一、概述

OPENI是一基于启智开源项目的集群管理工具和资源调度平台。平台提供了一系列接口，能够支持主流的深度学习框架，如CNTK, TensorFlow等。

OPENI分管理用户和普通用户，该手册主要供普通用户使用。用户首先创建任务，提交任务至平台执行。用户可通过任务列表查看已有任务，并通过ssh工具或tracking page查看和管理运行中的任务。

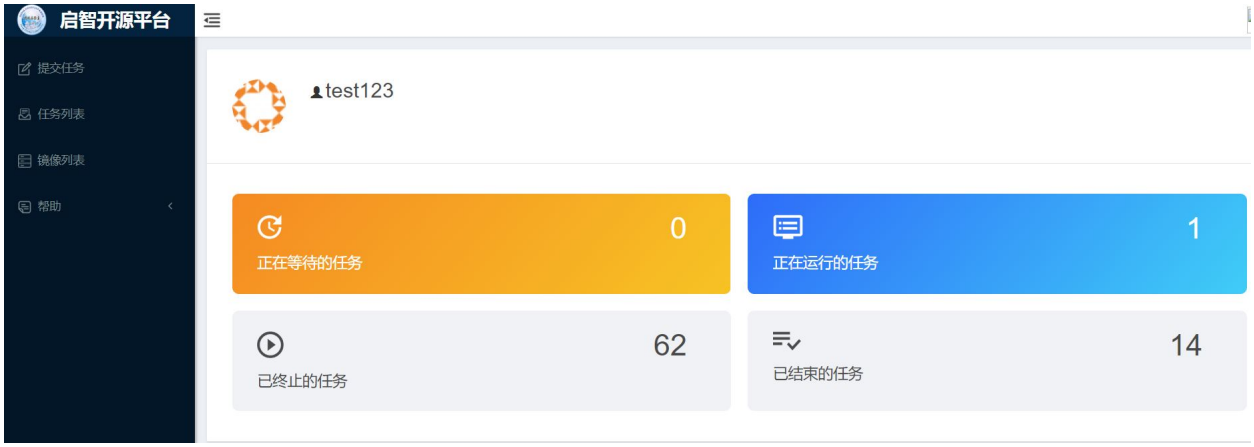
OPENI会帮助用户管理硬件资源，调度资源创建容器，运行用户任务。

二、使用说明

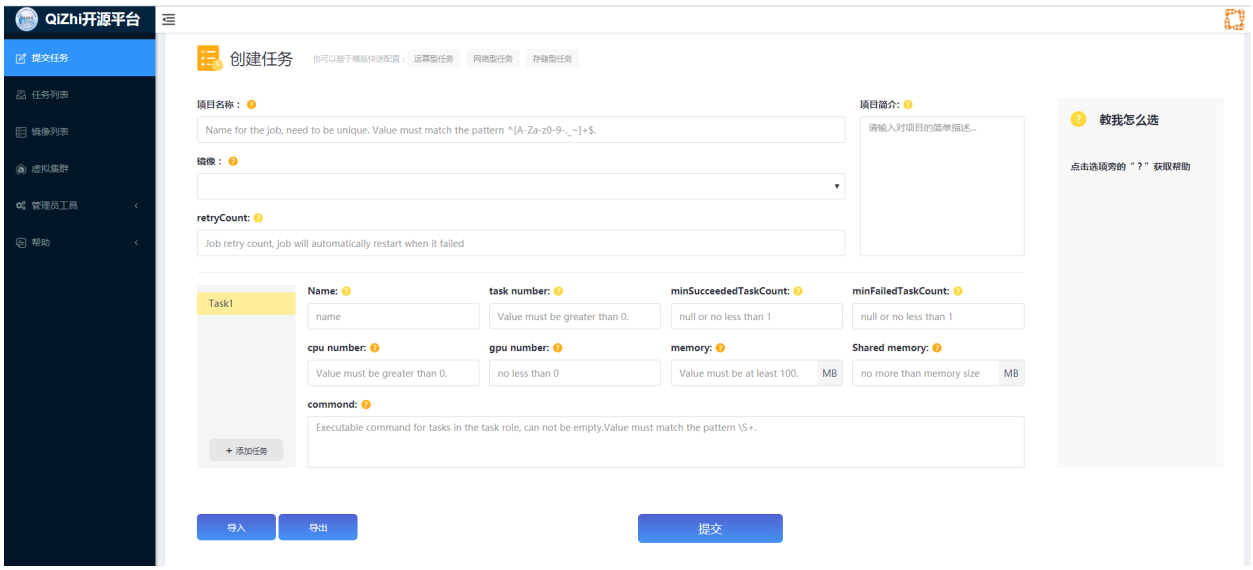
前提：假定用户已成功部署OPENI平台（部署文档请参考……），OPENI对外访问地址为http://192.168.113.221:9286，用户可使用浏览器访问该地址，推荐浏览器为chrome。用户可请求管理员注册账号（包括用户名及密码），如用户名=test123，密码=123456.

1. 浏览器中打开OPENI地址，进入login界面，输入用户名密码后点击登录：

2. 进入登录页后可看到用户当前任务面板，包括当前用户不同状态下任务的总数。左侧边栏包括提交任务、任务列表、镜像列表和帮助。



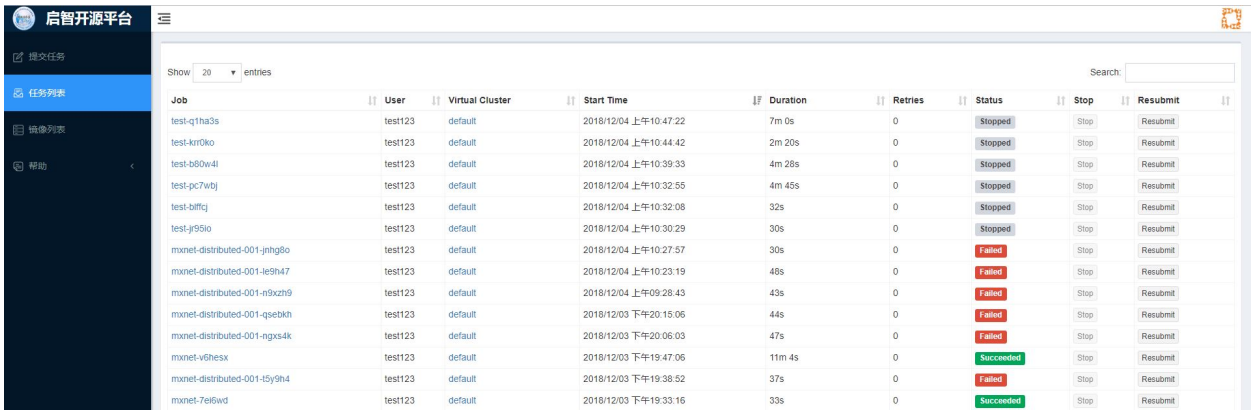
3. 提交任务，参数除了minSucceededTaskCount、minFailedTaskCount和retryCount，其余为必填项。具体用法参见第三节任务类型。



4. 参数填写完毕后，点击提交后，即完成第一个任务的提交。

后续步骤为以提交任务的查看和管理：

5. 可点击任务列表，查看当前用户的所有任务

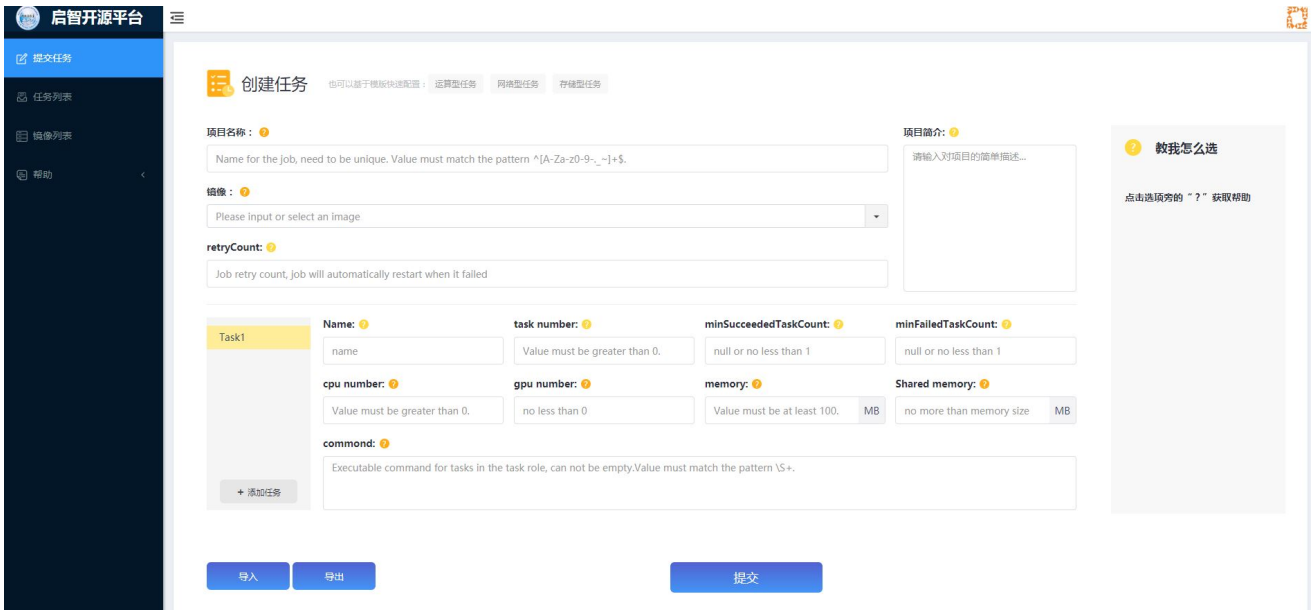


6. 可点击镜像列表，查看平台提供的各种镜像，用于创建用户的容器

三、任务类型：

I、提交单一任务

1）登录系统后，用户需要先提交任务，提交任务有两种方式：a、通过导入json文件提交，b、在页面填写



- 复制以下内容保存为`job.json`文件

```
{
"jobName": "JOBNAME",
"image": "192.168.113.221:5000/user-images/deepo:v2.0",
"taskRoles": [
  {
    "name": "demo",
    "taskNumber": 1,
    "cpuNumber": 4,
    "memoryMB": 8196,
    "gpuNumber": 1,
    "command": "sleep TIME(该值需要修改)"
  }
]
```

```
}
```

其中，`JOBNAME`用户自定义，提交时会自动增加后缀; `TIME`自定义，一旦提交，该容器申请的资源为用户独占。

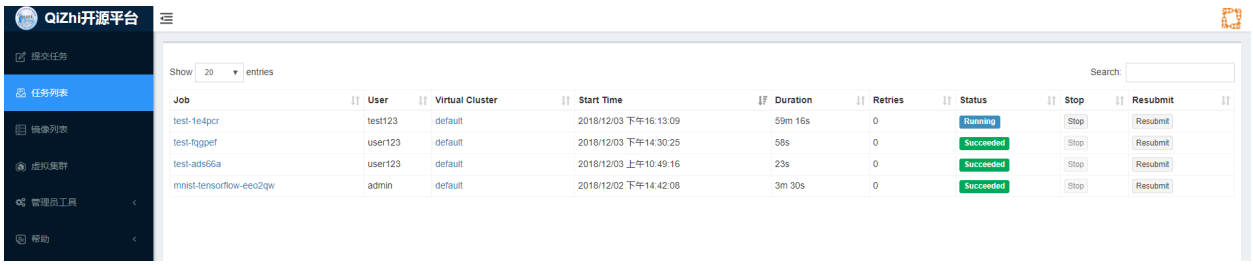
参数述说：

jobName	**任务的名称，全局唯一**
image	任务引用镜像
taskRole	当前任务中的角色，list列表，可添加多个
taskRole.minFailedTaskCount	最小失败任务数量，当taskRole中任务失败的个数大于等于该值时，判定该job为失败
taskRole.minSucceededTaskCount	最小成功任务数量，当taskRole中任务成功的个数大于等于该值时，判定该job为成功
taskRole.name	taskRole的名称
taskRole.taskNumber	当前任务角色中任务的个数（可理解为启动相同任务的容器个数）
taskRole.cpuNumber	当前任务的CPU个数（单个容器CPU个数）
taskRole.memoryMB	当前任务的内存（单个容器内存）
taskRole.gpuNumber	当前任务的GPU个数（单个容器GPU个数）
taskRole.command	当前任务需要运行的命令

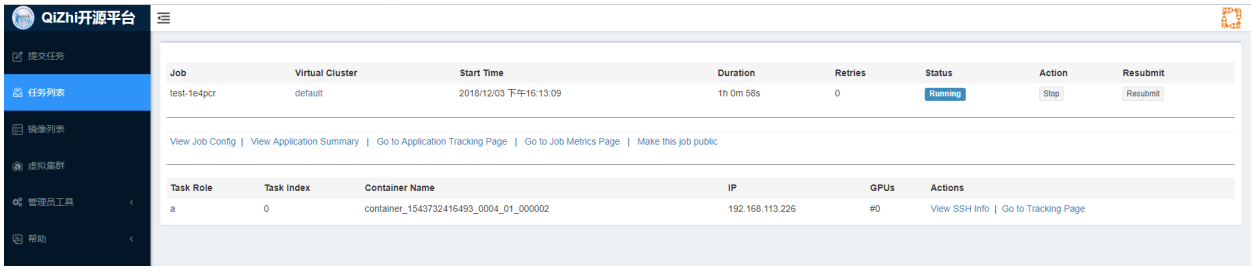
- 选择左侧边栏`提交任务`，点击`choose file（英文系统）`或`选择文件（中文系统）`选择保存的job.json文件后点击右下方`提交`即可，点击`提交`前，用户也可在界面修改提交的任务内容
- 若用户需要自定义job.json文件，可以参考下文`准备提交脚本`，**当前实现，`command`中命令执行完成后，容器会自动销毁**

四、任务查看

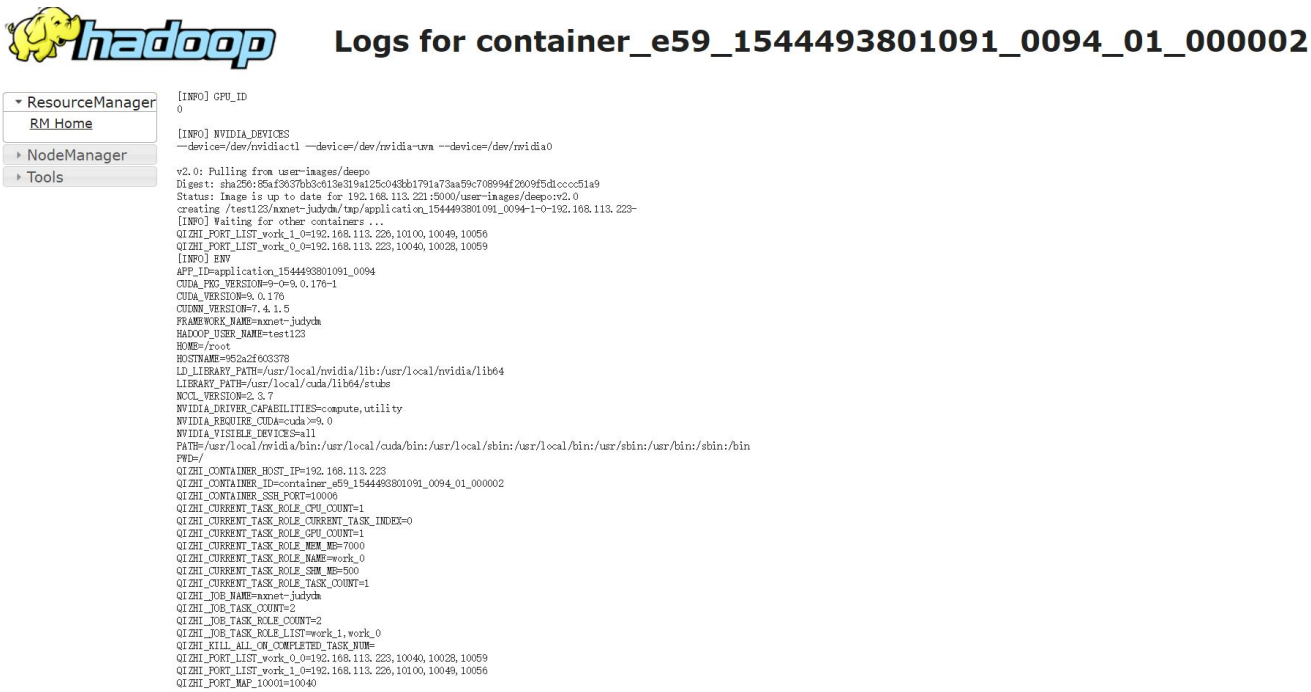
1、可在任务列表页查看当前运行的所有任务：



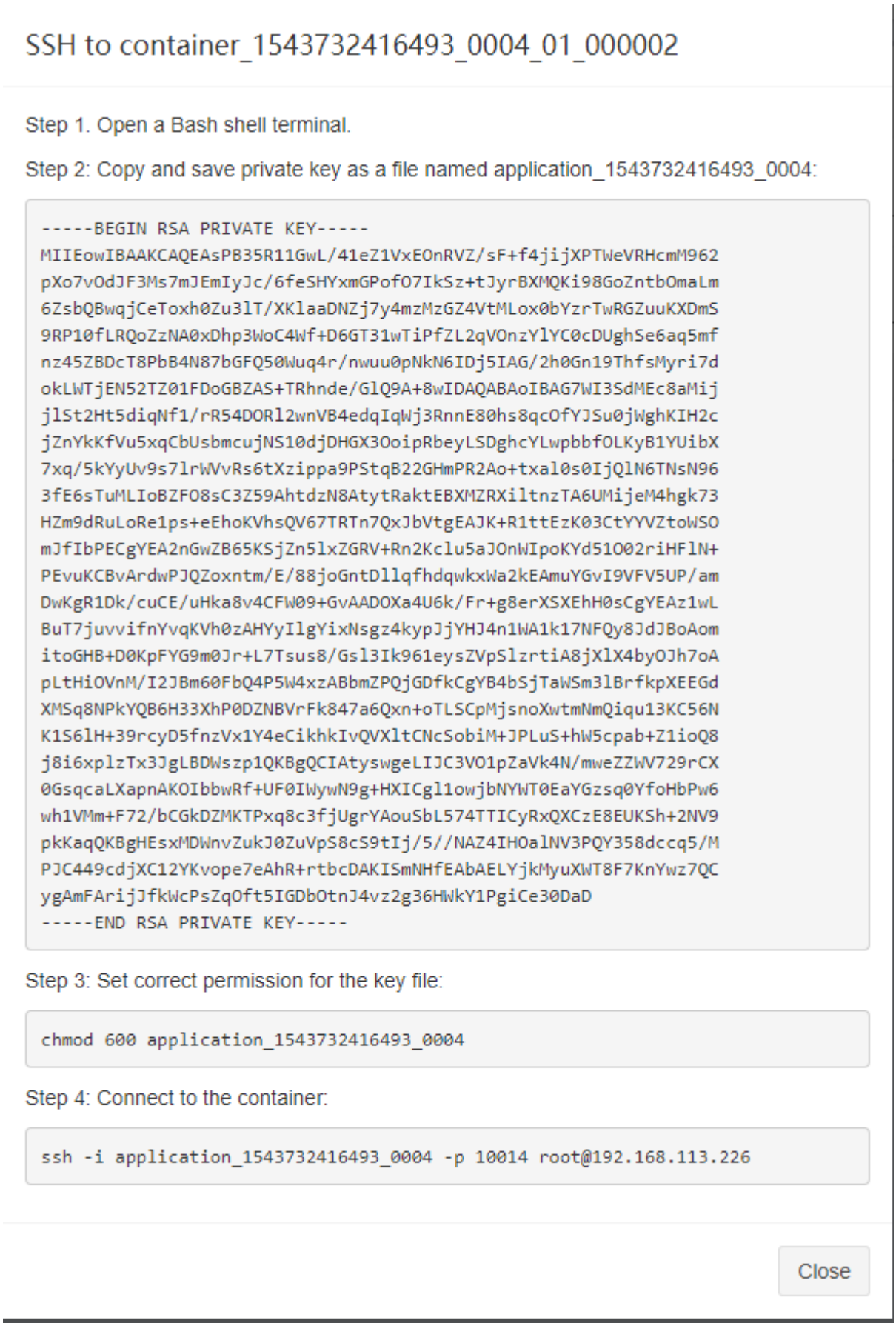
2、点击任务列表中状态为running的任务，可查看该任务详情：



3、点击Go to Tracking Page 可查看任务的输出的日志



4、点击View SSH Info，通过该方式可查看容器的地址和登录密钥，用于远程登录容器



- 使用ssh工具进行登录 linux/mac下按照sshinfo的步骤操作：

  第一步：复制上图step2中的密钥并保存在本地文件命名`application_1530350083555_0723`

第二步：修改文件权限：`chmod 600 application_1530350083555_0723`

第三步：连接容器：`ssh -i application_1530350083555_0723 -p 10014 root@192.168.113.226`

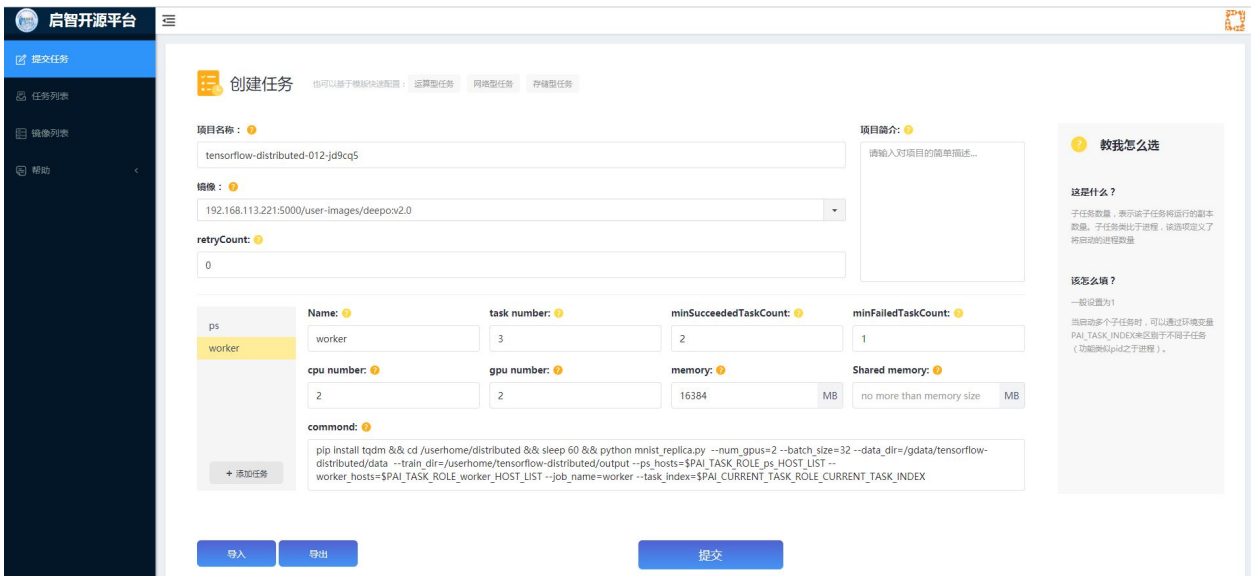
windows下：
第一步：打开cmd，创建密钥文件`application_1530350083555_0723`（同linux方法）
第二步：使用修改密钥文件权限：
`cacls application_1530350083555_0723 /G user:F` 其中，`application_1530350083555_0723`为文件名称，`/G` 指赋予用户指定权限，user为**当前操作系统用户名称**，F为全部权限.

第三步：登录`ssh -i application_1530350083555_0723 -p 10014 root@192.168.113.226`

- 使用ssh工具进行数据和源代码传输
  linux下使用SCP命令传输数据：
  `scp -i your_ssh_key -P your_ssh_port your_file ssh_username@ssh_ip:/userhome/` windows下使用winscp软件进行传输，参考方法
  注：每个用户都有唯一路径/userhome，用户可将个人资料、代码、数据放置在该路径下，云主机其他路径下内容会在重启时删除且无法恢复。

  II、提交多任务

  1）页面提交，



其中增加task后，即可在创建任务时生成多个容器，完成多容器任务

2）json文件提交

```
{
  "jobName": "tensorflow-distributed",
  "image": "192.168.113.221:5000/user-images/deepo:v2.0",

  "taskRoles": [
    {
      "name": "ps",
      "taskNumber": 2,
      "cpuNumber": 2,
      "memoryMB": 8192,
      "gpuNumber": 0,
      "minSucceededTaskCount": 2,
```

```
      "minFailedTaskCount": 1,
      "command": "python /gdata/tensorflow-distributed/code/mnist_replica.py  --
num_gpus=0 --batch_size=32 --data_dir=/gdata/tensorflow-distributed/data  --
train_dir=/userhome/tensorflow-distributed/output --
ps_hosts=$PAI_TASK_ROLE_ps_HOST_LIST --worker_hosts=$PAI_TASK_ROLE_worker_HOST_LIST
--job_name=ps --task_index=$PAI_CURRENT_TASK_ROLE_CURRENT_TASK_INDEX"
    },
    {
      "name": "worker",
      "taskNumber": 2,
      "cpuNumber": 2,
      "memoryMB": 16384,
      "gpuNumber": 2,
      "minSucceededTaskCount": 2
      "minFailedTaskCount": 1,
      "command": "python /gdata/tensorflow-distributed/code/mnist_replica.py  --
num_gpus=2 --batch_size=32 --data_dir=/gdata/tensorflow-distributed/data  --
train_dir=/userhome/tensorflow-distributed/output --
ps_hosts=$PAI_TASK_ROLE_ps_HOST_LIST --worker_hosts=$PAI_TASK_ROLE_worker_HOST_LIST
--job_name=worker --task_index=$PAI_CURRENT_TASK_ROLE_CURRENT_TASK_INDEX"
    }
  ],
  "retryCount": 0
}
```

具体可参见github

三、自定义镜像

平台提供的云主机不能满足个人需求时，可以根据个人需求创建自定义镜像：

用户可以参考dockerhub教程，打包镜像到dockerhub，提交任务时指明来自于dockerhub的path即可，比如：若使用deepo的镜像，可以填写`ufoym/deepo`。也可以将dockerfile上传到平台共有路径，并通知管理员build镜像并加入镜像列表即可 。

四、更多经典案例

更多案例可以参考github，其中包括入门级案例和进阶分布式案例等，也可参考项目。