HAES: A New Hybrid Approach for Movie Recommendation with Elastic Serendipity

Xueqi Li
lee_xq@hnu.edu.cn
Hunan University

Wenjun Jiang[*]
jiangwenjun@hnu.edu.cn
Hunan University

Weiguang Chen
cwg@hnu.edu.cn
Hunan University

Jie Wu
jiewu@temple.edu
Temple University

Guojun Wang
csgjwang@gzhu.edu.cn
Guangzhou University

ABSTRACT

Recommendation systems provide good guidance for users to find their favorite movies from an overwhelming amount of options. However, most systems excessively pursue the recommendation accuracy and give rise to over-specialization, which triggers the emergence of serendipity. Hence, serendipity recommendation has received more attention in recent years, facing three key challenges: subjectivity in the definition, the lack of data, and users' floating demands for serendipity. To address these challenges, we introduce a new model called **HAES**, a **H**ybrid **A**pproach for movie recommendation with **E**lastic **S**erendipity, to recommend serendipitous movies. Specifically, we (1) propose a more objective definition of serendipity, *content difference* and *genre accuracy*, according to the analysis on a real dataset, (2) propose a new algorithm named JohnsonMax to mitigate the data sparsity and build weak ties beneficial to finding serendipitous movies, and (3) define a novel concept of elasticity in the recommendation, to adjust the level of serendipity flexibly and reach a trade-off between accuracy and serendipity. Extensive experiments on real-world datasets show that **HAES** enhances the serendipity of recommendations while preserving recommendation quality, compared to several widely used methods.

CCS CONCEPTS

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

KEYWORDS

Movie Recommendation, Serendipity, Elasticity

ACM Reference Format:
Xueqi Li, Wenjun Jiang, Weiguang Chen, Jie Wu, and Guojun Wang. 2019. **HAES**: A New **H**ybrid **A**pproach for Movie Recommendation with **E**lastic **S**erendipity. In *The 28th ACM International Conference on Information and Knowledge Management (CIKM '19), November 3–7, 2019, Beijing, China.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3357384.3357868

[*]Wenjun Jiang is the corresponding author.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
CIKM '19, November 3–7, 2019, Beijing, China
© 2019 Association for Computing Machinery.
ACM ISBN 978-1-4503-6976-3/19/11…$15.00
https://doi.org/10.1145/3357384.3357868

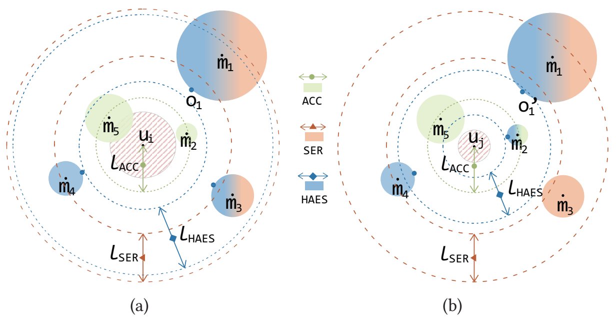


Figure 1: An example to illustrate the difference among accuracy-oriented method (ACC), recommendation with fixed serendipity (SER), and our method (HAES).

Table 1: Statistics of recommendations in Figure 1

# of subfigure	ACC	SER	HAES
(a)	m_2, m_5	m_1, m_3	m_1, $\mathbf{m_3}$, m_4
(b)	m_2, m_5	m_1, m_3	m_1, $\mathbf{m_2}$, m_4

1 INTRODUCTION

With the development of the Internet, it becomes more and more flexible for users to choose where, when and how to watch movies, leading to the rapid growth in the number and genre of movies available. However, the delightful variation also brings users a huge trouble in finding movies they potentially like. Recommendation systems play an indispensable role in mitigating this problem, by providing interesting options in line with the users' profile [26]. For catering to users' preferences, the majority of methods analyze their past behaviors and adopt collaborative filtering to generate the corresponding recommendations [19, 32, 38]. As a result, over-specialization and the long tail effect grow increasingly obvious, calling for more researches on serendipity-oriented recommendation. Although some work has been done, they usually assume that there is a fixed level of serendipity suitable for all users, which results in unrelated recommendations. In this paper, we strive to develop a deep study of elastic serendipity and its application in movie recommendation. To be specific, we propose the concepts of *user elasticity* and *movie elasticity*, and put forward a hybrid approach for movie recommendation with elastic serendipity (**HAES**).

We provide an example in Figure 1 to show the effect our approach tries to reach, where we compare our method with two other commonly used approaches: **ACC** (accuracy-based method) and **SER** (recommendation approach with fixed serendipity). In the figure, the solid circles denote the elasticity (defined in Section 3.1) of users (e.g., u_i, u_j) or movies (e.g., m_1, ... , m_5). The larger a circle is, the stronger is the user's ability to accept different movies (the bigger is the possibility of movie being adopted). The dashed circles represent the ranges of recommendation, e.g., l_{ACC} denotes the recommendation range of **ACC**, l_{SER} is that of **SER**, and l_{HAES} is that of **HAES**. We would recommend a movie if the edge closest to the user (e.g., o_1 for m_1 in Figure 1(a), o_1' for m_1 in (b)) is within our recommendation range, while the other two methods generate recommendations depending on whether the center of the movie is within their recommendation ranges. The difference indicates the sensitivity of **HAES** to *movie elasticity*. In addition, our approach is also aware of *user elasticity*, which is reflected in the comparison between (a) and (b) in Figure 1. The user in (a), u_i, has the same associations with five movies (m_1, ... , m_5) as u_j in (b), but the elasticity of u_i is larger than that of u_j. Among three methods, only our approach recommends different movies for u_i and u_j (see Table 1). For example, we recommend m_3 only to u_i, since the elasticity of u_j is too small to accept it. Considering u_i's elasticity is bigger, it's reasonable to recommend movies with weak relevance (m_2 is not a suitable option) to improve recommendation serendipity.

Existing approaches primarily aim at increasing the accuracy of recommendations [4, 7, 37], achieving great success with the powerful aid of deep learning [2, 27, 35]. However, user satisfaction doesn't continually increase with the improvement of recommendation accuracy. On the contrary, it shows a correlation with the serendipity of recommendations [6, 20]. Consequently, it draws increasing attention to researches on serendipity-oriented recommendation. Researchers in the area generally split serendipity into related attributes that are easy to measure [28], such as the relevance, novelty and unexpectedness in [15]. Then, they recommend movies in accordance with these attributes, alleviating over-specialization. However, most researchers define serendipity based on some subjective views and generate recommendations with a fixed degree of serendipity for different users, causing the risk of irrelevant recommendations and decreasing user satisfaction [6, 22]. It's not an easy task to enhance serendipity without lowering the quality of recommendations. There are three major challenges. (1) The definitions of serendipity lack objective evidences. (2) There is very limited labeled data available in serendipity-oriented recommendation. (3) It's difficult to judge what level of serendipity is the most suitable for different movies and users.

Our Motivations. We have three motivations for addressing the above challenges in this paper. (1) *providing a more objective definition on serendipity.* We strive to define serendipity according to the analysis on real-world datasets instead of subjective ideas. (2) *mitigating the lack of data.* We try to dig out attributes related to serendipity and build weak ties among users and movies for alleviating the data sparsity. (3) *recommending movies with elastic serendipity.* We manage to adjust the degree of recommendations' serendipity flexibly for different users and movies.

We try to recommend movies with an adaptive level of serendipity for different users, so as to mitigate the over-specialization, reduce the risk of unrelated recommendations, and make a balance between serendipity and accuracy. To be specific, our objective mainly consists of two parts: digging out user-movie elastic associations and predicting genres with high accuracy. Combining these two parts, we propose a new serendipity-based algorithm named **HAES** and verify its effectiveness on real-world datasets. Our contributions are fivefold, as follows:

- On the basis of a deep analysis on a real-world dataset, Serendipity-2018[1] [14], we develop a more objective definition of *serendipity* by decomposing it into two attributes, *content difference* and *genre accuracy*. (Section 3.1)
- We propose the concepts of *user elasticity* and *movie elasticity*, dynamically balancing serendipity and accuracy of recommended lists. (Sections 3.1 and 4.1)
- We propose JohnsonMax algorithm to optimize user-movie associations, alleviating the data sparsity. (Section 4.2)
- We predict user preferences in movie genre with Recurrent Neural Network (RNN) [21] and achieve significant performance improvements. To the best of our knowledge, this is the first work to apply RNN models into serendipity-oriented recommendation methods. (Section 4.3)
- We conduct comprehensive experiments on MovieLens-1m[2] [8] and MovieLens-latest-small[3] [8], to compare our method with widely used methods. Experimental results demonstrate that **HAES** provides significantly better guide for users to find serendipitous movies while preserving high *genre accuracy*. For example, **HAES** improves *micro_F1* by 53.93% and increases 2.3 times on *difference* compared with the accuracy-oriented method on MovieLens-1m. (Section 5)

2 RELATED WORK

There is a long history of accuracy recommendation systems [1, 3, 25], where the more accurate they are, the more obvious over-specialization becomes, which accelerates the development of serendipity recommendation. In this section, we would briefly review related works on definitions, methods and metrics of serendipity.

Research on definitions of serendipity. The definition of serendipity stems from some abstract descriptions in literature. For example, Walpole describes it as the experience of discovering interesting things by accident [29]. Some researchers crystallize it with some related attributes, such as unexpectedness and usefulness in [6], novelty, relevance and unexpectedness in [17].

Research on methods of serendipity. Some proposed approaches are based on accuracy recommendation methods, and the others are novel , of which the most popular ones are reviewed here. Kotkov et al. [16] apply a greedy strategy to improve recommendation serendipity through resorting the recommended lists. Karpus et al. [13] introduce ontology-based contextual pre-filtering to remove movies particularly familiar to users. Zhang et al. [36] implement three basic approaches to generate corresponding lists respectively. Asides from those accuracy based recommendations, there are also many novel approaches. Niu [23] sets up a framework with the components of unexpectedness, value and learning to recommend

[1]https://grouplens.org/datasets/serendipity-2018/
[2]https://grouplens.org/datasets/movielens/1m/
[3]https://grouplens.org/datasets/movielens/

serendipitous items. Serendipitous recommendations should meet the requirements of low pre-interest and high post-satisfaction in the views of Yang et al. [34]. Transfer learning is adopted by Pandee et al. [24], and Nguyen et al. [22] pose that users are keen on diversity, popularity and serendipity to different extent.

Among existing works, SIRUP [18] is the most similar one to ours, taking into account users' ability to accept serendipity. There are two significant differences between SIRUP and ours: (1) the possibility of movies being adopted is only considered by us; (2) we propose elasticity to quantify the acceptance ability rather than divide users into two groups, which is more flexible and fine-grained.

Research on metrics of serendipity. Here we divide evaluation strategies into two types: online and offline. For online evaluations, most researchers evaluate the performance of recommendation systems with questionnaires [14, 16], while the others take advantage of users' implicit feedback. For example, Ge et al. [6] use facial expression recognition for assessment on the serendipity of recommendations. For offline evaluations, it's a common choice to measure serendipity with related attributes proposed in the definition [18, 36], and we employ this manner in our experiments.

3 PROBLEM STATEMENT AND MODEL OVERVIEW

We illustrate the system settings and concepts, define the problem we solve, and give a brief overview of our solution. Notations are described in Table 2.

3.1 System Settings

We consider the scenario in a movie recommendation system (e.g., MovieLens), where there are users, movies, and ratings. These entities constitute the input space, $I(U, M, S)$, of our work.

3.1.1 *Data Analysis.*

Due to the subjectivity of serendipity and the lack of large-scale datasets, it's almost impossible to recommend movies according to the value of serendipity directly. Motivated by the great progress achieved by methods based on serendipity decomposition, we employ a similar approach to define serendipity. Our difference lies in that we exploit the objective analysis rather than subjective ideas as the foundation for defining serendipity.

We analyze a real-world dataset, Serendipity-2018 [14] with 2150 records, where 481 users rate 1678 movies on eight statements (see Table 3). In these statements, the objectives of serendipity recommendation are s7 and s8, and the others are attributes related to the serendipity. We adopt linear regression to acquire the associations w between the objective and other attributes as follows:

$$s7 + \theta * s8 = w * [s1, s2, s3, s4, s5, s6] + b, \tag{1}$$

where θ indicates the contribution of s8 (broadening user horizons), we use $\theta = 1$ in this paper, b is the random error. We learn the value of w using the dataset Serendipity-2018 and display the weights of related attributes in Figure 2. It indicates that there are stronger associations between the serendipity and s3, s4, s5 (interesting to users, beyond their ability to discover, and different from history).

3.1.2 *The Key Concepts.*

Based on the above observation, we conclude that serendipitous movies should meet the requirements of being different from users' past behaviors and relevant to their interests. Here, we embody the difference as *content difference* and the relevance as *genre accuracy*, and we would verify its justifiability in the experiments (Section 5.1). Consequently, we propose the definitions of *content difference*, *genre accuracy*, and *serendipity*.

Definition 1: Content Difference. A movie with *content difference* is different from the history of the target user u_t (i.e., a movie with weak similarity to the movies u_t has rated).

Definition 2: Genre Accuracy. A movie with *genre accuracy* is the one that meets u_t's short-term preferences in movie genre.

Table 2: Notations

Symbol	Description
U	users $\{u_0, u_1, \dots, u_n\}$
M	movies $\{m_0, m_1, \dots, m_k\}$
G	movie genres $\{G_0, G_1, \dots, G_k\}$
V	nodes in relevance network, $V = U \cup M$
S	rating matrix
I	the input space, $I = (U, M, S)$
$R(i, j)$	the relevance factor from node i to j
$E(u_i, m_j)$	the elasticity factor between u_i and m_j
N	the relevance network, $N = (V, R)$

Table 3: Serendipity-2018 Statements

No.	Description	Abbreviation
s1	The first time I heard of this movie was when MovieLens suggested it to me.	unknown
s2	MovieLens influenced my decision.	influence
s3	I expected to enjoy this movie before watching it for the first time.	interest
s4	This is the type of movie I would not normally discover on my own; I need a recommender system like MovieLens.	beyond users' ability to discovery
s5	This movie is different from the movies I usually watch.	different
s6	I was surprised that MovieLens picked this movie to recommend to me.	surprise
s7	I am glad I watched this movie.	satisfaction
s8	Watching this movie broadened my preferences. Now I am interested in a wider selection of movies.	broaden users' preferences

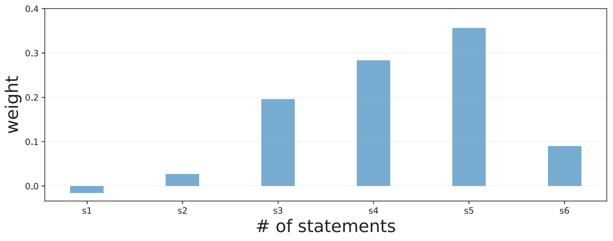


Figure 2: Weights of each related features.

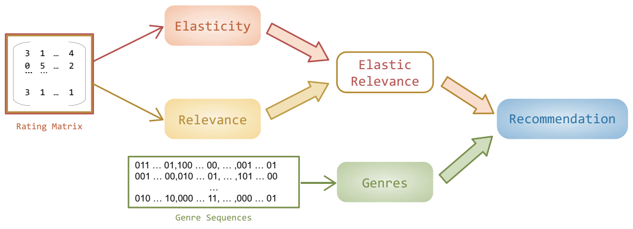


Figure 3: The framework of HAES.

Definition 3: Serendipity. Serendipity represents both attractiveness and surprise [9]. Based on the the results of *Data Analysis* (Section 3.1.1), we define it as *content difference* and *genre accuracy*.

In the serendipity-oriented recommendation, it's essential to be aware of individual characteristics (e.g., *user elasticity*, *movie elasticity*), and the associations among different users and movies.

Definition 4: User Elasticity. The user elasticity, $E(u_t)$, is the ability of u_t to accept movies different from the past behaviors, measured by the genre diversity of movies that are relevant to u_t.

Definition 5: Movie Elasticity. The movie elasticity of m_t, $E(m_t)$, is the possibility of m_t being adopted by different users, measured by the diversity of the corresponding user group $U(m_t)$.

Definition 6: Relevance Network. A relevance network is a directed graph, $N = (V, R)$, where $V = U \cup M$ is a set of nodes, standing for movies and users, and $R \subseteq V^2$ is a set of directed edges indicating the asymmetric associations between different nodes.

3.2 Problem Definition

In this subsection, we formally define the problem of Movie Recommendation with Elastic Serendipity (**MRES**).

Input. The input of **MRES** are an initial user-movie network, $I(U, M, S)$, and a target user u_t ($u_t \in U$), where U is a user set, M is a movie set, S is a user-movie rating matrix.

Output. The algorithm is to generate a list of potential movies, M_t, that may increase the satisfaction of u_t and broaden u_t horizons.

Objective. Based on our definition of serendipity, *content difference* (difference) and *genre accuracy* (accuracy), we try to make serendipity recommendations by solving the following problem:

$$Maximize \quad difference(M_t, u_t) + accuracy(M_t, u_t), \tag{2}$$

where $M_t \subset M$, and each movie of M_t is within u_t' elasticity range.

3.3 Solution Overview

We give a brief overview of the proposed **HAES** (see Figure 3): (1) **Elasticity.** This component measures the elasticity of users and movies to provide adaptive serendipity. (2) **Relevance.** For acquiring latent associations among users and movies, we introduce an asymmetric metric to build the relevance network at first, and then propose the JohnsonMax algorithm to update it. (3) **Genres.** In order to preserve recommendation quality, we apply GRU to predict users' short-term preferences in genre. (4) **Recommendation.** We filter the candidates with genres predicted by genre component and then recommend movies with elasticity and relevance, expecting to create a balance between serendipity and accuracy.

4 HAES: ALGORITHM DETAILS

In this section, we present the technical details of components in **HAES**: quantifying elasticity, building relevance network, predicting genres, and recommendation with elastic serendipity.

4.1 Quantifying Elasticity

We make a quantification of the elasticity to generate recommendations with elastic serendipity for users, so as to dynamically meet their varying demands for serendipity and accuracy [22].

User Elasticity. We measure the *user elasticity* of u_i with the diversity in movie genres related to u_i. As stated in [18], the more movie genres u_i has, the stronger is his ability to accept different movies. Suppose $G(u_i)$ is a genre set related to u_i, $G(U) = \{G(u_0), G(u_1), ..., G(u_n)\}$, and $G_{max}(U)$ is the element of $G(U)$ containing the most genres, the elasticity of u_i, $E(u_i)$, is calculated as follows:

$$E(u_i) = \frac{|G(u_i)|}{|G_{max}(U)|}. \tag{3}$$

Movie Elasticity. We regard the diversity of corresponding user (who has rated the movie) groups as a reference to measure the *movie elasticity*, considering users' age, occupation, and the group size. To be specific, we suppose $A(m_i)$ denotes the age set of users associated to m_i and $O(m_i)$ is about occupation; α and β indicate the contributions of age and occupation, respectively; $U(m_i)$ is a set of users related to m_i. We calculate the diversity factor of m_i, $D(m_i)$, as follows:

$$D(m_i) = \frac{\alpha * |A(m_i)| + \beta * |O(m_i)| + |U(m_i)|}{\alpha + \beta + 1}. \tag{4}$$

Suppose $D_{max}(M)$ is the maximum $D(m_j)$, where $m_j \in M$, the elasticity of m_i, $E(m_i)$, is calculated as in the following:

$$E(m_i) = \frac{D(m_i)}{D_{max}(M)}. \tag{5}$$

User-Movie Elasticity. We propose the user-movie elasticity $E(u_i, m_j)$, to indicate the possibility that u_i accepts m_j considering only the elasticity of u_i and m_j (not considering the relevance between them). It is calculated with $E(u_i)$ and $E(m_j)$, as follows:

$$E(u_i, m_j) = \frac{E(u_i) + \delta * E(m_j)}{1 + \delta}, \tag{6}$$

where δ denotes the contribution of $E(m_j)$.

4.2 Building Relevance Network

In order to make personalized recommendations, it is essential to obtain relatively accurate relationships among users and movies. We adopt two steps, building and updating the relevance network, to capture relevance among nodes (i.e., users and movies).

Building the Relevance Network. We propose new asymmetrical metrics on the basis of Jaccard index [10] instead of symmetrical ones common in prior works, so as to reveal relationships among users and movies in a more proper manner. We illustrate our idea with 4(a), where u_j is related to movies in genres of action, romance, and comedy, and u_i is related to all genres in the figure. Then, it's reasonable to recommend u_j's relevant movies to u_i, but the opposite may generate unrelated recommendations for u_j. It indicates the necessity of asymmetric measures. For associations between movies, let $U(m_i)$ denote users related to m_i, researchers usually

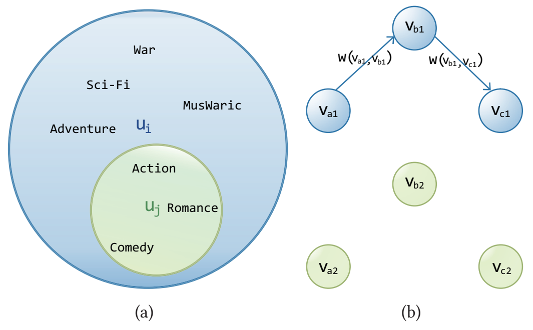


Figure 4: Examples in building the relevance network. (a) shows the necessity of asymmetrical measures, and (b) indicates the necessity of JohnsonMax.

utilize $\frac{U(m_i) \cap U(m_j)}{U(m_i) \cup U(m_j)}$ to measure both $R(m_i, m_j)$ and $R(m_j, m_i)$, neglecting the difference on sizes of $U(m_i)$ and $U(m_j)$.

In our work, we take asymmetrical metrics to measure the relevance between users and movies. We regard $R(u_i, u_j)$ as the influence from u_i to u_j, and $R(m_i, m_j)$ is the possibility of m_j being liked by users who are fond of m_i. We regard $R(u_i, m_j)$ as the degree of preference of u_i for m_j (centered on u_i), which is bound up with the u_i's average rating, $Avg(u_i)$; we view $R(m_i, u_j)$ as the importance of u_j to m_i (centered on m_i), related to the average of ratings received by m_i, $Avg(m_i)$. Suppose $M(u_i)$ is a movie set rated by u_i, $U(m_i)$ represents a user set related to m_i, $S(u_i, m_j)$ is the score u_i rates on m_j, we calculate the relevance factor from node v_i to v_j as follows:

$$R(u_i, u_j) = \frac{M(u_i) \cap M(u_j)}{M(u_i)}, \tag{7}$$

$$R(m_i, m_j) = \frac{U(m_i) \cap U(m_j)}{U(m_i)}, \tag{8}$$

$$R(u_i, m_j) = \frac{S(u_i, m_j)}{Avg(u_i)}, \tag{9}$$

$$R(m_i, u_j) = \frac{S(u_j, m_i)}{Avg(m_i)}. \tag{10}$$

Updating the Relevance Network. We build the original relevance network with users and movies as nodes (e.g., v_i, v_j), and relevance factors as the weights of edges, $w(v_i, v_j) = R(v_i, v_j)$. Next, we propose the JohnsonMax algorithm (see Algorithm 1) to update the relevance network. The basic idea is to capture relevance more accurately and identify weak ties. In the following, we will describe the necessity of updating the relevance network, and the details of the proposed JohnsonMax algorithm.

We provide an example to illustrate the necessity of updating using Figure 4(b). There are only two associations in the figure: $w(v_{a1}, v_{b1})$ from node v_{a1} to v_{b1}, and $w(v_{b1}, v_{c1})$ from v_{b1} to v_{c1}. It would build an indirect association from v_{a1} to v_{c1}, based on the theory of influence propagation [11, 30]. But there is neither direct nor indirect relevance from v_{a2} to v_{c2}. Therefore, the relevance from v_{a1} to v_{c1} is stronger than that from v_{a2} to v_{c2}, which is not reflected in the original network (as shown in Figure 4(b), $w(v_{a1}, v_{c1}) = w(v_{a2}, v_{c2}) = 0$). To overcome this limitation, we

Algorithm 1 JohnsonMax to update the relevance network

```
Input: N, the original relevance network
Output: N', the updated relevance network with more weak ties
1: let $mat$ be the adjacency matrix of N
2: //update the weights of the i-th row
3: function DijkstraMax($mat, i$)
4:     $stack \leftarrow [1, 2, 3 ... |V|]$ //the stack of index
5:     $stack.pop(i)$
6:     while $stack$ is not empty do
7:         //get the max weight from i to each node in $stack$
8:         $max_w, max_w\_index \leftarrow max(mat(i, V(stack))])$
9:         if $max_w == 0$ then
10:            $break$
11:        $stack.pop(max_w\_index)$
12:        for $j$ in $stack$ do
13:            $w_{new} \leftarrow (1 - \Gamma) * max_w * mat(max_w\_index, j)$
14:            if $w_{new} > mat(i, j)$ then
15:                $mat(i, j) \leftarrow w_{new}$
16: for $i$ in $|V|$ do //V, the node set of user-movie graph
17:     $mat(i) \leftarrow DijkstraMax(mat, i)$
```

propose to update the relevance from node v_i to v_j, $w(v_i, v_j)$, using the potential relevance $w'(v_i, v_j)$ generated by the influence propagation theory:

$$w'(v_i, v_j) = (1 - \Gamma) * w(v_i, v_p) * w(v_p, v_j), \tag{11}$$

where Γ is a loss factor, $\Gamma \in [0, 1]$, and $v_p \in \{V - v_i - v_j\}$, an intermediate node from v_i to v_j.

$$w(v_i, v_j) = \begin{cases} w'(v_i, v_j) & w'(v_i, v_j) > w(v_i, v_j) \\ w(v_i, v_j) & otherwise \end{cases}. \tag{12}$$

To more accurately capture the potential relevance and identify all possible weak ties, we need to find all the pairs with the strongest relevance by means of Equation 11 and 12. To be specific, in order to find the maximum $w(v_i, v_j)$, we calculate $w'(v_i, v_j)$ for each intermediate node in the relevance network. It is similar to finding the shortest path among all paths existing in the graph. We exploit Johnson [12] algorithm, which consists of multiple Dijkstra algorithms. It's particularly efficient for sparse graphs, and is easy to parallelize. Based on Johnson algorithm, we put forward the JohnsonMax algorithm to update the relevance network, and the details are shown in Algorithm 1. Lines 3 to 15 (Function DijkstraMax) update each row of adjacency matrix mat, which can be executed by multiple parallel processes. Note that lines 14 and 15 capture the potential associations between nodes, mitigating the data sparsity and discovering weak ties. Lines 16 and 17 update the whole relevance network.

The time complexity of the proposed JohnsonMax algorithm is $O(|V||R| + |V|^2 log|R|)$, which is efficient in sparse graphs such as the relevance network in our work.

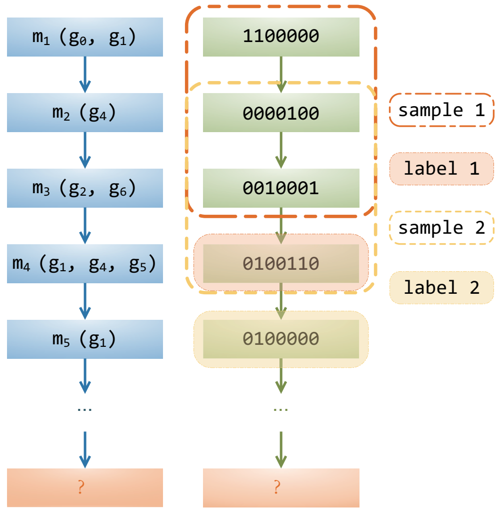


Figure 5: The pre-process for the inputs of GRU.

4.3 Predicting Genres

In serendipity-oriented recommendation systems, there is usually a risk of recommending unrelated movies accompanied with broadening user horizons. To address this issue, we propose to predict genres in line with users' short-term preferences with RNN.

It's necessary to make a pre-process for transforming data from original ratings to the input of RNN. First of all, we group movies by users, sort them within group according to rating timestamps, and convert each item of sequences into a genre vector (see Figure 5). Then, we adopt sliding windows to split these Boolean vectors for generating inputs of RNN, where we select w neighbors as an input sequence (in Figure 5, $w = 3$), and the next vector as a label.

Here we introduce a RNN with Gated Recurrent Units (GRU) [5] to predict user preferences in genre:

$$z_t = \sigma(W_z x_t + U_z h_{t-1}), \tag{13}$$

$$r_t = \sigma(W_r x_t + U_r h_{t-1}), \tag{14}$$

$$\hat{h}_t = tanh(W x_t + U(r_t h_{t-1})), \tag{15}$$

$$h_t = (1 - z_t) h_{t-1} + z_t \hat{h}_t, \tag{16}$$

where GRU leverage the update gate z_t and the reset gate r_t to capture long-term dependencies on user behaviors; the inputs are genre vectors, the output (the last hidden state, h_{end}) is a genre vector representing users' short-term preferences. We apply GRU instead of LSTM into genre prediction, because GRU reach the same performance with LSTM at a smaller time cost.

We train GRU on L samples, each of which is a g-dimension genre vector. Our goal is to predict higher scores for the true genres and lower scores for the false ones. Hence, we adopt binary cross entropy [31] as the loss function:

$$Loss(\hat{Y}, Y) = \sum_{1}^{L} \sum_{1}^{g} -y * \log \hat{y} + (y - 1) * \log(1 - \hat{y}), \tag{17}$$

where $\hat{y}$, a g-dimension vector, is the output of GRU, each element of $\hat{y}$ denotes the possibility score of the corresponding genre. We implement two filters, topK method and threshold filtering, to transform the possibility scores to the predicted genre sets.

Table 4: Statistics of datasets.

Item	Statistic	
	ml-1m	ml-latest-small
# users	6040	610
# movies	3900	9742
# ratings	100209	100836
movie release time (year)	1919-2000	1902-2018
rating time (year)	2000-2003	1996-2018

4.4 Recommendation with Elastic Serendipity

In the above subsections we gain the elasticity of users and movies, build the relevance network, and predict potential movie genres. Based on them, we will generate final recommendations.

We first filter options, keeping only those movies in the genres that GRU predict. Then, we calculate user-movie elasticity through Equation 6 and the relevance by Equation 9. Next, we combine elasticity and relevance to gain the elastic relevance between u_i and m_j, $RE(u_i, m_j)$, as follows:

$$RE(u_i, m_j) = \frac{R(u_i, m_j) + \lambda * E(u_i, m_j)}{1 + \lambda}, \tag{18}$$

where λ denotes the contribution of the elasticity $E(u_i, m_j)$.

For broadening user horizons, we try to recommend movies relatively different from their history. In addition, we also expect to decrease unacceptable recommendations, so as to increase user satisfaction. Hence, movies with median elastic relevance factors are suitable ones for recommendation, while the smaller are unacceptable for users and the bigger may cause over-specialization. We propose a new concept of the recommendation factor, $R\_factor(u_i, m_j)$,

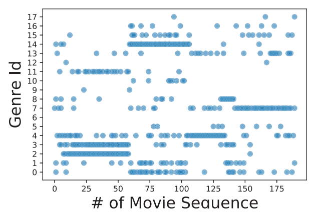
$$R\_factor(u_i, m_j) = |RE(u_i, m_j) - Avg(RE(U, M))|, \tag{19}$$

where $Avg(RE(U, M))$ is the average value of elastic relevance between all users and movies. Finally we recommend movies in the ascending order of the recommendation factor.

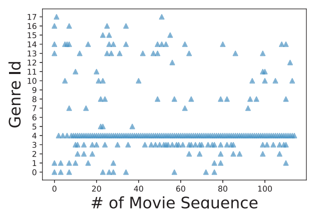
The intuition of Equation 19 is that when there is a strong elasticity between u_i and movies, it's desirable to recommend ones with weak relevance to u_i. There is a complementary relationship between the relevance and the elasticity in the serendipity-oriented recommendation [18]. Taking Figure 1(b) for instance, m_1 is further away from u_j (indicating weaker relevance), than the other two recommendations, m_2 and m_4. However, the elasticity of m_1 is bigger, indicating m_1 is easier to accept for users (even m_1 is somewhat unfamiliar to them). Thus, m_1 also is a serendipitous recommendation for u_j.

5 EXPERIMENTS

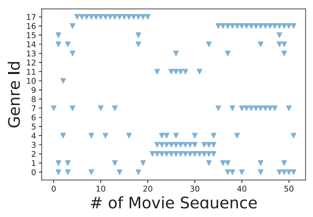
We evaluate our method based on two large-scale real-world datasets, MovieLens-1m (ml-1m) [8] and MovieLens-latest-small (ml-latest-small) [8], where we take the top 80% ratings as the training set and the rest as the testing set in chronological order. The statistics of datasets are shown in Table 4. Our experiments focus on the following issues:



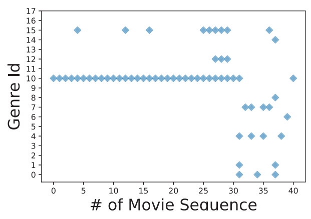
(a)



(b)



(c)



(d)

Figure 6: The preference distributions in movie genres of user samples.

- **Problem 1.** In this paper, we measure the serendipity with *content difference* and genre accuracy. Then, is it necessary to keep accuracy in genre, and how can we achieve it?
- **Problem 2.** We propose the JohnsonMax algorithm to update the relevance network based on the influence propagation theory. Then, what effect has been achieved by JohnsonMax in mitigating data sparsity and building weak ties?
- **Problem 3.** How does the proposed approach perform compared with other methods?

5.1 Verification on Problem 1

For **Problem 1**: whether it is necessary to keep accuracy in genre, and how to achieve it, we conduct the following experiments to demonstrate the necessity of genre accuracy, and adopt GRU to accurately predict user short-term preferences in movie genres.

5.1.1 Checking the necessity of genre accuracy. We visualize the distribution of movie genres for several individuals in chronological order. In order to make the results of visualization more reliable, we select four representative types of users to analyze: (1) users with a large number of related movies and wide interests in genre; (2) users who rate lots of movies in relatively single genres; (3) users related to a limited number of movies but with wide interests in genre and (4) users with few related movies in limited genre. We choose a representative user for each type, as shown in Figure 6 (a) (b) (c) (d) respectively.

We gain two main findings: (1) *local consistency.* Although it's almost impossible to capture overall preferences of users, we find it easy to predict genres of users' interest in the given context (at a fixed value of the X axis). (2) *global dynamics.* The dynamics of genre preferences widely exist in all users, not depending on the number of relevant movies and the scope of users' interest. Due to *local consistency* in users' preference, it's essential to guarantee the accuracy in genre to minimize unrelated recommendations. Considering *global dynamics*, the sequence model (e.g. GRU, LSTM) is a good choice to identify the movie genre that users may prefer.

5.1.2 Checking the effects of parameters. To improve the performance of genre prediction with GRU, we vary the length of inputs and the type of filters to check their effects. We adopt *micro_F*1 [33], *micro_precision* [33], and hamming distance [33] as evaluation metrics. The results are shown in Figure 7 and Figure 8, respectively.

For the length of sequences, we find that it's hardly possible to train a good model for predicting genres based on very short sequences (e.g., the length is within 10). This is because user preferences have a long-term dependence on the past behaviors. However, the performance wouldn't continue to be enhanced as the length of the sequence increases, due to the local consistency of user preferences. To make a trade-off, we use 20 as the length of input sequences, which is optimal with a comprehensive consideration of all metrics (i.e., *micro_F*1, *micro_precision*, and hamming distance).

We check the effects of filters, topK method and threshold filtering, in Figure 8. It indicates that the performance of genre prediction with threshold filtering fluctuates greatly, while topK is relatively stable. Possible reason for the fluctuation may be that the amount of positive predictions varies greatly in the prediction data; however, it is relatively stable in the label data. The fluctuation degree of threshold filtering performance is much stronger than that of topK, although the threshold filtering (t=0.3) has a slightly better performance than topK (k=2). Thus, we adopt topK method as the filter and use k=2 in the following experiments.

5.2 Verification on Problem 2

In this section, we verify **Problem 2**: what effect has been achieved by JohnsonMax in mitigating data sparsity and building weak ties.

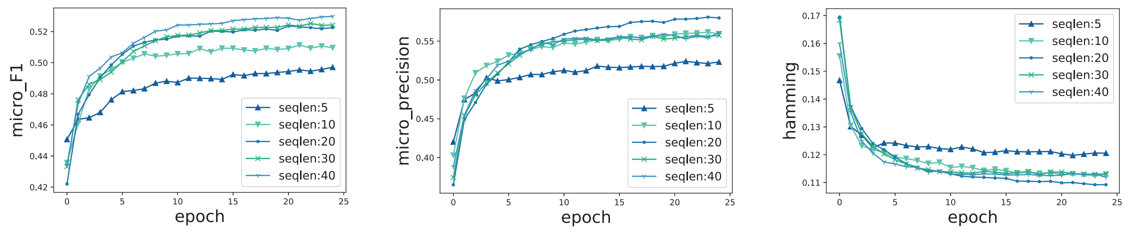
We visualize the distribution of weights originally existing in relevance network and updated by JohnsonMax in Figure 9. We have two main findings: (1) the JohnsonMax algorithm increases graph density[4] from 0.47 to 0.74, indicating its effectiveness in mitigating data sparsity; (2) the weights updated by JohnsonMax are distributed in (0, 0.3) and concentrated on (0, 0.1), indicating that it captures weak ties in the relevance network. Therefore, it meets the requirements of digging out movies related but not limited to users' histories in serendipity-oriented recommendation. Moreover, it is simple to implement, without the aid of auxiliary information.

5.3 Verification on Problem 3

We compare **HAES** model with some other methods on *content difference* and *genre accuracy* to verify **Problem 3**.

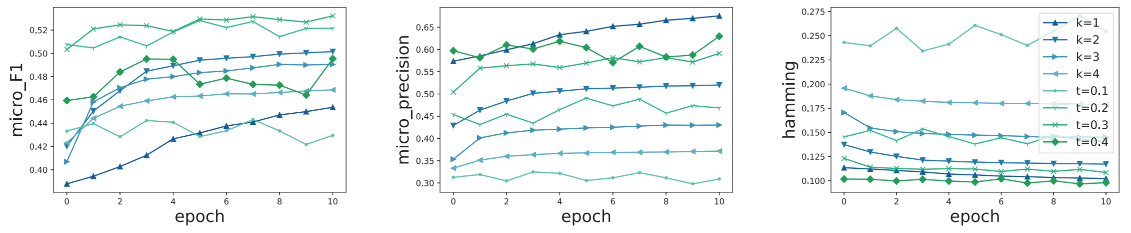
5.3.1 Baselines. Since there is no agreement on the definition of serendipity, researchers propose different algorithms for various purposes in line their definitions, which restricts the comparison among different serendipity-oriented methods. In this subsection, we adopt some widely used methods as benchmarks:

[4]$graph\ density = \frac{|R|}{|V|*(|V|-1)}$



(a) micro_F1 (b) micro_precision (c) hamming distance

Figure 7: Performance on genres prediction of sequences with different lengths.



(a) micro_F1 (b) micro_precision (c) hamming distance

Figure 8: Performance on genres prediction of different filters (topK and threshold). (a) and (b) use the same legend as (c), where "k" denotes the k-value of topK and "t" is the threshold.

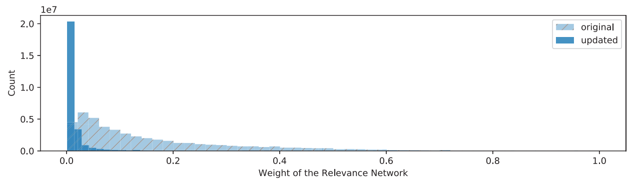


Figure 9: The effectiveness of updating the relevance network with JohnsonMax.

ACC. Accuracy-based approach is one of the most widely used personalized recommendation algorithms. It recommends movies consistent with users past behaviors to improve the accuracy.

NOV. Novelty-based method recommends movies based on their release time, aiming at recommending users the latest movies.

POP. Popularity-based recommendation systems generate results according to the prevalence of movies, where we adopt the number of ratings movies have received to measure their popularity.

RAND. Random-based algorithm randomly selects movies among the candidates that the target user hasn’t rated.

HAES-NE. HAES-No-Elasticity, **HAES** without regard to elasticity, recommends movies with a fixed level of serendipity, whose recommendation factor only depends on the relevance as follows:

$$R\_factor(u_i, m_j) = |R(u_i.m_j) - Avg(R)|. \tag{20}$$

HAES-NG. HAES-No-Genre is **HAES** without the component for genre prediction, generating recommendations in all genres.

5.3.2 Evaluation Metrics. From the definition of serendipity above, *genre accuracy* and *content difference*, we select four metrics: *micro_F1* and average hamming distance (denoted as *avg_hamming*) as metrics for the *genre accuracy*; difference from histories(denoted as *difference*) and diversity in recommendation lists(denoted as *diversity*) as metrics for the *content difference*.

micro_F1. We adopt *micro_F*1, a metric in multi-label prediction [33], to measure the recall, *micro_r*, and precision, *micro_p*, of recommendations comprehensively. Suppose G_i is a genre vector, $sum(G_i)$ is the sum of all elements in G_i, G is a label set of genre vectors, $\hat{G}$ is the predicted one, we calculate *micro_F*1 on a samples:

$$micro\_r(\hat{G}, G) = \frac{1}{a}\sum_{i=1}^{a}\frac{sum(\hat{G}_i * G_i)}{sum(G_i)}, \tag{21}$$

$$micro\_p(\hat{G}, G) = \frac{1}{a}\sum_{i=1}^{a}\frac{sum(\hat{G}_i * G_i)}{sum(\hat{G}_i)}, \tag{22}$$

$$micro\_F1(\hat{G}, G) = \frac{2 * micro\_r(\hat{G}, G) * micro\_p(\hat{G}, G)}{micro\_r(\hat{G}, G) + micro\_p(\hat{G}, G)}. \tag{23}$$

avg_hamming. avg_hamming considers not only positive samples but also negative ones. Suppose G_i is a g-dimension genre vector, we calculate *avg_hamming* on a samples as follows [33]:

$$avg\_hamming(\hat{G}, G) = \frac{1}{a * g}\sum_{i=1}^{a} sum(\hat{G}_i \oplus G_i). \tag{24}$$

difference. Based on the associations between users and movies, we calculate the difference, $difference(u_t, M_t)$, between the target

Table 5: Experimental results of different methods on content difference (MovieLens-1m). "HAES-NE" refers to our approach without the elasticity component, and "HAES-NG" is the one without genres prediction.

method	dif@5	dif@10	dif@15	div@5	div@10	div@15
ACC	0.2255	0.2511	0.2681	0.7037	0.714	0.71
NOV	0.7078	0.7289	0.7035	0.5995	0.6872	0.649
POP	0.5571	0.5571	0.5562	0.2118	0.2472	0. 2736
RAND	0.7428	0.7435	0.7438	0.7229	0.7227	0.7229
HAES-NE	0.5108	0.5186	0.5243	0.5209	0.5294	0.535
HAES-NG	0.7973	0.8062	0.8098	**0.7624**	**0.7479**	**0.7408**
HAES	**0.8134**	**0.82**	**0.8242**	0.683	0.6701	0.6606

Table 6: Results on genre accuracy (MovieLens-1m).

method	F1@5	F1@10	F1@15	h@5	h@10	h@15
ACC	0.2059	0.2036	0.2006	0.1577	0.1601	0.162
NOV	0.1964	0.1913	0.2127	0.1757	0.1684	0.1677
POP	0.2065	0.2034	0.2011	0.2351	0.2221	0.2179
RAND	0.2016	0.2008	0.2	0.1663	0.1667	0.1668
HAES-NE	0.3072	0.3035	0.3019	0.1604	0.1609	0.1614
HAES-NG	0.1867	0.1884	0.1881	0.1628	0.1644	0.1656
HAES	**0.3149**	**0.3119**	**0.3122**	**0.1482**	**0.1506**	**0.1514**

Table 7: Results on content difference (MovieLens-latest-small).

method	dif@5	dif@10	dif@15	div@5	div@10	div@15
ACC	0.2917	0.3036	0.3131	0.2346	0.2414	0.2464
NOV	0.5803	**0.7353**	0.5916	**0.722**	0.4531	**0.7307**
POP	0.1951	0.3131	0.2079	0.3045	0.1781	0.31
RAND	**0.6836**	0.7028	**0.6826**	0.7005	**0.6839**	0.7035
HAES-NE	0.4831	0.5163	0.4917	0.5064	0.4644	0.512
HAES-NG	0.6706	0.6652	0.6665	0.6654	0.6774	0.6653
HAES	0.642	0.6651	0.6537	0.6653	0.6375	0.6652

user, u_t, and recommendations, M_t ($|M_t| = a$), as follows [6]:

$$difference(u_t, M_t) = 1 - \frac{1}{a}\sum_{j=1}^{a} R(u_t, m_j). \tag{25}$$

diversity. We calculate the diversity of recommendations, $diversity(M_t)$ ($|M_t| = a$), based on the relevance between movies of M_t [6]:

$$diversity(M_t) = 1 - \frac{2}{a(a-1)}\sum_{i=1}^{a}\sum_{j=1}^{i-1} R(m_i, m_j). \tag{26}$$

5.3.3 Results. We compare **HAES** with baselines on *micro_F*1(F1), *avg_hamming*(h), $difference$(dif), and *diversity*(div). The results on Movielens-1m are shown in Table 5 and Table 6, and those on MovieLens-latest-small are shown in Table 7 and Table 8.

Comparison on content difference. **HAES** achieves the best performance on Movielens-1m (see Table 5), e.g., it even increases 2.3 times on *difference* compared with **ACC**. However, random-based (i.e., **RAND**) and novelty-based (i.e., **NOV**) method perform better on MovieLens-latest-small (see Table 7). The reason may be that the timespan of ratings on it is twenty-two years, which is too long to build weak ties for **HAES**. But the long span makes it easy to generate recommendations without relevance. To verify the above impact of the long timespan, we analyze all rating timestamps on MovieLens-latest-small in Figure 10. It indicates that most users provide ratings only within a limit timespan (the average span is 0.64 year), i.e., the longer the span is, the less the interaction of users is. Another finding is that **HAES-NE**, **HAES** without the elasticity component, performs poorer than **HAES** on both datasets, indicating the importance of the elasticity in broadening user horizons.

Table 8: Results on genre accuracy (MovieLens-latest-small).

method	F1@5	F1@10	F1@15	h@5	h@10	h@15
ACC	0.2198	0.223	0.2218	0.229	0.23	0.2316
NOV	0.2125	0.2134	0.2161	0.2578	0.2375	0.2328
POP	0.2343	0.2245	0.2199	0.2406	0.2404	0.2383
RAND	0.2223	0.2244	0.2255	0.2071	0.2072	0.2071
HAES-NE	0.2931	0.2899	0.2913	0.203	0.2046	0.2038
HAES-NG	0.2346	0.2356	0.2312	0.2058	0.2055	0.2063
HAES	**0.2963**	**0.2987**	**0.2967**	**0.1987**	**0.1971**	**0.1986**

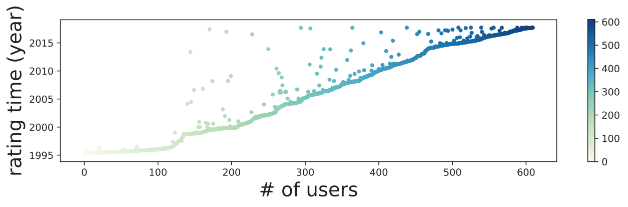


Figure 10: The visualization of rating timestamps (we choose the earliest and the latest rating time for each user in MovieLens-latest-small), to show the impact on the interaction among users of the long rating timespan.

Comparison on genre accuracy. **HAES** and **HAES-NE**, our approaches with the genre prediction component, achieve significant improvements on *genre accuracy* (see Table 6 and Table 8). Particularly, **HAES** improves the *micro_F1* by 53.93% compared with **ACC** on MovieLens-1m. It has a close association with the fact that we capture user short-term preferences with GRU, while the others either acquire user preferences as a whole or neglect user interests. In addition, we find that methods with consideration of global preferences (e.g., **ACC**) even are inferior to the random recommendation (**RAND**) on Movielens-latest-small (see Table 8). The reason may be that user global preferences can't represent their short-term demands, while they also limit the range of recommendations.

5.4 Summary of Experiments

In summary, we have the following findings in the experiments. (1) The threshold filtering method would cause the fluctuation of prediction performance, because the amount of the positive predictions varies greatly. (2) On MovieLens-latest-small, random-based (**RAND**) and novelty-based (**NOV**) approaches have a better performance on *genre accuracy*, than the one based on accuracy (**ACC**). One possible reason is that global preferences of users can't represent their short-term demands, while they also limit the range of potential recommendations.(3) Overall, our approach (**HAES**) achieves significant improvements, e.g., it improves the *micro_F1* by 53.93% and increases 2.3 times on *difference* compared with the accuracy-oriented method (**ACC**) on MovieLens-1m.

6 CONCLUSIONS AND FUTURE WORK

In this paper, we propose a hybrid approach for recommending movies with elastic serendipity, based on a more objective definition of serendipity. We propose and quantify the elasticity in recommendation system, for meeting users' varying demands for serendipity and accuracy. We put forward asymmetric measures to more accurately capture associations among users and movies, and then present JohnsonMax to mitigate the data sparsity and build weak ties. We introduce GRU to acquire users' short-term preferences in movie genres, which reduces unrelated movies and preserves the quality of recommendation. In addition, we believe HAES is not limited to movie recommendation but suitable for all systems with high aggregation and low coupling, and we will apply HAES to a wider range of scenarios (e.g., recommendation on book, music) in the future. We also would like to build an interactive serendipity-oriented recommendation system, capturing users' real-time implicit feedback with reinforcement learning.

ACKNOWLEDGMENTS

This research was supported by NSFC grant 61632009 and in part by NSF grants CNS 1824440, CNS 1828363, CNS 1757533, CNS 1618398, CNS 1651947, and CNS 1564128.

REFERENCES

[1] Shlomo Berkovsky, Tsvi Kuflik, and Francesco Ricci. 2007. Distributed collaborative filtering with domain specialization. In *RecSys 2007*. 33–40. https://doi.org/10.1145/1297231.1297238

[2] Rahul Bhagat, Srevatsan Muralidharan, Alex Lobzhanidze, and Shankar Vishwanath. 2018. Buy It Again: Modeling Repeat Purchase Recommendations. In *KDD 2018*. 62–70. https://doi.org/10.1145/3219819.3219891

[3] Òscar Celma and Paul Lamere. 2011. Music recommendation and discovery revisited. In *RecSys 2011*. 7–8. https://doi.org/10.1145/2043932.2043936

[4] Jin Yao Chin, Kaiqi Zhao, Shafiq R. Joty, and Gao Cong. 2018. ANR: Aspect-based Neural Recommender. In *CIKM 2018*. 147–156. https://doi.org/10.1145/3269206.3271810

[5] Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *CoRR* abs/1412.3555 (2014). arXiv:1412.3555 http://arxiv.org/abs/1412.3555

[6] Mouzhi Ge, Carla Delgado-Battenfeld, and Dietmar Jannach. 2010. Beyond accuracy: evaluating recommender systems by coverage and serendipity. In *RecSys 2010*. ACM, 257–260.

[7] Mihajlo Grbovic and Haibin Cheng. 2018. Real-time Personalization using Embeddings for Search Ranking at Airbnb. In *KDD 2018*. 311–320. https://doi.org/10.1145/3219819.3219885

[8] F. Maxwell Harper and Joseph A. Konstan. 2016. The MovieLens Datasets: History and Context. *TiiS* 5, 4 (2016), 19:1–19:19. https://doi.org/10.1145/2827872

[9] Jonathan L Herlocker, Joseph A Konstan, Loren G Terveen, and John T Riedl. 2004. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.* 22, 1 (2004), 5–53.

[10] Paul Jaccard. 1912. The distribution of the flora in the alpine zone. 1. *New phytologist* 11, 2 (1912), 37–50.

[11] Wenjun Jiang, Jie Wu, Guojun Wang, and Huanyang Zheng. 2014. FluidRating: A time-evolving rating scheme in trust-based recommendation systems using fluid dynamics. In *INFOCOM 2014*. 1707–1715. https://doi.org/10.1109/INFOCOM.2014.6848108

[12] Donald B. Johnson. 1977. Efficient Algorithms for Shortest Paths in Sparse Networks. *J. ACM* 24, 1 (1977), 1–13. https://doi.org/10.1145/321992.321993

[13] Aleksandra Karpus, Iacopo Vagliano, and Krzysztof Goczyla. 2017. Serendipitous Recommendations Through Ontology-Based Contextual Pre-filtering. In *BDAS 2017 (Communications in Computer and Information Science)*, Vol. 716. 246–259. https://doi.org/10.1007/978-3-319-58274-0_21

[14] 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*. ACM, 1341–1350.

[15] D. Kotkov, J. Veijalainen, and S. Wang. 2016. Challenges of serendipity in recommender systems. In *WEBIST 2016*, Vol. 2. 251–256.

[16] Denis Kotkov, Jari Veijalainen, and Shuaiqiang Wang. 2018. How does serendipity affect diversity in recommender systems? A serendipity-oriented greedy algorithm. *Computing* (2018), 1–19.

[17] Denis Kotkov, Shuaiqiang Wang, and Jari Veijalainen. 2016. A survey of serendipity in recommender systems. *Knowledge-Based Systems* 111 (2016), 180–192. https://doi.org/10.1016/j.knosys.2016.08.014

[18] Valentina Maccatrozzo, Manon Terstall, Lora Aroyo, and Guus Schreiber. 2017. SIRUP: Serendipity In Recommendations via User Perceptions. In *IUI 2017*, George A. Papadopoulos, Tsvi Kuflik, Fang Chen, Carlos Duarte, and Wai-Tat Fu (Eds.). ACM, 35–44. https://doi.org/10.1145/3025171.3025185

[19] Jarana Manotumruksa, Craig Macdonald, and Iadh Ounis. 2017. A Deep Recurrent Collaborative Filtering Framework for Venue Recommendation. In *CIKM 2017*. 1429–1438. https://doi.org/10.1145/3132847.3133036

[20] Sean M. McNee, John Riedl, and Joseph A. Konstan. 2006. Being accurate is not enough: how accuracy metrics have hurt recommender systems. In *CHI 2006*. ACM, 1097–1101. https://doi.org/10.1145/1125451.1125659

[21] Tomas Mikolov, Martin Karafiát, Lukás Burget, Jan Cernocký, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *INTERSPEECH 2010*. 1045–1048. http://www.isca-speech.org/archive/interspeech_2010/i10_1045.html

[22] Tien T. Nguyen, F. Maxwell Harper, Loren Terveen, and Joseph A. Konstan. 2018. User Personality and User Satisfaction with Recommender Systems. 20, 6 (2018), 1173–1189. https://doi.org/10.1007/s10796-017-9782-y

[23] Xi Niu. 2018. An Adaptive Recommender System for Computational Serendipity. In *ICTIR 2018*. ACM Press, 215–218. https://doi.org/10.1145/3234944.3234974

[24] Gaurav Pandey, Denis Kotkov, and Alexander Semenov. 2018. Recommending Serendipitous Items Using Transfer Learning. In *CIKM 2018 (CIKM '18)*. ACM, 1771–1774. https://doi.org/10.1145/3269206.3269268

[25] Wenjie Pei, Jie Yang, Zhu Sun, Jie Zhang, Alessandro Bozzon, and David M. J. Tax. 2017. Interacting Attention-gated Recurrent Networks for Recommendation. In *CIKM 2017*. 1459–1468. https://doi.org/10.1145/3132847.3133005

[26] P. Resnick and H.R. Varian. 1997. Recommender Systems. *Commun. ACM* 40, 3 (1997), 56–58. https://doi.org/10.1145/245108.245121

[27] Shaoyun Shi, Min Zhang, Yiqun Liu, and Shaoping Ma. 2018. Attention-based Adaptive Model to Unify Warm and Cold Starts Recommendation. In *CIKM 2018*. 127–136. https://doi.org/10.1145/3269206.3271710

[28] Andrei Martins Silva, Fernando Henrique da Silva Costa, Alexandra Katiuska Ramos Diaz, and Sarajane Marques Peres. 2018. Exploring Coclustering for Serendipity Improvement in Content-Based Recommendation. In *IDEAL 2018*. 317–327. https://doi.org/10.1007/978-3-030-03493-1_34

[29] Horace Walpole. 1937. *The Yale edition of Horace Walpole's correspondence*. Vol. 48. Yale University Press.

[30] Guojun Wang, Wenjun Jiang, Jie Wu, and Zhengli Xiong. 2014. Fine-Grained Feature-Based Social Influence Evaluation in Online Social Networks. *IEEE Trans. Parallel Distrib. Syst.* 25, 9 (2014), 2286–2296. https://doi.org/10.1109/TPDS.2013.135

[31] Shaowei Wang, Jian Wang, Xiaoyong Ji, and Yuhao Wang. 2009. Binary Sequences with Good Aperiodic Autocorrelations Using Cross-Entropy Method. In *ICIC 2009*. 381–385. https://doi.org/10.1007/978-3-642-04020-7_40

[32] Hongyi Wen, Longqi Yang, Michael Sobolev, and Deborah Estrin. 2018. Exploring recommendations under user-controlled data filtering. In *RecSys 2018*. 72–76. https://doi.org/10.1145/3240323.3240399

[33] Xi-Zhu Wu and Zhi-Hua Zhou. 2017. A Unified View of Multi-Label Performance Measures. In *ICML 2017*. 3780–3788. http://proceedings.mlr.press/v70/wu17a.html

[34] Yongjian Yang, Yuanbo Xu, En Wang, Jiayu Han, and Zhiwen Yu. 2018. Improving existing collaborative filtering recommendations via serendipity-based algorithm. *IEEE Transactions on Multimedia* 20, 7 (2018), 1888–1900.

[35] Yin Zhang, Haokai Lu, Wei Niu, and James Caverlee. 2018. Quality-aware neural complementary item recommendation. In *RecSys 2018*. 77–85. https://doi.org/10.1145/3240323.3240368

[36] Yuan Cao Zhang, Diarmuid Ó Séaghdha, Daniele Quercia, and Tamas Jambor. 2012. Auralist: introducing serendipity into music recommendation. In *WSDM 2012*. 13–22. https://doi.org/10.1145/2124295.2124300

[37] Xiangyu Zhao, Long Xia, Liang Zhang, Zhuoye Ding, Dawei Yin, and Jiliang Tang. 2018. Deep reinforcement learning for page-wise recommendations. In *RecSys 2018*. 95–103. https://doi.org/10.1145/3240323.3240374

[38] Junxing Zhu, Jiawei Zhang, Lifang He, Quanyuan Wu, Bin Zhou, Chenwei Zhang, and Philip S. Yu. 2017. Broad Learning based Multi-Source Collaborative Recommendation. In *CIKM 2017*. 1409–1418. https://doi.org/10.1145/3132847.3132976