摘要

具有非线性特征变换的广义线性模型被广泛用于稀疏特征的大量回归和分类问题。通过广泛的使用交叉特征转化，使得特征交互的记忆性是有效的，并且具有可解释性，但是不得不做大量的特征工作。而对于需要较少的特征工程，深度神经网络可以通过把稀疏特征转换成低维度的密集特征在进行深度学习，从而更好地泛化到不可见的特征组合。然而，当用户操作行为稀疏且项目排名高时，使用密集特征学习的深度神经网络会过度泛化和推荐不太相关的项目。在这篇论文中，提出了广度和深度学习联合训练的广度+深度学习模型，能满足推荐系统需要的记忆能力和泛化能力。我们在一个拥有超过 10 亿活跃用户和超过 100 万个应用的商业移动应用商店-Google Play ,应用并评估了这个模型。在线实验结果表明，与仅采用广度和深度的模型相比，广度+深度模型大大增加了应用的获取量。我们同时将这个发明在 TensorFlow 中的实现开源。

CCS Concepts -CCS 概念

计算方法-->机器学习；

神经网络；监督学习；

信息系统-->推荐系统

关键词

Wide & Deep Learning, Recommender Systems.

广度和深度学习，推荐系统。

1、引言

推荐系统可以看作是一个搜索排名系统，其中输入的查询时（包含有用户和上下文信息），输出的是项目的排名列表。给定一个查询，推荐系统任务是在数据库中找到相关项，然后根据某些目标（如点击或购买）对这些项进行排序。

类似于一般搜索排名问题，推荐系统中的一个挑战是同时实现记忆和泛化。记忆可以粗略地定义为学习项目或特征的频繁出现，并利用历史数据中可用的相关性。另一方面，泛化是基于相关性的传递性，探索过去从未或很少出现的新特征组合。基于记忆的推荐通常更具有热门话题性，并且与用户已经执行过操作的项目直接相关。与记忆相比，泛化倾向于提高推荐项目的多样性。在本文中，我们重点介绍 Google Play 商店的应用推荐问题，但该方法应适用于通用推荐系统。

对于工业环境下的大规模在线推荐和排名系统，logistic 回归等广义线性模型因其简单、可扩展和可解释性而得到广泛应用。这些模型通常使用热狗编码对二值化稀疏特征进行训练。例如，如果用户安装了 netflix，那么对应“user_installed_app=netflix”的训练数据加工后的值就处理为 1。通过对稀疏特征进行特征的点积变换，可以有效地实现记忆，例如（用户安装的 netflix 和 印象的 APP 为 pandora 点积），如果用户安装了 netflix，当出现值为 1 时，紧接会展示 pandora 出来。这解释了特征对的共现如何与目标标签相关。泛化的可以通过使用粒度较小的特征来增加，例如（用户安装的类别=视频，印象类别=音乐），但通常需要人工特征设计。特征点积变换的一个局限性在于它们不能推广到训练数据中未出现的查询项特征对。

基于密集特征的模型，如因子分解机或深度神经网络，可以通过学习每个查询项目特征的低维密集特征向量，将其推广到以前未发现的查询项特征对，减少了特征工程的负担。但是，当基础查询项目矩阵稀疏且排名较高时（例如具有特定偏好的用户或吸引力较小的项目），很难学习有效的查询和项目的低维表示形式。在这种情况下，大多数查询项对之间不应该有交互，但是密集特征将导致对所有查询项对的预测为非零，因此可能会过度泛化并提出不太相关的建议。另一方面，具有跨产品特征转换的线性模型可以用更少的参数记住这些“异常规则”。

在本文中，我们提出了一种广泛+深度学习框架，通过联合训练线性模型组件和神经网络组件来实现一个模型中的记忆和泛化，如图 1 所示。

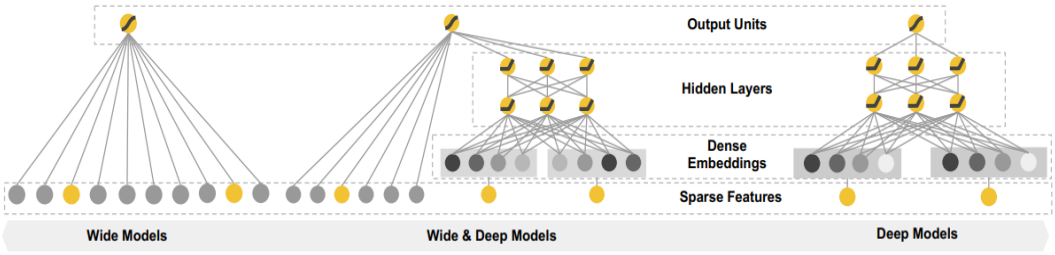

Figure 1: The spectrum of Wide & Deep models.

该论文的主要贡献包括：

- 针对具有稀疏特征输入的一般推荐系统，提出了一种基于密集特征的深度神经网络和基于特征变换的广度线性模型联合训练的广度+深度学习框架。

- 在 Google Play（是一家拥有超过 10 亿活跃用户和超过 100 万个应用的移动应用商店）上实现和评估度广度+深度模型应用后的推荐系统。
- 我们在 TensorFlow 中实现以一个高级的 API 开源了。

虽然思路简单，但我们证明了：广度+深度模型框架在满足训练和服务速度要求的同时，显著提高了移动应用商店的应用获取率。

2. 推荐系统概述

app 推荐程序系统的概述如图 2 所示。当用户访问 app 商城时，执行一个（包含各种用户及上下文特征信息）的查询。

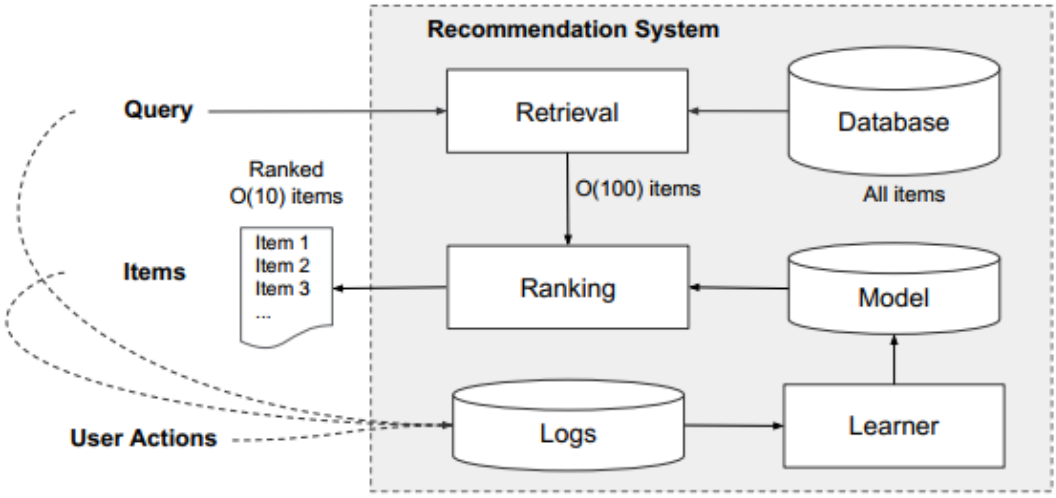

Figure 2: Overview of the recommender system.

推荐系统返回一个应用程序列表（也称为印象数），用户可以在其中执行某些操作，如点击或购买。这些用户操作,以及查询及查询结果印象数，将作为可学习的训练数据记录在日志中。

由于数据库中有超过一百万个应用程序，在服务延迟要求（通常为 0（10）毫秒）内，对每个应用程序的每个查询进行详尽的评分是很困难的。因此，接收到查询的第一步是检索。检索系统使用各种信号（通常是机器学习的模型和人工定义规则的组合）返回最匹配查询的简短列表。排名系统在将根据其得分对这些列表项目进行排名。分数通常为用函数 P（y | x）表示，向对于是给定 x 的特征对应用户动作标签 y 的概率 ，这些特性包括：用户特征（例如国家/地区，语言，人口统计资料），上下文特征（例如设备，一天中的时段，一天中的某天） 以及每周的展示次数功能（例如，应用程序上线年份，应用程序的历史统计信息）等。在本文中，我们重点研究使用广度+深度学习框架的排名模型。

3. 广度+深度模型的学习

3.1 广度部分的组成

广度模型部分的组成是，函数为 y = w ^ T * x + b 的广义线性模型，如图 1（左）所示。y 是预测值，x = [x1; x2; :::; xN]是 N 个特征的向量，w = [w1; w2; :::; wN]是模型参数，b 是偏差。特征集包括原始输入特征和转换特征。最重要的转换之一是特征叉乘转换，其定义如下：

$$\phi_k(\mathbf{x}) = \prod_{i=1}^{d} x_i^{c_{ki}} \quad c_{ki} \in \{0,1\} \qquad (1)$$

其中 cki 是一个布尔变量，如果第 i 个特征是第 k 个转换 Φk 的一部分，则为 1，否则为 0。对于二维特征，当且仅当组成特征（“gender=female”和“language=en”）均为 1 时，叉乘变换（gender=female，language=en）为 1，否则为 0。这捕获了二元特征之间的相互作用，并将非线性添加到广义线性模型中。

3.2 深度部分的组成

深度模型部分的组成是前馈神经网络，如图 1（右）所示。对于分类特征，原始输入是特征字符串（例如，“language=en”）。这些稀疏的、高维的分类特征中会先被转换成一个低维的、稠密的实值向量，通常被称为密集特征向量。密集特征向量的维数通常在 0（10）到 0（100）之间。对密集特征向量进行随机初始化后，分别将这些低维密集特征输入到前向通道中的神经网络隐藏层中进行训练，在模型训练过程中，不断地求解使损失函数最小。具体来说，每个隐藏层执行以下计算：

$$a^{(l+1)} = f(W^{(l)}a^{(l)} + b^{(l)}) \qquad (2)$$

其中 l 是层号，f 是激活函数，通常是 ReLU 。 a（l），b（l）和 W（l）分别是第 l 层的激活项，偏差和模型权重。

3.3 广深联合训练模型

以输出对数比的加权和作为预测，将广成分和深成分相结合，然后将其输入一个通用的 logistic 损失函数进行联合训练。注意，联合训练和集成是有区别的。在一个集成学习中，单独的模型在彼此不认识的情况下单独训练，它们的预测只在推理时组合，而不在训练时组合。相比之下，联合训练通过在训练时同时考虑广度和深度以及它们总和的权重来优化所有参数。模型大小也有影响：对于集成，由于训练是不相交的，因此每个单独的模型大小通常需要更大（例如，具有更多的特征和变换），以实现集成工作的合理精度。相比之下，对于联合训练，广度部分只需要用少量的跨产品特征转换来补充深度部分的弱点，而不需要全尺寸的广度模型。

广深模型的联合训练是利用小批量随机优化方法，将输出梯度同时反向传播到模型的宽深部分。在实验中，我们使用了以下正则化 leader（FTRL）算法[3]，其中 L1 正则化作为模型宽部分的优化器，AdaGrad[1]作为模型深部分的优化器。

组合模型如图 1（中间）所示。对于逻辑回归问题，模型的预测为：

$$P(Y=1|\mathbf{x}) = \sigma(\mathbf{w}_{wide}^{T}[\mathbf{x}, \phi(\mathbf{x})] + \mathbf{w}_{deep}^{T} a^{(l_f)} + b) \qquad (3)$$

其中 Y 是二元类标号，σ（·）是 sigmoid 函数，Φ（x）是原始特征 x 的叉积变换，b 是偏差项。W-wide 是所有宽模型权重的向量，W-deep 是应用于最终激活 a（lf）的权重。

4. 系统实现

应用程序推荐系统的实现包括三个阶段：数据生成，模型训练和模型服务，如图 3 所示。

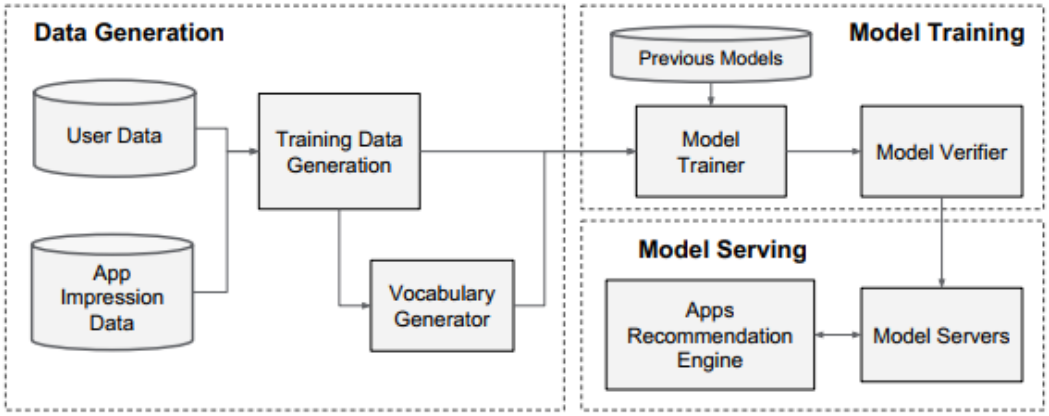

Figure 3: Apps recommendation pipeline overview.

4.1 数据生成

在此阶段，一段时间内的用户和应用印象数据将用于生成训练数据。每个样本示例对应一个印象。标签是应用程序获取：如果安装了印象深刻的应用程序，则为 1，否则为 0。

词汇表是将分类特征字符串映射到整数 id 的词表，也在这个阶段生成。系统为所有出现次数超过最小次数的字符串功能计算 ID 空间。通过将特征值 x 映射到其累积分布函数 P（X≤x），将连续实值特征归一化为[0，1]，并将其划分为 nq 分位数。对于第 i 分位数中的值，归一化值为（i-1）/（nq-1）。在数据生成期间计算分位数边界。

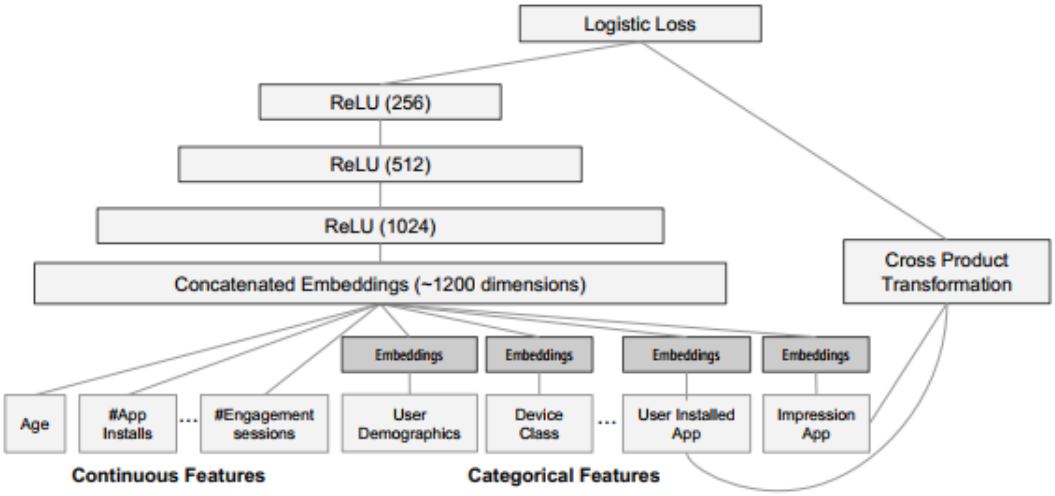

Figure 4: Wide & Deep model structure for apps recommendation.

4.2 模型训练

我们在实验中使用的模型结构如图 4 所示。在训练期间，我们的输入层接收训练数据和词汇表，并生成带标签的稀疏和密集的特征。广度部分的模型的组件包括用户安装的应用程序和展示应用程序的跨产品转换。对于模型的深度部分，将为从每个分类特征中学习出对应的 32 维度密集特征项量。我们将所有词表与密集特征联合在一起，从而产生近似 1200 维的密集向量。然后再将这个连接向量输入 3 个 ReLU 层，最后经由 logistic 单元输出。

广度和深度模型的训练样本超过 5000 亿了。每次收到一组新的训练数据时，都需要对模型进行重新训练。然而，每次从头开始的重新训练都会花上昂贵的计算上时间，并且会延迟提供更新后的模型时间。为了应对这一挑战，我们实施了热启动系统，该系统使用词向量和先前模型的线性模型权重来初始化新模型。

在将模型加载到模型服务器之前，需要对模型进行空运行以确保它不会在提供实时流量方面引起问题。我们根据先前的模型对模型质量进行了经验验证，作为一个健全性检查。

4.3 模型服务

对模型进行训练和验证后，我们才会将其加载到模型服务器中。对于每个请求，服务器从应用程序检索系统及用户特征接收一组应用程序待预测数据，来对每个应用程序进行评分。然后，从最高得分到最低得分对应用程序进行排名，然后按顺序将这些应用程序显示给用户。通过在 Wide＆Deep 模型上运行前向推理过程来计算分数。

为了满足 10 毫秒量级的每个请求，我们通过多线程并行运行来优化性能，方法是并行运行较小的批处理，而不是在整个批处理推理步骤中对所有候选应用程序评分。

5. 实验结果

为了评估广度和深度学习在现实推荐系统中的有效性，我们进行了现场实验，并从应用程序获取和服务表现两个方面对系统进行了评估。

5.1 应用程序获取

我们在 A / B 测试框架中进行了 3 周的在线实时实验。对于对照组，随机选择了 1%的用户，并向其提供由先前版本的排名模型生成的建议，该模型是高度优化的全范围逻辑回归模型，具有丰富的跨产品特征变换。对于实验组，向 1%的用户展示了由 Wide＆Deep 模型生成的建议，并接受了相同的特征

集的训练。如表 1 所示，与对照组相比，Wide&Deep 模型将应用商店主页上的应用程序获取率提高了+3.9%（具有统计学意义）。

Table 1: Offline & online metrics of different models. Online Acquisition Gain is relative to the control.

Model	Offline AUC	Online Acquisition Gain
Wide (control)	0.726	0%
Deep	0.722	+2.9%
Wide & Deep	0.728	+3.9%

结果还与另一个 1%的实验组进行了比较，实验组用相同的特性数据及相同的神经网络结构 使用得出，广度和深度模型比只使用深度模型的基础上有+1%的增益（具有统计学意义）。

除在线实验外，我们还通过离线 AUC 曲线面积来显示评价其效果。尽管 Wide&Deep 的离线 AUC 略高，但对在线流量的影响更为显著。一个可能的原因是离线数据集中的印象和标签是固定的，而在线系统可以通过将泛化和记忆结合起来产生新的探索性建议，并从新的用户响应中学习。

5.2 服务表现

在商业移动应用商店高流量下，提供高吞吐量和低延迟服务是一项挑战。在流量高峰时，我们的推荐服务器每秒可评价超过 1000 万个应用。对于单线程，在一个批次中为所有候选项打分需要 31 毫秒。我们实现了将每个批分成更小的批次分别多线程并行，这将客户端延迟显著减少到 14ms（包括服务开销），如表 2 所示。

Table 2: Serving latency vs. batch size and threads.

Batch size	Number of Threads	Serving Latency (ms)
200	1	31
100	2	17
50	4	14

6. 相关工作

将广度线性模型运算叉乘特征变换 以及 深神经网络运算密集特征相结合的想法是受到了前人工作的启发，例如因子分解机[5]，它通过将两个变量之间的相互作用作为两个低维嵌入向量之间的点积进行因子分解，从而为线性模型添加泛化。本文通过神经网络代替点积学习词表之间的高度非线性相互作用，扩展了模型容量。。

在语言模型中，已经提出了联合训练递归神经网络（RNNs）和具有 n-gram 特征的最大熵模型，以通过学习输入和输出之间的直接权重来显着降低 RNN 的复杂性（例如，隐藏层大小）[4]。在计算机视觉中，深度残差学习[2]已被用来减少训练更深层模型的难度并通过跳过一层或多层的快捷连接提高准确性。神经网络与图形模型的联合训练也已应用于图像的人体姿势判断预测[6]。在这项工作中，我们探索了前馈神经网络和线性模型的联合训练，并在稀疏特征和输出单元之间建立直接联系，以针对通用建议和稀疏输入数据的排序问题。

在推荐系统文献中，已经通过将内容信息的深度学习与评级矩阵的协作过滤（CF）耦合来探索协作深度学习[7]。之前也有关于移动应用推荐系统的工作，比如 AppJoy 在用户的应用使用记录中使用了 CF[8]。与之前的基于 CF 或基于内容的方法不同，我们联合针对应用推荐系统的用户和展示数据训练了 Wide＆Deep 模型。

7. 结论

记忆和泛化对于推荐系统都很重要。广度线性模型可以通过交叉乘特征变换有效地记忆稀疏特征交互，而深度神经网络可以通过低维密集特征将其推广到以前未发现的特征交互。我们提出了广度和深度学习框架，来结合两种类型的模型的优势。我们在大型商业应用商店 Google Play 的推荐系统上制作并评估了该框架。在线实验结果表明，广度+深度模型比单广度和单深度模型在应用程序获取方面有显著改善。

8. 参考文献

[1] 杜奇，艾哈桑，和艾辛格。在线学习和随机优化的自适应梯度方法。机器学习研究杂志，12:2121{2159 2011 年 7 月。

[2] 何恺明，张 X，任 S，孙 J。用于图像识别的深度残差学习。会报 IEEE 计算机视觉和模式识别会议，2016 年。

[3] H.B.麦克马汉。遵循正则化的先导和镜像下降：等价定理和 L1 正则化。在会报，艾斯塔茨（AISTATS），2011 年。

[4] T.Mikolov，A.Deoras，D.Povey，L.Burget 和 J.H.Cernocky。大规模神经网络语言模型的训练策略。在 IEEE 自动语音中认可与理解研讨会，2011 年。

[5] S.伦德尔。使用 libFM 的分解机（Factorization machines with libFM）。ACM Trans. Intell. Syst. Technol.，，3（3）：57:1{57:22，2012 年 5 月。

[6] 汤普森、杰恩、莱肯和布雷格勒。卷积网络与人体姿态估计图形模型的联合训练。在 Z.Ghahramani，M.Welling，C.Cortes，N.D.Lawrence 和 K.Q.Weinberger，编辑，NIPS，1799 页{1807。2014 年。

[7] H. Wang，N.Wang 和 D.-Y.Yeung. 推荐系统的协作深度学习。 在会报中 KDD，第 1235 页{1244，2015 年。

[8] B.Yan 和 G.Chen。AppJoy：个性化移动应用程序发现。在 MobiSys，113 页{1262011

补充说明

1、稀疏特征：他是离散值特征，且用 One-hot 表示，

如： 专业={计算机，经济学，其他}， ==》 经济学=[0,1,0];

如：词表={人工智能，你，他，目标，...} ===> 他=[0,0,1,0,...];

可以叉乘（点积）如上面两个叉乘=》 {（计算机，人工智能），（计算机，你），....}

叉乘之后：稀疏特征做叉乘获取共现信息； 实现记忆的效果

稀疏特征--优缺点：

优点： 有效，广泛应用，如 google，百度，亚马逊等

缺点： 需要人工设计，挑选几个有效的，（因为如所有特征都做叉乘，结果集合空间需要很大，同时会发生过拟合，相当于记住每一个样本），对于近意词泛化就差了。

2、密集特征：

向量表达

如：词表={人工智能，你，他，目标,...}　　==>　他[0.3,0.2,0.6,....(n 为向量)]

Word2vec 工具

向量之间距离衡量词语信息差距， 如： 男 - 女 == 国王 - 王后

密集特征的优缺点：

优点：带有语义信息，不同向量之间有相关性； 兼容没有出现过的特征组合； 更少人工参与

缺点： 过渡泛化，推荐不怎么相关的项目