A recommendation system for scientific water data

Zhaokun Xue¹ · Alva Couch¹

Received: 13 November 2020 / Accepted: 5 April 2021
© The Author(s) 2021

Abstract

We describe a recommendation system for HydroShare, a platform for scientific water data sharing. We discuss similarities, differences and challenges for implementing recommendation systems for scientific water data sharing. We discuss and analyze the behaviors that scientists exhibit in using HydroShare as documented by users' activity logs. Unlike entertainment system users, users on HydroShare tend to be task-oriented, where the set of tasks of interest can change over time, and older interests are sometimes no longer relevant. By validating recommendation approaches against user behavior as expressed in activity logs, we conclude that a combination of content-based filtering and a latent Dirichlet allocation (LDA) topic modeling of user behavior—rather than and instead of LDA classification of dataset topics—provides a workable solution for HydroShare and compares this approach to existing recommendation methods.

Keywords Recommendation systems · User behavior modeling · Scientific water data · Latent Dirichlet allocation · Content-based filtering

1 Introduction

Recommendation systems have been widely used in online applications, from e-commerce domains to video streams and e-news platforms. Recommendation systems deliver personalized content by predicting users' preferences based upon their previous activities, such as purchase history and ratings of previous choices. Such personalized experience not only helps users simplify the process for discovering desired items, but also leads to more profits for online retailers by encouraging more purchases and future business [6,40].

In this paper, we describe a recommendation system for scientific water data. Unlike entertainment, scientific water data browsing is often task-based; the user is trying to accomplish some science or engineering task that may not be of interest after the task is complete. By contrast, entertainment application users' interests usually do not change as much over time. Recommendations for scientific water data are intended to enable scientific workflows and encourage data reuse between related projects.

HydroShare is an online collaboration platform that allows users to share data and discover shared data of others [45,50]. Unlike many data publication systems, HydroShare allows pre-publication sharing and discovery of data that has not yet been issued a digital object identifier (DOI) or formally published. Users can share resources with one another in small groups and among larger “communities” of groups. Currently, there are more than 4000 registered users and 7800 resources available on HydroShare. Besides dealing with many kinds of data, we also cope with many social groups of users engaged in coordinated pursuits (unlike entertainment recommendation systems, where users' pursuits are often autonomous), although we do not enjoy large-scale user participation like that of other commercial or entertainment platforms.

Our contributions include:

- A topic modeling method using latent Dirichlet allocation (LDA) to analyze scientific users' activities to identify classes of user behavior and thus relevance of recommendations.
- An algorithm for using LDA to cluster users' interests instead of documents, to better represent distinct and multiple interests of each user.
- A content-based filtering algorithm to generate useful recommendations for scientific water data users with sev-

✉ Zhaokun Xue
Zhaokun.Xue@tufts.edu
Alva Couch
alva.couch@tufts.edu

¹ Computer Science Department, Tufts University, Medford, MA, USA

eral distinct interests based upon users' 30-day activity data.

The paper is structured as follows. In Sect. 2, we review the necessary background for LDA-based recommendation systems, recommendation systems based upon implicit feedback and the session-based recommendation systems. In Sect. 3, we describe the representation for the data in our system. In Sect. 4, we provide a detailed analysis for the usage pattern of our users. In Sect. 5, we discuss the content-based LDA model we constructed whose goal is to make recommendations based upon users' multiple-interest behavior in a period of time. In Sect. 6, we present the methodology for evaluation and experimental results for comparing our recommendation system with other baseline methods. We conclude the paper in Sect. 7 with discussion of future work.

2 Related work

Most existing recommendation systems that utilize the LDA algorithm apply the algorithm to the resource space. The intent of typical use of LDA is to reduce resources' feature dimension to a lower topic dimension. Bui et al. [5], Hariri et al. [14], Massquantity [29] use each topic generated by the LDA model to cluster available resources. Based upon each resource's probability distribution over topics, they assign the resource to the cluster (topic) that has the highest probability or assign data to clusters (topics) that are over a preset threshold value. Bui et al. [4] combines the LDA algorithm with the k-means clustering algorithm by first using LDA to reduce the data from a high feature dimension to a lower topic dimension space and then applying the k-means algorithm to cluster the resulting topics. Also, they show that distance metrics based upon probability distribution, such as Jensen–Shannon distance [10,13,32], are reasonable to use for calculating similarities of probability distributions. Saraswat et al. [39] builds a topic preference profile for each user based upon each document's topic probability distribution generated by the LDA model with a weight for each document from the user's review rating. Rosen-Zvi et al. [38] extends the basic LDA model to an *author-topic* model, which solves the problem of a document with multiple authors and summarizes each author's interests of topics.

Amami et al. [1] also uses the LDA topic modeling to build each user's profile for recommending scientific papers. They train an LDA model for each user based upon abstracts of papers that are published by the user. For each unseen paper, they calculate similarity between the word probability distribution of each topic from the LDA model with the word probability distribution of the unseen paper generated by language modeling. They only use the similarity score generated by the most similar topic for the unseen paper as

the similarity measurement between the unseen paper and the user's profile. They also assume that each user can be summarized by one topic.

In our system, recommendations are made passively based upon users' activity history. Whenever a user selects a resource, we implicitly record that action as a positive evidence of interest in that resource. This is also known as *implicit feedback* from users, which our algorithms learn. A main advantage of using implicit feedback is that it does not require extra effort from users to provide ratings or reviews [21]. The data rating system in our software is almost completely unused. Because not all missing values are actually negative feedback, by accounting for user feedback, we might lose useful data for making recommendations, which is considered to be a major drawback of implicit feedback.

By contrast to other uses of LDA, we use the LDA model to deal with users who have multiple interests in a period of time and weight each interest by the probability generated from the LDA model. We utilize users' implicit feedback to drive content-based filtering (CBF) by using the LDA algorithm to build user preference profiles for modeling users' interests. Unlike most existing LDA-based recommendation systems that use the LDA algorithm as a dimensionality reduction scheme, we utilize it to cluster users' preferences, in order to cope with users' multiple interests with overlapping sets of keywords. This copes with a number of features of our problem space, including that users express multiple interests in one time period and that research often requires combining resources of different kinds.

In recent years, considerable research has studied on using model-based collaborative filtering (CF) methods on users' implicit feedback. Hu et al. [18], Pan et al. [33] propose a regularized alternating least-squares optimization matrix factorization method by assigning varying weights to the positive and missing data in the rating matrix. Frederickson [11], Li et al. [27], Sindhvani et al. [42] propose further improvement for CF by applying different weighting strategies on the implicit rating matrix. Our algorithm uses CBF in combination with implicit feedback. Alas, the social aspects of our system are limited to organizing users in loosely coupled groups, so that collaborative filtering is not applicable at present.

Our users' task-oriented usage behavior is similar to the assumptions of the emerging research in the session-based recommendation systems (SBRS) that users' preferences will change over sessions [48]. The SBRS makes recommendations based upon a *session* of users' preferences, where a *session* is a set of items that users selected during a period of time [48]. The major advantage of SBRS over the conventional recommendation systems is that it takes users' short-term preferences into consideration, which generates more reliable and timely recommendations [48]. Our recommendation system is built upon the observation of our

users' task-oriented behavior that users' interests may vary over time, which makes our design belong to the topics of SBRS. Most recent works for SBRS are based upon sequential data with strict order in sessions. Shani et al. [41], Wu et al. [49] introduce and design solutions for SBRS based upon the first-order Markov Chain model, which makes prediction for the next items based upon the item co-occurrence matrix. After [16] introduces the idea of using recurrent neural networks (RNNs) to make recommendations for SBRS, RNN-based models become a major research area for SBRS because of its benefit for modeling sequential data [48]. A series of works including [15, 34, 44, 48] are proposed on making improvements based upon the RNN model.

In our data, a user's interests may change over time based upon the task on which the user is working; however, within the period of time in which the user focuses on the same task, we do not presume that order is meaningful for users' selections of resources. Two users who employ a different sequence for selecting the same resources may well be working on the same problem or task. Therefore, our work does not focus on solving the sequential dependency problem in SBRS. Approaches described in [28, 31, 46] have been developed for unordered session data based upon pattern/rule-based approaches. Most existing work for SBRS makes recommendations based upon the single interest of a user in one session.

By contrast, our users demonstrate a multiple-interest behavior in a period of time. We propose a possible solution for dealing with users' multiple interests in one period of time and periodically refresh the model to capture users' new interests evolved over time. To support this solution, we provide a detailed user behavior analysis for how we observe and prove the task-oriented behavior for our users. Based upon our research, such a detailed analysis for users' behavior is lacking in previous work.

3 Representing data resources

Scientific water resources on HydroShare are described by a complex superset of Dublin Core metadata, including title, abstract, keywords, author, etc. [9]. Resource keywords on HydroShare are specified by the resource's creator and not curated. Utilizing such uncontrolled vocabularies results in a complex keywords set for HydroShare resources.

To construct a document used for the LDA model, we select resource keywords from each resource's *Abstract* and *Subject* (Dublin Core) metadata. We found in experiments that selecting keywords from abstracts by omitting common English "stop-words" was not selective enough to provide acceptable results. Instead, we use the Observation Data Model2 (ODM2) [17] and CSDMS Standard Names scientific controlled vocabularies [8] to determine "keep-words"

from each abstract to use as keywords. Both of these controlled vocabularies are sets of terms that scientists consider meaningful in describing data. For entries in the CSDMS standard names, each standard name is constructed by concatenating *object name*, *attribute* and *measurement* parts. In CSDMS, we note that *attribute* and *measurement* parts are not appropriate as keywords to distinguish between different objects. Therefore, in our system, we only select CSDMS object names as "keep-words." As CSDMS object names contain standard English "stop-words," we must also remove these "stop-words" from our "keep-words" set. The "stop-words" set in our system is composed of common English "stop-words" and a customized "stop-words" set based upon words used in water science that are common to most abstracts and do not determine subject matter. Then, we add keywords from the resource's subject metadata to the "keep-words" set. Next, we perform lemmatization on the "keep-words" set to group inflected words to the same root word. The resulting "keep-words" set is considered as the keywords set for the resource, which is used as the resource's content in our LDA algorithm. Figure 1 illustrates the process for extracting keyword sets for our resources.

4 Patterns of use of water data

Before we discuss how to generate recommendations for scientific water data users, we discuss whether the user's interests are the same or change over time. The common assumption in recommendation systems—that user interests do not change—may not be valid for scientific data. This can highly influence the perceived quality of recommendation results.

Using detailed user activity data from the calendar years 2018 and 2019, we performed an analysis on HydroShare users' behavior, looking for behaviors in which users repeat or do not repeat patterns of keywords access. First, we draw a keywords similarity graph for each user based upon keywords selected by the user in every 30-day activity time window to demonstrate that our users exhibit multiple-interest behaviors. Then, we plot a graph to demonstrate similarity between resources on active days for each user by using the user's 2-year activity logging periods from the 2018 and 2019 calendar years.

4.1 Users' behavior in 30-day time window

Based upon our users' task-oriented behavior that users' selections of resources are driven by the tasks they work on in a period of time, first, we analyzed whether our users exhibit only one interest or multiple interests over a period of time. Based upon our observations, in our data, each task period is usually about 30 days. We performed the analysis by

Fig. 1 Process for extracting keywords for a resource that is used as a document in our LDA model

applying a 30-day sliding window method that is described in Wang et al. [47]. Using activity data from 2018 and 2019 calendar years, for each user, we apply Jaccard similarity to calculate the similarity of keyword sets from resources selected by the user during each 30-day activity logging data window that is sampled starting from 01/01/2018. The similarity of keyword sets of each 30-day activity logging data is defined as:

$$\text{Similarity} = \frac{\sum_{r \in R} \text{Jaccard_similarity}(\text{keywords}(r), \text{Union}_{r \in R}(\text{keywords}(r)))}{|R|} \quad (1)$$

where r is a resource from the set of resources R that are selected by the user during the sampled 30-day activity logging data. Then, we use the averaged result as the keywords similarity measurement for that 30-day activity logging data. The smaller result means lower similarity among resources' keywords selected by the user, which indicates the user interests in various keyword sets. In such case, representing a user's interests a single keywords set is insufficient to describe the users interests correctly.

Figure 2 shows a sample result for a user's keywords similarity analysis using 2018 and 2019 calendar years' activity logging data.

From this figure, we can conclude that most times the user selected various keyword sets during the 30-day activity logging data, which indicates that the user usually has several distinct interest sets during the 30-day activity logging data. This result also provides evidence that we should apply different strategies from the aforementioned session-based recommendation systems. As in [15], the conventional session-based recommendation systems try to learn the "theme" of a session and generate recommendations based upon that "theme." However, our users express multiple interests in 30-day activity logging data in our system, which requires us to develop an algorithm to handle distinct interests shown by the user during a period of time. In our system, more than 90% users express similar behavior pattern.

Fig. 2 A particular user's keywords' similarity graph for each 30-day activity logging data during 2018 and 2019 calendar years. x coordinates represent the i th 30-day sliding window in 2018 and 2019 calendar years. y coordinates represent the similarity scores

4.2 Building the original LDA model

First, we train an LDA topic model on our entire resources' space, which is used for analyzing similarities among resources selected by our users. We implement the LDA model using the Python *gensim* [35] package. To use the *gensim* LDA model, we must determine the number of topics for training the LDA model. To tune this parameter, we use the "c_v topic coherence score" (CS) in *gensim*'s *CoherenceModel* [37] to evaluate the topic modeling results for our LDA model. The CS for the LDA model is a qualitative approach to interpret coherence of a topic by measuring similarity between each pair of the top high-scored words in the topic, i.e., measuring word coherence in each topic [20]. A topic is considered to be coherent, if the selected top high-scored words are related, i.e., high-scored words within a topic are more likely to co-occur [43], which will result in a higher CS.

As before, we use the "c_v topic coherence score" (CS) in *gensim*'s *CoherenceModel* [37] to measure the goodness for our LDA results, which varies from 0 to 1. We repeat the LDA training process on different numbers of topics 30 times to reduce the variance caused by different result from

Table 1 Averaged coherence scores of using different numbers of topics for training an original LDA model on the entire resource space

# LDA topics	CS
10	0.508
50	0.492
100	0.448
150	0.551
200	0.596
250	0.610
500	0.650
600	0.748
800	0.753

each training of the LDA model, and use the averaged CS as the topic coherence measurement for each LDA model. Table 1 shows results for tuning the LDA model with different numbers of topics and its corresponding CS. Based upon results shown in column “CS” of Table 1, the largest increase in the CS is when the number of topics is increased from 100 to 150. Based upon the method for evaluating topic modeling results in [26], picking a number that represents the end of a rapid growth of the CS usually provides an interpretable results, which is 150 in our case. Although as we continue to increase the number of topics, the CS becomes very high, which may be caused by the same keywords being repeated in multiple topics [26], which results in duplicated topics. Also, we show why we believe that using 150 topics results in the best performance statistically in Sect. 6.3.1; duplicated topics generated from larger number of topics lower the evaluation measurements. The result of many duplicated topics caused by a large number of topics is not desirable for our use case.

4.3 Analysis of users’ daily activity

Given the LDA model trained on our entire resources space, each resource selected by a user can be represented as a topic probability distribution vector over topics by fitting it into the LDA model. For each pair of the user’s active days, we calculate the Jensen–Shannon similarity, which is defined as (1-Jensen–Shannon distance), between topic probability distributions of each pair of resources that the user selected on these two days, and use the highest similarity score as the similarity measurement for activities on those two days. This can help us identify whether the user has repeated interests between two active days or whether the user will change topics periodically. The Jensen–Shannon distance (JSD) is used for measuring the similarity of two probability distributions, which is built upon the Kullback–Leibler divergence [10,13,32] that measures how one probability is different

Fig. 3 Sample of a user’s activity correlation graph. x and y coordinates represent active days of the user in 2018 and 2019 calendar years. Darker color in the graph means larger similarity between activities of days

from a second one [24,25]. The Jensen–Shannon distance between two probability distributions A and B is defined as:

$$JSD(A\|B) = \frac{1}{2}D_{KL}(A\|M) + \frac{1}{2}D_{KL}(B\|M), \text{ with } M = \frac{1}{2}(A+B) \quad (2)$$

Here D_{KL} refers to the Kullback–Leibler divergence. The Kullback–Leibler divergence for two discrete probability distributions P_1 and P_2 on the same probability space X is defined as:

$$D_{KL}(P_1\|P_2) = \sum_{x \in X} P_1(x) \log \frac{P_1(x)}{P_2(x)} \quad (3)$$

In our case, each probability distribution from the above equations corresponds to a resource’s topic probability distribution that generated by the LDA model.

The plotted results are depicted as a two-dimensional array with days on both axes. Figure 3 shows a sample output of the resources’ similarity graph. In the depiction, x and y coordinates represent the user’s active days over 2018 and 2019 calendar years, in time order. Each square represents a day that the user interacted with resources in the system. Each square in the graph is colored between white and black. Darker colors represent larger similarity between the most similar pair of resources on two active days. The diagonal represents the similarity of each day’s activities to itself and is thus always black.

Squares off the diagonal represent the similarity between the most similar pair of resources selected on those two days. Off the diagonal, black squares off the diagonal represent equality of topic probability distributions between two resources selected from two active days. Gray squares represent the most similar pair of resources on two days have similar topic probability distribution. White squares represent disjoint content over two different days. In Fig. 3, off the diagonal, the most similar pair of resources for day 1

and day 4 have the same topic probability distribution, while the most similar pair of resources for day 2 and day 3 have similar but not identical probability distributions.

Then, based upon each resource's topic probability distribution, we also draw a day-topic relationship graph on the same exact time steps for each user, where the x coordinate represents active days over 2018 and 2019 calendar years, in time order, and the y coordinate represents topics that we pre-define via an LDA model, which is 150 in our case. For each active day of a user, we examine the topic probability distribution for each resource selected by the user on that day, and for each topic, we pick the maximum value among the resources' probability distribution on that topic as the measurement for how much the user is interested in that topic on that day. This depiction shows whether a user's activities are concentrated upon one topic or multiple topics. Darker colors represent that the user is more interested in that topic.

Figure 4a, b describes a typical user's activity correlation graph and day-topic relationship graph over 150 topics generated from the original LDA model. From the graph, we can conclude that during a period of time there are some topics that the user focuses upon, which are depicted as "dashed lines" in Fig. 4b. Meanwhile, the user is also interested in some transient topics that may not be repeated in the user's future activities, which are shown as "dark squares" away from the "dashed lines" in Fig. 4b. This shows that long-term behavior might not be a good source of recommendation quality and that short-term behavior is likely a better choice.

In our data, based upon users' log in and log out activity, the "session" length is usually a day, which is similar to most of the aforementioned session-based recommendation systems. Based upon our observations in Fig. 4b, we cannot conclude that there is sequential dependency between sessions, especially for those transient interests that are varied over time. Also, unlike watching movies or TV series, a task or research project is usually not the result of one "session." A user task lasts for a period of time, and research often requires combining different kinds of resources and cross-domain knowledge. Therefore, in our work, we study users' multiple-interest behavior based upon their task-based activities in a period and periodically refresh our model to capture new interests evolved over time. Also, in our experiments and evaluation section, we do not compare the performance of our method to other session-based recommendation algorithms.

More than 90% of our users demonstrate such behavior, where their behavior exhibits multiple sets of interests in a period of time, which shows that representing a user's interests as a simple union of keywords is not sufficient to express a user's interests. This observation motivates us to apply clustering algorithm on users' activity data to summarize our users' interests into several categories based upon users' multiple-interest behavior can provide better recommendations. The reason to use clustering is to determine using sets

(a) Similarity between resources on active days for a user over 2018 and 2019 calendar years, in time order.

(b) The same user's daily interests for 150 topics generated by the LDA model trained on the entire resources space on the same active days in 2018 and 2019 calendar years. Topics are on the y axis while resources are on the x axis in time order.

Fig. 4 A typical user's daily activity correlation graph and corresponding daily topics' interests graph

of keywords representing different interests. If we have such a collection, we can recommend resources that are near to any one of the representative keyword sets. If a user's interests are consistent over a period of time, or the user is only interested in one topic, all resources selected by the user will be clustered into one interest set. Therefore, we instead apply the LDA model as a clustering algorithm on users' short-time period activities to help us identify users' interest sets.

Compared with other clustering methods, such as k -means and hierarchical clustering, we find the LDA algorithm—to be more useful. The major deficiency for k -means and hierarchical clustering algorithm is that it clusters data resources into k disjoint groups that does not allow overlaps between clusters. By contrast, the LDA algorithm assigns each data resource to a mixture of topics/clusters with a probability distribution that allows potential overlaps between distinct topics/clusters. In

addition, hierarchical clustering works better on data that demonstrates a hierarchical structure, which is not expressed in our data. On HydroShare platform, when a user works on one project, it usually requires a lot of cross-domain knowledge to support the research, where data resources from different domains may have overlapping content. Therefore, the LDA algorithm is more realistic and applicable to our data resources.

5 Method overview and design

Our approach to CBF for scientific water data users is to represent user preferences as a collection of sets of keywords for resources in which each user has expressed interest. A behavior monitor is used to store the history of user selection of data resources, including visiting the description page, downloading data or invoking processing of the data. This serves as implicit (passive) evidence of user interest in the keywords associated with the data. We then recommend resources with keywords similar to those, but which the user has not yet selected.

5.1 Latent Dirichlet allocation

The latent Dirichlet allocation (LDA) model was first introduced by Blei et al. [3]. The LDA model is widely used for identifying topics of a collection of documents. The LDA model is trained by observed words collected from a set of documents with the pre-defined number of topics. Under the LDA model, each document is represented as a mixture of multiple latent topics with a probability distribution, which describes the contribution of each topic to the document. Also, each topic also has a probability distribution over a set of words, which indicates the probability of each word being a part of that topic.

In our method, each data resource is represented by a set of keywords. Each data resource that is selected by a user is considered as a document in the LDA model, and all data resources selected by a user in a given time period serve as the input set of documents for the LDA model. Keywords extracted from resources are used for building the corpus to train the LDA model.

5.2 Build user preference profile via LDA

We build user preference profiles based upon each user's activity history. The resources that are selected by each user—for any purpose—are considered to be resources of interest for the user.

After extracting keyword sets from resources selected by each user, we apply the LDA algorithm on these keyword sets (which are equivalent to documents in the general LDA

algorithm) for each user. The LDA model not only associates each resource's keywords set with a mixture of topics, but also assigns each keyword within a topic a probability that describes the probability of that keyword being a part of that topic. We train an LDA model for each user based upon resources selected by the user. For each topic, we extract the 10 highest probability keywords as the representative keywords set for that particular topic. Each topic generated from the LDA model corresponds to one set of keywords that are potentially interesting to the user.

5.3 Making recommendations

Our CBF LDA algorithm is illustrated in Fig. 5. We apply the LDA algorithm on each user's activity data and build the preference profiles based upon LDA topic modeling results. Then, we make recommendations based upon each user's preference profile. Algorithm 1 illustrates how the recommendations are made for each user.

Algorithm 1: Making recommendations for each LDA modeled profile.

```

1  Given a user's LDA modeled profile,  $LDA_{u_i}$ ;
2   $candidates = []$ ;
3  foreach  $resource\_id\ rid, resource\_keywords\ r\_bow$  in  $SOLR\_pre-filtered\_resources\_dataset$  do
4       $similarity\_score = 0$ ;
5       $resource\_topic\_prob\_dist = LDA_{u_i}.get\_doc\_topic(r\_bow)$ ;
6       $resource\_probable\_topics =$ 
7           $find\_probable\_topics(resource\_topic\_prob\_dist)$ ;
8       $common\_topics = resource\_probable\_topics$ 
9           $.intersect(user\_probable\_topics\_set)$ ;
10     if  $common\_probable\_topics.isEmpty()$  then
11         continue;
12     foreach  $common\_topic\ t$  in  $common\_topics$  do
13          $t\_bow = LDA_{u_i}.get\_topic\_keywords(t)$ ;
14          $similarity = Jaccard\_Similarity(r\_bow, t\_bow)$ ;
15          $scaled\_similarity = similarity *$ 
16              $resource\_topic\_prob\_dist.get\_topic\_prob\_dist(t)$ ;
17          $similarity\_score += scaled\_similarity$ ;
18     end
19  $candidates.append(rid, similarity\_score)$ ;
20 end
21 return 10 most similar resources from  $candidates$ 

```

For each user, after we train an LDA model based upon the user's activity history, we obtain the topic probability distribution for each resource selected by the user, which indicates how likely the resource is assigned to each topic. Based upon this, we define a "probable topics set" in which the user is interested. As in [5,14,29], we set a threshold to be $(1/\#topics)$ for determining each resource's probable topics by selecting all topics with probability over this threshold. We use the union of all probable topics of resources selected by the user as the user's "probable topics set."

Fig. 5 An overview of our CBF LDA approach for making recommendations

As all resources have already been indexed in Apache SOLR [2]—a search engine used in the HydroShare discovery system—we use SOLR to pre-filter resources containing any keywords from the union of the user’s selected keywords. SOLR is used to decrease the runtime for identifying those resources but does not change overall results. Then, we fit each data resource in the pre-filtered set into the user’s LDA model and get its topic probability distribution, which is used for weighting the resource’s similarity to each topic keywords set of interest, and determine the “probable topics set” of this resource. If the resource does not have any topic in common with the user’s “probable topics set,” we skip this resource. Otherwise, we get a “common probable topics set” between each unselected resource and the user preferences profile, which is implemented from Line 5 to Line 9 in Algorithm 1.

The next step is to calculate the similarity between the resource keywords set and keywords set of each “common probable topic.” Several popular similarity metrics are appropriate for our case, including the Jaccard similarity and cosine similarity. Based upon our experimental results, these two similarity metrics do not generate significantly different recommended results. As we calculate a similarity score between two sets of keywords, using Jaccard similarity provides more interpretable results for our case. The Jaccard similarity between the resource keywords set R_k and the topic representative keywords set T_k is defined as:

$$\text{Jaccard Similarity}(R_k, T_k) = \frac{|R_k \cap T_k|}{|R_k \cup T_k|} \quad (4)$$

Then, the similarity score is scaled by the resource’s probability of belonging to that topic, which is assigned by the LDA model, and which is implemented on Lines 12 and 13 in Algorithm 1. Next, we sum up the resource’s similarity scores for all “common probable topics” as the similarity score of that unselected resource to the user’s preferences. In

this way, we summarize the similarity between the unselected resource and the user’s multiple sets of interests, instead of assuming users only interest in one topic as in Amami et al. [1]. Also, this can generate recommendations for users’ transient interests. Finally, the 10 highest scored resources are delivered as the recommended results.

6 Experiments and evaluation

We implement our LDA model using the Python *gensim* [35] package. To evaluate our method, we utilize detailed trace records of user behavior over two years (2018 and 2019 calendar years) of observation, which are also used for determining the number of topics for our LDA model. As we do not have specific split for “sessions” in our data, we apply the *time sliding window* methods [47,48] on our users’ activity logging data to make “sessions.” To find the best size for training data, we tried using different lengths of short-term activity data, including 7, 15, 30 and 60 days. If this window is too short, resources still of interest will be omitted, while if it is too long, resources will be included that are no longer of interest. We found using 30-day activity data for the time sliding window as training data provides the most reasonable results based upon our experiments and user behavior analysis. Thus, to evaluate our method, we randomly select 30-day activity data subsets from the two years of observation data as training datasets for each of our experiments. Also, we compare the performance of our CBF LDA method with some other well-known methods using the same dataset.

6.1 Datasets and experimental methodology

In HydroShare, resources are either private or public. Users’ interaction with both kinds of resources are monitored, but only public resources are recommended. Because our key-

words for resources are not curated, some resources have very few keywords, so we only consider “candidate resources” with more than 2 keywords as qualified for being recommended. In our dataset, we have more than 4500 candidate resources for being recommended and around 5200 distinct keywords in total. On average, each resource is marked with about 6 distinct keywords. Keywords are specified by resources’ creators and not limited to terms in controlled vocabularies.

To design the experiments, we need to determine the number of topics to use in the LDA model, and identify the users whose activity is sufficient for recommendations to be made. We will discuss each of these separately.

6.1.1 Determining the number of LDA topics

To use the *gensim* LDA model, we must determine the number of topics for training the LDA model. For our training data, we always use 30 days of history. The choice for using 30-day history data as training data size is discussed in Sect. 6.3.2. To determine the number of topics for this LDA model, we performed a similar analysis as we have described in tuning the number of topics for building the original LDA model in Sect. 4.2. Table 2 shows results for tuning the LDA model with different numbers of topics. The CS measurements are averaged over the individual measure of the coherence score for each user’s LDA model. In order to avoid duplicate results in the topic-word probability distribution results, we limit users to those who have interacted with a number of resources at least equal to the number of topics for the LDA model. These are the “candidate users” to which recommendations will be made.

As we increase the number of topics for the LDA model, the number of candidate users may decrease. Based upon our experimental results shown in column “CS” of Table 2, the largest increase in the CS is when the number of topics is increased from 4 to 5. According to the method for evaluating topic modeling results in [26], picking a number that represents the end of a rapid growth of the CS usually provides an interpretable results, which is 5 in our case. Choosing a higher value of the number of topics with higher CS results in diminishing returns. Although as we continue to increase the number of topics, it gives us a higher CS, we reduce the number of candidate users as shown in column “Average number of candidate users” of Table 2. Also, this increase of CS might also be caused by setting the number of topics too large for the dataset so that the same keywords are repeated in multiple topics [26]. Therefore, setting the number of topics to 5 provides the most desirable result when mapping 30-day user activity data. Thus, a “candidate user” in our system is defined as a user who selects at least 5 candidate resources in the given time period.

Table 2 Averaged coherence values of using different numbers of topics for training an LDA model for each user

# LDA topics	CS	Average number of candidate users
3	0.454	20
4	0.487	17
5	0.512	14
6	0.529	11
7	0.534	10
8	0.544	9
9	0.543	8
10	0.524	7

6.1.2 Selecting test data and candidate users

To test the algorithm upon real user behavior, we randomly pick 100 start dates for collecting users’ activities in the 2018 and 2019 calendar years, from which we will select users’ 30-day activity data as training dataset. To evaluate our recommended results, we compare the similarity between our recommended results and resources that specific test users select in the next time frame from the selected training dataset time. To test whether our algorithm can generate meaningful recommendations for short or long term, we first train on 30 days of history and then compare results with user behavior over the next 7-, 15- and 30-day data from the selected training dataset time frame, which is used as testing data.

We only evaluate data for users that are candidate users on training datasets and appear in both training and testing datasets. As we increase the length of time frame for selecting testing datasets, more users may be taken into consideration. Table 3 describes the datasets for each experiment on different lengths of time frame as testing datasets. In Table 3, the average number of candidate users is noted as “Avg # users”; the average number of candidate resources in each user’s training dataset is noted as “Avg # training res/user”; the average number of candidate resources in each user’s testing dataset is noted as “Avg # testing res/user”; the average number of distinct keywords in each user’s training dataset is noted as “Avg # of distinct training KW/user”; and the average number of distinct keywords in each user’s testing dataset is noted as “Avg # of distinct testing KW/user.”

6.1.3 Goodness of recommendations

When we try to recommend resources based upon users’ prior resource selections, we utilize the keywords that are specified by the resource owner to determine similarity between resources. Two resources are similar if the sets of keywords are similar. The keywords are uncured and not selected from any specific controlled vocabulary. Thus, the suitability

Table 3 Description of our experimental dataset of using different lengths of time frame as testing datasets with which to compare our predictions of interesting resources

Testing dataset	7 days	15 days	30 days
Avg # users	14	18	20
Avg # training res/user	13	13	12
Avg # testing res/user	5	7	10
Avg # of distinct training KW/user	48	46	44
Avg # distinct testing KW/user	19	25	34

measurement of whether a resource should be recommended is a function of these keywords and the user's prior browsing history. T_{KW} represents the union of all keywords from a user's testing resources, and R_{KW} represents the union of all keywords from our recommended results for a user. We apply a similar goodness measurement used in [19] for tag recommendations to compute the precision (KW Prec), recall (KW Rec) and F-measure (KW FM) for our keyword sets T_{KW} and R_{KW} . For each experiment, given U candidate users, the KW Prec, KW Rec and KW FM for each experiment are defined as:

$$KW\ Prec = \frac{1}{|U|} \sum_{u \in U} \frac{|T_{KW} \cap R_{KW}|}{|R_{KW}|} \quad (5)$$

$$KW\ Recall = \frac{1}{|U|} \sum_{u \in U} \frac{|T_{KW} \cap R_{KW}|}{|T_{KW}|} \quad (6)$$

$$KW\ FM = 2 * \frac{KW\ Prec * KW\ Recall}{KW\ Prec + KW\ Recall} \quad (7)$$

6.2 Comparison of other LDA-based and model-based recommendations

We compared the performance of our CBF LDA method with several other recommendation algorithms, including making recommendations based upon training an original LDA model on the entire resources space [22,23], building a user's preference profile as a topic-distribution vector based upon the LDA model trained from resources selected by each user [39], two model-based collaborative filtering algorithms on users' implicit feedback: the alternating least-squares model on implicit feedback [18,33] and Bayesian personalized ranking methods [36], and the recurrent neural networks method for session-based recommendations [16]. We perform each testing task on the same dataset with the same evaluation method. For each experiment, we train on users' 30-day activity data, which is randomly selected from 2018 and 2019 calendar years, and evaluate recommended results from each algorithm with resources (testing dataset) selected by users over the next 7, 15 and 30 days from the training dataset time frame.

6.2.1 Using LDA to classify data resource space

The common method for using the LDA model in recommendation system is to use it to reduce features space to a lower-dimensional topics space for data resources. As in [22,23], we implement another CBF algorithm based upon the original LDA approach. In this experiment, we use the same LDA model that we have trained for analyzing users' behavior in Sect. 4.2. Given the LDA model trained on the entire of our resources space, each resource r_j is now represented as $P(z|r_j)$, a probability distribution over each topic $z \in Z$ generated from the LDA model.

In this method, we also need to represent each user's preferences, a collection of keywords extracted from resources that the user selected, as a probability distribution over topics by the LDA model. For each user u_i , we fit keywords from resources that are selected by the user to the trained LDA model to get a probability distribution $P(z|u_i)$ for each topic $z \in Z$ generated from the LDA model. Then, we generate the top 10 recommended resources for each user by calculating cosine similarity between the user's preferences $\overrightarrow{P(Z|u_i)}$ and the topic probability distribution $\overrightarrow{P(Z|r_j)}$ of each unselected resource $r_j \in R$ in our dataset. The cosine similarity for these two probability distribution vectors is defined as:

$$\begin{aligned} \text{Cosine Similarity}(\overrightarrow{P(Z|u_i)}, \overrightarrow{P(Z|r_j)}) \\ = \frac{\sum_{k=1}^Z P(z_k|u_i) P(z_k|r_j)}{\sqrt{\sum_{k=1}^Z (P(z_k|u_i))^2} \sqrt{\sum_{k=1}^Z (P(z_k|r_j))^2}} \end{aligned} \quad (8)$$

The evaluation results for this method are labeled as "Original LDA" in Figs. 7, 8 and 9.

6.2.2 Using LDA to build a topic-distribution profile

We compared our algorithm with a modified version of the method proposed by Saraswat et al. [39], which uses the LDA model to build a topic-distribution profile for each user. In our implementation, instead of weighting each document by the rating explicitly provided by the user as described in Saraswat et al. [39], we assign a weight to each keyword based upon the user's browsing of resources. First, we apply users' 30-day activity logs to build a user-resource relation matrix M_{UR} with each row corresponds to a user and each column represents a resource. If a user selects a resource, we mark the corresponding entry in the matrix as 1, which serves as a positive implicit feedback from the user to that resource. Then, we build a resource-keyword relation matrix M_{RK} with each row represents a resource and each column represents a keyword. If a keyword is applied to a resource, we mark the corresponding entry to 1. Multiplying the M_{UR} with M_{RK} gives a user-keyword relation matrix M_{UK} with

each row represents a user and each column corresponds a keyword. Each nonzero entry r_{uk} in M_{UK} represents the implicit rating for keyword k from user u .

For each user u , first, we train an LDA model based upon resources selected by the user, which is the same training process as we have for our CBF LDA approach. Then, given the topic-word distribution Φ_u from the user u 's LDA model, $\Phi_u(z, k)$ represents the topic-word probability of topic z for keyword k . As in Saraswat et al. [39], we represent user u 's interest for topic z as $P(u, z)$, which is defined as:

$$P(u, z) = \frac{\sum_k \Phi_u(z, k) * r_{uk}}{\sum_k r_{uk}} \quad (9)$$

Then, each user u 's preferences profile is represented as a topic-distribution vector, $\{P(u, 1), P(u, 2), \dots, P(u, z)\}$, in which each entry represents the user u 's interest for one topic. Finally, for each user, we fit each unselected resource into the user's LDA model to get its topic probability distribution, and calculate the cosine similarity between the unselected resource's topic probability distribution and the user's topic-distribution preference profile vector to generate the recommended results. The evaluation results of this method are labeled as "LDA Topic Dist." in Figs. 7, 8 and 9.

6.2.3 Model-based CF implicit algorithms

We also performed experiments for testing our dataset on some popular model-based collaborative filtering SVD algorithms using implicit data. To test these matrix-based algorithms, we use the user-resource relation matrix M_{UR} that we construct for the previous test. We compare our methods with the alternating least squares (ALS) [18,33] and Bayesian personalized ranking methods (BPR) [36] from the *implicit* Python package [12] with the optimal settings for our dataset. The results of these two methods are labeled as "ALS" and "BPR" in Figs. 7, 8 and 9.

6.2.4 Recurrent neural networks for session-based recommendations

We also compared the performance of our proposed method with the state-of-the-art recurrent neural networks (RNN) algorithm for session-based recommendations [16]. We use the implementation for this algorithm from the *LibRecommender* Python package [30]. We train the model with the *cross-entropy* loss function and the Gated Recurrent Unit (GRU) [7] RNN type. In order to obtain data comparable to our other experiments, we train the model using users' 30-day activity data, considering each day's activity as one session. Then, we evaluate the predictions using the next 7-, 15- and 30-day data as testing datasets. We can only reasonably test on our users' long-term interaction; we do not have many

Table 4 Averaged evaluation measurements of using different numbers of topics for training an LDA model on 30-day activity for each user and using the next 7-day activity data as testing datasets

# LDA topics	KW Prec	KW Rec	KW FM
3	0.211	0.396	0.275
4	0.216	0.391	0.278
5	0.224	0.394	0.285
6	0.217	0.339	0.265
7	0.225	0.338	0.270
8	0.234	0.341	0.278
9	0.241	0.333	0.280
10	0.251	0.331	0.286

interactions on consecutive days. The evaluation results of this method are labeled as "SBRs RNN" in Figs. 7, 8 and 9.

6.3 Experimental results

We performed experiments both on evaluating our choices of the number of topics for our LDA models and comparing the performance of our CBF LDA algorithm with the other methods. In each experiment, we used a training dataset to generate results that were then compared against the user's actual selections of future resources, treated as testing datasets.

6.3.1 Evaluation of different numbers of topics

Instead of only using the CS to tune the number of topics for this LDA model, we also used our evaluation metrics to measure recommended results generated from each LDA model with different numbers of topics using the next 7-day activity data as the testing datasets. Based upon experimental results in Table 4, we can also conclude that setting the number of topics to 5 provide the most desirable results for our user base. Although setting the number of topics to 10 provides similar results, we undesirably reduce the number of candidate users as we have shown in Table 2. The experimental results for using the next 15- and 30-day data as testing datasets demonstrate a similar result.

Also, we performed the same evaluation task for our original LDA model as well. Table 5 shows evaluation measurements for using the next 7-day activity data as testing datasets with different numbers of topics. Based upon evaluation measurements in Table 5, we can also conclude that setting the number of topics to 150 for training the LDA model on our entire resources' space provides the most desirable results. Although setting the number of topics to 150 and 100 show similar results in the evaluation measurements, setting the number of topics to 150 results in a higher CS as shown in Table 1, which indicates a better topic modeling

Table 5 Averaged evaluation measurements of using different numbers of topics for training an original LDA model on the entire resource space and using the next 7-day activity data as testing datasets

# LDA topics	KW Prec	KW Rec	KW FM
10	0.047	0.112	0.066
50	0.084	0.176	0.114
100	0.093	0.180	0.122
150	0.094	0.183	0.124
200	0.074	0.146	0.098
250	0.065	0.127	0.086
500	0.026	0.070	0.038
600	0.028	0.051	0.036
800	0.029	0.051	0.037

Fig. 6 Precision, recall and F-measure for 100 trials of recommended resources using different sizes of randomly sampled training data for our CBF LDA model with resources selected by users in the next 7 days as testing dataset

results. The experimental results for using the next 15- and 30-day data as testing datasets demonstrate a similar result.

6.3.2 Evaluation of different training data sizes

The size of training dataset plays an important role in the performance of models. We evaluate the recommended results generated from using different sizes of training data on our model. As our method is supposed to reflect users' recent sets of distinct interests, we perform experiments on using 7 days, 15 days, 30 days and 60 days of users' history data as training data. Figure 6 shows evaluation measurements for recommended results for using the next 7 days of activity data as testing datasets with different sizes of training dataset.

The x axis represents evaluation measurements for the experiments. On the y axis, scatter error bars depict average, standard deviation, minimum and maximum scores for 100

trials. For each bar, the dot shows the mean of the experiments; the thicker part is (mean $\pm$ standard deviation); and the endpoints are min and max values for the experiments. Based upon these results, we conclude that there is not much difference among the means of measurements for using different sizes of training datasets; however, when we increase the training data size from 7 days to 30 days, there is an obvious decrease on the standard deviation, which provides more reliable results. As we continue to increase the training data size from 30 to 60 days, there is not much change in either the mean or the standard deviation. Also, increasing the amount of data used for training the model means that extra computational cost needed. As increasing the training data size from using 30 days of history data to 60 days of training data does not demonstrate a significant difference in terms of our evaluation measurements, therefore, we set the training data length to 30 days of history data for our recommendation system, which can be used to reflect our users' recent interests in a relatively short time period. The experimental results for using the next 15- and 30-day data as testing datasets demonstrate a similar pattern.

6.3.3 Comparison with other methods

In order to check whether our random sample data strategy can cause any bias for the evaluation results, we performed experiments on training the model with every single month data from January of 2018 to December of 2019. We train the model with every month's data and make recommendations for the next 7, 15 and 30 days. Figure 7 shows evaluation measurements for recommended results of each comparison method for using the next 7 days of activity data as testing datasets with the training dataset sampled from January of 2019 to December of 2019. The experimental results for sampling experimental datasets from January of 2018 to December of 2018 demonstrate a similar pattern. Also, experimental results for using the next 15- and 30-day data as testing datasets show a similar pattern. In general, our proposed CBF LDA method performs better than others.

The result of averaging these measurements is shown in Fig. 8. Our proposed CBF LDA method shows the best performance in both precision and recall, which results in the best F measure.

Then, we generalize our experiments to randomly sampled data. We randomly pick 100 start dates from 2018 and 2019 calendar years, from which we select users' 30-day activity data as training dataset. Figure 9 shows the evaluation measurements for recommended results from each method and testing data selected from using the next 7-day time frame. For each method, we generate the top 10 recommended results for comparison with testing data. Overall, our CBF LDA method achieves better results than other methods. We also performed experiments for using different training

(a) Precision

(b) Recall

(c) F Measure

Fig. 7 Precision, recall and F-measure for trials of recommended resources from each method using the calendar month data from January of 2019 to December of 2019 as training data with resources selected by users in the next 7 days as testing dataset

data sizes; the results show a similar pattern; our CBF LDA model still performs better compared to the others.

In these tests, CBF algorithms, such as our CBF LDA algorithm and the general LDA approach generate better and more reliable results than model-based CF approaches for implicit feedback data, including ALS and BPR, on our dataset. Our CBF LDA algorithm provides better results than the original

Fig. 8 Averaged precision, recall and F-measure for trials of recommended resources from each method using the calendar month data from January of 2018 to December of 2019 as training data with resources selected by users in the next 7 days as testing dataset

Fig. 9 Averaged precision, recall and F-measure for 100 trials of recommended resources from each method with resources selected by users in the next 7 days as testing dataset

LDA algorithm that represents users' interest as one single set and uses the LDA on the entire data space.

Our CBF LDA method also generates better results than the session-based RNN method. Our users' activity data does not exhibit a strong sequential dependency, which makes the session-based methods perform poorly for our case. In addition, the experimental setup in [16] filters out clicks from test dataset where the item click is not in the training dataset. Our proposed approach instead takes all candidate resources into consideration for making recommendations.

CBF approaches provide a workable solution for making personalized recommended results for our specific use case. In addition, these experiments also periodically update the model when new interest data is introduced. Each experiment

for randomly picking a starting date can be used to represent the process of introducing new interest data to each user and refreshing each testing model based upon the new data. Based upon our experimental results, we can also conclude that our CBF LDA works best compared with performing a frequent model refresh for other methods. The experimental results for using the next 15- and 30-day data as testing datasets demonstrate similar results.

7 Conclusions and future work

In this work, we propose a workable solution for implementing recommendations systems for water data. Based upon scientific water data users' behavior, we design and discuss a CBF LDA method for making recommendations. We evaluate its performance by comparing the recommended result with users' selections in the next time frames from the time stamp for picking the training dataset.

We plan to release our recommendation systems to the HydroShare platform and collect feedback from real users. We also plan to assign different weights to users' different actions on resources, which may improve the accuracy of implicit feedback. We also plan to add more keywords to our resources' keywords space, which will lend more challenges for us to measure similarities between resources, especially for hydrologic data that have a large space of geospatial features. This will raise another challenge for us, which is about how to reduce computation cost while preserving acceptable precision and recall.

Acknowledgements This work was supported by the National Science Foundation under collaborative Grants 1148453, 1664061 and 1849458 for the development of HydroShare. Any opinions, findings, conclusions or recommendations expressed are those of the authors and do not necessarily reflect the views of the NSF. We thank our colleagues and developers from the HydroShare project and from CUAHSI, who provided efforts and expertise that greatly assisted the work. We also thank Professor Liping Liu for helping review and giving advice to our paper.

Declarations

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the

permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

1. Amami, M., Pasi, G., Stella, F., Faiz, R.: An lda-based approach to scientific paper recommendation. In: International conference on applications of natural language to information systems, pp. 200–210. Springer (2016)
2. Solr is the popular, blazing fast, open source enterprise search platform built on apache lucene (2017). <https://lucene.apache.org/solr>. Accessed 2019-04-16
3. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. *J. Mach. Learn. Res.* **3**, 993–1022 (2003)
4. Bui, Q.V., Sayadi, K., Amor, S.B., Bui, M.: Combining latent dirichlet allocation and k-means for documents clustering: effect of probabilistic based distance measures. In: Asian Conference on Intelligent Information and Database Systems, pp. 248–257. Springer (2017)
5. Bui, Q.V., Sayadi, K., Bui, M.: A multi-criteria document clustering method based on topic modeling and pseudoclosure function. In: Proceedings of the sixth international symposium on information and communication technology, pp. 38–45 (2015)
6. Chen, P.Y., Wu, S.Y.: Does collaborative filtering technology impact sales? empirical evidence from amazon. com. *Empirical Evidence from Amazon. Com* (July 8, 2007) (2007). <https://ssrn.com/abstract=1002698>
7. Cho, K., Van Merriënboer, B., Bahdanau, D., Bengio, Y.: On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259* (2014)
8. CSDMS: Citations—csdmsStandard Names (2019). <https://csdms.colorado.edu/>
9. Dublin core metadata element set, version 1.1 (2017). <http://dublincore.org/documents/dces/>. Accessed 2017-05-30
10. Endres, D.M., Schindelin, J.E.: A new metric for probability distributions. *IEEE Trans. Inform. Theo.* **49**(7), 1858–1860 (2003)
11. Fang, Y., Si, L.: Matrix co-factorization for recommendation with rich side information and implicit feedback. In: Proceedings of the 2nd International Workshop on Information Heterogeneity and Fusion in Recommender Systems, pp. 65–69. ACM (2011)
12. Frederickson, B.: Fast python collaborative filtering for implicit feedback datasets (2019). <https://github.com/benfred/implicit>
13. Fuglede, B., Topsoe, F.: Jensen-shannon divergence and hilbert space embedding. In: Proceedings of the international symposium on information theory, 2004. ISIT 2004, p. 31. IEEE (2004)
14. Hariri, N., Mobasher, B., Burke, R.: Context-aware music recommendation based on latent topic sequential patterns. In: Proceedings of the sixth ACM conference on Recommender systems, pp. 131–138 (2012)
15. Hidasi, B., Karatzoglou, A.: Recurrent neural networks with top-k gains for session-based recommendations. In: Proceedings of the 27th ACM international conference on information and knowledge management, pp. 843–852 (2018)
16. Hidasi, B., Karatzoglou, A., Baltrunas, L., Tikk, D.: Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939* (2015)
17. Horsburgh, J.S., Aufdenkampe, A.K., Mayorga, E., Lehnert, K.A., Hsu, L., Song, L., Jones, A.S., Damiano, S.G., Tarboton, D.G., Valentine, D., et al.: Observations data model 2: a community information model for spatially discrete earth observations. *Environ. Model. Softw.* **79**, 55–74 (2016)

18. Hu, Y., Koren, Y., Volinsky, C.: Collaborative filtering for implicit feedback datasets. In: Proceedings of the 2008 eighth IEEE international conference on data mining, pp. 263–272. IEEE (2008)
19. Jäschke, R., Marinho, L., Hotho, A., Schmidt-Thieme, L., Stumme, G.: Tag recommendations in folksonomies. In: European conference on principles of data mining and knowledge discovery, pp. 506–514. Springer (2007)
20. Kapadia, S.: Evaluate topic models: Latent dirichlet allocation (lda) (2019). <https://towardsdatascience.com/evaluate-topic-model-in-python-latent-dirichlet-allocation-lda-7d57484bb5d0>
21. Kelly, D., Teevan, J.: Implicit feedback for inferring user preference: a bibliography. SIGIR Forum **37**, 18–28 (2003)
22. Krestel, R., Fankhauser, P., Nejdl, W.: Latent dirichlet allocation for tag recommendation. In: Proceedings of the third ACM conference on recommender systems, pp. 61–68 (2009)
23. Kuang, W., Luo, N., Sun, Z.: Resource recommendation based on topic model for educational system. In: Proceedings of the 2011 6th IEEE joint international information technology and artificial intelligence conference, vol. 2, pp. 370–374. IEEE (2011)
24. Kullback, S.: Information theory and statistics. Courier Corporation (1997)
25. Kullback, S., Leibler, R.A.: On information and sufficiency. Ann. Math. Stat. **22**(1), 79–86 (1951)
26. Kumar, K.: Evaluation of topic modeling: topic coherence (2018). <https://datascienceplus.com/evaluation-of-topic-modeling-topic-coherence/>
27. Li, Y., Hu, J., Zhai, C., Chen, Y.: Improving one-class collaborative filtering by incorporating rich user information. In: Proceedings of the 19th ACM international conference on information and knowledge management, pp. 959–968. ACM (2010)
28. Lin, W., Alvarez, S.A., Ruiz, C.: Efficient adaptive-support association rule mining for recommender systems. Data Min. Knowl. Discov. **6**(1), 83–105 (2002)
29. Lu, Y., Mei, Q., Zhai, C.: Investigating task performance of probabilistic topic models: an empirical study of plsa and lda. Inform. Retr. **14**(2), 178–203 (2011)
30. massquantity: Librec recommender. <https://github.com/massquantity/LibRecommender> (2021)
31. Mobasher, B., Dai, H., Luo, T., Nakagawa, M.: Effective personalization based on association rule discovery from web usage data. In: Proceedings of the 3rd international workshop on Web information and data management, pp. 9–15 (2001)
32. Osterreicher, F., Vajda, I.: A new class of metric divergences on probability spaces and its applicability in statistics. Ann. Inst. Stat. Math. **55**(3), 639–653 (2003)
33. Pan, R., Zhou, Y., Cao, B., Liu, N.N., Lukose, R., Scholz, M., Yang, Q.: One-class collaborative filtering. In: Proceeding of the 2008 eighth IEEE international conference on data mining, pp. 502–511. IEEE (2008)
34. Quadrana, M., Karatzoglou, A., Hidasi, B., Cremonesi, P.: Personalizing session-based recommendations with hierarchical recurrent neural networks. In: Proceedings of the eleventh ACM conference on recommender systems, pp. 130–137 (2017)
35. Řehůřek, R., Sojka, P.: Software framework for topic modelling with large corpora. In: Proceedings of the LREC 2010 workshop on new challenges for NLP frameworks, pp. 45–50. ELRA, Valletta, Malta (2010). <http://is.muni.cz/publication/884893/en>
36. Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-Thieme, L.: Bpr: Bayesian personalized ranking from implicit feedback. In: Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence, pp. 452–461. AUAI Press (2009)
37. Röder, M., Both, A., Hinneburg, A.: Exploring the space of topic coherence measures. In: Proceedings of the eighth ACM international conference on Web search and data mining, pp. 399–408 (2015)
38. Rosen-Zvi, M., Griffiths, T., Steyvers, M., Smyth, P.: The author-topic model for authors and documents. arXiv preprint [arXiv:1207.4169](https://arxiv.org/abs/1207.4169) (2012)
39. Saraswat, M., Chakraverty, S., Sharma, A.: Based topic distribution profile for recommender systems. In: Advances in data sciences, security and applications, pp. 433–443. Springer (2020)
40. Schafer, J.B., Konstan, J., Riedl, J.: Recommender systems in e-commerce. In: Proceedings of the 1st ACM conference on electronic commerce, pp. 158–166. ACM (1999)
41. Shani, G., Heckerman, D., Brafman, R.I.: An mdp-based recommender system. J. Mach. Learn. Res. **6**, 1265–1295 (2005)
42. Sindhvani, V., Bucak, S.S., Hu, J., Mojsilovic, A.: One-class matrix completion with low-density factorizations. In: Proceedings of the 2010 IEEE international conference on data mining, pp. 1055–1060. IEEE (2010)
43. Syed, S., Spruit, M.: Full-text or abstract? examining topic coherence scores using latent dirichlet allocation. In: Proceedings of the 2017 IEEE international conference on data science and advanced analytics (DSAA), pp. 165–174. IEEE (2017)
44. Tan, Y.K., Xu, X., Liu, Y.: Improved recurrent neural networks for session-based recommendations. In: Proceedings of the 1st workshop on deep learning for recommender systems, pp. 17–22 (2016)
45. Tarboton, D.G., Idaszak, R., Horsburgh, J.S., Heard, J., Ames, D., Goodall, J.L., Band, L., Merwade, V., Couch, A., Arrigo, J., et al.: Hydroshare: advancing collaboration through hydrologic data and model sharing (2014)
46. Wang, S., Cao, L.: Inferring implicit rules by learning explicit and hidden item dependency. IEEE Trans. Syst. Man Cybern. Syst. (2017)
47. Wang, S., Cao, L., Wang, Y.: A survey on session-based recommender systems. arXiv preprint [arXiv:1902.04864](https://arxiv.org/abs/1902.04864) (2019)
48. Wang, S., Hu, L., Cao, L., Huang, X., Lian, D., Liu, W.: Attention-based transactional context embedding for next-item recommendation. In: AAAI, pp. 2532–2539 (2018)
49. Wu, X., Liu, Q., Chen, E., He, L., Lv, J., Cao, C., Hu, G.: Personalized next-song recommendation in online karaokes. In: Proceedings of the 7th ACM conference on Recommender systems, pp. 137–140 (2013)
50. Xue, Z., Couch, A., Tarboton, D.: Map based discovery of hydrologic data in the hydroshare collaboration environment. Environ. Model. Softw. **111**, 24–33 (2019)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.