Computer Systems Science & Engineering
DOI:10.32604/csse.2022.019507
Article

Tech Science Press

Optimal Data Placement and Replication Approach for SIoT with Edge

B. Prabhu Shankar[1,*] and S. Chitra[2]

[1]Department of Information Technology, Er Perumal Manimekalai College of Engineering, Hosur, 635117, Tamilnadu, India
[2]Department of Computer Science and Engineering, Er Perumal Manimekalai College of Engineering, Hosur, 63117, Tamilnadu, India
*Corresponding Author: B. Prabhu Shankar. Email: prabhuresearch1@yahoo.com
Received: 15 April 2021; Accepted: 31 May 2021

Abstract: Social networks (SNs) are sources with extreme number of users around the world who are all sharing data like images, audio, and video to their friends using IoT devices. This concept is the so-called Social Internet of Things (SIot). The evolving nature of edge-cloud computing has enabled storage of a large volume of data from various sources, and this task demands an efficient storage procedure. For this kind of large volume of data storage, the usage of data replication using edge with geo-distributed cloud service area is suited to fulfill the user's expectations with low latency. The major issue is the way to store the data and replicate these large data items optimally and allocate the request from the data center efficiently. For efficient storage of these data, we use edge server, which is part of the cloud server, in this study. Thus, the data are distributed and stored with quick access, which will reduce the latency with response. The proposed data placement approach learns with machine learning (ML) algorithm called radial basis kernel function assisted with support vector machine (RBF-SVM) to classify the data center for storing the user and friend's data from the SIoT devices. These learning algorithms will be used to predict the workload of the data stored in the data center as either edge or cloud depending on the existing time slots. The data placement with dynamic nature is also optimized using the proposed dynamic graph partitioning (GP) method to meet the individual user's demand of low latency with minimum costs. This way will keep the SIoT data placement efficient and effective over time. Accordingly, this proposed data placement and replication approach introduces three kinds of innovations compared with the existing data placement approach. (i) Rather than storing the user data in a single cloud, this study uses the edge server closest to the SIoT devices for faster access with reduced response time. (ii) The classification algorithm called RBF-SVM is used to find storage for user for reducing data replication. (iii) Dynamic GP is introduced for data placement with reduced latency and minimum cost to fulfil the dynamic nature of the SN. The simulation result of this approach obtains reduced latency of 130 ms and minimum cost compared with those of the existing data placement approaches. Therefore, our proposed data placement with ML-based learning on edge provides promising results in terms of efficiency, effectiveness, and performance with reduced latency and minimum cost.

Keywords: Data placement; data replication; social network; social internet of things; edge computing; cloud computing; graph partitioning; support vector machine; machine learning; radial basis function; latency; storage cost

1 Introduction

Social network (SN) users are dispersed around the world and are having friendship with others from different places. Contents like images, videos, and audios of different sizes are shared among the friends and their users. Popular social media networks like Facebook, YouTube, WhatsApp, and Instagram are used to share larger data content with 2 billion active users per month in Facebook, 1.5 billion active users per month in YouTube, 1.2 billion active users per month in WhatsApp, and 700 million active users per month in Instagram [1]. The users of these SNs expect quality of service, low latency, data availability, and privacy from the service providers. Various cloud service providers with numerous data centers are available, such as Amazon S3 [2], Google Cloud Storage [3], and Microsoft Azure [4], and maintain and manage setup of the aforementioned privacies.

Data items from various locations to the data centers are stored by the SN providers. The use of cloud resources requires payment. The cost to store data items become large when storing huge data with replication. Thus, the users' information needs to be stored in the cloud with reduced replication to save storage cost. For data replication and placement, the user of the Social Internet of Things (SIoT) has the primary copy and secondary replicas that are needed to be stored to ensure access to the data by the user and their friends. The main issue addressed here is the approximate count of number of replicas of the data for the users with their location, which increases the latency and cost. Thus, an optimized data placement algorithm is necessary to satisfy the users' latency with minimum storage cost of the cloud for guaranteeing the service-level agreement. In most applications [5], the replication percentage of 90% is ideal.

This study aims to minimize data replication to save storage cost. For this purpose, our proposed approach uses edge server and machine learning (ML)-based classification to split the user data storage with reduced replication. The proposed approach uses dynamic GP algorithm to reduce the latency of the user, and this algorithm supports the dynamic nature of the SNs that can add or remove the user/friend, add or remove the data center, and create/delete/update the content of the user data or friend data. The main contributions of this study are summarized as follows:

- The proposed approach uses edge server to find the minimum count of replicas for each user information stored in the data center. The user data are distributed and stored in cloud or edge to reduce replication to satisfy the latency requirement and increase the response time of the user.
- ML-based classification algorithm called radial basis kernel function assisted with support vector machine (RBF-SVM) is used to classify the user data for selecting the optimal data center storage. The data center will be chosen as cloud or edge depending on the user data to reduce the storage cost of the cloud.
- An optimized dynamic GP approach is used to guarantee the optimal SN data placement over time. This way will assure the low individual latency to be less than 150 ms for the user who accesses the data and their friends who want to access the data.
- The simulation environment uses Facebook dataset. The classification accuracy on selecting the storage data center is evaluated, and the result proves that RBF-SVM classifier obtains the best accuracy compared with other existing classifiers.
- The experimental results are compared with those of the existing data placement approaches like genetic algorithm (GA)-based data placement, GP-based data placement, and social twin edge with IoT data placement to evaluate the efficiency of the proposed dynamic data placement scheme on

edge. The evaluated results are promising with low latency and minimum cost for the proposed ML-assisted dynamic data placement on edge.

- The proposed data placement and replication scheme is proven to be the best in terms of efficiency, effectiveness, performance, latency, and cost.

The rest of this article is organized as follows. Section 2 reviews the literature related to data replication and placement of SN. Section 3 introduces the innovative approaches of the dynamic data placement learned with ML techniques on edge. Section 4 analyzes the simulation results and comparison of data placement schemes. Section 5 concludes the proposed data placement with future work.

2 Literature Review

This section describes the various studies related to data placement and replication. A novel data placement approach to resolve the dynamic set cover problem was proposed in [6]. This work consisted of the grouping of two algorithms like dynamic greedy and greedy method. Among the SN users, individual latencies with the percentiles of 99.9 were guaranteed minimum cost for the operations to store and synchronize data over time. The comparative study of their approaches obtained 26% cost savings. The optimal storage and replication of the large volume of data items and distribution of the user access requests to various data centers were addressed in [7]. GP algorithm was also used to find the nearest data center, and this algorithm satisfied the latency requirement with minimum monetary cost. They used Facebook dataset for evaluation.

Optimal data placement of social twin edge with edge IoT was proposed in [8]. They used two approaches to reduce the latency like efficient data exchange between physical devices and its respective digital twins (DTs) and the DTs of the friend devices. The chaining procedure and service discovery speed were increased. They formulated the limited sources in the IoT devices tied with edge and cloud. ML-assisted data placement approach was proposed in [9]. The access pattern of the incoming data was predicted using the ML method, and the data were optimized with the data placement algorithm. The data placement method was implemented in hybrid storage medium by similar data and its corresponding features. Their experimental results on File bench dataset showed an improvement of 49% in system performance compared with those of other algorithms.

GA-based data placement and replication was proposed in [10]. GA was used to find the optimal count of replicas and placement of replicas that minimize the cost while fulfilling the latency requirement of the user. They used Facebook dataset for evaluation. Data placement optimization on multi cloud environment was proposed in [11]. They used data balancing techniques of the server with the data from the cloud to overcome the overhead issue. The latency delay was calculated on the basis of the distance between the cloud and the user on transferring the data. They evaluated the proposed work with actual user's locations and times. The obtained results reduced 59% of the resource utilization, decreased 50% of storage units, and presented latency delay of 50 ms.

The data placement on cloud computing using GA was proposed in [12]. The data scheduling based on mathematical model was developed. The generational evolution of the GA produced better estimate of data placement. The experimental results proved that GA produced optimal data placement with reduced data scheduling between the data centers. The new data placement algorithm for SN called balanced distribution of each age group was proposed in [13]. This work used multiple counters located in each storage center. Each counter contained the data of similar age group. The data related to the similar age group were plotted over the various storage centers. The evaluated experiments reduced the I/O load by 30.4% from 11.6%. An ant colony optimization-based data placement and virtual machine allocation was proposed in [14]. The network traffic and bandwidth would be reduced by placing the needed number of

virtual and physical machines. The metaheuristic algorithm called ant colony optimization was used to select the nearest virtual and physical machines. The data stored in the physical storage devices were located in physical machine. Corresponding virtual machines were located depending on the physical machine processing capacity to process the physical machine data. The physical machine was selected with adjacent proximity, and the jobs were executed in virtual machine with better allocation.

Efficient data placement approach used for dynamic and optimal scaling of SN in geo-distributed cloud was proposed in [15]. The numbers of data centers and users were fixed and the number of videos was varied in this method. The issues in data placement of distributed cloud were investigated in [16] to minimize the operational cost. Data placement associated with the co-location of relevant data were proposed in [17]. They used localized data serving method to handle the workload balance of the nodes, and incremental adjustment of replicas was also considered. Lightweight replica data placement was proposed in [18,19]. They examined the latency constraint cost optimization problem in online SN of geo-distributed cloud.

The overview of these works reveals that no comprehensive method can handle all the needs of SN data placement, such as learning by itself for prediction, user addition/deletion, and data replication. As mentioned in Section 1, our proposed work will satisfy all these requirements with reduced latency and minimum cost. Therefore, the method is innovative compared with existing data placement methods.

3 Proposed Methodology

This section describes the proposed methodologies in this study. Three approaches are introduced and compared with the existing data placement approaches. (i) Edge server is introduced to distribute the workload of cloud for quick data access of the user. (ii) The data center for storing the data without replication is classified using the ML algorithm called RBF-SVM. (iii) Data placement using dynamic GP algorithm is utilized. In this study, ML-based data placement algorithm on edge cloud computing is proposed with reduced latency. This algorithm is based on a two-step approach. First, the ML-based algorithms are used to predict the pattern of the incoming data. Thus, the trained neural network will use the result to decide the storage medium that is suitable to store the files without replication. Second, the files are placed into the storage depending on the classifier result assisted by the data placement algorithm to avoid data replication and accessing the user request with low latency. Efficient data placement algorithm called dynamic GP is also used, where the SN is divided into small networks with limited size. The neural network-based learning is used to predict this workload of the user and friends for the future based on the previous time slots. The proposed data placement architecture is shown in Fig. 1.

3.1 System Model

The proposed work addresses the problem of data placement and replication of online SN data services. The monetary cost of the service provider needs to be optimized while using the geo-distributed cloud, and the latency for service users should be provided. Each user has their own replica of the data stored in the data center. The assumption is that each user of SIoT can read their own data from the primary DC and each of their friends also can read their data from the nearest DC which stored secondary replication of the original data. We also consider M edge servers, as shown in Fig. 1, associated with N SIoT devices which are located and connected to SIoT model.

The users in the SIoT are represented as $U = \{u_1, u_2, \ldots, u_N\}$, and each user is associated with one data item represented as $D = \{d_1, d_2, \ldots, d_N\}$. The data center in the model is represented as $Dc = \{dc_1, dc_2, \ldots, dc_M\}$. The connection between the user and friend is represented as $C = \{c_1, c_2, \ldots c_n$. Each of its element is assigned as row in the relationship matrix X. The relationship between the users with each other are described by the space matrix called X with row as users and columns j as friends, and it is represented as

$$X_{ij} = \begin{cases} 1 & Data\ of\ the\ user\ i\ is\ stored\ in\ datacentre\ j \\ 0 & otherwise \end{cases} \quad (1)$$

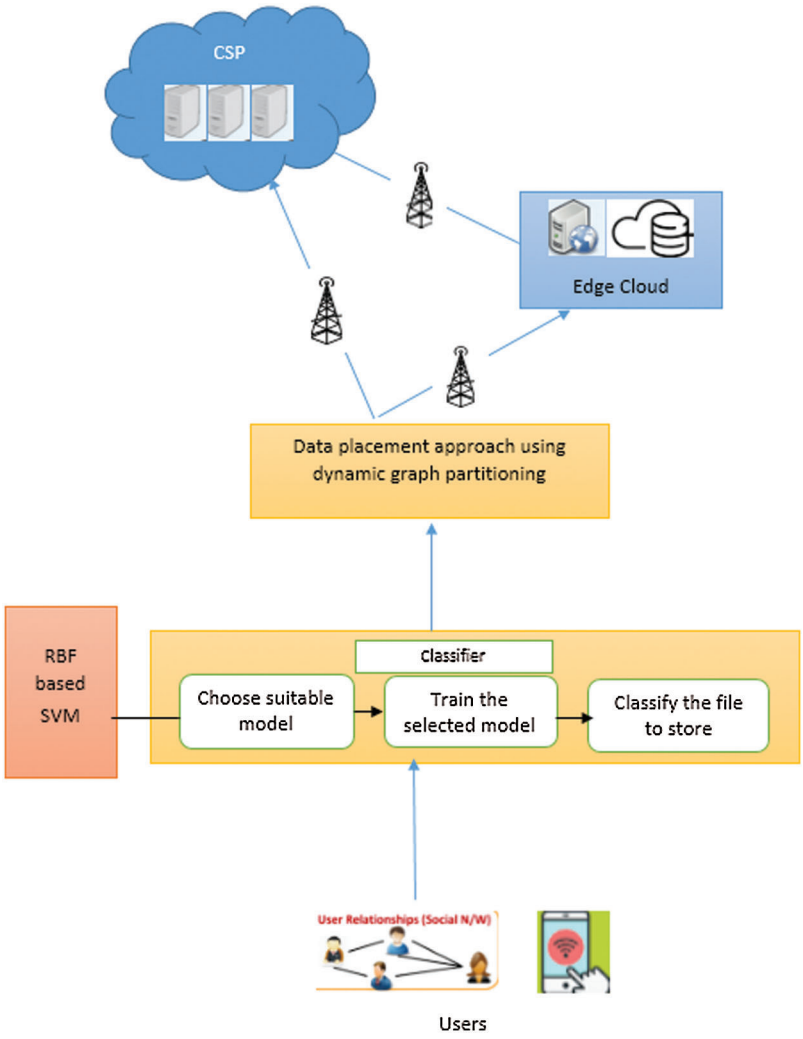

Figure 1: Overall architecture of the proposed data placement approach

These parameters called set of users, data centers, connections, and space matrix are updated constantly if any change occurs like friendship created or removed, a new user joined, existing user left, or a data center in the edge is added or removed. The SIoT is represented using the graph called G = {V, E}, where V is the set of vertices of physical devices connected using the link E, which is the social relationship between devices mentioned in matrix X. The weight of this connection link is represented as wij, which is the binary variable represented as

$$w_{ij} = \begin{cases} 1 & if\ device\ i\ have\ social\ relationship\ with\ j\ based\ on\ SIoT \\ 0 & otherwise \end{cases} \quad (2)$$

The edge network is denoted using the graph GE = {VE, EE}, where VE is the finite set of edge servers and EE is the set of links between the edge servers. The latency between these edge servers is L_{ij}, where ij are

weight between the two vertices $\{i,j \in VE\}$. Similarly, the latency among node i ∈ V and node k ∈ VE is represented as L_{ik}.

3.2 Cost Model

The proposed work aims to allocate the SIoT data to the edge servers using the cost function to be minimized. Cost is referred to as the cost for storing the data into the data centers as different edge servers. As mentioned above, N is the user numbers and R_i is the number of replicas of the user i. The total cost is calculated as the total monetary cost for storing the main copy and replicas of all users located in different edge servers. The cost function is calculated as

$$Storage\ Cost\ (SC)_i = DCstoragecost \times Datasize_i \times R_i \tag{3}$$

$$R_i = \sum_{j=1}^{M} X_{ij} \tag{4}$$

$$Cost(\$) = \sum_{i=1}^{N} Storage\ Cost_i \tag{5}$$

where

DC storage cost-price for storing the data in the data center per month

$Datasize_i$-data size of the user i

Storage Cost-cost for storing the user main data and replicas stored for 1 month in different edge servers.

3.3 Problem Model

To define the problem of storing the SIoT data to the data centers on the given edge servers, decision variable is used. It is represented using the binary variable $d_{ik} \in \{0,1\}$, which means the data of the SIoT device i is allotted to the edge server k. x is the binary variable denoted as

$$d_{ik} = \begin{cases} 1 & if\ SIoT\ device\ data\ is\ assigned\ to\ edge\ server\ k \\ 0 & otherwise \end{cases} \tag{6}$$

The mathematical representation of the problem [3] is

$$\min_{x} \sum_{i \in V} \sum_{k \in VE} d_{ik} L_{ik} + \sum_{i \in V} \sum_{k \in VE} \sum_{j \in V} \sum_{l \in VE} d_{ik} d_{il} w_{ij} L_{kl} \tag{7}$$

The constraints are listed as follows:

1. $\sum_{k \in VE} d_{ik} = 1\ \forall i \in \forall VE$-SIoT of device k is assigned to one edge server only
2. $\sum_{i \in V} d_{ik} = \gamma\ \forall i \in \forall V$–maximum number of SIoT device deployed to the edge server is limited to γ. The two friends of SIoT called i, j deployed to edge server k and l are represented in Fig. 2.
3. $L_{ik} \leq L_{max}\ \forall i \in VE,\ \forall k \in V$–limitation between the edge server and SIoT device is limited to the maximum value called L_{max}
4. $d_{ik} d_{il} \in \{0,1\} \forall i,j \in VE,\ \forall k,l \in V$–data placement problem.

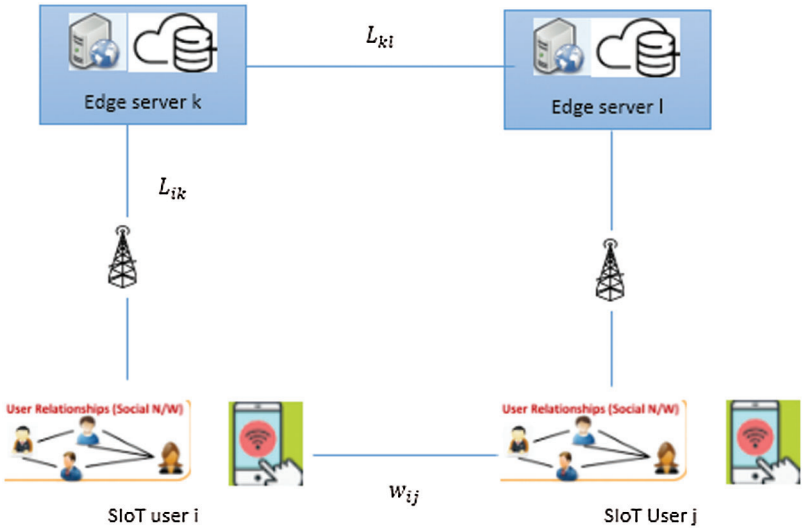

Figure 2: SIoT device data deployed to edge server of two friends

3.4 Network Model using RBF-SVM

This section describes the ML algorithm called RBF-SVM classifier to choose the storage location of the SIoT data in the edge server to avoid replication. The mathematical model of Section 3.3 is trained using the RBF-SVM classifier to choose the edge server for storing the SIoT data. The network model of this proposed work uses SVM with the kernel function as RBF to find the location where the data of the user and friend need to be stored. This way will reduce the data replication. SVMs are ML methods used to divide two classes of data [20,21] using the optimal hyperplane. It can be used for binary and multiclass classification, and it is also used to solve linear and nonlinear regression and classification problems. In SVM, the training data for each input x are defined as

$$D = \{(x_i, y_i) \in R^n,\ i = 1, 2, 3 \ldots N\} \tag{8}$$

The SVM hyperplane H is defined as

$$H : (\omega.x) + b \tag{9}$$

where ω is the weighting vector (boundary of the different class) and b is the bias/threshold. The linear separable classification of SVM satisfies the following constraints:

$$\begin{cases} \omega \times x_i + b \geq 1 & if\ y_i = 1 \\ \omega \times x_i + b \leq -1 & if\ y_i = -1 \end{cases} \tag{10}$$

In the proposed work, the two classes are edge and cloud servers. The input user and friend data are stored in either edge or cloud server to avoid replication based on the SVM classification. The optimal hyperplane H_o will increase the margin M, which is calculated as the minimum distance between the user and the edge and cloud. The sum of the distance between the two classes with respect to H_o will be maximized to boost the margin M. The margin M is represented as

$$M = \min_{x_i | y_i = 1} \frac{\omega.x}{||\omega||} - \max_{x_i | y_i = 1} \frac{\omega.x + b}{||\omega||} = \frac{1}{||\omega||} - \frac{-1}{||\omega||} = \frac{2}{||\omega||} \tag{11}$$

The optimal hyperplane will be obtained by maximizing the M. This procedure is equal to minimizing the following equation:

$$\min_{\omega} \frac{||\omega||^2}{2} \tag{12}$$

This equation will be solved as the quadradic optimization using the Lagrangian function as

$$L(\omega, b, \alpha) = \frac{1}{2}||\omega||^2 - \sum_{i=1}^{N} \alpha_i [y_i(\omega.x_i + b) - 1] \tag{13}$$

where α_i is the Langrangian multiplier factor, which is $\alpha_i = (\alpha_1, \alpha_2, \ldots \alpha_N) > 0$. The weight vector ω is derived from the abovementioned equation as

$$\omega = \sum_{i=1}^{N} \alpha_i.y_i.x_i \tag{14}$$

$\sum_{i=1}^{N} \alpha_i.y_i = 0$. The threshold b is evaluated as

$$b = y_j - \sum_{i,j=1}^{N} \alpha_i.y_i.(x_i.x_j) \tag{15}$$

The classification is defined using the signum function [5] as

$$class(x) = sgn(\omega.x_i + b) = sgn\left[\sum_{i=1}^{N} \alpha_i.y_i(x_i x) + b\right] \tag{16}$$

For nonlinear classification, SVM introduces the slack variable called φ and kernel function, which changes the training data slightly. The training input vectors must satisfy the following constraint:

$$y_i(\omega.x_i + b) \geq 1 - \varphi_i \quad i = 1, 2, \ldots N \tag{17}$$

The kernel function is used to map the data into the transformed space where the hyperplane is used for linear separation. The original data are transformed into feature space using nonlinear function. The final class of nonlinear classification is defined as

$$class(x) = sgn\left[\sum_{x_i \in VE} \alpha_i.y_i.K(x_i, x) + b\right] \tag{18}$$

where $K(x_i, x)$ is the kernel function. In the proposed work, radial basis function is used as a kernel function, which is represented as

$$K(u, v) = \exp(-\frac{||u - v||^2}{2\sigma^2}) \tag{19}$$

The linear and nonlinear classification of the SVM are illustrated in Figs. 3 and 4 respectively. Accordingly, the linear and nonlinear classification of the input training data are classified. In this work, the user data are stored on the basis of the weight vector calculation between the user and edge server or user and cloud server. Edge server is closer to the SIoT user data. Storing the data on edge server will increase the data availability and reduce the latency. The data replication is avoided by storing the data in either cloud server or edge server for quick access.

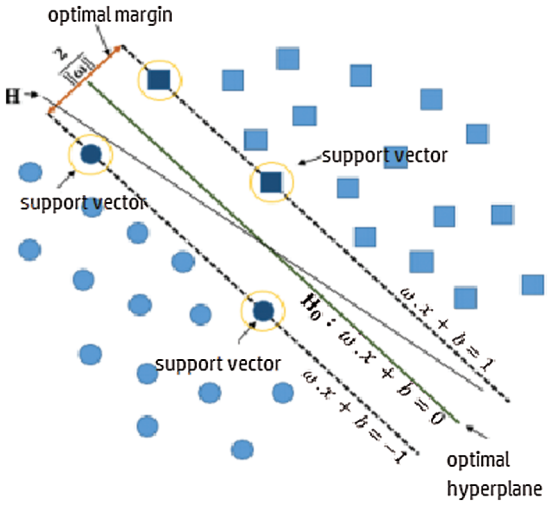

Figure 3: SVM separable hyperplane of linear classification

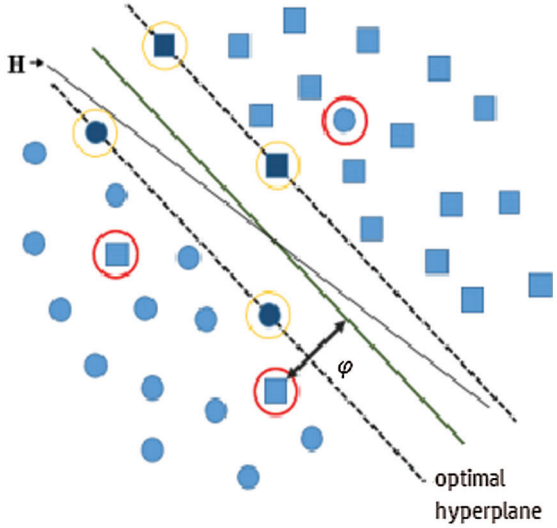

Figure 4: SVM separable hyperplane of nonlinear classification

3.5 Data Placement using Dynamic GP

Once the classifier produces the file classification, the files are placed into the storage medium using the data placement approaches. As discussed in Section 3.4, data placement and geo replication of online SN services should be considered. The data need to be placed to the selected data centers to minimize the latency for most of the users. Given that the SIoT users are increasing rapidly, rather than doing the data replication and placement on individual users, this study places the data using the partitioned graph based on the learning from Section 3.4, which is described in following steps. (i) Every user partition has the primary data center called edge server that is the nearest data center to the main user. The main user is the SIoT user with a greater number of friends. Data corresponding to all SIoT users of that partition are stored in the primary data center called edge server. (ii) All the data centers are sorted according to the distance from the main users, and a greater number of replications are in the nearest data centers until the latency requirement is satisfied.

With these points, the data placement is conducted for the users who are located in a greater number of partitions using the dynamic GP approach. In the existing GP-based data placement, novel GP using vertex cut method is used to partition the interconnected users. Then, the data placement and replication are performed by placing the data to the nearest data center with a greater number of friends. In this approach, the user data can be replicated because of vertex cut algorithm assigned to the partitions. The data storage used in the existing work is cloud. This existing work has been enhanced with edge computing to reduce the latency and complexity of the data placement. SNs have dynamic nature that the user can join or leave, travel or move to various location around the world, and conduct friendship creation or deletion. Data centers can also be added or removed. Thus, an optimized data placement with dynamic and learning by itself using ML techniques and storage medium like edge node are needed to be developed with minimum cost. These challenges are addressed in this study compared with existing approaches. The process of the proposed data placement scheme is shown in Fig. 5. The choice of storage medium learned with ML algorithm before data placement will improve the system performance while fulfilling latency of the SIoT users. The data placement of the user data will reduce the data replication and latency due to the proposed ML and edge computing approaches.

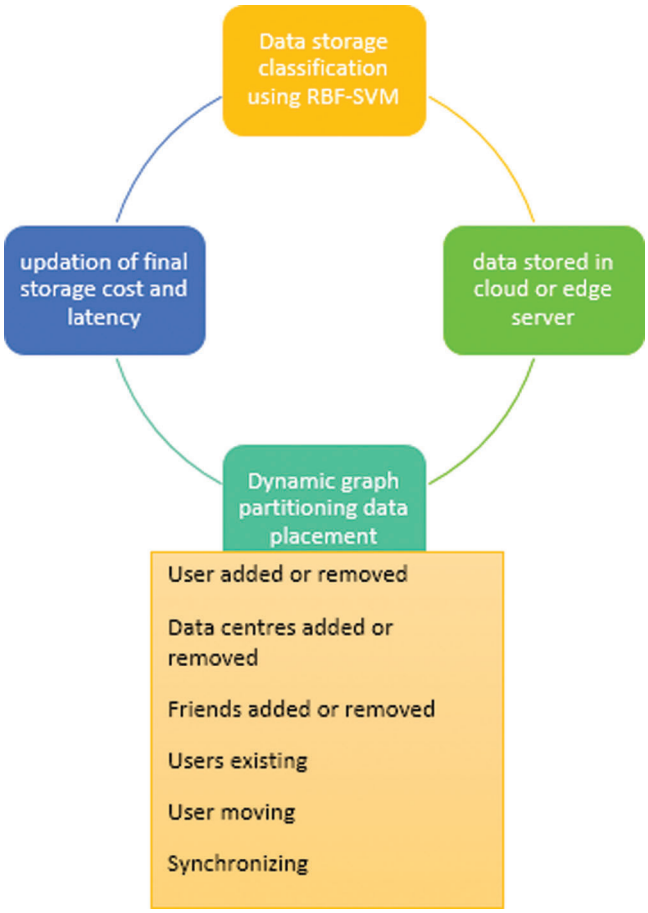

Figure 5: Process of dynamic graph partitioning

The latency of the users and data centers are calculated here on the basis of the actual latency of the man users located in the cloud and edge DC. By placing the various placements of data with different data centers, each SIoT user can access the data to nearest data center that may have the replica of the data. The final

latency of each SIoT user is the latency between the user and their data with the latency of 90 percentile between all their respective friends in the partition, and it is calculated as

$$latency_i = p^{th}(latency_k) \tag{20}$$

where k = 1..f_i and f is the number of friends for the SIoT user i. The acceptable delay between all the users and friends is calculated with the pth percentile of the latencies as

$$p^{th}\left(latency_p(t)\right) \leq delay \tag{21}$$

where p = 1, 2, 3…P. $P = \sum_{i=1}^{n} \sum_{j=1}^{No.of\ friends} ReqNum_{ij}(t)$

The request p latency in time t is the time taken by the friend to send request to access the data from the nearest DC. The total latency of the user may change depending on the access of that user's data frequently by the friends. The algorithm for the proposed dynamic GP data placement is stated as

Algorithm 1: Dynamic GP data placement

Inputs: SN of SIoT graph (SIoT users and connections), interval (time period), the existing solution set {s1, s2, .. Sm} of each user, c number of connections, N users, M data centers, and k number of partitions.

Output: Optimized solution set of each user; the data placement DP = {d1, d2, .. dn}, cost, and latency of the file placement

Step 1: for i = 1 to c //finding the friends list for every user

Step 2: assign every connection user to the friendslist

Step 3: increase the friend num

Step 4: end for

Step 5: do // check for the changes until time period over

Step 6: start time = current time

Step 7: while (CT-ST) <= interval)

Step 8: add/remove the users/dc with latency update

Step 9: CT = the current time

Step 10: end while

Step 11: the replicas avoided using the RBF-SVM method in Section 3.3 and the storage space is selected using the classifiers.

Step 12: for all connections c = 1 of the user i and j

Step 13: replica = data center with the lowest latency of the user j has the replica of the user i

Step 14: end for

Step 15: update and return final cost using Eq. (3)

Step 16: update the final latency using Eq. (20)

Step 17: return the number of replicas of the user

Step 18: end do

In the proposed work, the GP with dynamic nature to add or remove users, data centers, and friends in SN is discussed. Our novel dynamic GP data placement with ML will reduce the data replication with minimum cost to store, transfer, and synchronize data over time. Compared with the existing approaches, our proposed data placement scheme reduces 30% of cost while meeting the latency requirement of the SIoT users due to the edge server storage and ML-based learning.

4 Simulation Results and Discussion

In this section, we evaluate our proposed dynamic GP-based data placement learned with RBF-SVM on the Facebook dataset [22] with 63,731 users and 1545686 connections. The initial number of users is 54005, and the initial number of friends for each user is 28% of total friends. First, we evaluate the classification accuracy effect. Then, the efficiency of the proposed data placement algorithm is compared with those of the existing replica and data placement algorithms with different simulated data centers. The existing data placement algorithms for evaluation are as follows. (i) In GA, only one copy of the data is stored in the nearby data center. This algorithm is used to find the nearest data replication number and nearest optimal placement. The crossover rate is fixed as 0.8. (ii) GP-based data placement and replication is used to find the near optimal data placement of replicas to reduce the monetary cost while satisfying the latency requirement. (iii) Optimal placement of social ET in edge IoT networks is used to formulate the data placement as a mixed integer programming model to limit the computing resources at the edge cloud and relationship among SIoT devices. The proposed work simulation uses IBM ILOG CPLEX optimization studio 12.10.0 software suite. The simulation parameters include 150 SIoT devices, 10 edge servers, capacity of the edge server, maximum latency between the SIoT devices and edge servers, and latency coefficient distance of 3.31 ms/km [23].

4.1 Effect of Classification Accuracy

For the evaluation of classification algorithm, the dataset is divided into training and test data with a ratio of 7:3. The proposed classification using RBF-SVM has been compared with the Bayes, SVM, NN16, and NN3 [9] with the storage capacity of 1:1:1. The accuracy evaluation shows that the proposed RBF-SVM classification-based data placement obtains 92%, which is higher than those of other algorithms. The average accuracy of the classifiers is shown in Fig. 6.

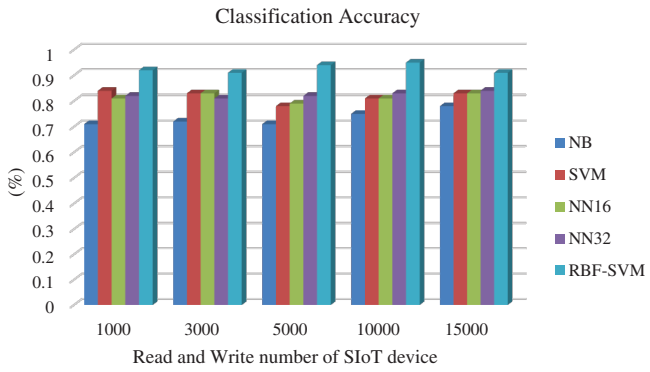

Figure 6: Comparison of classifiers accuracy

The comparative illustration of the graph shows that the highest accuracy is obtained by the RBF-SVM classifier. For naïve Bayes, the accuracy decreases while the sampled data increase. NN16 and NN32 have 16 and 32 neurons, respectively, and they obtain decreased level of accuracy compared with other approaches.

4.2 Efficiency of the Proposed Data Placement

Effectiveness of the proposed methodology is considered in terms of cost of the data placement and latency with respect to the time slot. Cost and latency of the user and friends added or removed and data center added or removed are evaluated. The evaluated results of the latency and cost with respect to the user/friend added/removed and data center added/removed are shown in Tabs. 1 and 2 and illustrated in Figs. 7–10.

Table 1: Latency and cost of new user/friend added or removed

T time slot	User/friends added		User/friend removed	
	Latency (ms)	Cost ($)	Latency (ms)	Cost ($)
100	230	1300	220	1250
200	220	1200	235	1200
300	245	1000	240	1100

Table 2: Latency and cost of new data centre added/removed

Data centre	DC added		DC removed	
	Latency (ms)	Cost ($)	Latency (ms)	Cost ($)
10	130	1150	220	1200
8	140	1130	130	1220
6	230	1050	120	1350
4	250	850	110	1400

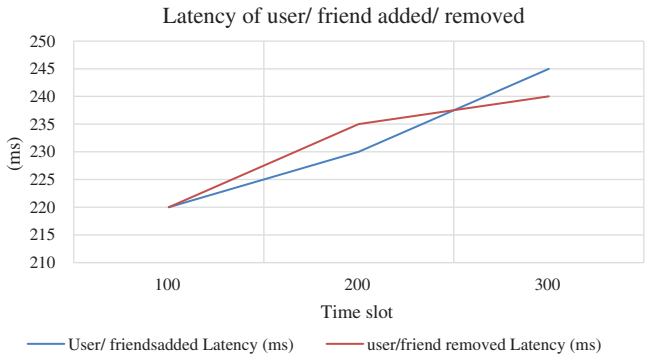

Figure 7: Latency of new user added/removed

The evaluated results of the proposed dynamic GP with ML-based technique obtain minimum latency and cost for various evaluation time slots and data centers. A comparative study with previous approaches is conducted to verify the efficiency of the proposed work.

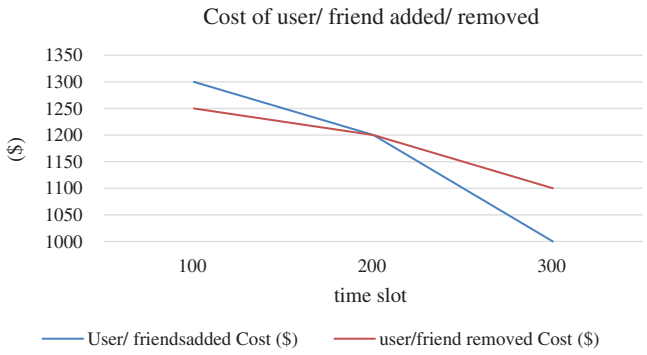

Figure 8: Cost of new user/friend removed

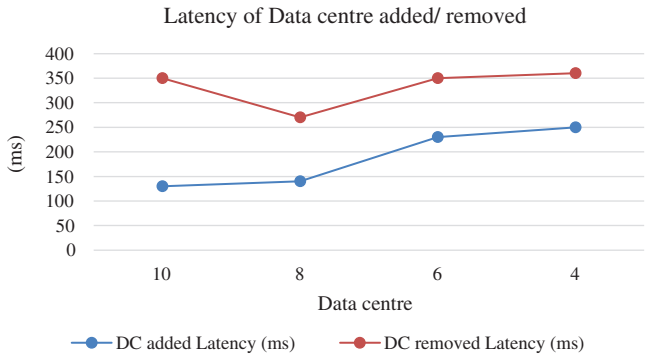

Figure 9: Latency of data centre added/removed

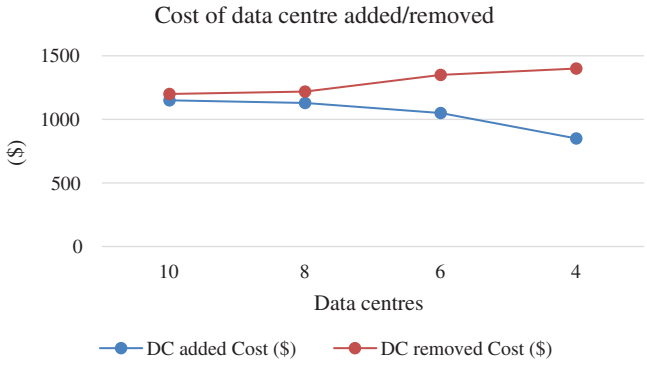

Figure 10: Cost of data centre added/removed

4.3 Comparative Analysis of Data Placement Approaches

The proposed approach is compared with the existing approaches like GA-based data placement, GP, and social DT edge-based data placement. The evaluated results in terms of latency and cost. From the evaluated results, our proposed approach obtains minimum latency of 130 ms compared with other existing approaches. Specifically, GA obtains 230 ms, GP obtains 150 ms, and DT obtains 180 ms. The minimum cost obtained by our proposed approach is 850$ compared with other existing algorithms. The next best method on data replication and placement is GP-based data placement algorithm.

Therefore, our proposed approach, which is improved with the GP-based data placement with the enhancement like ML-based learning, data center in edge, and GP data placement with dynamic nature,

performs better in user latency of **130 ms** and minimum cost. The proposed dynamic GP with RBF-SVM-based learning on edge is the best approach for data replication and placement for SIoT.

5 Conclusions

Cost effective SN data placement and replication strategy using novel dynamic partitioning with RBF-SVM is proposed in this research. The dynamic natures of the activity and mobility of the user in the SN are handled with our proposed work. In our method, the users can leave or join the network, friends can join or be removed, the dataset can be created, modified, or removed, and a new data center like cloud or edge can be added or removed if needed. Our novel data placement method can adapt to this dynamic nature of the SN. The ML-based assistance on the proposed work will reduce data replication by choosing the storage location in the data center for SIoT users. The data stored in the edge can be quickly accessed by the user given that it is located near the SIoT devices. Once the classifier chooses the data storage medium as cloud or edge with reduced replication, the data placement algorithm called dynamic GP is used to reduce the user latency. The storage cost of storing the user's data also reduces by distributing the user's data into cloud and edge. The simulation results show that the proposed dynamic data placement with ML-based learning on edge will reduce the latency, cost, and replication effectively.

The proposed work is compared with the existing data placement approaches like GA-based data placement, GP-based data placement, and social twin edge on IoT. The experimental results are promising with reduced latency of 130 ms and minimum cost for the proposed work compared with those of other contemporary approaches. In terms of classification accuracy, our ML-based learning method called RBF-SVM obtains higher level of accuracy than other ML-based algorithms. This performance will enable prediction of the workload of the data center timeslot based on the learning of existing time slots. The evaluation of Facebook datasets shows the efficiency of our proposed work. Thus, the proposed strategy of data placement on SNs will be proven in terms of efficiency, effectiveness, reduced replication, low latency, and minimum storage cost.

In the future, weighting mechanism will be added in the edge to satisfy frequent request of accessing the user data by friends. The reason is that some friends' devices may require to interact and access the user data more frequently than others. We plan also to extend our work to the dynamic virtual machine management system based on cloud and edge conditions.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] Josh Constine, "Facebook now has 2 billion monthly user and responsibility," 2017. [Online]. Available: https://techcrunch.com/2017/06/27/facebook-2-.

[2] Amazon S3, "Object storage built to store," 2021. [Online]. Available: http://aws.amazon.com/s3.

[3] Google cloud storage, 2021. [Online]. Available: http://cloud.google.com/storage.

[4] Microsoft, Windows Azure, 2015. [Online]. Available: http://www.microsoft.com/windowsazure.

[5] X. Liu, J. Chen and Y. Yang, "A probabilistic strategy for setting temporal constraints in scientific workflows," in *Proc. BPM, Seville*, Spain, pp. 180–195, 2008.

[6] K. Hourieh, Y. Dong, B. B. Zhou, J. Grundy and Y. Yun, "Cost effective dynamic data placement for efficient access of social networks," *Parallel and Distributed Computing*, vol. 141, pp. 82–98, 2020.

[7] H. Khalajzadeh, D. Yuan, J. Grundy and Y. Yang, "Cost-effective social network data placement and replication using graph-partitioning," in *Proc. ICCC*, Honolulu, HI, pp. 64–71, 2017.

[8] C. Olga, N. Chukhno, G. Araniti, C. Campolo, A. Iera *et al.,* "Optimal placement of social digital twins in edge IoT networks," *Sensors*, vol. 20, no. 21, pp. 61–81, 2020.

[9] J. Ren, X. Chen, Y. Tan, D. Liu, M. Duan *et al.,* "Archivist: A machine learning assisted data placement mechanism for hybrid storage systems," in *Proc. ICCD*, Abu Dhabi, United Arab Emirates, pp. 676–679, 2019.

[10] H. Khalajzadeh, D. Yuan, J. Grundy and Y. Yang, "Improving cloud-based online social network data placement and replication," in *Proc. CLOUD*, San Francisco, CA, USA, pp. 678–685, 2016.

[11] S. Han, B. Kim, J. Han, K. Kim and J. Song, "Adaptive data placement for improving performance of online social network services in a multi-cloud environment," *Hindawi Scientific Programming*, vol. 2017, pp. 1–17, 2017.

[12] Q. Xu, Z. Q. Xu and T. Wang, "A data-placement strategy based on genetic algorithm in cloud computing," in *Proc. WISA*, Yangzhou, China, pp. 145–157, 2013.

[13] X. Luo, G. Xin and X. Gui, "Data placement algorithm for improving I/O load balance without using popularity information," *Hindawi Mathematical Problems in Engineering*, vol. 2019, pp. 1–11, 2019.

[14] T. P. Shabeera, S. D. Madhu Kumar, S. M. Salam and K. Muralikrishnan, "Optimizing VM allocation and data placement for data-intensive applications in cloud using ACO metaheuristic algorithm," *Engineering Science and Technology, an International Journal*, vol. 20, no. 2, pp. 616–628, 2017.

[15] Y. Wu, C. Wu, B. Li, L. Zhang, Z. Li *et al.,* "Scaling social media applications into geo-distributed clouds," *IEEE Transactions on Networking*, vol. 23, pp. 689–702, 2012.

[16] Q. Xia, W. Liang and Z. Xu, "The operational cost minimization in distributed clouds via community-aware user data placements of social networks," *Computer Networks*, vol. 112, no. 4, pp. 263–278, 2017.

[17] B. Yu and J. Pan, "Location-aware associated data placement for geo-distributed data-intensive applications," in *Proc. INFOCOM*, Kowloon, Hong Kong, pp. 603–611, 2015.

[18] J. Zhoua, J. Fan, J. Jia, B. Cheng and Z. Liu, "Optimizing cost for geo-distributed storage systems in online social networks," *Computer Science*, vol. 26, no. 12, pp. 363–374, 2018.

[19] M. Saber, A. E. Rharras, R. Saadane, H. K. Aroussi and M. Wahbi, "Artificial neural networks, support vector machine and energy detection for spectrum sensing based on real signals," *International Journal of Communication Networks and Information Security*, vol. 11, no. 1, pp. 52–60, 2019.

[20] C. J. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[21] C. Cortes and V. Vapnik, "Support vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[22] B. Viswanath, A. Mislove, M. Cha and K. P. Gummadi, "On the evolution of user interaction in facebook," in *Proc. WOSN*, Barcelona, Spain, pp. 37–42, 2009.

[23] R. Landa, J. T. Araújo, R. G. Clegg, E. Mykoniati, D. Griffin *et al.,* "The large-scale geography of internet round trip times," in *Proc. IFIP Networking Conf.*, Brooklyn, NY, USA, pp. 1–9, 2013.