Knowledge-Based Systems 111 (2016) 180–192

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

A survey of serendipity in recommender systems

Denis Kotkov*, Shuaiqiang Wang, Jari Veijalainen

University of Jyvaskyla, Department of Computer Science and Information Systems, P.O.Box 35, FI-40014 University of Jyvaskyla, Finland

ARTICLE INFO

Article history:
Received 5 December 2015
Revised 9 August 2016
Accepted 10 August 2016
Available online 11 August 2016

Keywords:
Novelty
Serendipity
Recommender systems
Evaluation metrics
Evaluation strategies
Algorithms

ABSTRACT

Recommender systems use past behaviors of users to suggest items. Most tend to offer items similar to the items that a target user has indicated as interesting. As a result, users become bored with obvious suggestions that they might have already discovered. To improve user satisfaction, recommender systems should offer serendipitous suggestions: items not only relevant and novel to the target user, but also significantly different from the items that the user has rated. However, the concept of serendipity is very subjective and serendipitous encounters are very rare in real-world scenarios, which makes serendipitous recommendations extremely difficult to study. To date, various definitions and evaluation metrics to measure serendipity have been proposed, and there is no wide consensus on which definition and evaluation metric to use. In this paper, we summarize most important approaches to serendipity in recommender systems, compare different definitions and formalizations of the concept, discuss serendipity-oriented recommendation algorithms and evaluation strategies to assess the algorithms, and provide future research directions based on the reviewed literature.

1. Introduction

Many online stores offer thousands of different products, such as movies, songs, books and various services. With a wide range of goods, a store is likely to have a product that fits user preferences. However, the growth of choice does not always lead to user satisfaction, as sometimes it is difficult to find a product a user would like to purchase due the overwhelming number of available products [1].

To overcome the flood of information, online stores have widely adopted recommender systems [2]. In this paper, the term *recommender system* (RS) refers to a software tool that suggests items interesting to users [1]. An item is "a piece of information that refers to a tangible or digital object, such as a good, a service or a process that an RS suggests to the user in an interaction through the Web, email or text message" [3]. For example, an item can refer to a movie, a song or even a friend in an online social network. A *user* is generally understood to mean a person who uses an RS for any purpose. For example, a user might be an individual looking for a movie to watch or a company representative looking for goods to purchase.

Recommendation algorithms can be divided into categories: non-personalized and personalized [4]. Non-personalized algorithms suggest the same items to each user. For example, recommendations based on the popularity of items can be considered non-personalized. Personalized algorithms suggest different items depending on the target user. In this case, two users might receive different recommendations. In this paper, we focus on personalized RSs.

As a rule, personalized RSs provide suggestions based on user profile. A user profile might include any information about a user, such as an ID, age, gender and actions the user has performed with items in the past. In this paper, the term *user profile* refers to a set of items rated by the user [3]. Generation of recommendations can be based on (1) ratings a target user and others have given to items, (2) attributes of items rated by the target user or (3) both ratings and attributes [5]. One RS might suggest items that could be interesting to users who like many items in common with the target user. Another might recommend items that have many common attributes with items liked by the target user. For example, if a user rates many comedies, an RS will suggest more comedies.

Currently, RSs are widely adopted for different purposes. The ultimate goal of services that host RSs is to increase turnover [1]. Depending on the application scenarios, ways to achieve the goal might differ. One RS might suggest expensive items over interesting ones for a particular user, as the profit of the hosting service depends on user purchases. The RS of a service that sells subscriptions might suggest items interesting to users to encourage users to visit the service again. The business goal of the RS thus may differ from the goals of the users of the service [6].

* Corresponding author.
E-mail addresses: deigkotk@student.jyu.fi (D. Kotkov), shuaiqiang.wang@jyu.fi (S. Wang), jari.veijalainen@jyu.fi (J. Veijalainen).

http://dx.doi.org/10.1016/j.knosys.2016.08.014
0950-7051/

1.1. Motivation

Users may interact with an RS for different purposes, such as expressing themselves, influencing others or just browsing the catalog of items. However, the main reason the majority of individuals would use an RS is to discover novel and interesting items. It is demanding to look for items manually among an overwhelming number of options [3,6,7].

Most recommendation algorithms are evaluated based on accuracy, which does not correspond to user needs, as high accuracy indicates that the algorithm has prediction power but neglects the novelty and unexpectedness of suggested items [3,8–10]. To achieve high accuracy, RSs tend to suggest items similar to a user profile [3,11]. Consequently, the user receives recommendations only of items in some sense similar to items the user rated initially when she/he started to use the service (the so-called overspecialization problem). This has been observed to lead to a low user satisfaction [3,6,11,12].

One of the ways to overcome the overspecialization problem is to increase the serendipity of an RS. At this point we tentatively consider an RS to be serendipitous if it is able to suggest novel, interesting and unexpected items to a particular user. We will elaborate various definitions in Section 2.2. Serendipity-oriented algorithms can be used to slightly expand user tastes [13] or they can be applied as an additional "surprise me" option of an RS [14]. It is challenging to suggest serendipitous items for the following reasons [3]:

- There is no consensus on the definition of serendipity in RSs, as many papers present different definitions and formalizations of the concept [3,15].
- It is not clear how to measure serendipity, as there is no agreement on the definition of the concept [12].
- Serendipity includes an emotional dimension, which is very subjective [9,16]. Consequently, it is difficult to infer reasons why a user considers an item to be serendipitous [2].
- Serendipitous suggestions are less frequent than relevant ones, as not only must serendipitous items be relevant, but also novel and unexpected to a user [3]. As a result, it is often more difficult to capture and induce serendipitous recommendations than relevant ones in an experiment [17].

1.2. Methods and logistics

In this paper, we focus on serendipity, a poorly investigated concept that remarkably influences user satisfaction [11,18,19]. We aim at answering the following research questions:

RQ1 What is serendipity in recommender systems? What makes certain items serendipitous for particular users?
RQ2 What are the state-of-the-art recommendation approaches that suggest serendipitous items?
RQ3 How can we assess serendipity in recommender systems?
RQ4 What are the future directions of serendipity in RSs?

To address the research questions, we collected articles that mention serendipity using Google Scholar[1], Scopus[2] and Web of Science[3].

1. *Preliminary analysis.* By investigating articles received by entering the query "serendipity in recommender systems," we inferred that researchers employ different definitions of serendipity and different metrics to measure serendipity in RSs [19–21].
2. *Serendipity definitions and metrics.* To answer *RQ1* and *RQ3*, we looked for papers that proposed definitions of serendipity in RSs and methods to measure it. We selected the first 20 articles retrieved by the search engines in response to the search queries "definition of serendipity in recommender systems," "serendipity in recommender systems," "measure surprise in recommender systems" and "unexpectedness in recommender systems." To find papers that propose definitions of the concept and methods to measure it, we paid attention to related works and evaluation metrics used in the articles (forward search [22]). Eventually, we found 18 qualifying articles, six of which presented serendipity evaluation metrics (more detail is provided in Section 5).
3. *Serendipity-oriented algorithms.* To answer *RQ2*, we looked for algorithm proposals that improve the serendipity metrics discovered, searching papers that cite articles with proposals of serendipity metrics (backward search [22]).
4. *Filter.* Many of the articles mention serendipity in RSs, but mainly focus on other topics. We therefore primarily selected relevant high-quality articles.

The goal of the paper is to give a definition of serendipity and an overview of existing recommendation approaches and evaluation metrics, and thus to guide and inspire future efforts on recommendation serendipity. The main contributions of this paper are summarized as follows:

- Review of definitions and formalizations of serendipity and related concepts, such as relevance, novelty and unexpectedness.
- Review and classification of serendipity-oriented recommendation algorithms.
- Review of evaluation strategies to assess serendipity in RSs.
- Future directions to improve serendipity in RSs.

This paper is organized as follows. We answer *RQ1* in Section 2, which is dedicated to the definition of serendipity in RSs. In Section 3, we review serendipity-oriented recommendation algorithms and answer *RQ2*. Sections 4 and 5 describe strategies to assess serendipity and evaluation metrics (*RQ3*), respectively. Finally, we provide future directions (*RQ4*) and conclude in Sections 6 and 7, respectively.

2. Concepts

The degree of uncertainty around the terminology related to serendipity requires us to set up vocabulary to be used throughout the paper. In this section we answer *RQ1* by defining serendipity in RSs and describing what kinds of items are serendipitous for particular users.

2.1. Components of serendipity

Serendipity is a complex concept which includes other concepts. In this section, we provide those related concepts, such as relevance, novelty and unexpectedness.

2.1.1. Relevance

Depending on a particular application scenario, we may consider different actions that a user performs with items to indicate his/her interest. For example, we might regard an item as relevant for a user if she/he has gaven it a high rating and/or if she/he has purchased it. Therefore, to combine different terms that mean the same in the context of this paper, we define a *relevant item* as an item that a user likes, consumes or is interested in, depending on the application scenario of a particular RS.

[1] https://scholar.google.com/.
[2] http://www.scopus.com/.
[3] https://webofknowledge.com/.

2.1.2. Novelty

The term *novel item* has different meanings that can be summarized as follows [3]:

1. *Item novel to an RS.* A recently added item that users have not yet rated [23] (so-called cold start problem [24]).
2. *Forgotten item.* A user might forget that she/he consumed the item some time ago [23].
3. *Unknown item.* A user has never consumed the item in his/her life [23].
4. *Unrated item.* An unrated by the target user regardless of whether she/he is familiar with the item [7].

In this paper, the term *novel item* corresponds to definition 3 due to the popularity of the definition in studies on serendipity [9,12,25,26]. A novel item is a relevant or irrelevant item that a target user has never seen or heard about in his/her life [23]. However, a user does not normally rate all the items with which she/he is familiar.

It is often necessary to estimate how probable it is that a user has already seen an item. A user is more likely to be familiar with items similar to what she/he consumes and popular among other users [6,27]. In this paper, popularity indicates how likely users are to be familiar with certain items. Popularity might correspond to a number of ratings assigned to an item in a dataset used by an RS. Popular items are widely recognized by an individual, since she/he might have heard about the items from others or mass media, such as TV, radio or popular web-sites. A user might also be aware of items that are similar to what she/he usually consumes. For example, a jazz lover is likely to attend a small jazz concert rather than a rock concert. By attending the jazz concert she/he is likely to discover some unpopular jazz music. Novelty therefore includes two subcomponents: *unpopularity* and *dissimilarity to a user profile* [13]. For example, a jazz lover is likely to consider a very unpopular jazz song and relatively popular classical song as novel. In contrast, the user might have already listened to a classical hit on the radio as well as a relatively popular jazz song in a jazz club.

2.1.3. Unexpectedness

Unexpectedness and *surprise* are terms frequently used in the literature [10,20,28,29], but to the best of our knowledge there is no consensus on the terms' definitions. In this paper, the terms *unexpected* and *surprising* refer to items that *significantly differ* from the profile of the user regardless of how novel or relevant those items are. For example, a classical song recommended to a jazz lover is likely to be more unexpected than a jazz song.

There are two differences between novelty and unexpectedness. First, a user may find an item unexpected even if she/he is familiar with the item. Second, to surprise a user, unexpected items have to differ from the user profile more than novel items.

2.2. Concept of serendipity

Serendipity is a difficult concept to study [3], as it includes an emotional dimension [9,16]. It is challenging to define serendipity in recommender systems as well as what kind of items are serendipitous and why [3,15], since generally serendipitous encounters are very rare [17].

The term *serendipity* has been recognized as one of the most untranslatable words [30]. The first known use of the term was found in a letter by Horace Walpole to Sir Horace Mann on January 28, 1754. The author described his discovery by referencing a Persian fairy tale, "The Three Princes of Serendip". The story described a journey taken by three princes of the country Serendip to explore the world [31]. In the letter, Horace Walpole indicated that the princes were "always making discoveries, by accidents & sagacity, of things which they were not in quest of" [32].

According to the dictionary [33], serendipity is "the faculty of making fortunate discoveries by accident." The word "discovery" indicates the novelty of serendipitous encounters, while the word "fortunate" indicates that the discovery must be relevant and unexpected.

2.2.1. Serendipity in a computational context

As serendipity is a complicated concept, Van Andel discussed whether a computer can generate serendipitous information [34]:

> *[P]ure serendipity is not amenable to generation by a computer. The very moment I can plan or programme 'serendipity' it cannot be called serendipity anymore. All I can programme is, that, if the unforeseen happens, the system alerts the user and incites him to observe and act by himself by trying to make a correct abduction of the surprising fact or relation.*

The author suggests that computer programs cannot generate but can help a user find serendipitous information. Based on [34], we suppose that as recommender systems are designed to "assist and augment" a process of making "choices without sufficient personal experience of the alternatives" [35], they can also assist a process of making "fortunate discoveries" [33].

Corneli et al. investigated serendipity in a computational context and proposed a framework to describe the concept [36]. The authors considered *focus shift* a key condition for serendipity, which happens when something initially perceived irrelevant, neutral or even negative becomes relevant. According to the authors, the concept of serendipity includes four components:

- *Prepared mind* (focus on something an explorer is looking for). The component indicates that an explorer should be focused on what she/he wants to investigate.
- *Serendipity trigger* (inspiration for a novel result). The act of drawing initial attention to a serendipitous object or phenomenon.
- *Bridge* (inference). Investigation of a discovered object or phenomenon.
- *Result* (an outcome). This can be a new artifact, process, phenomenon, hypothesis or problem.

The framework can be illustrated using the example of the microwave oven invention. Once Percy Spencer was working with a radar set and noticed that a candy bar in his pocket had melted. Later, the researcher investigated this phenomenon and invented the first microwave oven. In the example, a *focus shift* happened when the once uninteresting melted candy bar attracted the researcher's attention. Percy Spencer had a *prepared mind*, as he was initially working with radar. The researcher experienced a *serendipity trigger* when he noticed that the candy bar had melted. A *bridge* occurred when he investigated why it had melted. Finally, the *result* is the invention of the microwave oven.

2.2.2. Serendipity in the context of recommender systems

The framework proposed by Corneli et al. can also apply to RSs. In this case, a *focus shift* happens when initially uninteresting items become interesting. A user has a *prepared mind*, as she/he expects that an RS will suggest items similar to his/her profile [10]. The attention of the user drawn by the items represents a *serendipity trigger*. The investigation of the items by the user is a *bridge*. Finally, the *result* for the user is the obtained knowledge.

According to the framework, an item serendipitous for a user should meet certain requirements. First, the serendipitous item is required to be unexpected and relevant to create a *focus shift*. A user is likely to consider an item irrelevant at first sight if the item is significantly different from his/her profile (unexpected). Meanwhile, the item is required to be relevant to eventually become interesting for the user. To create a *serendipity trigger*, the item must

Table 1
Definitions of serendipity in related works.

Articles	Definition of serendipity	Components
McNee et al. [29]	"serendipity in a recommender is the experience of receiving an unexpected and fortuitous item recommendation. But even if we remove that component, the unexpectedness part of this concept the novelty of the received recommendations is still difficult to measure"	Relevance, novelty and unexpectedness
Adamopoulos and Tuzhilin [10,28]	"serendipity, the most closely related concept to unexpectedness, involves a positive emotional response of the user about a previously unknown (novel) [...] serendipitous recommendations are by definition also novel."	Relevance, novelty and unexpectedness
Sridharan [37]	"serendipity is defined as the accident of finding something good or useful while not specifically searching for it. Serendipity is thus closely related to unexpectedness and involves a positive emotional response of the user about a previously unknown item. It measures how surprising the unexpected recommendations are [...] serendipity is concerned with the novelty of recommendations and in how far recommendations may positively surprise users"	Relevance, novelty and unexpectedness
Zhang et al. [19]	"serendipity represents the "unusualness" or "surprise" of recommendations"	Novelty and unexpectedness
Maksai et al. [39]	"Serendipity is the quality of being both unexpected and useful."	Relevance and unexpectedness

be novel for the user; otherwise the user would not pay attention to the item. As a result, to be serendipitous for a user, items must be relevant, novel and unexpected.

Many authors indicate that novelty, relevance and unexpectedness are important for serendipity, yet these authors often do not define these concepts, which might cause confusion [8,9,21,28,29,37]. The majority of papers include each of these three components to the definition of serendipity [9,28,29,37]. One example is the definition proposed by Iaquinta et al. [26]:

> *A serendipitous recommendation helps the user to find a surprisingly interesting item that she might not have otherwise discovered (or it would have been really hard to discover). [...] Serendipity cannot happen if the user already knows what is recommended to her, because a serendipitous happening is by definition something new. Thus the lower is the probability that user knows an item, the higher is the probability that a specific item could result in a serendipitous recommendation.*

Iaquinta et al. suggest that serendipitous items are always relevant, novel and unexpected to a user [26]. In contrast, Kaminskas and Bridge consider even items a user is aware of to be serendipitous [21]. Their definition includes only relevance and unexpectedness [21]:

> *Serendipity consists of two components - surprise and relevance [...] we also do not require a surprising item to be novel, but only different from the user's expectations, which are represented by a set of items.*

Another definition was proposed by Chiu et al. [38]. The authors proposed completely different definitions adopted towards online social networks. According to Chiu et al. [38], a serendipitous item is novel for a target user, but familiar to the user's friends, defined as contacts in an online social network: "Items are not yet accessed (implicit) or rated (explicit). But the user's friends (in the social network) have accessed before" [38]. Table 1 provides additional summarized examples of definitions of serendipity.

To answer *RQ1*, in this paper, we refer to serendipity as a property of an RS that indicates how good the RS is at suggesting serendipitous items [7], where *serendipitous items* are relevant, novel and unexpected for a user [3,10,12,26,29,37].

In terms of popularity and similarity to a user profile, serendipitous items are limited by novelty and unexpectedness. First, as serendipity includes novelty, serendipitous items should be unpopular [40] and dissimilar to a user profile [19], otherwise the user is likely to be familiar with them (Section 2.1.2). Second, the unexpectedness component requires serendipitous items to significantly differ from a user profile (Section 2.1.3).

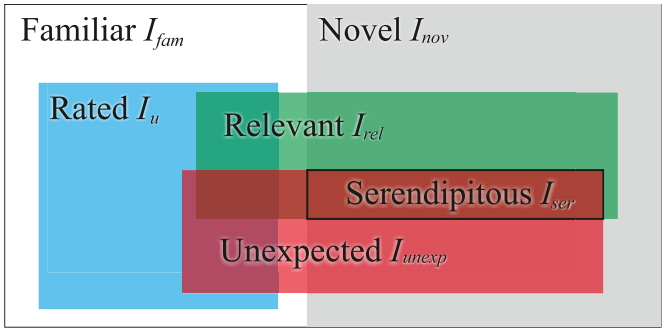

Fig. 1. Euler diagram of items from a user's point of view at a given moment of time (snapshot).

Fig. 1 illustrates set intersections of familiar, novel, rated, relevant, unexpected and serendipitous items perceived by a user at a particular moment of time. Suppose that an RS contains I items and a user is familiar with items $I_{fam} \subseteq I$. The items that the user has never heard of are novel $I_{nov} = I \setminus I_{fam}$. The RS has information that the user is aware of items she/he rated $I_u \subseteq I_{fam}$. We assume that the user rates only items with which she/he is familiar. Items interesting to the user are relevant $I_{rel} \subseteq I$. They are distributed among the categories familiar, rated, novel and unexpected. Unexpected items $I_{unexp} \subseteq I$ differ from items the user rated and can be novel or familiar to the user. In rare cases, unexpected items can also be rated. Rated items are likely to be unexpected if they differ from the rest of the rated items. Serendipitous items are relevant, novel and unexpected $I_{ser} = I_{rel} \cap I_{nov} \cap I_{unexp}$. The task of the serendipity-oriented RS is to suggest to the user serendipitous items I_{ser} based on rated items I_u.

The proposed restrictions of serendipitous items can have exceptions. For example, a user might consider a popular item novel and serendipitous. We apply these restrictions for simplification and assume they are reasonable in most cases.

In general, the increase of serendipity decreases accuracy for two reasons. First, as users tend to like items similar to items they already find interesting, many items different from the users' profiles are irrelevant. Second, considering the number of low-quality items among unpopular ones [6], increasing serendipity is likely to increase the number of irrelevant items recommended. An algorithm has to offer risky suggestions to increase serendipity.

Depending on user needs, requirements for serendipity-oriented algorithms might differ. A user might want to discover relevant items. In this case, it is important to keep a trade-off between accuracy and serendipity [19,21,41]. However, a user might

look for serendipitous items in particular. For example, a "surprise me" algorithm could be offered to the user as an additional option [14]. In this case, it might be more important to increase serendipity regardless of accuracy.

2.2.3. Difference between serendipity and similar concepts

It is important to clarify the difference between novelty and serendipity. First, serendipitous items are relevant, while novel items can be irrelevant. Second, as serendipitous items are unexpected, they significantly differ from items a user rated in an RS, which is not necessary for novel items. For example, let us assume a jazz lover logs into an RS to discover a few songs. A suggestion of a jazz song that the user is unaware of, regardless of whether she/he likes it, is likely to be novel, but not serendipitous.

Another concept which might be confused with serendipity is diversity. The concept reflects how different items are from each other [7,27]. Diversity is a desirable property in an RS, as users often become bored with recommendation lists containing items very similar to each other. For example, a poker player might enjoy a recommendation list of books about poker, combinatorics and math more than a list containing only poker books.

There are two key differences between diversity and serendipity. First, diversity is based on the comparison of recommended items between each other, while serendipity is based on the comparison of recommended items and a user profile [27]. Second, diversity does not include relevance, a necessary component for serendipity.

2.2.4. Conclusions and discussion

In this section, we answered *RQ1* based on the reviewed literature related to serendipity in RSs. Serendipity is a property that indicates how good an RS is at suggesting serendipitous items that are relevant, novel and unexpected for a particular user. Serendipitous items are by definition unpopular and significantly different from the user profile.

It is worth mentioning that serendipity is not always a necessary component in suggesting items. Some services might need an RS to suggest only relevant items, regardless of how serendipitous those items are. For example, an RS that suggests friends might be required to offer only those individuals with whom a target user is familiar. Furthermore, users might prefer obvious suggestions over serendipitous ones [42]. For example, user intention to receive risky recommendations might increase with the growth of experience of using the system [6]. However, in many cases, serendipity is perceived as a purely positive property of an RS [28,29,41].

3. Approaches

In the section, we are going to review serendipity-oriented algorithms and answer *RQ2*. Recommendation algorithms can be divided into the following categories [43].

- *Content-based filtering* (CBF) utilizes items' attributes to generate recommendations [43]. This kind of algorithm considers choices a user has made in the past to suggest items the user will consume in the future. For example, if a user has watched a comedy movie, then a CBF algorithm would suggest another comedy movie to the user.
- *Collaborative filtering* (CF) considers only ratings users have gaven to items [43]. For example, if two users have rated many common items similarly, then a CF algorithm would probably recommend items highly rated by the first user that the second user has not yet rated and vice versa.
  - *Memory-based CF* typically uses similarity measures based on user ratings [5,43]. This approach generates recommendations for a target user based on what similar users have rated or based on items similar to items the target user has consumed.
  - *Model-based CF* utilizes a model that receives user ratings as input and generates recommendations [5,43].
- *Hybrid filtering* takes into account both ratings and attributes of items [43–45].

Serendipity-oriented RSs can be classified depending on the data they use (CF, CBF, hybrid) or on the architecture of recommendation algorithms. We divide serendipity-oriented approaches into three categories:

- *Reranking algorithms (Reranking).* To improve serendipity, an algorithm can leverage ratings predicted by an accuracy-oriented approach and rerank the output of an RS. An accuracy-oriented algorithm finds items relevant for a target user, while a reranking algorithm assigns low ranks to obvious suggestions.
- *Serendipity-oriented modification (Modification).* This category represents modifications of accuracy-oriented algorithms. The main difference between modification and reranking is that reranking algorithms can use any accuracy-oriented algorithms that assign ranking scores to items, while modifications can only be applied to certain kinds of algorithms. An example is the k-furthest neighbor approach [18], which is a modification of the k-nearest neighbor algorithm (user-based CF). Instead of suggesting items liked by users similar to a target user, a k-furthest neighbor algorithm recommends items disliked by users dissimilar to a target user.
- *Novel algorithms (New).* This category includes serendipity-oriented algorithms that are not based on any common accuracy-oriented algorithms. Approaches from this category are very diverse and use different techniques, such as clustering [46] and random walk with restarts [8,14]. One of the examples is the TANGENT algorithm, which detects groups of like-minded users and suggests items simultaneously liked by users from the group of the target user and other groups [14]. Recommended items are related to previous choices of the user and likely to be surprising, as these items are chosen by users from a group different than the one of the target user.

We may take actions to improve the serendipity of recommendation at different stages of the recommendation process. Similarly to [47], we suggest three paradigms for serendipity-oriented recommendation algorithms:

- Pre-filtering: A recommendation algorithm preprocesses the input data for an accuracy-oriented algorithm to improve serendipity.
- Modeling: A recommendation algorithm improves serendipity in the phase of generating recommendations.
- Post-filtering: A recommendation algorithm reranks the results of accuracy-oriented algorithms.

Table 2 demonstrates classification of the state-of-the-art serendipity-oriented approaches. The majority of approaches are either modifications or of uncertain basis. In terms of the data that algorithms use, we reviewed at least one algorithm from each category (CF, CBF and hybrid). Serendipity subcomponents include popularity (*pop*) and similarity to a user profile (*sim*), which the algorithms take into account. By assessing the parameter, we examined whether the model takes into account the subcomponents. Among the approaches presented, the algorithms proposed by Zheng et al. [13] and Zhang et al. [19] capture both popularity and similarity to a user profile.

To review serendipity-oriented algorithms, we present notation in Table 3. Let I be a set of available items and U be a set of users. User u rates or interacts with items I_u, $I_u \subseteq I$. A recommender system suggests R_u items to user u. Each item i, $i \in I$ is represented as

Table 2
Classification of serendipity-oriented algorithms (MF - matrix factorization, kNN - k-nearest neighbor, unpop - unpopularity, dissim - dissimilarity to a user profile).

Name	Approach type	Serendipity subcomponents	Paradigm	Recommendation generation
Utility Model [10]	Hybrid	dissim	Post-filtering	Reranking
Full Auralist [19]	Model-based CF	unpop + dissim	Post-filtering	Reranking
K-Furthest Neighbor [18]	Memory-based CF	unpop	Pre-filtering	Modification of kNN
kNN with item taxonomy[48]	Hybrid	dissim	Modeling	Modification of kNN
Search-time-based-RS [49]	Hybrid	unpop	Modeling	Modification of kNN
Unexpectedness-Augmented Utility Model [13]	Model-based CF	unpop + dissim	Modeling	Modification of MF
Serendipitous Personalized Ranking [41]	Model-based CF	unpop	Modeling	Modification of MF
Distance-based Model [46]	CBF	dissim	Modeling	New
TANGENT [14]	Model-based CF	dissim	Modeling	New
RWR-KI [8]	CBF	unpop	Modeling	New

Table 3
Notations.

$I = (i_1, i_2, \ldots, i_n)$	the set of items
$F = (f_1, f_2, \ldots, f_z)$	feature set
$i = (f_{i,1}, f_{i,2}, \ldots, f_{i,z})$	representation of item i
$U = (u_1, u_2, \ldots, u_m)$	the set of users
I_u, $I_u \subseteq I$	the set of items rated by user u (user profile)
R_u, $R_u \subseteq I$	the set of items recommended to user u
$rel_u(i)$	1 if item i relevant for user u and 0 otherwise
U_i	the set of users who rated item i

a vector $i = (f_{i,1}, f_{i,2}, \ldots, f_{i,z})$ in a multidimensional feature space F. For example, a feature can be a genre of a movie on a web-site. If $F = (drama, comedy, romance)$ then the movie "Forrest Gump" can be represented as $i_{Forrest} = (0.4, 0.2, 0.4)$, where the numbers indicate how strongly the movie belongs to the genre.

3.1. Reranking

This section is dedicated to serendipity-oriented recommendation algorithms that rerank results of accuracy-oriented ones to increase serendipity.

3.1.1. Utility model

Adamopoulos and Tuzhilin proposed a serendipity-oriented algorithm that reranks the results of any accuracy-oriented algorithm that produces relevance scores [10]. The algorithm assigns overall scores to items based on two metrics: (1) unexpectedness utility and (2) quality utility. Unexpectedness utility indicates obviousness of items for a user, while quality utility reflects their relevance for the user. The algorithm consists of the following steps:

1. Computation of the set of expected items E_u;
2. Filtering of items that are likely to be irrelevant based on the relevance score provided by an accuracy-oriented algorithm;
3. Filter of items that are too obvious for a user based on expected items E_u;
4. Overall utility calculation;
5. Recommendation of items based on the overall utility.

The set of expected items E_u is represented by items previously consumed by a user I_u or by items similar or related to I_u, where relatedness is inferred by various data mining techniques. In steps 2 and 3, the algorithm uses a threshold to remove items obvious or irrelevant to a user. On the last step, the algorithm suggests items with the highest overall utility to the user. Overall utility consists of unexpectedness utility and quality utility. Unexpectedness utility depends on the average distance between an item and user expectations E_u, where the distance is based on user ratings (collaborative) and on item attributes (content-based). Quality utility is based on ratings provided by an accuracy-oriented algorithm.

The proposed algorithm seems to recommend more items that differ from items a target user usually consumes due to filtering based on user expectations. However, the algorithm might still recommend popular items, as it does not explicitly take popularity into account.

3.1.2. Full auralist

Seeking to improve diversity, novelty and serendipity while keeping acceptable accuracy, Zhang et al. proposed a recommendation algorithm called Full Auralist [19]. Full Auralist consists of three algorithms:

- *Basic auralist* is responsible for the relevance of recommendations.
- *Listener diversity* provides a diversified recommendation list.
- *Declustering* is designed to suggest items unexpected for a user.

Each algorithm returns a ranked list of items. *Full Auralist*, in turn, integrates generated ranks using their linear combination.

The algorithm appears to capture both the popularity of items and their similarity to a user profile. However, Full Auralist might be quite difficult to implement, as it contains three different algorithms.

3.2. Modification

This section is dedicated to serendipity-oriented recommendation algorithms based on accuracy-oriented ones.

3.2.1. K-furthest neighbor

Seeking to improve novelty, Said et al. has proposed k-furthest neighbor (kFN) recommendation algorithm, similar to kNN [18]. As kNN is biased towards popular items, which results in poor personalization, kFN is designed to overcome this problem. Instead of suggesting items that neighborhood users like, kFN forms neighborhoods of users dissimilar to a target user. By selecting items dissimilar users dislike, kFN is supposed to overcome the bias of items liked by the majority [18].

By overcoming the popularity bias, the algorithm seems to suggest more novel items, at least with respect to kNN. However, it is questionable whether it recommends serendipitous items, as it does not explicitly improve the dissimilarity of recommendations to a target user profile.

3.2.2. Knn with item taxonomy

Nakatsuji et al. designed a novelty-oriented recommendation algorithm [48]. The authors modified kNN by replacing the similarity measure with *relatedness*. By forming a neighborhood of related users and picking items from their profiles, the algorithm is supposed to improve the novelty of recommendations, over that of a classical kNN.

The algorithm is a modification of kNN, which calculates ratings based on relatedness. Relatedness is inferred by utilizing random walk with restarts (RWR) on a user similarity graph [50]. In RWR a random particle travels from node to node with a probability equivalent to an edge weight. During each step, the particle has a probability to return to its starting node. The particle visits nodes a different number of times depending on the starting node. After a sufficient number of random walks, the ratio of the number of transitions and number of visits of a certain node stabilizes. The obtained probabilities indicate relatedness between starting node and other nodes.

In a user similarity graph, nodes correspond to users, while edges correspond to similarities. User similarities are based on a taxonomy of items. The similarities are calculated considering items and classes of items that a user rated.

By picking users who are dissimilar but related to a target user, the algorithm seems to suggest more items that are dissimilar to the target user profile. Furthermore, user similarity calculation might be especially useful with a rich taxonomy of items. However, even related users might still give high ratings to popular items, which might result in suggestions with which a target user is already familiar.

3.2.3. Search-time-based-RS

Kawamae proposed a recommendation algorithm based on estimated search time [49,51]. The author employed the following definition of serendipity: "a serendipitous recommendation helps the user find a surprisingly interesting item he might not have otherwise discovered" [49]. In other words, the more difficult it is to find an item, the more probable that the item is serendipitous to a user.

The algorithm consists of three steps [49]. First, for each target user the algorithm detects similar users (innovators) who have common tastes and who discover recently released items better than the target user. Second, the algorithm measures how likely it is that a target user will consume a particular item from a profile of an innovator. Third, the algorithm combines these two probability parameters into ranking score and sorts a suggestion list.

The advantage of the algorithm is that it takes into account when items were released and consumed by users. In terms of similarity to a user profile and popularity, the algorithm seems to suggest unpopular items, as personal innovator probability gives high scores to recently released items that have not yet become popular. However, as the algorithm searches for users who have rated many common items with the target user, it is likely that a recommended item would be similar to a target user profile.

3.2.4. Unexpectedness-augmented utility model

Zheng et al. proposed a recommendation algorithm [13] based on PureSVD (variation of singular value decomposition (SVD)) [52]. The main difference between PureSVD and the proposed algorithm is the optimization task:

$$min \sum_{u \in U} \sum_{i \in I} (\tilde{r}_{u,i} - p_u \cdot q_i^T)^2 w_{u,i} + \lambda (|p_u|^2 + |q_i|^2) \tag{1}$$

$$w_{u,i} = \left(1 - \frac{|U_i|}{|U|}\right) + \frac{\sum_{j \in I_u} dist(i,j)}{|I_u|}, \tag{2}$$

where p_u and q_i are user and item feature vectors, respectively, while $\tilde{r}_{u,i}$ corresponds to the ratings user u gave to item i with the absence of a rating represented by zero. The set of users who rated item i is represented by U_i. The weight $w_{u,i}$ increases the contribution of unexpected items, which forces a gradient descent algorithm to pay additional attention to the ratings of those items [13].

The proposed algorithm captures both popularity and similarity to a user profile. Consequently, it suggests more novel and serendipitous items than PureSVD.

3.2.5. Serendipitous personalized ranking

Lu et al. suggested a serendipitous personalized ranking algorithm (SPR), based on SVD. The objective of SPR is to maximize the serendipitous area under the ROC (receiver operating characteristic) curve (SAUC). SAUC is based on AUC (area under the curve) and defined as follows [41]:

$$SAUC(u) = \frac{1}{|I_u^+||I \setminus I_u^+|} \sum_{i \in I_u^+} \sum_{j \in (I \setminus I_u^+)} \sigma(\hat{r}_{u,i} - \hat{r}_{u,j})(pop(j))^{\alpha} \tag{3}$$

where σ is a 0–1 loss function: $\sigma(x) = 1$ if $x > 0$, $\sigma(x) = 0$ otherwise, while $\hat{r}_{u,i}$ is a preference prediction of user u for item i. Relevant items are represented by I_u^+, consequently, $I \setminus I_u^+$ corresponds to unrated and irrelevant items. Popularity weight corresponds to $pop(i)$, $pop(i) \propto |U_i|$.

The proposed algorithm appears to have high novelty due to the popularity parameter. However, the algorithm might not suggest serendipitous items, as it disregards dissimilarity to a user profile.

3.3. New

This section is dedicated to serendipity-oriented recommendation algorithms that are not based on any common accuracy-oriented algorithms.

3.3.1. Distance-based model

Akiyama et al. proposed a content-based algorithm to improve the serendipity of an RS. First, the recommendation algorithm clusters items from a user profile based on item attributes. Second, the algorithm assigns scores to unrated items based on their distance from discovered clusters and on how unexpected they are for the user. Finally, the algorithm ranks unrated items according to their scores and suggests the recommendation list to the user [46].

The algorithm does not have many parameters, which makes it easy to control. Besides, an algorithm's content-based nature might positively result in computational time depending on the dataset. However, the algorithm disregards the popularity subcomponent of serendipity.

3.3.2. TANGENT

Onuma et al. proposed the TANGENT algorithm to broaden user tastes [14]. The algorithm performs on a bipartite graph, where users and items correspond to nodes, while ratings correspond to edges. TANGENT detects groups of likeminded users and suggests items relevant to users from different groups. For example, if a target user belongs to comedy fans, the algorithm will suggest a movie relevant not only to comedy fans, but also to users from other groups, such as action fans or romance fans.

The proposed algorithm appears to broaden user tastes and retain the accuracy of recommendations. However, TANGENT might suggest popular items that users already know well, as items liked by users from different groups are likely to be popular.

3.3.3. RWR-KI

De Gemmis et al. proposed a serendipity-oriented RWR-KI algorithm [8]. RWR-KI stands for *random walk with restarts enhanced with knowledge infusion*. The algorithm suggests items based on the results of RWR that exploits an item similarity graph, where similarity indicates relatedness of items. The authors used WordNet and Wikipedia to calculate item similarities. By using uncommon similarity measure, the algorithm is supposed to suggest serendipitous items.

The algorithm suggests items based on the result of RWR, which receives a graph of item similarities and a set of starting nodes as an input. The item similarity graph contains nodes that correspond to items, and weighted edges, where weights correspond to similarities between connected items. Item similarities are inferred using item descriptions, WordNet and Wikipedia. Given an item similarity graph and a set of starting nodes I_u (items rated by a user), RWR returns relatedness between items and a user profile. Items the user has not yet rated are ranked according to inferred relatedness values and suggested to the user.

The RWR-KI algorithm might suggest serendipitous items, as the algorithm uses an uncommon similarity measure that indicates relatedness of items instead of similarity. Furthermore, as the algorithm does not depend on user ratings, recommendations seem to be unbiased towards popular items. However, suggested items might still be similar to a user profile, since they might have similar descriptions. Besides, RWR-KI requires information enrichment using sources, such as WordNet or Wikipedia, which requires additional effort.

4. Evaluation strategies

To address *RQ3*, in this section we discuss evaluation strategies used to assess serendipity in recommender systems (RSs). We review offline and online evaluation strategies.

Depending on the specific domain, it is necessary to choose a suitable objective for an RS. For example, the goal of an RS that suggests friends might be to offer items with which a user is likely to be familiar. However, a music RS might suggest only novel items. It is crucial for the friend RS to have high accuracy. Meanwhile, the music recommendations require both accuracy and novelty [6]. Requirements of an RS therefore influence the choice of evaluation metrics and the decision on a recommendation algorithm [7]. According to different objectives, the evaluation metrics of the recommendation algorithms could be different as well.

Traditionally, evaluation strategies are divided into two categories: offline evaluation and online evaluation. In offline evaluation, researchers conduct experiments using pre-collected or simulated datasets. In the experiments, researchers hide some ratings and let an algorithm predict them based on the rest of the data. The performance of algorithms is usually presented by evaluation metrics. Based on results, researchers may choose a suitable algorithm [2,7,53].

Online evaluation involves real users interacting with a recommender system. For example, users might explicitly indicate whether they find recommended items novel or interesting. Online experiments can be conducted by using an RS prototype or a deployed functioning RS [2,7].

Usually, online evaluation is demanding and complicated, as it involves real users. Researchers, therefore, conduct offline experiments prior to the implementation of an online RS [7].

4.1. Offline evaluation

In offline experiments, researchers use pre-collected implicit or explicit user preferences to simulate the behavior of users and assess recommendation algorithms. Even though this kind of experiment only lets researchers answer a limited number of questions normally related to the prediction power of algorithms, it is often necessary to conduct offline experiments prior to online evaluation to decrease the number of candidate algorithms [7]. Regarding the evaluation of serendipity, many papers provide both qualitative and quantitative analysis of recommendation algorithms [7,21].

4.1.1. Quantitative analysis

In offline experiments, researchers select a number of test users and hide some of their ratings. The hidden data are regarded as a test set and the rest as a training set. The candidate algorithms receive the training data as an input and predict the test set. The researchers filter candidate algorithms by their predictive ability [2,7].

Current datasets contain data regarding items relevant to a particular user, but lack data regarding serendipitous items. To assess serendipity in offline experiments, researchers make assumptions regarding serendipitous items and measure serendipity using the metrics described in Section 5.

4.1.2. Qualitative analysis

It is possible to manually assess the serendipity of recommendations by comparing recommended items and user profiles. For example, Kaminskas and Bridge presented a qualitative analysis of suggestion lists offered by three popular recommendation algorithms [21]. One of the goals in the paper was to propose evaluation metrics for unexpectedness. To support the results of quantitative analysis, the authors compared a target user profile and recommended items for the user, including values of the proposed evaluation metrics. The items in user profiles and the recommended items referred to music artists.

4.2. Online evaluation

We regard online evaluation as evaluation that involves users interacting with the recommender system. This kind of evaluation may have different forms, such as questionnaires and experiments where users interact with a deployed RS [7].

Conducting online experiments is especially important for an assessment of novelty and serendipity, since it allows asking a user if she/he finds a recommended item unexpected or novel. On the one hand, these kinds of experiments are more representative than offline experiments, since they involve real users interacting with the RS [29]. On the other hand, online experiments are more demanding to conduct [7].

The main challenge in assessing serendipity in an online experiment is that it is difficult for a user to detect serendipitous items, since they are new to a user. Many papers have used different methods to investigate serendipity and novelty in RSs. In this section, we review the most common approaches mentioned in papers dedicated to serendipity.

4.2.1. Questionnaire

Many RSs provide suggestions and explicitly ask users whether they find an item serendipitous [18–20,54]. A user might detect relevant and novel items, but it is difficult for the user to determine whether an item is unexpected, since the concept of unexpectedness is quite vague and might depend on many factors, such as mood, time of day and experimental setting. It is important to formulate a question about serendipity that a user would understand. For example, in [18] the authors asked participants whether they thought the recommendations were serendipitous. The results of the experiment were not statistically significant. The authors assumed this happened because most users were not sure what serendipity meant, as the term is difficult to translate to other languages. Table 4 reviews questions that were used in questionnaires to assess the serendipity of recommended items.

Regarding novelty, it is possible to utilize the setting suggested in [25]. In the context of music recommendation, a user might be offered a list of songs without any metadata. The user would listen to each song for 30 s and then inform a researcher whether she/he knows and likes the song.

Table 4
Serendipity assessment in experiments.

Article	Application	Question
[19]	Music songs	"Exactly what I listen to normally... Something I would never have listened to otherwise" (Likert scale)
[54]	Music artists	"Did you find artists you wouldn't have found easily on your own and which you would like to listen to from now on?"
[20]	TV shows	"We inquired about serendipity for recommended shows, which means whether the recommended shows had hardly ever been heard of before by viewers but seemed to be interesting or whether they were already known but were unexpected"
[18]	Movies	"Are the recommendations serendipitous?"
[55]	IBM Connections	Participants indicated whether an item is surprising to them.
[8]	Movies	Analysis of facial expressions

4.2.2. Qualitative analysis

It might be beneficial to ask users to give feedback on the recommendation list as a whole. Users might indicate what they liked or disliked the most or what they would improve and why [19,56]. For example, Zhang et al. presented a qualitative analysis based on users' comments on an RS [19]. The authors proposed a recommendation approach which was evaluated online. In the experiment, different RSs suggested music artists to users. Users not only rated suggested artists but also left comments on provided recommendation lists. The authors analyzed the comments and inferred factors that affected user perception of recommendation lists in the experiment.

4.2.3. Deployed RS

Sometimes it is necessary to improve an algorithm inside an already existing and deployed RS. Users can be divided into groups and introduced to the output of different recommendation algorithms. In the settings, researchers assume that users are not aware of the experiment. By analyzing user behavior, researchers make a decision between algorithms [7].

Assessing the impact of serendipitous items on user satisfaction in an online experiment with a deployed RS might be challenging for two reasons. First, by introducing serendipitous items, an RS might affect users differently depending on their level of experience [6]. Second, it might be difficult to detect an impact of serendipitous items on user behavior due to the emotional dimension of serendipity [9,16]. However, to the best of our knowledge, the number of experiments on serendipity in RSs that involve deployed RS is very limited.

5. Formal definitions of serendipity

To answer *RQ3*, in this section, we are going to review evaluation metrics that measure serendipity in recommender systems (RSs). The section provides a comparison of different metrics that have been proposed [19–21], the section provides their comparison, including their advantages and disadvantages.

Serendipity metrics can be divided into two categories: component metrics and full metrics. Component metrics measure different components of serendipity, such as novelty or unexpectedness, while full metrics measure serendipity as a whole.

5.1. Component metrics

Each component of serendipity can be measured in different ways. In this section, we are going to review serendipity component metrics that measure novelty and unexpectedness.

5.1.1. Novelty

Vargas and Castells proposed two novelty metrics [27]. The first metric is based on an items distance from items a user has consumed (user profile) and has two variations:

$$nov_{va}^{dist1}(i,u) = \min_{j \in I_u} dist(i,j) \tag{4}$$

$$nov_{va}^{dist2}(i,u) = \frac{1}{|I_u|}\sum_{j \in I_u} dist(i,j) \tag{5}$$

where $dist(i, j)$ indicates the distance between items i and j and is formalized as follows:

$$dist(i,j) = 1 - sim(i,j), \tag{6}$$

where $sim(i, j)$ is any kind of similarity between items i and j ($sim(i, j) \in [0, 1]$). For example, it might be content-based cosine distance [57]. The second novelty metric is based on popularity and has two variations:

$$nov_{va}^{pop1}(i,u) = 1 - \frac{|U_i|}{|U|}, \tag{7}$$

$$nov_{va}^{pop2}(i,u) = -\log_2 \frac{|U_i|}{|U|}. \tag{8}$$

Nakatsuji et al. proposed a novelty metric similar to Eq. (4) [48]. They considered a taxonomy of items, where an item belongs to one of classes $CLS = (cls_1, cls_2, \ldots, cls_t)$. The novelty metric is based on a minimum distance between an item and a user profile in an item taxonomy.

$$nov_{na}(i,u) = \min_{j \in I_u}(dis(cls_j, cls_i)), \tag{9}$$

where cls_k is a class of item k in an item taxonomy, while $dis(cls_j, cls_i)$ is the distance between classes cls_j and cls_i in the taxonomy.

5.1.2. Unexpectedness

Kaminskas and Bridge suggested that serendipity includes two components: unexpectedness and relevance [21]. The authors indicated that unexpectedness reflects how dissimilar a suggested item is to a user profile, while relevance can be measured by accuracy metrics [2]. To measure unexpectedness, Kaminskas and Bridge suggested two pair-wise similarity metrics [21]: (1) point-wise mutual information and (2) content-based similarity.

Point-wise mutual information indicates how similar two items are based on the numbers of users who have rated both items and each item separately:

$$PMI(i,j) = -\log_2 \frac{p(i,j)}{p(i)p(j)} / \log_2 p(i,j), \tag{10}$$

where $p(i)$ is the probability that any user has rated item i, while $p(i, j)$ is the probability that items i and j are rated together. *PMI* ranges from −1 to 1, where −1 indicates that two items are never rated together, while 1 indicates that two items are always rated together. Based on point-wise mutual information, unexpectedness metrics have two variations:

$$unexp_{kam}^{co-occ1}(i,u) = \max_{j \in I_u} PMI(i,j) \tag{11}$$

$$unexp_{kam}^{co-occ2}(i,u) = \frac{1}{|I_u|}\sum_{j \in I_u} PMI(i,j) \tag{12}$$

Table 5
Primitive systems used in serendipity metrics.

Metric	Article	Primitive system
ser_{ad}	Adamopoulos and Tuzhilin [10,28]	Used two systems: one based on users' profiles and another based on the highest rating among the most popular items.
	de Gemmis et al. [8]	Used two systems: popularity and average rating.
ser_{ge}	Ge et al. [25]	Suggested a system based on popularity.
	Lu et al. [41]	Used a system based on popularity.
	Sridharan [37]	Used a systems based on popular and highly rated items.
ser_{mur}	Murakami et al. [20]	Used models based on users' profiles.
	Maksai et al. [39]	Used a systems based on popular items.

The content-based unexpectedness metric is based on item attributes and has two variations. Both variations correspond to novelty metrics proposed by Vargas and Castells [27] (Eq. ([4,5])), where similarity is represented by Jaccard similarity.

5.2. Full metrics

Murakami et al. proposed a serendipity metric based on a primitive recommender system [20]:

$$ser_{mur}(u) = \sum_{i \in R_u} max(Pr_u(i) - Prim_u(i), 0) \cdot rel_u(i), \tag{13}$$

where $Pr_u(i)$ and $Prim_u(i)$ is the confidence of recommending item i to user u by the examined and the primitive models (RSs), respectively. The list of recommendations is represented by R_u, while the relevance of an item corresponds to $rel_u(i)$ (Table 3). A primitive RS is chosen arbitrarily and required to provide suggestions with low serendipity. Furthermore, Murakami et al. proposed the rank sensitive metric [20]:

$$ser\_r_{mur}(u) = \frac{1}{|R_u|} \sum_{j=1}^{|R_u|} max(Pr_u(i_j) - Prim_u(i_j), 0) \cdot rel_u(i_j) \cdot \frac{count_u(j)}{j}, \tag{14}$$

where $count_u(j)$ is the number of relevant items for user u with a rank lower or equal to j. Later Ge et al. modified proposed metric (13) [25]:

$$ser_{ge}(u) = \frac{1}{|R_u|} \sum_{i \in (R_u \setminus PM)} rel_u(i), \tag{15}$$

where PM is a set of items generated by a primitive recommender model. Adamopoulos and Tuzhilin then modified metric (15) by using an additional set of items [28]:

$$ser_{ad}(u) = \frac{1}{|R_u|} \sum_{i \in (R_u \setminus (E_u \cup PM))} rel_u(i), \tag{16}$$

where E_u is a set of items that matches the interests of user u.

Table 5 demonstrates the classification of papers in terms of primitive RS. The serendipity metric ser_{ad} includes popularity, similarity to a user profile and relevance. Other metrics ser_{ge}, ser_{mur} and ser_r_{mur} would also capture these components if the primitive model simultaneously captured user interests and the popularity of items.

De Pessemier et al. generalized the serendipity metric originally proposed in [7]. The metric is calculated as follows [58]:

$$ser_{pe}(i, u) = \frac{1 + n_{I_u,max} - n_{I_u,f(i)}}{1 + n_{I_u,max}} \cdot rel_u(i), \tag{17}$$

$$n_{I_u,max} = \max_{f \in F}(n_{I_u,f}), \tag{18}$$

where $n_{I_u,f}$ is the number of items with attribute f from user profile I_u, while $n_{I_u,max}$ is the maximum number of items with the same attribute from the user profile. The attribute of item i is represented by $f(i)$.

5.3. Analysis of the evaluation metrics

Both component metrics and full metrics have their advantages and disadvantages [3]. Component metrics could be useful in measuring different aspects, such as popularity or similarity to a user profile. However, these metrics could be mistaken. For example, an algorithm suggests items 1, 2, 3, and 4, where items 1 and 2 are irrelevant, novel and unexpected, while items 3 and 4 are relevant but familiar and expected. The algorithm does not suggest any serendipitous items, but it might have high novelty, unexpectedness and accuracy.

Full metrics measure serendipity as a whole, but they also have disadvantages. Most full metrics are sensitive to primitive recommender systems [3,21,58]. By changing this parameter, one might obtain completely different and even contradictory results. Full metrics also disregard multiple levels of serendipity. As the relevance component of serendipity can be assessed using multiple judgments [59], one item might be more serendipitous than another. Full metrics that are not based on a primitive recommender system often disregard some serendipity subcomponents. For example, $ser_{pe}(i, u)$ measures relevance and dissimilarity of items to a user profile and ignores unpopularity.

To answer *RQ3*, in this section we classified and reviewed proposed metrics to assess serendipity in RSs. We did not include serendipity metrics designed to measure serendipity in particular domains such as friend recommendation [12], as they might not be applicable in other domains.

6. Future directions

In this section we are going to answer *RQ4* by providing ways to improve serendipity in recommender systems (RSs) that have not yet been widely adopted but that seem promising. Based on the reviewed literature, we are going to indicate four directions: (1) popularity and similarity in RSs, (2) context-aware RSs, (3) cross-domain RSs and (4) group RSs.

6.1. Popularity and similarity in RSs

According to the reviewed literature, most novelty-oriented and serendipity-oriented recommendation algorithms as well as evaluation metrics consider either popularity of items [8,18,41,49] or their similarity to a user profile [10,46,48]. One of the promising directions is the development of algorithms and evaluation metrics that capture both subcomponents of serendipity.

As we mentioned in Section 2.2, popularity is an important subcomponent of serendipity, as a user is likely to be familiar with globally popular items. A user might become aware of these items through different channels of information, such as radio, TV or friends [11].

Similarity of recommended items to a user profile is also important, since a user tends to look for items similar to what she/he likes, with the aim of finding more items she/he would enjoy. A

user therefore might be aware of items similar to his/her profile [27].

It is challenging to suggest items that are relevant, unpopular and dissimilar to a user profile for two reasons. First, as most unpopular items are of low quality, a user is likely to consider them irrelevant [6]. Second, as a user tends to enjoy items similar to what she/he already likes, by increasing the number of items dissimilar to the user profile, an RS is likely to suggest items the user simply dislikes [42].

One of the ways to suggest items that are relevant, unpopular and dissimilar to a user profile is to combine each of these requirements into an optimization objective. For example, Zheng et al. suggested an optimization function of an SVD algorithm that combines relevance, popularity and similarity to a user profile [13]. Another way is to split a recommendation algorithm into different stages and handle one requirement per stage. For example, Full Auralist consists of three algorithms, each of which is responsible for one of the objectives [19].

6.2. Context-aware RSs

In this paper, we assume that user tastes do not change with time; neither do they depend on the environment. However, many factors, such as weather, mood or location, can influence user preference for items [47,60,61]. For example, a user might prefer listening to soft music in the evening and energetic music in the morning. Suggesting songs according to the time of day would result in the improvement of user satisfaction.

Many recommender systems aim at predicting ratings users would give to items based on available ratings [47,61]. In contrast, context-aware RSs consider additional information, such as time, mood or location that might be relevant for generating recommendations [61,62].

The definition of serendipity should include the context, as each component of the concept is context-dependent [3]. An item that was serendipitous yesterday might not be serendipitous tomorrow. Serendipity thus could consist of relevance, novelty and unexpectedness in a given context.

Leveraging the context can be beneficial for suggesting serendipitous items, as context-aware RSs aim at suggesting items relevant to a user in a particular context. By using context, an RS can recommend items different from what the user usually consumes but also relevant, since the suggestions would fit a particular situation. For example, if a user who does not usually listen to rock and roll drives to Las Vegas, then the novel for the user song “Viva Las Vegas” by Elvis Presley would likely be considered serendipitous.

6.3. Cross-domain RSs

Most RSs generate recommendations based on a single domain. In this paper, the term domain refers to “a set of items that share certain characteristics that are exploited by a particular recommender system” [63]. Cross-domain RSs take advantage of data from multiple domains [64]. For example, a cross-domain RS might leverage information from a book domain to improve recommendation performance in a movie domain [65].

Several types of cross-domain RSs can improve the serendipity of recommendations:

- User modeling. By using information from multiple domains, this kind of system can infer items with which users are already familiar. Leveraging additional information may also help discover additional interests of users, which might result in serendipitous suggestions.
- Relevance. Use of additional domains may help to recommend more items relevant for a user, improving serendipity, as relevance is one of serendipity's components.
- Context. Additional domains may contain information regarding context [64,66]. By using this kind of data, one may apply context-aware recommendation algorithms and improve serendipity, as indicated in Section 6.2.

Cross-domain RSs can suggest combinations of items from different domains. For example, a user could receive a recommendation of a venue to visit and a song to listen to in that particular venue. Combinations of items affect the definition of the concept, as it is not clear whether a combination or each item in the combination has to be novel, unexpected and relevant [3].

6.4. Group RSs

Most recommender systems suggest items to an individual user [67]. However, in some cases it is important to generate recommendations for a group of users. For example, a few friends might need to choose a movie to watch together.

Suggesting items serendipitous to a group of users is more challenging than suggesting these kinds of items to an individual, as items serendipitous to the group must be novel, unexpected and relevant to each individual in the group. Items serendipitous to the group could be represented by an intersection of sets containing items serendipitous to each user in the group. The cardinality of the intersection is less than or equal to the cardinality of each user’s set of serendipitous items. The serendipity of a group recommendation could be assessed by measuring serendipity for each user in the group.

The presence of different individuals in the group could be considered as the context of the recommendation. An item not serendipitous to a particular user might become serendipitous when it is consumed around certain individuals. To the best of our knowledge, efforts on recommendation of serendipitous items to a group of users are very limited.

7. Conclusion

In this paper, we investigated serendipity in recommender systems (RSs) and answered our research questions.

RQ1. What is serendipity in recommender systems? What makes certain items serendipitous for particular users?

In Section 2, we defined serendipity as a property that reflects how good an RS is at suggesting serendipitous items that are relevant, novel and unexpected. Novelty and unexpectedness require serendipitous items to be relatively unpopular and significantly different from a user profile.

RQ2. What are the state-of-the-art recommendation approaches that suggest serendipitous items?

We overviewed serendipity-oriented algorithms and provided two classifications of them in Section 3. The first classification is based on the architecture of the algorithms, while the second classifies algorithms depending on the phase of recommendation that is responsible for serendipity.

RQ3. How can we assess serendipity in recommender systems?

We overviewed evaluation strategies to assess serendipity in RSs in Section 4. We discussed qualitative and quantitative methods used in experimental studies related to serendipity.

We also presented two categories of serendipity metrics in Section 5: component metrics and full metrics. Component metrics measure serendipity components, while full metrics measure serendipity as a whole.

RQ4. What are the future directions of serendipity in RSs?

In Section 6, we indicated four future directions of serendipity-oriented algorithms. The first direction indicates that serendipity-oriented algorithms and evaluation metrics should take into account both item popularity and similarity to a user profile. The second direction involves context-aware recommender systems. As context-aware RSs can pick items based on the context rather than user tastes, this kind of system might suggest more serendipitous items. The third direction includes cross-domain RS. Having information from additional domains, an RS can infer which items are familiar to a user, suggest items relevant to the user and select items that fit the context. The fourth direction is dedicated to group recommendations. It is difficult to suggest items serendipitous to each user in a group of individuals, as the tastes of each user must be considered.

We hope not only that this paper will make the reader aware of serendipity and related concepts in RSs, but also that it will help one conduct one's own experiments by picking suitable algorithms, evaluation strategies and metrics. We also hope that, as a result, the presented overview will contribute to the development and improvement of recommendation algorithms focused on user satisfaction rather than on accuracy metrics.

Acknowledgement

The research at University of Jyväskylä was performed in the MineSocMed project, partially supported by the Academy of Finland, grant #268078. The research was also partially supported by Natural Science Foundation of China #71402083.

References

[1] F. Ricci, L. Rokach, B. Shapira, Recommender Systems Handbook, Springer US, pp. 1–35.
[2] M.D. Ekstrand, J.T. Riedl, J.A. Konstan, Collaborative filtering recommender systems, Found. Trends Hum. Comput. Int. 4 (2) (2011) 81–173.
[3] D. Kotkov, J. Veijalainen, S. Wang, Challenges of serendipity in recommender systems, in: Proceedings of the 12th International Conference on Web Information Systems and Technologies., vol. 2, SCITEPRESS, 2016, pp. 251–256.
[4] J.B. Schafer, J. Konstan, J. Riedl, Recommender systems in e-commerce, in: Proceedings of the 1st ACM Conference on Electronic Commerce, ACM, New York, NY, USA, 1999, pp. 158–166.
[5] J. Bobadilla, F. Ortega, A. Hernando, A. Gutierrez, Recommender systems survey, Knowl. Based Syst. 46 (2013) 109–132.
[6] Ò. Celma Herrada, Music Recommendation and Discovery in the Long Tail, Ph.D. thesis, Universitat Pompeu Fabra, 2009.
[7] G. Shani, A. Gunawardana, Recommender Systems Handbook, vol. 51, Springer US, pp. 695–717.
[8] M. de Gemmis, P. Lops, G. Semeraro, C. Musto, An investigation on the serendipity problem in recommender systems, Inf. Process. Manage. 51 (2015) 695–717.
[9] L. Iaquinta, M. de Gemmis, P. Lops, G. Semeraro, M. Filannino, P. Molino, Introducing serendipity in a content-based recommender system, in: Proceedings of the 8th International Conference on Hybrid Intelligent Systems, 2008, pp. 168–173.
[10] P. Adamopoulos, A. Tuzhilin, On unexpectedness in recommender systems: or how to better expect the unexpected, ACM Trans. Intell. Syst. Technol. 5 (4) (2014) 1–32.
[11] E. Tacchini, Serendipitous Mentorship in Music Recommender Systems, Ph.D. thesis, Università degli Studi di Milano, 2012.
[12] M. Manca, L. Boratto, S. Carta, Behavioral data mining to produce novel and serendipitous friend recommendations in a social bookmarking system, Inf. Syst. Front. (2015) 1–15.
[13] Q. Zheng, C.-K. Chan, H.H. Ip, An unexpectedness-augmented utility model for making serendipitous recommendation, in: Advances in Data Mining: Applications and Theoretical Aspects, vol. 9165, Springer International Publishing, 2015, pp. 216–230.
[14] K. Onuma, H. Tong, C. Faloutsos, Tangent: a novel, 'surprise me', recommendation algorithm, in: Proceedings of the 15th ACM International Conference on Knowledge Discovery and Data Mining, ACM, New York, NY, USA, 2009, pp. 657–666.
[15] Q. Meng, K. Hatano, Visualizing basic words chosen by latent dirichlet allocation for serendipitous recommendation, in: Proceedings of the 3rd International Conference on Advanced Applied Informatics, 2014, pp. 819–824.
[16] A. Foster, N. Ford, Serendipity and information seeking: an empirical study, J. Doc. 59 (3) (2003) 321–340.
[17] P. André, M.C. Schraefel, J. Teevan, S.T. Dumais, Discovery is never by chance: designing for (un)serendipity, in: Proceedings of the Seventh ACM Conference on Creativity and Cognition, ACM, New York, NY, USA, 2009, pp. 305–314.
[18] A. Said, B. Fields, B.J. Jain, S. Albayrak, User-centric evaluation of a k-furthest neighbor collaborative filtering recommender algorithm, in: Proceedings of the 2013 Conference on Computer Supported Cooperative Work, ACM, New York, NY, USA, 2013, pp. 1399–1408.
[19] Y.C. Zhang, D.O. Séaghdha, D. Quercia, T. Jambor, Auralist: introducing serendipity into music recommendation, in: Proceedings of the 5th ACM International Conference on Web Search and Data Mining, ACM, New York, NY, USA, 2012, pp. 13–22.
[20] T. Murakami, K. Mori, R. Orihara, Metrics for evaluating the serendipity of recommendation lists, in: New Frontiers in Artificial Intelligence, vol. 4914, Springer Berlin Heidelberg, Berlin, Heidelberg, 2008, pp. 40–46.
[21] M. Kaminskas, D. Bridge, Measuring surprise in recommender systems, in: Proceedings of the Workshop on Recommender Systems Evaluation: Dimensions and Design (Workshop Programme of the 8th ACM Conference on Recommender Systems), 2014.
[22] Y. Levy, T.J. Ellis, A systems approach to conduct an effective literature review in support of information systems research, Inf. Sci. 9 (2006) 181–212.
[23] K. Kapoor, V. Kumar, L. Terveen, J.A. Konstan, P. Schrater, "I like to explore sometimes": adapting to dynamic user novelty preferences, in: Proceedings of the 9th ACM Conference on Recommender Systems, ACM, New York, NY, USA, 2015, pp. 19–26.
[24] A.I. Schein, A. Popescul, L.H. Ungar, D.M. Pennock, Methods and metrics for cold-start recommendations, in: Proceedings of the 25th Annual International ACM Conference on Research and Development in Information Retrieval, ACM, New York, NY, USA, 2002, pp. 253–260.
[25] M. Ge, C. Delgado-Battenfeld, D. Jannach, Beyond accuracy: evaluating recommender systems by coverage and serendipity, in: Proceedings of the 4th ACM Conference on Recommender Systems, ACM, New York, NY, USA, 2010, pp. 257–260.
[26] L. Iaquinta, G. Semeraro, M. de Gemmis, P. Lops, P. Molino, Can a recommender system induce serendipitous encounters?, IN-TECH, 2010.
[27] S. Vargas, P. Castells, Rank and relevance in novelty and diversity metrics for recommender systems, in: Proceedings of the 5th ACM Conference on Recommender Systems, New York, NY, USA, 2011, pp. 109–116.
[28] P. Adamopoulos, A. Tuzhilin, On unexpectedness in recommender systems: or how to expect the unexpected, in: Workshop on Novelty and Diversity in Recommender Systems, at the 5th ACM International Conference on Recommender Systems, 2011, pp. 11–18.
[29] S.M. McNee, J. Riedl, J.A. Konstan, Being accurate is not enough: how accuracy metrics have hurt recommender systems, in: Proceedings of the Conference on Human Factors in Computing Systems, ACM, New York, NY, USA, 2006, pp. 1097–1101.
[30] Most Untranslatable Word, (http://www.todaytranslations.com/blog/most-untranslatable-word/). [Online; accessed 20-November-2015].
[31] The Three Princes of Serendip, (http://livingheritage.org/three_princes.htm). [Online; accessed 20-November-2015].
[32] T.G. Remer, Serendipity and the Three Princes: From the Peregrinaggio of 1557, University of Oklahoma Press, p. 20.
[33] Serendipity - Definition of Serendipity by the Free Dictionary, (http://www.thefreedictionary.com/serendipity). [Online; accessed 20-November-2015].
[34] P. Van Andel, Anatomy of the unsought finding. serendipity: origin, history, domains, traditions, appearances, patterns and programmability, Brit. J. Philos. Sci. 45 (1994) 631–648.
[35] P. Resnick, H.R. Varian, Recommender systems, Commun. ACM 40 (1997) 56–58.
[36] J. Corneli, A. Pease, S. Colton, A. Jordanous, C. Guckelsberger, Modelling serendipity in a computational context, Comput. Res. Repos. abs/1411.0440 (2014).
[37] S. Sridharan, Introducing Serendipity in Recommender Systems Through Collaborative Methods, Ph.D. thesis, University of Rhode Island, 2014.
[38] Y.-S. Chiu, K.-H. Lin, J.-S. Chen, A social network-based serendipity recommender system, in: Proceedings of the International Symposium on Intelligent Signal Processing and Communications Systems, 2011, pp. 1–5.
[39] A. Maksai, F. Garcin, B. Faltings, Predicting online performance of news recommender systems through richer evaluation metrics, in: Proceedings of the 9th ACM Conference on Recommender Systems, ACM, New York, NY, USA, 2015, pp. 179–186.
[40] P. Bieganski, J.A. Konstan, J.T. Riedl, System, method and article of manufacture for making serendipity-weighted recommendations to a user, 2001, Patent US6334127 B1.
[41] Q. Lu, T. Chen, W. Zhang, D. Yang, Y. Yu, Serendipitous personalized ranking for top-n recommendation, in: Proceedings of the The IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technology, volume 1
[42] L.W. James, Nudging Music Serendipity, Ph.D. thesis, University of Cambridge, 2013.
[43] X. Su, T.M. Khoshgoftaar, A survey of collaborative filtering techniques, Adv. Artif. Intell. 2009 (2009) 1–20.
[44] A. Tejeda-Lorente, C. Porcel, J. Bernabé-Moreno, E. Herrera-Viedma, Refore: a recommender system for researchers based on bibliometrics, Appl. Soft Comput. 30 (2015) 778–791.
[45] C. Martinez-Cruz, C. Porcel, J. Bernabe-Moreno, E. Herrera-Viedma, A model to represent users trust in recommender systems using ontologies and fuzzy linguistic modeling, Inf. Sci. 311 (2015) 102–118.

[46] T. Akiyama, K. Obara, M. Tanizaki, Proposal and evaluation of serendipitous recommendation method using general unexpectedness, in: Proceedings of the Workshop on the Practical Use of Recommender Systems, Algorithms and Technologies at 4th ACM Conference on Recommender Systems, Barcelona, Spain, 2010, pp. 3–10.
[47] G. Adomavicius, A. Tuzhilin, Recommender Systems Handbook, Springer US, Boston, MA, pp. 217–253.
[48] M. Nakatsuji, Y. Fujiwara, A. Tanaka, T. Uchiyama, K. Fujimura, T. Ishida, Classical music for rock fans? Novel recommendations for expanding user interests, in: Proceedings of the 19th ACM International Conference on Information and Knowledge Management, ACM, New York, NY, USA, 2010, pp. 949–958.
[49] N. Kawamae, Serendipitous recommendations via innovators, in: Proceedings of the 33rd International ACM Conference on Research and Development in Information Retrieval, ACM, New York, NY, USA, 2010, pp. 218–225.
[50] H. Tong, C. Faloutsos, J.-Y. Pan, Random walk with restart: fast solutions and applications, Knowl. Inf. Syst. 14 (3) (2008) 327–346.
[51] N. Kawamae, H. Sakano, T. Yamada, Personalized recommendation based on the personal innovator degree, in: Proceedings of the 3rd ACM Conference on Recommender Systems, ACM, New York, NY, USA, 2009, pp. 329–332.
[52] P. Cremonesi, Y. Koren, R. Turrin, Performance of recommender algorithms on top-n recommendation tasks, in: Proceedings of the 4th ACM Conference on Recommender Systems, ACM, New York, NY, USA, 2010, pp. 39–46.
[53] J.L. Herlocker, J.A. Konstan, A. Borchers, J. Riedl, An algorithmic framework for performing collaborative filtering, in: Proceedings of the 22Nd Annual International Conference on Research and Development in Information Retrieval, ACM, New York, NY, USA, 1999, pp. 230–237.
[54] M. Taramigkou, E. Bothos, K. Christidis, D. Apostolou, G. Mentzas, Escape the bubble: guided exploration of music preferences for serendipity and novelty, in: Proceedings of the 7th ACM Conference on Recommender Systems, ACM, New York, NY, USA, 2013, pp. 335–338.
[55] I. Guy, R. Levin, T. Daniel, E. Bolshinsky, Islands in the stream: a study of item recommendation within an enterprise social stream, in: Proceedings of the 38th International ACM Conference on Research and Development in Information Retrieval, ACM, New York, NY, USA, 2015, pp. 665–674.
[56] M.N. Jelassi, S.B. Yahia, E.M. Nguifo, Towards more targeted recommendations in folksonomies, Soc. Netw. Anal. Min. 5 (2015) 1–18.
[57] P. Lops, M. de Gemmis, G. Semeraro, Recommender Systems Handbook, Springer US, Boston, MA, pp. 73–105.
[58] T. Pessemier, S. Dooms, L. Martens, Comparison of group recommendation algorithms, Multimed. Tools Appl. 72 (2013) 2497–2541.
[59] K. Järvelin, J. Kekäläinen, Cumulated gain-based evaluation of ir techniques, ACM Trans. Inf. Syst. 20 (2002) 422–446.
[60] K. Verbert, N. Manouselis, X. Ochoa, M. Wolpers, H. Drachsler, I. Bosnic, E. Duval, Context-aware recommender systems for learning: a survey and future challenges, IEEE Trans. Learn. Technol. 5 (4) (2012) 318–335.
[61] F.V. Steeg, Context-Aware Recommender Systems, Utrecht University, Master's Thesis, 2015.
[62] L. Baltrunas, Context-aware collaborative filtering recommender systems, Ph.D. thesis, 2011.
[63] I. Fernández-Tobías, I. Cantador, M. Kaminskas, F. Ricci, Cross-domain recommender systems: a survey of the state of the art, in: Proceedings of the 2nd Spanish Conference on Information Retrieval, 2012, pp. 187–198.
[64] I. Cantador, P. Cremonesi, Tutorial on cross-domain recommender systems, in: Proceedings of the 8th ACM Conference on Recommender Systems, New York, NY, USA, 2014, pp. 401–402.
[65] P. Winoto, T. Tang, If you like the devil wears prada the book, will you also enjoy the devil wears prada the movie? A study of cross-domain recommendations, New Generat. Comput. 26 (3) (2008) 209–225.
[66] I. Cantador, P. Cremonesi, Cross-domain Recommender Systems, 2014, (http://recsys.acm.org/wp-content/uploads/2014/10/recsys2014-tutorial-cross_domain.pdf) [Online; accessed 20-November-2015].
[67] J. Masthoff, Recommender Systems Handbook, Springer US, Boston, MA, pp. 677–702.