ffm优化加速效果和原理

wangpeng@zenmen.com
Last updated on 2017-08-15

效果总结

速度

单条样本ffm模型训练和预测时间复杂度从$O(nnk)$降为$O(nfk)$，其中n是样本非零特征数，f是field个数，k是隐向量长度。

在我们的广告ctr预估数据集上，每个样本的n平均在57左右，f目前为3，k=8，使用单线程对比优化前后速度， 优化后速度提升8.5倍，小于$n/f = 19$可能是优化后有个预计算过程，加上预计算过程时间复杂度是$O(2nfk)$，对于preidct加上预计算时间复杂度是$O(nfk + ffk + nk)$，所以在线预估ctr估计加速了10到15倍（这个加速没包含特征提取部分），从线上结果来看，特征提取耗时应该占大头。

稳定性

实验对比发现优化后的算法，收敛更快，更稳定，loss更低点（低千分之几）。见最后一小节理论解释。这个测试还不充分，待后续发论文时，进一步用公开数据集测试。

用了170万条样本的小数据测试了两次，验证集比例分别为6%和30%，一共迭代15轮，本发明优化版均在第3或4次迭代时在验证集上收敛到最优值，并且误差都是先减小在增大这个稳定过程， 而原版在15轮迭代，在验证集上误差多次（3到4次）反复变大变小。

其他

线上50%流量对比效果，5天数据平均，ffm模型比lr收入提升8%左右（ffm模型本身因素，跟本文优化无关）

cpm提升：8.0783/7.4821 = 1.0796835113137

用的8月12 到 16 号5天数据，17号有秒拍固定位置影响没有统计，也差不多这个比例

背景知识

LR

$$\theta(w, x) = \sum_{i=1}^{n} w_i x_i \qquad (1)$$

$$ctr = \frac{1}{1+e^{-\theta(w,x)}} \qquad (2)$$

上式中w是权重向量，即模型训练时要求解的参数，w_i是一个实数，x是样本特征，可以认为"color:red"，"interest:football"，这样每个不同的"属性:值"对就是一个不同特征，x_i取1代表存在这个特征，对于连续特征x_i是一个实数，很多时候是将连续特征经过人工或gbdt等非线性处理变为离散特征使用。

由于LR模型本身是线性的，特征之间无交叉，各个特征之间可以认为是独立训练。而现实中各个特征一般存在紧密关系，比如用户的性别特征，跟广告的行业特征存在一些紧密关系，年轻女性喜欢美颜，拍照，可以打美拍app，面膜之类广告。**所以为了达到更好的效果，常常利用人工组合特征生成一些非线性特征**（还有gbdt等方法）。

$$\theta(w,x) = \sum_{i=1}^{n} w_i x_i + \sum_{\{(i,j)\,|\,(i,j)\in P\}} w_{i,j} x_i x_j \qquad (3)$$

上面公式中后面的一项表示人工组合的特征（为了描述方便，只考虑二阶），P是人工组合的特征集合，只有$x_i x_j$都非0时，这个组合特征才存在，$w_{i,j}$表示对于组合特征(i,j)的参数。

当特征x_i, x_j都比较稀疏时，组合特征(i,j)就更稀疏，在训练时$w_{i,j}$得不到充分训练，容易出现过拟合的情况，所以一般都是训练前丢掉低频特征，但这种组合特征方式还是存在两个问题：

- **丢掉的低频特征可能含有不少有价值信息**

  对于丢掉的单个组合特征来说，由于是低频，丢掉数量比较少，但特征不同组合方式有很多，所以总的丢弃的特征数量可能很大。这里面可能包含大量有价值信息。

  假设有1000个机型，每个机型出现概率相等，同时有100个广告行业，出现概率也相等，那机型-广告行业交叉特征出现概率10万分之一，很容易就被过滤掉了，但几乎每一个样本都会同时有这两个特征。
- **缺乏泛化能力**

  假设机型-A和广告行业B出现频次比较高，进入模型训练出参数权重，当在线预估ctr时，一条流量同时含有A-B特征，那就可以利用这个权重，如果流量是A-C特征，这个组合特征在模型里面没有（可能训练样本没出现过，或被低频过滤），那就无法利用A-C特征对CTR进行精细预估。

MF(matrix factorization)

假设通过lr训练出来了各个权重$w_{i,j}$，用W表示各个组合特征的权重，则W是对称矩阵。假设W正定，则W可以分解为如下形式：

$$W = P \wedge P^T \qquad (4)$$

其中P是W的特征向量组成的正交矩阵，$\wedge$是特征值组成的对角矩阵，特征值都大于0，选取前k个最大特征值及对应的特征向量可以将W近似表示成，

$$W \approx VV^T \qquad (5)$$

其中$V = (\sqrt{\lambda_1} p_1, \sqrt{\lambda_2} p_2, \ldots, \sqrt{\lambda_k} p_k)$
这样$W_{i,j} \approx V_i . V_j$，其中$V_i, V_i$均是$k$维向量

- **通过低秩近似，保留主要信息的同时，去掉了小特征值对应的噪声或不稳定成分**
- **特征组合权重由两个向量点击取得，这样可以泛化到其他未参与训练的组合特征**

FM(Factorization Machines)

FM模型公式如下，省去一阶项：

$$\theta(w,x)=\sum_{i=1}^{n}\sum_{j=i+1}^{n}(w_i.w_j)x_ix_j \quad (6)$$

其中w_i, w_j是k维向量，k由用户指定，特征之间的交互权重由两个特征对应向量的点积表示，这样：

- **解决了模型训练时，组合特征样本太少，无法充分训练问题**
- **训练和预测都是线性时间复杂度$O(nk)$**
- **相比LR，省去了人工组合特征的耗时耗力**

FFM(Field-aware Factorization Machines)及优化

$$\theta(w,x)=\sum_{i=1}^{n}\sum_{j=i+1}^{n}(w_{i,f_j}.w_{j,f_i})x_ix_j \quad (7)$$

FM是每个特征一个向量，**FFM**加入域的概念，每个特征都归属于一个域，并且每个特征对每个域有一个不同向量，跟不同域的特征做点积时，使用那个特征对应的域的向量。

- **对每个参数来说，可用训练数据没有变稀疏，参数仍然与FM一样得到充分训练**
  一般来说一个样本至少含有同一个域的一个特征，相当于训练数据没有变稀疏，即一般来说假设$w_{i,k}$在**FM**的一个样本中可以得到训练，$w_{i,f,k}$在**FFM**的这个样本中可以得到训练
- **与不同域特征交互时使用单独向量，更精细化特征交互**
  假设同样的数据上跑**FM**和**FFM**模型，k相同，$f>1$，**FFM**，更多参数，并且在上面一条的解释下，每个参数跟**FM**一样能得到充分训练，一般效果自然更好

predict

原始predict

$$\theta(w,x)=\sum_{i=1}^{n}\sum_{j=i+1}^{n}(w_{i,f_j}.w_{j,f_i})x_ix_j \quad (8)$$

计算复杂度是$O(nnk)$

优化后predict

$$
\begin{cases}
\theta(w,x) = \sum_{i=1}^{n}\sum_{j=i+1}^{n}(w_{i,f_j}\cdot w_{j,f_i})x_i x_j \\
\quad = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}(w_{i,f_j}\cdot w_{j,f_i})x_i x_j - \frac{1}{2}\sum_{i=1}^{n}(w_{i,f_i}\cdot w_{i,f_i})x_i x_i \\
\quad = \frac{1}{2}\sum_{i=1}^{n}\sum_{f_b=1}^{f} x_i w_{i,f_b}\cdot\left(\sum_{\{j|f_j=f_b\}} x_j w_{j,f_i}\right) - \frac{1}{2}\sum_{i=1}^{n}(w_{i,f_i}\cdot w_{i,f_i})x_i x_i \\
\quad = \frac{1}{2}\sum_{f_a=1}^{f}\sum_{f_b=1}^{f}\left(\sum_{\{i|f_i=f_a\}} x_i w_{i,f_b}\right)\cdot\left(\sum_{\{j|f_j=f_b\}} x_j w_{j,f_a}\right) - \frac{1}{2}\sum_{i=1}^{n}(w_{i,f_i}\cdot w_{i,f_i})x_i x_i
\end{cases} \qquad (9)
$$

上面公式中$\sum_{\{i|f_i=f_a\}} x_i w_{i,f_b}$可以预先计算出来，时间复杂度是$O(nfk)$,然后在计算$\theta(w,x)$,时间复杂度是$O(ffk+nk)$,所以总的时间复杂度是$O(nfk)$。

train

假设$loss(\theta(w,x))$为误差损失函数，train就是一个寻找最优w的过程，为了描述简单，省略正则化项：

$$
min_w \sum_i loss(\theta(w,x_i)) \qquad (10)
$$

对w求导，得出梯度

$$
\frac{\partial loss(\theta(w,x))}{\partial w_{i,f}} = \frac{\partial loss(\theta(w,x))}{\partial \theta(w,x)}\frac{\partial \theta(w,x)}{\partial w_{i,f}} \quad (11)
$$

其中$\frac{\partial loss(\theta(w,x))}{\partial \theta(w,x)}$被一个样本所有$w_{i,f}$的梯度共用，可以预先计算出来，预计算时间为$\theta(w,x)$的计算复杂度。接下来分别看看作者train和本文优化后train方法的$\frac{\partial \theta(w,x)}{\partial w_{i,f}}$计算时间复杂度。

原始train

ffm作者对$\sum_{i=1}^{n}\sum_{j=i+1}^{n}(w_{i,f_j}\cdot w_{j,f_i})x_i x_j$中的每对$i,j$（一共$n^2$对），计算一次梯度$\frac{\partial (w_{i,f_j}\cdot w_{j,f_i})x_i x_j}{\partial w_{i,f_j}}$和$\frac{\partial (w_{i,f_j}\cdot w_{j,f_i})x_i x_j}{\partial w_{j,f_i}}$并更新一次$w_{i,f_j}$和$w_{j,f_i}$，所以计算时间复杂度是$nnk$

优化后train

对于确定的feature i和field f_b,使用优化后的predict公式对w_{i,f_b}求导得

$$
\begin{cases}
\frac{\partial\theta(w,x)}{\partial w_{i,f_b}} = x_i \sum_{\{j|f_j=f_b\}} x_j w_{j,f_i} & if\ f_i != f_b \\
\frac{\partial\theta(w,x)}{\partial w_{i,f_b}} = x_i \sum_{\{j|f_j=f_b\}} x_j w_{j,f_i} - w_{i,f_b} x_i x_i & if\ f_i = f_b
\end{cases}
\tag{12}
$$

同predict一样，上面公式中$\sum_{\{j|f_j=f_b\}} x_j w_{j,f_i}$可以预先计算出来，时间复杂度是$O(nfk)$,然后在计算所有w的梯度,时间复杂度是$O(nfk)$,所以总的时间复杂度是$O(nfk)$，而原始公式计算复杂度是$O(nnk)$。

优化前后train收敛速度和稳定性分析

ffm作者train方法存在的问题：

1 对于随机选取的一个样本，计算$\frac{\partial loss(\theta(w,x))}{\partial\theta(w,x)}$和计算$\frac{\partial(w_{i,f_j} \cdot w_{j,f_i})x_i x_j}{\partial w_{i,f_j}}$时，所使用的$w$值不一样，并且对于特定的参数$w_{i,f_j}$在一个样本里面也会被多次更新，每次更新使用的$w$值也不一样，这样导致梯度计算不准确。首先为了效率考虑，一个样本只计算一次$\frac{\partial loss(\theta(w,x))}{\partial\theta(w,x)}$（记为$g_{loss}$），记计算时所用的$w$为$w^l$，后续求梯度$\partial w_{i,f_j}$时，复用这个$g_{loss}$值，但由于采用Hogwild!算法多线程同时更新$w$，导致即使对于同一个样本，求$\partial w_{i,f_j}$时用的$w$每次可能都不同，与$w^l$不一致，导致计算出来的梯度各个维度大小都与单线程$SGD$不同，并且当一个feature i与其他多个同一个field（f_b）的feature相交时，会多次求$\partial w_{i,f_b}$和更新w_{i,f_b}，这时某次求出的$\partial w_{i,f_b}$可能与整个样本作为一个batch求出的$\partial w_{i,f_b}$方向相反，即$\frac{\partial(w_{i,f_j} \cdot w_{j,f_i})x_i x_j}{\partial w_{i,f_j}}$与$\frac{\partial\theta(w,x)}{\partial w_{i,f_b}}$的正负号可能不同，在多线程情况下，不准确的梯度方向和大小，可能会导致train收敛不稳定。Hogwild！算法里面有写明"Read current state x_e and evaluate $G_e(x)$"，即对于一个样本是先读取w当前值，再用当前w值，进行下一步计算梯度，所以对于一个样本求解过程中，用的w是固定的。

2 原始train方法，AdaGrad累加梯度有偏，会使含特征多的field对应的AdaGrad因子累加更快，使其不能充分利用样本得到充分训练。原始train方法是每个样本每对特征i,j，计算一次梯度$\frac{\partial(w_{i,f_j} \cdot w_{j,f_i})x_i x_j}{\partial w_{i,f_j}}$和$\frac{\partial(w_{i,f_j} \cdot w_{j,f_i})x_i x_j}{\partial w_{j,f_i}}$并更新一次$w_{i,f_j}$和$w_{j,f_i}$，同时AdaGrad也累加一次，所以对于含特征数更多的field（记为f），AdaGrad累加更快，后面的样本对w中f相关部分影响就更小

Future Work

- group lasso
- AdaGad –> Adadelta，Adam等
- 在cpu抗的住的情况下，调节参数f,k及训练参数等看看效果，之前试的f及其分组方式影响也比较大
- 目前做了些工程化加速，但离线训练还有不少可以优化加速的地方
- ……