xDeepFM理论与实践

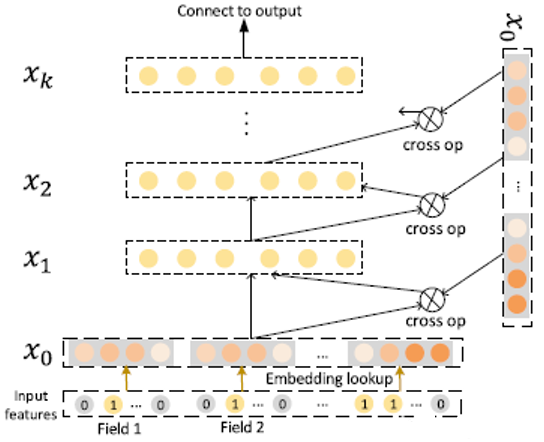
# 前言

特征工程是很多预测任务成功与否的关键，然而其过程不仅耗时耗力，而且需要对业务有较深的理解。基于FM的模型，可以自动学习特征的交叉和组合。DNN依靠神经网络强大的学习能力，可以在一定程度上实现自动学习特征组合,但是很难学到所有的组合特征。于是，业界探索和实践了诸多FM与DNN相结合的CTR预估模型。如Wide & Deep、Deep & Cross Network、DeepFM等，这些模型可以同时学习低阶和高阶组合特征。

在xDeepFM中，作者设计了一种新型的网络Compressed Interaction Network(CIN), 可以像Deep & Cross Network(DCN)那样随着Cross Network层数的递增，特征的交叉维度也会递增。并弥补了DCN存在的一些不足。xDeepFM最出名的在于它的特征交叉学习部分,也就是CIN层，可谓是一种艺术般的交叉。其也在海量的数据竞赛中展现了不俗的成绩。

# DCN的局限性

DCN的Cross Network架构如下所示（只是cross network部分，不是全部DCN）：

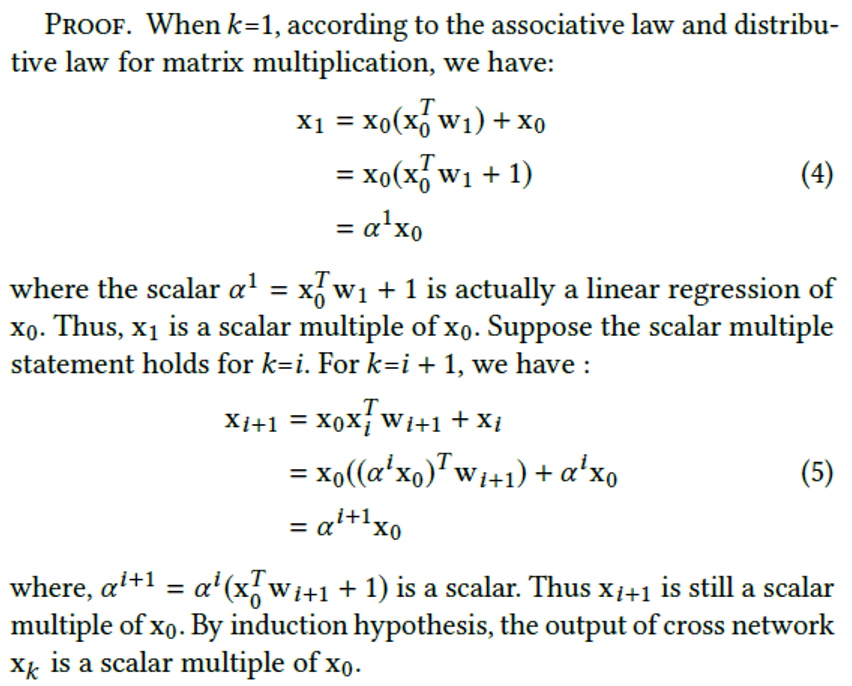


公式如下：



其中w，b和x分别表示网络的weights，bias和output，k表示第k层。

Cross Network可以显示地学习到高维的特征组合，但它存在一个问题，就是最终的xk 结果是x0乘以一个标量(scalar)。



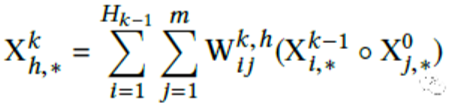
虽然xk结果是x0乘以一个标量，但是并不意味着x0和xk是线性关系的，因为对于不同的x0，这个标量是不一样的。 因此，由于这个的局限性，限制了Cross Network的表达。

# Compressed Interaction Network(CIN)

## CIN

了解Cross Network的局限性后，paper作者提出了Compressed Interaction Network(CIN)。

假设输入一个有m个field(raw feature)，首先将这m个field都转换成都是D维的 embedding。则可以得到一个m\*D维的embedding矩阵，用X0来表示。 因此对于第k层Xk，使用下面的公式进行计算

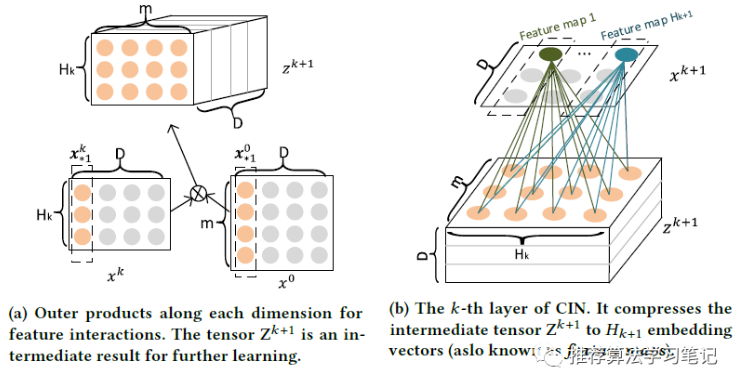


表示第k层第h个field对应的embedding，是一个scalar，空心圆表示Hadamard product，例如⟨a1, a2, a3⟩◦⟨b1,b2,b3⟩ = ⟨a1b1, a2b2, a3b3⟩

这个公式就是将第k-1层的embeding和第0层的embedding两两做Hadamard product，然后weight sum后输出一个embedding。同时，我们可以决定第k层有多少个field，使用Hk表示field的个数。

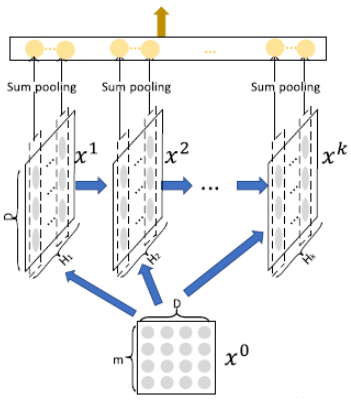
## CIN和CNN的关系

CIN其实是可以转换成CNN的，如下图所示。具体细节可参见paper。

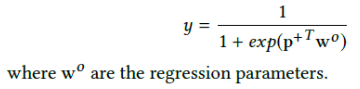


## CIN输出

在得到每一层的Xk输出后，将每个field（D维向量）的元素相加起来，得到一个Hk维的向量作为输出。所有层的输出构成一个sum pooling层，如下图所示



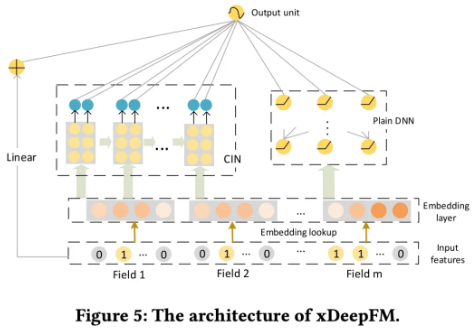
最后sum pooling层使用sigmoid函数作为最终输出，公式如下所示



# xDeepFM

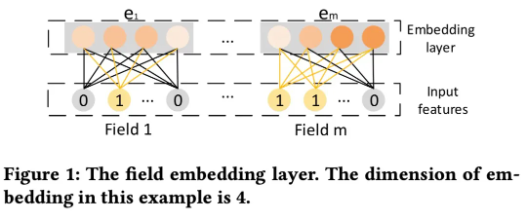
将CIN和DNN结合起来，构造一个更强大的模型，这就是xDeepFM。

整体架构图如下所示



从上图中可以发现xDeepFM可以细分为三大块,一块是基于底层特征的线性部分，一块是基于特征Embedding的DNN部分(implicit feature interactions)以及CIN部分(Explicit feature interactions)。我们按照该图看看模型每一步都在做什么，尤其是CIN层做了哪些操作，为什么能在诸多数据竞赛中拿到相较于DeepFM等模型那么大的优势。

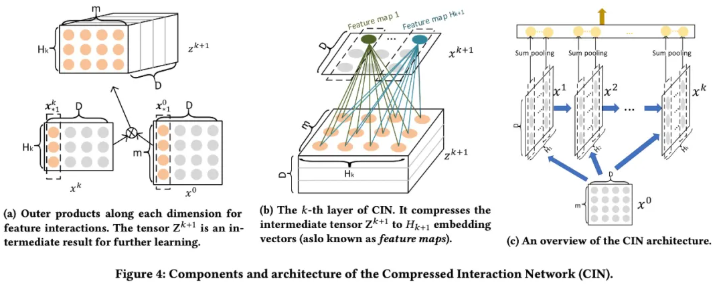
## Embedding Layer



此处embedding做的事情就是将传统的单个特征映射到一个D维的dense特征上,假设有m个field,最终我们将得到：



## CIN(Compression Interaction Network)



xDeepFM在CIN层实现了特征的显示交叉，究竟是如何做到的呢？就是M层 =(M-1)层 + 0层的思路，即第M阶的交叉特征是由(M-1)层的特征和第0层(原始特征)交叉得到的。具体地，假设第0层的原始特征为, ,同时假设第k层的特征为,其中Hk为第k层的网络的特征向量个数,所以H0=m,那么要想得到第k层的特征,就可以通过下面的式子进行计算：



其中, , 为第h层特征向量的参数矩阵。也就是说第k层的第h个特征向量是由第(k-1)层的每一个特征向量与第0层的每一个特征向量进行Hadamard乘积然后乘上一个系数矩阵最后全部相加得到的。所以说特征交叉是显示的。

为了能显示利用到每一层的交叉特征,最后需要将每一层的交叉特征输出,但是如果我们直接全部输出的话,可能会带来一个较大的问题,就是特征太多了，后面再接入全连接层的话会占据更多的内存和计算资源。所以我们使用sum pooling。这样第k层第i个向量的输出为：



那么第k层的输出为：



最终CIN的输出为：



1. CIN与RNN的关系

CIN中下一层的输出都依赖于上一层的输入以及额外的输入，和RNN是非常相似的。

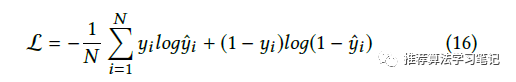
2. CIN与CNN的关系

我们发现第k层的每一个新的向量都是由第(k-1)层的所有向量以及第0层的所有向量分别进行element-wise 相乘, 然后形成“图像”,我们再使用filter - 与其进行操作得到下一层的新向量，最终将压缩为了Hk个向量,这也是compressed名字的由来。

公式如下所示

https://mmbiz.qpic.cn/mmbiz_png/OibBxV7NSmubUC4g8bm420ia0V4uXfc3pWD0By8kzB3oNiaoPiceAh62OJfvb8dsnY0tBaAicic0Erf9qhDoXSo7WNdQ/640?wx_fmt=png&tp=webp&wxfrom=5&wx_lazy=1&wx_co=1

损失函数为

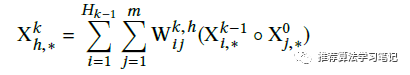


加上L2正则化

https://mmbiz.qpic.cn/mmbiz_png/OibBxV7NSmubUC4g8bm420ia0V4uXfc3pWXm2EFQMklaa0rc2LULBK2lv7a8Uibtn9hUZgDhCJicgEABdPSgkIPP2Q/640?wx_fmt=png&tp=webp&wxfrom=5&wx_lazy=1&wx_co=1

# 和DeepFM的关联

根据公式



将k设为1并且第1层field的个数H1=1，然后sum pooling后，就是DeepFM。可见xDeepFM是DeepFM的泛化版本。

# 总结