Recommender System

Zijun Xue, Wen Shi and Xu Wu

March 17, 2014

Abstract

hello world

1 Review of Background

Recommendation systems are a subclass of information filtering system that seek to predict the 'rating' or 'preference' that user would give to an item. Recommender systems have become extremely common in recent years, and are applied in a variety of areas, such as movies, music, news, books, research articles, search queries, social tags, so and so forth[1] ($http://en.wikipedia.org/wiki/Recommender_system$). The recommendation systems are usually tailored to a specific user and are highly personalized. Recommendation systems typically produce a list of recommendations using different models (as listed below) and these approaches are often combined (Hybrid).

1. Collaborative: This system generates recommendations based on similarity measures between users and/or items. The items recommended to a user are those preferred by the similar users [http://josquin.cs.depaul.edu/ rburke/pubs/burke-umuai02.pdf].
2. Content-Based: Content-based filtering methods are based on a description of the item and a profile of the user?s preference. In a content-based recommender system, keywords are used to describe the items; beside, a user profile is built to indicate the type of item this user likes. In other words, these algorithms try to recommend items that are similar to those that a user liked in the past [Peter, Brusilovsky (2007). The Adaptive Web. p.325. ISBN978-3-540-72078-2.]
3. Knowledge-Based: A knowledge-based recommender suggests products based on inferences about a user?s needs and preferences. This knowledge will sometimes contain explicit functional knowledge about how certain product features meet user needs.
4. Demographic: A demographic recommender provides recommendations based on a demographic profile of the user. Recommended products can be produced for different demographic niches, by combining the ratings of users in those niche
5. Hybrid: Hybrid recommender systems combine two or more recommendation techniques to gain better performance with fewer of the drawbacks of any individual one. Most commonly, collaborative filtering is combined with some other technique in an attempt to avoid the ramp-up problem [http://josquin.cs.depaul.edu/ rburke/pubs/burke-umuai02.pdf].In the following are some of the combination methods that have been employed.
   (a) Weighted: The score of different recommendation components are combined numerically.
   (b) Switching: The system chooses among recommendation components and applies the selected one.
   (c) Mixed: Recommendations from different recommenders are presented together.
   (d) Feature Combination: Features derived from different knowledge sources are combined together and given to a single recommendation algorithm.
   (e) Feature Augmentation: One recommendation technique is used to compute a feature or set of features, which is then part of the input to the next technique.
   (f) Cascade: Recommenders are given strict priority, with the lower priority ones breaking ties in the scoring of the higher ones.

(g) Meta-level: One recommendation technique is applied and produces some sort of model, which is then the input used by the next technique.

In this project we mainly focused on the hybrid recommender system that combining collaborative and content-based systems.

2 Pros and Cons of the Existing Work

Collaborative filtering (CF) recommender system is a system that filters items through the opinions of similar people. It is an effective model and is widely used in many fields. However, it does have some drawbacks, one is usually referred as cold-start problem. For example, in the beginning stage, the users have little history information and the items have very few feedbacks. So it is a dilemma of where the system can start to work until it runs for enough time. Also, even after the system go through the cold-start period, it still faces balance between stability and plasticity. For example, some people?s interests change along with time. Someone who tends to watch science fictions two years ago might now grow more interests in watching romantic movies. In order to deal with that case the timestamp might be considered as a weight in deciding the recommendation candidates, i.e., the movies watched years ago may has less influence in the recommendation candidates that the system generates now. However, by doing so some long-term but discontinuous interests have been neglected, e.g., the romance movie Before Sunrise came up in 1995, with its sequels Before Sunset in 2004, and Before Midnight in 2013. Some fans of this series do not necessarily have interest in other romance movies, but watch every one of the three once they came up. Similarly long-term interests may be neglected and weakened if we simply undermine the weight of old items.
Content-based recommendation systems also have several drawbacks that make it a poor recommendation algorithm in some circumstances. The first one is that content-based techniques have a natural limit in the number and type of features that are associated, with the objects they recommend. No content-based recommendation system can provide suitable suggestions if the analyzed content does not contain enough information to discriminate items the user likes from items the user does not like. Another shortcoming for this system is called serendipity for the limited degree of novelty. For example, when a user has only rated movies directed by Stanley Kubrick, she will be recommended just that kind of movies. A ?perfect? content-based technique would rarely find anything novel, limiting the range of applications for which it would be useful. [$http://www.ics.uci.edu/\ welling/teaching/CS77Bwinter12/handbook/ContentBasedRS.pdf$].
In order to avoid the drawbacks of different recommendation approaches and increase the recommendation accuracy, the hybrid model that combines two or more approaches together are often implemented in modern recommendation systems. A hybrid approach that combine collaborative filtering and content-based filtering are the most widely used hybrid approach and could be more effective in some cases. Such hybrid approaches can be implemented in several ways: by making content-based and collaborative-based predictions separately and then combining them; by adding content-based capabilities to a collaborative-based approach (and vice versa); or by unifying the approaches into one model [$http://en.wikipedia.org/wiki/Recommender_system$]. Several studies empirically compare the performance of the hybrid with the pure collaborative and content-based methods and demonstrate that the hybrid methods can provide more accurate recommendations than pure approaches. These methods can also be used to overcome some of the common problems in recommender systems such as cold start and the sparsity problem.

3 Algorithms and Proposed Architecture

3.1 Memory-based Algorithms

Memory-based algorithms approach the collaborative filtering problem by using the entire database. Such algorithm classify users into similar groups, and based on such classification make predictions for the active user using others' preferences.

3.1.1 Similarity Measurements

The correlation between two users, with a value from -1 to 1, is used to describe the similarity of the two users. The more close the value is to 1, the more similar manners they share, and vice versa.
In our implementation we used two similarity measurements. One was the Pearson Correlation Coefficient. [description]
The second one is called *vector similarity*. Two users are treated as vector in n-dimensional space, where n is the number of items in the database. By calculating the angle formed by the two vectors we get a measurement of the similarity between the users. In practice we took a cosine of the angle value which has a result ranging from -1 to 1.

3.2 Model-based Recommendation System– Item-based Collaborative Filtering

Memory-based recommendation systems are not always that fast. Therefore, in the context of actual systems that generate real-time recommendations on the basis of very large datasets, *model-based recommendation systems* are used.
Model-based recommendation systems involve building a model based on the dataset of ratings. In short, we use some information extracted from the dataset, which will potentially offers the benefits of both speed and the scalability.

3.2.1 Similarities Between Items

The similarity between items(in this case, the movies) are measured by observing all the users who have rated both the movies.

3.2.2 Similarity Measurements

1. Cosine-based Similarity
   Also known as vector-based similarity, this formulation views two items and their ratings as vectors, and defines the similarity between them as the angle between these vectors,

$$sim(i,j) = cos(\overrightarrow{i}, \overrightarrow{j}) = \frac{\overrightarrow{i} \cdot \overrightarrow{j}}{||\overrightarrow{i}||^2||\overrightarrow{j}||^2} \tag{1}$$

2. Pearson(Correlation)-based Similarity
   This similarity measure is based on how much the ratings by common users for a pair of items deviate from average ratings for those items,

$$sim(i,j) = \frac{\sum_{u \in U}(R_{u,i} - \overline{R_i})(R_{u,j} - \overline{R_j})}{\sqrt{\sum_{u \in U}(R_{u,i} - \overline{R_i})^2 \sum_{u \in U}(R_{u,j} - \overline{R_j})^2}} \tag{2}$$

3. Adjusted Cosine Similarity
   This similarity measurement is a modified form of vector-based similarity where we take into the fact that different users have different ratings schemes; in other words, some users might rate items highly in general, and others might give items lower ratings as a preference. To remove this drawback from vector-based similarity, we subtract average ratings for each user from each user's rating for the pair of items in question,

$$sim(i,j) = \frac{\sum_{u \in U}(R_{u,i} - \overline{R_u})(R_{u,j} - \overline{R_u})}{\sqrt{\sum_{u \in U}(R_{u,i} - \overline{R_u})^2 \sum_{u \in U}(R_{u,j} - \overline{R_u})^2}} \tag{3}$$

3.3 Association Rules

3.3.1 Fundamentals of Association Rules

Association rules are used to describe the relationships between set of items. They usually follow the form like,

$$\{A_1, A_2, A_3, ...,\} \Rightarrow \{B_1, B_2, B_3, ...,\} \tag{4}$$

Association rules try to show how a series of items can determine another. For a specific example, if $AB \Rightarrow C$, we may say the appearance of both A and B in the history record will lead to the appearance of C. To describe how applicable a rule is we also introduce the term of *confidence* of a rule, with a range from 0 to 1. If the confidence of a rule is more close to 0, we may say the dependency relationship is more weak, otherwise we may say the dependency relationship is stronger. Especially, when the confidence equals to 1, the association rule describes exactly a functional dependency in the relations.

3.3.2 Association Rules for the Recommendations

For our purposes we use association rules of the form $A \Rightarrow B$. That is, if a single user watched the movie A, how likely would the user also watch the movie B?
First we create a square matrix $A[0, 1, 2..., n-1][0, 1, 2..., n-1]$of all these movie relationships describing how likely two movies will show together. $A[i][j]$ will indicate the confidence of the association rule between movie i and movie j. Also we treat user as a vector in n-dimensional space. The vector will contain elements of either 0 or 1 describing whether the user has watched the movie yet. By multiplying the matrix by the vector, we get what is called a *recommendation vector* – the most likely items that the user will watch.
Now we select the movies from the resulting vector and take out those movies having been watched by the user, we get use such a recommendation vector to order preferences of a user.

3.4 Hybrid System

The figure below elaborates how our recommendation systems work based on the algorithms mentioned above,

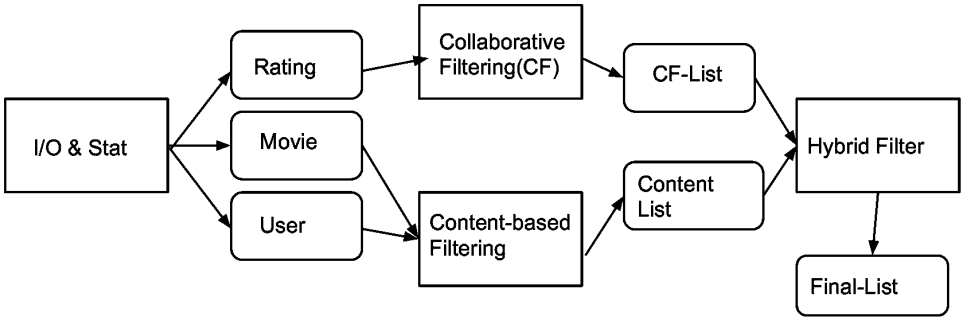

Figure 1: Algorithm Architecture

$$CF_{score} = \frac{common}{total} \cdot similarity \cdot \frac{(score - avg)}{\sqrt{var}} \cdot \frac{1}{recommendation_{count}[mid]} \tag{5}$$

$$Content_{score} = Overall_{rating} \cdot (UserPreference \cdot Genre) \tag{6}$$

$$F_{score} = \lambda \cdot CF_{score} + (1 - \lambda) \cdot Content_{score} \tag{7}$$

4 Evaluation

4.1 Beyond Accuracy

Typically, research on recommender systems is concerned about finding the most accurate recommendation algorithms. However, there is a number of factors that are also important. Diversity - Users tend to be more satisfied with recommendations when there is a higher intra-list diversity, i.e. items from e.g. different artists.[24]
Recommender Persistence - In some situations it is more effective to re-show recommendations,[25] or let users re-rate items,[26] than showing new items. There are several reasons for this. Users may ignore items when they are shown for the first time, for instance, because they had no time to inspect the recommendations carefully.
Privacy - Recommender systems usually have to deal with privacy concerns[27] because users have to reveal sensitive information. Building user profiles using collaborative filtering can be problematic from a privacy point of view. Many European countries have a strong culture of data privacy and every attempt to introduce any level of user profiling can result in a negative customer response. A number of privacy issues arose around the dataset offered by Netflix for the Netflix Prize competition. Although the data sets were anonymized in order to preserve customer privacy, in 2007, two researchers from the University of Texas were able to identify individual users by matching the data sets with film ratings on the Internet Movie Database.[28] As a result, in December 2009, an anonymous Netflix user sued Netflix in Doe v. Netflix, alleging that Netflix had violated U.S. fair trade laws and the Video Privacy Protection Act by releasing the datasets.[29] This led in part to the cancellation of a second Netflix Prize competition in 2010.[30] Much research has been conducted on ongoing privacy issues in this space. Ramakrishnan et al. have conducted an extensive overview of the trade-offs between personalization and privacy and found that the combination of weak ties (an unexpected connection that provides serendipitous recommendations) and other data sources can be used to uncover identities of users in an anonymized dataset.[31]
User Demographics - Beel et al. found that user demographics may influence how satisfied users are with recommendations.[32] In their paper they show that elderly users tend to be more interested in recommendations than younger users.
Robustness - When users can participate in the recommender system, the issue of fraud must be addressed.[33]
Serendipity - Serendipity is a measure "how surprising the recommendations are".[34] For instance, a recommender system that recommends milk to a customer in a grocery store, might be perfectly accurate but still it is not a good recommendation because it is an obvious item for the customer to buy. Trust - A recommender system is of little value for a user if the user does not trust the system.[35] Trust can be build by a recommender system by explaining how it generates recommendations, and why it recommends an item.
Labelling - User satisfaction with recommendations may be influenced by the labeling of the recommendations.[36] For instance, in the cited study click-through rate (CTR) for recommendations labeled as "Sponsored" were lower (CTR=5.93%) than CTR for identical recommendations labeled as "Organic" (CTR=8.86%). Interestingly, recommendations with no label performed best (CTR=9.87%) in that study.

References

est