E-ISSN 2586-9027
Homepage : https://tci-thaijo.org/index.php/SciTechAsia
Science & Technology Asia
Vol.31 No.3 July - September 2026
Page: [147-159]
Original research article

The Development a Job Position Recommendation System Using Collaborative Filtering Technique

Natratanon Kanraweekultana, Jintana Polsri, Phatthira Keawkerd, Poorivat Kampeerapaappat, Rungthip Cobal*

Information Technology and Digital Business Program, Faculty of Business Administration, Rajamangala University of Technology Krungthep, Bangkok 10120, Thailand

Received 30 July 2025; Received in revised form 7 June 2026
Accepted 20 July 2026; Available online 30 September 2026

ABSTRACT

Unemployed individuals and recent graduates often lack the experience or skills necessary to effectively search and filter job positions on job search websites, hindering their ability to access job listings and requirements. While traditional job search platforms rely heavily on manual keyword filtering, which often fails to provide personalized guidance for users with limited professional profiles, there is a critical need for a more intelligent matching approach. To address this issue, a research process was conducted involving extracting job information from the top job search website in Thailand. The data was then prepared, transformed, and subjected to Matrix Factorization. The data was randomly split into 80% for training and 20% for testing, out of a total of 30,000 records. This data was used to test the performance of a recommendation system utilizing the Collaborative Filtering technique. The Learning Curve, which measures the efficiency of model learning through Training Loss and Validation Loss, showed a continuous decrease in value until it reached its lowest validation loss of 0.0442, indicating a well-fitted model. This research suggests that the model can be effectively used for predicting and creating a prototype of a job recommendation system that connects job opportunities via API to efficiently meet the needs of job seekers and recent graduates, ultimately bridging the gap between inexperienced applicants and complex job market demands to enhance overall employment accessibility. Furthermore, prototype testing confirmed a successful real-world user experience, demonstrating that the system can quickly recommend the top 10 relevant jobs tailored to individual needs.

Keywords: Autoencoder; Collaborative filtering; Job position; Matrix factorization; Recommendation system

***Corresponding author:** rungthip.c@mail.rmutk.ac.th doi.org/10.70730/sta.v31i3.255265

1. Introduction

Currently, guidelines for selecting job positions include a variety of factors such as compensation, job descriptions, benefits, popularity, company image, etc. These require experience in surveying and searching for job positions that will help job seekers learn and lead to decision-making [1]. Traditional job search platforms rely heavily on manual keyword filtering, which often fails to provide personalized guidance for users with limited professional profiles. Using keyword search methods to search for jobs may not result in the exact jobs. Other job positions with different names but the same meaning may not show in the results. This specific gap within the context of the Thai job market, where diverse job titles and language inconsistencies frequently occur, clearly illustrates the critical necessity for developing a more intelligent job recommendation system. Creating algorithms that can learn from data and can predict data, relies on learning from a model of the input data. This allows computers to learn, understand, and predict data. Most recommendation systems use machine learning techniques for decision support systems that recommend products or services appropriate to each customer's style and behavior. It relies on user data combined with external data for the analysis and filtering to obtain something that is appropriate for the user. This type of recommendation system is used by many websites and applications such as Amazon.com and Netflix. It comprises three main algorithms: 1) Collaborative Filtering, 2) Content-based Filtering, and 3) Hybrid Filtering. Each of these recommendation system techniques helps a recommendation system to have good performance and meet the needs of users [2]. In developing a job recommendation system, the user-based data filtering technique relies on information about user satisfaction with various items. For example, this technique can be used for book recommendations by using information from other users to help predict what the user will like. Previously, the user had used or collected information about the categories of interest. The amount of time spent reading each article was recorded to increase the accuracy of a model to help increase the efficiency of the recommendation system.

Therefore, this study aimed to develop a job position recommendation model. Recommendation techniques are used to model job recommendations based on the current popularity based on the number of visitors or the number of people interested at that time. Thus, users can use the model created for job searches that meets their needs which reduces the problems of new jobseekers who lack experience in job screening.

2. Related Work

Recommendation systems are becoming increasingly popular due to complexity of the issue. The use of recommendation systems has been studied and discussed in the context of a learning business applied tool.

2.1 Classification of recommendation systems The research categorizes recommendation systems into main three groups which are 1) popularity-based recommendation systems, 2) content-based recommendation systems, and 3) collaborative filtering-based recommendation systems. One way to optimize the recommendation system is machine learning using collaborative filtering, content-based filtering and hybrid filtering. Using machine learning increases the accuracy and decreases the complexity of the calculating recommendation process.

2.2 Popularity-based recommendation systems

The system utilizing machine learning in searching and filtering job

positions efficiently meet individual needs, as demonstrated in the research of Jobaer et al. [4] who studied recommendation systems that are based on popularity. The technique aims to provide general suggestions for every user based on the popularity.

2.3 Content-based recommendation systems

As seen the agreeable direction in Gupta, A., Singh, N., & Hossain' [10] work who studied Content-Based Filtering Job Recommendation System using Machine Learning, in this study, it improves job suggestion system by using learning model from Android application and using Kotlin, Jetpack Compose, Ktor, FastAPI, and MongoDB.

2.4 Collaborative filtering-based recommenddation systems

The crucial limitation of collaborative filtering-based recommendation systems for general users is relevance issues, in which the research proposes a method to simplify the complexity of the recommendation process. As illustrated in the research of Khademizadeh et al. [5], utilizing the data mining technology, library management system and data center are important due to the huge amount of information. Data is analyzed in the collaborative filtering method based on the list and the tree decision method, leading to the rule of book suggestion for users. This execution of analysis can be reused for solving various kinds of problems. Autoencoder-based collaborative filtering [8] research applies recommendation systems via the collaborative filtering method with autoencoder that pre-training and stacking skills. The experimental result and movie data set, MovieLens, illustrates that this a potential and efficient method, the identical technique used by Yu et al. [9]. However, displacement is a hard-to-handle problem in collaborative filtering-based recommendation systems which use an algorithm based on stacked autoencoder (MCFSAE) for solving displacement in the MCFSAE model. The first step is converting a scoring matrix into a high-dimensional classification dataset to be the same size as the number of ratings. Hence, they decided to use stacked autoencoder which is a nonlinear feature reduction model to continuously demonstrate the characteristic. The study proposes an additional way to utilize job recommendation using a collaborative filtering algorithm, which is increasing the scope of machine learning models.

2.5 Hybrid approach

Moreover, the research also suggests the algorithm development that integrates content-based recommendation systems and collaborative filtering-based recommendation systems by using machine learning technology to improve the efficiency of the recommendation process for users who share similar characteristics. The referred system can be applied to diverse business models, both with and without machine learning [3]. Mishra, R., Rathi, S. [13] showed that content-based recommendation systems and collaborative filtering-based recommendation systems can use hybrid filtering combined with the characteristics of both systems. There is research that demonstrates the combination between these two systems for job recommendation system in Yang,S., Korayem,M., AlJadda, K. [14]. The study proposes the direct way of combining the two systems via the ability to express probabilistic relationships between directly related object attributes. Jain, H. and Kakkar, M. [12] developed a job recommendation system using machine learning on the large scale dataset which required a lot of time to process and create recommendations. Moreover, it uses knowledge models from data sources to make appropriate recommendations. Suharyadi, J., and Kusnadi, A. [11],

designed and developed a job recommendation system based on two dominants on psychotest results using the KNN algorithm to measure the shortest distance between train and test data for creating job recommendations. The referred data training model is brought from the expert and books reference. The system is tested by giving users psychological tests and questionnaires regarding satisfaction with the use of the system. The results, based on response from users, show that the satisfaction rate of the system is as high as 85%; they state that the system can provide recommendations that are consistent with the user's psychological test results. Shrivastava, & Gupta [6] proposed a system that predicts and diagnoses illness using the Flask API for extracting data from online sources, similar to Abodayeh et al [7], whose research used web scraping as a crucial apparatus for extracting data from a huge data application. There are many ways of using web scraping according to Python's Beautiful Soup Library. The referred library uses web scraping to improve its execution-analysis phrase before entering the technique-applied phrase.

2.6 Conclusion of conventional methods

Accordingly, this study decided to use a learning model that combines content-based recommendation systems and collaborative filtering-based recommendation systems, because the results are going to be based on the shared behavior and characteristics of users. The study converts data into a matrix by collecting at least 3 datasets: job position, job position review scores and user's information. Data is collected by web scraping a job search website. There will be a minimum of 50,000 records to generate the model that reaches user's needs. Conducting a study of related research for modeling using autoencoding model training will help the model know the characteristics of users and predict their behavior, and most of the results from research studies will be applied using the collaborative filtering technique as shown in Table 1.

Table 1. Summary of the Related Studies' Recommendation System.

Research	Year	Technique		
		Content Based Filtering	Collaborative Filtering	Hybrid Approach
Hafizovic, N. [3]	2019			✓
Al Jobaer, A., Sanchi, B. B., Javed, A., Islam, M. S., & Jameel, A. S. M. [4]	2021		✓	
Khademizadeh, S., Nematollahi, Z., & Danesh, F. [5]	2022		✓	
Shrivastava, N., & Gupta, S. [6]	2021	✓	✓	✓
Ouyang,Y., Liu, W., Rong,W., & Xiong, Z. [8]	2014		✓	
M Yu, Quan, Peng, X Yu & Liu [9]	2022		✓	✓
Gupta, A., Singh, N., & Hossain, A. M. I. [10]	2023	✓		
Suharyadi, J., and Kusnadi, A. [11]	2019			✓
Jain, H. and Kakkar, M. [12]	2019	✓	✓	✓
Mishra, R., Rathi, S. [13]	2020	✓	✓	
Yang,S., Korayem,M., AlJadda, K., Grainger, T., Natarajan, S. [14]	2017	✓	✓	
Mikhaylov, A., Bhatti, I. M., Dinçer, H., & Yüksel, S. [15]	2024		✓	
Shokrzadeh, Z., Feizi-Derakhshi, M. R., Balafar, M. A., & Mohasefi, J. B [17]	2024		✓	
Summary		5	**10**	5

3. Methodology

Data analysis of the development a job position recommendation system using collaborative filtering is in accordance with the research framework. The research processes are as follows:

3.1 Problem definition

Data analysis for a job position recommendation system is going to be an assistant and help individuals who are looking for a job that fits their skills and interests. Because of the huge amount of data about job positions combined with the lack of suitable filter and recommendation systems and, moreover, the incompatibility between an individual's skill and the market's needs, all these factors lead to the displacement and disorder of the data, causing inconvenience for analysis and utilization that requires experience and expertise. Therefore, tools and techniques that are well-suited to developing the prototype are recommendation systems using the collaborative filtering technique. [15, 17]

3.2 Data collection

In the process of data collection, the prototype development depends on a dataset that is extracted by scraping a job-search website. The dataset is categorized in the three following steps:

1) **Language Filtering** Due to the variety of languages in the data (English, Thai, and Chinese) we need to translate all the data into English to make it convenient for creating the model.

2) **Message Filtering** There will be text type data with a variety of positions. And there are some duplicate messages. which cannot be used to create a model. Therefore, duplicate information must be filtered out.

3) **Number Filtering** Collecting data on the number of visitors to the website for that job position from 0 or more will make the data too large. Therefore, the data has been filtered using a scoring format, such as job positions with 0 – 100 visitors equal to 1 point, 101 – 200 equals to 2 points.

Converting these raw visitor counts into a standardized scoring format significantly improves the model's processing efficiency by mitigating data sparsity, reducing computational complexity, and preventing extreme outlier values from distorting the weight updates during the matrix factorization and model training phases.

3.3 Data preparation

Data selection comes from the process of creating models using the collaborative filtering technique. The data is in the form of a CSV file in order to perform matrix factorization and then convert the matrix of user purchasing scores for products or services into smaller matrices and develop a model.

The data used has 4 attributes: 1) Position is the data that will be used to create a job recommendation system model that are similar or have high-scoring job positions, 2) Company is the name of company accepting applications, 3) Rating is information that has been filtered from the number of visitors so that the size of the data is not too large, and 4) User is information that has been created to provide a sample of users who have rated job positions. The data filtering for the collaborative filtering method is as follows: 1) Users: Each user must provide job ratings no less than 30 times. and 2) each job position must have users rating it no less than 30 times [16]. The data will be used to create a model, as detailed in Table 2.

Table 2. Data attribute details.

Attribute	Type of Data	Descriptive
Position	text	Job positions posted on job search websites
Company	text	Name of company accepting applications
Rating	int	Rating (filtered by number of visitors)

Attribute	Type of Data	Descriptive
Position	text	Job positions posted on job search websites
Company	text	Name of company accepting applications
User	int	Number of users (example)

The data were randomly split, 80% for training and 20% for testing.

3.4 Modeling

To address the limitations of existing systems mentioned in Section 3.1 and 3.2 a creative adaptation to the standard collaborative filtering workflow is implemented because conventional job boards are troubled by data disorder, multi-language inconsistencies and extreme data sparsity due to unscaled visitor numbers. Traditional recommendation systems usually work directly on raw user-item matrices, which fails when confronting highly disordered and duplicated web-scraped data. Our proposed architecture is different from the standard one as shown in section "Data Analytics & Data Processing" of Fig. 1, where we add a multi-stage filtering pipeline (Language, Message, and Number filtering) and a dense matrix normalization step before model training. This structural change ensures that only high-quality structured interaction data is fed into the neural network, effectively removing the displacement and lack of fit in existing systems

Model creation uses the autoencoder train model to receive job rating data for 30,000 positions (dataset Size) for each user (Users), using mean squared error as the loss function, and using Sigmoid Activation Function to convert the Output Data to be in the range of 0. -1 by Encoder [18] The original data is converted into data with fewer dimensions, called latent representation. This data will be stored in the latent space, and then the decoder will convert the data in the latent space back to the original data. During the training of the autoencoder model, there will be loss between the input and output data of the model. The model learns to reduce this loss by improving the model parameters. [19]

3.5 Evaluation

The performance of the model using the learning curve is a representation of the learning performance of the model from the training dataset. The x-axis of the graph will be the epoch, and the y-axis will be the model's performance. The model's performance will be measured after improvements in weight and bias. The loss values obtained in training loss and validation loss continually decrease until a certain point, at which they will stabilize. The learning curve model shows that it is a good-fitting model or a model with good learning that can be used to predict new, unseen data by obtaining the value from the mean squared error. (MSE) is a loss function measuring the difference between the predicted value and the actual value. Epoch 100 has the lowest val_loss value. [21]

3.6 Deployment

A prototype was developed by creating an API connected to the database that collects the job position information. Then use the model to develop an API [22] to connect to the database. The model can predict and recommend job positions to users. The model is connected to the database and predicts 10 job positions by using the Jupyter notebook program, Visual Studio Code program, and SQL Server Management Studio as tools to develop the system and activate it. [23-25] The research process can be summarized as shown in Fig. 1.

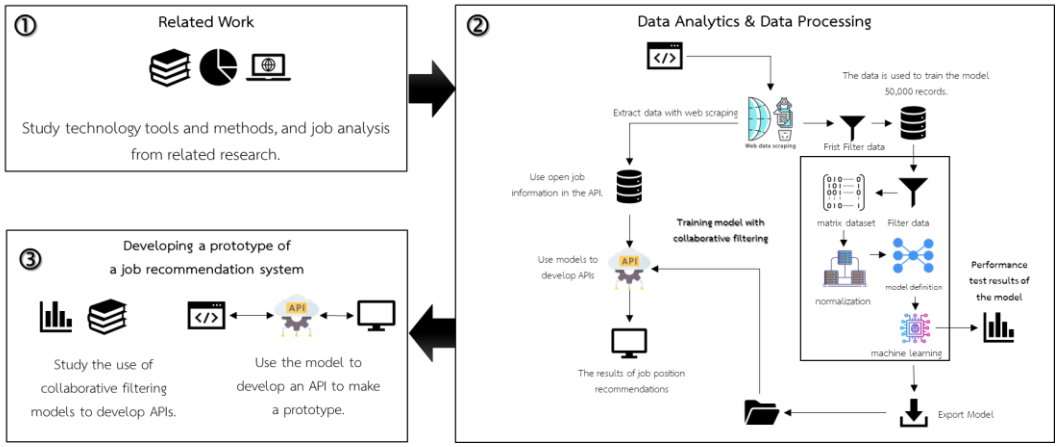


Fig. 1. Methodology framework.

4. Results

The researchers collected data for a period of 4 months. The most complete data will be 30,000 sets, including job positions (positions), scores (ratings), users (users) stored in the .CSV file. In the learning process of the model, x_train collects the data for training, x_val collects the testing data, and in the Y direction, y_val collects the testing data. In the Y direction, test_size = 0.2, since the data were divided into 80% for training and 20% for testing. Random_state = 1 value has a constant data set. This means that every time the code is run with the random_state value of 1, it will create a dataset. Then the matrix is converted in the form of a Data Frame into a multidimensional array. The NumPy array function is used to normalize the data to be within the appropriate value range or the same standard. This helps the model train and learn more efficiently by keeping the data in the value range between 0 and 1.

The results of the data analysis and modelling of the encoding configuration here are: The first dense layer with 10 nodes, which has 1) batch normalization, serves to improve the speed of the train model and decrease overfitting by fixing input values that enter the next layer, causing a mean value of 0 and a standard deviation of 1. and 2) Leaky ReLU Activation (LeakyReLU): this function is used in the dense layer to allow the model to learn complex characteristics. In training the model with autoencoder, which has the main feature of reducing the dimensionality of the data and being able to create output data that is similar to the original input data. Using Batch Normalization and Leaky ReLU activation function helps to increase training efficiency and reduce overfitting of the model. The results of this model can be used to reduce the dimensionality of the data or create data that is similar to the original data. The results of training the model for 1 0 0 rounds produces loss and val_loss values, which can be used to plot the loss graph. In both the training set, training loss and the test set, validation loss has continuously decreased until at one point it will be constant, and both graphs will have very little gap between them. The learning curve model shows that it is a good-fitting model or a model with good learning that can be used to predict data that has never been seen before, obtained from the mean squared error (MSE). It is a loss function measuring the difference between the predicted value and the actual value epoch 100. In addition, the test results, when presented in the form of visualization, show that the training loss and val_loss values start to run in parallel lines and become stable around the 70th cycle. It shows the efficiency of the model, as shown in Fig. 2.

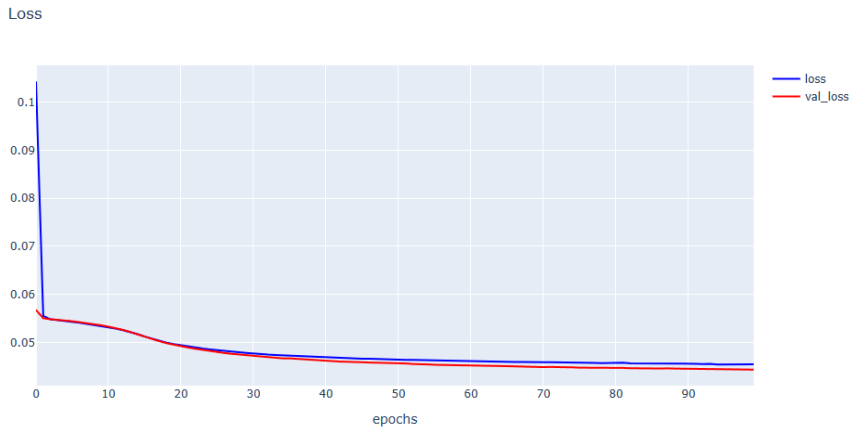


Fig. 2. Plot training & validation loss values.

In the process of tuning the parameters with embeddings of the data after dimensionality reduction with the autoencoder, which is in latent space with a reduced number of dimensions from the original autoencoder training process, the model often learns the characteristics of the data used in the training process. Predicting data that is similar to the data used for training has a better chance of producing better results. And if the data used to test or predict has characteristics that are too different from the data used in training, it can help in improving the performance of machine learning models. and reducing the data dimensions makes the model learn faster and more efficiently, as shown in Fig. 3.

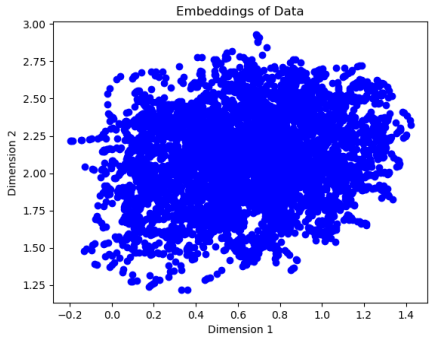


Fig. 3. Graph showing embeddings of data.

When the most effective model is found, an API will be developed to use the model to make a prototype recommendation system to test the introduction of the model and the compatibility of the data. The researcher chose to use the Flask API for development because of the data transmission format of the Python library, which is a Rest Framework working on the HTTP/HTTPS prototype. The work is used through various methods. And the communication is done through a data set formatted as JSON, handling requests that come in from the client side, responding to the client, and retrieving information about job positions to store in the SQL Server, including job positions, company names, ratings, users, and links. Then, a prototype web application of the job recommendation system, using the Flask framework, is created to return job recommendation results after users search terms for job positions in the search box on the web page, as shown in Fig. 4.

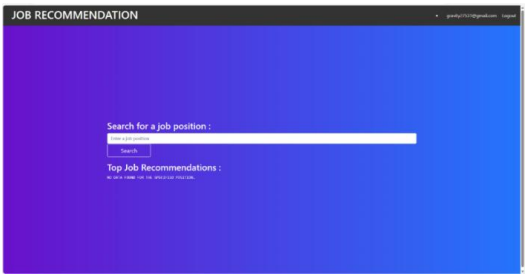


Fig. 4. Input job position keyword.

The tokenizer in Keras is used to convert the job position's text into a sequence of numbers (tokens) and add padding to make the sequence of numbers the same length. To predict scores for each job position, the predicted score is stored and connected to the job position. Job positions are sorted according to predicted scores from highest to lowest and the 10 job positions with the highest scores are selected. Finally, they are recommended further on the application web page. Qualitatively, the relevance of these top 10 recommended results demonstrates a strong alignment with the user's initial search keyword. Because the collaborative filtering model leverages latent representations learned through the autoencoder, it effectively ranks and retrieves job positions that share deep underlying characteristics with the user's inquiry. This ensures that the suggested roles accurately reflect the specific professional contexts and requirements of the applicant, going beyond simple keyword matching. as shown in Fig. 5.

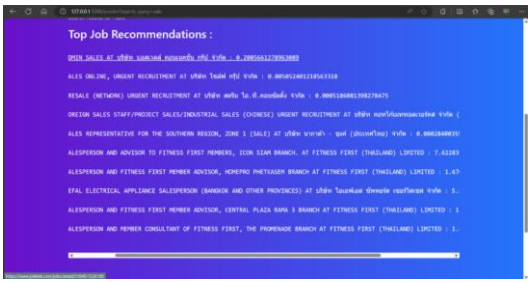


Fig. 5. Display Recommendations.

5. Discussion

The purpose of this research is to develop a prototype of a job position recommendation system with machine learning by using collaborative filtering. To achieve this and solve the core research problem of data disorder and lack of experience among job seekers, a multi-stage action was successfully implemented. Data of 30,000 records was collected via web scraping from a top job search website in Thailand. This data was then passed through a multi-stage filtering pipeline (including language, message, and number filtering) and dense matrix normalization via matrix factorization to eliminate duplicates and noise before entering the model training phase. Finally, a prototype web application was deployed using the Flask API connected to a SQL Server database to effectively recommend the top 10 highest-scoring job positions to users.

The loss obtained in both training loss and validation loss continuously decreased until a certain point where it remained constant. The learning curve format shows the learning efficiency of a good-fitting model or a model that has good learning and is able to predict. For a detailed error analysis, the model's performance was evaluated using mean squared error (MSE) as the loss function. The visualization of the learning curve demonstrated that both training loss and validation loss started running in parallel lines and became stable around the 70th cycle. The test results showed that the val_loss was as low as 0.0442 at epoch 100, which clearly confirms a well-fitting model with minimal error and robust learning ability on unseen datasets.

To actively prevent errors and overfitting during the development, strict data filtering and architectural constraints were applied. All data containing null values were strictly removed, and duplicate messages were filtered out. Furthermore, a threshold was implemented requiring each user and job position to have no less than 30 ratings to mitigate data sparsity issues. In the text predictive phrase, a tokenizer in Keras combined with sequence padding was used to ensure identical lengths. Crucially, the model's first dense layer integrated batch normalization to fix input values (resulting in a mean value of 0 and standard deviation of 1) alongside a Leaky ReLU Activation function, which successfully increased training efficiency and protected the model from overfitting.

However, this study involves certain operational trade-offs that need to be addressed. Adjusting parameters with data embeddings after dimensionality reduction with autoencoder can help improve the performance of machine learning models. and reducing the dimensions makes the model learn faster and more efficiently. The trade-off here is that while data reduction in latent space accelerates training speeds, the model becomes highly optimized for data similar to the training dataset; therefore, if the prediction data contains characteristics that are too different, the model's prediction accuracy might face challenges. Another trade-off occurred during the number filtering process, where raw website visitor numbers were scaled down into a 1–5 scoring format (e.g., 0–100 visitors equal to 1 point) to keep the dataset size from becoming too large, thereby sacrificing raw count granularity for computational efficiency.

This finding is consistent with the study of Hafizovic [3] that used machine learning to replace the experience of job seekers by using recommendation systems as [4-5] and collaborative filtering. Using the autoencoder collaborative filtering method [8-10] based on the rating data of the system users shows that this method has potential and efficiency in this research. And to solve the problem of sparsity in the model, matrix factorization was applied, as in [9], to convert the scoring matrix into a high-dimensional classification dataset the same size as the number of ratings. The results of the performance test of the model using the learning curve indicate that the model is a good fitting graph. There is good learning, and the model can be used to predict with new data sets that will come in, which corresponds to [8-10, 12-13]. In the development process, the prototype recommendation system uses the Flask API to work together, like [6], which used a machine learning model. In this paper, we propose a prediction system using the Flask API and link it between databases and a prototype job recommendation system to help those who are looking for a job and have no experience in searching for or screening job positions using machine learning.

Regarding future works, the current operational limitations of this research can be expanded by incorporating additional data attributes into the training model, such as detailed user profiles and job types to continuously enhance performance. Furthermore, alternative recommendation system techniques, specifically content-based filtering or hybrid filtering, should be investigated to expand the scope to other industries. This developed model framework can also be adapted to create job recommendation functions on existing job search platforms or transferred to other applications, such as movie or book recommendation systems.

Furthermore, while the current Flask API efficiently handles the 30,000 record dataset, scaling to larger, dynamic platforms may introduce speed constraints and latency. Future developments should incorporate robust frameworks, such as asynchronous processing or cloud-based micro services, to ensure real-time scalability and rapid recommendation delivery.

6. Conclusions

The research on the development a job position recommendation system using collaborative filtering was conducted to find an appropriate and efficient model for job position recommendation through a collection of web scraping methods. Data was retrieved from a job search website and taken through the process of filtering language data, text type data, and numeric data type. The dataset was prepared in the form of a table (CSV file) to use the data for matrix factorization. The data used has 4 attributes: 1) Position, 2) Company, 3) Rating, and 4) User. The method randomly

divides the data for training 80% and testing 20%, then creates a model using the autoencoder train model to receive job rating data from 30,000 records of each user (Users) using mean squared error as a loss function to test the learning efficiency of the model through the learning curve, which represents the learning efficiency of the model from the training dataset. From the results of the study, it was found that the learning curve of the graph obtained from training the model is good fitting. It indicates that the model has good learning ability. The model can be used to accurately predict new data sets that have never been seen before. This led to the application of the model to create a prototype job recommendation system that connects to work through the Flask API according to the prediction results of the model developed according to the Model.h5 format for use in predicting scores for each job position and connecting to job positions. Then data were sorted from highest to lowest and the top 10 highest-rated job positions were selected and further recommended on the prototype job recommendation system.

Regarding the explicit findings of this research, the study demonstrated that the integration of a rigorous multi-stage filtering pipeline (language, message, and number filtering) and matrix gactorization successfully eliminated web-scraped data noise and disorder. Consequently, the autoencoder-based collaborative filtering model achieved high accuracy, reflected in a remarkably low validation loss of 0.0442. Furthermore, essential lessons learned from this research indicate that while dimensionality reduction in the latent space accelerates training speeds, it introduces an operational trade-off where the model becomes highly optimized specifically for data sharing characteristics with the training set. Another crucial lesson learned is the necessity of scaling raw website visitor numbers into standardized rating scores to maintain computational efficiency and prevent data sparsity. Ultimately, the development of this prototype provides a practical lesson that integrating machine learning can successfully replace manual keyword filtering, directly bridging the gap between complex job market demands and the limited search experience of recent graduates and unemployed individuals.

In the future, developing a prototype job recommendation system can be an aid in decision making by recommending job positions that are appropriate for users, and also can be applied to create a job recommendation function on job search websites or other recommendation systems such as movie recommendation systems or book recommendation system, etc. Moreover, the model of the job recommendation system can be applied to job search websites that are currently available. Specifically, the primary target audience who will directly benefit from the practical application of this system includes university career centers, small-scale recruitment platforms, and local government employment agencies, enabling them to provide tailored and efficient job matching services for their candidates. However, this research still has operational limitations. Information about other job positions should be used to develop the model (Train Model) more, such as user information, job type information (Type job), etc. The researcher should study methods for developing models of recommendation systems by using alternative techniques, content-based filtering or hybrid filtering, to be able to expand the scope to other industries and develop its efficiency continuously.

References

[1] Smith, J. A. (2020). Challenges and Needs of Recent Graduates in Job Seeking. University of Employment Studies.

[2] Promrit, N. (2021). Fundamental of deep learning in practice. Bangkok: Chulalongkorn University.

[3] Hafizovic, N. (2019). Job recommendation system: Building a prototype of a machine learning – based recommendation system for an online recruitment company. Master's degree Thesis, Department of Computer Science, Department of computer science and media technology (CM), Faculty of Technology, Linnaeus University.

[4] Al Jobaer, A., Sanchi, B. B., Javed, A., Islam, M. S., & Jameel, A. S. M. M. (2021). An Advanced Recommendation System by Combining Popularity-Based and User-Based Collaborative Filtering Using Machine Learning. In 2021 International Conference on Science & Contemporary Technologies (ICSCT) 1-5.

[5] Khademizadeh, S., Nematollahi, Z., & Danesh, F. (2022). Analysis of book circulation data and a book recommendation system in academic libraries using data mining techniques. Library & Information Science Research, 44(4), 101191.

[6] Shrivastava, N., & Gupta, S. (2021). Analysis on item-based and user-based collaborative filtering for movie recommendation system. In 2021 5th International Conference on Electrical, Electronics, Communication, Computer Technologiesand Optimization Techniques (ICEECCOT); 654-6.

[7] Abodayeh, A., Hejazi, R., Najjar, W., Shihadeh, L., & Latif, R. (2023). Web Scraping for Data Analytics: A BeautifulSoup Implementation. In 2023 Sixth International Conference of Women in Data Science at Prince Sultan University (WiDS PSU) (pp. 65-9). IEEE.

[8] Ouyang, Y., Liu, W., Rong, W., & Xiong, Z. (2014). Autoencoder-based collaborative filtering. In Neural Information Processing: 21st International Conference, ICONIP 2014, Kuching, Malaysia, November 3-6, 2014. Proceedings, Part III 21 (pp. 284-91). Springer International Publishing.

[9] Yu, M., Quan, T., Peng, Q., Yu, X., & Liu, L. (2022). A model-based collaborate filtering algorithm based on stacked AutoEncoder. Neural Computing and Applications, 1-9.

[10] Gupta, A., Singh, N., & Hossain, A. M. I. (2023). Job recommendation system using machine learning. In Proceedings of the 2023 IEEE 2nd International Conference on Computing, Power and Communication Technologies (GUCON) 456-62.

[11] Suharyadi, J., and Kusnadi, A. (2019). Design and Development of Job Recommendation System Based on Two Dominants on Psychotest Results Using KNN Algorithm. International Journal of New Media Technology 5(2):116-20.

[12] Jain, H. and Kakkar, M. (2019). Job Recommendation System based on Machine Learning and Data Mining Techniques using RESTful API and Android IDE. 2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence), Noida, India, 416-21.

[13] Mishra, R., Rathi, S. (2020). Efficient and Scalable Job Recommender System Using Collaborative Filtering. In: Kumar, A., Paprzycki, M., Gunjan, V. (eds) ICDSMLA 2019. Lecture Notes in Electrical Engineering, vol 601. Springer, Singapore.

[14] Yang,S., Korayem,M., AlJadda, K., Grainger, T., Natarajan, S. (2017). Combining content-based and collaborative filtering for job recommendation system: A cost-sensitive Statistical Relational Learning approach.

[15] Mikhaylov, A., Bhatti, I. M., Dinçer, H., & Yüksel, S. (2024). Integrated decision recommendation system using iteration-

enhanced collaborative filtering, golden cut bipolar for analyzing the risk-based oil market spillovers. Computational Economics, 63(1), 305-38.

[16] Ajitsaria, A. (2019). Rating Matrix. Retrieved September 1 9 , 2 0 2 3. from https://realpython.com/ build-recommendation-engine-collaborative-filtering/.

[17] Shokrzadeh, Z., Feizi-Derakhshi, M. R., Balafar, M. A., & Mohasefi, J. B. (2024). Knowledge graph-based recommendation system enhanced by neural collaborative filtering and knowledge graph embedding. Ain Shams Engineering Journal, 15(1), 102263.

[18] Deng, X. (2021). Automated Essay Scoring for English Using Different Neural Network Models for Text Classification.

[19] Bjerrum, E. J., & Sattarov, B. (2018). Improving chemical autoencoder latent space and molecular de novo generation diversity with heteroencoders. Biomolecules, 8(4), 131.

[20] Chaiaroon, P., Promrit, N., Sitdhisanguan, K., Waijanya, S., & Kanraweekultana, N. (2025). Digital Workforce Matching: A Machine Learning Approach for Skill-Based Job Classification and Recommendation. Journal of Current Science and Technology, 15(4), 137-7.

[21] Hadisantoso, Q. C. M., Nathaniel, G., & Kurniadi, F. I. (2022, August). Online Book Recommendation System Based on Optimized Collaborative Filtering Using Ant Colony Optimization. In 2022 5th International Conference on Information and Communications Technology (ICOIACT) (pp. 457-61). IEEE.

[22] Qi, L., He, Q., Chen, F., Dou, W., Wan, S., Zhang, X., & Xu, X. (2019). Finding all you need: web APIs recommendation in web of things through keywords search. IEEE Transactions on Computational Social Systems, 6(5), 1063-72.

[23] West, R., Zacharias, M., Assaf, W., Aelterman, S., Davidson, L., & D'Antoni, J. (2020). SQL Server 2019 Administration Inside Out. Microsoft Press.

[24] Sharma, K., Shetty, A., Jain, A., & Dhanare, R. K. (2021, January). A Comparative Analysis on Various Business Intelligence (BI), Data Science and Data Analytics Tools. In 2021 International Conference on Computer Communication and Informatics (ICCCI) (pp. 1-11). IEEE.

[25] Bin Uzayr, S. (2022). Mastering Visual Studio Code: A Beginner's Guide. CRC Press.