



联想智能超算平台用户手册

V5.1.0



日期: 2018/05/03

版本号: v1.0



目录

1.	产品简介	4
1.1.	名词解释	4
1.2.	前提和假定	5
1.3.	运行环境	5
2.	使用说明	5
2.1.	登录	5
2.2.	登出	6
2.3.	修改密码	7
2.4.	查看集群资源及队列状态	7
2.5.	上传作业程序	8
2.6.	上传容器镜像	12
2.7.	提交作业	14
2.7.1.	提交 General 作业	14
2.7.2.	提交 Common 作业	16
2.8.	提交 HPC 作业	19
2.8.1.	提交 MPI 作业	19
2.8.2.	提交 ANSYS 作业	21
2.8.3.	提交 COMSOL 作业	24
2.9.	提交 AI 作业	27
2.9.1.	提交 TensorFlow 作业	27
2.9.2.	提交 Caffe 作业	31
2.9.3.	提交 Intel Caffe 作业	33
2.9.4.	提交 MXNet 作业	34
2.9.5.	提交 Neon 作业	36
2.9.6.	GPU 作业监控	38
2.10.	作业生命周期管理	40
2.10.1.	取消作业	40
2.10.2.	重新运行作业	41
2.10.3.	删除作业	42
2.11.	训练 AI 图像分类模型	42
2.11.1.	导入图像数据集	43

2.11.2. 创建网络拓扑	47
2.11.3. 训练模型	50
2.11.4. 参数调整	54
2.11.5. 测试及导出模型	55
2.11.6. 管理预训练模型	57
2.12. 自定义模板	58
2.12.1. 创建自定义模板	59
2.12.2. 发布自定义模板	66
2.13. 专家模式	66
2.13.1. 命令行提交作业	69
2.13.2. 作业文件编写	70
2.14. VNC 管理	70
3. 注意事项	71
3.1. 用户相关目录的绝对路径	71
3.2. 解决作业提交失败	71
3.3. VNC 查看或删除失败	72
3.4. SLURM 命令参考	72
3.5. Caffe 网络拓扑定义参考	72
3.6. GPU 监控数据来源	72

前言

欢迎使用联想智能超算平台（以下简称 LiCO），LiCO 致力于提供简单、易用、丰富的高性能计算及人工智能平台。本文档的读者需要具备一定的高性能计算、服务器集群的基础知识，同时对高性能计算中的并行开发、作业调度、人工智能（以下简称 AI）有一定的了解。

1. 产品简介

联想智能超算平台（Lenovo intelligent Computing Orchestration 以下简称 LiCO）是联想基于超性能计算（HPC）集群的一站式解决方案，其功能包括计算机集群管理，集群监控，作业调度管理，集群用户管理，账户管理，文件系统管理等。通过 LiCO 可以实现对超算（supercomputing）集群资源的统一调度，同时支持 HPC 作业和 AI 作业。随着人工智能、高性能计算和大数据的广泛应用，LiCO 已被越来越多的政府机关、大专院校、气象环保、石油石化、机械制造和生命科学研究等单位使用。

LiCO 基于 B/S 架构设计，用户可以方便的通过网页来对集群进行全面而细致的管控。LiCO 系统的主要功能有：

1. **集群的管理、监控：**提供机房直观的物理视图，集群中各节点 CPU、内存、硬盘、温度、系统负载以及网络使用状况等详细监控数据，并能对各节点进行逻辑分组，方便统一规划和管理；
2. **作业管理、监控：**直观的作业运行结果和状态管理，并支持各种主流调度器，支持丰富的作业类型（包括 AI 相关的作业如 Tensor Flow、Caffe 等）；
3. **用户管理、计费：**统一的界面支持本地和域用户的集成，支持用户充值、扣费，并设置计费组、费率等；
4. **警告、报警：**丰富的警报策略设置，并支持邮件、短信、微信等通知方式；
5. **报表：**多种报表类型，如集群报表，报警报表，作业报表，计费组作业报表等；
6. **定制化：**提供多样的定制化服务，如企业作业模板定制，报表定制，3D 机房定制化等服务

用户也可以通过其他 Shell 终端工具登陆到集群的登陆节点进行命令行操作。

1.1. 名词解释

计算机集群：对包含管理节点、登录节点、计算节点等在内的服务器资源的统称。

作业：完成特定任务的命令序列。

作业状态：指作业在调度系统中的状态标识，包括等待、排队、保留、运行、挂起、结束等。

节点状态：指节点的状态标识，包括空闲、已占用、繁忙、停机等。

作业调度系统：也称作业调度器或调度器，指负责接收、分发、执行、记账作业的分布式程序。

管理节点：指集群中运行作业调度，集群管理，用户计费等管理程序的服务器。

登录节点：指集群中用户可以通过 Linux 终端工具登录并进行操作的服务器。

计算节点：指集群中执行作业的服务器。

用户组： 用户集合，系统可针对用户组来定义对集群资源的访问控制策略，属于同一个用户组的所有用户可以访问相同的集群资源。

计费组： 也称计费账户，通过账户来对集群使用者进行计费，结算等操作。一个计费账户可由一个用户或多个用户共同使用。

1.2. 前提和假定

本文描述的内容主要针对基于作业调度器 Slurm 的情况，目前 LiCO 支持三种调度器：Slurm, Torque 以及 LSF。下文中 Slurm 相关命令不适用于 Torque 或 LSF 等调度器，如需使用，请参考对应文档。

1.3. 运行环境

集群服务器：

联想 Think System 服务器系列。

集群服务器支持的操作系统：

CentOS 6.8/7.3, Red Hat 6.8/7.3

客户端要求：

硬件：CPU 主频 2.0GHz 以上，内存大小 1GB 以上。

浏览器：推荐使用 Chrome 或 Firefox。

显示分辨率：不小于 1280 X 800

2. 使用说明

本文档主要介绍基于管理系统界面的基本操作方法和流程。对于命令行操作的方法可以参考 2.13.1 命令行提交作业 以及 3.4 SLURM 命令参考。

2.1. 登录

打开浏览器输入集群登录节点的 IP 地址，如 <https://10.220.112.21>（客户端必须可以直接访问集群登录节点）。可看到如下图所示的 LiCO 登录界面：

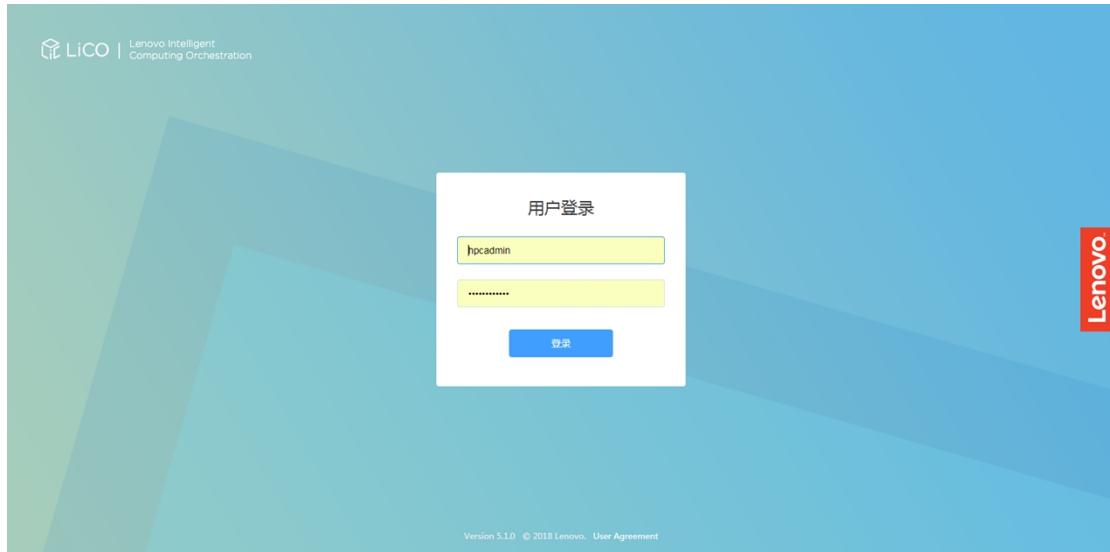


图 1 登录界面

输入用户名和密码（初始用户名和密码一般由管理员创建并告知）后点击登录，可登录 LiCO 系统并看到如下图所示的主界面：

图 2 用户主界面

2.2. 登出

LiCO 系统的用户登录会话令牌 (Token) 有自动延时功能，在正常使用系统的情况下并不会出现登录超时现象，所以当不使用系统时，请按如下步骤登出系统。

点击界面右上角的  按钮可弹出如下图所示的用户信息框：



图 3 用户信息对话框

点击框内右上角 按钮并在确认登出，即可登出 LiCO 系统。

2.3. 修改密码

用户在登录系统后，可以按如下步骤自行修改登录密码。

点击界面右上角的 按钮可弹出如图 3 所示的用户信息框，点击框内上方 按钮可弹出如下图所示的密码修改对话框：



图 4 修改密码对话框

输入当前密码及修改密码后点击提交，即可完成密码修改。

2.4. 查看集群资源及队列状态

点击位于主界面右侧菜单中的首页，可进入如下图所示的集群概况界面：

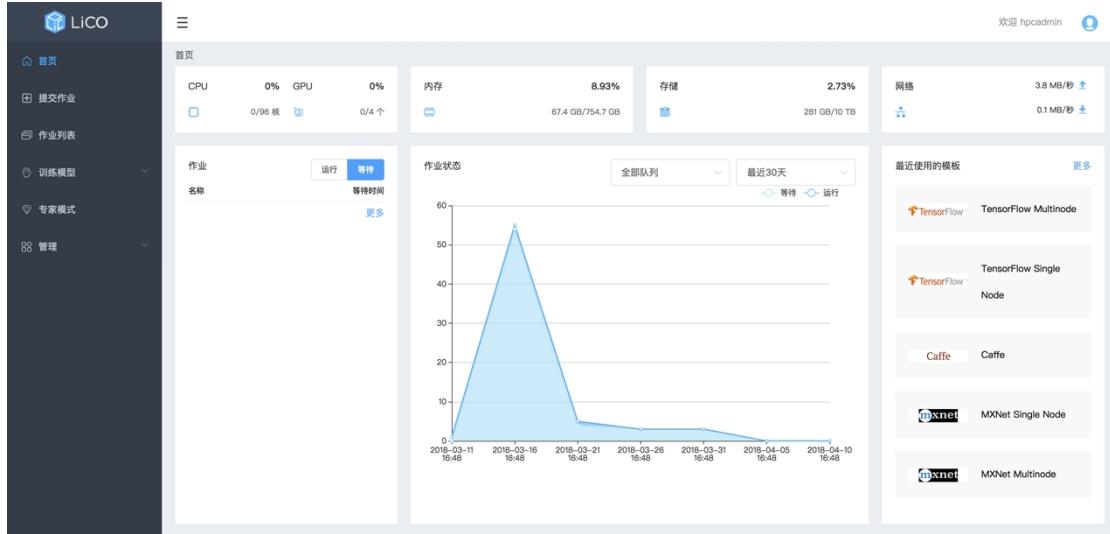


图 5 集群概况界面

CPU: 集群 CPU 的利用率，包括已经被占用的 CPU 核数和集群总共具有的 CPU 核数。

GPU: 集群 GPU 的利用率，包括已经被占用的 GPU 个数和集群总共具有的 GPU 个数。

内存: 集群内存的利用率，包括已经被占用的内存大小和集群总共具有的内存大小。

存储: 集群存储空间的利用率，包括已经被占用的存储空间和集群总存储空间。

网络: 集群网络的吞吐量，包括读速率和写速率。

作业: 当前用户已经提交运行和提交等待的作业信息。运行和等待状态下切换可以查看当前作业的名称和运行的时间。点击更多跳转到作业详情界面，可以查看具体的作业执行情况。

作业状态: 当前用户下所有提交历史作业的情况。可以根据队列和时间段进行查看，其中队列包含集群下所有队列的名称，时间段包含最近 1 小时，最近 24 小时，最近 7 天和最近 30 天。图表中可以选择运行和等待两种状态下的历史作业，并且可以查看具体某一个时间点的作业运行数量。

最近使用的模板: 展示当前用户最近使用较多的作业模板，可快速提交作业。

2.5. 上传作业程序

依次点击菜单：管理 -> 文件管理，进入如下图所示的文件管理界面：

The screenshot shows a file management interface with a sidebar on the left containing a tree view of folder contents under 'MyFolder'. The main area displays a detailed list of files and folders with columns for Name, Permissions, Modified, Size, and Kind. The list includes items like 'aaa', 'ai_datastore', 'caffe', 'ckTEST', 'containers', 'data', 'dch', 'demo', 'demo_back', 'intel_python', 'jobs', 'ls', and 'ls_tmp'. The total size of the folder is 4.20 GB.

Name	Permissions	Modified	Size	Kind
aaa	read and write	Apr 05, 2018 00:13 AM	146 b	Folder
ai_datastore	read and write	Mar 28, 2018 04:17 PM	94 b	Folder
caffe	read and write	Mar 30, 2018 01:41 PM	4 KB	Folder
ckTEST	read and write	Mar 18, 2018 05:00 PM	306 b	Folder
containers	read and write	Yesterday 10:44 AM	81 b	Folder
data	read and write	Apr 05, 2018 00:22 AM	4 kB	Folder
dch	read and write	Mar 30, 2018 02:03 PM	4 kB	Folder
demo	read and write	Today 10:34 AM	138 b	Folder
demo_back	read and write	Apr 04, 2018 05:08 PM	4 kB	Folder
intel_python	read and write	Mar 16, 2018 10:11 AM	4 kB	Folder
jobs	read and write	Mar 30, 2018 09:50 AM	151 b	Folder
ls	read and write	Today 09:44 AM	4 kB	Folder
ls_tmp	read and write	Apr 08, 2018 03:41 PM	10 b	Folder
nervana	read and write	Mar 14, 2018 03:54 PM	66 h	Folder

图 6 文件管理界面

每个用户会被分配到一个专有的文件夹“*My folder*”，在此文件夹下，用户拥有全部权限。此文件夹之外，用户没有任何访问及其他操作的权限。注意：“*My folder*”只是文件夹的显示名称，如果需要“*My folder*”在操作系统中的绝对路径，请与参考：3.1 用户相关目录的绝对路径。

在文件管理界面，用户可进行如下操作：

后退：返回上一部操作（有些操作不可逆）

前进：重复下一步操作（有些操作不可重复）

新建文件夹

删除文件夹

重命名文件夹

新建文件

上传文件

下载文件

删除文件

重命名文件

编辑文件

图标和列表示图切换

排列文件：按名称，按大小，按类型，按日期

搜索

刷新

注：用户可以通过工具栏或者右键快捷菜单使用以上功能。

以下是创建文件夹并上传文件的示例：

- 在空白处点击鼠标右键，选择创建文件夹：

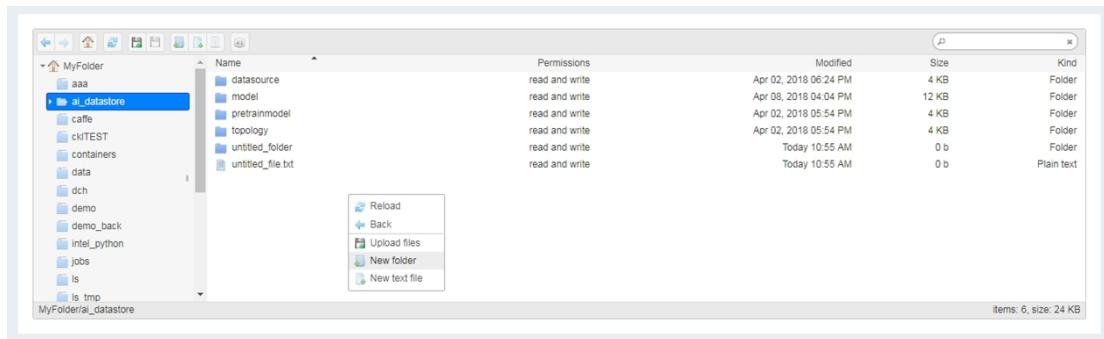


图 7 创建文件夹

2. 命名文件夹:

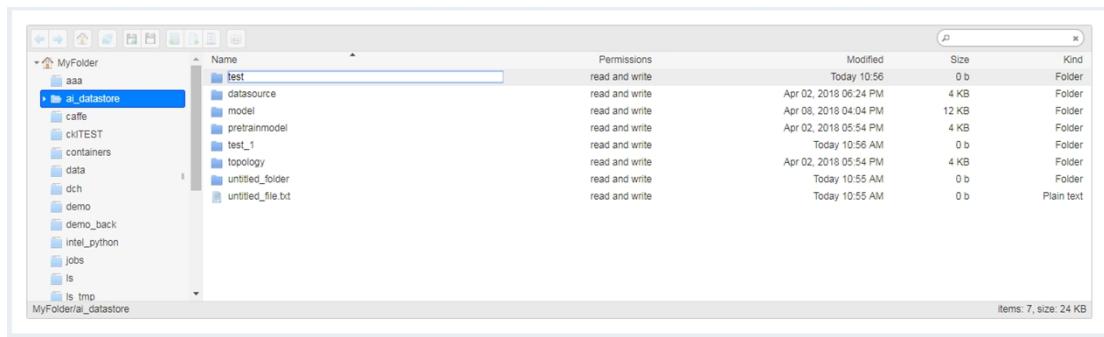


图 8 命名文件夹

3. 双击进入文件夹，在空白处点击鼠标右键，选择上传文件：

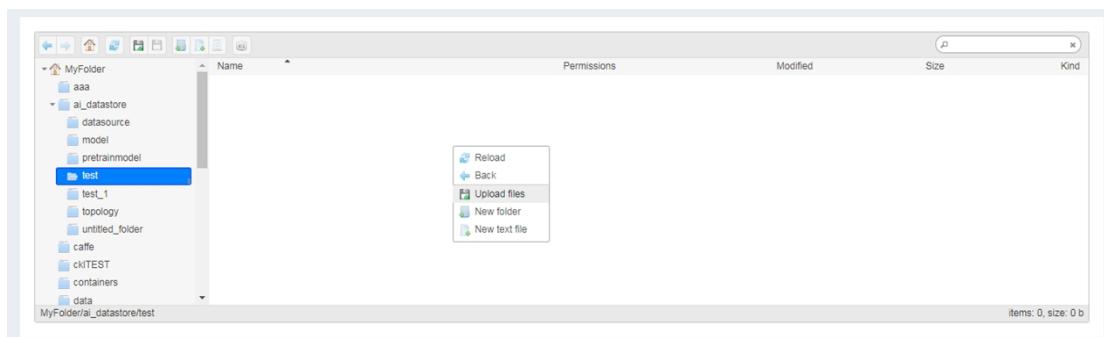


图 9 上传文件

4. 系统弹出文件上传对话框：



图 10 文件上传对话框

5. 可拖拽文件至对话框内虚线区域，或点击下方按钮来选择本地文件进行上传（可多选）：

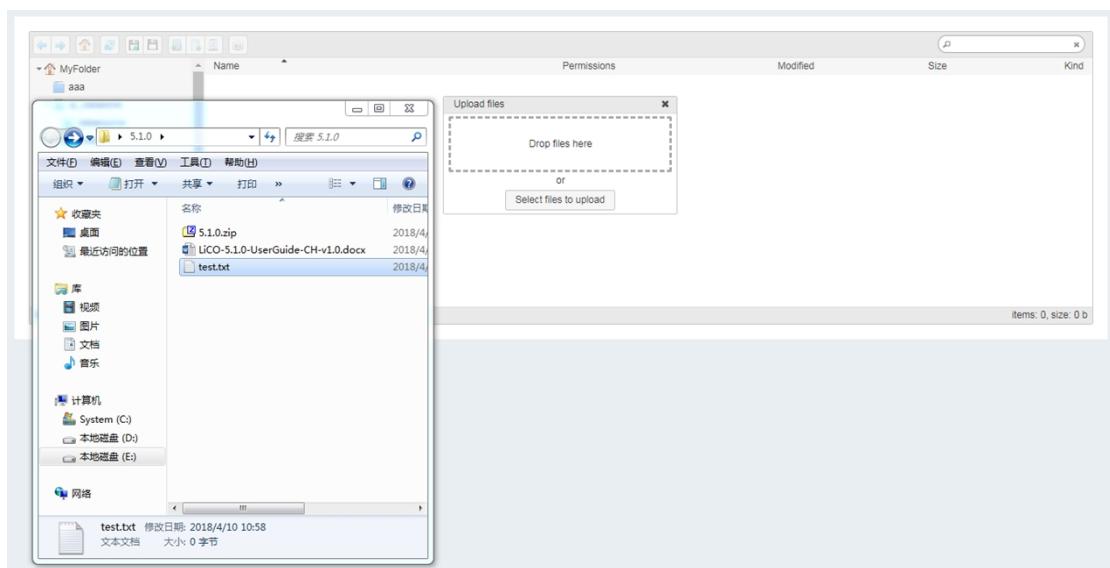


图 11 选择本地文件

6. 上传成功后可在文件管理界面查看到文件：



图 12 上传成功

2.6. 上传容器镜像

为了能够满足用户对运行 AI 作业的多样化、定制化的需求，LiCO 系统内 AI 的作业均在容器内运行，目前系统支持 Singularity 容器平台，不同的 AI 作业模板运行在不同的容器镜像上。

LiCO 系统在提供若干基础容器镜像之外，也允许上传用户自定义镜像。用户可按如下步骤进行操作来上传镜像：

1. 依次点击菜单：管理 -> 容器镜像，可进入如下图所示的容器镜像管理界面：

名称	框架	类型	所属用户	描述	状态	操作
1dd	Tensorflow	私有	hpcadmin		上传成功	
caffe-1.0-cpu	Caffe	系统	system		上传成功	
caffe-gpu	Caffe	系统	system		上传成功	
datasour	Intel-Caffe	私有	hpcadmin		上传成功	
intel-cafe-1.0-cpu	Intel-Caffe	系统	system		上传成功	
intel_python	Image.Framework.Intel-python	私有	hpcadmin		上传成功	
ls1	Intel-Caffe	私有	hpcadmin		上传成功	
mxnet-1.1-cpu	MXNet	系统	system		上传成功	
mxnet-1.1-gpu-cuda90	MXNet	系统	system		上传成功	
mxnet_ls	MXNet	私有	hpcadmin		上传成功	

图 13 容器镜像管理界面

2. 点击**创建**按钮，系统将弹出镜像上传对话框：



图 14 镜像上传对话框

3. 点击**浏览**按钮来选择镜像文件（镜像文件需要先上传至集群文件系统），完成信息填写后点击**提交**按钮；
4. 对话框将自动关闭，并返回镜像管理界面，可查看到镜像正在上传：

名称	框架	类型	所属用户	描述	状态	操作
mxnet_yang	MXNet	私有	hpcadmin	liangshuang image	上传成功	
neon-2.4-cpu	Neon	系统	system		上传成功	
ngraph	Tensorflow	私有	hpcadmin		上传成功	
python2	Image.Framework.intel-python	私有	hpcadmin		上传成功	
python2_Neon	Neon	私有	hpcadmin		上传成功	
tensorflow-1.6-cpu	Tensorflow	系统	system		上传成功	
tensorflow-1.6-gpu-cuda90	Tensorflow	系统	system		上传成功	
tensorflow_cuda9	Tensorflow	私有	hpcadmin		上传成功	
test	Tensorflow	私有	hpcadmin		上传中	

图 15 镜像上传中

5. 等待镜像上传完成后（状态变更为**上传成功**），则该镜像可被使用。

2.7. 提交作业

2.7.1. 提交 General 作业

General 作业是完全由用户自行控制运行方式及使用资源的计算任务，用户需要自行编写作业文件及启动脚本，如何编写请参考 2.13 专家模式：。

用户可以通过以下步骤来向 LiCO 系统提交并运行 General 作业：

- 在菜单中选择：提交作业，可进入如下图所示的作业模板选择界面：

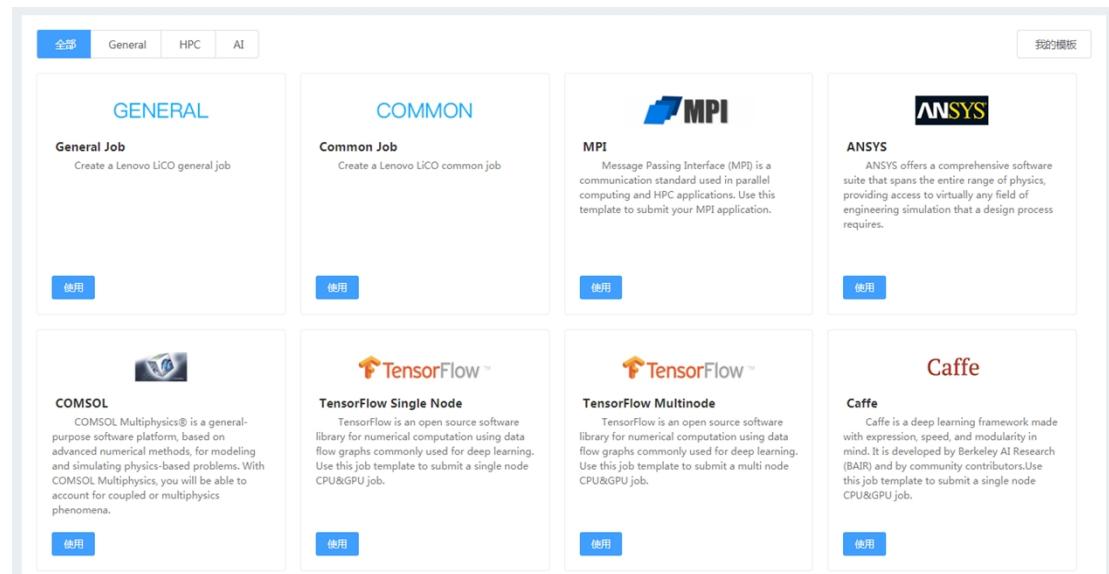


图 16 作业模板选择界面

- 在模板中选择 General Job 模板，可进入如下图所示的 General 作业提交界面：

This screenshot shows the 'General Job' submission form. It includes sections for '模板信息' (Template Information) where the user can enter the job name, and '模板参数' (Template Parameters) where they can select the job file. A large blue '提交' (Submit) button is located at the bottom left of the form area.

图 17 General 作业提交界面

- 填写作业名称，并点击浏览按钮来选择作业文件，作业文件及相关资源需要提前上传至集群文件系统；

4. 点击**提交**按钮来提交作业，提交成功可进入如下图的作业详情界面：

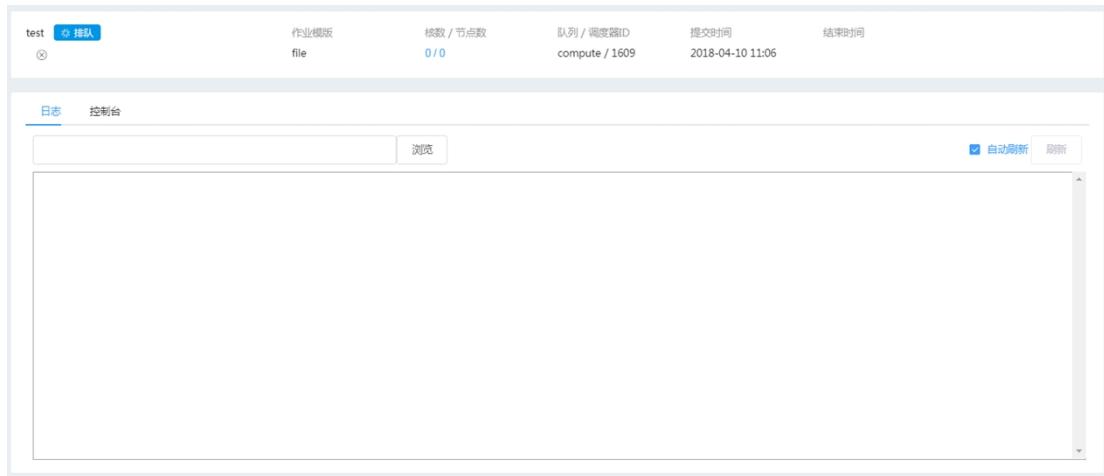


图 18 作业详情界面

5. 在作业详情界面，可以对某一个作业输出日志文件进行实时监控，也可以通过 Web 控制台登录集群登陆节点来对程序运行进行监控；
6. 用户可以通过点击菜单: **作业列表**, 进入如下图所示作业管理界面来监控作业运行情况：

The screenshot shows a list of completed jobs. The top navigation bar includes filters for '状态' (Status) set to '已完成' (Completed), '队列' (Queue) set to '全部' (All), and a search bar. The main table lists 1428 completed jobs, each with columns for ID, job name, scheduler ID, status, queue, submission time, completion time, and operations (view, download).

ID	作业名称	调度器ID	状态	队列	提交时间	完成时间	操作
1860	test	1609	完成	compute	2018-04-10 11:06	2018-04-10 11:06	<input type="button"/> <input type="button"/> <input type="button"/>
1836	dfd	1586	完成	compute	2018-04-09 15:00	2018-04-09 15:00	<input type="button"/> <input type="button"/> <input type="button"/>
1832	dfd	1584	完成	compute	2018-04-09 10:24	2018-04-09 10:26	<input type="button"/> <input type="button"/> <input type="button"/>
1831	dfd	1583	完成	compute	2018-04-09 10:12	2018-04-09 10:14	<input type="button"/> <input type="button"/> <input type="button"/>
1830	dfd	1582	完成	compute	2018-04-09 10:01	2018-04-09 10:03	<input type="button"/> <input type="button"/> <input type="button"/>
1829	dff	1581	完成	compute	2018-04-09 10:00	2018-04-09 10:00	<input type="button"/> <input type="button"/> <input type="button"/>
1828	dfd	1580	完成	compute	2018-04-09 09:54	2018-04-09 09:55	<input type="button"/> <input type="button"/> <input type="button"/>
1826	222	1579	取消	compute	2018-04-09 09:33	2018-04-09 09:55	<input type="button"/> <input type="button"/> <input type="button"/>
1822	dfd	1578	取消	compute	2018-04-09 09:32	2018-04-09 09:32	<input type="button"/> <input type="button"/> <input type="button"/>
1821	dff	1577	完成	compute	2018-04-09 09:28	2018-04-09 09:31	<input type="button"/> <input type="button"/> <input type="button"/>

Total 1428 10/page < 1 2 3 4 5 6 ... 143 > Go to 1

图 19 作业管理界面

7. 等待作业运行结束后，可以点击作业管理界面中作业所在行或作业详情页面的 进入作业的工作目录，如下图所示可以对作业的输出文件进行下载：

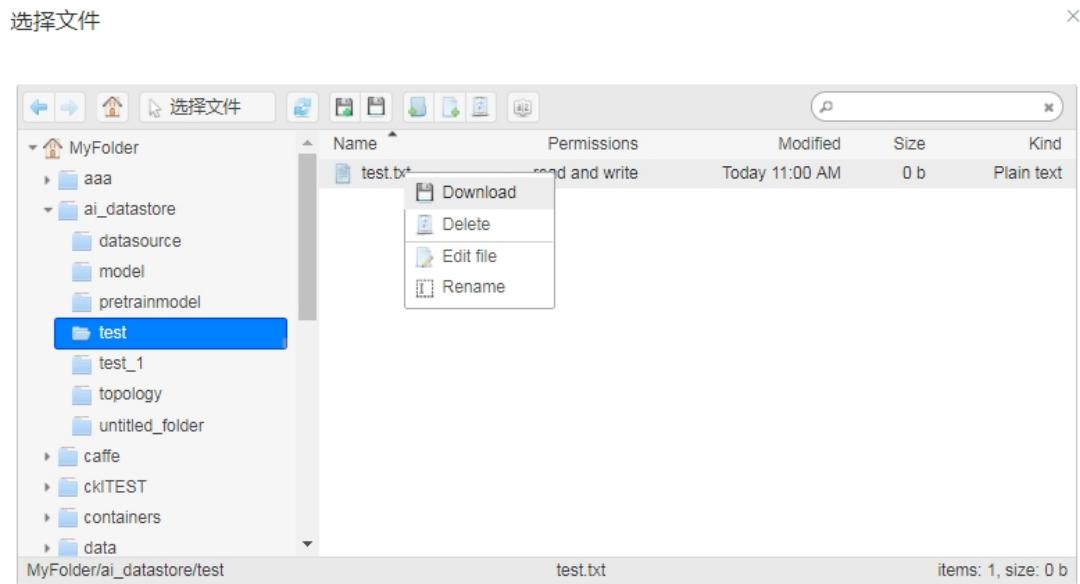


图 20 作业输出下载

2.7.2. 提交 Common 作业

Common 作业是指用户在 Web 界面上快速编写并运行的脚本类作业。

用户可以通过以下步骤来向 LiCO 系统提交并运行 Common 作业：

1. 进入作业模板选择界面；
2. 在模板中选择 **Common Job** 模板，可进入如下图所示的 Common 作业提交界面：

COMMON

Common Job
Create a Lenovo LiCO common job

模板信息

* 作业名称

模板参数

* 工作目录 浏览

* 运行脚本

资源选项

* 队列

节点数

每个节点的核数 核

内存使用 MB

预计运行时间 小时

图 21 Common 作业提交界面

3. 填写作业名称，工作目录，并在运行脚本框内填写需要运行的脚本（支持 Linux Shell 命令）；
4. 用户可进行运行所需要资源的选择：
队列（必选）：作业所属的队列名，这里可选择的队列是用户有权限使用的队列；
节点数（可选）：应用程序运行所需要的计算节点数目；
每个节点的 CPU 核数（可选）：应用程序运行所需要每个计算节点的 CPU 核数；
内存容量（可选）：应用程序运行所需要内存大小；
预计运行时间（可选）：应用程序运行所需要时间；
5. 用户可进行运行提醒的配置：
运行状态通知（可选）：应用程序运行状态的通知：作业完成时通知和作业被挂起时通知；
邮箱（可选）：用于接收通知的邮箱；
6. 下图是一个 Common 作业填写的示例：

COMMON

Common Job
Create a Lenovo LICO common job

模板信息

* 作业名称

模板参数

* 工作目录 浏览

* 运行脚本

资源选项

* 队列

节点数

每个节点的核数 核

内存使用 MB

预计运行时间 小时

提交

The screenshot shows the 'Common Job' creation interface. In the 'Template Information' section, the job name is set to 'test_COMMON'. Under 'Template Parameters', the working directory is specified as 'MyFolder/ai_datastore/test'. The 'Run Script' field contains the value 'SCRIPT'. In the 'Resource Options' section, the queue is selected as 'compute'. The 'Submit' button at the bottom is highlighted in blue.

图 22 Common 作业填写示例

7. 填写完成后，点击**提交**按钮即可提交运行。

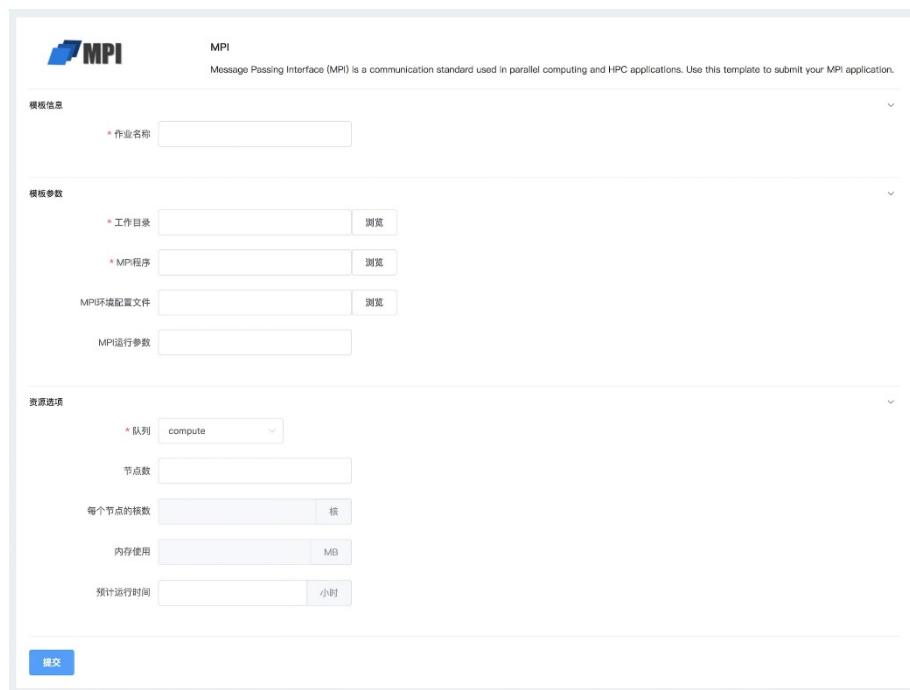
2.8. 提交 HPC 作业

2.8.1. 提交 MPI 作业

MPI 作业是指基于 MPI 运行的分布式计算作业，LiCO 系统内置 MPI，可运行符合 MPI 规范编写的分布式计算程序。

用户可以通过以下步骤在 LiCO 系统中提交并运行 MPI 作业：

1. 进入作业模板选择界面；
2. 在模板中选择 MPI 模板，可进入如下图所示的 MPI 作业提交界面：



The screenshot shows the 'MPI' submission form. At the top, it says 'Message Passing Interface (MPI) is a communication standard used in parallel computing and HPC applications. Use this template to submit your MPI application.' Below this are three main sections:

- 模板信息 (Template Information):** Contains a field labeled '作业名称' (Job Name) with a placeholder '作业名称'.
- 模板参数 (Template Parameters):** Contains fields for '工作目录' (Working Directory), 'MPI程序' (MPI Program), 'MPI环境配置文件' (MPI Environment Configuration File), and 'MPI运行参数' (MPI Run Parameters). Each field has a '浏览' (Browse) button.
- 资源选项 (Resource Options):** Contains fields for '队列' (Queue) set to 'compute', '节点数' (Number of Nodes), '每个节点的任务数' (Tasks per Node), '内存使用' (Memory Usage) in MB, and '预计运行时间' (Estimated Runtime) in hours. There is also a '节点数' (Number of Nodes) dropdown.

At the bottom is a blue '提交' (Submit) button.

图 23 MPI 作业提交界面

3. 填写作业名称及 MPI 作业相关运行参数：

工作目录（必选）：作业的工作目录；

MPI 程序（必选）：MPI 程序，也是最终要运行计算任务的程序。为了让这个应用程序能够在多台服务器上并行计算，这个程序要符合 MPI 标准；

MPI 环境文件（可选）：MPI 的环境变量；

MPI 运行参数（可选）：mpirun 的运行参数；

4. 填写资源选项及提醒选项；

5. 下图是一个 MPI 作业填写的示例：

The screenshot shows a web-based MPI job submission interface. At the top, there is a logo and a brief description: "MPI Message Passing Interface (MPI) is a communication standard used in parallel computing and HPC applications. Use this template to submit your MPI application." Below this, the form is divided into several sections:

- 模板信息**: Contains a field for the job name, which is currently set to "test_MPI".
- 模板参数**: Includes fields for the work directory ("MyFolder/demo/mpi") and MPI program ("MyFolder/demo/mpi/slurm-B03.out"). There are also fields for MPI environment configuration files and MPI run parameters.
- 资源选项**: Allows specifying the queue ("compute"), number of nodes, number of cores per node, memory usage (in MB), and estimated run time (in hours).
- 提交**: A blue button at the bottom left for submitting the job.

图 24 MPI 作业填写示例

6. 填写完成后，点击**提交**按钮即可提交运行。

2.8.2. 提交 ANSYS 作业

ANSYS 作业是指基于 ANSYS 运行的分布式计算作业, LiCO 系统内置 ANSYS, 可运行符合 ANSYS 规范编写的分布式计算程序(包括 GUI 方式)。

用户可以通过以下步骤来向 LiCO 系统提交并运行 ANSYS 作业:

1. 进入作业模板选择界面;
2. 在模板中选择 ANSYS 模板, 可进入如下图所示的 ANSYS 作业提交界面:

The screenshot shows the ANSYS job submission interface. It includes sections for job information, template parameters, and resource options, with various input fields and dropdown menus.

图 25 ANSYS 作业提交界面

3. 填写作业名称及 ANSYS 作业相关运行参数:

运行模式: 勾选图形界面表示 ANSYS 以 GUI 模式运行, 提交作业后, 会打开一个 VNC session, 并在 VNC session 里面将 anasys 图形界面运行起来。可通过 VNC 管理界面进行访问;

工作目录 (必选): 程序的工作目录;

ANSYS 可执行文件 (必选): ANSYS 软件的位置;

ANSYS 环境配置文件：ANSYS 运行时需要的环境变量；

ANSYS 运行参数（可选）：ANSYS 运行的参数；

ANSYS 输入文件（必选）：应用程序运行时输入的文件；

ANSYS 输出文件（可选）：应用程序运行时输出的文件；

4. 填写资源选项及提醒选项；

5. 下图是一个 ANSYS 作业填写的示例：

The screenshot shows the ANSYS job submission interface. At the top, there is a banner with the ANSYS logo and a brief description: "ANSYS offers a comprehensive software suite that spans the entire range of physics, providing access to virtually any field of engineering simulation that a design process requires." Below the banner, there are three main sections: "模板信息" (Template Information), "模板参数" (Template Parameters), and "资源选项" (Resource Options).
模板信息: Contains a field for "作业名称" (Job Name) with the value "test_ANSYS".
模板参数: Contains fields for "运行模式" (Run Mode) set to "图形界面" (Graphical Interface), and several input file selection fields:

- * 工作目录 (Working Directory): MyFolder/demo_back/abc
- * ANSYS可执行文件 (ANSYS Executable File): MyFolder/slurm-1176.out
- ANSYS环境配置文件 (ANSYS Environment Configuration File): (empty)
- ANSYS运行参数 (ANSYS Run Parameters): (empty)
- * ANSYS输入文件 (ANSYS Input File): MyFolder/test/slurm-293.out
- ANSYS输出文件 (ANSYS Output File): (empty)

资源选项: Contains fields for resource allocation:

- * 队列 (Queue): compute
- 节点数 (Number of Nodes): (empty)
- 每个节点的核数 (Number of Cores per Node): (empty)
- 内存使用 (Memory Usage): (empty) MB
- 预计运行时间 (Expected Run Time): (empty) 小时

At the bottom left is a blue "提交" (Submit) button.

图 26 ANSYS 作业填写示例

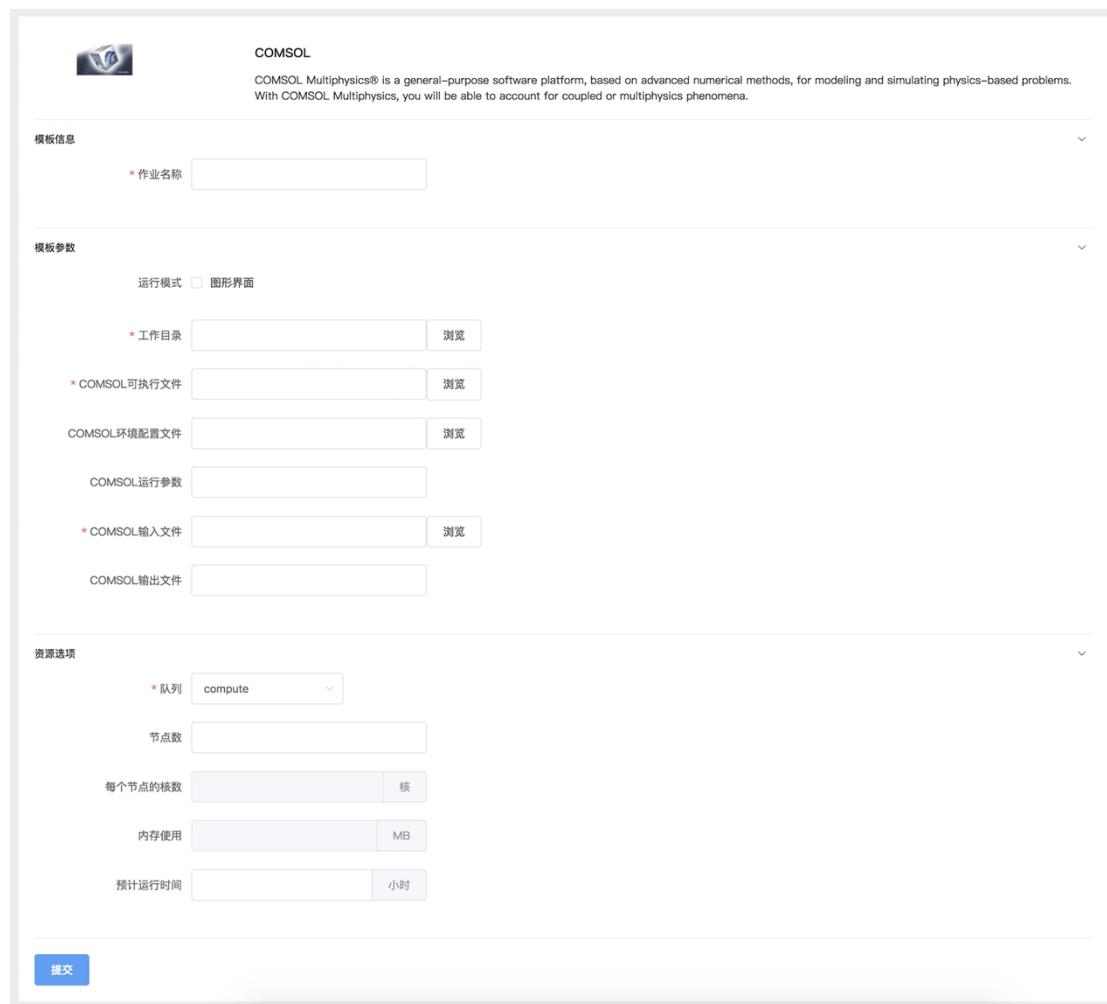
6. 填写完成后，点击**提交**按钮即可提交运行。

2.8.3. 提交 COMSOL 作业

COMSOL 作业是指基于 COMSOL 运行的分布式计算作业，LiCO 系统内置 COMSOL，可运行符合 COMSOL 规范编写的分布式计算程序（包括 GUI 方式）。

用户可以通过以下步骤来向 LiCO 系统提交并运行 ANSYS 作业：

1. 进入作业模板选择界面；
2. 在模板中选择 COMSOL 模板，可进入如下图所示的 COMSOL 作业提交界面：



The screenshot shows the COMSOL job submission interface. It consists of several sections:

- 模板信息**: Contains a required field for the **作业名称**.
- 模板参数**: Contains fields for:
 - 运行模式**: A checkbox labeled "图形界面".
 - 工作目录**: A required field with a browse button.
 - COMSOL可执行文件**: A required field with a browse button.
 - COMSOL环境配置文件**: An optional field with a browse button.
 - COMSOL运行参数**: An optional field.
 - COMSOL输入文件**: A required field with a browse button.
 - COMSOL输出文件**: An optional field.
- 资源选项**: Contains fields for:
 - 队列**: A dropdown menu set to "compute".
 - 节点数**: An optional field.
 - 每个节点的核数**: An optional field.
 - 内存使用**: An optional field.
 - 预计运行时间**: An optional field.

At the bottom is a blue **提交** (Submit) button.

图 27 COMSOL 作业提交界面

3. 填写作业名称及 COMSOL 作业相关运行参数：

运行模式：勾选**图形界面**表示 COMSOL 以 GUI 模式运行，提交作业后，会打开一个 VNC session，并在 VNC session 里面将 COMSOL 图形界面运行起来。可通过 VNC 管理界面进行访问；

工作目录（必选）：程序的工作目录；

COMSOL 可执行文件（必选）：COMSOL 软件的位置；

COMSOL 环境配置文件：COMSOL 运行时需要的环境变量；

COMSOL 运行参数（可选）：COMSOL 运行的参数；

COMSOL 输入文件（必选）：应用程序运行时输入的文件；

COMSOL 输出文件（可选）：应用程序运行时输出的文件；

4. 填写资源选项及提醒选项；

5. 下图是一个 COMSOL 作业填写的示例：

The screenshot shows the COMSOL job submission interface. At the top, it displays the COMSOL logo and a brief description: "COMSOL Multiphysics® is a general-purpose software platform, based on advanced numerical methods, for modeling and simulating physics-based problems. With COMSOL Multiphysics, you will be able to account for coupled or multiphysics phenomena."

模板信息

* 作业名称: test_COMSOL

模板参数

运行模式: 图形界面

* 工作目录: MyFolder/ai_datastore/topology

* COMSOL可执行文件: MyFolder/untitled_file_3.txt

COMSOL环境配置文件:

COMSOL运行参数:

* COMSOL输入文件:

COMSOL输出文件:

资源选项

* 队列: compute

节点数:

每个节点的核数: 核

内存使用: MB

预计运行时间: 小时

图 28 COMSOL 作业填写示例

6. 填写完成后，点击**提交**按钮即可提交运行。

2.9. 提交 AI 作业

2.9.1. 提交 TensorFlow 作业

TensorFlow 作业是指基于 TensorFlow 运行的单机或者分布式计算作业，LiCO 系统通过 Singularity 容器镜像来运行基于 TensorFlow 框架编写的 AI 程序，可支持 CPU 或 GPU。

用户可以通过以下步骤在 LiCO 系统中提交并运行 TensorFlow 分布式作业：

1. 进入作业模板选择界面；
2. 在模板中选择 TensorFlow Multinode 模板，可进入如下图所示的 TensorFlow 作业提交界面：

TensorFlow™

TensorFlow Multinode

TensorFlow is an open source software library for numerical computation using data flow graphs commonly used for deep learning. Use this job template to submit a multi node CPU&GPU job.

模板信息

* 作业名称

模板参数

* 工作目录 浏览

* 容器镜像 tensorflow_gpu_4

* TF程序(.py) 浏览

* 节点数 1

* PS任务名 ps

* Worker任务名 worker

* 每个PS任务的CPU核数 1

* 每个Worker任务的CPU核数 1

每个Worker任务的GPU数

* job_name参数名 job_name

* task_index参数名 task_index

* ps_hosts参数名 ps_hosts

* worker_hosts参数名 worker_hosts

运行参数

资源选项

* 队列 compute

提交

图 29 TensorFlow 作业提交界面

3. 填写作业名称及 TensorFlow 作业相关运行参数：

工作目录（必填）： 作业的工作目录；

容器镜像（必填）： 作业所使用的 Singularity 镜像；

TF 程序（必填）： TesorFlow 程序，也是最终要运行计算任务的程序。这个程序要符合 TensorFlow 标准，该程序必须是 .py 文件；

节点数（必填）： 作业运行所需要的计算节点数目；

PS 任务名（必填）： PS 任务名；

Worker 任务名（必填）： Worker 任务名；

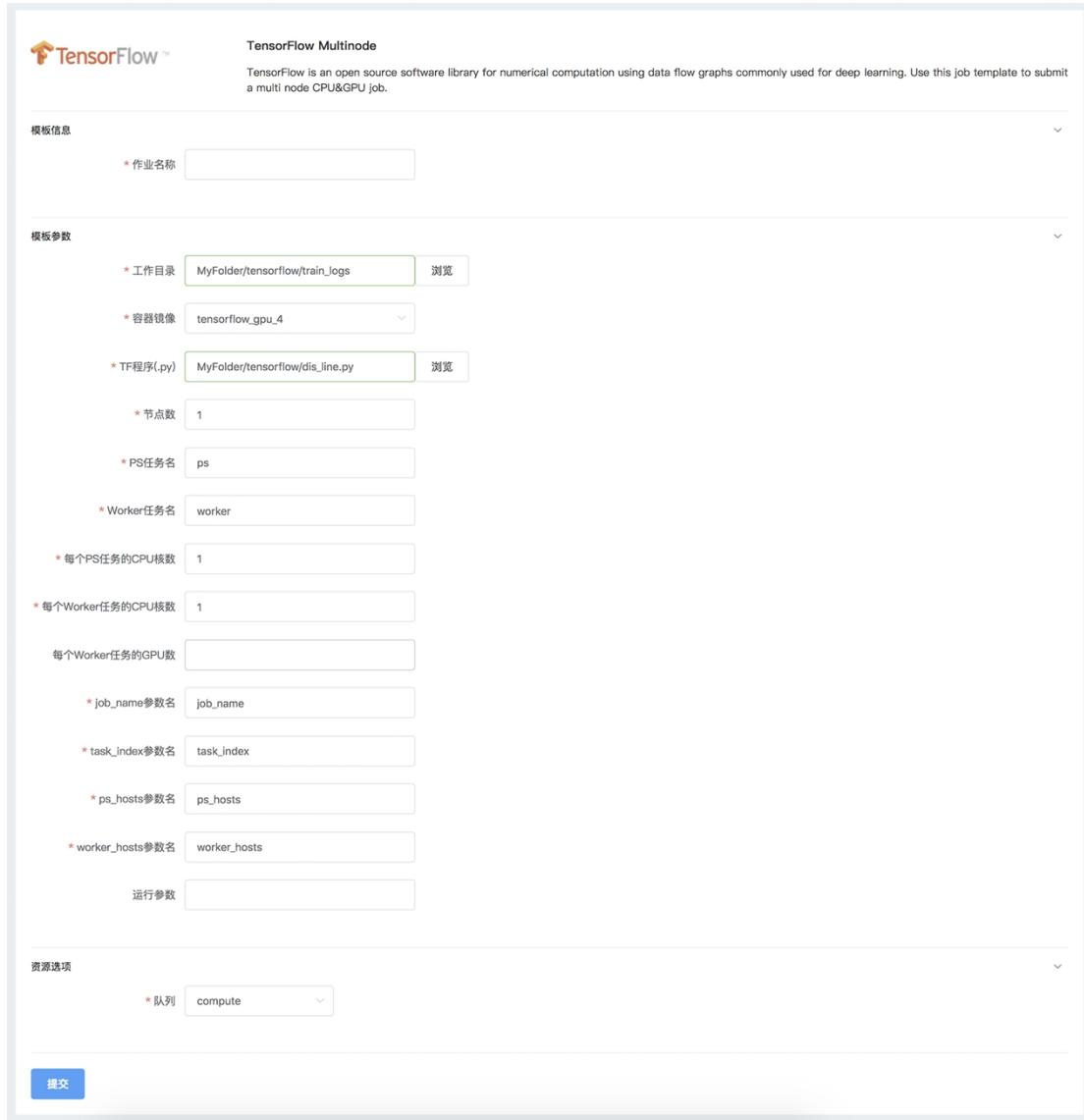
每个 PS 任务的 CPU 核数（必填）： 每个 PS 任务分配的 CPU 核数；

每个 Worker 任务的 CPU 核数（必填）： 每个 Worker 任务分配的 CPU 核数；

每个 Worker 任务的 GPU 数（选填）： 每个 Worker 任务分配的 GPU 个数；

job_name 参数名（必填）： job_name 的参数名；
task_index 参数名（必填）： task_index 的参数名；
ps_hosts 参数名（必填）： ps_hosts 的参数名；
worker_hosts 参数名（必填）： worker_hosts 的参数名；
运行参数（选填）： 运行所需的其他参数；

4. 填写资源选项；
5. 下图是一个 TensorFlow 作业填写的示例：



The screenshot shows the 'TensorFlow Multinode' job template configuration interface. It includes sections for 'Template Information', 'Template Parameters', and 'Resource Options'. The 'Template Parameters' section contains fields for 'Job Name', 'Work Directory', 'Container Image', 'TF Program (.py)', 'Number of Nodes', 'PS Task Name', 'Worker Task Name', 'CPUs per PS Task', 'CPUs per Worker Task', 'GPUs per Worker Task', 'job_name parameter name', 'task_index parameter name', 'ps_hosts parameter name', 'worker_hosts parameter name', and 'Run Parameters'. The 'Resource Options' section contains a 'Queue' field set to 'compute'. A 'Submit' button is located at the bottom left.

图 30 TensorFlow 作业填写示例

6. 填写完成后，点击提交按钮即可提交运行。

7. LiCO 针对 TensorFlow 程序内置了 TensorBoard 功能，可以帮助用户以图形化方式对 TensorFlow 程序的运行过程进行监控。在成功提交作业后，可进入作业详情界面，并选择 TensorBoard 栏
8. 点击浏览按钮，并选择 TensorFlow 程序的输出目录，然后点击查看按钮即可启动 TensorBoard，可以点击新窗口打开按钮来获取完整视图：



图 31 TensorBoard 界面

2.9.2. 提交 Caffe 作业

Caffe 作业是指基于 Caffe 运行的单机计算作业，LiCO 系统通过 Singularity 容器镜像来运行基于 Caffe 框架编写的 AI 程序，可支持 CPU 或 GPU。

用户可以通过以下步骤在 LiCO 系统中提交并运行 Caffe 作业：

1. 进入作业模板选择界面；
2. 在模板中选择 Caffe 模板，可进入如下图所示的 Caffe 作业提交界面：

The screenshot shows the 'Caffe' job submission interface. At the top, there's a brief introduction about Caffe. Below it, the 'Template Information' section contains a field for the 'Job Name'. The 'Template Parameters' section includes fields for 'Working Directory' (必选), 'Container Image' (必选, set to 'caffe_gpu_new_build'), 'Caffe Program' (必选), 'Run Parameters' (empty), 'Number of CPU Cores' (必选, set to 1), and 'Number of GPUs' (empty). The 'Resource Options' section has a dropdown for 'Queue' (必选, set to 'compute'). At the bottom is a blue 'Submit' button.

图 32 Caffe 作业提交界面

3. 填写作业名称及 Caffe 作业相关运行参数：

工作目录（必选）：程序的工作目录；

容器镜像（必选）：作业所使用的 Singularity 镜像；

Caffe 程序（必选）：选择需要运行的 Caffe 程序文件；

运行参数：Caffe 程序自带的一些运行参数；

使用 CPU 核数（必选）：运行 Caffe 程序所用的 CPU 核数；

使用 GPU 数：运行 Caffe 程序所用的 GPU 个数；

4. 填写资源选项；

5. 下图是一个 Caffe 作业填写的示例：

The screenshot shows a Caffe job submission interface. At the top, it says "Caffe" and provides a brief description: "Caffe is a deep learning framework made with expression, speed, and modularity in mind. It is developed by Berkeley AI Research (BAIR) and by community contributors. Use this job template to submit a single node CPU&GPU job." Below this, there are three main sections:

- 模板信息**: Contains a field labeled "作业名称" with the value "testz_caffe".
- 模板参数**: Contains several fields:
 - "工作目录": "MyFolder/demo/caffe"
 - "容器镜像": "caffe_gpu_new_build"
 - "Caffe程序": "MyFolder/demo/caffe/cifar10/cifar10_prep"
 - "运行参数": An empty input field.
 - "使用CPU核数": "1"
 - "使用GPU数": An empty input field.
- 资源选项**: Contains a field labeled "队列" with the value "compute".

At the bottom right of the form is a blue "提交" (Submit) button.

图 33 Caffe 作业示例

6. 填写完成后，点击**提交**按钮即可提交运行。

2.9.3. 提交 Intel Caffe 作业

Intel Caffe 作业是指基于 Intel Caffe 运行的单机或分布式计算作业，LiCO 系统通过 Singularity 容器镜像来运行基于 Intel Caffe 框架编写的 AI 程序，可支持 CPU 或 GPU。

用户可以通过以下步骤在 LiCO 系统中提交并运行 Intel Caffe 作业：

1. 进入作业模板选择界面；
2. 在模板中选择 Intel Caffe Standard 模板，可进入如下图所示的 Intel Caffe 作业提交界面：

The screenshot shows the 'Intel Caffe' job submission interface. At the top, it says 'This fork of BVLC/Caffe is dedicated to improving performance of this deep learning framework when running on CPU, in particular Intel® Xeon processors (iSWs) and intel® Xeon Phi processors. Use this job template to submit a multi-node CPU job.' Below this, there are sections for '模板信息' (Template Information) and '模板参数' (Template Parameters). In the '模板参数' section, there are fields for '工作目录' (Working Directory), '容器镜像' (Container Image selected as 'intel-caffe-1.0-cpu'), 'Intel Caffe程序' (Intel Caffe Program), '运行参数' (Run Parameters), '节点数' (Number of Nodes set to 1), and '每个节点使用CPU核数' (Number of CPU Cores per Node set to 1). At the bottom, there is a '资源选项' (Resource Options) section with a '队列' (Queue) dropdown set to 'compute' and a '提交' (Submit) button.

图 33 Intel Caffe 作业提交界面

3. 填写作业名称及 Intel Caffe 作业相关运行参数：

工作目录（必选）：程序的工作目录；

容器镜像（必选）：选择 Singularity 容器的镜像；

Intel Caffe 程序（必选）：选择需要运行的 Intel Caffe 程序文件；

运行参数：Intel Caffe 程序自带的一些运行参数；

节点数（必选）：分布式运行 Intel Caffe 程序所用的节点数；

每个节点使用 CPU 核数（必选）：在每个节点使用的 CPU 核数；

4. 填写资源选项；

5. 下图是一个 Intel Caffe 作业填写的示例：

The screenshot shows the 'Intel Caffe' job submission interface. It includes sections for 'Template Information' (作业名称: test_Intelcaffe), 'Template Parameters' (工作目录: MyFolder/demo/caffe/cifar10/cifar10_prep, 镜像: intel-caffe-1.0-cpu, 程序: MyFolder/caffe/caffe.sh), 'Run Parameters' (节点数: 1, 每个节点使用CPU核数: 1), and 'Resource Options' (队列: compute). A 'Submit' button is at the bottom.

图 34 Intel Caffe 作业示例

6. 填写完成后，点击**提交**按钮即可提交运行。

2.9.4. 提交 MXNet 作业

MXNet 作业是指基于 MXNet 运行的单机或分布式计算作业，LiCO 系统通过 Singularity 容器镜像来运行基于 MXNet 框架编写的 AI 程序，可支持 CPU 或 GPU。

用户可以通过以下步骤在 LiCO 系统中提交并运行 MXNet 作业：

1. 进入作业模板选择界面；
2. 在模板中选择 MXNet Single Node 模板，可进入如下图所示的 MXNet 作业提交界面：

The screenshot shows the 'MXNet Single Node' job submission interface. It includes sections for 'Template Information' (作业名称: 未输入), 'Template Parameters' (工作目录: mxnet-1.1-gpu-cuda90, 程序: mxnet_train.py), 'Run Parameters' (运行参数: 未输入, 使用CPU核数: 1, 使用GPU数: 1), and 'Resource Options' (队列: compute). A 'Submit' button is at the bottom.

图 35 MXNet 作业提交界面

3. 填写作业名称及 MXNet 作业相关运行参数：

工作目录（必选）：程序的工作目录。

容器镜像（必选）：选择 Singularity 容器的镜像。

MXNet 程序（必选）：选择需要运行的 MXNet 程序文件。

运行参数：MXNet 程序自带的一些运行参数，必选。

使用 CPU 核数（必填）：MXNet 程序运行时使用的 CPU 核数。

使用 GPU 数：MXNet 程序运行时使用的 GPU 个数。

4. 填写资源选项；

5. 下图是一个 MXNet 作业填写的示例：

The screenshot shows the 'MXNet Single Node' job template configuration. Key fields filled in:

- 作业名称**: test_MXNetsinglenode
- 工作目录**: MyFolder/demo/mxnet
- 容器镜像**: mxnet-1.1-gpu-cuda90
- MXNet程序(p)**: MyFolder/demo/mxnet/download_train_da
- 使用CPU核数**: 1
- 队列**: compute

图 36 MXNet 作业示例

6. 填写完成后，点击**提交**按钮即可提交运行。

2.9.5. 提交 Neon 作业

Neon 作业是指基于 Neon 运行的单机计算作业，LiCO 系统通过 Singularity 容器镜像来运行基于 Neon 框架编写的 AI 程序，可支持 CPU 或 GPU。

用户可以通过以下步骤在 LiCO 系统中提交并运行 Neon 作业：

1. 进入作业模板选择界面；

2. 在模板中选择 Neon 模板，可进入如下图所示的 Neon 作业提交界面：

neon

Neon
Neon is Intel Nervana's reference deep learning framework committed to best performance on all hardware, especially Intel hardware. Designed for ease-of-use and extensibility. Use this job template to submit a single node CPU job.

模板信息

* 作业名称

模板参数

* 工作目录 浏览

* 容器镜像 neon-2.4-cpu

* neon程序(.py) 浏览

运行参数

* 运行模式 MKL

* 使用CPU核数 1

资源选项

* 队列 compute

提交

图 38 Neon 作业提交界面

3. 填写作业名称及 Neon 作业相关运行参数：

工作目录（必选）：程序的工作目录。

容器镜像（必选）：选择 Singularity 容器的镜像。

neon 程序（必选）：选择需要运行的 Neon 程序文件。

运行参数：Neon 程序自带的一些运行参数。

运行模式（必选）：可以选择 CPU、MKL 和 GPU 三种模式。

使用 CPU 核数（必选）：Neon 程序运行时使用的 CPU 核数。

4. 填写资源选项；

5. 下图是一个 Neon 作业填写的示例：

The screenshot shows the Neon job submission interface. At the top, it says "neon" and "Neon is Intel Nervana's reference deep learning framework committed to best performance on all hardware, especially intel hardware. Designed for ease-of-use and extensibility. Use this job template to submit a single node CPU job." Below this is a "模板信息" (Template Information) section with a "作业名称" (Job Name) field containing "test_neon". Under "模板参数" (Template Parameters), there are fields for "工作目录" (Working Directory) set to "MyFolder/demo/neon", "容器镜像" (Container Image) set to "neon-2.4-cpu", and "neon程序(.py)" (Neon Program (.py)) set to "MyFolder/demo/neon/mnist_mlp.py". There are also fields for "运行参数" (Run Parameters), "运行模式" (Run Mode) set to "MKL", and "使用CPU核数" (Use CPU Cores) set to "1". In the "资源选项" (Resource Options) section, there is a "队列" (Queue) field set to "compute". At the bottom right is a blue "提交" (Submit) button.

图 39 Neon 作业示例

6. 填写完成后，点击**提交**按钮即可提交运行。

2.9.6. GPU 作业监控

通常为了加快 AI 作业的执行速度会使用 GPU 来进行加速，系统针对使用了 GPU 的作业提供了对执行作业的计算节点上 GPU 进行监控的功能。

用户提交作业后或者通过作业管理功能可以进入到作业详情界面，如果该作业使用了 GPU，则会出现 GPU 使用信息及监控 Tab 页，如下图：

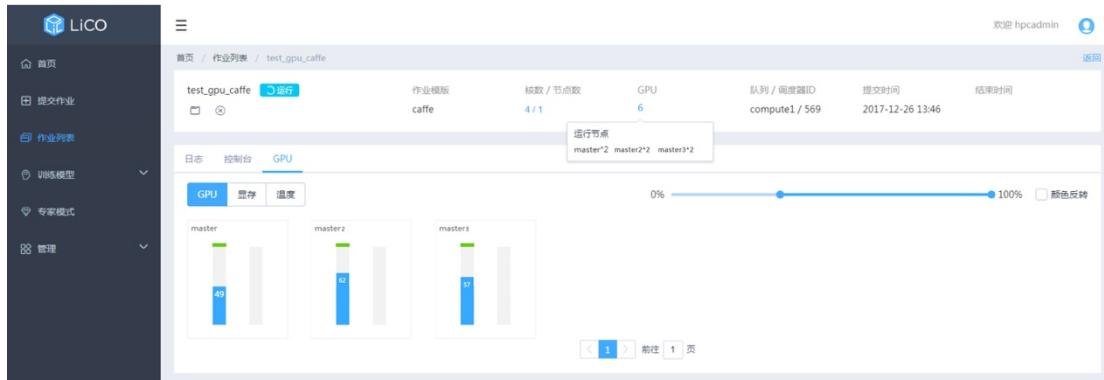


图 37 GPU 作业监控

GPU 使用情况: 系统能够列出当前作业订购的 GPU 个数，点击个数可以显示作业具体使用了哪些计算节点及在该节点上使用的 GPU 个数（实际使用的数量可能会小于订购数量）。

GPU 使用率、显存、温度监控: 点击 GPU Tab 可以切换到 GPU 实时监控界面，该界面以图形化的方式展示了运行此作业的所有节点上的 GPU 实时数据，并可以切换显示 GPU 使用率、显存、温度。图中每个框表示一个节点，框体右上角为节点名称，每一个柱体表示一个 GPU，柱体蓝色部分标示具体的监控指标数值，如果柱体顶部有桔黄色标记则表明此 GPU 已被使用。通过移动界面右上方的滑动条可以调整柱体的颜色来达到筛选并高亮数值范围内的 GPU，勾选滑动条右侧的颜色反转选项可以将范围内和范围外的标记颜色对换。

2.10. 作业生命周期管理

LiCO 系统中，作业生命周期的管理包括取消作业、重新运行作业和删除作业。

2.10.1. 取消作业

可按以下步骤操作来对正在运行的作业进行取消：

1. 进入作业管理界面，在界面头部状态选择栏点击运行中按钮，如下图所示；

ID	作业名称	调度器ID	状态	队列	开始时间	运行时间	操作
1871	test	1620	正在运行	compute	2018-04-10 14:18	1m	

图 38 查看运行中的作业

2. 在下方表格中找到要取消的作业，点击所在行的 按钮，并在弹出的确认对话框点击 **提交** 按钮即可取消作业，如下图所示；

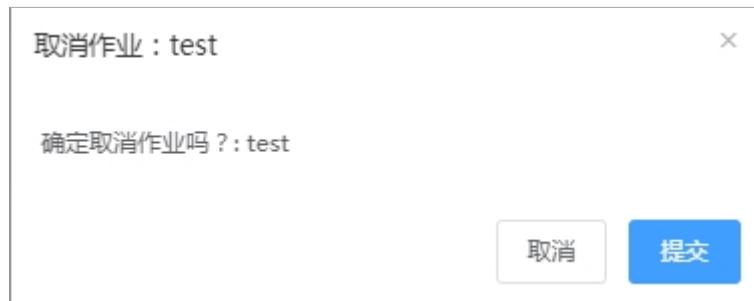


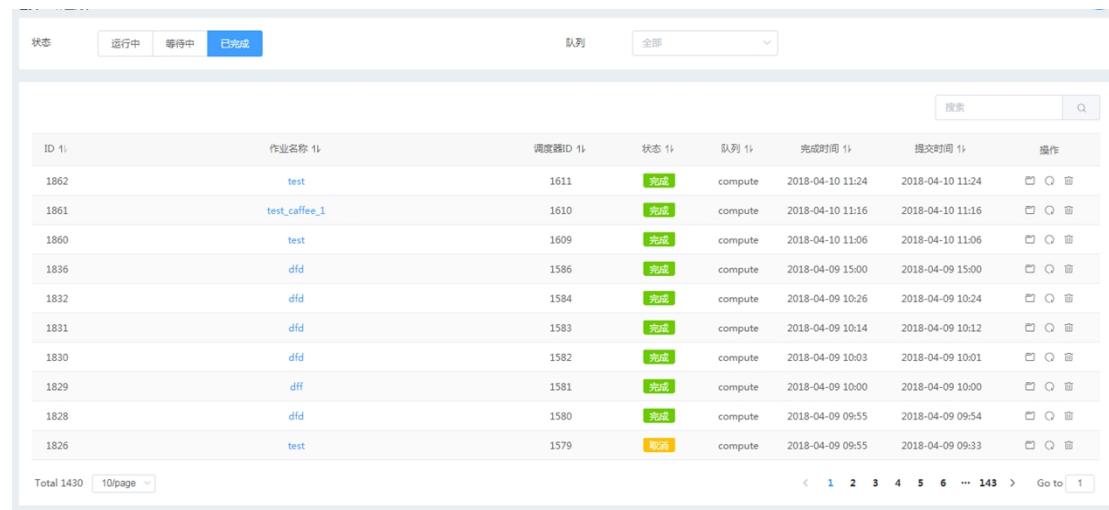
图 42 确认取消作业

3. 取消后的作业可以完成的作业列表中被找到。

2.10.2. 重新运行作业

可按以下步骤操作来对已经完成运行的作业进行重新运行：

- 进入作业管理界面，在界面头部状态选择栏点击**完成**按钮，如下图所示；



The screenshot shows a web-based job management system. At the top, there is a navigation bar with tabs: '运行中' (Running), '等待中' (Waiting), and '已完成' (Completed). The '已完成' tab is currently selected. Below the navigation bar is a search bar with a placeholder '搜索' and a magnifying glass icon. The main area is a table displaying completed jobs. The columns in the table are: ID, 作业名称 (Job Name), 调度器ID (Scheduler ID), 状态 (Status), 队列 (Queue), 完成时间 (Completion Time), 提交时间 (Submission Time), and 操作 (Operations). The table contains 1430 rows of data. The status column for most rows is '完成' (Completed), except for one row which is '取消' (Cancelled). The '操作' column for each row contains three icons: a magnifying glass, a circular arrow, and a trash can.

ID	作业名称	调度器ID	状态	队列	完成时间	提交时间	操作
1862	test	1611	完成	compute	2018-04-10 11:24	2018-04-10 11:24	
1861	test_coffee_1	1610	完成	compute	2018-04-10 11:16	2018-04-10 11:16	
1860	test	1609	完成	compute	2018-04-10 11:06	2018-04-10 11:06	
1836	dfd	1586	完成	compute	2018-04-09 15:00	2018-04-09 15:00	
1832	dfd	1584	完成	compute	2018-04-09 10:26	2018-04-09 10:24	
1831	dfd	1583	完成	compute	2018-04-09 10:14	2018-04-09 10:12	
1830	dfd	1582	完成	compute	2018-04-09 10:03	2018-04-09 10:01	
1829	dfd	1581	完成	compute	2018-04-09 10:00	2018-04-09 10:00	
1828	dfd	1580	完成	compute	2018-04-09 09:55	2018-04-09 09:54	
1826	test	1579	取消	compute	2018-04-09 09:55	2018-04-09 09:53	

图 43 查看运行完成的作业

- 在下方表格中找到要重新运行的作业，点击所在行的 按钮，并在弹出的确认对话框点击**提交**按钮即可重新运行作业，如下图所示；

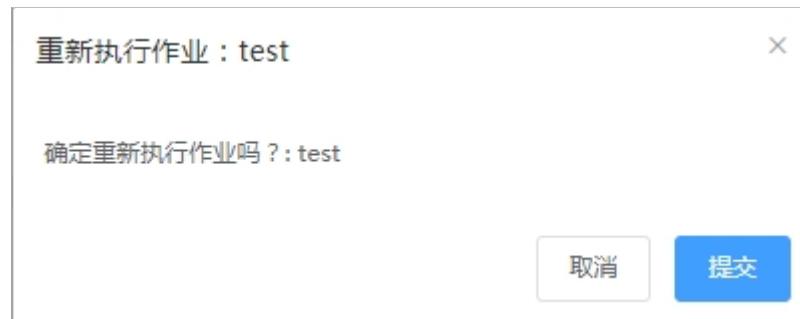
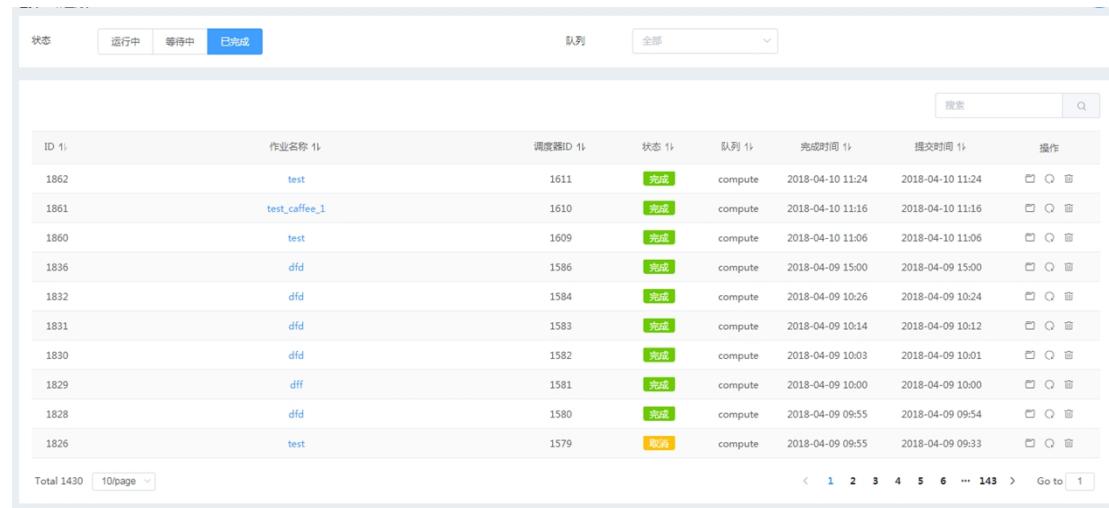


图 44 确认重新运行作业

2.10.3. 删除作业

可按以下步骤操作来对已经完成运行的作业进行删除：

1. 进入作业管理界面，在界面头部状态选择栏点击**完成**按钮，如下图所示；



The screenshot shows a table of completed jobs. The columns are: ID, 作业名称 (Job Name), 调度器ID (Scheduler ID), 状态 (Status), 队列 (Queue), 完成时间 (Completion Time), 提交时间 (Submission Time), and 操作 (Operations). The status column shows green '完成' (Completed) boxes for most rows. The last row shows an orange '取消' (Cancelled) box. The table has 1430 total rows and 10 pages.

ID	作业名称	调度器ID	状态	队列	完成时间	提交时间	操作		
1862	test	1611	完成	compute	2018-04-10 11:24	2018-04-10 11:24			
1861	test_coffee_1	1610	完成	compute	2018-04-10 11:16	2018-04-10 11:16			
1860	test	1609	完成	compute	2018-04-10 11:06	2018-04-10 11:06			
1836	dfd	1586	完成	compute	2018-04-09 15:00	2018-04-09 15:00			
1832	dfd	1584	完成	compute	2018-04-09 10:26	2018-04-09 10:24			
1831	dfd	1583	完成	compute	2018-04-09 10:14	2018-04-09 10:12			
1830	dfd	1582	完成	compute	2018-04-09 10:03	2018-04-09 10:01			
1829	dfd	1581	完成	compute	2018-04-09 10:00	2018-04-09 10:00			
1828	dfd	1580	完成	compute	2018-04-09 09:55	2018-04-09 09:54			
1826	test	1579	取消	compute	2018-04-09 09:55	2018-04-09 09:53			

图 39 查看运行完成的作业

3. 在下方表格中找到要删除的作业，点击所在行的 按钮，并在弹出的确认对话框点击**提交**按钮即可删除作业，如下图所示；

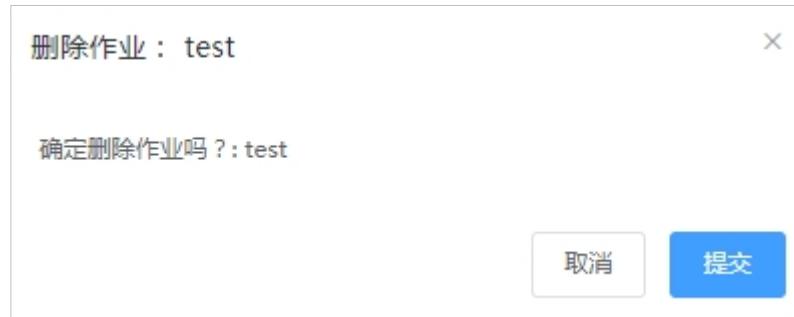


图 40 确认删除作业

2.11. 训练 AI 图像分类模型

LiCO 支持用户在不编写程序的情况下训练基于深度学习的图像分类模型。目前该模型的训练过程是基于 Intel Caffe 框架开发的，可支持分布式训练，并针对 Intel Xeon Phi 进行了性能优化。

LiCO 支持对图像数据集、网络拓扑、模型的管理，并能够对模型的训练过程进行实时监控、测试及导出。

2.11.1. 导入图像数据集

图像数据集是带有分类信息的图片集和，是模型训练的素材。在进行模型训练之前首先要导入图像数据集。

LiCO 系统支持导入以 ZIP 或 TAR 方式打包的图片压缩包。同时要求压缩包内的图片按照不同的分类分别放置在不同的子文件夹内，子文件夹的文件名需为分类名。

当用户按照规范将图片压缩包准备好并且上传到集群文件系统后，可按照如下步骤进行操作来导入图像数据集：

1. 依次点击菜单：训练模型 -> 数据集，进入如下图所示的数据集管理界面：

The screenshot shows a table of dataset entries. Each entry includes an ID, name, dimensions, size, description, creation time, status, and operations. The table has 13 rows, with page 1 of 10 shown at the bottom.

ID	名称	图像尺寸	大小	描述	创建时间	状态	操作
61	a222	28 x 28	1.4 MB		2018-04-02 18:24	完成	<input type="checkbox"/> <input type="checkbox"/>
53	ls22	28 x 28	1.4 MB		2018-03-29 10:27	完成	<input type="checkbox"/> <input type="checkbox"/>
51	cifar10_3	28 x 28	53.1 MB		2018-03-27 14:20	完成	<input type="checkbox"/> <input type="checkbox"/>
49	ls3	28 x 28	1.4 MB		2018-03-27 14:17	完成	<input type="checkbox"/> <input type="checkbox"/>
38	cifar10_2	28 x 28	53.1 MB		2018-03-26 17:52	完成	<input type="checkbox"/> <input type="checkbox"/>
32	cifar10_1	28 x 28	53.1 MB		2018-03-26 10:16	完成	<input type="checkbox"/> <input type="checkbox"/>
23	uiiu	28 x 28	1.4 MB		2018-03-22 16:04	完成	<input type="checkbox"/> <input type="checkbox"/>
20	cifar100_227	227 x 227	138.6 MB		2018-03-21 17:13	完成	<input type="checkbox"/> <input type="checkbox"/>
19	cifar100	32 x 32	138.6 MB		2018-03-21 15:22	完成	<input type="checkbox"/> <input type="checkbox"/>
18	cifar10	32 x 32	53.1 MB	cifar10数据集，训练校验测试分别为70, 20, 10	2018-03-21 14:01	完成	<input type="checkbox"/> <input type="checkbox"/>

图 41 数据集管理界面

2. 点击创建按钮可弹出创建数据集对话框：

The dialog box is titled "创建数据集". It contains fields for "名称" (Name), "源文件" (Source File) with a "浏览" (Browse) button, "队列" (Queue) set to "compute", and "形式" (Type) with checkboxes for "训练" (Training), "校验" (Validation), and "测试" (Testing). Below these are input fields for "训练" (70), "校验" (20), and "测试" (10). There is also a "图像尺寸" (Image Size) field (28) and a "图片类型" (Image Type) radio button group between "色彩" (Color) and "灰度" (Grayscale). A "描述" (Description) text area is at the bottom, and "取消" (Cancel) and "提交" (Submit) buttons are at the bottom right.

图 48 数据集创建对话框

3. 填写数据集的相关信息：

名称（必填）：数据集的名字，不可和其他数据集重名；

源文件（必填）：数据集的源文件，点击**浏览**并选择之前准备好的图像压缩包；

路径（必填）：数据集导入的工作目录；

训练：数据集中用作训练的数据占比；

校验：数据集中用作校验的数据占比；

测试：数据集中用作测试的数据占比；

训练、校验、测试集中至少选一个，且占比之和必须在 1-100 之间；

尺寸（必填）：数据的图片尺寸，可选的有 28 x 28, 32 x 32, 256 x 256；

图片类型（必填）：数据的图片类型，可选项为灰度，色彩；

描述：对本数据集的描述；

4. 下图是数据集填写的示例：

The screenshot shows the 'Create Dataset' dialog box. The fields are as follows:

- 名称**: test
- 源文件**: MyFolder/caffe/20180317034 (with a **浏览** button)
- 队列**: compute
- 形式**: 训练 校验 测试
- 训练**: 70
- 校验**: 20
- 测试**: 10
- 图像尺寸**: 28
- 图片类型**: 色彩 灰度
- 描述**: 测试

At the bottom right are the **取消** (Cancel) and **提交** (Submit) buttons.

图 49 创建数据集示例

5. 点击**提交**按钮即可提交一个数据集的预处理作业，用户可以在数据集列表界面查看到处理运行的状况，同时也可以在作业管理界面找到相同名称的作业。

ID	名称	图像尺寸	大小	描述	创建时间	状态	操作
62	test	28 x 28	725.4 MB	测试	2018-04-10 11:32	排队	
61	a222	28 x 28	1.4 MB		2018-04-02 18:24	完成	
53	ls22	28 x 28	1.4 MB		2018-03-29 10:27	完成	
51	cifar10_3	28 x 28	53.1 MB		2018-03-27 14:20	完成	
49	ls3	28 x 28	1.4 MB		2018-03-27 14:17	完成	
38	cifar10_2	28 x 28	53.1 MB		2018-03-26 17:52	完成	
32	cifar10_1	28 x 28	53.1 MB		2018-03-26 10:16	完成	
23	uiuu	28 x 28	1.4 MB		2018-03-22 16:04	完成	
20	cifar100_227	227 x 227	138.6 MB		2018-03-21 17:13	完成	
19	cifar100	32 x 32	138.6 MB		2018-03-21 15:22	完成	

图 42 查看数据集的处理状况

6. 当数据集导入处理完成后，可以点击对应行的 按钮来浏览工作文件夹，如下图所示：

Name	Permissions	Modified	Size	Kind
caffe_test_lmdb	read and write	Apr 02, 2018 06:24 PM	50 b	Folder
caffe_train_lmdb	read and write	Apr 02, 2018 06:24 PM	50 b	Folder
caffe_validate_lmdb	read and write	Apr 02, 2018 06:24 PM	50 b	Folder
test	read and write	Apr 02, 2018 06:24 PM	160 b	Folder
train	read and write	Apr 02, 2018 06:24 PM	161 b	Folder
valid	read and write	Apr 02, 2018 06:24 PM	161 b	Folder
mean.binaryproto	read and write	Apr 02, 2018 06:24 PM	9 KB	Plain text
verify.txt	read and write	Apr 02, 2018 06:24 PM	20 b	Plain text

图 43 浏览数据集工作目录

其中以 `lmbd` 结尾的目录为实际处理完成并在 Intel Caffe 中使用的 `lmbd` 库文件，`train`、`valid`、`test` 三个文件夹分别存放了测试、效验、测试三个子集的处理后的图片，用户可以通过文件管理器来浏览这三个文件夹中的图片。

7. 点击数据集的名称，可以进入如下图所示的数据集详细信息界面：

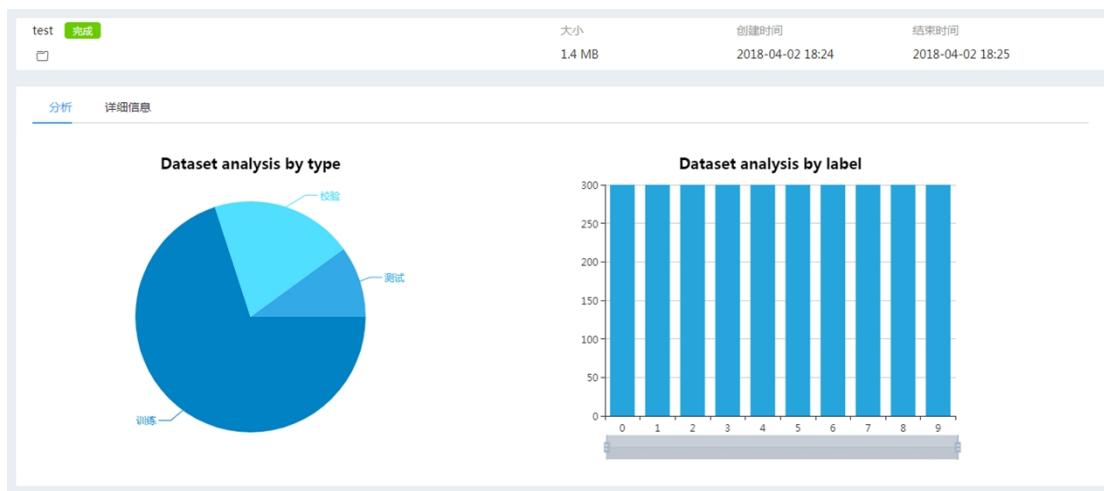


图 44 数据集详细界面

8. 用户可以在数据集详细页面以图形方式查看各子集、各分类的分布情况。

2.11.2. 创建网络拓扑

LiCO 系统使用网络拓扑来定义基于深度学习的神经网络结构和算法，目前 AI 模型的训练是基于 Intel Caffe 框架的，所以网络拓扑的定义采用 Caffe 的定义方式，数据格式为 Google Protocol Buffers Text。

网络拓扑分为两种类型：系统、私有。系统类型为 LiCO 内置提供的类型，无法修改删除，目前提供 LeNet、AlexNet、GoogLeNet 三种标准网络拓扑；私有类型为用户管理的类型，用户可以创建、修改、删除私有网络拓扑。

用户可以按照如下步骤来创建私有网络拓扑：

1. 依次点击菜单：**训练模型** → **网络拓扑**，进入如下图所示的网络拓扑管理界面：

The screenshot shows a table-based interface for managing network topologies. The columns are labeled: ID, 名称 (Name), 框架 (Framework), 类型 (Type), 描述 (Description), 创建时间 (Creation Time), and 操作 (Operations). The table contains 10 entries, each with a unique ID and name, the Intel-Caffe framework, and the '私有' (Private) type. The '操作' column includes icons for edit and delete. At the bottom, there are pagination controls showing 'Total 35' and '10/page'.

ID	名称	框架	类型	描述	创建时间	操作
69	haha	Intel-Caffe	私有		2018-04-02 17:54	
68	ls111	Intel-Caffe	私有		2018-03-30 15:22	
63	lenet_input	Intel-Caffe	私有		2018-03-27 15:15	
62	lenet_loss	Intel-Caffe	私有		2018-03-27 10:39	
61	lenet_add_input	Intel-Caffe	私有		2018-03-27 09:45	
60	test_nosize_a	Intel-Caffe	私有		2018-03-24 19:13	
55	ls_cifar10_rename	Intel-Caffe	私有	rename FN layer	2018-03-22 19:11	
54	dfsfsds	Intel-Caffe	私有		2018-03-22 18:23	
53	ls_cifar10	Intel-Caffe	私有		2018-03-22 18:16	
52	ls_32_6	Intel-Caffe	私有	3 conv 2 pool	2018-03-22 17:50	

图 45 网络拓扑管理界面

2. 点击**创建**按钮，系统将弹出如下图所示的**创建网络拓扑**对话框：



图 46 创建网络拓扑对话框

3. 输入网络拓扑名称等信息，后点击**提交**按钮：

名称（必选）：不能与已存在的网络拓扑相同；

图像尺寸（必选）：图像尺寸大小，与将要训练的图像数据集尺寸一致才可进行训练；

框架（必选）：目前只支持 Caffe 框架；

预训练模型：如果在下拉框中选择创建好的预训练模型，则采用预训练模型的网络拓扑；

描述：描述信息；

4. 正常**提交**完成后，可以在管理界面看到新增的网络拓扑，点击网络拓扑的名称，进入如下图所示的网络拓扑编辑界面：

```
9 }
10 transform_param {
11     scale: 0.00390625
12     mean_file: "/home/hpcadmin/ai_datastore/datasource/cifar10_3/cifar10_preprocess/mean.binaryproto"
13 }
14 data_param {
15     source: "/home/hpcadmin/ai_datastore/datasource/cifar10_3/cifar10_preprocess/caffe_train_lmdb"
16     batch_size: 64
17     backend: LMDB
18     shuffle: true
19 }
20 }
21 layer {
22     name: "data"
23     type: "Data"
24     top: "data"
25     top: "label"
26     include {
27         phase: TEST
28     }
29     transform_param {
30         scale: 0.00390625
31         mean_file: "/home/hpcadmin/ai_datastore/datasource/cifar10_3/cifar10_preprocess/mean.binaryproto"
32     }
33     data_param {
```

图 47 网络拓扑编辑界面

5. 在编辑界面的编辑框内填入网络拓扑定义文本并保存后，该网络拓扑可以开始使用。AutoSave 按钮为 On 时，表示编辑窗口会自动保存修改内容。
6. 点击编辑界面的可视化按钮，可以对编辑框内的网络拓扑进行可视化，如下图所示：

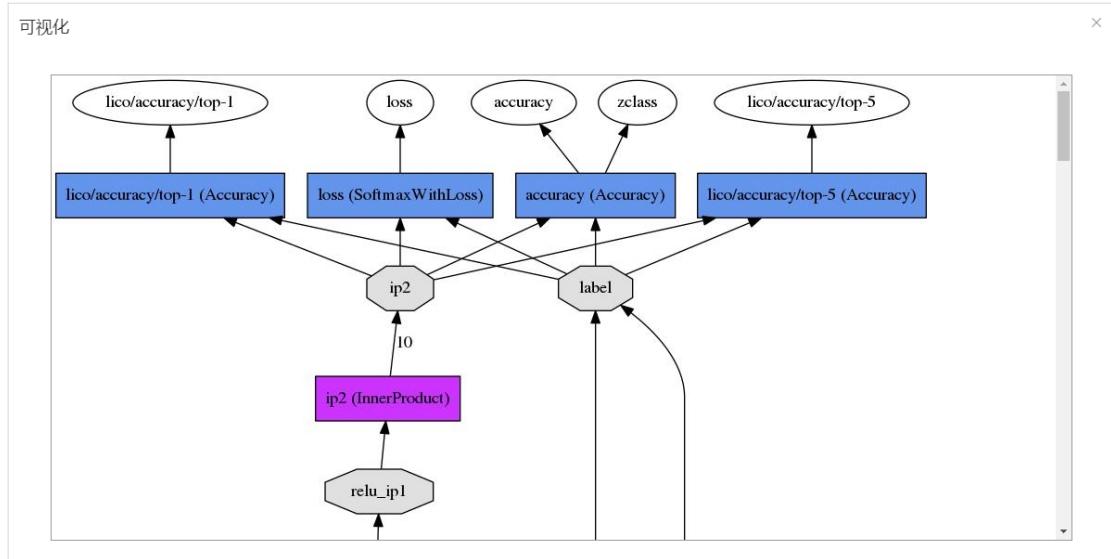


图 48 网络拓扑的可视化

2.11.3. 训练模型

LiCO 系统支持在不编写代码的情况下对 AI 图像分类模型进行训练，并可以对训练过程进行监控。在有可用图像数据集及网络拓扑条件下，用户可以按照如下步骤进行模型训练：

1. 依次点击菜单：训练模型 -> 模型，进入如下图所示的模型管理界面：

ID	名称	训练数据集	网络拓扑	描述	创建时间	状态	操作
539	aaa_20180408160433	ls22	ls111		2018-04-08 16:04	取消	编辑
538	aaa_20180408160414	ls22	ls111		2018-04-08 16:04	完成	编辑
537	aaa_20180408160341	ls22	ls111		2018-04-08 16:03	错误	编辑
536	aaa_20180408160304	ls22	ls111		2018-04-08 16:02	错误	编辑
535	aaa_20180408160239	ls22	ls111		2018-04-08 16:02	错误	编辑
532	aaa_20180404174540	ls22	ls111		2018-04-04 17:48	取消	编辑
531	aaa_20180404174515	ls22	ls111		2018-04-04 17:45	取消	编辑
530	aaa_20180404174300	ls22	ls111		2018-04-04 17:42	取消	编辑
529	aaa_20180404174003	ls22	ls111		2018-04-04 17:39	取消	编辑
528	aaa_20180403180419	ls22	ls111		2018-04-04 17:36	取消	编辑

图 49 模型管理界面

2. 点击创建按钮，可以弹出如下图的模型创建对话框：

The dialog box contains the following fields:

- 名称: aaa_20180408160433
- 训练Epochs: 32
- 训练批大小: 64
- 测试批大小: 64
- 校验批大小: 64
- 快照间隔: 1 Epochs
- 权值衰减: 0.0005
- 正则化类型: L2
- 图像归一化: 是
- 剪裁尺寸: 256x256
- 起始学习速率: 0.001
- 学习速率规则: 恒定不变
- 队列: Select
- 节点数量: 1
- 每个节点的CPU核数: 4
- 容器镜像: Select
- 描述: 无

图 58 创建模型对话框

3. 输入模型训练相关参数，并点击**提交**按钮：

名称（必选）：模型的名称，模型名称应唯一；

Python 层文件：可以选择已有的 python 层文件，相应地，后续所选择的网络拓扑文件中可以使用此 python 层文件，Python 层定义需要符合 Caffe 接口标准；

容器镜像（必选）：可以选择已有的镜像，目前只支持 Intel Caffe 镜像；

网络拓扑（必选）：点击选择按钮，在来源处选择网络拓扑，可以选择事先创建好的网络拓扑，在来源处选择预训练模型，可选择预训练模型的网络拓扑，如下图所示：



图 59 选择网络拓扑界面

初始权重（必选）：点击选择按钮，从源代码中选择一个预先训练好的模型（默认选项），您可以为预培训的模型选择初始权重。基于这些设置运行测试，如下图所示：



图 60 选择权重界面

训练数据集、校验数据集、测试数据集（必选）：选择创建好的数据集，数据集的图片尺寸与网络拓扑的图片尺寸相同；

描述：描述信息；

队列（必选）：作业所属的队列名，这里可选择的队列是用户有权限使用的队列；

引擎：可以选择 MKL2017 和 MKLDNN 两种引擎；

节点数量：本平台支持多节点分布式训练，节点数量填写想要使用的节点的数量即可；

每个节点的 CPU 核数（必选）：每个节点使用的 CPU 核数；

训练 Epochs（必选）：需要训练的 epoch 数；

训练 Batch size、测试 Batch size、校验 Batch size（必选）：训练集、测试集、校验集每次输入数据的 batch size 大小；

快照间隔（必选）：相隔多少周期后对模型进行一次保存；

权值衰减（必选）：权值衰减系数；

正则化类型（必选）：选择 L1 正则化、L2 正则化；

图像归一化（必选）：选择是否对图像进行归一化处理；

起始学习速率、学习速率规则（必选）：学习速率可以在训练过程中改变，平台支持七种学习速率规则以控制学习速率的改变：

恒定不变：指学习速率一直不变，始终等于基础学习速率；

均匀分布递增下降：需填写步长和衰减系数，指学习速率每隔一定的步长进行衰减；

多分布递增下降：需填写步值和衰减系数，指迭代次数与步值相等时，进行衰减；

指数衰减：需填写衰减系数，指学习速率呈指数衰减；

逆衰减：需填写指数和衰减系数，学习速率呈逆衰减；

多项式衰减：需填写指数，学习速率呈多项式衰减；

Sigmoid 衰减：需填写步长和衰减系数，学习速率呈 Sigmoid 衰减；

正常提交完成后，系统将切换到如下图的模型详细界面：



图 50 模型详细界面

4. 模型详细界面显示实时的训练进度、训练准确率，测试准确率以及校验准确率；在界面下方选择**监控**栏可以查看到准确率，损失函数值和学习速率的实时曲线，图形的横坐标均为周期数；选择**日志**栏可以实时查看训练的输出日志，如下图：

aaa_20180408160414 ④ ⌂ ⌂

Epoch
100% 训练准确率
100% 测试准确率
95.7% 校验准确率
97.31%

完成 2018-04-08 16:04 - 2018-04-08 16:05

监控 日志 数据分析 测试 导出 详细信息

MyFolder/ai_datastore/model/aaa_20180408160414/aaa_20180408160414

Epoch: 100% | Training Accuracy: 100% | Testing Accuracy: 95.7% | Validation Accuracy: 97.31%

```

0: I0408 16:05:15.270232 26000 caffef.cpp:501] lico/accuracy/top-1 = 0.957031
0: I0408 16:05:15.270243 26000 caffef.cpp:501] lico/accuracy/top-5 = 1
0: I0408 16:05:15.270254 26000 caffef.cpp:501] loss = 0.120114 (* 1 = 0.120114 loss)
0: I0408 16:05:15.270258 26000 caffef.cpp:501] zclass = 1
0: I0408 16:05:15.270308 26000 caffef.cpp:501] zclass = 1
0: I0408 16:05:15.270318 26000 caffef.cpp:501] zclass = 0.972222
0: I0408 16:05:15.270329 26000 caffef.cpp:501] zclass = 0.847222
0: I0408 16:05:15.270339 26000 caffef.cpp:501] zclass = 0.95
0: I0408 16:05:15.270349 26000 caffef.cpp:501] zclass = 0.972222
0: I0408 16:05:15.270359 26000 caffef.cpp:501] zclass = 0.9
0: I0408 16:05:15.270373 26000 caffef.cpp:501] zclass = 1
0: I0408 16:05:15.270377 26000 caffef.cpp:501] zclass = 1
0: I0408 16:05:15.270393 26000 caffef.cpp:501] zclass = 0.9375
0: I0408 16:05:15.270404 26000 caffef.cpp:501] zclass=lico = 1
0: I0408 16:05:15.270416 26000 caffef.cpp:501] zclass=lico = 1
0: I0408 16:05:15.270426 26000 caffef.cpp:501] zclass=lico = 0.972222
0: I0408 16:05:15.270437 26000 caffef.cpp:501] zclass=lico = 0.847222
0: I0408 16:05:15.270447 26000 caffef.cpp:501] zclass=lico = 0.95
0: I0408 16:05:15.270457 26000 caffef.cpp:501] zclass=lico = 0.972222
0: I0408 16:05:15.270468 26000 caffef.cpp:501] zclass=lico = 0.9
0: I0408 16:05:15.270478 26000 caffef.cpp:501] zclass=lico = 1
0: I0408 16:05:15.270489 26000 caffef.cpp:501] zclass=lico = 1
0: I0408 16:05:15.270499 26000 caffef.cpp:501] zclass=lico = 0.9375
0: +1m
job end time is Sun Apr 8 16:05:15 CST 2018

```

图 51 训练日志输出

5. 模型详细界面的数据分析栏显示不同类别预测的准确率，如下图：

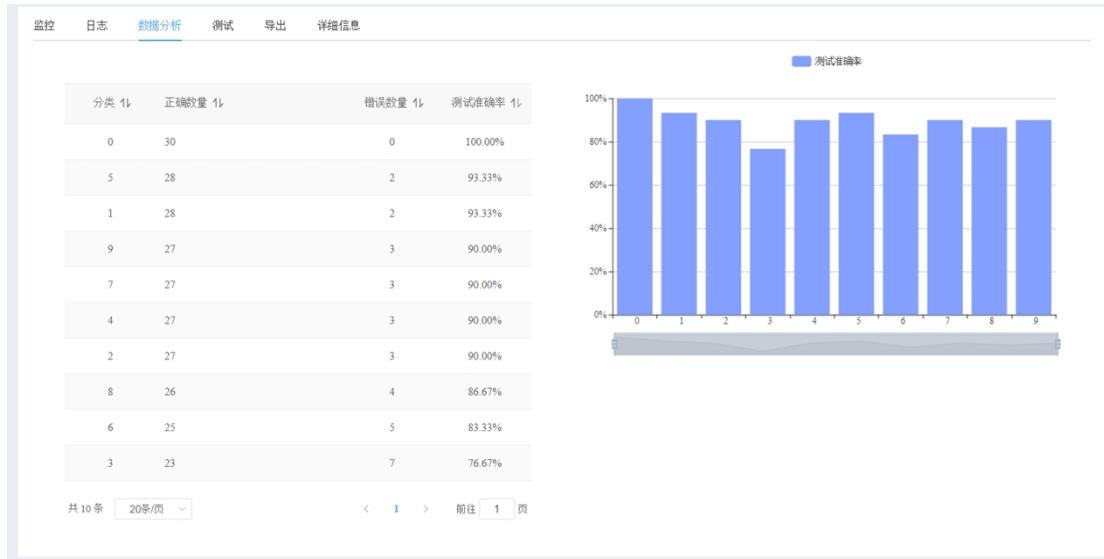


图 52 数据分析界面

2.11.4. 参数调整

LiCO 系统支持训练过程中调整训练参数并重新开始训练，在模型训练过程中，在模型详情界面点击  按钮，可以弹出参数调整对话框，如下图：



图 64 训练参数调整对话框

将参数调整完成后可以点击**提交**按钮，训练将中断并按照新参数重新运行。

2.11.5. 测试及导出模型

当模型训练结束后，按照如下步骤用户可以对模型进行测试并导出其训练结果：

1. 进入模型详情界面，选择**测试**栏，可进入如下测试界面；
2. 在快照选择处选择待测试模型，可以选择不同 epoch 时期保存的模型，默认为 Epoch#END，即最后保存的模型文件。点击界面中的**浏览**按钮，并选择一张测试图片（需要提前上传至集群文件系统），点击**测试**按钮，系统将使用模型最终训练结果来分类该图片，并显示结果，如下图：

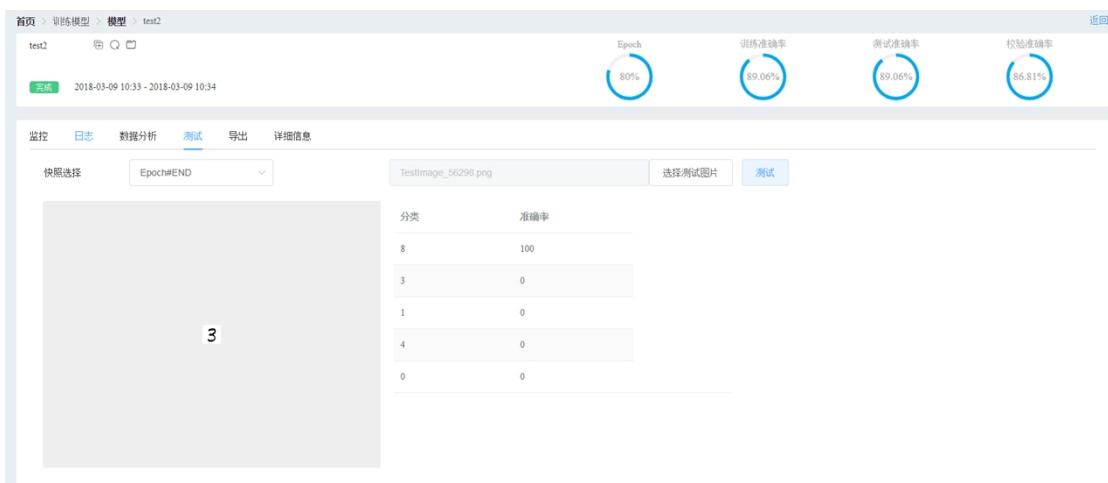


图 53 模型测试界面

3. 进入模型详情界面，选择**导出**栏，可进入导出界面，如下图：

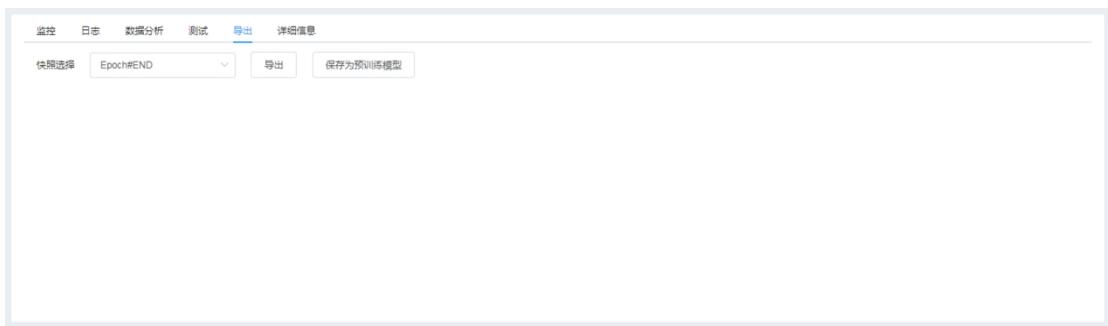


图 54 导出栏界面

4. 在快照选择处选择待测试文件，点击**导出**按钮，选择保存路径，点击**提交**按钮，即可将模型保存在集群的文件系统中，保存好的文件类型为 zip，包含网络拓扑文件和 caffemodel 文件，如下图所示：



图 55 模型导出至文件系统

5. 在快照选择处选择待测试文件，点击**保存为预训练模型**按钮，填写预训练模型名称，将模型文件以及网络拓扑保存为预训练模型，如下图所示：



图 56 导出为预训练模型

2.11.6. 管理预训练模型

LiCO 支持预训练模型管理，并支持在预训练模型的基础上进行训练。用户可以将训练结果保存为预训练模型，也可以按照如下步骤导入外部模型：

- 依次点击菜单：训练模型 -> 预训练模型，进入如下图所示的预训练模型管理界面；

ID	名称	框架	描述	创建时间	状态	操作
62	haha	Intel-Caffe		2018-04-02 17:54	完成	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
61	fdsafdsaf	Intel-Caffe		2018-03-30 18:23	完成	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
60	test12132312	Intel-Caffe		2018-03-30 18:22	完成	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
59	testssss	Intel-Caffe		2018-03-30 18:19	完成	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
55	test1098	Intel-Caffe		2018-03-30 14:52	完成	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
51	testpp	Intel-Caffe		2018-03-30 13:30	完成	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
50	test10	Intel-Caffe		2018-03-30 13:23	错误	<input type="checkbox"/>
49	test9	Intel-Caffe		2018-03-30 12:42	上传中	<input type="checkbox"/>
48	testjj	Intel-Caffe		2018-03-30 11:49	上传中	<input type="checkbox"/>
47	test11	Intel-Caffe		2018-03-30 11:47	上传中	<input type="checkbox"/>

图 57 预训练模型管理界面

- 点击创建按钮，可以弹出如下图的预训练模型创建对话框：

创建预训练模型

* 名称

* 源文件 浏览

* 框架 Intel-Caffe

* 权重文件名 请输入文件名全称

* 网络拓扑文件名 请输入文件名全称

描述

取消 提交

图 58. 预训练模型创建界面

输入预训练模型相关参数，并点击提交按钮；

名称（必选）：预训练模型的名称；

源文件（必选）：选择预训练模型，文件是 zip 或者 tar 包，须包含网络拓扑文件和 caffemodel 文件；

框架（必选）：目前只支持 Intel-Caffe；

图像尺寸（必选）：图像尺寸大小，将要训练的图像数据集尺寸一致才可进行训练；

图片类型（必选）：数据的图片类型，可选项为灰度，色彩；

权重文件名（必选）：源文件中包含的 caffemodel 文件的名称；

网络拓扑文件名（必选）：源文件中包含的 prototxt 文件的名称；

描述：描述信息；

3. 正常提交完成后，可以在管理界面看到新增的预训练模型，点击所在行的  按钮可以看到预训练模型的文件。

2.12. 自定义模板

LiCO 支持用户自定义模板，方便用户提交常用的作业。管理员在创建好自定义模板后还可以将模板发布，以供其他用户使用。发布后的模板不能编辑，需要取消发布后再编辑。

2.12.1. 创建自定义模板

用户可以通过以下步骤在 LiCO 系统中创建自定义模板：

1. 点击提交作业，之后点击右上角我的模板，如下图所示：

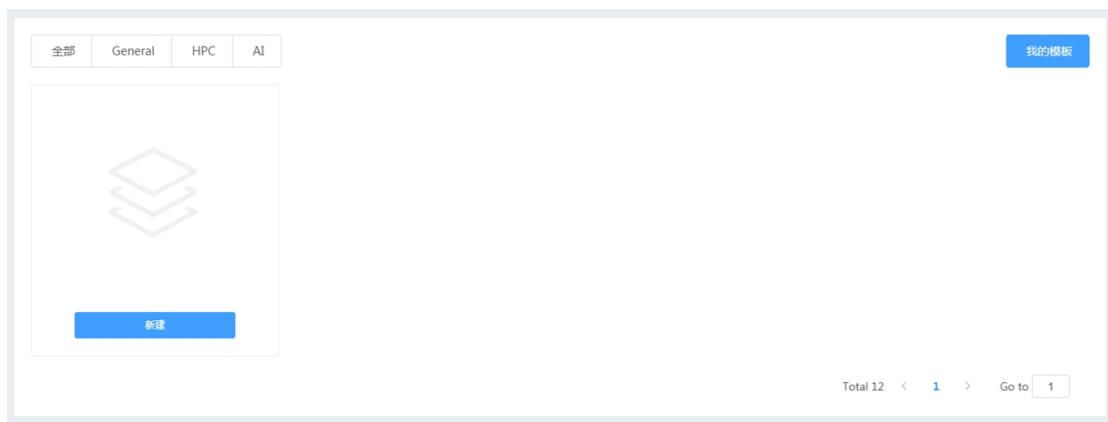


图 59 用户模板界面

2. 点击新建按钮，填写模板信息：

A screenshot of the 'Create New Template' dialog box. It has several sections:

- 模板信息**: Fields for '名称' (Name), '图标' (Icon), and '描述' (Description). There is also a note: '图标大小必须小于100KB' (Icon size must be less than 100KB).
- 模板参数**: A table showing system parameters:

ID	名称	所属资源	数据类型	必须	操作
job name	作业名称	模板信息	字符串	是	
job workspace	工作目录	模板参数	目录	是	
job queue	队列	资源选项	队列	是	
- 模板作业文件**: A code editor containing a bash script:

```
1 #!/bin/bash
2 #SBATCH --job-name={job_name}
3 #SBATCH --workdir={job_workspace}
4 #SBATCH --partition={job_queue}
```

At the bottom are '提交' (Submit) and '取消' (Cancel) buttons.

图 72 创建自定义模板

- 1) 基础信息部分：

名称（必选）：模板的名称；

Logo（必选）：上传 logo 图片，图片格式为 jpg, png, jpeg, bmp 四种，推荐大小为 180*40；

描述：描述信息。

- 2) 运行参数部分：

默认已有三个系统参数，系统参数不能被编辑与删除。用户可点击**添加参数**按钮来添加新的参数，可以添加以下几种类型的参数：

- a) 字符串类型参数，且输入方式为输入

添加参数

* ID: test

* 名称: test

* 所属段落: 模板信息

* 数据类型: 字符串

* 输入方式: 输入

* 最大长度: 255

校验规则: 字母 + 数字

默认值:

必填:

取消 提交

* ID	test
* 名称	test
* 所属段落	模板信息
* 数据类型	字符串
* 输入方式	输入
* 最大长度	255
校验规则	字母 + 数字
默认值	

图 60 添加字符串参数

ID（必选）：参数的 ID，应唯一；

名称（必选）：参数的名称，应唯一；

所属分类（必选）：参数的类别，可选择“基础信息”、“运行参数”、“资源选项”这几类；

数据类型（必选）：参数的数据类型，选择“字符串”；

输入方式（必选）：选择“输入”，以输入方式给参数赋值；

最大长度（必选）：限定参数的最大长度；

校验规则：对参数进行校验的规则，可以选择“字母+数字”、“手机”、“邮箱”；

默认值：参数的默认值；

必填：勾选则参数必填；

b) 字符串类型参数，且输入方式为选择

添加参数

* ID	test		
* 名称	test		
* 所属段落	模板信息		
* 数据类型	字符串		
* 输入方式	选择		
可选项	名称	取值	操作
添加			
默认值			
必填 <input checked="" type="checkbox"/>			
取消		提交	

图 61. 添加字符串参数

ID（必选）：参数的 ID，应唯一；

名称（必选）：参数的名称，应唯一；

所属分类（必选）：参数的类别，可选择“基础信息”、“运行参数”、“资源选项”这几类；

数据类型（必选）：参数的数据类型，选择“字符串”；

输入方式（必选）：选择“选择”，以输入方式给参数赋值；

可选项：可以在此处添加键值对，代表不同的选项。名称处填写下拉框中显示的名称，取值则为此选项的值，最多只能写 10 个选项；

默认值：参数的默认值；

必填：勾选则参数必填；

c) 数字类型参数

添加参数

* ID	test
* 名称	test
* 所属段落	模板信息
* 数据类型	数字
* 输入方式	输入
* 最小值	0
* 最大值	99999999
* 小数位数	0
默认值	

必填

取消 提交

图 62. 添加数字参数

ID（必选）：参数的 ID，应唯一；

名称（必选）：参数的名称，应唯一；

所属分类（必选）：参数的类别，可选择“基础信息”、“运行参数”、“资源选项”这几类；

数据类型（必选）：参数的数据类型，选择“数字”；

输入方式（必选）：以输入或选择的方式给参数赋值，选择“选择”时可参考上部分内容；

最小值（必选）：参数的最小值；

最大值（必选）：参数的最大值；

小数位数（必选）：参数允许的小数位数；

默认值：参数的默认值；

必填：勾选则参数必填；

d) 输入参数选择的数字参数

添加参数

* ID	test
* 名称	test
* 所属段落	模板信息
* 数据类型	数字
* 输入方式	选择
可选项	
名称	取值
<input type="text"/>	<input type="text"/>
添加	
默认值	
必填 <input checked="" type="checkbox"/>	
取消 提交	

图 63. 添加数字参数

导入的方法（要求）：选择“选择”；

选项：添加键值对来代表不同的选项。在下拉框中填写姓名，而且返回的值将此选项的值。最多 10 个选项可以添加；

e) 文件类型参数

添加参数

* ID	test
* 名称	test
* 所属段落	模板信息
* 数据类型	文件
必填 <input checked="" type="checkbox"/>	
取消 提交	

图 77. 添加文件类型参数

ID（必选）：参数的 ID，应唯一；

名称（必选）：参数的名称，应唯一；

所属分类（必选）：参数的类别，可选择“基础信息”、“运行参数”、“资源选项”这几类；

数据类型（必选）：参数的数据类型，选择“文件”；

必填：勾选则参数必填；

f) 目录类型参数



图 64. 添加目录类型参数

ID（必选）：参数的 ID，应唯一；

名称（必选）：参数的名称，应唯一；

所属分类（必选）：参数的类别，可选择“基础信息”、“运行参数”、“资源选项”这几类；

数据类型（必选）：参数的数据类型，选择“目录”；

必填：勾选则参数必填；

g) 容器镜像类型参数：



图 65. 添加容器镜像参数

ID（必选）：参数的 ID，应唯一；

名称（必选）：参数的名称，应唯一；

所属分类（必选）：参数的类别，可选择“基础信息”、“运行参数”、“资源选项”这几类；

数据类型（必选）：参数的数据类型，选择“目录”；

框架（必选）：容器镜像的框架类型，可选择 Tensorflow, Caffe, Intel-Caffe, MXNet, Neon 这几类。

必填：勾选则参数必填；

3) 模板文件部分：

在此处可以编辑模板文件内容，在模板文件中可通过 {{参数 ID}} 的方式来获取相应的参数值，例如，使用参数“作业名称”其参数 ID 为“job_name”，则可以用 {{job_name}} 来获取参数值；

3. 点击**提交**按钮，成功创建自定义模板。点击**我的模板**按钮，即可看到新建的模板。

2.12.2. 发布自定义模板

管理员可以通过以下步骤在 LiCO 系统中发布创建好的自定义模板：

1. 点击**我的模板**，可以看到已创建的模板；
2. 点击☒按钮，填写分类，点击**提交**按钮，即可发布模板，如下图所示：



图 80 发布自定义模板

发布成功后可以看到作业类别栏中增加了所填写的分类，此时普通用户也可以使用这个模板。管理员如果想取消发布，点击模板下方的☒按钮，可以取消发布。

2.13. 专家模式

LiCO 系统的专家模式提供了一个整合了 Web 控制台、文件管理、作业管理的工作界面，方便习惯于通过命令行进行作业提交和管理的高级用户在 Web 界面进行快速操作的可能。

用户可以依次点击**专家模式**菜单进入如下图所示的界面：

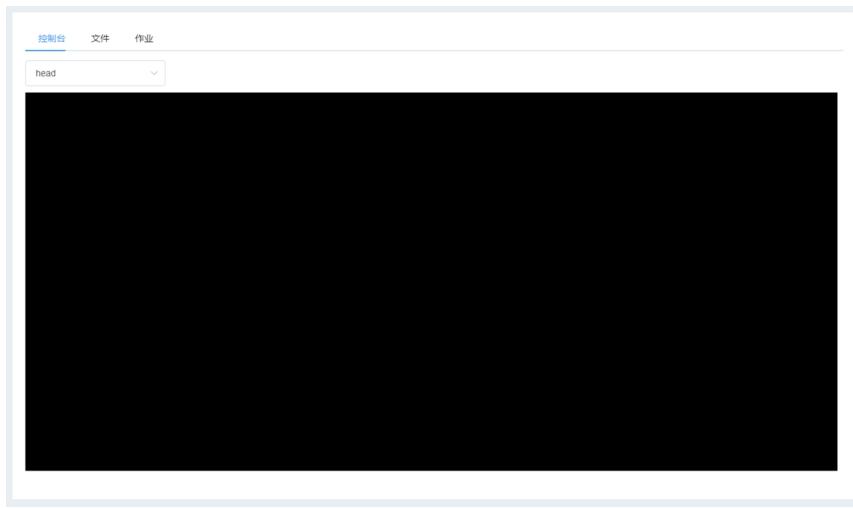


图 81 专家模式界面

专家模式页面分为控制台，文件和作业三个子界面，可以通过界面上方的导航栏进行切换。

- 控制台：系统列出当前集群内所有的登陆节点，可选择需要登录的节点进行登录：



图 82 专家模式-控制台界面

- 文件：显示当前登录用户的文件结构，功能同文件管理界面：

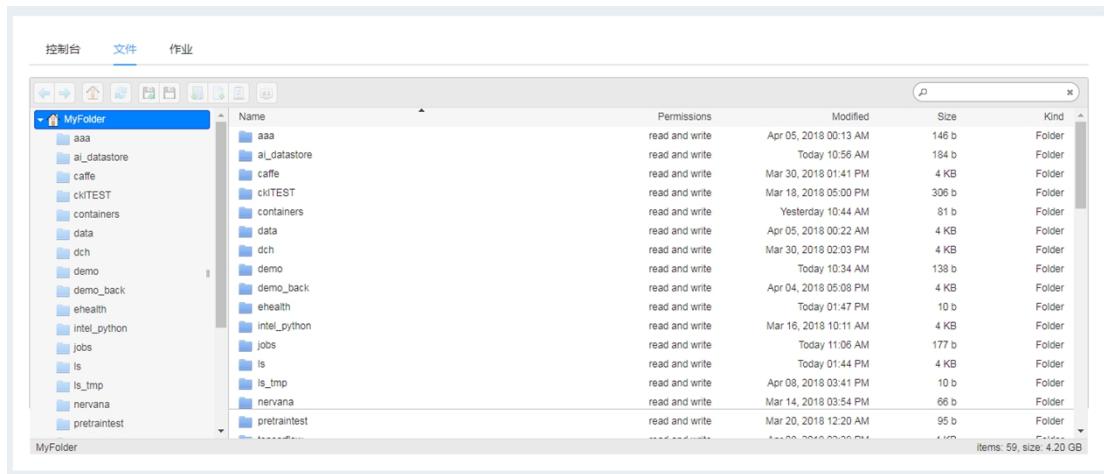


图 83 专家模式-文件界面

- 作业：显示当前集群内的作业情况，功能同作业管理界面：

ID	作业名称	调度器ID	状态	队列	提交时间	完成时间	操作
1862	test	1611	完成	compute	2018-04-10 11:24	2018-04-10 11:24	<input type="checkbox"/> <input type="radio"/> <input type="button" value="删除"/>
1861	test_caffe_1	1610	完成	compute	2018-04-10 11:16	2018-04-10 11:16	<input type="checkbox"/> <input type="radio"/> <input type="button" value="删除"/>
1860	test	1609	完成	compute	2018-04-10 11:06	2018-04-10 11:06	<input type="checkbox"/> <input type="radio"/> <input type="button" value="删除"/>
1836	dfd	1586	完成	compute	2018-04-09 15:00	2018-04-09 15:00	<input type="checkbox"/> <input type="radio"/> <input type="button" value="删除"/>
1832	dfd	1584	完成	compute	2018-04-09 10:24	2018-04-09 10:26	<input type="checkbox"/> <input type="radio"/> <input type="button" value="删除"/>
1831	dfd	1583	完成	compute	2018-04-09 10:12	2018-04-09 10:14	<input type="checkbox"/> <input type="radio"/> <input type="button" value="删除"/>
1830	dfd	1582	完成	compute	2018-04-09 10:01	2018-04-09 10:03	<input type="checkbox"/> <input type="radio"/> <input type="button" value="删除"/>
1829	dff	1581	完成	compute	2018-04-09 10:00	2018-04-09 10:00	<input type="checkbox"/> <input type="radio"/> <input type="button" value="删除"/>
1828	dfd	1580	完成	compute	2018-04-09 09:54	2018-04-09 09:55	<input type="checkbox"/> <input type="radio"/> <input type="button" value="删除"/>
1826	222	1579	取消	compute	2018-04-09 09:33	2018-04-09 09:55	<input type="checkbox"/> <input type="radio"/> <input type="button" value="删除"/>

图 84 专家模式-作业管理

2.13.1. 命令行提交作业

专家用户可以通过专家模式以 Web 控制台方式登录 LiCO 系统，并通过命令行提交作业。

可按照如下步骤进行提交：

1. 通过文件界面将编写好的作业文件上传到 My folder，也可以通过控制台文本编辑工具进行编写（如 vi 等），假定作业文件上传到 My folder/demo_job/ 下；
2. 在控制台方式登录某个登陆节点，并进入 My folder 对应绝对路径的 demo_job 目录下（绝对路径请参考：3.1 用户相关目录的绝对路径）
3. 执行命令 `sbatch /share/users_root/hpcadmin/jobfile.slurm`
注：建议使用作业文件的绝对路径，这样在作业管理界面重新运行的时候可避免相对路径导致的异常问题。
4. 作业提交以后，执行命令 `sinfo` 查看作业运行状态。当作业状态更新为 **C** 时，代表作业执行完成。同时也可通过作业界面来进行监控和管理。

2.13.2. 作业文件编写

以下是一个 SLURM 作业文件的示例：

```
#!/bin/bash
#!/bin/bash
#SBATCH --job-name test
#SBATCH --partition compute
#SBATCH --nodes=2
#SBATCH --tasks-per-node=2
#SBATCH --cpus-per-task=1
cd /share/users_root/user1
python /home/licoshare/lico_dl_run.py
```

文件各部分说明如下：

#SBATCH --job-name test: 作业名；
#SBATCH --partition compute: 指定作业使用队列 batch；
#SBATCH --nodes=2: 定义作业需要的资源，2 个计算节点；
#SBATCH --tasks-per-node=2: 定义作业需要的资源，每个节点 2 个任务；
#SBATCH --cpus-per-task=1: 定义作业需要的资源，每个任务 1 个 cpu 资源；
python /home/licoshare/lico_dl_run.py: 需要运行的任务；

2.14. VNC 管理

通过 VNC 管理功能可以对集群中的 VNC session 进行管理，依次点击菜单: 工具、VNC 可进入如下 VNC 管理界面。

名称	节点	端口	用户	PID	索引	状态	操作
No Data							
Total 0	10/page						< > Go to 1

图 85 VNC 管理

在 VNC 管理界面实时显示集群中当前用户创建的所有 VNC session，包括 session 所在机器、端口号、进程 ID 等。点击 VNC session 所在行的打开按钮可以进行查看，如果该 session 是锁屏状态，则需要用户输入帐号及密码。

注：一个用户在一个节点上请尽量保持只有一个 VNC session。作业完成后，如果作业中没有将作业创建的 VNC session 删除，则可以通过 VNC 管理界面将其删除。经过测试一个用户在一个节点上有超过 20 个 VNC session，继续在这个节点上创建有可能导致失败。

用户也可以登录到集群中的节点上管理 VNC session：

在集群中的一个节点上，通过 vncserver -list 查看自己创建的 VNC session

在集群中的一个节点上，通过 vncserver -kill 删除自己创建的 VNC session

上述命令行的操作结果会在 VNC 管理上反应出来，但大约有 30 秒的延迟。

3. 注意事项

3.1. 用户相关目录的绝对路径

在 LiCO 系统中用户拥有完全控制权限的目录是在文件管理界面中看到的名为 My folder 的目录。该目录为映射目录，其在操作系统中的实际目录为配置文件 lico_5.x/etc/conf.yaml 中 user_rootdir 参数对应的用户总目录下的以用户名命名的文件夹。

例如，user_rootdir 设置的路径为 /share/users_root，用户 hpcadmin 的根目录（My folder 目录）的绝对路径是 /share/users_root/hpcadmin，同时该路径也是此用户在操作系统中

的 home 目录。用户在使用 Web 控制台或登陆系统的登陆节点，可以通过此绝对路径或 home 目录路径来访问。

3.2. 解决作业提交失败

如果在 LiCO 系统上提交作业失败，多数情况下是由于调度器服务配置不正确引起的。可以按照以下方法进行检查来排除故障：

- 使用 SSH 工具登录到系统的登录节点，以命令行的方式重新提交这个作业：

进入用户目录并找到作业文件，执行 sbatch jobfile.slurm 提交这个作业，并查看错误日志。其中最常见的问题是资源超限，比如集群中共有 80 个 CPU 核，而作业需要 100 个，则会导致提交失败。

- 登陆头节点，运行 scontrol show nodes，查看集群计算节点的状态和资源情况。

如果命令返回结果为空，那么说明节点没有加入到调度节点里面，请通知管理员

如果命令返回结果中有节点处于 down 的状态，那么说明有些节点不工作，请通知管理员

- 登陆头节点，运行 sinfo，查看队列设置情况，发现异常的话请通知管理员

3.3. VNC 查看或删除失败

如果在 VNC 管理界面上查看 VNC 失败，请联系管理员。

如果在 VNC 管理界面，删除 vnc session 失败，可以通过 SSH 登录 vnc session 所在的节点，执行 vncserver -list 查看 session 状况，并使用 vncserver -kill 删除 vnc session。删除后等待 30s 左右，重新刷新 VNC 管理界面。

3.4. SLURM 命令参考

请参考如下网页：

<https://slurm.schedmd.com/quickstart.html>

3.5. Caffe 网络拓扑定义参考

Caffe 网络拓扑定义采用 Google Protocol Buffers Text 格式，各字段定义请参考如下网页：

<http://caffe.berkeleyvision.org/tutorial/>

3.6. GPU 监控数据来源

LiCO 系统只能对 Nvidia 公司出品的 GPU 进行监控，其监控数据（包括 GPU 使用率、显存、温度、占用情况）是通过 Nvidia 官方 API 获取的。

如果您需要在节点的操作系统内查看 GPU 监控数据，可以在命令行运行 nvidia-smi 来查看。