Computer Science and Application 计算机科学与应用, 2023, 13(9), 1748-1755
Published Online September 2023 in Hans. https://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2023.139173

Hans汉斯

训推一体平台架构设计与关键技术研究

梁秉豪，张传刚

浪潮通信信息系统有限公司，山东 济南

收稿日期：2023年8月21日；录用日期：2023年9月19日；发布日期：2023年9月26日

摘　要

近年来，以ChatGPT为代表的大规模预训练模型不断突破AI技术瓶颈，AI应用场景碎片化问题有望在短期内从根本上得到解决。未来，集中式AI应用研发将会取代传统的小作坊式生产，这一趋势对支撑AI模型训练、微调和部署等环节的人工智能平台提出了更高的要求。本文针对主流人工智能平台存在部分问题，设计了一套训练、推理一体化平台。该平台通过工作流引擎实现了机器学习流水线的高效调度，利用虚拟化和容器化技术解决了硬件资源分配和调度问题，此外基于自动化表单工具实现了算子的组件化和插件化管理。本文所设计的训推一体平台将有效降低AI应用的开发门槛，促进AI应用集中式和规模化生产，推动大规模预训练模型快速渗透到各个垂直行业应用场景。

关键词

预训练大模型，训推一体，任务调度，算力调度，自动表单

Architecture Design and Key Technology Research of Training-Reasoning Integrated Platform

Binghao Liang, Chuangang Zhang

Inspur Communication Information System Co., Ltd., Jinan Shandong

Received: Aug. 21st, 2023; accepted: Sep. 19th, 2023; published: Sep. 26th, 2023

Abstract

In recent years, the large-scale pre-trained model represented by ChatGPT has continuously broken through the existing bottleneck of AI technology, and the problem of fragmentation of AI application is expected to be fundamentally solved in the short term. In the future, centralized AI application development will replace traditional individual workshop production, and this trend

文章引用: 梁秉豪, 张传刚. 训推一体平台架构设计与关键技术研究[J]. 计算机科学与应用, 2023, 13(9): 1748-1755.
DOI: 10.12677/csa.2023.139173

puts higher requirements on artificial intelligence platforms that support AI model training, fine-tuning and deployment. Aiming at the existing problems in the main stream artificial intelligence platform, this paper designs a training-reasoning integrated platform. This platform realizes the efficient scheduling of machine learning pipeline through workflow engine, solves the problem of hardware resource allocation and scheduling by using virtualization and containerization technology, and realizes the componentization and pluggability of AI operators based on automatic form tools. The training-reasoning integrated platform designed in this paper will effectively lower the development threshold of AI applications, facilitate the centralized and large-scale production of AI applications, and accelerate the penetration of large-scale pre-training models into various vertical industry.

Keywords

Large-Scale Pre-Trained Model, Training-Reasoning Integrated, Task Scheduling, Computing Power Scheduling, Automatic Forms

1. 引言

随着 ChatGPT 等大规模预训练模型不断突破人们的预期，人工智能技术正在引发新一轮的科技浪潮。近年来，随着人工智能算法逐步从实验室走向大规模商用，落地过程中的各种问题不断浮现。传统方式通过直接登录服务器运行代码脚本的方式进行模型训练，存在资源利用不充分，数据、模型等过程资产无法归集等一系列问题。此外模型训练完成后，由于训练环境和部署环境的软硬件资源情况可能有差异，模型迁移到目标设备部署推理服务时可能出现各种软硬件环境适配问题。

本文基于机器学习开发运维一体化(MLOps)理念，设计并实现了人工智能算法训推一体平台，向下纳管异构算力资源，向上提供零代码、低代码和全代码三类算法开发工具。通过该平台可以支撑 AI 模型的大规模训练和推理，对研发过程产生的数据和模型资产进行统一管理。

2. 相关工作

随着生成式 AI 引发的新一轮智能化浪潮，各大云服务厂商先后发布了用于 AI 应用开发的人工智能云平台。例如百度 BML 平台[1]，阿里机器学习平台 PAI [2]和腾讯 TI 平台等产品，主要通过 SaaS 服务的方式，向个人或企业开发者提供人工智能算法的一站式自助开发服务。相比于传统的人工智能算法开发模式，通过此类云服务可以省去环境搭建等繁琐工作，大幅降低数据采集、数据标注、特征工程和模型选择等工作的学习门槛，从而有效控制人工智能应用开发成本。目前，各类人工智能平台已广泛应用在互联网、通信、金融和医疗等行业，助力千行百业完成垂直领域的 AI 算法快速研发和上线运行。2021 年，为进一步规范和引导人工智能平台有序发展，中国人工智能产业发展联盟和中国信息通信研究院联合发布了首个面向人工智能平台的行业标准[3]。人工智能平台打通了底层的软硬件资源，为开发者提供了人工智能应用开发工具包和资源分配调度方案。

在算法研发方面，针对传统人工智能算法开发模式的各种局限性，束柬等人[4]提出了一种面向深度学习模型训练的四层架构，解决了传统小作坊研发模式训练和推理过程割裂的问题。Waldemar 等人[5]提出了一种基于云化的人工智能模型全生命周期管理平台(ModelOps)，实现模型研发流水线的自动化，

模型的可信任和可复现。Kim 等人[6]针对社交网络软件中的机器学习任务，构建了从特征存储、模型训练到在线推理的工作流，充分利用 CPU 和 GPU 资源完成海量数据分析工作。

在任务调度和资源分配方面，黄巨涛等人[7]以最大化资源利用为优化目标，利用 ARIMA 和 RBF 进行资源预测，采用 Docker 和 Kubernetes 技术实现资源监控和调度，提高了人工智能开发训练平台的集群节点资源利用率。Wu 等人[8]设计了一个自动机器学习平台，对服务器资源进行统一管理，基于配置文件描述业务所需资源，根据集群负载自动进行任务调度，提高了集群资源利用率，缓解了机器负载分布不均匀等问题。

通过对现有文献和主流技术的分析可以发现，现有研究已经实现了人工智能算法训练和推理流程的打通，同时也能实现对算力资源的监控和管理。然后，针对人工智能平台底层算力资源虚拟化和算力调度方面的研究较少，在任务调度过程中也没有充分考虑设备可用性等因素，存在资源闲置等现象。此外，主流人工智能平台大多不支持用户自行对算子进行扩展，导致平台使用场景受到一定限制。针对上述问题，本文基于主流开源技术，提出了一种人工智能算法训推一体平台框架设计，系统阐述了平台总体架构和关键技术。该平台采用虚拟化技术对底层算力资源进行抽象，并通过容器化技术对 Pytorch 等计算框架进行封装。基于 k8s 等云原生组件监控容器和硬件资源运行状态，然后利用群体智能算法实现训练和推理任务的实时调度，大幅提升了资源利用效率。此外，基于工作流引擎完成机器学习流水线的任务调度，通过自动化表单生成器实现自定义算子的按需扩展。

3. 训推一体平台体系结构

3.1. 开发运营一体化流程体系

在人工智能算法开发过程中，存在数据、算法和模型资产难管理，算法开发、部署和迭代周期长，模型指标监控手段缺失，团队协作缺少工具支撑四大核心问题。如图 1 所示，通过训推一体平台可以打通需求设计、模型开发、模型交付和模型运营四大阶段，实现人工智能开发运营一体化。

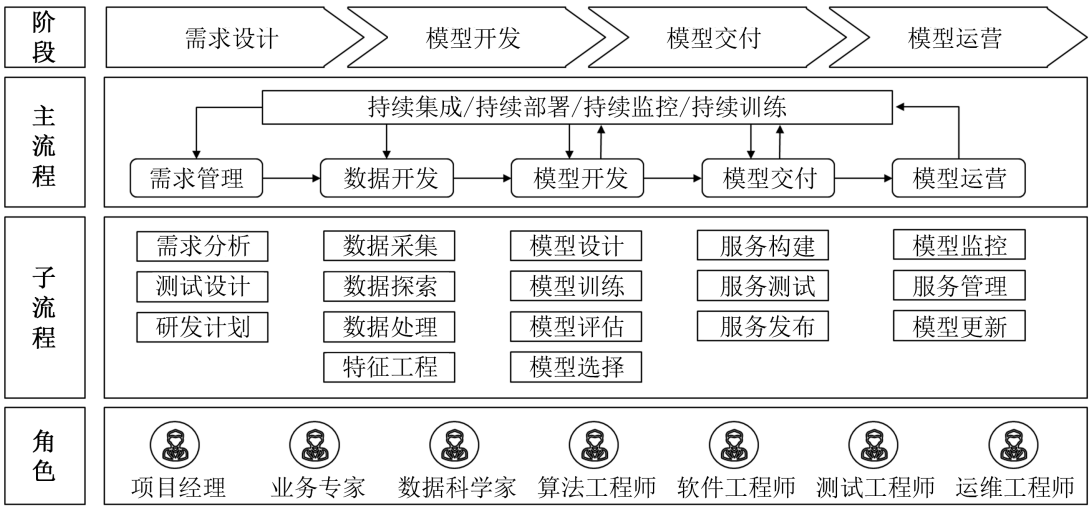

Figure 1. Integrated process of AI model development and operation
图 1. 人工智能模型开发运营一体化流程

1) 在需求设计阶段，项目经理或业务专家根据实际业务需求，将业务问题抽象成数学模型，选择一个或者多个潜在适用的备选算法进行效果验证。数据科学家将收集到的业务数据进行数据清洗，并完成特征工程设计等准备工作。

2) 在模型开发阶段，业务专家首先将处理后的业务数据和特征数据导入备选算法，并对其效果进行验证，如备选算法均不能满足业务需求，则需要算法工程师按照备选算法存在的问题进行改进或重新设计算法。当算法基本满足业务需求后，业务专家将通过模型训练模块将算法训练成对应的人工智能模型。

3) 在模型交付阶段，软件工程师将模型部署成 API 服务或者打包成 SDK 应用到业务系统中，测试和运维工程师对 API 服务和打包后的程序进行测试和运维。

4) 在模型运营阶段，通过训推一体平台的服务监控模块对各项指标进行监控，当出现服务不可用或模型漂移等问题时，自动分析故障环节并完成服务重启或模型重训等操作。

此外，如图 2 所示，本文所设计的训推一体平台主要包含了数据处理流水线和机器学习流水线，其中针对数据挖掘场景和深度学习场景的技术特点，又将机器学习流水线细分成数据挖掘分支和深度学习分支。

1) 数据处理流水线包含数据采集、数据探索、数据处理、数据增强和特征工程 5 大过程，其中数据增强主要应用在深度学习场景，增强后的数据将存入平台的样本库进行持久化存储。例如在光学字符识别场景中对原始图像进行尺度变换等操作。特征工程主要应用在数据挖掘场景，得到的特征数据将存入平台的特征库进行持久化存储，例如在网络流量预测场景中对异常值进行过滤。

2) 机器学习流水线主要以人工智能算法和处理后的样本数据作为输入，通过算法选择、模型构建、模型训练、模型评估、模型部署和模型监控 6 大过程完成人工智能应用开发。训练得到的人工智能模型将统一存储到平台的模型库中，其中数据挖掘任务主要数据源来自于特征库，深度学习任务主要数据源来自于样本库。

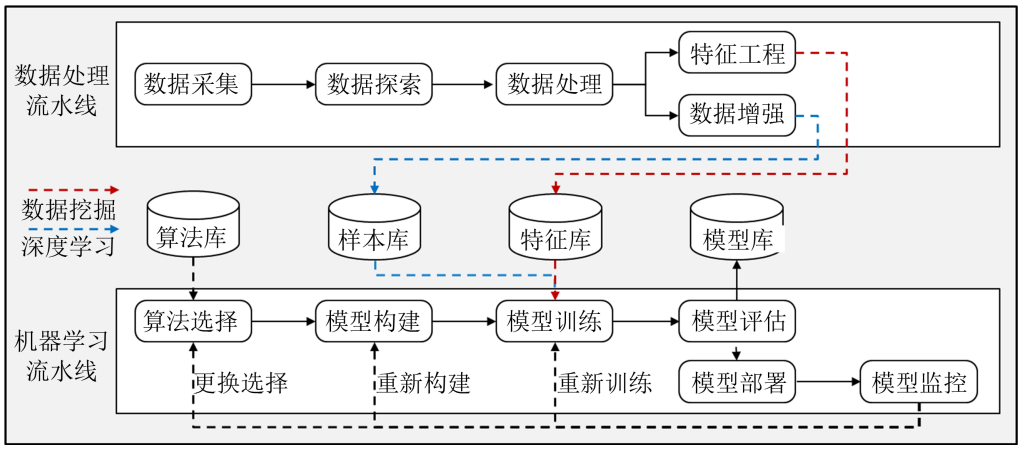

Figure 2. Design of artificial intelligence model development pipeline
图 2. 人工智能模型开发流水线设计

3.2. 训推一体平台整体架构

3.2.1. 平台功能架构

训推一体平台主要基于 MLOps 理念进行功能规划和架构设计，如图 3 所示，平台包括需求开发流水线、机器学习流水线、持续训练流水线和持续运营流水线。考虑到业务专家、数据科学家和算法工程师等主要用户需求，平台支持零代码、低代码和纯代码三种建模方式。业务专家可通过向导式模块，选择平台算法库中的成熟算法，通过配置数据源和算法参数实现零代码建模。数据科学家可通过拖拽式模块，对平台算子库中的成熟算子进行灵活编排和快速组装，满足大部分数据挖掘业务需求。算法工程师可通过在线编程模块，免去环境搭建等繁琐步骤，专注于代码编写，快速验证算法可行性。

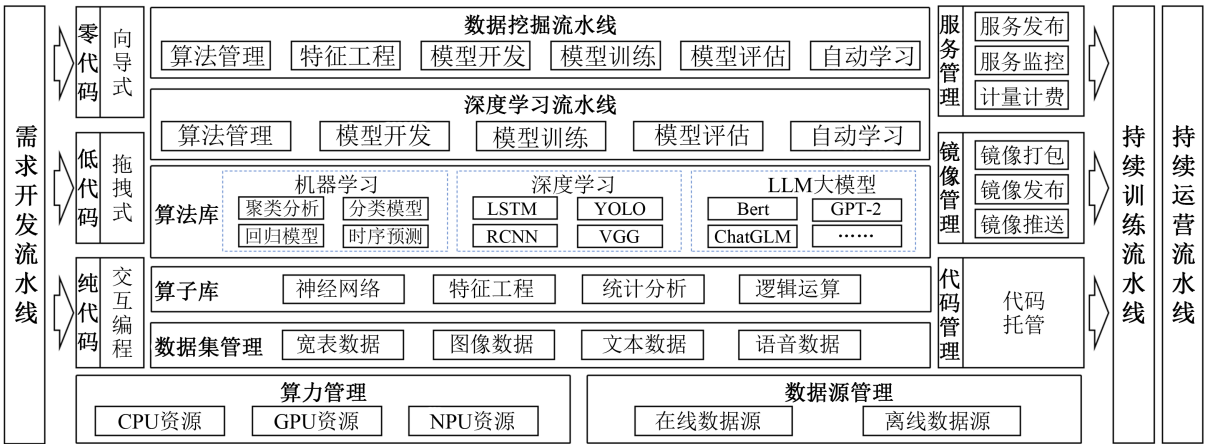

Figure 3. Functional architecture diagram of the training-reasoning integrated platform
图 3. 训推一体平台功能架构图

3.2.2. 平台技术架构

训推一体平台需要利用异构算力设备资源，帮助算法研发人员完成数据预处理、模型训练和服务部署等一系列操作，从而完成人工智能应用构建。从后台技术架构上看(如图 4)，本平台主要包括平台应用层、计算任务层、数据处理层和资源管理层。其中平台应用层主要用于支撑零代码、低代码和纯代码三种建模方式。计算任务层集成了 Sklearn 等机器学习框架以及 Pytorch 等深度学习框架，采用 Azkaban 完成工作流调度。数据处理层分别通过 HDFS 和 Ceph 存储结构化的宽表数据和非结构化的图像、文本数据。利用 MapReduce 和 OpenCV 等数据加工工具对原始数据进行预处理、数据增强和特征工程等操作，预处理后的特征向量将存储在向量数据库(Milvus)中。资源管理层对底层异构算力芯片如：CPU、GPU 和 NPU 等进行抽象，通过 K8s 和 vCUDA 实现资源的分配，利用 Promethues 和 DCGM 实时监控容器和硬件资源运行状态。

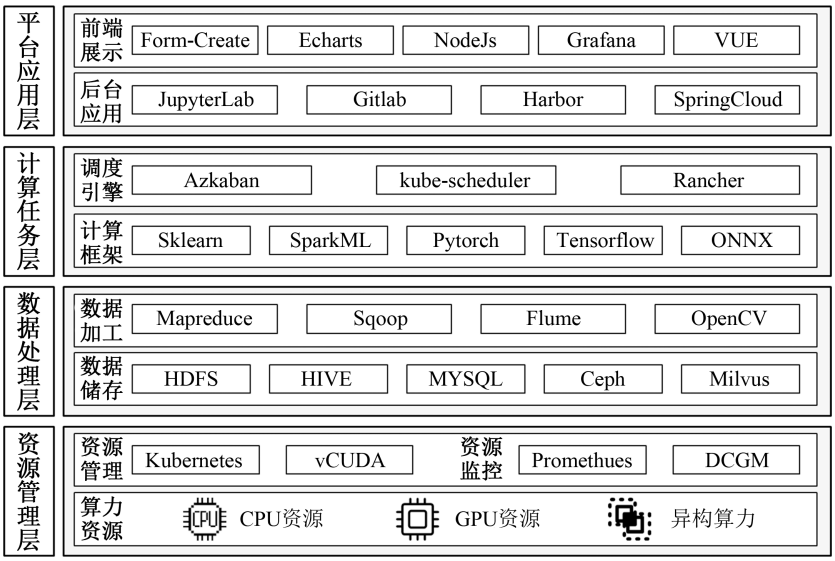

Figure 4. Technical architecture diagram of training-reasoning integrated platform
图 4. 训推一体平台技术架构图

在计算框架方面，目前主流的深度学习框架包括国外谷歌公司开源的 TensorFlow [9] [10]，脸书开源的 PyTorch [11]，以及国内用户规模较大的飞桨[1]。在传统机器学习方面，Sklearn [12]具备比较完善的

开源生态，支持了分类、回归和聚类等常用机器学习算法，同时可以完成特征提取、数据预处理和模型评估三类辅助任务，受到了数据科学家和科研工作者等群体的青睐。针对海量数据分析场景，平台支持SparkML [13]算子，可以实现分布式算法训练。该框架在通信行业应用较多，由于运营商各类基站和网络通信设备产生了大量结构化数据，针对故障发现和根因定位等网络运维场景，可以实现对亿级数据进行在线分析。

4. 训推一体平台关键技术

为提升训推一体平台任务调度能力和算力资源调度能力，增强平台功能可扩展性，平台涉及三大核心技术：算子插件化技术、工作流任务调度技术和算力资源调度技术。

4.1. 算子插件化技术

在训推一体平台使用过程中，需要不断集成新的算子丰富平台功能，目前大部分主流厂商和开源的AI平台均不支持用户自行扩展平台算子。本文主要基于自动化表单生成技术，提升平台可扩展性，方便用户添加自定义算子。如图 5 所示，在平台管理端，支持以向导方式自行维护算子，完成平台算子的动态扩充。基于自动化表单生成技术，通过编写 json 格式的方式，自动生成前端算子参数配置页面。在完成算子后台 python 代码编写后，可以运行单元测试，通过测试后可以发布到平台进行低代码调用(如图 6 所示)。

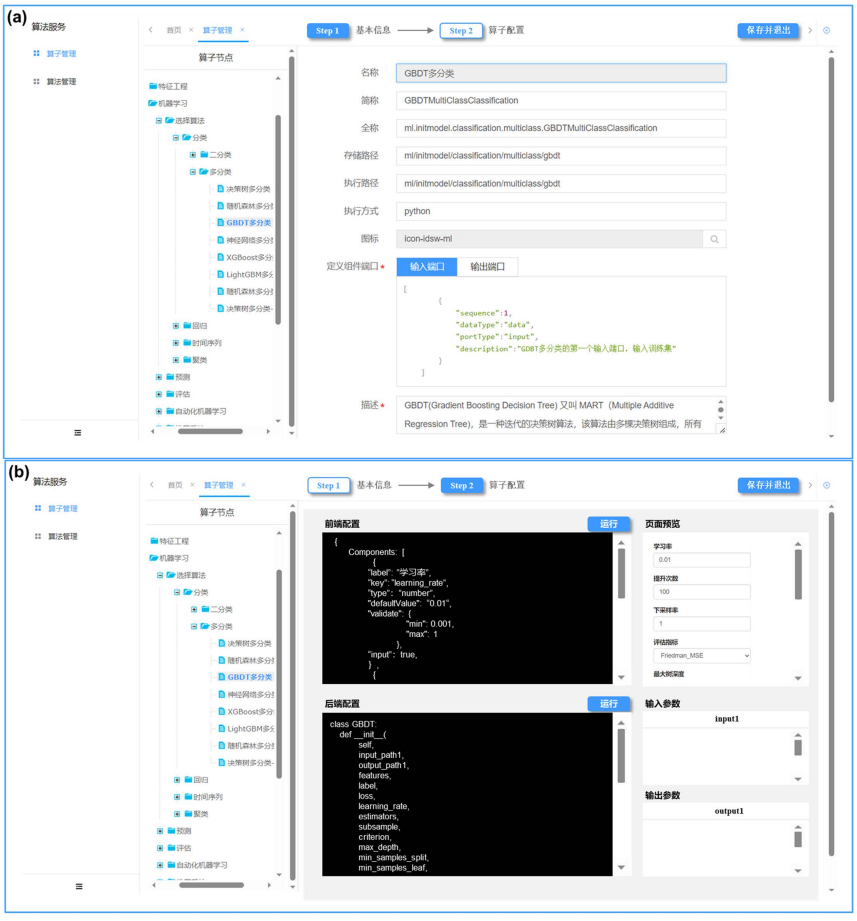

Figure 5. Operator plug-in technology: (a) Operator basic information management; (b) Operator configuration information management
图 5. 算子插件化技术：(a) 算子基本信息管理；(b) 算子配置信息管理

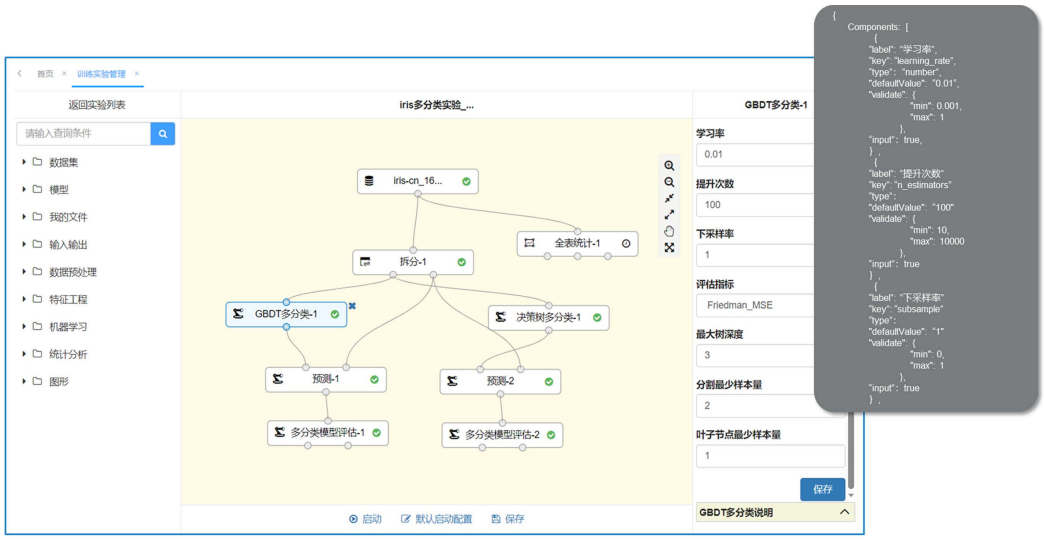

Figure 6. Task scheduling process
图 6. 任务调度流程

4.2. 工作流任务调度技术

面向数据科学家和数据分析师等群体，平台提供低代码建模工具，通过 azkaban 工作流调度引擎完成数据挖掘流水线中的工作流调度。如图 6 所示，一个工作流里面包含多个任务节点，包括数据集导入、数据预处理、机器学习和算法评估等算子，每个算子之间有执行的先后逻辑关系。平台通过拖拽式建模的方式，将算子组装成 DAG 图并形成工作流，在点击启动后交由 azkaban 进行调度和执行。

4.3. 算力资源调度技术

在算力资源调度方面，训练和推理芯片，云侧和边侧芯片架构差异加大，平台在算力调度过程中需要考虑算法在异构算力之间的适配问题。以英伟达系列为例，训练任务一般采用 A100 等 GPU 卡，推理任务一般执行 T4 或 Jetson Nano 系列芯片上。如图 7 所示，训推一体平台将底层算力资源进行隔离，形成训练环境、测试环境和推理环境三大环境。训练环境和推理环境分别采用训练芯片和推理芯片，在测试环境同时具有训练和推理芯片，方便完成模型转换等操作。

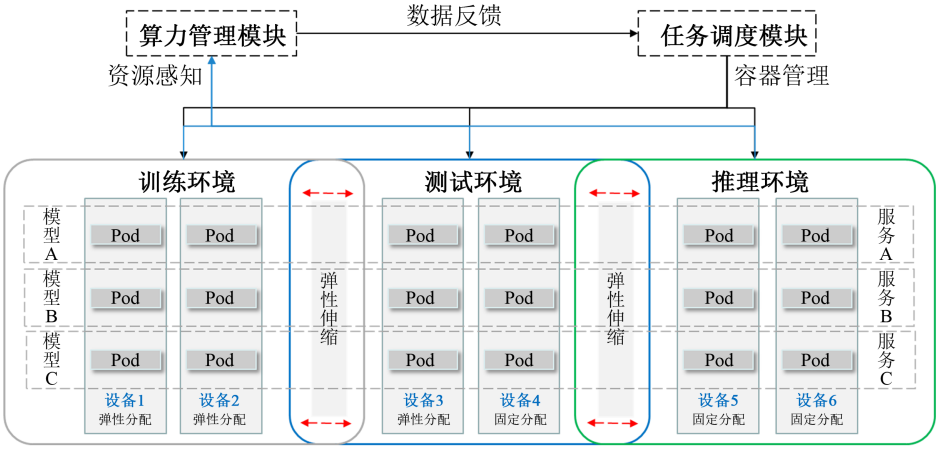

Figure 7. Computing power resource scheduling process
图 7. 算力资源调度流程

算力资源分配需要考虑使用场景和业务需求，平台在训练环境、测试环境和推理环境分别采用了不同的分配策略。通过虚拟化技术进行 GPU 资源池化，主要分配策略包括弹性分配和固定分配两种方式。由于在模型训练过程中，对训练任务实时性要求不高，训练环境主要通过弹性分配的方式提升 GPU 资源利用率，相同资源可供多个任务复用。当两个训练任务峰值算力资源需求综合不超过物理资源时，可以充分使用 GPU 算力，满足性能需求。当超过物理资源时，可以根据优先级进行排队和分配，优先级高的任务分配更多的算力，保障重要任务优先运行。由于推理任务实时性要求较高，在推理环境中主要通过固定分配的方式防止计算资源抢占，保障业务正常运行。测试环境混合采用弹性分配和固定分配方式，执行功能测试任务时可以通过弹性分配方式，最大化资源利用效率；执行压力测试时，需要按照固定分配方式实现资源硬隔离，确保测试结果准确性。

5. 总结

本文主要针对现有人工智能平台存在的局限性，基于 MLOps 理念设计了一套训练、推理一体化平台。平台利用算子插件化技术、工作流任务调度技术和算力资源调度技术，实现了机器学习流水线的高效调度和平台算子的灵活扩容。本文所设计的训推一体平台将有效降低大规模预训练模型和 AI 算法的开发门槛，促进集中式和规模化生产，推动 AI 应用快速渗透到各个垂直行业应用场景。

参考文献

[1] 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.

[2] Jia, X., Jiang, L., Wang, A., Xiao, W., Shi, Z., Zhang, J., Li, X., Chen, L., And, Y.L., Zheng, Z., Liu, X. and Lin, W. (2022) Whale: Efficient Giant Model Training over Heterogeneous GPUs. 2022 *USENIX Annual Technical Conference*, Carlsbad, California, July 11 2022, 673-688.

[3] 人工智能开发平台系统功能要求第 1 部分: 功能要求: AIIA/P 0006-2022 [S]. 中国人工智能产业发展联盟, 中国信息通信研究院, 2022.

[4] 束柬, 陈剑波. 深度学习平台体系架构及其关键技术[J]. 计算机应用研究, 2023, 40(11): 38.

[5] Hummer, W., Muthusamy, V., Rausch, T., Dube, P. and Oum, P. (2019) ModelOps: Cloud-Based Lifecycle Management for Reliable and Trusted AI. *Proceedings of the* 2019 *IEEE International Conference on Cloud Engineering* (IC2E), Prague, 24-27 June 2019, 113-120. https://doi.org/10.1109/IC2E.2019.00025

[6] Hazelwood, K., Bird, S., Brooks, D., Chintala, S. and Diril, U. (2018) Applied Machine Learning at Facebook: A Datacenter Infrastructure Perspective. *Proceedings of the IEEE International Symposium on High Performance Computer Architecture*, Vienna, 24-28 February 2018, 620-629. https://doi.org/10.1109/HPCA.2018.00059

[7] 黄巨涛, 郑杰生, 高尚, 刘文彬, 林嘉鑫, 董召杰, 王尧. 基于云平台的人工智能开源开发平台框架研究[J]. 自动化与仪器仪表, 2020(7): 5.

[8] Wu, C., Haihong, E. and Song, M. (2020) An Automatic Artificial Intelligence Training Platform Based on Kubernetes. *Proceedings of the BDET* 2020: 2020 2*nd International Conference on Big Data Engineering and Technology*, Singapore and China, 3-5 January 2020, 58-62. https://doi.org/10.1145/3378904.3378921

[9] Abadi, M., Barham, P., Chen, J., Chen, Z. and Zhang, X. (2016) TensorFlow: A System for Large-Scale Machine Learning. USENIX Association, Carlsbad, California.

[10] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J. and Devin, M. (2015) TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. https://arxiv.org/abs/1603.04467

[11] Paszke, A., Gross, S., Massa, F., Lerer, A. and Chintala, S. (2019) PyTorch: An Imperative Style, High-Performance Deep Learning Library.

[12] Swami, A. and Jain, R. (2013) Scikit-Learn: Machine Learning in Python. *Journal of Machine Learning Research*, **12**, 2825-2830.

[13] Meng, X., Bradley, J., Yavuz, B., Sparks, E., Talwalkar, A. (2015) MLlib: Machine Learning in Apache Spark. *Journal of Machine Learning Research*, **17**, 1235-1241.