Collaborative Filtering in Recommender Systems

Bao Vinh Nguyen

Recommender Systems (RS) are software tools and techniques providing recommendations or suggestions for items that the user should use [10]. There are two main techniques, which are *content-based* (CB)[1] and *collaborative filtering* (CF). The paper focuses on the later technique and explains how Google applies CF in news recommendations.

Collaborative Filtering (CF) is the process of filtering or evaluating items based on other people's opinions [12]. The idea is based on word-of-mouth phenomena, when people share to each other about books they have read, restaurants they have tried, and movies they have seen, then use these discussions to form opinions. For example, Matt observes that among his colleagues, Amy has a history of recommending films that he likes, Bob usually recommends movies that he finds not enjoyable, and Jacob seems to recommend everything. Overtime, Matt learns whose opinions he should listen to in terms of determining movie qualities. Recommender systems go beyond a small word-of-mouth circle, they take opinions of thousands to millions of people then make personalized recommendations of items that the user should choose.

The original idea of collaborative filtering dates to work from the 1990's as a solution for dealing with information overload [5]. The original collaborative filtering system, Tapestry, allows the users to query for items in an information domain (e.g.: company's email) based on opinions

---

[1] For a brief explanation of CB and a comparison between CF and CB, see page 4 of the paper.

of others. Although the system requires effort from users, it supports the users overall by providing relevant information in a timely manner [4].

The paper uses movie recommendation to illustrate collaborative filtering's concepts, particularly the site IMDB[2], which is an online database of information related to films, television programs, etc. A *rating* consists of two factors, which are user and item, and can be visualized in a matrix form. A *user* means any individual providing ratings to a system [12]. A user of IMDB rates movies using 1 to 10 stars, where 1 is "Extremely not recommended" and 10 is "Must See". There are two types of ratings. *Explicit ratings* mean where a user is directly asked for opinions on an item. *Implicit ratings* are those inferred from users' actions, which differ from past behaviors [12].

The choice of ratings is the ultimate question in the beginning of system design process. While explicit ratings provide the most accurate description of users' preferences with least amount of data, collecting ratings requires much additional work from users. On the other hand, implicit ratings require little or no cost to users; however, ratings sometimes can be inaccurate [12]. However, explicit ratings recently have gained more popularity. First, we only need a small number of "early adopters" who rate frequently, providing sufficient information to create recommendations for the remaining users. Second, other users are more likely to enjoy the benefits from rating the products, as they might have a positive feeling from contributing to the community and gain gratification from other users [6].

In order to produce a CF system, we can use *probabilistic* algorithms, which explicitly represent probability distributions when calculating ranked recommendation lists or predicted

---

[2] https://www.imdb.com/

ratings [12]. The other branch of algorithms is *non-probabilistic*, which is more popular in the practitioner community. We will focus on two main algorithms under the non-probabilistic branch, which are *user-based nearest neighbor algorithms* and *item-based nearest neighbor algorithms* [12]. User-based nearest neighbor algorithm replicates the word-of-mouth process given enough users. The outcome of this method is to generate *neighbors,* which are users with similar choices. It analyzes ratings from users' neighborhood then generates item predictions. On another hand, item-based nearest neighbor algorithms are considered the transpose of the previous method. This approach generates predictions based on similarities between items [11].

However, each approach has its own challenges. For user-based nearest neighbor algorithm, *skewed neighbors,* where ratings data is sparse and pairs of users having few co-ratings are prone to have skewed correlations, can dominate a user's neighborhood [12]. Furthermore, the way this algorithm calculates correlation (Pearson) does not capture the distinction between agreement about a universally loved and a controversial movie, where the first should be much less weighted than the later. In terms of item-based algorithm, going through each single available item in the database may sometimes cause time and speed issues, as the number of correlations stored is too large [11].

There are three main usages for a CF system. First, it can show *recommended items,* where a list of items appears in an order of how useful they might be [12]. The rationale for the order is based on predictions of what the user would rate and rank the items. In the IMDB case, this can be what would be the movie's list for Matt given his past behaviors. Second, CF system shows *predictions for a given item*. This means given a specific item, the system shows potential ratings even for rarely rated ones. For example, IMDB gives potential ratings for newly introduced

movies. Third, CF deals with *constrained recommendations*. For instance, given that Matt only prefers comedy or family movies, what can be movies that he is likely to enjoy the most.

However, CF is not the only one method in recommender system. A potential issue for CF system is *cold-start,* situations that a CF system is unable to generate meaningful recommendations due to the lack of initial ratings [12]. This can happen due to new user, new item, and new community. For instance, newly signed up users have no previous record. New items introduced to the system has no ratings; therefore, they will not be recommended. New products or services have a small community, which cannot generate enough incentives for other users to provide ratings. To deal with those challenges, there is *content-based filtering* (CB) method, which is based on the assumption that items having similar features should be rated similarly [12]. This method is very effective in dealing with shortage of past items' ratings. Therefore, CF and CB are normally considered as complementary methods [1]. CB is normally used to predict relevance for items without ratings; and CF is used to analyze previous content and come up with a more personalized suggestion.

There has been no well-accepted metric that can accurately evaluate performance of a CF system. Current popular metrics depend on specific types of items and users. However, the most universal metric is *accuracy*, which has been used in all other machine learning problems [12]. In simple words, accuracy measures the gap between the predicted rating and the true rating, or the magnitude of errors between predicted and real ranking. Specifically, *predictive accuracy* means the ability of a CF system to predict users' ratings, in which the standard error is *mean absolute error (MAE)* – the average absolute difference between the prediction and the actual rating [12]. MAE is widely used due to its simplicity. However, it does not capture the importance of errors of items belonging to top of the list and the remaining items, in which users perceive errors of the top

recommended items more costly than errors of the remaining [9]. Another version of accuracy is *classification accuracy metrics,* measuring the frequency in which a recommender system generates correct or incorrect decisions [7]. Besides accuracy, a few other popular metrics are *novelty,* which is the ability of a CF system to recommends items that users are not potentially aware of without the help of the system and *learning rate,* which is the speed for the CF system to quickly produce low-error predictions [13].

Google has been applying these methods to develop personalized news recommendation systems in Google News. Google News is a computer-generated news website consolidating headlines from worldwide new sources [8]. The motivation stems from the nature of news reading: when a user visits a news website, he or she is looking for *new information,* which is information that the person did not know before [8]. Therefore, an important use case for recommender system is to understand how users' news interests change over time and the effectiveness of using past user activities to predict future behaviors.

Google gathers data by conducting a large-scale log analysis of its Google News users in order to gauge the stability of users' news interests. The data set includes anonymized click logs of users signing into their Google accounts from 2007/7/1 to 2008/6/30, with the total randomly sample size of 16,848 users [8].

Google previously applies CF method by recommending new stories read by users with similar click history [3]. The rationale behind this is that in the context of Google News, item ratings are binary, in which a click on a story equals to a 1 rating, while a non-click equals to a 0 rating [3]. Using such binary rating system, Google come up with news recommendations to users. Also, in such domains as news, a user's interest cannot always be classified by the topics or terms presented in a document [3].

However, they find two major drawbacks doing this way. First, they found *first-rater problem,* which is identical to the cold-start problem mentioned above. In this case, the system cannot recommend stories that have not been read by other users, causing serious problems as news websites normally updates the most updated information to users within a quick time frame [3]. Specifically, Google's CF system has to wait for several hours in order to collect clicks for recommendations, leading to time lags between suggestions and break-out news. Second, CF does not account for variability between users' preferences [2]. For instance, as entertainment news stories are generally popular, they are always the top suggestion list despite users' true preferences. Google solves this issue by building user profiles by recording users' click history from Web History and building a personalized section on Google News [8]. This provides enough information for CF to work and makes sure users' favorite topics always appear in the top.

Google then decides to combine both information (a.k.a. content based) filtering method with collaborative filtering method. Based on users' reading history, they suggest news articles having similar topics with the ones that have been clicked before. News articles in those categories will have higher ranks, which mean they are more likely to be recommended to the user. However, they choose articles falling under large categories instead of "fine grained" topics to avoid showing news that users already know [8].

They conduct live experiments by randomly assigning a control and a treatment group. For the test group, each time a user logs in Google News, a recommended news section is generated specifically for that user. For the control group, the users receive recommended news from the existing CF system. In this case, they use *click-through rates (CTR)* of the recommended news section, CTR of Google News homepage, and frequency of visiting Google News website as metrics to measure success [8]. CTR is calculated by the number of clicks by number of visits,

measuring the quality of the recommendations. The result shows that CTR in the treatment group is consistently higher than the CTR in the control group, within 33 of 34 days in the experiment [8]. Also, the frequency of website visit for treatment group is higher than the frequency for the control group. Therefore, the combined recommender system improves the quality of news recommendations.

REFERENCES

[1] Adomavicius, G. and Tuzhilin, A. (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), pp.734-749.

[2] Carreira, R., Crato, J., Gonçalves, D. and Jorge, J. (2004). Evaluating Adaptive User Profiles for News Classification. *9th international conference on Intelligent user interfaces*, [online] pp.1-7. Available at: http://web.ist.utl.pt/~daniel.j.goncalves/publications/2004/p2777-carreira.pdf [Accessed 17 Dec. 2019].

[3] Das, A., Datar, M., Garg, A. and Rajaram, S. (2007). Google news personalization: scalable online collaborative filtering. In: *16th international conference on World Wide Web*.

[4] Ekstrand, M., Riedl, J. and Konstan, J. (2010). Collaborative Filtering Recommender Systems. *Foundations and Trendsin Human–Computer Interaction*, 4(2), pp.81–173.

[5] Goldberg, D., Nichols, D., Oki, B. and Terry, D. (1992). Using Collaborative Filtering to Weave an Information Tapestry. *Communications of the ACM*, 35(12), pp.61-70.

[6] Harper, F., Li, X., Chen, Y. and Konstan, J. (2005). An Economic Model Of User Rating In An Online Recommender System. In: *10th International Conference on User Modeling*. Edinburgh, pp.216-307.

[7] Herlocker, J., Konstan, J., Terveen, L. and Riedl, J. (2004). Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems*, 22(1), pp.5-53.

[8] Liu, J., Pedersen, E. and Dolan, P. (2010). Personalized News Recommendation Based on Click Behavior. *2010 International Conference on Intelligent User Interfaces*, pp.1-10.

[9] McLaughlin, M. and Herlocker, J. (2004). A Collaborative Filtering Algorithm and Evaluation Metric that Accurately Model the User Experience. In: *SIGIR Conference on Research and Development in Information Retrieval*. pp.329-336.

[10] Resnick, P. and Varian1997, H. (1997). Recommender systems. *Communications of the ACM*, 40(3), pp.56-58.

[11] Sarwar, B., Karypis, G., Konstan, J. and Riedl, J. (2001). Item-Based Collaborative Filtering Recommendation Algorithms. In: *10th international conference on World Wide Web*. ACM Press, pp.285-295.

[12] Schafer J.B., Frankowski D., Herlocker J., Sen S. (2007) Collaborative Filtering Recommender Systems. In: Brusilovsky P., Kobsa A., Nejdl W. (eds) The Adaptive Web. Lecture Notes in Computer Science, vol 4321. Springer, Berlin, Heidelberg.

[13] Schein, A., Popescul, A. and Ungar, L. (2001). Generative Models for Cold-Start Recommendations. In: *Twenty-third Annual International ACM SIGIR Workshop on Recommender Systems*.