项目信息

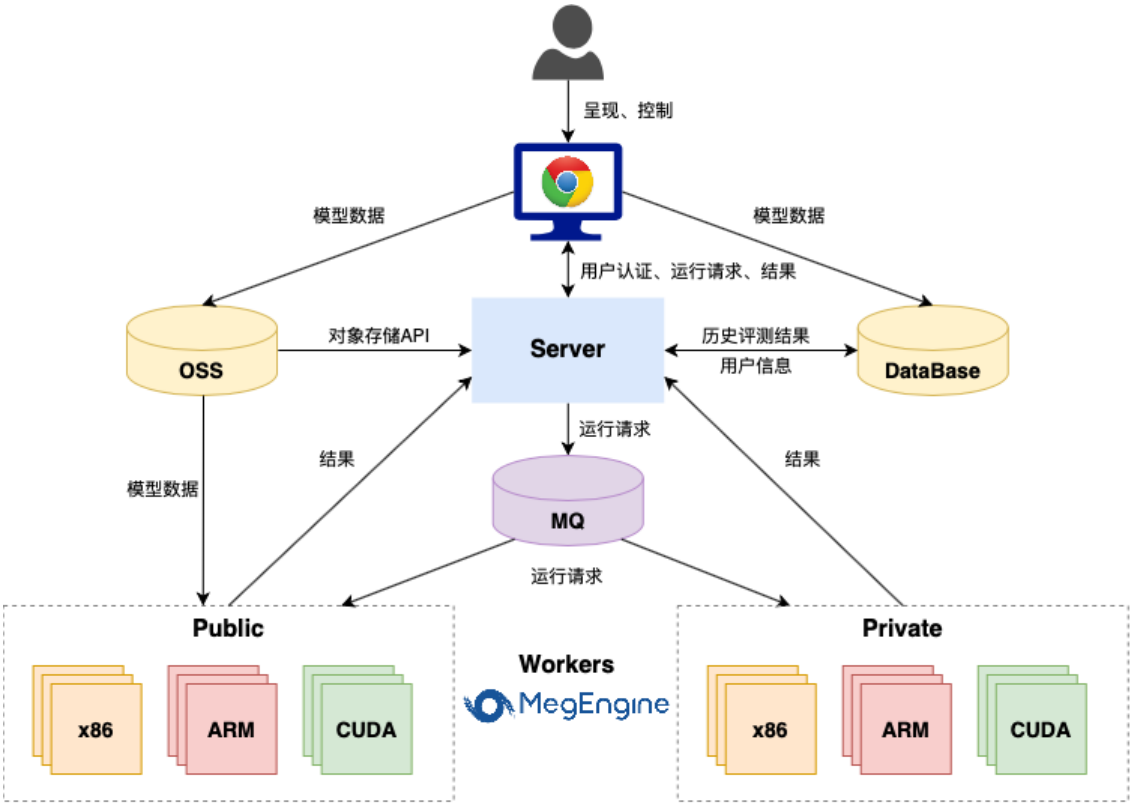
1 项目名称

一个对用户友好的 web profile 工具

210040021

2 方案描述

项目整体架构如下图所示：



- 用户通过web前端提交mge模型和输入数据，选择后端平台和MegEngine版本；登录用户可选择使用具有权限的私有设备
- Server后端为用户请求生成唯一的任务ID，并通过数据库维护任务当前状态
- 任务流程：Server后端生成->队列中等待运行资源->worker获取任务、运行并返回结果->Server存储结果至数据库并等待前端获取
- 前端使用Vue实现
- 使用`megengine.tools.network_visualize`的`visualize`函数，可实现模型结构的描述，结合开源项目`Netron`，可实现前端模型结构的展示
    - 使用`megengine.tools.profile_analyze`，可以生成模型的profile json结果，并根据用户需要在前端显示结果
- 后端（Server、Worker等）使用Python
    - 用户登录验证用户名密码匹配后，生成用户token（用户令牌），并且设置登录时效性，将token返回给前端设置本地缓存；
- Worker编译MegEngine的`load_and_run`运行模型

- 对象存储服务使用MinIO

3 时间规划

时间	前端	Server	Worker
07.01-07.05	用户登录、注册界面	用户管理相关的函数	/
07.06-07.10	上传页面、用户所有项目展示页面，上传功能的实现	上传、展示的函数；创建对象存储来存放文件；编译了load_and_run；试运行profile	初步的获得文件并运行load_and_run
07.17-07.23	测试已完成函数	测试已完成函数	/
07.24-07.30	profile展示页面，并加上用户选择（最慢/总耗时最大等的几个算子）	迁移profile_analyze功能；处理profile返回需要展示的内容给前端（先用后端处理profile）	/
07.31-08.06		部署MQ	部署MQ
08.07-08.15	mge模型展示页面的搭建	解析mge模型函数	/
08.16-08.20	管理员页面及功能（用户管理，worker管理）	管理员功能（私有worker的处理，worker信息获取等）	arm版本worker部署
08.21-08.27	/	worker注册、心跳函数	worker的config，worker注册，worker心跳，实现一个worker多版本
08.28-09.03	任务列表和结果页面连接（跳转）	/	/
09.04-09.10	完善页面、支持上传多个数据文件、load_and_run附加功能	/	多个数据文件的处理，arm版worker测试

时间	内容
09.11-09.17	完成项目api文档，完善各处的报错消息
09.18-09.25	完成项目Dockfile和部署流程文档
09.26-09.30	项目总结

项目总结

1 项目产出

计划产出	完成情况
前端可以收集模型、可视化、打印 profile 结果	完成了前端web页面的实现，包括用户登录、注册页面；创建任务页面；模型可视化页面；展示任务profile结果页面；管理员管理用户页面；管理员管理Worker页面。上述页面及功能覆盖了本项目提出的要求，并形成了基本的可应用于实际的web profile工具
后端负责实际跑模型，支持可以覆盖 x86/cuda/arm	后端由Server端和Worker端两部分组成。Server端接收前端请求，与数据库和对象存储MINIO相连，进行用户、任务、worker信息等数据进行管理，并根据前端需要查询整理数据并返回结果。Worker核心为一个python脚本，根据具体硬件更改脚本中架构及支持的MegEngine版本等信息，运行脚本即可增加可用的硬件。任务的分配与管理通过RabbitMQ实现

其余产出包括：

1. 项目部署说明（README.md）
2. 项目使用说明
3. 项目api文档

2 方案进度

原定方案和规划为：

时间	原定规划	是否完成
07.01-07.15	完成前端到Server的API设计；调研数据库、MQ的选型	是
07.16-07.31	完成基本的任务生命周期管理，暂时跳过MQ机制，完成单个节点上的任务运行并返回结果（实现L0、L4功能）	是
08.01-08.07	实现跨平台的worker（从MQ获取数据、运行并返回结果）（L3）	是
08.08-08.23	任务队列和调度机制的实现（基于MQ工具或自行实现），设计前端界面	是
08.24-09.07	完善前端功能，包括前端基本界面的实现、前端实现模型结构预览、前端处理profile json 结果等（L1、L2）	是
09.08-09.15	系统整合联调	是
09.16-09.30	综合调试与测试，修复错误，完善文档	是

实际完成顺序和规划与原定方案基本吻合，正常地按照计划完成了项目。

3 遇到的问题及解决方案

1. MegEngine编译

   MegEngine当前缺少发行版的包，需要手动编译。

2. MQ部署

   在项目设计中，任务队列和调度机制需由消息队列MQ来实现，但在项目初期我仅了解MQ的功能及概念。当项目进行到需要加入MQ时，我通过调研四大常用MQ选择了易于开发且使用人数较多的RabbitMQ，并根据手册实现了本项目中所需的队列架构。

3. 前端模型展示

   项目需求中指出需要前端支持模型结构预览。最初设想的简单方法是在server端通过tensorboard生成模型结构图片并放到前端展示，但这样的展示过于单调，缺少用户交互功能；而在后端将tensorboard的页面反代出来，难以进行二次开发，同时需要对服务的生命周期进行管理；前端使用wasm的graphviz，直接对MegEngine生成的dot文件进行渲染速度太慢；使用d3.js直接开发可视化页面工程量过大；最后选择开源项目netron，将其静态页面嵌入当前网站中。

4. 项目部署

   为了便于其他用户使用并部署该项目，我决定采用docker对项目的各个组件分别进行封装。在封装Server端代码时，由于代码对flask、MegEngine的版本有严格的依赖，项目中所需的pip 包不能简单使用最新版本，因此采用requirements.txt对python的依赖进行维护。在测试docker部署时，发现有些服务器上，docker容器内部无法访问宿主的公网ip，多方查找资料后解决无果，更换到其他发行版的docker上则没有此问题，怀疑可能是docker一些版本的bug。

4 项目完成质量

本项目基本实现了预期项目要求，完成度较高，可以帮助MegEngine的开发人员测试并管理其mge模型，并方便地分析结果，形成了一个基本功能完备的web profile工具。

5 与导师沟通及反馈情况

在整个项目期间，我和导师定期在每周五晚9点进行交流，我会展示上周的完成情况并对下周工作作出计划，而导师会帮助我解决出现的问题，并对下一步需要完成的任务作技术指导。非常感谢导师在整个项目期间的充分指导和参与，让我时时获得正向反馈，从而更有动力去完成项目并学习知识。