计算广告

王超

点击率预估背景介绍

- 计算广告核心问题：为一系列用户u与环境c的组合在这T次展示上找到最合适的广告投放策略a以优化整体的ROI。

PS → PV，负责提升流量

Ecom → eCPM，负责变现

$$\max_{a_{1,\dots,T}} \sum_{i=1}^{T} ctr(a_i, \underbrace{u_i, c_i}) * acp(a_i, u_i) / \sum_{i=1}^{T} p(i)$$

Ctr提升：

- 广告主转化提升
- 用户体验提升
- 公司收入提升

受限优化：在线分配

微观优化：点击率预估

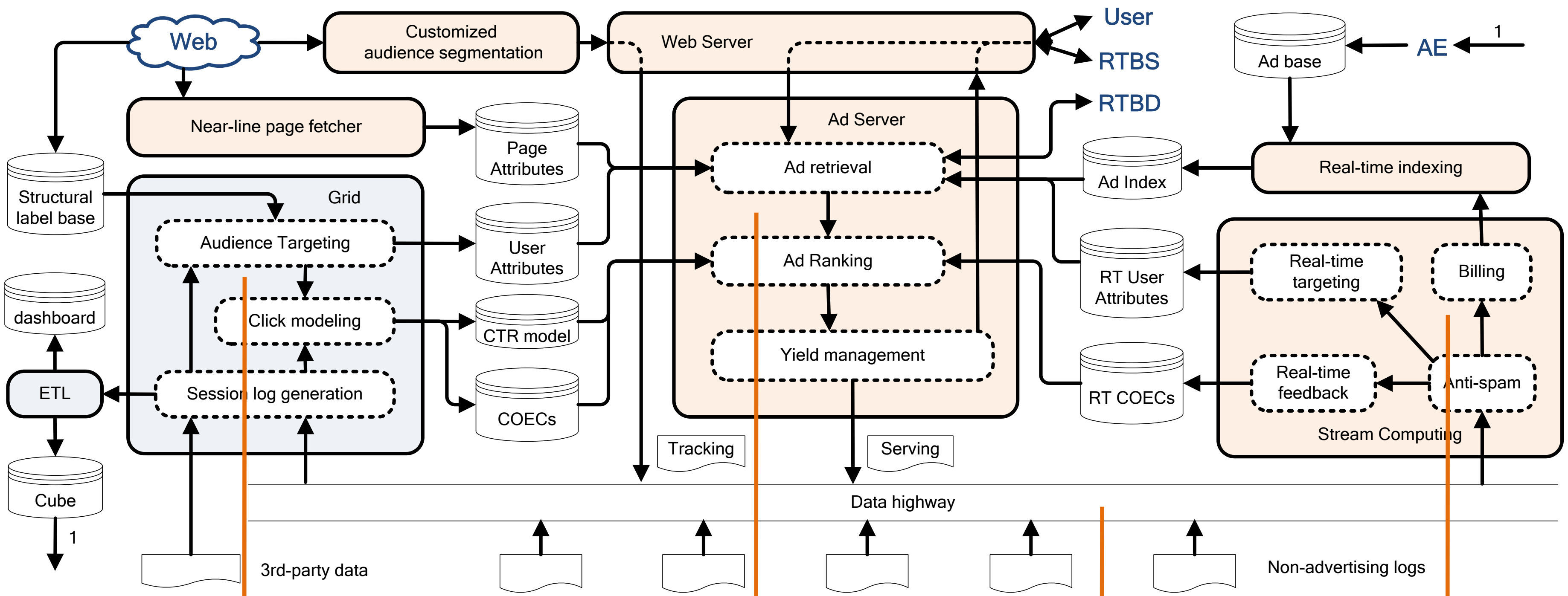
强化学习：探索与利用

特征提取：受众定向

推荐算法：个性化重定向

宏观优化：竞价市场机制

点击率预估背景介绍



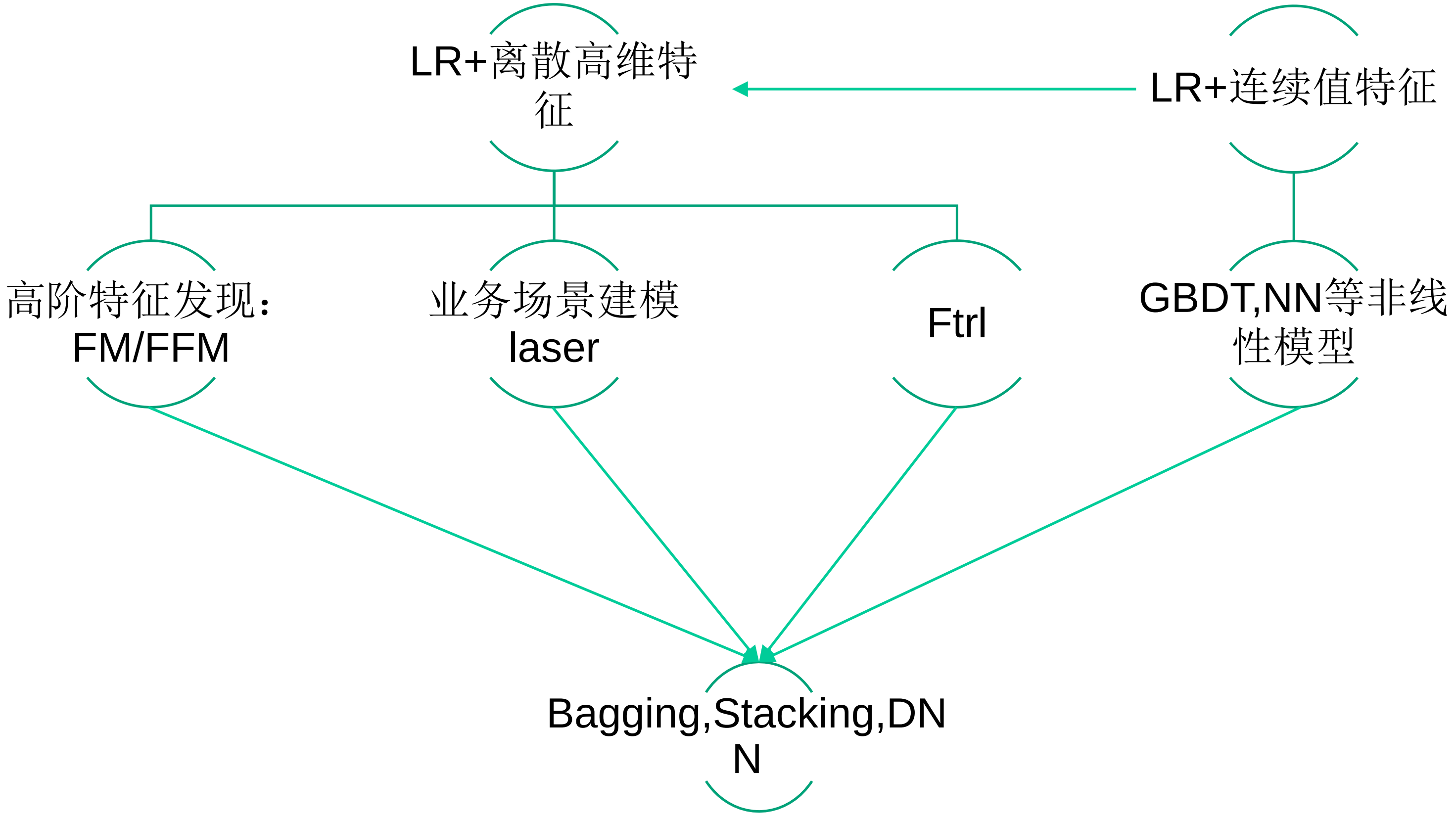
受众定向平台
灵活的海量数据挖掘平台
前沿机器学习算法的分布式架构

高并发投送系统
十毫秒级别的实时决策
百亿次/天的广告和内容推荐

数据高速公路
内部及外部TB级数据实时收集处理

流式计算平台
日志的准实时挖掘和反馈
反作弊与计价

ctr预估模型演进


LR+离散高维特征
LR+连续值特征
高阶特征发现：FM/FFM
业务场景建模laser
Ftrl
GBDT,NN等非线性模型
Bagging,Stacking,DNN

特征工程

长尾场景下，发现并处理好湮没在噪声中的大量弱信号，才能做好相关性检索。我们把这一过程称为特征工程。

常见特征变换：
- 连续特征离散化：等值/等频分bin，卡方/信息增益，Gbdt路径，…
- 非线性变换： Log/指数/开方，平滑，…
- 特征组合：人工构造笛卡尔积， FM/GBDT ，DNN ， …
- 静态特征->动态特征

常见特征选择：
- Filter：卡方，互信息…
- Wrapper：依赖于特定的分类器，单特征auc/AUC/MAE等，…
- Embedded：Lasso，GBDT，DNN，…
- 领域知识依然重要

动态特征 - 多层次点击反馈

- 将离散的静态特征变换为点击反馈的统计特征，模型往往采用gbdt/dnn等

优势：

- 模型无需实时更新(与在线学习相比)

缺点：

- 在线特征的存储量大(相对模型的方案)，更新要求高
- 特征值不置信，需要去噪消除bias，$\mathrm{COEC} = \sum_i click \,/\, \sum_i EC$
- 特征值不置信，需要平滑，如基于二项/beta共轭分布做经验贝叶斯平滑
- High level特征难以设计，复杂的特征工程+简单模型

1）make_feature_tag_simi

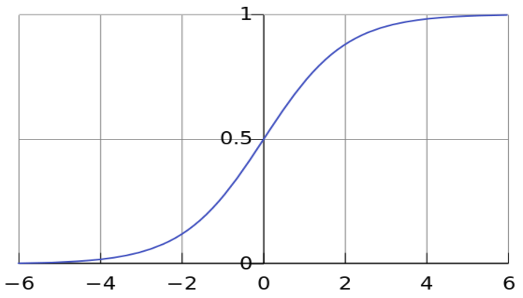
feature "tag_simi " 计算用户tag和url tag之间的相似度。if(user.tag[i].t==doc.tag[j].t) tag_simi += user.tag.wei[i]*doc.tag.wei[i]

9）make_feature_matched_utag_dtag_ctr_rtf

feature "mtag_ctr"

$$mtag_{ctr} = \left(\sum_{k=0}^{matched\_tag} (tag[k].ctr * tag[k].weight) \right) \Big/ \left(\sum_{k=0}^{matched\_tag} (tag[k].weight) \right)$$

经典模型LR回顾



为什么是sigmoid?

$$p(click|a,u,c) = \frac{1}{1+e^{-w^T x(a,u,c)}}$$

视角1：Generalized linear model在Binomial分布下的特例。LR的link func是logit函数。GLM家族拥有很多好的性质如充分统计量(对数据分布的压缩)等。

视角2：逻辑回归$p(y|x) = \frac{1}{Z_x} exp\overrightarrow{\theta_y} \cdot \vec{g}(x)$ 是最大熵$p(y|x) = \frac{1}{Z_x} exp\vec{\theta} \cdot \vec{f}(x,y)$ 在二分类且特征函数设计为 $\vec{f}(x,y) = \begin{cases} \vec{g}(x), y = y_1 \\ 0 \quad\quad , y = y_0 \end{cases}$ 时的特例：

$$\forall f_i: \sum_{x,y} p(y|x,\theta)\bar{p}(x)f_i(x,y) = \sum_{x,y} \bar{p}(x,y)f_i(x,y)$$

知之为知之，特征的期望分布=经验分布

$$\operatorname{argmax} \sum_x \bar{p}(x)\left(-\sum_y p(y|x,\theta)\log p(y|x,\theta)\right)$$

不知为不知，最大条件熵

LR模型参数求解

$$obj = \sum_{i=1}^{n}[y_i \ln\left(1 + e^{-w^T x_i}\right) + (1 - y_i)\ln(1 + e^{w^T x_i})]$$

$$grad = -\frac{\partial obj}{\partial w_\theta} = -\sum_{i=1}^{n}[y_i(1 - p_i) - w_\theta + (1 - y_i)p_i w_\theta] = -\sum_{i=1}^{n} w_\theta(y_i - p_i)$$

凸优化的经典算法都可以求解，详见计算广告一书：
SGD/LBFGS/Trust Region/ADMM/...

并行化的经典框架包括：
Parameter server(异步更新)/MPI(Allreduce/Broadcast)/Hadoop,Spark(Map Reduce)/...

LR+人工特征工程风光不再

LR+人工特征工程风光不再，近年具参考意义的一次比赛：

http://www.kaggle.com/c/criteo-display-ad-challenge/leaderboard/public

名次	team	方案
Rank1	Kdd2013 1st	ffm
Rank2	Kdd2012 2nd	特征工程+多模型
Rank6	Kdd2015 1st	
Rank7	本人	Fm + gbdt
Rank11	Kdd2012 1st	Gbdt 单模型
Rank12	Kaggle平台第一	
Rank50	品友互动比赛第一	

常见非线性模型：

MF: Matrix Factorization
Fm: Factorization Machines
Ffm: Field-aware Factorization Machine
Gbdt: Gradient Boosting Decision Tree
Dnn: Deep Neural Network

ctr预估-模型选择

- Online VS Offline：

  Online 模型善于fine tuning the head part，Offline模型善于刻画长尾。sgd对头部数据学习更快更充分，对于长尾稀疏特征，没有batch算法精细。

- 采用Dense特征模型的考虑：

  知识解耦于模型和特征值俩处，通过特征变化快速捕捉信号从而使得模型侧不用快速更新。除了模型侧的工作外，特征侧的特征值不再置信，需要复杂的特征工程(如pv+ctr/coec)。

- 人工特征工程 VS 非线性模型

  非线性模型如能在训练代价和精度上达到更好的tradeoff，泛化能力更强，长期优势

  基于领域知识的人工特征工程，短期优势

ctr预估前沿-FM

$$\mathrm{p}(click|a,u,c)=\frac{1}{1+e^{-f(x)}} \qquad f(x)=w_0+\sum_{i=1}^{n}w_ix_i+\sum_{i=1}^{n}\sum_{j=i+1}^{n}<\vec{v_i},\vec{v_j}>x_ix_j$$

采用向量化分解的方式来描述特征i-特征j的pairwise耦合信息，而非用单点实数$w_{i,j}$。

$$W=\begin{bmatrix} w_{11} & w_{11} & \dots & w_{11} \\ w_{21} & w_{22} & \dots & w_{2n} \\ & & \dots & \\ w_{n1} & w_{n2} & \dots & w_{nn} \end{bmatrix}=VV^T=\begin{bmatrix} v_1^Tv_1 & v_1^Tv_2 & \dots & v1^Tv_n \\ v_2^Tv_1 & v_2^Tv_2 & \dots & v2^Tv_n \\ & & \dots & \\ v_n^Tv_1 & v_n^Tv_2 & \dots & v_n^Tv_n \end{bmatrix}$$

$$\begin{aligned}
&\sum_{i=1}^{n}\sum_{j=i+1}^{n}\langle \mathbf{v}_i,\mathbf{v}_j\rangle x_i x_j \\
=&\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\langle \mathbf{v}_i,\mathbf{v}_j\rangle x_i x_j-\frac{1}{2}\sum_{i=1}^{n}\langle \mathbf{v}_i,\mathbf{v}_i\rangle x_i x_i \\
=&\frac{1}{2}\left(\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{f=1}^{k}v_{i,f}\,v_{j,f}\,x_i x_j-\sum_{i=1}^{n}\sum_{f=1}^{k}v_{i,f}\,v_{i,f}\,x_i x_i\right) \\
=&\frac{1}{2}\sum_{f=1}^{k}\left(\left(\sum_{i=1}^{n}v_{i,f}\,x_i\right)\left(\sum_{j=1}^{n}v_{j,f}\,x_j\right)-\sum_{i=1}^{n}v_{i,f}^2\,x_i^2\right) \\
=&\frac{1}{2}\sum_{f=1}^{k}\left(\left(\sum_{i=1}^{n}v_{i,f}\,x_i\right)^2-\sum_{i=1}^{n}v_{i,f}^2\,x_i^2\right)
\end{aligned}$$

ctr预估前沿-FFM

Fm会引入同一类特征的组合，比如user-user，ad-ad，是否可以指定哪类特征与哪类特征的embedding矩阵去做交互呢？引入field-aware概念。

$$\sum_{i=1}^{n}\sum_{j=i+1}^{n} < \vec{v_i}, \vec{v_j} > x_i x_j \quad \rightarrow \sum_{i=1}^{n}\sum_{j=i+1}^{n} < \overrightarrow{v_{i,f_j}}, \overrightarrow{v_{j,f_i}} > x_i x_j$$

举例，chao：user, jian：user, comedy：genre, drama：genre。chao和comedy的交互特征表达为：

$$< V_{chao,genre}, V_{comedy,user} > x_{chao} x_{comedy}$$

模型复杂度较fm显著提升，除比赛外至今工业界暂无大规模使用。

ctr预估前沿-FFM

For FM, $\phi(\mathbf{w}, \mathbf{x})$ is:

$$
\begin{aligned}
\langle \mathbf{w}_1, \mathbf{w}_2 \rangle \cdot 1 \cdot 1 + \langle \mathbf{w}_1, \mathbf{w}_3 \rangle \cdot 1 \cdot 1 + \langle \mathbf{w}_1, \mathbf{w}_4 \rangle \cdot 1 \cdot 1 + \langle \mathbf{w}_1, \mathbf{w}_5 \rangle \cdot 1 \cdot 9.99 \\
+ \langle \mathbf{w}_2, \mathbf{w}_3 \rangle \cdot 1 \cdot 1 + \langle \mathbf{w}_2, \mathbf{w}_4 \rangle \cdot 1 \cdot 1 + \langle \mathbf{w}_2, \mathbf{w}_5 \rangle \cdot 1 \cdot 9.99 \\
+ \langle \mathbf{w}_3, \mathbf{w}_4 \rangle \cdot 1 \cdot 1 + \langle \mathbf{w}_3, \mathbf{w}_5 \rangle \cdot 1 \cdot 9.99 \\
+ \langle \mathbf{w}_4, \mathbf{w}_5 \rangle \cdot 1 \cdot 9.99
\end{aligned}
$$

For FFM, $\phi(\mathbf{w}, \mathbf{x})$ is:

$$
\begin{aligned}
\langle \mathbf{w}_{1,2}, \mathbf{w}_{2,1} \rangle \cdot 1 \cdot 1 + \langle \mathbf{w}_{1,3}, \mathbf{w}_{3,1} \rangle \cdot 1 \cdot 1 + \langle \mathbf{w}_{1,3}, \mathbf{w}_{4,1} \rangle \cdot 1 \cdot 1 + \langle \mathbf{w}_{1,4}, \mathbf{w}_{5,1} \rangle \cdot 1 \cdot 9.99 \\
+ \langle \mathbf{w}_{2,3}, \mathbf{w}_{3,2} \rangle \cdot 1 \cdot 1 + \langle \mathbf{w}_{2,3}, \mathbf{w}_{4,2} \rangle \cdot 1 \cdot 1 + \langle \mathbf{w}_{2,4}, \mathbf{w}_{5,2} \rangle \cdot 1 \cdot 9.99 \\
+ \langle \mathbf{w}_{3,3}, \mathbf{w}_{4,3} \rangle \cdot 1 \cdot 1 + \langle \mathbf{w}_{3,4}, \mathbf{w}_{5,3} \rangle \cdot 1 \cdot 9.99 \\
+ \langle \mathbf{w}_{4,4}, \mathbf{w}_{5,3} \rangle \cdot 1 \cdot 9.99
\end{aligned}
$$

ctr预估前沿-GBDT

优点：非线性模型，无需做特征归一化/组合/选择。可以适应多种损失函数。

原理：boosting(降低bias)迭代构造一组树形式的加性弱(控制variance)学习器。

$$g_m(x_i) = -\frac{\partial L(y_i, F)}{\partial F}\Big|_{F=F_{m-1}}$$

决策树拟合函数空间下降方向

$$(\alpha_m, \beta_m) = arg\min_{\alpha_m, \beta_m} \sum_{i=1}^{N} L(y_i, F_{m-1}(x_i) + \beta h(x_i; a))$$

α控制下降方向，β控制步长

传统gbdt和xgboost的详细推导参见王超-xgboost导读和实战。

ctr预估前沿-MF

对比FM，简要做下介绍，重点介绍Mxnet。$r_{\widehat{ui}} = {q_i}^T p_u$

```
user = mx.symbol.Embedding(data = user, input_dim = max_user, output_dim = 1000)
user = mx.symbol.Flatten(data = user)
user = mx.symbol.FullyConnected(data = user, num_hidden = hidden)
item = mx.symbol.Embedding(data = item, input_dim = max_item, output_dim = 1000)
item = mx.symbol.FullyConnected(data = item, num_hidden = hidden)
item = mx.symbol.Flatten(data = item)
pred = user * item
pred = mx.symbol.sum_axis(data = pred, axis = 1)
pred = mx.symbol.Flatten(data = pred)
pred = mx.symbol.LinearRegressionOutput(data = pred, label = score)
```

引用自项亮一文用Mxnet实现矩阵分解一文，自动求导(前面没推出公式的不用再推公式了）。

前述模型的融合

Bagging(Bootstrap Aggregating)：
降低variance，比赛中常用，将多个强分类器的结果求平均。主要目的是防止单模型过拟合。

Stacking：
如Facebook的gbdt + lr，目的是提升模型的非线性表达能力，也可以gbdt + fm。主要目的是利用gbdt自动化一些特征工程的工作。

Boosting：
如前述gbdt。多个弱分类器组成一个强分类器。

ctr预估前沿-DNN

前面都说错了，现在是NN的天下，前面的都不算是前沿。

训练工程架构演进 BSP->SSP

BSP(Bulk Synchronous Parallel)：

- LBFGS，OWLQN，ADMM…
- BSP底层架构：MPI AllReduce
- 优点：单轮迭代快
- 缺点：迭代轮数多

- SSP(Staleness Synchronous Parallel)：
  - minibatch SGD
  - SSP底层架构：async Push-Pull
  - 优点：minibatch SGD扫一遍数据更新多次模型，架构优雅，解非凸问题
  - 缺点：数据量少的时候优势相对不明显