Identifying Users behind Shared Accounts in Online Streaming Services

Jyun-Yu Jiang[†], Cheng-Te Li[‡], Yian Chen[*] and Wei Wang[†]

[†]Department of Computer Science, University of California, Los Angeles
[‡]Institute of Data Science, Department of Statistics, National Cheng Kung University
[*]Research Center, KKBOX Inc.
jyunyu@cs.ucla.edu,chengte@mail.ncku.edu.tw,annchen@kkbox.com,weiwang@cs.ucla.edu

ABSTRACT

Online streaming services are prevalent. Major service providers, such as Netflix (for movies) and Spotify (for music), usually have a large customer base. More often than not, users may share an account. This has attracted increasing attention recently, as account sharing not only compromises the service provider's financial interests but also impairs the performance of recommendation systems and consequently the quality of service provided to the users.

To address this issue, this paper focuses on the problem of user identification in shared accounts. Our goal is three-fold: (1) Given an account, along with its historical session logs, we identify a set of users who share such account; (2) Given a new session issued by an account, we find the corresponding user among the identified users of such account; (3) We aim to boost the performance of item recommendation by user identification. While the mapping between users and accounts is unknown, we propose an unsupervised learning-based framework, *Session-based Heterogeneous graph Embedding for User Identification* (SHE-UI), to differentiate and model the preferences of users in an account, and to group sessions by these users. In SHE-UI, a heterogeneous graph is constructed to represent items such as songs and their available metadata such as artists, genres, and albums. An item-based session embedding technique is proposed using a normalized random walk in the heterogeneous graph. Our experiments conducted on two large-scale music streaming datasets, Last.fm and KKBOX, show that SHE-UI not only accurately identifies users, but also significantly improves the performance of item recommendation over the state-of-the-art methods.

CCS CONCEPTS

• **Information systems** → *Personalization*; *Recommender systems*; *Data mining*;

KEYWORDS

User identification, shared accounts, heterogeneous graph embedding, user session clustering, recommender systems

SIGIR '18, July 8–12, 2018, Ann Arbor, MI, USA

ACM ISBN 978-1-4503-5657-2/18/07…$15.00
https://doi.org/10.1145/3209978.3210054

ACM Reference Format:
Jyun-Yu Jiang[†], Cheng-Te Li[‡], Yian Chen[*] and Wei Wang[†]. 2018. Identifying Users behind Shared Accounts in Online Streaming Services. In *SIGIR '18: The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, July 8–12, 2018, Ann Arbor, MI, USA.* ACM, New York, NY, USA, Article 4, 10 pages. https://doi.org/10.1145/3209978.3210054

1 INTRODUCTION

Online streaming services, such as Netflix[1] and Spotify[2], have become popular and accumulated massive user bases. Premium users have the privileges to enjoy high-quality contents and real-time streaming events, but these services usually come at a fee. Because of the membership fee of the premium accounts, it is not rare that users share a premium account to split the cost between themselves. However, illegal sharing may compromise not only the service provider's financial interests but also the service quality in general. First, account sharing implies loss of potential customers who may bring additional revenue to the service provider. Second, current customer profiling and recommendation systems operate under the assumption that each account is used by a single user and hence cannot accurately model individual user preferences from a mixture of activities by multiple users. This may impair its ability to provide high quality recommendations to users. Consequently, unsatisfied users may decide to switch to other providers.

To detect account sharing and enhance the quality of recommender systems in the presence of account sharing, this paper aims to identify individual users behind shared accounts. The goal of our work is three-fold. First, given a list of registered accounts, along with the corresponding *session logs* that record the activities of the accounts, we aim to accurately identify the set of users behind each account based on its session activities from the set of users who are using this account. Then we can accordingly predict whether an account is shared by multiple users. Second, given a newly-coming session issued by a certain account, we aim to identify the corresponding user from the identified users of that account. Third, we will enhance the performance of item recommendation by integrating account-level and user-level item recommendation. The session log of an account contains lists of entries. Each entry records the item requested and the timestamp of such request. We organize the log of each account into a list of consecutive sessions. In addition, each item may be associated with several metadata attributes (e.g., a song may have genres, artists, albums, and published years).

[1]Netflix: https://www.netflix.com/
[2]Spotify: https://www.spotify.com/

It has been shown in the literature that modeling multi-user behaviors in shared accounts [1, 37, 38, 42, 44] and session-based recommendations [19, 36] successfully improve the performance of item recommendation. However, these studies do not attempt to identify individual users. To the best of our knowledge, Zhang et al. [43] is the first and only attempt to identify users in shared accounts by a specialized subspace clustering, which is employed as a baseline in our experimental studies. In addition, the information of metadata is not taken into account in their approach.

Since we do not know the accounts that may be shared by multiple users and the users that share the same account, we propose an unsupervised learning-based framework, *Session-based Heterogeneous graph Embedding for User Identification* (**SHE-UI**). The main idea is to model the preference of individual users via a novel technique of session embedding that learns a unique feature representation for each session. We first create a heterogeneous information network to represent the relationships among items and their meta information. Then, by applying a specialized random walk mechanism, the feature representation of each node can be derived using the skip-gram learning architecture [25, 27]. Subsequently the item-based session embedding is learned through the node embedding. For each account, the user identification problem is then mapped to the problem of session clustering. We develop a clustering algorithm based on Affinity Propagation [15] to simultaneously determine the number of clusters and group sessions into clusters. Each cluster represents the sessions issued by the same user, and the number of clusters represents the number of users sharing this account. For any incoming session of this account, we may find the potential user who issues the session by computing its representation (from the first few items in the session) in the space of session embedding and finding its nearest cluster. Last, to boost the performance of recommender systems, we propose a hybrid recommender, termed **AURec**, which combines conventional account-level and user-level (derived by SHE-UI) item recommendation.

We summarize the contributions of this work in the following.

- We propose to deal with two research tasks: (1) identifying users behind a shared account based on historical sessions and metadata of the items, and (2) given a new session initiated by a multi-user account, identifying which user issued this session. The former can benefit the service provider to detect multi-user accounts so that new pricing strategies can be established, while the latter boosts the performance of item recommendation. It is also worth mentioning that no prior knowledge about the mappings between users and accounts are given here.
- We develop an unsupervised framework SHE-UI that cannot only identify users in shared accounts, but also learn the preferences of individual users. Through a novel session embedding technique, SHE-UI effectively learns feature representations from a heterogeneous graph that represents the relationships between items and their metadata.
- Experiments conducted on two large-scale datasets of online streaming services, Last.fm and KKBOX, demonstrate that SHE-UI clearly outperforms existing item-based and embedding-based methods on both tasks of user identification. A study of parameter sensitivity also manifests the robustness of SHE-UI.
- Based on the identified users behind accounts, we devise a hybrid recommender AURec that combines account-level and user-level item recommendation. Experiments on KKBOX data show that AURec is able to significantly outperform the state-of-the-art account-level recommendation methods by 39% in terms of Precision@1.

Table 1: Comparison of relevant studies. Note that "User Identification" indicates whether users behind shared accounts are explicitly identified. The "Items" field shows the types of items that are considered, "metadata" indicates whether meta information about items is modeled, and "Recommend" shows whether the goal of a study is to improve the performance of recommendations.

	User Identification	Items	Metadata	Recommend
Wang et al. [38]		IPTV		√
Yang et al. [42]		IPTV	√	√
Verstrepen et. al. [37]		Movie		√
Zhao et al. [44]		Ticket		√
Zhang el al. [43]	√	Movie		√
Aharon et al. [1]		TV		√
Bajaj et al. [5]		TV		√
Our work	√	Music	√	√

The remainder of this paper is organized as follows. After presenting the relevant work in Section 2, we present the problem statement in Section 3. The proposed methodology is described in Section 4. We show the experimental results in Section 5, and conclude this work in Section 6.

2 RELATED WORK

2.1 Modeling User Preferences in Shared Accounts

Several studies have attempted to *model* user behaviors from session logs [1, 5, 37, 38, 42, 44]. They improve the performance of item recommendation according to the (latent) preferences of individual users. Diverse types of items have been investigated, including TV [1, 38, 42], movie [37, 43], and flight ticket [44]. The common approach to model user preferences is to de-convolute a high dimensional feature space that characterizes the relationships among accounts, items, and time [38, 42]. Techniques, such as subspace clustering [43], graph partition [38], collaborative filtering [37], topic model [44], and latent factor model with LDA [5], are used to obtain latent features so that the user preferences can be captured. Although the performance of the recommender systems can be improved, these studies assume each account is associated with one user and thus do not distinguish individual users sharing an account. We argue that identifying individual users can bring additional values, as it allows for recovering lost revenue, better targeted marketing, designing new service plans, among many other useful applications. In addition, a sequence of items from a user can be regarded as a Markov process, so modeling interleaved Markov processes [24] can be also treated as a related work. Table 1 summarizes the comparison between our work and previous studies.

2.2 User Identification

To the best of our knowledge, Zhang et al. [43] is the first and only attempt that can report whether an account is shared by multiple users and explicitly identify these users. They focus on item ratings

and show how conventional methods such as expectation maximization and principal component analysis can be used for user identification via a specialized subspace clustering. However, their models cannot incorporate metadata. The experiments in Section 5 will compare our SHE-UI with Zhang et al. [43] (referred to as SS in Table 3.)

Fraud detection can be treated as a special case of user identification. Previous work detects malicious users based on context information such as social network structures [2, 8] and unusual behavior patterns [21, 23]. However, all of them identify the only one user in an account and cannot deal with the situations with multiple users.

2.3 Session-based Item Recommendation

Some recent studies have investigated session-based recommendation [19, 36], i.e., tracking and modeling subsequent sessions of the same users (assuming we have the knowledge of user for each session) for item recommendation. Conventional approaches, including Markov Decision Process [31] and Matrix Factorization [20], are adapted for session-based recommendation. More recently, Recurrent Neural Networks (RNN) are also used for session-based recommendation. Although these studies have shown that session-based recommendation outperforms conventional recommender systems, they do not address the problem of user identification in shared accounts.

2.4 Personalized Search via Individualization

Aside from recommendation systems, it is also important to model individual users in search engines so that personalized search can be provided. Under the context of search engine, what users share are devices [41], rather than accounts. Therefore, White et al. [40, 41] and Singla et al. [33] have investigated how to improve user satisfaction of search engines based on identifiers of individual users behind shared devices. However, they only make efforts on personalizing relevance ranking with given user identifiers.

2.5 Learning Feature Representation of Users

Our work takes advantage of network embedding-based approach to learn the feature representation of users under the setting of graph structure so that individual users in shared accounts can be characterized. Therefore, we would like to briefly review some of recent studies. By simulating uniform random walks, Deepwalk [28] is the first work that performs feature learning for nodes in a graph, along with the application to multi-label classification. LINE [35] further considers Breadth First Search in the random walk, and apply the network embedding method for node classification and link prediction. More recently, node2vec [18] develops a generalized bias random walk mechanism that can parameterize Breadth First Search and Depth First Search, and its efficacy outperforms the previous two methods in both multi-label classification and link prediction. In addition to learning features on homogeneous network, some studies focus on heterogeneous network embedding. Chang et al. [10] applied deep neural networks to learn the representations of networks with both texts and images. Dong et al. [13] enhanced Deepwalk by conducting random walks based on a set of meta-paths. Chen and Sun [11] built a heterogeneous network to predict authors of anonymous papers with augmented meta-paths.

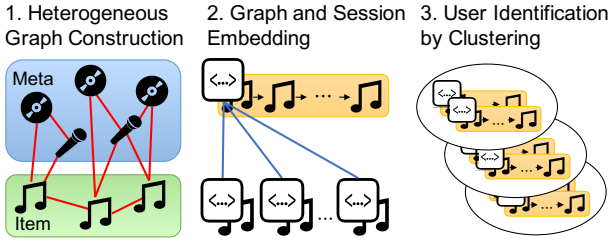


Figure 1: The framework overview of SHE-UI.

3 PROBLEM STATEMENT

In this section, we first formally define the problem of user identification in online streaming services. Let I be the set of items, e.g., songs and movies. For each item $i \in I$, it may have multiple attributes, e.g., artist(s) and genre(s) of a song, denoted by M_i, as its metadata. We denote the collection of all metadata as $M = \bigcup_{i \in I} M_i$. Here the metadata can be any discrete attribute that can describe items. Relationships may exist between attributes in metadata. For example, an album m_j may include a song performed by an artist m_k. These relationships can be denoted as $R = \{(m_j, m_k) \mid (m_j, m_k) \in M^2, m_j \neq m_k, m_j \text{ is related to } m_k\}$.

Let A be the set of accounts. For each account $a \in A$, the set of users of the account is denoted as $U(a)$, which is unknown in advance. In the following discussion, we refer to accounts with only one user as *single-user accounts* and the remaining ones as *multi-user accounts*. The activity log of each account $a \in A$ contains a sequence of sessions $S(a)$. Each session $s \in S(a)$ is a sequence of T_s items (successively requested by a user u_s without a long period of inactivity): $s = \langle i_1, i_2, \cdots, i_{T_s} \rangle \in I^{T_s}$. For multi-user accounts, we assume that each session may be issued by one user. Note that the actual user u_s of every session s is also unknown to the system. The two goals of this work are as follows:

(1) **User Identification in Past Sessions (UI-Past):** Given a set of accounts A and their corresponding sessions, for each account $a \in A$, the first goal is to group sessions $S(a)$ into K_a clusters (i.e., users), $C(a) = \{c_1^a, c_2^a, \cdots, c_{K_a}^a\}$ such that the sessions from the same user are grouped into the same cluster where $1 \leq K_a \leq |S(a)|$. K_a is also unknown and needs to be estimated from the data. In other words, we would like to estimate the ideal clusters $C^*(a)$ grouping sessions by their actual users.
(2) **User Identification for New Sessions (UI-New):** Given the identified users $C(a)$ of an account a, for any new incoming session $s \notin S(a)$ of account a, the next goal is to predict which user is the actual issuer of this session as early as possible. Based on the first few items in s, we want to identify the cluster c_k^a to which s belongs.

4 SESSION-BASED HETEROGENEOUS GRAPH EMBEDDING FOR USER IDENTIFICATION

In this section, we present the proposed framework, Session-based Heterogeneous graph Embedding for User Identification (SHE-UI).

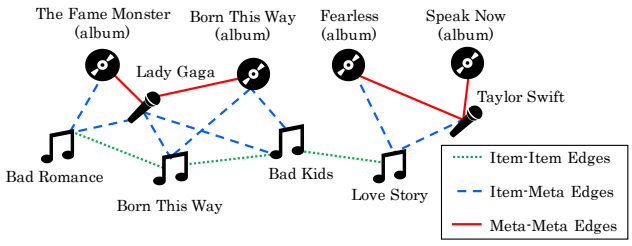


Figure 2: An example of heterogeneous graph construction with a session that begins with three songs by Lady Gaga and ends with a song by Taylor Swift.

4.1 Framework Overview

Figure 1 shows the framework of SHE-UI. A heterogeneous graph is constructed to represent the relations among items and metadata. We first compute node embeddings from which we generate session embeddings. Then an algorithm based on affinity propagation [15] is proposed to simultaneously determine the cluster number K_a for each account a and to group sessions $S(a)$ into clusters.

4.2 Node Embedding in Heterogeneous Graph

As the first stage of SHE-UI, we encode items and metadata into an undirected graph $G = (V, E)$ with heterogeneous nodes $V = \{I, M\}$ and edges E. Specifically, each node in the graph represents an item or an attribute in metadata; each edge represents a relationship between nodes. The set of edges E can be constructed in the following three manners. Figure 2 shows an example of these three types of edges within a session.

(1) **Item-Item Edges**: Any two items consecutively requested in the same session are linked to each other;
(2) **Item-Meta Edges**: The node of each item is connected to nodes representing attributes in its metadata;
(3) **Meta-Meta Edges**: Each meta relationship $(m_j, m_k) \in R$ is represented by an edge between the two corresponding nodes.

Given the heterogeneous graph $G = (V, E)$, we will first compute a low-dimensional feature representation for each node. We aim to find a mapping function $f : V \rightarrow \mathbb{R}^d$ from nodes to their low-dimensional feature representations, where d is the number of dimensions of the feature representation, and f can be considered as a $|V| \times d$ matrix, where $|V|$ denotes the number of nodes.

4.3 Learning Node Features

We extend the skip-gram architecture [25, 27] from natural language processing to learn the feature representations of nodes in a heterogeneous graph. In natural language processing, the skip-gram architecture learns relations between words and their context. Here each node in the network is treated as a word, and some random walk paths are sampled as sentences. We define $N_S(v) \subseteq V$ as the neighbor nodes for each node v via a sampling method S. Here we use a sampling method based on normalized random walk, which is presented in Section 4.4. The skip-gram model is extended to optimize the log-likelihood of the observed $N_S(v)$, conditioned on node v's feature representation $f(v)$ as follows:

$$\max_f \sum_{v \in V} \log P(N_S(v) \mid f(v)).$$

To make optimization more efficient, we adopt two standard assumptions [18]. First, we assume that, given node v's feature representation, v's neighbor nodes $N_S(v)$ can be observed conditionally independent of each other. Then $P(N_S(v) \mid f(v))$ can be factorized by the neighbor nodes as follows:

$$P(N_S(v) \mid f(v)) = \prod_{n \in N_S(v)} P(n \mid f(v)).$$

Second, we assume that any pair of neighboring nodes symmetrically affect each other in the d-dimensional space of feature representation. Therefore, given a node v, the conditional likelihood of every neighbor node $n \in N_S(v)$ can be modeled as a softmax unit [3] by reversing the previous formula:

$$P(n \mid f(v)) = \frac{\exp(f(n) \cdot f(v))}{\sum_{u \in V} \exp(f(u) \cdot f(v))}.$$

With these assumptions, the objective function can be rewritten as:

$$\max_f \sum_v \left(-\log Z_v + \sum_{n \in N_S(v)} f(n) \cdot f(v) \right),$$

where $Z_v = \sum_{u \in V} \exp(f(u) \cdot f(v))$ can be approximated by negative sampling [26]. In addition, this objective function can be optimized by stochastic gradient descent [7]. Algorithm 1 presents the detailed procedure to learn node features. Each node in the graph will be treated as the source of t random walks. These generated $t \times |V|$ random walks will be exploited to learn the node features by stochastic gradient descent. After learning node features, we will use the feature representations of items to compute session embeddings in Section 4.5.

Algorithm 1: LearningNodeFeatures(G, t, d, l)

Input: the graph $G = (V, E)$, walks per node t, the feature dimensions d, the fixed length l
Output: the embedding function f

```
walkset = ∅
for iter = 1 to t do
    foreach v ∈ V do
        W = NormalizedRandomWalk(v, l, G)
        walkset = walkset ∪ {W}
f = StochasticGradientDescent(walkset, d)
return f
```

4.4 Normalized Random Walk

We now present our sampling method based on normalized random walk. Random walk is one of the most popular solutions for graph-based embedding [18, 28]. However, traditional random walk that treats every edge equally important is not suitable for the heterogeneous graph constructed above in which popular items and metadata attributes may have much higher node degrees than the rest. For example, popular songs can be requested by more than ten thousand sessions while others are requested by ten sessions. Consequently, a random walk is likely to be confined to a small

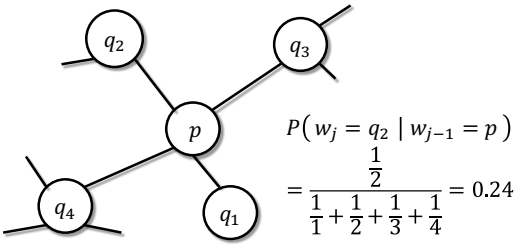


Figure 3: An illustration for the procedure of normalized random walks. The walk is going to be transited from p and evaluating transition probabilities of neighbor nodes.

Algorithm 2: NormalizedRandomWalk(r, l, G)

Input: the source node r, the fixed length l and the graph G
Output: the random walk W

$w_0 = r$
$W = [w_0]$
for $j = 1$ *to* l **do**
  Draw $w_j \sim P(w_j \mid w_{j-1})$
  W.append(w_j)
return W

number of high degree nodes and ignores the rest of the graph. To solve this problem, we use the normalized random walk to learn node embedding in the heterogeneous graph.

Consider a source node $r \in V$ and a random walk W from r with a given length l. Let w_j be the j-th node in the walk, where $w_0 = r$ is the first node in W. The degree of node v is denoted as $d(v)$; $N(v)$ denotes the set of neighbors of node v. Then node w_j can be generated by the normalized probability $P(w_j \mid w_{j-1})$:

$$P(w_j = q \mid w_{j-1} = p) = \begin{cases} \frac{1/d(q)}{Z_p} & \text{if } (p,q) \in E \\ 0 & otherwise \end{cases},$$

where $Z_p = \sum_{q' \in N(p)} \frac{1}{d(q')}$ is the term for normalization. Figure 3 shows an example. Node p has four neighboring nodes $N(p) = \{q_1, q_2, q_3, q_4\}$; the degree of q_3 is $d(q_3) = 3$, and the probability of transiting to q_2 is $P(w_j = q_2 \mid w_{j-1} = p) = 0.24$. The detailed procedure of generating a l-length random walk from r is provided in Algorithm 2. We will investigate how l affects the performance in Section 5.3.

4.5 Item-based Session Embedding

Since each session $s = \langle i_1, i_2, \cdots, i_{T_s} \rangle \in I^{T_s}$ consists of a sequence of items, the feature representation of a session can be computed by a combination of item features. A naïve way to derive session features is to simply compute the average features over all item occurrences. However, a user's affinity to an item may not be linearly correlated with the number of occurrences of the item in the session. For example, a user may play a song 100 times and another song 10 times. The affinity to the latter song is underestimated if we treat each play equally important.

Algorithm 3: UserIdentification($S(a)$, f)

Input: the set of sessions $S(a)$ and the embedding function f
Output: The set of cluster exemplars C(a)

Initialize X and Y as $|S(a)| \times |S(a)|$ matrices with zeros
repeat
  for $j = 1$ *to* $|S(a)|$ **do**
    for $k = 1$ *to* $|S(a)|$ **do**
      $\Delta_{jk} = \max_{k' \neq k} \{Y_{jk'} + Score(s_j, s_{k'})\}$
      $X_{jk} = Score(s_j, s_k) - \Delta_{jk}$
  for $j = 1$ *to* $|S(a)|$ **do**
    for $k = 1$ *to* $|S(a)|$ **do**
      if $j \neq k$ **then**
        $Y_{jk} = \min\left(0, \sum_{j' \notin \{j,k\}} \max\left(0, X_{j'k}\right)\right)$
      else
        $Y_{kk} = \sum_{j' \neq k} \max(0, X_{j'k})$
until *Convergence*;
return $C(a) = \{s_k \mid \forall s_k \in S(a), X_{kk} > 0\}$

To alleviate this problem, we model user's affinity to an item in a session by a kernel function of the number of item occurrences in this session. It has been shown [16, 17] that item sequences modeled after user behaviors tend to follow a Poisson distribution. Then we can adopt a square-root function to approximate the variance-stabilizing transformation to model user's affinity [14]. Let $\Gamma(s)$ be the set of distinct items in session s, and $Occ(s, i)$ be the number of occurrences of item i in session s. The feature representations of session s can be defined as follows:

$$f(s) = \frac{1}{\sum_{i \in \Gamma(s)} \sqrt{Occ(s,i)}} \sum_{i \in \Gamma(s)} \sqrt{Occ(s,i)} \cdot f(i).$$

These session features $f(s)$ can appropriately represent the characteristics of items in the session.

4.6 User Identification

After obtaining session features, we want to detect the number of users of each account a and group sessions by their actual issuers automatically. While most of the clustering algorithms require the number of clusters K_a as an input parameter, we propose using affinity propagation [15] algorithm to automatically discover the appropriate clustering number.

Specifically, we propose to cluster these sessions via a message passing mechanism between sessions, in which the exemplars are found and considered as the cluster representatives. The algorithm passes messages between sessions and iteratively updates two $|S(a)| \times |S(a)|$ matrices: *responsibility* matrix X and *availability* matrix Y. The responsibility value X_{jk} represents how session s_k is suitable to be the exemplar of session s_j compared to other exemplars. The availability value Y_{jk} estimates how appropriate for session s_j to pick s_k as its exemplar. Both X and Y are log-probability matrices. At the beginning, they are initialized to zero. In each iteration, all elements in X are estimated by Y_{jk} and a score function $Score(s_j, s_k)$ between features of two sessions s_j and s_k. Here $Score(s_j, s_k)$ is defined as the L2-distance between two feature vectors. Then we update Y_{jk} by summing up responsibilities in

X. We iteratively update X and Y until convergence. The sessions which remain positive responsibilities are the exemplars. The cluster number is the number of exemplars found in an account. This procedure is described in Algorithm 3. These exemplars will be used for user identification in past and future sessions.

User Identification using Cluster Exemplars. Recall that Section 3 introduced two goals of user identification, UI-Past and UI-New. To identify users from past sessions (i.e., UI-Past), Algorithm 3 can directly output the clusters of past sessions issued by an account, and each cluster corresponds to a user. To predict the user of a new session (i.e., UI-New) in account a, we need to, from all users detected for account a from UI-Past, find the one who is most likely to issue the new session only using the first few (say, ρ) items in the new session. These ρ items are treated as a shorter session from which we derive its feature representation follow the same procedure in Section 4.5. Then we can obtain its corresponding exemplar and cluster assignment by computing the L2-distance between feature vectors. The cluster with the shortest distance to the given new session is considered as the corresponding user. Here, ρ should be a small integer because our task is to identify the user as soon as he/she issues a new session. We will also investigate how ρ affects the performance in Section 5.3.

4.7 Complexity Analysis

Here we analyze the time and space complexity of SHE-UI.

Time Complexity. It costs $O(|R| + \sum_{i \in I} |t(i)| + \sum_{a \in A} \sum_{s \in S(a)} T_s)$ time to construct the heterogeneous graph. Assume that the number of metadata $|t(i)|$ for each item i and the length T_s of each session s are small constants. It becomes $O(|R| + |I| + |S|)$, where $S = \bigcup_{a \in A} S(a)$ is the set of all sessions of all accounts. Then SHE-UI spends $O(t \cdot |V| \cdot l) = O(|I| + |M|)$ time on collecting normalized random walk paths, where t and l are also treated as small constants. To obtain feature representations of nodes, the stochastic gradient descent process takes $O(|V|^2) = O(|I|^2 + |M|^2)$. Finally, the affinity propagation-based method determines the cluster number and clusters sessions in $O(\sum_{a \in A} |S(a)|^2)$ time. In addition, user identification cluster each account independently, so the algorithm is parallelizable. In summary, the time complexity of SHE-UI is acceptable as $O(|R| + |I|^2 + |M|^2 + \sum_{a \in A} |S(a)|^2)$.

Space Complexity. The heterogeneous graph occupies $O(E) = O(|R| + |I| + |S|)$ to store the edges. The random walk paths need $O(|V|) = O(|I| + |M|)$. Node and session features takes $O(|V| + |S|) = O(|V| + |I| + |M|)$ space. Finally, the affinity propagation costs $O(|S(a)|^2)$ space to create the responsibility and availability matrices for each account. If the algorithm runs sequentially, it takes $O(\max_{a \in A} |S(a)|^2)$; otherwise, the parallelized algorithm has to spend $O(\sum_{a \in A} |S(a)|^2)$ on storing matrices for all accounts simultaneously. Therefore, the space complexity of SHE-UI is $O(|R| + |I| + |M| + \max_{a \in A} |S(a)|^2)$ or $O(|R| + |I| + |M| + \sum_{a \in A} |S(a)|^2)$.

5 EXPERIMENTS

5.1 Datasets and Experimental Settings

The experiments are conducted on two datasets:

- **Synthetic Last.fm (Last.fm).** Last.fm [9] provides publicly available datasets for music recommendation. The Last.fm-1K dataset contains streaming (or listening) history of 1K users from Feb 2005 to May 2009. Since Last.fm-1K does not provide any information about account sharing, we manually create *synthetic accounts* by merging several users' history together following a similar procedure to that in [37]. 25% of accounts have 1, 2, 3, and 4 users respectively. Each user only belongs to one account.
- **Real data from KKBOX (KKBOX).** The dataset comprises listening logs of 100K accounts from the KKBOX music streaming service from December 1, 2014 to November 30, 2015. In this dataset, we use the device ID as the bronze standard to evaluate the accuracy of our user identification. Sessions with the same device ID are treated as being issued by the same user. Figure 4 shows the percentage of accounts over different number of sharing users. More than 75% of accounts in the real-world dataset are shared by multiple users. Figure 5 further illustrates the percentage of session pairs sharing common items or metadata. We observe that, among all sessions of a given multi-user account, sessions issued by the same user are much more likely to share common items or metadata than that of different users.

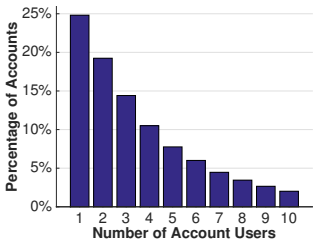


Figure 4: The percentage of accounts over different numbers of account users in the real-word KKBOX dataset.

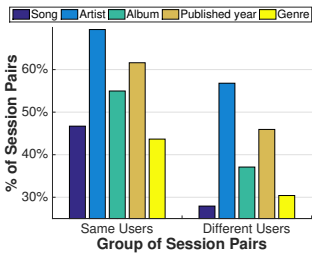


Figure 5: The percentage of session pairs sharing items or metadata within accounts in the KKBOX dataset. A pair of sessions can be issued by the same user or two different users.

Table 2: Statistics of Two Datasets.

(a) Session Information

	Last.fm	KKBOX
training sessions	209,313	10,783,556
testing sessions	209,925	10,782,507
accounts	370	88,399
unique users	922	343,723
items	314,763	564,164

(b) Metadata

Last.fm	
artists	60,410
KKBOX	
artists	43,157
albums	253,896
published years	77
genres	48

Data Preprocessing. In online streaming services, the log of each account is a sequence of entries, each of which contains a selected item with a timestamp. The log of each account can be partitioned into a list of sessions. We use 30 minutes of inactivity to define session boundaries. We exclude items of fewer than ten occurrences and accounts and sessions with fewer than ten entries. For the Last.fm dataset, the only available metadata are the artists of songs. For the KKBOX dataset, the available metadata include artists, albums, published years and genres. For both Last.fm and KKBOX

datasets, we use 50% sessions for training and 50% sessions for testing for the UI-Past task (clustering sessions into groups of users sharing an account) and the UI-New task (identifying the user of an incoming session), respectively. Table 2 shows the statistics of the two datasets after data cleaning and preprocessing.

Default Parameter Settings. Unless we specify otherwise, we set the length of random walks l to 5, and the dimension of features d to 512. For each node in the graph, 10 random walks are generated (i.e., $t = 10$ in Algorithm 1). For each session in UI-New, the first 5 items are used to derive session features (i.e., $\rho = 5$ in Section 4.6). The effects of these parameters are analyzed in Section 5.3.

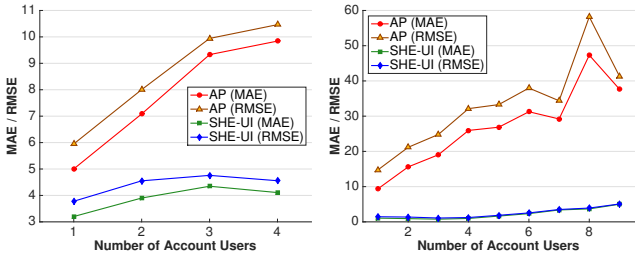
Evaluation Tasks. There are three evaluation tasks as follows.

- **(1) User Number Estimation**: we examine whether SHE-UI can accurately determine the number of users using the same account (i.e., the number of clusters among sessions of an account). Note that this evaluation can be regarded as *Multi-user Account Detection*. That says, accounts with two or more estimated users can be treated as multi-user ones. We consider this task as a regression problem, and thus use *Mean Absolute Error* (MAE) and *Root Mean Squared Error* (RMSE) [22] as the evaluation metrics to measure the difference between the number of users $|U(a)|$ and the number of estimated clusters $|C(a)|$ for every account $a \in A$. We examine the performance of our approach with affinity propagation (AP) [15], which can automatically determine the number of clusters (i.e., users).
- **(2) Performance in UI-Past** and **(3) Performance in UI-New**: we aim to evaluate the performance of SHE-UI for the tasks of UI-Past and UI-New, compared with a series of baseline methods. To further evaluate the effectiveness of session embedding in SHE-UI, experiments were conducted two times, with and without giving the number of users (i.e., session clusters) as input. If the numbers of users are known, K-Means++ [4] is applied to cluster sessions; otherwise, Algorithm 3 is applied. We compare the performance of SHE-UI using three conventional clustering evaluation metrics, including *Normalized Mutual Information* (NMI) [34], *Macro F-Score* (MAF) and *Micro F-Score* (MIF) [29].

Baseline Methods. To evaluate the performance in UI-Past and UI-New, we compare SHE-UI with several state of the art item-based clustering and embedding-based clustering methods. The item-based clustering methods treat items in sessions as features, which include KMeans++ (KM) [4], affinity propagation (AP) [15] and subspace clustering (SS) [43]. The embedding-based clustering methods derive d-dimensional representations of sessions for clustering, which include word2vec (W2V) [26], LINE [35] and DeepWalk (DW) [28]. Note that these embedding-based clustering methods only derive the feature representations of items and need to apply our approach in Section 4.5 and Section 4.6 to obtain session embedding and the clustering results. We do not furnish a comparison with node2vec [18] since it is too time-consuming to compute probabilities for alias edges in a large dense graph.

5.2 User Identification Performance

User Number Estimation. Figure 6 shows the performance of detecting cluster numbers for accounts with different numbers of users. SHE-UI significantly outperforms AP, especially in accounts



(a) Last.fm (b) KKBOX

Figure 6: The performance of determining the cluster number. The lower MAE and RMSE indicate the better performance. All improvements are statistically significant at 95% confidence level by a paired t-test.

with multiple users. Such results prove the usefulness of session embedding in SHE-UI. The improvement is more significant in KKBOX dataset. KKBOX has much richer metadata that translate into a much larger and more diverse graph, which creates significant challenges for AP. We can further observe that both MAE and RMSE of AP fluctuate as the number of users increases. In contrast, SHE-UI consistently has lower errors. Such results demonstrate that SHE-UI can make accurate prediction even for accounts shared by a large number of users (e.g., 8, in Figure 6(b)).

Performance in UI-Past. Table 3 shows the results. Note that the result of AP is not reported under the section "Known Numbers of Users" since it cannot take a predefined cluster number. Also note that under "Unknown Numbers of Users", all compared methods except AP utilize our method (Section 4.6) to determine cluster numbers because they need it as an input parameter. We observe that SHE-UI consistently outperforms other methods in every metric for the real-world data and almost all metrics for the synthetic dataset, especially the improvement in NMI is more than 10% over the best competitor DW. Note that NMI values are relatively low (compared with MAF and MIF) since NMI is sensitive to the cluster size. As expected, when the cluster number is unknown, the performance of all methods degrades (comparing to the case where the cluster number is unknown). Nevertheless, SHE-UI can still perform reasonably well since it can accurately estimate the numbers of users in accounts (as proved in Figure 6). Furthermore, the embedding-based methods generally outperform item-based methods. Note that LINE performs the worst in Last.fm dataset because the information of metadata is too sparse for LINE to capture appropriate features through edge sampling. Such result not only shows the effectiveness of embedding-based approaches but also verifies the capacity of SHE-UI to accurately derive session embedding.

Performance in UI-New. Table 3 shows the results of UI-New as well. We observe similar performance to that in UI-Past: SHE-UI outperforms other competing methods in every metric for the real-world dataset and almost all metrics for the synthetic dataset. For instance, SHE-UI outperforms the best competitor (i.e., DW) by around 15% of NMI in Last.fm and by around 12% in KKBOX on average for both cases with known and unknown numbers of users. In addition, the performance in UI-New is generally worse than

Table 3: The results of user identification behind shared accounts. The higher values in metrics indicate the better performance. All improvements of SHE-UI against DW [28] are statistically significant at 95% confidence level by a paired t-test.

Dataset	Synthetic Last.fm						Real Data from KKBOX					
	UI-Past			UI-New			UI-Past			UI-New		
Metric	NMI	MAF	MIF	NMI	MAF	MIF	NMI	MAF	MIF	NMI	MAF	MIF
Known Numbers of Users												
KM [4]	0.2956	0.6109	0.7400	0.2802	0.6106	0.7400	0.3640	0.5710	0.6516	0.3286	0.5644	0.6592
SS [43]	0.2954	0.6109	0.7405	0.2793	0.6105	0.7403	0.3627	0.5707	0.6612	0.3258	0.5642	0.6585
W2V [26]	0.4865	0.7022	0.7982	0.4428	0.6823	0.7769	0.3828	0.5855	0.6524	0.3571	0.5739	0.6488
LINE [35]	0.2667	0.5611	0.6544	0.2622	0.5724	0.6768	0.3830	0.5874	0.6463	0.3456	0.5634	0.6183
DW [28]	0.5597	0.7372	0.8162	0.5148	0.7161	0.7947	0.3995	0.5976	0.6656	0.3587	0.5775	0.6419
SHE-UI	**0.6108**	**0.7613**	**0.8393**	**0.5718**	**0.7455**	**0.8236**	**0.4281**	**0.6111**	**0.6804**	**0.3880**	**0.5948**	**0.6625**
Unknown Numbers of Users												
AP [15]	0.1677	0.3413	0.3474	0.1546	0.4825	0.5408	0.1884	0.4828	0.4978	0.1783	0.5225	0.5569
KM [4]	0.1189	0.5842	**0.7003**	0.1061	0.5622	**0.6697**	0.1856	0.5264	0.5849	0.1516	0.5041	0.5642
SS [43]	0.1518	0.5838	0.6856	0.1312	0.5616	0.6582	0.1927	0.5312	0.5904	0.1841	0.5151	0.5851
W2V [26]	0.2981	0.6413	0.6587	0.2560	0.6148	0.6347	0.2081	0.5337	0.6025	0.1807	0.5149	0.5818
LINE [35]	0.0813	0.5641	0.6687	0.0964	0.5546	0.6552	0.1955	0.5365	0.6083	0.1010	0.4782	0.5394
DW [28]	0.3053	0.6286	0.6557	0.2669	0.5966	0.6244	0.2158	0.5508	0.6249	0.1941	0.5322	0.6024
SHE-UI	**0.3375**	**0.6563**	0.6782	**0.3214**	**0.6323**	0.6568	**0.2426**	**0.5610**	**0.6309**	**0.2218**	**0.5451**	**0.6117**

that in UI-Past because only the first ρ items of a new session are used. In summary, SHE-UI is able to efficiently identify the user of any incoming session (among all users sharing an account). We will demonstrate its utility in improving the performance of online item recommendation customized for the identified user in Section 5.4.

5.3 Study of Parameter Sensitivity

This section presents how parameter settings in SHE-UI may affect its performance in user number estimation and user identification. We present only the performance using the KKBOX dataset, as we observe a similar performance on Last.fm.

Figure 7(a) and 7(e) exhibit the performance of user identification measured by NMI as a function of the length of normalized random walk (i.e., l in Algorithm 2) for the scenarios of known and unknown numbers of users respectively. Even though the performance generally improves as the length increases, we observe that it saturates when the length reaches five. Longer random walk does not necessarily warrant additional benefit. We thus set our default length to 5.

Figure 7(b) and 7(f) show the performance by varying the feature dimensions (i.e., d in Algorithm 1). The results show that, while higher dimensions can in general lead to more accurate user identification, the performance plateaus at 512 dimensions. We thus choose 512 as our default setting.

Figure 7(c) and 7(g) exhibit how the performance of SHE-UI is affected by the number of random walks generated for each node (i.e., t in Algorithm 1). The results are generally consistent with the intuition that more sampled walks lead to better performance. However, the performance saturates when the number reaches to 10, which is our choice of default value. This is because too many random walks starting from a node tend to bring duplicate information into feature representation.

Figure 7(d) and 7(h) demonstrate the performance of SHE-UI in UI-New by varying the number of items used to derive session features (i.e., ρ in Section 4.6). The performance is significantly better when the number of users is known than otherwise. It is also reasonable that more items seen in a new session lead to better performance of identifying the corresponding user. We set the default value to 5.

5.4 Item Recommendation with SHE-UI

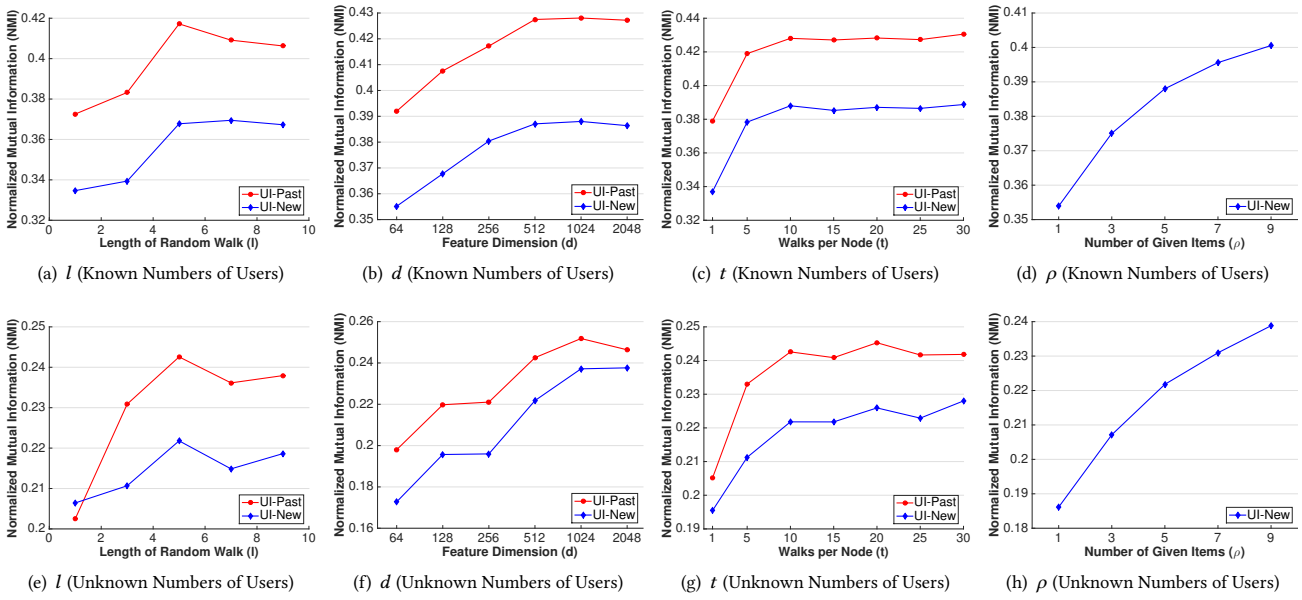
Conventional recommender systems [6, 9, 20] do not attempt to distinguish users sharing the same account and thus can only recommend items to "accounts", termed *account-level recommendation* (ARec) here. The preferences of individual users are not adequately captured. It is expected that by adding *user-level recommendation* (URec), we can effectively model the user preferences. Therefore, we propose a hybrid recommender that linearly combines the Account-level and User-level item RECommendation, which is termed AURec.

Let $\overline{R_A}(a,i)$ and $\overline{R_U}(u,i)$ be item i's recommendation scores that ARec gives to account a and URec gives to the identified user u of account a, respectively. We employ the Bayesian personalized ranking matrix factorization (BPRMF) [30] model to compute the $\overline{R_A}$ and $\overline{R_U}$ scores. For a session s issued by the user u in the account a, AURec estimates the score of the item i by

$$\overline{R_{AU}}(a,u,i) = (1-\alpha)\cdot\overline{R_A}(a,i) + \alpha\cdot\overline{R_U}(u,i),$$

where α is the parameter to control the weights of URec and ARec, and we set $\alpha = 0.6$ by default.

Evaluation Settings. The evaluation is conducted by using KKBOX data, in which items are songs. The split of training and testing is the same as in Section 5.1. We ran these experiments under the setting of UI-New with unknown numbers of users. We consider item recommendation as a ranking task. For each session s in the testing data, we want to examine how well we can predict the remaining items in the session based on the first 5 items. We first compute the $\overline{R_{AU}}$ for every item and sort them by descending order



(a) l (Known Numbers of Users) (b) d (Known Numbers of Users) (c) t (Known Numbers of Users) (d) ρ (Known Numbers of Users)

(e) l (Unknown Numbers of Users) (f) d (Unknown Numbers of Users) (g) t (Unknown Numbers of Users) (h) ρ (Unknown Numbers of Users)

Figure 7: Results of Parameter Sensitivity Study.

of their $\overline{R_{AU}}$ scores. We then examine the rankings of the items appear in the remainder part of the session s. Ideally, we want to see that these items are top ranked by the $\overline{R_{AU}}$ scores. To quantitatively measure the recommendation performance, we use standard evaluation metrics [12], including *Mean Reciprocal Rank* (MRR), *Mean Average Precision* (MAP) and *Precision at k* (P@k) to compare AURec with four ARec recommender systems: recommendation by item popularity (PopRec), maximum margin matrix factorization (MMMF) [39], Bayesian personalized ranking matrix factorization (BPRMF) [30], and collaborative less-is-more filtering (CLiMF) [32].

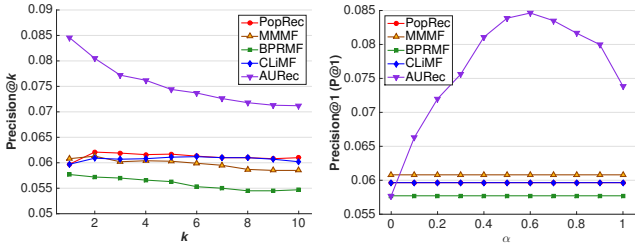
Experimental Results. Table 4 shows the results. It is clear that AURec significantly outperforms the best ARec competitor (i.e., MMMF) by a wide margin (from 22% to 39% in the three metrics). Such result reveals that user identification using SHE-UI can truly enhance the accuracy of item recommendation. To understand how parameters influence the recommendation performance, we further vary k in P@k and α in AURec. The results are shown in Figure 8. We observe that AURec consistently outperforms others as k increases. In Figure 8(b), the best performance is achieved at $\alpha = 0.6$ which demonstrates the need for combining URec and ARec. Using either ARec ($\alpha = 0$) or URec ($\alpha = 1$) along will produce substantially worse performance. It is worth noting that this task is extremely challenging. A session may be relatively short, consisting only a small number of items. A user may have broad interests and may not always play all of his/her favorite songs in one single session. In the above setting, a favorite item is counted as a negative instance if it does not appear in the current testing session.

6 CONCLUSIONS AND DISCUSSIONS

This paper investigates the problem of identifying individual users behind shared accounts in two settings: for historical data (UI-Past), and for incoming sessions (UI-New). An unsupervised learning-based framework, Session-based Heterogeneous graph Embedding for User Identification (SHE-UI), is proposed. Experiments conducted on KKBOX and Last.fm datasets demonstrate that SHE-UI can not only outperform the best competitor by at least 10% in UI-Past and 12% in UI-New in terms of NMI, but also significantly improve the performance of music recommendation by 39% measured by Precision@1. Accurate user identification is beneficial to both users and service providers. Users can enjoy a "true" recommendation while service providers can establish more desirable marketing strategies according to the behaviors of sharing accounts.

Table 4: Results of item recommendation with AURec.

	PopRec	MMMF [39]	BPRMF [30]	CLiMF [32]	AURec
MRR	0.1242	0.1421	0.1353	0.1400	**0.1727 (+22%)**
MAP	0.0317	0.0331	0.0330	0.0337	**0.0439 (+30%)**
P@1	0.0597	0.0608	0.0577	0.0597	**0.0846 (+39%)**



(a) Precision@k (P@k) (b) P@1 over α

Figure 8: Results of recommendation by varying k and α.

The content providers (e.g. artists and publishers) can also gain insights in the taste and trend of different user groups.

The Session-based Heterogeneous graph Embedding (SHE) learns the feature representation for each session, which may be applicable to inferring account attributes, predicting session life-cycle, and detecting abnormal accounts, in addition to user identification. Moreover, the proposed SHE-UI can be applied to any activity and event sequences, allowing for incorporation of metadata. For example, one may study activity traces from smart devices and apply SHE-UI for activity recognition.

ACKNOWLEDGMENT

We would like to thank the anonymous reviewers for their helpful comments. The work was partially supported by NIH U01HG008488, NIH R01GM115833, NIH U54GM114833, and NSF IIS-1313606. This work was also supported by Ministry of Science and Technology (MOST) Taiwan: MOST 107-2636-E-006-002 and MOST 106-3114-E-006-002, and by Academia Sinica AS-107-TP-A05.

REFERENCES

[1] Michal Aharon, Eshcar Hillel, Amit Kagian, Ronny Lempel, Hayim Makabee, and Raz Nissim. 2015. Watch-It-Next: A Contextual TV Recommendation System. In *Proceedings of European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD '15)*. 180–195.

[2] Leman Akoglu, Rishi Chandy, and Christos Faloutsos. 2013. Opinion Fraud Detection in Online Reviews by Network Effects. *ICWSM* 13 (2013), 2–11.

[3] Yuichiro Anzai. 2012. *Pattern recognition and machine learning*. Elsevier.

[4] David Arthur and Sergei Vassilvitskii. 2007. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, 1027–1035.

[5] Payal Bajaj and Sumit Shekhar. 2016. Experience Individualization on Online TV Platforms Through Persona-based Account Decomposition. In *Proceedings of ACM International Conference on Multimedia (MM '16)*. 252–256.

[6] Jesus Bobadilla, Fernando Javier Ortega, Antonio Hernando, and Abraham Gutierrez. 2013. Recommender Systems Survey. *Knowledge-Based Systems* 46 (2013), 109–132.

[7] Léon Bottou. 1991. Stochastic gradient learning in neural networks. *Proceedings of Neuro-Nımes* 91, 8 (1991).

[8] Qiang Cao, Xiaowei Yang, Jieqi Yu, and Christopher Palow. 2014. Uncovering large groups of active malicious accounts in online social networks. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 477–488.

[9] Oscar Celma. 2010. *Music Recommendation and Discovery in the Long Tail*. Springer.

[10] Shiyu Chang, Wei Han, Jiliang Tang, Guo-Jun Qi, Charu C Aggarwal, and Thomas S Huang. 2015. Heterogeneous network embedding via deep architectures. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 119–128.

[11] Ting Chen and Yizhou Sun. 2017. Task-guided and path-augmented heterogeneous network embedding for author identification. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM, 295–304.

[12] W Bruce Croft, Donald Metzler, and Trevor Strohmann. 2010. *Search engines: Information Retrieval in Practice*. Pearson Education.

[13] Yuxiao Dong, Nitesh V Chawla, and Ananthram Swami. 2017. metapath2vec: Scalable representation learning for heterogeneous networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 135–144.

[14] Brian S Everitt. 2006. *The Cambridge dictionary of statistics*. Cambridge University Press.

[15] Brendan J Frey and Delbert Dueck. 2007. Clustering by passing messages between data points. *science* 315, 5814 (2007), 972–976.

[16] Prem Gopalan, Francisco J Ruiz, Rajesh Ranganath, and David M Blei. 2014. Bayesian Nonparametric Poisson Factorization for Recommendation Systems.. In *AISTATS*. 275–283.

[17] Prem K Gopalan, Laurent Charlin, and David Blei. 2014. Content-based recommendations with poisson factorization. In *NIPS*. 3176–3184.

[18] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)*. 855–864.

[19] BalÃązs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based Recommendations with Recurrent Neural Networks. (2016).

[20] Balazs Hidasi and Domonkos Tikk. 2016. General Factorization Framework for Context-aware Recommendations. *Data Mining and Knowledge Discovery (DMKD)* 30, 2 (March 2016), 342–371.

[21] Bryan Hooi, Neil Shah, Alex Beutel, Stephan Günnemann, Leman Akoglu, Mohit Kumar, Disha Makhija, and Christos Faloutsos. 2016. Birdnest: Bayesian inference for ratings-fraud detection. In *Proceedings of the 2016 SIAM International Conference on Data Mining*. SIAM, 495–503.

[22] Rob J Hyndman and Anne B Koehler. 2006. Another look at measures of forecast accuracy. *International journal of forecasting* 22, 4 (2006), 679–688.

[23] Nitin Jindal, Bing Liu, and Ee-Peng Lim. 2010. Finding unusual review patterns using unexpected rules. In *Proceedings of the 19th ACM international conference on Information and knowledge management*. ACM, 1549–1552.

[24] Niels Landwehr. 2008. Modeling interleaved hidden processes. In *Proceedings of the 25th international conference on Machine learning*. ACM, 520–527.

[25] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *Proceedings of Workshop at International Conference on Learning Representations*.

[26] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[27] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global Vectors for Word Representation.. In *Proceedings of Empirical Methods in Natural Language Processing (EMNLP '14)*, Vol. 14. 1532–1543.

[28] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '14)*. 701–710.

[29] David Martin Powers. 2011. Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. (2011).

[30] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*. AUAI Press, 452–461.

[31] Guy Shani, David Heckerman, and Ronen I. Brafman. 2005. An MDP-Based Recommender System. *Journal of Machine Learning Research (JMLR)* 6 (2005), 1265–1295.

[32] Yue Shi, Alexandros Karatzoglou, Linas Baltrunas, Martha Larson, Nuria Oliver, and Alan Hanjalic. 2012. CLiMF: learning to maximize reciprocal rank with collaborative less-is-more filtering. In *RecSys*. ACM, 139–146.

[33] Adish Singla, Ryen W. White, Ahmed Hassan, and Eric Horvitz. 2014. Enhancing Personalization via Search Activity Attribution. In *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR '14)*. 1063–1066.

[34] Alexander Strehl and Joydeep Ghosh. 2002. Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *Journal of machine learning research* 3, Dec (2002), 583–617.

[35] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web (WWW '15)*. 1067–1077.

[36] Bartlomiej Twardowski. 2016. Modelling Contextual Information in Session-Aware Recommender Systems with Neural Networks. In *Proceedings of the 10th ACM International Conference on Recommender Systems (RecSys '16)*. 273–276.

[37] Koen Verstrepen and Bart Goethals. 2015. Top-N Recommendation for Shared Accounts. In *Proceedings of the 9th ACM Conference on Recommender Systems (RecSys '15)*. 59–66.

[38] Zhijin Wang, Yan Yang, Liang He, and Junzhong Gu. 2014. User Identification within a Shared Account: Improving IP-TV Recommender Performance. In *Proceedings of the 18th East European Conference on Advances in Databases and Information Systems (ADBIS '14)*. 219–233.

[39] Markus Weimer, Alexandros Karatzoglou, and Alex Smola. 2008. Improving maximum margin matrix factorization. *Machine Learning* 72, 3 (2008), 263–276.

[40] Ryen W. White and Ahmed Hassan Awadallah. 2015. Personalizing Search on Shared Devices. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '15)*. 523–532.

[41] Ryen W. White, Ahmed Hassan, Adish Singla, and Eric Horvitz. 2014. From Devices to People: Attribution of Search Activity in Multi-user Settings. In *Proceedings of the 23rd International Conference on World Wide Web (WWW '14)*. 431–442.

[42] Yan Yang, Qinmin Hu, Liang He, Minjie Ni, and Zhijin Wang. 2015. Adaptive Temporal Model for IPTV Recommendation. In *Proceedings of the 16th International Conference on Web-Age Information Management (WAIM '15)*. 260–271.

[43] Amy Zhang, Nadia Fawaz, Stratis Ioannidis, and Andrea Montanari. 2012. Guess Who Rated This Movie: Identifying Users Through Subspace Clustering. In *Proceedings of the Twenty-Eighth Conference on Uncertainty in Artificial Intelligence (UAI'12)*. 944–953.

[44] Yafeng Zhao, Jian Cao, and Yudong Tan. 2016. Passenger Prediction in Shared Accounts for Flight Service Recommendation. In *Proceedings of the 10th Asia-Pacific Services Computing Conference on Advances in Services Computing (APSCC '16)*. 159–172.