 和 friedman_mse，squared_error 表示均方差，利用末端节点的平均最小化 l2 损失，mae 表示平均绝对误差，利用每个末端节点的中值最小化 l1 损失，默认 squared_error。
  - Splitter：选择每个节点的分割策略，包含 best 和 random，best 表示选择最好的特征进行分割;random 表示随机选择特征进行分割，默认 best。
  - max_depth：正整数或 None，设置决策树的最大深度，None 表示展开节点直到所有叶子都是纯的,或者直到所有叶子包含的样本小于 m

in_samples_split，默认 None。

- min_samples_split：正整数，设置分割内部节点所需的最小样本数，默认 2,参数为整数时不小于 2,参数为浮点型时参数范围为(0,1.0]。
- min_samples_leaf：正整数，设置叶节点所需的最小样本数，默认 1，参数为整数时应不小于 1，参数为浮点数时参数范围为(0,0.5]。
- min_weight_fraction_leaf：浮点数，叶节点所需的(所有输入样本的)总权值的最小加权分数。当不提供 sample_weight 时，样本的权重相等，默认 0.0，参数范围为[0.0,0.5]。
- max_features：整数或 None 或 auto 或 sqrt 或 log2，指定寻找最佳分割时要考虑的特性的数量，None 表示将所有特征作为 max_features,auto 表示 max_features=sqrt(n_features)，sqrt 表示 max_features=sqrt(n_features)，log2 表示 max_features=log2(n_features)，默认 None。
- random_state：正整数或 None，随机数的种子值，如果为 None，随机数生成器就是 np.random 使用的 RandomState 实例，默认 None。
- min_impurity_decrease：浮点数，设置节点分裂阈值，如果节点分裂导致杂质的减少大于或等于该值，该节点将被分裂，默认 0.0。
- max_leaf_nodes：正整数或 None，最大叶子节点数，通过限制该值可以防止过拟合，None 表示不限制最大的叶子节点数，默认 None。
- ccp_alpha：非负浮点数，最小剪枝系数，该参数用来限制树过拟合的剪枝参数，ccp_alpha=0 时，决策树不剪枝；ccp_alpha 越大，越多的节点被剪枝，默认 0.0。

- ARDRegression
  - ARDRegression：ARDRegression，贝叶斯自相关回归。
  - 模型名称：保存模型的名称。
  - n_iter：正整数，指定最大迭代次数，默认 300。
  - tol：0 到 1 之间的小数，停止训练的误差值大小，如果 w 已经收敛，则停止算法，默认 0.001。

- alpha_1：浮点数，超参数，在 alpha 参数之前的 Gamma 分布的形状参数，默认 1e-6。
- alpha_2：浮点数，超参数：在 alpha 参数之前的 Gamma 分布的反比例参数（速率参数），默认 1e-6。
- lambda_1：浮点数，超参数，在 lambda 参数之前的 Gamma 分布的形状参数，默认 1e-6。
- lambda_2：浮点数，超参数：在 lambda 参数之前的 Gamma 分布的反比例参数（速率参数），默认 1e-6。
- compute_score：True 表示在模型的每一步计算目标函数，False 则不计算，默认 False。
- threshold_lambda：浮点数，从计算中去除(修剪)高精度权值的阈值，默认 10000.0。
- fit_intercept：是否计算该模型的截距，False 表示不会在计算中使用截距，默认 True。
- normalize：当 fit_intercept 设置为 False 时，将忽略该参数。若为真，则回归前对回归量 X 进行归一化处理，默认 False。
- copy_X：True 表示复制 X，False 表示覆盖 X，默认 True。
- verbose：是否启用详细输出，默认 False。

- BayesianRidge
  - BayesianRidge：BayesianRidge，贝叶斯岭回归。
  - 模型名称：保存模型的名称。
  - n_iter：正整数，指定最大迭代次数，默认 300。
  - tol：0 到 1 之间的小数，停止训练的误差值大小，如果 w 已经收敛，则停止算法，默认 0.001。
  - alpha_1：浮点数，超参数，在 alpha 参数之前的 Gamma 分布的形状参数，默认 1e-6。
  - alpha_2：浮点数，超参数：在 alpha 参数之前的 Gamma 分布的反比例参数（速率参数），默认 1e-6。

- lambda_1：浮点数，超参数，在 lambda 参数之前的 Gamma 分布的形状参数，默认 1e-6。
- lambda_2：浮点数，超参数：在 lambda 参数之前的 Gamma 分布的反比例参数（速率参数），默认 1e-6。
- compute_score：True 表示在模型的每一步计算目标函数，False 则不计算，默认 False。
- fit_intercept：是否计算该模型的截距，False 表示不会在计算中使用截距，默认 True。
- normalize：当 fit_intercept 设置为 False 时，将忽略该参数。若为真，则回归前对回归量 X 进行归一化处理，默认 False。
- copy_X：True 表示复制 X，False 表示覆盖 X，默认 True。
- verbose：是否启用详细输出，默认 False。

- ElasticNet
  - ElasticNet：ElasticNet，弹性网络回归算法。
  - 模型名称：保存模型的名称。
  - alpha：浮点数，乘以惩罚项的常数，默认 1.0。
  - l1_ratio：介于 0 到 1 之间的浮点数，弹性网络的混合参数，l1_ratio=0 表示 l2 惩罚，l1_ratio=1 表示 l1 惩罚，0 到 1 之间表示 l1 和 l2 的组合，默认 0.5。
  - fit_intercept：是否计算该模型的截距，False 表示不会在计算中使用截距，默认 True。
  - normalize：当 fit_intercept 设置为 False 时，将忽略该参数。若为真，则回归前对回归量 X 进行归一化处理，默认 False。
  - precompute：是否使用预先计算好的 Gram 矩阵来加速计算，默认 False。
  - max_iter：正整数，要运行的最大迭代次数，默认 1000。
  - copy_X：True 表示复制 X，False 表示覆盖 X，默认 True。
  - tol：浮点数，优化公差，默认 0.0001。

- warm_start：True 表示重用前一个调用的解决方案以适应初始化，False 表示擦除前一个解决方案，默认 False。
- positive：当设置为 True 时，强制系数为正，默认 False。
- random_state：正整数或 None，随机数的种子值，如果为 None，随机数生成器就是 np.random 使用的 RandomState 实例，默认 None。
- selection：random 表示每一次迭代都会更新一个随机系数，cyclic 表示按顺序遍历特性，默认 cyclic。

- Lars
  - Lars：Lars，最小角回归。
  - 模型名称：保存模型的名称。
  - fit_intercept：是否计算该模型的截距，False 表示不会在计算中使用截距，默认 True。
  - verbose：是否启用详细输出，默认 False。
  - normalize：当 fit_intercept 设置为 False 时，将忽略该参数。若为真，则回归前对回归量 X 进行归一化处理，默认 True。
  - precompute：是否使用预先计算好的 Gram 矩阵来加速计算，默认 False。
  - n_nonzero_coefs：正整数，非零系数的目标数，np.inf 暂时不支持，默认 500。
  - eps：浮点数，计算 Cholesky 对角因子的机器精度规范，默认 2.220446049250313e-16。
  - copy_X：True 表示复制 X，False 表示覆盖 X，默认 True。
  - fit_path：True 表示将完整路径存储在 coef_path_属性中，默认 True。
  - random_state：正整数或 None，随机数的种子值，如果为 None，随机数生成器就是 np.random 使用的 RandomState 实例，默认 None。

- Lasso

- Lasso：Lasso，用 l1 先验作为正则化器(即 Lasso)的线性模型。
- 模型名称：保存模型的名称。
- alpha：浮点数，乘以 l1 惩罚项的常数，默认 1.0。
- fit_intercept：是否计算该模型的截距，False 表示不会在计算中使用截距，默认 True。
- normalize：当 fit_intercept 设置为 False 时，将忽略该参数。若为真，则回归前对回归量 X 进行归一化处理，默认 False。
- precompute：是否使用预先计算好的 Gram 矩阵来加速计算，默认 False。
- copy_X：True 表示复制 X，False 表示覆盖 X，默认 True。
- max_iter：正整数，要运行的最大迭代次数，默认 1000。
- tol：浮点数，优化公差，默认 0.0001。
- warm_start：True 表示重用前一个调用的解决方案以适应初始化，False 表示擦除前一个解决方案，默认 False。
- positive：当设置为 True 时，强制系数为正，默认 False。
- random_state：正整数或 None，随机数的种子值，如果为 None，随机数生成器就是 np.random 使用的 RandomState 实例，默认 None。
- selection：random 表示每一次迭代都会更新一个随机系数，cyclic 表示按顺序遍历特性，默认 cyclic。

- LinearRegression
  - LinearRegression：LinearRegression，普通最小二乘线性回归。
  - 模型名称：保存模型的名称。
  - fit_intercept：是否计算该模型的截距，False 表示不会在计算中使用截距，默认 True。
  - normalize：当 fit_intercept 设置为 False 时，将忽略该参数。若为真，则回归前对回归量 X 进行归一化处理，默认 False。
  - copy_X：True 表示复制 X，False 表示覆盖 X，默认 True。
  - n_jobs：正整数或 None 或-1，要用于计算的作业数，None 表示 1，

-1 表示所有的作业数，默认 None。

- AgglomerativeClustering
  - AgglomerativeClustering：AgglomerativeClustering，聚合聚类。
  - 模型名称：保存模型的名称。
  - n_clusters：正整数或 None，要查找的聚类数量，distance_threshold 为 None 时该值必须设置为 None，默认 2。
  - Affinity：选择用于计算 linkage 的度量方法，包含 euclidean、precomputed、cosine、l1、l2 和 manhattan，如果 linkage 为 ward，只能选用 euclidean，默认 euclidean。
  - Memory：用于缓存计算树的输出，None 表示不进行缓存，如果给定一个字符串，即是缓存目录的路径，默认 None。
  - Connectivity：类数组格式或 None，连接矩阵，根据给定的数据结构为每个样本定义相邻的样本，默认 None。
  - compute_full_tree：是否在 n_clusters 处提前停止树的构造，默认 auto。
  - Linkage：选择使用哪种连接标准，决定了观察集之间使用的距离，该算法将合并最小化该准则的簇对，ward 最小化簇合并的方差，average 使用两组观测值距离的平均值，complete 使用两个集合的所有观测值之间的最大距离，single 使用两个集合的所有观测值之间的最小距离，默认 ward。

- KMeans
  - KMeans：KMeans，K 均值聚类。
  - 模型名称：保存模型的名称。
  - n_clusters：正整数，要形成的簇数以及要生成的质心数，默认 8。
  - init：选择初始化方法，k-means++表示智能选择 k 均值聚类的初始聚类中心加快收敛速度，random 表示从初始质心的数据中随机选择 k 个观测值，暂时不支持输入 ndarray，默认 k-means++。

- n_init：正整数，k-means 算法在不同的质心种子下运行的时间，默认 10。
- max_iter：正整数，k-means 算法单次运行的最大迭代次数，默认 300。
- tol：浮点数，惯性收敛的相对公差，默认 1e-4。
- verbose：是否启用详细输出，0 表示不输出，默认 0。
- random_state：正整数或 None，随机数的种子值，如果为 None，随机数生成器就是 np.random 使用的 RandomState 实例，默认 None。
- copy_x：True 表示复制 X，False 表示覆盖 X，默认 True。
- Algorithm：选择 K-means 使用的算法，full 表示经典的 EM-style 算法，elkan 使用三角不等式有更高的效率（不支持稀疏数据），auto 会自动对稀疏数据选择 full，对稠密数据选择 elkan，默认 auto。

- SpectralClustering
  - SpectralClustering：SpectralClustering，谱聚类，将聚类应用于标准化拉普拉斯投影。
  - 模型名称：保存模型的名称。
  - n_clusters：正整数，投影子空间的维数，默认 8。
  - eigen_solver：选择特征值分解策略，默认 None。
  - random_state：正整数或 None，随机数的种子值，如果为 None，随机数生成器就是 np.random 使用的 RandomState 实例，默认 None。
  - n_init：正整数，k-means 算法在不同的质心种子下运行的时间，默认 10。
  - gamma：非负数，rbf、poly、sigmoid、laplacian 和 chi2 的核系数，affinity=nearest_neighbors 时忽略，默认 1.0。
  - affinity：选择构建 affinity 矩阵的方法，nearest_neighbors 通过计算最近邻的图来构造 affinity 矩阵，rbf 利用径向基函数(RBF)核构造 affinity 矩阵，precomputed 将 X 解释为一个预先计算的 af

finity 和矩，默认 rbf。

- n_neighbors：正整数，使用最近邻法构造 affinity 矩阵时要使用的近邻数，当 affinity=rbf 时忽略，默认 10。
- eigen_tol：浮点数，eigen_solver=arpack 是拉普拉斯矩阵特征分解的停止准则，默认 0.0。
- assign_labels：选择用于在嵌入空间中分配标签的策略，默认 kmeans。
- degree：浮点数，多项式核的维度，其他内核忽略，默认 3。
- coef0：浮点数，polynomial 和 sigmoid 核函数的零系数，其他内核忽略，默认 1。
- kernel_params：None，核参数，可忽略，默认 None。
- n_jobs：正整数或 None 或-1，要用于并行的作业数，None 表示 1，-1 表示所有，默认 None。

- DBSCAN
  - DBSCAN：DBSCAN，基于密度的空间聚类，从向量阵列或距离矩阵执行 DBSCAN 聚类。
  - 模型名称：保存模型的名称。
  - eps：正浮点数，两个样本之间的最大距离，其中一个被认为是在另一个的邻域内，默认 0.5。
  - min_samples：正整数，将一个点视为一个核心点的邻域内样本的数量(或总重量)，默认 5。
  - Metric：选择在计算特征数组中实例之间的距离时使用的度量，默认 euclidean。
  - metric_params：字典或 None，度量函数的附加关键字参数。默认 None。
  - Algorithm：选择将被最近邻模块用来计算点距离和找到最近邻的算法，默认 auto。
  - leaf_size：正整数，传递给 BallTree 或 cKDTree 的叶子大小，默

认 30。

- p：浮点数或 None，用来计算点之间距离的 Minkowski 距离的幂，默认 None。
- n_jobs：正整数或 None 或-1，要用于计算的作业数，None 表示 1，-1 表示左右，默认 None。

- MiniBatchKMeans
  - MiniBatchKMeans：MiniBatchKMeans，小批量 K 均值聚类。
  - 模型名称：保存模型的名称。
  - n_clusters：正整数，要形成的簇数以及要生成的质心数，默认 8。
  - init：选择初始化方法，k-means++表示智能选择 k 均值聚类的初始聚类中心加快收敛速度，random 表示从初始质心的数据中随机选择 k 个观测值，暂时不支持输入 ndarray，默认 k-means++。
  - max_iter：正整数，要运行的最大迭代次数，默认 100。
  - batch_size：正整数，最小批次大小，默认 100。
  - verbose：是否启用详细输出，0 表示不输出，默认 0。
  - compute_labels：选择是否在小批量优化收敛到合适的程度后计算整个数据集的标签分配和惯性，默认 True。
  - random_state：正整数或 None，随机数的种子值，如果为 None，随机数生成器就是 np.random 使用的 RandomState 实例，默认 None。
  - tol：浮点数，根据中心位置变化的平滑、方差标准化的相对中心变化来控制早期停止，默认 0.0。
  - max_no_improvement：正整数，根据连续的小批量数量控制提前停止，默认 10。
  - init_size：正整数或 None，初始化大小，通过在数据的随机子集上运行批处理 KMeans 进行初始化，如果为 None，则 init_size= 3 * batch_size，默认 None。
  - n_init：正整数，尝试随机初始化的次数，默认 3。
  - reassignment_ratio：浮点数，控制要重新分配的中心的最大计数

的分数，默认 0.01。

- IsolationForest
  - IsolationForest：IsolationForest，孤立森林。
  - 模型名称：保存模型的名称。
  - n_estimators：正整数，构建多少个 itree，指定该森林中生成的随机树数量，默认 100。
  - max_samples：采样数，auto 是 256，浮点数或者 auto，默认 auto。
  - contamination：浮点数或者'auto'，异常数据占给定的数据集的比例，就是数据集中污染的数量，定义该参数值的作用是在决策函数中定义阈值。如果设置为'auto'，则决策函数的阈值就和论文一样。默认值为'auto'。
  - max_features：正整数，最大特征数，指定从总样本 X 中抽取来训练每棵树 iTree 的属性的数量，默认只使用一个属性，默认为 1。
  - bootstrap：构建 Tree 时，下次是否替换采样，为 True 为替换，则各个树可放回地对训练数据进行采样；为 False 为不替换，即执行不放回的采样，默认 False。
  - n_jobs：正整数或 None，在运行 fit() 和 predict() 函数时并行运行的作业数量。除了在 joblib.parallel_backend 上下文的情况下，None 表示为 1，设置为 -1 则表示使用所有可以使用的处理器，默认 None。
  - random_state：整数或者 None，如果设置为 int 常数，则该 random_state 参数值是用于随机数生成器的种子；如果设置为 None，则该随机数生成器就是使用在 np.random 中 RandomState 实例.默认 0.0
  - verbose：整数，控制树构建过程的冗长性，默认 0。
  - warm_start：当设置为 TRUE 时，重用上一次调用的结果去 fit，添加更多的树到上一次的森林 1 集合中；否则就 fit 一整个新的森林，默认 False。

（二）视觉算法

- 图像分类算法
  - 图像分类算法：图像分类训练。
  - 模型名称：保存模型的名称；当模型名称为空时，保存模型名称默认为选择的算法名称，如 VGG16，则保存为 VGG16；组件用于预测时，该属性没有实际意义。
  - 3D 可视化：是否开启 3D 可视化，3D 可视化仅支持深度学习模型且为 Sequential 式模型，其他类型的模型暂不支持。
  - 分类模型：图像分类算法常用来判断图像内容所属的类别，每张图片输出一个类别。该组件提供了常用的 Resnet、Inception 系列、mobilenet 系列模型，并提供了基于 imagenet 预训练的模型，方便用户 finetune。默认情况下，使用 'VGG16'。
  - weights：None 代表随机初始化，'imagenet'代表加载在 ImageNet 上预训练的权值。
  - input_shape：输入尺寸元组，仅当 include_top=False 时有效，否则输入形状必须是 (244, 244, 3)（对于 channels_last 数据格式），或者 (3, 244, 244)（对于 channels_first 数据格式）。它必须拥有 3 个输入通道，且宽高必须不小于 32。例如 (200, 200, 3) 是一个合法的输入尺寸，默认为 None。
  - pooling：当 include_top 为 False 时，该参数指定了特征提取时的池化方式。None 代表不池化，直接输出最后一层卷积层的输出，该输出是一个四维张量；'avg' 代表全局平均池化（GlobalAveragePooling2D），相当于在最后一层卷积层后面再加一层全局平均池化层，输出是一个二维张量。
  - classes：正整数，图片分类的类别数，仅当 include_top 为 True 并且不加载预训练权值时可用，默认 1000。
  - 迭代次数：训练模型的迭代次数，默认为 100。
  - 损失函数：模型训练的损失函数，暂时不支持 Reduction 函数。
  - 批次大小：一次训练所选取的样本数。

- metrics：metrics 函数，暂时不支持 MeanRelativeError，MeanIoU，PrecisionAtRecall，RecallAtPrecision，SensitivityAtSpecificity 和 SpecificityAtSensitivity。
- 优化器：设置模型优化器。
- 学习率调整策略：设置学习率调整策略：None 表示不设置任何学习策略，学习率为固定值；CosineDecay，余弦衰减策略，与该策略相关的参数有：initial_learning_rate(初始学习率),decay_steps(衰减步数),alpha(最小学习率值作为 initial_learning_rate 的一部分);CosineDecayRestarts，包含重新启动的余弦衰减策略，与该策略相关的参数有：initial_learning_rate(初始学习率),first_decay_steps(要衰减的步数),t_mul(用于导出 i-th 周期内的迭代次数),m_mul(用于导出 i-th 周期的初始学习率),alpha(最小学习率值作为 initial_learning_rate 的一部分);ExponentialDecay，指数衰减策略，与该策略相关的参数有：initial_learning_rate(初始学习率),decay_steps(衰减步数),decay_rate(衰减率),staircase(如果 True 以离散间隔衰减学习率);InverseTimeDecay，反时间衰减策略，与该策略相关的参数有：initial_learning_rate(初始学习率),decay_steps(衰减步数),decay_rate(衰减率),staircase(是否在离散梯度中使用衰减，而不是在连续梯度);PiecewiseConstantDecay，分段常熟衰减策略，与该策略相关的参数有：boundaries(Tensors 或 ints 或 floats 的列表具有严格增加的条目，并且所有元素具有与优化器步骤相同的类型),values(Tensor 或 float 或 int 列表，指定由 boundaries 定义的间隔的值，它应该比 boundaries 多一个元素，并且所有元素都应该具有相同的类型);PolynomialDecay，多项式学习能力衰减策略，与该策略相关的参数有：initial_learning_rate(初始学习率),decay_steps(衰减步数),end_learning_rate(最小的最终学习率),power(多项式的幂),cycle(是否应该循环超过 decay_steps)。
- learning_rate：设置固定学习率，0 到 1 之间的浮点数，默认 0.00

1。

- initial_learning_rate：初始学习率，0 到 1 之间的浮点数，默认 0.1。
- decay_steps：衰减步数，正整数，默认 1。
- alpha：最小学习率值作为 initial_learning_rate 的一部分，0 到 1 之间的浮点数，默认 0.0。
- first_decay_steps：衰减步数，正整数，默认 1。
- t_mul：用于导出 i-th 周期内的迭代次数，非负浮点数，默认 2.0。
- m_mul：用于导出 i-th 周期的初始学习率，非负浮点数，默认 1.0。
- decay_rate：衰减率，非负浮点数，默认 0.5。
- staircase：是否在离散梯度中使用衰减，而不是在连续梯度，默认'False'。
- boundaries：严格递增的由浮点型或者整型组成的列表，默认[100000,110000]。
- values：指定由 boundaries 定义的间隔值，它应该比 boundaries 多一个元素，并且所有元素都应该具有相同的类型，默认[1.0,0.5,0.1]。
- end_learning_rate：最小的最终学习率，非负浮点数，默认 0.5。
- power：多项式的幂，非负浮点数，默认 1.0。
- cycle：是否应该循环超过 decay_steps，默认'False'。
- rho：Adadelta 梯度平方移动均值的衰减率，非负浮点数，默认 0.95。
- epsilon：epsilon，防止除 0 错误，0 到 1 之间的浮点数或者科学计数法，默认 1e-07。
- initial_accumulator_value：初始梯度累加和，非负浮点数，默认 0.1。
- beta_1：0 到 1 之间，非常接近 1 的浮点数，默认 0.9。
- beta_2：0 到 1 之间，非常接近 1 的浮点数，默认 0.999。
- amsgrad：是否应用此算法的 AMSGrad 变种，默认'False'。

- centered：如果 True，则通过梯度的估计方差对梯度进行归一化；如果为 False，则通过非居中的第二时刻。将此设置为 True 可能有助于训练，但在计算和内存方面稍微贵一些，默认'False'。
- momentum：用于加速 SGD 在相关方向上前进，并抑制震荡，非负浮点数，默认 0.0。
- nesterov：是否使用 Nesterov 动量，默认'False'。
- beta_2：浮点值，必须小于或等于零，控制在训练期间学习率如何降低，使用零表示固定的学习率，默认-0.5。
- l1_regularization_strength：浮点值，必须大于或等于零。默认为 0.0。
- l2_regularization_strength：浮点值，必须大于或等于零。默认为 0.0。
- l2_shrinkage_regularization_strength：浮点值，必须大于或等于零。这与上面的 L2 不同，因为上面的 L2 是一个稳定惩罚，而这个 L2 收缩是一个幅度惩罚。当输入稀疏时，收缩只会发生在活动权重上。默认为 0.0。
- beta：浮点值，beta 值。默认为 0.0。

- ssd
  - ssd：ssd 目标识别算法。
  - 模型名称：保存模型的名称；当模型名称为空时，保存模型名称默认为选择的算法名称，如 ssd，模型名称保存为 ssd;组件用于预测时，该属性没有实际意义。
  - 3D 可视化：是否开启 3D 可视化，3D 可视化仅支持深度学习模型且为 Sequential 式模型，其他类型的模型暂不支持。
  - weights：None 代表随机初始化，'VOC2007' 代表加载在 VOC2007 上预训练的权值。
  - 迭代次数：训练模型的迭代次数，默认为 100。
  - 批次大小：一次训练所选取的样本数。

- learningrate：学习率，0 到 1 之间的浮点数，默认 0.001。

- yolov3
  - yolov3：yolov3 目标识别算法。
  - 模型名称：保存模型的名称；当模型名称为空时，保存模型名称默认为选择的算法名称，如 yolov3，则模型名称保存为 yolov3;组件用于预测时，该属性没有实际意义。
  - 3D 可视化：是否开启 3D 可视化，3D 可视化仅支持深度学习模型且为 Sequential 式模型，其他类型的模型暂不支持。
  - weights：None 代表随机初始化，'coco' 代表加载在 coco 上预训练的权值。
  - 迭代次数：训练模型的迭代次数，默认为 100。
  - 批次大小：一次训练所选取的样本数。
  - learningrate：学习率，0 到 1 之间的浮点数，默认 0.001。

- yolov4
  - yolov4：yolov4 目标识别算法。
  - 模型名称：保存模型的名称；当模型名称为空时，保存模型名称默认为选择的算法名称，如 yolov4，模型名称保存为 yolov4;组件用于预测时，该属性没有实际意义。
  - 3D 可视化：是否开启 3D 可视化，3D 可视化仅支持深度学习模型且为 Sequential 式模型，其他类型的模型暂不支持。
  - weights：None 代表随机初始化，'coco' 代表加载在 coco 上预训练的权值。
  - 迭代次数：训练模型的迭代次数，默认为 100。
  - 批次大小：一次训练所选取的样本数。
  - learningrate：学习率，0 到 1 之间的浮点数，默认 0.001。

- yolov5

- yolov5:yolov5 目标识别算法。
- 模型名称：保存模型的名称；当模型名称为空时，保存模型名称默认为选择的算法名称，如 yolov5，模型名称保存为 yolov5；组件用于预测时，该属性没有实际意义。
- 3D 可视化：是否开启 3D 可视化，3D 可视化仅支持深度学习模型且为 Sequential 式模型，其他类型的模型暂不支持。
- weights：None 代表随机初始化，'coco' 代表加载在 coco 上预训练的权值。
- backbone：表示使用的 yolov5 的版本。
- 迭代次数：训练模型的迭代次数，默认为 100。
- 批次大小：一次训练所选取的样本数。
- learningrate：学习率，0 到 1 之间的浮点数，默认 0.001。

- m2det
  - m2det：m2det 目标识别算法。
  - 模型名称：保存模型的名称；当模型名称为空时，保存模型名称默认为选择的算法名称，如 m2det，模型名称保存为 m2det；组件用于预测时，该属性没有实际意义。
  - 3D 可视化：是否开启 3D 可视化，3D 可视化仅支持深度学习模型且为 Sequential 式模型，其他类型的模型暂不支持。
  - weights：None 代表随机初始化，'VOC2007' 代表加载在 VOC2007 上预训练的权值。
  - 迭代次数：训练模型的迭代次数，默认为 100。
  - 批次大小：一次训练所选取的样本数。
  - learningrate：学习率，0 到 1 之间的浮点数，默认 0.001。

- faster_rcnn
  - faster_rcnn：faster_rcnn 目标识别算法。
  - 模型名称：保存模型的名称；当模型名称为空时，保存模型名称默

认为选择的算法名称，如 faster_rcnn，模型名称保存为 faster_rcnn；组件用于预测时，该属性没有实际意义。

- 3D 可视化：是否开启 3D 可视化，3D 可视化仅支持深度学习模型且为 Sequential 式模型，其他类型的模型暂不支持。
- weights：None 代表随机初始化，'VOC2007' 代表加载在 VOC2007 上预训练的权值。
- weights：resnet50 代表使用 resnet50 主干网络训练，'vgg' 代表使用 vgg 主干网络。
- 迭代次数：训练模型的迭代次数，默认为 100。
- 批次大小：一次训练所选取的样本数。
- learningrate：学习率，0 到 1 之间的浮点数，默认 0.001。

- centernet
  - centernet：centernet 目标识别算法。
  - 模型名称：保存模型的名称；当模型名称为空时，保存模型名称默认为选择的算法名称，如 centernet，模型名称保存为 centernet；组件用于预测时，该属性没有实际意义。
  - 3D 可视化：是否开启 3D 可视化，3D 可视化仅支持深度学习模型且为 Sequential 式模型，其他类型的模型暂不支持。
  - weights：None 代表随机初始化，'VOC2007' 代表加载在 VOC2007 上预训练的权值。
  - backbone：表示使用的 centernet 的版本。
  - 迭代次数：训练模型的迭代次数，默认为 100。
  - 批次大小：一次训练所选取的样本数。
  - learningrate：学习率，0 到 1 之间的浮点数，默认 0.001。

- maskrcnn
  - maskrcnn：maskrcnn 图像实例分割算法。
  - 模型名称：保存模型的名称；当模型名称为空时，保存模型名称默

认为选择的算法名称，如 maskrcnn，模型名称保存为 maskrcnn；组件用于预测时，该属性没有实际意义。

- 3D 可视化：是否开启 3D 可视化，3D 可视化仅支持深度学习模型且为 Sequential 式模型，其他类型的模型暂不支持。
- weights：None 代表随机初始化，'coco' 代表加载在 coco 上预训练的权值。
- 迭代次数：训练模型的迭代次数，默认为 100。
- 批次大小：一次训练所选取的样本数。
- learningrate：学习率，0 到 1 之间的浮点数，默认 0.001。

- Unet
  - Unet：Unet 图像语义分割算法。
  - 模型名称：保存模型的名称；当模型名称为空时，保存模型名称默认为选择的算法名称，如 Unet，模型名称保存为 Unet；组件用于预测时，该属性没有实际意义。
  - 3D 可视化：是否开启 3D 可视化，3D 可视化仅支持深度学习模型且为 Sequential 式模型，其他类型的模型暂不支持。
  - backbone：unet 图像分割模型使用的 backbone。默认情况下，使用 'vgg16'。
  - weights：None 代表随机初始化，'CamVid' 代表加载在 CamVid 上预训练的权值。
  - CLASSES：需要训练的类别名称，原始模型使用的是 CamVid 数据集总类别数为 33，分别为 ['Animal','Archway','Bicyclist','Bridge','Building','Car','CartLuggagePram','Child','Column_Pole','Fence','LaneMkgsDriv','LaneMkgsNonDriv','Misc_Text','MotorcycleScooter','OtherMoving','ParkingBlock','Pedestrian','Road','RoadShoulder','Sidewalk','SignSymbol','Sky','SUVPickupTruck','TrafficCone','TrafficLight','Train','Tree','

Truck_Bus','Tunnel','VegetationMisc','Void','Wall', 'unlabelled']，该属性所输名称应该为该类别的子集 ，默认['Car', 'Pedestrian']。

- 迭代次数：训练模型的迭代次数，默认为 100。
- 批次大小：一次训练所选取的样本数。
- learningrate：学习率，0 到 1 之间的浮点数，默认 0.001。

- FPN
  - FPN：FPN 图像语义分割算法。
  - 模型名称：保存模型的名称；当模型名称为空时，保存模型名称默认为选择的算法名称，如 FPN，模型名称保存为 FPN；组件用于预测时，该属性没有实际意义。
  - 3D 可视化：是否开启 3D 可视化，3D 可视化仅支持深度学习模型且为 Sequential 式模型，其他类型的模型暂不支持。
  - backbone：FPN 图像分割模型使用的 backbone。默认情况下，使用 'vgg16'。
  - weights：None 代表随机初始化，'CamVid' 代表加载在 CamVid 上预训练的权值。
  - CLASSES：需要训练的类别名称，原始模型使用的是 CamVid 数据集总类别数为 33，分别为 ['Animal','Archway','Bicyclist','Bridge','Building','Car','CartLuggagePram','Child','Column_Pole','Fence','LaneMkgsDriv','LaneMkgsNonDriv','Misc_Text','MotorcycleScooter','OtherMoving','ParkingBlock','Pedestrian','Road','RoadShoulder','Sidewalk','SignSymbol','Sky','SUVPickupTruck','TrafficCone','TrafficLight','Train','Tree','Truck_Bus','Tunnel','VegetationMisc','Void','Wall', 'unlabelled']，该属性所输名称应该为该类别的子集 ，默认['Car', 'Pedestrian']。

- 迭代次数：训练模型的迭代次数，默认为 100。
- 批次大小：一次训练所选取的样本数。
- learningrate：学习率，0 到 1 之间的浮点数，默认 0.001。

- PSPNet
  - PSPNet：PSPNet 图像语义分割算法。
  - 模型名称：保存模型的名称；当模型名称为空时，保存模型名称默认为选择的算法名称，如 PSPNet，模型名称保存为 PSPNet；组件用于预测时，该属性没有实际意义。
  - 3D 可视化：是否开启 3D 可视化，3D 可视化仅支持深度学习模型且为 Sequential 式模型，其他类型的模型暂不支持。
  - backbone：PSPNet 图像分割模型使用的 backbone。默认情况下，使用 'vgg16'。
  - weights：None 代表随机初始化，'CamVid' 代表加载在 CamVid 上预训练的权值。
  - CLASSES：需要训练的类别名称，原始模型使用的是 CamVid 数据集总类别数为 33，分别为 ['Animal','Archway','Bicyclist','Bridge','Building','Car','CartLuggagePram','Child','Column_Pole','Fence','LaneMkgsDriv','LaneMkgsNonDriv','Misc_Text','MotorcycleScooter','OtherMoving','ParkingBlock','Pedestrian','Road','RoadShoulder','Sidewalk','SignSymbol','Sky','SUVPickupTruck','TrafficCone','TrafficLight','Train','Tree','Truck_Bus','Tunnel','VegetationMisc','Void','Wall', 'unlabelled']，该属性所输名称应该为该类别的子集 ，默认['Car', 'Pedestrian']。
  - 迭代次数：训练模型的迭代次数，默认为 100。
  - 批次大小：一次训练所选取的样本数。
  - learningrate：学习率，0 到 1 之间的浮点数，默认 0.001。

- Linknet
  - Linknet：Linknet 图像语义分割算法。
  - 模型名称：保存模型的名称；当模型名称为空时，保存模型名称默认为选择的算法名称，如 Linknet，模型名称保存为 Linknet；组件用于预测时，该属性没有实际意义。
  - 3D 可视化：是否开启 3D 可视化，3D 可视化仅支持深度学习模型且为 Sequential 式模型，其他类型的模型暂不支持。
  - backbone：Linknet 图像分割模型使用的 backbone。默认情况下，使用 'vgg16'。
  - weights：None 代表随机初始化，'CamVid' 代表加载在 CamVid 上预训练的权值。
  - CLASSES：需要训练的类别名称，原始模型使用的是 CamVid 数据集总类别数为 33，分别为 ['Animal','Archway','Bicyclist','Bridge','Building','Car','CartLuggagePram','Child','Column_Pole','Fence','LaneMkgsDriv','LaneMkgsNonDriv','Misc_Text','MotorcycleScooter','OtherMoving','ParkingBlock','Pedestrian','Road','RoadShoulder','Sidewalk','SignSymbol','Sky','SUVPickupTruck','TrafficCone','TrafficLight','Train','Tree','Truck_Bus','Tunnel','VegetationMisc','Void','Wall', 'unlabelled']，该属性所输名称应该为该类别的子集 ，默认['Car', 'Pedestrian']。
  - 迭代次数：训练模型的迭代次数，默认为 100。
  - 批次大小：一次训练所选取的样本数。
  - learningrate：学习率，0 到 1 之间的浮点数，默认 0.001。

- FaceNet
  - FaceNet：FaceNet 人脸识别算法。
  - 模型名称：保存模型的名称；当模型名称为空时，保存模型名称默认为选择的算法名称，如 Linknet，模型名称保存为 Linknet；组件

用于预测时，该属性没有实际意义。

- 3D 可视化：是否开启 3D 可视化，3D 可视化仅支持深度学习模型且为 Sequential 式模型，其他类型的模型暂不支持。
- backbone：FaceNet 人脸识别模型使用的 backbone。默认情况下，使用 'mobilenet'。
- weights：None 代表随机初始化，'CASIA-WebFace' 代表加载在 CASIA-WebFace 上预训练的权值。
- 迭代次数：训练模型的迭代次数，默认为 100。
- 批次大小：一次训练所选取的样本数，需要是 3 的倍数，默认 24。
- 优化器：设置模型优化器。
- learningrate：学习率，0 到 1 之间的浮点数，默认 0.001。

- humanpose
  - humanpose：人体关键点检测算法。
  - 模型名称：保存模型的名称；当模型名称为空时，保存模型名称默认为选择的算法名称，如 humanpose，模型名称保存为 humanpose；组件用于预测时，该属性没有实际意义。
  - 3D 可视化：是否开启 3D 可视化，3D 可视化仅支持深度学习模型且为 Sequential 式模型，其他类型的模型暂不支持。
  - weights：None 代表随机初始化，'coco' 代表加载在 coco 上预训练的权值。
  - 迭代次数：训练模型的迭代次数，默认为 100。
  - 批次大小：一次训练所选取的样本数。
  - learningrate：学习率，0 到 1 之间的浮点数，默认 0.001。

（三）仿真回归算法

- 基于网格和工况预测仿真云图结果的模型
  - meshgraphnets：基于网格和工况预测仿真云图结果的模型。

- 模型名称：保存模型的名称；当模型名称为空时，保存模型名称默认为选择的算法名称，如 meshgraphnets，模型名称保存为 meshgraphnets；组件用于预测时，该属性没有实际意义。
- 3D 可视化：是否开启 3D 可视化，3D 可视化仅支持深度学习模型且为 Sequential 式模型，其他类型的模型暂不支持。
- weights：None 代表不加载预训练权重，'w1' 代表加载在 w1 上预训练的权值。
- 迭代次数：训练模型的迭代次数。
- 批次大小：一次训练所选取的样本数，过大可能造成内存不足的问题。
- 潜在向量宽度：潜在向量或者 embedding 的宽度。
- MLP 层数：多层感知器（MLP）的层数。
- 信息传递次数：图神经网络节点间信息传递次数。
- 前向传播步数：模型在训练开始时仅进行前向传播的步数。
- 最大 adapt 次数：正则化层进行 adapt 的最大次数，建议为 batch 数的整数倍。
- 初始学习率：多项式学习率衰减算法初始时的学习率，0 到 1 之间的浮点数。
- 结束学习率：多项式学习率衰减算法结束时的学习率，0 到 1 之间的浮点数。
- 递减次数：多项式学习率衰减算法的递减次数。
- 多项式的幂：多项式学习率衰减算法的幂。
- ε :Adam 优化器的 epsilon，0 到 1 之间的浮点数。

- Sequential（序贯模型）
  - 名称(name)：字符串，作为输入层下的第一层，用于指定模型框架。

- Dense（全连接层）
  - 名称（name）：输入框，字符串，为该 Dense 层指定名称，作为其

标识，默认 Dense。

- 神经元数（units）：输入框，整型数值，定义该层的神经元数量，即输出空间维度，默认 64。
- 激活函数(activation)：选择框，为该层选择激活函数，选项包括 relu、tanh、sigmoid、linear、softmax、softplus、softsign、hard_sigmoid、exponential、None，选 None 表示不指定激活函数，默认选择 None。
- 输入维度(Input Shape)：元组或整数类型输入框，n 维张量，定义输入数据的维度，该层作为 Sequential 层下的第一层时，需根据明确指定输入维度(batch_size, ..., input_dim)，输入为二维时输入维度为(batch_size, input_dim)，非第一层时输入 None，默认 None。
- 阈值(use_bias)：选择框（True 或 False），布尔型，定义是否使用偏置向量，默认 True。
- 内核初始化（kernel_initializer）：选择框，定义内核权值矩阵的初始化方法，选项包括 initializer、random_normal、Random_uniform、truncated_normal、identity、lecun_uniform、lecun_normal、orthogonal、zeros、glorot_normal、glorot_uniform、he_normal、he_uniform、ones，默认 glorot_uniform。
- 阈值初始化（bias_initializer）：选择框，定义偏置向量的初始化方法，选项内容同内核初始化，默认 zeros。
- 内核正则化(kernel_regularizer)：输入框，字符串，定义内核权值矩阵的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01, l2=0.01)或 None，默认 None。
- 阈值正则化（bias_regularizer）：输入框，字符串，定义偏置向量的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01, l2=0.01)或 None，默认 None。
- 活动正则化(activity_regularizer)：输入框，字符串，定义层的输出的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01, l2=

0.01)或 None，默认 None。

- 内核约束(kernel_constraint)：选择框，定义权值矩阵的约束函数，选项内容包括 MaxNorm、NonNeg、UnitNorm、MinMaxMorm 或 None，默认 None。
- 阈值约束(bias_constraint)：选择框，定义偏置向量的约束函数，选项内容包括 MaxNorm、NonNeg、UnitNorm、MinMaxMorm 或 None，默认 None。

- Conv1D（一维卷积层）
  - 名称(name)：字符串，为该 Conv1D 层指定名称(name)，作为其标识，默认 Conv1D；
  - 滤波器数(filters)：输入框，整型数值，定义卷积核的数目，即输出的维度，默认 32；
  - 卷积核大小(kernel_size)：输入框，整数或由单个整数构成的列表或元组，定义卷积核的空域或时域窗口的长度，默认 2；
  - 步幅(strides)：输入框，整数或由单个整数构成的列表或元组，定义卷积的步长，默认 1；
  - 填充(padding)：选择框，定义补 0 策略，包括 valid、same 和 causal，valid 表示只进行有效的卷积，即对边界数据不处理，same 表示保留边界处的卷积结果，使输出形状和输入形状相同，causal 表示将产生因果（膨胀的）卷积，即 output[t]不依赖于 input[t+1:]，当对不能违反时间顺序的时序信号建模时有用，默认 valid；
  - 数据格式(data_format)：选择框，包括 channels_last 和 channels_first，定义输入中维度的顺序，channels_last 对应输入形状为（batch, steps, channels），channels_first 对应输入形状为（batch, channels, steps），默认 channels_last；
  - 膨胀率(dilation_rate)：输入框，整数或由单个整数构成的列表或元组，定义膨胀卷积的膨胀率，默认 1；
  - 激活函数(activation)：选择框，为该层选择激活函数，选项包括 r

elu、tanh、sigmoid、linear、softmax、softplus、softsign、hard_sigmoid、exponential、None，选 None 表示不指定激活函数，默认选择 None；

- 阈值(use_bias)：选择框，布尔型，定义是否使用偏置项，默认 True；
- 内核初始化（kernel_initializer）：选择框，定义内核权值矩阵的初始化方法，选项包括 initializer、random_normal、Random_uniform、truncated_normal、identity、lecun_uniform、lecun_normal、orthogonal、zeros、glorot_normal、glorot_uniform、he_normal、he_uniform、ones，默认 glorot_uniform；
- 阈值初始化（bias_initializer）：选择框，定义偏置向量的初始化方法，选项内容同内核初始化，默认 zeros；
- 内核正则化(kernel_regularizer)：输入框，字符串，定义内核权值矩阵的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01, l2=0.01)或 None，默认 None；
- 阈值正则化（bias_regularizer）：输入框，字符串，定义偏置向量的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01, l2=0.01)或 None，默认 None；
- 活动正则化(activity_regularizer)：输入框，字符串，定义层的输出的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01, l2=0.01)或 None，默认 None；
- 内核约束(kernel_constraint)：选择框，定义权值矩阵的约束函数，选项内容包括 MaxNorm、NonNeg、UnitNorm、MinMaxMorm 或 None，默认 None；
- 阈值约束(bias_constraint)：选择框，定义偏置向量的约束函数，选项内容包括 MaxNorm、NonNeg、UnitNorm、MinMaxMorm 或 None，默认 None；
- 输入维度(input_shape)：输入框，整数元组，定义输入数据的维度，该层作为 Sequential 层下的第一层时，需指定输入维度，例如(10,

128)代表一个长为 10 的序列，序列中每个信号为 128 向量，而(None, 128)代表变长的 128 维向量序列，非第一层时输入 None，默认 None。

- Conv2D（二维卷积层）
  - 名称(name)：输入框，字符串，为该 Conv2D 层指定名称，作为其标识，默认 Conv2D。
  - 滤波器数(filters)：输入框，整型数值，定义卷积核的数目，即输出的维度，默认 32。
  - 卷积核大小(kernel_size)：输入框，单个整数或由两个整数构成的列表或元组，定义卷积核宽度和长度，默认（2,2）。
  - 步幅(strides)：输入框，单个整数或由两个整数构成的列表或元组，定义卷积的步长，默认（1,1）。
  - 填充(padding)：选择框，定义补 0 策略，包括 valid 和 same，valid 表示只进行有效的卷积，即对边界数据不处理，same 表示保留边界处的卷积结果，使输出形状和输入形状相同，默认 valid。
  - 数据格式(data_format)：选择框，包括 channels_last 和 channels_first，定义输入中维度的顺序，channels_last 对应输入形状为（batch, height, width, channels），channels_first 对应输入形状为（batch, channels, height, width），默认 channels_last。
  - 膨胀率(dilation_rate)：输入框，单个整数或由两个整数构成的列表或元组，定义膨胀卷积的膨胀率，默认（1，1）。
  - 激活函数(activation)：选择框，为该层选择激活函数，选项包括 relu、tanh、sigmoid、linear、softmax、softplus、softsign、hard_sigmoid、exponential、None，选 None 表示不指定激活函数，默认选择 None。
  - 阈值(use_bias)：选择框，布尔型，定义是否使用偏置项，默认 True。

- 内核初始化（kernel_initializer）：选择框，定义内核权值矩阵的初始化方法，选项包括 initializer、random_normal、Random_uniform、truncated_normal、identity、lecun_uniform、lecun_normal、orthogonal、zeros、glorot_normal、glorot_uniform、he_normal、he_uniform、ones，默认 glorot_uniform。
- 阈值初始化（bias_initializer）：选择框，定义偏置向量的初始化方法，选项内容同内核初始化，默认 zeros。
- 内核正则化(kernel_regularizer)：输入框，字符串，定义内核权值矩阵的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01, l2=0.01)或 None，默认 None。
- 阈值正则化（bias_regularizer）：输入框，字符串，定义偏置向量的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01, l2=0.01)或 None，默认 None。
- 活动正则化(activity_regularizer)：输入框，字符串，定义层的输出的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01, l2=0.01)或 None，默认 None。
- 内核约束(kernel_constraint)：选择框，定义权值矩阵的约束函数，选项内容包括 MaxNorm、NonNeg、UnitNorm、MinMaxMorm 或 None，默认 None。
- 阈值约束(bias_constraint)：选择框，定义偏置向量的约束函数，选项内容包括 MaxNorm、NonNeg、UnitNorm、MinMaxMorm 或 None，默认 None。
- 输入维度(Input Shape)：输入框，整数元组，定义输入数据的维度，该层作为 Sequential 层下的第一层时，需指定输入维度，例如 input_shape = (128,128,3)代表 128*128 的彩色 RGB 图像（data_format='channels_last'），非第一层时输入 None，默认 None。

- Conv2DTranspose（转置卷积层）
  - 名称(name)：字符串，为该 Conv2DTranspose 层指定名称(name)，

作为其标识，默认 Conv2DTranspose。

- 滤波器数(filters)：输入框，整型数值，定义卷积核的数目，即输出的维度，默认 32。
- 卷积核大小(kernel_size)：输入框，单个整数或由两个整数构成的列表或元组，定义卷积核宽度和长度，默认（2,2）。
- 步幅(strides)：输入框，单个整数或由两个整数构成的列表或元组，定义卷积的步长，默认（1,1）。
- 填充(padding)：选择框，定义补 0 策略，包括 valid 和 same，valid 表示只进行有效的卷积，即对边界数据不处理，same 表示保留边界处的卷积结果，使输出形状和输入形状相同，默认 valid。
- 输出填充(output_padding)：输入框，单个整数或由两个整数构成的列表或元组，指定沿输出张量的高度和宽度的填充量，沿给定维度的输出填充量必须小于沿同一维度的步幅，默认 None，表示输出形状根据网络自动推算。
- 数据格式(data_format)：选择框，包括 channels_last 和 channels_first，定义输入中维度的顺序，channels_last 对应输入形状为（batch, height, width, channels），channels_first 对应输入形状为（batch, channels, height, width），默认 channels_last。
- 膨胀率(dilation_rate)：输入框，单个整数或由两个整数构成的列表或元组，定义膨胀卷积的膨胀率，默认（1，1）。
- 激活函数(activation)：选择框，为该层选择激活函数，选项包括 relu、tanh、sigmoid、linear、softmax、softplus、softsign、hard_sigmoid、exponential、None，选 None 表示不指定激活函数，默认选择 None。
- 阈值(use_bias)：选择框，布尔型，定义是否使用偏置项，默认 True。
- 内核初始化（kernel_initializer）：选择框，定义内核权值矩阵的初始化方法，选项包括 initializer、random_normal、Random_u

niform、truncated_normal、identity、lecun_uniform、lecun_normal、orthogonal、zeros、glorot_normal、glorot_uniform、he_normal、he_uniform、ones，默认 glorot_uniform。

- 阈值初始化（bias_initializer）：选择框，定义偏置向量的初始化方法，选项内容同内核初始化，默认 zeros。
- 内核正则化(kernel_regularizer)：输入框，字符串，定义内核权值矩阵的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01, l2=0.01)或 None，默认 None。
- 阈值正则化（bias_regularizer）：输入框，字符串，定义偏置向量的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01, l2=0.01)或 None，默认 None。
- 活动正则化(activity_regularizer)：输入框，字符串，定义层的输出的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01, l2=0.01)或 None，默认 None。
- 内核约束(kernel_constraint)：选择框，定义权值矩阵的约束函数，选项内容包括 MaxNorm、NonNeg、UnitNorm、MinMaxMorm 或 None，默认 None。
- 阈值约束(bias_constraint)：选择框，定义偏置向量的约束函数，选项内容包括 MaxNorm、NonNeg、UnitNorm、MinMaxMorm 或 None，默认 None。
- 输入维度(Input Shape)：输入框，整数元组，定义输入数据的维度，该层作为 Sequential 层下的第一层时，需指定输入维度，例如 input_shape = (128,128,3)代表 128*128 的彩色 RGB 图像（data_format='channels_last'），非第一层时输入 None，默认 None。

- Conv3D（三维卷积层）
  - 名称(name)：字符串，为该 Conv3D 层指定名称，作为其标识，默认 Conv3D。
  - 滤波器数(filters)：输入框，整型数值，定义卷积核的数目，即输

出的维度，默认 32。

- 卷积核大小(kernel_size)：输入框，单个整数或由 3 个整数构成的列表或元组，定义卷积核深度、高度和宽度，默认（2,2,2）。
- 步幅(strides)：输入框，单个整数或由 3 个整数构成的列表或元组，定义卷积沿每个方向的步长，默认（1,1,1）。
- 填充(padding)：选择框，定义补 0 策略，包括 valid 和 same，valid 表示只进行有效的卷积，即对边界数据不处理，same 表示保留边界处的卷积结果，使输出形状和输入形状相同，默认 valid。
- 数据格式(data_format)：选择框，包括 channels_last 和 channels_first，定义输入中维度的顺序，channels_last 对应输入形状为（batch, spatial_dim1, spatial_dim2, spatial_dim3, channels），channels_first 对应输入形状为（batch, channels, spatial_dim1, spatial_dim2, spatial_dim3），默认 channels_last。
- 膨胀率(dilation_rate)：输入框，单个整数或由 3 个整数构成的列表或元组，定义膨胀卷积的膨胀率，默认（1,1,1）。
- 激活函数(activation)：选择框，为该层选择激活函数，选项包括 relu、tanh、sigmoid、linear、softmax、softplus、softsign、hard_sigmoid、exponential、None，选 None 表示不指定激活函数，默认选择 None。
- 阈值(use_bias)：选择框，布尔型，定义是否使用偏置项，默认 True。
- 内核初始化（kernel_initializer）：选择框，定义内核权值矩阵的初始化方法，选项包括 initializer、random_normal、Random_uniform、truncated_normal、identity、lecun_uniform、lecun_normal、orthogonal、zeros、glorot_normal、glorot_uniform、he_normal、he_uniform、ones，默认 glorot_uniform。
- 阈值初始化（bias_initializer）：选择框，定义偏置向量的初始化方法，选项内容同内核初始化，默认 zeros。
- 内核正则化(kernel_regularizer)：输入框，字符串，定义内核权

值矩阵的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01, l2=0.01)或 None，默认 None。

- 阈值正则化（bias_regularizer）：输入框，字符串，定义偏置向量的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01, l2=0.01)或 None，默认 None。
- 活动正则化(activity_regularizer)：输入框，字符串，定义层的输出的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01, l2=0.01)或 None，默认 None。
- 内核约束(kernel_constraint)：选择框，定义权值矩阵的约束函数，选项内容包括 MaxNorm、NonNeg、UnitNorm、MinMaxMorm 或 None，默认 None。
- 阈值约束(bias_constraint)：选择框，定义偏置向量的约束函数，选项内容包括 MaxNorm、NonNeg、UnitNorm、MinMaxMorm 或 None，默认 None。
- 输入维度(Input Shape)：输入框，整数元组，定义输入数据的维度，该层作为 Sequential 层下的第一层时，需指定输入维度，例如 input_shape = (10,128,128,3)代表对 10 帧 128*128 的彩色 RGB 图像进行卷积（data_format='channels_last'），非第一层时输入 None，默认 None。

- Conv3D（三维卷积层）
  - 名称(name)：字符串，为该 Conv3D 层指定名称，作为其标识，默认 Conv3D。
  - 滤波器数(filters)：输入框，整型数值，定义卷积核的数目，即输出的维度，默认 32。
  - 卷积核大小(kernel_size)：输入框，单个整数或由 3 个整数构成的列表或元组，定义卷积核深度、高度和宽度，默认（2,2,2）。
  - 步幅(strides)：输入框，单个整数或由 3 个整数构成的列表或元组，定义卷积沿每个方向的步长，默认（1,1,1）。

- 填充(padding)：选择框，定义补 0 策略，包括 valid 和 same，valid 表示只进行有效的卷积，即对边界数据不处理，same 表示保留边界处的卷积结果，使输出形状和输入形状相同，默认 valid。
- 数据格式(data_format)：选择框，包括 channels_last 和 channels_first，定义输入中维度的顺序，channels_last 对应输入形状为（batch, spatial_dim1, spatial_dim2, spatial_dim3, channels），channels_first 对应输入形状为（batch, channels, spatial_dim1, spatial_dim2, spatial_dim3），默认 channels_last。
- 膨胀率(dilation_rate)：输入框，单个整数或由 3 个整数构成的列表或元组，定义膨胀卷积的膨胀率，默认（1,1,1）。
- 激活函数(activation)：选择框，为该层选择激活函数，选项包括 relu、tanh、sigmoid、linear、softmax、softplus、softsign、hard_sigmoid、exponential、None，选 None 表示不指定激活函数，默认选择 None。
- 阈值(use_bias)：选择框，布尔型，定义是否使用偏置项，默认 True。
- 内核初始化（kernel_initializer）：选择框，定义内核权值矩阵的初始化方法，选项包括 initializer、random_normal、Random_uniform、truncated_normal、identity、lecun_uniform、lecun_normal、orthogonal、zeros、glorot_normal、glorot_uniform、he_normal、he_uniform、ones，默认 glorot_uniform。
- 阈值初始化（bias_initializer）：选择框，定义偏置向量的初始化方法，选项内容同内核初始化，默认 zeros。
- 内核正则化(kernel_regularizer)：输入框，字符串，定义内核权值矩阵的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01, l2=0.01)或 None，默认 None。
- 阈值正则化（bias_regularizer）：输入框，字符串，定义偏置向量的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01, l2=0.01)或 None，默认 None。

- 活动正则化(activity_regularizer)：输入框，字符串，定义层的输出的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01, l2=0.01)或 None，默认 None。
- 内核约束(kernel_constraint)：选择框，定义权值矩阵的约束函数，选项内容包括 MaxNorm、NonNeg、UnitNorm、MinMaxMorm 或 None，默认 None。
- 阈值约束(bias_constraint)：选择框，定义偏置向量的约束函数，选项内容包括 MaxNorm、NonNeg、UnitNorm、MinMaxMorm 或 None，默认 None。
- 输入维度(Input Shape)：输入框，整数元组，定义输入数据的维度，该层作为 Sequential 层下的第一层时，需指定输入维度，例如 input_shape = (10,128,128,3)代表对 10 帧 128*128 的彩色 RGB 图像进行卷积（data_format='channels_last'），非第一层时输入 None，默认 None。

- Conv3DTranspose（转置卷积层）
  - 名称(name)：字符串，为该 Conv3DTranspose 层指定名称，作为其标识，默认 Conv3DTranspose。
  - 滤波器数(filters)：输入框，整型数值，定义卷积核的数目，即输出的维度，默认 32。
  - 卷积核大小(kernel_size)：输入框，单个整数或由 3 个整数构成的列表或元组，定义卷积核深度、高度和宽度，默认（2,2,2）。
  - 步幅(strides)：输入框，单个整数或由 3 个整数构成的列表或元组，定义卷积沿深度、高度和宽度的步长，默认（1,1,1）。
  - 填充(padding)：选择框，定义补 0 策略，包括 valid 和 same，valid 表示只进行有效的卷积，即对边界数据不处理，same 表示保留边界处的卷积结果，使输出形状和输入形状相同，默认 valid。
  - 输出填充(output_padding)：输入框，单个整数或由 3 个整数构成的列表或元组，指定沿深度、高度和宽度的填充量，沿给定维度的

输出填充量必须小于沿同一维度的步幅，默认 None，表示输出形状根据网络自动推算。

- 数据格式(data_format)：选择框，包括 channels_last 和 channels_first，定义输入中维度的顺序，channels_last 对应输入形状为（batch, spatial_dim1, spatial_dim2, spatial_dim3, channels），channels_first 对应输入形状为（batch, channels, spatial_dim1, spatial_dim2, spatial_dim3），默认 channels_last。
- 膨胀率(dilation_rate)：输入框，单个整数或由 3 个整数构成的列表或元组，定义膨胀卷积的膨胀率，默认（1,1,1）。
- 激活函数(activation)：选择框，为该层选择激活函数，选项包括 relu、tanh、sigmoid、linear、softmax、softplus、softsign、hard_sigmoid、exponential、None，选 None 表示不指定激活函数，默认选择 None。
- 阈值(use_bias)：选择框，布尔型，定义是否使用偏置项，默认 True。
- 内核初始化（kernel_initializer）：选择框，定义内核权值矩阵的初始化方法，选项包括 initializer、random_normal、Random_uniform、truncated_normal、identity、lecun_uniform、lecun_normal、orthogonal、zeros、glorot_normal、glorot_uniform、he_normal、he_uniform、ones，默认 glorot_uniform。
- 阈值初始化（bias_initializer）：选择框，定义偏置向量的初始化方法，选项内容同内核初始化，默认 zeros。
- 内核正则化(kernel_regularizer)：输入框，字符串，定义内核权值矩阵的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01, l2=0.01)或 None，默认 None。
- 阈值正则化（bias_regularizer）：输入框，字符串，定义偏置向量的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01, l2=0.01)或 None，默认 None。
- 活动正则化(activity_regularizer)：输入框，字符串，定义层的

输出的正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01, l2=0.01)或 None，默认 None。

- 内核约束(kernel_constraint)：选择框，定义权值矩阵的约束函数，选项内容包括 MaxNorm、NonNeg、UnitNorm、MinMaxMorm 或 None，默认 None。
- 阈值约束(bias_constraint)：选择框，定义偏置向量的约束函数，选项内容包括 MaxNorm、NonNeg、UnitNorm、MinMaxMorm 或 None，默认 None。
- 输入维度(Input Shape)：输入框，整数元组，定义输入数据的维度，该层作为 Sequential 层下的第一层时，需指定输入维度，例如 input_shape = (10,128,128,3)代表对 10 帧 128*128 的彩色 RGB 图像进行卷积（data_format='channels_last'），非第一层时输入 None，默认 None。

- Cropping1D（一维裁剪层）
  - 名称(name)：输入框，字符串，为该 Cropping1D 层指定名称，作为其标识，默认 Cropping1D；
  - 裁剪大小：输入框，整数或长为 2 的整数元组，指定在序列的首尾要裁剪掉多少个元素，默认（1,1）。

- Cropping2D（二维裁剪层）
  - 名称(name)：输入框，字符串，为该 Cropping2D 层指定名称，作为其标识，默认 Cropping2D。
  - 裁剪大小(cropping)：输入框，整数或 2 个整数的元组或 2 个整数的 2 个元组的元组，如果为整数（如 1），表示相同的对称裁剪应用于高度和宽度；如果为 2 个整数的元组（如（1,1）），表示对高度和宽度的两个不同的对称裁剪值：(symmetric_height_crop, symmetric_width_crop)；如果为 2 个整数的 2 个元组的元组：解释为((top_crop, bottom_crop), (left_crop, right_crop))，默认((0,

0),(0,0))。

- 数据格式(data_format)：选择框，包括 channels_last 和 channels_first，定义输入中维度的顺序，channels_last 对应输入形状为（batch, height, weight, channels），channels_first 对应输入形状为（batch, channels, height, weight），默认 channels_last。

- Cropping3D（三维裁剪层）
  - 名称(name)：输入框，字符串，为该 Cropping3D 层指定名称，作为其标识，默认 Cropping3D。
  - 裁剪大小(cropping)：输入框，整数或 3 个整数的元组或 2 个整数的 3 个元组的元组，如果为整数，将对深度、宽度和高度应用相同的对称裁剪；如果为 3 个整数的元组：解释为对深度、高度和宽度的 3 个不同的对称裁剪值：(symmetric_dim1_crop, symmetric_dim2_crop, symmetric_dim3_crop)；如果为 2 个整数的 3 个元组的元组：解释为((left_dim1_crop, right_dim1_crop), (left_dim2_crop, right_dim2_crop), (left_dim3_crop, right_dim3_crop))，默认((1,1),(1,1),(1,1))。
  - 数据格式(data_format)：选择框，包括 channels_last 和 channels_first，定义输入中维度的顺序，channels_last 对应输入形状为（batch, spatial_dim1, spatial_dim2, spatial_dim3, channels），channels_first 对应输入形状为（batch, channels, spatial_dim1, spatial_dim2, spatial_dim3），默认 channels_last。

- Cropping3D（三维裁剪层）
  - 名称(name)：输入框，字符串，为该 Cropping3D 层指定名称，作为其标识，默认 Cropping3D。
  - 裁剪大小(cropping)：输入框，整数或 3 个整数的元组或 2 个整数的 3 个元组的元组，如果为整数，将对深度、宽度和高度应用相同

的对称裁剪；如果为 3 个整数的元组：解释为对深度、高度和宽度的 3 个不同的对称裁剪值：(symmetric_dim1_crop, symmetric_dim2_crop, symmetric_dim3_crop)；如果为 2 个整数的 3 个元组的元组：解释为((left_dim1_crop, right_dim1_crop), (left_dim2_crop, right_dim2_crop), (left_dim3_crop, right_dim3_crop))，默认((1,1),(1,1),(1,1))。

- 数据格式(data_format)：选择框，包括 channels_last 和 channels_first，定义输入中维度的顺序，channels_last 对应输入形状为（batch, spatial_dim1, spatial_dim2, spatial_dim3, channels），channels_first 对应输入形状为（batch, channels, spatial_dim1, spatial_dim2, spatial_dim3），默认 channels_last。

- UpSampling1D（一维上采样层）
  - 名称(name)：字符串，为该 UpSampling1D 层指定名称，作为其标识，默认 UpSampling1D。
  - 上采样大小(size)：整数，定义上采样因子，指沿着时间轴重复每个时间步的次数，默认 2。

- UpSampling2D（二维上采样层）
  - 名称(name)：字符串，为该 UpSampling2D 层指定名称，作为其标识，默认 UpSampling2D。
  - 上采样大小(size)：整数或 2 个整数的元组，定义行和列的上采样因子，指沿着数据的行和列分别重复的次数，默认（2,2）。
  - 数据格式(data_format)：选择框，包括 channels_last 和 channels_first，定义输入中维度的顺序，channels_last 对应输入形状为（batch, height, weight, channels），channels_first 对应输入形状为（batch, channels, height, weight），默认 channels_last。
  - 插值：选择框，nearest 或 bilinear，指定插值方法，默认 neares

t。

- UpSampling3D（三维上采样层）
  - 名称(name)：字符串，为该 UpSampling3D 层指定名称，作为其标识，默认 UpSampling3D。
  - 上采样大小(size)：整数或 3 个整数的元组，定义 3 个维度的上采样因子，指沿着数据的 3 个维度分别重复的次数，默认(2,2,2)。
  - 数据格式(data_format)：选择框，包括 channels_last 和 channels_first，定义输入中维度的顺序，channels_last 对应输入形状为（batch, spatial_dim1, spatial_dim2, spatial_dim3, channels），channels_first 对应输入形状为（batch, channels, spatial_dim1, spatial_dim2, spatial_dim3），默认 channels_last。

- Activation（激活层）
  - 名称(name)：输入框，字符串，为该 Activation 层指定名称，作为其标识，默认 Activation。
  - 激活函数(activation)：选择框，为该层选择激活函数，选项包括 relu、tanh、sigmoid、linear、softmax、softplus、softsign、hard_sigmoid、exponential、None，选 None 表示不指定激活函数，默认选择 relu。

- MaxPooling1D（时序最大池化层）
  - 名称(name)：输入框，字符串，为该 MaxPooling1D 层指定名称，作为其标识，默认 MaxPooling1D。
  - 池化大小(pool_size)：输入框，整数，定义最大池化的窗口大小，默认 2。
  - 步幅(strides)：输入框，整数或 None，表示下采样因子，例如 2 表示将使得输出 shape 为输入的一半，None 表示默认使用 pool_size 的值，默认 None。

- 填充(padding)：选择框，定义补 0 策略，包括 valid 和 same，valid 表示只进行有效的卷积，即对边界数据不处理，same 表示保留边界处的卷积结果，使输出形状和输入形状相同，默认 valid。
- 数据格式(data_format)：选择框，包括 channels_last 和 channels_first，定义输入中维度的顺序，channels_last 对应输入形状为（batch, steps, features），channels_first 对应输入形状为（batch, features, steps），默认 channels_last。

- MaxPooling2D（空间最大池化层）
  - 名称(name)：字符串，为该 MaxPooling2D 层指定名称，作为其标识，默认 MaxPooling2D。
  - 池化大小(pool_size)：输入框，整数或 2 个整数的元组，表示沿（垂直，水平）方向的下采样因子，如取（2，2）将使图片在两个维度上均变为原长的一半，为整数意为各个维度值相同且为该数字，默认(2,2)。
  - 步幅(strides)：整数或长为 2 的整数元组或 None，定义步长大小，如果为 None，表示 pool_size 的值，默认 None。
  - 填充(padding)：选择框，定义补 0 策略，包括 valid 和 same，valid 表示只进行有效的卷积，即对边界数据不处理，same 表示保留边界处的卷积结果，使输出形状和输入形状相同，默认 valid。
  - 数据格式(data_format)：选择框，包括 channels_last 和 channels_first，定义输入中维度的顺序，channels_last 对应输入形状为（batch, height, weight, channels），channels_first 对应输入形状为（batch, channels, height, weight），默认 channels_last。

- MaxPooling3D（时/空间最大池化层）
  - 名称(name)：字符串，为该 MaxPooling3D 层指定名称，作为其标识，默认 MaxPooling3D。

- 池化大小(pool_size)：输入框，3 个整数的元组，表示在 3 个维度上的下采样因子，如取（2，2，2）表示将每个维度的 3D 输入减半，默认(2,2,2)。
- 步幅(strides)：长为 3 的整数元组或 None，定义步长大小，如果为 None，表示 pool_size 的值，默认 None。
- 填充(padding)：选择框，定义补 0 策略，包括 valid 和 same，valid 表示只进行有效的卷积，即对边界数据不处理，same 表示保留边界处的卷积结果，使输出形状和输入形状相同，默认 valid。
- 数据格式(data_format)：选择框，包括 channels_last 和 channels_first，定义输入中维度的顺序，channels_last 对应输入形状为（batch, spatial_dim1, spatial_dim2, spatial_dim3, channels），channels_first 对应输入形状为（batch, channels, spatial_dim1, spatial_dim2, spatial_dim3），默认 channels_last。

- AveragePooling1D（时序平均池化层）
  - 名称(name)：字符串，为该 AveragePooling1D 层指定名称，作为其标识，默认 AveragePooling1D。
  - 池化大小(pool_size)：输入框，整数，定义平均池化的窗口大小，默认 2。
  - 步幅(strides)：输入框，整数或 None，表示下采样因子，例如 2 表示将使得输出 shape 为输入的一半，None 表示默认使用 pool_size 的值，默认 None。
  - 填充(padding)：选择框，定义补 0 策略，包括 valid 和 same，valid 表示只进行有效的卷积，即对边界数据不处理，same 表示保留边界处的卷积结果，使输出形状和输入形状相同，默认 valid。
  - 数据格式(data_format)：选择框，包括 channels_last 和 channels_first，定义输入中维度的顺序，channels_last 对应输入形状为（batch, steps, features），channels_first 对应输入形状为（batch, features, steps），默认 channels_last。

- AveragePooling2D（空间平均池化层）
  - 名称(name)：字符串，为该 AveragePooling2D 层指定名称，作为其标识，默认 AveragePooling2D。
  - 池化大小(pool_size)：输入框，整数或 2 个整数的元组，表示沿（垂直，水平）方向的下采样因子，如取（2，2）将两个空间维度的输入减半，为整数意为各个维度值相同且为该数字，默认(2,2)。
  - 步幅(strides)：整数或长为 2 的整数元组或 None，定义步长大小，如果为 None，表示 pool_size 的值，默认 None。
  - 填充(padding)：选择框，定义补 0 策略，包括 valid 和 same，valid 表示只进行有效的卷积，即对边界数据不处理，same 表示保留边界处的卷积结果，使输出形状和输入形状相同，默认 valid。
  - 数据格式(data_format)：选择框，包括 channels_last 和 channels_first，定义输入中维度的顺序，channels_last 对应输入形状为（batch, height, weight, channels），channels_first 对应输入形状为（batch, channels, height, weight），默认 channels_last。

- AveragePooling3D（时空间平均池化层）
  - 名称(name)：字符串，为该 AveragePooling3D 层指定名称，作为其标识，默认 AveragePooling3D。
  - 池化大小(pool_size)：输入框，3 个整数的元组，表示在 3 个维度上的下采样因子，如取（2，2，2）表示将每个维度的 3D 输入减半，默认(2,2,2)。
  - 步幅(strides)：长为 3 的整数元组或 None，定义步长大小，如果为 None，表示 pool_size 的值，默认 None。
  - 填充(padding)：选择框，定义补 0 策略，包括 valid 和 same，valid 表示只进行有效的卷积，即对边界数据不处理，same 表示保留边界处的卷积结果，使输出形状和输入形状相同，默认 valid。

- 数据格式(data_format)：选择框，包括 channels_last 和 channels_first，定义输入中维度的顺序，channels_last 对应输入形状为（batch, spatial_dim1, spatial_dim2, spatial_dim3, channels），channels_first 对应输入形状为（batch, channels, spatial_dim1, spatial_dim2, spatial_dim3），默认 channels_last。

- BatchNormalization（批量标准化层）
  - 名称(name)：输入框，字符串，为该 BatchNormalization 层指定名称，作为其标识，默认 BatchNormalization。
  - 特征轴(axis)：输入框，整数，指定需要标准化的轴，默认-1。
  - 动量(momentum)：输入框，浮点数，指定移动均值和移动方差的动量，默认 0.99。
  - epsilon：输入框，浮点数，为方差增加一个小的浮点数，避免被除数为零，默认 0.001。
  - 中心(center)：选择框，布尔值，如果为 True，表示把偏移量加到标准化张量上，如果为 False，则忽略，默认 True。
  - 缩放(scale)：选择框，布尔值，如果为 True，则乘以 gamma，如果为 False，则不使用 gamma。
  - beta 初始化(beta_initializer)：选择框，指定 beta 权重的初始化方法，选项包括 initializer、random_normal、Random_uniform、truncated_normal、identity、lecun_uniform、lecun_normal、orthogonal、zeros、glorot_normal、glorot_uniform、he_normal、he_uniform、ones，默认 zeros。
  - gamma 初始化(gamma_initializer)：选择框，指定 gamma 权重的初始化方法，选项包括 initializer、random_normal、Random_uniform、truncated_normal、identity、lecun_uniform、lecun_normal、orthogonal、zeros、glorot_normal、glorot_uniform、he_normal、he_uniform、ones，默认 ones。
  - 移动平均初始化(moving_mean_initializer)：选择框，指定移动平

均的初始化方法，选项包括 initializer、random_normal、Random_uniform、truncated_normal、identity、lecun_uniform、lecun_normal、orthogonal、zeros、glorot_normal、glorot_uniform、he_normal、he_uniform、ones，默认 zeros。

- 移动方差初始化(moving_variance_initializer)：选择框，指定移动方差的初始化方法，选项包括 initializer、random_normal、Random_uniform、truncated_normal、identity、lecun_uniform、lecun_normal、orthogonal、zeros、glorot_normal、glorot_uniform、he_normal、he_uniform、ones，默认 ones。
- beta 正则化(beta_regularizer)：输入框，字符串，为 beta 权重指定正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01, l2=0.01)或 None，默认 None。
- Gamma 正则化(gamma_regularizer)：输入框，字符串，为 gamma 权重指定正则化函数，如 l1(0.01)、l2(0.01)、l1_l2(l1=0.01, l2=0.01)或 None，默认 None。
- beta 约束(beta_constraint)：选择框，指定 beta 权值的约束函数，选项内容包括 MaxNorm、NonNeg、UnitNorm、MinMaxMorm 或 None，默认 None。
- Gamma 约束(Gamma_constraint)：选择框，指定 Gamma 权值的约束函数，选项内容包括 MaxNorm、NonNeg、UnitNorm、MinMaxMorm 或 None，默认 None。

- Dropout（丢失层）
  - 名称（name）：输入框，字符串，为该 Dropout 层指定名称，作为其标识。
  - 比例因子(rate)：输入框，0 到 1 之间的浮点数，定义输入数据需要丢弃的比例，默认 0.0。

- Flatten（扁平化层）

- 名称（name）：输入框，字符串，为该 Activation 层指定名称，作为其标识。
- 数据格式（data_format）：选择框，包括 channels_last 和 channels_first，定义输入中维度的顺序，channels_last 对应输入形状为（batch, height, weight, ..., channels），channels_first 对应输入形状为（batch, channels, height, weight, ...），默认 channels_last。

- Reshape（重塑层）
  - 名称（name）：字符串，为该 Reshape 层指定名称，作为其标识。
  - 目标形状(target_shape)：整型元组，将数据转换为特定的形状，默认(None, None)。

- RepeatVector（重复层）
  - 名称（name）：字符串，为该 RepeatVector 层指定名称，作为其标识。
  - 重复次数(n)：整型数值，定义输入的重复系数，默认 0。

附录四：模型预测组件属性说明

- 数值模型预测
  - 数值模型预测：机器学习模型预测，包括分类算法，回归算法和聚类算法。
  - 待推理模型类型：算法框架，默认使用 'classification'。
  - Confidence：输入 0 到 1 之间的整数或者浮点数，定义输入置信度，默认 0.5。

- 图像模型预测
  - 图像模型预测：图像模型预测。
  - 待推理模型类型：图像检测使用的算法框架，默认使用 'ssd'。
  - Confidence：输入 0 到 1 之间的整数或者浮点数，定义输入置信度，默认 0.5。
  - Confidence：浮点数，定义输入两张图片的差异，数值越大差异越大，数值越小差异越小，0.0 的距离表示面是相同的，4.0 对应的是相反的，两个不同的身份，默认 1.1。
  - nms_iou：输入 0 到 1 之间的整数或者浮点数，定义非极大抑制所用到的 nms_iou 大小，默认 0.3。
  - backbone:unet 图像分割模型使用的 backbone。默认情况下，使用 'vgg16'。
  - backbone:FaceNet 人脸识别模型使用的 backbone。默认情况下，使用 'mobilenet'。
  - Activation：选择激活函数进行非线性转换，如'softmax'，默认使用'softmax'。

- 仿真结果预测
  - 仿真结果预测：仿真结果预测。

- 待推理模型类型：仿真回归使用的算法框架，默认使用 ’meshgraph nets’。

- 自定义预测
  - 自定义预测：自定义模型预测。
  - 选择数据文件：选择需要的依赖的文件。
  - 模型预测函数名称：填写模型预测函数名称。
  - 自定义读取数据脚本：按照模板添加自定义读取数据脚本。

附录五：镜像说明以及相关 python 包说明

（一）jhinno/tensorflow:py37-2.5

- 镜像说明：tensorflow 作业，方案设计使用镜像。

Package	Version
absl-py	0.15.0
albumentations	1.3.0
alembic	1.9.1
anyio	3.5.0
argon2-cffi	21.3.0
argon2-cffi-bindings	21.2.0
astunparse	1.6.3
attrs	21.4.0
Babel	2.9.1
backcall	0.2.0
beautifulsoup4	4.11.1
bitarray	2.6.2
bleach	4.1.0
cached-property	1.5.2
cachetools	5.2.1
certifi	2022.6.15
cffi	1.14.0
chardet	3.0.4
click	8.1.3

cloudpickle	2.2.0
conda	4.14.0
conda-package-handling	1.7.0
cryptography	2.9.2
cx-Oracle	8.0.0
cycler	0.11.0
cytoolz	0.11.0
dask	2022.2.0
databricks-cli	0.17.4
debugpy	1.5.1
decorator	5.1.1
defusedxml	0.7.1
distributed	2022.2.0
docker	6.0.1
entrypoints	0.4
fastjsonschema	2.16.2
Flask	2.2.2
flatbuffers	1.12
fsspec	2022.11.0
future	0.18.2
gast	0.4.0
gitdb	4.0.10
GitPython	3.1.30
google-auth	2.15.0
google-auth-oauthlib	0.4.6
google-pasta	0.2.0
greenlet	2.0.1
grpcio	1.34.1
gunicorn	20.1.0

h5py	3.1.0
HeapDict	1.0.1
horovod	0.26.1
idna	2.9
imageio	2.24.0
imgaug	0.4.0
importlib-metadata	4.11.3
importlib-resources	5.2.0
impyla	0.16.2
ipykernel	6.9.1
ipynb-py-convert	0.4.6
ipython	7.31.1
ipython-genutils	0.2.0
itsdangerous	2.1.2
jedi	0.18.1
Jinja2	3.0.3
joblib	1.2.0
json5	0.9.6
jsonschema	4.4.0
jupyter-client	7.1.2
jupyter-core	4.10.0
jupyter-server	1.18.1
jupyterlab	3.4.4
jupyterlab-pygments	0.1.2
jupyterlab-server	2.12.0
Keras-Applications	1.0.8
keras-nightly	2.5.0.dev2021032900
Keras-Preprocessing	1.1.2
kiwisolver	1.4.4

libaiflow	5.4
llvmlite	0.39.1
locket	1.0.0
Mako	1.2.4
Markdown	3.4.1
MarkupSafe	2.1.1
matplotlib	3.1.1
matplotlib-inline	0.1.6
matrixprofile	1.1.10
mistune	0.8.4
mlflow	1.26.2.dev0
msgpack	1.0.4
nbclassic	0.3.5
nbclient	0.5.13
nbconvert	6.4.4
nbformat	5.3.0
nest-asyncio	1.5.5
networkx	2.6.3
notebook	6.4.12
numba	0.56.4
numpy	1.19.2
oauthlib	3.2.2
opencv-python	4.6.0.66
opencv-python-headless	4.6.0.66
opt-einsum	3.3.0
packaging	21.3
pandas	1.1.0
pandocfilters	1.5.0
parso	0.8.3

partd	1.3.0
patsy	0.5.3
pexpect	4.8.0
pickleshare	0.7.5
Pillow	6.2.0
pip	20.0.2
plotly	5.11.0
ply	3.11
prometheus-client	0.14.1
prometheus-flask-exporter	0.21.0
prompt-toolkit	3.0.20
protobuf	3.20.1
psutil	5.9.4
psycopg2-binary	2.9.3
ptyprocess	0.7.0
pyasn1	0.4.8
pyasn1-modules	0.2.8
pycocotools	2.0.6
pycosat	0.6.3
pycparser	2.20
pycryptodome	3.15.0
Pygments	2.11.2
PyHDFS	0.3.1
PyHive	0.6.2
PyJWT	2.6.0
PyMySQL	0.9.3
pyOpenSSL	19.1.0
pyparsing	3.0.9
pyrsistent	0.18.0

PySocks	1.7.1
python-dateutil	2.8.2
pytz	2022.1
PyWavelets	1.3.0
PyYAML	6.0
pyzmq	22.3.0
qudida	0.0.4
querystring-parser	1.2.4
read-jhdata	5.4
requests	2.23.0
requests-oauthlib	1.3.1
rsa	4.9
ruamel-yaml	0.15.87
scikit-image	0.16.2
scikit-learn	1.0.2
scipy	1.4.1
Send2Trash	1.8.0
setuptools	46.4.0.post20200518
shapely	2.0.0
simplejson	3.18.1
six	1.15.0
smmap	5.0.0
sniffio	1.2.0
sortedcontainers	2.4.0
soupsieve	2.3.1
SQLAlchemy	1.4.46
sqlparse	0.4.3
statsmodels	0.13.5
stumpy	1.11.1

tabulate	0.9.0
tblib	1.7.0
tenacity	8.1.0
tensorboard	2.11.0
tensorboard-data-server	0.6.1
tensorboard-plugin-wit	1.8.1
tensorflow	2.5.0
tensorflow-estimator	2.5.0
tensorflow-gpu	2.5.0
tensorflow-hub	0.9.0
tensorflowjs	3.6.0
termcolor	1.1.0
terminado	0.13.1
testpath	0.6.0
threadpoolctl	3.1.0
thrift	0.16.0
thriftpy2	0.4.11
toolz	0.11.2
tornado	6.1
tqdm	4.46.0
traitlets	5.1.1
tsfresh	0.19.0
typing-extensions	3.7.4.3
urllib3	1.24
wcwidth	0.2.5
webencodings	0.5.1
websocket-client	0.58.0
Werkzeug	2.2.2
wheel	0.38.4

wrapt	1.12.1
zict	2.2.0
zipp	3.8.0

- conda 环境包。

```
# packages in environment at /apps/miniconda3:
#
# Name                    Version
_libgcc_mutex             0.1
absl-py                   0.15.0
albumentations            1.3.0
alembic                   1.9.1
anyio                     3.5.0
argon2-cffi               21.3.0
argon2-cffi-bindings      21.2.0
astunparse                1.6.3
attrs                     21.4.0
babel                     2.9.1
backcall                  0.2.0
beautifulsoup4            4.11.1
bitarray                  2.6.2
bleach                    4.1.0
ca-certificates           2022.07.19
cached-property           1.5.2
cachetools                5.2.1
certifi                   2022.6.15
cffi                      1.14.0
chardet                   3.0.4
click                     8.1.3
```

cloudpickle	2.2.0
conda	4.14.0
conda-package-handling	1.6.1
cryptography	2.9.2
cx-oracle	8.0.0
cycler	0.11.0
cytoolz	0.11.0
dask	2022.2.0
databricks-cli	0.17.4
debugpy	1.5.1
decorator	5.1.1
defusedxml	0.7.1
distributed	2022.2.0
docker	6.0.1
entrypoints	0.4
flask	2.2.2
flatbuffers	1.12
fsspec	2022.11.0
future	0.18.2
gast	0.4.0
gitdb	4.0.10
gitpython	3.1.30
google-auth	2.15.0
google-auth-oauthlib	0.4.6
google-pasta	0.2.0
greenlet	2.0.1
grpcio	1.34.1
gunicorn	20.1.0
h5py	3.1.0

heapdict	1.0.1
horovod	0.26.1
idna	2.9
imageio	2.24.0
imgaug	0.4.0
importlib-metadata	4.11.3
importlib_metadata	4.11.3
importlib_resources	5.2.0
impyla	0.16.2
ipykernel	6.9.1
ipynb-py-convert	0.4.6
ipython	7.31.1
ipython_genutils	0.2.0
itsdangerous	2.1.2
jedi	0.18.1
jinja2	3.0.3
joblib	1.2.0
json5	0.9.6
jsonschema	4.4.0
jupyter_client	7.1.2
jupyter_core	4.10.0
jupyter_server	1.18.1
jupyterlab	3.4.4
jupyterlab_pygments	0.1.2
jupyterlab_server	2.12.0
keras-applications	1.0.8
keras-nightly	2.5.0.dev2021032
keras-preprocessing	1.1.2
kiwisolver	1.4.4

ld_impl_linux-64	2.33.1
libaiflow	5.4
libedit	3.1.20181209
libffi	3.3
libgcc-ng	9.1.0
libsodium	1.0.18
libstdcxx-ng	9.1.0
llvmlite	0.39.1
locket	1.0.0
mako	1.2.4
markdown	3.4.1
markupsafe	2.1.1
matplotlib	3.1.1
matplotlib-inline	0.1.6
matrixprofile	1.1.10
mistune	0.8.4
mlflow	1.26.2.dev0
msgpack	1.0.4
nbclassic	0.3.5
nbclient	0.5.13
nbconvert	6.4.4
nbformat	5.3.0
ncurses	6.2
nest-asyncio	1.5.5
networkx	2.6.3
notebook	6.4.12
numba	0.56.4
numpy	1.19.2
oauthlib	3.2.2

opencv-python	4.6.0.66
opencv-python-headless	4.6.0.66
openssl	1.1.1q
opt-einsum	3.3.0
packaging	21.3
pandas	1.1.0
pandocfilters	1.5.0
parso	0.8.3
partd	1.3.0
patsy	0.5.3
pexpect	4.8.0
pickleshare	0.7.5
pillow	6.2.0
pip	20.0.2
plotly	5.11.0
ply	3.11
prometheus-flask-exporter	0.21.0
prometheus_client	0.14.1
prompt-toolkit	3.0.20
protobuf	3.20.1
psutil	5.9.4
psycopg2-binary	2.9.3
ptyprocess	0.7.0
pyasn1	0.4.8
pyasn1-modules	0.2.8
pycocotools	2.0.6
pycosat	0.6.3
pycparser	2.20
pycryptodome	3.15.0

pygments	2.11.2
pyhdfs	0.3.1
pyhive	0.6.2
pyjwt	2.6.0
pymysql	0.9.3
pyopenssl	19.1.0
pyparsing	3.0.9
pyrsistent	0.18.0
pysocks	1.7.1
python	3.7.7
python-dateutil	2.8.2
python-fastjsonschema	2.16.2
pytz	2022.1
pywavelets	1.3.0
pyyaml	6.0
pyzmq	22.3.0
qudida	0.0.4
querystring-parser	1.2.4
read-jhdata	5.4
readline	8.0
requests	2.23.0
requests-oauthlib	1.3.1
rsa	4.9
ruamel_yaml	0.15.87
scikit-image	0.16.2
scikit-learn	1.0.2
scipy	1.4.1
send2trash	1.8.0
setuptools	46.4.0

shapely	2.0.0
simplejson	3.18.1
six	1.15.0
smmap	5.0.0
sniffio	1.2.0
sortedcontainers	2.4.0
soupsieve	2.3.1
sqlalchemy	1.4.46
sqlite	3.31.1
sqlparse	0.4.3
statsmodels	0.13.5
stumpy	1.11.1
tabulate	0.9.0
tblib	1.7.0
tenacity	8.1.0
tensorboard	2.11.0
tensorboard-data-server	0.6.1
tensorboard-plugin-wit	1.8.1
tensorflow	2.5.0
tensorflow-estimator	2.5.0
tensorflow-gpu	2.5.0
tensorflow-hub	0.9.0
tensorflowjs	3.6.0
termcolor	1.1.0
terminado	0.13.1
testpath	0.6.0
threadpoolctl	3.1.0
thrift	0.16.0
thriftpy2	0.4.11

tk	8.6.8
toolz	0.11.2
tornado	6.1
tqdm	4.46.0
traitlets	5.1.1
tsfresh	0.19.0
typing-extensions	3.7.4.3
urllib3	1.24
wcwidth	0.2.5
webencodings	0.5.1
websocket-client	0.58.0
werkzeug	2.2.2
wheel	0.38.4
wrapt	1.12.1
xz	5.2.5
yaml	0.1.7
zeromq	4.3.4
zict	2.2.0
zipp	3.8.0
zlib	1.2.11

（二）jhinno/pytorch:py37-1.10.0

- 镜像说明：提交 pytorch 作业所用镜像。

absl-py	1.3.0
aiohttp	3.8.3
aiosignal	1.3.1
alembic	1.9.1
anyio	3.5.0

argon2-cffi	21.3.0
argon2-cffi-bindings	21.2.0
async-timeout	4.0.2
asynctest	0.13.0
attrs	21.4.0
Babel	2.9.1
backcall	0.2.0
beautifulsoup4	4.11.1
bitarray	2.6.2
bleach	4.1.0
cachetools	5.2.1
certifi	2022.6.15
cffi	1.14.0
chardet	3.0.4
charset-normalizer	2.1.1
click	8.1.3
cloudpickle	2.2.0
conda	4.14.0
conda-package-handling	1.7.0
cryptography	2.9.2
cx-Oracle	8.0.0
cycler	0.11.0
cytoolz	0.11.0
databricks-cli	0.17.4
debugpy	1.5.1
decorator	5.1.1
defusedxml	0.7.1
docker	6.0.1
entrypoints	0.4

fastjsonschema	2.16.2
Flask	2.2.2
frozenlist	1.3.3
fsspec	2022.11.0
gitdb	4.0.10
GitPython	3.1.30
google-auth	2.15.0
google-auth-oauthlib	0.4.6
greenlet	2.0.1
grpcio	1.51.1
gunicorn	20.1.0
horovod	0.26.1
idna	2.9
importlib-metadata	4.11.3
importlib-resources	5.2.0
impyla	0.16.2
ipykernel	6.9.1
ipython	7.31.1
ipython-genutils	0.2.0
itsdangerous	2.1.2
jedi	0.18.1
Jinja2	3.0.3
joblib	1.2.0
json5	0.9.6
jsonschema	4.4.0
jupyter-client	7.1.2
jupyter-core	4.10.0
jupyter-server	1.18.1
jupyterlab	3.4.4

jupyterlab-pygments	0.1.2
jupyterlab-server	2.12.0
kiwisolver	1.4.4
libaiflow	5.4
Mako	1.2.4
Markdown	3.4.1
MarkupSafe	2.1.1
matplotlib	3.1.1
matplotlib-inline	0.1.6
mistune	0.8.4
mlflow	1.26.2.dev0
multidict	6.0.4
nbclassic	0.3.5
nbclient	0.5.13
nbconvert	6.4.4
nbformat	5.3.0
nest-asyncio	1.5.5
notebook	6.4.12
numpy	1.21.6
oauthlib	3.2.2
opencv-python	4.2.0.32
packaging	21.3
pandas	1.1.0
pandocfilters	1.5.0
parso	0.8.3
pexpect	4.8.0
pickleshare	0.7.5
Pillow	6.2.0
pip	20.0.2

plotly	5.11.0
ply	3.11
prometheus-client	0.14.1
prometheus-flask-exporter	0.21.0
prompt-toolkit	3.0.20
protobuf	3.20.1
psutil	5.9.4
psycopg2-binary	2.9.3
ptyprocess	0.7.0
pyasn1	0.4.8
pyasn1-modules	0.2.8
pycosat	0.6.3
pycparser	2.20
pycryptodome	3.15.0
pyDeprecate	0.3.2
Pygments	2.11.2
PyHDFS	0.3.1
PyJWT	2.6.0
PyMySQL	0.9.3
pyOpenSSL	19.1.0
pyparsing	3.0.9
pyrsistent	0.18.0
PySocks	1.7.1
python-dateutil	2.8.2
pytorch-lightning	1.7.1
pytz	2022.1
PyYAML	6.0
pyzmq	22.3.0
querystring-parser	1.2.4

requests	2.23.0
requests-oauthlib	1.3.1
rsa	4.9
ruamel-yaml	0.15.87
scikit-learn	0.21.3
scipy	1.7.3
Send2Trash	1.8.0
sentencepiece	0.1.97
setuptools	46.4.0.post20200518
simplejson	3.18.1
six	1.14.0
smmap	5.0.0
sniffio	1.2.0
soupsieve	2.3.1
SQLAlchemy	1.4.46
sqlparse	0.4.3
tabulate	0.9.0
tenacity	8.1.0
tensorboard	2.11.0
tensorboard-data-server	0.6.1
tensorboard-plugin-wit	1.8.1
terminado	0.13.1
testpath	0.6.0
thrift	0.16.0
thriftpy2	0.4.16
toolz	0.11.2
torch	1.10.0+cu113
torchaudio	0.10.0+cu113
torchmetrics	0.11.0

torchtext	0.6.0
torchvision	0.11.1+cu113
tornado	6.1
tqdm	4.64.1
traitlets	5.1.1
typing-extensions	4.3.0
urllib3	1.25.8
wcwidth	0.2.5
webencodings	0.5.1
websocket-client	0.58.0
Werkzeug	2.2.2
wheel	0.34.2
yarl	1.8.2
zipp	3.8.0

- conda 环境包。

```
# packages in environment at /apps/miniconda3:
#
# Name                    Version
_libgcc_mutex             0.1
absl-py                   1.3.0
aiohttp                   3.8.3
aiosignal                 1.3.1
alembic                   1.9.1
anyio                     3.5.0
argon2-cffi               21.3.0
argon2-cffi-bindings      21.2.0
async-timeout             4.0.2
asynctest                 0.13.0
```

attrs	21.4.0
babel	2.9.1
backcall	0.2.0
beautifulsoup4	4.11.1
binutils_impl_linux-64	2.33.1
binutils_linux-64	2.33.1
bitarray	2.6.2
bleach	4.1.0
ca-certificates	2022.07.19
cachetools	5.2.1
certifi	2022.6.15
cffi	1.14.0
chardet	3.0.4
charset-normalizer	2.1.1
click	8.1.3
cloudpickle	2.2.0
conda	4.14.0
conda-package-handling	1.6.1
cryptography	2.9.2
cx-oracle	8.0.0
cycler	0.11.0
cytoolz	0.11.0
databricks-cli	0.17.4
debugpy	1.5.1
decorator	5.1.1
defusedxml	0.7.1
docker	6.0.1
entrypoints	0.4
flask	2.2.2

frozenlist	1.3.3
fsspec	2022.11.0
gcc_impl_linux-64	7.3.0
gcc_linux-64	7.3.0
gitdb	4.0.10
gitpython	3.1.30
google-auth	2.15.0
google-auth-oauthlib	0.4.6
greenlet	2.0.1
grpcio	1.51.1
gunicorn	20.1.0
gxx_impl_linux-64	7.3.0
gxx_linux-64	7.3.0
horovod	0.26.1
idna	2.9
importlib-metadata	4.11.3
importlib_metadata	4.11.3
importlib_resources	5.2.0
impyla	0.16.2
ipykernel	6.9.1
ipython	7.31.1
ipython_genutils	0.2.0
itsdangerous	2.1.2
jedi	0.18.1
jinja2	3.0.3
joblib	1.2.0
json5	0.9.6
jsonschema	4.4.0
jupyter_client	7.1.2

jupyter_core	4.10.0
jupyter_server	1.18.1
jupyterlab	3.4.4
jupyterlab_pygments	0.1.2
jupyterlab_server	2.12.0
kiwisolver	1.4.4
ld_impl_linux-64	2.33.1
libaiflow	5.4
libedit	3.1.20181209
libffi	3.3
libgcc-ng	9.1.0
libsodium	1.0.18
libstdcxx-ng	9.1.0
mako	1.2.4
markdown	3.4.1
markupsafe	2.1.1
matplotlib	3.1.1
matplotlib-inline	0.1.6
mistune	0.8.4
mlflow	1.26.2.dev0
multidict	6.0.4
nbclassic	0.3.5
nbclient	0.5.13
nbconvert	6.4.4
nbformat	5.3.0
ncurses	6.2
nest-asyncio	1.5.5
notebook	6.4.12
numpy	1.21.6

oauthlib	3.2.2
opencv-python	4.2.0.32
openssl	1.1.1q
packaging	21.3
pandas	1.1.0
pandocfilters	1.5.0
parso	0.8.3
pexpect	4.8.0
pickleshare	0.7.5
pillow	6.2.0
pip	20.0.2
plotly	5.11.0
ply	3.11
prometheus-flask-exporter	0.21.0
prometheus_client	0.14.1
prompt-toolkit	3.0.20
protobuf	3.20.1
psutil	5.9.4
psycopg2-binary	2.9.3
ptyprocess	0.7.0
pyasn1	0.4.8
pyasn1-modules	0.2.8
pycosat	0.6.3
pycparser	2.20
pycryptodome	3.15.0
pydeprecate	0.3.2
pygments	2.11.2
pyhdfs	0.3.1
pyjwt	2.6.0

pymysql	0.9.3
pyopenssl	19.1.0
pyparsing	3.0.9
pyrsistent	0.18.0
pysocks	1.7.1
python	3.7.7
python-dateutil	2.8.2
python-fastjsonschema	2.16.2
pytorch-lightning	1.7.1
pytz	2022.1
pyyaml	6.0
pyzmq	22.3.0
querystring-parser	1.2.4
readline	8.0
requests	2.23.0
requests-oauthlib	1.3.1
rsa	4.9
ruamel_yaml	0.15.87
scikit-learn	0.21.3
scipy	1.7.3
send2trash	1.8.0
sentencepiece	0.1.97
setuptools	46.4.0
simplejson	3.18.1
six	1.14.0
smmap	5.0.0
sniffio	1.2.0
soupsieve	2.3.1
sqlalchemy	1.4.46

sqlite	3.31.1
sqlparse	0.4.3
tabulate	0.9.0
tenacity	8.1.0
tensorboard	2.11.0
tensorboard-data-server	0.6.1
tensorboard-plugin-wit	1.8.1
terminado	0.13.1
testpath	0.6.0
thrift	0.16.0
thriftpy2	0.4.16
tk	8.6.8
toolz	0.11.2
torch	1.10.0+cu113
torchaudio	0.10.0+cu113
torchmetrics	0.11.0
torchtext	0.6.0
torchvision	0.11.1+cu113
tornado	6.1
tqdm	4.64.1
traitlets	5.1.1
typing-extensions	4.3.0
typing_extensions	4.3.0
urllib3	1.25.8
wcwidth	0.2.5
webencodings	0.5.1
websocket-client	0.58.0
werkzeug	2.2.2
wheel	0.34.2

xz	5.2.5
yaml	0.1.7
yarl	1.8.2
zeromq	4.3.4
zipp	3.8.0
zlib	1.2.11

（三）jhinno/mxnet:py37-1.9.1

- 镜像说明：提交 mxnet 作业所用镜像。

Package	Version
alembic	1.8.0
anyio	3.5.0
argon2-cffi	21.3.0
argon2-cffi-bindings	21.2.0
attrs	21.4.0
Babel	2.9.1
backcall	0.2.0
beautifulsoup4	4.11.1
bitarray	2.6.2
bleach	4.1.0
certifi	2022.6.15
cffi	1.14.0
chardet	3.0.4
charset-normalizer	2.0.12
click	8.1.3
cloudpickle	2.1.0
conda	4.14.0

conda-package-handling	1.7.0
cryptography	2.9.2
cx-Oracle	8.0.0
cycler	0.11.0
cytoolz	0.11.0
databricks-cli	0.17.4
debugpy	1.5.1
decorator	5.1.1
defusedxml	0.7.1
docker	5.0.3
entrypoints	0.4
fastjsonschema	2.16.2
Flask	2.1.2
gitdb	4.0.9
GitPython	3.1.27
graphviz	0.8.4
greenlet	1.1.2
gunicorn	20.1.0
horovod	0.26.1
idna	3.3
importlib-metadata	4.11.4
importlib-resources	5.8.0
impyla	0.16.2
ipykernel	6.9.1
ipython	7.31.1
ipython-genutils	0.2.0
itsdangerous	2.1.2
jedi	0.18.1
Jinja2	3.1.2

joblib	1.2.0
json5	0.9.6
jsonschema	4.4.0
jupyter-client	7.1.2
jupyter-core	4.10.0
jupyter-server	1.18.1
jupyterlab	3.4.4
jupyterlab-pygments	0.1.2
jupyterlab-server	2.12.0
kiwisolver	1.4.4
libaiflow	5.4
Mako	1.2.0
MarkupSafe	2.1.1
matplotlib	3.1.1
matplotlib-inline	0.1.6
mistune	0.8.4
mlflow	1.26.2.dev0
mxnet-cu112	1.9.1
nbclassic	0.3.5
nbclient	0.5.13
nbconvert	6.4.4
nbformat	5.3.0
nest-asyncio	1.5.5
notebook	6.4.12
numpy	1.21.6
oauthlib	3.2.0
opencv-python	4.2.0.32
packaging	21.3
pandas	1.3.5

pandocfilters	1.5.0
parso	0.8.3
pexpect	4.8.0
pickleshare	0.7.5
pip	20.0.2
plotly	5.11.0
ply	3.11
prometheus-client	0.14.1
prometheus-flask-exporter	0.20.2
prompt-toolkit	3.0.20
protobuf	4.21.1
psutil	5.9.4
psycopg2-binary	2.9.3
ptyprocess	0.7.0
pycosat	0.6.3
pycparser	2.20
pycryptodome	3.15.0
Pygments	2.11.2
PyHDFS	0.3.1
PyJWT	2.4.0
pyOpenSSL	19.1.0
pyparsing	3.0.9
pyrsistent	0.18.0
PySocks	1.7.1
python-dateutil	2.8.2
pytz	2022.1
PyYAML	6.0
pyzmq	22.3.0
querystring-parser	1.2.4

requests	2.28.0
ruamel-yaml	0.15.87
scikit-learn	0.21.3
scipy	1.7.3
Send2Trash	1.8.0
setuptools	62.6.0
simplejson	3.18.1
six	1.16.0
smmap	5.0.0
sniffio	1.2.0
soupsieve	2.3.1
SQLAlchemy	1.4.37
sqlparse	0.4.2
tabulate	0.8.10
tenacity	8.1.0
terminado	0.13.1
testpath	0.6.0
thrift	0.16.0
thriftpy2	0.4.16
toolz	0.11.2
tornado	6.1
tqdm	4.46.0
traitlets	5.1.1
typing-extensions	4.2.0
urllib3	1.26.9
wcwidth	0.2.5
webencodings	0.5.1
websocket-client	1.3.3
Werkzeug	2.1.2

wheel	0.34.2
zipp	3.8.0

- conda 环境包。

```
# packages in environment at /apps/miniconda3:
#
# Name                    Version
_libgcc_mutex             0.1
alembic                   1.8.0
anyio                     3.5.0
argon2-cffi               21.3.0
argon2-cffi-bindings      21.2.0
attrs                     21.4.0
babel                     2.9.1
backcall                  0.2.0
beautifulsoup4            4.11.1
binutils_impl_linux-64    2.33.1
binutils_linux-64         2.33.1
bitarray                  2.6.2
bleach                    4.1.0
ca-certificates           2022.07.19
certifi                   2022.6.15
cffi                      1.14.0
chardet                   3.0.4
charset-normalizer        2.0.12
click                     8.1.3
cloudpickle               2.1.0
conda                     4.14.0
conda-package-handling    1.6.1
```

cryptography	2.9.2
cx-oracle	8.0.0
cycler	0.11.0
cytoolz	0.11.0
databricks-cli	0.17.4
debugpy	1.5.1
decorator	5.1.1
defusedxml	0.7.1
docker	5.0.3
entrypoints	0.4
flask	2.1.2
gcc_impl_linux-64	7.3.0
gcc_linux-64	7.3.0
gitdb	4.0.9
gitpython	3.1.27
greenlet	1.1.2
gunicorn	20.1.0
gxx_impl_linux-64	7.3.0
gxx_linux-64	7.3.0
horovod	0.26.1
idna	3.3
importlib-metadata	4.11.4
importlib-resources	5.8.0
importlib_metadata	4.11.3
impyla	0.16.2
ipykernel	6.9.1
ipython	7.31.1
ipython_genutils	0.2.0
itsdangerous	2.1.2

jedi	0.18.1
jinja2	3.1.2
joblib	1.2.0
json5	0.9.6
jsonschema	4.4.0
jupyter_client	7.1.2
jupyter_core	4.10.0
jupyter_server	1.18.1
jupyterlab	3.4.4
jupyterlab_pygments	0.1.2
jupyterlab_server	2.12.0
kiwisolver	1.4.4
ld_impl_linux-64	2.33.1
libaiflow	5.4
libedit	3.1.20181209
libffi	3.3
libgcc-ng	9.1.0
libsodium	1.0.18
libstdcxx-ng	9.1.0
mako	1.2.0
markupsafe	2.1.1
matplotlib	3.1.1
matplotlib-inline	0.1.6
mistune	0.8.4
mlflow	1.26.2.dev0
mxnet-cu112	1.9.1
nbclassic	0.3.5
nbclient	0.5.13
nbconvert	6.4.4

nbformat	5.3.0
ncurses	6.2
nest-asyncio	1.5.5
notebook	6.4.12
numpy	1.21.6
oauthlib	3.2.0
opencv-python	4.2.0.32
openssl	1.1.1q
packaging	21.3
pandas	1.3.5
pandocfilters	1.5.0
parso	0.8.3
pexpect	4.8.0
pickleshare	0.7.5
pip	20.0.2
plotly	5.11.0
ply	3.11
prometheus-flask-exporter	0.20.2
prometheus_client	0.14.1
prompt-toolkit	3.0.20
protobuf	4.21.1
psutil	5.9.4
psycopg2-binary	2.9.3
ptyprocess	0.7.0
pycosat	0.6.3
pycparser	2.20
pycryptodome	3.15.0
pygments	2.11.2
pyhdfs	0.3.1

pyjwt	2.4.0
pyopenssl	19.1.0
pyparsing	3.0.9
pyrsistent	0.18.0
pysocks	1.7.1
python	3.7.7
python-dateutil	2.8.2
python-fastjsonschema	2.16.2
python-graphviz	0.8.4
pytz	2022.1
pyyaml	6.0
pyzmq	22.3.0
querystring-parser	1.2.4
readline	8.0
requests	2.28.0
ruamel_yaml	0.15.87
scikit-learn	0.21.3
scipy	1.7.3
send2trash	1.8.0
setuptools	62.6.0
simplejson	3.18.1
six	1.16.0
smmap	5.0.0
sniffio	1.2.0
soupsieve	2.3.1
sqlalchemy	1.4.37
sqlite	3.31.1
sqlparse	0.4.2
tabulate	0.8.10

tenacity	8.1.0
terminado	0.13.1
testpath	0.6.0
thrift	0.16.0
thriftpy2	0.4.16
tk	8.6.8
toolz	0.11.2
tornado	6.1
tqdm	4.46.0
traitlets	5.1.1
typing-extensions	4.2.0
urllib3	1.26.9
wcwidth	0.2.5
webencodings	0.5.1
websocket-client	1.3.3
werkzeug	2.1.2
wheel	0.34.2
xz	5.2.5
yaml	0.1.7
zeromq	4.3.4
zipp	3.8.0
zlib	1.2.11

（四）jhinno/paddle:py37-2.3.2

- 镜像说明：提交 paddle 作业所用镜像。

Package	Version
alembic	1.9.1

anyio	3.5.0
argon2-cffi	21.3.0
argon2-cffi-bindings	21.2.0
astor	0.8.1
attrs	21.4.0
Babel	2.9.1
backcall	0.2.0
beautifulsoup4	4.11.1
bitarray	2.6.2
bleach	4.1.0
certifi	2022.6.15
cffi	1.14.0
chardet	3.0.4
click	8.1.3
cloudpickle	2.2.0
conda	4.14.0
conda-package-handling	1.7.0
cryptography	2.9.2
cx-Oracle	8.0.0
cycler	0.11.0
cytoolz	0.11.0
databricks-cli	0.17.4
debugpy	1.5.1
decorator	5.1.1
defusedxml	0.7.1
docker	6.0.1
entrypoints	0.4
fastjsonschema	2.16.2
Flask	2.2.2

gitdb	4.0.10
GitPython	3.1.30
greenlet	2.0.1
gunicorn	20.1.0
idna	2.9
importlib-metadata	4.11.3
importlib-resources	5.2.0
impyla	0.16.2
ipykernel	6.9.1
ipython	7.31.1
ipython-genutils	0.2.0
itsdangerous	2.1.2
jedi	0.18.1
Jinja2	3.0.3
joblib	1.2.0
json5	0.9.6
jsonschema	4.4.0
jupyter-client	7.1.2
jupyter-core	4.10.0
jupyter-server	1.18.1
jupyterlab	3.4.4
jupyterlab-pygments	0.1.2
jupyterlab-server	2.12.0
kiwisolver	1.4.4
libaiflow	5.4
Mako	1.2.4
MarkupSafe	2.1.1
matplotlib	3.1.1
matplotlib-inline	0.1.6

mistune	0.8.4
mlflow	1.26.2.dev0
nbclassic	0.3.5
nbclient	0.5.13
nbconvert	6.4.4
nbformat	5.3.0
nest-asyncio	1.5.5
notebook	6.4.12
numpy	1.21.6
oauthlib	3.2.2
opencv-python	4.2.0.32
opt-einsum	3.3.0
packaging	21.3
paddle-bfloat	0.1.7
paddlepaddle-gpu	2.3.2.post112
pandas	1.1.0
pandocfilters	1.5.0
parso	0.8.3
pexpect	4.8.0
pickleshare	0.7.5
Pillow	9.4.0
pip	20.0.2
plotly	5.11.0
ply	3.11
prometheus-client	0.14.1
prometheus-flask-exporter	0.21.0
prompt-toolkit	3.0.20
protobuf	3.20.0
psycopg2-binary	2.9.3

ptyprocess	0.7.0
pycosat	0.6.3
pycparser	2.20
pycryptodome	3.15.0
Pygments	2.11.2
PyHDFS	0.3.1
PyJWT	2.6.0
pyOpenSSL	19.1.0
pyparsing	3.0.9
pyrsistent	0.18.0
PySocks	1.7.1
python-dateutil	2.8.2
pytz	2022.1
PyYAML	6.0
pyzmq	22.3.0
querystring-parser	1.2.4
requests	2.23.0
ruamel-yaml	0.15.87
scikit-learn	0.21.3
scipy	1.7.3
Send2Trash	1.8.0
setuptools	46.4.0.post20200518
simplejson	3.18.1
six	1.14.0
smmap	5.0.0
sniffio	1.2.0
soupsieve	2.3.1
SQLAlchemy	1.4.46
sqlparse	0.4.3

tabulate	0.9.0
tenacity	8.1.0
terminado	0.13.1
testpath	0.6.0
thrift	0.16.0
thriftpy2	0.4.16
toolz	0.11.2
tornado	6.1
tqdm	4.46.0
traitlets	5.1.1
typing-extensions	4.3.0
urllib3	1.26.13
wcwidth	0.2.5
webencodings	0.5.1
websocket-client	0.58.0
Werkzeug	2.2.2
wheel	0.34.2
zipp	3.8.0

- conda 环境包。

```
# packages in environment at /apps/miniconda3:
#
# Name                    Version
_libgcc_mutex             0.1
alembic                   1.9.1
anyio                     3.5.0
argon2-cffi               21.3.0
argon2-cffi-bindings      21.2.0
astor                     0.8.1
```

attrs	21.4.0
babel	2.9.1
backcall	0.2.0
beautifulsoup4	4.11.1
binutils_impl_linux-64	2.33.1
binutils_linux-64	2.33.1
bitarray	2.6.2
bleach	4.1.0
ca-certificates	2022.07.19
certifi	2022.6.15
cffi	1.14.0
chardet	3.0.4
click	8.1.3
cloudpickle	2.2.0
conda	4.14.0
conda-package-handling	1.6.1
cryptography	2.9.2
cx-oracle	8.0.0
cycler	0.11.0
cytoolz	0.11.0
databricks-cli	0.17.4
debugpy	1.5.1
decorator	5.1.1
defusedxml	0.7.1
docker	6.0.1
entrypoints	0.4
flask	2.2.2
gcc_impl_linux-64	7.3.0
gcc_linux-64	7.3.0

gitdb	4.0.10
gitpython	3.1.30
greenlet	2.0.1
gunicorn	20.1.0
gxx_impl_linux-64	7.3.0
gxx_linux-64	7.3.0
idna	2.9
importlib-metadata	4.11.3
importlib_metadata	4.11.3
importlib_resources	5.2.0
impyla	0.16.2
ipykernel	6.9.1
ipython	7.31.1
ipython_genutils	0.2.0
itsdangerous	2.1.2
jedi	0.18.1
jinja2	3.0.3
joblib	1.2.0
json5	0.9.6
jsonschema	4.4.0
jupyter_client	7.1.2
jupyter_core	4.10.0
jupyter_server	1.18.1
jupyterlab	3.4.4
jupyterlab_pygments	0.1.2
jupyterlab_server	2.12.0
kiwisolver	1.4.4
ld_impl_linux-64	2.33.1
libaiflow	5.4

libedit	3.1.20181209
libffi	3.3
libgcc-ng	9.1.0
libsodium	1.0.18
libstdcxx-ng	9.1.0
mako	1.2.4
markupsafe	2.1.1
matplotlib	3.1.1
matplotlib-inline	0.1.6
mistune	0.8.4
mlflow	1.26.2.dev0
nbclassic	0.3.5
nbclient	0.5.13
nbconvert	6.4.4
nbformat	5.3.0
ncurses	6.2
nest-asyncio	1.5.5
notebook	6.4.12
numpy	1.21.6
oauthlib	3.2.2
opencv-python	4.2.0.32
openssl	1.1.1q
opt-einsum	3.3.0
packaging	21.3
paddle-bfloat	0.1.7
paddlepaddle-gpu	2.3.2.post112
pandas	1.1.0
pandocfilters	1.5.0
parso	0.8.3

pexpect	4.8.0
pickleshare	0.7.5
pillow	9.4.0
pip	20.0.2
plotly	5.11.0
ply	3.11
prometheus-flask-exporter	0.21.0
prometheus_client	0.14.1
prompt-toolkit	3.0.20
protobuf	3.20.0
psycopg2-binary	2.9.3
ptyprocess	0.7.0
pycosat	0.6.3
pycparser	2.20
pycryptodome	3.15.0
pygments	2.11.2
pyhdfs	0.3.1
pyjwt	2.6.0
pyopenssl	19.1.0
pyparsing	3.0.9
pyrsistent	0.18.0
pysocks	1.7.1
python	3.7.7
python-dateutil	2.8.2
python-fastjsonschema	2.16.2
pytz	2022.1
pyyaml	6.0
pyzmq	22.3.0
querystring-parser	1.2.4

readline	8.0
requests	2.23.0
ruamel_yaml	0.15.87
scikit-learn	0.21.3
scipy	1.7.3
send2trash	1.8.0
setuptools	46.4.0
simplejson	3.18.1
six	1.14.0
smmap	5.0.0
sniffio	1.2.0
soupsieve	2.3.1
sqlalchemy	1.4.46
sqlite	3.31.1
sqlparse	0.4.3
tabulate	0.9.0
tenacity	8.1.0
terminado	0.13.1
testpath	0.6.0
thrift	0.16.0
thriftpy2	0.4.16
tk	8.6.8
toolz	0.11.2
tornado	6.1
tqdm	4.46.0
traitlets	5.1.1
typing-extensions	4.3.0
typing_extensions	4.3.0
urllib3	1.26.13

wcwidth	0.2.5
webencodings	0.5.1
websocket-client	0.58.0
werkzeug	2.2.2
wheel	0.34.2
xz	5.2.5
yaml	0.1.7
zeromq	4.3.4
zipp	3.8.0
zlib	1.2.11

（五）jhinno/webide:cuda11.2.0

- 镜像说明：开发中心中新建、启动 webide 环境实例所用镜像。包含三个 python 环境，base,tensorflow2 和 pytorch。
- Tensorflow2。

Package	Version
absl-py	0.15.0
alembic	1.8.0
astunparse	1.6.3
backcall	0.2.0
bitarray	2.7.3
cached-property	1.5.2
cachetools	5.3.0
certifi	2022.6.15
charset-normalizer	2.0.12
click	8.1.3
cloudpickle	2.1.0

cx-Oracle	8.0.0
cycler	0.11.0
databricks-cli	0.17.0
decorator	5.1.1
docker	5.0.3
entrypoints	0.4
Flask	2.1.2
flatbuffers	1.12
future	0.18.3
gast	0.4.0
gitdb	4.0.9
GitPython	3.1.27
google-auth	2.16.1
google-auth-oauthlib	0.4.6
google-pasta	0.2.0
greenlet	1.1.2
grpcio	1.34.1
gunicorn	20.1.0
h5py	3.1.0
idna	3.3
imageio	2.26.0
importlib-metadata	4.11.4
importlib-resources	5.8.0
impyla	0.16.2
ipykernel	5.4.3
ipython	7.34.0
ipython-genutils	0.2.0
itsdangerous	2.1.2
jedi	0.18.2

Jinja2	3.1.2
joblib	1.2.0
jupyter_client	7.4.9
jupyter_core	4.12.0
keras-nightly	2.5.0.dev2021032900
Keras-Preprocessing	1.1.2
kiwisolver	1.4.4
libaiflow	5.4
Mako	1.2.0
Markdown	3.4.1
MarkupSafe	2.1.1
matplotlib	3.1.1
matplotlib-inline	0.1.6
mlflow	1.26.2.dev0
nest-asyncio	1.5.6
networkx	2.6.3
numpy	1.19.5
oauthlib	3.2.0
opencv-python	4.2.0.32
opt-einsum	3.3.0
packaging	21.3
pandas	1.3.5
parso	0.8.3
pexpect	4.8.0
pickleshare	0.7.5
Pillow	9.4.0
pip	22.1.2
plotly	5.13.1
ply	3.11

prometheus-client	0.14.1
prometheus-flask-exporter	0.20.2
prompt-toolkit	3.0.38
protobuf	3.20.0
psycopg2-binary	2.9.3
ptyprocess	0.7.0
pyasn1	0.4.8
pyasn1-modules	0.2.8
pycryptodome	3.15.0
Pygments	2.14.0
PyHDFS	0.3.1
PyHive	0.6.2
PyJWT	2.4.0
PyMySQL	0.9.3
pyparsing	3.0.9
python-dateutil	2.8.2
pytz	2022.1
PyWavelets	1.3.0
PyYAML	6.0
pyzmq	25.0.0
querystring-parser	1.2.4
requests	2.28.0
requests-oauthlib	1.3.1
rsa	4.9
scikit-image	0.19.3
scikit-learn	0.21.3
scipy	1.7.3
setuptools	62.6.0
simplejson	3.18.3

six	1.15.0
smmap	5.0.0
SQLAlchemy	1.4.37
sqlparse	0.4.2
tabulate	0.8.10
tenacity	8.2.2
tensorboard	2.11.2
tensorboard-data-server	0.6.1
tensorboard-plugin-wit	1.8.1
tensorflow	2.5.0
tensorflow-estimator	2.5.0
tensorflow-gpu	2.5.0
termcolor	1.1.0
thrift	0.16.0
thriftpy2	0.4.11
tifffile	2021.11.2
tornado	6.2
traitlets	5.9.0
typing-extensions	3.7.4.3
urllib3	1.26.9
wcwidth	0.2.6
websocket-client	1.3.3
Werkzeug	2.1.2
wheel	0.37.1
wrapt	1.12.1
zipp	3.8.0

- pytorch

Package	Version

---------- ---------

absl-py 1.4.0

aiohttp 3.8.4

aiosignal 1.3.1

alembic 1.8.0

async-timeout 4.0.2

asynctest 0.13.0

attrs 22.2.0

backcall 0.2.0

cachetools 4.2.4

certifi 2022.6.15

charset-normalizer 2.0.12

click 8.1.3

cloudpickle 2.1.0

cycler 0.11.0

databricks-cli 0.17.0

decorator 5.1.1

docker 5.0.3

entrypoints 0.4

Flask 2.1.2

fonttools 4.38.0

frozenlist 1.3.3

fsspec 2023.1.0

gitdb 4.0.9

GitPython 3.1.27

google-auth 1.35.0

google-auth-oauthlib 0.4.6

greenlet 1.1.2

grpcio 1.51.3

gunicorn	20.1.0
idna	3.3
imageio	2.26.0
importlib-metadata	4.11.4
importlib-resources	5.8.0
ipykernel	5.4.3
ipython	7.34.0
ipython-genutils	0.2.0
itsdangerous	2.1.2
jedi	0.18.2
Jinja2	3.1.2
jupyter_client	7.4.9
jupyter_core	4.12.0
kiwisolver	1.4.4
libaiflow	5.4
Mako	1.2.0
Markdown	3.4.1
MarkupSafe	2.1.1
matplotlib	3.5.3
matplotlib-inline	0.1.6
mlflow	1.26.2.dev0
multidict	6.0.4
nest-asyncio	1.5.6
networkx	2.6.3
numpy	1.21.6
nvidia-cublas-cu11	11.10.3.66
nvidia-cuda-nvrtc-cu11	11.7.99
nvidia-cuda-runtime-cu11	11.7.99
nvidia-cudnn-cu11	8.5.0.96

oauthlib	3.2.0
opencv-python	4.2.0.32
packaging	21.3
pandas	1.3.5
parso	0.8.3
pexpect	4.8.0
pickleshare	0.7.5
Pillow	9.4.0
pip	22.1.2
plotly	5.13.1
prometheus-client	0.14.1
prometheus-flask-exporter	0.20.2
prompt-toolkit	3.0.38
protobuf	3.20.0
psycopg2-binary	2.9.3
ptyprocess	0.7.0
pyasn1	0.4.8
pyasn1-modules	0.2.8
pycryptodome	3.15.0
pyDeprecate	0.3.2
Pygments	2.14.0
PyJWT	2.4.0
pyparsing	3.0.9
python-dateutil	2.8.2
pytorch-lightning	1.7.1
pytz	2022.1
PyWavelets	1.3.0
PyYAML	6.0
pyzmq	25.0.0

querystring-parser	1.2.4
requests	2.28.0
requests-oauthlib	1.3.1
rsa	4.9
scikit-image	0.19.3
scipy	1.7.3
setuptools	58.0.4
six	1.16.0
smmap	5.0.0
SQLAlchemy	1.4.37
sqlparse	0.4.2
tabulate	0.8.10
tenacity	8.2.2
tensorboard	2.3.0
tensorboard-data-server	0.6.1
tensorboard-plugin-wit	1.8.1
tifffile	2021.11.2
torch	1.10.0+cu113
torchaudio	0.10.0+cu113
torchmetrics	0.11.3
torchvision	0.11.1+cu113
tornado	6.2
tqdm	4.64.1
traitlets	5.9.0
typing_extensions	4.2.0
urllib3	1.26.9
wcwidth	0.2.6
websocket-client	1.3.3
Werkzeug	2.1.2

wheel	0.37.1
yarl	1.8.2
zipp	3.8.0

- base

Package	Version
absl-py	0.15.0
anyio	3.5.0
api-ext	0.1
argon2-cffi	21.3.0
argon2-cffi-bindings	21.2.0
astunparse	1.6.3
attrs	21.4.0
Babel	2.9.1
backcall	0.2.0
beautifulsoup4	4.11.1
bleach	4.1.0
cached-property	1.5.2
cachetools	5.3.0
certifi	2022.12.7
cffi	1.14.0
chardet	3.0.4
conda	23.1.0
conda-package-handling	1.7.0
cryptography	2.9.2
cytoolz	0.11.0
debugpy	1.5.1
decorator	5.1.1

defusedxml	0.7.1
deprecation	2.1.0
entrypoints	0.4
fastjsonschema	2.16.2
flatbuffers	1.12
gast	0.4.0
google-auth	2.16.1
google-auth-oauthlib	0.4.6
google-pasta	0.2.0
grpcio	1.34.1
h5py	3.1.0
idna	2.9
importlib-metadata	4.11.3
importlib-resources	5.2.0
ipykernel	6.9.1
ipynb-py-convert	0.4.6
ipython	7.31.1
ipython-genutils	0.2.0
jedi	0.18.1
Jinja2	3.0.3
json5	0.9.6
jsonschema	4.4.0
jupyter-client	7.1.2
jupyter-core	4.10.0
jupyter-lsp	1.5.1
jupyter-packaging	0.12.3
jupyter-server	1.18.1
jupyter-tensorboard	0.2.0
jupyterlab	3.4.4

jupyterlab-lsp	3.10.2
jupyterlab-pygments	0.1.2
jupyterlab-server	2.12.0
jupyterlab-topbar	0.6.0
keras-nightly	2.5.0.dev2021032900
Keras-Preprocessing	1.1.2
lab-ext	0.0.0
Markdown	3.4.1
MarkupSafe	2.1.1
matplotlib-inline	0.1.6
mistune	0.8.4
nbclassic	0.3.5
nbclient	0.5.13
nbconvert	6.4.4
nbformat	5.3.0
nest-asyncio	1.5.5
notebook	6.4.12
numpy	1.19.5
oauthlib	3.2.2
opt-einsum	3.3.0
packaging	21.3
pandocfilters	1.5.0
parso	0.8.3
pexpect	4.8.0
pickleshare	0.7.5
pip	20.0.2
pluggy	1.0.0
prometheus-client	0.14.1
prompt-toolkit	3.0.20

protobuf	3.19.0
ptyprocess	0.7.0
pyasn1	0.4.8
pyasn1-modules	0.2.8
pycosat	0.6.3
pycparser	2.20
Pygments	2.11.2
pyOpenSSL	19.1.0
pyparsing	3.0.9
pyrsistent	0.18.0
PySocks	1.7.1
python-dateutil	2.8.2
pytz	2022.1
pyzmq	22.3.0
requests	2.23.0
requests-oauthlib	1.3.1
rsa	4.9
ruamel-yaml	0.15.87
ruamel.yaml	0.16.12
ruamel.yaml.clib	0.2.6
Send2Trash	1.8.0
setuptools	67.4.0
six	1.15.0
sniffio	1.2.0
soupsieve	2.3.1
tensorboard	2.11.2
tensorboard-data-server	0.6.1
tensorboard-plugin-wit	1.8.1
tensorflow	2.5.0

tensorflow-estimator	2.5.0
termcolor	1.1.0
terminado	0.13.1
testpath	0.6.0
tomlkit	0.11.6
toolz	0.11.2
tornado	6.1
tqdm	4.46.0
traitlets	5.1.1
typing-extensions	3.7.4.3
urllib3	1.25.8
wcwidth	0.2.5
webencodings	0.5.1
websocket-client	0.58.0
Werkzeug	2.2.3
wheel	0.34.2
wrapt	1.12.1
zipp	3.8.0

- conda 环境

```
# packages in environment at /apps/miniconda3:
#
# Name                    Version
_libgcc_mutex             0.1
absl-py                   0.15.0
anyio                     3.5.0
argon2-cffi               21.3.0
argon2-cffi-bindings      21.2.0
astunparse                1.6.3
```

attrs	21.4.0
babel	2.9.1
backcall	0.2.0
beautifulsoup4	4.11.1
bleach	4.1.0
ca-certificates	2023.01.10
cached-property	1.5.2
cachetools	5.3.0
certifi	2022.12.7
cffi	1.14.0
chardet	3.0.4
conda	23.1.0
conda-package-handling	1.6.1
cryptography	2.9.2
cytoolz	0.11.0
debugpy	1.5.1
decorator	5.1.1
defusedxml	0.7.1
deprecation	2.1.0
entrypoints	0.4
flatbuffers	1.12
gast	0.4.0
google-auth	2.16.1
google-auth-oauthlib	0.4.6
google-pasta	0.2.0
grpcio	1.34.1
h5py	3.1.0
icu	68.1
idna	2.9

importlib-metadata	4.11.3
importlib_metadata	4.11.3
importlib_resources	5.2.0
ipykernel	6.9.1
ipynb-py-convert	0.4.6
ipython	7.31.1
ipython_genutils	0.2.0
jedi	0.18.1
jinja2	3.0.3
json5	0.9.6
jsonschema	4.4.0
jupyter-lsp	1.5.1
jupyter-packaging	0.12.3
jupyter-tensorboard	0.2.0
jupyter_client	7.1.2
jupyter_core	4.10.0
jupyter_server	1.18.1
jupyterlab	3.4.4
jupyterlab-lsp	3.10.2
jupyterlab-topbar	0.6.0
jupyterlab_pygments	0.1.2
jupyterlab_server	2.12.0
keras-nightly	2.5.0.dev2021032
keras-preprocessing	1.1.2
lab-ext	0.0.0
ld_impl_linux-64	2.33.1
libedit	3.1.20181209
libffi	3.3
libgcc-ng	9.1.0

libsodium	1.0.18
libstdcxx-ng	9.1.0
libuv	1.40.0
markdown	3.4.1
markupsafe	2.1.1
matplotlib-inline	0.1.6
mistune	0.8.4
nbclassic	0.3.5
nbclient	0.5.13
nbconvert	6.4.4
nbformat	5.3.0
ncurses	6.2
nest-asyncio	1.5.5
nodejs	16.13.1
notebook	6.4.12
numpy	1.19.5
oauthlib	3.2.2
openssl	1.1.1t
opt-einsum	3.3.0
packaging	21.3
pandocfilters	1.5.0
parso	0.8.3
pexpect	4.8.0
pickleshare	0.7.5
pip	20.0.2
pluggy	1.0.0
prometheus_client	0.14.1
prompt-toolkit	3.0.20
protobuf	3.19.0

ptyprocess	0.7.0
pyasn1	0.4.8
pyasn1-modules	0.2.8
pycosat	0.6.3
pycparser	2.20
pygments	2.11.2
pyopenssl	19.1.0
pyparsing	3.0.9
pyrsistent	0.18.0
pysocks	1.7.1
python	3.7.7
python-dateutil	2.8.2
python-fastjsonschema	2.16.2
pytz	2022.1
pyzmq	22.3.0
readline	8.0
requests	2.23.0
requests-oauthlib	1.3.1
rsa	4.9
ruamel.yaml	0.16.12
ruamel.yaml.clib	0.2.6
ruamel_yaml	0.15.87
send2trash	1.8.0
setuptools	67.4.0
six	1.15.0
sniffio	1.2.0
soupsieve	2.3.1
sqlite	3.31.1
tensorboard	2.11.2

tensorboard-data-server	0.6.1
tensorboard-plugin-wit	1.8.1
tensorflow	2.5.0
tensorflow-estimator	2.5.0
termcolor	1.1.0
terminado	0.13.1
testpath	0.6.0
tk	8.6.8
tomlkit	0.11.6
toolz	0.11.2
tornado	6.1
tqdm	4.46.0
traitlets	5.1.1
typing-extensions	3.7.4.3
urllib3	1.25.8
wcwidth	0.2.5
webencodings	0.5.1
websocket-client	0.58.0
werkzeug	2.2.3
wheel	0.34.2
wrapt	1.12.1
xz	5.2.5
yaml	0.1.7
zeromq	4.3.4
zipp	3.8.0
zlib	1.2.11

（六）jhinno/tensorboard:v5.4

- 镜像说明：启动 tensorboard 服务，所用镜像。

Package	Version
absl-py	0.15.0
anyio	3.6.2
argon2-cffi	21.3.0
argon2-cffi-bindings	21.2.0
asttokens	2.2.1
astunparse	1.6.3
attrs	22.2.0
Babel	2.11.0
backcall	0.2.0
beautifulsoup4	4.11.1
bleach	5.0.1
cachetools	4.2.4
certifi	2022.12.7
cffi	1.15.1
charset-normalizer	3.0.1
comm	0.1.2
debugpy	1.6.5
decorator	5.1.1
defusedxml	0.7.1
entrypoints	0.4
executing	1.2.0
fastjsonschema	2.16.2
flatbuffers	1.12
gast	0.4.0
google-auth	1.35.0

google-auth-oauthlib	0.4.6
google-pasta	0.2.0
grpcio	1.34.1
h5py	3.1.0
idna	3.4
importlib-metadata	6.0.0
importlib-resources	5.10.2
ipykernel	6.20.2
ipython	8.8.0
ipython-genutils	0.2.0
jedi	0.18.2
Jinja2	3.1.2
json5	0.9.11
jsonschema	4.17.3
jupyter-client	7.4.9
jupyter-core	5.1.3
jupyter-server	1.23.5
jupyter-tensorboard	0.2.0
jupyterlab	3.3.2
jupyterlab-pygments	0.2.2
jupyterlab-server	2.19.0
keras-nightly	2.5.0.dev2021032900
Keras-Preprocessing	1.1.2
Markdown	3.4.1
MarkupSafe	2.1.1
matplotlib-inline	0.1.6
mistune	2.0.4
nbclassic	0.4.8
nbclient	0.7.2

nbconvert	7.2.8
nbformat	5.7.3
nest-asyncio	1.5.6
notebook	6.4.12
notebook-shim	0.2.2
numpy	1.19.5
oauthlib	3.2.2
opt-einsum	3.3.0
packaging	23.0
pandocfilters	1.5.0
parso	0.8.3
pexpect	4.8.0
pickleshare	0.7.5
pip	20.0.2
pkgutil-resolve-name	1.3.10
platformdirs	2.6.2
prometheus-client	0.15.0
prompt-toolkit	3.0.36
protobuf	3.19.0
psutil	5.9.4
ptyprocess	0.7.0
pure-eval	0.2.2
pyasn1	0.4.8
pyasn1-modules	0.2.8
pycparser	2.21
Pygments	2.14.0
pyrsistent	0.19.3
python-dateutil	2.8.2
pytz	2022.7.1

pyzmq	25.0.0
requests	2.28.2
requests-oauthlib	1.3.1
rsa	4.9
Send2Trash	1.8.0
setuptools	45.2.0
six	1.15.0
sniffio	1.3.0
soupsieve	2.3.2.post1
stack-data	0.6.2
tensorboard	2.5.0
tensorboard-data-server	0.6.1
tensorboard-plugin-wit	1.8.1
tensorflow	2.5.0
tensorflow-estimator	2.5.0
termcolor	1.1.0
terminado	0.17.1
tinycss2	1.2.1
tornado	6.2
traitlets	5.8.1
typing-extensions	3.7.4.3
urllib3	1.26.14
wcwidth	0.2.6
webencodings	0.5.1
websocket-client	1.4.2
Werkzeug	2.2.2
wheel	0.38.4
wrapt	1.12.1
zipp	3.11.0

（七）jhinno/tf-serving:py37-2.5

- 镜像说明：部署 tensorflow 模型所用镜像。

Package	Version
absl-py	0.15.0
astunparse	1.6.3
cached-property	1.5.2
cachetools	5.3.0
certifi	2020.4.5.1
cffi	1.14.0
chardet	3.0.4
conda	4.8.3
conda-package-handling	1.7.0
cryptography	2.9.2
cycler	0.11.0
flatbuffers	1.12
fonttools	4.38.0
gast	0.4.0
google-auth	2.16.0
google-auth-oauthlib	0.4.6
google-pasta	0.2.0
grpcio	1.34.1
h5py	3.1.0
idna	2.9
imageio	2.25.0
importlib-metadata	6.0.0
keras-nightly	2.5.0.dev2021032900

Keras-Preprocessing	1.1.2
kiwisolver	1.4.4
Markdown	3.4.1
MarkupSafe	2.1.2
matplotlib	3.5.3
networkx	2.6.3
numpy	1.19.5
oauthlib	3.2.2
opencv-python	4.6.0.66
opt-einsum	3.3.0
packaging	23.0
Pillow	9.4.0
pip	20.0.2
protobuf	3.19.0
pyasn1	0.4.8
pyasn1-modules	0.2.8
pycocotools	2.0.6
pycosat	0.6.3
pycparser	2.20
pyOpenSSL	19.1.0
pyparsing	3.0.9
PySocks	1.7.1
python-dateutil	2.8.2
PyWavelets	1.3.0
requests	2.23.0
requests-oauthlib	1.3.1
rsa	4.9
ruamel-yaml	0.15.87
scikit-image	0.19.3

scipy	1.4.1
setuptools	46.4.0.post20200518
six	1.15.0
tensorboard	2.11.2
tensorboard-data-server	0.6.1
tensorboard-plugin-wit	1.8.1
tensorflow	2.5.0
tensorflow-estimator	2.5.0
termcolor	1.1.0
tifffile	2021.11.2
tqdm	4.46.0
typing-extensions	3.7.4.3
urllib3	1.25.8
Werkzeug	2.2.2
wheel	0.38.4
wrapt	1.12.1
zipp	3.12.0

（八）jhinno/torch-serving:py3-1.10

- 镜像说明：部署 pytorch 模型的镜像。

Package	Version
captum	0.6.0
cycler	0.11.0
dataclasses	0.8
enum-compat	0.0.3
future	0.18.3
kiwisolver	1.3.1

matplotlib	3.3.4
numpy	1.19.5
packaging	21.3
Pillow	8.4.0
pip	21.3.1
pkg_resources	0.0.0
psutil	5.9.4
pyparsing	3.0.9
python-dateutil	2.8.2
PyYAML	6.0
setuptools	39.0.1
six	1.16.0
torch	1.10.0+cpu
torch-model-archiver	0.7.0
torch-workflow-archiver	0.2.6
torchaudio	0.10.0+rocm4.1
torchserve	0.7.0
torchvision	0.11.0+cpu
typing_extensions	4.1.1
wheel	0.37.1

（九）jhinno/ubuntu-desktop:18.04-cudag111

- 镜像说明：启动 ubuntu 桌面容器所需镜像，镜像中包含三个 python 环境，base,tensorflow2,pytorch。
- base 环境 python 包列表。

Package	Version
absl-py	0.15.0

anyio	3.5.0
api-ext	0.1
argon2-cffi	21.3.0
argon2-cffi-bindings	21.2.0
astunparse	1.6.3
attrs	21.4.0
Babel	2.9.1
backcall	0.2.0
beautifulsoup4	4.11.1
bleach	4.1.0
cached-property	1.5.2
cachetools	5.2.0
certifi	2022.12.7
cffi	1.14.0
chardet	3.0.4
conda	22.11.1
conda-package-handling	1.7.0
cryptography	2.9.2
cytoolz	0.11.0
debugpy	1.5.1
decorator	5.1.1
defusedxml	0.7.1
deprecation	2.1.0
entrypoints	0.4
fastjsonschema	2.16.2
flatbuffers	1.12
gast	0.4.0
google-auth	2.15.0
google-auth-oauthlib	0.4.6

google-pasta	0.2.0
grpcio	1.34.1
h5py	3.1.0
idna	2.9
importlib-metadata	4.11.3
importlib-resources	5.2.0
ipykernel	6.9.1
ipynb-py-convert	0.4.6
ipython	7.31.1
ipython-genutils	0.2.0
jedi	0.18.1
Jinja2	3.0.3
json5	0.9.6
jsonschema	4.4.0
jupyter-client	7.1.2
jupyter-core	4.10.0
jupyter-lsp	1.5.1
jupyter-packaging	0.12.3
jupyter-server	1.18.1
jupyter-tensorboard	0.2.0
jupyterlab	3.4.4
jupyterlab-lsp	3.10.2
jupyterlab-pygments	0.1.2
jupyterlab-server	2.12.0
jupyterlab-topbar	0.6.0
keras-nightly	2.5.0.dev2021032900
Keras-Preprocessing	1.1.2
Markdown	3.4.1
MarkupSafe	2.1.1

matplotlib-inline	0.1.6
mistune	0.8.4
nbclassic	0.3.5
nbclient	0.5.13
nbconvert	6.4.4
nbformat	5.3.0
nest-asyncio	1.5.5
notebook	6.4.12
numpy	1.19.5
oauthlib	3.2.2
opt-einsum	3.3.0
packaging	21.3
pandocfilters	1.5.0
parso	0.8.3
pexpect	4.8.0
pickleshare	0.7.5
pip	20.0.2
pluggy	1.0.0
prometheus-client	0.14.1
prompt-toolkit	3.0.20
protobuf	3.19.0
ptyprocess	0.7.0
pyasn1	0.4.8
pyasn1-modules	0.2.8
pycosat	0.6.3
pycparser	2.20
Pygments	2.11.2
pyOpenSSL	19.1.0
pyparsing	3.0.9

pyrsistent	0.18.0
PySocks	1.7.1
python-dateutil	2.8.2
pytz	2022.1
pyzmq	22.3.0
requests	2.23.0
requests-oauthlib	1.3.1
rsa	4.9
ruamel-yaml	0.15.87
ruamel.yaml	0.16.12
ruamel.yaml.clib	0.2.6
Send2Trash	1.8.0
setuptools	65.6.3
six	1.15.0
sniffio	1.2.0
soupsieve	2.3.1
tensorboard	2.11.0
tensorboard-data-server	0.6.1
tensorboard-plugin-wit	1.8.1
tensorflow	2.5.0
tensorflow-estimator	2.5.0
termcolor	1.1.0
terminado	0.13.1
testpath	0.6.0
tomlkit	0.11.6
toolz	0.11.2
tornado	6.1
tqdm	4.46.0
traitlets	5.1.1

typing-extensions	3.7.4.3
urllib3	1.25.8
wcwidth	0.2.5
webencodings	0.5.1
websocket-client	0.58.0
Werkzeug	2.2.2
wheel	0.34.2
wrapt	1.12.1
zipp	3.8.0

- tensorflow2 环境 python 包列表。

Package	Version
absl-py	0.15.0
alembic	1.9.1
astunparse	1.6.3
backcall	0.2.0
bitarray	2.6.2
cached-property	1.5.2
cachetools	5.2.0
certifi	2022.6.15
charset-normalizer	2.1.1
click	8.1.3
cloudpickle	2.2.0
cx-Oracle	8.0.0
cycler	0.11.0
databricks-cli	0.17.4
decorator	5.1.1
docker	6.0.1

entrypoints	0.4
Flask	2.2.2
flatbuffers	1.12
future	0.18.2
gast	0.4.0
gitdb	4.0.10
GitPython	3.1.30
google-auth	2.15.0
google-auth-oauthlib	0.4.6
google-pasta	0.2.0
greenlet	2.0.1
grpcio	1.34.1
gunicorn	20.1.0
h5py	3.1.0
idna	3.4
imageio	2.23.0
importlib-metadata	6.0.0
importlib-resources	5.10.2
impyla	0.16.2
ipykernel	5.4.3
ipython	7.34.0
ipython-genutils	0.2.0
itsdangerous	2.1.2
jedi	0.18.2
Jinja2	3.1.2
joblib	1.2.0
jupyter_client	7.4.8
jupyter_core	4.12.0
keras-nightly	2.5.0.dev2021032900

Keras-Preprocessing	1.1.2
kiwisolver	1.4.4
libaiflow	5.4
Mako	1.2.4
Markdown	3.4.1
MarkupSafe	2.1.1
matplotlib	3.1.1
matplotlib-inline	0.1.6
mlflow	1.26.2.dev0
nest-asyncio	1.5.6
networkx	2.6.3
numpy	1.19.5
oauthlib	3.2.2
opencv-python	4.2.0.32
opt-einsum	3.3.0
packaging	22.0
pandas	1.1.0
parso	0.8.3
pexpect	4.8.0
pickleshare	0.7.5
Pillow	9.4.0
pip	22.1.2
plotly	5.11.0
ply	3.11
prometheus-client	0.15.0
prometheus-flask-exporter	0.21.0
prompt-toolkit	3.0.36
protobuf	3.19.0
psycopg2-binary	2.9.3

ptyprocess	0.7.0
pyasn1	0.4.8
pyasn1-modules	0.2.8
pycryptodome	3.15.0
Pygments	2.14.0
PyHDFS	0.3.1
PyHive	0.6.2
PyJWT	2.6.0
PyMySQL	0.9.3
pyparsing	3.0.9
python-dateutil	2.8.2
pytz	2022.7
PyWavelets	1.3.0
PyYAML	6.0
pyzmq	24.0.1
querystring-parser	1.2.4
requests	2.28.1
requests-oauthlib	1.3.1
rsa	4.9
scikit-image	0.19.3
scikit-learn	0.21.3
scipy	1.4.1
setuptools	63.4.1
simplejson	3.18.0
six	1.15.0
smmap	5.0.0
SQLAlchemy	1.4.45
sqlparse	0.4.3
tabulate	0.9.0

tenacity	8.1.0
tensorboard	2.11.0
tensorboard-data-server	0.6.1
tensorboard-plugin-wit	1.8.1
tensorflow	2.5.0
tensorflow-estimator	2.5.0
tensorflow-gpu	2.5.0
termcolor	1.1.0
thrift	0.16.0
thriftpy2	0.4.11
tifffile	2021.11.2
tornado	6.2
traitlets	5.8.0
typing-extensions	3.7.4.3
urllib3	1.26.13
wcwidth	0.2.5
websocket-client	1.4.2
Werkzeug	2.2.2
wheel	0.37.1
wrapt	1.12.1
zipp	3.11.0

- pytorch 环境 python 包列表。

Package	Version
absl-py	1.3.0
aiohttp	3.8.3
aiosignal	1.3.1
alembic	1.9.1

async-timeout	4.0.2
asynctest	0.13.0
attrs	22.2.0
backcall	0.2.0
cachetools	4.2.4
certifi	2022.6.15
charset-normalizer	2.1.1
click	8.1.3
cloudpickle	2.2.0
cycler	0.11.0
databricks-cli	0.17.4
decorator	5.1.1
docker	6.0.1
entrypoints	0.4
Flask	2.2.2
fonttools	4.38.0
frozenlist	1.3.3
fsspec	2022.11.0
gitdb	4.0.10
GitPython	3.1.30
google-auth	1.35.0
google-auth-oauthlib	0.4.6
greenlet	2.0.1
grpcio	1.51.1
gunicorn	20.1.0
idna	3.4
imageio	2.23.0
importlib-metadata	6.0.0
importlib-resources	5.10.2

ipykernel	5.4.3
ipython	7.34.0
ipython-genutils	0.2.0
itsdangerous	2.1.2
jedi	0.18.2
Jinja2	3.1.2
jupyter_client	7.4.8
jupyter_core	4.12.0
kiwisolver	1.4.4
libaiflow	5.4
Mako	1.2.4
Markdown	3.4.1
MarkupSafe	2.1.1
matplotlib	3.5.3
matplotlib-inline	0.1.6
mlflow	1.26.2.dev0
multidict	6.0.4
nest-asyncio	1.5.6
networkx	2.6.3
numpy	1.21.6
nvidia-cublas-cu11	11.10.3.66
nvidia-cuda-nvrtc-cu11	11.7.99
nvidia-cuda-runtime-cu11	11.7.99
nvidia-cudnn-cu11	8.5.0.96
oauthlib	3.2.2
opencv-python	4.2.0.32
packaging	22.0
pandas	1.3.5
parso	0.8.3

pexpect	4.8.0
pickleshare	0.7.5
Pillow	9.4.0
pip	22.1.2
plotly	5.11.0
prometheus-client	0.15.0
prometheus-flask-exporter	0.21.0
prompt-toolkit	3.0.36
protobuf	3.20.1
psycopg2-binary	2.9.3
ptyprocess	0.7.0
pyasn1	0.4.8
pyasn1-modules	0.2.8
pycryptodome	3.15.0
pyDeprecate	0.3.2
Pygments	2.14.0
PyJWT	2.6.0
pyparsing	3.0.9
python-dateutil	2.8.2
pytorch-lightning	1.7.1
pytz	2022.7
PyWavelets	1.3.0
PyYAML	6.0
pyzmq	24.0.1
querystring-parser	1.2.4
requests	2.28.1
requests-oauthlib	1.3.1
rsa	4.9
scikit-image	0.19.3

scipy	1.4.1
setuptools	59.5.0
six	1.16.0
smmap	5.0.0
SQLAlchemy	1.4.45
sqlparse	0.4.3
tabulate	0.9.0
tenacity	8.1.0
tensorboard	2.3.0
tensorboard-data-server	0.6.1
tensorboard-plugin-wit	1.8.1
tifffile	2021.11.2
torch	1.10.0+cu113
torchaudio	0.10.0+cu113
torchmetrics	0.11.0
torchvision	0.11.1+cu113
tornado	6.2
tqdm	4.64.1
traitlets	5.8.0
typing_extensions	4.4.0
urllib3	1.26.13
wcwidth	0.2.5
websocket-client	1.4.2
Werkzeug	2.2.2
wheel	0.37.1
yarl	1.8.2
zipp	3.11.0

（十）jhinno/centos-desktop:7-cudag111

- 镜像说明：启动 centos 桌面容器所需镜像，镜像中包含三个 python 环境，base,tensorflow2,pytorch。
- base 环境 python 包列表。

Package	Version
absl-py	0.15.0
anyio	3.5.0
api-ext	0.1
argon2-cffi	21.3.0
argon2-cffi-bindings	21.2.0
astunparse	1.6.3
attrs	21.4.0
Babel	2.9.1
backcall	0.2.0
beautifulsoup4	4.11.1
bleach	4.1.0
cached-property	1.5.2
cachetools	5.2.0
certifi	2022.12.7
cffi	1.14.0
chardet	3.0.4
conda	22.11.1
conda-package-handling	1.7.0
cryptography	2.9.2
cytoolz	0.11.0
debugpy	1.5.1
decorator	5.1.1
defusedxml	0.7.1

deprecation	2.1.0
entrypoints	0.4
fastjsonschema	2.16.2
flatbuffers	1.12
gast	0.4.0
google-auth	2.15.0
google-auth-oauthlib	0.4.6
google-pasta	0.2.0
grpcio	1.34.1
h5py	3.1.0
idna	2.9
importlib-metadata	4.11.3
importlib-resources	5.2.0
ipykernel	6.9.1
ipynb-py-convert	0.4.6
ipython	7.31.1
ipython-genutils	0.2.0
jedi	0.18.1
Jinja2	3.0.3
json5	0.9.6
jsonschema	4.4.0
jupyter-client	7.1.2
jupyter-core	4.10.0
jupyter-lsp	1.5.1
jupyter-packaging	0.12.3
jupyter-server	1.18.1
jupyter-tensorboard	0.2.0
jupyterlab	3.4.4
jupyterlab-lsp	3.10.2

jupyterlab-pygments	0.1.2
jupyterlab-server	2.12.0
jupyterlab-topbar	0.6.0
keras-nightly	2.5.0.dev2021032900
Keras-Preprocessing	1.1.2
Markdown	3.4.1
MarkupSafe	2.1.1
matplotlib-inline	0.1.6
mistune	0.8.4
nbclassic	0.3.5
nbclient	0.5.13
nbconvert	6.4.4
nbformat	5.3.0
nest-asyncio	1.5.5
notebook	6.4.12
numpy	1.19.5
oauthlib	3.2.2
opt-einsum	3.3.0
packaging	21.3
pandocfilters	1.5.0
parso	0.8.3
pexpect	4.8.0
pickleshare	0.7.5
pip	20.0.2
pluggy	1.0.0
prometheus-client	0.14.1
prompt-toolkit	3.0.20
protobuf	3.19.0
ptyprocess	0.7.0

pyasn1	0.4.8
pyasn1-modules	0.2.8
pycosat	0.6.3
pycparser	2.20
Pygments	2.11.2
pyOpenSSL	19.1.0
pyparsing	3.0.9
pyrsistent	0.18.0
PySocks	1.7.1
python-dateutil	2.8.2
pytz	2022.1
pyzmq	22.3.0
requests	2.23.0
requests-oauthlib	1.3.1
rsa	4.9
ruamel-yaml	0.15.87
ruamel.yaml	0.16.12
ruamel.yaml.clib	0.2.6
Send2Trash	1.8.0
setuptools	65.6.3
six	1.15.0
sniffio	1.2.0
soupsieve	2.3.1
tensorboard	2.11.0
tensorboard-data-server	0.6.1
tensorboard-plugin-wit	1.8.1
tensorflow	2.5.0
tensorflow-estimator	2.5.0
termcolor	1.1.0

terminado	0.13.1
testpath	0.6.0
tomlkit	0.11.6
toolz	0.11.2
tornado	6.1
tqdm	4.46.0
traitlets	5.1.1
typing-extensions	3.7.4.3
urllib3	1.25.8
wcwidth	0.2.5
webencodings	0.5.1
websocket-client	0.58.0
Werkzeug	2.2.2
wheel	0.34.2
wrapt	1.12.1
zipp	3.8.0

- tensorflow2 环境 python 包列表。

Package	Version
absl-py	0.15.0
alembic	1.9.1
astunparse	1.6.3
backcall	0.2.0
bitarray	2.6.2
cached-property	1.5.2
cachetools	5.2.0
certifi	2022.6.15
charset-normalizer	2.1.1
click	8.1.3

cloudpickle	2.2.0
cx-Oracle	8.0.0
cycler	0.11.0
databricks-cli	0.17.4
decorator	5.1.1
docker	6.0.1
entrypoints	0.4
Flask	2.2.2
flatbuffers	1.12
future	0.18.2
gast	0.4.0
gitdb	4.0.10
GitPython	3.1.30
google-auth	2.15.0
google-auth-oauthlib	0.4.6
google-pasta	0.2.0
greenlet	2.0.1
grpcio	1.34.1
gunicorn	20.1.0
h5py	3.1.0
idna	3.4
imageio	2.23.0
importlib-metadata	6.0.0
importlib-resources	5.10.2
impyla	0.16.2
ipykernel	5.4.3
ipython	7.34.0
ipython-genutils	0.2.0
itsdangerous	2.1.2

jedi	0.18.2
Jinja2	3.1.2
joblib	1.2.0
jupyter_client	7.4.8
jupyter_core	4.12.0
keras-nightly	2.5.0.dev2021032900
Keras-Preprocessing	1.1.2
kiwisolver	1.4.4
libaiflow	5.4
Mako	1.2.4
Markdown	3.4.1
MarkupSafe	2.1.1
matplotlib	3.1.1
matplotlib-inline	0.1.6
mlflow	1.26.2.dev0
nest-asyncio	1.5.6
networkx	2.6.3
numpy	1.19.5
oauthlib	3.2.2
opencv-python	4.2.0.32
opt-einsum	3.3.0
packaging	22.0
pandas	1.1.0
parso	0.8.3
pexpect	4.8.0
pickleshare	0.7.5
Pillow	9.4.0
pip	22.1.2
plotly	5.11.0

ply	3.11
prometheus-client	0.15.0
prometheus-flask-exporter	0.21.0
prompt-toolkit	3.0.36
protobuf	3.19.0
psycopg2-binary	2.9.3
ptyprocess	0.7.0
pyasn1	0.4.8
pyasn1-modules	0.2.8
pycryptodome	3.15.0
Pygments	2.14.0
PyHDFS	0.3.1
PyHive	0.6.2
PyJWT	2.6.0
PyMySQL	0.9.3
pyparsing	3.0.9
python-dateutil	2.8.2
pytz	2022.7
PyWavelets	1.3.0
PyYAML	6.0
pyzmq	24.0.1
querystring-parser	1.2.4
requests	2.28.1
requests-oauthlib	1.3.1
rsa	4.9
scikit-image	0.19.3
scikit-learn	0.21.3
scipy	1.4.1
setuptools	63.4.1

simplejson	3.18.0
six	1.15.0
smmap	5.0.0
SQLAlchemy	1.4.45
sqlparse	0.4.3
tabulate	0.9.0
tenacity	8.1.0
tensorboard	2.11.0
tensorboard-data-server	0.6.1
tensorboard-plugin-wit	1.8.1
tensorflow	2.5.0
tensorflow-estimator	2.5.0
tensorflow-gpu	2.5.0
termcolor	1.1.0
thrift	0.16.0
thriftpy2	0.4.11
tifffile	2021.11.2
tornado	6.2
traitlets	5.8.0
typing-extensions	3.7.4.3
urllib3	1.26.13
wcwidth	0.2.5
websocket-client	1.4.2
Werkzeug	2.2.2
wheel	0.37.1
wrapt	1.12.1
zipp	3.11.0

- pytorch 环境 python 包列表。

Package	Version
absl-py	1.3.0
aiohttp	3.8.3
aiosignal	1.3.1
alembic	1.9.1
async-timeout	4.0.2
asynctest	0.13.0
attrs	22.2.0
backcall	0.2.0
cachetools	4.2.4
certifi	2022.6.15
charset-normalizer	2.1.1
click	8.1.3
cloudpickle	2.2.0
cycler	0.11.0
databricks-cli	0.17.4
decorator	5.1.1
docker	6.0.1
entrypoints	0.4
Flask	2.2.2
fonttools	4.38.0
frozenlist	1.3.3
fsspec	2022.11.0
gitdb	4.0.10
GitPython	3.1.30
google-auth	1.35.0
google-auth-oauthlib	0.4.6
greenlet	2.0.1
grpcio	1.51.1

gunicorn	20.1.0
idna	3.4
imageio	2.23.0
importlib-metadata	6.0.0
importlib-resources	5.10.2
ipykernel	5.4.3
ipython	7.34.0
ipython-genutils	0.2.0
itsdangerous	2.1.2
jedi	0.18.2
Jinja2	3.1.2
jupyter_client	7.4.8
jupyter_core	4.12.0
kiwisolver	1.4.4
libaiflow	5.4
Mako	1.2.4
Markdown	3.4.1
MarkupSafe	2.1.1
matplotlib	3.5.3
matplotlib-inline	0.1.6
mlflow	1.26.2.dev0
multidict	6.0.4
nest-asyncio	1.5.6
networkx	2.6.3
numpy	1.21.6
nvidia-cublas-cu11	11.10.3.66
nvidia-cuda-nvrtc-cu11	11.7.99
nvidia-cuda-runtime-cu11	11.7.99
nvidia-cudnn-cu11	8.5.0.96

oauthlib	3.2.2
opencv-python	4.2.0.32
packaging	22.0
pandas	1.3.5
parso	0.8.3
pexpect	4.8.0
pickleshare	0.7.5
Pillow	9.4.0
pip	22.1.2
plotly	5.11.0
prometheus-client	0.15.0
prometheus-flask-exporter	0.21.0
prompt-toolkit	3.0.36
protobuf	3.20.1
psycopg2-binary	2.9.3
ptyprocess	0.7.0
pyasn1	0.4.8
pyasn1-modules	0.2.8
pycryptodome	3.15.0
pyDeprecate	0.3.2
Pygments	2.14.0
PyJWT	2.6.0
pyparsing	3.0.9
python-dateutil	2.8.2
pytorch-lightning	1.7.1
pytz	2022.7
PyWavelets	1.3.0
PyYAML	6.0
pyzmq	24.0.1

querystring-parser	1.2.4
requests	2.28.1
requests-oauthlib	1.3.1
rsa	4.9
scikit-image	0.19.3
scipy	1.4.1
setuptools	59.5.0
six	1.16.0
smmap	5.0.0
SQLAlchemy	1.4.45
sqlparse	0.4.3
tabulate	0.9.0
tenacity	8.1.0
tensorboard	2.3.0
tensorboard-data-server	0.6.1
tensorboard-plugin-wit	1.8.1
tifffile	2021.11.2
torch	1.10.0+cu113
torchaudio	0.10.0+cu113
torchmetrics	0.11.0
torchvision	0.11.1+cu113
tornado	6.2
tqdm	4.64.1
traitlets	5.8.0
typing_extensions	4.4.0
urllib3	1.26.13
wcwidth	0.2.5
websocket-client	1.4.2
Werkzeug	2.2.2

wheel	0.37.1
yarl	1.8.2
zipp	3.11.0

（十一）jhinno/training-center:v5.4

- 镜像说明：实验管理镜像，无 python 环境。

（十二）jhinno/makesense:v1.0

- 镜像说明：图像标注功能依赖镜像，无 python 环境。

（十三）jhinno/portainer:v5.4

- 镜像说明：容器管理功能依赖镜像，无 python 环境。

（十四）jhinno/traefik:v2.4.8

- 镜像说明：启动 traefik 代理所用镜像。

附录六：libaiflow API

（一）实验管理相关 API

实验初始化

```
libaiflow.init(experiment_name: str, experiment_tags: Union[Dict[str, Any], NoneType] = None) -> Experiment
```

参数：

experiment_name：实验名称，实验名称具有唯一性且区分大小写，一个有效的实验名称可以包含 a-z、A-Z、全中文、0-9、_和-，最大长度为 255 个字符。

experiment_tags：设置实验标记。

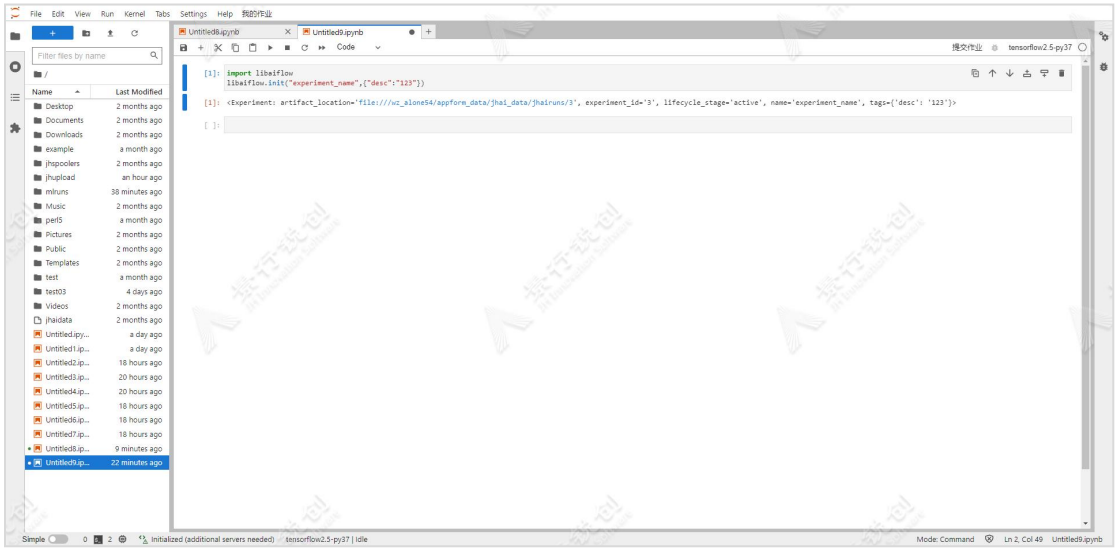
返回：

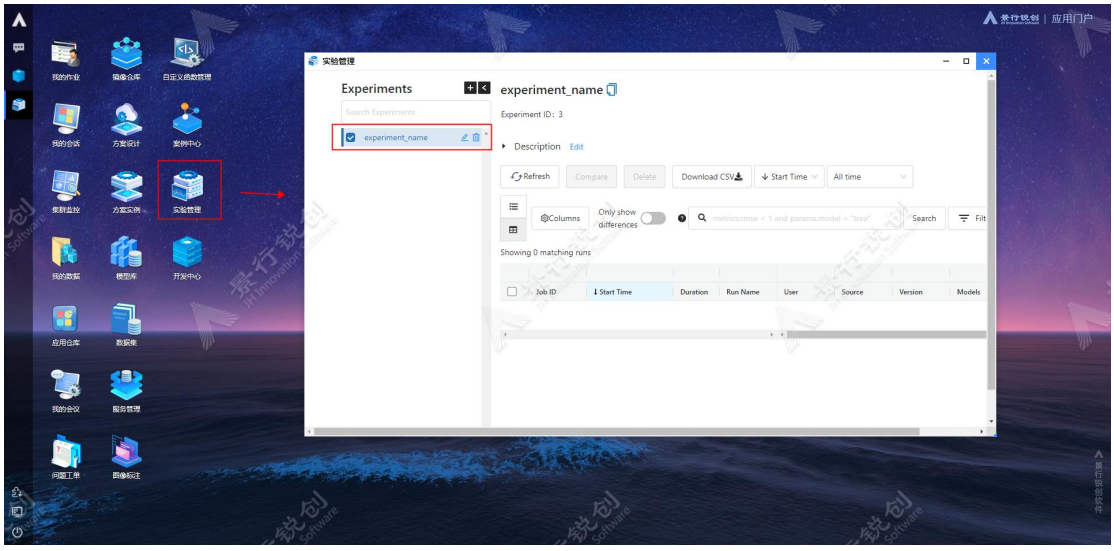
返回实验（Experiment）对象

样例：

```
import libaiflow
libaiflow.init("experiment_name",{"desc":"123"})
```

具体操作详情，如下图所示：





启动一个运行实例

启动一个新的运行，将其设置为活动运行，在活动运行下将记录指标和参数。返回值可以结合 with 块一起使用；否则，必须调用 end_run()来终止当前运行。（类似文件 IO 的 open 和 close），使用 with 方式可避免忘记结束 run。

```
libaiflow.start_run(experiment_id: Union[str, NoneType], run_name:
 Union[str, NoneType] = None, nested: bool = False, tags: Union[Di
ct[str, Any], NoneType] = None, description: Union[str, NoneType]
= None) -> ActiveRun
```

参数：

experiment_id：运行实例所属实验的实验 id，此参数是必须的。

run_name：运行实例名称。

nested：控制“运行”是否嵌套在“父运行”中。如果 True 创建嵌套运行。

tags：一个可选的字典，包含在运行时设置为标记的字符串键和值。如果正在恢复运行，则在恢复的运行上设置这些标记。如果正在创建新运行，则在新运行上设置这些标记。

description：填充运行的描述框的可选字符串。如果正在恢复运行，则在恢复的运行上设置描述。如果正在创建新运行，则在新运行上设置描述。

返回：

返回 libaiflow.ActiveRun 对象，封装的运行的状态。

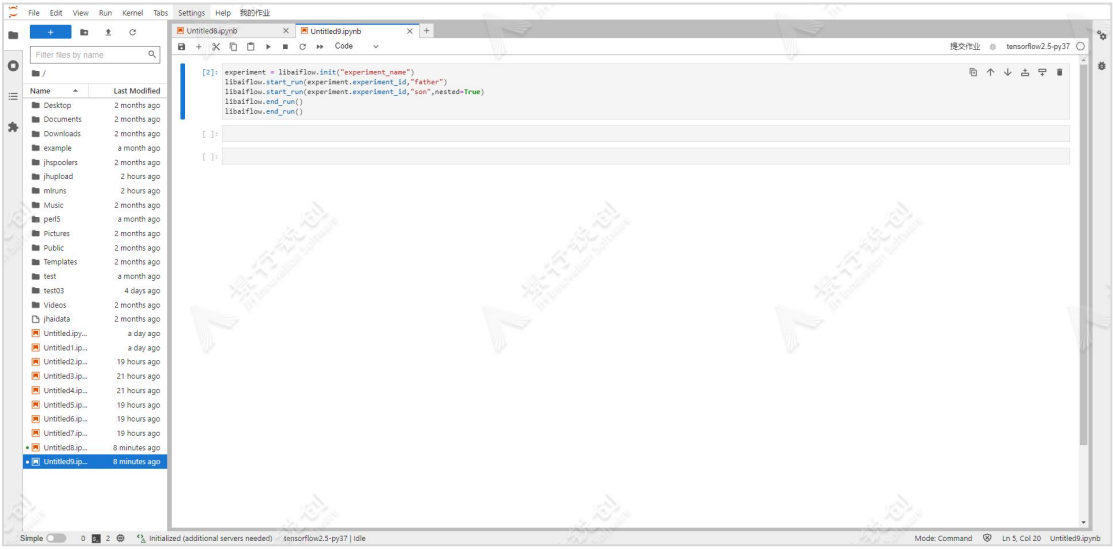
样例：

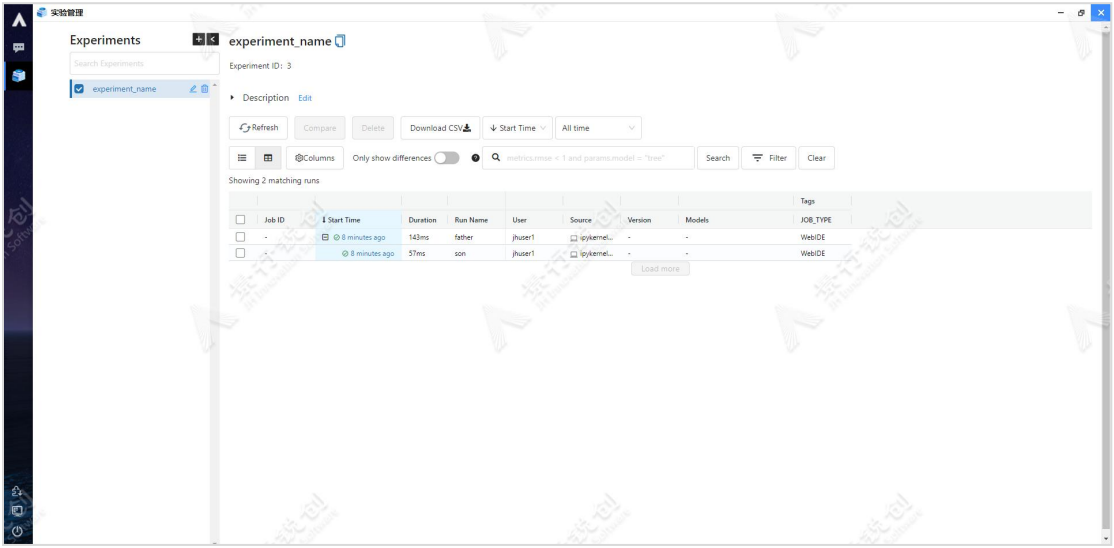
```
experiment = libaiflow.init("experiment_name")
libaiflow.start_run(experiment.experiment_id,"father")
libaiflow.set_tag("type", "father")
libaiflow.start_run(experiment.experiment_id,"son",nested=True)
libaiflow.set_tag("type", "son")
libaiflow.end_run()
libaiflow.end_run()
```

或者

```
experiment = libaiflow.init("experiment_name")
with libaiflow.start_run(experiment.experiment_id,"father"):
    libaiflow.set_tag("type", "father")
    with libaiflow.start_run(experiment.experiment_id,"son",nested=
True):
        libaiflow.set_tag("type", "son")
```

具体操作详情，如下图所示：





（二）指标和超参数相关 API

记录当前运行下的指标。如果没有活动的运行，此方法将创建一个新的活动运行。

```
libaiflow.log_metric(key: str, value: float, step: Union[int, None Type] = None) -> None
```

参数：

key：Metric 名称(String)。该字符串只能包含字母数字、下划线(_)、破折号(-)、句号(.)、空格()和斜线(/)。所有后端存储都将支持长度不超过 250 的键。

value：Metric 值(float)。所有后端存储将支持长度不超过 5000 的值。

step：Metric step(int)如果未指定，默认为零。

返回：

无返回值

样例：

```
experiment = libaiflow.init("experiment_name")
with libaiflow.start_run(experiment.experiment_id):
    log_metric("mse", 2500.00)
```

记录当前运行下的多个指标。如果没有活动的运行，此方法将创建一个新的活动运行。

```
libaiflow.log_metrics(metrics: Dict[str, float], step: Union[int, NoneType] = None) -> None
```

参数：

metrics：字典类型的 Metric: Metric 名称(String) -> Metric 值(Float)

step：记录 Metric 步长值。如果未指定，则在步骤 0 记录每个指标

返回：

无返回值

样例：

```
experiment = libaiflow.init("experiment_name")
metrics = {"mse": 2500.00, "rmse": 50.00}
with libaiflow.start_run(experiment.experiment_id):
    libaiflow.log_metrics(metrics)
```

记录当前运行下的一个参数。如果没有活动的运行，此方法将创建一个新的活动运行。

```
libaiflow.log_param(key: str, value: Any) -> None
```

参数：

key：参数名称(String)。该字符串只能包含字母数字、下划线(_)、破折号(-)、句号(.)、空格()和斜线(/)。所有后端存储都将支持长度不超过 250 的键。

value：参数值(字符串，如果不是将被字符串化)。所有后端存储将支持长度不超过 5000 的值。

返回：

无返回值

样例：

```
experiment = libaiflow.init("experiment_name")
with libaiflow.start_run(experiment.experiment_id):
    libaiflow.log_param("learning_rate", 0.01)
```

记录当前运行的一批参数。如果没有活动的运行，此方法将创建一个新的活动运行。

```
libaiflow.log_params(params: Dict[str, Any]) -> None
```

参数：

params：字典类型的 param: Param 名称（String） -> Param 值:(字符串，如果不是将被字符串化)。

返回：

无返回值

样例：

```
experiment = libaiflow.init("experiment_name")
params = {"learning_rate": 0.01, "n_estimators": 10}
with libaiflow.start_run(experiment.experiment_id):
    libaiflow.log_params(params)
```

（三）文件和可视化相关 API

将本地文件或目录记录为当前活动运行的 Artifact。如果没有活动的运行，此方法将创建一个新的活动运行。

```
libaiflow.log_artifact(local_path: str, artifact_path: Union[str,
NoneType] = None) -> None
```

参数：

local_path：要写入的文件的路径。

artifact_path：如果指定目录，将写入到 artifact_uri 中的目录。

返回：

无返回值

样例：

```
experiment = libaiflow.init("experiment_name")
features = "rooms, zipcode, median_price, school_rating, transport"
with open("features.txt", 'w') as f:
    f.write(features)

with libaiflow.start_run(experiment.experiment_id):
    libaiflow.log_artifact("features.txt","data")
```

日志文本作为 artifact。

```
libaiflow.log_text(text: str, artifact_file: str) -> None
```

参数：

text：包含要记录的文本的字符串。

artifact_file：文本保存的 posixpath 格式的运行相关工件文件路径(例如“dir/file.txt”)。

返回：

无返回值

样例：

```
experiment = libaiflow.init("experiment_name")
with libaiflow.start_run(experiment.experiment_id):
    # Log text to a file under the run's root artifact directory
    libaiflow.log_text("text1", "file1.txt")

    # Log text in a subdirectory of the run's root artifact directory
    libaiflow.log_text("text2", "dir/file2.txt")

    # Log HTML text
    libaiflow.log_text("<h1>header</h1>", "index.html")
```

将 figure 对象作为 Artifact

```
libaiflow.log_figure(figure: Union[ForwardRef('matplotlib.figure.F
igure'), ForwardRef('plotly.graph_objects.Figure')], artifact_fil
e: str) -> None
```

参数：

figure：figure 对象，支持以下 figure 对象：

- matplotlib.figure.Figure
- plotly.graph_objects.Figure

artifact_file：以 posixpath 格式保存图形的运行相关工件文件路径(例如“dir/file.png”)

返回：

无返回值

样例：

Matplotlib 样例代码如下所示：

```
import libaiflow
import matplotlib.pyplot as plt

experiment = libaiflow.init('experiment_name')

fig, ax = plt.subplots()
ax.plot([0, 1], [2, 3])

with libaiflow.start_run(experiment.experiment_id):
    libaiflow.log_figure(fig, "figure.png")
```

Plotly 样例代码如下所示：

```
import libaiflow
from plotly import graph_objects as go


experiment = libaiflow.init('experiment_name')


fig = go.Figure(go.Scatter(x=[0, 1], y=[2, 3]))


with libaiflow.start_run(experiment_id= experiment. experiment_id):
    libaiflow.log_figure(fig, "figure.html")
```

将 image 对象记录为 Artifact

```
libaiflow.log_image(image: Union[ForwardRef('numpy.ndarray'), Forw
ardRef('PIL.Image.Image')], artifact_file: str) -> None
```

参数：

image：figure 对象，支持以下 image 对象：

- numpy.ndarray
- PIL.Image.Image

artifact_file：以 posixpath 格式保存图形的运行相关工件文件路径(例如“dir/file.png”)

返回：

无返回值

样例：

Numpy 样例代码如下所示：

```
import libaiflow
import numpy as np

experiment = libaiflow.init('experiment_name')

image = np.random.randint(0, 256, size=(100, 100, 3), dtype=np.uint8)

with libaiflow.start_run(experiment.experiment_id):
    libaiflow.log_image(image, "image.png")
```

Pillow 样例代码如下所示：

```
import libaiflow
from PIL import Image

experiment = libaiflow.init('experiment_name')

image = Image.new("RGB", (100, 100))

with libaiflow.start_run(experiment.experiment_id):
    libaiflow.log_image(image, "image.png")
```

（四）模型相关 API

启用自动记录 tensorflow 和 tf.keras 的参数， 指标和模型。

```
libaiflow.tensorflow.autolog(every_n_iter=1)
```

参数：

every_n_iter：它是训练指标的每个日志之间的训练时期数。 例如，如果传递的值为 2，则每 2 个时期记录一次训练指标(损失，准确性和验证损失等)。

返回：

没有返回值

启用自动记录 PyTorch Lightning 的参数、指标和模型。

```
libaiflow.pytorch.autolog(log_every_n_epoch=1)
```

参数：

log_every_n_epoch：它是训练指标的每个日志之间的训练时期数。 例如，如果传递的值为 2，则每 2 个时期记录一次训练指标(损失，准确性和验证损失等)。

返回：

没有返回值

（五）数据集相关 API

根据数据集名称获取数据集信息

```
libaiflow.get_dataset_by_name(dataset_name: str) -> Union[Dict[str, Any], NoneType]
```

参数：

dataset_name：数据集名称。

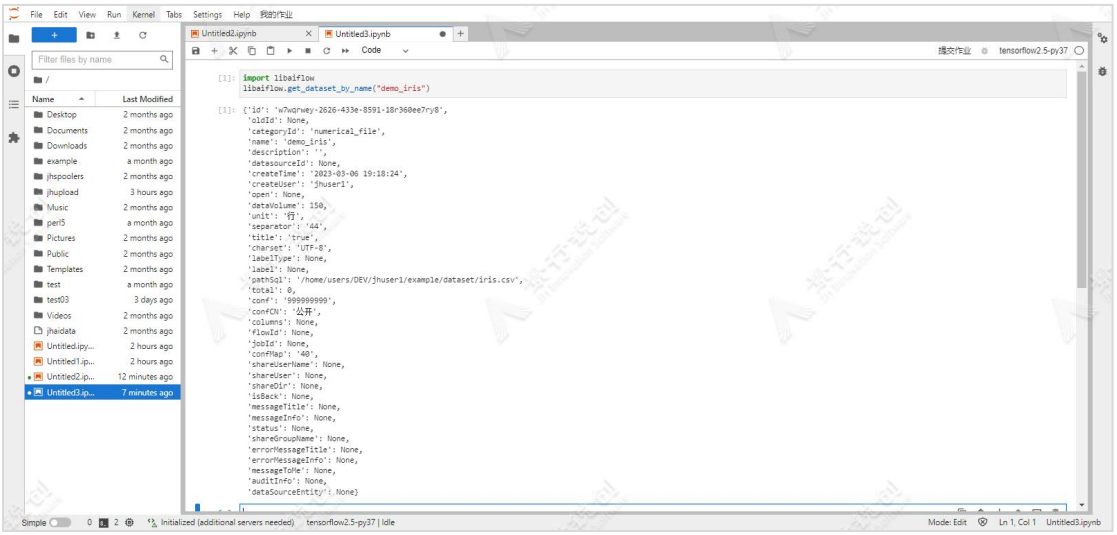
返回：

返回 JSON 格式的数据集信息

样例：

```
import libaiflow
libaiflow.get_dataset_by_name("demo_iris")
```

具体操作详情，如下图所示：



根据数据集 ID 获取数据集信息

```
Libaiflow.get_dataset_by_id(dataset_id: str) -> Union[Dict[str, Any], NoneType]
```

参数：

dataset_id：数据集 ID

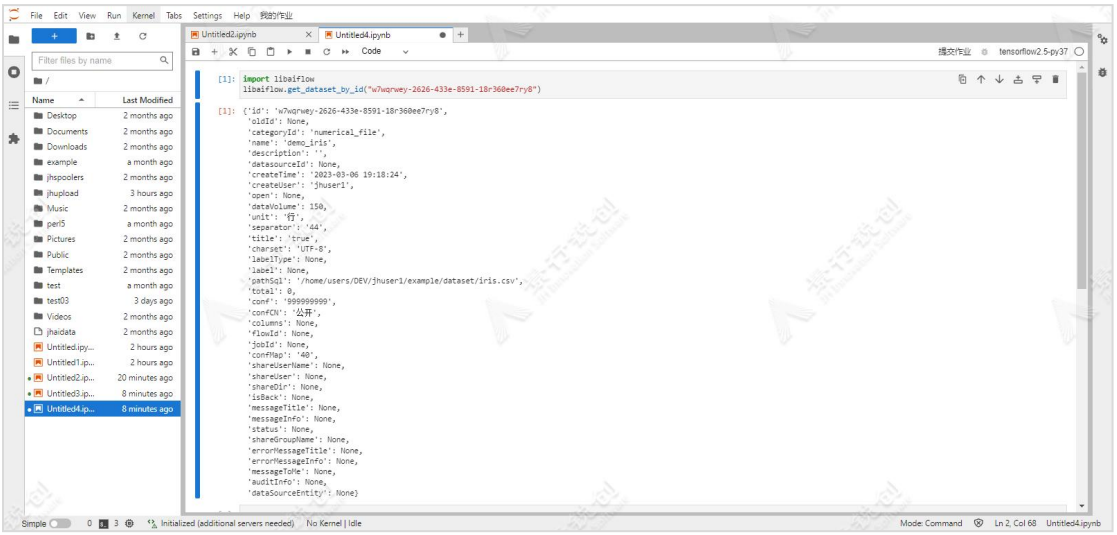
返回：

返回 JSON 格式的数据集信息

样例：

```
import libaiflow
libaiflow.get_dataset_by_id("w7wqrwey-2626-433e-8591-18r360ee7ry8"
)
```

具体操作详情，如下图所示：



创建数据集，目前仅支持创建“数值数据-数据文件”数据集：

```
libaiflow.create_dataset(name: Union[str, NoneType], file_path: Union[str, NoneType], description: Union[str, NoneType] = '', category: Union[libaiflow.client.datasets.Category, NoneType] = <Category.numerical_file: 'numerical_file'>, separator: Union[libaiflow.client.datasets.Separator, NoneType] = <Separator.comma: '44'>, title: Union[bool, NoneType] = True, charset: Union[libaiflow.client.datasets.Charset, NoneType] = <Charset.UTF8: 'UTF-8'>) -> Union[Dict[str, Any], NoneType]
```

参数：

name：数据集名称。

file_path：数据文件路径。

description：数据集描述。

category：数据集类型，默认为 numerical_file，用于创建“数值数据-数据文件”数据集；目前仅支持创建“数值数据-数据文件”数据集。

separator：分隔符，默认为逗号，包含以下分隔符：

- 逗号：Separator.comma
- 空格：Separator.space
- 分号：Separator.semicolon
- 制表符：Separator.tab
- 下划线：Separator.underline
- 中划线：Separator.minus

title：是否有表头，默认有。

charset：字符集，默认为 UTF-8，包括以下字符集：

- UTF-8：Charset.UTF8
- GB2312：Charset.GB2312
- GBK：Charset.GBK
- ISO-8859-1：Charset.ISO8895_1

UTF-16：Charset.UTF16

UTF-32：Charset.UTF32

Big5：Charset.BIG5

US_ASCII：Charset.US_ASCII

返回：

返回 JSON 格式的数据集信息

样例：

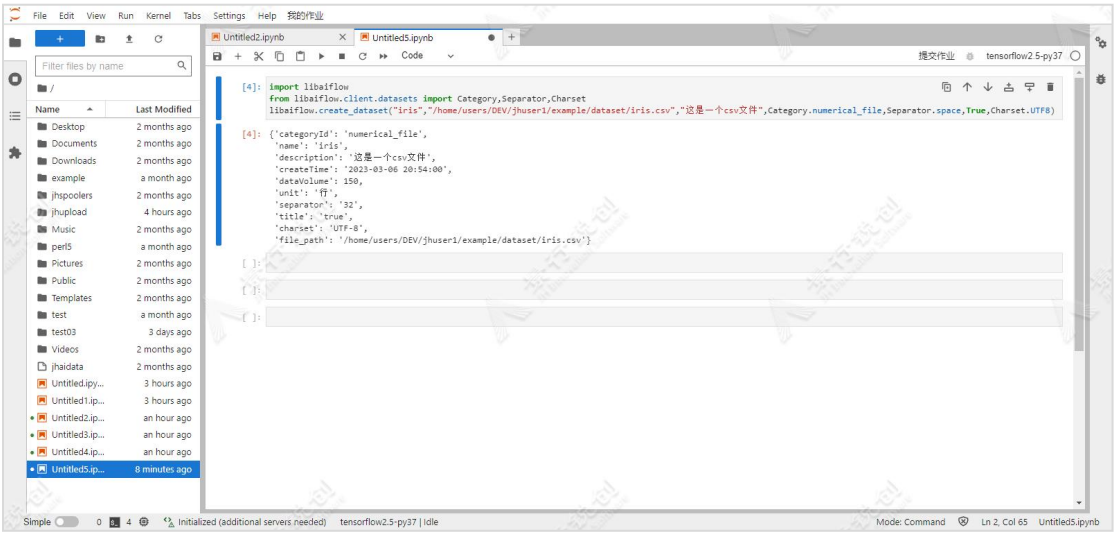
样例代码如下所示：

```
import libaiflow
from libaiflow.client.datasets import Category,Separator,Charset
libaiflow.create_dataset("iris","/home/users/DEV/jhuser1/example/dataset/iris.csv","这是一个 csv 文件",Category.numerical_file,Separator.space,True,Charset.UTF8)
```

或者

```
import libaiflow
from libaiflow.client.datasets import Category,Separator,Charset
libaiflow.create_dataset("iris","/home/users/DEV/jhuser1/example/dataset/iris.csv","这是一个 csv 文件",separator=Separator.space,charset=Charset.UTF8)
```

具体操作详情，如下图所示：



修改数据集

```
libaiflow.modify_dataset(name: Union[str, NoneType], file_path: Union[str, NoneType] = None, description: Union[str, NoneType] = None, category: Union[libaiflow.client.datasets.Category, NoneType] = None, separator: Union[libaiflow.client.datasets.Separator, NoneType] = None, title: Union[bool, NoneType] = None, charset: Union[libaiflow.client.datasets.Charset, NoneType] = None) -> Union[Dict[str, Any], NoneType]
```

参数：

name：数据集名称。

file_path：数据文件路径。

description：数据集描述。

category：数据集类型，默认为 numerical_file，用于创建“数值数据-数据文件”数据集；目前仅支持创建“数值数据-数据文件”数据集。

separator：分隔符，默认为逗号，包含以下分隔符：

逗号：Separator.comma

空格：Separator.space

分号：Separator.semicolon

制表符：Separator.tab

下划线：Separator.underline

中划线：Separator.minus

title：是否有表头，默认有。

charset：字符集，默认为 UTF-8，包括以下字符集：

UTF-8：Charset.UTF8

GB2312：Charset.GB2312

GBK: Charset.GBK

ISO-8859-1：Charset.ISO8895_1

UTF-16: Charset.UTF16

UTF-32: Charset.UTF32

Big5: Charset.BIG5

US_ASCII: Charset.US_ASCII

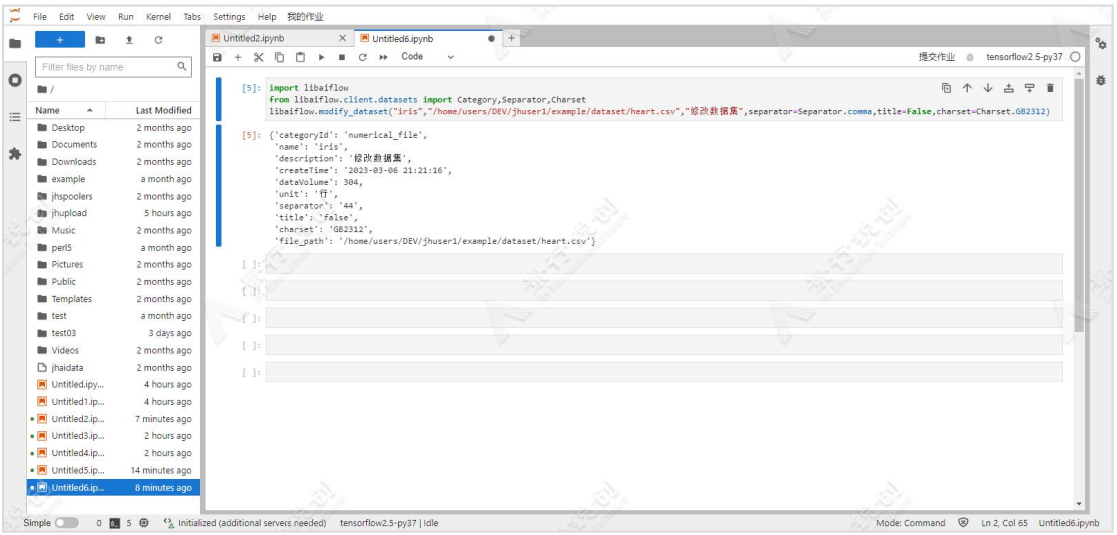
返回：

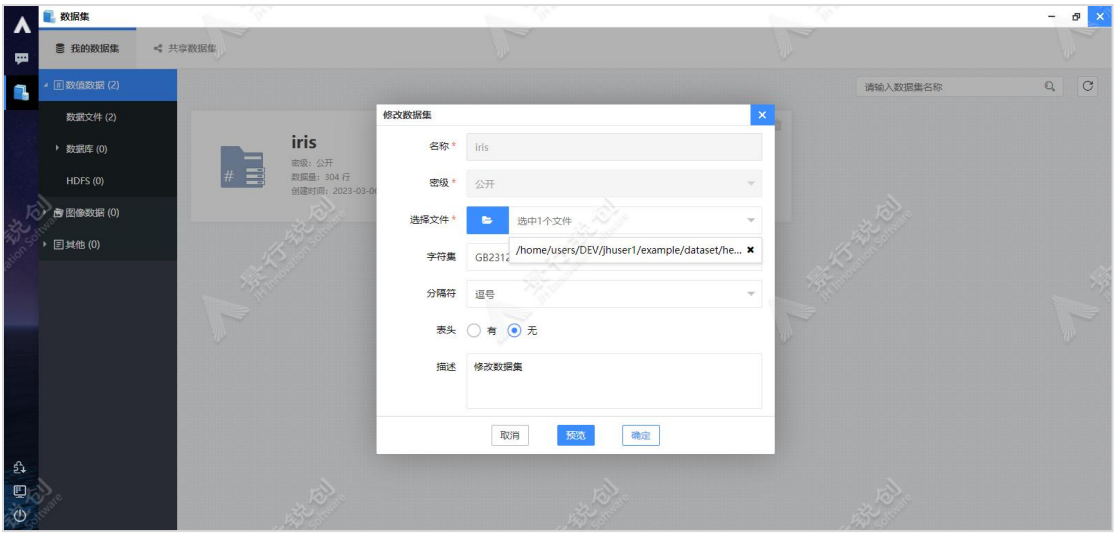
返回 JSON 格式的数据集信息

样例：

样例代码如下所示：

```
import libaiflow
from libaiflow.client.datasets import Category,Separator,Charset
libaiflow.modify_dataset("iris","/home/users/DEV/jhuser1/example/dataset/heart.csv","修改数据集",separator=Separator.comma,title=False,charset=Charset.GB2312)
```





删除“数值数据-数据文件”数据集

```
libaiflow.delete_dataset(name: str = None)
```

参数：

name：“数值数据-数据文件”数据集名称。

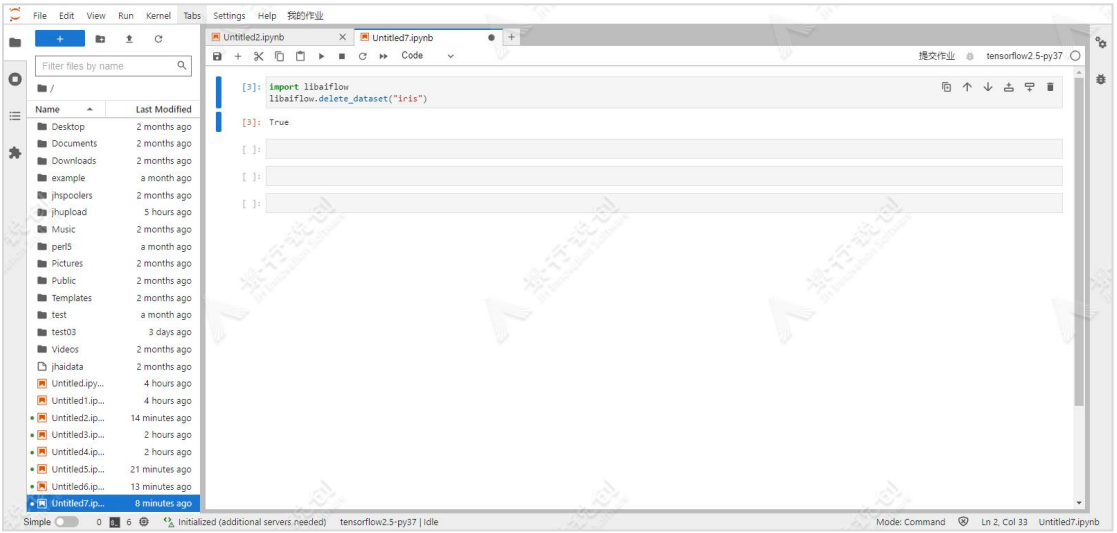
返回：

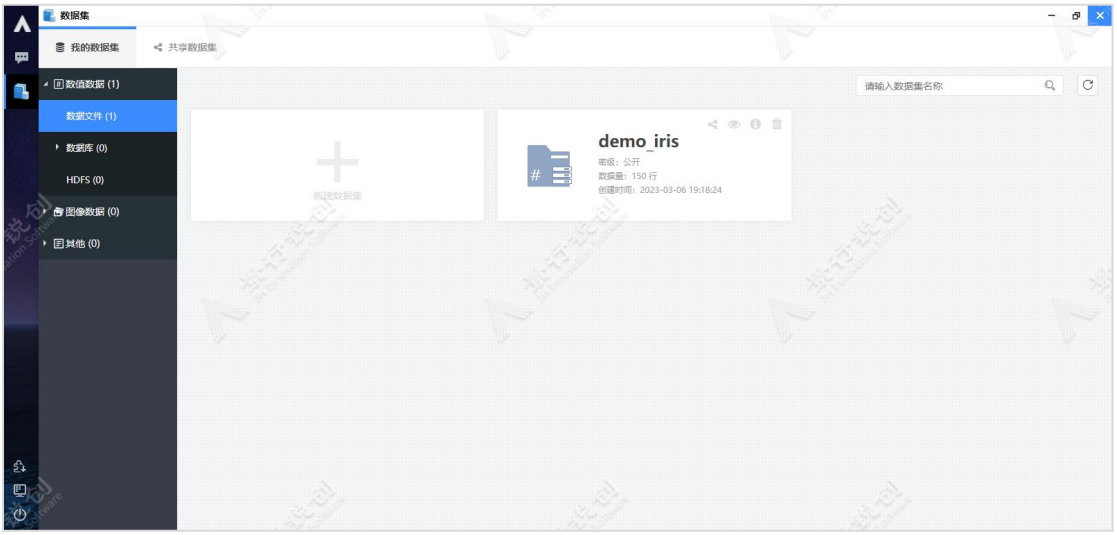
返回 True，代表着删除成功了。

样例：

```
import libaiflow
libaiflow.delete_dataset("iris")
```

具体操作详情，如下图所示：




数据集
我的数据集
共享数据集
数值数据 (1)
数据文件 (1)
数据库 (0)
HDFS (0)
图像数据 (0)
其他 (0)
请输入数据集名称
新建数据集
demo_iris
密级：公开
数据量：150 行
创建时间：2023-03-06 19:18:24

附录七：实验管理 SDK 代码样例

样例一：

```
import tensorflow as tf
from tensorflow.keras import layers, Sequential, optimizers, losses,
metrics, datasets
import os
import numpy as np
import libaiflow
# 预处理函数
def preprocess(x, y):
    x = tf.cast(x, tf.float32) / 255.0 # 将 MNIST 数据映射到[0, 1]
    x = tf.expand_dims(x, axis=-1) # 由于卷积层维度为[None, 28, 28, 1],
故在 axis=3 扩展一维
    y = tf.one_hot(y, depth=10)
    return x, y

# 加载数据
def load_data():
  path = "./data/mnist.npz"
  with np.load(path) as f:
    x_train, y_train = f['x_train'], f['y_train']
    x_test, y_test = f['x_test'], f['y_test']
    return (x_train, y_train), (x_test, y_test)

expt = libaiflow.init(experiment_name="mnist_tf2_cpu")
libaiflow.tensorflow.autolog()

(x, y), (x_test, y_test) = load_data()

print(x.shape, y.shape, x.min(), x.max()) # 显示加载进来的数据的维度信
息，便于理解

train_db = tf.data.Dataset.from_tensor_slices((x, y))
train_db = train_db.shuffle(1000).map(preprocess).batch(100)

test_db = tf.data.Dataset.from_tensor_slices((x_test, y_test))
test_db = test_db.map(preprocess).batch(100)

# 创建网络模型并装配模型
network = Sequential([
```

```
    layers.Conv2D(6,    kernel_size=5,    strides=1,    padding='SAME',
activation='relu'),
    layers.MaxPooling2D(pool_size=2, strides=2),
    layers.Conv2D(16,    kernel_size=5,    strides=1,    padding='SAME',
activation='relu'),
    layers.MaxPooling2D(pool_size=2, strides=2),

    layers.Flatten(),

    layers.Dense(256, activation='relu'),
    layers.Dense(64, activation='relu'),
    layers.Dense(10, activation=None)
])
network.compile(optimizer=optimizers.Adam(lr=0.01),

loss=losses.CategoricalCrossentropy(from_logits=True),
                metrics=['accuracy'])

tf.summary.trace_on() #显示结构图

# 显示网络结构信息
network.build(input_shape=[None, 28, 28, 1])
network.summary()
pb_file_path = "model"
if not os.path.exists(pb_file_path):  #判断是否存在文件夹如果不存在则
创建为文件夹
    os.makedirs(pb_file_path)

# 设置回调功能
filepath = './model/my_model.h5' # 保存模型地址

saved_model = tf.keras.callbacks.ModelCheckpoint(filepath, verbose = 1)
# 回调保存模型功能
tensorboard = tf.keras.callbacks.TensorBoard(log_dir = './logs') # 回
调可视化数据功能

# 执行训练与验证
history = network.fit(train_db, epochs = 2, validation_data = test_db,
validation_freq = 1,
                      callbacks = [saved_model, tensorboard])
```

样例二：

```
import logging
import libaiflow
import pandas as pd
from sklearn.model_selection import train_test_split
import sys
from sklearn.linear_model import ElasticNet
import numpy as np
from                        sklearn.metrics                        import
mean_squared_error,mean_absolute_error,r2_score

logging.basicConfig(level=logging.WARN)
logger = logging.getLogger(__name__)

def eval_metrics(actual, pred):
    rmse = np.sqrt(mean_squared_error(actual, pred))
    mae = mean_absolute_error(actual, pred)
    r2 = r2_score(actual, pred)
    return rmse, mae, r2

if __name__ == "__main__":
   np.random.seed(40)
   # Read the wine-quality csv file from the URL
   csv_url
='http://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality
/winequality-red.csv'
   try:
       data = pd.read_csv(csv_url, sep=';')
   except Exception as e:
       logger.exception(
           "Unable to download training & test CSV, check your internet
connection. Error: %s", e)

   # Split the data into training and test sets. (0.75, 0.25) split.
   train, test = train_test_split(data)

   # The predicted column is "quality" which is a scalar from [3, 9]
   train_x = train.drop(["quality"], axis=1)
   test_x = test.drop(["quality"], axis=1)
   train_y = train[["quality"]]
   test_y = test[["quality"]]
   alpha = 0.5
   l1_ratio = 0.5
   expt = libaiflow.init(experiment_name="mnist_tf2_test1")
```

```
   with libaiflow.start_run(expt.experiment_id):
       lr = ElasticNet(alpha=alpha, l1_ratio=l1_ratio, random_state=42)
       lr.fit(train_x, train_y)
       predicted_qualities = lr.predict(test_x)
       (rmse, mae, r2) = eval_metrics(test_y, predicted_qualities)
       print("Elasticnet  model  (alpha=%f,  l1_ratio=%f):"  %  (alpha,
l1_ratio))
       print("  RMSE: %s" % rmse)
       print("  MAE: %s" % mae)
       print("  R2: %s" % r2)
       libaiflow.log_param("alpha", alpha)
       libaiflow.log_param("l1_ratio", l1_ratio)
       libaiflow.log_metric("rmse", rmse)
       libaiflow.log_metric("r2", r2)
       libaiflow.log_metric("mae", mae)
```

附录八：常见问题及解决办法

1. 应用 3D 可视化功能，报：“浏览器未开启 WebGL 功能，请检查浏览器设置”

人工智能平台支持对卷积神经网络模型进行 3D 可视化，但是 3D 可视化功能依赖浏览器的 WebGL 功能。如果使用 Google Chrome84 以上版本，却无法使用 3D 可视化功能，需要在 Google Chrome 中打开 WebGL2 功能。

解决方案：

首先，测试下浏览器是否支持 WebGL2。在线检测，请在浏览器地址栏中输入以下网址：

```
https://webglreport.com/?v=2
```

当出现下列字样时，说明你的浏览器支持 WebGL2，如下所示：

✓ This browser supports WebGL 2

启用 WebGL，在 Google Chrome 中设置 WebGL2 相关参数，浏览器输入

```
chrome://flags
```

搜索参数并设置

```
WebGL Draft Extensions 设置为 Enable
Choose ANGLE graphics backend 设置为 OpenGL
```

2. 如何在容器服务中使用用户自己的数据

解决方案：

容器服务启动时会默认挂载用户家目录、作业数据区、共享数据区，可以在启动的容器服务中直接访问用户家目录、作业数据区、共享数据区下的

数据文件，目录绝对路径和宿主机保持一致。

3. **如何保存在容器服务中创建的文件**

解决方案：

容器服务启动时会默认挂载用户家目录、作业数据区和共享数据区，可以将需要保存的文件拷贝到用户家目录、作业数据区或者共享数据区下。

4. **如何在离线内网环境安装额外的 pip 或 conda 依赖**

解决方案：

可以从外网下载 pip、 conda 的离线包，导入内网安装。或者购买景行锐创离线 pip、conda 源服务在内网搭建源，方便安装依赖。

5. **如何在非 root 启动的镜像中安装依赖包**

解决方案：

使用 sudo 执行 yum 或 apt-get 命令。

6. **模型训练任务异常退出，显示 out of memory**

解决方案：

调整训练程序代码或数据大小。

7. **容器服务启动失败报错，output.txt 中提示：“service start failed,current status is: rejected, reason is: No such image: ahaha/aaa:v_testa1”**

原因：

创建容器服务时，选择/输入的镜像名是不存在的，或者是浏览器页面缓存的镜像名实际上已经不存在。

解决方案：

清除浏览器缓存，输入/选择真正存在的镜像。

8. **pytorch、tensorflow 等 AI 页面加载失败，没有正常显示**

原因：

appform session 过期，在 portal.log 和 jhai.log 中可以看到过期信息。

解决方案：

重新登录门户。

9. **挂起 TensorFlow、Pytorch 等 AI 作业。作业状态显示挂起，但后台进程仍然运行**

原因：

作业容器启动前无法被挂起。

10. **计算节点无法访问景行应用门户节点外网 IP 时，在 jupyter 容器中，无法使用模型部署的容器服务做推理。**

原因：

jupyter 容器需要能访问景行应用门户节点外网 IP。

解决办法：

允许计算节点访问景行应用门户节点外网 IP 或在 jupyter 中使用门户内网 IP 访问模型部署的服务，自行调整容器服务 url 来使用。内网 IP 的获取需要联系管理员。

11. **数值数据集列名中包含符号.的数据集,预览无法显示数据。**

原因：

不支持列名中包含符号.的数据集。

解决办法：

不要在列名中包含符号。

12. 创建一个新的开发环境，此时开发环境一直启动不了，报错：“The agent was unable to contact any other agent located on a manager node。”

原因：

发现和虚拟机虚拟网卡有关系。

解决办法：

当出现这种情况时关闭网卡的 checksums，使用以下命令执行：

ethtool -K <interface> tx off

13. 容器桌面中，执行 mount，报错：“mount: /ubuntuxwf/: mount failed: Operation not permitted.”

原因：

容器桌面安全考虑，没有开放 privileged 权限。

解决方法：

把需要挂载的文件解压后放到容器中使用。

14. tensorboard 有时打开后为空页面，没有加载出 tensorboard 主页面。

原因：

可能 tensorboard 需要的 json 文件没有加载完。

解决方案：

关闭页面重新打开。

15. 开发环境容器桌面中启动 vscode 闪退。

原因：

vscode 在容器桌面中启动需要使用非隔离模式。

解决办法：

启动参数增加--no-sandbox，例如：./code --no-sandbox

16. 使用 libaiflow.log_figure 保存并记录图表 html 时，在实验管理界面中 html 无法显示。

原因：

plotly 生成的图表 html 文件中对于 plotly 库的引用使用的是 cdn 模式，需要能够联网加载 plotly.js 能正常显示图表。

解决办法：

建议保存和记录 plotly 时使用.png, .jpeg 等图片格式替代 html 格式

17. 桌面容器在 firefox 中获取剪切板内容失败，导致无法粘贴从本地复制的内容

原因：

Firefox 存在剪贴板访问安全限制。

解决方案 A：

可使用门户登录页推荐的浏览器。

解决方案 B：

步骤 1、打开火狐浏览器地址栏输入：about:config

步骤 2、点击接收风险并继续，搜索：

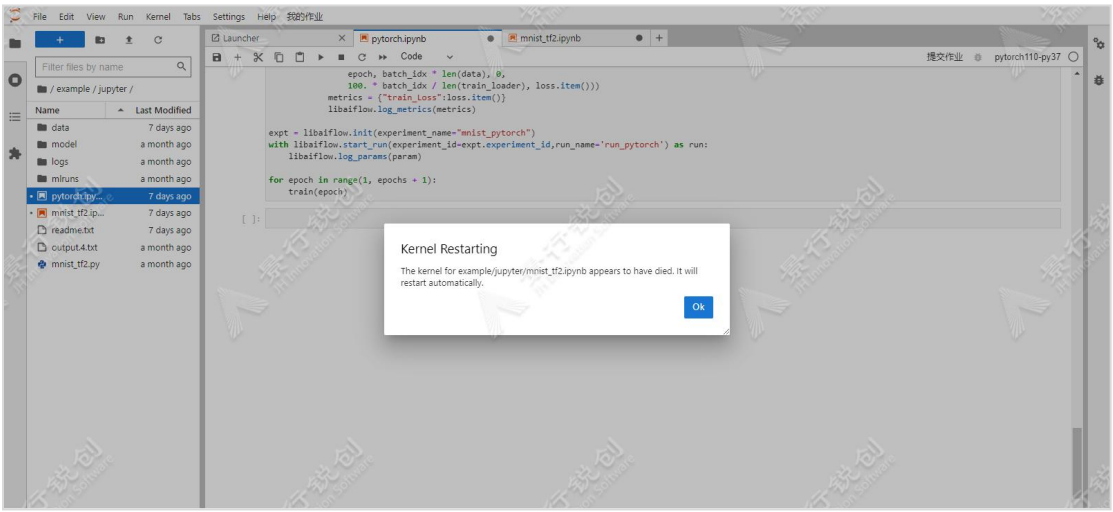
dom.events.testing.asyncClipboard、

dom.events.asyncClipboard.readText 将其值修改为 true；(注：如果 dom.events.asyncClipboard.readText 不存在可以不用修改)

步骤 3、重启浏览器即可成功。

18. 开发环境的内存耗尽后可能会引起下面一系列问题

1）Jupyter 中 kernel Restarting



原因：

可能是因为内存占满而导致 Kernel 重启。

解决方案：

可以尝试切换更高版本内存的硬件规格。

2）Jupyter 报错内存已满或者“CUDA error: out of memory”

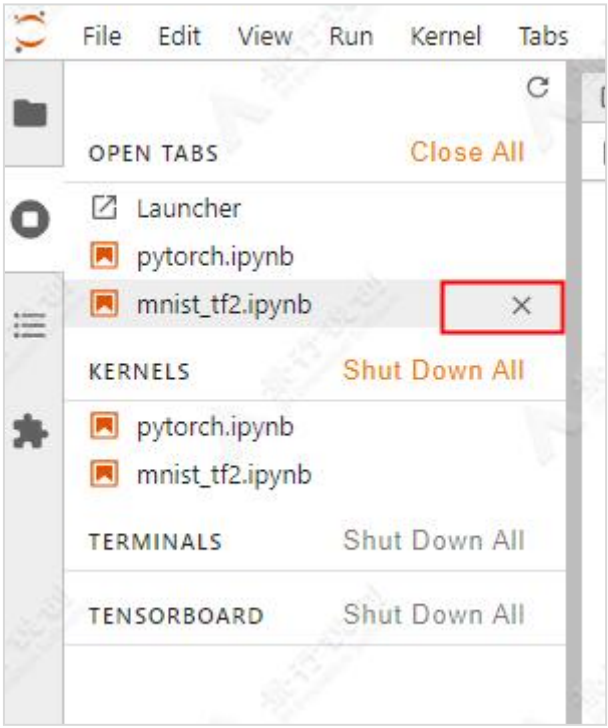
原因：

这类内存或显存满了的错误，原因是正在运行中的案例脚本太多。

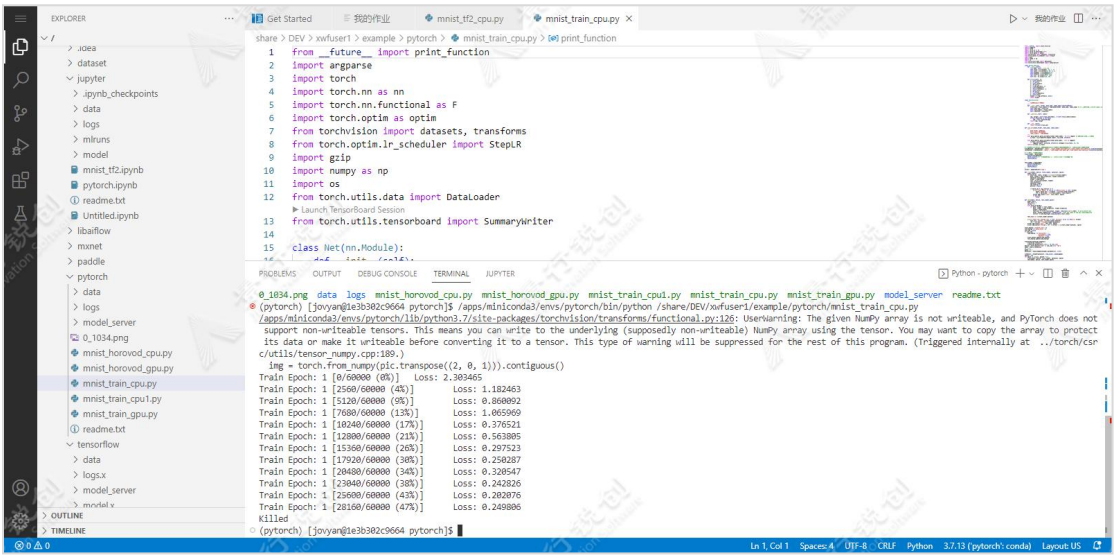
解决方案：

关掉一些正在运行的脚本即可释放内存或显存，操作方法如下：

进入到如下页面，检查后台正在运行的脚本，点击“×”关闭不需要运行的脚本。



3）问题：vscode 中运行的程序直接被 Killed。



原因：

可能是内存占满导致。

解决方法：

检查开发环境内存是否用完，释放内存，或者可以尝试切换更高版本内存的硬件规格，或调整程序减少内存使用。

4）问题：CentOs/Ubuntu 桌面开发环境连不上，提示错误：“会话登录超时！”。

原因：

可能是内存占满导致。

解决方法：

检查 CentOs/Ubuntu 桌面开发环境内存是否用完，可以等待一段时间，CentOs/Ubuntu 桌面中使用的内存释放后就会恢复正常，或者可以尝试切换更高版本内存的硬件规格。