Q 输入搜索文本...

如何安装 MegEngine 用户迁移指南 當见问题汇总

模型开发 (基础篇)

深入理解 Tensor 数据结构

Rank, Axes 与 Shape 属性 Tensor 元素索引

Tensor 数据类型

Tensor 所在设备

Tensor 具象化举例

Tensor 内存布局

使用 Functional 操作与计算使用 Data 构建输入 Pipeline使用 Module 定义模型结构 Autodiff 基本原理与使用使用 Optimizer 优化参数保存与加载模型(S&L)使用 Hub 发布和加载预训练模型

模型开发 (进阶篇)

通过重计算节省显存 <u>(Recomputation)</u> <u>分布式训练(Distributed</u> <u>Training)</u>

量化 (Quantization)

<u>自动混合精度(AMP)</u>

模型性能数据生成与分析 _(Profiler)_

使用 TracedModule 发版

推理部署篇

即时编译 (JIT)

模型部署总览与流程建议

使用 MegEngine Lite 部署模型

MegEngine Lite 使用接口

使用 MegEngine Lite 部署模型进阶

使用 Load and run 测试与验证模型

工具与插件篇

参数和计算量统计与可视化

Tensor 具象化举例



我们可以借助上面这张魔方(图片来源)来直观地理解 Tensor:

- 首先,我们假设这个魔方是"实心同质"的,是一个存在于现实世界中的 Tensor;
- 这个 Tensor 里面的每个元素的类型(<u>dtype</u>)都是一致的(方方正正的形状、外加一样的做工);
- 而且这是一个维度 (<u>ndim</u>) 为 3 的结构,形状 (<u>shape</u>) 为 (3,3,3);
- 对应地,该 Tensor 的总元素个数 (size)是 3*3*3=27.

如果你将每种颜色代表一个值,而每个魔方块的值可以用其具有的颜色值之和来表示(此时中间块为零),那么不同的魔方块就具有了各自的取值,就好像 Tensor 中的每个元素可以有自己的取值一样。事实上,除了魔方以外,还有很多东西可以抽象成 Tensor 数据结构,意味着可以利用 MegEngine 进行相关的计算。

└──页
Tensor 所在设备

下一页 Tensor **内存布局**