A Latent Pairwise Preference Learning Approach for Recommendation from Implicit Feedback

Yi Fang
Computer Engineering Department
Santa Clara University
Santa Clara, CA 95053, USA
yfang@scu.edu

Luo Si
Department of Computer Science
Purdue University
West Lafayette, IN 47907, USA
lsi@cs.purdue.edu

ABSTRACT

Most of the current recommender systems heavily rely on explicit user feedback such as ratings on items to model users' interests. However, in many applications, it is very hard to collect the explicit feedback, while implicit feedback such as user clicks may be more available. Furthermore, it is often more suitable for many recommender systems to address a ranking problem than a rating predicting problem. This paper proposes a latent pairwise preference learning (LPPL) approach for recommendation with implicit feedback. LPPL directly models user preferences with respect to a set of items rather than the rating scores on individual items, which are modeled with a set of features by analyzing clickthrough data available in many real-world recommender systems. The LPPL approach models both the latent variables of group structure of users and the pairwise preferences simultaneously. We conduct experiments on the testbed from a real-world recommender system and demonstrate that the proposed approach can effectively improve the recommendation performance against several baseline algorithms.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Information filtering*

Keywords

Recommender systems, Implicit feedback, Pairwise preferences

1. INTRODUCTION

Recommender systems utilize different types of user input. The most common one is the high quality explicit feedback such as movie ratings. This is the explicit input by users regarding their interests in items. The vast majority of the literature in the field is focused on explicit feedback, due to the convenience of using this kind of explicit information. However, in many real-world applications, these explicit ratings are hard to collect and user feedback can be implicitly expressed by user behaviors such as clicks, bookmark, purchase history and even mouse movement. In the past decade, the implicit feedback, mainly in the form of clickthrough, has been heavily studied and utilized in web search to improve the relevance of the ranking models. Consequently, an active research field in information retrieval, learning to rank [9], has emerged.

In fact, it is more suitable for many recommendation applications to address a ranking problem than a rating predicting problem [8]. For example, most real-world recommender systems provide the services of the Top-N recommendation, which essentially involves solving a ranking problem. The rating prediction accuracy, which is the objective in many existing methods, is not always consistent with ranking effectiveness. Ratings are often predicted independently for each item while rankings characterize relations among multiple items. Therefore, models for relations and preference comparisons are more desirable than models for individual ratings in recommendation algorithms.

In this paper, we propose a latent pairwise preference learning (LPPL) approach for recommendation from implicit feedback. LPPL directly models user preferences with respect to a set of items rather than the rating scores on individual items. Pairwise preference relations over items are derived from clickthrough data which are abundantly available in many real-world recommender systems. In particular, the pairwise preferences are modeled by a logistic function over a set of features. The latent variables in LPPL can capture the group structure of users. The experiments on the testbed from an online scientific community (i.e., nanoHUB) show that the proposed models can effectively improve the recommendation performance. To the best of our knowledge, this is the first learning to rank work proposed for real-world recommender systems with implicit feedback.

2. RELATED WORK

To address the implicit feedback, matrix factorization techniques are proposed in [4] to incorporate rich user and item information into recommendation, but the work mainly targets on solving a prediction problem, not a ranking problem. In the recent years, Learning to rank (L2R) has been intensively investigated for web search. The goal is to construct a model or a function for ranking entities. Three main classes of L2R approaches are pointwise, pairwise and listwise approaches, respectively [9]. These methods are built

CIKM'12, October 29–November 2, 2012, Maui, HI, USA.

on a solid foundation because it has been shown that they are closely related to optimizing the commonly used ranking criteria. Although valuable work has been done for learning to rank for ad-hoc retrieval, very limited research has been conducted for recommender systems.

A more general formulation, which is called preference learning, has been studied in the machine learning community. Preference learning is about inducing predictive preference models from empirical data, and L2R can be viewed as one of its special cases. A review paper on preference learning for recommender systems can be found in [3]. A more related work to ours is the probabilistic latent preference analysis for collaborative filtering [8] which is based on the Bradley-Terry model for modeling preferences on pairs of items. However, they only use the collaborative information, while our LPPL models utilize both collaborative and content information such as item and user profiles. Furthermore, their evaluations are still based on explicit user feedback, while our experiments are conducted on implicit feedback collected from a real-world recommender system.

3. THE APPROACH

3.1 Setting

The recommendation task in this paper is motivated by the one in nanoHUB[1]. nanoHUB is a popular online scientific community for research, education and collaboration in nanotechnology. It comprises numerous resources with an active user base. These resources include lectures, seminars, tutorials, publications, events and so on. There exists very rich information about resources and users. Most resources contain detailed information such as titles, abstracts and tags. Many registered users also provide detailed profiles about themselves such as their research interest, education and affiliation. The recommendation task is to show a list of other useful resources to the user when he/she is viewing a specific resource, which is something like a "See Also" functionality. This task is similar to Top-N recommendation where a few specific items are suggested to the user so that they are likely to be very appealing to him/her. While our models are presented in the context of the "See Also" recommendation, they can be readily adapted to more general recommendation tasks such as Top-n recommendation. In fact, for both tasks, a ranking problem is a more natural formulation than a rating prediction problem. In addition, the users in the scientific communities such as nanoHUB tend not to give explicit ratings to the resources, even though they may have clear preference in their minds.

3.2 Pairwise Preference from Implicit Feedback

One simple choice of utilizing implicit feedback is to assume that clicked resources are relevant resources and resources not clicked are irrelevant. However, this is not an optimal choice because clicked resources may not be equally relevant and some resources are not clicked due to some other reasons. For example, in a list of 10 recommended resources, a user may only carefully look at the top 2 resources, access the second one and ignore all of the rest 8 resources. It is not reasonable to assume all the 8 resources are irrelevant. Based on the above observation, we adopt a pairwise comparison approach to utilizing implicit feedback. In particular, the pairwise approach compares the probability of relevance of two resources. If a resource in the recommendation list is accessed by the user, the resource tends to be more relevant than the other resources that are not accessed but ranked higher in the list than the accessed resource. A similar assumption was adopted to utilize implicit feedback information for improving accuracy in web search [7].

[1]http://www.nanohub.org

3.3 Latent Pairwise Preferences Learning

In this section, we propose two latent pairwise preferences learning models (i.e., LPPL 1 and LPPL 2) for learning from pairwise preferences derived from implicit feedback.

3.3.1 LPPL 1

For a user u viewing the resource c, if resource i is labeled as being more relevant than resource j, we denote $l_{i,j}^{u,c} = 1$; otherwise $l_{i,j}^{u,c} = -1$. We let Ψ denote the set of $(u; c; i; j)$ quadruples for which the preference $l_{i,j}^{u,c}$ is observed.

Similar to probabilistic latent semantic indexing (pLSA) [5], we introduce a hidden state variable z to capture the group structure of the users. $P(l_{i,j}^{u,c}|u, c, i, j)$ can then be decomposed as:

$$P(l_{i,j}^{u,c}|u, c, i, j) = \sum_{z=1}^{K} P(z|u)P(l_{i,j}^{u,c}|c, i, j, z) \qquad (1)$$

where z is a multinomial variable ($z \in K$) that denotes the user groups while the mixing proportions $P(z|u)$ capture the strength of a user's membership with each group. $P(l_{i,j}^{u,c}|c, i, j, z)$ is the mixture component with each component belonging to the same parametric family of distributions. In pLSA for collaborative filtering [5], Gaussian distribution is often assumed to model rating values which are of numeric scale. In LPPL 1, we designate a logistic function over the preference comparisons to model $P(l_{i,j}^{u,c}|c, i, j, z)$ as:

$$P(l_{i,j}^{u,c}|c, i, j, z) = \sigma\Big(\sum_{v=1}^{V} \lambda_{zv}\big(f_v(c, i) - f_v(c, j)\big)\Big) \qquad (2)$$

where $\sigma(x) = 1/\big(1+\exp(-x)\big)$ is the standard logistic function and λ_{zv} is the weight of the user group z for the v^{th} feature $f_v(c, i)$.

Different types of features can be incorporated in $f_v(c, i)$ for making pairwise comparison between a pair of resource candidates. For example, one type of features can be used to measure the content similarity between a resource candidate i and the current resource c being accessed. The intuition of this type of features suggests that users may prefer resource candidates that is more similar to the current resource that the user is accessing. Another type of features can be used to measure the authority of the authors for different resource candidates. The intuition of this type of features suggests that users may prefer resource candidates from authoritative creators than less authoritative creators.

Similar to pLSA, we can derive an Expectation-Maximization (EM) algorithm to estimate the parameters by iterating E-step and M-step until convergence as follows [5]. In E-step, we compute the posterior probability of z given the quadruple $(u; c; i; j)$ as:

$$P(z|u, c, i, j) = \frac{P(z|u)\sigma\Big(l_{i,j}^{u,c}\sum_{v=1}^{V}\lambda_{zv}\big(f_v(c, i) - f_v(c, j)\big)\Big)}{\sum_{z'=1}^{K}P(z'|u)\sigma\Big(l_{i,j}^{u,c}\sum_{v=1}^{V}\lambda_{z'v}\big(f_v(c, i) - f_v(c, j)\big)\Big)}$$

By optimizing the auxiliary Q-function, we can derive the following M-step update rules:

$$\lambda_{z\cdot}^{*} = \arg\max_{\lambda_{z\cdot}} \sum_{(u,c,i,j)\in\Psi} P(z|u,c,i,j) \log\Big(\sigma\Big(l_{i,j}^{u,c} \sum_{v=1}^{V} \lambda_{zv}\big(f_v(c,i) - f_v(c,j)\big)\Big)\Big)$$

$$P(z|u) = \frac{\sum_{(u',c,i,j)\in\Psi:u'=u} P(z|u,c,i,j)}{\sum_{z'=1}^{K}\sum_{(u',c,i,j)\in\Psi:u'=u} P(z'|u,c,i,j)}$$

As there is no closed form solution to the optimization problem of $\lambda_{z\cdot}$, we can resort to Quasi-Newton methods [10].

3.3.2 LPPL 2

In LPPL 1, $P(z|u)$ is modeled with a separate non-parametric multinomial distribution for each user. In consequence, the model cannot easily generalize $P(z|u)$ to unseen users beyond the training collection, because each parameter in multinomial distribution specifically corresponds to a training instance. pLSA encounters a similar problem and a "fold-in" process is suggested in [5] by re-learning all training documents with the new document to generate an updated parameter estimation. However, the "fold-in" process is time-consuming, and moreover, we do not have any relevance judgment for new users to learn from.

To address this problem, we propose LPPL 2 to model the mixing proportions by a soft-max function, i.e., $P(z|u) = \frac{1}{Z_u}\exp(\sum_{s=1}^{S}\alpha_{zs}g_{us})$ where Z_u is the normalization factor that scales the exponential function to be a proper probability distribution (i.e., $Z_u = \sum_{z=1}^{K}\exp(\sum_{s=1}^{S}\alpha_{zs}g_{us})$). In LPPL 2, user u is represented by a bag of user features $(g_1, ...g_S)$ which can be derived from the user's profile and usage information. α_{zs} is the parameter associated with the features. The mixture component is still a single logistic function as in Eqn. (2). By plugging the soft-max function into Eqn. (1), we can get

$$P(l_{i,j}^{u,c}|u,c,i,j) = \frac{1}{Z_u}\sum_{z=1}^{K}\exp(\sum_{s=1}^{S}\alpha_{zs}g_{us})\sigma\Big(\sum_{v=1}^{V}\lambda_{zv}\big(f_v(c,i)-f_v(c,j)\big)\Big)$$

Because α_{zj} is associated with each user feature instead of each training instance, the above model allows the estimated α_{zs} to be applied to any unseen user. The advantage of LPPL 2 over LPPL 1 is that LPPL 2 is inherently generalizable to new users and it can exploit the rich user information such as those in nanoHUB by conveniently incorporating them as features.

A similar EM algorithm can be derived as follows (the update equation for λ_z is the same as LPPL 1):
E-step:

$$P(z|u,c,i,j) = \frac{\exp(\sum_{s=1}^{S}\alpha_{zs}g_{us})\sigma\big(l_{i,j}^{u,c}\sum_{v=1}^{V}\lambda_{zv}(f_v(c,i)-f_v(c,j))\big)}{\sum_{z'=1}^{K}(\exp(\sum_{s=1}^{S}\alpha_{z's}g_{us})\sigma\big(l_{i,j}^{u,c}\sum_{v=1}^{V}\lambda_{z'v}(f_v(c,i)-f_v(c,j))\big)}$$

M-Step:

$$\alpha_{z\cdot}^{*} = \arg\max_{\alpha_{z\cdot}} \sum_{(u,c,i,j)\in\Psi} P(z|u,c,i,j)\log\Big(\frac{1}{Z_u}\exp(\sum_{s=1}^{S}\alpha_{zs}g_{us})\Big)$$

Based on the pairwise preferences, finding the optimal ranking turns out to be a NP-complete problem, which can be shown via reduction from the cyclic-ordering problem [2]. Similar to the approximation strategy in [8], we can use the following scoring function to efficiently produce a ranking

Table 1: Features used in the LPPL models. "B" denotes the feature takes boolean values and "N" represents numerical values

Feature	Description	Type
f_1	Sim in title between i and c	N
f_2	Sim in full text between i and c	N
f_3	Overlap in author between i and c	B
f_4	Overlap in tag between i and c	N
f_5	Overlap in category between i and c	B
f_7	# of total clicks on i	N
f_6	# of clicks on i in the past 3 months	N
f_8	# of contributions of the author	N
f_9	Category: Courses	B
f_{10}	Category: Simulation tools	B
f_{11}	Category: Publications	B
f_{12}	Category: Downloads	B
f_{13}	Category: Animations	B
g_1	Organization	B
g_2	# of contributions	N
g_3	Top 50 tags	B
g_4	tf-idf of the most 100 common words	N

for LPPL:

$$r_i^{u,c} = \sum_{z=1}^{K} P(z|u)\sigma\Big(\sum_{v=1}^{V}\lambda_{zv}f_v(c,i)\Big)$$

4. EXPERIMENTS

4.1 Experimental Setup

We test our proposed models on the nanoHUB dataset. We use the clickthrough data from March 1, 2011 to June 30, 2011 as training data, and use the month of July, 2011 as test data. The training set contains 3,064 users and 2,335 resources and the test set includes 1,609 users and 1,928 resources. We use the strategy described in Section 3.2 to extract the pairwise preferences from the clickthrough data. One baseline method is content based (CB): rank candidate resources according to the descending order of cosine similarity between the resource being viewed and the candidate resources. The tf-idf weighting scheme is used after the stop words are removed. Another baseline includes wAMAN proposed in [6] for weighting implicit feedback, which is a collaborative filtering based approach. In addition, a hybrid model (CB+wAMAN) is also adopted as a baseline, which is a heuristic linear combination of CB and wAMAN.

As discussed in Section 3.3, LPPL 1 needs to infer the group membership $P(z|u)$ for the unseen user u in the test set. In the experiments, we estimate $P(z|u)$ based on the average of the group memberships of the user's 5 most similar neighbors that appear in the training set. The user similarity is also computed in the same way with the resource similarity (based on the vector space model (VSM) by default). Table 1 contains the features f and g that are used in the experiments for the LPPL models.

We use Mean Percentage Ranking (MPR) [6] to evaluate the prediction accuracy, which is a typical evaluation metric for recommendation with implicit feedback. MPR is recall-oriented because precision based metrics are not very suitable as they require knowing which resources are undesired to a user. Lower values of MPR are more desirable.

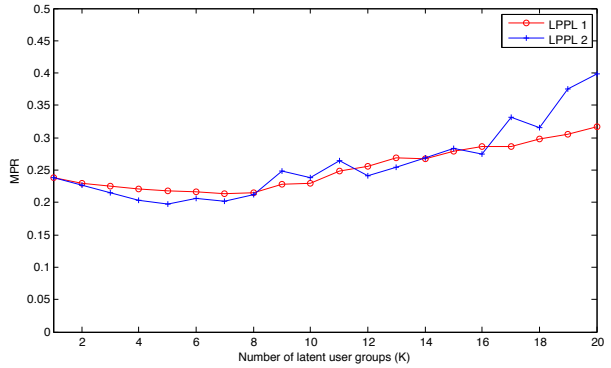

Figure 1: Impact of varying the number of latent factors in LPPL 1 and LPPL 2

The expected value of MPR for random predictions is 50%, and thus MPR $> 50\%$ indicates an algorithm no better than random.

4.2 Effect of Number of Latent Factors

Figure 1 shows the evaluation results of the LPPL models with various number of latent user groups (K), ranging from 1 to 20. When $K = 1$, both models are degenerated to the same model with the single component of logistic function, and thus give the identical results. LPPL 1 and LPPL 2 achieve their best performance at $K = 7$ and $K = 5$, respectively. After that, the performance of the models tends to degrade as K increases, probably due to over-fitting. When $K > 16$, the performance of LPPL 2 deteriorates much faster than LPPL 1. This indicates that LPPL 2 may suffer more from over-fitting when the number of latent factors increases. We will explore a regularization approach to alleviate the over-fitting problem in the future work.

4.3 Comparison with Other Methods

In this subsection, we compare the LPPL models with other methods. From Table 2, we can see that both models perform better than the baselines (CB, wAMAN and CB+wAMAN), with LPPL 2 showing more substantial improvement. It is worth noting that the initial recommendation results in nanoHUB are based on CB and thus there is a presentation bias in favor of the CB method. It is expected that LPPL can show better results when the presentation bias is eliminated. In addition, wAMAN performs much worse than the other methods, probably due to the lack of collaborative information in the training data. An ensemble of CB and wAMAN cannot achieve as good results as CB.

Table 2: Comparison of various methods in MPR

	CB	wAMAN	CB+wAMAN
MPR	0.225	0.366	0.243
		LPPL 1	LPPL 2
MPR		0.214	0.197

4.4 Effect of Document Similarity

The evaluations rely on the computation of document similarity, such as in extracting features for LPPL, in identifying similar users, and in the content-based (CB) recommendation method. In this section, we investigate the effect of document similarity. Specifically, we compare three models of document similarity: Vector space model (VSM), Language model (LM), and Jaccard Index frequency-based model (Jaccard) [1]. Table 3 shows the results. LM seems not a desired choice as all the three methods have the worst performance on LM. On Jaccard, CB obtains a worse result than on VSM, but both LPPL models get better performance than on VSM while the improvement is not substantial.

Table 3: Comparison of various document similarity models in MPR.

	VSM	LM	Jaccard
CB	0.225	0.247	0.239
LPPL 1	0.214	0.223	0.206
LPPL 2	0.197	0.208	0.192

5. CONCLUSION

This paper proposes a latent pairwise preference learning approach for recommendation from implicit feedback. We conduct the experiments on the testbed from an online scientific community and demonstrate that the proposed approach can effectively utilize implicit feedback. In the future work, we will conduct more comprehensive evaluations for the proposed models. We will also extend the models to capture latent groups of resources.

6. ACKNOWLEDGMENTS

We thank the anonymous reviewers for their valuable comments. This research was partially supported by the NSF research grants IIS-1017837 and EEC-0634750. Any opinions, findings, conclusions, or recommendations expressed in this paper are the authors', and do not necessarily reflect those of the sponsors.

7. REFERENCES

[1] J. Akinyemi. *Similarity and Diversity in Information Retrieval*. PhD thesis, University of Waterloo, 2012.
[2] W. Cohen, R. Schapire, and Y. Singer. Learning to order things. *Journal of Artificial Intelligence Research*, 10:243–270, 1999.
[3] M. De Gemmis, L. Iaquinta, P. Lops, C. Musto, F. Narducci, and G. Semeraro. Preference learning in recommender systems. 2009.
[4] Y. Fang and L. Si. Matrix co-factorization for recommendation with rich side information and implicit feedback. In *RecSys Workshop*, pages 65–69, 2011.
[5] T. Hofmann. Collaborative filtering via gaussian probabilistic latent semantic analysis. In *SIGIR*, pages 259–266, 2003.
[6] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *ICDM*, pages 263–272, 2008.
[7] T. Joachims. Optimizing search engines using clickthrough data. In *SIGKDD*, pages 133–142, 2002.
[8] N. Liu, M. Zhao, and Q. Yang. Probabilistic latent preference analysis for collaborative filtering. In *CIKM*, pages 759–766, 2009.
[9] T. Liu. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 3(3):225–331, 2009.
[10] J. Nocedal and S. Wright. *Numerical optimization*. Springer Verlag, 1999.