User Participation Prediction in Online Forums

Zhonghua Qu and **Yang Liu**
The University of Texas at Dallas
{qzh,yangl@hlt.utdallas.edu}

Abstract

Online community is an important source for latest news and information. Accurate prediction of a user's interest can help provide better user experience. In this paper, we develop a recommendation system for online forums. There are a lot of differences between online forums and formal media. For example, content generated by users in online forums contains more noise compared to formal documents. Content topics in the same forum are more focused than sources like news websites. Some of these differences present challenges to traditional word-based user profiling and recommendation systems, but some also provide opportunities for better recommendation performance. In our recommendation system, we propose to (a) use latent topics to interpolate with content-based recommendation; (b) model latent user groups to utilize information from other users. We have collected three types of forum data sets. Our experimental results demonstrate that our proposed hybrid approach works well in all three types of forums.

1 Introduction

Internet is an important source of information. It has become a habit of many people to go to the internet for latest news and updates. However, not all articles are equally interesting for different users. In order to intelligently predict interesting articles for individual users, personalized news recommendation systems have been developed. There are in general two types of approaches upon which recommendation systems are built. Content based recommendation systems use the textual information of news articles and user generated content to rank items. Collaborative filtering, on the other hand, uses co-occurrence information from a collection of users for recommendation.

During the past few years, online community has become a large part of internet. More often, latest information and knowledge appear at online community earlier than other formal media. This makes it a favorable place for people seeking timely update and latest information. Online community sites appear in many forms, for example, online forums, blogs, and social networking websites. Here we focus our study on online forums. It is very helpful to build an automatic system to suggest latest information a user would be interested in. However, unlike formal news media, user generated content in forums is usually less organized and not well formed. This presents a great challenge to many existing news article recommendation systems. In addition, what makes online forums different from other media is that users of online communities are not only the information consumers but also active providers as participants. Therefore in this study we develop a recommendation system to account for these characteristics of forums. We propose several improvements over previous work:

- Latent topic interpolation: This is to address the issue with the word-based content representation. In this paper we used Latent Dirichlet Allocation (LDA), a generative multinomial mixture model, for topic inference inside threads. We build a system based on words and latent topics, and linearly interpolate their results.

- User modeling: We model users' participation inside threads as latent user groups. Each latent group is a multinomial distribution on users. Then LDA is used to infer the group mixture inside each thread, based on which the probability of a user's participation can be derived.
- Hybrid system: Since content and user-based methods rely on different information sources, we combine the results from them for further improvement.

We have evaluated our proposed method using three data sets collected from three representative forums. Our experimental results show that in all forums, by using latent topics information, system can achieve better accuracy in predicting threads for recommendation. In addition, by modeling latent user groups in thread participation, further improvement is achieved in the hybrid system. Our analysis also showed that each forum has its nature, resulting in different optimal parameters in the different forums.

2 Related Work

Recommendation systems can help make information retrieving process more intelligent. Generally, recommendation methods are categorized into two types (Adomavicius and Tuzhilin, 2005), content-based filtering and collaborative filtering.

Systems using content-based filtering use the content information of recommendation items a user is interested in to recommend new items to the user. For example, in a news recommendation system, in order to recommend appropriate news articles to a user, it finds the most prominent features (e.g., key words, tags, category) in the document that a user likes, then suggests similar articles based on this "personal profile". In Fabs system (Balabanovic and Shoham, 1997), Skyskill & Webert system (Pazzani et al., 1997), documents are represented using a set of most important words according to a weighting measure. The most popular measure of word "importance" is TF-IDF (term frequency, inverse document frequency) (Salton and Buckley, 1988), which gives weights to words according to its "informativeness". Then, base on this "personal profile" a ranking machine is applied to give a ranked recommendation list. In Fabs system, Rocchio' algorithm (Rocchio, 1971) is used to learn the average TF-IDF vector of highly rated documents. Skyskill & Webert's system uses Naive Bayes classifiers to give the probability of documents being liked. Winnow's algorithm (Littlestone, 1988), which is similar to perception algorithm, has been shown to perform well when there are many features. An adaptive framework is introduced in (Li et al., 2010) using forum comments for news recommendation. In (Wu et al., 2010), a topic-specific topic flow model is introduced to rank the likelihood of user participating in a thread in online forums.

Collaborative-filtering based systems, unlike content-based systems, predict the recommending items using co-occurrence information between users. For example, in a news recommendation system, in order to recommend an article to user c, the system tries to find users with similar taste as c. Items favored by similar users would be recommended. Grundy (Rich, 1979) is known to be one of the first collaborative-filtering based systems. Collaborative filtering systems can be either model based or memory based (Breese et al., 1998). Memory-based algorithms, such as (Delgado and Ishii, 1999; Nakamura and Abe, 1998; Shardanand and Maes, 1995), use a utility function to measure the similarity between users. Then recommendation of an item is made according to the sum of the utility values of active users that participate in it. Model-based algorithms, on the other hand, try to formulate the probability function of one item being liked statistically using active user information. (Ungar et al., 1998) clustered similar users into groups for recommendation. Different clustering methods have been experimented, including K-means and Gibbs Sampling. Other probabilistic models have also been used to model collaborative relationships, including a Bayesian model (Chien and George, 1999), linear regression model (Sarwar et al., 2001), Gaussian mixture models (Hofmann, 2003; Hofmann, 2004). In (Blei et al., 2001) a collaborative filtering application is discussed using LDA. However in this model, re-estimation of parameters for the whole system is needed when a new item comes in. In this paper, we formulate users' participation differently using the LDA mixture model.

Some previous work has also evaluated using a hybrid model with both content and collaborative features and showed outstanding performance. For example, in (Basu et al., 1998), hybrid features are used to make recommendation using inductive learning.

3 Forum Data

We have collected data from three forums in this study.[1] Ubuntu community forum is a technical support forum; World of Warcraft (WoW) forum is about gaming; Fitness forum is about how to live a healthy life. These three forums are quite representative of online forums on the internet. Using three different types of forums for task evaluation helps to demonstrate the robustness of our proposed method. In addition, it can show how the same method could have substantial performance difference on forums of different nature. Users' behaviors in these three forums are very different. Casual forums like "Wow gaming" have much more posts in each thread. However its posts are the shortest in length. This is because discussions inside these types of forums are more like casual conversation, and there is not much requirement on the user's background, and thus there is more user participation. In contrast, technical forums like "Ubuntu" have fewer average posts in each thread, and have the longest post length. This is because a Question and Answer (QA) forum tends to be very goal oriented. If a user finds the thread is unrelated, then there will be no motivation for participation.

Inside forums, different boards are created to categorize the topics allowed for discussion. From the data we find that users tend to participate in a few selected boards of their choices. To create a data set for user interest prediction in this study, we pick the most popular boards in each forum. Even within the same board, users tend to participate in different threads base on their interest. We use a user's participation information as an indication whether a thread is interesting to a user or not. Hence, our task is to predict the user participation in forum threads. Note this approach could introduce some bias toward negative instances in terms of user interests. A users' absence from a thread does not necessarily mean the user is not interested in that thread; it may be a result of the user being offline by that time or the thread is too behind in pages. As a matter of fact, we found most users read only the threads on the first page during their time of visit of a forum. This makes participation prediction an even harder task than interest prediction.

In online forums, threads are ordered by the time stamp of their last participating post. Provided with the time stamp for each post, we can calculate the order of a thread on its board during a user's participation. Figure 1 shows the distribution of post location during users' participation. We found that most of the users read only the posts on the first page. In order to minimize the false negative instances from the data set, we did thread location filtering. That is, we want to filter out messages that actually interest the user but do not have the user's participation because they are not on the first page. For any user, only those threads appearing in the first 10 entries on a page during a user's visit are included in the data set.

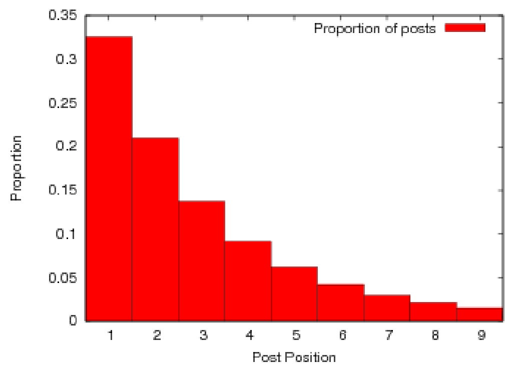

Figure 1: Thread position during users' participation.

In the pre-processing step of the experiment, first we use online status filtering discussed above to remove threads that a user does not see while offline. The statistics of the boards we have used in each forum are shown in Table 1. The statistics are consistent with the full forum statistics. For example, users in technical forums tend to post less than casual forums. We define active users as those who have participated in 10 or more threads. Column "Part. @300" shows the average number

[1]Please contact the authors to obtain the data.

of threads the top 300 users have participated in. “Filt. Threads@300” shows the average number of threads after using online filtering with a window of 10. Thread participation in “Ubuntu” forum is very sparse for each user, having only 10.01% participating threads for each user after filtering. “Fitness” and “Wow Forum” have denser participation, at 18.97% and 13.86% respectively.

4 Interesting Thread Prediction

In the task of interesting thread prediction, the system generates a ranked list of threads a user is likely to be interested in based on users’ past history of thread participation. Here, instead of predicting the true interestedness, we predict the participation of the user, which is a sufficient condition for interestedness. This approach is also used by (Wu et al., 2010) for their task evaluation. In this section, we describe our proposed approaches for thread participation prediction.

4.1 Content-based Filtering

In the content-based filtering approach, only content of a thread is used as features for prediction. Recommendation through content-based filtering has its deep root in information retrieval. Here we use a Naive Bayes classifier for ranking the threads using information based on the words and the latent topic analysis.

4.1.1 Naive Bayes Classification

In (Pazzani et al., 1997) Naive Bayesian classifier showed outstanding performance in web page recommendation compared to several other classifiers. A Naive Bayes classifier is a generative model in which words inside a document are assumed to be conditionally independent. That is, given the class of a document, words are generated independently. The posterior probability of a test instance in Naive Bayes classifier takes the following form:

$$P(C_i|f_{1..k}) = \frac{1}{Z}P(C_i)\prod_j P(f_j|C_i) \quad (1)$$

where Z is the class label independent normalization term, $f_{1..k}$ is the bag-of-word feature vector for the document. Naive Bayes classifier is known for not having a well calibrated posterior probability (Bennett, 2000). (Pavlov et al., 2004) showed that normalization by document length yielded good empirical results in approximating a well calibrated posterior probability for Naive Bayes classifier. The normalized Naive Bayes classifier they used is as follows:

$$P(C_i|f_{1..k}) = \frac{1}{Z}P(C_i)\prod_j P(f_j|C_i)^{\frac{1}{|f|}} \quad (2)$$

In this equation, the probability of generating each word is normalized by the length of the feature vector $|f|$. The posterior probability $P(interested|f_{1..k})$ from (normalized) Naive Bayes classifier is used for recommendation item ranking.

4.1.2 Latent Topics based Interpolation

Because of noisy forum writing and limited training data, the above bag-of-word model used in naive Bayes classifier may suffer from data sparsity issues. We thus propose to use latent topic modeling to alleviate this problem. Latent Dirichlet Allocation (LDA) is a generative model based on latent topics. The major difference between LDA and previous methods such as probabilistic Latent Semantic Analysis (pLSA) is that LDA can efficiently infer topic composition of new documents, regardless of the training data size (Blei et al., 2001). This makes it ideal for efficiently reducing the dimension of incoming documents.

In an online forum, words contained in threads tend to be very noisy. Irregular words, such as abbreviation, misspelling and synonyms, are very common in an online environment. From our experiments, we observe that LDA seems to be quite robust to these phenomena and able to capture word relationship semantically. To illustrate the words inside latent topics in the LDA model inferred from online forums, we show in Table 2 the top words in 3 out of 20 latent topics inferred from “Ubuntu” forum according to its multinomial distribution. We can see that variations of the same words are grouped into the same topic.

Since each post could be very short and LDA is generally known not to work well with short documents, we concatenated the content of posts inside each thread to form documents. In order to build a valid evaluation configuration, only posts before the first time the testing user participated are used for model fitting and inference.

Forum Name	Threads	Posts	Active Users	Part. @300	Filt. Threads @300
Ubuntu	185,747	940,230	1,700	464.72	4641.25
Fitness	27,250	529,201	2,808	613.15	3231.04
Wow Gaming	34,187	1,639,720	19,173	313.77	2264.46

Table 1: Data statistics after filtering.

Topic 1	Topic 2	Topic 3
lol'd	wine	email
lol.	Wine	mail
imo.	game	Thunderbird
,'	fixme	evolution
-,-	stub	send
lulz.	not	emails
lmao.	WINE	gmail
rofl.	play	postfix

Table 2: Example of LDA topics that capture words with different variations.

After model fitting for LDA, the topic distributions on new threads can be inferred using the model. Compared to the original bag-of-word feature vector, the topic distribution vector is not only more robust against noise, but also closer to human interpretation of words. For example in topic 3 in Table 2, people who care about "Thunderbird", an email client, are also very likely to show interest in "postfix", which is a Linux email service. These closely related words, however, might not be captured using the bag-of-word model since that would require the exact words to appear in the training set.

In order to take advantage of the topic level information while not losing the "fine-grained" word level feature, we use the topic distribution as additional features in combination with the bag-of-word features. To tune the contribution of topic level features in classifiers like Naive Bayes classifiers, we normalize the topic level feature to a length of $L_t = \gamma|f|$ and bag-of-word feature to $L_w = (1-\gamma)|f|$. γ is a tuning parameter from 0 to 1 that determines the proportion of the topic information used in the features. $|f|$ is from the original bag-of-word feature vector. The final feature vector for each thread can be represented as:

$$F = L_w w_1, ..., L_w w_k \cup L_t\theta_1, ..., L_t\theta_T \qquad (3)$$

where $\theta_1, ..., \theta_t$ is the multinomial distribution of topics for the thread.

4.2 Collaborative Filtering

Collaborative filtering techniques make prediction using information from similar users. It has advantages over content-based filtering in that it can correctly predict items that are vastly different in content but similar in concepts indicated by users' participation.

In some previous work, clustering methods were used to partition users into several groups, Then, predictions were made using information from users in the same group. However, in the case of thread recommendation, we found that users' interest does not form clean clusters. Figure 2 shows the mutual information between users after doing an average-link clustering on their pairwise mutual information. In a clean clustering, intra-cluster mutual information should be high, while inter-cluster mutual information is very low. If so, we would expect that the figure shows clear rectangles along the diagonal. Unfortunately, from this figure it appears that users far away in the hierarchy tree still have a lot of common thread participation. Here, we propose to model user similarity based on latent user groups.

4.2.1 Latent User Groups

In this paper, we model users' participation inside threads as an LDA generative model. We model each user group as a multinomial distribution. Users inside each group are assumed to have common interests in certain topic(s). A thread in an online forum typically contains several such topics. We could model a user's participation in a thread as a mixture of several different user groups. Since one thread typically attracts a subset of user groups, it is reasonable to add a Dirichlet prior on the user group mixture.

The generative process is the same as the LDA used above for topic modeling, except now users

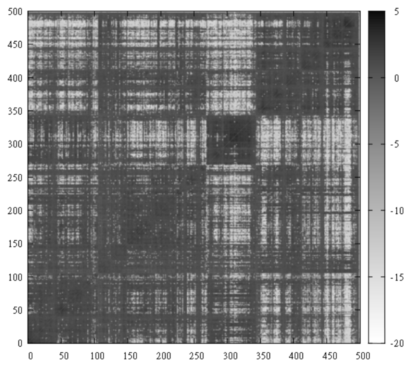

Figure 2: Mutual information between users in Average Link Hierarchical clustering.

are 'words' and user groups are 'topics'. Using LDA to model user participation can be viewed as soft-clustering of users in a sense that one user could appear in multiple groups at the same time. The generative process for participating users is as follows.

1. Choose $\theta \sim Dir(\alpha)$
2. For each of N participating users, u_n:
   (a) Choose a group $z_n \sim Multinomial(\theta)$
   (b) Choose a user $u_n \sim p(u_n|z_n)$

One thing worth noting is that in LDA model a document is assumed to consist of many words. In the case of modeling user participation, a thread typically has far fewer users than words inside a document. This could potentially cause problem during variable estimation and inference. However, we show that this approach actually works well in practice (experimental results in Section 5).

4.2.2 Using Latent User Groups for Prediction

For an incoming new thread, first the latent group distribution is inferred using collapsed Gibbs Sampling (Griffiths and Steyvers, 2004). The posterior probability of a user u_i participating in thread j given the user group distribution is as follows.

$$P(u_i|\theta_j, \phi) = \sum_{k \in T} P(u_i|\phi_k) P(k|\theta_j) \qquad (4)$$

In the equation, ϕ_k is the multinomial distribution of users in group k, T is the number of latent user groups, and θ_j is the group composition in thread j after inference using the training data. In general, the probability of user u_i appearing in thread j is proportional to the membership probabilities of this user in the groups that compose the participating users.

4.3 Hybrid System

Up to this point we have two separate systems that can generate ranked recommendation lists based on different factors of threads. In order to generate the final ranked list, we give each item a score according to the ranked lists from the two systems. Then the two scores are linearly interpolated using a tuning parameter λ as shown in Equation 5. The final ranked list is generated accordingly.

$$\begin{aligned} C_i =& (1-\lambda) Score_{content} \\ &+ \lambda Score_{collaborative} \end{aligned} \qquad (5)$$

We propose several different rescoring methods to generate the scores in the above formula for the two individual systems.

- **Posterior:** The posterior probabilities of each item from the two systems are used directly as the score.

  $$Score_{dir} = p(c_{like}|item_i) \qquad (6)$$

  This way the confidence of "how likely" an item is interesting is preserved. However, the downside is that the two different systems have different calibration on its posterior probability, which could be problematic when directly adding them together.

- **Linear rescore:** To counter the problem associated with posterior probability calibration, we use linear rescoring based on the ranked list:

  $$Score_{lin} = 1 - \frac{pos_i}{N} \qquad (7)$$

  In the formula, pos_i is the position of item i in the ranked list, and N is the total number of items being ranked. The resulting score is between 0 and 1, 1 being the first item on the list and 0 being the last.

- **Sigmoid rescore:** In a ranked list, usually items on the top and bottom of the list have higher confidence than those in the middle. That is to say more "emphasis" should be put on both ends of the list. Hence we use a sigmoid function on the $Score_{linear}$ to capture this.

$$Score_{sig} = \frac{1}{1 + e^{-l(Score_{lin} - 0.5)}} \quad (8)$$

A sigmoid function is relatively flat on both ends while being steep in the middle. In the equation, l is a tuning parameter that decides how "flat" the score of both ends of the list is going to be. Determining the best value for l is not a trivial problem. Here we empirically assign $l = 10$.

5 Experiment and Evaluation

In this section, we evaluate our approach empirically on the three forum data sets described in Section 3. We pick the top 300 most active users from each forum for the evaluation. Among the 300 users, 100 of them are randomly selected as the development set for parameter tuning, while the rest is test set. All the data sets are filtered using an online filter as previously described, with a window size of 10 threads.

Threads are tokenized into words and filtered using a simple English stop word list. All words are then ordered by their occurrences multiplied by their inverse document frequencies (IDF).

$$idf_w = log \frac{|D|}{|\{d : w \in d\}|} \quad (9)$$

The top 4,000 words from this list are then used to form the vocabulary.

We used standard mean average precision (MAP) as the evaluation metric. This standard information retrieval evaluation metric measures the quality of the returned rank lists from a system. Entries higher in the rank are more accurate than lower ones. For an interesting thread recommendation system, it is preferable to provide a short and high-quality list of recommendation; therefore, instead of reporting full-range MAP, we report MAP on top 10 relevant threads (MAP@10). The reason why we picked 10 as the number of relevant document for MAP evaluation is that users might not have time to read too many posts, even if they are relevant.

During evaluation, a 3-fold cross-validation is performed for each user in the test set. In each fold, MAP@10 score is calculated from the ranked list generated by the system. Then the average from all the folds and all the users is computed as the final result.

To make a proper evaluation configuration, for each user, only posts up to the first participation of the testing user are used for the test set.

5.1 Content-based Results

Here we evaluate the performance of interest thread prediction using only features from text. First we use the ranking model with latent topic information only on the development set to determine an optimal number of topics. Empirically, we use hyper parameter $\beta = 0.1$ and $\alpha = 1/K$ (K is the number of topics). We use the performance of content-based recommendation directly to determine the optimal topic number K. We varied the latent topic number K from 10 to 100, and found that the best performance was achieved using 30 topics in all three forums. Hence we use $K = 30$ for content based recommendation unless otherwise specified.

Next, we show how topic information can help content-based recommendation achieve better results. We tune the parameter γ described in Section 4.1.2 and show corresponding performances. We compare the performance using Naive Bayes classifier, before and after normalization. The MAP@10 results on the test set are shown in Figure 3 for three forums. When $\gamma = 0$, no latent topic information is used, and when $\gamma = 1$, latent topics are used without any word features.

When using Naive Bayes classifier without normalization, we find relatively larger performance gain from adding topic information for the γ values of close to 0. This phenomenon is probably because of the poor posterior probabilities of the Naive Bayes classifier, which are close to either 1 or 0.

For normalized Naive Bayes classifier, interpolating with latent topics based ranking yields performance improvement compared to word-based results consistently for the three forums. In "Wow Gaming" corpus, the optimal performance is achieved with a relatively high γ value (at around 0.5), and it is even higher for the "Fitness" forum.

This means that the system relies more on the latent topics information. This is because in these forums, casual conversation contains more irregular words, causing more severe data sparsity problem than others.

Between the two naive Bayes classifiers, we can see that using normalized probabilities outperforms the original one in “Wow Gaming” and “Ubuntu” forums. This observation is consistent with previous work (e.g., (Pavlov et al., 2004)). However, we found that in “Fitness Forum”, the performance degrades with normalization. Further work is still needed to understand why this is the case.

5.2 Latent User Group Classification

In this section, collaborative filtering using latent user groups is evaluated. First, participating users from the training set are used to estimate an LDA model. Then, users participating in a thread are used to infer the topic distribution of the thread. Candidate threads are then sorted by the probability of a target user’s participation according to Equation 4. Note that all the users in the forum are used to estimate the latent user groups, but only the top 300 active users are used in evaluation. Here, we vary the number of latent user groups G from 5 to 100. Hyper parameters were set empirically: $\alpha = 1/G, \beta = 0.1$.

Figure 4 shows the MAP@10 results using different numbers of latent groups for the three forums. We compare the performance using latent groups with a baseline using SVM ranking. In the baseline system, users’ participation in a thread is used as a binary feature. LibSVM with radius based function (RBF) kernel is used to estimate the probability of a user’s participation.

From the results, we find that ranking using latent groups information outperforms the baseline in almost all non-trivial cases. In the case of “Ubuntu” forum, the performance gain is less compared to other forums. We believe this is because in this technical support forum, the average user participation in threads is much less, thus making it hard to infer a reliable group distribution in a thread. In addition, the optimal number of user groups differs greatly between “Fitness” forum and “Wow Gaming” forum. We conjecture the reason behind this is that in the “Fitness” forum, users

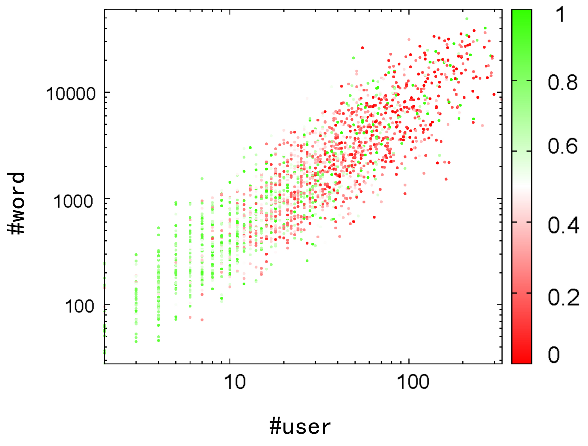

Figure 5: Position of items with different #users and #words in a ranked list. (red=0 being higher on the ranked list and green being lower)

may be interested in a larger variety of topics and thus the user distribution in different topics is not very obvious. In contrast, people in the gaming forum are more specific to the topics they are interested in.

It is known that LDA tends to perform poorly when there are too few words/users. To have a general idea of how much user participation is “enough” for decent prediction, we show a graph (Figure 5) depicting the relationships among the number of users, the number of words, and the position of the positive instances in the ranked lists. In this graph, every dot is a positive thread instance in “Wow Gaming” forum. Red color shows that the positive thread is indeed getting higher ranks than others. We observe that threads with around 16 participants can already achieve a decent performance.

5.3 Hybrid System Performance

In this section, we evaluate the performance of the hybrid system output. Parameters used in each forum data set are the optimal parameters found in the previous sections. Here we show the effect of the tuning parameter λ (described in Section 4.3). Also, we compare three different scoring schemes used to generate the final ranked list. Performance of the hybrid system is shown in Table 3.

We can see that the combination of the two systems always outperforms any one model alone.

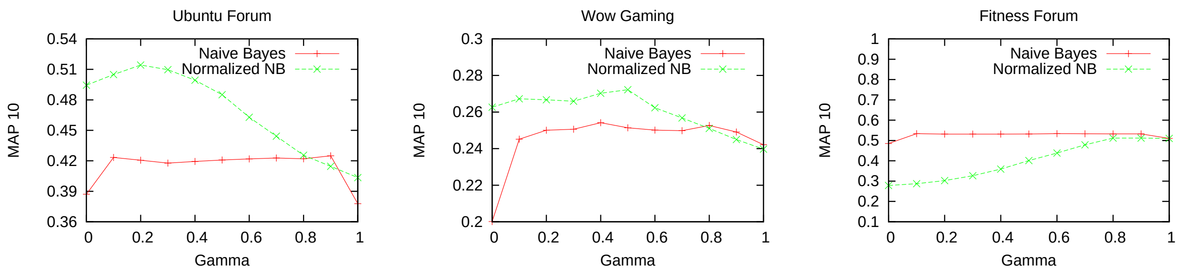

Figure 3: Content-based filtering results: MAP@10 vs. γ (contribution of topic-based features).

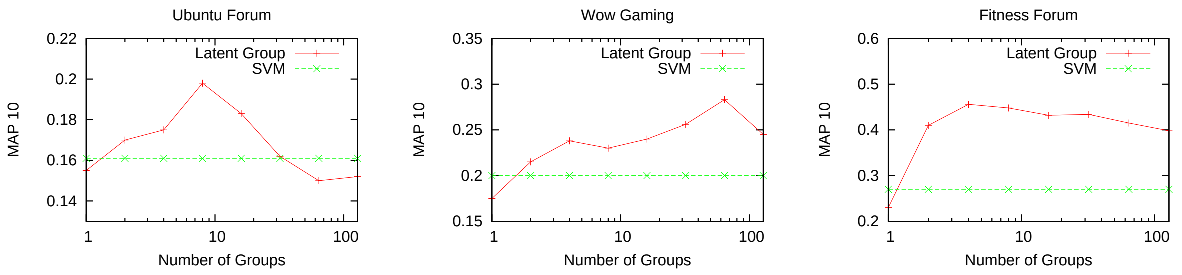

Figure 4: Collaborative filtering results: MAP@10 vs. user group number.

Forum	**Contribution Factor λ**		
	0.0	1.0	Optimal
Ubuntu	0.523	0.198	**0.534** ($\lambda = 0.9$)
Wow	0.278	0.283	**0.304** ($\lambda = 0.1$)
Fitness	0.545	0.457	**0.551** ($\lambda = 0.85$)

Table 3: Performance of the hybrid system with different λ values.

This is intuitive since the two models use different information sources. A MAP@10 score of 0.5 means that around half of the suggested results do have user participation. We think this is a good result considering that this is not a trivial task.

We also notice that based on the nature of different forums, the optimal λ value could be substantially different. For example, in "Wow gaming" forum where people participate in more threads, a higher λ value is observed which favors collaborative filtering score. In contrast, in "Ubuntu" forum, where people participate in far fewer threads, the content-based system is more reliable in thread prediction, hence a lower λ is used. This observation also shows that the hybrid system is more robust against differences among forums compared with single model systems.

6 Conclusion

In this paper, we proposed a new system that can intelligently recommend threads from online community according to a user's interest. The system uses both content-based filtering and collaborative-filtering techniques. In content-based filtering, we solve the problem of data sparsity in online content by smoothing using latent topic information. In collaborative filtering, we model users' participation in threads with latent groups under an LDA framework. The two systems compliment each other and their combination achieves better performance than individual ones. Our experiments across different forums demonstrate the robustness of our methods and the difference among forums. In the future work, we plan to explore how social information could help further refine a user's interest.

References

Gediminas Adomavicius and Alexander Tuzhilin. 2005. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING*, 17(6):734–749.

Marko Balabanovic and Yoav Shoham. 1997.

Fab: Content-based, collaborative recommendation. *Communications of the ACM*, 40:66–72.

Chumki Basu, Haym Hirsh, and William Cohen. 1998. Recommendation as classification: Using social and content-based information in recommendation. In *In Proceedings of the Fifteenth National Conference on Artificial Intelligence*, pages 714–720. AAAI Press.

Paul N. Bennett. 2000. Assessing the calibration of naive bayes' posterior estimates.

David Blei, Andrew Y. Ng, and Michael I. Jordan. 2001. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:2003.

John S. Breese, David Heckerman, and Carl Kadie. 1998. Empirical analysis of predictive algorithms for collaborative filtering. pages 43–52. Morgan Kaufmann.

Y H Chien and E I George, 1999. *A bayesian model for collaborative filtering*. Number 1.

Joaquin Delgado and Naohiro Ishii. 1999. Memory-based weighted-majority prediction for recommender systems.

Thomas L. Griffiths and Mark Steyvers. 2004. Finding scientific topics. *Proceedings of the National Academy of Sciences of the United States of America*, 101(Suppl 1):5228–5235, April.

Thomas Hofmann. 2003. Collaborative filtering via gaussian probabilistic latent semantic analysis. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, SIGIR '03, pages 259–266, New York, NY, USA. ACM.

Thomas Hofmann. 2004. Latent semantic models for collaborative filtering. *ACM Trans. Inf. Syst.*, 22(1):89–115.

Qing Li, Jia Wang, Yuanzhu Peter Chen, and Zhangxi Lin. 2010. User comments for news recommendation in forum-based social media. *Inf. Sci.*, 180:4929–4939, December.

Nick Littlestone. 1988. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. In *Machine Learning*, pages 285–318.

Atsuyoshi Nakamura and Naoki Abe. 1998. Collaborative filtering using weighted majority prediction algorithms. In *Proceedings of the Fifteenth International Conference on Machine Learning*, ICML '98, pages 395–403, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Dmitry Pavlov, Ramnath Balasubramanyan, Byron Dom, Shyam Kapur, and Jignashu Parikh. 2004. Document preprocessing for naive bayes classification and clustering with mixture of multinomials. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '04, pages 829–834, New York, NY, USA. ACM.

Michael Pazzani, Daniel Billsus, S. Michalski, and Janusz Wnek. 1997. Learning and revising user profiles: The identification of interesting web sites. In *Machine Learning*, pages 313–331.

Elaine Rich. 1979. User modeling via stereotypes. *Cognitive Science*, 3(4):329–354.

J. Rocchio, 1971. *Relevance Feedback in Information Retrieval*.

Gerard Salton and Christopher Buckley. 1988. Term-weighting approaches in automatic text retrieval. In *INFORMATION PROCESSING AND MANAGEMENT*, pages 513–523.

Badrul Sarwar, George Karypis, Joseph Konstan, and John Reidl. 2001. Item-based collaborative filtering recommendation algorithms. In *WWW '01: Proceedings of the 10th international conference on World Wide Web*, pages 285–295, New York, NY, USA. ACM.

Upendra Shardanand and Pattie Maes. 1995. Social information filtering: Algorithms for automating "word of mouth". In *CHI*, pages 210–217.

Lyle Ungar, Dean Foster, Ellen Andre, Star Wars, Fred Star Wars, Dean Star Wars, and Jason Hiver Whispers. 1998. Clustering methods for collaborative filtering. AAAI Press.

Hao Wu, Jiajun Bu, Chun Chen, Can Wang, Guang Qiu, Lijun Zhang, and Jianfeng Shen. 2010. Modeling dynamic multi-topic discussions in online forums. In *AAAI*.