International Journal of Soft Computing and Engineering (IJSCE)
ISSN: 2231-2307 (Online), Volume-9 Issue-3, September 2019

Job Recommendation System using Content and Collaborative Filtering Based Techniques

Juhi Dhameliya, Nikita Desai

Abstract*: Internet based recruiting platforms decrease advertisement cost, but they suffer from information overload problem. The job recommendation systems (JRS) have achieved success in e-recruitment process but still they are not able to capture the complexity of matching between candidates' desires and organizations' requirements. Thus, we propose a hybrid JRS which combines recommendations of content-based filtering (CBF) and collaborative filtering (CF) to overcome their individual major shortcomings namely overspecialization and over-fitting. In proposed system, CBF model makes recommendations based on candidates' skills identified from past jobs in which they have applied and CF model makes recommendations based on jobs in which similar users have applied and also those jobs in which that user has applied frequently together in very similar contexts using Word2Vec's skip-gram model. We used k-Nearest Neighbors technique and Pearson Correlation Coefficient. The recall of our proposed model is found to be 63.97% on a data set which had nearly 1900+ jobs and 23,000 job applicants.

Index Terms*: Collaborative Filtering (CF), Content Based Filtering (CBF), Hybrid Approach, Information Filtering, Job Recommendation System (JRS), Knowledge based Approach, Word2Vec models.

I. INTRODUCTION

Nowadays the internet has become the biggest source of information as the vast amount of advertisements related to jobs is posted every day on this platform. The amount of these online ads is much more than the number of job advertisements posted on print media, job centres, newspapers, articles, etc. Thus this huge quantity of data raises new challenges of information overload from the fact that candidates cannot exploit available resources effectively. With the excess of available online data, job seekers need to get job postings almost in a real-time, but traditional information system fails as the traditional Information Retrieval (IR) systems are based on Boolean search techniques. These search techniques are not useful as decision is made only on keyword matching and are not capable to capture the complexity between the candidates' desires and jobs' requirements.

The latest technology designed to overcome information overload problem is a recommendation system (RS) that generate a list of top "N" recommendations of items to users . The RS have demonstrated success in many areas such as for Amazon.com [1] and Netflix [2] but job recommendation domain is less explored. JRS is different from RS in other domains, as it not only recommends jobs (items) to job seekers (users) but also recommends one type of users (i.e. candidates) to other type of users (i.e. recruiters).

In this paper, we introduce a hybrid JRS which combines the recommendations from two models namely, CBF model using simple content based filtering approach and CF model using Word2Vec skip-gram model to overcome the individual shortcomings namely overspecialization and overfitting. Our model will only generate recommendations of jobs to candidate. Our JRS is limited to providing recommendations for computer science candidates only. To generate recommendations of candidates to the recruiters is not our objective as of now. In our proposed work, we don't take CVs of candidates to make recommendations, only the information about job applications made, is used to generate recommendation list.

The rest of paper is organized as follows; section II outlines the state of the art approaches about recruiting and job recommendation systems. Our proposed system is described in section III. Section IV discusses the dataset and experimental setup used to test proposed system. Finally, Section V summarises the results and points out some possible future works.

II. RELATED WORK

CF based approach has achieved huge success for building RS. The fundamental assumption of CF is that if users X and Y rate "N" items similarly or have similar behaviors, they will rate other items similarly [3]. Existing CF methods belong to two categories: i) memory-based (a.k.a. heuristic-based), and ii) model-based methods [4][5]. They use different similarity measures such as the Jaccard Index [6], Cosine Similarity [7], Euclidean distance [8] and Pearson Correlation Coefficient (PCC) [8] to select applicants or jobs for active applicants. Then, the prediction is done from the ratings of these neighbours or it generates a list of top N jobs as recommendations. Thus they are also called neighbourhood-based CF algorithms. Most of these algorithms can be classified as user based or item based algorithms depending upon whether the process of getting neighbours is focused on finding similar users or similar items [9].

Revised Manuscript Received on August 25, 2019.
* Correspondence Author
Juhi Dhameliya*, Department of Information Technology, Dharmsinh Desai University, Nadiad, India.
Nikita Desai, Department of Information Technology, Dharmsinh Desai University, Nadiad, India.

Item-based Collaborative filtering techniques are 28 times faster than traditional user-neighbourhood based collaborative filtering techniques [10].

On the other hand, model-based methods focus on learning an off-line model from the past activity of users using machine learning techniques such as clustering [6][8][11][12],

linear algebra methods such as Singular Value Decomposition with Latent Semantic Analysis [11], neural network [12], latent probabilistic methods [13], graph modeling [12][14][15], and Naive Bayes techniques.

One of the shortcomings associated with CF involves scenarios in which users or items do not have enough interaction data (e.g. users with unusual tastes, new users or jobs) and it only can recommend popular jobs. So, in such a scenario JRS is not able to cover unpopular jobs in the recommendation list. One way to overcome these shortcomings is to exploit another approach of RS known as CBF approach which build models using explicit domain-specific features of users and items [16]. Systems that combine both CF and CBF approaches are known as hybrid RSs. But, issue with CBF is that it suffers from overspecialization problem as it can only recommend jobs which are similar to applicant's profile and this can be handled by using CF.

Recent progress in neural embedding methods for linguistic tasks has dramatically advanced state-of-the-art NLP capabilities. These methods attempt to map words and phrases to a low dimensional vector space that captures semantic relations between words. Especially, Word2Vec models [18] achieved great success in other domains such as NLP and text mining fields [19]. Thus, we propose to apply word2vec skip-gram model to CF model which generates recommendations under these categories.

Since, our proposed system combines recommendations generated by both models CBF and CF model to overcome individual issues namely overspecialization and overfitting respectively, it is considered as hybrid model which falls under 'Mixed' category as per categorization of Bruke [17].

A. Job Recommendation Systems

After survey on several JRS proposed by different researchers, we observed that there is no gold standard dataset for testing any proposed JRS. Every researcher has used their own datasets. Also, as descriptions related to jobs are in unstructured form; many researchers have used different implicit and explicit information as features and have used Boolean values representing availability of that feature in building the feature vector.

Rafter et al. [6] designed a hybrid JRS CASPER for Job Finder Search Engine to generate job recommendations, but it suffers from scalability and sparsity issues. Other hybrid JRS which are proposed by Jochen Malinowski et al. [13] and Yao Lu et al. [14] also suffers from scalability and sparsity issues respectively. Ioannis Paparrizos et al. [15] proposed a system using graph modeling techniques which required lots of memory. Wenxing Hong et al. [12] have proposed online JRS called iHR for different users. It requires more computation time to generate recommendations.

Dr. S. Choudhary et al. [8] and G. Domeniconi et al. [11] also proposed a CF based JRS which has a problem of grey sheep users and cold start problem. Cold start problem of new user and new jobs can be overcome by using Knowledge-based JRS such as proposed by D. H. Lee, and P. Brusilovsky [20] and M. Hutterer [21] but they need more domain-specific knowledge.

B. Word2Vec based Recommendation Systems

Some of the recommendation methods [22], [23] use techniques from Word2Vec to represent their text based features. M. G. Ozsoy [23] proposed a system which generates a list the top-k venues/locations (e.g. restaurant, cafe) that the target user will visit/check-in in the future using Word2Vec skip-gram and Continuous bag of words model [24] which are combined with different CF techniques. Oren Barkan et al. [25] proposed a model 'Item2Vec' in which Word2Vec skip-gram with negative sampling is used to get a sequence of words is equivalent to a set or basket of items. They claimed 68% accuracy on 10k unpopular data.

III. PROPOSED JRS

In this work, we have implemented a hybrid JRS which provides a recommendation list of jobs to job seekers based on their personal preferences by combining recommendation lists of both models a simple CBF and CF using Word2Vec model. Architecture of proposed method is given in Fig. 1.

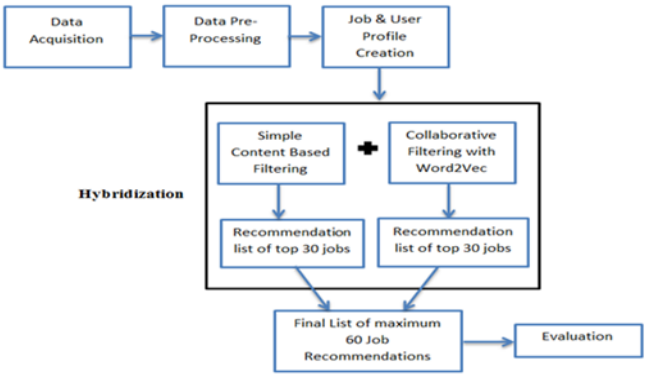

Figure 1: Architecture of Proposed Method

The steps to be implemented are as follows-

[Step 1] Data Pre-Processing

We pre-processed the dataset to extract useful skills from the jobs' description for simple CBF model. The steps are as follows-

(1.1) Remove HTML tags

Example:- , , , ,
, </p>, <style>, </style>, &bull, etc.

(1.2) Remove stop words/Irrelevant words

Example:- skills, good, knowledge, excellent, environment, degree, graduation, etc.

(1.3) Convert all words to lowercase

Example:-the words 'JAVA', 'Java' and 'java' have the same meaning, so we have converted it to lowercase.

(1.4) Add an underscore between the phrases of connected skill/technology

Example- 'cake php' is replaced by 'cake_php'.

(1.5) Synonyms of same skills are replaced by a base skill

Example- 'javascript', 'javascripts' and 'js' have the same meaning so replace these words with word 'javascript'. Also 'cake_php' is replaced by 'php'.

[Step 2] User and Job Profile Creation

In CBF, job profile contains unique skills such as 'java', 'php', 'oracle' etc. This profile is created by extracting unique skill using the TF-IDF method. User profile contains skills of those jobs in which they applied in the past. For example suppose user1 has applied in three jobs namely 'Job 1', 'Job 2', and ' Job 3', which have ' java, python, css, c#', 'java, sql, c', and 'php, html, xml' respectively. Then these skills become profile of user1.

To create a user profile for the CF model, we used the user applications sorted by date.

[Step 3] Generate recommendations by CBF Model

As input, this model takes all the Job IDs in which active user has applied and gives recommendation list of top 30 jobs to active users. The detailed steps are as follows-

(3.1) Create Users-Skills matrix

It is a utility matrix in which non-zero values represents the availability of skills a particular user has. In Users-Skills matrix, each row represents user and each column represents a particular skill. This matrix is used as input for the training of CBF model.

(3.2) Generate a job recommendation list

We have used k-Nearest Neighbors technique to find most similar users of active users. PCC is used as distance matrix to compute similarity between users. We take k= 80 and then jobs in which these similar users have applied, gives us job list. After that, we remove those jobs whose date exceeds the deadline in order to get recommendation list in timely manner.

(3.3) Get the weighted Job recommendation list

We have categorized the users as of two types: Random and Focused, based on their job application behavior. Users who have applied in five or less than five jobs are treated as focused users ,as it was found that they typically apply only to the related jobs as per skill set. For example if a user is focused, and he has applied in "Php, Java, and Oracle" based jobs, he would typically target and apply only to those jobs which have these specific requirements. Users who have applied in more than five jobs are treated as random users and they are found to be applying in any job in the domain irrespective of the skills. .Thus we gave weights to each job in the list extracted in above step, based on whether that job is applied for by the focused users or random users. If the job is applied by a focused user, we have given more weightage as compared to a job added to list from the list of random user.

(3.4) Finalize the Job recommendation list.

Sort the list into descending order of their weights and keep top 30 jobs in the final list.

Experiments which are performed on CBF model are mentioned in the section IV. In these experiments, we have tried various setups on Users-Skills matrix to improve the recommendation list and subsequently we selected the best one based on empirical results.

[Step 4] Generate recommendations by CF Model

CF model is an implementation of CF approach which takes number of jobs in which active user has applied as input and gives recommendation list of top 30 jobs to active users.

(4.1) Creation of User-Job matrix

It is utility matrix of binary values in which each row represents user and each column represents a particular column. In User-Job matrix, "1" indicates that user has applied in that job, otherwise "0". This matrix is used as input for the training of CF model.

(4.2) Get the weighted Job recommendation list

Use the same mechanism mentioned in step 3.2 to 3.4

[Step5] Generate a final recommendation list.

Merge the recommendation lists of both models obtained in step 2.1.2 and 2.2.2 and generate a final recommendation list with maximum 60 jobs, sorted as per weights in descending order of date.

A. Evaluation of Proposed Work

To evaluate the proposed model, we give first 10 job applications of test users as input to both CBF and CF models. By using these jobs as past behavior of test users, models try to recommend those jobs in which users should apply. We would then compare this recommendation list given by the model with the "actual" list which we have with us in the data set. And based on the recall measure, we try to judge the performance of our proposed system.

IV. EXPERIMENTS AND RESULTS

A. Dataset

We have used WUZZUF dataset which is an online recruitment site in Egypt helping employers and job seekers find their right match through real-time recommendations and around the clock support. Dataset includes job posting data and its corresponding applications of 2014 to 2016. Mainly it contains two csv files;

- Wuzzuf_Job_Posts_Sample.csv: This file contains data related jobs which has19 columns.
- Wuzzuf_Applications_Sample.csv: This file contains data related users' application which has 4 columns.

We pre-processed the dataset to filter out the jobs related to computer science domain and its corresponding applicants from both csv files. From total 21190 jobs and 1048576 applicants, we extracted 1957 job samples and 23620 corresponding applicants which we used to make recommendations.

We pre-processed the dataset to filter out the jobs related to computer science domain and its corresponding applicants from both csv files. From total 21190 jobs and 1048576 applicants, we extracted 1957 job samples and 23620 corresponding applicants which we used to make recommendations.

To evaluate the proposed system we have split users' applications into training users and testing users. Users who have applied in jobs in 2014 and 2015 are treated as training users. Also, those users who have applied in less than 10 jobs in 2016 are treated as training users. In this set up, we get 23090 users in which maximum applications by a user is 179 and minimum is 1. Users who have applied in more than 10 jobs in year of 2016 are treated as test users in our proposed work. Subsequently, we get 590 users in which maximum applications by a user is 71 and minimum application by a user is 10. If user has applied in more than 10 jobs in 2016 as well as he has applied in jobs in year of 2014 and/or 2015, he will be present in training users as well as testing users.

B. EVALUATION MEASURE

The effectiveness of a job recommendation system can be evaluated in terms of its precision, recall and F-measure. A precision is a percentage of correctly recommended jobs among total number of recommended jobs. A recall can be defined as fraction of relevant jobs that are also part of the set of recommended jobs.

$$precision = \frac{TP}{TP + FP} \quad (1)$$

$$Recall = \frac{TP}{TP + FN} \quad (2)$$

$$F - measure = \frac{2}{\frac{1}{precision} + \frac{1}{recall}} \quad (3)$$

C. EXPERIMENTS

First, we experimentally evaluated performance of various similarity measures by performing experiments on a sample data set. We have used entire training data and first 10 test users' data. The k-NN model was used to evaluate the performance of four similarity measures namely Jaccard Index, Cosine Similarity, Spearman Rank Correlation (SRC) and PCC. We have concluded that PCC gives more accurate results with recall of 57%, as compared to other three similarity measures as is depicted in the Fig. 2.

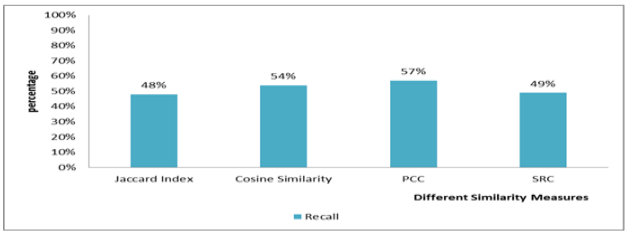

Figure 1: Evaluation of Different Similarity Measure

In the next experiment, we tried to determine the most effective value of 'k' to be used for k-NN technique. To achieve it , we experimented with values of "k" in a range from 10 to 100 and empirically found that as "k" increases precision, recall and f-measure also increase till k=80. But later, as value of k increases recall also increases but precision and f-measure decrease. Thus, we have set k=80.

C.1) Experiments performed on CBF.

Experiment C.1.1:-

Setup- Basic CBF which is described in section III.

Size of Users-Skills matrix- 23090*575.

Recall obtained -36.32%.

Experiment C.1.2:-

Setup-We replaced various frameworks of programming languages by its generic name. E.g. frameworks such as 'laravel', 'cake_php', 'yii', etc are frameworks of php. Thus, we replaced these frameworks with 'php'. By doing this, our set of unique skills is decreased from 575 to 315.

Size of Users-Skills matrix- 23090*315.

Recall obtained -35.03%.

As every user don't know every framework and libraries of languages this approach fails to give accurate results.

Experiment C.1.3:-

Setup- We used Word2Vec skip-gram model technique for user profile creation. To create user profile, we selected skills mentioned in those jobs in which user had applied in past and took top-10 most similar skills from these skills. To find the similar skills we have used the word vector given by Word2Vec. In this approach, we have taken jobs' description as a sentence, with each word being a skill of a particular job.

Several different settings are evaluated when the input data is modeled using skip-gram technique. We have used gensim toolbox and set the appropriate parameters to optimum. The details of the parameters and how they are tuned are as follows:

- min_count: The technique ignores the items whose frequency is less than min_word (minimum expected words) count parameter. We have set the parameter min_count to 1 in order not to lose any skills that is not used frequently.
- size: size parameter represents the dimension of the feature vectors and its default value is 100. It is stated that bigger size value can lead more accurate model, but requires more data. As our dataset is not too large, we have set the size parameter to 4.
- window: window parameter assigns the maximum distance between the current and the predicted items and its default value is 5. In our experiments, we have used this default value.

Once, we get the user profile, we generate a recommendation list of 30 jobs using k-NN with PCC for the active user. By using Word2Vec, we get irrelevant skill-set of users. Our approach finds the most similar neighbors using this skill-set of active users. Thus, in this method we get false recommendations and subsequently recall drops to 30.03% as compared to 36.32 of Basic CBF technique.

Hence, we have finalized basic CBF for our proposed system.

C.2) Experiments performed on CF.

Experiment C.2.1:-

Setup-Basic CF which is described section III.

Size of User-Job matrix- 23090*581.

Recall obtained -37.92%

The basic CF fails in a scenario when all neighborhood of the active user apply in one/two same jobs.

Experiment C.2.2:-

We have used Word2Vec's skip-gram model implemented in gensim library of python. In our approach, we have taken users' application list as a sentence, with each word being a job in which user has applied.

So, then training the Word2vec model on those sentences essentially means that for each job the user has applied to in the past, we're using the jobs they have applied to before and after to teach our model that those jobs somehow belong to the same context. Fig. 3 represents an idea of what the neural network would look like with jobs instead of words.

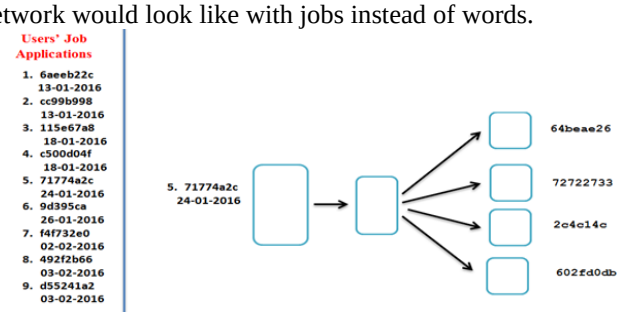

Figure 2: Word2Vec with Jobs

At the end of training phase, we get a model where each job is represented by a vector of weights in a high dimensional space. The vectors of similar jobs will have weights that are closer together than the vectors of jobs that are unrelated. We look at the weights as coordinates in a high dimensional space, with each job being represented by a point in that space. This space is defined by dozens of dimensions, which we cannot easily visualize as humans, but we can use dimensionality reduction techniques such as t-SNE [26] to reduce the high dimensional vectors to 2 dimensions, and plot them on a graph as shown in Fig. 4. Each point in the figure represents a job, and the closer the points are to each other, the more similar the jobs are.

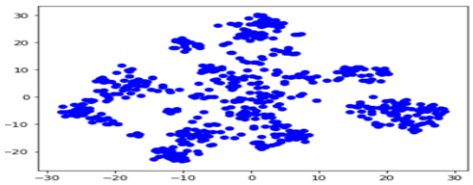

Figure 3: Jobs in 2d Space

After getting these job vectors, we created user profile for active user by averaging job vectors of those jobs in which the user has applied. This user profile represents the user in the same vector space in which our jobs are located. After that, by using k-NN with PCC as distance matrix, we give top 30 jobs as a recommendation list to the active user.

We trained the Word2Vec model with different values of model's parameters. We used 11 different setups for setting parameter values of model to get best results. We set parameter min_count to 1, not to lose any jobs that are not used frequently. From experiments, we found that best results are obtained with size parameter set to 30, window parameter is 15, and epochs parameter also 15 . We concluded that CF with Word2Vec is better than Basic CF. Thus; we have finalized CF with Word2Vec for our system of hybridization.

At the last, we have combined results of different models of both CBF and CF. We combined these models into three setups as follows;

- Setup1 -Hybridization of basic CBF and basic CF.
- Setup2 -Hybridization of CBF with Word2Vec and CF with Word2Vec.
- Setup3 -Hybridization of basic CBF and CF using Word2Vec.

The table 1 represents results obtained under various setups of hybrid models. Consecutively it can be derived that by combining the recommendation lists of basic CBF and CF with word2Vec gives more accurate results than other hybrid models.

Table 1: Results of Different Hybrid Models

	Setup1	Setup2	Setup3
Precision	9.23	8.81	**9.6**
Recall	52.89	61.38	**63.97**
F-measure	14.68	14.52	**15.71**

V. CONCLUSION

From the survey based on Job Recommendation Systems, it was concluded that by combining two or more approaches we can get more reliable and accurate recommendations of jobs. Further, it was found that no benchmark dataset is available on which implementations and testing can be done. CBF needs lots of pre-processing on data as it is very difficult to identify matching skills as recruiters write job description mentioning "expected skill set", in highly diversified formats. From the experiments, we observed that usage of Word2Vec in CBF approach generates false recommendations as it included irrelevant skills in a user profile. But usage of Word2Vec in CF gave more accurate results than basic CF. Further; we have found that our proposed hybrid model which is a combination of recommendation lists of both models namely - a basic CBF and CF with Word2Vec, overcomes the issues of overspecialization and overfitting. The recall of our proposed model is hence 63.97%. Proposed model fails to give recommendations for new users. This mainly occurs as system does not take any user's "explicit" skill set, besides as new user has not applied to many jobs, so automatic identification of his skill set gets challenging.

In future, we will create taxonomy for the job domain to give more accurate recommendation list to job seekers. We can also expand our proposed work by including CurriculumVitae of new users in order to handle the current shortcoming of our proposed model.

REFERENCES

1. G. Linden, B. Smith, and J. York, "Amazon.com Recommendations: Item-to-Item Collaborative Filtering," Published by the IEEE Computer Society, IEEE Internet Comput., vol. 7, no.1, pp. 76-80, 2003.
2. S. Lanning, and J. Bennett, "The Netflix Prize," in ACM digital library, San Jose, California, USA, August 12, 2007.
3. T. Khoshgoftaar, X. Su, "A Survey of Collaborative Filtering Techniques," Adv. Artif. Intell., pp. 421-425, 2009.
4. J. Breese, D. Heckerman, and C. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering," In Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence, Madison, WI, ACM, pp. 43-52, July 24 - 26, 1998.
5. G. Adomavicius, and A. Tuzhilin, "Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions," IEEE Trans. Knowl. Data Eng. , pp. 734-749,2005.
6. R Rafter, K. Bradley, and B. Smyth, "Personalised Retrieval for Online Recruitment Services," AH '00 Proceedings of the International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems, pp.62-72, August 28 - 30, 2000.

7. W. Shalaby, B. AlAila, M. Korayem, L. Pournajaf, K. AlJadda, S. Quinn, and W. Zadrozny, "Help Me Find a Job: A Graph-based Approach for Job Recommendation at Scale," Proceedings of the International Conference on Big Data (Bif Data), Boston, MA, USA, December 11-14, 2017.
8. S. Choudhary, S. Koul, S. Mishra, A. Thakur, and R. Jain, "Collaborative Job Prediction based on Naïve Bayes Classifier using Python Platform," In proceeding of International Conference on Computation System and Information Technology for Sustainable Solutions (CSITSS), Bangalore, India, October 6-8,2016.
9. Z. Huang, D. Zeng, and H. Chen, "A Comparative Study of Recommendation Algorithms in Ecommerce." IEEE Intell. Syst., vol. 22, no. 5, pp. 68-78, September, 2007.
10. G. Karypis, "Evaluation of Item-Based Top-N Recommendation Algori thms," proceeding of the tenth international conference on Information and knowledge management, pp. 247-254 Atlanta, Georgia,USA, Octbor 05-10, 2001.
11. G. Domeniconi, G. Moro, A. Pagliarani, K. Pasini, and R. Pasolini, "Job Recommendation From Semantic Similarity of LinkedIn Users' Skills," Proceedings of the 5th International Conference on Pattern Recognition Applications and Methods, pp. 270-277, ICPRAM-2016.
12. W. Hong, S. Zheng, and H. Wang; "Dynamic User Profile-Based Job Recommender System," In proceedings of 8th International Conference on Computer Science & Education, Colombo, Sri Lanka IEEE, April 26-28, 2013.
13. J. Malinowski, T. Keim, O. Wendt, and T. Weitzel, "Matching People and Jobs. A Bilateral Recommendation Approach," Proceedings of the 39th Hawaii International Conference on System Sciences, Kauia, HI, USA IEEE, January 23rd , 2006.
14. Y. Lu, S. Helou, and D. Gillet, "A Recommender System for Job Seeking and Recruiting Website,"
15. I. Paparrizos, B. Cambazoglu, and A. Gionis "Machine Learned Job Recommendation," RecSys '11 Proceedings of the fifth ACM conference on Recommender systems, Chicago, Illinois, USA, pp. 325-328, October 23–27, 2011.
16. R. Mooney, and L Roy, "Content-Based Book Recommending Using Learning for Text Categorization," Proceedings of the Fifth ACM Conference on Digital Libraries, New York, NY, ACM pp. 195-204, 2000.
17. R. Bruke, "Hybrid Recommender Systems:Survey and Experiments," User Model. User-Adapt. Interact, vol. 12, no. 4, pp. 331-370, 2002.
18. T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J Dean, "Distributed Representations of Words and Phrases and their Compositionality," in CoRR-Computing Research Repository – ArXiv, October 16, 2013.
19. C. Musto, G. Semeraro, M. de Gemmis, and P. Lops, "Word embed-ding techniques for content-based recommender systems: An empirical evaluation," in Poster Proceedings of the 9th ACM Conference on Recommender Systems, RecSys 2015, Vienna, Austria, September 16, 2015.
20. D. H. Lee, and P. Brusilovsky, "Fighting Information Overload with Personalized Comprehensive Information Access: A Proactive Job Recommender," Third International Conference on Automatic and Autonomous Systems (ICAS'07), IEEE, June 19-25, 2007.
21. M. Hutterer, "Enhancing a job recommender with implicit user feedback," in Fakultät für Informatik, Technischen Universität Wien., pp. 107, 2011.
22. D. Shin, S. Cetintas, and K. Lee, "Recommending tumblr blogs to follow with inductive matrix completion," in Poster Proceedings of the 8th ACM Conference on Recommender Systems, RecSys 2014, Foster City, Silicon Valley, CA, USA, October 6-10, 2014.
23. M. G. Ozsoy, "From Word Embeddings to Item Recommendation," in CoRR-Computing Research Repository – ArXiv, June 15, 2016.
24. X. Rong, "word2vec Parameter Learning Explained," in CoRR-Computing Research Repository – ArXiv, June 5, 2016.
25. O. Barkan, and N. Koenigstein, "ITEM2VEC: Neural item embedding for collaborative filtering," 26th International Workshop on Machine Learning for Signal Processing (MLSP), September 13-16, 2016.
26. L. Van der Maaten, & G. Hinton, "Visualizing data using t-SNE," Journal of Machine Learning Research, pp. 2579-2605, 2008.

AUTHORS PROFILE

Juhi Dhameliya, Juhi Dhameliya is a scholar researcher. She is skilled in Machine Learning (ML), Deep Learning, Data Structures, Image Processing (IP), Data Mining (DM), Natural Language Processing (NLP), Recommendation Systems (RS) and Information Retrieval (IR). She is strong educational professional with master of technology focused in Information & Technology Engineering. She will be associated at Infosys as a System Engineer soon. She has published a survey paper on Job Recommendation Systems in an international conference. She is also a member of International Association of Engineers (IAENG).

Nikita Desai, Prof Nikita Desai is an Experienced Associate Professor with a demonstrated history of working in the education management industry from nearly 18 years. She is skilled in Automaton Theory, Artificial Intelligence (AI), Data Structures, Compiler Design, Natural Language Processing (NLP), and Algorithms Design with analysis. She is strong education professional with a Masters of Engineering focused in Computer Engineering. Currently she is associated with Information Technology Department of Dharmsinh Desai University, Nadiad, Gujarat. She has published 15 research papers in National /International Conferences and 5 papers in International journals, till date .She has successfully guided 14 candidates for their dissertation work at Post Graduate level.