Hybrid Parallel Approach for Personalized Literature Recommendation System

Kun Ma*, Tingting Lu*, Ajith Abraham†‡
*Shandong Provincial Key Laboratory of Network Based Intelligent Computing, University of Jinan, Jinan, China
ise_mak@ujn.edu.cn, lutingting_ujn@163.com
†Machine Intelligence Research Labs, Scientific Network for Innovation and Research Excellence, Auburn, USA
‡IT4Innovations - Center of excellence, VSB - Technical University of Ostrava, Czech Republic
ajith.abraham@ieee.org

***Abstract*—Researchers regularly access and review large amounts of literatures. In the previous work, we presented a bookmarklet-triggered literature sharing system, which combines bibliography functionalities along with DOI content negotiation services. In this paper, we have made secondary development work to integrate literature recommendation functionalities into this system. We introduce a hybrid approach in parallel to recommend related articles to researchers. First, we collect a large amount of published and new articles using crawlers and RSS listeners to address cold start issue. Second, we adopt Latent Dirichlet Allocation (LDA) as the topic model to category literatures. For one kind of literatures related to researchers' interest, we use collaborative filtering techniques to make further analysis based on implicit user feedbacks in this system. Finally, we take matrix factorization with Alternating Least Squares (ALS) in parallel to compute the top-N recommendations per user.**

***Index Terms*—Literature recommendation; collaborative filtering; topic model; matrix factorization; MapReduce**

I. Introduction

These days, many academic literatures are coming out from a lot of conferences and journals. Modern researchers have access to large archives of scientific literatures in the publishing system. These archives are growing as new articles placed online. Although this growth has allowed researchers to quickly access more scientific information, it has also made it more difficult for them to find articles relevant to their interests [1]. Some new integrated systems provide a simple way to broadly search for scholarly literature, such as Google scholars, Microsoft Libra Academic, CiteSeer, CrossRef search and DBLP. At one extreme, you can search across many disciplines and sources from academic publishers. However, these methods require researchers to spend their time in searching articles, which is labor-intensive, and also do not guarantee that they will find exact articles especially for the beginners.

In order to reduce their workload, recommendation systems for literatures are becoming increasingly popular. It is a useful application for researchers to automatically detect their research topics, and recommend related articles they might be interested in based on collaborative recommendation algorithm [2] [3] [4]. The larger the amount of recommendation systems is, the more important this system is. One evaluation criterion of this system is determined by user-friendly interpolation. That is to say that this system will automatically push literatures without too much workload.

In this paper, we aim to architect a novel recommendation system, which differs from traditional recommendation systems. Researchers do not even enter or import the literature list, since it only gathers user behaviors and activities on literatures to make further recommendation, and pushes any new and historical articles related to their current works. The researcher is identified by the unique researcher ID, such as ORCID and Scopus Author ID. This system will save researchers' time to search articles and increase the accuracy of finding articles for beginners.

Among recommendation methods, content-based approaches analyze the content of items to identify related items [5] [6], while collaborative filtering [7] [8] uses the aggregated behaviors of a large number of users to suggest relevant items to specific user. Literature recommendation is a very significant research area newly developed along with recommendation in the area of e-commerce. On one hand, large-scale literatures are as the input for the recommendation in the context of literature recommendation. On the other hand, the sparsity of implicit user feedbacks becomes more serious. Therefore, a parallel recommendation approach becomes a pressing need.

In our paper, we introduce a hybrid approach to recommend personalized literature. This method is implemented with MapReduce [9] framework in parallel. On this basis, we design and implement the literature recommendation system with browser plugins. The contributions of this paper are several folds. First, content-based and user-based recommendation algorithms are combined together to address classification and recommendation issues. Second, our collaborative recommendation approach is simple and scales well to very large data sets with parallel framework. Third, implicit user feedbacks are collected when researchers view or comment the literature publisher pages using a lightweight browser plugin. Compared with current recommendation systems, it is general to heterogeneous publishers to enable behavior and activity gathering in an unobtrusive way. Last, this approach works inside a browser instead of entering the input interface.

The rest of the paper is organized as follows. Section II discusses the related work, and section III presents the requirements of our personalized literature recommendation

978-1-4799-5940-2/14/$31.00 ©2014 IEEE

system in parallel. In section IV, the recommendation method is presented in detail. Section V shows the user interface of the prototype system. Brief conclusions are outlined in the last section.

II. Related Work

A. Recommendation Methods

Recommendation approaches are mainly categorized into content-based methods [5] [6], collaborative filtering [7] [8], and hybrid methods [10] [11] in both academia and industry. First, content-based recommendation systems recommend items based on their characteristics as well as specific preferences of a user. Collaborative filtering, on the other hand, involves aggregated behavior/taste of massive users to suggest relevant items to specific user. Applications of collaborative filtering typically involve very large data sets. Recommendations generated by collaborative filtering are based solely on the user-user and/or item-item similarities. Recently, making use of matrix factorization [12] [13], a kind of model-based approach, is known as most efficient and accurate, especially after those approaches have won the Netflix prize. Last category, hybrid approach, tries to combine both content-based approach and collaborative filtering. Koren suggested effectively combining rating information and user, item profiles for more accurate recommendation [14].

B. Literature Recommendation Methods

With the propagation of academic conferences and journals, and wide range of available literatures, it becomes more difficult for researchers to easily locate resources. Indeed, the number of scientific articles published in international recognized peer-reviewed scientific and engineering journals covered by the CrossRef is over 67 million in 2014. Hence various attempts at using recommendation techniques to help researchers locate suitable research materials.

Recommender systems have concentrated on recommending media items such as movies and products, but recently they are extending to academy. Most popular application is citation recommendation. He et al. produced a successful recommendation system by examining the relevance between segments in a query manuscript and the representative segments extracted from a document corpus [15]. Lee et al. proposed a personalized academic research paper recommendation system, which recommends related articles for each researcher [2].

C. Recommendation Systems in Parallel

MapReduce is a framework originally developed at Google that allows easy large scale distributed computing across a number of domains [9]. It scales well to many thousands of nodes to handle big data. For recommendations where we have to find similar literatures to a literature you are interested in, we must calculate how similar pairs of items are. Besides the computation, the correlation data will be sparse, because it is unlikely that each pair of items will have some users who are interested in them. Thus, we have a large and sparse dataset. And we have also to deal with the temporal aspect since the user interest in products changes with time, so we need the correlation calculation done periodically so that the results are up to date. For these reasons, the best way to handle with this scenarion and problem is going after a divide and conquer strategy, and MapReduce is a powerful framework to be used to implement recommendation tasks.

Mahout's item based collaborative filtering is a flexible and easily implemented algorithm with a diverse range of applications [16]. The minimalism of the primary input file's structure and availability of ancillary filtering controls can make sourcing required data and shaping a desired output both efficient and straightforward. Mahout's Alternating Least Squares (ALS) recommender is a matrix factorization algorithm that uses ALS with Weighted-Lamda-Regularization (ALS-WR) [17]. It factors the user-to-item matrix A into the user-to-feature matrix U and the item-to-feature matrix M, which is running in a parallel manner. It is shown empirically that the performance of ALS-WR monotonically improves with both the number of features and the number of iterations.

Unlike the much more extensively researched explicit ratings, feedback is often in the implicit way, such as browsing activity and user behavior. Some experts proposed treating the data as indication of positive and negative preference associated with vastly varying confidence levels. They identify unique proper ties of implicit feedback data sets [18].

III. Requirements

We have the following requirements while designing the personalized literature recommendation system.

- Computing of implicit user preferences: Except user ratings, we use multi-dimensional implicit user feedbacks as the reference, such as clicking count, commenting count, and concern level.
- Acquiring user preference automatically: We want to acquiring user feedbacks by providing gathering services in the daily process of searching, viewing and commenting literature publisher pages with user permission.
- Addressing cold start issue: The introduction of new users or new literatures can cause the cold start problem, as there will be insufficient data on these new entries for the collaborative filtering to work accurately. In our solution, we provide several ways to solve this problem, such as acquiring user feedbacks by the browser plugin and interest input in question & answer manner.
- Large-scale parallel computing: Due to large and sparse data set, we have to handle with this problem after a divide and conquer pattern in strategy.

IV. Methodology

A. Literature Source

First, we have made some efforts to develop literature acquisition services. The class diagram of the crawler and RSS listener are shown in Figure 1. Crawler is categorized to capture historical articles, and $RSSExtractor$ and $LatestParser$ are categorized to capture latest published/online articles.

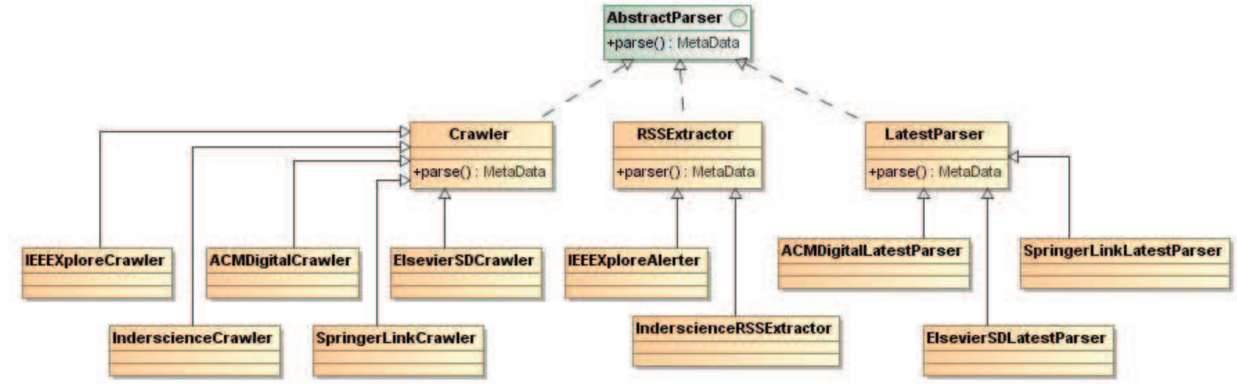


Fig. 1. Class diagram of crawlers and RSS listeners.

- For the historical articles, we develop robotic crawlers to parse literature web pages of different publishers. In order to make crawlers more general, we design different template-based parsers for different page styles. For example, we provide Xplore template to parse literature metadata in IEEE Xplore, provide ACM template to parse literature metadata in ACM Digital Library, and provide ScienceDirect template to parse literature metadata in Elsevier database.
- For the new published or online articles, we develop RSS listener to subscribe the alert service of latest articles, and extract the metadata to save in our literature database. For example, Inderscience publisher provides anonymous RSS interface of latest articles to capture the changes of literature database, and IEEE Xplore provides RSS alert services to registered members.

B. Topic Acquisition

Next, we introduce a topic modeling approach to categorize the literatures for further collaborative filtering. Topic modeling algorithms have been widely used for tasks like corpus exploration, content classification, and information retrieval. They provide an interpretable low-dimensional representation of the contents. As for literature recommendation, the contents come from the metadata (title, keyword and abstract) of the literature. Thus, we will exploit the discovered topic structure for recommendation. We introduce a topic modeling algorithm to find a set of topics from a large collection of literatures, where a topic is a distribution over terms under a single theme.

We adopt Latent Dirichlet Allocation (LDA) [19] as the topic model. Assume there are K topics, denoted as matrix β, each of which is a distribution over a fixed vocabulary. The generative process of LDA is as follows. For each article a_j in the literatures,

1) Draw topic proportions $\theta_j \approx$ Dirichlet(α).
2) For each word n, draw topic assignment $z_{jn} \approx$ Mult(θ_j), and draw word $a_{jn} \approx$ Mult($\beta_{z_{jn}}$).

This process reveals how the words of each literature are assumed to come from a mixture of topics: the topic proportions are content specific, but the set of topics is shared. Given a collection of literatures, the posterior distribution of the topics reveals the Ktopics that likely generated its literature contents. Unlike a clustering model, where each literature is assigned to one cluster, LDA allows literatures to exhibit multiple topics. Given a corpus of literatures, we can use variational EM [20] to learn the topics and decompose the literatures. Further, given a new literature, we use variational inference to situate its content in terms of the topics. Our goal is to use topic modeling to give a content-based representation of items in a recommendation system. The output of topic acquisition is several topics of literatures, which are also considered as tags of each literature. Next, we category the literatures by the topics.

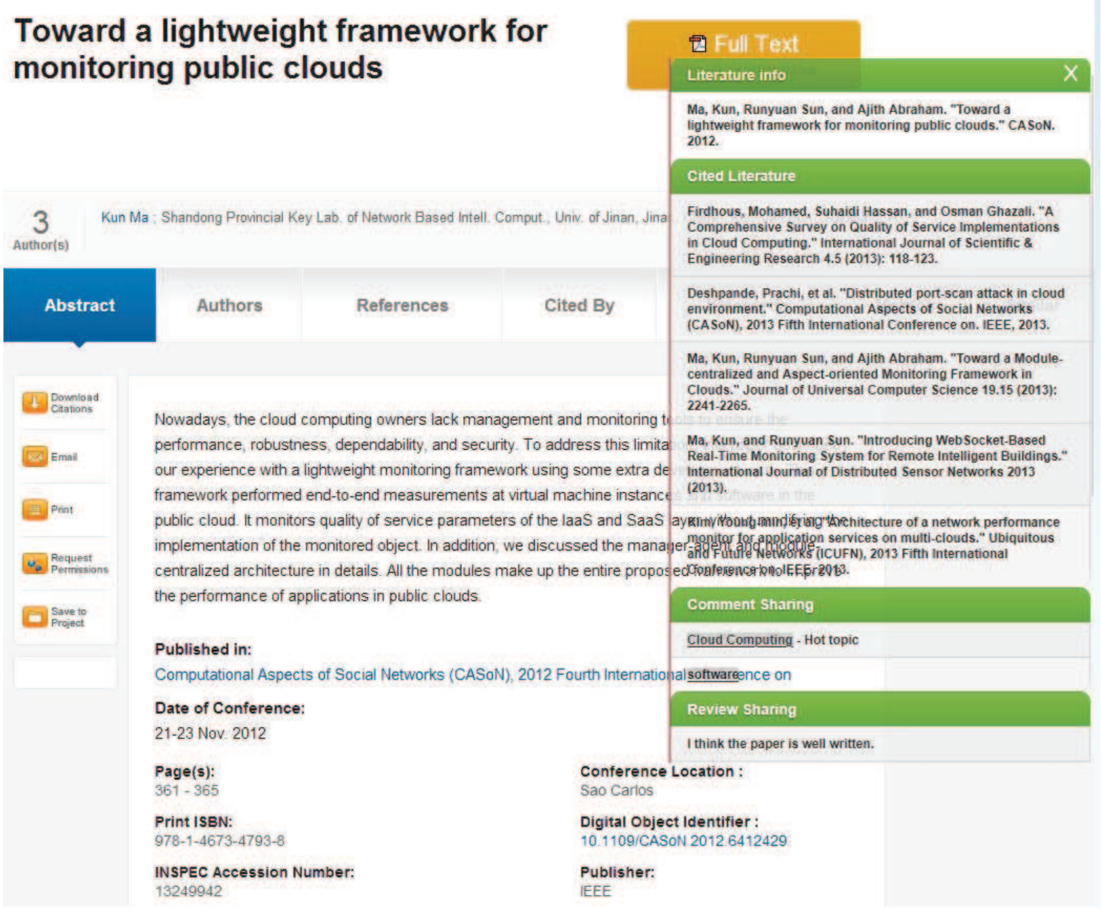
C. Implicit User Feedback Acquisition

First, we have made some efforts to develop a Chrome browser plugin to record user behaviors on viewing literature. All the literature release pages in the publishers that researchers access are firstly recorded by this plugin in the client, and then the literature metadata and concrete behaviors are sent to the server for further analysis with user permission. Clicking count, commenting count, remarking count are all considered as implicit user feedbacks, and concern level and concrete rating are all considered as explicit user ratings.

In our previous work [21], we design a bookmarklet-triggered literature sharing system. In this paper, we have done secondary development work to capture user behaviors using this browser independent tool. For instance, when someone cites, comments or remarks on the literature, we quantize this behavior as a kind of generalized user feedbacks.

Figure 2 shows the user interface of our bookmarklet-triggered literature sharing system. By this tool, we can gather implicit user feedbacks of literature that they are interested in. First, the DOI is extracted in the literature web page of the publisher, and then the metadata is extracted using DOI content negotiation service [22]. Next, some feedbacks such as commenting and sharing behaviors are recorded for further analysis.

Next, user preferences are calculated in the server. We reserve special indexing letters for distinguishing users from



Toward a lightweight framework for monitoring public clouds

3 Author(s)

Kun Ma ; Shandong Provincial Key Lab. of Network Based Intell. Comput., Univ. of Jinan, Jina

Nowadays, the cloud computing owners lack management and monitoring t
performance, robustness, dependability, and security. To address this limita
our experience with a lightweight monitoring framework using some extra de
framework performed end-to-end measurements at virtual machine instance
public cloud. It monitors quality of service parameters of the IaaS and SaaS lay
implementation of the monitored object. In addition, we discussed the manager-
centralized architecture in details. All the modules make up the entire proposed
the performance of applications in public clouds.

Published in:
Computational Aspects of Social Networks (CASoN), 2012 Fourth International Conference on

Date of Conference:
21-23 Nov. 2012

Page(s):
361 - 365

Print ISBN:
978-1-4673-4793-8

INSPEC Accession Number:
13249942

Conference Location :
Sao Carlos

Digital Object Identifier :
10.1109/CASoN.2012.6412429

Publisher:
IEEE

Literature info

Ma, Kun, Runyuan Sun, and Ajith Abraham. "Toward a lightweight framework for monitoring public clouds." CASoN. 2012.

Cited Literature

Firdhous, Mohamed, Suhaidi Hassan, and Osman Ghazali. "A Comprehensive Survey on Quality of Service Implementations in Cloud Computing." International Journal of Scientific & Engineering Research 4.5 (2013): 118-123.

Deshpande, Prachi, et al. "Distributed port-scan attack in cloud environment." Computational Aspects of Social Networks (CASoN), 2013 Fifth International Conference on. IEEE, 2013.

Ma, Kun, Runyuan Sun, and Ajith Abraham. "Toward a Module-centralized and Aspect-oriented Monitoring Framework in Clouds." Journal of Universal Computer Science 19.15 (2013): 2241-2265.

Ma, Kun, and Runyuan Sun. "Introducing WebSocket-Based Real-Time Monitoring System for Remote Intelligent Buildings." International Journal of Distributed Sensor Networks 2013 (2013).

Kim, Young-min, et al. "Architecture of a network performance monitor for application services on multi-clouds." Ubiquitous and Future Networks (ICUFN), 2013 Fifth International Conference on. IEEE, 2013.

Comment Sharing

Cloud Computing - Hot topic

software

Review Sharing

I think the paper is well written.

Fig. 2. Bookmarklet-triggered literature sharing system.

literatures: for user u, v, and for literature i, j. The input data associate users and literatures through $r_{u,i}$ values, which we henceforth call observations. Considering different aspects that affect user preferences, $r_{u,i} = \sum_{k=1}^{n}(\lambda_k \times r_{k,u,i})$, where $r_{k,u,i}$ means observations in the aspect k. For implicit feedbacks, those values would indicate observations for user behaviors. For example, $r_{a,u,i}$ indicates the number of times, where aspect a is that user u visited literature i; $r_{b,u,i}$ indicates the time, where aspect b is that user u spent on web page of literature i. In our literature recommendation case, $r_{u,i}$ indicates how many times u clicking, commenting, citing or remarking the web page of literature i, or the co-authorship with literature i.

Next, we formalize the notion of confidence which the $r_{u,i}$ variables measure. We introduce a set of variables, $p_{u,i}$, which measure user preference (user u to literature i). A plausible choice for $p_{u,i}$ would be $p_{u,i} = 1 + \alpha \times r_{u,i}$. The rate of increase of user preference $p_{u,i}$ is controlled by the constant α.

D. Matrix Factorization with Alternating Least Squares in Parallel

We introduce a matrix factorization algorithm that uses Alternating Least Squares with Weighted Lamda Regularization (ALS-WR) [23]. It factors the researcher to literature matrix A into the researcher-to-feature matrix U and the literature-to-feature matrix M, denoted as $A = U \times M$. This recommendation algorithm is used to recommend literatures to researchers in a parallel manner. Unlike the user or item based recommenders that computes the similarity of users or items to make recommendations, this algorithm uncovers the latent factors that explain the observed user to literature ratings and tries to find optimal factor weights to minimize the least squares between predicted and actual ratings. This recommendation algorithm takes $p_{u,i}$ as user preferences by literature i and generates an output of recommending items for a user u. The input of user preference is implicit feedback computed in the last section. Compared user or item based collaborative filtering, the strength of this algorithm is its ability to handle large sparse data sets and its better prediction performance. It could also gives an intuitive rationale of the factors that influence recommendations. We have made a MapReduce implementation of this algorithm, which is composed of two jobs: a parallel matrix factorization job and a recommendation job.

1) Parallel Matrix Factorization: The matrix factorization (MF) job computes the user-to-feature matrix U and literature-to-feature matrix M given implicit user to literature feedbacks. The input is a set of researcher to literature preference data: researcherID, literatureID and preference. We use the implicit feedback $p_{u,i}$ as the preference. It outputs the matrices in sequence file format. The authentication of the researcher is determined by the unique researcher ID, such as ORCID and Scopus Author ID.

To use matrix factorization, we must compute the latent representations of the users and literatures given an observed matrix of feedbacks. The common approach is to minimize the regularized squared error loss with respect to U and M, $min(\sum(r_{i,j} - u_i \times m_j)^2 + \lambda_u||u_i||^2 + \lambda_m||m_j||^2)$, where λ_u and λ_m are regularization parameters.

We parallelize this algorithm by parallelizing the updates of U and of M with MapReduce. The function is called $parallelMF$, which is divided into three MapReduce jobs: $itemRatings$ Job (computing M), $userRatings$ Job (computing U), and $averageRatings$ Job.

Before running the algorithm, dimension of feature space, the number of iterations to run the MF algorithm, and a confidence parameter are determined. Using 10 features and 15 iterations is a reasonable default.

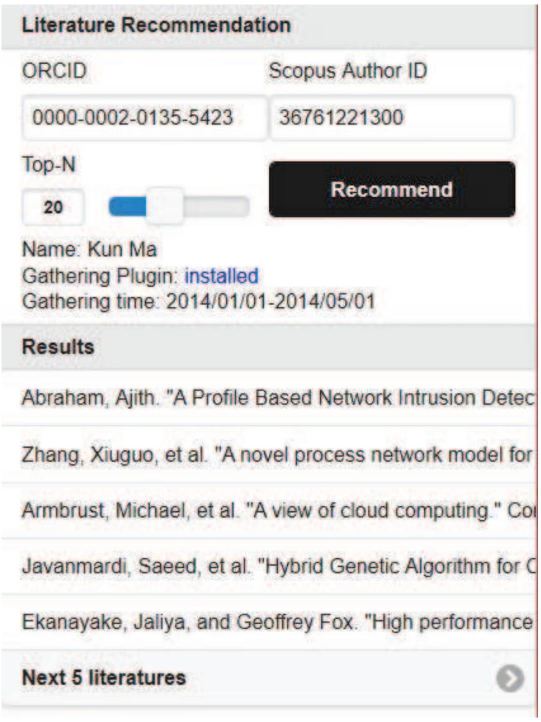
2) Make Recommendations: Based on the output feature matrices, we could make recommendations for researchers. The recommendation job uses the researcher feature matrix and literature feature matrix calculated from the factorization job to compute the top-N recommendations per user. It outputs a list of recommended literature ids for each user. The predicted rating between user and literature is a dot product of the researcher's feature vector and the literature's feature vector.

V. Evaluation

Our literature recommendation prototype system provides simple Graphical User Interface (GUI), which is shown in Figure 3. It requires researcher ID (ORCID or Scopus Author ID) and the number of literatures the user wants to be recommended. The user feedbacks are collected by the browser plugins. After the computation of topic modeling and matrix factorization with MapReduce, top-N literatures are displayed in the interface.

VI. Conclusions

Literature recommendation involves two aspects: content-based methods and collaborative filtering. However, existing solutions do not properly address the cold start and big data bottleneck issue. The main focus of this article is to combine item-based and user-based recommendation to propose a hybrid parallel approach. This system firstly gathers the published articles of different publishers to form the initial literature database, and listens different RSS sources and alerts to supplement the literature database. Next, We adopt hybrid topic modeling and matrix factorization to compute the top-N recommendations per user. Finally, we recommend these results to the researchers.



Literature Recommendation

ORCID: 0000-0002-0135-5423

Scopus Author ID: 36761221300

Top-N: 20

Name: Kun Ma

Gathering Plugin: installed

Gathering time: 2014/01/01-2014/05/01

Results

Abraham, Ajith. "A Profile Based Network Intrusion Detec

Zhang, Xiuguo, et al. "A novel process network model for

Armbrust, Michael, et al. "A view of cloud computing." Co

Javanmardi, Saeed, et al. "Hybrid Genetic Algorithm for C

Ekanayake, Jaliya, and Geoffrey Fox. "High performance

Fig. 3. Literature recommendation prototype system.

Acknowledgment

This work was supported by the Doctoral Fund of University of Jinan (XBS1237), and the Teaching Research Project of University of Jinan (J1344). Ajith Abraham acknowledges the support from IT4Innovations Centre of Excellence project, reg. no. CZ.1.05/1.1.00/02.0070 funded by Structural Funds of the European Union and state budget of the Czech Republic.

References

[1] D. L. Lorenzetti and W. A. Ghali, "Reference management software for systematic reviews and meta-analyses: an exploration of usage and usability," *BMC medical research methodology*, vol. 13, no. 1, p. 141, 2013.

[2] J. Lee, K. Lee, and J. G. Kim, "Personalized academic research paper recommendation system," *arXiv preprint arXiv:1304.5457*, 2013.

[3] A. Naak, H. Hage, and E. Aïmeur, "Papyres: A research paper management system," in *2008 10th IEEE Conference on E-Commerce Technology and the Fifth IEEE Conference on Enterprise Computing, E-Commerce and E-Services*. IEEE, 2008, pp. 201–208.

[4] A. Naak, H. Hage, and E. Aimeur, "A multi-criteria collaborative filtering approach for research paper recommendation in papyres," in *E-Technologies: Innovation in an Open World*, 2009, pp. 25–39.

[5] P. Lops, M. de Gemmis, and G. Semeraro, "Content-based recommender systems: State of the art and trends," in *Recommender Systems Handbook*. Springer, 2011, pp. 73–105.

[6] C.-J. Lin, T.-T. Kuo, and S.-D. Lin, "A content-based matrix factorization model for recipe recommendation," in *Advances in Knowledge Discovery and Data Mining*. Springer, 2014, pp. 560–571.

[7] Y. Koren and R. Bell, "Advances in collaborative filtering," in *Recommender Systems Handbook*. Springer, 2011, pp. 145–186.

[8] J. Bobadilla, F. Ortega, A. Hernando, and J. Bernal, “A collaborative filtering approach to mitigate the new user cold start problem,” *Knowledge-Based Systems*, vol. 26, pp. 225–238, 2012.

[9] J. Dean and S. Ghemawat, “Mapreduce: a flexible data processing tool,” *Communications of the ACM*, vol. 53, no. 1, pp. 72–77, 2010.

[10] S. Yoshizaki, Y. Yoshitomi, C. Koro, and T. Asada, “Music recommendation hybrid system for improving recognition ability using collaborative filtering and impression words,” *Artificial Life and Robotics*, vol. 18, no. 1-2, pp. 109–116, 2013.

[11] X. Chen, X. Liu, Z. Huang, and H. Sun, “Regionknn: A scalable hybrid collaborative filtering algorithm for personalized web service recommendation,” in *2010 IEEE International Conference on Web Services (ICWS)*. IEEE, 2010, pp. 9–16.

[12] L. Baltrunas, B. Ludwig, and F. Ricci, “Matrix factorization techniques for context aware recommendation,” in *Proceedings of the fifth ACM conference on Recommender systems*. ACM, 2011, pp. 301–304.

[13] R. Gemulla, E. Nijkamp, P. J. Haas, and Y. Sismanis, “Large-scale matrix factorization with distributed stochastic gradient descent,” in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2011, pp. 69–77.

[14] Y. Koren, “Factorization meets the neighborhood: a multifaceted collaborative filtering model,” in *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2008, pp. 426–434.

[15] Q. He, D. Kifer, J. Pei, P. Mitra, and C. L. Giles, “Citation recommendation without author supervision,” in *Proceedings of the fourth ACM international conference on Web search and data mining*. ACM, 2011, pp. 755–764.

[16] Z.-D. Zhao and M.-S. Shang, “User-based collaborative-filtering recommendation algorithms on hadoop,” in *Third International Conference on Knowledge Discovery and Data Mining, 2010. WKDD’10.* IEEE, 2010, pp. 478–481.

[17] H.-F. Yu, C.-J. Hsieh, S. Si, and I. S. Dhillon, “Scalable coordinate descent approaches to parallel matrix factorization for recommender systems.” in *ICDM*, 2012, pp. 765–774.

[18] Y. Hu, Y. Koren, and C. Volinsky, “Collaborative filtering for implicit feedback datasets,” in *Eighth IEEE International Conference on Data Mining, 2008. ICDM’08.* IEEE, 2008, pp. 263–272.

[19] D. M. Blei, A. Y. Ng, and M. I. Jordan, “Latent dirichlet allocation,” *the Journal of machine Learning research*, vol. 3, pp. 993–1022, 2003.

[20] R. Nallapati, W. Cohen, and J. Lafferty, “Parallelized variational em for latent dirichlet allocation: An experimental evaluation of speed and scalability,” in *Seventh IEEE International Conference on Data Mining Workshops, 2007. ICDM Workshops 2007.* IEEE, 2007, pp. 349–354.

[21] K. Ma and L. Zhang, “Bookmarklet-triggered unified literature sharing system in the cloud,” *International Journal of Grid and Utility Computing*, vol. online, 2014.

[22] K. Ma, B. Yang, and G. Chen, “Doi proxy framework for automated entering and validation of scientific papers,” in *Web-Age Information Management*. Springer, 2013, pp. 799–801.

[23] Y. Zhou, D. Wilkinson, R. Schreiber, and R. Pan, “Large-scale parallel collaborative filtering for the netflix prize,” in *Algorithmic Aspects in Information and Management*. Springer, 2008, pp. 337–348.