LitePred: Transferable and Scalable Latency Prediction for Hardware-Aware Neural Architecture Search

NSDI 2024

汇报人：姚昌硕

2024年7月25日

作者团队

- 谈海生 中国科学技术大学
- 研究领域: 网络计算算法与系统；物联网；边缘计算；移动计算；AI+Edge；人工智能系统与网络

- Li Lyna Zhang
- Systems and Networking Groups of Microsoft Research Aisa
- (1) compression for pre-trained transformer models and LLMs, and (2) hardware-aware NAS for edge AI.

背景知识①：算子融合 Kernel

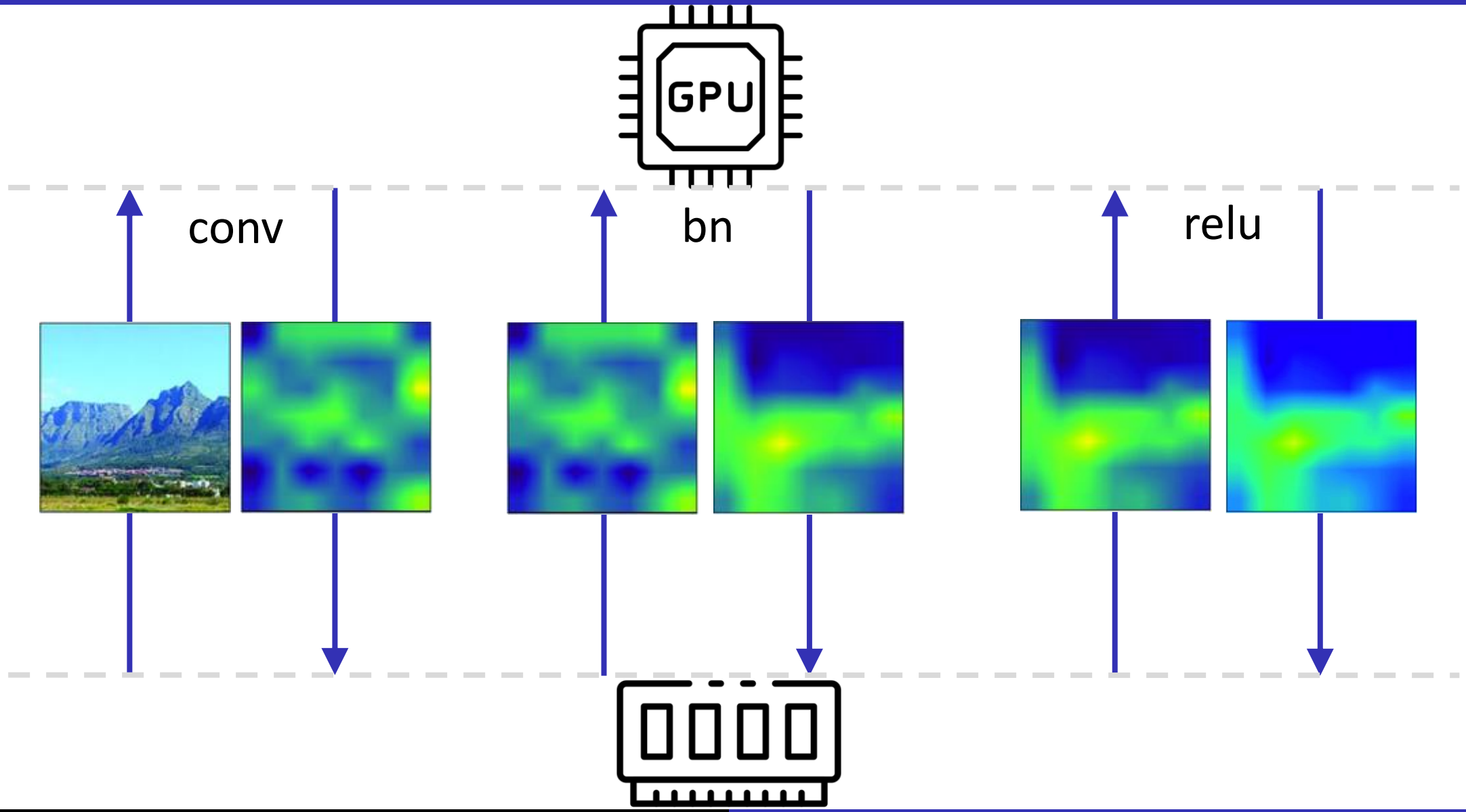
- 算子融合: 将多个算子组合成一个单一的算子来执行，以减少计算开销和内存访问。

- Kernel: 在GPU或其他加速硬件上执行的单个计算任务或函数

背景知识①：算子融合 Kernel (示例)

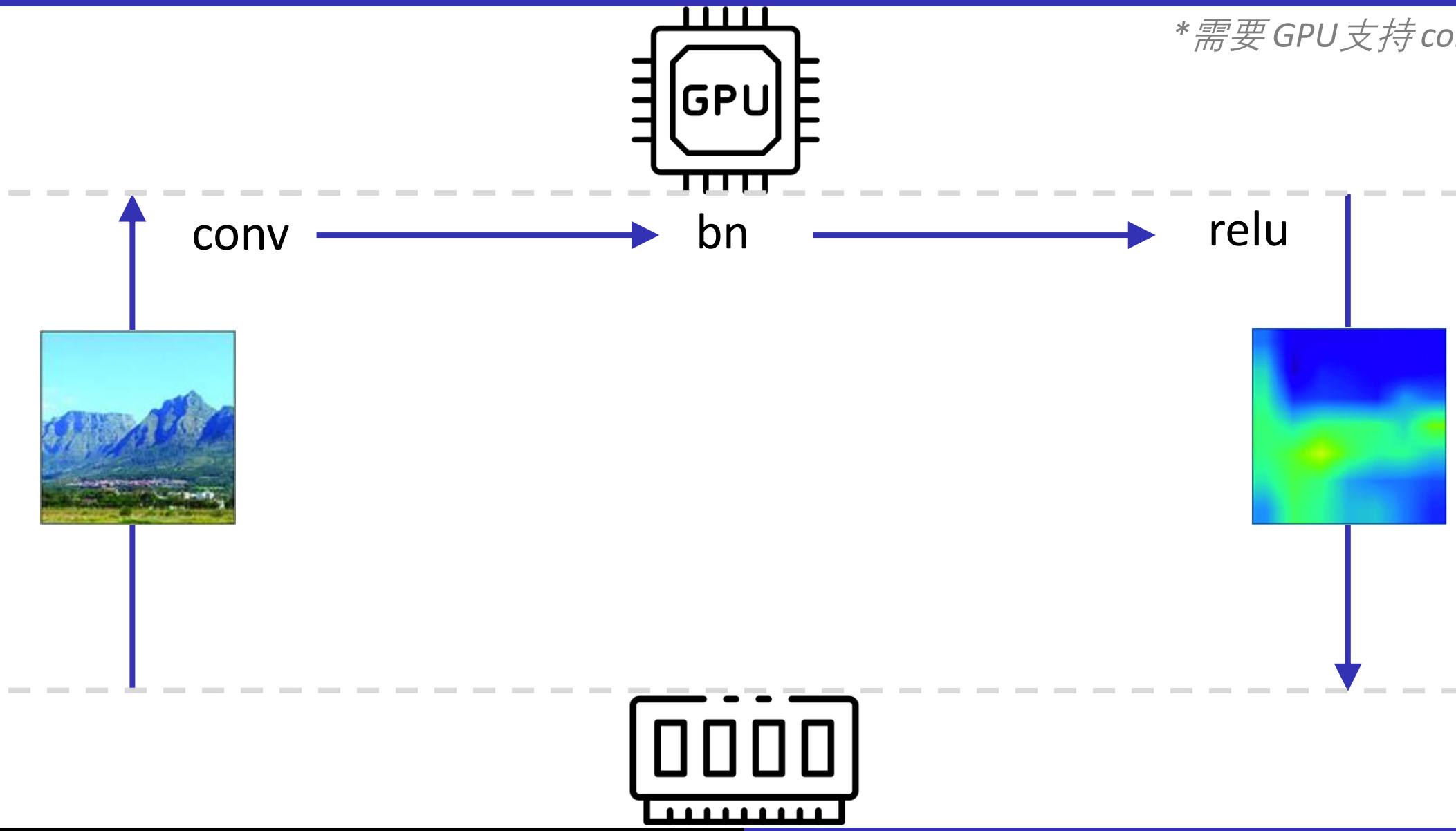
- 以Conv + BatchNorm + ReLU三个算子为例
- 融合前：
  - 从内存读原始图像→计算Conv →写入内存
  - 从内存读特征图+ 归一化参数 → 计算BatchNorm →写入内存
  - 从内存读归一化后的特征图 → 计算ReLU →写入内存
- 融合后：
  - 从内存读原始图像→计算Conv

背景知识①：算子融合 Kernel (示例)

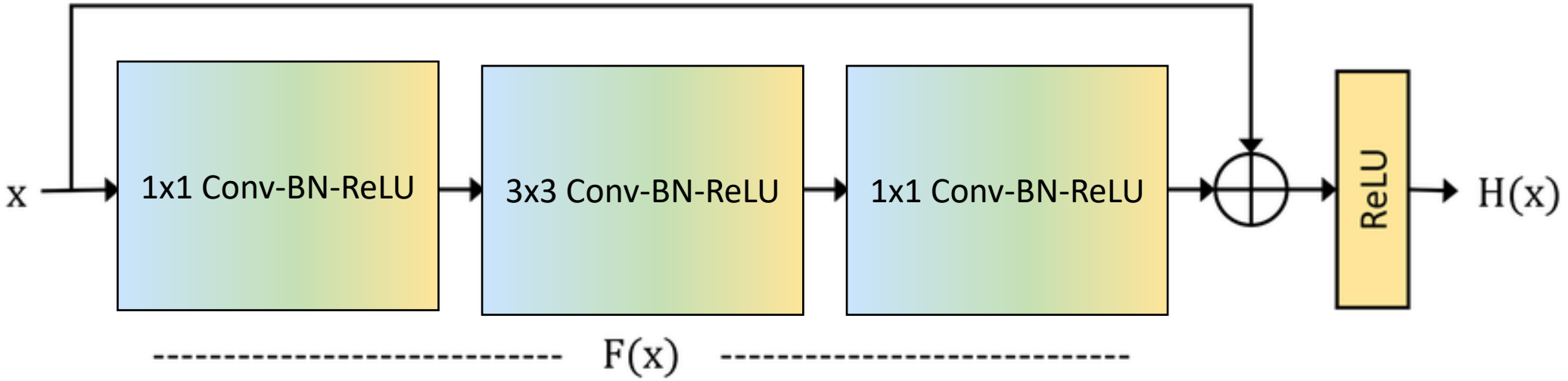


背景知识①：算子融合 Kernel (示例)

*需要GPU支持conv-bn-relu

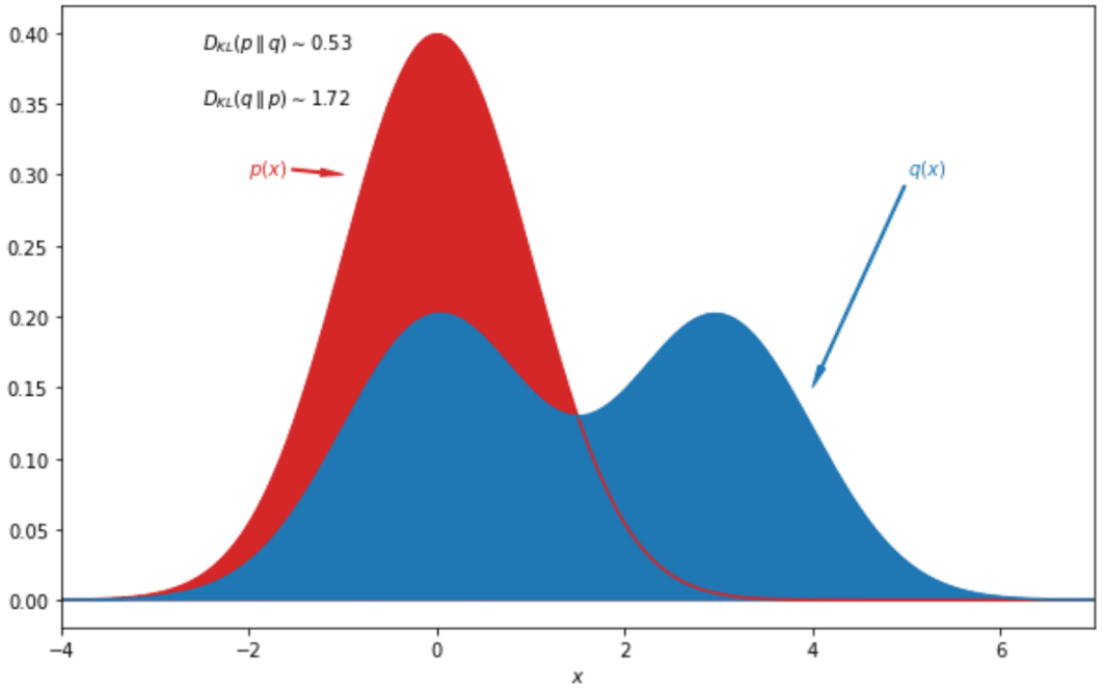


背景知识①：算子融合 Kernel (示例)



背景知识②: KL散度

- KL散度(Kullback-Leibler Divergence), 又称相对熵
- 衡量两个概率分布之间的差异
- 度量了使用 Q分布来近似P分布时，平均每次事件所“丢失”的信息量。
- 信息量的丢失反映了使用Q而不是 P 所带来的不准确性。



报告结构

- 研究背景
- 相关工作
- 问题分析与发现
- 系统结构与原理
- 实验
- 总结

研究背景

- DNN在边缘硬件上应用广泛
- 硬件感知神经架构搜索(NAS) Hardware-Aware Neural Architecture Search
- 延迟预测是NAS中重要步骤

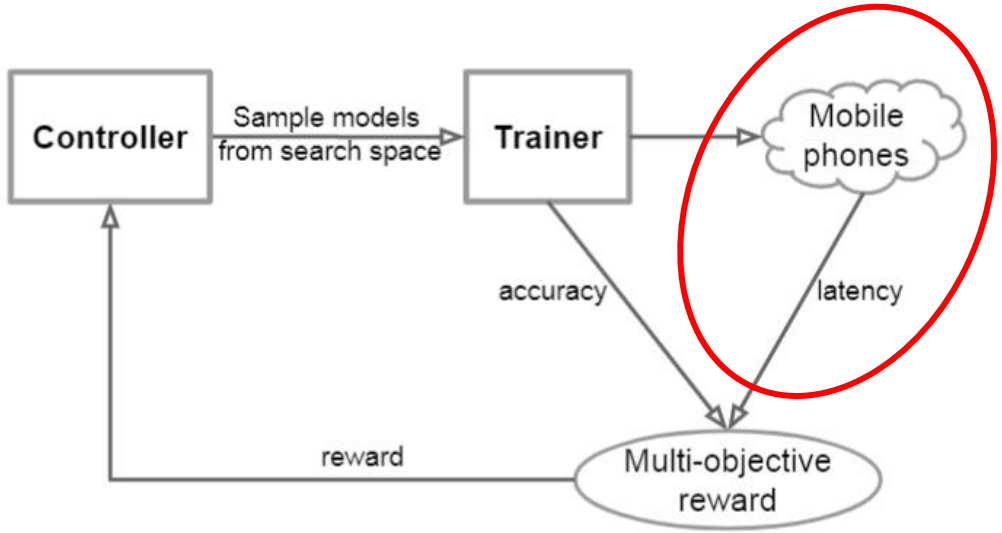


Figure 1: **An Overview of Platform-Aware Neural Architecture Search for Mobile.**

图源: MnasNet: Platform-Aware Neural Architecture Search for Mobile (CVPR2019)

研究背景

- NAS搜索空间：所有可能配置的集合
- e.g. Conv操作的配置可以构成 5维空间（HW, K, S, Cin, Cout）
- 在空间中采样：在所有可能出现的配置中选择一个配置

研究背景

- 实际设备测量：不实际 (搜索空间大，设备多样)
- 简单表征：FLOPs 不准确
  - for example, MobileNet [11] and NASNet [36] have similar FLOPS (575M vs. 564M), but their latencies are significantly different (113ms vs. 183ms, details in Table 1)*.
- 需要一些方法，来较为准确的估计模型在边缘设备上的推理时延

**源: MnasNet: Platform-Aware Neural Architecture Search for Mobile (CVPR2019)*

相关工作

- 平台相关(platform-specific)的方法
  - BRP-NAS: model-level, 模型变化→重新构建
  - Nn-Meter: kernel-level

  } 平台变化→重新构建
- 平台无关(platform-adaptive)的方法
  - HELP: model-level, 元学习需要在不同异构平台上进行大量测试
  - OneProxy: 仅排名, 无实际时延值
- 共同问题：现有采样方法不能很好的采样“关键”数据

相关工作对本文的启发

- Kernel级预测，对不同模型无需重新构建预测器

- NAS空间内的高质量采样（生成高质量的配置数据）
  - Challenge 1: 如何实现高质量采样 → VAE数据采样器
- 易于迁移到新平台（少量数据进行微调）
  - Challenge 2: 如何利用不同平台的相似性 → 知识库
    使用KL散度计算分布相似度

影响推理时延的因素

- 设备（CPU/GPU, 设备型号）
- 推理框架（TFLite/ONNX）
- CPU/GPU频率
- 量化精度 FP32/INT8 …

- 影响因素多样，迁移难度大
- 需要发现内在的一些规律

影响推理时延的因素

- 作者推测：存在类似的Pattern
  - 移动SoC的相似性
  - 推理框架的共同目标：最大化底层资源利用率，以最小化推理时延
- 为了证明，两个实验
  - Conv 3x3在不同设备上的时延
  - Conv 3x3在不同推理框架下的时延

影响推理时延的因素

- 不同设备上时延

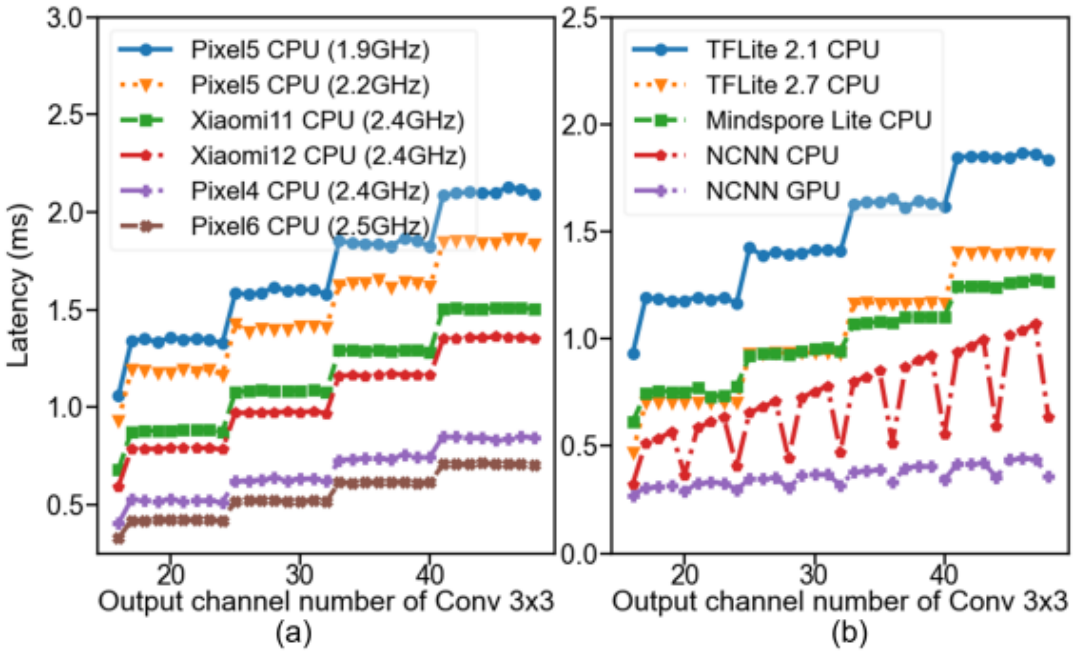


Figure 1: (a) Under the same TFLite 2.1, mobile ARM CPUs exhibit a similar staircase latency pattern; (b) On Xiaomi11, various frameworks exhibit similar latency patterns. Config: HW=56, Stride S=1, Input channel C_{in}=16.

- 不同推理框架时延

- Key Idea: 平台知识 → 通用知识库 →(微调)→ 新平台

NAS搜索空间数据分布

- 问题分析与发现
  - 影响时延的因素
  - NAS搜索空间数据分布
- 系统结构与原理
- 实验
- 总结

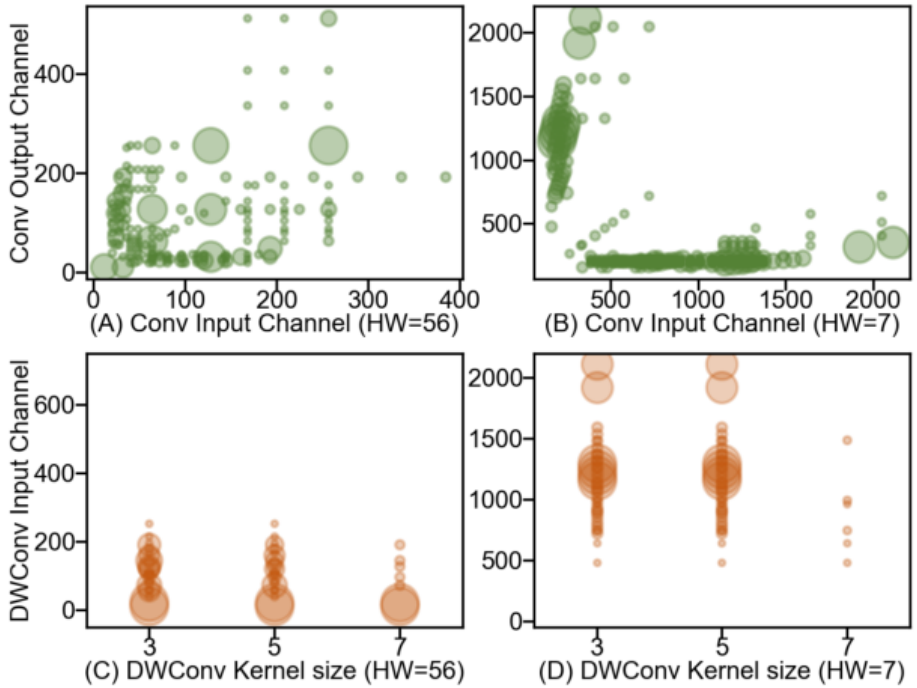


Figure 3: The multi-dimensional distributions of Conv/DW-Conv configurations in 5 widely-used NAS search spaces. (AB): C_{in} vs. C_{out} exhibit different patterns under different HW. (CD): The kernel size KS and C_{in} of DWConv also exhibit different distributions under different HW. Larger circle size indicates that the configurations have larger frequency.

LitePred 系统概述

- 问题分析与发现
- 系统结构与原理
  - 系统概述
  - 训练VAE
  - 构建初始预测器
  - 迁移到新平台
- 实验
- 总结

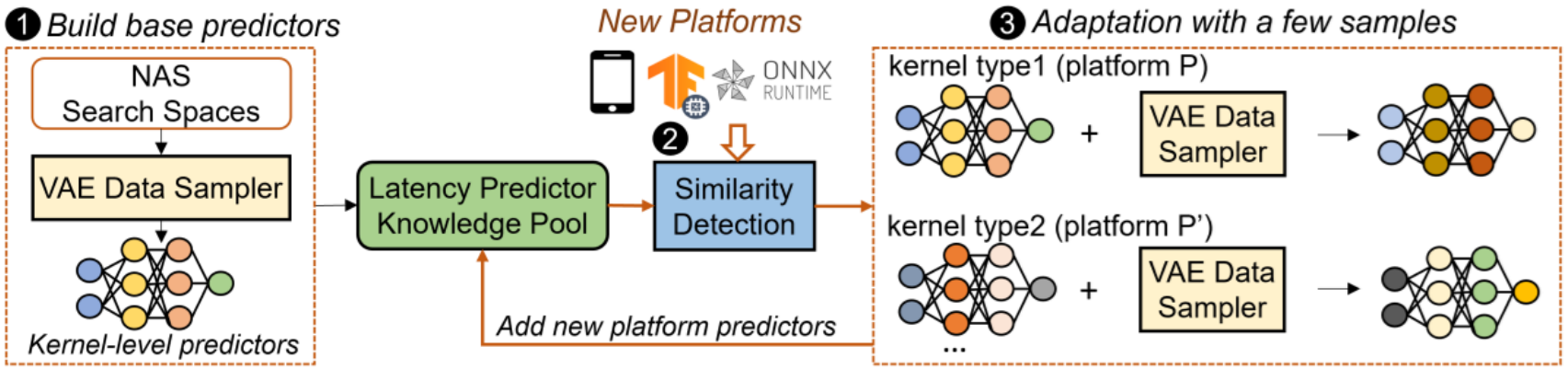
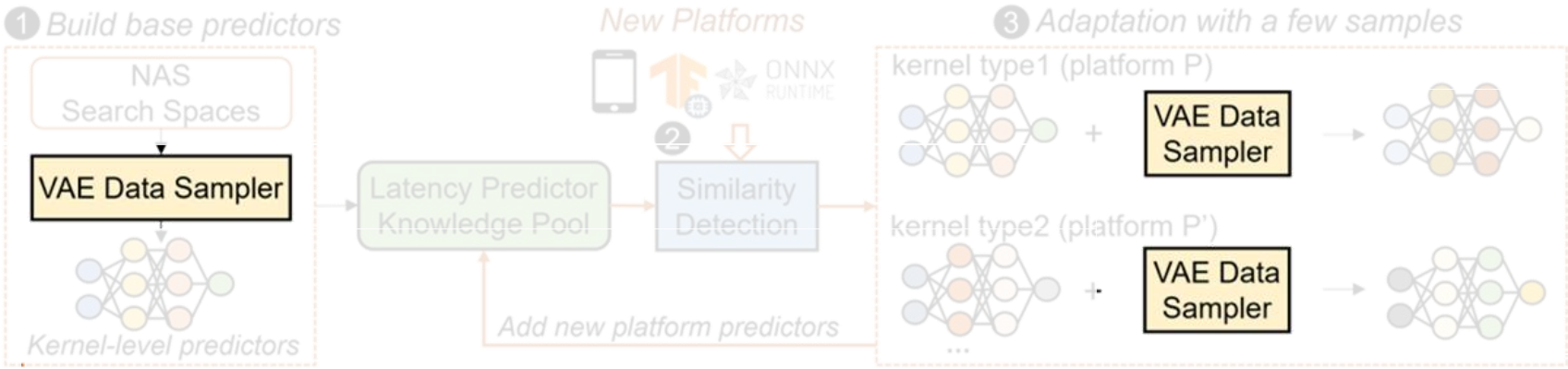


Figure 2: The overview of LitePred. (1) LitePred starts with building accurate base predictors for warmup platforms and stores them in a knowledge pool. (2) For a new platform, we match the most similar latency predictors for each kernel from knowledge pool through similarity detection. (3) We adapt identified latency predictors with a few samples from the new platform.

- 训练VAE Data Samplet (Challenge 1)
- ① 针对热身平台构建若干初始预测器 存入知识库
- ② 针对新平台，在知识库中选取最相似预测器 (Challenge 2)
- ③ 基于此预测器进行微调

训练 VAE Data Sampler

- 问题分析与发现
- 系统结构与原理
  - 系统概述
  - 训练VAE
  - 构建初始预测器
  - 迁移到新平台
- 实验
- 总结

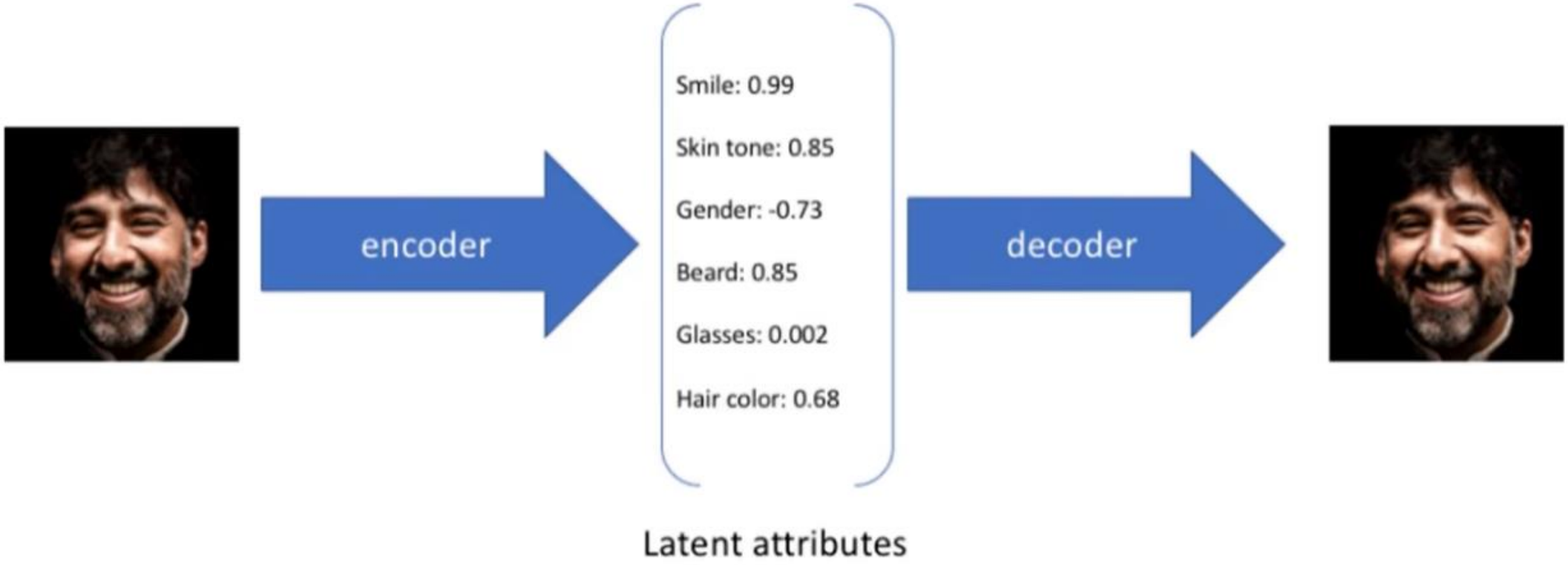


什么是VAE?

- 自编码器 (AutoEncoder)
  - 一种无监督学习算法
  - 将输入数据压缩到一个较低维度的空间(潜在空间), 然后重构原始数据

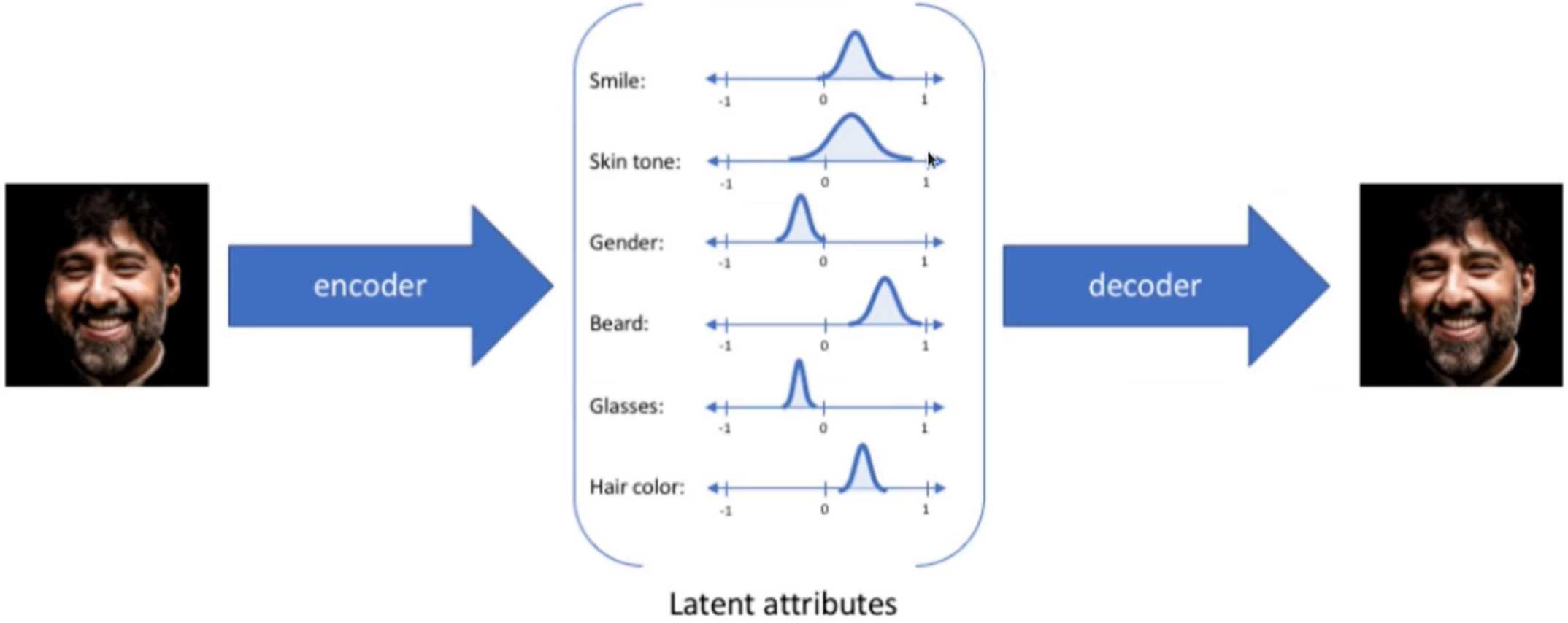
- 变分自编码器 (Variational AutoEncoder)
  - “变分”: 变分推理，用一个相对简单的分布去近似一个复杂的分布
  - 将输入数据映射到潜在空间的一个分布(而不是确定值)

AutoEncoder



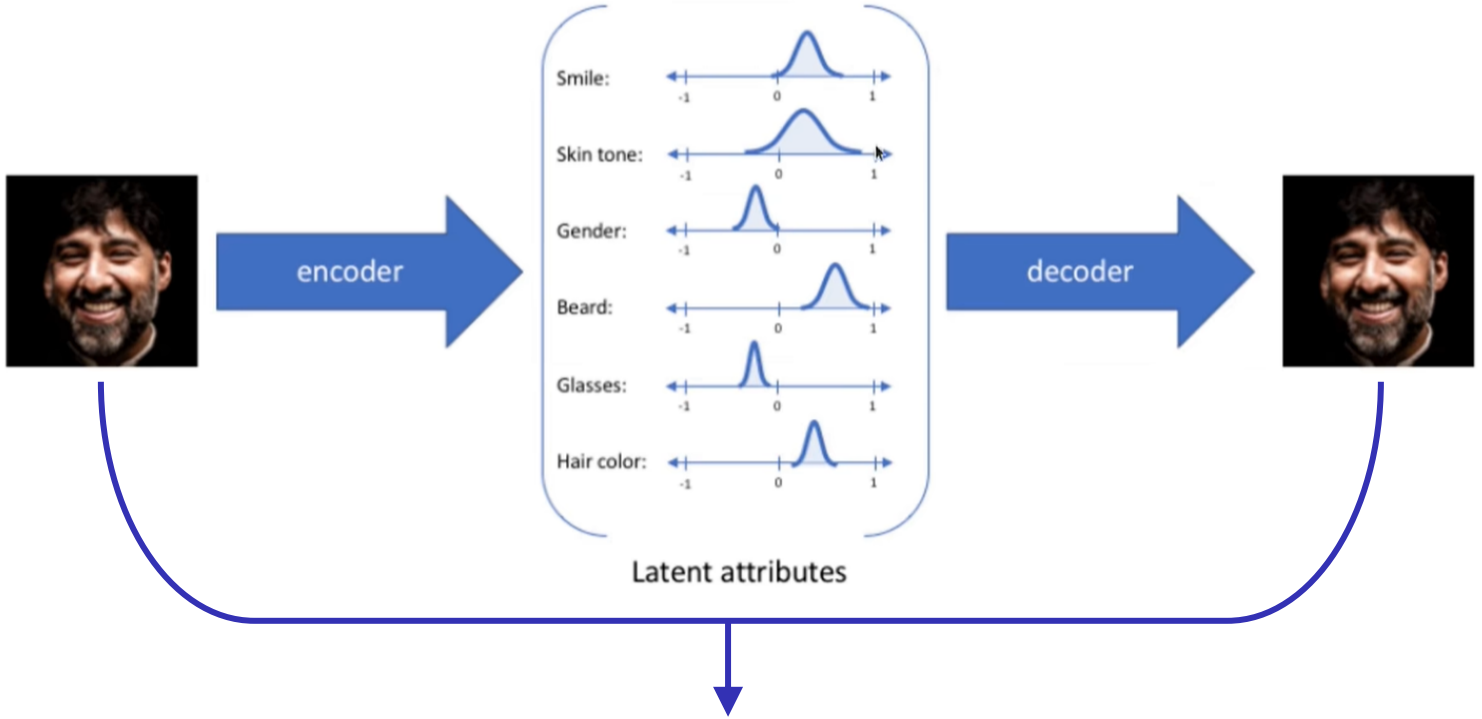
- 输入数据 → 隐变量 → 输出数据

Variational AutoEncoder



- 输入数据 → 隐变量的**分布** → 输出数据

训练 VAE



- (1) 重建损失 (均方误差MSE)
- (2) 分布差异 (KL散度)

Loss

LitePred 中的 VAE Data Sampler

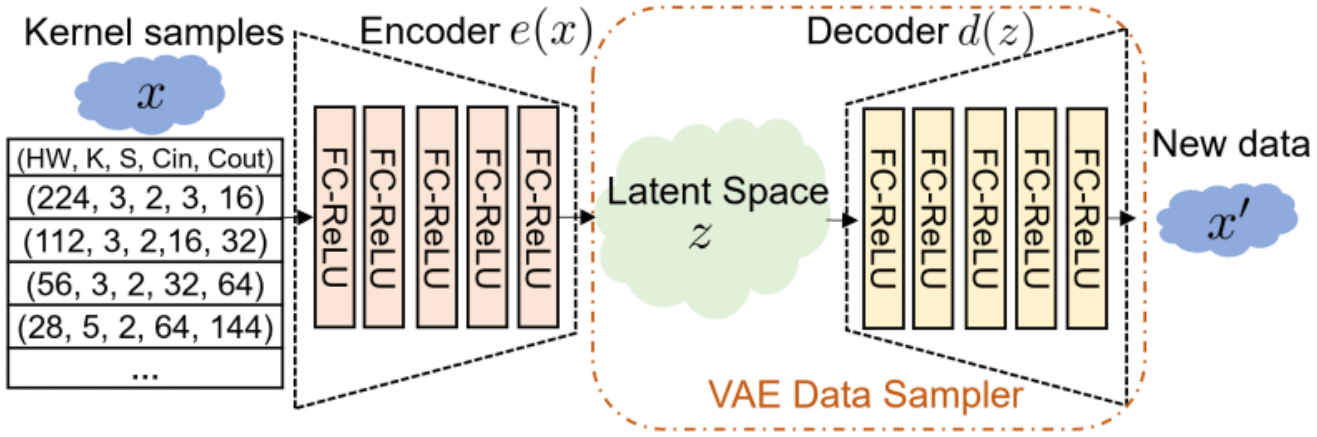


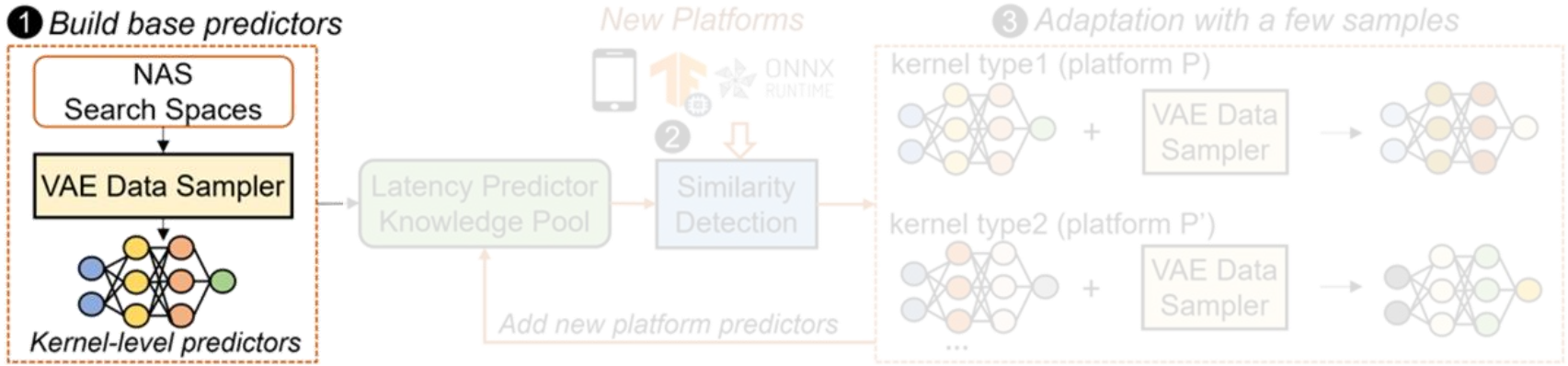
Figure 4: Our VAE data sampler for collecting training data.

- Kernel sample: 从NAS搜索空间中按比例收集
- 重点关注对时延影响较大的Kernel (如Conv DwConv)

NAS search space collection. We collect 5 widely-used CNN and 1 vision transformer NAS search space, including OFA-MobileNetV3 [22], ProxylessNAS [9], OFA-ResNet [8], BigNAS [53], FBNetV3 [14] and AutoFormer [10].

构建初始预测器

- 问题分析与发现
- 系统结构与原理
  - 系统概述
  - 训练VAE
  - 构建初始预测器
  - 迁移到新平台
- 实验
- 总结

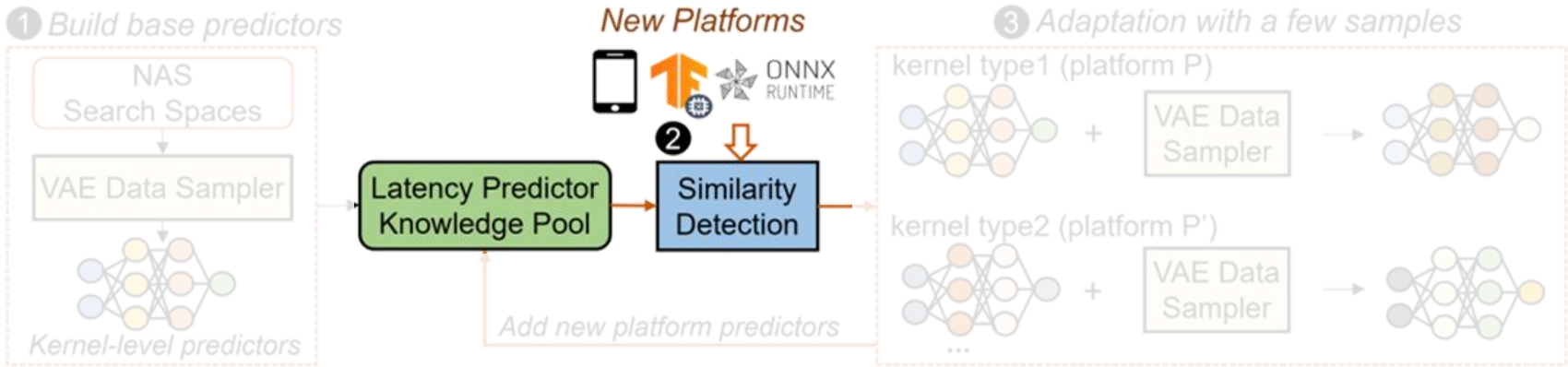


构建初始预测器

- 使用VAE生成高质量配置数据
  - 在热身平台上测量实际时延
- 预测器: 16层MLP
  - 输入: Kernel配置数据 + FLOPs + parameter size
  - 输出: 预测时延
  - 计算预测时延和实际时延的Loss (MAPE)

迁移到新平台

- 问题分析与发现
- 系统结构与原理
  - 系统概述
  - 训练VAE
  - 构建初始预测器
  - 迁移到新平台
    - 挑选最相似预测器
- 实验
- 总结



从知识库中挑选最相似预测器

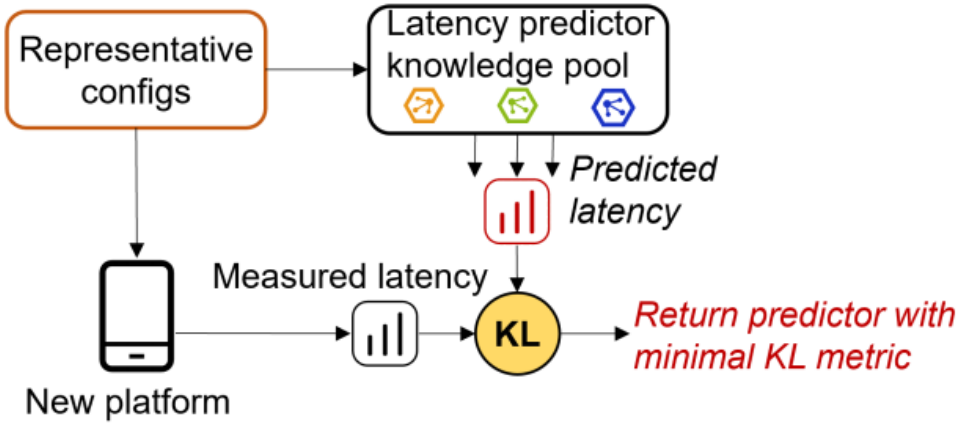
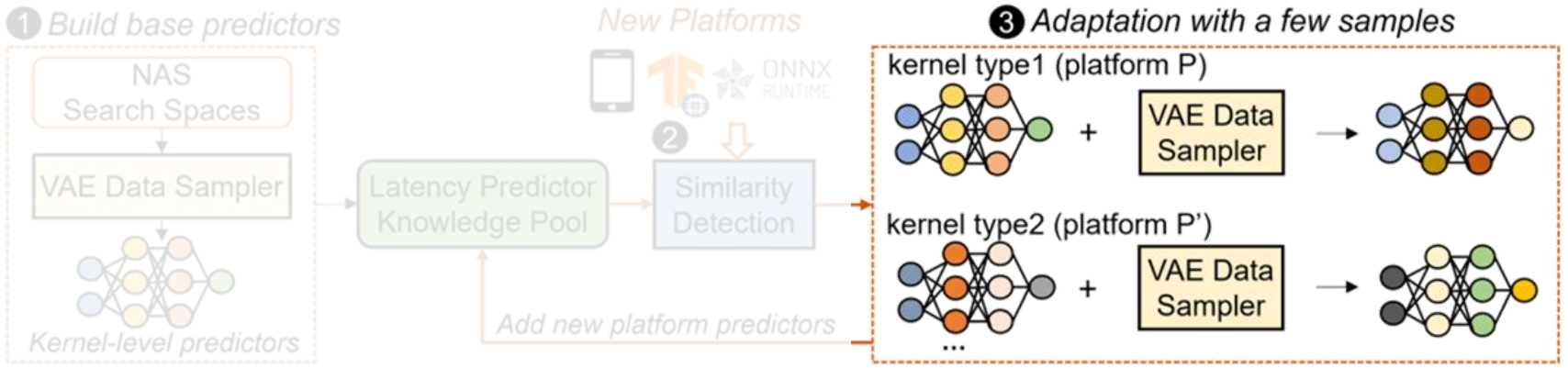


Figure 5: Our proposed similarity detection to identify the most similar predictor for each kernel type on new platforms.

- 使用一组代表性配置数据(人工设计)
  - 测量实际时延
- 针对知识库里的预测器，使用同样数据，得到预测时延
- 分别计算每一个预测器的预测时延与实际时延的相似度
  - 相似度：KL散度 越低，越相似

进行微调

- 问题分析与发现
- 系统结构与原理
  - 系统概述
  - 训练VAE
  - 构建初始预测器
  - 迁移到新平台
    - 挑选最相似预测器
    - 进行微调
- 实验
- 总结



进行微调

- 使用VAE生成少量微调数据(~500)
  - 在目标平台上测量实际时延
- 使用预测器预测时延
- 计算Loss

实验设置

- 实验平台
  - 共85个平台，覆盖10种不同的硬件和CPU频率，5种边缘推理框架，2个数据精度

Device	CPU	GPU	CPU Frequency
Pixel 4	Qualcomm Snapdragon 855	Adreno 640	2.4GHz, 2.1GHz
Pixel 5	Qualcomm Snapdragon 765G	Adreno 620	2.2GHz, 1.9GHz
Pixel 6	Google Tensor SoC	Mali-G78	2.5GHz, 2.2GHz
Xiaomi 11	Qualcomm Snapdragon 888	Adreno 660	2.4GHz, 2.1GHz
Xiaomi 12	Qualcomm Snapdragon 8 Gen 1	Adreno 730	2.4GHz, 2.1GHz
Inference engines	TFLite 2.1, TFLite 2.7, NCNN, Mindspore Lite, Onnxruntime		
Precision	FP32, INT8		

Table 3: Our **85** evaluated platforms, including 10 different hardware and CPU frequencies, 5 popular inference frameworks on edge and 2 data precision.

- 实验数据集
  - 从6个高质量NAS搜索空间构建数据集
  - 从每个搜索空间随机抽取4k个模型，在85个平台上测量延迟 (1.86 million数据)

NAS search space collection. We collect 5 widely-used CNN and 1 vision transformer NAS search space, including OFA-MobileNetV3 [22], ProxylessNAS [9], OFA-ResNet [8], BigNAS [53], FBNetV3 [14] and AutoFormer [10].

实验

- 对比实验：在若干平台上与HELP和nn-Meter进行预测对比
  - 使用LitePred、HELP、nn-Meter在不同平台上进行延迟预测
  - 测试迁移到新平台的能力
  - 指标：均方根误差RMSE、预测精度、分析成本
- 结果：
  - 87%的CNN模型预测误差在5%以内，优于HELP和nn-Meter
  - LitePred的预测准确性更高，并显著降低了迁移到新平台的成本 (0.05~1.73 hour)

实验

- 消融实验：
  - 检测VAE Data Sampler的有效性 (与nn-Meter中的自适应采样对比)
    - 结果：VAE数据采样器在所有平台上都显著优于自适应数据采样器，在卷积核（Conv）和深度可分离卷积核（DWConv）的预测精度上都有较大提升
  - 检测相似性检测方法的有效性 (与OneProxy和随机选择对比)
    - 结果：通过本文的相似性检测方法选择的预测器在新平台上的微调精度更高，所需的适应数据量更少

实验

- 在OFA(目前最先进的NAS方法)中使用LitePred
  - 在4个不同的边缘平台上进行延迟约束搜索。对于每个给定的延迟约束，搜索5000个模型架构，并选择在ImageNet 2012数据集上验证准确度最高的模型，然后评估最终模型的测试准确度和设备上的延迟。
  - 结果：与MobileNetV2和MobileNetV3（为边缘平台设计的轻量级CNN的最新技术）相比，OFA与LitePred结合的搜索模型在ImageNet上的准确度更高，延迟更低。在4个不同的边缘平台上，搜索到的模型比MobileNetV2高出最多4.4%的ImageNet准确度

实验

- Transformer模型实验
  - 对5个视觉Transformer模块进行了时延预测
  - 结果：99.9%的模型预测误差在10%以内。
  - 说明LitePred不仅对CNN有效，对Transformer的预测效果也较好

总结

- 问题分析与发现
- 系统结构与原理
- 实验
- 总结

- LitePred:
  - 准确预测DNN在边缘设备上的推理时延
  - 轻量级, 可迁移
- 设计原则:
  - 针对某平台的延迟预测器，其知识可以转移到其他具有相似性的新平台
- 关键技术:
  - VAE Data Sampler
  - 基于KL散度的相似平台检测方法
- 实验:
  - 低成本(1小时内)实现了99.3%的平均延迟预测精度
  - 准确率提高了5.3%，分析成本降低了50.6倍
  - 应用于NAS，在ImageNet上提高了4.4%的准确率

Idea

- VAE Data Sampler能否更换
- 类似迁移思想能否用在其他领域

请各位老师和同学批评指正

汇报人：姚昌硕

2024年7月25日