Applied Artificial Intelligence
An International Journal

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/uaai20

Contextual Bandit Approach-based Recommendation System for Personalized Web-based Services

Akshay Pilani, Kritagya Mathur, Himanshu Agrawald, Deeksha Chandola, Vinay Anand Tikkiwal & Arun Kumar

To cite this article: Akshay Pilani, Kritagya Mathur, Himanshu Agrawald, Deeksha Chandola, Vinay Anand Tikkiwal & Arun Kumar (2021) Contextual Bandit Approach-based Recommendation System for Personalized Web-based Services, Applied Artificial Intelligence, 35:7, 489-504, DOI: 10.1080/08839514.2021.1883855

To link to this article: https://doi.org/10.1080/08839514.2021.1883855

Published online: 06 Apr 2021.

APPLIED ARTIFICIAL INTELLIGENCE
2021, VOL. 35, NO. 7, 489–504
https://doi.org/10.1080/08839514.2021.1883855

Taylor & Francis
Taylor & Francis Group

ARTICLE

Contextual Bandit Approach-based Recommendation System for Personalized Web-based Services

Akshay Pilani[a], Kritagya Mathur[a], Himanshu Agrawald[a], Deeksha Chandola[b], Vinay Anand Tikkiwal[b], and Arun Kumar[c]

[a]Department of Computer Science & Engineering and Information Technology, Jaypee Institute of Information Technology, Noida, India; [b]Department of Electronics and Communication Engineering, Jaypee Institute of Information Technology, Noida, India; [c]Department of CSE, National Institute of Technology, Rourkela, India

ABSTRACT

In recent years, recommendation systems have started to gain significant attention and popularity. A recommendation system plays a significant role in various applications and services such as e-commerce, video streaming websites, etc. A critical task for a recommendation system is to model users' preferences so that it can attain the capability to suggest personalized items for each user. The personalized list suggested by a suitable recommendation system should contain items highly relevant to the user. However, many a times, the traditional recommendation systems do not have enough data about the user or its peers because the model faces the cold-start problem. This work compares the existing three MAB algorithms: LinUCB, Hybrid-LinUCB, and CoLin based on evaluating regret. These algorithms are first tested on the synthetic data and then used on the real-world datasets from different areas: Yahoo Front Page Today Module, Lastfm, and MovieLens20M. The experiment results show that CoLin outperforms Hybrid-LinUBC and LinUCB, reporting cumulated regret of 8.950 for LastFm and 60.34 for MovieLens20M and 34.10 for Yahoo FrontPage Today Module.

ARTICLE HISTORY
11 February 2021

INTRODUCTION

Existing recommender systems have a vast pool of items to serve user needs. The books, articles or music provided by the recommender systems are referred to as items. Recommending item(s) that is most relevant to users becomes difficult, as the user cannot explore all the items available in the system. The main goal of a recommender system is to provide similar items to the user in which they are interested. Personalized recommender systems gradually learn about the interests of the users and provide them with items, which cater to their interests. The main three types of recommender systems are Content-based Recommender Systems (Lops, De Gemmis, and Semeraro 2011), Collaborative Filtering (CF)

CONTACT Vinay Anand Tikkiwal vinay.anand@jiit.ac.in; Himanshu Agrawald himanshu.agrawal@jiit.ac.in Department of Computer Science & Engineering and Information Technology, Jaypee Institute of Information Technology, Noida, India; Arun Kumar kumararun@nitrkl.ac.in

based Recommender Systems (Ben Schafer et al. 2007) and Hybrid Recommender Systems (Burke 2002). Content-based recommendation systems analyze the nature of each item. In this technique, the users are recommended with those similar items which they have preferred in the past. In this way, the user profile is built up. When implementing a content-based filtering system (Lops, De Gemmis, and Semeraro 2011), several issues are considered. First, terms can either be assigned manually or automatically. Second, the terms have to be represented in such a way that both the item and the user profile can be compared in a meaningful way. Third, a learning algorithm has to be chosen that can learn the user profile based on items seen by that user and can recommend items based on the user profile.

For instance, recommending movies to a user by analyzing features of a song such as a genre, artist, era, and many more. The content-based approach is based on abstract similarity for an academic recommendation system, which uses eTBLAST model based on z-score which is computed using a set of extracted keywords and journal score as reported in the journals' website. Wang D. et al. (Wang et al. 2018) proposed a Publication Recommender System (PRS) to help the authors for a fast submission process and to find a suitable venue, particularly for computer science publications. They developed a new content-based filtering model using softmax regression and chi-square. It covers five digital libraries such as Springer, IEEE, ACM, AAAI and SIAM, for more than 50 publications. Carla, B. et al. (Binucci et al. 2017) worked on a travel recommender system, which suggests the best tourist attractions based on given POI's (Point of Interest) and user's preference. The content analyzer accepts a point of interest as input and computes the relevance of these POIs concerning TOI's (topic of interest). Emrah, I. et al. (Inan, Tekbacak, and Ozturk 2018) proposed a Movie Recommender system with goal programming (Moreopt) system. This system is a combination of a content-based approach and collaborative filtering and aims at improving the drawback of scalability and scarcity of data.

The idea for the collaborative filtering came from the fact that people often get the best recommendations from someone with a similar taste, i.e. it relates the users' having a similar history to that of a current user. Collaborative filtering recommends items by using techniques that match people with similar interests. Collaborative filtering-based systems work by analyzing and collecting a huge amount of data about users' preferences, activities, behaviors and then predicting the preferences of users based on their similarity to other users. The main advantage of the collaborative filtering systems is that they are not dependent on the content analyzable by machine and therefore they can recommend/complex/items (like news, movies, songs, etc.) accurately without having any understanding of the item. Anam, M. et al. (Mustaqeem, Anwar, and Majid 2020) proposed an improved collaborative flltering health recommendation system using clustering and sub clustering techniques. They

worked on four different types of cardiovascular disease dataset, including angina, non-cardiac chest pain, silent ischemia, and myocardial infarction. In this system, the patient's data is divided into different clusters according to the disease. Once the query patient is directed toward the correct disease, a similarity score is computed, which later on helps in improving the efficiency of the system. Shun et al. (Shun et al. 2019) worked on a personalized electricity tariff recommendation system. The proposed system collects group electricity consumption profiles of end-users with the help of intelligent metering. Using this information, the system will generate a new factorization matrix based on content-based filtering that aims to recommend the most suitable electricity tariff plan to a target user. This system is validated on the Australian "Smart Grid, Smart City" dataset. Shunmei Meng et al. (Meng et al. 2019) worked on designing a safe and secure system that can aim to minimize security attacks on sensitive data. This privacy-preserving collaborative recommender system is a location-based prediction method. Daniel Valcarc et al. (Valcarce et al. 2019) proposed a novel memory-based word embedding technique prefs2vec. It is an extension of the bag-of-words model and represents items and users mainly for the memory-based system. In addition to this method considers preferences registered in the past and is capable of building a collaborative filtering embedding for both item and user.

Hybrid Recommender Systems are a class of systems that combine content-based flltering and collaborative flltering. These systems take advantage of both the similarities among users as well as the content. There are several ways in which Hybrid approach can be implemented: by making content-based and collaborative-based predictions independently and then combining them; by using content-based capabilities in a collaborative-based approach (or vice versa); or by unifying the two approaches into a single model. Thuan and Puntheeranurak (Pradhan and Pal 2019) proposed a hybrid model, which includes review-helpfulness and product-preference, which aims at predicting the unknown ratings. This model uses target users as nearest neighbors. In a news article recommendation system, recommendations to a user are based on his/her current interests, make them explore new news articles, to know better and to know more about their interests. This creates a dilemma between exploitation and exploration, where exploitation means recommending the optimal items to a user, based on the payofis observed and exploration means learning about the payofi of new items' for a particular user by recommending new items to that user. New articles are added every day, and outdated articles are removed. Also, this constant change in the pool of news articles is incorporated into news article recommendation. When recommending news articles to users, some users may not have any history, i.e. if he/she is a new user on the system. Recommending articles to a new user can be a challenging task as there is no information present in the system about the user to gather his/her interests. This is known

as the cold-start problem in recommender systems. Traditional approaches like collaborative filtering and content-based filtering cannot handle these problems. Kiran R et al. (Kiran, Kumar, and Bhasker 2020) proposed Deep Learning-based Hybrid Recommender System (DNNRec) to improve the accuracy of the existing collaborative filtering systems using deep learning. This helps them to overcome Cold-start problem, i.e. for new users who have no history. They validated the system on different datasets such as MovieLens 1 M, FilmTrust, MovieLens 100 K, and Book-Crossing. Qian et al. (Qian et al. 2013) worked on a system, which combines personal influence, interest similarity including personal interest, and social factors, to recommend those items in which users are highly interested. Yongfeng Qiana (Qian et al. 2019) proposed an Emotion-aware recommender system (EARS), which considers implicit and explicit feedback information mainly reviews, clicks and ratings but most importantly, the user's emotions in the final purchasing of the product. This aims at improving the quality of the collaborative filtering recommender systems. Guo et al. (Guo, Zhang, and Yorke-Smith 2015) developed a trust-based recommender model (Merge) that only includes the trusted users' ratings for improving the accuracy which incorporates the ratings of trusted neighbors in providing recommendations. These approaches improve recommendation performance with the support of the social relationships in the network. Hei-Chia Wang et al. (Wang, Jhou, and Tsai 2018) worked on building a topic-level recommender system that aims to overcome the two main problems of low model scalability and cold-start of the existing recommender system. This model considers the relationship between social media influence and user preference.

Multi-armed bandit problem is a classical problem in the field of Computer Science. Multi-armed bandit algorithms provide a solution to the exploration-exploitation dilemma. Contextual Bandits are a type of bandit which uses expected payoff to recommend news articles. The expected payoff of a user is calculated using context and unknown bandit parameters. Here, context is a feature vector obtained by using the information of both user and news articles. Some Contextual Bandit algorithms consider the users to be independent of each other, i.e. unknown bandit parameters are calculated for each user independently.

Considering users to be independent of each other, it is noted that there is no social influence factor involved in the system. Since, this factor is involved in a lot of fields like movie recommendation, news article recommendation, etc. In this field, information about the interests of some user can also be used for recommending items to his/her friends. So, the observed payoff from some user might be a result of both his/her interests and social influence between users in the system. For evaluating the performance of various contextual bandit algorithms, performance metrics such as Cumulated Regret and Click Through Rate Ratio (CTR Ratio) are commonly used. The goal of the

recommendation algorithm is to optimize some performance metric for all users and over some time horizon.

In this paper, a detailed investigation of the current personalized recommendation MAB algorithms is performed on synthetic and large-scale real-world datasets, namely Yahoo! Front page today module, LastFM, and MovieLens20M. A comparative analysis of the three algorithms is presented. The rest of the paper is organized as follows: Section II presents the dataset description and pre-processing steps. The methodology is explained in Section III. Section IV presents and discusses the results obtained on different datasets. Furthermore, finally, the paper is concluded in section V.

Dataset description

This section explains the dataset used in this research work. Empirical evaluations of LinUCB, LinUCB with Hybrid Linear Models and Collaborative LinUCB algorithm on different parameters are performed, including N independent LinUCB (Qian et al. 2013). Out of these algorithms, LinUCBwith Hybrid model exploits dependency between users by a set of hybrid linear models. All these algorithms are/tested on a simulated dataset, Yahoo! Today Module dataset (Errami et al. 2007) – a large dataset in the form of user click logs from Yahoo! Today news site, LastFM (Errami et al. 2007) – a real-world dataset extracted from music streaming service LastFM and MovieLens20M (Wang, Jhou, and Tsai 2018) – a movie rating and recommending site (MovieLens.org).

Synthetic dataset

For Synthetic dataset, user feature vector and article feature vector are generated using random strategy. LinUCB, Hybrid-LinUCB and CoLin algorithm are compared using simulation setting. Assuming the number of users to be N, each user has a d-dimensional feature vector θ^*. Feature Vector of each user is generated by uniformly selecting a random number for every dimension and normalizing the vector by L2 norm. In each iteration of the algorithm, the users' are recommended an article from a pool of articles which is of fixed size called "pool size". The pool is created by randomly selecting a number of articles "pool size". After this, the algorithms are applied to the dataset, and a comparative analysis is performed by plotting graph of accumulated regret versus iterations for each algorithm. The accumulated regret eventually converges for each algorithm, which shows that parameters are also converged. The best performance is judged by observing how fast it starts to converge.

Yahoo! Today module

The dataset (Errami et al. 2007) contains a fraction of user clicks log for news articles displayed in the Featured Tab of the Today Module on Yahoo! FrontPage. In the dataset, information about both the users and articles is given in the form of respective feature vectors of dimension six. Since the data contains information in the form of logged events, so it cannot be directly fed into the algorithm. So, the dataset is converted into a format, which can be easily used to test the algorithms. To reduce the processing time, clustering using the k-means algorithm is performed on all users based on their features. For each user cluster, a feature vector from its article pool is required. For finding the article from a pool of cluster, this work considers the top 20 most occurring articles for every user in that cluster.

LastFM

The LastFM dataset (Turrin et al. 2015) is obtained from the last.fm music streaming service. The dataset contains tagging, social networking, and artist listening information of the users on the last.fm service. The first step in data pre-processing is to parse the dataset to create the feature vectors of each user and article. For each user, the information of artists is used to flnd payoffs, i.e. the payoff is considered to be one if the user has listened to an artist at least once; otherwise, it is 0. The second step is to create feature vectors for articles; the tags are broken into smaller tags which are made up of a single word. TF-IDF (Qaiser and Ali 2018) vectorization technique is applied to all the tags associated with an artist to create a vector. Finally, with the help of Principle Component Analysis (PCA) (Abdi and Williams 2010), the dimension of the feature vectors was reduced to 25 and is thus used as a resultant artist feature vector.

As the number of users is significant, to make algorithms computationally feasible, Graclus Clustering is implemented. It is graph-based clustering which considers user relation while clustering users into M clusters. The clusters are formed taking into consideration the users who are close friends and like the similar genre of songs on different values of M, to obtain an optimum value of M. The centroid of a cluster represents a common feature vector for all users in that cluster. The parameters trained by algorithms are shared by all users of the cluster to which the new user vector belongs.

MovieLens20M

MovieLens20M dataset (Cantador, Brusilovsky, and Kuflik 2011) describes star rating (out of 5) and free-text tagging activity. Firstly, the processing of dataset is done to extract the feature vector for users and movies. In the

dataset, there are tags associated with the movies, and each tag has a relevance score, i.e. how appropriate that tag is for that movie. Users also give these tags to all the movies. Again, the TF-IDF vectorization technique is applied to tags associated with an artist to create a TF-IDF vector. To reduce the dimension of the vector PCA is used to reduce the dimension of the vector to 25 and used the resultant vector as a movie feature vector. Similarly, for creating users' feature vector also TF-IDF vectorization technique is applied for each user using the tags that are related to movies that are watched and rated. Rating given by the user to a movie has been used as a weight to calculate the relevance of tags related to a movie for each user. To make the algorithm, computationally feasible K-means Clustering is performed. The centroid of the clusters acts as a new user vector. The algorithm runs on this new user vector and all the users of that cluster share the parameters trained by the algorithm.

Methodology

Contextual bandit algorithms are a combination of multi-armed bandits and user context. Specifically, for application such as personalized news recommendations, where each news article can be treated as an arm and the payoff for each news article will be different for different users. Thus, for contextual algorithms instead of the payoff for an arm being an estimate, the algorithm will model the payoff as a function of the user context. Thus, complexity is introduced in the model when applying Upper Confidence Bound (UCB) to the contextual bandit algorithms setting. In order to efficiently compute the confidence intervals, the three contextual bandit algorithms such as LinUCB (Chu et al. 2011), Hybrid-LinUCB (Lihong et al. 2010), and CoLin (Qingyun et al. 2016) are introduced. We investigate the three algorithms on different datasets and estimate the regret. The LinUCB algorithm evaluates the reward rate of each arms as a linear function with an assumption that the arms do not share features and are distinct from each other. Although for some applications such as news recommendation the arms are not mutually exclusive as there are some articles (arms) that are similar to each other and hence, will have shared features. Thus, adapting this approach will be more beneficial for personalized recommendation. Thus, the reward payoff in Hybrid LinUCB for each arm is a linear function of shared and non-shared components. In collaborative contextual bandit algorithm, the observed payoffs on each user are evaluated based on user's neighbors. Thus, the overall bandit parameters are evaluated in a collaborative manner i.e. considering the received payoffs of all users as well as the context from their neighbors. As compared to the Linear regret bandit algorithms, CoLin algorithm achieves an exceptional reduction of upper regret bound with high probability.

As described by the pseudo-code the following approach is followed to evaluate the accumulated regret. Data pre-processing on each dataset is performed as described in the previous section.

μ_u is a set of all user vectors, and v_a is a set of all item (i.e. to be recommended) vectors. Clustering is performed on set μ_u to get M clusters centroid that is stored in C, and each centroid behaves as a user for the algorithm. It should be noted that, in case of the synthetic dataset, clustering is not done, so 'M' will be the number of users. To compare the performance of the algorithms, it is necessary to run them parallelly in the same environment and on the same data set. In the proposed approach from line 7–12 is the algorithm that runs on each user or centroid. In line 7 optimal arm and its reward is calculated and from line 8–12 is the algorithm that runs on each user to get the reward from the picked arm or item. This reward is used to calculate the regret, i.e. the difference between Optimal reward (O_r) achieved by selecting optimal arm and reward achieved by the algorithm. For each iteration, the regret is summed, and this accumulated regret for each iteration is plotted to compare the convergence.

Algorithm 1 Recommendation Approach
Input: α, dim, T, M

1: *Pre-process the dataset*
2: *Extract* μ_u *where* $u \in 1 \ldots U$
3: *Extract* v_a *where* $a \in 1 \ldots A$
4: C $\leftarrow$ **Cluster**(U,M) where U is list of feature vectors of all users.
5: **for** $t \leftarrow 1$ to T **do**
6: **for** $c \leftarrow 1$ to M **do**
7: O_a,O_r $\leftarrow$ **OptimalArm**(A,μ_c)
8: **for** *each Algorithm* **do**
9: $picked\_arm \leftarrow Algorithm(\alpha, \mu_c)$
10: $reward \leftarrow Reward(\mu_c, v_{o_a})$
11: *Update algorithm's parameters acc to algorithm*
12: $regret[algorithm] \leftarrow regret[algorithm] + (O_r - rewd)$
13: **end for**
14: **end for**
15: $accumulated\_regret[i] \leftarrow accumulated\_regret[i-1] + regret$
16: **end for**

Symbols used in the algorithm pseudocode are explained in table 1.

Results

This section presents and discusses the results obtained.

Table 1. Symbol table.

SYMBOL	MEANING
α	Exploration Parameter
M	Number of Clusters
N	Number of Users
d	Dimension of feature vector
U	Feature vector set for Users
A	Feature vector set for Articles
O_a	Optimal Reward
O_r	Optimal Arm
T	Number of Iterations
C	Set of Cluster Centroids
μ_u	User feature vector for user u
v_u	Article feature vector for article a

Experiments on synthetic dataset

The experiment is performed by taking the number of users (N) to be 10, 20, 40, the number of articles to be 1000 and fixed feature dimension to be six for both users' and articles' feature vector. For each contextual bandit algorithm, a few input parameters are varied to get the optimal set of input values. The input parameters have the optimal values when the accumulated regret corresponding to them is maximum out of all possible combinations of input parameters. From figure 1, it can be seen that the algorithm which produces the worst regret is the LinUCB algorithm for independent bandits. Hybrid LinUCB, which is LinUCB with a set of hybrid linear models, produces better regret than LinUCB. CoLin algorithm produces the best regret among all contextual bandit algorithms. Figure 2 compares the accuracy of bandit parameters learned from different algorithms. LinUCB and hybrid LinUCB cannot directly estimate correct bandit parameters for all users. But Colin can estimate these parameters for all users and hence produces the least L2 difference

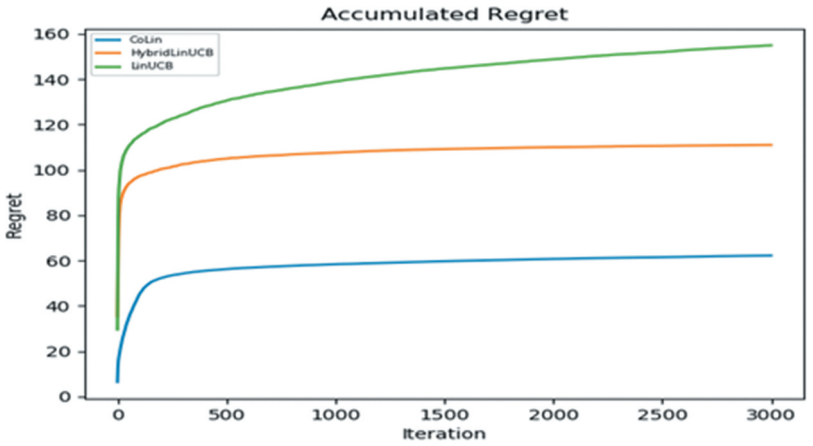

Figure 1. Convergence of cumulated regret.

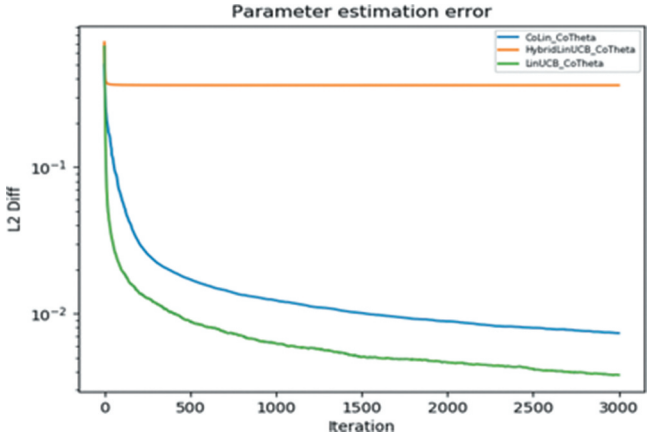

Figure 2. Accuracy of bandit parameter estimation.

Table 2. Cumulated regret for *nusers* = 20.

alpha(α)	LinUCB	HybridLinUCB	CoLin
0.3	46.2468	22.2828	11.5598
0.2	21.1141	13.4912	11.5545
0.4	74.1557	31.0022	10.6086

Table 3. Cumulated regret for *nusers* = 10.

alpha	LinUCB	HybridLinUCB	CoLin
0.3	23.4665	9.9635	5.6060
0.2	11.8230	6.3285	6.1569
0.4	36.5001	13.1408	6.8308

Table 4. Cumulated regret for *nusers* = 40.

alpha(α)	LinUCB	HybridLinUCB	CoLin
0.3	344.4860	47.5695	34.1053
0.2	47.5195	31.6257	16.3011
0.4	133.7844	69.6611	19.8591

between estimated parameters and actual parameters, as shown in figure 2. L2 difference measures the accuracy of the algorithm used. Hybrid LinUCB uses a personalized and shared model, hence produces worst L2 difference.

To obtain the best result the poolSize is chosen as 10,20 and 40. The exploration parameter α was also varied (0.2, 0.3, 0.4) as shown in tables 2, 3 and 4. All the algorithms are run up to 30000 iterations. After running the algorithms on all possible input parameter combinations, the best result is found when the exploration parameter α is 0.2 for Pool Size 20.

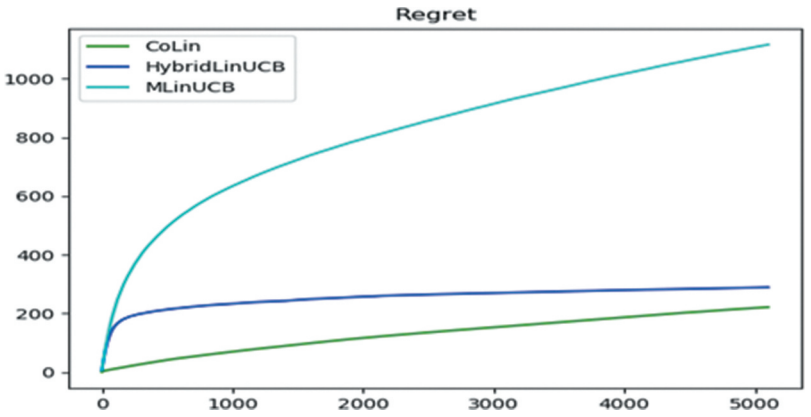

Figure 3. Accummulated regret (when k = 160).

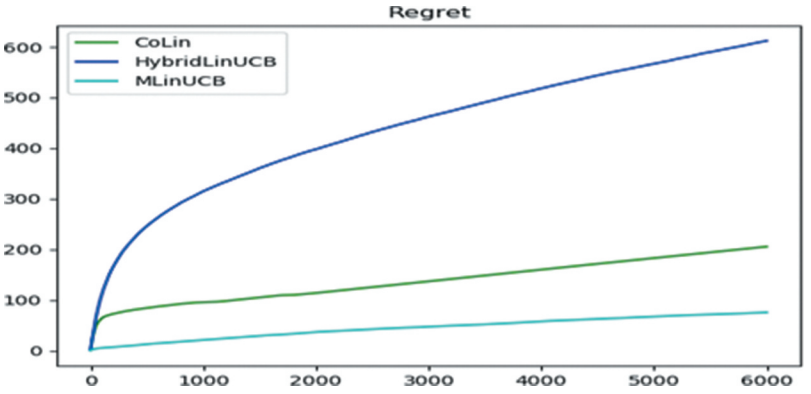

Figure 4. Accummulated regret (when k = 80).

Experiments for yahoo today module

Accumulated Regret is used to evaluate the performance of bandit algorithms. For each bandit algorithm, in figures 3 and 4, Accumulated Regret of each MAB algorithm is plotted for the number of clusters. The number of clusters, K used is 40, 80, and 160. It can be seen that CoLin outperforms all baseline bandit algorithms except for the first few iterations. Since CoLin incorporates collaborative preference learning, it controlled the exploration-exploitation trade-off much better than other bandit algorithms and timely recognized the item popularity changes. However, CoLin algorithm has a limitation; it has high computational complexity because of the global matrix formation. So, the running time of the CoLin algorithm quadratically scales with the number of users (or user groups). Thus, CoLin may not suit well in practical applications in which there are a large number of users, which can be seen in figure 4. The cumulated regret for each contextual bandit algorithm on different values of the number of clusters is reported in table 5.

Table 5. Cumulated regret.

No. of Clusters	LinUCB	HybridLinUCB	CoLin
40	344.4860	47.5695	34.1053
80	46.2468	22.2828	11.5598
160	74.1557	31.0022	10.6086

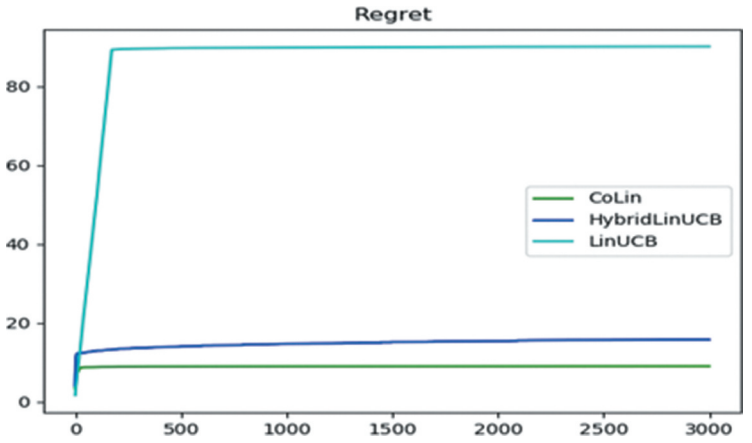

Figure 5. Convergence of accumulated regret.

Experiments on LastFM

In this experiment, the number of clusters (M) chosen is 50, 100, 200. In the dataset, changing M does not produce any significant change in the results. Dataset is used on three different algorithms LinUCB, hybrid LinUCB, and CoLin. All these algorithms run on 1892 users and 17632 artists of LastFM. The accumulated regret is collected over 3000 iterations, and cumulated regret is plotted for all the algorithms. Figure 5 shows that CoLin outperforms other two on LastFM datasets. The reason for such result is that since LinUCB assumes the users are independent and there is no shared information among them, the performance of LinUCB, not changes under the cold start and the warm start settings. While in CoLin, because of the collaboration among users, the information is propagated among users. Table 6 presents the cumulated regret for M = 200.

Experiments on MovieLens20M

Since this dataset contains a large number of users, it is not feasible to run bandit algorithms corresponding to each user. Therefore, clustering was performed on the 138493 users, and similar users are kept in one group/cluster. To determine the optimal value of K, it is varied in the range of 50

Table 6. Cumulated regret.

LinUCB	HybridLinUCB	CoLin
90.04692	15.70729	8.950207

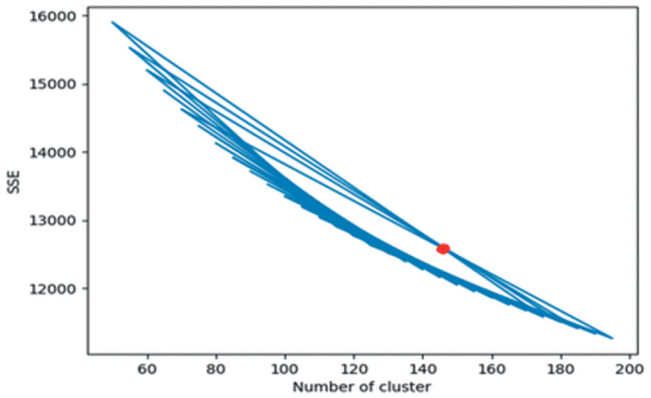

Figure 6. Optimum value of K in range (50 to 200).

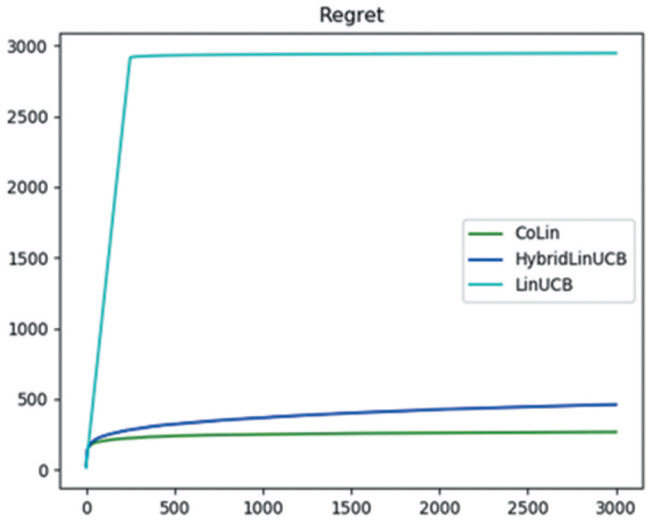

Figure 7. Convergence of accumulated regret for alpha = 0.2.

to 200. A graph is plotted between K and squared sum error/(SSE). Optimal K is the point at which the graph starts converging, which comes out to be 150 as per figure 6. After these users are clustered into 150 clusters and centroids from each cluster is now considered as a new user. For the experiment, the values of the input parameters, i.e. exploration parameter, are varied. To evaluate the performance of bandit algorithms, Cumulated Regret is used. Figure 7, figure 8, and table 7 shows that at a = 0.2, 0.3, CoLin produces the

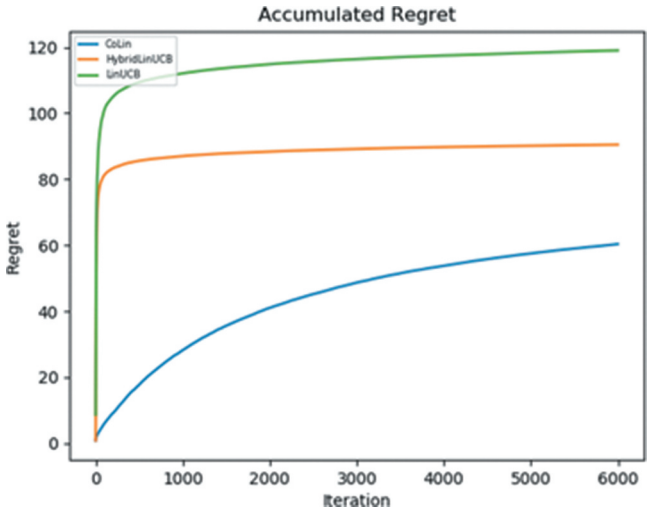

Figure 8. Convergence of accumulated regret for alpha = 0.3.

Table 7. Table title 6 regret at $k = 150$.

alpha	LinUCB	HybridLinUCB	CoLin
0.2	2947.24	460.7695	266.7983
0.3	119.01	90.48	60.34

least cumulated regret among all bandit algorithms, which implies that for this experiment also the CoLin outperformed LinUCB and Hybrid LinUCB.

Conclusion

This paper presents a comparative analysis of existing personalized recommendation techniques such as LinUCB, Hybrid-LinUCB, and CoLin. Each algorithm is implemented on structurally different datasets, i.e. Synthetic, Yahoo FrontPage Today Module, LastFM, and MovieLens20M. In personalized news recommendation, it has been observed that this dataset consists of a large volume of news articles that required efficient pre-processing techniques. The least cumulated regret reported is 34.10 using CoLin. The application of these techniques on LastFM and MovieLens20M is different from the Yahoo Today module method. In this dataset, feature vectors are pre-processed for all users and articles. In contrast, LastFM and MovieLens20M are raw datasets, so the pre-processing techniques have been modified. In these datasets, creating the best feature vectors for users, songs, and movies is primarily essential. After the implementation and execution of these algorithms on the above datasets, it has been observed that the CoLin outperforms Hybrid-LinUBC and LinUCB, reporting cumulated regret of 8.950 for LastFm and 60.34 for MovieLens20M.

The application of these algorithms on different datasets suggests these techniques can design a better-personalized recommendation system.

ORCID

Akshay Pilani http://orcid.org/0000-0002-6869-0277
Himanshu Agrawald http://orcid.org/0000-0001-6996-2521
Deeksha Chandola http://orcid.org/0000-0003-3654-0351
Vinay Anand Tikkiwal http://orcid.org/0000-0002-9165-7635
Arun Kumar http://orcid.org/0000-0002-4309-3396

References

Abdi, H., and L. J. Williams. 2010. Principal component analysis. *Wiley Interdisciplinary Reviews: Computational Statistics* 2 (4):433–59. doi:10.1002/wics.101.

Ben Schafer, J., D. Frankowski, J. Herlocker, and S. Sen. 2007. Collaborative filtering recommender systems. In *The adaptive web*, 291–324. Springer.

Binucci, C., F. De Luca, E. D. Giacomo, G. Liotta, and F. Montecchiani. 2017. Designing the content analyzer of a travel recommender system. *Expert Systems with Applications* 87:199–208. doi:10.1016/j.eswa.2017.06.028.

Burke, R. 2002. Hybrid recommender systems: Survey and experiments. *User Modeling and User-adapted Interaction* 12 (4):331–70. doi:10.1023/A:1021240730564.

Cantador, I., P. Brusilovsky, and T. Kuflik. Movielens dataset, 2011.

Chu, W., L. Lihong, L. Reyzin, and R. Schapire. Contextual bandits with linear payoff functions. In Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, 208–214, JMLR Workshop and Conference Proceedings, 2011

Errami, M., J. D. Wren, J. M. Hicks, and H. R. Garner. 2007. etblast: A web server to identify expert reviewers, appropriate journals and similar publications. *Nucleic Acids Research* 35 (suppl_2):W12–W15. doi:10.1093/nar/gkm221.

Guo, G., J. Zhang, and N. Yorke-Smith. 2015. Leveraging multiviews of trust and similarity to enhance clustering-based recommender systems. *Knowledge-Based Systems* 74:14–27. doi:10.1016/j.knosys.2014.10.016.

Inan, E., F. Tekbacak, and C. Ozturk. 2018. Moreopt: A goal programming based movie recommender system. *Journal of Computational Science* 28:43–50. doi:10.1016/j.jocs.2018.08.004.

Kiran, R., P. Kumar, and B. Bhasker. 2020. Dnnrec: A novel deep learning based hybrid recommender system. *Expert Systems with Applications* 144:113054.

Lihong, L., W. Chu, J. Langford, and R. E. Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, 661–70, 2010.

Lops, P., M. De Gemmis, and G. Semeraro. 2011. Content-based recommender systems: State of the art and trends. In *Recommender systems handbook*, 73–105. Springer.

Meng, S., Q. Lianyong, L. Qianmu, W. Lin, X. Xiaolong, and S. Wan. 2019. Privacy-preserving and sparsity-aware location-based prediction method for collaborative recommender systems. *Future Generation Computer Systems* 96:324–35. doi:10.1016/j.future.2019.02.016.

Mustaqeem, A., S. M. Anwar, and M. Majid. 2020. A modular cluster based collaborative recommender system for cardiac patients. *Artificial Intelligence in Medicine* 102:101761.

Pradhan, T., and S. Pal. 2019. A hybrid personalized scholarly venue recommender system integrating social network analysis and contextual similarity. In Future Generation Computer Systems 110 (2020): 1139–1166

Qaiser, S., and R. Ali. 2018. Text mining: Use of tf-idf to examine the relevance of words to documents. *International Journal of Computer Applications* 181 (1):07. doi:10.5120/ijca2018917395.

Qian, X., H. Feng, G. Zhao, and T. Mei. 2013. Personalized recommendation combining user interest and social circle. *IEEE Transactions on Knowledge and Data Engineering* 26 (7):1763–77. doi:10.1109/TKDE.2013.168.

Qian, Y., Y. Zhang, M. Xiao, Y. Han, and L. Peng. 2019. Ears: Emotion-aware recommender system based on hybrid information fusion. *Information Fusion* 46:141–46. doi:10.1016/j.inffus.2018.06.004.

Qingyun, W., H. Wang, G. Quanquan, and H. Wang. Contextual bandits in a collaborative environment. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, 529–38, 2016.

Shun, L., F. Luo, J. Yang, G. Ranzi, and J. Wen. 2019. A personalized electricity tariff recommender system based on advanced metering infrastructure and collaborative filtering. *International Journal of Electrical Power & Energy Systems* 113:403–10. doi:10.1016/j.ijepes.2019.05.042.

Turrin, R., M. Quadrana, A. Condorelli, R. Pagano, and P. Cremonesi. 2015. 30music listening and playlists dataset. In 9th ACM Conference on Recommender Systems, RecSys 2015. Vol. 1441. CEUR-WS, 2015

Valcarce, D., A. Landin, J. Parapar, and Á. Barreiro. 2019. Collaborative filtering embeddings for memory-based recommender systems. *Engineering Applications of Artificial Intelligence* 85:347–56. doi:10.1016/j.engappai.2019.06.020.

Wang, D., Y. Liang, X. Dong, X. Feng, and R. Guan. 2018. A content-based recommender system for computer science publications. *Knowledge-Based Systems* 157:1–9. doi:10.1016/j.knosys.2018.05.001.

Wang, H.-C., H.-T. Jhou, and Y.-S. Tsai. 2018. Adapting topic map and social influence to the personalized hybrid recommender system. In *Information Sciences*.