International Journal of Emerging Technology in Computer Science & Electronics (IJETCSE)
ISSN: 0976-1353 Volume 21 Issue 1 – APRIL 2016.

RELATIONAL COLLABORATIVE TOPIC REGRESSION FOR RECOMMENDER SYSTEM

M.Vinothkumar[1],P.Indra priya[2]

[1] *PG Student, Department of Computer Science and Engineering, Tagore Engineering College*
[2] *Assistant Professor, Department Of Computer Science and Engineering, Tagore Engineering College*

***Abstract*--Due to its successful application in recommender system, collaborative filtering (CF) has become a hot research topic in data mining and information retrieval. In traditional CF methods, only the feedback matrix, which contains explicit feedback or implicit feedback on the items given by users, is used for training and prediction. Typically, the feedback matrix is spares, which means that most users interact with item. Due to this sparcity problem, traditional CF only feedback matrix is sparse, which means that most users interact with few items. Due to this sparcity problem, rational CF with only feedback information will suffer from unsatisfactory performance. Recently, may researchers have proposed to utilize auxiliary information, such as item content, tp alleviate the data sparcity problem in CF. collaborative topic regression(CTR) is one of the methods which has achieved promising performance by successfully integrating both feedback information and item content information. I many real application, besides the feedback and item content information, there may exist relations among the items which can be helpful for recommendation. In this paper, we develop a novel hierarchical Bayesian model called Relational Collaborative Topic Regression (RCTR), which extends CTR by seamlessly integrating user-item feedback information, item content information, and network structure among items into the same model. Experiments on real-world datasets show that our model can achieve better prediction accuracy than the state-of-the-art methods with lower empirical training time. Moreover, RCTR can learn good interpretable latent stricter which are useful for recommendation.**

I. INTRODUCTION

Recommender System (RS) play an important role to enable us to make effective use of information. For example, Amazon adopts RS for product recommendation, and Netfix uses RS for movie recommendation. Existing RS methods can be roughly categorized into three classes: content based method, collaborative filtering (CF) methods, and Hybrid methods. Content based methods, adopt the profile of the users or products for recommendation. CF based methods, use past activities or preferences, such as the rating on items given by users, for prediction, without using any user or product profiles Hybrid methods, combine both content based method and CF based methods by ensemble techniques. Due to privacy issues, it is harder in general to collect user profiles than past activities. Hence,

CF based methods have become more popular than content based methods in recent years. In most traditional CF methods, only the feedback matrix, which contains either explicit feedback (also called ratings) or implicit feedback on the items given by users, is used for training and prediction. Typically, the feedback matrix is sparse, which means that most items are given feedback by few users or most users only give feedback to few items. Due to this sparsity problem, traditional CF with only feedback information will suffer from unsatisfactory performance. More specifically, it is difficult for CF methods to achieve good performance in both item-oriented setting and user-oriented setting when the feedback matrix is sparse. In an item-oriented setting where we need to recommend users to items, it is generally difficult to know which users could like an item if it has only been given feedback by one or two users. This adds to the difficulty companies face when promoting new products (items). Moreover, users′ ignorance of new items will result in less feedback on the new items, which will further harm the accuracy of their recommendations. For the user-oriented setting where we recommend items to users, it is also difficult to predict what a user likes if the user has only given feedback to one or two items. However, in the real world, it is common to find that most users provide only a little feedback. Actually, providing good recommendations for new users with little feedback is more important than for frequent users since new users will only come back to the site (service) depending on how good the recommendation is. However, for frequent users, it is most likely that they are already satisfied with the site (service). If we man-age to boost the recommendation accuracy for new or infrequent users, more of them will become frequent users, and then better recommendations can be expected with more training data. Therefore, improving the recommendation accuracy at an extremely sparse setting is key to getting the recommender systems working in a positive cycle. information into the model training and prediction procedures. Some methods utilize the item content (attributes) to facilitate the CF training. One representative of these methods is collaborative topic regression (CTR) which jointly models the user-item feedback matrix and the item content information (texts of articles). CTR seamlessly incorporates topic modeling with CF to improve the performance and

interpretability. For new items, CTR is able to perform out-of-matrix prediction (cold-start prediction) using only the content information. Some other methods try to use social networks among users to improve the performance. Among these methods, CTR-SMF extends CTR by integrating the social net-work among users into CTR with social matrix factorization (SMF) techniques, which has achieved better performance than CTR.

In many real applications, besides the feedback and item content information, there may exist relations (or networks) among the items which can also be helpful for recommendation. For example, if we want to recommend papers (references) to users in Cite ULike, the citation relations between papers are informative for recommending papers with similar topics. Other examples of item net-works can be found in hyperlinks among webpages, movies directed by the same directors, and so on.In this paper, we develop a novel hierarchical Bayesian model, called Relational Collaborative Topic Regression (RCTR), to incorporate item relations for recommendation. The main contributions of RCTR are outlined.

II. BACKGROUND

In this section, we give a brief introduction about the back-ground of RCTR, including CF based recommendation, matrix factorization (MF) (also called latent factor model) based CF methods and CTR.

A.CF Based Recommendation

Collaborative topic regression is proposed to recommend documents (papers) to users by seamlessly integrating both feedback matrix and item (document) content information into the same model, which can address the problems faced by MF based CF. By combining MF and latent Dirichlet allocation (LDA), CTR achieves better prediction performance than MF based CF with better interpretable results. Moreover, with the item content information, CTR can predict feedback for out-of-matrix items.

The graphical model of CTR is shown in Fig. 1.
CTR introduces an item latent offset ε_j between the topic proportions u_j in LDA and the item latent vectors v_j in CF. The offset can be explained by the gap between what the

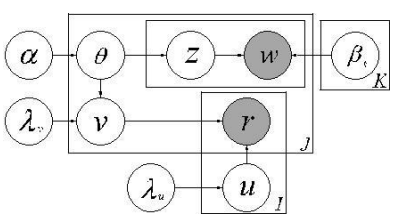

Fig. 1.The graphical model of collaborative topic regression.

III. RELATIONAL COLLABORATIVE TOPIC REGRESSION

In this section, we describe the details of our proposed model, called Relational Collaborative Topic

Regression. Besides the feedback and item content information modeled by CTR, RCTR can also model the relations among the items which are informative for recommendations.

A. Model Formulation

To better illustrate the graphical model of RCTR, we adopt a way different from that in Fig. 1 which is adopted by the authors of CTR. The graphic model of RCTR is shown in Fig. 2, in which the component in the dashed rectangle is what differentiates RCTR from CTR

B. Time Complexity

According to the update rules in the RCTR learning procedure, we can see that for each iteration the time complexity for updating h is OðKLÞ where K is the dimensionality.

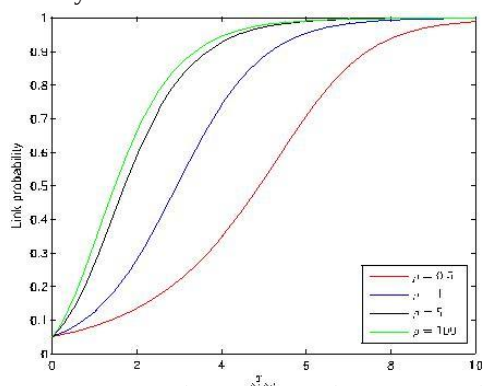

Fig.2 A comparison of link probability functions with different

From our experiments, we find that RCTR needs a smaller number of learning iterations than CTR to achieve satisfactory accuracy. As a consequence, the total empirical measured runtime of training RCTR is lower than that of training CTR even if the time complexity of each iteration of RCTR is slightly higher than that of CTR. This is verified in the experimental results.

C. Discussion on Link Probability Function

Another key property of RCTR is the family of link probability functions, which is inspired by the relational topic model (RTM). The authors in RTM find that different link probability functions can achieve different prediction performance. In RCTR, we use a single parameter r to control the choice of the link probability function. Since r is a non-negative real number, the family of link probability functions actually contains an infinite number of candidate link probability functions. However, only two link

probability functions are proposed in. Hence, our new family of link probability functions can increase the modeling capacity of RCTR, and consequently better performance can be expected. From the perspective of optimization, r can simply be regarded as a necessary regularization hyper-parameter to control the tradeoff between relations and other observations, which can easily be seen. Comparison between link probability functions with different r is shown in Fig. 4, from which we can see

that our link probability functions are flexible enough to model different cases

IV. EXPERIMENTS

We design several experiments and compare the prediction performance between RCTR and the state-of-the art methods on two real-world datasets. The questions we are trying to answer are:

To what degree does RCTR outperform the state-of-the-art methods, especially when the data is extremely sparse?

To what degree does the family of link probability functions help improve the prediction performance?

How is the prediction performance affected by the relational parameter _r and other parameters?

A. Datasets

We use two real-world datasets to conduct our experiments. Both of them are from CiteULike,[2] but they are collected in different ways with different scales and degrees of sparsity. For the feedback matrix in the datasets, if a user reads (or posts) a paper, the corresponding feedback is 1. Otherwise, if a user has not read (or posted) a paper, the corresponding feedback is missing (denoted by 0). The first dataset, citeulike-a. Note that the original dataset does not contain relations between items. We collect the items' relational information from CiteULike and Google Scholar. The second dataset, citeulike-t, we collect independently from the first one. We manually select 273 seed tags and collect all the articles with at least one of these tags. We also crawl the citations between the articles from Google Scholar. Note that the final number of tags associated with all the collected articles is far more than the number (273) of seed tags. Similar to, we remove any users with fewer than three articles. The description of these two datasets is shown in Table 1. We can see that the number of users and items in our collected citeulike-t dataset is larger than that of citeulike-a. Furthermore, the ratios of non-missing entries (equal to 1—sparsity) in the user-item matrices of citeulike-a andciteulike-t are 0:0022 and 0:0007 respectively, which means that the second dataset is sparser than the first.

The text information (item content) of citeulike-a is pre-processed by following the same procedure as that in and we also use their articles' titles and abstracts for the text information of citeulike-t. After removing the stop words,

B. Evaluation Scheme

We design evaluation schemes to evaluate models in both user-oriented and item-oriented settings.

For the user-oriented setting:

Select some percentage Q (e.g. 10 percent) of the users as test users. The training set contains two parts: one part includes all feedbacks of the other (1—Q) of the users, and the other part includes P positive feedbacks (with value 1) for each test user.

Perform prediction on the remaining feedbacks of the test users.

Repeat the above procedure for 1=Q rounds. For each round, we select different test users. For example, if Q ¼ 10%, we perform 1=Q ¼ 10 rounds of tests. This is equivalent to a 10-fold cross validation procedure where each user appears one time in a test set. If P is small, the test set actually contains some new users with little feedback.

We evaluate the predictive performance with two cases: Q ¼ 10% and Q ¼ 100%. The case Q ¼ 10% means that the recommendation system has been running for a long time and only a small number of users are new. The case Q ¼ 100% means that the system is online for only a while and most of the users are new. As stated in Section 1, pro-viding a good recommendation for new users with little feedback is more important than that for frequent users. Hence, it is more interesting to study the performance of recommendation algorithms in extremely sparse settings. We let P vary from 1 to 10 in our experiments and the smaller the P , the sparser the training set. Note that when P ¼ 1 and Q ¼ 100%, only 2:7 percent of the entries with value 1 are put in the training set for dataset citeulike-a and the number for dataset citeulike-t is 5:8 percent.

As in and, we use recall as our evaluation metric since zero feedback may be caused either by users who dis-like an item or by users who do not know the existence of the item, which means precision is not a proper metric here. Like most recommender systems, we sort the predicted feedback of candidate items which are any remaining items that are not put into the training data, and recommend the top M items (articles) to the target user

C. Baselines and Experimental Settings

The models we used for comparison are listed as follows:

MP. The most-popular baseline which orders users or items by how often they appear in the training set.

MC. The most-cited baseline which orders items (papers) by how often they are cited in the user-oriented setting. For the item-oriented setting, the MC baseline will order the

users by the total number of citations of the items (papers) rated by each user.

A variant of content-based methods to incorporate the citation and tag information. We first construct a dictionary containing the original words from the text information and the citations and tags as additional words. The bag-of-words of an article is used as its feature vector. The feature vector of a user is calculated as the average of the feature vectors of the articles s/he gave feedback to. We recommend the items to users with the largest cosine similarities

D. Performance

As stated in Section 4.2, we have two different recommendation settings: user-oriented recommendation

A.User-Oriented Recommendation

User-oriented recommendation tries to recommend items to target users. Fig. 5 shows the recall@300 on dataset citeulike-a when Q ¼ 10% and P is set to be 1, 2, 5, 8, 10.[4]We can see that the baselines MP and MC perform poorly. For all other settings, MP and MC always achieve the worst performance. To avoid clutter and better demonstrate the differences between other stronger models, we choose to drop the corresponding lines for baselines MC and MP in the following experiments.

V. MODULES FOR RECOMMENDER SYSTEM:

A. User Interface Design:

To connect with server user must give their username and password then only they can able to connect the server. If the user already exits directly can login into the server else user must register their details such as

username, password, Email id, City and Country into the server. Database will create the account for the entire user to maintain upload and download rate. Name will be set as user id.Logging in is usually used to enter a specific page. It will search the query and display the query.

B. Website Visiting:

The Internet is supposed to be a global network that links the entire world, but many websites are confined to specific countries. Unsurprisingly, piracy is higher in countries where content isn′t legally available. Some services work through some DNS wizardry.

Web service selection is the action or fact of carefully choosing someone or something as being the best or most suitable.A process in which environmental or genetic influences determine which types of organism thrive better than others, regarded as a factor in evolution.

C. Response Time Calculation:

Response time is the total amount of *time* it takes to respond to a request for service. That service can be anything from a memory fetch, to a disk IO, to a complex database query, or loading a full web page. Ignoring transmission *time* for a moment, the *response time* is the sum of the service *time* and wait *time*.*Response time* may refer to: The *time* lagged between the input and the output signal which depends upon the value of passive components used. *Response time* (technology), the *time* a generic system or functional unit takes to react to a given input. Responsiveness, how quickly an interactive system responds to user input.

D. Time Chart Generation:

A *chart*, also called a graph, is a graphical representation of data, in which "the data is represented by symbols, such as bars in a bar *chart*, lines in a line *chart*, or slices in a pie *chart*". A *chart* can represent tabular numeric data, functions or some kinds of qualitative structure and provides different info.A *chart* is a set of coordinates. When you make a *chart* you start with an empty, two-dimensional space, a vertical dimension (y) and a horizontal dimension (x). You also have a data source. Your job is to translate the data into distances and plot data points in a way that their relative distances are kept. This chart is developed based on the response time of the web services.

E. User Feedback:

This module is used to add user feedback about web services. Feedback is essential to the working and survival of all regulatory mechanisms found throughout living and non-living nature, and in man-made systems such as education system and economy.Information about reactions to a product, a person's performance of a task., etc. This is used as a basis for improvement. The modification or control of a process or system by its results or effects, for example in a biochemical pathway or behavioral response.

F. Service Improvement:

Quality and service improvement tools applied to a healthcare setting can help health care organizations to improve the quality, efficiency and productivity of patient care they provide. Used correctly, these tools and techniques can help healthcare staff to identify and resolve problems as quickly and as cost-effectively as possible while ensuring that any improvements in patient care are sustainable.

REFERENCES

[1] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possi-ble extensions," IEEE Trans. Knowl. Data Eng., vol. 17, no. 6, 734–749, Jun. 2005.

[2] D. Agarwal and B.-C. Chen, "Regression-based latent factor mod-els," in Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Min-ing, 2009, pp. 19–28.

[3] D. Agarwal and B.-C. Chen, "fLDA: Matrix factorization through latent dirichlet allocation," in Proc. 3rd Int. Conf. Web Search Data Mining, 2010, pp. 91–100.

[4] D. Almazro, G. Shahatah, L. Albdulkarim, M. Kherees, R. Marti-nez, and W. Nzoukou, "A survey paper on recommender systems,"CoRR, abs/1006.5278, 2010.

[5] M. Balabanovi_c and Y. Shoham, "Fab: Content-based, collabora-tive recommendation,"Commun. ACM, vol. 40, no. 3, pp. 66–72, 1997.

[6] A. B. Barrag_ans-Mart_ınez, E. Costa-Montenegro, J. C. Burguillo-Rial, M. Rey-Lopez,_ F. A. Mikic-Fonte, and A. Peleteiro-Ramallo, "A hybrid content-based and item-based collaborative filtering approach to recommend tv programs enhanced with singular value decomposition," Inf. Sci., vol. 180, no. 22, pp. 4290–4311, 2010.

[7] I. Bartolini, Z. Zhang, and D. Papadias, "Collaborative filtering with personalized skylines," IEEE Trans. Knowl. Data Eng., vol. 23, no. 2, pp. 190–203, Feb. 2011.

[8] J. Bennett and S. Lanning, "The Netflix prize," in Proc. KDD Cup Workshop, 2007, pp. 3–6.

[9] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," J. Mach. Learn. Res., vol. 3, pp. 993–1022, 2003.

[10] J. Bobadilla, F. Ortega, A. Hernando, and A. Guti_errez, "Recommender systems survey,"Knowl. Based Syst., vol. 46, pp. 109–132, 2013.

[11] J. Breese, D. Heckerman, and C. Kadie, "Empirical analysis of pre-dictive algorithms for collaborative filtering," in Proc. 4th Conf. UncertainityArtif. Intell., 1998, pp. 43–52.

[12] R. D. Burke, "Hybrid recommender systems: Survey and experiments," User Model. User Adapted Interaction, vol. 12, no. 4, pp. 331–370, 2002.
qq.

[13] Y. Cai, H. fung Leung, Q. Li, H. Min, J. Tang, and J. Li, "Typicality-based collaborative filtering recommendation," IEEE Trans. Knowl. Data Eng., vol. 26, no. 3, pp. 766–779, Mar. 2014.

[14] J. Chang and D. M. Blei, "Relational topic models for document networks," in Proc. Int. Conf. Artif. Intell. Stat., 2009, pp. 81–88.