硕 士 学 位 论 文

推荐系统中动态推荐算法研究

A Study on Dynamic Recommendation Algorithms in Recommendation Systems

作者：唐伟康

导师：陈一帅

北京交通大学

2019 年 5 月

学位论文版权使用授权书

本学位论文作者完全了解北京交通大学有关保留、使用学位论文的规定。特授权北京交通大学可以将学位论文的全部或部分内容编入有关数据库进行检索，提供阅览服务，并采用影印、缩印或扫描等复制手段保存、汇编以供查阅和借阅。同意学校向国家有关部门或机构送交论文的复印件和磁盘。学校可以为存在馆际合作关系的兄弟高校用户提供文献传递服务和交换服务。

(保密的学位论文在解密后适用本授权说明)

学位论文作者签名：　　导师签名：

签字日期：　　年　　月　　日　　签字日期：　　年　　月　　日

学校代码：10004 密级：公开

北京交通大学

硕士学位论文

推荐系统中动态推荐算法研究

A Study on Dynamic Recommendation Algorithms in Recommendation Systems

作者姓名：唐伟康 学 号：16120125

导师姓名：陈一帅 职 称： 副教授

学位类别：工学 学位级别： 硕士

学科专业：通信与信息系统 研究方向： 信息网络

北京交通大学

2019 年 5 月

致谢

本论文的研究工作是在陈一帅老师的悉心指导下完成的。三年来，陈一帅老师如师如友，无论是学习上的指导，还是生活上的帮助，都是我能顺利完成硕士学业的关键。在学习上，陈老师有着灵活的教授方法，他的创新性的思维和广阔的学术视野总能在我科研陷入困境时给我灵感。三年来，陈老师无数次与我讨论学术问题，并肩调试代码，他对待工作一丝不苟和严谨科学的态度深深影响着我，激励着我不断前进，告诫自己路还很长。同时，陈一帅老师在生活上也给予了我无微不至的帮助，他总是在我骄傲时告诫我，失意时鼓励我，让我有勇气面对生活和学习上的各种困难。千言万语汇成一句话：感谢陈老师的培养与付出，我将继续努力，不负您的期望！

同样感谢郭宇春老师三年来对我的指导和帮助，三年来，多次与郭老师讨论学术难点，郭老师敏锐的洞察力，广阔的知识面总是能鞭辟入里地分析问题，带领我走出思维的误区，让我前进。同时，郭老师严谨的治学态度，对待工作认真负责的风范同样是我今后人生道路上的明灯和榜样，指引我一路向前！

感谢实验室的所有老师，感谢赵永祥老师为我论文提出的各种意见和建议，感谢张立军老师和李纯喜老师在研究生期间给我学习工作中的各种指导，让我受益匪浅。

另外，在实验室科研过程中，苏健、艾方哲、冯梦菲、陈滨等同学对我的科研工作给予了热心的帮助，他们扎实的工作方式、认真严谨的工作作风、灵活的思维及一丝不苟的态度都是我不断学习的内容。

最后感谢一直在默默支持我的父母、家人，正是他们的无私奉献才能让我完成学业，让我成为真正对社会有用的人。

摘要

互联网技术的飞速发展使人类进入了大数据时代，“信息过载”成为亟待解决的问题之一。推荐系统作为一种解决“信息过载”问题的技术已被广泛应用于互联网应用。

传统基于统计学习和深度学习的推荐技术通过定期更新模型来应对物品流行度的变化和候选集的更新，不能及时根据物品流行度的变化而更新推荐优先级，新物品加入时也不能迅速地完成冷启动。动态推荐算法（如多臂老虎机 Bandit）能够一定程度上解决上述问题，但准确度有待提高，这是因为：1）它们的模型能力有限，以 Contextual-Bandit 类算法中的 LinUCB 为例，LinUCB 算法采用线性模型拟合用户对特定物品的兴趣，表征能力有限，由此限制了算法性能；2）它们没有考虑用户特征分布的异质性，推荐效果不佳。

针对以上两个问题，本文选定新闻推荐作为动态推荐算法的具体场景，基于一个大规模、真实的在线新闻系统的用户行为日志，测量了该新闻推荐系统中新闻流行度的动态变化、新闻上下架的模式，观察了用户特征分布。基于观察结果，提出了两个算法来分别改进上述问题，并基于实际数据对算法进行了评估。主要贡献如下：

(1) 针对现有模型表达能力欠佳的问题，本文提出使用神经网络代替常规数学模型来建模用户和期望回报之间的关系，解决了神经网络在线更新和损失函数选择的两个难题。具体来说，为解决神经网络的在线训练在样本不均衡的情况下难以收敛的问题，我们提出了用户反馈敏感的训练方法：根据不同的用户反馈采用不同迭代次数。该方法相对于传统的训练方式取得了近 40%的增益。其次，本文将推荐问题建模为回归、分类和策略梯度问题，系统地尝试了分类、回归和策略梯度三种损失函数。通过实验发现：在合理的配置下，采用策略梯度的损失函数，我们的算法相较于 LinUCB 算法取得了 2.1%的性能增益，证明了算法的性能。

(2) 针对传统 Contextual-Bandit 算法没有考虑用户特征异质性的问题，本文创新性地提出了一种对用户特征敏感的分级推荐算法。该算法能够动态判别用户所属的类别，然后根据用户的类别，动态匹配合适的推荐器，来获得最佳的推荐性能。实验表明，该算法相较于传统的 LinUCB 推荐算法取得了 3.3% 的性能增益，证明了算法的性能。

本文在 Contextual-Bandit 动态推荐算法上的研究，进一步提高了当前主流动态推荐算法的性能，具有一定的理论价值和应用价值。

关键词：动态推荐；推荐系统；新闻推荐；上下文老虎机

ABSTRACT

With the rapid development of Internet technology, human beings have entered the era of big data, the problem of "information overload" has become one of the urgent problems to be solved. Recommendation system, which is one of the technologies to solve the problem of “information overload”, has been widely used in Internet applications.

Traditional recommendation technologies which based on statistical learning and deep learning cope with the dynamic change of item popularity and the update of recommendation candidate set by periodically updating the model, which can not update the recommendation priority in time according to the change of item popularity and the cold start process for new added items can not be completed quickly. Dynamic recommendation algorithms (such as Bandit) can solve the above problems to some extent. However, the performance also needs to be improved, that is because: 1) Their ability of models is limited, for example, LinUCB utilizes a linear model to fit the user’s interest on a particular item, the representational ability is limited, which caused the limitation of the model. 2) They do not take into account the heterogeneity of distribution for users’ features, the recommendation performance is not good.

In view of the above two problems, this paper chosen news recommendation as a specific scenario for dynamic recommendation algorithm study. Based on a large-scale and real-world online news system user behavior log, we measured and modeled the dynamic change of news popularity in the news recommendation system, the pattern of news for added or removed from candidate set, and observed the distribution of user features. Based on the observation results, we proposed two algorithms to solve the above two problems and evaluated our proposed algorithms based on the real-world data. The main contributions are as follows:

(1) Aiming at the problem of poor representational ability of existing models, this paper proposed to use neural network to model the relationship between the user feature and the expected reward, besides, we solved two difficult problems of neural network updating and loss function selection. Specifically, first of all, in order to solve the problem that the online training of neural network is difficult to converge in the case when the samples are unbalanced, we proposed a user feedback aware training method: feeding the neural network with different times according to different user feedbacks, compared to the traditional training method, this method achieves nearly 40% performance gain.

Secondly, this paper modeled the recommendation problem as regression, classification and policy gradient. We tried various loss functions, including classification, regression and policy gradient. Experiments show that under a reasonable configuration, using the loss function of policy gradient, our algorithm achieves 2.1% performance gain compared to LinUCB algorithm, which proves the performance of the algorithm.

(2) In view of the fact that the traditional Contextual-Bandit algorithm does not consider the heterogeneity of users' features, this paper proposed an innovative hierarchical recommendation algorithm which is aware of user features. This algorithm can dynamically identify the categories of users, and then dynamically match the appropriate recommenders according to the categories of users to obtain the best recommendation performance. Experiments show that the proposed algorithm achieves 3.3% performance gain compared to the traditional LinUCB recommendation algorithm, which proves the performance of the proposed algorithm.

The study on the Contextual-bandit recommendation algorithms in this paper, improved the performance of the current mainstream dynamic recommendation algorithms, has certain theoretical values and application values.

KEYWORDS ： dynamic recommendation; recommendation system; news recommendation , Contextual-Bandit

目录

缩略词表

英文缩写	英文全称	中文全称
NN	Neural Network	神经网络
RS	Recommendation System	推荐系统
PG	Policy Gradient	策略梯度
t-SNE	t-distributed stochastic neighbor embedding	t 分布随机邻域嵌入
CTR	Click Through Rate	点击率
RL	Reinforcement Learning	增强学习
UCB	Upper Confidence Bound	上界置信区间
LinUCB	Linear UCB	线性 UCB
LR	Logistic Regression	逻辑回归
SVD	Singular Value Decomposition	奇异值分解
EE	Exploitation-Exploration	探索利用

1 引言

1.1 研究背景及意义

随着互联网和通信技术的发展，人类已进入了大数据时代，推荐系统变得尤为重要。大数据时代一个重要的挑战就是“信息过载”(Information overload)，在当前时代，无论是信息的制造者、提供者还是信息的消费者都充满着挑战和机遇。一方面对于普通的互联网用户来说，网络信息量的激增使得人们从海量数据中获得自己感兴趣的信息变得越来越难。另一方面，对于信息的生产者和提供者来说，如何从海量数据中精准向用户提供感兴趣的信息也变得至关重要。推荐系统（Recommendation System）作为解决“信息过载”问题的一类算法，在互联网领域的地位举足轻重且已经成为了当下互联网生态系统的一个重要模块，已被业界广泛应用[1-3]。

推荐系统要解决的主要问题是针对不同口味的用户，精准地从推荐候选集中挑选出用户可能感兴趣的物品推荐给用户。若推荐的物品的确是当前用户感兴趣的，则用户很有可能接受该推荐，比如产生一个点击或者购买行为，推荐的物品若不是当前用户实际感兴趣的，则该用户有很大的可能性不会接受本次推荐。

提高推荐准确率是推荐系统的一个首要任务。用户接受推荐的概率为接受率，推荐系统的一个重要目的是尽可能提高推荐的被接受率，比如，在新闻推荐领域该指标为点击率 CTR（Click-Through-Rate）。在一个系统中，若用户的接受率很低，则说明该系统不能精准地捕获用户的口味，继而不能针对不同用户做出精准地推荐，那么当用户经常被推送到自己不感兴趣的物品时，用户的使用体验被极大地衰减，这很有可能造成系统的用户流失。因此，提高用户的推荐接受率是一个首要问题。

当前的推荐算法主要分为三大门类：1. 基于传统机器学习的方法；2. 基于深度学习的方法；3. 两者结合的方法。基于传统的机器学习的推荐方法，如协同过滤[4]、SVD[5]等方法，这一类方法根据用户的行为记录来进行推荐，但是这类算法在用户记录比较稀疏的时候会带来冷启动问题，只能在用户的行为数据比较丰富的时候能够取得不错的效果。近年来，随着深度学习技术的不断发展，自然语言处理（NLP）等深度学习技术在推荐系统也取得了较为广泛的应用。比如，word2vec 这一在 NLP 领域较为流行的算法被应用在推荐系统中对用户进行高维 Embedding 处理[6]，从而挖掘相似用户和相似信息。深度神经元网络的应用使得模型的拟合能

力更强，继而能够处理比较复杂的预测问题。另外一类方法将传统的机器学习方法和深度学习方法相结合，融合了传统方法和深度学习方法的优势。然而这些算法的一大特点是把推荐系统当作一个相对静态的系统，没有从细粒度上考虑待推荐物品的流行度变化趋势。比如，对于 word2vec 这一技术来说，往往是收集所有用户一段时间内的行为记录（如一天），然后进行模型训练，这类模型无法细粒度地考虑物品的流行度动态信息，同时也不能及时处理候选集的上下架给系统带来的问题。

在现实互联网环境中的很多推荐场景下(比如新闻推荐)，物品的流行度是高度动态化的，这会造成各种各样的问题。一方面，在一个新闻推荐的场景中，某一篇突发新闻的流行度可能在上线的几个小时内激增到达一个高潮后急剧下降，这种新闻的流行度的高度动态特性往往使得传统的推荐方法性能不佳。另一方面，推荐候选集的动态变化，往往会给推荐系统带来冷启动问题，这使得基于行为的矩阵分解的推荐方法表现不佳。因此，在这样的具有高度动态特性的系统中，必须设计一类动态的推荐方法来解决上述问题，所以，对动态推荐算法的研究意义重大。

1.2 国内外研究现状

本小节介绍主流的推荐技术，推荐系统的国内外研究现状以及现存研究存在的的问题，具体来说，可以分为以下几个方面：

1. **传统推荐技术通过定期更新模型，应对物品流行度的动态变化和推荐候选集的更新，不能及时处理物品流行度变化给系统带来的影响和新物品的加入带来的冷启动问题，无法适用于高度动态的推荐场景**

推荐技术的研究一直是学术界一个炙手可热的话题，在深度学习兴起之前，推荐系统一直采取传统的机器学习方法来进行推荐。比如，协同过滤、MF[7]、SVD 等技术的基本思路是将推荐问题抽象为矩阵补全的问题，其基本思路是利用已经收集到的用户的行为矩阵来对未知的用户-物品评分进行预测，然后选取具有最高评分的物品向用户进行推荐。这类算法的一个主要问题是当有新物品进入到推荐候选集合的时候，冷启动问题会严重影响推荐系统的性能，同时，这类算法也没有考虑时效性对于推荐系统的影响。

近年来，深度学习在推荐系统中的应用越来越广。例如，通过 Word2Vec 技术可以将物品、用户抽象为一个高维向量（embedding），抽象为高维空间向量之后，可以在高维空间中匹配相近的向量从而实现协同过滤、关联分析等功能。利用卷积神经网络（CNN）等技术可以提取待推荐物品的标题等文本信息来提高预测的准确率[8] 。此外，推荐问题也可以被抽象为序列预测问题[9]，这在一定程度上引入了

时序信息：将推荐问题引申为序列预测问题后，可以使用循环神经网络(RNN)、长短期记忆（LSTM）神经网络进行预测[10,11]。然而，这些先进的深度模型往往训练成本是巨大的，在训练数据量稀疏的时候往往效果不尽如人意。此外，传统深度学习方法同样不能很好地处理由于推荐候选集合的动态变化、即新上架的物品所带来的冷启动问题。

目前学术界对于推荐系统中物品流行度的动态变化的研究主要集中在待推荐物品的流行度变化上，一些存在的经典研究[12]系统地讨论了季节性、用户兴趣漂移对推荐系统带来的影响。例如，在夏季，裙子等夏装的推荐优先级要高一些，在冬季，这类服饰的推荐优先级应该比较低。此外，该文还详细地研究了用户的口味随着时间的变化规律，并提出了针对用户口味漂移所做出的改进方案。除了以上研究，将时间信息考虑进来的矩阵分解的变种，如 SVD++[13-15]等算法引入了时间信息，在一定程度上缓解了原始矩阵分解没有考虑时间信息的问题。但是上述算法都是针对待推荐物品流行度变化具有一定规律而且变化速率比较慢的推荐场景。然而在新闻推荐等类似推荐场景下，物品的流行度的变化往往是非常剧烈且无规律可循的。比如，一则新闻的流行度在几个小时内可能就会发生很大的变化（一则突发新闻可能会在短短几个小时内达到点击率高潮，然后急剧衰减下去），而且，不同的新闻的流行度的变化趋势也非常不同。除此之外，新的新闻的出现也是没有明确规律的，这也使得在新闻推荐中，系统的冷启动问题变得无规律可循。以上算法对于这种推荐场景并没有做深入研究。

学术界处理新物品冷启动问题的一类解决方案是学习新物品的基本属性（如类别、名称）到高维隐式空间的映射关系[16]，然后根据其学习到的新物品在高维空间中的相似性来进行推荐。这类算法的一大缺点是在很多情况下新物品的基本属性可能是未知的。在这种场景下，该类方法就不再适用，因此，也有着一定的局限性。

2.多臂老虎机算法已经被广泛应用于动态新闻推荐，大量的研究工作集中于用户特征和新闻得分的数学关系，模型表征能力存在瓶颈且普适性不强

多臂老虎机算法（Bandit）是增强学习中一类比较简单但是非常实用且应用面非常广泛的算法。Bandit 算法已经被广泛地应用在推荐系统、网络优化等相关问题上[17-20]。在新闻推荐场景中，每个待推荐的新闻都被当作是一个可以被采取的动作（臂），Bandit 算法根据每个新闻的推荐历史情况来计算被推荐的得分从而决定推荐结果。Bandit 算法已经被广泛地应用于推荐系统中新物品的冷启动问题中[21-25]，同时，由于 Bandit 算法是一种在线算法（online-algorithm），它可以实时侦测新闻的流行度，及时根据新闻的流行度调整其推荐的优先级。现如今，Bandit 算法已经被广泛地应用于工业界的推荐系统中[26]。

Bandit 算法根据是否利用了用户的特征可分为 Contextual-Bandit 和 Context free-Bandit 两类算法。LinUCB 算法是 Contextual-Bandit 推荐算法中最经典的一类算法，也是当下新闻推荐领域中最为流行的一类算法[19]，最先由 Yahoo 的科学家提出并且成功应用在 Yahoo 的新闻推荐场景中。UCB 算法每次计算每个动作的均值和置信区间的和，然后在动作集中选取具有最大值的动作进行推荐。为了充分利用用户的信息（context），LinUCB 算法创新性地将用户特征融入到 UCB 算法的均值估计和方差计算中。这种方法实现了推荐的个性化，相比较于传统的 UCB 算法，LiuUCB 算法取得了较高的准确率，开创了将用户特征融入到 Bandit 算法进行新闻动态推荐的先河。

自 LinUCB 算法以后，学术界受到 LinUCB 算法的启发，对于 Contextual-Bandit 算法做了较为广泛的研究，但是大部分都是数学模型上的修正与优化。LinUCB 的基本假设是对于每个新闻的预测得分和用户的特征之间是呈线性的关系。为了表征更复杂的关系，广义线性模型被提出用来模拟用户和预测均值之间的关系[27]；为了利用未观测到的一些隐式信息，也有研究在可以观测到的用户特征之外引入了一个隐式特征的概念用来建模未观测到的信息以提高推荐的准确率[28]。为了利用用户的关联信息，一种利用用户之间关联矩阵信息来提高预测准确率的方法也被提出[29,30]。此外，核方法等技术也同样被用来建模用户特征和建模得分之间的关系[31,32]，同时，也有基于动态记忆网络的设计思路来计算用户偏好的 Bandit 推荐算法[33]。然而，这些算法都假定新闻的得分与用户的特征（context）之间存在着某种特定的数学关系。而这种数学关系是可以有闭式解且通过在线更新的方式不断更新闭式解的参数来优化推荐性能。但是，在真实的场景下，并不是所有的推荐场景都符合我们特定的数学假设，因此普适性不强；此外，这些数学模型的表达能力有限，在某些复杂的场景下并不能很好地建模比较复杂的关系。

综上所述，推荐系统的研究一直是学术界的热门话题。机器学习、深度学习在推荐系统中都得到了非常广泛的应用[34-37]。在动态推荐算法领域，关于待推荐物品的流行度的动态变化特性，季节效应，用户的兴趣漂移等相关问题也有了较为细致的研究[12]，这类研究能够很好地处理动态性不很强的场景。此外，诸如新闻推荐等动态性比较强的场景下，Contextual-Bandit 推荐算法已经被广泛应用，而且基于 LinUCB 的众多的算法变种也都被提出[38-40]。但是这些模型都是基于特定的数学假设，普适性不强而且表达能力有限。

1.3 研究内容及难点

1.3.1 研究内容

在以上的介绍中我们可以发现，在新闻推荐领域，Bandit 算法一直是动态新闻推荐领域中一类比较重要的算法。但是，其大部分都是基于特定的数学假设，条件限定性强，模型表达能力有限。神经网络作为目前机器学习领域一类重要的分类、回归工具，并没有给输入和输出之间设置特定的假设和关系分布，完全通过历史数据拟合相关关系，因此，神经网络具备着强大泛化能力，相较于特定的数学模型下的设计，其可拓展性更强。基于此，本文拟采用神经网络来代替传统的数学模型，通过神经网络拟合用户特征和新闻预测得分之间的关系，然后在神经网络的基础上引入探索机制，即 Bandit 问题的一般解决方案，从而实现新闻的在线推荐，以期获得更高的推荐准确率。此外，本文也对用户的特征做了深度细致的测量和观察，发现了用户特征分布的一般规律，分析了在这种用户特征分布下传统的推荐算法可能存在的问题，为了进一步提高性能，我们也对用户特征分布不均匀的推荐场景做了研究和算法设计。

1.3.2 研究难点

本论文的研究难点主要体现在以下几个方面：

(1) **对于基于神经网络的推荐算法神经网络损失函数的设计和选取存在难点**。神经网络在传统的分类、和回归问题上已经取得了相当不错的效果，激活函数（ReLU、Sigmoid）的引入为神经网络带来了强大的非线形表征能力，可以使其拟合输入到输出之间复杂的非线形关系。与经典的分类、回归问题不同，在分类问题中，我们有每个样本分类的真实标签。这使得我们可以使用交叉熵来计算输出层向量的预测误差然后优化神经网络。然而在我们现在的在线推荐问题中，每次我们暴露给用户的新闻只有一个，用户会给我们明确的反馈（接受或者是拒绝），这就导致了我们只对于推荐过的新闻有真实的标签。然而，未被暴露出去的新闻的真实标签我们是未知的，因此，在这种情况下，如何设计和选取损失函数是一大设计难点。

(2) **对于基于神经网络的推荐算法如何有效地在线训练神经网络存在难点**。在传统的机器学习、深度学习领域，神经网络的训练一般都不是在线的（on-line）的。其一般的训练模式是在训练数据上不断学习从而拟合出输入到出的函数关系。然而，现在的新闻推荐场景是一个在线的、动态的推荐场景。神经网络每次推荐一篇新闻给当前到访系统的用户，然后根据用户的反馈来优化自身的策略，如此往复。因此，在本文的推荐场景中，神经网络的更新和训练与传统方式不同，如何有效地在线训练神经网络来让系统获得最佳性能也是一大值得探索和改进的问题。

(3) **如何设计合理的推荐算法解决在用户口味分布不均匀情况下使用一个推荐兴趣模型（推荐器）性能不高的问题**。当用户的特征非常发散的时候，如果只用

一个推荐器来对所有的用户进行推荐，则模型参数可能拟合不佳，性能可能达不到最好的效果。如何处理这个问题也是一大值得设计和思考的空间。

1.4 本论文的主要贡献

针对现存的主流动态推荐算法 LinUCB 存在的表征能力不强，没有考虑用户特征分布的两大问题，为了提高推荐的准确性。我们拟从两大角度对 LinUCB 算法存在优化。一种是通过神经网络的方式提高模型表达能力，另一种是通过解决 LinUCB 没有考虑用户特征分布的问题来提高推荐准确率。

本文的主要贡献如下：

(1) 针对现有模型表达能力欠佳的问题，我们提出使用神经网络代替常规数学模型来建模用户和期望回报之间的关系，解决了网络在线更新和损失函数选择的两个难题。具体来说，为解决神经网络的在线训练在样本不均衡的情况难以收敛的问题，我们提出了用户反馈敏感的训练方法：根据不同的用户反馈采用不同迭代次数，该方法相对于传统的训练方式取得了近 40%的增益。其次，我们将推荐问题建模为回归、分类和策略梯度问题，系统地尝试了分类、回归和策略梯度三种损失函数。通过实验发现：在合理的配置下，采用策略梯度的损失函数，我们的算法相较于 LinUCB 算法取得了 2.1%的性能增益，证明了算法的性能。

(2) 针对传统 Contextual-Bandit 算法没有考虑用户异质性的特点，我们创新性地提出了一种对用户特征敏感的分级推荐算法。该算法能够动态判别用户所属的类别，然后根据用户的类别，动态匹配合适的推荐器，来获得最佳的推荐性能。实验表明，该算法相较于传统的 LinUCB 推荐算法取得了 3.3% 的性能增益，证明了算法的性能。

1.5 本论文的组织结构

本文的组织结构如下：

第二章介绍当前学术界比较流行的动态新闻推荐算法：Contextual-Bandit 推荐算法。本章将重点介绍推荐系统中的 EE 问题，传统 Bandit 问题的解决方案。并对新闻推荐领域的一个经典的推荐算法 LinUCB 做了详细深入的介绍和探讨。同时，介绍本文中用到的技术的相关背景。

第三章基于 Yahoo 新闻推荐的大规模真实推荐记录，测量和建模了新闻推荐系统的一些特性。挖掘出了新闻流行度的动态变化特性，推荐候选集合中新闻的上下架等动态特性，同时给出了用户特征在空间内的分布。指出了以上两方面特性可

能给推荐系统带来的问题，传统的推荐算法在该场景下可能存在的问题以及动态新闻推荐算法的必要性。

第四章将新闻推荐抽象为回归、分类和策略梯度三类问题。基于以上三种损失函数，详细介绍了基于神经网络的动态新闻推荐算法的设计原型，探讨了几种损失函数设计的具体方案并且评估了基于神经网络的动态推荐算法在不同的三种损失函数下的性能表现。

第五章介绍了我们新提出的用户特征敏感的分级推荐算法，给出了算法的基本思想、算法结构和伪代码。并且评估了该算法在推荐问题上的性能，对于结果作了全方位、系统的分析。

第六章对本文的主要工作进行了概括，总结了本文的主要贡献点，并对未来的研究工作进行了展望。

2　技术背景

本章介绍相关技术背景。首先介绍推荐系统中的 EE 问题以及 Bandit 算法在推荐系统中的应用，详细介绍一类重要的算法：Contextual-Bandit 算法在动态新闻推荐领域的应用，介绍其基本思想和核心原理。然后，系统介绍 Contextual-Bandit 算法在学术界的国内外研究现状，指出本文的研究方向。最后介绍本文中应用到的一些基本知识作为后面章节的先验知识。

2.1　推荐系统中的 EE 问题

推荐系统的核心任务是从海量数据中匹配到用户感兴趣的侯选项推荐给用户，其最终的目标是使得推荐系统累积的被接受的数量最高。一个好的推荐系统能够精准地建模用户和物品之间的相关性并进行个性化推荐。然而，在现实的推荐场景中，每次都向用户推荐最为相关的物品有时候并不是最明智的。具体原因如下：

(1) 推荐的追打效应会造成推荐面的越来越窄，用户的新鲜感下降：

推荐系统的追打效应在推荐系统中尤为常见。追打效应描述的这样一种现象：举例来说：用户 A 购买了一本机器学习的书“machine learning”，与“machine learning”最接近的一本书可能是关于深度学习的书“deep learning”，而与“deep learning”最相关的书可能是推荐系统的书“recommendation system”，与“recommendation system”最相关的书可能又会是“machine learning”，这就形成了一种闭环。这种效应被称为推荐系统中的追打效应。由上面的分析可见，这种效应很容易造成“看什么推什么的”问题，这样就很容易给用户推荐的物品同属于一个类别，限制了用户的口味，使得用户对推荐的新鲜感与惊喜感下降，长久以来就会造成用户体验的下降甚至会造成用户的流失。

(2) 有些优质的资源会被埋没，不利于优质资源的挖掘：

在推荐系统中，有些物品可能还没有被充足的曝光，展现量不够。这样这些物品所获得的正反馈就相对较少，因此被推荐的可能性就会很低，然而这些物品中可能有一些很流行的资源，但是由于前期的推荐量少造成的推荐优先级低，就隐性地降低了推荐系统的推荐性能。这些比较流行的资源被“埋没”在推荐候选集中，无法定向投放给用户，使得推荐系统获得不了最大的推荐准确率。

在推荐系统中，每次给用户推荐匹配度最高的物品，这样的策略被称为利用（exploitation），由上面的分析可知，如果我们只采取利用策略，会造成上述的两个问题，并不能达到最大的收益（最大化推荐接受数量）。因此，为了缓解上述问

题，必须引入探索机制，让推荐系统以一定的策略去探索（exploration）推荐和用户匹配不太相关的物品，以期通过这样的方式去拓宽用户的口味，挖掘优质资源。但是，如何使用一个智能的策略去控制探索的力度非常重要。当探索的力度过小的时候，上述的两个问题不能得到很好地缓解，当探索力度很大的时候，用户的体验会急剧下降。因此，需要寻找到探索与利用之间的平衡以提高推荐系统的性能。寻找推荐系统中探索与利用的平衡和折中被称为推荐系统的 EE(exploitation-exploration)问题[41]。

2.2 Bandit 算法简介

解决 EE 问题一类非常著名的算法是多臂老虎机算法（Bandit 算法）[42]。Bandit 算法是解决 EE 问题的一类非常重要的算法。Bandit 起源于赌博学，它主要解决以下的问题：一个赌徒，要去摇老虎机，走进赌场一看，一排老虎机，外观一样，赌徒不知道每个老虎机背后吐钱的概率分布是什么，那么每次应该摇哪一个老虎机才能使得长久的收益最大化。一种最为简单的策略是每次都摇截止到当前实验吐钱概率最高的老虎机，这即是利用（exploitation），另一种方案是以较大的概率去推荐当前吐钱概率最高的老虎机，以较小的概率去探索吐钱概率相对较少的老虎机，通过这种方式来以期探索到更佳的老虎机，这即是探索（exploration）。因此，如何平衡探索和利用的力度，是 Bandit 算法主要考虑的问题。关于 Bandit 算法的相关研究算法较多，主要包含以下几类算法：

(1) $\varepsilon-$ greedy算法

该算法是解决 EE 问题最为简单的一种启发式算法。简单来讲，该算法的超参数只有一个，即：ε，该超参数是一个在（0，1）之间相对较小的数，每次我们以 1- ε 的概率选择当前期望回报最大的老虎机，以 ε 的概率从非期望收益最大的老虎机的集合中随机选取一个老虎机。即：以 1- ε 的概率进行利用，以 ε 的概率探索。这类算法相对于简单的贪心策略引入了探索机制，然而，这是一种启发式算法，最佳的超参数 ε 往往不好确定，在大多数情况下并不能使系统达到最佳的效果。

(2) 置信区间上届算法（UCB）

UCB 算法是 2002 年提出的一种解决 Bandit 问题的算法[43]，一直被广泛应用于解决 Bandit 问题。UCB 算法将置信区间考虑在内，为每一个老虎机维持了两个要素：期望和方差（置信区间）。该算法是一个不确定性乐观的算法，每次该算法挑选具有最大的期望方差两项之和的动作。其中，每个动作 j 的得分公式为：

$$\bar{x}_j + \sqrt{\frac{2\ln n}{n_j}} \tag{2-1}$$

其中，$\bar{x}_j$ 是动作 j 的平均期望，n_j是该动作当前被选择的次数，n 是当前所有的动作总共被选择的次数。该算法对于每个可以选择的动作（老虎机）都计算这个得分，然后挑选出具有最大得分的动作。可见，当一个动作被实验次数较少的时候，它的第二项（方差）较大，最终的分较高，因此被采取的可能性较大（探索）。当一个动作实验次数较多的时候，第二项趋近于零，该动作得分的主要决定为第一项，即该动作的平均得分。UCB 算法相比教于 $\varepsilon-greedy$ 算法的一大优势是将开发和探索融为一体，且智能调节了探索的力度，因此能够获得相对较佳的性能。

(3) Softmax 选择策略

Softmax 选择策略的基本思路是对于高收益的动作选择概率（权重）相对要高，对于低收益的动作选择概率相对要低。这种思想通过概率的方式实现探索和利用的统一。具体来说，假设共有 N 个动作可以选择，$Q(j)$ 表示第 j 个动作的估计平均收益，W_j 代表着该动作的权重（即采取该动作的概率）。则动作的权重为：

$$W_j = \frac{Q(j)}{\sum_{i=1}^{N} Q(i)} \tag{2-2}$$

实际中经常采用的计算方式为：

$$\mathrm{W_j} = \frac{Q_n(j)-\min(Q_n)}{\sum_{i=1}^{N}[Q_n(j)-\min(Q_n)]} \tag{2-3}$$

通过以上两个公式，每个动作以一定的概率被选择。由于概率的存在，收益较大的动作以高概率被选择，这样就实现了探索和利用的均衡。在实际的算法中，经常引入一个超参数 τ 来放缩推荐的权重。当 τ 非常小时，则该策略更加趋近于利用，当 τ 非常大时，更趋向于探索。特殊地，当τ 取值为 0 时，该策略退化为贪心策略，当 τ 取值为无穷大时，该策略退化为随机策略。

(4) 汤普森采样（Thompson sampling）

汤普森采样是贝叶斯学派解决 Bandit 问题的一种策略。该算法为每个动作期望背后都维护了一个 Beta 分布，然后每次从每个动作所服从的 Beta 分布中抽样出一个随机数出来，再挑选出最大的随机数所对应的动作执行该动作。得到该动作的反馈后修正该动作的后验 Beta 分布，通过不断地修正这个后验分布得到每个动作真实的期望概率分布。当动作的实验次数较少的时候，该动作的 Beta 分布的密度函数分布范围较广，更趋向于探索；当动作的实验次数较多的时候，该动作的 Beta 分布的密度函数分布范围较窄，更趋向于利用。

以上几种解决 Bandit 问题的算法都是一个动态的、在线更新的算法。都是不断地通过与用户进行交互来修正自己的策略，然后以期得到最大的收益。以上几种解决 Bandit 问题的方法及许多变种已经被广泛应用于推荐系统、网络优化。

2.3 基于 Bandit 的动态推荐算法

本小节我们重点介绍 Bandit 在推荐算法领域中的应用，以及推荐问题在 Bandit 问题中的抽象，重点讲解一类重要的动态推荐算法：Contextual-Bandit 推荐算法的基本原理和研究现状，指出现有研究工作相对较少的区域，然后指出本文的主要研究切入点。

2.3.1 Bandit 算法在推荐系统中的应用

与传统的协同过滤、深度学习等算法不同。Bandit 算法是一种动态的、在线的算法。Bandit 算法不断地与环境环境进行交互，然后更新自身的策略，与传统的离线算法相比，更加适合处理时效性给系统所带来的问题。在推荐系统中，每个可以把推荐的候选物品被抽象为一个动作（老虎机），Bandit 算法在这个场景中正是通过每次选取不同的动作（推荐物品）来使得长远的收益（点击/购买数量）最大。另外一点非常重要的是，由于 Bandit 算法在 EE 问题上的出色表现，该算法也被广泛应用于推荐系统中的冷启动问题，对于新物品的冷启动问题，该类算法可以在新物品上架的时候很快地试验出该新物品的流行度，然后根据试验结果决定之后的推荐优先级，完成新物品的冷启动。综上所述，Bandit 算法现如今已经被广泛应用于推荐系统领域且取得了不错的效果：Bandit 算法的 EE 特性使得推荐系统的冷启动问题得到缓解，Bandit 算法的在线、动态特性使得推荐系统能够实时调整推荐物品的流行度变化给系统带来的影响。

2.3.2 上下文 Contextual-Bandit 推荐算法

(1) Contextual-Bandit 的基本原理

前面对 Bandit 算法的基本原理、基本解决方案做了相关介绍。传统的 Bandit 解决方案是基于统计学的，每次算法统计每个动作的收益均值、置信区间等要素，然后根据不同的算法计算出推荐权重进行推荐。这种传统的 Bandit 推荐算法没有考虑用户的特征，完全基于统计特性，因此，个性化程度不高，继而达不到较好的推荐效果。基于上下文的 Contextual-Bandit 推荐算法的一大创新点是将用户的特征考虑在内，该算法的基本出发点是认为每个动作（推荐物品）的期望收益与用户的特征之间存在着某种关系（比如：线性）。然后不断观察到访的用户的特征，为每个待推荐的物品计算出得分，最后引入探索机制，实现动态推荐。该类算法首次被 Yahoo 的科学家们提出[19]，并且被广泛应用于新闻推荐等动态推荐领域。在该算法

中，其基本的假设是物品的期望收益与用户的特征呈现线性关系，然后利用线性模型（岭回归）求解各个参数，在线优化参数，以期得到最大的累积收益。通过这种方式，不仅实现了新闻推荐的个性化，而且解决了 EE 问题。

(2) 岭回归简介

岭回归（ridge regression）是一种著名的线性回归技术，它是一种改良的最小二乘估计法。通过正则项的加入可以解决过拟合、矩阵不可逆等问题，虽然加入了一定的扰动，但是往往能取得不错的效果，在病态数据上表现良好。在回归问题中，一种比较常见的解决方案是通过最小二乘法进行求解，它的目标函数如下：

$$\min\|X\theta - Y\|^2 \tag{2-4}$$

该目标函数可以通过梯度下降的方式不断优化参数得到局部最优解，也可以通过解线性方程组的形式得到闭式解如下：

$$\theta = (X^T X)^{-1} X^T Y \tag{2-5}$$

然而，在很多情况下$X^T X$是不可逆的，因此会造成以上方程无解。为了解决这个问题和过拟合问题，岭回归在目标函数后面加入了正则项：

$$\min\|X\theta - Y\|^2 + \|\alpha\theta\|^2 \tag{2-6}$$

最后我们可以通过求解以上的目标函数得到该目标函数的闭式解，其中α是正则化参数。该目标函数的闭式解为：

$$\theta = (X^T X + \alpha I)^{-1} X^T Y \tag{2-7}$$

由于加入了正则项，解决了该方程组在绝大多数情况下不可解的问题，而且，正则项的加入缓解了过拟合问题。

(3) LinUCB 简介

LinUCB 算法是 Contextual-Bandit 中最为著名的一类算法，它开创了将 context 信息引入到 Bandit 算法中的先河，已被众多学术研究当作一个基准算法[19]。它的基本假设是每个动作的期望回报与用户的特征之间呈线性的关系，并且使用岭回归进行拟合，再利用 UCB 的思想对用户推荐新闻。

当 LinUCB 经过每次推荐，得到用户真实的反馈后，会根据当前用户的 context 向量，用户的反馈和文章的 id 更新被推荐过的新闻的 θ 向量，如此往复不断优化 θ 向量的准确度。LinUCB 每次得到用户的反馈后都会更新自身的模型参数，因此是一个在线的算法。该算法可以根据用户的反馈不断调整自己的策略，因此，可以及时降低非流行新闻的推荐优先级和提高热门新闻的推荐优先级。

其基本的符号及其含义如表 2-1 所示：

表 2-1 LinUCB 符号对照表
Table 2-1 Symbol description of LinUCB

符号	含义
Λ	候选物品集合
x_t	t 时刻用户特征
θ_a	每个推荐物品的权重向量
$r_{t,a}$	在 t 时刻推荐 a得到的真实回报
A_a	新闻a被过去被推荐过的特征记录矩阵
b_a	新闻 a的历史回报反馈值存储矩阵
a	推荐集合中待推荐的物品

将 UCB 算法应用进来，考虑每个物品的期望回报是用户特征的线性组合，采用岭回归，并且使用在线迭代的方式，则有以下算法：

算法 2-1 Linear UCB [19]
Algorithm 2-1 Linear UCB[19]

Algorithm 2-1: Linear UCB Bandit Algorithm

0: **Input**: α
1:**for** t =1,2,3….T, **do**
2: Observe the current feature x_t
3: **for** all $a \in \Lambda$:
4: **if** a is new **then** :
5: $A_a \leftarrow I_d$
6: $b_a \leftarrow 0_{(d,1)}$
7: **end if**
8: $\theta_a \longleftarrow A_a^{-1} b_a$
9: $p_{t,a} \leftarrow \theta_a x_t + \alpha\sqrt{x_t^T A_a^{-1} x_t}$
10: **end for**
11: Choose arm $a_t = argmax_{a\in\Lambda}\, p_{t,a}$ and observe a real-valued payoff $r_{t,a}$
12: $A_a \leftarrow A_a + x_t x_t^T$
13: $b_a \leftarrow b_a + r_{t,a} x_t$
14: **end for**

由算法 2-1 可见，该算法正是通过不断地迭代为每个推荐物品维护了一个 θ_a 向量，对于当前到访的用户，该物品的最终的期望回报是用户的特征与该向量的内积，即第 9 行的第一项，其中第九行的第二项为线性回归的误差上限，已由 Walsh

et. al 在文章中证明。超参数 α 控制探索的力度。在第 12 行和第 13 行，在得到了本次推荐的真实反馈后（用户接受或不接受），用户的向量和本次推荐的结果将被更新到当前被推荐物品的参数矩阵中，从而该物品的参数向量 θ_a得到更新。从数学角度上来理解，对于每一个物品，对于特定的用户产生推荐并得到反馈（接受/不接受），实际上就是得到了一个以用户的特征向量为线性方程系数，θ_a为解向量，用户反馈（0，1）为结果的方程。该方程会对在原有的基础上在线、增量地修改已有的方程组得出的解向量。每产生一次有效推荐，则多一条约束方程，被推荐的物品的参数向量 θ_a 就会得到一次更新。

2.4 Contextual-Bandit 推荐算法研究现状

LinUCB 作为 Contextual-Bandit 的开山之作，创新性地将用户的特征引入到 Bandit 算法中，与以往的 Context-free 的算法相比，该算法更加个性化，也因此获得了更高的推荐准确率。继 LinUCB 之后，受 Contextual-Bandit 算法的启发，学术界对基于 context 的 Bandit 推荐算法做了较为深入的研究，本小节将全面综述 Contextual-Bandit 算法的研究现状。

基本的 Contextual-Bandit 推荐算法：LinUCB 假定物品的期望得分与用户的特征向量之间服从严格的线性关系，这种约束在大多数情况下都过于严格，为了使得模型有着更大的适用性，基于广义线性模型的 Contextual-Bandit 的新闻推荐算法被提出[26]；协同过滤的思想也被应用到 LinUCB 的算法中[44]，具体来说，在该算法中，系统中用户之间的关联信息通过一个图模型被关联在一起，在这个图中，边上的权重被当作用户之间的相互影响的强度大小，这种算法尽可能地挖掘出用户与用户之间的关联信息以辅助 LinUCB 算法取得较高的性能，这本质上还是第三方信息的引入给算法带来的收益。在[45]中，一种实时动态聚类的方法被提出，这个算法在算法初始化的时候首先将所有的用户进行聚类，在当某一个用户到访系统计算每个推荐物品的期望回报时，都将该用户所属类别中的所有用户的信息利用起来计算期望回报，然后在得到反馈后，实时聚类，更新类簇。这类算法虽然性能取得了提高，但是算法对计算性能的消耗也是庞大的。此外，基于汤普森采样的 Contextual-Bandit 算法也被提出[46]，它的基本假设同样是每个物品的期望回报与用户特征之间呈现出线性关系。考虑时间变化的 Bandit 推荐算法也有相关研究[47,48]。此外，核技术（高斯核）也被应用于期望均值的估计中以期得到更加准确的估计。除了上述算法，在一些基于协同过滤的算法和基于概率匹配的算法，也取得了不错的实验效果[49]。此外，为了减弱 Bandit 算法的表现对超参数的敏感性，一种集成的方案也被提出[50]，用于增强 Bandit 算法的健壮性。

以上的算法基于 LinUCB 的优化主要体现在两个方面；一个是数学模型上的修正，这些算法与 LinUCB 基本的出发点相同，大多认为期望回报与用户的特征之间存在着某种显式的数学关系，通过解方程的方式得到闭式解，然后不断迭代更新。另外一种方式是通过利用第三方的信息来提高推荐的准确率，比如用户的关联信息等。

以上对传统的 LinUCB 算法的修正，虽然性能相对于传统的 LinUCB 算法性能取得了一定的提升，但是大多数都是数学模型上的修正，都是事先假设推荐物品的期望回报和用户的特征之间存在着某种特殊关系，这种关系在很多场景下并不是严格成立的，因此，在很多情况下并不能达到最优的效果。此外，据我们所知，目前尚缺少相关工作来研究针对于用户特征分布的异质性如何智能地选择 Contextual-Bandit 推荐器作为基本组件构造性能更强的推荐模型的相关问题。

现如今深度学习技术已经得到了飞速发展，神经网络有着强大的非线形表征能力，然而现阶段还缺少将神经网络与 Bandit 算法相结合的推荐算法。此外，如何根据现有的用户特征的分布情况，利用现有的推荐算法，设计合理的推荐模型，从而达到最好的推荐效果。本文将针对以上学术界研究较少的领域展开研究。

2.5 相关技术背景

本小节将对本文中所用到的主要技术、基本算法作简要的说明和讲解。

2.5.1 神经网络简介

神经网络是一种由生物学启发而发明的一类算法，它做为机器学习中一类非常重要的算法模型，无论是在分类问题上还是在回归问题上已经取得了巨大的成功且已经被广泛应用于各行各业。结构最简单的神经网络为全连接神经网络，在深度学习领域，一些复杂结构的神经网络，如卷积神经网络（CNN）和循环神经网络（RNN，LSTM，GRU）等由于各自结构的优势，弥补了全连接神经网络在处理空间和时间信息上的不足，被分别用来处理图像和时间序列问题[51,52]。由于本文用户特征是一个由 Logistic 回归拟合出的六维向量，特征向量相对来说比较简单，不包含图像的空间信息、时间序列的时序信息等比较复杂类型的信息，我们拟合的核心目标是训练出用户的多维向量和用户偏好之间的关系，简单的全连接神经网络能够很好地处理此类问题，因此，本文拟采用普通的全连接神经网络作为我们基于神经网络的推荐算法的原型。

一个简单的全连通神经网络的基本结构如图 2-1 所示[53]：

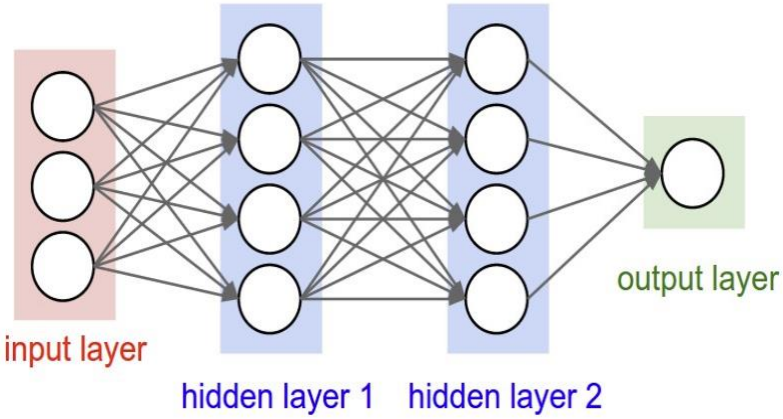


图 2-1 全连接神经网络结构[53]
Figure 2-1 The structure of fully connected neural network[53]

由图可见，基本的全连通神经网络分为三部分，输入层，隐藏层和输出层。输入层作为模型的输入，即多维向量，比如用户的特征。隐藏层作为输入层和输出层之间的层，可以有一层或者多层。每个神经元之间的连接都有其特定的权重且每个神经元上都有固定的偏置项。激活函数的引入给神经网络带来了强大的非线性表征能力，常见的激活函数有两种，一种是 Sigmoid 激活函数，另一种是 Relu 激活函数。其表达式分别如下：

$$f(x) = \frac{1}{1+e^{-x}} \tag{2-8}$$

$$f(x) = \max(0,x) = \begin{cases} 0(x \le 0) \\ x(x > 0) \end{cases} \tag{2-9}$$

其中 x 为输入神经元的值，经过以上两种激活函数，为神经网络引入了非线形表征能力，原则上当神经网络的层数足够多的时候（也就是网络层次足够深的时候），神经网络可以拟合任何的线性和非线性函数，这也是深度学习有着强大的表征能力的原因。但是，当神经网络的层数过多的时候，也有可能造成过拟合问题的产生。

传统的机器学习领域主要解决两类问题：分类和回归。对于分类问题，当特征输入到输入层后，经过隐藏层运算，输出层每个神经元的取值经过 Sigmoid 激活函数被映射为一个（0，1）之间的数，可以被当作样本被分为该类的概率，最后最大的概率的神经元被当作样本分作的类。对于回归问题，输出层每个神经元不会经过 Sigmoid 激活函数，每个神经元直接输出预测值。对于以上两种问题，输出的预测值会根据不同的机器学习任务计算不同的损失函数（loss function），然后通过反向传播算法和梯度下降算法优化神经网络的权重，最后将损失函数降到最小从而实

现较佳的分类和回归效果。

2.5.2 分类问题与回归问题

(1) 分类问题：分类问题是机器学习中的一类经典问题，输出层神经元的个数等于要分类的类别个数。在真实的场景中，对于特定的输入向量 X ，只属于一个特定的类别，也就是真实的标签向量只有一个分量（真实的标签位）的取值为 1，其余的分量的取值均为 0。然而，在神经网络预测的时候，经过 Sigmoid 函数，每个神经元被映射成一个 0 到 1 的数，真正好的预测向量应让正确的类别所对应的维度取值尽可能接近 1，其他维度的分量取值尽可能接近 0。交叉熵损失函数能够很好地度量预测向量与真实标签向量之间的距离，其中，分类问题的交叉熵损失函数定义如下所示：

$$L = \sum_i [z_i \ln y_i + (1 - y_i) \ln(1 - y_i)] \tag{2-10}$$

其中，z_i是输出层神经网络每个神经元经过激活函数后的预测值，y_i 代表着输出层神经网络每个神经元的真实取值。

(2) 回归问题：与分类问题不同，神经网络输出层的输出取值不再被映射到 0 到 1 之间。同理，在回归任务中，同样也需要一个损失函数来度量预测向量与真实向量之间的差距，一般可以用平方差损失函数来计算，其中，平方差损失函数的定义如下所示：

$$L = \sum_i (y_i - z_i)^2 \tag{2-11}$$

其中 z_i 是输出层神经网络每个神经元输出的预测值，y_i代表神经元真实值。

2.5.3 策略梯度简介

增强学习不同于传统的机器学习、深度学习，增强学习通过与环境的不断交互来优化自身的策略来获得最大收益。

在增强学习领域，策略的优化方式主要分为两类。一类是以 Q-learning 为代表的表格型值函数迭代优化算法，这一类算法会迭代产生在每个状态下采取每个动作的得分大小（Q-table），在算法收敛后，智能体在特定状态下会检索 Q-table 继而选取得分最大的动作执行，这类算法适合状态空间不太大的增强学习任务[54]。

另一类比较著名的算法是直接建模策略的算法[54]，该算法能够处理状态空间庞大的增强学习任务。基于策略的增强学习算法直接将智能体的策略参数化，然后

通过梯度下降算法优化策略的参数，从而将策略移动到策略空间中回报高的区域。最为经典的基于策略的增强学习算法为策略梯度（policy-gradient）算法[55]。下面我们将重点介绍策略梯度算法，推导出策略梯度算的损失函数。

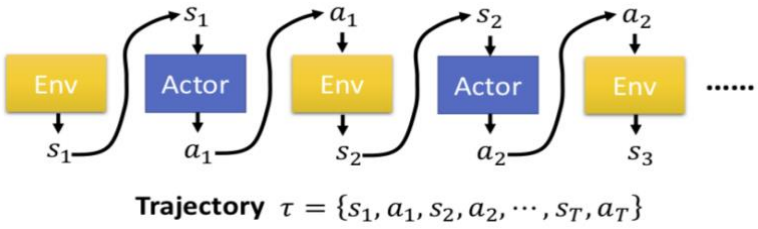


图 2-2 一幕实验示意图[55]

Figure2-2 A diagram for one particular experiment[55]

如图 2-2 所示，Env 代表环境，Actor 代表智能体，智能体不断与环境交互，采取动作获得回报然后进入到下一个状态直到本次回合结束。需要注意的是在传统的增强学习任务下，一个回合中，智能体可能采取了多次动作（即多步），比如在下棋游戏中，一次竞技（回合）中包含多个下子的动作。我们将整个回合内的状态、动作序列定义为 Trajectory，该 Trajectory 描述了整个回合的历史详情。

假设智能体的策略的参数为θ，则产生一个特定的 Trajectory（τ）的概率为：

$$p_\theta(\tau) = p(s_1)p_\theta(a_1|s_1)p_\theta(s_2|s_1,a_1)p_\theta(s_3|s_2,a_2) .... \quad (2\text{-}12)$$

$$p_\theta(\tau) = p(s_1)\prod_{t=1}^{T} p_\theta(a_t|s_t)p_\theta(a_{t+1}|s_t,a_t) \quad (2\text{-}13)$$

如下图所示，智能体在每个状态下采取动作都有可能产生即时回报。

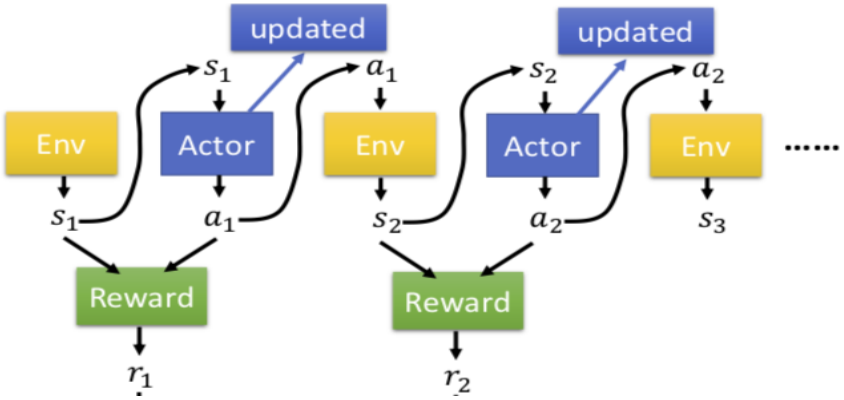


图 2-3 策略梯度示意图[55]

Figure 2-3 The diagram of policy gradient[55]

则一个特定的 Trajecory（τ）产生的总收益为：

$$R(\tau) = \sum_{t=1}^{T} r_t \quad (2\text{-}14)$$

当智能体在策略 Θ 进行了多次实验产生了多个回合后，则该策略在环境下综合所产生的收益的期望为：

$$\bar{R}_\theta = \sum_\tau R(\tau) p_\theta(\tau) \tag{2-15}$$

为了使策略的回报越来越大，则应该使得回报函数的参数向梯度方向移动，即梯度提升。则，策略的梯度为：

$$\nabla \bar{R}_\theta = \sum_\tau R(\tau) \nabla p_\theta(\tau) = \sum_\tau R(\tau) p_\theta(\tau) \frac{\nabla p_\theta(\tau)}{p_\theta(\tau)} = \sum_\tau R(\tau) p_\theta(\tau) \nabla \log p_\theta(\tau) \tag{2-16}$$

$$\nabla \bar{R}_\theta = \frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T_n} R(\tau^n) \nabla \log p_\theta(a_t^n | s_t^n) \tag{2-17}$$

则为了让收益的期望越来越大，应该让期望的参数向着回报函数的梯度方向移动，则有策略 Q 的更新公式如下：

$$\theta \leftarrow \theta + \eta \nabla \bar{R}_\theta \tag{2-18}$$

那么可以等价定义神经网络的损失函数如下：

$$L = -\nabla \bar{R}_\theta = -\frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T_n} R(\tau^n) \nabla \log p_\theta(a_t^n | s_t^n) \tag{2-19}$$

其中，N 代表的是回合的个数，T_n 代表的是每个回合中智能体所采取的动作的次数。经过以上迭代更新，智能体的策略可以不断得到优化，从而得到最大的收益。

2.5.4 t-SNE 简介

t-SNE 是一种比较先进的高维数据可视化技术[56]，t 分布式随机邻域嵌入（t-SNE）是由 Laurens van der Maaten 和 Geoffrey Hinton 开发的可视化机器学习算法[57]。它是一种非线性降维技术，非常适合嵌入高维数据，以便在二维或三维的低维空间中进行数据可视化。具体地，它通过二维或三维点对每个高维对象进行建模，使得在低维空间中建模高维空间的相对关系。

t-SNE 算法包括两个主要阶段。首先，该算法在高维对象对上构造一个概率分布，这样相似的对象被拾取的概率很高，而不同的点被拾取的概率非常小。其次，t-SNE 定义了低维图中点的类似概率分布，并将两个分布之间关于图中点位置的 KL 散度最小化。经过这样处理，我们可以在低维空间内可视化高维空间内的样本分布的相对情况。

2.5.5 开发平台简介

在本文中，我们主要使用 Python 语言进行编程，Python 语言 [58]作为一种解释型语言，其轻量级、简单性的特点已经被广泛应用于数据科学中。本文中采用的机器学习算法主要是用 sklearn[59]机器学习开源库进行实现，sklearn 作为一个著名的机器学习开源库实现了大部分的传统的机器学习算法。此外，神经网络的实现我们主要采用 Tensorflow[60]和 keras[61]机器学习库来搭建相应的神经网络模型。

2.6 本章小结

本章介绍了推荐系统中的 EE 问题以及 Bandit 算法在推荐系统中的应用，详细介绍了一类非常重要的算法：Contextual-Bandit 算法在动态新闻推荐领域的应用，介绍了其基本思想和核心原理。此外，系统介绍了 Contextual-Bandit 算法在学术界的国内外研究现状，指出本文的研究方向。本章还重点介绍了本文中应用到的一些基本知识：神经网络。损失函数，策略梯度等基本算法和模型，作为后面章节的先验知识。

3 新闻推荐系统的测量与观察

本章基于一个著名的门户网站 Yahoo 的新闻推荐日志，测量和观察新闻推荐领域的动态特性：新闻的流行度变化特性和新闻的推荐候选集动态特性；以及用户特征异质性的分布规律。通过观察，我们分析了传统的推荐算法在具有高度动态性的推荐场景，即新闻推荐场景下的不足和引入动态推荐算法的必要性。同时，我们也分析了用户特征分布异质化的情况下传统算法可能存在的问题。

3.1 新闻推荐系统

3.1.1 新闻推荐系统简介

在互联网领域，推荐系统已经被广泛应用。推荐技术的应用主要分为两大部分，一类是电商领域的推荐，电商平台正是根据用户的注册信息、用户的浏览、购买记录来向用户推荐可能感兴趣的商品。另一类应用领域则是视频、图片、文章、新闻等信息分发领域。虽然视频、图片、文章的流行度也会随着时间发生变化，但是这些资源的流行度变化一般都是一个比较缓慢的过程而且资源在推荐系统中留存的生命周期较长。因此，使用文献[12]中的改进方案一般能取得较好的推荐效果。

新闻推荐是一个比较特殊的推荐场景。新闻具有突发性、高时效性的特点。相对于其他类型的互联网资源，新闻具有高的时效性的特点且新闻的生命周期非常短暂，通常来说，一则新闻的生命周期会从几个小时到几天不等。另外，每个不同的新闻其本身的流行度的大小、流行度变化的趋势都不尽相同。另外一个方面，每个新闻的生命周期是不同的，可能某篇新闻的热度经久不衰，而另一篇新闻从上线就没怎么引起用户关注。此外，新的新闻的上线也是具有突发性的，例如，在现实的推荐中，某篇新闻可能是突发新闻，那么该篇新闻则需要加入到推荐候选集作为一篇可以被推荐的候选新闻进行推荐，由此可能会给新闻推荐系统带来冷启动问题。

总的来说，新闻系统作为一个特定的推荐场景，与其他推荐场景的不同的一个显著的特点是新闻的生命周期短，流行度的高度动态性（时效性），且推荐的候选集合有着较强的动态性。

3.1.2 Yahoo 数据集介绍

在本文的研究工作中，所采取的数据集来自于国际知名的搜索引擎网站：Yahoo。Yahoo 是美国著名的互联网门户网站，业务遍及全球 24 个国家和地区，为全球超过五亿的用户提供多元化的网络服务。

本文选取了 Yahoo 门户网站新闻推荐的离线日志数据。本文采用的数据集来自于 Yahoo 网站的“Today”模块的新闻推荐日志。这是一个采用随机推荐策略（random traffic）进行新闻推荐的离线日志数据集，记录了当时该新闻推荐系统的推荐详情。具体来说，包含以下几个字段：

（1）时间戳：Timestamp。记录了用户到访推荐系统的具体时间戳，单位为秒，也即是推荐行为发生的时间戳。

（2）新闻 id：News_id。每一篇新闻都有独属于自己的一个新闻 ID，记录了当前推荐算法具体推荐（暴露）给用户的是哪一篇新闻。

（3）用户特征向量：User_feature。记录了到访的用户的特征向量，是一个六维的向量。其中，最后一个维度是偏置项，为常数 1. 其他维度的数值为用户的历史数据采用 Logisitc 回归（LR）拟合得出的特征向量。因此，真正刻画用户信息的维度是前五维分量。

（4）推荐结果：Result。记录了当前用户对此次推荐是否接受。是一个布尔值，0 和 1。0 用户未接受本次推荐，1 代表用户接受了本次推荐。在真实的系统中，“0”的数量要远大于“1”的数量，代表着“点击”行为是一个比较昂贵的行为，即点击率是一个比较低的指标。

（5）推荐候选集：Candidate_Set。本字段列举了在当前时间段可以被推荐的新闻集合。该字段里的新闻是动态变化的，可能在某一时间段某篇新的新闻被编辑加入到候选集中，也有可能某篇新闻在某一时刻被编辑从候选集中撤销，即下架。

本小节将基于 Yahoo 数据集测量和挖掘该新闻推荐系统中的一般模式和规律，分析可视化推荐系统的动态特性。此外，本文算法的评估阶段也将在此离线数据集上进行评估和进行算法的可视化解释。

3.2 新闻推荐系统的测量与观察

本小节将基于 3.1.2 中介绍的 Yahoo 新闻数据集展开全面细致的测量和可视化分析，分析出新闻推荐系统中的动态特性，以及用户特征的空间分布情况。

3.2.1　新闻流行度的高度动态特性

在本小节中，我们使用 2.1.2 中全天的数据测量了四篇具有代表性的新闻的流行度的变化趋势，绘制了每篇新闻在其生命周期内的流行度变化曲线图，具体如图 3-1 所示。

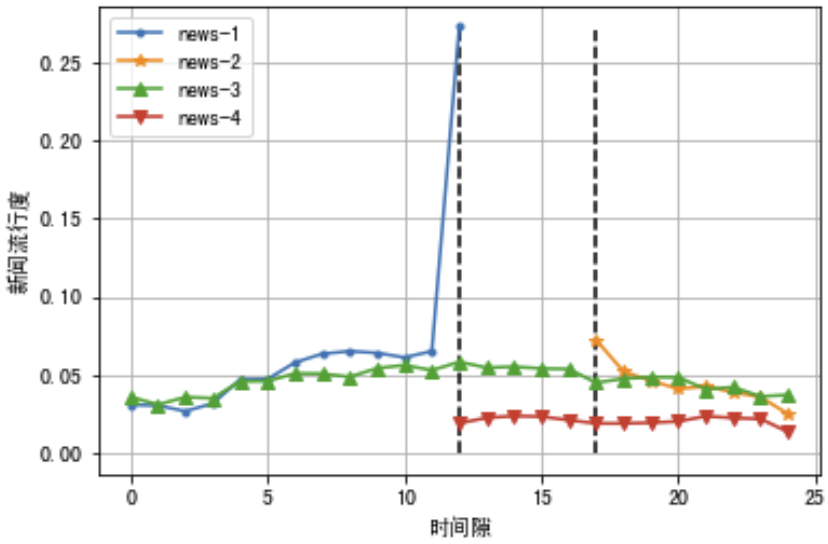


图 3-1　不同新闻流行度曲线
Figure 3-1　The popularity curve of different news

在图 3-1 中，我们统计了每篇新闻在一天内流行度的变化趋势。图中黑色虚线所在位置为推荐候选集发生变化的时刻，在这个时间点，编辑将上线新的新闻或者是将某篇新闻从推荐候选集中移除。横轴的粒度是小时，以上四条曲线代表了经典的四类不同的新闻流行度的变化趋势。首先，每篇新闻的流行度的大小是不同的，而且，这四篇新闻的流行度分别有着不同的变化模式。News1 的新闻流行度一直呈现上升的趋势，在最后该新闻的流行度呈现了一个激增，然而最后却被编辑下架。News2 从被编辑加入到候选集中以来他的流行度就一直呈现出下降的趋势。News3 的流行度一开始呈现出上升的趋势，之后它的流行度一直下降，而 News4 的流行度一直呈现出比较平稳的态势。

因此，我们不难发现，在新闻推荐领域中，待推荐物品（新闻）的流行度是高度变化的，而且流行度变化的模式也不尽相同。

3.2.2　候选集的动态特性

在新闻推荐领域，在 3.2.1 中我们已经详细观察到了新闻流行度的动态变化。另外一个比较重要的动态特性是，新闻的推荐候选集合也是高度动态变化的。在

Yahoo 的新闻推荐场景中，某一篇突发新闻的发生，编辑有可能将该突发新闻加入到推荐候选集中，也有可能在某一个时刻，编辑会将推荐候选集中的某一篇新闻移除。由此造成推荐候选集的动态变化。为了详细说明这一现象，我们详细测量了 Yahoo 的新闻推荐数据集，绘制了该新闻推荐系统内每篇新闻一天内的流行度热力图详情如图 3-2 所示。

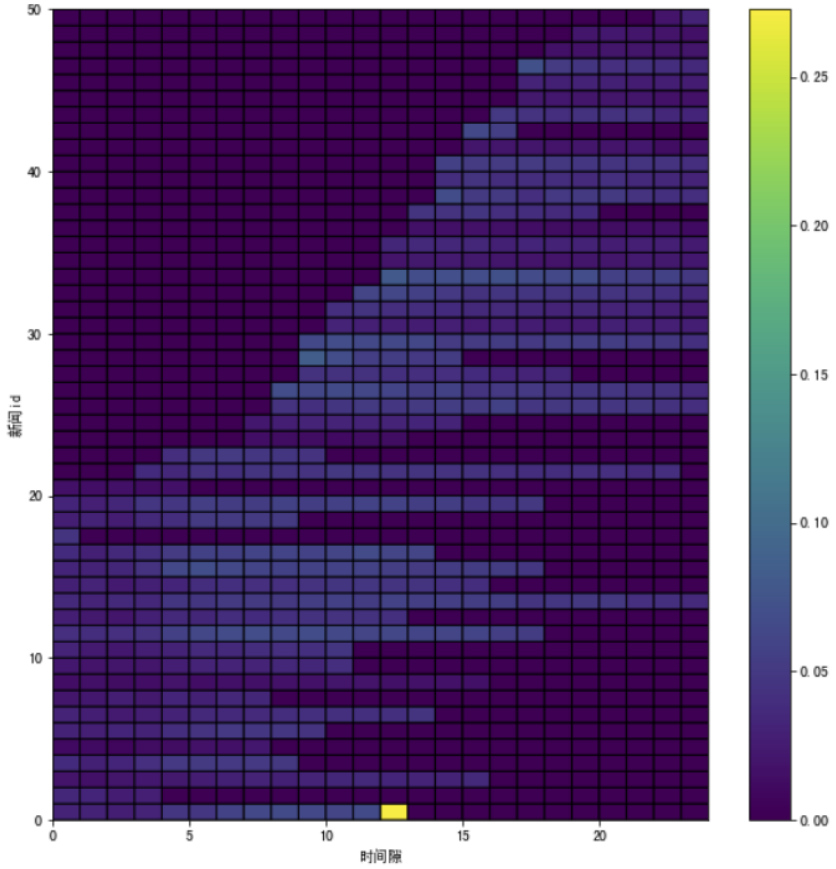


图 3-2 系统详情热力图
Figure 3-2 The thermodynamic chart of system detail

由图 3-2 可见，我们对系统内一天内所出现在推荐候选集中的新闻的点击率（流行度）做了热力图分析。如图所示，横轴代表着时间粒度，单位是小时。纵轴代表新闻 ID，每个小的区域代表着该篇新闻在该小时内的流行度。每一行代表每一篇新闻的流行度变化趋势，每一列代表每个时隙（小时）内的所有新闻的流行度信息。

图中流行度为 0 的区域代表本篇新闻在该时隙内不在推荐候选集合中。比如，对于 news23 来说，他从第五个时隙被编辑加入到推荐候选集合中，又在第十个时隙被编辑从推荐候选集合中下架。我们可以看到，对于特定的某一行（一篇新闻）来说，在他的生命周期内，不同的时隙内新闻的流行度是不同的，这说明了新闻的

流行度是高度的动态变化的，这一点正好和上一小节的观察吻合。从另一个角度看，对于特定的某一列（某一个时隙），流行度不为零的区域是不同的，这说明在同一个时隙内，推荐候选集中的内容是不同的。我们可以观察到从第八个时隙到第十七个时隙几乎每个时隙都有新的新闻被加入到系统的推荐候选集合中。

由图 3-2 可见，新闻的流行度在小时粒度上都呈现出了很大的动态性，每个新闻的生命周期、流行度的变化趋势都是不尽相同的。而且，在不同的时隙内，推荐候选集中的内容也是高度动态，整个过程内有频繁的新闻的上下架操作。

3.2.3　推荐系统中用户兴趣的异质性

在该系统中，我们分析了该推荐系统内用户的特征分布。由 3.1.2 可知，用户的特征是由历史数据经过 LR（logistic 回归）拟合出的用户的特征，每个用户的特征是包含六个浮点数的一个高维向量，其中，第六个分量为偏置项，每个用户特征的第六个分量均为常数 1。于是，我们对每个用户的前五维分量做了降维和可视化分析。

由于用户的数量过于庞大，我们挑选出了 1000 个不同的用户特征，利用 t-SNE 将五维的特征降为二维，然后在二维平面上做了可视化。为了保证无偏性，这一千名用户的挑选方式是随机的，用户的特征在空间内的分布的一个示例如图 3-3 所示：

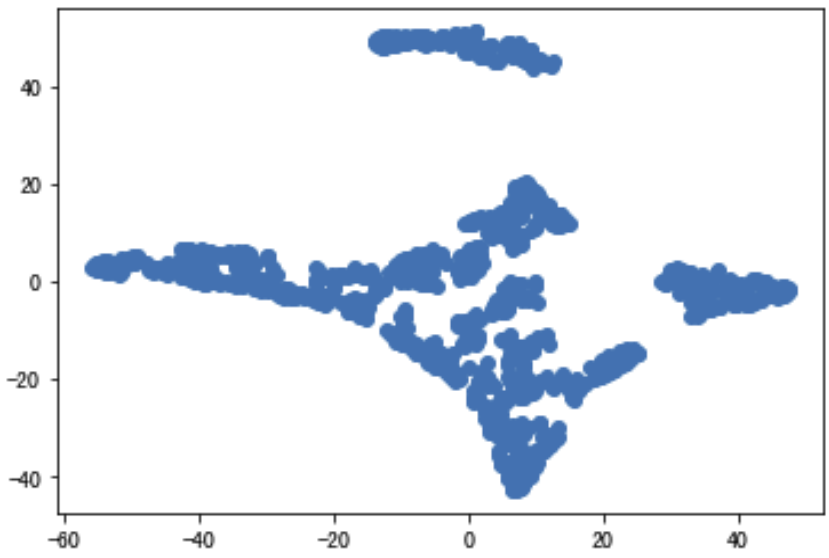


图 3-3 用户特征分布详情
Figure 3-3　The detail of user feature distribution

由图 3-3 所示，用户的特征在二维空间内分布极不均匀，即用户群体可以分为

几个类簇。用户的特征在空间内分布同时也是比较发散的，并没有集中分布在空间中的某一个区域。

在这种用户群体的分布下，由于用户的特征在空间内分布不均匀，在传统的Contextual-Bandit 推荐算法中，推荐器只有一个不能取得好的效果。在这种情况下，每篇新闻的参数向量（偏好向量）是所有的用户向量共同训练的结果，由于用户的兴趣（类别）在空间内分布不均匀，因此使用一个模型适用于所有的用户可能取不到最好的效果。

3.3 流行度和候选集的动态性给传统推荐算法的挑战

在上一小节中，我们系统分析了新闻推荐系统的动态特性，主要包含两个方面：1. 新闻流行度的动态变化特性；2. 推荐候选集合的动态变换特性。本小节将全面分析这两方面的动态特性给传统的推荐算法带来的问题和挑战，并指出动态新闻推荐算法的必要性。

(1) 新闻流行度的动态性带来的问题。在推荐系统中，推荐物品的流行度的变化是非常正常的，在非新闻领域的推荐场景中，待推荐物品的流行度也是动态变化的，但是一般来说变化速度比较缓慢，呈现天级别、甚至季节级别的变化特性。在这种推荐场景中，时间敏感的推荐算法往往可以取得不错的推荐效果。然而，在新闻推荐领域，由于新闻流行度的高度动态性，每篇新闻的流行度往往会在几个小时内发生巨大的变化，时间敏感的相关算法很难处理这种场景。此外，传统的推荐算法，如协同过滤、矩阵分解等算法并不是在线更新的，它们把推荐系统当作一个静态的过程，这显然是不合理的。举例来说，在某个时刻，协同过滤算法通过用户的历史行为数据可能得出 A 新闻和 B 新闻非常接近的结论，当用户点击了 A 新闻后，很有可能再向用户推荐 B 新闻。然而由于新闻流行度的高度动态性，在当前时刻，很有可能 B 新闻的流行度已经非常低（即该新闻已经过时），用户点击的可能性也是非常小的。因此，在这种流行度有个高度动态性的推荐场景中，一个能够实时感应物品的流行度变化并根据流行度变化动态调整推荐优先级的推荐算法是非常必须的。

(2) 推荐候选集合动态变化带来的问题。由于新闻的突发性，新闻加入到推荐候选集合的模式是无规律的。这就造成了新闻推荐领域候选集的高度动态性，在这种情况下，当一篇新的新闻被加入到推荐候选集的时候，由于系统中还没有用户对这篇新闻的行为数据，基于行为的推荐算法在当新的新闻收集到充分多的行为数据之前无法得到较高的推荐准确度，这就是著名的推荐系统的“冷启动”问题，由于在新闻推荐领域，新的新闻的加入是频繁的且无规律的，因此这给推荐系统带来

的冷启动问题就更为明显。因此，需要一个算法能够应对新物品（新闻）的冷启动问题。

3.4　用户特征异质性带来的挑战和问题

在上一小节中，我们对用户的特征做了降维和可视化分析，我们发现，在空间内用户的特征分布是比较发散的。传统的 Contextual-Bandit 推荐算法，如 LinUCB，对于所有的用户共用一个推荐器对象（兴趣模型），每篇文章的参数向量是拟合所有类型的用户得到的，由于用户的特征分布并不均匀和集中，所以，采用一个推荐器进行学习可能取得不到最好的效果。

以上的问题可用如下示例说明：假设在线性回归问题中，样本分布同样不均匀，在二维空间内呈现出二类分布的情况。倘若我们对所有的样本点进行线性回归分析，则拟合出的直线如图 3-4 所示。显然在这种情况下，样本点的所有的预测值都由这一条直线给出，在这种情况下，所有的样本点所造成的预测误差无疑来说是非常大的。这是由于样本点在平面上分布不均匀，有着明显的类簇分布（两类样本点的输入特征 X 互不重合），只使用一条曲线（一个模型，一套参数）无法细致刻画出不同类簇上的样本点输入特征（X）和预测值（Y）之间的关系，最终的拟合结果只能是平面内所有样本折中的结果，因此效果不佳。

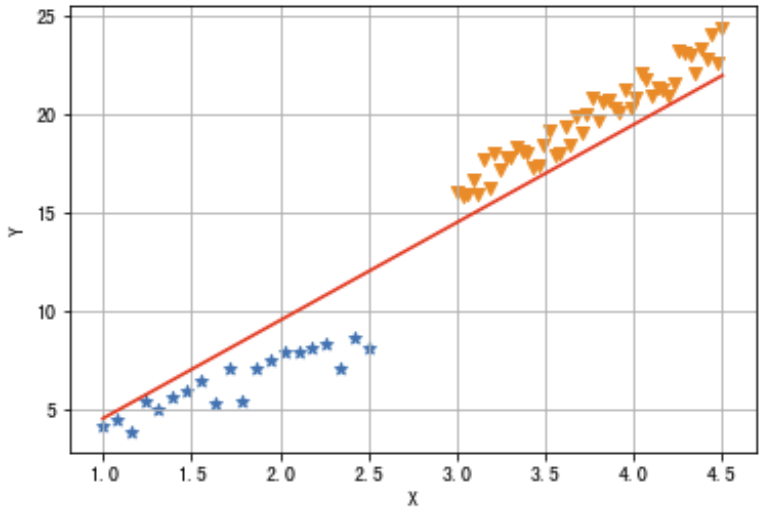


图 3-4　单模型线性回归示意图
Figure 3-4　Singe model for linear regression

3.5 本章小结

本章重点对新闻推荐进行了测量和观察，研究了新闻推荐领域的两方面的动态特性：新闻流行度的高度动态特性和推荐候选集的高度动态特性。通过测量分析，可视化了这两种动态特性并且分析了这两种特性可能给推荐系统带来的问题，说明了传统的推荐算法在该场景中的不足和引入动态新闻推荐算法的必要性。

此外，本章也从用户的维度分析了用户的特征在空间内的分布，使用先进的降维技术 t-SNE 将用户的高维向量由五维降到了二维，然后在二维平面上做了可视化分析。发现用户的特征在空间内的分布并不是均匀的，我们同样分析了该用户特征分布可能给动态推荐算法带来的问题和影响，该观察也将作为我们后面分级算法的一个基本切入点。

4　基于神经网络的动态新闻推荐算法

LinUCB 算法使用线性模型建模新闻推荐得分均值与用户特征之间的关系，表征能力有限，在本章，我们提出基于神经元网络的动态推荐算法，将神经元网络引入到 Contextual-Bandit 动态推荐框架中，解决神经元网络更新和损失函数选择的两个难题，利用神经元网络的表征能力，改进动态推荐算法的性能，最后给出算法的性能评估和结果分析。

4.1　基于神经网络的动态新闻推荐算法

在本小节中，我们拟将经典的神经网络与经典的 Bandit 的解决方案相结合实现推荐的探索机制，并给出具体的算法实现的伪代码。

4.1.1　基本想法

本文提出使用神经网络代替 LinUCB 的线性模型来拟合用户特征和新闻回报之间的关系。目前，首创将深度神经元网络引入到 Contextual-Bandit 框架中的一个主要工作是使用神经网络拟合汤普森采样的后验分布[62]，与该研究不同的是，我们引入神经网络来代替 LinUCB 中的线性模型来拟合用户和新闻回报之间的关系。在 LinUCB 算法中，一个基本的假设是每篇新闻的期望回报与当前到访系统的用户的特征之间呈现的是一种简单的线性相关的关系。而在现实的场景中，新闻的期望回报与用户的特征之间不一定呈现线性关系，有可能呈现某种复杂的关系。事先给用户的特征和期望回报之间设定某种特定的关系往往是不合理的。得益于神经网络的强大的表征能力，我们提出使用神经网络建模新闻的期望回报与用户的特征向量之间的关系，以期获得更加准确的推荐结果。

在该场景中中引入神经元网络主要面临着两个问题：1. 神经元网络损失函数的选取与设计；2. 如何高效地在线训练神经网络。针对于第一个问题我们将推荐问题抽象为回归、分类、策略梯度三个问题进行建模，针对于第二个问题，我们提出用户反馈敏感的在线训练方式，根据用户不同的反馈 (接受/不接受) 采用不同的训练次数的方式。

4.1.2 损失函数选取与设计的问题

在 Contextual-Bandit 场景中使用神经网络主要面临着两个难点：神经网络损失函数的设计，选取以及神经网络的在线训练。本小节将分别对与这两个问题给出解决方案。

(1) 神经网络损失函数设计与选取

神经网络建模需要解决的一个首要问题是神经网络损失函数的设计，在本小节，我们将新闻推荐具体抽象为三类问题，给出使用三种不同的损失函数所对应的实际意义，给出并详述神经网络所发挥的具体作用。

1) 回归问题。当我们使用平方差损失函数的时候，该问题可以转化为回归问题。当每个用户到访的时候用户的特征输入到神经网络，对于输出层每个特定的神经元，都将输出一个在当前用户特征下的一个值。我们可以将每个神经元的值理解成为在当前用户特征下的推荐得分。

2) 分类问题。在我们使用交叉熵损失函数的时候，该问题可以转化为分类问题。当每个用户到访的时候用户的特征输入到神经网络，对于输出层每个特定的神经元，其值被压缩到（0，1）之间，可以被理解为概率值。每个神经元经过激活函数的取值可以理解为当前用户特征归属于这一类的置信度。

3) 策略梯度问题。在使用策略梯度损失函数的时候，该问题转化为增强学习中的一个比较简单的问题。神经网络相当于增强学习中的智能体，用户的特征相当于智能体所处的状态，输出层每个神经元的输出同样是一个经过了 Sigmoid 激活函数压缩的一个在（0，1）之间的数，可以理解为在当前状态下智能体应该采取的动作的概率。值得注意的一点是，在我们的任务中，不同于传统的增强学习：智能体（神经网络）所处的前后状态之间并没有相关性，也就是在实际的系统中，前一个时刻到访的用户是谁与后一个时刻到访的用户是谁是相互独立的。也就是说在本问题中，每个回合智能体只采取一次动作，即只产生一次推荐行为，推荐行为完毕后，智能体接受到用户的反馈后（接受/不接受），则本回合结束，当下一个用户到访系统的时候，下一个新的回合开始，如此往复。因此，在策略梯度的损失函数中，N 和 T_n的取值都为 1.则，神经网络的损失函数退化为：

$$-r_t \nabla \log(p_\theta(a_i|X_t)) \tag{4-1}$$

无论采用哪种损失函数，神经网络的输出层神经元的取值越大的文章其推荐的优先级应该越高。我们将每个神经元的取值定义为其编码的新闻的的效用（utility）。显然，对于贪心策略来说，我们应该每次推荐效用最高的新闻，然而我们在第二章中已经知道探索是必须的，因此在接下来将结合经典的 Bandit 问题的解决方案实现探索机制。

(2) 损失函数的修正

当模型推荐出某一篇新闻后，被推荐的新闻将暴露给当前到访系统的用户。此时，用户会给出反馈，接受本次智能体做出的推荐还是拒绝本次推荐（0/1）。

由于我们的算法是一个在线（on-line）的算法，因此，每当一个特定的用户给出反馈时，神经网络都要根据用户的反馈和该用户的特征更新自身的参数。因此，一个三元组，即（用户特征，推荐的新闻，推荐结果）将被用来训练神经网络来优化相应的损失函数。

与普通的分类和回归问题不同，在传统的分类和回归问题中，当我们使用交叉熵损失函数或者使用平方差损失函数的时候，在输出层计算损失函数的时候，每个神经元的取值与真实的取值的差异都会被计算。然而在我们现在的场景中，原生的这种损失函数的定义是不合理的，具体原因与解决方案如下：在我们的推荐场景中，无论采用交叉熵损失函数还是平方差损失函数，我们只有一个神经元的真实标签。即，每次暴露给用户的新闻只有一篇，当用户给出反馈后，我们只能得到被曝光的这篇新闻的真实标签。而其他剩余的未被曝光的新闻，它们的真实标签是未知的。也就是，我们不能确定如果推荐其他的新闻，该用户是否会接受推荐。所以，在这种特定的场景下，我们只将被推荐的那个新闻所编码的神经元置为用户的真实反馈（0 或 1），其他的未被暴露的神经元保持原来的预测值不变，这样，相当于对损失函数做了修正。从而实现上述功能。对于第三种策略梯度的损失函数: $-r_t \nabla \log(p_\theta(a_i|X_t))$，我们可以直接以最终推荐的神经元为节点进行反向传播优化算法。

4.1.3 神经网络在线训练的问题

与传统的机器学习任务不同，在我们的问题中神经网络是在线更新的。在离线训练的机器学习任务中，在样本不均衡的时候，神经网络的训练效果往往会受到很大影响。在推荐场景中，点击行为是一件很昂贵的行为，也就是说不点击的行为的数量是远远大于点击行为的数量的。这就造成了在在线训练神经网络的过程中也产生了很大的样本不均衡的问题。为了解决这个问题，使神经网络充分发挥其能力，我们采用了一种“回报敏感” (reward-aware) 的在线训练算法。当我们收到了一个正反馈 (用户接受了本次推荐：1) 的时候，用这个正向的三元组 (用户特征，推荐的新闻，推荐结果) 去运行反向传播次数 P 次，当收到负反馈 (用户未接受本次推荐：0) 的时候，运行反向传播次数 N 次。其中 P>N，在不同的推荐场景下 P 和 N 可以通过实验确定从而获得最佳性能。

4.1.4 过滤模块与探索机制的引入

(1) 过滤模块的引入

在我们的算法中，每个新闻都被编码成神经网络输出层的一个神经元。然而，在新闻推荐的场景中，一个比较特殊的问题是，新闻的候选集存在上下架的问题。也就是说，输出层的可用神经元在不同的时刻可能是不同的。这取决于编辑的上下架行为。因此，在本问题中，我们在神经网络的输出层后面加入了一个过滤模块，该过滤模块实时检测当前网络的输出层哪些神经元是可用的，即筛选出那些在当前时刻处于推荐候选集的文章，此时，推荐候选集中的每个新闻已经在特定的损失函数下计算出了它的效用得分。在此后，可以根据每个新闻的效用得分，利用探索机制最终实现新闻推荐。

(2) 探索机制的引入

第三章已经明确地指出了推荐系统中的 EE 问题，并且说明了在推荐系统中，探索是必须的。如果推荐的策略是贪心的，那么在神经网络的输出层，我们将每次选择效用 (utility) 得分最高的新闻进行推荐。这显然在很多情况下不是一个明确的选择，因此，虽然神经网络计算出了每篇新闻的效用得分，当我们经过了过滤模块筛选出当前可用的推荐候选集合后，还需要根据效用得分引入探索机制，这样才能解决推荐系统的 EE 问题。

在 LinUCB 等基于数学模型的迭代算法中，方差的上限通过闭式解的方式给出，然后可以通过经典的 UCB 算法实现探索和利用的平衡。在经过神经网络后，虽然神经网络计算出了每篇新闻的效用得分，但是，每个效用得分的方差的上限无法通过闭式解的方式给出，因此，再集成 UCB 策略是行不通的。

最简单的实现探索机制的算法是$\varepsilon-$greedy算法，但是该算法是启发式的算法，性能表现通常不是最佳的。因此，在此处我们使用 Bandit 问题中的 Softmax 探索策略，我们将每个神经元的效用得分输入进 Softmax 算法，经过公式 (2-2) 和公式 (2-3) 的计算，从而得出每个每篇新闻的推荐概率。最后按照概率向量推荐某一篇新闻给当前到访系统的用户。

4.1.5 算法框架与实现

经过上述介绍，本小节将给出整个推荐流程的示意图，并给出算法实现的伪代码。

(1) 推荐流程

如图所示，我们将给出一个算法流程的例子。在 t 时刻，当用户到访系统的时

候，用户的特征被输入到神经网络的输入层，经过若干隐藏层最后在输出层计算出每个神经元（新闻）的效用得分。经过过滤模块，我们筛选掉了 N4，N5 两篇新闻(这两篇新闻已经下架)，接下来将可用的新闻的效用得分送入 Softmax 算法，最终得到推荐给用户的新闻。由图可见，若使用贪心策略，在最后可被推荐的新闻中，应该推荐 N2 (效用得分最高)，然而由于探索机制的存在，最终展现给用户的是 N6。

当将新闻展现给用户后，用户会给出即时反馈。比如用户接受了本次推荐。则对于推荐问题，算法将被推荐的新闻所对应的神经元的值置为 1，其他未推荐给用户的新闻所编码的神经元保持原来的预测值不变。通过这种方式，我们实现了损失函数的修正，将修正后的预测向量作为真实的标签向量，然后和当前到访该系统的用户的特征向量去训练神经网络，使用“回报敏感”的方式去训练神经网络，完成一次模型的更新。

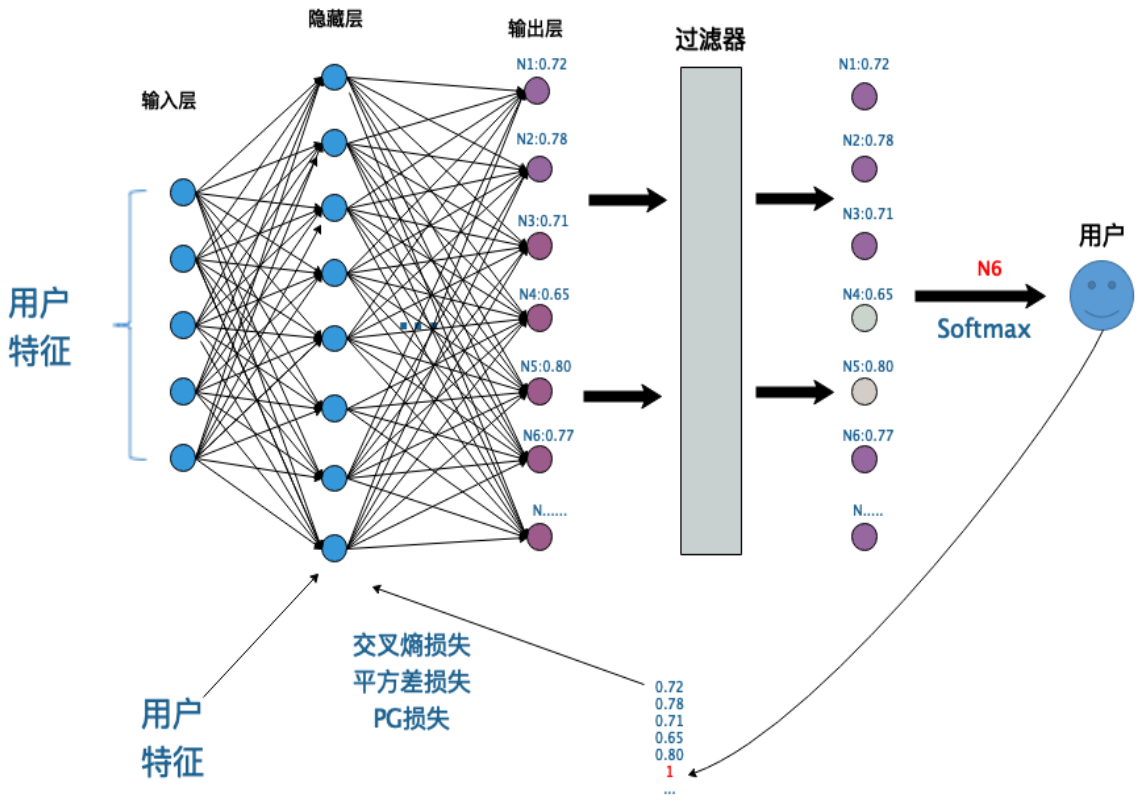


图 4-1 基于神经网络的动态推荐算法结构图

Figure 4-1 The structure of NN based dynamic recommendation algorithm

(2) 算法伪代码

经过以上算法的示例说明，我们在本小节给出基于神经网络的动态新闻推荐算法的伪代码，如算法 4-1 所示：

算法 4-1　基于神经网络的上下文老虎机算法
Algorithm4-1　Neural-network based Contextual bandit method

Algorithm 4-1:　Neural-network based Contextual bandit methods

0: τ: The temperature in **Softmax** to control the degree of exploration
d: The dimension of the user feature.
K: The number of actions
1: Initialize neural network $\Pi(\Theta)$ with input-dimension d, output dimension K
2: **for** t= 1,2,3….T, **do**
3:　Observe the current user feature $X_t \in R^d$
4:　Feed X_t into $\Pi(\Theta)$ and generate the utility vector $U_k = [u_1, u_2, \dots u_k]$
5:　$U_k = [u_1 - \min(U_k), u_2 - \min(U_k), \dots u_k - \min(U_k)]$
6:　Filter the available actions
7:　**for** each action a_k，do
8:　　Generate the probability $p_t(a_k) = \frac{\exp(\frac{\mu_k}{\tau})}{\sum_{k=1}^{K} \exp(\frac{\mu_k}{\tau})}$
9:　**end for**
10:　Recommend some news a_i according to its $p_t(a_i)$ and observe feedback
11:　Generate a particular modified *loss* function
12:　Carry on the back-propagation operation to optimize policy with $r_{t,a}$
13: **end for**

该算法首先初始化了神经网络的结构 (0，1 行)，然后利用该神经网络在线训练，在每个时刻，对于到访的用户，神经网络观察用户的特征向量，并将用户的特征向量输入进神经网络 (3，4 行)，在第四行我们抽取出神经网络输出层每个神经元的取值，再经过第五行的过滤器得到当前在系统中可以被推荐的新闻以及它们各自的得分 (神经网络输出神经元取值大小)，第七行我们使用 Softmax 探索机制为每篇新闻计算了它们的推荐概率，然后第九行向当前用户进行推荐并观察用户反馈，第十行利用用户反馈生成特定的损失函数然后第十一行完成神经网络的训练。

4.2　评估方式简介

4.2.1　在线评估算法

本文中的推荐算法是一个在线的，动态的推荐算法，推荐算法不断通过与用户的交互来进行自身策略的优化，因此，与传统的机器学习、深度学习的验证方式不同，在离线数据集上划分训练集和测试集的验证方式不再适用，需使用在线方式。

我们采用文献[19]中的在线评估的方式来进行我们算法评估，文献[19]中的在线评估方式已经被证明是无偏的且已经被广泛应用于在线动态新闻推荐算法的性能评估上。我们在 Yahoo 数据集上验证我们的算法模型，主要衡量推荐的点击率 (CTR) 这一指标。其中，在线验证算法如下所示：

算法 4-2　策略评估算法[19]
Algorithm 4-2　Policy_Evaluator[19]

Algorithm 4-2:　Policy_Evaluator

0: Inputs:　$T > 0$, policy Π, stream of events
1:　$h_0 \leftarrow \Phi$　{An initially empty history}
2:　$R_0 \leftarrow \Phi$　{An initially zero total payoff}
3: **for**　t =1,2,3,….,T **do**
4:　　**repeat**
5:　　Get next event $(x_1, \dots x_K, a, r_a)$
6:　　**until**　$\Pi(h_{t-1}, (x_1, \dots x_K)) = a$
7:　　$h_t \leftarrow CONCATENATE(h_{t-1}, (X_1, \dots X_K, a, r_a))$
8:　　$R_t \leftarrow R_{t-1} + r_a$
9: **end for**
10: Output:　R_T/T

该算法的一个基本思想是：在某一固定时刻，我们模型推荐的新闻可能不是离线数据记录的当时推荐给用户的那篇新闻。比如，在 t1 时刻，离线数据中记录的当时推荐给用户的新闻是 N1，而当前算法如果推荐给当前用户的新闻不是 N1 的话，我们无从得知推荐的这条非 N1 的新闻用户是否接受。因此这条推荐将不是一次有效的推荐。在这种情况下，本次推荐行为不记录到最终的性能指标的计算中去，我们的算法模型也不会被更新。这种验证在线算法的方式已经被证明是无偏的[19]。

4.2.2　具体评估方案

在上一小节中，我们已详细介绍了在离线数据上评估在线算法的方法，但是有这样一个现象存在：当算法模型初始化的时候，产生第一次推荐的时候，由于每篇新闻还未与用户进行过交互，所以每篇新闻的得分是一样的，因此，系统在第一次

推荐产生的新闻推荐应该是随机的。该问题可以类比为如下问题：给定一定的摇老虎机的次数，最后的整体收益与第一次摇的是哪个老虎机是有一定关系的，因此最后的整体收益会根据第一次摇的是哪个老虎机而产生细微变化。这是因为，本文采用的算法是在线的、动态的、实时更新的，策略在 t 时刻之前的行为会对 t 时刻之后的推荐结果产生一定的影响。

为了获得更高的评价准确度，在同一个算法相同的参数配置下，我们将随机以不同的新闻作为第一次推荐的选择，做多次实验，最后取多次实验的均值，作为该算法在该参数配置下的推荐性能。此外，在基本的 LinUCB 算法中，在 Yahoo 数据集上，当探索系数 α 为 0.2 的时候算法能够取得最佳性能[19]。因此，我们将以 LinUCB 在该超参数下的性能作为基本参照性能。

4.3 基于神经网络的新闻推荐算法评估

4.3.1 性能表现详情

在本小节，我们在表 4-1 中报告了基于神经网络的推荐算法在不同损失函数下的性能表现。其中τ是 **Softmax** 探索算法的超参数。

表 4-1 基于神经网络的动态推荐算法性能表现
Table 4-1 The performance of NN based dynamic recommendation algorithm

算法	τ	CTR
随机推荐	--	3.98%
LinUCB	--	6.53%
cross-entropy	0	6.48%
	0.001	6.56%
	0.005	6.52%
RMSE	0	6.44%
	0.001	6.46%
	0.005	6.50%
PG-loss	0	6.66%
	0.001	**6.67%**
	0.005	6.56%

由表 4-1 可见，当采用 PG-loss 时，探索力度为 τ=0.005 的时候，算法的性能

能够取得最佳，比 Lin-UCB 的最佳性能提高了 (6.67-6.53)/6.53=2.1%。接下来，我们将可视化该配置下的模型的推荐详情，然后分析算法的收益所在。

对于以上的神经网络，我们采取了“回报敏感”的在线训练方式得到的结果。正反馈的最佳训练次数和负反馈的最佳训练次数通过网格搜索的方式获得。同时，我们也使用传统在线训练方式评估了模型的性能，模型在这种在线训练方式下，基于神经网络的推荐方法相对于随机推荐方法只取得了十分微弱的增益，平均 CTR 约为 4.1%，这意味着我们的“回报敏感”的在线训练方式最大可获得约 40%的性能收益，这充分证明了我们提出的在线训练方式的有效性。

4.3.2　推荐详情分析

在本小节，我们将选取基于策略梯度损失函数的的推荐模型来可视化推荐详情，分析该策略在每个时隙内对每篇新闻的推荐详情以及推荐准确率，讨论算法的收益来源和模型的表现。

由于该新闻推荐算法是一个在线的动态算法，图 4-1 给出了整个推荐过程的平均 CTR 变化曲线，我们将在接下来展示模型在每个时隙内（半个小时）的表现详情。我们计算了 LinUCB 和 PG-loss 相对于随机推荐策略的相对 CTR，以半个小时为粒度，绘制了其相对 CTR 在一天内的变化趋势，如下图所示：

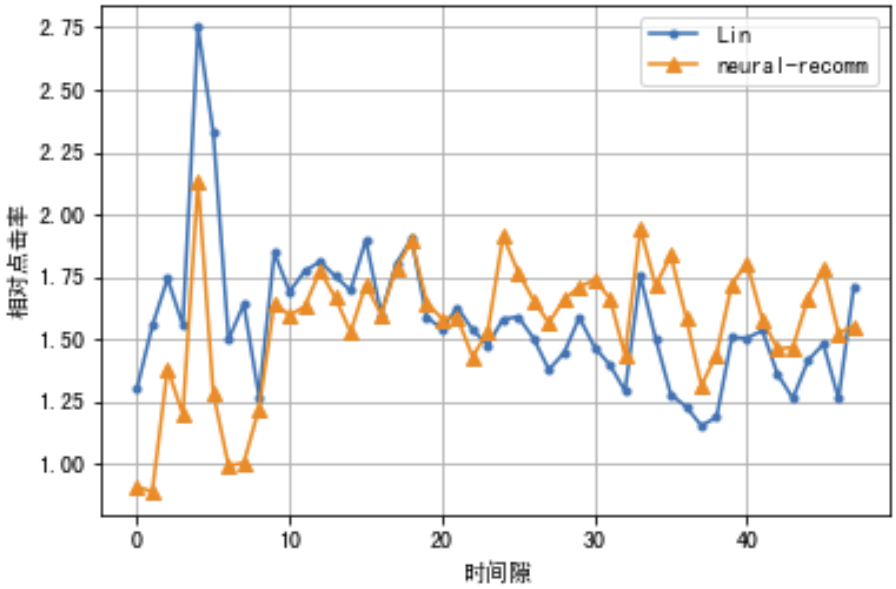
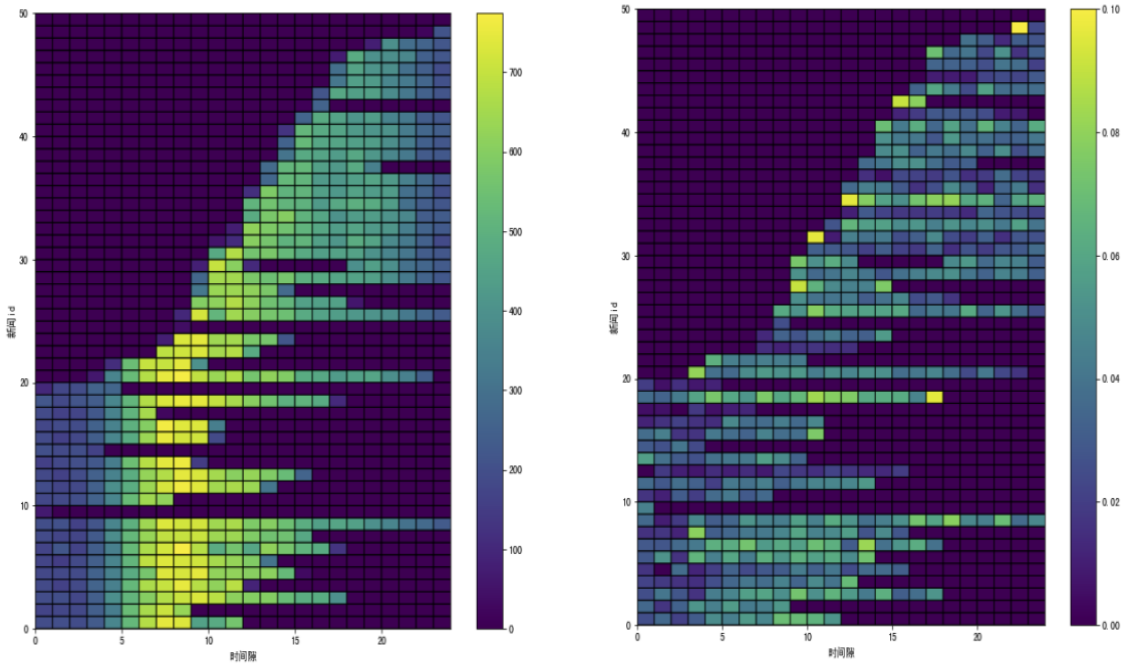


图 4-2 神经网络算法 CTR 变化对比图
Figure 4-2　The CTR contrast curve of NN based algorithm

由图 4-2 可见，两个模型的 CTR 曲线保持着相近的变化趋势且存在着一定的抖动性，CTR 曲线的抖动主要是由两方面的因素造成的: 1. 编辑对于推荐候选集

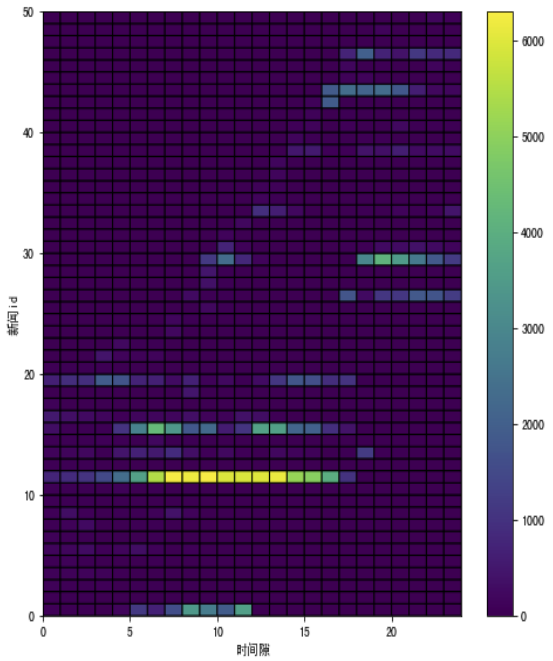
的上下架的操作会给推荐系统带来冷启动问题，这就造成在加入新新闻的一段时间内推荐系统对新加入的新新闻的推荐准确度不高，这是不可避免的，从而在一定程度上影响整体的模型推荐准确度，造成 CTR 曲线的抖动；2. 人们对新闻的点击率在一天内本来就呈现出一定的不稳定性，往往会因为处于不同的时间段而发生变化。比如：我们对该系统的离线日志进行测量分析发现：白天的点击率可能要高于晚上的点击率。另外，我们也可以发现在前 22 个时隙内，LinUCB 模型的相对 CTR 曲线几乎一直保持在神经网络模型的上方，在 22 到最后的时间范围内，神经网络模型的 CTR 曲线实现反超，一直处于 LinUCB 模型的上方。这是由于：神经网络模型相较于线性模型来说比较复杂，在训练的过程中需要消耗更多的样本点，也就需要较长的时间才能收敛才能达到较好的算法性能。这种现象也是不可避免的，因此在前 22 个时隙内的性能表现略低于 LinUCB 这种简单的线性模型。但是，当神经网络经过足够多的训练次数达到收敛以后，其拟合能力要远比 LinUCB 这种线性模型强，因此能够精准地根据用户的特征来预测用户的偏好，从而达到较高的推荐准确率。正是由于这种原因，所以在后面的时间间隔内神经网络模型的推荐准确率要远高于 LinUCB 算法的推荐准确率，相对 CTR 也就相应地高于 LinUCB 模型的相对 CTR。

此外，为了更加全面地分析每种策略的推荐详情以及其工作机制，我们可视化了随机推荐策略、LinUCB 策略以及基于神经网络的推荐策略的推荐详情如图 4-3 所示。

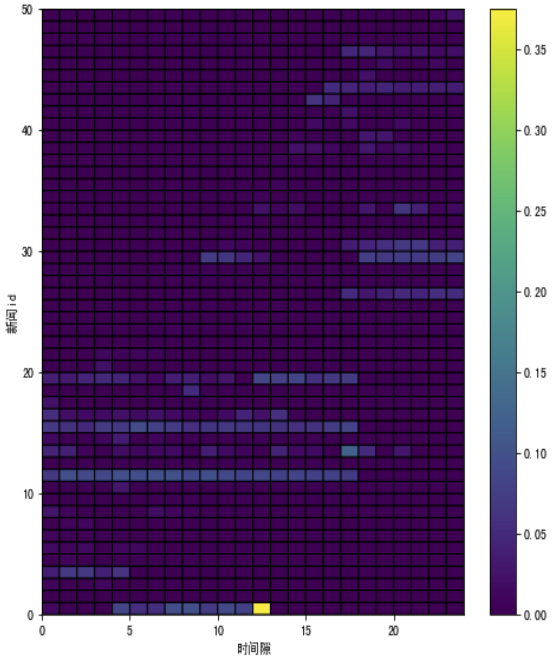


a　随机策略推荐详情　　b　随机策略推荐 CTR

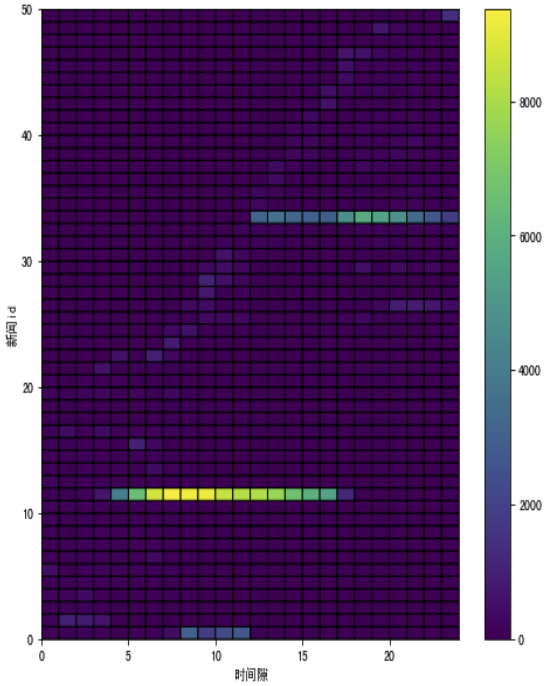
a　The recommendation detail of random policy　　b　The CTR of random policy



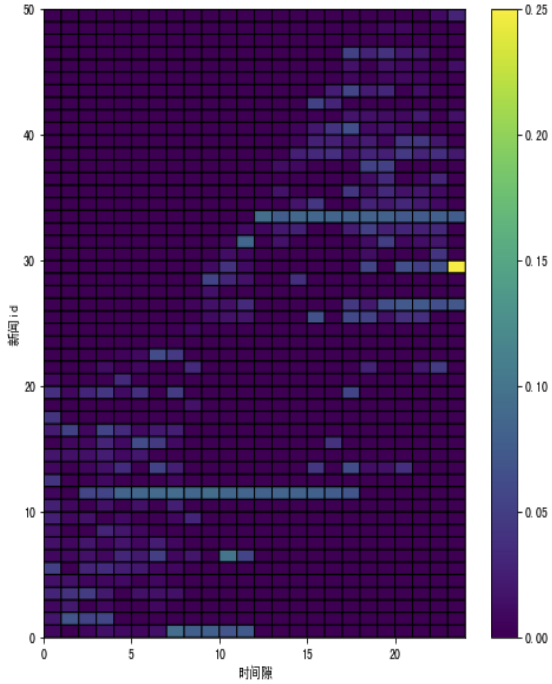
c　LinUCB 推荐详情
c　The recommendation of LinUCB policy



d　LinUCB 推荐策略 CTR
d　The CTR of LinUCB policy



e　神经网络推荐详情
e　The recommendation detail of NN policy



f　神经网络推荐 CTR
f　The CTR of NN policy

图 4-3　不同策略推荐详情
Figure 4-3　The recommendation detail for different policies

由图 4-3 可见，对于每一个子图，横轴索引代表着时隙 (小时粒度)，纵轴索引

代表着新闻 id，在图 a 中，推荐数量为 0 的区域代表着在当前时隙内，该新闻不在推荐候选集内（还未被加入到推荐候选集或者已经被编辑移出推荐候选集），其他子图的对应位置也表示该新闻不在推荐候选集中。

对比图 a，图 c 和图 e 可见，对于随机推荐策略没有任何个性化，因此对每篇新闻都推荐的非常均匀，并没有个性化或者挑选出来非常热门的新闻，其推荐的 CTR 也都相对较低。对比图 c 和图 e，我们会发现无论是 LiuUCB 算法还是神经网络模型，其推荐的范围比较集中，在整个过程中重点推荐几篇特定的新闻。LinUCB 模型和神经网络模型都发现了一篇非常热门的新闻并重点推荐：新闻 12。我们也可以发现，在每篇新闻加入到候选集的时候，比如（15，37）等位置，LinUCB 模型和神经网络模型都会尝试对新上架的新闻进行推荐以完成新上架新闻的冷启动。此外，无论 LinUCB 还是神经网络模型都可以根据的新闻的准确率来动态调整对这篇新闻的推荐数量（优先级），这是因为 LinUCB 模型和神经网络模型都是在线的、动态的模型，可以根据当前推荐的结果来动态调整策略。

对比 LinUCB 算法的推荐详情和神经网络的推荐详情，我们可以看到，两者在推荐次数较多的新闻上都能取得不错的推荐准确率。但是，相较于 LinUCB 模型，在推荐次数较少的新闻上，神经网络模型依然能够给出较高的推荐准确率。对比图 d 和图 f，在图 e 比图 c 更集中的情况下，图 f 比图 d 更密集，说明了神经网络模型的推荐更加个性化，能够精准地捕获用户的特征作出精准推荐。

4.4　本章小结

为了克服传统的以 LinUCB 为代表的 Contextual-Bandit 的数学上的强假设问题，提高模型的非线性表征能力，本章设计了基于神经网络的动态新闻推荐算法。该算法接收用户的特征作为输入，然后对于每篇新闻给出推荐的得分，最后算法集成了 Softmax 探索策略解决推荐系统中的 EE 问题，在这种配置下，我们分别应用了三种不同的损失函数来训练神经网络并对传统的损失函数进行了修正以适应我们的特殊场景。我们的在线训练神经网络的方法同样也可以被其他类似问题借鉴。

我们验证了基于神经网络的推荐算法在三种不同的损失函数下的性能表现，发现在使用策略梯度损失函数的情况下，在合理的参数配置下，我们的算法取得了（6.67%-6.53%）/6.53%＝2.1% 的算法收益。通过推荐详情的可视化分析，我们发现：基于神经网络的推荐算法能够在推荐数量相对较少的文章上取得较高的推荐准确率，这说明基于神经网络的推荐算法更能够把握用户特征的细节，这正是得益于神经网络较为强大的非线性拟合能力。

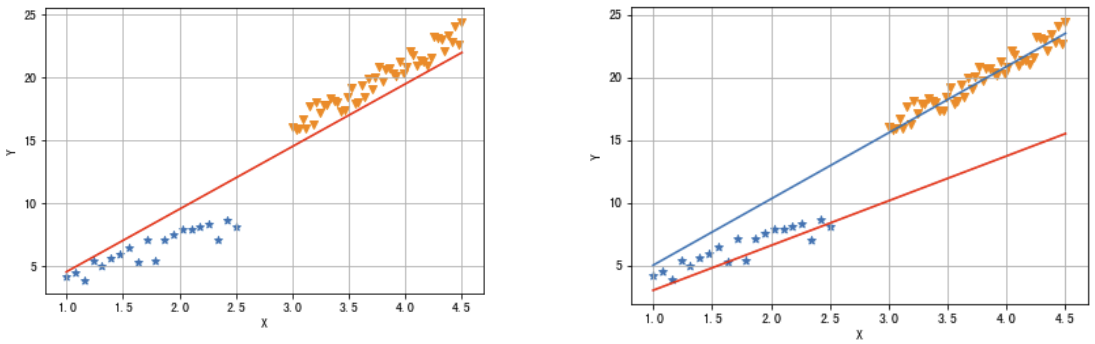
5　用户特征敏感的分级推荐算法

针对传统 Contextual-Bandit 算法没有考虑用户异质性的特点，推荐性能不高的问题，本章提出一种对用户特征敏感的分级推荐算法。该算法能够动态判别用户所属的类别，然后根据用户的类别，动态匹配合适的推荐器，来获得最佳的推荐性能。最后本章将评估算法的性能。

5.1　用户特征敏感的分级推荐算法

5.1.1　基本想法

在第三章中，我们系统地分析了在用户特征分布不均匀的情况下可能给推荐系统造成的问题，为了解决类似于图 5-1 中 a 图存在的问题，我们拟使用两条直线（模型）来拟合不同的样本点。



a　单模型线性回归
a　Single model for linear regression

b 双模型线性回归
b　Double models for linear regression

图 5-1 线性回归示意图
Figure 5-1　The diagram for linear regression

在图 b 中，针对于同样的样本点数据分布，对于不同的样本类簇，我们使用不同的直线去拟合不同类簇的样本点，可见，对于特定的一个类簇，我们使用更符合它的变化规律的直线进行拟合，对于不同的输入（X），如果我们能够辨别出它属于哪个类别，继而采用不同的直线来预测，这样产生的样本预测值明显会比图 a 采用一个直线（模型）来进行预测要准确的多。

5.1.2 多推荐器的引入

经过上一小节的示例分析，我们可以看到，在输入特征（X）在空间内被分为若干类的情况下，采用多个模型来进行预测，当新的样本点输入，当算法能够准确分辨出新输入的样本点属于哪个类簇的时候，将新输入的样本点使用正确分类的那个模型进行预测，则可以取得较好的预测结果。

我们可以将一维特征输入（X）当作五维特征在数轴上的特例，则每条直线可以理解为对不同样本点类簇的预测模型。对应地，在我们的新闻推荐问题中，输入特征变为用户的高维特征向量，预测模型变为 LinUCB 算法。经过第三章的测量观察，由于用户的特征空间内分布极不均匀，因此，可以使用多个推荐器（LinUCB）分别适用于不同的用户群体。

由于每个 LinUCB 算法中都为当前可以被推荐的每篇新闻维护了不同的偏好向量，所以即使是同一篇新闻，在不同的 LinUCB 算法模型中，他们的偏好向量也是不同的。在这种情况下，针对于不同的用户群体，相当于都为其量身定做了一个推荐器（兴趣模型），针对于不同的样本簇，每个簇的样本点所拟合出的直线（模型）都有着不同的斜率和截距。这就相当于丰富了模型预测的细节，使得模型的预测更有针对性。

5.1.3 用户类别分类器的引入

经过上一小节的示例分析，我们可以看到，在输入特征（X）在空间内互不交叉且被分为若干类的情况下，采用多个模型来进行预测，当新的样本点输入，当算法能够准确分辨出新输入的样本点属于哪一个特征类簇时，系统就可以将推荐器置为系统所要采取的推荐器，系统最后将该推荐器的推荐结果作为推荐结果展示给用户。

为了实现上述功能，我们必须在两个推荐器的上一级引入判别模块，该判别模块的作用也就是根据当前到访用户的特征，来决定第二级的推荐器到底采用哪一个。这与传统的机器学习中的分类问题划分训练集测试集不同，在本问题中，推荐行为是沿着时间轴在线、动态地向前推进的，根据用户的特征而选择不同二级推荐器的策略也应该是不断地调整和变化的，因此该策略也是一个动态、在线的算法。

基于以上分析，我们在第一级中选用的算法依然是 LinUCB 算法，只是对于这个 Bandit 对象来说，它所采取的动作集合不再是新闻候选集合，而是第二级的多个推荐器所构成的集合。

在这种设计下，我们希望对于每个到访的用户，都可以经过第一级的判定模块

被分配到第二级中适合自己的推荐器中，从而提高模型的预测精度，近而提高整体的推荐准确率。因此，推荐器对于用户的特征分布是敏感的。

5.1.4 算法框架和实现

经过上述介绍，本小节将给出整个推荐流程的示意图如图 5-2 所示，并给出算法实现的伪代码。

(1) 模型结构

整个模型的结构是分为两级的，每一级的子模型都是 LinUCB 算法。如图 5-2 所示，在第一级中，只有一个 LinUCB 对象，它的动作集合是第二级中的多个 LinUCB 对象，我们称之为 Master。第二级中有多个 LinUCB 对象，他们的动作集合都是一样的且都是新闻推荐集合中的新闻。他们从新闻候选集合中挑选新闻推荐给当前的用户，我们称第二级的 LinUCB 对象为 Slave。因此，整个的推荐模型是一个“主从”式的推荐结构。Master 负责甄别用户特征所属的类别，Slave 用于对当前的用户做（预测）推荐。

(2) 推荐和模型更新流程

在明确了推荐模型的结构后，我们将以一个示例来说明整个模型的推荐流程：如图所示，在 t 时刻，当用户到访系统时，Master 首先根据当前用户的特征向量确定应由第二级的哪一个 Slave 推荐器对当前用户产生推荐，即根据用户的特征首先对用户做相应的类别划分。假设 Master 选择器选定的是使用第二级中的 Slave1 作为推荐器最终对用户产生推荐，那么 Slave1 同样观察用户的特征向量，根据其 Slave1 本身的参数推荐一篇新闻给当前用户。比如，推荐第三篇新闻 N3。而需要注意的一点是，由于 Slave2 和 Slave1 中的模型参数不同，在同样的用户特征下，推荐结果可能不同。比如，Slave2 对于同样的用户给出的新闻推荐可能是 N4，由于当前用户已经被判别为更适用于 Slave1 所给出的结果，因此，在大概率的情况下，N3 被用户接受的概率要大于 N4 被用户接受的概率。由于最终产生的推荐结果是 Master 选择的结果和 Slave1 从新闻候选集中选择的结果共同作用得到的，因此，我们应该由用户的反馈同时更新 Master 和选定的 Slave。即使用（推荐的新闻，用户特征，用户反馈）这一个三元组去更新选定的 Slave1，使用（选定的 Slave，用户特征，用户反馈）这一个三元组去更新 Master。这样，第二级的推荐器更新关于新闻推荐的策略，第一级的类别选择器更新关于用户特征类别判定的策略，两者都持续得到动态更新，相互配合，从而提升性能。

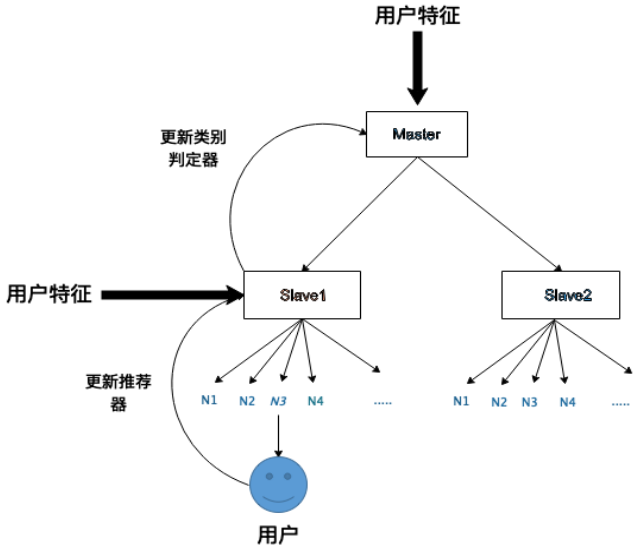


图 5-2 分级策略示意图
Figure 5-2 The diagram of level based policy

(3) 算法伪代码

经过以上算法的示例说明，本小节我们给出算法的伪代码：

算法 5-1 两级新闻推荐算法
Algorithm 5-1 A two-level method for news recommendation

Algorithm 5-1: A two-level method for news recommendation

0: Input: α
1: Initialize **Master** model and **Slave** models
2: **for** t =1,2,3….T, **do**
3: Observe the current user feature $X_t \in R^d$
4: Feed X_t into **Master M** and choice the picked **Slave PS**
5: Feed X_t into **PS** and recommened some particular news a_i
6: Observe the user's feedback r_t
7: Update **M** with (X_t, **PS**, r_t)
8: Update **PS** with (X_t, a_i, r_t)
9: **end for**

该算法对于每个到访的用户，首先将用户的特征送入到 Master 类别分类器中(3 行)，Master 分类器会给出一个动作输出，即选择第二级的哪个推荐器，第五行将用户的特征送入到选定的第二级分类器然后得到推荐给当前用户的新闻进行推荐，第六行观察用户的反馈，第七行利用用户的特征，选定的推荐器和用户的反馈

更新 Master 类别分类器，第八行利用用户的特征，推荐的新闻和用户的反馈更新选定的推荐器。

5.2 用户特征敏感的分级推荐算法评估

5.2.1 性能表现详情

在本小节，我们将报告基于用户特征敏感的分级推荐算法的性能表现。在线评估方法仍然复用在 4.2 中的在线评估方式，数据集复用 3.1.2 中介绍的 Yahoo 数据集。为了对比我们算法的有效性，第二级的推荐器使用的是 LinUCB 算法。推荐器的探索系数依然被设定为在 Yahoo 数据集上最佳的超参数 0.2，第一级的分类探索器 LinUCB 对象的探索系数 α 是我们的算法超参数。

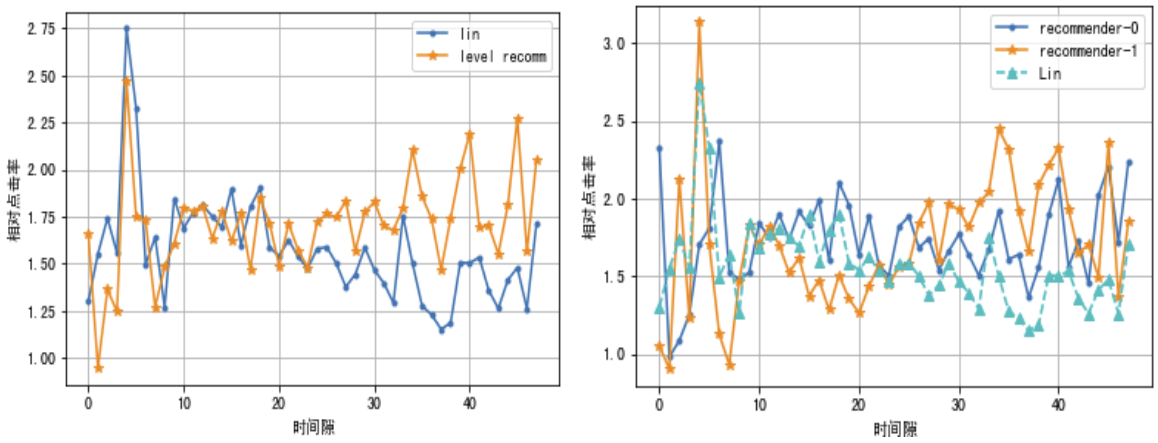
表 5-1 分级算法性能表现
Table 5-1 The performance for level policy

算法	类别决策器α	CTR
LinUCB	--	6.53%
分级推荐算法	0.1	6.6%
	0.2	6.66%
	0.3	**6.75%**
随机策略	--	3.98%

由表 5-1 可得，当第一级 Master 类别决策器的超参数为 0.3 的情况下，使用分级推取得了最高的推荐准确率。且在其他超参数的情况下分级算法的性能也要优于传统的 LinUCB 推荐方法。这很好地证明了我们算法的有效性，接下来我们将分析算法的具体推荐详情。

5.2.2 推荐详情分析

为了详细分析整个算法的工作细节，得到我们分级算法的收益所在。我们同样绘制了每个推荐器的推荐详情，在这种情况下，我们分别绘制了每个子推荐器的相对于随机策略的 CTR 变化曲线和整个分级策略的相对 CTR 变化曲线如图 5-3 所示。



a　分级推荐策略 CTR
a　The CTR of level policy

b　子推荐器推荐 CTR
b　The CTR of slave recommender

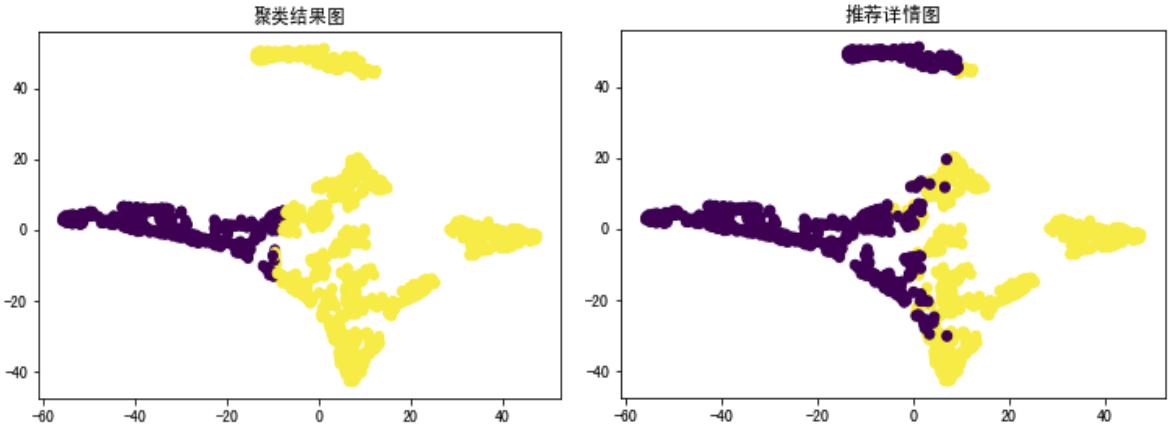
图 5-3 分级推荐策略 CTR 详情
Figure 5-3　The CTR detail of level policy

如图 5-3 所示，我们可以看到，在图 a 中，在前 20 个左右的时隙内，传统的 LinUCB 推荐算法比我们的分级推荐算法表现要好一些，而在后面的时隙中，分级推荐算法的表现要明显高于传统的 LinUCB 算法。造成推荐初期分级策略推荐效果不佳的原因来自于两方面：1. 第一级的用户特征分类器在学习初期还未收敛，不能根据用户的特征作出精准分类；2. 第二级的推荐器，与单个的 LinUCB 推荐器相同，在相同的时间区间内，每个子推荐器能够获得的训练数据被减少，模型也未收敛，因此比只用一个 LinUCB 推荐器推荐效果低。后面的时间区间内，分级推荐策略反超传统的 LinUCB 推荐策略的原因同样来自于两个方面：1. 第一级用户特征分类器能够精准分配用户的特征；2. 每个子推荐器都在各自负责的用户特征空间内趋向于收敛，能够获得较佳的推荐准确度。我们可以从图 b 中清晰地分析这个结论，当子推荐器收敛后，每个子推荐器的性能确实都要优于传统的 LinUCB 算法。

以上分析也可以得出一个值得关注的问题：增加推荐器的数量也是有代价的。增加推荐器的数量意味着在固定的时间间隔内，每个推荐器能够学习到的样本也会随之减少，推荐器的收敛也会变慢。极端情况下，为每个不同的用户特征都分配一个推荐器能够获得理论最佳的推荐效果，但是这种情况下推荐器能够利用的数据将会大大减少，推荐器得不到收敛，反而会取得很差的推荐效果。我们同样实验了第二级有三个推荐器、四个推荐器的情况，发现效果并不佳，正是由于上述原因引起的。

为了观察 Master 模块对用户的特征的分配情况，我们对上文中提到的 1000 个不同的用户特征向量同样采用 t-SNE 降维技术将高维用户特征向量降为二维，在

二维平面上做了可视化，如图 5-4 中 b 图所示。同时，我们也是用 K-means 聚类技术对该一千个不同的用户特征向量聚类，然后使用 t-SNE 技术将高维向量在二维空间做了可视化，如图 5-4 中 a 图所示。



a　聚类详情图　　b　推荐详情图

a　The detail of cluster　　b　The detail of recommendation

图 5-4 聚类和推荐器用户特征分布图

Figure 5-4　The user feature distribution for cluster and recommendation

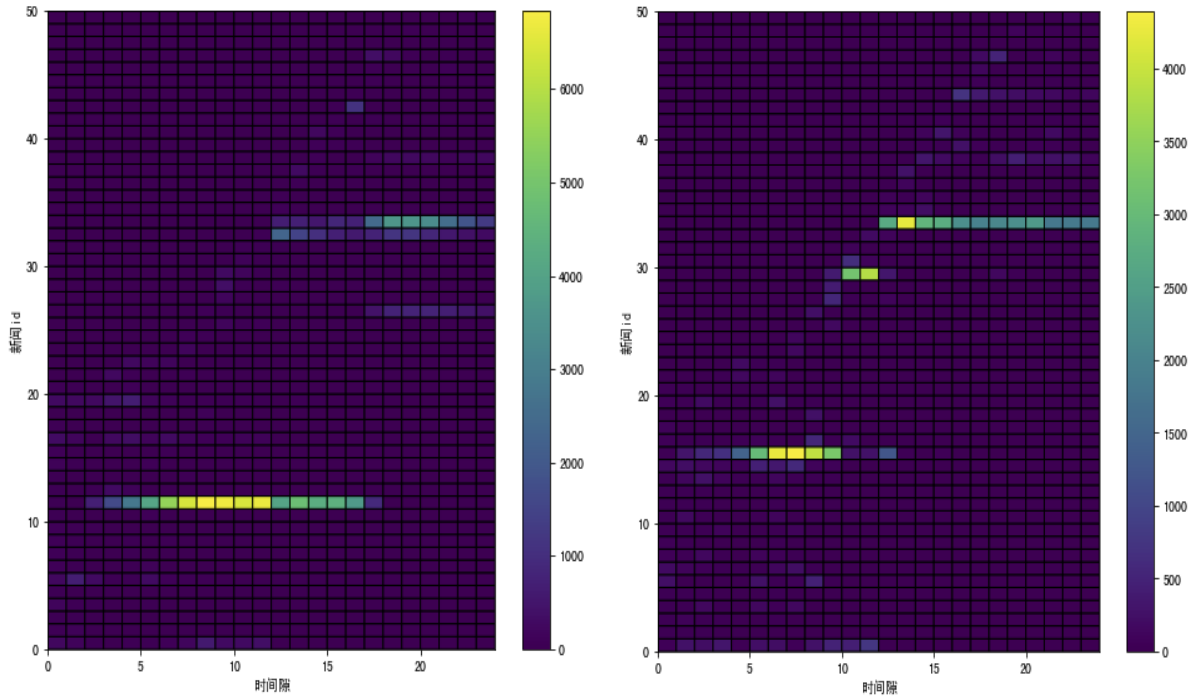
如图 5-4 所示，图中是一千个不同的用户特征点。a 图是使用 K-means 算法对用户特征进行聚为两类的结果，图 b 是两个子推荐器对这一千个不同的特征向量推荐的结果。其中紫色的特征点是第一个推荐器推荐过的用户特征点，黄色特征点是第二个推荐器推荐过的用户特征点。对比两张图来看，K-means 算法对用户特征的分类和我们的动态的分级推荐策略的分类界限大致相同，而且每个子类的轮廓也基本类似。这说明：与 K-means 算法类似，我们的 Master 用户特征分类器实现了对用户群体的划分，两类特征点的样本簇被明显地分割开，也就是说，每个子推荐器“各司其职”，分别负责各自推荐较为准确的用户特征空间的用户。但是，也需要注意的是，Master 分类器对用户特征的分割界面与 K-means 也存在一些不同和差异。这主要是由两方面因素造成的：1. K-means 算法是一个静态的算法，该算法是在所有的可见的数据集上运算得到的分类界面，而 Master 用户类别分类器是一个动态的、在线的算法，该算法通过不断与环境交互从而从而优化自身分类准则，不同时刻分类规则也是不同的。2.与 K-means 算法不同，K-means 的分类界面是通过严格计算用户特征向量之间的距离来衡量的，而 Master 分类器是通过用户的反馈与特征之间的关系来对用户的特征作出的分类决策，用户对于同一篇文章的表现差异并非严格地与用户特征之间的差异存在某种特定关系。

另外一个现象是在图 b 中我们可以看到样本点也有个别的样本位于另一个类簇中，这同样是由两部分原因造成的：1.在推荐初期，用户特征分类器并不能很好

地对用户特征进行分类；2.我们的用户特征分类器采用的也是 LinUCB 方法，同样也具有探索性。

综上所述，相对于全体用户的特征簇，每个子推荐器所负责的用户群体的特征簇相对来说比较集中，因此，每个子模型在表征能力不变的情况下，在分布简单的数据集合上取得了相对于针对于整个用户特征簇更高的推荐准确率。这也就很好地解释了为什么在后面的时间段上每个子推荐器的推荐效果都要高于传统的单个 LinUCB 推荐算法。

我们同样可视化了两个子推荐器的推荐详情如图 5-5 所示，由图可见，对于两个子推荐器，虽然推荐器的基本参数配置是完全一样的，但是两个推荐器的推荐详情有着很大的差异，在相同的时间区间内，两个推荐器重点推荐的文章也不相同。这是因为两个推荐器所负责的用户特征簇不同，对于不同类型的用户，用户最感兴趣的新闻应该也是不同的，因此，两个子推荐器的推荐情况也相应地存在变化。



a　子推荐器 0 推荐详情
a The recommendation detail for recommender 0

b 子推荐器 1 推荐详情
b　The recommendation detail for recommender 1

图 5-5 子推荐器推荐详情
Figure 5-5 The recommendation detail for slave recommenders

5.3 本章小结

为了解决用户特征分布异质化给推荐器带来的问题，使得模型能够更加细粒度地去预测，得到更加准确的结果，本章提出了一种分级的推荐策略，使得到访的用户被分配到更适合于自己的那个推荐器中，来获得更佳的推荐结果。这种设计思路同样也可以应用到其他类似的相关问题中。

我们系统地评估了分级推荐算法的性能，实验结果显示：相较于传统形式的LinUCB 推荐算法也取得了不错的算法提升（6.75%-6.53%）/6.53%=3.3%。在我们的算法中，经过第一级的类别鉴定器，模型能够动态感知用户的特征来选择第二级的推荐器对用户进行推荐。经过可视化分析，我们发现：第二级的推荐器将用户特征在高维空间内进行了划分，各自负责归属于自身特征簇的部分用户的推荐。当子模型收敛后，每个子推荐器的推荐准确率都要高于传统的单个 LinUCB 推荐器，且两个子推荐器的推荐详情也不同。由此说明：每个子推荐器的推荐更加个性化。

6 总结及展望

6.1 总结

本文选取了一个具有代表性的动态推荐算法场景：新闻推荐场景，基于一个大规模在线数据集，首先测量并观察了新闻推荐场景中的新闻流行度的动态特性和推荐候选集的动态特性和用户特征在空间内的分布规律，发现用户特征在空间内分布不均匀。然后，为进一步提高以 LinUCB 为代表的 Contextual-Bandit 类算法的表征能力和解决其没有考虑用户特征在空间内分布情况的问题，提出两类算法来提高算法性能。具体贡献如下：

(1) 针对现有模型表达能力欠佳的问题，我们提出使用神经网络拟合用户特征和新闻期望之间的关系，解决了神经元网络更新和损失函数选择的两个难题，得以利用神经元网络的表征能力，改进动态推荐算法的性能。为了提高神经网络在线训练的效率，我们提出了重复利用用户反馈多次迭代的在线训练方式，并且，不同的用户反馈采用不同迭代次数。实验结果表明：该方法相对于传统的训练方式取得了近 40%的增益，证明了这种在线训练方式的可行。其次，我们系统尝试了各种损失函数，包括分类、回归和策略梯度三种，通过大量的实验发现：在合理的配置下，采用策略梯度的损失函数，我们的算法相较于 LinUCB 算法取得了 2.1%的性能增益，证明了算法的性能。

(2) 针对传统 Contextual-Bandit 算法没有考虑用户异质性的特点，我们创新性地提出了一种对用户特征敏感的分级推荐算法。该算法能够动态判别用户所属的类别，然后根据用户的类别，动态匹配合适的推荐器，来获得最佳的推荐性能。实验表明，该算法相较于传统的 LinUCB 推荐算法取得了 3.3% 的性能增益，证明了算法的性能。

同时，我们也针对于提出的模型的工作机制、算法增益来源都进行了详细的可视化分析和讨论，我们的设计思路和解决相关问题的方案同样可以应用到其他类似问题中。

6.2 未来工作展望

Contextual-Bandit 作为一类非常重要的动态推荐算法已经在工业界取得了广泛应用，它很好地解决了冷启动、物品流行度高度变化给推荐系统带来的问题。但

是，此类算法对于超参数的选择是非常敏感的，在实际的系统中，需要通过实验验证来确定最优参数。近年来，“元学习”（meta-learning）的研究已经取得了不错的进展，他的主要任务是让算法“学会如何学习”，即让算法根据以往的经验来调整自身的策略来更加高效、智能地进行训练和学习，以期得到最佳的算法性能。因此，将元学习和 Contextual-Bandit 推荐算法相结合必是将来研究的一大热点。

参考文献

[1] http://www.yahoo.com.
[2] https://www.bytedance.com
[3] https://www.taobao.com
[4] Hu Y , Koren Y , Volinsky C . Collaborative filtering for implicit feedback datasets[C]// 2008 Eighth IEEE International Conference on Data Mining. IEEE, 2009.
[5] Yan M, Shang W, Li Z. Application of SVD technology in video recommendation system[C]// IEEE/ACIS International Conference on Computer & Information Science, 2016.
[6] Barkan O, Koenigstein N. ITEM2VEC: Neural item embedding for collaborative filtering[C]//IEEE International Workshop on Machine Learning for Signal Processing, 2016.
[7] Koren Y, Bell R, Volinsky C. Matrix factorization techniques for recommender systems[J]. Computer, 2009 (8): 30-37.
[8] Vieira J P A, Moura R S. An analysis of convolutional neural networks for sentence classification[C]// Computer Conference, 2017.
[9] Tang J, Wang K. Personalized top-n sequential recommendation via convolutional sequence embedding[C]//Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining. ACM, 2018: 565-573.
[10] Zhao P, Zhu H, Liu Y, et al. Where to Go Next: A Spatio-temporal LSTM model for Next POI Recommendation[J],2018.
[11] Jannach D, Ludewig M. When recurrent neural networks meet the neighborhood for session-based recommendation[C]// Eleventh Acm Conference on Recommender Systems. 2017.
[12] 项亮. 动态推荐关键技术研究[D]. 中国科学院大学，2011
[13] Jia Y, Zhang C, Lu Q, et al. Users' brands preference based on SVD++ in recommender systems[C]// Advanced Research & Technology in Industry Applications. 2014.
[14] Manzato M G. gSVD++:supporting implicit feedback on recommender systems with metadata awareness[C]// ACM Symposium on Applied Computing. 2013.
[15] Cao J, Hu H, Luo T, et al. Distributed design and implementation of SVD++ algorithm for e-commerce personalized recommender system[C]//National Conference on Embedded System Technology. Springer, Singapore, 2015: 30-44.
[16] Gantner Z, Drumond L, Freudenthaler C, et al. Learning Attribute-to-Feature Mappings for Cold-Start Recommendations[C]//ICDM, 2010, 10: 176-185.
[17] Bouneffouf D, Bouzeghoub A, Arski A L. A contextual bandit algorithm for mobile context-aware recommender system[M]// Neural Information Processing. 2012.
[18] Jung T, Martin S, Ernst D, et al. Contextual multi-armed bandits for web server defense[C]// International Joint Conference on Neural Networks. 2012.
[19] Li L, Chu W, Langford J, et al. A contextual-bandit approach to personalized news article recommendation[C]//Proceedings of the 19th international conference on World wide web. ACM, 2010: 661-670.
[20] Tang L, Jiang Y, Li L, et al. Personalized recommendation via parameter-free contextual bandits[C]// International ACM SIGIR Conference on Research & Development in

Information Retrieval. 2015.
[21] Lu W, Wang C, Wang K, et al. BiUCB: A contextual bandit algorithm for cold-start and diversified recommendation[C]// IEEE International Conference on Big Knowledge. 2017.
[22] Hai T N, Kofodpetersen A. Using multi-armed bandit to solve cold-start problems in recommender systems at Telco[C]// International Conference on Mining Intelligence & Knowledge Exploration. 2014.
[23] Caron S, Bhagat S. Mixing bandits: A recipe for improved cold-start recommendations in a social network[C]//Proceedings of the 7th Workshop on Social Network Mining and Analysis. ACM, 2013: 11.
[24] Mary J, Gaudel R, Philippe P. Bandits warm-up cold recommender systems[J]. arXiv preprint arXiv:1407.2806, 2014.
[25] Schein A I, Popescul A, Ungar L H, et al. Methods and metrics for cold-start recommendations[C]//Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval. ACM, 2002: 253-260.
[26] Adrian Colyer. An efficient bandit algorithm for real-time multivariate optimization. https://blog.acolyer.org/2017/09/27/an-efficient-bandit-algorithm-for-real-time-multivariate-optimization/
[26] Filippi S, Cappe O, Garivier A, et al. Parametric bandits: The generalized linear. case[C]//Advances in Neural Information Processing Systems, 2010: 586-594.
[28] Wang H, Wu Q, Wang H. Learning hidden features for contextual bandits[C]//Proceedings of the 25th ACM International on Conference on Information and Knowledge Management. ACM, 2016: 1633-1642.
[29] Wu Q, Wang H, Gu Q, et al. Contextual bandits in a collaborative environment[C]//Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval. ACM, 2016: 529-538.
[30] Wang H, Wu Q, Wang H. Factorization bandits for interactive recommendation[C]//Thirty-First AAAI Conference on Artificial Intelligence, 2017.
[31] Krause A, Ong C S. Contextual gaussian process bandit optimization[C]//Advances in Neural Information Processing Systems, 2011: 2447-2455.
[32] Valko M, Korda N, Munos R, et al. Finite-time analysis of kernelised contextual bandits[J]. arXiv preprint arXiv:1309.6869, 2013.
[33] 张佃磊. 基于自适应上下文多臂赌博机推荐算法研究[D]. 山东大学，2018.
[34] Van den Oord A, Dieleman S, Schrauwen B. Deep content-based music recommendation[C]//Advances in neural information processing systems, 2013: 2643-2651.
[35] Wang H, Zhang F, Xie X, et al. DKN: Deep knowledge-aware network for news recommendation[C]//Proceedings of the 2018 World Wide Web Conference on World Wide Web. International World Wide Web Conferences Steering Committee, 2018: 1835-1844.
[36] Fu M, Qu H, Yi Z, et al. A novel deep learning-based collaborative filtering model for recommendation system[J]. IEEE transactions on cybernetics, 2018 (99): 1-13.
[37] Zhang L, Luo T, Zhang F, et al. A recommendation model based on deep neural network[J]. IEEE Access, 2018, 6: 9454-9463.
[38] Dumitrascu B, Feng K, Engelhardt B. PG-TS: Improved thompson sampling for logistic

contextual bandits[C]//Advances in Neural Information Processing Systems, 2018: 4629-4638.

[39] Langford J , Zhang T . The epoch-greedy algorithm for multi-armed bandits with side information[J]. Nips’, 2007:817-824.

[40] Slivkins A. Contextual bandits with similarity information[J]. The Journal of Machine Learning Research, 2014, 15(1): 2533-2568.

[41] Ten Hagen S, Van Someren M, Hollink V. Exploration/exploitation in adaptive recommender systems[J]. Proceedings of Eunite 2003, 2003.

[42] Kuleshov V, Precup D. Algorithms for multi-armed bandit problems[J]. arXiv preprint arXiv:1402.6028, 2014.

[43] Peter Auer M L . Using confidence bounds for exploitation-exploration trade-offs[J]. Journal of Machine Learning Research, 2002, 3(3):397-422.

[44] Wang Q, Zeng C, Zhou W, et al. Online Interactive Collaborative Filtering Using Multi-Armed Bandit with Dependent Arms[J]. IEEE Transactions on Knowledge and Data Engineering, 2018: 1-1.

[45] Nguyen T T, Lauw H W. Dynamic clustering of contextual multi-armed bandits[C]//Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management. ACM, 2014: 1959-1962.

[46] Agrawal S, Goyal N. Thompson sampling for contextual bandits with linear Payoffs.[C]// International Conference on International Conference on Machine Learning,2013.

[47] Zeng C , Wang Q , Mokhtari S , et al. Online context-aware recommendation with time varying multi-Armed bandit[C]// ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. ACM, 2016.

[48] Russo D, Van Roy B. Satisficing in Time-Sensitive Bandit Learning.[J]. arXiv: Learning, 2018.

[49] u S T , Zhu L J . A bandit method using probabilistic matrix factorization in recommendation[J]. Journal of Shanghai Jiaotong University (Science), 2015, 20(5):535-539.

[50] Tang L , Jiang Y , Li L , et al. Ensemble contextual bandits for personalized recommendation[C]// ACM Conference on Recommender Systems. ACM, 2014.

[51] Lecun Y, Bengio Y, Hinton G. Deep learning.[J]. Nature, 2015, 521(7553):436.

[52] Schmidhuber J. Deep learning in neural networks: an overview[J]. Neural Netw, 2015, 61:85-117.

[53] 焦李成, 赵进, 养淑媛,等. 深度学习、优化与识别[M]. 清华大学出版社, 2017.

[54] Barto A G. Reinforcement learning[J]. A Bradford Book, 1998, volume 15(7):665-685.

[55] David Silver. Policy-Based Reinforcement Learning, Policy Gradient. http://www0.cs.ucl.ac.uk/staff/d.silver/web/Teaching_files/pg.pdf

[56] https://en.wikipedia.org/wiki/T-distributed_stochastic_neighbor_embedding

[57] Maaten L, Hinton G. Visualizing data using t-SNE[J]. Journal of machine learning research, 2008, 9(Nov): 2579-2605.

[58] https://www.python.org

[59] https://scikit-learn.org

[60] https://tensorflow.google.cn

[61] https://keras.io

[62] Riquelme C, Tucker G, Snoek J. Deep bayesian bandits showdown[C]//International Conference on Learning Representations, 2018.

作者简历及攻读硕士学位期间取得的研究成果

一、作者简历

唐伟康，男，1994 年 4 月生。2012 年 9 月至 2016 年 7 月就读于哈尔滨理工大学电气与电子工程学院电子信息工程专业，取得工学学士学位。2016 年 9 月至 2019 年 6 月就读于北京交通大学通信与信息系统专业，研究方向是信息网络，取得工学硕士学位。攻读硕士学位期间，主要从事机器学习、推荐系统的研究工作。

独创性声明

本人声明所呈交的学位论文是本人在导师指导下进行的研究工作和取得的研究成果，除了文中特别加以标注和致谢之处外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得北京交通大学或其他教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

学位论文作者签名：　　签字日期：　　年　　月　　日

学位论文数据集

关键词*	密级*	中图分类号	UDC	论文资助
动态推荐；推荐系统；新闻推荐；上下文老虎机	公开			
学位授予单位名称*		学位授予单位代码*	学位类别*	学位级别*
北京交通大学		10004	工学	硕士
论文题名*		并列题名		论文语种*
推荐系统中动态推荐算法研究				中文
作者姓名*	唐伟康		学号*	16120125
培养单位名称*		培养单位代码*	培养单位地址	邮编
北京交通大学		10004	北京市海淀区西直门外上园村 3 号	100044
学科专业*		研究方向*	学制*	学位授予年*
通信与信息系统		信息安全	3.0	2019
论文提交日期*	2019.06.03			
导师姓名*	陈一帅		职称*	副教授
评阅人	答辩委员会主席*		答辩委员会成员	
	郭宇春			
电子版论文提交格式 文本( ) 图像( ) 视频( ) 音频( ) 多媒体( ) 其他( ) 推荐格式：application/msword；application/pdf				
电子版论文出版(发布)者		电子版论文出版(发布)地		权限声明
论文总页数*	55 页			
共 33 项，其中带*为必填数据，为 21 项。				