NV Hackthon 阿里安全AI平台路线安排

# 技术路线

## quake编译器整体架构

目前安全部使用的AI平台quake包含编译器组件，编译器整体架构如Figure 1所示。主要组件分为4部分。

### 前端：解析器（parser）/生成器（writer）

用于对接前端框架，解析器对原始计算图进行解析得到中间描述（IR），生成器用于根据中间描述生成匹配inference环境的计算图（注，inference环境可以是Tensorflow，Pytorch等深度学习框架，也可以是TensorRT，TVM等部署引擎）

### 优化层：变形器（transformer）/剪枝器（pruner）/量化器（quantizer）

用于优化模型，优化方式包括典型的节点融合，张量折叠，子图剪枝等，也包括针对后端平台进行的变换，例如针对TRT进行l2normalization/layer normalization/group normalization等层的拆分映射；剪枝量化用于有损压缩，需要配合训练使用，此次比赛不涉及，不展开说明。

### 后端：编译器（compiler）

用于针对硬件要求进行模型参数和数据的准备，并针对硬件进行模型微调。借助生成器进行代码生成并构建出引擎。在生成引擎的基础上，还可以将引擎镶嵌回前端（例如tensorflow的模型生成TRT引擎之后，在使用tf的plugin算子包裹引擎，构建tensorflow计算图）完成引擎和算法的无缝衔接。

### 自动优化（gym）

高级主题，用于结合强化学习/进化算法/搜索算法等进行自动模型优化，比赛中不涉及，不展开说明。

|  |
| --- |
| page3image53085408 |
| Figure 编译器整体架构 |

## quake编译器工作流程

编译器工作流程如Figure 2所示，模块对应Figure 1中展示部分。主要流程说明如下

|  |
| --- |
|  |
| Figure 编译器整体流程 |

全局管理：Router(a)对全局流程进行管理，针对编译需求进行组件实例的生成，并管理每一步骤的输入/产出

IR生成：Parser(b)和Transformer(c)完成核心IR的生成，得到的是优化后可以在前端部署的模型IR和变形过程参数改变的记录pattern，pattern和参数加载器loader（d）共同作用，将元计算图的参数变形为适配优化IR的参数，供后续步骤使用。

前端计算图生成：Writer(e)和Loader(d)共同作用生成前端格式的计算图，并使用前端的runtime进行inference，这一步骤是为了验证产生的IR的正确性，并进行数据采集。Collector(f)进行数据采集，存储的数据可以用于后端引擎生成的debug/核验过程，也可用于Int8量化的数据输入。

验证与数据采集之后，再次使用Transformer(g)进行图变换，针对硬件限制进行计算图的微调

使用微调之后的IR，前端runtime验证之后的计算图，以及数据采集器得到的数据对整图进行数据以及参数填充，compiler(h)在此基础上尽心代码生成以，引擎生成以及部署框架匹配。代码生成得到构建引擎使用的原始代码和需要的参数（例如TRT格式的weights，验证数据，量化用校验数据等等）。在没有硬件环境的情况下，编译可以不生成最终引擎，只保留原始代码，方便在硬件环境中进行构建。部署框架适配是可选步骤，一般用于将构建的引擎和部署用环境进行结合，此次比赛不涉及不详细说明。

# 任务分工

1. 文档描述清晰 (20分)（孟通，田惠舟）
2. 代码整齐，逻辑清晰 (30分)（孟通，田惠舟）
3. 模型顺利运行，有加速效果 (30分)（孟通）
4. 如果模型开发过程中发现 TensorRT 的 bug，提交 bug，得到确认 (每个 bug 5 分)（孟通，刘彬）
5. 开发了相应 Plugin 或 CUDA 代码 (20分)（孟通）
6. 进行了Profiling，对性能优化提出进一步的意见（20分）（田惠舟，孟通）
7. 进行了INT8量化的工作（20分）（孟通）

# 时间安排

4.13-4.16:完成原始算法模型的inference，前端解析部分完成，可以得到合法的IR计算图结构

4.19-4.23:优化+编译部分完成，得到结果正确的引擎

4.26-4.30:查看profiling，准确度评估，总结bug等，完善文档