CCTC 2017 中国云计算技术大会
Cloud Computing Technology Conference 2017
CSDN

基于Spark的大规模机器学习在微博的应用

新浪微博
wulei3@staff.weibo.com

自我介绍

- 吴磊
- 现任职于新浪微博，负责计算框架设计&实现
- 曾就职于IBM、联想研究院，从事数据仓库、大数据应用

- 微博：小生活与大数据
- 微信：方块K
- 邮箱：wulei3@staff.weibo.com

议题

- 微博业务场景
- 大规模机器学习
- 微博机器学习框架

新浪微博

- 中国领先社交媒体平台
- 数据
  - MAU：3.40亿+（2017Q1）
  - DAU：1.54亿+（2017Q1）
  - 同比增长：30%左右
  - 用户分布：91％移动端
  - 刷新数：百亿级
  - 曝光数：千亿级

微博业务场景

业务场景复杂

业务场景多样性
(Feed, Hot, Rec, PUSH, Anti-spam, etc)

用户体量大　　高频访问

用户间关系纷杂

微博内容体量大

微博内容数据多样
(文本、图片、音频、视频，等)

特征类别多　　特征维度巨大

近百亿级别特征维度

近万亿级别样本量

算法模型多样化
(LR, SVM, GBDT, RF, NN, FP, FM,etc)

大规模机器学习

大规模机器学习

- 基于Spark Mllib的尝试&实践

经验之谈：

- Too many RDD union >> stackoverflow
- Driver out of memory >> spark.driver.maxResultSize
- Model AUC=0.5 >> lower learning rate
- Integer.MAX_VALUE >> partition.size less than 2G
- **Shuffle fetch failed >> spark.local.dir**
- **Shuffle fetch failed >> JVM GC adjustment**
- **Shuffle fetch failed >> spark.network.timeout**

挑战：

亿维特征空间

参数矩阵巨大

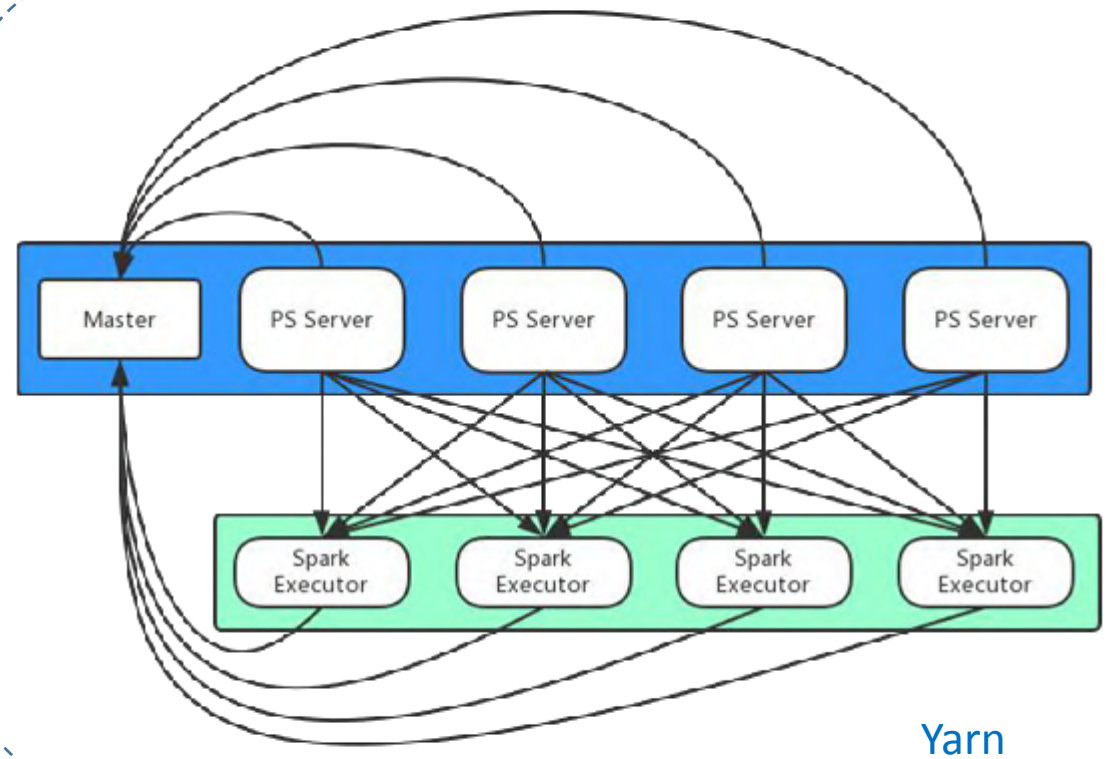
- 内存开销
- 网络开销

大规模机器学习

- 基于Spark的参数服务器

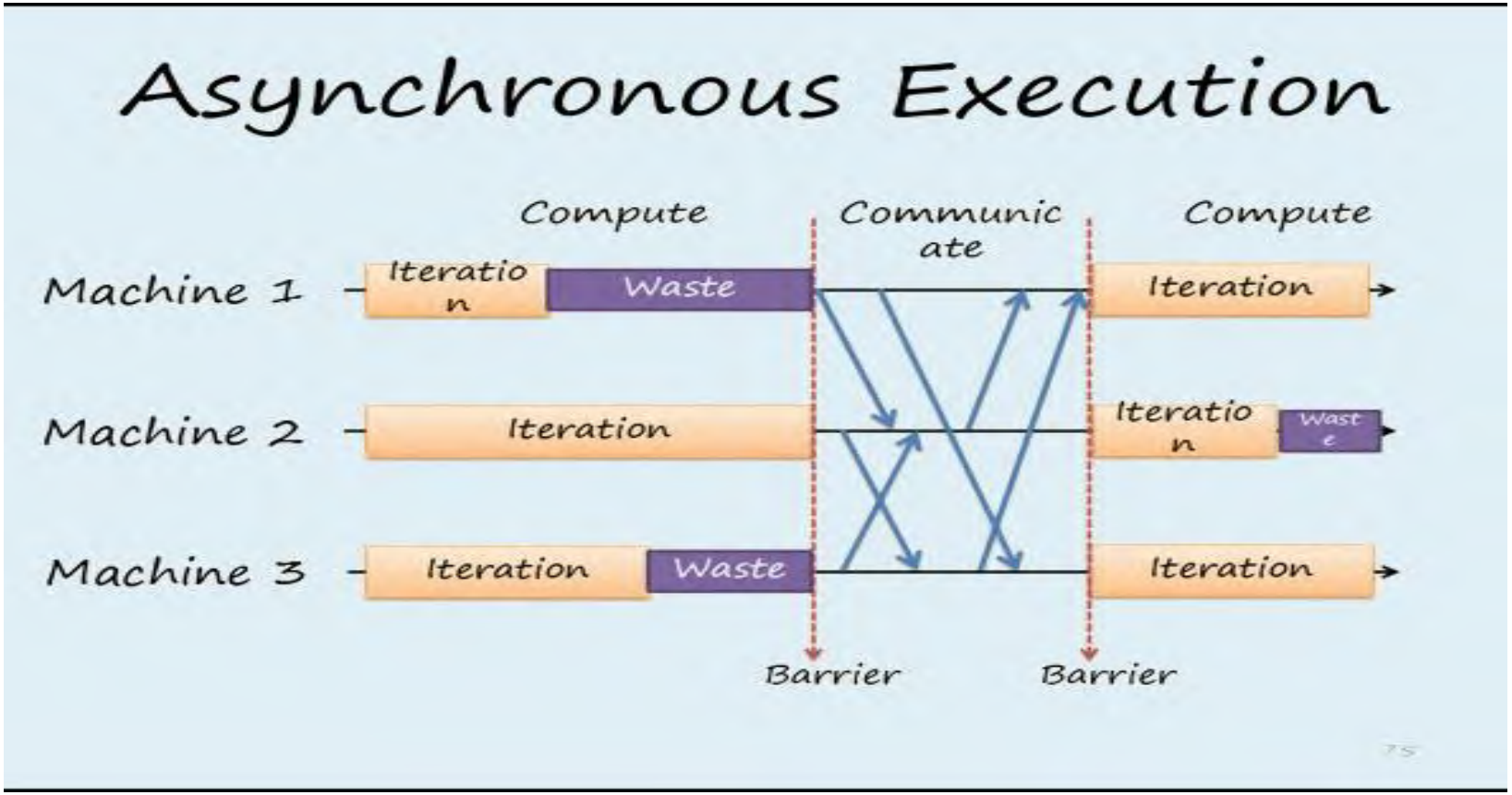
PS Server：
- 主从架构
- 服务化
- 梯度更新
- 权重更新
- 多参数副本

PS Client：
- Spark executors
- Summon PS actors
- 读取参数(PUSH)
- 拉取参数(PULL)



大规模机器学习

- 基于Spark的参数服务器

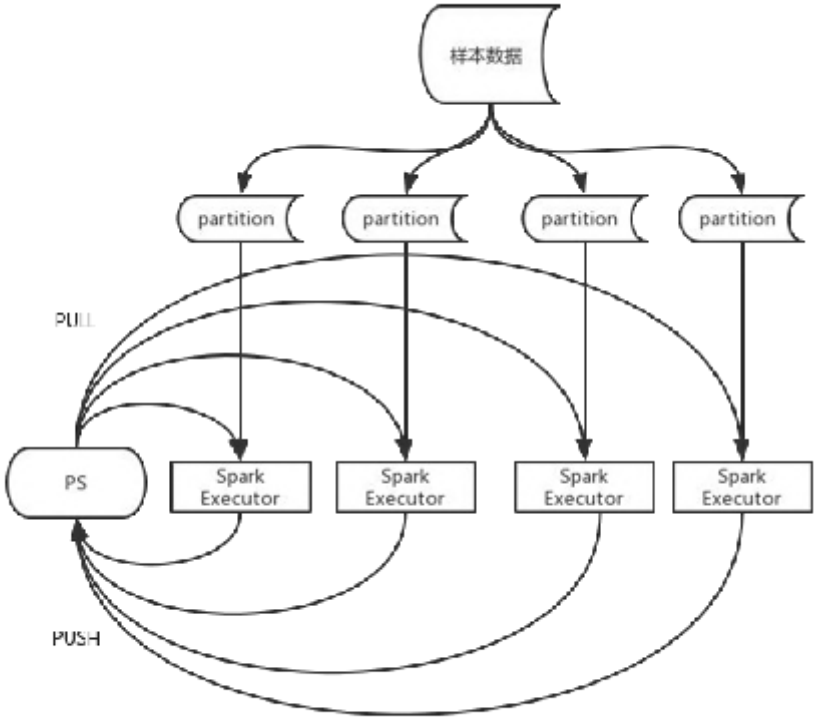


大规模机器学习

- 基于Spark的参数服务器

异步随机梯度下降算法

- 将数据分块
- 分块数据拉取参数
- 计算分块数据梯度
- 将梯度更新到参数服务器
- 重复上述操作



大规模机器学习

- 基于Spark的参数服务器

性能优化：

- Batch Size
- PS server count
- Sparse
- Partitioning
- Spark memory tuning

大规模机器学习

- 基于Spark的参数服务器

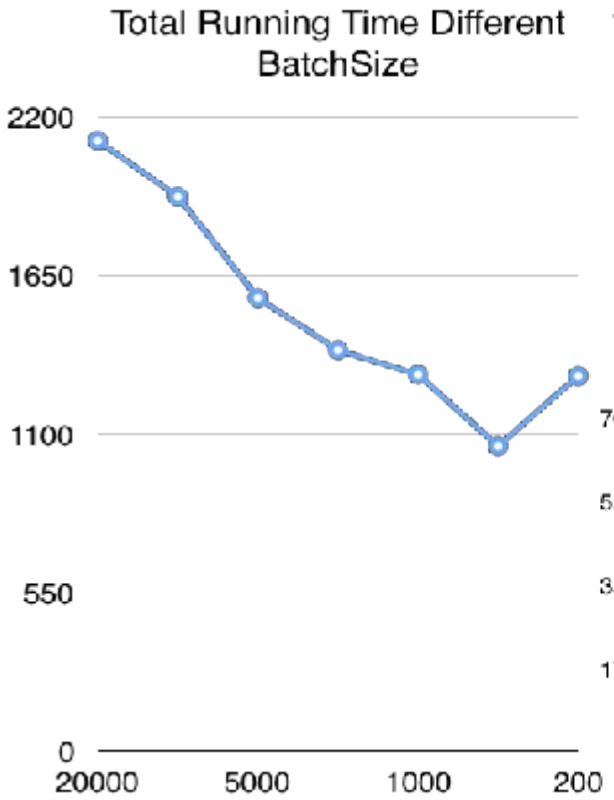
优化实例： Batch Size对性能的影响

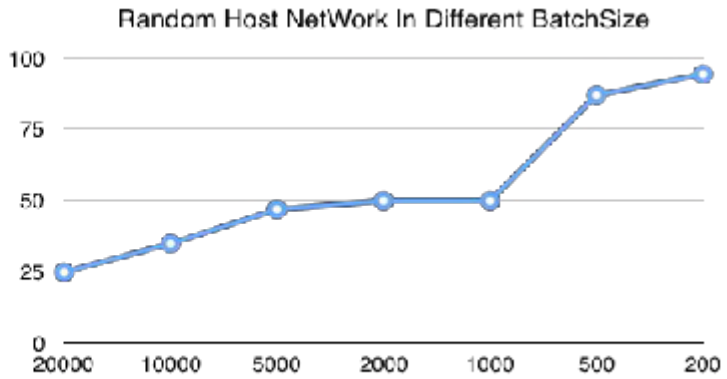
BatchSize	Parameter(MB)	Tx(MB)	Pull(ms)	Push(ms)	Time(s)
20000	60	25	1925.91	6868.88	2118
10000	30	35	862.373	3013.54	1924
5000	15	47	300	1500	1573
2000	6	50	98	404	1392
1000	3	50	55.56	199.79	1307
500	**1.5**	**87**	**63.95**	**193.22**	**1059**
200	0.6	94.3	87.64	176.587	1302

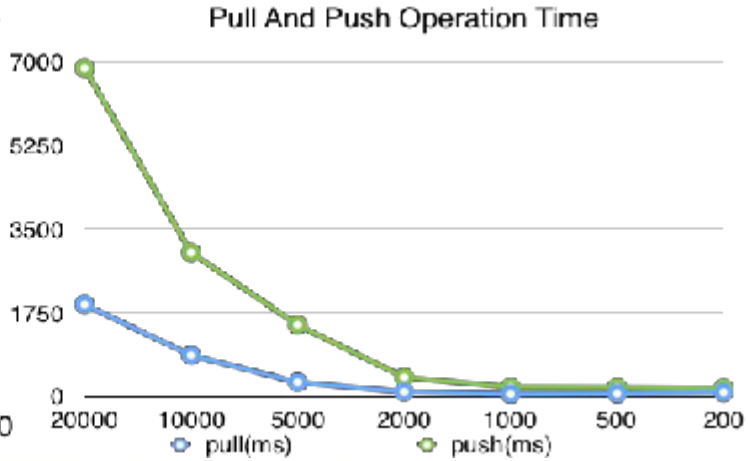
大规模机器学习

- 基于Spark的参数服务器

优化实例：
Batch Size







大规模机器学习

- 基于Spark的参数服务器

新的挑战：

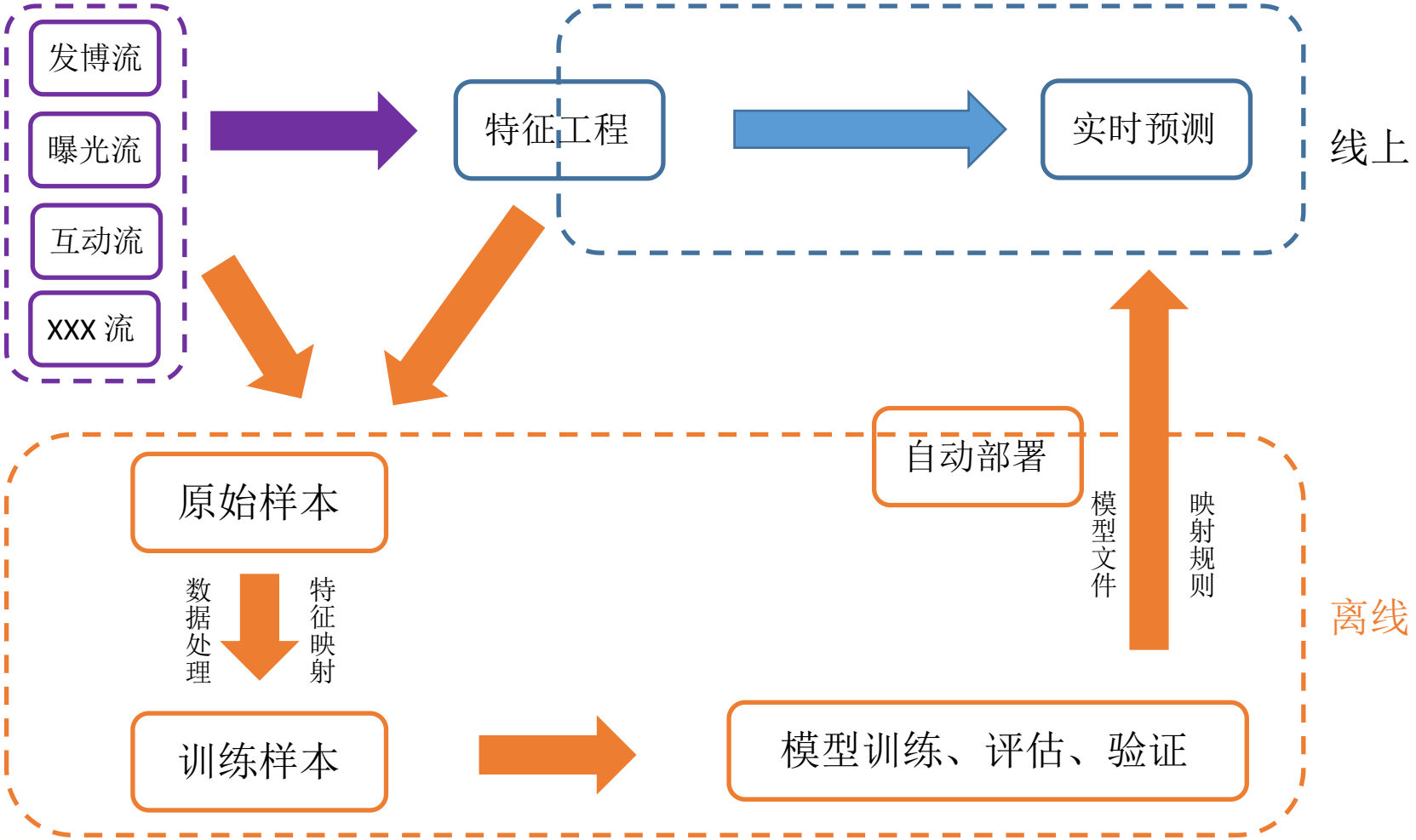
- 更多的算法支持
- 性能优化
- 半同步尝试

What's next?

- Contained with Docker
- 通过ZooKeeper实现配置化

微博机器学习框架

机器学习流图


发博流
曝光流
互动流
XXX 流
特征工程
实时预测
线上
原始样本
数据处理
特征映射
训练样本
模型训练、评估、验证
自动部署
模型文件
映射规则
离线

现存挑战&问题

迭代上线

业务开发流程冗长

业务脚本调用混乱

python tensorflow

计算框架多样性 redis

storm hive hadoop

系统运行门槛高 spark

算法多样性

模型多样性

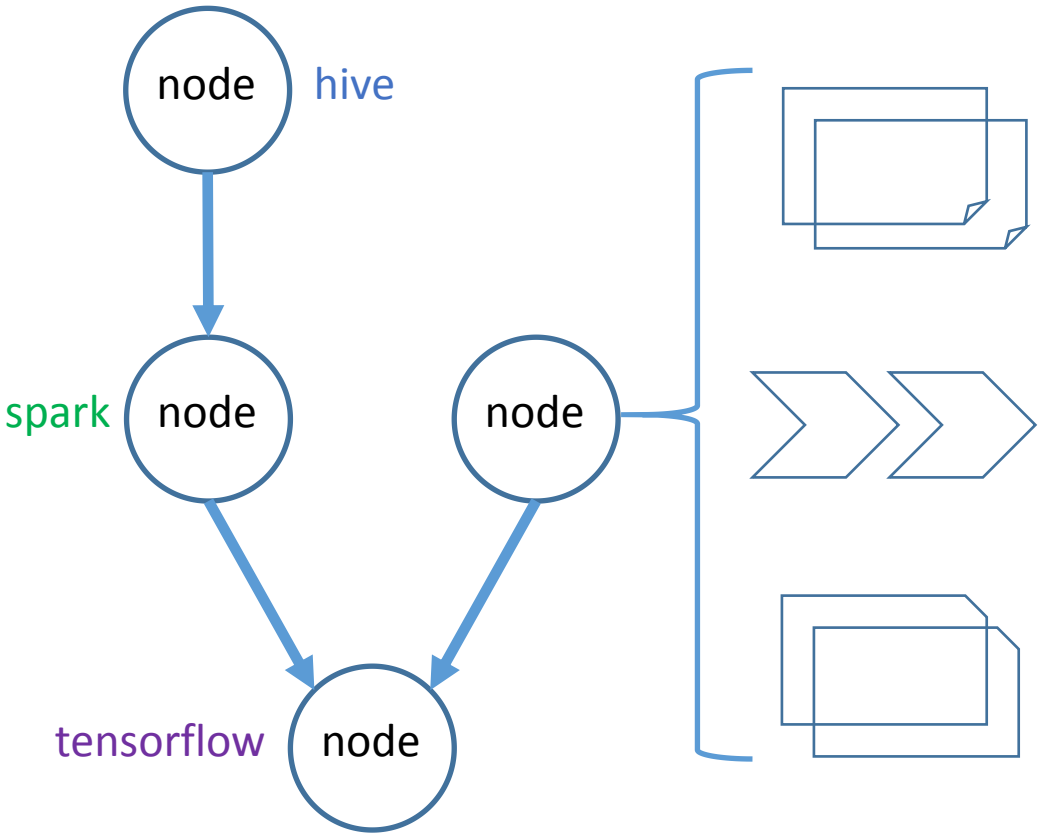
特征、数据处理繁琐

平台化需求

执行性能差

沟通效率低

weiflow 统一计算框架



- Text
- Csv
- Libsvm
- Parquet
- Orc
- Json

- 统计
- 清洗
- 过滤
- 组合
- 采样
- 转换
- 。。。

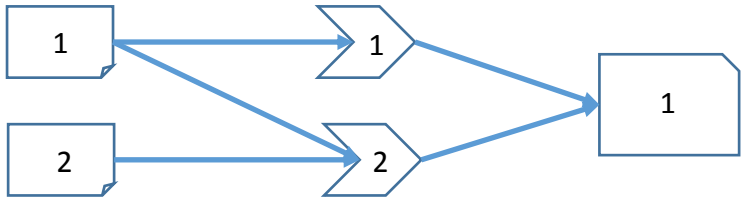
One XML to rule them all

```
<weiflow>

  <node id="1" preid="-1">HiveDataJoin</node>
  <node id="2" preid="-1">DataFilteringAndSampling</node>
  <node id="3" preid="-1">RunShellCommand</node>

</weiflow>
```

Node DAG构造

Node内input、process、output

```
<input name="input1">
    <className>com.weibo.datasys.dataflow.input.InputSparkText</className>
    <dataPath>hdfs://path/of/your/data</dataPath>
    <metaPath>/path/of/your/meta</metaPath>
    <fieldDelimiter>\u0001</fieldDelimiter>
</input>
<input name="input2">
    <className>com.weibo.datasys.dataflow.input.InputSparkParquet</classNa
    <dataPath>hdfs://path/of/your/data</dataPath>
</input>

<process name="process1">
    <className>com.weibo.datasys.dataflow.process.ProcessSparkDataFilter</
    <dependency>input1</dependency>
    <command type="sql">select * from table where age < 18 and device = 'ip
</process>
<process name="process2">
    <className>com.weibo.datasys.dataflow.process.ProcessSparkDataJoin</cla
    <dependency>input1,input2</dependency>
    <command type="sql">select T1.*, T2.expo, T2.act from T1 inner join T2
</process>

<output name="output1">
    <className>com.weibo.datasys.dataflow.output.OutputSparkText</classNam
    <dependency>process1, process2</dependency>
    <dataPath>hdfs://path/of/your/data</dataPath>
    <metaPath>/path/of/your/meta</metaPath>
    <fieldDelimiter>;</fieldDelimiter>
</output>
```



weiflow 开发API

支持语言：

- Scala
- Java

```scala
trait ProcessSpark extends ProcessBase {

  def init(userConf: Map[String, String]): Unit
  def transform(spark: SparkSession, df: Array[AnyRef]): AnyRef

}
```

```scala
object ProcessSparkDataJoin extends ProcessSpark {
```

```java
public final class ProcessSparkLRWithDataFrame implements ProcessSpark {
```

weiflow 实现

框架部分：

XML解析

JVM反射

Scala语言特性

业务部分：

- Array >> HashMap
- map >> mapPartitions
- Dense >> Sparse
- Currying、Partial functions
- Busy driver
- Broadcast variables
- Spark SQL
- Spark ML
- DataFrame

weiflow 功效

执行性能(6T,1000亿)
20hours >> 20mins

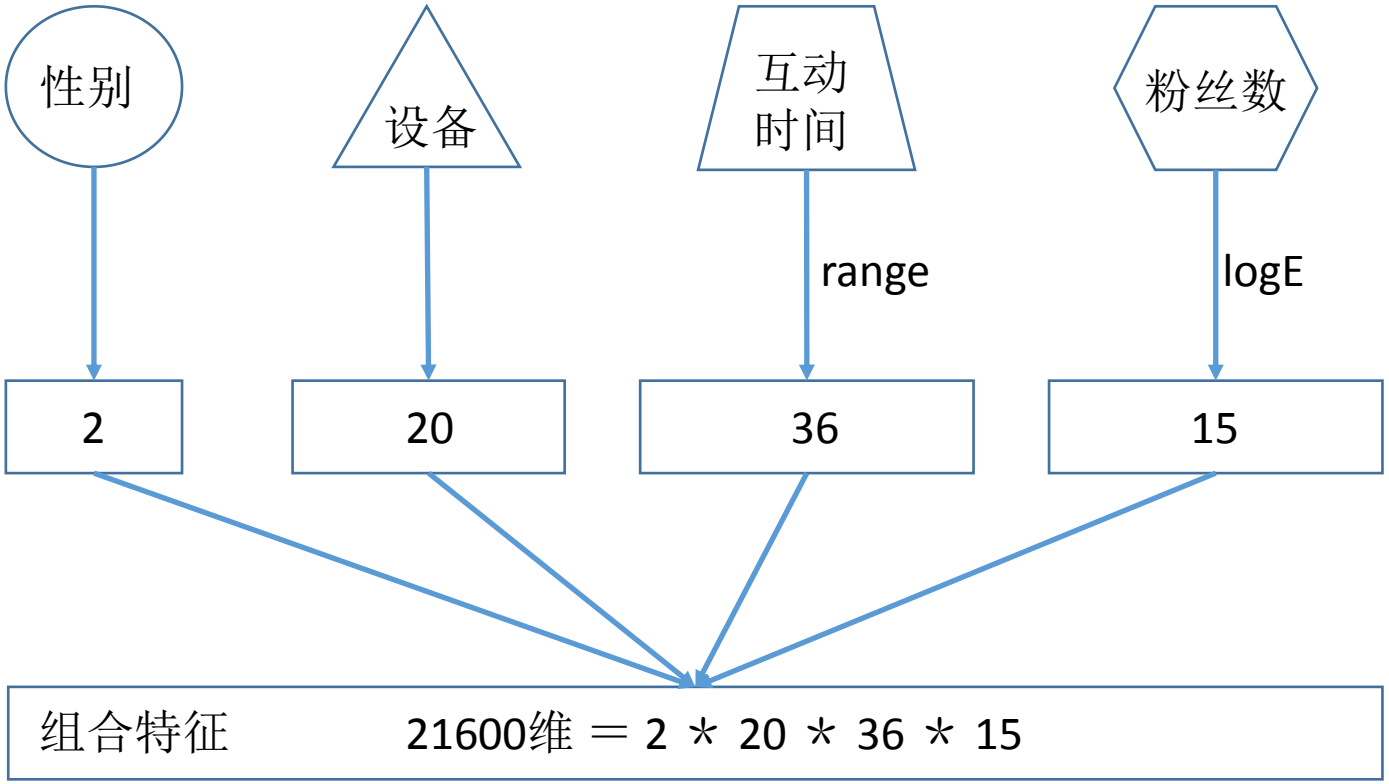
业务开发效率
提升显著

模型性能(GBDT+LR)
AUC up 0.05～0.1

业务代码贡献
更加活跃

业务部门
沟通效率提升

weiflow 应用(特征组合)



weiflow 应用(特征组合)

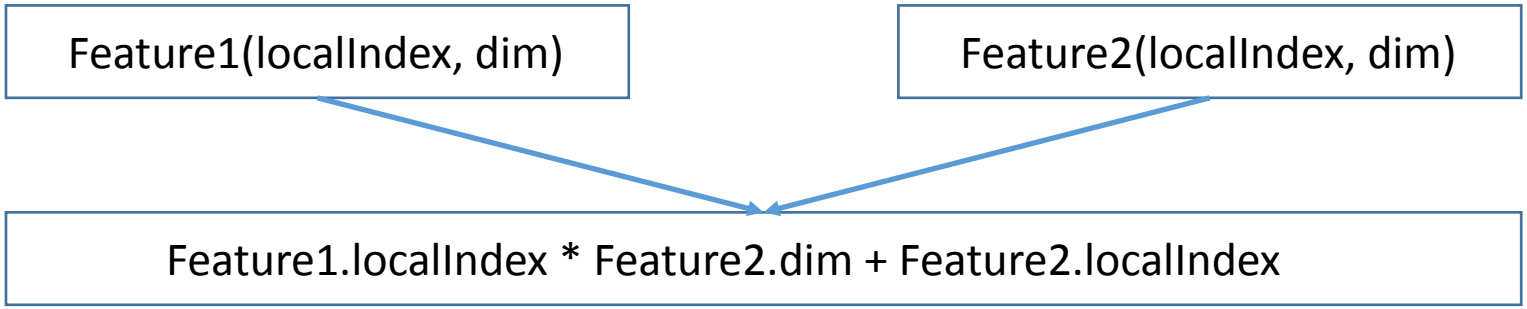
组合特征的使用

featureIndex@featureName@mapType@operator@args

16@combinedFeauture1@enum@**cartesian**@**f1+f2+f3+…+fN**

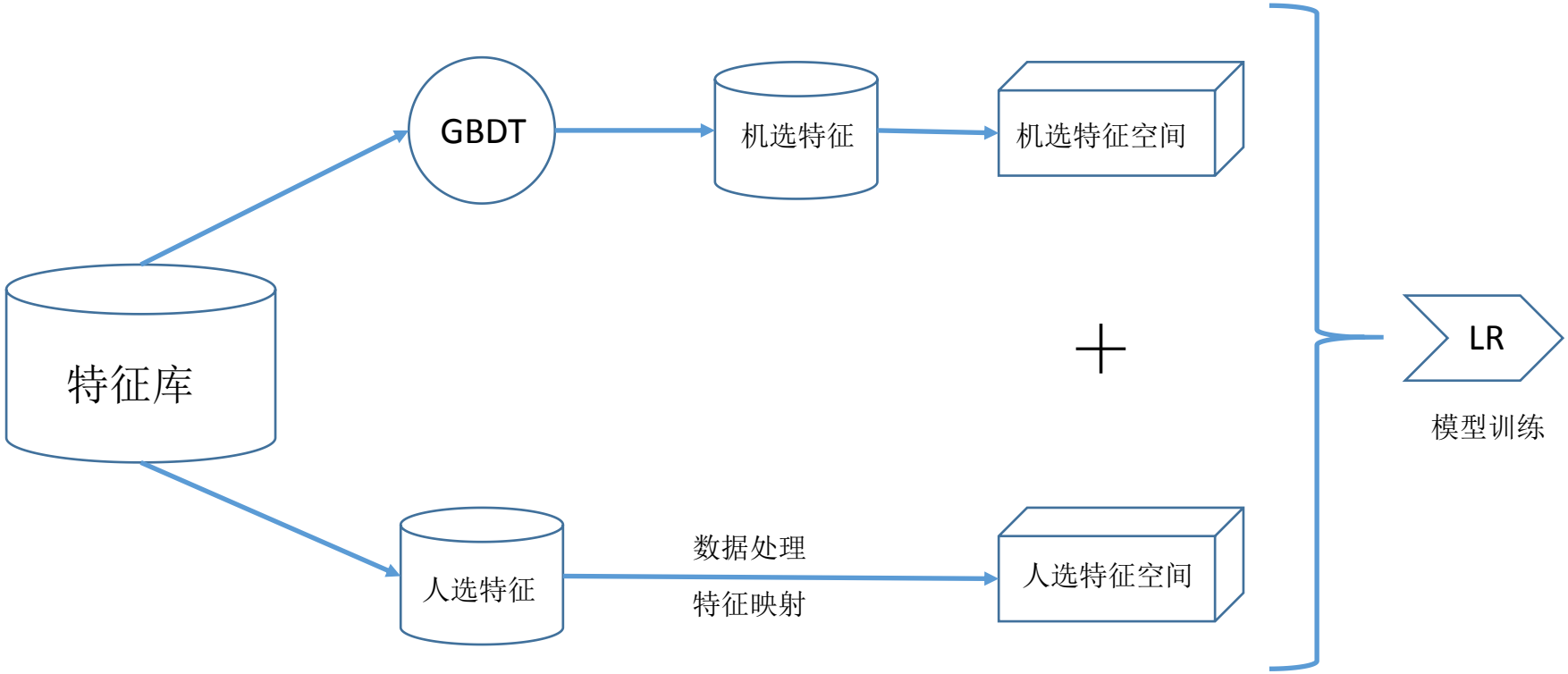
weiflow 应用(特征组合)

组合特征的实现



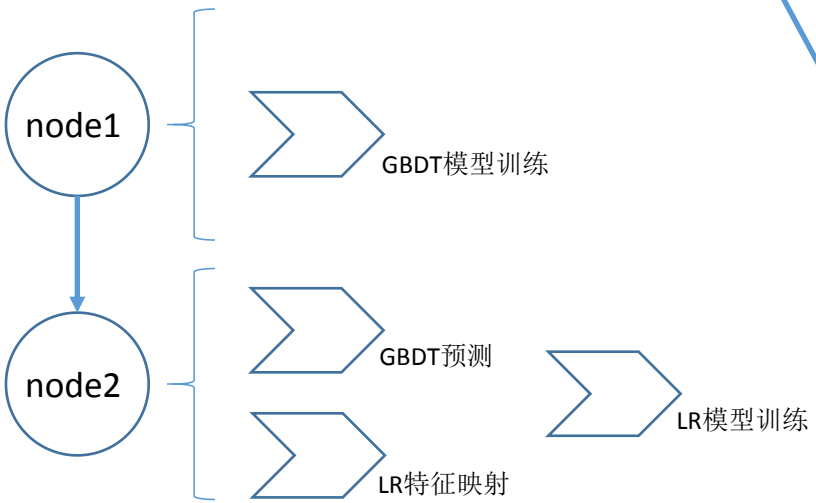

```
val localIndexAndMaxDim: Array[(Long, Long)] = new Array[(Long, Long)](subElemNum)
/* Populate your localIndexAndMaxDim,
   this data structure contains all pairs of
   local indexes and feature dimensions of
   all the composing features.
 */
localIndexAndMaxDim.reduceLeft((f1, f2) => compoundCross(f1, f2))
```

weiflow 应用(GBDT+LR)



weiflow 应用(GBDT+LR)

```
<weiflow>

  <node id="1" preid="-1">GBDTtraining</node>
  <node id="2" preid="1">GBDTplusLR</node>

</weiflow>
```

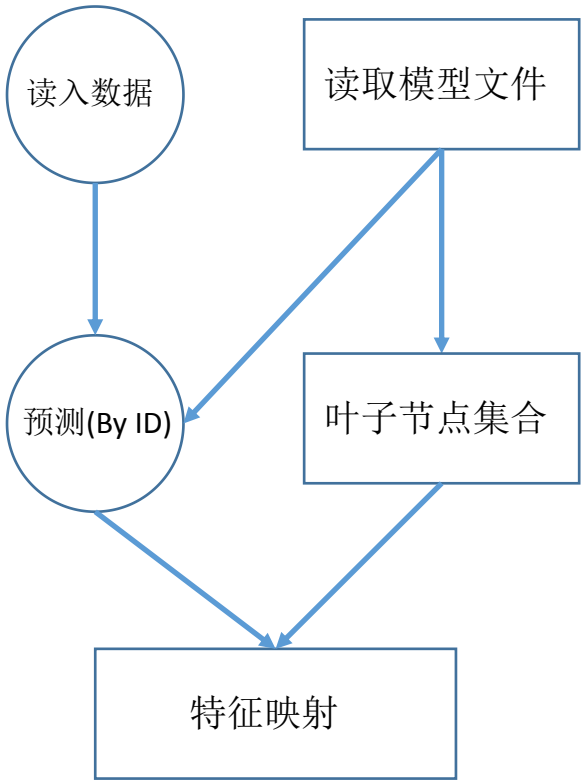



```
<input name="input1">
    <className>com.weibo.datasys.dataflow.input.InputSparkText</className>
    <dataPath>hdfs://path/of/your/data</dataPath>
    <metaPath>/path/of/your/meta</metaPath>
    <fieldDelimiter>\t</fieldDelimiter>
</input>

<process name="process1">
    <className>com.weibo.datasys.dataflow.process.ProcessSparkGBDTTraining</className>
    <dependency>input1</dependency>
    <conf>gbdt.data.conf</conf>
</process>

<output name="output1">
    <className>com.weibo.datasys.dataflow.output.OutputSparkGBDTModel</cla
    <dependency>process1</dependency>
    <modelPath>hdfs://path/of/your/data</dataPath>
</output>
</node>
```

```
<input name="input1">
    <className>com.weibo.datasys.dataflow.input.InputSparkText</className>
    <dataPath>hdfs://path/of/your/data</dataPath>
    <metaPath>/path/of/your/meta</metaPath>
    <fieldDelimiter>\t</fieldDelimiter>
</input>

<process name="process1">
    <className>com.weibo.datasys.dataflow.process.ProcessSparkGBDTPredict</
    <dependency>input1</dependency>
    <conf>gbdt.data.conf</conf>
    <model>/path/to/your/model</model>
</process>
    <process name="process2">
    <className>com.weibo.datasys.dataflow.process.ProcessSparkLRfeatureMap
    <dependency>input1</dependency>
    <conf>lr.feature.map</conf>
</process>
<process name="process3">
    <className>com.weibo.datasys.dataflow.process.ProcessSparkLRtraining</
    <dependency>process1,process2</dependency>
</process>

<output name="output1">
    <className>com.weibo.datasys.dataflow.output.OutputSparkLRModel</classN
    <dependency>process3</dependency>
    <modelPath>hdfs://path/of/your/data</dataPath>
</output>
</node>
```

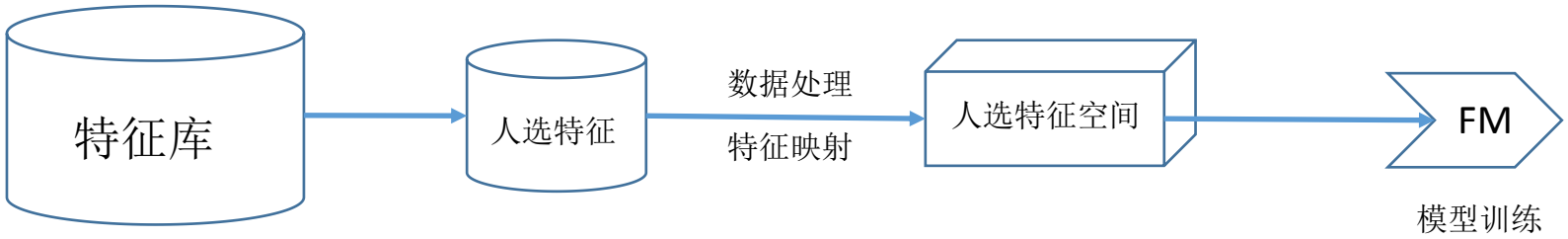
weiflow 应用(GBDT+LR)

GBDT预测/映射



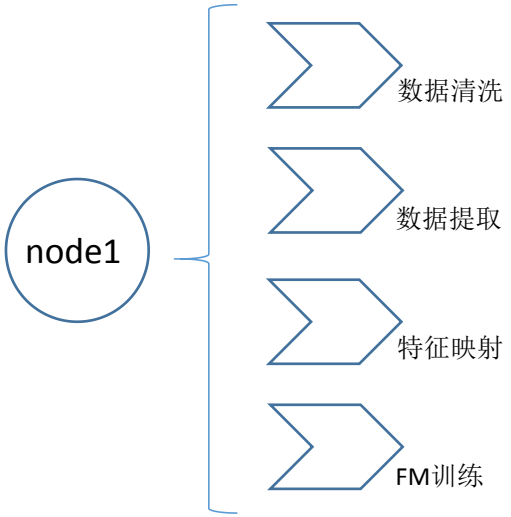

```
class NodeUtil (override val id: Int,
                var predictUtil: Predict,
                var impurityUtil: Double,
                val isLeafUtil: Boolean,
                val splitUtil: Option[Split],
                val leftNodeUtil: Option[Node],
                val rightNodeUtil: Option[Node],
                val statsUtil: Option[InformationGainStats])
  extends Node(id, predictUtil, impurityUtil, isLeafUtil, splitU

  override def toString: String = {...}

  def predictId(features: Vector) : Int = {...}

  def getLeafNodeId(features: Vector) : Int = {
    if (isLeaf) {
      return id
    } else {
      if (split.get.featureType == Continuous) {
        if (features(split.get.feature) <= split.get.threshold)
          val leftNodeUtil = new NodeUtil(leftNode.get.id, leftN
            leftNode.get.split, leftNode.get.leftNode, leftNode.
          return leftNodeUtil.getLeafNodeId(features)
        } else {
          val rightNodeUtil = new NodeUtil(rightNode.get.id, rig
            rightNode.get.split, rightNode.get.leftNode, rightNo
          return rightNodeUtil.getLeafNodeId(features)
        }
      } else {
```

weiflow 应用(FM)



weiflow 应用(FM)

```
<weiflow>
  <node id="1" preid="-1">FMTraining</node>
</weiflow>
```




```
<input name="input1">
    <className>com.weibo.datasys.dataflow.input.InputSparkText</className>
    <dataPath>hdfs://path/of/your/data</dataPath>
    <format>parquet</format>
</input>
<process name="process1">
    <className>com.weibo.datasys.dataflow.process.ProcessSparkDataClean</c
    <dependency>input1</dependency>
    <dataPath>clean.spec</dataPath>
</process>
<process name="process2">
    <className>com.weibo.datasys.dataflow.process.ProcessSparkDataExtract</
    <dependency>process1</dependency>
    <dataPath>extract.spec</dataPath>
</process>
<process name="process3">
    <className>com.weibo.datasys.dataflow.process.ProcessSparkFeatureMappi
    <dependency>process2</dependency>
    <dataPath>feature.map</dataPath>
</process>
<process name="process4">
    <className>com.weibo.datasys.dataflow.process.ProcessSparkFMTrain</cla
    <dependency>process3</dependency>
    <dataPath>fm.model</dataPath>
</process>
```

You are WANTED!

微博算法平台

- 分布式系统研发
- 算法系统研发
- 深度学习系统研发

联系方式

- 微博：小生活与大数据
- Email：wulei3@staff.weibo.com

以微博之力，让世界更美！