A hybrid approach to improving collaborative filtering recommendation system performance using k-nearest neighbors and genetic algorithm.

Mohit Kumar Singh
Indian Institute of Information Technology, Gwalior
imt_2019061@iiitm.ac.in

Elvis Theyo
Indian Institute of Information Technology, Gwalior
imt_2019026@iiitm.ac.in

Himanshu Pawar
Indian Institute of Information Technology, Gwalior
imt_2019040@iiitm.ac.in

Prof. Pramod Kumar Singh
Indian Institute of Information Technology, Gwalior
pksingh@iiitm.ac.in

***Abstract*—Recommendation systems have become an essential part of online platforms to provide personalized recommendations to users. Collaborative filtering (CF) is a commonly used technique in recommendation systems, but it suffers from several limitations such as the cold start problem, sparsity, and scalability. In this paper, we propose a hybrid approach to improving the performance of CF-based recommendation systems using k-nearest neighbors (k-NN) and genetic algorithms (GA). The proposed approach calculates the similarity between users in a collaborative filtering recommendation system and is formulated via a simple linear combination of values and weights. A group of similar users are initially obtained using the k-nearest neighbors and values are then calculated for each pair of users in this group between which the similarity is obtained, whilst weights are only calculated once, making use of a prior stage in which a genetic algorithm extracts weightings from the recommender system which depend on the specific nature of the data from each recommender system. The results obtained present significant improvements in prediction quality, recommendation quality, and performance.**

***Index Terms*—recommendation systems, collaborative filtering, genetic algorithms, k-nearest neighbors**

I. Introduction

Recommendation systems have become increasingly popular in recent years due to the vast amounts of data available online and the need to make sense of this data. Recommendation systems provide personalized recommendations to users based on their past behavior, preferences, and feedback. These systems are used in various domains such as e-commerce, social media, entertainment, and healthcare, among others. The goal of a recommendation system is to suggest relevant items to users, thereby improving their experience and satisfaction with a particular platform. Collaborative filtering (CF) is a commonly used technique in recommendation systems that utilizes the past behavior of users to make personalized recommendations. It is based on the idea that users who have similar preferences in the past are likely to have similar preferences in the future. There are two types of collaborative filtering techniques: user-based and item-based. User-based collaborative filtering finds similar users based on their past behavior and recommends items that have been rated highly by these similar users. Item-based collaborative filtering, on the other hand, identifies similar items based on the ratings given by users and recommends items that are similar to the ones that a user has rated highly. Collaborative filtering suffers from several limitations such as the cold start problem, sparsity, and scalability. The cold start problem arises when a new user or item joins the system and there is not enough data to make accurate recommendations. Sparsity is a common issue in recommendation systems because users typically rate or interact with only a small fraction of the items available. Scalability is a challenge when the size of the user-item matrix is very large, and the computation time required to make recommendations becomes a bottleneck.

A genetic algorithm is an optimization algorithm inspired by the principles of natural selection and evolution. GAs have mostly been used in two aspects of RS: clustering and hybrid user models. A common technique for improving the features of RS is to first perform clustering on all of the users, resulting in a group of classes of similar users; then, the desired CF techniques can be applied to each of the clusters, yielding similar results but in much shorter calculation times; in these cases, common genetic clustering algorithms such as GA-based K-mean are used. The RS hybrid user models commonly use a combination of CF with demographic filtering or CF with content-based filtering, to exploit the merits of each one of these techniques. In these cases, the chromosome structure can easily contain demographic characteristics and/or those related to content-based filtering.

The proposed approach in this paper leverages GA, but it has the advantage of not requiring the additional information offered by hybrid user models, so it can be utilized in all contemporary RS that are based solely on CF techniques.

II. Literature Survey

Punam Bedi et al. [1] introduced TARS, a trust-based ant recommender system that generates relevant recommendations by including the concept of dynamic trust among users and selecting the best neighborhood based on the genetic image of ant colonies. They used two datasets, the Jester dataset, and MovieLens dataset, for performance evaluation.

Sang Hyun Choi et al. [2] proposed HYRED, a hybrid recommendation algorithm that combines collaborative filtering with content-based filtering to yield useful information from a large dataset and avoid sparsity and scalability problems. They used modified Pearson's binary correlation coefficients for collaborative filtering and generalized distance to boundary-based rating for content-based filtering.

Beel et al. [3] analyzed several recommendation techniques with both online and offline evaluations and found contradictions among the results of both evaluation methods. They concluded that offline evaluations may be unsuitable for the evaluation of research paper recommender systems.

Qing Li et al. [4] proposed a technique that integrates content information into item-based collaborative filtering to resolve the cold start problem. They used clustering techniques to analyze the proposed technique and conducted experiments on the MovieLens dataset, which showed an improvement in recommendations provided by using item-based collaborative filtering and resolved the cold start problem.

J. Bobadilla et.al. [5] proposed a metric for computing user similarity in collaborative filtering-based recommender systems. The metric is formulated using a linear combination of values and weights and can be applied to improve the prediction quality and performance. The weights are calculated only once, while the values are calculated for every pair of users with similar preferences. The proposed approach has shown significant improvements in the prediction quality and performance of collaborative filtering-based recommender systems.

III. Methodology

A. k-Nearest Neighbors

The k-nearest neighbors (k-NN) algorithm is a popular technique used in machine learning and data mining for classification and regression. k-NN algorithm is a supervised machine learning algorithm that assumes that similar things exist in close proximity and puts a new unknown data point into a category that is closest to the other available categories. It is a non-parametric and lazy learning algorithm, meaning it doesn't make any assumptions about the underlying data distribution and doesn't require any training to make predictions.

The k-NN algorithm works by calculating the distance between the new data point and all other data points in the dataset. It then selects the k-nearest neighbors to the new data point based on the calculated distances. The neighbors are typically determined using Euclidean distance or other distance metrics. Once the k-nearest neighbors are identified, the algorithm can make a prediction based on the class labels or values of the neighbors.

One of the main advantages of the k-NN algorithm is its simplicity and ease of implementation. It can be used for both classification and regression problems, and it can handle non-linear and complex data distributions. Additionally, the k-NN algorithm is non-parametric, meaning it can be used with any type of data, regardless of its underlying distribution.

B. Proposed Approach

In this paper, the k-NN algorithm is used to identify a group of similar users in the system. Given a user, the k-NN algorithm calculates the similarity between that user and all other users in the system. This helps us to obtain an initial set of users that are highly likely to have similar preferences. This allows us to significantly reduce the number of comparisons required to generate recommendations, as we only need to calculate values for each pair of users in the initial group, rather than comparing all pairs of users in the system. The weights are calculated and optimized using a genetic algorithm, which extracts weightings from the recommender system based on the specific nature of the data from each recommender system. By only calculating the weights once, our approach reduces the computational overhead required for the recommendation process, while ensuring that the weights are tailored to the specific dataset being used.

C. Values

We will consider a RS with a set of U users, {1, ... , U}, and a set of I items {1, ... , I}. Users rate those items they know with a discrete range of possible values {m, ... , M}. The ratings made by a particular user x can be represented by a vector, $r_x = (r_x^{(1)}, r_x^{(2)}, ..., r_x^{(I)})$ with dimension I (the number of items in the RS) in such a way that r_x^i represents the rating that the user x has made over the item i. Obviously, a user may not rate all the items in the RS. We will use the symbol • to represent that a user has not rated an item.

Next, we will consider an example. We will consider that m=1 and M=5 (that is to say the possible set of ratings is 1, ... , 5), there are nine items, I = 9, and there are two users whose ratings are represented by the following r_1 and r_2 vectors:

$\mathbf{r_1} = \{4, 5, \bullet, 3, 2, \bullet, 1, 1, 4\}$

$\mathbf{r_2} = \{4, 3, 1, 2, \bullet, 3, 4, \bullet, 2\}$

As may be seen, while user 1 has rated item 5, $r_1^5 = 2$, user 2 has not rated item 5 yet: $r_2^5 = \bullet$.

In order to compare both vectors, r_x, r_y, we can consider another vector $v_{x,y} = (v_{x,y}^0, ..., v_{x,y}^{M-m})$ whose dimension is the number of possible ratings a user can make over an item. Each component $v_{x,y}^i$ of the vector $v_{x,y}$, represents the ratio of items, j, rated by both users (that is to say, $r_x^j \neq \bullet$ and $r_y^j \neq \bullet$) and over which the absolute difference between the ratings of both users is i ($|r_x^j - r_y^j = i|$) , to the number of items rated by both users. That is to say, $v_{x,y}^i = a/b$ where b is the number of items rated by both users and a is the number of items rated by both users over which the absolute difference in the ratings of both users is i.

In this way, the vector $v_{1,2}$ for the previous example is the following (just observe that there are only 5 items rated by both users 1 and 2): $v_{1,2} = (\frac{1}{5}, \frac{1}{5}, \frac{2}{5}, \frac{1}{5}, 0)$

D. Similarity

We will consider a family of similarity functions. For each vector $\mathbf{w} = (w^0, ..., w^{M-m})$ whose components lie in the range [1, 1], we will consider the following similarity function:

$$sim_w(x, y) = \frac{1}{M-m+1} \sum_{i=0}^{M-m} w^{(i)} v_{x,y}^{(i)}$$

Consequently, for each vector, w, there is a similarity function whose component, w_i , represents the importance of the component $v_{x,y}^i$ for calculating the similarity between two users.

The question related to obtaining the most suitable similarity function represented by a vector **w** for a recommender system depends on the nature of the data in this recommender system. We will try to find, among all the possible vectors **w**, one representing a similarity function that provides a minimal Mean Absolute Error (MAE) in the recommender system. In order to obtain this similarity function, we will use a genetic algorithm.

E. Genetic Algorithm

In order to find an optimal similarity function, sim_w, we use a genetic algorithm to find the vector **w** associated to the optimal similarity function sim_w.

We use a supervised learning task whose fitness function is the Mean Absolute Error MAE of the RS. In this way, the population of our genetic algorithm is the set of different vectors of weights, **w**. Each individual, that is to say, each vector of weights, represents a possible similarity measure, for which we will evaluate the MAE of the RS using this similarity measure. While running our genetic algorithm, the successive population generations tend to improve the MAE in the RS. Our genetic algorithm stops generating populations when MAE in the RS for a vector of weight is lower than a threshold, γ (which has been previously established).

As usual, we use only a part of the whole recommender system (training users and training items) in order to obtain an optimal similarity function. Once obtained this similarity function, we carry out our experiments to evaluate the similarity function obtained by using only the test users and the test items (that is to say, those users and items which have not been used in the GA algorithm).

F. Genetic Representation

As usual, individuals of populations are represented in binary form as strings of 0s and 1s. As we have previously stated, in our genetic algorithm, each vector of weights, **w**, representing a similarity vector is a possible individual of the population. Each component w^i in the vector of **w** will be represented by 10 bits. Consequently, the vector $w = (w^0, ..., w^{M-m})$ will be represented by the following string of 0s and 1s (with a length of $2^{10(M-m+1)}$ bits):

$b_{M-m}^9 ... b_{M-m}^1 b_{M-m}^0 b_{M-m-1}^9 ... b_{M-m-1}^1 b_{M-m-1}^0 ... b_1^9 ... b_1^1 b_1^0 b_0^9 ... b_0^1 b_1 b_1^0$

where each component of the vector $w_i \in [-1, 1]$ can be obtained through the following expression:

$$w_i = \frac{2 \sum_{j=0}^{9} 2^j b_i^j}{2^{10} - 1} - 1$$

G. Initial Population

For determining population size, we considered the criterion of employing a number of individuals in the population that is double of the number of bits used to represent each individual. In order to generate the initial population we have considered the following criteria:

- 50% of the initial population is obtained using random values.
- 50% of the initial population is seeded in areas where optimal vectors ***w*** may lie.

The value w_0 thus represents how the similarity function weights the number of items rated identically by two users. As a result, we can expect an optimal similarity function to have a high positive value of w(0). The value of w_0 of individuals will thus be picked at random to lie within a high positive range of values. In the same manner, the value $w_{(M-m)}$ measures how the similarity function weights the number of items rated in a totally opposite way by two users. As a result. we can reasonably expect that an optimal similarity function will have a high negative value of $w_{(M-m)}$. In this way, the value $w_{(M-m)}$ of individuals will be chosen randomly to lie within a high negative range of values.

H. Fitness function

The fitness function of a genetic algorithm is used to prescribe the optimality of a similarity function sim_w (for the individual **w**). In our genetic algorithm, the fitness function will be the Mean Absolute Error (MAE) of the RS (indeed we only use the training users and the training items since we are trying to find an optimal similarity function) for a particular similarity function sim_w.

The MAE is obtained by comparing the real ratings with the predicted ratings made according to the similarity function. In order to calculate the MAE of the RS for a particular similarity function, sim_w, we need to follow the next steps for each user x:

1) Obtaining the set of K neighbors of x, K_x (the K most similar users to a given user x).
2) Calculating the prediction of the user x and item i, p_x^i. This is obtained through the Deviation From Mean (DFM) as an aggregation approach:

$$P_x^i = \bar{r_x} + \frac{\sum_{n \in K_x} [sim_w(x,n) * (r_n^i - \bar{r_n})]}{\sum_{n \in K_x} sim_w(x,n)}$$

where $\bar{r_x}$ is the average of ratings made by the user x.

Once every possible prediction according to the similarity function sim_w is calculated, we obtain the MAE of the RS as follows:

$$\text{fitness} = \text{MAE} = \frac{1}{\#U} \sum_{u \in U} \frac{\sum_{i \in I_u} |p_u^i - r_u^i|}{\#I_u}$$

When running the genetic algorithm, $\#U$ and $\#I_u$ represent respectively the number of training users and the number of training items rated by the user u.

I. Genetic Operators

As usual, our genetic algorithm uses the common operators in genetic algorithms: selection, crossover, and mutation. The features of our genetic operators are:

- *Selection*: The chosen method is the fitness proportional selection. That is to say, the selection probability of an individual depends on its fitness level.
- *Crossover*: We use the one-point crossover technique. The crossover probability is 0.8.
- *Mutation*: We use a single-point mutation technique in order to introduce diversity. The mutation probability is 0.02.

J. Reproduction and termination

When using the reproduction operator, 1000 individuals are generated in each generation using random crossover between the best-fit parent individuals. The number of individuals keeps constant through every generation. We only keep the 5% of the best individuals from each generation to obtain the next one (elitist selection). The genetic algorithm stops when there is an individual in the population with a fitness value lower than a constant γ.

IV. Conclusion

In this paper, we have presented an approach for obtaining optimal similarity functions. The similarity functions obtained provide better quality and quicker results than the ones provided by the traditional metrics. Improvements can be seen in the system's accuracy (MAE), coverage, and in precision & recall recommendation quality measures.

Our approach for improving collaborative filtering recommendation system performance using k-nearest neighbors and genetic algorithm provides significant improvements in both prediction quality and overall system performance. By identifying a group of similar users using the k-NN algorithm and optimizing weights with a genetic algorithm, our approach reduces the number of comparisons required for generating recommendations while ensuring that the recommendations generated are highly accurate and tailored to the specific dataset being used. Our approach is effective for a variety of different recommender systems and datasets, making it a versatile and powerful tool for improving recommendation performance. With the increasing importance of recommendation systems in modern applications, our approach provides a valuable contribution to the field, helping to improve the quality and efficiency of recommendation systems in a range of different domains.

References

[1] P. Bedi, R. Sharma, Trust-based recommender system using ant colony for trust computation, 2012.

[2] S. Choi, Y. Jeong, A Hybrid Recommendation Method with Reduced Data for Large-Scale Application, 2010.

[3] J. Beel, S. Langer, A comparison of Offline Evaluations, Online Evaluations, and User Studies, 2014

[4] Q. Li, A content-enhanced approach for cold-start problem in collaborative filtering, 2011.

[5] J. Bobadilla, F. Ortega, A. Hernando, J. Alcalá, Improving collaborative filtering recommender system results and performance using genetic algorithms, 2011.

[6] J. Bobadilla, F. Serradilla, J. Bernal, A new collaborative filtering metric that improves the behavior of recommender systems, Knowl. Based Syst. 23 (6) (2010) 520–528.

[7] J. Bobadilla, F. Ortega, A. Hernando, A collaborative filtering similarity measure based on singularities, Inf. Proc. Manage. (2011), doi:10.1016/j.ipm.2011.03.007.

[8] H. Denis. Managing collaborative learning processes, e-learning applications. 29th Int. Conf. Inf. Tech. Interfaces, (2007) 345–350.

[9] L.Q. Gao, C. Li, Hybrid personalized recommended model based on genetic algorithm, Int. Conf. Wireless Comm. Netw. Mobile Comput. (2008) 9215–9218.

[10] G.M. Giaglis, Lekakos, Improving the prediction accuracy of recommendation algorithms: approaches anchored on human factors, Interact. Comp. 18 (3) (2006) 410–431.

[11] J.L. Herlocker, J.A. Konstan, J.T. Riedl, L.G. Terveen, Evaluating collaborative filtering recommender systems, ACM Trans. Inf. Syst. 22 (1) (2004) 5–53.

[12] Y. Ho, S. Fong, Z. Yan, A hybrid ga-based collaborative filtering model for online recommenders, Int. Conf. e-Business (2007) 200–203.

[13] H. Ingoo, J.O. Kyong, H.R. Tae, The collaborative filtering recommendation based on SOM cluster-indexing CBR, Exp. Syst. Appl. 25 (2003) 413–423.

[14] H. Jinghua, W. Kangning, F. Shaohong, A survey of e-commerce recommender systems, Int. Conf. Service Syst. Service Manage. (2007) 1–5.

[15] K. Kim, H. Ahn, Using a clustering genetic algorithm to support customer segmentation for personalized recommender systems, Artif. Intell. Simulat. 3397 (2004) 409–415.

[16] K. Kim, H. Ahn, A recommender system using GA K-means clustering in an online Shopping market, Expert Syst. with Appl. 34 (2) (2008) 1200–1209.

[17] M. Knights. Web 2.0. IET Comm. Eng. (2007) 30–35.

[18] J.A. Konstan, B.N. Miller, J. Riedl, PocketLens: toward a personal recommender system, ACM Trans. Inf. Syst. 22 (3) (2004) 437–476.

[19] B. Krulwich, Lifestyle finder: intelligent user profiling using large-scale demographic data, Artif. Intell. Magaz. 18 (2) (1997) 37–45.

[20] K. Lang. NewsWeeder: Learning to filter netnews, 12th Int. Conf. Machine Learning, (1995) 331–339.

[21] P. Li, S. Yamada, A movie recommender system based on inductive learning, IEEE Conf. Cyber. Intell. Syst. 1 (2004) 318–323.

[22] K.J. Lin. Building Web 2.0. Computer, (2007) 101–102.

[23] Y. Manolopoulus, A. Nanopoulus, A.N. Papadopoulus, P. Symeonidis, Collaborative recommender systems: combining effectiveness and efficiency, Exp. Syst. Appl. 34 (4) (2008) 2995–3013.

[24] M. Papagelis, D. Plexousakis, T. Kutsuras, Alleviating the sparsity problem of collaborative filtering using trust inferences, LNCS 3477 (2005) 224–239.

[25] C. Porcel, E. Herrera-Viedma, Dealing with incomplete information in a fuzzy linguistic recommender system to disseminate information in university digital libraries, Knowl. Based Syst. 23 (1) (2010) 32–39.

[26] C. Porcel, J.M. Moreno, E. Herrera-Viedma, A multi-disciplinary recommender system to advice research resources in university digital libraries, Exp. Syst. Appl. 36 (2009) 12520–12528.

[27] F. Zhang, H.Y. Chang, A collaborative filtering algorithm employing genetic clustering to ameliorate the scalability issue, IEEE Int. Conf. e-Business Eng. (2006) 331–338.

[28] A. Verma, H.K. Virk, A Hybrid Recommender System using Genetic Algorithm and kNN Approach.