Outer Product-based Neural Collaborative Filtering

Accepted by IJCAI-2018，论文链接：http://www.comp.nus.edu.sg/~xiangnan/papers/ijcai18-ConvNCF.pdf

摘要

本文指出了目前较为流行的各种 NCF 及其变体模型大多采用特征向量内积、拼接操作的局限性。为此，提出了一种基于外积的新的特征交互模型，通过特征交互操作得到一个特征矩阵，在此基础上应用 CNN 学习到特征中每一维的高阶相互关系。从实验中验证了这一观点，代码开源在 https://github.com/duxy-me/ConvNCF

背景简介

推荐系统问题是在 IR （信息检索）领域中很重要的研究方向，而协同过滤方法又是最为常用且有效地推荐算法和模型。一个协同过滤模型主要有两部分：使用特征向量表示用户和物品；特征向量的交互计算，即用户对物品的感兴趣程度。（注：表示学习得到用户、物品的特征向量 p 和 q，而模型设计中，学习一个打分函数 $f:(\mathbb{R}^n,\mathbb{R}^n)\rightarrow\mathbb{R}$。）因此，在引入神经网络模型后，大部分工作都在如何学习更好的表示和交互模型的构建上。

尽管大大小小的改进，都取得了一定的成果，但是 CF 方法存在固有的局限：在模型学习中，采用一种固定的操作——内积，假设 $f(p,q)=p^Tq\in\mathbb{R}$。在这样的假设下，特征向量 p 和 q 的每一维（每一个特征的角度、语义）都是平等的，而且这样的操作每一维都是独立的计算乘积最后求和。作者指出，特征向量的每一个维度都有特定的语义，表征了对象的某个方面的信息，在使用时应该有所侧重的。此外，当前对于特征映射函数 f 的改进，如从数据中用 MLP 等其他抽象形式来学习，也显示出能取得更出色的效果。

本文的作者团队与 2017 年提出的 Neural CF 模型，在用户、物品的特征输入层上，使用多个非线性层（深度网络）学习函数 f，已经成为"标配"，处理方式主要有拼接和按位相乘。尽管 MLP 理论上能够拟合任意复杂的连续函数 f，但这样的做法仍然没有对特征向量的每个不同的维度进行限制。

本文提出了一种新的 NCF 模型，意在模型中融合一种特征维度之间的关系。具体而言，就是通过特征向量的外积，得到交互矩阵（Interaction Map）$E\in\mathbb{R}^{K\times K}$，$K$ 是特征向量维数。如此构建的交互矩阵，体现出了每个维度下特征之间的关系，而其中也包含了传统 CF 的内积信息（E 中的主对角线求和即为向量内积），最终能刻画特征维度之间的高阶关系。此外，在特征矩阵上引入 CNN 处理方式，也比全连接 MLP 更容易泛化、也更容易建立更深的网络。

本文的贡献主要有：

- 提出了一种基于外积的 NCF 模型，ONCF，刻画特征向量的每个维度的相互关系；
- 在特征交互的矩阵上采用 CNN，从局部和全局，对每个维度进行高阶的交互；
- 通过扩展实验，验证了 ONCF 模型的理论正确和有效性；
- 第一个使用 CNN 对特征映射函数 f 进行建模。

模型

模型设计

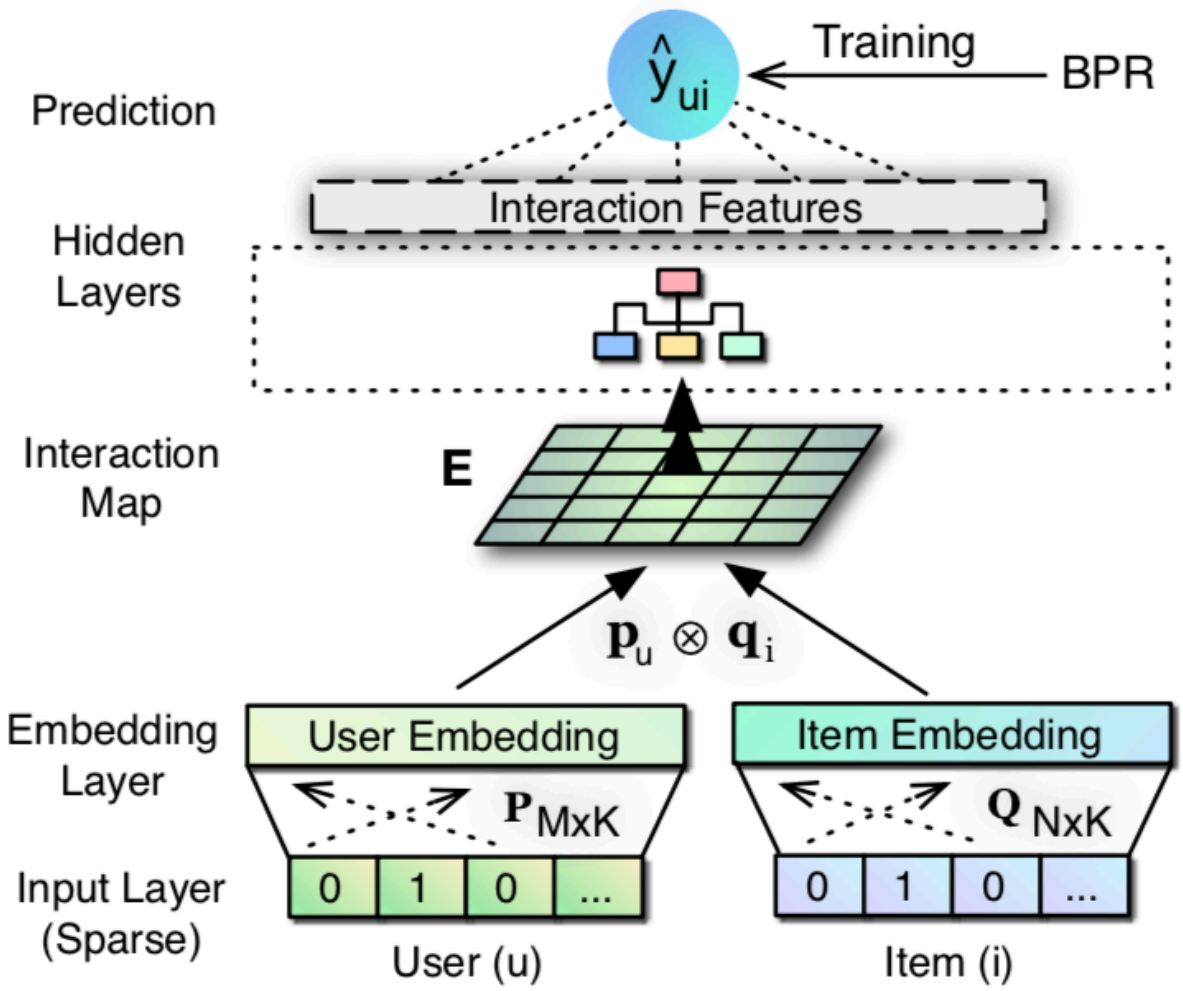

Figure 1: Outer Product-based NCF framework

- ONCF 模型的输入是一个用户、物品对 u 和 i，输入层是一个 one-hot 编码的向量，通过一个 embedding 映射 P 和 Q 得到一个表示 p_u 和 q_i
- 计算外积，得到 Interaction Map $E = p_u \otimes q_i = p_u q_i^T$ 矩阵。与一般方法相比，具有三个优势：
  - 对于矩阵分解的方法是一种泛化，矩阵分解只使用了 E 的主对角元素；
  - 比矩阵分解考虑到了更多的每一个特征维度之间的关系；
  - 比特征向量的拼接更有可解释性，是从一个综合的角度来看建模的。

  与此同时，相关的实践经验表示，这样的特征之间的交互操作，更有利于深度模型的学习。（注：作者团队提出过 NFM 模型验证）另一方面，E 是一个二维矩阵，与（单通道）图片是类似的，CNN 操作尤其能够抓住这个矩阵中的局部、全局信息特征。
- 隐藏层，输入是交互矩阵 E，输出则是隐藏状态 $g = f_\Theta(E)$，其中 f_Θ 是一个从矩阵到向量的映射，Θ 是模型参数。在这个模型中，是一个 CNN 结构
- 预测层，计算预测得分 $\hat{y}_{ui} = w^T g$，至此，整个模型需要学习的参数有 P, Q, Θ, w
- 模型的损失函数是一个 BPR 损失，能够刻画观测到的正例比负例有更高的得分排名：
  $L(\Delta) = \sum_{(u,i,j)\in\mathcal{D}} -\ln \sigma(\hat{y}_{ui} - \hat{y}_{uj}) + \lambda_\Delta \|\Delta\|^2$，其中 $\mathcal{D} := \{(u,i,j)|i \in \mathcal{Y}_u^+ \wedge j \notin \mathcal{Y}u^+\}$，
  $\mathcal{Y}^+$ 是用户 u 观测到的正例。
- 值得一提的是，预测层参数 w 的等比例放大，会增大 $(\hat{y}_{ui} - \hat{y}_{uj})$ 的值，体现在训练过程中，会使得 w 的分量值较大，因此需要引入 L_2 或者是 maxnorm 的限制。

卷积 NCF

动机：MLP 的缺陷。

可以看到，隐藏层，从特征交互得到的隐藏状态是整个模型中最核心的部分。对于这里的 E，可以使用一个 MLP 对向量化（拉直操作）的 E 处理和运算。可是这样做带来的问题是：如果 E 的规模是 $K \times K$，取 $K = 64$，那么 $e = \text{vec}(E)$ 是一个 $64 \times 64 = 4096$ 的特征向量，那么标准的第一层 MLP 的权重就已经至少是 4096×2048 的规模了，这样的参数规模是可怕的：需要极其庞大的计算资源和开销、需要更多的数据来拟合参数、调参问题。

卷积 NCF 模型

相比较 MLP 模型，CNN 的参数规模大大减小，也更容易往深度学习。

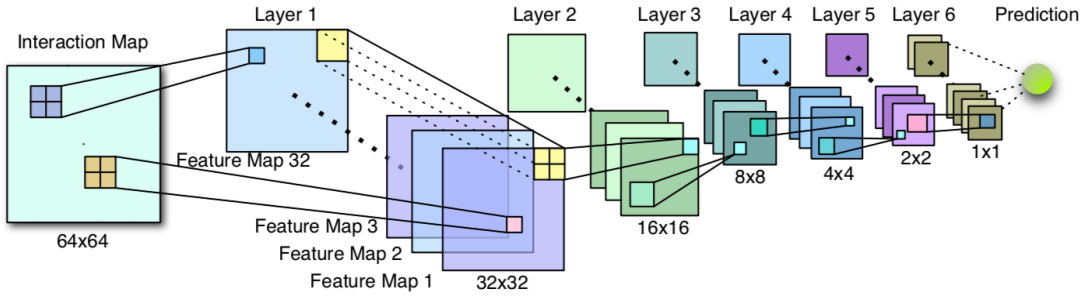

Figure 2: An example of the architecture of our ConvNCF model that has 6 convolution layers with embedding size 64.

- 输入的是一个 $K \times K$ 的矩阵 E，图中假设 $K = 64$
- 模型一共有 6 层，每一层都有 32 个 Feature Map，每个卷积核都是 2×2 的，那么每一层过后，Feature Map 的两个维度都缩小为上一层的一半，激活函数选择 RELU
- 最后输出第六层是一个 $1 \times 1 \times 32$ 的张量，可以视作一个向量 g，即为隐藏层的输出
- 从参数规模上，比第一层的 MLP 就要小了几百倍，因此在数据不太多的情况下，具有更好的稳定性和泛化能力。

卷积 NCF 的理论说明

作者给出了一些直观上的说明，在从前一层去往后一层的过程中，后一层的每一个元素都是由前一层的 4 个元素计算得来的，可以认为是一个 4 阶关系的刻画。直到最后的输出层，降到 1×1 后，即包含了特征每一个维度之间的交互信息。

实验

数据集选择的是经过处理的 Yelp 和 Gowalla，评价指标是 HR 和 NDCG，基线对比实验有 ItemPop，MF-BPR，MLP，JRL 和 NMF，结果如下：

Table 1: Top-k recommendation performance where $k \in \{5, 10, 20\}$. RI indicates the average improvement of ConvNCF over the baseline. * indicates that the improvements over all other methods are statistically significant for $p < 0.05$.

	Gowalla						Yelp						
	HR@k			NDCG@k			HR@k			NDCG@k			RI
	$k=5$	$k=10$	$k=20$	$k=5$	$k=10$	$k=20$	$k=5$	$k=10$	$k=20$	$k=5$	$k=10$	$k=20$	
ItemPop	0.2003	0.2785	0.3739	0.1099	0.1350	0.1591	0.0710	0.1147	0.1732	0.0365	0.0505	0.0652	+227.6%
MF-BPR	0.6284	0.7480	0.8422	0.4825	0.5214	0.5454	0.1752	0.2817	0.4203	0.1104	0.1447	0.1796	+9.5%
MLP	0.6359	0.7590	0.8535	0.4802	0.5202	0.5443	0.1766	0.2831	0.4203	0.1103	0.1446	0.1792	+9.2%
JRL	0.6685	0.7747	0.8561	0.5270	0.5615	0.5821	0.1858	0.2922	0.4343	0.1177	0.1519	0.1877	+3.9%
NeuMF	0.6744	0.7793	0.8602	0.5319	0.5660	0.5865	0.1881	0.2958	0.4385	0.1189	0.1536	0.1895	+3.0%
ConvNCF	**0.6914***	**0.7936***	**0.8695***	**0.5494***	**0.5826***	**0.6019***	**0.1978***	**0.3086***	**0.4430***	**0.1243***	**0.1600***	**0.1939***	-

毫无悬念，ConvNCF 是最好的。另外，JRL 比 MLP 要好，也从侧面说明了多个特征角度之间的交互，是有作用的。

外积与 CNN 的作用

对于外积的作用，作者的对比试验 MLP 中采用的是向量的拼接、GMF、JRL 是向量点乘，在训练过程中，ConvNCF 始终要优于其他方法；对于 CNN 的作用，作者也使用了一个 MLP 对特征交互矩阵 E 进行抽象，尽管使用的 MLP 参数规模要巨大得多。而实验结果显示，MLP 即便是有更大的参数规模，性能还是比不上 ConvNCF。

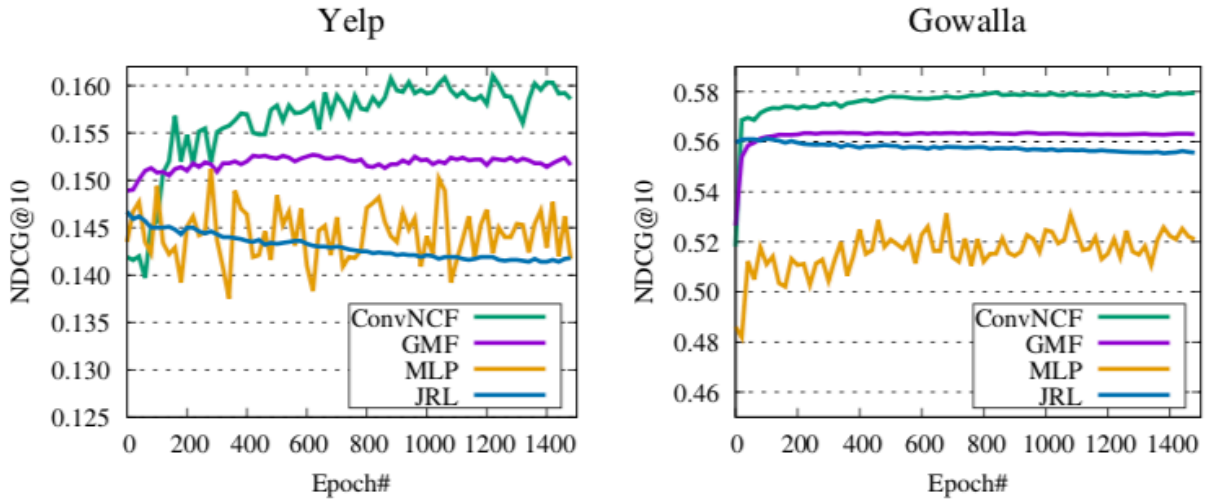

Figure 3: NDCG@10 of applying different operations above the embedding layer in each epoch (GMF and JRL use element-wise product, MLP uses concatenation, and ConvNCF uses outer product).

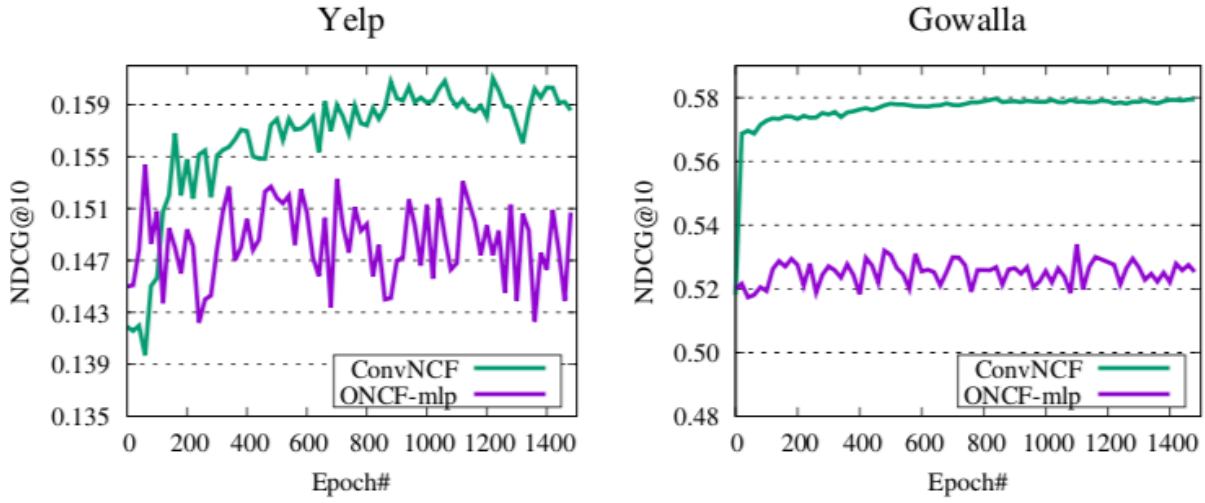

Figure 4: NDCG@10 of using different hidden layers for ONCF (ConvNCF uses a 6-layer CNN and ONCF-mlp uses a 3-layer MLP above the interaction map).

超参分析

对于 E 的 embedding size，发现不同的大小在训练过程中性能都得到了一些提高，选择一个不太大的超参规模，可以避免过拟合。另一方面，也显示出 ConvNCF 模型具有更好的学习泛化能力。

总结

作者从协同过滤算法的模型设计上进行了改进，引入向量的外积操作，从向量式的表示到矩阵，再选择 CNN 进行抽象，刻画了更为复杂的特征之间的关系，最终获得了性能的提升。

本文虽然想法简单，但是的确是很有道理的，结构也比较清楚，符合现在的 simple、elegant、reasonable and effective 的特点。引入 CNN 处理特征是很有启发式的做法，也可以用来学习用户的表示，to be continued...