推荐系统课程实验报告

实验计划及安排

霍超凡

1 实验概述

回想上学期的实验，我在一个数据集上对比了不同检索算法的性能，并且重点关注了布尔模型中英文词干提取和中文分词这两个子步骤对布尔检索模型性能的影响。本学期实验仍然参照上学期的传统，打算以实现算法、测数据为主，而不关注如何搭建一个具体的推荐系统，课程配套的实验和课程设计不同，它的重点在于实际动手操作所学的内容以加深对知识的理解，出于这样的考虑，我把自己的关注点放在算法的内部细节。

协同过滤推荐算法是众多算法中最成功的一种算法，在这领域有无穷无尽的文献可供参考，这方面的数据集也非常好找，计划沿着课程线索实现几个比较经典的算法，在数据集上测量这些算法的性能指标，在操作层面理解这些算法。下面将分别介绍实验的其它具体细节。

2 实验环境

Python 提供了大量丰富的库可供调用，可以快速实现自己的想法，选用 Python 作为编程语言，为了理解算法的内部细节，不提倡使用一些开源框架，但也不会从零开始编写算法，而是借鉴别人已经写好的代码，把自己的想法融入进去，通过阅读代码并修改是一个效率较高的提升自己编程能力的一种方法，我一直沿用。随着实验进程的推进，如果可以省出一些时间的话，计划实现神经网络的一些算法，关于深度学习的开源框架，推荐使用 Baidu 的 Paddle，这个框架发布的时间较近，它克服了诸如 tensorflow、Pytorch 的一些缺点，只有性能的提升或者有其它独特特性才能挑战传统框架，在 Paddle 的官方文档中实现了推荐系统的一些深度模型，文档详细，更重要的 Baidu 为开发者提供免费的计算资源，这是学生党的福音。

3 实验所用到的算法

读过文献的同学都知道协同过滤推荐算法的数量可以用 overwhelming 来形容，从上世纪末至今诞生了很多算法，不可能一一涉及，好在这些算法的思想很相近，一些算法相对另一些算法只是微小的改动，课程也基本涵盖了这些方法的核心思想，抓住主线，打算使用以下这些算法。协同过滤致力于推荐与已知物品相似的物品，其推荐的依据是用户和物品之间的评分矩阵，如果只考虑用户和用户之间的相似度，这是基于用户的协同过滤算法，对称地，如果只考虑物品之间的相似度，那么这是基于物品的协同过滤算法，如果考虑物品和用户之间的相似度，这是以分析矩阵为主的基于 SVD 的协同过滤算法，同时考虑以上所有相似度关系，这就是基于图模型的协同过滤算法的思想。仅仅考虑评分矩阵还不够，需要引入对用户、物品自身的内容描述信息，这就是混合基于内容推荐算法的协同过滤。

3.1 基于用户/物品的协同过滤

基于用户的协同过滤认为用户的历史购买记录反映这个用户的内在性质，同类用户具有相似的偏好，利用购买记录计算用户之间的相似度，参考用户的邻居（与该用户相似度较高的用户）的购买记录推荐物品。基于物品的协同过滤算法认为用户的偏好在短时间内不会发生变化，将与用户历史购买的物品相似的物品推荐给它。不管是基于用户还是基于物品的推荐算法都可以分成两大部分，首先要计算相似度，其次根据相似度加权求评分。在实验过程中应注意探讨以下几个方面：

- 不同相似度计算公式的性质及其对推荐结果的影响，讨论余弦相似度、皮尔逊相关系数和~~惩罚热门商品和偏好大众化用户~~的计算公式。
- 讨论对算法计算复杂度和空间复杂度修改之后的算法，主要是 k 近邻（Top-n）算法。
- 讨论 k 近邻算法中 k 的取值对结果的影响。

3.2 基于矩阵分解的协同过滤

基于领域的协同过滤算法存在计算复杂度、空间复杂度高的缺陷，基于矩阵分解的协同过滤算法转而分析评分矩阵，从评分矩阵中挖掘用户和物品之间的相似度。通过矩阵分解，物品和用户被向量化表示，使之映射到相同的语义空间中，评分矩阵中的评分反映物品和用户的相似度，用户对物品的评分越高，两者在语义空间中的距离越近。这种思想也体现在 word2vec 中，这里的物品、用户好比 word2vec 中的中心词、上下文词，通过一个优化函数或矩阵分解，将相关物品和用户、中心词和上下文词映射到相近的语义空间中，出现在相同上下文的中心词语义相近，与此对应的是，购买过相同物品的用户具有相同的偏好，所以结果语义分析得到的向量不仅可以反映用户与物品的相似度，还可以反映用户与用户之间、物品与物品之间的相关度。在实验中要着重讨论以下几个问题：

- 经过隐语义挖掘后所得到的向量的维度为多少合适？
- 比较经过矩阵分解和经过优化函数得到的模型之间的不同。
- 利用矩阵分解得到的隐语义向量结合上基于领域的协同过滤算法，相对于传统的基于领域的相同过滤是否有提升。

3.3 基于图模型的协同过滤

// 待完善

3.4 引入内容描述的协同过滤

// 待完善

4 衡量指标

评价算法的指标有很多，从结果方面考虑，可以分成两个方面：面向评分的和面向最终

推荐结果的，在面向评分的指标主要是 MAE、RMSE 等，得到评分不是我们推荐算法的最终目的，而是要把正确的商品推荐给用户，所以在面向最终推荐结果的评价指标中要考虑推荐的准确度、召回率、F 指标。从算法本身考虑，有算法的时间和空间计算复杂度。最后还有其它细节，算法的覆盖率等等。下面将详细介绍在实验中的测评方式：

机器学习算法最常用验证算法性能的方式是 k 折交叉验证，但是考虑到计算量的缘故，在实验中并不打算尝试，使用留出法，将原数据集划分形成训练集和测试集，训练集用于训练模型，测试集用于测试性能，有时会将算法在整个数据集上评测。在算法的运行效率方面，使用建立模型所耗费的时间来测试算法离线计算复杂度，使用模型为一名用户推荐若干个物品这个过程所需的时间来测试算法在线计算复杂度，除了以上两个计算时间复杂指标外，还应该测试空间复杂度，但是获取模型的具体占用内存空间多少比较复杂，在实验中只是简单分析。训练完成模型之后分别在训练集和测试集测量 MAE 和 RMSE 两个指标，对比测试集和训练集上的 MAE、RMSE 可以分析模型是否过拟合，MAE 是绝对平均误差，设用户 u 对物品 i 的实际评分为$\mathrm{r_{ui}}$，预测的结果为$\hat{r}_{ui}$，则 MAE 的计算方式为

$$\mathrm{MAE} = \frac{1}{n}\sum_{\mathrm{u,i}} |r_{ui} - \hat{r}_{ui}|$$

RMSE 为

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{ui} |r_{ui} - \hat{r}_{ui}|^2}$$

从式子上看，RMSE 比 MAE 稍严格些，先平方再平均，最后开方得到的结果会放大“大误差”对最终结果的影响，RMSE 要比 MAE 略大些。由于我们数据集中大多是评分，在计算精确度、召回率和 F 指标时，如何定义负样例是个问题，这里我们规定一个阈值，评分高于指定阈值设为正例，将没有评分（评分为 0）统一视为负样例。使用 nDCG 来衡量推荐列表的排列顺序，最后还有一个新鲜度的指标，那些数据集评分为 0 的占推荐列表的比重定义为新鲜度。

5 实验所使用的数据集

GroupLens 是 Minnesota 大学计算机科学工程学院的一个研究小组，他们的研究方向包含推荐系统，在他们的官网（https://grouplens.org）上有很多数据集，我实验所用的数据集大都来自他们所提供的数据集。

MovieLens 数据集

该数据集的数据是从 MovieLens 网站（http://movielens.org）收集的，根据收集的时间跨度不同，数据集从 100,000 条评分记录到 20,000,000 条记录不等，评分为 1-5 整数。这些数据集中部分数据集的包含用户对电影评分的时间信息、电影的标签信息、用户标签信息和用户对电影添加的标签信息。

Book-Crossing 数据集

该数据集是由 Cai-Nicolas Ziegler 花费四周时间从 Book-Crossing 网站（https://www.bookcrossing.com）爬取的，包含 278，858 用户对 271，379 本书的 1，149，780 条评分，包含用户的地理位置信息和年龄信息，评分从 0-10 不等。

Jester 数据集

该数据集包含了 73，421 个用户对 100 个笑话的 4.1M 条评分，评分从-10 到 10，原作者在所使用的算法在这个数据集上的 NMAE 为 0.2 左右，而随机猜评分可以达到 0.33。

Personality 数据集

该数据集包含了 1834 个用户的个性描述和对电影的偏好。个性描述包含用户的坦率、感情稳定性、严谨性、责任感等多种人格特性。

Netflix 数据集

这个数据集是一个 Netflix Prize 竞赛的一个数据集，目前官网已经不再提供该数据集的下载链接，但是在网络的其它位置可以找到它的备份链接。该数据集包含了 48 万个匿名用户对 1.7 个电影的评分。

6 实验时间安排

时间安排会随着进度动态调整，以下表格仅供参考。

时间	任务
第十周	实验规划
第十一周	基于用户\物品的协同过滤
第十二周	基于矩阵分解的协同过滤
第十三周	基于图模型的协同过滤
第十四周	基于深度学习的协同过滤
第十五周	
第十六周	整理最终报告