Hybrid Collaborative Filtering Algorithms Using a Mixture of Experts

Xiaoyuan Su[1] Russell Greiner[2] Taghi M. Khoshgoftaar[1] Xingquan Zhu[1]
[1] *Computer Science and Engineering, Florida Atlantic University, Boca Raton FL 33431, USA*
[2] *Computing Science, University of Alberta, AB T6G 2E8, Canada*
xsu@fau.edu, greiner@cs.ualberta.ca, {taghi, xqzhu}@cse.fau.edu

Abstract

Collaborative filtering (CF) is one of the most successful approaches for recommendation. In this paper, we propose two hybrid CF algorithms, sequential mixture CF and joint mixture CF, each combining advice from multiple experts for effective recommendation. These proposed hybrid CF models work particularly well in the common situation when data are very sparse. By combining multiple experts to form a mixture CF, our systems are able to cope with sparse data to obtain satisfactory performance. Empirical studies show that our algorithms outperform their peers, such as memory-based, pure model-based, pure content-based CF algorithms, and the content-boosted CF (a representative hybrid CF algorithm), especially when the underlying data are very sparse.

1. Introduction

Collaborative filtering (*CF*), one of the most successful recommendation techniques to date, uses known preferences of a group of users to recommend products (e.g., movies, books, …) to new users. Collaborative filtering techniques have been popularly deployed in commercial systems such as *Amazon.com*. The recent *Netflix prize* [7] for movie recommendations has re-fueled interest in *CF* research.

Collaborative filtering techniques can be broadly classified into several categories: memory-based *CF* techniques such as the Pearson correlation-based *CF* algorithm [1][8]; model-based *CF* techniques such as Bayesian belief net *CF* algorithms [6] and clustering *CF* algorithms; and hybrid *CF* techniques such as the content-boosted *CF* algorithm [5]. Memory-based and model-based *CF* algorithms predict recommendation values based only on the rating matrix; content-based recommender systems use the regularities found within content information to make predictions; and hybrid *CF* algorithms use both content information and the rating matrix.

To be effective, a collaborative filter must deal with major challenges including sparseness of the data (that is, most people do not rate most movies), and large number of users and items (scalibility). The traditional *Pearson correlation*-based *CF* algorithm (*Pearson CF*), a pure memory-based *CF* algorithm, addresses the scalability problem by calculating similarities between item pairs co-rated by a user, or between the pair of users who rate the same items [8]. Although this type of algorithm is easy to implement and very effective in practice, its recommendations become less accurate as the data become sparser. Model-based *CF* algorithms, such as naïve Bayes (*NB*) and tree augmented naïve Bayes (*TAN*), whose parameters are optimized using extended logistic regression (*NB-ELR* and *TAN-ELR* [2]), are able to deal with incomplete data and thus address the sparsity problem of *CF*. However, existing research results show that the performance improvement over *Pearson CF* is not significant [9].

A hybrid recommender system combines *CF* and content-based techniques in an attempt to avoid the limitations of either recommender system and thereby improve recommendation performance. A representative hybrid *CF* algorithm, content-boosted *CF* recommender [5], uses *NB* to fill in the missing values of the rating matrix of the *CF* data with the predictions of the pure content-based predictor, to form a "pseudo rating matrix". In an example shown in Table 1, the *NB* classifier would map values of {Age, Sex, Occupation, Postcode} to a rating in {1,2,3,4,5} to fill in the missing entries of the rating matrix to form a pseudo rating matrix. After that, a *weighted Pearson correlation* scheme is applied to this pseudo rating matrix to produce the specific response sought, for a particular user/item pair.

One shortcoming of hybrid recommender systems is that the content information is not always available for the reasons such as privacy protection.

	I_1	I_2	I_3	I_4	I_5
U_1			**4**		
U_2	**2**		**4**	**3**	
U_3		**1**			
U_4		**3**	**3**	**3**	**3**
U_5	**1**				
U_6		**4**		**2**	

I_1	I_2	I_3	I_4	I_5
2	*3*	**4**	*3*	*2*
2	*2*	**4**	**3**	*2*
3	**1**	*3*	*4*	*3*
3	**3**	**3**	**3**	**3**
1	*2*	*4*	*1*	*2*
2	**4**	*3*	**2**	*4*

Table 1: (left) the original rating matrix; (right) the pseudo rating matrix

We design and implement two novel hybrid *CF* algorithms: (1) Sequential mixture *CF* (*SMCF*) first uses the predictions from a *TAN-ELR* content-based predictor (instead of *NB*) to fill in the missing values of the *CF* rating matrix to form a pseudo rating matrix, then predicts user ratings by using the *Pearson CF* algorithm instead of *weighted Pearson CF* on the pseudo rating matrix. (2) Joint mixture *CF* (*JMCF*) combines the predictions from three independent experts: *Pearson correlation-based CF*, a pure *TAN-ELR content-based predictor*, and a pure *TAN-ELR model-based CF* algorithm.

To evaluate our systems, we use real-world data from *MovieLens* [3] as our test-bed, and use the commonly-used *CF* criterion *mean absolute error* (*MAE*) as the performance metric, which is defined as the average of the absolute difference between the predictions and true ratings [1]:

$$MAE = \frac{\sum |p_{ij} - r_{ij}|}{N} \quad (1)$$

where N is the total number of ratings over all users, p_{ij} is the predicted rating for user i on item j, and r_{ij} is the actual rating. Good predictors are those that produce low *MAE* values.

Section 2 presents the framework of our hybrid collaborative filtering algorithms. Experimental design, results and discussions are in Section 3, Section 4, and Section 5 respectively.

2. Hybrid Collaborative Filtering Algorithms Using a Mixture of Experts

The traditional *Pearson CF* algorithm first calculates the similarity (aka "weight"), $w_{i,j}$, between two users or two items, i and j. The algorithm then computes the weighted average of all the ratings on a certain item, or uses a simple weighted average [8] to produce a prediction for the *active user*. The ability of a *CF* algorithm to produce good predictions can be measured by its *robustness*, which we define as the number of predictions made by using the algorithms (without using *default voting*) divided by the total number of predictions that are expected be made. A problem of the *Pearson CF* algorithm is its robustness decreases quickly as the data become sparser.

We propose two approaches (sequential mixture *CF* and joint mixture *CF*) to improve the performance of a hybrid system, basically by enhancing the individual components.

The sequential mixture *CF (SMCF)* is very similar to the *content-boosted CF* algorithm [5], differing mainly by not using *NB* as the content predictor. *NB* is problematic as it assumes the attribute independency, given the class variable. By contrast, a *tree augmented naive Bayes* network (*TAN*) can include some dependencies between features, which means it generally has better classification performance than *NB*. However, both *NB* and *TAN* seek the parameters that produce best *generative* performance, even though our goal is the best *discriminator*. Logistic regression (*LR*) seeks the parameters that optimize *discriminative* performance. However, it is based on a structure that is as simple as *NB*. *TAN-ELR* [2] combines both advantages: it first seeks a Bayesian net structure that optimizes *discriminative* performance, and then seeks the parameters that work effectively for discrimination. *Greiner et al.* [2] demonstrate that *TAN-ELR* produces high classification accuracy for both complete and incomplete data. We therefore use *TAN-ELR* as the pure content predictor and also as the pure model-based *CF* predictor in this work.

Our *SMCF* first uses the observed ratings as class values and the associated user information as attribute values to train a *TAN-ELR* model; it then uses the model to give predictions for the unobserved ratings from their corresponding content information. This process is repeated for every column (item) in the rating matrix. The missing ratings in the rating matrix are then replaced by the predictions from the model to create a pseudo rating matrix (without missing values; see the example in Table 1 (right)).

Other major differences between our *SMCF* and the content-boosted *CF* are: we used divide and conquer (described in Section 3) to handle large datasets instead of working on the original large rating data, and we applied *Pearson CF* directly on the pseudo rating matrix to give predictions for the *CF* task instead of using *weighted Pearson CF*, which gives a higher weight for the items rated by more users, as well as gives a higher weight for the active user [5]. Figure 1 provides a high-level outline of this algorithm.

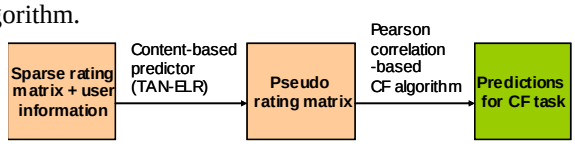

Figure 1: Sequential mixture CF (SMCF)

We also developed a second novel *CF* system by producing many "experts", each capable of predicting the rating for a user/item pair, then combining their predictions. We anticipate that a mixture of the decisions of several diverse experts, might well outperforms any component expert. Our joint mixture *CF* algorithm (*JMCF*) is an ensemble method that combines predictions from three different experts: the *Pearson CF* algorithm, a pure *content predictor using TAN-ELR*, and a pure model-based *CF* using the *TAN-ELR* algorithm (Figure 2).

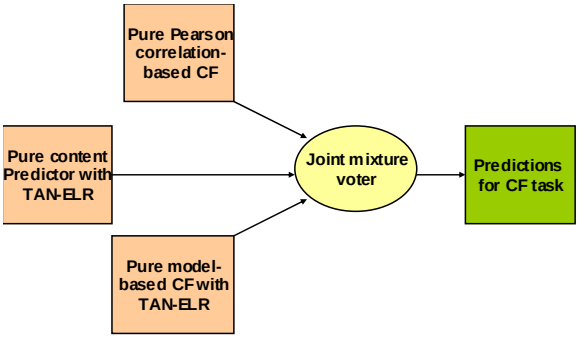

Figure 2: the joint mixture CF (JMCF) algorithm

We use *weighted average* to produce a final vote for three experts.

$$class = \frac{\sum_i W_i P_i}{\sum_i W_i} \qquad (2)$$

where the summation i is over the experts, W_i and P_i are the weight and the prediction of the expert i. The predicted class is the weighted average value rounded to the nearest integer.

Within the *JMCF* algorithm, the *TAN-ELR CF* expert uses each column of the rating matrix in turn as the class column and all remaining columns of ratings as the attributes. This is basically the same strategy that we applied to the pure *content-based predictor using TAN-ELR*, differing only by using the *user information* (content) of the respective observed ratings as the features. We then use each of the other columns (items) of the rating matrix in turn as the class column and repeat the above steps to generate the predictions. After computing the predictions from the three experts, the *JMCF* algorithm uses a joint mixture voter (Equation 2) to predict ratings.

3. Experimental Design

In addition to implementing our newly-proposed hybrid *CF* algorithms, *SMCF* and *JMCF*, we also implemented a pure *content-based predictor using TAN-ELR*, pure model-based *CF* using *TAN-ELR* (*TAN-ELR CF*), *Pearson CF* algorithm, as well as content-boosted *CF*, and a *naïve hybrid* recommender, which takes the simple average of the predictions from *Pearson CF* and a *content-based predictor using TAN-ELR*.

To investigate the impact of sparsity to the *CF* performance, and to work on datasets of sizes that collaborative filters can easily handle, we used a divide-and-conquer strategy to get subsets of the real-world *MovieLens* data [3] with different data sparsity. *MovieLens* is a web-based movies recommender system. We rank each item based on the number of users that have rated it and use this ranking to sort the items into the 10 disjoint subsets. (We found the missing rates (sparsity) ranged from ranging from 63.7% to 96.1%). Each sub-dataset contains all 943 users and a specific subset of 60 movies: the first dataset has 943 users and 60 most rated movies, the 2nd dataset has the next 60 movies, etc. Each of the datasets has observed/unobserved ratings for 943 users on 60 movies, with integer rating values from 1 to 5. The content information of the datasets contains four demographic attributes: age, sex, occupation, and postal code.

For *Pearson CF* algorithm, we use an *all-but-one* strategy to make the prediction for an observed rating using all other observed ones. When the algorithm is not able to give a prediction, we use the *universal average rating*. When the *Pearson CF* algorithm is applied to the pseudo rating matrix for the *SMCF* algorithm, it pretends that each rating was observed.

The *JMCF* algorithm makes predictions by using the weighted average as the joint mixture voter, whose weights are based on the rank of performance of the three experts: *Pearson CF* algorithm, *TAN-ELR CF*, and *pure content-based predictor using TAN-ELR*. Our empirical studies found that using a higher weight for a superior expert appeared to work the best, over a hold-out subsample. We used 4/9 as the weight for the best performing expert, and 3/9 and 2/9 as the weights for the second and third performing experts respectively. In our study, when the data sparsity is below 94%, *Pearson CF* is ranked the *1st*, and *TAN-ELR CF* and *content-predictor* ranked *2nd* and *3rd* respectively; but when the sparsity is above 94%, the rankings were (in order): *TAN-ELR CF*, *Pearson CF* and content-predictor. We used the weights for them accordingly, that is, for the first seven sub-datasets, we used the first set of weights, but for the remaining ones, we used the 2nd set.

When learning the *TAN-ELR* models, we use 5-fold cross-validation to train and make predictions for the *JMCF* algorithm, and use hold-out training and testing for the content predictor in the *SMCF* algorithm. We

use the options that appear to work the best for the *ELR* algorithm [2]: use observed frequency estimate (*OFE*) to initialize the conditional probability tables, and use 5-fold "cross-tuning" to determine the number of iterations during training, to a maximum of 20. For evaluation purposes, we only make predictions for the observed ratings that are known (to us, but not the learner) so that we can compare the predictions with the true values.

4. Results

Table 2 and Figure 3 present our results. The overall *MAE* is calculated over all ratings of the datasets, which equals the weighted average *MAE*, weighted by the percentage of the total ratings. The main comparisons are made between our algorithms and the traditional *Pearson CF* and the content-boosted *CF*.

Of the three experts used by *JMCF*, experimental results show that the traditional *Pearson CF* performs better than the pure content-predictor and the pure model-based *CF* when the sparsity of the dataset is below 94%, while *TAN-ELR CF* takes the lead when the sparsity is higher (Table 2). This suggests that *TAN-ELR CF* is more robust than the *Pearson CF* against the highly sparse data.

Of all the *CF* algorithms considered in this work, the *SMCF* algorithm has the best prediction performance, with an *MAE* score that is 4.67% better than the traditional *Pearson CF*; this difference is significant at a $p<0.002$, using a 1-sided t-test. The *SMCF* algorithm scores 3.84% better than content-boosted *CF* ($p<0.0003$), and 1.75% better than the *JMCF* algorithm ($p<0.002$). The *SMCF* gives even more accurate predictions for extremely sparse data. The resulting predictions of the *JMCF* are significantly better than the best of the three experts (better than *Pearson CF* with a 1-sided t-test $p<0.003$).

missing rate %	Pearson CF	Model-based CF	content predictor	CBCF	JMCF	SMCF
63.75	0.6901	0.7592	0.8055	0.6974	**0.6818**	0.6820
68.24	0.6976	0.7670	0.8203	0.7033	0.6885	**0.6883**
76.50	0.7108	0.7800	0.8178	0.7091	0.6981	**0.6932**
76.74	0.7325	0.8084	0.8359	0.7244	0.7221	**0.7088**
85.30	0.7723	0.8296	0.8479	0.7680	0.7433	**0.7155**
87.55	0.7895	0.8458	0.8664	0.7822	0.7538	**0.7303**
91.54	0.8166	0.8657	0.8952	0.7910	0.7797	**0.7416**
94.64	0.8937	0.8921	0.9178	0.8705	0.8135	**0.7785**
95.59	0.8858	0.8437	0.8669	0.8014	0.7836	**0.7335**
96.14	0.9803	0.9450	1.0174	0.8818	0.8786	**0.8200**
overall	0.7407	0.8000	0.8378	0.7344	0.7188	**0.7062**

Table 2: MAE scores of the CF algorithms on the 943 users and 60 items datasets

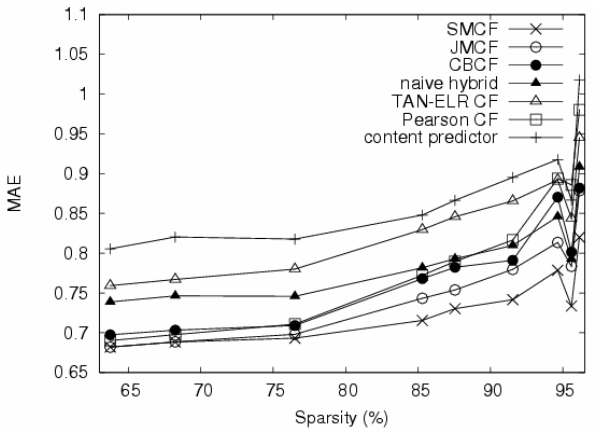

Figure 3: Prediction performance of the CF algorithms: the SMCF and JMCF have significantly better performance than others

A *naive hybrid* recommender, which takes the simple average of *Pearson CF* and a *content-based predictor with TAN-ELR*, has worse performance than *JMCF*, *SMCF* and even *Pearson CF*, as here the good performance from *Pearson CF* is degraded by being combined with a worse-performing expert (Figure 3).

5. Discussions

Theoretically, ensemble methods such as boosting and bagging may produce better classifiers than using the simple weighted average. These systems, however, are designed to deal with conditional probabilities. As we cannot easily get the conditional probabilities for each rating using *Pearson CF*, we use *weighted average* as our joint mixture strategy; our empirical results demonstrate that this works very well.

We conjecture that our *SMCF* significantly improved over the pure *Pearson CF* and content-boosted *CF* as it used *TAN-ELR* as the content-based predictor. We applied the *Pearson CF* algorithm (rather than the *weighted Pearson CF*) directly on the pseudo rating matrix, for the subsets of *MovieLens* data. Although existing literature suggests that a *weighted Pearson CF* shows good performance [4], it is suitable for the original rating data. Our divide-and-conquer strategy allows us to use our Bayesian predictors for reasonably-sized datasets instead of directly working on original large data. User-based *CF* and item-based Pearson *CF* are two types of *Pearson CF*s, with the same fundamental principle: making *CF* recommendations by aggregating similarities based on the observed ratings. We let the data determine which type of Pearson *CF* to use: when there are more users than items in the data, a user-based *CF* is preferred. As each sub-dataset has 943

users and 60 items, we used user-based *Pearson CF*. We plan to increase the number of items in the future. Also, as there are many datasets with around one thousand instances and less than 60 attributes, we plan to apply our algorithms to other classification tasks, especially for classifying very sparse incomplete data.

The experiments with the *JMCF* algorithm suggest that independent collaborative filtering experts can, when combined appropriately, produce joint predictions that are much better than the best member of the experts. However, for the mutually-dependent experts, there is no guarantee of performance improvement and empirically the improvement using this joint matrix scenario is limited.

We worked on 10 subsets, each of which has 943 users and 60 items, which collectively represent about 87% of the original 100,000 *MovieLens* ratings. We evaluated the performance of our algorithms over all these ratings. We used *MAE* and statistical analysis as evaluation metrics. This experimental design helps us to reach reliable conclusions on our algorithms. We only used the content information for users, but did not use movie information, because the user information in the *MovieLens* data can be directly used for training models, while the movie information can not as it has too few attributes and does not have a unique category for each movie.

It will be interesting to find other *CF* predictors to incorporate into the *SMCF* algorithm. For example, while our hybrid algorithms use a Bayesian network classifier, *TAN-ELR*, as both the content-based predictor and also the pure model-based *CF* predictor, other designers can instead use their own classifiers and keep the underlying framework of our algorithms.

Although the model-building expense of the *TAN-ELR* is not cheap (about 6 minutes on average to train and test each *TAN-ELR CF* model and about 1 minute on average for the content-based model, using computers with AMD Athlon XP 1.1GHz processors and 1GB memory), for online recommendation systems, the model-building process can be performed offline and the online prediction-producing process will then require a much shorter time.

6. Conclusions

It is always a good idea to make predictions based on all available information. In this paper, we proposed two hybrid *CF* algorithms, which take advantage of additional content information to make recommendations by using both collaborative filtering and a content-based predictor. The sequential mixture *CF* algorithm (*SMCF*) creates a *pseudo rating matrix* by replacing missing values with the predictions from a *TAN-ELR* based content predictor, then makes predictions using the traditional *Pearson CF* algorithm on the *pseudo rating matrix*, instead of using a more complex *weighted Pearson CF*. The joint mixture *CF* algorithm (*JMCF*) votes for predictions from three independent sources: the *Pearson CF* algorithm, a pure model-based *CF* algorithm, and a pure content-based predictor, using a *weighted average* voter. Both model-based *CF* recommender and content-based predictor of these algorithms use *TAN-ELR*, a Bayesian network algorithm with the ability to deal with incomplete data and with a structure allowing dependency between attributes. We used a divide-and-conquer strategy, which applies Bayesian predictors to reasonably-sized sub-datasets instead of directly working on original large data. Empirical results show that both *SMCF* and *JMCF* have better performances than the traditional *Pearson CF* and *content-boosted CF* in terms of *MAE*.

Acknowledgement: We thank the members of Data Mining and Machine Learning Lab at Florida Atlantic University for their help. RG is supported by NSERC and Alberta Ingenuity Centre of Machine Learning.

7. References

[1] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins. Eigentaste: A Constant Time Collaborative Filtering Algorithm, *Information Retrieval*, 4(2), pp. 133-151, 2001.

[2] R. Greiner, X. Su, B. Shen, and W. Zhou. Structural Extension to Logistic Regression: Discriminative Parameter Learning of Belief Net Classifiers, *Machine Learning*, 59(3), pp. 297 - 322, 2005.

[3] GroupLens. http://movielens.umn.edu, GroupLens Research group, Department of Computer Science and Engineering, University of Minnesota, 2006.

[4] R. Jin, J. Chai, and L. Si. An Automated Weighting Scheme for Collaborative Filtering, *the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, 2004.

[5] P. Melville, R.J. Mooney, and R. Nagarajan. Content-Boosted Collaborative Filtering for Improved Recommendations, *Proceedings of the 18th National Conference of Artificial Intelligence*, 2002.

[6] K. Miyahara, and M.J. Pazzani, Improvement of Collaborative Filtering with the Simple Bayesian Classifier, *Information Processing Society of Japan*, 43(11), 2002.

[7] Netflix prize, http://www.netflixprize.com.

[8] B.M. Sarwar, G. Karypis, J.A. Konstan, and J. Riedl. Item-based Collaborative Filtering Recommendation Algorithms, *10th International World Wide Web Conference*, pp. 285-295, 2001.

[9] X. Su, and T.M. Khoshgoftaar. Collaborative Filtering for Multi-class Data Using Belief Net Algorithms, *the 18th IEEE International Conference on Tools with Artificial Intelligence (ICTAI)*, pp. 497-504, 2006.