**宜宾学院**

**本科毕业论文(设计)**

院(系)级班：\_\_\_\_\_\_\_\_\_ \_计算机与信息工程学院\_\_ \_\_\_\_\_\_\_

专 业：\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_计算机科学与技术\_\_\_\_\_\_\_\_\_\_

题 目：\_ 基于机器学习的电影推荐系统设计与实现 \_

指导教师 ：\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 王常远\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_

导师职称 ：\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_讲师\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_

学 号：\_\_\_\_\_\_\_\_\_\_\_\_ \_161101034\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_

姓 名：\_\_\_\_\_\_\_\_\_\_\_\_\_ \_银小宝\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

同 组 人 ：\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_无\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_

宜宾学院

2020年 5月

摘 要

科技在进步,信息产品迅速融入人类社会。同时信息给人们带来了很大的便利,但是任何事物都有好也有坏。推荐系统的流行和信息时代的发展有不可分割的关系。发展过程种也出现了很多丰富的算法和思想。矩阵分解是一类协同过滤模型，主要思想是将一个大的矩阵分解成两个规模较小的矩阵相乘。尽管矩阵分解（MF）模型在评估和预测任务方面取得了良好的效果，但它本质上是一个线性模型。由于MF是线性的，因此不能描述复杂的非线性模型和更高维度的联系，而这些联系对于预测用户偏好是很有用的。所以Sedhain等学者在2015发表的论文一种非线性神经网络协同过滤模型AutoRec就解决了MF的不足之处。因子分解机是一种监督学习的算法，它很快引起了人们的注意，并成为一种流行的、有影响力的预测和推荐方法。特别是对线性回归模型和矩阵分解模型的推广。利用深度神经网络还可以捕获输入信号的非线性固有结构。而深度因子分解机 （DeepFM），它将FM和神经网络相结合，更好的拟合高阶和低阶特征。

本文将对MovieLens的数据集做数据预处理工作，利用当前火热的深度学习框架Tensorflow2来实现DeepFM，最终将利用Python爬虫补充电影的数据，实现一个电影推荐系统。

关键词 机器学习；数据处理；推荐系统

Abstract

With the development of science and technology, the accompanying information products quickly integrated into people's lives. People are increasingly enjoying the convenience brought by the information age. The rise of recommendation systems is closely related to the development of the Internet. The recommendation system has many rich algorithms and ideas. Matrix factorization is a type of collaborative filtering model. The main idea is to decompose a large matrix into two smaller-scale matrices. Although the matrix factorization (MF) model has achieved good results in evaluation and prediction tasks, it is essentially a linear model. Because MF is linear, it cannot describe complex non-linear models and higher-dimensional connections, and these connections are useful for predicting user preferences. So the paper published by Sedhain and other scholars in 2015, a nonlinear neural network collaborative filtering model AutoRec, solves the shortcomings of MF. Factorization machine is a supervised algorithm proposed by Steffen Rendle in 2010, which can be used to classify, return and sort tasks. It quickly attracted people's attention and became a popular and influential method of prediction and recommendation. Especially for the generalization of linear regression model and matrix decomposition model. The deep neural network can also capture the nonlinear inherent structure of the input signal. The deep factorization machine (DeepFM) combines FM and neural network to better fit high-order and low-order features.

This paper will do data preprocessing on the MovieLens data set, use the current hot deep learning framework Tensorflow2 to implement DeepFM, and finally use the Python crawler to supplement movie data to implement a movie recommendation system.

**Key words**  machine learning; data processing; recommendation system;

目 录

[第1章 绪论 1](#_Toc40504482)

[1.1 研究背景 1](#_Toc40504483)

[1.2 国内外发展趋势 1](#_Toc40504484)

[1.3 论文结构 2](#_Toc40504485)

[第2章 机器学习 3](#_Toc40504486)

[2.1 机器学习的定义 3](#_Toc40504487)

[2.2 损失函数 3](#_Toc40504488)

[2.3 优化方法 3](#_Toc40504489)

[2.3.1 梯度下降法 3](#_Toc40504490)

[2.3.2 Mini-batch 梯度下降法 4](#_Toc40504491)

[2.3.3 Adam 4](#_Toc40504492)

[2.4 多层感知器 4](#_Toc40504493)

[2.5 One-Hot 编码 5](#_Toc40504494)

[第3章 推荐系统 6](#_Toc40504495)

[3.1 推荐系统介绍 6](#_Toc40504496)

[3.1.1 推荐系统的定义 6](#_Toc40504497)

[3.1.2 推荐系统的作用 7](#_Toc40504498)

[3.2 推荐算法介绍 7](#_Toc40504499)

[3.2.1 协同过滤 7](#_Toc40504500)

[3.2.2 矩阵分解 8](#_Toc40504501)

[3.2.3 自动编码器 9](#_Toc40504502)

[3.2.4 因子分解机 9](#_Toc40504503)

[3.2.5 深度因子分解机 10](#_Toc40504504)

[第4章 电影推荐系统设计与实现 11](#_Toc40504505)

[4.1 功能需求 11](#_Toc40504506)

[4.2 各个模块设计与实现 11](#_Toc40504507)

[4.2.1 数据收集与处理模块设计与实现 11](#_Toc40504508)

[4.2.2 推荐模型搭建与训练设计与实现 13](#_Toc40504509)

[4.2.3 爬虫模块设计与实现 17](#_Toc40504510)

[第5章 系统实验及分析 19](#_Toc40504511)

[5.1 实验设置 19](#_Toc40504512)

[5.1.1 参数设置 19](#_Toc40504513)

[5.1.2 模型设计 19](#_Toc40504514)

[5.1.3 实验数据集 21](#_Toc40504515)

[5.2 实验结果及分析 21](#_Toc40504516)

[5.2.1 独立提取用户/电影特征对DNN学习性能的影响 21](#_Toc40504517)

[5.2.2 FM/DeepFM对学习性能的影响 22](#_Toc40504518)

[结 论 24](#_Toc40504519)

[致 谢 25](#_Toc40504520)

[参考文献 26](#_Toc40504521)

第1章 绪论

1.1 研究背景

技术在不断发展，信息产品已迅速融入人们的生活，信息给人们带来极大的便利。但是任何事物都有好也有坏，技术也是如此。技术在进步，互联网也在发展。人们可以在网上快速低成本地制作和接收信息。因此，互联网上各种复杂庞大的信息不断增加，直接产生了大量的数据。太多的信息远远超出了人们所能接受的范围，这使得用户无法快速有效地找到他们所需要的有价值的信息。电商网站需要每天处理大量信息，信息接收者不仅很难找到适合自己的信息，信息提供者也面临着巨大的挑战。

随着信息变得越来越丰富，人们主动寻找信息的精力越来越少。当信息稀缺并且有充足的时间和精力时，用户将主动搜索信息。如今，在大数据时代用户可以通过搜索引擎和推荐系统进行搜索，不过，大量的信息让用户难以选择。

信息源以推荐的方式占有用户的时间和精力，这也是很多媒体或电商公司的主要利益。因为用户的时间和精力有它们自己的价值。吸引用户的能力是平台最具价值的资产。推荐的质量联系着平台上的内容被消费的程度，而消费的内容会带来平台的收入。平台将通过对平台内容的运营来保持用户的注意力，从而达到业务运营的目的。

1.2 国内外发展趋势

推荐系统的流行和信息时代的发展有不可分割的关系。协同过滤（CF）系统可以说起源于上个世纪的90年代。GroupLens研究小组曾经设计了一个名为GroupLens的新闻推荐系统[1]。

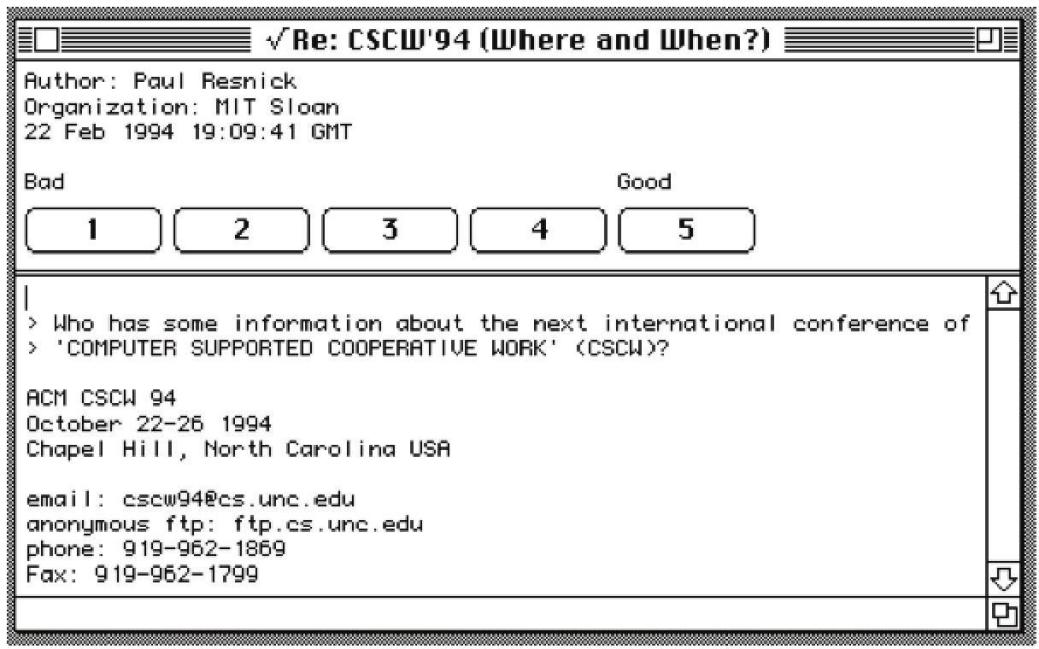


图1-1 Resnick推出的GroupLens系统

这项工作不仅首次提出了CF的思想，而且建立了一个正式的推荐问题模型，对今后推荐系统的发展产生了巨大的影响。后来他们创建了一个名为MovieLens 的推荐网站，这是一个对该领域影响极深的平台。

推荐系统背后的技术可以大致分为三种类型。1.基于内容的推荐模型。2.基于协同过滤的模型。3.混合模型[2]。

基于内容的推荐模型主要是分别建立用户和项目的矩阵，从而计算用户和项目之间的相似度。项目档案通常由各种属性数据组成。以衣服为例，它可以是价格、品牌、性别等。用户的记录可以来自人口统计数据，或者可以从他们的历史和交互项目记录中构建。例如，用户经常购买一种风格的服装，说明这个用户更喜欢这种风格，系统可以基于这一特点进行个性化定制。用户和对象文件建立后，可以直接计算相似度，也可以作为机器学习模型中的特征。基于内容的推荐模型的优点是，只要获得项目或用户的文件，它就可以处理冷启动问题。其次，因为文件都是显式的特征，所以最终的模型具有更好的可解释性。

基于协同过滤（CF）的推荐模型是一个比较热门的模型。它只需要收集一些日志，就可以探索用户和用户、项目和项目之间的潜在联系，根据这些联系进行预测。协同过滤模型可以分为两种类型：基于邻近的方法和基于模型的方法。

基于邻近的方法的核心是根据历史行为记录构建用户-用户或项目-项目相似度矩阵。一般来说，相似度计算方法可以是余弦相似度。基于模型的推荐中最常用的方法是隐式子模型。在这种模型中，用户和项目被嵌入到低维向量，中，用户和项目之间的相关性反映在它们对应的隐藏的向量点积关系中。这种方法的优点是比较高效率。一旦模型被训练好，点积就可以很容易地计算出用户和对象之间的关系，同时精度也有不错的表现。它的缺点也很明显。这解决不了冷启动的问题，而且学过的隐藏的向量也不容易解释。混合模型是指将各种推荐模型结合起来，相互学习对方的优点，以获得更好的推荐效果。

工业中常用的模型通常是混合模型。这种模型可以通过多种集成学习方法进行组合，也可以是端到端的模型，例如SVDFeature，Wide&Deep 等模型。

1.3 论文结构

本文主使用机器学习，对电影进行推荐，其中涉及一些数据处理，模型学习与预测，数据爬虫等相关内容。

本文的章节安排如下：

第一章 绪论。本章主要见了研究推荐系统的背景，国内外的发展趋势。

第三章 机器学习。主要介绍推荐系统所涉及的机器学习知识。

第三章 推荐系统。本章主要介绍了比较经典的几个推荐算法。

第四章 电影推荐系统设计与实现。本章主要介绍电影推荐系统实现相关。

第五章 系统实验及分析。本章主要对系统进行实验与对实验结果进行分析。

第2章 机器学习

2.1 机器学习的定义

机器学习就是利用和学习经验来来改善系统的性能

Mitchell 的论文[3]给出了一个更形式化的定义：假设利用P（Performace）评价程序在任务T（Task）上的性能，如果程序通过利用经验E（Experience）在T中任务上获得了性能方面的优化，那么就可以说关于T和P，这个程序对于E进行了学习。

2.2 损失函数

损失函数（Loss Function）就是计算预测值和真实值的一个函数与优化不可分割，因为在机器学习中需要通过最小化损失函数来优化模型。例如，它用于统计和机器学习中模型的参数估计[4]，宏观经济学中的风险管理和决策[5]，以及控制理论中的最优控制理论[6]。

2.3 优化方法

2.3.1 梯度下降法

梯度下降法[7]是最常用的优化算法。其思路是通过计算函数的导函数，找到当前位置梯度下降最快的方向，一般是损失函数切线方向，然后沿着这一方向进行优化。离最优值越近，下降的速度就越缓慢。迭代是梯度下降法的本质，一般需要多次的迭代，常用于求解机器学习算法的模型参数。每一次迭代，如公式2-1，就会对参数的一次更新。

 （2-1）

理想情况下，经过一定次数的迭代过后，损失函数所计算出的损失值将收敛，如图2-1 所示：

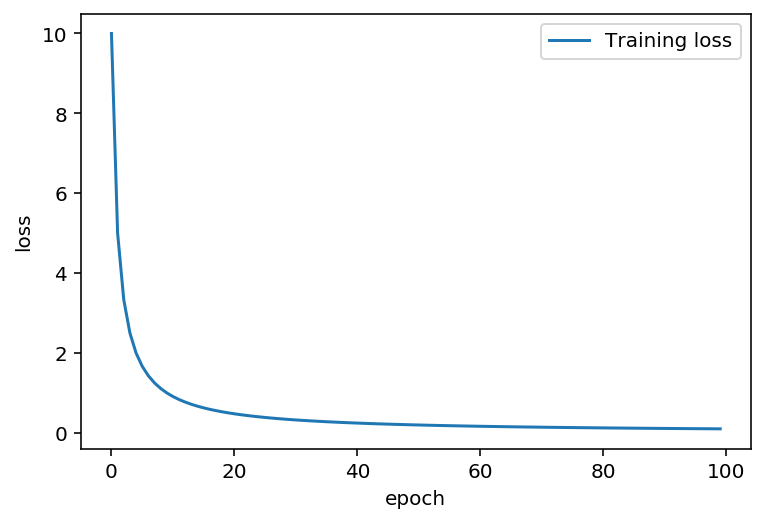


图2-1 Mini-batch

2.3.2 Mini-batch 梯度下降法

Mini-batch梯度下降法[8]就是对于*n*个样本，我们采用*m*个样本块来迭代，其中*1* < *m* < *n* 。Mini-batch用于加速训练过程，可能在迭代次数过小的情况下，损失值会出现不平滑的抖动，不过随着迭代次数增加，最终损失值会在稳定值附近波动，如图2-2所示

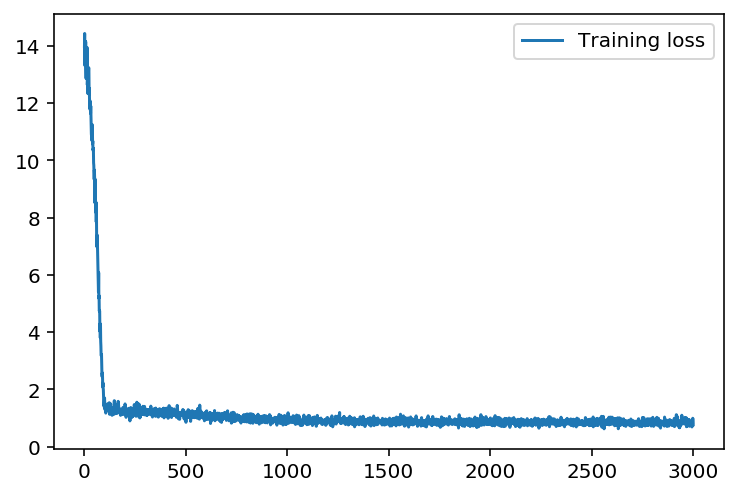


图2-2 Mini-batch

2.3.3 Adam

Adam即Adaptive Moment Estimation[9]，是一种能够自适应的估计方法，能够根据参数调节学习率。Adam在深度学习领域是一种很受欢迎的算法，因为它能很快取得好的成果。实证结果表明：在实践中，Adam的工作表现良好，并优于其他随机优化方法。

2.4 多层感知器

多层感知器(MLP, Multilayer Perceptron)[10]将多个输入数据集映射到单个输出数据集。它是一个前馈神经网络模型。

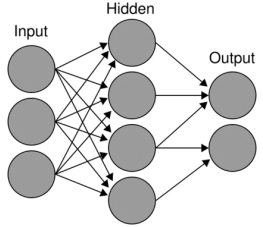


图2-3 MLP

多层感知器一般会有激活函数（Activation Function），激活函数起到值域映射的功能。常见的激活函数有Sigmoid函数，ReLU函数。

Sigmoid函数[11]也称为S形生长曲线，在不同的学科领域均有运用。信息科学中，由于该函数的导函数大于零的特点，常被用作神经网络的激活函数。它将输入变量映射到0到1之间。

整流线性单元(ReLU)[12]，又称修正线性单元，这是一种常见的且效果很好的激活函数，曲线如图2-4所示。ReLU将变量映射到非负实数。

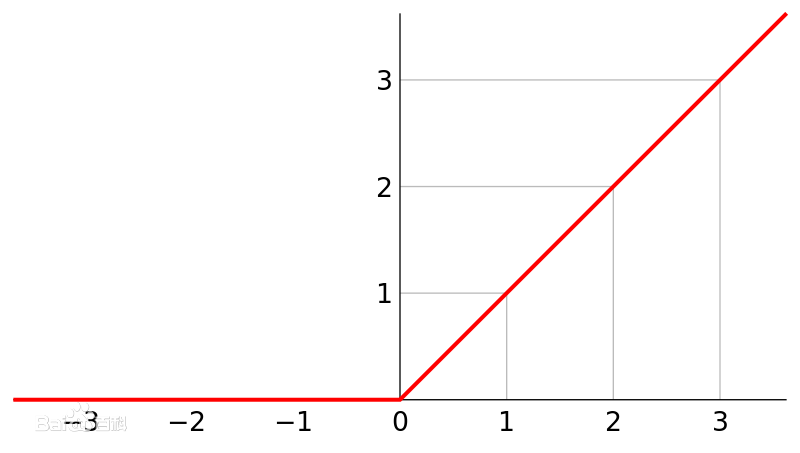


图2-4 ReLU

2.5 One-Hot 编码

One-Hot编码也可以说一比特有效编码，主要是将多个状态进行编码，例如性别可以分男女，使用One-Hot编码就将性别这一列转为男女两列，其中只有一列为1，如果是多列那么其余列均为0。

第3章 推荐系统

3.1 推荐系统介绍

推荐系统在人们生活着无处不在，帮助着用户挑选他们喜爱的项目，例如媒体，电子商务和电影。它微妙地影响着人们的生活[13]。

3.1.1 推荐系统的定义

面对快餐生活，人们越来越喜欢不用出门就能获得信息的方式。工作了一个星期后，有的人会在家中观看电影，而面对大量的电影信息难以选择合适自己的。为了解决信息过多引起的麻烦，采用了搜索引擎和推荐系统。人们可以通过在谷歌搜索中输入需要查找的信息关键字来找到他们需要的信息。但是，当需求不明确或者我不能用精确的语言描述需要的信息时，该怎么办呢?这时候推荐系统就有作用了。图2-1是推荐系统的简单结构图。

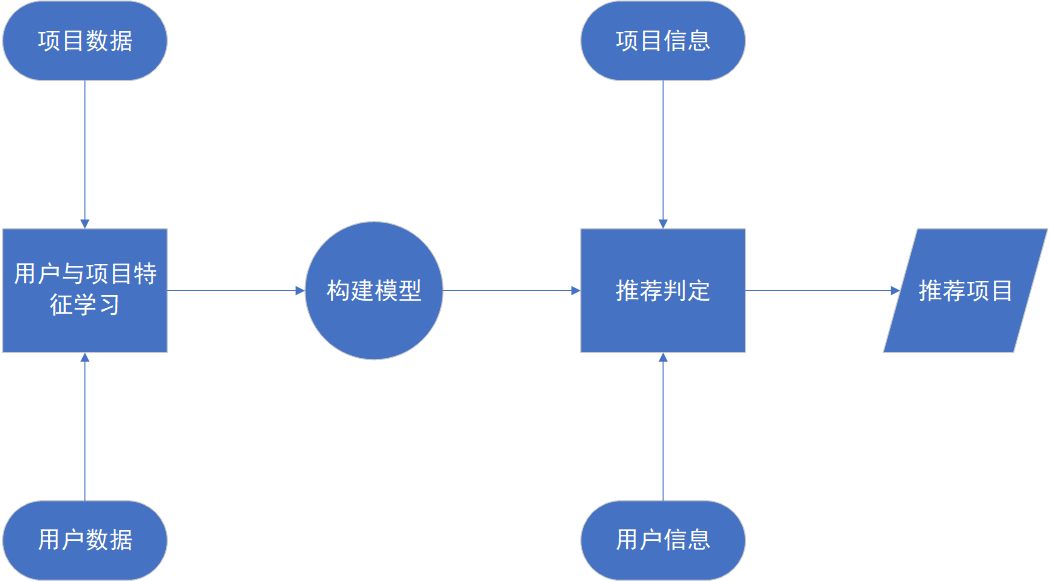


图3-1 推荐系统结构图

图2-2是JD推荐系统的业务架构。可以看书推荐系统主要包括用户，数据内容，和机器学习算法。



图3-2 京东推荐系统的业务架构

3.1.2 推荐系统的作用

推荐系统本质上就是一种过滤的方法，让信息更好的呈现在用户的面前。信息可以展示在感兴趣的用户面前，认识到信息存在的意义，最终使用户有更好的体验。它必须取决于特定的应用。随着Web 2.0的发展和推荐算法的成熟，不同领域的许多网站都结合了推荐系统，为用户提供个性化服务，如电影、音乐、电子商务等。

推荐系统在不同的领域有不同的特点。最成功和最重要的应用是在例如淘宝网这种电商网站。归根到底，这是因为一个好的推荐系统可以找到“长尾”商品“长尾”是由里斯安德森首先提出并定义的，“长尾商品”是需求差或销售差的产品[25]。例如，随着电子商务的迅速发展，淘宝，亚马逊和京东等，的年营业额达到数亿，远远超出了实体店的销售额。推荐系统就是来有针对性地探索用户的需求，向用户推荐这些商品，一方面来说也是满足了用户的需求，另一方面提升了销售效率。从某方面来看，电影视频网站也是一种电子商务网站，只销售电影视频。因此，推荐系统对于电影制造商和电影爱好者来说，都有很重要的意义。

3.2 推荐算法介绍

3.2.1 协同过滤

协同过滤(CF)是推荐系统中的一个重要概念，它首先在Tapestry系统[14]中使用。它提到了如何通过用户之间的关系来处理大量的邮件和消息。CF可以使用多种形式和方法。

如基于记忆的协同过滤(memory-based CF)，基于模型的协同过滤(model-based CF)，以及上述混合的[15]。

最典型的基于记忆的协同过滤技术是基于最近邻的协同过滤，如基于用户的协同过滤和基于物品的协同过滤[16]。一般来说，我们需要建立用户与项目之间的关系矩阵，而这种关系矩阵往往非常大，使得基于记忆的协同过滤在处理稀疏和大规模数据时存在局限性。

基于模型的协同过滤的一个例子是矩阵分解（MF）。基于模型的方法因其在处理稀疏性和可伸缩性方面的拥有更好的能力，而变得更加流行。许多基于类似于基于模型的协同过滤方法可以用过神经网络进行扩展，使得其更具有高度拟合的灵活性，计算时间开支的可靠性[17]。

通常来说协同过滤只使用用户和项目的关系就行推荐。除了协同过滤之外，基于内容和基于上下文的推荐系统在合并用户项目的内容方面也很有用。所以应当具体情况具体分析，当输入数据不同时，我们应该调整模型的结构使其适应。

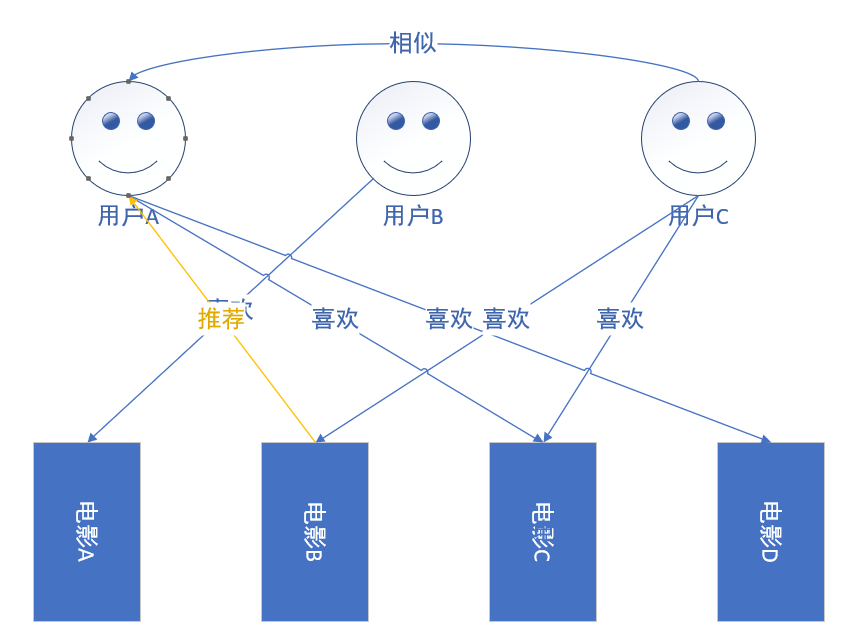


图3-3 基于用的协同过滤

3.2.2 矩阵分解

矩阵分解[引用矩阵分解的文献]的流行要归功于Netflix在2006年举办的一次竞赛，只要参赛者能在Netflix推荐算法基准上面提高10%的性能的最佳团队，那么将获得百万美金的奖金。尽管最终获奖者使用了多种算法的组合，当矩阵分解在里面起着至关重要的作用[18]。

矩阵分解是一类协同过滤模型。主要思想是将一个大的矩阵分解成两个规模较小的矩阵相乘。设内容用户/项目矩阵，其中 表示大小为的实数矩阵集合。然后用户项目矩阵将被分解为用户隐因子矩阵，和项目隐因子矩阵，*k*一般远小于*m*或*n*。于是，可以通过以下方式预测评分：

（2-1）

其中是评分预测矩阵，大小与用户项目矩阵相同，如图2-2所示。

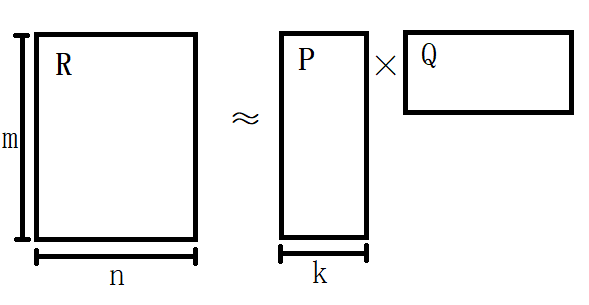


图3-4 矩阵分解示意图

3.2.3 自动编码器

尽管矩阵分解模型MF在评估和预测任务方面取得了良好的效果，但它本质上是一个线性模型。由于MF是线性的，因此不能描述复杂的非线性模型和更高维度的联系，而这些联系对于预测用户偏好是很有用的。所以Sedhain等学者在2015发表的论文[19]一种非线性神经网络协同过滤模型AutoRec就解决了MF的不足之处。它使用自动编码器结构来识别协同过滤CF，并基于显式反馈将非线性变换集成到协同过滤中。神经网络已被证明能够逼近任何连续函数，使其适用于解决矩阵分解的局限性，丰富了矩阵分解的表达能力。

一方面，AutoRec具有与自动编码器相同的结构，由输入层、隐藏层和重建(输出)层组成。自动编码器是学习将输入复制到输出的神经网络，以便将输入编码成隐藏的(通常是低维的)表示。在AutoRec中，它不是将用户/项目显式嵌入到低维空间中，而是使用交互矩阵的列/行作为输入，然后在输出层重构交互矩阵。

另一方面，与传统的自动编码器不同，AutoRec不学习隐藏的参数，而是专注于学习/重构输出层。它使用部分观察到的相互作用矩阵作为输入来重建完整的评估矩阵。同时，通过重构将缺失的输入项填充到输出层，达到推荐的目的。

模型方面，基于项目的AutoRec可以用公式表示：

（2-2）

其中表示评分矩阵的第*i*列，函数与函数代表激活函数，和表示权重矩阵，函数表示整个AutoRec的网络，所以表示评分矩阵第*i*列的重构。

3.2.4 因子分解机

因子分解机(Factorization machines, FM) [20]是Steffen Rendle在2010年提出的一种监督算法，可用于对任务进行分类、回归和排序。它很快引起了人们的注意，并成为一种流行的、有影响力的预测和推荐方法。特别是对线性回归模型和矩阵分解模型的推广。此外，它让人联想到带有多项式核的支持向量机。分解机器与线性回归和矩阵分解相比较的优势是:第一，它可以设计*X*-way变量的交互，其中*X*是多项式阶数，通常设置为2。第二，一种与因子分解机相结合的快速优化算法，可以将多项式的计算时间减少到线性复杂度，特别对于高维稀疏输入，效率极高。由于这些原因，因子分解机被广泛应用于现代广告和产品推荐中。

形式上，令表示样本的特征向量，y表示对应的标签，可以是值标签，也可以是类标签，如二分类“1/0”。二阶的因子分解机的模型定义为:

（2-3）

其中，是偏移变量；表示第*i*行的权重；表示特征嵌入；表示*V*的第*i*行；表示两个向量的点积。大多数其他特性交互都隐藏在数据中，很难识别。因此，对特征交互进行自动建模可以大大减少特征工程的工作量。很明显，前两项对应于线性回归模型，最后一项是矩阵分解模型的扩展。如果特征*i*代表一个项目，而特征*j*代表一个用户，那么第三项就是用户和项目嵌入之间的点积。

上面的是式子可以通过以下变换，将算法复杂度从*O*(*kd2*)优化到*O*(*kd*):

（2-4）

3.2.5 深度因子分解机

学习有效的特征组合是预测任务的性能关键。因子分解机模型通常以线性范式(例如，双线性交互)来描述。这通常是不够的真实世界的数据，固有的特征交叉结构通常是非常复杂和非线性的。更糟糕的是，在实际应用中，二阶特征交互通常用于因子分解机。然而利用因子分解机器对高阶特征组合进行建模在理论上是可行的，但由于数值不稳定和计算复杂度高，通常不被考虑采用。

一个有效的解决方案是使用深度神经网络。深度神经网络具有强大的特征表示学习能力，具有学习复杂特征交互的潜力。因此，将深度神经网络集成到因式分解机器中是很自然的。在因子分解机器中加入非线性变换层，使其能够对低阶特征组合和高阶特征组合进行建模。此外，利用深度神经网络还可以捕获输入信号的非线性固有结构。而deep factorization machines (DeepFM) [21]，它将FM和神经网络相结合，更好的拟合高阶和低阶特征，如图3-5所示。

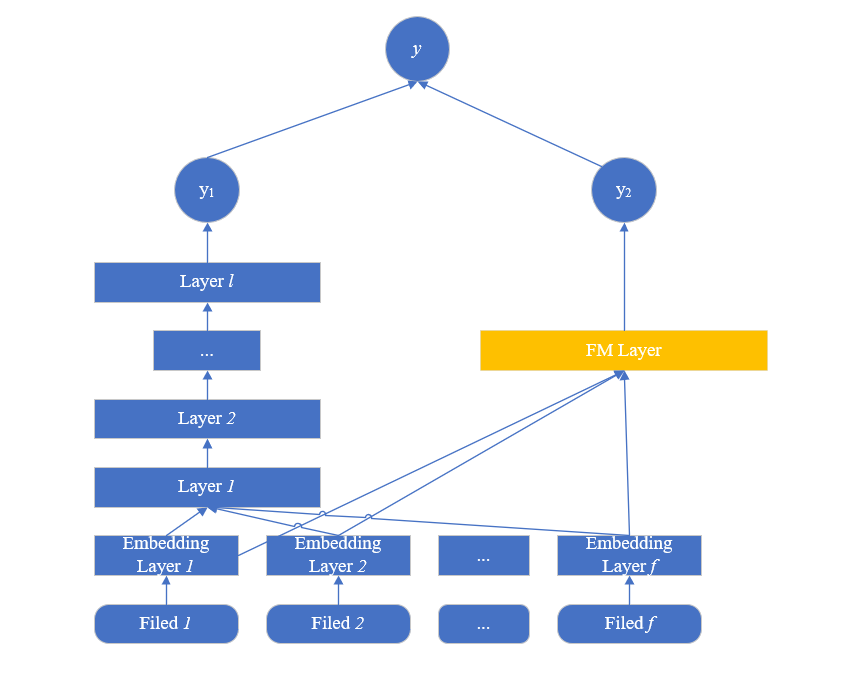


图3-5 DeepFM

第4章 电影推荐系统设计与实现

随着信息爆炸式地增长，人们在网络上找到自己感兴趣的电影信息的需求越来越迫切。本文将神经网络相关知识应用在电影推荐系统中，提供用户喜好相关的电影，提升用户体验感。

4.1 功能需求

电影评分，为我推荐，相似电影，喜欢该电影的人还喜欢，数据管理，日志记录

1.电影评分：该功能可以让用户对电影评分。

2.为我推荐：该功能为用户推荐可能喜欢的电影。

3.相似电影：该功能为电影关联相似的电影。

4.喜欢该电影的人还喜欢：该功能为一种协同过滤推荐电影的方式。

5.结果展示：展示推荐的电影。

4.2 各个模块设计与实现

系统的模块按照输入-处理-输出可分为，数据收集与处理模块、推荐算法模型搭建及训练预测模块、爬虫模块、推荐结果模块。

数据收集模块为推荐算法模型提供数据训练来源，显示模块为用户显示推荐信息。

4.2.1 数据收集与处理模块设计与实现

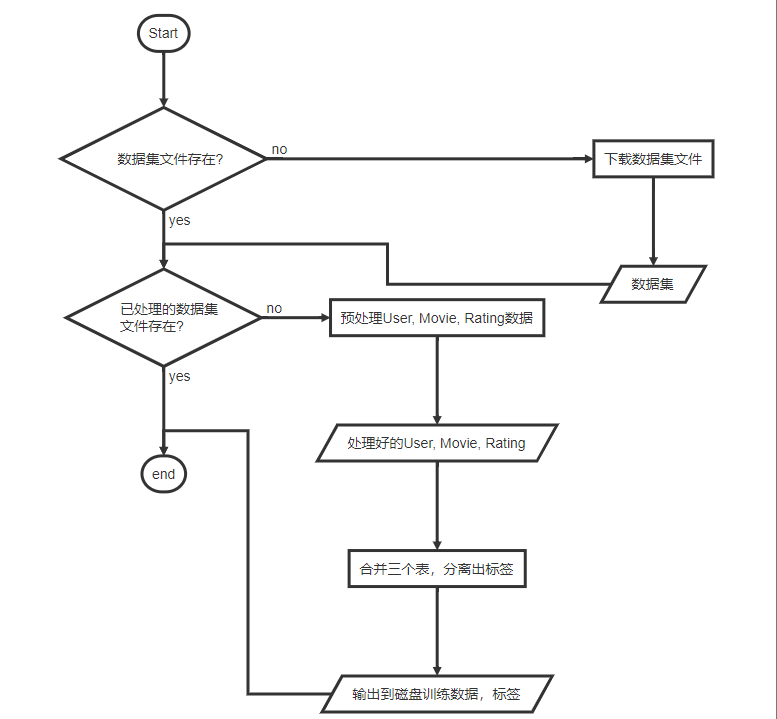


图4-1 数据收集与处理流程

通过编写代码，调用函数对数据集进行下载，然后分析数据，对MovieLens数据集经过格式化解码，并存入表格，最后将处理后的数据保存本地方便训练使用。实现部分如表4-1所示。

表4-1 数据处理

|  |  |  |
| --- | --- | --- |
| 序号 | 参数 | 值 |
| 1 | 编程语言 | Python |
| 2 | 数据集 | MovieLens-1M |
| 3 | 数据文件 | 电影，用户，评分文件 |
| 4 | 保存内容 | 处理后的电影特征集，处理后的用户特征集，评分数据，原始数据，训练用数据 |

查看用户数据文件前五行，如图，我们可以看到Gender字段的值是F和M，那么可以处理成0和1方便计算机进行计算。其中Zip-code邮编这一字段是对推荐系统没有太大影响力的信息，所以可以删除。

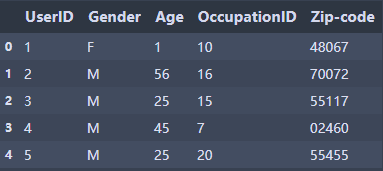


图4-2 用户信息

电影数据文件则如图4-6所示，可以看到电影的体裁Genres字段需要进行处理，数据集所提供的是字符串的类型，所以要转成分类。而电影的标题对于本文所使用的模型没有特征可以提取到，所以可以删除处理。

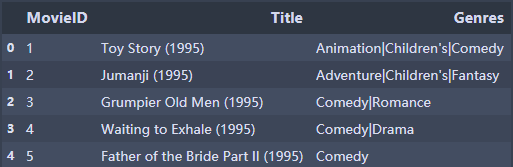


图4-3 电影信息

评分信息如图4-6所示，评分Rating就是系统需要学习的目标，即使得预测值更接近真实值。而时间戳信息是用户在什么时候评分的，对于系统也没影响力，所以可以删除处理。

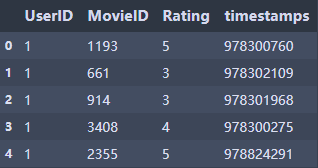


图4-4 评分信息

最终将处理好的文件保存到本地以便后面训练使用。

4.2.2 推荐模型搭建与训练设计与实现

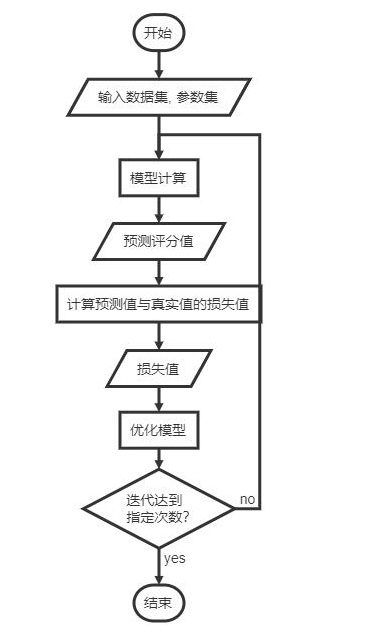


图4-5 模型训练流程

本文采用的是3.2.4深度因子分解机算法模型，用Tensorflow深度学习框架进行DeepFM模型的复现。然后适当的对模型就行改造，调节参数和优化，接着进行模型的训练，为以后使用模型进行推荐最好准备。模型训练环境如表格4-2所示，模型训练的流程如图4-5所示。其中，输入数据集就是数据收集与处理模块所保存的本地数据文件，参数设置在本文的5.1.1有详细的介绍。将数据输入用Tenseoflow搭建好的模型后，模型会输出一个预测评分值，根据模型输出的评分值去计算损失函数得到一个损失值，Tensorflow框架会根据损失值和设置的优化参数自动计算梯度进行优化模型。

表4-2 模型训练环境

|  |  |  |
| --- | --- | --- |
| 序号 | 参数 | 值 |
| 1 | 深度学习框架 | Tensorflow |
| 2 | 操作系统 | Windows 10 |
| 3 | GPU | GTX 1660 |
| 4 | CUDA版本 | CUDA 10.0 |

然后将训练好的模型上传到服务器，服务器的配置采用阿里云的云端服务器。

我们将模型里面训练好的用户特征和电影特征的结果提取出来，方便后面一些多功能推荐的使用。

为我推荐：为我推荐这个功能是核心功能，根据用户的历史点击记录行为，经过模型的训练，最后输入到模型，经过模型计算输出一个电影列表，取出根据预测评分排序的前k个电影作为结果返回。流程图如图4-6所示。

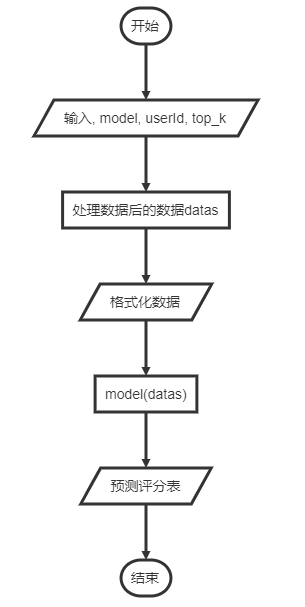


图4-6 为我推荐流程图

数据的处理是基于以下原理。模型的功能是对一个格式化的矩阵输出预测评分，矩阵的每一行是[uid, user\_gender, user\_age, user\_job, movie\_id, movie\_genres]这样的格式，一共有*m*行，*m*是电影的个数。而推荐的时候，需要做的数据处理工作就是将*1*个用户特征和*m*个电影特征进行连接。这里使用的连接方法是numpy的广播机制，如图4-7所示。

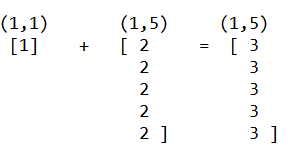


图4-7 numpy的广播机制

最后输出的数据是一个电影评分列表排序的电影ID，电影ID是和movielens网站的电影ID对应的，后面可以调用爬虫模块提取电影数据。

代码 4-1 为我推荐

|  |
| --- |
| *def get\_rating\_list(my\_model, user\_id\_val, top\_k=10):*  *# 数据格式化，因为模型需要格式化的输入*  *n = movies.values.shape[0]*  *genres = np.zeros([n, 18])*  *for i in range(n):*  *genres [i] = np.array(movies.values[i, 2]).astype(np.float32)*  *titles = np.zeros([n, sentences\_size])*  *for i in range(n):*  *titles[i] = np.array(movies.values[i, 1]).astype(np.float32)*  *np\_zeros = np.zeros([n, 1])*  *# 调用模型向前传播，返回评分表。*  *result = my\_model(x) #其中x是上面处理好信息的联合。*  *# argsort函数返回根据评分排序的索引（索引可以后面可以转成movieId，供给爬虫模块使用）*  *result = result.numpy()[:, 0].argsort()[:top\_k]*  *return result* |

2.推荐同类型的电影：根据指定电影的特征向量，与所有电影特征向量（特征向量集来自于推荐模型最后一层，见图5-2所示），计算余弦相似度（公式4-1），最终将相似度列表排序，取出最相似的前*k*个电影作为结果返回。

(4-1)

代码 4-2 同类型电影推荐

|  |
| --- |
| *def get\_same\_type(movie\_id\_val, top\_k=10):*  *# 余弦相似度Ai部分*  *norm\_movie\_matrix = tf.sqrt(tf.reduce\_sum(tf.square(movie\_matrix), 1, keepdims=True))*  *normalized\_movie\_matrix = movie\_matrix / norm\_movie\_matrix*  *# 电影特征矩阵与之单个电影特征相乘，得到相似度矩阵*  *embeddings = (movie\_matrix[mid2idx[movie\_id\_val]]).reshape([1, dnn\_ dims [1]])*  *similarity = tf.matmul(embeddings, tf.transpose(normalized\_movie \_ matrix))*  *# 得到相似度矩阵*  *m= similarity.numpy()*  *results = -m[0].argsort()[:top\_k]*  *return results* |

3.喜欢该电影的用户还喜欢哪些电影：根据该电影打分的用户列表，根据打分进行排行，最终以前*k*个用户的最喜欢的电影最为结果返回。

代码 4-3 喜欢该电影的用户还喜欢哪些电影

|  |
| --- |
| *def get\_other\_movie(movie\_id\_val, top\_k=20):*  *# 计算电影对所有用户评分*  *movie\_embeddings = (movie\_matrix[mid2idx[movie\_id\_val]]).reshape([1, dnn\_dims [1]])*  *user\_favorite\_similarity = tf.matmul(movie\_embeddings, tf.transpose (users\_ matrix))*  *# 取前k个最喜欢该电影的用户*  *favorite\_user\_id = np.argsort(user\_favorite\_similarity.numpy())[0][-top\_k:]*  *# 计算这些用户最喜欢的电影*  *users\_embeddings = (users\_matrix[favorite\_user\_id - 1]).reshape([-1, dnn\_dims[1]])*  *similarity = tf.matmul(users\_embeddings, tf.transpose(movie\_matrix))*  *# 排序返回索引结果*  *m = similarity.numpy()*  *results = -m[0].argsort()[:top\_k] # 将结果按评分排序，但返回索引*  *return results* |

4.2.3 爬虫模块设计与实现

网络爬虫，是一种按照一定的规则，自动地抓取万维网信息的程序或者脚本。

目的：本项目中爬虫的目的是为了补全作为训练数据集的movieLens数据集中电影数据的不足，以获得电影图片，导演，演员，等信息。

方法：利用chrome开发者工具进行网页结构的分析，python的request库进行数据的获取。

过程：首先movieLens只对登录用户提供网站浏览功能。所以python爬虫里面需要用的request的session做一个模拟浏览器的post登录请求。然后经过对网页的分析，发现获取电影json数据的API。

代码 4-4 电影数据json获取

|  |
| --- |
| *import requests as re*  *sess = re.session()*  *# 登录 movieLens 获取 cookie*  *r = sess.post(url='https://movielens.org/api/sessions', data=json.dumps({"userName": "nspyia", "password": "\*\*\*\*\*\*"}),headers=headers\_login )*  *# 根据 movieID 获取 json 格式的电影数据 https://movielens.org/api/movies/318*  *mid = 318*  *r = sess.get('https://movielens.org/api/movies/'+str(mid))*  *# 打印处json数据*  *print(r.json())* |

打印出电影数据json，图4-7所示，可以发现 imdbMovieId 字段可以用来获取更多专业的信息，导演，演员，背景图，youtube预告地址，电影简介，均在这个json格式里面。



图4-8 爬虫获取的部分电影数据

第5章 系统实验及分析

本文采用的是基于深度因子分解机（DeepFM）的一种电影推荐方法，接下来将介绍实现细节

5.1 实验设置

本文的实现 DeepFM 的细节将采用 MSE 损失函数来评估预测值和真实值之间的差别， Mini-batch 和 Adam 算法进行优化拟合，还有表5-1表明了一些对照组参数的设置。

5.1.1 参数设置

表5-1 参数设置

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 参数 | 值 | 解释 |
| 1 | dnn1\_dim | 300 | DeepFM的DNN部分第一层感知器个数 |
| 2 | dnn2\_dim | 200 | DeepFM的DNN部分第二层感知器个数 |
| 3 | n\_fm\_factor | 128 | DeepFM的FM部分第二层感知器个数 |
| 4 | embed\_dim | 32 | 嵌入层维度 |
| 5 | batch\_size | 256 | Mini-batch每一次迭代的batch大小 |
| 6 | learning\_rate | 0.0001 | 每个感知器的学习速率 |
| 7 | num\_epochs | 50 | 迭代次数 |

5.1.2 模型设计

按照上面的参数设置，图5-1就是模型结构示意图，其中Filed *i*代表第*i*个特征变量的经过One-hot 编码后的Filed，在本项目中，例如电影ID，电影类型，用户ID，用户职业等离散型特征变量，均为一个Filed，每一个Filed经过Embedding层就变成了一个32维度的向量，Embedding层可以看作一个特征提取的过程，最后分别经过DNN的高阶网络和FM低阶层过后，预测结果对高阶和低阶特征均有良好的表现。

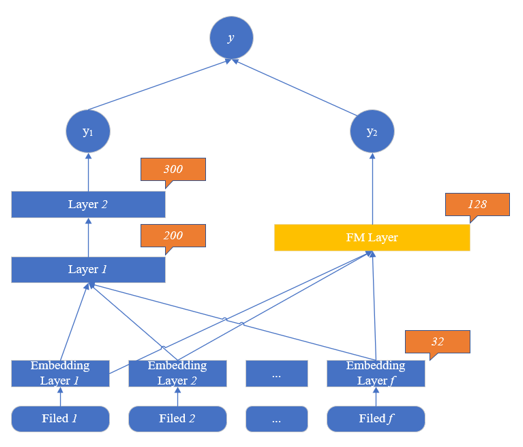


图5-1 DeepFM模型总体结构

而DNN这一部分为了体现独立提取用户和电影特征，这样做也许会对模型性能有所帮助，本文后面将讨论是否独立提取用户和电影特征对模型性能的影响。这里先给出如图5-2所示的模型设计，左边是单独提取用户特征，右边单独提取电影特征，中间是将混合提取用户电影特征，最后再将其输出。

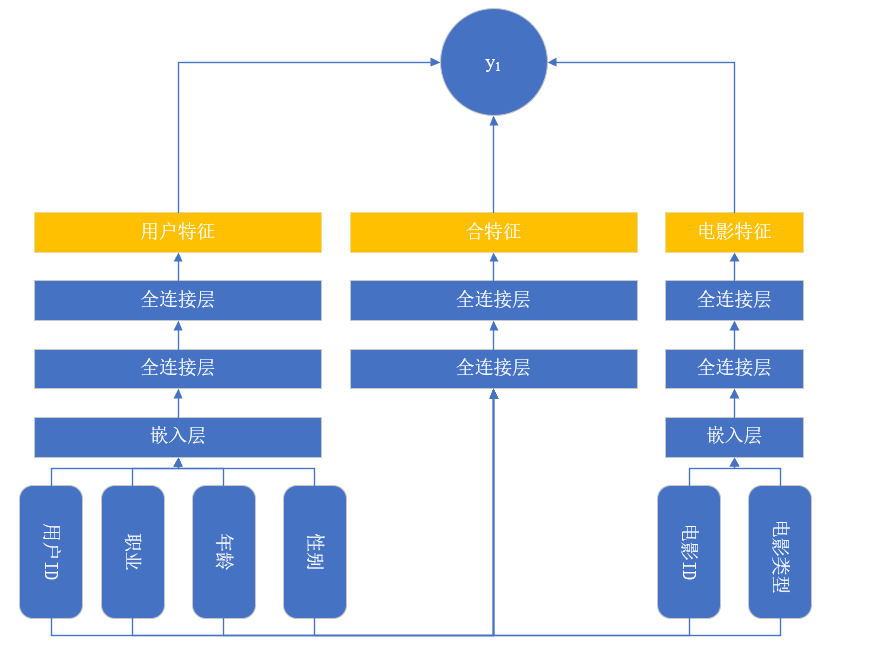


图5-2 DNN部分的模型结构

5.1.3 实验数据集

数据集是实验的必不可少的输入。在实验中，实验数据集通常分为训练集和测试集。训练集作为已知信息用于训练模型参数和优化推荐算法。测试集用于测试由训练集训练的模型，顾名思义。实验前，需要准备实验所需的电影数据集。为了保证实验的可靠性，本文使用了MovieLens上的电影数据集 MovieLens-1M。

5.2 实验结果及分析

5.2.1 独立提取用户/电影特征对DNN学习性能的影响

如图5-3所示，将用户和电影特征分开训练，最后在经过线性组合输出。这样做的好处是，可以得到两个分别的特征，在实现推荐功能时使用。得到这两个特征过后，就可以选择任意的方式来拟合评分了。而这里将讨论是否将独立提取与混合提取进行线性组合，来设计更优的DNN模型。

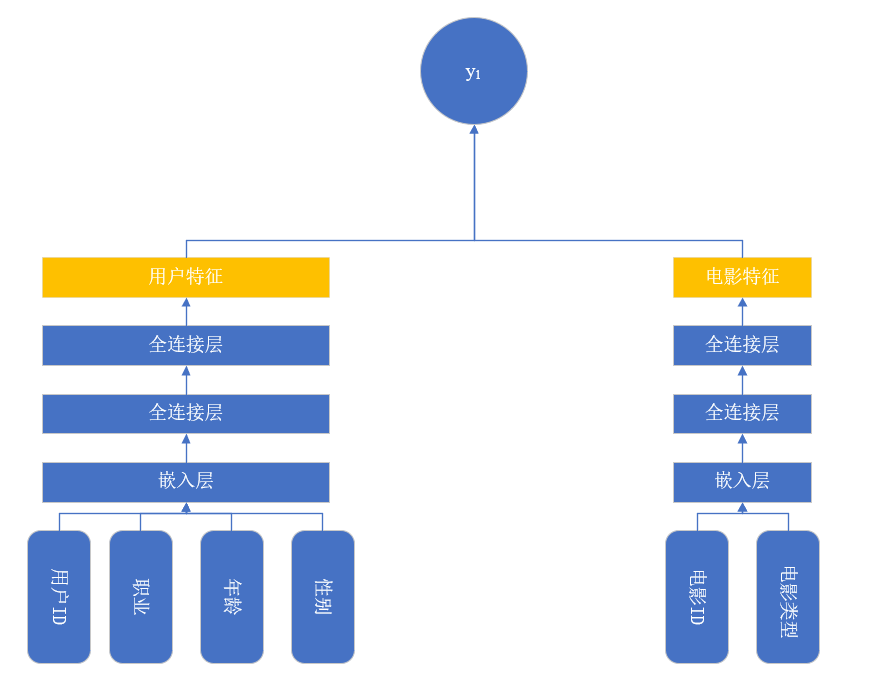


图5-3 独立提取用户/电影特征

当然这样做很容易忽略掉例如职业与电影类型这种关系的低阶特征组合，比如职业是医生的用户可能会喜欢一些关于医生类型的纪录片，所以本文设计了如图5-4融合提取特征结构图。

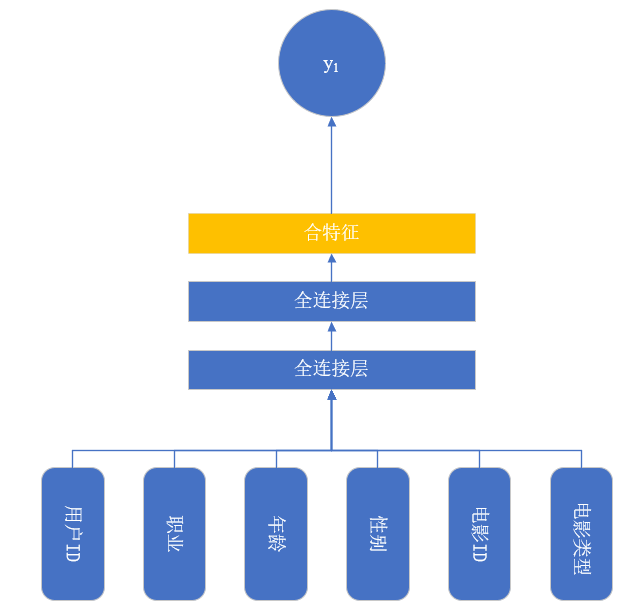


图5-4 融合提取特征

观察实验结果图5-5，不难发现曲线1分开提取的结果和曲线2融合提取的结果收敛速度上来说曲线1收敛更快。曲线3混合了这两种提取方式则使得模型更佳稳定，准确度也稍微高一些。说明将两种方式混合是有必要的。

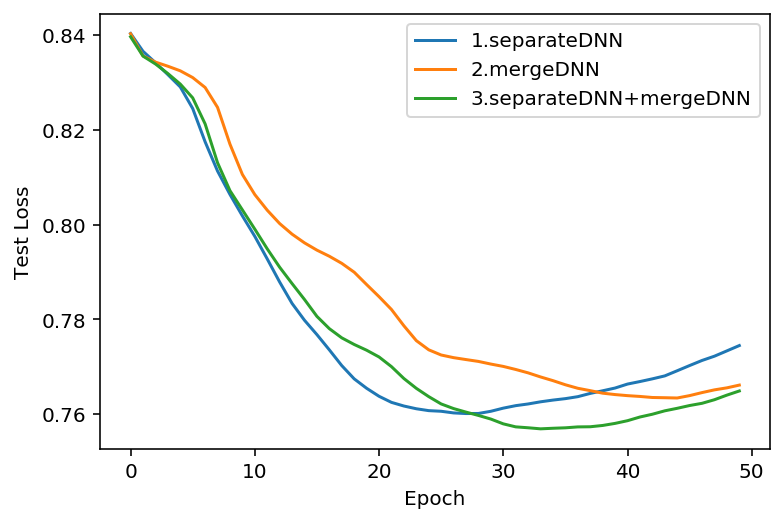


图5-5 特征提取方式对模型学习性能影响

5.2.2 FM/DeepFM对学习性能的影响

因子分解机(FM)通过特征之间隐含关系特征矩阵来提取特征效果通常表现很好。不过值得一提的是，FM虽然可以提取高阶的特征，但是由于算法时间复杂度很高，所以一般采用二阶的特征组合。

所以对于高阶的特征组合，我们需要使用另一种工具多层感知器，一般实际应用时可以称为DNN。DeepFM就是这样一种思维方式。图5-6是研究DNN的采用是否影响FM的性能。这里使用的DNN模型是图5-5中曲线3的模型。。

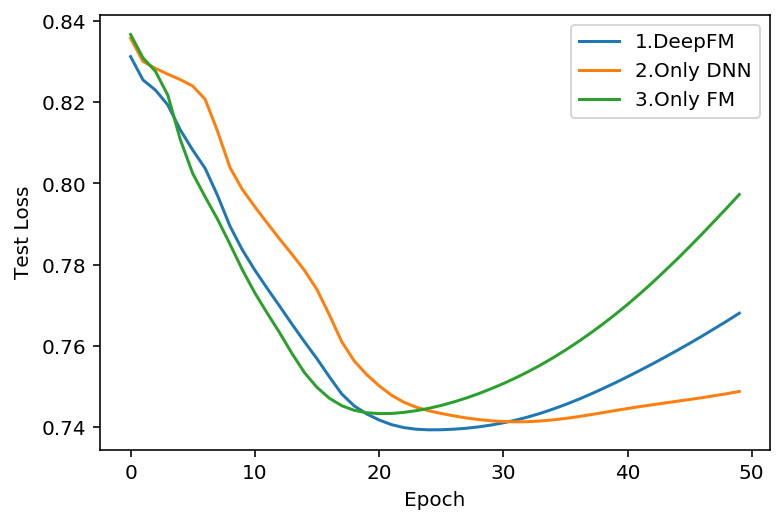


图5-6 网络结构

曲线3，也就是FM的结果，最先达到收敛，这也是因为FM层数比较低，训练速度比较快。

曲线1，也就是DeepFM的结果，可以看到准确度最高，收敛速度介于FM和没有FM的DNN层之间。这是很合理的结果，FM可以让模型更快地学习到低阶特征组合。

曲线2，也就是只有DNN的结果，可以看到收敛速度较慢，因为神经网络模型结构层次深，结构复杂，所以收敛速度慢。

结 论

推荐系统是一种信息过滤方法，可以消除信息过载带来的各种麻烦。本文介绍了推荐系统的常见算法，实现了深度因子分解机（DeepFM）。训练用的数据集采用的是MovieLens数据集，利用Tensorflow框架搭建模型。

对于深度因子分解机模型，本文研究了独立提取用户/电影特征对深度因子分解机的DNN部分学习性能的影响，以及FM/DeepFM对学习性能的影响。

实验结果表明独立提取特征对于网络性能是有优化的。而深度因子分解机比因子分解机（FM）有更好的性能表现，它联合训练了一个深层的感知器组件和一个因子分解机组件，所以能够学习高阶和低阶特征。

致 谢

“大学之道，在明明德，在亲民，在止于至善。”而在计算机与信息工程学院的这四年里，我不仅学到了为人之道，也获得了许多学业上的知识。

首先要感谢计算机学院为我提供了良好的校园环境，让我能够潜心学习；感谢领导、老师们四年来对我无微不至的关怀和指导，让我得以在这四年中学到很多有用的知识；感谢在班里同学和朋友，道之所存，师之所存也，感谢你们在我遇到困难的时候对我的帮助，为我指点迷津，也让我明白了合作的重要性；感谢我的家人，一直支持我的学业，给我以真挚的关怀与爱护。

在此还要尤为感谢我的指导教师王常远，在本系统的开发过程中给予我悉心指导，从系统的开发到结束过程中遇到很多困难和疑惑都是他给我鼓励与指引，使我能够克服重重困难，最终将系统完成，在此谨向王常远老师致以诚挚的谢意和崇高的敬意。

我将永远铭记师长们的谆谆教诲，在毕业后也要努力学习，不求功名，但求无愧于您们对我的栽培。于风雪中为吾抱薪者，怎敢忘怀！

参考文献

1. Resnick P, Iacovou N, Suchak M, et al. GroupLens: an open architecture for collaborative filtering of netnews[C]//Proceedings of the 1994 ACM conference on Computer supported cooperative work. 1994: 175-186.
2. 许海玲, 吴潇, 李晓东, 等. 互联网推荐系统比较研究[J]. 软件学报, 2009, 20(2): 350-362.
3. Mitchell T M. Machine learning [J]. 1997.
4. 邱锡鹏 著，神经网络与深度学习，第二章 机器学习概述．Github Inc.
5. Mayer, T., 2003. The macroeconomic loss function: A critical note. Applied Economics Letters, 10(6), pp.347-349.
6. Todorov, E., 2006. Optimal control theory. Bayesian brain: probabilistic approaches to neural coding, pp.269-298.
7. 刘颖超, 张纪元. 梯度下降法[J]. 南京理工大学学报 (自然科学版), 1993 (2): 2.
8. Li M, Zhang T, Chen Y, et al. Efficient mini-batch training for stochastic optimization[C]//Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. 2014: 661-670.
9. Kingma D P, Ba J. Adam: A method for stochastic optimization [J]. arXiv preprint arXiv:1412.6980, 2014.
10. Pal S K, Mitra S. Multilayer perceptron, fuzzy sets, classifiaction [J]. 1992.
11. Zadeh M R, Amin S, Khalili D, et al. Daily outflow prediction by multi layer perceptron with logistic sigmoid and tangent sigmoid activation functions [J]. Water resources management, 2010, 24(11): 2673-2688.
12. Schmidt-Hieber J. Nonparametric regression using deep neural networks with ReLU activation function [J]. arXiv preprint arXiv:1708.06633, 2017.
13. 刘建国 [1, 周涛, 汪秉宏 [1. 个性化推荐系统的研究进展[D]. , 2009.
14. Goldberg D, Nichols D, Oki B M, et al. Using collaborative filtering to weave an information tapestry [J]. Communications of the ACM, 1992, 35(12): 61-70.
15. Su X, Khoshgoftaar T M. A survey of collaborative filtering techniques [J]. Advances in artificial intelligence, 2009, 2009.
16. Sarwar B, Karypis G, Konstan J, et al. Item-based collaborative filtering recommendation algorithms[C]//Proceedings of the 10th international conference on World Wide Web. 2001: 285-295.
17. Zhang L, Zhang L, Du B, et al. Hyperspectral image unsupervised classification by robust manifold matrix factorization [J]. Information Sciences, 2019, 485: 154-169.
18. Andreas Töscher, Michael Jahrer. The BigChaos Solution to the Netflix Grand Prize [J]. netflix prize documentation, 2009.
19. Suvash Sedhain, Aditya Krishna Menon, Scott Sanner,等. AutoRec: Autoencoders Meet Collaborative Filtering[C]// International Conference on World Wide Web. ACM, 2015.
20. Rendle S. Factorization machines[C]//2010 IEEE International Conference on Data Mining. IEEE, 2010: 995-1000.
21. Guo D Y, Shi H Z, Qian Y P, et al. Fabrication of β-Ga2O3/ZnO heterojunction for solar-blind deep ultraviolet photodetection[J]. Semiconductor Science and Technology, 2017, 32(3): 03LT01.