Wisdom of Crowds and Fine-Grained Learning for Serendipity Recommendations

Zhe Fu
College of Computing and Informatics
University of North Carolina at Charlotte
Charlotte, North Carolina, USA
zfu2@uncc.edu

Xi Niu*
College of Computing and Informatics
University of North Carolina at Charlotte
Charlotte, North Carolina, USA
xniu2@uncc.edu

Li Yu
School of Information
Renmin University of China
Beijing, China
buaayuli@ruc.edu.cn

ABSTRACT

Serendipity is a notion that means an unexpected but valuable discovery. Due to its elusive and subjective nature, serendipity is difficult to study even with today's advances in machine learning and deep learning techniques. Both ground truth data collecting and model developing are the open research questions. This paper addresses both the data and the model challenges for identifying serendipity in recommender systems. For the ground truth data collecting, it proposes a new and scalable approach by using both user generated reviews and a crowd sourcing method. The result is a large-scale ground truth data on serendipity. For model developing, it designed a self-enhanced module to learn the fine-grained facets of serendipity in order to mitigate the inherent data sparsity problem in any serendipity ground truth dataset. The self-enhanced module is general enough to be applied with many base deep learning models for serendipity. A series of experiments have been conducted. As the result, a base deep learning model trained on our collected ground truth data, as well as with the help of the self-enhanced module, outperforms the state-of-the-art baseline models in predicting serendipity.

CCS CONCEPTS

• **Information systems** → **Recommender systems**.

KEYWORDS

Serendipity; Recommender systems; Crowd-sourcing; Deep learning models; Transformers

ACM Reference Format:
Zhe Fu, Xi Niu, and Li Yu. 2023. Wisdom of Crowds and Fine-Grained Learning for Serendipity Recommendations. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '23), July 23–27, 2023, Taipei, Taiwan.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3539618.3591787

*Corresponding author

SIGIR '23, July 23–27, 2023, Taipei, Taiwan.

ACM ISBN 978-1-4503-9408-6/23/07…$15.00
https://doi.org/10.1145/3539618.3591787

1 INTRODUCTION

Recommender systems are designed to alleviate information overload by recommending items based on user preferences. With the wide applications of deep learning models, tremendous success has been achieved in recommender systems to capture complex user-item relationships from data. However, these deep learning-based recommendation models, like any machine learning models, tend to overly focus on recommendation accuracy. On the other side, many people have expressed their hope that recommender systems could play a role in facilitating incidental exposure to serendipitous information, whereby individuals "stumble upon" unexpected but valuable items that they did not actively seek. Therefore, in the recent decade, the notion of serendipity has been advocated by many recommender researchers. The word serendipity was created in 1754 to describe unexpected but valuable discoveries [17]. In the early 2000s, serendipity was first introduced to the context of recommender systems [4] in order to improve users' engagement and satisfaction [7]. Although currently there is no consensus on the definition of serendipity in the context of recommender systems, **most researchers interpret and operationalize this notion with two components: unexpectedness and relevance [5, 9, 23].**

Today deep learning models have gained huge success in modeling user-item relevance relationships in recommender systems. However, modeling user-item serendipity relationship between a user and an item is challenging due to the elusive nature of serendipity. The element of unexpectedness in serendipity means surprise and accident, which are susceptible to modeling and prediction. In addition, there are two data related challenges. First, the ground truth data on serendipity is difficult to collect. Serendipity's occurrence is not always immediate. A period of "incubation" is sometimes necessary before serendipity is recognized [13]. Therefore, the common ground truth data collection methods, such as surveys, online tracking, or direct observations, do not work well. The second challenge is that serendipity does not occur frequently in a natural environment. Serendipity by nature will have a data sparsity problem in the collected ground truth dataset.

In this paper, we addressed the challenges for both ground truth data collecting and model developing for identifying serendipity in recommender systems. For collecting the ground truth data, we proposed a new method called *URCW (User Reviews plus Crowd Wisdom).* The method first collects user generated reviews that describe a naturally occurring serendipity encounter in their daily life in a reflective manner, thus avoiding the need of instant recall from users about their serendipity experiences. The method then

consults crowd workers to confirm whether a piece of review describes the user (the review writer)'s serendipity experience. The result is a large ground truth data, called *SerenLens*, as compared to the well-known dataset for recommendation models, *MovieLens* [3]. For developing models, we designed a self-enhanced module called *SerenEnhance*. The module is to learn the fine-grained facets of serendipity to supplement a base deep learning model that is trained on *SerenLens*. There is an inherent data sparsity problem in *SerenLens* due to the fact that serendipity by nature happens infrequently in real life. The self-enhanced module is to mitigate the data sparsity problem by leveraging the auto-generated labels from data itself, rather than a human labeling process. The self-enhanced module is sufficiently general to be used together with any deep learning model as a base model. Extensive experiment results show that a base deep learning model trained on *SerenLens* and strengthened by the *SerenEnhance* module, outperforms the state-of-the-art baseline models in predicting serendipity with relatively lower sacrifice to relevance.

The main contributions of this paper are summarized as two-fold:

- A new ground truth data collection approach called *URCW*, as well as the resulting large ground truth dataset on serendipity called *SerenLens*.
- A novel self-enhanced module, *SerenEnhance*, to strengthen a base deep learning model for predicting serendipity.

2 RELATED WORK

This study draws on two research lines: serendipity research and deep learning techniques.

2.1 The Concept of Serendipity and Its Distinction with Diversity and Novelty

First coined by Harold Walpole in 1754, the word "serendipity" is used to describe the process of making discoveries by accident, but it received little attention until the mid-1900s when it was used as a descriptor of accidental or unplanned discovery in the scientific context [14]. Although there is some disagreement as to the precise nature of serendipity, all accounts agree that the following two aspects are central: **an unexpected chance and a relevant discovery**. These two aspects have informed us on the ground truth data collection and deep learning model loss function design. It is worth distinguishing between serendipity, diversity, and novelty because they share some common characteristics. We believe diversity increases the chance of serendipity, but not every diversified piece is serendipitous: only those unexpected and relevant items are serendipity. As to novelty, it means being new and unknown, not necessarily unexpecting or surprising. In contrast, serendipity suggests how strongly an item violates an expectation.

2.2 Recent Deep Learning Models in Serendipity Recommender Systems

Recently deep learning advances have been changing recommender systems research dramatically and bringing more opportunities to improve the relevance-oriented performance. Since 2018, a few information retrieval (IR) researchers have attempted to build deep learning models for serendipity recommendations. We believe Pandey et al. [16] is the first effort to build a deep learning model to predict serendipity. The model, called SerRec (Serendipity Recommender), used a pre-training and fine-tuning mechanism to firstly train a deep neural network for relevance scores using a large MovieLens dataset and then fine-tune the model for serendipity scores using the smaller dataset *Serendipity 2018* collected by GroupLens Research. Their pre-train and fine-tune approach mitigated the issue of a small serendipity dataset and achieved reasonable NDCG (normalized discounted cumulative gain) scores in predicting serendipity. However, the dataset *Serendipity 2018* was collected using a small-scale survey with only 481 participants. In addition, the controlled environment that relied on participants' recall was not ideal for collecting serendipity experiences.

With the lack of large-scale ground truth data for serendipity, other researchers are working to define serendipity in ways to leverage various existing relevance-oriented datasets. For example, Li et al. [10] defined serendipity as content difference and genre accuracy for a movie recommender, and then developed an algorithm called HAES (Hybrid Approach for movie recommendations with Elastic Serendipity), to achieve the two aspects. In their follow-up study, Li et al. [11] adjusted the definition of serendipity as an item with a direction pointing from the short-term demand to the long-term preference as well as a suitable distance to the short-term demand. They developed a deep learning algorithm called DESR (Directional and Explainable Serendipity Recommendation), to achieve the computational definition. Also, Xu et al. [22] defined serendipity as high satisfaction and low initial interest, and achieved both by using a neural network, called NSR (Neural Serendipity Recommendation), which balances between accuracy and novelty. Li et al. [9] presented a novel PURS (Personalized Unexpected Recommender System) model, the first deep learning model to incorporate unexpectedness into the recommendations. They modeled the item's unexpectedness level as the item's average distance to each cluster center of a user's interests. However, the study pre-calculated the level of unexpectedness without putting the unexpectedness module into the model training process. In the most recent study, Zhang et al. [23] "engineered" the ground truth data on serendipity by defining a way to calculate the level of serendipity. The core part of the "engineering" process is how to calculate an item's level of unexpectedness. They calculated it as the sum of item difference and the category difference. With the engineered ground truth data available, they then trained and tested a Transformer-based deep learning model, called SNPR (Serendipity-oriented Next POI Recommendation model). Like other existing research, it did not have direct serendipity-oriented ground truth data, and therefore relied completely on a self-defined calculation approach to convert a relevance-oriented data to serendipity-oriented data.

In summary, these deep learning efforts mentioned above collectively demonstrate the effectiveness of deep learning in representing users' preferences. However, the serendipity definitions varied and those self-defined evaluation metrics are subject to debate. Most of these efforts are designed to leverage existing data and avoid the direct ground truth on serendipity, making both the models and the results not comparable and generalizable.

Table 1: Example book reviews of true and false serendipity experiences

Keyword	True Serendipity Experience	False Serendipity Experience
stumble on/stumble upon	"*I **stumbled on** this book by accident, it was on an iPad I had borrowed and was checking out the features... but it grabbed my attention from the first page and I could not put it down. I recommended it to friends who loved it just as much. It is not my usual genre, but I loved it. It had everything, tragedy, suspense, romance, and an interesting backdrop for a story. The movie was 'awwright'".*	"*Set in both present time and the 1920's, it tells the tale of young Jacob Jankowski's circus days: how he **stumbled upon** the spectacle, how he made himself useful, and how he found true love.*"
by chance	"*I am not at all a tennis fan and have never followed the sport so I have no idea why I decided to read this book but **by chance** I downloaded in onto my kindle and absolutely couldn't put it down after reading the first page.*"	"*The book starts out with a murderer killing three members of a family and, purely **by chance**, missing the opportunity to kill the fourth member, a toddler who has wandered off into the neighborhood graveyard.*"
serendipity/serendipitous	"*Then **serendipitously**, while browsing in a discount bookstore, I found a British copy of Rachel's Holiday just sitting there waiting to be discovered by an American reader.*"	"*I bought this book after reading the charming '**Serendipity**'s Secret' which also uses the fable format to good effect. I thought 'The Monk...' had some great ideas and I enjoyed reading it. Just like **Serendipity**'s Secret, I loved the idea of a 'coach' and a 'coachee' as the central characters and the useful summaries at the end of each chapter were effective.*"

3 *URCW*: OUR PROPOSED METHOD FOR COLLECTING SERENDIPITY GROUND TRUTH

As mentioned in Introduction, serendipity does not occur frequently or immediately in a natural environment. A period of "incubation" is sometimes necessary before serendipity is recognized [13]. Therefore a survey instrument or a lab-based user study may not be ideal for collecting serendipity ground truth data because of their instant nature and controlled environment, making the "incubation" period impossible. Also importantly, the data collected by a survey or a lab-based user study usually cannot reach the scalability that is needed by a machine learning model. Therefore, we propose a new serendipity data collection method, named *URCW (User Reviews plus Crowd Wisdom)*, which is a two-stage process.

In the first stage of *URCW*, we conducted a variety of keyword searches to retrieve the contents from a large corpus of user reviews. We expected user reviews to provide valuable self-reports of naturally occurring serendipity, and we tapped into this source of data that is highly underused in the serendipity research. Our prior examination of user reviews from a book review corpus indicated that some keywords were good candidates. These keywords occurred reasonably frequently in the corpus, and the reviews containing these keywords reflected a reasonably high fraction of true serendipity experiences. For example, "stumble on" is a good keyword. We also found some keywords were not good candidates because they were not sufficiently frequent in the corpus or the reviews with those keywords always had nothing to do with a serendipity experience. For example, the keyword "discovered $*$ when I $*$ not looking for" ($*$ is a regular expression) did not occur sufficiently frequently. The keyword "serendipity" occurred reasonably frequently, but this keyword was often used in a way that did not have much to do with users' serendipity experiences. Sometimes it was used simply as part of a name, such as the movie *Serendipity* in 2001 or a restaurant name *Serendipity 3* in New York City. Table 1 lists more examples using sample keywords to retrieve true or false serendipity experiences. We manually curated a set of seed keywords, some with regular expressions to retrieve instances of serendipity. We brainstormed to expand the keywords through several iterations. We tested the resulting keywords by assessing the retrieved accounts from user reviews. Please note that we neither aimed for a high precision nor a high recall from these keywords to retrieve true serendipity experiences from the review corpus. We only needed a reasonable amount of reviews with a reasonable "concentration" of serendipity for human crowd workers to further make judgements on, in order to make a ground truth dataset.

In the second stage of *URCW*, we used Amazon Mechanical Turk (MTurk), the well-known crowdsourcing platform, to reach a large number of crowd workers to make a judgement on whether a piece of review is about the user (review writer)'s serendipity experience. We defined serendipity experiences based on literature: 1) having the two elements of unexpectedness and relevance in a discovery experience, and 2) the experience is positive. Each crowd worker was instructed to make a binary decision on whether a piece of review described the review writer's serendipity experience or not. To obtain quality human opinions, we provided workers with succinct instructions and examples on what is and is not a serendipity experience. In addition, we asked each worker two verification questions for quality checks. Specifically, we created a straightforward three-step process for each HIT (Human Intelligence Task) in the MTurk platform, as shown in Figure 1. The HIT started by presenting a brief introduction on what is serendipity experience and what is not, along with both true and false examples. Then, it displayed a list of 5 pieces of reviews, soliciting the worker's yes or no decision on each review. Third, it presented a quality check question about the concept of serendipity to make sure the crowd worker was paying attention and understood the HIT. Each HIT recruited two workers initially. An agreement between the two workers established the final label on that particular piece of

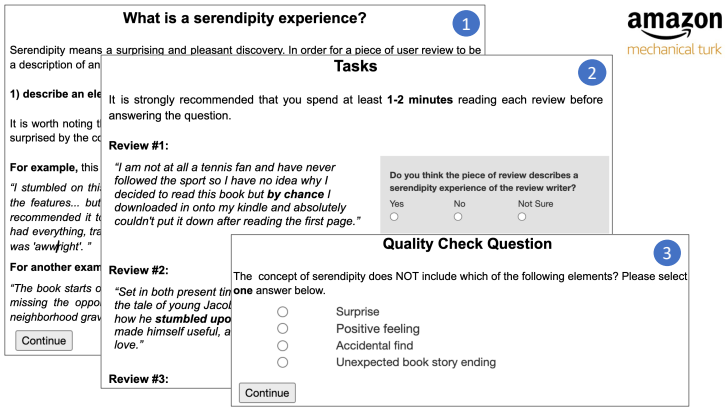

Figure 1: Three steps of an HIT task at Amazon Mechanical Turk (MTurk)

review. In case of any disagreement, a third worker was recruited to serve as a tie breaker.

4 *SERENENHANCE*: OUR PROPOSED SELF-ENHANCED MODULE

After collecting *SerenLens*, we realized that serendipity is sparse in the dataset, even *SerenLens* is large-scale. This is due to the fact that serendipity rarely happens in real life. Therefore we selected a deep learning model as the base and designed a self-enhanced module called *SerenEnhance* to strengthen the base model trained on *SerenLens*. The *SerenEnhance* module is trained on auto-generated labels in order to mitigate the sparsity problem in *SerenLens*.

4.1 Problem Formulation

We believe a serendipity recommendation problem, like any other information retrieval problems, is a matching problem. Let $I = \{i_1, i_2, \ldots, i_{|I|}\}$ represents the set of items, and $T_u = \{i_1^u, i_2^u, \ldots, i_n^u\}$ represents a history of interacted items for the user u. Our goal is to generate a recommendation list L with the length k, which contains the most likely serendipity items for user u. The task can be formulated as learning a matching function for serendipity, and then ranking the items according to the matching score. The matching function is:

$$\hat{y}_{u,i} = f(\Theta, \Phi_u(T_u), \Phi_i(i)) \tag{1}$$

where $\hat{y}_{u,i}$ denotes the matching score: a probability score that the user u with a history T_u feels the item i ($i \in I$) serendipitous. Θ denotes the set of parameters of the matching function f. Φ_u and Φ_i are the functions to represent T_u and i respectively.

Previous research shows that deep learning recommendation models using Transformers [20] as Φ_u to represent users are effective [1, 6, 21]. Transformers are especially useful for serendipity recommendations because serendipity occurrence does not necessarily rely on a user's strict order of the historical sequence, and the recent user interactions may not have the most impacts on serendipity discovery. Serendipity could be a potential, long-time-ago, or dormant need being recognized. Although not relying on the strict temporal order of history, serendipity heavily depends on the user context, meaning having a complex interdependence among potentially all the items in the user history. Therefore, in this paper, we selected a Transformer based model as the base model.

4.2 The Base Model

The architecture of the base model is presented in the blue colored components on the left side of Figure 2. The base model serves two purposes: 1) **serendipity representation**: it encodes each historical item of a user using a Transformer encoder, and then aggregates the representation; and 2) **serendipity learning**: it adopts a learning loss function to guide the training of the serendipity representation toward the collected ground truth of serendipity.

4.2.1 Serendipity Representation. For each item $i \in I$, we obtained a set of reviews $\{r_1^i, r_2^i, \ldots, r_l^i\}$ that are about the item i. This set of reviews can be used for this item i's representation. We first learned the vector representation for each piece of review as $\mathbf{e_j^i}$. Then we represented each item by aggregating all its reviews as $\mathbf{q_i} \in \mathbb{R}^{1\times d}$. Therefore, each user's history can be represented as a sequence of the user's interacted items: $\mathbf{T_u} = \{\mathbf{q_1}, \mathbf{q_2}, \ldots, \mathbf{q_n}\}$.

As mentioned earlier, this study leverages the state-of-the-art Transformer technique [20] to model all the potential dependencies among items in a user's history. For each $\mathbf{q_i}$, we obtained the hidden representation $\mathbf{z_i}$ using a stack of L Transformer layers. Each Transformer layer consists of a multi-head sublayer and a feed-forward network sublayer. We leveraged an average pooling layer to aggregate $\mathbf{z_1}, \mathbf{z_2}, \ldots, \mathbf{z_n}$ and further put through a dense layer to obtain the user serendipity representation $\mathbf{h_{seren}}$:

$$\begin{aligned} \mathbf{z} &= Ave([\mathbf{z_1}, \mathbf{z_2}, \ldots, \mathbf{z_n}]) \\ \mathbf{h_{seren}} &= ReLU(\mathbf{W}\mathbf{z} + \mathbf{b}) \end{aligned} \tag{2}$$

where $Ave(*)$ is the average pooling layer, $\mathbf{W} \in \mathbb{R}^{d\times d}$ is the weight matrix for the dense layer, and $\mathbf{b}$ is the bias vector.

4.2.2 Serendipity Learning. After obtaining the representation of serendipity $\mathbf{h_{seren}}$, the model takes in the vector $\mathbf{h_{seren}}$ and generates the predicted serendipity score as:

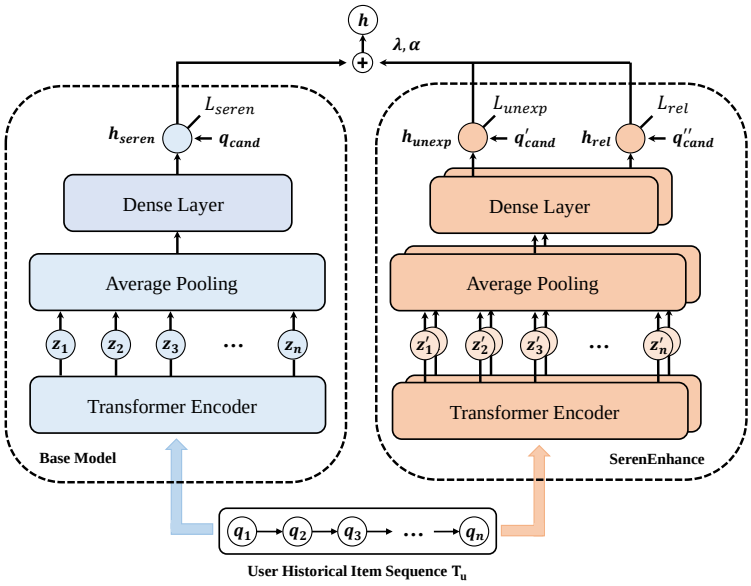

Figure 2: Structure of our proposed deep learning framework: a base model plus *SerenEnhance*

$$\hat{y}_{u,i} = \sigma(\mathbf{q}_{\mathbf{cand}}^{\mathbf{T}} \cdot \mathbf{h}_{\mathbf{seren}}) \tag{3}$$

where the vector $\mathbf{q}_{\mathbf{cand}}^{\mathbf{T}}$ is a candidate item in the item set $\mathbf{I}$ and $\sigma(*)$ denotes the sigmoid activation function. Since our serendipity recommendation task is a personalized ranking task, it is reasonable to assume that the observed serendipity should be ranked higher than the unobserved ones, meaning having a larger $\hat{y}_{u,i}$ than the unobserved cases. To implement this idea, we used the well-established Bayesian Personalized Ranking (BPR) [18] objective function as follows:

$$L_{seren}(\Theta) = \sum_{(u,i,j)\in\mathcal{Y}} -log\,\sigma(\hat{y}_{u,i} - \hat{y}_{u,j}) \tag{4}$$

where Θ is the set of model parameters. $\mathcal{Y}$ denotes the set of training instances: $\mathcal{Y} : \{(u,i,j)|i \in \mathcal{Y}_u^+ \wedge j \in \mathcal{Y}_u^-\}$, where $\mathcal{Y}_u^+$ denotes the set of items that has been regarded as serendipitous by the user u. By minimizing the BPR loss, we tailored the model for correctly predicting the relative orders between items rather than their absolute serendipity probability scores as optimized in pointwise loss. This can be more beneficial for addressing the serendipity recommendation task where serendipity cases are relatively rare.

4.3 The Proposed Self-Enhanced Module: *SerenEnhance*

As mentioned, serendipity is sparse in *SerenLens*, an inherent problem in any serendipity ground truth data due to the fact that serendipity rarely happens in real life. In order to strengthen *SerenLens* and the base deep learning model trained on it, we designed this *SerenEnhance* module, as in the orange-colored components on the right side of Figure 2, to leverage some auto-generated labels to learn the fine-grained facets of serendipity. As mentioned in Introduction, serendipity is a complex concept, most researchers interpret and operationalize the notion with two core facets: unexpectedness and relevance. Therefore, in the *SerenEnhance* module, we designed two learning loss functions as the two auxiliary loss functions. One is for learning unexpectedness and the other is for learning relevance. For these two auxiliary loss functions, we were able to generate the ground truth label either from calculation or observation on the data itself, rather than from a human labeling process. Therefore we call *SerenEnhance* a self-enhanced module.

4.3.1 Unexpectedness Learning and Unexpectedness Ground Truth Generation. We adopted the base model again to obtain an unexpectedness representation $\mathbf{h}_{\mathbf{unexp}}$ from the user history. The predicted unexpectedness score is then derived from this representation:

$$\hat{n}_{u,i} = \sigma(\mathbf{q}_{\mathbf{cand}}^{\prime\mathbf{T}} \cdot \mathbf{h}_{\mathbf{unexp}}) \tag{5}$$

where the vector $\mathbf{q}_{\mathbf{cand}}^{\prime\mathbf{T}}$ is a candidate item in the item set $\mathbf{I}$ for the unexpectedness learning. For $\hat{n}_{u,i}$, we adopted a pairwise loss function to learn the unexpectedness facet of serendipity:

$$L_{unexp}(\Delta) = \sum_{(u,i,j)\in\mathcal{N}} -log\,\sigma(\hat{n}_{u,i} - \hat{n}_{u,j}) \tag{6}$$

Δ is the set of model parameters. $\mathcal{N}$ denotes the set of training instances for unexpectedness: $\mathcal{N} : \{(u,i,j)|i \in \mathcal{N}_u^+ \wedge j \in \mathcal{N}_u^-\}$, where $\mathcal{N}_u^+$ denotes the set of items that has been regarded as unexpecting by the user u. In contrast, $\mathcal{N}_u^-$ denotes the negative samples.

The key problem becomes how to generate the ground truth for unexpectedness in order to select positive and negative samples. In Psychology, unexpectedness is defined as violation of expectation [15]. We propose a computational operationalization of this definition in order to "calculate" the ground truth of unexpectedness. Specifically, we first calculated the conditional likelihood of seeing an item given a user's sequence of interacted items. We then used a low level of such conditional likelihood as a high level of

unexpectedness. Let $I = \{i_1, i_2, \ldots, i_{|I|}\}$ denotes the set of items in the data, and $T_u = \{i_1^u, i_2^u, \ldots, i_n^u\}$ denotes a sequence of interacted items for the user u. The level of unexpectedness is calculated as:

$$unexp_{(u,i)} = -log\, p(i|T_u) \tag{7}$$

The conditional likelihood $p(i|T_u)$ could be seen as a user's expectation of seeing the item i. The low value of $p(i|T_u)$ could be seen as a violation of such an expectation, and therefore is unexpectedness. The negative sign is to indicate the opposite relationship and the logarithm function is to smooth the larger values. Using the Law of Total Probability, we could rewrite the conditional probability in Equation 7 as:

$$unexp_{(u,i)} = -log\, p(i|T_u) = -log \sum_{i_s^u \in T_u} p(i|i_s^u) p(i_s^u) \tag{8}$$

where i_s^u is a user's historically interacted item, and $p(i|i_s^u)$ could be calculated as:

$$p(i|i_s^u) = \frac{n(i, i_s^u)}{\sum_{i \in I} n(i, i_s^u)} \tag{9}$$

$n(i, i_s^u)$ is the count of co-occurrences for an item i and i_s^u in all user profiles. The denominator is the sum of the co-occurrence counts for each item in the item set with i_s^u . Considering that in most recommendation datasets, it is usually the case that most items in the item set are not observed to co-occur with a specific user interacted item i_s^u. Therefore, $p(i|i_s^u)$ may end up with being 0 for many items undiscriminatingly. Therefore, we added a smoothing factor ε to address the undiscriminating problem. ε represents a small pseudo count of co-occurrence of every item i and i_s^u. There would be a problem with using such a constant smoothing factor ε. All of the unobserved co-occurrences would be assigned with the same pseudo account ε. A better way is to make this ε proportional to the item i's popularity (or occurrence frequency) in the dataset, because the popular items are more likely to co-occur with other items. Therefore, we used a popularity-sensitive smoothing factor $\mu p(i)$. The $p(i|i_s^u)$ with such a smoothing factor becomes:

$$p(i|i_s^u) = \frac{n(i, i_s^u) + \mu p(i)}{\sum_{i \in I} n(i, i_s^u) + \mu} \tag{10}$$

Therefore, the level of unexpectedness $unexp_{(u,i)}$ is calculated as:

$$unexp_{(u,i)} = -log\, p(i|T_u) = -log \sum_{i_s^u \in T_u} \frac{n(i, i_s^u) + \mu p(i)}{\sum_{i \in I} n(i, i_s^u) + \mu} p(i_s^u) \tag{11}$$

All of the components on the right side of this Equation 11 could be pre-calculated from the dataset before the model training. After calculating all the items' $unexp_{(u,i)}$ values for a user u, we selected the items with the top values as the positive samples for unexpectedness for the loss function L_{unexp} in Equation 6. Meanwhile the negative samples are the items with the bottom values.

4.3.2 Relevance Learning and Relevance Ground Truth Generation. Relevance is also an essential component for serendipity. Recommending serendipity is not recommending items randomly but should be relevant to users. It is believed that at the heart of the experience of serendipity was the emergence of powerful personal relevance out of seemingly random coincidence of events [8]. The component of relevance is to restrict the scope of serendipity, preventing unlimited recommendations from annoying users. Relevance is a well-established target for recommendation models. We adopted the base model again to obtain the relevance representation $\mathbf{h_{rel}}$ from the user history. Then the predicted relevance score is:

$$\hat{r}_{u,i} = \sigma(\mathbf{q''^{T}_{cand}} \cdot \mathbf{h_{rel}}) \tag{12}$$

where the vector $\mathbf{q''^{T}_{cand}}$ is a candidate item in the item set $\mathbf{I}$ for the relevance learning. We then adopted a pairwise loss function of relevance:

$$L_{rel}(\Lambda) = \sum_{(u,i,j) \in \mathcal{R}} -log\, \sigma(\hat{r}_{u,i} - \hat{r}_{u,j}) \tag{13}$$

Λ is the set of model parameters. $\mathcal{R}$ denotes the set of training instances: $\mathcal{R} : \{(u, i, j)|i \in \mathcal{R}_u^+ \wedge j \in \mathcal{R}_u^-\}$, where $\mathcal{R}_u^+$ denotes the set of positive samples for relevance, and $\mathcal{R}_u^-$ denotes the negative samples.

For the purpose of auto-generating labels of relevance, we used the common approach in the recommendation research community: the observed interaction between a user u and an item i establishes a relevance label for this user-item pair. Other user-item pairs with no observed interactions are deemed as irrelevance.

4.3.3 Self-Enhanced Serendipity Learning. Under the guidance of the two auxiliary loss functions, we have obtained two auxiliary representations $\mathbf{h_{unexp}}$ and $\mathbf{h_{rel}}$ to supplement the main representation $\mathbf{h_{seren}}$ for serendipity. We used the hyper-parameters λ and α to aggregate them as a unified representation $\mathbf{h}$:

$$\mathbf{h} = \mathbf{h_{seren}} + \lambda(\alpha \mathbf{h_{unexp}} + (1 - \alpha)\mathbf{h_{rel}}) \tag{14}$$

where λ is the importance weight of the *SerenEnhance* module to the base deep learning model; and α is the trade-off importance weight between $\mathbf{h_{unexp}}$ and $\mathbf{h_{rel}}$. In the experiment section, we will experiment different λ and α values.

After obtaining the final representation $\mathbf{h}$, the model predicts the final serendipity score as:

$$\hat{s}_{u,i} = \sigma(\mathbf{q^{T}_{cand}} \cdot \mathbf{h}) \tag{15}$$

5 EXPERIMENTS

5.1 The Ground Truth Data

We used the domain of books to construct our ground truth dataset on serendipity. Book reading behavior is highly driven by personal taste, and its experience is highly subjective, laying a rich ground for serendipity occurrence. Book reviews datasets are largely available. The scale and richness of the dataset warrant the high quality of the ground truth dataset and high performance of a deep learning model. We used the Amazon Review Data [12] as the base. Its book review data contains 51,311,621 book reviews written by 3,096,817 users for 1,887,748 books from 05/20/1996 to 07/23/2014. The book review data is in the *JSON* format. We manually curated 7 keywords, as in the Table 2, in the first stage of *URCW*.

We have spent 10 months to collect this ground truth data. As the result, we collected 41,340 user judgements on 10,000 reviews from MTurk. Among these 10,000 reviews, 2,557 reviews were labeled as serendipity experiences. For the 2,557 serendipity reviews, they were written by 2,346 users (review writers) on 2,227 books. This

Table 2: Keywords for extracting reviews about serendipity experiences

Keyword	No. of reviews with this keyword
1. stumble on/stumble upon	14,535
2. find/discover/buy/purchase∗by chance	4,242
3. find/discover/buy/purchase∗by accident	6,426
4. serendipity/serendipitous	1,852
5. find/discover/buy/purchase∗unexpectedly	2,584
6. find/discover/buy/purchase∗surprisingly	15,367
7. find/discover/buy/purchase∗accidentally	21

is only the "base" of our engineered ground truth dataset. What makes the "base" valuable is that through these 2,346 users, we are able to link all the other reviews and items for the 2,346 users, not only those reviews of serendipity or books of serendipity. That way, the records in the "base" are not by their own, but are put into a context of the users with their entire history. As the result, *SerenLens* contains a total number of 265,037 reviews on 113,876 books. Table 3 shows the key statistics of *SerenLens* and its engineering process. Our *SerenLens* dataset is publicly available at https://github.com/zhefu2/SerenLens.

Table 3: Key statistics of *SerenLens* and of its collecting process using MTurk

HITs	Total HIT tasks assigned	8,268
	Total HIT tasks accepted	4,396
Worker Judgements	Total worker judgements collected	41,340
	Total worker judgements accepted	21,980
	Judgements with initial agreement	16,040
	Judgements need a third opinion	3,960
	Reviews with judgements	10,000
	Reviews of serendipity	2,557
	Reviews of non-serendipity	7,443
***SerenLens* Dataset**	Users involved in the reviews of serendipity	2,346
	Other reviews involved for the involved users	262,691
	Total reviews involved	**265,037**
	Books involved in the reviews of serendipity	2,227
	Total books involved	**113,876**

5.2 Evaluation Metrics and Baseline Models

Since serendipity is sparse ($2,557/265,037 \approx 1.0\%$) in *SerenLens*, we adopted a recall-based metric, Hit Ratio (HR). HR_{seren}@k measures the proportion of times a serendipity item is retrieved in the top-k position (1 for yes and 0 otherwise). In order to take the rank information into consideration and assign higher weights on higher ranks, we propose another metric called Serendipity-Based Normalized Discounted Cumulative Gain ($\text{NDCG}_{\text{seren}}$) based on the well-known metric Normalized Discounted Cumulative Gain (NDCG). $\text{NDCG}_{\text{seren}}$@k is calculated as:

$$NDCG_{seren}@k = \sum_{i=1}^{k} \frac{serendipity\ score(1\ or\ 0)}{log_2(i+1)} \tag{16}$$

In addition to evaluating serendipity, we conducted a second set of experiments to evaluate how much sacrifice on relevance the recommendation models were making (if there was) in order to accommodate serendipity. Therefore, we used the two standard metrics for relevance evaluation: HR@k (the "hit" here means hitting a relevant item in the top-k position) and Normalized Discounted Cumulative Gain (NDCG). For all of the metrics above, a higher value indicates a better performance.

To evaluate the performance of the proposed models, we selected the following two groups of representative baseline recommendation models. The first group consists of the well-known deep learning recommendation models for serendipity:

- **DESR** [11]: It is a state-of-the-art serendipity-oriented recommendation model. It combines user-item relevance with the user preference direction to recommend serendipitous items. It infers the user preference direction through the user's long-term preferences and short-term demands.
- **PURS** [9]: It is a state-of-the-art serendipity-oriented recommendation model. It is built on top of a traditional relevance oriented GRU model, and adds an unexpectedness component using self-attentive MLPs. The self-attention weights are pre-calculated according to the self-defined unexpectedness as the item's average distance to each cluster center of a user's interests.
- **SNPR** [23]: This is the most recent serendipity-oriented recommendation model. It leverages the state-of-the-art transformer technique to model all the potential dependencies among items in a user's history. It treats serendipity learning as two separate learning tasks: relevance learning using the traditional training dataset, and unexpectedness learning using a self-calculated training dataset. Then the model combines the two separate learning tasks with a pre-specified coefficient.

The second group is the Transformer-based recommendation models for the relevance task:

- **SASRec** [6]: It is a next-item sequential recommendation method based on the Transformer's architecture, which employs a multi-head self-attention mechanism to explore implicit user interactions.
- **BERT4Rec** [19]: It is an improvement of SASRec and also the state-of-the-art sequential recommendation method, which models user behavior sequences with a bidirectional self-attention network via a Cloze task.

Both of the two groups of models are the state-of-the-art deep learning models published in top venues in recent years.

5.3 Deep Learning Models' Setting

We trained the proposed model, *Base+SerenEnhance*[1], and the baseline models on the *SerenLens* data. We implemented the PURS, SASRec, and BERT4Rec based on open-source codes and implemented the DESR and SNPR based on the description of the original papers. We compared *Base+SerenEnhance* and the baseline models using the serendipity-based metrics (HR_{seren}@k and $\text{NDCG}_{\text{seren}}$@k) and the relevance-based metrics (HR@k and NDCG@k). 80% of the data

[1]the code is released at https://github.com/zhefu2/SerenEnhance

Table 4: The performance comparison of different models. The reported number is the average of 5 folds. The best results in each column are bolded and the second best results are underlined. $*$ denotes that our proposed model has statistically significant differences with at least three of the six baseline models under a two-tailed t-test with $p < 0.05$.

	Serendipity-based metrics					Relevance-based metrics				
	$HR_{seren}@1$	$HR_{seren}@5$	$HR_{seren}@10$	$NDCG_{seren}@5$	$NDCG_{seren}@10$	HR@1	HR@5	HR@10	NDCG@5	NDCG@10
PURS	5.76%	22.60%	32.20%	0.139	0.170	2.35%	6.82%	13.86%	0.039	0.059
DESR	6.25%	23.02%	36.67%	0.144	0.178	1.92%	6.61%	16.63%	0.034	0.072
SNPR	7.46%	24.09%	38.81%	0.149	0.192	2.20%	7.04%	18.76%	0.040	0.082
SASRec	6.13%	25.33%	41.37%	0.157	0.209	**3.33%**	11.25%	20.83%	0.068	0.099
BERT4Rec	8.03%	27.02%	41.79%	0.166	0.214	3.13%	**13.28%**	**23.05%**	**0.076**	**0.107**
Base + SerenEnhance ($\lambda = 0.6, \alpha = 0.6$)	**9.81%***	**30.49%***	**45.63%***	**0.329***	**0.364***	2.35%*	9.81%*	20.04%*	0.042*	0.102*

in *SerenLens* was used for training and the rest 20% was for testing. For all models, we adopted 5-fold cross-validation to evaluate the performance. We trained our models using the Adam optimizer. We set the learning rate 0.0001, the hidden dimension 128, the dropout rate 0.2, and the regularizer decay 0.001 for all the models. Other model-specific hyper-parameters were set either following their original studies or adjusting for the training performance in this study. For all of the baseline models, we reported the results using the optimal hyper-parameter settings. For the proposed *Base+SerenEnhance* model, the optimal values of λ and α will be examined and discussed in the experiments. In addition, for fair comparison, we set the number of the Transformer layers as 3 and the number of heads in the multi-head attention sublayer as 2 for the models involving Transformers: *Base+SerenEnhance*, SASRec, BERT4Rec, and SNPR. For the second set of the experiments for evaluating relevance, we used the same trained models (trained on *SerenLens*) as the first set of experiments, but the test set was changed to the relevance ground truth automatically collected for the *SerenEnhance* module, as described in Section 4.3.2.

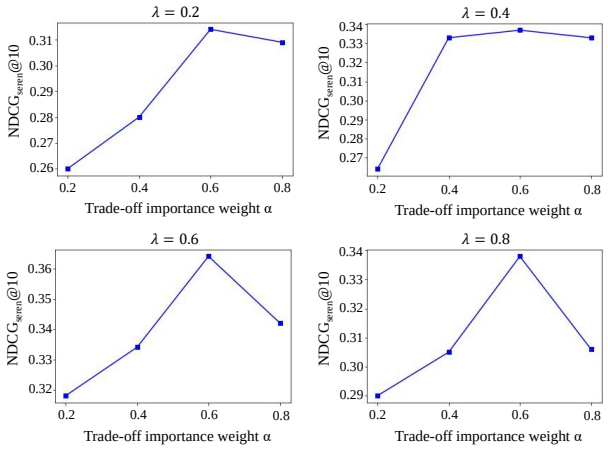

Figure 3: The recommendation performances of *Base+SerenEnhance* under different λ and α values

5.4 Results

Effects of Hyper-Parameters. We investigated the influence of the self-enhanced module importance weight λ and the unexpectedness-relevance trade-off importance weight α on the recommendation performance. As in Figure 3, when λ is 0.6, $NDCG_{seren}@10$ could reach the highest value. Therefore we select the λ value as 0.6. When λ is fixed, the $NDCG_{seren}@10$ score increases when α changes from 0.2 to 0.6, and then decreases when α changes from 0.6 to 0.8. The highest $NDCG_{seren}@10$ is achieved when α takes the value of 0.6. We observed similar trends using other evaluation metrics, such as $HR_{seren}@5$, $NDCG_{seren}@5$, etc. Therefore, both λ and α's optimal values are 0.6. In the following subsection, we will only report the result of *Base+SerenEnhance* with $\lambda = 0.6$ and $\alpha = 0.6$.

Overall Performance Comparison. From Table 4, it can be observed that our proposed *Base+SerenEnhance* model achieves the best performance among all the models on the serendipity-based metrics. In general, the Transformer-based models (SASRec, BERT4Rec, SNPR, and *Base+SerenEnhance*) have a better performance than the non-Transformer-based models (PURS and DESR). It is interesting to note that SASRec and BERT4Rec were originally designed for relevance-oriented recommendation tasks, but now perform better than PURS and DESR, the two serendipity-oriented baseline models, suggesting the power of Transformers. PURS and DESR are not based on Transformers. PURS leveraged sequential networks of Gated Recurrent Unit (GRU) and DESR chronological ordered Capsule Network. The experiment results confirmed our choice of the Transformer technique for representing a user for identifying serendipity.

In addition to serendipity, we conducted a second set of experiments on relevance to evaluate how much sacrifice (if there is) the serendipity-oriented model needs to make on relevance in order to accommodate serendipity. Compared to SASRec and BERT4Rec that were designed for the recommendation relevance tasks, our model has lower relevance performance. What is worth to mention is that the BERT4Rec model, the vanilla version of the BERT model [2] used in recommender systems, is reasonably good at discovering serendipity while maintaining a reasonable relevance, demonstrating the power and the wide adaptability of the BERT model. On the other hand, compared with the state-of-the-art serendipity models PURS, DESR, and SNPR, the *Base+SerenEnhance* model has better relevance results, meaning our proposed models are able to achieve higher serendipity at a lower cost of relevance.

Ablation Study. We further conducted experiments to evaluate the effectiveness of the key components of our proposed *Base+SerenEnhance* model. The evaluated components are the main serendipity learning in the base deep learning model (BaseSeren), the unexpectedness learning in *SerenEnhance* (Unexp), and the relevance learning in *SerenEnhance* (Rel). The evaluation was conducted by removing each of them or a combination of them, to test the model performance change. The results are presented in Table 5.

As shown in Table 5, without the unexpectedness learning (w/o Unexp) or without the relevance learning (w/o Rel) in *SerenEnhance*, the performance drops on both HR_{seren} and $NDCG_{seren}$ metrics. It demonstrates the effectiveness of both the unexpectedness learning and the relevance learning in *SerenEnhance*. More interestingly, removing the unexpectedness learning results in more drastic model performance decreases than removing the relevance learning, suggesting that the unexpectedness learning has a relatively higher impact on the serendipity recommendation performance.

In addition, we removed both the unexpectedness learning and relevance learning at the same time (w/o Unexp & Rel). That way, the model downgraded to the base deep learning model only. Its performance drops around 10% for HR_{seren} and more than 20% for $\text{NDCG}_{\text{seren}}$. The results demonstrate the value of the *SerenEnhance* module to strengthen the base deep learning model.

Furthermore, we removed the base serendipity learning in the base deep learning model (w/o BaseSeren), making the model training process only rely on the *SerenEnhance* module. That way, this model was not trained on the *SerenLens* dataset, but only on those auto-generated labels on unexpectedness and relevance. We observed the largest model performance decrease on both HR_{seren} scores and $\text{NDCG}_{\text{seren}}$ scores. The results suggest that learning serendipity representation directly from the collected *SerenLens* dataset is the core component of our model, suggesting the core value of *SerenLens* and the base deep learning model.

Table 5: The ablation study results on different model variants. The reported number is the average of 5 folds. The best results in each column are bolded and ↓ indicates performance drop more than 20% relative to the original *Base+SerenEnhance* model.

Architecture	HR_{seren}@1	HR_{seren}@5	HR_{seren}@10	$\text{NDCG}_{\text{seren}}$@5	$\text{NDCG}_{\text{seren}}$@10
Base+SerenEnhance ($\lambda = 0.6, \alpha = 0.6$)	**9.81%**	**30.49%**	**45.63%**	**0.329**	**0.364**
w/o Unexp ($\lambda = 0.6, \alpha = 0.0$)	7.46%↓	24.52%	39.87%	0.173↓	0.233↓
w/o Rel ($\lambda = 0.6, \alpha = 1.0$)	8.52%	27.29%	42.00%	0.281	0.312
w/o Unexp & Rel ($\lambda = 0.0$)	8.10%	25.37%	40.94%↓	0.166↓	0.230↓
w/o BaseSeren ($\alpha = 0.6$)	2.13%↓	10.66%↓	18.55%↓	0.122↓	0.136↓

5.5 A Case Study

To have an intuitive understanding of model results, we selected an example user to showcase the better serendipity recommendation results of *Base+SerenEnhance*. As shown in Figure 4, a user who loves thriller books as in his/her historical book sequences, had a serendipity experience on a book titled *The Princess Bride*, as in his/her review:

"...This is one of my all time favorite books. I stumbled upon the paperback in the high school library and was immediately hooked..."

This book is an adventure book with the romance element in it. It is not quite the user's typical type of book. As shown in Figure 4, the Base Model recommended it in the 5^{th} rank (as highlighted in the red box in the recommendation list) but *Base+SerenEnhance* was able to identify this book as the top serendipity recommendation.

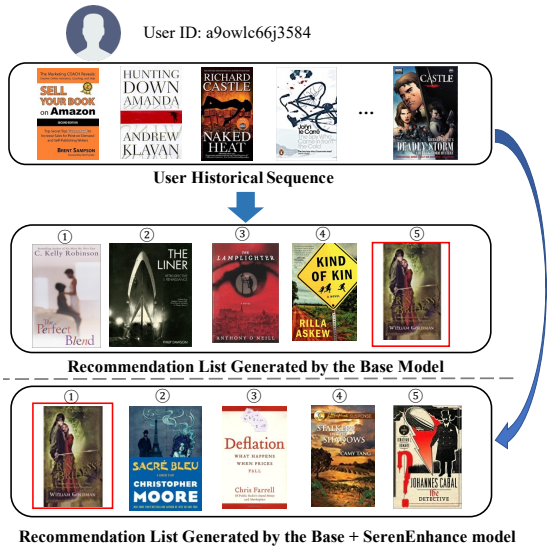

Figure 4: Two top-5 recommendation lists generated by the Base Model and *Base+SerenEnhance*

Other than the *The Princess Bride* (the only ground truth serendipitous book for this user in *SerenLens*), if we look at the other recommendations, it can be observed that the recommended book list generated by *Base+SerenEnhance* was subtly different from the list generated by the Base Model: a bit bolder and more deviating from the usual type of thriller books as in the user's historical book sequence.

6 CONCLUSION

This paper contributes a new ground truth data collecting approach called *URCW*, a large serendipity ground truth dataset called *SerenLens*, and a novel self-enhanced module for serendipity recommendation called *SerenEnhance*. Extensive experimental results show that a base deep learning model trained on *SerenLens* as well as with the help of the *SerenEnhance* module, outperforms the state-of-the-art baseline models in predicting serendipity at relatively lower cost of relevance. We analyzed the effectiveness of different model components and also used a case study to demonstrate how our proposed *SerenEnhance* helps serendipity recommendations. For future works, we are interested in extending the current data and model to cross-domain recommendations for serendipity.

ACKNOWLEDGMENTS

This research is supported by National Science Foundation (NSF) (Award #1910696). We are grateful to NSF to make this research possible.

REFERENCES

[1] Keping Bi, Qingyao Ai, and W Bruce Croft. 2020. A transformer-based embedding model for personalized product search. In *Proceedings of the 43rd International ACM Conference on Research and Development in Information Retrieval (SIGIR)*. ACM, 1521–1524.

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[3] F Maxwell Harper and Joseph A Konstan. 2015. The movielens datasets: History and context. *Acm Transactions on Interactive Intelligent Systems (TIIS)* 5, 4 (2015), 1–19.

[4] Jonathan L Herlocker, Joseph A Konstan, Loren G Terveen, and John T Riedl. 2004. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems (TOIS)* 22, 1 (2004), 5–53.

[5] Jizhou Huang, Shiqiang Ding, Haifeng Wang, and Ting Liu. 2018. Learning to recommend related entities with serendipity for web search users. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)* 17, 3 (2018), 1–22.

[6] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 197–206.

[7] Denis Kotkov, Joseph A Konstan, Qian Zhao, and Jari Veijalainen. 2018. Investigating serendipity in recommender systems based on real user feedback. In *Proceedings of the 33rd Annual ACM Symposium on Applied Computing (SAC)*. ACM, 1341–1350.

[8] Tuck Wah Leong, Peter Wright, Frank Vetere, and Steve Howard. 2010. Understanding experience using dialogical methods: The case of serendipity. In *Proceedings of the 22nd Conference of the Computer-Human Interaction Special Interest Group of Australia on Computer-Human Interaction*. 256–263.

[9] Pan Li, Maofei Que, Zhichao Jiang, Yao Hu, and Alexander Tuzhilin. 2020. PURS: personalized unexpected recommender system for improving user satisfaction. In *Proceedings of the 14th ACM Conference on Recommender Systems (RecSys)*. ACM, 279–288.

[10] Xueqi Li, Wenjun Jiang, Weiguang Chen, Jie Wu, and Guojun Wang. 2019. Haes: A new hybrid approach for movie recommendation with elastic serendipity. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM)*. ACM, 1503–1512.

[11] Xueqi Li, Wenjun Jiang, Weiguang Chen, Jie Wu, Guojun Wang, and Kenli Li. 2020. Directional and explainable serendipity recommendation. In *Proceedings of The Web Conference 2020*. ACM, 122–132.

[12] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th ACM International Conference on Research and Development in Information Retrieval (SIGIR)*. ACM, 43–52.

[13] Lori McCay-Peet and Elaine G Toms. 2010. The process of serendipity in knowledge work. In *Proceedings of the 3rd Symposium on Information Interaction in Context (IIix)*. 377–382.

[14] Robert K Merton and Elinor Barber. 2011. The travels and adventures of serendipity. In *The Travels and Adventures of Serendipity*. Princeton University Press.

[15] Wulf-Uwe Meyer, Rainer Reisenzein, and Achim Schützwohl. 1997. Toward a process analysis of emotions: The case of surprise. *Motivation and Emotion* 21, 3 (1997), 251–274.

[16] Gaurav Pandey, Denis Kotkov, and Alexander Semenov. 2018. Recommending serendipitous items using transfer learning. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM)*. ACM, 1771–1774.

[17] Theodore G Remer. 1965. *Serendipity and the three princes: From the Peregrinaggio of 1557*. Norman, U. Oklahoma P.

[18] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*. 452–461.

[19] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM)*. ACM, 1441–1450.

[20] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. 30 (2017).

[21] Liwei Wu, Shuqing Li, Cho-Jui Hsieh, and James Sharpnack. 2020. SSE-PT: Sequential recommendation via personalized transformer. In *Proceedings of the 14th ACM Conference on Recommender Systems (RecSys)*. ACM, 328–337.

[22] Yuanbo Xu, Yongjian Yang, En Wang, Jiayu Han, Fuzhen Zhuang, Zhiwen Yu, and Hui Xiong. 2020. Neural serendipity recommendation: Exploring the balance between accuracy and novelty with sparse explicit feedback. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 14, 4 (2020), 1–25.

[23] Mingwei Zhang, Yang Yang, Rizwan Abbas, Ke Deng, Jianxin Li, and Bin Zhang. 2021. SNPR: A Serendipity-Oriented Next POI Recommendation Model. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management (CIKM)*. ACM, 2568–2577.