# 深度学习系统

Tensorflow、Pytorch和MindSpore都是当前深度学习领域最受欢迎和广泛应用的深度学习框架，它们都支持图形化计算图，可以方便地搭建神经网络模型，同时也提供了丰富的工具和API，方便用户进行模型训练、评估和部署。

## TensorFlow

### 什么是TensorFlow

TensorFlow是由Google公司在2015年推出的开源深度学习框架，是最受关注的深度学习框架之一。TensorFlow的名字源于其核心数据结构Tensor（张量）和流动的计算图（Flow）。

TensorFlow的发展历程：

* 2011年，Google Brain团队在其内部使用的深度学习基础架构 DistBelief 框架上进行改进，逐渐形成了 TensorFlow，此时TensorFlow还没有开源。
* 2015年，TensorFlow 正式开源，此时处在 0.x 的非正式版本。
* 2016年12月期间，TensorFlow 逐渐支持了分布式、移动设备 与Windows 系统。
* 2017年，TensorFlow 正式发布了 1.0.0 版本。
* 2019年，TensorFlow 逐步加入了 TPU、张量处理单元等工具。
* 2019年10月，TensorFlow2.0 正式发布，内部嵌入了 Keras。

截至本书书写时，TensorFlow的最新稳定版本为2.12版本。

### TensorFlow的核心概念与架构

TensorFlow的核心概念是“计算图”（Computation Graph）和“张量”（Tensor）。在TensorFlow中，所有的计算都被表示为一个有向无环图（DAG）的形式，这个图被称为计算图，计算图中的节点表示计算操作，边表示数据流动的方向。张量则是TensorFlow中的基本数据类型，它可以被看作是多维数组或矩阵。

TensorFlow的系统结构以C API为界，将整个系统分为前端和后端两个子系统。

1. 前端系统：提供编程模型，负责构造计算图
2. 后端系统：提供运行时环境，负责执行计算图

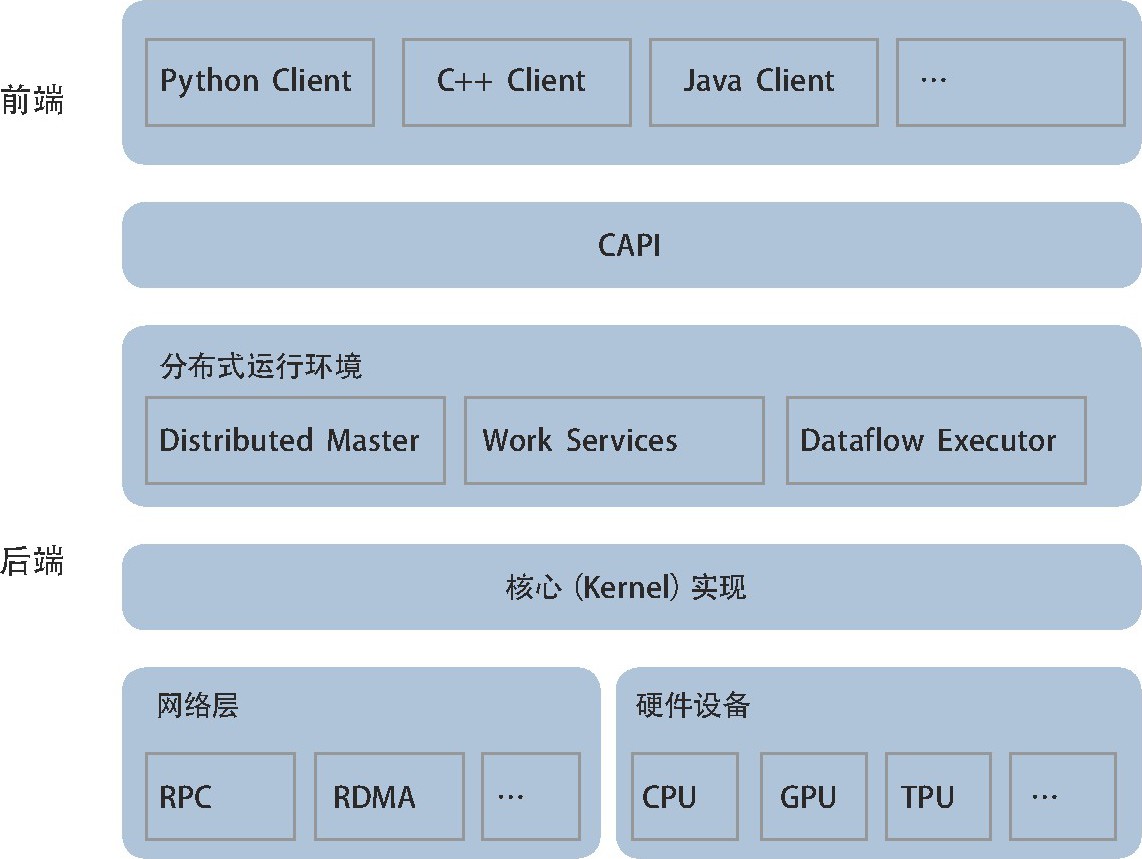


图 8‑1 TensorFlow系统架构图

TensorFlow的计算图和张量的特点使其具有高度的可移植性和可扩展性，可以运行在不同的硬件和平台上，例如CPU、GPU、TPU等，也可以在不同的操作系统上运行。同时，TensorFlow也具有丰富的社区和生态系统，支持各种第三方库的集成和使用，如TensorFlow Hub、TensorFlow Serving等。

### TensorFlow的使用和示例代码

Todo：导入包+手写数字项目的示例代码（Pytorch、MindSpore一致）

## Pytorch

### 什么是Pytorch

Pytorch是由Facebook开发的一款基于Python的深度学习框架。它于2016年首次发布，其前身是Torch框架。

Pytorch最初的设计目的是为了能够更好地进行动态计算图，即使在运行中改变计算图，也能够自动地构建梯度。这使得Pytorch在一些需要动态计算图的领域，如自然语言处理和图像处理方面，具有很大的优势。另外，Pytorch也支持静态计算图，这是由于其后端使用了TorchScript进行编译。同时，Pytorch也提供了强大的GPU加速能力。

### Pytorch的核心概念与架构

PyTorch是一个基于Python的深度学习框架，其核心概念包括张量（Tensor）、自动微分（Autograd）和动态计算图（Dynamic Computational Graph）。

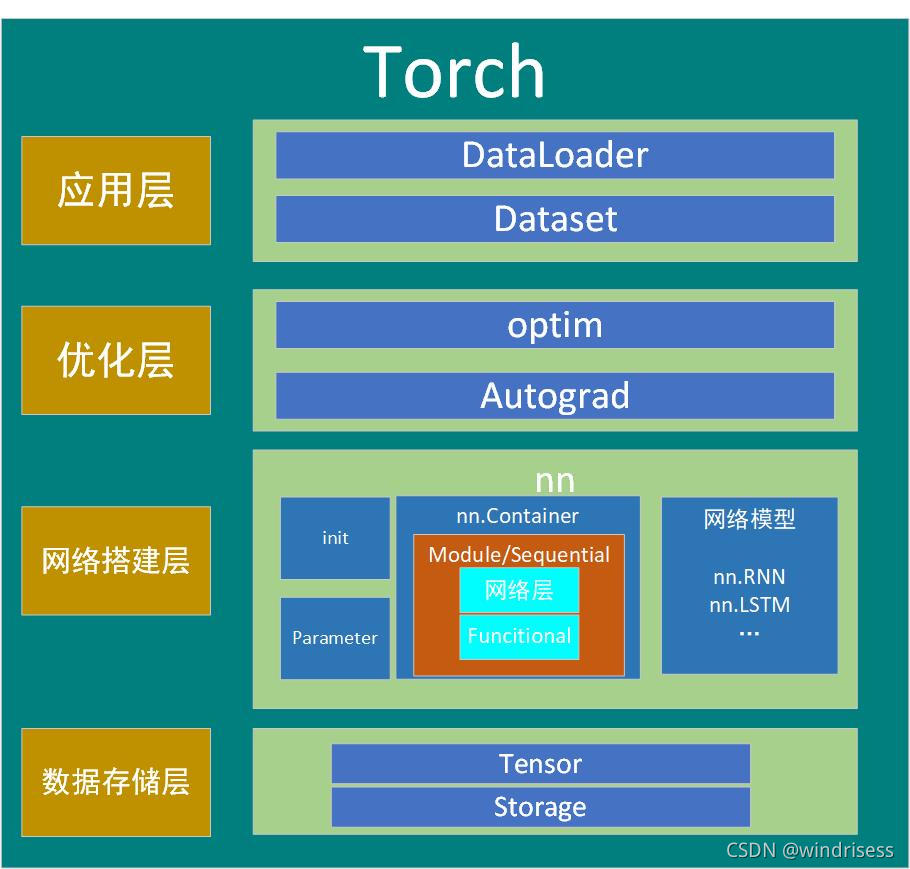


图 8‑2 Pytorch系统架构图

### Pytorch的使用和示例代码

## MindSpore

### 什么是MindSpore

MindSpore是华为于2019年发布的深度学习框架，并于2020年宣布开源。该框架是面向“端-边-云”全场景设计的AI框架，旨在弥合AI算法研究与生产部署之间的鸿沟。

### MindSpore的核心概念与架构

MindSpore在算法研究阶段，为开发者提供动静统一的编程体验以提升算法的开发效率；在生产阶段，自动并行可以极大加快分布式训练的开发和调试效率，同时充分挖掘异构硬件的算力；在部署阶段，基于“端-边-云”统一架构，应对企业级部署和安全可信方面的挑战。

MindSpore整体架构分为四层：

1. 模型层，为用户提供开箱即用的功能，该层主要包含预置的模型和开发套件，以及图神经网络（GNN）、深度概率编程、科学计算库等热点研究领域拓展库；
2. 表达层 （MindExpression），为用户提供AI模型开发、训练、推理的接口，支持用户用原生 Python语法开发和调试神经网络，其特有的动静态图统一能力使开发者可以兼顾开发效率和执行性能，同时该层在生产和部署阶段提供全场景统一的C++/Python接口；
3. 编译优化（MindCompiler），作为AI框架的核心，以全场景统一中间表达（MindIR）为媒介，将前端表达编译成执行效率更高的底层语言，同时进行全局性能优化，包括自动微分、代数化简等硬件无关优化，以及图算融合、算子生成等硬件相关优化；
4. 运行时，按照上层编译优化的结果对接并调用底层硬件算子，同时通过“端-边-云”统一的运行时架构，支持包括联邦学习在内的“端-边-云”AI协同。

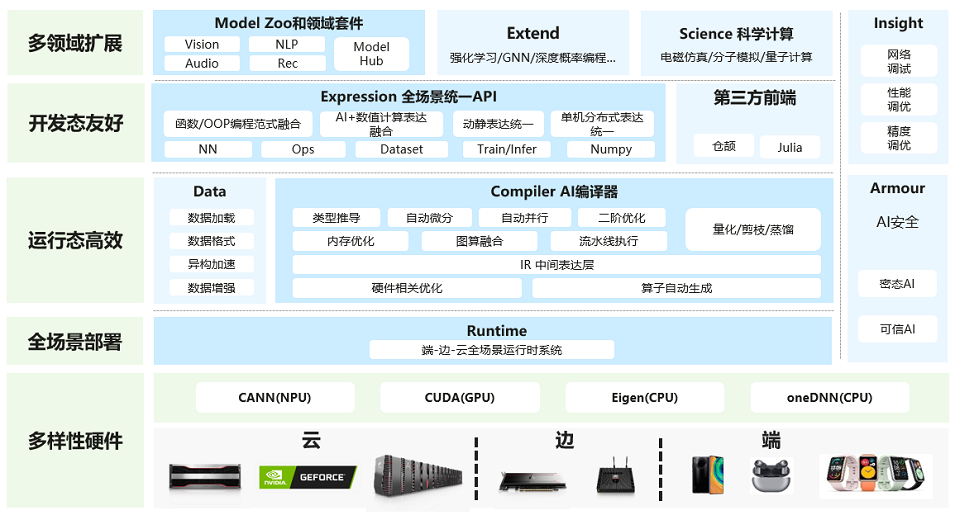


图 8‑3 MindSpore系统架构图

### MindSpore的使用和示例代码

## 区别与联系

Todo：制作一个表格进行优缺点、核心概念的比较。

TensorFlow的优点

1. 将科研和产品联系在一起。在谷歌, 科学家用它尝试新的算法, 产品团队则用它来训练和使用计算模型, 并直接提供给在线用户。
2. 真正的可移植性。它在CPU和GPU上运行, 比如说可以运行在台式机、服务器、手机移动设备等等。
3. 高度的灵活性。它不是一个严格的“神经网络”库。只要你可以将你的计算表示为一个数据流图。
4. 支持多种语言。
5. 性能最优化。由于它给予了线程、队列、异步操作等以最佳的支持。

TensorFlow的缺点

1. 调试较难, 调试时出现各种古怪的小问题较多。
2. 底层接口写起来繁琐, 高层接口又不灵活, 而且它的高层封装不好实现。
3. 它主要面向超大数据量的训练, 因此计算速度相对较慢。
4. 各个版本之间的兼容性较差, 版本迁移时需要进行修改。

Pytorch的优点

1. 动态计算图：PyTorch 的计算图是动态的，这意味着它可以根据需要在运行时构建，从而使得模型的建立和调试更加容易。这也使得 PyTorch 更加适合研究和实验。
2. 简单易用：PyTorch 的 API 非常简单易用，开发者可以很容易地理解和使用。这使得 PyTorch 成为深度学习初学者和研究人员的首选框架。
3. 灵活性：PyTorch 提供了许多灵活的选项和接口，使得开发者可以根据需求进行自定义。这也使得 PyTorch 更加适合进行研究和实验。
4. 计算速度：PyTorch 的计算速度非常快，尤其是在 GPU 上运行时。它还提供了许多优化方法，使得开发者可以更加轻松地优化模型性能。

Pytorch的缺点

1. 效率：相比其他深度学习框架，如 TensorFlow 和 Keras，PyTorch 的效率可能稍低。这意味着在处理大型数据集时，可能需要更多的时间和计算资源。
2. 缺乏成熟的生态系统：PyTorch 虽然功能强大，但与其他深度学习框架相比，其生态系统可能相对较小。这意味着可能需要更多的工作来构建自己的解决方案。
3. 不适合部署在移动端：由于 PyTorch 模型的体积较大，需要较高的计算资源和存储空间，因此不适合部署在移动端。

MindSpore优点

1. 支持昇腾芯片、GPU、CPU等异构平台，在昇腾+MindSpore组合下能做到性能遥遥领先。
2. GPU+MindSpore组合下在分布式性能高。
3. 接口易用性能有保障

MindSpore缺点

1. 社区目前较小。
2. API接口友好度需进一步提升。

## 参考文献

使用Zotero工具栏中的Add/Edit Bibliography插入参考文献。

[1]加日拉·买买提热衣木,常富蓉,刘晨等.主流深度学习框架对比[J].电子技术与软件工程,2018,No.129(07):74.

[1] 庞涛.开源深度学习框架发展现状与趋势研究[J].互联网天地,2018(04):46-54.