User Cold-start Problem in Multi-armed Bandits: When the First Recommendations Guide the User's Experience

NICOLLAS SILVA, Universidade Federal de Minas Gerais, Brazil
THIAGO SILVA, HEITOR WERNECK, and LEONARDO ROCHA, Universidade Federal de São João del-Rei, Brazil
ADRIANO PEREIRA, Universidade Federal de Minas Gerais, Brazil

Nowadays, Recommender Systems have played a crucial role in several entertainment scenarios by making personalised recommendations and guiding the entire users' journey from their first interaction. Recent works have addressed it as a Contextual Bandit by providing a sequential decision model to explore items not tried yet (or not tried enough) or exploit the best options learned so far. However, this work noticed these current algorithms are limited to naive non-personalised approaches in the first interactions of a new user, offering random or most popular items. Through experiments in three domains, we identify a negative impact of these first choices. Our study indicates that the bandit performance is directly related to the choices made in the first trials. Then, we propose a new approach to balance exploration and exploitation in the first interactions and handle these drawbacks. This approach is based on the Active Learning theory to catch more information about the new users and improve their long-term experience. Our idea is to explore the potential information gain of items that can also please the user's taste. This method is named Warm-Starting Contextual Bandits, and it statistically outperforms 10 benchmarks in the literature in the long run.

CCS Concepts: • **Information systems** → **Recommendation systems;**

Additional Key Words and Phrases: Recommender Systems, multi-armed bandits, user cold-start

ACM Reference format:
Nicollas Silva, Thiago Silva, Heitor Werneck, Leonardo Rocha, and Adriano Pereira. 2023. User Cold-start Problem in Multi-armed Bandits: When the First Recommendations Guide the User's Experience. *ACM Trans. Recomm. Syst.* 1, 1, Article 2 (January 2023), 24 pages.
https://doi.org/10.1145/3554819

1 INTRODUCTION

After the past three decades of research and advances, **Recommendation Systems (RSs)** have assumed a crucial role in several online services of commerce and entertainment [20, 33]. By providing personalised recommendations, i.e., identifying the most relevant items (e.g., movies, books, songs, etc.) according to the user's preferences, the RSs have directly contributed to maximising the user's engagement, avoiding customer churn, and increasing the company profit over

This work was partially supported by CNPq, CAPES, Fapemig, AWS and INWEB.
Authors' addresses: N. Silva and A. Pereira, Universidade Federal de Minas Gerais, Belo Horizonte, Brazil; emails: {ncsilvaa, adrianoc}@dcc.ufmg.br; T. Silva, H. Werneck, and L. Rocha, Universidade Federal de São João del-Rei, São João del-Rei, Brazil; emails: {thiagosilva, werneck}@aluno.ufsj.edu.br, lcrocha@ufsj.edu.br.

2770-6699/2023/01-ART2 $15.00
https://doi.org/10.1145/3554819

the years [19]. In recent reports, most online companies have recognised the business value of their recommendation engine. Netflix announced that its recommendation and personalisation engines had produced a business value higher than 1 billion US dollars per year [12]. Similarly, Amazon and YouTube reported that RSs have been directly responsible for generating 35% of sales and 60% of the clicked videos in their systems [7].

Nowadays, the recommendation systems have also become responsible for guiding the entire users' journey from their first interactions [42, 44, 48]. In real-world scenarios, an RS must continually choose one item (or a set of items), receive the user's feedback, and update itself at each iteration. Recent research have addressed this task as a sequential decision model in a **Multi-Armed Bandit (MAB)** problem [3, 13, 40, 43]. MAB is a classic problem from the Reinforcement Learning theory in which a fixed limited set of resources (arms) must be selected to maximise the expected gain (reward). At each trial, the system must decide which arms to select, which order to select, and whether to continue with it or try a different option. For this reason, MAB usually faces the dilemma between (1) *exploiting* the arm that seems the best option so far or (2) *exploring* an arm not tried out yet (or not tried enough). *Exploitation* has notions of *being greedy* and a high probability of bringing a faster payback but can introduce a regret of missing unexplored opportunities. In turn, *exploration* usually has notions of *gaining info*, but it can take a long time to ensure enough knowledge for the model and may introduce regret by wasting time on failures. In general, the optimal solution is achieved by combining these *information* and *greedy* gains.

In the recommendation field, items are usually modelled as the arms to be pulled and the reward is the user's feedback on that recommendation (e.g., clicks, acceptance, satisfaction, etc.) [30]. Selecting an arm is equivalent to recommending an item to a system user. Then, while exploitation means selecting items with a high probability of being rated, exploration means recommending different items in an attempt to gain more information about users [30, 47]. Again, the challenge is in how to overcome the dilemma of exploration and exploration. However, the system should now perform it while providing personalised items to the users. For this reason, current solutions have handled personalised MAB by implementing *Contextual Bandit* models that assume a linear representation among users, items, and rewards [42, 45]. Thus, the bandit model has worked individually for each user and provided personalised recommendations that are updated according to each feedback received [23, 38, 43, 47]. In this case, users and items are modelled as feature vectors by representing their information into z dimensions of interest. The idea is similar to those applied by traditional Matrix Factorisation techniques but implemented in an online environment where these features are continually updated at each iteration [24, 38].

However, despite the current advances, there still are several limitations in the applicability of these contextual bandit solutions over the entire user's journey. In scenarios like the *pure cold-start* problem, when a new user has just arrived on the platform, it is not possible to define the utility of any item (i.e., arm) for him/her and it is even harder to mitigate the exploration/exploitation challenge. There is no information to initialise the users' features vector used to represent their interests for each dimension z of the system. Hence, as shown by this work, most linear contextual bandit models have performed as naive non-personalised RSs and made a pure-exploitation of the information known a prior (similar to a most popular recommendation) or a pure-exploration of random items. Only a few models, based on Bayesian inference (e.g., Thompson Sampling algorithms) [46], may indirectly combine these concepts of exploration and exploitation since the first user's interactions. However, they do not explore smart approaches to improve the system knowledge. They are simple random approaches over the item's distribution known a prior, like a random recommendation of the most popular items.

In this sense, this work raises three main research questions:

RQ1: Are these naive non-personalised approaches enough to ensure the requirable knowledge for a linear bandit algorithm in the first interactions of new users?
RQ2: How do get more knowledge about the new users in their first interactions?
RQ3: What is the impact of a new strategy that gets more information about the new users in the first interactions of linear bandit models?

Although the applicability of non-personalised recommendations seems reasonable for new users, we identify a negative impact of *naive approaches* by answering the RQ1. Applying a pure-exploration approach (methods like random or entropy-based) in the first interactions of a new user can improve his/her long-term satisfaction, but it requires at least three more times until making personalised recommendations. In turn, applying a pure-exploitation approach (methods like most popular or best-rated items) often recommend potentially relevant items for users at the first recommendations. However, these items do not add much knowledge to the model, because people usually like them—they are the most popular items. The result is a bandit model with high precision in the first interactions that cannot maximise the accuracy in the long run.

Then, we propose to answer the RQ2 by addressing both exploration and exploitation goals to overcome the drawbacks of such naive approaches. Our idea is to apply exploitation to achieve relevance and exploration to get more information in the new users' first interactions. As the system does not know anything about the new user, we explore the current information available about the items: popularity and entropy. Popularity is the number of distinct users who rated the item, and entropy is the variation of rating values received by each item. While using popularity means increasing the probability that a user will rate an item (a notion of exploitation), applying entropy means increasing the amount of information that can be achieved if the user rates the item (a notion of exploration). Both features are deeply exploited in the Active Learning theory, where the system must get the most useful items to improve its accuracy in future actions. We use them to generate the first recommendations for new users and get the requirable knowledge for such linear bandit models. Contrasting the other two approaches of pure-exploration and pure-exploitation aforementioned, this new one can identify the user's preferences quickly and introduce significant improvements in the user's long-term experience.

These previous results lead us to assume that such changes in the first interactions may positively affect the user's experience in the long run. Then, we propose a new method to make a **Warming-Start of Contextual Bandits (WSCB)** and analyse its performance to answer the RQ3. This method uses a non-personalised Active Learning approach to initialise the features vector of new users (unknown in the first interaction). Similarly, as mentioned before, we apply the combination of entropy and popularity to handle the first user's interactions in a linear bandit algorithm. The idea is to guide linear bandit models to choose items that will reach enough information about the new user in the first interactions and then produce an impact in the long run. In our opinion, there is a large room to be explored when new users join the system, because their expectation for high-quality personalised results in the first interactions tends to be smaller. Indeed, contrasting the WSCB with the most recent approaches of Contextual Bandits, we identify a significant impact on the accuracy in the long run. WSCB statistically outperforms 10 benchmarks of the literature in this same scenario.

The main contributions of this work are as follows:

(1) An empirical methodology that demonstrates the impact of non-personalised algorithms on the user's experience and the system's learning in bandit models;

(2) The combination of an Active Learning approach in Contextual Bandit algorithms to address exploration and exploitation since the first recommendations;
(3) A new algorithm named WSCB that outperforms all linear contextual bandit recently published in the literature.

The remainder of this article is organised as follows. Section 2 explains the background concepts of this work. Section 3 discusses the pure cold-start problem in current Contextual Bandits. Section 4 answers our first two research questions. Next, Section 6 describes our new approach and answers the third question. Finally, Section 6 presents our conclusions and directions for future works.

2 BACKGROUND CONCEPTS

Online recommendation scenarios have required a method that can learn from data arriving and update the predictor model at each iteration [36, 47]. Recent works have applied concepts from the Reinforcement Learning theory to handle it as a MAB problem [13, 25, 40].

2.1 Multi-Armed Bandits

Facing MAB means proposing a sequential decision model to continually choose an action a among a set of actions $\mathcal{A}$ – a.k.a. arms. The selection of action $a \in \mathcal{A}$ at each iteration t results in a certain reward $R(a_t) \in \mathbb{R}$. The main goal in the bandit problem is to maximise the expected rewards returned $\sum R(a_t)$ over time [30, 47]. However, as the optimal solution is unknown until the user rates the arm, the system should try distinct arms and learn with them. One possible solution is more conservative, pulling arms with the highest rewards in the past—*exploitation*. Another solution tries different arms to gain information and make better future decisions—*exploration*.

Traditional solutions, such as ϵ-Greedy [47], **Upper Confidence Bounds (UCB)** [38], and **Thompson Sampling (TS)** [6] handle these competitive goals in distinct ways. While ϵ-Greedy only performs a naive diversification with the parameter ϵ, the other two perform smarter strategies. UCB models a confidence range for each arm and first selects the one with the highest uncertainty. TS draws an unknown distribution for each arm to predict the expected reward according to the arm uncertainty. Even after the years, these algorithms have been usually applied as a baseline for any online environment. However, these approaches usually require that each item must be recommended at least once for each user—an unfeasible possibility in real-world scenarios these days.

2.2 Contextual Bandits

In most bandit algorithms in RS, each item $i \in I$ to be recommended is modelled as an arm a to be pulled and the reward $R(a_t)$ is the user response $r_{u,i}(t)$ (e.g., clicks, ratings, satisfaction, etc.) for that item in the trial t [30]. Again, the goal is to maximise the expected accumulated reward (i.e., the positive user's interactions). However, in the recommendation field, the expected reward should be defined by the relevance of each i for a specific user u. In general, this predicted rating $\hat{r}_{u,i}$ is based on the correlation between the item i and the user u profile (**Collaborative Filtering (CF)**). Thus, the goal is to maximise the predicted rating $\hat{r}$ or minimise the difference between the $\hat{r}$ and the real user reward r over the T trials,

$$\text{Maximise} \sum_{t=1}^{T} \hat{r}_t \quad \equiv \quad \text{Minimise} \sum_{t=1}^{T} (r_t - \hat{r}_t). \tag{1}$$

In this sense, current efforts have been applied in Contextual Bandit models, because they estimate $\hat{r}$ as the combination of CF concepts and MAB characteristics [38, 47]. A group of works

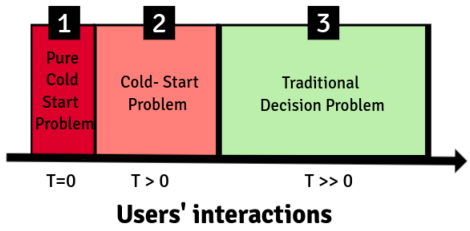


Fig. 1. Distinct stages of the user experience.

describes the reward structure in clusters of users and/or items to define items' utility similarly to memory-based methods [11, 24, 32]. In turn, model-based approaches usually apply a **Probabilistic Matrix Factorisation (PMF)** formulation to represent users and items in two low-rank matrices $P \in \mathbb{R}^{m \times z}$ and $Q \in \mathbb{R}^{z \times n}$. While the first matrix represents the multiple interests of each user u in the z features (p_u), the second one represents how relevant is each item for the z features (q_i). The predicted rating is estimated as $\hat{r}_{i,u} = p_u^\top q_i$ [38, 49]. Thus, the recommended item i will have the highest $\hat{r}_{u,i}$ during the T trials as follows. Current efforts have focused on how to optimise this objective function in bandit approaches, such as ϵ-Greedy, UCB, and TS [6, 23, 38, 47],

$$i^* = \arg\max \sum_{t \in T} \mathbb{E}\left[r_{u,i(t)}|t\right] = \arg\max \sum_{t \in T} \mathbb{E}\left[\boldsymbol{p}_{\boldsymbol{u}}^\top \boldsymbol{q}_{\boldsymbol{i(t)}}|t\right]. \tag{2}$$

3 PURE COLD-START PROBLEM IN CONTEXTUAL BANDITS

Contextual Bandits have been highly effective in the recommendation task, because they can capture non-trivial relations between items and users [4]. The disadvantage, however, is in the absence of data, where the system would not ideally have enough instances to represent these relations. The challenge has been centred on the absence of user information, especially after the recent advances in textual processing and categorical interpretation, where the systems could better represent the item's metadata. And unfortunately, it is a common issue in interactive scenarios when the system will simulate the complete user's journey since his first interaction [10, 17, 39].

3.1 Problem Formulation

There are at least two critical challenges related to the lack of information during the user's journey into the system. They are illustrated in Figure 1. The first one is known as Pure Cold-Start, and it happens during the first user's interaction or when s/he logs into the system without identifying yourself. In this case, the system does not have any information about the user and cannot provide personalised recommendations [17, 34]. After some interactions, RSs will know more about the user, but it would not be enough to define his/her profile. This scenario is called Cold-Start and has been studied for several works over the past few years [4, 18, 36].

Mitigating both challenges is crucial for the success of any interactive RSs [18, 36]. However, we noticed that the current works had underestimated the Pure Cold-Start problem in linear bandit models. As the users and items are usually modelled according to the dataset $D \in \mathbb{R}^{m \times n}$ previous available, these works have worked to define the multiple interests of each user u in z features (p_u) and how relevant is each item for the same z features (q_i). Nevertheless, when the system does not have any information about a new user (i.e., $D_u = \emptyset$), this linear-based representation cannot define the user's features vector p_u. Despite being apparently obvious, it opens an optimisation problem about how to estimate the features vector p_u when $D_u = \emptyset$ for online recommendation scenarios without compromising his/her subsequent interactions in the system.

3.2 Practical Outgrowths

Assuming that the bandit can always learn regardless of the first items recommended for each user, most bandit models do not apply any strategy to mitigate the pure cold-start problem. Their idea is to let the system learns the user's preferences with their interactions. Current works have initialised the user's features by evidencing that the new user has no preferences (not yet) about any item or feature in the system. In the practice, evidence that users do not have any preference over the items in the system means to initialise the its p_u vector with constant values: $p_u = \{c\}_{\forall z}$ ($c \geq 0$). However, this simple initialisation can introduce undesirable bias in the recommendations, as this work notices.

If the values used to initialise the features vector are 0, a usual approach in the current implementations, then the user features vector p_u will guide the objective function to 0, independently of the item features vector. The result is a uniform selection of items (i.e., random or based on the actual items order) with the same relevance score for this user: $\{s(i,u) = 0 : \forall i \in I\}$. In turn, if these values are represented by a constant $c > 0$, another possible approach, then the items features vector will become the unique relevant term of the objective function and the relevance score will follow its distribution – $\{s(i,u) \sim q_i : \forall i \in I\}$. The result is a recommendation fully biased for the most representative items that were defined according to the linear representation made before,

$$\textsc{Prediction Rule:}\ \ i^*(t) = \arg\max_{i \in I}\ p_u^\top q_i$$

$$(t=0)\ \begin{cases} p_u = \{0 : z \in Z\} \rightarrow i^*(0) = \arg\max_{i \in I}\ \mathbf{0} \cdot q_i & \text{(random)} \\ p_u = \{c : z \in Z\} \rightarrow i^*(0) = \arg\max_{i \in I}\ \boldsymbol{c} \cdot q_i & \text{(biased).} \end{cases}$$

Therefore, every contextual bandit algorithm is compelled to choose between two distinct options:

(1) a pure exploration approach by recommending items randomly; or
(2) a pure exploitation approach by recommending items biased by the previous knowledge.

The first option assumes that users are willing to interact for a long time, and the algorithm is concerned with learning more about the users' preferences. In turn, the second one assumes that users can leave the system after a few interactions and, thus, the system has to recommend items potentially relevant for users as soon as possible. Indeed, both assumptions are highly relevant. Nevertheless, here, they represent the well-known dilemma between exploration and exploitation from a new point of view: in the first users' interactions. Should we allow users to make their own choices and thus possibly lose the opportunity to sell something? Or should we assume they are impatient and recommend the best options, losing the opportunity to learn what they like? We believe both options must be addressed in this work to improve the system's user experience. An ideal system should offer its best options and help learn the users' preferences as soon as possible.

3.3 Problem Impact in Current Bandit Solutions

Inspecting the current linear approaches for traditional bandit algorithms (ϵ-Greedy, UCB, and TS), we notice that each of them will follow one specific option according to the way they have implemented the prediction rule.

(1) As **linear ϵ-greedy algorithms** usually measures the item's relevance by the product of features vectors p_u and q_i, its predictions can change according to the values used to initialise the user's features vector. If $c = 0$, then the algorithm will perform a pure-exploration of the items by

recommending them randomly. Otherwise, when $c > 0$, it will perform a pure-exploitation of the items' features.

(2) In turn, **linear UCB algorithms** implement a confidence bound to represent its uncertainty over the items and users. This confidence bound is usually measured by the combination of the item's features vector q_i with an uncertainty over the user at that trial t, named $\Sigma_{u,t}$. This component Σ is initialised as a simple identity matrix I and it is updated at each trial t. It usually performs,

$$i^*(t) = \arg\max_{i \in I} \; p_{u,t}^\top q_i + \sqrt{\boldsymbol{q_i \cdot \Sigma_{u,t} \cdot q_i}^\top}. \tag{3}$$

Thus, even if $c = 0$, the vector q_i will guide the prediction rule due to the confidence bound associated. In other words, these algorithms will always perform a pure exploitation of the previous item's knowledge,

$$\text{if } p_u = \{0\}, \text{ then: } i^*(t) = \arg\max_{i \in I} \; \sqrt{q_i \cdot \Sigma_{u,t} \cdot {q_i}^\top}. \tag{4}$$

(3) **Linear TS algorithms** usually estimate the feature vectors by sampling $\widehat{\boldsymbol{p}}_{\boldsymbol{u,t}}$ from $\mathcal{N}(p_{u,t}|\mu_{u,t}, \Sigma_{u,t})$. The variance Σ is the uncertainty around that user and starts from an identity matrix I. In turn, the mean μ is measured based on the items that the user rated and is initialised by the distribution of values c. In this case, regardless of the value assumed by c, the algorithm will sample p_u from a distribution centred in c. Thus, TS algorithms will create a normal distribution for p_u and disturb the prediction rule. As p_u can assume any value around c, the prediction will be defined by combining these "random" values over the q_i distribution. Thus, these algorithms can, in a certain way, combine exploration and exploitation to perform the first recommendations.

Moreover, studying the item's features vector q_i in three usual datasets from distinct scenarios (movies, books, and songs) in the recommendation field,[1] we figured out that q_i is extremely biased toward the most popular items. Assuming that the missing values of each dataset are 0 and applying a simple **Singular Value Decomposition (SVD)** algorithm with 10 eigenvalues,[2] we can notice that the items feature vector significantly correlates with the popularity. In Figure 2, each blue point represent an item of the system. While the x-axis is a summarisation of the item's features (i.e., $x_i = q_i \cdot {q_i}^\top$), the y-axis contains the popularity of each item (i.e., the number of ratings received normalised by the bigger popularity value). The correlation can be noticed by the linear representation of these points and the Spearman value highlighted in green. Thus, recommendations based on the item feature vector q_i tend to be similar to the most popular items.

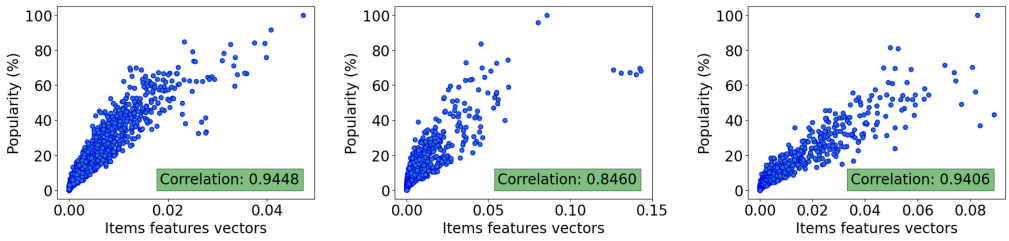
4 RELATED WORKS

The problem of addressing exploration and exploitation, since the first recommendations in Contextual Bandits has not been deeply studied yet. The most similar works are those related to the cold-start problem due to their concern about the first user's interactions [17, 34]. However, most of them do not fit the context of this work, where the algorithms should be able to (1) handle users since the pure cold-start problem, where there is *no previous knowledge* available (e.g., without 1 item rated or no social information), and (2) adapt themselves after each new user's feedback, as proposed in an interactive scenario. In this sense, we only identified three classes of algorithms that can handle it:

(1) *State-of-the-art Contextual Bandits.* In the literature, there are several algorithms to work with partially observable or unobservable contextual factors. Algorithms like *LinUCB* [23],

[1] These datasets are described on details in Section 5.2.

[2] If we increase the number of eigenvalues, then the correlation grows proportionally.



(a) Movies – Netflix (b) Books – GoodBooks (c) Music – Yahoo R1

Fig. 2. There is a significant correlation between the items features vector of MF algorithms (represented by the vector q_i) and the most popular items in three distinct scenarios.

FactorUCB [38], *hLinUCB* [37], and *CoLin* [43] explore explicit information about the user (demographic, geographical, and others) and the items (descriptions, categories, and others). Other ones, like *FactorUCB*, *CoLin* [43], **Linear UCB** [47], and **GLM-UCB** [47], explore unobservable factors by modelling users and items with a Probabilistic Matrix Factorisation approach. Moreover, more recently, new algorithms like the *hLinUCB* [37] have been proposed to aggregate the partially observable information about the users with the latent factors usually applied in the field. However, as this work proposes to study the scenario with the total absence of user's information, we do not apply any strategy that requires partially observable factors. We decided to use the algorithms highlighted in bold, because they are competitive baselines for the WSCB algorithm.

(2) *Meta-Learning approaches.* Similarlyaw to the Active Learning theory, Meta-learning is a class of Reinforcement Learning algorithms recently popularised for training easily generalised machine learning models. The idea is to create models that can rapidly adapt to a new task that is not used during the training with only a few examples [22]. It is inspired by the human learning process, which can quickly learn new tasks based on a few examples. The recommendation field has been adapted to help the system deal with the huge number of users that must receive a personalised recommendation. The idea is to consider each user as a single task and create a learning environment with a set of users to teach algorithms how to deal with new tasks (i.e., new users). The first work proposed in this sense was MeLU [22], a meta-learning strategy to get candidate items for new users. However, it is not an interactive approach that updates the knowledge about the user's preferences at each trial. Then, a recent work has proposed an interactive model that uses Meta-Learning to guide a neural network learning [50]. Such an approach is named **NICF** and we selected it as a baseline for our work.

(3) *Bayesian Inference methods.* These algorithms are based on probabilistic distributions to make inferences when very little evidence is available. Thus, they have been directly applied to deal with the cold-start scenario in the recommendation field. They usually are variations of the traditional **Thompson Sampling** by applying a Beta distribution to fit each arm's probability of success and failure with two positive parameters: α and β. They can be used to learn the arms relevance before the interactive recommendation, through a training sample, or even learn it over the user's interactions. If they choose to learn in live, then they will *explore* different arms in the first recommendations until they learn the best options. In turn, if they choose to learn before the recommendations, then they will *exploit* the most successful arms (i.e., the most popular options). In this class of algorithms, we can highlight **PTS** [21], **ICTRTS** [40], and **Cluster-Bandit** [30]. PTS introduces a particle filtering process to guide the recommendations made by a PMF formulation. In turn, ICTRTS applies a TS and particle filtering approach combined with a topic regression model to handle the dilemma of exploration and exploitation. Finally, Cluster-Bandit is a variant of

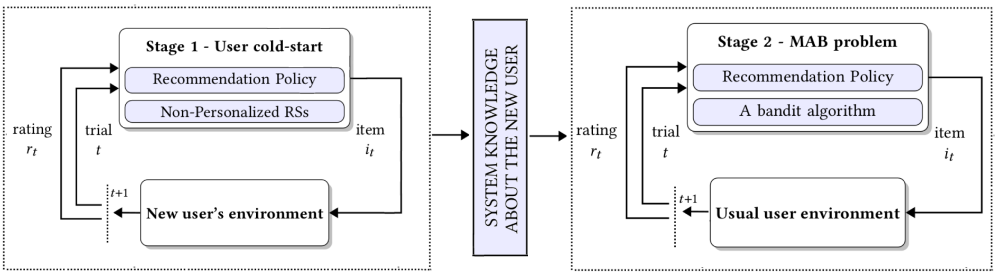


Fig. 3. An illustration of our methodology created to identify the problem's impact on contextual bandits.

the nearest-neighbours collaborative filtering algorithm but endowed with a controlled stochastic exploration from a Beta distribution. All of these three algorithms are baselines for this work.

Other works, such as those based on graph neural networks [14–16], are not in the scope of this work, because they require at least 10% of the users' interactions and they have not been adapted to the online scenario yet.

5 CONTEXTUAL BANDIT INITIALISATION

In general, applying non-personalised RSs to face the pure cold-start problem is plausible and well accepted by the RS community. Indeed, these approaches have been applied in different commerce and entertainment systems over the years until now [34]. However, as raised by our RQ1, *are these naive non-personalised approaches enough to ensure the required knowledge for the linear bandit algorithms?* We believe that naive approaches can waste the opportunity to learn more about new users when they do not have a high expectation of personalised recommendations. Thus, as raised by RQ2, the question is *how to get more knowledge about the new users in their first interactions?* To study both of them, we create an evaluation methodology that simulates the system's behaviour when it faces new users in three distinct scenarios.

5.1 Evaluation Methodology

To study our first research question about the initialisation of current contextual bandits, we design an evaluation methodology to simulate the same experience of a new user in the system. This methodology splits the user's interaction into two main stages: (1) when s/he is joining the system (to simulate the cold-start problems) and (2) his/her normal interaction (to simulate the sequential decision problem). Thus, the first f interactions of the new users happen in stage 1, and, after that, s/he is moved to stage 2. Similarly to the real behaviour of current systems, stage 2 will use the previously acquired knowledge to make new recommendations. We aim to measure the impact of this knowledge created in stage 1 in the recommendation task during stage 2. Each stage consists of one interactive algorithm with its selection policies and recommendation rules, as shown in Figure 3. While in the second one, we should set the policies related to a bandit algorithm, in the first one, we should apply the strategies we intend to initialise the models.

Stage 1: It is an interactive scenario to simulate the first recommendations of Contextual Bandit to new users and get their feedback on each recommended item. Basically, it receives an entirely new user (without any information) and performs non-personalised recommendations for f interactions until s/he becomes no longer a cold-start one. However, as there is no consensus in the literature about the value f, we define it by the amount of information σ got by the system for each new user. This knowledge σ can be represented by the number of relevant items hit from

Table 1. An Overview of the Datasets Applied in This Work

Datasets	# Users	# Items	Sparsity
Netflix	10,000	17,372	98.67%
GoodBooks	53,423	10,000	98.88%
Yahoo Music R1	10,000	13,214	99.22%

the user's historical actions. Thus, in an experiment, we can assume that a new user is no longer a cold-start after the system hits 10% of his/her relevant items. Then, we can change this value to 20%, 30%, . . . , 80% and do an exhaustive experiment to answer our first two research questions. It creates at least eight possible scenarios for each new user before s/he moves to stage 2. To make this experiment available, we can assume that the system will recommend five items per each interaction.

Stage 2: It is another interactive scenario where an agent continuously recommends a list of items to the target user. However, in this stage, the user is no longer completely new. After stage 1, this user has already interacted with some items, and there is an amount of knowledge σ about his/her preferences. So, the idea is to make more recommendations based on this knowledge according to a standard prediction rule of Contextual Bandits. Thus, the effectiveness of the bandit model on stage 2 will be directly related to the amount of knowledge achieved before by the approach applied on stage 1. At this stage, the algorithm can recommend one item per interaction.

The main advantage of this methodology is the possibility to measure the impact of the problem discussed in this work. Based on the number of interactions f in stage 1, we can measure how long the new user had to wait until the system starts to apply a personalised method. Moreover, by measuring the accuracy of the linear bandit algorithm in stage 2, we can also measure the impact of the first recommendations on the user's experience.

5.2 Experimental Setup

We build an experimental setup to simulate the scenario of new users joining the system for the first time and apply our evaluation methodology.

Datasets. First, we select the three traditional datasets described in Table 1. Then, we select the last 20% of users that joined the system (i.e., users with the highest timestamps) to represent the new users. It means 2,000 users on Netflix, 10,648 users on GoodBooks, and 2,000 on Yahoo. The other 80% of data is used to train the algorithms, representing the information already in a system. All information about these new users (ratings, demographic information, and others) is removed from the training set. To make possible our interactive experiment, we filtered all datasets to ensure that users will have at least 20 items rated, because we want to evaluate the user's long-term preference similar to other works [10]. For those with many users (Netflix and Yahoo), we selected 10,000 random users in the dataset to facilitate our exhaustive experiments.

To study our research questions, we select naive and innovative approaches to be applied in stage 1 of our evaluation methodology. The naive ones represent the first recommendations made by current Contextual Bandits, as discussed in Section 3. First, we want to answer whether these approaches are insufficient to minimise the system uncertainty about new users. Then, the innovative approach represents a potential solution to our second research question. The idea is to improve the learning process and maximise the bandit accuracy in the long run. For stage 2 of our methodology, we select the Linear ϵ-Greedy algorithm (based on SVD formulation) as the

interactive RS, because it has a higher variance in its results. It will provide 10 more interactions after each experiment is finished in stage 1.

Naive Non-Personalised Approaches. To analyse the impact of the naive assumptions made by current Contextual Bandit models, we select four non-personalised approaches. They are related to the naive assumptions of the following:

(1) **pure-exploration:** *Random* and *Entropy-based* algorithms;
(2) **pure-exploitation:** *Most Popular* and *Best-Rated* algorithms.

Innovative Approach. In addition, as a potential solution for our second research question, we also evaluate the performance of another non-personalised strategy. This strategy is smarter than the others, because it can ensure the tradeoff between *exploration and exploitation* even in the first interactions of the new users. It comes from the Active Learning theory [9, 27], and it combines the ***log*** of *popularity* and the *entropy* of the items. In this work, both concepts are defined as follows:

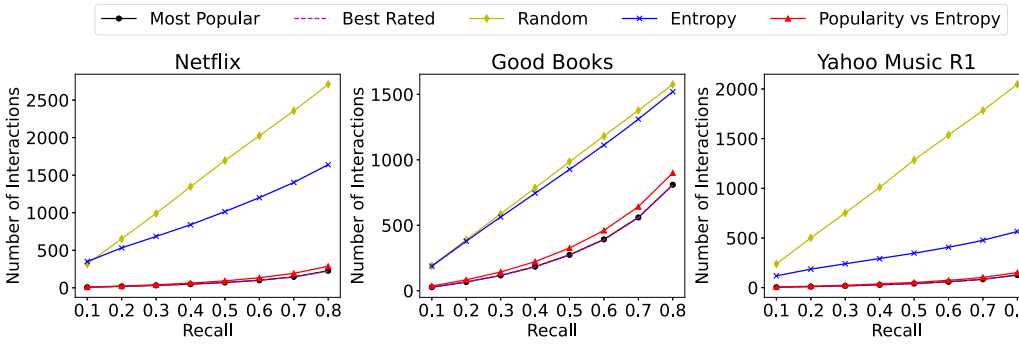
— Popularity of an item i is the number of users who have rated i;
— Entropy of an item i is measured by $\sum_{r \in R} -P(r|i) \cdot \log P(r|i)$, where $P(r|i)$ is the probability of one item i being rating by a value $r \in R$. This probability $P(r|i)$ is the proportion of ratings r received by the item i from all users.

Introducing popularity, we want to increase the probability that a user will rate an item (i.e., a notion of exploitation). However, by applying entropy, we intend to increase the amount of information that it is possible to be achieved if the user rates the item (i.e., a notion of exploration).

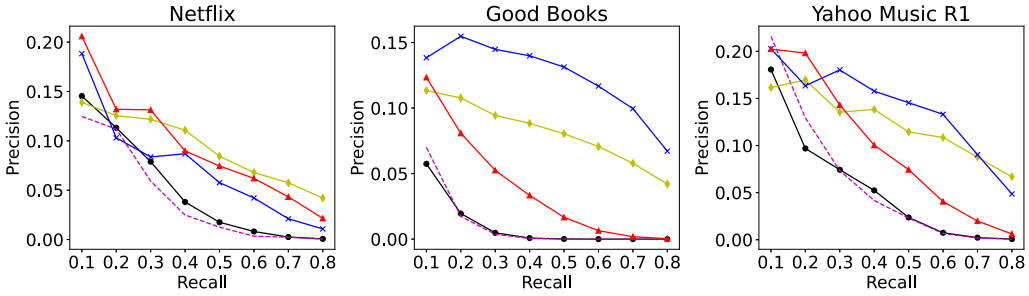
5.3 Discussion

Figure 4 shows the results of our evaluation methodology when applied to each dataset previous selected. In both figures, the x-axis represents the level of knowledge σ achieved by stage 1 of our methodology. As previously mentioned, this knowledge is measured by the percentage of relevant items hit by the recommendation model from the total of relevant items available—the same concept as the recall metric. Then, in Figure 4(a), in the y-axis, we plot the average of interactions spent by the new users to reach each level of recall. In turn, the y-axis of Figure 4(b) presents the average precision achieved by 10 other interactions of these users with the bandit model in stage 2. Thus, we can measure the effort of each non-personalised approach (in stage 1) to get the user's preferences and the quality of the recommendations made with this previous knowledge.

In a simple observation, pure-exploration approaches seem the best ones for new users, because they guide the bandit algorithm to achieve the highest level of precision (Figure 4(b)). However, to ensure this level of precision, these approaches require that users face so many items. In our experiments, for instance, Random or Entropy-based approaches require that users analyse more than 8,000 items (~2,000 interactions × five items recommended) to reach 80% of the user's history (i.e., recall) Netflix and Yahoo (Figure 4(a)), which is not feasible in practice, mainly because the user may not be so patient. Indeed, this is why current systems usually choose pure-exploitation approaches (like Most Popular and Best-Rated) to make the first recommendations. On average, they often require less than 500 iterations (2,500 items) to achieve 80% of relevant items for each user. However, our analysis shows that these approaches do not have the same impact on the bandit's performance (Figure 4(b)). These non-personalised RSs are based on global preferences. Then, as a consequence, they do not add so much knowledge about the system's users. Applying a bandit algorithm after approaches purely based on exploitation does not achieve high levels of precision. *It answers our RQ1 by empirically demonstrating the negative impact of these naive*



(a) Average of interactions performed by non-personalised RSs.



(b) Precision@10 achieved by linear ϵ-Greedy with the previous knowledge.

Fig. 4. The impact of a naive initialisation of current Contextual Bandit models. While pure exploration (Random and Entropy-based) takes more interactions to achieve higher recall, pure exploitation (Most Popular and Best-Rated) does so fast. However, this fast learning considerably impacts the model's performance as a whole. The initialisation based on exploitation and exploration simultaneously (Popularity vs. Entropy) can improve the system's precision without requiring many initial interactions.

approaches in such bandit models. Only applying most popular or random algorithms to make the first recommendations is not the best option for the other bandit interactions.

In turn, *the approach that combines Popularity vs. Entropy answers the RQ2*. By weighing items' popularity with the entropy, the system can identify interesting items for users and increase its knowledge. The practical result is an approach that requires only a few interactions (quite similar to pure exploitation approaches) but achieves more precision gains in the user's long-term run. Although this combination does not bring the same knowledge achieved by pure exploration approaches, it also does not require many interactions to identify the user's taste. Thus, evaluating both results of Figure 4(a) and (b) simultaneously, we can notice that our innovative approach is more effective than others. *It answers our second question, RQ2, by emphasising that exploration and exploitation concepts should be addressed from the first interactions of the new users.*

6 ACTIVE LEARNING IN CONTEXTUAL BANDITS

Answering RQ1 and RQ2 about the impact of the first interactions of a new user in the whole scenario, we open a new opportunity to improve bandit algorithms. Our innovative approach based on concepts from the Active Learning theory seems effective in improving the learning stage of linear bandit algorithms. Thus, the current challenge is related to the question raised by our RQ3: *What is the impact of a new strategy that gets more information about the users in the first interactions*

of current linear bandit models? In this section, we study this question by proposing a new bandit algorithm that has never been studied in the literature yet.

6.1 Active Learning

In the literature, Active Learning approaches have been proposed to improve the training processes of several Machine Learning algorithms that often require a considerable amount of high-quality data [31]. Their idea is to select the best candidate data points by querying for certain types of instances based on the system's data so far. In recommendation systems, Active Learning has been constantly motivated by the need to implement more effective sign-up processes [8]. In the sign-up stage, the system actively selects and proposes individual items or groups of items to be rated by the users [1, 28, 29]. For that, the system evaluates the entire set of items and selects the items that are estimated to be the most useful ones. The idea is to select the items that may improve the accuracy of the system.

However, to the best of our knowledge, these approaches have not been addressed in Contextual Bandit algorithms for the interactive scenario. The few works that have applied Active Learning approaches in this way have been focused on introducing a learning stage before the item recommendation [5, 9]. The idea is to search for the best items between a set of potential candidates by minimising the loss function, as proposed in the Active Learning theory. Despite its advantages, this approach is not applicable in real-world scenarios, and the algorithms may introduce some bias by sub-selecting a set of items from the system. In this work, we propose a new way to combine one of the most promising Active Learning strategies with linear Contextual Bandits. By applying this strategy, we assume we can address both concepts of exploration and exploitation from the first user's interaction.

6.2 Warming-Start Contextual Bandits

Changing the current bias of Contextual Bandit algorithms on delivering naive non-personalised items during their first recommendations is not easy. As the problem is intrinsically related to their prediction rule, any huge change can wrongly influence these methods to combine the user's and item's features. Thus, we do not propose changing how these algorithms usually work. We propose introducing more information about the system to help these algorithms to mitigate the Pure Cold-Start problem. The idea is to use the information available about the items to warm-start the user's profile and improve the first recommendations.

In this sense, we address two concepts around the items to collect more information about the system: (1) popularity (to maximise the probability of an item being rated); and (2) entropy (to increase the amount of information that is possible to obtain if the user rates the recommended item). While the first one introduces a notion of exploitation, because it suggests, in general, the best items in the system, the second represents a notion of exploration once it is interested in the potential knowledge of each item. In this work, the popularity ρ of an item i is measured by the number of distinct users who rated i. In turn, the entropy ϕ is $\sum_r -P(i|r) \cdot \log P(i|r)$, where $P(i|r)$ is the probability of an item i being rated with the rating r (usually define in a range of 1–5). We propose to make the first recommendation (when $t = 0$) being equivalent to the non-personalised Active Learning strategy that combines popularity and entropy, as illustrated in Equation (5),

$$i^*_{(t=0)} = \arg\max_{i \in I} p_u^\top q_i \quad \equiv \quad i^*_{(t=0)} = \arg\max_{i \in I} \log \rho_i \cdot \phi_i. \tag{5}$$

Thus, to make this approach available, we should approximate the rule $p_u^\top \cdot q_i$ of the values achieved by multiplying popularity and entropy. As the item features vector $q_i \in Q$ can be measured even on the user cold-start scenario, the challenge is to estimate the user's feature vector p_u.

This vector is then called as $\mathbf{x}$ and the goal is to minimise the difference to the set $\mathbf{y}$ (popularity vs. entropy) by minimising Equation (6),

$$f(x) = \sum_{i \in I} (\mathrm{y}_i - \mathrm{x}^\top \cdot q_i)^2, \quad \text{where } \mathrm{y}_i = \log \rho_i \cdot \phi_i. \tag{6}$$

In this sense, we apply a quasi-Newton method of BFGS [26] to estimate values for $\mathbf{x}$. BFGS aims to gradually minimise the loss function $f(x)$ obtained through a gradient evaluation method. Starting from a vector of constants $\overrightarrow{\mathrm{x}_0} = \{1\}_z$, for n iterations, the method estimates the next vector minimising the difference for the previous one: $\mathrm{x}_{n+1} = \mathrm{x}_n - [H(\mathrm{x}_n)]^{-1} \nabla f(\mathrm{x}_n)$. The Hessian matrix H is a square matrix of second-order partial derivatives of $f(x)$. Then, it creates a new vector x with dimensions $(1 \times z)$, where z is the number of latent factors of the Matrix Factorisation,

$$n \text{ iterations} \begin{cases} \overrightarrow{\mathrm{x}_1} \leftarrow \overrightarrow{\mathrm{x}_0} - [H(\overrightarrow{\mathrm{x}_0})]^{-1} \nabla f(\overrightarrow{\mathrm{x}_0}) \\ \overrightarrow{\mathrm{x}_2} \leftarrow \overrightarrow{\mathrm{x}_1} - [H(\overrightarrow{\mathrm{x}_1})]^{-1} \nabla f(\overrightarrow{\mathrm{x}_1}) \\ \vdots \\ \mathbf{X} \leftarrow \overrightarrow{\mathrm{x}_{n-1}} - [H(\overrightarrow{\mathrm{x}_{n-1}})]^{-1} \nabla f(\overrightarrow{\mathrm{x}_{n-1}}) \end{cases}.$$

This information added is not associated with any sensitive information about the new users, like social or demographic. For this reason, it can be adapted for any other Contextual Bandit. In this work, we adapted a linear UCB-based algorithm, as illustrated in Algorithm 1. Our proposal is named WSCB, and it gets the item features of a SVD method and uses it to compose the prediction rule. First, we estimate the values x that must be used to initialise p_u when $t = 0$ (line 1). This estimation is made by a BFGS method using the combination of popularity and entropy (y) and the item's feature vectors (Q). This new vector x is the same for all new users and does not substantially increase the execution time. At each trial, the user's features vector p_u will be estimated from this set x (line 4) and used to estimate the next item to be recommended (line 5). In this way, the bandit algorithm will not start its search from a random or biased point but from a more promising strategy. Then, this vector x will be updated at each iteration based on the latent values of the rated item (line 8).

ALGORITHM 1: Warm-Starting Contextual Bandits (WSCB)

Require: Item features vector $Q = \{q_1, \ldots, q_n\}$ from SVD and the vector y with the items popularity weighted by the entropy

1: $\mathbf{x} \leftarrow BFGS: \text{ Min } f(x) \text{ using } (y, Q)$
2: $\Sigma_{u,t} \leftarrow I_d$
3: **for** $t = 0, 1, 2, \ldots, T$ **do**
4: Estimate $p_{u,t} = \Sigma_{u,t}^{-1} \cdot \mathbf{x}$
5: ${i_t}^* = \arg\max_{i \in I} p_{u,t}^\top q_i + ||q_i||_{2, \Sigma_{u,t}}$
6: Receive the reward $r_{u, i^*(t)}$
7: Update $\Sigma_{u,t} \leftarrow \Sigma_{u,t} + q_{i^*(t)} \cdot q_{i^*(t)}^\top$
8: Update $\mathbf{x} \leftarrow \mathbf{x} + r_{u, i^*(t)} \cdot q_{i^*(t)}$

6.3 Experimental Setup

To evaluate our proposal and answer our third research question, we perform an experimental evaluation to compare the WSCB with several baselines of the literature in traditional datasets. This

evaluation is done through the iRec [35], a framework for interactive recommendation systems recently published in the literature. All codes and experiments are also available in the GitHub.[3]

Datasets. As described in Section 5.2, to simulate the cold-start scenario, we select the new users (test set) as the last 20% to join the system and remove all their information from the training. The other 80% of data is used to train the algorithms. For these experiments, we also added the traditional MovieLens 1M dataset. Our idea is to allow an easier comparison of our results with the previous works published in the literature.

Baselines. In this work, we compare the WSCB with 11 baselines: two simple non-personalised algorithms (**Random** and **Popular**); three traditional MAB algorithms adapted to the RS field (ϵ-Greedy, UCB, and TS); and competitive baselines from the three classes of algorithms related to this challenge (those marked in bold and discussed in detail in Section 4). They are as follows:

- **ϵ-Greedy** [2]: This is a classical bandit model that random explores other arms with probability ϵ.
- **UCB** [2]: This calculates a confidence interval for each item and tries to shrink the confidence bounds at each iteration.
- **TS** [6]: This follows a Gaussian distribution of items and users to predict based on samples.
- **Linear UCB** [47]: This is an adaptation of the LinUCB [23] with the PMF latent dimensions.
- **GLM-UCB** [47]: This is an adaptation of the Linear UCB with a sigmoid form that makes a time-dependent exploration.
- **ICTRTS** [41]: This is a topic regression model that utilises the TS and controls the items dependency by a particle learning strategy.
- **NICF** [50]: This is a CF based on a neural network that performs a meta-learning of the user's preferences.
- **PTS** [21]: This is a PMF formulation for the original TS that applies particle filtering to guide the exploration of items over time.
- **Cluster-Bandit (CB)** [32]: This is a bandit algorithm based on clusters to face the cold-start problem.

For all baselines, we perform an extensive grid search for the best parameters in a validation set (20% of the training). We followed the same parameters suggested by each original paper when the authors were reported in this step. We followed these reported parameters for most algorithms and tried to identify other values that could improve the recommendation quality. In Reference [47], for instance, the authors only report some parameters of Linear UCB and GLM-UCB without providing the range of values studied. Here, we explored their parameters more by focusing on the PMF model. Unfortunately, for other algorithms, like NICF [50], we had to follow the parameters available in their code and try to infer other ranges of values according to our knowledge, because the authors did not report any parameter. The final result of this extensive exploration is then described in Table 2, where we showed the best parameters applied for each algorithm in the scenarios evaluated in this work.

Evaluation Policy. As the interactive scenario has been recently studied in the literature, there is no consensus about the best practices to evaluate bandit algorithms. In this sense, we searched for all policies available on the articles from our SLR and then proposed the evaluation policy described on the Algorithm 2. This evaluation is similar to a recent one proposed in Reference [30]. Each trial represents a user's interaction with the system when one item is recommended, and the user will

[3]Available at: https://github.com/ncsilvaa/wscb.

Table 2. Best Parameters Identified for Each Algorithm after Using the Grid Search

	Netflix	Good Books	Yahoo Music R1	MovieLens 1M
Random	—	—	—	—
Popular	—	—	—	—
UCB	c = 0.01	c = 0.25	c = 0.01	c = 0.001
TS	$\alpha_0 = 1, \beta_0 = 100$	$\alpha_0 = 1, \beta_0 = 100$	$\alpha_0 = 1, \beta_0 = 100$	$\alpha_0 = 0.2, \beta_0 = 100$
ϵ-Greedy	$\varepsilon = 0.001$	$\varepsilon = 0.1$	$\varepsilon = 0.0001$	$\varepsilon = 0.001$
Linear UCB	• $\alpha = 1, T = 20$ • $\sigma_q^2 = 0.01, \sigma^2 = 0.05$ • num_lat = 20 • stop_criteria = 0.0009 • $q_p^2 = 0.01$	• $\alpha = 1, T = 20$ • $\sigma_q^2 = 0.01, \sigma^2 = 0.05$ • num_lat = 10 • stop_criteria = 0.0009 • $q_p^2 = 0.01$	• $\alpha = 1, T = 20$ • $\sigma_q^2 = 0.01, \sigma^2 = 0.05$ • num_lat = 20 • stop_criteria = 0.0009 • $q_p^2 = 0.01$	• $\alpha = 0.7, T = 20$ • $\sigma_q^2 = 0.01, \sigma^2 = 0.05$ • num_lat = 15 • stop_criteria = 0.0009 • $q_p^2 = 0.01$
GLM-UCB	• c = 4, $T = 20$ • $\sigma_q^2 = 0.01, \sigma^2 = 0.05$ • $num\_lat = 10$ • $stop\_criteria = 0.0009$ • $q_p^2 = 0.01$	• c = 4, $T = 20$ • $\sigma_q^2 = 0.01, \sigma^2 = 0.05$ • $num\_lat = 10$ • $stop\_criteria = 0.0009$ • $q_p^2 = 0.01$	• c = 4, $T = 20$ • $\sigma_q^2 = 0.01, \sigma^2 = 0.05$ • $num\_lat = 10$ • $stop\_criteria = 0.0009$ • $q_p^2 = 0.01$	• c = 4, $T = 20$ • $\sigma_q^2 = 0.01, \sigma^2 = 0.05$ • $num\_lat = 10$ • $stop\_criteria = 0.0009$ • $q_p^2 = 0.01$
NICF	• batch = 256 • clip_param = 0.2 • dropout_rate = 0.01 • gamma = 0.0 • inner_epoch = 50 • latent_factor = 10 • learning_rate = 0.001 • num_blocks = 1 • num_heads = 2 • restore_model = False • rnn_layer = 2 • time_step = 100 • training_epoch = 4000	• batch = 256 • clip_param = 0.2 • dropout_rate = 0.01 • gamma = 0.0 • inner_epoch = 50 • latent_factor = 10 • learning_rate = 0.001 • num_blocks = 1 • num_heads = 2 • restore_model = False • rnn_layer = 2 • time_step = 100 • training_epoch = 4000	• batch = 256 • clip_param = 0.2 • dropout_rate = 0.01 • gamma = 0.0 • inner_epoch = 50 • latent_factor = 10 • learning_rate = 0.001 • num_blocks = 1 • num_heads = 2 • restore_model = False • rnn_layer = 2 • time_step = 100 • training_epoch = 4000	• batch = 256 • clip_param = 0.2 • dropout_rate = 0.01 • gamma = 0.0 • inner_epoch = 50 • latent_factor = 10 • learning_rate = 0.001 • num_blocks = 1 • num_heads = 2 • restore_model = False • rnn_layer = 2 • time_step = 100 • training_epoch = 4000
PTS	• num_lat = 10, D = 5 • $\sigma^2 = 0.01$ • $\sigma_U^2 = 0.173, \sigma_V^2 = 2.65$	• num_lat = 5, D = 2 • $\sigma^2 = 0.5$ • $\sigma_U^2 = 1.0, \sigma_V^2 = 1.0$	• num_lat = 10, D = 5 • $\sigma^2 = 0.173$ • $\sigma_U^2 = 0.01, \sigma_V^2 = 0.3$	• num_lat = 10, D = 5 • $\sigma^2 = 0.5$ • $\sigma_U^2 = 0.01, \sigma_V^2 = 0.3$
ICTRTS	• num_lat = 2, $B = 5$	• num_lat = 2, $B = 5$	• num_lat = 2, $B = 5$	• num_lat = 10, $B = 2$
CB	• B = 5, C = 0.5, D = 3 • num_clusters = 8 • num_lat = 10	• B = 5, C = 0.5, D = 3 • num_clusters = 8 • num_lat = 20	• B = 5, C = 0.5, D = 3 • num_clusters = 4 • num_lat = 20	• B = 5, C = 0.5, D = 3 • num_clusters = 8 • num_lat = 10
WSCB	• $\alpha = 1$, num_lat = 10	• $\alpha = 0.25$, num_lat = 10	• $\alpha = 0.5$, num_lat = 10	• $\alpha = 0.95$, num_lat = 10

rate or not it. At each trial $t \in T$, the new user u is selected uniformly to simulate the random accessing pattern of users in the real world. Then, the system will recommend 1 item i for this user according to the method Π. Method Π will recommend the new item based on the training set $\mathcal{D}$, the available items (i.e., the items that have not been recommended yet—$I \setminus R_u$), and the knowledge Δ achieved until the trial t. After that, the system will update the current knowledge for the next iterations. Unlike other works, the system is not compelled only to recommend previous rated items (i.e., those from the test) to avoid a biased solution. All items can be recommended at each trial as long as it was not recommended in another trial before. Moreover, in our case, we compel the system to always make T iterations for each new user u.

After that, the recommendations quality are then evaluated by the following:

- **Cumulative Reward** [30]: It is the cumulative number of hits achieved over T trials. A hit happens when the item was rated for the target user with a value equals or higher than 4 in the test set.

ALGORITHM 2: Evaluation Policy

Require: Training set $\mathcal{D}$, testing set $\mathcal{T}$, number of trials T, and the number of items k to be recommended for each user per trial

$R_u \leftarrow \emptyset_{\forall u \in \mathcal{U}}$ // Recommendation list
$\Delta \leftarrow \emptyset$ // Knowledge around the users
$N \leftarrow |U| \times T$ // Total number of iterations to play
for $t = 1, 2, 3, \ldots, N$ **do**
  $\Delta' \leftarrow \emptyset$ // Saves the current knowledge
  $u \leftarrow |U|$ // Random selects a user u (limited to T trials)
  $i_t \leftarrow \Pi(u, \mathcal{D}, I \setminus R_u, \Delta)$ // Gets the item recommended
  $R_u \leftarrow R_u \cup \{i_t\}$ // Saves this item
  $\Delta \leftarrow \Delta \cup \{\mathcal{T}(u, i_t)\}$ // Updates the system knowledge

- **Precision** [47]: It is the number of positive rewards (i.e., items rated with $r \geq 4$) after T interactions.
- **Recall** [47]: It is the number of positive rewards after T interactions according to the amount of possible positive rewards.

6.4 Results and Discussion

First, we measure and analyse the cumulative reward over time. Then, we measure the quality of each algorithm by reward, precision, and recall in each stage of the user's journey (recommendations made for an interval of trials).

6.4.1 The Expected Reward. In Table 3, we report the performance of cumulative precision and recall throughout 100 trials. In general, the results are pretty consistent with our assumptions. Methods based on pure-exploration approaches to make the first recommendations (e.g., ϵ-Greedy and TS) perform worst in the first trials and then get better in the long run (sometimes only after 100 iterations). In turn, methods that perform a pure-exploitation approach to make the first recommendations (e.g., Popular, UCB, Linear UCB, and NICF) are effective in the first trials (5 or 10) by achieving high values of precision, but without great benefits in the long run. In this class of methods, the results of two algorithms are noteworthy: Linear UCB and GLM-UCB. They can achieve higher accuracy values by handling exploration and exploitation over time. For instance, the Linear UCB almost achieves the same values as our method in Netflix and Yahoo datasets. In our opinion, it happens in this dataset due to the strong correlation between the item's popularity and its features vector (Figure 2). However, the Linear UCB does not perform as well as our method in other datasets with weaker correlation, like Good-Books. Even so, the WSCB outperforms the Linear UCB for T equals 20 and higher in all datasets.

However, methods proposed to handle exploration and exploitation to make the first recommendations (e.g., PTS and our WSCB) can achieve higher levels of precision and recall just after the first 20 interactions. Contrasting the pure-exploitation approaches, they propose attracting the user's attention in the first interactions and learning more about his/her preferences. So, when $T = 5$ and $T = 10$, the results are close to Popular or NICF. In turn, since the user has given enough feedback to the system, such methods outperform other baselines, because they learned more about the users. However, PTS and WSCB explores distinct strategies to get more information about the users. In the Bayesian inference theory used in the PTS, the algorithm makes a *random* estimation over the prior knowledge when there is no evidence. In turn, our WSCB applies a smarter approach that selects the items based on the entropy, i.e., how much knowledge each item can provide for

Table 3. The Performance of Different Bandit Models on the Selected Datasets

Measure	Cumulative Expected Reward				
Dataset	MovieLens 1M				
T	5	10	20	50	100
Random	0.125	0.243	0.499	1.223	2.464
Popular	**2.033**	**3.806**	7.003	13.978	22.198
UCB	1.305	2.422	4.344	9.340	16.274
TS	1.567	2.986	5.498	11.549	19.135
e-Greedy	1.309	2.459	4.488	9.550	16.383
Linear UCB	1.633	3.201	6.450	15.073	26.879
GLM-UCB	1.224	2.012	4.988	13.762	25.187
NICF	**1.966**	**3.717**	6.984	14.369	21.942
PTS	1.823	**3.796**	**7.488**	15.454	24.305
ICTRTS	0.175	0.676	2.069	7.471	16.047
Cluster Bandit	1.282	3.015	6.314	13.569	22.156
WSCB	1.536▼	2.508▼	7.041▼	**17.859**▲	**30.791**▲
Dataset	Netflix				
T	5	10	20	50	100
Random	0.023	0.051	0.087	0.220	0.453
Popular	**1.490**	**2.977**	5.071	10.325	18.422
UCB	0.693	1.325	2.417	5.395	9.870
TS	0.967	1.895	3.509	7.433	13.002
e-Greedy	0.664	1.297	2.393	5.369	9.874
Linear UCB	0.638	1.877	4.595	12.289	22.636
GLM-UCB	0.556	1.231	3.853	11.903	22.424
NICF	1.468	2.678	4.842	10.408	15.723
PTS	0.281	0.688	1.890	5.636	9.742
ICTRTS	0.014	0.056	0.320	2.567	5.674
Cluster-Bandit	0.896	1.968	4.229	9.148	16.317
WSCB	1.089▼	1.461▼	**5.213**●	**15.184**▲	**25.947**▲
Dataset	Good Books				
T	5	10	20	50	100
Random	0.036	0.070	0.148	0.382	0.764
Popular	1.238	2.327	4.246	8.082	12.149
UCB	0.410	0.751	1.348	2.947	5.464
TS	0.773	1.352	2.311	4.536	7.185
e-Greedy	0.445	0.807	1.422	3.090	5.659
Linear UCB	0.644	1.628	2.974	7.154	13.024
GLM-UCB	0.457	0.839	2.822	7.013	12.645
NICF	**1.517**	**2.920**	4.588	7.902	10.620
PTS	0.560	1.031	2.056	5.403	11.183
ICTRTS	0.380	1.227	2.806	6.993	11.314
Cluster-Bandit	0.944	2.160	4.031	7.666	11.879
WSCB	1.087▼	1.776▼	**5.469**▲	**11.759**▲	**18.623**▲
Dataset	Yahoo Music				
T	5	10	20	50	100
Random	0.021	0.037	0.083	0.188	0.368
Popular	1.614	2.932	5.051	10.460	17.194
UCB	0.503	1.369	3.035	7.339	13.355
TS	0.978	1.922	3.669	8.268	14.675
e-Greedy	0.589	1.429	3.100	7.404	13.307
Linear UCB	1.562	3.103	6.460	15.671	25.508
GLM-UCB	0.865	1.569	5.056	14.120	24.434
NICF	**1.846**	**3.514**	6.224	12.114	16.230
PTS	1.571	3.243	6.642	14.416	19.955
ICTRTS	0.013	0.157	1.405	6.042	13.341
Cluster-Bandit	1.019	2.344	4.616	9.845	16.497
WSCB	1.389▼	2.414▼	**7.249**▲	**17.914**▲	**27.247**▲

As the WSCB uses the first interactions to explore other items and get more knowledge about the new user, its accuracy is smaller than the best ones in these trials. However, after the first 10 interactions, WSCB outperforms all other baselines by achieving high improvements in the last trials. The symbols ▲, ●, ▼ denote positive gains, non significant changes, and negative losses by applying a Wilcoxon test with a p value = 0.05 over the best baseline.

the system. Thus, WSCB outperforms the PTS after $T = 20$ trials in all datasets studied except in the MovieLens due to the small number of items (i.e., arms). In this case, it happens after the $T = 50$.

In short, we can highlight that

(1) Linear ϵ-Greedy quality increases over time, because it requires more time to learn about users.
(2) TS, PTS, and ICTRTS usually improve their results in the long run as other learning algorithms. These methods start from random choices between the most popular items in the first iterations, improving their performance over time.
(3) Linear UCB and GLM-UCB work better than traditional UCB, TS, and ϵ-Greedy, since they are Parametric Bandit algorithms and get the user's preferences after some interactions.
(4) WSCB outperforms all algorithms over time due to its learning ability that captures the user's preferences since the first interactions.

6.4.2 Quality in the User's Journey. Second, we intend to evaluate the models effectiveness over the recommendations made at each stage of the user's experience. In this sense, we split the 100 interactions of each user in three stages. These user's stages help us to identify the impact of little changes in the pure cold-start stage. As there is no consensus about the number of trials for each stage, we inspired on works with distinct analysis about new users [50] and defined some arbitrary values:

(1) Pure cold-start: evaluating the first trial ($1 \leq T \leq 5$);
(2) Cold-start: evaluating the trials in ($6 \leq T \leq 20$);
(3) Sequential problem: evaluating the trials in ($21 \leq T \leq 100$).

Results are highlighted in Table 4, where we measure the cumulative reward at each stage of the user's experience (i.e., the reward achieved based on the items recommended in that stage). Contrasting the current Parametric Bandit algorithms, WSCB is not focused on simply hitting the new user's preferences with best-seller items in the first trials. By balancing exploration and exploitation in the first recommendations, our method uses the first interactions to learn more about the new users' preferences. Thus, as expected by our assumptions, we notice that WSCB is not the best method in the first stage. However, it is clear that our algorithm is not so far from the other models that are entirely focused on this first stage, such as Most Popular, NICF, and others. After it, WSCB can learn quickly and improve its recommendations by fitting the user's preferences after 10 items recommended (trials 11–15). Moreover, neither other algorithms can achieve the same reward level as our method in the following stages. Only the Linear-UCB and the GLM-UCB can get more rewards in a specific stage in the Yahoo dataset. But it happens due to the limitations of the offline evaluation where the number of rewards are limited to a few items. In this case, algorithms that hit only few items in the first intervals have more probability to hit more items in the remaining intervals, because more items left to be hit. And, as aforementioned, such algorithms have a slow learning process that starts with higher precision (once it is biased for pure-exploitation at the beginning). However, only WSCB is mainly focused on delivering more items that match the user's taste over his journey in the system, achieving more chances to improve his engagement.

This behaviour is even more evident in Figure 5, where is highlighted the precision and recall (y-axes) for each number of items recommended (similarly to the number of trials, once each trial recommends one item). In terms of precision, we can notice the exact stage in which WSCB overcomes the other top algorithms—usually after 20 items are recommended. In turn, by analysing Recall (i.e., the percentage of possible relevant items recommended for each user), we can clearly

Table 4. The Performance of Different Models on Each Stage of the New User

Measure	Reward in each user's stage					
Dataset	MovieLens 1M					
T	1–5	6–10	11–15	16–20	21–50	51–100
Random	0.125	0.118	0.137	0.118	0.724	1.242
Popular	**2.033**	1.773	1.651	1.546	6.974	8.220
UCB	1.305	1.117	0.970	0.951	4.997	6.934
TS	1.567	1.419	1.297	1.215	6.050	7.586
e-Greedy	1.309	1.151	1.033	0.996	5.062	6.833
Linear UCB	1.633	1.568	1.671	1.578	8.623	11.806
GLM-UCB	1.224	0.787	1.416	1.560	8.774	11.425
NICF	**1.966**	1.751	1.703	1.565	7.385	7.573
PTS	1.823	**1.973**	1.926	1.766	7.967	8.850
ICTRTS	0.175	0.501	0.657	0.735	5.402	8.576
Cluster Bandit	1.282	1.733	1.790	1.509	7.255	8.587
WSCB	1.553▼	1.067▼	**2.315**▲	**2.196**▲	**10.849**▲	**12.769**▲
Dataset	Netflix					
T	1–5	6–10	11–15	16–20	21–50	51–100
Random	0.023	0.028	0.019	0.018	0.132	0.233
Popular	**1.490**	**1.486**	0.990	1.105	5.254	8.097
UCB	0.693	0.632	0.549	0.542	2.978	4.474
TS	0.967	0.929	0.826	0.787	3.924	5.569
e-Greedy	0.664	0.632	0.571	0.525	2.976	4.505
Linear UCB	0.638	1.239	1.377	1.341	7.694	10.347
GLM-UCB	0.556	0.674	1.234	1.388	8.050	10.521
NICF	1.468	1.210	1.149	1.016	5.566	5.315
PTS	0.281	0.407	0.572	0.630	3.747	4.106
ICTRTS	0.014	0.042	0.088	0.176	2.247	3.107
Cluster-Bandit	0.896	1.072	1.067	1.195	4.918	7.169
WSCB	1.089▼	0.371▼	**1.834**▲	**1.917**▲	**9.972**▲	**10.762**●
Dataset	Good Books					
T	1–5	6–10	11–15	16–20	21–50	51–100
Random	0.036	0.034	0.037	0.041	0.235	0.382
Popular	1.238	1.089	1.116	0.804	3.835	4.068
UCB	0.410	0.341	0.305	0.293	1.599	2.517
TS	0.773	0.580	0.514	0.445	2.224	2.650
e-Greedy	0.445	0.362	0.320	0.296	1.668	2.569
Linear UCB	0.644	0.985	0.687	0.659	4.180	5.869
GLM-UCB	0.457	0.382	1.100	0.883	4.191	5.633
NICF	**1.517**	**1.403**	0.910	0.758	3.314	2.718
PTS	0.560	0.471	0.520	0.504	3.348	5.780
ICTRTS	0.380	0.847	0.800	0.780	4.187	4.322
Cluster-Bandit	0.944	1.216	1.065	0.806	3.635	4.213
WSCB	1.087▼	0.689▼	**2.057**▲	**1.637**▲	**6.290**▲	**6.864**▲
Dataset	Yahoo Music					
T	1–5	6–10	11–15	16–20	21–50	51–100
Random	0.021	0.017	0.024	0.022	0.104	0.181
Popular	1.614	1.317	1.083	1.036	5.409	6.734
UCB	0.503	0.866	0.829	0.839	4.303	6.016
TS	0.978	0.944	0.907	0.840	4.599	6.406
e-Greedy	0.589	0.840	0.848	0.824	4.304	5.903
Linear UCB	1.562	1.541	1.655	1.702	9.211	9.836
GLM-UCB	0.865	0.704	1.714	1.774	9.064	**10.314**
NICF	**1.846**	1.667	1.450	1.262	5.890	4.115
PTS	1.571	**1.671**	1.711	1.688	7.774	5.539
ICTRTS	0.013	0.143	0.500	0.748	4.637	7.298
Cluster-Bandit	1.019	1.325	1.200	1.073	5.229	6.652
WSCB	1.389▼	1.026▼	**2.493**▲	**2.342**▲	**10.665**▲	9.334▼

WSCB performs better than other algorithms after some stages by achieving statistically improvements, because it uses the first stages to get more information about the users. The symbols ▲, ●, ▼ denote respectively positive gains, non significant changes, and negative losses by applying a Wilcoxon test with a p value = 0.05 over the best baseline.

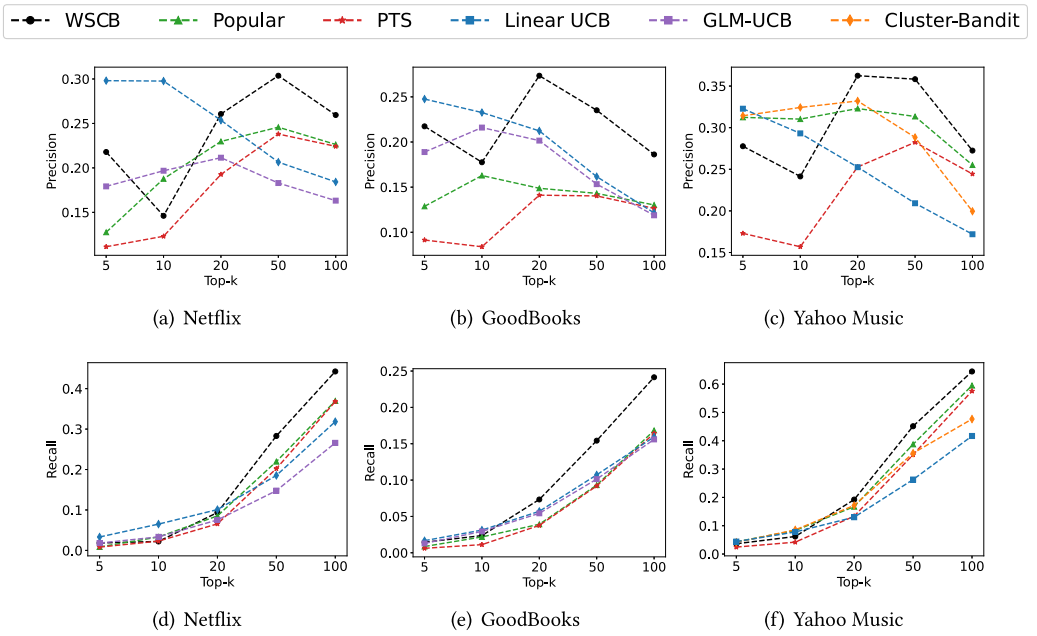


Fig. 5. The precision and recall achieved by our WSCB algorithm and the top baselines with k recommended items. The first line (precision) show how much recommended items were relevant for each user. The second line (recall) shows how much relevant items were recommended over the total of relevant items available. As the user receives one item in each trial, the number k also represents the number of trials faced by each user.

notice the impact of changing the first recommendations to learn more about each new user. Only WSCB can achieve more than 40%, 25%, and 60% of the relevant items in Netflix, GoodBooks, and Yahoo, respectively. As assumed by us, our algorithm seems able to retrieve more interesting items for each user—more potential evidence that WSCB can improve the user's engagement.

7 CONCLUSION AND FUTURE WORKS

This work shows that inadequate strategies to make the first recommendations for new users in Contextual Bandits can influence user experience in the long run. According to their implementations to deal with new users, user's preference vectors have guided the system to forget the exploitation/exploration tradeoff and choose only one option for this dilemma. Thus, current parametric bandit algorithms are making first recommendations based on a random sampling of the items (i.e., a pure-exploration) or a biased selection of the most popular ones (i.e., a pure-exploitation). It has created a chain reaction and guided the user to bad experiences in the system. In this sense, we propose a new WSCB method, introducing popularity and entropy to address exploration and exploitation concepts since the first recommendation. We hypothesise that it may maximise the user's satisfaction in the long run. Indeed, as shown by our experimental evaluation, WSCB improves the user's experience by quickly learning the user's preferences. After a few user's interactions, in general, 10 items recommended, WSCB can provide a higher level of precision and recall than the other 10 baselines. Especially our method is also the one responsible for hitting more items for the most significant number of distinct users in all datasets. As future work, we suggest three distinct directions. First, we intend to evaluate our initialisation method in other contextual bandits, especially the non-linear ones. Second, we intend to investigate the behaviour

of such algorithms if the system has more information about the users. In some real-world applications, there is an on-boarding stage where the user fills out a form or answers some questions about his/her preferences. For this one, we could apply our Active Learning approach to select the candidate items to fill these forms and then use this background information to create the first feature vector for the new user. Finally, we also intend to keep investigating how to improve other stages of the user's journey. Recent works pointed to some problems related to user-dynamic preferences. We intend to study how to adapt such bandit models to address these challenges.

REFERENCES

[1] Rabaa Alabdulrahman, Herna Viktor, and Eric Paquet. 2019. Active learning and deep learning for the cold-start problem in recommendation system: A comparative study. In *International Joint Conference on Knowledge Discovery, Knowledge Engineering, and Knowledge Management*. Springer, 24–53.

[2] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. 2002. Finite-time analysis of the multiarmed bandit problem. *Mach. Learn.* 47, 2–3 (2002), 235–256.

[3] Andrea Barraza-Urbina and Dorota Glowacka. 2020. Introduction to bandits in recommender systems. In *Proceedings of the 14th ACM Recommender Systems Conference (RecSys'20)*. 748–750.

[4] Jesús Bobadilla, Fernando Ortega, Antonio Hernando, and Abraham Gutiérrez. 2013. Recommender systems survey. *Knowl.-Bas. Syst.* (2013).

[5] Djallel Bouneffouf, Romain Laroche, Tanguy Urvoy, Raphael Féraud, and Robin Allesiardo. 2014. Contextual bandit for active learning: Active thompson sampling. In *International Conference on Neural Information Processing*. Springer, 405–412.

[6] Olivier Chapelle and Lihong Li. 2011. An empirical evaluation of thompson sampling. In *Advances in Neural Information Processing Systems*. 2249–2257.

[7] James Davidson, Benjamin Liebald, Junning Liu, Palash Nandy, Taylor Van Vleet, Ullas Gargi, Sujoy Gupta, Yu He, Mike Lambert, Blake Livingston, et al. 2010. The YouTube video recommendation system. In *Proceedings of the 4th ACM Conference on Recommender Systems*. 293–296.

[8] Christian Desrosiers and George Karypis. 2011. A comprehensive survey of neighborhood-based recommendation methods. In *Recommender Systems Handbook* (2011), 107–144.

[9] Mehdi Elahi, Francesco Ricci, and Neil Rubens. 2016. A survey of active learning in collaborative filtering recommender systems. *Comput. Sci. Rev.* 20 (2016), 29–50.

[10] Crícia Felício, Klérisson Paixão, Celia Barcelos, and Philippe Preux. 2017. A multi-armed bandit model selection for cold-start user recommendation. In *Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization*.

[11] Claudio Gentile, Shuai Li, Purushottam Kar, Alexandros Karatzoglou, Giovanni Zappella, and Evans Etrue. 2017. On context-dependent clustering of bandits. In *Proceedings of the International Conference on Machine Learning*. PMLR, 1253–1262.

[12] Carlos A. Gomez-Uribe and Neil Hunt. 2015. The netflix recommender system: Algorithms, business value, and innovation. *ACM Trans. Manage. Inf. Syst.* 6, 4 (2015), 1–19.

[13] Anjan Goswami, Chengxiang Zhai, and Prasant Mohapatra. 2019. Learning to diversify for e-commerce search with multi-armed bandit. In *Proceedings of the SIGIR Workshop on eCommerce (eCOMSIGIR'19)*.

[14] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, Vol. 30.

[15] Bowen Hao, Jing Zhang, Hongzhi Yin, Cuiping Li, and Hong Chen. 2021. Pre-training graph neural networks for cold-start users and items representation. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*. 265–273.

[16] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 639–648.

[17] Antonio Hernando, Jesús Bobadilla, Fernando Ortega, and Abraham Gutiérrez. 2017. A probabilistic model for recommending to new cold-start non-registered users. *Inf. Sci.* 376 (2017), 216–232.

[18] Steven Hoi, Doyen Sahoo, Jing Lu, and Peilin Zhao. 2018. Online learning: A comprehensive survey.

[19] Dietmar Jannach and Michael Jugovac. 2019. Measuring the business value of recommender systems. *ACM Trans. Manage. Inf. Syst.* 10, 4 (2019), 1–23.

[20] Dietmar Jannach, Markus Zanker, Alexander Felfernig, and Gerhard Friedrich. 2010. *Recommender Systems: An Introduction*. Cambridge University Press.

[21] Jaya Kawale, Hung H. Bui, Branislav Kveton, Long T. Thanh, and Sanjay Chawla. 2015. Efficient thompson sampling for online matrix-factorization recommendation. In *Advances in Neural Information Processing Systems*, Vol 28, 1297–1305.

[22] Hoyeop Lee, Jinbae Im, Seongwon Jang, Hyunsouk Cho, and Sehee Chung. 2019. MeLU: Meta-learned user preference estimator for cold-start recommendation. In *Proceedings of the 25th ACM Conference of the Special Interest Group on Knowledge Discovery and Data Mining (SIGKDD'19)*. 1073–1082.

[23] Lihong Li, Wei Chu, John Langford, and Robert E. Schapire. 2010. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th International Conference on World Wide Web*. 661–670.

[24] Shuai Li, Alexandros Karatzoglou, and Claudio Gentile. 2016. Collaborative filtering bandits. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 539–548.

[25] Yanghao Li, Cuiling Lan, Junliang Xing, Wenjun Zeng, Chunfeng Yuan, and Jiaying Liu. 2016. Online human action detection using joint classification-regression recurrent neural networks. In *Proceedings of the European Conference on Computer Vision*. Springer, 203–220.

[26] Jorge Nocedal and Stephen J. Wright. 2006. Sequential quadratic programming. *Numer. Optim.* (2006), 529–562.

[27] Al Mamunur Rashid, Istvan Albert, Dan Cosley, Shyong K. Lam, Sean M. McNee, Joseph A. Konstan, and John Riedl. 2002. Getting to know you: Learning new user preferences in recommender systems. In *Proceedings of the 7th International Conference on Intelligent User Interfaces*. 127–134.

[28] Neil Rubens, Mehdi Elahi, Masashi Sugiyama, and Dain Kaplan. 2015. Active learning in recommender systems. In *Recommender Systems Handbook*. Springer, 809–846.

[29] Neil Rubens and Masashi Sugiyama. 2007. Influence-based collaborative active learning. In *Proceedings of the ACM Conference on Recommender Systems*. 145–148.

[30] Javier Sanz-Cruzado, Pablo Castells, and Esther López. 2019. A simple multi-armed nearest-neighbor bandit for interactive recommendation. In *Proceedings of the 13th ACM Conference on Recommender Systems*. 358–362.

[31] Burr Settles. 2009. Active learning literature survey.

[32] Sulthana Shams, Daron Anderson, and Douglas Leith. 2021. Cluster-based bandits: Fast cold-start for recommender system new users.

[33] Bracha Shapira, Francesco Ricci, Paul B. Kantor, and Lior Rokach. 2011. *Recommender Systems Handbook*.

[34] Nícollas Silva, Diego Carvalho, Adriano C. M. Pereira, Fernando Mourão, and Leonardo Rocha. 2019. The pure cold-start problem: A deep study about how to conquer first-time users in recommendations domains. *Inf. Syst.* 80 (2019), 1–12.

[35] Thiago Silva, Nícollas Silva, Heitor Werneck, Carlos Mito, Adriano CM Pereira, and Leonardo Rocha. 2022. iRec: An interactive recommendation framework. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 3165–3175.

[36] Richard S. Sutton and Andrew G. Barto. 2018. *Reinforcement Learning: An Introduction*. MIT Press.

[37] Huazheng Wang, Qingyun Wu, and Hongning Wang. 2016. Learning hidden features for contextual bandits. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. 1633–1642.

[38] Huazheng Wang, Qingyun Wu, and Hongning Wang. 2017. Factorization bandits for interactive recommendation. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*.

[39] Lu Wang, Chengyu Wang, Keqiang Wang, and Xiaofeng He. 2017. Biucb: A contextual bandit algorithm for cold-start and diversified recommendation. In *Proceedings of the IEEE International Conference on Big Knowledge (ICBK'17)*. IEEE, 248–253.

[40] Qing Wang, Chunqiu Zeng, Wubai Zhou, Tao Li, S. Sitharama Iyengar, Larisa Shwartz, and Genady Ya Grabarnik. 2018. Online interactive collaborative filtering using multi-armed bandit with dependent arms. *IEEE Trans. Knowl. Data Eng.* 31, 8 (2018), 1569–1580.

[41] Qing Wang, Chunqiu Zeng, Wubai Zhou, Tao Li, S. Sitharama Iyengar, Larisa Shwartz, and Genady Ya Grabarnik. 2018. Online interactive collaborative filtering using multi-armed bandit with dependent arms. *IEEE Trans. Knowl. Data Eng.* 31, 8 (2018), 1569–1580.

[42] Qingyun Wu, Naveen Iyer, and Hongning Wang. 2018. Learning contextual bandits in a non-stationary environment. In *Proceedings of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. 495–504.

[43] Qingyun Wu, Huazheng Wang, Quanquan Gu, and Hongning Wang. 2016. Contextual bandits in a collaborative environment. In *Proceedings of the 39th International ACM SIGIR Conference on Development in Information Retrieval*.

[44] Qitian Wu, Hengrui Zhang, Xiaofeng Gao, Peng He, Paul Weng, Han Gao, and Guihai Chen. 2019. Dual graph attention networks for deep latent representation of multifaceted social effects in recommender systems. In *Proceedings of the World Wide Web Conference*. 2091–2102.

[45] Xiao Xu, Fang Dong, Yanghua Li, Shaojian He, and Xin Li. 2020. Contextual-bandit based personalized recommendation with time-varying user interests. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI'20)*. 6518–6525.

[46] Xiaoying Zhang, Hong Xie, Hang Li, and John C. S. Lui. 2020. Conversational contextual bandit: Algorithm and application. In *Proceedings of the Web Conference*. 662–672.

[47] Xiaoxue Zhao, Weinan Zhang, and Jun Wang. 2013. Interactive collaborative filtering. In *Proceedings of the 22nd ACM International Conference on Information and Knowledge Management*. 1411–1420.

[48] Sijin Zhou, Xinyi Dai, Haokun Chen, Weinan Zhang, Kan Ren, Ruiming Tang, Xiuqiang He, and Yong Yu. 2020. Interactive recommender system via knowledge graph-enhanced reinforcement learning. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 179–188.

[49] Yu Zhu, Jinghao Lin, Shibi He, Beidou Wang, Ziyu Guan, Haifeng Liu, and Deng Cai. 2019. Addressing the item cold-start problem by attribute-driven active learning. *IEEE Trans. Knowl. Data Eng.* 32, 4 (2019), 631–644.

[50] Lixin Zou, Long Xia, Yulong Gu, Xiangyu Zhao, Weidong Liu, Jimmy Xiangji Huang, and Dawei Yin. 2020. Neural interactive collaborative filtering. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 749–758.

Received 2 May 2022; revised 16 July 2022; accepted 24 July 2022