Evolving Good Recommendations

Supiya Ujjin **Peter J. Bentley**

Department of Computer Science, University College London
Gower Street, London S.Ujjin@cs.ucl.ac.uk
+44 (0)20 7679 2878

1 SYSTEM OVERVIEW

Recommender systems are new types of internet-based software tools, designed to help users find their way through today's complex on-line shops and entertainment websites. This paper focuses on the use of evolutionary search to fine-tune a profile-matching algorithm within a recommender system, to find profiles similar to the current user (or *active user*, *A*). Selected data from those profiles are then used to build recommendations. By evolving profile-matching, we tailor it to the preferences of individual users. This enables the recommender system to make more accurate predictions of users' likes and dislikes, and hence better recommendations to users.

In this research, the MovieLens dataset was used for initial experiments. The evolutionary recommender system uses 22 features from this data set: movie rating, age, gender, occupation and 18 movie genre frequencies.

Before recommendations can be made, the movie data is processed into separate profiles, one for each person, defining that person's movie preferences. We define *profile(j,i)* to mean the profile for user *j* on movie item *i*, see fig. 1. The profile of *j*, *profile(j)* is therefore a collection of *profile(j,i)* for all the items *i* that *j* has seen.

1	2	3	4	..22
Rating	Age	Gender	Occupation	18 Genre frequencies
5	23	0	45	000000100010000000

Figure 1: *profile*(*j*,*i*) - profile for user *j* with rating on movie item *i*, if *i* has a rating of 5.

Once profiles are built, the process of recommendation can begin. Given an active user *A*, a set or neighbourhood of profiles similar to *profile(A)* must be found.

The neighbourhood selection algorithm consists of three main tasks: *profile selection*, *profile matching* and *best profile collection*. It is not always feasible to use the entire database of profiles to select the best possible profiles. As a result, most systems opt for random sampling – performed by *profile selection*. Next, the profile matching process computes the distance or similarity between the selected profiles and the active user's profile using a modified Euclidean distance function (employing multiple features such as user's age, gender and movie genres). Every user places a different importance or priority on each feature. Our approach shows how weights defining user's priorities can be evolved by a genetic algorithm. Once all the Euclidean distances have been found, the *best profile collection* picks users most similar to *A* to form the neighbourhood of *A*. Because the neighbourhood set contains those users who are most similar to *A*, movies that these users like have a reasonable probability of being liked by *A*.

To calculate a fitness measure for an evolved set of weights for the active user, *w*(*A*), the recommender system finds a set of neighbourhood profiles for *A*. Three movie items that *A* has seen are then selected, where items with more ratings have a higher probability of being picked. The ratings of these users are then employed to compute the predicted rating for *A* on each movie item. The predicted vote, *predict_vote*(*A*,*i*), for *A* on item *i*, can be defined as:

$$predict\_vote(A,i) = mean_A + k \sum_{j=1}^{n} euclidean(A,j)(vote(j,i) - mean_j)$$

where: $mean_j$ is the mean vote for user *j, k* is a normalising factor such that the sum of the euclidean distances is equal to 1, *vote*(*j*,*i*) is the actual vote that user *j* has given on item *i, n* is the size of the neighbourhood.

Because *A* has already rated the movie items, it is possible to compare the actual rating with the predicted rating. So, the average of the differences between the three actual and predicted votes are used as fitness score to guide future generations of weight evolution.

2 EXPERIMENTS

Four sets of experiments were designed to observe the difference in performance between the evolutionary recommender system and a standard, non-adaptive recommender system based on the Pearson algorithm.

In all experiments, the GA recommender performed equally well (or better) than the Pearson algorithm. The results also suggested that random sampling is a good choice for the *profile selection* task of retrieving profiles from the database.

3 CONCLUSIONS

In conclusion, experiments demonstrated that, compared to a non-adaptive approach, the evolutionary recommender system was able to successfully fine-tune the profile matching algorithm. This enabled the recommender system to make more accurate predictions, and hence better recommendations to users.[1]

---

[1] Full details of this work can be found on-line: http://www.cs.ucl.ac.uk/staff/S.Ujjin/