Accurate Fault Location using Deep Neural Evolution Network in Cloud Data Center Interconnection

Hui Yang, *Senior Member, IEEE*, Xudong Zhao, Qiuyan Yao, Ao Yu, Jie Zhang, and Yuefeng Ji

Abstract—Due to the threat of failure and the discrete distribution of data center users, the research of distributed cloud data center provides real-time cloud services with robustness, reliability and security. Faced with data center interconnection, network failures cause mass services delay and interruption, which do a great damage to cloud computing. Many researchers have studied fault location methods in data center interconnection, which are easy to trap in local optimum limited by search capability and reduce the accuracy of location, especially when confronted with large-scale alarm information. In this article, the deep neural evolution network is introduced to extract deep-hidden fault features from massive collected alarm information in cloud data center interconnection. It has the prominent capacity of global search without the constraint of gradient to realize the breakthrough of fault location accuracy. The fault location method based on deep neural evolution network (FL-DNEN) is applied which uses the alarm set and suspicious scope of fault getting from fault propagation model as input and export deterministic faults accurately. The emulations demonstrate that the proposed method dramatically improves the accuracy of fault location to 92 percent with large-scale alarm information, which improves the resilience of cloud data center interconnection dramatically.

Index Terms—Deep neural evolution network, fault location, cloud data center interconnection

1 Introduction

With the cloud data center interconnection being the support technology of cloud computing [1], any failure in cloud inter-datacenter may have serious impact on the performance of cloud computing. For instance, in August 2013, Amazon aws services went down for approximately half an hour. Forbes estimated the revenue loss due to the data center downtime to be $66,240 per minute [2]. In the same month, all Google services carried by data centers went down for about 5 minutes causing a 40 percent drop in worldwide web traffic and an estimated loss of $500,000 [3]. In this scenario, the complexity of distributed cloud data center reflects in the following points. First of all, the sharp increase of cloud node and link scale would lead to more potential fault risks as well as massive alarm information. Furthermore, the fault location becomes more difficult because of the increase in flexibility of data centers [4]. Specifically, there are a large number of services carried by multi-datacenters with complex calling relationship, which have a great deal of interaction demand in inter-datacenter [5], [6]. In addition, the migration and backup of large amounts of data also increase the interaction between data centers [7]. Owing to the complexity of potential network states and inter-datacenter interactions, there will be more alarm sets when a failure occurs, which makes it difficult to ascertain the precise root cause of failure manifestation. Although the deployment of redundant backups on the key nodes can increase the robustness of data center, it is difficult to locate the faults and avoid them next time [8], [9]. In view of this, there is an urgent need for effective fault location. Based on a fault location method with accuracy and timeliness, the routing protocols of each layer could adjust the routing and avoid fault components in order to restore the data transmission in time. Moreover, in the era of 5G and beyond, the sharp increase of data center node scale and services categories would lead to more potential fault risks as well as massive alarm information in cloud data center interconnection, which increases the difficulty of highly accurate and timely fault location. Consequently, proposing a powerful fault location method is a burning issue to enhance the survivability and resilience of cloud data center interconnection.

The fault location issue in cloud data center interconnection could be abstracted as a mapping procedure mathematically. It is a highly non-convex problem to map the large-scale alarm information to the fault set, and an accurate mapping function corresponds to the global optimum solution. In fact, how to locate the fault from massive alarm sets has been proved to be a NPC problem [10] and has caught the attention of many researchers [11], [12], [13], [14]. Conventional fault location methods can be classified into active location and passive location. The principle of active location is to set up the monitor at the destination node in the data center network. The passive location technology is to locate faults according to the state of the working path in the networks. In general, the fault location accuracy rates of

• *The authors are with the State Key Laboratory of information Photonics and Optical Communications, Beijing University of Posts and Telecommunications, Beijing 100876, China. E-mail: {yanghui, lexzhao, yqy89716, yuao, lgr24, jyf}@bupt.edu.cn.*

Manuscript received 5 Apr. 2019; revised 19 Jan. 2020; accepted 12 Feb. 2020. Date of publication 17 Feb. 2020; date of current version 7 June 2022.
(Corresponding author: Hui Yang.)
Recommended for acceptance by V. C.M. Leung.
Digital Object Identifier no. 10.1109/TCC.2020.2974466

these approaches are lower than 50 percent, or the fault location time is at minute level. Consequently, proposing a powerful fault location method like FL-DNEN is a burning issue to improve the location accuracy rate and decrease the location time. Meanwhile, with the unmatched capacity of fitting and self-learning, artificial intelligence including machine-learning and deep-learning has been extensively introduced in various areas. Many researchers have proposed fault location algorithms based on AI methods such as SVM, bayesian network or neural network and so on [15], [16], [17], [18] in communication networks including backbone network. Ref. [15] proposed a fault location method to improve the quality of services in the virtual network topology through a Bayesian network. Ref. [16] proposed a fault location model based on Deep Belief Network (DBN-FL) to locate single-link fault of optical fronthaul network in 5G and beyond. Ref. [17] proposed a multi-sensor algorithm using Deep Belief Network (DBN) to solve the fault location problem in grid network. Ref. [18] presented a cognitive assurance architecture for backbone network failure management, which can find out suspicious fault sets by analyzing monitored data through machine learning model.

It can be seen from these papers that deep-learning based methods take advantages of deep neural network to extract deep-hidden fault features from massive collected alarm information and realize the breakthrough of fault location accuracy when compared with manual detection-based methods. However, the insufficiency of deep-learning based methods comes from following several aspects.

- *Restriction of Gradient.* Deep neural networks have fixed network topology and merely adjust network parameters based on gradient. In addition, because of heavy reliance on gradient, gradient descent is easy to fall into local optimum and hardly break away from it [19], [20], which reduces the accuracy of fault location when confronted with massive alarm sets.
- *Complicated Neural Network Structure.* According to the universal approximation theorem, deep neural networks tend to improve fitting effect by increasing the number of hidden units [21], [22]. However, the computational complexity of deep neural network with complicated topology increases the location latency because of the additional computational cost.
- *Disturb of Noise.* Irrelevant alarms intermingling together would increase the noise in alarm sets. The lack of an effective noisy processing will disturb the result and further affect the accuracy of fault location [23], [24].

Motivated by the drawback above, this paper proposes an accurate fault location method based on deep neural evolution network (FL-DNEN) to improve the accuracy of fault location in the scenario of massive alarm sets [25]. Deep neural evolution network (DNEN), evolving artificial neural networks with evolution strategy, has the capacity of global search without the constraint of gradient. With the appropriate encoding of neural network topology, offspring neural network C is generated from parent neural networks A and B by crossover and mutation and selected by the survival of fittest. Both the neural network weights and topologies can be optimized throughout the iteration, thus, DNEN is easier to find global optimum based on more freedom and possibilities and has been considered as the future of neural network. The proposed FL-DNEN consists of two steps. The first step is to use the improved fault propagation model to construct a mapping of alarm information to suspicious scope of faults. Additionally, in order to depress the noise of alarm information, this paper introduces a threshold mechanism in the fault propagation model. The second step uses the suspicious scope of fault obtained in the previous step as the input of DNEN supervised learning model and exports deterministic fault accurately. Meanwhile, FL-DNEN has the ability to decrease the fault location latency because of the good parallel computing capacity of genetic algorithm.

The rest of the paper is organized as follows. Section 2 analyzes the situations and difficulties of fault location in cloud data center interconnection. Section 3 introduces the principle of deep neural evolution network. The fault location method based on DNEN is proposed in Section 4. Then we describe the testbed and present the demonstration results and analysis in Section 5. Section 6 concludes the whole paper by summarizing our contribution and discussing our future work on this area.

2 Problem Analysis of Fault Location

In cloud data center interconnection, faults are reflected on various network monitoring devices through causal dependencies among various physical and logical entities, which generate a large number of alarm information. There are many indicators used as alarm information such as Q-factor, Bit Error Rate (BER), Optical Signal Noise Ratio (OSNR), Optical Power and so on [26], [27]. These indicators are gathered by a series of optoelectronic detectors, optical spectrum analyzers (OSAs) and other equipment deployed in cloud data center interconnection.

In addition, we have to consider the uniqueness of fault location in cloud inter-datacenter network compared with a regular backbone network. First of all, there is a serious requirement of convergence velocity in inter-datacenter network when a fault occurs, which means the lower tolerance of fault location time [28]. Second, the cloud datacenter interconnection increases the network scale, which is more complicated than a regular backbone network [29]. Third, there is a huge demand of interaction among inter-datacenter network [30], which causes massive alarm information.

There is a non-linear spatial-temporal correlation between faults and alarms. In the spatial domain, when a fault occurs, all downstream nodes passing through the fault will be affected. In the temporal domain, the generation of alarm has the characteristics of time sequence, which means a fault may trigger multiple alarm information even other faults. Furthermore, due to the complexity of network topology and the fault propagation model, the spatial-temporal relationship is more complicated. Specifically, owing to an increase in potential system states and in the complexity of component interactions, it can be difficult to ascertain the precise root cause of failure manifestation and its dependencies on components across the system. Meanwhile, due to the popularity of distributed and micro-service software architecture, services invocation relationship becomes more and more complex. When a failure occurs, the propagation

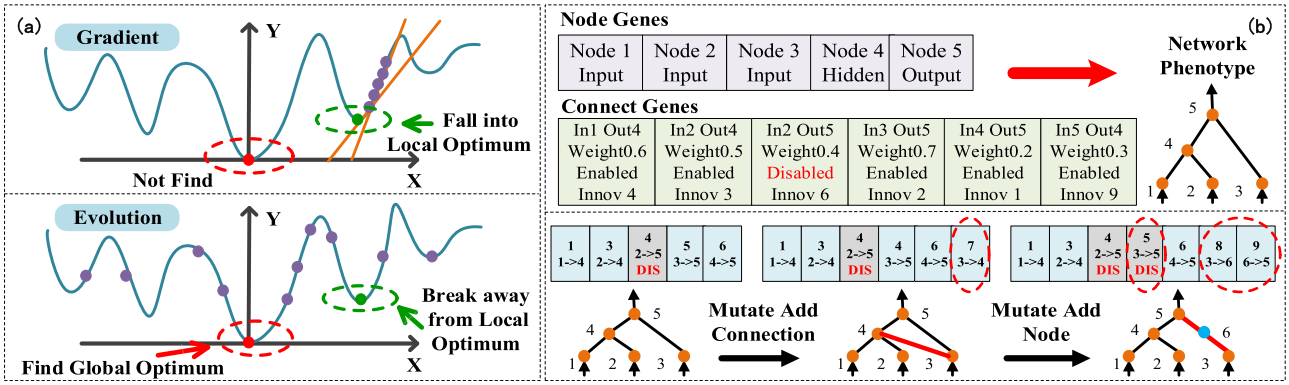

Fig. 1. (a) Different search forms between gradient and evolution, (b) Neural network evolution strategy.

of the fault is hard to control. Hence, there are still some challenges for fault location in cloud data center interconnection.

1) *Promote the Accuracy*. As mentioned above, it is a highly non-convex problem to map the large-scale alarm information to the deterministic faults, and an accurate mapping function corresponds to the global optimum solution. The larger the alarm sets, the larger the search space for the fault. Additionally, in cloud data center interconnection of 5G era, the impact of faults may be reflected in multi-dimensional alarm information. For instance, when an optical interface fault Reference Point 3 (RP3) occurs in network, the alarm information of both links and nodes will increase. It is hard to figure out whether the primary cause of the failure is from a link or a device node, or even the cross fault between both of them, which further increases the search space. Consequently, a fault location method with global optimization is of great importance.
2) *Decrease the Latency*. There is an ultra-high demand on the quality of service and user experience, resulting in low delay tolerance for fault location. Only if the faults are located rapidly, the routing protocols of each layer could adjust the routing and avoid fault components to restore the data transmission in time [31]. Out of the consideration of latency, a lot of approximate inference methods such as bayesian network are proposed to reduce complexity at the expense of accuracy. Thus, the balance between accuracy and latency is crucial to the reliability of future 5G network.
3) *Depress the Noise*. The alarm information and faults are not strongly correlated, specifically, one fault might cause multiple alarm information successively, and one alarm information might come from multiple faults. For instance, a Common Public Radio Interface (CPRI) failure may cause dozens of fiber link alarms and several optical module power alarms. So the increase of alarm information does not mean the increase of useful information, in fact, some alarms are false information as noise, which reduce the accuracy of fault location. Therefore, it is necessary to depress the noise effectively in the fault propagation model, which can enhance the accuracy and stability of fault location.

3 Principle of Deep Neural Evolution Network (DNEN)

In terms of fault location issue in cloud data center, it is a highly non-convex problem to map the large-scale alarm information to the fault set, and an accurate mapping function corresponds to the global optimum solution. Deep Neural network (DNN) and DNEN are two different ways to solve the problem. DNN uses the gradient descent method to optimize a single network model where the gradient points out the objective of optimization. The gradient will slide to the extreme point where the optimal solution will be got by adjusting parameters of neural network. Thus, the methods based on gradient such as stochastic gradient descent (SGD) are highly dependent on the initial network topology and parameters. Once network topology is inappropriate, the output of neural network is affected significantly. Instead, DNEN provides another alternative to make both topologies and parameters change. It generates many network model populations initially, and then the initial population screened by fitness (loss function) generate new models by crossover and mutation until the network models meet the accuracy requirement [32]. Both weights and topologies can be optimized throughout the iteration, thus, DNEN is easier to find global optimum based on more degrees of freedom and possibilities rather than more hidden layers. For instance, Ref. [33] proposed a tree-based encoding of a DNN that is searched through genetic programming and improved LSTM performance on based standard language modelling by 0.9 perplexity points. In Fig. 1a, the curve represents the entire search space, obviously, the gradient descent method is more likely to slide into local optimum, while DNEN can jump out of local optimum at any moment without the restriction of gradient. Therefore, DNEN with excellent global search capacity can be well applied to locate the fault in cloud data center interconnection.

As shown in Fig. 1b, the genome of neural network includes a series of connection genes, which connects two node genes. The connection gene consists of an input-node, an output-node, the weight of connection, an enable bit referring to express whether or not the connection and an innovation number used to find corresponding genes during crossover. Although the figure above evolves network with a single output, DNEN can evolve networks with any number of inputs or outputs. Mutation in DNEN optimizes both connection weights and network topology. There are

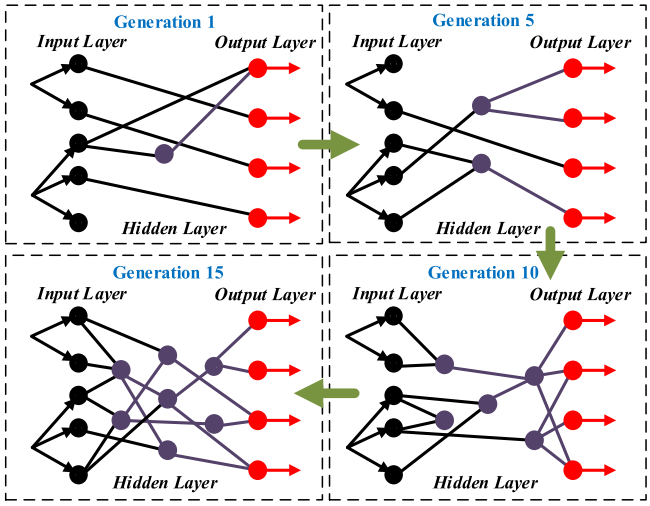

Fig. 2. The evolution of neural network topology based on DNEN.

two methods to expand the genome for structural mutations namely connection mutation and node mutation. In the mutation of adding node, a new connection gene is generated connecting two previously unconnected nodes. In the mutation of adding node, an existing connection is split to two new connections and a new node placed where the old connection used to be. In conventional evolutionary algorithms, it is likely that two individuals encode the same phenotype but with completely different genotype, which is called competing conventions [34]. The offspring is likely to be worse after the crossover of such individuals. In terms of the issue above, DNEN solves it by keeping historical markings of new structural elements. When a new structural element is generated including a new node or a new connection, it will be assigned an innovation number [35]. During crossover, the genotypes of two individuals are aligned by matching corresponding innovation numbers and only the different elements are exchanged.

Fig. 2 shows the evolution of neural network topology based on DNEN. A simple neural network is initialized with an input layer, an output layer and a single hidden layer and the connections among the neurons are sparse. With the crossover and mutation going on, the number of hidden layers grow more and more. Additionally, the connections between layers are more complicated from generation 1 to generation 15. It is obvious that DNEN offers more possibilities for neural network, rather than constrained to one fixed topology. In addition, each neural network in population could evolve independently by parallel computation. Using multi-thread computing or multi-core computing model, DNEN increases the training speed dramatically and meets the requirement of fault location with low latency in cloud data center interconnection.

4 The Proposed FL-DNEN

Based on DNEN described above, this paper proposes an accurate fault location method based on the deep neural evolution network (FL-DNEN). The method consists of two parts, an improved fault propagation model and a supervised learning model based on DNEN. With a threshold mechanism to depress noise, the improved fault propagation model maps alarm information to the suspicious scope of fault. The supervised learning model based on DNEN uses the suspicious fault sets as input and exports deterministic fault position accurately.

4.1 Improved Fault Propagation Model

In the most fault location systems, a fault propagation model is important as the basis of fault location, so is it in FL-DNEN proposed. In fault propagation model, the alarm information and the faults reflect appearance and essence respectively on fault location issue. As stated earlier, the alarm information and faults are not strongly correlated, the fault propagation model is capable to help us to establish the relationship between them and infer the suspicious faults scope from alarm information. Considering the noise in alarm sets, we propose a threshold mechanism limiting the number of node faults to improve the fault propagation model. Specifically, because of the requirement of network construction and average fault-free time (AFFT) for deployed services, cloud data center interconnection have certain reliability. Thus, the probability of multiple node faults occurring simultaneously is quite low. For instance, the AFFT of a data center node is usually one thousand to ten thousand hours, which means the probability of failure is 0.001 to 0.0001. For instance, in a data center topology with 1000 nodes, based on the probability of failure mentioned above, the probability of more than five simultaneous node failures is less than 0.0001. In this case, the threshold of the number of node failures is set to 4 and there is almost no influence on the location accuracy.

A typical software-defined data center (SDDC) is shown in Fig. 4a [36]. In the physical plane, optical packet switching (OPS) nodes constitute a mesh network, in the control plane, a central controller collects operation and maintenance data including alarm information from a serial of equipment deployed in network as well as analyzes the alarm information based on fault propagation model. When a failure occurs in link f, there is a large number of alarm information, such as the abnormal input and output optical power and FEC error, collected by the controller because of the connectivity of network topology. In order to reduce the impact of the failure link on the services carried on it, the improved fault propagation model map the alarm sets to suspicious faults, which is used as the input of supervised model based on DNEN. The pseudo code of concrete algorithm is shown in Fig. 3.

First is the initialization process. Collect all possible fault sets that may cause alarm set A_N and record them as F_{AN}. For each fault in F_{AN}, a threshold is introduced, which is the minimum number of alarm information that the fault can lead to. If the number of alarm information explained by the fault is less than the threshold, the alarm corresponding to the fault is considered to be false and will be removed from A_N. In addition, the corresponding fault will be removed from F_{AN}. Meanwhile, the original model is filtered, and only the directed links and nodes associated with A'_N and F'_{AN} after the treatment above are retained. With all possible fault set F'_{AN}, the alarm information in A'_N is sorted in descending order according to the number of corresponding faults in F'_{AN}. Take out of the first alarm A_i and find out the fault set F_{Si} leading to A_i. For each fault F_k in F_{Si}, remove the alarm explained by it from A'_N and generate

Algorithm 1 : Improved Fault Propagation Model

```
begin
Preprocess the model and Sort the element of A_N' descent
Get the first element A_i of A_N' and get F_Si
for all F_k in F_Si do
    if { A_N' - A_i} = ø
        Add {F_k} into F_re
    else
        Put {F_k} into L_f
        Put {A_N' - A_i} into L_a
    end
end for
while(L_f != null) do
    Get the first element of L_f : F_anaFi, and remove it from L_f
    Get the first element of L_a: A_needana, and remove it from L_a
    Get the first element of A_needana : A_i, and the set F_Si
    for all F_k in F_Si do
        if |F_anaFi| + 1 >= Ai_F or |A_needana - A_i| == 0
            Add F_anaFi ∪ {F_k} into F_re
        else
            Add F_anaFi ∪ {F_k} into L_f
            Add A_needana - A_i into L_a
        end
    end for
end while
end begin
```

Fig. 3. The pseudo code of improved fault propagation model.

an unexplained alarm set $A_{needana}$. Put F_k to L_f and put $A_{needana}$ to L_a. After the initialization process, we will get two Queue L_f and L_a, namely the suspicious fault set to be extended and alarm information to be explained.

Then, we will introduce how to find suspicious fault sets based on all observed alarm set A'_N. Take out the fault set F_{anaFi} from L_f in turn, and then extend this suspicious fault set according to corresponding alarm set $A_{needana}$. The detailed procedure is described as follows. Take out the first alarm A_i in $A_{needana}$ and get the fault set F_{Si} who is likely to lead to A_i. Since the elements in $A_{needana}$ are in order, A_i is the alarm with the largest number of corresponding faults definitely. For each fault F_k in F_{Si}, extend original F_{anaFi} using it and remove all the alarm information explained by it from $A_{needana}$. Finally, put the updated set of faults to be extended F_{anaFi} and the set of alarm to be explained $A_{needana}$ back into the corresponding queues. When in the process, once all the alarm is explained, or the size of the fault set reaches the maximum number of simultaneous faults Ai_F we set, the fault set is added to the result queue F_{re}, and the corresponding F_{anaFi} and $A_{needana}$ are no longer put back into the corresponding queue.

4.3 Supervised Learning Model Based on DNEN

In this subsection, we train the supervised learning model based on DNEN with the suspicious fault set F_{sus} obtained from the improved fault propagation model above as input. The training process of FL-DNEN is shown in Fig. 5.

To satisfy randomness of DNEN, we have to design initial neural network topology population and create a pool of them. Because the structure of DNEN is optimized continuously in evolution, there is no need to carefully design a complicated initial structure. Consistent with the traditional neural network, in DNEN model, the initial neural network is composed of input layer, hidden layers and output layer. In order to increase the non-linear nature, a variety of activation layers are attached to neural network appropriately. The hidden layers map input data to eigenspace, while output layer maps eigenspace to label space. As shown in Fig. 4b, the input data is the suspicious fault set F_{sus}, and the output of the model is the real location F in cloud data center interconnection. The input vector $x = [x_1, x_2, \ldots, x_n]$ and the corresponding real fault location y constitute a training sample $< x, y >$.

The neural network model between input and output is as Eq. (1). Where L is the layers number of neural network, $[n_0, n_1, n_2, \ldots, n_L]$ correspond to the dimensions of each layer, $[\varphi^{(1)}, \varphi^{(2)}, \varphi^{(3)}, \ldots, \varphi^{(n)}]$ are activation function.

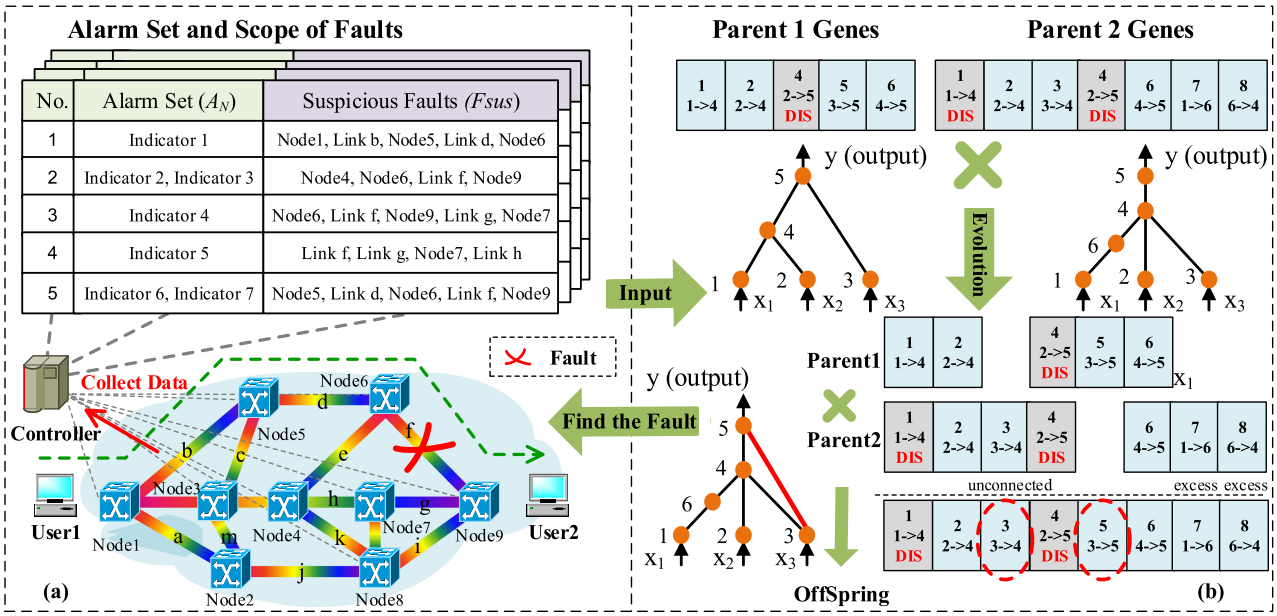

Fig. 4. (a) The fault scenario and alarm information in cloud data center interconnection. (b) The evolution of neural network in the training phase.

Algorithm 2 : FL-DNEN

Input: The suspicious fault set F_{sus}, where F_{sus}={$F_{sus}(i)|i=1,2,...$}
Output: The deterministic fault sets F, where F={$F(i)|i=1,2,...$}
01: Initialize the structure of neural network
02: Encode the neural network genome
03: Generate initial network population M, size(M)=m
04: **while** *bestFitness* < *threshold* **do**
05: **for** $i = 1$ **to** m **do**
06: $M_{i,fitness}$ = calculateFitness(M_i)
07: Selection the neural network with high fitness
08: Generate offspring by crossover and mutation
09: *bestFitness* = max(*bestFitness*, $M_{i,fitness}$)
10: The new network population is $M^{`}$ ,size($M^{`}$) = m
11: **end for**
12: **end**

Fig. 5. The pseudo code of FL-DNEN.

$$\begin{cases} h^{(l)} = \varphi^{(l)}\left(\sum_{i=1}^{n_l-1} {h_i}^{(l-1)} w_i^{(l)} + b^{(l)}\right) \\ l = 1, 2, \cdots, L \\ h^{(0)} = x \\ h^{(L)} = y \end{cases} \tag{1}$$

The relationship between input and output is as Eq. (2).

$$\begin{aligned} y = h^{(L)} &= \varphi^{(L)}\left(\sum_{i_L=1}^{n_L} {h_{i_L}}^{(L-1)} \cdot {w_{i_L}}^{(L)} + b^{(L)}\right) \\ &= \varphi^{(L)}\left(\sum_{i_L=1}^{n_L} \varphi^{(L-1)}\left(\sum_{i_L=1}^{n_L} {h_{i_{L-1}}}^{(L-2)} \cdot {w_{i_{L-1}}}^{(L-1)} + b^{(L-1)}\right) {w_{i_L}}^{(L)} + b^{(L)}\right) \\ &= \cdots = \varphi^{(L)}(\varphi^{(L-1)}(\cdots \varphi^{(1)}(x, \theta_1) \cdots, \theta_{L-1}), \theta_L). \end{aligned} \tag{2}$$

The objective function is shown as Eq. (3), which consists of loss term $L(\theta)$ and regularization term $R(\theta)$. $R(\theta)$ is introduced to avoid model over-fitting, and hyper parameter $\boldsymbol{\lambda}$ is introduced to control the degree of parameter penalty and make a trade-off between performance and generalization.

$$\min_{\theta} J(\theta) = L(\theta) + \lambda R(\theta). \tag{3}$$

The reasonable and effective encoding of neural network is an important step in the realization of neural evolution. As mentioned above, the genome of neural network includes a series of connection genes, which connects two node genes. The connection gene consists of an input-node, an output-node, the weight of connection, an enable bit referring to express whether or not the connection and an innovation number used to find corresponding genes during the crossover process. For the connection between neurons the adjacent relationship between vertices in adjacency matrix can be represented by graphs. For adjacent matrix A, if there is directed edge $< \mathrm{i,j} >$, $A[i, j]$ is equal to 1, otherwise, $A[i, j]$ will be 0. Because the structure of neural network is directed acyclic graph, it can be represented by encoding adjacency matrix as Eq. (4).

$$q^t = \begin{pmatrix} |\Phi_{11}^t > |\Phi_{12}^t > \cdots |\Phi_{1n}^t \\ |\Phi_{21}^t > |\Phi_{22}^t > \cdots |\Phi_{2n}^t \\ \vdots \\ |\Phi_{n1}^t > |\Phi_{n2}^t > \cdots |\Phi_{nn}^t \end{pmatrix} = \begin{pmatrix} \alpha_{11}^t & \alpha_{12}^t & \cdots & \alpha_{1n}^t \\ \beta_{11}^t & \beta_{12}^t & \cdots & \beta_{1n}^t \\ \alpha_{21}^t & \alpha_{22}^t & \cdots & \alpha_{2n}^t \\ \beta_{21}^t & \beta_{22}^t & \cdots & \beta_{2n}^t \\ \vdots & \vdots & & \vdots \\ \alpha_{n1}^t & \alpha_{n2}^t & \cdots & \alpha_{nn}^t \\ \beta_{n1}^t & \beta_{n2}^t & \cdots & \beta_{nn}^t \end{pmatrix}. \tag{4}$$

In the directed acyclic graph, $A[i, j]$ will be 0, so the initialization structure is as Eq. (5).

$$q^{t0} = \begin{pmatrix} 1 & 2/2 & \cdots & 2/2 \\ 0 & 2/2 & \cdots & 2/2 \\ 2/2 & 1 & \cdots & 2/2 \\ 2/2 & 0 & \cdots & 2/2 \\ \vdots & \vdots & & \vdots \\ 2/2 & 2/2 & \cdots & 1 \\ 2/2 & 2/2 & \cdots & 0 \end{pmatrix}. \tag{5}$$

The next step is crossover and mutation. Fig. 4b shows two parent neural networks generate offspring by crossover. For crossover, it is pivotal that two genomes with the same historical genesis have the same structure, since they were generated by the same parent in a past epoch. Therefore, we have to know which genes should be aligned to track the historical origin of each gene. Specifically, it is necessary to keep a global counter. When a connection or a neuron is added, the value of the counter should be assigned and set up increment. When it performs to crossover, the genes of both parents neural network with same innovation numbers are aligned. The compatibility of a pair of parents' genes can be measured by the number of unconnected and excess genes. The more unconnected a pair of genes are, the less historical origin they have, and the less compatible they are. Thus, we can compute the compatibility δ of two neural network topologies in DNEN on the basis of the unconnected (U), excess (E), and the average weight distance of matching (W) as Eq. (6). The coefficient a, b and c are used to adjust the weights of the three factors. N is the number of genes in the larger genome, which is used to normalize for the genome size.

$$\delta = \frac{aE}{N} + \frac{bU}{N} + c \cdot W. \tag{6}$$

After that, the reward or the utility will update parameters. In addition to mutation and crossover, selection by fitness is also indispensable. The loss function is as fitness of each neural network, the lower loss function is, the higher fitness is. In other words, the neural network with higher fitness has advantages of breeding offspring and retaining good characteristics. After each epoch of evolution, the current population will be updated. The progress mentioned above continues uninterruptedly until the fitness of new neural network satisfies the accurate demand of fault location.

5 Emulation Assessment

In this section, we present results with actual network failure data. Our goal is to demonstrate the fault location accuracy

TABLE 1
Alarm Indicator Names

Indicators	Indicators
output optical Power (dBm)	FEC error rate before
input optical Power (dBm)	FEC error rate after
temperature of Laser(°C)	average FEC error rate before
laser bias (mA)	average FEC error rate after
laser cooling (mA)	veneer temperature (°C)

and location time of the proposed method FL-DNEN with large-scale alarm sets.

5.1 Emulation Design and Model Training

In distributed cloud inter-datacenter, the raw alarm information consists of the alarm name, the device port, beginning and ending time, the name of network elements, the name of the circuit, the types of alarm, and the level of alarm, etc. Given the sensitive nature of network topology and device procurement data, we used a static snapshot of our network encompassing 4510 devices and tens of thousands of interfaces spread across tens of data centers.

Our testbed is a multi-core server with 8 physical 2.20 GHz CPU cores and 2 NVIDIA GTX TITAN XP GPU cores. The code is based on Keras 2.1.0 in Ubuntu 16.04.3. In order to verify the validity of the proposed method, we collect the fault data and the corresponding alarm information from the management system and OpenStack system of the network operator. The failure data were collected during a 90 days period from November 2017, which have 500 thousand level alarm information from 4510 nodes in cloud data center interconnection. Each fault occurred independently, which means no relativity among them. The alarm information has a consistent format including occurrence time, alarm address, alarm text and data. These training samples are divided into labeled sample and unlabeled sample. Given L and U as sets of labeled sample and unlabeled sample respectively, x and y represent the input vector and the output vector respectively.

Table 1 shows the primary indicator names of the alarm information, such as output optical power and input optical power. The suspicious fault sets obtained by the fault propagation model are used as the input of DNEN model, which are spilt to training, validation and testing set in proportion of 8:1:1.

In order to train the DNEN model, we have to design initial neural network topology population and create a pool

TABLE 2
Initial Neural Network Structure

Layer (type)	Output Shape	Param
dense_1 (Dense)	(None, 50)	1050
batch_normalization_1	(None, 50)	200
dropout_1 (Dropout)	(None, 50)	0
dense_2 (Dense)	(None, 10)	510
activation_1 (Activation)	(None, 10)	0
dense_3 (Dense)	(None, 1)	11
activation_2 (Activation)	(None, 1)	0
Total params: 1771		
Trainable params: 1671		
Non-trainable params: 100		

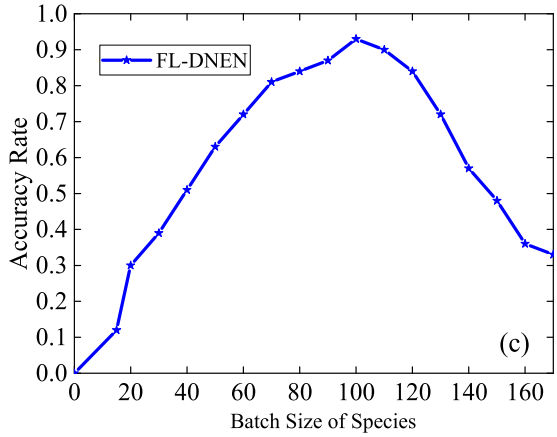

Fig. 6. The contrast of location accuracy rates using different batch size of species.

of them with a value of 150 in our emulation. Because the structure of DNEN is optimized continuously in evolution, there is no need to carefully design a complicated initial structure. In terms of the fault propagation model in our emulation, the maximum number of failures in suspicious fault set is 20, so the input layer dimension is 20. The model layers are shown in Table 2 successively. Besides of the three dense layers, a batch-normalization layer is introduced to accelerate convergence, and a dropout layer is used to avoid over-fitting. In addition, there are activation layers attached to the neural network, specifically a relu function and a sigmoid function, to ensure the non-linear nature of the issue. The output dimension is 1 corresponding to the deterministic fault location.

In the training phase, the maximum iteration epoch is 50, and the batch size of species is 100 (In Table 2, none means batch size is variable). It can be seen from the Fig. 6 that the batch size of species with 100 has the highest accuracy rate after training. Obviously, if the batch size of species is too small, the sample size of the neural network structures is insufficient, and if the value is too large, the convergence of the training results will be affected. To define the crossover function, we select the set of weights in our network that connect the adjacent layers and then swap these weights for the given two networks from the pool. For random mutations, we select the weights randomly with a probability of 0.15 and then change its value with a random number between -0.3 to +0.3, because Fig. 7 shows that the weights of random mutation between -0.3 to + 0.3 have a desired training result.

5.2 Fault Location results and Analysis

We compare the proposed FL-DNEN method with several deep learning-based and machine learning-based fault location methods, including DNN and SVM.

Figs. 8 and 9 show the contrast of location accuracy rates between noise processing and non-noise processing using FL-DNN and FL-DNEN respectively. As mentioned above in Section 4, a threshold mechanism in the fault propagation model is introduced to depress the noise of alarm information. In Figs. 8 and 9, when confronted with small-scale alarm information, the accuracy rates of noise processing are as the same as non-noise processing, even slightly lower, which is

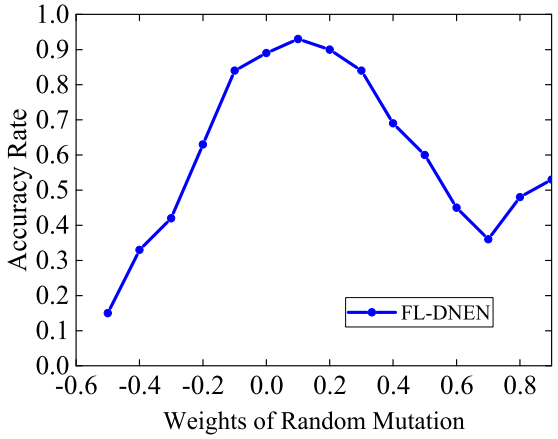

Fig. 7. The contrast of location accuracy rates using different weights of random mutation.

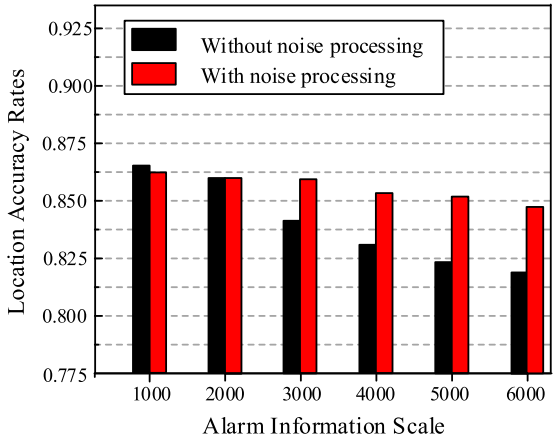

Fig. 8. The contrast of location accuracy rates using FL-DNN.

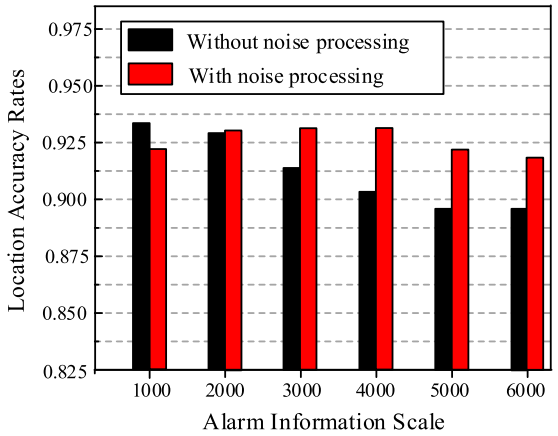

Fig. 9. The contrast of location accuracy rates using FL-DNEN.

because the useful information is likely to be eliminated by the threshold mechanism. But with the increase of alarm information, the accuracy rates of noise processing are significantly higher than non-noise processing. The larger the alarm information scales are, the greater the accuracy rate improvements are, which proves that the noise processing is necessary when confronted with large-scale alarm information.

As shown in Fig. 10, different colors are assigned to signify different neural network topologies. Along with the

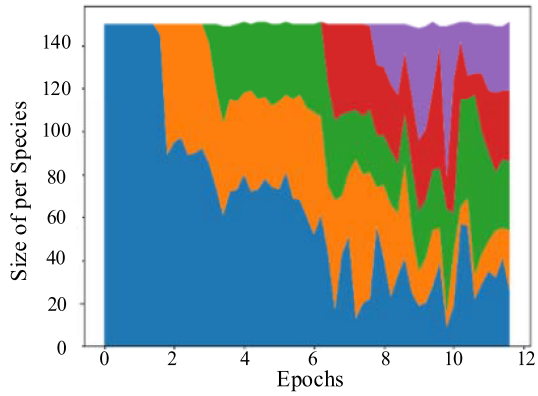

Fig. 10. The size of per species of deep neural network during training process.

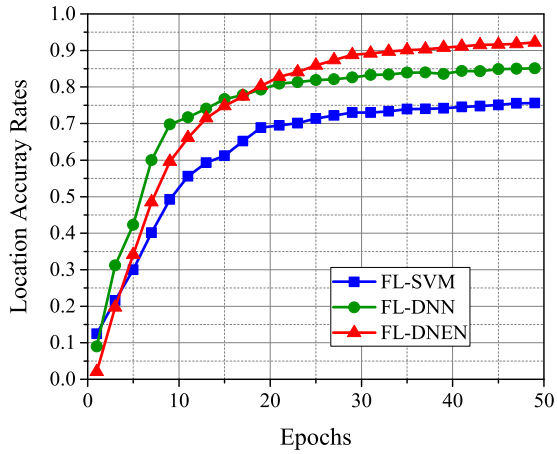

Fig. 11. The fault location accuracy rates with 10000 alarm information during training process.

increase of training epochs, the size of different neural network topologies, which are called species in DNEN, is in a continuous state of change. At the very beginning, there is just one species corresponding to the initial neural network structure in Table 2. After continuous evolution including crossover and mutation of parents, there are more and more species in different colors. The fitness function is simply the accuracy rate of fault location. For a given neural network, the higher the fitness score is for a given neural network, the fitter it is compared to the others, and more likely to produce offspring. The neural network topology meeting accurate demand of fault location will be selected eventually.

From the results shown in Fig. 11, at an early stage, FL-DNEN is less potent compared with FL-DNN and FL-SVM in terms of accuracy rates, because the initial species are generated randomly. However, in the wake of training epochs, the neural networks with higher fitness are screened out, which manifests in the strong uptrend of FL-DNEN and the highest accuracy rate after convergence. Result shows the accuracy rate of FL-DNEN with 10000 alarm information is more than 92 percent, which profits from the superiority of global search. The graph demonstrates that the accuracy rate of using FL-DNEN is over 5 percent to 8 percent higher than that of normal deep neural network, which has great prospects in realizing highly accurate fault location.

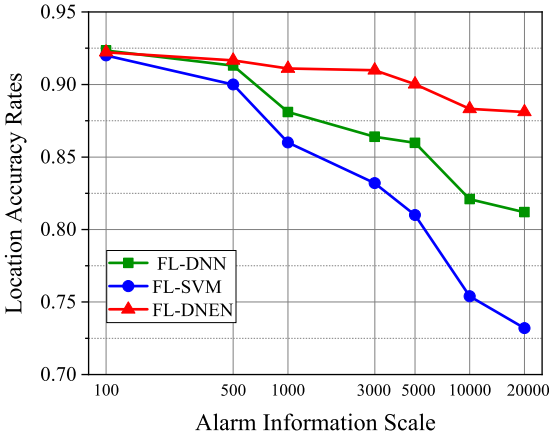

Fig. 12. The fault location accuracy rates with different scale of alarm information.

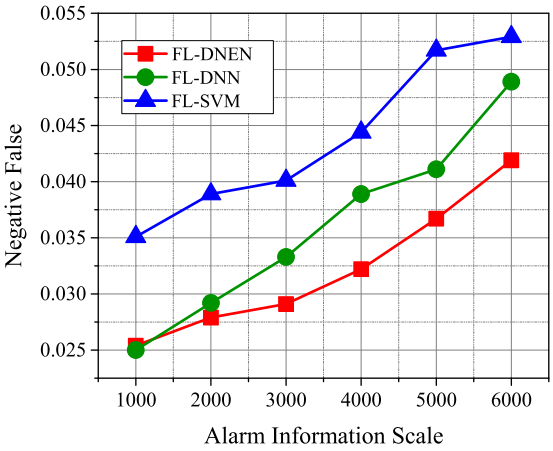

Fig. 13. The negative false rates with different scale of alarm information.

Fig. 12 shows the location accuracy rates of FL-DNEN compared with FL-SVM and FL-DNN in a large alarm scale interval from 100 to 20000. Because of their own advantages in small-scale data sets, the effects of the three methods are not of great difference from 100 to 500. But when comes to large-scale alarm, the potential logical relations between alarm and faults are more complex, and the global search ability of FL-DNEN is fully reflected. Limited by the defect of quadratic program in large-scale data set, the accuracy rate of FL-SVM is only about 75 percent. Additionally, compared with FL-DNN, FL-DNEN breaks through the bottleneck of local optimum due to neural evolution, and the accuracy rate is improved by 6 percent to 8 percent.

Excessive negative false means many faults cannot be located and avoided in time, which results in mass services connection interruption and severe network blocking. Thus, the fault location method should reduce the negative false as much as possible. As shown in Fig. 13, FL-DNEN has lowest negative false, especially confronted with large-scale alarm information. The essence of reducing negative false is to extract deep-hidden fault features from massive alarm as far as possible, more fault features mapped to eigenspace by neural network, more accurate it is to obtain the result. Obviously, due to the superiority of global search, FL-DNEN has more remarkable capability to extract features.

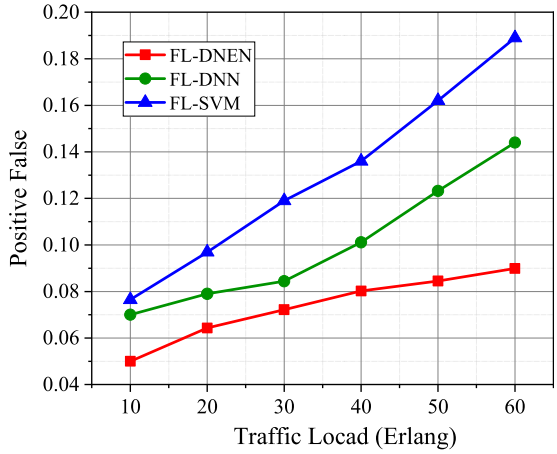

Fig. 14. The positive false rates with different scale of alarm information.

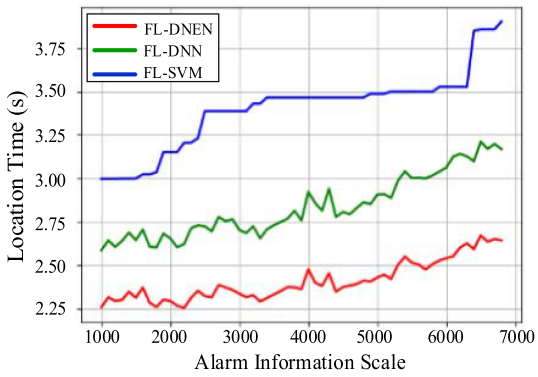

Fig. 15. The time of the fault location.

Excessive positive false means many normal components are located as faults. Although the avoidance of components doesn't lead to services interruption, it causes serious waste of network resources and even network congestion. In fact, the positive false of FL-DNN in validation set is well below it in test set, which means FL-DNN exists over-fitting. On the contrary, FL-DNEN help us exclude the neural network with over-fitting risk, just as Fig. 14 shows, the positive false of FL-DNEN is controlled at a low level in test set.

The location time is the total of time spent on fault propagation model and DNEN model. Fig. 15 indicates the fault location time of FL-DNEN is lower compared with FL-SVM and FL-DNN, especially when confronted with large-scale alarm information. The DNEN algorithm has the global convergence characteristics for fault location with initial drop and fast iteration based on the advantage of the above two methods. The time complexity of DNEN algorithm is $O(e^2)$ in this paper, because the DNEN algorithm is quadratic convergence. As for SVM, it is strongly associated with samples N and feature dimensions d, and the time complexity is $O(dN^2)$. Meantime, different from FL-SVM and FL-DNN, which are relying on two super-parameters obtained by exhaustive test and multiple layers network structure respectively, in FL-DNEN, each neural network in population could evolve independently using multi-thread computing or multi-core computing model. Just as Figs. 15 and 16 show, DNEN can save more than half of training time compared with DNN and SVM. In addition, compared with traditional

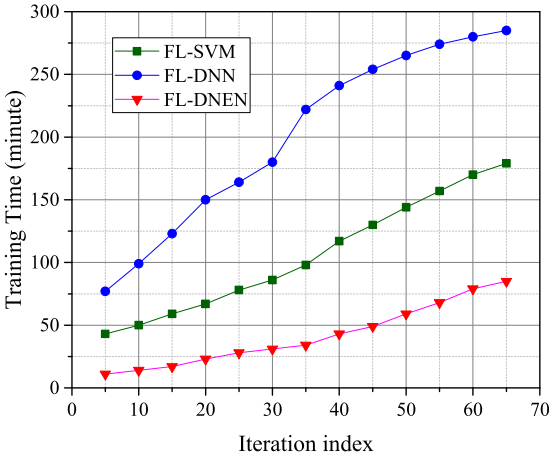

Fig. 16. The training time with iteration index.

fault location approaches which are at minute level, the 2.25 s fault location time is a significant progress in inter-datacenter. Moreover, with relevant fault self-healing strategies, when a fault occurs, the closed-loop strategy can be automatically executed based on the faults located by our approach. Obviously, while improving the accuracy, FL-DNEN is not at the expense of time.

6 Conclusion

Motivated by the drawback of insufficient fault location accuracy for existing methods in cloud data center interconnection, this paper proposes an accurate fault location method based on deep neural evolution network (FL-DNEN) to improve the accuracy of fault location in the scenario of massive alarm sets. The proposed FL-DNEN consists of two steps. The first step constructs mapping from alarm information to suspicious scope of faults using the improved fault propagation model. Additionally, in order to depress the noise of alarm information, this paper introduces a threshold mechanism in the fault propagation model. The second step uses the suspicious scope of fault getting from the previous step as input of DNENE model and exports deterministic fault. The emulations demonstrate that FL-DNEN can improve the accuracy of fault location to 92 percent with large-scale alarm information dramatically. Meanwhile, FL-DNEN has the ability to decrease fault location latency because of the good parallel computing capacity of genetic algorithm. In our future work, FL-DNEN can be used in more complicated scenario of fault location in cloud data center interconnection. In addition, FL-DNEN may be evaluated with larger alarm sets.

Acknowledgments

This work has been supported in part by NSFC project (61871056), in part by Young Elite Scientists Sponsorship Program by CAST (2018QNRC001), in part by Beijing Natural Science Foundation (4202050), in part by Fundamental Research Funds for the Central Universities (2018XKJC06, 2019PTB-009), in part by Fund of SKL of IPOC (BUPT) (IPOC2018A001, IPOC2019ZT01), and in part by ZTE Research Fund and Key Laboratory Fund (6142411182112, 614210419042, 61400040503, CEPNT-2017KF-04).

References

[1] H. Yang *et al.*, "CSO: Cross stratum optimization for optical as a service," *IEEE Commun. Mag.*, vol. 53, no. 8, pp. 130–139, Aug. 2015.
[2] Amazon.com Goes Down, Loses $66,240 Per Minute, 2013. [Online]. Available: https://www.forbes.com/sites/kellyclay/2013/08/19/amazon-com-goesdown-loses-66240-per-minute
[3] Google's downtime caused a 40% drop in global traffic, 2013. [Online]. Available: https://engineering.gosquared.com/googles-downtime-40-drop-in-traffic
[4] J. L. García-Dorado and S. G. Rao, "Cost-aware multi data-center bulk transfers in the cloud from a customer-side perspective," *IEEE Trans. Cloud Comput.*, vol. 7, no. 1, pp. 34–47, First quarter 2019.
[5] K. A. Saed *et al.*, "Data governance cloud security assessment at data center," in *Proc. 4th Int. Conf. Comput. Inf. Sci.*, 2018, pp. 1–4.
[6] H. Yang, Y. Liang, J. Yuan, Q. Yao, A. Yu, and J. Zhang,"Distributed blockchain-based trusted multi-domain collaboration for mobile edge computing in 5G and beyond," *IEEE Trans. Ind. Inf.*, to be published: 2020, doi: 10.1109/TII.2020.2964563.
[7] N. Zhang, P. Yang, J. Ren, D. Chen, L. Yu, and X. Shen, "Synergy of big data and 5G wireless networks: Opportunities, approaches, and challenges," *IEEE Wireless. Commun*, vol. 25, no. 1, pp. 12–18, Feb. 2018.
[8] H. Yang, J. Zhang, Y. Zhao, J. Han, Y. Lin, and Y. Lee, "SUDOI: Software defined networking for ubiquitous data center optical interconnection," *IEEE Commun. Mag.*, vol. 54, no. 2, pp. 86–95, Feb. 2016.
[9] H. Emesowum *et al.*, "Achieving a fault tolerant and reliable cloud data center network," in *Proc. IEEE Int. Conf. Services Comput.*, 2018, pp. 201–208.
[10] G. F. Cooper *et al.*, "The computational complexity of probabilistic inference using bayesian belief networks," *Artif. Intell.*, vol. 42, pp. 393–405, 1990.
[11] D. Dey, B. Chatterjee, S. Dalai, S. Munshi, and S. Chakravorti, "A deep learning framework using convolution neural network for classification of impulse fault patterns in transformers with increased accuracy," *IEEE Trans. Dielectr. Electr. Insul.*, vol. 24, no. 6, pp. 3894–3897, Dec. 2017.
[12] Y. Yu *et al.*, "FTDC: A fault-tolerant server-centric data center network," in *Proc. IEEE/ACM 24th Int. Symp. Quality Service*, 2016, pp. 1–2.
[13] H. Yang, J. Yuan, H. Yao, Q. Yao, A. Yu, and J. Zhang, "Blockchain-based hierarchical trust networking for JointCloud," *IEEE Internet Things J.*, 2019. to be published, doi: 10.1109/JIOT.2019.2961187.
[14] H. Emesowum *et al.*, "Achieving a fault tolerant and reliable cloud data center network," in *Proc. IEEE Int. Conf. Services Comput.*, 2018, pp. 201–208.
[15] M. Ruiz *et al.*, "Service-triggered failure identification /localization through monitoring of multiple parameters," in *Proc. 42nd Eur. Conf. Opt. Commun.*, Sep. 2016, pp. 1–3.
[16] A. Yu *et al.*, "Accurate fault location using deep belief network for optical fronthaul networks in 5G and beyond," *IEEE Access*, 7, no. 1, pp. 77932–77943, 2019.
[17] E. Ruiz M *et al.*, "Hierarchical text categorization using neural networks," *Inf. Retrieval*, vol. 5, no. 1, pp. 87–118, Feb. 2002.
[18] D. Rafique *et al.*, "Cognitive assurance architecture for optical network fault management," *J. Lightw. Technol.*, vol. 36, no. 7, pp. 1443–1450, Apr. 1, 2018.
[19] M. Majidi, M. Etezadi-Amoli, and M. S. Fadali, "A novel method for single and simultaneous fault location in distribution networks," *IEEE Trans. Power Syst.*, vol. 30, no. 6, pp. 3368–3376, Nov. 2015.
[20] A. Choromanska *et al.*, "The loss surfaces of multilayer networks," *J. Mach. Learn. Res.*, vol. 38, pp. 192–204, 2015.
[21] Q. Yao, H. Yang, A. Yu, J. Zhang, "Transductive transfer learning-based spectrum optimization for resource reservation in seven-core elastic optical networks," *IEEE/OSA J. Lightw. Technol.*, vol. 37, no. 16, pp. 4164–4172, Aug. 2019.
[22] U. Shaham *et al.*, "Provable approximation properties for deep neural networks," *Appl. Comput. Harmon. Anal.*, vol. 44, pp. 537–557, 2018.
[23] M. Raghu *et al.*, "On the expressive power of deep neural networks," in *Proc. 34th Int. Conf. Mach. Learn.*, vol. 70, pp. 2847–2854, 2017.
[24] F. Paolucci *et al.*, "Demonstration of gRPC telemetry for soft failure detection in elastic optical networks," in *Proc. Eur. Conf. Opt. Commun.*, 2017, pp. 1–3.

[25] X. Zhao *et al.*, "Accurate fault location based on deep neural evolution network in optical networks for 5G and beyond," in *Proc. Opt. Fiber Commun. Conf.*, 2019, pp. 1–3.
[26] S. Zhang *et al.*, "Precise failure location and protection mechanism in long-reach passive optical network," *J. Lightw. Technol.*, vol. 34, pp. 5175–5182, 2016.
[27] P. T. Endo *et al.*, "Minimizing and managing cloud failures," *Computer*, vol. 50, no. 11, pp. 86–90, Nov. 2017.
[28] H. Yang, Q. Yao, A. Yu, Y. Lee, and J. Zhang, "Resource assignment based on dynamic fuzzy clustering in elastic optical networks with multi-core fibers," *IEEE Trans. Commun.*, vol. 67, no. 5, pp. 3457–3469, May 2019.
[29] M. Yuang *et al.*, "OPTUNS: Optical edge datacenter network architecture and prototype testbed for supporting 5G," in *Proc. Opt. Fiber Commun. Conf. Exhib.*, 2019, pp. 1–3.
[30] A. Yu *et al.*, "Long-term traffic scheduling based on stacked bidirectional recurrent neural networks in inter-datacenter optical networks," *IEEE Access*, vol. 7, no. 1, pp. 182296–182308, 2019.
[31] H. Yang, B. Wang, Q. Yao, A. Yu, and J. Zhang, "Efficient hybrid multi-faults location based on hopfield neural network in 5G coexisting radio and optical wireless networks," *IEEE Trans. Cogn. Commun. Netw.*, vol. 5, no. 4, pp. 1218–1228, Dec. 2019.
[32] Y. Zhi *et al.*, "Single nanoparticle detection using optical microcavities," *Adv. Materials*, vol. 29, no. 12, 2017, Art. no. 1604920.
[33] A. Rawal and R. Miikkulainen, "From nodes to networks: Evolving recurrent neural networks," 2018, *arXiv:1803.04439*.
[34] K. O. Stanley *et al.*, "Efficient evolution of neural network topologies," in *Proc. Congr. Evol. Comput.*, vol. 2, pp. 1757–1762, 2002.
[35] E. David *et al.*, "Genetic algorithms for evolving deep neural networks," in *Proc. ACM Genetic Evol. Comput. Conf.*, 2014, pp. 1451–1452.
[36] P. Kathiravelu *et al.*, "Software-defined networking-based enhancements to data quality and QoS in multi-tenanted data center clouds," in *Proc. IEEE Int. Conf. Cloud Eng. Workshop*, 2016, pp. 201–203.

Hui Yang (Senior Member, IEEE) is currently a vice dean and an associate professor with the Beijing University of Posts and Telecommunications (BUPT). His research interests include SDN, 6G, fixed-mobile access networks, data center network, optical networks, blockchain etc. He has authored or coauthored 100 papers in prestigious journals and conferences, and is the first author of more than 50 of them. He have served as the editor of *IEEE Journal on Selected Areas in Communications* and *ATE of IEEE ComMag* and general chair of ISAI'16. He has received the Best Paper Award at IEEE IWCMC'19/NCCA'15 and Young Scientist Award in IEEE ICOCN'17.

Xudong Zhao received the BS degree from Xidian University, Shaanxi, China, in 2017. He is working toward the master's degree in information and communication engineering in the Beijing University of Posts and Telecommunications (BUPT). His main research interests include optical network survivability, optical access networks, software defined optical networks, and radio over fiber.

Qiuyan Yao received the MS degree in computer science and technology from the Hebei University of Engineering, Handan, China, in 2015. She is currently working toward the PhD degree in the Beijing University of Posts and Telecommunications (BUPT), China. Her research interests include routing and spectrum assignment in elastic optical networks and space division multiplexing networks.

Ao Yu received the BS degree from the China University of Petroleum (UPC), Shandong, China, in 2014. He is working toward the PhD degree in information and communication engineering in the Beijing University of Posts and Telecommunications. His research interests include cross stratum allocation, data center networks, software defined networks, and communication security.

Jie Zhang is currently a professor and dean of the Institute of Information Photonics and Optical Communications at BUPT. He is sponsored by more than 10 projects of the Chinese government. He has published eight books and more than 100 articles. Seven patents have also been granted. He has served as a TPC member for ACP 2009, ONDM 2010 and so on. His research interests include optical transport networks, packet transport networks and so on.

Yuefeng Ji is currently a professor with the Beijing University of Posts and Telecommunications. He is the executive deputy director of the State Key Laboratory of Information Photonics and Optical Communications, a member of the Discipline Appraisal Group of the Academic Degree Commission of the State Council, and a recipient of the National Science Fund for Distinguished Young Scholars.