intel

英特尔中国物流与交通AI实战手册

Contents
目 录

趋势篇

全球经济的高速发展，推动着物流与交通产业在人们的生产生活、贸易交流中承担起更为重要的作用。物流产业是融合运输、仓储和信息流，通过物流资源聚合来推动物品从供应地向接收地流动的复合产业。而交通运输业是通过各类交通工具和路网设施，让人与物品在不同的空间和位置进行转移。在今天的千行百业中，无论是工业制造、商品贸易等行业中的产业链、供应链管理，还是人文旅游、电商零售等领域中的产创融合、服务提升，都离不开物流与交通产业对海量人流、物流和信息流的支撑，物流与交通正成为不同行业、不同区域之间的有效连接纽带。

作为社会经济发展的“大动脉”，物流与交通为经济的高速增长、城市的繁荣发展、人们的便利生活提供了强有力的运行基座。随着更多变革与转型需求在经济发展进程中出现，其也正面临一系列新的机遇与挑战。主要的挑战体现在以下两个方面：

- **效率提升**：物流与交通产业规模正以令人惊叹的速度扩展。在交通领域，至 2023 年 8 月，中国机动车保有量已达 4.3 亿辆[1]。而在物流行业，中国 2023 年上半年快递业务量将超 600 亿件，同比增速超 17%[2]。传统以人力为主的业务处理与管理模式，在如此巨大的业务量面前，势必面临巨大挑战，带来城市拥堵、交通运输延误、快递错发、漏发等问题。
- **成本控制**：物流与交通产业规模的扩大，同时也带来了巨大的成本控制压力。一方面，在人员成本日益高涨的趋势下，传统人力为主的模式必然会带来成本的急剧增长。另一方面，在交通管理和物流处理模式日趋多样化、差异化的前提下，更大规模的交通和物流处理需求，也意味着更大的系统建设和设备维护成本。

巨大的挑战也意味着更多的机遇。可以看到，要应对上述交通与物流面临的挑战，核心目标是实现全方位的降本增效。这也为新技术、新产品和新方案的引入提供了契机与动力。这一过程中，得益于信息科技的进步与人工智能（Artificial Intelligence，AI）技术的发展，更多物流与交通领域的用户正借助计算机、互联网、大数据、云计算以及 AI 等前沿 IT 能力，以更细的颗粒度去推进物流与交通领域各个流程的优化，实现能力与管理创新，最终实现效率的大幅提升。

在物流行业，越来越多的物流企业正通过 AI 与新一代自动化、仓储、光学字符识别（Optical Character Recognition，OCR）和运输技术的融合，提升揽收、出入库、分拣、运输及派送等各个环节的效率。例如在揽收、派送等环节，基于 AI 的 OCR 技术正与互联网应用结合，加速信息核验等繁琐的流程。在运输环节，物联网、大数据和 AI 等技术的融合可用于路径规划，提高运输效率和精度。在出入库环节，智能机器人正取代人工，高速准确地完成包裹出入库、上下货架流程，实现更具效率的库存管理。而在以往极为耗费人力，且最易出现错误的分拣环节，结合 OCR 和自动化技术的智能分拣设备，可实现快速、准确地将不同重量、品类、形状的包裹进行分类和分拣。

不断迭代的技术与日益高涨的需求也正推动智能物流市场行驶在快车道。数据表明，在未来五年内，智能物流市场预计将以超过 3% 的复合年增长率实现快速增长[3]，从而满足高速扩容的物流行业所需。

而在交通领域，以 AI 为代表的信息化前沿技术也正成为交通安全、车路协同、缓解拥堵以及交通管理等方向上实现数字化、智能化转型的重要抓手。例如对于城市中常面临的交通拥堵问题，由 AI 驱动的交通流量预测系统能根据海量数据建模，对未来路况做出实时预测，进而通过动态控制交通信号、交通运输路线优化等方法来缓解高峰期交通压力。又例如基于 AI 的视频分析处理技术已能实现对监控视频图像和数据的高效分析和处理。这一技术与交通监控系统相融合，可在不需要人为干预的情况下，通过图像自动分析，对监控场景中的变化进行定位、识别和跟踪，实现对道路交通事件的自动监控。同时也能根据车流量、车速、车型、突发事件紧急程度进行预测分析，为道路安全运行与危险情况营救提供必要的数据支持。

得益于智能交通所能带来的巨大收益，智能交通系统也受到了市场的广泛认可。数据显示，2023 年，全球智能交通系统市场规模将达到 205.2 亿美元，预计在预测期内将以 8.6% 的复合年增长率增长[4]。

[1] 数据援引自公开信息：http://www.scio.gov.cn/xwfb/bwxwfb/gbwfbh/gab/202308/t20230804_749646.html
[2] 数据援引自公开信息：https://www.gov.cn/govweb/lianbo/bumen/202307/content_6891636.htm
[3] 数据援引自 Mordor Intelligence 发布的《HIGH-TECH LOGISTICS MARKET SIZE & SHARE ANALYSIS - GROWTH TRENDS & FORECASTS (2023 - 2028)》：https://www.mordorintelligence.com/industry-reports/global-high-tech-logistics-market
[4] 数据援引自 globaldata 发布的《Intelligent Transportation Systems (ITS) Market Size, Share, Trend and Analysis by Region, System Type (ATMS, ATIS, ATPS, APTS, CVO, Others), Application (Intelligent Traffic Control, Road Safety and Security, Freight Management), and Segment Forecast, 2023-2026》：https://www.globaldata.com/store/report/intelligent-transportation-systems-market-analysis/

在技术发展与市场培育之余，物流与交通的智能化、数字化进程也获得了政策层面的大力支持。以智能交通为例，交通运输部与工业和信息化部、国家标准化管理委员会在 2021 年 3 月联合印发了《国家车联网产业标准体系建设指南（智能交通相关）》，推进先进技术在智能交通领域的应用，促进自动驾驶和车路协同技术应用和产业健康发展。而交通运输部在 2021 年 12 月印发的《数字交通“十四五”发展规划》中也提出，到 2025 年，基本建成“一脑、五网、两体系”的发展格局，交通新基建取得重要进展，行业数字化、网络化、智能化水平显著提升。

目前的物流与交通领域中，两个方向上的 AI 应用正备受关注，分别是基于 AI 视觉的各类智能物流、交通系统以及 AI 与大数据相融合的智能预测系统。得益于视频采集硬件、深度学习算法以及跨平台算力的蓬勃发展，基于 AI 的计算机视觉应用正在物流与交通系统中获得广泛地运用。与人工方式相比，AI 视觉能力的优势包括：

- **准确度更高**：得益于强大的自动特征学习能力，AI 视觉能从各类原始数据中学习到更广泛的语义知识，在复杂环境下对物流交通场景中的细节有着更好的捕捉，与人眼观察等方式相比，识别准确度更高；
- **工作效率更高**：与人工方式相比，基于强劲算力支持的 AI 视觉能高效处理海量的视频、图像数据，并高速、稳定地完成目标检测、目标识别及目标跟踪等任务，且不会出现因疲劳等因素引发的效率下降问题；
- **适应性更好**：AI 视觉系统能与物联网、机器人、自动驾驶、道路指挥、安防等系统实现数据互联互通，在人力无法工作的环境、时间和场景中长时间发挥作用。

而 AI 与大数据相融合的智能预测系统，则能帮助物流与交通领域在预测性维护、资源协同等方面获得先机。在解决方案的实际运行过程中，企业可获得以下的收益：

- 物流企业能通过对不同时间、不同区域内包裹量的预测，灵活调配物流工作人员和设备，降低电商促销活动等高峰期间的爆仓风险；
- 物流企业能通过预测性维护技术对物流设备、节点予以及时保障、更新和替换，保证各个设备和节点都处于高可用状态，提升物流整体效率；
- 城市与交通管理部门能通过交通流量预测，更准确地评估各类市政交通设施的优化空间，并通过与交通管理系统的联动来实现拥堵和事故率的下降；
- 交通运输企业也能通过道路情况预测，来制定更具效率的运输策略，降低因延误带来的经济损失。

实现上述方案离不开强有力软硬件基础设施提供支持。来自英特尔的一系列先进产品与技术，包括英特尔® 至强® 可扩展平台、英特尔® 数据中心 GPU Flex 系列、英特尔® DL Streamer、英特尔® Deep Link Workload Scheduler、OpenVINO™ 工具套件以及 BigDL 2.0 等，都成为了相关方案成功实施和运行的坚实基础。通过这些高性能硬件基础设施和软件框架，物流与交通领域的用户可以方便地联动不同平台的算力，构建高效的端到端媒体处理管道，并对 AI 工作任务实现有效优化，从而帮助用户打造更具战斗力的物流和交通 AI 应用。

在下一部分中，本文将结合多个物流与交通领域中的 AI 应用场景，共同探讨英特尔相关技术与产品在这些真实场景中的应用和优化方案。

实战篇

AI 视觉处理，加速物流自动化进程

物流自动化中的 AI 视觉处理

自动化技术正在物流行业全流程发挥作用

随着物流行业规模的不断增长，自动化技术正在物流行业全流程中发挥越来越重要的作用。传统小车推送、人力分拣的人工作业模式，不仅费时费力，也无法杜绝错送、漏拣等问题发生，无法进一步提升物流整体效率。当出现节假日、电商营销等购物高峰时，就容易出现挤压、爆仓等问题。同时，人力成本的持续上涨也在另一方面推动物流企业部署更多自动化设备来降本增效。

物流自动化是指通过前沿的信息化系统和 IT 技术，对物流系统的各个流程进行规划、协调和管理，并尽可能地使用各类机械、光电和运输设备代替人工操作，高效快速、精准可靠地完成物流端到端全过程。

■ **物流自动化系统一般包括：**

- 自动检测系统；
- 自动分拣系统；
- 自动输送系统；
- 自动跟踪系统等。

■ **常见的设备包括：**

- **自动导引车（Automated Guided Vehicle，AGV）：**AGV 小车能根据预设路线和任务，自动完成货物输送、存储等操作；
- **搬运车 / 机器人：**搬运车 / 机器人可通过与条形码、射频识别（Radio Frequency Identification，RFID）等技术结合实现货物搬运、转运以及存储等；
- **智能输送线：**智能输送线可根据货物大小、属性以及物流信息等，将货物分门别类地传输到不同的分拣、打包或装车位置；
  智能分拣机：智能分拣机能够根据预设分拣规则，通过对物流信息的识别，自动将其分拣到正确位置。

拥有了大幅提升物流效率和有效压降人力成本方面的亮眼表现，物流自动化在近年来得到了相关企业用户的高度青睐。一些市场数据表明，2023 年，物流自动化市场规模预计为 684.6 亿美元，预计到 2028 年将达到 1097.6 亿美元，在预测期内（2023-2028 年）会以 9.90% 的复合年增长率增长。[5]

基于 AI 视觉的物流自动化

传统的自动化设备通常采用预设规则，或通过机械、光电设备对货物进行测量并分拣。这些方式存在如下弊端：

- 易出现未检测出货物或识别错误。电商行业的大发展使物流包裹越来越纷繁复杂，导致传统的机械、光电设备无法准确检测货物的大小、类别等，从而导致分装不合理、包装错误等情况发生；
- 传统方法在检测速度上存在短板，在一些物流环节上仍然无法获得令人满意的速度。

解决以上问题的有效途径是提升物流自动化各个环节中的识别精确度和速度。基于 AI 视觉来提升物流自动化的效率，是目前物流行业的共识之一。一方面，通过与高清摄像头相结合，日趋成熟的 AI 视觉应用能够全方面介入物流各环节；另一方面，强劲算力的加入能帮助自动化系统大幅提升处理效能，减少延迟。

图 2-1 基于 AI 视觉的物流效能提升

这其中，通过 AI 视觉来推动物流自动化，提升物流全流程效能，目标检测（Object Detection）无疑是其中的核心环节。借助基于 AI 视觉的目标检测，物流自动化系统能够对输送带、货架等处的快递件、包装箱以及物流包裹等进行大小件测量、体积评估等，再配合 OCR、条形码以及 RFID 等技术，完成物流自动化全流程。

例如在快递件的体积测量上，就可以通过 AI 视觉对快递件的体积进行识别和预判。通过装备在分拣、输送系统上的高清摄像头，AI 视觉应用能采集快递件的图像信息，并传送到后端进行测量。测量完成后，数据返回至输送系统，并据此将不同大小的快递件送至合适的分拣和装车处。由此各环节工作人员可以得到更为合理的分拣、装车和配送规划，来降低单件配送时长及成本。

[5] 数据援引自 Mordor Intelligence 发布的《LOGISTICS AUTOMATION MARKET SIZE & SHARE ANALYSIS - GROWTH TRENDS & FORECASTS (2023 - 2028)》：https://www.mordorintelligence.com/industry-reports/logistics-automation-market

事实上，在今天的物流自动化系统中，各个环节以及各类自动化设备，包括 AGV 小车、智能输送线以及智能分拣机等，都逐渐转向以 AI 视觉为核心的技术基座。

AI 视觉任务中常见的目标检测模型

各类基于深度学习的目标检测算法一直是 AI 视觉领域重要的技术发展方向。一般来说，目标检测算法主要有两步法（Two-stage）和单步法（One-stage）。相对而言，One-stage 模型在处理速度上更具优势，在“寸时寸金”的物流行业中更受青睐。目前常见的网络模型包括 RetinaNet，SSD（Single Shot MultiBox Detector）以及 YOLO（You Only Look Once）系列等。下文将对目前常用的 YOLO v4 模型做一个简单介绍。

YOLO v4 在原有 YOLO 目标检测模型的基础上加入了多种新的优化策略，其核心是深度卷积神经网络，使用单个神经网络对输入图像进行处理，得到所有目标的边界框和类别概率，从而可在物流自动化场景中对目标予以实时高精度检出。

与之前的目标检测模型相比，YOLO v4 有着以下几方面的优势：

1. 其提出了一个更为高效的目标检测模型，且对硬件基础设施要求较低；
2. 验证了 BoF（Bag-of-Freebies）和 BoS（Bag-of-Specials）两类目标检测优化方法在检测器训练时的作用；
3. 对传统目标检测算法中的激活函数、归一化方法以及空间注意力机制，包括 ReLU、CBN（Cross-Iteration Batch Normalization）、PAN（Path Aggregation Network）和 SAM（Spitial Attention Mask）等进行了改良。

YOLO v4 的网络结构中，Backbone 由全新的主干网路结构 CSPDarknet53 构成。Neck 由 SPP、PAN 组成，而 Head 由 YOLO v3 构成。

YOLO v4 采用了两类优化的目标检测优化方法 BoF 和 BoS。BoF 包括了不增加模型复杂度，也不增加推理计算量，但可以提高模型准确度的训练方法，例如 CutMix、Mosaic 数据增强等。BoS 包括了增加少许模型复杂度或计算量，但可以显著提高模型准确度的训练方法，例如 Mish 激活函数等。

更多 YOLO v4 模型介绍，请参考：
YOLOv4: Optimal Speed and Accuracy of Object Detection: https://arxiv.org/abs/2004.10934。

英特尔产品与技术为 AI 视觉提效

从前文基于 AI 视觉的物流自动化工作流程可以看出，系统一方面需要具备高效并行多线程处理物流环节中所采集的视频图像的能力，同时也需要有应对目标检测工作负载所需的足够算力。来自英特尔的英特尔® 数据中心 GPU Flex 系列，能在基于英特尔® 架构的平台（例如第三 / 第四代英特尔® 至强® 可扩展处理器、英特尔® 至强® CPU Max 系列等）或其它兼容平台上，为目标检测工作负载提供强劲的算力支持。

与此同时，英特尔也在英特尔® 分发版 OpenVINO™ 工具套件的基础上，推出英特尔® DL Streamer 流水线框架来便捷地构建 AI 视觉端到端流程，加速 AI 模型推理。下文给出了英特尔® DL Streamer 的安装优化配置示例。同时，借助英特尔® Extension for PyTorch（IPEX）等优化框架，也可以提升 AI 视觉模型在各类物流实战中的效率。下文也同样给出了相关的安装配置和验证示例。

英特尔® DL Streamer

■ 安装英特尔® DL Streamer

英特尔® DL Streamer 流水线框架是基于英特尔® 分发版 OpenVINO™ 工具套件来构建媒体分析流水线的一种简单易用的工具套件。其一方面引入了开源媒体框架 GStreamer 提供的媒体优化操作，另一方面也加入了 OpenVINO™ 深度学习推理引擎提供的推理优化工具套件。

英特尔® DL Streamer 提供了 API 接口和命令行（Command Line Interface，CLI）工具，用户在前期性能评估和验证阶段可使用 CLI 命令行模拟线上业务流水线。不同的参数都有着其不同的功能定义，包括：

- **decodebin**：通过自动插入，通过可用的解码器和分流器构建一个解码 pipeline；
- **vaapipostproc**：包含各类应用于 vasurface 的后处理算法，如用于缩放、去隔行扫描（抖动、运动自适应、运动补偿）、降噪或锐化；
- **gvadetect**：使用目标检测模型，如 YOLO v3-v5、MobileNet SSD、Faster RCNN 等在整帧或感兴趣区域（Region Of Interest，ROI）上执行对象检测，输出检测对象的 ROI；
- **gvatrack**：使用零项跟踪（Zero-term tracking）、零项无图像（zero-term-imageless）或短期无图像跟踪（short-term-imageless tracking algorithms）算法执行目标跟踪。零项跟踪指定唯一的对象 ID，并需在每帧上运行目标检测。短期跟踪允许在帧之间跟踪目标，从而减少在每帧上运行目标检测的需要。无图像跟踪基于目标的运动和形状形成对象目标，且不使用图像数据；
- **gvafpscounter**：测量单个进程中多个视频流的每秒帧数。

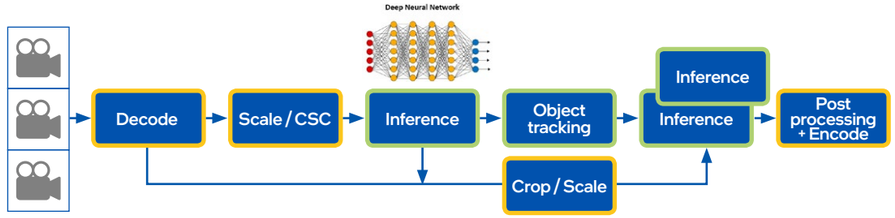


图 2-2 英特尔® DL Streamer 流水线框架

除了对模型的推理之外，其还包含了 bounding box 等目标检测网络的后处理操作，能帮助更加高效地构建流水线。以下是安装英特尔® DL Streamer 并验证性能的参考步骤。

■ 基于 Docker 在英特尔® 数据中心 GPU Flex 系列上部署英特尔® DL Streamer

Docker 是一个开源容器平台，让用户以标准化的可执行组件将应用程序源代码与操作系统（OS）库，以及在任何环境中运行该代码。所需的依赖项相结合，从而便捷地构建、部署、运行、更新和管理容器。

 # 步骤 1

安装英特尔® DL Streamer 和 OpenVINO™ 工具套件。在面向英特尔® 数据中心 GPU Flex 系列推出的镜像包（2022.3.0-ubuntu22-gpu555 * ）中包括了：

- 英特尔® DL Streamer 2022.3.0；
- OpenVINO™工具套件 2022.3.0；
- 英特尔® 数据中心 GPU Flex 系列驱动软件，版本号 555。

其中，运行带有 –devel 的镜像包有附加样例和开发文件，运行带有 -dpcpp 的镜像包带有英特尔® oneAPI DPC++/C++ 编译器。

用户根据需求，运行以下不同的镜像包。

```
1. docker pull intel/dlstreamer:2022.3.0-ubuntu22-gpu555
2. docker pull intel/dlstreamer:2022.3.0-ubuntu22-gpu555-devel
3. docker pull intel/dlstreamer:2022.3.0-ubuntu22-gpu555-dpcpp
4. docker pull intel/dlstreamer:2022.3.0-ubuntu22-gpu555-dpcpp-devel
```

 # 步骤 2

在安装英特尔® DL Streamer 结束后，下文将以步骤示例展示面向 YOLO v4 目标检测模型构建 DL Streamer pipeline。用户可从 https://www.pexels.com/photo/5325136/download 下载视频、并从 GitHUB 克隆 yolov4-tf 文件。

```
1. git clone https://github.com/dlstreamer/pipeline-zoo-models.git
```

进入 Docker，并将文件复制到 Docker 容器中。

```
1 docker run -it --device /dev/dri --user root --rm intel/dlstreamer:2022.3.0-ubuntu22-gpu555-dpcpp
```

打开另一个终端（terminal），将文件复制到容器中，将视频和模型复制到 docker 容器中。

```
1. docker cp pexels-george-morina-5325136.mp4 <DockerContainerID>:/home/dlstreamer
2. docker cp pipeline-zoo-models/storage/yolo-v4-tf_INT8/ <DockerContainerID>:/home/dlstreamer
```

运行 AI 视觉目标检测 pipeline。

```
1. gst-launch-1.0 filesrc location=/path/to/pexels-george-morina-5325136-1080p.mp4 ! decodebin ! capsfilter caps="video/x-raw(memory:VASurface)" ! gvadetect model=/path/to/yolo-v4-tf_INT8/yolo-v4-tf_INT8.xml model_proc=/path/to/yolo-v4-tf_INT8/yolo-v4-tf_INT8.json inference-interval=1 device=GPU batch-size=32 pre-process-backend=vaapi-surface-sharing ! queue ! gvatrack tracking-type=short-term-imageless ! gvafpscounter ! fakesink sync=false
```

如果用户想对检测结果进行编码并保存为视频文件，可以使用以下脚本。

```
1. gst-launch-1.0 filesrc location=/path/to/pexels-george-morina-5325136-1080p.mp4 ! decodebin ! capsfilter caps="video/x-raw(memory:VASurface)" ! gvadetect model=/path/to/yolo-v4-tf_INT8/yolo-v4-tf_INT8.xml model_proc=/path/to/yolo-v4-tf_INT8/yolo-v4-tf_INT8.json inference-interval=1 device=GPU.0 batch-size=32 pre-process-backend=vaapi-surface-sharing ! queue ! gvatrack tracking-type=short-term-imageless ! meta_overlay device=GPU ! gvafpscounter ! vaapipostproc ! vaapih265enc rate-control=cbr bitrate=4096 ! filesink location=./encoded_video_track.265 sync=false
```

编码后的视频文件将保存在容器中，并可以在新的终端中复制出来。

```
1. 1. docker cp <DockerContainerID>:/home/dlstreamer/encoded_video_track.265 ./
```

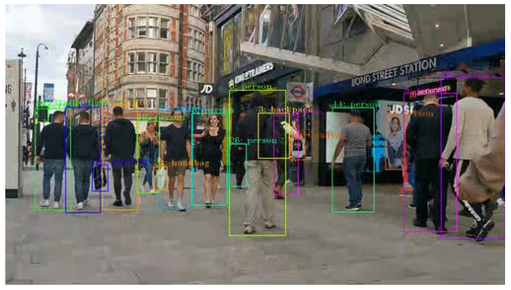


图 2-3 面向 YOLO v4 目标检测模型构建 DL Streamer pipeline 检测结果

值得一提的是，工作任务在图像解码之后、连接模型推理之前，可能需要对解码后的图像进行预处理（包括 resize，颜色空间转换等），为提高 pipeline 的整体效率，减少内存 / 显存拷贝和数据通过 PCIe 总线传输时的开销，该类操作仍然需要计算运行在 GPU 上。在 OpenVINO™ 工具套件模型推理模块中使用 pre-process-backend=vaapi-surfacesharing 同时在 decodebin 插件后面使用过滤器 "video/x-raw(memory:VASurface)"。该模块将使用到 OpenVINO™ 工具套件的 remote tensor API，把解码后的图片地址传给 opencl context，并可以将图片预处理操作编译集成到模型结构中进行推理，这样就避免了数据拷贝到内存中去的过程，减少 Host to Device（H2D）和 D2H 的开销，提高 pipeline 的整体性能。

■ 使用源码包安装英特尔® DL Streamer 并验证性能

同时，用户也可以从 GitHUB 克隆资源包并安装英特尔 ® DL Streamer，以下是安装和验证性能的参考步骤。

 # 步骤 1

安装所需的软件包。

```
$ sudo apt-get update &&sudo apt-get install curl gpg-agent software-properties-common
$ sudo apt-get update &&sudo apt-get install intel-dlstreamer-dev
```

 # 步骤 2

从 GitHUB 克隆资源包，并切换到英特尔® DL Streamer 所在目录。

```
$ mkdir -p ~/intel
$ git clone https://github.com/dlstreamer/dlstreamer.git ~/intel/dlstreamer_gst
$ cd ~/intel/dlstreamer_gst
```

 # 步骤 3

安装 OpenVINO™ 工具套件。

```
$ curl https://apt.repos.intel.com/intel-gpg-keys/GPG-PUB-KEY-INTEL-SW-PRODUCTS.PUB |sudo apt-key add - echo "deb https://apt.repos.intel.com/openvino/2022 `. /etc/os-release && echo ${UBUNTU_CODENAME}` main"|sudo tee /etc/apt/sources.list.d/intel-openvino-2022.list
$ source <OPENVINO_INSTALL_PATH>/ov_dist/setupvars.sh
```

 # 步骤 4

安装英特尔® DL Streamer 以及所需依赖项，安装 Python 所需依赖项。

```
$ sudo apt-get install -y --no-install-recommends intel-dlstreamer-gst cmake build-essential libpython3-dev python-gi-dev
$ cd ~/intel/dlstreamer_gst
$ python -m pip install --upgrade pip
$ python -m pip install -r requirements.txt
```

 # 步骤 5

安装 OpenCL、消息代理以及英特尔® oneAPI DPC++/C++ 编译器（非必选项），安装前需要先注册英特尔® 图形 APT 仓库（repository）和英特尔® oneAPI APT 仓库。

```
$ curl -sSL https://repositories.intel.com/graphics/intel-graphics.key |sudo apt-key add - && sudo apt-add-repository 'deb [arch=amd64] https://repositories.intel.com/graphics/ubuntu focal main'
$ sudo apt-get update &&sudo apt-get install -y intel-opencl-icd
$ cd ~/intel/dlstreamer_gst/scripts/
$ sudo ./install_metapublish_dependencies.sh
$ curl -ssL https://apt.repos.intel.com/intel-gpg-keys/GPG-PUB-KEY-INTEL-SW-PRODUCTS.PUB |sudo apt-key add - &&\sudo add-apt-repository "deb https://apt.repos.intel.com/oneapi all main"
$ sudo apt-get update &&sudo apt-get install -y intel-oneapi-compiler-dpcpp-cpp intel-level-zero-gpu level-zero-dev
$ source /opt/intel/oneapi/compiler/latest/env/vars.sh
```

 # 步骤 6

安装 VA-API 所需依赖项并验证。

```
$ sudo apt-get install -y intel-dlstreamer-gst-vaapi libva-dev vainfo intel-media-va-driver-non-free
$ export LIBVA_DRIVER_NAME=iHD
$ vainfo
```

 # 步骤 7

构建并编译英特尔® DL Streamer pipeline 源代码。

```
$ source <OPENVINO_INSTALL_PATH>/ov_dist/setupvars.sh
$ source /opt/intel/dlstreamer/gstreamer/setupvars.sh
$ source /opt/intel/oneapi/compiler/latest/env/vars.sh
$ mkdir ~/intel/dlstreamer_gst/build
$ cd ~/intel/dlstreamer_gst/build
$ cmake -DCMAKE_INSTALL_PREFIX=/opt/intel/dlstreamer ..
$ make -j
$ sudo make install
$ source ~/intel/dlstreamer_gst/scripts/setup_env.sh
```


在完成上述英特尔® DL Streamer 的构建后，用户可通过运行 DL Streamer pipeline 基准测试（benchmark）来对其性能进行评估，以便开展下一步的方案设计与推进。

基准测试可按照以下步骤展开。

 # 步骤 1

初始化环境并检查硬件配置。

```
$ source ~/1-virtualenv/py3.8-ov2.0-ats-env/bin/activate
$ source ~/8-sourcecode/dlstreamer_performance_eval/va_env.sh
$ source ~/8-sourcecode/dlstreamer_performance_eval/source env_2022_2.sh
$ vainfo
$ clinfo
```

 # 步骤 2

对基准测试、DL Streamer pipeline 进行评估。

```
$ cd ~/8-sourcecode/openvino/ov_dist/samples/cpp
$ ./inference_cv2_demo/intel64/Release/benchmark_app -m ~/5-model/yolov5/yolov5s/FP32/yolov5s.xml -d GPU -t 5
$ cd ~/8-sourcecode/dlstreamer_performance_eval
$ ./benchmark_yolov4.sh <VIDEO_FILE> <DECODE_DEVICE> <INFERENCE_DEVICE> <VIDEO_CHANNEL_COUNT> <PROCESSE_COUNT> <ENCODE_FLAG>
```

 # 步骤 3

运行基准测试。

```
$ ./benchmark_yolov4.sh /root/6-dataset/pexels-george-morina-5325136_8.mp4 GPU GPU 30 2 ENCODE
$ ./benchmark_yolov4.sh /root/6-dataset/pexels-george-morina-5325136_8.mp4 GPU GPU 361
```

 # 步骤 4

监控运行状态。

```
$ htop
$ sudo ~/8-sourcecode/atsm-va-sample-demo/tools/intel_gpu_top
$ sudo watch -n1 /opt/xpum/bin/xpumcli stats -d 0
```

英特尔® PyTorch 扩展包（Intel® Extension for PyTorch，IPEX）

作为最常用的深度学习框架之一，PyTorch 可基于动态计算图的方式实现神经网络的构建和优化，具有灵活、高效和易用等优势，在 AI 视觉领域有着广泛的运用。为有效提升 PyTorch 在英特尔® 架构平台上的运行性能，英特尔结合平台的软硬件与架构特性，通过 IPEX 来提供多项优化和性能增强，包括基于英特尔® AVX-512_VNNI 和英特尔® AMX 等新功能特性和优化技术。IPEX 已作为开源项目发布于 https://github.com/intel/intel-extension-for-pytorch。为了让 IPEX 在实战中发挥更优效能，用户可通过对 IPEX 的优化配置和使用来使其发挥更大优势。

■ 安装 IPEX

 # 步骤 1

在安装 IPEX 之前，首先需要部署 PyTorch（用户可访问 PyTorch 官网：https://pytorch.org/）。目前，英特尔® oneDNN 已经集成至 PyTorch 的正式版本中，因此无需额外的安装步骤就可以在英特尔® 至强® 可扩展平台上实现性能提升。

```
pip3 install torch torchvision torchaudio --extra-index-url https://download.pytorch.org/whl/cpu
```

 # 步骤 2

用户可使用以下任意命令安装 IPEX。

```
python -m pip install intel_extension_for_pytorch
python -m pip install intel_extension_for_pytorch -f https://software.intel.com/ipex-whl-stable
```

 # 步骤 3

在使用 IPEX 前，用户需要先通过代码微调来使其支持 PyTorch 的 imperative 和 TorchScript 两种模式。用户进行模型推理时，在模型对象中应用 ipex.optimize 函数，进行训练时，在模型对象和优化器对象中应用 ipex.optimize 函数。以下展示了用 BF16 / FP32 两种不同数据类型进行训练的代码示例。

```
import torch
import intel_extension_for_pytorch as ipex

model = Model()
model = model.to(memory_format=torch.channels_last)
criterion = ...
optimizer = ...
model.train()
# For FP32
model, optimizer = ipex.optimize(model, optimizer=optimizer)
# For BF16
model, optimizer = ipex.optimize(model, optimizer=optimizer, dtype=torch.bfloat16)
# Setting memory_format to torch.channels_last could improve performance with 4D input data. This is optional.
data = data.to(memory_format=torch.channels_last)
optimizer.zero_grad()
output = model(data)
```

■ 基于 PyTorch 的深度学习模型训练和推理优化

尽管 PyTorch 和 IPEX 的默认原语都已进行了高度优化，但用户仍然可通过额外配置选项来进一步提升性能。通常情况下，用户可通过能够自动设置配置选项的启动脚本（launch script）来应用，这些配置选项主要针对下列内容：

- **OpenMP 库：**[英特尔® OpenMP 库（默认）| GNU OpenMP 库]
- **内存分配器：**[PyTorch 默认内存分配器 | Jemalloc | TCMalloc（默认）]
- **实例数量：**[单个实例（默认）| 多个实例]

除启动脚本外，其他软硬件设置包括英特尔® 处理器设置、非一致性内存访问（Non Uniform Memory Access，NUMA）配置等也可借助软件配置，利用 Channels Last 内存格式、OpenMP 以及 numactl 等软件与工具，用户在 IPEX 的支持下能充分挖掘处理器的计算资源以提升性能。

使用低精度的数据类型，如 BF16、INT8 等，可以让计算工作负载更轻、内存使用空间更小，进而加速 AI 模型的推理速度。英特尔® 至强® 可扩展处理器系列对低精度数据类型具有良好的支持，例如第四代英特尔® 至强® 可扩展处理器可基于英特尔® DL Boost 和英特尔® AMX 所提供的 AVX - 512_BF16、AVX - 512_VNNI 和 AMX_BF16、AMX_INT8 等多种指令来加速 AI 模型。

这其中，IPEX 支持的 torch.float16 或 torch.bfloat16 等低精度浮点数类型可为深度学习工作负载带来诸多裨益。用户也可以在 IPEX 的协助下实现自动混合精度（Automatic Mixed Precision，AMP），IPEX 提供的 Torch.cpu.ampkezai 让运行期间的数据类型自动转换更易实现，自动调优各算子间的数据类型转换。

以 AMX_BF16 为例，其启用首先需要使用 lscpu 命令确认处理器平台是否支持 AMX_BF16 指令。Torch.cpu.amp.autocast 允许各脚本域以混合精度运行。在这些域内，所有操作均能够以 autocast class 所选择的数据类型运行，进而在保证准确性的同时提高性能。以下示例代码简要展示了采用混合精度实现加速的网络模型（SimpleNet）。

```
class SimpleNet(torch.nn.Module):
  def __init__(self):
    super(SimpleNet, self).__init__()
    self.conv = torch.nn.Conv2d(64, 128, (3, 3), stride=(2, 2), padding=(1, 1), bias=False)
  def forward(self, x):
    return self.conv(x)
```

作为一个上下文管理器，Torch.cpu.amp.autocast 可让脚本域以混合精度运行。例如，AMX_BF16 是比 AVX - 512_BF16 更新的版本，具有更多高级内联函数（intrinsics），并可提供更加出色的性能来支持 AI 应用。因此，对于 BFloat16 而言，AMX_BF16 拥有最高的执行优先级。IPEX 会优先选择 AMX_BF16，次优选择 AVX - 512_BF16。

```
model = SimpleNet().eval()
x = torch.rand(64, 64, 224, 224)
with torch.cpu.amp.autocast():
  y = model(x)
```


实战案例

中通快递基于英特尔产品，在物流自动化系统中打造更高性价比 AI 视觉应用

■ 项目背景

通过更先进、易用的技术方案来优化物流分拣、运输和管理作业，是中通快递（下文简称为“中通”）一贯以来的宗旨。随着业务的快速发展以及技术应用的不断深入，中通对其物流自动化系统中的 AI 视觉方案也提出了更高的要求。

例如在分拣作业中，自动化系统需借助 AI 视觉技术来识别和分析分拣过程中可能存在的小件堵包、流水线拥堵和挂包等情况，以做到“实时发现、实时告警”，进而降低错分率，并减少因错分造成的错派、人工核对成本增加和时效性低等问题。而在装卸车管理方面，则需要利用 AI 视觉技术来检测车辆到达与发车的准时性、装载率以及装卸车作业规范等，以保证时效和降低运输成本。

针对以上需求，英特尔与中通一起，基于其既有的网络视频录像机（Network Video Recorder，NVR）平台，通过实时流传输协议（Real Time Streaming Protocol，RTSP）拉流输入视频，通过高效视频编解码和 AI 检测 pipeline 的构建，实现上述物流自动化系统中的 AI 视觉方案。

■ 解决方案

新方案的核心理念是以英特尔® 数据中心 GPU Flex 170 产品为基础，使用英特尔® DL Streamer 移植和搭建视频业务流。英特尔® DL Streamer 一方面可以使用英特尔® 数据中心 GPU 产品的 fix function 硬件单元来进行编解码加速以及 video frame 处理，另一方面也可使用 OpenVINO™ 工具套件，将深度学习模型部署到 GPU 产品的 X^e Core 计算单元中进行 AI 推理加速。英特尔® DL Streamer 将这两部分的接口和调用封装到 gstreamer 的 elements 中，通过连接 gstreamer 的 elements 来构建完整的视频编解码与 AI 推理的 pipeline。

在完成前期性能分析后，英特尔与中通一起开展了项目的集成和迁移工作。例如双方协作将 OpenCV 实现的光流操作封装到一个自定义的 element 中，并且通过 OpenCL 或 SYCL 的接口来进行 buffer 的调用。

在新方案中，英特尔® 数据中心 GPU Flex 170 产品提供了优异的 media-AI pipeline 整体性能，与其它产品相比编解码性能更优。同时，方案中还引入了 OpenVINO ™ 工具套件中的多项优势能力，例如 training extension 功能来优化压缩模型，模型量化工具 POT 来实现 INT8 模型低精度优化，以及 OpenVINO Model Server 来部署线上服务等。

■ 方案成效

该方案在中通进行实地部署，双方也一起对 AI 视觉的处理效能进行了验证与测评。如图 2-4 所示，在目标检测模型 YOLO v4 中，各个使用场景组合都有着优异的表现，符合中通的预期。

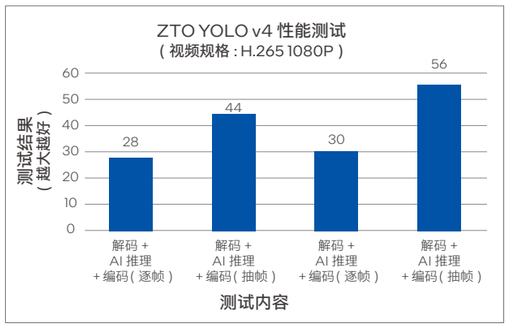


图 2-4 基于英特尔® 数据中心 GPU Flex 170 产品的 YOLO v4 性能测试结果 [6]

新方案一方面帮助中通有效提升了其物流自动化系统的效能，降低了作业高峰期的拥堵和错漏风险。另一方面，也帮助中通实现了降本增效的目标。据中通估算，新方案可帮他们实现约 34.8% 的成本节约 [7]。

[6] 性能测试结果基于中通快递于 2022 年 10 月进行的测试。配置详情：单节点，双路英特尔® 至强® 金牌 6348 处理器（28 核 / 路，56 线程 / 路），启用超线程，启用睿频；GPU：英特尔® 数据中心 GPU Flex 系列 170；内存总容量：256 GB（16 x 16 GB，DDR 2933）；操作系统：Ubuntu 20.04；内核版本：5.10.54；工作负载：dlstreamer；编译器：gcc；库：英特尔® oneAPI 工具套件；其他软件：英特尔® OpenVINO™ 工具套件 2022.2 版。测试中的 AI 推理业务流包括 YOLO v4 目标检测、追踪和分类。

[7] 数据援引自中通快递内部估算结果

融合 AI，OCR 技术为物流全流程提效

物流行业中的 OCR 技术

OCR 技术正在物流行业全流程发挥作用

在物流行业的工作链条中，物流信息的录入、传递和转换等操作无疑会起到至关重要的作用。这些通过手写、打印，或由预置标签粘贴的文字信息，包括收件人、地址、电话等，不仅数量庞大、格式繁琐，且必须在规定时间内准确无误地被处理后，才能确保物流快递整体效率的达标。

近年来，随着电商等新兴行业的快速崛起和发展，物流快递业务总量也获得迅猛增长。数据统计[8]，2022 年上半年全国快递服务企业的业务量已累计完成 512.2 亿票，同比增长 3.7%。这一形势下，依靠人工处理文字信息的传统方式不仅效率低，且存在失误、错漏的风险，由此造成的错单、落单以及送达延误等问题，在很大程度上降低了用户的体验。对于时效为王、竞争激烈的物流快递行业来说，这无疑是亟待得到提升的短板。

基于对文字信息的快速分析识别能力， OCR 技术正在物流行业获得越来越多的运用。与手写笔画、逐一核对的人工方式不同，借助先进光电设备（扫描仪、数码相机等）的使用，OCR 技术可以轻松完成各类文字信息的识别、转换和录入，在更多环节上实现自动化处理，使效率和准确性获得质的提升。

事实上，在物流行业的全流程中，OCR 技术都已经起到了大幅提升工作效率的重要作用：

- **发送端**：寄件人填报地址、身份信息，电商仓库核对出库货品信息等场景中，可借助 OCR 技术实现一键录入、一键认证等；
- **物流链**：下单、分拣、转运以及配送等过程中的信息传递、核对，可以由标签读取等 OCR 技术协助实现自动化操作，加速货品传递流程；
- **配送端**：快递员可通过具有 OCR 功能的手持设备读取快递单，快速获取配送信息，完成签收等流程；
- **监管层**：借助 OCR 技术，监管层可以对物流商品、信息进行全面监控、核查。在规避运输违禁、危险品等风险的同时，也能进一步统筹物流资源，提升效率。

来自市场调研的数据显示[9]，2022 年全球 OCR 市场规模已达 106.2 亿美元，且未来（2023 年至 2030 年）将以 14.8% 的年复合增长率（Compound Annual Growth Rate，CAGR）继续高速发展，而交通物流行业的应用在其中占有显著的比重。

OCR 与 AI 技术的融合

传统的 OCR 技术一般是通过模版匹配的方式来进行文字识别和提取，如图 3-1 所示，其一般可分为图像采集、预处理、版面处理、图像分割、字符识别以及后处理等主要阶段。这些流程中，字符的识别往往容易受到各种因素的影响并带来识别效果不佳的问题。包括：

- **场景适应性差**：在不同物流场景下容易出现切分错误、识别失误等问题；
- **抗干扰性不足**：容易将干扰背景的信息（如边框、花纹等）误判为文本信息；
- **识别泛化不佳**：无法较好地解决图像光照不均、扭曲等问题。

图 3-1 传统 OCR 处理流程

各类 AI 方法，包括机器学习、深度学习，以及近年来大模型的出现，正为以上问题带来解决之道。基于 AI 方法的 OCR 处理流程如图 3-2 所示，其不再需要繁琐的版面处理、后处理以及图像分割等步骤。通过不同的 AI 模型完成字符检测和字符识别过程，即可快速将标签、快递单上的文字信息提取出来。与传统 OCR 技术相比，AI 方法在应对各类识别难点，例如手写潦草，光影遮蔽等问题上有着远胜于模版匹配方式的效率和准确性。

图 3-2 基于 AI 方法的 OCR 处理流程

OCR 任务中常见的 AI 模型

■ DBNet 模型

如上文所述，在基于 AI 方法的 OCR 处理流程中，首先需要定位所采集的标签、快递单等图像中的文字位置（即字符检测），然后才能进行文字的识别（即字符识别）。字符检测解决的问题是文字在哪里，范围有多大。而文字识别是对定位好的文字区域进行识别，进而转化为文字信息。

[8] 相关数据援引自英特尔官网信息：https://www.intel.cn/content/www/cn/zh/customer-spotlight/cases/yunda-smart-logistics-strategy-visual-ai-solution.html
[9] 数据援引自 GRAND VIEW RESEARCH 发布的《Optical Character Recognition Market Size, Share & Trends Analysis Report By Type (Software, Services), By Vertical (BFSI, Retail, Transport And Logistics), By End-use (B2B, B2C), By Region, And Segment Forecasts, 2023 - 2030》：https://www.grandviewresearch.com/industry-analysis/optical-character-recognition-market

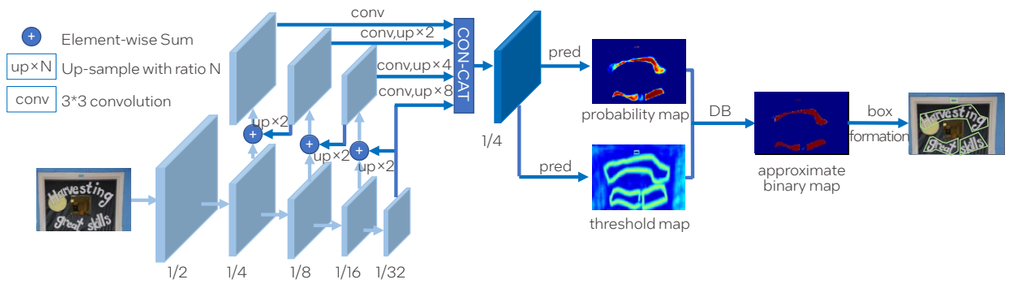


图 3-3 DBNet 网络模型架构

其中常见的字符检测 AI 方法包括支持向量机（Support Vector Machine，SVM）、可微分二值化网络（Differentiable Binarization Network，DBNet）、连接文本预选网络（Connectionist Text Proposal Network，CTPN）、YOLO 等不同的模型。

DBNet 采用了基于分割的字符检测方法，即对每一个字符块都进行语义分割，然后对分割概率图进行简单二值化并转化为检测结果。其引入了可微分二值化（Differentiable Binarization）模块来优化分割预测结果，这一模块能增加错误预测梯度，并可将二值化操作插入到分割网络中进行联合优化，从而获得更好的语义概率图。

DBNet 网络模型架构如图 3-3 所示，其主干网络采用了残差网络（Residual Network，ResNet）。输入的图像首先经过一个特征金字塔（Feature Pyramid Networks，FPN）网络结构，获得分别为 1/4、1/8、1/16 和 1/32 大小的四个特征图。将这四个特征图分别做上采样（1/4 大小）后执行 CONCAT 操作得到特征图（F），由该特征图再分别得到 probability map（概率图 P）和 threshold map（阈值图 T）。通过可微分二值化操作 P 和 T，计算得到近似二值化特征图（approximate binary map）并获得最终检测结果。

更多 DBNet 模型介绍，请参考：
Real-time Scene Text Detection with Differentiable Binarization: https://arxiv.org/abs/1911.08947。

■ CTPN 模型

CTPN 也是目前 OCR 领域常用的字符检测模型，其结构与 Faster R-CNN 基本类似，但加入了 LSTM 层，能有效地检测出复杂场景中分布的文字，对长文本有着较好的检测能力。如图 3-4 所示，CTPN 首先通过 VGG 模型提取特征，获得大小为 N*C*H*W 的 conv5 特征图，然后在其中设置一个 3*3 的滑动窗口（绿框所示），卷积得到 N*9C*H*W 的特征图。经过 Reshape 后，设定 Batch=NH 且最大时间长度为 W 输入双向 LSTM（BLSTM）模型，学习其中每一行的序列特征。

双向 LSTM 模型的结果再进行 Reshape 后，可得到一个 N*256*H*W 的特征图，其既包含了空间特征，也包含了在长短期记忆网络（Long Short-Term Memory，LSTM）模型学习到的序列特征。在经过全卷积（Fully Convolutional，FC）层后，最后加上类似 Faster R-CNN 的 RPN 网络来获得文字区域。

由于 CTPN 仅能检测水平方向的文本，为此在一些场景中，可以使用在 CTPN 的基础上引入了 Segment（即带方向的预选框）的 SegLink 模型，实现多方向的文本检测，这能够让检测的适用场景更多样化。

更多 CTPN 模型介绍，请参考：
Detecting Text in Natural Image with Connectionist Text Proposal Network：https://arxiv.org/abs/1609.03605。

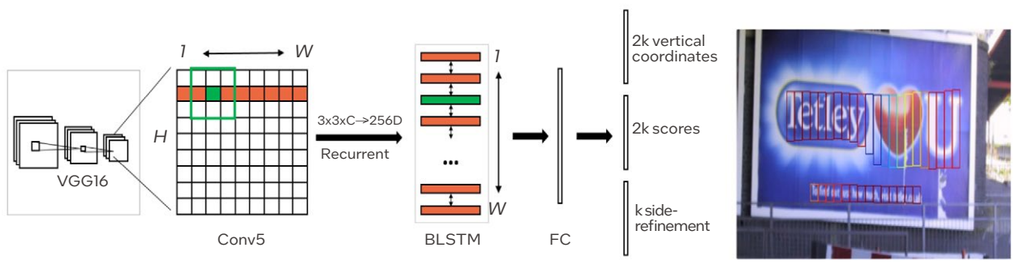


图 3-4 CTPN 网络模型结构

■ CRNN + CTC 模型

传统的 OCR 技术中，字符识别一般有基于连通域、基于边缘特征、基于笔画特征以及基于纹理等方法。在 AI 方法中，目前有基于 CTC 和基于 Attention 两大类字符识别方法。同时，近年来 transformer 的方法也获得了较多的运用。以下就以卷积循环神经网络（Convolutional Recurrent Neural Network，CRNN）+ CTC（Connectionist Temporal Classification）模型为例，简述基于深度学习的字符识别方法。

CRNN 模型主要用于对不定长的文本序列进行端到端识别，其是将文本识别转化为一个时序依赖的序列学习问题。CRNN 的网络模型结构如图 3-5 所示，从下往上的结构依次为：卷积层、循环层和翻译层。

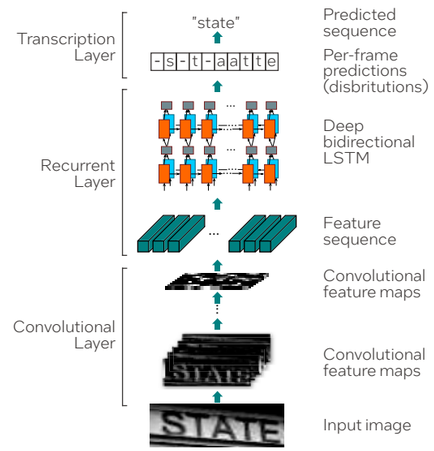


图 3-5 CRNN 网络模型结构

其中，卷积层是使用卷积神经网络（Convolutional Neural Network，CNN）模型对图像的局部特征进行提取。CNN 层得到的特征图（512，1，40），还需要经过 Map-to-Sequence 环节变成一个 40*512 的矩阵输入循环层。循环层使用双向 LSTM 进行时序特征的提取，这里每个 LSTM 的时间步可设为 40，每个时间点的特征维度为 512。这一步的目标是预测每个时间步的字符内容，即输出所有字符的 softmax 概率分布，然后将这个后验概率矩阵传入 CTC 层。

对于 CTC 模型而言，其核心是对一个给定的输入序列 X，CTC 会给出所有可能的输出分布 Y。根据这个分布，系统可以输出最可能的结果或者给出某个输出的概率。CTC 中的预测函数可以表示为 Y=argmax（P（Y|X））。CTC 的工作流程主要分为几个步骤，包括序列合并（对齐），训练以及预测（Greedy Search 和 Beam Search 方案）。

由于 CRNN + CTC 模型是采用了所有字符的 softmax，每个字符的识别是单独的，因此在字符潦草的情况下效果不佳。另一种常见的字符识别方案，是用 Attention 替代 CTC 层，这一方案的优势是解码时利用了上下文信息，精度较高，但不适合用于过长或过短的识别场景中。

更多 CRNN+CTC 模型介绍，请参考：
An End-to-End Trainable Neural Network for Image-based Sequence Recognition and Its Application to Scene Text Recognition: https://arxiv.org/abs/1507.05717。

英特尔产品与技术为基于 AI 的 OCR 技术提效

从前文基于 AI 的 OCR 工作流程可以看出，提升 OCR 任务的关键一方面在于保证处理流水线对图像有着高效的并行多线程处理、预处理能力，同时也有足够的算力应对 AI 推理等工作负载所需的算力。另一方面是对 AI 方案具有加速能力。英特尔® 数据中心 GPU Flex 系列和英特尔® 分发版 OpenVINO™ 工具套件，能在基于英特尔® 架构的处理器平台（例如第三 / 第四代英特尔® 至强® 可扩展处理器、英特尔® 至强® CPU Max 系列等）或其它兼容平台上，为用户的 OCR 任务提供上述技术提效。

英特尔® 数据中心 GPU Flex 系列

英特尔® 数据中心 GPU Flex 系列支持开放、灵活、基于标准的软件堆栈和 oneAPI，因此，开发人员可以构建跨架构的高性能应用和解决方案。这有助于组织降低全新解决方案上市的复杂性、成本和时间要求，以便工程师和程序员进行创新，而不是将时间花在维护代码上。其中，英特尔® 数据中心 GPU Flex 系列 170 图形处理器拥有多达 32 个英特尔® Xᵉ 核心及光线追踪单元、2 个英特尔® Xᵉ 媒体引擎，具备用于 AI 加速的英特尔® Xᵉ Matrix Extensions（英特尔® XMX），并支持基于硬件的 SR-IOV 虚拟化。

更多英特尔® 数据中心 GPU Flex 系列信息，请参考英特尔官网：
https://www.intel.cn/content/www/cn/zh/products/docs/discrete-gpus/data-center-gpu/flex-series/overview.html

英特尔® 分发版 OpenVINO™ 工具套件

另一方面，英特尔® 分发版 OpenVINO™ 工具套件也为物流行业中 OCR 任务的加速提供了强劲的动力。英特尔® 分发版 OpenVINO™ 工具套件包含模型优化器和推理引擎两大组件。如图 3-6 所示，OpenVINO™ 模型优化器（Model Optimizer，MO），用户可将 Caffe、TensorFlow、PyTorch 以及 PaddlePaddle 等多种常见 AI 框架的模型转换为 OpenVINO™ 中间数据格式（Intermediate Representation, IR）的离线模型，并对这些模型的性能进行优化；推理引擎则可以为跨英特尔多种芯片（包括 CPU、GPU 以及 FPGA 等）的计算机视觉异构计算提供加速支持。通过 OpenVINO™ 工具套件，用户可以获得以下优势：

- 提升计算机视觉、自动语音识别、自然语言处理和其他常见任务中的深度学习性能；
- 提升 TensorFlow、PyTorch 等流行框架训练模型的推理效能；
- 减少资源需求，并在从边缘到云的一系列基于英特尔® 架构的平台上高效部署。

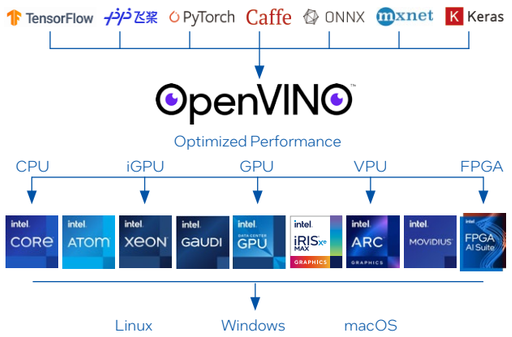


图 3-6 基于异构计算平台、面向不同 AI 框架提供加速的 OpenVINO™ 工具套件

■ 英特尔® 分发版 OpenVINO™ 工具套件的安装与运行

英特尔® 分发版 OpenVINO™ 工具套件一般分为两个部分：

- **OpenVINO™ Runtime**：内含用于在算力设备上运行模型推理的核心库文件集合；
- **OpenVINO™ Development Tools**：这是一组用于处理 OpenVINO™ 和 OpenVINO™ 模型的实用工具，包括：
  - OpenVINO™ Runtime
  - Model Optimizer
  - Post-Training Optimization Tool
  - Benchmark Tool
  - Accuracy Checker and Annotation Converter
  - Model Downloader and other Open Model Zoo tools

安装英特尔® 分发版 OpenVINO™ 工具套件，用户可以直接安装 OpenVINO™ Development Tools（内含 OpenVINO™ Runtime），也可以单独安装 OpenVINO™ Runtime，或者由源代码编译安转。以下是以 Linux（macOS）+ Python 环境为例，安装 OpenVINO™ Development Tools 过程的简要描述。其它安装方式，请参阅：https://docs.openvino.ai/2022.3/openvino_docs_install_guides_overview.html

在安装过程之前，用户需要检查 Python 的版本，使用 Python 3.7 或更高版本，值得注意的是，从版本 2022.1 开始，OpenVINO™ Development Tools 仅能通过 PyPI 安装。

 # 步骤 1

安装并激活 Python 虚拟环境，并将 pip 升级到最新版本。

```
python3 -m venv openvino_env
source openvino_env/bin/activate
python -m pip install --upgrade pip
```

 # 步骤 2

安装软件包。

```
pip install openvino-dev
```

如果用户在 OpenVINO™ 工具套件和深度学习框架之间遇到兼容性问题，可以通过 extra 参数将 OpenVINO™ 开发工具安装到单独的环境中。例如：

```
pip install openvino-dev[extras]
pip install openvino-dev[tensorflow2,onnx]
```

 # 步骤 3

验证安装过程。

```
mo -h
```

用户可以利用英特尔® 分发版 OpenVINO™ 工具套件进行一次单目深度估计（Monodepth Estimation）的工作任务，来验证其运行状况。

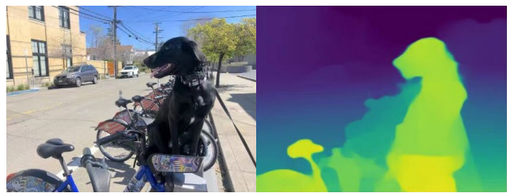


图 3-7 利用英特尔® 分发版 OpenVINO™ 工具套件进行一次单目深度估计任务

单目深度估计是一种使用单个图像来估计场景深度的任务，其在机器人、三维重建、医学成像和自主系统中有着良好的应用前景。下文的验证过程使用 MiDaS 神经网络模型。

MiDaS 神经网络模型由 Embedded AI 基金会开发，更多信息请参阅。R.Ranftl、K.Lasinger、D.Havner、K.Schindler 和 V.Koltun，“Towards Robust Monocular Depth Estimation: Mixing Datasets for Zero-shot Cross-dataset Transfer”，https://ieeexplore.ieee.org/document/9178977。

鉴于篇幅有限，以下仅列出面向图像处理的关键工作步骤供参考：

 # 步骤 1

导入、resize、reshape 图像。

```
IMAGE_FILE = "../data/image/coco_bike.jpg"
image = load_image(path=IMAGE_FILE)

# Resize to input shape for network.
resized_image = cv2.resize(src=image, dsize=(network_image_height, network_image_width))

# Reshape the image to network input shape NCHW.
input_image = np.expand_dims(np.transpose(resized_image, (2, 0, 1)), 0)
```

 # 步骤 2

对图像进行推理。

```
result = compiled_model([input_image])[output_key]

# Convert the network result of disparity map to an image that shows
# distance as colors.
result_image = convert_result_to_image(result=result)

# Resize back to original image shape. The `cv2.resize` function expects shape
# in (width, height), [::-1] reverses the (height, width) shape to match this.
result_image = cv2.resize(result_image, image.shape[:2][::-1])
```

 # 步骤 3

展示单目深度估计图像。

```
fig, ax = plt.subplots(1, 2, figsize=(20, 15))
ax[0].imshow(to_rgb(image))
ax[1].imshow(result_image);
```

更完整的任务执行过程以及面向视频的任务执行，可参阅：https://docs.openvino.ai/2022.3/notebooks/201-vision-monodepth-with-output.html

■ 在 OCR 工作任务中使用英特尔® 分发版 OpenVINO™ 工具套件

在完成英特尔® 分发版 OpenVINO™ 工具套件的编译安装后，用户就可以借助其开展对 AI 应用的加速。如图 3-8 所示，在一个典型加速场景中，用户可以使用 TensorFlow 和 PyTorch 等流行 AI 框架训练 OCR 任务中的字符检测或字符识别模型，或者从 OpenVINO™ 工具套件内置的 Model Zoo 中选择一个预先训练的模型。

图 3-8 使用 OpenVINO™ 工具套件加速 AI 应用

运行模型优化器以执行静态模型分析，并生成可使用 OpenVINO™ 工具套件进行推理的模型 IR 文件。使用 OpenVINO Runtime API 读取模型 IR 文件，并在选定的计算平台上执行推理过程。这一过程中，工具套件能通过量化、修剪、预处理优化等特殊优化方法，对整个 pipeline 进行调整和优化，以提高性能。在完成所有工作后，用户可使用 OpenVINO™ 工具套件对 AI 应用进行部署。

值得一提的是，OpenVINO™ 工具套件提供了简单清晰的通用 API 接口来实现跨环境的异构执行，即方案中的 OCR 模型可以在 CPU、GPU、FPGA 等不同异构计算平台上使用。以 CPU 为例，可以通过如下代码进行网络初始化：

```
Core ie;
CNNNetwork net = ie.ReadNetwork("model.xml");
ExecutableNetwork executable_network = ie.LoadNetwork(network=net, device_name="CPU");
```

其中 device 可设置成 CPU、GPU 以及其它设备，config 为可选项，可以添加附加设置，例如绑定、优先级设置等。

■ 通过模型量化提升性能

来自第三 / 第四代英特尔® 至强® 可扩展处理器、英特尔® 至强® CPU Max 系列等处理器的英特尔® 深度学习加速（英特尔® DL Boost）、英特尔® 高级矩阵扩展（英特尔® AMX）等技术，对 INT8、AVX - 512_BF16 或 AMX_BF16 等低精度数据格式提供了良好的支持，使用低精度数据格式可为 OCR 任务带来以下性能优势（以 BF16 为例）：

- BF16 数据的尾数较短，因此两个 BF16 数据相乘更快，并可实现与 FP32 相互快速转换；
- BF16 数据位宽是 FP32 的二分之一，对内存资源消耗更低，让用户在方案中可选择更大规模的模型；
- 方案中数据量传输降低，减少数据转换时间；
- 英特尔® 分发版 OpenVINO™ 工具套件对低精度数据格式有着良好的支持，并可通过量化功能将 FP32 数据格式的 OCR 模型转换为所需的低精度数据格式。

查询 CPU 等计算平台是否支持某种数据格式（如 AVX - 512_BF16）可采用以下方式：

- 请使用 openvino.runtime.Core.get_property 查询 ov::device::capabilities 属性；
- 执行 lscpu | grep avx512_bf16 或者 cat /proc/cpuinfo | grep avx512_bf16；

OpenVINO™ 工具套件默认支持 BF16，这种情况 KEY_ENFORCE_BF16 默认设置为 YES，以下代码示例演示了如何检查是否设置支持 BF16:

```
InferenceEngine::Core core;
auto network = core.ReadNetwork("sample.xml");
auto exeNetwork = core.LoadNetwork(network, "CPU");
auto enforceBF16 = exeNetwork.GetConfig(PluginConfigParams::KEY_ENFORCE_BF16).as<std::string>();
```

如果要禁用 BF16 的内部转换，将 KEY_ENFORCE_BF16 设置为 NO 即可：

```
InferenceEngine::Core core;
core.SetConfig({{CONFIG_KEY(ENFORCE_BF16), CONFIG_VALUE(NO)}}, "CPU");
```

■ 英特尔® 分发版 OpenVINO™ 工具套件性能验证

用户可以通过运行基准测试来验证 OCR 模型在使用英特尔® 分发版 OpenVINO ™ 工具套件后的性能表现。

 # 步骤 1

初始化测试环境。

```
$ cd ~
$ source ~/ov_py38/bin/activate
$ source ~/openvino/install_gpu_onednn/setupvars.sh
```

 # 步骤 2

检查硬件状态。

```
$ sudo vainfo
$ sudo clinfo
```

结果示例如下（红色箭头处为重要检查点）：

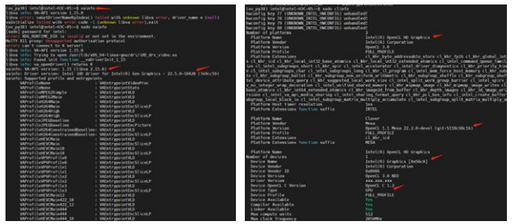


图 3-9 OpenVINO ™ 工具套件硬件状态检查结果

 # 步骤 3

基于英特尔® 数据中心 GPU Flex 170 平台评估 OCR 模型性能。在 DBNet、ShuffleNet 以及 Text Recognition 三个不同的 OCR 模型上分别使用 OpenVINO™ 工具套件执行模型转换和性能测试。

```
#Dbnet
#Model convert
mo --framework paddle --input_model dbnet/inference.pdmodel --input_shape "[1,3,640,640]" -o dbnet
#Performance evaluation
$ /home/intel/openvino/install_gpu_onednn/samples/cpp/intel64/Release/benchmark_app -m ~/ov_models/ocr_pipeline/dbnet/inference.xml -d GPU -hint none -t 5 -shape [1,3,640,640]

#Shufflenet
#Model convert
mo --framework onnx --input_model shufflenet/shufflenetv2_05.onnx --input_shape "[1,3,400,400]" -o shufflenet
#Performance evaluation
$ /home/intel/openvino/install_gpu_onednn/samples/cpp/intel64/Release/benchmark_app -m ~/ov_models/ocr_pipeline/shufflenet/shufflenetv2_05.xml -d GPU -hint none -t 5 -shape [1,3,400,400]

#TestRecognition
#Model convert
mo --framework paddle --input_model recognition_model/inference.pdmodel --input_shape "[1,3,48,320]" -o recognition_model
#Performance evaluation
$ /home/intel/openvino/install_gpu_onednn/samples/cpp/intel64/Release/benchmark_app -m ~/ov_models/ocr_pipeline/recognition_model/inference.xml -d GPU -hint none -t 5 -shape [1,3,48,320]
```

验证测试结果如下，可以看到，三种模型下的处理延迟都控制在了 20 毫秒以内 [10]。

Model_Name	input_shape	Precision	Device	batchsize	hint	Throughput	Latency
dbnet	1,3,640,640	FP32	GPU	1	none	248.01 FPS	16.10 ms
shufflenet	1,3,400,400	FP32	GPU	1	none	1588.36 FPS	2.51 ms
Text Recognition	1,3,48,320	FP32	GPU	1	none	810.43 FPS	4.92 ms

表 1 基于英特尔® 数据中心 GPU Flex 170 平台评估 OCR 模型性能验证结果

[10] 性能测试结果基于韵达于 2022 年 10 月进行的测试。配置详情：单节点，双路英特尔® 至强® 金牌 6348 处理器（28 核 / 路，56 线程 / 路），启用超线程，启用睿频；GPU：英特尔® 数据中心 GPU Flex 系列 170；内存总容量：256 GB (16 x 16 GB, DDR 2933)；操作系统：Ubuntu 20.04；内核版本：5.10.54；工作负载：dlstreamer；编译器：gcc；库：英特尔® oneAPI 工具套件；其他软件：英特尔® OpenVION™ 工具套件 2022.2 版

实战案例

韵达基于英特尔产品，优化网点三段码 OCR 项目

■ 项目背景

作为物流快递领域的巨头，韵达 [11] 目前拥有全国 5,000 余家一级网点，2,000 余家二级网点。所处的行业龙头地位，也使韵达面临着巨大效率压力。以上海分拨中心为例，其日常派单量超 200 万单，在遇到电商节等营销高峰期时，所面临的处理量更是一个天量数字。这一情况下，如不能有效提升各个环节的效率，就会面临网点挤压，甚至爆仓的风险。

为此，韵达正与合作伙伴一起，借助 IT 领域的前沿技术来应对这一风险。如何帮助网点用更少人力去派件和揽收，增加网点派单量，韵达给出的方案是搭建基于网格仓的网点三段码 OCR 自动化分拣线。网格仓是韵达介于分拨中心和网点之间的一个物流环节，所有进入网格仓的快递件将具备三段码（包含所有快递信息，根据面单上的三段码决定包裹的流向）。

为了进一步提升网点三段码 OCR 方案的效率，韵达与英特尔开展合作，通过对方案的进一步优化，以及英特尔® 数据中心 GPU Flex 系列和英特尔® 分发版 OpenVINO™ 工具套件的引入来提升效能。韵达希望通过这一项目的实施，有效实现快递件的自动分拣，提高网格仓吞吐量。

■ 解决方案

在新的网点三段码 OCR 自动化分拣线工作流程中，分拨中心将快递件送到网格仓，工作人员搬件到工件台，运单向上，由相机拍摄扫描运单，三段码包含所有快递信息，比如地址识别、地址关联分拨中心、网点、快递员等。上述文字信息在被 OCR 识别读取后，会进一步转换为分类信息，并由系统统筹下一步作业流程。

首先在图像采集上，传统上，使用单一相机顶部扫描的 OCR 工作流程中，要求工作人员必须将面单朝上放置。而新方案将通过 5 个相机从不同方位扫描，实现五面扫，减少工作人员放件时间。

在三段码 OCR 识别流程上，新方案包括了面单区域识别、角度检测、截取面单区域，并做仿射变化（提取面单区域，并转换位平行于屏幕）、三段码文字区域识别等流程。如图 3-10 所示，方案中所用的 OCR 模型包括 Paddle Paddle OCR 模型（DBNet）以及韵达自研的模型（类 ShuffleNet）。

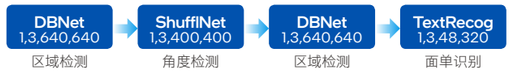


图 3-10 韵达三段码 OCR 识别流程

■ 方案成效 [12]

在识别精度上，新方案可达到 97-98% 的精度，大于行业所需的 95% 精度。符合韵达的预期（如识别错误率过大，无法识别三段码会导致分拣线回流，需要重新实施人工分拣，进而加重网点负担）。

而在性能上，整个方案包括视频拉流、解码、抓图以及三段码 OCR 识别等，在三段码 OCR 测试中，共对 2,450 张图片进行了识别，测试结果显示平均运行时间为 114 ms，优于韵达 130 ms 的期望标准，可很好地满足韵达对低时延的需求。目前新方案已部署覆盖 70 余个网格仓，未来会继续扩大规模，推广到更大的网点。

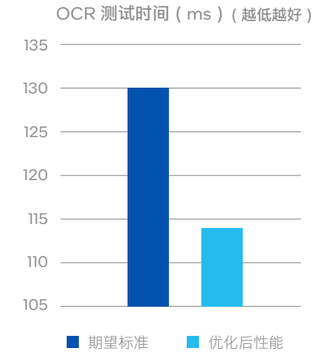
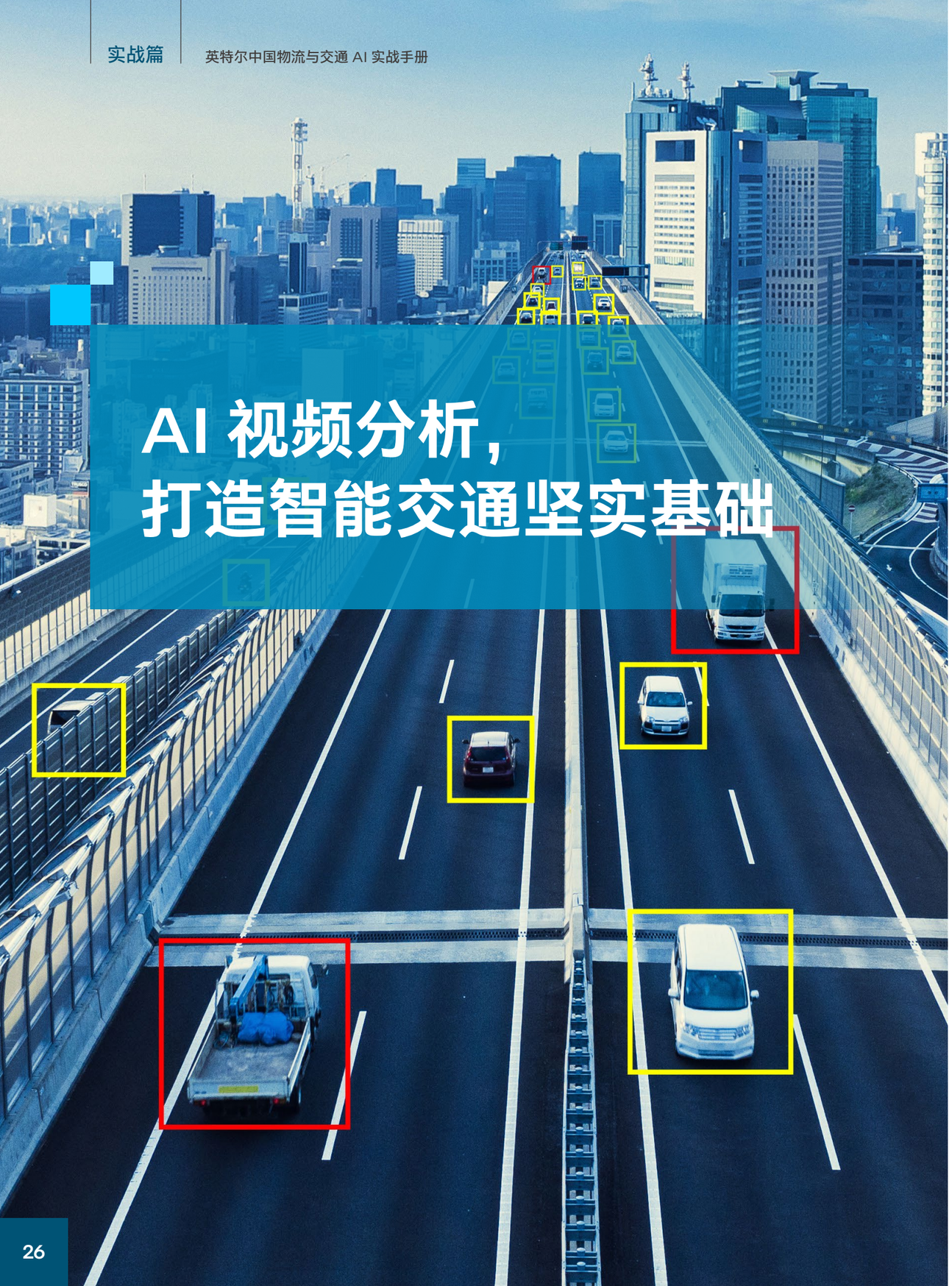


图 3-11 韵达三段码 OCR 优化效果

[11] 数据源自：http://www.yundaex.com/cn/

[12] 性能测试结果基于韵达于 2022 年 10 月进行的测试。配置详情：单节点，双路英特尔® 至强® 金牌 6348 处理器（28 核 / 路，56 线程 / 路），启用超线程，启用睿频；GPU：英特尔® 数据中心 GPU Flex 系列 170；内存总容量：256 GB (16 x 16 GB, DDR 2933)；操作系统：Ubuntu 20.04；内核版本：5.10.54；工作负载：dlstreamer；编译器：gcc；库：英特尔® oneAPI 工具套件；其他软件：英特尔® OpenVION™ 工具套件 2022.2 版



AI 视频分析，打造智能交通坚实基础

交通领域中的 AI 视频分析

视频分析正成为智能交通的坚实基座

一直以来，由高清交通摄像头、视频服务器以及后端监控设备等设备所构建的各类监控系统都在交通领域扮演着重要的角色，并逐渐成为交通基础设施向着数字化转型的重要抓手。在城市街道、交通枢纽、桥梁卡口以及高速公路等不同交通场景中，以交通摄像头为核心的监控系统正为交通系统的高效运行提供全方位助力，包括：

- **路况 / 流量监控**：通过将监控区域的实时路况图像传回各级交管中心，让交管部门可实时了解道路和交通流量实况，并根据流量情况对拥堵等交通状况及时作出预判和反馈；
- **车辆 / 车牌识别**：通过对道路、停车场以及卡口区域内的车辆情况及车辆牌照进行识别，既可以加速车辆通行、通过效率，也能让交管部门对违规、违法行为做实时判别；
- **事故预警**：借助视频监控对人、车、路等交通数据开展全方位收集并进行实时分析，并通过各种预警机制来协助驾驶者做出正确决策，避免追尾、碰撞等事故发生。

而随着交通规模的不断扩大，传统人工监控方式已逐渐不能满足交管部门在实时性、主动性和联动性等方面的需求。一方面，人眼在处于疲劳状况下容易错漏关键信息，无法达到百分百覆盖的高标准；另一方面，人工监控也无法有效将多个信息进行关联，实现预防和预警的效果。这些都对交通监控系统效率的进一步提升提出了挑战。

智能化的视频分析技术正成为这一痛点的解决之道。通过对所采集视频流中的图像进行多元化的处理和数据提取，识别出不同的交通场景、车辆信息以及人物行为特征等，就可以实现对车辆、道路、行人和驾驶者之间数据的有效汇集、分析、预测、运筹和交互，从而对交通运行进行有效监控和管理。事实上，通过与先进的信息与计算机、通信、自动控制以及物联网等技术的融合，由视频分析技术作为基座之一的智能交通（Intelligent Traffic）正在帮助人们解决当前交通领域所面临的诸多问题，为通畅出行和可持续经济发展提供更加高效和安全的交通运行体系。

由高清交通摄像头与智能视频分析技术作为基础的智能交通摄像头在近年来获得了巨大的市场规模和发展前景。来自市场的研究报告表明，2023 年全球智能交通摄像头市场预期达到 121.5 亿美元。而到 2027 年，这一数据有望突破 200 亿美元，CAGR 超过 10%[13]。

基于 AI 的视频分析

计算机视觉（Computer Vision，CV）是智能化视频分析的技术核心之一，传统的交通视频分析，通常是基于一定规则，借助 CV 技术来对一定的事件、目标和行为进行监测和分析。例如对于行驶的车辆，系统会将其作为一个运动目标，通过与周围事物的参照比较来判定行驶速度、状态等信息。又如在道路交规监测中，系统会根据车辆、行人等对象与行道线、交通标识线的距离和碰撞情况来判断是否违反交通规则。

但这些视频分析方法较为容易受到外界因素影响，如在夜晚或雨雪大雾天气，或是路面标识出现破损等情况下，视频图像中的对象跟踪和识别就变得更为困难。在准确率下降的同时也会造成识别分析速度的下降。同时，对于交通场景中大量的快速运动目标，传统视频分析方法在处理时可能有一定的目标偏差。

得益于各类 AI 技术的加持，近年来基于 AI 的视频分析技术正在交通领域获得高速发展并在多个应用场景获得落地，使交通领域的视频分析在分析深度与广度，预测精度和准度等方面获得飞跃，从而为交通参与者与管理者提供了大量实时监测、目标跟踪、目标识别的解决方案。

例如基于深度学习的 OCR 技术，能够快速识别传统视频分析难以甄别的污损号牌，提升停车场等交通枢纽的通过效率。而基于 AI 的视觉分析技术，也能更好地感知车辆位置，有助于车路协同、路网优化工作的进一步推进。在提升交通运行效率之余，交管部门对交通突发事件反应的灵敏度也有所提升。

与其它 AI 应用相比，在交通领域构建基于 AI 的视频分析，一方面需要选择前沿的 AI 网络模型（目标检测、目标识别等），并为之配备高效的推理算力，以便应对交通场景中瞬息万变的需求。另一方面，图像的编解码效率也是影响分析速度的重要因素。因此，如何帮助用户深入挖掘基础设施平台潜能，为智能交通提供更强 AI 视频分析能力输出，正成为 AI 在交通领域落地的焦点。

[13] 数据援引自 The Business Research Company《Smart Traffic Camera Global Market Report 2023》：https://www.thebusinessresearchcompany.com/report/smart-traffic-camera-global-market-report

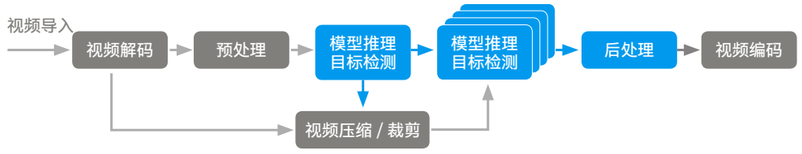


图 4-1 典型的 AI 视频分析工作流程

AI 视频分析中常见的 AI 模型与编解码

如图 4-1所示，一种典型的 AI 视频分析工作流程包括以下主要步骤。压缩传输的视频首先需要解码以便于后续处理，常见的编解码技术标准包括高级视频编码（Advanced Video Coding, AVC（也称为 H.264））、高效率视频编码（High Efficiency Video Coding，HEVC（也称为 H.265））。

在经过亮度、对比度和饱和度调整等预处理步骤后，系统会使用所选择的目标检测模型进行推理，得到图像中人、车、道路等目标的位置框图。目标检测的结果将与压缩（Scale）/ 裁剪（Crop）后的视频融合，对多个检测目标进行推理识别，完成后再进行追踪等后处理步骤，最后将处理完成的视频重新进行编码处理。

■ SSD 模型

SSD 模型、YOLO 系列模型都是目前常见的深度学习目标检测模型，在视频分析场景中有些广泛的运用（YOLO 模型简介详见 12 页物流部分）。如图 4-2 所示，SSD 算法通常基于 VGG、残差网络（Residual Net，ResNet）等模型展开（图中为 VGG-16）。经过一系列的卷积层和池化层，得到一定尺寸的特征图，再在这个特征图上进行回归，得到物体的位置和类别。

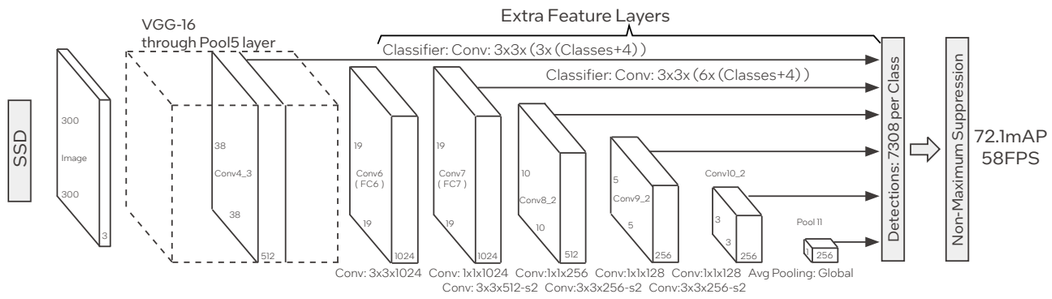


图 4-2 SSD 网络模型架构

与其他算法不同的是，SSD 模型架构中引入了多个候选框和特征图。在深度神经网络中，浅层的特征图往往包含更多细节信息，适合较小目标的检测；较深的特征图包含更多全局信息，适合大目标的检测。因此，通过在不同特征图上对不同尺寸的候选框进行回归，SSD 算法在对不同尺寸的物体进行检测时结果更加精准。

更多 SSD 网络详情，请参阅
SSD: Single Shot MultiBox Detector，Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, Alexander C. Berg：https://arxiv.org/pdf/1512.02325.pdf

■ ResNet 模型

一般而言，深度神经网络中网络层数越多，能够提取到的不同层次的特征越丰富。同时，更深的网络能够使得提取到的特征更抽象，更富有语义信息。但深度的增加也会带来退化（Degradation）问题，即准确率会先上升直至饱和，而继续增加深度，却导致准确率逐渐下降。ResNet 通过一种巧妙的结构来解决这一问题。

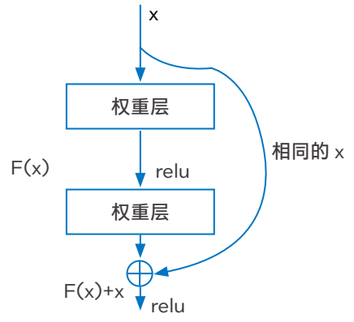


图 4-3 ResNet 残差块构造

如图 4-3 所示，在 ResNet 中可以构成多个残差块结构，其输入与期望输出相等，构成一种恒等映射的关系。整个 ResNet 网络由多个残差块和恒等映射块拼接而成。这样的结构，能让深度神经网络在不断增加深度的同时保持准确率。目前，ResNet50 模型（即共有 50 层）已在图形分类等 AI 场景中被广泛应用。

更多 ResNet 网络详情，请参阅
Deep Residual Learning for Image Recognition，Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun：https://arxiv.org/pdf/1512.03385.pdf

■ 视频编解码

来自高清摄像头、视频存储系统等设备的原始视频通常体积极为庞大，因此需要经过编码后再进行传输。例如以目前交通领域较为普遍的全高清格式（1080P / 24 位色 / 每秒 30 帧）来说，其原始图像码率高达 1.5Gbps，这显然无法实现有效的网络传输。通过 AVC、HEVC 等视频编码技术，码率可以被压缩至数 Mbps，便于传输。但在进行视频分析时，系统仍然需要对其进行解码操作以便进行后续的分析，同时分析结束后还需要重新进行编码。

视频编解码一般可分为软件、硬件两种方式，前者主要由处理器承担主要工作负载，优点是方案简洁、编解码质量高，但缺点是会占用大量的系统资源且编码效率不高。硬件编码是借助 GPU、FPGA 等专用硬件来进行编解码，优点是效率高且不占用系统资源，但也有灵活性不高等问题。目前常见的视频编码技术标准有 AVC、HEVC 等。

在视频编解码之外，视频分析同样也需要其它的一些媒体处理，如压缩、裁剪、亮度 / 饱和度调整等。在交通视频分析场景中，往往需要对多路视频进行上述媒体处理后再开展分析，因此能并行处理的视频流越多，系统整体效率越高。值得一提的是，在视频媒体处理方案中，可以引入一些专门的视频处理库来提升效能，例如英特尔® oneAPI 视频处理库（英特尔® oneVPL）。

英特尔产品与技术为交通 AI 视频分析提效

从交通场景视频分析的流程可以看到，其一般可分为媒体处理和 AI 推理两类主要的工作负载。为应对不同工作负载所需的处理能力，用户通常也会为之配备相适应的硬件基础设施。这不仅提升了建设和运维成本，不同硬件之间的协作流程也会造成效率的下降。

为了帮助交通以及其它领域的用户应对这一问题，英特尔在为 AI 视频分析场景提供了基于英特尔® 架构的第三 / 第四代英特尔® 至强® 可扩展处理器、英特尔® 至强® CPU Max 系列、英特尔® 数据中心 GPU Flex 系列等作为核心算力引擎外，也提供了基于英特尔® 分发版 OpenVINO™ 工具套件的英特尔® 深度学习 Streamer（英特尔® DL Streamer）流水线框架来帮助用户为 AI 视频分析构建端到端的流水线，加速媒体处理和 AI 模型推理进程。

与此同时，英特尔也推出 Deep Link Workload Scheduler 微服务，来帮助用户更合理地在异构计算平台（CPU、GPU、VPU）上部署工作负载并实现处理效率最优化。

英特尔® Deep Link Workload Scheduler

如图 4-4 所示，用户部署在本地数据中心或云端的分析平台很可能属于异构计算平台。在部署工作负载时，如何将 AI 视频分析工作负载中的媒体处理任务（视频编解码、裁剪压缩、预处理等）和 AI 推理任务（目标检测、目标识别、动作识别等）分配到合适的硬件设备达到更优的硬件使用效率，并为不同硬件模式下的工作负载选择最优的参数是用户面临的 2 个重要挑战。

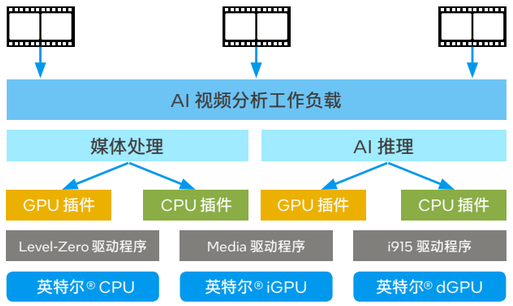


图 4-4 基于异构计算平台的 AI 视频分析

这对于保证及提升 AI 视频分析以及其它类似工作负载的性能尤为重要。例如异构平台可以使平台对支持更多 AI 模型（某些 AI 模型可能仅可在单一算力硬件上运行或得到优化），即使相同的 AI 模型在不同算力硬件下推理性能也会表现不同。因此，平台面临的挑战是如何让 CPU 与 GPU 等不同的算力硬件同时被自动调度并参与到 AI 视频分析工作负载中，这对于深度挖掘平台中每一个算力硬件的潜能，帮助用户提升工作负载的性能以及资源利用率，并有效降低总拥有成本（Total Cost of Ownership，TCO）而言非常重要。

由英特尔在英特尔® Deep Link 技术[14] 的基础上推出的 Deep Link Workload Scheduler 微服务正帮助用户应对这一挑战。如图 4-5 所示，Deep Link Workload Scheduler 微服务是在 AI 视频分析工作负载与硬件之间提供一种快速有效的工作负载分析和调度方法，以自动化的方法来帮助工作负载获得不同设备组合、以及工作负载参数集的更优选。

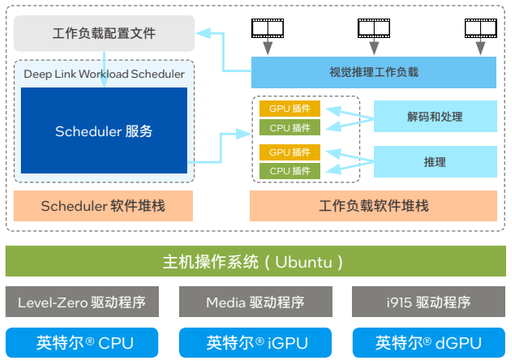


图 4-5 异构计算平台上引入 Deep Link Workload Scheduler 微服务的 AI 视频分析

Deep Link Workload Scheduler微服务支持各类英特尔硬件平台。无论是面向服务器市场的英特尔® 至强® 可扩展处理器，英特尔® 数据中心 GPU Flex 系列，或是面向客户端市场的英特尔® 酷睿™ 处理器系列，英特尔锐炬® X[e] 显卡以及英特尔锐炫™ 显卡，Deep Link Workload Scheduler 微服务都可以在这些硬件平台上进行 AI 视频分析工作负载的分析和调度。

Deep Link Workload Scheduler 微服务支持多种部署方式：

- 直接安装在裸金属服务器上；
- 以容器的形式部署在集群的工作节点上。

用户可根据实际场景选择合适的部署方式。同时，用户也可在不改变工作负载代码的情况下，方便地将 Deep Link Workload Scheduler 微服务与其集成，并且用户也可以自定义工作负载的可调参数和目标 KPI (Key Performance Indicator)。

为了更加直观地向用户展示硬件以及工作负载的状态，Deep Link Workload Scheduler 微服务会将硬件的遥感 (Telemetry) 数据、以及工作负载的配置参数和 KPI 输出到数据库中，并提供了基于 Grafana 的预定义可视化界面，用户也可以根据数据库中的信息自定义界面以满足不同的可视化需求。

在多个面向 AI 视频分析的测试示例中，Deep Link Workload Scheduler 微服务就表现出了显著的性能提升。在一个使用 YOLO v5 模型做 AI 视频分析的测试场景中，测试基于由第四代英特尔® 至强® 可扩展处理器和英特尔® 数据中心 GPU Flex 系列构建的异构计算平台展开。与基准运行方式（简单调参、人工调参）相比，使用 Deep Link Workload Scheduler 微服务的运行项在视频流并行分析处理数量上有着可观的提升。同时在使用优化的参数集之后，各个硬件资源使用率达到更充分的利用。

在另一个面向 AI 视频分析的测试场景中，使用 SSD 模型（SSD-Mobilenet-V1-coco) 进行目标检测、以及使用 ResNet50 (ResNet50-TF）进行目标识别。测试同样基于由第四代英特尔® 至强® 可扩展处理器和英特尔® 数据中心 GPU Flex 系列构建的异构计算平台展开，与基准运行方式（只使用 GPU 运行工作负载）相比，基于异构计算平台且使用 Deep Link Workload Scheduler 微服务调优的方式下的视频流并行分析处理数量可增加数倍，使视频处理效率大幅提升。同样，这一场景中的硬件资源也获得了更有效的挖掘，达到了预期的目标。

[14] 更多英特尔® Deep Link 技术信息，请参阅：https://www.intel.cn/content/www/cn/zh/architecture-and-technology/adaptix/deep-link.html

实战案例

捻果科技利用英特尔产品，打造机场 AI 视频分析系统

■ 项目背景

航空业的持续增长，让机场的运行效能和安全管理受到越来越多的重视。在机场的飞行区等多个区域，不仅车辆种类多、数量多，机场内路线也极为复杂，行车道与飞机滑行道交叉，车机冲突频繁，给机场的安全运行带来很大风险。

传统上，除通过人力、车辆巡察之外，视频监控也是机场最重要的管理手段之一。过去，机场监控往往采用人工方式，但这存在许多监控漏洞与安全隐患。例如人工监控具有滞后性和被动性等特点，不能对目标进行实时分析和主动处理，因此无法对机场内的异常侵入、线路冲突以及不规范行为做出及时有效的预防、预警作用。

为帮助机场构建智能化的视频分析系统，长期致力于各类机场解决方案的捻果科技与英特尔展开合作，利用英特尔® 硬件平台打造全新的机场 AI 视频分析方案。借助 AI 视频分析技术，新方案可用于检测飞行区域内操作人员、作业车辆、无动力设备的操作规范和标准操作条件，实现对车辆路线和驾驶员的实时监控，避免车辆与人员、车辆与飞机的冲突，从而有效降低安全风险，优化现场资源配置，提高应急处置能力。

■ 解决方案

如图 4-6 所示，新方案分为前端采集端、视频分析平台以及可视化管理界面等几个组件。在前端，通过高清摄像头采集的图像可直接编码压缩后传输到视频分析平台，也可通过预处理后将关键帧传输至后端。

在视频分析平台，捻果科技与英特尔协作构建了高效的端到端视频分析流水线。视频图像首先会利用第三 / 第四代英特尔® 至强® 可扩展处理器进行视频解码、关键帧识别、图像预处理等媒体处理任务，然后根据不同场景的分析需求进行 AI 推理，例如航空器识别、车辆识别、人员识别、违章识别以及入侵识别等。

在实战中，借助上述 AI 视频分析能力以及所预设的不同告警规则，新方案一旦检测到目标在场景中出现了违规越规、具有潜在危险的行为，系统就会自动告警。机场管理者可通过告警信息采取措施，实现安全管理目标。

捻果科技借助 OpenVINO™ 工具套件对多种视频分析 AI 模型开展了优化。AI 推理工作可以利用英特尔® 至强® 可扩展处理器或英特尔® 数据中心 GPU Flex 系列运行，方案中也借助 Deep Link Workload Scheduler 微服务调优对异构计算平台的分析处理效能进行了调优。

同时，方案中还为管理者提供了可视化管理界面，帮助其设定告警规则，了解系统运行全貌。同时也将丰富的监控视频数据资源进一步挖掘转化为安全管理价值，为安全监管覆盖面的扩大做出贡献。

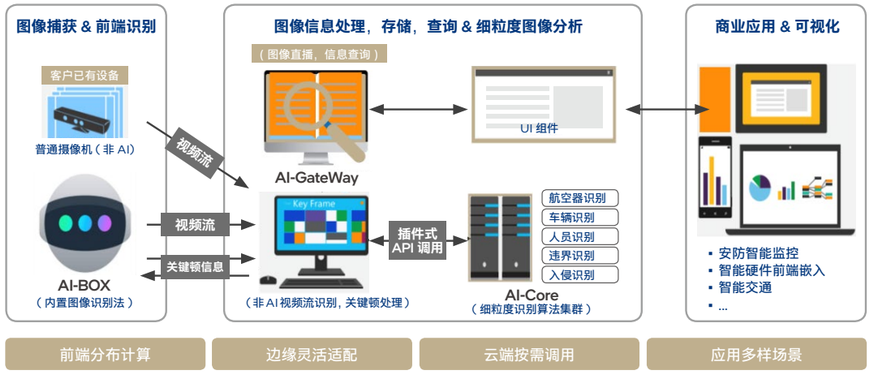


图 4-6 捻果科技新方案架构

■ 方案成效

目前，新方案已在多个机场的试点运行中取得了令人满意的效果。一方面，方案有效降低了人力成本投入，能帮助机场实现 7 X 24 小时不间断智能巡视，有效提升了巡视效率和告警准确度。另一方面，方案也为机场的安全运行、安全防范提供了非现场、智能化的管理手段，提高了精准监管和科学化安全运行管理能力。

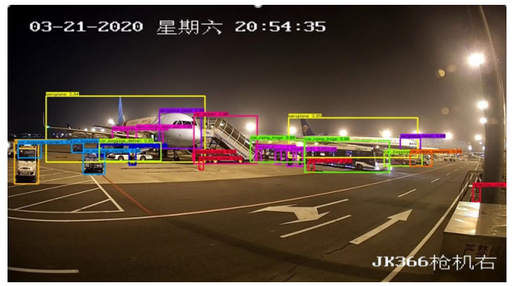


图 4-7　捻果科技新方案成效

除了机场安全管理之外，新方案中已被验证的 AI 视频分析能力，也可被延伸至机场施工作业、航空物流货站保障等场景中应用，从而形成一套完整的民航生产安全监管综合解决方案。

AI 与大数据协同推进智能交通管理

基于 AI 与大数据的智能交通管理

AI 与大数据正助力交通管理实现更高效率

近年来，持续推进的交通建设为国家经济发展以及人们日常出行带来了巨大助力和便捷性。随着交通流量不断提升、交通情况日益复杂，交通管理部门开始借助高速发展的大数据以及 AI 技术来建设智能化的交通管理方案，实现提升通行效率和出行安全性、减少违章和事故发生率的目标。这些方法包括：

- **智能交通监控**：得益于视频监控技术的创新升级和数字图像处理、计算机视觉和 AI 的应用，并结合大数据能力，智能交通监控能有效完成车辆超速、车辆逆行、交通堵塞、道路烟雾和火灾等事件的自动监控，并且就车流量、车速、车型、突发事件紧急程度进行预测分析，为道路安全运行与危险情况营救提供必要的数据支持。
- **智能交通指挥**：通过对路况进行图像处理与视频分析，并综合各方面的大数据信息得出整体性的道路情况播报，自动完成拥堵、畅通等级的划分。一旦发生交通事故，智能交通指挥能马上评测道路交通状况，给出交通指挥建议，生成和发布交通诱导方案和信息，从而更好地服务于交通指挥者和出行者。
- **智能违章管理**：通过检测和跟踪运动车辆，准确地判断车辆的驾驶行为，详细记录车辆的违章行为。例如闯红灯、不按导向车道行驶、骑压车道线实线行驶、车辆逆行行驶等违章行为都能够予以判别和记录，并能够对监控路面过往的每一辆机动车的车辆和车号牌图像进行连续全天候实时记录。

值得一提的是，近年来智能交通预测能力正得到越来越广泛的部署和运用。交通预测可以帮助城市和交通管理部门在开展城市建设和交通设施规划时，进行高质量的预研和优化。传统的预测方法受限于采样率低、数据源单一以及获取方法有限等因素，往往只能通过小范围模型等方法进行近似评估。随着交通信息网络和智能交通监控系统的逐步完善，来自各类监控设备的视频图像信息正在 AI 与大数据方法的加持下，帮助人们获得更大范围的交通出行分布规律，从而得到准确可靠的交通预测模型。

智能交通预测中常见的 AI 模型

■ XGBoost 模型

基于树的机器学习方法是常用的预测模型之一，其优势在于预测结果通常具备较好的结果可解释性。其中 XGBoost 是一种重要的机器学习模型，是 boosting 的集成学习，由大量 CART 回归树集合而成的强分类器。CART 回归树是在二叉树上根据特征不断进行分叉，并在该特征维度上对样本空间进行划分。

XGBoost 方法的核心思想，是通过不断进行特征分裂来生成新的分叉树，每添加一个树，其实就是学习一个新函数来拟合上次预测的残差。与其它机器学习方法相比，XGBoost 有着如下的优势：

- XGBoost 对并行计算有着良好的支持，可充分利用处理器的多线程能力；
- XGBoost 在其代价函数中引入了正则化项，可以有效控制模型的复杂度来防止模型过拟合；
- XGBoost 支持列抽样（column subsampling）方式，不仅能够防止过拟合，还能降低计算复杂度；
- GBDT 等机器学习方法在进行优化时只用到一阶导数信息，而 XGBoost 则对代价函数进行了二阶泰勒展开，同时用到了一阶和二阶导数，有着更好的预测效果。

■ LSTM 模型

近年来，越来越多的深度学习方法也被用于智能交通预测之中，例如长短期记忆网络（Long Short-Term Memory，LSTM）就是其中一种常见的深度学习模型。LSTM 是递归神经网络（Recurrent Neural Networks，RNN）重要的衍伸模型，其可以通过 3 个特别的“门”结构设计来避免典型 RNN 结构中的长期依赖问题，使其大幅度提升记忆时长。

如图 5-1 所示，LSTM 中的“门”是 Sigmod 神经网络和位乘法的结合体，以 Sigmod 为激活函数的全连接神经网络层会输出一个 0 到 1 之间的值，描述当前可通过该结构的输入信息量，当 Sigmod 输出为 1 时，全部信息都可通过；反之，当 Sigmod 输出为 0 时，任何信息都无法通过。

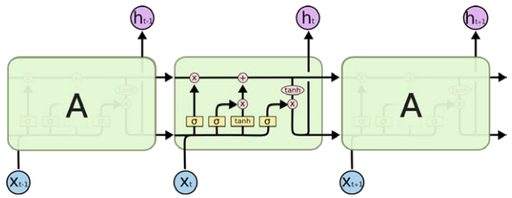


图 5-1 LSTM 网络结构

基于 LSTM 的深度学习方法可以良好地运用在基于序列的预测分析任务中。在实践运用中，通过与交通流量的时间分布特征相结合，其能够有效地对交通流量的变化趋势进行预测分析。

英特尔产品与技术为 AI 与大数据的融合提供通途

日益庞大的数据规模能有效推动机器学习、深度学习等 AI 方法在交通领域的发展，使各类智能交通应用在效率和准确性上获得长足进步。但在实际生产环境中，来自各个节点的海量数据要为 AI 系统所用，就需要对数据开展大量的计算处理工作。例如交通领域常见的视频图像，通常需要进行补色、合并、压缩以及裁剪等步骤。同时，经 AI 系统处理后的数据也需要与交管系统产生联动，发挥数据价值。

传统上，在智能交通或其它实际运用中，用户通常需要部署两套独立的工作集群来对大数据和 AI 任务进行分别处理。如图 5-2 所示，在一个典型的处理流程中，来自外部设备及数据仓库的海量数据首先会在左侧的分布式文件系统之上进行分布式数据处理和存储。而需要运用这些数据推进 AI 任务时，需要借助特定的数据通道将数据传输到 AI 工作集群，并在这里使用 TensorFlow、PyTorch 等 AI 框架进行模型训练和推理。

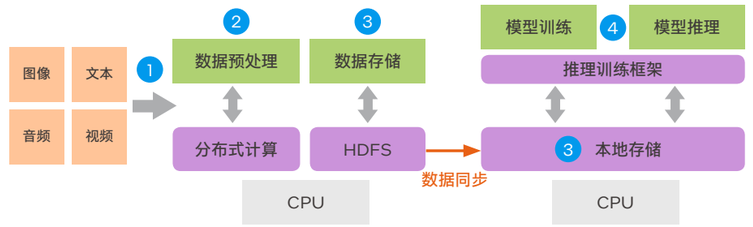


图 5-2 传统各自独立的大数据和 AI 处理流程

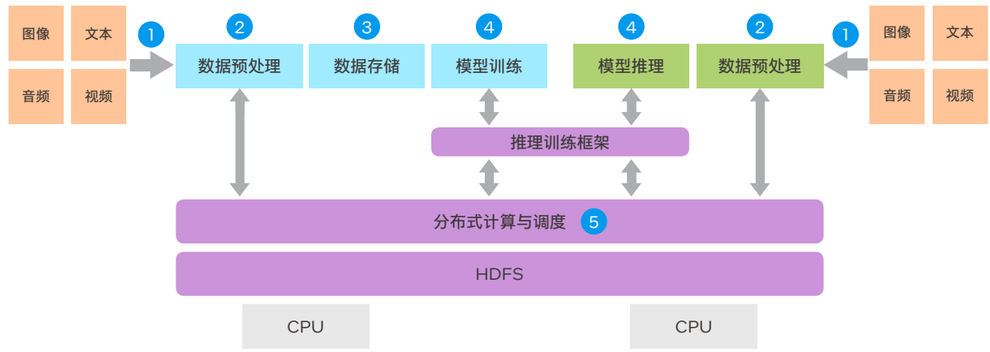


图 5-3 基于统一架构的大数据和 AI 处理流程

这不仅会带来巨大的数据传输开销，也需要额外的编排工作来保证数据通路的高效性。随着用户数据规模和应用复杂度的不断提升，这一模式在性能、便捷性以及可维护性等方面也面临着越来越多的挑战。而将分布式数据处理和 AI 任务无缝集成到一个流水线（Pipeline）中，是应对以上挑战的有效手段。

BigDL 2.0 简介

BigDL 2.0（以下简称“BigDL”）通过一个统一的架构，为分布式大数据处理与 AI 任务提供了高效的端到端管道。这帮助用户不仅能在同一集群中使用 Spark 等工具处理数据仓库中的数据，也可让 TensorFlow、PyTorch 等 AI 框架透明地调用经处理的数据，同时还能方便地在其中引入 OpenVINO™ 工具套件等优化组件。

除内置了面向 Apache Spark 的分布式深度学习库 DLlib（具有 Keras 风格的 API，并支持 Spark 机器学习管道的原始 BigDL 框架），BigDL 还集成了以下工具组件：

- Orca：为分布式大数据提供面向 TensorFlow 和 PyTorch 管道的无缝扩展；
- PPML：面向大数据分析和机器学习提供隐私保护功能；
- Friesian：一个大规模的端到端推荐解决方案框架；
- Chronos：使用 AutoML 的可扩展时间序列分析。

■ 分布式深度学习库 DLlib 代码示例：

```
#Keras-like API for BigDL
model = Sequential().add(InputLayer(inputShape = Shape(10)) \
  .add(Dense(12)).add(Activation("softmax"))
model.compile(...)

#Spark Dataframe preprocessing
trainingDF = spark.read.parquet("train_data")
validationDF = spark.read.parquet("val_data")

#Spark ML Pipeline for BigDL
scaler = MinMaxScaler(inputCol="in", outputCol="value")
estimator = NNEstimator(model, CrossEntropyCriterion()) \
  .setBatchSize(size).setOptimMethod(Adam()).setMaxEpoch(epoch)
pipeline = Pipeline().setStages([scaler, estimator])
pipelineModel = pipeline.fit(trainingDF)
predictions = pipelineModel.transform(validationDF)
```

在大数据处理或 AI 任务中，用户面临的一个重要挑战是如何将本地开发调试完毕的程序代码无缝地延展到分布式集群上运行。以往这项工作往往需要用户重新编写代码来实现。通过 Orca，用户可以方便地将本地 Jupyter Notebook[15] 等工具上运行的程序代码部署到分布式集群中去（例如 Hadoop 集群、Kubernetes 集群等），由此用户可以轻松地为分布式数据集构建并行预处理过程，并让用户在同一程序中将预处理过的数据应用于 AI 模型的训练与推理。

■ Orca 代码示例：

```
#PySpark DataFrame
train_df = sqlcontext.read.parquet(...).withColumn(...)

#TensorFlow Model
from tensorflow import keras
...
model = keras.Model(inputs=[user, item], outputs=outputs)
model.compile(optimizer= "adam",
              loss= "sparse_categorical_crossentropy",
              metrics=['accuracy'])

#Distributed training on Spark
from bigdl.orca.learn.tf.estimator import Estimator
est = Estimator.from_keras(keras_model=model)
est.fit(train_df, feature_cols=['user', 'item'], label_cols=['label'])
```

另一方面，随着数据安全和隐私保护受到越来越多的关注，“可用不可见”也逐步成为数据运用时的“必选项”之一。BigDL 通过 PPML 组件让用户能在大数据与 AI 平台中创建一个可信环境。用户可使用加密技术保护相关的数据或 AI 模型后，通过 PPML 直接基于加密数据运行 AI 任务和大数据处理任务，PPML 能在确保数据安全性的前提下，在可信环境中读取加密数据，解密并运行相应的工作任务。基于这一功能特性，用户也可以很方便地在 BigDL 平台上引入英特尔® SGX、英特尔®TDX 等技术构建硬件级安全环境，开展联邦学习等多方数据参与的 AI 任务。

与此同时，BigDL 也对诸多的 AI 应用场景提供了支撑。例如大规模端到端推荐解决方案框架 Friesian，其预置了丰富的特征工程操作、推荐算法和参考样例，能帮助用户快速构建一个完整的推荐系统。而 Chronos 组件则可以帮助客户快速构建一个高效、准确、可扩展的时间序列分析应用框架，其中包括了对 AutoML 技术的支持。

■ Chronos 代码示例：

```
sc = init_orca_context(...,
    init_ray_on_spark=True)
auto_est = AutoProphet(...)

#auto_est = AutoXGBRegressor(...)
data = get_data()
search_space = {
  "changepoint_prior_scale": ...,
  "seasonality_prior_scale": ...,
  ...
}
auto_est.fit(data=data,
  search_space=search_space,
  ...)
best_model = auto_est.get_best_model()
```

Chronos 内置了数据处理 & 特征工程、内置模型及可选的 HPO 三个组件，功能分别为：

- **数据处理 & 特征工程（Data Processing & Feature Engineering）组件**：其内置了 70 多个数据处理和特征工程工具，通过 TSDataset API 接口来帮助用户方便地调用，从而快捷高效地完成数据预处理和特征工程流程，用于时序数据的预处理包括滑动窗口取样、缩放、重采样、补全等以及自动的特征提取；
- **内置模型（Built-in Models）组件**：内置了 10 余个可用于时间序列预测、检测和模拟的独立深度学习和机器学习模型，功能涵盖预测器（Forecasters）、检测器（Detectors）以及模拟器（Simulators）；

[15] Jupyter Notebook 是一款基于网页的应用程序，可方便地应用于包括开发、文档编写、代码运行和结果展示的全过程，是数据科学家和 AI 工程师们常用的工具之一。

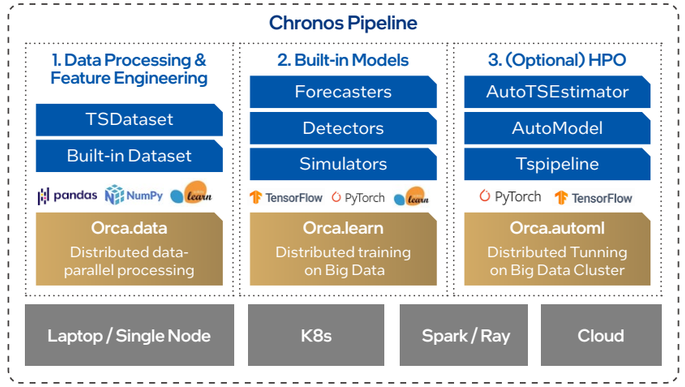


图 5-4 Chronos 框架基本架构

- HPO（Hyperparameter optimization）组件：其高度集成、可扩展和自动化的工作流（通过 AutoTSEstimator 等 API 实现），能帮助用户实现从数据预处理、特征工程到模型训练、模型选择和超参调优等全栈的自动化机器学习过程。英特尔提供的多种优化方式，例如所集成的 ONNX runtime 以及英特尔® oneAPI AI Analytics Toolkit 等，均能在推理过程中提供良好支持。

借助 Chronos 组件，用户可以基于 XGBoost、LSTM 等算法打造有效的交通流量预测方案。方案可以使用交通流量、天气等历史数据进行建模，并由 Chronos 框架提供的 TSDataset API 接口对时间序列数据快速执行填充、缩放等操作，并开展自动特征生成。然后通过 AutoTSEstimator 等 API 进行超参数搜索，并根据预测目标检索出最佳超参数集，进而优化模型和数据处理工序并形成时间序列预测模型。

一般地，BigDL 可为用户带来以下显著优势：

- 一个大数据分析和人工智能的统一平台；
- 提高资源利用率，加快 AI 应用程序的部署；
- 提高系统在分布式训练和推理上的可扩展性；
- 增强基础架构效率并降低 TCO。

BigDL 2.0 的安装与运行

BigDL 不仅可以方便地安装在 Linux、Windows 等不同操作系统上，也可以在 Docker、Hadoop 等不同环境中部署。同时，BigDL 也与 Jupyter Notebook 等工具有着良好的协同能力。以下是 BigDL 在 Linux、Windows 以及 Docker 环境中的安装示例。

■ BigDL 在 Docker 环境中的安装与运行

用户可以方便地在容器环境（Docker）中部署 BigDL，以下是对安装运行过程的简要描述。

 # 步骤 1

在拉取 Docker 镜像后，用户可以启动一个 BigDL Docker 容器。

```
sudo docker run -it --rm --net=host \
  -e http_proxy=http://your-proxy-host:your-proxy-port \
  -e https_proxy=https://your-proxy-host:your-proxy-port \
  intelanalytics/bigdl:2.3.0 bash
```

 # 步骤 2

容器成功启动后，用户会自动登录到容器中并转到 /opt/work 目录，该目录包含了克隆自 https://github.com/intel-analytics/BigDL.git 的 BigDL，包括应用程序及使用 BigDL 的示例，同时该目录下还有以下内容：

- start-notebook.sh：用于启动 Jupyter Notebook，用户可以为之指定启动时的环境设置和 Spark 设置；
- bigdl-${ bigdl_VERSION }：BigDL 发行版的 BigDL 主页；
- spark-${spark_VERSION}：Spark 的主页；
- opt/download-bigdl.sh：用于下载 bigdl 发行版。

```
root@[hostname]:/opt/work#
```

 # 步骤 3

在 Docker 容器启动并登录到该容器后，用户可以在容器内启动 Jupyter Notebook 服务。在 /opt/work 目录中，运行以下命令行以启动 Jupyter Notebook 服务。

```
./start-notebook.sh
```

运行该命令后，如果看到以下输出消息则意味着 Jupyter Notebook 服务已在容器中成功启动。

```
[I 07:40:39.354 NotebookApp] Serving notebooks from local directory: /opt/work/bigdl-2.1.0-SNAPSHOT/apps
[I 07:40:39.355 NotebookApp] Jupyter Notebook 6.4.6 is running at:
[I 07:40:39.355 NotebookApp] http://(the-host-name):12345/?token=...
[I 07:40:39.355 NotebookApp]  or http://127.0.0.1:12345/?token=...
[I 07:40:39.355 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
```

Jupyter Notebook 服务成功启动后，用户可以从浏览器连接服务。在获取容器的 IP 地址后，启动浏览器，并使用以下 URL 连接到服务：https://container-ip-address:port-number/?token=your-token，如下图所示：

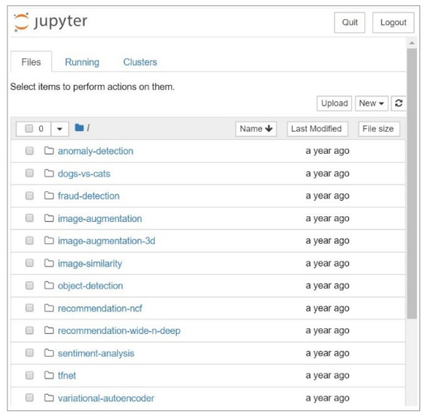


图 5-5 Jupyter Notebook 服务

用户可在其中运行多个 BigDL Jupyter Notebook 实例，例如以下使用模型微调来解决 “dogs-vs-cats（猫狗分类）” 问题。这里将使用预训练的 Inception_V1 模型进行操作，包括冻结前几层，替换顶部的分类器，然后对整个模型进行微调。

以下是其中的一些关键步骤说明：

 # 数据集和预训练模型导入

用户首先可在 https://www.kaggle.com/c/dogs-vs-cats 下载并解压缩数据集（内含 1100 幅猫狗照片），同时在 https://s3-ap-southeast-1.amazonaws.com/bigdl-models/imageclassification/imagenet/bigdl_inception-v1_imagenet_0.4.0.model 下载预训练的 Inception_V1 模型。

```
transformer = ChainedPreprocessing(
    [RowToImageFeature(), ImageResize(256, 256), ImageCenterCrop(224, 224),
     ImageChannelNormalize(123.0, 117.0, 104.0), ImageMatToTensor(), ImageFeatureToTensor()])
```

移除最后几个层，这里模型的输出层为：pool5/drop_7x7_s1。

```
for layer in full_model.layers:
  print(layer.name())
model = full_model.new_graph(["pool5/drop_7x7_s1"])
```

冻结前几个层，这里是从输入层到 pool4/3x3_s2。

```
model.freeze_up_to(["pool4/3x3_s2"])
```

添加一些新的层。

```
inputNode = Input(name="input", shape=(3, 224, 224))
inception = model.to_keras()(inputNode)
flatten = Flatten()(inception)
logits = Dense(2)(flatten)
lrModel = Model(inputNode, logits)
classifier = NNClassifier(lrModel, CrossEntropyCriterion(), transformer) \
    .setLearningRate(0.003).setBatchSize(4).setMaxEpoch(1).setFeaturesCol("image") \
    .setCachingSample(False)
pipeline = Pipeline(stages=[classifier])
```

 # 模型训练

```
catdogModel = pipeline.fit(trainingDF)
predictionDF = catdogModel.transform(validationDF).cache()

predictionDF.select("name","label","prediction").sort("label", ascending=False).show(10)
predictionDF.select("name","label","prediction").show(10)
correct = predictionDF.filter("label=prediction").count()
overall = predictionDF.count()
accuracy = correct * 1.0 / overall
print("Test Error = %g " % (1.0 - accuracy))
```

 # 结果可视化

用户可以随机选择一些预测结果的图像进行展示。

```
samplecat=predictionDF.filter(predictionDF.prediction==1.0).limit(3).collect()
sampledog=predictionDF.filter(predictionDF.prediction==2.0).sort("label", ascending=False).limit(3).collect()

from IPython.display import Image, display
for cat in samplecat:
  print ("prediction:", cat.prediction
  display(Image(cat.image.origin[5:]))

for dog in sampledog:
  print ("prediction:", dog.prediction
  display(Image(dog.image.origin[5:]))
```

上述任务的更多细节，可参阅：https://github.com/intel-analytics/BigDL/blob/main/apps/dogs-vs-cats/transfer-learning.ipynb

■ BigDL 在 Windows 环境中的安装与运行

对于 Windows 用户而言，可使用 Windows Subsystem for Linux 2 (WSL2) 环境来运行 BigDL，建议使用 Windows 10 version 2004 以上 (Build 19041 或更高版本)，或者 Windows 11。

 # 步骤 1

用户需以管理员身份打开 Windows 系统的 PowerShell 或 Windows 命令提示符，然后输入以下命令安装 WSL2，重新启动计算机并等待 WSL2 成功安装。值得注意的是，此处默认的 Linux 发行版为 Ubuntu。

```
wsl --install
```

 # 步骤 2

用户可以使用 conda 来管理 BigDL 环境，用户可先下载并安装 conda。

```
wget https://repo.continuum.io/miniconda/Miniconda3-4.5.4-Linux-x86_64.sh
chmod +x Miniconda3-4.5.4-Linux-x86_64.sh
./Miniconda3-4.5.4-Linux-x86_64.sh
```

安装 conda 结束后，可使用 conda 创建并激活 BigDL 的安装环境，并安转 BigDL，安装方法与 Linux 相同。

```
conda create -n bigdl-env
conda activate bigdl-env
pip install bigdl
```

 # 步骤 3

使用 pip 命令安装 Jupyter Notebook 环境并启动 JupyterLab。

```
pip install jupyterlab
jupyter lab
```

■ BigDL 在 Linux 环境中的安装与运行

BigDL 在 Linux 环境安装时，需要准备相应的 Python 或 Scala 语言环境，以下以 Python 语言为例对安装运行过程做简要描述。在 Python 语言中，可借助 pip 命令快捷地安装 BigDL。

 # 步骤 1

准备 Python 环境，（本示例使用 conda 以及 python 3.7，用户可根据实际情况调整），安装 JDK，并正确设置环境变量 JAVA_HOME（建议使用 JDK8）。代码示例如下：

```
conda create -n bigdl python=3.7
# "bigdl" is conda environment name, you can use any name you like.
conda activate bigdl

# JDK For Ubuntu
sudo apt-get install openjdk-8-jre
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64/
# JDK For CentOS
su -c "yum install java-1.8.0-openjdk"
export JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.282.b08-1.el7_9.x86_64/jre

export PATH=$PATH:$JAVA_HOME/bin
java -version  # Verify the version of JDK.
```

 # 步骤 2

安装最新版本的 BigDL（默认构建在 Spark 2.4.6 之上，也可构建在 Spark 3.1.3 之上）时会自动安装所有 BigDL 包，包括 BigDL-nano、BigDL-dllib、BigDL orca、BigDL-chronos、BigDL friesian、BigDL-serving 及相关依赖项。

```
#构建在 Spark 2.4.6 之上
pip install bigdl
pip install --pre --upgrade bigdl  # Install the nightly build version
#构建在 Spark 3.1.3 之上
pip install bigdl-spark3
pip install --pre --upgrade bigdl-spark3  # Install the nightly build version
```

 # 步骤 3

使用 Python shell 来验证安装是否成功，然后启动 Jupyter Notebook，并直接在 Jupyter Notebook 中运行 BigDL 程序。在 BigDL 程序编写上，用户可以直接在 Python 文件（例如 script.py）中编写程序，并在命令行中作为普通 Python 程序运行：

```
from bigdl.orca import init_orca_context
#在集群上初始化 BigDL
sc = init_orca_context()
#在 Jupyter Notebook 中运行 BigDL 程序
jupyter notebook --notebook-dir=./ --ip=* --no-browser
#在 Python 中运行 BigDL 程序
python script.py
```

BigDL 在其它环境的安装与运行方法，可参阅 BigDL 官网：https://bigdl.readthedocs.io/en/latest/index.html

实战案例

某铁路运营商基于英特尔产品，加速释放铁路运力

■ 项目背景

在许多国家，无论是客运还是货运，铁路都是运输基础设施的重要组成部分。随着需求的增长，铁路行业正面临着如何增加运力的挑战。由于没有足够的土地和预算来扩大铁路网，因此铁路行业需要能够利用现有基础设施提升运力的技术解决方案。

运力的提升有赖于延误率的降低和事故的减少。但长期以来，铁路运输部门的调度人员需要通过与沿线进行电话沟通才能就某些变动，例如延误、铁路维护等达成一致，这无疑极大阻碍了铁路运输效率的进一步提升。某铁路运营商希望通过 AI 与大数据能力的引入来开展铁路延误情况的预测，并结合这一预测能力，使运输调度人员第一时间就可看到线路端到端的运行状况并做出正确的策略。通过调度指挥中心之间的数字化数据流动，使某铁路运营商在不新建新基础设施的情况下提高铁路网运力，提高旅客满意度并提升货物运输效率。

■ 解决方案

新的解决方案引入 AI 与大数据能力对未来的延误情况进行预测。一方面，方案通过 LSTM 模型对目标铁路网 10 年的历史数据进行训练，构建可识别导致运输中断规律的模型。

同时，方案也接入了大量的可用数据来与预测方案进行协同。例如所部署的物联网 (IoT) 网关也会从轨道旁收集列车行进数据，系统也可以随时访问时间表数据，以及有关维护、临时速度限制以及资产故障的信息。结合上述数据，方案可以确定是否有可能发生运输中断，并向运输调度人员发出警报，便于他们采取预防措施。

方案中，运输调度人员可以实时查看线路端到端的运行状况。每天早上，他们首先根据轨道封闭或列车可用情况来预测可能导致最大问题出现的原因。当运输调度人员确定好需要采取的措施时，系统可以对决策进行模拟，便于他们看到变更带来的影响并对方案进行测试。如果测试结果让运输调度人员满意，系统就可以自动与信号传输系统通信实施变更。

例如方案可以预测到因两列火车中有一列发生延误，所以在一小时后它们可能会需要使用同一站台。这时，运输调度人员就可以进行干预，让其中一列火车停靠到另一个空闲站台。这样一来，不仅可以避免当前运输服务出现延误，还能保证未来需要这两列火车及时投入使用，使得提供的服务不受影响。

来自英特尔的 BigDL 为新方案中的大数据与 AI 处理提供了统一且高效的工作平台，便于来自大量传感器、信号机以及历史数据仓库的海量数据在其上得到高速处理。而英特尔® 至强® 可扩展处理器系列也为大数据与 AI 工作负载提供了强有力的算力支持。

■ 方案成效

在新方案获得部署后，该铁路运营商的调度指挥中心之间依靠运输调度人员相互电话沟通的方式得到了完全的改变。新的信息数字流极大地简化了决策和实施工作。在出现延误可能性时，运输调度人员能够提前介入，通过重新安排列车来减少延误，从而大幅提升了该公司旗下铁路网的运行效率以及旅客满意度。

浪潮信息基于 BigDL，打造端到端智能交通平台

■ 项目背景

在智慧交通建设已成为各地深化交通领域改革重要抓手的今天，浪潮电子信息产业股份有限公司（以下简称“浪潮信息”）也通过 Insight 智能交通解决方案的构建，为各级交通运输部门提供了端到端的智能交通解决方案，并在多个省市落地应用。

■ 解决方案

如图 5-6 所示，Insight 智能交通解决方案是在浪潮信息既有 Insight 大数据平台的基础上，通过引入 BigDL 2.0 来加入了一系列 AI 能力。其中，Insight 大数据平台是在优化的基础设施层上，集成了计算引擎、存储服务、数据服务、数据工作台和管理平台，构建起了端到端的大数据分析和深度学习应用程序管道。同时，平台也可以将 Spark、TensorFlow、Keras 和 BigDL 程序无缝集成到一个集成的管道中，该管道可以透明地扩展到大规模的 Apache Hadoop/Spark 集群，用于分布式训练或推理。

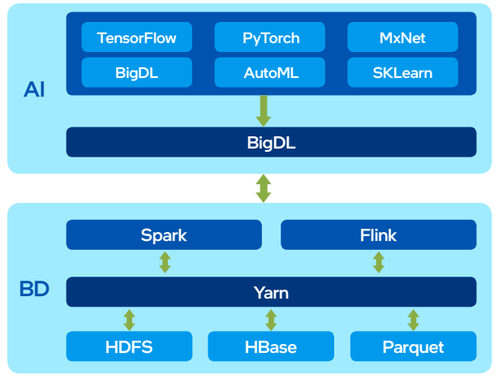


图 5-6 浪潮 Insight 智能交通解决方案

平台的工作流程示例如下（示例代码基于 Insight v6.3 版本，使用 BigDL-Analytics Zoo 0.9 完成）

 # 步骤 1

构建分布式大数据读取，使用 write_parquet 将本地数据转换为 parquet 数据，并将其保存到 HDFS。使用 read_parquet 从 HDFS 加载的 parquet 数据中获取 tf.data.Dataset。

```
from bigdl.orca.data.image.parquet dataset import write_parquet
write_parquet('kitti tracking',
    output_path="hdfs://output_dir",
    image_dir="~/kitti tracking/training/image_02",
    anno_file="./annotations/tracking_train.json")

from bigdl.orca.data.image import read_parquet
import tensorflow as tf
train_datasets = read parquet(format="tf dataset"
    path="hdfs: //output dir",
    output types=(tf.float32, tf.float64),
    output shapes=((516,516,3),(129,129,9)))
```

 # 步骤 2

基于 BigDL 构建分布式训练任务，包括构建数据生成器并加载训练数据集和验证数据集，以及构建 ML/DL 模型。

```
def train data creator(config)
    #load data
    train_datasets = read_parquet(format="tf_dataset",
        path="hdfs://output dir",
        output types=(tf.float32, tf.float64),
        output shapes=((516,516,3),(129,129,9)))
    #augment data
    datasets = datasets.map(transform).shuffle(100).batch(config['batch_size'])
    return datasets

def valid data creator(config):...

def model_creator(config):
    #using keras api build training model
    net = FairMot(config)
    return net
```

 # 步骤 3

基于 BigDL 构建分布式训练任务，包括分布式环境初始化以及分布式训练 ML/DL 模型。

```
init_orca_context(cluster mode="yarn-client",
    num nodes=8，cores=20,
    init_ray_on_spark=True,
    memory="20g",driver memory="16g",
    hadoop_user_name= 'hdfs',
    hadoop_conf="/etc/hadoop/3.0.1.0-187/0/*")

#Estimator load keras Model
est = Estimator.from_keras(model_creator, config=config)

#distributed training keras Model
est.fit(train_data_creator,
  epochs=10
  steps_per_epoch=train_num // batch size,
  validation_data_creator=valid_data_creator,
  validation_steps=valid_num // batch size)
```

在实战中，以该平台在视频监控领域的应用场景为例，平台会将大数据平台的视频流数据输入到多目标跟踪模型中进行推理，并预测图像中目标的位置和 ID 信息。预测的车辆位置信息被用来截取车辆数据，并且车辆数据被传递到 OCR 模型中以识别车牌信息。将绑定车辆 ID 信息和车牌信息的车辆轨迹发送到 Insight 大数据平台。

在其中的多目标跟踪模型工作流程中，如图 5-7 所示，时间 T-1 是视频的前一帧的图像。通过算法预测，得到目标的类别、位置和特征向量。时间 T 是视频的当前帧的图像。模型预测后，将与前一帧的预测结果（类别、位置、特征向量）进行融合，以匹配相同的目标。

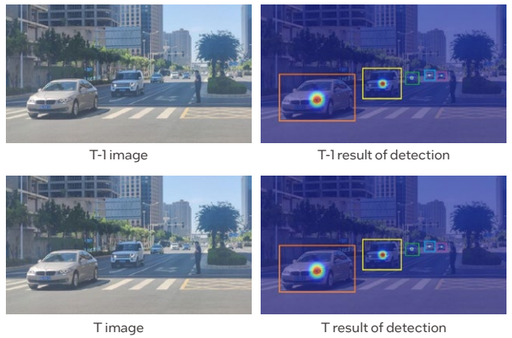


图 5-7 Insight 智能交通解决方案多目标跟踪模型工作流程

■ 方案成效

目前，浪潮 Insight 智能交通解决方案已经在交通路口车辆检测与跟踪系统、交通路口车辆轨迹记录系统以及基于大数据平台的车辆轨迹跟踪系统等智能交通系统中得到了运用，并取得了良好的应用效果。

技术篇

第四代英特尔® 至强® 可扩展处理器

第四代英特尔® 至强® 可扩展处理器旨在为人工智能、数据分析、存储和科学计算方面快速增长的工作负载提供性能加速。该处理器具备多种内置加速器，帮助客户将零信任安全策略付诸实践，同时利用先进的安全技术，即使面对敏感或受监管的数据，也能解锁新的商业合作机会和洞察。使用这款处理器可跨多个云和边缘环境进行扩展，满足自身的部署需求。英特尔® 至强® 可扩展处理器具有很强的灵活性，可在其上选择不同的云服务，帮助企业顺利实现应用移植。

基础性能进一步大幅提升

- 第四代英特尔® 至强® 可扩展处理器采用全新架构，单核性能比上一代产品更高，每路配备多达 60 个内核。每个系统支持单路、双路、四路或八路配置。为了与内核数增加这种情况相匹配，该平台在内存和 I/O 子系统方面也做了相应改进。DDR5 内存提供的带宽和速度与 DDR4 相比提高多达 1.5 倍，速率达到 4,800 MT/s [16]。此外，该平台还具有每路 80 条 PCIe Gen5 通道的特点，与之前的平台相比，I/O 得到显著提升。本代处理器还可提供 CXL 1.1 连接，支持高网络带宽并使附加加速器能够高效运行。第四代英特尔® 至强® 可扩展处理器支持的技术支持根据工作负载要求的变化灵活扩展和调整。此外，本代处理器还可助力实现以下优势：
- 进一步提升网络、存储和计算性能，并通过将繁重的任务卸载到英特尔® 基础设施处理单元（Intel® Infrastructure Processing Unit，英特尔® IPU）来提高 CPU 利用率；
- 通过英特尔® UPI 2.0 提高多路带宽（高达 16 GT/s）；
- 使用英特尔® Speed Select 技术（英特尔® SST）调整 CPU 配置，满足特定工作负载的需求；
- 增加三级缓存（LLC）共享容量（所有内核共享多达 100 MB LLC）；
- 通过硬件增强型安全功能加强对安全态势的掌控；
- 使用英特尔® Virtual RAID on CPU（英特尔® VROC），从而无需再用单独的 RAID 卡。

第四代英特尔® 至强® 可扩展处理器的新特性或新功能

PCI Express Gen5（PCIe 5.0）

带来全新的 I/O 速度，可在 CPU 和互联设备之间实现更高的吞吐量。第四代英特尔® 至强® 可扩展处理器具有多达 80 条 PCIe 5.0 通道，非常适合高速网络、高带宽加速器和高性能存储设备。PCIe 5.0 的 I/O 带宽是 PCIe 4.0 的两倍，仍具备向后兼容性并提供用于 CXL 连接的基础插槽 [17]。

DDR5

以更高内存带宽克服数据瓶颈，提高计算性能。与 DDR4 相比，DDR5 的带宽提高多达 1.5 倍，因此有机会提升性能、容量和能效并降低成本 [18]。借助 DDR5，第四代英特尔® 至强® 可扩展处理器提供的速率可高达 4,800 MT/s（1 DPC）或 4,400 MT/s（2 DPC）。

CXL

借助面向下一代工作负载的 CXL 1.1，降低数据中心的计算时延并帮助减少 TCO。CXL 是另一种跨标准 PCIe 物理层运行的协议，可以在同一链路上同时支持标准 PCIe 设备和 CXL 设备。CXL 可带来的一大关键能力是在 CPU 和加速器之间创建统一且一致的内存空间，它将革新未来数年数据中心服务器架构的构建方式。

内置众多加速引擎，重新定义性能

与增加 CPU 内核数相比，内置加速器是一种提升性能更有效的方法。其不但可以提高 CPU 利用率，降低功耗，并提高投资回报率（ROI），同时还能帮助企业实现可持续发展目标。英特尔® 至强® 可扩展处理器支持广泛且独特的内置加速器，有助于提高性能和效率，减少另行添置专用硬件的需求。在云端和本地环境中，这些专用功能支持人工智能、安全性、科学计算、数据分析、存储和网络等目前最为常见的严苛工作负载。

16、17、18 https://www.intel.cn/content/www/cn/zh/products/docs/processors/xeon-accelerated/4th-gen-xeon-scalable-processors-product-brief.html

- **英特尔® 高级矩阵扩展（Intel® Advanced Matrix Extensions，英特尔® AMX）**可加速自然语言处理（NLP）、推荐系统和图像识别等深度学习（DL）推理和训练工作负载。
- **英特尔® 数据流加速器（Intel® Data Streaming Accelerator，英特尔® DSA）**可通过优化流数据的传输和转换操作，大幅提升存储、网络和数据密集型工作负载的性能。
- **英特尔® 存内分析加速器（Intel® In-Memory Analytics Accelerator，英特尔® IAA）**可提高数据分析性能，同时从 CPU 内核上卸载任务，为数据库查询及其他工作负载加速。
- **英特尔® 动态负载均衡器（Intel® Dynamic Load Balancer，英特尔® DLB）**可随系统负载的变化将网络数据动态地分配到多个 CPU 内核上，基于硬件高效实现负载平衡。
- **面向 vRAN 的英特尔® 高级矢量扩展（Intel® Advanced Vector Extensions，英特尔® AVX）**在相同功耗范围内可将虚拟无线接入网络（vRAN）的密度较上一代产品提高多达 2 倍 [19]。
- **英特尔® 高级矢量扩展 512（Intel® Advanced Vector Extensions 512，英特尔® AVX-512）**支持多达两个融合乘加（FMA）单元，并包含多项优化，可为要求严苛的计算任务提高性能。
- **英特尔® 数据保护与压缩加速技术（英特尔® QAT）**可加速加解密和数据压缩，它通过从处理器内核卸载这些任务，帮助降低系统资源消耗。
- **英特尔® 密码操作硬件加速（Intel® Crypto Acceleration）**降低了实施普遍数据加密的影响，并提高了安全套接字层（SSL）Web 服务器、5G 基础设施和 VPN/防火墙等加密敏感型工作负载的性能。

英特尔® 至强® 可扩展处理器 PLATINUM	英特尔® 至强® 可扩展处理器 GOLD	英特尔® 至强® 可扩展处理器 SILVER
多达 8 路的可扩展性	多达 4 路的可扩展性	多达 2 路的可扩展性
4 个英特尔® UPI 端口，速率为 16 GT/s	3 个英特尔® UPI 端口，速率为 16 GT/s	2 个英特尔® UPI 端口，速率为 16 GT/s
80 条 PCIe 5.0 通道 + CXL	80 条 PCIe 5.0 通道 + CXL	80 条 PCIe 5.0 通道 + CXL
支持 DDR5，速率高达 4,800 MT/s（每通道 1 个 DIMM）或 4,400 MT/s（每通道 2 个 DIMM）	支持 DDR5，速率高达 4,800 MT/s（每通道 1 个 DIMM）或 4,400 MT/s（每通道 2 个 DIMM）	支持 DDR5，速率高达 4,800 MT/s（每通道 1 个 DIMM）或 4,400 MT/s（每通道 2 个 DIMM）
支持英特尔® 傲腾™ 持久内存 300 系列	支持英特尔® 傲腾™ 持久内存 300 系列	英特尔® AVX-512（两个 512 位 FMA）
英特尔® AVX-512（两个 512 位 FMA）	英特尔® AVX-512（两个 512 位 FMA）	英特尔® 超线程技术和英特尔® 睿频加速技术
英特尔® 超线程技术和英特尔® 睿频加速技术	英特尔® 超线程技术和英特尔® 睿频加速技术	英特尔® 深度学习加速技术和英特尔® AMX
英特尔® AMX	英特尔® 深度学习加速技术和英特尔® AMX	英特尔® SGX 最大飞地容量高达 64 GB
英特尔® SST	英特尔® SST	可通过英特尔® QAT、英特尔® DLB、英特尔® DSA 和英特尔® IAA 加速工作负载
先进的可靠性、可用性和可维护性（RAS）	先进的 RAS	
英特尔® SGX 最大飞地容量高达 128 GB（在特定型号的 SKU 上最大飞地容量高达 512 GB）	英特尔® SGX 最大飞地容量高达 128 GB	
可通过英特尔® QAT、英特尔® DLB、英特尔® DSA 和英特尔® IAA 加速工作负载	可通过英特尔® QAT、英特尔® DLB、英特尔® DSA 和英特尔® IAA 加速工作负载	

19 https://www.intel.cn/content/www/cn/zh/products/docs/processors/xeon-accelerated/4th-gen-xeon-scalable-processors-product-brief.html

英特尔® 至强® 铂金 8400 处理器是打造安全且敏捷的混合云数据中心的基石，专为高级数据分析、AI、高密度基础设施和多云工作负载而设计。这些处理器具备更高性能、强大的平台功能和出色的工作负载加速能力。它们具有更出色的基于硬件的安全性和强大的多路处理性能——特定型号的英特尔® 至强® 铂金 8400 处理器支持多达 8 路配置。借助值得信赖且经过硬件增强的数据服务交付以及全新的 I/O 和连接技术，这些处理器在 I/O、内存、存储和网络技术方面均实现提升，因而能够更好地在数据驱动程度日益加深的世界中挖掘可执行洞察。相关提升包括：

- 每个英特尔® 至强® 可扩展处理器具备多达 60 个内核；
- 每个处理器有 8 条内存通道，速率高达 4,800 MT/s（1 DPC）；
- 英特尔® AMX 支持下的 AI 加速带来深度学习推理和训练性能的巨大飞跃。

AI：凭借更优的矢量指令和矩阵乘法运算，第四代英特尔® 至强® 可扩展处理器展现出更为出色的 AI 推理和训练性能。英特尔® AMX 可以显著提高推荐系统、NLP、图像识别、媒体处理和分发以及媒体分析等深度学习工作负载的性能。

扫码了解更多第四代英特尔® 至强® 可扩展处理器详情

访问链接了解更多第四代英特尔® 至强® 可扩展处理器详情：
https://www.intel.cn/content/www/cn/zh/products/docs/processors/xeon-accelerated/4th-gen-xeon-scalable-processors.html

扫码了解第四代英特尔® 至强® 可扩展处理器的配置和 AI 调优指南

访问链接了解更多第四代英特尔® 至强® 可扩展处理器的配置和 AI 调优指南：
https://www.intel.cn/content/www/cn/zh/artificial-intelligence/tuning-guide-ai-4th-gen-xeon-scalable-processors.html?token=6b6ebdd9e1f06549ee42bed0caee30a3c30f9f25188b69a3c8499ed49af45d4a&utm_content=2023LP_014&utm_campaign=Enterprise&utm_source=Intel%20Website&scino_token=6b6ebdd9e1f06549ee42bed0caee30a3c30f9f25188b69a3c8499ed49af45d4a

第三代英特尔® 至强® 可扩展处理器

英特尔对面向四路和八路的第三代英特尔® 至强® 可扩展处理器（Cooper Lake）和面向单路和双路的第三代英特尔® 至强® 可扩展处理器（Ice Lake）在多样化的工作负载类型和性能需求方面进行了优化。

基础性能

- 第三代英特尔® 至强® 可扩展处理器基于平衡、高效的架构构建，该架构可提升内核性能、内存和 I/O 带宽，为处理从数据中心到边缘的各种工作负载提速。在单路和双路配置中，支持每处理器多达 40 个内核，在四路和八路配置中则支持每处理器达 28 个内核，在八路配置下，单平台支持多达 224 个内核；
- 单个处理器支持 8 条 DDR4 内存通道（Ice Lake）或 6 条 DDR4 内存通道（Cooper Lake），最高速率为 3,200 MT/s。同时每路多达 64 条 PCI Express Gen4 通道，实现更高的每核 I/O 带宽；
- 多达 6 条英特尔® 超级通道互联（英特尔® UPI）通道有效提高了平台可扩展性以及 I/O 密集型工作负载的 CPU 间带宽，从而在提高吞吐量和能效之间达成平衡。

增强的 AI 加速与安全能力

- 第三代英特尔® 至强® 可扩展处理器加入了基于英特尔® AVX - 512 的增强版英特尔® 深度学习加速技术，同时支持 16 位 Brain Floating Point（BF16）和矢量神经网络指令（VNNI），有效加速人工智能推理和训练性能。其中 BF16 适用于特定型号的第三代英特尔® 至强® 可扩展处理器，其在视觉、自然语言处理和强化学习等需要兼顾吞吐量和准确率的 AI 应用场景可以提供更有效的训练与推理加速能力。而矢量神经网络指令（VNNI）能够充分提高计算资源和缓存的利用率、减少潜在的带宽瓶颈，以此增强推理工作负载；
- 单路和双路配置的第三代英特尔® 至强® 可扩展处理器对英特尔® SGX 提供支持，帮助用户无论是从边缘到数据中心还是到多租户公有云，都可以在更好地保护数据和应用代码安全的前提下，采用联邦学习等方法，以多源数据加强 AI 应用的应用效能。

自定义性能助推各种工作负载

- 第三代英特尔® 至强® 可扩展处理器增强了英特尔® SST（英特尔® Speed Select 技术）功能，其可以对处理器性能实施精细控制，有助于优化 TCO。大部分第三代英特尔® 至强® 铂金和金牌处理器都支持英特尔® SST– BF、英特尔® SST –CP 和英特尔® SST – TF 等不同模式的 SST，而第三代英特尔® 至强® 可扩展处理器 Y SKU 支持新的英特尔® SST- PP 模式，可以为用户提供更多内核、频率、外形尺寸和功率配置选择。

适用于不同工作负载的第三代英特尔® 至强® 可扩展处理器

- 英特尔® 至强® 铂金 8300 处理器是打造可靠、敏捷的混合云数据中心的基石。处理器具备增强型硬件安全功能以及出色的多路处理性能，适用于关键业务的实时分析、机器学习、人工智能、科学计算和多云工作负载。
- 英特尔® 至强® 金牌 6300 和 5300 处理器支持更高的内存速度、更大的内存容量以及多达四路的可扩展性，带来更出色的性能和内存功能、硬件增强型安全和工作负载加速。
- 英特尔® 至强® 银牌 4300 处理器提供基本性能、更快的内存速度以及更高的能效，为入门级数据中心计算、网络和存储带来所需的硬件增强性能。

访问链接了解更多第三代英特尔® 至强® 可扩展处理器详情
https://www.intel.cn/content/www/cn/zh/products/docs/processors/xeon/3rd-gen-xeon-scalable-processors-brief.html

扫码了解更多第三代英特尔® 至强® 可扩展处理器详情

英特尔® 至强® CPU Max 系列

过去十年，随着人工智能技术的加入，峰值算力大幅增长，但由于在向内核传输数据时效率低，因此工作负载性能未能同步提升。英特尔® 至强® CPU Max 系列的诞生，使英特尔® 至强® 平台如虎添翼，它是英特尔唯一一个基于 x86 架构并采用高带宽内存（HBM）的 CPU 系列，可释放和加速内存密集型科学计算和 AI 工作负载。

更高带宽，更优性能

英特尔® 至强® CPU Max 系列采用全新微架构，支持一系列可提升平台能力的特性，包括更多内核、先进的 I/O 与内存子系统，以及可加速重大发现的内置加速器。英特尔® 至强® CPU Max 系列具有以下特性：

- 多达 56 个 P-core（性能核）：内核由 4 个小芯片构成，采用英特尔的嵌入式多芯片互连桥接（EMIB）技术连接，功耗为 350 W；
- 64 GB 高带宽封装内存及 PCIe 5.0 和 CXL 1.1 I/O。英特尔® 至强® CPU Max 系列每核均具备 HBM 容量，可满足大多数常见科学计算工作负载的要求；
- 与其他 CPU 相比，在使用 Numenta 的 AI 技术进行自然语言处理时，其 HBM 优势可带来高达 20 倍的性能提升 [20]。

加速科学创新

英特尔® 至强® CPU Max 系列能够与英特尔® 至强® 平台实现轻松整合，不但可以获得处理要求严苛的工作负载所需的性能与能效，还可得到各种出色的内置加速器（包括英特尔® AMX，英特尔® DSA 等，具体详见第 53 页详细介绍）的助力。利用面向科学计算和 AI 工作负载的关键加速器，提高 CPU 使用效率、降低功耗、实现更高的投资回报率（ROI）。另外，由于处理器插槽（Socket）配置相同，可轻松将英特尔® 至强® CPU Max 系列处理器添加到第四代英特尔® 至强® 可扩展平台，并且在大多数部署方案中都无需更改代码。

访问链接了解更多英特尔® 至强® CPU Max 系列详情：
https://www.intel.cn/content/www/cn/zh/products/details/processors/xeon/max-series.html

扫码了解更多英特尔® 至强® CPU Max 系列详情

灵活应对各种科学计算和 AI 工作负载

英特尔® 至强® CPU Max 系列处理器具备出色的灵活性，可根据工作负载的特性，在不同的内存模式或配置下运行：

- **“仅 HBM” 模式**：该模式支持每颗 CPU 内存容量需求不超过 64 GB 的工作负载，同时无需更改代码和另购 DDR，即可启动系统，内存带宽会使用 HBM 的带宽，达到更优的性能；
- **“HBM Flat” 模式**：该模式可为需要大内存容量的应用提供灵活性，HBM 和 DRAM 同时作为内存使用，适用于内存需求大于每颗 CPU64G 的工作负载。使用该模式时需要显示使用 API 或者工具例如 numactl 显式指定使用的内存是 HBM 还是 DRAM；
- **“HBM 缓存” 模式**：旨在提升内存容量需求大于每颗 CPU64 GB 且无需更改代码的情况，HBM 部分作为 DDR 的缓存。

英特尔® 至强® CPU Max 系列	
内核数	32-56
HBM2e 内存	64 GB
DDR5 最大传输速率	4800MT/s（1 个 DPC） 4400 MTs（2 个 DPC）
加速器	AMX，4 个英特尔® DSA
AMX 支持的数据精度	AMX_BF16 和 AMX_INT8

跨多架构加速科学计算和 AI 工作负载

整个英特尔® 至强® CPU Max 系列的产品均得到 oneAPI 的支持。oneAPI 是一个统一的、基于标准的开放式通用编程模型，可释放生产力并解锁性能。开发人员可利用英特尔® oneAPI 工具套件以及面向特定领域的专用工具套件，打造跨多种架构运行的通用计算、科学计算和 AI 应用，并对其进行分析、优化和扩展。这些资源包括矢量化、多线程、多节点并行和内存优化方面的前沿技术，可轻松构建随时能为科学计算所用的高性能、多架构软件。

访问链接了解更多英特尔® 至强® CPU Max 系列配置和调优指南：
https://www.intel.cn/content/www/cn/zh/now/data-centric/xeon-cpu-max-series-configuration-tuning-guide.html

扫码了解英特尔® 至强® CPU Max 系列配置和调优指南

[20] https://www.intel.cn/content/www/cn/zh/products/docs/processors/xeon/xeon-max-series-product-brief.html

英特尔® 数据中心 GPU Flex 系列

越来越多的媒体处理和交付、AI 视觉推理、云游戏和桌面虚拟化任务在数据中心进行。但是这种迅速增长的势头因行业依赖专有的许可编码模型（如面向 GPU 编程的 CUDA）而大大受到限制。基于 CUDA 的软件也仅限用于专有 GPU，无法移植到其他加速器架构或 CPU 上。由此产生的 TCO 上升压力使专有 GPU 编程无法大规模进行。

英特尔® 数据中心 GPU Flex 系列克服了上述限制，同时还为视觉云工作负载提供了出色的计算密度和能效。该系列产品基于英特尔® Xe HPG（高性能显卡）微架构打造，内置视觉处理和 AI 加速技术。其提供的功能和优势包括：

- 支持开放、灵活、基于标准的软件堆栈以及 oneAPI 统一编程，其中包括用于构建高性能、跨架构媒体应用和解决方案的开源组件与库、工具及框架。这种开放的方法有助于生态系统摆脱使用专有编程模型带来的技术和经济负担；
- 开创性地在 GPU 内配置了基于硬件的开源 AV1 编码器，在相同质量下将带宽提高 30%，从而每年每十万名观众节省 2,300 万美元，或者在相同带宽下提高流媒体质量 [21]。

高达 68 路
720p30 游戏视频流（基于特定游戏），单个英特尔® GPU Flex 170 [22]

高达 46 路
720p30 游戏视频流（基于特定游戏），单个英特尔® GPU Flex 140 [23]

硬件规格 [24]

该系列以两种 SKU 形式提供：英特尔® 数据中心 GPU Flex 系列 170（峰值性能更高）和英特尔® 数据中心 GPU Flex 系列 140（密度更高）。图形处理器拥有多达 32 个英特尔® Xe 内核及光线追踪单元、多达 4 个英特尔® Xe 媒体引擎，具备用于 AI 加速的英特尔® Xe Matrix Extensions（英特尔® XMX），并且支持基于硬件的 SR-IOV 虚拟化。每卡配备两个图形处理单元（GPU）的 Flex 系列 140 利用英特尔® oneVPL Deep Link Hyper Encode 功能可满足业内的 “一秒时延” 要求，同时提供 8K60 实时转码能力。此功能适用于 AV1 和 HEVC HDR 格式。

	英特尔® 数据中心 GPU Flex 140	英特尔® 数据中心 GPU Flex 170
目标工作负载	媒体处理和交付、基于 Windows 和 Android 的云游戏、虚拟桌面基础设施、AI 视觉推理[2]	
显卡外形规格	半高、半长、单宽、被动散热	全高、四分之三长、单宽、被动散热
显卡 TDP	75 瓦	150 瓦
每卡 GPU 数量	2	1
GPU 微架构	Xe HPG	
Xe 内核数量	16 个（8 个/GPU）	32
Fixed Function Media	4（2 个/GPU）	2
光线追踪	是	
峰值算力（脉动阵列浮点运算）	8 TFLOPS (FP32)/105 TOPS (INT8)	16 TFLOPS (FP32)/250 TOPS (INT8)
内存类型	GDDR6	
内存容量	12 GB（6 GB/GPU）	16 GB
虚拟化（实例）	SR-IOV（62 个）	SR-IOV（31 个）
操作系统	Linux（Ubuntu、CentOS、Debian）、Windows Server 2019/2022、Windows Client 10、Red Hat® Enterprise Linux	
主机总线	PCIe Gen 4	
主机 CPU 支持	第三代 / 第四代英特尔® 至强® 可扩展处理器	

[21]、[22]、[23]、[24] https://www.intel.cn/content/www/cn/zh/customer-spotlight/cases/data-center-gpu-flex-series.html

按用例划分的软件堆栈

Flex 系列 GPU 支持开放、灵活、基于标准的软件堆栈和 oneAPI 跨架构编程。堆栈包括开源的组件与库、工具及框架。开发人员可以利用它们开发高性能、跨架构媒体应用和解决方案，满足广泛的用例需求。这种开放的方法消除了专有模型形成的障碍——使用专有模型、代码可移植性和采用跨多供应商的新架构的能力会受到限制。

通用软件功能集可集成到主流中间件和框架中，而堆栈能以经过验证的产品化容器或参考堆栈形式交付。开发人员可以在裸机上利用 Kubernetes 对这些容器进行编排，或者利用 SR-IOV 虚拟化技术及相关工具在虚拟机中进行编排，实现工作负载分配和管理。此工具集旨在加快上市速度，并支持在同一 GPU 上灵活部署多个工作负载。

英特尔通过开展行业合作、参与和资助多项计划以及参加各种标准组织为软件生态系统赋能，并为开源社区提供持续的领导力、投资和技术贡献。

注：oneDNN 指 oneAPI 深度神经网络库。oneDAL 指 oneAPI 数据分析库。oneVPL 指 oneAPI 视频处理库。oneVPL、oneDNN、oneDAL 和英特尔® VTune™ Profiler 包含在英特尔® oneAPI 基础工具套件内（各个工具可单独下载）。面向英特尔® 架构优化的 TensorFlow 和 PyTorch 包含在英特尔® AI 分析工具套件内。

英特尔锐炫™ 显卡（Intel ARC™ Graphics）

英特尔锐炫™ 显卡是英特尔全新高性能显卡产品品牌，专为消费端打造，涵盖硬件、软件和服务三方面。基于 X^e HPG 微架构，面向笔记本电脑、台式机、工作站分别规划有不同型号和功能的产品。其硬件产品将涉及多代，不仅包括首代基于 X^e HPG 微架构的 Alchemist 显卡（DG2），还将包括代号分别为 Battlemage、Celestial 和 Druid 的后续几代产品。

架构：英特尔锐炫™ 显卡基于 X^e HPG

X^e HPG 是一种针对游戏和桌面性能进行优化的全新独立显卡微架构，专为游戏和创作工作负载提供发烧级的高性能，采用全新的 X^e 内核，添加了基于 GDDR6 的新内存子系统以提高性价比，具有加速的光线跟踪支持。X^e HPG 为英特尔锐炫™ 显卡提供了出色的性能、能效和可扩展性。

技术支持：

- **DirectX* 12 Ultimate：**所有英特尔锐炫™ 显卡全面支持 DirectX* 12 Ultimate，通过支持全新的图形技术，包括光线追踪、可变速率着色、网格着色和采样器反馈等次世代游戏的基本要素，将游戏的逼真度提高到一个新的水平；
- **X^e 超级采样（X^{e}SS）：**英特尔® X^e 超级采样（X^{e}SS）使用人工智能强化升级提高性能和图像保真度，提升玩家的游戏体验。X^{e}SS 已针对英特尔锐炫™ 显卡产品进行了优化，能够利用 XMX AI 硬件加速技术。英特尔今年新发布了 X^{e}SS 1.1 版本，相比 X^{e}SS 1.0 进一步提高了图像质量和时间稳定性，并添加了自动曝光功能支持以提高视觉质量；
- **英特尔® XMX AI 加速引擎：**英特尔® XMX AI 加速引擎可为加速 AI 工作负载提供更强大的计算能力。与传统的 GPU 矢量单元相比，XMX AI 引擎能够为 AI 推理提供强大的算力，这也将有助于提升生产力、游戏和内容创作所需的性能；
- **X^e 媒体引擎：**X^e 媒体引擎可以为众多视频编解码器提供加速并支持广泛的行业标准，它还支持业界领先的 AV1 硬件加速编解码；
- **Deep Link：**英特尔锐炫™ 显卡支持英特尔® Deep Link 技术，该技术使锐炫独立显卡能够与英特尔 CPU 和集成显卡无缝配合，提升游戏、创作和直播的体验；
- **oneAPI 跨架构工具：**英特尔锐炫™ 显卡提供领先的 oneAPI 跨架构工具，能够帮助游戏开发人员和内容创作者释放英特尔 CPU 和锐炫 GPU 的独特优势，并优化应用程序，更大程度提升视觉性能。

英特尔锐炫™ Pro A 系列显卡产品

- 英特尔锐炫™ Pro A 系列旨在面向领先的专业软件应用认证，包括建筑、工程和施工行业以及设计和制造行业。

英特尔® 高级矢量扩展 512（英特尔® AVX-512）

快速分析日益增多的数据，并将其转化为有价值的洞察力，这种能力将为商业、科学研究乃至人们的日常生活创造新的机遇。英特尔® 至强® 可扩展处理器和英特尔® 至强融核™ 处理器产品家族，增添了旨在加速数据分析的创新功能。

当前的工作负载，通常需要在多个数据元素上执行同样的操作，在传统的“标量处理”时代，指令在同一时间，只能在一个单一数据元素上执行，以致在处理海量数据时极为耗时。认识到标量处理的不足之后，从上世纪 90 年代后期开始，英特尔开始将单指令多数据流（Single Instruction Multiple Data，SIMD）矢量功能整合到英特尔® 处理器中。英特尔® SSE 技术刚推出时，提供了 128 位寄存器和 SIMD 指令，可同时处理多达 4 个 32 位数据元素，大大加快了相关操作的处理速度。在此之后，英特尔® AVX 指令集和英特尔® AVX2 指令集又将寄存器宽度扩展了一倍，使相关操作的处理性能实现近乎翻倍的提升。

如今，英特尔® AVX – 512 指令集将矢量计算性能提升至新高度，寄存器的宽度和数量又在英特尔® AVX 指令集和英特尔® AVX2 指令集的基础上扩展了一倍，寄存器已由最初的 64 位升级到了 512 位，且具备两个 512 位的 FMA 单元，这意味着应用程序可同时执行 32 次双精度、64 次单精度浮点运算，或操作八个 64 位和十六个 32 位整数。

英特尔® 至强® 可扩展处理器可支持多种工作负载，包括复杂的 AI 工作负载。英特尔® 至强® 可扩展处理器通过英特尔® 深度学习加速（英特尔® DL Boost）进一步提升了 AI 计算性能。英特尔® 深度学习加速（英特尔® DL Boost）技术的精髓，就是把对低精度数据格式的操作指令融入到了 AVX - 512 指令集中，即 AVX-512_VNNI（矢量神经网络指令）和 AVX-512_BF16（bfloat16），分别提供了对 INT8（主打推理）和 BF16（兼顾推理和训练）的支持。

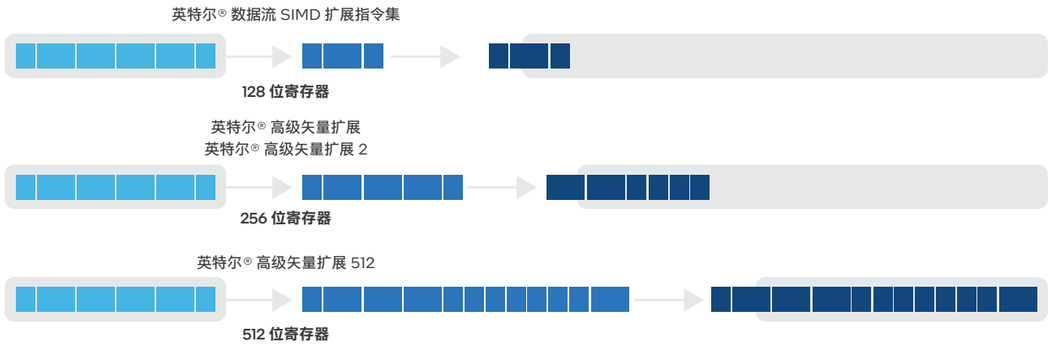


图 6-1 英特尔® SSE、英特尔 AVX2 和英特尔® AVX-512 之间的寄存器大小和计算效率的差异说明

英特尔® 高级矩阵扩展（英特尔® AMX）

第四代英特尔® 至强® 可扩展处理器内置 AI 加速器——英特尔® AMX，是企业和机构优化 AI 流水线的理想选择。平衡推理是 CPU 在 AI 应用中的主要用例，英特尔® AMX 专为该用例设计并且具备更多训练能力。目前，在所有运行 AI 推理工作负载的已装机数据中心处理单元中，英特尔® 至强® 可扩展处理器的占比高达 70%[25]；因此，为新的 AI 部署选择内置英特尔® AMX 的第四代英特尔® 至强® 可扩展处理器，是一种既高效又具有成本效益的 AI 工作负载加速方式。

英特尔® AMX 是什么？

英特尔® AMX 是内置于第四代英特尔® 至强® 可扩展处理器中的加速器，可优化深度学习（DL）训练和推理工作负载。借助英特尔® AMX，第四代英特尔® 至强® 可扩展处理器可在优化通用计算和 AI 工作负载间快速转换。开发人员可以编写非 AI 功能代码来利用处理器的指令集架构（ISA），也可编写 AI 功能代码，以充分发挥英特尔® AMX 指令集的优势。英特尔已将其 oneAPI DL 引擎——英特尔® oneAPI 深度神经网络库（Intel® oneAPI Deep Neural Network Library，英特尔® oneDNN）集成至包括 TensorFlow、PyTorch、PaddlePaddle 和 ONNX 在内的多个主流 AI 应用开源工具当中。

英特尔® AMX 架构

英特尔® AMX 架构由两部分组件构成：

- 第一部分为 TILE，由 8 个 1 KB 大小的 2D 寄存器组成，可存储大数据块；
- 第二部分为平铺矩阵乘法（TMUL），它是与 TILE 连接的加速引擎，可执行用于 AI 的矩阵乘法计算。

图 6-2 英特尔® AMX 架构由 2D 寄存器文件（TILE）和 TMUL 组成

英特尔® AMX 支持两种数据类型：INT8 和 BF16，两者均可用于 AI 工作负载所需的矩阵乘法运算。

- 当推理无需 FP32（AI 经常使用的单精度浮点格式）的精度时可使用 INT8 这种数据类型。由于该数据类型的精度较低，因此单位计算周期内运算次数就更多；
- BF16 这种数据类型实现的准确度足以达到大多数训练的要求，必要时它也能让 AI 推理实现更高的准确度。

凭借这种新的平铺架构，英特尔® AMX 实现了大幅代际性能提升。与运行英特尔® AVX - 512_ VNNI 的第三代英特尔® 至强® 可扩展处理器相比，运行英特尔® AMX 的第四代英特尔 ® 至强® 可扩展处理器将单位计算周期内执行 INT8 运算的次数从 256 次提高至 2,048 次。此外，如图 6-3 所示，第四代英特尔® 至强® 可扩展处理器可在单位计算周期内执行 1,024 次 BF16 运算，而第三代英特尔® 至强® 可扩展处理器执行 FP32 运算的次数仅为 64 次[26]。

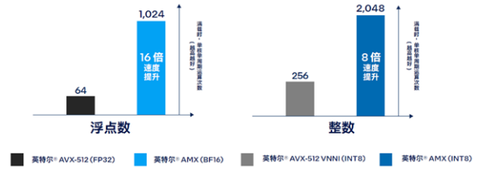


图 6-3 与英特尔® AVX - 512 _VNNI 相比，英特尔® AMX 在处理 INT8 和 BF16 两种数据类型时表现更出色[27]

使用英特尔® AMX 立启新加速

借助英特尔® AMX，几乎无需费力，即可提升性能。这得益于多个默认使用的框架都经过英特尔® oneDNN 的优化。Windows 和 Linux 操作系统、基于内核的虚拟机（KVM）和多个主流虚拟机管理程序都支持英特尔® AMX 指令集。INT8 和 BF16 运算在 TensorFlow 和 PyTorch 等开源框架内可自动优化。开发人员可借助英特尔® 分发版 OpenVINO™ 工具包（Intel® Distribution of OpenVINO™ Toolkit）实现 AI 推理的自动化、优化、微调和运行，且几乎或者完全不需要具备编码知识。而且，开发人员只需使用英特尔® Neural Compressor 便可将训练模型量化为 INT8 数据类型。

访问链接了解更多英特尔® AMX 的详情
https://www.intel.cn/content/www/cn/zh/customer-spotlight/cases/accelerate-ai-workload-with-amx.html

扫码了解更多英特尔® AMX 详情

[25] 基于英特尔对截至 2021 年 12 月运行 AI 推理工作负载的全球数据中心服务器装机容量的市场建模。
[26、27] https://www.intel.cn/content/www/cn/zh/customer-spotlight/cases/accelerate-ai-workload-with-amx.html

英特尔® 软件防护扩展（英特尔® SGX）

通过数据协同，引入更多源、多维、高质量的数据来打破数据孤岛，已成为各行各业深入开展大数据和 AI 应用，充分挖掘数据价值，进而加速推进数字化和智能化转型进程的共识。如今，对存储和传输状态下的数据进行加密处理已是行业的标准做法。然而，企业在数据保护方面的薄弱之处却是数据在处理器或内存中处于使用状态时。在这种情况下，个人可识别信息、电子病历和金融交易等敏感数据存在较高的被利用风险、很容易发生意外暴露或违反合规要求。

英特尔® 至强® 可扩展处理器的内置安全技术为各种数据（包括敏感、保密和处于监管之下的数据）保驾护航，使其可用于分析，进而帮助企业加速创新步伐。英特尔® SGX 是英特尔的独有技术，能够从硬件层面帮助保护使用中的数据。使用英特尔® 至强® 可扩展处理器的企业不必从数据分析和 AI 模型中剔除敏感数据，而是可通过英特尔® SGX 创建访问受限的数据安全“飞地”。这样的隔离环境可帮助企业在更好地保护敏感数据始终处于保密状态的前提下，充分发挥其价值。

英特尔® SGX 作为先进的安全机制，可与现有基础设施一同使用，更好地保护敏感型工作负载或服务。通过使用英特尔® SGX，应用可以把代码和数据隔离在安全“飞地”中加以保护。至强® 处理器在高达 TB 级的内存空间中管理这些飞地。经过配置后，同一系统，甚至同一 CPU 内核上运行的其他进程将无法访问该飞地中的数据和代码，即便是具有“根”访问权限的进程也是如此。此外，英特尔® SGX 还解决了可信远程计算的一个基本问题：如果数据所有者想要通过某个进程来处理数据，但不能或者不想直接控制该进程，就只能信任并依赖该进程的所有者。而在英特尔® SGX 中，远程认证服务器会使用哈希值来验证“飞地”中的代码与开发人员发布的原始代码是否匹配，并且能够检测并阻止在飞地中植入操纵代码的企图。

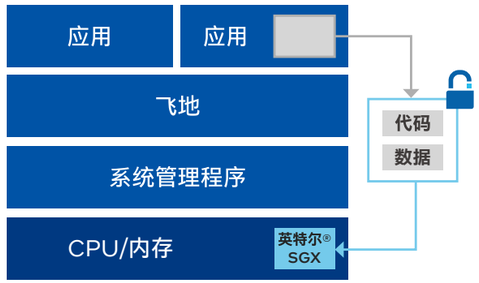


图 6-4 英特尔® SGX 通过将敏感数据隔离在容量高达 1TB 的飞地中，帮助保护敏感数据

英特尔® SGX 经过广泛部署和研究，是数据中心可信执行环境（TEE）的重要技术实现，能够大幅减少系统内的攻击面，提供基于硬件的安全解决方案，通过专用应用隔离技术帮助保护使用中的数据。开发人员可以通过保护选定的代码和数据不被查看或修改，在“飞地”内执行涉及敏感数据的操作，帮助提高应用的安全性和保护数据的机密性。由英特尔® SGX 提供支持的机密计算可实现应用层面、虚拟机（VM）、容器和功能层面的数据隔离。无论是在云端、边缘还是本地环境，客户都能确保自身的计算与数据始终获得私密性和安全性更高的保护，不会暴露给云服务提供商、未经授权的管理员和操作系统，甚至是特权应用。

访问链接了解更多英特尔® SGX 的详情
https://www.intel.cn/content/dam/www/central-libraries/cn/zh/documents/2023-02/xeon-accelerated-security-product-brief-v3-q123-chinesesimplified.pdf

扫码了解更多英特尔® SGX 详情

英特尔® SST

作为目前企业级计算 CPU 代表选手的英特尔® 至强® 可扩展处理器产品家族，除了对微架构、核心数量与性能、内存通道数量和速度，以及 I/O 性能等方面持续改良以巩固优势外，它在过去数年来，还应日趋复杂的应用需求，为用户提供了越来越多面向特定场景和负载的创新技术特性。

英特尔® SST（Intel® Speed Select Technology）能为企业多样化、差异化应用需求提供更优支持，其独特之处在于能让 CPU 根据不同应用场景或应用负载的特点及其对算力的特定要求，对处理器单个及多个核心的运行状态、频率和功耗进行精细化控制，从而能在保障更优能效的前提下满足不同负载的差异化需求。

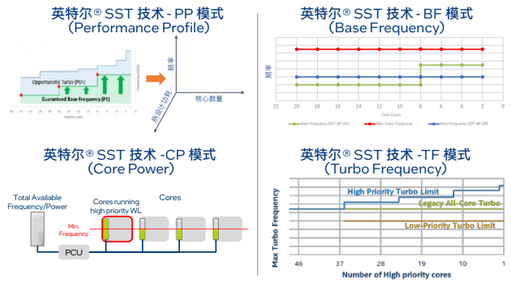


图 6-5 英特尔® SST 现有的四种模式

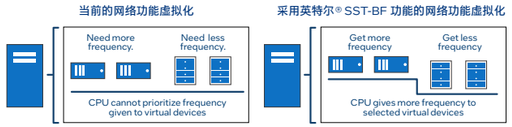
英特尔® SST- 性能配置文件（PP）：通过配置 CPU 来适应不断变化的工作负载

英特尔® SST-PP 是英特尔® SST 家族的首款出色功能。客户可以配置一台（而不是多台）服务器来满足不断变化的工作负载处理需求，从而提高服务器利用率，降低鉴定成本。只要有一台灵活的服务器和多种配置，优化的总拥有成本是水到渠成的结果。

采用英特尔® SST-PP 的 CSP 基础设施。一台服务器、多种配置

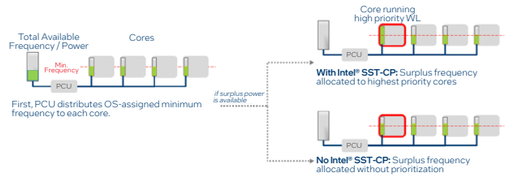
英特尔® SST- 基频（BF）：随时随地发挥强劲动力

使用英特尔® SST-BF 功能，可以控制和分配基频，从而在关键时刻为关键工作负载提供强劲动力。如果可以对性能进行精细分配，就可以提高整体性能。



英特尔® SST-CP 模式

可对多个核心进行分组，并通过赋予各组不同的频率状态，来应对不同优先级的负载的需求。在处理器负载非常高时，SST-CP 模式会对执行低优先级任务的核心组进行降频，以确保执行高优先级任务的核心组不受影响。



访问链接了解更多英特尔® SST 的详情
https://networkbuilders.intel.com/solutionslibrary/intel-speed-select-technology-intel-sst-performance-enhancements-for-3rd-gen-intel-xeon-scalable-processor-technology-guide

扫码了解更多英特尔® SST 详情

英特尔® oneAPI 工具套件

英特尔® oneAPI 工具套件是基于新一代标准的英特尔软件开发工具，用于跨各种架构构建和部署以数据为中心的高性能应用程序。它能够通过充分利用一流的硬件特性加速计算进程，并全面兼容现有的编程模型和代码库，可确保开发者已经编写的应用能够在 oneAPI 上无缝运行。此外，开发者只需一个代码库，便可以将应用轻松迁移到新系统和加速器上，大幅缩短了迁移时间，减轻了迁移工作量。

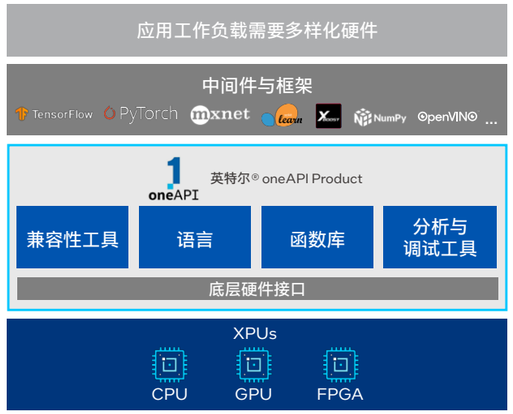


图 6-6 英特尔® oneAPI 工具套件架构

通过英特尔® oneAPI 工具套件，开发者能够使用一种通用、开放且基于行业标准的编程模型访问英特尔® CPU/GPU/FPGA。这不仅能够释放底层硬件的性能潜力，同时能降低软件开发和维护成本，并且在部署加速计算方面，英特尔® oneAPI 工具套件与专用的、受限于特定厂商的方案相比风险更低。

英特尔® oneAPI 工具套件充分利用了先进的硬件性能和指令，如用于 CPU 的英特尔® AVX-512 和英特尔® DL Boost，以及 XPU 独有的功能。英特尔® oneAPI 工具套件基于经受过长久考验的英特尔开发者工具，为开发者提供熟悉的编程语言和标准，同时与现有代码保持完全的连续性，其包括英特尔® oneAPI Base 工具包、英特尔® oneAPI AI Analytics 工具包、英特尔® oneAPI HPC 工具包及 OpenVINO™ 工具套件等不同工具。

英特尔® oneAPI AI Analytics 工具包

英特尔® oneAPI AI Analytics 工具包为数据科学家、人工智能开发人员和研究人员提供了熟悉的 Python 工具和框架，利用面向英特尔® 架构优化的库加速端到端人工智能和数据分析流水线，实现基于面向英特尔® 架构优化的深度学习框架和工具提升训练和推理性能，并使用计算密集型 Python 包为数据分析和机器学习工作流提供落地加速。

访问链接了解更多英特尔® oneAPI 工具套件详情
https://www.intel.com/content/www/us/en/developer/tools/oneapi/toolkits.html

扫码了解更多英特尔® oneAPI 工具套件详情

英特尔® 数据分析加速库（oneDAL）

英特尔为行业用户部署机器学习，也推出了一套高性能系统化方案，涵盖处理器、经优化的软件和开发人员支持，以及强大的生态系统等丰富资源。

机器学习需要强劲的计算能力。英特尔® 至强® 处理器提供了一个可扩展的基准，专门用于满足机器学习所特有的高度并行工作负载，及其对内存和架构（网络）的需求。

此外，英特尔还提供了软件支持。英特尔® oneAPI 数据分析库（oneDAL）是一套旨在帮助数据科学家和分析师们快速建立从数据预处理，到数据特征工程、数据建模和部署的端到端软件方案。它提供了建立机器学习和分析所需的各种数据分析及算法所需的高性能构建模块。目前已经支持线性回归、逻辑回归、LASSO、AdaBoost、贝叶斯分类器、支撑向量机、K 近邻、Kmeans 聚类、DBSCAN 聚类、各种决策树、随机森林、Gradient Boosting 等经典机器学习算法。这些算法经过高度优化，可在英特尔® 处理器上实现高性能。

为了开发人员在基于英特尔环境中的机器学习应用中更加方便地使用英特尔® 数据分析库 oneDAL，英特尔开源了整个项目：https://github.com/oneapi-src/onedal，并针对不同的大数据使用场景，提供全内存的、流式的和分布式的算法支持。比如 oneDAL Kmeans 可以很好地和 Spark 结合，在 Spark 集群上进行多节点聚类。另外，英特尔® oneDAL 提供了 C++ 和 Python 接口。

访问链接了解更多英特尔® oneDAL 详情
https://www.intel.com/content/www/us/en/developer/tools/oneapi/onedal.html

扫码了解更多英特尔® oneDAL 详情

英特尔® oneAPI 数学内核库（oneMKL）

oneMKL 是高度优化、快速、完整的数学函数库，常用于科学、工程和金融应用。oneMKL 定义了一套用于科学计算和其他应用的基本数学程序。作为 oneAPI 的一部分，oneMKL 允许在包括 CPU、GPU、FPGA 和其他加速器等各种计算设备上运行。

oneMKL 能够加快数学处理程序，提高应用性能，并减少开发时间，其具备如下特点：

- 增强的数学程序使开发人员和数据科学家能够创建高性能的科学、工程或金融应用程序；
- 核心功能包括 BLAS、LAPACK、稀疏求解器、快速傅里叶变换（FFT）、随机数生成器功能（RNG）、汇总统计、数据拟合和矢量数学；
- 针对下一代 CPU 和 GPU 进行了额外矩阵乘法优化；同时，增加了 CUDA 库函数 API 对 BLAS、LAPACK、稀疏 BLAS、向量数学、汇总统计、样条等的兼容覆盖，简化了代码向 oneAPI 和英特尔® GPU 的迁移。
- 支持第四代英特尔® 至强® 可扩展处理器的英特尔® AMX bfloat16 数据类型和英特尔® AVX-512 bfloat16 数据类型。
- 对于以前的英特尔® 数学内核库（Intel® MKL）用户来说，是一种无缝升级。

访问链接了解更多英特尔® oneMKL 详情
https://www.intel.cn/content/www/cn/zh/developer/tools/oneapi/onemkl.html

扫码了解更多英特尔® oneMKL 详情

英特尔® 深度神经网络库（oneDNN）

英特尔® oneAPI 深度神经网络库（oneDNN）是一款面向深度学习应用的开源跨平台性能增强库，也是英特尔为了帮助开发人员充分利用英特尔® 架构，推进深度学习的研究和应用而创建的基础库。

作为开源跨平台性能库，oneDNN 针对英特尔® 架构处理器、英特尔显卡和基于 ARM 64 位架构的处理器进行了优化。oneDNN 包含了高度矢量化和线程化的构建模块，支持利用 C 和 C++ 接口实现深度神经网络，具备广泛的深度学习研究、开发和应用生态系统。目前已支持 TensorFlow、 PyTorch、MXNet、PaddlePaddle、BigDL、OpenVINO™ 工具套件等丰富的深度学习软件产品。

为了有效提升深度学习模型在基于英特尔® 架构的基础设施上的运行速度，以及提升各类神经网络中其他性能敏感型应用的效率，oneDNN 提供了众多优化的深度学习运行和操作基元，可应用于不同的深度学习框架，以确保通用构建模块的高效实施。

oneDNN 目前已成为众多深度学习框架在 CPU 上运行时的基本配置，开发者可在深度学习框架的安装和应用中，直接获取 oneDNN 带来的性能提升。

访问链接了解更多英特尔® oneDNN 工具套件详情
主页：https://www.intel.com/content/www/us/en/developer/tools/oneapi/onednn.html
代码：https://github.com/oneapi-src/oneDNN

扫码了解更多英特尔® oneDNN 详情

面向英特尔® 架构优化的 TensorFlow 扩展包（ITEX）

面向英特尔® 架构优化的 TensorFlow 扩展包（Intel® Extension for TensorFlow，ITEX），是基于 TensorFlow Pluggable Device 接口的异构高性能深度学习扩展插件，可将英特尔® XPU（GPU、CPU 等）设备引入 TensorFlow 开放 AI 工作负载加速的开源社区。为了显著提升性能，英特尔持续采用多种措施对 TensorFlow 进行优化：

- **运算符优化**：针对 CPU 中的运算符进行优化，并通过英特尔® oneAPI DPC++ 编译器实现所有 GPU 运算符优化。用户无需任何额外设置即可默认获得这些算子优化收益，此外，还开发了多个用于使用 itex.ops 命名空间提升性能的定制运算符，以扩展 TensorFlow 公共 API 实现以获得更好的性能。
- **计算图优化**：将指定的运算模式融合到新的单一运算中，以获得更好的性能，例如 Conv2D+ReLU、Linear+ReLU 等。混合精度使用较低精度的数据类型（FP16 或 BF16），使模型在训练和推理过程中运行速度更快，内存消耗更少。
- **低精度优化**：TensorFlow 开发者可以通过低精度数据类型 Bfloat16 为训练与推理模型加速，获得至多 2 倍的性能提升且保持模型精度不变。ITEX 提供了包括手动、自动等多种 Bfloat16 模型转换解决方案，用户可以根据实际情况选择不同的方案来满足对于性能或使用体验的要求。

访问链接了解更多 TensorFlow 扩展包详情
主页：https://www.intel.com/content/www/us/en/developer/tools/oneapi/optimization-for-tensorflow.html
性能数据：https://github.com/intel/intel-extension-for-tensorflow

扫码了解更多 TensorFlow 扩展包详情

面向英特尔® 架构优化的 PyTorch 扩展包（IPEX）

PyTorch 是一款开源的深度学习库，旨在帮助数据科学家和开发人员提高深度学习的训练和推理效率，具备高度的灵活性、易用性和训练、推理高速度，也深受业界青睐。PyTorch 依托简洁、灵活的结构，提供了多项优异特性。

为了提升 PyTorch 在英特尔硬件上的性能，英特尔推出了面向英特尔® 架构优化的 PyTorch 扩展包（Intel® Extension for PyTorch, IPEX）。IPEX 在原生 PyTorch 的基础上，添加面向深度学习优化的扩展能力，同时让用户在英特尔® 架构上使用 PyTorch 时，获得“开箱即用”式的优化体验。面向英特尔® 架构优化的 PyTorch 扩展包会根据基于英特尔® 架构的硬件特性的变化而不断升级，随着第三代和第四代英特尔® 至强® 可扩展处理器的全面铺开，新版扩展包也针对新平台的各项特性进行了专门优化。

IPEX 提供了针对 eager 模式和 graph 模式的优化。在 eager 模式下，PyTorch 前端通过自定义 Python 模块（例如融合模块）、最优优化器和 INT8 量化 API 进行扩展。通过扩展图融合通道将 eager 模式模型转换为 graph 模式，可以进一步提升性能。在 graph 模式下，融合减少了运算符 / 内核调用开销，从而提高了性能。在 CPU 上，IPEX 根据 ISA（指令集架构）自动将运算符分派到其底层内核中，ISA 检测并利用英特尔硬件上可用的矢量化和矩阵加速单元。自动混合 Float32 和 BFloat16 之间的运算符数据类型精度，以减少计算工作量和模型大小。IPEX 运行时扩展通过更细粒度的线程运行时控制和权重共享带来更高的效率。在 GPU 上，优化的算子和内核通过 PyTorch 调度机制实现和注册，这些运算符和内核通过英特尔 GPU 硬件的矢量化和矩阵计算功能进行加速。

访问链接了解更多英特尔® PyTorch 扩展包详情
主页：https://www.intel.com/content/www/us/en/developer/tools/oneapi/optimization-for-pytorch.html
代码：https://github.com/intel/intel-extension-for-pytorch

扫码了解更多英特尔® PyTorch 扩展包详情

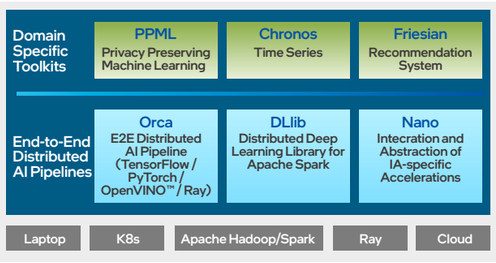
开源的、统一的大数据分析 +AI 平台 BigDL

BigDL 是英特尔开源的统一的大数据和人工智能平台，BigDL 可以将用户的数据分析或者 AI 应用无缝地从笔记本扩展到集群和云端。

BigDL 的特性包括：

- Orca：在 Spark 和 Ray 上构建分布式的大数据和 AI（PyTorch / TensorFlow）流水线
- Nano：在 XPU 上对 PyTorch / TensorFlow 应用进行透明加速
- Chronos：可扩展的自动时间序列数据分析应用
- Friesian：构建端到端推荐系统
- PPML：在 SGX/TDX 上构建更加安全的大数据和 AI 应用

此外，BigDL 还发布了大语言模型（LLM）的库，可以在英特尔平台上进行大语言模型的高效推理。



访问链接了解更多 BigDL 详情
https://www.intel.cn/content/www/cn/zh/developer/tools/bigdl/overview.html

扫码了解更多 BigDL 详情

英特尔® Deep Learning Streamer

英特尔® Deep Learning Streamer（英特尔® DL Streamer）是一个开源的流媒体分析框架，基于 GStreamer 多媒体框架，用于创建云端或边缘的复杂媒体分析管道。

英特尔® DL Streamer 让媒体分析变得更简单：

- 编写更少代码，获得更佳性能；
- 在云端和边缘快速开发、优化、基准测试和部署视频与音频分析流水线；
- 分析视频和音频流、创建可操作的结果，获取并将结果发送到云端；
- 充分利用基于英特尔® 架构的硬件平台的效率和计算能力；
- 在更多英特尔® 架构平台，不同代或不同种类的 CPU 或 XPU 上，都能编写可移植的代码。
- 充分发挥 GStreamer 开源项目社区在新特性和 bug 修复方面的投资能力；
- 借助管线互操作性，发挥 NNStreamer 在特定 AI 推理要素上的影响力；
- 通过审查、分析和修改英特尔® DL Streamer 开源代码，定制和扩展针对客户需求的解决方案。

访问链接了解更多英特尔® DL Streamer 详情
https://dlstreamer.github.io

扫码了解更多
英特尔® DL Streamer 详情

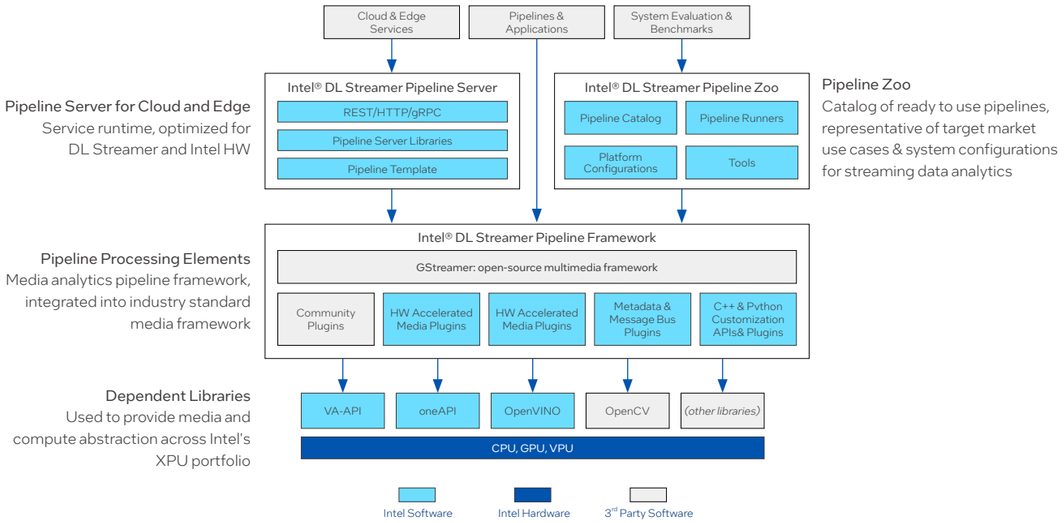


图 6-7 英特尔® DL Streamer 软件栈

OpenVINO™ 工具套件

OpenVINO™ 工具套件是英特尔推出的一款加速深度学习推理及部署的软件工具套件，用以加快高性能计算机视觉处理和应用。该工具允许异构执行，支持 Windows 与 Linux 系统，以及 Python/C++ 语言，能够有效推进计算机视觉技术在从智能摄像头、视频监控、机器人，到智能交通、智能医疗等领域的深入应用。

本工具套件提高了计算机视觉解决方案的性能，缩短了开发时间，简化了从英特尔提供的丰富硬件选项中获得效益的途径，而这些选项可以提高性能、降低功耗并最大化硬件利用率——让用户可以低资源获得高收益，并为新的产品设计提供个性化空间。

通过基于深度卷积神经网络（CNN），扩展英特尔硬件（包括加速器）的工作负载，使得 OpenVINO™ 工具套件可依托基于英特尔® 架构的 CPU 和 GPU 来增强视觉系统的功能和性能。最新发布的 OpenVINO™ 版本已能支持第四代英特尔® 至强® 可扩展处理器，并通过英特尔® AMX、英特尔® AVX-512 以及采用 VNNI 的英特尔® DL Boost 技术来提升推理性能，可帮助客户在不改变软件的基础上，快速完成硬件产品升级和算法移植，从而助其在边缘侧快速实现高性能计算机视觉与深度学习应用的开发：

- 释放 CNN-based 的网络在边缘设备的性能瓶颈
- 基于通用 API 接口在基于英特尔 ® 架构的 CPU 和 GPU 上运行

基于英特尔平台优化的 OpenVINO™ 工具套件主要包括模型优化器（Model Optimizer）和推理引擎（Inference Engine）两个核心组件：

■ **模型优化器：**

- 模型优化器是一种跨平台命令行工具，它可将训练后的网络模型从其源框架转换为开源、且与 nGraph 兼容的 IR（中间表示）文件以用于推理操作，IR 文件是由 bin（经训练的数据文件）和 xml（描述网络拓扑的文件）两种格式文件组成。一方面，模型优化器可以在导入由 Caffe、TensorFlow、MXNet、PyTorch、Keras 以及 ONNX 等流行框架训练好的模型后，执行相应优化，包括去除多余的层，并在可能的情况下将操作分组为更简单、更快的图（Graph）等。
- 另一方面，OpenVINO™ 工具套件也可以进行模型量化过程。PyTorch 等流行框架中训练的模型通常为 FP32 精度数据格式，而英特尔® 至强® 可扩展处理器等计算平台已经支持 INT8 等低精度数据格式下的模型推理，其可在损失很小精度的前提下实现更高的推理效率。量化过程是将基于高精度数据格式的模型（由 IR 文件表示）转为低精度，并加入校准过程以保证精度不受损失。

■ **推理引擎：**

- 支持硬件指令集层面的深度学习模型加速运行，支持的硬件设备主要包括：英特尔® CPU 和 GPU。实现用户一次开发，即可面向不同平台部署并获得一致的性能表现，有效提升 AI 开发与部署效率。

OpenVINO™ 工具套件在英特尔平台上让视觉成为现实，已帮助众多用户轻松开发和快速部署计算机视觉应用程序，在多种深度学习应用场景展示了人工智能解决方案所蕴藏的巨大潜力。

访问链接了解更多 OpenVINO™ 详情
主页：https://www.intel.cn/content/www/cn/zh/developer/tools/openvino-toolkit/overview.html
代码：https://github.com/openvinotoolkit/openvino
性能数据：https://docs.openvino.ai/latest/openvino_docs_performance_benchmarks.html

扫码了解更多
OpenVINO™ 详情

英文全称	英文缩写	中文全称
Advanced Video Coding	AVC	高级视频编码
Approximate Binary Map		二值化特征图
Automated Guided Vehicle	AGV	自动导引车
Automatic Mixed Precision	AMP	自动混合精度
Bag-of-Freebies	BoF	
Bag-of-Specials	BoS	
Bounding box		边界框
Built-in Models		内置模型
Column Subsampling		列抽样
Command Line Interface	CLI	命令行
Compound Annual Growth Rate	CAGR	年复合增长率
Computer Vision	CV	计算机视觉
Connectionist Temporal Classification	CTC	
Connectionist Text Proposal Network	CTPN	连接文本预选网络
Convolutional Neural Network	CNN	卷积神经网络
Convolutional Recurrent Neural Network	CRNN	卷积循环神经网络
Crop		裁剪
Cross-Iteration Batch Normalization	CBN	
Data Processing		数据处理
Deep Learning	DL	深度学习
Degradation		退化
Detectors		检测器
Differentiable Binarization		可微分二值化
Differentiable Binarization Network	DBNet	可微分二值化网络
Embedded Multi-die Interconnect Bridge	EMIB	嵌入式多芯片 互连桥接
Feature Engineering		特征工程
Feature Pyramid Networks	FPN	特征金字塔
Flat Memory Region		平面内存区域
Forecasters		功能涵盖预测器
Fully Convolutional	FC	全卷积
High Efficiency Video Coding	HEVC	高效率视频编码
Host to Device	H2D	

英文全称	英文缩写	中文全称
Hyperparameter optimization	HPO	
Industry Subversive Alliance	ISA	指令集架构
Inference Engine		推理引擎
Intel ARC ™ Graphics		英特尔锐炫™ 显卡
Intel® Advanced Matrix Extensions	Intel® AMX	英特尔® 高级矩阵扩展
Intel® Advanced Vector Extensions	Intel® AVX	英特尔® 高级矢量扩展
Intel® Advanced Vector Extensions 512	Intel® AVX - 512	英特尔® 高级矢量扩展 512
Intel® Crypto Acceleration		英特尔® 密码操作硬件加速
Intel® Data Streaming Accelerator	Intel® DSA	英特尔® 数据流加速器
Intel® Deep Learning Boost	Intel® DL Boost	英特尔® 深度学习加速
Intel® Deep Learning Streamer	Intel® DL Streamer	英特尔® DL Streamer
Intel® Distribution of OpenVINO™ Toolkit		英特尔® 分发版 OpenVINO™ 工具包
Intel® Dynamic Load Balancer	Intel® DLB	英特尔® 动态负载均衡器
Intel® Extension for PyTorch	IPEX	英特尔® PyTorch 扩展包
Intel® Extension for TensorFlow	ITEX	面向英特尔® 架构优化的 TensorFlow 扩展包
Intel® Infrastructure Processing Unit	Intel® IPU	英特尔® 基础设施处理单元
Intel® In-Memory Analytics Accelerator	Intel® IAA	英特尔® 存内分析加速器
Intel® oneAPI Data Analytics Library	Intel® oneDAL	英特尔® oneAPI 数据分析库
Intel® oneAPI Deep Neural Network Library	Intel® oneDNN	英特尔® oneAPI 深度神经网络库
Intel® oneAPI Math Kernel Library	Intel® oneMKL	英特尔® oneAPI 数学内核库
Intel® oneAPI Video Processing Library	Intel® oneVPL	英特尔® oneAPI 视频处理库
Intel® Quick Assist Technology	Intel® QAT	英特尔® 数据保护与压缩加速技术
Intel® Software Guard Extensions	Intel® SGX	英特尔® 软件防护扩展
Intel® Speed Select Technology	Intel® SST	英特尔® Speed Select 技术
Intel® Virtual RAID on CPU	Intel® VROC	英特尔® VROC
Intel® X^e Matrix Extensions	Intel® XMX	英特尔® XMX
Intermediate Representation	IR	中间数据格式
Kernel-based Virtual Machine	KVM	基于内核的虚拟机
launch script		启动脚本
Long Short-Term Memory	LSTM	长短期记忆网络
Model Optimizer	MO	模型优化器
Network Video Recorder	NVR	网络视频录像机

英文全称	英文缩写	中文全称
Non Uniform Memory Access	NUMA	非一致性内存访问
Object Detection		目标检测
One-stage		单步法
Operating System	OS	操作系统
Optical Character Recognition	OCR	光学字符识别
Path Aggregation Network	PAN	
Radio Frequency Identification	RFID	射频识别
Real Time Streaming Protocol	RTSP	实时流传输协议
Recurrent Neural Networks	RNN	递归神经网络
Recursive Pyramid Network	RPN	递归金字塔网络
Region of Interest	RoI	感兴趣区域
Residual Network	ResNet	残差网络
Return on Investment	ROI	投资回报率
Short-term-imageless tracking algorithms		短期无图像跟踪
Simulators		模拟器
Single Instruction Multiple Data	SIMD	单指令多数据流
Single Shot MultiBox Detector	SSD	
Socket		处理器插槽
Spatial Pyramid Pooling	SPP	
Spitial Attention Mask	SAM	
Support Vector Machine	SVM	支持向量机
Telemetry information		遥测信息
Total Cost of Ownership	TCO	总拥有成本
Two-stage		两步法
You Only Look Once	YOLO	
Zero-term tracking		零项跟踪
Zero-term-imageless		零项无图像

扫码访问英特尔官网
了解更多英特尔在物流与交通领域的技术实践

扫码下载
《英特尔中国物流与交通 AI 实战手册》

免责声明：

性能测试中使用的软件和工作负荷可能仅在英特尔微处理器上进行了性能优化。诸如 SYSmark 和 MobileMark 等测试均系基于特定计算机系统、硬件、软件、操作系统及功能。上述任何要素的变动都有可能导致测试结果的变化。请参考其他信息及性能测试（包括结合其他产品使用时的运行性能）以对目标产品进行全面评估。更多信息，详见 www.intel.com/benchmarks。

在特定系统的特殊测试中测试组件性能。硬件、软件或配置的差异将影响实际性能。当您考虑采购时，请查阅其他信息来源评估性能。关于性能和基准测试程序结果的更多信息，请访问 www.intel.com/benchmarks。

英特尔技术特性和优势取决于系统配置，并可能需要支持的硬件、软件或服务得以激活。产品性能会基于系统配置有所变化。没有任何产品或组件是绝对安全的。更多信息请从原始设备制造商或零售商处获得，或请见 intel.com。

优化声明：英特尔编译器针对英特尔微处理器的优化程度可能与针对非英特尔微处理器的优化程度不同。这些优化包括 SSE2、SSE3 和 SSSE3 指令集和其他优化。对于非英特尔微处理器上的任何优化是否存在、其功能或效力，英特尔不做任何保证。本产品中取决于微处理器的优化是针对英特尔微处理器。不具体针对英特尔微架构的特定优化为英特尔微处理器保留。请参考适用的产品用户与参考指南，获取有关本声明中具体指令集的更多信息。

没有任何产品或组件是绝对安全的。

描述的成本降低情景均旨在在特定情况和配置中举例说明特定英特尔产品如何影响未来成本并提供成本节约。情况均不同。英特尔不保证任何成本或成本降低。

英特尔并不控制或审计第三方数据。请您审查该内容，咨询其他来源，并确认提及数据是否准确。