KEYWORD BASED USER PROFILING FOR NEWS RECOMMENDATION

[1]AADHITHYA SANKAR, [2]ELA NILAVAZHAGAN

[1,2]Department of CSE, Sri Sairam Engineering College, Chennai, Tamilnadu, India
E-mail: [1]aadhithya.s@outlook.com, [2]nilavazhagan1696@gmail.com

Abstract— News Recommendation is increasingly being deployed by online news publishers. Recommender Systems are used as a way to deal with information overload and to increase page views. Collaborative filtering and content based filtering are two common approaches used in recommendation systems. In this paper, we propose a News Recommendation System Model that uses keywords obtained from the articles to profile the user. We then use a collaborative filtering model to generate recommendations to the users. Coping with ever changing user interests is also a major challenge for recommendation system. This model also proposes a decay function to deal with such a challenge.

Index Terms— Recommender Systems, Time Based, Keywords, Collaborative Filtering, Matrix Factorization.

I. INTRODUCTION

In today's information world, the user has to constantly face the issue of information overload, where the user is bombarded with hundreds and thousands of information from which one has to choose from[1]. This issue is particularly more visible when it comes to news articles; with each news publisher publishing over a thousand articles per day, it becomes all the more difficult for the user to find the article that is needed. Recommender Systems is one tool that helps the user to make the decision making process easier. This is done by automatically analyzing the articles, filtering and then ranking the articles that are more relevant to the user. To know what is relevant to the user and what is not, the Recommender System needs to accurately know what the user's interests are. The user profile is used to keep tabs on the user's preferences. User's preferences may be explicitly learned or implicitly learned.

Explicit learning involves directly inquiring the users about their interests. This is done when a new user joins the system. Explicit learning is also based on the ratings given to the articles the users read. But, explicit learning is not reliable. Users may not rate the article often enough to rely on it. This takes us to the use of implicit learning. Implicit learning is a kind of automatic learning by the machine. The machine learns the interests of the user automatically. This is done by reviewing the type of articles the user chose to read and chose not to. Again, implicit learning is not accurate by itself.

Therefore, this paper uses a combination of implicit and explicit learning to achieve better results. The user is made to rate the article that is read and also a score for the article for each user is calculated based on the user's profile. The final article score is a combination of the user's rating for that article and a computer generated score for that article for the user.

This model uses a two stage approach: Grouping & Recommendation. In the *Grouping stage*, the user is grouped with similar users. The *Grouping* is done by taking into consideration, the user's profile, location and demography of the user. In the *Recommendation* stage, some articles that the user has not read but might be interested are recommended to the user. This is done by assigning scores for articles not read by the user, but by other users in the group. These articles are then ranked according to the scores and presented to the user. Moreover user's interests vary over time. So, we introduce a decay function to adapt to the changes in the user's interests.

The major contributions of this model are:

- This model, finds similar users and groups them based on the content of the articles read by the user, rather than the article itself.
- It also considers the location of the users when computing the similarity between them.
- It employs a decay function to cope up with the varying interests of the user and to adapt with the new interests.

The rest of the paper is structured as follows: in the next section we briefly talk about the recent development in news recommendation. In Section 3, the structure and management of the user profile are discussed. Sections 4 to 7 describes the algorithm employed. The final section of the paper summarizes the contributions of the paper, its limitations and future scope.

II. RELATED WORKS

Moreno et al. [3] have proposed content based filtering recommender system that constructs a user profile based on the keywords from the articles that the user has already read. Their works propose a cyclic process to improve automatically and gradually the accuracy of the recommendations given to users by a news service. The first step rates and ranks the available news using the knowledge about the user's preferences stored in the user profile. Both news and

preferences are represented by textual keywords. In the second step, the system analyses the news read by the user and infers in an automatic and unsupervised fashion the changes to be applied in the user profile to represent better the user's interests. Li et al. [2] define a two level user profile, one with long-term interests and another with short-term interests. In this paper, we initially provide an experimental study on the evolution of user interests in real-world news recommender systems. To better capture the interest evolution issue, our proposed recommender seamlessly integrates the long-term and short-term reading preferences of users when recommending news items.

Han et al. [7] use a hybrid recommendation algorithm which combined with collaborative filtering algorithm and improved association rules to achieve the personalized news recommendation. Lu et al. [8] have proposed a Content-based Collaborative Filtering (CCF) approach for the news topic recommendation in Bing. By utilizing the rich contexts and focusing on the long-tail users, the proposed CCF combines both the advantages of Content-based Filtering approach and the features of Collaborative Filtering approach.

III. USER PROFILING

A profile is generated for each user. This profile keeps getting updated as the user reads more articles. The profile is used for grouping the users in the *grouping stage.* The profile of each user consists of the following data: *article history, concept-score list, entity-score list, keyword list, author list, demography vector.*

- **Article History**
  The article history is a list of all the articles the user has read. The list gets updated with each article the user reads. The article the user reads and related information are added to the list. It consists of the *article id, user's rating, machine score and personalized score* fields.
- **Concept-Score List**
  The Concept-Score List consists of concepts encountered in the articles the user has read and corresponding scores for each concept. The Concepts extraction from the articles is done using AlchemyAPI. Concept extraction identifies concepts that aren't necessarily directly referenced in the article. For example, an article with BMW, Audi and Ford in it may return *Automotive Industry* as a concept. The API returns the concepts and its relevance scores when a text block is passed as input. The range of the relevance score is between *0* and *1.* The score for each concept is calculated as follows:

$$C_{S_i} = count_i * relevence_{avg} \quad (1)$$

Where,
C_{Si} is the score for the i[th] concept,
$Count_i$ is the number of articles in which *concept i* was encountered and
$relevence_{avg}$ is the average relevance of the *concept i.*

- **Entity-Score List**
  The Entity-Score List consists of entities encountered in the articles the user has read and corresponding scores for each entity. The entity extraction from the articles is done using the AlchemyAPI. Entity extraction is task of information extraction that seeks to locate and classify named entities in text into pre-defined categories such as the names of persons, organizations, locations, expressions of times, quantities, monetary values, percentages, etc. The API returns the entities and its relevance scores when a text block is passed as input. The range of the relevance score is between *0* and *1.* The score for each entity is calculated as follows:

$$E_{S_i} = count_i * relevence_{avg} \quad (2)$$

Where,
E_{Si} is the score for the i[th] entity,
$Count_i$ is the number of articles in which *entity i* was encountered and
$relevence_{avg}$ is the average relevance of the *entity i.*

- **Keyword List**
  Keyword List is the union of the Concept-Score List and Entity-Score List. This list is used while grouping similar users.
- **Author List**
  The Author List is a list of authors whose articles the user has read. The score for each author is incremented by 1 each time the user reads an article written by the author. The score for each author is calculated as

$$A_S = count_i \quad (3)$$

where,
$count_i$ is the number of times the author's work has been read by the user.

- **Demography Vector**
  The Demography Vector is a 41 feature vector representing the demography of the user. The demography vector is used to find the demographic similarity between the users. This similarity is then combined with the user similarity to obtain the effective similarity between the users. The effective similarity is used as the deciding criteria to group the users. The structure of the demographic vector is depicted in Table 1.

Table 1. Structure of DemogDemography Vector

F#	Constraint	Value
0	Age<=18	Set bit to 1 if it applies, else set it to 0.
1	18<Age<=23	
2	23<Age<=28	
3	28<Age<=38	
4	Age>38	
5	Gender	0 if Male, 1 if Female
6-40	Indian States & UTs	States and UTs arranged in alphabetical order. Set 1 for user's state, others, set 0.

A

IV. ARTICLE SCORE GENERATION

Each article the user reads is assigned a score. This score is used to calculate the scores of articles that the user hasn't read yet. The effective score for each article the user has read is a combination of two scores: *user's rating* U_{Si} (5point scale) for that article and the *Machine generated score* M_{Si} for that article for the user.

The Machine Generated Score is calculated on a 10 point scale and takes as parameters, the entity, and concept and author scores. It is to be noted that the relative scores for the entities, concepts and the authors are calculated on a 10 point scale. That is, the entity with the highest score among the other entities is assigned a score of 10. The other entities are assigned a score relative to the score of this entity. Similarly, relative scores are assigned to authors and concepts. The Machine Generated Score can then be calculated as

$$M_{S_i} = \frac{1}{9}\left(\frac{\sum E_S}{N_E} + \frac{\sum C_S}{N_S} + A_S\right)T_S \quad \textbf{(4)}$$

Where,

M_{Si} is the Machine Generated Score for *article i* the user has read,

$\sum E_S$is the sum of relative scores of *entities* common between the *article i* and the *user's entity list*,

$\sum C_S$is the sum of relative scores of *concepts* common between the *article i* and the *user's concept list*,

A_S is the relative score of the *author* of the *article i*, and

T_S is the time spent (in minutes) by the user reading the article.

Adults on an average can read about 275 words per minute. And a news article on average has about 700 words. Using these data we decided to cap the read time T_S at 3 minutes.

The effective score $P_{Si,j}$ for the article j for user i is calculated as:

$$P_{S_{i,j}} = 0.6U_{S_i} + 0.7M_{S_i} \quad \textbf{(5)}$$

More weightage is given to the Machine Generated Score because many times the users are reluctant to give feedback to the system. The effective score $P_{Si,j}$ is on a 10 point scale.

I. V. DECAY FUNCTION

We introduce a decay function that acts on theentity and concept scores. The decay function is introduced to account for the changing interests of the user. With the decay function, the scores of the old, unvisited interests of the users get reduced. The decay function is shown below

$$C/E_{S_T} = C/E_{S_{T0}}e^{-(0.004N_i)} \quad \textbf{(6)}$$

Where,

$C/E_{ST}$is the score of the *entity/concept*,

$C/E_{ST0}$is the score when that *entity/concept* was last active,

$N_i$is the number of *irrelevant* news articles with respect to that entity/concept.

The score of the entity/concept becomes *0.996* times its original score every time the user reads an article irrelevant to the entity/concept under consideration.

VI. GROUPING THE USERS

Grouping the users is the first stage of our proposed recommendation system. We compare the users' profiles to find their similarity. We use Pearson's Correlation [4] to find the similarity between the users.

$$S(u,v) = \frac{\sum_{i\in K_u\cap K_v}(K_{s_{u,i}}-\overline{K_{s_u}})(K_{s_{v,i}}-\overline{K_{s_v}})}{\sqrt{\sum_{i\in K_u\cap K_v}(K_{s_{u,i}}-\overline{K_{s_u}})}\sqrt{\sum_{i\in K_u\cap K_v}(K_{s_{v,i}}-\overline{K_{s_v}})}} \quad \textbf{(7)}$$

Where,

$S(u,v)$is the *similarity* between users *u* and *v*,

K_u and K_v are the *keyword lists* of users *u* and *v* respectively,

$K_{Su,i}$ and $K_{Sv,i}$ are the *Keyword scores* of keyword i in K_u and K_v respectively and,

$\overline{K_{S_u}}$ and $\overline{K_{S_v}}$ are *mean scores* of K_u and K_v respectively.

The result of the above equation is a value between *-1* and *+1* where, *-1* means users u and v are *strongly dissimilar* and a score of *+1* means the users are *strongly similar.*

We also calculate the demographic similarity between the users *u* and *v* and combine it with $S(u,v)$to obtain the effective similarity. The

demographic similarity $S_D(u, v)$ is obtained by applying cosine similarity between the demogVectors of the users *u* and *v*[5].

$$S_D(u, v) = \frac{(demogVector_u \cdot demogVector_v)}{\sqrt{demogVector_{u_i}{}^2}\sqrt{demogVector_{v_i}{}^2}} \quad (8)$$

The range of $S_D(u, v)$is between 0 and 1 where 0 indicates u and v are strongly dissimilar and 1 indicates they are strongly similar.
The effective similarity between users u and v is calculated as follows [5]

$$S_e(u, v) = S(u, v) + |S(u, v) * S_D(u, v)| \quad (9)$$

The value of Effective similarity Index varies between *-1* and *+2*. A *-1* indicates the users u and v are *strongly dissimilar* and a *+2* indicates they are *strongly similar.*
Users with effective similarity greater than 0.5 are added to the group.

VII. GENERATING RECOMMENDATIONS

This is the second stage of our proposed model. Here we form an *M-Users x N-Article Scores* matrix for each User Group. This matrix is called the *Rmatrix*. Here,

$$R[i, j] = P_{S_{u_i j}} \quad (10)$$

Where $P_{S_{u_i j}}$is the article score for article j for user i.
If the user i hasn't read article j, then,

$$R[i, j] = 0 \quad (11)$$

$$R = \begin{pmatrix} P_{S_{0,0}} & \cdots & P_{S_{0,m}} \\ \vdots & \ddots & \vdots \\ P_{S_{n,0}} & \cdots & P_{S_{n,m}} \end{pmatrix}$$

Figure 1. NxM R Matrix

The *R matrix* is then factorized [6] to find two matrices *P*and *Q* such that,

$$R' = PQ^T \quad (12)$$

$$R \approx R' \quad (13)$$

The resultant matrix *R'* is almost equal to *R*, and all the elements in *R'* have a value greater than 0. Matrix Factorization is performed to find the article scores of the articles the users in the group haven't read. From *R'* we obtain the Predicted scores for the articles the users of the group haven't read. The Articles the users haven't read having a *predicted score greater than 6.5* are filtered and recommended to the users.

CONCLUSION

In this paper, we proposed a news recommendation model that profiled users based on the keywords in the articles they read. We grouped the users by the content they read rather than the articles they read and also took into account the user's demography. We also introduced a decay function that accounts for the users' changing interest over time.
Our proposal still has its disadvantages. Firstly, it requires a large number of users to give meaningful recommendation. Scalability is another issue. When the number of users increases, the matrix factorization operations can be tedious. Finally, the recommendations cannot be suggested to new users. The user's behavior must be learned in order to present meaningful recommendations.
We are working on implementing the recommendation system. Using the results of our implementation, we will be able to further refine our algorithm. We are also working to continually improve the recommendation algorithm and to surmount the disadvantages.

REFERENCES

[1] Angela Edmunds, Anne Morris, "The problem of information overload in business organisations: a review of the literature", International journal of information management, 2000.

[2] L. Li, L. Zheng, F. Yang, and T. Li, "Modeling and broadening temporal user interest in personalized news recommendation," Expert Syst. Appl., vol. 41, pp. 3168-3177, 2014.

[3] Antonio Moreno, Lucas Marin, David Isern, David Perelló, "Dynamic learning of keyword-based preferences for news recommendation", Web Intelligence (WI) and Intelligent Agent Technologies (IAT), 2014 , vol. 1, pp. 347-354.

[4] M. D. Ekstrand, J. T. Riedl and J. A. Konstan, "Collaborative Filtering Recommender Systems", Foundations and Trends in Human–Computer Interaction, vol. 4, pp. 81-173, 2011.

[5] Vozalis, Manolis, and Konstantinos G. Margaritis. "Collaborative filtering enhanced by demographic correlation." *AIAI symposium on professional practice in AI, of the 18th world computer congress*. 2004.

[6] Koren, Yehuda, Robert Bell, and Chris Volinsky. "Matrix factorization techniques for recommender systems." *Computer* 42.8 (2009): 30-37.

[7] X. Han, W. Shang and S. Feng, "The design and implementation of personalized news recommendation system," *Computer and Information Science (ICIS), 2015 IEEE/ACIS 14th International Conference on*, Las Vegas, NV, 2015, pp. 551-554.

[8] Lu, Zhongqi, Zhicheng Dou, Jianxun Lian, Xing Xie, and Qiang Yang. "Content-Based Collaborative Filtering for News Topic Recommendation." In *AAAI*, pp. 217-223. 2015.

★★★