AN ADAPTIVE HESITANT FUZZY SETS BASED GROUP RECOMMENDATION SYSTEM

Rohith Jayaraman[1], V. Subramaniyaswamy[2*], Logesh Ravi[3]

[1,2]School of Computing, SASTRA Deemed University, Thanjavur, India

[3]Sri Ramachandra Faculty of Engineering and Technology, Sri Ramachandra Institute of Higher Education and Research, Chennai, India

Email: the.rohith.jayaraman@gmail.com[1], swamy@cse.sastra.edu[2*] (corresponding author), LR@sret.edu.in[3]

DOI: https://doi.org/10.22452/mjcs.sp2020no1.9

ABSTRACT

Accurate group movie recommendation systems are a need in society today. We find that people tend to watch movies in groups rather than by themselves. However, the groups of people that tend to watch movies together are very diverse. In the existing methods, the characteristics of individual users are simply aggregated to obtain the group's attributes and most of the time useful data is not utilized. This can be improved upon by ensuring the utilization of all the data that we are presented with from the scenario. The method proposed in this paper is termed integrated as we weighed in the individual traits of each user in the group when predicting the group's rating for a movie. We used the concept of Hesitant Fuzzy Sets (HFS) in order to keep track of the characteristics of each of the users individually. The method we proposed in this paper employs Matrix Factorisation (MF) based Collaborative Filtering (CF) along with hesitant fuzzy sets. The way we performed MF based CF for a group is that we found the factors first and then formed the groups. The ratings were then predicted for these groups. The groups we have considered are of three sizes - 3 users, 5 users, and 10 users.

Keywords: ***Group Movie recommendation, Collaborative Filtering, Hesitant Fuzzy Sets, Matrix Factorization***

1.0 INTRODUCTION

Movies have become a part of everyday life. We see different movies almost every week now. Each of them with their own genre and storyline. Most people tend to go and watch these movies in groups. People are often indecisive about which movie they are going to watch. Generally, people try and solve this problem by asking friends or family members for suggestions. They may also search online for movies with good reviews and ratings so that they may watch them. This is where accurate recommendations can come in and help [7].

A recommender system can suggest movies for the group based on different methods. The most popular recommender systems for movies are Content-based systems, Collaborative Filtering based systems and Hybrid systems (which as the name suggests, tries to combine the first two methods) [15, 16, 23]. Now any of these may also be used for group recommendation. The only change is that when employed for groups of users instead of a single user, these methods perform an average of individuals' traits to obtain the group's characteristics. In this paper, we propose the use of HFS to model hesitation and store individuals' traits, in opposition to the popular mechanism of simply taking the mean of individual users' characteristics. A group can have any number of people but in this paper, we have specifically considered groups of 3 sizes - 3, 5 and 10 members respectively. We have randomly chosen the set of users that are picked to be a part of the group. We can also make use of a different method or algorithm to specifically group similar or dissimilar users as well.

The problem statement is that our algorithm needs to give a good movie recommendation to a particular group of users. We achieve this through our new proposed method which takes hesitant fuzzy sets to map the characteristics of different users in a group. We use this along with a Matrix Factorisation based Collaborative Filtering model in order to predict a good recommendation.

The key contributions of this paper are:

- Introducing an integrated approach that uses Hesitant Fuzzy Sets to keep track of the characteristics of individual users in a group
- Taking a weighted average of users' characteristics to obtain a group's characteristics rather than just simply calculating the average
- Proposing a new approach to group movie recommendations that is an improvement on the traditional Collaborative Filtering model

This paper has its own limitations. The proposed method is quite slow. It might take a long time to process for some huge datasets. This method also considers only one type of formation of group characteristics i.e. this paper considers only the case when group factors are calculated after Matrix Factorisation is done. There are multiple other methods that can also be used and this paper does not consider them.

The paper is organized in the following way: Section 2 elaborates on the literature review, Section 3 explains the model used and related concepts, Section 4 details the experimentation of the method on a dataset as well as the results of it and Section 5 gives a conclusion of the paper.

2.0 LITERATURE REVIEW

Group movie recommendation is not a completely new area. Ever since the discovery of movies, there has been a need for movie recommendations and in more recent times, group movie recommendations [20]. The springing up of companies like Netflix and amazon prime amongst others has only greatened the need for an accurate recommendation system. Though it may seem easy on the surface, group movie recommendation has always had its own complications [11, 21]. However, there have been many methods that have been proposed to give accurate movie recommendations.

The initial idea was to bring in a CF method used in movie recommendations to group movie recommendations as well [24]. On that trail of thought, one of the first approaches was to group the users using a k-means algorithm [1, 9, 10]. The groups were then sent to a CF algorithm that predicted the ratings which was used to predict the recommendations [22]. This method was great as it gave a fairly accurate prediction. The idea to use k-means was one that seemed to work well.

Another popular approach that was used was Content-based feature preferences [2, 12, 17-19] which acted as a base for many methods that used CF to recommend movies. The core idea of this method was to have a set of features and to find which features a particular user seemed to be highly appreciative of. If a user liked a particular feature to a very high extent then we recommend movies that have said feature, which increases the probability of the user liking that movie. Why this paper in particular was important was that it does not follow the traditional method of considering only genres of movies as features. It took the actors, directors, producers, etc into consideration as features which can definitely increase the accuracy of the recommendation system.

The problem was that the methods mentioned above introduced a sort of bias because using a k-means algorithm effectively ensures that similar users get grouped together and the chances of similar users liking the same movie are very high. If the group is made up of similar users then CF will perform well. To this end, the authors of [5, 14] took a random group and then performed a group movie recommendation via MF based CF. They mention three different forms of MF based CF-based purely on when the factors are calculated and when the groups are formed. This provided a great recommendation system as it performed well even for a random set of users in a group, not only similar ones.

Eventually, people realized that individual users' traits cannot just be averaged. So they decided to keep track of individual traits of users using HFS [6]. Here, they tried to bring in individual users' traits into the model in order to ensure higher accuracy. They used K-Nearest Neighbours (KNN) based CF to predict the ratings and recommend a movie. This method was unique when compared to most others as it did not simply take the mean of individual users'

characteristics. This gave each group a uniqueness during the prediction which can be beneficial for a more accurate system.

We kept all the previously mentioned methods in mind and proposed our methodology. For a group recommendation, it is important that the users were randomly chosen as similar users always made it easy to get a great performance from a recommendation system. The actual preference of each of the users in a group mattered as well. We could not simply average the ratings of all users in a group to get the group's ratings.

To explain this further, we considered an example. If there is a group of 4 users, who for a particular movie gave a rating of 3,3,3,3 respectively (out of 5) then the average rating for that movie is 3. However, if they gave 5,1,1,5 as the ratings then again, the average is 3. We see that in both cases the average rating of the group for the given movie is 3. The average ratings from both the scenarios is the same even though the first scenario is one where all the users liked the movie to approximately the same extent but in the second, two users loved the movie while two hated it. This information is completely lost due to the calculation of group rating as the mean of individuals' rating. This further showed the need for keeping track of the individual traits of each user using HFS. Also, we used a method like MF based CF because a KNN often performs poorly when compared to MF when it is used for high amounts of data or for users that are dissimilar in nature. Since the group is formed randomly, we cannot be sure of what kind of users are to be paired together and so we must use a robust method like MF based CF. So the proposed approach tried to give a more accurate recommendation system using what we termed as an integrated approach.

3.0 PROPOSED MODEL

The model proposed is one that uses HFS and MF based CF. It is necessary to first get an introduction to some of the methods used in this experiment.

3.1 HFS (Hesitant Fuzzy Set)

A Hesitant Fuzzy Set is basically an extension of the concept of fuzzy sets. A fuzzy set is one that maps a given value to a probabilistic value between 0 and 1 (limits inclusive) [3, 8]. This is known as the relational degree of the input value. An HFS is a set of fuzzy sets. In a fuzzy set, each value is mapped only to one value in the range [0,1] [4]. So an HFS can have multiple degree mappings for a single value. So for a given input, there can be a set of values in [0, 1] that are mapped to it. This is the concept of HFS. Using this concept, we mapped multiple degrees (users' characteristics) to a single input value(group characteristics). An overall diagram of the proposed model is given in Figure 1.

3.2 MF based CF (Matrix Factorisation based Collaborative Filtering)

MF based CF is just one of the many methods that can be used in a recommender system. Collaborative Filtering is based on the core principle that when there are similar users, what is liked by one user will be liked by another as well. However, this method suffers from a cold start and data sparsity problem. The data sparsity problem is solved by using the method of mean normalisation. Matrix Factorisation is a method that tries to break down a matrix (AxB) into two matrices (AxC and CxB) where C represents factors of the problem. So if we can break the input matrix down into these two matrices then we can estimate what the value of B will be for a given A by using the data we have to estimate values for C. So Matrix factorization is responsible for assigning right values to the factors (Figure 1).

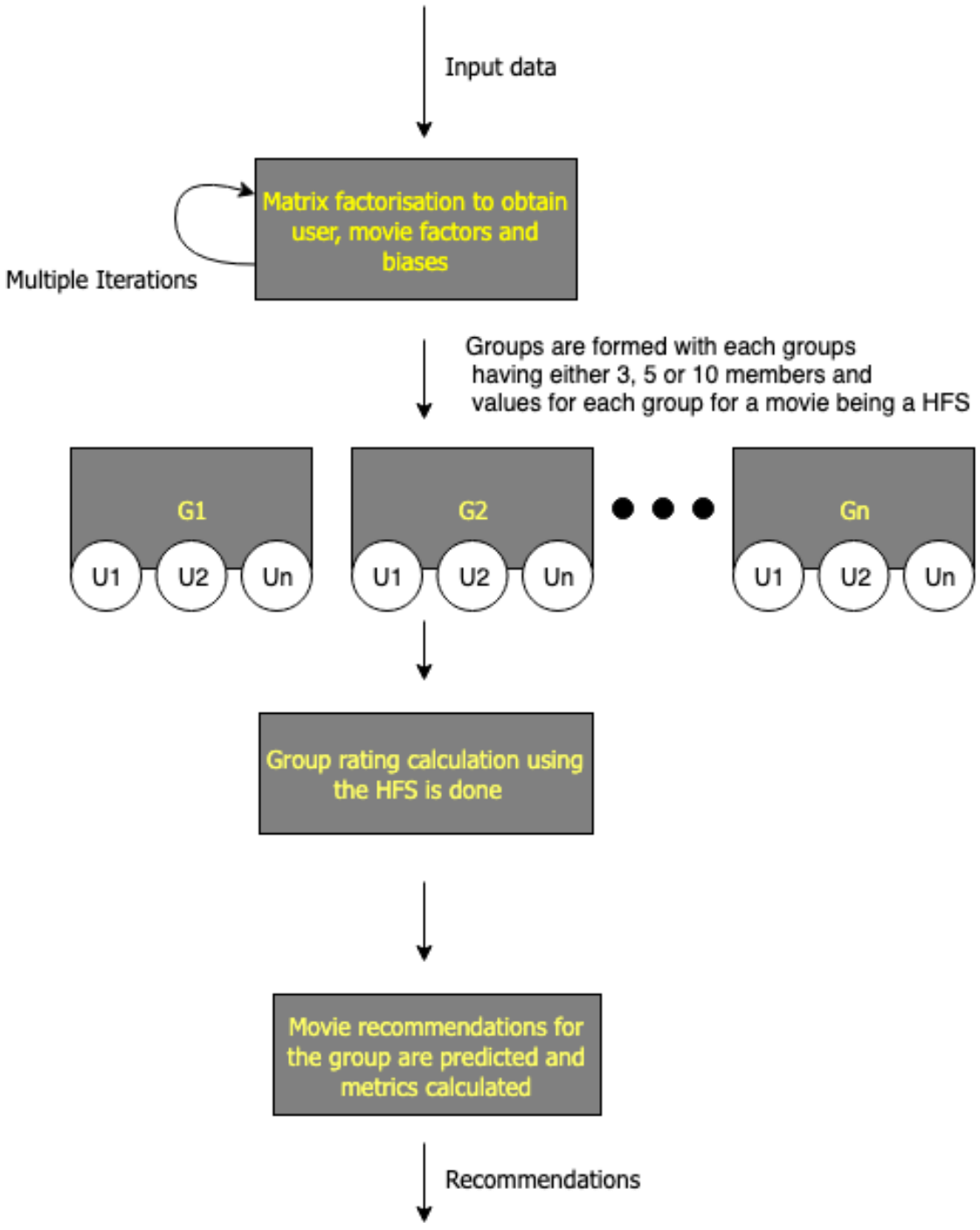

Fig.1: Basic Workflow of the proposed model

3.3 SVD++ (Single Value Decomposition)

In this paper, we also used a technique known as Single Value Decomposition ++ (SVD++) which tries to find out the two matrices AxC and CxB (mentioned above) in one shot. Often, trying to factorize the matrix can be a difficult task but SVD tries to help with that by estimating the 2 matrices simultaneously. We assumed a random initial value for the 2 matrices and performed multiple iterations in which we try to minimize the matrices until they are equal to the

original matrix. The way to do this is to use a gradient descent mechanism and to try and minimise the mean squared error that we got in each step. The gradient descent mechanism tries to find the local minimum by generally considering the slope and seeing its value. Each time we made certain adjustments to the values based on how much the Mean Squared Error (MSE) was. It was an iterative process that we stopped when the 2 matrices obtained were close to the original matrix when combined. This helped us to factorize the original matrix easily.

3.4 Algorithm

```
max_iterations = 20
group_sizes = [3, 5, 10]
max_no_of_groups = 10
average_rating = average of all ratings given by all users for all movies
Assume initial value for users matrix and items matrix as small random numbers
no_of_iterations = 0
while no_of_iterations < max_iterations
        factorise the original matrix using  SVD ++ along with SGD and regularisation to split it into 2 matrices
        calculate the root mean squared error of the matrix obtained when the 2 individual matrices are combined
        make adjustments to correct this error (the aim is to try and minimise the value obtained from eq. 3.1)
for size in group_sizes
        no_of_groups = 0
        while no_of_groups < max_no_of_groups
                pick {size} number of users randomly from all possible users
                if users have at least 50 common movies in test set
                        store group in groups
                        no_of_groups +=1
for group in groups
        group_bias[group]=0
        for user in group_users[group]
                group_bias[group] += user_bias[user]
                group_factor[group][user] = user_factor[user]/highest_rating[user]
                (this is mapped as a Hesitant fuzzy set)
                average_group_rating += average_user_rating[user]
        group_bias[group] /= group_size
        average_group_rating[group] /= group_size
for group in groups
        for movie in movies
                weighted_average = 0
                for each user in group
                        base_rating[user] = transpose of group_factor[group][user] * movie_factor[movie]
                weighted_average = weighted_average(base_rating)
                rating[group][movie] = average_rating + average_group_rating[group] + group_bias[group] +
                movie_bias[movie] + weighted_average
for movies in test_set
        compare actual_rating with rating
        (if at least one person has seen the movie then we take it into consideration for testing)
```

3.5 Processes involved in the model

The various calculations in the model are described in this section.

3.5.1 Factorisation

Our initial matrix was the Users x Movies matrix. We used an MF based CF algorithm that employs SVD++ or Single Value Decomposition ++ which is an extension of the SVD algorithm. It is similar to the SVD algorithm in the sense that it used the SGD or Stochastic Gradient Descent algorithm to learn and approximate the value of the factors via the minimization of traditional squared error or regularized squared error. The algorithm corrects itself in each iteration to try and get a more accurate value of the factors and biases of the users and items. We initially assumed that the values of these terms are small random numbers and then performed a number of iterations of SVD++ to obtain the required values. Our method assumed that the intermediate factor matrix has 20 factors as this is the number of genres in the given data. We tried to approximately map the behavior of each user to each genre. We used a regularisation mechanism in order to avoid overfitting. We did this so our model was generally correct and not specific just to a single case. L1 regularisation was used.

We represented the known values as follows:

- $q_m = (q_{m,1} \ldots q_{m,k})$ - a vector consisting of the factors for movie m
- b_m = movie m's bias
- $p_u = (p_{u,1} \ldots p_{u,k})$ - a vector consisting of the factors of user u.
- b_u - user u's bias
- μ - average rating of the dataset
- $r_{u,m}$ - user u's rating of movie m

So here, we say q_m is a vector for a given movie. We considered each genre to be a factor and so the vector was a collection of the values of each factor (unique to each movie). Each movie will also have a common bias that is represented for a movie by the term b_m. This bias was to undo the mean normalisation of a movie. Similarly, we had a vector and bias for each user as well. The overall average rating for the dataset was represented by mu. Finally, the term that represented the rating of a movie by a user is $r_{u,m}$

We then said that the goal of the matrix factorization was to minimize the following equation:

$$\sum(r_{u,m} - \mu - b_u - b_m - p_u^T q_m)^2 + \lambda(||p_u||^2 + ||q_m||^2 + b_u^2 + q_m^2) \quad (1)$$

The above equation was summed over all possible ratings which are non-zero i.e. for all ratings given by a user after having watched a movie. From the above equation, we find the factors and biases. So the equation was the same for all factors and movies. We used this equation and substituted the known values in order to find the intermediate matrix using which we estimated future ratings given a user and a movie.

3.5.2 Movie Rating Prediction

For predicting the movie rating based on the given factors and bias, we needed to decide when we were doing the factorization. We did the factorization calculation initially and then divided users into groups. We chose this method as this had the most room for implementing HFS. There are multiple other methods that can be implemented like aggregating the group's score before factorization or by applying weights. However, the method we used is a very efficient method for a general scenario. We also considered groups of 3 sizes to give a better understanding of the performance of the method.
The rating of a movie for a given user may then be predicted as:

$$m_{u,i} = \mu + b_i + b_u + p_u^T q_i \quad (2)$$

where $m_{u,i}$ is the predicted rating for item i by user u.

3.5.3 Group Formation

Once matrix factorization is done we got the users' factors, biases as well as the movies' factors, biases. We then formed groups randomly by choosing a set of random users. We checked if it is feasible for this group to be formed which was basically a check to see if the members of the group had 50 or more movies in common in the test data set so that we may efficiently evaluate our method. If this was true, then the group was formed. This was done because if the users did not have a common set of movies then our method will suffer from the cold start problem and so we would not be able to measure the performance properly. We formed a total of 10 groups in each of the sizes (3 members, 5 members, and 10 members) with each having unique members i.e. one member was not a part of multiple groups. The average of the performance across the 10 groups in each size gave us the performance of our method for that group. It was important to consider different group sizes as this ensured that we have variety and diversity amongst our execution. If we took only a particular size then it might not be applicable to many cases and so we tried to cover as many cases as we could.

3.5.4 Group Characteristics and Rating Prediction

Once the groups were formed we then moved on to the group's characteristics. Now we had two sets of characteristics, namely the users' factors, users' biases and the movies' factors, biases. We had to aggregate the factors and the biases. However, if these were done as a simple average, as soon as the group is formed then there was a huge loss in data. This is where our method differed from most others. Most methods simply average the factors here and so the factors are turned into averages. The individual uniqueness of each factor was then lost. So tried to retain the factors and biases for as long as possible without having to aggregate them. However, we knew that the biases are unique to each user and independent of any other interaction. So, the bias was not affected by the group at all. Therefore, we calculated the bias for the group as the simple average of the bias of all of its group members since this would not change irrespective of when we calculated it. For the group, we did the following to calculate characteristics :

1. Calculated bias of group as average of bias of all users
2. Calculated average rating of group as average of the average rating of all users
3. Represented characteristics of all users in the group as a member of hesitant fuzzy set :
   (($u'_{0,0}$ ... $u'_{0,k-1}$), ... ,($u'_{n,0}$ ... $u'_{n,k-1}$))

where u' is calculated as $u_{i,j}$ / highest-rating of user

So we modified Eq. (2) for group ratings and got:

$$y_{G,i} = \mu + \mu_G + b_i + b_G + f(G,q_i) \quad (3)$$

where $y_{G,i}$ was the predicted rating of members of group G for the movie i, μ_G was the average of the highest rating of users in group G (this was to compensate for the normalized mapping used as part of the hesitant fuzzy set where every rating is mapped onto a value that is between 0 and 1). $f(G,q_i)$ was a function that represents the weighted average of the values obtained by the multiplication $p_u^T q_i$ for every member u in group G. The weights for the function f were obtained by the number of ratings given by each user.

The logic behind the calculation of weights for function f is that the user in the group who has watched the most number of movies would be the one that the other group members would listen to or agree with the most. So the function represented a value that would otherwise be simply obtained as the dot product between $p_G^T q_i$ which would be neglecting the effect of each user on the group. Our proposed method takes the weights into consideration thereby producing a better result. The reason that we stressed on using this method is that when calculating what the rating of the group is for a particular movie, we could take the individual aspects into consideration rather than just have a simple average. This ensures that each group was well represented in the prediction and so the recommendation that was finally given considered each member of a group when giving the prediction.

3.5.5 Evaluation Metrics

The evaluation measures that were used for the data include Precision, Recall, Normalized Root Mean Squared Error (NRMSE) and Normalized Discounted Cumulative Gain (NDCG). For precision and recall, we needed the system to classify the data as positive or negative. So we needed a method to classify the data. Here we followed the method used by Ortega and took a threshold for the rating. If the rating for a given movie in the test set was above the threshold, then it must be above the threshold in the predicted ratings as well. If so, then this rating was considered as positive.

Otherwise, the rating was considered as negative. This was the basis via which we calculated Precision and recall. We considered the threshold to be 4. So movies in the testing data that have been given a rating of 4 or above by a user in the given group were taken as the true positives. Any other movie rated by a user of interest is a true negative. We then compared this value with the predicted ratings to get precision and recall. Since the dataset is somewhat sparse we have taken groups into consideration even if there were movies that only a few members have watched. This should not be confused with the initial condition that the group must have 50 movies in common. There were at least 50 common movies but we tested with the entire dataset. Therefore, there will be cases where some of the members have not seen the movie being considered but due to the sparsity, they were still considered. The rating of the group was compared to the ratings of the users from the group who have watched the movie in order to test. Though this may seem odd, this is an efficient way to tackle the problem of data sparsity. Now let us look at the actual calculation of precision and recall.

Precision and recall were calculated as:

$$\text{precision}_G = \#TP_G / \#\,(TP_G \cup FP_G) \tag{4}$$

$$\text{recall}_G = \#TP_G / \#T_G \tag{5}$$

where $\#TP_G$ denoted number of True positives, $\#FP_G$ denoted number of false positives and $\#T_G$ denoted the number of expected recommendations.

We also used the NRMSE or normalized root mean squared error measure. This was given by the formula:

$$NRMSE = \sqrt{(1/N\,(\sum (p_r - a_r / max - min)^2)} \tag{6}$$

where p_r was the predicted rating for the movie, a_r was the actual rating, max was the max possible rating, min was the minimum possible rating and the summation was done overall movies (size N). The lower the NRMSE, the better was the performance.

Lastly we also used the NDCG or Normalised Discounted Cumulative Gain. This was given by the formula:

$$NDCG = DCG/IDCG \tag{7}$$

$$DCG = \sum (p_r - 1)/\log_2(i+1) \tag{8}$$

where p_r was the predicted rating and i was the current movie number or rank. IDCG (Ideal Discounted Cumulative Gain) was calculated in the same way as DCG except that we put the highest possible rating for the current movie in the denominator. So we needed to first calculate DCG as given in Eq. 3.8 and then calculate IDCG in a similar manner as described above. In Eq, 3.8 i may represent either the movie number or the rank of the movies. It does not matter which method you use as long as you use the same one for calculating both DCG and IDCG. This will ensure that the method is still correct. We took it to be the movie number and used it to calculate both DCG and IDCG. The higher the NDCG, the better the performance.

The aforementioned three metrics were picked as they are some of the most common ones used in measuring recommendation accuracy for a recommender system.

3.5.6 Steps

The input data was first put through multiple iterations of matrix factorization to obtain the factors and bias for the users as well as the movies. For each iteration, there was also an evaluation of the error in the training set as well as the testing set values in comparison to the original values. The more iterations we did, the more accurate the factors and biases were. Once the factors and biases were calculated, we divided the users in the dataset into groups and then represented all of the group's factors not as an average of the users' factors but as a set of the users' factors. Once this was done we predicted the movie rating using the proposed and traditional methods. A flowchart explaining the working of the proposed method is shown in Fig.2.

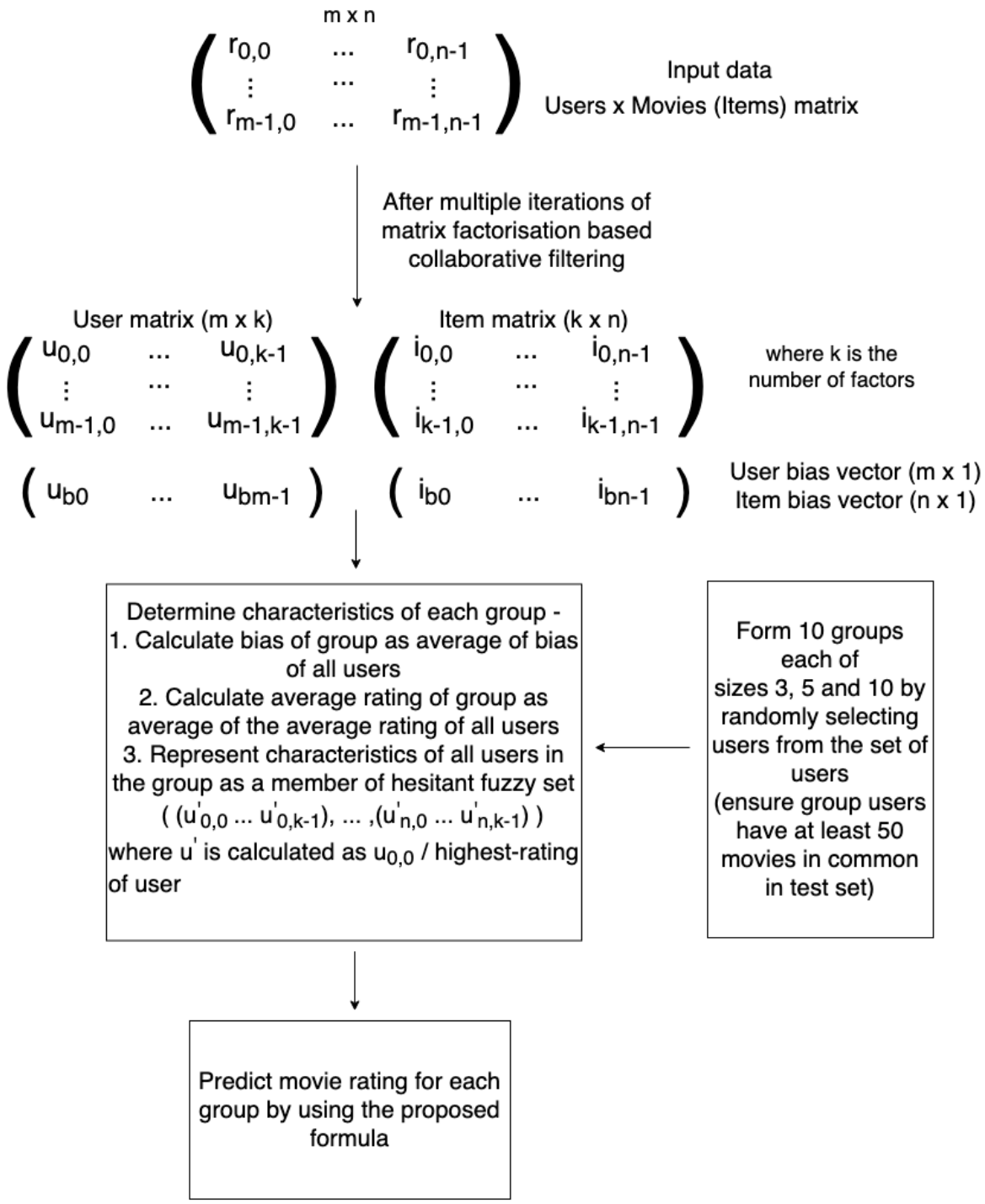

Fig.2: Architecture diagram

4.0 EXPERIMENT AND RESULTS

4.1 Experiment

The proposed method was evaluated for 2 datasets from the MovieLens database namely the 100K dataset and the 1M dataset [13]. The given dataset in each case was divided into training and testing data. 20% of the dataset was randomly chosen and made as to the testing data while the remaining 80% was used as the training data. The datasets had certain features which are as follows:

a. Each user rates a movie between 1 and 5 (1 being the lowest/bad and 5 being the best/good)

b. Every user has rated a minimum of 20 movies
c. A 0 rating is given if the user has not watched the movie

The proposed model was used on these 2 datasets and the results are mentioned in the next section. The 100K dataset was basically 100K items when considering 1000 users across 1700 movies. The 1M dataset was made with 1M entries while considering 6000 users across 4000 movies. The results from NRMSE and NDCG were compared with that of the traditional MF based CF method's results for the same data.

4.2 Trends and results

4.2.1 Precision and Recall

We calculated precision and recall to verify if our method is correct and if it gives the correct response. These metrics have the ability to prove the rigour of the method and establish its authenticity.

For calculating the precision and recall, we considered a threshold of 4 and said that any rating above this is a positive rating while any rating below this is a negative rating. So the precision and recall we observed are as shown in the table in Table 1.

Table 1: Precision and Recall of 100K dataset

	Small Groups (3 users)	Medium Groups (5 users)	Large Groups (10 users)
Precision	0.868	0.872	0.871
Recall	0.150	0.159	0.166

So as you can see the precision value is quite high which implies that our model predicted all of the positives correctly i.e. all the positives it predicted were more or less true positives. This is a desirable trait as it means our model is working properly. The recall levels however are very low. This may be attributed to the fact that the dataset as mentioned before is very sparse. So all the zeros are also considered as negatives and this may actually lead to a very low recall rate. Since we do not have the data for a lot of values we cannot predict the rating for those values. However, our method assumed these to be negative outcomes rather than outcomes that should not be considered. So we see lower recall values. Using a dataset that is less sparse might help with this. The precision and recall values for the 1M dataset are given in Table 2.

Table 2: Precision and Recall of 1M dataset

	Small Groups (3 users)	Medium Groups (5 users)	Large Groups (10 users)
Precision	0.866	0.869	0.867
Recall	0.133	0.135	0.138

For the 1M matrix, we observed from the table that the precision values are almost the same as the values for 100K but the recall values have a considerable change. We can analyse this further to understand why the trends are such.

Consider the 1M dataset -

6000 users across 4000 movies = 24000000 total possible entries
1 million entries are non-zero
Empty or zero entries = 24000000 - 1000000 = 23000000
Ratio of empty entries to non empty ones = 23000000 / 1000000 = 23

Now consider 100K -

1000 users across 1700 movies = 1700000 total possible entries
100K entries are non zero
Zero entries = 1700000 - 100000 = 1600000
Ratio of empty entries to non empty entries = 1600000 / 100000 = 16

So the ratio of empty to non empty entries is significantly lower for the 100K than the 1M dataset. This means that there is more sparsity in the 1M than the 100K thereby more false negatives as per our assumption. Due to this reason, the recall of the 1M is significantly lower than the 100K even though the precision is more or less the same. A graph showing the difference between the precision and recall for the 2 datasets is shown in Figure 2. It is important for us to analyse both the datasets here and in other result evaluations as it shows us the difference in the functioning of the algorithm based on the size of the data. We need to know if the algorithm will scale well and will perform equally as well for different sizes and so it is necessary for us to evaluate with 2 datasets and analyse the data from both.

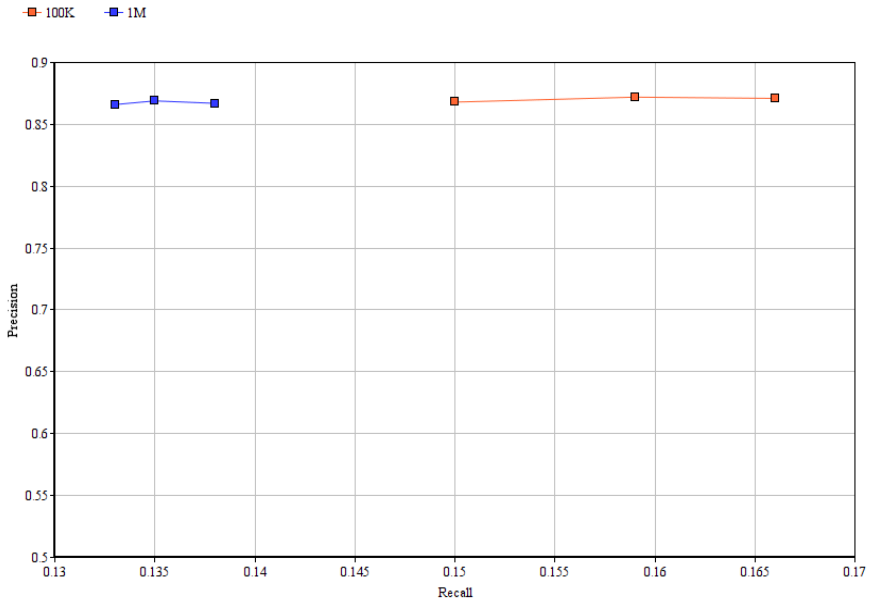

Fig.3: Precision vs Recall for both datasets

4.2.2 NRMSE (Normalised Root Mean Squared Error)

We observe the following with respect to NRMSE as shown in Table 3.

Table 3: NRMSE for 1M dataset

	Small Groups (3 users)	Medium Groups (5 users)	Large Groups (10 users)
NRMSE (traditional MF)	0.569	0.761	0.767
NRMSE (proposed method)	0.548	0.758	0.762

As you can see the proposed method performs slightly better than the traditional MF approach. This proves that our method gives a comparatively smaller error and hence is a better method for group movie recommendation for these

conditions. We obtain a lesser error via our method because we get a more accurate rating by predicting using an HFS rather than using mean values. So, this is mainly attributed to the fact that we are not taking a simple average as this would lead to a loss in values. It is a slight improvement on the traditional Matrix Factorization method as we see a lower amount of error. Similarly, the data for 100K can be found below in Table 4.

We can see that the results are similar for the 100K as well in the sense that it slightly outperforms the existing or traditional MF based CF method. However, it heavily outperforms the traditional method for large groups. This is probably because there is not enough data for this method to completely work as intended. This is the same reason why 1M results are better than 100K for the proposed method as there are considerably higher amounts of data to work with when taking the 1M dataset. We see that the loss is considerably lower for 100K than 1M which is an anomaly.

So when comparing the 2 datasets used by us, we can say that 1M performs better than 100K. This might be because the iterative process of factorisation was more accurate for the 1M because there are more values to choose from and hence a more accurate factorisation was possible. However, as mentioned in the previous paragraph, there exists an anomaly for the 100K dataset in the case of large groups. This might be because we are taking a large group into consideration and the data is pretty small and so there is overfitting that happens during the matrix factorization which causes the results to be this way for large groups. The presence of overfitting is supposed to be battled by the use of L1 regularisation in our formula but for some reason, large groups in the 100K dataset seem to be showing the characteristics of an outlier.

Table 4: NRMSE for 100K dataset

	Small Groups (3 users)	Medium Groups (5 users)	Large Groups (10 users)
NRMSE (traditional MF)	0.620	0.791	0.807
NRMSE (proposed method)	0.584	0.772	0.200

The overall comparison of the NRMSE for both methods is best represented by the graph shown in Fig.4. We can now compare easily the traditional and proposed methods. We see that the traditional method has a greater error across all 3 groups, namely - small, medium and large. It is important to note the behaviour of the methods across all the different sizes of groups since we need to know if there is consistency in performance. So our method has lower error across all the cases.

Additionally, we also use a weighted average when we are calculating the movie rating. This is necessary to account for certain social factors. If a person has seen more movies then they are clear on what they want to watch and know exactly what they do and do not like. This might not be the case for someone who has not watched many movies. We take this into consideration when calculating the value of the predicted rating of a movie for a group. So, we take the weighted average where the weight is based upon how many movies the user has watched. This ensures that we take each user's individual interaction into consideration and also account for the social factors that were referred to in the previous line.

If we look at the graph again now, it points out that the proposed method is better than the traditional method involving MF for all cases. So we can say that the proposed methodology improves the method slightly and that the integrated approach indeed does reduce the error caused on part of the recommendation system. We also know why the traditional method does not do as well as the proposed method and also why there is such an anomaly in the rating.

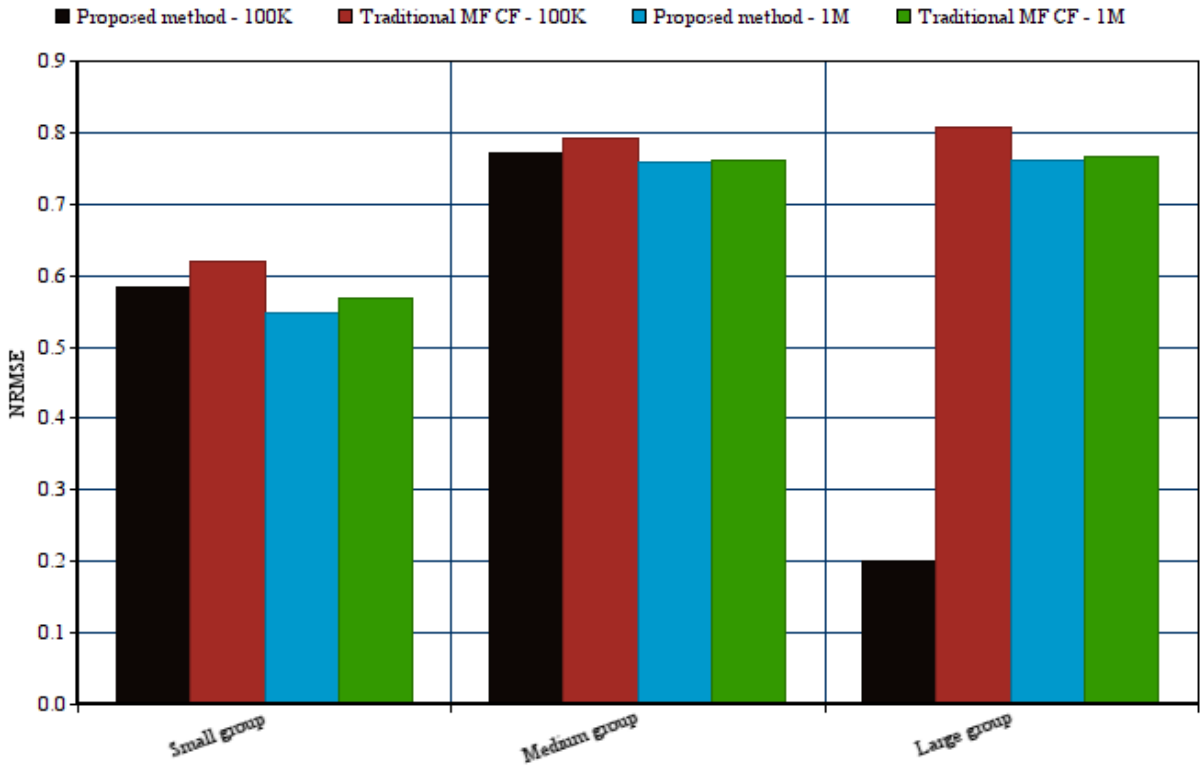

Fig.4: Comparison of NRMSE for both datasets across all group sizes

4.2.3 NDCG (Normalised Discounted Cumulative Gain)

If we look at the NDCG value, we see that it is indeed slightly higher for our method when compared to the traditional MF method. This once again shows that our method works better as we use the HFS concept rather than simple aggregation. So our model performs better and is more accurate. The values for the NDCG across 1M and 100K datasets are given in tables 5 and 6 respectively.

Table 5: NDCG for 1M dataset

	Small Groups (3 users)	Medium Groups (5 users)	Large Groups (10 users)
NDCG (traditional MF)	0.060	0.091	0.253
NDCG (proposed method)	0.062	0.091	0.260

Table 6: NDCG for 100K dataset

	Small Groups (3 users)	Medium Groups (5 users)	Large Groups (10 users)
NDCG (traditional MF)	0.043	0.080	0.136
NDCG (proposed method)	0.055	0.082	0.743

Again we see that the 1M dataset outperforms the 100K dataset. If you notice closely you will see that just like in the case of NRMSE, for the 100K dataset, the proposed model does way better than the traditional model in the large category. Also, there is once again an anomaly in the sense that the value for large groups in both the datasets has a huge variation just like in the previous case with NRMSE. This only goes on to further prove the anomaly and it shows that the model was overfitted. So the 100K has very low amounts of data and since we are trying to consider it for huge groups of users simultaneously, there is an anomaly.

All the NDCG values are shown in Fig.5 for ease of comparison. We can see the same trend as before which further substantiates the validity of the proposed model and gives proof that it does perform slightly better than the traditional method.

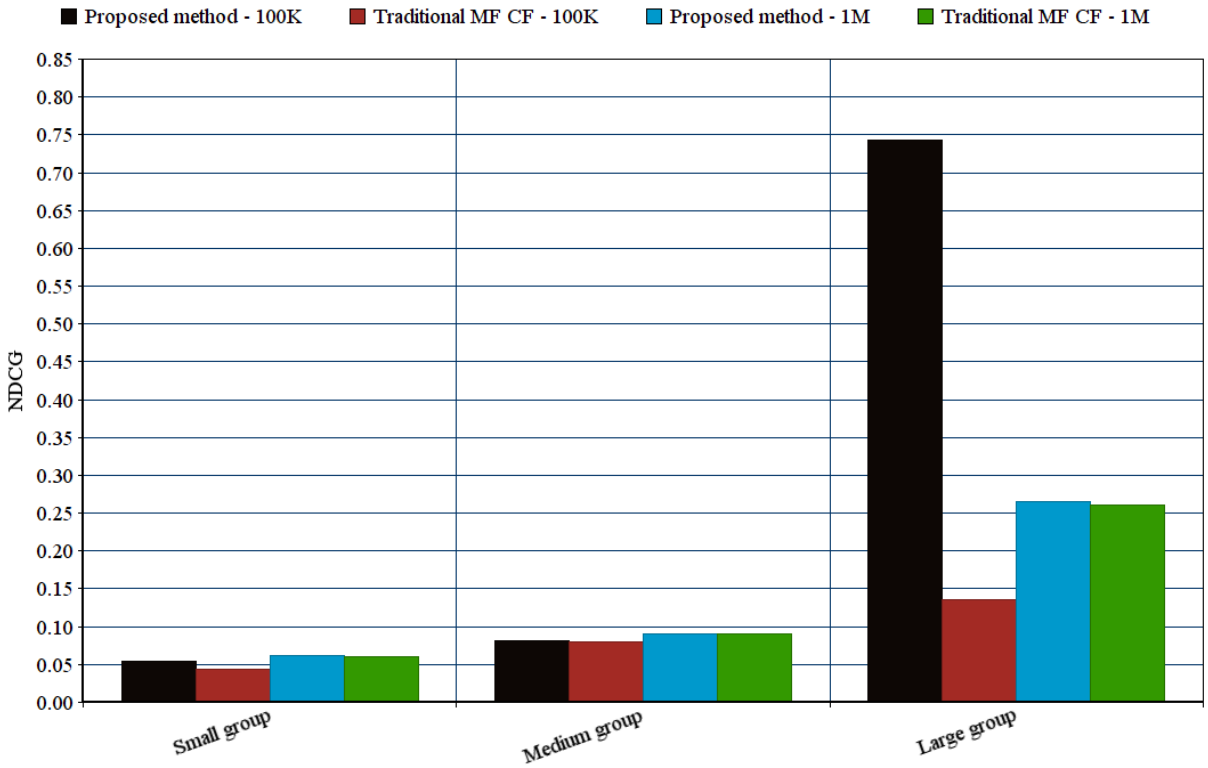

Fig.5: Comparison of NDCG for both datasets across all group sizes

As shown by figure 5, just like in the case of NRMSE, our method performs slightly better in terms of NDCG than the traditional method. The reasons for this are the same as the ones provided as part of NRMSE which is that our

method takes each users' traits into consideration rather than perform a simple average for obtaining group characteristics. We can say that our method is definitely better than the traditional MF CF method by a small amount.

5.0 CONCLUSION

The proposed method uses a combination of hesitant fuzzy sets and matrix factorization. Hence, it manages to slightly outperform traditional matrix factorization. It also performs better than the normal HFS method. This is because it tries to avoid the aggregation process as much as possible(it does so by taking into account individual characteristics of people in the group and it also takes into consideration features that are specific to the group in general). The method takes into consideration individual users' factors in the prediction of the movie rating. However, when it comes to bias, which is common to users of the group, aggregation at any stage is possible.

We may conclude that our method slightly improves the existing system for group recommendation and provides a more accurate solution. This slight improvement can be important when it comes to a movie recommendation as we need to decide which movie a group is watching based only on whether everyone will like the movie. So we need movie recommendations to be as accurate as possible. We have conclusive proof of the validity of the proposed model based on the numbers in the trends and results section.

The proposed method aside from taking individual characteristics into consideration as mentioned above, also accounts for a certain social factor by taking into account the fact that users who have watched a lot of movies will definitely know which movies, genres they do and do not like. A group may be a mixture of users who have watched a lot of movies and users who have not. The method takes a weighted average of the values during movie rating prediction in order to account for this behaviour.

This is again not the best solution as we can definitely improve this method further. But given the assumptions and conditions we have now, this method seems to perform well and is slightly better than others as we have shown.

This work can be improved upon. Firstly, the method is very slow for large amounts of data. It can be modified to use a different kind of data structure or set in order to make the calculations fast and light. Secondly, we have only considered one of the kinds of the grouping which is after factorisation. There are multiple other methods of grouping and forming characteristics before the factorisation is even done. Third, there is an anomaly in the large group case for the 100K dataset. Whether this was due to overfitting or an anomaly in general must be investigated.

ACKNOWLEDGEMENT

Authors thank the Department of Science and Technology, Science and Engineering Research Board for their financial support under the MATRICS Scheme (MTR/2019/000542). Authors also express their gratitude to SASTRA Deemed University for the infrastructure facilities and support provided to conduct the research.

REFERENCES

[1] G. Adomavicius & A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of the-art and possible extensions", *IEEE Trans. on Knowl. and Data Eng*, 2005, pp: 734-749.

[2] Deepa Anand, "Group Movie Recommendations via Content Based Feature Preferences", International *Journal of Scientific & Engineering Research*, 2013, Volume 4, Issue 2.

[3] S. Natarajan, S. Vairavasundaram, & L. Ravi, "Optimized fuzzy-based group recommendation with parallel computation", *Journal of Intelligent & Fuzzy Systems*, 2019.

[4] N. S. Selvan, S. Vairavasundaram, & L. Ravi, "Fuzzy ontology-based personalized recommendation for internet of medical things with linked open data", *Journal of Intelligent & Fuzzy Systems*, 2019.

[5] F. Ortega, A. Hernando, J. Bobadilla & J.H. Kang, "Recommending Items to Group of Users using Matrix Factorization based Collaborative Filtering", Information *Sciences, Elsevier*, 2016, pp: 345.

[6] Castro, M. Barranco, J. Rodríguez, & L. Martínez, "Group Recommendations Based on Hesitant Fuzzy Sets", *International Journal of Intelligent Systems*, 2018, Issue 33, pp: 2058-2077.

[7] R. Logesh, V. Subramaniyaswamy, V. Vijayakumar, X. Z. Gao, & G. G. Wang, "Hybrid bio-inspired user clustering for the generation of diversified recommendations", *Neural Computing and Applications*, pp: 1-20.

[8] L. Ravi, M. Devarajan, G. Jeon, O. Bayat, & V. Subramaniyaswamy, "An intelligent fuzzy-induced recommender system for cloud-based cultural communities", *International Journal of Web Based Communities*, 2019, Volume 15, Issue 3, pp: 271-288.

[9] P. Phorasim & L. Yu, Lasheng, "Movies recommendation system using collaborative filtering and k-means", *International Journal of Advanced Computer Research*, 2017, Issue 7, pp: 52-59.

[10] F. Ricci, L. Rokach & B. Shapira, "Recommender systems handbook", *Berlin, Springer*, 2011.

[11] R. Logesh, V. Subramaniyaswamy, V. Vijayakumar, & X. Li, "Efficient user profiling based intelligent travel recommender system for individual and group of users", *Mobile Networks and Applications*, 2019, Volume 24, Issue 3, pp: 1018-1033.

[12] JL. Herlocker, JA. Konstan, A. Borchers & J. Riedl, "An algorithmic framework for performing collaborative filtering", *Proceedings: 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1999, pp: 230–237.

[13] "Movielens dataset", *http://grouplens.org/datasets/movielens/*, 2000.

[14] V.R. Kagita, A.K. Pujari & V. Padmanabhan, "Virtual user approach for group recommender systems using precedence relations", *Information Sciences, Elsevier*, 2015, Issue 294, pp: 15–30.

[15] L. Ravi, V. Subramaniyaswamy, V. Vijayakumar, S. Chen, A. Karmel, & M. Devarajan, "Hybrid Location-based Recommender System for Mobility and Travel Planning", *Mobile Networks and Applications*, 2019, pp: 1-14.

[16] R. Logesh, & V. Subramaniyaswamy, "Exploring hybrid recommender systems for personalized travel applications", *In Cognitive informatics and soft computing*, 2019, pp: 535-544, Springer, Singapore.

[17] Y. Koren, R. Bell & C. Volinsky, "Matrix factorization techniques for recommender systems", *IEEE, Computer Journal* 42(8), 2009, pp: 30-37.

[18] G. Lekakos & P. Caravelas, "A Hybrid approach for movie recommendations", *Multimedia Tools Applications*, 2015, Issue 36, pp: 55-70.

[19] D. Jannach, M. Zanker, A. Felfernig, G. Friedrich, "Recommender systems: an introduction", *Cambridge University Press*, 2010.

[20] V. Subramaniyaswamy, M. V. Vaibhav, R. V. Prasad, & R. Logesh, "Predicting movie box office success using multiple regression and SVM", In *2017 international conference on intelligent sustainable systems* (*ICISS*), 2017, pp: 182-186, IEEE.

[21] X. Pham, J. Jung, V. Le Anh & S. Park, "Exploiting Social Contexts for Movie Recommendation", *Malaysian Journal of Computer Science*, Vol. 27(1), 2014

[22] V. Subramaniyaswamy, R. Logesh, M. Chandrashekhar, A. Challa, & V. Vijayakumar, "A personalised movie recommendation system based on collaborative filtering", *International Journal of High Performance Computing and Networking*, 2017, Volume 10, Issue 1-2, pp: 54-63.

[23] V. Balakrishnan & H. Arabi. (2018), "HyPeRM: A hybrid personality-aware recommender for movie", *Malaysian Journal of Computer Science*, Vol. 31, pp: 48-62, 2018

[24] L. Ravi, & S. Vairavasundaram, "A collaborative location based travel recommendation system through enhanced rating prediction for the group of users", *Computational intelligence and neuroscience*, 2016, 7