第1卷 第1期 2019年10月
Vol.1 No.1 Oct.2019

ISSN 2096-742X
CN 10-1649/TP

文献DOI：
10.11871/jfdc.issn.
2096-742X.2019.
01.011

文献PID：
21.86101.2/jfdc.
2096-742X.2019.
01.011

页码：105-115

开放科学标识码
（OSID）

飞桨：源于产业实践的开源深度学习平台

马艳军*，于佃海，吴甜，王海峰

百度公司，北京 100085

摘　要：【目的】深度学习是近年来人工智能取得突破的驱动性核心技术，深度学习框架也被称作智能时代的操作系统，本文对国内唯一功能完备的开源深度学习平台飞桨（PaddlePaddle）进行了系统性介绍。【方法】首先介绍深度学习框架的发展历程，并概述飞桨深度学习平台的技术全景和生态建设进展，然后详细介绍飞桨核心框架的关键技术，包括前端语言、组网编程范式、核心架构图、算子库以及高效率计算核心五个部分。【结果】飞桨经过多年来产业实践中持续迭代创新，已经在超大规模分布式训练、多端高速推理等方面形成了独特的优势。【结论】系统性总结飞桨的主要创新点并对未来发展趋势进行展望。

关键词：飞桨；人工智能；深度学习；深度学习框架

PaddlePaddle: An Open-Source Deep Learning Platform from Industrial Practice

Ma Yanjun*，**Yu Dianhai**，**Wu Tian**，**Wang Haifeng**

Baidu Inc.,Beijing 100085,China

Abstract: **[Objective]** Deep learning is widely recognized as core technology driving the breakthroughs in artificial intelligence. Deep learning frameworks can be considered as the operating system in the era of artificial intelligence. PaddlePaddle, as the only fully-functioning open-source deep learning platform in China, is introduced comprehensively. **[Methods]** In this paper, a brief history of the deep learning frameworks is introduced, followed by an overview of PaddlePaddle, which is comprised of the core framework, toolkits and service platforms. After that, we elaborate on the core technologies of PaddlePaddle, including the front-end programming language, the modeling paradigm etc. Finally, the main innovations in PaddlePaddle are summarized. **[Results]** PaddlePaddle has been intensively tested in Baidu production for years, with unique features in supporting distributed training with ultra-large data and fast inference on server, mobile as well as edges. **[Conclusions]** The main innovations, research and development trends are discussed systematically.

Keywords: PaddlePaddle; artificial intelligence; deep learning; deep learning framework

* 通讯作者（E-mail: mayanjun02@baidu.com）

引言

深度学习已成为最有影响的人工智能关键共性技术。人们在使用深度学习技术进行科学实验或产品研发时，通常都是基于某个深度学习框架开展工作。深度学习框架处于硬件层和应用层之间，不仅支撑人工智能应用的开发，对人工智能芯片的设计也有很大影响，是智能时代的操作系统。因此，业界普遍认为，深度学习框架与人工智能芯片，构成人工智能领域最为核心的两项“基础设施”。目前，基于深度学习的人工智能的应用场景正在极大丰富，并内置到终端和云端，让人工智能技术更快速地普及到各行各业，推动融合创新。在全球人工智能争夺战中，各国都企图占领深度学习框架这一制高点，进而全面把控人工智能生态链，形成新的垄断。因此，发展我国自主的深度学习框架，建立人工智能关键共性技术体系，对于捍卫我国科技安全有重要的战略意义。

深度学习框架最初主要用于深度学习的科研工作，代表性开源深度学习框架包括 Theano[1]（2010，蒙特利尔大学）、Caffe[2]（2013，加州大学伯克利分校）等。近年来，国外的大型科技公司开始纷纷布局这一领域并开源，影响力比较大的包括谷歌的 TensorFlow[3]（2015）和脸书的 PyTorch[4]（2017）。飞桨[5]（PaddlePaddle）是国内首个也是当前唯一一个功能完备的开源深度学习平台，基于百度在深度学习领域的长期积累自主研发，并在 2016 年开源。

本文的组织结构如下：引言部分主要介绍深度学习框架意义和国内外现状；第一节介绍飞桨深度学习平台的技术全景和生态建设进展；第二节详解飞桨核心框架的关键技术；第三节总结飞桨深度学习平台的主要创新点；最后进行总结和展望。

1　飞桨深度学习平台简介

1.1　技术全景

百度从 2012 年开始研发具备完全自主知识产权的深度学习框架飞桨，并建成了国内唯一的集深度学习训练和预测框架、模型库、工具组件等为一体的开源深度学习平台。飞桨的全景图如图 1 所示。

1.1.1 核心框架

核心框架是飞桨平台的基础和核心，主要包括工业级的深度学习训练和预测框架。目前深度学习框架主要有动态图和静态图两种编程方式。静态图属于符号化编程，代码执行的时候不会返回运算的结果，需要事先定义神经网络的结构，然后执行整个图结构；而动态图属于命令式编程，代码能够直接返回运算的结果。通常来说，静态图能够对编译器做最大的优化，更有利于性能的提升，而动态图则非常便于用户对程序进行调试。

飞桨的训练框架源自百度海量规模的业务场景实践，提供了大规模分布式训练和工业级数据处理的能力，同时支持稠密参数和稀疏参数场景的超大规模深度学习并行训练，支持万亿规模参数、数百个节点的高效并行训练。面对实际业务上数据实时变化或者膨胀的特点，对模型做流式更新，可提供实时更新的能力。

对于模型的部署，飞桨提供了针对服务器端在线服务（Paddle Serving）和移动端部署库（Paddle Lite）。Paddle Serving 是在线预测部分，可以与模型训练环节无缝衔接，提供深度学习的预测云服务。开发者也可以对预测库编译或直接下载编译好的预测库后，调用预测应用程序编程接口（Application Programming Interface，API），对模型进行 C++ 预测。在移动端，Paddle Lite 支持将模型部署到 ARM CPU（Advanced Reduced instruction set computing Machine Central Processing Unit），Mali GPU（Graphics Processing Unit），Adreno GPU，树莓派等各种硬件以及安卓、苹果、Linux 等软件系统之上。

飞桨开源了计算机视觉、自然语言处理、推荐和语音四大类 70 多个官方模型，覆盖了各领域的主流和前沿模型。其中飞桨视觉模型库（PaddleCV）提供了图像分类、目标检测、图像分割、关键点检

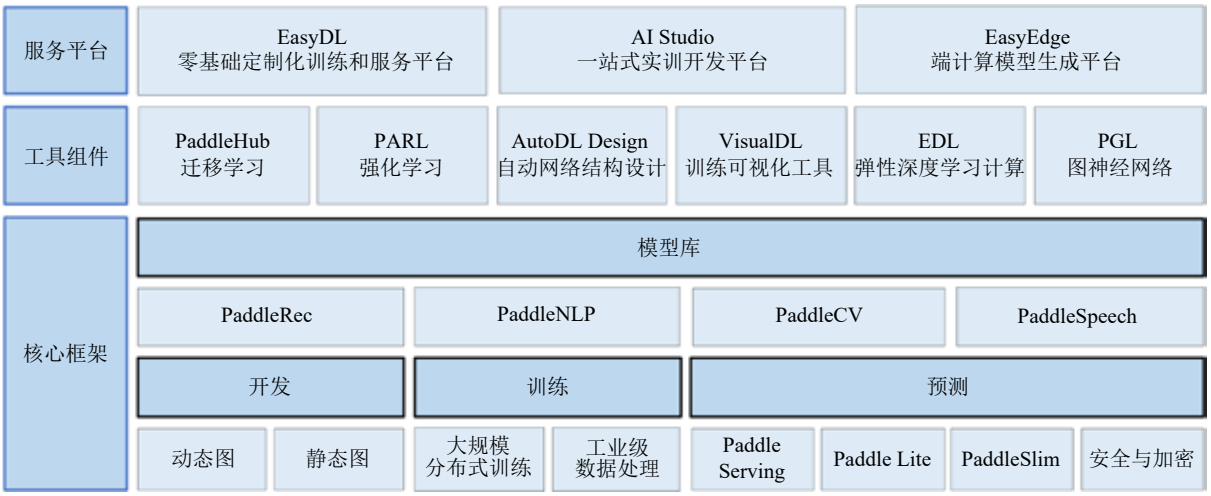

图 1　飞桨全景图

Fig. 1 PaddlePaddle overview

测、图像生成、文字识别、度量学习和视频分类与动作定位等多类视觉算法模型。飞桨自然语言处理模型库（PaddleNLP）包括了词法分析、语言模型、语义表示和文本生成等多项自然语言处理任务的算法模型实现，包括百度自研的增强语义表示模型 ERNIE[6]（Enhanced Representation from kNowledge IntEgration）。ERNIE 通过建模海量数据中的词、实体及实体关系，直接对先验语义知识单元进行建模。与谷歌语义表示模型 BERT[7]（Bidirectional Encoder Representation from Transformers）相比，BERT 仅学习原始语言信号，ERNIE 增强了模型语义表示能力。

1.1.2 工具组件

飞桨开源了迁移学习（PaddleHub）、强化学习（PARL）、自动化网络结构设计（AutoDL Design）、训练可视化（VisualDL）、弹性深度学习（EDL）和图神经网络（PGL）等一系列深度学习的工具组件，用于配合深度学习相关技术的开发、训练、部署和应用。PaddleHub 是飞桨预训练模型管理和迁移学习工具，便于用户将预训练模型更好地适配到自己的应用场景。PARL 是飞桨深度强化学习工具，曾获得 NeurIPS 2018 假肢挑战赛冠军。PARL 不仅提供了 DQN[8]、DDPG[9]、PPO[10] 等主流强化学习模型，还提供了 IMPALA[11]、GA3C[12]、A2C[13] 等并行算法，以及相应的并行开发接口。AutoDL Design 是飞桨自动化网络设计工具，基于强化学习实现，在 CIFAR10 数据上准确率超过 98%。VisualDL 是一个面向深度学习任务的可视化工具，原生支持 Python 的使用，只需在模型中增加少量的代码，对 VisualDL 接口进行调用，便可以为训练过程提供丰富的可视化支持。EDL 通过与 Kubernetes 协作来实现弹性作业调度，能够支持飞桨在人工智能（Artificial Intelligence，AI）云上进行弹性调度。PGL 是飞桨图神经网络工具，提供 Python 接口用于存储 / 读取 / 查询图数据结构，并且提供基于游走（Walk-based）以及消息传递（Message Passing）两种计算范式的计算接口。利用这些接口，可以搭建不同的图学习算法。

1.1.3 服务平台

飞桨提供了包含 EasyDL、AI Studio 和 EasyEdge 的服务平台，可以满足不同层次开发者的深度学习开发需求。EasyDL 是定制化训练和服务平台，为企业用户和开发者提供高精度的 AI 模型定制服务，已在零售、工业、安防、医疗、互联网、物流等 20 多个行业中落地应用。AI Studio 是针对学习者的在线一体化开发实训平台，集合了 AI 教程、深度学习样例工程、各领域的经典数据集、云端的运算及存储资源、以及比赛平台和社区，可以支撑高

校老师进行 AI 教学。EasyEdge 是基于 Paddle Lite 研发的端计算模型生成平台，能够帮助深度学习开发者将自建模型快速部署到设备端。开发者只需上传模型，最快 2 分种即可生成端计算模型并获取软件开发工具包（Software Development Kit，SDK）。

1.2 飞桨生态建设

飞桨为企业开发者提供不同层次的培养体系，帮助开发者成长，助力企业 AI 创新。具体包括黄埔学院、AI 快车道、PaddleCamp 等。其中，黄埔学院致力于为中国产业界培养第一批首席 AI 架构师；AI 快车道侧重应用实战，为 1000 家企业提供深度学习技术应用落地支持；PaddleCamp 面向深度学习初学者进行入门培训。除了以上培训，飞桨还持续提供丰富的 AI 开发者竞赛，构建开发者生态体系。此外，飞桨还通过深度学习师资培训班、在线教育平台和深度学习工程师能力评估标准等助力高校与教育机构快速搭建完整的 AI 教学体系。

目前，飞桨在开源社区中的活跃度较高，已经覆盖超过 16000 家企业和 130 万名开发者。随着在开源社区的影响力逐步增强，包括英特尔、英伟达、华为、寒武纪、比特大陆等诸多芯片厂商也纷纷支持飞桨，并基于此开展合作。后续飞桨还需要在如下几个方面进一步完善：核心框架的高层 API 目前还比较缺乏，易用性有待进一步提升；基于飞桨的学术论文还不够多，生态还需要持续加大力度建设等。

2 飞桨深度学习框架核心技术

飞桨深度学习框架核心技术，主要包括前端语言、组网编程范式、核心架构、算子库以及高效率计算核心五部分。

2.1 前端语言

为了方便用户使用，选择 Python 作为模型开发和执行调用的主要前端语言，并提供了丰富的编程接口 API。Python 作为一种解释型编程语言，代码修改不需要重新编译就可以直接运行，使用和调试非常方便，并且拥有丰富的第三方库和语法糖，拥有众多的用户群体。同时为了保证框架的执行效率，飞桨底层实现采用 C++。对于预测推理，为方便部署应用，则同时提供了 C++ 和 Java API。

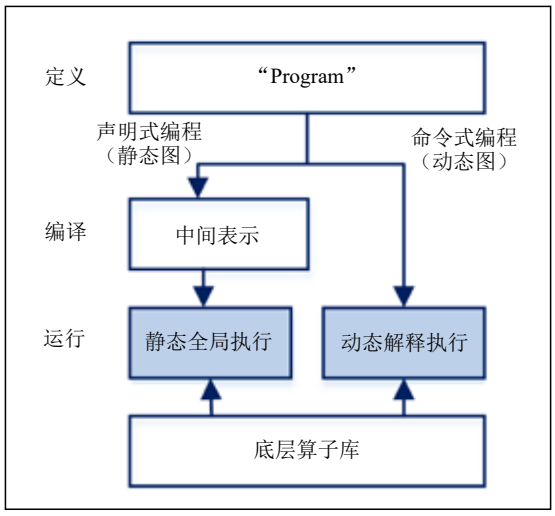

图 2 飞桨的组网编程范式和执行逻辑

Fig. 2 PaddlePaddle programming paradigm and execution logic

2.2 组网编程范式

在介绍飞桨的组网方式之前，首先介绍两种编程范式，声明式编程（也叫符号式编程）和命令式编程。命令式编程是一种非常直观的编程方式，程序的定义对应着执行，可立即返回结果，大部分 Python 的程序都是命令式编程，比如运行“x=10; y=x+2;”这段代码，y 的值能够立马获取，这种编程方式非常方便调试。而声明式编程，程序在定义的时候并不会立即执行。对于深度学习模型开发来讲，相当于先是构建一个完整的网络结构，例如表示成一张图，然后再统一执行前向反向计算。由于在执行前可以进行整体优化，因此这种方式往往可以做到执行效率更高。但是由于执行和定义的分离，所以不太方便测试。

在飞桨中同时兼容这两种编程范式，如图 2 所示。飞桨以程序化“Program”的形式动态描述神经网络模型计算过程，并提供对顺序、分支和循环三种执行结构的支持，可以组合描述任意复杂的模型，并可在内部自动转化为中间表示的描述语言。“Program”的定义过程就像在写一段通用程序。使用声明式编程时，相当于将“Program”先编译再执行，可类比静态图模式。首先根据网络定义代码构造“Program”，然后将“Program”编译优化，最后通过执行器执行“Program”，具备高效性能；同时由于存在静态的网络结构信息，能够方便地完成模型的部署上线。而命令式编程，相当于将“Program”解释执行，可视为动态图模式，更加符合用户的编程习惯，代码编写和调试也更加方便。飞桨后面会增强静态图模式下的调试功能，方便开发调试；同时提升动态图模式的运行效率，加强动态图自动转静态图的能力，快速完成部署上线；同时更加完善接口的设计和功能，整体提升框架易用性。

2.3 核心架构

飞桨核心架构采用分层设计，如图 3 所示。前端应用层考虑灵活性，采用 Python 实现，包括了组网 API、IO API、Optimizer API 和执行 API 等完备的开发接口；框架底层充分考虑性能，采用 C++ 来实现。框架内核部分，主要包含执行器、存储管理和中间表达优化；内部表示方面，包含网络表示（ProgramDesc）、数据表示（Variable）和计算表示（Operator）几个层面。框架向下对接各种芯片架构，可以支持深度学习模型在不同异构设备上的高效运行。

2.3.1 执行器

飞桨神经网络结构是由一系列的 Operator 和 Variable 构成。用户将定义好的“Program”传入 Executor，Executor 根据“Program”中的 Operator 的顺序依次调用运行。飞桨还提供了 Executor 的一个升级版 Parallel Executor，它的执行对象是编译优化过的“Program”（Compiled Program），相当于优化过的计算图，比如显存优化、Operator 合并等等。但 Parallel Executor 对用户是透明的，统一使用 Executor 接口来执行，区别在于传入的是“Program”还是经过了编译优化的“Compiled Program”。相比于 Executor，Parallel Executor 主要有两个改进的地方：减少了每次迭代过程中的同步操作；支持并发执行 Operator。

（1）减少同步的次数

在使用 Parallel Executor 执行图像处理单元 GPU 多卡训练时，会根据“Program”和卡数构建出一个静态单流赋值图（Static Single Assignment Graph，SSA Graph），在迭代过程中多卡之间只在聚合参数梯度的时候才会有同步操作。

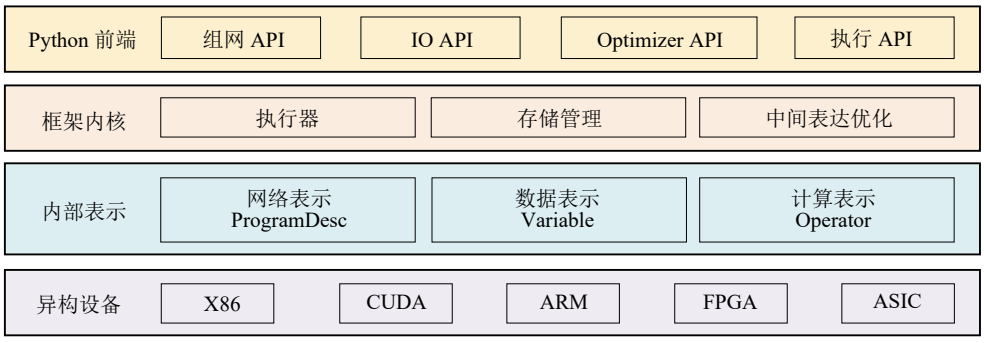

图 3 核心架构图

Fig 3 The core architecture

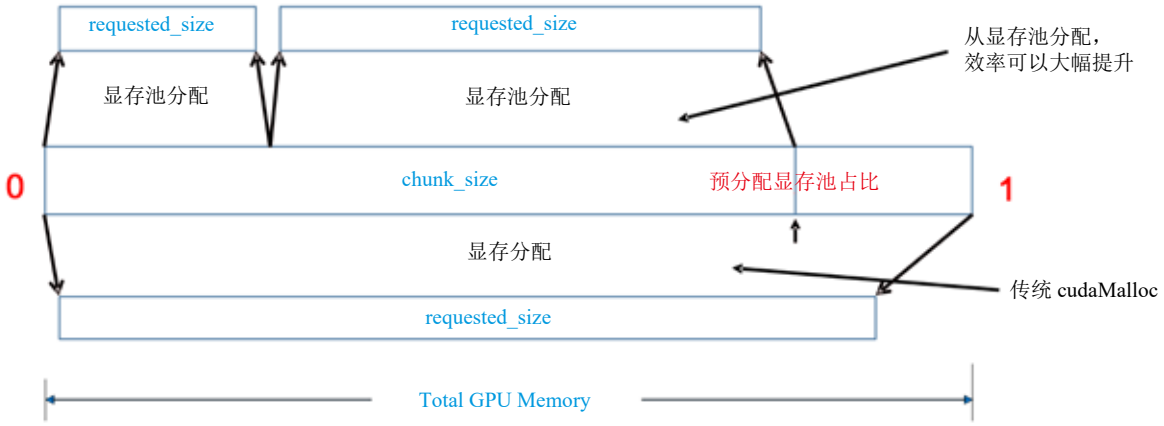

图 4 显存分配机制

Fig. 4 The strategy of GPU memory allocation

（2）支持并发执行 Operator

Parallel Executor 在执行过程中根据输入变量是否就绪来决定是否执行 Operator，通过一个线程池来管理调度。所以 Parallel Executor 中运行 Operator 的顺序并不是固定的，可以通过并发执行提升效率。

2.3.2 显存管理

飞桨为用户提供技术领先、简单易用、兼顾显存回收与复用的显存优化策略，在很多模型上的表现优于其他开源框架。

（1）显存分配机制

原生的 CUDA 系统调用（cudaMalloc）和释放（cudaFree）均是同步操作，非常耗时。为了加速显存分配，飞桨采用了显存预分配的策略，具体方式如图 4 所示。设置一个显存池 chunk，定义其大小为 chunk_size。若分配需求 requested_size 不超过 chunk_size，则框架会预先分配 chunk_size 大小的显存池 chunk，并从中分出 requested_size 大小的块返回。之后每次申请显存都会从 chunk 中分配。若 requested_size 大于 chunk_size，则框架会调用 cudaMalloc 分配 requested_size 大小的显存。chunk_size 一般依据初始可用显存大小按比例确定。同时飞桨也支持按实际显存占用大小的动态自增长的显存分配方式。

（2）显存垃圾及时回收机制

显存垃圾及时回收机制 GC（Garbage Collection）的原理是在网络运行阶段释放无用变量的显存空间，达到节省显存的目的。GC 策略会积攒一定大小的显存垃圾后再统一释放。GC 内部会根据变量占用的显存大小，对变量进行降序排列，且仅回收前面满足占用大小阈值以上的变量显存。GC 策略默认生效于使用 Executor 或 Parallel Executor 做模型训练预测时。

（3）Operator 内部显存复用机制

Operator 内部显存复用机制（Inplace）的原理是 Operator 的输出复用 Operator 输入的显存空间。例如，数据整形（reshape）操作的输出和输入可复用同一片显存空间。Inplace 策略可通过构建策略（BuildStrategy）设置生效于 Parallel Executor 的执行过程中。

2.4 算子库

飞桨算子库目前提供了 500 余个算子，并在持续增加，能够有效支持自然语言处理、计算机视觉、

语音等各个方向模型的快速构建。同时提供了高质量的中英文文档，更方便国内外开发者学习使用。文档中对每个算子都进行了详细描述，包括原理介绍、计算公式、论文出处，详细的参数说明和完整的代码调用示例。

飞桨的算子库覆盖了深度学习相关的广泛的计算单元类型。比如提供了多种循环神经网络（Recurrent Neural Network，RNN），多种卷积神经网络（Convolutional Neural Networks，CNN）及相关操作，如深度可分离卷积（Depthwise Deparable Convolution）、空洞卷积（Dilated Convolution）、可变形卷积（Deformable Convolution）、池化兴趣区域池化及其各种扩展、分组归一化、多设备同步的批归一化，另外涵盖多种损失函数和数值优化算法，可以很好地支持自然语言处理的语言模型、阅读理解、对话模型、视觉的分类、检测、分割、生成、光学字符识别（Optical Character Recognition，OCR）、OCR 检测、姿态估计、度量学习、人脸识别、人脸检测等各类模型。

飞桨的算子库除了在数量上进行扩充之外，还在功能性、易用性、便捷开发上持续增强：例如针对图像生成任务，支持生成算法中的梯度惩罚功能，即支持算子的二次反向能力；而对于复杂网络的搭建，将会提供更高级的模块化算子，使模型构建更加简单的同时也能获得更好的性能；对于创新型网络结构的需求，将会进一步简化算子的自定义实现方式，支持 Python 算子实现，对性能要求高的算子提供更方便的、与框架解耦的 C++ 实现方式，可使得开发者快速实现自定义的算子，验证算法。

2.5 高效率计算核心

飞桨对核心计算的优化，主要体现在以下两个层面。

2.5.1 Operator 粒度层面

飞桨提供了大量不同粒度的 Operator（Op）实现。细粒度的 Op 能够提供更好的灵活性，而粗粒度的 Op 则能提供更好的计算性能。飞桨提供了诸如 softmax_with_cross_entropy 等组合功能 Op，也提供了像 fusion_conv_inception、fused_elemwise_activation 等融合类 Operator。其中大部分普通 Op，用户可以直接通过 Python API 配置使用，而很多融合的 Op，执行器在计算图优化的时候将会自动进行子图匹配和替换。

2.5.2 核函数实现层面

飞桨主要通过两种方式来实现对不同硬件的支持：人工调优的核函数实现和集成供应商优化库。

（1）针对 CPU 平台

飞桨一方面提供了使用指令 Intrinsic 函数和借助于 xbyak[14] JIT 汇编器实现的原生 Operator，深入挖掘编译时和运行时性能；另一方面，飞桨通过引入 OpenBLAS[15]、Intel® MKL[16]、Intel® MKL-DNN[17] 和 nGraph[18]，对 Intel CXL 等新型芯片提供了性能保证。

（2）针对 GPU 平台

飞桨既为大部分 Operator 用 CUDA C 实现了经过人工精心优化的核函数，也集成了 cuBLAS[19]、cuDNN[20] 等供应商库的新接口、新特性。

表 1 中给出了 Paddle Fluid 1.5.0 使用单个 GPU 训练的性能评测数据。在 Paddle Fluid 1.5.0 的静态图基准测试中，相对于开源对标框架，Mask-RCNN、YOLOv3、DeepLab V3+、CycleGAN、BERT 这 5 个模型的速度表现出显著的优势，SE-ResNeXt 和 Transformer 这 2 个模型也达到与对标框架持平。在未来的工作中，飞桨将持续致力于计算的优化，除了针对更多不同硬件平台特性，开发更多高效率计算的算子外，飞桨也在结合编译器优化技术和自动代码生成技术，研发一套通用、自动的优化方案，使更多的模型能够获得性能提升。

表 1 不同框架训练性能比较

Table 1 The training performance comparison with other frameworks

领域	模型名称	对标开源框架	Paddle Fluid 1.5.0	对标开源框架
智能视觉（CV）	SE-ResNeXt50	PyTorch 1.1.0	168.33 images/s	163.13 images/s
	Mask-RCNN	PyTorch 1.1.0	3.81 images/s	3.24 images/s
	YOLOv3	MXNet 1.4.1	29.90 images/s	18.58 images/s
	DeepLab V3+	TensorFlow 1.12.0	13.70 images/s	6.40 images/s
	CycleGAN	TensorFlow 1.12.0	7.51 images/s	6.45 images/s
智能文本处理（NLP）	BERT	TensorFlow 1.12.0	129.41 sentences/s	109.44 sentences/s
	Transformer	TensorFlow 1.12.0	19927 tokens/s	19456 tokens/s

注：测试平台使用 Tesla V100-SXM2 GPU 单卡，NVIDIA 驱动版本为 418.39，软件版本为 CUDA 9.0、cuDNN 7.4。同样模型使用一致的 batchsize 等训练配置。

3 创新性介绍

3.1 基于 Program 的神经网络模型定义表示

不同于其他大部分深度学习框架，飞桨在神经网络模型定义和描述时，去掉计算图的概念，代之以 Program 的形式动态描述计算过程。这种程序化的计算描述方式，兼具网络结构修改的灵活性和模型搭建的便捷性，极大地提高了框架对模型的表达能力的同时也保证了高性能。

3.2 基于变长张量表示的计算加速

在神经网络计算中，矩阵维度足够大，才能最大化对计算资源的利用，充分发挥 GPU 或是其他异构计算设备的计算能力，从而加速运算。在飞桨的设计中，通过对序列进行排序，然后沿时间步维度切取出多个 Batch 并行计算，飞桨对序列模型没有增加计算量，也无需用户处理填充 / 去填充相关的细节，从而提升变长序列场景的计算效率。

3.3 基于训练推理一体化的多端推理引擎

飞桨框架同时充分考虑训练和预测部署问题，一体化设计支持多端多平台的推理引擎，在高性能、轻量化和普适性方面开展重点优化。基于此设计，在多个硬件平台多种模型的对照评估中，飞桨预测框架显示出了领先的性能优势。

3.4 多机多卡扩展性组件智能组合技术

针对大规模数据场景，GPU 多机多卡训练的扩展性至关重要。飞桨研发了一整套完备的扩展性组件包，包括多种通信拓扑的支持、梯度智能聚合、通信与计算节点依赖分析、多流并发通信等。同时，在不同模型结构下，通过对扩展性组件包中的组件进行智能组合，实现对用户透明的最佳并行扩展。

3.5 超大规模稀疏参数服务器技术

飞桨针对大规模点击率预估的场景，创新性地提出了大规模稀疏参数服务器技术。设计方面，将训练部分与通信部分、参数存储进行了分离，可以对通信和存储更新做针对性的并行优化，最大化地提升速度。通信方面，通过稠密参数共享，多线程训练可以仅用一个线程去拉取参数，减少一半的通信量，从而大幅提升了训练速度。参数存储方面，

采用了存和取解耦的模式，具备了良好的扩展性。

3.6 低配网络环境稀疏通信并行训练算法

飞桨还针对特殊场景研发内建（Built-in）并行能力。通过不断地累计本地梯度，并同步最有代表性的少量梯度，在保证模型收敛的前提下可以将通信量减小为原始通信量的 1% 以下，大大降低了网络通信负载。在带宽压缩到 1Gb/s 的情况下，通用的多机多卡并行训练方法在吞吐方面已经趋近于 0，而基于稀疏通信的并行训练方法依然可以保持较高的吞吐。

3.7 局部拉德马赫正则化方法

为提升飞桨模型的泛化性能，提出局部拉德马赫正则化方法。该方法采用最近的拉德马赫复杂度的估计方法，对折页损失函数（Hinge Loss）和交叉熵（Cross Entropy）推得了固定值的表达式，并且将其称之为局部拉德马赫正则化项，加在经验损失函数上。实验证明该正则化方法作用在模型集成之后，得到了 CIFAR-10 上目前最好的准确率。

3.8 知识增强语义表示学习方法

飞桨知识增强的语义表示模型 ERNIE，是飞桨源于产业实践的典型成果。通过建模海量数据中的实体概念等先验语义知识，学习真实世界的语义关系。最新发布的 ERNIE 2.0[21] 模型，在共计 16 个中英文任务上超越了 BERT 和 XLNet[22]，取得了 SOTA 效果。

4 发展趋势展望

4.1 深度学习门槛进一步降低

深度学习框架的开源已经大幅降低了应用门槛。未来，随着应用越来越广泛，深度学习技术的门槛还会进一步降低。对于有一定的深度学习经验的开发者，可以轻松结合预训练模型开展迁移学习工作，利用 Fine-tune API 并结合少量代码即可完成大规模预训练模型的迁移学习。例如，飞桨迁移学习工具 PaddleHub 引入“模型即软件”的设计理念，支持通过 Python API 或者命令行工具，一键完成基于预训练模型的迁移学习和快速部署。对于缺乏 AI 算法基础的开发者，EasyDL 的定制化训练和服务平台，可以提供高精度的 AI 模型定制服务，实现快速行业应用。

4.2 软硬一体充分释放硬件性能

随着 AI 应用场景的丰富和异构硬件技术的快速发展，深度学习的算法模型将会运行在多种多样的终端硬件上。因此，深度学习框架如何与硬件联合优化、充分释放硬件的计算性能，将会是一个重要的发展方向。飞桨将会结合底层编译优化的思想，对计算图的中间表示在各种硬件上的适配做更深入的优化。

4.3 无处不在的连接，安全的 AI 计算

随着 5G 技术的快速发展，传输能力将不再是瓶颈，终端—边缘—云端将会实现无处不在的连接。但这对深度学习框架的架构设计提出了新的更高的要求，也必将对数据的安全和隐私提出更高的要求。因此如何让 AI 计算更加安全，也将是飞桨的一个重要发展方向。

利益冲突声明

所有作者声明不存在利益冲突关系。

参考文献

[1] Bergstra J, Breuleux O, Bastien F, et al. Theano: a CPU and GPU math expression compiler[C]. Proceedings of the Python for scientific computing conference (SciPy). 2010, 4(3).

[2] Jia Y, Shelhamer E, Donahue J, et al. Caffe: Convolutional

architecture for fast feature embedding[C]. Proceedings of the 22nd ACM international conference on Multimedia. ACM, 2014: 675-678.

[3] Abadi M, Barham P, Chen J, et al. Tensorflow: A system for large-scale machine learning[C]. 12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16). 2016: 265-283.

[4] Paszke A, Gross S, Chintala S, et al. Pytorch: Tensors and dynamic neural networks in python with strong gpu acceleration[J]. PyTorch: Tensors and dynamic neural networks in Python with strong GPU acceleration, 2017, 6.

[5] 飞桨: 源于产业实践的开源深度学习平台[EB/OL]. https://www.paddlepaddle.org.cn/.

[6] Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. Ernie: Enhanced representation through knowledge integration.. arXiv:1904.09223, 2019.

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805, 2018.

[8] Mnih V, Kavukcuoglu K, Silver D, et al. Human-level control through deep reinforcement learning[J]. Nature, 2015, 518(7540): 529.

[9] Lillicrap T P, Hunt J J, Pritzel A, et al. Continuous control with deep reinforcement learning[J]. arXiv preprint arXiv:1509.02971, 2015.

[10] Schulman J, Wolski F, Dhariwal P, et al. Proximal policy optimization algorithms[J]. arXiv preprint arXiv:1707.06347, 2017.

[11] Espeholt L, Soyer H, Munos R, et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures[J]. arXiv preprint arXiv:1802.01561, 2018.

[12] Babaeizadeh M, Frosio I, Tyree S, et al. GA3C: GPU-based A3C for deep reinforcement learning[J]. CoRR abs/1611.06256, 2016.

[13] OpenAI Baselines: ACKTR & A2C[EB/OL]. https://openai.com/blog/baselines-acktr-a2c/.

[14] XBYAK – X86, X64 JIT ASSEMBLER[EB/OL]. http://herumi.in.coocan.jp/soft/xbyak_e.html.

[15] OpenBLAS: An optimized BLAS library[EB/OL]. https://www.openblas.net/.

[16] Intel(R) Math Kernel Librray (Intel(R) MKL) [EB/OL]. https://software.intel.com/en-us/mkl.

[17] Intel(R) Math Kernel Library for Deep Neural Networks (Intel(R) MKL-DNN) [EB/OL]. https://github.com/intel/mkl-dnn.

[18] nGraph Compiler stack (Beta) [EB/OL]. https://github.com/NervanaSystems/ngraph.

[19] NVIDIA cuBLAS, Dense Library Algebra on GPUs[EB/OL]. https://developer.nvidia.com/cublas.

[20] NVIDIA CUDA(R) Deep Neural Network library (cuDNN) [EB/OL]. https://developer.nvidia.com/cudnn.

[21] Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Hao Tian, Hua Wu, Haifeng Wang. ERNIE 2.0: A Continual Pre-training Framework for Language Understanding. arXiv preprint arXiv:1907.12412v1, 2019.

[22] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, Quoc V. Le. XLNet: Generalized Autoregressive Pretraining for Language Understanding. arXiv preprint arXiv:1906.08237, 2019.

收稿日期：2019年8月15日

马艳军，1981年生，博士，百度深度学习技术平台部总监，负责深度学习平台飞桨（PaddlePaddle）的产品和研发工作，曾在ACL等自然语言处理顶级会议期刊发表论文数20余篇，并多次担任国际会议的Area

Chair等。2015年，相关成果曾获国家科技进步二等奖。

马艳军参与文章整体框架设计并完成论文写作。于佃海承担了文章第二部分的设计与写作。吴甜参与了文章第一和第四部分的设计和写作。王海峰对文章整体框架进行了设计，并对核心内容进行了修订。

Dr. Ma Yanjun is a director of deep learning platform at Baidu, overseeing the development of deep learning framework PaddlePaddle. He was born in 1981 in China. He has authored and co-authored over 20 research publications in Natural Language Processing, and served as area co-chairs for a number of top international conferences. In 2015, He received National Technology Advancement Award.

As to this paper, he contributed to the organization of the paper and wrote the manuscript. Yu Dianhai contributed to the design and writing of Section 2. Wu Tian contributed to Section 1 and 4.Wang Haifeng oversaw the overall design and revised the main contents.

E-mail: mayanjun02@baidu.com

引文格式：马艳军,于佃海,吴甜,王海峰.飞桨:源于产业实践的开源深度学习平台[J].数据与计算发展前沿,2019,1(1):105-115.DOI:10.11871/jfdc.issn.2096-742X.2019.01.011.PID: 21.86101.2/jfdc.2096-742X.2019.01.011.