Improving Web Service Recommendation using Clustering with K-NN and SVD Algorithms

Amith M. Weerasinghe[1] and Rupasingha A. H. M. Rupasingha[2*]
[1]Faculty of Applied Sciences, Sabaragamuwa University of Sri Lanka
Belihuloya, Sri Lanka
[e-mail: lrmamweerasinghe@gmail.com]
[2]Faculty of Social Sciences and Languages, Sabaragamuwa University of Sri Lanka
Belihuloya, Sri Lanka
[e-mail: hmrupasingha@gmail.com]
*Corresponding author: Rupasingha A. H. M. Rupasingha

Received September 19, 2020; revised January 24, 2021; revised March 1, 2021; accepted March 28, 2021; published May 31, 2021

Abstract

In the advent of the twenty-first century, human beings began to closely interact with technology. Today, technology is developing, and as a result, the world wide web (www) has a very important place on the Internet and the significant task is fulfilled by Web services. A lot of Web services are available on the Internet and, therefore, it is difficult to find matching Web services among the available Web services. The recommendation systems can help in fixing this problem. In this paper, our observation was based on the recommended method such as the collaborative filtering (CF) technique which faces some failure from the data sparsity and the cold-start problems. To overcome these problems, we first applied an ontology-based clustering and then the k-nearest neighbor (KNN) algorithm for each separate cluster group that effectively increased the data density using the past user interests. Then, user ratings were predicted based on the model-based approach, such as singular value decomposition (SVD) and the predictions used for the recommendation. The evaluation results showed that our proposed approach has a less prediction error rate with high accuracy after analyzing the existing recommendation methods.

Keywords: Clustering, Collaborative Filtering, K-Nearest Neighbor, Recommendation, Singular Value Decomposition, Web Services

1. Introduction

Web services are supported by server-side data and information sharing between computers through the Internet. Humans commonly interact with technology, and as a result, the Web has been expanding to a large number of Web services which support developers in their making of products or services. Consequently, most Web service users face the problem of finding complementary Web services. Therefore, the solution to this problem is "recommendation" which can help Web users in discovering Web services effectively and efficiently.

This recommendation has been described with three types of classifications: [1] content-based, CF, and hybrid filtering. The CF technique is categorized into model-based, memory-based, and hybrid-based [2], [3] types of service recommendation. In our proposed approach we used a model-based CF method that was built with an SVD algorithm.

We selected the CF approach because the content-based approach has some drawbacks [4]. The main problem with the content-based approach is that it requires the users' personal information. Collecting and using the users' personal information is risky because hackers can breakdown or damage the system and filch users' information. However, CF only uses the users' past interests and assumes them to be their future interests. Therefore, users' personal information is not highly affected. Hybrid filtering is a combination of CF and content-based filtering [5].

We focused on the main problems of the CF approach, such as data sparsity and cold-starting problems and solved them using the proposed approach. These problems occur from a weak user's cooperation with the given rate on the rating mechanism and comments on a feedback mechanism. Therefore, it is difficult to recognize the similarity between users. However, CF also has problems with system scalability, shilling attack, and synonymy.

There are available methods for reducing data sparsity and cold-starting. Some of these methods use transitive association [6], make clustering for reducing dimensionality by latent semantic indexing [2],[7], use binary preferences for recommendation [8], and use correlation and cosine methods [3] that can be applied to give good recommendation results by reducing the data sparsity. We selected clustering-based methods for reducing the data sparsity and there are several available clustering methods [9]–[13] . We selected a specificity-based novel ontology generation method [13], [14] as our clustering approach to classifying the better cluster groups. This clustering approach presented higher precision, recall, and f-measure performance by comparing it with other existing clustering approaches. After classifying the cluster groups, we applied the KNN powerful machine learning algorithm for alleviating the sparsity in each cluster group [15] separately. This algorithm computes the nearest neighbors and assigns a similarity value for the user's missing fields in the data set. Next, we applied the latent semantic analysis (LSA) method such as the SVD algorithm to predict user ratings using the updated user-service data set by KNN. The reason for applying SVD for the CF process is that it shows better performance than other existing model-based approaches. SVD-based recommendation algorithms can make high-quality recommendations quickly. The result showed that it can produce a low dimensional representation of the original user-service space and these predicted ratings were used to enhance the recommendation.

For calculating the accuracy of the predicted dataset, we used mean absolute error (MAE) and root mean square error (RMSE). We compared our approach with three different sparsity levels and a different number of Web service data, different sparsity alleviating methods including different clustering approaches, different types of SVD improvement techniques, and also compared it with the existing recommendation methods. Based on the result of the

comparison, we identified that our proposed method generated the best recommendation result with lower MAE and RMSE error rates.

We discussed the remainder of the paper as follows. In section two, we discuss related work, section three is the motivation, section four is the overview of the proposed recommendation approach, section five is the experiments and evaluations, and section six is the conclusion and implications for future work.

2. Related Work

2.1 Collaborative Filtering

The CF approach [2], [3] is applied to predict and recommend new services for a particular user based on other users' opinions. Our proposed approach is based on the user service matrix and it gets the result by analyzing the matrix. The assumption is that "users who adopted the same behavior in the past will tend to agree also in the future" [16].

The CF approach can be constructed using the following methods: memory-based and model-based recommendations[16]. However, the memory-based recommendation method depends on the user-item correlation. The memory-based approach uses the neighborhood-based methods [17], like KNN [18], the Pearson correlation coefficient (PCC) [2]. This method computes the similarity between user and item and converts the preference knowledge into predictions [19], [20]. The drawback is that they want to access the entire dataset to be able to predict with specific indexing techniques. Therefore, the data scale is increasing. The model-based recommendation method considers the preference matrix and [19] uses online and offline phases to predict the recommendation. In the offline phase analyzed by the personalization model and in the online phase, the method predicts user interest with items. The dimensionality reduction technique [16] also uses the model-based method. The famous model-based techniques [21] are matrix factorization [22], artificial neural network [23], and Bayesian network. According to the matrix factorization, using SVD can capture the latent relationship between users and services. That allows us to compute the likeliness of a certain service by a user. Otherwise, SVD can produce a low dimensional representation of the original user-service space and then compute the neighborhood in reduced space [22]. Zainab Al-Zanbouri [24] has implemented a context-aware matrix factorization model to build a recommender system where energy consumption is the main Quality of Services (QoS) attribute considered for energy-efficient service recommendation. By considering these different techniques, we selected the SVD-based recommendation since it has a high accuracy with minimum wastage time [25].

Further study of the recommendation revealed that in addition to the above-mentioned methods, the recommendation made by deep learning gives more effective results [26].

2.2 Challenges of User-based CF Algorithms

One of the best recommendation methods is CF which can use the user-service matrix. However, when increasing the user services on the Web, the recommendation system is adapted to facing some problems. In the dataset, most users do not actively interact with the rating mechanism. Therefore, the dataset is filled with unrating (null) values. This problem is referred to as a data sparsity problem. CF faces other problems, such as new users or services coming into the system. This problem is called the cold-starting. These problems interrupt the system to find the user-service relationship and challenge to construct an effective recommendation.

2.2.1 Data Sparsity Problem

One of the main problems of CF is the data sparsity problem. This can reduce the recommendation performance. There are available approaches to reduce data sparsity. One of these approaches uses ontology-based cluster results [2] to overcome the sparsity problem and improve the recommendation performance. Another approach uses association retrieval technology [27] based on the user's feedback data. Another approach is based on a simplified similarity measuring method [28]. They use ratings and then convert them into binary preference values and establish the similarity groups of users. The simple machine learning technique KNN that is used for sparsity data reduction [29] is that it selects a *k* number of users and finds their similarities. And using KNN for uncertain data [30], it makes KNN not to wait for test tuple, because it is categorized as a lazy learner. Therefore, the prediction time is less than that of other approaches and we selected the KNN-based sparsity reduction for our proposed approach. These methods support the sparsity reduction and improve the recommendation performance.

2.2.2 Cold Starting Problem

Another problem with CF is the cold-starting problem. These problems occur with empty ratings in user and item profiles.

Generating the ontology-based clustering method [2], the constructed item-based clustering groups and the clusters help to alleviate the item-based cold-starting problem. They also use the heuristic similarity measurement [31] method for avoiding cold-staring. Another existing method [32] for alleviating the cold-starting problem is the use of the social balance theory to check enemies and find available friends. Then they make a recommendation upon the preferred service of friends and non-preferred services of enemies of their target users.

3. Motivation

The main problems of collaborative filtering-based recommendations are data sparsity and cold starting. These problems in the user-service data set occur with limited user interaction with the user-service rating matrix or the empty user-service profiles. As a result, the recommendation system cannot predict the best recommendations. Therefore, this kind of system becomes a failure, because it has a high error rate.

Table 1. Example of user-service rating graph

Users	Web Services							
	w_1	w_2	w_3	w_4	w_5	w_6	w_7	w_8
u_1	1	0	1	0	0	0	0	0
u_2	0	0	0	0	0	0	1	0
u_3	1	0	0	1	0	0	0	1
u_4	0	0	0	0	1	0	0	0
u_5	0	0	0	1	0	0	0	0
u_6	0	0	0	0	0	0	0	0
u_7	0	1	0	0	0	0	0	1

Seven users and eight Web services ratings are shown in **Table 1**. There, all users do not actively interact with the rating system. When increasing the number of users and services, it is often difficult to generate good results.

3.1 Example of Data Sparsity Problem

See **Table 1**, let active user u_4. The user u_4 only interested to service w_5. Then find who interested in the w_5. But, except u_4 no one likes to w_5. Therefore, when the system cannot be built by any relations for u_4, the system struggles to make a prediction. This situation is called a data sparsity problem.

3.2 Example of Cold-start Problem

When *u6* and *w6* are not active yet, their profile is empty. That is, there is no information for making any kind of relationship with users and/or services. Therefore, the system cannot make predictions. This problem is called the cold-start.

When making the Web service recommendation, these sparsity and cold-start problems also exist. Web service users also have to face some troubles with wastage of time and cost as a result of these problems. This motivated us to use a proper approach to overcome these problems effectively.

According to the literature review, we found that clustering methods help to categorize the available services accurately and KNN is a powerful machine learning algorithm that makes the similarity between users. Also, the SVD method showed better results in the model-based collaborative filtering. Therefore, we chose the clustering approach for alleviating the above problems with the support of the KNN algorithm. That fills non-voting values calculating the similarity between users. Then we applied SVD and it helped to make better recommendation results.

4. Proposed Recommendation Approach

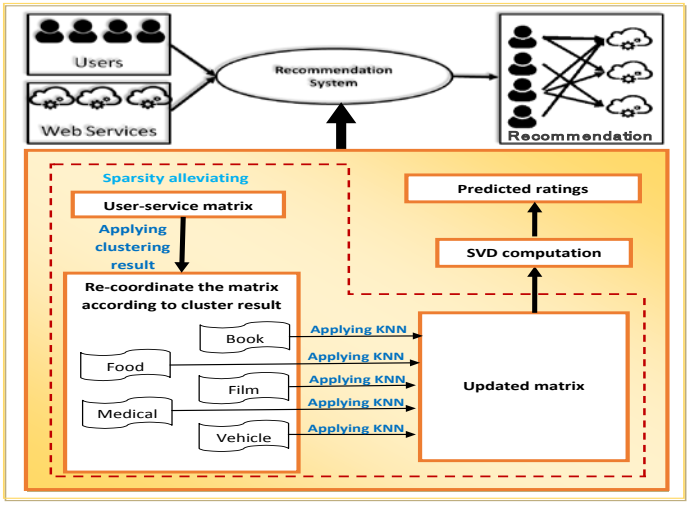

Fig. 1. Architecture of the proposed method for recommending Web services

Fig. 1 shows the architecture of the proposed recommendation approach, which contains four main phases.

Step 1: First, we have collected the user service dataset.

Step 2: (i) For reducing the sparsity, initially we applied the ontology-based clustering method [13] to categorize Web services.

(ii) Then we applied the KNN algorithm for each categorized group. It calculated the similarities between each user and assigned ratings for non-votings.

Step 3: After reducing the sparsity, we applied the SVD method to a new matrix dataset for the users' rating prediction.
Step 4: These prediction results were used for getting the recommendation results.

4.1 Data Collection

First, we collected the user-service input data as 400 real Web services with 200 simulated users' ratings. In the rating matrix (u,w), its rows represent each user (u) and its columns represent each service (w). User (u) can vote 1 to 5 value for each service (w). If it has 0 ratings, that means user u cannot vote for Web service w.

$$r_{u,w}\begin{cases} r, & r = 1,2,3,4,5 \; if \; user \; u \; rated \; service \; w, \\ 0. & Otherwise \end{cases} \tag{1}$$

4.2 Resolving the CF Problems

As shown in **Fig. 1**, the sparsity alleviation process is used for resolving the CF problems before the rating prediction. It is highly important for maintaining the recommendation accuracy. Therefore, we first applied a clustering-based approach and then used the KNN algorithm to reduce the CF problems.

4.2.1 Resolving Sparsity Problem

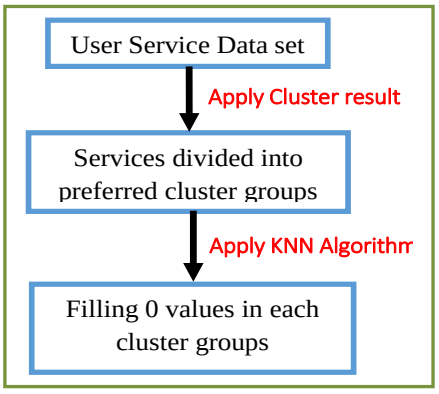

Fig. 2. Process of solving the sparsity problem

Algorithm 1: Sparsity Alleviating

Input W: Web Service Dataset
U: User's dataset
R: User Service invoke data
C: Web service clustering results
Output *O*: Recommendation results
For each user U **do**
  C_g = divided rating data (r_{uw}) in R separately for each C cluster groups;
  For each cluster C_g **do**
    R` = Update non-rate Web services in C_g using KNN;
  end-for
end-for
For each user, invoke data (r_{uw}) in R` **do**
  N = new predicted rating using SVD;

9: **end-for**
10: O= Do recommendation using N;

Since the user-service matrix is very sparse, we cannot capture the meaningful latent relationship. For reducing the sparsity, we first used the ontology-based clustering result. It divided Web services into preferred cluster groups. Therefore, similar users will come to the same cluster. After that, we applied the KNN algorithm for calculating user similarities in each cluster group. The KNN results were used for getting the rating values to fill in the non-rating services. **Fig. 2** presents our method of solving the sparsity problem and algorithm 1 describes the overall process of solving the sparsity problem and continuing recommendation.

4.2.1.1 Apply Clustering

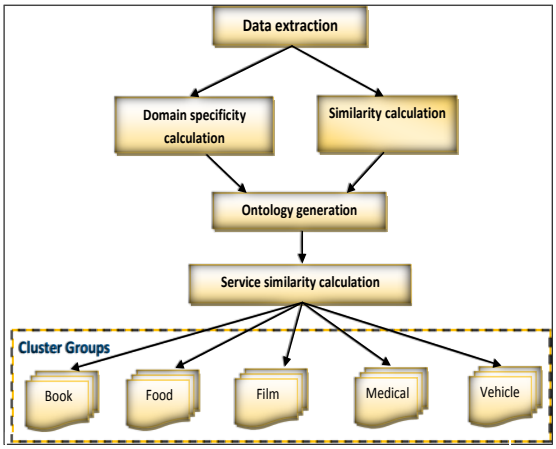

Fig. 3. Ontology-based clustering method

As a process of alleviating the sparsity problem, we first applied the clustering approach for re-coordinating the matrix. As a clustering approach, we used the previous ontology-based clustering results [15]. This clustering approach is applied only for Web services in our user-service data set. According to **Fig. 3**, we first selected real Web services from Web services repositories and Web Ontology Languages for Services (OWL-S) (http://projects.semwebcentral.org/projects/owls- tc/). This has Web service description languages (WSDL), documents related to the five domains of Film, Food, Medical, Vehicle, and Book. Then they extracted five features as the service name, operation name, port name, input, and output messages from each WDSL documents. After extracting the features, they calculated the domain specificity and similarity weight [15] based on each of the extracted terms, and using these calculation results, they generated ontology. Then they calculated the service similarity based on each of the node relationships of the generated ontology, and finally, they found cluster results using the similarity values and agglomerative clustering algorithm. All Web services were grouped into the above five cluster groups. The reason for using this approach is that it showed better clustering results than other existing approaches as described in [15].

4.2.1.2 KNN Calculation

After re-coordinating the matrix according to the clustering result, as shown in **Fig. 1**, We applied KNN algorithm for these cluster groups (Film, Food, Medical, Vehicle, and Book)

separately. It calculated the similarities of the most similar users in there and filled missing ratings according to the similarity values. However, we did not use the KNN as a clustering approach. We used it here only to compare similarities and predict missing ratings.

We used KNN because KNN [33] is a simple and fast machine learning technique for CF (See **Fig. 4**, the process of KNN), it selected *k* number of nearest neighbors (Users), and *k* was used as a smoothing parameter. This method used *k = 40*, according to the [33] and they represented their result help with the 100K and 1M dataset with *k=40* and proved that it has a better result with minimum time. And also we used evaluation for finding the value of *k* and we selected as *k* = 40. We found their similarities using Equation (4), and then used Equation (2) for generating the prediction.

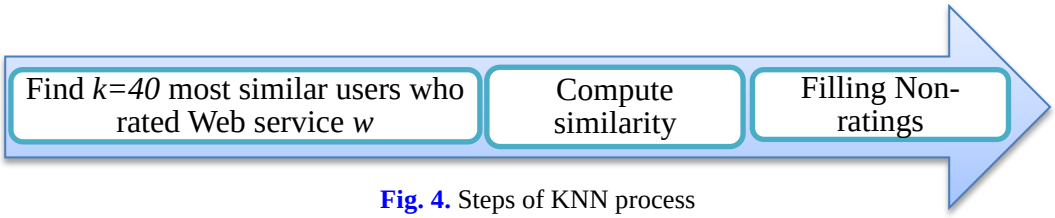

Fig. 4. Steps of KNN process

In our KNN calculation, the prediction $\hat{R}_{u,w}$ is set as:

$$\hat{R}_{u,w} = \frac{\sum_{v \in N_w^k(u)} sim(u,v).r_{v,w}}{\sum_{v \in N_w^k(u)} sim(u,v)} \quad , r_{u,w} = 0; \tag{2}$$

$$\hat{R}_{u,w} = r_{u,w} \quad , r_{u,w} \in 1,2,3,4,5; \tag{3}$$

$N_w^k(u)$ - k nearest neighbors of user u that rated for the Web service w
K - max number of neighbors
$r_{u,w}$ and $r_{v,w}$ - actual rating values of user u and user v on the Web service w
The similarity is calculated using Equation (4).

$$\boldsymbol{sim(u,w)} = \frac{\sum_{w \in W}(\boldsymbol{r_{u,w}} - \bar{\boldsymbol{r}}_U)(\boldsymbol{r_{v,w}} - \bar{\boldsymbol{r}}_V)}{\sqrt{\sum_{w \in W}(\boldsymbol{r_{u,w}} - \bar{\boldsymbol{r}}_U)^2}\sqrt{\sum_{w \in W}(\boldsymbol{r_{v,w}} - \bar{\boldsymbol{r}}_V)^2}} \tag{4}$$

$\bar{r}_U$ and $\bar{r}_V$ is the average rating values of different services of user u and user v.

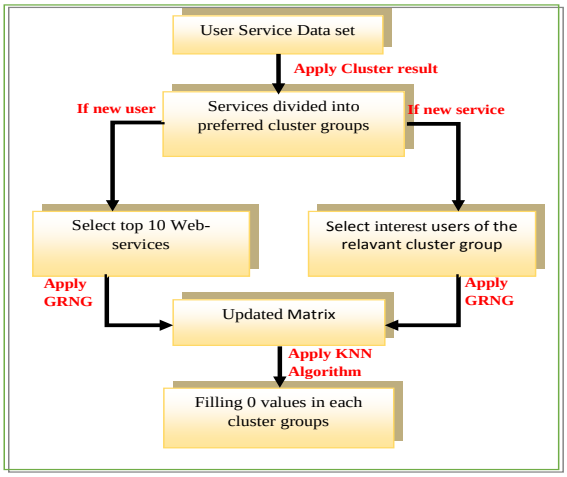

Fig. 5. Process of solving the cold-start problem

4.2.2 Resolving the Cold-start Problem

Cold-start is a difficult problem of the CF recommendation. If the rating matrix has a new user or service, CF cannot build a relationship between others. Therefore, we applied the following process to avoid the cold-start problem as shown in **Fig. 5**.

4.2.2.1 Avoiding User-based Cold-start Problem

Using the user-service rating matrix, we calculate the sum of the ratings in each service and then select the maximum ten ratings as top Web services (trendings). The rating values for those selected services are applied using the Gaussian random number generation (GRNG) and those ratings are used as the newly entered user's ratings. After that, we continue the process using the KNN. The overall process of avoiding the user-based cold-start problem and continuing the recommendation process is described in algorithm 2.

Algorithm 2: Avoiding the User-based Cold-start Problem

Input W: Web Service Dataset
U: User's dataset
R: User Service invoke data
C: Web service clustering results
C_g: Selected cluster groups
U_{new} : New users
Output O: Recommendation results

For each service w **do**
  W_s = calculate summation of rating data (r_{uw}) in R;
end-for
W_{Max} = 10 Maximum Ws;
For each U_{new} **do**
For each W_{Max} **do**
  $R`$ = Update W_{Max} using Gaussian distribution;
end-for
end-for
$C_{new} = C_g + R`$
For each cluster C_g **do**
$R``$ = Update non-rate Web services in C_{new} using KNN;
end-for
For each user, invoke data (r_{uw}) in $R``$ **do**
N = new predicted rating using SVD;
end-for
O= Do recommendation using N;

4.2.2.2 Avoiding Service-based Cold-start Problem

When a new Web service is available in the system, we first apply ontology-based cluster generation into the new service, and through that, we can identify the relevant cluster group of the new service. Then we calculate each user's average of ratings in each separate cluster and identify each user's preference cluster groups. Finally, GRNG is used to apply ratings for the new Web service based on the cluster group of the new service and preference users of that cluster. After that, we continue the process using the KNN. The overall process of avoiding the service-based cold-start problem and continuing the recommendation process is described in algorithm 3.

Algorithm 3: Avoiding the Service-based Cold-start Problem

Input W: Web Service Dataset
U: User's dataset
R: User Service invoke data
C: Web service clustering results
C_g: Selected cluster groups
W_{new} : New services
Output O: Recommendation results

For each U_{new} **do**
  C_{new} = find related cluster group
end-for
For each user U **do**
C_s = calculate summation of rating data (r_{uw}) in R separately for each C cluster group; //for each user separately for five domain clusters
Maximum clusters C_s=user's preference clusters
end-for
For each U_{new} **do**
  If C_{new} = Maximum clusters C_s;
  If $r_{uw} == 0$
$R`$ = Update non-rate Web services in C_{new} using Gaussian distribution;
  end-if
  end-if
end-for
$C_{new} = C_g + R`$
For each cluster C_g **do**
  R'' = Update non-rate Web services in C_{new} using KNN;
end-for
For each user, invoke data (r_{uw}) in R'' **do**
  N = new predicted rating using SVD;
end-for
O= Do recommendation using N;

4.3 Rating Prediction

After reducing the sparsity using KNN, then we can turn it into the rating prediction and improve our results. In such a case, we apply the matrix factorization technique as the SVD [34] method. SVD is built from linear algebra. It is used as a dimensionality reduction technique in generating better recommendation results. The rating prediction process is shown

in **Fig 6.**

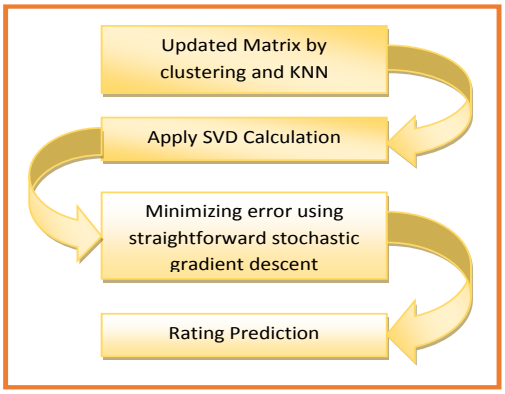

Fig. 6. Process of rating prediction

The prediction result $\hat{r}_{u,w}$ is set as:

$$\hat{r}_{u,w} = \mu + b_u + b_w + q_w^T . p_u \tag{5}$$

μ - mean of all ratings, it depends on our dataset.
b_u, b_w - Observed deviation of user u, Web service w
If user u is unknown, then the b_u and p_u are assumed to be zero. The same applies to Web service w with b_w and q_w.

Then we minimize the regularized squared error for an estimate of all the unknown values.

$$\sum_{r_{u,w} \in R_T} (r_{u,w} - \hat{r}_{u,w})^2 + \lambda(b_w^2 + b_w^2 + ||q_u||^2 + ||p_u||^2) \tag{6}$$

The minimization is performed by a very straightforward stochastic gradient descent:

$$b_u \leftarrow b_u + \gamma(e_{u,w} - \lambda b_u) \qquad p_u \leftarrow p_u + \gamma(e_{u,w} . q_w - \lambda p_u)$$
$$b_w \leftarrow b_w + \gamma(e_{u,w} - \lambda b_w) \qquad q_w \leftarrow q_w + \gamma(e_{u,w} . p_u - \lambda q_w)$$
$$e_{u,w} = r_{u,w} - \hat{r}_{u,w} \tag{7}$$

γ = learning rate
λ = regularization term
$r_{u,w}$ = input dataset
$e_{u,w}$ = error of training and predicted dataset
R_T = set of the (u,w) pairs for which $r_{u,w}$ is known

For the above SVD calculation, we used γ=0.005, λ=0.02 and μ. μ is the mean of all ratings and it depends on the dataset. γ and λ found by the evaluations. And also, according to [33], they represented their result help with the 100K and 1M dataset with those values and proved that it has a better result with minimum time.

With the support of SVD, we predicted all user ratings that could be used for service recommendations.

4.4 Recommendation

We combined the recommendation approach with the above-mentioned methods of resolving the CF problems (such as sparsity and cold-start) by clustering and KNN and then rating prediction by SVD. Those prediction results were used for the recommendation.

5. Experiments and Evaluations

For implementing this approach, we used the Surprise package [33], powered by the Python language on the Microsoft Windows 10 Operating System. That includes Intel Core i5-7200U at 2.5GHz and 6GB Random Access Memory (RAM).

First, we collected ratings according to the user-services. Our dataset has simulated 200 users' ratings and 400 real Web services. We could not find actual rating data for these real Web services. Therefore, we used a simulated dataset generated by Rupasingha et al [2], [7] using the GRNG. The dataset has high accuracy since they [2], [7] used and proved by experiments. This simulated dataset helped us to measure and evaluate performance with MAE and RMSE. We used three sparsity levels (55%, 70%, and 85%) to represent and analyze performance properly.

$$Sparsity\ level = \left(1 - \frac{Number\ of\ specified\ ratings}{Number\ of\ all\ possible\ ratings}\right) * 100\% \quad (8)$$

$$MAE = \frac{1}{N}\sum_{w=1}^{N} |r_{u,w} - \hat{r}_{u,w}| \quad (9) \qquad RMSE = \sqrt{\frac{1}{N}\sum_{w=1}^{N} (r_{u,w} - \hat{r}_{u,w})^2} \quad (10)$$

$r_{u,w}$ = training dataset user u, on Web service w w = Web services
$\hat{r}_{u,w}$= predicted dataset user u, on Web service w N = number of predicted values

We used MAE for measuring the deviation between prediction and actual ratings and used RMSE for measuring the square root of the average of differences between prediction and actual ratings. When the result showed smaller measuring values, it meant this approach has better performance.

5.1 Evaluation with different techniques

We alleviated sparsity using the clustering approach with the KNN algorithm and then applied SVD for rating predictions in three sparsity levels (55%, 70%, 85%). **Fig. 7** (a) and (b) draft the MAE and RMSE for recommendation prediction results. It compared performance as following,

1. Only SVD – Directly applied SVD algorithm for our dataset to rating prediction and recommendation.
2. KNN + SVD – Directly applied the KNN algorithm for finding users' similarities and filling the missing ratings (sparsity alleviating) in our dataset. Then applied SVD algorithm for updated dataset to rating prediction and recommendation.

3. Clustering + KNN + SVD – Applied ontology-based clustering approach to classifying the Web services cluster groups and then applied KNN algorithm for these cluster groups separately. That finding users' similarities and filling the missing ratings (sparsity alleviating) in our dataset. Then applied SVD algorithm for updated dataset to rating prediction and recommendation.

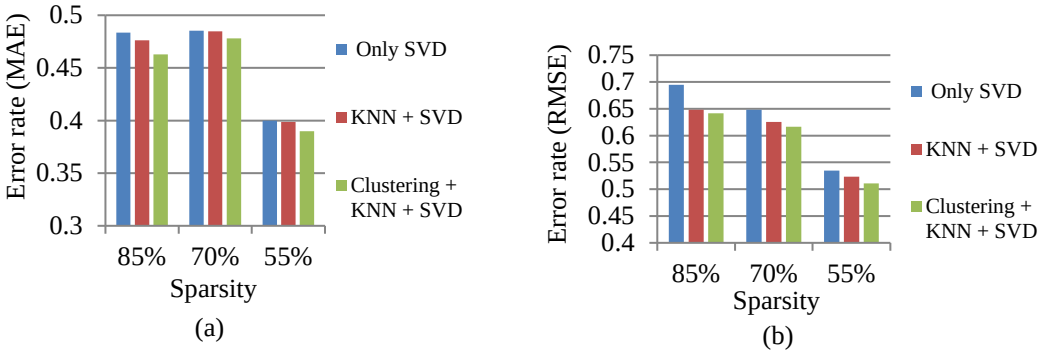

Fig. 7. Result improvement using SVD: (a) MAE, (b) RMSE

As shown in **Fig. 7**, Our proposed approach of ontology-based clustering, KNN algorithm, and SVD combined method (Clustering + KNN + SVD) showed better performance, and its 55% sparsity level has the lowest error rate. Therefore, we can consider that this combined approach is better for improving the recommendation performance.

Next, we compared the new sparsity filling technique (KNN algorithm) with different types of sparsity filling techniques such as average, median, GRNG. Before applying the above-mentioned techniques we applied the clustering method for classifying Web services and classified groups are used for **Fig. 8** evaluation.

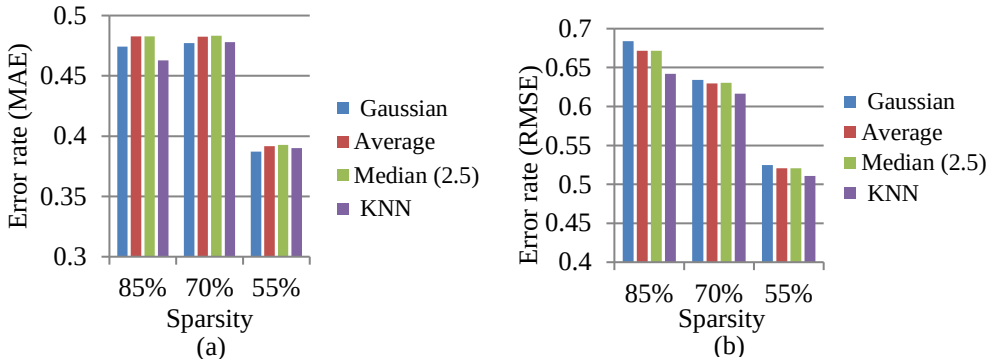

Fig. 8. Comparison with different sparsity alleviating methods: (a) MAE, (b) RMSE

Here, the average value was got using the average user rating value of the user-specific cluster group that previously user-rated, median value select as 2.5 since all ratings are in 1-5 range, Gaussian (GRNG) was used as the random number generation, and KNN algorithm.

The results are shown in **Fig. 8** (a) and (b) and it presents KNN having a minimum error rate. We can see that KNN can successfully alleviate the sparsity by selecting the most suitable rating value compared with the nearest neighbors than other sparsity alleviating methods. It seems that in KNN comparing nearest neighbors' ratings is able to improve the performance than other methods.

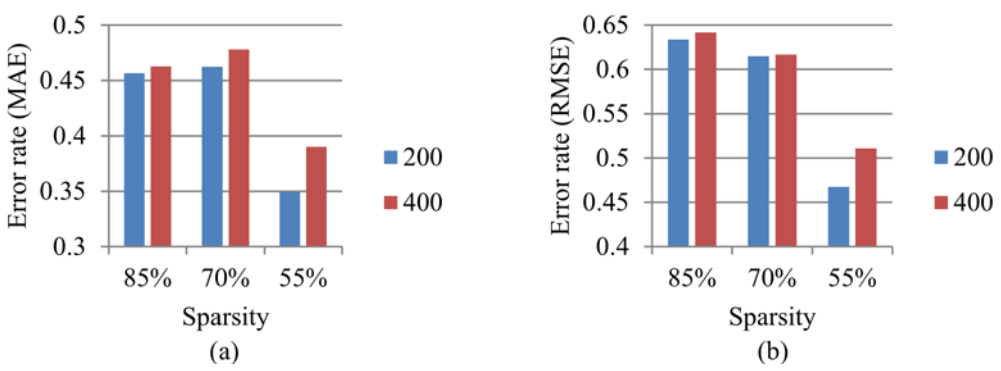

Fig. 9. Comparison with different number of Web services

The next evaluation is based on a different number of Web services. We used 200 and 400 Web services and **Fig. 9** (a) and (b) show the results. According to the results, when the matrix population increases, its performance decreases.

5.2 Evaluation with Existing Methods

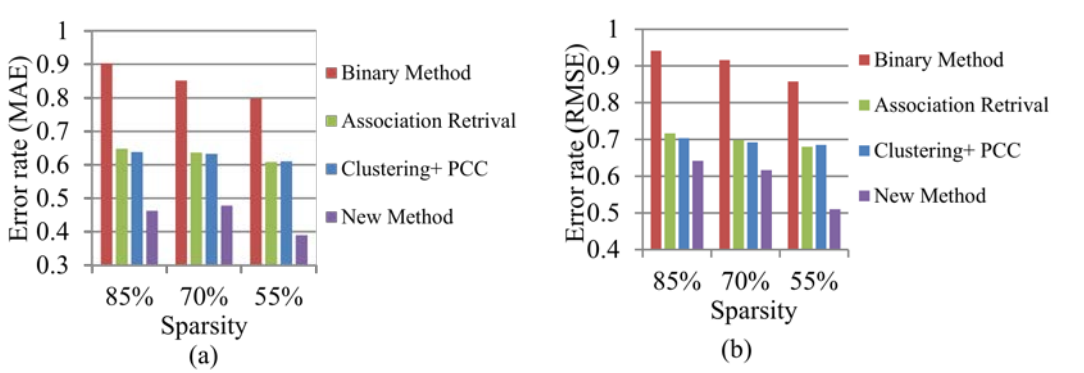

Fig. 10. Comparison with existing recommendation methods: (a) MAE, (b) RMSE

Then we compared the new method with the following existing recommendation methods which were based on the rating prediction with sparsity alleviation such as binary method [28], association retrieval method [27], and ontology-based clustering with the PCC [2].

1. The binary method [28] solved the sparsity problem using the simplified similarity measure (SSM) approach. It converts ratings into binary values and therefore, easily finds similar users. Then PCC is used for predicting the ratings.

2. The associative retrieval method [27] used a transitive association of users' feedback data and the relative distance between the users' ratings. Then they calculated their similarities and then combined them for reducing the sparsity. Then PCC is used for predicting the ratings.

3. The Clustering+PCC method [2] used Ontology-based clustering results and GRNG for sparsity alleviating and PCC for predicting the ratings.

The comparison results of these methods are shown in **Fig. 10** (a) and (b). We can see that our proposed method, alleviating the sparsity using clustering and KNN and rating prediction using SVD has better performance with minimum MAE and RMSE values. And also we can see that SVD showed better prediction results comparing with other methods such as PCC. SVD can capture the latent relation between users and Web services easily and quickly. It allows us to calculate the likeliness of certain Web services by the user and able to improve the performance.

Then we compared our method clustering approach with the following existing clustering approaches to find which clustering approach is best for use. See **Fig. 11** (a) and (b), it presented without clustering, using the context-aware similarity (CAS) clustering method [11], using the hybrid term similarity (HTS) [10], [12] clustering method and using the specificity aware ontology-based clustering method [13].

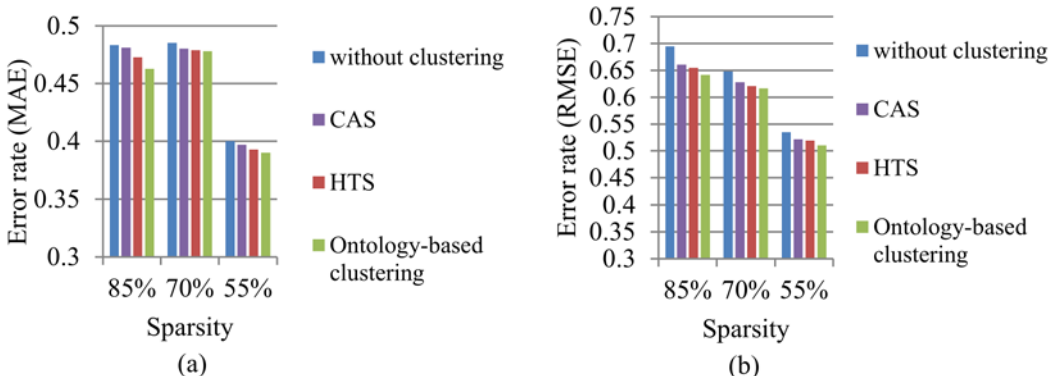

Fig. 11. Comparison with existing clustering approaches: (a) MAE, (b) RMSE

1. Context-aware similarity (CAS) clustering method [11] calculated its similarity by applying the support vector machine (SVM) and visualization of the cluster groups taken by the spherical associated keyword space algorithm.

2. Hybrid term similarity (HTS) [10], [12] clustering method calculated similarity by generating ontology using the hidden semantic patterns existing within complex terms and used agglomerative clustering algorithm for the clustering.

3. Specificity-aware ontology-based clustering approach [13] is applied in our proposed method and it calculated its similarity by generating the ontology based on the domain specificity and continue clustering using the agglomerative clustering algorithm

According to this result, the specificity-aware ontology-based clustering approach [13] showed a minimum error rate. That clustering approach uses information in specific terms rather than depend on more general terms. When analyzing the domain-related information, specific terms are more powerful than general terms and that helps to give better clustering results and able to improve the performance of our approach.

5.3 Evaluation with different parameter

Then we measure the proposed approach performance by changing different parameters. The value of the *k* in the KNN algorithm in Equation (2) is presented in **Fig. 12** below. However,

when the value of K = 40, its error rate appears to be lower. Next, we considered the λ in Equation (5) and γ in Equation (6) variants of the SVD algorithm separately. The results are presented in **Fig. 13** and **Fig. 14** below respectively. When γ = 0.005 and λ = 0.02, its error rate is kept to a minimum.

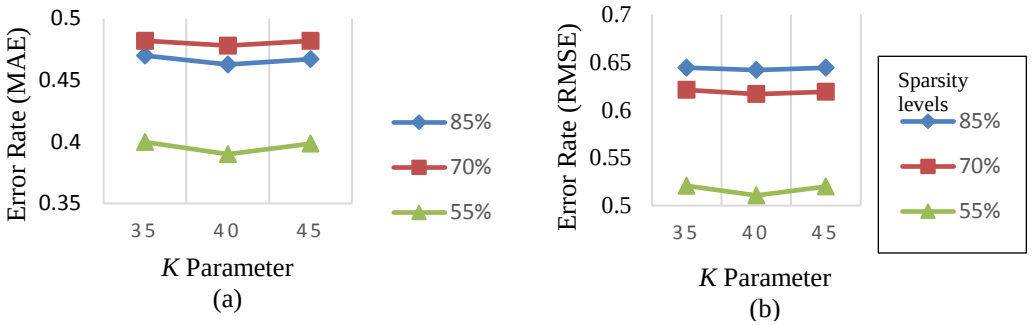

Fig. 12. Comparison with different K parameters (a) MAE, (b) RMSE

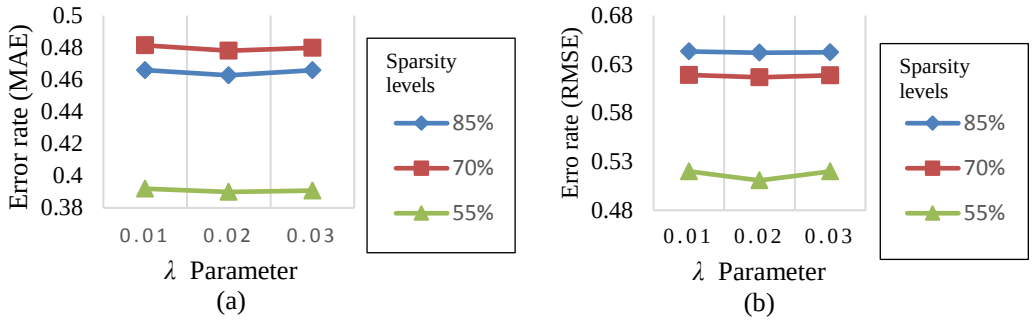

Fig. 13. Comparison with different λ parameters (a) MAE, (b) RMSE

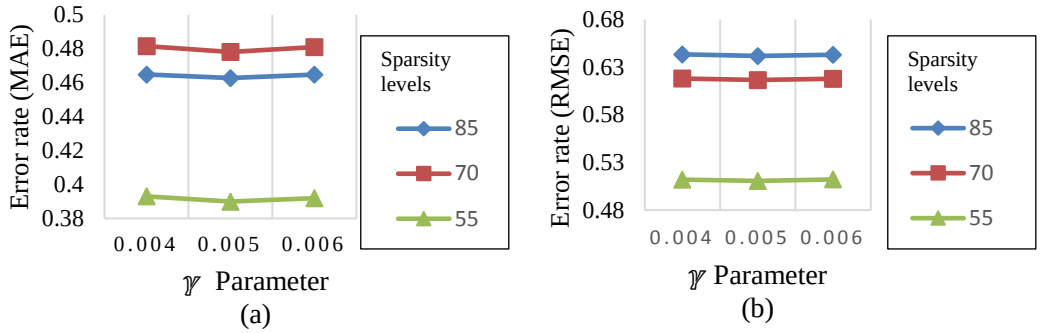

Fig. 14. Comparison with different γ parameters (a) MAE, (b) RMSE

Analyzing the above all evaluations, only a 70% sparsity level of MAE evaluation has a small deviation with 85%. However, this is not highly impacted by our performance measurement because we also calculated RMSE and this kind of deviation for 70% did not occur.

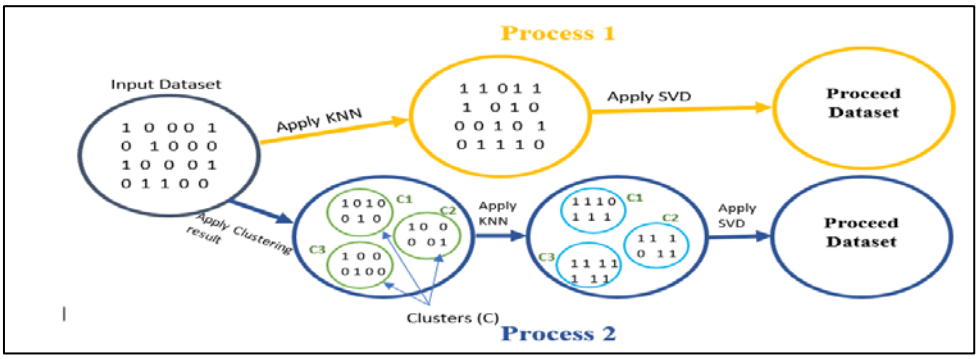

Fig. 15. Example process of the KNN+SVD and clustering+KNN+SVD approaches

A lot of existing approaches used KNN and SVD algorithms for recommendation [35]. However, the new point of our method is we used ontology-based clustering results with the KNN and SVD algorithms. In **Fig. 15**, Process 1 (KNN+SVD) applied KNN and SVD algorithms only and it is applied to the whole dataset directly. In Process 2 (clustering+KNN+SVD), applied clustering results before applying the KNN and SVD algorithms. Here, firstly dataset was clustered according to the different domains using the clustering result. And then KNN applied for the separate clusters rather than applying to the whole dataset. Then applied SVD for whole dataset. According to the evaluation in **Fig. 7**, Process 2 (clustering+KNN+SVD) shows the better performance. Here, identifying the domain first by the clustering help to improve the result. Process 2 reduced Process 1 error rate by nearly 2.5%. In **Fig. 15**, 1 used to represent the rated values, and 0 used to represent the non-rated values.

6. Conclusion and Future Work

Having CF problems such as data sparsity and cold-start problems in existing Web service recommendation approaches shows a low performance. In this paper, our objective is to improve the recommendation result by solving the above CF problems. To achieve this objective, we used the specificity aware ontology-based clustering method combined with the KNN algorithm. This method reduced the data sparsity and cold-start problems of the user rating matrix. After that, we used the matrix factorization-based SVD algorithm for rating prediction and those predictions were used to arrive at the recommendation results. The evaluation shows that our approach has better performance with a minimum MAE and RMSE rate by comparing the existing methods and different performance evaluation methods.

In future research, we plan to apply SVD++ to improve recommendation performance and solve other CF problems such as scalability and synonymy, and simultaneously we would like to link our ontology-based clustering method with deep learning to improve the recommendation results.

References

[1] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 6, pp. 734–749, 2005. Article (CrossRef Link).

[2] R. A. H. M. Rupasingha and I. Paik, "Alleviating sparsity by specificity-aware ontology-based clustering for improving web service recommendation," *IEEJ Trans. Electr. Electron. Eng.*, vol. 14, no. 10, pp. 1507–1517, 2019. Article (CrossRef Link).

[3] X. Han, L. Tian, M. Yoon, and M. Lee, "A Big Data Model Supporting Information Recommendation in Social Networks," in *Proc. of 2012 Second International Conference on Cloud and Green Computing*, pp. 810–813, 2012. Article (CrossRef Link).

[4] L. Li, W. Chu, J. Langford, and R. E. Schapire, "A contextual-bandit approach to personalized news article recommendation," in *Proc. of 19th Int. Conf. World Wide Web, WWW '10*, pp. 661–670, 2010, Article (CrossRef Link).

[5] Z. Jia, Q. Feng, and J. Yu, "A service model based on recommendation trust in pervasive computing environment," in *Proc. of ICPCA10 - 5th Int. Conf. Pervasive Comput. Appl.*, pp. 393–397, 2010. Article (CrossRef Link).

[6] P. Shoval, V. Maidel, and B. Shapira, "An Ontology- Content-based Filtering Method," vol. 15, pp. 303–314, 2008.

[7] R. A. H. M. Rupasingha and I. Paik, "Evaluation of Web Service Recommendation Performance via Sparsity Alleviating by Specificity-Aware Ontology-Based Clustering," in *Proc. of 2018 9th Int. Conf. Aware. Sci. Technol. iCAST 2018*, pp. 279–284, 2018. Article (CrossRef Link).

[8] K. I. Ghauth and N. A. Abdullah, "Learning materials recommendation using good learners' ratings and content-based filtering," *Educ. Technol. Res. Dev.*, vol. 58, no. 6, pp. 711–727, 2010. Article (CrossRef Link).

[9] R. A. H. M. Rupasingha, I. Paik, B. T. G. S. Kumara, and T. H. A. S. Siriweera, "Domain-aware web service clustering based on ontology generation by text mining," in *Proc. of 7th IEEE Annu. Inf. Technol. Electron. Mob. Commun. Conf. IEEE IEMCON 2016*, October, 2016. Article (CrossRef Link).

[10] B. T. G. S. Kumara, I. Paik, W. Chen, and K. H. Ryu, "Web service clustering using a hybrid term-similarity measure with ontology learning," *Int. J. Web Serv. Res.*, vol. 11, no. 2, pp. 24–45, 2014. Article (CrossRef Link).

[11] B. T. G. S. Kumara, I. Paik, H. Ohashi, Y. Yaguchi, and W. Chen, "Context-Aware Filtering and Visualization of Web Service Clusters," in *Proc. of 2014 IEEE International Conference on Web Services*, pp. 89–96, 2014. Article (CrossRef Link).

[12] R. A. H. M. Rupasingha and I. Paik, "Calculating Web Service Similarity using Ontology Learning with Machine Learning," in *Proc. of 2015 IEEE Int. Conf. Comput. Intell. Comput. Res. IEEE*, pp. 1-8, 2015.

[13] R. A. H. M. RUPASINGHA, I. PAIK, and B. T. G. S. KUMARA, "Specificity-Aware Ontology Generation for Improving Web Service Clustering," *IEICE Trans. Inf. Syst.*, vol. E101.D, no. 8, pp. 2035–2043, Aug. 2018. Article (CrossRef Link).

[14] R. A. H. M. Rupasingha, I. Paik, and B. T. G. S. Kumara, "Improving Web Service Clustering through a Novel Ontology Generation Method by Domain Specificity," in *Proc. of 2017 IEEE 24th Int. Conf. Web Serv. ICWS 2017*, pp. 744–751, 2017. Article (CrossRef Link).

[15] S. Zhang, X. Li, M. Zong, X. Zhu, and D. Cheng, "Learning k for kNN Classification," *ACM Trans. Intell. Syst. Technol.*, vol. 8, no. 3, 2017. Article (CrossRef Link).

[16] E. R. Nicola Barbieri, Giuseppe Manco, *Probabilistic Approaches to Recommendations Synthesis Lectures on Data Mining and Knowledge Discovery*, 2014.

[17] Y. Hu, Q. Peng, and X. Hu, "A time-aware and data sparsity tolerant approach for Web service recommendation," in *Proc. of 2014 IEEE Int. Conf. Web Serv. ICWS 2014*, pp. 33–40, 2014. Article (CrossRef Link).

[18] A. Paterek, "Improving regularized singular value decomposition for collaborative filtering," in *Proc. of KDD Cup Work.*, pp. 2–5, 2007.

[19] X. Chen, X. Liu, Z. Huang, and H. Sun, "RegionKNN: A scalable hybrid collaborative filtering algorithm for personalized web service recommendation," in *Proc. of ICWS 2010 - 2010 IEEE 8th Int. Conf. Web Serv.*, pp. 9–16, 2010. Article (CrossRef Link).
[20] K. Onuean, "An Improved Items Recommendation for Memory-Based Collaborative Filtering Technique," vol. 27, no. 3, pp. 51–58, 2019.
[21] C. Ramesh, K. V. C. Rao, and A. Govardhan, "Ontology based web usage mining model," in *Proc. of Int. Conf. Inven. Commun. Comput. Technol. ICICCT 2017*, pp. 356–362, 2017. Article (CrossRef Link).
[22] M. G. Vozalis and K. G. Margaritis, "Applying SVD on item-based filtering," in *Proc. of 5th International Conference on Intelligent Systems Design and Applications (ISDA'05)*, pp. 464–469, 2005. Article (CrossRef Link).
[23] A. Haghighi, M. S. Shadloo, and A. Maleki, "applied sciences Using Committee Neural Network for Prediction of Pressure Drop in Two-Phase Microchannels,".
[24] Z. Al-Zanbouri and C. Ding, "Data-Aware Web Service Recommender System for Energy-Efficient Data Mining Services," in *Proc. of IEEE 11th Int. Conf. Serv. Comput. Appl. SOCA 2018*, pp. 57–64, 2019. Article (CrossRef Link).
[25] Y. Koren, R. Bell, and C. Volinsky, "C OV ER F E AT U RE MATRIX TECHNIQUES FOR," pp. 30–37, 2009.
[26] S. Zhang, L. Yao, A. Sun, and Y. Tay, "Deep learning based recommender system: A survey and new perspectives," *ACM Comput. Surv.*, vol. 52, no. 1, pp. 1–35, 2019. Article (CrossRef Link).
[27] Y. Chen, "Solving the Sparsity Problem in Recommender Systems Using Association Retrieval," vol. 6, no. 9, pp. 1896–1902, 2011. Article (CrossRef Link).
[28] L. Li, F. Lee, B. Chen, and S. Chen, "A Simplified Method for Improving the Performance of Product Recommendation with Sparse Data," no. iCAST, pp. 318–323, 2017.
[29] A. K. Sahu and P. Dwivedi, "User profile as a bridge in cross-domain recommender systems for sparsity reduction," *Appl. Intell.*, vol. 49, no. 7, pp. 2461–2481, 2019. Article (CrossRef Link).
[30] R. Agrawal, "K-Nearest Neighbor for Uncertain Data," *Int. J. Comput. Appl.*, vol. 105, no. 11, pp. 975–8887, 2014.
[31] H. J. Ahn, "A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem," *Inf. Sci. (Ny).*, vol. 178, no. 1, pp. 37–51, 2008. Article (CrossRef Link).
[32] L. Qi, W. Dou, and X. Zhang, "An inverse collaborative filtering approach for cold-start problem in web service recommendation," in *Proc. of ACM Int. Conf. Proceeding Ser.*, pp. 1-9, 2017. Article (CrossRef Link).
[33] N. Hug, "Surprise, a Python library for recommender systems," 2017. [Online]. Available: https://surpriselib.com
[34] M. L. Solvang, "Video Recommendation Systems Finding a Suitable Recommendation Approach for an Application Without Sufficient Data," August, 2017.
[35] L. Susan, "How Did We Build Book Recommender Systems in An Hour Part 2 — kNN and Matrix Factorization,". [Online]. Available: https://towardsdatascience.com/how-did-we-build-book-recommender-systems-in-an-hour-part-2-k-nearest-neighbors-and-matrix-c04b3c2ef55c

Amith M. Weerasinghe is an undergraduate student at the department of Computing and Information Systems, Faculty of Applied Sciences, Sabaragamuwa University of Sri Lanka. His research interests in recommendation systems.

Rupasingha A. H. M. Rupasingha received her bachelor's degree in 2013 from Sabaragamuwa University of Sri Lanka. She obtained her master's and the PhD degree in 2016 and 2019, respectively from the School of Computer Science and Engineering, the University of Aizu, Japan. Her research interests include Web service clustering, ontology learning, and recommendation systems.