DeepFM

1. CTR 预估

CTR 预估数据特点：

1. 输入中包含类别型和连续型数据。类别型数据需要 one-hot,连续型数据可以先离散化再 one-hot，也可以直接保留原值
2. 维度非常高
3. 数据非常稀疏
4. 特征按照 Field 分组

CTR 预估重点在于学习组合特征。注意，组合特征包括**二阶、三阶**甚至更高阶的，阶数越高越复杂，越不容易学习。**Google 的论文研究得出结论：***高阶和低阶的组合特征都非常重要，同时学习到这两种组合特征的性能要比只考虑其中一种的性能要好。*

那么关键问题转化成：**如何高效的提取这些组合特征。**一种办法就是引入领域知识人工进行特征工程。这样做的弊端是高阶组合特征非常难提取，会耗费极大的人力。而且，有些组合特征是隐藏在数据中的，即使是专家也不一定能提取出来，比如著名的“尿布与啤酒”问题。

在 DeepFM 提出之前，已有 LR，FM，FFM，FNN，PNN（以及三种变体：IPNN,OPNN,PNN*）,Wide&Deep 模型，这些模型在 CTR 或者是推荐系统中被广泛使用。

2. 模型演进历史

2.1 线性模型

最开始 CTR 或者是推荐系统领域，一些线性模型取得了不错的效果。比如：**LR，FTRL**。线性模型**有个致命的缺点：无法提取高阶的组合特征(线性 y=w0+w1+w2 等)**。所以常用的做法是人为的加入 **pairwise feature interactions**。即使是这样：对于那些出现很少或者没有出现的组合特征以及高阶组合特征依旧无法提取。

LR 最大的缺点就是**无法组合特征，依赖于人工的特征组合，这也直接使得它表达能力受限，基本上只能处理线性可分或近似线性可分的问题**。

2.2 FM 模型

线性模型差强人意，直接导致了 FM 模型应运而生（在 Kaggle 上打比赛提出来的，取得了第一名的成绩）。**FM 通过隐向量 latent vector 做内积来表示组合特征**，从理论上解决了低阶和高阶组合特征提取的问题。但是实际应用中受限于计算复杂度，一般也就**只考虑到 2 阶交叉特征**。

后面又进行了改进，提出了 FFM，增加了 Field 的概念。

2.3 遇上深度学习

随着 DNN 在图像、语音、NLP 等领域取得突破，人们见见意识到 DNN 在特征表示上的天然优势。相继提出了使用 CNN 或 RNN 来做 CTR 预估的模型。但是，CNN 模型的缺点是：偏向于学习相邻特征的组合特征。 RNN 模型的缺点是：比较适用于有序列(时序)关系的数据。

FNN 的提出，应该算是一次非常不错的尝试：先使用预先训练好的 FM，得到隐向量，然后作为 DNN 的输入来训练模型。缺点在于：受限于 FM 预训练的效果。

随后提出了 PNN，PNN 为了捕获高阶组合特征，在 embedding layer 和 first hidden layer 之间增加了一个 product layer。根据 product layer 使用内积、外积、混合分别衍生出 IPNN, OPNN, PNN*三种类型。

无论是 FNN 还是 PNN，他们都有一个绕不过去的缺点：对于低阶的组合特征，学习到的比较少。而前面我们说过，低阶特征对于 CTR 也是非常重要的。

Google 意识到了这个问题，为了同时学习低阶和高阶组合特征，提出了 Wide&Deep 模型。它混合了一个线性模型（Wide part）和 Deep 模型(Deep part)。这两部分模型需要不同的输入，而 Wide part 部分的输入，依旧依赖人工特征工程。

但是，这些模型普遍都存在两个问题：

1. 偏向于提取低阶或者高阶的组合特征。不能同时提取这两种类型的特征。
2. 需要专业的领域知识来做特征工程。

DeepFM 在 Wide&Deep 的基础上进行改进，成功解决了这两个问题，并做了一些改进，其优势/优点如下：

1. 不需要预训练 FM 得到隐向量
2. 不需要人工特征工程
3. 能同时学习低阶和高阶的组合特征
4. FM 模块和 Deep 模块共享**Feature Embedding**部分，可以更快的训练，以及更精确的训练学习

3. DeepFM

DeepFM 闪亮登场！

主要做法如下：

1. FM Component + Deep Component。FM 提取低阶组合特征，Deep 提取高阶组合特征。但是和 Wide&Deep 不同的是，DeepFM 是端到端的训练，不需要人工特征工程。

2. 共享 feature embedding。FM 和 Deep 共享输入和 feature embedding 不但使得训练更快，而且使得训练更加准确。相比之下，Wide&Deep 中，input vector 非常大，里面包含了大量的人工设计的 pairwise 组合特征，增加了他的计算复杂度。

DeepFM 架构图：

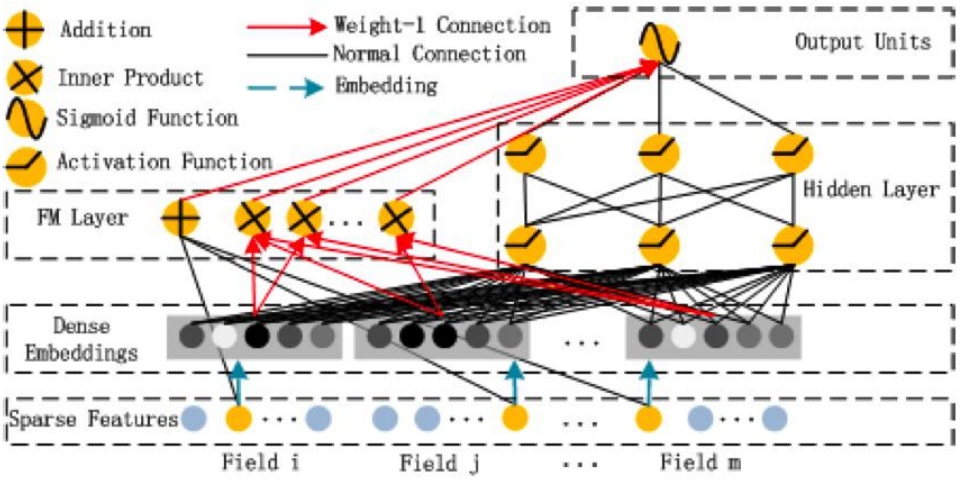


Figure 1: Wide & deep architecture of DeepFM. The wide and deep component share the same input raw feature vector, which enables DeepFM to learn low- and high-order feature interactions simultaneously from the input raw features.

3.1 FM Component

FM 部分的输出由两部分组成：一个 Addition Unit，多个内积单元。

$$y_{FM} = \langle w, x \rangle + \sum_{j_1=1}^{d} \sum_{j_2=j_1+1}^{d} \langle V_i, V_j \rangle x_{j_1} \cdot x_{j_2},$$

这里的 d 是输入 one-hot 之后的维度，我们一般称之为 feature_size。对应的是 one-hot 之前的特征维度，我们称之为 field_size。

FM 架构图：

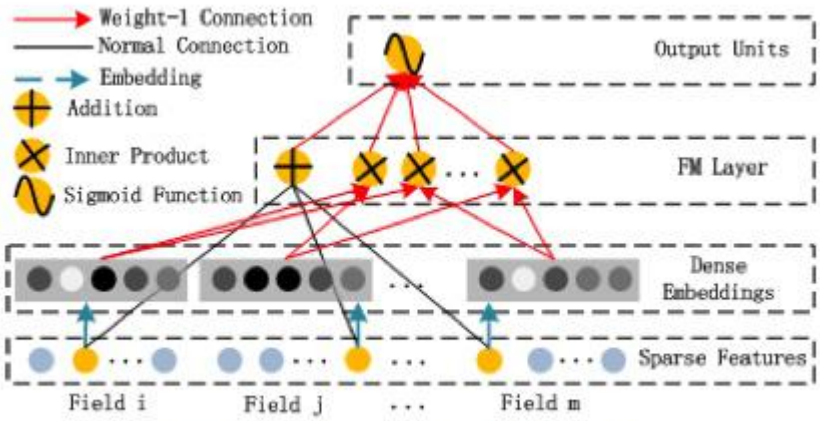


Figure 2: The architecture of FM.

Addition Unit

3.2 Deep Component

Deep Component 架构图：

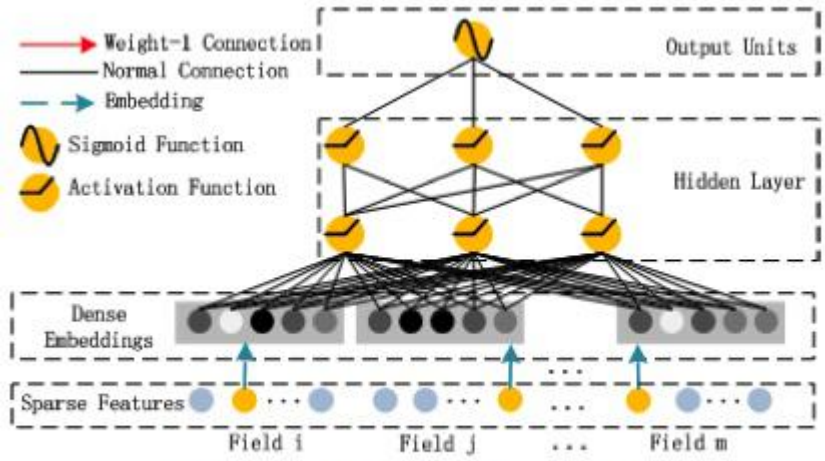


Figure 3: The architecture of DNN.

Deep Component 是用来学习高阶组合特征的。网络里面黑色的线是全连接层，参数需要神经网络去学习。

由于 CTR 或推荐系统的数据 one-hot 之后特别稀疏，如果直接放入到 DNN 中，参数非常多，我们没有这么多的数据去训练这样一个网络。**所以增加了一个 Embedding 层，用于降低纬度。**

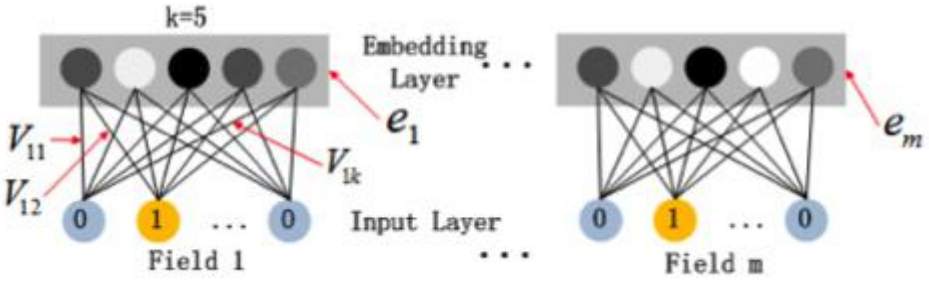
这里继续补充下 Embedding 层，两个特点：

1. 尽管输入的长度不同，但是映射后长度都是相同的.embedding_size 或 k

2. embedding 层的参数其实是**全连接的 Weights**，是通过神经网络自己学习到的。

Embedding 层的架构图：

注意：原来 one-hot 编码得到特征 100 维度，根据 embedding 降低到 5 维度



值得注意的是：FM 模块和 Deep 模块是共享 feature embedding 的（也就是 V）。

好处：

1. 模型可以从最原始的特征中，同时学习低阶和高阶组合特征
2. 不再需要人工特征工程。Wide&Deep 中低阶组合特征就是同过特征工程得到的。

3.3 对比其他模型

模型图：

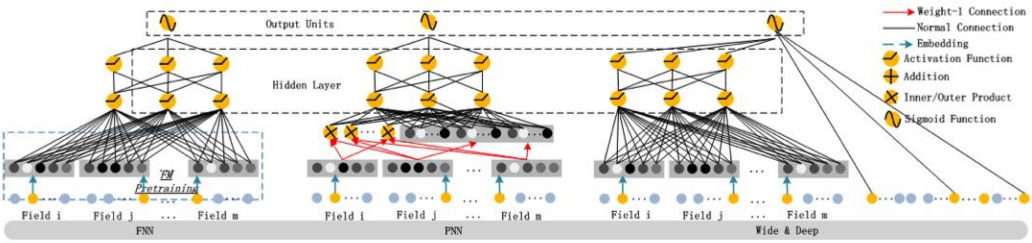


Figure 5: The architectures of existing deep models for CTR prediction: FNN, PNN, Wide & Deep Model

FNN

FNN is a FM-initialized feedforward neural network.

FNN 使用预训练的 FM 来初始化 DNN，然后只有 Deep 部分，不能学习低阶组合特征。

FNN 缺点：

Embedding 的参数受 FM 的影响，不一定准确
预训练阶段增加了计算复杂度，训练效率低
FNN 只能学习到高阶的组合特征。模型中没有对低阶特征建模。

PNN

PNN：为了捕获高阶特征。PNN 在第一个隐藏层和 embedding 层之间，增加了一个 product layer。

根据 product 的不同，衍生出三种 PNN：IPNN，OPNN，PNN* 分别对应内积、外积、两者混合。

作者为了加快计算，采用近似计算的方法来计算内积和外积。内积：忽略一些神经元。外积：把 m*k 维的 vector 转换成 k 维度的 vector。由于外积丢失了较多信息，所以一般没有内积稳定。

但是内积的计算复杂度依旧非常高，原因是：product layer 的输出是要和第一个隐藏层进行全连接的。

PNN 缺点：

内积外积计算复杂度高。采用近似计算的方法外积没有内积稳定。
product layer 的输出需要与第一个隐藏层全连接，导致计算复杂度居高不下
和 FNN 一样，只能学习到高阶的特征组合。没有对于 1 阶和 2 阶特征进行建模。
Wide&Deep
Wide & Deep 设计的初衷是想同时学习低阶和高阶组合特征，但是 wide 部分需要领域知识进行特征工程。

Wide 部分可以用 LR 来替换，这样的话就和 DeepFM 差不多了。
但是 DeepFM 共享 feature embedding 这个特性使得在反向传播的时候，模型学习 feature embedding，而后又会在前向传播的时候影响低阶和高阶特征的学习，这使得学习更加的准确。

Wide&Deep 缺点：

需要特征工程提取低阶组合特征

DeepFM

优点：

- 没有用 FM 去预训练隐向量 V，并用 V 去初始化神经网络。（相比之下 FNN 就需要预训练 FM 来初始化 DNN）
- FM 模块不是独立的，是跟整个模型一起训练学习得到的。（相比之下 Wide&Deep 中的 Wide 和 Deep 部分是没有共享的）
- 不需要特征工程。（相比之下 Wide&Deep 中的 Wide 部分需要特征工程）
- 训练效率高。（相比 PNN 没有那么多参数）

什么？太多了乱七八糟记不住？OK，那就记住最核心的：

没有预训练（no pre-training）
共享 Feature Embedding，**没有特征工程**（no feature engineering）
同时**学习低阶和高阶组合特征**（capture both low-high-order interaction features）

3.4 超参数建议

论文中还给出了一些参数的实验结果，直接给出结论，大家实现的时候可以参考下。

超参数　建议　　备注
激活函数　　1. IPNN 使用 tanh 2. 其余使用 ReLU
学习方法　　Adam
Dropout　0.6~0.9
隐藏层数量　3~5　根据实际数据大小调整
神经元数量　200~600　根据实际数据大小调整
网络形状　　constant　一共有四种：固定、增长、下降、菱形。
PS: constant 效果最好，就是隐藏层每一层的神经元的数量相同。

最后，模型对比图镇：

Reference
DeepFM: A Factorization-Machine based Neural Network for CTR Prediction