Intelligent Human Systems Integration (IHSI 2024), Vol. 119, 2024, 841–850
https://doi.org/10.54941/ahfe1004604

Mobile Application for Job Recommender System Connecting Students and Startups/SMEs for Practical Experience and Skill Development

Noura Alhakbani

Information Technology Department, College of Computer and Information Sciences, King Saud University, Riyadh 11543, Saudi Arabia

ABSTRACT

This paper proposes a mobile application with a job recommender system (JRS) based on machine learning (ML) techniques to address the challenges faced by undergraduate and fresh graduate students in finding suitable job opportunities. The JRS utilizes content-based filtering (CB) with the Count Vectorizer function and cosine similarity to match student profiles with job requirements. The application provides personalized job recommendations for students and suggests potential candidates to companies. The findings demonstrate the efficacy of the proposed approach in bridging the gap between students and industries, aiding in practical experience acquisition for students and addressing recruitment challenges for startups and SMEs.

Keywords: Job recommender system (JRS), Content-based filtering (CBF), Count vectorizer

INTRODUCTION

Practical experience plays a vital role in helping students determine their desired fields, bridging the gap between the labor market and the academic output of university students. Startups and small and medium enterprises (SMEs) have been recognized as effective drivers of economic and social growth by providing valuable services and employment opportunities. These companies also contribute to the development of technical and managerial skills through training programs. However, SMEs often face challenges in recruiting and training manpower due to their limited resources and the absence of dedicated human resources (HR) departments (Shah et al. 2020).

Technological advancements have revolutionized recruitment methods, offering solutions to various challenges. The recruitment process, a critical function of HR departments, aims to identify the most suitable candidates for company positions. Tech giants have introduced innovative approaches, utilizing technical services and improved e-recruitment platforms to enhance employee recruitment and extend work flexibility. While e-recruitment platforms efficiently reach a large pool of potential job seekers, they often struggle with accurately matching applicants with job requirements using the Boolean search technique (Mashayekhi et al. 2022).

To address these challenges, artificial intelligence (AI) techniques are increasingly employed in e-recruitment processes. AI algorithms excel in handling repetitive tasks, enhancing hiring processes, increasing work flexibility, and improving decision-making, thereby reducing time and effort required for hiring. Organizations are encouraged to adopt these advanced technologies to gain a competitive advantage in recruitment and selection, with recommender systems (RSs) becoming crucial in achieving optimal matches between job seekers and jobs (Abia and Brown 2020; Gryncewicz et al. 2023).

This research project aims to design and develop a mobile application that supports the Arabic language. The application targets both students and startups/SMEs, offering a platform for startups and SMEs to find suitable employees for essential tasks and projects. Simultaneously, students, including high school, university, and fresh graduates, can access job opportunities aligned with their interests, developing their skills and building their careers. The application provides flexibility, enabling students to utilize their free time and gain practical experience. By strengthening students' resumes and providing practical experience before graduation, they become familiar with market needs and can actively work towards meeting them.

The paper is organized into five sections. Section 2 reviews related works in the field. Section 3 presents the system design. Section 4 describes the challenges anticipated during and after implementation. Finally, Section 5 concludes with a summary of our findings, future work, and design considerations.

BACKGROUND

Recommender Systems General Framework

RSs have proven to be highly advantageous in various domains, including e-business, where job recommender systems (JRSs) play a crucial role (Zangerle and Bauer 2022). JRSs differ from other RSs as they facilitate a bidirectional recommendation process between job seekers and job candidates, providing recommendations for both parties (Al-Otaibi 2012; Dhameliya and Desai 2019).

Previous studies such as (Isinkaye et al. 2015; Schafer 2011) have outlined a general framework for RS operations, comprising three essential phases: Information Collection Phase: In this phase, the RS gathers comprehensive user information to generate accurate recommendations. Information collection can occur through three main approaches. The first approach is explicit feedback, where users provide information directly through system interfaces. The second approach is implicit feedback, which involves inferring information from user behavior, such as navigation history, button clicks, and page dwell time. The third approach is hybrid feedback, which combines explicit and implicit feedback to overcome the limitations of each approach. Learning Phase: Learning algorithms are employed to analyze the user data collected during the information collection phase. These algorithms help identify patterns that are relevant for specific recommendation scenarios. Recommendation Phase: This phase is the core component of RSs, where recommendations are generated based on the patterns learned during the previous phase (Schafer 2011). Recommendation generation can be accomplished by directly utilizing the datasets acquired during the information collection phase, either through memory or model-based approaches, or indirectly by leveraging observed user actions within the system.

Recommender System Technique

RSs usually predict that good recommendations can be made to the user using effective recommendation techniques. Several recent studies (Das et al. 2017) have stated that the RSs can be divided into personalized and non-personalized RSs, as represented in Figure 1. The Personalized RSs aim to recommend items to users according to their previous behavior. This type of RSs is classified into three main approaches namely collaborative filtering (CF), CBF, Hybrid filtering and other approaches for recommendation systems.

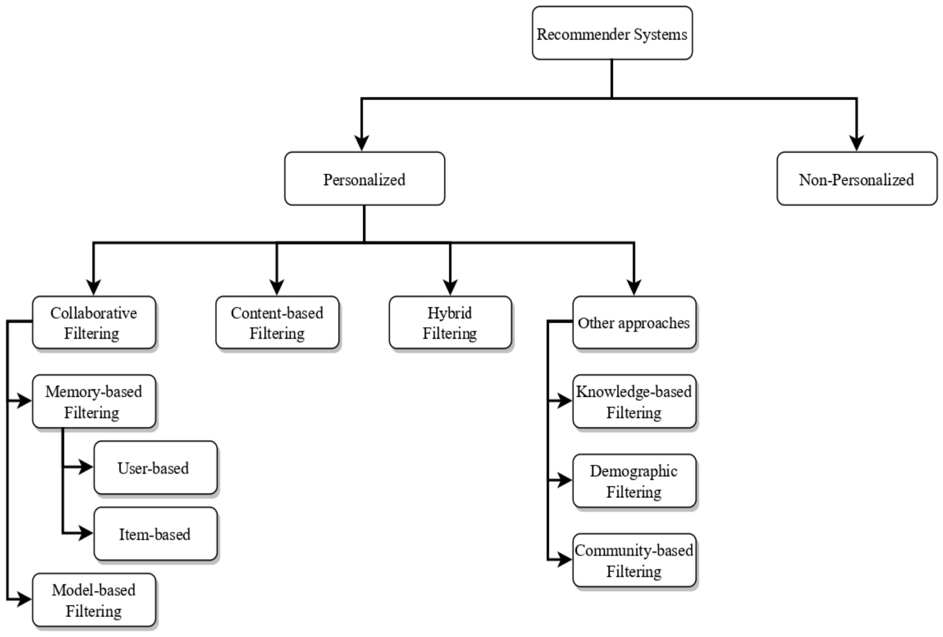

Figure 1: Recommender system techniques.

Major Challenges in Recommender Systems

Recommender systems (RSs) encounter various challenges that can significantly impact their effectiveness. These challenges include the cold-start problem, data sparsity, scalability issues, overspecialization, and shilling attacks. This section discusses these challenges and examines proposed solutions by researchers.

The cold-start problem arises when RSs encounter new users or items without any historical data, making it difficult to provide accurate recommendations. Collaborative filtering (CF) approaches are particularly affected by this problem (Mohamed et al. 2019). To address the cold-start problem, (Halder et al. 2012) introduce the concept of "movie swarm mining." This approach mines popular and interesting movies to generate initial recommendations for new users. For new items, a set of movies suitable for planning recommendations is identified through movie swarm mining, and these movies are suggested to related interested users who form a swarm within a group. While this method demonstrates effectiveness, it has limitations in identifying user groups solely based on movie genres.

Data sparsity is another challenge in RSs, as users often rate only a fraction of the available items, resulting in a sparse user-item matrix (Thorat et al. 2015). One potential solution to the data sparsity problem is to reduce the dimensionality of the user-item interaction matrix by generating clusters of the most relevant users and items. These reduced matrices can then be used for prediction purposes (Chen et al. 2011). Another approach involves utilizing singular value decomposition (SVD) to create a low-dimensional representation of the original customer-product space (Sarwar et al. n.d.).

As RSs accumulate and process increasingly large and dynamic datasets, the computational complexity of making predictions can become a bottleneck. Memory-based CF algorithms suffer from scalability issues as they use the entire dataset for recommendations, while model-based approaches offer better scalability (Dhameliya and Desai 2019).

Overspecialization occurs when RSs fail to provide diverse item recommendations that may interest the user. Content-based filtering (CBF) approaches are particularly susceptible to this problem, as they heavily rely on the user's previous preferences for similar items (Dhameliya and Desai 2019). To mitigate overspecialization, (Adamopoulos and Tuzhilin 2014) propose a probabilistic neighbourhood selection technique for neighbourhood-based CF recommendations. Rather than solely considering the highest similarity, the technique considers different degrees of similarity between users. Initial weights are assigned to each neighbour using a distance metric, resulting in the selection of diverse neighbours and diverse recommendations.

Furthermore, RSs are vulnerable to malicious attacks aiming to manipulate item ratings to deceive the system and generate poor recommendations (Mohamed et al. 2019). Several detection algorithms, categorized as supervised and unsupervised methods, can be employed to identify and mitigate these attacks.

RELATED WORK

In this section, we reviewed research in this filed. All mentioned studies are implemented an RS based on a CF technique for job portals. Initially, the data is obtained through the job portal, which records all users' preferences and previous activities. The data is cleaned, and noise is removed from the users' documents. The information is initialized before being processed. After that, the computation model deals make calculations. Finally, users are given recommendations based on the result set produced through the model's computations.

Zhang et al. (2014) compared user-based and item-based CF algorithms by constructing a student job hunting system with these two algorithms and evaluating the system with different similarity methods. They used a job-hunting website dataset that included 2,503 jobs and 7,610 students' resumes. The system was evaluated using precision and recall. The item-based algorithm achieved higher results than the user-based algorithm, achieving 58.33% precision and recall based on the log-likelihood similarity method, while the user-based algorithm with 40 neighborhood numbers achieved 56.41% precision and recall.

Yadalam et al. (2020) proposed a career RS that suggests applicable careers for engineering students based on their interests and abilities; the system allows users to rate the job recommendations based on their experience. The researchers used a filtering approach to solve cold-start, trust, and privacy problems associated with CF approaches. The cosine similarity method was used to find similarities between user preferences and the jobs, and the Python language was used to apply machine learning algorithms. The dataset they created contained 17 suggested job-role labels as columns and 20,000 entries from multiple databases.

Mpela and Zuva (2020) implemented a mobile proximity job employment RS with clientserver architecture to identify the most suitable job seeker for temporary job employment in a particular area. They used a CBF recommendation algorithm to find the similarity between resumes of job seekers and job descriptions; a VSM was used to represent selected features and cosine similarity to calculate similarity. In addition, google maps were used to find the distance between the job seeker and the job area to incorporate the job seeker's proximity into the recommendation process. The dataset consisted of 203 job descriptions and 1,431 resumes extracted from the Kaggle website. Results showed that the system achieved an impressive performance, with a 0.994 precision value and 0.975 recall value.

SYSTEM DESCRIPTION

The proposed system has three main users: An employer who posts the jobs, a student who applies for the posted jobs and an admin who maintains the database.

The system's front end provides an interactive interface to potential users and manages access to and storage of users' data in the database. In general, the process begins with initializing the data that has been read from the database, then getting the numeric features from the data as vectors. Then, the similarity measurement is applied to the vectors to get the top-recommended jobs/students according to the similarity score. The architecture of the system is shown in Figure 2.

To meet the needs in the field of HR through the development of modern techniques in the hiring process. The algorithm is created for this project to make proper job recommendations. So relevant information was extracted regarding the student's profile and job description for the jobs offered posted by the company. The job dataset was collected manually from the Ns3a hiring application (نسعى - ns3a | منصة نسعى للتوظيف. n.d.) containing different fields including job title, company name, city, job field, required major, and required skills. The student dataset was also collected through a web-based questionnaire that was published to the students to collect data, containing different fields including student name, city, major, skills, and preferences. Therefore, the application deals with Arabic dataset consists of 69 jobs and 159 students. After the information is collected, it needs to be prepared for the data initialization phase of the data in the algorithm, so it is stored as comma-separated values (CSV) files. The CSV file contains different columns as mentioned above. Then all the students' and companies' information are stored in the database.

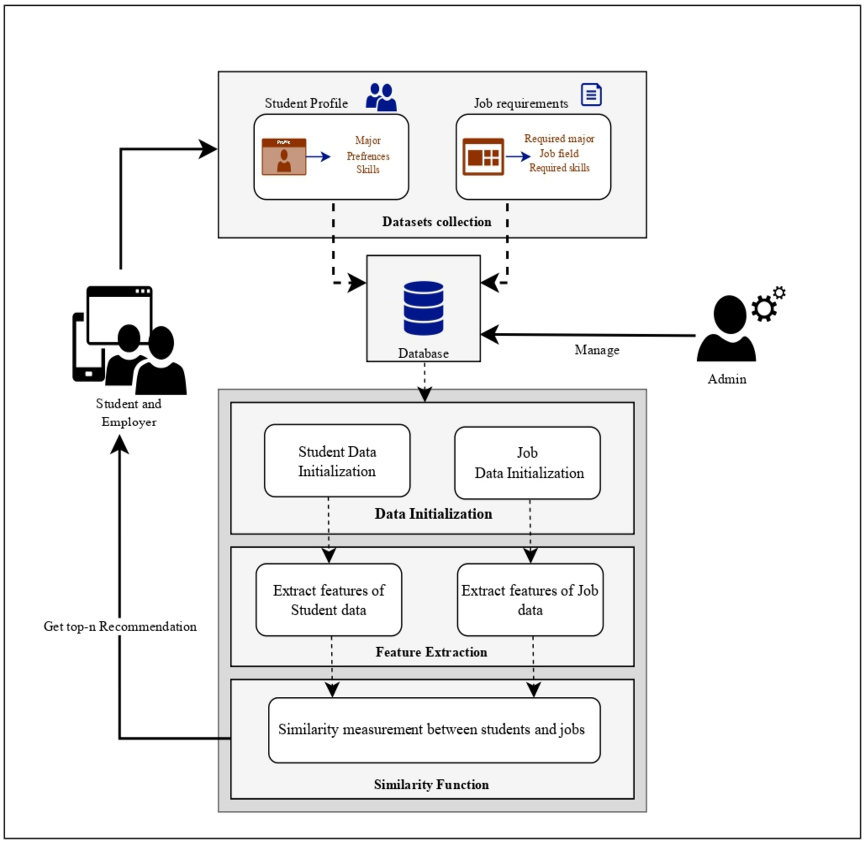

Figure 2: System architecture.

SYSTEM IMPLEMENTATION

Student-job application is based on achieving the best match between the student job seeker and the employer. To be more specific, the JRS mimics two things: (1) recommend suitable jobs for students based on their profiles, (2) recommend suitable students for the offered job based on the job requirements. Mpela and Zuva (2020) state that the aim of CBF involves matching the content of the user profile with the same content of the descriptions of the items. Therefore, the CBF approach is used in this application. The following subsections presents the process needed to demonstrate the RS implementation stage

Data Initializing

To initialize the data, the columns used in the recommendation process are selected from the database: form the student's table (Major, Preferences, Skills) and jobs (Job title, Job field, Required major, Required skills), then combined the selected columns of each table to create a corpus of text for each job and student.

The data in the corpus of text is cleaned up by removing the stop words and non-alphanumeric characters, then assigned the data of each corpus to the "Features" column to deal with the ML model for vectorized and similarity computations.

In the clean_text function, NLTK library is used to provide access to algorithms to complete a task, as well as access to tokenization, stop words and normalization packages. The two functions were called from the NLTK library are word_tokenize() function which is used to break the sentence into words, and the stopwords() function that remove stop words in Arabic and English languages. In addition, some common Arabic words were removed by using external text file "arabic_common_words_file".

Feature Extraction

Since the ML model deals with only data machine-readable, the vectorization process is the first step towards making text content machine-readable. There are many techniques that can be used to extract the features, in this project, two techniques were used by calling them from Scikit-Learn (Sklearn) library, each one is explained below. Sklearn is a ML library in Python that contains a lot of efficient tools for ML and statistical modeling, including feature extraction, similarity metrics, and clustering.

TF-IDF Vectorizer: The tfidfVectorizer() function is used for term weighting in information retrieval systems (Dhanya and Harish 2021), it converts a collection of raw documents to a matrix of TF-IDF features. TF measures how frequently a term occurs in a "Features" column of the jobs and student profiles, while IDF measures how important a term is according to the following formula:

$$\begin{aligned}
\mathrm{TF}(t) &= \frac{\text{Number of times term t appears in a document}}{\text{Total number of terms in the document}} \\
\mathrm{IDF}(t) &= \log\left(\frac{\text{Total number of document}}{\text{Number of documents with term in it}}\right) \\
\mathrm{TF}-\mathrm{IDF}(t) &= \mathrm{TF}(t) * \mathrm{IDF}(t)
\end{aligned}$$

This means the word with a higher TF-IDF score is the rarest word in each document and does not appear commonly in other documents and vice versa.

Count Vectorizer: The CountVectorizer() function calculates the weight of a word by counting how many times it appears in a "Features" column of the jobs and student profiles. It converts a collection of documents to a matrix of token counts by separating the text as a token and each token represented as an index, then creates a matrix of the token's indices against the repetition number of each token.

Similarity Metrics

JRS can use different measures of similarity measurements approaches. The cosine similarity is the most common measure used to measure the similarity between two vectors in information retrieval (Mpela and Zuva 2020). Since the cosine similarity is mostly used in content-based RSs (Diaby et al. 2014) and several JRS based on content (Mpela and Zuva 2020; Yadalam et al. 2020) have used cosine similarity to measure the similarity, and their results were satisfactory. Our experiments used the cosine similarity to measure the similarity by calculating the cosine of the angle between the student profile and the jobs vectors.

EXPERIMENTS AND RESULTS

In order to determine the best model, this project conducted three independent experiments to get the best top 10 recommendations: (1) CBF recommendation based on TF-IDF with K-Means clustering (2) CBF recommendation based on CountVectorizer() with K-Means clustering based on weighting. (3) CBF recommendation based on CountVectorizer() based on weighting. Since the evaluation metrics of the RS required a training dataset to obtain the accuracy score and our dataset is a testing dataset, we assess the relevance of our recommendations manually by selecting random users from the dataset.

First Experiment: the scores of cosine similarity between the student's profile and all the offered jobs results are arranged based on the top 10 similarities for a student's recommendation where the TF-IDF Vectorizer algorithm was applied to the column. To improve the order of the top-10 recommendations, the K-means clustering function called from the sklearn library to make the jobs that have the same cluster as the given student profile appear at the beginning of the list. K-means clustering algorithm partitions the jobs dataset into k different clusters such that each subset belongs to a group with similar characteristics (i.e., close to the same center point) (Al-Otaibi 2012).

Second Experiment: to improve the result, the "Skills" column of the student profile table and the "Required skills" column in the jobs table were divided into three columns: for student table (Specialized skills, General skills, Languages), and the jobs table (Required specialized skills, Required general skills, Required languages). The s_column_weight() and j_column_weight() functions give specific weight for each column used in the recommendation process according to the priority we need when listing the recommendations. After that, the CountVectorizer algorithm is used before applying cosine similarity to convert the given student information text to a vector instead of the

TF-IDF algorithm. Because in our case, the frequency of a word is important, and the TF-IDF algorithm as mentioned before, gives the words that are more present in the other documents less importance.

Third Experiment: this experiment is similar to experiment 2, except that we tried to use the K-means clustering algorithm to get the top-10 recommendation based on matching between the jobs cluster and the given student cluster.

The result of the first experiment did not have satisfying order of the recommendations list. That is why in the other experiments, we prioritized some columns to make the recommendation result based on them. The second and third experiments' results indicated that the order of recommendations was similar. However, in some cases, the results of the third experiment were worse due to the inaccuracy of the clustering. Therefore, the second experiment had the best results for the recommendations list than the other two experiments described above as it was adopted as the proposed CBF recommendation algorithm.

A cold start problem may occur in JRS for new users since the system cannot recognize their preferences and interests. To overcome this problem in our system, we ask the student when creating the account to enter his major and preferences so that the recommendation results depend on them when signing in for the first time. Once the profile is filled out, the recommendation results are based on it. The proposed algorithm was applied to the company side to recommend suitable students for the posted job. We assess the relevance of our recommendations manually by selecting random jobs from the dataset.

CONCLUSION

It is recommended that organizations adopt the latest technology, such as recommender systems (RS), to gain a competitive advantage in the recruitment and selection process. These techniques can help achieve the optimal match between job seekers and jobs. This project successfully developed an Android application that utilizes a Job Recommender System (JRS) based on a Content-Based Filtering (CBF) algorithm. The JRS connects students with startups and SMEs by matching their profiles with job requirements. Despite challenges related to dataset availability and the cold-start problem, the algorithm showcased convincing results. Future work includes further improving the algorithm by incorporating Collaborative Filtering (CF) and enhancing result filters for different operating systems. By embracing these advancements, organizations can enhance their recruitment and selection processes to effectively meet the needs of both job seekers and employers.

REFERENCES

Abia, M. and Brown, I. (2020). Conceptualizations of E-recruitment: A Literature Review and Analysis. In: pp. 370–379.

Adamopoulos, P. and Tuzhilin, A. (2014). On over-specialization and concentration bias of recommendations. In: *Proceedings of the 8th ACM Conference on Recommender systems*. New York, NY, USA: ACM, pp. 153–160.

B. Thorat, P., M. Goudar, R. and Barve, S. (2015). Survey on Collaborative Filtering, Content-based Filtering and Hybrid Recommendation System. *International Journal of Computer Applications*, 110(4), pp. 31–36.

Chen, Y., Wu, C., Xie, M. and Guo, X. (2011). Solving the Sparsity Problem in Recommender Systems Using Association Retrieval. *Journal of Computers*, 6(9).

Das, D., Sahoo, L. and Datta, S. (2017). A Survey on Recommendation System. *International Journal of Computer Applications*, 160(7), pp. 6–10.

Dhameliya, J. and Desai, N. (2019). Job Recommender Systems: A Survey. In: *2019 Innovations in Power and Advanced Computing Technologies (i-PACT)*. IEEE, pp. 1–5.

Dhanya, N. M. and Harish, U. C. (2021). Detection of Rumors in Tweets Using Machine Learning Techniques. In: pp. 3095–3111.

Diaby, M., Viennet, E. and Launay, T. (2014). Exploration of methodologies to improve job recommender systems on social networks. *Social Network Analysis and Mining*, 4(1), p. 227.

Gryncewicz, W., Zygała, R. and Pilch, A. (2023). AI in HRM: case study analysis. Preliminary research. *Procedia Computer Science*, 225, pp. 2351–2360.

Halder, S., Sarkar, A. M. J. and Lee, Y.-K. (2012). Movie Recommendation System Based on Movie Swarm. In: *2012 Second International Conference on Cloud and Green Computing*. IEEE, pp. 804–809.

Isinkaye, F. O., Folajimi, Y. O. and Ojokoh, B. A. (2015). Recommendation systems: Principles, methods and evaluation. *Egyptian Informatics Journal*, 16(3), pp. 261–273.

Mashayekhi, Y., Li, N., Kang, B., Lijffijt, J. and De Bie, T. (2022). A challenge-based survey of e-recruitment recommendation systems., 12 September 2022. Available from: https://arxiv.org/abs/2209.05112.

Mohamed, M. H., Khafagy, M. H. and Ibrahim, M. H. (2019). Recommender Systems Challenges and Solutions Survey. In: *2019 International Conference on Innovative Trends in Computer Engineering (ITCE)*. IEEE, pp. 149–155.

Mpela, M. D. and Zuva, T. (2020). A Mobile Proximity Job Employment Recommender System. In: *2020 International Conference on Artificial Intelligence, Big Data, Computing and Data Communication Systems (icABCD)*. IEEE, pp. 1–6.

Sarwar, B. M., Karypis, G., Konstan, J. A. and Riedl, J. T. *Application of Dimensionality Reduction in Recommender System-A Case Study*. Available from: www.cdnow.com.

Schafer, J. Ben. (2011). Application of Data Mining to Recommender Systems. In: *Encyclopedia of Data Warehousing and Mining*. IGI Global.

Shah, N., Michael, F. and Chalu, H. (2020). Conceptualizing Challenges to Electronic Human Resource Management (e-HRM) Adoption: A case of Small and Medium Enterprises (SMEs) in Tanzania. *Asian Journal of Business and Management*, 8(4).

T. Al-Otaibi, S. (2012). A survey of job recommender systems. *International Journal of the Physical Sciences*, 7(29).

Yadalam, T. V, Gowda, V. M., Kumar, V. S., Girish, D. and M, N. (2020). Career Recommendation Systems using Content based Filtering. In: *2020 5th International Conference on Communication and Electronics Systems (ICCES)*. IEEE, pp. 660–665.

Zangerle, E. and Bauer, C. (2022). Evaluating Recommender Systems: Survey and Framework. *ACM Computing Surveys*, 55(8).

Zhang, Y., Yang, C. and Niu, Z. (2014). A Research of Job Recommendation System Based on Collaborative Filtering. In: *2014 Seventh International Symposium on Computational Intelligence and Design*. IEEE, pp. 533–538.

نسعى - ns3a | منصة نسعى للتوظيف.