高性能特征工程 Pipeline 设计要点

钱烽

当前深度学习技术已经在搜索、广告、推荐等点击率预估及类似场景中得到了广泛和有效的运用，取得了一些突破性的进展。其中一部分进展得益于在隐藏层之外显示构造特征组合，从而弥补神经网络的局部低效表达能力。从这个角度看，特征交叉等传统的特征工程技术仍将在一定时间内继续存在。另一方面，在工业场景中，作为基线的 LR/FTRL 模型或者 GBDT 模型，也是考察深度神经网络效果的必要对照实验组，甚至是初次场景建模的首选打底方案。

本文主要介绍配合这类方法的关键技术设计要点：高性能特征工程的 Pipeline 设计。

一、设计范围

数据类型：结构化数据（Tabular Data）

场景类型：大规模样本、高维稀疏特征、实时特征

非结构化数据不是本文的设计范围，下面只做简单描述。

（1）非结构化数据类型
- 文本、图像、语音数据等
- 特点：具有空间或时间相关性

（2）一般采用深度学习端到端建模
- 不同框架提供不同的一致性端到端解决方案
- Pytorch/Fastai（离线）— Caffe2（在线）
- Keras（离线） — TFX（在线）

（3）拥有特定数据预处理工具
- TorchText、PyText
- TorchImage
- TorchAudio

（4）特定情况可转换为结构化数据
- 文本数据：BOW (高维稀疏：Set Categorical)

二、设计目标

（1）一致性

- 线上排序与离线训练使用同一数据处理 Pipeline（必须完全一致）

（2）复杂性

- 十亿级特征维度、支持近实时特征
- 支持所有的属性转换操作、特征转换操作

（3）高性能

- 经验条件举例：单机 48cores，单次排序请求包含 300 个物料、十亿级别特征
- 经验性能举例：平均延迟：30ms；QPS：300 次请求/s （每秒约十万次打分）
- 可以做得更好（例如，指令集优化）

（4）扩展性

- 在线排序
  1. 支持不同服务架构（传统单点、微服务、容器）
  2. 弹性扩展到超大规模并发请求
- 离线训练
  1. 支持不同数据流框架（批处理、分布式、流式）
  2. 弹性扩展到超大规模样本数量

三、数据模型

所有系统中只存在三种数据状态类型：

（1）属性数据

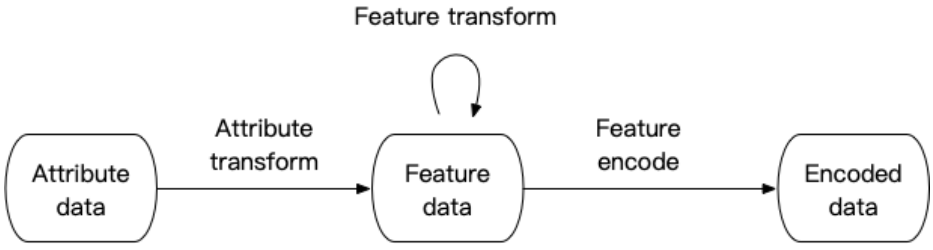
- 业务数据的原始形式
- 是对业务数据的高度抽象（结构化数据的前提下）

（2）特征数据

- 特征工程的数据闭包

（3）编码数据

- 机器学习模型使用的数据形式



四、属性数据及其转换

（1）属性数据类型（冒号前是属性名）

- Categorical：例如 Sex: Male（程序表示：string/int）
- Set-Categorical：例如 Interests: {sport: 3, ent: 6}（程序表示：[<string, float>]）
- Real Valued：例如 Fail Times: 3（程序表示：float）
- Unix Timestamp：例如 Occur time: 1546333884（程序表示：float）

（2）属性转换

- Input：属性数据 =》Output：特征数据
- One Output Op
    1. Sex: Male => Sex_Male: 1.0
    2. Fail Times: 3 => (Direct Transform) Fail Times: 3.0
- Multiple Output Op
    1. Interests: {sport: 3, ent: 6} => {Interests_Sport: 0.33, Interest_Ent: 0.67}
    2. Occur time: 1546333884 =>
        Weekday: 1.0
        Afternoon: 1.0
        Festival: 1.0
        Tuesday: 1.0
        First day of month: 1.0

五、特征数据及其转换

（1）特征数据类型（冒号前是特征名）

- Discrete Feature Data
    1. 特征值只有 0/1 两种值
    2. Sex_Male: 1.0
    3. Weekday：1.0
- Continuous Feature Data
    1. 特征值是连续数值
    2. Fail Times：3
    3. Interest_Sport:0.33

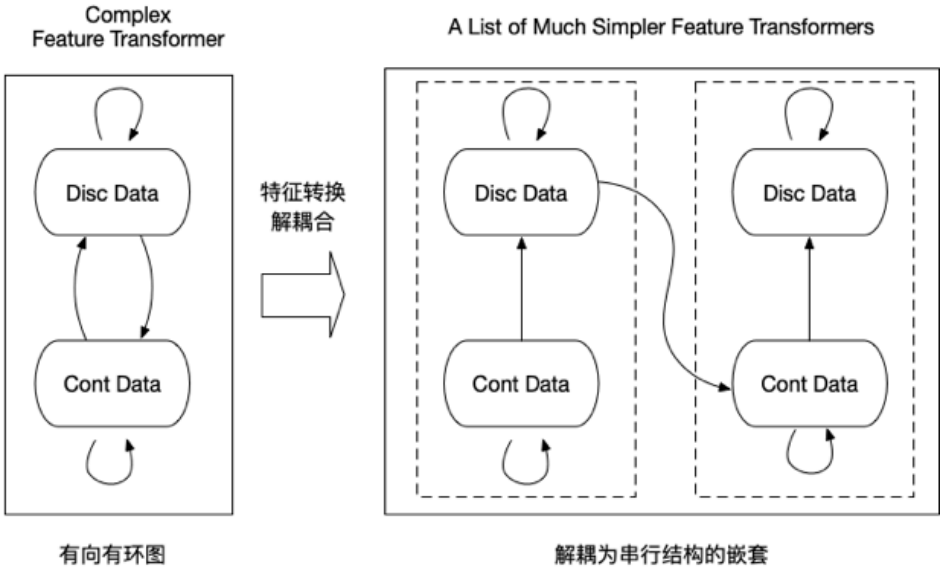
（2）特征转换

- Unary Transform Op
    1. Cont =》Disc，e.g., Bucketize
    2. Disc =》Cont，e.g., Lookup Historical Statistics
- Binary Transform Op
    1. Disc^Disc => Disc，e.g., Feature Cross
    2. Cont^Cont => Cont，e.g., Similarity Feature

- 有向有环图
  1. 解耦为串行结构的嵌套
  2. 见下节

六、特征转换的解耦设计

- 有向有环图 =》串行结构的嵌套
- 优点
  1. 简化了工程实现：复杂结构 =》简单结构的串行
  2. 虚框之间的转换可以在外部系统中进行（例如，实时特征的构造）



七、编码数据及其转换

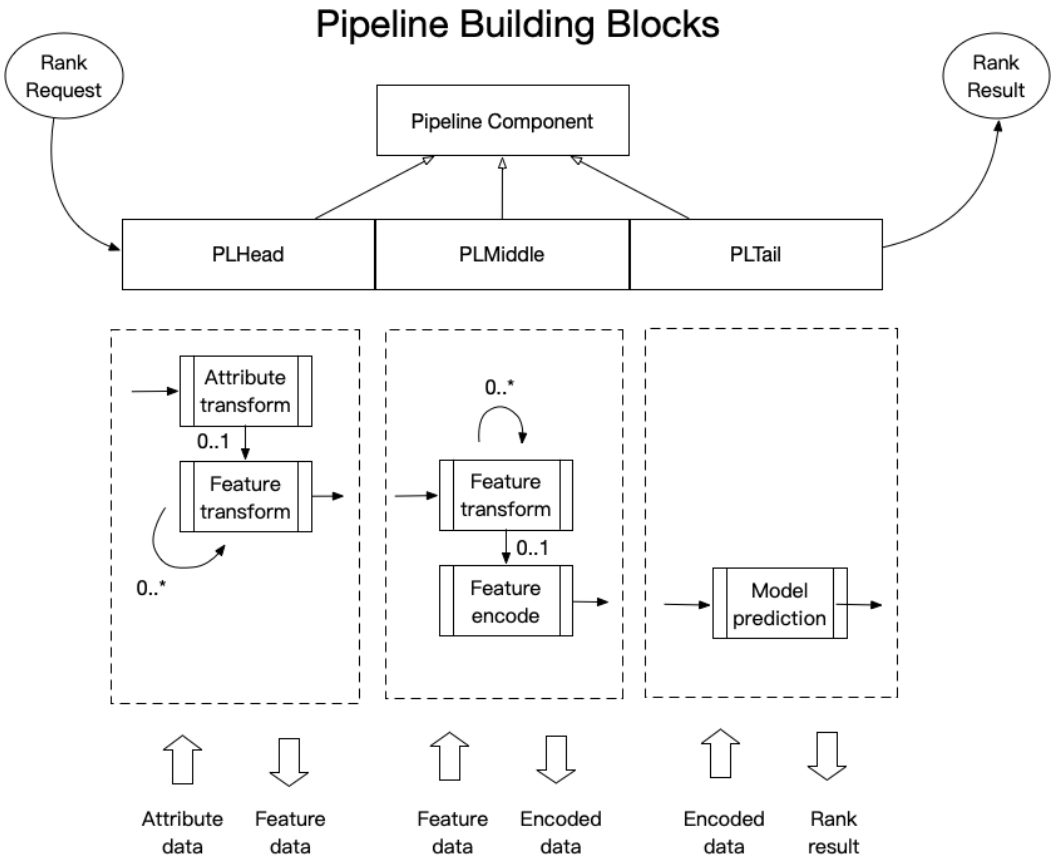
编码数据类型（冒号前是特征编码，冒号后是特征值）

- Hash Encoding
  1. 高维稀疏特征 （LR/FTRL 模型）
  2. 工业界一般使用 Murmur Hash
  3. 32 bits encode roughly 4 billion features

- Sequence Encoding
  1. 低维连续特征（GBDT 模型）

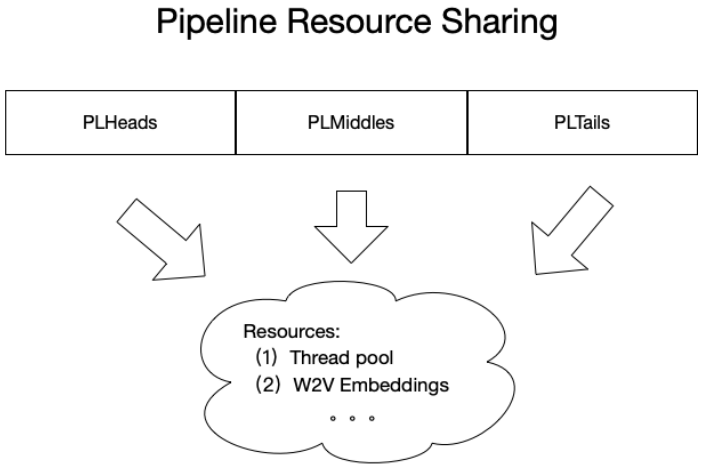
八、特征工程 Pipeline 设计

将整个 Pipeline 拆分为三个粒度适中的部分，兼顾了工程实施的灵活性和便捷性。

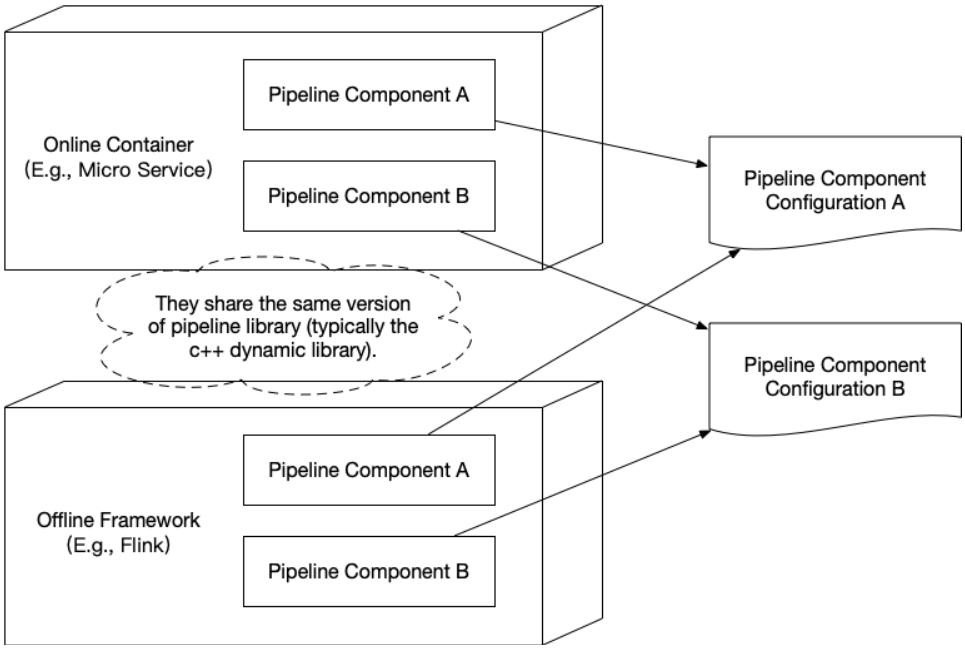


九、特征工程 Pipeline 的资源共享

基于动态规划的理念，自底向上实现资源的最大化重复利用。

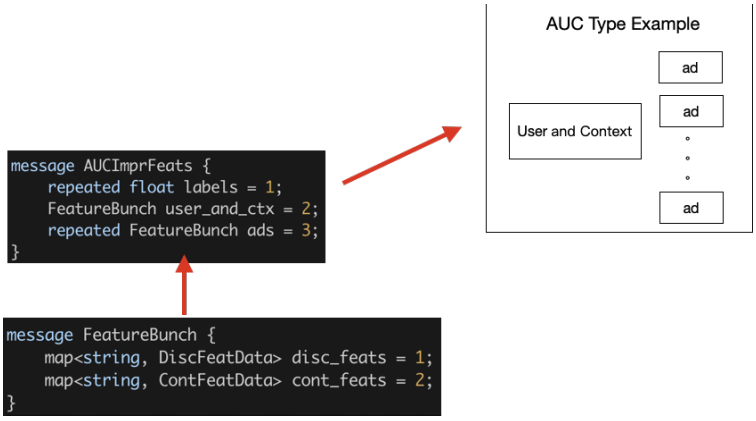


十、线上线下计算的强一致性保证



十一、排序任务的数据组织结构

- User and Context 处理只计算一次
- 多线程实施位置：ads



十二、系统模块间的交互协议

- 系统模块的交互场景
  1. 离线：e.g., Flink 内调用 C++ DSL
  2. 在线：e.g., 微服务内调用 C++ DSL
- 程序设计语言的交互协议
  1. Swig（Java/Python => C++）
- 数据的交换协议
  1. Protobuf 格式

十三、特征工程及 Pipeline 的可配置性

不同粒度的特征工程部件：属性转换、特征转换、模型、资源管理、Pipeline 等都做到可配置的程度，添加新特征或新算子非常方便。