深度CTR之xDeepFM：融合了显式和隐式特征交互关系的深度模型推荐系统

原创 大厂机器学习 大厂机器学习实战 3月4日

1 解决的问题

文章发表于KDD 2018，由中科大和微软合作推出。paper-《xDeepFM: Combining Explicit and Implicit Feature Interactions for Recommender Systems》

paper解决的问题如下：

1. 交互特征或者说组合特征非常重要，但是由于数据的多样性等原因，导致通过**手动生成交互特征的成本比较高**。
2. 尽管DNN模型具有从数据中学习任意的函数模式的强大能力，但是**普通的DNN网络只能生成隐式的特征交互关系，而且只能在bit-wise水平**。

文章提出了一个新颖的模型结构-压缩交互网络CIN，CIN可以显式的生成特征交互，并且是**在vector-wise水平生成的**。paper中证明了CIN与CNN和RNN结构具有一些共性，并将CIN与DNN结合，形成了paper的模型结构-xDeepFM，xDeepFM模型不仅**可以显式的生成一定阶数内的特征交互关系，而且可以隐式的生成任意低阶和高阶的特征交互**。

xDeepFM模型在三个数据集上进行了实验，均取得优于当前SOTA模型的效果。

2 介绍部分

首先还是强调特征变换的广泛使用和重要作用，而对于类别特征的**计算交叉积**的变换方式是一种重要的变换方式，计算交叉积可以称为 cross features 或者 multi-way features。

自动学习交互特征的方法，主要是针对传统的生成交互特征的特征工程工作：

1. 获取到有用的特征需要耗费的成本比较高；
2. 对于具有大量原始特征的任务，想要提取到所有的交叉特征是不可行的；
3. 人工设计特征无法泛化到训练样本中未出现的特征。

下面是作者通过对比前人的工作的优缺点，引出了自己的xDeepFM模型结构。

1. FM：由于paper介绍的是xDeepFM模型，所以在说到FM模型时，对于FM模型中的每个特征的embedding向量中的每一个元素，paper中称之为 `bit` 。经典的FM模型可以引入任意高阶的交互特征；
2. AFM：但是建模所有特征的交互关系有可能引入无用的特征，甚至给模型带来噪音；
3. FNN：在DNN之前有使用FM预训练好的field embedding，可以学习高阶特征；
4. PNN：在embedding层和DNN层之间有product层，而且不需要依赖预训练。但是PNN和FNN都有共同的缺点-都聚焦在高阶交互特征上而对低阶交互特征关注较少；
5. Wide&Deep：通过引入浅层模块和深层模块的组合结构，使得学习过程具有记忆性和泛化性，可以同时学习到低阶和高阶的交互特征。
6. DeepFM：通过引入浅层模块和深层模块的组合结构构，使得学习过程具有记忆性和泛化性，可以同时学习到低阶和高阶的交互特征；
7. DCN：捕捉有界阶数的交互特征；
8. xDeepFM：以vector-wise形式来显式生成特征交互，使用CIN模块来代替DCN中的cross网络模块，CIN可以显式的学习交互关系，并且随着网络的加深，特征交互关系的阶数也在变大。并且仿照Wide&Deep和DeepFM的组合式网络结构，**xDeepFM组合了显式高阶交互模块、隐式高阶交互模块和传统的FM模块**。

可以看到xDeepFM相比于DeepFM模多了一个显式高阶交互模块。该显式高阶交互模块是在vector-wise层面建模的。

paper中作者提到的三个主要贡献(其中前两个也是本文xDeepFM模型相比于其他模型的最主要特点和优势)：

1. **有效结合了显式高阶交互模块、隐式高阶交互模块，不需要人工特征工程**；
2. **设计了CIN模块可以学习显式高阶交互，并且特征交互阶数往后每一层都会增加，以vector-wise方式代替普通DNN的bit-wise方式**；
3. 在3个数据集上对比了其他的SOTA算法，证明了xDeepFM模型结构的有效性。

> paper中一个有意思的论述式：DNN模型多是在bit-wise层面建模特征交互关系，而FM模型则是在vector-wise层面建模特征交互关系。所以还很难说在推荐领域，DNN模型就是用来建模高阶交互关系的最有效的模型。

3 已有模块的介绍

3.1 Embedding层

paper中也提到了在推荐系统中，不像cv和nlp领域的数据具有时空关联的特性，通常对于大规模级别的推荐系统中，输入特征通常是高维离散特征，因此，对于一个任务中具有多个离散特征特征的的情况也是比较常见的。

Embedding层其实是非常常见的，它的作用就是将原始特征压缩成低维连续特征。通常有两种情况：

1. **单个特征**，例如特征名为'性别'，该特征对应的embedding可以直接用于DNN的输入；
2. **复合特征**，或者说交叉特征，例如[user_id=s02,gender=male, organization=msra,interests=comedy&rock]这种交叉特征，对于每个特征user_id、gender、organization、interests都有他们自身对应的embedding，一般是计算交叉特征中单个特征embedding的sum pooling结果用于DNN的输入。

Embedding层的输出结果一般是表示成如下的wide concatenated vector形式：

$$e = [e_1, e_2, \cdots, e_m], \qquad e_i \in \mathbb{R}^D$$

其中m表示特征的数量，D表示原始特征embedding后的维度。因此尽管每个instance的特征数量是不一样的（具体样本中有些特征取值为空），但是最终每个instance的wide concatenated vector的长度都是一样的，即 `m * D` 。

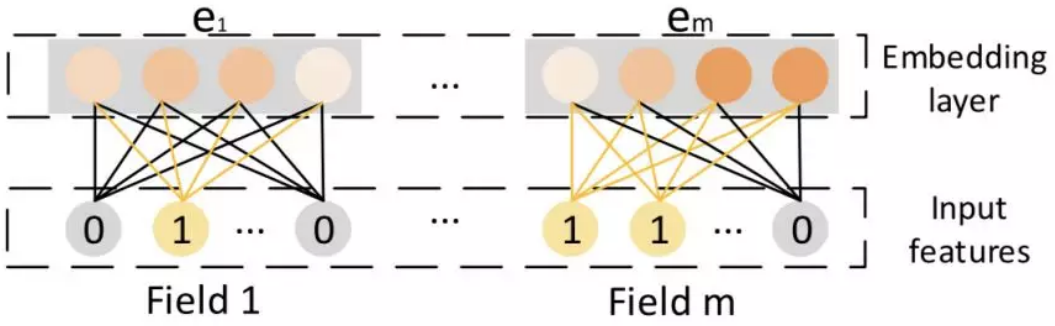


Figure 1: The field embedding layer. The dimension of embedding in this example is 4.

图中的 Field 1 应该是单个特征，而 Field m 应该是复合特征。

> 注意：
>
> 1. paper中将每个原始特征分别当做一个field，例如原始特征一共有10个，那么该数据集的fields数量为10。这里的field跟FFM模型中的field是一个意思;
> 2. paper中每一条样本为为一个instance。

3.2 隐式高阶交互

FNN、DCN、Wide&Deep等模型都是在field embedding向量 e 上使用前馈神经网络充分学习高阶特征交互的信息，前向的计算公式如下：

$$x^1 = \sigma(W^1 e + b^1)$$
$$x^k = \sigma(W^k e^{k-1} + b^k)$$

其中，k表示神经网络层的深度，σ表示激活函数，x^k表示第k层神经网络的输出，这与图2中的深度网络部分比较相似（当然不包括FM层或者Product层）这种前向神经网络的模型架构是以bit-wise的方式来建模交互关系的，也就是说，即使是同一个field embedding向量内的 bit 元素也会互相影响。

而FNN、DeepFM模型对上面的前向神经网络的模型架构进行了略微的调整，除了保持原有的前向DNN结构之外，还会加入一个two-way的交互层，因此**在FNN、DeepFM模型中同时存在bit-wise和vector-wise的特征交互关系**，而FNN、DeepFM结构的主要区别在于FNN是将product的输出连接到DNN上，而DeepFM是将FM层直接连接到输出单元上。

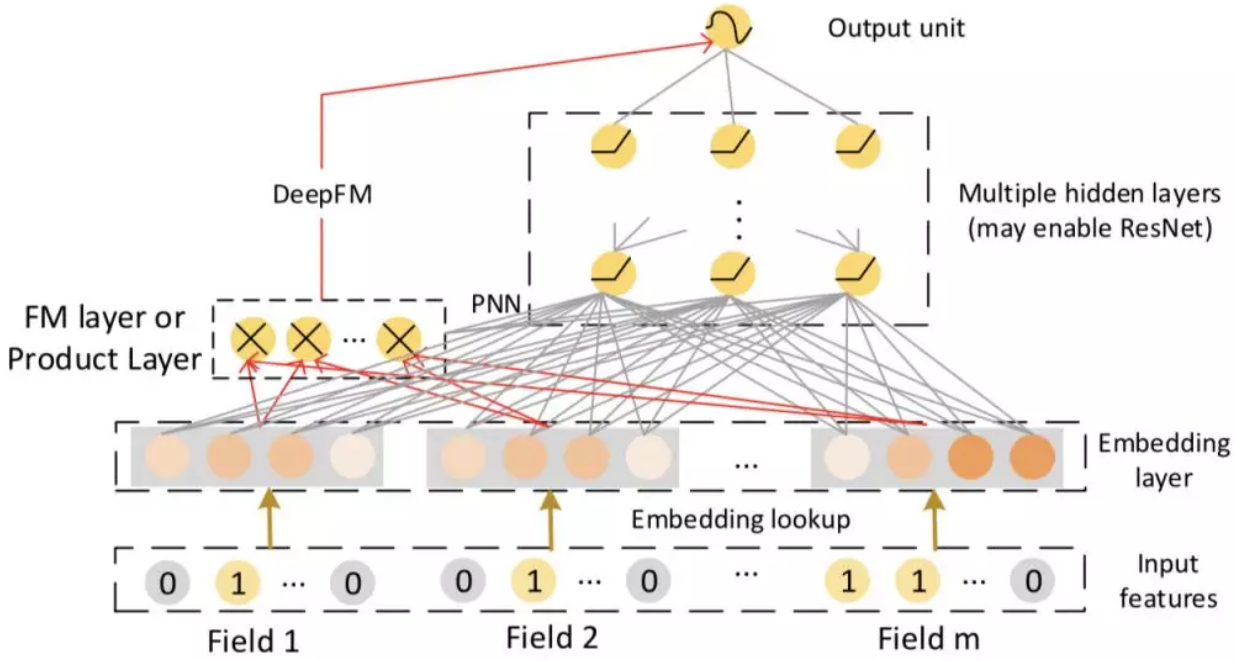


Figure 2: The architecture of DeepFM (with linear part omitted) and PNN. We re-use the symbols in [9], where red edges represent weight-1 connections(no parameters) and gray edges represent normal connections(network parameters).

3.3 显式高阶交互

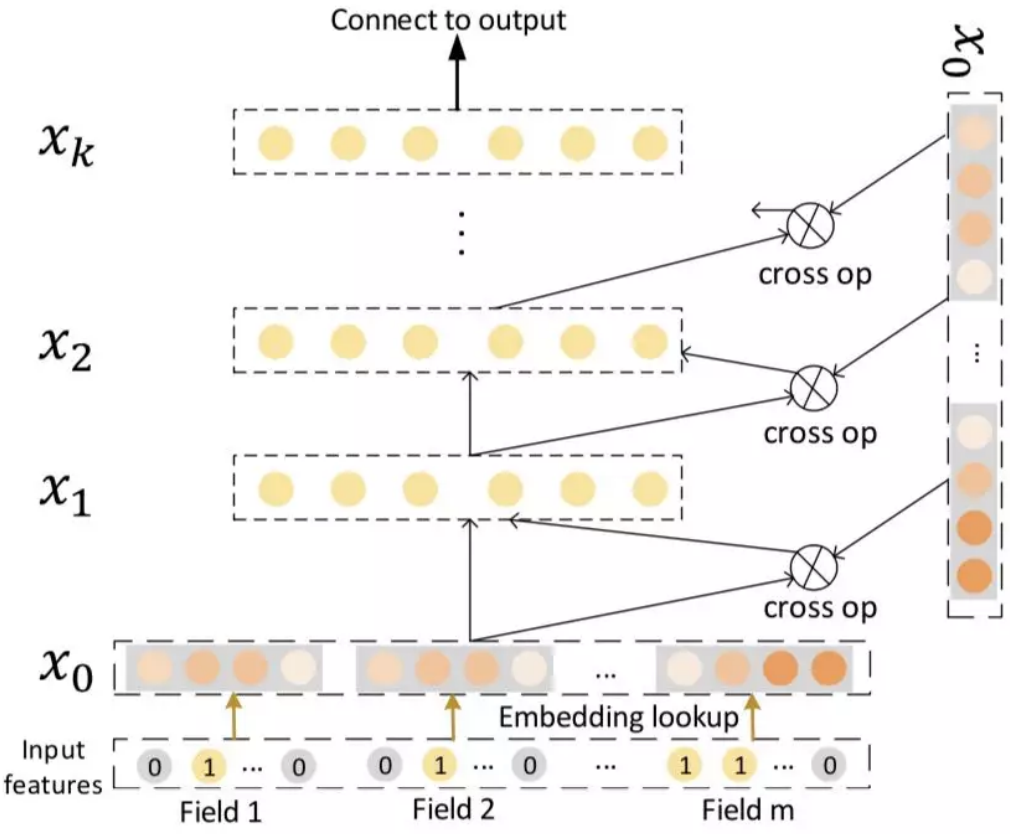


Figure 3: The architecture of the Cross Network.

在如上的DCN模型中，显式的建模了高阶特征，不像前向神经网络只能隐式的建模高阶特征，DCN建模高阶特征公式如下：

$$x_k = x_0 x_{k-1}^T w_k + b_k + x_{k-1}$$

其中CrossNet中的每一个层都是x0的标量倍数，同时paper中也证明了这一理论：

> Theory: 一个k层交叉的网络的第(i+1)层计算公式为：$x_{i+1} = x_0 x_i^T w_{i+1} + x_i$， 那么交叉网络 x_k的输出为x_0的标量倍数。

通过归纳假设证明如下：

1. 当k=1时，

$$\begin{aligned} x_1 &= x_0(x_0^T w_1) + x_0 \\ &= x_0(x_0^T w_1 + 1) \\ &= \alpha^1 x_0 \end{aligned}$$

其中，\alpha^1 就是x0的线性回归形式，因此，x1是x0的标量倍数。

2. 我们假设k=i时的多重标量成立，那么当k=i+1时，

$$x_{i+1} = x_0 x_i^T w_{i+1} + x_i$$

$$= x_0((\alpha^i x_0)^T w_{i+1}) + \alpha^i x_0$$
$$= \alpha^{i+1} x_0$$

其中，

$$\alpha^{i+1} = \alpha^i (x_0^T w_{i+1} + 1)$$

是一个标量，因此x_{i+1}仍然是x0的标量倍数。因此通过上面的归纳假设证明了交叉网络层的输出x_k是x0的的标量倍数的关系。

但是，一个需要注意的问题是，这并不意味着x_k是x_0的线性关系，因为α^{i+1}是关于x0敏感的，CrossNet可以高效学习出来特征交互关系，但是CrossNet也就有其自身的局限性，即：

1. CrossNet的输出只能是指定的形式，即是关于x0的标量倍数；
2. 特征交叉也是也是以bit-wise形式得到的。

4 新模型-XDeepFM

4.1 Compressed Interaction Network

CIN网络设计的优点：

1. 是以vector-wisee而不是bit-wise形式得到特征交叉关系；
2. 可以得到显式的特征高阶交叉关系；
3. 参数容量不会随着网络层数的加深而呈指数形式上升。

embedding向量是看做vector-wise形式的特征交互，我们之后将多个field embedding表示成矩阵，

$$X^0 \in \mathbb{R}^{m \times D}$$

其中，X^0 的第i行向量表示第i个field特征的embedding向量，将其表示成：

$$X^0_{i,*} = e^i$$

D 表示field embedding向量的维度。

而对应的CIN的第k层的输出也是一个矩阵，表示成：

$$X^k \in \mathbb{R}^{H_k \times D}$$

其中，H_k 表示第k层的（embedding）特征向量的数量，且对应的 H_0=m ，对于每一层，X^k 是通过下面的方式计算的，：

$$X^k_{h,*} = \sum_{i=1}^{H_{k-1}} \sum_{j=1}^{m} W^{k,h}_{ij} (X^{k-1}_{i,*} \circ X^0_{j,*}) \qquad (6)$$

其中，

$$1 \le h \le H_k, \qquad W^{k,h} \in \mathbb{R}^{H_{k-1} \times m}$$

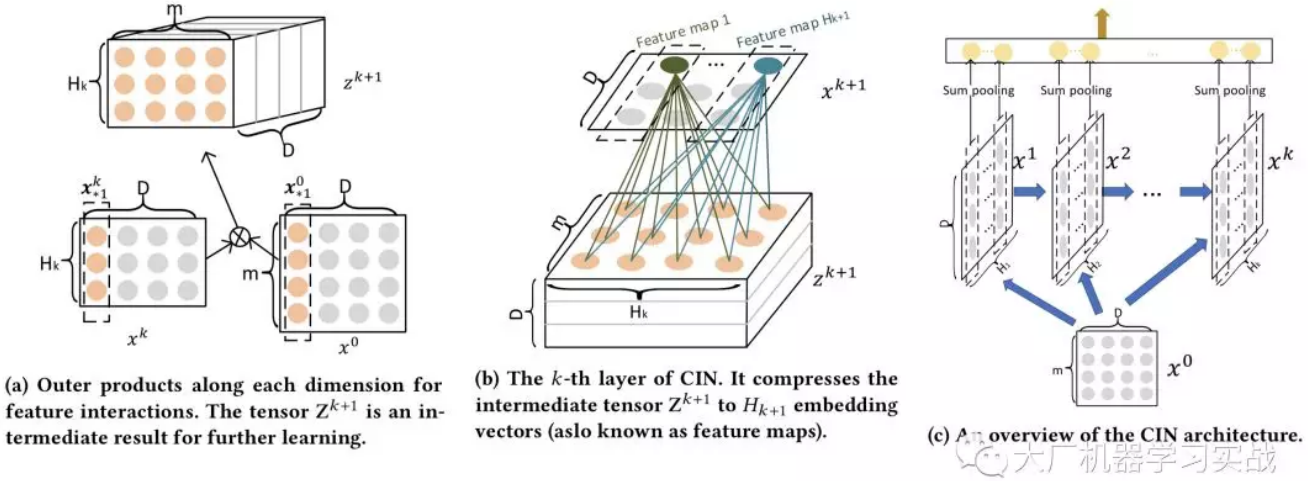
其中，$W^{k,h}$表示用于计算第k层输出中的第h行向量$X^k_{i,*}$的一个参数矩阵，**因此$W^{k,h}_{ij}$是一个标量数值**，即对哈达玛积计算标量倍数。

> 计算符号 ∘ 表示哈达玛积，即两个相同维度的向量的相同位置的元素相乘的值，作为结果中相同位置处的输出元素。

上面需要注意的是，X_k是通过X_{k-1}和X0计算得到的，因此特征交互关系是通过显性的计算的并且特征交互啊的阶数随着层数的增加也在加深。

paper中说到，上面的这种网络结构非常类似于**RNN结构**，即**下一层输出的结果取决于上一层输出的结果和一个额外的输入**，而且我们在每层中都是用这样的结构，因此特征交互关系就是在vector-wise水平上得到的。

下面来看下CIN结构特点：



(a) Outer products along each dimension for feature interactions. The tensor Z^{k+1} is an intermediate result for further learning.

(b) The k-th layer of CIN. It compresses the intermediate tensor Z^{k+1} to H_{k+1} embedding vectors (aslo known as feature maps).

(c) An overview of the CIN architecture.

Figure 4: Components and architecture of the Compressed Interaction Network (CIN).

除了上面说的CIN结构与RNN结构的相似性之外，CIN还与CNN有一定的可类比性，如图4a所示，我们引入一个中间张量(intermediate tensor) Z^{k+1}，它是关于隐藏层X_k和X0的一个outer products（沿着embedding维度），因此可以将Z^{k+1}看做一幅图像，并且 $W^{k,h}$是一个filter，

> **filter size：可以将m和H_k看做filter的宽高，D看做filter的层数（深度）D;**
>
> **filter的元素：每一层中的每个元素为 $W^{k,h}_{ij}$，且然后将其复制成D层的深度即可。**

如图4b所示，我们将filter沿着Z^{k+1}的embedding维度方向做平滑操作，即可得到隐向量 $X^{k+1}_{i,*}$，类比到CV领域中，我们将其称为一个feature map，因此X^k就是H^k个不同feature map的集合，

X^{k+1}就是H^{k+1}个不同feature map的集合。CIN中的名词 compressed 是指第k个隐层将$H^{k-1} * m$个向量的潜在空间压缩至H_k个向量（对于$H^{k-1} * m$个向量这样的结构，在潜在空间中一共有H_k个，每一个这样的结构只能压缩成为输出中的一个向量）。

> 当CIN的深度为1，隐层的feature map也只有1个，那么其就完全等价于FM模型了（可能会多了特征的平方项，少了特征的非交叉项，这些先不管了，你懂！）

如图4c所示，表达CIN的整体结构，用T表示CIN的深度，每一个隐藏层$X^k, k \in [1, T]$和输出单元都具有关联，在第k个隐层，在其中每个feature map上，我们做sum pooling操作，即：

$$p_i^k = \sum_{j=1}^{D} X_{i,j}^k, \qquad i \in [1, T] \qquad (7)$$

因此，对于第k个隐层，我们可以得到poolng之后的长度为H_k的向量：

$$\boldsymbol{p^k} = [p_1^k, p_2^k, \cdots, p_{H_k}^k]$$

对于所有隐层的pooling之后的向量，我们将它们concat之后作为输出单元，

$$\boldsymbol{p^+} = [\boldsymbol{p^1}, \boldsymbol{p^2}, \cdots, \boldsymbol{p^T}] \in \mathbb{R}^{\sum_{i=1}^{T} H_i}$$

哈哈，到这里，如果我们直接使用CIN用于二分类，那么输出单元就是一个在p^+上的sigmoid节点：

$$y = \frac{1}{1 + exp^{p^{+T} w^o}}$$

其中，w^o表示回归参数。

4.2 CIN分析

来看下CIN模块的复杂度，以及这样的结构是否有效果。

4.2.1 空间复杂度

CIN的空间复杂度和普通DNN的空间复杂度做对比。

分为一共T层的CIN的参数和最后一层回归层的参数，参数量为：

$$\sum_{k=1}^{T} H_k \times + \sum_{k=1}^{T} H_{k-1} \times m$$
$$= \sum_{k=1}^{T} H_k \times (1 + H_{k-1} \times m)$$

因此CIN的空间复杂度与每个特征的embedding向量的维度D的大小无关。

对比之下，一个普通的DNN模型的参数量为：

$$m \times D \times H_1 + H_T + \sum_{k=2}^{T} H_k \times H_{k-1}$$

在CIN模型中，由于m和 H_k 一般不会太大，**因此总体上来看CIN的参数量是是可接受的**，paper还针对了m和 H_k较大时的一个处理方案，就是进行矩阵分解，具体见paper中公式（9），这里就不再讲解了。

$$\mathbf{W}^{k,h} = \mathbf{U}^{k,h}(\mathbf{V}^{k,h})^T \qquad (9)$$

其中，

$$\boldsymbol{W^{k,h}} \in \mathbb{R}^{H_{k-1} \times m}, \boldsymbol{U^{k,h}} \in \mathbb{R}^{H_{k-1} \times L}, \boldsymbol{V^{k,h}} \in \mathbb{R}^{m \times L}, L \ll H, L \ll m$$

4.2.2 时间复杂度

CIN的时间复杂度和普通DNN的时间复杂度做对比。

CIN中，每个中间张量Z^{k+1}(feature map)的计算时间为O(mHD)，计算H个feature map，有T层CIN，时间 复杂度为$O(mTH^2D)$，而普通DNN的计算时间复杂度为$O(mHD + H^2T)$，因此相比之下，**CIN的主要缺点在于时间复杂度的劣势**。

4.2.3 多项式逼近

通过对问题进行简化，即假设CIN中不同层的feature map的数量全部一致，均为fields的数量m，并且用 [m] 表示小于等于m的正整数。CIN中的第一层的第h个feature map表示为$x_h^1 \in \mathbb{R}^D$，即：

$$\boldsymbol{x_h^1} = \sum_{i \in [m], j \in [m]} \boldsymbol{W}_{i,j}^{1,h}(x_i^0 \circ x_j^0) \qquad (10)$$

因此，在第一层中是通过O(m^2)个参数来建模成对的特征交互关系，相似的，第二层的第h个feature map表示为：

$$\begin{aligned} \boldsymbol{x_h^2} &= \sum_{i \in [m], j \in [m]} \boldsymbol{W}_{i,j}^{2,h}(x_i^1 \circ x_j^0) \\ &= \sum_{i \in [m], j \in [m]} \sum_{l \in [m], k \in [m]} \boldsymbol{W}_{i,j}^{2,h} \boldsymbol{W}_{l,k}^{1,h}(x_j^0 \circ x_k^0 \circ x_l^0) \qquad (11) \end{aligned}$$

上面公式（11）中有关下标为k和l的相关计算都已经在前面一层计算完成了，这里只是为了方便观察才又写出来的，知道就好。这里我们呢可以看到第二层中feature map的只用了$O(m^2)$个参数就建模出了3-way的特征交互关系。

我们知道一个经典的k阶多项式一般是需要$O(m^k)$个参数的，而我们展示了CIN在一系列feature map中只需要$O(km^2)$个参数就可以近似此类多项式。而且paper使用了归纳假设的方法证明了一下，具体见paper的公式（13）和公式（14）。

4.3 Combination with Implicit Networks

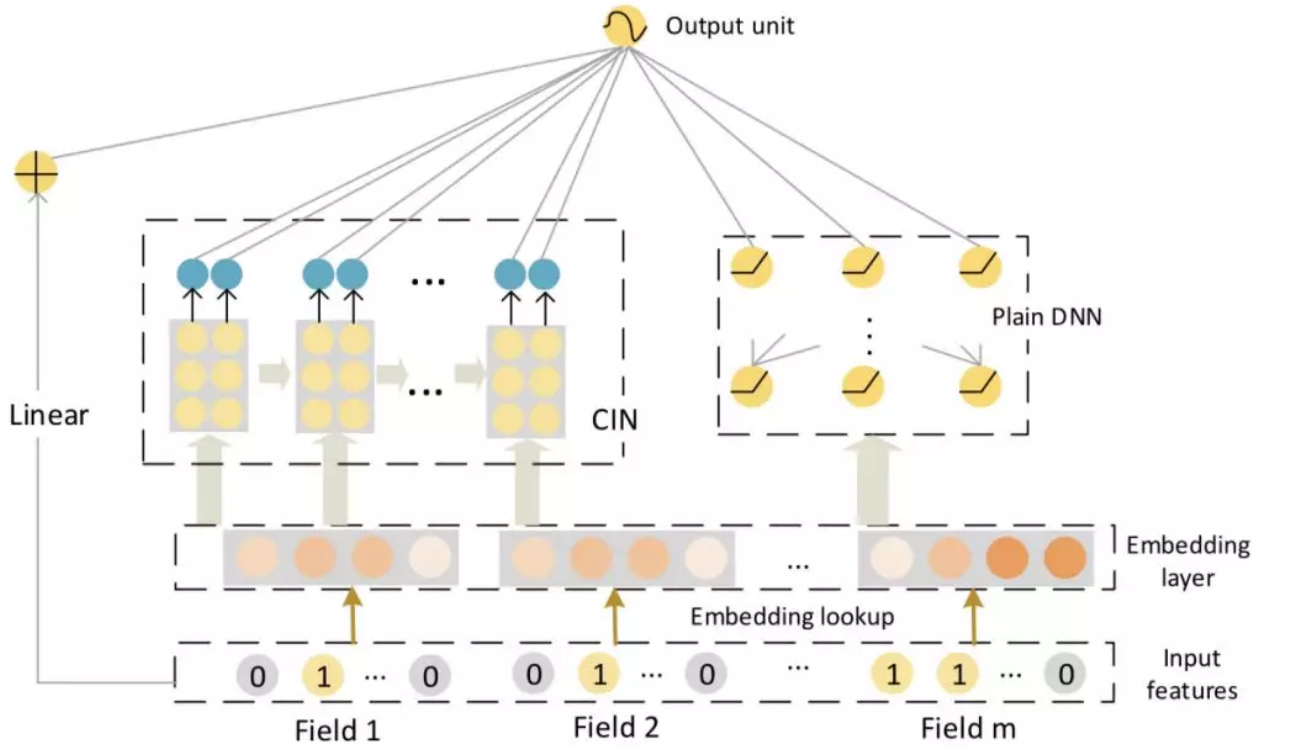


Figure 5: The architecture of xDeepFM.

xDeepFM将线性模块、CIN模块、DNN模型三者组合起来互为补充，分别提供的姐特征、显式高阶特征、隐式高阶特征。

预测结果的计算公式、loss函数、优化的目标函数分为如公式（15）、公式（16）、公式（17）所示，这里比较简单，不再解释。

$$\hat{y} = \sigma(\mathbf{w}_{linear}^T \mathbf{a} + \mathbf{w}_{dnn}^T \mathbf{x}_{dnn}^k + \mathbf{w}_{cin}^T \mathbf{p}^+ + b) \quad (15)$$

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N} y_i log\hat{y}_i + (1-y_i)log(1-\hat{y}_i) \quad (16)$$

$$\mathcal{J} = \mathcal{L} + \lambda_* ||\Theta|| \quad (17)$$

对于paper中提到的xDeepFM和FM、DeepFM的关系：

1. 对于xDeepFM，将CIN模块的层数设置为1，feature map数量也为1时，其实就是DeepFM的结构，因此DeepFM是xDeepFM的特殊形式，而xDeepFM是DeepFM的一般形式；
2. 在1中的基础上，当我们再将xDeepFM中的DNN去除，并对feature map使用一个常数形式的 `sum filter` ，那么x就DeepFM退化成了FM形式了。

5 实验环节

实验环节也是在解决三个问题：

1. CIN中的特征交互关系如何学习；
2. 对于推荐系统来说，是否有必要同时学习隐式和显式的高阶特征交互关系；
3. xDeepFM的参数设置对模型效果有怎样的影响？

5.1 实验数据集配置

使用了三个数据集：Criteo Dataset、Dianping Dataset、Bing News Dataset，这里不细说，具体细节见paper。

5.2 单独神经模块之间的对比

在三个数据集上，这里对比了FM、DNN、CrossNet、CIN四个单独模型的效果，实验证明CIN均取得了最好的效果，并且在DNN、CrossNet、CIN均优于FM的结果，也证明了在sparse feature上的高阶交互特征是有正向收益的。

Table 2: Performance of individual models on the Criteo, Dianping, and Bing News datasets. Column Depthindicates the best network depth for each model.

Model name	AUC	Logloss	Depth
Criteo			
FM	0.7900	0.4592	-
DNN	0.7993	0.4491	2
CrossNet	0.7961	0.4508	3
CIN	**0.8012**	0.4493	3
Dianping			
FM	0.8165	0.3558	-
DNN	0.8318	0.3382	3
CrossNet	0.8283	0.3404	2
CIN	**0.8576**	**0.3225**	2
Bing News			
FM	0.8223	0.2779	-
DNN	0.8366	0.273	2
CrossNet	0.8304	0.2765	6
CIN	**0.8377**	**0.2662**	5

5.3 组合模型之间的效果

从LR模型一路对比到xDeepFM 模型，验证了同时组合了显式高阶交互特征和隐式高阶交互特征的xDeepFM 模型可以去的最好的实验效果。

并且还看到了一个有意思的现象：实验中**所有的神经网络模型，在最好的参数设置中，都不需要太深的网络层数，一般只需要2～3层的网络深度即可**。

Table 3: Overall performance of different models on Criteo, Dianping and Bing News datasets. The column Depth presents the best setting for network depth with a format of (cross layers, DNN layers).

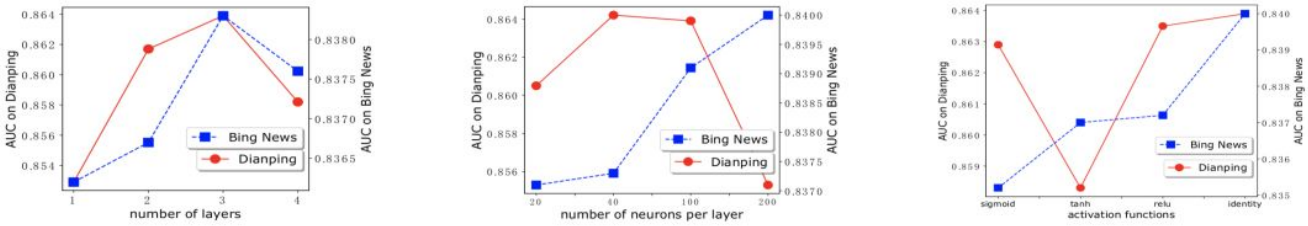
	Criteo			Dianping			Bing News		
Model name	AUC	Logloss	Depth	AUC	Logloss	Depth	AUC	Logloss	Depth
LR	0.7577	0.4854	-,-	0.8018	0.3608	-,-	0.7988	0.2950	-,-
FM	0.7900	0.4592	-,-	0.8165	0.3558	-,-	0.8223	0.2779	-,-
DNN	0.7993	0.4491	-,2	0.8318	0.3382	-,3	0.8366	0.2730	-,2
DCN	0.8026	0.4467	2,2	0.8391	0.3379	4,3	0.8379	0.2677	2,2
Wide&Deep	0.8000	0.4490	-,3	0.8361	0.3364	-,2	0.8377	0.2668	-,2
PNN	0.8038	0.4927	-,2	0.8445	0.3424	-,3	0.8321	0.2775	-,3
DeepFM	0.8025	0.4468	-,2	0.8481	0.3333	-,2	0.8[illegible]	0.2671	-,3
xDeepFM	**0.8052**	**0.4418**	3,2	**0.8639**	**0.3156**	3,3	**0.8406**	**0.2649**	3,2

5.4 CIN中超参数的设置

主要是隐层的深度、每层神经元的数量、激活函数。

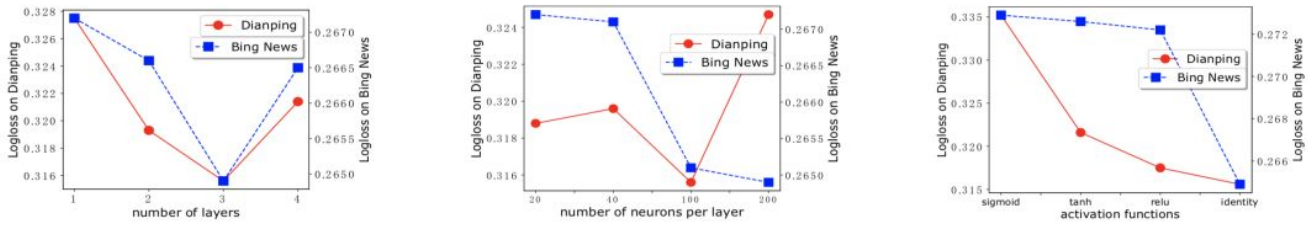
其中隐层的深度、每层神经元的数量，没有什么特别，就不再说了。值得一提的是，在激活函数的选取上，paper中在公式（6）中使用的激活函数是恒等式（哈哈，其实就是没有激活函数），作者这里也让实验了其他的例如sigmoid、tanh、relu等常用的非线性激活函数，但是结果显示在公式（6）上的恒等式变换的效果反而是最好的，即**可以不用其他的非线性激活函数效果就很好，用了的话效果反而会下降**。

对于这种现象，有什么合理的解释吗，还是就只能理解为这事deep learning中玄学问题（就是不知道怎么解释，哈哈），哪位大佬有这方面的理解，还请不吝赐教？



(a) Number of layers. (b) Number of neurons per layer. (c) Activation functions

Figure 6: Impact of network hyper-parameters on AUC performance.



(a) Number of layers. (b) Number of neurons per layer. (c) Activation functions

Figure 7: Impact of network hyper-parameters on Logloss performance.

6 总结

来总结下吧。

模型主要优势：

1. CIN可以学习高效的学习有界的高阶特征；
2. xDeepFM模型可以同时显示和隐式的学习高阶交互特征；
3. 以vector-wise方式而不是bit-wise方式学习特征交互关系。

模型可改进方面：

1. 目前对于multivalent fields的特征是使用sum pooling的方式处理的，未来可以借鉴DIN的思想进行改进；
2. CIN的时间复杂度比较高，paper中打算未来在GPU集群上使用分布式的方式来训练模型。

7 参考代码

1. https://github.com/Leavingseason/xDeepFM