Middle-East Journal of Scientific Research 24 (Techniques and Algorithms in Emerging Technologies): 450-461, 2016
ISSN 1990-9233;
© IDOSI Publications, 2016
DOI: 10.5829/idosi.mejsr.2016.24.TAET23614

A Comparative Analysis of Recommendation Systems

[1]S. Prabha and [2]K. Duraisamy

Associate Professor, K.S.Rangasamy College of Technology,
Tiruchengode, Tamilnadu, India
[2]Dean/Academic, K.S.Rangasamy College of Technology,
Tiruchengode, Tamilnadu, India

Abstract: Recommendation systems are extensively used on the internet to help customers in identifying the products or services that fits best with their individual preferences. While current implementations effectively reduce information overload by providing personalized suggestions when searching for objects such as books or movies. In this paper we analyze different approaches to develop recommendation systems. Recommendation system developed so far cannot be used in another potential field of application: the personalized search for subjects such as applicants in a recruitment scenario. Theory shows that a fine match between persons and jobs needs to consider both the preferences of the provider and the candidate. We present different approaches to distinct recommendation systems to the field in order to improve the match between people and jobs.

Key words: Content based algorithm · Collaborative filtering algorithm · Cold Start · Hybrid approach · Item based algorithm · Over Specialization · Sparsity · Recommender system · User based algorithm

INTRODUCTION

Recommender System or Recommendation system is a subclass of information filtering system that look for to predict "rating" or "preference" that user would give to an particular item. The recommender system was first developed by Goldberg, Nichols, Oki & Terry in 1992. Recommender system as defined by M.Deshpande and G. Karypis is a personalized information filtering technology used to predict whether a specific seeker will be interested in a particular item or to recognize a set of N items that will be of interest to a certain user. Normally, a recommender system compares a user profile [1] to some reference characteristics and seeks to predict the 'rating' or 'preference' that a user would give to an item they had not yet considered. Examples of recommender systems are amazon.com, Reel.com, CDNOW, eBay, Levis, Moviefinder.com. The main objective of recommender system is to predict ratings of the non-rated user/item combination and thus providing appropriate recommendations.

Background: The general concepts involved in recommender system.

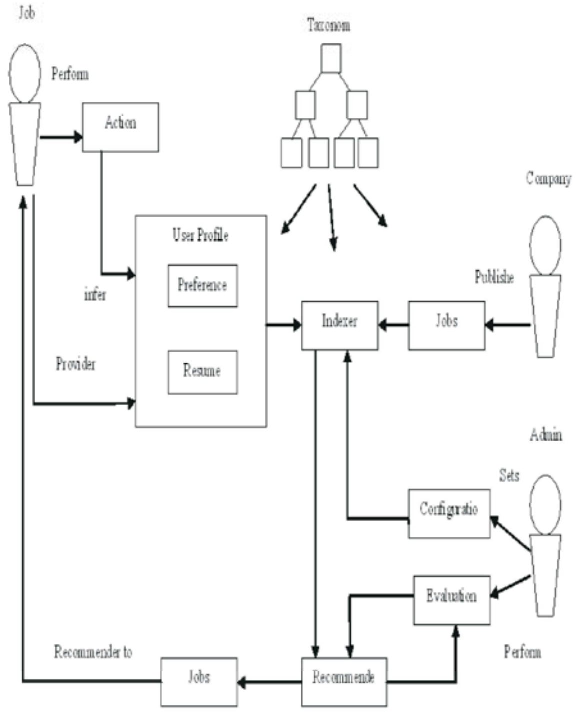


Fig. 1: Recommender System

The Functionality of each blocks are as follows:

Hybrid User Profile: The hybrid profile considers information from the resume and the relevance feedback into one depiction. For calculating recommendations, the user profile requires at least a few preferences or some data from the resume.

Indexer: The Indexer is responsible for transforming the various data structures into a common for the matching algorithm optimized representation. For that, different information for the user profile and for the jobs is obtained from the database and transformed into the proper representation. Furthermore, this component interacts with the taxonomy and adds derived concepts to the profiles and jobs.

Recommender: The Recommender[2] uses the information stored and does content-based queries for matching the user profile and the jobs. For regulating the process, the queries take several configuration parameters.

Configuration: An administrator is in charge for creating and managing settings of the recommender. This includes selecting the correct user actions for the relevance feedback, defining the mapping rules to different parameters for the recommender. All settings have to be available for the site administrator. A correct user interface will be offered in the implementation chapter.

Evaluation: Evaluation component is needed to test the performance of different parameter settings. A pre-defined set of relevant items is compared with results from the recommender with which performance measuring of the system like the precision, recall and f1score is measured.

Recommender systems typically construct a list of recommendations [3] using collaborative filtering approach or content-based filtering approach.

Table I: Techniques used in recommendation system

S.No	Methods and Algorithms Used	Results
1	Collaborative Filtering Approach using KNN algorithm with explicit feedback	Limitation of scalability and performance
2	Collaborative Filtering Approach with implicit feedback	Improves Scalability of Collaborative filtering
3	k-means clustering	Improves prediction accuracy
4	Longest Common Subsequence Algorithm	Improves quality of system for Recommendation
5	Formal Concept Analysis Approach	Provides personalization and recommendation
6	Model based Clustering Approach	Discovers user's interest in session
7	Integration of clustering, association rules and markov models	Web page prediction accuracy improved

Content Based Approach: Content based algorithm [4, 5] recommender systems work with profiles of users that are formed at the beginning. A profile contains information about a user and his taste. The taste is based on how user rates the items. In the recommendation process, the engine compares the items that were already positively rated by the user with the items he didn't rate and looks for similarities. Those items that are mostly similar to the positively rated ones, will be recommended to the user.

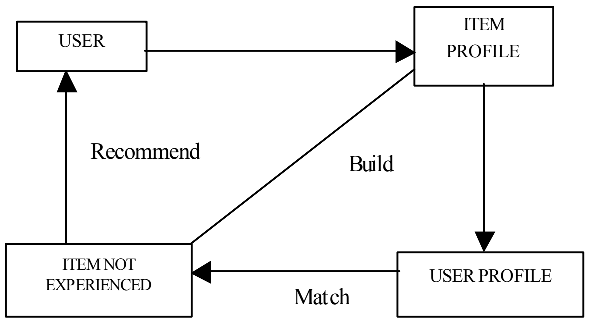


Fig. 2: Content – Based Process

- System has huge database consisting of the recommended item [6] and the items features which is the Item profile.
- The user gives some information about their preferences to the system. Adding that information to the item profile, the system creates the user profile.
- According to the information available in the user's profile, the system recommends most relevant items to the user.

The main objective of collecting user information is to generate a profile that describes user characteristics. The common techniques are explicit profiling, implicit profiling and use of legacy data:

Explicit Profiling: Each user is requested to fill in a form when visiting the web site which has the advantage of allowing users specify directly their interests.

Implicit Profiling: The user's behavior is identified automatically by the system and it is transparent to the user. Often, user registration is saved in cookie that is kept in the browser and updated at each visit. Behavior information is generally maintained in a log file.

Legacy Data: The Legacy data gives a rich source of profile information for recognized users.

Some drawbacks of this technique are as follows:

- *problem of New-user* -Same as for collaborative filtering, user profiles are required as input
- *Limited analysis of content* -The recommender highly depends on the information available from the documents. Therefore, the documents contain either some machine readable text or they need to be classified by users manually. The popularity of two documents, which has the same vectors, cannot be differentiated by such content-based systems
- *Over specialization*-The recommender recommends the items similar to the ones and the one which is already known, leading to a portfolio effect

Collaborative Filtering Approach: Collaborative filtering [7] Algorithm recommender system became one of the most researched techniques of recommender systems [8]. The idea of collaborative filtering is in finding users in a community that share appreciations. If two users have similar or almost similar rated items in common, then they have same tastes. These users form a group called neighborhood. A user receives recommendations to those items hasn't rated before, but that were already positively rated by users in their neighborhood. Some of the different methods in collaborative filtering are Used based approach, Item based approach and Hybrid recommendation approach.

Workflow of Collaborative Filtering:

1) Expressing a User's his/her preference by rating the items.
2) Finds the people with most similar taste by matching their rating with other users rating.
3) Finally, the most highly rated by users are recommended by the system.

The advantage of CF approach is that it will not consider the content of item being recommended rather it matches user to the item based on content attributes and their drawbacks are dependency on human ratings.

There are several kinds of collaborative filtering approach:

1. User-based algorithm
2. Item-based algorithm
    1. User based approach

User based algorithm is also known as memory based algorithm. In the user-based algorithm, the users do the main role. If majority of the users has the similar taste then they form into one group. Recommendations are provided to user, based on evaluation of items by other users from the same group, with they share common preferences. If the item was positively rated by the community, it will be recommended to the user.

Item Based Approach: Item based algorithm is also known as model based algorithm. Considering to the fact that the taste of users remains same or change very slightly similar items form neighborhoods' based on appreciations of users. Afterwards the system generates recommendations that a user would prefer with items in the neighborhoods which mainly depends on relationship between Items.

Item-based algorithms have two steps. In the first step, the algorithms scan the past information of the users and the ratings they gave to items are collected. Similarities between items are built and inserted into an item-to-item matrix M1 from these ratings. The element yij of the matrix M1 represents the similarity between the items in row i and the item in column j. In the second step, the algorithm selects items that are most relevant to the particular item's user rating. Deshpande and Karypis give a method to construct M1 (Algorithm 1) after computing the similarities between the items. For each item j, the algorithm computes the similarity

```
Algorithm
    for i = 1 to do
      for j = 1 to m do
        if i ¡≠ j then
          $Im_{i,j}$ = sim(R*i, R*,j )
        else
          $Im_{i,j}$ = 0
        end if
    end for
      for i =1 to m do
        if i ¡≠ among the n largest values in $IM_{i,j}$ then
          $Im_{i,j}$ = 0
        end if
      end for
    end for
```

between j and the other items and stores the results in the i^{th} column of M1 (line 1). After that it zeroes all the entries in M1 that less similarity than the n^{th} largest similarity. The second inner for-loop makes sure that an item not recommends itself.

Similarity in item based collaborative filtering can be computed by using two approaches: implicit or explicit.

Prediction Based on Explicit Ratings: In this approach user requires to specifically rate on items.

Let m be the total number of users in database, y_{ij} the set of users that have both rated item i and item j, the Pearson correlation coefficient of their related columns in the user-item matrix and is given by the following formula.

$$sim(i,j) = \frac{\sum_{h=1}^{m'}(R_{u_h,i} - \overline{R_i})(R_{u_h,j} - \overline{R_j})}{\sqrt{\sum_{h=1}^{m'}(R_{u_h,i} - \overline{R_i})^2}\sqrt{\sum_{h=1}^{m'}(R_{u_h,j} - \overline{R_j})^2}}$$

$R_{uh,i}$ is the explicit rating given by an user u_h to an item i. And $R_i^{'}$ is the average of the ratings given on item I.

Prediction Based on Implicit Ratings: The implicit user based algorithm, the ratings provided to items can be implicitly computed by considering the similarity between two items by using the Pearson correlation coefficient of their associated rows in the item-category bitmap matrix.

$$sim(i,j) = \frac{\sum_{h=1}^{p}(v_{c_h,i} - \overline{v_i})(v_{c_h,j} - \overline{v_j})}{\sqrt{\sum_{h=1}^{p}(v_{c_h,i} - \overline{v_i})^2}\sqrt{\sum_{h=1}^{p}(v_{c_h,j} - \overline{v_j})^2}}$$

p is the number of categories and $v_{ch,i}$ is a Boolean value that equals to 1 if the item i belongs to the category h or 0 otherwise.

Compared to the user-based algorithms, item-based algorithms sparse better and scale well. Their major drawback is the cost involved to build the item-to-item matrix M1. In order to construct M1, we need to compute the similarity between every pair of items. Once this is done, item-based algorithms perform more rapidly and scale better compared to user-based algorithms. In spite of their slowness, experiments analysis shows that user-based algorithm provides more accurate recommendation than item-based algorithms.

The choice of the algorithm is based on how much trade-off can be made among the prediction performance and the scalability.

Some drawbacks of this technique are as follows:

- *Problem of New-user* -For recommending items, the users must specify their preferences first without this information, no recommendations can be made
- *Problem of New-Item* -Items that are new in the system need ratings by users before they are being used in the recommendation process.

The above two problems are often referred as cold start or ramp-up problem

- *Ratings of Sparsity* -User often tends to rate similar items, leading to a sparsity of ratings, where only a small items have many evaluations. This makes it complex to make recommendations in all situations. If the critical mass isn't reached, demographic filtering can facilitate to categorize users on a different basis.
- *Problem of Gray sheep* -Especially users interested in rare items, are difficult to categorize.

Hybrid Recommendation Approach: Hybrid recommendation approach combines multiple recommendation techniques. Several researchers have attempted to combine collaborative filtering and content based approaches in order to reduce their disadvantages and increase the performance while recommendations. Depending on the domain and data, several hybridization techniques are possible by combining collaborative filtering and content based filtering technique. Hybridization techniques are:

- Combining individual predictions by implementing CF and CB separately and incorporating some content based characteristics into collaborative approach.
- Using some collaborative characteristics into content based approach.
- Generate a unified recommender system, which brings together both approaches.

Knowledge Based Approach: Knowledge based recommender system depends on domain knowledge and about the learners [9] knowledge. Extracting the knowledge of learner's and knowledge about the learning materials, is the major task in knowledge based recommender system. Knowledge-based recommender systems will not consider building long-term generalizations about their users but they prefer generating a recommendation based on matching between user's need, preferences and available set of items. This approach does not involve the problem of sparsity and also the over specialization because this approach is independent of another user and the statistical evidence. Furthermore, this approach is sensitive to change in the learner interest and preferences of learner and also it doesn't have any dependency on information rating. Knowledge-based approach need not have an initial database of learner's preference and also it is capable to exploit the knowledge about the learning domain to provide the best solution to the learners. However, its main objective is to generate the most relevant recommendations and reasoning about how learning materials of the domain meets the learner's need.

Recommendation Metrics: Items and users information are getting increased in systems where recommender use is important. Therefore, recommender system must ensure the accuracy, efficiency and scalability of the items recommended.

Accuracy: Accuracy is a most important recommendation metric. This is measured by the closeness of the result of a recommendation that matches a user's preference.

Efficiency: Recommender system must process the request within the reasonable time by making use of resource available and process hundreds of request per seconds. Memory utilization and Computation time are two vital metrics that calculate the efficiency of a recommender system.

Scalability: Good Recommender systems that process thousands of requests must handle hundred of thousand requests in the future.

The performance of a recommender system can be estimated by comparing recommendations to a test set of known user ratings. These systems are evaluated using predictive accuracy metrics, where the predicted ratings are directly compared to actual user ratings. The most frequently used metric is Mean Absolute Error (MAE) – defined as the average absolute difference between predicted ratings and actual ratings, give by:

$$MAE = \frac{\sum_{\{u,i\}} |p_{u,i} - r_{u,i}|}{N}$$

where pu, I is the predicted ratings for user u on item i, ru,I is the actual rating and N is the total number of ratings in the test set.

A related commonly-used metric, Root Mean Squared Error (RMSE), which emphasis on larger absolute errors and is given by

$$RMSE = \sqrt{\frac{\sum_{\{u,i\}} (p_{u,i} - r_{u,i})^2}{N}}$$

One of the best approach for maintaining accuracy, efficiency and scalability is to use hashing techniques. It compresses large data sets, very large number of users are scaled and obtain a good performance within the reasonable time.

Similarity Functions

Similarity Based on Query Contents: There are different ways to represents query contents using keywords, words in order and phrases.

Similarity Based on Keywords or Phrases: Keywords are the words, except for function words included in a stop-list. The keyword stemming is done using the Porter algorithm. The keyword-based similarity function is defined as follows:

$$similarity_{keyword}(p, q) = KN(p, q) / Max(kn(p), kn(q)) \quad (1)$$

where $k_n(.)$ is the number of query keywords, KN(p, q) is the number of common keywords in two queries.

If weighted query terms are used then the following modified formula can be used instead:

$$similarity_{w\text{-}keyword}(p, q) = \sum_{i=1}^{N} (w(k_i(p)) + w(k_i(q)))/Max(kn(p), kn(q)) * 2 \quad (2)$$

where *w* $(k_i(p))$ is the weight of the *i*-th common keyword in query *p*and *kn(.)*becomes the sum of weights of the keywords in a query. In our case, we use *tf***idf*for keyword weighting.

The above measures easily can be extended to phrases case. If phrases are identified in the queries, easily we can calculate the query similarity.

Similarity Based on String Matching: It uses all the query words among queries for similarity

Calculation including the stop words. Similarity may be calculated by the edit distance, which is a measure based on the number of edit operations like insertion, deletion, etc. necessary to unify two strings which is nothing but queries. The similarity is inversely proportional to edit distance:

$$similarity_{edit}(r, s) = 1 - EditDistance(r, s) \quad (3)$$

The advantage of this measure is that it considers the word order, as well as words that denote query types such as "who" and "what" if they occur in a query.

Similarity Based on User Feedback

Similarity Through Single Document: A feedback-based similarity measure considers each document seperately. This similarity is proportional to the shared number of clicked which is selected documents, taken individually, as follows:

$$similarity_{document} = RD(r,s)/Max(rd(r),rd(s)) \quad (4)$$

wherer d(.) is the number of clicked documents for a query, RD (s,s) is the number of document clicks in common.

Inspite of its simplicity, this measure demonstrates clustering of semantically related queries that contain different words. Below are some queries from one such cluster:

Query 1: Resume
Query 2: CV
Query 3: curriculum vita
Query 4: Database
Query 5: c
Query 6: oops/c++/small talk
Query 7: java
……

They all match to a particular document "ID: 7, Title: Resume". In addition, this measure is also very useful in analyzing between queries that has similar word but stem from different needs of information. For example, if one user asked for "law" and clicked on the articles about legal problems and another user asks the same "law" and clicked the articles about the order of nature, the two cases can be easily differentiated by the user clicks. This kind of distinction can be used for sense disambiguation in a user interface.

Similarity Through Document Hierarchy: The concept hierarchy allows us to extend the previous estimation by taking into account a conceptual distance between documents. This distance is calculaed as follows: the lower the common parent node if the two documents have the shorter conceptual distance between them. Let P(di, dj) denote the lowest common parent node for documents di and dj, L(x)the level of node x, L_is the total levels in the hierarchy (i.e. 4 for Encarta). The conceptual similarity between two documents is defined as follows:

$$r\,(di, dj) = (\,L(q(di, dj)) - 1) \,/\, (L\_Total - 1) \qquad (5)$$

In particular, $s(d_i, d_i)= 1$; and $s(d_i, d_j)= 0$ if P(di, dj)= root. Now let us incorporate this document similarity measure into thecalculation of query similarity.Let d_i $(1 \leq i \leq m)$and $d_j(1 \leq j \leq m)$be the clicked documents for queries pand qrespectively and rd(q) and rd(x)the number of document clicks for each query. Thehierarchy-based similarity is defined as follows:

$$similarity_{concept}(p, q) = \frac{\sum_{i=1}^{m} (\max_{j=1}^{n} s(d_i, d_j)) \Big/ rd(p) + \sum_{j=1}^{n} (\max_{i=1}^{m} s(d_i, d_j)) \Big/ rd(q)}{2} \qquad (6)$$

Using the above formula, the following two queries are recognized as being similar:

Query 1: <query text> image processing
<clicked document>ID: 2 Title: Graphics
Query 2: <query text> image rendering
<clicked documents>ID:7Title: Animation

Both documents have a common parent node "Computer Science". According to formula (5), the similarity between the two documents is 0.66. If these two documents are selected for the two queries, then the similarity between the queries is also 0.66 according to formula (6). In contrast, their similarity based on formula (4) using common clicks is 0. Hence, we see that this new similarity function can recognize a wider range of similar queries.

Recommender Technologies: For implementing a job recommender, four different and freely available technologies were taken into consideration, including Apache Mahout, easyrec, Drupal's Recommender API and Apache Solr.

Apache Mahout: Mahout is an open source Java library, which supports many Machine Learning algorithms, including a wide range of recommender techniques, like collaborative filtering and content-based recommenders. A main goal of this library is to provide a scalable solution, which can be achieved by distributing it via a Hadoop cluster.

Advantage:

- Supports many algorithms out of the box
- Scalable implementation
- Integrated evaluation component

Disadvantage:

- No integration for Drupal yet exists, even though it is planned to be added during a Google Summer of Code Project
- Difficult to realize a hybrid user profile, as it mainly focuses on user preferences

Easyrec: Easyrec is an open source Web application [10] that provides personalized recommendation using. It is another example for a recommender library, which is written in Java and available as open source project.

Advantage:

- Easy to use via a REST API
- Integration into Drupal 6 & 7 exists, although it hasn't been available when starting with the implementation.

Disadvantage:

- Rather designed for simple recommender use cases, like web shops
- No content-based algorithms are available at the moment
- As Mahout, based on user preferences and hence an integration of information out of a resume seems to be difficult.

Drupal's Recommender API: Drupal's Recommender API is a contributed PHP module, which can be easily installed on any Drupal website. Its algorithms are implemented in PHP and it mainly focuses on the web shop use case.

Advantage:

- Drupal module
- It's planned to use Apache Mahout in future.

Disadvantage:

- Drupal 6 only
- Implemented in PHP, which isn't fast nor scalable for such resource-intensive tasks
- Similar user model as Mahout and easyrec.

Apache SoIr: Apache Solr is an open source search framework, which is also written in Java. Internally it is based on Apache Lucene and provides a standalone server for it, which communicates with other applications via REST-like HTTP requests. It is mainly used for full text search and offers many additional features, like faceted searches and so. The use of vectors for the document representation makes it also useable as recommendation framework, which has been done on Absolventen.at so far.

Advantage:

- Drupal 7 integration already exists
- Fast and scalable implementation
- Possibility to extend Solr with own plugins for the current use case
- Lots of experiences with Solr are already available.

Disadvantage:

- Focus on fulltext searches
- Fixed vector-based implementation and due to that, other algorithms are not possible.
- Requires own plug-in for recommendations.

Online Job Recommender System: A JRS (Job Recommender System) consists of a job applicant subsystem which is designed for job applicants and an eRecruiting subsystem that is used by recruiters.Four well known online job recommender systems are.

- CASPER
- Proactive
- PROSPECT
- eRecruiter

Table II: The comparison of job applicant subsystem

System Elements	CASPER	Proactive	PROSPECT	eRecruiter
User Profile	Individual Information and behavior	Individual information	Individual information	Individual Information and behavior
Approach	CFR	CBR	CBR	CBR
	CBR	KBR		KBR
Layout	Comprehensive List	Modular List	Comprehensive List	Comprehensive List
User Behavior	Apply Collect	Apply	Lack of Website	Email

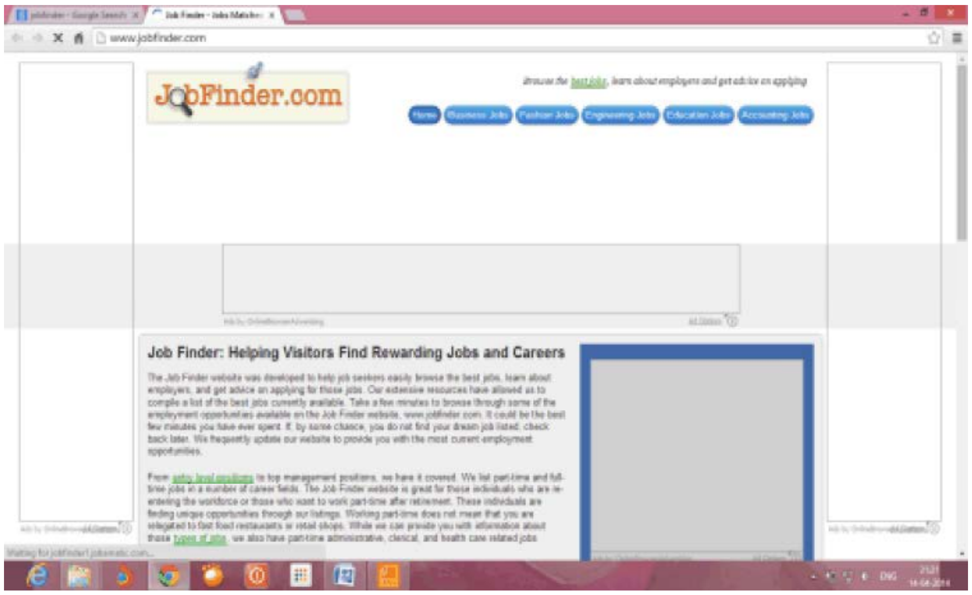
Casper: CASPER (Case-based Profiling for Electronic Recruitment) is a classical job applicant subsystem that is used for enhancing the performance of the JobFinder.

Advantage:

- Hybrid profile and approach
- feature importance can be set by the User
- Based on user feedback update profile

Disadvantage:

- Contentof profile is simple
- Using one way recommendation



Job Finder: Helping Visitors Find Rewarding Jobs and Careers

The Job Finder website was developed to help job seekers easily browse the best jobs, learn about employers, and get advice on applying for those jobs. Our extensive resources have allowed us to compile a list of the best jobs currently available. Take a few minutes to browse through some of the employment opportunities available on the Job Finder website, www.jobfinder.com. It could be the best few minutes you have ever spent. If, by some chance, you do not find your dream job listed, check back later. We frequently update our website to provide you with the most current employment opportunities.

From entry level positions to top management positions, we have it covered. We list part-time and full-time jobs in a number of career fields. The Job Finder website is great for those individuals who are re-entering the workforce or those who want to work part-time after retirement. These individuals are finding unique opportunities through our listings. Working part-time does not mean that you are relegated to fast food restaurants or retail shops. While we can provide you with information about these types of jobs, we also have part-time administrative, clerical, and health care related jobs

Fig. 3: Screenshot of CASPER online Recruiting Website

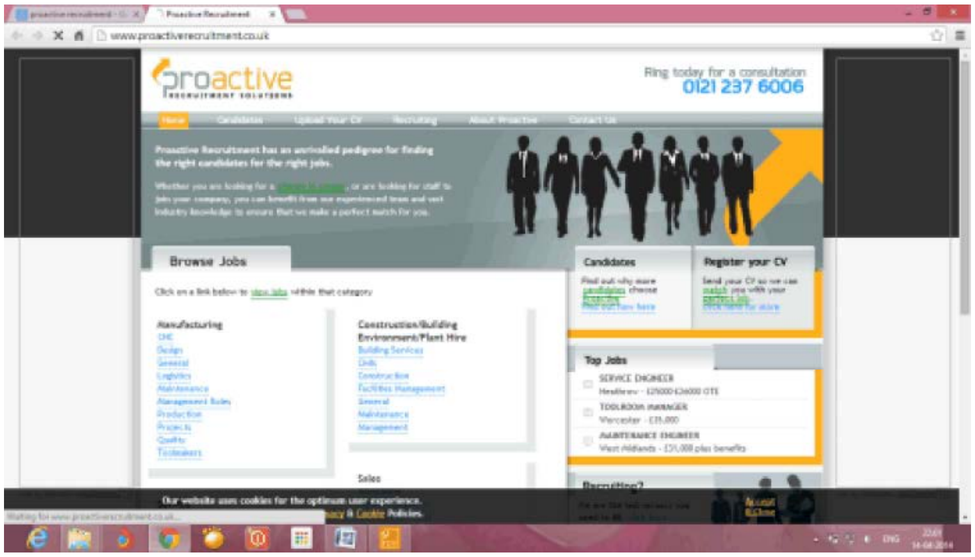
Proactive: The Proactive has different recommendation modulesapplied to its own website. Proactive confine the user preference based on the description of a preferred job.

Advantage:

- Hybrid approach
- Provide four recommendation modules
- Using ontology to classify jobs.

Disadvantage:

- Single profile
- Knowledge engineering problem
- Only email about user feedback.



Proactive Recruitment has an unrivalled pedigree for finding the right candidates for the right jobs.

Click on a link below to view jobs within that category

Manufacturing: CNC, Design, General, Logistics, Maintenance, Management Roles, Production, Projects, Quality, Toolmakers

Construction/Building Environment/Plant Hire: Building Services, Civils, Construction, Facilities Management, General, Maintenance, Management

Sales

SERVICE ENGINEER – Heathrow – £25000-£26000 OTE

TOOLROOM MANAGER – Worcester – £35,000

MAINTENANCE ENGINEER – West Midlands – £31,000 plus benefits

Our website uses cookies for the optimum user experience. Privacy & Cookie Policies.

Fig. 4: Screenshot of Proactive online Recruiting [11] Website

PROSPECT: PROSPECT is a resume miner for analyzing and mining the resume. It analyzes the resume to generate the user profile.

Advantage:

- Resume miner
- Batch processing.

Disadvantage:

- Single profile and approach
- Simple resume match
- Use one way recommendation.

eRecruiter: The eRecruiter is planned for increasing the functionality and improving the accurateness of the Absolventen.at. Similar to PROSPECT, It also analyzes the resume to generate the user profile.

Advantage:

- Hybrid profile and approach
- Use ontology to classify jobs and users.

Disadvantage:

- Single method of calculating similarity
- Use one way recommendation.

Fig. 5: Screenshot of eRecruiter online Recruiting Website

Application

Recommender system in E-Commerce: Amazon.com is an e-commerce [12] website in which users can buy books, music and others goods. It has databases containing more than 29 million customers and several million catalogue items. Amazon.com uses algorithm based on item-based collaborative filtering to make their recommendations. Item based filtering scales independently the number of users and the number of items. The item-to-item collaborative filtering, works by first matching [13] each of the users' purchased and rated items to similar items. Then, it will combine similar items with recommendation list.

Music recommender system in iTunes: The Music Recommender System for iTunes is one of the most famous recommender systems. It is software for iTunes, which is used for the integrated rating system, not for music download. Collaborative filtering technique is used to provide music recommendations. To see the simple steps how it works, the system takes ratings from each user's iTunes play lists and compares the ratings with those of other iTunes users who also have rated for their own music's.

Recommendation system in like- I –like: "like–i–like.org" is a website for movie recommender system. The assumptions underlying in this recommendation system is "Those who agreed to one thing tend to agree to the similar thing again" and "People with similar taste can be advisor for each other." By considering those assumptions, the system tries to find out if a new user has a similar interest with the existing user by finding users who rated similar numbers from 1 to 10 for specific movies and categorizes them by their preferences.

Table III: Shifts in matrix models outlining the evolution of recommender systems from information retrieval

Concept	Modeling matrix
Information retrieval	terms×documents
Information filtering	features×documents
Content-based filtering	features×artifacts
Collaborative filtering	people×documents
Recommender systems	people×artifacts

Challenges and Issues: Various techniques used in recommender system experiences some of the hurdles that will be described as follows:

Sparsity: Sparsity is the problem of lack of information. It is one of the problem encountered in recommender system and data sparsity has great control on the quality of recommendation. The main reason behind sparsity of data is that most users may not rate most of the items and the available ratings are usually sparse. Collaborative filtering [14] suffers from this problem because it depends on the rating matrix in most cases.

Cold Start Problem: Cold start problem refers to the situation when a new user or item just enters the system. Three kinds of cold start problems are: new user problem, new item problem and new system problem. In such cases, it is very difficult to provide recommendation as in case of new user, there is very less information about the user, for a new item and there is no rating are available. This problem will be solved by using hybrid approach.

Scalability: With the growth of numbers of users and items, the system needs more resources for processing information and forming recommendations. Majority of resources is consumed with the purpose of identifying users with similar tastes and things with similar descriptions. This problem is solved by considering the combination of various types of filters and systems physical improvement.

Privacy: Privacy is one of the major problems. In order to get the most accurate and valid recommendation, the system must get the large amount of possible information about the user, including demographic data and the location of a particular user. Obviously, the question of reliability, security and confidentiality of the given information arises. Many online shops provide effective privacy protection of the users by utilizing specialized algorithms and programs.

Over Specialization Problem: Users are restricted to receiving recommendations which look like to those already known or defined in their profiles in some cases and it is known as over specialization problem. It protects user from discovering new items and various available options. But, diversity of recommendations is a important feature of all recommendation system. The problem is solved using genetic algorithms and provides with a set of different and a wide range of alternatives.

Security and Privacy Issues: Collaborative filtering requires personal information from a user to give personalized recommendations. The more users express their preferences on items, the more accurate the recommendations they receive. The users must trust the recommender to protect their information appropriately. The user does not know how the recommendation is done, he/she should trust the accuracy of the recommender. The recommender should not break the trust of the users.

Shilling Attack: A shilling attack is an attack in which the system's recommendation for a particular item is calculated by submitting misrepresented opinions to the system. The attack has two objectives: decreasing the ratings of all the items outside its target item-set (push attack) to make them more recommended and increasing the ratings (nuke attack) of other items to make its target item-set less recommended. The different types of shilling attacks are RandomBot and AverageBot.

- A RandomBot is filterbot who randomly rate items outside of the target item-set with either the minimum rating (for nuke attack) or maximum rating (for pushattack).
- An AverageBot is a filterbot where the rating is based on the average rating of each item following a normal distribution with a mean equal to the average rating for that item. Another type of attack that may affect recommender is so called Sybil attack in which a dishonest user may create multiples users account in other to improve the recommendation of another user or another item. Recommender shall then provide ways to protect itself against those attacks since they are well known. Some systems provided CAPTCHA to stop filterbots fromcorrupting the ratings.

Experiment

Input Pattern in User Profile: In job seeking [15, 16] and recruiting websites, user profiles are important for both employers and candidates. Taking candidates for example, they can either input their information online or upload their CV files. In many websites it support users to import their profile from other websites like LinkedIn. Online profile forms contain specified fields. The profile fields for candidates mainly consists of personal information (name, gender, birthday, etc.), educational background and work experience. For companies and jobs, the main fields are location, industry, description and requirements. Data processing will be easier when it is dealing with structured and standardized profile information.

Profile similarity consists of measuring the extent to which two user profiles are similar in terms of content. The profile similarity measurement is shown in Figure 3.

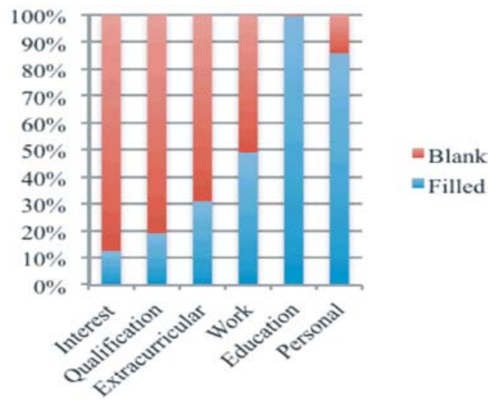


Fig. 6: Profile completion rate per field

Structured profiles contains several predefined fields that can be filled online like name, age and education. On the other hand, Unstructured profiles are uploaded with no standardized format (e.g. uploaded CV files). The overall similarity consists of a linear combination of weighted field similarity in the case of fields having predefined values and normalized content-based similarity in the case of free text fields for structured profiles.

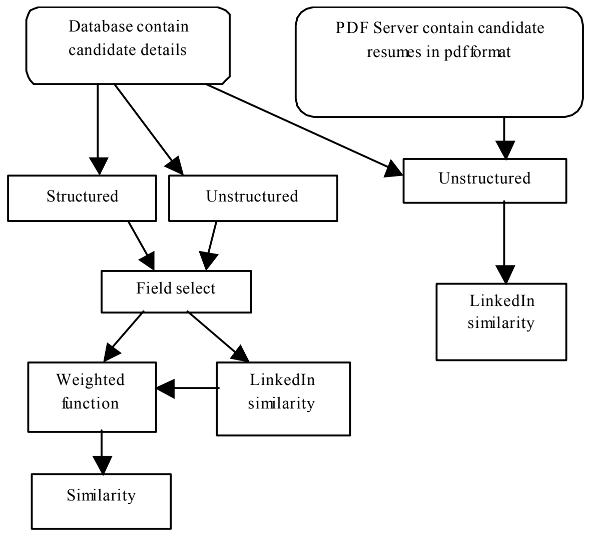


Fig. 4: Similarity computation process for candidate profile

Table IV shows the fields in candidate's profile we select and the weight assigned to each field. The choice of the fields and their corresponding weights are set after discussion with recruiters. They mainly consider the educational background, the university and the degrees are the most important fields since their target customers are graduating students. When we consider companies and jobs, the industry field, location, job title/position and its requirements are important fields for similarity measurement. When it comes to unstructured profiles, it is difficult to extract corresponding fields. When comparing two PDF files or one PDF file and one structured profile, both are parsed into unstructured data and LSA (Latent Semantic Analysis) [17] is used to compute their similarity. Both structured and unstructured profiles have textual content (i.e. description fields in online filled forms).

Table IV: Selected fields and weights of candidates' online profile

Field	Weight (%)	Field	Weight (%)
Gender	5	Age	10
University	10	Study course	20
Diploma	10	Language	5
Work experience	20	Qualification	10
Extra-curricular	10	Total	100

Based on these weight predicted for each fields suitable candidate will be selected.

User Interaction Patters: Job seeking and recruiting usually give some social media features [18] like connect, like, share and recommend to friends [19]. These features do not only help user discover interest and opportunities, but can also be exploited in recommender Systems. The possible actions are summarized below:

- Visit
- Share
- Like/Dislike
- Rating
- Recommend to friends
- Add to favorites list (or bookmark)
- Apply (for a job)

On the interface, most of the buttons provide interactions and used to express interest are located in easy-to-use places.

Table 4 Shows the precision of recommendation results of our hybrid, PS and CF.

Table V: Recommendation Results

Query	Candidate	Hybrid	PS	CF
Query1	UID12	0.8	0.1	0.3
	UID10	0.5	0	0.2
	UID3	0.1	0	0.2
Query2	UID4	0.7	0.3	0.5
	UID1	0.5	0.4	0.4
	UID6	0.5	0.3	0.4
Query3	UID20	0.5	0.2	0.2
	UID11	0.8	0.8	0.6
	UID16	0.7	0.2	0.5
Query4	UID8	0.7	0.3	N/A
	UID18	0.1	0	N/A
	UID13	0.2	0.8	N/A

The following graph shows the technique which has the highest efficiency in the recommender system.

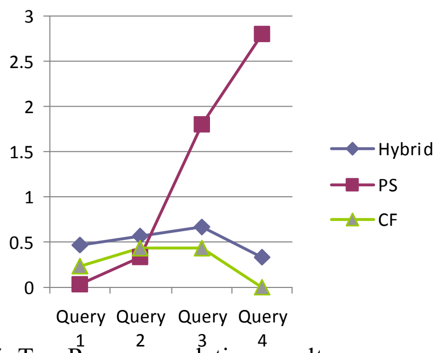


Fig. 5: Top Recommendation results

CONCLUSION

This paper presented various algorithms and techniques used to build the recommender system. Each of the algorithms and techniques has its advantage and disadvantage: user based approach are accurate but not scalable, item based approach are scalable but not precise as user based approach. Hybrid recommender system combines the features of user based and item based algorithm [20]. Research on recommender system is mainly focus on finding ways to improve the performance, scalability or accuracy of the algorithm. The research may be carried out in this area to explore and come with new methods to overcome the challenges. Thus the current recommendation system needs to be improved for providing better recommendation qualities.

REFERENCES

1. Joseph A. Konstan and John Riedl, 2012. Recommender systems: from algorithms to user experience.
2. Joeran Beel, Stefan Langer, Marcel Genzmehr and BelaGipp, 2013. Research Paper Recommender System Evaluation: A Quantitative Literature Survey.
3. Lalita Sharma and Anju Gera, 2013. A survey of recommendation system:Research Challenges.
4. Michael J. Pazzani and Daniel Billsus, 2007. Content based Recommendation Systems.
5. Marko Balabanovic and Yoav Shoham, 1997. Content-based, Collaborative Recommendation.
6. Dhoha Almazro, Ghadeer Shahatah, Lamia Albdulkarim and Mona Kherees, 2010. A Survey Paper on Recommender system.
7. Nathaniel Good, J. Ben Schafer, Joseph A. Konstan, Al Borchers, BadrulSarwar, Jon Herlocker and John Riedl, 2008. Combining Collaborative filtering with Personal Agents for Better Recommendations.
8. Lu, L.I.U. and W.U. Lihua, 2005. User Modelling for Personalized Recommender Systems.
9. Katrien Verbert and Hendrik Drachsle, 2007. Context-aware Recommender Systems for Learning: a Survey and Future Challenges.
10. Kumar, A. and Dr. P. Thambidurai, 2010. Collaborative Web Recommendation Systems -A Survey Approach.
11. Rafter, Bradley and Smyth, 2004. Automated collaborative filtering applications for online recruitment services.
12. Tarek Helmy, 2007. Collaborative multi-agent-based E-Commerce framework.
13. Sovren Group, 2006. Overview of the Sovren Semantic Matching Engine and Comparison to Traditional Keyword Search Engines.
14. Nitai B. Silva, Ing-Ren Tsang, George D.C. Cavalcanti and Ing-Jyh Tsang, 2010. A Graph-Based Friend Recommendation System Using Genetic Algorithm.
15. Lu, Helou and Gillet, 2012. Analyzing User Patterns to Derive Design Guidelines for Job Seeking and Recruiting Website.
16. Yao Lu, Sandy El Helou and Denis Gillet, 2013. A Recommender System for Job Seeking and Recruiting Website.
17. Senthilkumaran, V. and A. Sankar, 2013. Recommendation System for Adaptive E-learning using Semantic Net.
18. Pei-Shan Chang, I-Hsien Ting and Shyue-Liang Wang, 2012. Towards Social Recommendation System Based on The Data from Micro blogs.
19. Sanjog Ray and Anuj Sharma, 2010. A Collaborative Filtering Based Approach for Recommending Elective Courses.
20. Shweta Tyagi and Kamal K. Bharadwaj, 2012. A Hybrid Recommender System Using Rule-Based and Case-Based Reasoning.