推荐系统与深度学习（十一）——xDeepFM模型原理

原创 livan 数据python与算法 9月2日

模型介绍

随着深度学习发展，高维特征的价值凸显出来，能够更清晰的表示特征交互，当输入数据的内容变得丰富时，就需要高阶的交叉特征。

于是有了很多高维特征交互方法，开始的特征交互是人工的，比如：Deep&Wide等，这一方式的问题在于：

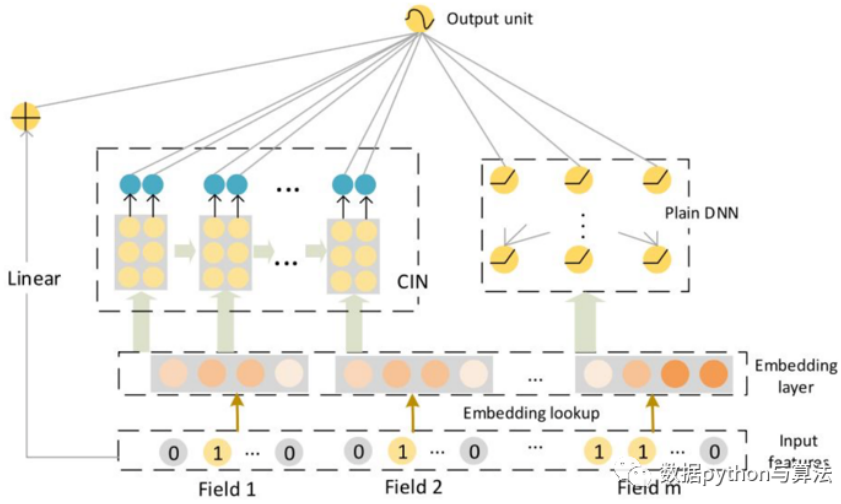
1） 工程师需要花费大量时间了解数据，提出高维交叉特征，人力成本比较昂贵；

2） 由于稀疏的问题，特征交叉容易产生维度灾难；

3） 泛化能力较弱；

因此，很多大佬提出了FNN、PNN、DeepFM等支持自动特征交叉的模型，但由于其中深度模型的部分，特征交互后无法显性呈现，特征不可控等特点，引发工程师的病垢，紧接而来的大佬发明了xDeepFM和DCN，本文我们来看一下xDeepFM。

虽然叫xDeepFM，但xDeepFM更像是DCN的升级版，其总结构图为：



从图中我们看到，xDeepFM模型是将DeepFM中的FM部分修改成了CIN部分，同时添加了线性部分，公式为：

$$\hat{y} = \sigma(\mathbf{w}_{linear}^T \mathbf{a} + \mathbf{w}_{dnn}^T \mathbf{x}_{dnn}^k + \mathbf{w}_{cin}^T \mathbf{p}^+ + b)$$

这一修改的优势在于：

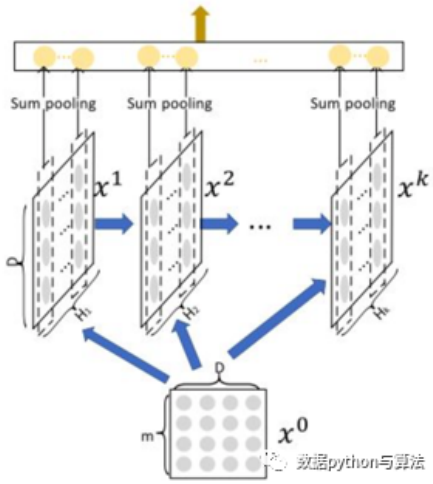
1） 显式呈现特征交互，特征交互的形式可控；

2） 特征交互的发生是在向量级，而不是元素级别；

模型的整体部分比较清晰，数据one-hot之后进入Embedding层，而后数据同时进入到CIN和DNN模型中，在两个并行的模型中训练之后，融合到sigmod函数中，最终形成输出结果。

我们来深度了解一下CIN的结构：

CIN中的每一层神经元都是根据前一层的隐层以及原特征向量推算而来，如图：



这一结构的特征是最终学习的阶数由网络模型的层数决定，每一层隐层都通过一个池化层连接到输出层，从而保证输出单元能够见到不同阶数的特征交互模式，这一结构与RNN的结构很类似，即每一层的状态是由前一层隐层的值与一个额外的输入数据计算所得，不同的是CIN中额外的输入数据是固定的x0。

上图每层神经层的计算公式为：

$$X_{h,*}^{k} = \sum_{i=1}^{H_{k-1}} \sum_{j=1}^{m} W_{ij}^{k,h} (X_{i,*}^{k-1} \circ X_{j,*}^{0})$$

从公式可以看出是向量级别的特征交互。

对应的计算顺序为：

第一层x1的第h个神经元向量为：

$$\mathbf{x}_h^1 = \sum_{\substack{i\in[m] \\ j\in[m]}} \mathbf{W}_{i,j}^{1,h} (\mathbf{x}_i^0 \circ \mathbf{x}_j^0)$$

以此类推，第二层x2的第h个神经元向量为：

$$\begin{aligned} \mathbf{x}_h^2 &= \sum_{\substack{i\in[m] \\ j\in[m]}} \mathbf{W}_{i,j}^{2,h} (\mathbf{x}_i^1 \circ \mathbf{x}_j^0) \\ &= \sum_{\substack{i\in[m] \\ j\in[m]}} \sum_{\substack{l\in[m] \\ k\in[m]}} \mathbf{W}_{i,j}^{2,h} \mathbf{W}_{l,k}^{1,i} (\mathbf{x}_j^0 \circ \mathbf{x}_k^0 \circ \mathbf{x}_l^0) \end{aligned}$$

……

最后，第k层的第h个神经元向量为：

$$\mathbf{x}_h^k = \sum_{\substack{i\in[m]\\ j\in[m]}} \mathbf{W}_{i,j}^{k,h}(\mathbf{x}_i^{k-1} \circ \mathbf{x}_j^0)$$

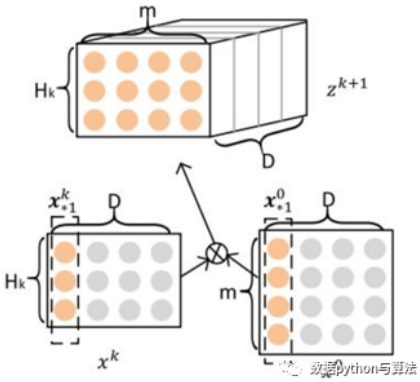
$$= \sum_{i\in[m]} \cdots \sum_{r\in[m]} \sum_{l\in[m]} \mathbf{W}_{i,j}^{k,h} \ldots \mathbf{W}_{l,s}^{1,r}(\underbrace{\mathbf{x}_j^0 \circ \ldots \circ \mathbf{x}_s^0 \circ \mathbf{x}_l^0}_{\text{vectors}})$$

看到这些公式有没有一些头大，感觉多少会有吧~

我们用图形样式来直观的描述一下CIN的计算过程：

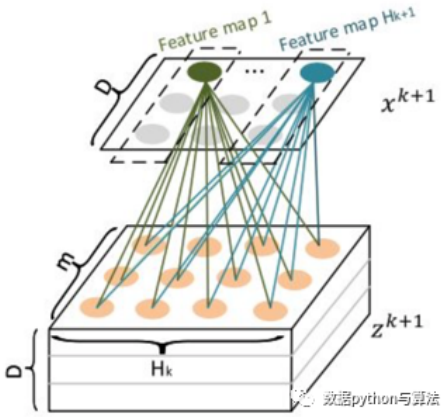
CIN主要分两步：

第一步：



上图中的两矩阵计算相当于（X_{*1}^k内积X_{*1}^0）计算，即把上一层隐层的输出结果xk和原始的x0进行组合，组合的方式是一种内积的形式。黄点位置是对应的一次内积计算(3点 **内积** 4点 **得到**12点)，可以通过单向量与单向量的计算得到二维的结果。通过多次（D次）内积就得到上面厚度为D的三维的矩阵，这个三维矩阵即为第一步的结果集（Hk，m，D）。

第二步：



此步即将第一步中的结果集（Hk，m，D）进行卷积运算，卷积矩阵即为w，m*Hk是原特征图的维度，卷积核的大小也是m*Hk，一个卷积矩阵一个值，D个卷积运算之后，生成 D*1个数值，因为D是厚度，一次相当于生成一个向量。这样使用Hk个卷积核相当于生成的特征图为Hk*D大小的矩阵，即为xk+1层。

经过这一描述，xDeepFM的结构大家清晰了吧~

欢迎大家关注公众号：

数据python与算法

微信扫描二维码，关注我的公众号

数据python与算法

来都来了，点个关注再走呗~