An Empirical Framework for Recommendation-based Location Services Using Deep Learning

Vinita Rohilla
Department of Computer Science and Engineering, School of Engineering and Technology, Sharda University, Greater Noida, India and
Department of CSE, Maharaja Surajmal Institute of Technology, New Delhi, India
vinita_rohilla@msit.in

Mandeep Kaur
Department of Computer Science and Engineering
School of Engineering and Technology
Sharda University
Greater Noida, India
mandeep.kaur@sharda.ac.in

Sudeshna Chakraborty
Department of Computer Science and Technology
Lloyd Institute of Engineering and Technology
Greater Noida, India
sudeshna2529@gmail.com

Received: 8 June 2022 | Revised: 8 July 2022 | Accepted: 10 July 2022

Abstract-The large amount of possible online services throws a significant load on the users' service selection decision-making procedure. A number of intelligent suggestion systems have been created in order to lower the excessive decision-making expense. Taking this into consideration, av RLSD (Recommendation-based Location Services using Deep Learning) model is proposed in this paper. Alongside robustness, this research considers the geographic interface between the client and the service. The suggested model blends a Multi-Layer-Perceptron (MLP) with a similarity Adaptive Corrector (AC), which is meant to detect high-dimensional and non-linear connections, as well as the location correlations amongst client and services. This not only improves recommendation results but also considerably reduces difficulties due to data sparseness. As a result, the proposed RLSD has strong flexibility and is extensible when it comes to leveraging context data like location.

Keywords-location services; recommendation system; deep learning; similarity corrector

I. INTRODUCTION

Numerous corporations and organizations have started to wrap their created business apps into more and more widespread multiple online services [1-4]. Many lightweight recommendation methods have also been created to reduce the expenses of consumer decision-making. Generally, by evaluating available client data, a recommender system can deduce a person's approximated tastes and preferences and then offer reliable suggestions [5-9]. It efficiently captures the relevant information and preferences of service clients. Location is an important component in decision recommendations since the Quality of Service (QoS) is sometimes significantly dependent on client or service location. Moreover, location-aware QoS data frequently contain sensitive user information, such as a user's most recently requested service set, the amount of time it takes a user to summon a services located in a nation, etc. Service-Oriented Architecture (SOA) is commonly employed in decentralized computing environments [10], including cloud computing and mobile computing. Location-based services are more successful to provide recommendations based on user's interests [12, 13].

Collaborative Filtering (CF) is a popular way for creating a recommendation system. In recommendation systems, the current user-based CF mechanism seeks for a comparable user based on their prior behavior. The ideas must be according to the user's interest. For example, a user's visited locations are denoted by pins of various forms. The 4 customers check in at 9 distinct locations: 'Exhibition,' 'Hotel,' 'Campground,' 'Shopping Centre,' 'Cineplex,' 'Opera Cinema,' 'Cafe,' 'Library,' and 'College.' The CF approach often creates a similarity measure between various users and offers suggestions based on the maximum value. Let's create a suggestion for customer A1. We could observe that user A1s activities are more comparable to users A2s activities because both went into 'Cineplex,' 'Opera Cinema,' and 'Cafe.' Based on the activities of user A2, the CF technique now recommends 'Library' to user A1. Unfortunately, this is not the greatest fit for user A1 when user A1's visiting information indicates a preference in 'Entertainment' venues, like 'Gallery.' To address this constraint, we will construct an interested profile for every customer so that suggestions may be made as needed.

The following are the major contributions to this study:

- An RLSD framework is developed, which blends an MLP with a similarity Adaptive Corrector in a new way.

Corresponding author: Vinita Rohilla

- The performance of the suggested strategy was assessed and compared with 9 advanced options.
- The proposed technique not only improves recommendation performance, but it also significantly reduces difficulties due to data sparseness.

II. LITERATURE SURVEY

Authors in [14] focused mostly on the cold-start dilemma for service advice depending on user and service locations. To resolve this concern, they offered a service recommendation approach based on CF and geo location, which takes into account the characteristics of services at the edges, movement, and customer requests over time. To begin, they used a multidimensional weighted approach to synthesize the service quality for every dimension in different temporal chunks. In respect of recommendations accuracy, the empirical findings suggest that their multidimensional inverse similarity recommendation method (MDITCF) built on time-aware CF surpasses the Inverted CF recommendation method. Authors in [15] suggested a novel deep CF approach to service recommendations, dubbed Location-aware Deep CF (LDCF). These main advancements included in this approach are: 1) the location attributes are located into high-dimensional dense imbed matrices. MLP encapsulates the HD and non - linear features. Similarity ACs are integrated in the output nodes to first rectify the forecasting QoS. With this, LDCF not only understands the high-dimension and non-linear correlation among the user and services, but it can also dramatically reduce data sparseness. Extensive studies on a real-world Web service [15] database show that LDCF's recommendations efficiency clearly exceeds 9 state-of-the-art services recommendations approaches. Authors in [16] used location information in the factorization machine for QoS predictions, while contextualized information helps the CF similarity calculation. These computations understand the lower dimension and linear characteristics of customers and service providers. Authors in [17] suggested a tailored location-aware CF approach for web service recommendations. Picking comparable neighbors for the user based service the suggested technique takes advantage of both customer location and online services. The technique also incorporated an improved similarity metric for user and web services through accounting for their customized contribution. The test findings demonstrate that this methodology greatly increases QoS predictive performance and computing efficiency when compared to earlier CF-based systems.

III. PROPOSED METHODOLOGY

The major problem in recommending services is estimating QoS. CF is the most extensively utilized QoS predicting technology currently available. Conventional CF techniques, on the other hand, suffer from two flaws: 1) The similarity modeling approach used by conventional CF-based techniques can only gain knowledge of low-dimension and linear characteristics or features from previous networks of user and service and 2) the commonly used data sparseness dilemma in reality has a massive effect on recommendation systems' efficiency. This section provides a detailed description of the proposed model and methodology. The section briefly describes the loss function, the optimizer, and the algorithm used in this work.

A. *Deep Neural Networks*

A Deep Neural Network (DNN) is one type of Neural Network with more than 2 layers: an input layer, at least one hidden layer, and an output layer. In a process known as "feature hierarchy," every layer conducts certain kinds of sorting and ordering. Handling with unorganized or unstructured input is among the most important applications of highly powerful neural network models [18].

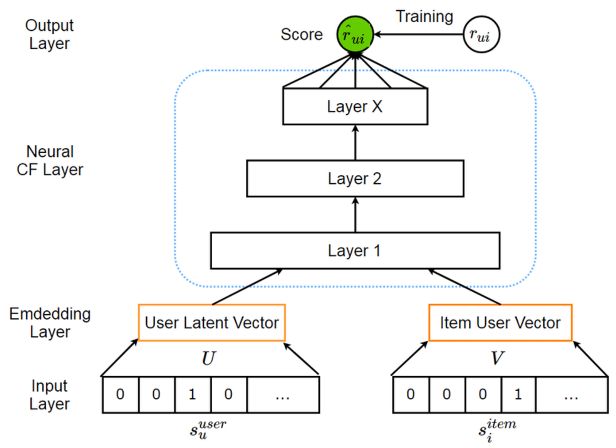

Fig. 1. Recommendation based location services based on deep learning architecture.

The RLSD model is a multiple-layered feed forward neural network with 3 main layers. An Input Layer is used to create the input variables needed from the Intermediate Layer and the similarities needed for the Output Nodes. The Intermediate Layer is employed for centralized training to ensure getting high-dimension and non-linear characteristics.

1) *Input Layer*

The Input layer's primary function is to regulate the initial Input. The Input contains user identification, location-based info, service identifiers, and service geo locations in Keras1's Embedding Layer, that may be thought as a special fully-connected layer lacking bias terms. Essentially, the Embedded Layer uses one-hot encoding on the inputs to build a '0' vector with a given dimension, while the vector's *i*-th point is set to 1 [19]. The category characteristics are translated to high-dimension fully connected embedding vectors using this procedure. The mapping procedure is depicted in (1) - (4):

$$I_u^k = f_1(P_1^T i_u + b_1) \quad (1)$$

$$G_u^k = f_1(P_1^T g_u + b_1) \quad (2)$$

$$I_s^k = f_1(Q_1^T i_s + b_1) \quad (3)$$

$$I_u^k = f_1(Q_1^T g_s + b_1) \quad (4)$$

where i_u and i_s are the user's identifier and the service's identifier, correspondingly, g_u and g_s are the initial inputs for the user's and service's location, P_1 and Q_1 are user's and service's embedding weighted matrices, b_1 is a bias term initialized to 0, f_1 is the embedding layer's activation function, which is considered as the identity function for this work, I_u^k and G_u^k are the *k*-dimensional user_id and location embedding vector correspondingly, and U and S are the *k*-dimensional service_id and location embedding vector correspondingly.

These identifying feature vectors representations are integrated with the appropriate location features set to generate the user feature map and service feature map. Next, we append both feature maps to generate the intermediate layers input vector sequence. The equation is written in the following form:

$$U = \phi(I_u^k, G_u^k) = \begin{bmatrix} I_u^k \\ G_u^k \end{bmatrix} \quad (5)$$

$$S = \phi(I_s^k, G_s^k) = \begin{bmatrix} I_s^k \\ G_s^k \end{bmatrix} \quad (6)$$

$$X = \phi(\mathrm{U}, \mathrm{V}) = \begin{bmatrix} u \\ v \end{bmatrix} \quad (7)$$

where Φ is the mergence operation, U and S are user and service embedding vectors, and X is the input vector.

An Adaptive Corrector (AC) that computes similarities among user-location embedding and service-location embedding is present. The AC employs a similarity approach, including location similarities among consumers and services into the neural network's forward propagation phase. As a result, the AC bridges gaps across deep learning with CF. The AC's operation outcome is immediately transferred to the Output Nodes, bypassing the Intermediate Layer. The AC is adaptable and may be used for all other similarity computations like cosine and Euclidean similarity. Equations (8) and (9) describe the procedure:

$$O^{AC} = \mathrm{cosider}(G_u, G_s) = \frac{G_u \cdot G_s}{||G_u||\,||G_u||} \quad (8)$$

$$O^{AC} = \mathrm{cosider}(G_u, G_s) = \sqrt{G_u \cdot G_s} \quad (9)$$

where O^{AC} is the AC similarity output. Cosine and Euclidean similarities are both present in this. When no specific comment is made, (8) is applied to determine the cosine similarity outcome of the AC.

2) *Intermediate Layer*

The Intermediate Layer is responsible for processing the input vectors from the Input Nodes in order to capture non-linear information. The high-dimensional non-linear correlation among consumers and services is learned using a densely integrated MLP architecture in this work. First but foremost, we must select the non-linear activation. The activation function SeLU (Scaled Exponential Linear Unit) presents numerous benefits. It allows building a map featuring qualities that led to SNNs (Self-normalized Neural Networks). Secondly, in order for the DNN to train and learn many attributes, the network structure must adhere to the standard tower architecture [20]. Lastly, to avoid over fitting, we employ L2 regularization on weights. In the Intermediate Layer, the forward propagating of the input vector sequence is described by:

$$o_2^{mlp} = f_2(W_2^T x + b_2) \quad (10)$$

$$o_2^{mlp} = f_1(W_1^T f_2 o_{i-1}^{mlp} + b_1), I = 3, 4, \ldots, N\text{-}1 \quad (11)$$

$$o^{mlp} = o_{n-1}^{mlp} \quad (12)$$

where o_{n-1}^{mlp} is the *i*th layer of the Intermediate Layer's output, b_1 the Intermediate Layer bias term, and o^{mlp} the Intermediate Layer's output.

3) *Output Layer*

The Output Layer is largely responsible for producing the overall forecast results. The FRLS paradigm divides users and services into two paths. We immediately combine the outcomes of these two paths, as in [20, 21]. We mix the similarity outcome o^{AC} with the output vector o^{mlp} to create a new output vector sequence *o*. Finally, FRLS makes the final projections using a single-layered neural network. Gaussian distribution is used to initialize the parameters of this layer, as illustrated in (13)-(14):

$$o = \phi(o^{AC}, o^{mlp}) = \begin{bmatrix} o^{AC} \\ o^{mlp} \end{bmatrix} \quad (13)$$

$$\hat{Q}_{u,s} = f_n(W_n^T o + b_n) \quad (14)$$

where $\hat{Q}_{u,s}$ is the predicted QoS values of user *u* raising the *s* service, and b_n is the ordinary identity function that signifies the activation function of the last layer.

We get a typical matrix factorization network base Gaussian Matrix Factorization (GMF) distribution on one side and an MLP network on the other.

B. *Multi-Layer Perceptron Networks*

An MLP network is simply a formal moniker for the most fundamental type of DNN. An MLP network is made up of 5 dense hidden layers and the SeLu activation function. We began by embedding the user and items, then flattened and appended them. Afterwards, we added the first dropout layer, followed by 4 hidden layers with 64 neurons, 32 neurons, 16 neurons, 8 neurons, and finally one densely integrated neuron. Then a dropout and batch normalization layer was added between hidden layers 1 and 2 and 2 and 3. Lastly, we got our final output layer, which is nothing more than a one fully-connected neuron. We utilized log cosh as the error function and Nadam as optimizer.

C. *Matrix Factorization Networks*

A Gaussian MF networks is essentially a conventional point-wise MF that uses Stochastic Gradient Decent (Adam with this context) to approximate the factoring of a matrix (user×item) into two matrices, i.e. user×latent feature and latent feature×item.

D. *Combining the two Networks: NeuMF*

Finally, the Matrix factorization and MLP network are merged into the NeuMF network. Essentially, this merely appends the results of two networks. The two networks are then fused, and a single output node is added to the newly integrated model. From this, the configuration is like above, with a binary log cosh loss and the Nadam optimizer.

To improve the accuracy of representing probability distributions, many variants have been proposed. An alternative approach for improvement is through the reconstruction loss:

$$f(t: a) = \frac{1}{a}\log(\cosh(at)) = \frac{1}{a}\log\frac{e^{at}+e^{-at}}{2} \quad (15)$$

Nadam optimizer can be written as:

$$w_{t+1} = w_t - \frac{\alpha}{\sqrt{\hat{v}_t}+\epsilon}\left(\beta_1\hat{m}_{t-1} + \frac{1-\beta_1}{1-\beta_1^t}\cdot\frac{\partial L}{\partial w_t}\right) \quad (16)$$

Nesterov is used by Nadam optimizer to modify the gradients 1 step forward by substituting the prior m_vec (m_{t-1}) in (16) with the current m_vec (m_t):

$$w_{t+1} = w_t - \frac{\alpha}{\sqrt{\hat{v}_t}+\epsilon}\left(\beta_1\hat{m}_{t} + \frac{1-\beta_1}{1-\beta_1^t}\cdot\frac{\partial L}{\partial w_t}\right) \quad (17)$$

Algorithm 1: RLSD (Recommendation based location services using the Deep Learning Algorithm. Required: user_service request matrix (*Rm*), density of matrix (*Dm*), DNN topology design (*t*), learning rates (*lr*), decayed ratios (*Dr*), no. of iterations (*Ni*).

Output: Weighted matrix and the bias P_1, Q_1, W_1, …, W_n, b_1, b_2, …, b_n.

Strategy:

```
step 1. Scarce Rm based on Dm
step 2. Make training inputs entries for $R_{train}$ and test input entries $R_{test}$
step 3. Create input attributes/features $i_u$, $g_u$, $i_s$, $g_s$;
step 4. Construct the DNN using t and (10)-(12)
step 5. Set $P_1$, $Q_1$, $W_1$, …, $W_n$ according to the Gaussian distribution;
step 6. Set $b_1$, $b_2$, …, $b_n$ to 0;
step 7. For each epoch = 1 to Ni do
step 8. For each user and service in $R_{train}$ do
step 9. Create embedding vectors sequence using (1)-(4);
step 10.Create input vector sequence using (5)-(7)
step 11.Create the AC output from either (8) or (9)
step 12.Make prediction, $\hat{Q}_{u,s}$ using (13)-(14)
step 13.end for
step 14.Give lr and Dr to Nadam optimizer
step 15.Modify parameters of the model using the Nadam minimization (15)
step 16.for each user and service in $R_{test}$ do
step 17.Use (20)-(21) to measure the performance of the model
step 18.end for
step 19.end for
```

IV. Results and Discussion

This section discusses the experimentation results for location-based services on the WS-Dream dataset using Python programming. To assess the effectiveness of the proposed method, two evaluation measures are used.

A. *Dataset*

We ran tests upon the WS-Dream dataset which is a substantial Web services dataset gathered and managed in [22], which consists of 19,74,675 QoS values from 339 users on 5,825 services. This dataset includes information on users' and services' locations. QoS data are presented in this work as a user-service matrix, with the row indicator representing the user identifier and the column indicator representing the services identifier. This work utilized response time and throughput as inputs to the RLSD.

B. *Pre-processing*

This research makes use of two spatially relevant dataset features/attributes: Country Name (CN) and Autonomous System Number (ASN). Users were spread throughout 30 nations and 136 ASNs, whereas services were spread across 990 ASNs across 73 countries, according to database figures. For CN, we utilized Sklearn3's category encoding to convert the classed characteristics into numerical embeddings, such that each classified attribute is expressed by a nation code. The data can be viewed like after pre-processing:

$$U = (U\_ID, UASN, UCN) \quad (18)$$

$$S = (S\_ID, SASN, SCN) \quad (19)$$

where *U* and *S* are the user input embedding vector and service input embedding vector correspondingly, U_ID is the user identifier, UASN is an autonomous system numeric user code, UCN is the nationwide user code, and S_ID, SASN, and SCN are the same as above.

C. *Parameter Setting*

For CF methods (UPCC, LACF, IPCC, RegionKNN, etc.),the best-k value is set to 10, the learning rate is fixed at 1e^{-3}, the total of implicit feature factors (size) is fixed to 10, the number of total iterations is fixed to 300, the regularization variables are fixed to 0.1, and the randomized factor for the rarefy matrix is fixed to 7. Deep learning techniques (NCF, RLSD), can be developed using the Keras API, wherein the Gaussian distribution function is utilized (average is 0, stdev is 0.01) to set the model hyper parameters and (15) to modify the hyper parameters. Then we configured the batch size to 256, learning rate to 1e^{-4}, 4 MLPs, and Nadam optimizer.

D. *Evaluation Metrics*

The recommendation success of the specified approach is measured using two fundamental statistical evaluation metrics: Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE):

$$\text{MAE} = \frac{\sum_{u,s}|Q_{u,s} - \hat{Q}_{u,s}|}{N} \quad (20)$$

$$\text{RMSE} = \frac{1}{N}\sum_{u,s}(Q_{u,s} - \hat{\hat{Q}}_{u,s})^2 \quad (21)$$

where $Q_{u,s}$ is the QoS initial value for the user u raising the service s, $\hat{\hat{Q}}_{u,s}$ is the QoS prediction value u raising the service s, and N is the overall QoS value.

Tables I and II show the obtained results by the algorithms performed in [21], with which the proposed model of this paper is compared to. The values in the Tables show that the proposed model is better in MAE and RMSE for response time (RT) and TP (throughput).

TABLE I. RESPONSE TIME EXPERIMENTAL RESULTS

Approach	Density									
	0.5		0.10		0.15		0.20		0.25	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
UMEAN	0.876	0.853	0.873	1.857	0.874	1.857	0.873	1.858	0.874	1.858
IMEAN	0.703	0.567	0.686	1.542	0.684	1.533	0.681	1.529	0.680	1.525
UPCC	0.634	0.377	0.553	1.311	0.511	1.258	0.483	1.220	0.467	1.189
IPCC	0.633	1.397	0.591	1.341	0.507	1.258	0.454	1.208	0.431	1.175
UIPCC	0.624	1.386	0.579	1.328	0.498	1.247	0.448	1.197	0.425	1.165
RegionKNN	0.594	1.641	0.577	1.637	0.569	1.627	0.569	1.617	0.562	1.619
LACF	0.682	1.500	0.650	1.468	0.610	1.416	0.582	1.381	0.562	1.357
PMF	0.568	1.537	0.487	1.321	0.451	1.221	0.430	1.171	0.416	1.139
NCF	0.440	1.325	0.385	1.283	0.372	1.253	0.362	1.205	0.349	1.138
LDCF	0.246	0.7398	0.2787	0.895	0.2291	0.7573	0.2370	0.8004	0.2585	0.8819
RLSD (proposed)	0.1765	0.4977	0.1699	0.530	0.1892	0.6155	0.1831	0.5977	0.1710	0.5643

TABLE II. THROUGHPUT EXPERIMENTAL RESULTS

Approach	Density									
	0.5		0.10		0.15		0.20		0.25	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
UMEAN	54.333	110.296	53.947	110.345	53.971	110.201	53.906	110.190	53.862	110.194
IMEAN	27.342	65.844	26.962	64.843	26.757	64.266	26.669	64.069	26.595	63.873
UPCC	27.559	60.757	22.687	54.598	20.525	50.906	19.243	48.834	18.253	47.135
IPCC	27.102	62.665	26.270	60.479	25.487	57.561	23.726	54.564	22.286	52.293
UIPCC	27.070	60.510	22.440	54.506	20.256	50.585	18.888	48.238	17.863	46.392
RegionKNN	26.857	69.614	25.352	68.015	24.947	67.365	24.687	66.923	24.746	66.831
LACF	27.419	65.770	24.847	62.057	22.943	58.816	21.562	56.507	20.587	54.785
PMF	18.943	57.020	16.004	47.933	14.668	43.642	13.988	41.652	13.398	40.025
NCF	15.468	49.703	13.616	46.034	12.284	42.317	11.833	41.263	11.312	39.534
LDCF	17.629	57.746	14.711	50.730	13.952	48.871	13.199	47.323	12.332	45.142
RLSD (proposed)	19.517	61.822	16.154	53.344	14.958	50.681	14.388	49.393	14.332	49.195

Figures 2 and 3 show the loss values obtained by both models in response time and throughput respectively. The lesser the loss value, the more accurate the simulation (unless the model was over fitted to the training set). The loss value can be computed during training and testing and is used to determine how good the model performs for the two different sets. Loss, like accuracy, is also not a proportion. It is the summarization of the errors incurred in training and test sets with each sample. It is clearly visible that the proposed RLSD model performed better than the existing methods by reducing the loss values in both response time and throughput.

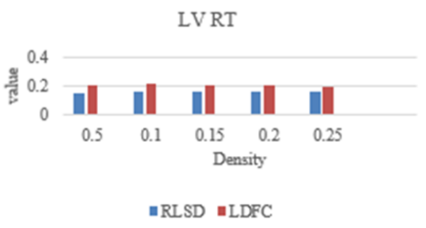

Fig. 2. Loss values of response time.

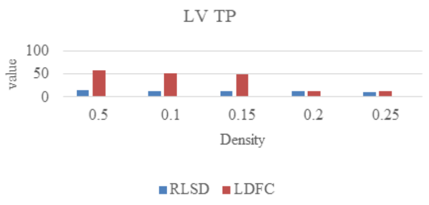

Fig. 3. Loss values of throughput.

V. CONCLUSION

A recommendation system is the backbone for retrieving and processing information that aims to forecast user interests. This study offers a novel deep learning-based approach called RLSD to provide recommendation services to predict QoS which incorporates a similarity AC to improve the prediction values. In the future, we plan to analyze and solve additional privacy concerns in location service recommendations, as well as improve the performance, using a parallel and distributed recommendations system to manage the required massive data. This model may also improve the results of [15, 23-29].

REFERENCES

[1] Y. Wang, Z. Cai, Z.-H. Zhan, Y.-J. Gong, and X. Tong, "An Optimization and Auction-Based Incentive Mechanism to Maximize Social Welfare for Mobile Crowdsourcing," *IEEE Transactions on Computational Social Systems*, vol. 6, no. 3, pp. 414–429, Jun. 2019, https://doi.org/10.1109/TCSS.2019.2907059.

[2] Y. Zhang, Y. Zhou, F. Wang, Z. Sun, and Q. He, "Service recommendation based on quotient space granularity analysis and covering algorithm on Spark," *Knowledge-Based Systems*, vol. 147, pp. 25–35, May 2018, https://doi.org/10.1016/j.knosys.2018.02.014.

[3] S. Zhang, X. Li, Z. Tan, T. Peng, and G. Wang, "A caching and spatial K-anonymity driven privacy enhancement scheme in continuous location-based services," *Future Generation Computer Systems*, vol. 94, pp. 40–50, May 2019, https://doi.org/10.1016/j.future.2018.10.053.

[4] S. Zhang, G. Wang, and Q. Liu, "A Dual Privacy Preserving Scheme in Continuous Location-Based Services," in *IEEE Trustcom/BigDataSE/ ICESS*, Sydney, NSW, Australia, Aug. 2017, pp. 402–408, https://doi.org/10.1109/Trustcom/BigDataSE/ICESS.2017.264.

[5] L. Qi, R. Wang, C. Hu, S. Li, Q. He, and X. Xu, "Time-aware distributed service recommendation with privacy-preservation," *Information*

Sciences, vol. 480, pp. 354–364, Apr. 2019, https://doi.org/10.1016/j.ins.2018.11.030.

[6] W. Dou, X. Zhang, J. Liu, and J. Chen, "HireSome-II: Towards Privacy-Aware Cross-Cloud Service Composition for Big Data Applications," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 2, pp. 455–466, Oct. 2015, https://doi.org/10.1109/TPDS.2013.246.

[7] Y. Xu, L. Qi, W. Dou, and J. Yu, "Privacy-Preserving and Scalable Service Recommendation Based on SimHash in a Distributed Cloud Environment," *Complexity*, vol. 2017, Dec. 2017, Art. no. e3437854, https://doi.org/10.1155/2017/3437854.

[8] A. Ramlatchan, M. Yang, Q. Liu, M. Li, J. Wang, and Y. Li, "A survey of matrix completion methods for recommendation systems," *Big Data Mining and Analytics*, vol. 1, no. 4, pp. 308–323, Dec. 2018, https://doi.org/10.26599/BDMA.2018.9020008.

[9] Y. Zhang *et al.*, "Covering-Based Web Service Quality Prediction via Neighborhood-Aware Matrix Factorization," *IEEE Transactions on Services Computing*, vol. 14, no. 5, pp. 1333–1344, Sep. 2021, https://doi.org/10.1109/TSC.2019.2891517.

[10] I. M. Delamer and J. L. M. Lastra, "Service-Oriented Architecture for Distributed Publish/Subscribe Middleware in Electronics Production," *IEEE Transactions on Industrial Informatics*, vol. 2, no. 4, pp. 281–294, Aug. 2006, https://doi.org/10.1109/TII.2006.885188.

[11] B. AlBanna, M. Sakr, S. Moussa, and I. Moawad, "Interest Aware Location-Based Recommender System Using Geo-Tagged Social Media," *ISPRS International Journal of Geo-Information*, vol. 5, no. 12, Dec. 2016, Art. no. 245, https://doi.org/10.3390/ijgi5120245.

[12] N. Kumar, A. Hashmi, M. Gupta, and A. Kundu, "Automatic Diagnosis of Covid-19 Related Pneumonia from CXR and CT-Scan Images," *Engineering, Technology & Applied Science Research*, vol. 12, no. 1, pp. 7993–7997, Feb. 2022, https://doi.org/10.48084/etasr.4613.

[13] A. B. S. Salamh and H. I. Akyuz, "A Novel Feature Extraction Descriptor for Face Recognition," *Engineering, Technology & Applied Science Research*, vol. 12, no. 1, pp. 8033–8038, Feb. 2022, https://doi.org/10.48084/etasr.4624.

[14] M. Yu, G. Fan, H. Yu, and L. Chen, "Location-based and Time-aware Service Recommendation in Mobile Edge Computing," *International Journal of Parallel Programming*, vol. 49, no. 5, pp. 715–731, Oct. 2021, https://doi.org/10.1007/s10766-021-00702-5.

[15] N. Kumar and D. Aggarwal, "LEARNING-based Focused WEB Crawler," *IETE Journal of Research*, pp. 1–9, Feb. 2021, https://doi.org/10.1080/03772063.2021.1885312.

[16] Y. Yang, Z. Zheng, X. Niu, M. Tang, Y. Lu, and X. Liao, "A Location-Based Factorization Machine Model for Web Service QoS Prediction," *IEEE Transactions on Services Computing*, vol. 14, no. 5, pp. 1264–1277, Sep. 2021, https://doi.org/10.1109/TSC.2018.2876532.

[17] J. Liu, M. Tang, Z. Zheng, X. Liu, and S. Lyu, "Location-aware and personalized collaborative filtering for web service recommendation," *IEEE Transactions on Services Computing*, vol. 9, no. 5, pp. 686–699, Sep. 2016, https://doi.org/10.1109/TSC.2015.2433251.

[18] U. Khan, K. Khan, F. Hassan, A. Siddiqui, and M. Afaq, "Towards Achieving Machine Comprehension Using Deep Learning on Non-GPU Machines," *Engineering, Technology & Applied Science Research*, vol. 9, no. 4, pp. 4423–4427, Aug. 2019, https://doi.org/10.48084/etasr.2734.

[19] L. Zhang, T. Luo, F. Zhang, and Y. Wu, "A Recommendation Model Based on Deep Neural Network," *IEEE Access*, vol. 6, pp. 9454–9463, 2018, https://doi.org/10.1109/ACCESS.2018.2789866.

[20] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural Collaborative Filtering," in *26th International Conference on World Wide Web*, Perth, Australia, Apr. 2017, pp. 173–182, https://doi.org/10.1145/3038912.3052569.

[21] H.-J. Xue, X.-Y. Dai, J. Zhang, S. Huang, and J. Chen, "Deep matrix factorization models for recommender systems," in *26th International Joint Conference on Artificial Intelligence*, Melbourne, Australia, Aug. 2017, pp. 3203–3209.

[22] Z. Zheng, H. Ma, M. R. Lyu, and I. King, "QoS-Aware Web Service Recommendation by Collaborative Filtering," *IEEE Transactions on Services Computing*, vol. 4, no. 2, pp. 140–152, Apr. 2011, https://doi.org/10.1109/TSC.2010.52.

[23] N. Kumar, M. Gupta, D. Sharma, and I. Ofori, "Technical Job Recommendation System Using APIs and Web Crawling," *Computational Intelligence and Neuroscience*, vol. 2022, Jun. 2022, Art. no. e7797548, https://doi.org/10.1155/2022/7797548.

[24] N. Kumar, "Segmentation based Twitter Opinion Mining using Ensemble Learning," *International Journal on Future Revolution in Computer Science & Communication Engineering*, vol. 3, no. 9, pp. 1–9, Sep. 2017.

[25] N. Kumar and P. Dahiya, "Weighted similarity page rank: an improvement in WPR and WSR," *International Journal of Computer Engineering and Applications*, vol. 11, no. 8, pp. 1–11, 2017.

[26] J. Mor, N. Kumar, and D. Rai, "Effective presentation of results using ranking & clustering in meta search engine," *COMPUSOFT, An international journal of advanced computer technology*, vol. 7, no. 12, Art. no. 2957, 2018.

[27] J. Mor, D. D. Rai, and D. N. Kumar, "An XML based Web Crawler with Page Revisit Policy and Updation in Local Repository of Search Engine," *International Journal of Engineering & Technology*, vol. 7, no. 3, pp. 1119–1123, Jun. 2018, https://doi.org/10.14419/ijet.v7i3.12924.

[28] N. Kumar and P. Singh, "Meta Search Engine with Semantic Analysis and Query Processing," *International Journal of Computational Intelligence Research*, vol. 13, no. 8, pp. 2005–2013, 2017, https://doi.org/10.37622/IJCIR/13.8.2017.2005-2013.

[29] N. Kumar, "Document Clustering Approach for Meta Search Engine," *IOP Conference Series: Materials Science and Engineering*, vol. 225, Dec. 2017, Art. no. 012291, https://doi.org/10.1088/1757-899X/225/1/012291.