

# 美团外卖广告模型演进之路

谢乾龙  
2018.11

外卖广告

品牌广告



CPM

推荐广告



CPC

搜索广告



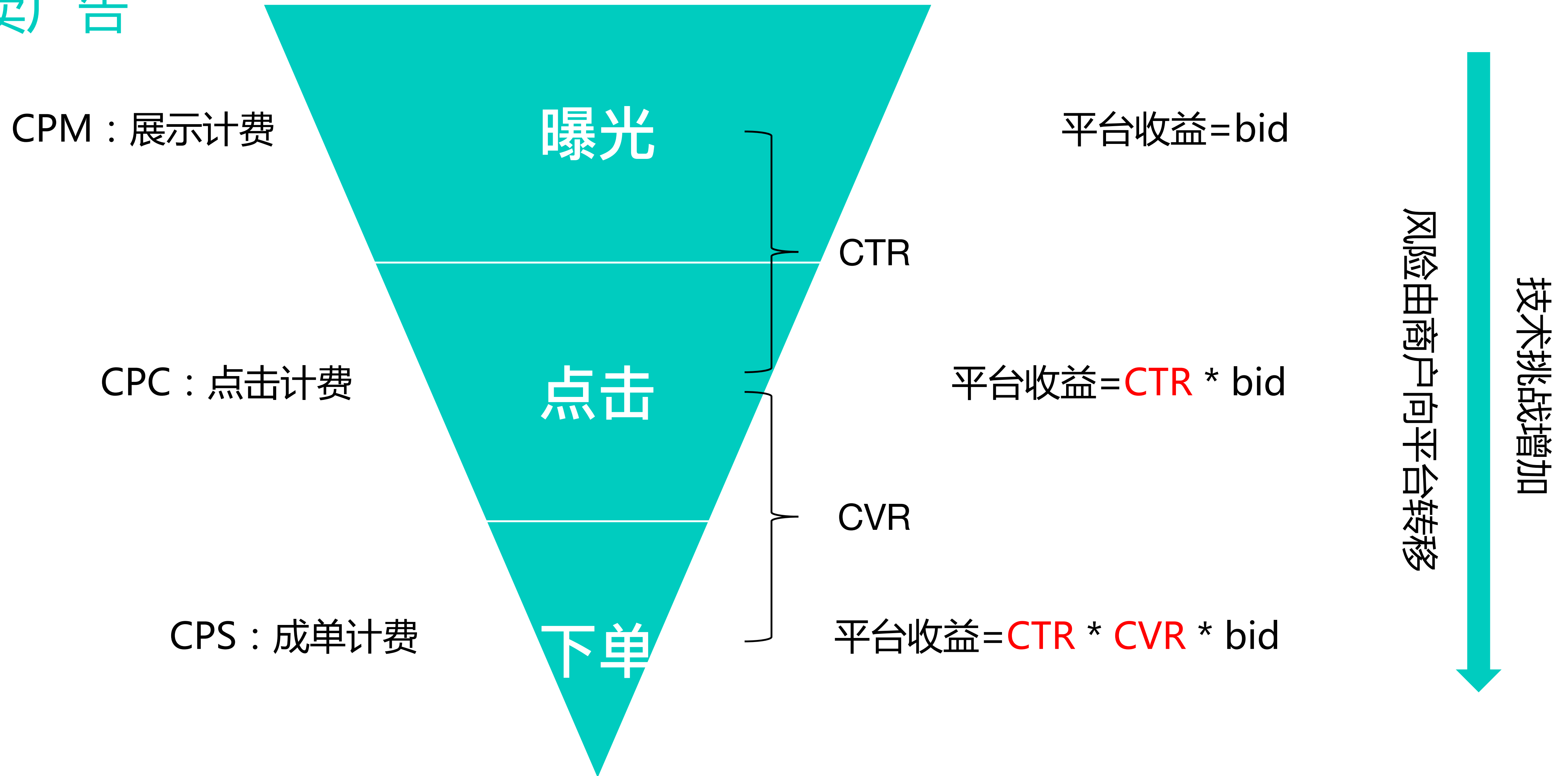
CPC

营销广告

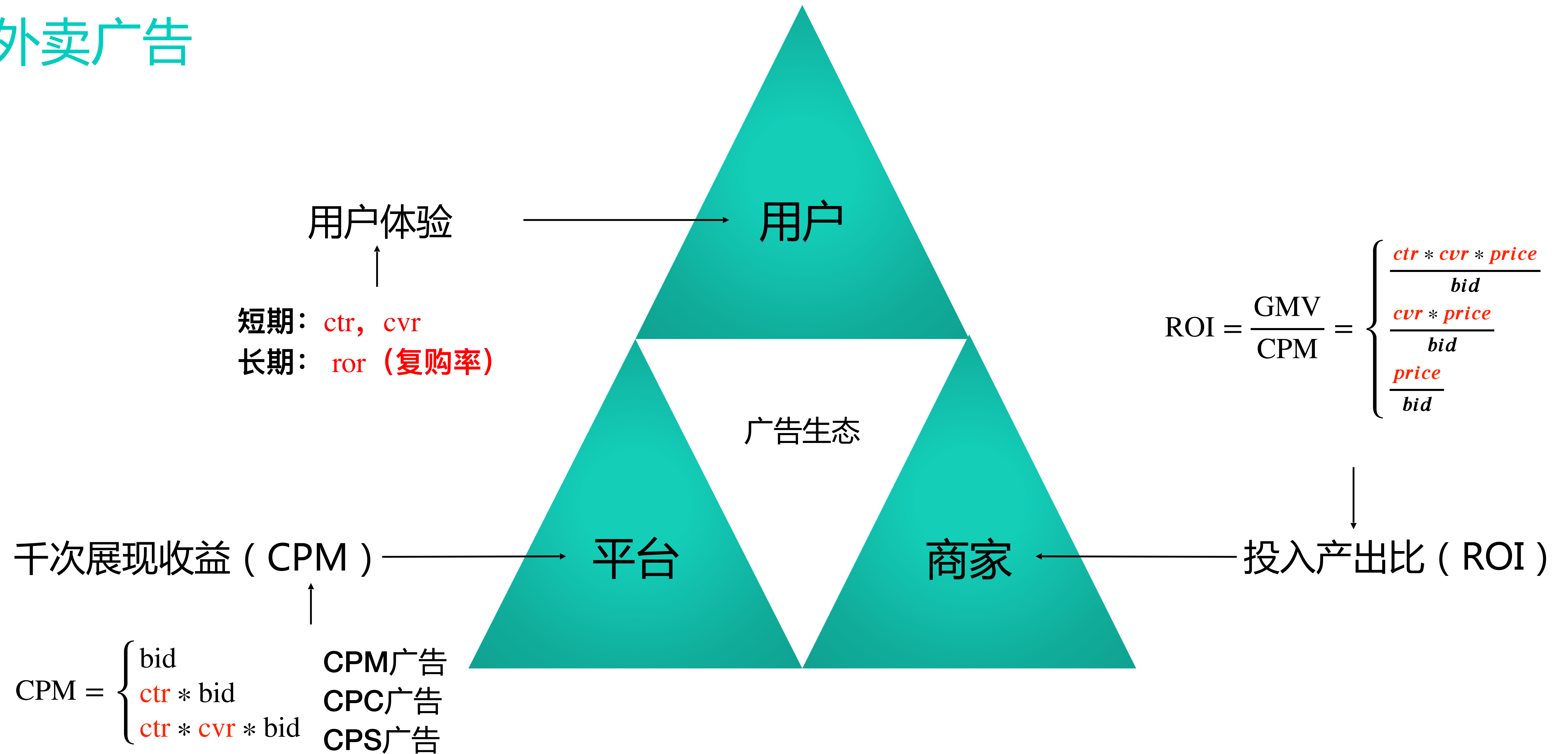


CPS

# 外卖广告



# 外卖广告

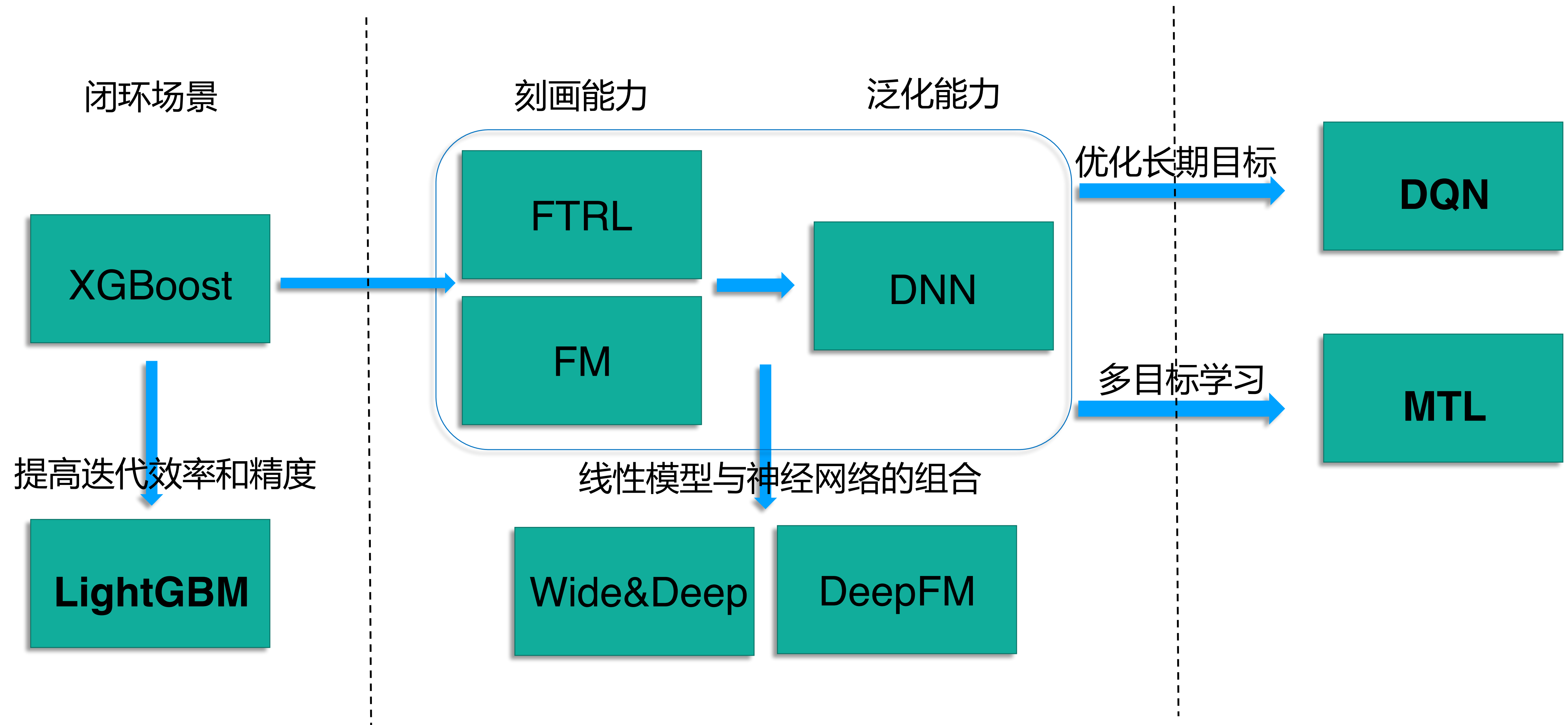




# 业界模型发展历程



# 外卖广告模型演进



# 目录

## 树模型

- XGBoost
- LightGBM

## 神经网络

- DNN
- MTL

## 强化学习

- DQN

# 目录

## 树模型

- XGBoost
- LightGBM

## 神经网络

- DNN
- MTL

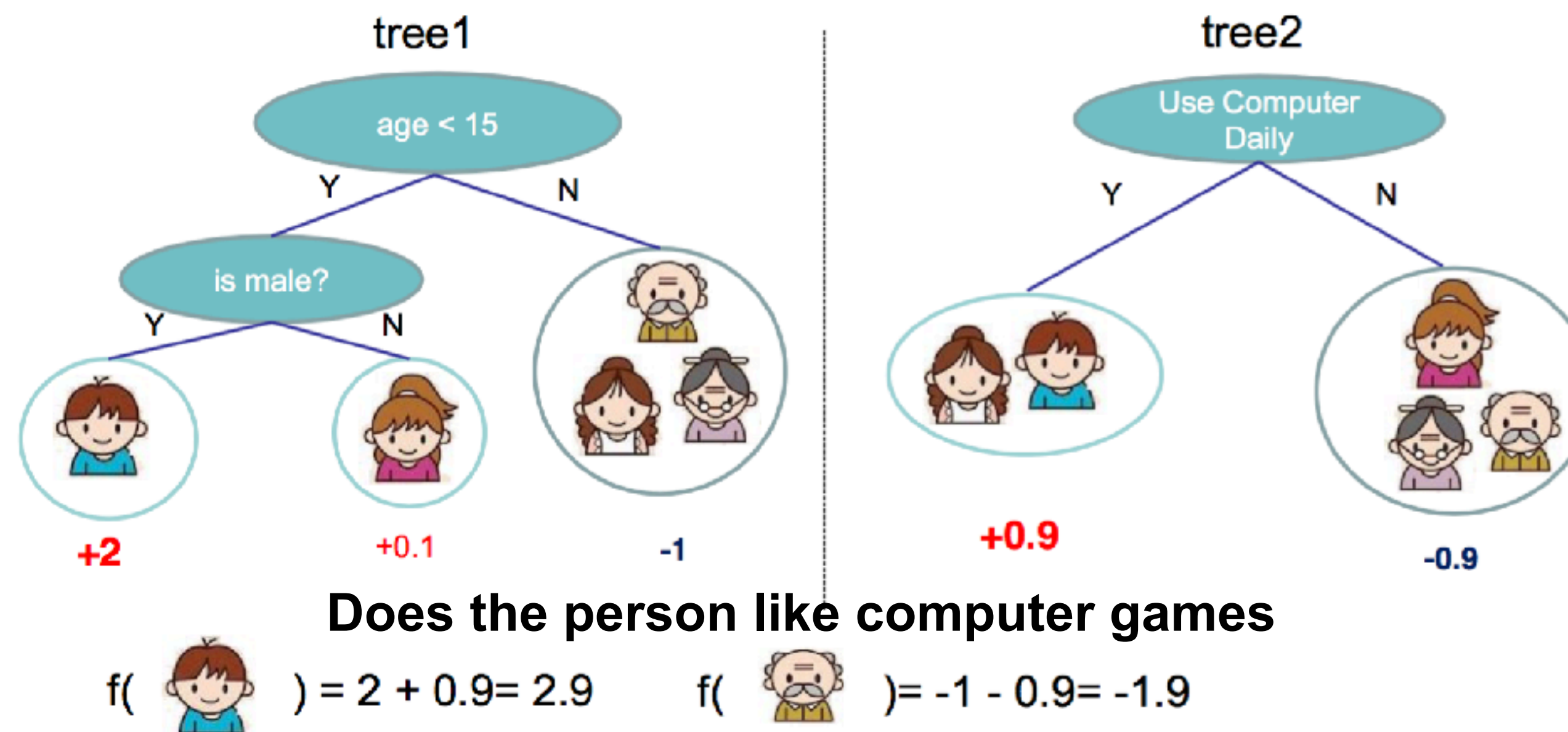
## 强化学习

- DQN



# GBDT & XGBoost

## 01 GBDT: Gradient Boosting Decision Tree



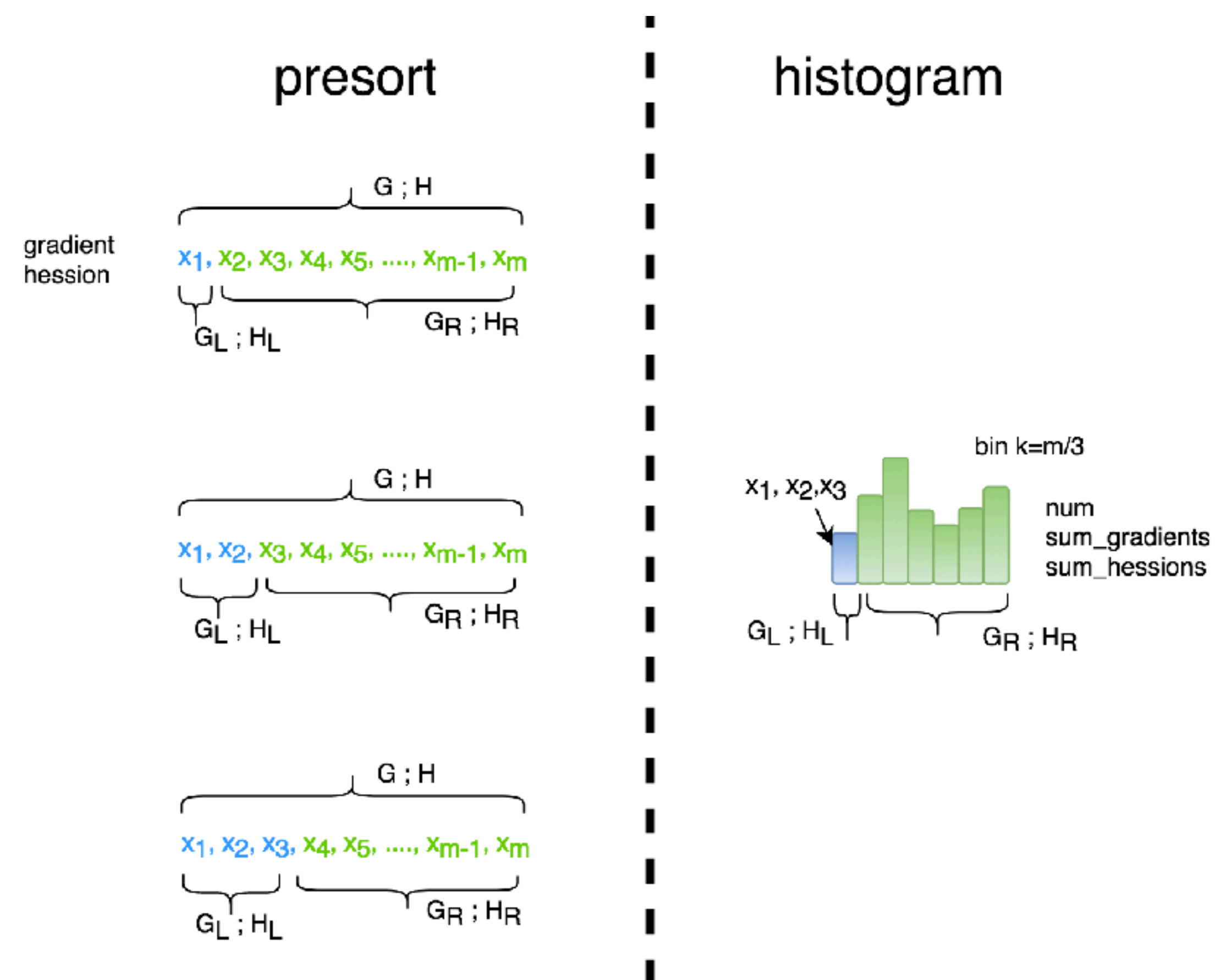
## 02 XGBoost: eXtreme Gradient Boosting

XGBoost是GBDT的一种实现

# XGBoost

## • XGBoost的实现方式

- 1, 二阶泰勒展开：损失函数更精准，收敛速度更快
- 2, 引入正则项：防止过拟合
- 3, 缺失值处理：自动学习特征缺失值的分裂方向
- 4, 直方图算法：如右图



# LightGBM

## • LightGBM改进

### 精度提升

- Leaf-wise分裂

### 性能优化

- 样本采样 ( GOSS )
- 特征合并 ( EFB )
- 分布式通信优化

### 易用性

- 支持类别特征
- 支持忽略特征

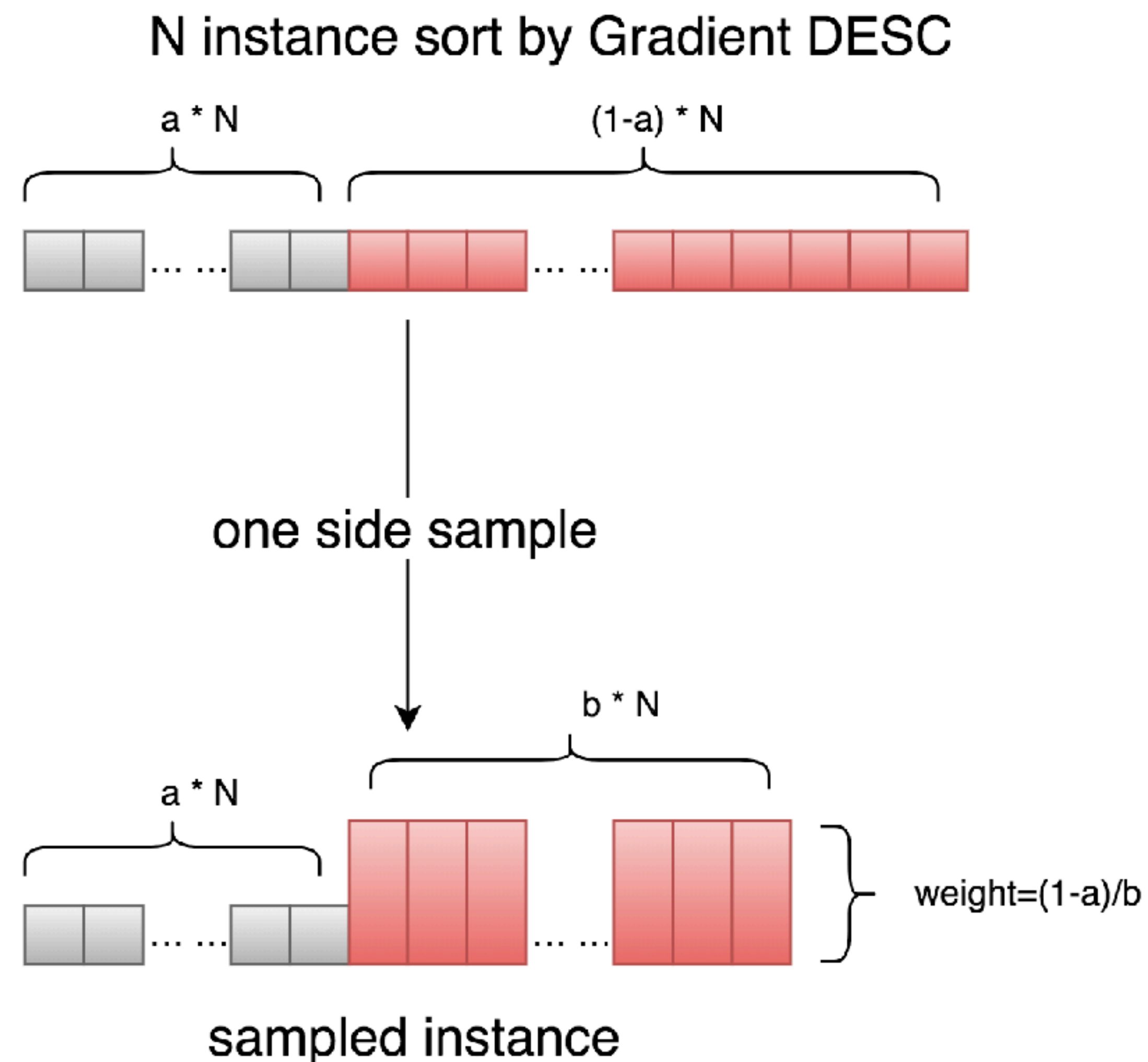
# LightGBM—性能优化

## • GOSS ( Gradient-based One-Side Sampling )

保留梯度大的样本，采样梯度小的样本，并提高其权重

## • 优点

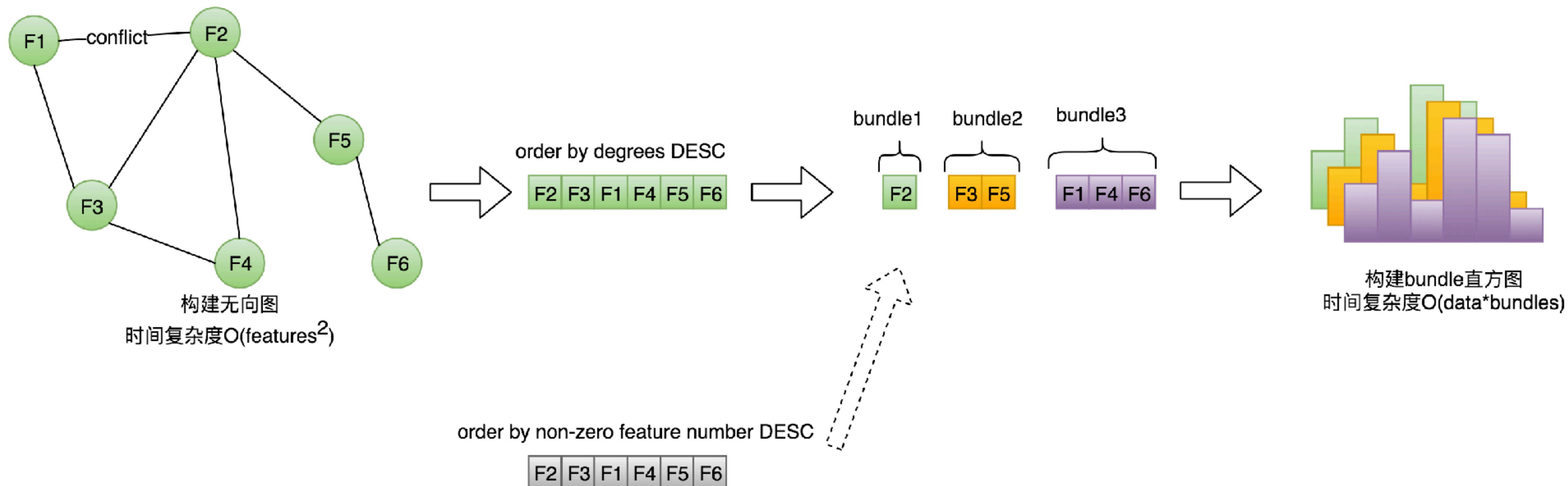
既减少了数据量，又能有效保持精度



# LightGBM—性能优化

## • EFB ( Exclusive Feature Bundling )

建立直方图的时间复杂度可以由  $O(\#data * \#feature)$  减小到  $O(\#data * \#bundle)$

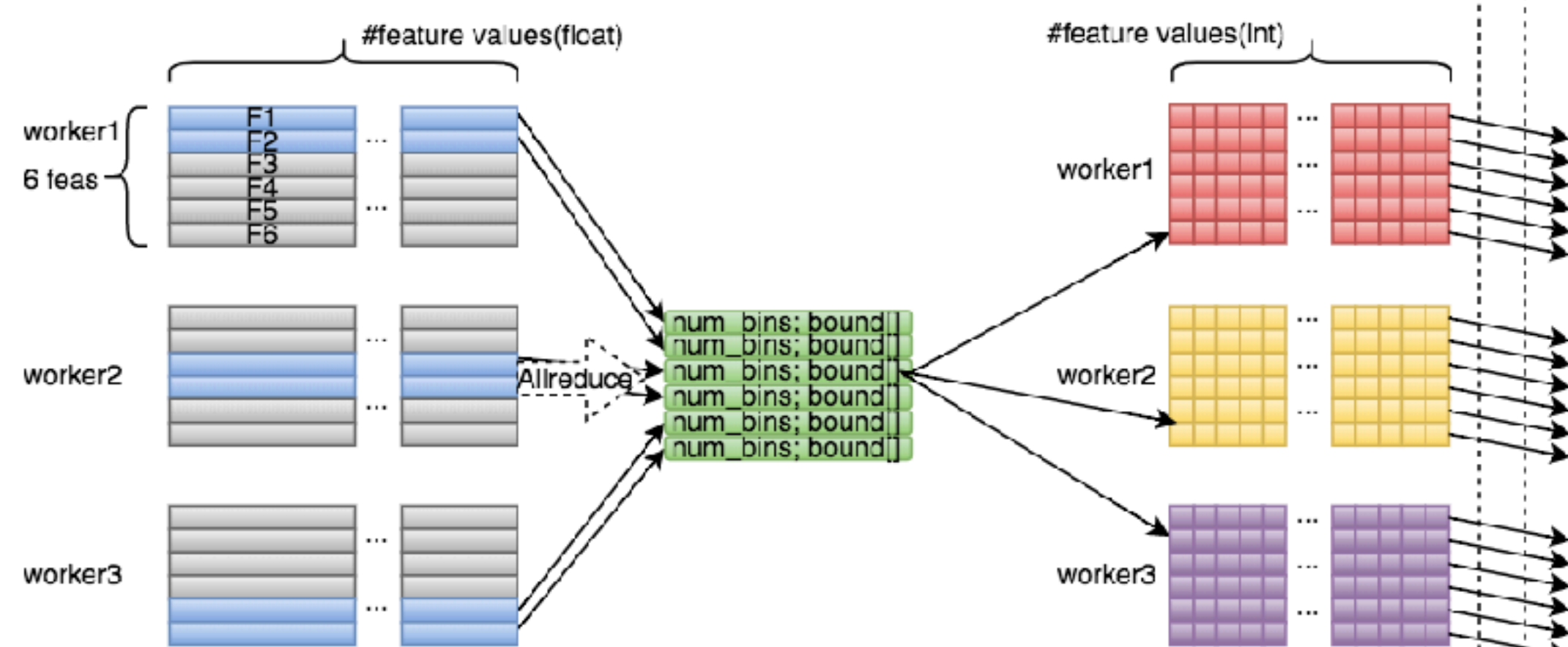




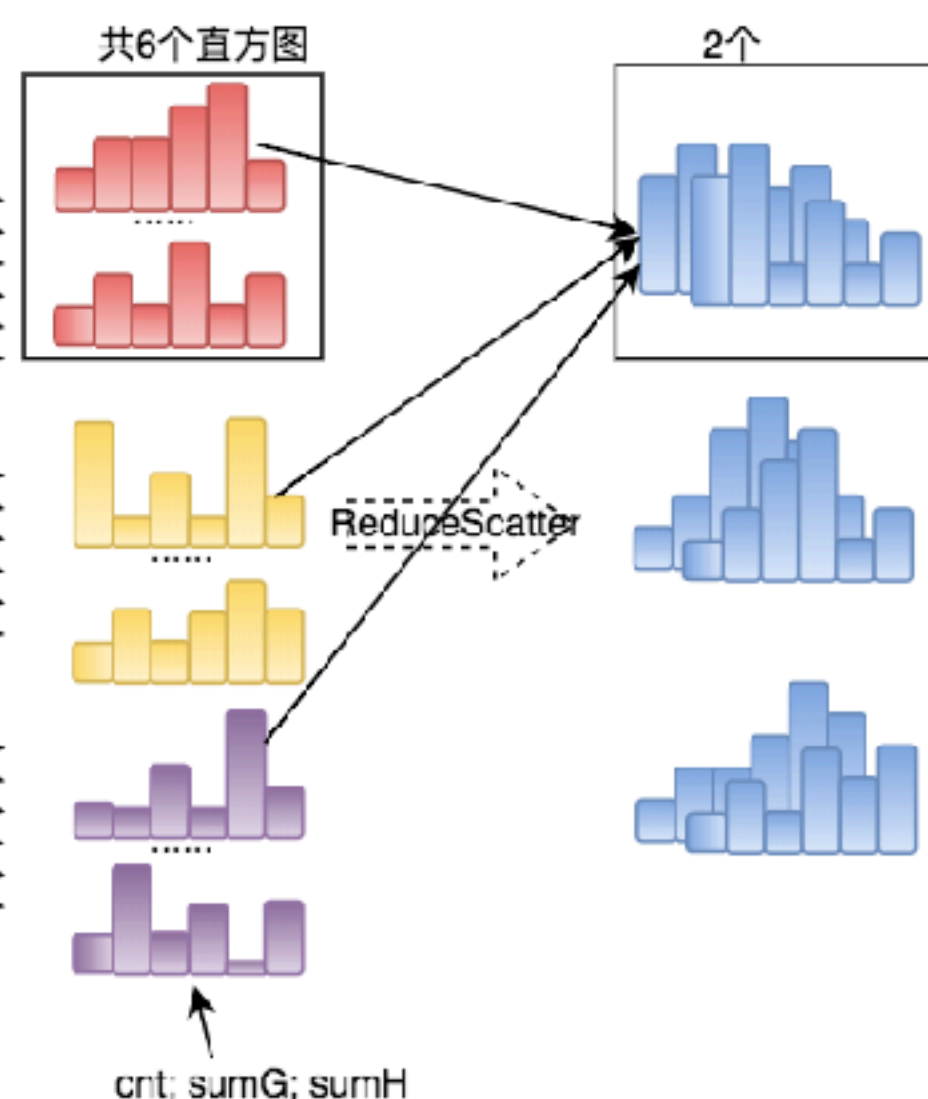
# LightGBM—并行化

## • LightGBM中的并行

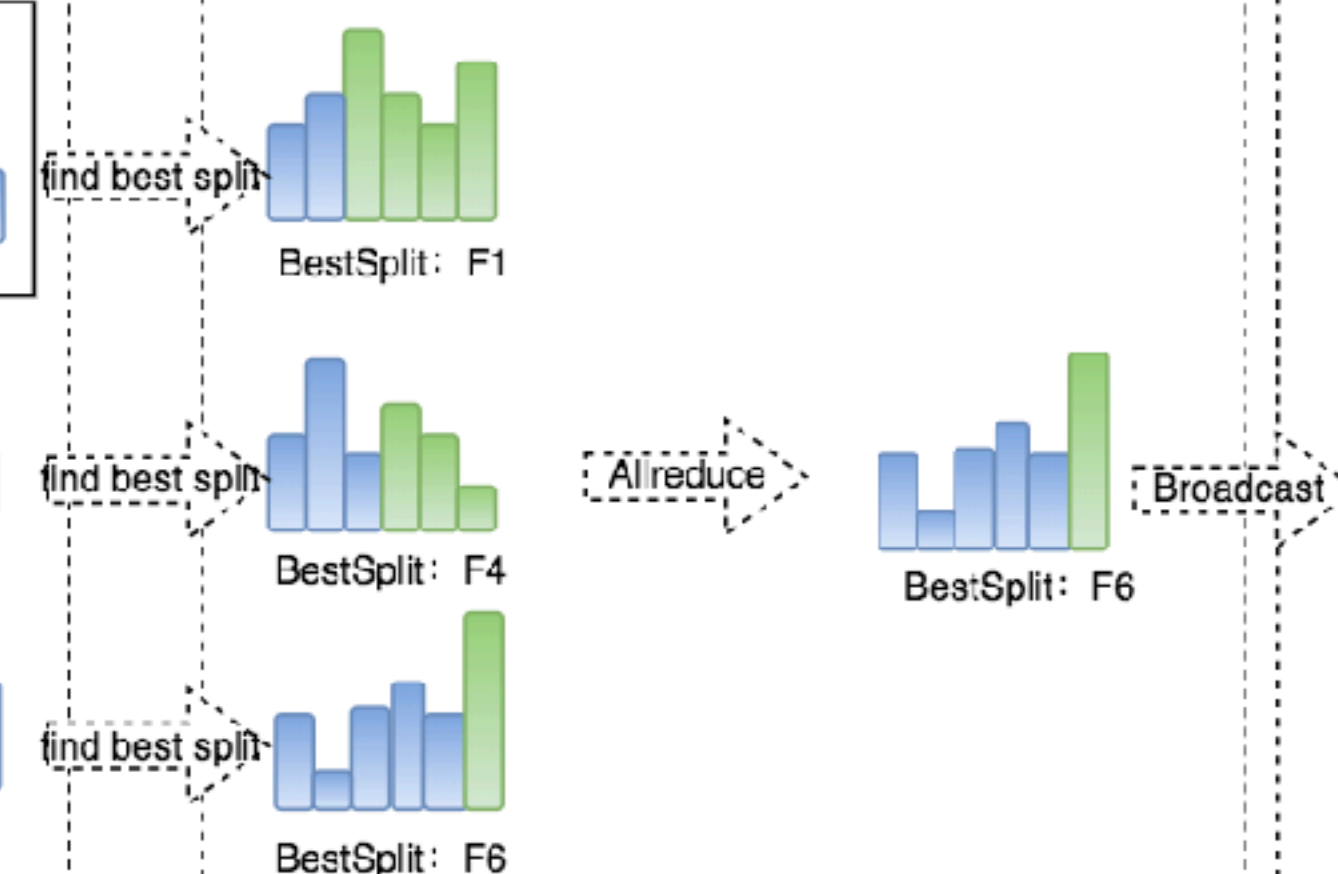
### 1, 计算特征分桶



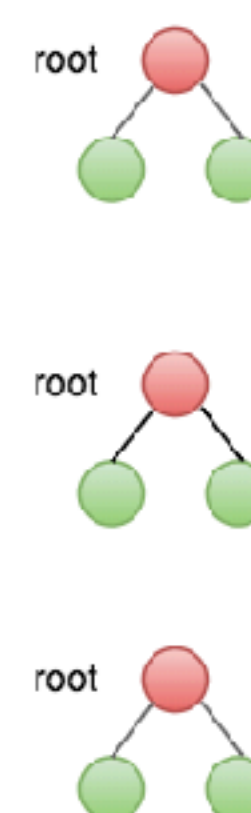
### 2, 计算直方图



### 3, 计算最优分裂



### 4, 分裂节点



5, 分裂出的节点重新计算直方图



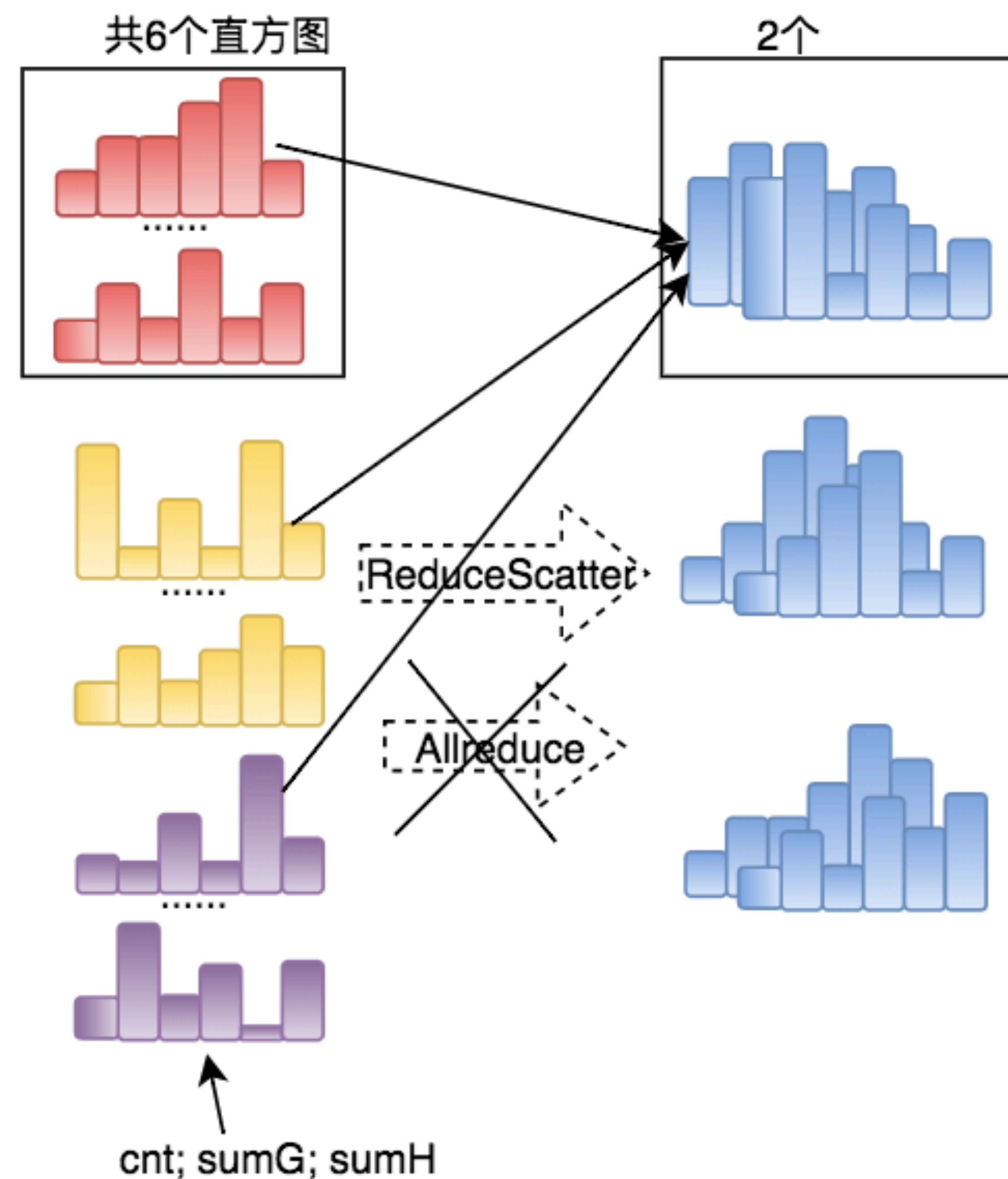
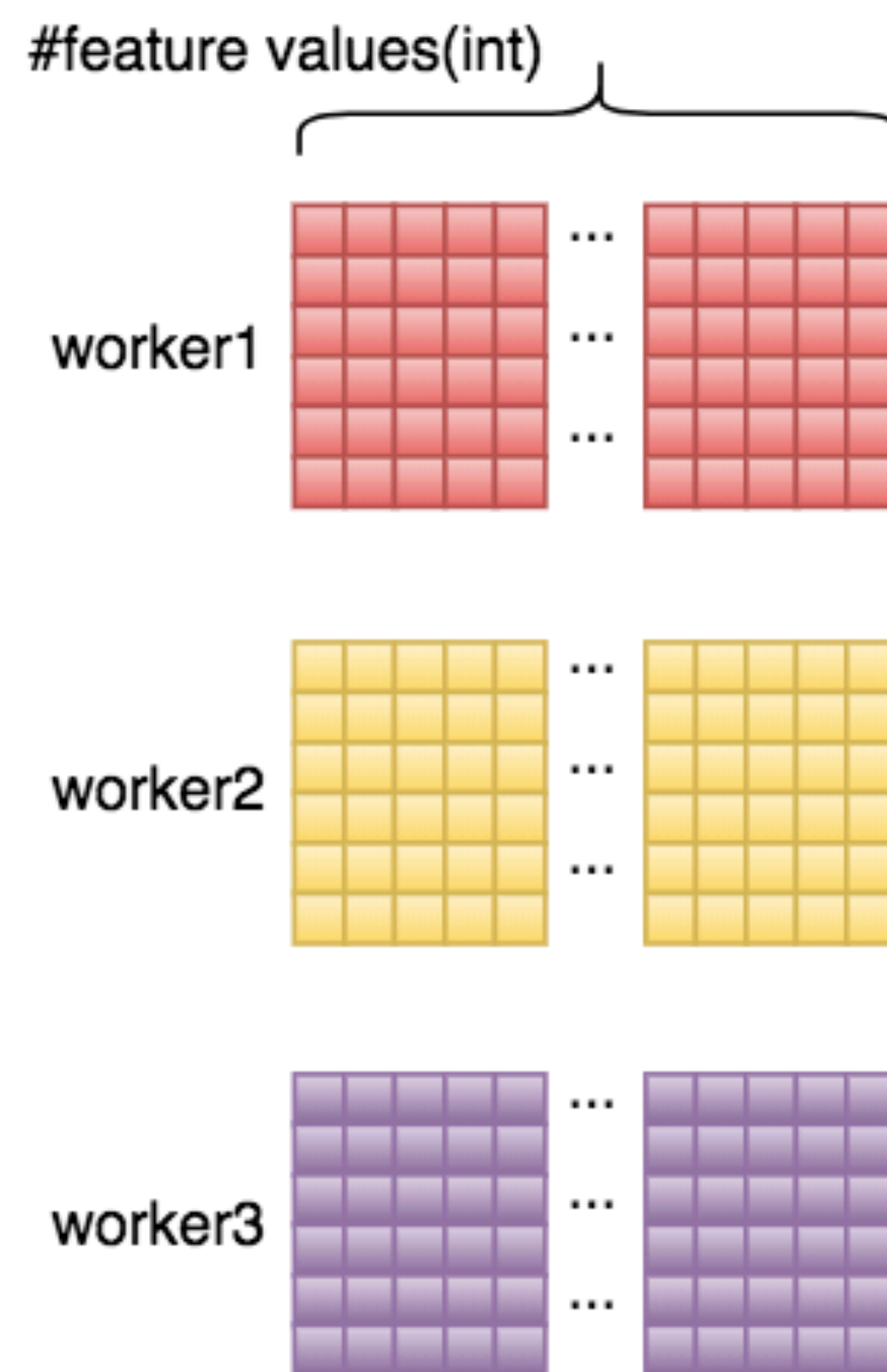
# LightGBM—并行化

## 02 建立特征直方图

### 1. 建立本地直方图

### 2. 合并全局直方图

- AllReduce : 对所有计算节点上的数值进行归约操作
- ReduceScatter : 只保留归约后的部分结果

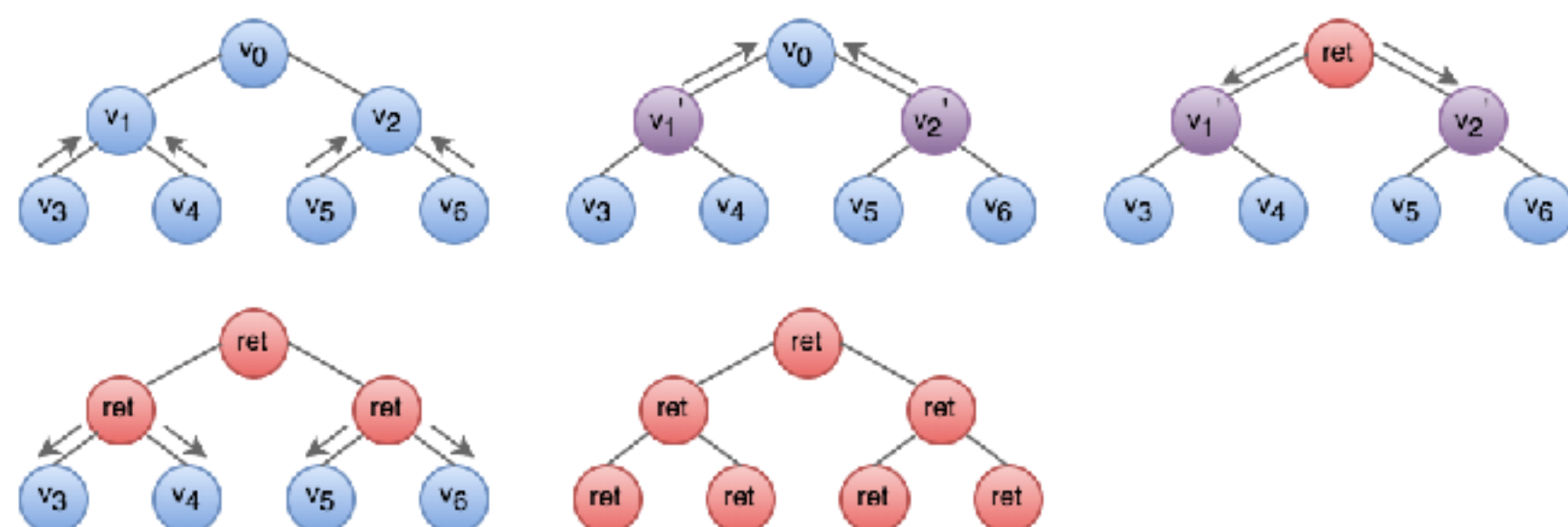


# LightGBM—并行化

## • ReduceScatter

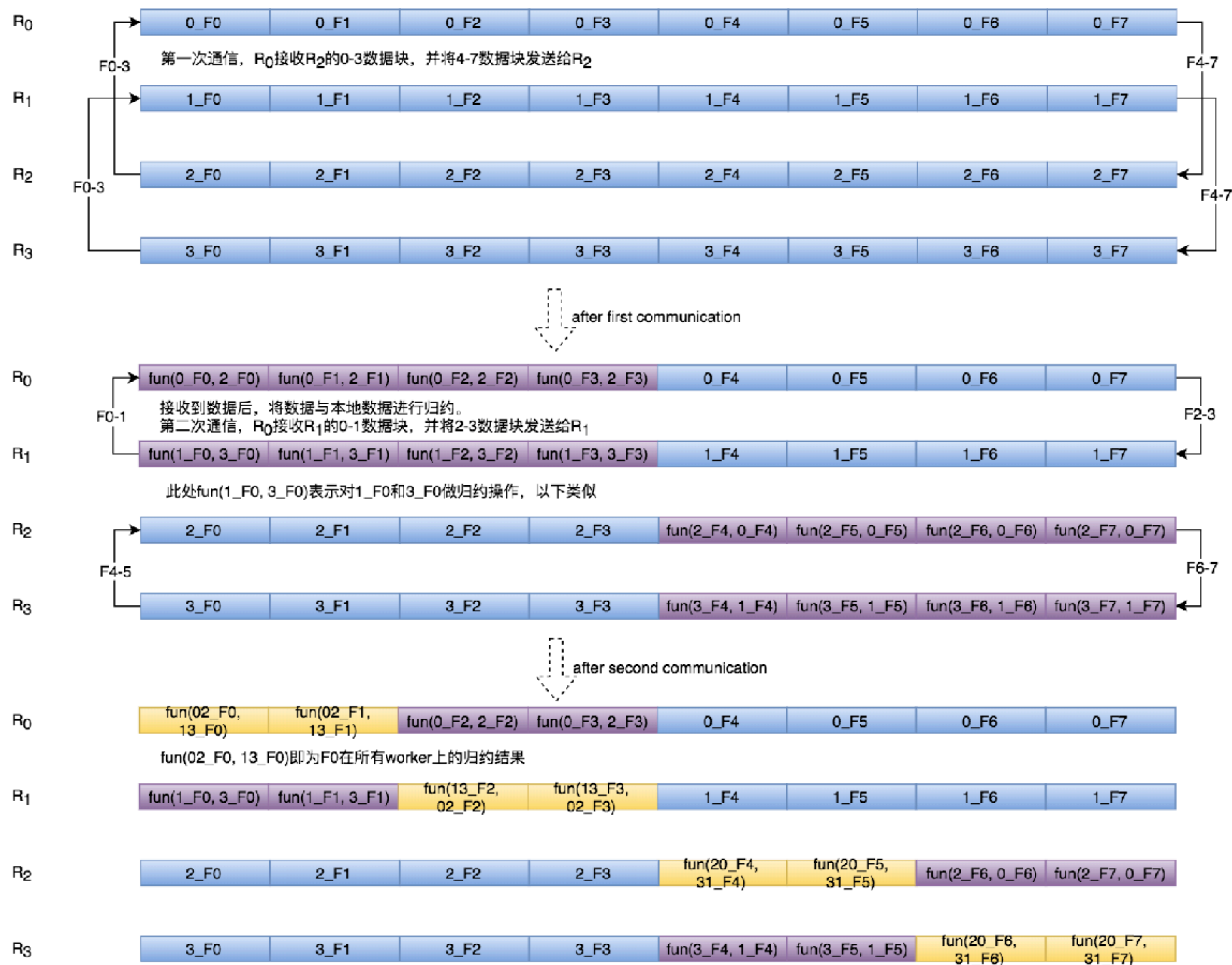
XGBoost->AllReduce

LightGBM->Recursive Halving



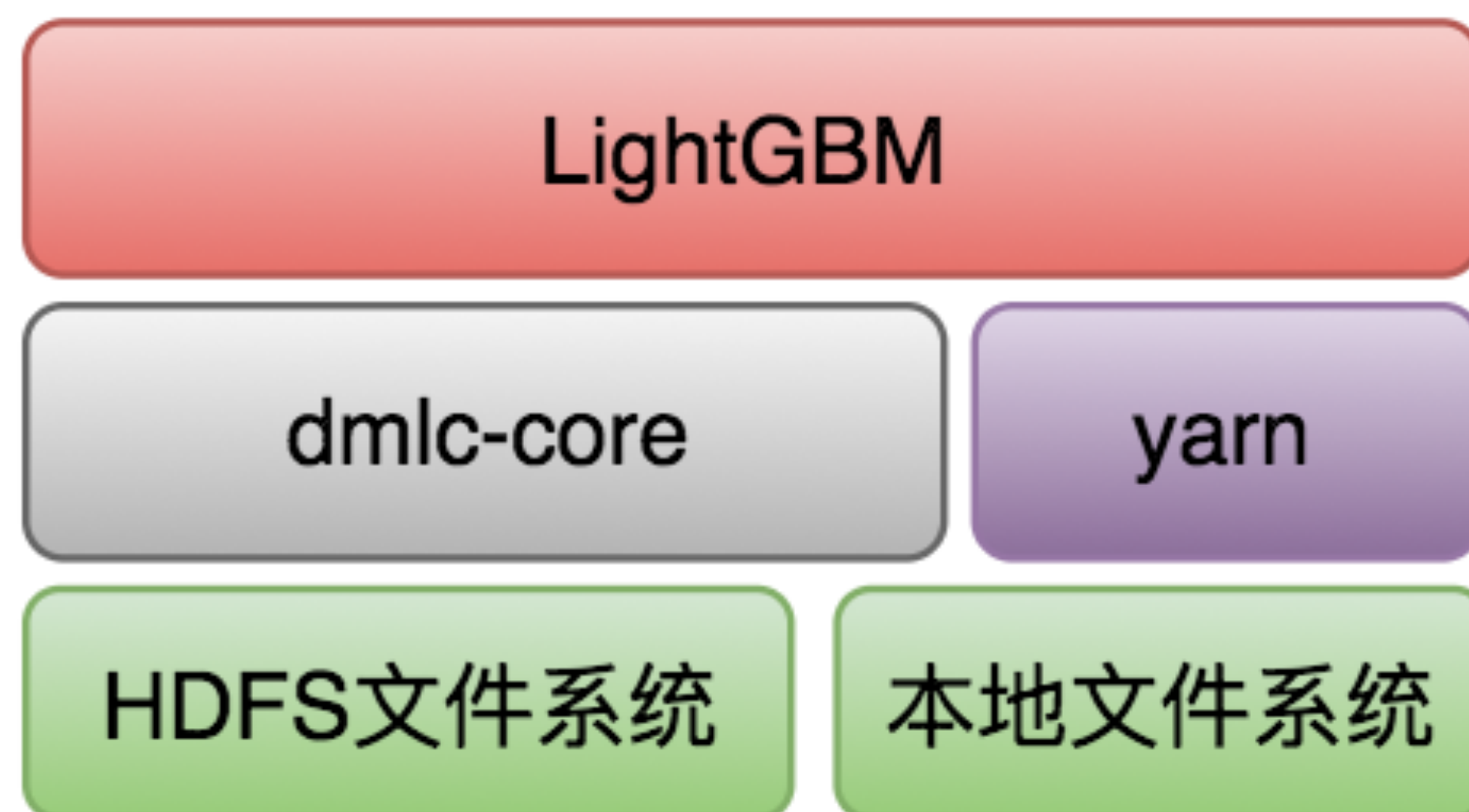
优点：大大减少通信数据量

缺陷：只适用于 $2^k$  worker情况



## 外卖广告实践—Yarn-LightGBM

- 支持yarn环境
- 支持HDFS文件



架构图

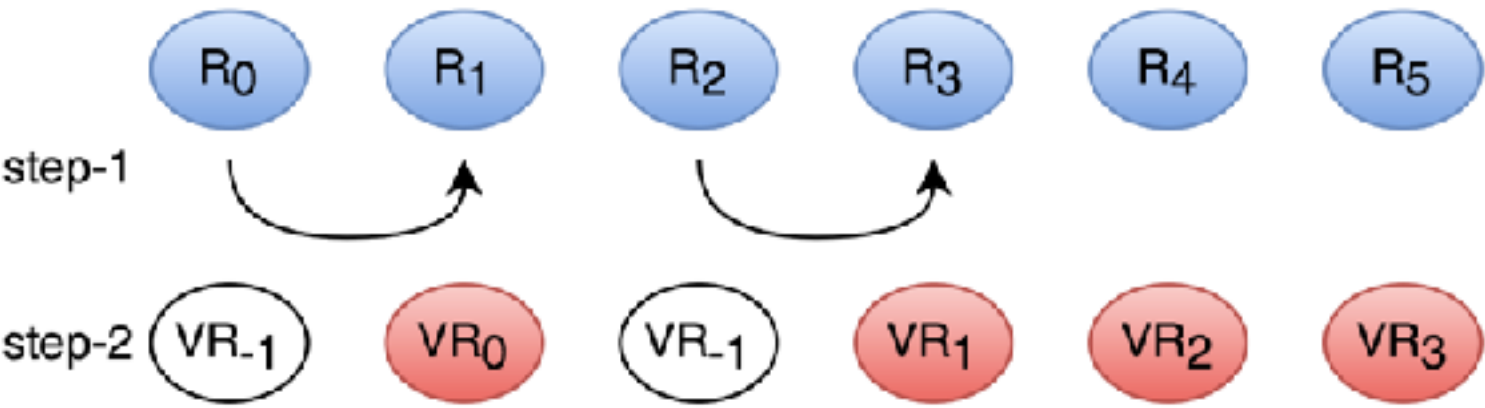


# 外卖广告实践—Yarn-LightGBM

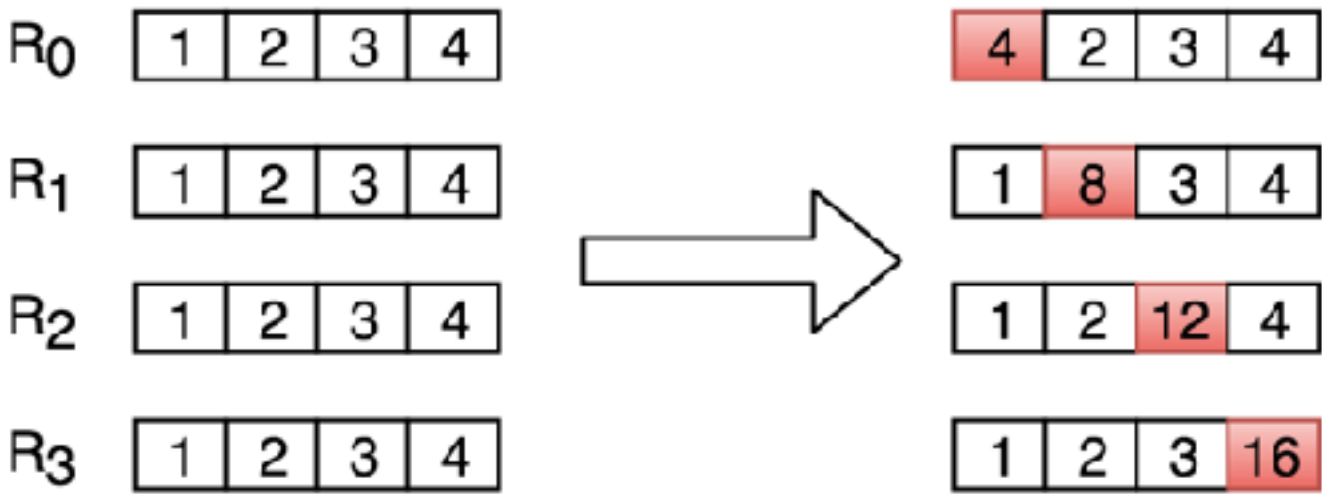
• 优化非 $2^k$ 个worker时的通信时间

执行步骤	XGBoost 64workers	LightGBM 64workers	LightGBM 65workers	Yarn-LightGBM 65workers
建立本地直方图	2.032276s	1.134293s	1.962546s	1.812981s
对所有直方图加和	13.681048s	2.097936s	28.615881s	3.19341s
同步分裂属性	19.017187s	0.907549s	3.051622s	1.851814s
分裂叶子	0.259526s	0.084525s	0.177973s	0.09139s
Trained a tree	35.272322s	4.436463s	34.086826s	6.949595s

优化：Virtual Rank + Recursive Halving



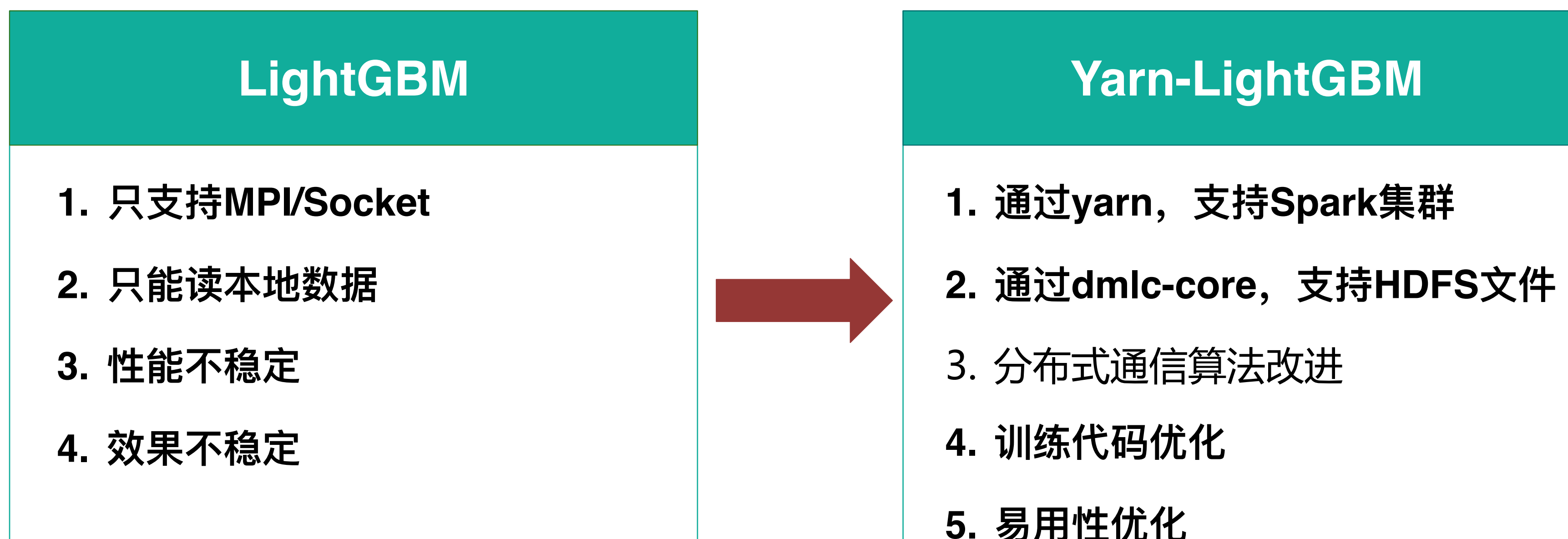
Step 1



Step 2

# 外卖广告实践—Yarn-LightGBM

## ➤ 总结：



## ➤ 效果：

✓ 以上特性已经被合并到github开源版本中

- 训练速度比XGBoost提升2倍
- 线上效果正向
- 目前在集团内部多个团队应用，性能提升1-3倍

# 目录

## 树模型

- XGBoost
- LightGBM

## 神经网络

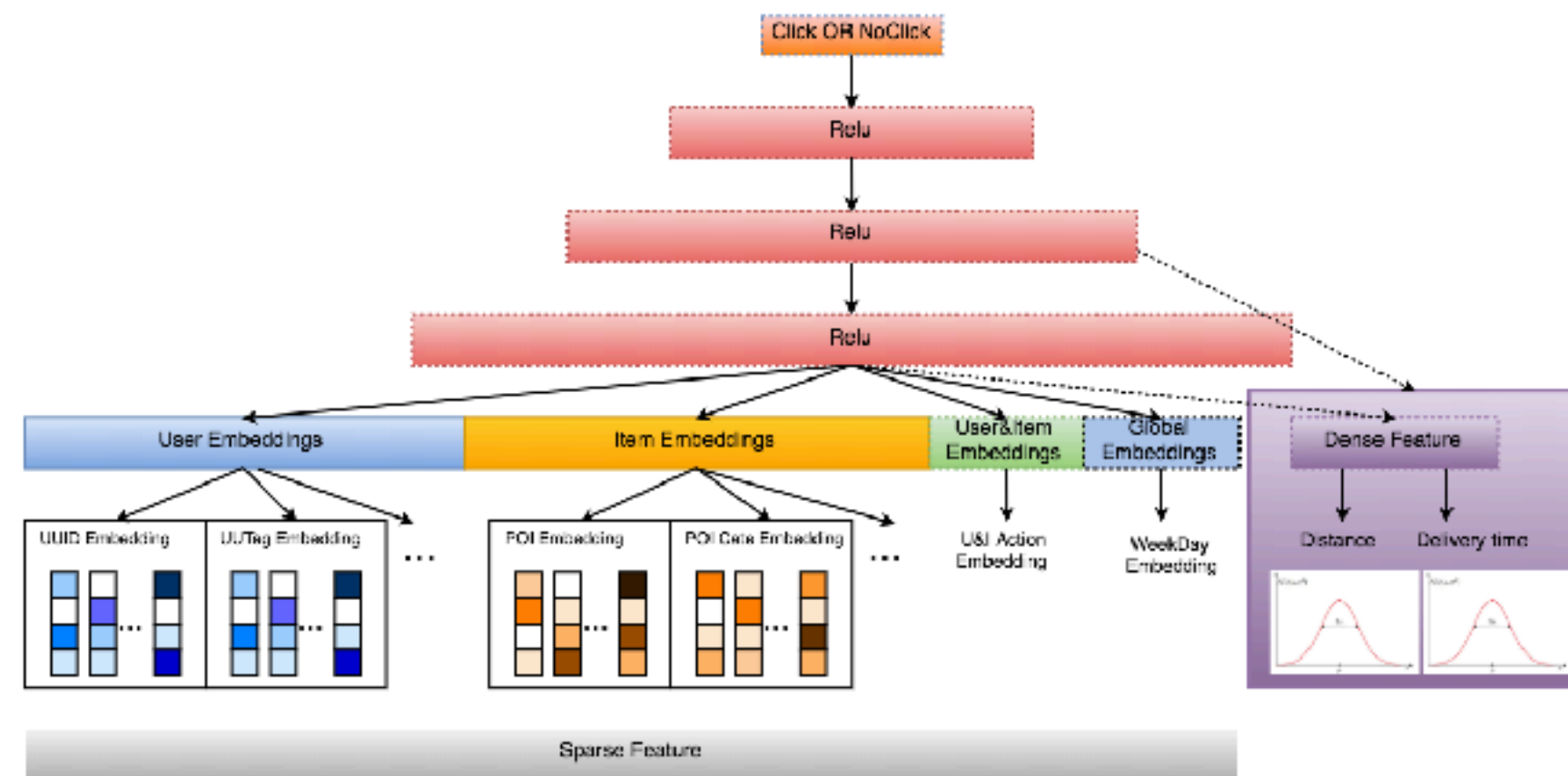
- DNN
- MTL

## 强化学习

- DQN



# 神经网络

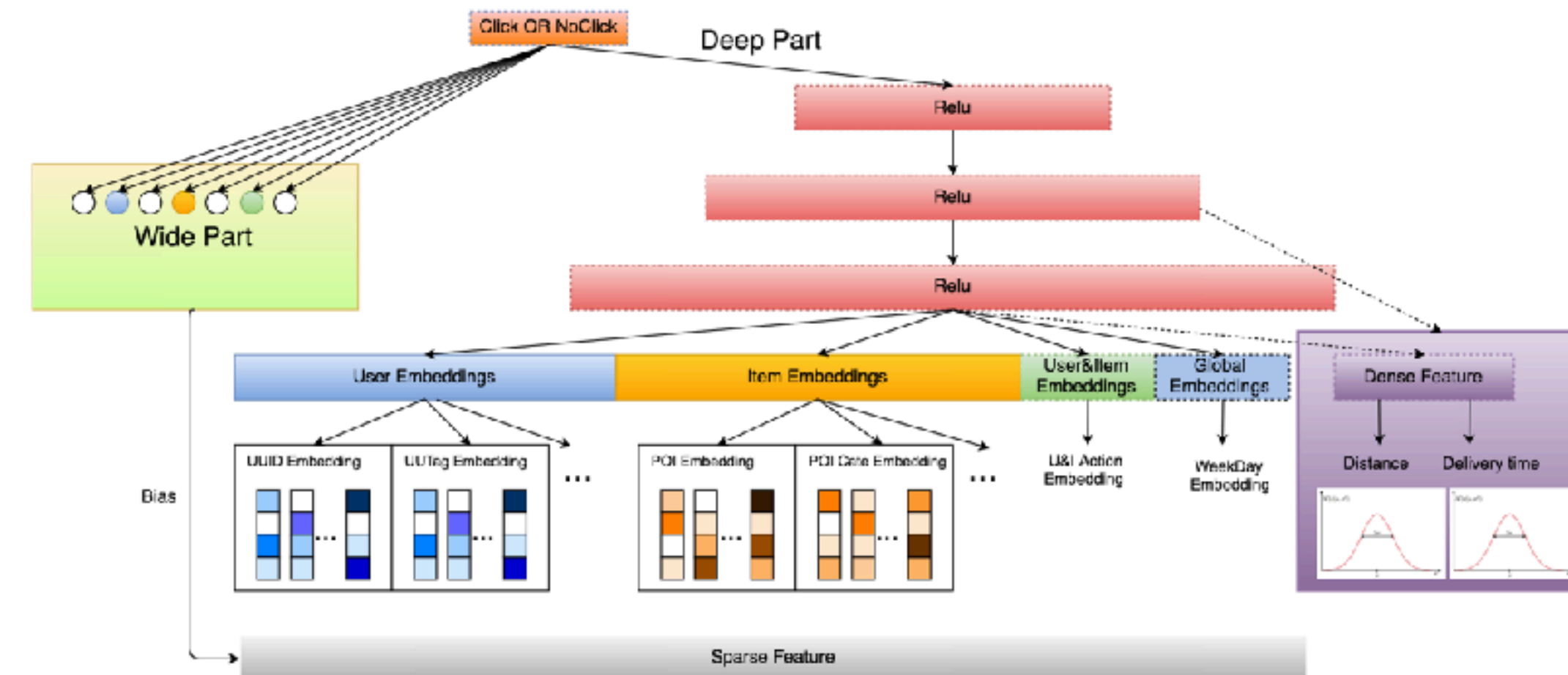


DNN

V1

LR+DNN

V2

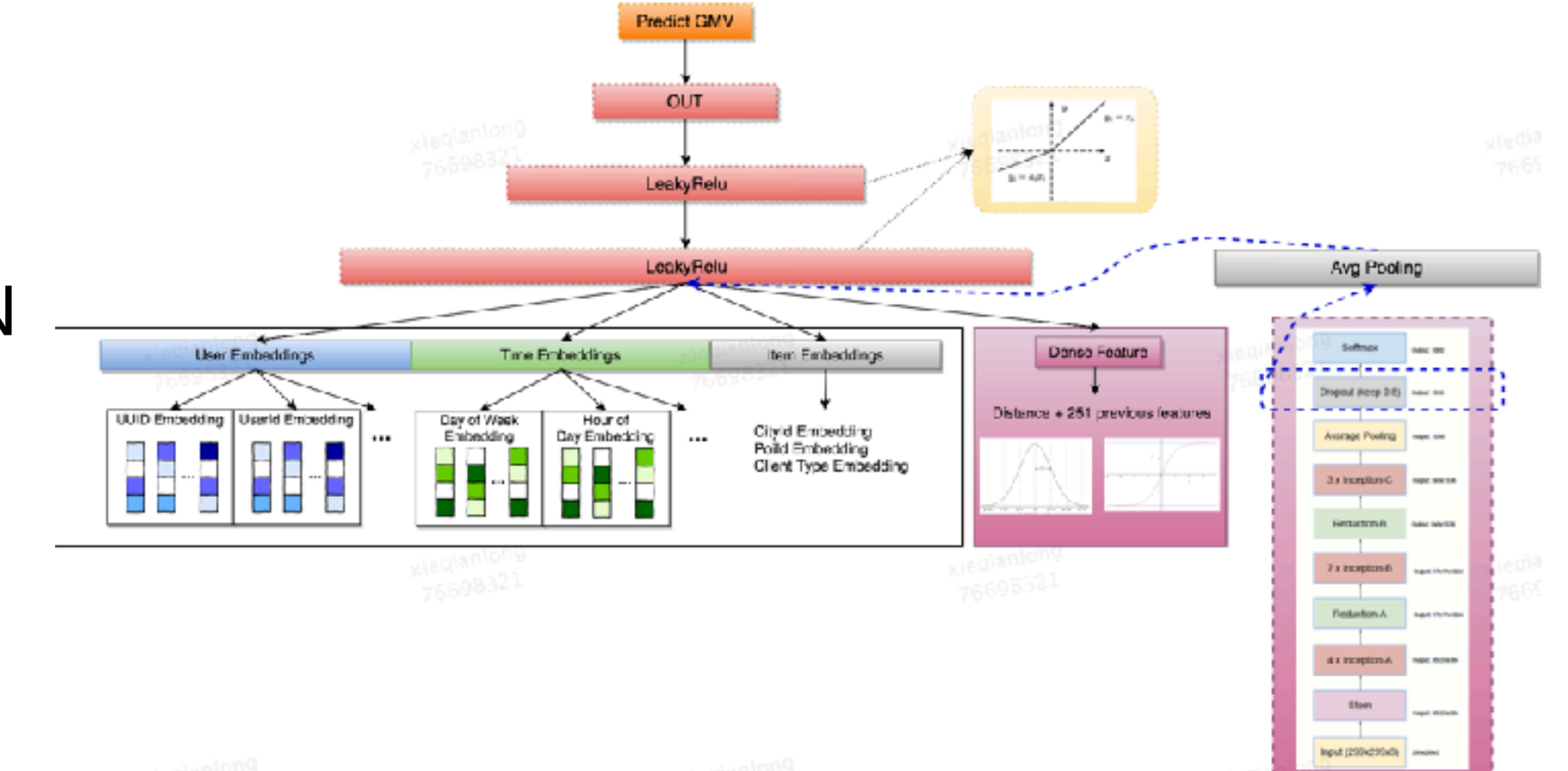
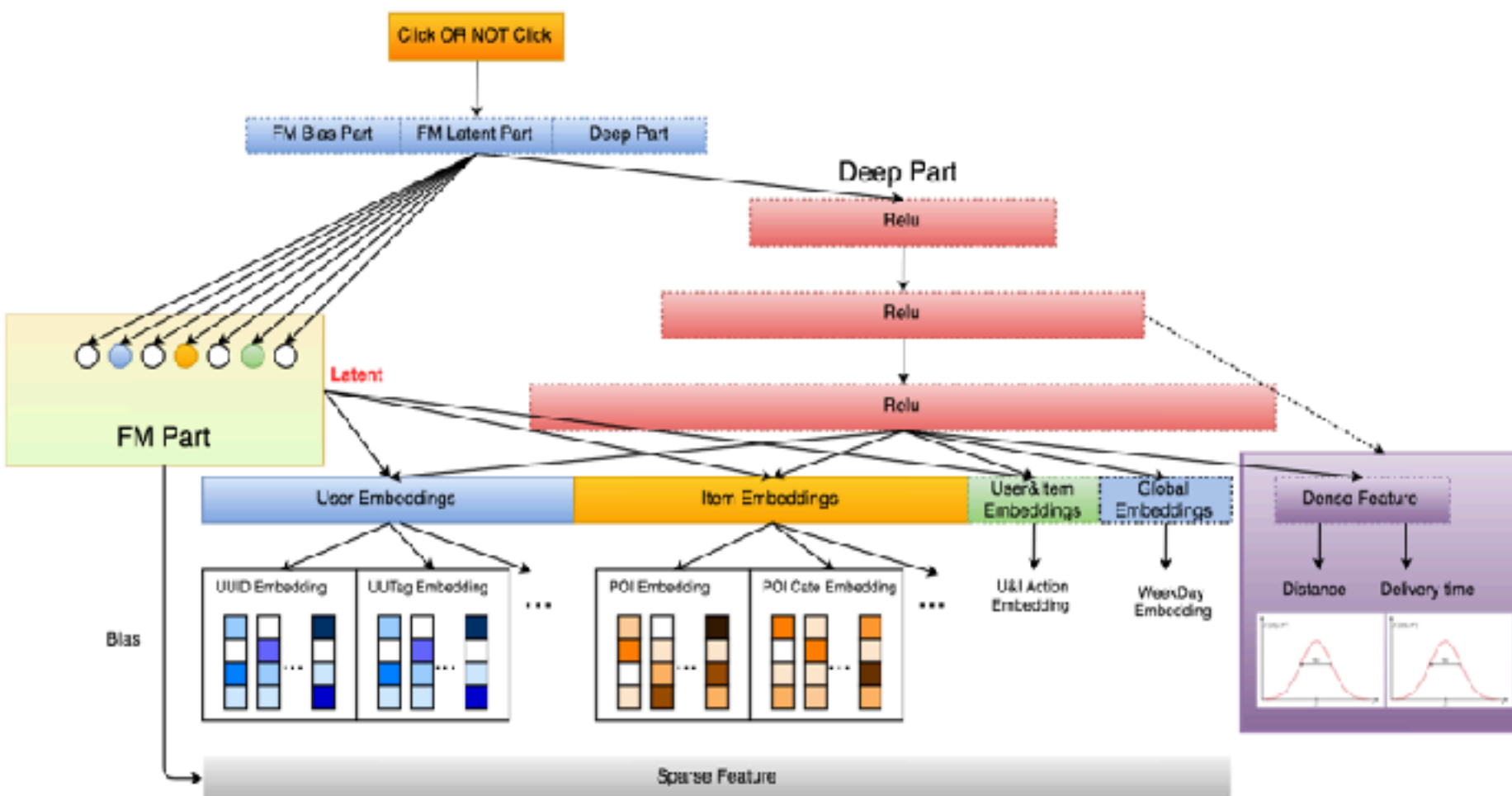


FM+DNN

V3

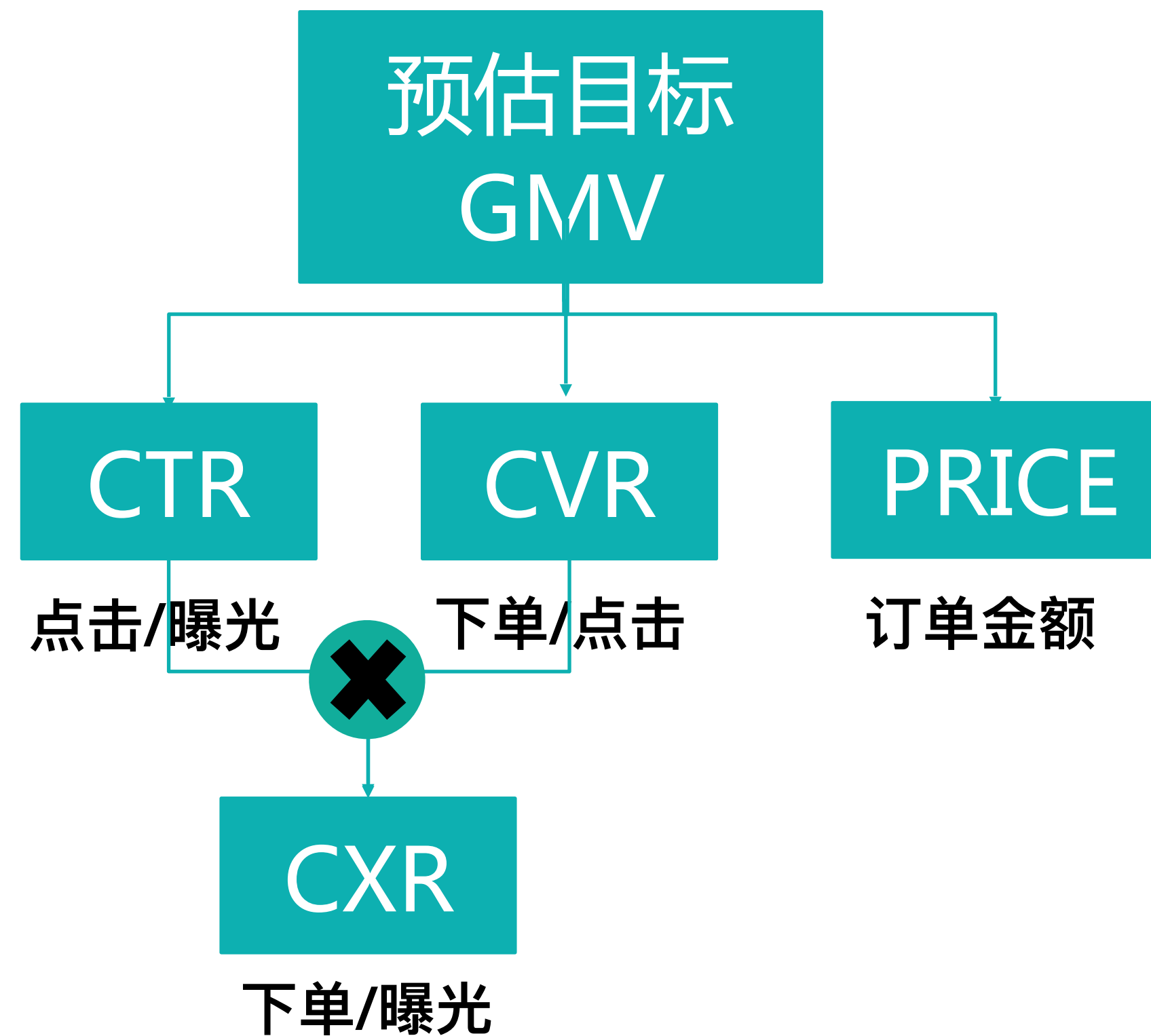
CNN+DNN

V4



## MTL-背景

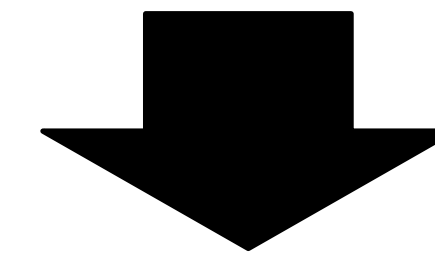
整体预估：不够灵活



拆分预估：误差大，样本稀疏

### 问题特点

- ✓ 输入几乎相同
- ✓ 任务高度相关



### 解决方案

#### Multi-Task Learning

- ✓ 同时学习和预测多个目标（DNN多输出）
- ✓ 解决CVR，PRICE样本稀疏问题

## MTL-技术选型

### 业内解决方案

#### 经典MTL方案

- ✓ CTR、CVR loss线性加权
- ✓ Joint/Alternate training
- ✓ 共享隐层

#### 阿里ESMM

- ✓ CTR, CXR loss加权
- ✓ CVR作为中间结果
- ✓ 共享Embedding特征表达

### 外卖广告解决方案

同时学习CTR和CVR

#### 共性

- ✓ loss线性加权
- ✓ Joint training
- ✓ 共享隐层

PRICE怎么办？

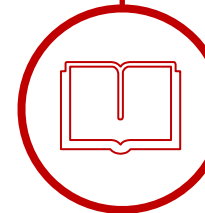
#### 特性

- ✓ 样本更加稀疏
- ✓ label的scale

# 迭代路径

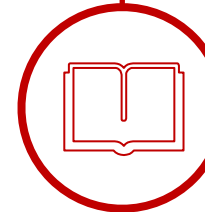
1

## 初步探索



方法

- MTL同时学习CTR、CXR



效果

- ctr和cxr相对dnn正向

2

## 上线落地



方法

- MTL同时学习CTR、CVR



效果

- 离线AUC有所提升
- 线上表现正向



问题

- Price如何解决

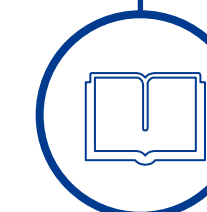
3

## 终极方案



方法

- MTL同时学习CTR、CVR、PRICE



效果

- CTR, CVR AUC持平
- MAE正向



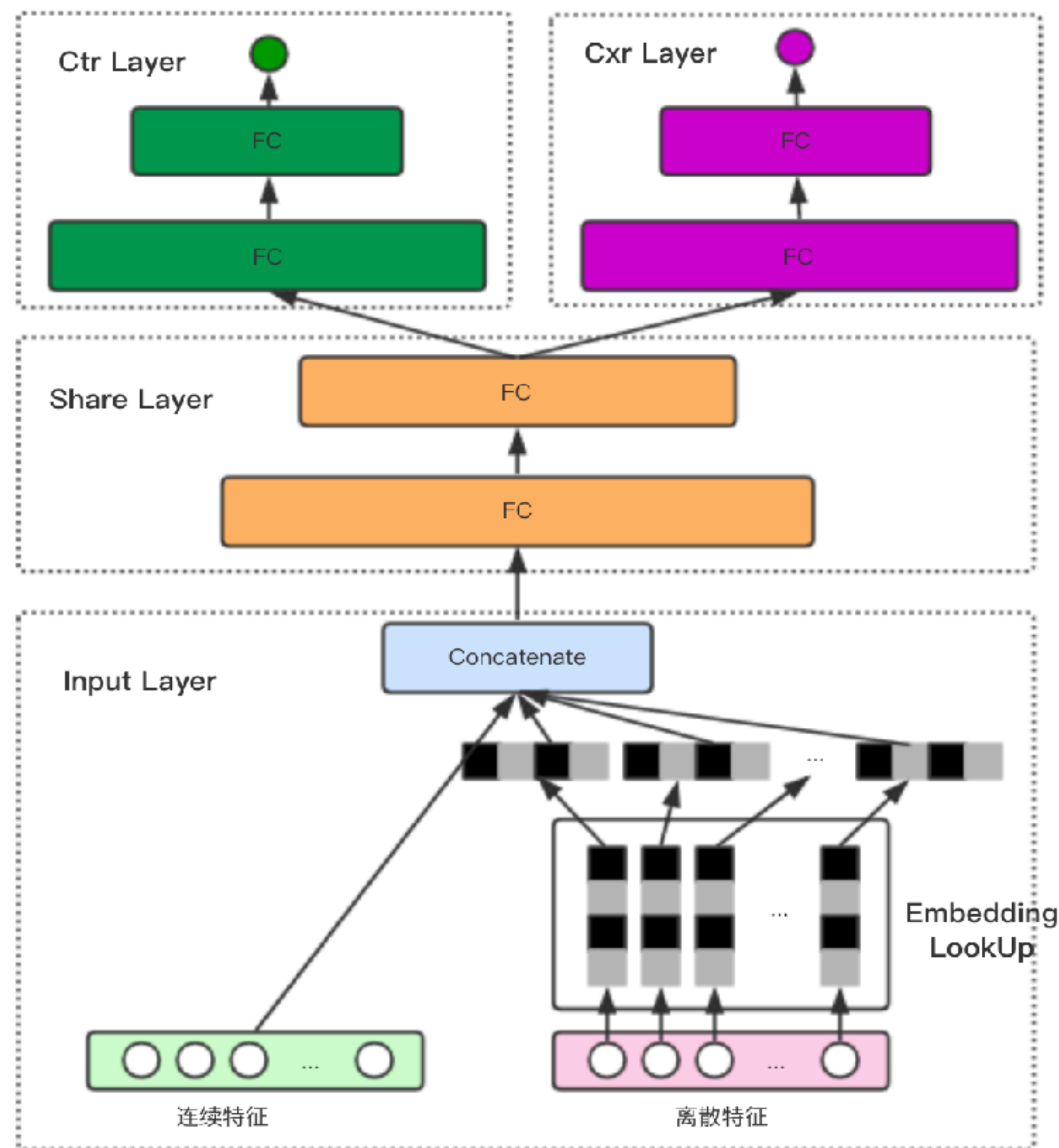
# MTL-阶段一

✓样本  
✓曝光

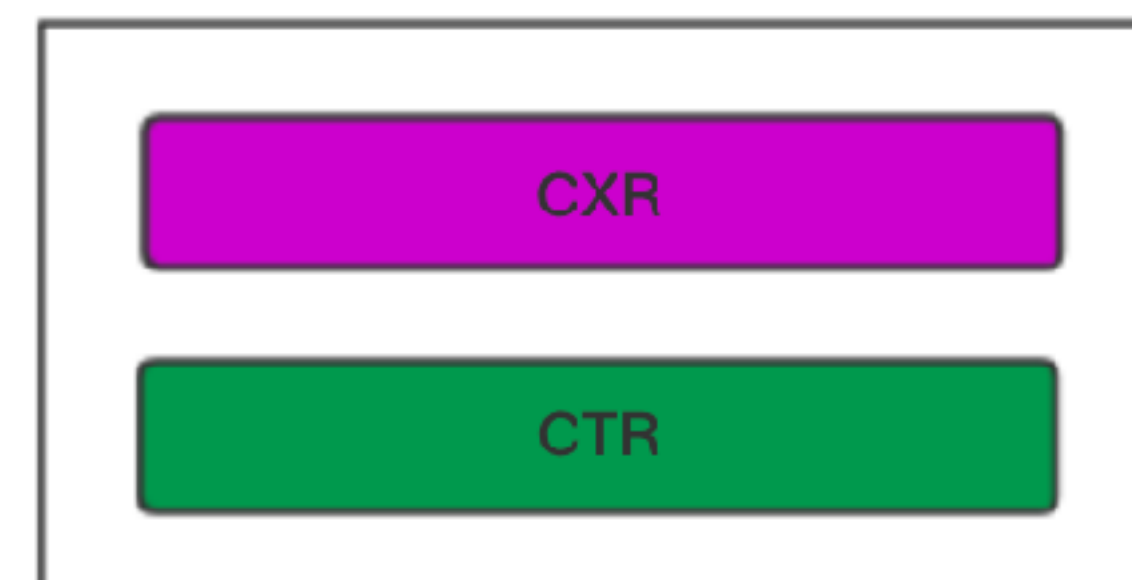
✓Label  
✓点击、成单

## 效果 (对比DNN)

✓ CTR和CXR均比单目标DNN有所提升



单个Batch的样本分布片段



## MTL-阶段二

改进

加入CVR Layer

挑战

样本不均

物理含义

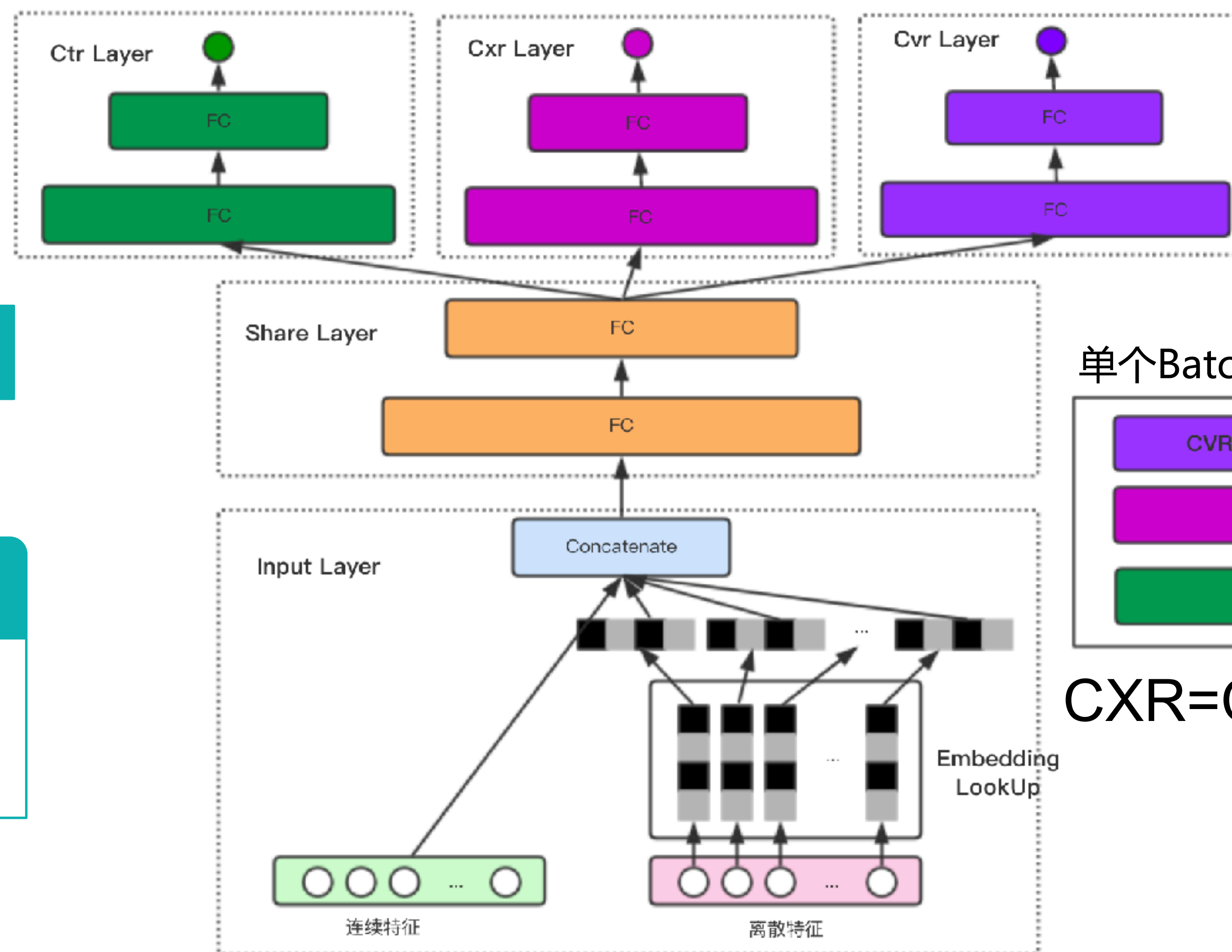
应对

Loss加权

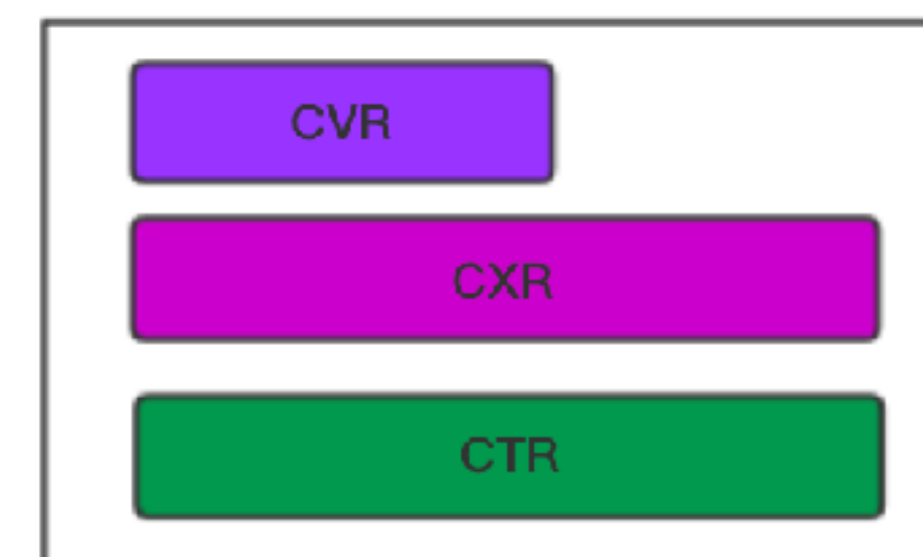
Loss约束

### 效果 (对比DNN)

- ✓ CTR和CVR均比单目标DNN有所提升
- ✓ 线上业务指标正向



单个Batch的样本分布片段



$$\text{CXR} = \text{CTR} * \text{CVR}$$



# MTL-阶段三

改进

加入Price Layer

挑战

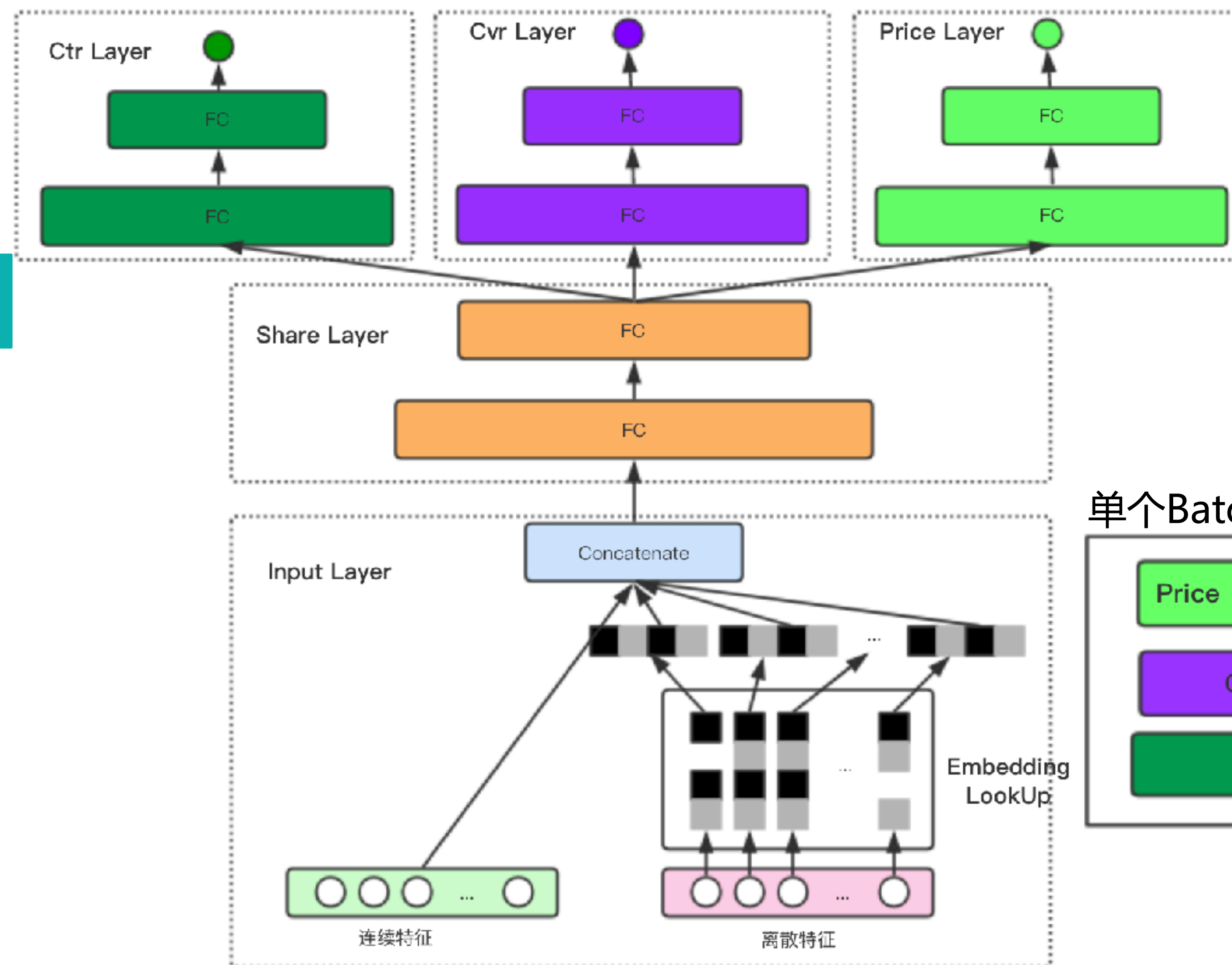
样本不均

Loss冲突

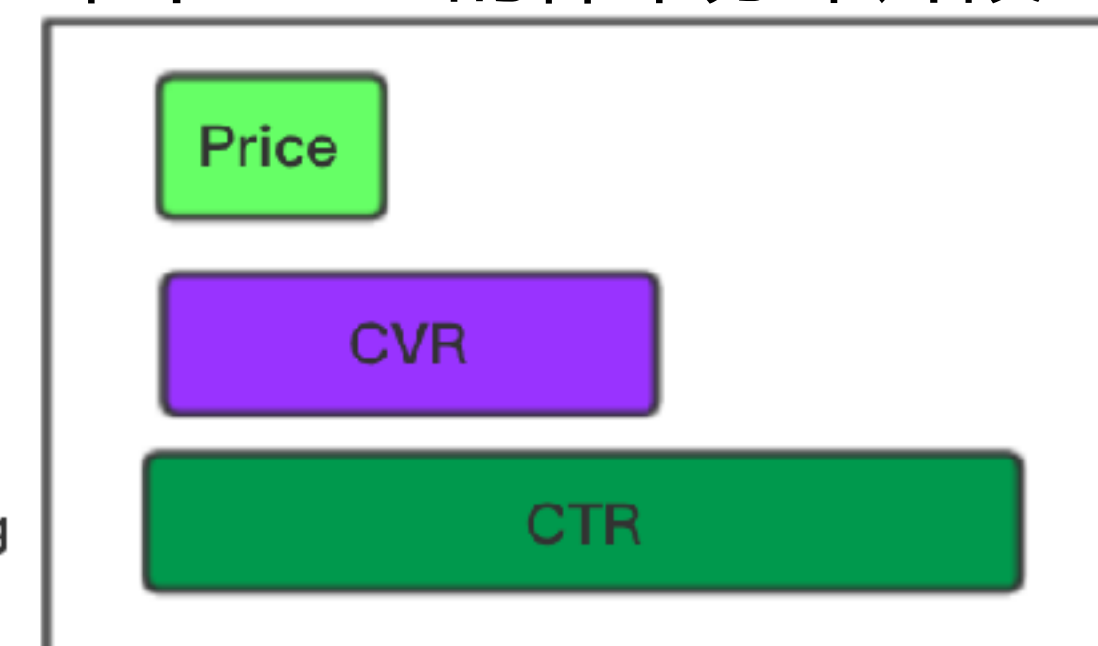
应对

Loss加权

- 无法同时收敛



单个Batch的样本分布片段



# MTL-阶段三

挑战

样本不均

Loss冲突

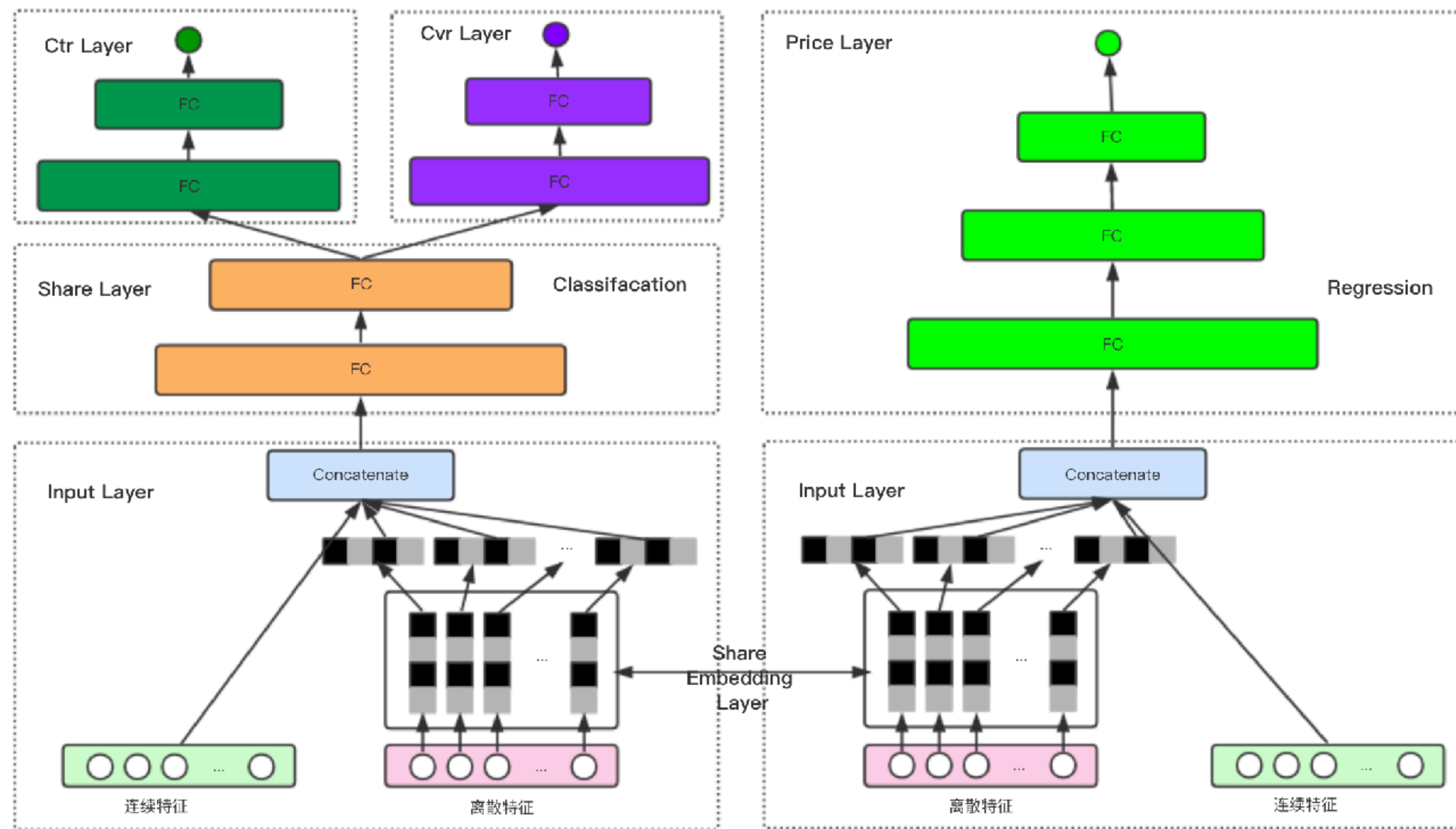
应对

网络拆分

共享Embedding

## 效果

- ✓ 同时收敛
- ✓ CTR, CVR AUC持平
- ✓ PRICE MAE正向



# 目录

## 树模型

- XGBoost
- LightGBM

## 神经网络

- DNN
- MTL

## 强化学习

- DQN

# 用户体验 (UEQ)

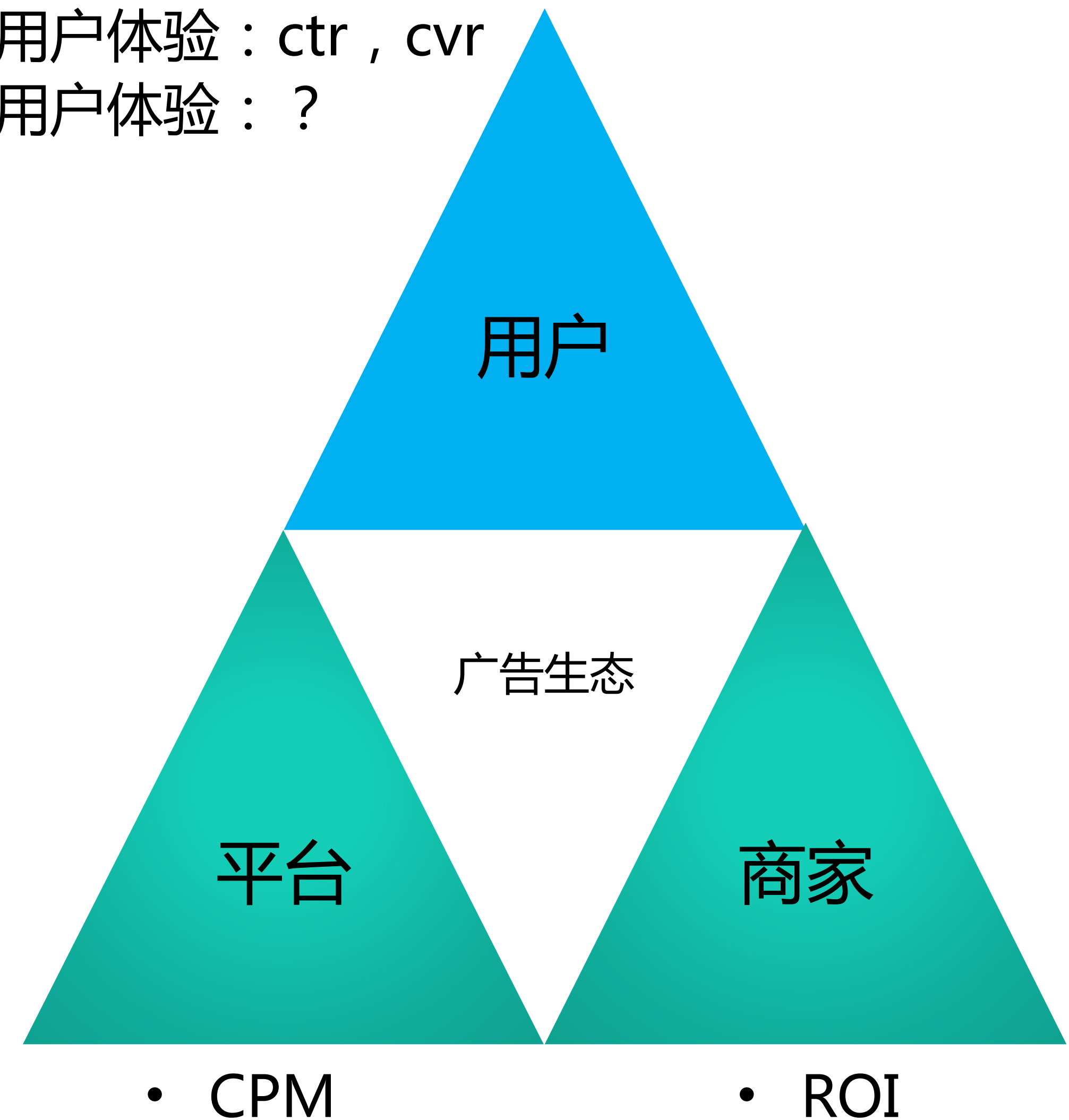
## ✓ 为何要关注用户体验？

- 影响对资源位入口的认知
- 影响用户留存，平台的长期发展

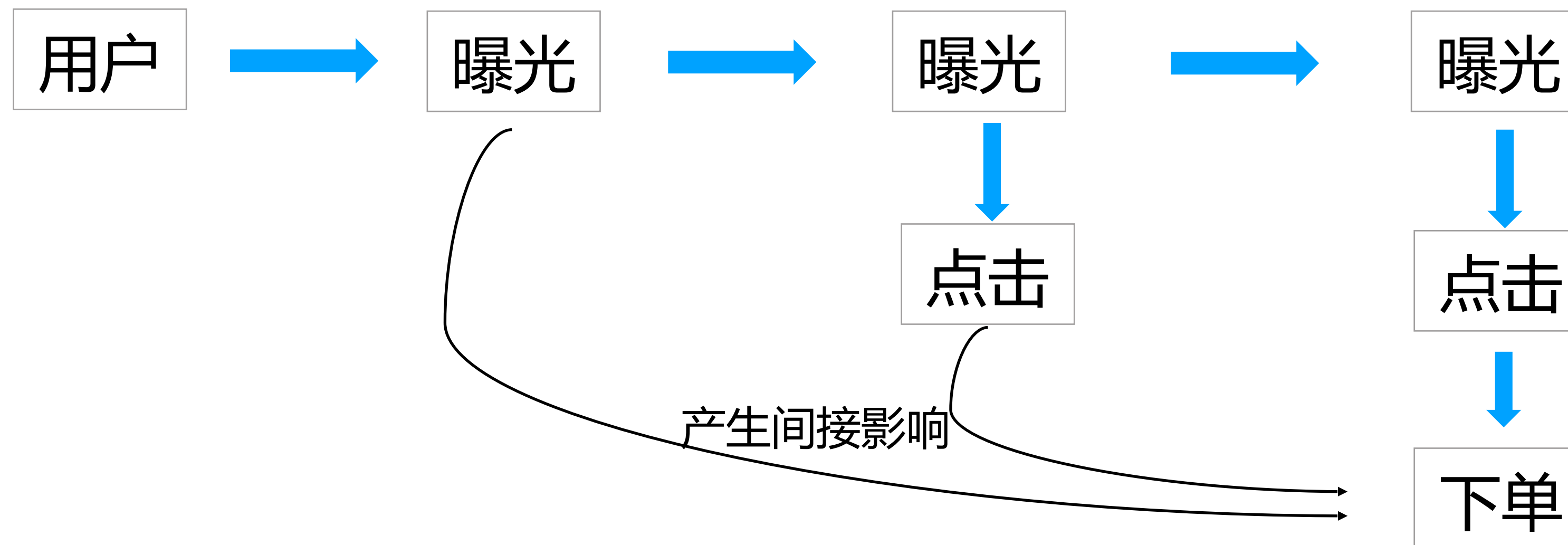
## ✓ 用户体验如何定义？

- 平台复购率
- 用户活跃度
- 推荐多样性

- 短期用户体验：ctr，cvr
- 长期用户体验：？

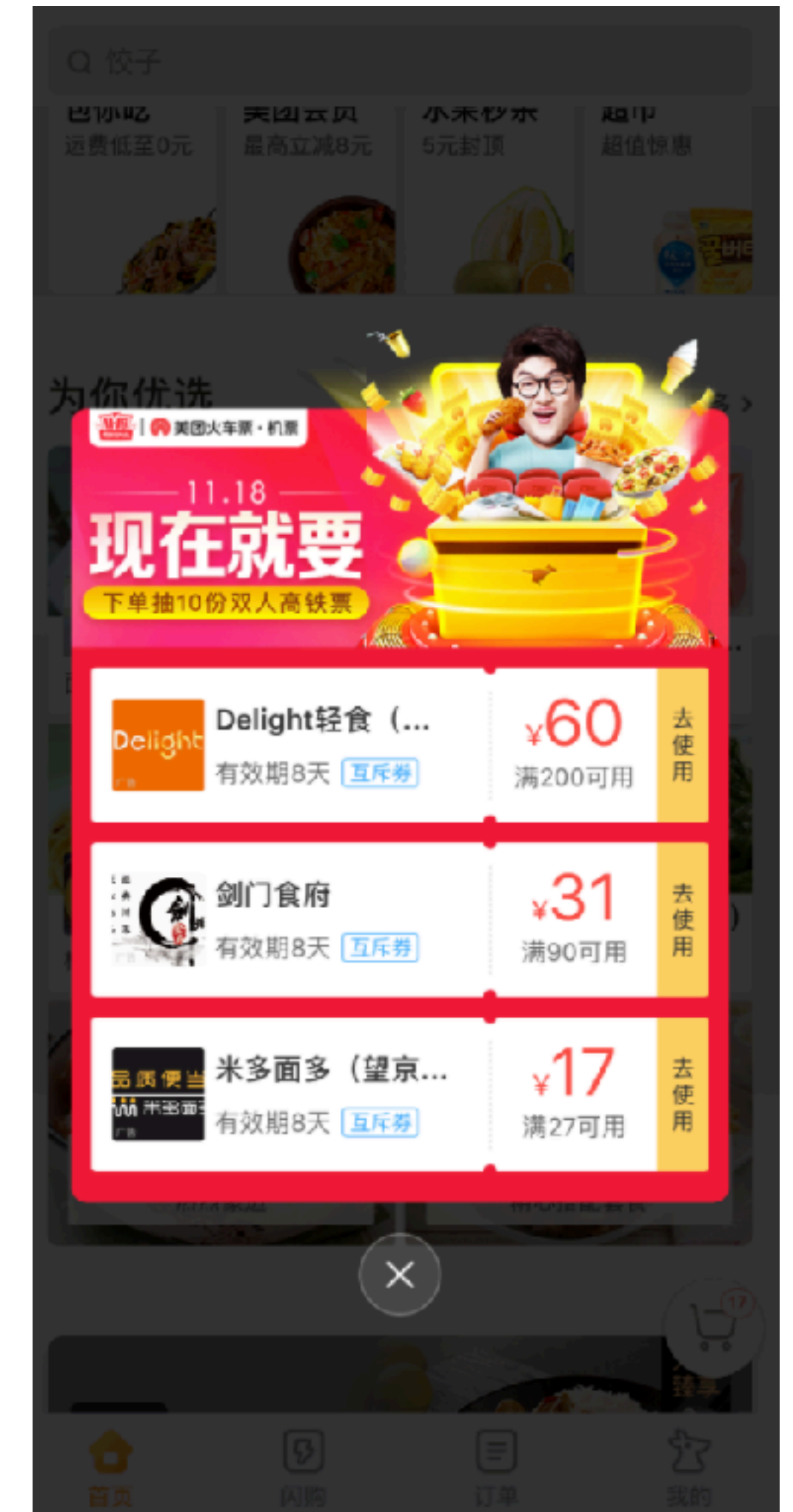


## 业务场景



### ✓ 为什么用强化学习？

- 长期指标，全局最优
- 多目标（复购+活跃度）
- 探索产生多样性

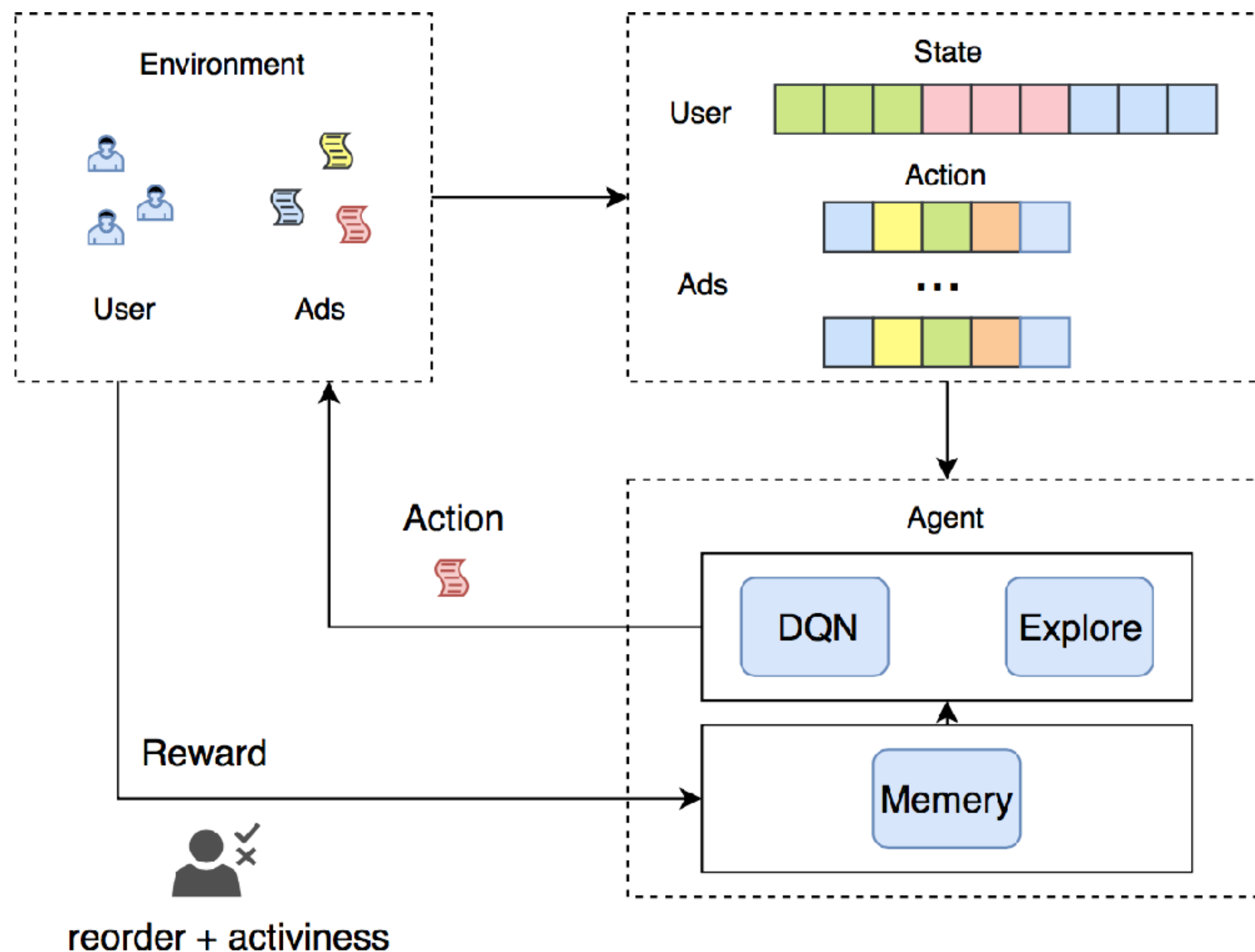




# 系统框架

## 强化学习的5大要素：

- environment：用户集和广告集
- state：用户的特征表达
- agent：推荐系统
- action：推荐的广告商家
- reward：复购率+活跃度





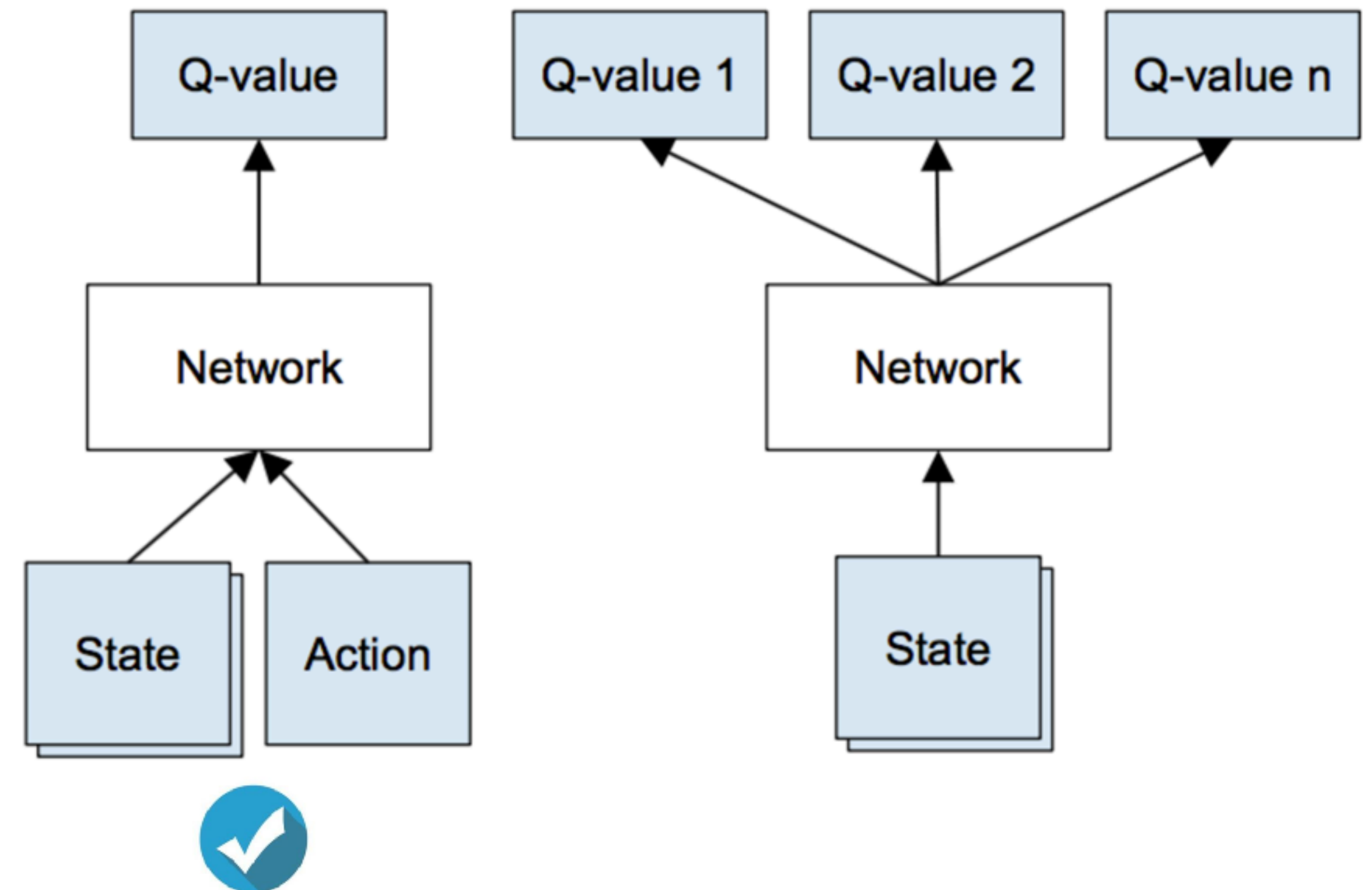
## 要点详解

- state：用户的特征表达
  - 用户历史浏览/点击序列
  - 用户画像特征：性别，年龄，历史单均价等
  - 地理位置，时间
- action：推荐的广告商家
  - 广告历史统计特征
  - 广告ID类特征
- reward：复购率+活跃度

$$r = \begin{cases} 1 & \text{在平台下单} \\ 0 & \text{未下单} \end{cases}$$

$$r = r * \frac{1}{T} \quad \text{其中} T \text{为距上一次下单的天数，间隔越小reward越大}$$

- agent：DQN->Double DQN->Dueling DQN



## Explore ( 多样性 )

### ➤ e-greedy

一定概率随机推荐，获取充分丰富的样本。

代价大，效果有损

### ➤ UCB(Upper Confidence Bound)

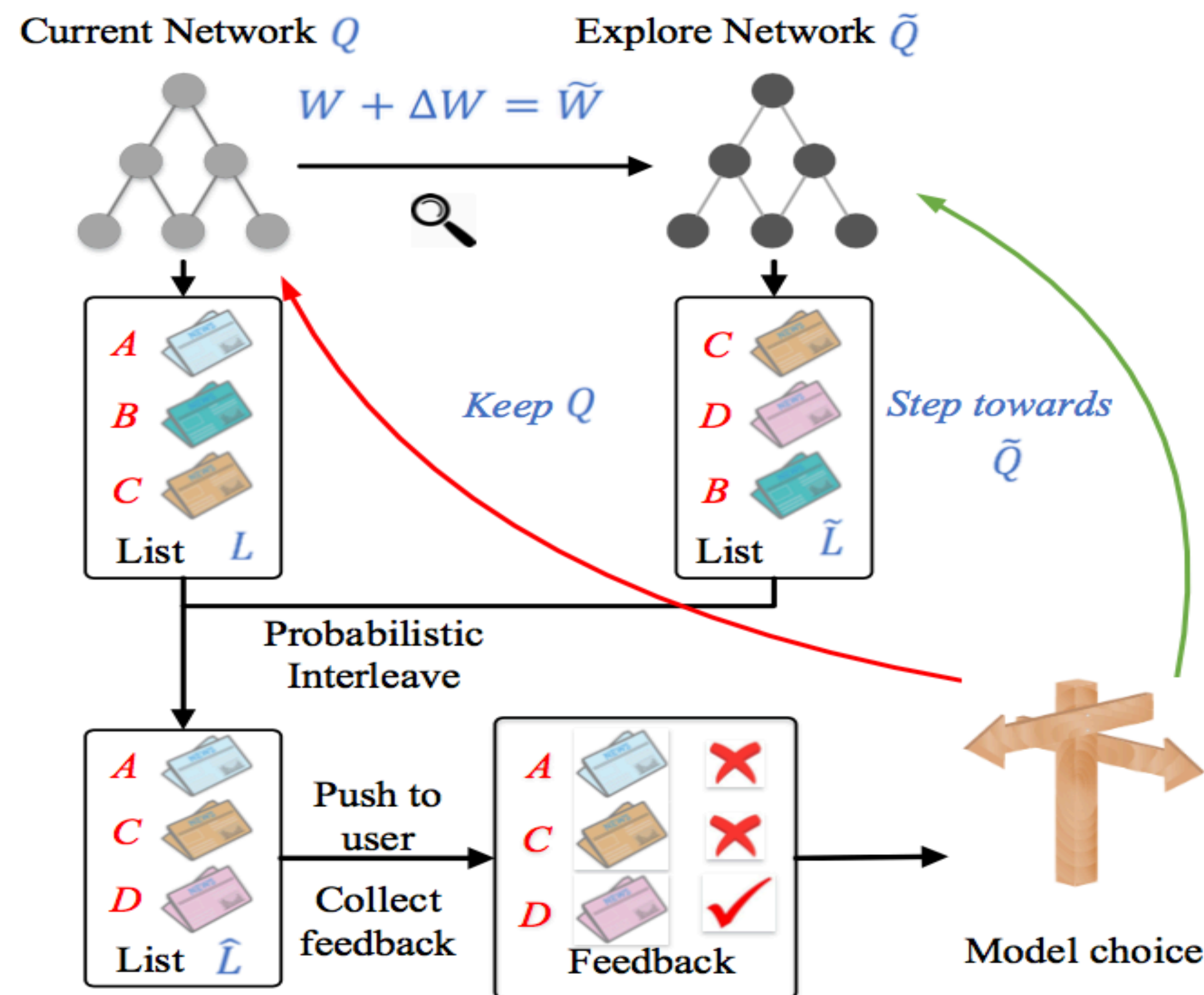
在目前输出Q value的基础上，乘以一个系数： $Q * (1 + \sqrt{\frac{1}{n+1}})$

其中n为广告历史展现的次数，倾向于曝光少的广告

效果优于  
e-greedy，但仍然不够准确

# Explore ( 多样性 )

## ➤ DBGD(Dueling Bandit Gradient Descent )



1, 对模型参数随机扰动, 倾向选择当前策略推荐的item附近的候选集, 效果损失最小

2, 可以对模型进行实时更新

Q&A