A Simple Multi-Armed Nearest-Neighbor Bandit for Interactive Recommendation

Javier Sanz-Cruzado
Universidad Autónoma de Madrid
javier.sanz-cruzado@uam.es

Pablo Castells
Universidad Autónoma de Madrid
pablo.castells@uam.es

Esther López
Universidad Autónoma de Madrid
esther.lopezramos@estudiante.uam.es

ABSTRACT

The cyclic nature of the recommendation task is being increasingly taken into account in recommender systems research. In this line, framing interactive recommendation as a genuine reinforcement learning problem, multi-armed bandit approaches have been increasingly considered as a means to cope with the dual exploitation/exploration goal of recommendation. In this paper we develop a simple multi-armed bandit elaboration of neighbor-based collaborative filtering. The approach can be seen as a variant of the nearest-neighbors scheme, but endowed with a controlled stochastic exploration capability of the users' neighborhood, by a parameter-free application of Thompson sampling. Our approach is based on a formal development and a reasonably simple design, whereby it aims to be easy to reproduce and further elaborate upon. We report experiments using datasets from different domains showing that neighbor-based bandits indeed achieve recommendation accuracy enhancements in the mid to long run.

CCS CONCEPTS

• Information systems → Recommender systems; Collaborative Filtering • Computing methodologies → Reinforcement learning.

KEYWORDS Multi-armed bandits; Nearest-neighbors; Interactive recommendation; Thompson sampling.

ACM Reference format:

Javier Sanz-Cruzado, Pablo Castells and Esther López. 2019. A Simple Multi-Armed Nearest-Neighbor Bandit for Interactive Recommendation. In *Proceedings of ACM RecSys'19. ACM, New York, NY, USA, 5 pages.*

1 Introduction

The cyclic nature of the recommendation task is being increasingly taken into account in recommender systems research in the last few years [7,8,9,21,23]. In many applications, a considerable fraction of the input for recommendation algorithms is obtained from the end-users' reaction and feedback to the algorithms' output, thus framing interactive recommendation as a genuine reinforcement learning problem [19]. In this view, multi-armed bandit (MAB) perspectives have been increasingly considered as a natural approach to cope with the dual role of recommendation: pleasing users in the single next recommendation (exploitation) and gaining knowledge about their tastes (exploration) in order to further improve user satisfaction in the long run [12].

In this perspective the field has adapted principles, formulations and solutions from the reinforcement learning area [20]. Some of this trend may have put an emphasis on developing machine learning techniques, taking recommendation as an example application domain [5,7,8,21]; resulting algorithms are often technically involved, computationally complex, not easy to reproduce, and/or consider simplified variants of the recommendation task. Other work has been undertaken from the perspective of solving a recommender system problem, by adapting and elaborating on machine learning tools [9,10,12,23]; our present research can be framed in this outlook.

In this paper we develop a simple multi-armed bandit elaboration of neighbor-based collaborative filtering. The approach can be seen as a variant of the nearest-neighbors scheme, but endowed with a controlled stochastic exploration capability of the users' neighborhood. By a formal development and a reasonably simple design our approach aims to be easy to reproduce and further elaborate upon.[1] Our approach works as a pure collaborative filtering algorithm that does not require any side information about items or users. Compared to both common collaborative filtering algorithms, and prior work on recommendation bandits, our method does not require any other parameters than the ones involved in the adapted bandit paradigm. We report experiments using datasets from different domains showing that our approach indeed achieves recommendation accuracy enhancements in the mid to long-run.

2 Background and Related Work

2.1 Multi-Armed Bandits

The multi-armed bandit problem [20] considers a set of actions $\mathcal{A}$ –a.k.a. arms– that one has to choose among successively. The selection of an action $a \in \mathcal{A}$ at a point in time t results in a certain reward $R_t(a) \in \mathbb{R}$, which can be summarized as a real number. The reward is not known until the arm is selected; this uncertainty is modeled as an unknown reward distribution for each arm. The goal in the bandit problem is simple: decide on a sequence of actions that maximizes the expected obtained returns $\sum_{t=1}^{n} R_t(a_t)$.

If the reward distributions of arms are stationary over time, the optimal choice would be to select the action with the maximum mean reward (so-called "arm value" in the MAB literature) all the time. While this mean is unknown, it is possible to build and update estimates based on the information gained when selecting arms. One can then make conservative decisions by opting for the best arm according to estimates based on a few initial observations. But one can also invest in trying different arms to gain information and make even better future decisions. The solutions to the problem hence have to deal with this tension between the exploitation of available knowledge for best immediate payoffs, and the exploration of the decision space for enhanced knowledge and better future rewards.

Many solutions to the bandit problem have been developed, with different properties and trade-offs. Popular methods include

RecSys'19, September 16–20, 2019, Copenhagen, Denmark.

ACM ISBN 978-1-4503-6243-6/19/09…$15.00 https://doi.org/10.1145/3298689.3347040

[1] The source code implementing all the methods and experimental procedures described in this paper are publicly available at https://github.com/ir-uam/kNNBandit.

ε-greedy [20], Upper Confidence Bounds (UCB) [1], Thompson sampling [4], EXP3 [2], and multiple variants thereof [1,20]. We have selected Thompson sampling as one of the most widely used, compact and effective bandit methods to develop our approach. Our method is however not particularly dependent on this choice, and the adoption of other bandit techniques should be straightforward, which we envision as future work.

2.2 Bandit Recommender Systems

The vision of recommendation as a reinforcement learning problem can be traced back to nearly a couple of decades ago [19]. Work in this and related directions has been growing since then; an exhaustive literature overview would exceed the scope of this paper (see e.g. [9] and [21] for good related work lists). We just summarize here the most relevant high-level trends for our present research, selecting representative literature for each aspect.

In most if not all of the work developed so far, the items to be recommended are modeled as the arms to be pulled. Selecting an arm is equivalent to recommending an item, and the reward is the user response to the recommendation (e.g. clicks, acceptance, satisfaction, etc.). An important strand of research in this area has built upon a probabilistic matrix factorization (PMF) formulation [15]. Instead of directly modeling the reward distribution, these approaches model the latent factors of users and/or items into which PMF assumes the reward can be decomposed [9,21,23]. In a similar spirit, many other approaches describe the reward structure in terms of clusters of users and/or items [5,8,9,21]. Some methods additionally rely on the availability of item metadata [5,8], and most formulations introduce a fair number of parameters that require –sometimes involved and costly– model learning or training procedures.

Our approach is different from prior work, first, in how we model recommendation as a MAB problem, and how we develop our solution. As we shall describe next, in our bandit model the arms are users instead of items. Rather than elaborating on a matrix factorization approach and/or relying on clustering techniques, we develop a very simple neighbor selection scheme. The resulting algorithm is easy to reproduce and does not involve complex supervised or unsupervised learning steps. On top of an alternative and easy formulation, our method has additional advantages: it introduces zero additional parameters to the underlying bandit algorithm, from which considerable computational complexity savings are derived. The approach does not require a training phase –except for tuning, if needed, the basic bandit subroutine parameters. It does not require any side information either, working in a purely collaborative filtering mode.

3 Nearest-Neighbor Bandits

While items are the arms to be chosen for direct recommendation in prior work, the arms in our formulation are users to be chosen as potential neighbors. We consider recommendations are to be produced by advice from a neighbor of the target user, where neighbor selection is defined as a stochastic choice, rather than by deterministic similarity (as in regular kNN), as follows.

Let $\mathcal{U}$ and $\mathcal{I}$ denote the sets of all users and items, respectively. Given a target user $u \in \mathcal{U}$ (bandit context), all other users are viewed as arms that can be picked as potential neighbors for u. Once a user $v \in \mathcal{U} \setminus \{u\}$ is selected as a neighbor, she picks an item $i \in \mathcal{I}$, which is then recommended to the target user. Neighbors pick every time the item they like most, as governed by a certain distribution $P(i|v)$. The obtained reward is then binary: it is 1 if the target user u likes this item too, and 0 if she does not. The *value* of v as neighbor of u can then be defined as the fraction of times the target user would be pleased by an item selected (liked) by the neighbor. The reward from a neighbor can be thus modeled as a Bernoulli distribution with mean $P(u|v)$.

Algorithm 1: Contextual Neighbor-Based Bandit Recommender.

Input: $r: \mathcal{R} \subset \mathcal{U} \times \mathcal{I} \to \{0,1\}$ // Rating dataset
$\alpha_0 \in \mathbb{R}, \beta_0 \in \mathbb{R}$ // Initial TS parameters
begin
  $\mathcal{R}_0 \leftarrow \emptyset$ // Observed rating set at time $t = 0$
  $\mathcal{T}_0 \leftarrow \mathcal{U} \times \mathcal{I}$ // Recommendable "target" user-item pairs
  for $v \in \mathcal{U}$ **do** $n_0(v) \leftarrow \alpha_0 + \beta_0$
  for $t \leftarrow 0$ **to** N **do**
    $u \leftarrow$ pick a target user $u \in \mathcal{U}$ (e.g. uniformly at random)
    $i_t \leftarrow$ recommend $(\mathcal{R}_t, \mathcal{T}_t, \alpha_t, n_t, u)$
    $\mathcal{T}_{t+1} \leftarrow \mathcal{T}_t \setminus \{(u, i_t)\}$ // Do not recommend i_t to u again
    if $(u, i_t) \in \mathcal{R} \setminus \mathcal{R}_t$ **then**
      $\mathcal{R}_{t+1} \leftarrow \mathcal{R}_t \cup \{(u, i_t)\}$
      $n_{t+1}(u) \leftarrow n_t(u) + r(u, i_t)$
      for $v \in \mathcal{U}: (v, i_t) \in \mathcal{R}_t$ **do**
        $\alpha_{t+1}(v|u) \leftarrow \alpha_t(v|u) + r(u, i_t)\, r(v, i_t)$
end

Function recommend $(\mathcal{R}_t, \mathcal{T}_t, \alpha_t, n_t, u)$
begin
  for $v \in \mathcal{U} \setminus \{u\}$ **do**
    $p_t(u|v) \leftarrow$ draw from Beta$\big(\alpha_t(v|u), n_t(v) - \alpha_t(v|u)\big)$
  $v_t \leftarrow \arg\max_{v \in \mathcal{U} \setminus \{u\}} p_t(u|v)$
  return $\arg\max_{i \in \mathcal{I}: (u,i) \in \mathcal{T}_t, (v_t, i) \in \mathcal{R}_t} r(v_t, i)$
end

Considering binary rewards, we represent user preference observations as binary ratings $r: \mathcal{R} \subset \mathcal{U} \times \mathcal{I} \to \{0,1\}$, where $\mathcal{R}$ is a set of observations. Using a Thompson sampling approach [4], in order to produce a recommendation at time t we draw an estimate $p_t(u|v)$ –with lowercase p– of the unknown Bernoulli parameter $P(u|v)$ (the arm value) for each neighbor v. We draw this estimate from a Beta posterior with the following parameters (successes α_t and failures β_t) for each neighbor v given a target user u:

$$\alpha_t(v|u) = \alpha_t(u|v) = \sum_{j \in \mathcal{J}} r(u, j)\, r(v, j)$$
$$\beta_t(v|u) = \sum_{j \in \mathcal{J}} r(v, j)\big(1 - r(u, j)\big) = n_t(v) - \alpha_t(v|u)$$

where $n_t(v) = \sum_{j \in \mathcal{J}} r(v, j)$, and $\alpha_t(v|u)$ is simply the number of items u and v like in common. We then select the neighbor v_t with highest sampled value $p_t(u|v_t)$, and then we pick the item i that v_t likes most —ties broken at random– based on $P_t(i|v) \propto r(v, i)$, as the one to be recommended. Every time an item is recommended to some user, all neighbor arms –the $\alpha_t(v|u)$ and $\beta_t(v|u)$ parameters– are updated with the user's reaction to the item, and the cycle goes on. We shall assume, as a most usual scenario for our experiments, that items should not be recommended twice.

A detailed stepwise description of this bandit recommendation approach is shown in Algorithm 1. Note that since α_t is a symmetric function, we need only keep and update a single α_t value for each pair of users. Likewise, β_t can be derived from α_t and n_t.

A straightforward generalization of our bandit scheme is possible, where we use not just one but k neighbors. This can be achieved by simply selecting the set $N_t^k[u]$ of k users that maximize $p_t(u|v)$ –rather than just one v_t as in line 20 of Algorithm 1. And then, each of these neighbors can have a weighted vote on the item to recommend, which would then be selected as $i_t \leftarrow \arg\max_i \sum_{v \in N_t^k[u]} p_t(u|v)\, r(v,i)$ –instead of just one vote as in line 22 of Algorithm 1. This represents a multiple play bandit that selects several arms per hand [10], and our basic approach is a particular case with $k = 1$. In our experiments we have found that taking $k > 1$ would appear to be less effective in general, but we also found exceptions as we shall describe, and the generalized version might be an interesting direction to explore in the future.

The neighbor bandit algorithm has equivalent computational complexity to item-based Thompson sampling: the latter has a cost of $|\mathcal{I}| \cdot b$ per time step, where b represents the (considerable) cost of sampling from a Beta distribution. The complexity of our neighbor bandit scheme (in its plain $k = 1$ version) is $\sim |\mathcal{U}| \cdot b$, as the neighbor update operations are negligible in comparison to b. As to the regular kNN scheme, for the optimal settings of k the resulting cost was similar to that of our approach in our experiments. We have also observed that Thompson sampling (and hence our approach) is orders of magnitude faster than common matrix factorization approaches for exploitation-oriented collaborative filtering [6,14].

4 Experiments

We test our approach with offline data to confirm the basic empirical effectiveness of the neighbor bandits scheme, checking that it improves over plain kNN, and simple item-oriented bandit methods. Our evaluation procedure consists in the simulation of a recommendation loop where the arms are iteratively updated by the available feedback on the recommended items. Using an offline rating dataset, and similarly to e.g. [21], we start the simulation with an empty set of observations, and grow it using the dataset to simulate user feedback. The experimental procedure is described in detail by Algorithm 1 (with random user sampling in the main loop –line 8). As an evaluation metric we use the incremental ratio of discovered positive preferences, which can be seen as a global cumulative recall metric (transversal to the set of users), and is equivalent to the cumulative returns in bandit terms. The metric is informative as a function of time: an algorithm is effective the faster it makes recall grow in the recommendation loop.

4.1 Data

We test our approach on data with different density, type of feedback, and domains: movies, venues, and social networks, the details for which are given in Table 1. The datasets include:

- Foursquare check-ins in New York City and Tokyo, made available by Yang & Zhang [22].
- Two sets of Twitter data described in [18], built by snowball-sampling exploration of the interaction network from a seed user. The "follow" links between the collected set of users play the part of ratings for a contact recommendation task [17].
- MovieLens 1M [11], where we binarize the ratings taking as positive the values equal or higher to 4.

4.2 Algorithms

Along with our neighbor bandit approach, we include the following alternatives:

Table 1: Dataset details.

	#Users	#Items	#Ratings
Foursquare New York	1,083	38,333	227,428
Foursquare Tokyo	2,293	61,858	573,703
Twitter 1 month	9,511	9,511	650,937
Twitter 200 tweets	9,253	9,253	475,608
MovieLens 1M	6,040	3,706	1,000,209

- Plain exploitation-oriented collaborative filtering algorithms: a user-based kNN algorithm with cosine similarity [13], and matrix factorization for implicit data (implicit MF) [6,14].
- Two basic item-oriented bandit algorithms: ε-greedy [20] and Thompson sampling [4]. We have tested further algorithms such as UCB [1], with similar results.
- As a frame of reference, two non-personalized recommendations: most popular item, and random recommendation.

We set the algorithm parameters by grid search. For ε-greedy, we try $\varepsilon = 0.1$ to 1 by steps of 0.1, plus $\varepsilon = 0.05$. For Thompson sampling we set $\alpha_0 = 1$ and try powers of 10 for β_0 (1, 10, 100). For the number of neighbors k of kNN we likewise try powers of 10 up to 1,000. For matrix factorization [6,14], we take competitive values reported in the literature for this algorithm on each dataset: MovieLens 1M [3], Foursquare [16], and Twitter data [17]. Table 3 shows all the resulting parameter settings.

4.3 Results

Fig. 1 shows the comparison of the tested approaches, in a time interval where users have been delivered about 500 item recommendations on average. As an overall observation we can see that the neighbor-based bandit is better than all the tested alternatives in most cases and perspectives. But our results show further interesting insights that we discuss next.

4.3.1 Cumulative Gain. Plain collaborative filtering algorithms (kNN and implicit MF) have a poor start after which they slightly improve gradually. This is a natural consequence of their inability to deal with a cold start. After running for long enough (well beyond the time intervals we show in the graphs), these myopic algorithms may catch up with the bandit approaches –possibly too late to avoid user abandonment in a real application. In the Foursquare datasets, this recovery never even happens because of the low data density per item: MF is below non-personalized popularity, and kNN does not even work at all (even for very high values of k that we tried). It is quite noteworthy that an alternative (bandit) kNN scheme is able not just to function but to outperform all the tested approaches in these harsh conditions.

Popularity is among the best options in the very few initial steps (hardly visible in the graphics), after which it stagnates. This is to be expected: when barely any information is available, aggregating it is the best one can do.

The item-based bandits (Thompson sampling and ε-greedy) sometimes start out better than the neighbor bandit at the first few steps, but this is reversed very early. It is worth noting that the substantial effectiveness improvement by our approach is achieved *without introducing any additional parameters* with respect to the item-arm bandits. The neighbor approach seems to better capture structure in the data without incurring in any further parameter setting burden or computational cost.

The dataset where our approach achieves the lesser advantage is MovieLens. We can attribute this to two reasons: first, the MF

Table 2: Parameter settings of compared algorithms.

	FourSquare New York	FourSquare Tokyo	Twitter 1 month	Twitter 200 tweets	MovieLens 1M
Bandit kNN	$\alpha_0 = 1, \beta_0 = 10$	$\alpha_0 = 1, \beta_0 = 100$	$\alpha_0 = 1, \beta_0 = 100$	$\alpha_0 = 1, \beta_0 = 100$	$\alpha_0 = 1, \beta_0 = 100$
Th. sampling	$\alpha_0 = 1, \beta_0 = 100$	$\alpha_0 = 1, \beta_0 = 100$	$\alpha_0 = 1, \beta_0 = 100$	$\alpha_0 = 1, \beta_0 = 100$	$\alpha_0 = 1, \beta_0 = 100$
ε-greedy	$\varepsilon = 0.1$	$\varepsilon = 0.2$	$\varepsilon = 0.1$	$\varepsilon = 0.1$	$\varepsilon = 0.05$
Implicit MF	$k = 10, \alpha = 10, \lambda = 10$	$k = 10, \alpha = 10, \lambda = 10$	$k = 270, \alpha = 40, \lambda = 150$	$k = 270, \alpha = 40, \lambda = 150$	$k = 20, \alpha = 1, \lambda = 0.1$
kNN	$k = 100$	$k = 100$	$k = 100$	$k = 100$	$k = 100$

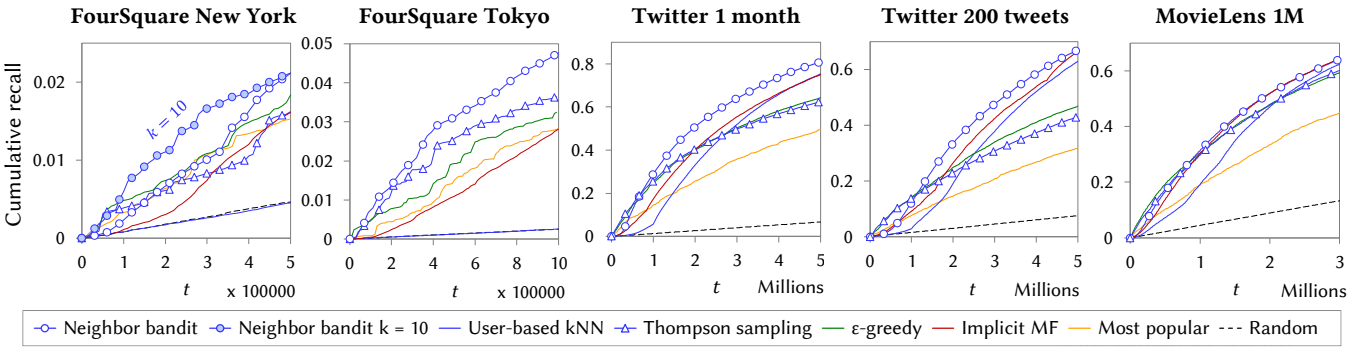

Figure 1: Cumulative recall over time for the compared recommendation approaches.

algorithm might be able to take advantage of the negative ratings available in this dataset [6]; this may suggest that our approach is better suited for implicit than explicit feedback. Second, MovieLens is about ten times denser than the other datasets, whereby the algorithms might learn ten times faster per time step. We may just be observing in the graphic ten times ahead in the future, where myopic algorithms are expected to eventually improve.

We have also tested the multiple play version of our algorithm (see section 3), but it yields inferior results in general, whereby we omit it in Fig. 1. The only exception is Foursquare New York, where we show a multiple play bandit with $k = 10$ that is substantially better than the basic version.

4.3.2 Exploration vs. Exploitation. The neighbor bandit seems to find a good balance between exploration and exploitation compared to the other alternatives. Fig. 2 gives an idea of the degree of exploration of the different methods, by measuring the Gini index of the cumulative frequency by which items have been recommended.

The least explorative approach is, naturally, recommendation by popularity (superimposed with a multiple play neighbor bandit with $k = 100$), and the most exploratory option is random recommendation. We see that Thompson sampling and ε-greedy seem

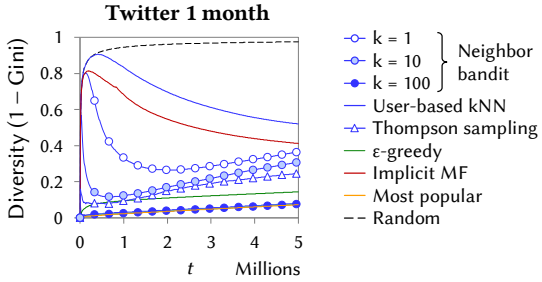

Figure 2: Aggregate recommendation diversity on (as an example) Twitter 1 month, measured as (one minus) the Gini index of the recommendation frequency distribution over items. Similar patterns are seen in the other datasets.

leant towards exploitation –more exploratory (optimistic) configurations are possible but yield inferior results in cumulative recall. Note that Gini is useful to reveal exploration but is not *the same thing* as exploration. While exploration causes item diversity, an algorithm might try hard to exploit its available knowledge and yet end up delivering very different items to each user. This is the case of myopic collaborative filtering algorithms, which would seem to have a very exploratory start. We can safely attribute this to the erratic behavior caused by data scarcity; as more data becomes available, their exploitative nature becomes increasingly apparent.

Interestingly, we have also observed that increasing the number of plays in our neighbor bandit results in a reduction of exploration. We see in Fig. 2 that the larger the neighborhood size k, the more our algorithm concentrates recommendations over a few popular items. Apparently, this restraint on exploration has a positive effect in the Foursquare New York dataset.

5 Conclusions

We have developed a novel multi-armed neighbor-based bandit approach that achieves effective collaborative filtering when recommendation is understood to be an interactive process with a feedback loop. In comparison to the common kNN scheme, our approach can be described as being sensitive to the uncertainty in the available observations of user-user preference similarity, modeling this uncertainty in a well-established stochastic scheme. As a result, our approach explores user neighborhoods further than the basic kNN algorithm does. Moreover, the selection of a single neighbor results in a compact algorithm that appears to be, in our experiments, generally better than the use of multiple neighbors. Salient properties we may stress in our approach include simplicity and economy in parameters, requirements, design complexity, and computational cost.

ACKNOWLEDGMENTS

This work was supported by the Spanish Government (grant nr. TIN2016-80630-P).

REFERENCES

[1] P. Auer, N. Cesa-Bianchi and P. Fischer (2002). Finite-time Analysis of the Multiarmed Bandit Problem. *Machine Learning*, 47 (May 2002), 235–256.

[2] P. Auer, N. Cesa-Bianchi, Y. Freund and R. E. Schapire (2003). The nonstochastic multiarmed bandit problem. *SIAM journal on computing*, 32, 1 (January 2003), 48–77.

[3] R. Cañamares and P. Castells (2017). A Probabilistic Reformulation of Memory-Based Collaborative Filtering – Implications and Popularity Biases. In *Proceedings of the 40th Annual International Conference on Research and Development in Information Retrieval* (SIGIR 2017). ACM, New York, NY, USA, 215–224.

[4] O. Chapelle and L. Li (2011). An empirical evaluation of Thompson Sampling. In *Proceedings of Neural Information Processing Systems* (NIPS 2011). Curran Associates, Inc., Red Hook, NY, USA, 2249–2257.

[5] C. Gentile, S. Li and G. Zappella (2014). Online Clustering of Bandits. In *Proceedings of the 31st International Conference on Machine Learning* (ICML 2014). Proceedings of Machine Learning Research, Sheffield, UK, 757–765.

[6] Y. Hu, Y. Koren and C. Volinsky (2008). Collaborative Filtering for Implicit Feedback Datasets. In *Proceedings of the 8th IEEE International Conference on Data Mining* (ICDM 2008). IEEE Computer Society, Washington, DC, USA, 15–19.

[7] J. Kawale, H. H. Bui, B. Kveton, L. Tran-Thanh and S. Chawla (2015). Efficient Thompson Sampling for Online Matrix-Factorization Recommendation. In *Proceedings of Neural Information Processing Systems* (NIPS 2015). Curran Associates, Inc., Red Hook, NY, USA, 1297–1305.

[8] L. Li, W. Chu, J. Langford and R. Schapire (2010). A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th International Conference on World Wide Web* (WWW 2010). ACM, New York, NY, USA, 661–670.

[9] S. Li, A. Karatzoglou and C. Gentile (2016). Collaborative Filtering Bandits. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval* (SIGIR 2016). ACM New York, NY, USA, 539–548.

[10] J. Louëdec, M. Chevalier, J. Mothe, A. Garivier and S Gerchinovitz (2015). A Multiple-Play Bandit Algorithm Applied to Recommender Systems. In *Proceedings of the 28th International Florida Artificial Intelligence Research Society Conference* (FLAIRS 2015). AAAI Press, Menlo Park, CA, USA, 67–72.

[11] F. M. Maxwell and J. A. Konstan (2015). The MovieLens Datasets: History and Context. *ACM Transactions on Interactive Intelligent Systems*, 5, 4 (December 2015).

[12] J. McInerney, B. Lacker, S. Hansen, K. Higley, H. Bouchard, A. Gruson and R. Mehrotra (2018). Explore, exploit, and explain: personalizing explainable recommendations with bandits. In *Proceedings of the 12th ACM Conference on Recommender Systems* (RecSys 2018). ACM, New York, NY, USA, 31–39.

[13] X. Ning, C. Desrosiers and G. Karypis (2015). A Comprehensive Survey of Neighborhood-Based Recommender Systems. In: F. Ricci, L. Rokach and B. Shapira (Eds.), *Recommender Systems Handbook* (2nd ed.). Springer, New York, NY, USA, 37–76.

[14] I. Pilászy, D. Zibriczky and D. Tikk (2010). Fast ALS-based Matrix Factorization for Explicit and Implicit Feedback Datasets. In *Proceedings of the 4th ACM Conference on Recommender Systems* (RecSys 2010). ACM, New York, NY, USA, 71–78.

[15] R. Salakhutdinov and A. Mnih (2007). Probabilistic matrix factorization. In *Proceedings of Neural Information Processing Systems* (NIPS 2011). Curran Associates, Inc., Red Hook, NY, USA, 1257–1264.

[16] P. Sánchez and A. Bellogín (2018). A novel approach for venue recommendation using cross-domain techniques. In *Proceedings of the 2nd* Workshop on Intelligent Recommender Systems by Knowledge Transfer and Learning (RecSysKTL) at the 12th ACM Conference on Recommender Systems (RecSys 2018). ACM, New York, NY, USA, 260–268.

[17] J. Sanz-Cruzado and P. Castells (2018). Contact Recommendations in Social Networks. In: I. Cantador, S. Berkovsky, D. Tikk (Eds.), *Collaborative Recommendations: Algorithms, Practical Challenges and Applications*. World Scientific Publishing, Singapore, 2018, 519–569.

[18] J. Sanz-Cruzado and P. Castells (2018). Enhancing Structural Diversity in Social Networks by Recommending Weak Ties. In *Proceedings of the 12th ACM Conference on Recommender Systems* (RecSys 2018). ACM, New York, NY, USA, 233–241.

[19] G. Shani, D. Heckerman and R. I. Brafman (2005). An MDP-Based Recommender System. *Journal of Machine Learning Research* 6 (December 2005), 1265–1295.

[20] R. Sutton and A. Barto (2018). *Reinforcement Learning: An Introduction* (2nd ed.). MIT Press, Cambridge, MA, USA, 2018.

[21] Q. Wang, C. Zeng, W. Zhou, T. Li, S. S. Iyengar, L. Shwartz and G. Grabarnik (2019). Online Interactive Collaborative Filtering Using Multi-Armed Bandit with Dependent Arms. *IEEE Transactions on Knowledge and Data Engineering*, 31, 8 (August 2019), 1569–1580.

[22] D. Yang, D. Zhang, V. W. Zheng and Z. Yu (2015). Modeling User Activity Preference by Leveraging User Spatial Temporal Characteristics in LBSNs. *IEEE Transactions on Systems, Man and Cybernetics: Systems*, 45, 1 (January 2015), 129–142.

[23] X. Zhao, W. Zhang and J. Wang (2013). Interactive Collaborative Filtering. In *Proceedings of the 22nd ACM International Conference on Information and Knowledge Management* (CIKM 2013). ACM, New York, NY, USA, 1411–1420.