A Unified Approach to Building Hybrid Recommender Systems

Asela Gunawardana
Microsoft Research
One Microsoft Way
Redmond, WA 98052
aselag@microsoft.com

Christopher Meek
Microsoft Research
One Microsoft Way
Redmond, WA 98052
meek@microsoft.com

ABSTRACT

Content-based recommendation systems can provide recommendations for "cold-start" items for which little or no training data is available, but typically have lower accuracy than collaborative filtering systems. Conversely, collaborative filtering techniques often provide accurate recommendations, but fail on cold start items. Hybrid schemes attempt to combine these different kinds of information to yield better recommendations across the board.

We describe unified Boltzmann machines, which are probabilistic models that combine collaborative and content information in a coherent manner. They encode collaborative and content information as features, and then learn weights that reflect how well each feature predicts user actions. In doing so, information of different types is automatically weighted, without the need for careful engineering of features or for post-hoc hybridization of distinct recommender systems.

We present empirical results in the movie and shopping domains showing that unified Boltzmann machines can be used to combine content and collaborative information to yield results that are competitive with collaborative technique in recommending items that have been seen before, and also effective at recommending cold-start items.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*information filtering*; G.3 [**Probability and Statistics**]: *correlation and regression analysis*; I.2.6 [**Artificial Intelligence**]: Learning—*parameter learning*

General Terms

Algorithms, Performance

Keywords

recommender systems, collaborative filtering, content-based filtering, cold start, Boltzmann machines

1. INTRODUCTION

Recommender systems suggest items of interest to users based on available information such as previous usage patterns, the usage patterns of other users, and features of the items themselves [20]. Collaborative filtering techniques provide recommendations to a user by using the preferences of other users that have similar preferences to him [5, 17, 28]. For example, the familiar Amazon item-to-item system [17] recommends items for a user viewing a current item on the basis of other items purchased by other users that have viewed the current item. Such systems have the drawback that they suffer from the item cold start problem—an item cannot be recommended until it has been rated number of existing users.

This problem can be alleviated by recommender systems that use information about the content of items. This information can be meta-data about the item such as actors appearing in movies for movie recommendation or information derived from the item such as counts of words in documents in the case of document recommendation [21]. Unfortunately, purely content-based approaches often do not perform as well as collaborative filtering approaches when cold-start is not an issue [8]. Hybrid systems [6] that combine collaborative and content information are therefore used.

We extend our previous work on tied Boltzmann machines [10], yielding a model that can naturally combine content and collaborative features, which we term *unified Boltzmann machines*. Both are improvements upon Boltzmann machines, which model the joint distribution of a set of binary variables through their pairwise interactions. In our context, the binary variables indicate whether or not a user has acted on each item of interest. Thus, we use Boltzmann machines to learn weights that reflect the importance of different pairwise interactions such as "people who buy item A also buy item B" in explaining usage data. Tied Boltzmann machines are variants of Boltzmann machines that explain such usage data using content features of the form "people who buy one dairy item also buy other dairy items." Because they do not use features that explicitly model interactions between individual items, they provide the same predictions for items that share the same content information, potentially reducing accuracy. Unified Boltzmann machines make use of both kinds of information. They learn weights that reflect how much each feature contributes in explaining usage data, and in doing so they automatically learn to balance and combine the different information sources to make better predictions. We present empirical results comparing the performance of untied, tied, and unified Boltzmann ma-

chines in both cold start and non-cold-start scenarios in the MovieLens and Ta-Feng shopping [13] data sets. We show that the unified Boltzmann machine provides a single unified approach that naturally combines content and collaborative features to yield competitive results in both cold-start and non-cold-start scenarios.

2. PREDICTING USER ACTIONS

In this paper, we examine the problem of predicting the user's future actions on items given the user's past actions on items. We will restrict attention to binary actions such as listening to a song, watching a movie, or buying a book. Extending our models to more general actions such as rating a movie with an integer between 1 and 5 or spending $3.99 on rice is left for future work.

Our goal is to predict the probability that a user will act on an item i given whether or not he has acted on all other items. In other words, we wish to obtain a good estimate of $p(a_i|a_{-i})$ where a is a column vector of binary variables a_i that specify whether or not a user has acted on an item i, and a_{-i} is shorthand for $\{a_{i'}|i' \neq i\}$. We learn models of the interactions between actions on items from the action vectors $a^{(u)}$ of a population of users $u = 1, \cdots, M$. For example, the models will learn that people who buy hot dogs may also buy hot dog buns, or that people who watch the movie *Sleepless In Seattle* may also watch the movie *You've Got Mail*. Given such models, as yet unused items i can be recommended if they have high $p(a_i^{(u)}|a_{-i}^{(u)})$.

2.1 Modeling Item Interactions with Boltzmann Machines

Instead of estimating separate conditional distributions $p(a_i^{(u)}|a_{-i}^{(u)})$ for all items i, we model the joint distribution of a user's action vector a by a Boltzmann machine:

$$p(a;\ \lambda) = \frac{1}{z(\lambda)}\ \exp\left(\sum_i \lambda_i a_i + \sum_{i<j} \lambda_{ij} a_i a_j\right) \qquad (1)$$

The parameter vector λ has components λ_i and λ_{ij} corresponding to all items i and item pairs (i, j). We use $\sum_{i<j}$ to denote the sum over unordered pairs (i, j). Since λ_{ij} is only defined for $i < j$, we abuse notation somewhat and use λ_{ij} and λ_{ji} interchangeably to denote the weight associated with the unordered pair (i, j) without introducing any ambiguity. The *partition function* $z(\lambda)$ is a normalizer that ensures that $p(a;\ \lambda)$ is properly normalized over all configurations of the action vector a. The pairwise weights λ_{ij} capture pairwise collaborative effects—a high value of λ_{ij} indicates that users with $a_i = 1$ also tend to have $a_j = 1$. The per-item weights λ_i capture popularity effects—a high value of λ_i indicates a higher likelihood of $a_i = 1$ irrespective of the user's other actions

Example 1. Let us restrict attention to an inventory of three movies: *Sleepless in Seattle*, *You've Got Mail*, and *When Harry Met Sally*. The variables a_{SS}, a_{YGM} and a_{WHMS} will represent whether the user watched each of these movies respectively. The Boltzmann machine relating these variables will have a parameter vector λ consisting of 3 unary weights λ_{SS}, $\cdots$, λ_{WHMS} and 3 pairwise weights $\lambda_{SS,YGM}$, $\cdots$, $\lambda_{YGM,WHMS}$. A high value of λ_{SS} would indicate that a_{SS} tends to be 1—i.e. that many users watch *Sleepless in Seattle*. A high value of $\lambda_{SS,YGM}$ would indicate that a_{SS} and a_{YGM} are correlated—i.e. that users who watch *Sleepless in Seattle* also tend to watch *You've Got Mail*.

Note that the partition function $z(\lambda)$ is expensive to compute [15]. It requires summing over all 2^N configurations of the action vector. However, our desired conditional probability of a user acting on a single item given his actions on all other items is easy to compute, and is given by

$$p(a_i = 1|a_{-i};\ \lambda) = \frac{\exp\left(\lambda_i + \sum_{j\neq i|a_j=1} \lambda_{ij}\right)}{1 + \exp\left(\lambda_i + \sum_{j\neq i|a_j=1} \lambda_{ij}\right)}. \qquad (2)$$

This conditional distribution can be seen to be a logistic regression of a_i on a_{-i}. Because the logistic regressions for predicting each a_i come from the single joint model given by the Boltzmann machine (1), the weight for regressing a_i on a_j and the weight for regressing a_j on a_i are constrained to be equal for every item pair (i, j).

2.2 Content-Based Parameter Tying

One weakness of the model discussed in section 2.1 is that reliable estimation of the model parameters λ, in particular the pairwise weights λ_{ij}, requires sufficiently many observations. Since real data tends to be sparse, with many items having low probability of occurrence, pairwise weights involving these items are difficult to estimate in practice. Thus, the Boltzmann machine model described above may not make good predictions for such items. In the extreme case where an item is not seen during training, we have the item cold start problem, where none of the weights associated with that item can be reliably estimated.

We alleviate this difficulty through the use of tied Boltzmann machines. Tied Boltzmann machines are Boltzmann machines where the parameters have been tied so that they are no longer estimated independently. When the parameters are tied, the data used to estimated them can be pooled, allowing more reliable estimation.

In order to retain the ability to model the interactions between items, we use content information to guide the parameter tying. We assume that the content associated with item i is represented by a feature vector $f^{(i)} \in \mathbb{R}^D$ composed of D components. For example, features could be counts or TF-IDF weights of words in documents, or binary flags indicating whether specific actors appeared in a movie. Features with different semantics could be combined in a single vector. For example, some feature components could correspond to actors in a movie, while others could correspond to genres, while still others could take on numerical values such as movie length in minutes. We constrain λ to satisfy

$$\lambda_i = \mu^T f^{(i)} \qquad (3)$$

$$\lambda_{ij} = {f^{(i)}}^T \eta f^{(j)} \qquad (4)$$

where $\mu \in \mathbb{R}^D$ and $\eta \in \mathbb{R}^{D\times D}$ is symmetric. In this paper, we will only consider diagonal η, in order to reduce the number of parameters that need be estimated, and use η_k to denote the kth diagonal component of η.

Example 2. Let us tie the parameters of Example 1 using a 2-dimensional feature vector with the components f_{MR} and f_{TH} corresponding to Meg Ryan and Tom Hanks appearing in a movie. The new parameters that correspond to

this case are μ_{MR}, μ_{TH} which reflect how much Meg Ryan and Tom Hanks contribute to the popularity of a movie, and η_{MR}, η_{TH} which reflect how much having Meg Ryan and Tom Hanks respectively as a common actor contribute to a pair of movies both being watched by the same user. Note that the number of parameters has been reduced from 6 to 4. In addition, note that because *Sleepless in Seattle* and *You've Got Mail* both have Tom Hanks and Meg Ryan in the cast, we have $\lambda_{SS} = \lambda_{YGM} = \mu_{MR} + \mu_{TH}$. These weights have been tied together by the content information. Similarly $\lambda_{SS,WHMS} = \lambda_{YGM,WHMS} = \eta_{MR}$ because the pair *Sleepless in Seattle* and *When Harry Met Sally* and the pair *You've Got Mail* and *When Harry Met Sally* both share the single common actor Meg Ryan. If we had modeled off-diagonal terms in η, we would also have had parameters such as $\eta_{MR,TH}$ which models how much watching movies starring Meg Ryan makes a user more likely to also watch movies starring Tom Hanks.

In effect, we have reparametrized the Boltzmann machine, so that the joint distribution of equation (1) is now parametrized through through μ and η:

$$p(a;\ \mu,\eta) = \frac{1}{z(\mu,\eta)} \times \exp\left(\sum_i \sum_k \mu_k f_k^{(i)} a_i + \sum_{i<j} \sum_k \eta_k f_k^{(i)} f_k^{(j)} a_i a_j\right) \quad (5)$$

where $z(\mu,\eta) = z(\lambda(\mu,\eta))$.

The conditional probability of a user acting on item i given her actions a_{-i} on all other items can be efficiently computed as

$$p(a_i = 1 | a_{-i};\ \mu,\eta) = \frac{\exp\left(\sum_k \mu_k f_k^{(i)} + \sum_{j\neq i | a_j=1} \left(\sum_k \eta_k f_k^{(i)} f_k^{(j)}\right)\right)}{1 + \exp\left(\sum_k \mu_k f_k^{(i)} \sum_{j\neq i | a_j=1} \left(\sum_k \eta_k f_k^{(i)} f_k^{(j)}\right)\right)} \quad (6)$$

In contrast to equation (2), this conditional distribution is a logistic regression of a_i on the features of the item i and the common features of item i and other items j with $a_j = 1$, instead of on i and other such items j directly.

2.3 Unifying Untied and Tied Boltzmann Machines

One drawback of the parameter tying scheme described above is that the approximate estimates of equation (4) are used even if good untied estimates of λ_{ij} and λ_i can be made from the training data. Thus, the use of tied Boltzmann machines can lead to loss of expressive power. In addition, it is possible for the estimates of η_k to be heavily influenced by a few common item pairs (i,j) with non-zero $f_k^{(i)} f_k^{(j)}$, leading the resulting tied estimate of λ_{ij} to be not as applicable to other item pairs. Similarly the estimates of μ_k can be dominated by a few common items so the the resulting tied estimates of λ_i for other items may not be as applicable to other items.

One solution to these problems is to use a unified Boltzmann machine with the reparametrization

$$\lambda_i = \mu^T f^{(i)} + \delta_i \quad (7)$$

$$\lambda_{ij} = {f^{(i)}}^T \eta f^{(j)} + \delta_{ij} \quad (8)$$

where the new parameters δ_i and δ_{ij} explicitly model the residuals in the tied estimates of λ_i and λ_{ij}. The residual parameters will be estimated when they can be estimated reliably, and be kept close to zero when they cannot, via the use of regularization as described below in section 2.4. Thus, the tied estimates are corrected when reliable estimates of λ_i and λ_{ij} can be obtained, and conversely, the influence of common items or item-pairs in estimating μ or η is corrected by the residual terms.

Example 3. We now unify the parametrizations of examples 1 and 2. We will now use the 4 content parameters μ_{MR}, μ_{TH}, η_{MR} and η_{TH}, 3 unary item parameters δ_{SS}, $\cdots$, δ_{WHMS} and 3 pairwise item parameters $\delta_{SS,YGM}$, $\cdots$, $\delta_{YGM,WHMS}$. We now have

$$\lambda_{SS} = \mu_{MR} + \mu_{TH} + \delta_{SS}$$
$$\lambda_{YGM} = \mu_{MR} + \mu_{TH} + \delta_{YGM}$$

and

$$\lambda_{SS,WHMS} = \eta_{MR} + \delta_{SS,WHMS}$$
$$\lambda_{YGM,WHMS} = \eta_{MR} + \delta_{YGM,WHMS}$$

In contrast to example 2 the existence of the parameters δ means that the weights λ are no longer constrained. However, they still make use of content information.

The joint distribution of a user's actions is now given by

$$p(a;\ \mu,\eta,\delta) = \frac{1}{z(\mu,\eta,\delta)} \times \exp\left(\sum_i \left(\sum_k \mu_k \sum_i f_k^{(i)} + \delta_i\right) a_i + \sum_{i<j} \left(\sum_k \eta_k f_k^{(i)} f_k^{(j)} + \delta_{ij}\right) a_i a_j\right) \quad (9)$$

As in the tied and untied cases, the partition function $z(\mu,\eta,\delta)$ requires exponential computation, but the conditional probability of a user acting on item i given her actions a_{-i} on all other items can be efficiently computed as

$$p(a_i = 1 | a_{-i};\ \mu,\eta) = \frac{\exp\left(\sum_k \mu_k f_k^{(i)} + \delta_i + \sum_{j\neq i | a_j=1} \left(\sum_k \eta_k f_k^{(i)} f_k^{(j)} + \delta_{ij}\right)\right)}{1 + \exp\left(\sum_k \mu_k f_k^{(i)} + \delta_i + \sum_{j\neq i | a_j=1} \left(\sum_k \eta_k f_k^{(i)} f_k^{(j)} + \delta_{ij}\right)\right)} \quad (10)$$

This conditional probability is also a logistic regression, this time on both the items and their features.

We note in passing that predictions can easily be conditioned on other kinds of information by adding features that depend on that information to this model. For example, if user age groups are available in the movie domain, features such as "user is under 10 and the genre is cartoons" can be used.

For convenience, we will denote the joint distribution given by the untied, tied, and unified Boltzmann machines given in equations (1), (5) and (9) as $p(a;\ \theta)$ and the conditional distributions of equations (2), (6) and (10) as $p(a_i | a_{-i};\ \theta)$ where the parameter vector θ contains components corresponding to λ_i or δ_i, λ_{ij} or δ_{ij}, and μ_k, η_k as appropriate.

2.4 Training the Models

The models are trained from a training set consisting of $M \times N$ binary variables $a_i^{(u)}$ for all N items $i \in 1, \cdots, N$ and all M users $u \in 1, \cdots, M$, which specifies whether or not each user acted on each item. As discussed above, all items a user has not yet read, viewed, bought, etc., depending on the application, will have $a_i^{(u)} = 0$. In addition, tied and unified Boltzmann machines will use a content database which specifies the feature vectors $f^{(i)}$ for all items. Modeling user action vectors as being i.i.d.[1], the log likelihood of the training set is given by

$$\sum_u \log p(a^{(u)};\ \theta) \tag{11}$$

This log likelihood is hard to optimize because it requires the evaluation of the partition function $z(\theta)$ for each candidate value of θ.

Because of this, we approximate the log likelihood of the training set by its log pseudo-likelihood [3, 14]:

$$\sum_u \log p(a^{(u)};\ \theta) \approx \sum_u \sum_i \log p(a_i^{(u)} | a_{-i}^{(u)};\ \theta) \tag{12}$$

As discussed above, the conditional likelihood $p(a_i|a_{-i})$ is easily computed for all three models. If we treat each user u's action on each item i as an independent training event given u's other actions, the pseudo-likelihood can be interpreted as the conditional likelihood of the training set under a conditional model given by equation (2), (6), or (10). We note that the pseudo-likelihood is convex in the parameters, and therefore easily optimized via gradient methods.

In order to get robust estimates, we train the parameters θ by maximizing the regularized pseudo-likelihood

$$\mathcal{L}(\mu, \eta) = \sum_u \sum_i \log p(a_i^{(u)} | a_{-i}^{(u)};\ \theta) - \alpha ||\theta||^2 \tag{13}$$

The regularization parameter α prevents parameters from growing (and therefore having a large influence on the model) without significantly improving the likelihood of the data. This can be viewed as MAP estimation of the set of tied logistic regressions mentioned above under a Gaussian prior [7]. Typically, when α is high, parameters are not allowed to grow unless they better explain a large portion of the training data.

2.5 Sparse Training Through Stratified Sampling

With large data sets, optimizing the training criterion of equation (13) may be too expensive since it requires computing a conditional likelihood for each of $M \times N$ user-item pairs, where M and N may both be large. Since the set of items for which $a_i^{(u)} = 1$ is sparse, a further speedup is possible. We will denote this sparse set of positive training items by $I_+^{(u)}$ and its complement by $I_-^{(u)}$. A sampled set of negative items $I_-^{(u)}(p_s)$ will be chosen by sampling items from $I_-^{(u)}$ with probability p_s [18]. We choose p_s so that the total number of sampled negative items $\sum_u |I_-^{(u)}(p_s)|$ is on the order of the total number of positive items $\sum_u |I_+^{(u)}|$.

[1]If user information such as demographics is available, we would model user action vectors as being i.i.d. given the user information

The regularized pseudo-likelihood of equation (13) is then approximated through stratified sampling by

$$\mathcal{L}(\mu, \eta) \approx \sum_u \Bigg(\sum_{i \in I_+^{(u)}} \log p(a_i = 1 | a_{-i}^{(u)};\ \theta) + \frac{1}{p_s} \sum_{i \in I_-^{(u)}(p_s)} \log p(a_i = 0 | a_{-i}^{(u)};\ \theta) \Bigg) - \alpha ||\theta||^2 \tag{14}$$

Now, evaluating the training criterion only requires computing the conditional likelihood for $O(A)$ items, where A is the total number of actions observed. Of course, the speedup depends on how sparse actions are, and the loss in accuracy is data dependent. This approximate training criterion, like the exact criterion of equation (13), can be optimized using gradient methods.

In our experiments, we optimized the training criterion using the RPROP [25] algorithm. This is a non-linear gradient technique that maintains an adaptive step size for each dimension. The step size during optimization was initialized to the inverse dimensionality of the parameter vector, the step growth and shrinkage parameters of the RPROP algorithm were set to 1.2 and 0.5, while backtracking was enabled. The algorithm was run for at most 500 iterations. The regularization parameter α was chosen by optimizing likelihood on a held-out set, but we observed that the results were not very sensitive to this value. The sampling parameter p_s was set to $\frac{|A|}{MN - |A|}$. We note that past work suggested that sampled training results in little accuracy loss [10]. Finally, in order to control memory utilization, we use count cutoffs as follows. First, we count the number of users for whom each item, item-pair, and content vector component $f_j^{(i)}$ occur in the training set. We then select the ones that occur only a small number of times and drop the corresponding parameters from the model.

3. EXPERIMENTAL EVALUATION

We evaluated our models on two data sets—the "1,000,000 rating" MovieLens movie rating corpus (www.movielens.org) and the Ta-Feng supermarket corpus (aiia.iis.sinica.edu.tw) [13]. We evaluated the models using a "random" and "cold-start" protocol on each of these data sets, as described below.

3.1 MovieLens Data

This data set contains 1,000,209 ratings of 3,706 movies by 6,040 users. Not all movies are rated by all users. Movies are tagged with 18 genres. A movie can belong to more than one genre. The MovieLens movie description pages were crawled to obtain the headlining actors of each of these movies. This yielded 8406 actors. The task in experiments on this data was to predict which movies users rated given other movies they rated. More formally, we set $a_i^{(u)} = 1$ if user u had rated movie i. For tied and unified Boltzmann machines, we used binary feature vectors $f^{(i)}$ with components corresponding to actors and genres. We present results comparing the use of genre only, actors only, and both genre and actors.

For the "random" protocol, we randomly selected 800,167 (80%) of the ratings to form training partition, with the remaining 200,042 forming a test partition. We then selected the test ratings of 500 randomly selected users, yielding a test set of $3{,}706 \times 500 = 1{,}853{,}000$ possible recommendations,

16,310 of which corresponded to actual ratings. For the "cold-start" protocol, we randomly drew 2,962 (80%) of the movies to be training movies and remaining 744 to be test movies. We randomly selected 500 users to evaluate recommendations on, yielding a test set of $744 \times 500 = 372{,}000$ possible recommendations, 16,717 of which corresponded to actual ratings. The training set consisted only of 816,527 ratings for the 2,962 training movies. Since users tend to rate movies they have watched, we believe that the results of these experiments are suggestive of how well our models would predict movie watching.

3.2 Ta-Feng Supermarket Data

This is a data set containing the purchase records of 32,267 supermarket customers. A total inventory of 23,812 distinct items were purchased. We ignored repeat purchases, yielding 743,228 total items bought. Each item is categorized according to a three-level hierarchy with 26 coarse categories, 201 medium categories and 2,012 fine categories. Each item belongs to exactly one coarse, medium, and fine category each. We set $a_i^{(u)} = 1$ if user u had purchased item i. For tied and unified Boltzmann machines, we used binary feature vectors $f^{(i)}$ with components corresponding to coarse, medium, and fine categories. We present results comparing the use of coarse categories only, coarse and medium categories, and all categories.

For the "random" protocol, we randomly selected 594,582 (80%) of the purchases to train on. From the remaining 148,646 purchases, we then selected those of 500 randomly selected users, yielding a test set of $23{,}812 \times 500 = 11{,}906{,}000$ possible recommendations, only 2,621 of which corresponded to actual purchases. For the "cold-start" protocol, we randomly selected 19,050 (80%) of the items to be training items, leaving 4,762 test items. We randomly selected 500 users to evaluate recommendations on, yielding a test set of $4{,}762 \times 500 = 2{,}381{,}000$ possible recommendations, 2,915 of which corresponded to actual ratings. The training set consisted of 590,474 purchase records for the 19,050 training items.

3.3 Baselines

We compare our models to three simple baselines. When using the "random" protocol, we compared to Pearson correlation [24, 5] and to item-item recommendation [17]. For Pearson correlation based-recommendations, we used user neighborhoods of size 25, which was found to work well across a number of domains.

In the case of item-item recommendation, we used count-cutoffs to keep the item-item co-occurrence matrix sparse for memory efficiency. The count cut-offs were chosen to ensure that the item-item systems and untied Boltzmann machine systems had the same memory utilization. In order to generate item-item recommendations given a set of items, we computed the item-item similarity of each potential recommendation to each given item. We then used the maximum similarity over the given items for ranking potential recommendations.

In the case of the "cold-start" protocol we compare to the Naive Bayes approach of [21]. However, since we have binary content information rather than free text, we use a multivariate Bernoulli event model instead of a multinomial event model [19]. We investigated the use of aspect models [30] as a baseline, but they failed to improve upon the Naive Bayes approach in preliminary experiments, despite a much higher training cost.

3.4 Results

We compare the performance of our algorithms using precision of a fixed length list of recommendations. To compute this metric for an algorithm, we first rank items given the training items used by a test user using the algorithm. We then exclude the training items used by the user from the ranked items, and define the per-user precision at K recommendations to be the proportion of the top K recommendations that were actually used by the user in the test set. We average the resulting per-user precisions over all test users to get an average precision at K recommendations. The results we obtained were consistent for K = 5, 10, 25, and 50.

Figure 1 shows the average precision at 5 recommendations when evaluating with the "random" protocol on the MovieLens and Ta-Feng data sets. Among the purely collaborative algorithms, untied Boltzmann machines outperform both item-item and Pearson correlation on both data sets. Note that the performance of these algorithms does not depend on the content information used since they ignore it. Unified Boltzmann machines match or beat untied Boltzmann machines in both domains, and gives the best performance across the board. The purely content based Naive Bayes approach has the worst performance across the board. The performance of tied Boltzmann machines improve with the amount of content information provided. The results obtained at 10, 25, and 50 recommendations was consistent with the results at 5 recommendations.

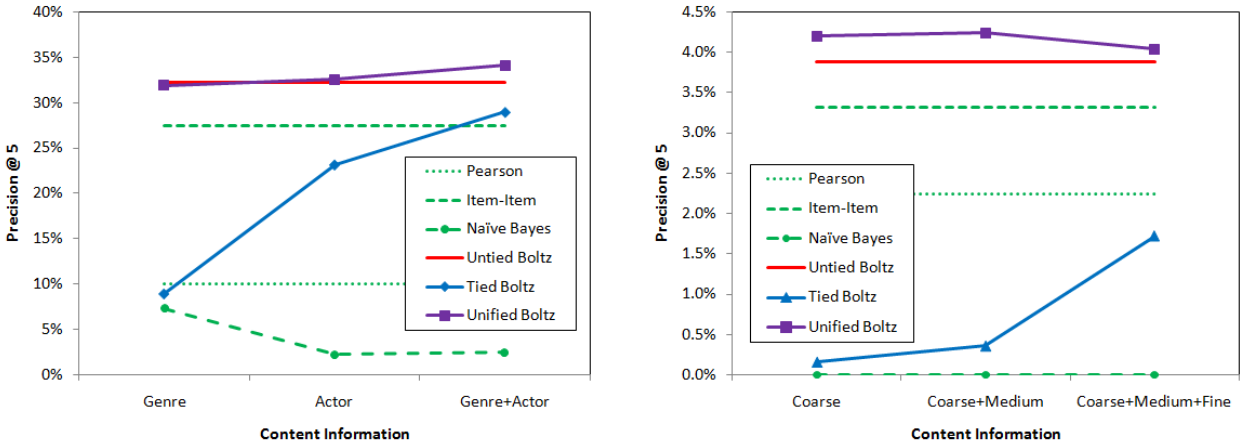
Figure 2 shows the precision at 5 recommendations when evaluating on the "cold-start" protocol. Item-item, Pearson, and untied Boltzmann machines perform no better than chance in this scenario. Note that in cold-start tests, it is not possible to use simple baselines such as recommending the most popular items, since the popularity of a new item is not yet known. Once again, unified Boltzmann machines are competitive across the board. The results obtained at 10, 25, and 50 recommendations was consistent with the results at 5 recommendations.

We note that most cases, unified Boltzmann machines benefit from additional content information, although the addition of the finest-grained category information does cause a small degradation in the Ta-Feng data when using the "random" protocol. With the MovieLens data, using both genre and actor information always outperforms using only actor information or only genre information.

Surprisingly, unified Boltzmann machines usually outperform tied Boltzmann machines even under the "cold-start" protocol, especially when the content information becomes richer, even though the item-item weights are never used during test time. We speculate that the item-item weights learn biases in the training data, allowing the content weights to generalize better to tests sets where item frequencies are different from the training set. However, verifying this conjecture is left for future work.

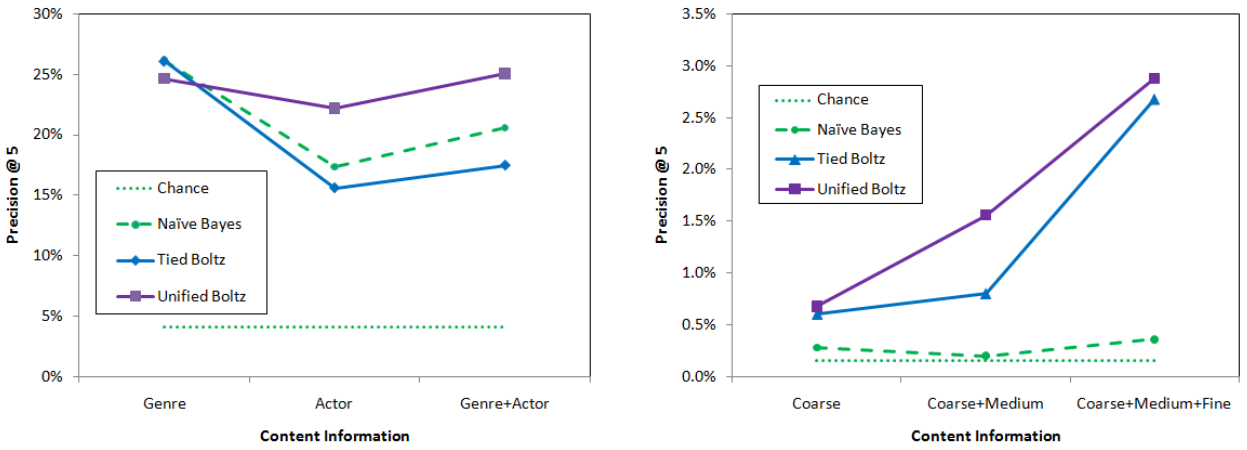
4. RELATED WORK

While our untied Boltzmann machines are pure collaborative recommenders, tied and unified Boltzmann machines are hybrid recommenders. Tied Boltzmann machines combine content and collaborative information is a somewhat subtle way–even though the model uses only content-based



(a) MovieLens (b) Ta-Feng

Figure 1: Average precision at 5 recommendations using the "random" protocol.



(a) MovieLens (b) Ta-Feng

Figure 2: Average precision at 5 recommendations using the "cold-start" protocol.

features, they are trained using collaborative data. In the terminology of Burke [6], unified Boltzmann machines are examples of "feature combination." Other approaches include combining the results of different recommenders by weighting the resulting scores [8] or voting [23], switching between different recommenders [4], and filtering or reranking the results of one recommender with another [6]. These approaches all require building separate recommender systems using techniques that are specialized to each kind of information used, and then combining the outputs of these systems. Feature combination schemes such as ours are attractive because they allow a single unified technique to be used regardless of the types of information used. However, Burke [6] points out that previous attempts at hybridization through feature combination, such as rule induction from collaborative and content data [2], require careful engineering of the features used. Our approach learns weights which reflect features are most predictive, and uses regularization to ensure that the model generalizes to unseen data, so that such engineering of the features is unnecessary.

The use of aspect models [30] has been proposed as an approach to recommendation that is robust in the face of the item cold-start problem. Extending this approach to use many types of meta-data (e.g. actors, genres, and directors for movies) was not reported to lead to improvements [29]. The Naive Bayes approach of Mooney [21] explicitly uses meta-data of different kinds in making recommendations, but does not take collaborative data into account. In our experiments, the Naive Bayes approach outperformed user-actor aspect models.

The Hybrid Poisson-Aspect Model [13] approach combines a user-item aspect model with a content-based user cluster model. Restricted Boltzmann machines [27] and dyadic data models [1] are also latent variable models, in the same spirit as soft clustering methods such as user-item aspect models [12] and matrix factorizations such as SVD, and do not attempt to directly model pairwise interactions between items. Our approach could be extended to use latent variables that model user clusters. Approaches such as restricted Boltzmann machines [26], dyadic models [1], and online SVD [16] are typically used with ratings data where most ratings are missing, and take advantage of this fact for computational tractability. In our setting, where we have an observed binary value for each user and item, and where there are large number of users and ratings, it may be intractable to use these approaches directly.

There has been some recent work on using filter-bots [9] for hybrid recommendations [22]. This work attempts to use bots implementing various heuristics, some based on content information, to generate synthetic data for training recommender systems on. Our approach learns joint models of user actions using collaborative and other information. Our models could be used as more principled "filter-bots" that generate synthetic data by sampling from the joint distributions.

Finally, we note that tied and unified Boltzmann machines can also be viewed as dependency networks [11] where the conditional probabilities are logistic regressions instead of decision trees, and where the parameters are tied across nodes. It is this parameter tying that distinguishes tied and unified Boltzmann machines from dependency networks, and that improves generalization to rarely seen items.

5. DISCUSSION

We have presented unified Boltzmann machines, an improvement over tied Boltzmann machines that allow the use of both content-based and item-specific features that allows better recommendations of both cold-start and non-cold-start items. We showed that unified Boltzmann machines outperform simple collaborative filtering techniques such as item-item recommendation and Pearson correlation in recommending non-cold-start items while being competitive with content-based recommendation in recommending cold-start items. Thus, unified Boltzmann machines provide a convenient and natural approach for combining collaborative information with collaborative information, without the need for special-purpose hybridizing techniques.

Boltzmann machines explicitly model the interactions between pairs of items. The use of Boltzmann machines to model the set of items acted on by a user assumes that any higher order interactions between these items can be explained in terms of pairwise effects. For example, while the model can learn that hot dogs and hot dog buns are bought together, it cannot explicitly learn that users typically buy hot dogs with buns of brand A or brand B, but usually not both. However, it is hoped that this higher order interaction can be learned implicitly by learning the three binary interactions "hot dogs are often bought with brand A buns," "hot dogs are often bought with brand B buns," and "brand A buns and brand B buns are seldom bought together." We note that while item-to-item recommendations also make this implicit assumption, user-based approaches such as Pearson correlation do not. The good performance of untied Boltzmann machines and item-to-item recommendation over Pearson correlation suggests that this assumption is not too harmful to make.

A number of directions for further inquiry remain. The models we have presented only predict binary user actions. The models need to be extended to predict non-binary user actions, such as numeric ratings of movies. Our models do not explicitly take temporal effects into account, even though these effects can be important in some domains [31]. Extending our models to take such effects into account is left for future work.

We have so far only investigated the combination of content and collaborative information. After training on this kind of information, the models provide an estimate that a previously unseen user will use a particular item, since the marginal probability that an item will be used is well-defined. However, this estimate is not likely to be very informative. Our models can easily be conditioned on other information such as demographics or user context in order to provide better recommendations for users whom we have little or no history. How well the models perform when combining this kind of information with content and collaborative information remains to be seen.

The ability of the models to learn to use different kinds of features while generalizing to new data depends on careful regularization of the parameters. We have so far used a simple squared length regularization of the parameters. The fact that addition of the finest-level category information caused a small degradation in the "random" protocol experiments on the Ta-Feng data indicates that there is room for improvement. Investigation of more sophisticated regularization schemes remains to be done.

In the work described here, we have used maximum pseudo-

likelihood estimation combined with stratified sampling to efficiently train our models. Investigating the use of recent advances in variational and Monte Carlo techniques for training Boltzmann machines [26] is another direction for future research.

6. ACKNOWLEDGMENTS

The authors wish to thank Guy Shani for many useful discussions, and Galen Andrews for the use of his model training software.

7. REFERENCES

[1] D. Agarwal and S. Merugu. Predictive discrete latent factor models for large scale dyadic data. In *SIGKDD*, 2007.

[2] C. Basu, H. Hirsh, and W. Cohen. Recommendation as classification: Using social and content-based information in recommendation. In *AAAI*, 1998.

[3] J. Besag. Statistical analysis of non-lattice data. *The Stat.*, 24(3):179–195, Sept. 1975.

[4] D. Billsus and M. J. Pazzani. User modeling for adaptive news access. *User Model. User-Adapt. Interact.*, 10(2-3):147–180, 2000.

[5] J. S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predctive algorithms for collaborative filtering. In *UAI*, 1998.

[6] R. D. Burke. Hybrid recommender systems: Survey and experiments. *User Model. User-Adapt. Interact.*, 12(4):331–370, 2002.

[7] S. F. Chen and R. Rosenfeld. A Gaussian prior for smoothing maximum entropy models. Technical Report CMU-CS-99-108, Carnegie Mellon University, 1999.

[8] M. Claypool, A. Gokhale, T. Miranda, P. Murnikov, D. Netes, and M. Sartin. Combining content-based and collaborative filters in an online newspaper. In *SIGIR Wkshp. Rec. Sys.*, 1999.

[9] N. Good, J. B. Schafer, J. A. Konstan, A. Borchers, B. Sarwar, J. Herlocker, and J. Riedl. Combining collaborative filtering with personal agents for better recommendations. In *AAAI*, 1999.

[10] A. Gunawardana and C. Meek. Tied Boltzmann machines for cold start recommendations. In *ACM RecSys*, 2008.

[11] D. Heckerman, D. M. Chickering, C. Meek, R. Rounthwaite, and C. Kadie. Dependency networks for inference, collaborative filtering, and data visualization. *JMLR*, 1:49–75, Oct. 2000.

[12] T. Hofmann and J. Puzicha. Latent class models for collaborative filtering. In *IJCAI*, 1999.

[13] C.-N. Hsu, H.-H. Chung, and H.-S. Huang. Mining skewed and sparse transaction data for personalized shopping recommendation. *Mach. Learn.*, 57(1-2):35–59, 2004.

[14] A. Hyvärinen. Consistency of pseudolikelihood estimation of fully visible Boltzmann machines. *Neur. Comp.*, 18:2283–2292, 2006.

[15] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to varational methods for graphical models. In M. I. Jordan, editor, *Learning in Graphical Models*, pages 105–162. MIT Press, 1999.

[16] M. Kurucz, A. A. Benczúr, and K. Csalogány. Methods for large scale SVD with missing values. In *KDD Cup*, 2007.

[17] G. Linden, B. Smith, and J. York. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Comp. Mag.*, 7(1), 2003.

[18] R. F. Lyon and L. S. Yaeger. On-line hand-printing recognition with neural networks. In *MicroNeuro*, 1996.

[19] A. McCallum and K. Nigam. A comparison of event models for naive bayes text classification. In *AAAI Wkshp. Lrn. Txt. Cat.*, 1998.

[20] M. Montaner, B. López, and J. L. de la Rosa. A taxonomy of recommender agents on the internet. *AI Rev.*, 19(4), 2003.

[21] R. J. Mooney and L. Roy. Content-based book recommending using learning for text categorization. In *ACM DL*, pages 195–204, 2000.

[22] S.-T. Park, D. Pennock, O. Madani, N. Good, and D. DeCoste. Naïve filterbots for robust cold-start recommendations. In *SIGKDD*, pages 699–705, 2006.

[23] M. J. Pazzani. A framework for collaborative, content-based and demographic filtering. *AI Rev.*, 13:393–408, 1999.

[24] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. GroupLens: an open architecture for collaborative filtering of netnews. In *ACM CSCW*, pages 175–186, 1994.

[25] M. Riedmiller and H. Braun. A direct adaptive method for faster backpropagation learning: The RPROP algorithm. In *IEEE Int. Conf. Neur. Net.*, volume 1, pages 586–591, 1993.

[26] R. Salakhutdinov. Learning and evaluating boltzmann machines. Technical Report UTML TR 2008-002, Dept. of Computer Science, Univ. of Toronto, 2008.

[27] R. Salakhutdinov, A. Mnih, and G. Hinton. Restricted Boltzmann machines for collaborative filtering. In *ICML*, 2007.

[28] B. Sarwar, G. K. nad Joseph Konstan, and J. Reidl. Item-based collaborative filtering recommendation algorithms. In *WWW*, 2001.

[29] A. I. Schein, A. Popescul, L. H. Ungar, and D. M. Pennock. Generative models for cold-start recommendations. In *SIGIR Wkshp. Recom. Sys.*, 2001.

[30] A. I. Schein, A. Popescul, L. H. Ungar, and D. M. Pennock. Methods and metrics for cold-start recommendations. In *SIGIR*. ACM, Aug. 2002.

[31] A. Zimdars, D. M. Chickering, and C. Meek. Using temporal data for making recommendations. In *UAI*, 2001.