Collaborative Learning of Preference Rankings

Tim Salimans
Erasmus School of Economics
Rotterdam, The Netherlands
salimans@ese.eur.nl

Ulrich Paquet
Microsoft Research
Cambridge, UK
ulripa@microsoft.com

Thore Graepel
Microsoft Research
Cambridge, UK
thoreg@microsoft.com

ABSTRACT

We propose a model for learning user preference rankings for the purpose of making product recommendations. The model allows us to learn from pairwise preference statements or from (incomplete) rankings over more than two items. We present two algorithms for performing inference in this model, both with excellent scaling in the number of users and items. The superior predictive performance of the new method is demonstrated on the well-known sushi preference data set. In addition, we show how the model can be used effectively in an active learning setting where we select only a small number of informative items for learning.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning

Keywords

Recommendation, Collaborative Learning, Preferences, Ranking, Bayes, Active Learning, Approximate Inference

1. INTRODUCTION

Collaborative recommendation has mostly been studied based on explicit feedback in the form of ratings, or based on implicit binary feedback such as observed purchases or clicks. Often real world data sources lie in between these extremes. Explicit ratings of items are rare and hard to obtain, but often the information is richer than a simple binary signal such as click/non-click. For example, users may express relative value judgments in comparing two different products, or they may provide a partial preference ranking over available items. Such rankings can be explicit such as lists of favorite songs, or inferred from implicit information such as play counts for songs. To make efficient use of such information we propose a new bilinear factor model that maps latent user preferences to observed pairwise comparisons or rankings over items. Since feedback is relative to other items, this modeling approach is more robust than models of user preferences on an absolute scale. Yet it makes more efficient use of available data compared to methods that only allow for binary feedback. Research [2] also shows that people find it easier to formulate their preferences in such a relative way. An additional advantage is that modeling preference rankings directly leads to a ranking of items to be recommended to users, which is the end goal of many recommendation systems.

We present the new user preference model in Section 2, and develop two methods of performing inference in this model in Section 3. In Section 4 we apply the new model to learn correspondents' preferences over sushi items and show that the new method compares favourably to other existing methods. In Section 5 we discuss the potential of the model to guide a more active learning strategy, where we actively and selectively ask the user for relative feedback on different items. Finally, Section 6 concludes.

RecSys'12, September 9–13, 2012, Dublin, Ireland.

2. THE MODEL

Each of the N users and M items are represented with low-rank factors: user i with a $K \times 1$ parameter vector $\boldsymbol{u}_i$, and item j with a $K \times 1$ parameter vector $\boldsymbol{v}_j$. As some items are predominantly more popular than others, a univariate bias parameter b_j is added to each. One might also add similar user-specific offsets to the model.

The user and item features are combined into a latent *score* $s_{i,j}$,

$$s_{i,j} \sim N(\boldsymbol{u}_i' \boldsymbol{v}_j + b_j, 1) \ , \tag{1}$$

which represents how much user i likes item j. This latent score is generated by a bilinear model similar to the one used in Matchbox [7] and many other papers in the collaborative filtering literature.

The relative ordering of a set of scores determines a user's preference of one item over the next. If user i prefers items $j_1 \succ j_2$ ($\succ$ meaning "is preferred to"), we require that $s_{i,j_1} > s_{i,j_2}$. We then observe a number of pairwise comparisons between the different latent scores $s_{i,j}$. For each user, we denote these by $\boldsymbol{C}^i$, a sparsely filled matrix of dimension $M \times M$, with elements

$$\begin{aligned} c^i_{j,j'} &= 1 && \text{if } s_{i,j} > s_{i,j'} \\ c^i_{j,j'} &= -1 && \text{if } s_{i,j} < s_{i,j'} \\ c^i_{j,j'} &= \text{empty} && \text{if unknown .} \end{aligned} \tag{2}$$

These observed preferences can be explicitly provided by the user in the form of a ranking or a number of pairwise preference statements, or they can be inferred from the behavior of the user, for example by ordering the time spent

interacting with different items. Section 4 gives an example application, where we use the model to learn stated preferences over sushi items. Importantly, if the preferences are expressed as a ranking of items, one might always find a set $\{s_{i,j}\}$ that is consistent over the ranked items j, and hence consistent with user i's observations $\boldsymbol{C}^i$. The data likelihood is therefore

$$p(\boldsymbol{C}|\boldsymbol{S}) = \prod_{i=1}^{N} \prod_{(j,j')\in c^i} \mathbb{I}\Big[c^i_{j,j'}(s_{i,j} - s_{i,j'}) > 0\Big] \;, \qquad (3)$$

where $\mathbb{I}[\text{true}] = 1$ and $\mathbb{I}[\text{false}] = 0$. This is similar to the TrueSkill model of [1].

We assign independent normal priors to the user and item vectors, with

$$\begin{aligned} (\boldsymbol{u}_i)_k &\sim N(0,\pi) \;, \qquad \text{for } i = 1,\dots,N,\ k = 1,\dots,K \\ (\boldsymbol{v}_j)_k &\sim N(0,\pi) \;, \qquad \text{for } j = 1,\dots,M,\ k = 1,\dots,K \\ b_j &\sim N(0,\psi) \;, \qquad \text{for } j = 1,\dots,M \;. \end{aligned} \qquad (4)$$

The two hyperparameters π and ψ are set manually, but can also be inferred from the data, as explained in [6]. For brevity we do not consider this here. The full Bayesian network of our model is given in Figure 1.

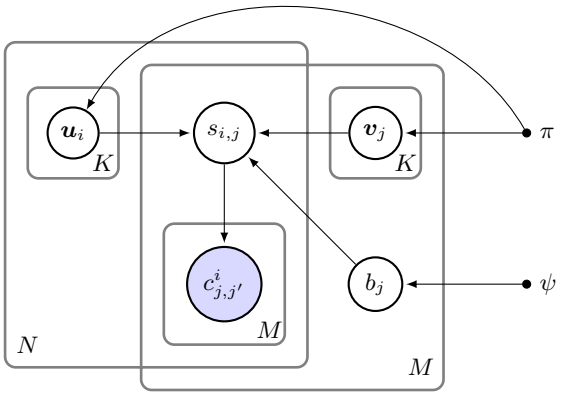

Figure 1: The proposed Bayesian factor model for learning preference rankings

The next section discusses how to infer the posterior distribution of the parameters $\boldsymbol{U}, \boldsymbol{V}, \boldsymbol{b}$ conditional on the observed preferences.

3. BAYESIAN INFERENCE

The model proposed in the last section does not admit a closed form posterior distribution for the parameters $\boldsymbol{U}, \boldsymbol{V}, \boldsymbol{b}$ that we need in order to make recommendations. We therefore propose two strategies for approximating this posterior distribution: a Gibbs sampling algorithm to generate samples from the posterior distribution, and a hybrid message passing algorithm to minimize local divergence measures between the posterior distribution and a factored approximation. The performance and scaling of these two algorithms is evaluated on real world data in Section 4.

3.1 Gibbs Sampling

We can generate correlated samples from the posterior distribution using a Gibbs sampling algorithm that iteratively samples from the conditional distributions $p(\boldsymbol{u}_i|\boldsymbol{V}, \boldsymbol{b}, \boldsymbol{S})$, $p(\boldsymbol{v}_j, b_j|\boldsymbol{U}, \boldsymbol{S})$ and $p(s_{i,j}|\boldsymbol{s}_{i,/j}, \boldsymbol{u}_i, \boldsymbol{v}_j, b_j, \boldsymbol{C}^i)$, for all i, j, where $\boldsymbol{s}_{i,/j}$ denotes the vector of all scores for user i excluding the j-th. The conditional distributions $p(\boldsymbol{u}_i|\boldsymbol{V}, \boldsymbol{b}, \boldsymbol{S})$ and $p(\boldsymbol{v}_j, b_j|\boldsymbol{U}, \boldsymbol{S})$ are Gaussian and have been used by several authors before. See [6] for their precise form. The full conditional distributions $p(s_{i,j}|\boldsymbol{s}_{i,/j}, \boldsymbol{u}_i, \boldsymbol{v}_j, b_j, \boldsymbol{C}^i)$ are univariate truncated normal, which follows from the Gaussian conditional prior (1) and the truncating likelihood (3).

The $s_{i,j}$ are most efficiently updated by first performing a forward pass over all scores for a given user i, sampling the scores $s_{i,j}$ in the order of the observed preference ranking, followed by a backward pass sampling in the reversed order. (Observe that we do not have to sample those $s_{i,j}$ for which we have no feedback.) We find that this updating schedule does a good job of sampling the relative differences between the scores, but that it is slow in changing the overall level of the scores. To further improve the mixing of the Gibbs sampling algorithm we therefore follow the forward and backward pass by an additional Monte Carlo step that simultaneously shifts all scores for a given user, while leaving the stationary distribution of the Markov chain invariant. The update equation for this step is given as

$$\begin{aligned} s_{i,j} &\leftarrow s_{i,j} + d_i, \text{ with } j = 1,\dots,L_i \\ d_i &\sim N(\bar{f}_i - \bar{s}_i, L_i^{-1}) \,, \end{aligned} \qquad (5)$$

with L_i the number of items for which user i has provided feedback, $\bar{f}_i$ the mean predicted score for those items, and $\bar{s}_i$ the mean sampled score.

Since sampling the scores using the steps outlined here is relatively quick compared to sampling $\boldsymbol{U}, \boldsymbol{V}$ and $\boldsymbol{b}$, we find that the most efficient implementation of Gibbs sampling resamples $\boldsymbol{S}$ multiple times per iteration.

3.2 Hybrid VB/EP posterior approximation

The Gibbs sampling algorithm outlined in the last section is relatively fast and can be applied at quite a large scale, however for very large data sets a deterministic approximation of the posterior distribution may provide a better trade-off between accuracy and computational cost. An additional advantage of such a deterministic approximation is that it converges to a single mode of the posterior distributions and that it can be represented more compactly than the Gibbs sampling approximation, which reduces the computational cost of generating new recommendations for users given the posterior approximation. We develop a new algorithm to construct such a deterministic approximation, making use of Expectation Propagation (EP) [5] for the ranking likelihood (3) and Variational Bayes for the latent factor model. EP provides an excellent approximation for the uni-modal posterior resulting from the truncated Gaussian in (3), whereas Variational Bayes picks and locally approximates the posterior mode resulting from the product factor in (1).

We approximate the posterior distribution $p(\boldsymbol{U}, \boldsymbol{V}, \boldsymbol{b}|\boldsymbol{C})$ with a fully factorized Gaussian

$$q(\boldsymbol{U}, \boldsymbol{V}, \boldsymbol{b}) = \prod_{i,k} q(u_{i,k}) \prod_{j,k} q(v_{j,k}) \prod_{j} q(b_j), \qquad (6)$$

although our inference algorithm can also be used with a Gaussian approximation that preserves some of these dependencies, e.g. $q(\boldsymbol{U}, \boldsymbol{V}, \boldsymbol{b}) = \prod_j q(\boldsymbol{v}_j, b_j) \prod_i q(\boldsymbol{u}_i)$. In order to optimize this approximate posterior distribution we first approximate the likelihood term $p(\boldsymbol{C}|\boldsymbol{S})$ by a product

of univariate Gaussian density functions in $s_{i,j}$, i.e.

$$q(\boldsymbol{C}|\boldsymbol{S}) = \prod_{i,j} \phi(s_{i,j}; \mu_{i,j}, \sigma_{i,j}^2), \tag{7}$$

with $\phi(\cdot)$ a Gaussian pdf, which we initialize to have infinite variance. The parameters of the likelihood approximation, $\mu_{i,j}$ and $\sigma_{i,j}^2$ are then set using EP. This EP step starts with the construction of a 'pseudo prior' on the $s_{i,j}$,

$$\begin{aligned} q(s_{i,j}) &\propto \phi(s_{i,j}; \mu_{i,j}^*, \sigma_{i,j}^{2*})/\phi(s_{i,j}; \mu_{i,j}, \sigma_{i,j}^2), \\ \mu_{i,j}^* &= \mathbb{E}_q s_{i,j} \text{ and } \sigma_{i,j}^{2*} = \mathrm{Var}_q\, s_{i,j}. \end{aligned} \tag{8}$$

Using this pseudo prior, the algorithm for determining the approximate likelihood terms $\phi(s_{i,j}; \mu_{i,j}, \sigma_{i,j}^2)$ is identical to that used by the TrueSkill rating system [1], with the scores $s_{i,j}$ taking the place of the 'player skills' in that system. We refer the reader to [1] for the specifics on the EP step.

After the EP step, we optimize the posterior approximation using Variational Bayes, i.e., we choose our posterior approximation to solve

$$\max_{q(\boldsymbol{U},\boldsymbol{V},\boldsymbol{b})} \mathbb{E}_q \log q(\boldsymbol{C}|\boldsymbol{S}) p(\boldsymbol{U},\boldsymbol{V},\boldsymbol{b}) - \log q(\boldsymbol{U},\boldsymbol{V},\boldsymbol{b}). \tag{9}$$

This step can be implemented efficiently using the Variational Bayes Expectation Maximization (VBEM) algorithm. The resulting update equations can be found in [6]. However, note that for our application the expectations with respect to $s_{i,j}$ in (9) follow from $q(\boldsymbol{U},\boldsymbol{V},\boldsymbol{b})$ rather than from a separate posterior approximation on $s_{i,j}$, as is more commonly used (e.g. [6]). By avoiding this explicit approximation of $p(\boldsymbol{S}|\boldsymbol{C})$, the posterior approximation $q(\boldsymbol{U},\boldsymbol{V},\boldsymbol{b})$ gains in accuracy without increasing computational cost. The VBEM and EP steps are repeated until convergence.

3.3 Parallel Computation

For many real world applications of recommendation algorithms, both the number of users as well as the number of items is very large, necessitating the use of parallel computation to speed up inference. Both algorithms described above can be completely parallelized over users when updating $\boldsymbol{S}$ and $\boldsymbol{U}$, and over items when updating $\boldsymbol{V}$ and $\boldsymbol{b}$, which is an important advantage over the message passing algorithm used in [7]. Since in our application the number of items is quite small compared to the number of users, we found it most efficient to distribute the users and their feedback over multiple threads. Within each thread, $\boldsymbol{S}$ and $\boldsymbol{U}$ can then be updated without requiring any communication across threads. Every update of $\boldsymbol{V}$ and $\boldsymbol{b}$ then requires each thread to submit the sufficient statistics for the update of these variables and to receive the updated values. Since the number of items is relatively low this adds very little overhead and it allows us to speed up inference almost linearly with the number of available computation nodes.

4. LEARNING SUSHI PREFERENCES

In order to compare the new algorithms to existing methods we evaluate them on the sushi preference data of [3]. This data set was generated by asking 5,000 survey correspondents to order a subset of 100 sushi types according to their preferences. Each correspondent provided two such ordered lists containing 10 different sushi types. [3] evaluate their collaborative ranking approach by training on list 'B' and using the model to predict the order of list 'A'. They measure the performance of their method by the average Spearman correlation between the predicted and realized ranking. We use this measure to compare the performance of the new method to the 'Nantonac' algorithm of [3], and also to compare our two inference algorithms against each other. Using this measure, we found that the maximum predictive accuracy was reached after about 1000 draws of the Gibbs sampler after a burn-in period of 100 draws, or after 50 iterations of the VB/EP algorithm. The corresponding results are shown in Table 1 below.

Table 1: Prediction accuracy of different methods on Sushi preference data

Method	Spearman cor. test
New factor model, Gibbs	0.56
New factor model, VB/EP	0.54
Nantonac [3]	0.49

The results in Table 1 show that the new method compares favorably to that of [3]: The Gibbs sampling version of the new algorithm improves the Spearman correlation of the predictions with the test set by 0.07 in comparison with the Nantonac method, while the deterministic posterior approximation gives an improvement of 0.05. The relatively small performance difference between the Gibbs sampling inference algorithm and the deterministic posterior approximation suggests that the latter is the more practical choice for real world applications, taking into account its benefits discussed in Section 3.2.

5. ACTIVE LEARNING

In order to improve our recommendations we may actively ask users to provide explicit feedback on certain items. This is most commonly done on an absolute rating scale, i.e. by asking the users to rate items. However, some studies indicate that people are better able to formulate their preferences in a relative way, by ranking multiple items, see e.g. [2]. Such relative preference statements can be used directly by the model presented in Section 2.

Asking the user for feedback is costly as it will take time for the user to think about his or her preferences. In addition, users may find it difficult to provide a full ranking of a very large list of items, so the the number of items we can enquire about is limited. When selecting this limited number of items we should take into account that not every item will be equally informative. A popular measure of the amount of information contained in a data point is the entropy reduction in our posterior distribution that we can expect upon conditioning on that data point [4]. By maximizing the expected entropy reduction in our posterior we can select the most informative items to present to the user for feedback.

Since the posterior distribution of the model given in Section 2 is not available in closed form, we cannot maximize the expected entropy distribution exactly. However, we can get an estimate of the amount of information in each possible observation by making use of posterior approximations. In doing so we will focus on the entropy reduction in the posterior distribution of the user parameters $q(\boldsymbol{u}_i)$, which – due to their greater number – are generally much more uncertain than the parameters of the items. To derive an expression for the approximate entropy reduction after obtaining a new observation, we assume a factorized posterior

approximation over $\boldsymbol{U}, \boldsymbol{V}, \boldsymbol{d}$ and $\boldsymbol{S}$, optimized using Variational Bayes. Note that this is not exactly the same as the approximation presented in Section 3.2, where we integrated out $\boldsymbol{S}$ in updating $\boldsymbol{U}, \boldsymbol{V}$ and $\boldsymbol{d}$. The Variational Bayes EM algorithm then uses the following update equation for the approximate posterior distribution on the user parameters:

$$q(\boldsymbol{u}_i) = N(\mu, \Sigma), \text{ with} \tag{10}$$

$$\Sigma = \left[\frac{1}{\pi}\mathrm{I}_K + \sum_j \mathbb{E}_q\left[\boldsymbol{v}_j \boldsymbol{v}_j'\right]\right]^{-1} \quad \mu = \Sigma\left[\sum_j \mathbb{E}_q\left[\boldsymbol{v}_j\left(s_{i,j} - b_j\right)\right]\right]$$

The entropy of this approximate posterior distribution is given by

$$H(q(\boldsymbol{u}_i)) \propto 0.5 \log|\Sigma|\,. \tag{11}$$

After adding a new item l to the ranking of the user we can update the approximate posterior distribution $q(\boldsymbol{u}_i)$ to $q'(\boldsymbol{u}_i) = N(\mu', \Sigma')$, while keeping $q(\boldsymbol{V}, \boldsymbol{d})$ fixed. The new entropy of $q'(\boldsymbol{u}_i)$ is then given by

$$\begin{aligned} H(q'(\boldsymbol{u}_i)) &\propto 0.5 \log|\Sigma'| = -0.5 \log|\Sigma^{-1} + \mathbb{E}_q \boldsymbol{v}_l \boldsymbol{v}_l'| \\ &\propto H(q(\boldsymbol{u}_i)) - 0.5 \log(1 + \mathbb{E}_q \boldsymbol{v}_l' \Sigma \boldsymbol{v}_l) \end{aligned} \tag{12}$$

In order to maximize the information gain, or entropy reduction, we should thus ask the user to rank that item for which the parameter vector $\boldsymbol{v}_l$ has the highest expected Mahalanobis norm $\|\boldsymbol{v}_l\|_\Sigma$ with respect to the covariance matrix of the current posterior approximation. This has the effect of selecting items that are most informative for exactly those elements of the user vector $\boldsymbol{u}_i$ of which we are most uncertain. Note that for the approximate entropy (12) it does not matter what other item we compare the new item l to, or even whether we have a complete ranking with the new item or just a partial ranking. While this is obviously a very crude approximation, it still gives us a useful rule for actively selecting training examples as shown below.

We evaluate this active selection strategy using the sushi preference data, and we compare the resulting prediction accuracy with that obtained under random selection of the training examples. For each user the data set contains a training set ranking of 10 items of sushi. We actively select a subset of these items for each user by starting out with an empty selection set and subsequently adding that sushi item that minimizes the expected entropy in Equation (12). We then use the resulting selection of training examples to predict the ranking of the test set. For comparison, we do the same while selecting randomly from the remaining sushi items at each iteration. We display the accuracy of the resulting predictions for different numbers of selected items from a minimum of 3 to the maximum of 10. As can be seen from Figure 2 the active selection method leads to faster learning of the correct preferences than random selection of training examples.

Note that the performance measures of the two selection methods in Figure 2 converge as the number of training examples increases because both methods select from the same limited set of 10 potential examples. For small numbers of examples the performance of the active selection method improves much faster than under random selection, indicating the practical value of such an active learning strategy for real life applications, where the user typically only provides feedback on a relatively small fraction of items.

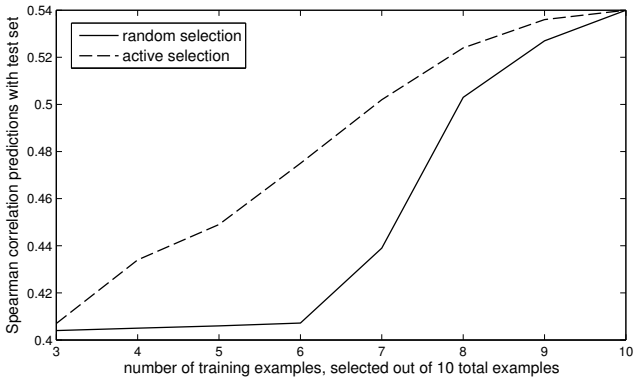

Figure 2: Prediction accuracy obtained using active versus random selection of training examples

6. CONCLUSION

We have proposed a Bayesian factor model to learn users' preference rankings for the purpose of product recommendation. Learning preference rankings with this model can be done quickly and efficiently at large scale, using the two inference algorithms we have developed. The accuracy of our model was demonstrated on a real world data set and was shown to improve upon existing methods. In addition, we have shown that the model can also be used effectively for active preference elicitation. By actively selecting product comparisons to present to the user, we can uncover the user's preferences without requiring large amounts of feedback. This makes the process of preference elicitation much less burdensome on the user, and it can dramatically improve prediction accuracy for real life applications.

7. REFERENCES

[1] P. Dangauthier, R. Herbrich, T. Minka, and T. Graepel. Trueskill through time: Revisiting the history of chess. *Advances in Neural Information Processing Systems*, 20:931–938, 2008.

[2] S. R. Jaeger, A. S. Jørgensen, M. D. Aaslyng, and W. L. Bredie. Best-worst scaling: An introduction and initial comparison with monadic rating for preference elicitation with food products. *Food Quality and Preference*, 19(6):579 – 588, 2008.

[3] T. Kamishima and S. Akaho. Nantonac collaborative filtering. In *Proceedings of The International Workshop on Data-Mining and Statistical Science*, pages 117–124, 2006.

[4] D. J. MacKay. Information-based objective functions for active data selection. *Neural Computation*, 4:590–604, 1992.

[5] T. P. Minka. Expectation propagation for approximate Bayesian inference. In *UAI '01: Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence*, pages 362–369, 2001.

[6] U. Paquet, B. Thomson, and O. Winther. A hierarchical model for ordinal matrix factorization. *Statistics and Computing*, 21(3):1–13, 2011.

[7] D. H. Stern, R. Herbrich, and T. Graepel. Matchbox: large scale online Bayesian recommendations. In *Proceedings of the 18th international conference on World wide web*, pages 111–120, 2009.