International Journal of Research in Engineering, Science and Management
Volume-1, Issue-10, October-2018
www.ijresm.com | ISSN (Online): 2581-5782

Enhanced Job Recommendation System

Shivraj Hulbatte[1], Amit Wabale[2], Suraj Patil[3], Nikhilkumar Sathe[4]
[1,2,3,4]*Student, Department of Computer Engineering, Sinhgad Academy of Engineering, Pune, India*

***Abstract*—We address the problem of recommending suitable jobs to people who are seeking a new job. We formulate this recommendation problem as a supervised machine learning problem. Our technique exploits all past job transitions as well as the data associated with employees and institutions to predict an employee's next job transition. Dealing with the enormous amount of recruiting information on the Internet, a job seeker always spends hours to find useful ones. To reduce this laborious work, we design and implement a recommendation system for online job hunting. In this paper, we contrast user-based and item-based collaborative filtering algorithm to choose a better performed one. We also take background information including students' resumes and details of recruiting information into consideration, bring weights of co-apply users (the users who had applied the candidate jobs) and weights of student used liked jobs into their recommendation algorithm. At last, the model we proposed is verified through experiments study which is using actual data. The recommended results can achieve higher score of precision and recall, and they are more relevant with users' preferences.**

***Index Terms*—Collaborative filtering, content-based filtering, Vector Space Model (VSM)**

I. Introduction

The increase in usage of Internet has heightened the need for online job hunting. According to Job site's report 2014, 68% of online job seekers are college graduates or post graduates. The key problem is that most of the job-hunting websites just display the recruitment information to website viewers. Students have to go through all the information to find the jobs they want to apply. The whole procedure is tedious and inefficient. We need an easy job recommendation system where everyone will have a fair and square chance. This saves a lot of potential time and money both, on the industrial as well as the job seeker's side. Moreover, as the candidate gets a fair chance to prove his talent in the real world it is a lot more efficient system. The basic agenda of every algorithm used in today's world, be it a traditional algorithm or a hybrid algorithm, is to provide a suitable job that the user actually seeks and wishes for.

Recently, job recommendation has attracted a lot of research attention and has played an important role on the online recruiting website. Different from traditional recommendation systems which recommend items to users, job recommender systems (JRSs) recommend one type of users (e.g., job applicants) to another type of users (e.g., recruiters). In particular, job recommender system is designed to retrieve a list of job descriptions to a job applicant based on his/her preferences or to generate a list of job candidates to a recruiter based on the job requirements. To obtain a good recommendation results, many recommendation approaches are presented and applied in the JRS. Typically, given a user, existing JRSs employ a specific recommendation approach to generate a ranked list of jobs/candidates. However, different users may have different characteristics and a single recommendation approach may not be suitable for all users. Therefore, a high-quality JRS should have the capability of choosing the appropriate recommendation approaches according to the user's characteristic.

II. Objectives of the Work

- To recommend suitable jobs to the candidates
- To make the recruitment system more secure and easier.
- To suggest skills to the users so that they can acquire them and get a suitable job.
- To show the deserving candidates to the companies who may fit into their working culture.
- To make the process of job hunt easier for the fresher as well as experienced candidates.

III. Problem Statement

Dealing with the enormous amount of recruiting information on the Internet, a job seeker always spends hours to find useful ones. Many times, people who lack industry knowledge are unclear about what exactly they need to learn in order to get a suitable job for them. We address the problem of recommending suitable jobs to people who are seeking a new job. We formulate this recommendation problem as a supervised machine learning problem.

LITERATURE SURVEY

The JRS has been studied from many aspects. Al-Otaibi et al. [1] summarized the categories of existing online recruiting platforms and listed the advantages and disadvantages of technical approaches in different JRSs. For example, bidirectional recommendation is accomplished but only binary representation is allowed in the probabilistic hybrid approach. We also had done some research on feature extraction, resume mining, recommendation approach, ranking, and explanation for the JRS. S. T. Zheng [2] explained that user profiling and calculating similarity are presented as the prevailing process of a JRS.

From the technical perspective, JRS has been classified into five categories described as follows:

a) *Content-based Recommendation (CBR):* The principle of a content-based recommendation is to suggest items that have similar content information to the corresponding users. For example, in the recommendation that recommends jobs to a job applicant, the content is the personal information and their job desires. While recommending candidates to recruiters, the job description posted by recruiters, including the background description of enterprises, are used as the content for recommendation. The basic process of content-based recommendation is acquiring the content information of job applicants and jobs and calculating their similarities. So, the content information plays an important role in the content-based recommendation [3]. Yu et al. [4] presented a cascaded extraction approach for resumes to obtain the more effective information. Yi et al. [5] built a relevance-based language model -Structured Relevance Models for modeling and retrieving semi-structured documents. Furthermore, Paparrizos et al. [6] trained a machine learning model to predict candidates' next job transition based on their past job histories as well as the data of both candidates and enterprises in the web.

b) *Collaborative Filtering Recommendation (CFR):* Collaborative filtering recommendation, known as the user-to-user correlation method, finds similar users who have the same taste with the target user and recommends items based on what the similar users like. The key step in CFR is computing the similarities among users. Collaborative filtering recommendation algorithm can be classified into memory-based and model-based [7, 8]. In the memory-based collaborative filtering recommendation, a user-item rating matrix is usually used as the input [9, 10]. Applied in the job recruiting domain, some user behaviors or actions can generate the user-item rating matrix according to the predefined definitions and transition rules. Färber et al. [11] presented an aspect model to produce a rating matrix that assigns assessed values to candidate's profile using the Expectation Maximization (EM) algorithm. Collaborative Filtering works by building a database of preferences for items by users. For example, a new user, John, is matched against the database to discover neighbors, which are other users who have historically had similar taste to John. Items that the neighbors like are then recommended to John, as he will also probably like them.

For the purpose of subsequent discussion, we assume that the user-item ratings matrix is an incomplete m × n matrix R= [r_{uj}] containing m users and n items. It is assumed that only a small subset of the ratings matrix is specified or observed. Like all other collaborative filtering algorithms, neighborhood-based collaborative filtering algorithms can be formulated in two ways:

1. Predicting the rating value of a user-item combination: This is the simplest and most primitive formulation of a recommender system. In this case, the missing rating ruj of the user u for item j is predicted.
2. Determining the top-k items or top-k users: In most practical settings, the merchant is not necessarily looking for specific ratings values of user-item combinations. Rather, it is more interesting to learn the top-k most relevant items for a particular user, or the top-k most relevant users for a particular item. The problem of determining the top-k items is more common than that of finding the top-k users. This is because the former formulation is used to present lists of recommended items to users in Web centric scenarios. In traditional recommender algorithms, the "top-k problem" almost always refers to the process of finding the top-k items, rather than the top-k users. However, the latter formulation is also useful to the merchant because it can be used to determine the best users to target with marketing efforts.

Step 1: we assume that the ratings matrix is denoted by R, and it is an m × n matrix containing m users and n items. Therefore, the rating of user u for item j is denoted by r_{uj}. Only small subsets of the entries in the ratings matrix are typically specified. The specified entries of the matrix are referred to as the training data, whereas the unspecified entries of the matrix are referred to as the test data. There are two basic principles used in neighborhood-based models:

1. *User-based models:* Similar users have similar ratings on the same item. Therefore, if Alice and Bob have rated jobs in a similar way in the past, then one can use Alice's observed ratings on the job Software Engineer to predict Bob's unobserved ratings on this job.
2. Item-based models: Similar items are rated in a similar way by the same user. Therefore, Bob's ratings on similar software engineer jobs like web developer can be used to predict his rating on software engineer.

Step 2: In this approach, user-based neighborhoods are defined in order to identify similar users to the target user for whom the rating predictions are being computed. In order to determine the neighborhood of the target user *i*, his/her similarity to all the other users is computed. Therefore, a similarity function needs to be defined between the ratings specified by users. Such a similarity computation is tricky because different users may have different scales of ratings. One user might be biased toward liking most items, whereas another user might be biased toward not liking most of the items. Furthermore, different users may have rated different items. Therefore, mechanisms need to be identified to address these issues.

For the $m \times n$ ratings matrix $R = [r_{uj}]$ with m users and n items, let Iu denote the set of item indices for which ratings have been specified by user (row) u. For example, if the ratings of the first, third, and fifth items (columns) of user (row) u are specified (observed) and the remaining are missing, then we have $Iu = \{1, 3, 5\}$. Therefore, the set of items rated by both users u and v is given by $Iu \cap Iv$. For example, if user v has rated the first four items, then $Iv = \{1, 2, 3, 4\}$, and $Iu \cap Iv = \{1, 3, 5\} \cap \{1, 2, 3, 4\}$

= {1, 3}. It is possible (and quite common) for $Iu \cap Iv$ to be an empty set because ratings matrices are generally sparse. The set $Iu \cap Iv$ defines the mutually observed ratings, which are used to compute the similarity between the u^{th} and v^{th} users for neighborhood computation. One measure that captures the similarity Sim(u, v) between the rating vectors of two users u and v is the Pearson correlation coefficient. Because $Iu \cap Iv$ represents the set of item indices for which both user u and user v have specified ratings, the coefficient is computed only on this set of items. The first step is to compute the mean rating μ_u for each user u using his/her specified ratings:

$$\mu_u = \frac{\sum_{k \in I_u} r_{uk}}{|I_u|} \quad \forall u \in \{1 \ldots m\} \tag{1}$$

Then, the Pearson Correlation Coefficient between the two rows (users) u and v is defined as follows:

$$sim(u,v) = \frac{\sum_{i \epsilon C}(r_{u,i} - \bar{r}_u)(r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{i \epsilon C}(r_{u,i} - \bar{r}_u)^2}\sqrt{\sum_{i \epsilon C}(r_{v,i} - \bar{r}_v)^2}} \tag{2}$$

Strictly speaking, the traditional definition of Pearson (u, v) mandates that the values of μ_u and μ_v should be computed only over the items that are rated *both* by users u and v. Unlike Equation 2.1, such an approach will lead to a different value of μ_u, depending on the choice of the other user v to which the Pearson similarity is being computed. However, it is quite common (and computationally simpler) to compute each μ_u just once for each user u, according to Equation 2.1. It is hard to make an argument that one of these two ways of computing μ_u always provides strictly better recommendations than the other. In extreme cases, where the two users have only one mutually specified rating, it can be argued that using Equation 2.1 for computing μ_u will provide more informative results, because the Pearson coefficient will be indeterminate over a single common item in the traditional definition. Therefore, we will work with the simpler assumption of using Equation 2.1 in this chapter. Nevertheless, it is important for the reader to keep in mind that many implementations of user-based methods compute μ_u and μ_v in pair wise fashion during the Pearson computation.

c) *Knowledge-based Recommendation (KBR):* In the knowledge-based recommendation, rules and patterns obtained from the functional knowledge of how a specific item meets the requirement of a particular user are used for recommending items [12]. For example, employees who have one or more years of work experience exhibit better performance as compared to those without experience. This can be used as a job performance rule in the online recruiting. Chien et al. [13] developed a data mining framework based on decision tree and association rules to generate useful rules for selecting personnel feature and enhancing human capital. In addition, other types of knowledge such as ontology can also be used in the job recommendation. Lee and Brusilovsky [14] employed an ontology checker to match information with ontology and perform the classification in the JRS.

d) *Reciprocal Recommendation (ReR):* Firstly proposed by Luiz Pizzato et al. [15], reciprocal recommender is a special kind of recommender systems. The preferences of all the users are taken into account and need to be satisfied at the same time. As a result, ReR achieves a win-win situation for users and improves the accuracy of recommender systems that match people and jobs. Yu et al. [16] proposed a similarity calculation method for calculating the reciprocal value and achieving the reciprocal recommendation based on the explicit preferences obtained from users' resumes and the implicit preferences acquired from the user's interaction history. Malinowski et al. [17] also used a bilateral recommendation approach which considers the two parts of JRS to match the job applicants and jobs. Li et al. [18] proposed a generalized framework for reciprocal recommendation that is applied to online recruiting, in which they model the correlations among users by a bipartite graph.

e) *Hybrid Recommendation (HyR):* All recommendation approaches mentioned above have their limitations. To overcome the limitation, these approaches have been integrated to obtain better performance. Burke [12, 19] presented seven categories of the hybrid recommender system as follows: weighted, switching, mixed, feature combination, cascade, feature augmentation, and model. Malinowski et al. [17] applied the probabilistic model to two parts of JRS: a CV-recommender and a job recommender separately and integrated the result in order to improve the match between job applicants and jobs. Keim [20] integrated the prior research into a unified multilayer framework supporting the matching of individuals for recruitment and team staffing processes. Fazel-Zarandi and Fox [21] combined different matchmaking strategies in a hybrid approach for matching job applicants and jobs by using logic-based and similarity-based matching.

IV. Basic Flow

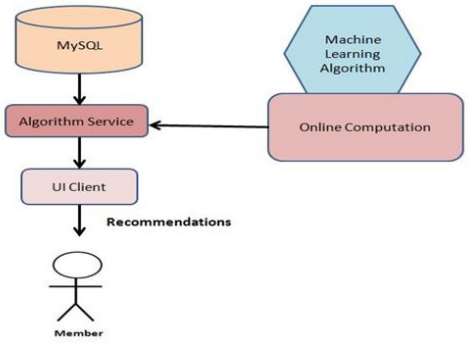

Fig. 1. A basic structure

We are aiming to use structured database such as MySQL to achieve high level space efficiency and time efficiency. For the algorithmic part we have to check the efficiency of all to get the most of it. Also we can have collaboration of two or more

algorithms to enhance the efficiency.

V. Proposed Architecture

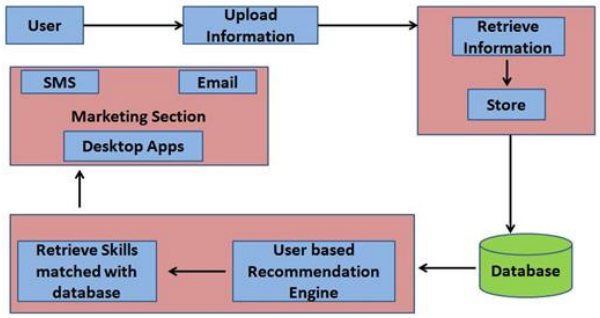

Fig. 2. Skills recommendation

VI. Hardware Requirements

System	:	Pentium IV 2.4 GHz
Hard Disk	:	40 GB
Ram	:	512 Mb

VII. Software Requirements

Operating system	:	Windows 7
Coding Language	:	JAVA, JSP and Servlets
IDE	:	JAVA Eclipse
Database	:	MySQL

VIII. Conclusion

On the basis of this study and various techniques to research and after implementation of algorithms, the collaborative filtering based algorithm is considered for its better performance and overall factors. Of course a lot of improvement and hybrid algorithms need to be implemented alongside collaborative filtering algorithm. To further optimize the recommendation system, and integrate the system for better performance we keep in check the sparsity of user profile and use some methods for filling user's preference matrix and how it can be utilized.

References

[1] S. T. Al-Otaibi and M. Ykhlef, "A survey of job recommender systems," International Journal of the Physical Sciences, vol. 7(29), pp. 5127-5142, July, 2012.
[2] S. T. Zheng, W. X. Hong, N. Zhang and F. Yang, "Job recommender systems: a survey," In Proceedings of the 7th International Conference on Computer Science & Education (ICCSE 2012), pp. 920-924, Melbourne, Australia, July, 2012.
[3] M. Gao and Y. Q. Fu, "User-Weight Model for Item-based Recommendation Systems," Journal of Software, vol. 7(9), pp. 2133-2140, 2012.
[4] K. Yu, G. Guan and M. Zhou, "Resume information extraction with cascaded hybrid model," In Proceedings of the 43rd Annual Meeting of the ACL, pp. 499-506, Ann Arbor, Michigan, June, 2005.
[5] X. Yi, J. Allan and W. B. Croft, "Matching resumes and jobs based on relevance models," In Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 809-810, Amsterdam, The Netherlands, 2007.
[6] I. Paparrizos, B. B. Cambazoglu and A. Gionis, "Machine learned job recommendation," In Proceedings of the fifth ACM Conference on Recommender Systems, pp. 325-328, Chicago, USA, October, 2011.
[7] J. S. Breese, D. Heckerman and C. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering," In Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence, pp. 42-52, 1998.
[8] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: a survey of the stateof-the-art and possible extensions," Knowledge and Data Engineering, IEEE Transactions on, vol. 17(6), pp. 734- 749, 2005.
[9] H. W. Ye, "A Personalized Collaborative Filtering Recommendation Using Association Rules Mining and Self-Organizing Map," Journal of Software, vol. 6(4), pp.732-739, 2011.
[10] L. Hu, W. B. Wang, F. Wang, X. L. Zhang and K. Zhao, "The Design and Implementation of Composite Collaborative Filtering Algorithm for Personalized Recommendation," Journal of Software, vol. 7(9), pp. 2040-2045, 2012.
[11] F. Färber, T. Weitzel and T. Keim, "An automated recommendation approach to selection in personnel recruitment," In Proceedings of the 2003 Americas Conference on Information Systems, pp. 2329-2339, Tampa, USA, 2003.
[12] R. Burke, "Hybrid recommender systems: survey and experiments," User Modeling and User-Adapted Interaction, vol. 12(4), pp. 331-370, 2002.
[13] C. F. Chien and L. F. Chen, "Data mining to improve personnel selection and enhance human capital: A case study in high-technology industry," Expert Systems with Applications, vol. 34(1), pp. 280-290, 2008.
[14] D. H. Lee and P. Brusilovsky, "Fighting information overflow with personalized comprehensive information access: a proactive job recommender," In Proceedings of the Third International Conference on Autonomic and Autonomous Systems, Washington, DC, USA, 2007.
[15] L. Pizzato, T. Rej, T. Chung, K. Yacef, I. Koprinska and J. Kay, "Reciprocal recommenders," In Proceedings of 8th Workshop on Intelligent Techniques for Web Personalization and Recommender Systems, held in conjunction with the 18th International Conference on User Modeling, Adaptation and Personalization (UMAP 2010), Hawaii, USA, June, 2010.
[16] H. T. Yu, C. R. Liu and F. Z. Zhang, "Reciprocal recommendation algorithm for the field of recruitment," Journal of Information & Computational Science, vol. 8(16), pp. 4061-4068, 2011.
[17] J. Malinowski, T. Keim, O. Wendt and T. Weitzel, "Matching people and jobs: a bilateral recommendation approach," In Proceedings of The 39th Hawaii International Conference on System Sciences, pp. 1-9, Hawaii, USA, 2006.
[18] L. Li and T. Li, "MEET: a generalized framework for reciprocal recommender systems," In Proceedings of the 21st ACM International Conference on Information and Knowledge Management, pp. 35-44, Hawaii, USA, 2012.
[19] R. Burke, "Hybrid web recommender systems," The Adaptive Web, vol. 4321, pp. 377-408, 2007.
[20] T. Keim, "Extending the applicability of recommender systems: a multilayer framework for matching human resources," In Proceedings of 40th Annual Hawaii International Conference on System Sciences, pp. 169-178, January, 2007.
[21] M. Fazel-Zarandi and M. S. Fox, "Semantic matchmaking for job recruitment an ontolgy based hybrid approach," In Proceedings of the 3rd International Workshop on Service Matchmaking and Resource Retrieval in the Semantic Web at the 8th International Semantic Web Conference, Washington D. C., USA, 2010.