International Journal of Electrical and Computer Engineering (IJECE)
Vol. 13, No. 1, February 2023, pp. 638~650
ISSN: 2088-8708, DOI: 10.11591/ijece.v13i1.pp638-650

Location-aware deep learning-based framework for optimizing cloud consumer quality of service-based service composition

Alshaimaa M. Mohammed[1], Samar Shaaban Abdelfattah Haytamy[2], Fatma A. Omara[3]
[1]Computer Science and Mathematics Department, Faculty of science, Suez Canal University, Ismailia, Egypt
[2]Department of Computer Science, Faculty of Computer and Information, Fayoum University, Faiyum, Egypt
[3]Computer Science Department, Faculty of Computer and AI, Cairo University, Cairo, Egypt

Article Info

Article history:

Received Jul 8, 2021
Revised Jul 15, 2022
Accepted Aug 2, 2022

Keywords:

Cloud service composition
dimensional reduction
Deep learning
Location-aware
Quality of service

ABSTRACT

The expanding propensity of organization users to utilize cloud services urges to deliver services in a service pool with a variety of functional and non-functional attributes from online service providers. brokers of cloud services must intense rivalry competing with one another to provide quality of service (QoS) enhancements. Such rivalry prompts a troublesome and muddled providing composite services on the cloud using a simple service selection and composition approach. Therefore, cloud composition is considered a non-deterministic polynomial (NP-hard) and economically motivated problem. Hence, developing a reliable economic model for composition is of tremendous interest and to have importance for the cloud consumer. This paper provides "A location-aware deep learning framework for improving the QoS-based service composition for cloud consumers". The proposed framework is firstly reducing the dimensions of data. Secondly, it applies a combination of the deep learning long short-term memory network and particle swarm optimization algorithm additionally to considering the location parameter to correctly forecast the QoS provisioned values. Finally, it composes the ideal services need to reduce the customer cost function. The suggested framework's performance has been demonstrated using a real dataset, proving that it superior the current models in terms of prediction and composition accuracy.

Corresponding Author:

Alshaimaa M. Mohammed
Computer Science and Mathematics Department, Faculty of science, Suez Canal University
Ismailia Governorate, Egypt
Email: shaimaa_mostafa@science.suez.edu.eg

1. INTRODUCTION

Cloud computing is a network-based system in which information technology and computing resources such as hardware, operating systems, storage, networks, databases, and even whole applications are conveyed to users as on-demand facilities just through the internet. Cloud computing displays several benefits, such as dynamic environment, on-demand services, scalability, yet additionally has several challenges that ought to be considered by experts such as privacy and security, virtualization, scheduling, resource discovery, reducing the attacks, service discovery, data replication, service recommendation, service composition, and selection. Numerous businesses, including Amazon, IBM, Microsoft, and Google offer several services for cloud computing [1], [2]. Subsequently, cloud computing offers a pool of on-demand computing resources to users to find their needed service(s) that fulfill their request. As per National Institute of Standards and Technology (NIST) definition, cloud computing has five essential characteristics: on-demand self-service, broad network access, resource pooling, rapid elasticity, and admeasured services. It

is deployed according to three deployment models: public, private, and hybrid models. Furthermore, software as a service (SaaS), platform as a service (PaaS), and infrastructure as a service (IaaS) are the three service categories it provides [3], [4]. These services draw in more taking into account scaling up and down, dependable resources for high-performance hardware and software, as well as eliminating security risks and maintenance costs. Accordingly, numerous organizational leaders and Information technology (IT) organizations' stakeholders have urged moving to cloud computing.

Over the last few years, with IT improvement, conventional services cannot accomplish the demand of users. Henceforth, blending IT and services is commendable for present-day intelligent systems. Users can carry out their requirements identified with education, transportation, commerce, social networking, and numerous other utility cloud services [3], [5]. The foremost critical objective for users is to effectively utilize their requested services for lower operational cost, increasing scalability, improving performance, and utilizing their resources in efficient ways. Subsequently, cloud computing has developed to meet users' computing prerequisites by advertising new features designed to enhance the sharing of computing resources and applications on demand and based on pay-as-you-go through the internet [2], [6].

As a result of the presence of complex and diverse services, a single simple service cannot fulfill the existing functional requirements for many real-world cases [7]. To complete a complex service, it is essential to have a combination of simple services that work with each other. In this manner, there is a strong need to embed a cloud service composition system in cloud computing which has pulled in numerous researchers' inspection [8]. Each cloud service has its own quality of service (QoS) parameters like response time, throughput and cost which depicts the non-functional attributes of the service. As the quantity of services is expanded in the cloud, various services can meet the users' request functionality. Yet, these services are varying in their QoS values. In this way, the service selection decision-making process brings a major challenge in the service composition process. Thus, choosing the proper and optimal simple services that would be consolidated to provide composite complex services is one of the most important challenges in cloud service composition. The cloud service composition problem in cloud computing can be characterized as figuring out what atomic simple services ought to be chosen with the acquired complex composite service fulfills both the functional and QoS requirements based on the consumer requirements. Considering different and plentiful effective parameters and countless simple services provided by many service providers in the cloud pool, the problem with cloud service composition is a combinatorial problem of optimization and also has been viewed as a non-deterministic polynomial (NP-hard) problem [8].

Subsequently, cloud service composition is a method for creating with fundamental meta-services features for a complicated composite service that satisfies the functional needs of the consumers. It can be relatively straightforward if each function's meta-service is distinct. There is no standalone cloud service, which is the fundamental fact of the cloud environment [9]. Likewise cloud computing innovation's pay-as-you-go feature enables service providers to offer their services in a variety of configurations in accordance with service level agreements (SLA) [10]. In this way, the increase in the number of meta-services diverse cloud providers poses a problem. The functional characteristics of these services are similar, but their non-functional actions differ (i.e. QoS) [11]. Accordingly, when different functionally overlapping services are supplied at different QoS levels, cloud consumers will find it difficult to choose the right services. Consequently, there is a pressing demand for cloud service composition models.

There are more research concerns about solving the problem of selecting the most appropriate services for the QoS-based cloud service composition [12], [13]. Notwithstanding, a large portion of this research focuses on static QoS values observed during composition time. In practice, QoS varies over time, such future variations should be thought about while designing composite services.

Contrasted with conventional composition of services, composition process is typically focused on long-term goals and economics. Conventional techniques for composition that are focused on quality typically take the attributes into account when creating the composition [14]. Which composite service, for instance, performs the best right now? This is fundamentally different in cloud environments since the cloud service composition in these environments is meant to last for a long time. What hybrid cloud service, for instance, will perform best in the coming years even though it may not be the greatest one right now? Multiple study fields, including multimedia, statistics, and others, frequently use time-series databases. To examine economic models successfully and efficiently in cloud computing, numerous methodologies have been developed.

There is now a solution to the composition of the QoS-aware services issue using a number of strategies [13], [15], [16]. These methods ignore the volatility in the cloud environment and instead focus on service quality at the time of composition, which does not consent to the commonsense circumstances. Nowadays, enormous enterprises and with cloud service providers, institutions are typically more eager to establish long-term commercial relationships. Be that as it may, the consumers' actions (i.e., QoS specifications) and the effectiveness of the chosen services (i.e., QoS characteristics) are essentially unstable. Accordingly, the composite service with the top QoS performance at the moment is not an awesome

timeframe. So compared to regular online services, The cloud service composition is thought of as an economic strategy based on the long term. Typically, the economic model of a cloud consumer seeks to achieve things like increased throughput, decreased reaction times, and low capital costs. We presume that QoS features are expressed as a time sequence in this research. The correlation between the QoS characteristic and one or more other QoS attributes is possible. In study [17], when the correlation operator between QoS attributes is put to the test, it is found that throughput and response time attributes have a very significant negative correlation. By taking into account all of the connected time series, the correlation factor between QoS characteristics can be used to forecast the time series of certain QoS qualities. The majority of QoS-aware prediction models that are currently available do not take the correlations between QoS variables into account. Then again, because of economic considerations, enormous companies and organizations generally are more able to construct a long-term business relationship with cloud service providers. On the other hand, the end user's requests (i.e., QoS requirements) and the selected services' performance (i.e., QoS attributes) are both fluctuant in practice. Accordingly, the composite service with the best current QoS performance is not necessarily the best after a while or is not awesome sometimes. Thus, cloud service composition systems provide a value-added service.

To tackle the problem of composing the best cloud services that cut down on the function of consumer costs, this paper's work provides a location-aware framework based on deep learning for improving cloud consumer QoS-based service composition location-aware deep learning-based service composition (L-DLSC). This framework is an enhanced framework based on our earlier work [1]. It addresses the problem as a multivariate time series analysis. The proposed L-DLSC framework comprises three phases. As per the primary phase, the filtration phase is acquainted and applied to select the best-contributed attributes to reduce the total execution time of learning and composition. In the subsequent phase, the long short-term memory (LSTM) network is used to accurately predict future behavior. In the third phase, the composition of the service is handled as a multi-objective problem using the particle swarm optimization algorithm (PSO) with considering the location between services and the consumer.

Therefore, this paper's contribution can be summed up as: i) by combining the deep learning network LSTM and the PSO algorithm, a framework to compose services using deep learning-based service composition (DLSC) has been improved; ii) to speed up execution, the number of existing characteristics has been decreased utilizing an autoencoder (AE) network. When choosing the best composite service to reduce latency, the location parameter is taken into account.

The remaining portions of the essay are structured; the fundamentals of cloud service composition are introduced in section 2. The related work is presented in section 3. Section 4 introduces the proposed location-aware deep learning-based cloud service composition architecture. Section 5 presents the experimental findings. Section 6 presents conclusions and ideas for additional research.

2. METHOD

Utilizing a combination of cloud services is intended to provide value-added services can complete demanding tasks for customers. The consumer's QoS limitations and preferences must be met as one of the service composition's requirements [1]. The choice of the best individual services that must be combined to provide value-added composed services is one of the main issues with the service composition challenge [18]. Mathematically, the multi-dimensional, multichoice knapsack problem (MMKP), an NP-hard problem, can be used to represent the composition of a QoS aware service. As a result, the composite service (CS) problem, which was mentioned in our earlier work, may be characterized as being the make-up classifications of the essential services (SC), which are depicted by (1) [1].

$$SC = [C_1, C_2, \dots, C_j, \dots, C_n] \quad (1)$$

With C stands for a particular a solitary service class, n stands for the necessary quantity classifications of individual services, the (2) defines a single service class called C_j.

$$C_j = \{S_1, S_2, \dots, S_f\} \quad (2)$$

Where, the single service class C_j contains *f (f>1)* functionally equivalent services with different QoS values. So, the quality of SC can be determined using (3).

$$QoS(SC) = [QoS(C_1), QoS(C_2), \dots, QoS(C_j), \dots, QoS(C_n)] \quad (3)$$

Finding the best composited services with the best QoS characteristics to satisfy the needs of cloud consumers is a clear objective in the selection process. Cost, throughput, and reaction time are taken into account as QoS features in this study.

For the cloud service composition, which regularly changes in accordance with the complex job specification and its needs, there have been a number of necessary workflows. Sequential pattern, conditional pattern, loop pattern, and parallel pattern are the four fundamental composition patterns (e.g., workflow patterns) that can be used to construct a composite service. The aggregation functions of various QoS parameters vary often for each composition pattern. Therefore, as stated in our earlier work, Table 1 displays the composition pattern and its accompanying aggregated functions. Ought to be taken into consideration to specify and assess the general level of the cloud composite service quality as shown in Figure 1 [1], [14], [15].

Table 1. Aggregation functions, where k is the number of loop times, p is a probabilistic variable, and n is the total number of composing services [19]

	Cost	Response Time	Throughput
Sequential	$\sum_{i=1}^{n} Q_{cost}(i)$	$\sum_{i=1}^{n} Q_{rt}(i)$	$\min_{i=1\ldots n}(Q_{th}(i))$
Parallel	$\sum_{i=1}^{n} Q_{cost}(i)$	$\max_{i=1\ldots n}(Q_{rt}(i))$	$\min_{i=1\ldots n}(Q_{th}(i))$
Conditional	$\sum_{i=1}^{n} (P(i) * Q_{cost}(i))$	$\sum_{i=1}^{n} (P(i) * Q_{rt}(i))$	$\sum_{i=1}^{n} (P(i) * Q_{th}(i))$
Loop	K * Q_{cost} (i)	K * Q_{rt} (i)	Q_{th} (i)

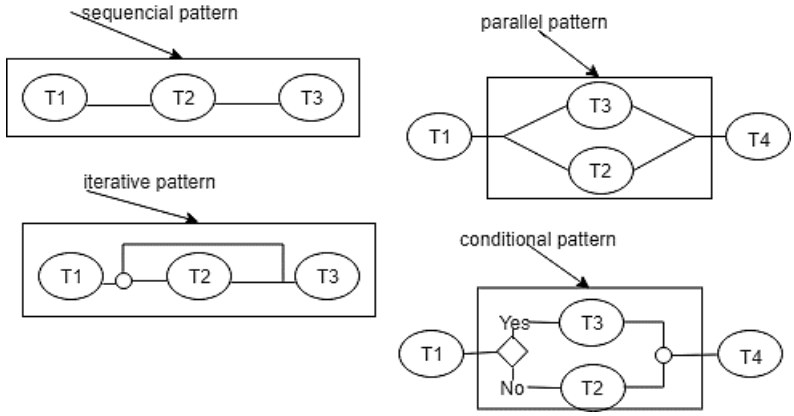

Figure 1. Basic composition pattern [20]

The best way to solve the composite service problem is to maximize or minimize the vector's component values while maximizing the QoS_{total} (SC) vector. The composing system is unaware of how the final composited service will be delivered. will behave over an extended period of time because it has chosen the concrete service providers for the composited service. As a result, it is crucial to foresee the long-term desires of cloud users, which are illustrated by the economic framework. As opposed to that, economic frameworks are described as "theoretical constructs that characterize economic processes by a collection of variables and a set of logical and quantitative linkages between them". The composition of cloud services challenge is seen as an optimization issue with multiple objectives to boost reaction times, cut costs, and increase throughput [12].

The customer's actualized QoS values within a specific time period *t*, together with their corresponding weights, a scoring feature *S* indicated using the common weighted score function [4]:

$$S(t) = W_{th}(t) * q_{th}(t) + W_{cost}(t) * q_{cost}(t) + W_{rt}(t) * q_{rt}(t) \quad (4)$$

As a result, for the purpose of creating a composite plan, the score function, which is established using, the (5) is the total contains all QoS ratings for the full time *T*.

$$S(plan) = \int_0^T s(t)dt \quad (5)$$

Since there are numerous cloud infrastructure and software suppliers offering various combinations. These mixtures result in different plans. Finding the composing services that maximize *S(plan)* is the primary goal of the cloud service composition. The research presented in this essay suggests a location-aware framework based on deep learning to effectively forecast recommended services that optimize the consumer objective function in order to increase composition accuracy.

3. RELATED WORK

Problems with the composition of cloud services is split into two groups: both functional and non-functional characteristics, or QoS. Consumers can differ in their preferences, requirements, and actions. As a result, approaches to composition that are QoS-based, long-term, and location-aware are necessary.

3.1. QoS cloud service composition

Asghari *et al.* [16] have offered an approach for internet of things (IoT) environments for providing composite cloud services. This approach classifies the candidate services into four categories based on their QoS fitness value and privacy-preserving level. To determine the QoS fitness value for each candidate service, an SFLA-GA hybrid algorithm is proposed of genetic algorithm (GA) and shuffled frog leaping algorithm (SFLA) based on nine QoS parameters (i.e., availability, response time, reliability, throughput, latency, compliance, success ability, documentation, and best practice). These nine parameters are normalized and aggregated together to determine the fitness value for each candidate service. On the other hand, the privacy level is determined by considering three factors (i.e., sensitivity, delegation depth, and retention time). Finally, the candidate service is belonging to one of four categories: low fitness/low privacy level, low fitness/high privacy level, high fitness/low privacy level, and high fitness/high privacy level. Finally, the authors have supposed that the last category contains the most proper services for the consumers' request.

In study [13] uses an eagle strategy to develop a QoS-aware cloud service composition. In this approach, exploration and exploitation are balanced in order to address problems like poor convergence rate. The interdependencies and correlations between cloud services as well as QoS parameters are not taken into account by this approach. The authors presumptively used a single repository for all services. As a result, it also ignores the necessity to compose services from a variety of service repositories while limiting connectivity across the various clouds. An algorithm for QoS-aware cloud service composition based on discrete immune fruit fly optimization algorithm (DIFOA) is proposed in [12]. According to this algorithm, the initial population is generated which consider as the initial optimal weights for the resultant services from Pareto applied on initial candidate services. At that point, it finds an optimal composed service based on user preferences attains good fitness value, execution time, and error rate. Since the workflow and concrete services were similar in terms of range, the effect of larger variations in the request-workflow pairs is not considered in this algorithm. Additionally, it does not consider the different selection matching functions.

The composition problem is addressed using integer linear programming (ILP) in [21]. It converts the restrictions and objectives of the issue into linear equations, which are then solved by modifying a special software solver like LPSolve. Surprisingly, as the size of the problem grows, the integer program technique requires exponential time. In order to address the time cost shortage issue, the service composition problem is solved using a GA, in which each chromosomal gene represents a composite service's abstract service and value denotes an applicant services [22]. A strategy based on GA conducts quick seeks to discover almost ideal answers. Nevertheless, because it takes into account static levels of QoS at the moment of construction, not at all well suited for cloud service composition.

The matrix factorization of the user-service matrix is used to leverage the predicted future QoS for planning the composite service in [23]. The grid of user services is then divided into the sum of the user and service matrices using the singular value decomposition method. However, this paradigm has relied on quick QoS values. To maximize resource consumption, the network of deep recurrent neurons is how to predict future QoS for cloud service providers [15]. It is regarded as having just one QoS feature. As a result, it is unable to take into account the relationships among various QoS parameters.

3.2. Long-term service composition

The study [15] proposes a deep learning strategy for long-term QoS-based service composition. It uses deep recurrent long short-term memories (LSTMs) to predict long-term QoS trends. It provides a composition middleware that sends two types of recommendation requests to the long-term QoS compliance checker (LQCC); composition and substitution requests. A composition recommendation request includes the ID of a potential component along with QoS requirements to LQCC. LQCC requests the QoS prediction trend for the component from the QoS predictor. Then, it checks compatibility between the QoS prediction

and QoS requirement intervals and returns a composition recommendation to the middleware. A substitution recommendation request includes IDs of the services to substitute and potential substitutes to LQCC. LQCC obtains the QoS prediction trends for the component to substitute and potential substitute from the QoS predictor. Then, it checks whether the two trends are close enough to each other and returns a substitution recommendation to LQCC.

A framework for DLSC was put out in [1]. This framework works by combining both the PSO technique and the deep learning LSTM network to forecast the QoS values. The long-term based QoS service composition has been formulated in another model as an optimization issue employing meta-heuristic methods such as genetic algorithms, annealing, and tabu search [24]. This model shows that only the most effective services with the typical long-term QoS are chosen. The relationships among the QoS measures, however, have not been taken into account.

In [25], ARIMA or the auto-regressive integrated moving average model to forecast how service requests will behave going forward. It is not appropriate for long-term composition designing because it has not collected stochastic request arrivals. In [26] defines a sustained qualitative approach that cloud service providers must compile customer requests. It saw the IaaS composition as an optimization problem including preferences.

3.3. Location-aware service composition

The presentation of a unique location-aware collaborative filtering approach for QoS-based web service recommendation in [27]. First, it considers the individual QoS traits of both services and consumers to calculate the likeness between them based on using Pearson's correlation coefficient (PCC). Then, it obtains location information of a user including the network and the country according to the users and services and finds the similarity between them. Finally, according to the QoS and location similarities, the required service is recommended.

A location-aware service recommendations with privacy-preservation for the internet of things (IoT) environment is proposed in [28]. First, the QoS matrix is divided for recommendation decision-making into multiple QoS sub-matrices based on service location information. Second, the neighbors of a target user are found through a less-sensitive manner according to the QoS sub-matrices and the locality-sensitive hashing (LSH) technique. Finally, the appropriate candidate services to the target user are recommended with the help of derived neighbors.

4. PROPOSED LOCATION-AWARE DEEP LEARNING-BASED SERVICE COMPOSITION FRAMEWORK

Building precise future economic strategies is key to the cloud business' success. The work in this paper describes the introduction of a location-aware framework for the composition of services using deep learning L-DLSC based on reducing the space dimensions of the multiple time series and combining the PSO and LSTM networks are used to create an economic framework and choose the best service providers for the clients. Three phases make up the suggested model's flowchart as shown in Figure 2. In the sections that follow, the phases of the proposed L-DLSC framework will be thoroughly covered.

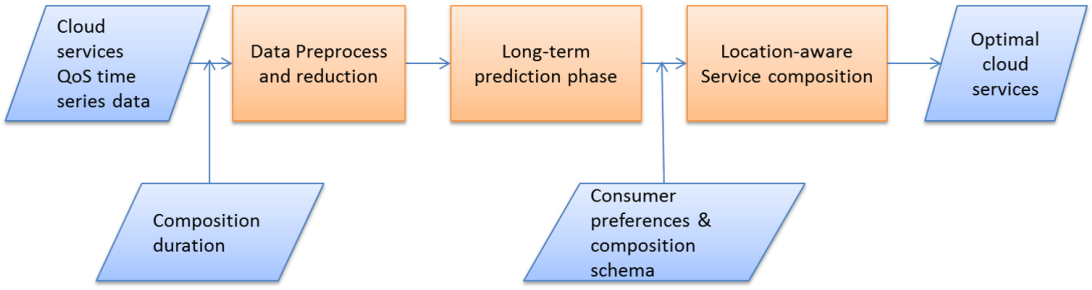

Figure 2. The suggested structure for service composition

4.1. Data preprocess and reduction phase

In this phase, the QoS data will be preprocessed and normalized. Then, their dimensions are reduced using an autoencoder network. The main goal of this phase is to reduce the execution time required for predicting the long-term QoS data.

4.1.1. Data preprocess

As described in our previous work [1], there are three data preprocessing stages used to increase the forecasting accuracy. In accordance with the first step, noise is removed from the QoS time series by using the wavelet transform. Data normalization is necessary in the second stage and is done by applying (6) to scale converting information from the original space to actual numbers between 0 and 1.

$$y = \frac{x-min}{max-min} \tag{6}$$

Where the observation's initial value (x) and its new scaled value (y) are represented, respectively, and min and max represent the low and high points of the series. At stage three, in order to transform QoS information into supervised machine learning form *(X, Y)*, where *X* represents patterns for input and *Y* represents patterns for output/label, it must first be converted using time-series format, which is a list of observational metrics beginning with *x1*, *x2*, ..., *xn*. The shift function in this phase is used to advance and regress the observations using the anticipated/next observations and the planned duration of lagged observations *(X)*, *(Y)*.

4.1.2. Dimensionality reduction

Many dimensionality reduction methods, including the discrete Fourier transform (DFT) and singular value decomposition (SVD), have been proposed for the transformation of time series data [29], principal component analysis (PCA) [30], discrete wavelet transform (DWT) [31] and piecewise aggregate approximation (PAA) [29]. Time series segmentation is a different method for dimensionality reduction [32]. These methods now in use automatically select the amount of hidden variables to be used while also identifying hidden variables among n input streams. According to their distributions and correlations, the observed values of the hidden variables display the general pattern of various input series. Due to the lack of a well-defined similarity measure, these current methods cannot be easily expanded for the tags similarity search problem. Most crucially, the relationships between the various time series cannot be handled by present techniques.

Recently, the autoencoder network is widely used for dimension reduction. Autoencoders are simple learning networks that aim to transform inputs into outputs with the least possible amount of distortion [33], [34]. As shown in Figures 3 and 4, Autoencoders are artificial neural networks (ANN) with symmetric structure, where the middle layer represents an encoding of the input data [35]. They are trained to reconstruct their input onto the output layer while verifying certain restrictions which prevent them from simply copying the data along with the network [36]. Starting with a short code that ignores noise, Autoencoders condense the original data. The code is then uncompressed to create an image that is as near to the original input as feasible. According to the work in this paper, the Autoencoder network has been constructed to reduce the search dimensions.

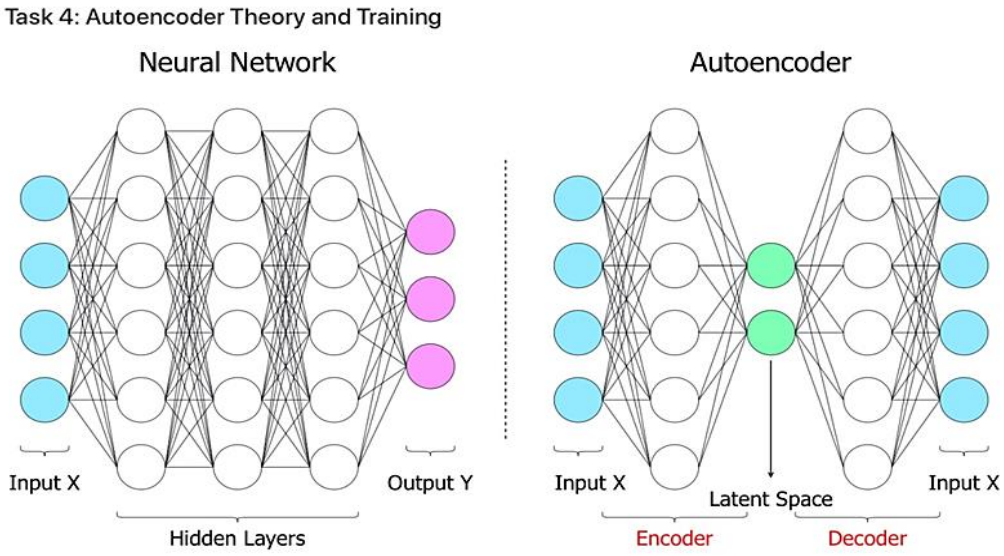

Figure 3. The autoencoder network [34]

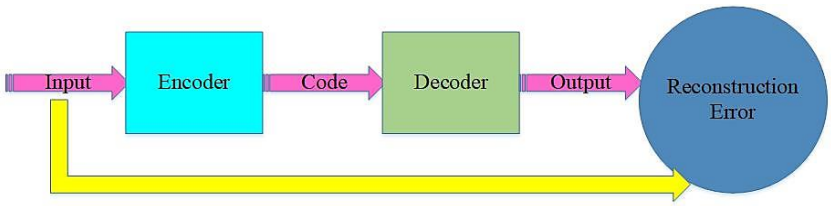

Figure 4. visualization description of autoencoder [37]

4.2. Long term prediction

In our earlier work [1], we showed how extended sequences may be learned and predicted using LSTM. The forget gate, input gate, update gate, and output gate are the four components that make up an LSTM memory cell [26]. These gates regulate the interactions between memory cells that are close to one another as well as the memory cell itself. The status of the cell is compared to a conveyor. The unrolled LSTM network and the method for updating each gate value are shown in Figure 5. Where, at time *t*, the memory cell's input vector (x_t) and output result (h_t) are, respectively, the input vector and result. The input gate's i_t, forget gate's f_t, and output gate's o_t values, respectively, are those at time *t*. i_t, f_t and o_t are the values of the input gate, the forget gate, and the output gate at time *t*, respectively. The memory cell's potential state at time *t* is denoted by $\tilde{C}_t$.

According to our experiments, the layer LSTM, the layer's density and dropout procedure make up the majority of the prediction model architecture as shown in Figure 6. First, input matrices comprised of four lagged the LSTM layer receives four next values/look ahead values in addition to values for each QoS attribute *(X) (Y)*. The dropout technique is then used on the output of the LSTM layer to avoid over-fitting the DLSC prediction model. The dense layer, a neural network (NN) that is fully linked and has a linear activation function, is where the output from the dropout process passes after that. MSE is then determined as the loss function. The "Adam" optimizer is used to optimize the model parameters in order to minimize the loss function. The dropout parameter is adjusted to 0.2 following extensive studies to determine the optimal way to train the network and obtain the best generalization scenario. The network will not exist without the dropout.

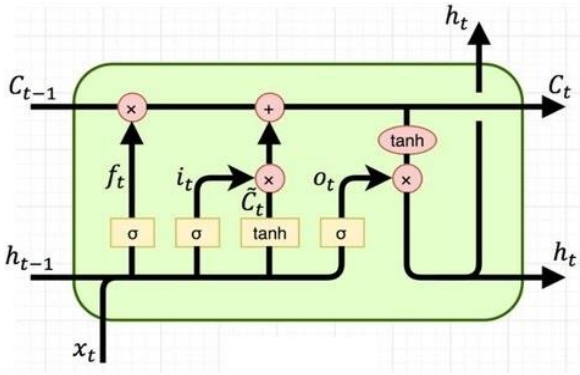

Figure 5. LSTM cell [26]

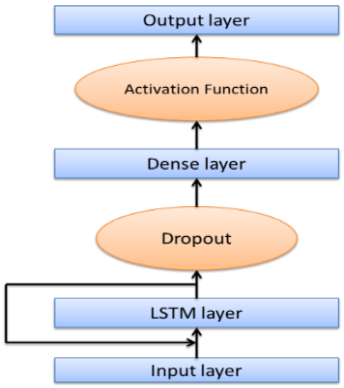

Figure 6. Flow chart of the L-DLSC prediction model

4.3. Location-aware service composition

Location is a crucial factor for recommendation of decision making where the QoS often depends heavily on the user locations or service locations. For example, a user in Australia may access a web service hosted in Australia very quickly, but it is very slow to access a web service hosted in China [27]. Therefore, location is a crucial factor that impacts the accuracy of composition results and the satisfaction of the user.

In this paper, the Haversine formula has been used to calculate the location factor [38]. The haversine formula determines the great-circle distance between two points on a sphere given their longitudes and latitudes. The law of haversines is important in navigation; it is a special case of a more general formula in spherical trigonometry, which relates the sides and angles of spherical triangles. Therefore, the haversine formula is considered an accurate way of computing distances between two points on the surface of a sphere

using the latitude and longitude of the two points. The haversine formula is a re-formulation of the spherical law of cosines, but the formulation in terms of haversines is more useful for small angles and distances.

$$hav(\theta) = sin^2\left(\frac{\theta}{2}\right) = \frac{1-\cos(\theta)}{2} \quad (7)$$

Where, θ is the central angle between any two points on a sphere, such that, $\theta=d/r$, d is the distance between the two points along a great circle of the sphere, r is the radius of the sphere. Therefore, during the composition process using the PSO algorithm, the distance of the candidate services is determined using the haversine formula as illustrated in the following algorithm [39].

Algorithm. Calculate distance

```
Inputs:
1. Lat1: customer location latitude in decimal
2. lon1: customer location longitude in decimal
3. lat2: candidate cloud service location latitude in decimal
4. lon2: candidate cloud service location latitude in decimal
Output: distance: distance between the customer and the candidate cloud service
   R=6373      // radius of Earth
   Convert Lat1,lon1,lat2, lon2 into degrees
   dlon= lon2-lon1     //change in coordinates
   dlat= lat2-lat1
   //Haversine formula
   a= math.sin(dlat/2)^2+ math.cos(lat1)*math.cos(lat2)*math.sin(dlon/2)^2
   C=2 * math.atan2(math.sqrt(a),math.sqrt(1 − a))
   distance=R*c
   Return distance
```

The services with a high distance value from the client are penalized under our methodology. The following is a possible way to represent a penalty on service *x* in a certain candidate plan.

$$\text{Penalty(X)} = 1 - \frac{\text{distance(X)}}{\sum_{i=1}^{n}\text{distance}(x_i)} \quad (8)$$

Therefore, the objective function of this candidate plan is determined over the entire anticipated period *T*, written as (9).

$$\text{F(objective)}=\max\{\left(\int_0^T s(t)dt\right) * \prod_{i=1}^{n}\text{Penalty}(x_i)\} \quad (9)$$

In a candidate constructed service plan, *n* denotes the total number of cloud services that are engaged.

5. RESULTS AND DISCUSSION

Several tests are conducted and contrasted with the existing framework in [16] and the earlier framework provided in [1]. These tests are using the same environment setting in order to evaluate the performance of the proposed service composition (L-DLSC) framework. The tests are being conducted on a PC with a 2.80 GHz processor, 8.0 GB of RAM, and Windows 10 and Python 3.7 installed.

5.1. Data description

During our experiments, we adopted a popular real-world web service dataset, the web services (WS-Dream) dataset [40]. This data set includes 339 users (from 31 countries) and 5825 services (hosted in 74 countries), as well as a historical time series QoS data for 100 cloud service providers, response time, throughput, and cost metrics were gathered over the course of six months as 28 time slots [41]. The WS-Dream dataset is used to conduct the IoT-existing framework while the later QoS dataset is used in our previous framework. Here, all experiments are conducted using WS-Dream after adding cost attribute using the same strategy as in [41].

5.2. Execution time

Figure 7 shows the comparison results among the proposed service composition L-DLSC framework, the existing IoT framework in [16], and the previous DLSC framework given in [1] concerning the execution time parameter. To conduct the cost-time analysis, Figure 7 highlights the value of time is growing by increasing the quantity of composing services. Nevertheless, the current IoT framework [16]

whose execution duration is the longest in comparison to our proposed L-DLSC framework due to the exhaustive task of decomposing the QoS-matrix into sub-matrices based on the user location then building user indices based on the similarity measures between the user and the candidates' services. Concerning the proposed L-DLSC framework, a suitable penalty on the long distances between the candidate services and the user is needed to locate the top writing services at a fair price. The typical time required to discover the ideal composition plan using the proposed L-DLSC framework 50% more is added in comparison to our previous DLSC framework, while the time cost of the IoT Composition framework is increased by 96% relative to our previous DLSC framework. Therefore, according to the output of these experiments, the proposed L-DLSC framework achieves significantly improved results for the time cost.

5.3. Optimality

In Figure 8, the values of the objective function in relation to the number of iterations are shown. According to the results given in Figure 8, it is clear that our proposed L-DLSC framework converges faster than the others, after which the current IoT framework [16]. Comparing our suggested L-DLSC framework to the prior DLSC framework presented in [1], the average best objective F score is raised by 44%. Although the average optimal objective F value utilizing the current IoT framework is higher than the proposed L-DLSC framework by 32%. This is due to the proposed L-DLSC framework lessens the services' weight which with a high distance from the customer, while increases the weight for the services which have a low distance from the customer. On the other hand, the existing IoT framework searches about the most similar ones to find the excellent services came from the decomposed matrices that are considered a pricey procedure. Besides, this proposed L-DLSC framework converges faster with higher composition values than the previous DLSC framework presented in [1].

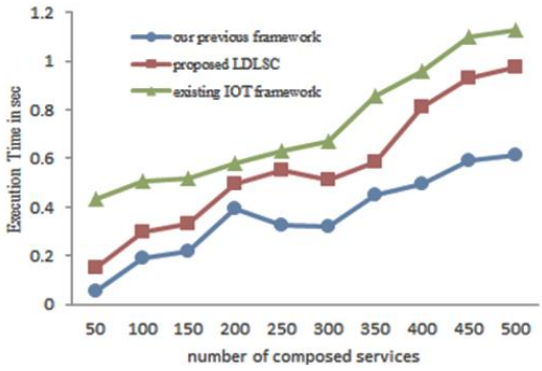

Figure 7. Execution time comparison

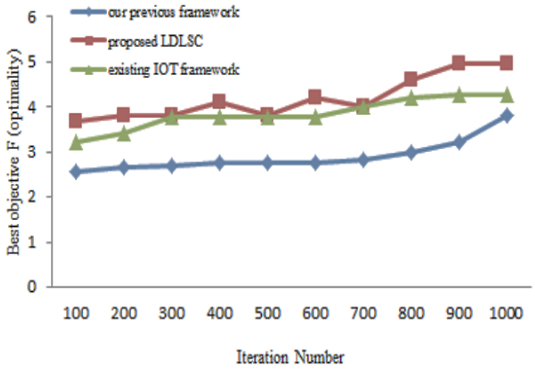

Figure 8. Optimality comparison

5.4. Effect of dimensional reduction

According to the proposed L-DLSC framework, the first aim is to reduce the dimensions of the composition problem. Therefore, the autoencoder network is added to perform this task. Figure 9 shows the effect of adding the autoencoder network. According to Figure 9, the accuracy of the long-term prediction process is compared after and before adding the autoencoder process in terms of the root mean squared error (RMSE) where smaller RMSE means better performance. According to the results in Figure 9, it is found that adding the autoencoder network significantly affects both the accuracy and the time required. Also, the prediction error has been reduced to 17% after adding the autoencoder due to capturing the most important features.

5.5. Effect of composing services aware of customer location

Figure 10 shows the comparison between our proposed L-DLSC framework and the previous DLSC framework in [1]. The comparison is done after adding the location parameter to pick the best service. As stated by findings in Figure 10, which demonstrate that the typical best objective F value using the new proposed L-DLSC framework is increased by 44% relative to the previous DLSC framework in [1].

5.6. Composition output

The output of the suggested L-DLSC framework, or the best composition services, is shown in Figure 11, where each number in the matrix corresponds to a service ID. Additionally, based on their functional requirements, the impact for the average whole dataset using the DLSC framework is divided into

three groups. Additionally, a straightforward design for service composition is taken into account, utilizing three service classes that are continuously running throughout four periods to create the composite service.

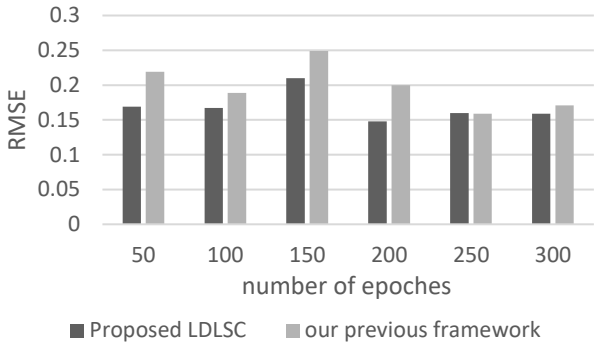

Figure 9. Effect of adding autoencoder

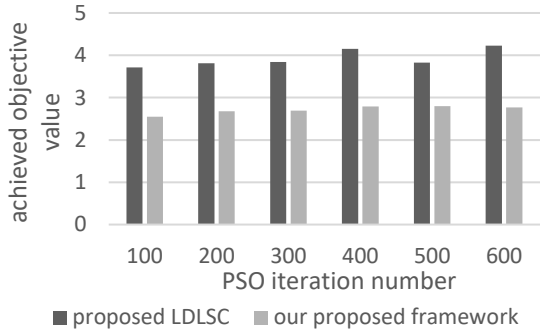

Figure 10. Effect of the location parameter

```
Best Composition Plan:
Best Global Positions During the long Term Period:
    SC1  SC2  SC3
T1    2    0   20
T2    0    0   20
T3    1   18   20
T4    0   22   20
Best Global Fitness value:   3.320404344914844

: <__main__.PSO at 0x6322803848>
```

Figure 11. Composition output

6. CONCLUSION AND FUTURE WORK

To reliably recommend cloud services based on QoS features, the location-aware framework for service composition based on deep learning (L-DLSC) is presented in this work. First, by lowering the dimensionality of the data, the Autoencoder network has been employed to narrow the search space. Then, a combination of the deep learning LSTM network and PSO algorithm is used to solve multi-objective problems while also leveraging the strength of the LSTM network when keeping track of time series' multivariate QoS characteristics dependencies. Last but not least, the location factor has been taken into account when choosing the best composite services. We may infer from the trials that have been done that our suggested L-DLSC architecture significantly improves the outcomes in the cloud service composition environment. Our proposed framework will be enhanced in the future by the inclusion of new elements like security and privacy in the cloud composition issue.

ACKNOWLEDGEMENTS

This work is financially supported by the Academy for Scientific Research and Technology (ASRT), Egypt, Science UP Grant No. 6646.

REFERENCES

[1] S. Haytamy and F. Omara, "A deep learning based framework for optimizing cloud consumer QoS-based service composition," *Computing*, vol. 102, no. 5, pp. 1117–1137, May 2020, doi: 10.1007/s00607-019-00784-7.

[2] I. Odun-Ayo, T.-A. Williams, M. Odusami, and J. Yahaya, "A systematic mapping study of performance analysis and modelling of cloud systems and applications," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 11, no. 2, pp. 1839–1848, Apr. 2021, doi: 10.11591/ijece.v11i2.pp1839-1848.

[3] D. Sudha and S. Chitnis, "An energy optimization with improved QOS approach for adaptive cloud resources," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 10, no. 5, pp. 4881–4891, Oct. 2020, doi: 10.11591/ijece.v10i5.pp4881-4891.

[4] L. Qian, Z. Luo, Y. Du, and L. Guo, "Cloud computing: an overview," in *IEEE international conference on cloud computing*, 2009, pp. 626–631.

[5] F. Mohammed, A. M. Ali, A. S. A.-M. Al-Ghamdi, F. Alsolami, S. M. Shamsuddin, and F. E. Eassa, "Cloud computing services: taxonomy of discovery approaches and extraction solutions," *Symmetry*, vol. 12, no. 8, Aug. 2020, doi: 10.3390/sym12081354.

[6] A. Ibrahimi, "Cloud computing: pricing model," *International Journal of Advanced Computer Science and Applications*, vol. 8, no. 6, 2017, doi: 10.14569/IJACSA.2017.080658.

[7] Y. Jin, K. Wang, Y. Zhang, and Y. Yan, "Neighborhood-aware web service quality prediction using deep learning," *EURASIP Journal on Wireless Communications and Networking*, no. 1, Dec. 2019, doi: 10.1186/s13638-019-1525-y.

[8] A. Jula, E. Sundararajan, and Z. Othman, "Cloud computing service composition: A systematic literature review," *Expert Systems with Applications*, vol. 41, no. 8, pp. 3809–3824, Jun. 2014, doi: 10.1016/j.eswa.2013.12.017.

[9] Q. She, X. Wei, G. Nie, and D. Chen, "QoS-aware cloud service composition: A systematic mapping study from the perspective of computational intelligence," *Expert Systems with Applications*, vol. 138, Dec. 2019, doi: 10.1016/j.eswa.2019.07.021.

[10] M. Hosseinzadeh *et al.*, "A hybrid service selection and composition model for cloud-edge computing in the internet of things," *IEEE Access*, vol. 8, pp. 85939–85949, 2020, doi: 10.1109/ACCESS.2020.2992262.

[11] L. J. Gadhavi and M. D. Bhavsar, "Prediction based efficient resource provisioning and its impact on QoS parameters in the cloud environment," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 8, no. 6, pp. 5359–5370, Dec. 2018, doi: 10.11591/ijece.v8i6.pp5359-5370.

[12] B. Savarala and P. Chella, "An improved fruit fly optimization algorithm for QoS aware cloud service composition," *International Journal of Intelligent Engineering and Systems*, vol. 10, no. 5, pp. 105–114, Oct. 2017, doi: 10.22266/ijies2017.1031.12.

[13] S. K. Gavvala, C. Jatoth, G. R. Gangadharan, and R. Buyya, "QoS-aware cloud service composition using eagle strategy," *Future Generation Computer Systems*, vol. 90, pp. 273–290, Jan. 2019, doi: 10.1016/j.future.2018.07.062.

[14] Z. Ye, A. Bouguettaya, and X. Zhou, "QoS-aware cloud service composition using time series," in *Service-Oriented Computing*, Springer Berlin Heidelberg, 2013, pp. 9–22. doi: 10.1007/978-3-642-45005-1_2.

[15] H. Labbaci, B. Medjahed, and Y. Aklouf, "A Deep learning approach for long term QoS-compliant service composition," in *International Conference on Service-Oriented Computing*, Springer International Publishing, 2017, pp. 287–294.

[16] P. Asghari, A. M. Rahmani, and H. H. S. Javadi, "Privacy-aware cloud service composition based on QoS optimization in Internet of Things," *Journal of Ambient Intelligence and Humanized Computing*, Jan. 2020, doi: 10.1007/s12652-020-01723-7.

[17] Z. Ye, S. Mistry, A. Bouguettaya, and H. Dong, "Long-term QoS-aware cloud service composition using multivariate time series analysis," *IEEE Transactions on Services Computing*, vol. 9, no. 3, pp. 382–393, May 2016, doi: 10.1109/TSC.2014.2373366.

[18] Q. Liang, X. Wu, and H. C. Lau, "Optimizing service systems based on application-level QoS," *IEEE Transactions on Services Computing*, vol. 2, no. 2, pp. 108–121, Apr. 2009, doi: 10.1109/TSC.2009.13.

[19] L. Qu, Y. Wang, and M. A. Orgun, "Cloud service selection based on the aggregation of user feedback and quantitative performance assessment," in *2013 IEEE International Conference on Services Computing*, Jun. 2013, pp. 152–159. doi: 10.1109/SCC.2013.92.

[20] B. Wu, C.-H. Chi, Z. Chen, M. Gu, and J. Sun, "Workflow-based resource allocation to optimize overall performance of composite services," *Future Generation Computer Systems*, vol. 25, no. 3, pp. 199–212, Mar. 2009, doi: 10.1016/j.future.2008.06.003.

[21] L. Zeng, B. Benatallah, A. H. H. Ngu, M. Dumas, J. Kalagnanam, and H. Chang, "QoS-aware middleware for Web services composition," *IEEE Transactions on Software Engineering*, vol. 30, no. 5, pp. 311–327, May 2004, doi: 10.1109/TSE.2004.11.

[22] Z. Ye, X. Zhou, and A. Bouguettaya, "Genetic algorithm based QoS-aware service compositions in cloud computing," in *Database Systems for Advanced Applications*, Springer Berlin Heidelberg, 2011, pp. 321–334. doi: 10.1007/978-3-642-20152-3_24.

[23] J. Zhu, P. He, Z. Zheng, and M. R. Lyu, "Towards online, accurate, and scalable QoS prediction for runtime service adaptation," in *2014 IEEE 34th International Conference on Distributed Computing Systems*, Jun. 2014, pp. 318–327, doi: 10.1109/ICDCS.2014.40.

[24] S. Liu, Y. Wei, K. Tang, A. K. Qin, and X. Yao, "QoS-aware long-term based service composition in cloud computing," in *2015 IEEE Congress on Evolutionary Computation (CEC)*, May 2015, pp. 3362–3369. doi: 10.1109/CEC.2015.7257311.

[25] S. Mistry, A. Bouguettaya, H. Dong, and A. K. Qin, "Predicting dynamic requests behavior in long-term IaaS service composition," in *2015 IEEE International Conference on Web Services*, Jun. 2015, pp. 49–56. doi: 10.1109/ICWS.2015.17.

[26] C. Olah, "Understanding LSTM Networks," *colah's blog*. 2015. [Online]. Available: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

[27] 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, doi: 10.1109/TSC.2015.2433251.

[28] W. Lin *et al.*, "Location-aware service recommendations with privacy-preservation in the internet of things," *IEEE Transactions on Computational Social Systems*, vol. 8, no. 1, pp. 227–235, Feb. 2021, doi: 10.1109/TCSS.2020.2965234.

[29] E. Keogh, K. Chakrabarti, M. Pazzani, and S. Mehrotra, "Dimensionality reduction for fast similarity search in large time series databases," *Knowledge and Information Systems*, vol. 3, no. 3, pp. 263–286, Aug. 2001, doi: 10.1007/PL00011669.

[30] S. Papadimitriou, J. Sun, and C. Faloutsos, "Streaming pattern discovery in multiple time-series," in *Proceedings of the 31st international conference on Very large data bases*, 2005, pp. 697–708.
[31] T. Kahveci and A. Singh, "Variable length queries for time series data," in *Proceedings 17th International Conference on Data Engineering*, pp. 273–282. doi: 10.1109/ICDE.2001.914838.
[32] W. Jiang, D. Lee, and S. Hu, "Large-scale longitudinal analysis of SOAP-based and RESTful web services," in *2012 IEEE 19th International Conference on Web Services*, Jun. 2012, pp. 218–225. doi: 10.1109/ICWS.2012.45.
[33] P. Baldi, "Autoencoders, unsupervised learning, and deep architectures," in *Proceedings of ICML workshop on unsupervised and transfer learning*, 2012, pp. 37–49.
[34] S. A. Ebiaredoh-Mienye, E. Esenogho, and T. G. Swart, "Artificial neural network technique for improving prediction of credit card default: A stacked sparse autoencoder approach," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 11, no. 5, pp. 4392–4402, Oct. 2021, doi: 10.11591/ijece.v11i5.pp4392-4402.
[35] D. Charte, F. Charte, S. García, M. J. del Jesus, and F. Herrera, "A practical tutorial on autoencoders for nonlinear feature fusion: Taxonomy, models, software and guidelines," *Information Fusion*, vol. 44, pp. 78–96, Nov. 2018, doi: 10.1016/j.inffus.2017.12.007.
[36] Y. Wang, H. Yao, and S. Zhao, "Auto-encoder based dimensionality reduction," *Neurocomputing*, vol. 184, pp. 232–242, Apr. 2016, doi: 10.1016/j.neucom.2015.08.104.
[37] H. O. A. Ahmed, M. L. D. Wong, and A. K. Nandi, "Intelligent condition monitoring method for bearing faults from highly compressed measurements using sparse over-complete features," *Mechanical Systems and Signal Processing*, vol. 99, pp. 459–477, Jan. 2018, doi: 10.1016/j.ymssp.2017.06.027.
[38] P. V Ingole and M. K. Nichat, "Landmark based shortest path detection by using Dijkestra algorithm and haversine formula," *International Journal of Engineering Research and Applications (IJERA)*, vol. 3, no. 3, pp. 162–165, 2013.
[39] Movable Type, "Calculate distance, bearing and more between latitude/longitude points," *Movable Type Scripts*. Accessed: Feb. 10, 2021. [Online]. Available: https://www.movable-type.co.uk/scripts/latlong.html
[40] WS-Dream Team, "WS-DREAM," *The Chinese University of Hong Kong*. Accessed: Mar. 30, 2020. [Online]. Available: https://wsdream.github.io/
[41] S. S. H. Haytamy, H. A. Kholidy, and F. A. Omara, "ICSD: integrated cloud services dataset," in *Services 2018*, Springer International Publishing, 2018, pp. 18–30. doi: 10.1007/978-3-319-94472-2_2.

BIOGRAPHIES OF AUTHORS

Alshaimaa M. Mohammed is a teacher at Faculty of Science, Suez Canal University, Ismailia, Egypt where she obtained her BSc. In 2009 she completed Pre- Master courses in computer at faculty of Science, Suez Canal University, Ismailia, Egypt. In 2014 she completed her master's degree. In 2019 she completed her PH. D degree. Her interests are security and trust in cloud computing environment, IoT, and big data. She can be contacted at email: shaimaa_mostafa@science.suez.edu.eg.

Samar Shaaban Abdelfattah Haytamy is an assistant lecturer in the department of computer science, faculty of computers and information, Fayoum university, Egypt. Samar received her MSc in computer science in 2020. Her research interests are machine learning, federation learning, service-oriented computing, open-source software, cloud computing, parallel programming, and big data. She can be contacted at email: Ssa10@fayoum.com.

Fatma A. Omara is a Professor in the Computer Science Department, Faculty of Computers and Information, Cairo University. She has published over 120 research papers in prestigious international journals, and conference proceedings. She has served as Chairman and member of Steering Committees and Program Committees of several national and International Conferences. She has supervised over 50 Ph.D. and M. Sc thesis. Prof. Omara is a member of the IEEE and the IEEE Computer Society. Prof. Omara interests are parallel and distributed computing, distributed operating systems, high performance computing, grid computing, cloud computing, big data, IoT. She can be contacted at email: f.omara@fci-cu.edu.eg.