Efficient Bayesian Hierarchical User Modeling for Recommendation Systems

Yi Zhang , Jonathan Koren
School of Engineering
University of California Santa Cruz
Santa Cruz, CA, USA
{yiz, jonathan}@soe.ucsc.edu

ABSTRACT

A content-based personalized recommendation system learns user specific profiles from user feedback so that it can deliver information tailored to each individual user's interest. A system serving millions of users can learn a better user profile for a new user, or a user with little feedback, by borrowing information from other users through the use of a Bayesian hierarchical model. Learning the model parameters to optimize the joint data likelihood from millions of users is very computationally expensive. The commonly used EM algorithm converges very slowly due to the sparseness of the data in IR applications. This paper proposes a new fast learning technique to learn a large number of individual user profiles. The efficacy and efficiency of the proposed algorithm are justified by theory and demonstrated on actual user data from Netflix and MovieLens.

1. INTRODUCTION

Personalization is the future of the Web, and it has achieved great success in industrial applications. For example, online stores, such as *Amazon* and *Netflix*, provide customized recommendations for additional products or services based on a user's history. Recent offerings such as *My MSN*, *My Yahoo!*, *My Google*, and *Google News* have attracted much attention due to their potential ability to infer a user's interests from his/her history.

One major personalization topic studied in the information retrieval community is content-based personal recommendation systems[1]. These systems learn user-specific profiles from user feedback so that they can recommend information tailored to each individual user's interest without requiring the user to make an explicit query. Learning the user profiles is the core problem for these systems.

A user profile is usually a classifier that can identify whether a document is relevant to the user or not, or a regression model that tells how relevant a document is to the user. One major challenge of building a recommendation or personalization system is that the profile learned for a particular user is usually of low quality when the amount of data from that particular user is small. This is known as the "cold start" problem. This means that any new user must endure poor initial performance until sufficient feedback from that user is provided to learn a reliable user profile.

There has been much research on improving classification accuracy when the amount of labeled training data is small. The semi-supervised learning approach combines unlabeled and labeled data together to achieve this goal [26]. Another approach is using domain knowledge. Researchers have modified different learning algorithms, such as Naïve-Bayes [17], logistic regression [7], and SVMs [22], to integrate domain knowledge into a text classifier. The third approach is borrowing training data from other resources [5][7]. The effectiveness of these different approaches is mixed, due to how well the underlying model assumption fits the data.

One well-received approach to improve recommendation system performance for a particular user is borrowing information from other users through a Bayesian hierarchical modeling approach. Several researchers have demonstrated that this approach effectively trades off between shared and user-specific information, thus alleviating poor initial performance for each user[27][25].

In order to learn a Bayesian hierarchical model, the system usually tries to find the most likely model parameters for the given data. A mature recommendation system usually works for millions of users. It is well known that learning the optimal parameters of a Bayesian hierarchical model is computationally expensive when there are thousands or millions of users. The EM algorithm is a commonly used technique for parameter learning due to its simplicity and convergence guarantee. However, a content based recommendation system often handles documents in a very high dimensional space, in which each document is represented by a very sparse vector. With careful analysis of the EM algorithm in this scenario (Section 4), we find that the EM algorithm converges very slowly due to the sparseness of the input variables. We also find that updating the model parameter at each EM iteration is also expensive with computational complexity of $O(MK)$, where M is the number of users and K is the number of dimensions.

This paper modifies the standard EM algorithm to create an improved learning algorithm, which we call the "Modified EM algorithm." The basic idea is that instead of calculat-

[1]Content-based recommendation is also called adaptive filtering, or item-based collaborative filtering. In this paper, the words "filtering" and "recommendation" are used interchangeably.

SIGIR'07, July 23–27, 2007, Amsterdam, The Netherlands.

ing the numerical solution for all the user profile parameters, we derive the analytical solution of the parameters for some feature dimensions, and at the M step use the analytical solution instead of the numerical solution estimated at E step for those parameters. This greatly reduces the computation at a single EM iteration, and also has the benefit of increasing the convergence speed of the learning algorithm. The proposed technique is not only well supported by theory, but also by experimental results.

The organization of the remaining parts of this paper is as follows: Section 3 describes the Bayesian hierarchical linear regression modeling framework used for content-based recommendations. Section 4 describes how to learn the model parameters using the standard EM algorithm, along with using the new technique proposed in this paper. The experimental setting and results used to validate the proposed learning technique are reported in Sections 5 and 6. Section 7 summarizes and offers concluding remarks.

2. RELATED WORK

Providing personalized recommendations to users has been identified as a very important problem in the IR community since the 1970's. The approaches that have been used to solve this problem can be roughly classified into two major categories: content based filtering versus collaborative filtering. **Content-based filtering** studies the scenario where a recommendation system monitors a document stream and pushes documents that match a user profile to the corresponding user. The user may read the delivered documents and provide explicit relevance feedback, which the filtering system then uses to update the user's profile using relevance feedback retrieval models (e.g. Boolean models, vector space models, traditional probabilistic models [20] , inference networks [3] and language models [6]) or machine learning algorithms (e.g. Support Vector Machines (SVM), K nearest neighbors (K-NN) clustering, neural networks, logistic regression, or Winnow [16] [4] [23]). **Collaborative filtering** goes beyond merely using document content to recommend items to a user by leveraging information from other users with similar tastes and preferences in the past. Memory-based heuristics and model based approaches have been used in collaborative filtering task [15] [8] [2] [14] [12] [11].

This paper contributes to the content-based recommendation research by improving the *efficiency and effectiveness* of Bayesian hierarchical linear models, which have a strong theoretical basis and good empirical performance on recommendation tasks[27][25]. This paper does not intend to compare content-based filtering with collaborative filtering or claim which one is a better solution to the problem. We think each complements the other, and that content-based filtering is extremely useful for handling new documents/items with little or no user feedback. Similar to some other researchers[18][1][21], we found that a recommendation system will be more effective when both techniques are combined. However, this is beyond the scope of this paper and thus not discussed here.

3. BAYESIAN HIERARCHICAL LINEAR REGRESSION

Assume there are M users in the system. The task of the system is to recommend documents that are relevant to each user. For each user, the system learns a user model from the user's history. In the rest of this paper, we will use the following notations to represent the variables in the system.

$m = 1, 2, ..., M$**:** The index for each individual user. M is the total number of users.

w_m**:** The user model parameter associated with user m. w_m is a K dimensional vector.

$j = 1, 2, ..., J_m$**:** The index for a set of data for user m. J_m is the number of training data for user m.

$D_m = \{(x_{m,j}, y_{m,j})\}$**:** A set of data associated with user m. $x_{m,j}$ is a K dimensional vector that represents the mth user's jth training document.[2] $y_{m,j}$ is a scalar that represents the label of document $x_{m,j}$.

$k = 1, 2, ..., K$**:** The dimensional index of input variable x.

The Bayesian hierarchical modeling approach has been widely used in real-world information retrieval applications. Generalized Bayesian hierarchical linear models, one of the simplest Bayesian hierarchical models, are commonly used and have achieved good performance on collaborative filtering [25] and content-based adaptive filtering [27] tasks. Figure 1 shows the graphical representation of a Bayesian hierarchical model. In this graph, each user model is represented by a random vector w_m. We assume a user model is sampled randomly from a prior distribution $P(w|\Phi)$. The system can predict the user label y of a document x given an estimation of w_m (or w_m's distribution) using a function $y = f(x, w)$. The model is called generalized Bayesian hierarchical linear model when $y = f(w^T x)$ is any generalized linear model such as logistic regression, SVM, and linear regression. To reliably estimate the user model w_m, the system can borrow information from other users through the prior $\Phi = (\mu, \Sigma)$.

Now we look at one commonly used model where $y = w^T x + \epsilon$, where $\epsilon \sim N(0, \sigma_\epsilon^2)$ is a random noise [25][27]. Assume that each user model w_m is an independent draw from a population distribution $P(w|\Phi)$, which is governed by some unknown hyperparameter Φ. Let the prior distribution of user model w be a Gaussian distribution with parameter $\Phi = (\mu, \Sigma)$, which is the commonly used prior for linear models. $\mu = (\mu_1, \mu_2, ..., \mu_K)$ is a K dimensional vector that represents the mean of the Gaussian distribution, and Σ is the covariance matrix of the Gaussian. Usually, a Normal distribution $N(0, aI)$ and an Inverse Wishart distribution $P(\Sigma) \propto |\Sigma|^{-\frac{1}{2}b} \exp(-\frac{1}{2} c \mathrm{tr}(\Sigma^{-1}))$ are used as hyperprior to model the prior distribution of μ and Σ respectively. I is the K dimensional identity matrix, and a, b, and c are real numbers.

With these settings, we have the following model for the system:

1. μ and Σ are sampled from $N(0, aI)$ and $IW_\nu(aI)$, respectively.
2. For each user m, w_m is sampled randomly from a Normal distribution: $w_m \sim N(\mu, \Sigma^2)$
3. For each item $x_{m,j}$, $y_{m,j}$ is sampled randomly from a Normal distribution: $y_{m,j} \sim N(w_m^T x_{m,j}, \sigma_\epsilon^2)$.

[2]The first dimension of x is a dummy variable that always equals to 1.

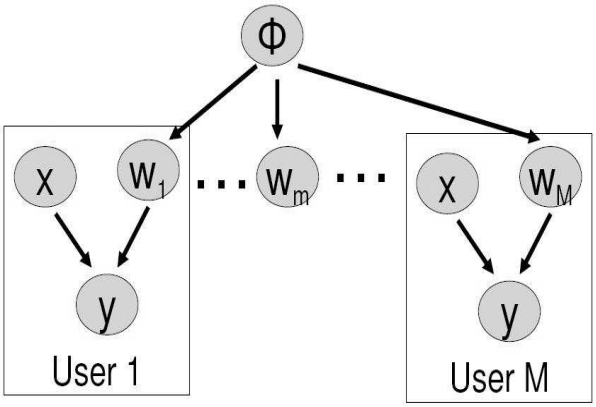


Figure 1: Illustration of dependencies of variables in the hierarchical model. The rating, y, for a document, x, is conditioned on the document and the user model, w_m, associated with the user m. Users share information about their models through the prior, $\Phi = (\mu, \Sigma)$.

Let $\theta = (\Phi, w_1, w_2, ..., w_M)$ represent the parameters of this system that needs to be estimated. The joint likelihood for all the variables in the probabilistic model, which includes the data and the parameters, is:

$$P(D, \theta) = P(\Phi) \prod_m P(w_m|\Phi) \prod_j P(y_{m,j}|x_{m,j}, w_m) \quad (1)$$

For simplicity, we assume a, b, c, and σ_ϵ are provided to the system.

4. MODEL PARAMETER LEARNING

If the prior Φ is known, finding the optimal w_m is straightforward: it is a simple linear regression. Therefore, we will focus on estimating Φ. The maximum a priori solution of Φ is given by

$$\begin{aligned} \Phi_{MAP} &= \arg\max_\Phi P(\Phi|D) && (2)\\ &= \arg\max_\Phi \frac{P(\Phi, D)}{P(D)} && (3)\\ &= \arg\max_\Phi P(D|\Phi)P(\Phi) && (4)\\ &= \arg\max_\Phi \int_w P(D|w, \Phi)P(w|\Phi)P(\Phi)dw && (5) \end{aligned}$$

Finding the optimal solution for the above problem is challenging, since we need to integrate over all $w = (w_1, w_2, ..., w_M)$, which are unobserved hidden variables.

4.1 EM Algorithm for Bayesian Hierarchical Linear Models

In Equation 5, Φ is the parameter needs to be estimated, and the result depends on unobserved latent variables w. This kind of optimization problem is usually solved by the EM algorithm.

Applying EM to the above problem, the set of user models w are the unobservable hidden variables and we have:

$$Q = \int_w P(w|\mu, \Sigma^2, D_m) \log P(\mu, \Sigma^2, w, D)dw$$

Based on the derivation of the EM formulas presented in [24], we have the following Expectation-Maximization steps for finding the optimal hyperparameters. For space considerations, we omit the derivation in this paper since it is not the focus of our work.

E step: For each user m, estimate the user model distribution $P(w_m|D_m, \Phi) = N(w_m; \bar{w}_m, \Sigma_m^2)$ based on the current estimation of the prior $\Phi = (\mu, \Sigma^2)$.

$$\bar{w}_m = ((\Sigma^2)^{-1} + \frac{S_{xx,m}}{\sigma_\epsilon^2})^{-1}(\frac{S_{xy,m}}{\sigma_\epsilon^2} + (\Sigma^2)^{-1}\mu) \quad (6)$$

$$\Sigma_m^2 = ((\Sigma^2)^{-1} + \frac{S_{xx,m}}{\sigma_\epsilon^2})^{-1} \quad (7)$$

$$where\ S_{xx,m} = \sum_j x_{m,j}x_{m,j}^T\ S_{xy,m} = \sum_j x_{m,j}y_{m,j}$$

M step: Optimize the prior $\Phi = (\mu, \Sigma^2)$ based on the estimation from the last E step.

$$\mu = \frac{1}{M}\sum_m \bar{w}_m \quad (8)$$

$$\Sigma^2 = \frac{1}{M}\sum_m \Sigma_m^2 + (\bar{w}_m - \mu)(\bar{w}_m - \mu)^T \quad (9)$$

Many machine learning driven IR systems use a point estimate of the parameters at different stages in the system. However, we are estimating the posterior distribution of the variables at the E step. This avoids overfitting w_m to a particular user's data, which may be small and noisy. A detailed discussion about this subject appears in [10].

4.2 New Algorithm: Modified EM

Although the EM algorithm is widely studied and used in machine learning applications, using the above EM process to solve Bayesian hierarchical linear models in large-scale information retrieval systems is still too computationally expensive. In this section, we describe why the learning rate of the EM algorithm is slow in our application and introduce a new technique to make the learning of the Bayesian hierarchical linear model scalable. The derivation of the new learning algorithm will be based on the EM algorithm described in the previous section.

First, the covariance matrices Σ^2, Σ_m^2 are usually too large to be computationally feasible. For simplicity, and as a common practice in IR, we do not model the correlation between features. Thus we approximate these matrices with K dimensional diagonal matrices. In the rest of the paper, we use these symbols to represent their diagonal approximations:

$\Sigma^2 =$

$$\begin{pmatrix} \sigma_1^2 & 0 & .. & 0 \\ 0 & \sigma_2^2 & .. & 0 \\ .. & .. & .. & .. \\ 0 & 0 & .. & \sigma_K^2 \end{pmatrix}$$

$\Sigma_m^2 =$

$$\begin{pmatrix} \sigma_{m,1}^2 & 0 & .. & 0 \\ 0 & \sigma_{m,2}^2 & .. & 0 \\ .. & .. & .. & .. \\ 0 & 0 & .. & \sigma_{m,K}^2 \end{pmatrix}$$

Secondly, and most importantly, the input space is very sparse and there are many dimensions that are not "related" to a particular user in a real IR application. For example, let us consider a movie recommendation system, with the input variable x representing a particular movie. For the jth movie that the user m has seen, let $x_{m,j,k} = 1$ if the director of the movie is "Jean-Pierre Jeunet" (indexed by k). Here we assume that whether or not that this director directed a specific movie is represented by the kth dimension. If the user m has never seen a movie directed by "Jean-Pierre Jeunet", then the corresponding dimension is always zero ($x_{m,j,k} = 0$ for all j) .

One major drawback of the EM algorithm is that the importance of a feature, μ_k, may be greatly dominated by users who have never encountered this feature (i.e. $\sum_j x_{m,j,k} = 0$) at the M step (Equation 8). Assume that 100 out of 1 million users have viewed the movie directed by "Jean-Pierre Jeunet", and that the viewers have rated all of his movies as "excellent". Intuitively, he is a good director and the weight for him (μ_k) should be high. Before the EM iteration, the initial value of μ is usually set to 0. Since the other 999,900 users have not seen this movie, their corresponding weights $(w_{1,k}, w_{2,k}, ..., w_{m,k}..., w_{999900,k})$ for that director would be very small initially. Thus the corresponding weight of the director in the prior μ_k at the first M step would be very low , and the variance $\sigma_{m,k}$ will be large (Equations 8 and 7). It is undesirable that users who have never seen any movie produced by the director influence the importance of the director so much. This makes the convergence of the standard EM algorithm very slow.

Now let's look at whether we can improve the learning speed of the algorithm. Without a loss of generality, let us assume that the kth dimension of the input variable x is not related to a particular user m. By which we mean, $x_{m,j,k} = 0$ for all $j = 1, ..., J_m$. It is straightforward to prove that the kth row and kth column of $S_{xx,m}$ are completely filled with zeros, and that the kth dimension of $S_{xy,m}$ is zeroed as well. Thus the corresponding kth dimension of the user model's mean, $\bar{w}_m$, should be equal to that of the prior: $\bar{w}_{m,k} = \mu_k$, with the corresponding covariance of $\sigma_{m,k} = \sigma_k$.

At the M step, the standard EM algorithm uses the numerical solution of the distribution $P(w_m|D_m, \Phi)$ estimated at E step (Equation 8 and Equation 7). However, the numerical solutions are very unreliable for $\bar{w}_{m,k}$ and $\sigma_{m,k}$ when the kth dimension is not related to the mth user. A better approach is using the analytical solutions $\bar{w}_{m,k} = \mu_k$, and $\sigma_{m,k} = \sigma_k$ for the unrelated (m, k) pairs, along with the numerical solution estimated at E step for the other (m, k) pairs. Thus we get the following new EM-like algorithm:

Modified E step: For each user m, estimate the user model distribution $P(w_m|D_m, \Phi) = N(w_m; \bar{w}_m, \Sigma_m^2)$ based on the current estimation of $\sigma_\epsilon, \mu, \Sigma^2$.

$$\bar{w}_m = ((\Sigma^2)^{-1} + \tfrac{S_{xx,m}}{\sigma_\epsilon^2})^{-1}(\tfrac{S_{xy,m}}{\sigma_\epsilon^2} + (\Sigma^2)^{-1}\mu) \quad (10)$$

$$\sigma_{m,k}^2 = ((\sigma_k^2)^{-1} + \tfrac{s_{xx,m,k}}{\sigma_\epsilon^2})^{-1} \quad (11)$$

$$where\ s_{xx,m,k} = \sum_j x_{m,j,k}^2\ and\ s_{xy,m,k} = \sum_j x_{m,j,k} y_{m,j}$$

Modified M Step Optimize the prior $\Phi = (\mu, \Sigma^2)$ based on the estimation from the last E step for related user-feature pairs. The M step implicitly uses the analytical solution for unrelated user-feature pairs.

$$\mu_k = \frac{1}{M_k} \sum_{m:related} \bar{w}_{m,k} \quad (12)$$

$$\sigma_k^2 = \frac{1}{M_k} \sum_{m:related} \sigma_{m,k}^2 + (\bar{w}_{m,k} - \mu_k)(\bar{w}_{m,k} - \mu_k)^T \quad (13)$$

where M_k is the number of users that are related to feature k

We only estimate the diagonal of Σ_m^2 and Σ since we are using the diagonal approximation of the covariance matrices. To estimate $\bar{w}_m$, we only need to calculate the numerical solutions for dimensions that are related to user m. To estimate σ_k^2 and μ_k, we only sum over users that are related to the kth feature.

There are two major benefits of the new algorithm. First, because only the related (m, k) pairs are needed at the modified M step, the computational complexity in a single EM iteration is much smaller when the data is sparse, and many of (m, k) pairs are unrelated. Second, the parameters estimated at the modified M step (Equations 12 – 13) are more accurate than the standard M step described in Section 4.1 because the exact analytical solutions $\bar{w}_{m,k} = \mu_k$ and $\sigma_{m,k} = \sigma_k$ for the unrelated (m, k) pairs were used in the new algorithm instead of an approximate solution as in the standard algorithm.

5. EXPERIMENTAL METHODOLOGY

5.1 Evaluation Data Set

To evaluate the proposed technique, we used the following three major data sets (Table 1):

MovieLens Data: This data set was created by combining the relevance judgments from the MovieLens[9] data set with documents from the Internet Movie Database (IMDB). MovieLens allows users to rank how much he/she enjoyed a specific movie on a scale from 1 to 5. This "likeability" rating was used as a measurement of how relevant the document representing the corresponding movie is to the user. We considered documents with likeability scores of 4 or 5 as relevant, and documents with a score of 1 to 3 as irrelevant to the user. MovieLens provided relevance judgments on 3,057 documents from 6,040 separate users. On average, each user rated 151 movies, of these 87 were judged to be relevant. The average score for a document was 3.58. Documents representing each movie were constructed from the portion of the IMDB database that is available for public download[13]. Based on this database, we created one document per movie that contained the relevant information about it (e.g. directors, actors, etc.).

Netflix Data: This data set was constructed by combining documents about movies crawled from the web with a set of actual movie rental customer relevance judgments from Netflix[19]. Netflix publicly provides the relevance judgments of 480,189 anonymous customers.

Table 1: Data Set Statistics. On Reuters, the number of rating for a simulated user is the number of documents relevant to the corresponding topic.

Data	Users	Docs	Ratings per User
MovieLens	6,040	3,057	151
Netflix-all	480,189	17,770	208
Netflix-1000	1000	17,770	127
Reuters-C	34	100,000	3949
Reuters-E	26	100,000	1632
Reuters-G	33	100,000	2222
Reuters-M	10	100,000	6529

There are around 100 million rating on a scale of 1 to 5 for 17,770 documents. Similar to MovieLens, we considered documents with likeability scores of 4 or 5 as relevant.

This number was reduced to 1000 customers through random sampling. The average customer on the reduced data set provided 127 judgments, with 70 being deemed relevant. The average score for documents is 3.55.

Reuters Data: This is the Reuters Corpus, Volume 1. It covers 810,000 Reuters English language news stories from August 20, 1996 to August 19, 1997. Only the first 100,000 news were used in our experiments. The Reuters corpus comes with a topic hierarchy. Each document is assigned to one of several locations on the hierarchical tree. The first level of the tree contains four topics, denoted as C, E, M, and G. For the experiments in this paper, the tree was cut at level 1 to create four smaller trees, each of which corresponds to one smaller data set: Reuters-E Reuters-C, Reuters-M and Reuters-G. For each small data set, we created several profiles, one profile for each node in a sub-tree, to simulate multiple users, each with a related, yet separate definition of relevance. All the user profiles on a sub-tree are supposed to share the same prior model distribution. Since this corpus explicitly indicates only the relevant documents for a topic(user), all other documents are considered irrelevant.

5.2 Evaluation

We designed the experiments to answer the following three questions:

1. Do we need to take the effort to use a Bayesian approach and learn a prior from other users?
2. Does the new algorithm work better than the standard EM algorithm for learning the Bayesian hierarchical linear model?
3. Can the new algorithm quickly learn many user models?

To answer the first question, we compared the Bayesian hierarchical models with commonly used Norm-2 regularized linear regression models. In fact, the commonly used approach is equivalent to the model learned at the end of the first EM iteration. To answer the second question, we compared the proposed new algorithm with the standard EM algorithm to see whether the new learning algorithm is better. To answer the third question, we tested the efficiency of the new algorithm on the entire Netflix data set where about half a million user models need to be learned together.

For the MovieLens and Netflix data sets, algorithm effectiveness was measured by mean square error, while on the Reuters data set classification error was used because it was more informative. We first evaluated the performance on each individual user, and then estimated the macro average over all users. Statistical tests (t-tests) were carried out to see whether the results are significant.

For the experiments on the MovieLens and Netflix data sets, we used a random sample of 90% of each user for training, and the rest for testing. On Reuters' data set, because there are too many relevant documents for each topic in the corpus, we used a random sample of 10% of each topic for training, and 10% of the remaining documents for testing. For all runs, we set $(a, b, c, \Sigma_\epsilon) = (0.1, 10, 0.1, 1)$ manually.

6. EXPERIMENTAL RESULTS

Figure 2, Figure 3, and Figure 4 show that on all data sets, the Bayesian hierarchical modeling approach has a statistical significant improvement over the regularized linear regression model, which is equivalent to the Bayesian hierarchical models learned at the first iteration. Further analysis shows a negative correlation between the number of training data for a user and the improvement the system gets. This suggests that the borrowing information from other users has more significant improvements for users with less training data, which is as expected. However, the strength of the correlation differs over data sets, and the amount of training data is not the only characteristics that will influence the final performance.

Figure 2 and Figure 3 show that the proposed new algorithm works better than the standard EM algorithm on the Netflix and MovieLens data sets. This is not surprising since the number of related feature-users pairs is much smaller than the number of unrelated feature-user pairs on these two data sets, and thus the proposed new algorithm is expected to work better.

Figure 4 shows that the two algorithms work similarly on the Reuters-E data set. The accuracy of the new algorithm is similar to that of the standard EM algorithm at each iteration. The general patterns are very similar on other Reuters' subsets. Further analysis shows that only 58% of the user-feature pairs are unrelated on this data set. Since the number of unrelated user-feature pairs is not extremely large, the sparseness is not a serious problem on the Reuters data set. Thus the two learning algorithms perform similarly. The results suggest that only on a corpus where the number of unrelated user-feature pairs is much larger than the number of related pairs, such as on the Netflix data set, the proposed technique will get a significant improvement over standard EM. However, the experiments also show that when the assumption does not hold, the new algorithm does not hurt performance.

Although the proposed technique is faster than standard EM, can it really learn millions of user models quickly? Our results show that the modified EM algorithm converges quickly, and 2 - 3 modified EM iterations would result in a reliable estimation. We evaluated the algorithm on the

Figure 2: Performance on a Netflix subset with 1,000 users. The new algorithm is statistical significantly better than EM algorithm at iterations 2 - 10. Norm-2 regularized linear models are equivalent to the Bayesian hierarchical models learned at the first iteration, and are statistical significantly worse than the Bayesian hierarchical models.

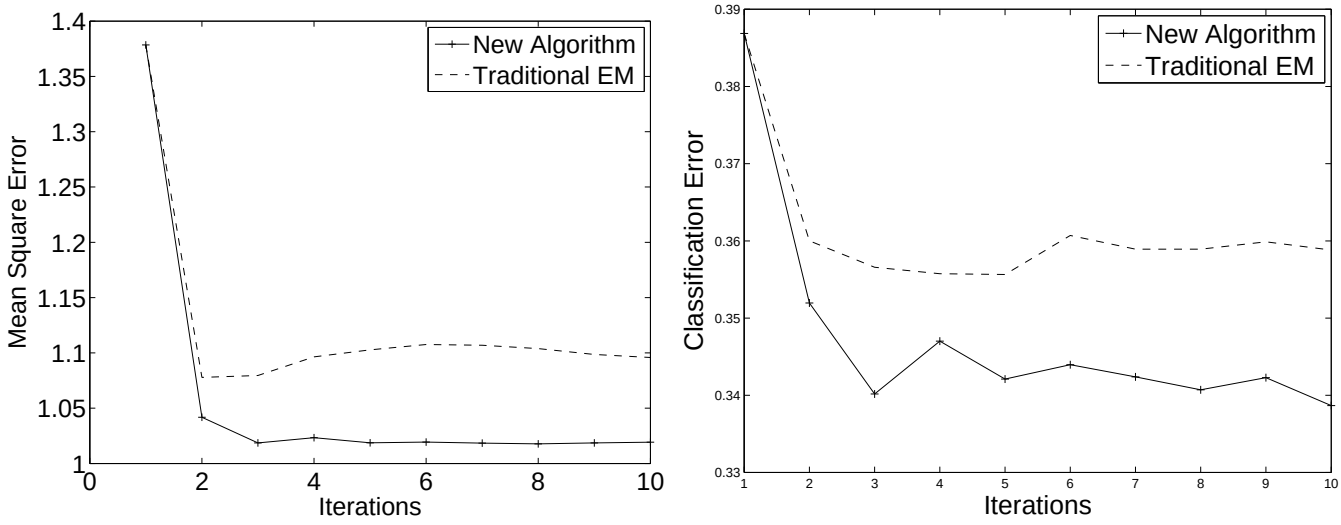


Figure 3: Performance on a MovieLens subset with 1,000 users. The new algorithm is statistical significantly better than EM algorithm at iteration 2 to 17 (evaluated with mean square error).

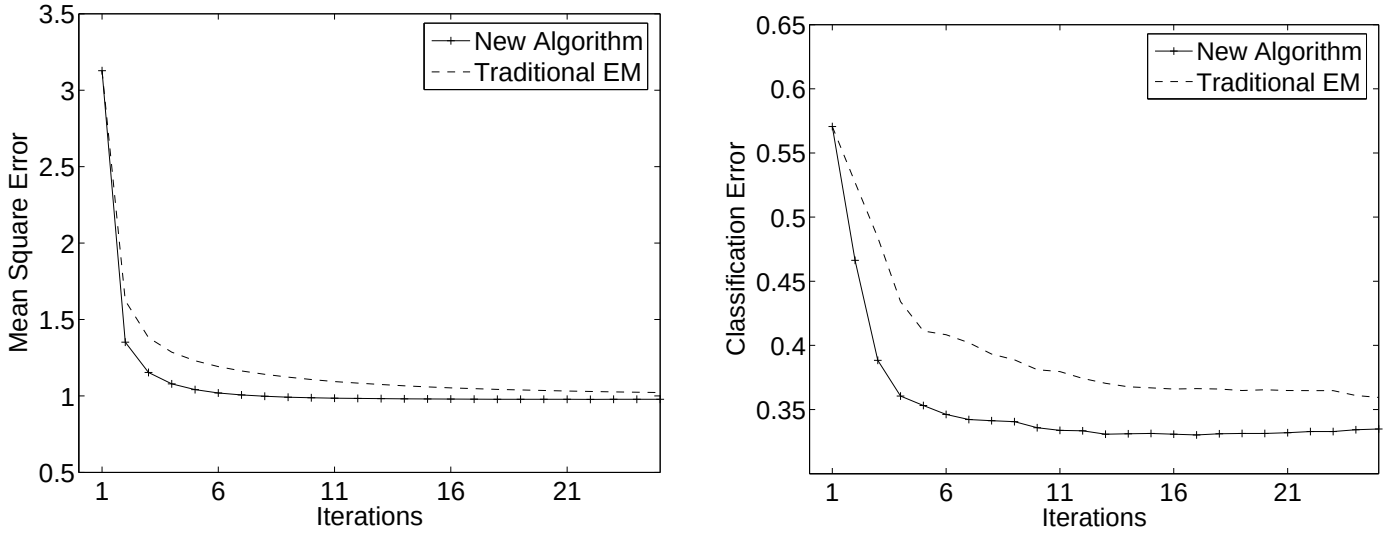
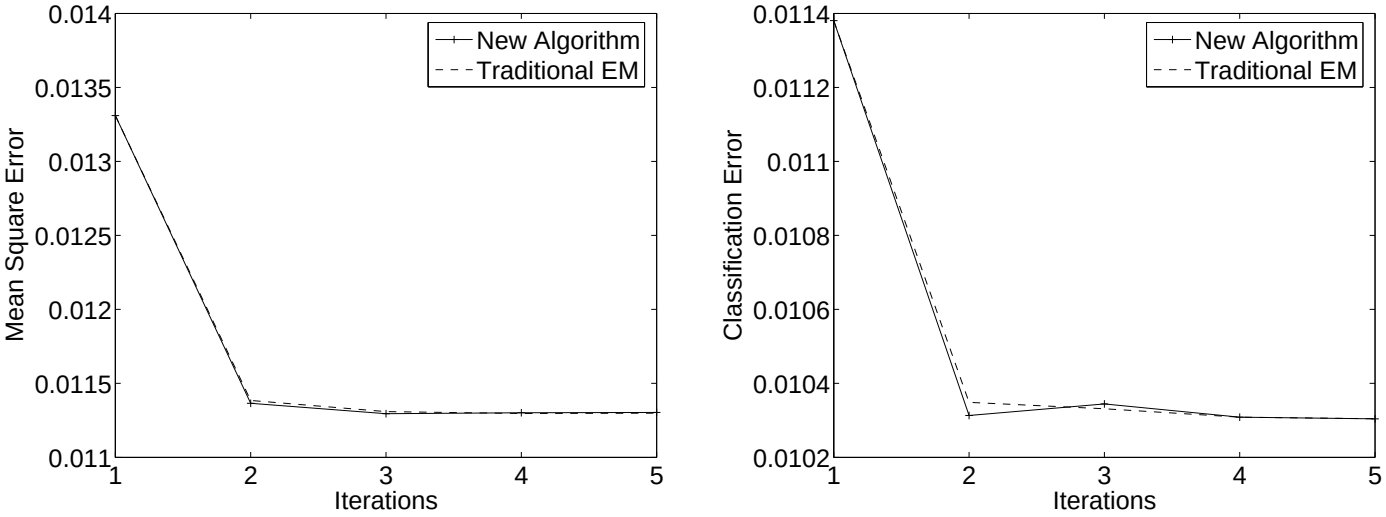


Figure 4: Performance on a Reuters-E subset with 26 profiles. Performances on Reuters-C, Reuters-M, Reuters-G are similar.



whole Netflix data set (480,189 users, 159,836 features, and 100 million ratings) running on a single CPU PC (2GB memory, P4 3GHz). The system finished one modified EM iteration in about 4 hours. This demonstrates that the proposed technique can efficiently handle large-scale system like Netflix.

7. CONCLUSION

Content-based user profile learning is an important problem and is the key to providing personal recommendations to a user, especially for recommending new items with a small number of ratings. The Bayesian hierarchical modeling approach is becoming an important user profile learning approach due to its theoretically justified ability to help one user through information transfer from the other users by way of hyperpriors.

This paper examined the weakness of the popular EM based learning approach for Bayesian hierarchical linear models and proposed a better learning technique called Modified EM. We showed that the new technique is theoretically more computationally efficient than the standard EM algorithm. Evaluation on the MovieLens and Netflix data sets demonstrated the effectiveness of the new technique when the data is sparse, by which we mean the ratio of related user-feature pairs to unrelated pairs is small. Evaluation on the Reuters data set showed that the new technique performed similar to the standard EM algorithm when the sparseness condition does not hold. In general, it is better to use the new algorithm since it is as simple as standard EM, the performance is either better or similar to EM, and the computation complexity is lower at each iteration. It is worth mentioning that even if the original problem space is not sparse, sparseness can be created artificially when a recommendation system uses user-specific feature selection techniques to reduce the noise and user model complexity. The proposed technique can also be adapted to improve the learning in such a scenario. We also demonstrated that the proposed technique can learn half a million user profiles from 100 million ratings in a few hours with a single CPU.

The research is important because scalability is a major concern for researchers when using the Bayesian hierarchical linear modeling approach to build a practical large scale system, even though the literature have demonstrated the effectiveness of the models in many applications. Our work is one major step on the road to make Bayesian hierarchical linear models more practical. The proposed new technique can be easily adapted to run on a cluster of machines, and thus further speed up the learning process to handle a larger scale system with hundreds of millions of users.

The research has much potential to benefit people using EM algorithm on many other IR problems as well as machine learning problems. EM algorithm is a commonly used machine learning technique. It is used to find model parameters in many IR problems where the training data is very sparse. Although we are focusing on the Bayesian hierarchical linear models for recommendation and filtering, the new idea of using analytical solution instead of numerical solution for unrelated user-feature pairs at the M step could be adapted to many other problems.

8. ACKNOWLEDGMENTS

We thank Wei Xu, David Lewis and anonymous reviewers for valuable feedback on the work described in this paper. Part of the work was supported by Yahoo, Google, the Petascale Data Storage Institute and the Institute for Scalable Scientific Data Management. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors, and do not necessarily reflect those of the sponsors.

9. REFERENCES

[1] C. Basu, H. Hirsh, and W. Cohen. Recommendation as classification: Using social and content-based information in recommendation. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, 1998.

[2] J. S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. Technical report, Microsoft Research, One Microsoft Way, Redmond, WA 98052, 1998.

[3] J. Callan. Document filtering with inference networks. In *Proceedings of the Nineteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 262–269, 1996.

[4] N. Cancedda, N. Cesa-Bianchi, A. Conconi, C. Gentile, C. Goutte, T. Graepel, Y. Li, J. M. Renders, J. S. Taylor, and A. Vinokourov. Kernel method for document filtering. In *The Eleventh Text REtrieval Conference (TREC11)*. National Institute of Standards and Technology, special publication 500-249, 2003.

[5] C. Chelba and A. Acero. Adaptation of maximum entropy capitalizer: Little data can help a lot. In D. Lin and D. Wu, editors, *Proceedings of EMNLP 2004*, pages 285–292, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[6] B. Croft and J. Lafferty, editors. *Language Modeling for Information Retrieval*. Kluwer, 2002.

[7] A. Dayanik, D. D. Lewis, D. Madigan, V. Menkov, and A. Genkin. Constructing informative prior distributions from domain knowledge in text classification. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 493–500, New York, NY, USA, 2006. ACM Press.

[8] J. Delgado and N. Ishii. Memory-based weightedmajority prediction for recommender systems. In *ACM SIGIR'99 Workshop on Recommender Systems*, 1999.

[9] GroupLens. Movielens. http://www.grouplens.org/taxonomy/term/14, 2006.

[10] D. Heckerman. A tutorial on learning with bayesian networks. In M. Jordan, editor, *Learning in Graphical Models*. Kluwer Academic, 1998.

[11] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 230–237, New York, NY, USA, 1999. ACM Press.

[12] T. Hofmann and J. Puzicha. Latent class models for

collaborative filtering. In *IJCAI '99: Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, pages 688–693, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.

[13] I. M. D. (IMDB). Internet movie database. http://www.imdb.com/interfaces/, 2006.

[14] R. Jin, J. Y. Chai, and L. Si. An automatic weighting scheme for collaborative filtering. In *SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 337–344, New York, NY, USA, 2004. ACM Press.

[15] J. A. Konstan, B. N. Miller, D. Maltz, J. L. Herlocker, L. R. Gordon, and J. Riedl. GroupLens: Applying collaborative filtering to Usenet news. *Communications of the ACM*, 40(3):77–87, 1997.

[16] D. Lewis. Applying support vector machines to the TREC-2001 batch filtering and routing tasks. In *Proceedings of the Eleventh Text REtrieval Conference (TREC-11)*, 2002.

[17] B. Liu, X. Li, W. S. Lee, , and P. Yu. Text classification by labeling words. In *Proceedings of The Nineteenth National Conference on Artificial Intelligence (AAAI-2004)*, July 25-29, 2004.

[18] P. Melville, R. J. Mooney, and R. Nagarajan. Content-boosted collaborative filtering for improved recommendations. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence (AAAI-2002)*, Edmonton, Canada, 2002.

[19] Netflix. Netflix prize. http://www.netflixprize.com (visited on Nov. 30, 2006), 2006.

[20] S. Robertson and K. Sparck-Jones. Relevance weighting of search terms. In *Journal of the American Society for Information Science*, volume 27, pages 129–146, 1976.

[21] J. Wang, A. P. de Vries, and M. J. T. Reinders. Unifying user-based and item-based collaborative filtering approaches by similarity fusion. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 501–508, New York, NY, USA, 2006. ACM Press.

[22] X. Wu and R. K. Srihari. Incorporating prior knowledge with weighted margin support vector machines. In *Proc. ACM Knowledge Discovery Data Mining Conf.(ACM SIGKDD 2004)*, Aug. 2004.

[23] Y. Yang, S. Yoo, J. Zhang, and B. Kisiel. Robustness of adaptive filtering methods in a cross-benchmark evaluation. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2005.

[24] K. Yu, V. Tresp, and A. Schwaighofer. Learning gaussian processes from multiple tasks. In *ICML '05: Proceedings of the 22nd international conference on Machine learning*, pages 1012–1019, New York, NY, USA, 2005. ACM Press.

[25] K. Yu, V. Tresp, and S. Yu. A nonparametric hierarchical bayesian framework for information filtering. In *SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 353–360. ACM Press, 2004.

[26] X. Zhu. Semi-supervised learning literature survey. Technical report, University of Wisconsin – Madison, December 9, 2006.

[27] P. Zigoris and Y. Zhang. Bayesian adaptive user profiling with explicit & implicit feedback. In *Conference on Information and Knowledge Mangement 2006*, 2006.