微博基于Flink的机器学习实践

于茜

yuqian8@staff.weibo.com

微博机器学习研发中心

Agenda

1. 关于微博
2. 微博机器学习平台（WML）总览
3. Flink 在 WML 中的应用
4. 使用 Flink 的下一步计划

关于微博

- 2008年上线
- 中国最大的、最流行的社交媒体平台
- 提供人们在线创作、分享和发现优质内容的服务
- 大规模机器学习平台可支持千亿参数，百万QPS

222M

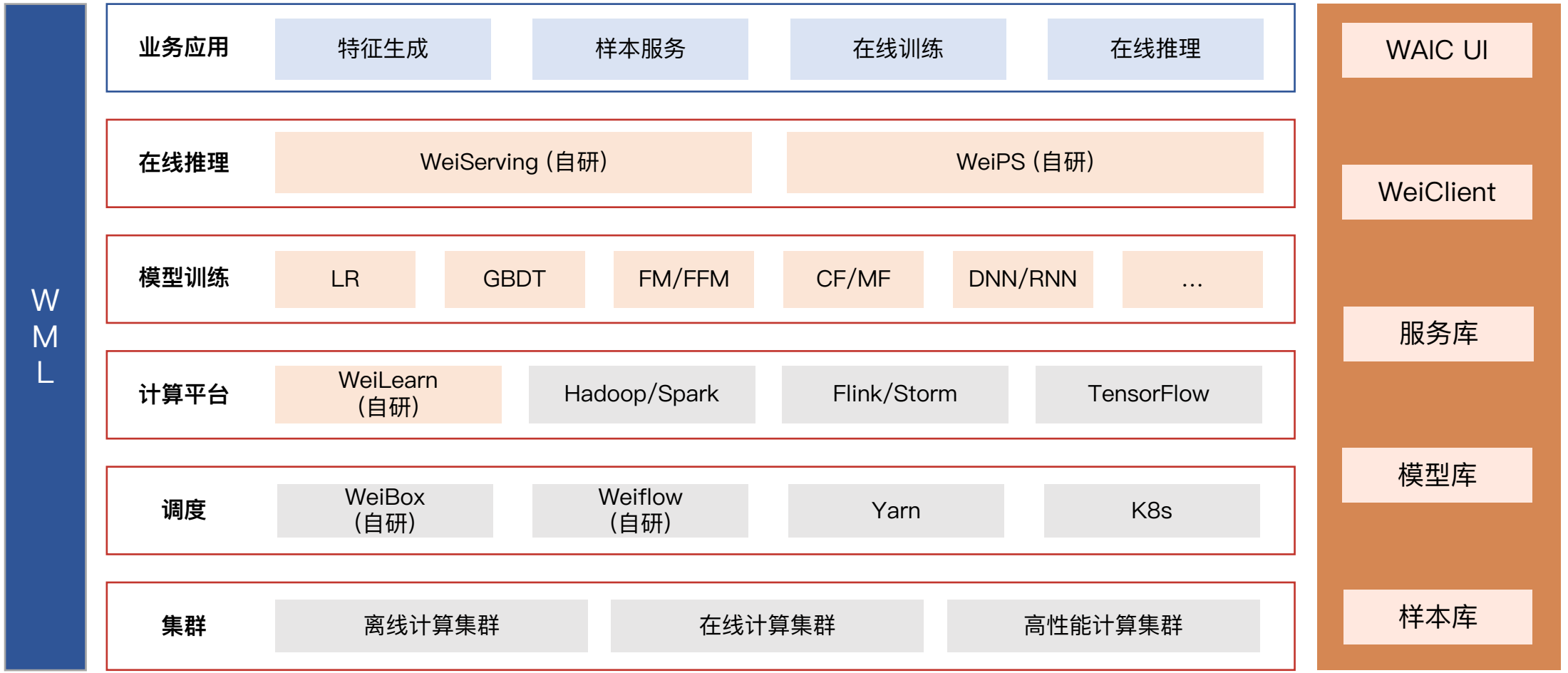
DAU

516M

MAU

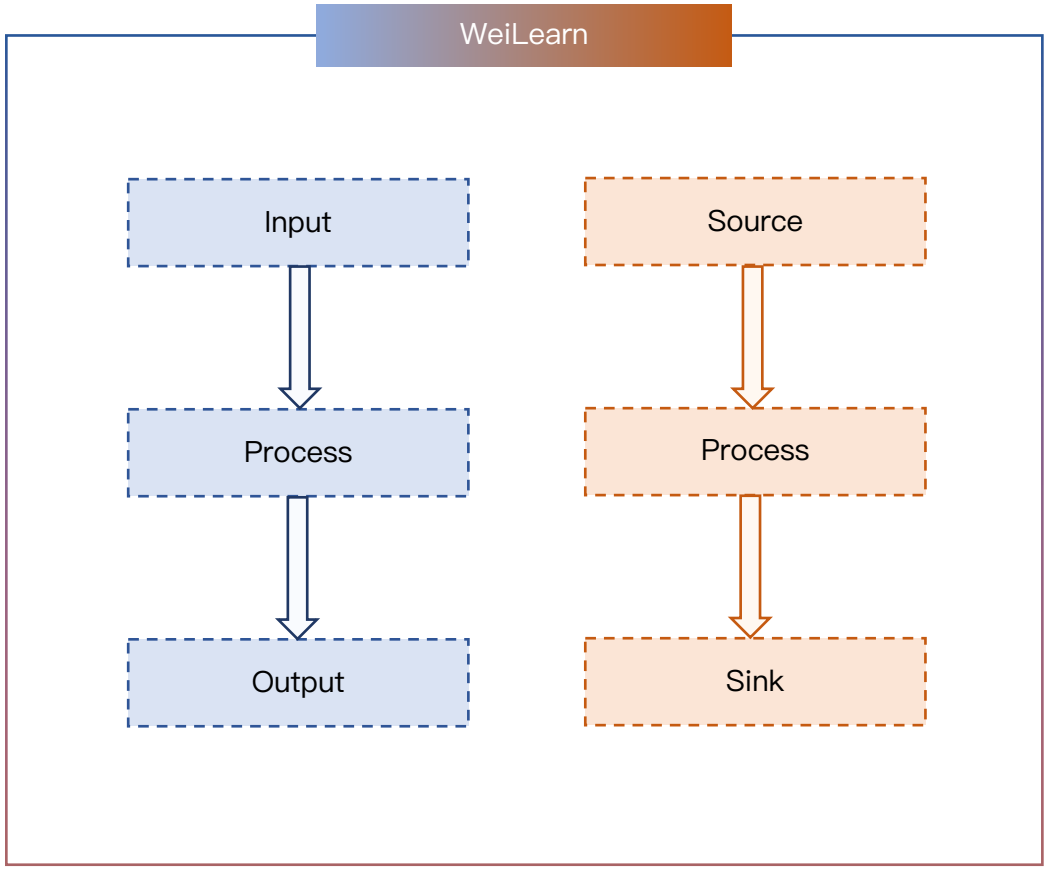
微博机器学习平台 (WML) —— 总览

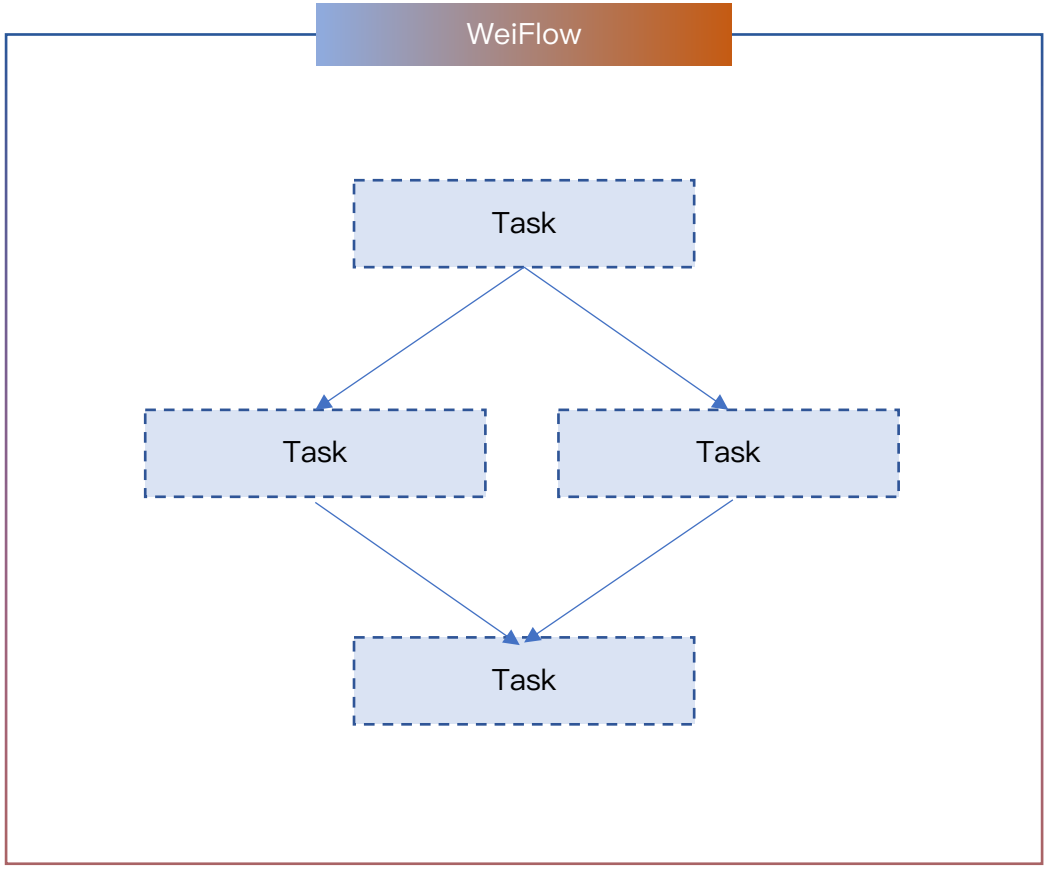
机器学习平台(WML)为CTR、多媒体等各类机器学习和深度学习算法提供从样本处理、模型训练、服务部署到模型预估的一站式服务



微博机器学习平台 (WML) —— 开发模式

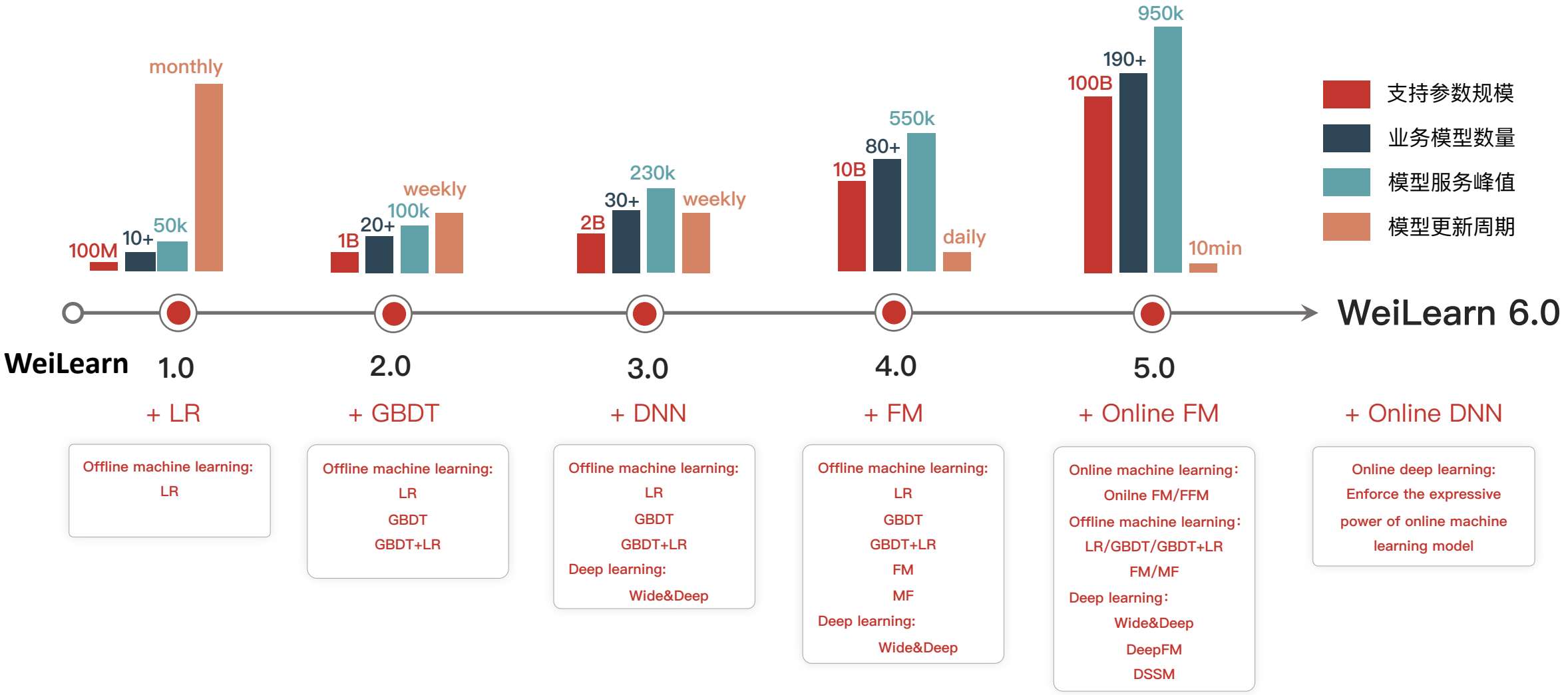
双层DAG设计:



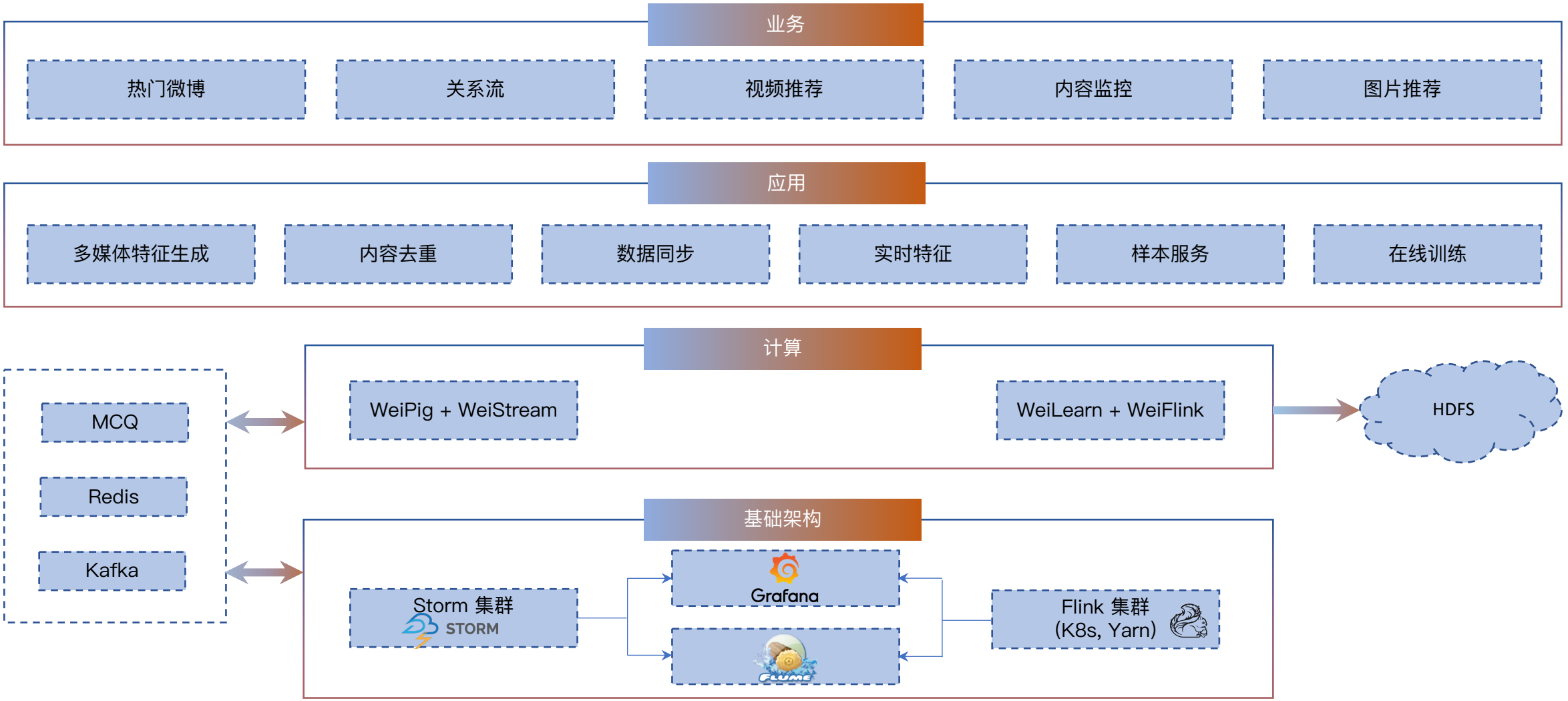


微博机器学习平台 (WML) —— CTR模型

机器学习平台(WML)经过历次迭代，目前支撑的参数规模达**千亿级**，服务峰值达**百万QPS**，模型更新达**10分钟级**

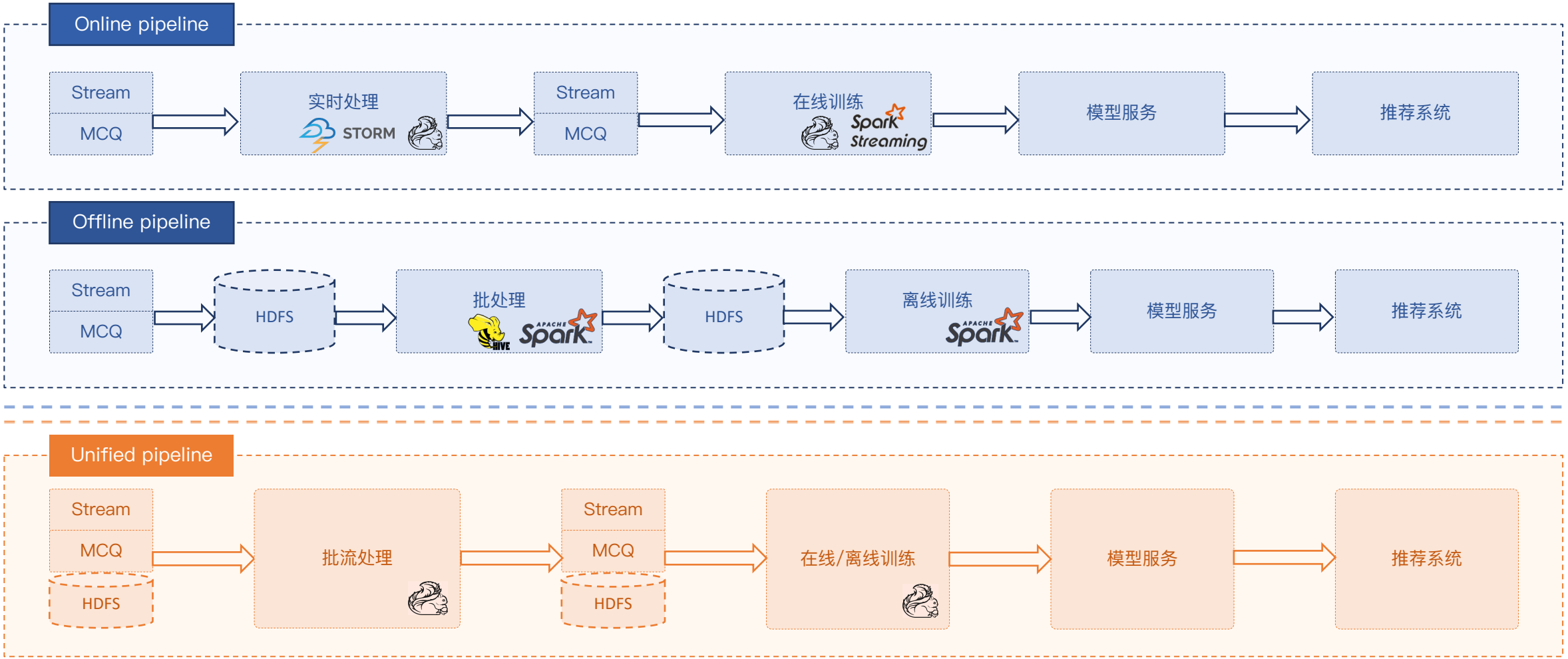


Flink在WML的应用 —— 概览

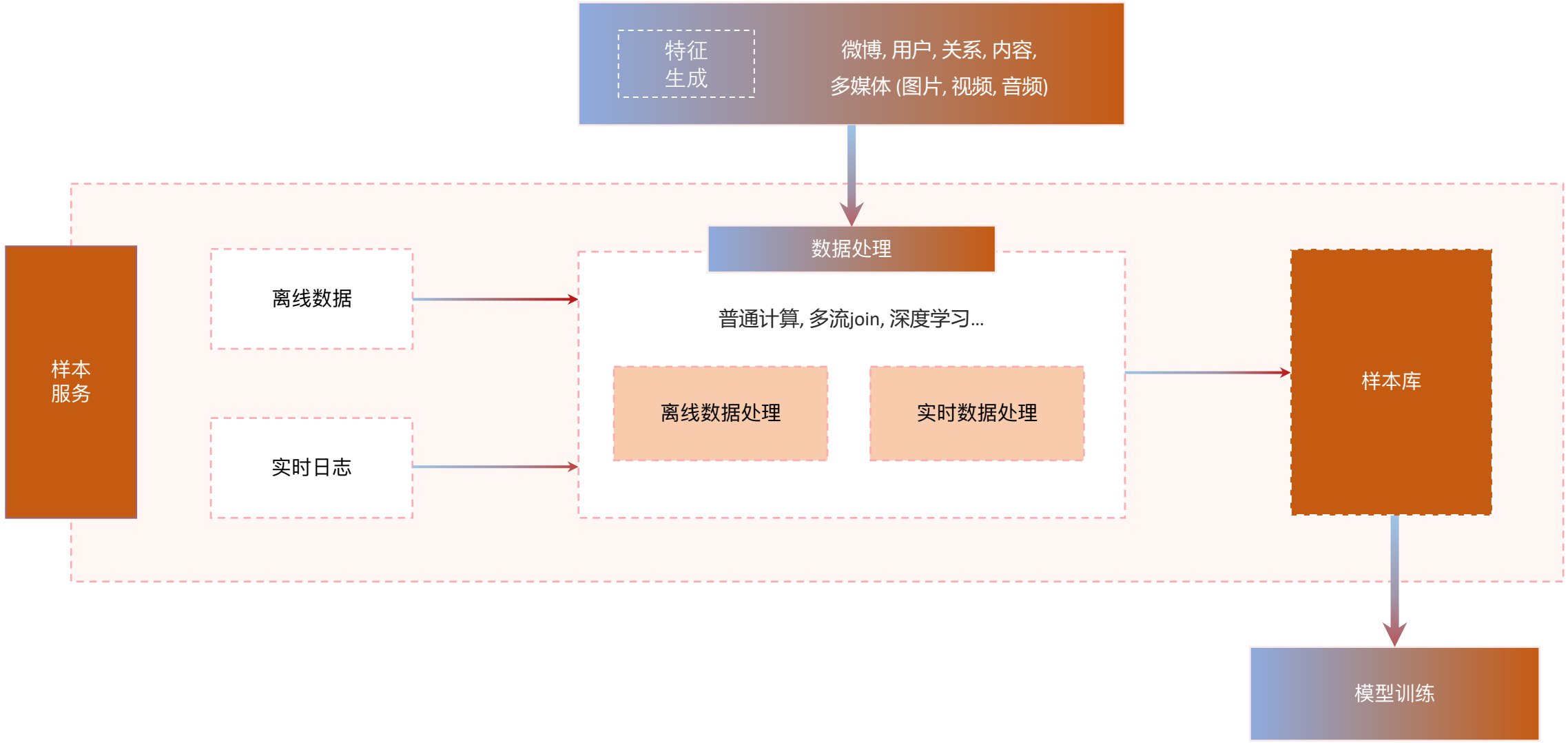


Flink在WML的应用 —— 概览

将统一的Flink API应用到实时ETL以及离线ETL中



Flink在WML的应用 —— 样本服务



Flink在WML的应用 —— 多流Join

多流Join的流程

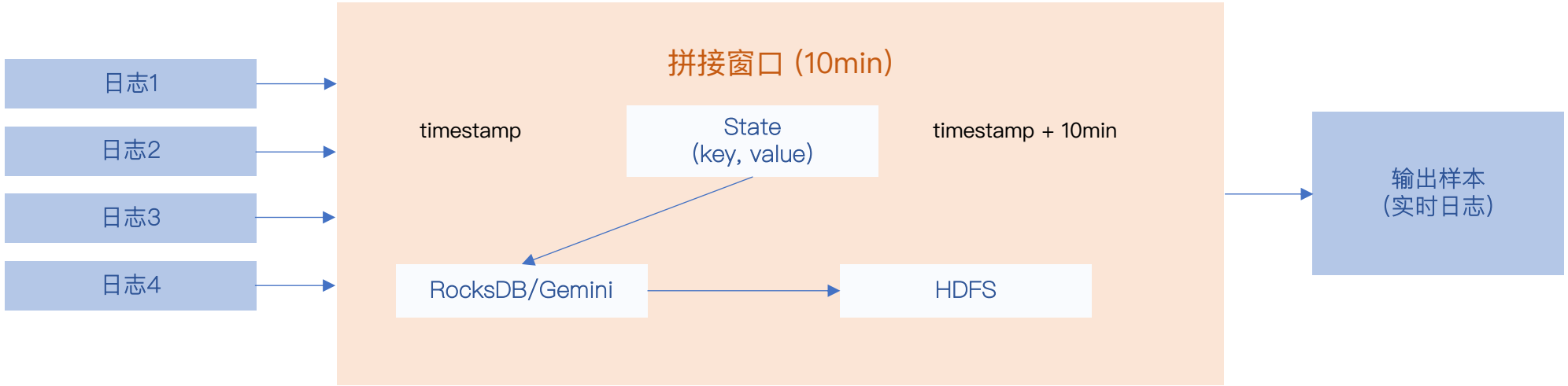
UDF:

```
@Override
public boolean filter(Tuple2<String, DefaultOutModel> data) throws Exception {
    if (this.isEmptyTuple(data)) {
        return false; // filter out empty records
    }

    DefaultOutModel outModel = data.f1;
    String business = outModel.getRecord("business");

    if (business.isEmpty() || !("xxx").equals(business)){
       return false; // only consider "xxx" business
    }
}
```

```
@Override
public Tuple2<String, DefaultOutModel> map(String source) throws Exception {
    Map<String, String> detailMap = JsonUtil.fromJsonToObject(source, Map.class);
    DefaultOutModel outModel = new DefaultOutModel();

    // put <like count in an hour> into output
    outModel.putRecord("lk_hour",
        String.valueOf(detailMap.get("lk_hour")));

    // put <user gender> read from feature engineering into output
    this.appendFeature(outModel, getFeature("userGender"));

    String blogId = String.valueOf(detailMap.get("blogId"));
    String userId = String.valueOf(detailMap.get("userId"));

    // key = userId_blogId, value = output
    this.processOut(Tuple2.of(userId + "_" + blogId, outModel));
}
```

Flink在WML的应用 —— 多流Join

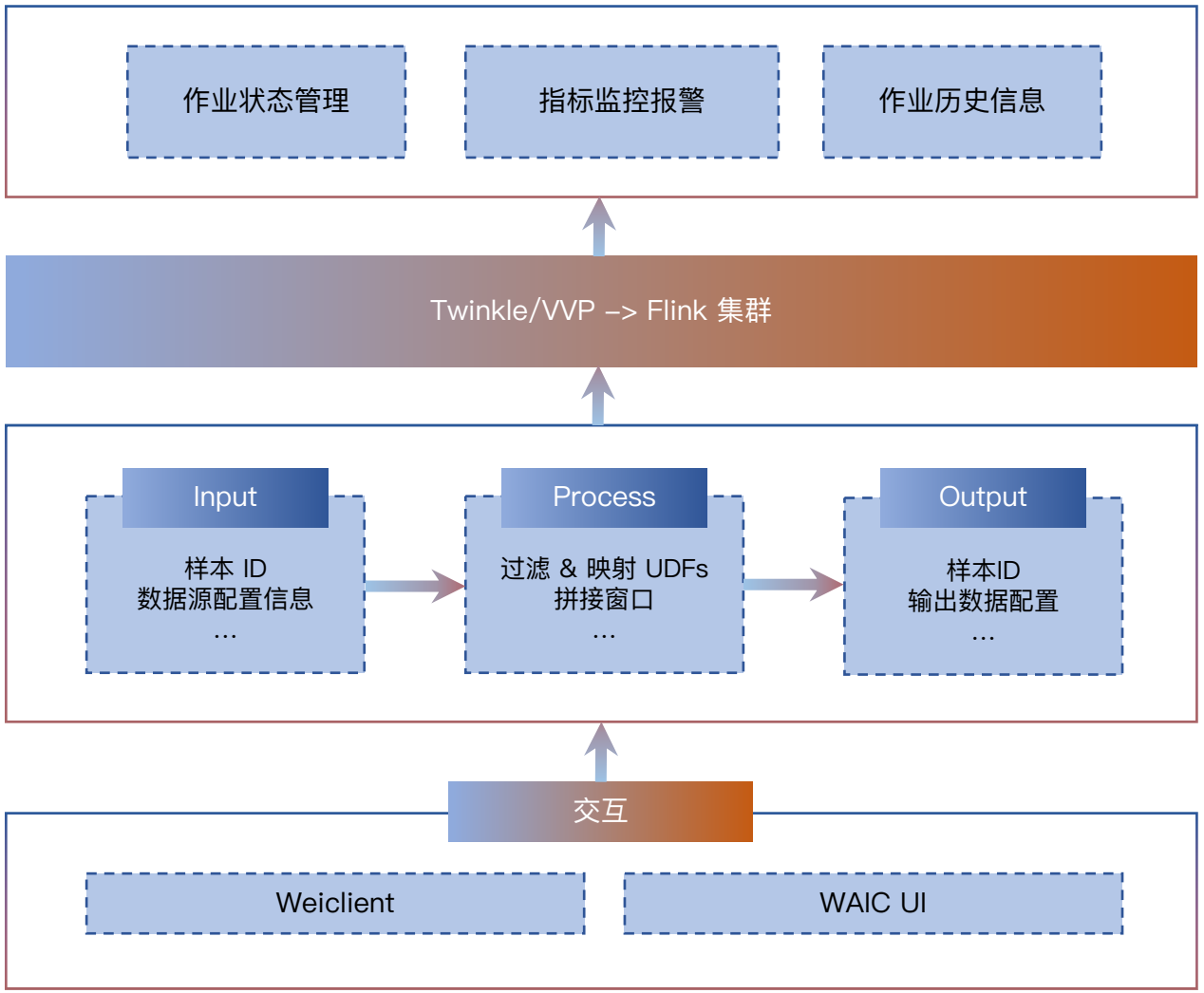
拼接时间窗口 & 样本对齐



WML基于Flink做的一些优化:

1. 自定义样本Trigger触发机制，拼接成功后立即输出样本
2. 样本补偿 PU loss
3. RocksDB vs Gemini
4. 成功率和拼接窗口大小的权衡

Flink在WML的应用 —— 样本服务



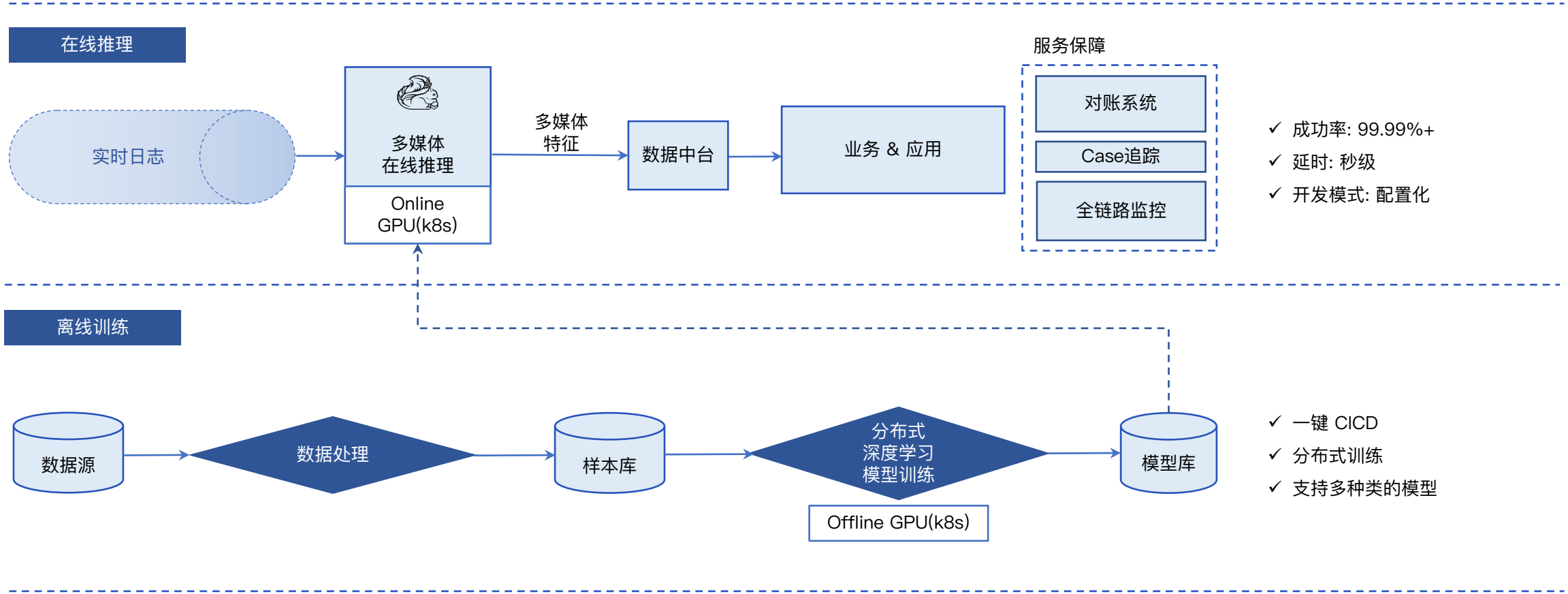
- Grafana: 监控和报警
- Twinkle/VVP 结合 WAIC UI：管理作业
- HDFS: 存储历史作业信息

- 基于Flink的进一步封装-> Weiflink + Weiplugin
- 使用Jenkins完成UDF的cicd
- 使用样本ID来统一输入源
- 在WeiLearn框架内做内层DAG的开发

- WAIC UI：选择数据源和 UDFs
- Weiclient：提交作业到不同的集群

Flink在WML的应用 —— 多媒体特征生成

离线深度学习模型训练 & 基于Flink的在线推理



Flink在WML的应用 —— 多媒体特征生成

高可用 & 容灾

1.全链路监控报警 & Case追踪

2. At least once

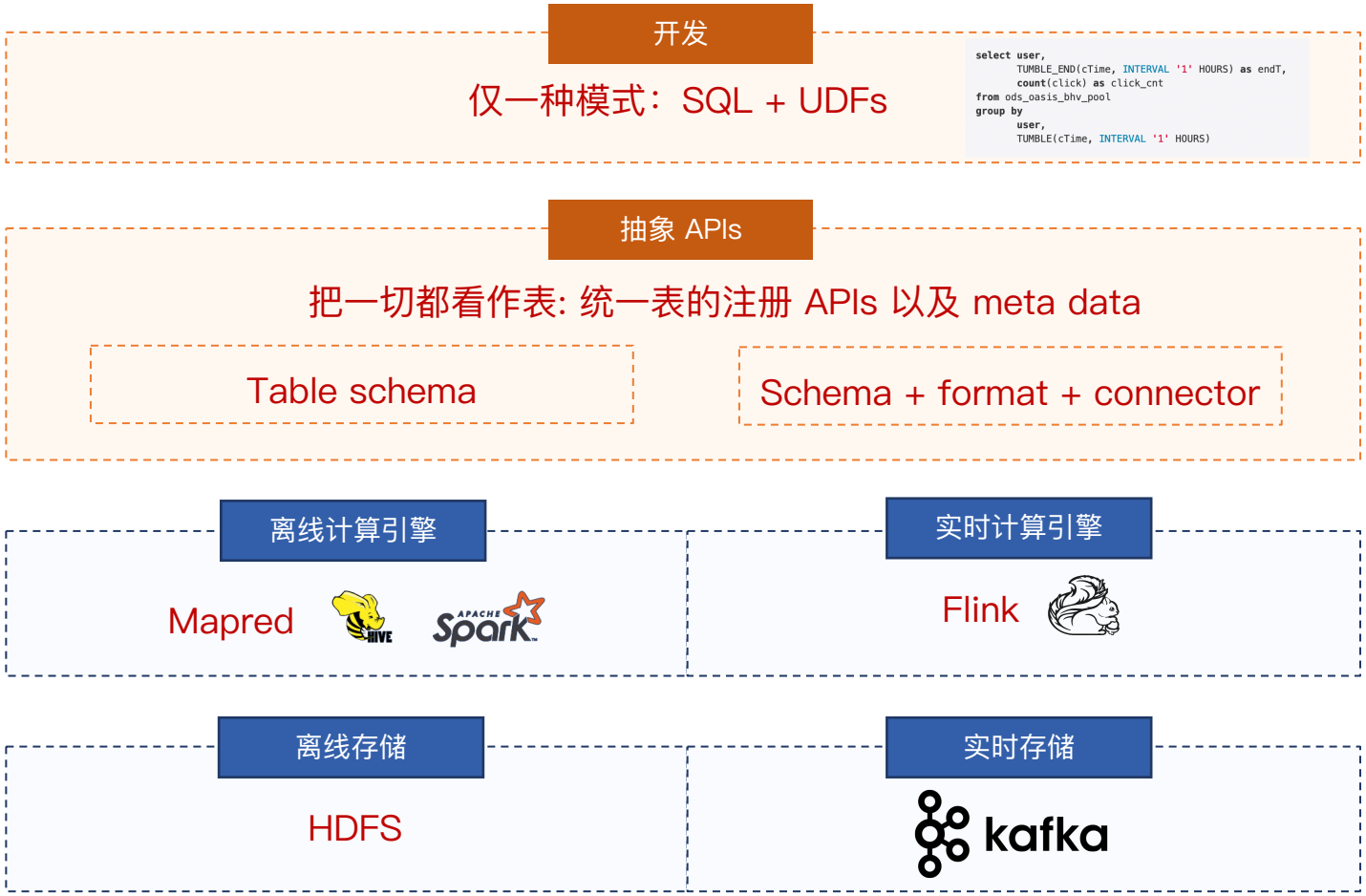
3. 自动重启

4. 从 checkpoints 中恢复数据和State

5. 重试队列 + 对账系统

使用Flink的下一步计划 —— 实时数仓

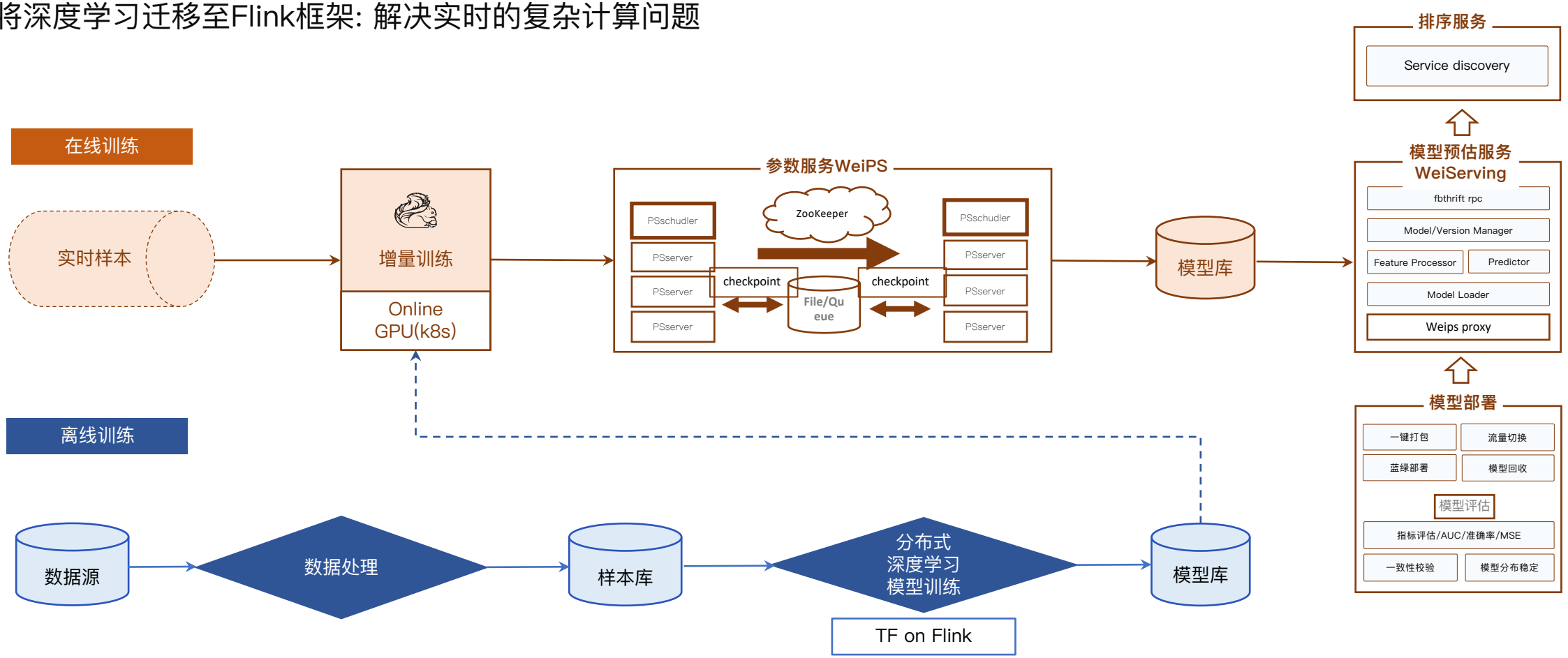
实时数仓: 统一离线数据和实时数据的schema及APIs




```yaml
tables:
  - name: TaxiRides
    type: source-table
    update-mode: append
    connector:
      property-version: 1
      type: kafka
      version: "0.11"
      topic: TaxiRides
      startup-mode: earliest-offset
      properties:
        zookeeper.connect: localhost:2181
        bootstrap.servers: localhost:9092
        group.id: testGroup
    format:
      property-version: 1
      type: json
      schema: "ROW<rideId LONG, lon FLOAT, lat FLOAT, rideTime TIMESTAMP>"
    schema:
      - name: rideId
        data-type: BIGINT
      - name: lon
        data-type: FLOAT
      - name: lat
        data-type: FLOAT
      - name: rowTime
        data-type: TIMESTAMP(3)
        rowtime:
          timestamps:
            type: "from-field"
            from: "rideTime"
          watermarks:
            type: "periodic-bounded"
            delay: "60000"
      - name: procTime
        data-type: TIMESTAMP(3)
        proctime: true
```

使用Flink的下一步计划 —— 基于Flink的DL

将深度学习迁移至Flink框架: 解决实时的复杂计算问题



Thanks!

于茜

yuqian8@staff.weibo.com

微博机器学习研发中心