arXiv:2303.00501v2 [cs.LG] 8 Jul 2023

OMNIFORCE：以人为本、大模型赋能的云边协同自动化机器学习系统

Chao Xue 刘伟 ShuaiXie 王振芳 李佳兴 彭旭阳 Liang Ding ShanshanZhao 曹琼 YiboYang Fengxiang He 蔡博华 卞荣成 赵一言 Heliang Zheng 刘向阳 刘东凯 刘大庆 Li Shen Chang Li 张世进 张宇康 GuanpuChen 陈世祥 Yibing Zhan Jing Zhang Chaoyue Wang *

Dacheng Tao

京东探索研究院

{xuechao19, wangchaoyue9}@jd.com

摘要

自动化机器学习（AutoML）旨在以最少的人力构建机器学习模型。尽管针对 AutoML 的总体研究已相当丰富，目标是让人类在构建人工智能（AI）应用时脱离流程，但很少有文献关注 AutoML 如何在开放环境场景中良好运行，例如大模型的训练与更新过程、工业供应链或工业元宇宙。在这些场景中，人们经常会在搜索过程中面临开环问题：必须持续收集数据、更新数据和模型，满足开发与部署环境的要求，支持海量设备，修改评估指标等。采用纯数据驱动的方法解决开放环境问题，需要大量数据、计算资源以及专职数据工程师付出的努力，这使当前的自动机器学习系统和平台效率低下且在计算上难以实现。人机交互是解决开放环境人工智能问题的一种切实可行的方式。本文介绍了 OmniForce，这是一种以人为中心的自动机器学习（HAML）系统，兼具人类辅助机器学习和机器学习辅助人类技术，旨在将自动机器学习系统付诸实践，并在开放环境场景中构建自适应人工智能。具体而言，我们从以下方面介绍 OmniForce：数据、标签、模型、算法和搜索空间的机器学习版本管理；流水线驱动的开发与部署协作；灵活搜索策略框架；以及广泛提供并支持众包的应用算法，包括大模型。我们提出的云原生 OmniForce 方法既可以运行在公有云/私有云上，也可以运行在本地部署环境中。此外，OmniForce 构建的（大）模型可以在几分钟内自动转换为远程服务；这一过程称为模型即服务（MaaS）。在多个搜索空间和真实世界用例中获得的实验结果证明了 OmniForce 的有效性和效率。

关键词 以人为中心的自动化机器学习（HAML）· 云边协同 · 大模型 ·模型即服务（MaaS）

*通讯作者

1 引言

近几十年来，机器学习（ML）在计算机视觉 [1, 2], 自然语言处理（NLP） [3, 4], 语音识别 [5, 6], 内容生成 [7, 8] 和表格数据处理 [9, 10]等领域取得了巨大成功。机器学习技术的快速发展催生了广受欢迎的人工智能（AI）产品，例如 Tesla Autopilot [11], 谷歌翻译 [12], Siri [13], 和 ChatGPT [14]。随着机器学习在训练数据量以及针对不同任务定制的模型/神经网络数量方面的需求呈指数级增长，以完全自动化的方式设计定制超参数并识别用于训练的神经网络，而无需人工干预——这一过程称为自动化机器学习（AutoML）——已经取得了巨大成就。

近年来，自动化机器学习的发展主要体现在算法和系统两方面。就算法而言，大量研究采用了基于遗传算法 [15], 随机搜索 [16], 贝叶斯优化 [17], 强化学习 [18] 和可微分技术 [19] 的方法。为实现后者，业界开发了 Optuna[20], Ray-Tune [21], HyperOpt [22], NNI [23] 和 Orion [24] 等多种超参数优化框架，以支持可扩展的试验和可定制的搜索算法。Auto-sklearn [25] 还支持利用元学习来借助历史记录，为搜索过程提供热启动。与需要额外工作才能支持 Kubernetes 的其他框架不同， [26], Katib [27] 是一种云原生框架，可切实部署于生产环境。与开源自动化机器学习系统的探索相比，许多公司都向市场推出了自己的自动化机器学习产品，例如 Google Cloud AutoML [28], IBM Watson AutoAI [29], Amazon SageMaker [30],和 H2O Driverless AI [31]。这类平台旨在帮助机器学习专业知识有限的开发者在短时间内构建人工智能模型。

如上所述，尽管已有众多自动化机器学习框架被提出，但我们尚未看到自动机器学习系统在工业界得到预期的广泛应用。我们认为，这些框架普及率较低的原因如下。

- 仅面向闭环问题——大多数自动化机器学习框架只关注闭环问题，即数据、算法和指标都是确定性的；因此，其设计理念是在构建人工智能应用时将人类排除在流程之外。然而，实践中的人工智能相关问题往往是开环任务，尤其是在训练和更新大模型的过程中，或是在工业供应链中。此时，人们需要持续收集数据、更新数据和模型的版本，并修改生产过程中产生的评估结果和奖励。使用当前的数据驱动 AutoML 系统解决开环问题效率低下，甚至在计算上难以实现，因为这类系统需要大量数据才能学习领域知识和业务逻辑。
- 缺乏部署考量——大多数自动化机器学习框架只关注搜索和训练阶段，而忽视了推理和部署阶段。在真实的工业场景或工业元宇宙中，会遇到大量具有不同部署（推理）要求的设备；在这些场景中，人们利用仿真或 XR [2]技术来降低实体生产流程中的故障风险，并构建高效的供应链，涵盖设计、开发、制造、定价、销售、存储、运输和售后服务等阶段。

- 应用算法有限——大多数自动化机器学习框架仅包含一些预定义或内置的应用算法及其对应的搜索空间，这些内容通常会向用户透明呈现，以便用户使用。然而，现实中存在大量各式各样的人工智能应用，而自动机器学习系统所能支持的内置应用算法数量有限。考虑到应用种类繁多，尤其是开发和部署要求各不相同的应用（例如云边协同应用），如果搜索过程仅提供预定义的应用算法，自动化机器学习系统的广泛应用就会受到限制。

为了解决上述问题，我们开发了 OmniForce，以支持开放环境中的自动化机器学习；OmniForce 围绕以下理念构建。

- 以人为中心的自适应自动化机器学习（HAML）——我们将 OmniForce 设计为既支持人类辅助机器学习，也支持机器学习辅助人类。因此，用户可以通过与 OmniForce 交互，高效处理业务逻辑和数据收集流程。

- 实际中的云边协同——我们提出一种流水线驱动的自动化机器学习框架，通过协同开发和部署，搜索具有不同训练和部署要求的人工智能应用。
- 众包应用算法——我们引入众包理念，将各种观测到的应用算法整合到 OmniForce平台中。通过标准化数据抽象范式、

[2]XR 是一个统称，涵　　盖虚拟现实（VR）、增强现实（AR）和　　混合现实（MR）。

应用算法和搜索空间，我们可以轻松整合并复用应用算法和搜索空间。

我们在图 1 中阐释了 HAML 的概念。用户以人类辅助机器学习和机器学习辅助人类的方式与自动机器学习系统进行交互。具体而言，HAML 任务包含数据收集与标注、特征、应用算法、搜索空间、搜索、训练与部署以及可视化等要素。对于数据收集与标注，一方面，用户通过主动学习收集数据，以提高机器学习算法的准确性；另一方面，机器学习算法帮助用户高效地标注数据。此外，数据隐私保护与安全性在人类与自动机器学习系统之间的交互中发挥着重要作用。OmniForce 支持差分隐私技术，以保护用户数据。在特征方面，OmniForce 通过用户交互和 SQL 支持定制化特征流水线。此外，用户可以查看和分析数据与特征的统计元数据信息。在应用算法方面，鉴于众包和超深度模型具有广泛的通用性，OmniForce 的适用范围比许多仅内置小规模应用算法的其他自动化机器学习系统更广。默认情况下，OmniForce 隐藏用于设置搜索空间的细节，但用户在通过可视化获得知识后，可以通过调优先验或偏好来配置搜索空间。对于搜索、训练和部署，用户定义开发环境与部署环境，设置需求和约束，并执行单目标或多目标优化。OmniForce 凭借强大的搜索能力，支持云边协同，以满足训练环境和部署环境的不同需求。对于作为 HAML 核心的可视化部分，用户可以从 OmniForce 中学习其自动机器学习流水线的相关知识，获得对搜索得到的架构和超参数的解释，并获取可用于指导后续工作的建议。例如，顾问可能会建议用户根据采样候选项的统计分布更新搜索空间。此外，顾问还可能鼓励用户从某个特定类别中收集更多数据，或放宽某些延迟约束和功耗限制。HAML 循环使用户能够充分参与人机协作，实现两方面的目标：利用机器增强人类能力，以及利用人类经验和操作提升机器智能。

我们的贡献如下。

1) OmniForce 是一款前沿的以人为中心的自适应自动化机器学习系统，支持开放环境场景，例如大模型的训练与更新、工业供应链和工业元宇宙。它包含一系列新颖的搜索策略、一种搜索空间更新策略，以及适用于计算机视觉（CV）、自然语言处理（NLP）和人工智能生成内容（AIGC）的大模型算法。因此，OmniForce 既适用于机器学习专业知识有限的开发者，也适用于数据科学家。

2) OmniForce 是一款云原生自动化机器学习系统，具有可扩展、容错和云边协同等特性，因此可在生产环境中运行。

3) OmniForce 遵循模型即服务（MaaS）模式，将搜索、训练、推理和部署流程全面贯通。当 OmniForce 成功完成模型构建后，用户不仅能获得模型，还能获得该模型的推理和部署服务。这使用户能够在几分钟内将模型转化为可部署在云端或边缘端的远程服务，帮助用户快速构建具备人工智能能力的前沿应用。

本文其余部分安排如下：第 2 节介绍 OmniForce 的系统架构和工作流。第 3 节介绍详细的设计理念，第 4 节展示所支持的特性，随后在第 5 节进行评估。第 6 节对相关工作进行比较，最后在第 7 节总结全文。

2 系统架构

如后文所述，OmniForce 由多个组件实现，包括作业估算器、工作节点、训练任务估算器（TTE）、任务部署估算器（DTE）、任务边车、调度器和管理器。图 2 展示了 OmniForce 系统概览的简化框图。

应用层负责处理业务逻辑，例如上传数据、选择应用算法和预训练模型，以及设置搜索空间以启动搜索过程。OmniForce 提供了丰富的数据和应用算法，其中包含基于大模型的方法，供专注于业务逻辑而不关注机器学习算法细节的用户使用。用户还可以利用并贡献集成到 OmniForce 中的众包资源，以满足日益增长的机器学习应用需求。

应用层的所有数据、算法和搜索空间都通过统一抽象与搜索引擎交互，这意味着这些资源均以统一视图的形式表示和组织。后续变更会反映在

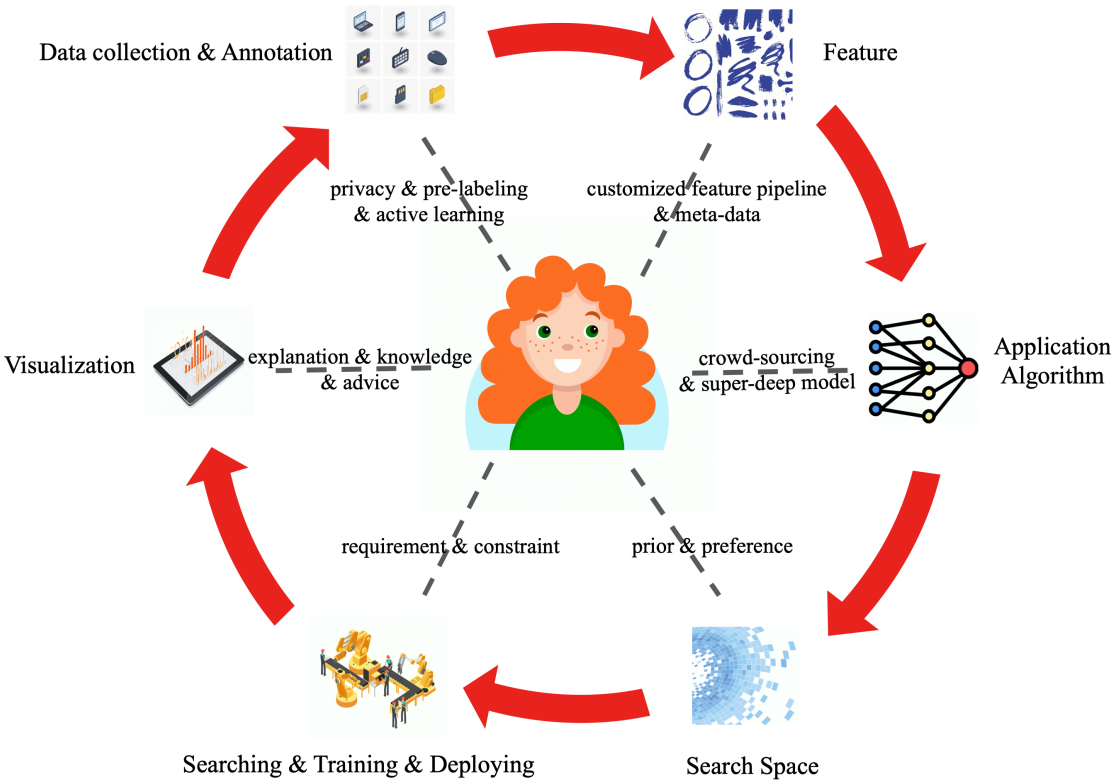


图 1：HAML。环形结构表示 HAML 的流水线。用户会与循环中的关键步骤/节点交互。与大多数只关注搜索和训练部分的自动化机器学习框架不同，HAML 关注整个机器学习流水线，在这一过程中，需要考虑数据收集；更新数据版本、特征流水线、算法和搜索空间；修改评估和奖励；并从可视化中获取知识，以指导下一步工作。

不同版本的 th　　这些资源中。接口层也涉及隐私和安全。有关更多详情　　g
接口层可以　　详见第 3.1 节。

搜索引擎通过一个具有多目标协作能力的搜索策略框架实现，该框架连接半同步控制器，用于多程序多数据（MPMD）分发；连接调度器，用于将作业和任务分配给云资源；还连接格式化器，用于学习历史知识并生成自动机器学习流水线（充当元学习器）。具体而言，我们实现了一个灵活的并行贝叶斯优化（BO）框架，该框架完全基于 PyTorch 运行，支持使用多种代理模型和采集函数进行 BO，其中包括我们提出的新模型，以应对大型离散空间（这是神经架构搜索（NAS）的应用场景）。我们还在搜索策略框架中支持修订版 Hyperband [32]，MF-NAS [33], 以及一种新颖的进化方法。当搜索空间较大时，搜索过程会变得更加困难。我们引入多个工作节点并行寻找优质候选项。考虑到并行效率与不可避免的同步之间的权衡（某些策略需要同步来保证性能），我们实现了一个半同步 MPMD 分发器。与 Pathways[34], 类似，Pathways 已展示单程序多数据（SPMD）范式在机器学习计算中的局限性，我们将 NAS 过程表示为 MPMD。有关搜索引擎的更多详情，请参见第 3.4 节。

OmniForce 运行于 Kubernetes 和 Kubeflow [35] 之上，以支持可扩展性、容错性和多租户。OmniForce 是一款可直接投入使用的云原生系统，可作为服务部署在公有云/私有云或本地部署环境中。

2.1 组件

本节将详细介绍 OmniForce 的一些基本组件，包括作业估算器、工作节点、TTE、DTE、任务边车、调度器（格式化器和资源调度器）、管理器和顾问。

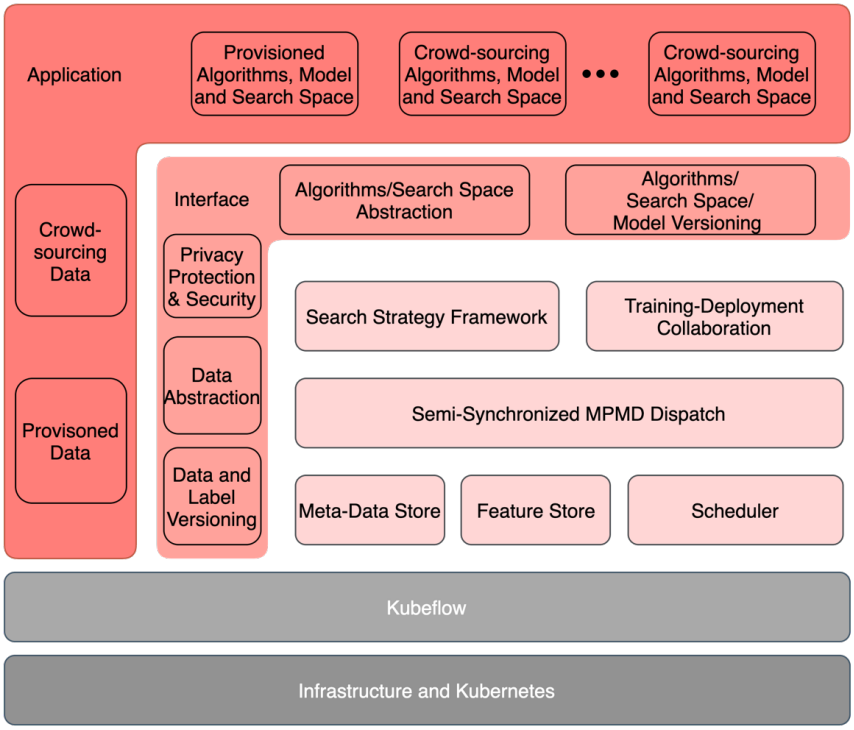


图 2：OmniForce 系统概览。OmniForce 基于 Kubernetes 和 Kubeflow 构建，是一个配备了丰富机器学习算法，并为众包算法、模型和搜索空间提供统一接口的自动化机器学习系统。

2.1.1 作业估算器

作业估算器是自动机器学习流程的半同步控制器，用于确定接下来应评估哪些任务、如何处理其奖励，以及何时开始下一次迭代。作业估算器包含两个关键组件：用于生成候选项的用户定义搜索空间，以及通过先进搜索算法执行串行和并行搜索过程的搜索策略。搜索空间和搜索策略的详情分别见第 3.2.2 节和第 3.4 节。

一些自动化机器学习框架倾向于采用完全同步机制，即只有当前轮次的所有任务都完成后，才会开始下一次迭代。然而，在多程序多数据设置中，同步框架无法保证高效率。相比之下，我们的作业估算器采用半同步机制，通过自适应最大等待时间控制下一次迭代的启动。超过该时间仍未完成的任务将被设为超时，作业估算器会在当前迭代中忽略这些任务，并根据已完成的观测值生成新任务。等待时间的计算方式灵活多样。例如，我们可以利用先前已完成任务的执行时间均值和方差来估算等待时间。考虑到多程序多数据设置下各任务的执行时间可能相差悬殊，我们还可以构建由贝叶斯优化框架提供的时间成本感知代理模型，为每个任务估算执行时间。

2.1.2 工作进程

工作进程是一组负责预留和评估候选任务的进程。在我们的设计中，作业估算器与工作节点并不直接通信，而是通过任务代理进行通信，任务代理通常由中间件实现。现有的大规模分布式自动化机器学习系统始终具有较高的计算需求，并高度重视可扩展性和容错性。我们的设计充分考虑了这些要求，将作业估算器与工作节点解耦，使用户能够根据实际资源自由增加或减少所使用的工作节点数量，从而实现可扩展性。此外，如果某些工作节点意外发生故障，我们的搜索过程无需额外操作即可适应这一新情况，只要仍有一个工作节点存活，就会继续进行搜索。此外，工作节点还能够处理许多常见异常，例如 GPU 内存耗尽和中间件连接丢失。

2.1.3 TTE

TTE 是用于评估候选任务的实际实体。工作节点预留任务后，将启动一个特定的 TTE 来评估该任务，并在完成后将结果报告给工作节点。OmniForce 开发了一个与 TTE 配套的任务边车服务，用于代理通信过程。OmniForce 支持众包，用户只需实现第 3.2.1 节中定义的算法接口，即可轻松将训练代码升级为 TTE 中可搜索的代码。更重要的是，由于运行在 Kubernetes 和 Kubeflow 上，OmniForce 能够利用 PytorchJob [36] 和 MPIJob [37]等云原生工作负载，实现弹性分布式训练。该功能有助于高效利用可用的集群资源，并处理可变的训练设置，例如大模型。

2.1.4 DTE

DTE 是与 TTE 协作的部署实体。当 TTE 完成模型训练过程后，训练后的模型会被发送至 DTE，以评估模型在部署环境中的性能。DTE 与 TTE 之间的通信通过任务边车完成。例如，在云边协同场景中，云侧凭借强大的云计算能力负责模型搜索和训练，而边缘侧则是实际的生产环境。OmniForce 关注这一需求，并提供多目标优化服务，以共同评估模型在训练环境和部署环境中的性能。该设计充分利用了云计算和边缘部署，弥合了开发环境与生产环境之间的差距。

2.1.5 任务边车

边车 [38] 是云原生系统中的一种设计模式，在同时运行两个紧密耦合的进程时非常有用。OmniForce 构建任务边车组件，将搜索逻辑与连接逻辑解耦，以适应复杂的训练和部署环境。在我们的自动机器学习工作流中，任务边车充当中间方，用于连接工作节点、TTE 和 DTE。该组件负责处理组件之间的琐碎操作，并扩展服务边界，使其适用于更多下游场景。

2.1.6 调度器

调度器会估算并调度作业所需的资源，由两个部分组成。第一部分将新作业格式化为适当的配置，其中包括搜索空间和估算资源；在我们的系统中，这个组件称为“格式化器”。第二部分称为资源调度器，它根据估算的资源和自动机器学习系统当前的负载，为给定作业分配实际的计算资源。

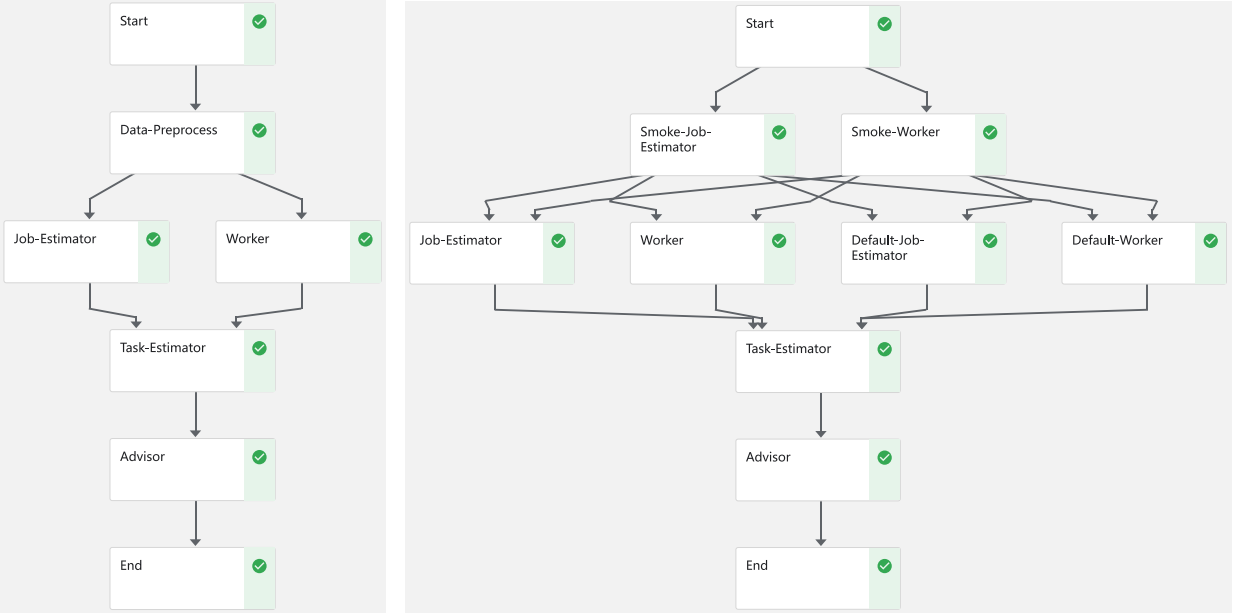
为估算时间段、内存和计算资源，格式化器会构建一个知识库，其中包含此前搜索过的历史实验。我们从以往作业中获取知识，例如超参数、指标配置、搜索过程的多保真度、内存和已使用的计算资源。由于采用现成的知识库，格式化器能够为新提交的作业找到合适的搜索算法和搜索空间，并向资源调度器提供估算的资源以及工作节点的并行度。此外，格式化器还定义搜索流水线的结构。

资源调度器管理所有集群资源，并防止因资源抢占而导致的死锁或待处理异常。具体来说，资源调度器根据格式化器提供的作业详情和系统当前负载，分配资源并调整搜索和训练阶段的并行度。它将作业划分为搜索、训练、推理和部署阶段，并分别调度各阶段的资源。资源调度器每次处理一个特定类型的作业。

2.1.7 管理器

管理器是一项管家服务，使 OmniForce 能够与云原生 Kubernetes 和 Kubeflow 资源交互。除了基本的创建、读取、更新和删除（CRUD）操作外，OmniForce 管理器秉持机器学习运维理念，借助 Kubeflow 流水线 [39]，（一种用于构建可移植、可扩展机器学习工作流的云原生服务）实现自动化机器学习工作流。此外，我们还构建了一些特定的自动机器学习流水线，并将各种云原生工作负载（如 Kubernetes 作业、Kubeflow PytorchJobs 和 KServe InferenceServices）集成到一条流水线中。因此，OmniForce 能以统一的方式管理这些云原生资源，从而简化自动化和复现自动化机器学习工作流的流程。

此外，考虑到自动化机器学习工作流的灵活需求以及集群计算资源的不可预测性，OmniForce 管理器将云原生工作负载抽象为模块化流水线元素，使 OmniForce 调度器和格式化器能够将这些元素编排成具有特定架构和资源



(a) (b)

图3：OmniForce 的自动机器学习流水线示例。（a）无代码流水线示例。（b）超参数优化流水线示例。每条流水线均以有向无环图（DAG）格式描绘一个具体的自动机器学习工作流，其中每个组件按照拓扑顺序执行。每个组件都会启动适当的云原生工作负载，以完成其工作。

并行度的流水线。生成流水线后，OmniForce 管理器会按拓扑顺序启动相关组件。我们在图3中展示了两个自动机器学习流水线示例：一个称为无代码流水线，另一个称为超参数优化流水线。在无代码流水线中，作业估算器和工作节点会同时启动，以搜索最佳架构和超参数。然后，任务估算器通过训练与部署协作构建模型。最后，顾问收集工作流中生成的元数据，并为用户提供有价值的见解。这里的起始和结束组件负责一些准备和清理步骤。在超参数优化流水线中，会添加冒烟测试作业估算器和工作节点，以便在进入实际搜索阶段前验证众包算法的正确性。随后，添加默认作业估算器和工作节点，以复现默认算法性能，从而与自动化机器学习搜索结果进行比较。

2.1.8 顾问

顾问是一种组件，可为用户提供有关其数据集和众包算法的全面洞察。OmniForce 顾问会在任务执行期间将搜索过程可视化。除损失和准确率等基本指标外，顾问还提供搜索空间中候选项的可视化图，以及超参数重要性等级的条形图，以展示其与选定指标之间的相关性，旨在理清超参数之间复杂的相互作用。这些信息有助于用户了解哪些超参数对其模型性能最为重要。此外，顾问还会针对搜索空间和数据集提供有价值的建议。例如，顾问可能会建议用户更新搜索空间，或鼓励用户收集某个特定类别的更多数据。

2.2 系统工作流

本节将介绍用户在 OmniForce系统 中进行交互时所使用的整体自动机器学习工作流，具体如图4所示。遵循以人为中心理念，OmniForce 设计了信息丰富且友好的用户界面，便于用户使用。此外，OmniForce 借助 Kubernetes 和 Kubeflow，以云原生、稳定且跨平台的方式运行服务。用户设定目标后，OmniForce 会引导其完成后续的模型构建和分析过程。主要步骤如下。

1. 用户输入会被转换为标准格式的模型需求，并发送至应用服务器 .

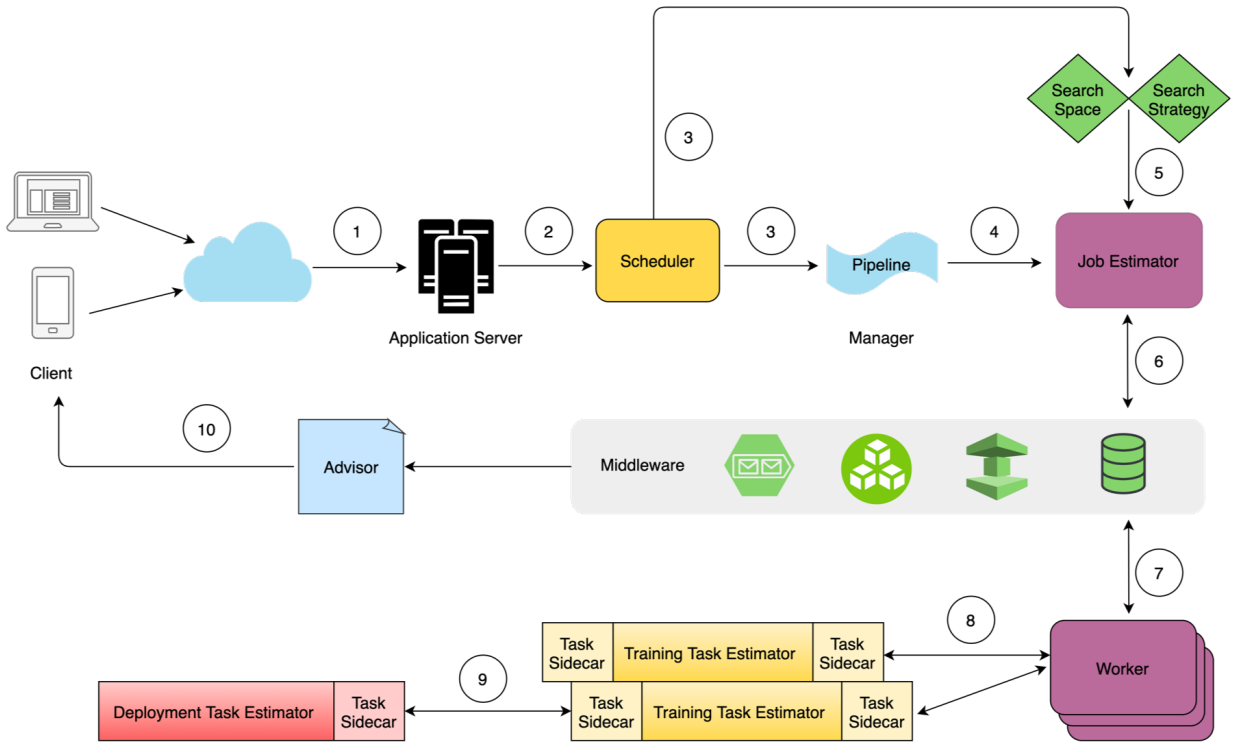


图4：OmniForce 的自动机器学习工作流。OmniForce系统的各组件按照该工作流相互交互。

2. 应用服务器验证用户的身份和权限，将模型需求转换为具体的自动化机器学习任务，并将任务信息存储在数据库中。最后，该任务会提交给调度器。3. 调度器根据任务信息和当前集群资源编排流水线。此过程可分为两个步骤。首先，格式化器生成搜索空间，根据任务信息和历史记录选择合适的搜索策略，然后将模块化流水线元素组织成逻辑流水线。其次，资源调度器根据当前集群资源计算合理的资源分配方案。值得注意的是，此步骤会将逻辑流水线转换为具体的资源流水线，供管理器启动。4. 管理器将资源流水线转换为 Kubeflow 流水线控制器支持的标准格式，并按照逻辑顺序启动流水线组件，如图3所示。5. 作业估算器解析搜索空间，并根据特定搜索策略（如 Hyperband 搜索和贝叶斯优化）生成搜索候选项。6. 上一步生成的候选项存储在中间件中，等待工作节点预留。7. 工作节点以互斥模式从中间件预留候选项，并启动特定的任务估算器，在任务边车的帮助下评估这些候选项。8. 任务边车负责衔接工作节点预留的候选项，并将其实例化为云原生工作负载（如作业和 PytorchJobs）以进行评估。这些工作负载完成后，系统会获取评估结果并报告给工作节点和作业估算器，以便生成下一轮候选项。9. 任务边车在训练与部署协同中也发挥着关键作用。训练过程完成后，训练端的任务边车会将训练后的模型传递到部署端，以便在部署环境中评估模型性能（如延迟和功耗）。此步骤是 OmniForce 多目标优化设计的重要组成部分。0. 在模型搜索过程中，顾问会全面分析元数据，并为用户提供有用的建议，例如数据集、算法或配置分析。

1

3 设计理念

3.1 面向以人为中心的人工智能的抽象与管理

作为一个以人为中心的人工智能平台，OmniForce 降低了机器学习运维流程的复杂性，让用户能够专注于最相关的工作，例如业务逻辑。具体而言，OmniForce 采用项目这一概念，对数据、特征、模型及其他元数据进行统一管理。一个项目包含构建特定业务解决方案所需的全部材料。在项目之下，生产环境中的人工智能所需的基本能力，例如数据隐私、版本控制、特征工程、模型训练、服务和监控，以及机器学习流水线自动化，都经过了良好的抽象，便于使用。本节将详细介绍 OmniForce 的数据、特征流水线和模型管理，以展示其人工智能生命周期管理能力。

3.1.1 数据管理

数据是构建成功的人工智能解决方案的基石。人们选择机器学习平台时，通常会关注以下几个核心问题：我的数据在这里安全吗？我的隐私会得到妥善保护吗？平台是否兼容多样化的数据，并能充分发挥数据价值？平台能否处理快速的数据迭代并及时更新方案？OmniForce 提供全面的数据管理服务，涵盖隐私、数据可访问性和版本管理、数据标注以及终身学习。

数据可访问性与版本管理 大多数人工智能平台开发者都认为，将用户连接到平台时，数据访问是主要问题。来自各行各业的多样化数据增加了数据标准化与治理的负担。为解决这一问题，OmniForce 开发了一组统一的数据访问与获取应用程序编程接口（API）。用户只需少量操作，即可通过数据访问 API 上传来自多个数据源的多样化数据，并将原始数据转换为标准格式。数据获取 API 抽象了数据源的复杂性，帮助开发者和数据科学家自由检索数据。
数据标准化后，OmniForce 使用 DVC [40], 进行数据版本控制，以处理大型文件、数据集、模型、配置和代码。用户上传数据时，OmniForce 会将所有媒体文件（表格数据文件除外）转换为元数据文件，其中保留了对原始数据的引用。用户可以自由地对元数据执行操作，并将更改保存为新版本。这些操作不会更改先前的元数据，且所有元数据均可追溯、可复用。这些操作主要包括以下内容：

1. 添加或修改标注。2. 数据筛选，例如从整个数据集中筛
选出 5 个类别。3. 合并两个版本的数据集。

值得注意的是，上传到 OmniForce 的所有数据都通过我们的隐私保护算法得到了安全托管。

主动学习与智能标注 主动学习 (AL) [41]是一项解决标注数据匮乏问题的实用技术。主动学习的核心问题是制定一种经济高效的数据排序策略，从庞大的未标注数据池中找出信息量最大的样本。在典型的主动学习循环中，模型会采用适当的排序策略来推荐样本。随后，专业标注员会评估这些推荐结果，并主动标注模型迫切需要的样本。满足标注预算要求后，模型会利用新生成的已标注数据集进行更新，通常数据集中的样本会更多。反过来，训练效果更好的模型将用于为下一轮推荐挑选更优质的数据。数据是推动人工智能浪潮的燃料。主动学习有助于收集高质量燃料，并快速构建高性能人工智能模型，这在工业领域具有重要意义。

主动学习（AL）将人类纳入机器学习（ML）生命周期，并通过人机协作快速构建高质量数据集。OmniForce 将这项能力与我们的其他服务相结合，设计出一项智能数据标注服务。该服务具有以下主要特征。

1. 丰富的数据标注支持最常见的人工智能任务。 2. 人机协同标注。OmniForce 支持在线和离线两种标注模式。在线模式下，用户可以实时纠正机器标注结果，并受益于 OmniForce 便捷的模型部署服务。离线模式下，用户可以一次性提交包含大量未标注数据的标注任务，并在自动标注流程完成后查看所有结果，这得益于 OmniForce 的批量推理服务。 3. 可自定义的机器标注能力。与市场上仅支持标注能力有限的预置模型的其他同类产品不同，OmniForce 得益于其众包设计，并

支持用户部署其标注模型。这些模型可以由用户直接上传，也可以由 OmniForce 生成。 4. 主动学习驱动的数据推荐服务。OmniForce 集成了许多实用且先进的主动学习算法，帮助用户找到信息量最大的 数据，并快速构建高质量数据集。

差分隐私 差分隐私（DP）技术最初被提出，旨在保障数据库查询操作的隐私 [42]。近年来，该技术也被拓展用于衡量算法的隐私保护程度[43, 44]。假设给定两个相邻数据集（S， S'），其中 S 和 S' 至多只相差一个样本，一个模型假设空间的任意子集 H ，以及一个算法 $\mathcal{A}$ 。 $\mathcal{A}$ 的差分隐私定义为将 $\mathcal{A}$ 应用于 S 和 S' 时，其输出假设的变化程度 $\mathcal{A}$。具体而言，（ε， δ）-差分隐私的数学定义为 $\log\left[\frac{\mathbb{P}_{\mathcal{A}(S)}(\quad \mathcal{A}\ S\in H}{\mathbb{P}_{\mathcal{A}(S')}\ \mathcal{A}\ S'\in H}-\delta\right]\leq\varepsilon$。这意味着，差分隐私（$\varepsilon$， δ）值较小的算法能够抵御个体训练样本的变化。因此，（ε， δ）的值表征差分隐私水平，即抵御差分攻击 的能力；这类攻击将个体样本用作探针，对机器学习算法发起攻击，进而通过输出假设的变化推断个体隐私。通常，迭代算法的隐私保护程度会随着迭代次数增加而降低，因为随着算法推进，泄露的信息量会不断累积 [43, 45, 46]。实践已证明，深度神经网络具有良好的泛化能力。然而，深度神经网络是一种过度参数化模型，现有统计学习理论难以对此作出解释 [47]。这引起了学界的关注，许多研究发现，差分隐私模型通常也具有有保证的泛化能力 [48, 49, 50, 51, 46]。

3.1.2 特征流水线管理

机器学习模型高度依赖输入数据的质量，而原始数据预处理通常是机器学习流水线中的关键环节。为帮助数据科学家和工程师高效、准确地将经验融入人工智能产品，OmniForce 提供探索性数据分析、交互式特征工程和高级 SQL 处理等数据处理能力。自定义特征流水线能够以自动化、低代码的方式减轻数据准备的负担，让用户能够更专注于数据收集、特征设计和模型选择方面的创新。

特征流水线版本控制 不同于数据版本控制，特征流水线版本控制侧重于追踪用户的特征工程操作。特征流水线包含一系列操作步骤。当用户在界面中对数据执行操作时，这些操作会作用于数据的一小部分，并以延迟加载的方式存储，以供后续处理。因此，我们在实时用户操作反馈与操作版本控制之间实现了优雅的平衡。基于历史操作版本，用户只需修改或合并操作步骤，就能轻松构建新的特征流水线。

定制化特征流水线 用户可以采用交互式特征工程，并编写 SQL 语句，基于人类经验生成特征，使人类辅助机器学习能够更快地取得令人满意的结果。具体而言，交互式特征工程分为以下几类：时序特征提取、单列计算、列间计算和特定条件处理，涵盖了数据处理过程中常用的原子操作。我们还支持用户通过自定义 SQL 语句对数据执行高级操作，显著提高其特征工程效率。此外，我们采用惰性处理策略来执行操作，以确保实时交互，使用户在处理大数据时点击操作更加流畅。

3.1.3 模型管理

本节介绍 OmniForce 中的模型管理技术，其包括三个主要部分：模型构建与版本管理、模型部署，以及生产环境中的监控。模型管理的核心理念是通过训练、部署、监控和优化构成的闭环，实现自动模型迭代，从而以敏捷的方式将模型发布到生产环境。

模型构建与版本管理 OmniForce 通过自动机器学习流水线以自动化方式构建模型，并对所有生成的模型进行版本管理。我们设计了一个三层模型管理架构，以展示模型之间的关系。在我们的设计中，顶层称为模型族，即强相关模型的集合。例如，在使用大模型执行下游任务时，OmniForce 会将衍生模型作为一个模型族进行管理。中间层称为模型注册项，用于指明当前正在使用的模型。底层称为模型资产，代表由自动机器学习流水线生成的原始模型。

模型部署 将训练后的模型部署到生产环境，是模型管理的重要组成部分；在模型即服务范式中也发挥着重要作用，因为模型部署后，许多开发者会利用人工智能能力，通过提供的 API 构建应用。

OmniForce 使用 KServe [52] 将模型部署到生产环境，并快速构建模型即服务架构。模型以无服务器形式运行在 Kubernetes 集群上，用户可通过提供的应用程序接口访问模型。OmniForce 采用众包代码规范进行设计，算法开发者按照这些规范编写代码所生成的模型可以快速部署到生产环境。部署后的模型将具备以下特性：

1. 支持滚动更新和回滚。2. 支持高并发和低延迟。3. 支持自动扩缩容，包括缩容至 0，以解决延迟敏感性与需求可预测性之间的冲突。4. 支持金丝雀发布、蓝绿发布和 A/B 测试等高级部署方式。5. 支持同时部署多个模型和级联模型。6. 支持多阶段条件模型推理。7. 支持模型解释。8. 支持模型监控。

模型监控 模型监控是机器学习生命周期中的一个运维阶段，位于模型部署之后。由于生产环境时刻都在变化，生产模型会导致性能损失，这种现象称为模型漂移。模型漂移有两种形式：

1. 数据漂移。数据漂移由数据分布变化引起。由于人工智能模型对给定的数据分布十分敏感，数据分布变化越剧烈，模型性能就会迅速下降。2. 概念漂移。概念漂移是指因因变量属性发生变化，导致模型不再适用于其所处环境的情况。

模型监控用于监测模型在当前环境中是否发生漂移。我们的模型监控系统可以实时监控生产环境中的模型，在发生模型漂移时发出警告，并利用近期收集的数据启动模型重新训练，从而迭代更新模型，使其性能保持在可接受的水平。模型重新训练生成的新模型会自动归档到原有模型族中，并以新版本更新。

3.1.4 信息可视化与洞察

作为以人为中心的人工智能的核心，可视化有助于用户理解其数据和算法。OmniForce 提供丰富的可视化信息，例如数据分布、搜索空间、超参数重要性等级和推理统计信息。用户可以通过分析这些信息，帮助 O mniForce 构建更多样化的模型，从而参与解决方案构建循环。例如，用户可以重新设计搜索空间或调整数据分布，以构建更优的模型。

超参数重要性分析 深度学习模型的性能高度依赖于其超参数设置。近年来，人们提出了一些现代优化方法，并成功实现了超参数的自动优化。然而，这些方法无法解释特定超参数如何影响最终的模型性能。为了向用户提供更易理解的超参数报告，OmniForce 采用一种名为方差函数分析（fANOVA）的超参数重要性评估方法，提取超参数与指标之间的关系 [53]。

给定模型指标及相应的超参数设置，fANOVA 通过拟合随机森林来近似超参数空间与性能空间之间的映射关系。随后，利用 fANOVA 评估每个超参数的重要性。此外，我们进行了一些改进，使该方法能够适用于具有层次依赖关系的超参数。

实时模型推理解释 目前，大多数深度模型都以黑盒方式运行，缺乏可解释性，阻碍了深度模型在许多领域的应用。遵循 HAML 的理念，OmniForce 提供实时模型解释服务，用于解释模型的输出。具体而言，OmniForce 构建了一个

可解释模型，并使用 KServe [52] 将该模型与目标模型一同部署。用户在线进行推理时，两个模型都会运行，而可解释模型则会分析目标模型的结果。借助解释信息，用户可以了解模型为何会输出这样的结果。此外，考虑到可解释方法应兼容平台支持的各种异构模型，我们主要选择黑盒模型解释算法，例如 Anchors [54]。该方法可以将神经网络模型的决策过程近似为基于规则的判别过程。其局部解释特性便于解释模型对单个样本的预测结果，并帮助用户直观地理解解释过程。

多目标模型性能评估

训练与部署综合指标 大多数现有自动化机器学习平台只关注训练阶段的模型性能，而忽略了部署阶段。OmniForce 借助任务边车弥合了这一差距。在训练阶段开发的模型会被发送到 Qualcomm A650[55] 和 NVIDIA Jetson 开发套件 [56] 等部署环境中，以对生产环境中的模型性能进行基准测试。系统会收集部署侧的指标，例如延迟和功耗，并据此计算综合奖励，用于下一轮模型搜索。

模型鲁棒性 模型鲁棒性是抵御外部扰动的能力，也是广泛应用人工智能的先决条件。鲁棒的模型能够产生稳定的输出，并适应生产环境中的各种情形。这一特性对某些应用至关重要，例如医疗保健、金融和安全。模型鲁棒性与底层数据分布密切相关。近期研究发现，轻微的数据偏移可能导致相关模型给出截然不同的结果，这表明当前的人工智能模型对数据过于敏感。在生产环境中，这些问题可能会意外发生，例如自然数据分布偏移；也可能是人为造成的，例如黑客攻击。OmniForce 提供模型鲁棒性评估服务，用于评估模型抵御噪声攻击的能力。用户可以根据模型的综合准确率和鲁棒性性能，选择要部署到生产环境中的模型。此外，OmniForce 还提供鲁棒性评估工具，其中包含两种常见的评估方法：模型对抗攻击评估 [57, 58] 和基于成员推理的模型隐私评估 [59, 46, 58]。前者评估模型在对抗样本下的鲁棒性，后者评估模型在成员推理场景下的数据隐私。

3.2 面向众包的流水线驱动训练、部署与协作

在 OmniForce 上上传并处理给定数据集后，要生成工业应用，用户需要上传自己的模型，或选择 OmniForce 格式化器推荐的众包模型。随后，OmniForce 通过自动化流水线组织从训练到部署的整个流程。尤其是，基于流水线驱动方法，可以自动完成大模型的适配与小型化。在搜索新模型时，工程师或数据科学家需要根据文档完成算法接口，并配置相应的搜索空间设计、运行设备（中央处理器、GPU）和优化指标。

3.2.1 应用算法接口

如清单 1 所示，我们在设计中实现了一种新的程序追踪器。要众包一个新模型，用户需要完成接口，并封装若干 Python 代码功能块。考虑到界面的易用性，复杂的消息收发操作均在内部实现。

清单 1：使用装饰器进行众包的 Python 用户应用算法接口。

```
来自全能工具导入估算器

类 YourEstimator(元类=ABCMeta): """这是 OmniForce 任务估算器的抽象基类。"""
@estimator.wrap定义 __初始化__(自身, 运行_训练周期: 整数, 数据_路径: 字符串, 是否
_训练器: 布尔值, ** 模型_参数, # 针对你的算法自定义的参数。) -> 无:
```

```
        s e l f . 模型    = YourModel ( * * 模型_args )

    @abstractmethod @estimator.wrap 定义 计算 _分数 ( 自身 )    -> 浮点数: """ 计算用于评估搜索
    模型的分数。 """ 返回 val score_定义 运行 ( 自身 )    -> 浮点数: """ OmniForce 通过此函数触发训
    练过程。 """ # 训练 自身 . 训练 ( 自身 . 训练 _加载器，self.run_训练周期 ) # 测试 val_分数    = 自
    身 . 计算 _分数 ( 自身 . 验证 _加载器 ) # 保存模型 如果 自身 . 是训练器:  _自身 . 保存 ( 自身 . 保存
    _目录 ) 返回 val score_
```

3.2.2 应用算法配置

搜索空间配置 在为工业应用生成和部署人工智能算法时，超参数调优是至关重要的一步。对于新的模型搜索任务，OmniForce 允许用户调优超参数并自定义搜索空间。在我们的配置工作流中，用户可以通过前端的便捷交互来配置搜索空间，也可以上传 YAML 文件来配置复杂的搜索空间。

不同任务的搜索空间定义可能大不相同。通常，深度学习模型的参数之间总是存在依赖关系。例如，在搜索神经网络的网络结构时，我们通常希望探索网络的深度和宽度。深度表示骨干网络中卷积块或层的数量，而宽度通常指每个块或层中的通道数量。因此，要采样的通道数组长度取决于采样得到的深度值。如果深度为三，则从搜索空间中采样三个通道值，如清单 2 所示。另一个复杂之处在于条件空间。例如，假设要在深度模型中搜索具有各种参数的不同类型的块。在这种情况下，只有在采样出相应块的类型后，才能确定要采样的参数。这些情况经常出现在各种算法的搜索空间中。因此，OmniForce 支持基于树和基于有向无环图的搜索空间采样规则，并将参数之间的依赖关系建模为图。

清单 2：多层搜索空间示例。

```
骨干网络块数量：_      _类型：整数范围：
  [ 2... 5 ]子模块：块_类型：类型：选项范围：
{resnet，transformer} 子模块：resnet：
数量_层：类型：整数范围：   [ 3... 7 ]子模块：
数量 通道：_类型：选项范围：{64，256}
transformer：mlp _扩展_比例：类型：选
项范围：{1，2，4，8}
```

推理配置 OmniForce 要求用户在众包批量推理阶段提供相应的配置，包括推理设备数量和推理资源使用量。这两个指标确保 OmniForce 能够为大量数据构建推理环境。

部署配置 在部署阶段，OmniForce 还要求用户提供适当的配置，以便将其模型快速部署到生产环境中。这些配置包括部署设备、部署资源使用量和单样本推理延迟。通过延迟，我们可以推断模型部署后的每秒查询数。

云边协同训练环境与要求 OmniForce 基于云边协同框架提出了一种新颖的自动化机器学习实践。通过安装 OmniForce 云边协同 Python 包并注册设备，用户能够根据性能和延迟指标，为其特定设备开发最合适的模型。此外，在训练大模型时，OmniForce 支持生产环境与超级计算环境之间的交互。

推理优化 由于缺乏部署考量，许多自动化机器学习框架仅考虑搜索侧所得模型的性能，而忽略模型在实际部署环境中的性能。针对 ARM [60] 和 ROCm [61], 等边缘设备，OmniForce 采用 TVM [62],这一深度学习编译器，使机器学习能够在各种环境中实现高性能运行。我们引入 TVM，通过任务边车连接训练环境与部署环境，并通过中继方法，在训练和部署测试期间完成自动化机器学习任务的协同搜索流程。这种全面的搜索策略有助于 OmniForce 找到在生产环境中表现出色的模型。OmniForce 还会针对特定设备使用不同的工具，例如用于 NVIDIA GPU 的 TensorRT [63] 和用于 Intel CPU 的 OpenVINO [64] 。此外，为了在不同机器学习框架之间转换模型，OmniForce 使用 ONNX [65] 作为中间格式。在这些场景中，我们会将 PyTorch 等高级框架中的模型转换为 ONNX 格式——这是一种常见的机器学习模型文件格式，然后进一步将其转换为 TensorRT 或 OpenVINO，以针对不同设备对深度学习推理进行定向优化。

3.2.3 应用算法注册

用户使用我们的标准接口实现应用算法并妥善准备好配置后，该算法便可封装为独立的 Docker 镜像，并注册到 OmniForce 众包系统中。我们的系统会进行冒烟测试，以验证算法的完整性，测试分为三个主要阶段: 搜索、推理和部署。

1. 搜索测试用于验证给定算法能否通过 OmniForce 的自动机器学习搜索策略进行搜索。此阶段会检查搜索空间配置。 2. 推理测试用于验证算法能否执行离线批量推理。此阶段会检查推理配置。 3. 部署测试用于验证算法能否部署，并生成相应的应用程序接口。此阶段会检查部署配置。

总体而言，OmniForce 会评估受测算法的四项能力：可搜索性、批量推理、云端部署和边缘侧优化。值得注意的是，上述测试不必全部通过，但建议尽可能多地通过，以便算法能应用于更多场景。例如，如果某个算法仅通过搜索测试，则只能用于超参数优化，无法部署到生产环境。通过冒烟测试后，用户的算法便会连同版本控制功能一起纳入 OmniForce 众包系统。Docker 镜像和配置都会进行版本控制，以便日后快速更新。

3.2.4 流水线生成

与第 2.1.7 节所述的自动化机器学习流水线类似，OmniForce 众包系统采用流水线驱动的方法，自动应用已提供和众包获取的算法。生成的流水线能够灵活支持不同任务。例如，用户可能需要一个由大模型赋能、可在资源有限的设备（如边缘设备）上运行的模型。OmniForce 可以采用包含两个步骤的流水线，即大模型适配和多目标优化，以生成高效模型。

3.3（大模型）即服务

借助大模型的强大能力，近年来人工智能社区的趋势揭示了规模扩展所带来的性能提升。大模型已在 ChatGPT [14] 和 Pathways [34] 等人工智能产品中取得成功。然而，大模型需要消耗大量内存和计算资源。在边缘设备上部署大模型，或将其应用于计算和内存资源有限的场景时，实用性可能会受到影响。尤其是在自动驾驶等要求低延迟的应用场景中，推理速度低的大模型无法保证在线预测的准确性，进而不可避免地引发安全问题。凭借强大的容量和先进的性能，大模型在大量任务中展现出较强的泛化能力。实际应用中，这通常伴随着高昂的预训练成本，因此这类模型不适用于工业场景中需要频繁进行模型适配和更新的应用。因此，在这些领域使用大模型时，需要通过适配和小型化流程来提升模型的迭代速度和推理速度，同时尽可能保留其性能。OmniForce 支持大模型适配和小型化技术，主要包括自动适配、过滤、知识蒸馏以及模型推理优化，如图 5 所示。

3.3.1 大模型支持

OmniForce 支持大模型技术。目前，人工智能面向各种行业和业务场景及其需求；例如，人们需要根据每个具体场景的硬件要求来设计神经网络架构、调整超参数并部署模型。大模型概念是通用人工智能的一项突破性技术，旨在解决人工智能应用的碎片化问题。OmniForce 在计算机视觉（3.5.2）和自然语言处理（3.5.3）任务中，以及自动适配（3.3.2）和小型化（3.3.3）方面，支持高效、统一的适配。用户可以通过众包界面搜索并训练自己的大模型，也可以使用 OmniForce 提供的大模型。通过简洁的应用程序接口和大模型服务调用模型，降低了用户的使用门槛，从而缩短了人工智能产品的开发与迭代周期。大模型工作流如图 6 所示。根据系统元知识和用户交互，OmniForce 会决定在实际应用中是否使用大模型或小模型，以及何时复用或更新模型。

3.3.2 适配

随着高性能加速器的发展，模型的规模呈指数级增长 [4]。由于缺乏带标签的数据资源，为了训练这类大模型，一些自监督方法取得了巨大成功，例如 MAE [66] 和掩码语言建模 [4]。随后，对由此得到的自监督预训练模型进行微调，使其适应特定任务和数据集。一些知名的预训练检查点，例如来自变换器的双向编码器表示（BERT） [67] 和 Clip [68], , 已广受欢迎，并推动了各种下游应用的发展。然而，在这一趋势下，面对实际的下游数据集，由于所需的计算和存储成本极高，从头开始训练模型，或针对每项任务频繁微调整个大模型，都十分困难。为解决这些问题，一些有效的参数调优方法引起了研究人员的关注。该领域最引人注目的三个分支是提示 [4], 、适配器 [69], 和低秩分解 [70], , 它们均受 OmniForce 支持。此外，在系统维护的知识库的帮助下，格式化器 2.1.6 可以自动创建合适的流水线，选择并搜索适用于下游任务和大模型的适配模块。完成适配后，大模型便可通过简单的应用程序接口为用户提供服务。

3.3.3 小型化

尽管大模型在许多任务中展现出强大的能力，但在某些场景下（例如真实的工业环境），需要在资源有限的设备（如物联网（IoT）边缘设备）上运行模型。在这种情况下，大模型的推理时间可能占据整个系统运行时间的很大一部分，导致响应时间过长。因此，开发人员更倾向于选择在性能与速度之间取得平衡的模型。OmniForce 支持两种借助大模型生成高效模型的方式：一种是在下游任务中精简高容量模型，另一种是在预训练（上游）阶段从大模型中迁移知识。

下游阶段的小型化过滤一直是减少大模型参数和计算成本的常用技术。目前的方法主要通过学习移除部分不重要的连接或通道来实现，这也称为神经网络剪枝或压缩 [71]。过滤变量可以用二进制掩码表示，其中 1 表示保留相应的连接或通道；否则就应将其剪枝。通过学习掩码来促使模型变得稀疏，从而减少参数量和计算量。过滤掉不重要的连接和通道后，通常会在原始训练任务上对较小的模型进行微调，以恢复

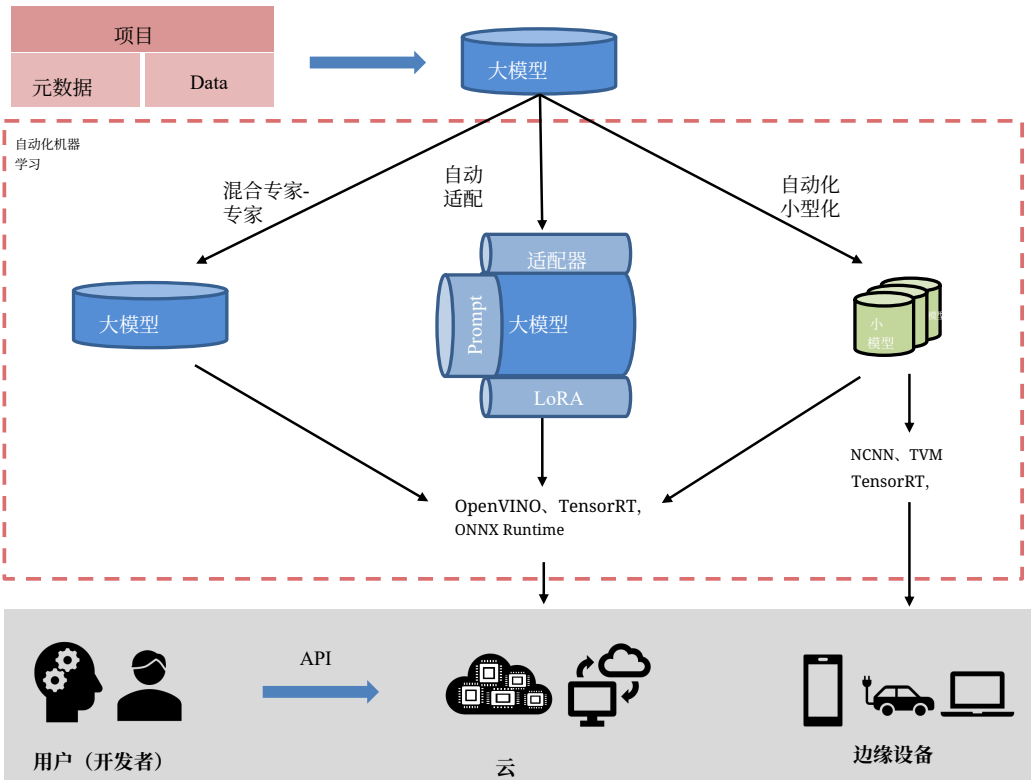


图 5：OmniForce 中大模型即服务范式示意图。给定输入元数据及相应数据，OmniForce 会自动创建自动化机器学习流水线，涵盖从算法选择到最终输出模型部署的全过程；输出模型可以是大模型，也可以是高效模型。大模型及其衍生的自适应模型可通过应用程序接口提供服务。为满足不同目标的需求，可通过组合一种或多种适配方法自动生成衍生适配模型。小型化方法可为边缘设备高效生成模型。OmniForce 将优化和加速后的模型作为应用程序接口提供给开发者，降低了使用人工智能技术的门槛。

性能下降。最近，针对 Transformer 的过滤研究通过学习稀疏掩码，仅保留总词元中的一小部分，以实现加速 [72]。知识蒸馏旨在将预训练模型（称为教师）的丰富知识蒸馏到一个新模型（称为学生）中 [73]。通常，教师模型是参数量很大的大模型。它的推理速度较慢，但性能出色。学生模型是通过人工设计或神经架构搜索（NAS）构建的小容量模型 [74, 75, 76, 77]。它运行速度快，但直接训练无法获得令人满意的性能。采用知识蒸馏技术 [78], , 我们将大模型用作教师，并利用从大模型最后一层获得的表征输出提供额外监督来训练小模型。经过知识蒸馏训练的小模型比其原始版本表现更好。在边缘设备上部署小模型时，我们能够兼顾低延迟和高性能。

上游阶段小型化小模型的知识吸收能力有限，难以从大型数据集中吸收知识；而使用海量数据训练的大模型，在下游任务中具有更强的迁移能力。然而，使用大型数据集预训练大模型成本高昂。在预训练阶段进行蒸馏是解决这一问题的一种新方法 [79, 80]。OmniForce 通过在预训练阶段采用知识蒸馏技术，无需从头开始训练大模型即可支持小型化方法。OmniForce 通过构建这样的流水线，促进大模型共享，并快速、频繁地生成迭代模型。

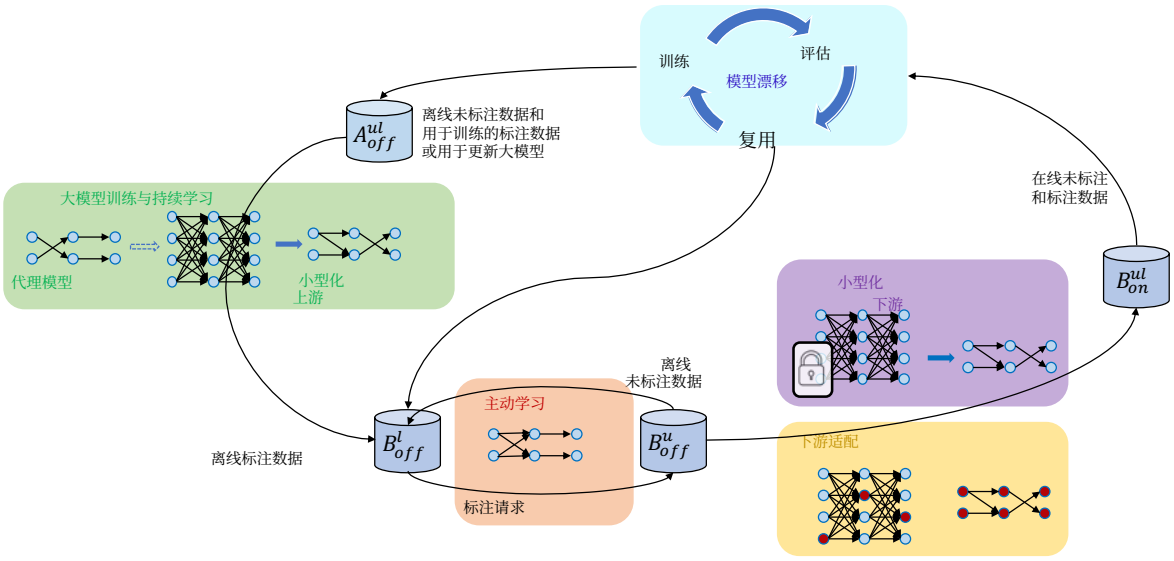


图 6：OmniForce 中大模型工作流示意图。在该工作流中，五个块与不同类型的数据相互作用，包括模型漂移检测器、大模型训练与更新、主动学习、小型化和适配。为高效训练和更新大模型，设计了代理模型。OmniForce 提供了优雅且一致的自动化机器学习技术，用于解决每个块中的复杂问题。

3.4 灵活搜索策略框架

搜索策略是自动化机器学习技术中最重要的部分之一，因为它会在给定搜索空间的情况下执行整个搜索过程。精心设计的搜索策略往往是提升效率和效能的关键。在本小节中，我们将介绍搜索策略的接口，以展示它在我们框架中的作用，以及 OmniForce 中灵活的搜索框架。

3.4.1 搜索策略接口

在我们的框架中，不附加任何其他限制时，搜索策略的接口由两个函数 generate_tasks 和 handle_rewards，以及一个搜索空间对象组成，如清单 3 所示。前一个函数根据观测值生成最适合给定搜索空间的超参数。当当前迭代中的所有任务均已完成或超时（见第 2.1.1 节）时，将调用后一个函数处理这些任务的奖励，并更新观测值以供下一次迭代使用。

清单 3：搜索策略接口。

```
类      搜索策略( 元类= ABCMeta) :
    """  这是 OmniForce 搜索策略的抽象基类。                    """

    def  绑定_搜索空间 (self,  搜索_space ) :
        self.搜索_space  =搜索_space

    @abstractmethod
    def  生成_任务 (self) :
        pass

    @abstractmethod
    def  处理_奖励 (self,          奖励 ) :
        pass
```

3.4.2 搜索策略框架

OmniForce 支持多种搜索策略，例如修订版 Hyperband [32], MF-NAS [33], 新颖的贝叶斯优化方法和进化方法。贝叶斯优化是一种样本高效方法，旨在寻找 $x^* = \arg\min_{x \in \chi} f(x)$，其中 f 是一个黑盒

函数，评估成本高昂，而 χ 是搜索空间或定义域 [81]。贝叶斯优化包含两个主要组成部分：用于对 f 的响应面建模的代理模型，以及用于权衡探索与利用的采集函数。尽管已有许多贝叶斯优化库问世，例如 Spearmint [82], GPyOpt [83],scikit-optimize [84],RoBO [85], ProBO [86], GPyTorch [87] 和 BoTorch [88], ，但它们都侧重于某个特定方面，目前还没有一个能够兼容所有方法的综合框架。例如，除 GPyOpt 外，上述方法都难以实现一些先进的批量贝叶斯优化方法，例如局部惩罚（LP） [89], 。
为适应日益先进的贝叶斯优化算法，OmniForce采用可组合的贝叶斯优化框架，该框架包含五组主要组件：代理模型、采集函数、采集函数优化器、候选生成器和建议器。与 BoTorch 类似，我们的框架基于 PyTorch [90] 构建，受益于自动微分和 GPU 加速；框架概览如图7所示。对于代理模型，我们实现了最流行的方法，例如高斯过程 [82], SMAC [91], 贝叶斯神经网络 [92, 93], ，以及一种用于解决离散优化问题的新方法（OF）。

我们提出的 OF 代理模型与 OF 采集函数相结合，在 NAS-Bench-201 [94]上实现了最先进的准确率。此外，针对多任务和高维优化问题，我们分别设计了 OF-Trans 和 OF-HD，并实现了一些其他贝叶斯优化库尚未提供的现有方法。在采集函数方面，除了期望改进（EI） [95], 、置信下界（LCB） [96] 和熵搜索（ES） [97] 等常用基础函数外，我们还支持 LP 和蒙特卡洛采集函数等强大的批量采集技术。为了提供一批新的查询，通常需要一些额外逻辑，这促使我们在贝叶斯优化框架中定义了一个名为建议器的新组件。在推理阶段，建议器接管所有其他组件，执行新查询的生成过程。例如，在大多数批量贝叶斯优化场景中，我们需要前瞻待处理任务，并使用蒙特卡洛样本 [82] 或常数谎言法 [98] 微调代理模型，以获取新的查询。为了优化采集函数，我们引入候选生成器，对足够多的候选项进行采样，作为 L-BFGS 优化器的起始点；这是贝叶斯优化领域最常见的选择。不同的采样方法可以轻松接入，且采样空间会随搜索空间变化，从而适配某些搜索空间收缩方法。

在训练阶段，我们使用观测值，通过 MCMC 或基于梯度的方法拟合代理模型。给定训练好的代理模型后，推荐器控制生成器生成候选项，并根据后验分布和采集函数优化器优化采集函数。随后，请求器建议的新查询被发送到数据库，并行工作器预留这些新任务以进行评估，然后将观测值发回，用于下一次迭代，直到预算耗尽。
总而言之，与其他现有的贝叶斯优化库相比，我们对贝叶斯优化框架的贡献主要体现在以下几个方面。

- 灵活且可组合。
- 针对多个问题的新方法。
- 自动微分和 GPU 加速。

3.5 广泛配置的应用算法

人工智能应用遍布工业生产和日常生活。人工智能的普及不仅体现在各种应用场景上，包括云边协同、虚拟现实集成和开放环境人工智能，也体现在所采用的应用算法种类繁多，例如表格数据分析与处理、自然语言处理、计算机视觉、人工智能生成内容和图表示学习。如上文所述，应用场景十分多样，本章主要聚焦于 OmniForce 中广泛提供的应用算法。

3.5.1 表格数据

表格数据分析是一项历史悠久的研究主题，通过特征组合和特征提取对结构化数据进行关联分析，并挖掘复杂的业务关系。该领域的常见任务包括时间序列分析[99][100], 异常检测[101][102],点击率预测[103][104], 等。

作为 HAML框架下的一种人工智能应用程序，OmniForce 鼓励用户将更多精力放在数据收集、自定义特征流水线和模型设计上，通过人类的经验提升性能。对于基础的探索性数据分析、数据清洗、空值填充、类别编码等繁琐但必要的流程，基于元学习的特征工程可自动完成上述任务，减轻人类的负担。通常，自动特征工程主要分为三个部分：清洗与填充、特征变换和特征组合。OmniForce 针对不同特征提供多种标准处理方法作为搜索空间，并为

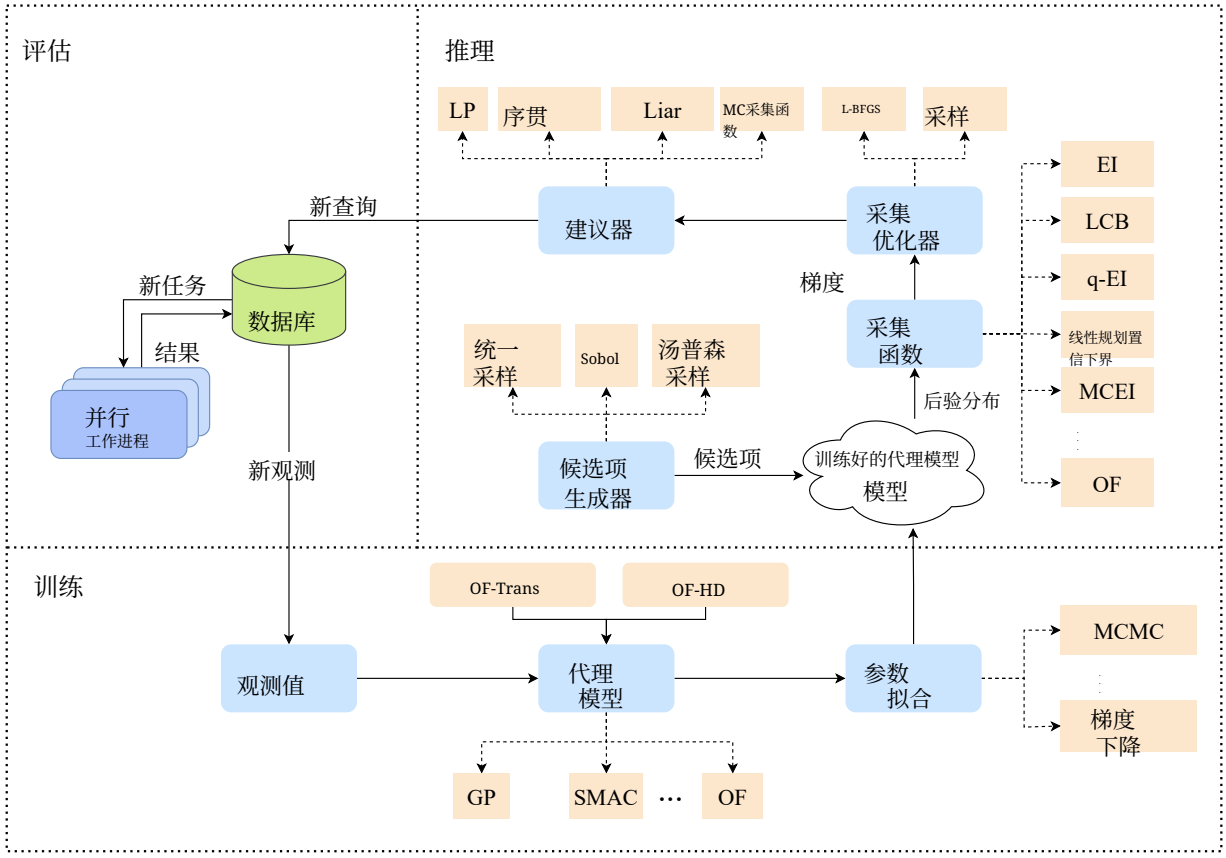


图7：OmniForce 的贝叶斯优化框架。该框架包含五个关键组件集：代理模型、采集函数、采集函数优化器、候选生成器和建议器。此外，我们将贝叶斯优化流程分为三个步骤。在训练阶段，我们通过梯度下降或 MCMC 方法训练代理模型，使其拟合观测值。随后，在推理阶段，我们利用训练好的代理模型和采集函数，建议将最优候选项作为新查询。在评估阶段，新查询发送至数据库后，我们使用 OmniForce 中的并行工作器评估这些候选项，并更新观测值以用于下一次迭代。

不同类型的数据寻找最合适的处理方法。例如，对于数值特征，空值填充方法包括均值、中位数、上四分位数和下四分位数；特征变换方法包括最大最小值、z分数和对数尺度归一化；特征组合方法包括乘法、除法及其他常见的数学变换。

基于决策树的集成模型，例如随机森林 [105] 和 LightGBM [106], , 由于计算成本低且可解释性强，一直深受业界青睐。近年来，神经网络领域的突破使深度学习模型备受关注，这些模型在推荐系统和点击率预测等任务中取得了令人瞩目的结果[103][107]。我们采用基于树的集成模型和深度学习模型构建搜索空间，以确保在各种高难度数据集上都能取得出色的性能。

大型表格数据在当前的大数据时代，机器学习算法被用于分析海量数据，服务于人们的生活和工业生产，并创造了巨大的价值[108]。因此，如何在大型表格数据集上高效训练模型，已成为 IT 公司面临的一项极具竞争力的课题。

与在 GPU 上训练深度模型相比，每个工作进程/任务估算器都要从磁盘读取数百 GB 的数据到内存中，这既冗余又非常耗时。因此，我们提出了一种名为“双共享内存”的系统化设计，以加快节点上大型表格数据的训练过程。一方面，我们将一份训练数据保存到 Linux系统的 /dev/shm 中，将其挂载到多个工作器容器中，并利用高速

访问内存快速加载数据。这样，同一节点上的任务估算器就能共享内存中的同一份数据，我们称之为外部共享内存。另一方面，由于自动特征工程，每个任务估算器使用的数据各不相同。为了避免特征工程所耗费的时间，我们启用了一种称为内部共享内存的共享机制。第一个训练轮次按批次进行特征工程和模型训练，并将处理后的数据写入容器，从而让后续训练轮次在文件加载和特征工程过程中节省时间。

这种用于数据加载的高效内存共享方法，可广泛应用于多层感知器（MLP）等深度学习模型。此外，对于内存较小的节点，调度器可以灵活地仅使用内部或外部共享内存来加速训练过程。

时间序列数据 作为最具挑战性的表格数据问题之一，时间序列预测有着众多应用，一直是人工智能社区尝试借助机器学习和深度学习[109][110]解决的主要研究领域之一。

秉持以人为本的理念，我们开放了时间序列区间和预测时长等交互功能。用户可以根据自身经验，结合数据特征和业务场景创建定制化模型。同样，自动化特征工程会清理并填充数据，识别并分组多序列案例，还会生成滑动时间窗口和滞后项等时间特征。

搜索空间包括传统自回归模型[111] 基于树的模型[9][106], 以及近期提出的基于Transformer的模型 [112]。第2.1.6节中的格式化器会根据可用的历史信息，针对统计特性不同的数据集为各模型分配权重，以确保获得最佳性能。

3.5.2 计算机视觉

计算机视觉是人工智能的主要研究领域，旨在提取、处理并理解数字图像和视频中包含的信息。传统的计算机视觉算法，例如支持向量机（SVM） [113]„ 通常通过手工特征工程来解决特定任务。得益于数据驱动的学习方式，深度学习算法在各类计算机视觉任务中取得了巨大成功。一方面，端到端训练和推理策略使深度学习模型能够轻松适配不同的计算机视觉任务。另一方面，随着深度神经网络架构的发展（例如 AlexNet [114] 和 ResNet [1]），深度学习模型的能力迅速提升，越来越多的深度学习模型被应用于实际场景。

尽管深度学习模型，尤其是深度卷积神经网络（CNN），在计算机视觉任务中展现出令人瞩目的性能，但要使这些模型成为稳健且通用的计算机视觉模型，仍存在较大差距。近年来，受计算资源快速增长和海量视觉数据的推动，超深度模型日益受到计算机视觉领域的关注 [115, 116, 117, 118, 119, 120]。凭借强大的学习能力，超深度模型能够学习通用且具有判别性的表示。此外，其强大的建模能力也使其能够利用少量标注数据适应新场景 [117, 118, 120]。这种能力对于实际应用至关重要。例如，我们可以以更低的成本验证技术方案的可行性，从而加快开发周期。为帮助用户快速开发并部署适用于特定应用的模型，我们针对不同的视觉任务提供了多种超深度视觉模型，例如二维/三维目标检测 [121, 122], 、语义分割 [123, 124], 、道路与车道检测 [125, 126], 、图像抠图 [127, 128], 、关键点检测 [129, 118],、场景文本检测与识别 [130, 131, 132]。用户只需提供任务描述和一些标注样本，即可从平台轻松获得高性能的任务专用模型。此外，超深度模型通常需要大量资源才能部署，而用户更青睐参数更少、推理效率更高的轻量级模型。为满足这些需求，我们提供了多种压缩和加速此类模型的解决方案。具体而言，我们开发了高效的模型压缩技术，例如基于量化和基于剪枝的技术，以精简并加速模型。此外，平台依托一个经过设计、包含各种运算、块、损失函数等的搜索空间，可以自动搜索轻量级模型，并在超深度模型的指导下通过知识蒸馏提升所搜索模型的性能。

3.5.3 自然语言处理

自然语言处理（NLP）是人工智能的主要分支之一，旨在利用计算机自动处理人类语言（包括口语和书面语），相关任务通常被归类为认知智能。自然语言处理源自多个学科，例如计算机科学、人工智能和语言学。自然语言处理基本上可分为两类：1）自然语言理解 （NLU）。自然语言理解探究使计算机能够

理解人类用户提供的文本指令。最常见的自然语言理解任务包括文本分类 [133],情感分析 [134, 135, 136], 问答 [137, 138], 命名实体识别 [139, 140], 等。2）自然语言生成 （NLG）。自然语言生成使计算机能够理解用户以英语和中文等人类语言输入的内容后，生成文本输出。常见的自然语言生成任务包括机器翻译 [141, 142, 143, 144, 145],摘要 [146, 147], 对话 [148, 149],等。
尽管基于深度学习的自然语言处理模型已在一系列任务中展现出良好的性能，但现有学习范式缺乏充分利用众多任务和数据的能力，因此带来了一系列问题，例如针对不同任务重复训练模型、数据孤岛问题、部署复杂，以及在低资源场景下学习能力较差。为了帮助用户高效、有效地为不同应用程序开发和部署模型，我们的系统由一系列基础模型组成，包括用于通用语言理解与生成的超模型 [150, 151, 152], 用于跨语言生成的超模型 [153, 154, 155, 156, 157], 以及它们的高效调优与蒸馏版本 [158], 基于蒸馏的提示学习 [159] 和 PESF-KD [160], 。用户只需提供任务描述和少量标注样本，即可通过平台获取由我们内置基础模型调优而成的高性能模型或小型适配器。值得一提的是，借助基于蒸馏的提示学习这一高效调优策略，我们的服务端基础模型只需微调 0.5% 的参数；与原始基础模型的参数规模相当，同时还能达到相当甚至更优的性能。通过这种方式，用户只需部署自己的小型提示或适配器，即可高效、私密地对其进行增量更新，而无需将宝贵的数据上传至我们的平台。

3.5.4 多模态深度生成模型的学习

人工智能和机器学习的主要目标之一，是学习和操纵真实世界数据的高维概率分布 [161]。通过这样做，这些技术能够从数据中提取有价值的见解，并将其用于改进许多相关任务 [162]。近年来，深度生成模型已成为学习数据分布的强大手段。这些模型包括生成对抗网络（GANs） [163, 164, 165, 166], 、向量量化变分自编码器（VQ-VAEs） [167, 168], 、自回归模型 [169, 170], 和扩散模型 [171, 172], , 它们在广泛的应用中展现出令人瞩目的能力。通过学习生成数据的底层概率分布，研究人员可以深入了解数据生成过程的内在机制。此外，经过良好训练的生成模型还可广泛用于内容生成相关任务。

我们的系统包含一系列内置的深度生成模型，旨在提升生成内容的真实感，并提供能够处理通用内容生成任务的生成模型。通过学习不同模态之间的特征对齐，这些模型能够生成更多样、质量更高的内容。为实现这些目标，我们开发了一套先进算法，可用于训练这些生成模型，并将其应用于各种实际场景。这些算法旨在提升生成模型在视觉概念探索、生成可控性和内容多样性等任务中的性能。总体而言，我们的内置模型和算法已应用于多项人工智能生成内容（AIGC）任务，包括视觉语言生成 [173], 复杂场景生成 [174], 人像动画 [175, 176], 3D 对象渲染 [177], 等。我们相信，我们的系统有望革新内容生成领域，并为未来 AIGC 技术的新型创新应用铺平道路。

3.5.5 图表示学习

图数据无处不在，例如社交图、知识图谱和蛋白质结构。通常，图由节点以及连接节点的边组成。句子和图像也可以用图来表示：句子中的词语和图像块可以视为节点，而节点之间的连接则代表边。鉴于图无处不在，分析图数据并学习图表示，对于解决节点级、边级和图级应用问题至关重要。

图神经网络（GNN）是在图域上运行的神经网络。我们的系统提供多种内置的图神经网络（GNN），例如图卷积网络（GCN） [178], 图注意力网络（GAT） [179], 和图Transformer [180]。例如，我们的系统提供了一种高效的图Transformer，即也就是Gapformer，它将图池化深度融入图Transformer。通过使用Gapformer，可以最大限度地减小存在多个无关节点所带来的负面影响，同时保留长程信息，并将消息传递的二次复杂度降低为线性复杂度。此外，我们的系统开发了许多多样化的插件模块，以提升图神经网络（GNN）的能力。例如，我们的系统采用SkipNode [181], , 在每个卷积层中对图节点进行采样，从而跳过卷积操作，缓解基于GCN的网络中的过平滑和梯度消失问题。我们的系统还采用了一种用于图池化的即插即用方案MID，该方案具有多维评分空间和两种评分操作，用于探索节点特征和图

中结构的多样性，以获得改进的图级表示。此外，我们的系统还提供典型的图应用，包括用于学习有符号网络嵌入的网络结构 [182],用于高光谱图像分类的多层次学习GCN [183], 以及用于场景图生成的异配网络 [184]。

4 个特征

助力开发与部署协同，使用更轻松 OmniForce 致力于构建适用于生产环境的模型，并设计了一套真正的开发-部署协同模型构建框架。与许多仅能借助 CI/CD 流水线，以敏捷方式在生产环境中发布模型的人工智能平台不同，OmniForce 打通开发环境与部署环境，并在搜索和训练阶段采用多目标优化方法，构建更加实用、通用的模型。OmniForce 旨在让机器学习专业知识有限的开发人员和数据科学家只需点击几下，即可部署自己的模型服务。

开放环境适配的工业级可用性 与传统自动化机器学习平台不同，我们提出 OmniForce 来研究开放环境与开环问题，因为在实践中，学习过程中数据、标签、特征、模型、评估和指标通常都会发生变化 [185]。我们的直觉是，需要让人在回路中参与进来，借助人类知识并增强人类能力；基于图 1 所示的顺畅交互，实现 HAML 的目标。

云原生生产环境与（大型）模型即服务 越来越多的大规模系统通过容器构建，并配备容器编排系统来管理所有组件，因为这类系统通常具有资源利用率高、隔离性强和支持持续交付等优势。OmniForce 基于 Kubernetes 设计，这意味着 OmniForce 是一个完全云原生的自动机器学习系统，并可利用众多优秀的云原生工具。基于 Kubernetes 和 Kubeflow，OmniForce 支持多租户、高可扩展性、强大的灾难恢复能力，以及将训练后的模型自动转换为部署服务。

众包 OmniForce 支持大规模算法，并扩展应用算法集合。借助一个在工程师群体中广泛使用的系统，过去可以通过众包应用算法，直接在新数据集上搜索并部署新任务。为启发众包这一理念，我们从数据、标签、模型、算法和搜索空间的机器学习版本管理、流水线驱动的开发与部署协作、灵活搜索策略框架，以及种类丰富的应用算法入手，其中包括基于超深（大型）模型的方法。

5 评估

本节包含三个部分。第一小节简要介绍工业元宇宙带来的创新。第二部分展示以人为中心的现实工业元宇宙场景中的一组应用案例，说明如何通过扩展现实仿真、连续数据采集、众包和云边协同的实际运行来解决开环人工智能问题。最后，我们通过有关可扩展性、容错性、搜索性能、算法性能以及以人为中心的自动化机器学习实践的实验，展示 OmniForce 的能力。

5.1 工业元宇宙的创新

标准装配线已采用人工智能技术来提高单个工厂的生产效率。近期工业元宇宙技术的发展将进一步推动人工智能的应用，并改变整个供应链的结构。工业元宇宙已经形成一种新的制造范式，促进工厂之间的协作，并加速产业链上下游部分之间的连接。例如，在传统的消费者对制造商（C2M）商业模式中，制造商会在产品最终发布前生产小批量的市场测试批次，然后根据市场反馈改进产品设计，或在产品畅销时扩大生产。然而，元宇宙的出现将改变传统消费者对制造商（C2M）商业模式的结构。

随着虚拟世界中的沉浸式体验不断发展，实体经济中的数字内容将作为主要数据，用于构建数字世界。技术重塑了现有产业链的形态。元宇宙让消费者能够在产品设计阶段体验产品并做出购买决策，使制造商在这一阶段获得更详细的反馈。他们会改进产品

设计，并根据客户反馈甚至销售数字内容服务（新的收入来源）。此外，元宇宙的出现将导致供需结构发生变化。它将通过消除中间环节，缩短制造商与客户之间的距离。

下一部分将介绍一个案例，展示如何在工业元宇宙背景下利用 OmniForce 帮助制造自动驾驶车辆。在设计阶段，会考虑用户体验和反馈。在元宇宙的开发和制造阶段，OmniForce 利用云边协同技术快速进行产品迭代，并根据反馈改进产品的性能和用户体验。此外，OmniForce 还协助整个供应链服务中的自动订单拆分、自动下单、供应链寻源和智能备货。

5.2 工业供应链与工业元宇宙案例

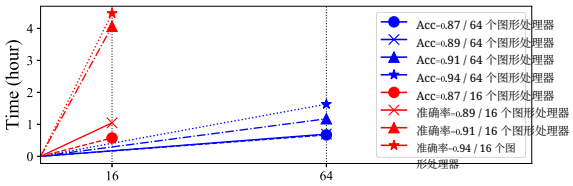
设计 在本案例中，制造商旨在设计适用于不同场景的自动驾驶汽车。例如，不同城市的居民可能偏好不同颜色和图案的汽车。一些小型车辆可能比大型卡车更适合配送包裹和在社区内行驶。这项任务可以借助 OmniForce 的人工智能生成内容能力。在这一生成任务中，客户希望该工具能够自动生成汽车模型。将汽车模型数据集输入系统后，OmniForce 会输出一个用于生成内容的模型。生成的汽车模型可通过扩展现实系统进行模拟和检查，并与客户互动。这一步缩短了消费者反馈所需的流程，也是实施工业元宇宙设计方案的一种方式，能够节省时间和预算资源。如果所得结果不符合消费者的要求，用户可以在 OmniForce 上调整模型的奖励，从而再次触发模型生产循环，直至结果可用于下一步。

开发与制造在这个案例中，客户是一家先进的配送公司，配备了由人工智能赋能的新技术设备。他们希望在智能车辆上运行一组模型，以适应不同场景；在这一真实的工业场景中，涉及大量设备，且这些设备对部署（推理）有不同要求。例如，他们需要三个模型，分别满足不同要求：

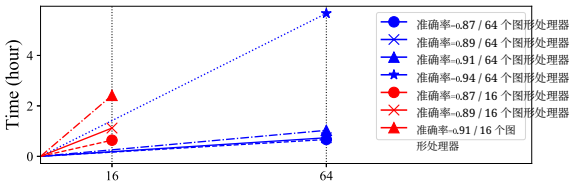
- 一辆在城市之间行驶的卡车；
- 一辆在城市主干道上行驶的皮卡；
- 以及一辆穿梭于社区和建筑物之间的微型配送货车；

具体来说，在城市之间的高速公路上行驶时，卡车在路况相对良好的情况下，车速可达每小时 80 公里或更高。因此，构建的模型需要快速响应卡车前方的障碍物。在主干道上行驶时，部署在皮卡上的模型应能应对复杂的路况，例如行人、自行车、摩托车和宠物。相比之下，微型配送车辆通常以较低的速度穿梭于社区和建筑物之间，但在极端天气下也可能在结冰的道路上行驶。
为应对这些挑战，OmniForce 持续应用涵盖数据收集和模型搜索的自动化流水线，从开发、部署和维护阶段中提取价值。要收集数据，我们可以从公共数据集或现实世界中收集一组示例图像。通常，我们需要一些数据要求，以便人工智能模型能够充分学习。其中一项要求是，道路上的场景以及汽车、行人和树木的类型应具有多样性。将数据上传至 OmniForce 并获得生成的数据集后，我们可能会发现，与其他类型的汽车或场景相比，图像中的“轿车”或“晴天的城市道路”出现频率高得多，从而导致长尾分布问题。因此，系统会触发以人为中心的开环，OmniForce 会提示用户持续收集其他类型的数据，例如跑车、厢式货车和豪华轿车。OmniForce 还可以生成不同场景类型的图像，其中包括高速公路、自行车拥挤的道路、摩托车、雨天的湿滑道路、树木的影子以及社区中的儿童。随着新数据的获取，OmniForce 会更新数据版本，以满足设定的平衡要求。为了在不同约束条件下搜索模型，OmniForce 支持云边协同，帮助机器学习专业知识有限的开发者和数据科学家调整准确率、召回率、功耗和延迟等多个目标，并修改评估和奖励，从而自动生成各种模型。生成模型后，我们可以在仿真系统中验证模型；在该系统中，数据由扩展现实应用中的驾驶员收集，随后还会在一些小规模场景中测试模型，再进行部署。当发生模型漂移时，OmniForce 会让人在回路中参与检查数据、按需收集数据、重新搜索或重新训练模型，并更新模型版本。

供应链服务 在工业元宇宙中，智能仓储和备货能够实现快速供应链，并降低原材料预算和生产时间。也就是说，人们可以利用自动化机器学习技术获得高效的供应链服务，包括订购、存储、运输和营销。
OmniForce 支持对表格数据和时间序列数据进行分析和预测。使用 OmniForce 自动生成模型后，



(a) 在 CIFAR-10 上获得的进化算法结果



(b) 在 CIFAR-10 上获得的 Hyperband 结果

图 8：OmniForce 使用进化算法和 Hyperband 时的可扩展性。红线表示使用 64 个图形处理器进行的实验，蓝线表示使用 16 个图形处理器进行的实验。不同线型表示实验得到的精度结果各不相同。与使用 16 个图形处理器的实验相比，使用 64 个图形处理器的实验用时更短即可达到相同的准确率。

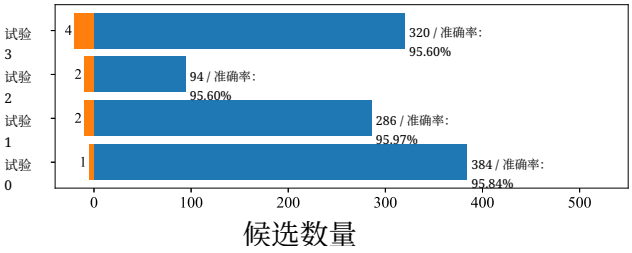


图 9：关于容错性的四次试验。这些试验在不同容错水平下取得了相近的性能。黄色柱表示有缺陷的候选项，蓝色柱表示存活的候选项。我们在柱旁标出了存活候选项的数量及相应的准确率。

可以使用元胞自动机和小规模测试等仿真系统，以相对较低的成本验证模型。根据对已搜索到的架构和超参数的可视化结果及说明，OmniForce 提供便捷的界面，让人在回路中参与其中，引导他们调整搜索空间、完善指标、收集组合数据、调整仿真系统、触发下一个模型生产周期，并更新数据和模型的版本。在此过程中，可以通过众包知识库实现算法。通过规范数据抽象流程、应用算法和搜索空间，OmniForce 可以轻松集成并复用应用算法和搜索空间。此外，用户可以通过 OmniForce 学习机器学习流水线相关知识。

5.3 实证结果

可扩展性与容错性 OmniForce 支持由调度器分配的可扩展搜索任务。如图 8 所示，实验在同一数据集上使用不同规模的可扩展资源和不同参数运行，并采用两种搜索算法：进化算法 [186] 和 Hyperband [187]。在大规模测试案例中，规模更大的组拥有更多计算资源，因此能以更短的时间达到与资源较少的组相同的准确率。例如，使用 64 个 GPU 进行的实验约一小时即可达到 90% 以上的测试准确率。相比之下，使用 16 个 GPU 进行的实验则需要四个多小时才能达到相同的准确率。因此，在图 8 (b) 所示的时间限制内，使用 16 个 GPU 训练的模型无法获得良好的性能。此外，使用 16 个 GPU 和 64 个 GPU 进行的实验达到较低准确率所需的时间相近，因为对于搜索任务而言，达到如此低的性能水平并不困难。

可以看出，图 9 中不同的故障率对性能影响很小。四项实验的性能相当，但候选项故障率各不相同。在这些实验中，我们添加扰动以杀死一些仍存活的候选项。借助我们精心设计的半同步方案，作业可以在有限数量的候选项失效时继续运行，也可以在搜索过程中暂停并重新启动。

搜索性能 基于我们提出的贝叶斯优化框架，我们设计了一种适用于离散优化问题的新型贝叶斯优化方法。我们在一个热门基准（NAS-Bench-201 [94]）上，将我们的方法与各种神经架构搜索算法和贝叶斯优化方法进行了比较。该基准包含 15625 个网络架构，以及它们在三个视觉

表1：NAS-Bench-201 上的分类 Top-1 测试准确率（%）。第一个块展示基于参数共享的 NAS 方法的结果。第二个块展示非参数共享算法和各种贝叶斯优化方法的结果。第三个块展示我们提出的贝叶斯优化方法的结果。† 符号表示相应方法的性能直接取自 NAS-Bench-201 [94]。

方法	CIFAR-10		CIFAR-100		ImageNet-16-120	
	验证集	test	有效	test	有效	test
RSPS† [188]	80.42±3.58	84.07±3.61	52.12±5.55	52.31±5.77	27.22±3.24	26.28±3.09
DARTS-V1† [189]	39.77±0.00	54.30±0.00	15.03±0.00	15.61±0.00	16.43±0.00	16.32±0.00
DARTS-V2† [189]	39.77±0.00	54.30±0.00	15.03±0.00	15.61±0.00	16.43±0.00	16.32±0.00
GDAS† [190]	89.89±0.08	93.61±0.09	71.34±0.04	70.70±0.30	41.59±1.33	41.71±0.98
SETN† [191]	84.04±0.28	87.64±0.00	58.86±0.06	59.05±0.24	33.06±0.02	32.52±0.21
ENAS† [74]	37.51±3.19	53.89±0.58	13.37±2.35	13.96±2.33	15.06±1.95	14.84±2.10
GibbsNAS† [192]	90.02±0.60	92.72±0.60	68.88±1.43	69.20±1.40	42.31±1.69	42.08±1.95
REA† [193]	91.19±0.31	93.92±0.30	71.81±1.12	71.84±0.99	45.15±0.89	45.54±1.03
RS† [16]	90.93±0.36	93.70±0.36	70.93±1.09	71.04±1.07	44.45±1.10	44.57±1.25
REINFORCE† [194]	91.09±0.37	93.85±0.37	71.61±1.12	71.71±1.09	45.05±1.02	45.24±1.18
BOHB† [195]	90.82±0.53	93.61±0.52	70.74±1.29	70.85±1.28	44.26±1.36	44.42±1.49
TPE [196]	91.30±0.18	94.07±0.17	71.93±0.89	72.08±0.83	45.71±0.68	45.94±0.83
SMAC [91]	91.23±0.21	94.05±0.23	72.17±0.61	72.21±0.76	45.51±0.33	46.08±0.74
BOHAMIANN [93]	91.36±0.16	94.13±0.23	72.36±0.82	72.38±0.81	45.93±0.66	46.18±0.60
OF-BO	**91.52±0.05**	**94.35±0.03**	**73.21±0.29**	**73.25±0.18**	**46.27±0.36**	**46.54±0.19**

分类数据集上的评估结果。按照 [94], 的设置，我们在训练 12 个训练轮次后，在 CIFAR-10 验证集上进行搜索，然后直接查阅其他数据集中的评估结果。我们使用 12 个初始点，将这些贝叶斯优化方法运行 80 次迭代，并报告 10 次重复运行中搜索过程中遇到的最佳观测值的均值和标准差。表1显示，我们的方法在所有三个数据集上均取得了最佳性能。

算法性能在此，我们展示 OmniForce 所提供模型的性能。首先，我们展示最先进的扩展型视觉模型（ViTAE [197, 117]），并在表 2 中将其与其他基于 Transformer 的深度模型进行比较。我们发现，这些模型的参数量超过一亿，能够实现出色的性能。凭借强大的容量，这些模型只需使用少量标注数据进行微调，便能在新场景中表现出色。因此，用户可以验证新方法并快速部署模型。

表 2：扩展后的 ViTAE 模型在 ImageNet1K 数据集上的性能。† 表示使用 ImageNet22K，以 224×224 分辨率对模型进一步微调 90 个训练周期。

	参数量	Test size	ImageNet Top-1	Real Top-1
Swin-L† [198]	197 M	384	87.3	90.0
SwinV2-L† [199]	197 M	384	87.7	-
CoAtNet-4† [200]	275 M	384	87.9	-
CvT-W24† [201]	277 M	384	87.7	-
ViT-L* [66]	304 M	224	85.5	90.1
ViT-L [202]	304 M	224	85.7	-
ViTAE-L [117]	311 M	224	86.0	90.3
ViTAE-L† [117]	311 M	224	87.5	90.8
ViTAE-L† [117]	311 M	384	88.3	91.1
SwinV2-H [203]	658 M	224	85.7	-
SwinV2-H [203]	658 M	512	87.1	-
ViTAE-H [117]	644 M	224	86.9	90.6
ViTAE-H [117]	644 M	512	87.8	91.2
ViTAE-H† [117]	644 M	224	88.0	90.7
ViTAE-H† [117]	644 M	448	88.5	90.8

此外，我们还报告了内置平台模型在自然语言处理方面的性能，包括自然语言理解（NLU，即 GLUE基准上的性能）和自然语言生成（NLG，即机器翻译任务）。表 3 展示了单个模型在 9 项自然语言理解任务上取得的对比结果，表明我们的方法可以利用任何现有的微调提示，获得更好的迁移学习性能。重要的是，借助我们的内置高效方法，只需调优原始参数的 0.5%，就能达到甚至超过完整模型调优的性能；这对于用户仅使用少量（甚至零）标注数据进行低资源、低成本训练和部署至关重要。图 10 展示了我们的 Vega-MT 翻译模型的性能。我们参与了 10 项共享任务，包括中文↔英语（Zh↔En）、德语↔英语（De↔En）、捷克语↔英语（Cs↔En）、俄语↔英语（Ru↔En）和日语↔英语（Ja↔En）。凭借我们的多语言基础模型，我们在 BLEU分数方面获得了 7 项冠军、2 项亚军和 1 项第三名。搭载 Vega-MT 的平台能够帮助用户轻松理解和生成各种跨语言内容。值得注意的是，通过切换到我们开发的非自回归生成算法 [204, 205, 206]，平台还可以加速语言生成过程。

表 3：基于基础语言模型的跨任务高效提示迁移实验部分结果（%）。请注意，我们的方法与模型无关，因此可用于增强任何基础模型。此处以 BERT-Large 为例。在 (a) 和 (b) 组中，每个单元格表示将提示从源任务（行）迁移至对应目标任务（列）所取得的目标任务性能。“AVG.” 表示所有目标任务的平均性能。值得注意的是，正向提示迁移结果以粗体标出，下标中的数字表示相对提升幅度。

方法	CB	COPA	WSC	RTE	WIC	CoLA	MRPC	STSB	Conll 04	AVG.
模型调优	94.6	69.0	68.3	75.8	74.9	60.6	88.0	90.0	85.6	78.5
提示调优	87.5	76.0	64.4	76.2	66.9	63.8	86.8	90.5	85.5	77.5
(a) 使用原始提示迁移方法进行迁移										
MNLI	**96.4**	71.0	**67.3**	**80.9**	66.5	58.9	**88.2**	**91.0**	83.0	**78.1**
QNLI	**89.3**	76.0	**65.4**	76.2	**70.4**	63.7	**88.5**	**90.7**	83.5	**78.2**
记录	78.6	63.0	**65.4**	53.8	51.7	0.0	77.7	85.0	82.7	62.0
SQuAD	87.5	74.0	**66.3**	71.8	51.7	6.0	**87.3**	89.3	82.5	68.5
CoNLL03	73.2	64.0	63.5	60.3	51.9	0.0	71.3	16.4	84.8	53.9
Ontonotes	78.6	65.0	**66.3**	56.7	54.1	59.3	82.4	84.5	**86.1**	70.3
CoNLL05	87.5	65.0	64.4	69.3	**68.3**	61.3	**88.7**	88.4	83.8	75.2
CoNLL12	**89.3**	62.0	**67.3**	63.2	**67.4**	58.7	90.4	88.5	83.6	74.5
SST2	**92.9**	74.0	64.4	71.8	66.8	60.1	**87.0**	89.6	84.3	76.8
（b）使用我们的内置高效方法进行迁移										
MNLI	**92.9**	**77.0**	**67.3**	**78.0**	**68.8**	**66.3**	**88.5**	**90.6**	85.4	**79.4**$_{1.3}$
QNLI	**92.9**	**77.0**	**66.3**	**77.3**	**70.8**	63.9	**87.5**	**90.8**	**86.6**	**79.2**$_{1.0}$
记录	87.5	76.0	**66.3**	**77.3**	**68.5**	62.4	**87.5**	**90.7**	84.9	**77.9**$_{15.9}$
SQuAD	**89.3**	75.0	**66.3**	75.5	**69.3**	63.1	**87.3**	88.9	**85.7**	**77.8**$_{9.3}$
CoNLL03	**91.1**	72.0	**68.3**	**76.9**	**67.4**	63.6	86.5	**90.6**	**85.6**	**78.0**$_{24.1}$
Ontonotes	**89.3**	74.0	**66.3**	76.2	**69.1**	**64.2**	**88.0**	**90.8**	**85.7**	**78.2**$_{7.8}$
CoNLL05	87.5	**79.0**	**65.4**	**77.6**	**69.6**	**63.7**	**87.5**	**90.8**	84.8	**78.4**$_{3.2}$
CoNLL12	87.5	76.0	**66.3**	74.4	**68.5**	**63.7**	**87.5**	**90.8**	85.0	**77.7**$_{3.3}$
SST2	**92.9**	**77.0**	**68.3**	**76.5**	**70.1**	**64.8**	**88.5**	**90.7**	**86.3**	**79.5**$_{2.7}$

最后，我们报告了使用 OmniForce 开展的复杂场景生成实验的比较结果。参与比较的方法在 COCO-stuff 和视觉基因组数据集上取得的定量结果见表4。为确保比较公平，我们采用了这些方法官方发布的预训练模型，或其论文中正式报告的分数。与基于 CNN 和基于 Transformer 的复杂场景生成方法相比，TwFA [207] 在所有指标上均取得了显著提升。此外，由于我们采用了与基于 Transformer 的方法相同的纹理标记化策略，HCSS [208]，的生成性能表明，Transformer 能够借助焦点注意力对复杂场景的构成进行出色建模。

HAML 实践 如上所述，人类辅助机器学习和机器学习辅助人类在自动机器学习系统中都发挥着重要作用。本节以 Kaggle 竞赛“梅赛德斯-奔驰测试时间成本预测”为例，介绍 OmniForce 平台如何践行以人为中心的理念，并实现高效的人机交互。

本次竞赛旨在根据提供的匿名数据集，预测用户定义的汽车通过安全测试所需的时间，从而帮助梅赛德斯-奔驰团队优化测试系统。用户将数据集上传到平台后，OmniForce 会自动将其识别为表格数据集。此外，

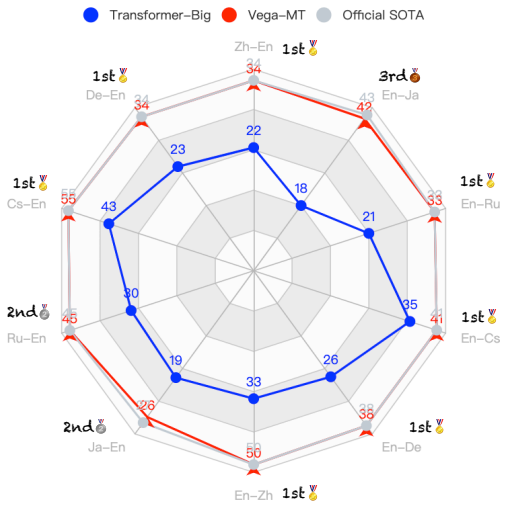


图 10：在 WMT-2022 的所有受限系统中，Vega-MT 在 10 个高资源翻译任务中的 7 个任务上取得了最先进的 BLEU分数，并且在 BLEU 分数方面显著优于具有竞争力的 transformer-BIG 基线。

表4：COCO-stuff[209]和视觉基因组（VG） [210]上的结果对比。所有结果均取自原始论文，并基于 256×256 分辨率。“-”表示相应论文中未提供相关数值。

	FID ↓		SceneFID ↓		Inception 分数 ↑		多样性分数 ↑	
	COCO	VG	COCO	VG	COCO	VG	COCO	VG
LostGAN-V2 [211]	42.55	47.62	22.00	18.27	18.01±0.50	14.10±0.38	0.55±0.09	0.53±0.09
OCGAN [212]	41.65	40.85	-	-	-	-	-	-
HCSS [208]	33.68	19.14	13.36	8.61	-	-	-	-
LAMA [213]	31.12	31.63	18.64	13.66	-	-	0.48±0.11	0.54±0.09
Frido [214]	37.14	-	14.91	-	18.62±0.54	-	-	-
TwFA [207]	**22.15**	**17.74**	**11.99**	**7.54**	**24.25±1.04**	**25.13±0.66**	**0.67±0.00**	**0.64±0.00**

系统会生成探索性数据分析报告，帮助人类从各个方面进一步探索数据的统计特征。相关界面如图 11 和图 12 所示。

面对种类繁多的数据集、有限的候选项以及复杂的搜索空间，纯机器驱动的自动化机器学习难以在一次迭代中找到最优解。因此，OmniForce 引入了结合人机交互的多轮优化流程。

第一轮中，我们将原始数据直接输入 OmniForce平台并运行自动化机器学习生命周期，包括自动合并表生成、自动数据处理、自动特征工程、模型搜索和超参数调优，以获得基线结果。由于缺乏人类干预，第2.1.6节所述的格式化器会根据知识库为回归任务配置搜索空间，然后生成待运行的自动机器学习流水线。图13展示了交互式搜索详情页，其中包括模型性能、候选项性能、平行超参数坐标等。通过批量推理获得结果后，我们在 Kaggle 网站上对其进行评估，以模拟模型的在线服务场景。我们的初步结果在私有排行榜上取得了 0.54956 的 r2 分数，在总计 3823 个团队中排名第 1540 位（迟交）。

在第二轮中，我们根据第2.1.8节介绍的 OmniForce顾问来修改搜索空间。该顾问会就如何更改搜索空间和进行特征工程提供建议。人类可以根据上述建议和自身经验，显著提升机器的效率。

搜索过程可视化是自动化机器学习系统中一个长期存在的议题，最近也引起了广泛关注[215][216][217][218]。此外，我们认为，高效地展示和修改搜索空间，是人类在 HAML 系统中协助机器的必要前提。在搜索详情页中，OmniForce 通过三种不同的图表展示当前候选项和搜索空间，分别见图14（完整视图）、图15（降维视图）和图16（成对视图）。

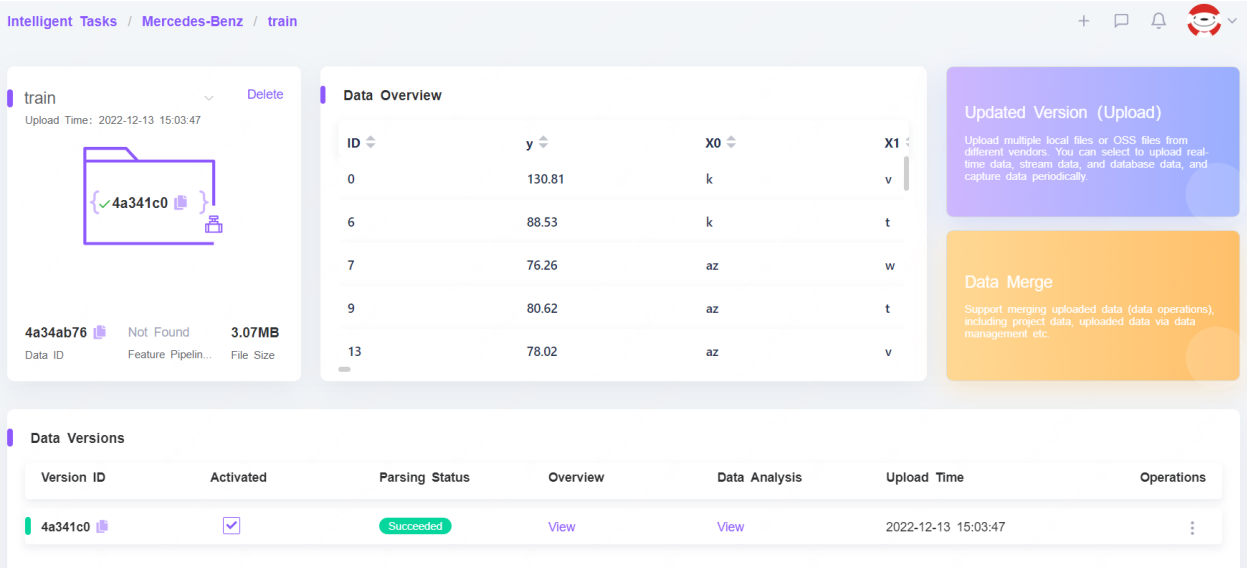


图 11：数据详情页。此处展示了元数据、数据预览和版本管理界面。元数据包含数据 ID、相关的特征流水线 ID 以及文件大小。可在此页面预览已上传的数据，以便进一步检查。版本管理用于处理快速的数据迭代，并及时更新方案。

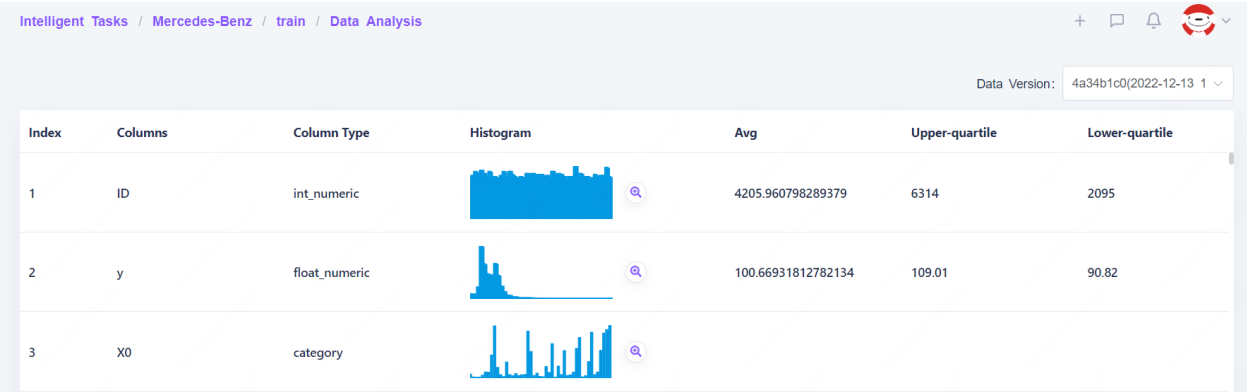


图 12：数据分析页。OmniForce 可通过基于自动显示的统计特征（如直方图和均值）分析数据，帮助用户高效地深入洞察数据。

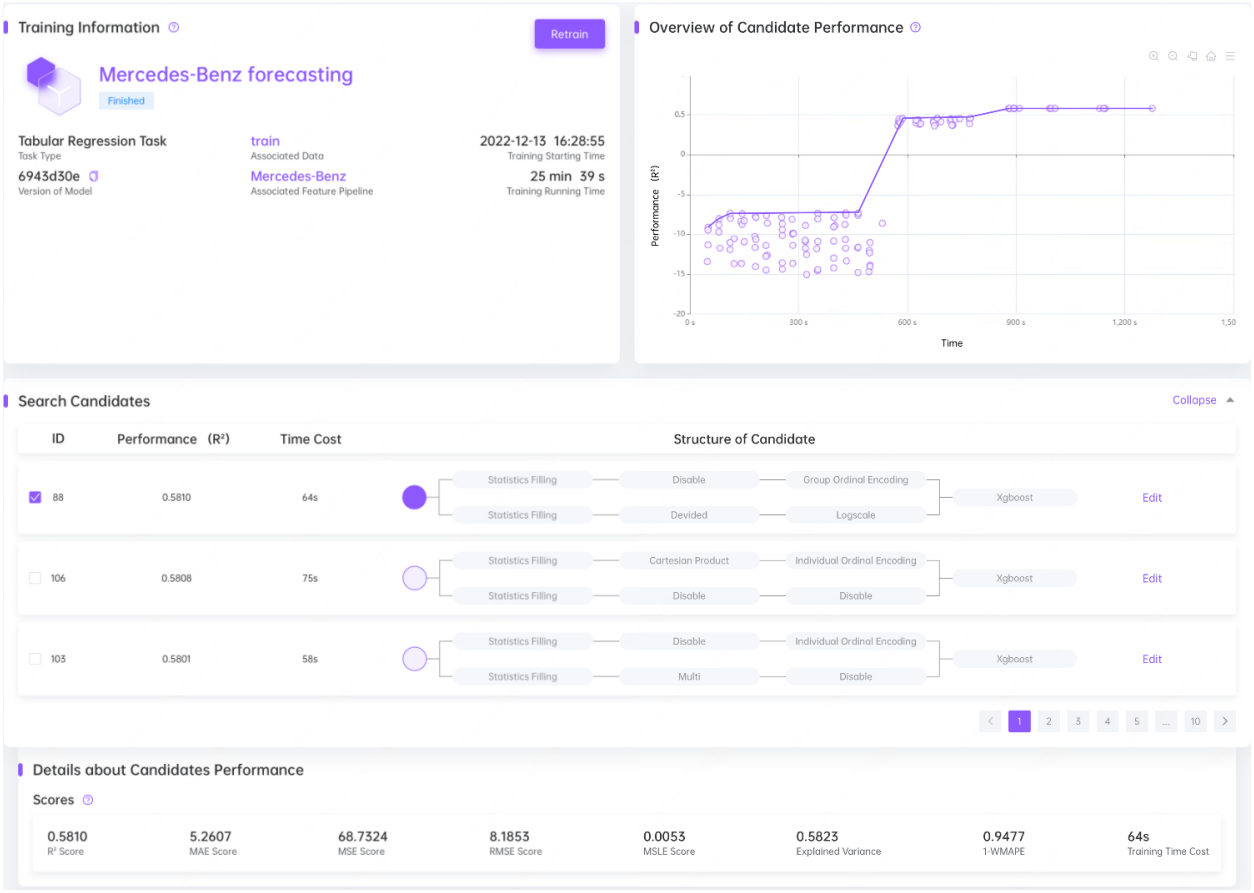


图 13：搜索详情页。此页面展示了任务的详细信息，例如数据版本和特征流水线版本。图表中还展示了性能预览和详细的候选信息。

编辑界面的初始状态是顾问建议的配置。OmniForce顾问根据数据统计、当前搜索结果和知识库，总结出一些优化方案。大多数建议具有通用性和可迁移性；用户可以选择是否采纳，也可以根据自身经验进一步完善。与图表交互后，修改后的搜索空间与原始搜索空间的对比会显示在图17中，用户可以预览并进一步编辑。
我们对上述搜索配置进行调整后，在私有排行榜上获得了 0.5511 的 r2 分数，排名为 820/3823，这表明顾问的有效性。

此外，人们可以关注搜索过程的各个方面，以增强机器的能力。上述三个与搜索空间相关的图表可通过点击或使用套索工具直接编辑。也就是说，只需点击几下，我们就可以进一步减小树的最大_深度和行采样比例，具体过程如图18所示。吸收人类的宝贵经验后，机器的搜索性能得到进一步提升，在私有排行榜上获得了 0.55184 的 r2 分数，排名也相应提升至 338。

接下来，顾问通过分析训练数据以及搜索过程中各特征的重要性等级，根据观测到的信息提供特征工程建议。具体而言，建议的特征如图19所示，用户可以根据自己的经验选择是否采纳这些建议。完成特征配置后，平台会创建一个新的特征流水线版本，并自动将所选更改应用于原始数据，如图20所示。

根据机器的建议完成上述修改后，我们获得了 0.55284 的分数，对应排名为第 76 名。

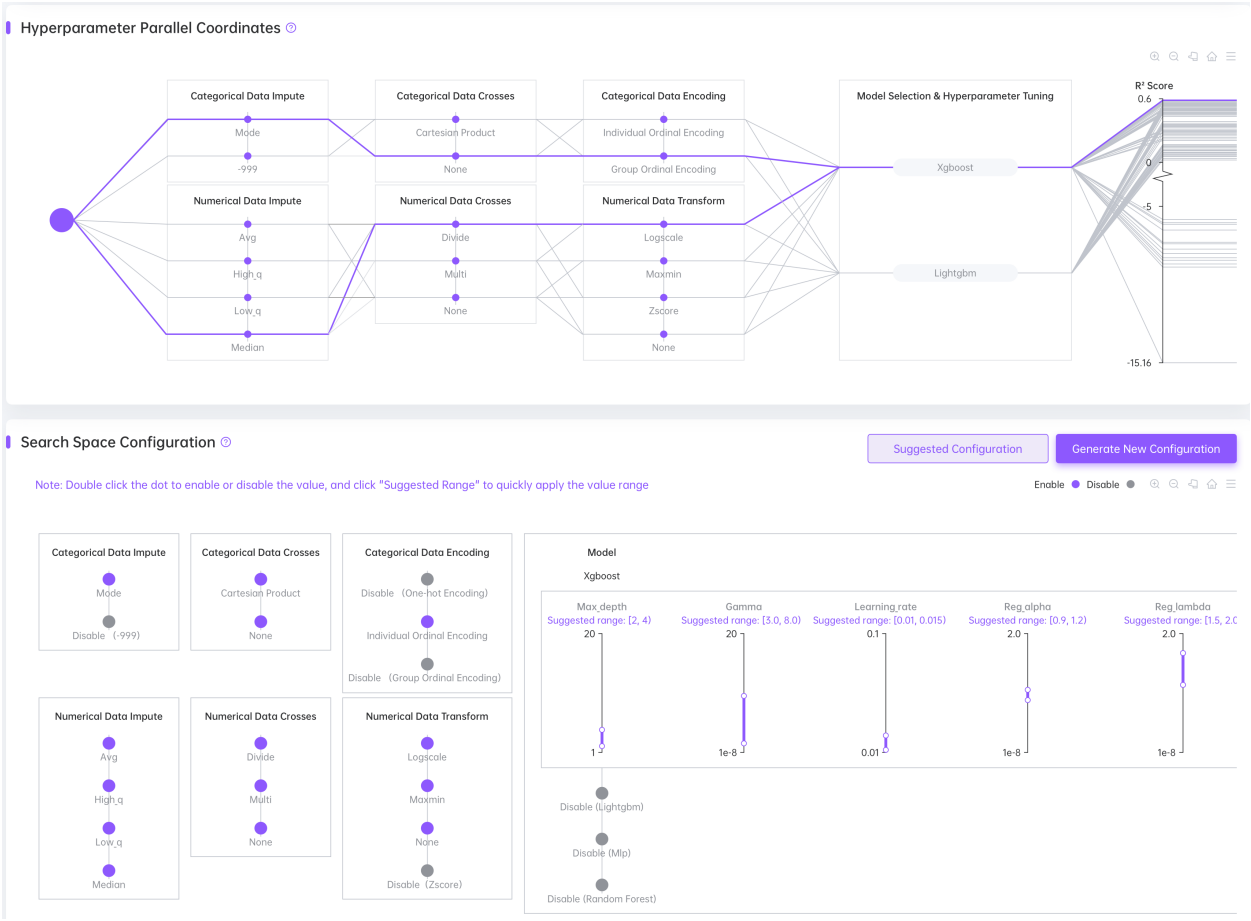


图14：并行超参数坐标图及机器的建议。用户可以从超参数与模型性能之间的关系中获取洞见。为方便调整超参数，OmniForce 在这些参数旁提供了多层级搜索空间配置界面。

在最后一轮中，用户可以充分发挥特征工程能力，进一步提升性能。具体来说，人类可以通过“前往特征流水线”按钮进入特征工程界面，并根据经验，通过执行 SQL 语句或使用预设方法来修改数据，如图21所示。最终，我们获得了 0.55394 的分数，在 3823 名参与者中排名第 11。

如图22所示，所有由此得出的改进点均源自人的经验和机器的建议，这体现了 OmniForce 系统所提出的人机交互功能的进步。上述尝试会保存为不同的特征流水线版本、搜索配置版本和模型版本，供用户按需切换。

6 相关工作

近年来，一些商业公司推出了自己的自动化机器学习平台，用于工业应用。有些公司会细分市场，并针对不同任务开发不同产品。例如，Amazon SageMaker Canvas [219] 允许业务分析师以无代码的方式构建人工智能模型，并对表格数据做出准确预测；Amazon Forecast [220]则采用时间序列预测服务。IBM Watson AutoAI [29] 支持针对表格数据构建分类和预测任务，而 Watson Natural Language Understanding [221]则专注于高级文本分析。其他平台，如 Abacus [222], 、Microsoft Azure AutoML [223] 和 Google Cloud Vertex AI [224]，则在一个平台上支持不同类型的数据和任务。OmniForce 选择了后一种方案，旨在打造通用且可复用的人工智能应用生产流水线。为打破任务自动化和单点优化各自孤立的局面，OmniForce 强调以人为中心的运营；让业务人员参与这一过程

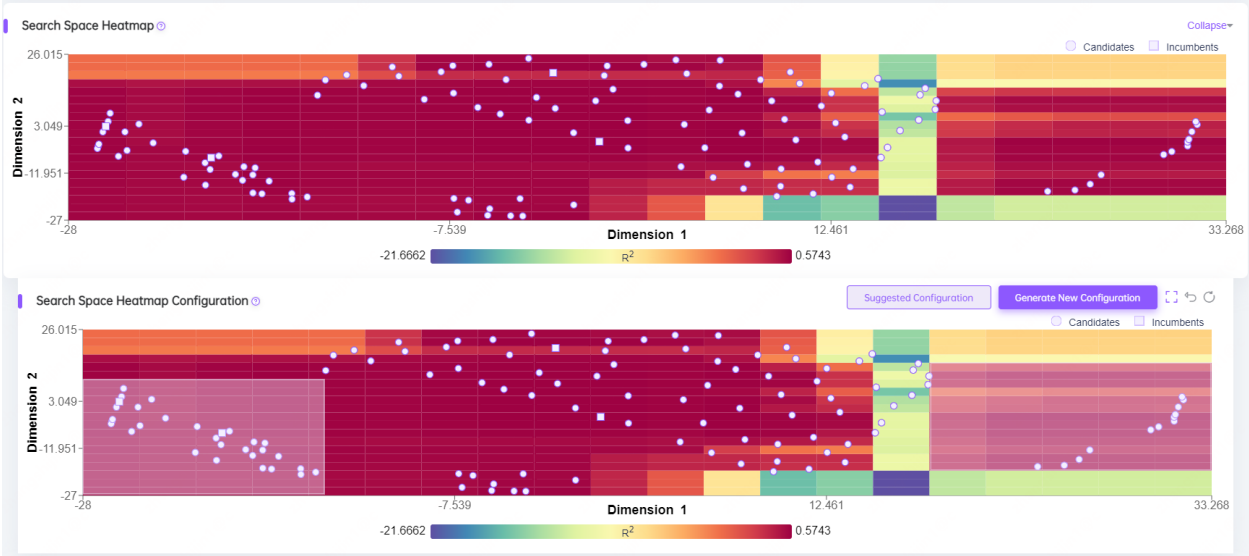


图15：搜索空间热力图与机器的建议。OmniForce 将高维、多层级搜索空间映射到二维空间，从而清晰呈现候选项与性能之间的关系。当前最优候选项代表能够带来显著性能提升的候选项，而套索回归选出的区域则代表建议的搜索空间。

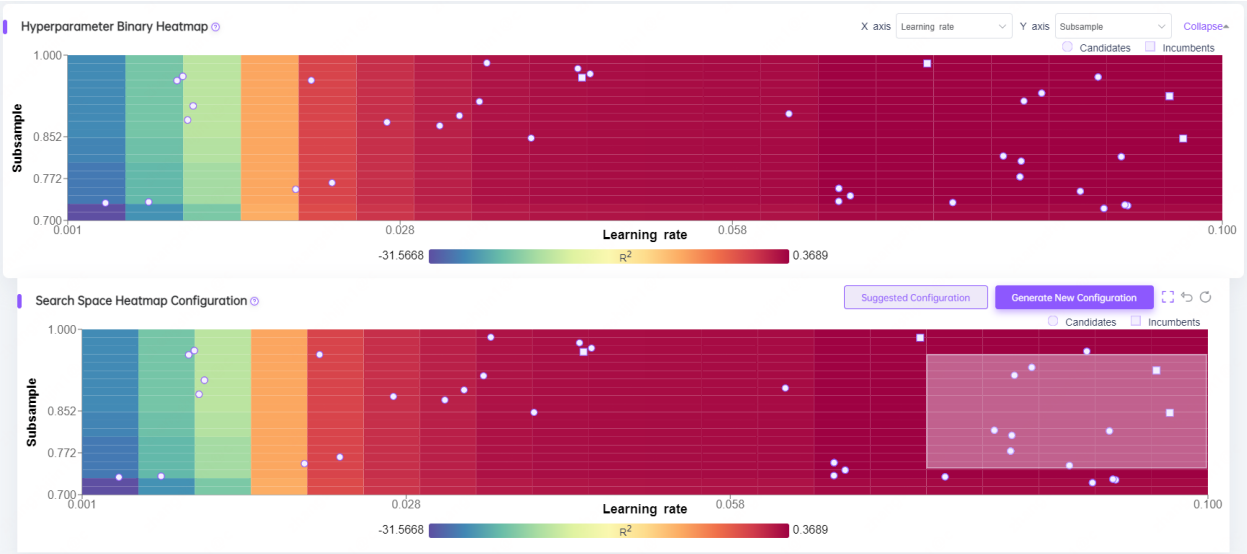


图16：超参数二元热力图与机器的建议。二元超参数热力图反映了所选两个超参数之间的交互关系，有助于人类进行更精细的调整。

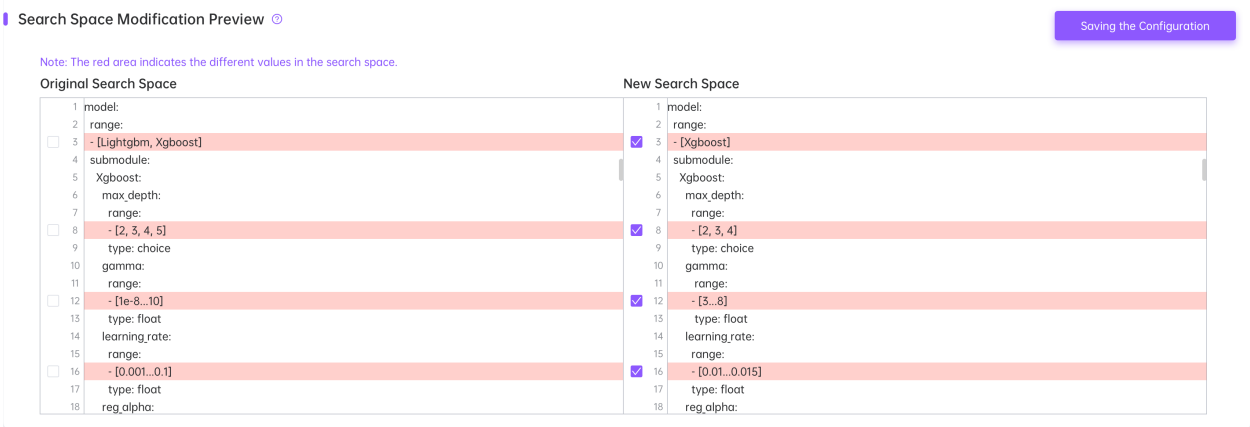


图17：修改后的搜索空间与原始搜索空间之间的比较。逐行比较可以清晰地突出搜索空间的修改之处。此外，用户可以直接使用 YAML 文件的语法修改搜索空间。

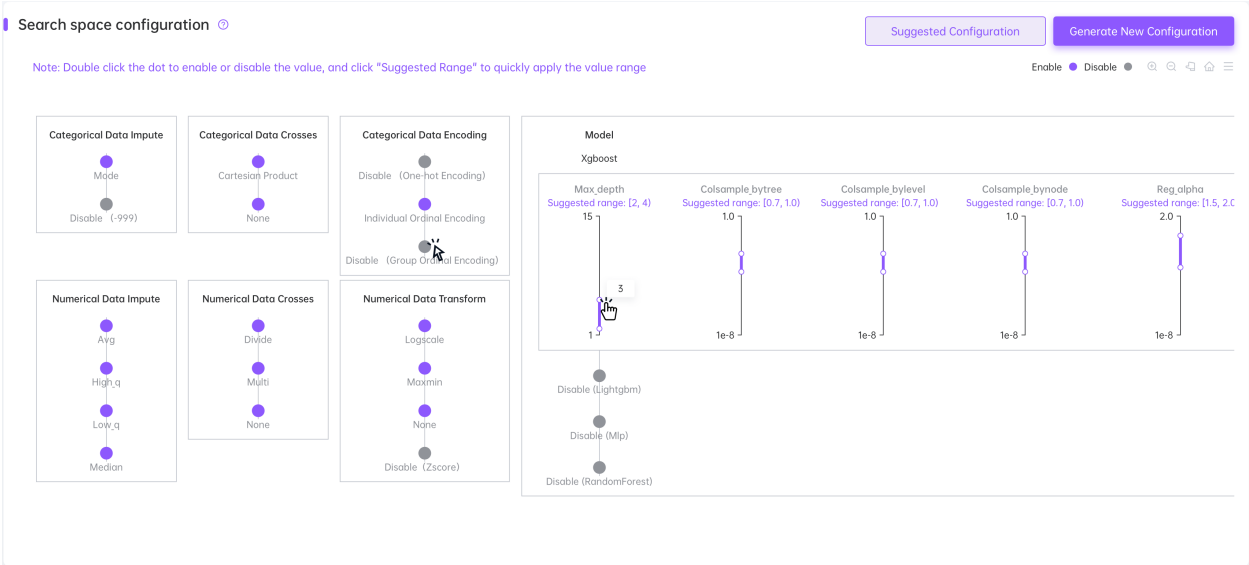


图18：修改搜索空间的过程。我们使用球形图标表示类别型超参数，并点击它们以更改其取值范围；使用滑块表示数值型超参数，并可通过拖动滑块修改搜索空间。

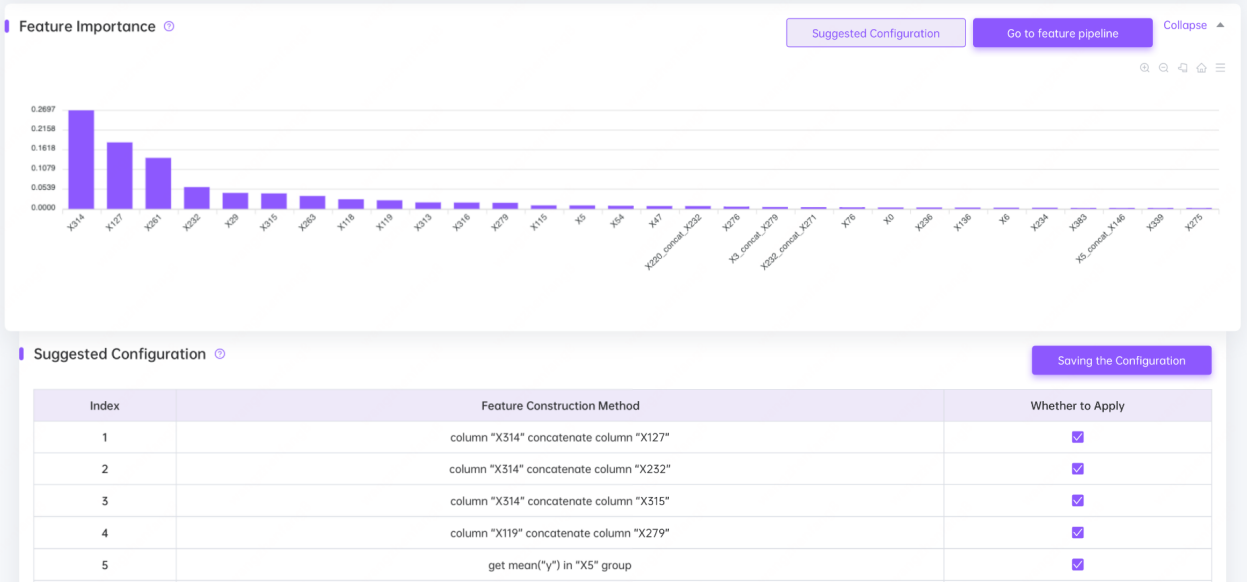


图19：特征重要性和机器的建议。搜索详情页会显示所有特征的重要性，顾问会提供进一步调整的建议。人类可以根据自身经验选择是否采纳这些建议。

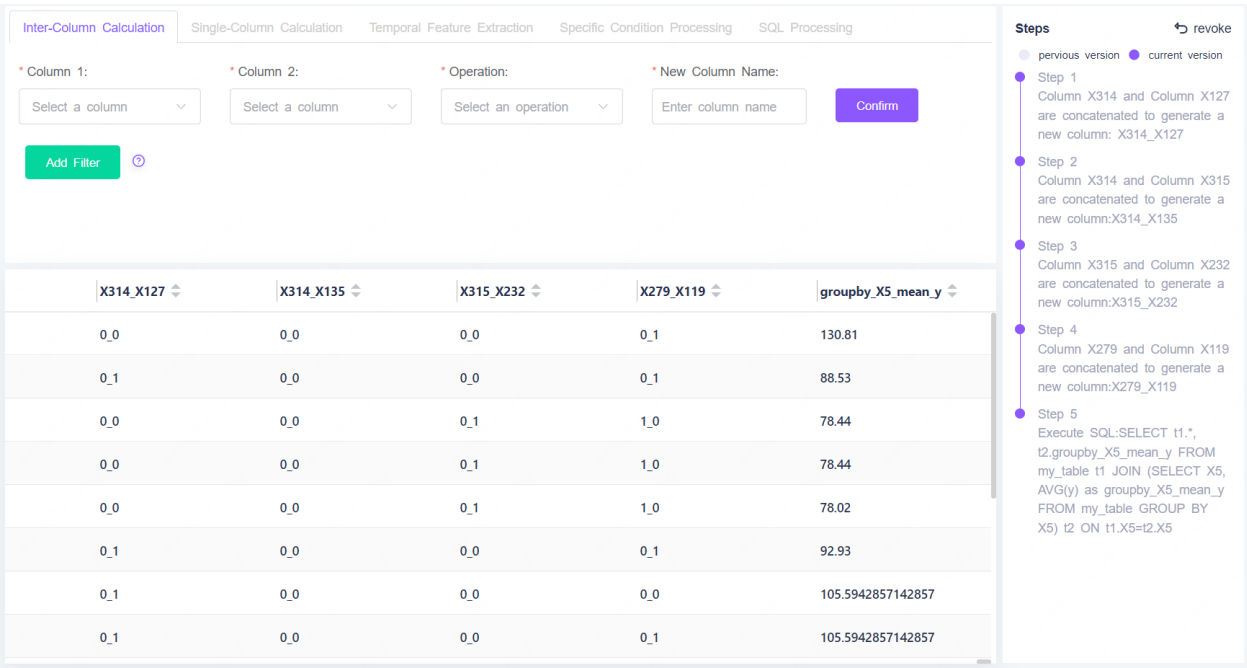


图20：特征流水线中建议的特征构造方法。所有应用的特征构造方法都会自动执行，并更新至新的特征流水线版本，该版本将在 3.1.2 节中介绍。

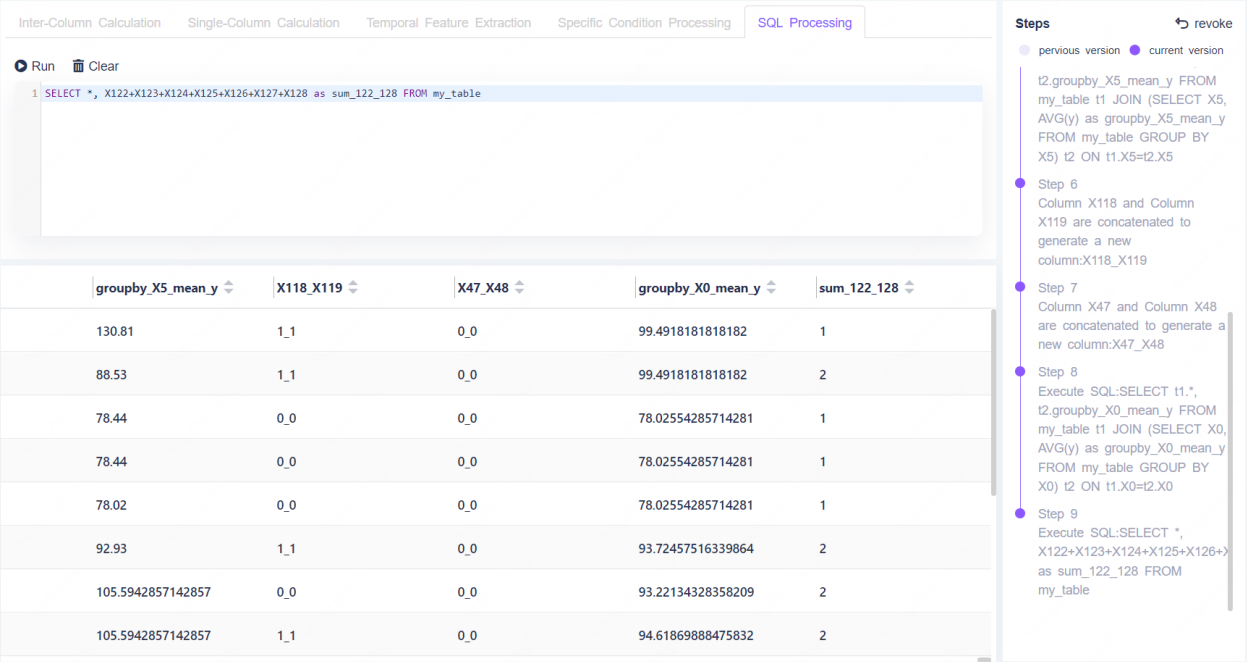


图21：定制化特征流水线。继承上一版本的特征流水线后，用户可以进一步基于对特征重要性的洞察创建特征。用户可以灵活切换特征流水线的版本，以比较最终性能或适应不同场景。

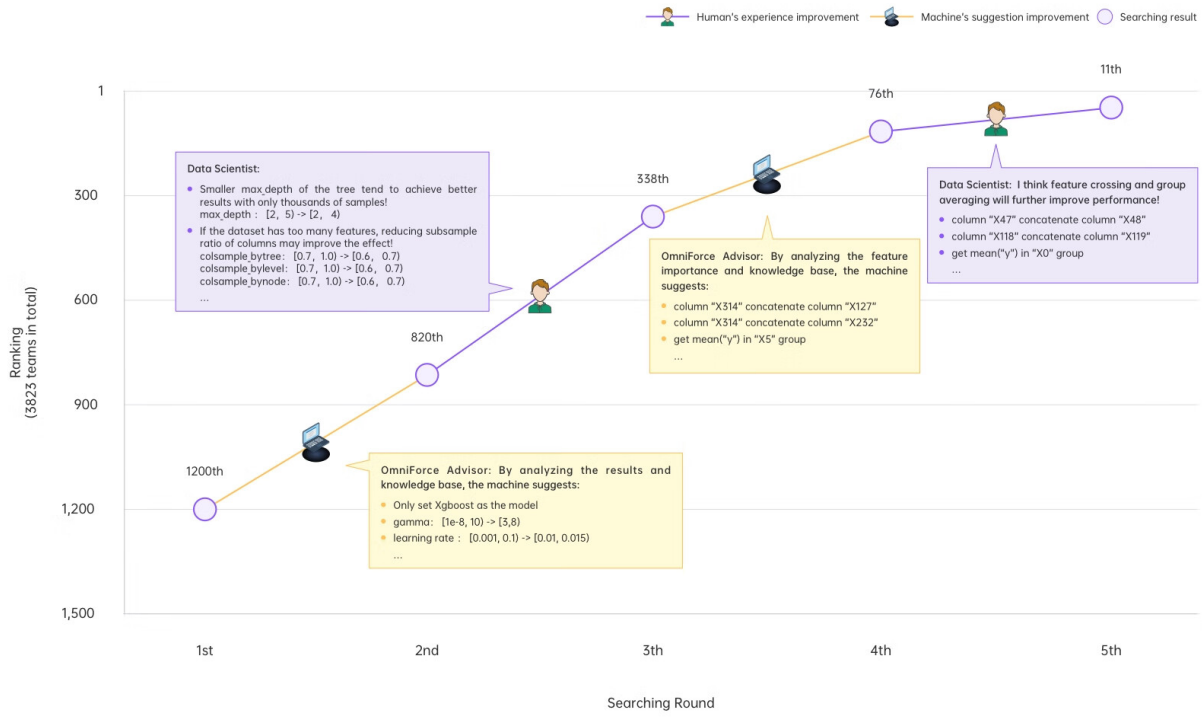


图22：OmniForce 中人机交互的示意图。基于搜索空间和特征流水线中的人机交互，HAML 系统能够实现更好的性能。

表5：开源和商业自动化机器学习平台中的特征对比。我们根据所提供的应用程序和系统视角进行比较。* 表示我们未找到足够的已发布信息来验证相应条目。V. G. 表示“很好”。

应用程序										
黑盒调优	Yes	Yes	Yes	Yes	No	Yes	No	No	No	Yes
计算机视觉	No	No	No	No	Yes	Yes	Yes	No	Yes	Yes
NLP	No	No	No	No	Yes	Yes	Yes	No	Yes	Yes
表格数据和时间序列	No	No	No	No	Yes	Yes	Yes	Yes	Yes	Yes
人工智能生成内容	No	No	No	No	No	No	No*	No	No*	Yes
系统										
自动可扩展性	No	Yes	Part	Yes	Yes	Yes	Yes	Yes*	Yes	Yes
云原生	No	Yes	Part	Part	Yes	Yes	Yes	Yes	Yes	Yes
可视化	Min	Min	Min	Min	Good	Good	很好	Good	Good	Good
云边协同	No	No	No	No	No	Part	No	No	Part	Yes
大模型支持	No	No	No	No	No*	Yes*	No	No	Yes	Yes
众包	No	Part	No	Yes	No	Yes	No	No	Yes*	Yes
高级搜索策略	Yes	No	Part	Yes	Yes	Yes	Yes	Yes	Yes	Yes
机器学习运维	No	Part	No	No	Yes	Part	Yes	Part	Yes	Yes

构建人工智能应用；并以协作、持续的方式整合业务、数据、算法和维护。

表5展示了自动化机器学习系统生态的概况，其中将一些开源平台（如 HyperOpt [22], Katib [27], NNI [23], Ray-Tune [21] ）以及商业平台（如 Abacus [222], Google Vertex AI [224], IBM Watson AutoAI [29], Amazon SageMaker Canvas [219],Microsoft Azure AutoML [223]）与 OmniForce 进行了比较。OmniForce 是少数支持众包、云边协同、超深度模型（大模型）和以人为中心的系统的系统之一。OmniForce 是一个云原生自动机器学习系统，可在生产环境中运行并用于商业用途。

7 结论

凭借所提出的以人为中心、云边协同的自动化机器学习方法，以及广泛提供的应用程序对大模型及其高效迁移策略的支持，OmniForce 为希望在开放环境场景中受益于人工智能技术的个人和企业构建了一个切实可行且功能强大的系统。这类场景包括工业供应链和工业元宇宙，其中人们常常面临开环问题，需要训练或更新大模型，并在具有不同约束和需求的大量设备上部署模型。OmniForce 提供统一的用户界面、无缝灵活的搜索策略框架和云原生特性，是一个易于使用、用途广泛且已具备产品化条件的系统，可帮助人们在实践中建立并改进人工智能工程，并持续从人工智能技术中挖掘商业价值。

致谢

我们感谢何晓东博士提供的宝贵支持和建设性意见。本研究（部分）是在张静博士访问京东探索研究院期间以及李佳兴在京东探索研究院实习期间完成的。

参考文献

[1] 何恺明、张祥雨、任少卿和孙剑。用于图像识别的深度残差学习。发表于*CVPR*，2016年。

[2] Joseph Redmon、Santosh Divvala、Ross Girshick 和 Ali Farhadi。只看一次：统一的实时目标检测。发表于 CVPR，2016年。[3] Ashish Vaswani、Noam Shazeer、Niki Parmar、Jakob Uszkoreit、Llion Jones、Aidan N Gomez、Łukasz Kaiser 和 Illia Polosukhin。注意力机制就是你所需要的一切。发表于 NeurIPS，2017年。[4] Tom Brown、Benjamin Mann、Nick Ryder、Melanie Subbiah、Jared D Kaplan、Prafulla Dhariwal、Arvind Neelakantan、Pranav Shyam、Girish Sastry、Amanda Askell 等。语言模型是少样本学习器。发表于 NeurIPS，2020年。[5] Qiantong Xu、Alexei Baevski、Tatiana Likhomanenko、Paden Tomasello、Alexis Conneau、Ronan Collobert、Gabriel Synnaeve 和 Michael Auli。自训练和预训练对于语音识别具有互补作用。发表于 ICASSP，2021年。[6] Anmol Gulati、James Qin、Chung-Cheng Chiu、Niki Parmar、Yu Zhang、Jiahui Yu、Wei Han、Shibo Wang、Zhengdong Zhang、Yonghui Wu 等。Conformer：用于语音识别的卷积增强型Transformer。arXiv 预印本 arXiv:2005.08100，2020年。[7] Alec Radford、Jong Wook Kim、Chris Hallacy、Aditya Ramesh、Gabriel Goh、Sandhini Agarwal、Girish Sastry、Amanda Askell、Pamela Mishkin、Jack Clark 等。通过自然语言监督学习可迁移视觉模型。发表于 ICML，2021年。[8] Aditya Ramesh、Mikhail Pavlov、Gabriel Goh、Scott Gray、Chelsea Voss、Alec Radford、Mark Chen 和 Ilya Sutskever。零样本文本到图像生成。发表于 ICML，2021年。[9] Tianqi Chen 和 Carlos Guestrin。XGBoost：一种可扩展的树提升系统。发表于 SIGKDD，2016年。[10] Heng-Tze Cheng、Levent Koc、Jeremiah Harmsen、Tal Shaked、Tushar Chandra、Hrishi Aradhye、Glen Anderson、Greg Corrado、Wei Chai、Mustafa Ispir 等。用于推荐系统的宽&深度学习。发表于第一届 DLRS 研讨会论文集，2016年。[11] Tesla 人工智能与自动驾驶。https://www.tesla.com/AI，2022年。[12] Google 翻译。https://translate.google.com/，2022年。[13] Apple Siri。https://www.apple.com/siri/，2022年。[14] OpenAI ChatGPT。https://openai.com/blog/chatgpt/，2022年。[15] Esteban Real、Sherry Moore、Andrew Selle、Saurabh Saxena、Yutaka Leon Suematsu、Jie Tan、Quoc V. Le 和 Alexey Kurakin。图像分类器的大规模进化。发表于 ICML，2017年。[16] James Bergstra 和 Yoshua Bengio。用于超参数优化的随机搜索。机器学习研究期刊，2012年。[17] Jasper Snoek、Hugo Larochelle 和 Ryan P. Adams。机器学习算法贝叶斯优化的实践方法。发表于 NeurIPS，2012年。[18] Bowen Baker、Otkrist Gupta、Nikhil Naik 和 Ramesh Raskar。使用强化学习设计神经网络架构。发表于 ICLR，2017年。[19] Hanxiao Liu、Karen Simonyan 和 Yiming Yang。DARTS：可微架构搜索。发表于 ICLR，2019年。[20] Takuya Akiba、Shotaro Sano、Toshihiko Yanase、Takeru Ohta 和 Masanori Koyama。Optuna：新一代超参数优化框架。发表于 SIGKDD，2019年。[21] Richard Liaw、Eric Liang、Robert Nishihara、Philipp Moritz、Joseph E Gonzalez 和 Ion Stoica。Tune：用于分布式模型选择和训练的研究平台。arXiv 预印本 arXiv:1807.05118，2018年。[22] James Bergstra、Brent Komer、Chris Eliasmith、Dan Yamins 和 David D Cox。Hyperopt：用于模型选择和超参数优化的 Python 库。Computational Science & Discovery，2015年。[23] NNI：用于神经架构搜索、模型压缩和超参数调优的开源自动机器学习工具包。https://github.com/microsoft/nni，2022年。[24] Orion：用于黑盒函数优化的异步框架。https://github.com/Epistimio/ orion，2022年。[25] Auto-sklearn：自动化机器学习工具包，可直接替代 scikit-learn 估计器。https://github.com/automl/auto-sklearn，2022年。[26] Kubernetes：适用于生产环境的容器编排。https://kubernetes.io/，2020年。[27] Johnu George、Ce Gao、Richard Liu、Hou Gang Liu、Yuan Tang、Ramdoot Pydipaty 和 Amit Kumar Saha。一种可扩展的云原生超参数调优系统。arXiv 预印本 arXiv:2006.02085，2020年。

[28] Google Cloud AutoML：以最少的工作量和机器学习专业知识，训练高质量的自定义机器学习模型。https://cloud.google.com/automl，2022年。[29] IBM Watson AutoAI：快速构建和训练高质量的预测模型，简化人工智能生命周期管理。https://www.ibm.com/cloud/watson-studio/autoai，2022年。[30] Amazon SageMaker：借助完全托管的基础设施、工具和工作流，为任何使用场景构建、训练和部署机器学习（ML）模型。https://aws.amazon.com/sagemaker，2022年。[31] H2O Driverless AI：屡获殊荣的自动机器学习（AutoML）平台。https://h2o.ai/platform/ ai-cloud/make/h2o-driverless-ai/，2022年。[32] Lisha Li、Kevin Jamieson、Giulia DeSalvo、Afshin Rostamizadeh 和 Ameet Talwalkar。Hyperband：用于超参数优化的基于多臂老虎机的配置评估。在 ICLR，2017年。[33] Chao Xue、Xiaoxing Wang、Junchi Yan 和 Chun-Guang Li。基于最大流的神经架构搜索方法。在 ECCV，2022年。[34] Paul Barham、Aakanksha Chowdhery、Jeff Dean、Sanjay Ghemawat、Steven Hand、Daniel Hurt、Michael Isard、Hyeontaek Lim、Ruoming Pang、Sudip Roy、Brennan Saeta、Parker Schuh、Ryan Sepassi、Laurent Shafey、Chandu Thekkath 和 Yonghui Wu。Pathways：面向机器学习的异步分布式数据流。在 MLSys，2022年。[35] Kubeflow：面向 Kubernetes 的机器学习工具包。https://github.com/kubeflow/kubeflow/，2022年。[36] Kubeflow 中的训练算子。https://github.com/kubeflow/training-operator，2022年。[37] Kubeflow 中的 MPI 算子。https://github.com/kubeflow/mpi-operator，2022年。[38] 分布式系统工具包：组合容器的模式。https://kubernetes.io/blog/2015/06/ the-distributed-system-toolkit-patterns/，2022年。[39] Kubeflow Pipelines：基于 Docker 容器、用于构建和部署可移植且可扩展机器学习（ML）工作流的平台。https://github.com/kubeflow/pipelines，2022年。[40] Amine Barrak、Ellis E Eghan 和 Bram Adams。机器学习流水线与源代码的协同演化——DVC 项目的实证研究。在 SANER，2021年。[41] Burr Settles。主动学习文献综述。2009年。[42] Frank McSherry 和 Kunal Talwar。通过差分隐私进行机制设计。在第48届 IEEE 计算机科学基础年会（FOCS'07），第94–103页。IEEE，2007年。[43] Cynthia Dwork 和 Aaron Roth。差分隐私的算法基础。《理论计算机科学基础与趋势》，2014年。[44] Cynthia Dwork。差分隐私。载于 Michele Bugliesi、Bart Preneel、Vladimiro Sassone 和 Ingo Wegener 编，《自动机、语言与编程》。Springer Berlin Heidelberg，2006年。[45] Peter Kairouz、Sewoong Oh 和 Pramod Viswanath。差分隐私的组合定理。IEEE 信息论汇刊，2017年。[46] Fengxiang He、Bohan Wang 和 Dacheng Tao。迭代差分隐私学习算法的更紧泛化界。在 UAI，2021年。[47] Fengxiang He 和 Dacheng Tao。深度学习基础。Springer，2018年。[48] Cynthia Dwork、Vitaly Feldman、Moritz Hardt、Toniann Pitassi、Omer Reingold 和 Aaron Leon Roth。在自适应数据分析中保持统计有效性。在 STOC，2015年。[49] Kobbi Nissim 和 Uri Stemmer。论差分隐私的泛化性质。计算机研究文库，2015年。[50] Martin Abadi、Andy Chu、Ian Goodfellow、H Brendan McMahan、Ilya Mironov、Kunal Talwar 和 Li Zhang。使用差分隐私进行深度学习。在 ACM CCS，2016年。[51] Luca Oneto、Sandro Ridella 和 Davide Anguita。差分隐私与泛化：具有应用的更紧界。《模式识别快报》，2017年。[52] KServe 作者。KServe：基于 Kubernetes、具有高可扩展性且遵循标准的可信人工智能模型推理平台。https://github.com/kserve/kserve，2022年。[53] Frank Hutter、Holger Hoos 和 Kevin Leyton-Brown。评估超参数重要性的高效方法。在 ICML，2014年。[54] Marco Tulio Ribeiro、Sameer Singh 和 Carlos Guestrin。Anchors：高精度的模型无关解释。在 AAAI，2018年。

[55] 高通 Adreno 650。https://developer.qualcomm.com/software/adreno-gpu-sdk/gpu。[56] 英伟达 Jetson 开发套件。https://developer.nvidia.com/embedded/jetson-developer-kits，2022年。[57] Aleksander Madry、Aleksandar Makelov、Ludwig Schmidt、Dimitris Tsipras 和 Adrian Vladu。迈向能够抵抗对抗攻击的深度学习模型。在 ICLR，2018年。[58] Fengxiang He、Shaopeng Fu、Bohan Wang 和 Dacheng Tao。对抗训练的鲁棒性、隐私与泛化。arXiv 预印本 arXiv:2012.13573，2020年。[59] Samuel Yeom、Irene Giacomelli、Matt Fredrikson 和 Somesh Jha。机器学习中的隐私风险：分析其与过拟合的联系。在 计算机安全基金会研讨会，2018年。[60] ARM。https://www.arm.com，2022年。[61] ROCm。https://docs.amd.com/，2022年。[62] Tianqi Chen、Thierry Moreau、Ziheng Jiang、Lianmin Zheng、Eddie Q. Yan、Haichen Shen、Meghan Cowan、Leyuan Wang、Yuwei Hu、Luis Ceze、Carlos Guestrin 和 Arvind Krishnamurthy。TVM：用于深度学习的自动化端到端优化编译器。在 操作系统设计与实现研讨会，2018年。[63] TensorRT。https://developer.nvidia.com/tensorrt，2022年。[64] OpenVINO。https://docs.openvino.ai/，2022年。[65] ONNX。https://onnx.ai/，2022年。[66] 何恺明、Xinlei Chen、Saining Xie、Yanghao Li、Piotr Dollár 和 Ross Girshick。掩码自编码器是可扩展的视觉学习器。在 CVPR，2022年。[67] Jacob Devlin、Ming-Wei Chang、Kenton Lee 和 Kristina Toutanova。BERT：用于语言理解的深度双向 Transformer 预训练。在 NAACL-HLT，2019年。[68] Alec Radford、Jong Wook Kim、Chris Hallacy、Aditya Ramesh、Gabriel Goh、Sandhini Agarwal、Girish Sastry、Amanda Askell、Pamela Mishkin、Jack Clark、Gretchen Krueger 和 Ilya Sutskever。从自然语言监督中学习可迁移视觉模型。在 ICML，2021年。[69] Neil Houlsby、Andrei Giurgiu、Stanislaw Jastrzebski、Bruna Morrone、Quentin de Laroussilhe、Andrea Gesmundo、Mona Attariyan 和 Sylvain Gelly。面向 NLP 的参数高效迁移学习。在 ICML，2019年。[70] Edward J. Hu、Yelong Shen、Phillip Wallis、Zeyuan Allen-Zhu、Yuanzhi Li、Shean Wang、Lu Wang 和 Weizhu Chen。LoRA：大语言模型的低秩适配。在 ICLR，2022年。[71] P Molchanov、S Tyree、T Karras、T Aila 和 J Kautz。为资源高效推理剪枝卷积神经网络。在 ICLR，2019年。[72] Yongming Rao、Wenliang Zhao、Benlin Liu、Jiwen Lu、Jie Zhou 和 Cho-Jui Hsieh。DynamicViT：通过动态令牌稀疏化实现高效视觉Transformer。在 NeurIPS，2021年。[73] Geoffrey Hinton、Oriol Vinyals、Jeff Dean 等。蒸馏神经网络中的知识。arXiv 预印本 arXiv:1503.02531，2015年。[74] Hieu Pham、Melody Guan、Barret Zoph、Quoc Le 和 Jeff Dean。通过参数共享实现高效的神经架构搜索。在 ICML，2018年。[75] Yibo Yang、Hongyang Li、Shan You、Fei Wang、Chen Qian 和 Zhouchen Lin。ISTA-NAS：通过稀疏编码实现高效且一致的神经架构搜索。在 NeurIPS，2020年。[76] Benteng Ma、Jing Zhang、Yong Xia 和 Dacheng Tao。自动学习注意力。《神经信息处理系统进展》，33：1488–1500，2020年。[77] Yibo Yang、Shan You、Hongyang Li、Fei Wang、Chen Qian 和 Zhouchen Lin。迈向提升可微神经架构搜索的一致性、效率和灵活性。在 CVPR，2021年。[78] Jianping Gou、Baosheng Yu、Stephen J Maybank 和 Dacheng Tao。知识蒸馏：综述。《国际计算机视觉杂志》，129：1789–1819，2021年。[79] Kan Wu、Jinnian Zhang、Houwen Peng、Mengchen Liu、Bin Xiao、Jianlong Fu 和 Lu Yuan。TinyViT：面向小型视觉变换器的快速预训练蒸馏。在 ECCV，2022年。[80] Yutong Bai、Zeyu Wang、Junfei Xiao、Chen Wei、Huiyu Wang、Alan Yuille、Yuyin Zhou 和 Cihang Xie。掩码自编码器能够实现高效的知识蒸馏器。arXiv 预印本 arXiv:2208.12256，2022年。[81] Bobak Shahriari、Kevin Swersky、Ziyu Wang、Ryan P. Adams 和 Nando de Freitas。将人类移出闭环：贝叶斯优化综述。《IEEE 汇刊》，2016年。

[82] Jasper Snoek、Hugo Larochelle 和 Ryan P. Adams。机器学习算法的实用贝叶斯优化。发表于 NeurIPS，2012年。[83] GPyOpt 作者。GPyOpt：一个 Python 贝叶斯优化框架。http://github.com/SheffieldML/GPyOpt，2016年。[84] Tim Head、Manoj Kumar、Holger Nahrstaedt、Gilles Louppe 和 Iaroslav Shcherbatyi。scikit-optimize/scikit- optimize。https://doi.org/10.5281/zenodo.5565057，2021年。[85] A. Klein、S. Falkner、N. Mansur 和 F. Hutter。RoBO：一个灵活、稳健的 Python 贝叶斯优化框架。发表于 NeurIPS 贝叶斯优化研讨会，2017年。[86] Willie Neiswanger、Kirthevasan Kandasamy、Barnabás Póczos、Jeff Schneider 和 Eric P. Xing。ProBO：一个在贝叶斯优化中使用概率编程的框架。计算机研究文库，2019年。[87] Jacob R Gardner、Geoff Pleiss、David Bindel、Kilian Q Weinberger 和 Andrew Gordon Wilson。GPyTorch：借助 GPU 加速实现黑盒矩阵-矩阵高斯过程推断。发表于 NeurIPS，2018年。[88] Maximilian Balandat、Brian Karrer、Daniel R. Jiang、Samuel Daulton、Benjamin Letham、Andrew Gordon Wilson 和 Eytan Bakshy。BoTorch：一个高效蒙特卡洛贝叶斯优化框架。发表于 NeurIPS，2020年。[89] Javier González、Zhenwen Dai、Philipp Hennig 和 Neil D. Lawrence。通过局部惩罚实现批量贝叶斯优化。发表于 AISTATS，2016年。[90] Adam Paszke、Sam Gross、Francisco Massa、Adam Lerer、James Bradbury、Gregory Chanan、Trevor Killeen、Zeming Lin、Natalia Gimelshein、Luca Antiga、Alban Desmaison、Andreas Köpf、Edward Z. Yang、Zachary DeVito、Martin Raison、Alykhan Tejani、Sasank Chilamkurthy、Benoit Steiner、Lu Fang、Junjie Bai 和 Soumith Chintala。PyTorch：一种命令式风格的高性能深度学习库。发表于 NeurIPS，2019年。[91] Frank Hutter、Holger H. Hoos 和 Kevin Leyton-Brown。面向通用算法配置的序贯模型优化。发表于 LION，2011年。[92] Jasper Snoek、Oren Rippel、Kevin Swersky、Ryan Kiros、Nadathur Satish、Narayanan Sundaram、Md. Mostofa Ali Patwary、Prabhat 和 Ryan P. Adams。使用深度神经网络实现可扩展的贝叶斯优化。发表于 ICML，2015年。[93] Jost Tobias Springenberg、Aaron Klein、Stefan Falkner 和 Frank Hutter。使用稳健的贝叶斯神经网络进行贝叶斯优化。发表于 NeurIPS，2016年。[94] Xuanyi Dong 和 Yi Yang。NAS-Bench-201：扩展可复现神经架构搜索的范围。发表于 ICLR，2020年。[95] J. Mockus、Vytautas Tiesis 和 Antanas Zilinskas。贝叶斯方法在极值搜索中的应用。《迈向全局优化》，2014年。[96] Niranjan Srinivas、Andreas Krause、Sham M. Kakade 和 Matthias W. Seeger。老虎机设定下高斯过程优化的信息论遗憾界。IEEE 信息论汇刊，2012年。[97] Philipp Hennig 和 Christian J. Schuler。面向高效信息获取的全局优化熵搜索。机器学习研究期刊，2012年。[98] Clément Chevalier 和 David Ginsbourger。多点期望改进的快速计算及其在批量选择中的应用。发表于 LION，2013年。[99] Haoyi Zhou、Shanghang Zhang、Jieqi Peng、Shuai Zhang、Jianxin Li、Hui Xiong 和 Wancai Zhang。Informer：超越高效Transformer，实现长序列时间序列预测。发表于 AAAI，2021年。[100] Ailing Zeng、Muxi Chen、Lei Zhang 和 Qiang Xu。Transformer 对时间序列预测是否有效？arXiv 预印本 arXiv:2205.13504，2022年。[101] Lukas Ruff、Robert Vandermeulen、Nico Goernitz、Lucas Deecke、Shoaib Ahmed Siddiqui、Alexander Binder、Emmanuel Müller 和 Marius Kloft。深度单类分类。发表于 ICML，2018年。[102] Chen Qiu、Aodong Li、Marius Kloft、Maja Rudolph 和 Stephan Mandt。利用潜在离群值暴露，在污染数据上进行异常检测。发表于 ICML，2022年。[103] Huifeng Guo、Ruiming Tang、Yunming Ye、Zhenguo Li 和 Xiuqiang He。DeepFM：一种基于因子分解机的点击率预测神经网络。发表于 IJCAI，2017年。[104] Guorui Zhou、Xiaoqiang Zhu、Chenru Song、Ying Fan、Han Zhu、Xiao Ma、Yanghui Yan、Junqi Jin、Han Li 和 Kun Gai。用于点击率预测的深度兴趣网络。发表于 SIGKDD，第1059–1068页，2018年。[105] Leo Breiman。随机森林。《机器学习》，2001年。

[106] Guolin Ke、Qi Meng、Thomas Finley、Taifeng Wang、Wei Chen、Weidong Ma、Qiwei Ye 和 Tie-Yan Liu。LightGBM：一种高效的梯度提升决策树。发表于 NeurIPS，2017 年。[107] Heng-Tze Cheng、Levent Koc、Jeremiah Harmsen、Tal Shaked、Tushar Chandra、Hrishi Aradhye、Glen Anderson、Greg Corrado、Wei Chai、Mustafa Ispir 等。面向推荐系统的宽&深度学习。发表于首届推荐系统深度学习研讨会论文集，2016年。[108] Seref Sagiroglu 和 Duygu Sinanc。大数据：综述。发表于 CTS，2013 年。[109] Jan G De Gooijer 和 Rob J Hyndman。时间序列预测的 25 年。《国际预测杂志》，2006年。[110] Bryan Lim 和 Stefan Zohren。基于深度学习的时间序列预测：综述。《英国皇家学会哲学汇刊 A》，2021年。[111] George EP Box 和 David A Pierce。自回归积分滑动平均时间序列模型中残差自相关的分布。《美国统计学会杂志》，1970年。[112] Haixu Wu、Jiehui Xu、Jianmin Wang 和 Mingsheng Long。Autoformer：用于长期序列预测的自相关分解Transformer。发表于 NeurIPS，2021年。[113] Corinna Cortes 和 Vladimir Vapnik。支持向量网络。《机器学习》，1995年。[114] Alex Krizhevsky、Ilya Sutskever 和 Geoffrey E. Hinton。使用深度卷积神经网络进行ImageNet分类。发表于 NeurIPS，2012年。[115] Xiaohua Zhai、Alexander Kolesnikov、Neil Houlsby 和 Lucas Beyer。扩展视觉Transformer。发表于 IEEE/CVF计算机视觉与模式识别会议论文集，第12104–12113页，2022年。[116] Ze Liu、Han Hu、Yutong Lin、Zhuliang Yao、Zhenda Xie、Yixuan Wei、Jia Ning、Yue Cao、Zheng Zhang、Li Dong 等。Swin Transformer V2：扩展容量和分辨率。发表于 IEEE/CVF计算机视觉与模式识别会议论文集，第12009–12019页，2022年。[117] Qiming Zhang、Yufei Xu、Jing Zhang 和 Dacheng Tao。VitaeV2：通过探索归纳偏置改进视觉Transformer，用于图像识别及更多任务。《国际计算机视觉杂志》，第1–22页，2023年。[118] Yufei Xu、Jing Zhang、Qiming Zhang 和 Dacheng Tao。Vitpose+：用于通用人体姿态估计的视觉Transformer基础模型。arXiv 预印本 arXiv:2212.04246，2022年。[119] Di Wang、Qiming Zhang、Yufei Xu、Jing Zhang、Bo Du、Dacheng Tao 和 Liangpei Zhang。推动纯视觉Transformer迈向遥感基础模型。IEEE 地球科学与遥感汇刊，2022年。[120] Mostafa Dehghani、Josip Djolonga、Basil Mustafa、Piotr Padlewski、Jonathan Heek、Justin Gilmer、Andreas Steiner、Mathilde Caron、Robert Geirhos、Ibrahim Alabdulmohsin 等。将视觉Transformer扩展至220亿个参数。arXiv 预印本 arXiv:2302.05442，2023年。[121] Wen Wang、Jing Zhang、Yang Cao、Yongliang Shen 和 Dacheng Tao。迈向数据高效的检测Transformer。发表于《计算机视觉—ECCV 2022：第17届欧洲计算机视觉会议，特拉维夫，以色列，2022年10月23—27日，论文集》，第九部分，第88–105页。Springer，2022年。[122] Chen Chen、Zhe Chen、Jing Zhang 和 Dacheng Tao。SASA：面向基于点的三维目标检测的语义增强集合抽象。发表于 AAAI 人工智能会议论文集，第36卷，第221–229页，2022年。[123] Haobo Yuan、Xiangtai Li、Yibo Yang、Guangliang Cheng、Jing Zhang、Yunhai Tong、Lefei Zhang 和 Dacheng Tao。PolyphonicFormer：用于深度感知视频全景分割的统一查询学习。发表于《计算机视觉—ECCV 2022：第17届欧洲计算机视觉会议，特拉维夫，以色列，2022年10月23—27日，论文集》，第二十七部分，第582–599页。Springer，2022年。[124] Yangyang Xu、Xiangtai Li、Haobo Yuan、Yibo Yang、Jing Zhang、Yunhai Tong、Lefei Zhang 和 Dacheng Tao。使用多查询Transformer进行密集预测的多任务学习。arXiv 预印本 arXiv:2205.14354，2022年。[125] Zhe Chen、Jing Zhang 和 Dacheng Tao。面向道路检测的渐进式激光雷达适配。《IEEE/CAA 自动化学报》，6(3):693–702，2019年。[126] Lefei Zhang、Meng Lan、Jing Zhang 和 Dacheng Tao。采用对抗式自训练的分阶段无监督域适应，用于遥感图像道路分割。IEEE 地球科学与遥感汇刊，60:1–13，2021年。[127] Jizhizi Li、Sihan Ma、Jing Zhang 和 Dacheng Tao。隐私保护人像抠图。发表于第29届 ACM 多媒体国际会议论文集，第3501–3509页，2021年。

[128] Sihan Ma、Jizhizi Li、Jing Zhang、He Zhang 和 Dacheng Tao。《重新思考隐私保护的人像抠图》。arXiv 预印本 arXiv:2203.16828，2022年。[129] Yufei Xu、Jing Zhang、Qiming Zhang 和 Dacheng Tao。《Vitpose：用于人体姿态估计的简易视觉Transformer基线》。载于《神经信息处理系统进展》。[130] Bo Du、Jian Ye、Jing Zhang、Juhua Liu 和 Dacheng Tao。《I3cl：用于任意形状场景文本检测的实例内与实例间协同学习》。《国际计算机视觉杂志》，130(8):1961–1977，2022年。[131] Maoyuan Ye、Jing Zhang、Shanshan Zhao、Juhua Liu、Bo Du 和 Dacheng Tao。《Dptext-detr：借助Transformer中的动态点改进场景文本检测》。arXiv 预印本 arXiv:2207.04491，2022年。[132] Maoyuan Ye、Jing Zhang、Shanshan Zhao、Juhua Liu、Tongliang Liu、Bo Du 和 Dacheng Tao。《Deepsolo：让带有显式点的Transformer解码器独立完成文本定位》。arXiv 预印本 arXiv:2211.10772，2022年。[133] Armand Joulin、Edouard Grave、Piotr Bojanowski 和 Tomas Mikolov。《高效文本分类的技巧集》。arXiv 预印本，2016年。[134] Walaa Medhat、Ahmed Hassan 和 Hoda Korashy。《情感分析算法及其应用：综述》。《艾因夏姆斯工程学报》，2014年。[135] Bing Wang、Liang Ding、Qihuang Zhong、Ximing Li 和 Dacheng Tao。《一种用于基于方面的情感分析的对比式跨通道数据增强框架》。载于 COLING，2022年。[136] Qihuang Zhong、Liang Ding、Juhua Liu、Bo Du、Hua Jin 和 Dacheng Tao。《面向基于方面的情感分析多视图表示学习的知识图谱增强网络》。《IEEE知识与数据工程汇刊》，2023年。[137] Lynette Hirschman 和 Robert Gaizauskas。《自然语言问答：我们的视角》。《自然语言工程》，2001年。[138] Hanhao Qu、Yu Cao、Jun Gao、Liang Ding 和 Ruifeng Xu。《通过迭代反向推理生成可解释的证明》。载于 NAACL，2022年。[139] Behrang Mohit。《命名实体识别》。《闪米特语言自然语言处理》，2014年。[140] Di Wu、Liang Ding、Fan Lu 和 Jian Xie。《Slotrefine：用于联合意图检测与槽位填充的快速非自回归模型》。载于 EMNLP，2020年。[141] Philipp Koehn。《统计机器翻译》。剑桥大学出版社，2009年。[142] Ilya Sutskever、Oriol Vinyals 和 Quoc V Le。《使用神经网络进行序列到序列学习》。载于 NeurIPS，2014年。[143] Liang Ding、Longyue Wang 和 Dacheng Tao。《结合跨语言位置表示的自注意力》。载于 ACL，2020年。[144] Liang Ding、Longyue Wang、Di Wu、Dacheng Tao 和 Zhaopeng Tu。《用于非自回归翻译的上下文感知交叉注意力》。载于 COLING，2020年。[145] Liang Ding、Longyue Wang、Xuebo Liu、Derek F. Wong、Dacheng Tao 和 Zhaopeng Tu。《理解并改进非自回归翻译中的词汇选择》。载于 ICLR，2021年。[146] Inderjeet Mani。《自动摘要》。约翰·本杰明斯出版公司，2001年。[147] Zheng Zhang、Liang Ding、Dazhao Cheng、Xuebo Liu、Min Zhang 和 Dacheng Tao。《Bliss：通过自监督输入表示实现稳健的序列到序列学习》。arXiv 预印本，2022年。[148] David Bohm、Peter M Senge 和 Lee Nichol。《论对话》。劳特利奇，2004年。[149] Yu Cao、Liang Ding、Zhiliang Tian 和 Meng Fang。《通过嵌入增强实现高效多样化的对话生成》。载于 ICASSP，2021年。[150] Qihuang Zhong、Liang Ding、Juhua Liu、Bo Du 和 Dacheng Tao。《E2s2：面向语言理解与生成的编码增强型序列到序列预训练》。arXiv 预印本，2022年。[151] Qihuang Zhong、Liang Ding、Keqin Peng、Juhua Liu、Bo Du、Li Shen、Yibing Zhan 和 Dacheng Tao。《有效语言模型预训练与下游适配的技巧集：以 GLUE 为例》。arXiv 预印本，2023年。[152] Qihuang Zhong、Liang Ding、Yibing Zhan、Y. Qiao、Yonggang Wen、Li Shen、Juhua Liu、Baosheng Yu、Bo Du、Yixin Chen、Xinbo Gao、Chun Miao、Xiaoou Tang 和 Dacheng Tao。《通过自进化实现高效语言模型预训练与下游适配：以 SuperGLUE 为例》。arXiv 预印本，2022年。[153] Liang Ding、Di Wu 和 Dacheng Tao。《通过双向训练改进神经机器翻译》。载于 EMNLP，2021年。

[154] Changtong Zan、Keqin Peng、Liang Ding、Baopu Qiu、Boan Liu、Shwai He、Qingyu Lu、Zheng Zhang、Chuang Liu、Weifeng Liu、Yibing Zhan 和 Dacheng Tao。《Vega-MT：京东探索研究院面向 WMT22 的翻译系统》。载于 WMT，2022年。[155] Changtong Zan、Liang Ding、Li Shen、Yu Cao、Weifeng Liu 和 Dacheng Tao。《论预训练与随机初始化在资源丰富的机器翻译中的互补性》。载于 COLING，2022年。[156] Liang Ding 和 Dacheng Tao。《悉尼大学面向 WMT19 的机器翻译系统》。载于 WMT，2019年。[157] Liang Ding、Di Wu 和 Dacheng Tao。《USYD-JD 的 IWSLT2021 语音翻译系统》。载于 IWSLT，2021年。[158] Shwai He、Liang Ding、Daize Dong、Miao Zhang 和 Dacheng Tao。《SparseAdapter：一种提升适配器参数效率的简易方法》。载于 EMNLP，2022年。[159] Qihuang Zhong、Liang Ding、Juhua Liu、Bo Du 和 Dacheng Tao。《PANDA：提示迁移与知识蒸馏相结合，实现高效模型适配》。arXiv 预印本，2022年。[160] Jun Rao、Xv Meng、Liang Ding、Shuhan Qi 和 Dacheng Tao。《参数高效且便于学生学习的知识蒸馏》。arXiv 预印本，2022年。[161] Ian J. Goodfellow、Yoshua Bengio 和 Aaron Courville。《深度学习》。麻省理工学院出版社，美国马萨诸塞州剑桥，2016年。http://www.deeplearningbook.org。[162] Chaoyue Wang、Chang Xu、Chaohui Wang 和 Dacheng Tao。《用于图像到图像转换的感知对抗网络》。《IEEE图像处理汇刊》，27(8):4066–4079，2018年。[163] Ian Goodfellow、Jean Pouget-Abadie、Mehdi Mirza、Bing Xu、David Warde-Farley、Sherjil Ozair、Aaron Courville 和 Yoshua Bengio。《生成对抗网络》。《ACM通讯》，63(11):139–144，2020年。[164] Chaoyue Wang、Chang Xu、Xin Yao 和 Dacheng Tao。《进化生成对抗网络》。《IEEE进化计算汇刊》，23(6):921–934，2019年。[165] Ziqiang Li、Muhammad Usman、Rentuo Tao、Pengfei Xia、Chaoyue Wang、Huanhuan Chen 和 Bin Li。《生成对抗网络正则化与归一化的系统综述》。《ACM计算调查》，2022年。[166] Yonghao Xu、Fengxiang He、Bo Du、Dacheng Tao 和 Liangpei Zhang。《用于跨域语义分割的自集成生成对抗网络》。《IEEE多媒体汇刊》，2022年。[167] Aaron Van Den Oord、Oriol Vinyals 等。《神经离散表示学习》。《神经信息处理系统进展》，30，2017年。[168] Ali Razavi、Aaron Van den Oord 和 Oriol Vinyals。《使用 VQ-VAE-2 生成多样化高保真图像》。《神经信息处理系统进展》，32，2019年。[169] Patrick Esser、Robin Rombach 和 Bjorn Ommer。《驾驭Transformer，实现高分辨率图像合成》。载于《IEEE/CVF计算机视觉与模式识别会议论文集》，第12873–12883页，2021年。[170] Jiahui Yu、Yuanzhong Xu、Jing Yu Koh、Thang Luong、Gunjan Baid、Zirui Wang、Vijay Vasudevan、Alexander Ku、Yinfei Yang、Burcu Karagol Ayan 等。《扩展自回归模型，实现内容丰富的文本到图像生成》。arXiv 预印本 arXiv:2206.10789，2022年。[171] Aditya Ramesh、Prafulla Dhariwal、Alex Nichol、Casey Chu 和 Mark Chen。《基于 CLIP 潜变量的分层文本条件图像生成》。arXiv 预印本 arXiv:2204.06125，2022年。[172] Robin Rombach、Andreas Blattmann、Dominik Lorenz、Patrick Esser 和 Björn Ommer。《使用潜扩散模型进行高分辨率图像合成》。载于《IEEE/CVF计算机视觉与模式识别会议论文集》，第10684–10695页，2022年。[173] Minghui Hu、Chuanxia Zheng、Zuopeng Yang、Tat-Jen Cham、Heliang Zheng、Chaoyue Wang、Dacheng Tao 和 Ponnuthurai N. Suganthan。《用于同步视觉语言生成的统一离散扩散》。载于《第十一届国际学习表征会议》，2023年。[174] Zuopeng Yang、Daqing Liu、Chaoyue Wang、Jie Yang 和 Dacheng Tao。《为复杂场景生成建模图像构图》。载于《IEEE/CVF计算机视觉与模式识别会议论文集》，第7764–7773页，2022年。[175] Zhuo Chen、Chaoyue Wang、Haimei Zhao、Bo Yuan 和 Xiu Li。《D2Animator：通过对 StyleGAN 进行双重蒸馏实现高分辨率人脸动画》。载于《第30届ACM国际多媒体会议论文集》，第1769–1778页，2022年。[176] Chaoyue Wang、Chang Xu 和 Dacheng Tao。《用于跨域图像动画的自监督姿态适配》。《IEEE人工智能汇刊》，1(1):34–46，2020年。

[177] Gang Li、Heliang Zheng、Chaoyue Wang、Chang Li、Changwen Zheng 和 Dacheng Tao。3ddesigner：借助文本引导扩散模型实现逼真的三维物体生成与编辑。arXiv 预印本 arXiv:2211.14108，2022年。[178] Thomas N. Kipf 和 Max Welling。基于图卷积网络的半监督分类。发表于国际学习表征会议（ICLR），2017年。[179] Petar Veli ˇckovi ´c、Guillem Cucurull、Arantxa Casanova、Adriana Romero、Pietro Liò 和 Yoshua Bengio。图注意力网络。国际学习表征会议，2018年。已作为海报论文接收。[180] Qitian Wu、Wentao Zhao、Zenan Li、David Wipf 和 Junchi Yan。Nodeformer：一种可扩展的图结构学习Transformer，用于节点分类。发表于神经信息处理系统进展（NeurIPS），2022年。[181] W. Lu、Y. Zhan、Z. Guan、L. Liu、B. Yu、W. Zhao、Y. Yang 和 D. Tao。SkipNode：缓解深层图卷积网络过平滑问题。2021年。[182] Pinghua Xu、Yibing Zhan、Liu Liu、Baosheng Yu、Bo Du、Ji a Wu 和 Wenbin Hu。用于有符号网络嵌入的双分支密度比估计。发表于2022年ACM Web会议，第1651–1662页，2022年。[183] S. Wan、S. Pan、S. Zhong、J. Yang、Y. Zhan 和 C. Gong。用于高光谱图像分类的多层次图学习网络。《模式识别：模式识别学会期刊》，第129页，2022年。[184] Xin Lin、Changxing Ding、Yibing Zhan、Zijian Li 和 Dacheng Tao。HL-net：用于场景图生成的异配性学习网络。发表于IEEE/CVF计算机视觉与模式识别会议论文集，第19476–19485页，2022年。[185] Zhi-Hua Zhou。开放环境机器学习。《国家科学评论》，2022年。[186] David R. So、Quoc V. Le 和 Chen Liang。进化Transformer。发表于ICML，2019年。[187] Lisha Li、Kevin G. Jamieson、Giulia DeSalvo、Afshin Rostamizadeh 和 Ameet Talwalkar。Hyperband：一种用于超参数优化的新型基于多臂老虎机的方法。《机器学习研究期刊》，2017年。[188] Liam Li 和 Ameet Talwalkar。神经架构搜索中的随机搜索与可复现性。发表于UAI，2019年。[189] Hanxiao Liu、Karen Simonyan 和 Yiming Yang。DARTS：可微架构搜索。发表于ICLR，2019年。[190] Xuanyi Dong 和 Yi Yang。用四个GPU小时搜索鲁棒的神经架构。发表于CVPR，2019年。[191] Xuanyi Dong 和 Yi Yang。通过自评估模板网络进行一次性神经架构搜索。发表于ICCV，2019年。[192] Chao Xue、Xiaoxing Wang、Junchi Yan、Yonggang Hu、Xiaokang Yang 和 Kewei Sun。重新思考神经架构搜索中的双层优化：吉布斯采样视角。发表于AAAI，2021年。[193] Esteban Real、Alok Aggarwal、Yanping Huang 和 Quoc V. Le。用于图像分类器架构搜索的正则化进化。发表于AAAI，2019年。[194] Ronald J. Williams。用于连接主义强化学习的简单统计梯度跟踪算法。《机器学习》，1992年。[195] Stefan Falkner、Aaron Klein 和 Frank Hutter。BOHB：大规模鲁棒且高效的超参数优化。发表于ICML，2018年。[196] James Bergstra、Rémi Bardenet、Yoshua Bengio 和 Balázs Kégl。超参数优化算法。发表于NeurIPS，2011年。[197] Yufei Xu、Qiming Zhang、Jing Zhang 和 Dacheng Tao。ViTAE：通过探索内在归纳偏置增强视觉Transformer。《神经信息处理系统进展》，34：28522–28535，2021年。[198] Ze Liu、Yutong Lin、Yue Cao、Han Hu、Yixuan Wei、Zheng Zhang、Stephen Lin 和 Baining Guo。Swin Transformer：采用移位窗口的层次化视觉Transformer。发表于ICCV，2021年。[199] Ze Liu、Han Hu、Yutong Lin、Zhuliang Yao、Zhenda Xie、Yixuan Wei、Jia Ning、Yue Cao、Zheng Zhang、Li Dong 等。Swin Transformer V2：扩展容量和分辨率。发表于CVPR，2022年。[200] Zihang Dai、Hanxiao Liu、Quoc V Le 和 Mingxing Tan。CoAtNet：将卷积与注意力机制结合，适用于各种数据规模。发表于NeurIPS，2021年。[201] Haiping Wu、Bin Xiao、Noel Codella、Mengchen Liu、Xiyang Dai、Lu Yuan 和 Lei Zhang。CvT：将卷积引入视觉Transformer。发表于ICCV，2021年。[202] Chen Wei、Haoqi Fan、Saining Xie、Chao-Yuan Wu、Alan Yuille 和 Christoph Feichtenhofer。用于自监督视觉预训练的掩码特征预测。发表于CVPR，2022年。

[203] Zhenda Xie、Zheng Zhang、Yue Cao、Yutong Lin、包建民、Zhuliang Yao、戴琦和Han Hu。SimMIM：一种简单的掩码图像建模框架。发表于*CVPR*，2022年。[204] Liang Ding、王龙跃、Xuebo Liu、Derek F. Wong、Dacheng Tao 和 Zhaopeng Tu。重振低频词：充分利用平行数据进行非自回归翻译。发表于*ACL*，2021年。[205] Liang Ding、王龙跃、Xuebo Liu、Derek F. Wong、Dacheng Tao 和 Zhaopeng Tu。面向非自回归翻译的渐进式多粒度训练。发表于*ACL*论文集，2021年。[206] Liang Ding、王龙跃、Xuebo Liu、Derek F. Wong、Dacheng Tao 和 Zhaopeng Tu。重新分配低频词：充分利用单语数据进行非自回归翻译。发表于*ACL*，2022年。[207] Zuopeng Yang、刘大庆、Chaoyue Wang、Jie Yang 和 Dacheng Tao。面向复杂场景生成的图像构图建模。发表于*CVPR*，2022年。[208] Manuel Jahn、Robin Rombach 和 Björn Ommer。利用Transformer进行高分辨率复杂场景合成。发表于计算机视觉与模式识别会议研讨会，2021年。[209] Holger Caesar、Jasper Uijlings 和 Vittorio Ferrari。COCO-stuff：情境中的实体与背景类别。发表于*CVPR*，2018年。[210] Ranjay Krishna、Yuke Zhu、Oliver Groth、Justin Johnson、Kenji Hata、Joshua Kravitz、Stephanie Chen、Yan- nis Kalantidis、李丽佳、David A Shamma 等。视觉基因组：利用众包密集图像标注连接语言与视觉。国际计算机视觉期刊，2017年。[211] Wei Sun 和吴天辅。学习布局与风格可重构的生成对抗网络，实现可控图像合成。*IEEE*模式分析与机器智能汇刊，2021年。[212] Tristan Sylvain、Pengchuan Zhang、Yoshua Bengio、R Devon Hjelm 和 Shikhar Sharma。根据布局进行以对象为中心的图像生成。发表于*AAAI*，2021年。[213] 李泽健、吴靖宇、Immanuel Koh、唐永川和孙凌云。利用局部性感知掩码自适应，根据布局合成图像。发表于*ICCV*，2021年。[214] 范万全、陈彦君、陈东东、程宇、Lu Yuan 和王裕强。Frido：用于复杂场景图像合成的特征金字塔扩散。*arXiv* 预印本 *arXiv:2208.13753*，2022年。[215] 王大阔、Josh Andres、Justin D Weisz、Erick Oduor 和 Casey Dugan。AutoDS：迈向以人为中心的数据科学自动化。发表于*2021*年计算机系统中的人为因素大会论文集，第1–12页，2021年。[216] Wenzhuo Yang、Hung Le、Silvio Savarese 和许史文。OmniXAI：一个用于可解释人工智能的库。*arXiv* 预印本 *arXiv:2206.01612*，2022年。[217] Marc-André Zöller、Waldemar Titov、Thomas Schlegel 和 Marco F Huber。XAutoML：用于建立对自动化机器学习信任的可视分析工具。*arXiv* 预印本 *arXiv:2202.11954*，2022年。[218] Janis Klaise、Arnaud Van Looveren、Giovanni Vacanti 和 Alexandru Coca。Alibi Explain：用于解释机器学习模型的算法。机器学习研究期刊，22(181)：1–7，2021年。[219] Amazon SageMaker Canvas：无需编写代码，即可生成准确的机器学习预测。https://aws.amazon.com/cn/sagemaker/canvas/，2022年。[220] Amazon Forecast：利用机器学习轻松准确地预测业务成果。
https://aws.amazon.com/cn/forecast/，2022年。[221] Watson Natural Language Understanding：用于高级文本分析的自然语言处理服务。
https://www.ibm.com/cloud/watson-natural-language-understanding，2022年。[222] Abacus AI：轻松大规模创建尖端人工智能系统。https://abacus.ai/，2022年。[223] 自动化机器学习：快速、大规模地自动构建机器学习模型。
https://azure.microsoft.com/en-us/services/machine-learning/automatedml/，2022年。[224] Vertex AI：在统一的人工智能平台中，利用预训练工具和定制工具，更快速地构建、部署和扩展机器学习模型。https://cloud.google.com/vertex-ai，2022年。