新一代深度学习框架研究

于璠
华为技术有限公司，广东 深圳 518000

摘要
从人工智能的历史出发，简述深度学习发展历程以及目前的挑战，通过介绍新一代深度学习框架的特点，分析总体框架，阐述自动并行、自动微分、自动调优等技术优势以及协同昇腾处理器的性能优势，希望可以为深度学习技术研究人员提供参考。

关键词
人工智能；机器学习；深度学习；计算框架；MindSpore

中图分类号：TP31 文献标识码：A doi: 10.11959/j.issn.2096-0271.2020034

Research on the next-generation deep learning framework

YU Fan
Huawei Technologies Co.,Ltd., Shenzhen 518000, China

Abstract
Started from the history of AI, the development and challenges of deep learning were described, the features of the next-generation deep learning framework was introduced, the overall framework was analyzed, and the technical advantages of auto parallel, auto differentiation, automatic tuning, as well as the performance advantages of collaborating with Ascend processors were expanded. This article can be used as a reference for deep learning technology researchers.

Key words
artificial intelligence, machine learning, deep learning, computing framework, MindSpore

1 引言

众所周知，人工智能（artificial intelligence，AI）作为影响广泛的颠覆性基础技术，将对未来各行业的发展产生深远影响。发展人工智能目前已上升到国家战略层面，但人工智能的发展并非一帆风顺。20世纪50年代，人工智能早期的发展极其缓慢，虽然在语音处理和问题求解等方面取得了不俗的表现，但仍存在很大的技术局限性；之后人工智能经历了两轮寒冬，直到20世纪90年代中期，遵循摩尔定律[1]，计算机的运算能力呈指数级增长，各种机器学习算法得以快速验证、训练和应用，从而引发了人工智能的复兴。

1997年，IBM公司制造的深蓝（DeepBlue）[2]计算机系统战胜了当时的国际象棋世界冠军卡斯帕罗夫，引起了社会各界对人工智能的高度关注，重燃了人们对人工智能的信心。自此，人工智能的新一波浪潮逐步席卷全球。在本轮人工智能浪潮中，最具价值也最具影响力的一项研究当属深度学习理论。得益于大数据的不断积累和计算机的飞速发展，海量数据解决了神经网络训练的过拟合问题，而高性能的硬件设备使模型训练成为可能。近年来，随着深度学习研究的不断深入，人们目睹了谷歌AlphaGo[3]成功击败人类世界围棋冠军以及包括无人车在内的各项智能技术的蓬勃发展，人们仿佛再一次看到了人工智能赶超人类的希望。总而言之，人工智能正在逐步改变人类的日常生活模式，并凭借其惊人的效果和迅猛的发展势头，广泛融入各个行业的实际应用中。深度学习的研究和应用在近几十年得到了爆炸式的发展，并且已经在图像识别[4]、语音识别[5]、机器翻译[6]以及游戏[7]等方面取得了巨大的成功。

为了获得更好的性能，深度学习的网络结构日益复杂，网络深度和数据集也日益增大，这给深度学习的计算框架带来了巨大的挑战。但现有的AI计算框架很难兼顾性能和易用性，而且深度学习应用已经从云侧扩展到了边缘和端侧，这对AI计算框架提出了新的挑战，如在保护用户隐私的前提下，让开发者能够实现AI应用在云、边缘和端侧的快速部署，全场景互联互通。因此，全场景的AI计算框架应满足以下3个要求：

- 实时训练或推理；
- 每个终端使用和训练其私有模型；
- 训练好的模型应能在多样化的硬件平台上运行。

为了更清晰地认识业界现有的计算框架，从易开发、高效执行和全场景3个角度对国外的TensorFlow[8]、PyTorch[9]、MXNet[10]以及国内的PaddlePaddle[11]进行了比较，见**表1**。

其中，易开发表现为API友好、调试难度低以及额外的自动化属性。高效执行包括计算效率、数据预处理效率和分布式训练效率。全场景指框架同时支持云、边缘以及端侧场景。可以看出，这些训练框架仍然不能满足之前提出的3个要求。因此，需要开发一个覆盖所有场景的深度学习框架，满足实时、专用、多样化的需求。本文详细阐述了笔者团队自研的新一代深度学习框架（MindScope）的架构、技术开发思路、性能优势。

2 MindSpore的总体框架

MindSpore是华为技术有限公司推出的新一代深度学习框架，其总体框架分为前端表示层（mind expression，ME）、计

表1 业界流行的训练框架对比

框架评价	对比项	TensorFlow	PyTorch	MXNet	PaddlePaddle
易开发	API	优秀	优秀	优秀	优秀
	Debug	优秀	优秀	优秀	优秀
	自动化	一般	一般	一般	一般
高效执行	计算效率	优秀	一般	优秀	优秀
	数据预处理效率	优秀	优秀	优秀	优秀
	分布式训练效率	优秀	优秀	优秀	优秀
全场景	云	优秀	优秀	优秀	优秀
	边缘	优秀	一般	一般	优秀
	端	一般	—	—	—

算图引擎（graph engine，GE）和后端运行时3层，如**图1**所示。

2.1 前端表示层

前端表示层向用户提供Python接口，并将用户的Python代码转换为数据流图。该部分包含Python API、MindSpore中间表示（intermediate representation，IR）统一计算图表达[12]、计算图高级别优化（graph high level optimization，GHLO）3个部分。

- Python API向用户提供统一的模型训练、推理、导出接口以及统一的数据处理、增强、格式转换接口。
- MindSpore IR提供统一的计算图表达，MindSpore基于此IR进行pass优化。
- GHLO包含与硬件无关的优化，如高级别优化（high level optimization，HLO）（如死代码消除等）、自动并行和自动微分等功能。

2.2 计算图引擎

计算图引擎负责对与硬件相关的资源进行

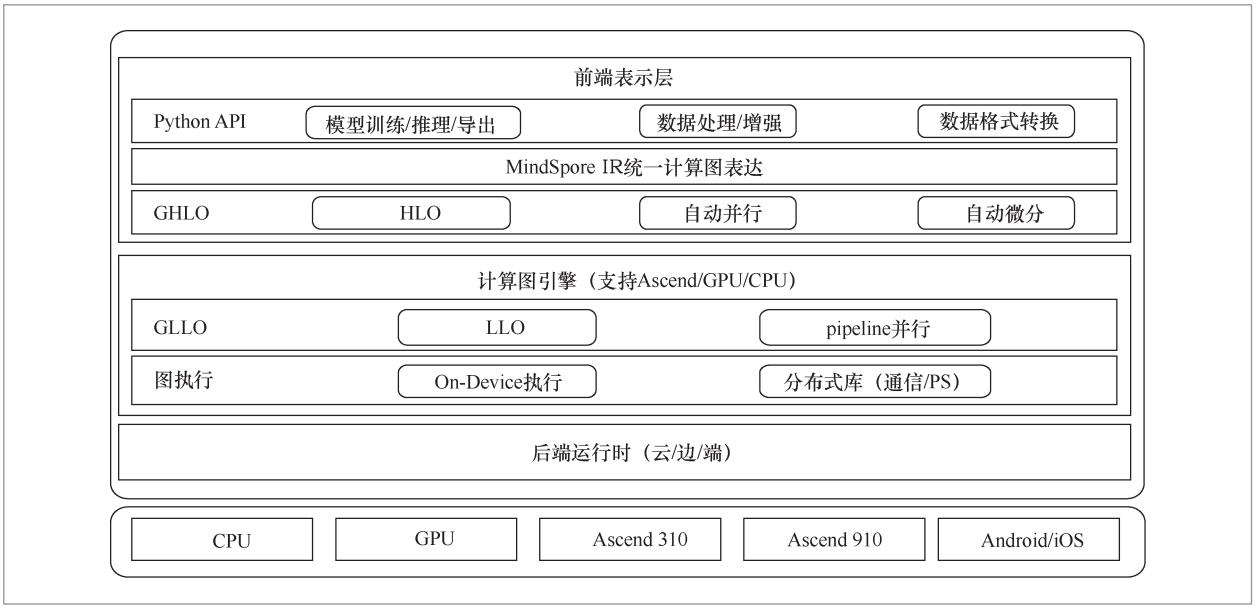

图1 MindSpore 总体框架

管理和优化，将平台特有的信息传递给ME。该部分包含计算图低级别优化（graph low level optimization，GLLO）、图执行。

- GLLO包含与硬件相关的优化，即低级别优化（low level optimization，LLO）以及算子融合、缓冲（buffer）融合等与软硬件结合相关的深度优化。
- 图执行提供离线图执行、分布式训练所需的通信接口等功能。通过图编译将各个算子组织成统一的模型并下发到设备（device）侧，在宿主（host）侧只需一个触发动作即可开启device侧的模型训练，大大减少了host与device在模型训练中的交互次数。而去中心化的梯度聚合归约自动分段融合，根据不同模型自动生成不同的融合策略，以提升通信效率，无中心点的设计可以支持更大的集群规模，降低了集群的开销。

2.3 后端运行时

后端运行时包含云、边、端上不同环境（CPU、GPU、华为昇腾处理器、Android/iOS）中的高效运行环境。

3 MindSpore的技术开发思路

MindSpore有三大技术优势，分别是自动并行、自动微分、自动调优，这些优势大大提高了深度学习模型的训练效率，同时也极大地方便了AI工程师进行编码和调试。

3.1 自动并行

随着数据集和模型越来越大，受限于单卡内存，数据并行遇到了瓶颈，需要混合并行。现在主流框架的做法是通过手动切分网络模型的方式来进行模型并行，手动切分网络模型难度非常大，对开发者的要求非常高，需要专家经验。而且在切分网络模型的同时，又要做数据并行，这极大地提高了开发的复杂度。最近有研究者提出了简化混合并行的方法，但这些方法在切分策略、适用的网络和速度上都存在局限性。

针对上述问题，MindSpore提供了自动并行特性，用于实现自动的数据并行和模型并行的混合并行训练。自动并行示意图如**图2**所示。

关键技术开发的核心思路是在并行切分方面打破样本和参数的边界，按算子的输入/输出数据进行维度切分，将算子切分到多个节点执行，进而实现并行。在MindSpore前端编译器中，通过算子切分、整图切分、集群拓扑感知调度、并行切分策略自动搜索等关键技术，实现计算的自动并行。

整图切分是指遍历整个计算图，将每个算子切分到多设备，相连算子间如果切分策略不同，则会自动插入一个张量（tensor）进行数据分布重排列，用于衔接两个算子间的计算。

由于节点内卡间通信的带宽、时延比节点间通信高，所以集群拓扑感知调度是指将通信量大的数据维度切分到节点内多卡上，将通信量小或者对时延、带宽不敏感的维度切分到节点间。

为了进一步帮助用户更加快速地进行并行网络训练，在前面的基础上，引入了并行切分策略自动搜索的特性，主要围绕昇腾910构建代价模型（cost model），进而计算出不同数据量、不同算子以及在不同切分策略下的计算通信比，然后通过动态规划等算法，自动搜索出在一定内存阈值下性能最优的切分策略。使用这个并行切分策略自动搜索算法，不需要用户指定模型如何切分，MindSpore框架可以实现自动的模型切分。用户只需要一行代码，就能对

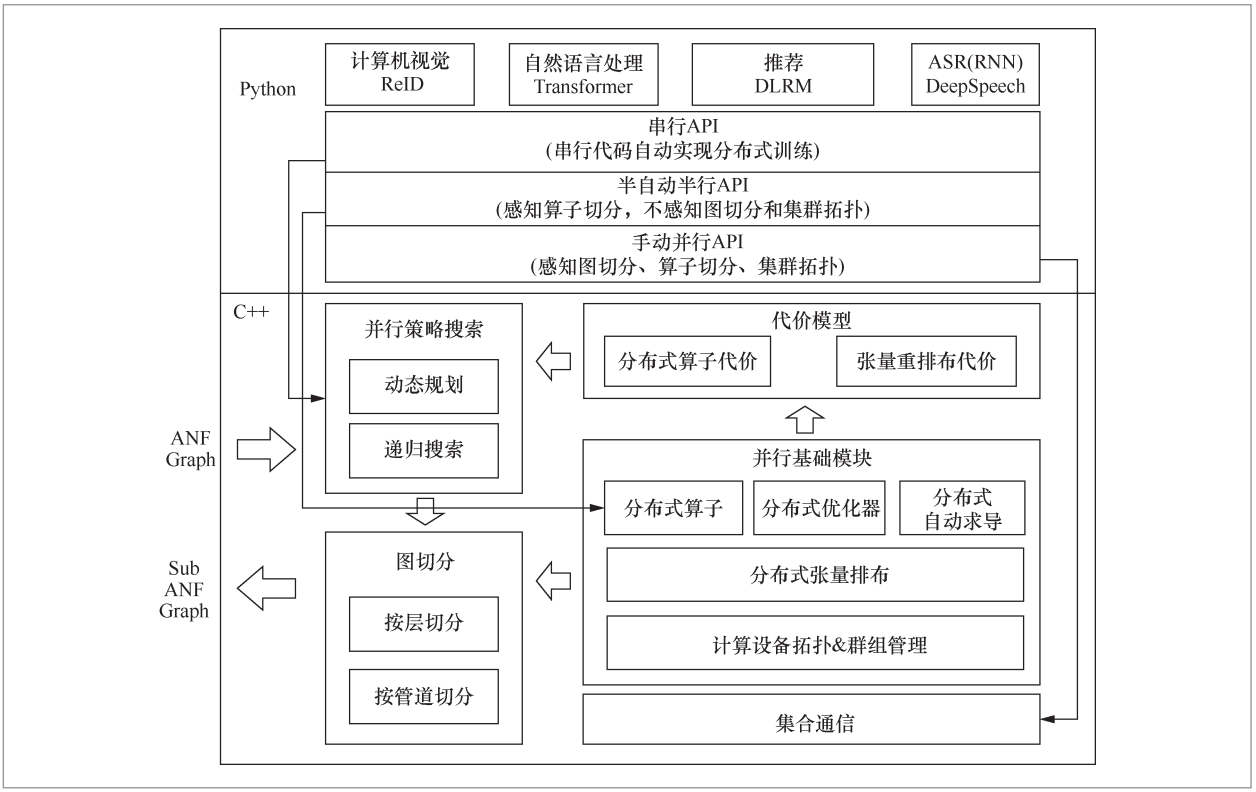

图 2 自动并行示意

一个单机的模型做自动的混合并行训练。

样例代码如下。

```
context.set_auto_parallel_context(parallel_mode=ParallelMode.AUTO_PARALLEL, device_num=dev_num)
```

3.2 自动微分

资深的深度学习开发者都知道，手动微分求解的过程不仅求导过程复杂，而且结果很容易出错，因此现有的深度学习框架都具有自动微分的特性，可以帮助开发者利用自动微分技术实现自动求导，解决这个复杂、关键的问题。

根据实现原理的不同，深度学习框架的自动微分分为以谷歌的TensorFlow为代表的图方法、以脸书的PyTorch为代表的运算符重载以及以MindSpore为代表的源码转化（source to source，S2S）方法[13]，自动微分技术路径如图3所示。

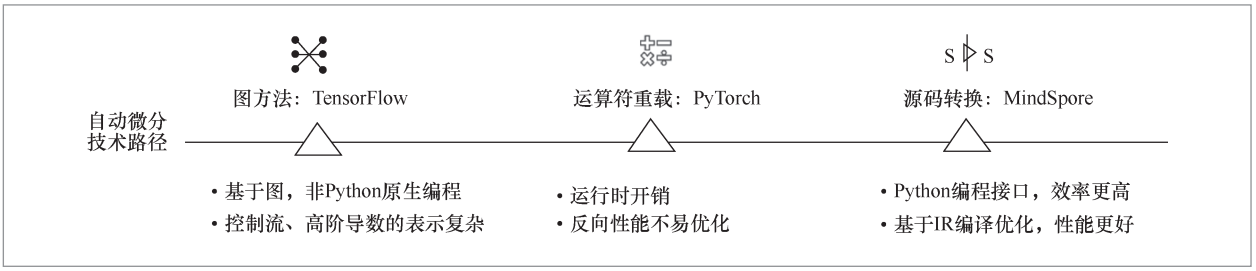

图 3 自动微分技术路径

图方法实现简单，并且图的数据结构容易进行优化和并行。不过图方法的可编程性一直饱受争议，用户需要理解图的概念和接口（如数据节点、通信节点、计算节点、数据边、依赖边、引用边等），存在一定的学习成本。并且，在图方法中控制流、高阶导数的表示较为复杂。

运算符重载比较符合用户尤其是研究人员的编程习惯，很受学术界欢迎。不过这种方式需要使用宿主语言（host language）的解释器，并且在使用过程中需要先运行所有算子，并使用类似磁带（tape）的模式记录运行过程，因此开销比较大，同时这种动态方式也不利于反向性能优化。

MindSpore采用的S2S自动微分技术兼顾了可编程性和性能，一方面它能够与编程语言保持一致的编程体验，另一方面它是IR粒度的可微分技术，可复用现代编译器的优化能力，性能也更好。

S2S自动微分技术使用了高效易调试的可微编程架构。首先在接口层提供Python编程接口，包括控制流表达，有利于用户快速入门。样例代码如下。

```
def cost(x, y): return x * (x + y)

@mindspore
def test_grad (x, y):
return grad_all(cost)(x,y)

def main():
test_grad (2, 1)
```

第一步：用Python代码定义一个计算图（函数）。

第二步：利用MindSpore提供的反向接口进行自动微分，这一步的结果是一个反向的计算图（函数）。

第三步：给定一些输入，就能获取第一步中的计算图（函数）在给定输入处的导数。

在此样例中，自动微分的结果是图中所有输入的导数。MindSpore的反向接口同样提供选项计算某一个或者一部分输入的导数。

其次，IR粒度的可微分技术能够把用户定义的网络源代码通过解析验证等过程，转化为MindSpore定义的IR，也就是MindSpore IR。在IR的基础上应用IR更变器方法（IR mutator method），最终生成反向代码。在此过程中也应用了算子融合等技术，进一步提升了反向性能。

MindSpore对控制流的表达如**图4**所示，包括循环和条件。可以看到代码编程风格与原生Python保持一致，更重要的是MindSpore在生成控制流的反向时不会对循环进行展开，而是在IR基础上进行反向计算，避免了表达式膨胀，从而提升了性能。

相比其他框架，MindSpore可以降低20%的核心代码量，降低了开发门槛，整体提升50%以上的效率。同时，MindSpore天然支持编译优化，进一步提升了代码运行效率，有效降低了科研工程门槛。

MindSpore图层面的自动微分样例代码如下。

```
class Net(Cell):
  def __init__(self):
  self.w = Parameter(Tensor(np.ones([10])))
  def forward(x, y):
  return x + y
  #定义网络
  net = Net()
  x = Tensor(np.ones([10]))
  y = Tensor(np.ones([10]))
  #自动微分推导
  gout = grad_all (net)(x, y)
```

除了图层面的自动微分以外，MindSpore同时支持算子层面的自动微分。在提供深度学习主要网络的算子的同

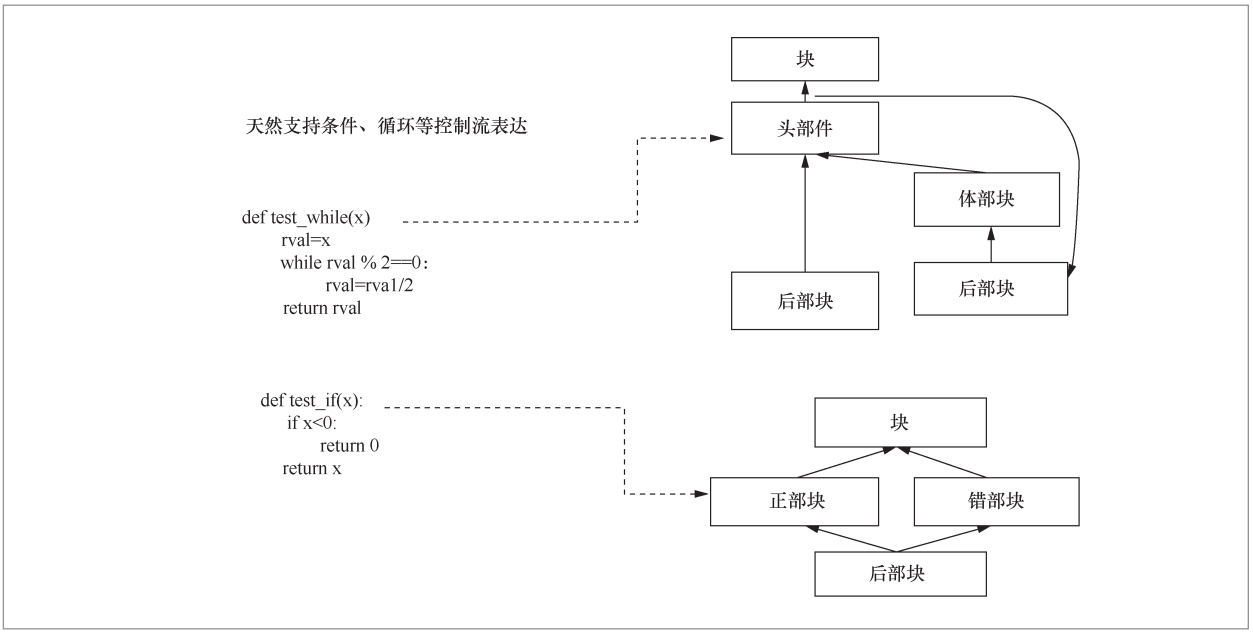

图 4 MindSpore 对控制流的表达

时，MindSpore自带的张量引擎（tensor engine）支持用户使用Python特定领域语言（domain specific language，DSL）自定义算子，并且提供算子级的自动微分接口。通过使用Python DSL，用户可以在Python中自定义算子，如同数学中用公式定义函数一样。而张量引擎的算子自动微分接口可以直接对DSL定义的算子进行微分，正如数学中使用统一的微分符号表示求导一样，这样一来，代码的书写更加简洁直观，更贴近用户的书写习惯。样例代码如下。

```
def sigmoid(x)
    #前向算子的 DSL 实现
    from te.lang.cce import vrec, vadds, vexp, vmuls
    res = vrec(vadds(vexp(vmuls(x, -1.0)), 1.0))
    return res

def sigmoid _ad(dout, x)
    import te
    #前向算子引用
    out = sigmoid(x)
    #前项算子自动微分后生成反向算子
    [dx] = te.differentiate(out, [x], dout)
    return dx
```

这里的前项算子是用户用DSL自定义的算子，也是算子级自动微分的求解目标。接下来利用张量引擎提供的反向接口推导出反向算子。对于多输入的算子来说，反向算子接口可以指定一个或者多个前向算子的输入，然后对这些输入同时进行自动微分计算。另外，与图层面的自动微分不同，算子级的自动微分额外接收反向图中上一层算子（对应正向图的下一层算子）的微分结果作为输入，然后使用链式法则计算出该层反向算子的结果。数学中高阶导数是通过对函数反复使用微分算子计算得到的，同样，在MindSpore中，用户可以通过对算子反复使用反向接口来计算算子的高阶导数。

MindSpore的算子级自动微分接口不仅可以自动生成反向算子，还提供了进一步手动优化导数公式的可能。MindSpore的算子级自动微分功能把算子分割成若干

步简单函数的复合运算后，先是利用已知基础函数的导数和求导法则分步求导，然后利用链式法则计算复合函数的导数，最后使用张量引擎内置的数学公式简化器进行化简。这可以满足绝大部分用户对自动微分的需要。但是对于部分有更高性能要求或者代码部署要求的用户，MindSpore提供接口让用户可以使用自己优化过的导数公式代替某一步或者若干步自动生成的导数公式。样例代码如下。

```
def sigmoid _ad_optimized(dout, x)
    import te
    #前向算子引用
    out = sigmoid(x)
    #手动调优
    def custom_sigmoid_fdiff(out, inputs, grad):
    return [out*(1.0-out)]
    #利用后生成反向算子
    [dx]= te.differentiate(out, [x], dout, override={out: ([x], custom_sigmoid_fdiff)})
    return dx
```

这里把手动推导的导数公式放入函数custom_sigmoid_fdiff(out, inputs, grad)中，并在自动微分中重载求导过程。那么在保持其他部分自动生成的情况下，自动微分使用custom_sigmoid_fdiff(out, inputs, grad)作为输出（out）对x的导数进行运算。这样MindSpore保持了自动微分反向算子和手动调优反向算子在风格上的统一，方便了图层对算子的调用。

总而言之，MindSpore在支持算子级自动微分的同时，对反向算子进行IR层面的优化，满足了算子开发者自动生成反向算子的需求。同时，MindSpore兼顾了用户对手动调优反向的需求，将自动和手动有机结合，简化了开发流程，提高了代码的可读性和运行效率。

3.3 自动调优

现在AI开发的瓶颈之一是AI的黑盒化，特征提取随机化、调试过程不可视、推理结果无解释，这些问题大大限制了AI技术的可信和广泛应用。

调试调优（mindinsight）的目标是打开AI的黑盒，通过可视化Debugger技术，帮助用户轻松地进行模型调试；利用模型训练过程可视化技术、模型溯源可视化技术、性能调优可视化技术，帮助用户轻松地进行性能和精度调优。MindSpore可视化界面如**图5**所示。

调试调优提供了训练看板，开发者在训练看板中可以总览训练过程中的数据抽样情况、参数分布变化情况、损失（loss）变化情况，掌握从训练数据到模型参数再到loss变化的整个模型优化过程。训练看板还集成了计算图可视化和数据图可视化功能。计算图可视化功能有助于用户理解模型结构，判断模型结构是否符合预期。数据图可视化功能有助于用户掌握训练数据的处理流程。

调试调优还提供了模型溯源和数据溯源可视化功能。在模型溯源页面，开发者可以方便地比较不同超参数、不同数据集等条件下的模型效果，选择模型效果更优的超参数。在数据溯源页面，开发者可以同模型溯源联动，精细分析不同的数据处理操作对模型效果的影响，选择更合适的数据处理流程。

而使用动静结合的开发调试模式时，开发者可以只开发一套代码，通过变更一行代码，从容切换动态图/静态图调试方式。需要高频调试时，选择动态图模式，通过单算子/子图，方便灵活地开发调试；需要高效运行时，可以切换为静态图模式，对整张图进行编译，通过高效的图编译优化获得高性能。样例代码如下。

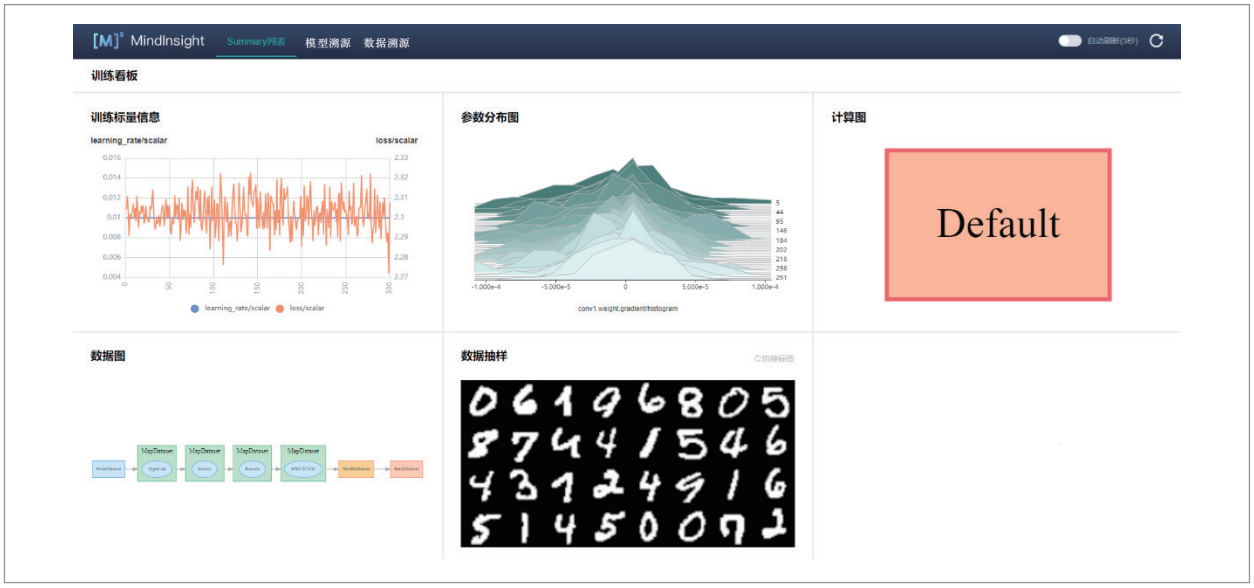

图 5 MindSpore 可视化界面

```
# 定义网络
net = Net()
x = Tensor(np.random.rand(1, 1, 4, 1024).astype(np.float32))

# 默认使用动态图模式
out = net(x)

# 切换静态图模式
context.set_context(mode=context.GRAPH_MODE)
```

综合总体框架和技术优势，可以看出自动微分等特性对MindSpore目标的贡献，见**表2**。

4 MindSpore的性能测试

4.1 性能卓越

MindSpore通过AI Native执行新模式，最大化地发挥了“端-边-云”全场景异构算力。它还协同华为昇腾芯片，通过On-Device执行、深度图优化、高性能数据处理流水线（pipeline）等多维度达到极致性能，帮助开发者缩短训练时间，提升推理性能。MindSpore协同华为昇腾芯片示意图如**图6**所示。

- On-Device执行：整图下沉到

表 2 自动微分等特性对 MindSpore 目标的贡献

目标	自动微分	自动并行	动态图	自动算子生成	图管理器	执行引擎
易开发	很大	很大	很大	很大	一般	一般
高效执行	很大	很大	一般	很大	很大	—
全场景	很大	一般	一般	很大	很大	很大

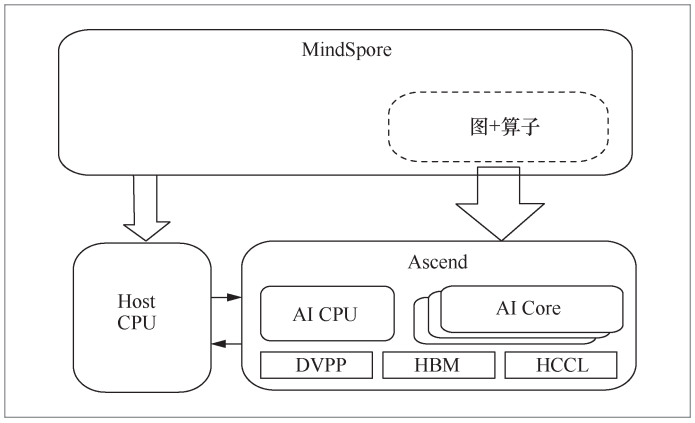

图 6 MindSpore 协同华为昇腾芯片示意

device执行，减少host-device交互开销。

● 深度图优化：包括整图的格式转换消除、类型转换消除、算子融合。

● 高性能数据处理pipeline：支持数据增强、全局混洗（shuffle）。

MindSpore协同华为昇腾芯片的实际训练性能数据见**表3**、**表4**，其中**表3**采用的是ResNet-50 v1.5网络，其网络类型为卷积神经网络，采用的数据集为ImageNet2012，MindSpore版本为0.2.0-alpha；**表4**中采用的是BERT-Large网络，其网络类型为Attention，采用的数据集为zhwiki，MindSpore版本为0.2.0-alpha。

4.2 端云协同

MindSpore针对全场景提供一致的开发和部署能力以及按需协同能力，使开发者能够实现AI应用在云、边缘和手机上的快速部署，全场景互联互通，实现更好的资源利用和隐私保护，创造更加丰富的AI应用。MindSpore端云协同框架如**图7**所示。

MindSpore端云协同框架整合了云侧框架和端侧框架，并打通了自动模型生成、模型压缩、编译优化和端侧学习的全流程。

● MindSpore提供了神经架构搜索（neural architecture search，NAS）[14]能力，用于自动化生成模型，构建模型库。

● MindSpore模型压缩模块用于对模型库中的模型进行剪枝和量化。

● MindSpore提供了编译优化能力，用于转换和优化模型，并通过神经处理单元（neural-network processing

表 3 不同资源条件下 ResNet 网络训练性能数据

对比项	1×Ascend 910，24核CPU	8×Ascend 910，192核CPU	16×Ascend 910，384核CPU
精度	混合	混合	混合
批大小	32	32	32
吞吐量	1 787图/s	13 689图/s	27 090图/s
加速比	—	0.95	0.94

注：以上数据基于华为云AI开发平台ModelArts测试获得，是训练过程整体下沉至Ascend 910 AI处理器执行所得的平均性能。

表 4 不同资源条件下 BERT 网络训练性能数据

对比项	1×Ascend 910，24核CPU	8×Ascend 910，192核CPU
精度	混合	混合
批大小	96	96
吞吐量	210句/s	1 613句/s
加速比	—	0.96

注：以上数据基于华为云AI开发平台ModelArts测试获得，其中网络包含24个隐藏层，句长为128个token，字典表包含21 128个token。

unit，NPU）、图形处理单元（graphics processing unit，GPU）等加速算子执行。

MindSpore端云协同框架具有以下特性。

● 快速多处部署。在实际场景中，模型需要快速适配不同机型硬件。通过神经架构搜索技术构建多元化的模型库，适配多种机型。针对特定应用场景，从模型库中搜索满足性能约束的模型，拿来即用，无须重新训练。

● 全栈性能优化。结合神经架构搜索、模型压缩（剪枝、蒸馏、量化）、编译优化（算子融合、常量折叠、硬件加速）等手段优化模型精度、大小、时延，追求极致性能。

● 灵活并且易用。支持多种策略组合使用，如模型生成、模型压缩和编译优化可以灵活组合；打通云到端全流程，集中管理全流程策略和配置，方便使用。

● 多种学习形态。MindSpore端云协同框架逐步支持多种学习形态，例如支持当前业界常用的端侧推理形态，并逐步支持迁移学习、联合学习等需要端侧训练能力的高级学习形态，满足开发者各种各样的场景需求。

5 结束语

深度学习现已成为人工智能发展重要的方向之一，已经深刻地改变了诸多应用领域，其中广为人知的领域包括自动语音识别、图像识别、自然语言理解以及很多其他交叉领域（如医疗、生物、金融等），并将在越来越多的领域取得成功。本文通过研究人工智能发展历史认识到深度学习的价值和影响力，介绍了新一代深度学习框架MindSpore的框架、技术开发思路和性能优势，希望可以为深度学习技术研究人员提供参考。

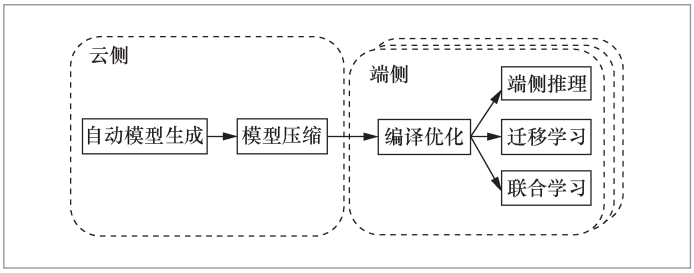

图 7 MindSpore 端云协同框架

致谢

感谢整个华为MindSpore团队的贡献。MindSpore已开源，可在相关网站下载。

参考文献:

[1] SCHALLER R R. Moore's law: past, present and future[J]. IEEE Spectrum, 1997, 34(6): 52-59.

[2] CAMPBELL M, JR A J H, HSU F. DeepDlue[J]. Artificial Intelligence, 2002, 134(1-2): 57-83.

[3] WANG F Y, ZHANG J J, ZHENG X, et al. Where does AlphaGo go: from church-turing thesis to AlphaGo thesis and beyond[J]. IEEE/CAA Journal of Automatica Sinica, 2016, 3(2): 113-120.

[4] KRIZHEVSKY A, SUTSKEVER I, HINTON G E. Imagenet classification with deep convolutional neural networks[J]. Advances in Neural Information Processing Systems. 2012: 1097-1105.

[5] HINTON G, DENG L, YU D, et al. Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups[J]. Signal Processing Magazine IEEE, 2012, 29(6): 82-97.

[6] PAPINENI K, ROUKOS S, WARD T, et al. BLEU: a method for automatic evaluation of machine translation[C]//The 40th Annual Meeting on Association for Computational Linguistics. [S.l.:s.n.], 2002: 311-318.

[7] MNIH V, KAVUKCUOGLU K, SILVER D, et al. Human-level control through deep reinforcement learning[J]. Nature, 2015, 518(7540): 529.
[8] ABADI M, AGARWAL A, BARHAM P, et al. TensorFlow: large-scale machine learning on heterogeneous distributed systems[J]. Computer Science, 2016, arXiv: 1603.04467.
[9] ADAM P, SAM G, SOUMITH C, et al. Automatic differentiation in PyTorch[Z]. 2017.
[10] CHEN T Q, LI M, LI Y T, et al. MXNet: A flexible and efficient machine learning library for heterogeneous distributed systems[J]. Computer Science, 2015, arXiv: 1512.01274.
[11] MA Y J, YU D H, WU T, et al. PaddlePaddle: an open-source deep learning platform from industrial practice[J]. Frontiers of Data and Computing, 2019, 1(1): 105-115.
[12] ADACHI Y, KUMANO T, OGINO K. Intermediate representation for stiff virtual objects[C]//Virtual Reality Annual International Symposium. Piscataway: IEEE Press, 1995: 203-210.
[13] HASCOËT L, NAUMANN U, PASCUAL V. "To be recorded" analysis in reverse-mode automatic differentiation[J]. Future Generation Computer Systems, 2005, 21(8): 1401-1417.
[14] ZOPH B, LE Q V. Neural architecture search with reinforcement learning[J]. Computer Science, 2016, arXiv: 1611.01578.

作者简介

于璠（1980- ），男，博士，华为技术有限公司AI计算框架的创新和生态架构师，曾主导华为云计算资源调度、SDN大规模路由等架构和算法的设计，发表专利30余篇。

收稿日期： 2020-06-15