Advances in Science, Technology and Engineering Systems Journal Vol. 2, No. 3, 617-622 (2017)

www.astesj.com

Special Issue on Recent Advances in Engineering Systems

ASTESJ

ISSN: 2415-6698

Optimisation of Software-Defined Networks Performance Using a Hybrid Intelligent System

Ann Sabih*, Yousif Al-Dunainawi, H. S. Al-Raweshidy, Maysam F. Abbod

Electronic and Computer Engineering Department, College of Engineering, Design and Physical Sciences Brunel University London, United Kingdom

A R T I C L E I N F O

Article history:
Received: 06 April, 2017
Accepted: 11 May, 2017
Online: 04 June, 2017

Keywords:
ANN
Evolutionary Optimisation
SDN

A B S T R A C T

This paper proposes a novel intelligent technique that has been designed to optimise the performance of Software Defined Networks (SDN). The proposed hybrid intelligent system has employed integration of intelligence-based optimisation approaches with the artificial neural network. These heuristic optimisation methods include Genetic Algorithms (GA) and Particle Swarm Optimisation (PSO). These methods were utilised separately in order to select the best inputs to maximise SDN performance. In order to identify SDN behaviour, the neural network model is trained and applied. The maximal optimisation approach has been identified using an analytical approach that considered SDN performance and the computational time as objective functions. Initially, the general model of the neural network was tested with unseen data before implementing the model using GA and PSO to determine the optimal performance of SDN. The results showed that the SDN represented by Artificial Neural Network ANN, and optmised by PSO, generated a better configuration with regards to computational efficiency and performance index.

1. Introduction

Over the past few years, upon entering the era of 'big data, there has been a change in the traditional internet traffic to a more complex traffic form. This complexity has required the increased flexibility and scalability of the modern data centre. In addition to the utilisation of numerous device types within the same area and an increase of advanced network applications, it is now possible for a multitude of end-point devices to share and exchange varying network traffic patterns. As a result, there is a need for a change in the infrastructure of the current network in order to modernise it in line with these differing network traffic patterns. As such, a new approach has been proposed [1].

In the recent years, hybridisation or combination of different machine learning and adaptation techniques has been employed for a large number of new intelligent system designs. The main aim of integrating these techniques is to overcome individual limitations and to achieve synergetic effects [2]. These techniques including, Neural Networks (NNs), the Adaptive Network Fuzzy Inferences System (ANFIS) are used for mapping and modelling purposes. Whilst evolutionary based optimisation approaches, such as

the Genetic Algorithm (GA) and Particle Swarm Optimisation (PSO) have been applied widely to produce powerful and optimised intelligent systems [3].

In many modern systems, artificial intelligence methods, such as intelligent transportation, have taken on notable roles. This adoption of new technology has encouraged the consideration of improvements in the conventional computer networks. The SDN paradigm's abstraction concept and AI techniques have a complex relationship that can be utilised to develop network elements with adaptive behaviours that can also introduce contemporary mechanisms that can overcome the common network issues as well as new issues related to SDN [1].

*Corresponding Author: Ann Sabih, Electronic and Computer Engineering Department, College of Engineering, Design and Physical Sciences Brunel University London, United Kingdom, Email: Ann.Sabeeh@brunel.ac.uk

Recently, with the fast development of the Internet, network topology, and number of applications have been changed gradually, which has led to be more complicated in structures and functions. A network based on the traditional TCP/IP architecture faces many challenges, especially the router, as the network core, takes on too many efforts to deal with. As a result, the validity and efficiency of data forwarding is being threatened. Hence, we need to find a new kind of network architecture to solve the existing problems. To this end, the study of future networks is proceeding all over the world, and the Software Defined Network (SDN) is one of them [4]. SDN is an architecture enabling rapid innovation, while hiding much of the complexity of the networking design. As such, it is a promising solution for network virtualisation that decouples control from the forwarding or data plane [5]. In doing so, it can provide the capability of remote and centralised control of the network forwarding-plane through the network control-plane.

SDN, as a network platform, has been studied widely in the literature and many researchers have proposed soft computing methods to model and optimise the network. Yilan and et al. [6] provided a genetic algorithm for solving the bandwidth-constrained multi-path optimisation problem in the SDN. Gao and et al. [7] contributed a Particle Swarm Optimisation algorithm to solve the control placement problem that takes into consideration both the latency between controllers and their capacities. Zhang and Fumin [8] explored applications of the SDN technique in the direction of the automation of network management, the unified control of optical transmission and IP bearing, smooth switching in a wireless network as well as network virtualisation and QoS assurance. Risdianto and et al. [5] evaluated the performance of an SDN-based virtual network on different virtualisation environments, including operating system based virtualisation, hardware-assisted virtualisation, and par virtualisation. Sgambelluri and et al. [9] proposed novel mechanisms that have been specifically introduced to maintain working and backup flows at different priorities and to guarantee effective network resource utilisation when a failed link is recovered. Ionita and Victor-Valeriu identified a method of avoiding DDoS attacks in the SDN environment. These scholars utilised a cyber defence system to assess risk that was structured around the neural network and the biological danger theory. This demo platform is able to achieve full packet capture in the SDN in addition to mitigating any attacks, providing it is considered necessary by the central command component. The benefit of these packet captures is that they can be utilised for forensic analysis to identify the attacker [10].

A novel open flow controller, which was structured around the intracerebral neural network was suggested by Wu and Huang in order to generate a independent media handover with wireless operation. As such, binary decisions are made by the controller that are determined by the link control parameters. These are instantly generated and gathered from the trained neural network, which is reliant on the interactions between mobile speed and the wireless link performance parameters. Using a mutated PSO, these authors managed to train the fundamental controller equation for the media independent handover that is interspersed with the intracerebral neural network's sigmoid and radial activation functions [11].

A hybrid intelligent system is proposed in this paper to optimise the performance (i.e. throughput, delay) of the SDN. The proposed system includes ANN to model inputs (flows of the flow tables)-outputs (throughput and delay) of the network. After, the evolutionary algorithms have been employed to find the optimal set of inputs that maximise this performance. This method grants to lessen the burden of SDN switch by making the network controller adaptively providing the rules/flows to the switches instead of making the switch suspends for the controller events every period of time.

2. Software-Defined Networks

It was previously noted that a software-defined network (SDN), could be centrally organised around the principle of the compartmentalisation of the control plane and data plane. The SDN is a budding architecture that is promoted by the ONF due to its ability in forwarding functions and network control decoupling. This decoupling allows the direct programming of the network control and the abstraction of the underlying infrastructure for applications and network services. The networking devices have their intelligence removed in this architecture and positioned within a centralised controller, which manages the functionality of the entire network. Software based SDN controllers also have this centralised network intelligence allowing them to maintain a global view of the network. This allows the infrastructure devices to take on a forwarding role that is able to process incoming packets. This role is determined by a set of rules generated instantaneously by the controller within the control layer that is based on some predefined program logic. A remote commodity server is a usual manner in which these controllers are run. A secure connection allows communication with the forwarding elements utilising a standardised command set. A high-level architecture for SDN is presented by Open Networking Foundation ONF [12], that has three main layers, application, control and infrastructure layers that are vertically split as detailed in Figure 1

As demonstrated by [9], the SDN controller serves to maintain the stream tables so as to engage in the management of every stream that is conveyed through the system. Application layer service requests are mapped by the SDN control layer into defined commands and directives to data plane switches. The SDN control layer then supplies applications with data plane topology and activity information. As such, the control layer acts as either a server or a collaborating set of servers, which are recognised as SDN controllers.

The OpenFlow procedure can be regarded as an open SDN specification that is constituted of a pair of foundational elements: first, the OpenFlow switch and, second, the controller. The former facilitates the execution of the data plane while the latter realises the control plane. It should be noted that the OpenFlow procedure as a totality is employed by the interchange that takes place with regard to the controller and the switches via a safe channel. The OpenFlow switch and controller communicate via a safe channel that is realised in the context of a distinctive practice and, according to [10], this is also referred to as OpenFlow. The controller transmits a Flow Table towards the OpenFlow switches and each stream passage in the former element is constituted of a regulation crafted from an organisation of fields, thereby coordinating the approaching bundles. It is notable that this is an activity that characterises the issue of how the coordinating parcels can be prepared, for example, by transmitting on a specific yield port.

In addition, a number of counters are employed in order to collect information relating to the stream. In a similar way, each passage can be linked to alternative information, one case of this being the requirement level and the hard and idle timeout of the pair of timing devices.

The bundle is captured and conveyed to the controller that employs the safe channel in those instances where a switch receives a parcel that is not engaging in the coordination of the passages that have been introduced. Having captured the bundle, the activity of the OpenFlow switch is regulated by the controller as a result of the overhaul of its Flow Tables, thereby transmitting the coordinating parcels to the end point [8]. One implication of this is that the relative casings are not required to partake in the controller another time as a result of the potential initiation of the Flow Tables' reformulation [3].

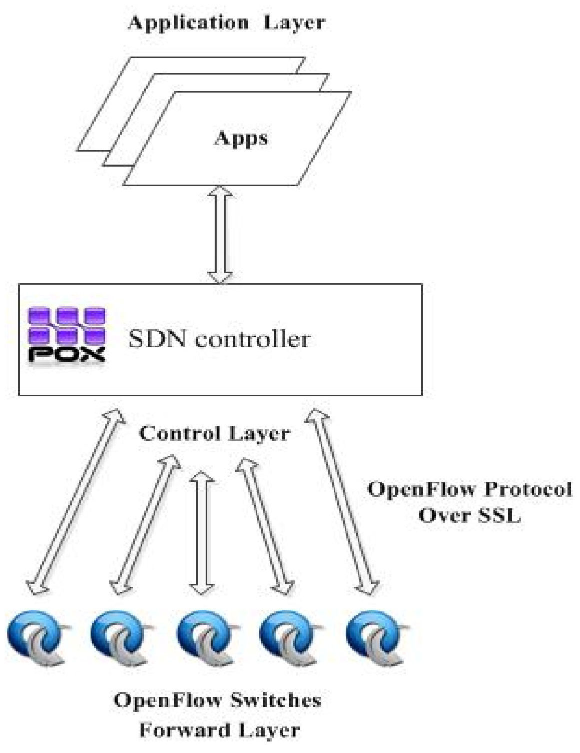

Figure 1. The software-defined network's architecture

3. Artificial Intelligence

3.1. Neural Networks

A branch of artificial intelligence, disseminated at a considerable pace in recent times owing to its suitability with regard to the modelling and forecasting ability it has in relation to dynamic systems, is the artificial neural network (ANN). In light of their promising potential, ANNs have emerged as a central branch of research into artificial intelligence. The registration of the input-output relationships of nonlinear and synthesis systems have been identified as one of the key advantages offered by ANNs and, notably, this relationship can be straightforwardly, rapidly, and cost- and time-effectively discerned by lowering the error with regard to the network output(s) and the actual output(s). A defining feature is that, following the appropriate preparation of the network, outputs can be estimated within a very short space of time (namely, in a matter of seconds). Frameworks that centre on artificial neural networks are currently seeing effective application in a range of areas – a few examples being including adaptive control, laser applications, nonlinear system identification, robotics, image and signal processing, medical areas, pattern recognition, error detection, process logging, and renewable and sustainable energy areas – in order to surmount obstacles faced by engineers [11]. The only appropriate connections in a feedforward neural network structure are between the outputs and inputs of each layer. As such, there are no connections between the outputs of one layer and the inputs of the same or previous layers [13].

3.2. Evolutionary Algorithms

Taking inspiration and motivation from natural processes and, in addition, basing the developments on factors relating to iterative and probabilistic processes, a range of evolutionary algorithms – including genetic algorithms (GAs), particle swarm optimisation (PSO), and simulated annealing – have been formulated in recent years. The primary purpose of such developments is for application in optimisation issues. Two multi-purpose and frequently employed algorithms include GA and PSO, and these are utilised in every domain [12].

- **Genetic algorithm**: Formulated by Holland (1975), GAs are self-modifying global optimisation probability search algorithms, the fundamental concept of which was inspired by the genetic mechanisms that form the basis of the theory of Darwinian natural selection and biological evolution. GAs operate by simulating the biological processes that are observed in the natural world as driving the phenomena of genetic and evolutionary development; according to the concept of natural selection, GAs provide solutions to deep problems by employing code technique and reproduction processes [13]. GAs has been extensively employed in a variety of domains with considerable efficacy in recent years, and this is primarily attributed to their almost universal relevance and promising results. The difference between the more traditional search algorithms and GAs[4] is that the latter have numerous candidate solutions rather than just one partial or candidate solution. With GAs, each problem's candidate solution is portrayed by a data structure that is termed an 'individual'. There are two parts to each individual; these are the chromosome and the fitness. An individual's chromosome is constructed from genes with each value assigned to the gene being referred to as alleles. These individuals combine to form a population, the size of which remains constant for the duration of the search for most GAs. Out of the current population, a number of 'parent' individuals are selected, based on their fitness, which are allowed to create offspring. Individuals with above average fitness have a higher than average chance of selection for parenting. Following selection, the parents are subjected to a number of reproductive operators, such as crossover and mutation. Those subjected to crossover have a copy of their genes taken to create an offspring's chromosome. This is comparable to the creation of living organisms that are created from a genetic mixture of both parents from sexual reproduction. However, only one parent is required for mutation. In this manner, the offspring is often an almost exact replica of the parent but with a few altered genes. Following the generation of the offspring, their represented candidate solutions can be evaluated, and the offspring's fitness is determined. As the population size remains largely static, before the offspring can be incorporated into the population, it is necessary for some individuals in the current population to be removed, or die. Removal of individuals is often decided from their fitness with those individuals with a below average fitness being more likely to be selected for removal than those with an average or above average fitness. Again, this is reminiscent of the evolution of living organisms and is termed natural selection. As such, those individuals that display better fitness are allowed to procreate and live longer. Interestingly, this process of fitness selection means that the original population does not need to be very good. Indeed, it is often the case that each individual in the initial population represents a randomly generated candidate solution; however, the repetitive application of natural selection and reproduction allows GAs to generate rapid and efficient solutions [13].

- **Particle Swarm Optimisation**: First developed by Kennedy and Eberhart (1995) [14] and built on by the researchers several years later (Ibid, 2001) [15], PSO algorithms have been applied with enormous success in optimising a broad range of applications [16]. PSO operates by locating all individuals and particles (usually in the range of 10-100) in randomised positions and, following this, intending that each particle engages in random motion in a determined direction in the search space. Following this, the direction of each particle is incrementally modified in order to proceed according to the optimal previous positions, thereby identifying more favourable positions on the basis of specifications or an objective function (i.e. fitness). The original particle speed and location are chosen randomly and, in turn, the velocity formula presented below is used to provide updates:

$$V_{c_i+1} = wV_i + C_1R_1 \times (Pb_i - x_i) + C_2R_2 \times (Gb - x_i) \quad (1)$$

Contrastingly, the new particle is computed by summing the new velocity to that which precedes it, as presented below:

$$x_{i+1} = x_i + V_{c_i+1} \quad (2)$$

Where: Vc denotes the particle's velocity; X denotes the particle's position; R1 and R2 are independent random variables uniformly distributed in [0, 1]; C1 and C2 are the acceleration coefficients, and w represents the inertia weight. The particle's new velocity can be calculated by employing Eq. 13, and the information required includes the previous velocity, the distance of the particle's present position from its optimal position (Pb), and the global best position (GB). Following this, on the basis of Eq. 14, the particle is conveyed to a new location in the search space and, notably, the way in which every particle performs is evaluated in relation to a predetermined objective function known as the performance index.

4. Equation Simulation and Results

4.1. SDN Simulation

SDN simulation has been performed using the Mininet platform which is the common emulation for SDN which is used by researchers to collect all datasets of inputs and outputs. Mininet has the ability to imitate various types of system components, for example, have, layer-2 switches, layer-3 switches, and interfaces. In addition, simulation experiments were carried out using POX controller with OpenFlow 1.0 and monitoring of the flows was required for all events. Regarding which, in order to build the learning system, the SDN controller gives the orders to SDN switch to monitor the flows, and the switch keeps on monitoring them to detect the events. When the flows are monitored, the entire events whether they occur only frequently or periodically are stored in the database to be used in the ANN learning system. The flows/inputs include the rules coming from the controller. In turn, the output will be represented by the system throughput and the network delay. Consequently, the data which are collected from the ANN learning system is considered as being an efficient input to the optimisation algorithms. This paper presents a new method to minimize the load of the SDN switch by making it changing adaptively by the controller, instead of wait for the event all the time.

4.2. SDN Identification

When the ANN training started, the dataset had been preprocessed by normalising them into the range between -1 and 1. The dataset of the inputs and outputs was divided randomly into three subsets: a training set, valediction set and testing set. The first subset was for establishing the gradient as well as updating the network weights and biases. The error regarding the second subset was observed during the training development. The validation error is usually reduced in the initial training phase, as is the training set error. Nevertheless, when the network overfits the data, the error in the validation set invariably starts to rise. In the current case, the network parameters were saved at the minimum of the validation set error.

Input values need to be normalised in the range [-1, 1], which corresponds to the minimum and maximum actual values. Subsequently, testing the ANN requires a new independent set (test sets) in order to validate the generalisation capacity of the prediction model. A multilayer feedforward network was implemented to estimate the performance of the SDN. In order to obtain a maximum accuracy of prediction, the network was trained in different topologies. For each network architecture, the training was run ten times for various random initial weights and biases using the Levenberg Marquardt algorithm (LMA). After investigating the performance of different architectures using the exhaustive search method, the best trained ANN with one hidden layer was found to consist of 17 neurons in this hidden layer, which gives the comparably better performance of MSE, with 2.488 ×10-8, see Figure.. 2. While, Figure 3 shows the performance of the network as a mean square error (MSE) versus the network architecture for the single hidden layer. Also, Figures 4 and 5 show the simulated and predicted SDN performance for both the training and test sets. It is noticed that the ANN model is efficiently accurate and the network is accepted as a general model to be integrated, as the next step, with GA or PSO so as to produce the proposed intelligent hybrid system.

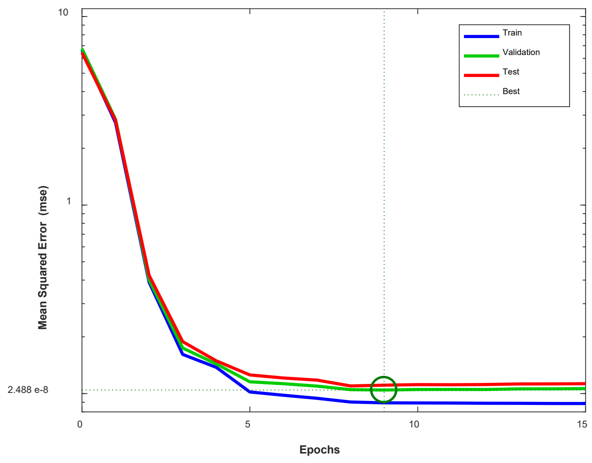

Figure 2. the progress of the ANN performance.

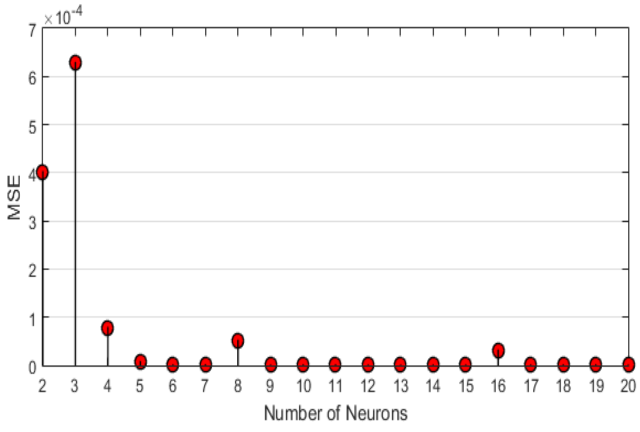

Figure 3 Performance of a single hidden layer ANN

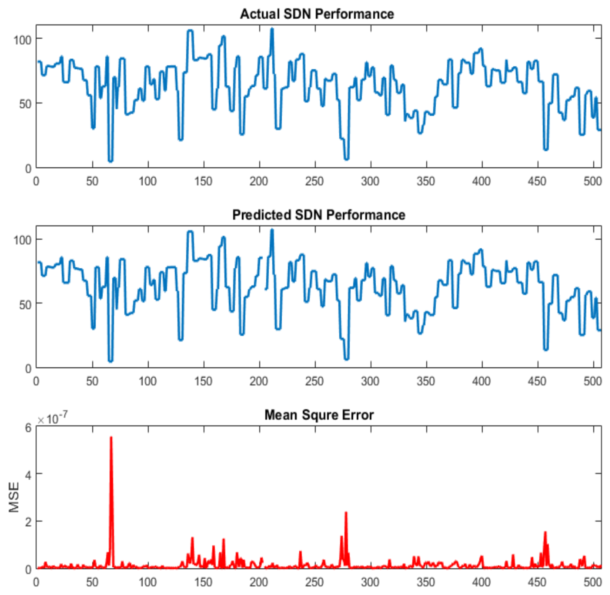

Figure 4 Comparing predicted with actual SDN performance for the training sets

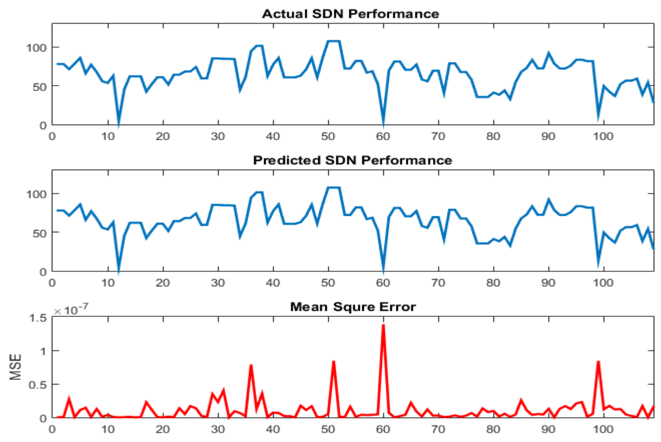

Figure 5 Comparing predicted with actual SDN performance for the testing sets

4.3. SDN Optimisation

GA and PSO were integrated separately with the trained ANN model to select the optimal set of inputs that make the network work as efficient as possible. Figure 6 shows the architecture of the system including the trained general ANN model as well as PSO and GA as an optimizer.

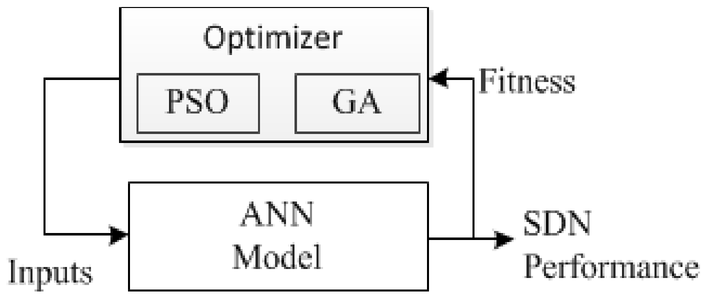

Figure 6 The architecture of the proposed method

TABLE I. PERFORMANCE AND COMPUTATION TIME COMPARISONS FOR GA AND PSO

Optimisation method	SDN Performance %	Computation Time (min)
PSO	96.321	3.23
GA	94.846	6.75

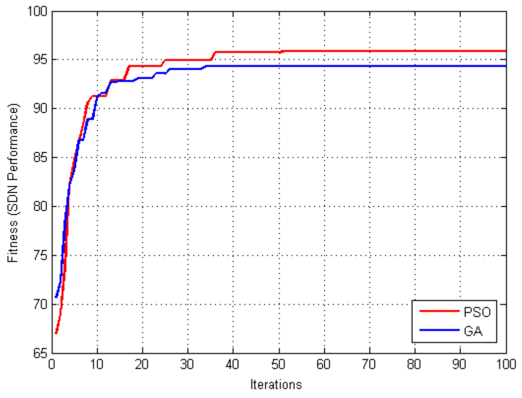

Figure 7 Convergence comparison of GA and PSO

The simulation experiments were carried out by MATLAB platform. PSO was employed to find the optimal structure of the network, and the best operational parameters of this and the GA algorithm were chosen after extensive simulations, which were set as follows:

- Size of the population or swarm: 50
- Maximum iterations or generations (max) :100
- Cognitive acceleration (C1): 1.2
- Social acceleration (C2): 0.12
- Momentum or inertia (w): 0.9

A comparison of the results of the SDN performance is provided in Table 1 and Fig. 7. PSO has outperformed GA regarding the performance, and computational time, the convergence is faster with fewer iterations, and the obtained fitness is higher. However, the results provided were obtained after running PSO/GA 20 times.

5. Conclusion

This paper has presented a novel hybrid intelligent system for the modelling and perfecting of the Software Defined Network (SDN). This involved the training of an artificial neural network (ANN), which had a single layer in the hidden zone, to map the inputs and network performance. An acceptable MSE was shown by the network that was demonstrated as being below $2.466\text{x}10^{-8}$. The application of unseen data as a test set to the trained ANN model proved its generality; this ANN model was then coordinated with evolutionary algorithms (EAs) to create the presented intelligent hybrid system. In order to optimise the EA, PSO and GA were utilised in order to identify the optimal input set for the SDN. This optimisation was based on the fitness function of the individuals. The comparison results showed that PSO was a more effective option in terms of computational time, convergence and performance.

References

[1] M. Latah and L. Toker, "Application of Artificial Intelligence to Software Defined Networking: A Survey," Indian J. Sci. Technol., vol. 9, no. 44, Nov. 2016.

[2] Y. Al-Dunainawi and M. F. Abbod, "Hybrid Intelligent Approach for Predicting Product Composition of Distillation column," Int. J. Adv. Res. Artif. Intell., vol. 5, no. 4, pp. 28–34, 2016.

[3] D. Ruan and P. Wang, "Intelligent hybrid systems: fuzzy logic, neural networks, and genetic algorithms". Springer, 1997.

[4] W. Xia, Y. Wen, C. H. Foh, D. Niyato, and H. Xie, "A Survey on Software-Defined Networking," IEEE Commun. Surv. Tutorials, vol. 17, no. 1, pp. 27–51, 2015.

[5] A. C. Risdianto and E. Mulyana, "Implementation and analysis of control and forwarding plane for SDN," in 2012 7th International Conference on Telecommunication Systems, Services, and Applications (TSSA), 2012, pp. 227–231.

[6] Y. Yilan Liu, Y. Yun Pan, M. Muxi Yang, W. Wenqing Wang, C. Chi Fang, and R. Ruijuan Jiang, "The multi-path routing problem in the Software Defined Network," in 2015 11th International Conference on Natural Computation (ICNC), 2015, pp. 250–254.

[7] C. Gao, H. Wang, F. Zhu, L. Zhai, and S. Yi, "A Particle Swarm Optimization Algorithm for Controller Placement Problem in Software Defined Network," Springer International Publishing, 2015, pp. 44–54.

[8] Z. F. Zhang Shunmiao, "Survey on software defined network research," Appl. Res. Comput., vol. 30, pp. 2246–2251, 2013.

[9] A. Sgambelluri, A. Giorgetti, F. Cugini, F. Paolucci, and P. Castoldi, "OpenFlow-Based Segment Protection in Ethernet Networks," J. Opt. Commun. Netw., vol. 5, no. 9, p. 1066, Sep. 2013.

[10] I. Mihai-Gabriel and P. Victor-Valeriu, "Achieving DDoS resiliency in a software defined network by intelligent risk assessment based on neural networks and danger theory," in 2014 IEEE 15th International Symposium on Computational Intelligence and Informatics (CINTI), 2014, pp. 319–324.

[11] Q. Wu, J. Huang, and O. Yang, "An Open Flow Controller Based on the Intercerebral Neural Network for Media Independent Handover," Int J Swarm Intel Evol Comput, 2014.

[12] D. Kreutz, F. M. V. Ramos, P. Esteves Verissimo, C. Esteve Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-Defined Networking: A Comprehensive Survey," Proc. IEEE, vol. 103, no. 1, pp. 14–76, Jan. 2015.

[13] M. Jamshidi and A. Zilouchian, "Intelligent control systems using soft computing methodologies". CRC Press, 2001.