曾今的CTR竞赛王者NFM

原创 二级炼丹师一元 炼丹笔记 9月6日

收录于话题
#搜索推荐前沿算法

14个

Neural Factorization Machines for Sparse Predictive Analytics

这是一篇SIGIR17年的论文,非常经典,本文提出的Bi-Interaction Pooling操作在后续非常多的工作中带来了帮助。在18年前后的竞赛中,前排选手将在Bi-Interaction中替换FFM并取得了top10的成绩，当时在推荐领域也算王者一般的存在。话不多说，我们直接进入主题。

模型

FM

假设我们的每个样本$x \in R^n$, 对于每个特征x_i, 我们对应有一个隐变量 $v_i \in R^k$表示特征i的embedding向量,那么我们的FM为:

$$\bar{y}_{fm}(x) = w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} v_i^T v_j \cdot x_i x_j$$

而FM可以被理解为是多元的线性模型，表示能力还是相对很低的。

NFM

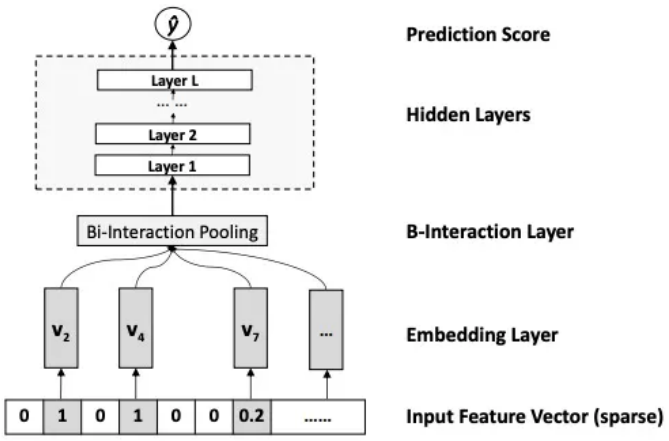

Figure 2: Neural Factorization Machines model (the first-order linear regression part is not shown for clarity).

那么NFM如何增强模型的表示能力同时更好地学习二阶以及高阶的特征表示呢?

$\bar{y}_{nfm}(x) = w_0 + \sum_{i=1}^{n} w_i x_i + f(x)$

其中:

$f(x) = h^T \sigma_L(W_L(\dots \sigma_1 W_1(f_{BI}(V_x) + b_1)\dots) + b_L)$

这么看来, NFM模型最神奇的地方就在于$f_{BI}(V_x)$这块,在大量的实验中,我们发现直接concat embedding向量之后再加入MLP网络进行模型的训练在捕捉模型的二层交叉方法效果是不佳的。如果说$f_{BI}(V_x)$后接的网络是为了捕捉高阶交叉信息,那么$f_{BI}(V_x)$需要弥补这些交叉信息。

$f_{BI}(V_x) = \sum_{i=1}^{n} \sum_{j=i+1}^{n} x_i v_i \odot x_j v_j$

上面的操作就是我们所说的Bi-Interaction, ⊙表示元素方面的相乘(element-wise), 即$(v_i \odot v_j)_k = v_{ik} v_{jk}$,上面的操作又可以被简化为:

$f_{BI}(V_x) = \frac{1}{2}\left[\left(\sum_{i=1}^{n} x_i v_i\right)^2 - \sum_{i=1}^{n} (x_i v_i)^2\right]$

NFM VS FM,WDL,DeepCross

NFM VS FM

NFM可以看做是FM的扩展,FM是NFM的特例,

$y_{NMF-0} = w_0 + \sum_{i=1}^{n} w_i x_i + h^T \sum_{i=1}^{n} \sum_{j=i+1}^{n} x_i v_i \odot x_j v_j$

$\rightarrow y_{NMF-0} = w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \sum_{f=1}^{k} h_f v_{if} v_{jf} \odot x_i x_j$

当h为(1,1,1...,1)的时候,就是FM模型.

NFM VS WDL(Wide and Deep) VS DeepCross

如果我们把Bi-Interaction pooling去掉,把它们concat起来再使用tower结构的MLP拼起来,就得到了我们的WDL(or DCS)。

- WDL过于依赖于模型通过自己的能力去学习特征之间的交叉,而大量的实验证明这是非常困难的，效果往往不尽如人意。

实验

实验主要回答下面3大问题:

1. Q1:NFM中Bi-Interaction pooling可以捕捉二阶特征交叉？Dropout和BN是否对于Bi-Interaction pooling有帮助?
2. Q2:NMF中后面加的隐藏层是否可以提高模型捕捉高阶特征并提升FM的表示能力;

3. Q3:NFM和高阶FM,WDL以及DCS相比是否可以取得更好的效果.

Q1

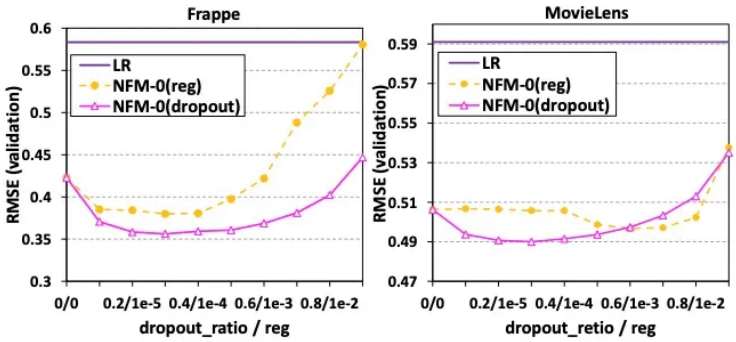

Figure 3: Validation error of NFM-0 *w.r.t.* dropout on the Bi-Interaction layer and L_2 regularization on embeddings.

- 从上面的实验中来看,Bi-Interaction pooling可以很好的帮助我们捕捉二阶特征交叉信息，获得更好的效果;

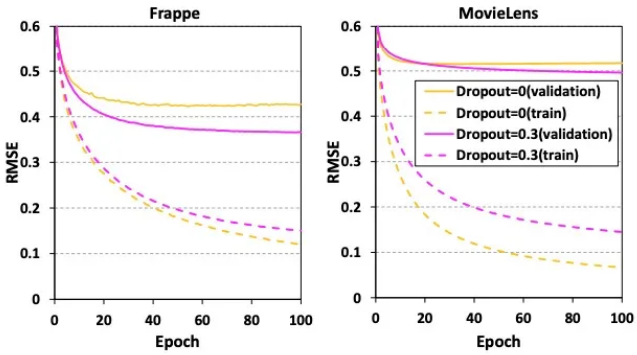

Figure 4: Training and validation error of each epoch of NFM-0 with and without dropout on Bi-Interaction layer.

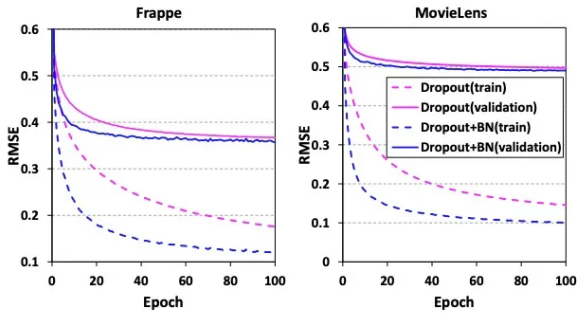

Figure 5: Training and validation error of each epoch of NFM-0 with and without BN on the Bi-Interaction layer.

- 从上面的实验中来看,Dropout可以帮助我们获得更好的泛化效果;
- 从上面的实验中来看,BatchNorm不仅可以加速模型训练,而且可以获得更好的效果;

Q2

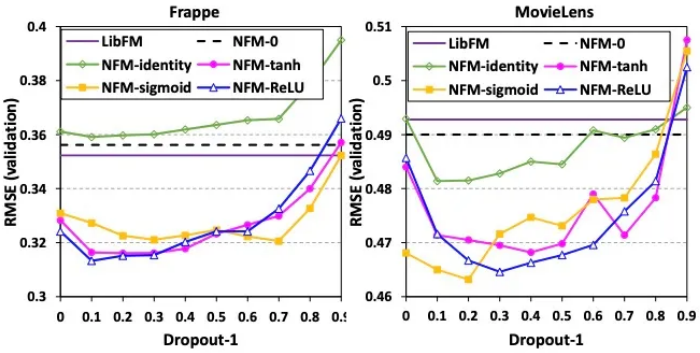

Figure 6: Validation error of LibFM, NFM-0 and NFM with different activation functions on the first hidden layer.

- 模型之后加入非线性激活函数可以大大提升模型效果,说明更加高阶的非线性关系可以更好地帮助我们模型的预测;

Table 2: NFM *w.r.t.* different number of hidden layers.

Methods	Frappe	MovieLens
NFM-0	0.3562	0.4901
NFM-1	**0.3133**	**0.4646**
NFM-2	0.3193	0.4681
NFM-3	0.3219	0.4752
NFM-4	0.3202	0.4703

- 加深模型的深度,可以大大提升模型的效果,加深一层是最为明显的,继续加效果相差不大。

Q3

Table 3: Test error and number of trainable parameters for different methods on latent factors 128 and 256. M denotes "million"; * and ** denote the statistical significance for $p < 0.05$ and $p < 0.01$, respectively, compared to the best baseline.

	Frappe				MovieLens			
	Factors=128		Factors=256		Factors=128		Factors=256	
Method	Param#	RMSE	Param#	RMSE	Param#	RMSE	Param#	RMSE
LibFM [28]	0.69M	0.3437	1.38M	0.3385	11.67M	0.4793	23.24M	0.4735
HOFM	1.38M	0.3405	2.76M	0.3331	23.24M	0.4752	46.40M	0.4636
Wide&Deep [9]	2.66M	0.3621	4.66M	0.3661	12.72M	0.5323	24.69M	0.5313
Wide&Deep (pre-train)	2.66M	0.3311	4.66M	0.3246	12.72M	0.4595	24.69M	0.4512
DeepCross [31]	4.47M	0.4025	8.93M	0.4071	12.71M	0.5885	25.42M	0.5907
DeepCross (pre-train)	4.47M	0.3388	8.93M	0.3548	12.71M	0.5084	25.42M	0.5130
NFM	**0.71M**	**0.3127****	**1.45M**	**0.3095****	**11.68M**	**0.4557***	**23.31M**	**0.4443***

- NFM相较于其他的模型都取得了巨大的提升，而且模型参数也更少。

小结

本文的Bi-Interaction Pooling非常有启发,在实验中也取得了非常好的效果。早期本人将Bi-Interaction Pooling部分加入模型当中也取得了不错的提升，比只用embedding concat的效果要好了2%左右，如果你的模型还是早期的，可以加入试一试哦。如果有提升了记得关注一波我们的公众号。

参考文献

1. Neural Factorization Machines：https://github.com/hexiangnan/neural_factorization_machine