Hybrid job recommendation model based on professional profile using data from job boards and Machine Learning libraries

Alejandro HUAMÁN
Faculty of Engineering, Universidad Peruana de Ciencias Aplicadas
Lima, Villa, Perú

Giusen REBAZA
Faculty of Engineering, Universidad Peruana de Ciencias Aplicadas
Lima, Villa, Perú

Daniel SUBAUSTE
Faculty of Engineering, Universidad Peruana de Ciencias Aplicadas
Lima, Villa, Perú

ABSTRACT

This scientific article presents that the ideal job is crucial for undergraduate software engineering students because it is related to their mental and financial health. The proposal seeks the creation of a hybrid recommendation model (collaborative filter (CF) and content-based (CBR)) of jobs with the ability to obtain the percentage of similarity between the work and the profile obtained from the student. The difficulty of software engineering students finding a job that fits their profile greatly affects these young people over time, resulting in the resulting in the loss of large amounts of offers and unemployment. The system uses machine learning methods to recommend jobs, and in turn, the percentage of similarity between the profiles is obtained. To identify these student profiles, a professional orientation test validated by a psychologist and websites with experts in technical subjects such as Testlify and TestGorilla are used. An experimental protocol was created to evaluate the effectiveness of the model in the recommendations.

Keywords: professional orientation test, hybrid model, job, machine learning, profile, students, FC, CBR, similarity

1. INTRODUCTION

The ideal job is crucial for young university undergraduate software engineering students as it positively impacts their mental and financial health, reducing depression and anxiety, and improving self-esteem [6]. It also positively affects their physical health [11], leading to happiness in their daily lives. However, young software engineering students struggle to find suitable jobs due to the overwhelming number of job offers, many of which don't match their profiles [8]. Besides, the time spent sifting through these incompatible offers is detrimental [4]. Furthermore, employers prioritize professional skills, with 75 percent of job success based on skills and only 25 percent on technical knowledge [10]. Without aligning their professional profiles with job offers, students face prolonged job searches and potential unemployment.

The objective of this work is to provide a job recommendation model that has the ability to obtain a percentage of similarity for each job recommended with the student's profiles obtained using Machine Learning libraries and thus guarantee a good job search for undergraduate students belonging to the field of Software Engineering.

The collaborative filtering (CF) [2] and content-based (CBR) [1] recommendation models are effective but require additional methods for accuracy. Hybrid recommendation, a combination of both, aims to improve results. However, precise data sets are required for successful implementation, and other methods may include coders or natural language processing in the collaborative filter or content-based model.

The proposed recommendation system model addresses the issue of recommendation systems not adapting to user preferences due to lack of detailed information and simple techniques [4]. It uses data sets of skills, competencies, labor experience, work interests, and entrepreneurship of applicants, each with their own percentage of development. The hybrid technique, combining collaborative filtering and content-based methods, improves precision and diversity of favorable results, resulting in better job recommendations.

The key components of the research involve orientation tests with a psychologist and web platforms like Testlify and TestGorilla to obtain applicant profiles. The Surprise library is used for recommended uses, but multiple techniques are needed for development. Web scraping on Linkedin is used to obtain job offers.

The main contributions that follow are the following:

- A method is being implemented to recommend jobs based on the user's profile.
- Identification of a professional profile is provided through orientation tests validated by a psychologist and web platforms like Testlify and TestGorilla.
- A method is developed to extract job offers that meet the recommendation.

This scientific article is divided into the following sections: A review of works related to a job recommendation system, detailing methods and results, is in Section II. Then, its contributions, detailing the context and method of the proposed work in Section III, Besides, experiments with the model are shown, with protocol, results, and discussion included in Section IV. Finally, the article concludes with project conclusions and future prospects in Section V.

2. RELATED WORKS

Recent years have seen the development of several job recommendation systems, each with its own advantages and disadvantages. This section reviews articles similar to the model project, including their information, process, results, and differences in qualities or differences from the model project.

In [8],Kumar et al. (2022) propose a hybrid recommendation system for job recommendations using APIs and web crawling techniques. They filter companies based on key details like financing, donors, employees, status, and technology stack. Jobs are added to the system database for job description, and hybrid filtering is performed to define recommended work. The system is found to be more efficient than other models, reducing time by 28.16% and increasing the chances of users finding the right job by 103.44 percent. Both include hybrid recommendation for students, but the model proposed differs from the model of Kumar et al. (2022), which uses APIs and web crawling. Instead, the proposed model uses web scraping and data sets for each student's profile.

In [1], Alsaif et al. (2022) developed a machinelearning-based job recommendation system that uses natural language processing to match applicants' skills and experience with job descriptions on job posting platforms. The system extracts skills and experience from CVs, compares them with job descriptions, and recommends relevant work. The authors suggest that this system could enhance the efficiency and accuracy of existing recommender systems. Instead, the proposed model uses scientifically validated tests to create detailed student profiles, enabling more accurate recommendations, and uses Linkedin as the main job board.

In [5], Freire & de Castro (2021) conducted a systematic review of recommender systems in electronic recruitment, evaluating their types, information usage, and evaluation. The review, spanning 2012–2020, found that 7.03 percent of works are eligible, with collaborative and content-based filtering models being the most widely used. The authors propose a hybrid model that combines both filters to overcome their limitations.

In [2], Borges & Stefanidis (2022) propose "feature-blind fairness" to eliminate bias in collaborative filtering recommendations. They use a geometric model and framework to treat each element equally, aiming to reduce bias and precision. The results show a significant reduction in bias and acceptable precision. However, the proposed model uses a content-based model and a collaborative filtering model for higher accuracy, which is an improvement over Borges & Stefanidis' previous approach.

In [4], Chou & Lu (2020) propose a hybrid recommendation system for job websites in Taiwan to improve applicant accuracy and diversity. The system collects, classifies, and processes applicant data using JSS-Tree, numeric ratings, and human expressions. The authors estimate an accuracy of 80 percent in user preferences and recommend jobs similar to previous searches and those related to applicants' interests. The hybrid method differs from the proposed approach due to a feedback hybrid.

3. CONTRIBUTION

A. Preliminary Concepts

This section will introduce the main concepts of the solution. These definitions will help the proposal meet its objective, which is the development of a job recommendation model that has on each possible job a percentage of similarity with the student's profile that supports Software Engineering students at the UPC in finding a job that fits their professional profile (skills, competencies, labor experience, work interests, and entrepreneurship).

Definition 1 (Professional profile [3]): It is a set of skills, competencies, and attributes essential for job performance in a specific field. The model will group these competencies, skills, labor experience, work interests, and entrepreneurship to provide a grouped summary of student characteristics.

Definition 2 (Professional orientation test [7]): It is a tool used to help individuals identify their interests, skills, competencies, and values, enabling informed career or job decisions. It is used in a questionnaire to identify labor market skills. The model uses this tool to obtain students' professional profiles, which are then stored in data sets.

Definition 3 (Hybrid Recommendation [8]): It is a technique that combines content-based and collaborative filtering methods to create a true hybrid recommendation engine. This approach is used in the job recommendation model, which will be developed using a machine learning library.

Definition 4 (Collaborative Filter Recommendation [8]): It is a method in recommender systems that generates user recommendations based on similar tastes. It is used by a hybrid recommendation model with the help of a library.

Definition 5 (Content-Based Recommendation [8]): A machine learning technique that uses similarities between data properties to make recommendations. It is used by a hybrid recommender model using a library.

Definition 6 (Web Scraping [8]): Web Scraping is a technique for obtaining data from websites using automated tools. As a result, the job extraction procedure for Linkedin in the recommendation model is created with the help of web scraping.

Definition 7 (Machine Learning Libraries [9]): Essential tools for developers and data scientists to build, train, and use machine learning models. These libraries include algorithms for data processing, feature selection, model building, and performance evaluation. Popular Python packages include Scikit-learn, TensorFlow, Keras, PyTorch, and Theano. They are used to build hybrid job recommendation models.

Definition 8 (Job Boards [10]): Tools used by applicants and companies to find suitable job opportunities. Linkedin, for example, is a job board that includes information on user skills and work experience. The solution will use Linkedin to obtain Software Engineering job offers, ensuring the recommended job is existing and in demand.

B. Method

The main contributions that have been proposed in this model will be described in this section.

One of the important contributions is the use of professional orientation tests to identify the professional profile (skills, competencies, work experience, work interests, and entrepreneurship) of undergraduate Software Engineering students at the UPC in order to obtain their information, save it in data sets, and thus send it to the recommendation model.

Figure 1 shows the Test process provided to the user.

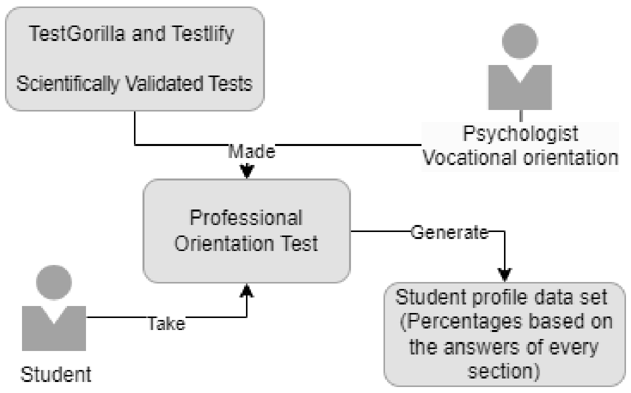

TestGorilla and Testlify
Scientifically Validated Tests
Psychologist Vocational orientation
Made
Professional Orientation Test
Generate
Student profile data set (Percentages based on the answers of every section)
Take
Student

Fig. 1. Professional orientation test flow.

A Software Engineering student takes a professional orientation test segmented into skills, competencies, work experience, interests, and entrepreneurship to form a profile. Each segment is made by a vocational guidance psychologist using TestGorilla and Testlify, which offer scientifically validated pre-employment tests designed by experts to evaluate candidates' skills, competencies, labor experience, work interests, and entrepreneurial spirit. After obtaining results, student profile data sets are created,which are determined by a percentage based on the answers during the test. The data sets store sections of professional test segments and rating percentages. This allows the model to accurately identify students' levels, providing a detailed profile and more accurate recommendations based on their performance.

The main important contribution that is used within the solution is the recommendation model for jobs provided to Software Engineering students according to their test result profile.

Figure 2 shows the job recommendation process using the Hybrid model, which is the combination of the Collaborative Filter and Content-Based methods.

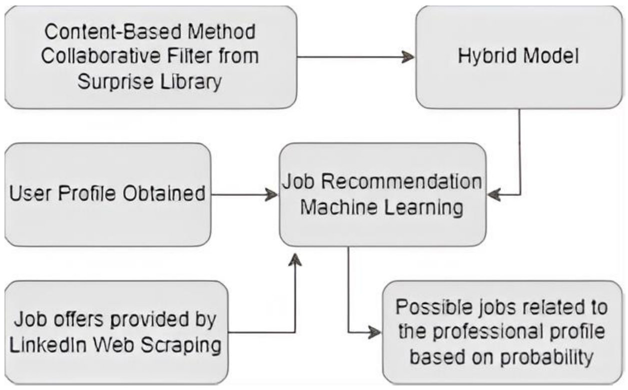

Content-Based Method Collaborative Filter from Surprise Library
Hybrid Model
User Profile Obtained
Job Recommendation Machine Learning
Job offers provided by LinkedIn Web Scraping
Possible jobs related to the professional profile based on probability

Fig. 2. Job recommendation flow.

The hybrid recommendation model, developed using the Surprise library and Python, uses web scraping on Linkedin to obtain Software Engineering job offers based on current labor demand. The model uses user profile data sets and job descriptions to generate results. Linkedin is a crucial source for employment due to its 10 years of experience, which has saved a lot of time and money for job seekers and organizations [1]. The model provides users with job results based on their relationship probability with their profile, which is calculated using a function that calculates similarity with sections and development percentages.

There are two data sets for student professional profiles: "Section data set" with SectionId and SectionName (name of the skill/competence or others within the tests), and "Ratings data set" with ResultTestId, SectionId, and DevelopmentPercentage (result of the section within the tests). Figure 3 shows a flowchart about how the model arranges the jobs retrieved during web scraping using public links provided by Linkedin. Every link is related to a profile in the software engineering field. In order to gather data, an algorithm uses a link to build a spider to carry out the extraction of data for each job result that appears on the page. The spider extracts the name, URL, description, location, date, and company. For every profile, a CSV is generated, which is eventually concatenated and saved as a single CSV file.

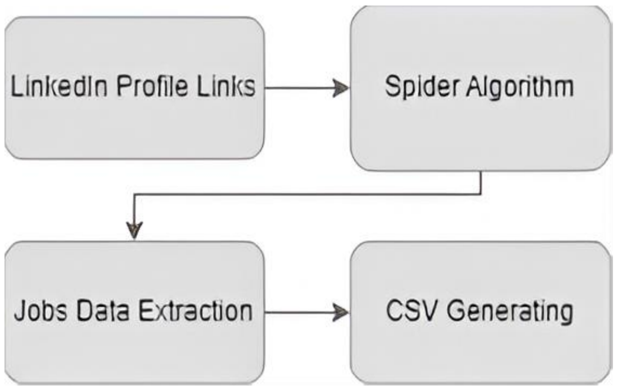


Fig. 3. Extract the important data from Linkedin

Listing 1 shows the algorithm for the three important data sets being read: section, ratings section and jobs (extracted by the Web scraping on Linkedin).Then, a combination is made in a dataframe with the 3 data sets and saved in the variable "merged df".

```
jobs_df = pd.read_csv('jobs.csv',sep='\t'
    )
ratings_df = pd.read_csv('ratings_section
    .csv')
sections_df = pd.read_csv('section.csv')
all_combinations = pd.MultiIndex.
    from_product([sections_df['SectionId'
    ], jobs_df['JobId']], names=['
    SectionId', 'JobId'])
all_combinations_df = pd.DataFrame(index=
    all_combinations).reset_index()
merged_df = all_combinations_df.merge(
    ratings_df, on='SectionId', how='left
    ')
merged_df = merged_df.merge(sections_df,
    left_on='SectionId',right_on='
    SectionId', how='left')
merged_df = merged_df.merge(jobs_df,
    left_on='JobId', right_on='JobId',
    how='left')
merged_df['DevelopmentPercentage'].fillna
    (0, inplace=True)
```

Listing 1. Extract the important data sets for the model

In Listing 2, the data is obtained to enter it into the models, these being the columns: 'sectionname', 'Description' (Job description) and 'developmentPercentage'. Then, the collaborative filter (KNNBasic) and content-based (SVD) recommendation models are created with the Surprise library.

```
reader = Reader(rating_scale=(1, 5))
data = Dataset.load_from_df(merged_df[['
    SectionName', 'Description', '
    DevelopmentPercentage']], reader)
trainset, testset = train_test_split(data
    ,test_size=0.2,  random_state=42)
knn_model = KNNBasic(sim_options={'name':
     'cosine','user_based': False})
knn_model.fit(trainset)
content_model = SVD()
content_model.fit(trainset)
predictions = []
max_rating = merged_df['
    DevelopmentPercentage'].max()
min_rating = merged_df['
    DevelopmentPercentage'].min()
```

Listing 2. Collaborative filter and Content based models

With both models, the Listing 3 shows the make of the first predictions with them of the section name with its development percentage within the job description. After obtaining this first prediction from both models, now it begins to enter the calculate similarity function and this prediction number of the models is entered as the new development percentage for the section names. Once the calculate similarity has been obtained for both models, the hybrid similarity begins to be obtained, which is the sum of the similarity results of both models and divide it by 2. Then, once we have the hybrid similarity prediction, it is stored within a predictions list along with the section name, job description and development percentage.

```
for test_section, test_description,
    test_rating in testset:
    knn_pred = knn_model.predict(
        test_section,test_description,
        test_rating).est
    content_pred = content_model.predict(
        test_section, test_description,
        test_rating).est
    similarity_pred_content =
        calculate_similarity(test_section
        ,test_description,content_pred)
    similarity_pred_knn =
        calculate_similarity(test_section
        ,test_description,knn_pred)
    similarity_hybrid_pred = min((
        similarity_pred_content +
        similarity_pred_knn) / 2, 1.0)
    similarity_hybrid_pred = round(
        similarity_hybrid_pred, 1)
    section_rating = merged_df.loc[
        merged_df['SectionName'] ==
        test_section,'
        DevelopmentPercentage'].iloc[0]
    normalized_rating = (section_rating -
         min_rating) / (max_rating -
        min_rating)
    similarity_hybrid_pred *=
        normalized_rating
    predictions.append((test_section,
        test_description, test_rating,
        similarity_hybrid_pred))
```

Listing 3. Hybrid process with the two models

In Listing 4, the calculate similarity function uses the three characteristics (section name, job description and development percentage) and calculate on the job description the section with his development percentage using the TfidfVectorizer class to be more accurate on the similarity.

```

def calculate_similarity(test_section,
    test_description,rating):
    if test_section is None or pd.isnull(
        test_description):
       return 0
    tfidf_vectorizer = TfidfVectorizer()
    tfidf_matrix = tfidf_vectorizer.
        fit_transform([test_section,
        test_description])
    similarity = (tfidf_matrix *
        tfidf_matrix.T).A[0, 1] * rating
    return similarity
```

Listing 4. Similarity function

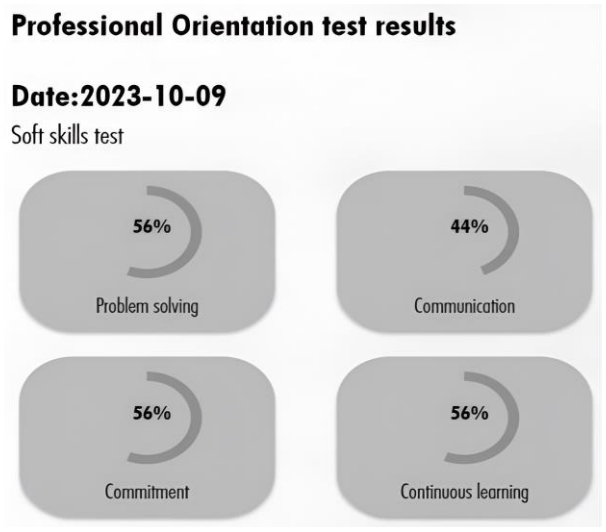
Finally, in Listing 5, the "recommendations" dataframe is created that will store the union of all the columns of the "merged df" dataframe and the hybrid similarity calculated previously stored in "df predictions" dataframe. In addition, the duplicates are eliminated and ordered from highest to lowest based on the hybrid similarity and those with 0.0 are eliminated.

```
df_predictions = pd.DataFrame(predictions
    , columns=['SectionName', '
    Description', 'DevelopmentPercentage'
    ,'Similarity'])
recommendations = merged_df[['JobId', '
    JobName', 'URL', 'Location', 'Date',
    'Company','Description']].merge(
    df_predictions, on='Description')
recommendations = recommendations.
    sort_values('Similarity', ascending=
    False)[['JobName','Description','URL'
    , 'Location','Date', 'Company', '
    Similarity']]
recommendations = recommendations.
    drop_duplicates(subset=['JobName'])
recommendations = recommendations.loc[
    recommendations['Similarity'] != 0.0]
```

Listing 5. Show results

The implementation of the model on the JobBrainProfile system is shown.

In the first place, there is Figure 4, which provides the results of the skill tests carried out by the student with their respective percentage of development.



Professional Orientation test results

Date:2023-10-09

Soft skills test

56% Problem solving

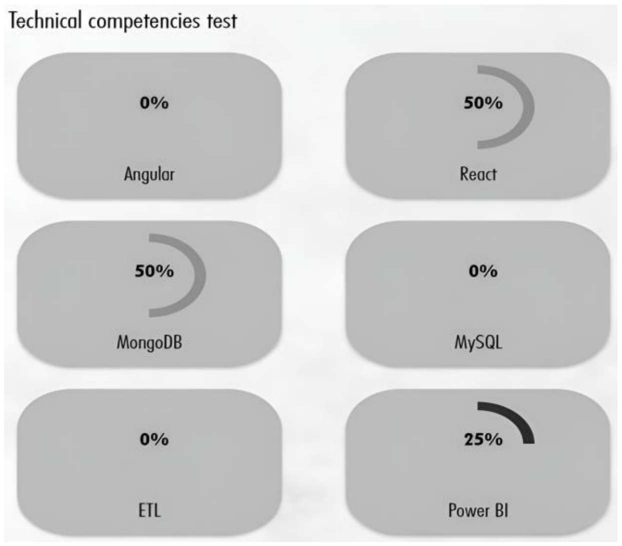
44% Communication

56% Commitment

56% Continuous learning

Fig. 4. Skills results of the student professional profile

Figure 5 provides the results of the competencies tests carried out by the student with their respective percentages of development.



Technical competencies test

0% Angular

50% React

50% MongoDB

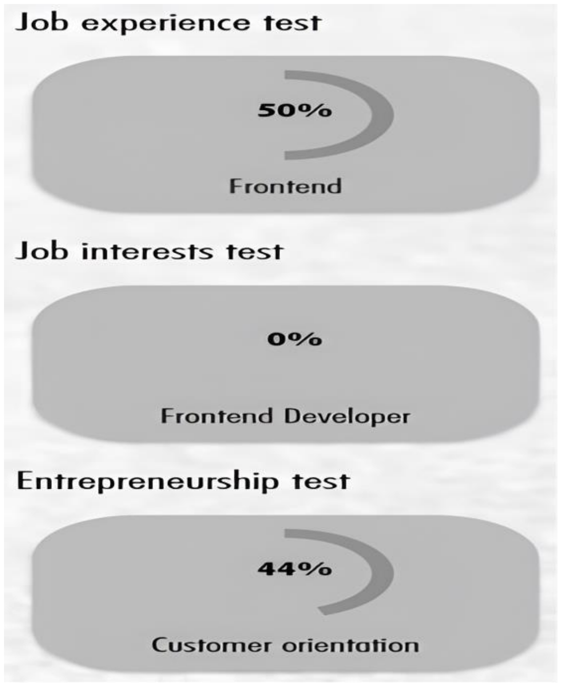
0% MySQL

0% ETL

25% Power BI

Fig. 5. Competencies results of the student professional profile

Figure 6 shows the results of the areas of experience, work interests and entrepreneurship tests carried out by the student with their respective percentages of development.



Job experience test

50% Frontend

Job interests test

0% Frontend Developer

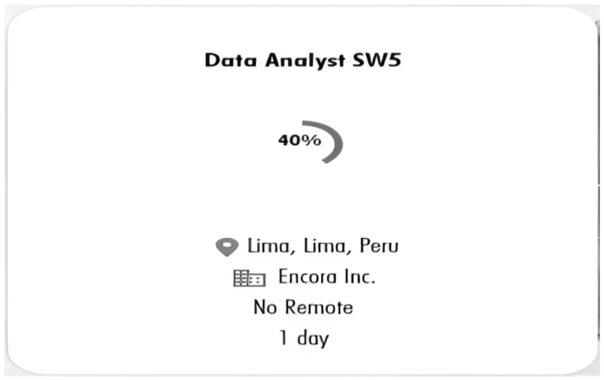
Entrepreneurship test

44% Customer orientation

Fig. 6. Experience, interests and entrepreneurship results of the student professional profile

The percentage is calculated thanks to the internal score that has every option inside the tests. With this, the two data sets are created.

For the main contribution, which is the recommendation of jobs, there is Figure 7, where the job recommendation model is started and the list of possible recommended jobs, stored in the recommendations dataframe obtained from the model, related to the student's profile is shown. The percentage of relationship probability, that is, the hybrid similarity prediction, with the student profile that appeared for each job recommended indicates how much relationship the student profile of the tests along with its development has with the description of each recommended job.



Data Analyst SW5

40%

Lima, Lima, Peru

Encora Inc.

No Remote

1 day

Fig. 7. Recommendation possible jobs related with the student profile

4. EXPERIMENTS

In this section, the protocol that is used for the development of the experiments, the results of said experiments and a detailed discussion about the obtained results are going to be presented.

A. Experimental Protocol

The experiment requires the deployment of all solution components, including AWS services like AWS S3 for front-end, AWS ElasticBeanstalk for back-end, and AWS RDS for database, to work together. Administrators can access the application via a web browser. During the experiment, there were 389 software engineering undergraduates belonging to cycle 8 to 10 and mostly aged 18 to 30 at the UPC. Therefore, this system was validated with 38 people. Besides, the five profiles that are used in the system validation are Frontend Developer, Backend Developer, Fullstack Developer, Mobile Developer, and Data Engineer.

Administrators conduct a system test, dividing it into sections: using the system to obtain job similarity results, comparing the recommendation engine's time, and comparing it with related work results. They spend 10-12 minutes on professional tests, then check job recommendations, evaluate their relevance to the profile, and assess the time it takes for the engine to show results. They also provide their conclusions and feedback on the test.

B. Results

In this section, what was stated in the experimentation protocol is carried out. A case validation was performed by a UPC undergraduate software engineering student who is looking for work in the labor field. His results are the following:

TABLE I
TECHNICAL SKILLS TEST

Technical Skill	Result
React	25%
Vue.js	50%
Excel	25%
Spring Boot	25%
Swift	25%
MySQL	50%
Kotlin	33%
.Net	25%
MongoDB	25%
Django	100%
Flutter	0%
PowerBi	25%
ETL	100%
Angular	0%

TABLE II
WORK EXPERIENCE TEST

Work experience	Result
Frontend	83%
Backend	50%
Android	50%
IOS	0%
Data engineer	83%
Fullstack	50%

TABLE III
JOB INTEREST TEST

Job interest	Result
Frontend	100%
Backend Developer	50%
Mobile Developer	75%
Data engineer	100%
Fullstack Developer	100%

TABLE IV
RESULT RELATED JOBS OF THE EXPERIMENT

Job	Location	Company	Percentage
Backend Developer	La Molina,Lima,Peru	Yape	40%
Full Stack Developer	Lima,Peru	Grupo Lucky	30%
Backend Engineer (Javascript)	Lima,Peru	ATekton Labs	30%
Power Bi Analyst Developer	Lima,Peru	GloboKas Peru S.A	30%
FullStack Developer (React,Java)	,San Borja ,Lima,Peru	CSTI	30%

TABLE V
TIME DURATION WITH THE RELATED WORKS

	Time of duration (S)
Hybrid Recomendation Proposed (Test 1)	16.23
Kumar et al. (Processed job description) (Test 1)	20.87

TABLE VI
PRECISION OF RESULTS WITH THE RELATED WORKS

	Precision of Results
Hybrid Recomendation Proposed (Test 1)	0.40
Kumar et al. (Processed job description) (Frontend Test 1)	0.124
Chou Lu (First job) (Test 1)	1.250
Alsaif et al. (First job) (Test 1)	0.222

C. Discussion

The recommendation for backend, frontend, fullstack and data engineer jobs is shown by the job interests in Table III and work experiences in Table II. Besides, according in Table I, the technical skills in Vue.js for fullstack, Django and MySQL in backend and ETL in data engineer.

In Table IV, the hybrid recommendation model, based more on guiding job paths based on experience and interest, shows the result of a Backend Developer position with a 40%.

In Table V, the model duration of [8] is 20.87 seconds. The authors carry out processing in the job description for Frontend Test 1 and comparing it with the proposed model that does not apply that method, a duration of 16.23 seconds is obtained. It is verified that the proposed model has a shorter duration, however, the model of [8] when applying its mentioned method obtains much more precise results, which generates greater confidence. In conclusion, in terms of time, it can be said that the proposed system is efficient.

Finally, in Table VI, the proposed model shows a 40% accuracy rate, which is higher compared to the other models' 10-22% accuracy. However, [4] produces outcomes for occupations with a 1.25 accuracy rate, which is higher than the model, when expressed as percentages. Therefore, compared to initial model results from [8] and [1], the early outcomes of the experiment are more acurrate. In contrast, there is still room for improvement in accuracy and aligning the results more closely with the profile.

5. CONCLUSIONS AND PERSPECTIVES

The hybrid model of job recommendation based on the professional profile offers significant contributions, such as the relationship with the user's profile through a method within the algorithm, personalized recommendations with similarity percentages, and the use of questionnaires approved by psychologists and experts from Testlify and TestGorilla. In addition, a method was developed to extract job offers from LinkedIn without the need for an account, using a public URL with a location filter in Lima. However, it is crucial to recognize its limitations, such as the possible lack of job offers when trying to relate them to the user, the possible confusion with technical descriptions in the results by the user due to their greater focus on their interests alongside their experience, and the temporary availability of job offers.

About potential added value, the proposal integrates psychology with technology and software engineering to address employment challenges (job inadequacy and unemployment), as well as promoting and prioritizing academic and professional development so that undergraduate software engineering students have future well-being.

Four perspectives are being considered for the solution, every recommended job can include the sections that contributed to its recommendation, The recommendation process should incorporate additional factors for greater precision, such as processing job descriptions, as suggested by [8],new job boards can be added through web scraping to increase the number of jobs and create new sections for profiles in other areas. With this knowledge, there is still interest in improving and expanding the model to offer better results to students.

REFERENCES

[1] S. A. Alsaif, M. Sassi Hidri, H. A. Eleraky, I. Ferjani, and R. Amami, **Learning-Based Matched Representation System for Job Recommendation**, Computers, 2022.

[2] R. Borges and K. Stefanidis, **Feature-blind fairness in collaborative filtering recommender systems**, Knowl Inf Syst, 2022.

[3] J. Bronstein and O. Nebenzahl, **Developing scales for identifying and classifying library and information science skills and competencies: An Israeli perspective**, Journal of Librarianship and Information Science, 2020.

[4] C. L. Chou and T. Y. Lu,**A hybrid-feedback recommender system for employment websites**, Journal of Ambient Intelligence and Humanized Computing, 2020.

[5] M. N. Freire and L. N. de Castro,**e-Recruitment recommender systems: a systematic review**, Knowledge and Information Systems, 2021.

[6] K.-Y. Kao, H.-T. Lee, A. Rogers, H.-H. Hsu, and M.-T. Lin,**Mentoring and Job Search Behaviors: A Moderated Mediation Model of Job Search Self-Efficacy**, Journal of Career Development, 2021.

[7] I. Khaouja, I. Kassou, and M. Ghogho,**A Survey on Skill Identification from Online Job Ads**, IEEE Access, 2021.

[8] N. Kumar, M. Gupta, D. Sharma, and I. Ofori,**Technical Job Recommendation System Using APIs and Web Crawling**, Computational Intelligence and Neuroscience, 2022.

[9] S. Raschka, J. T. Patterson, and C. J. Nolet,**Machine Learning in Python: Main Developments and Technology Trends in Data Science**, Machine Learning, and Artificial Intelligence, Information, 2020.

[10] M. C. Urdaneta-ponte, A. Méndez-zorrilla, and I. Oleagordia-ruiz, **Lifelong learning courses recommendation system to improve professional skills using ontology and machine learning**, Applied Sciences (Switzerland), 2021.

[11] E. A. J. van Hooft, G. Van Hoye, and S. M. van den Hee,**How to Optimize the Job Search Process: Development and Validation of the Job Search Quality Scale**, Journal of Career Assessment, 2022.