Princess Sumaya University for Technology
جامعة الأميرة سميّة للتكنولوجيا

صندوق دعم البحث العلمي والابتكار
Scientific Research and Innovation Support Fund

Jordanian Journal of Computers and Information Technology

June 2023 | VOLUME 09 | NUMBER 02 | ISSN 2415 - 1076 (Online) ISSN 2413 - 9351 (Print)

JJCIT

www.jjcit.org

jjcit@psut.edu.jo

An International Peer-Reviewed Scientific Journal Financed by the Scientific Research and Innovation Support Fund

Jordanian Journal of Computers and Information Technology (JJCIT)

The Jordanian Journal of Computers and Information Technology (JJCIT) is an international journal that publishes original, high-quality and cutting edge research papers on all aspects and technologies in ICT fields.

JJCIT is hosted and published by Princess Sumaya University for Technology (PSUT) and supported by the Scientific Research Support Fund in Jordan. Researchers have the right to read, print, distribute, search, download, copy or link to the full text of articles. JJCIT permits reproduction as long as the source is acknowledged.

AIMS AND SCOPE

The JJCIT aims to publish the most current developments in the form of original articles as well as review articles in all areas of Telecommunications, Computer Engineering and Information Technology and make them available to researchers worldwide. The JJCIT focuses on topics including, but not limited to: Computer Engineering & Communication Networks, Computer Science & Information Systems and Information Technology and Applications.

INDEXING

JJCIT is indexed in:

EDITORIAL BOARD SUPPORT TEAM

LANGUAGE EDITOR

Haydar Al-Momani

EDITORIAL BOARD SECRETARY

Eyad Al-Kouz

JJCIT ADDRESS

WEBSITE: www.jjcit.org
EMAIL: jjcit@psut.edu.jo
ADDRESS: Princess Sumaya University for Technology, Khalil Saket Street, Al-Jubaiha
B.O. BOX: 1438 Amman 11941 Jordan
TELEPHONE: +962-6-5359949
FAX: +962-6-7295534

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 09, No. 02, June 2023.

OPTIMAL ENERGY CONSUMPTION AND COST PERFORMANCE SOLUTION WITH DELAY CONSTRAINTS ON FOG COMPUTING

Zahra Mahmoudi, Elham Darbanian and Mohsen Nickray

(Received: 8-Dec.-2022, Revised: 9-Feb.-2023, Accepted: 24-Feb.-2023)

ABSTRACT

Cloud computing plays an essential role in the development of the Internet of Things, which provides-data processing and storage services. Fog computing, the evolution of cloud computing helps provide solutions to cloud-computing challenges, such as latency, location awareness and real-time mobility support. Fog computing fills the gap between the cloud and IoT devices within the close vicinity of IoT devices. So, computation, networking, storage, data management and decision-making occur along the path between the cloud and the IoT devices. The automatic and intelligent management of fog node resources and achieving an effective scheduling policy in the computing model are necessary requirements and will lead to the improvement of the overall performance of fog computing. Some optimization problems are modeled by mixed-integer nonlinear programming (MINLP). In this paper, a model; i.e., an MINLP optimization problem on fog computing, is designed. Our model has two goals: to increase cost performance as well as to reduce energy consumption. Cost performance is the price that users are charged as benefit/revenue. In other words, cost performance is defined as the ratio of the average data rate of each user to its cost. Then, the exact mathematical method with the GAMS program was used to prove its logical process. In the next step, we solved the model with genetic algorithm (GA), particle swarm optimization (PSO), simulated annealing+GA (SA+GA), teaching–learning-based optimization (TLBO), grey wolf optimizer (GWO), grasshopper optimization algorithm (GOA) and random method. According to the TOPSIS comparison, the SA+GA method with a value of 0.23 is the best one compared to other methods. Then come GWO, GA, TLBO, PSO and GOA methods, respectively.

KEYWORDS

Fog computing, Optimization, Mixed-integer nonlinear programming, MINLP, Energy consumption, Cost performance.

1. INTRODUCTION

According to the analysis carried out by Cisco by 2023, Internet of Things devices will make up about 50% of the devices in networks around the world. The Internet of Things (IoT) is one of the most influential technologies in the world. IoT deals with large amounts of data that are not easy to process and store. Cloud computing plays a significant role in the development of the Internet of Things, which provides data-processing and storage services. However, many of its applications encounter with cloud computing challenges, including latency, location awareness and real-time mobility support. Fog computing, that almost looks like the evolution of cloud computing, contributes to providing solutions to these challenges [1]. The growth of devices and subsequently the data of the Internet of Things is such that it is possible to exceed the capacity of information and this shows the necessity of using the models in the fog-computing technology infrastructure to process the data and as a complement to the cloud-computing model. On the other hand, fog nodes are the main entities of the fog-computing model and for this reason, the effective management of the resources of these nodes is of particular importance. Automation of operations in computer networks through the use of innovations can lead to increased productivity, reduced operational cost and better quality of service delivery. Therefore, the automatic and intelligent management of fog-node resources and achieving an effective scheduling policy in the computing model are necessary requirements and will lead to the improvement of the overall performance of fog computing. Fog computing fills the gap between the cloud and the IoT devices within the close vicinity of the IoT devices. So, computation, networking, storage, data management and decision-making occur along the path between the cloud and the IoT devices [2].

There is a topic called "optimization", which is related to maximization and minimization. The goal of

Z. Mahmoudi, E. Darbanian and M. Nickray are with Department of Computer Engineering and Information Technology, University of Qom, Qom, Iran. Emails: z.mahmoudi@stu.qom.ac.ir, e.darbanian@stu.qom.ac.ir and m.nickray@qom.ac.ir

optimization is finding the assignment of variables, maximizing or minimizing the value of a given function [3]. When the quantity to be optimized is expressed by just one objective function, it is deemed a uni-objective or single-objective problem. Otherwise, it is a multi-objective problem that must be optimized simultaneously [4].

It is a common misconception that most design or problem-solving activities should optimize a single goal; for example, maximizing profit or creating the lowest cost, even if there are several conflicting goals for optimization. However, the relationship between goals is usually complex and depends on available options. In addition, different goals are usually incomparable; therefore, combining them into one combined goal is challenging. Many decision-making and planning problems involve several conflicting goals that have to be studied simultaneously. Such problems are generally known as multi-criteria decision-making (MCDM) problems. Depending on the characteristics of the problem [5], MCDM can classify problems in many ways. Two main classes of MCDM are generally introduced: multiple-objective decision making (MODM) and multiple-attribute decision making (MADM) [6].

Many engineering and scientific optimization problems involve combinatorial and nonlinear relations. Some optimization problems are modeled by mixed-integer nonlinear programming (MINLP) that combines the capabilities of linear programming (LP) and nonlinear programming (NLP) [7].

Initially, the driving agent behind the development of the general algebraic modeling system (GAMS) was that mathematical-programming users are believing in optimization as a strong and subtle framework to solve real-life problems in science and engineering [8]. GAMS is a high-level modeling language for formulating models. Being composed of short algebraic statements, it is easily read by modelers and it formally looks like the generally-used programming languages [9].

Particle swarm optimization (PSO) is a population-based algorithm in which the movement of a flock of birds are simulated to find the optimum solution [10]. Genetic algorithm (GA) is one of the population-based optimization methods [11]. Simulated annealing (SA) is an algorithm to solve large combinatorial optimization problems [12]. Teaching–learning-based optimization (TLBO) is an algorithm for optimizing mechanical design problems. This method deals with the teacher's effect on learners. TLBO is a population-based method too [13]. A model, in order to perform optimization, is grey wolf optimizer (GWO) which models the hunting technique and the social hierarchy of grey wolves mathematically [14]. Another optimization algorithm is called the grasshopper optimization algorithm (GOA) [15].

In this paper, the single-objective model is proposed for minimizing the total energy of all mobile devices and maximizing the cost performance due to their service-latency limitations. Our problem is a mixed-integer nonlinear programming (MINLP) problem. Then, the proposed model with the exact mathematical method of GAMS is solved. Also, the problem with PSO, GA, SA+GA, TLBO, GWO, GOA and random, is solved in order to find the best solution. The research hypotheses in this paper include the solvability of the problem model using the exact mathematical method (GAMS) and meta-heuristic methods, examining our problem model that has an answer at an acceptable time with a high number of iterations, examining the existence of an optimal solution that optimizes the objective function (sum of energy-consumption values being minimum and cost performance values being maximum) and finally examining the environment in which resources should be allocated. In this paper, it is assumed that the number of fog nodes and users was limited. All users in the model, request for resource and resource allocation are applied as first-in, first-out (FIFO).

User satisfaction or user experience is one of the important criteria for SPs. Service latency is a measure of user satisfaction. Also, we must first ensure that the transmission quality between users and FNs is satisfied. To measure system performance, we consider mandatory revenue and price offers from users as the benefit/revenue of the SPs. The price offered by each user is related not only to the latency requirement, but also to data size. Each SP serves more than one user and therefore receives more than one offer. Two performance measures, user satisfaction and SP revenue, are essential for good resource allocation in fog computing. CP is defined as the ratio between each user’s average data rate and its price cost, in unit of Mbps/sec/dollar. Because the actual amount of delay is strongly related to the amount of user data to transmit and process, the data rate is considered, rather than the pure delay. It also makes sense to use the users’ monetary payment/offer for the respective fog computing service that they obtain for the cost factor. The cost-performance function for each user represents the quality of service for which the user pays [16]. Due to the increasing importance of energy consumption, fog-

computing architecture is an effective solution to enable energy-efficient and low-latency mobile applications due to its low-latency and high-bandwidth connections with mobile devices as well as cloud servers, agile mobility and location-awareness support [17].

The contributions of our article are as follows:

- A computational framework (single-objective model) which considers both vertical and horizontal cooperation between fog nodes and mobile devices is proposed. Due to service-latency limitations, the total energy of all mobile devices is minimized and the cost performance is maximized at the same time. Then, the proposed model is solved with the exact mathematical method of GAMS and Baron Solver.
- A task can be offloaded by a mobile device to one of the fog nodes, the cloud, through a fog node or to the cloud server directly. The main problem among thirty problems (according to the central limit theorem (CLT)) is considered with three sizes (small, medium and large). By changing the number of mobile devices and fog nodes, problems are obtained with some different variables, which can be used to check the scalability of the problem. The purpose of this scenario is to measure the behavior of algorithms by considering the values of two objective functions and their execution times.
- Then, the problem is solved with eight methods (PSO, GA, SA+GA, TLBO, GWO, GOA, random and exact mathematical method) to find the best solution. Also, twelve other issues are considered with different numbers of mobile devices and fog nodes, in two main stages: the number of fixed fog nodes and the number of variable mobile devices and conversely (the number of fixed mobile devices and the number of variable fog nodes). Therefore, a more accurate analysis of the values of objective function is obtained with two indices of execution time and the value of the objective function.
- Extensive simulations are performed to evaluate cost performance and energy consumption and finally, the best algorithm is selected by using the technique for order of preference by similarity to ideal solution (TOPSIS) method.

The rest of this paper is arranged as follows. Related work is presented in Section 2. In Section 3, the network model is discussed. Problem solving is presented in Section 4. Evolution is presented in Section 5. Finally, in Section 6, the conclusion is presented.

2. RELATED WORKS

The related papers are presented as follows. Authors in [18] proposed a joint offloading decision and a framework for resource allocation optimization for MEC with algorithms of relaxing-optimization policy (ROP) and index branch-and-bound algorithm (IBBA). Their paper has disregarded the optimal transmission power assignment. The model only included communications. In [19], an IoT-based remote health-monitoring system implementation was presented, that included a demonstration of a smart e-health gateway called UT-GATE. Applications were user-centric, whereas services were developer-centric. Authors in [20] have studied an MCC system with multiple users, one CAP and one remote cloud server. The weighted total cost of energy, computation and maximum delay were minimized among all users. Also, a new approach to the joint task offloading and computation and communication resource allocation with share CAP, an efficient heuristic algorithm using semi-definite relaxation (SDR) and a new approach to randomization mapping were proposed by them. They assumed that there were several mobile-phone users, each with only one task. In [21], the authors have proposed min-max fairness in a mixed fog/cloud computing system by joint optimization of offloading decision-making and resource allocation using computation offloading and resource allocation (CORA) algorithms, as well as the bisection method for computation-resource allocation (BCRA) algorithm. They only considered orthogonal multiple access (OMA). Researchers in [22] have investigated a framework for optimization of computation offloading, computation resource allocation, resource block (RB) pattern assignment, transmit power allocation and a low-complexity general algorithm framework known as fireworks algorithm based on joint computation offloading and resource allocation algorithm (FAJORA) to decompose the problem into several sub-problems. If the original LTE standard were to be considered, support for these processes would be costly. The joint task assignment, communication rate and computation frequency allocation for a device-to-device (D2D)-enabled multi-helper MEC system was proposed by the authors in [23]. Also, to create a sub-optimal task assignment solution for the MINLP

formulation and a benchmark scheme with fixed computation frequency and a greedy task assignment-based heuristic algorithm, they proposed a special convex-relaxation-based algorithm. This work only considered the users' cooperative computation under fixed energy supplies (e.g., batteries). In [24], a novel low-latency and trustworthy communication-computing system design was proposed to enable mission-critical applications by which the ultra-reliable low latency communications (URLLC) requirement has been formulated. In particular, relying solely on the average queue length did not meet the strict delay requirement for vehicle applications. In [25], a holistic strategy for a joint task offloading and resource allocation in a multi-cell MEC network was investigated by the researchers. To optimize the MINLP problem, the original problem was formulated into a resource-allocation problem with a fixed task offloading decision and a task offloading problem. This work did not consider the ultra-dense network and it was difficult to gain insight into the design of critical parameters. Authors in [26] decomposed the drone placement problem into two sub-problems and improved the latency ratio of the network. They placed drone base stations to the locations with higher user densities. In a dynamic network, each server had to process dynamically changing the amount of data load gathered in different clusters, which made the load on cluster servers unbalanced. In [27], by leveraging the vertical cooperation among devices, edge nodes and cloud servers with alternating-direction method of multipliers (ADMM) method and difference of convex functions (D.C.) programming, a three-tier cooperative computing network was inspected. The ADMM and D.C. programming-based methods were only sub-optimal. Authors in [28], using fog computing with low latency, performed the electric energy control in a microgrid. Their scheme proposed some services, including the proportional integral derivative (PID) controller and algorithm scheduling, to decrease consumers' bills and algorithms (FIFO and GA) using the PID calculations. In this work, just downlink was used. In [29], general joint computation offloading and resource allocation for the multiple-input multiple-output (MIMO)-based mobile cloud computing system considering perfect- channel state information (P-CSI) and imperfect-CSI (IP-CSI) were tackled by the researchers. In order to solve the underlying MINLP, the optimal and low-complexity algorithms were proposed. This work has discussed the network and communication resources individually, typically focusing independently on each of them. Authors in [30] inspected the optimization of offloading decision, local computation capability and computing resource allocation of fog node. The problem was decomposed into two independent sub-problems by them and an HGSA-based latency-minimum offloading decision algorithm was designed to tackle this MINLP problem with low complexity. Using traditional greedy search methods was challenging. In [31], the offline placement problem of IoT services supporting horizontal and vertical scaling in an edge computing environment was investigated. The authors formulated an MINLP problem and proposed linearization and genetic-based method to solve it. This article only minimized deadline violations due to limited resources at the edge. In [32], radio resource allocation between two network slices with heterogeneous performance metrics in fog radio access networks was investigated by the researchers and the problem as a Stackelberg game was modelled, where the global radio resource manager (GRRM) with a strong position acted as the leader and the local radio resource managers (LRRMs) of slices acted as followers. This work was inefficient in long-term resource allocation performance. Researchers in [33] proposed an advanced caching technique through which the energy efficiency and delays can be improved and an algorithm for load balancing in the advanced cached fog layer. It has become more challenging to connect and monitor many devices, the most critical feature of which is content security. In [34], the joint optimization of computation offloading decisions, service caching placement and system resource allocation was studied by the authors. The complicated MINLP problem was transformed to a pure 0-1 integer linear programming (ILP) problem and reduced-complexity algorithms were proposed. It may not work properly for multi-user systems. In [35], a declarative methodology, SecFog, was proposed, which may be used for quantitative assessment of the security level of multi-service application deployment to cloud-edge infrastructures. Also, an MINLP problem of placing application services for the purpose of assuring end-to-end delay constraints was formulated. The underlying model was limited to the security controls provided by the infrastructure. In [36], the authors studied the service-placement problem in fog computing using meta-heuristic approaches and proposed an improved parallel genetic algorithm. But, it seems to be costly to implement. In [37], researchers have proposed a multi-objective strategy including execution time, energy consumption and cost, based on a biogeography-based optimization algorithm, for MEC offloading to satisfied users' multiple requirements. Objective(s), network and environment of related works are shown in Table 1.

Table 1. Related works.

Ref.	Objective(s)	Network								Environment	
		MCC	MFC	MEC	Fog/cloud system	Edge computing	Mobile edge-cloud computing	Fog computing	Cloud-edge infrastructures	Simulation	Real
Vu et al., 2018 [18]	Energy and delay			*						*	
Rahmani et al., 2018 [19]	Energy efficiency, security, overall system intelligence, reliability performance, mobility and interoperability		*	*							*
Chen et al., 2018 [20]	Cost of energy, computation and delay	*								*	
Du et al., 2017 [21]	Cost, energy and delay				*					*	
Du et al., 2018 [22]	Computation-resource allocation, optimizing computation offloading, transmitting power allocation, resource block pattern assignment				*					*	
Xing et al., 2019 [23]	Energy and delay			*						*	
Liu et al., 2019 [24]	Power consumption and latency			*						*	
Tran and Pompili, 2018 [25]	Energy consumption			*						*	
Fan and Ansari, 2018 [26]	Latency				*					*	
Wang et al., 2019 [27]	Average task-duration subject			*						*	
Barros et al., 2019 [28]	Power demand and managing power production and response time				*					*	
Nguyen et al., 2019 [29]	Energy and delay						*			*	
Wang and Chen, 2020 [30]	Resource-allocation scheme, completion time and latency							*		*	
Maia et al., 2019 [31]	Potential violation of QoS requirements					*					*
Sun et al., 2019 [32]	Resource allocation							*		*	
Shahid et al., 2020 [33]	Energy and latency							*		*	
Bi et al., 2020 [34]	Energy and delay			*						*	
Forti et al., 2020 [35]	Security								*	*	
Wu et al., 2022 [36]	latency, cost and trust							*		*	
Li et al., 2022 [37]	time-energy consumption and cost			*						*	

3. NETWORK MODEL

Figure 1 shows a three-layer fog computing system with N mobile devices and M cooperative fog nodes. A cloud server *V* accesses directly by mobile devices. A task can be offloaded by a mobile device to one of the fog nodes, the cloud, through a fog node or directly to the cloud server. The model used in our study is similar to that used in paper [17]. The explanation of the parameters is given in Table 2. At each time slot, mobile device i can request to offload a computing task. The mobile devices, fog. $I_i\left(D_i^i, D_i^0, C_i, t_i^r\right)$ nodes, as well as the cloud server meet the delay requirements and are eligible for work processing. So, there are three modes for processing tasks, including mobile devices or local mode, fog nodes mode and cloud server mode.

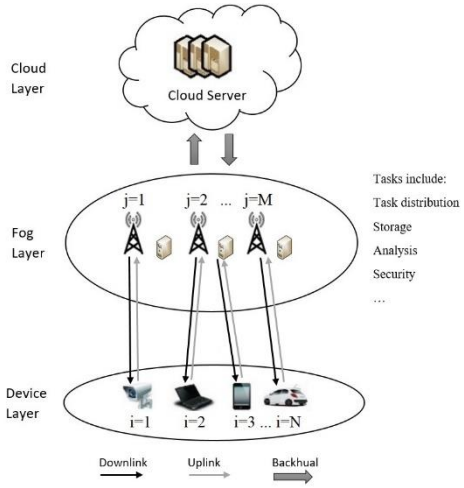


Figure 1. The architecture of fog-computing network.

Table 2. Explanation of parameters.

Parameters	Explanation
D_i^i	Input (including execution code and input data)
D_i^o	The lengths of output (result) data
C_i	The number of CPU cycles (required to execute the task)
t_i^r	The maximum delay requirement of the task
f_i^l	The processing rate of mobile devices i (cycles per second)
T_i^l	The time to perform the task
E_i^l	The energy consumed in the mobile devices (adequate to the CPU cycles required for task I_i) in local processing
v_i	The consumed energy per CPU cycle
R_j^u	The total uplink rate
R_j^d	The total downlink rate
R_j^f	CPU cycle rate
r_{ij}	Spectrum and computation resources for mobile device i
$r_{ij}^f \ r_{ij}^d \ r_{ij}^u$	The CPU cycle, downlink, uplink rates for task execution, input and output transmissions
e_{ij}^u	The energy consumption for transmitting a unit of data
$e_{ij}^{\ d}$	The energy consumption for receiving a unit of data
$T_{ij}^{\ f}$	The delay of mobile devices
E_{ij}^f	The energy consumed in the mobile devices in fog-node processing
$R_{(M+1)}^u$	The uplink rate by the cloud
$R_{(M+1)}^d$	The downlink rate by the cloud
$R_{(M+1)}^f$	The allocated CPU cycle rate by the cloud
w^c	The data rate between a fog node and the cloud server
f^c	The processing rate on the cloud server assigned to each task

3.1 Local Processing

The time to perform the task I_i when it is processed locally is:

$$T_i^l = C_i / f_i^l \tag{1}$$

where, C_i is the number of CPU cycles (required to execute the task) and f_i^l is the processing rate of mobile device i (cycles per second).

The energy consumed in the mobile device (E_i^l), adequate to the CPU cycles required for task I_i, is:

$$E_i^l = v_i C_i \tag{2}$$

and v_i denotes the consumed energy per CPU cycle.

3.2 Fog-node Processing

Fog node j has capabilities defined by a tuple (R_j^u, R_j^d, R_j^f). R_j^u, R_j^d, R_j^f are the total uplink rate, the total downlink rate and CPU cycle rate, respectively. If task I_i is processed at fog node j, this node will allocate spectrum and computation resources for mobile device i ($r_{ij} = (r_{ij}^u, r_{ij}^d, r_{ij}^f)$), where r_{ij}^u is the uplink rate for output transmissions, r_{ij}^d is the downlink rate for input transmissions for task execution and r_{ij}^f is the CPU cycle. Hereon, the energy consumption of the mobile device is to transfer input to and receive output from the fog node j. The delay includes the time of task processing, transmitting input and receiving output at the fog node, so that they are C_i / r_{ij}^f, D_i^i / r_{ij}^u and D_i^o / r_{ij}^d, respectively. D_i^i is input data and D_i^o is output data.

The delay T_{ij}^f and the consumed energy E_{ij}^f of the mobile device are:

$$T_{ij}^f = D_i^i / r_{ij}^u + D_i^o / r_{ij}^d + C_i / r_{ij}^f \tag{3}$$

and

$$E_{ij}^f = E_{ij}^u + E_{ij}^d \tag{4}$$

where $E_{ij}^u = e_{ij}^u D_i^i$ and $E_{ij}^d = e_{ij}^d D_i^o$. The energy consumption for transmitting a unit of data and the energy consumption for receiving a unit of data are denoted by e_{ij}^u and e_{ij}^d.

3.3 Cloud-server Processing

Assume that all fog nodes are connected to a public cloud server. If fog node j forwards task I_i to the cloud server, it will allocate resources for mobile device i, ($r_{ij} = (r_{ij}^u, r_{ij}^d, r_{ij}^f)$) and $r_{ij}^f = 0$. Fog node j sends the input data to the cloud server for processing after receiving the task. It then receives the result and sends it to the mobile device. Hereon, the consumed energy E_{ij}^c at the mobile device and the delay T_{ij}^c are:

$$T_{ij}^c = D_i^i / r_{ij}^u + D_i^o / r_{ij}^d + (D_i^i + D_i^o) / w^c + C_i / f^c \tag{5}$$

and

$$E_{ij}^c = E_{ij}^f = E_{ij}^u + E_{ij}^d \tag{6}$$

where w^c is the data rate between a fog node and the cloud server and f^c is the processing rate on the cloud server assigned to each task.

Because the cloud is in the top tier, it cannot move its task to the top tier. This is accomplished by setting the corresponding latency to infinite: $T_{i(M+1)}^c = \infty$. The total energy consumption $E_{i(M+1)}^c$ is set as a constant. The binary offloading decision variable for task I_i is $x_i = (x_i^l, x_{i1}^f, \ldots, x_{i(M+1)}^f, x_{i1}^c, \ldots, x_{i(M+1)}^c)$, in which $x_i^l = 1$, $x_{ij}^f = 1$ and $x_{ij}^c = 1$ indicate that task I_i is processed at the mobile device, fog node j or the cloud server, respectively. Suppose that $h_i = (T_i^l, T_{i1}^f, \ldots, T_{i(M+1)}^f, T_{i1}^c, \ldots, T_{i(M+1)}^c)$. From (1), (3) and (5), the delay T_i when task I_i is processed is equal to:

$$T_i = h_i^T x_i \tag{7}$$

Suppose that $e_i = (E_i^l, E_{i1}{}^f, \ldots, E_{i(M+1)}{}^f, E_{i1}{}^c, \ldots, E_{i(M+1)}{}^c)$. From (2), (4) and (6), the consumed energy E_i of the mobile device when task I_i is processed is:

$$E_i = e_i^T x_i \tag{8}$$

Assuming that $e = (e_1, ..., e_N)$ and $x = (x_1, ..., x_N)$, the consumed energy E_i of the mobile device is as follows:

$$E = e^T x \tag{9}$$

The purpose of our paper is to minimize the total energy consumption of all mobile devices in the delay requirement, which is a joint offloading decision (x) and resource allocation ($r = \{r_{ij}\}$) problem:

$$\mathrm{Min}_{x,r} e^T x \tag{10}$$

the restrictions of which are equal to:

$$\begin{cases} (\mathrm{C1}) \quad T_i \le t_i^r, \forall i \in N \\ (\mathrm{C2}) \quad \sum_{i=1}^{N} r_{ij}^f \le R_j^f, \forall j \in \mathrm{M}^*, \\ (\mathrm{C3}) \quad \sum_{i=1}^{N} r_{ij}^u \le R_j^u, \forall j \in \mathrm{M}^*, \\ (\mathrm{C4}) \quad \sum_{i=1}^{N} r_{ij}^d \le R_j^d, \forall j \in \mathrm{M}^*, \\ r_{ij}^f, r_{ij}^u, r_{ij}^d \ge 0, \forall (i, j) \in N \times \mathrm{M}^* \end{cases} \tag{11}$$

and

$$\begin{cases} (\mathrm{C5}) \quad x_i^l + \sum_{j=1}^{M+1} x_{ij}^f + x^c = 1, \forall i \in N, \\ x_i^l, x_{ij}^f, x_{ij}^c \in \{0,1\}, \forall (i, j) \in N \times \mathrm{M}^* \\ (\mathrm{C6}) \quad T_i^l < t_i^r \\ (\mathrm{C7}) \quad T_{ij}^c < t_i^r \\ (\mathrm{C8}) \quad T_{ij}^f < t_i^r \end{cases} \tag{12}$$

where (C1) is the delay requirement of tasks. Resource constraints at fog nodes are (C2), (C3) and (C4). Offloading decision constraints are (C5) and (C6) to (C8), indicating that the task delay for each mobile device should not exceed the maximum value (t_i^r).

According to the equations presented in [16], the revenue of the service provider (cloud) leads to better services for member users. Another factor in measuring cost performance is the price that users are charged as benefit/revenue. The price offered by each user depends on the delay requirement T_{ij} and data size D_i^i and D_i^o, where we assume a linear relationship between the price and the data size and an inversely linear relationship between the price and the delay requirement. Therefore, the offer from each user is equal to:

$$O_{ij} = f\left(D_i^i, D_i^o, T_{ij}\right) \tag{13}$$

where the function $f()$ must be a monotonic increasing function for D_i^i and D_i^o must be a monotonic increasing function for T_{ij}. For simplicity, the following function is used to define $f\left(D_i^i, D_i^o, T_{ij}\right)$:

$$O_{ij} = a \frac{D_i^i + D_i^o}{T_{ij}} \tag{14}$$

where a is a parameter with the unit of dollar/Mbps and O_{ij} is the price that the user of the mobile device i pays if it is compatible with V. Because V serves more than one user, it receives more than one offer. Revenue V is the sum of offers defined by all users. For simplicity, it is assumed that the cost of V is related to the power consumption of the transmission and its maintenance; in this work, it is fixed. In revenue V, the impact of fixed-service costs is ignored. The system objective in our article is named cost performance. Cost performance is defined as the ratio of the average data rate of each user to its cost in Mbps/sec/dollar.

The actual delay value is related to the size of the data D_i^i and D_i^o that must be transmitted and processed. Then, for the cost factor, use the user's payment/offer for the relevant fog calculation service that it acquires. As a result, the cost performance function is defined for combining two factors in one criterion for each user, which indicates the quality of services for which the user pays. The cost performance system (CP_{sys}) is equal to:

$$CP_{sys} = \frac{\sum_{u_i \in U} CP_{ij}}{N} \tag{15}$$

where CP_{ij} is the cost performance value for each mobile device i and is shown as follows:

$$CP_{ij} = x_{ij}^f \frac{\left(D_i^i + D_i^o\right) / T_{ij}^f}{O_{ij}} \tag{16}$$

The optimization problem of our article is shown below:

$$Max \;\; CP = \sum_{ij} x_{ij}^f \frac{(D_i^i + D_i^o)/T_{ij}^f}{O_{ij}} \tag{17}$$

the restrictions of which are equal to:

$$\begin{cases} \text{(C9)} \;\; \sum_{ij} x_{ij}^f \times t_{ij}^f < t_i^r \\ \text{(C10)} \; \sum_{ij} x_{ij}^f \times r_{ij}^f \le R_j^f \\ \text{(C11)} \;\; \sum_{ij} x_{ij}^f \times r_{ij}^u \le R_j^u \\ \text{(C12)} \; \sum_{ij} x_{ij}^f \times r_{ij}^d \le R_j^d \end{cases} \tag{18}$$

where (17) is the system objective and shows the overall system cost performance for users. (C9) indicates that each user who has been served has a task delay less than the maximum delay requirement of the task (t_i^r). In (C10), (C11) and (C12), for each user who has been served, the CPU cycle, downlink and uplink rates for task execution and output and input transmissions of all users must be less than the CPU cycle rate (R_j^f), the total uplink rate and the total downlink rate.

4. PROBLEM SOLVING

The Experimental parameters are shown in Table 3 and are similar to the experimental parameters in [17]. GAMZ solves the problem as a single-objective. The weighted-sum method is used to turn our two-objective problem into a single-objective problem. Eq. (19) has two goals: to increase cost performance and reduce energy consumption. Since one of the goals of the problem is maximization, to achieve the ultimate goal of the problem, which is a minimization of the objective function, must be multiplied by -1. First, through the weighted sum method, two weights with a value of 0.5 are assigned to the objective function and are added up to become a minimized objective function. The choice of weight for the objective function depends on the priority and importance of the selected objective function. Because both cost performance and energy consumption are equally important in this paper, a factor of 0.5 is considered for both.

$$Min \;\; (-0.5) \times \sum_{ij} x_{ij}^f \frac{\left(D_i^i + D_i^o\right)/T_{ij}^f}{O_{ij}} + 0.5 \times e^T x \tag{19}$$

In the next step, inside the GAMS, for a small model with five mobile devices, four fog nodes and a cloud, the model is entered into the program. Indices are displayed in GAMS with sets, which in this article include i for mobile devices at 5, j for fog nodes at 3, k for cloud server at 1 and m is an auxiliary index at 4. After entering the parameters, decision variables and sets, it is time to choose the appropriate solver. Both Eq. (10) and Eq. (17) are nonlinear and MINLP. Because the sum of Eq. (10) and (17) is nonlinear and MINLP, Eq. (19) is also nonlinear and MINLP. In this case, because our problem is MINLP, which is generally NP-hard to solve [38], the best solver is Baron, which solves these kinds of problems with a good track record. After execution, the results are as follows: the decision

variables are quantified, the optimal answer is obtained and resources are allocated to the devices. The decision variable x^f determines which mobile is served by which device. The lower bound is set r_{ij}^u, r_{ij}^d and r_{ij}^f from 0.01, because in equations where r_{ij}^u, r_{ij}^d and r_{ij}^f are at the denominator of the fraction, they cannot have a value of zero. As described, the weighting method is used to balance the objects. Considering w1, w2 with values of 0.5 and 0.5, the lowest amount obtained for the objective function was -1624.439.

Table 3. Experimental parameters.

Parameters	Value
Number of mobile devices N	10
Number of fog nodes M	4
Number of cloud servers V	1
CPU rate (in mobile devices) f_i^l	0.5 Giga cycles/s
Processing energy consumption rate v_i	1000/730 J/Giga cycles
Input data size D_i^i	U(a, b) MB
Output data size D_i^o	U(c, d) MB
Required CPU cycles C_i	$\alpha_i \times D_i^i$
Unit transmission energy consumption to fog nodes e_{ij}^u ($\forall j \leq M$)	0.142 J/Mb
Unit receiving energy consumption from fog nodes e_{ij}^d ($\forall j \leq M$)	0.142 J/Mb
Unit transmission energy consumption to cloud server V ($e_{i(M+1)}^u$)	0.658 J/Mb
Unit receiving energy consumption from cloud server V ($e_{i(M+1)}^d$)	0.278 J/Mb
Delay requirement t_i^r	[1, 10]s
Processing rate (each fog node) R_j^f	10 Giga cycles/s
Uplink data rate (each fog node) R_j^u	72 Mbps
Downlink data rate (each fog node) R_j^d	72 Mbps
CPU rate (the cloud server) f^c	10 Giga cycles/s
Data rate between FNs and the cloud w^c	5 Mbps
A parameter with the unit dollar/Mbps a	1
Delay requirement T_{ij}	[6, 7]s

5. EVALUATION

In this section, the implementation of meta-heuristic algorithms including evaluation setup, experimental results and results' analysis, is presented.

5.1 Evaluation Setup

The proposed model was implemented by use of PSO [10], GA [11], SA+GA [12], TLBO [13], GWO [14], GOA [15] and random search method for thirty problems of different sizes to solve our single-objective problem (19). The proposed models are solved using meta-heuristic algorithms to solve problems in small, medium and large sizes. Because the problem is of the MINLP type, meta-heuristic algorithms have a good track record of solving this type of problem. Also, the model is based on population and the algorithms used in this paper are the same. Meta-heuristic algorithms have acceptable speed and accuracy in finding the optimal solution in resource-allocation models. These algorithms are scalable for resource-allocation optimization problems (for more iterations and larger populations) [39]-[40]. Programming and execution of algorithms are also carried out using MATLAB version (R2016a). The algorithms run on a 64-bit system with a 2.5 GHz processor and a 2 GB memory.

To investigate the performance of the proposed model, PSO [10], GA [11], SA+GA [12], TLBO [13], GWO [14], GOA [15] and random method have been used. Then, a comparison of the results obtained from each of these methods is performed. To do this, different instances of different sizes must first be designed. The parameters of each algorithm are given in Table 4.

Table 4. The parameters of the algorithms.

Algorithm	Parameters	Values
PSO	maximum number of iterations	100
	population size (swarm size)	50
	inertia weight	1
	inertia weight damping ratio	0.99
	personal rating coefficient	2
	global rating coefficient	2
GA	maximum number of iterations	100
	population size	50
	Cross-over percentage	0.8
	number of off-springs	2×round (cross-over percentage × population size/2)
	mutation percentage	0.3
	number of mutants	round (mutation percentage × population size)
	gamma	0.05
	mutation rate	0.02
	beta	0.5
SA+GA	maximum number of iterations	100
	maximum number of sub-iterations	10
	initial temp.	10
	temp reduction rate	0.99
	population size	50
	Cross-over percentage	2
	number of parents (off-springs)	2×round (cross-over percentage × population size/2)
	mutation percentage	0.3
	number of mutants	round (mutation percentage × population size)
	Cross-over inflation rate	0.05
	mutation rate	0.02
	mutation mode	“rand”
	eta	0.1
TLBO	maximum number of iterations	100
	population size	50
GWO	maximum number of iterations	100
	population size	50
GAO	maximum number of iterations	100
	population size	50

The computational complexities of PSO [10], GA [11], SA+GA [12], TLBO [13], GWO [14], GOA [15] are O(nlogn),O(nm),O(nm),O(ldg),O(nm) and $O(m^3)$, respectively, where n is the number of population, m is the size of individuals, d is the number of subjects, g is the number of iterations and l is the number of learners.

In this research, instances in small, medium and large sizes are designed and the size of each parameter for each type of example is classified as follows (*i* for the number of mobile devices and *j* for the number of fog nodes):

- For small instances, the set $i = 5$ and $j = 3$ with 60 variables,
- For medium instances, the set $i = 15$ and $j = 5$ with 300 variables,
- For large instances, the set $i = 100$ and $j = 15$ with 6000 variables.

5.2 Experimental Results and Results' Analysis

The results related to the values of the objective function and the solution times for these algorithms are given in Table 5 and Table 6.

Table 5. The values of the objective function.

Problem type	No.	GA	PSO	SA+GA
Small problem	1	-3,624	-3,239	-7537250.18
	2	-5,137	-4,633	-26513.6
	3	-5,582	-4,840	-1128
	4	-3,045	-2,912	-15617.6
	5	-5,239	-4,782	-11922.9
	6	-3,488	-2,846	-608.9
	7	-4,823	-3,925	-579.9
	8	-3,256	-2,885	-69918.4
	9	-7,144	-6,547	-21484.3
	10	-6,739	-5,168	-11940.5
	11	-4,958	-4,151	-17363.5
	12	-5,786	-5,004	-5701.3
medium problem	13	-493,059	-435,452	-412371.2
	14	-626,800	-591,089	-446073.1
	15	-829,385	-781,866	-511573.6
	16	-743,971	-694,826	-648282.3
	17	-720,405	-685,978	-2437.3
	18	-534,126	-493,097	-397386.3
	19	-694,271	-562,966	-439611.8
	20	-495,543	-402,779	-211423.9
	21	-413,253	-354,656	-2813.9
	22	-748,903	-687,841	-315033.7
large problem	23	-3,433,470	-3,028,605	-49497370.4
	24	-3,737,970	-3,478,755	-19938346.9
	25	-3,891,427	-3,606,192	-31052647
	26	-3,759,422	-3,313,783	-14822865
	27	-3,027,824	-2,831,833	-32319789.1
	28	-3,487,582	-3,117,100	-29267467.9
	29	-2,517,827	-2,390,396	-27467320.4
	30	-4,728,847	-4,310,443	-32405902.4
Problem type	**TLBO**	**GOA**	**GOW**	**Random**
Small problem	-26848.6	-9506.1	-1436	-110.35
	-33697.5	-21977.9	-65731.6	-74.49
	-943.8	-1062.6	-1216.7	-221.76
	-2744.9	-5338.4	-13687.9	-11.25
	-4662.5	-8130.7	-984	-119.24
	-22366.2	-13209.1	-775.6	-67.47
	-1084.6	-7559.5	-3531.5	-32.69
	-78935	-4933.3	-33810.1	-84.93
	-1659.9	-17399.8	-12122.8	-210.67
	-7981.7	-14711.6	-24042.3	-203.19
	-8174.7	-759.4	-83975.9	-118.37
	-3080.8	-12755.8	-1591.3	-19.67
medium problem	-14450.6	-25405.9	-2501.9	309.11
	-501673.5	-35910	-71205.4	397.31
	-133464.3	-2098.5	-134118.1	100.33
	-136988.4	-165627.6	-93339.8	115.60
	-63350.8	-86799	-270145	220.36
	-199839.4	-82712.8	-75138.9	79.14

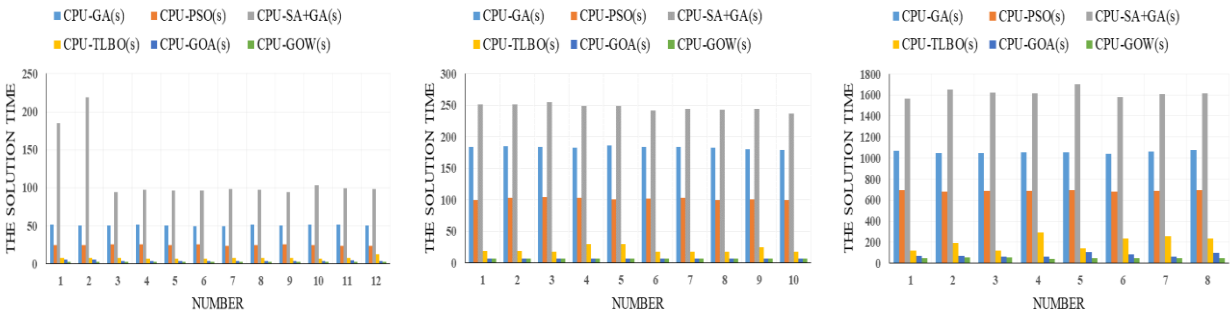
	-192184.3	-216008.7	-82738.5	19.35
	-148564.8	-206550.1	-138484.9	10.46
	-147154.4	-148449.6	-190762.6	24.89
	-166501.5	-70551.7	-65520.3	114.49
large problem	-3620749.4	-881734	-4128805	9.17
	-4048094.4	-858796.8	-504324.6	14.11
	-5092522.9	-830234.3	-776526	-25.21
	-4557760.8	-656639	-4638037	-7.82
	-2747554	-794339.2	-3261333.7	10.35
	-3628517.4	-361017.3	-3628227.1	-2.56
	-4323419	-946594.2	-4020436.5	-18.59
	-4323411	-1013361.6	-4952766.4	-4.90

In Table 5, the values of the objective function are shown. As can be seen in Table 5, the problem is investigated in small, medium and large dimensions. Cost is obtained at each stage and the larger the problem, the lower and more convergent the cost function becomes. Because the presence of all meta-heuristic algorithms at every stage, these algorithms try to get a more optimal answer. The solution times are indicated in Table 6 and Figure 2.

Table 6. The values of the solution time.

Problem type	**No.**	**CPU-GA(s)**	**CPU-PSO(s)**	**CPU-SA+GA(s)**	**CPU-TLBO(s)**	**CPU-GOA(s)**	**CPU-GOW(s)**
Small problem	1	51.11	24.77	185.5	7.3	5.3	2.9
	2	50.64	24.53	219.4	7.1	5.6	2.5
	3	50.37	25.25	94.2	7.4	3.5	2.6
	4	50.96	25.27	97.3	7	3.7	2.5
	5	50.44	24.48	96.3	7	3.9	2.5
	6	49.82	25.46	96.5	6.9	3.4	2.6
	7	49.88	23.69	98.1	7.5	3.6	2.5
	8	51.03	24.55	97.3	7.3	3.4	2.5
	9	50.81	25.29	93.9	7.4	3.4	2.8
	10	51.07	24.01	103.2	7	3.4	2.5
	11	51.24	23.34	99.8	7.1	4.8	2.5
	12	50.15	23.35	97.9	12.7	3.4	2.5
medium problem	13	184.12	99.56	251.5	18.7	7.1	6.4
	14	185.44	103.23	251.2	18.2	7.1	6.3
	15	183.34	103.83	255.2	17.6	6.9	7
	16	183.02	103.03	248.4	29.4	7.1	6.3
	17	186.32	100.16	248.6	29.8	7.1	6.4
	18	184.28	102.43	241.6	17.7	6.9	6.4
	19	184.06	102.99	244	17.8	7.1	6.3
	20	182.29	99.46	243	17.8	6.9	6.3
	21	180.19	100.85	244.7	25	6.9	6.3
	22	179.27	99.04	237.4	17.8	7	6.3
large problem	23	1068.48	693.68	1566.1	118.5	67.8	46.6
	24	1045.26	681.12	1651.1	193.1	67.5	58
	25	1049.38	687.39	1621.5	120.7	61.2	52.4
	26	1053.51	689.14	1617.7	294.2	60.1	44.2
	27	1054.37	694.76	1705.1	141.2	102.5	47.1
	28	1038.45	681.16	1577.4	237.2	83.7	50.3
	29	1062.94	686.94	1606.5	255.8	65.1	45.7
	30	1078.55	694.17	1616.1	238.2	99.6	46.4

In Table 6, due to the fact that small problems have fewer variables and parameters and the number of fogs and users is less, the execution time of the algorithm is shorter and they are executed faster than the next steps when the number of fogs and users increases. Finally, in larger problems, the number of variables reaches 6,000, which will certainly be solved in more time. It should be noted that to eliminate the uncertainty in the obtained outputs, each problem was performed three times and the average of these three problems was reported as the final answer variable.



a b c

Figure 2. The values of the solution time for a) small problems, b) medium problems and c) large problems.

The plot of a problem with fifty mobile users, ten fog nodes and a cloud sever is shown in Figure 3, while the plot of a problem with one hundred mobile users, ten fog nodes and a cloud sever is shown in Figure 4. As you can see in Figure 3, in a number of iterations below 10, the SA+GA [12] graph fluctuates upwards, which can be due to the lower stability of this algorithm in such number of iterations. In Figure 3 and Figure 4 in each step, the meta-heuristic algorithms perform the optimization process and try to reduce the total cost in each step and obtain a more optimal value. In fact, this leads to the most accurate answer, allocating resources of a certain size with the minimum delay to users through fog nodes or directly to the cloud server.

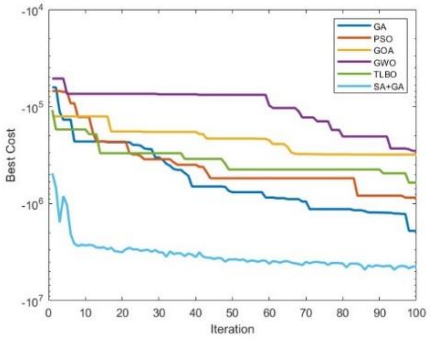


Figure 3. The value of the objective function in the number of iterations for fifty mobile users, ten fog nodes and a cloud sever.

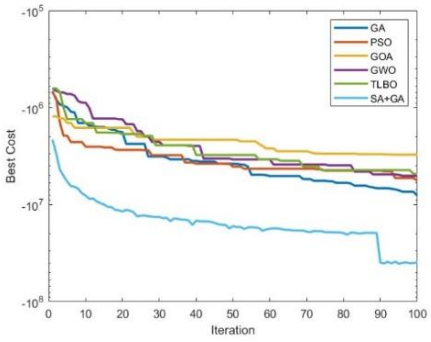


Figure 4. The value of the objective function in the number of iterations for one hundred mobile users, ten fog nodes and a cloud sever.

To study and compare the proposed algorithms accurately, decision-making with multiple criteria has been used. Multiple-attribute decision making (MADM) method is a multi-criteria decision-making method; by which TOPSIS method is used for comparing algorithms with various indicators. Since this method has been used in a variety of articles and ended in excellent and accurate results, it is also used here for comparison. In this model, two indicators of execution time (CPU time) and the value of the objective function are considered for comparison. Since not all the amounts can be taken into account due to their large number, the average of each column is obtained and then the values are entered into TOPSIS. On the other hand, because an algorithm is better in terms of each indicator, there is no such decision-making ability to choose a better algorithm. The result of TOPSIS method is shown in Table 7. According to Table 7 of the TOPSIS comparison, compared to other methods, the SA+GA method with a value of 0.23 is deemed the best. Then come GWO [14], GA [11], TLBO [13], PSO [10] and GOA [15] methods, respectively.

Table 7. The result of TOPSIS method.

Algorithms	Ranking
SA+GA	0.23
GWO	0.197
GA	0.167
TLBO	0.146
PSO	0.117
GOA	0.104

In Figure 5 (with ten fog nodes), as the number of mobile devices increases, the cost performance increases too. The highest cost performance is for the GWO [14], GOA [15], TLBO [13], SA+GA [12], PSO [10] and GA [11], respectively. GA has the lowest gradient, because it has a slower convergence. GWO has the highest cost performance increase because of its high convergence and iterability. With the increase in the number of users, the cost performance has increased as well, because more users try to get services and the size of input-output data (D_i^i and D_i^o) increases too and as the goal is to increase cost performance, the latency decreases at each stage.

In Figure 6 (with ten fog nodes), as the number of mobile devices increases, the energy consumption increases too. The highest energy consumption is for the GWO [14], GOA [15], TLBO [13], SA+GA [12], PSO [10] and GA [11], respectively. In Figure 6, because of the power of convergence and iterability, GWO has the highest and GA the lowest power consumption. Given the increase in the number of users, the increase in the need for services and the allocation of more resources, the number of users connecting to fog nodes has increased, while the delay may increase with regard to the constant number of fog nodes. So, energy consumption increases.

In Figure 7 (with forty mobile devices), as the number of fog nodes increases, the cost performance increases too. The highest cost performance is for the GWO [14], GOA [15], TLBO [13], SA+GA [12], PSO [10] and GA [11], respectively. With the number of fog 5, GOA has the highest cost performance, because the algorithm has lower convergence and TLBO performs better due to greater stability.

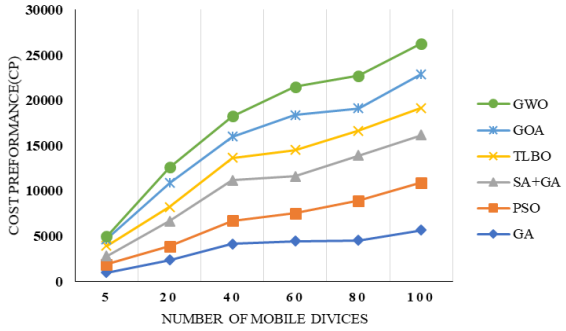


Figure 5. The cost performance with ten fog nodes.

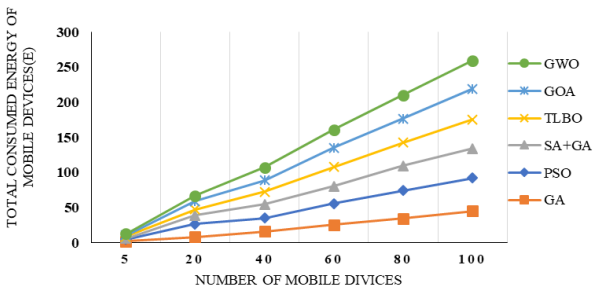


Figure 6. The energy consumption with ten fog nodes.

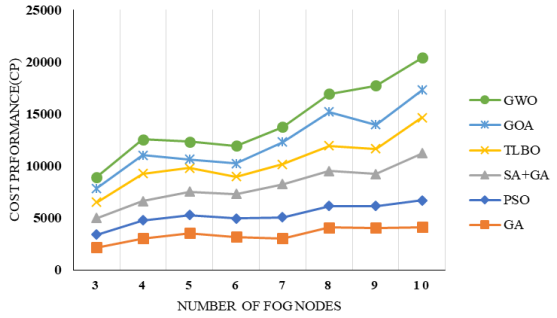


Figure 7. The cost performance with forty mobile devices.

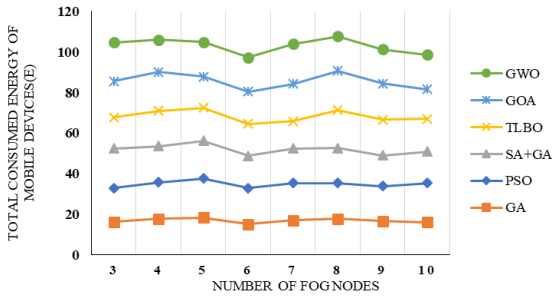


Figure 8. The energy consumption with forty mobile devices.

With the number of fog nodes of 9, the GOA again has the highest performance decline, which indicates the instability of this pattern compared to the rest. Figure 7 shows a slight upward slope of cost performance with increasing number of fog nodes. When the number of users is considered constant, with an increase of one fog node, the cost performance is slightly increased due to the fact that the size of input-output data (D_i^i and D_i^o) is slightly increased. As the number of available fog nodes increases, work is done faster and better load balancing occurs on the network, so better cost performance occurs at each step.

In Figure 8 (with forty mobile devices), as the number of fog nodes increases, the energy consumption increases too. The highest energy consumption is for the GWO [14], GOA [15], TLBO [13], SA+GA [12], PSO [10] and GA [11], respectively. All algorithms with the number of fog nodes of 6 have a reduction in consumed energy, which can be related to the number of users that have been set in the parameters and the initial amount of resources (R_j^u, R_j^d, R_j^f). With increasing the number of fog nodes, the changes in energy consumption remain constant, because the number of users has not changed, so the number of requests remains almost constant. When sending a request to fog nodes, the fog node that is closer to the user has a higher priority to communicate with the intended user, so changes in energy consumption remain almost constant.

6. CONCLUSIONS

Cloud computing plays an essential role in developing the Internet of Things, which provides data-processing and storage services. However, many of its applications suffer from cloud-computing challenges, such as latency, location awareness and real-time mobility support. Fog computing, which almost looks like the evolution of cloud computing, helps provide solutions to these challenges. In this paper, due to their service latency limitations, the total energy of all mobile devices is minimized and the cost performance is maximized at the same time. This generally leads to the increase in the quality of service (QoS) and in the quality of experience (QoE). A model is designed and the exact mathematical method with the GAMS program is used to prove its logical process. In the next step, the model is solved with GA, PSO, SA+GA, TLBO, GOA, GWO and random methods. To do this, sproblems are considered with three sizes: small, medium and large. The six main algorithms (GA, PSO, SA+GA, TLBO, GOA and GWO) are compared with two indicators; the value of the objective function and the execution time. According to the TOPSIS comparison, the SA+GA method with a value of 0.23 is the best one compared to other methods. Then come GWO, GA, TLBO, PSO and GOA methods, respectively. In this paper, the customer affairs is one of the practical applications of this modeling, in terms of meeting the needs of customers, improving the level of loyalty and establishing mutual, transparent, respectful relationships and satisfying them, which are among the priorities of the customer relationship center. Communication with customers and the method of responding to their needs and requests should be well managed. Also, the cost performance and energy consumption should be optimized. In project management, resource allocation means the distribution of available resources in the company (equipment, labor, budget, facilities, …etc.). In order to perform the tasks related to project management, two objectives of the problem and its solutions are implemented. Organizational resource management, in fact, seeks effective planning and control by modeling and providing solutions. All affairs related to receiving, producing and delivering to customers should be done for production, distribution and service companies with the highest cost performance and the lowest energy consumption. In fact, one of the concerns of this modeling is to provide the customer's desired service with the least delay and in a reasonable time. For future work, more goals can be added to the model to bring the system closer to reality. More attributes for comparison should be added. Also, the architecture of the fog computing network can be changed and newer technologies can be used in it. Customer service can be prioritized, too. New methods including machine learning and deep learning can be used to assign tasks intelligently and analyze data by fog computing. Also, neural networks can be used for the problems of load balancing and task scheduling.

REFERENCES

[1] G. Javadzadeh and A. M. Rahmani, "Fog Computing Applications in Smart Cities: A Systematic Survey," Wireless Networks, vol. 26, no. 2, pp. 1433–1457, 2020.

[2] A. Yousefpour et al., "All One Needs to Know about Fog Computing and Related Edge Computing Paradigms: A Complete Survey," J. of Systems Architecture, vol. 98, pp. 289–330, 2019.
[3] R. W. Cottle and M. N. Thapa, Linear and Nonlinear Optimization, 1st Edn., New York, Springer, 2017.
[4] S.-A. N. Alexandropoulos, C. K. Aridas, S. B. Kotsiantis and M. N. Vrahatis, "Multi-objective Evolutionary Optimization Algorithms for Machine Learning: A Recent Survey," Chapter in Book: Approximation and Optimization, vol. 145, pp. 35–55, 2019.
[5] K. Deb and K. Deb, "Multi-objective Optimization," Chapter in Book: Search Methodologies, pp. 403–449, DOI: 10.1007/978-1-4614-6940-7_15, 2014.
[6] G. Colson and C. de Bruyn, "Models and Methods in Multiple Objectives' Decision Making," Math. Comput. Model., vol. 12, no. 10–11, pp. 1201–1211, 1989.
[7] P. Muts et al., "The Decomposition-based Outer Approximation Algorithm for Convex Mixed-integer Nonlinear Programming," J. Global Optimization, vol. 77, no. 1, pp. 75–96, 2020.
[8] M. R. Bussieck and A. Meeraus, "General Algebraic Modeling System (GAMS)," Proc. of Modeling Languages in Mathematical Optimization, vol. 88, pp. 137-157, 2004.
[9] A. Brooke, D. Kendrick, A. Meeraus, R. Raman and U. America, "The General Algebraic Modeling System," GAMS Development Corporation, 1050, 1998.
[10] Rajani, D. Kumar and V. Kumar, "Impact of Controlling Parameters on the Performance of MOPSO Algorithm," Procedia Computer Science, vol. 167, pp. 2132–2139, 2020.
[11] Li, Mincan et al., "Multi-task Allocation with an Optimized Quantum Particle Swarm Method," Applied Soft Computing, vol. 96, p. 106603, 2020.
[12] P. J. Van Laarhoven and E. H Arts, "Simulated Annealing: Theory and Applications," Springer, Part of the Book Series: Mathematics and Its Applications, vol. 37, pp. 7-15, 1987.
[13] R. V. Rao et al., "Teaching– Learning-based Optimization: A Novel Method for Constrained Mechanical Design Optimization Problems," Computer Aided Design, vol. 43, no. 3, pp. 303–315, 2011.
[14] S. Mirjalili et al., "Grey Wolf Optimizer," Advances in Eng. Software, vol. 69, pp. 46–61, 2014.
[15] S. Saremi, S. Mirjalili and A. Lewis, "Grasshopper Optimization Algorithm: Theory and Application," Advances in Eng. Software, vol. 105, pp. 30–47, 2017.
[16] Y. Gu, Z. Chang, M. Pan, L. Song and Z. Han, "Joint Radio and Computational Resource Allocation in IoT Fog Computing," IEEE Transactions on Vehicular Technol., vol. 67, no. 8, pp. 7475–7484, 2018.
[17] T. T. Vu et al., "Optimal Energy Efficiency with Delay Constraints for Multi-layer Cooperative Fog Computing Networks," IEEE Transactions on Communications, vol. 69, no. 6, pp. 3911–3929, 2021.
[18] T. T. Vu, N. Van Huynh, D. T. Hoang, D. N. Nguyen and E. Dutkiewicz, "Offloading Energy Efficiency with Delay Constraint for Cooperative Mobile Edge Computing Networks," Proc. of the 2018 IEEE Global Comm. Conf. (GLOBECOM), DOI: 10.1109/GLOCOM.2018.8647856, Abu Dhabi, UAE, 2018.
[19] A. M. Rahmani et al., "Exploiting Smart e-Health Gateways at the Edge of Healthcare Internet-of-Things: A Fog Computing Approach," Future Generation Computer Systems, vol. 78, pp. 641–658, 2018.
[20] M.-H. Chen, M. Dong and B. Liang, "Resource Sharing of a Computing Access Point for Multi-user Mobile Cloud Offloading with Delay Constraints," IEEE Trans. on Mobile Computing, vol. 17, no. 12, pp. 2868–2881, 2018.
[21] J. Du, L. Zhao, J. Feng and X. Chu, "Computation Offloading and Resource Allocation in Mixed Fog/Cloud Computing Systems with Min-max Fairness Guarantee," IEEE Trans. on Communications, vol. 66, no. 4, pp. 1594–1608, 2018.
[22] J. Du, L. Zhao, X. Chu, F. R. Yu, J. Feng and Chih-Lin, "Enabling Low-latency Applications in LTE-A Based Mixed Fog/Cloud Computing Systems," IEEE Trans. on Vehicular Technol., vol. 68, no. 2, pp. 1757–1771, 2019.
[23] H. Xing, L. Liu, J. Xu and A. Nallanathan, "Joint Task Assignment and Resource Allocation for D2D-enabled Mobile-edge Computing," IEEE Trans. on Communications, vol. 67, no. 6, pp. 4193–4207, 2019.
[24] C.-F. Liu et al., "Dynamic Task Offloading and Resource Allocation for Ultra-reliable Low-latency Edge Computing," IEEE Trans. on Communications, vol. 67, no. 6, pp. 4132–4150, 2019.
[25] T. X. Tran and D. Pompili, "Joint Task Offloading and Resource Allocation for Multi-server Mobile-edge Computing Networks," IEEE Trans. Vehicular Technol., vol. 68, no. 1, pp. 856–868, 2019.
[26] Q. Fan and N. Ansari, "Towards Traffic Load Balancing in Drone-assisted Communications for IoT," IEEE Internet of Things J., vol. 6, no. 2, pp. 3633–3640, 2019.
[27] Y. Wang, X. Tao, X. Zhang, P. Zhang and Y. T. Hou, "Cooperative Task Offloading in Three-tier Mobile Computing Networks: An ADMM Framework," IEEE Transactions on Vehicular Technology, vol. 68, no. 3, pp. 2763–2776, 2019.
[28] E. B. C. Barroset al., "Fog Computing Model to Orchestrate the Consumption and Production of Energy in Microgrids," Sensors (Basel), vol. 19, no. 11, p. 2642, 2019.
[29] N. T. Ti, L. B. Le and Q. Le-Trung, "Computation Offloading in MIMO Based Mobile Edge Computing Systems under Perfect and Imperfect CSI Estimation," IEEE Trans. on Services Computing, vol. 14, no. 6, pp. 2011–2025, 2021.

[30] Q. Wang and S. Chen, "Latency-minimum Offloading Decision and Resource Allocation for Fog-enabled Internet of Things Networks," Trans. on Emerging Telecomm. Techn., vol. 31, no. 12, p. e3880, 2020.

[31] A. M. Maia et al., "Optimized Placement of Scalable IoT Services in Edge Computing," Proc. of the 2019 IFIP/IEEE Sympo. on Integrated Network and Service Manag. (IM), pp. 189-197, Arlington, USA, 2019.

[32] Y. Sun, M. Peng, S. Mao and S. Yan, "Hierarchical Radio Resource Allocation for Network Slicing in Fog Radio Access Networks," IEEE Trans. on Vehicular Technol., vol. 68, no. 4, pp. 3866–3881, 2019.

[33] M. H. Shahid et al., "Energy and Delay Efficient Fog Computing Using Caching Mechanism," Computer Communications, vol. 154, pp. 534–541, 2020.

[34] S. Bi et al., "Joint Optimization of Service Caching Placement and Computation Offloading in Mobile Edge Computing Systems," IEEE Trans. on Wireless Comm., vol. 19, no. 7, pp. 4947–4963, 2020.

[35] S. Forti, G.-L. Ferrari and A. Brogi, "Secure Cloud-edge Deployments with Trust," Future Generation Computer Systems, vol. 102, pp. 775–788, 2020.

[36] B. Wu, X. Lvl, W. D. Shamsi and E. G. Dizicheh, "Optimal Deploying IoT Services on the Fog Computing: A Metaheuristic-based Multi-objective Approach," J. King Saud Univ.-Computer Inf. Sci., vol. 34, no. 10, pp. 10010-10027, 2022.

[37] H. Li, P. Zheng, T. Wang, J. Wang and T. Liu, "A Multi-objective Task Offloading Based on BBO Algorithm under Deadline Constrain in Mobile Edge Computing," Cluster Computing, DOI: 10.1007/s10586-022-03809-7, 2022.

[38] D. Bertsimas and J. N. Tsitsiklis, Introduction to Linear Optimization, ISBN: 1886529191, Athena Sci., 1997.

[39] M. Abdel-Basset et al., "Energy-aware Metaheuristic Algorithm for Industrial IoT Task Scheduling Problems in Fog Computing Applications," IEEE IoT J., vol. 8, no. 16, pp. 12638–12649, 2021.

[40] K. P. N. Jayasena and B. S. Thisarasinghe, "Optimized Task Scheduling on Fog Computing Environment Using Meta Heuristic Algorithms," Proc. of the 2019 IEEE Int. Conf. on Smart Cloud (SmartCloud), pp. 53-58, DOI: 10.1109/SmartCloud.2019.00019, 2019.

ملخص البحث:

تلعبُ الحوسبة السّحابية دوراً أساسياً في تطوُّر إنترنت الأشياء، التي توفّر خدمات معالجة البيانات وتخزينها. حيث تساعد الحوسبة الضّبابية المتطورة من الحوسبة السّحابية في إيجاد حُلولٍ للتّحدّيات التي تواجه الحوسبة السّحابية، مثل التّأخير والوعي بالموضِع ودعم حَرَكية الزّمن الحقيقي. وتعمل الحوسبة الضّبابية على ملء الفجوة بين السّحابة وأجهزة إنترنت الأشياء بالقرب من تلك الأجهزة. وهكذا فإنّ الحوسبة، والتّشبيك، وتخزين البيانات، وإدارتها، واتّخاذ القرار تتمّ جميعها على طول المسار بين السّحابة وأجهزة إنترنت الأشياء. وإنّ الإدارة الأوتوماتيكية والذّكية لمصادر العُقد الضّبابية وتحقيق سياسة جدْولةٍ فعّالة في نموذج الحوسبة هما متطلّبان ضروريان ويقودان الى تطوير الأداء الإجمالي للحوسبة الضّبابية.

في هذه الورقة، تمّ تصميم نموذجٍ له هدفان هما: تحسين الأداء المتعلّق بالتّكلفة من جهةٍ، وتوفير استهلاك الطّاقة من جهةٍ ثانية. ويُعرَّف الأداء الخاص بالتّكلفة بأنّه النّسبة بين معدَّل البيانات للمُسْتخدِم وبين التّكلفة المتعلّقة بتلك البيانات. وقد تمّ استخدام طريقةٍ رياضيةٍ دقيقةٍ مع برنامج (GAMS) لإثبات العملية المنطقية للنّموذج. وفي الخطوة التّالية، قُمنا بحلّ النّموذج باستخدام عدّة طُرق. وبينت نتائج المقارنة أنّ (SA+GA) كانت أفضل طُرق الحلّ بقيمةٍ بلغت (0.23) مقارنةً بالطُّرق الأخرى، وجاءت بعدها الطُّرق (GWO؛ GA؛ TLBO؛ PSO؛ GOA)، على الترتيب. وقد أدّى استخدام النّموذج المقترح الى الحصول على أعلى أداءٍ للتّكلفة بأقلّ استهلاكٍ للطّاقة.

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 09, No. 02, June 2023.

SEMANTIC RETRIEVAL FOR INDONESIAN QURAN AUTOCOMPLETION

Rian Adam Rajagede, Kholid Haryono and Rizan Qardafil

(Received: 7-Dec.-2022, Revised: 1-Feb.-2023 and 22-Feb.-2023, Accepted: 5-Mar.-2023)

ABSTRACT

Attending lectures is a common way to learn Islamic knowledge. The speaker talks in front of the forum and participants take notes on the lecture material. Many participants listen to the lecture while taking notes either in books or on other digital devices to avoid forgetting the discussed topics. However, note-taking during the lecture can be challenging, with no complementing module from the speaker. Lecturers have different paces and varying ways of delivering. In addition, sometimes, participants cannot always focus during the lecture. Those factors can cause problems in the note-taking process: some details can be lost or even shift the meaning. For note-taking on sensitive topics, such as verses from the Quran, the note-taking process must be done carefully and avoid mistakes. In this study, we proposed an autocomplete system for the Indonesian translation of the Quran that will help the user in note-taking in Islamic lectures. The user writes down words, the parts of the Quran verse that he/she hears and the system will retrieve the most similar verses. With semantic retrieval, the user does not need to write down the exact words of the verses he/she heard. The system can also handle typographical-errors that usually occur in note-taking. We use FastText and calculate the cosine distance between the query and verses for the retrieval process. We also performed several optimization steps to create a robust system for the production stage. The system is evaluated by comparing how close the returned verse is with the ground truth. The proposed method's result in terms of accuracy reached 70.59% for the top 5 retrieved verses and 76.47% for the top 10 retrieved verses.

KEYWORDS

Semantic retrieval, Quran auto-completion.

1. INTRODUCTION

Islamic knowledge is commonly communicated through lectures, in which the speaker engages and informs a large group of listeners. This type of Islamic lecture is fairly common in Muslim-majority countries, such as Indonesia. Islamic lectures are held in mosques across the country. Because these lectures are given in a variety of settings, they are frequently delivered without supporting materials, such as presentation slides or audience modules. Most of the time, lecturers give content from their notes, Islamic textbooks or from the holy book Quran. As a result, some people listen to lectures while taking notes on books or other digital devices to help them remember the information. However, without the use of a module, the process of taking notes on Islamic lectures could be more difficult. Some participants might be unable to maintain their concentration throughout the lecture or some lecturers might deliver the content at a fast pace, which could affect the difficulty of taking notes. Particularly, for participants who took notes using mobile devices, in general, people were slower to take notes through mobile applications than through writing by hand on paper [1].

This situation can affect participants to overlook some important aspects or make some errors when taking notes. It is quite risky when it comes to noting the snippets of Quran verses; errors in note-taking can lead to different meanings or different verses. Another strategy is only to write the surah's number and the verse number. However, it will make the notes become difficult to read. One solution that can be used is to create a system that is able to suggest the suitable choice of Quran verses based on user notes.

We can formulate this problem as an information-retrieval problem in the domain of Quran verses. In this research, we tackle this problem by creating an auto-completion system in a mobile note-taking application. In the auto-completion system, when the user types in the application, the system typically suggests a word or sentence before the user completes it. Instead of getting a word or sentence recommendation, we can suggest Quran verses during user typing. The user's incomplete sentence can be seen as a query and the system will retrieve recommended verses as a suggestion to complete the

R. A. Rajagede, K. Haryono and R. Qardafil are with Department of Informatics, Universitas Islam Indonesia, Sleman, Indonesia. Emails: rian.adam@uii.ac.id, kholid.haryono@uii.ac.id and rizan.qardafil@students.uii.ac.id

note.

The creation of an information retrieval system for the Quran or Hadith has previously been researched with several different approaches [2]–[4]. Most of the methods used in that research rely on stemming algorithms, such as the Nazief-Adriani algorithm [4]-[5] or ECS (Enhance Confix Stripping) [6]. Other studies uses corpus and thesaurus to ensure input from the user is in the database [2]-[3]. However, those approaches will be difficult to use with note-taking in Islamic lectures. This is because when note-taking, users can make writing errors caused by several factors that have been mentioned previously. These writing errors can bring up words that do not exist in the database. Those words are commonly known as Out-of-Vocabulary (OOV) words. Two approaches can be taken to overcome OOV errors: using edit distance [7] or semantic embedding [8]-[9].

In the edit-distance approach, the system measures word similarity if the word is not found in the database, while the semantic-embedding approach uses an additional algorithm to convert words into vectors, then compares the similarity of documents in vector space. Several methods that can be used to convert words into vectors are Word2Vec [10], FastText [11] and BERT [12]. In previous studies, researchers have compared Word2Vec and FastText for the Quran and Hadith [13]. In that research, Word2Vec outperformed FastText both intrinsically and extrinsically. However, among the three embedding methods previously mentioned, Word2Vec cannot handle OOV words, which is essential in auto-completion.

In this study, we propose a Quran auto-completion system that will be embedded in a mobile application for note-taking in Islamic lectures. This auto-completion system focuses on retrieving the Indonesian translation of Quran verses based on words that are fragments of the Quran's verse. The system will use FastText to get the semantic embedding of the user query, then look for the most suitable verse.

This paper will be organized as follows. We will briefly introduce FastText, Sentence Embedding and Compress FastText in Section 2. Section 3 outlines the methodology used in the study, including the data preparation, similarity search algorithm and evaluation methods. In Sections 4 and 5, we present the results of the research and discuss the findings. Section 4 will focus more on sentence-embedding model evaluation, while Section 5 will focus more on the retrieval-system evaluation. Finally, in Section 6, we provide concluding remarks and recommendations for further research.

2. RELATED WORKS

2.1 FastText

FastText is an open-source, free, lightweight library by Meta that allows users to represent words in a continuous vector space, called word embedding [11]. Word embedding uses machine learning, particularly neural networks, to convert the representation of a word into a vector. Words with similar meanings are close together in their vector space. For instance, the word "fun" will be closer to "joy" than "fan" even though it has similar letters. The use of word embedding as a feature has improved the performance of several models in many NLP tasks, such as sentiment analysis [14], sentence similarity [15], emotion classification [16] and essay scoring [17].

The key idea behind FastText is to represent words as a bag of character n-grams (sub-strings of length n) instead of using a single vector for each word. This approach helps in handling Out-Of-Vocabulary (OOV) words effectively, as these words can still be represented by the character n-grams even if they were not seen during training.

2.2 Sentence Embedding

Sentence embedding converts a sentence consisting of several words into a vector. This method expands word-embedding methods, such as Word2Vec or FastText, changing the representation from the word level to the sentence level.

Similar to word embedding, sentence embedding will make sentences with semantic similarities have closeness in their vector space. There are various techniques to obtain sentence-embedding vectors. One way is to calculate the average word vector in a sentence. This method is used in FastText. The disadvantage of this method is that it ignores the word order, which may give some contextual meaning. Another method is to map the entire sentence directly to get its embedding vector. This method is used

in BERT Sentence Embedding (SBERT) [18]. SBERT is an extension of the Bidirectional Encoder Representations from Transformers (BERT) [12] architecture that uses the Siamese network model. SBERT constructs vectors by paying attention to the word's context in the sentence. Previous studies have shown that this method is superior to FastText [19].

2.3 Compress FastText

One of the challenges in developing models for embedding is the model's size. The size of this model will affect the system at the production stage, because it must provide more extensive storage or memory capacity. Although large storage is easy to achieve in the production stage, large-memory requirements are still quite expensive. In previous studies, the Compress FastText model [20] has been developed, which has a much smaller size than the original size with a less significant decrease in performance. In this study, we focus on using a compress FastText model, which is more efficient in memory usage.

Compress FastText was developed by utilizing several optimization methods applied to the vocab and n-gram matrix of the original FastText model. In this study, we examine several optimization parameters and see the effect on the size and performance of the model. The first optimization method we tested was using half-precision, which changed the original FastText data type from initially used 32-bit floats to 16-bit floats. In addition, we reduce the size of the matrix by eliminating less-frequent words and n-grams. We will also compare it with the smallest model created by the Compress FastText author. The model is obtained with additional optimization methods, such as matrix factorization and product quantization.

3. METHODS

This research consisted of two major parts. The first part focuses on converting the query and the verses into embedding vectors, q and v. Then, at the next stage, a vector-similarity search is carried out between the query and all verses in the Quran. In general, the process of the system in this study is shown in Figure 1.

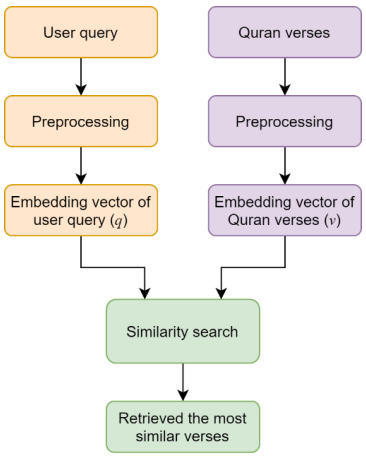


Figure 1. Steps to retrieve Quran verses in the aut-ocompletion system.

3.1 Pre-processing

In this stage, we pre-process the data before calculating the embedding vector, which includes case folding, stemming and stop-word removal. The data used in this study is a public Quran dataset taken from various sources and has been combined in a single JSON file [21]. The dataset contains data for the Arabic text of the Quran, transliteration, English and Indonesian translation of the Quran. This study only uses the Indonesian translation part for the retrieval process.

Each word in the database is processed through the following pre-processing stages:

1. Case folding; we change all letters into lowercase
2. Word stemming using the Nazief-Adriani algorithm [5] on PySastrawi [22]
3. Stop-word removal. Because currently, there is no publicly available data for this domain, we

choose the stop word based on word occurrences in the Quran. We try different threshold values and evaluate the model to the evaluation set. We explain this step later in Section 5.

3.2 Word and Sentence Embedding

For this research, we need to compute both word and sentence-embedding vectors of the user query and the Quran verses. We use the Compress FastText Model to calculate the word embedding for each query. To get the sentence embedding, first, we choose some words in the query and the Quran verses and calculate their word-embedding vectors. Then, we get the sentence-embedding vector by calculating the average of the word vectors. The pre-processing steps are performed before calculating the sentence vector query, as illustrated in Figure 2. Later in sub-section 3.3 will be explained how we select words for each query and verse to calculate sentence embedding.

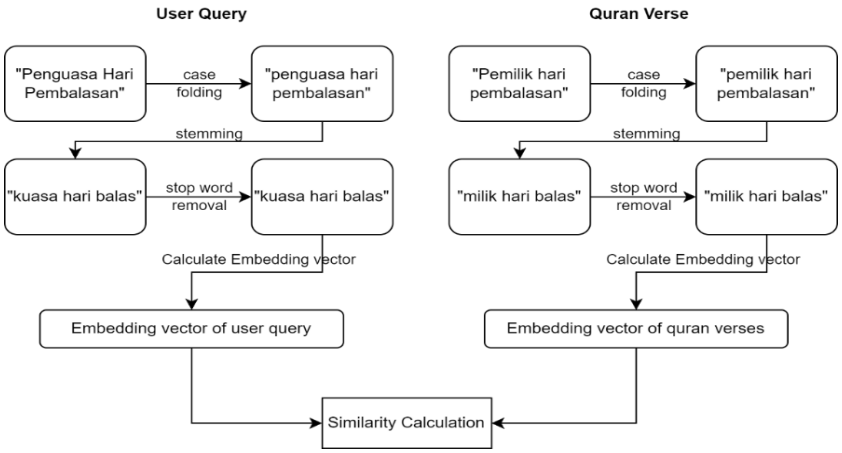


Figure 2. Similarity process using a single Indonesian verse (Quran 1:4) as an example.

To speed up the computing process, we store a dictionary file that contains all word embedding vectors of processed unique words in the Quran. This dictionary will help map a word with its vector representation faster. This additional data adds around 5 MB in size.

3.3 Similarity Search Algorithm

In this step, we measure the distance of the sentence vector of the user query (q) to the sentence vector of each verse (v) in the Quran. The smaller the distance between q and v^t, the more similar the meaning of the query to the t^{th} verse. The similarity is calculated using the cosine distance between vectors using:

$$\text{dist}(q, v^t) = \cos(\theta) = \frac{q \cdot v^t}{\|q\|\|v^t\|}$$

One characteristic of Quran retrieval is that the size of the search document is not too large and will not change over time. Therefore, we propose to combine sentence vector and word vector by aligning the words in the query and the words in the verse before averaging to get better performance.

We choose the corresponding word in the verse for each word in the query by calculating the distance between q_k and v_i^t, the word vectors of the k^{th} word in the query and the i^{th} word in the t^{th} verse. Then, the chosen words from the verse are averaged to get the sentence vector to represent the verse. This step is to avoid some words that may frequently appear in the verse, but not in the user query, affecting the average word vector of the verse. This method gives better results than directly calculating the distance between a sentence vector query and a verse. The explained retrieval algorithm can be seen in Table 1.

The drawback of this retrieval method is that the time complexity increases over the length of the query and the length of the verse. The additional step that iterates all words in the query for each verse increases the time complexity from $O(NM)$, where N is the number of verses and M is the length of the sentence vector into $O(NKLM)$, where K is the number of the word in the query and L is the number of the word in the verse. To address this problem, we speed up the distance calculation by leveraging parallelization calculation between cores using the Python multi-processing package. The processes at lines 3-8 from Table 1 will be computed in parallel.

Table 1. Similarity search algorithm.

Algorithm 1: Sentence + word vector retrieval

Input: query, verses

Output: List of distance *d* between query and verses

1: $q \leftarrow \frac{1}{|\text{query}|}\sum_k \text{word\_emb}(\text{query}_k)$ ▷ sentence embedding of the query

2: **for** t ← 1, |verses| **do** ▷ iterate for each verse in the Quran

3: **for** k ← 1, |query| **do** ▷ iterate for each word in the query

4: $q_k \leftarrow \text{word\_emb}(\text{query}_k)$ ▷ get the word embedding for the k[th] word in the query (q_k)

5: $l_k \leftarrow \underset{i=1,\ldots,|\text{verses}[t]|}{\text{argmin}} \text{dist}(q_k, v_i^t)$ ▷ calculate the distance between q_k and the word embedding for the i^{th} word in the t^{th} verses (v_i^t)

$$v_i^t \leftarrow \text{word\_emb}(\text{verses}_i^t)$$

▷ store the index of the closest word in the verse i to l_k

6: **end for**

7: $v^t \leftarrow \frac{1}{|\text{query}|}\sum_k \text{word\_emb}(v_{l_k}^t)$ ▷ sentence embedding of verses is calculated using only the closest word stored

8: $d^t \leftarrow \text{dist}(q, v^t)$ ▷ distance between sentence embedding of query and sentence embedding of verses

9: **end for**

3.4 Mobile Note-taking Application

As mentioned earlier, we can formulate this problem as an information-retrieval problem in the domain of Quran verses. However, in the user point-of-view, we format the retrieval process as an auto-completion system that suggests Quran verses during user typing. The user's incomplete sentence can be seen as a query and the system will retrieve recommended verses as a suggestion to complete the note.

We created a prototype mobile note-taking application based on Android to evaluate the system. Our models are stored on the virtual private server with 8 virtual CPU cores and communicated to the front-end *via* a Python API. The retrieval process is illustrated in Figure 3.

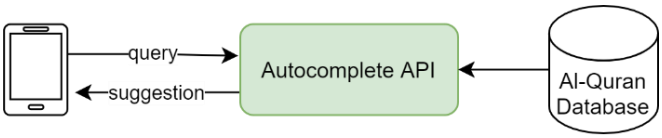


Figure 3. Illustration of auto-completion using API.

During writing in the note-taking application, the user starts writing a snippet of a verse using the "~" (tilde) symbol to trigger the auto-completion system. When the auto-completion system is triggered, the application will record what user type and use it as a retrieval query. After stopping typing for 1.5 seconds, the system will assume that the user completes the query and then the query will be sent to the API *via* the Internet. By using the algorithm presented in Table 1, the API will return ten recommended verses with high similarity with the query. We retrieved ten recommended verses to give users more options, as some verses may have high similarities.

3.5 Evaluation

In this study, we evaluate two parts. The first evaluation is the sentence-embedding model and the second part is the retrieval system. We dedicated Section 4 to explain in detail the evaluation steps of the embedding model.

For the retrieval process, we evaluate the retrieval system in two ways. The first method is done automatically by creating test data from the English translation of the Quran that is translated into Indonesian. We use English translation to create synthetic data that can be seen as a way to paraphrase the Indonesian translation of the Quran. This method is a common technique to enrich data for low-resource language tasks called back-translation [27]-[28]. The English translation was translated into Indonesian using Google Translate API.

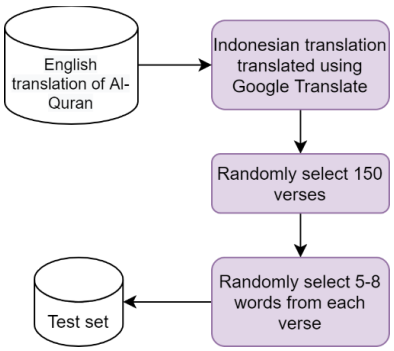


Figure 4. Steps to build test data to evaluate the retrieval system.

The test will be conducted by selecting 150 translated verses randomly, removing some words of each verse and then asking the system to retrieve the requested verses. The steps are illustrated in Figure 4.

The second evaluation method we do is to create test data containing Out-of-Vocabulary (OOV) words. This test data was made from the respondent's lecture notes, which contained verses from the Quran. Apart from lecture notes, this test data is also made by asking respondents who have memorized the Quran to recall the translation of a verse in the Quran and write it down.

We will compare the three retrieval methods at this evaluation stage. "Sentence + word vector" is the method we propose in Table 2. This method combines sentence and word-vector embedding. The second model we compared is "Sentence Vector" retrieval. This method will directly calculate the distance between the verse and query sentence vector and then return the verses that have the closest distance. The "Word matching" method is done by matching how many words in the query are found in the verse. The more query words are found in a verse, the higher the score returned for that verse. The details of the three models are shown in Table 2.

Table 2. Evaluated retrieval methods.

Model	Method
Sentence + vector retrieval	As explained in Table 1
Sentence retrieval	1. Calculate sentence vector for query and verse.
	2. Find the closest distance between query and verse.
Word matching	1. Count how many query words exist in each verse.
	2. Find the verse with the most number of query words found.

4. EMBEDDING MODEL EVALUATION

Before we implement the retrieval system, we need to evaluate the embedding model we will use in the system. In this section, we evaluated the FastText models after they had been compressed. It is worth noting that neither the FastText model nor Compress FastText used in this research is explicitly trained on Quranic verses. Therefore, we evaluated the model using general Indonesian text and expect that the model should give the best of its performance on the common words in the Indonesian language. We evaluated the sentence-embedding model both using intrinsic evaluation and extrinsic evaluation. In addition, we also evaluated the memory required by the system to run the model. The latter is quite essential for the production stage, as providing a system with a large RAM capacity is very expensive.

Intrinsic evaluation will measure how well the vector representation of the words generated by the model is. In this evaluation, we tested the aspect qualitatively by randomly taking Indonesian and Quran words and then looking at each model's five closest words in the vector space. For example, we try to find the

five closest words in the embedding space using the single word "rendang," which is the traditional food name in Indonesia. We evaluate those words qualitatively with the word "rendang."

For extrinsic evaluation, we also test the embedding-vector representation using an NLP task. The task that we chose was essay scoring using the Ukara dataset [24]. Ukara dataset is a dataset published by the Indonesian government for automatic essay scoring challenge for Indonesian language in 2019. The dataset contains student short answer and binary label for the correctness of the student answer. We evaluate the classifier performance when using embedding vector generated by the sentence-embedding model. We chose this task as the best model in that challenge is to use the FastText embedding model for Indonesian language.

Three models that compress FastText produced are compared here. Table 4 presents the compared FastText models as well as the optimization parameters that were utilized. Compress Model 1 is a model generated using default settings of the Compress Fasttext model [20]. We developed Compress Model 2 and Compress Model 3 by modifying the original FastText model for Indonesian language [23] using the parameters set up in Table 3 to compress the model size. We chose different vocab sizes and n-gram sizes to evaluate their effects on the model performance. The difference between the three models, apart from the vocab size and n-gram size, is that Compress Model 1 uses a matrix-quantization technique to reduce the model's size as suggested in [20]. Compared to the original model, compressed models use 16-bit floating points instead of 32-bit floating points.

Table 3. FastText and Compress FastText models comparison.

Model	Floating points	Matrix quant.	Vocab size	n-gram size
FastText Original Model [23]	32-bit	No	2,000,000	2,000,000
Compress Model 1	16-bit	Yes	20,000	100,000
Compress Model 2	16-bit	No	100,000	100,000
Compress Model 3	16-bit	No	100,000	300,000

The first thing that we look at is how much space the model takes up in storage, as well as the amount of memory that is required to transform words into vectors. The filprofiler package[1] is used to calculate the amount of memory capacity that is being used. Table 4 presents the findings of the study. We also compared the FastText compressed models with the SBERT model that was used in previous research [19].

Table 4. Storage and memory usage by FastText models.

Method	Storage (MB)	Memory (MB)
SBERT [19]	1075	2389
FastText Original Model [23]	6902	14484
Compress Model 1	14	77
Compress Model 2	117	254
Compress Model 3	233	384

4.1 Intrinsic Evaluation for Compressed Model

We also compared the quality of the four FastText models using intrinsic evaluation. Models retrieve the closest or similar words in their vector representation to the query words. We find the closest words using the most_similar() function from the Gensim package for FastText. This function will return the closest words whose distance is calculated using the cosine distance.

The model is tested by providing several Indonesian words and Quran words from different domains and assessing the returned words (human judgment). We use the same evaluation scheme from previous research, including how to calculate the model accuracy [13] by testing whether the most similar word returned is a synonym, antonym, related word or derivative form of the query. In this experiment, we found that the quality of the words returned by Compress Model 1 is sometimes very irrelevant to the

[1] https://pythonspeed.com/fil/

given query.

In Table 5, we show the query results on the word “Rendang”, one of Indonesia's popular foods. We chose the word “rendang” as illustration in Table 5, as that word is more common than the Quranic word. However, we evaluate the model using the Quranic word, as shown in Table 6. From the table, it can be seen that Compress Model 1 returns numerous words that are not categorized as synonyms, antonyms, related words or derived words to the given query (red text).

Table 5. Top-5 similar words for each model.

Method	Word query	Top-5 similar words	Category
FastText Original	Rendang (*Food name*)	- rendang (*lowercase of Rendang*)	Synonym
		- Kalio (*half-cooked Rendang*)	Related
		- Rendan (*mistype of Rendang*)	Synonym
		- Dendeng (*food from the same origin city of Rendang*)	Related
		- rendangnya (his/her/their rendang)	Derived
Compress Model 1	Rendang (*Food name*)	- Padang (*Padang City; Rendang origin city*)	Related
		- Ini (*this*)	Uncategorized
		- Dari (*from*)	Uncategorized
		- Tersebut (*the*)	Uncategorized
		- Dan (*and*)	Uncategorized
Compress Model 2	Rendang (*Food name*)	- rendang (*lowercase of Rendang*)	Synonym
		- Dendeng (*food from the same origin city of Rendang*)	Related
		- Balado (*food from the same origin city of Rendang*)	Related
		- Pindang (*food from the same origin city of Rendang*)	Related
		- Gulai (*food from the same origin city of Rendang*)	Related
Compress Model 3	Rendang (*Food name*)	- rendang (*lowercase of Rendang*)	Synonym
		- Dendeng (*food from the same origin city of Rendang*)	Related
		- Balado (*food from the same origin city of Rendang*)	Related
		- Pindang (*food from the same origin city of Rendang*)	Related
		- Gulai (*food from the same origin city of Rendang*)	Related

This also applies to several other words. We select 10 words that appear quite frequently in the Indonesian translation of Quran, then we calculate the accuracy of each model in returning the most similar words of the given word. The ten selected words and their accuracy results for each model are shown in Table 6 and Table 7, respectively.

Table 6. Selected frequent words from Indonesian translation of Quran for intrinsic evaluation.

ALLAH	BUMI	AZAB	KAUM	KITAB
IMAN	LANGIT	KAFIR	ZALIM	HATI

Table 7. FastText model’s accuracy returning most similar words.

Method	Accuracy (%)
FastText Original Model [23]	86.00
Compress Model 1	82.00
Compress Model 2	**88.00**
Compress Model 3	**88.00**

4.2 Extrinsic Evaluation for Compressed Model

For extrinsic evaluation, we evaluated the compressed models in a publicly available NLP task for the Indonesian language. Currently, there is no publicly available task specifically using Indonesian Quran translation. Therefore we chose Ukara Automatic Essay Scoring task [24] to evaluate compressed models. We used the same dataset, data split and experiment scheme as previous research [19]. We use Compress FastText models as a feature extractor for essay sentences before being inputted into the Neural Network with a single hidden layer. The hyper-parameter search process uses Optuna [25] and we also include the regularization technique proposed in the study, increasing batch [26].

Ukara dataset consists of two sets of problems. In this study, we only use data A. The evaluation results of Ukara's data A are shown in Table 8.

Table 8. Model performance on automatic essay scoring task.

Method	F1-Score
SBERT [14]	89.44
FastText Original	88.82
Compress Model 1	88.63
Compress Model 2	88.75
Compress Model 3	88.89

In Table 9, it appears that, in general, the FastText model is still not as good as SBERT for the case of Automatic Essay Scoring using a simple model. However, from the table, it appears that Compress Model 3 has the best results compared to other FastText models, although the differences are not so significant. This also indicates that the FastText model compression technique can maintain a good performance.

Based on the evaluations that have been conducted, in the next stage, we only use Compress Model 3 as the embedding model to change the representation of words in the query.

5. RESULTS AND DISCUSSION

5.1 Retrieval-system Evaluation

This evaluation focuses on evaluating the success rate of verse retrieval of the auto-completion system. The system is evaluated using two sets of data tests. Each data test contains pairs of a query and an intended verse that need to be retrieved. The first data test is synthetic data generated using the English translation of the Quran, as mentioned in sub-section 3.5. The second data test is related to data that contains Out-of-Vocabulary (OOV) words that we built by ourselves.

As mentioned, we try different stop-word threshold values in these experiments. We use retrieval evaluation on English Quran translation data to evaluate the stop-word threshold. We got 1000 occurrences as the best stop-word threshold based on model-retrieval performance on top-5 accuracy and top-10 accuracy. Results in the later part use 1000 occurrences as the stop-word threshold.

5.1.1 Retrieval Evaluation on English Quran Translation Data

For the first data test, we build it by randomly choosing 150 verses from the English translation of the Quran. Then, we translate them into Indonesian using the Google Translate API. From those 150 verses, we will randomly select 5-8 words that are not included in the stop-word list as a query. The system will return the ten verses closest to the given query. The illustration of this dataset processing and samples can be seen in Table 9.

We calculate the system's accuracy by checking whether the intended or correct verse is in the ten retrieved verses (top-10 accuracy), as given in the query. If the correct verse is one of the ten verses, the system is considered successful in retrieving the verse. In addition to top-10 accuracy, we also conduct the same evaluation to find top-5 accuracy. Top-5 accuracy is calculated when the correct verse is one of the five retrieved verses. The results of this evaluation are shown in Table 10.

Table 9. Sample queries of data test obtained from the English translation. Red-colored words are new words that do not appear in the original Indonesian translation.

Original Indonesian Translation	Original English Translation	English Translation Translated into Indonesian	Randomly Picking 5-8 Words
Kami menjadikan Al-Qur'an dalam bahasa Arab agar kamu mengerti.	Indeed, We have made it an Arabic Qur'an that you might understand.	Sesungguhnya Kami telah menjadikannya Al-Qur'an berbahasa Arab agar kamu mengerti.	Sesungguhnya Kami Al-Qur'an berbahasa Arab mengerti.
dengan membawa gelas, cerek dan sloki (piala) berisi minuman yang diambil dari air yang mengalir	With vessels, pitchers and a cup [of wine] from a flowing spring	Dengan bejana, kendi dan secangkir [anggur] dari mata air yang mengalir	bejana, secangkir [anggur] air mengalir
Wahai manusia! Kamulah yang memerlukan Allah; dan Allah Dialah Yang Mahakaya (tidak memerlukan sesuatu), Maha Terpuji.	O mankind, you are those in need of Allah, while Allah is the Free of need, the Praiseworthy	Wahai manusia, kamu adalah orang-orang yang membutuhkan Allah, sedangkan Allah adalah Yang Maha Bebas lagi Maha Terpuji	Wahai orang-orang membutuhkan Allah Maha Bebas Maha Terpuji.

Table 10. Top-5 accuracy and Top-10 accuracy for each method on test set built from English translation.

Method	Top-5	Top-10
Sentence + word vector	**55.33%**	**64.67%**
Sentence vector	38.67%	51.33%
Word matching	54.67%	62.67%

From Table 10, it appears that the use of word vectors to select the most relevant words prior to calculating the sentence vector affects the quality of the model's performance. Sentence method + word vector retrieval outperforms other methods.

The retrieval method with word matching achieved a fairly high score in Table 10 compared to the sentence vector method, which is almost similar to that in our proposed method. However, the evaluation process with the English translation scheme does not exploit the advantages of vector embedding in dealing with out-of-vocabulary words (OOV). Therefore, we tested these three methods in dealing with OOV cases by building a special test data that also includes OOV words, either synonyms, informal abbreviations or typos.

5.1.2 Retrieval Evaluation on OOV Data

This special data test that includes OOV words is used as the second part of system evaluation. This data test was created from two main sources. The first source is written lecture notes. The lecture notes are collected from several respondents who had taken notes during the Islamic lectures. We copied the part of the lecture notes that included a verse quote along with the letter number and verse number. The second source is obtained from the recall of several respondents who memorized the Quran. We asked them to remember some verses of the Indonesian translation and write down the verses. Verse paraphrasing, word abbreviations and writing errors usually occur during recall. In the case of the Quran note-taking, the process of OOV emergence is unavoidable. Some examples of data tests are shown in Table 11. The test data is publicly available and can be downloaded at https://bit.ly/iqrtest (4^{th} revision).

We use the test set for the three models and the results are shown in Table 12. In this table, it appears that the sentence + word vector embedding method is far outperforming word-matching retrieval. The sentence vector method produces the lowest performance. This can happen, because some quite long verses cause the average word vector for the entire verse not to represent the verse.

Table 11. Example of test data. The part highlighted in red is an example of OOV, because it is different or not in the original verse. The number next to it corresponds with the word in the original verse closest to it.

Original verse	Query
Dan (ingatlah) ketika **Lukman[1] berkata[2]** kepada anaknya, ketika dia memberi pelajaran kepadanya, ”Wahai anakku! Janganlah engkau mempersekutukan Allah, sesungguhnya mempersekutukan (Allah) adalah benar-benar kezaliman yang besar.”	Luqman[1] menasehati[2] anaknya
Wahai manusia! Sungguh, Kami telah menciptakan kamu dari seorang laki-laki dan seorang perempuan, kemudian Kami jadikan kamu **berbangsa-bangsa**[3] dan bersuku-suku agar kamu saling mengenal. Sesungguhnya yang paling mulia di antara kamu di sisi Allah ialah orang yang paling bertakwa. Sungguh, Allah Maha Mengetahui, Mahateliti.	manusia diciptakan berbeda-beda[3] untuk saling mengenal
Wahai orang-orang yang beriman! Bertakwalah **kepada**[4] Allah sebenar-benar takwa kepada-Nya dan janganlah kamu mati kecuali **dalam keadaan**[5] Muslim.	bertakwa kpd[4] Allah dan jangan mati selain mjd[5] muslim
Katakanlah, "Wahai hamba-hamba-Ku yang melampaui batas terhadap diri mereka sendiri! **Janganlah**[6] kamu berputus asa **dari rahmat**[7] Allah. Sesungguhnya Allah mengampuni dosa-dosa semuanya. Sungguh, Dialah Yang Maha Pengampun, Maha Penyayang.	jgn[6] putus asa utk meminta pd[7] Allah

Table 12. Top-5 accuracy and Top-10 accuracy for each method on test-set that included OOV.

Method	Top-5	Top-10
Sentence + word vector	**70.59%**	**76.47%**
Sentence vector	35.29%	47.06%
Word matching	55.89%	58.82%

5.2 Note-taking Application Prototype

The interface of the note-taking application prototype looks like in Figure 5. In Figure 5-left, users write queries starting with a tilde and choose a suggested verse from the top of the space. In Figure 5-right, after choosing a verse, it will automatically replace the query in the body of the note. Using a server with 8 virtual cores, the average retrieval takes about 2.2 seconds.

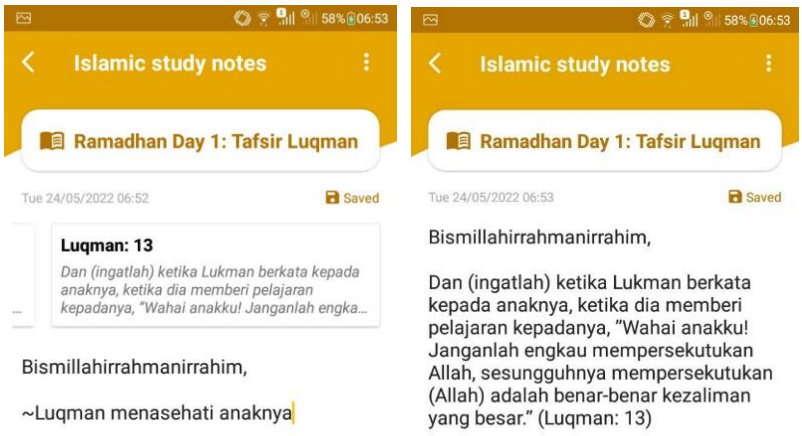


Figure 5. The interface when the user writes a query and gets recommendation (left). The interface when the recommendation auto-completes the query (right).

We tested this prototype on 32 respondents, consisting of Islamic lecturers and participants. The result showed that 96.9% of participants believe that smart Quran auto-completion in the note-taking

application can help the user in the note-taking process.

6. CONCLUSIONS

In this study, we propose an auto-completion system using semantic vector representation. To obtain vector embedding, we use Compress FastText, which is more efficient in storage and memory usage, so it is suitable to be used at the production stage. In the retrieval process, we propose a combination of word and sentence-vector embedding and cosine distance when searching for verses. This method proved to be more effective in the two types of evaluation performed compared to other methods.

In the future, much can still be developed, especially in terms of reliability at the production stage. The auto-completion system demands a model that can process queries in real time. For this reason, it is necessary to efficiently use resources and optimize algorithms to get the expected system's performance. The lack of available data might also affect the current system's retrieval result. We can use the current prototype or other ways to gather more data that fits in the Quran-retrieval domain.

ACKNOWLEDGEMENTS

This research was supported by the Directorate of Research and Community Service at Universitas Islam Indonesia through beginner research grant number 001/Dir/DPPM/70/Pen.Pemula/III/2022

REFERENCES

[1] B. J. Lee, "Smartphone Tapping *vs.* Handwriting: A Comparison of Writing Medium," The EuroCALL Review, vol. 28, no. 1, p. 15, DOI: 10.4995/eurocall.2020.12036, 2020.

[2] Z. Abu Bakar and N. Abdul Rahman, "Evaluating the Effectiveness of Thesaurus and Stemming Methods in Retrieving Malay Translated Al-Quran Documents," Lecture Notes in Computer Science, vol. 2911. pp. 653–662, 2003, DOI: 10.1007/978-3-540-24594-0_67.

[3] A. Aulia, D. Khairani and N. Hakiem, "Development of a Retrieval System for Al Hadith in Bahasa (Case Study: Hadith Bukhari)," Proc. of the 5th Int. Conf. on Cyber and IT Service Management (CITSM 2017), DOI: 10.1109/CITSM.2017.8089323, 2017.

[4] I. Humaini, T. Yusnitasari, L. Wulandari, D. Ikasari and H. Dutt, "Informatian Retrieval of Indonesian Translated Version of Al Quran and Hadith Bukhori &Muslim," Proc. of the 2018 Int. Conf. on Sustainable Energy, Electronics and Computing System (SEEMS 2018), DOI: 10.1109/SEEMS.2018.8687330, 2019.

[5] M. Adriani, J. Asian, B. Nazief, S. M. M. Tahaghoghi and H. E. Williams, "Stemming Indonesian: A Confix-stripping Approach," ACM Transactions on Asian Language Information Processing, vol. 6, no. 4, pp. 1–33, DOI: 10.1145/1316457.1316459, 2007.

[6] A. Z. Arifin, I. P. A. D. Mahendra and H. T. Ciptaningtyas, "Enhanced Confix Stripping Stemmer and Ants Algorithm for Classfying News Document in Indonesian Language," Proc. of the 5th Int. Conf. on Information & Communi. Technology and Systems, no. April 2014, pp. 149–158, 2009.

[7] D. K. Po, "Similarity Based Information Retrieval Using Levenshtein Distance Algorithm," Int. J. of Advances in Scientific Research and Engineering, vol. 06, no. 04, pp. 06–10, 2020.

[8] S. Wang and R. Koopman, "Semantic Embedding for Information Retrieval," Proc. of CEUR Workshop, vol. 1823, pp. 122–132, 2017.

[9] Y. Yuan, Improving Information Retrieval by Semantic Embedding, B.Sc. Essay, Faculty of Electrical Engineering, Mathematics and Computer Science, University of Twente, Netherlands, [Online], Available: http://essay.utwente.nl/82070/, 2020.

[10] T. Mikolov, K. Chen, G. Corrado and J. Dean, "Efficient Estimation of Word Representations in Vector Space," Proc. of the 1st Int. Conf. on Learning Representations (ICLR 2013), arXiv: 1301.3781, DOI: 10.48550/arXiv.1301.3781, 2013.

[11] P. Bojanowski et al., "Enriching Word Vectors with Subword Information," Trans. of the Association for Computational Linguistics, vol. 5, pp. 135–146, DOI: 10.1162/tacl_a_00051, 2017.

[12] J. Devlin, M. W. Chang, K. Lee and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," Proc. of the 2019 Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT 2019), vol. 1, pp. 4171–4186, 2019.

[13] M. Zidny Naf'an, Y. Sari and Y. Suyanto, "Word Embeddings Evaluation on Indonesian Translation of AI-Quran and Hadiths," IOPscience, vol. 1077, no. 1, p. 012025, DOI: 10.1088/1757-899X/1077/1/012025, 2021.

[14] A. Aziz Altowayan and A. Elnagar, "Improving Arabic Sentiment Analysis with Sentiment-specific Embeddings," Proc. of the 2017 IEEE Int. Conf. on Big Data, vol. 2018-Jan., pp. 4314–4320, DOI:

10.1109/BigData.2017.8258460, 2017.
[15] F. Alam, M. Afzal and K. M. Malik, "Comparative Analysis of Semantic Similarity Techniques for Medical Text," Proc. of the Int. Conf. on Information Networking, vol. 2020-Jan., pp. 106–109, DOI: 10.1109/ICOIN48656.2020.9016574, 2020.
[16] M. S. Saputri, R. Mahendra and M. Adriani, "Emotion Classification on Indonesian Twitter Dataset," Proc. of the 2018 Int. Conf. on Asian Language Processing (IALP 2018), pp. 90–95, DOI: 10.1109/IALP.2018.8629262, 2019.
[17] R. A. Rajagede and R. P. Hastuti, "Stacking Neural Network Models for Automatic Short Answer Scoring," Proc. of 5th Int. Conf. on Information Technology and Digital Applications (ICITDA 2020), vol. 1077, pp. 0–6, Yogyakarta, Indonesia, 2020.
[18] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence Embeddings Using Siamese BERT-networks," Proc. of 2019 Conf. on Empirical Methods in Natural Language Processing and 9th Int. Joint Conf. on Natural Language Processing (EMNLP-IJCNLP 2019), DOI: 10.18653/v1/d19-1410, 2020.
[19] R. A. Rajagede, "Improving Automatic Essay Scoring for Indonesian Language Using Simpler Model and Richer Feature," Kinetik: Game Technology, Information System, Computer Network, Computing, Electronics and Control, vol. 6, no. 1, pp. 11–18, DOI: 10.22219/kinetik.v6i1.1196, 2021.
[20] D. Dale, "Compress-FastText," Github Repository, [Online], Available: https://github.com/avidale/compress-fasttext, Accessed: Jun. 01, 2022.
[21] G. Nasution, "Quran - API," Github Repository, [Online], Available: https://github.com/gadingnst/quran-api, 2022.
[22] H. A. Robbani, "Sastrawi Python," Github Repository, [Online], Available: https://github.com/har07/Py Sastrawi, 2018.
[23] E. Grave et al., "Learning Word Vectors for 157 Languages," Proc. of the 11th Int. Conf. on Language Resources and Evaluation (LREC 2018), pp. 3483–3487, Miyazaki, Japan, 2019.
[24] G. B. Herwanto et al., "UKARA: A Fast and Simple Automatic Short Answer Scoring System for Bahasa Indonesia," Proc. of the Int. Conf. on Educat. Assessment and Policy, DOI: 10.26499/iceap.v2i1.95, 2018.
[25] T. Akiba, S. Sano, T. Yanase, T. Ohta and M. Koyama, "Optuna: A Next-generation Hyper-parameter Optimization Framework," Proc. of the ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, DOI: 10.1145/3292500.3330701, 2019.
[26] S. L. Smith, P. J. Kindermans, C. Ying and Q. V. Le, "Don't Decay the Learning Rate, Increase the Batch Size," Proc. of the 6th Int. Conf. on Learning Representations (ICLR 2018), [Online], Available: https://openreview.net/forum?id=B1Yy1BxCZ, 2018.
[27] R. Sennrich, B. Haddow and A. Birch, "Improving Neural Machine Translation Models with Monolingual Data," Proc. of the 54th Annual Meeting of the Association for Computational Linguistics, pp. 86-96, Berlin, Germany, 2016.
[28] D. R. Beddiar, M. S. Jahan and M. Oussalah, "Data Expansion Using Back Translation and Paraphrasing for Hate Speech Detection," Online Social Networks and Media, vol. 24, Article no. 100153, 2021.

ملخص البحث:

في هذا البحث، يجري تصميم وتجربة نظام للاسترجاع الاوتوماتيكي للألفاظ والجُمل في الآيات القرآنية، بناءً على التّرجمة الإندونيسية للقرآن الكريم، من شأنه ان يساعد المشاركين في حضور المحاضرات لاكتساب المعرفة الاسلامية في تدوين ملاحظاتهم أثناء تلك المحاضرات. حيث يقوم المستخدم بكتابة كلماتٍ معينة أو أجزاء من الآيات الكريمة التي يستمع إليها، وبناءً على ذلك يعمل النّظام على استرجاع الآيات القرآنية الأقرب الى تلك التي قام المستخدم بادخالها الى النّظام. في هذه الدراسة، استخدمنا نموذج (FastText) وقمْنا بحساب مسافة جيب التّمام بين الطّلب المقدّم من المستخدم والآيات القرآنية من أجل عملية الاسترجاع. كذلك قمْنا بخطواتٍ لجعل النّظام مثالياً، وأجرينا تقييماً للنّظام من خلال فحْص مدى قُرب النّتائج من الواقع الفعلي. وبلغت الدّقّة التي حصلنا عليها (70.59%) في حالة أوّل (5) آيات مُسترجَعة، و (76.47%) في حالة أوّل (10) آيات مُسترجَعة.

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 09, No. 02, June 2023.

DESIGN OF A COMPACT BROADBAND ANTENNA USING CHARACTERISTIC MODE ANALYSIS FOR MICROWAVE APPLICATIONS

Ahmad Abbas Al Rimi[1]*, Asmaa Zugari[1], Aicha Mchbal[1], Mohssine El Ouahabi[2] and Mohsine Khalladi[1]

(Received: 28-Dec.-2022, Revised: 1-Mar.-2023, Accepted: 20-Mar.-2023)

ABSTRACT

A compact broadband antenna of dimensions 27 mm × 28 mm × 1.6 mm and with good impedance matching is designed for high-bandwidth radio systems with a short range. To improve the impedance matching, two rectangular slots are created on the radiating element and the ground plane size is reduced to extend the ultra-wideband frequency band. The antenna bandwidth and radiation performance are analyzed using characteristic mode theory (TCM). The performance is compared to the desired specifications and the shape and size are modified to produce efficient radiation and dominant radiation patterns. The findings clearly demonstrate that the six modes are resonant with (λ_n=0). This implies that the eigenvalues of the six modes contribute strongly to dominant electromagnetic radiation and have high modal significance values around 1 at their respective frequencies. Furthermore, the characteristic angle indicates that the antenna resonates at 180°, since the six modes intersect the axis line at 180° at their respective frequencies. Experimental results show a bandwidth of 109.7% between 5.64 and 19.34 GHz, a maximum gain of 6.3 dB and a maximum efficiency of approximately 86.5%. These results make this antenna a versatile and effective choice for a wide variety of communications and electronic applications and easy to install in narrow spaces due to its easy design characteristics, small size and light weight.

KEYWORDS

Eigen currents, Eigenvalues, Characteristic modes, Satellite application, Ultra-wideband.

1. INTRODUCTION

Wireless, mobile and satellite communication technologies have developed quickly and the ability for data mobility, accessibility and the exchanging of high-quality data at high speeds between portable devices has grown incredibly quickly. In addition, as computer networks have grown in popularity, a diversity of electronic and handheld devices, as well as radio applications, have undergone a huge evolution, which continues the need to increase and enhance the bandwidth needs of measuring systems, since new specifications for measurements beyond 1 GHz are being developed for a number of standards around the world [1]. In 2002, the Federal Communications Commission (FCC) allocated a range of radio frequencies from 3.1 GHz to 10.6 GHz for unlicensed ultra-wideband (UWB) systems [2]. Since then, the practical design and implementation of a UWB system have become a challenge in academia and the telecommunications industry. The design of a UWB antenna faces challenges in terms of broadband performance, small size, radiation stability, manufacturing ease and low cost. The use of broadband antennas in systems is more practical and with the aim of enhancing antenna bandwidth, several designs have been made, including ultra-wideband antennas in the shapes of rectangles [3], patch antenna with tapered oval apertures [4], a square shape with slots serrated [5], a pyramidal horn [6], a serrated wheel [2], a rhombus-shaped [7], a triangular-shaped [8], fractal-shaped O-ring antenna [9], patch antenna grid-slotted [10], H-shaped microstrip patch antenna [11] and other shapes [12]-[13]. However, most of the relevent literature lacks a CMA analysis.

1. A. Al Rimi (Corresponding Author), A. Zugari, A. Mchbal and M. Khalladi are with Information Systems and Telecommunications Lab., Uni. of Abdelmalek Essaadi, Faculty of Science Tetuan, Morocco. Emails: alrimiahmed75@gmail.com, asmaa.zugari@gmail.com, aicha.mchbal8@gmail.com and m_khall@hotmail.com
2. M. El Ouahabi is with National School of Applied Sciences, Uni. of Abdelmalek Essaadi, Morocco. Email: moelouahabi@uae.ac.ma

For designing any antenna and determining the resonance in free space or analyzing the performance of an antenna, CMA is regarded as a basic approach that involves calculating the field radiated and the surface current. This theory is grounded on the MoM (Method of Moments) to solve the complete set of Maxwell's equations and evaluate the antenna performance based on eigenvalues [23]. In 1965, Garbacz suggested CMA analysis for the wire antenna. This technology was expanded for solid conducting structures by Harrington and Mautz [24]. Several more extensions of the theory were later developed by Harrington, Mautz and their students [25]-[26]. Although some researchers published their works late last century, approximately 42 years went by without the entire potential of CM being acknowledged. In 2007, the capabilities of CM were revisited by summarizing research that authors had conducted over the past several years to show that the Theory of Characteristic Modes may be utilized to execute a methodical design of various kinds of antennas [27]. Since then, interest in the theory of characteristic mode analysis has increased dramatically and the number of important articles and applications is a good indicator of the subsequent boom. Overall, characteristic analysis is a valuable tool for understanding the behavior of antennas and ensuring that they are designed for optimal performance. Micro-strip antennas represent a promising candidate in radio- communication systems because of their reduced cost, light weight and good performance. However, the small bandwidth of these antennas frequently restricts their range of use. By altering the radiating patch geometry using the etching technique, enhancing the impedance bandwidth of this kind of antenna becomes possible. In this paper, an original patch antenna is designed and investigated. The design of this latter is carried out using two rectangular slots together with a reduced-size ground plane in order to extend the UWB operating spectrum while keeping a low return loss. The analysis of the studied antenna performance is conducted utilizing the theory of characteristic modes, which is based on eigenvalues and eigen- current representation. The following sections provide a study of the design process and a discussion of the measured results.

2. THEORY OF CHARACTERISTIC MODES

The characteristic mode is the orthogonal current mode on an equipotential surface. CMA is a method for calculating the current-distribution modes of any conductor in a perfect vacuum with no feed port. Model significance, eigenvalues, characteristic angle, field radiation and surface current give a global insight into various resonances and modes in order to determine the dominant resonating modes of a specific antenna.

The antenna current distribution overlaps in various modes in CMT applications and each mode is featured by the modal weighting coefficient (distinctive angle α_n) and the radiation efficiency that the eigenvalue (λ_n) indicates. The real current density J on the antenna equipotential surface, as according to CMT, is written as a rectilinear combination of characteristic mode currents J_n:

$$J = \sum_n (J_n \alpha_n) \tag{1}$$

The eigenvalue (λ_n) describes the proportion of reactive power to radiative power. If $\lambda_n = 0$, the mode radiates well at this frequency. The characteristic current distribution modes are determined from the generalized eigenvalue formula, which is given as:

$$X(J_n) = \lambda_n R(J_n) \tag{2}$$

where R and X stand for the real and imaginary parts of the generalized impedance operator Z = R+jX and J_n represents the characteristic currents or eigenvectors which are solutions of the global eigenvalue formula. When the value of λ_n takes a very high value *vs.* the frequency plot, the eigenvalues cannot be used to categorize the dominant mode of the antenna. The characteristic angle (α_n) can be used to tackle this problem, which is given by the subsequent expression:

$$\alpha_n = 180 - tan^{-1}(\lambda_n) \tag{3}$$

Each mode contribution to the overall electromagnetic reaction to a certain source is measured by the modal significance (MS_n), which is defined by the subsequent expression:

$$MS_n = \left| \frac{1}{1 + j\lambda_n} \right| \tag{4}$$

The resonance point of MS typically has a value of one "1," while the modes that do not contribute to the

resonance have a value of zero "0." and the characteristic angle (α_n) has a 180◦ phase difference for the resonating mode; therefore, λ_n is equal to zero.

3. CHARACTERISTIC MODE PERFORMANCE ANALYSIS

In this part, we analyze the performance of a broadband antenna using the theory of characteristic modes. Knowing that the notion of characteristic mode is based on the study of the radiating element without a structure, it is simply dependent on the size and form of the conducting item without the feeding port, in order to analyze and predict its electromagnetic behavior. This analysis allows for the computation of the antenna electrical properties, such as its resonant frequency and impedance, without the need for a physical structure to be built. This can save time and resources, as well as provide insight into the optimal design for the antenna. This analysis can also be used to identify potential issues with the design and optimize the antenna for specific performance objectives. The eigenvalue gives insight into the nature of the mode. If λ_n=0, it means that the mode has an effective radiation. The goal is to achieve the large-scale performance of the proposed antenna that can be achieved through various excitation modes in a simultaneous way. Here are the steps for analyzing antenna performance based on characteristic modes (CMs): The first step is to create the antenna model, choose the materials and determine the physical structure (shape and size) of the antenna. The second step is to identify and analyze the resonance modes, calculate the resonance patterns of the antenna to determine its characteristics and radiation pattern and evaluate the antenna performance (the first six modes were studied).

Finally, the modified antenna design is validated using simulations and experimental measurements to ensure that it meets the required performance criteria. These steps contributed to improving the design and choosing the appropriate shape and size. Figure 1 represents the eigenvalue for the first six modes. In fact, the selection of the first six modes in a characteristic mode analysis is typically based on practical considerations, such as computation time and accuracy. These first six modes provide a good approximation of the antenna behavior in most cases and are often sufficient for many design and analysis purposes. In general, using more modes in the characteristic mode analysis can provide results that are more accurate, but it also increases computation time and complexity. It is noted that the first six modes are often a trade-off between accuracy and computational efficiency. Additionally, in many cases, the higher-order modes may not significantly affect the overall behavior of the antenna, making the first six modes a sufficient representation of the antenna behavior. So, as we can see in Figure 1, the results clearly show that the six modes are resonant with ($\lambda_n = 0$). It is noted that the eigenvalues of the six modes contribute to dominant electromagnetic radiation strongly at their own frequencies. They offer resonances simultaneously at 5.83 GHz, 7.79 GHz, 10.1 GHz, 10.54 GHz, 12.75 GHz and 16.58 GHz. Also, a comprehension of the EM properties of the antenna could be possible from eigenvalues without taking the absorbed power and current into account.

Figure 2 displays the graph of the frequency of the first six modes *versus* the model significance and characteristic angle. From Figure 2 (a), it is observed that mode-1, mode-2, mode-3, mode 4, mode-5 and mode-6 have large modal significance values around 1 at resonant frequencies of 5.83 GHz, 7.79 GHz, 10.1 GHz, 10.54 GHz, 12.75 GHz and 16.58 GHz, respectively. The ($\alpha_n = 180^o$) characteristic angle associated to these modes indicates that the antenna accumulates magnetic energy (inductive) when $\alpha_n < 180^0$ and when α_n>180°, it accumulates electric charge (capacitive). As shown in Figure 2 (b), it is clear that modes (1, 2, 3, 4, 5 and 6) intersect the 180-degree axis line at 5.83 GHz, 7.79 GHz, 10.1GHz, 10.54 GHz, 12.75 GHz and 16.58 GHz, respectively.

Figure 3 shows the modes of surface currents and the 3-D farfield pattern for the six modes of the studied antenna at their respective resonance frequencies without a feed port. The current distribution over the ground plane and radiator produces a mix of successive pairs of even and odd modes from orthogonal currents. This simultaneous excitement of the odd and even modes provides the broadband radiation behaviour of a certain antenna. Also, it can be seen from farfield presentation that all CMs are varied with respect to the resonant frequency and that all modes emit radiation in all directions at their respective resonance frequencies.

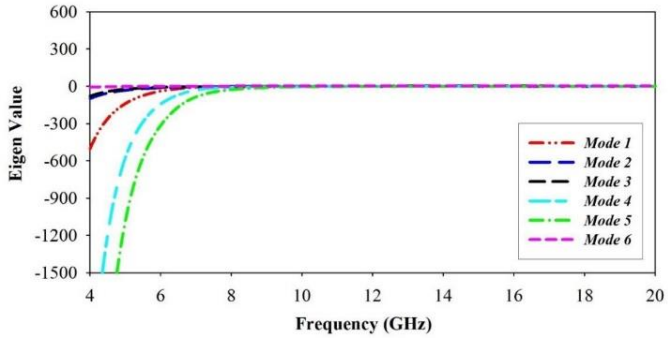
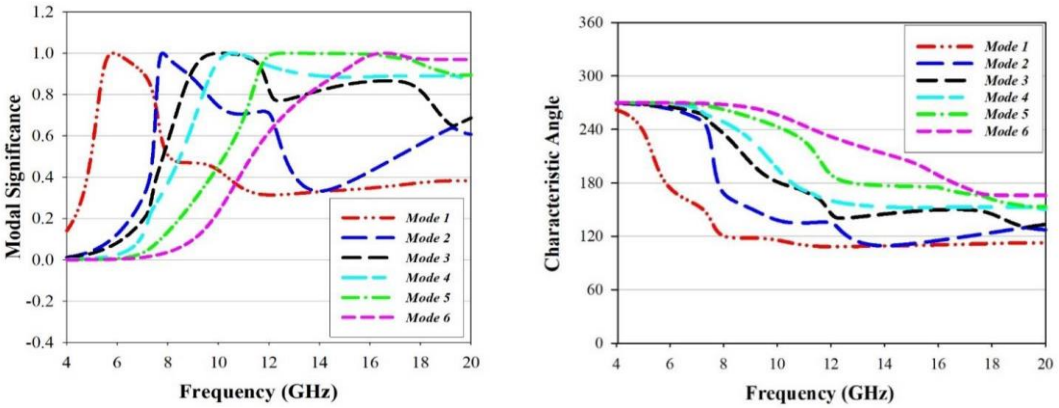


Figure 1. Eigenvalues as a function of frequency of the first six modes.



(a) (b)

Figure 2. (a) Model significance, (b) Characteristic angle of the first six modes.

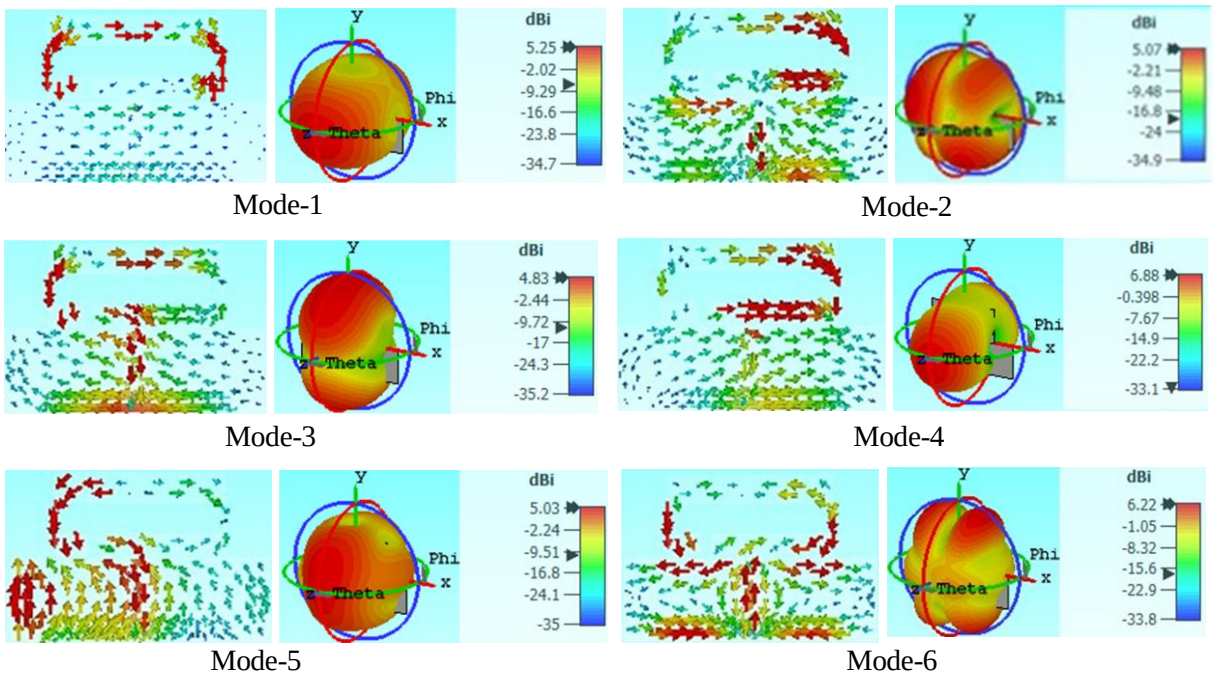


Figure 3. Surface current & 3-D farfield pattern at resonant CM frequencies.

4. ANTENNA DESIGN

In this section, the design process of the presented patch antenna is discussed. The antenna is mounted on a 28 x 27 x 1.6 mm^3 FR-4 substrate with a dielectric constant of 4.4 and a dielectric loss of 0.025, on which a 50Ω matched feedline with a length of L_f = 9.2 mm and a width of W_f = 3.14 mm is designed on the top face. Note that the reason behind the choice of the used laminate is its low profile characteristic. Figure 4 exhibits the final design of the proposed antenna.

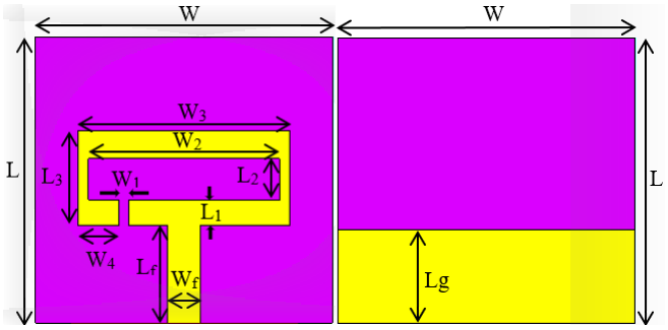


Figure 4. Final antenna geometry, (a) Front side, (b) Back side.

After a parametric study using CST Microwave Studio (MWS), the optimized parameters of the proposed UWB antenna are presented in Table 1.

Table 1. Final antenna dimensions.

Parameter	W	L	W_f	L_f	W_1	W_2	W_3	W_4	L_1	L_2	L_3	Lg	h
Value (mm)	28	27	3.1	9.2	1.	18	18.5	3.82	2.4	3.3	9	9	1.6

4.1 Design Evolution

As we can see from Figure 6, which depicts the simulated return loss at different steps of the presented antenna design, by utilizing a reduced ground plane in the second step, impedance matching has been enhanced at the lower and upper ranges of the operating spectrum, which has led to the manifestation of two resonant frequencies at 8 GHz and 18.5 GHz. Next, by moving to the third step that is based on forming a rectangular slot (L_2 x W_2) on the radiating element, two resonant frequencies that correspond to 14.8 GHz and 17.5 GHz have been produced. Finally, by taking the fourth step, which is also based on employing one more rectangular open-ended slot (L_1 x W_1) on the patch, three resonant frequencies have appeared at 6.2 GHz, 14.6 GHz and 17.2 GHz, with bandwidths of 1 GHz, 5.2 GHz and 3 GHz, respectively.

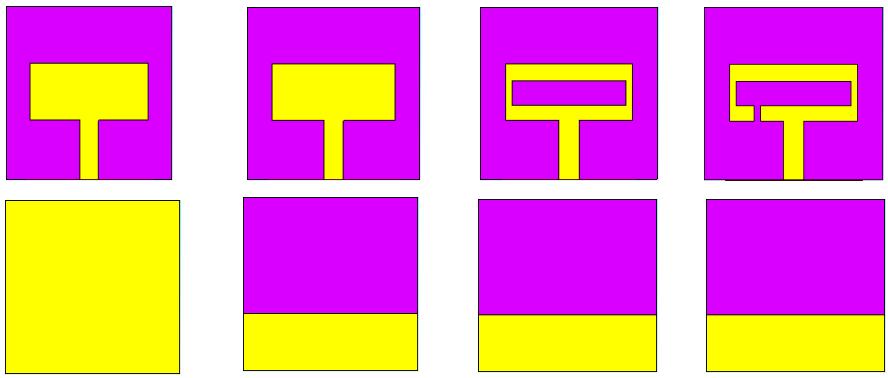


Figure 5. The proposed antenna design evolution.

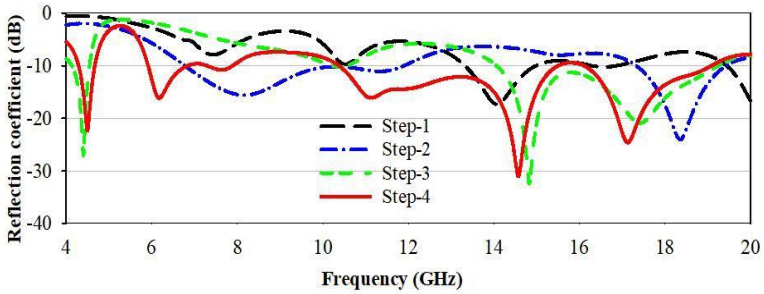
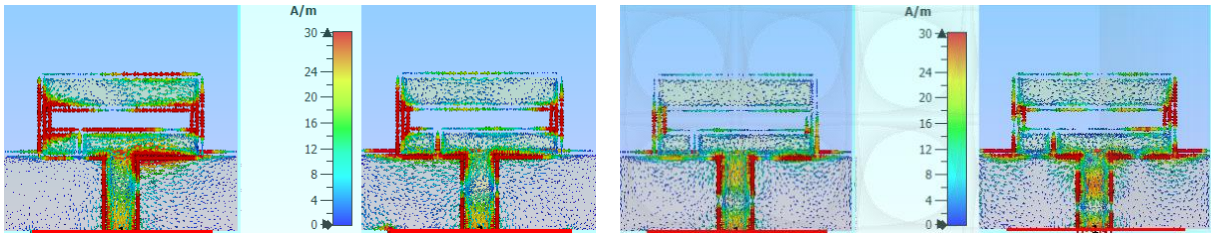


Figure 6. Simulated reflection coefficient for all different steps of the design.

4.2 Current Distribution

Figure 7 (a–d) shows the surface current of the presented antenna. It is obvious from this figure that the current propagates in all directions and is mainly concentrated at the feed point. Also, a significant amount of current is flowing around the edges of the openings L_1 x W_1 and L_2 x W_2, since the aperture acts as a slot that allows the electric field to penetrate into the antenna and create the surface current that flows around the edges of the aperture and generates the electromagnetic field necessary for radiation or reception. Finally, at higher frequencies, it is clear that the surface current is more concentrated near the edges of the antenna.



(a) (b) (c) (d)

Figure 7. Current distribution at (a) 5.8 GHz, (b), 7.44 GHz, (c) 11 GHz and (d) 14.48 GHz.

5. PARAMETRIC STUDY

5.1 L_1 and L_2 Effect

Figure 8 shows the simulated reflection coefficients for different values of L_1 and L_2. It is obvious that as L_1 decreases, the bandwidth decreases, noting that the optimal value of L_1 is 2.4 mm. Also, the rise in the L_2 value causes a decrease in the bandwidth, noting also that the optimal value of L_2 is 3.3 mm.

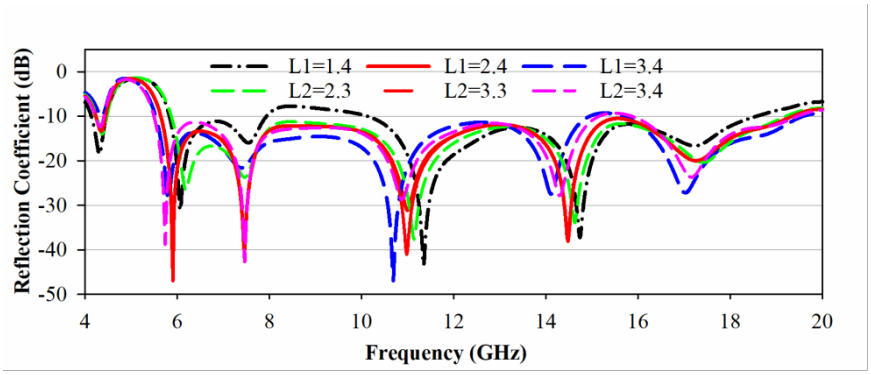


Figure 8. Simulated S_{11} with varying L_1 and L_2.

5.2 W_1 and W_2 Effect

Figure 9 shows the simulation reflection coefficients for different values of W_1 and W_2. It is clear that a

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 09, No. 02, June 2023.

rise or decrease in W_1 produces a decrease in bandwidth, knowing that W_1 = 1 mm is the optimal value. On the other hand, the rise in the W_2 value causes a decrease in the bandwidth, knowing also that the optimum value of W_2 is 18 mm.

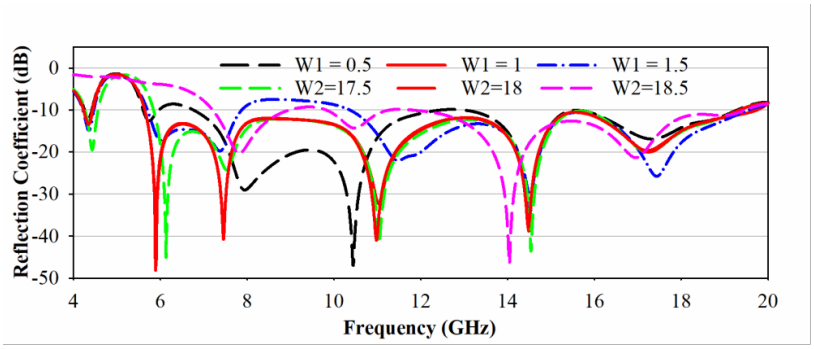


Figure 9. Simulated S_{11} with varying W_1 and W_2.

5.3 Lg Effect

Figure 10 shows the simulation reflection coefficients for various values of Lg. The value of Lg was changed from 8 mm to 10 mm. It is clear that an increase or decrease in the value of Lg generates a decrease in bandwidth, noting that the optimum value for Lg is 9 mm.

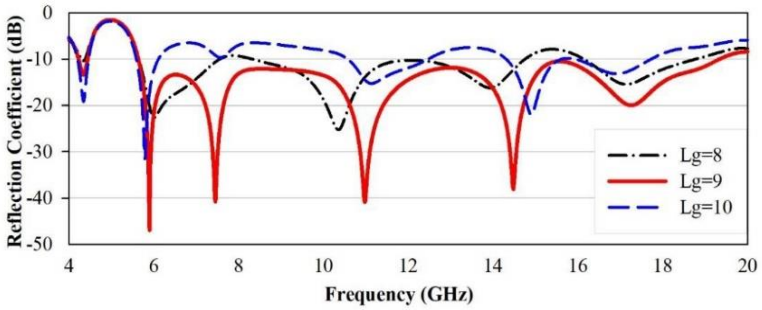
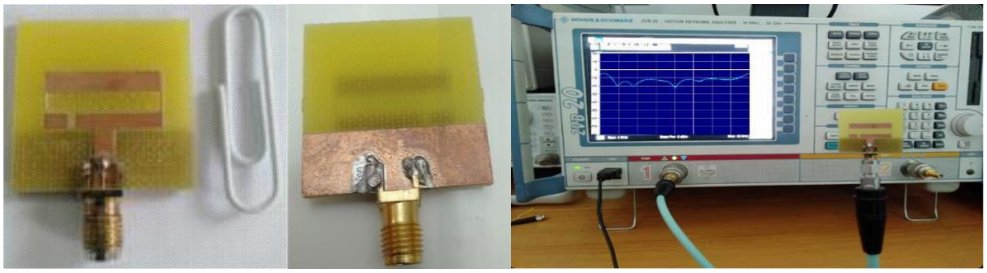


Figure 10. Simulated S_{11} with varying Lg.

6. Simulation Results and Discussion

For experimental validation and according to the optimized dimensions specified in Table 1, a prototype of the suggested antenna has been fabricated on an FR-4 substrate using the circuit-board plotter LPKF Protomat (as seen in Figure 11), while the measurement was carried out utilizing the ZVB20 vector network analyzer. Figure 12 illustrates the simulated and measured s-parameter. The experimental results indicate that the proposed antenna provides a good impedance bandwidth that equals 13.8 GHz from 5.2 GHz to 19 GHz. Note that there is a good agreement between measured and simulated results, with the exception of a slight difference caused by the lossy nature of the used laminate FR-4, SMA connections and soldering.



(a) (b)

Figure 11. (a) Front and back view of the fabricated antenna, (b) measurement setup.

6.1 Gain

The measured and simulated gain and efficiency are shown in Figure 13. As we can see from this figure, this antenna provides a good gain that ranges from 3.55 dBi to 6.3 dBi, noting that its maximum gain is attained at 9.6 GHz. Additionally, we can say that this antenna is highly efficient, since it has attained a maximum efficiency of about 86.8 % at 9 GHz while maintaining an efficiency of at least 73.8 % over all the frequency band of interest.

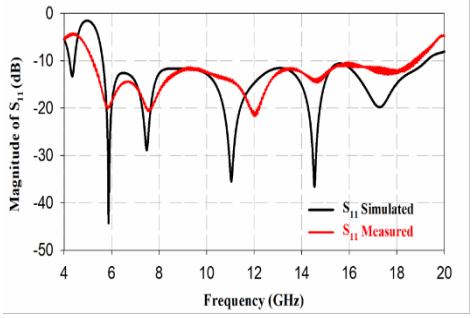
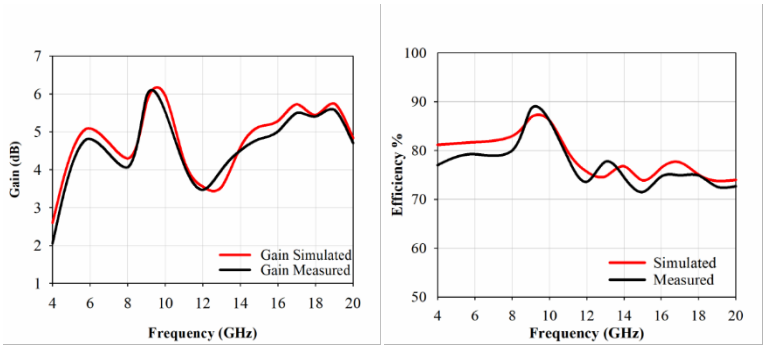


Figure 12. Measured and simulated magnitude of S_{11}.

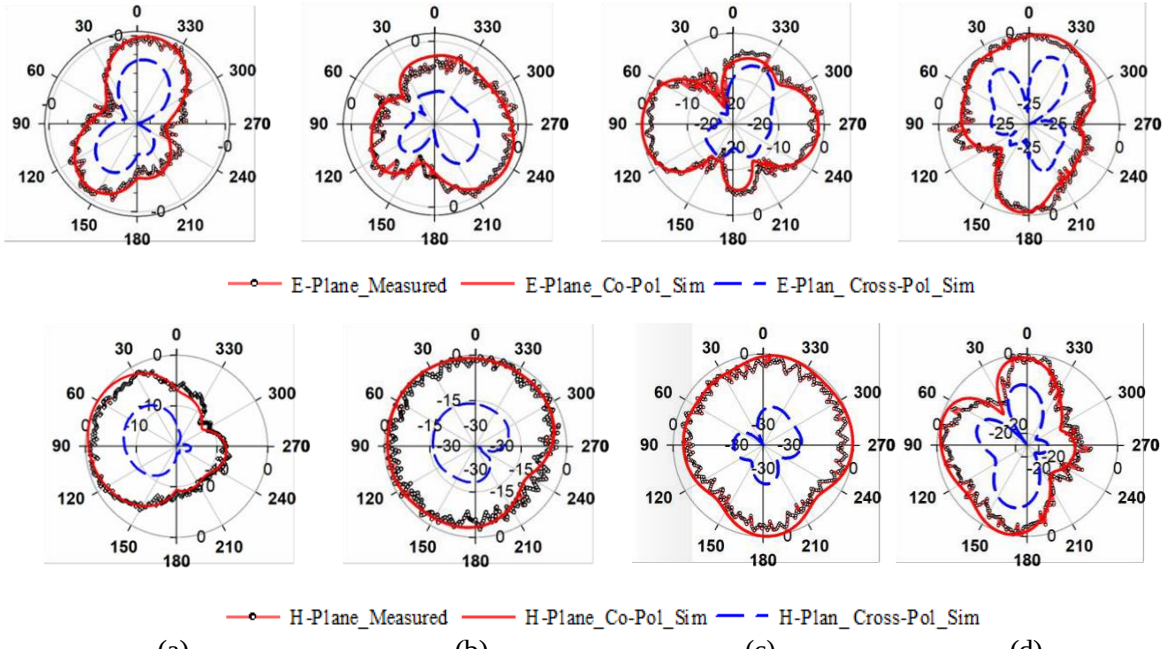


(a) (b)

Figure 13. The measured and simulated (a) peak gain, (b) efficiency.

6.2 Farfield Pattern

The measured and simulated E-plane and H-plane radiation patterns are depicted in Figure 14 (a–d), respectively. Due to the large bandwidth, only four frequencies are presented: 5.8 GHz, 7.44 GHz, 11 GHz and 14.48 GHz. So, from these diagrams, it can be seen that the proposed antenna is characterized by a quasi-omnidirectional radiation pattern in the H-plane. The radiation behavior of the antenna presented in the E-plane is similar to that of a dipole. Note that at 14.48 GHz, the 2D farfield patterns are less omnidirectional because of the higher-order resonances. Figure 15 displays the Geozondas antenna measurement system that was utilized. In order to determine the antenna radiation pattern, a horn antenna was utilized as the transmitting antenna and the receiving antenna was the antenna being tested.



(a) (b) (c) (d)

Figure 14. Measured and simulated E-plane & H-plane farfield patterns at (a) 5.8 GHz, (b) 7.44 GHz, (c) 11 GHz and (d) 14.48 GHz.

7. COMPARISON

Table 2 illustrates the performance comparison of the presented antenna with those of antennas presented in recently reported works based on bandwidth, gain, size and efficiency. Based on this comparison, it is clear that the studied UWB antenna is characterized by its small size, high gain, high efficiency and wide impedance bandwidth compared to those introduced in other works, which proves the presented antenna efficiency.

Figure 15. The proposed antenna measurement setup.

Table 2. Comparison between the proposed work and earlier works.

Ref.	Freq. Band	Gain (dBi)	Overall size (mm³) (λ^3)	Efficiency (%)	Operating band (GHz)
[13]	9.4	3.8– ~5.8	50×50×1.52 1.13λ_0×1.13λ_0×0.034λ_0	88–98	(2.1–11.5)138%
[14]	12.1	4.85 max	32×32×1.6 0.95λ_0×0.95λ_0×0.048λ_0	70–79.2	(2.9–15)135%
[15]	11.2	2.1–6.15	27 × 36 × 1.6 0.76λ0×1.01λ0×0.045λ0	- - -	(2.8-14)133%
[16]	2.59	-2–6.3	38.88 × 39.8 × 0.762 0.46λ_0×0.47λ_0×0.009λ_0	- - -	(2.26 – 4.85)72%
[17]	2.36	5.26 max	97 × 76 × 0.7 1.33λ0×1.04λ0×0.010λ0	- - -	(2.94–5.3)57.3%
[18]	7.5	7 – 10	100 × 45 × 0.5 2.42λ_0×1.09λ_0×0.012λ_0	- - -	(3.5 – 11)103%
[19]	2.6	4.64 max	30 ×30×0.8 0.39λ_0×0.39λ_0×0.010λ_0	- - -	(2.61–5.21)66.5%
[20]	2.6	3.65 max	54 × 54 × 1.6 0.52λ_0×0.52λ_0×0.015λ_0	- - -	(1.6 – 4.2)89.7%
[21]	4.22	5.32 max	50 × 50 × 1.6 0.76λ_0×0.76λ_0×0.024λ_0	88.5	(2.48 – 6.7)91.9%
[22]	0.98	2.68 max	50 x 55x 1.6 0.43λ_0×0.47λ_0×0.014λ_0	- - -	(2.08–3.06)38.1%
This work	**13.7**	**3.55 – 6.3**	**27 x 28 x 1.6** 1.12λ_0×1.16λ_0×0.066λ_0	**73.8–86.8**	**(5.64-19.34)109.7%**

NOTE: Due to the absence of measurement results for certain antennas, all values are from simulation results. Calculation of the electrical volume has been carried out at the center of the bandwidth.

8. CONCLUSION

In this work, a compact broadband (UWB) antenna is studied and analyzed by applying the theory of characteristic modes (TCM). The results of the simulation and measurement showed good performance

of the broadband antenna behaviour of about 109.7% (5.64–19.34 GHz) and 114% (5.2–19 GHz), a high efficiency of about 86.8% and a maximum gain of 6.3 dBi. These results prove that this antenna is a great candidate for UWB applications and it can also be used in radio systems such as WLAN (5.15–5.90 GHz), ISM (5.725–6.875 GHz), mobile satellite applications (7.250–7.375 GHz), mobile applications (8.020–8.200 GHz), broadcasting satellites (12.4–12.5 GHz) and define systems (14.62–15.23 GHz).

ACKNOWLEDGEMENTS

The authors would like to thank Professor Alia ZAKRITI of ENSA, the National School of Applied Sciences, Abdelmalek Essaadi University, Tetuan, Morocco, for his support with regards to the fabrication of the proposed antenna.

REFERENCES

[1] M. Windler and D. G. Camell, "Measuring Antennas above 1 GHz," Proc. of the 14th Intl. Electromagnetic Compatibility (EMC) Symposium, NIST, pp. 91–95, Feb. 2001.

[2] M. Gupta and V. Mathur, "Wheel Shaped Modified Fractal Antenna Realization for Wireless Communications," AEU - Int. J. Electronics and Communications, vol. 79, pp. 257–266, Sep. 2017.

[3] A. A. Deshmukh and P. V. Mohadikar, "Modified Rectangular Shape Patch Antennas for Ultra-wide Band and Notch Characteristics Response," Microwave and Optical Technol. Letters, vol. 59, no. 7, pp. 1524–1529, 2017.

[4] A. Raza, W. Lin, Y. Chen et al., "Wideband Tapered Slot Antenna for Applications in Ground Penetrating Radar," Microwave and Optical Technol. Letters, vol. 62, no. 7, pp. 2562– 2568, 2020.

[5] Y. Zehforoosh, M. Mohammadifar, A. Houshang Mohammadifar and S. R. Ebadzadeh, "Designing Four Notched Bands Microstrip Antenna for UWB Applications, Assessed by Analytic Hierarchy Process Method," J. Microwave, Optoelectronics and Electromagnetic Appl., vol. 16, no. 3, pp. 765–776, Sep. 2017.

[6] H. Du, X. Yu, H. Zhang and P. Chen, "Design of Broadband and Dual-polarized Dielectric-filled Pyramidal Horn Antenna Based on Substrate-integrated Waveguide," Microwave and Optical Technology Letters, vol. 61, no. 3, pp. 702-708, 2019.

[7] A. Chaabane and M. Guerroui, "Printed UWB Rhombus Shaped Antenna for GPR Applications," Iranian J. of Electrical and Electronics Eng., vol. 17, no. 4, pp. 2041–2041, Dec. 2021.

[8] B. Bhandari, A. Kumar, T. Giri and D. Shah, "Design of CPW-fed Triangular Shaped UWB Antenna for Multiband Applications," Int. Research J. of Eng. and Tech. (IRJET), vol. 4, no. 7, pp. 2123-2128, Jul. 2017.

[9] C.-J. Wu, I.-F. Chen, C.-M. Peng, W.-Y. Tsai and J.-S. Sun, "A Compact Fractal-shaped O-ring Monopole Antenna for Modern Broadband Wireless Applications," WSEAS Trans. Electron., vol. 12, pp. 93–99, 2021.

[10] K. R. Xiang and F.C. Chen "A Method for Increasing the Bandwidth of Slot and Patch Antennas Using Grid-slotted Patch," Int. J. of RF Microwave Computer-aided Engineering, vol. 31, no. 2, p. e22510, 2021.

[11] J.-Y. Deng et al., "Bandwidth and Gain Improvements of Low-profile H-shaped Microstrip Patch Antenna under Quadruple-mode Resonance," Int. J. RF Microw. Comp.-Aided Eng., vol. 30, no. 10, p. e22372, 2020.

[12] N. M. Awad and M. K. Abdelazeez, "Multi-slot Micro-strip Antenna for Ultra-wide Band Applications," J. of King Saud Uni. - Engineering Sciences, vol. 30, no. 1, pp. 38–45, Jan. 2018.

[13] P. M. Paul, K. Kandasamy, M. S. Sharawi and B. Majumder, "Dispersion-engineered Transmission Line Loaded Slot Antenna for UWB Applications," IEEE Antennas and Wireless Propagation Letters, vol. 18, no. 2, pp. 323–327, Feb. 2019.

[14] S. Mohandoss et al., "Fractal Based Ultra-wideband Antenna Development for Wireless Personal Area Communication Applications," AEU-Int. J. Electronics and Communications, vol. 93, pp. 95–102, Sep. 2018.

[15] H. S. Mewara et al., "A Printed Monopole Ellipzoidal UWB Antenna with Four Band Rejection Characteristics," AEU-Int. J. Electronics and Communications, vol. 83, pp. 222– 232, Jan. 2018.

[16] U. Ullah and S. Koziel, "A Novel Coplanar-strip-based Excitation Technique for Design of Broadband Circularly Polarization Antennas with Wide 3 dB Axial Ratio Beamwidth," IEEE Trans. on Antennas and Propagation, vol. 67, no. 6, pp. 4224–4229, Jun. 2019.

[17] W. An, S. Li, W. Sun and Y. Li, "Low-profile Wideband Microstrip Antenna Based on Multiple Modes with Partial Apertures," IEEE Antennas and Wireless Propagation Letters, vol. 18, no. 7, pp. 1372- 1376, 2019.

[18] Y. Sun and H. Jin, "Design of Broadband Endfire Antenna with Split-ring Resonator (SRR) Structures," Proc. of the 2019 IEEE Int. Conf. on Computational Electromagnetics (ICCEM), pp. 1–3, DOI: 10.1109/COMPEM.2019.8779224, Shanghai, China, Mar. 2019,

[19] R. S. Daniel, "Broadband μ-negative Antenna Using ELC Unit Cell," AEU-Int. J. Electronics and Communications, vol. 118, p. 153147, May 2020.

[20] M. Moore, Z. Iqbal and S. Lim, "A Size-reduced, Broadband, Bidirectional, Circularly Polarized Antenna for Potential Application in WLAN, WiMAX, 4G and 5G Frequency Bands," Progress In Electromagnetics Research C, vol. 114, pp. 1–11, Jan. 2021.

[21] O. Benkhadda et al., "Compact Broadband Antenna with Vicsek Fractal Slots for WLAN and WiMAX Applications," Applied Sciences, vol. 12, no. 3, Article no. 3, Jan. 2022.

[22] C. Güler, S. E. B. Keskin and R. B.aymaz, "The Development of Broadband Microstrip Patch Antenna for Wireless Applications," Bitlis Eren University J. of Science, vol. 11, no. 3, Article no. 3, Sep. 2022.

[23] J. Budania, M. Sharma, R. Yadav and R. Dhara, "Characteristic Mode Analysis and Design of Broadband Circularly Polarized CPW-fed Compact Printed Square Slot Antenna," Progress in Electromagnetics Research M, vol. 94, pp. 105-118, 2020.

[24] R. Harrington and J. Mautz, "Theory of Characteristic Modes for Conducting Bodies," IEEE Trans. on Antennas and Propagation, vol. 19, no. 5, pp. 622–628, Sep. 1971.

[25] Y. Chang and R. Harrington, "A Surface Formulation for Characteristic Modes of Material Bodies," IEEE Trans. on Antennas and Propagation, vol. 25, no. 6, pp. 789–795, Nov. 1977.

[26] R. F. Harrington and J. R. Mautz, "Characteristic Modes for Aperture Problems," IEEE Trans. on Microwave Theory and Techniques, vol. 33, no. 6, pp. 500–505, Jun. 1985.

[27] M. Cabedo-Fabres, E. Antonino-Daviu, A. Valero-Nogueira and M. F. Bataller, "The Theory of Characteristic Modes Revisited: A Contribution to the Design of Antennas for Modern Applications," IEEE Antennas and Propagation Magazine, vol. 49, no. 5, pp. 52–68, Oct. 2007.

[28] Y. Wu, H. Lin, J. Xiong, J. Hou, R. Zhou, F. Deng and R. Tang, "A Broadband Metamaterial Absorber Design Using Characteristic Modes Analysis," Journal of Applied Physics, vol. 129, no. 13, p. 134902, 2021.

[29] F. H. Lin and Z. N. Chen, "Low-profile Wideband Metasurface Antennas Using Characteristic Mode Analysis," IEEE Transactions on Antennas and Propagation, vol. 65, no. 4, pp. 1706-1713, 2017.

[30] T. Li and Z. N. Chen, "Wideband Sidelobe-level Reduced *Ka*-Band Metasurface Antenna Array Fed by Substrate-integrated Gap Waveguide Using Characteristic Mode Analysis," IEEE Transactions on Antennas and Propagation, vol. 68, no. 3, pp. 1356-1365, 2020.

ملخص البحث:

تـمّ تصـميم هـوائي صـغير الحجـم عـريض النّطـاق أبعـاده 27مـم×28مـم×1.6مـم ولـه مواءمـة ممانعـة جيـدة للأنظمـة الرّاديويـة عريضـة النّطـاق ذات المـدى القصـير. ولتحسـين مواءمـة الممانعـة، تـمّ إنشـاء شَـقّيْن مسـتطيلين علـى عنصـر الإشـعاع مـع تصـغير الحجـم فـي المسـتوى الأرضـي مـن أجـل توسـيع النّطـاق التـردّدي فـائق العـرْض للهـوائي. وقـد جـرى تحليـل عـرض النّطاق والأداء الإشعاعي للهوائي باستخدام نظرية الأنماط المميّزة (TCM).

تمّـت مقارنـة أداء الهـوائي بالمواصـفات المرغوبـة وكـذلك تعـديل الشّـكل والحجـم لإنتـاج إشـعاعٍ فعّـال ونمـط إشـعاعٍ سـائد. وكشـفت النتـائج أن الأنمـاط السّـتّة كانـت فـي حالـة رنـين. وهـذا يتضـمّن أن القـيم الذّاتيـة للأنمـاط السّـتّة تسـهم بقـوّة فـي الإشـعاع الكهرومغناطيسـي السّـائد وأنهـا تمتلـك قِـيَم أهميـة تقـارب (1) عنـد تـردّدات الـرّنين الخاصّـة بهـا. عـلاوة علـى ذلـك، تبـيّن مـن الزاويـة المميّـزة أنّ الهـوائي فـي حالـة رنـين عنـد (180) درجـة؛ لأنّ الأنمـاط السّـتّة تقطـع خـطّ المحـور عنـد 180 درجـة عنـد تـرددات الـرّنين الخاصّـة بهـا. وتبـيّن النتـائج التّجريبيـة نطاقـاً عرضـه (109.7%) بـين (5.64) جيجـاهيرتز و (19.34) جيجـاهيرتز، كمـا بينـت أنّ الكسـب الأعلـى للهـوائي بلـغ (6.3) ديسـيبل، بفاعليـة اقتربـت مـن (86.5%). وهـذه النتـائج تجعـل مـن الهـوائي الـذي تـم دراسـته فـي هـذا البحـث خيـاراً فعّـالاً لتشـكيلةٍ واسـعةٍ مـن تطبيقـات الاتصـالات والتّطبيقـات الإلكترونيـة، عـلاوةً علـى أنـه سَـهل التّركيـب فـي الأماكن الضّيّقة بفعل خصائصه التّصميمية البسيطة وحجمه الصغير ووزنه الخفيف.

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 09, No. 02, June 2023.

AUTOMATIC DETECTION OF PNEUMONIA USING CONCATENATED CONVOLUTIONAL NEURAL NETWORK

Ahmad T. Al-Taani and Ishraq T. Al-Dagamseh

(Received: 12-Dec.-2022, Revised: 8-Mar.-2023 and 14-Mar.-2023, Accepted: 30-Mar.-2023)

ABSTRACT

Pneumonia is a life-threatening disease and early detection can save lives. Many automated systems have contributed to the detection of this disease and currently, deep-learning models have become among the most widely used models for building these systems. In this study, two deep-learning models are combined: DenseNet169 and pre-activation ResNet models and used for automatic detection of pneumonia. Two methods are used to deal with the problem of unbalanced data: class weight, which enables to control the percentage of data to be used from the original data for each class of data, while the other method is resampling, in which modified images are produced with an equal distribution using data augmentation. The performance of the proposed model is evaluated using a balanced dataset that consists of 5856 images. Achieved results were promising compared to those obtained by several previous studies. The model achieved a precision value of 98%, an area under curve (AUC) based on ROC of 97% and a loss value of 0.23.

KEYWORDS

Deep learning, Convolutional neural networks, Chest X-rays, Pneumonia, Pre-activation ResNet, DenseNet169.

1. INTRODUCTION

Pneumonia is a disease that targets the lungs. The cause of the disease, whether it is a virus, bacteria or fungus [1], seeks to damage one or both lungs. According to a report issued by the World Health Organization (WHO), it has been indicated that pneumonia is one of the most serious life-threatening diseases, especially for children under the age of five, as it kills about 1.2 million children of this age category around the world [2]. Other statistics have shown how dangerous and threatening pneumonia can be to a person's life. For example, a study indicated that there are about 4.5 million people infected with pneumonia around the world each year and more than 50,000 people die as a result of this disease in the USA each year [3]. In another study in 2012, it was indicated that there were about 28,952 deaths in the UK that were caused by pneumonia that year. In two statistical studies in 2016, the first concluded that about 880,000 children under the age of two died as a result of this disease [4], while the second in the same year indicated that Indian reports indicated that the highest death rates around the world were due to pneumonia with about 158,176 deaths in that year [5]. According to a report by the WHO, one out of every three deaths is due to pneumonia in India [6]. Figure 1 shows three X-ray images representing a normal, bacterial pneumonia and viral pneumonia images.

Figure 1. Normal, bacterial and viral pneumonia images [17].

From Figure 1, we can clarify the difference between the images, where in the normal chest image in the left, specifically by focusing on the lungs area, which appears in a black color, the presence of gases and their normal exchange are expressed. The middle image expresses a lung infected with pneumonia of a bacterial type and the affected area is the one to which the two arrows point which

A. T. Al-Taani and I. T. Al-Dagamseh are with Department of Computer Science, Yarmouk University, Irbid, Jordan. E-mails: ahmadta@yu.edu.jo and ishraq500@gmail.com

appears as a blurring or gray area. The image in the right shows two lungs infected with a viral type of pneumonia which appears in a hazy color. The difference between the bacterial and viral types is that the bacterial type infects one side of the lungs, while the viral one may infect both lungs.

Radiologists focus on identifying white spots and foggy or blurry areas to locate the injury, but the fact that the X-ray images of the chest are in black and white causes another challenge for the diagnostician to make a proper diagnosis of the disease [7]. This needs experience and accuracy. Another problem is that the symptoms of this disease overlap which the symptoms of other diseases such as lung cancer, which appears as opacity in the picture [8]. Therefore, there is a need to build automated systems capable of diagnosing this disease with greater accuracy and less time. The stages of developed algorithms and methods used for diagnosing pneumonia can be divided into three main stages: the oldest stage which started in the nineties used textual analysis to detect the CXR images [9]-[10]. As for the intermediate stage, it started with the emergence of machine-learning algorithms like support vector machine (SVM), k-nearest neighbor (KNN), Naïve bayes (NB) and random forest (RF) [11]-[12]. Research in the current period; i.e., the third stage, focused on the use of deep-learning models, specifically convolutional neural network (CNN) models, such as VGGNetm ExNet and GoogleNet [13] and ResNet models, like ResNet-38, ResNet-50 and Resnet-101 [3], [14]-[15]. Deep-learning models are mostly used for feature extraction and some focus on identifying regions of interest in the image; i.e., identifying the most important and effective area in diagnosing the disease [8].

In this paper, we proposed a new model consisting of two integrated deep-learning models for the phase of extracting features from CXR images to automatically diagnose pneumonia and those models are: pre-activation ResNet and DensNet169. After a critical analysis of the literature, we have deduced that the pre-activation ResNet model has achieved better results than the ResNet model in the medical field for instance, in [16], it obtained efficient results for the detection of prostate cancer. We have also found that those two models are the most common models used by different researchers for solving similar problems in the medicine field like Alzheimer disease [47], but these modes are not used for the detection of pneumonia. The decision of combining the two models is taken experimentally, after conducting the experiments using each model individually; it was found that the combined model has achieved better results that the results obtained when applying each model individually. The other goal of this study is to show the importance of supporting the numerical results to evaluate the performance of the model with what is called the performance curve, which shows how the model learns during the training and validation processes.

The rest of this paper is organized as follows: In Section 2, we present the most prominent previous studies, specifically those that focused on deep-learning models and supported their results with a performance curve. Section 3 presents the proposed model in detail. Section 4 presents and discusses the experimental results. Finally, conclusions drawn from this study and directions for future work are presented in Section 5.

2. RELATED WORKS

Most of the recent studies, specifically those that were used to build systems for automatic detection of pneumonia, focused on the use of deep-learning models in general and CNN models in particular, as they have a high ability to extract the smallest details from images. We have highlighted in this section the most related studies that used deep-learning algorithms on the same database proposed in [17].

Most of the previous approaches used pre-trained models on large datasets. In [18], three types of pre-trained CNN models are used in the early feature-extraction stage; these include AlexNet, VGG-16 and VGG-19. Five algorithms are used, while in the classification stage, including KNN, decision tree (DT), linear discriminant analysis (LDA), linear regression (LR) and SVM are used. The best accuracy they reached was about 99%, but the performance curve showed a higher fluctuation in the validation phase than in the training phase.

In [19], a DenseNet-121 pre-trained model is used as a feature extractor with a deep neural network as a classifier. The best accuracy reached is 98% after conducting three experiments on the model, while the shape of the performance curve was unstable in the first and third experiments and reached a more

stable shape in the second experiment.

Another study based on pre-trained models is proposed by Chouhan et al. [1] using five pre-trained CNN models: ALexNet, DenseNet121, InceptionV3, ResNet18 and Google Net. The proposed model achieved an accuracy of 96.39% and slightly squiggly performance curves for all models.

In [20], a pre-trained model (Xception) is used in the feature-extraction step and an under-sampling technique is used to re-balance the dataset. The proposed model reached a value of 96% for AUC measure, a recall of 99%, a precision of 84%, an F1-score of 0.91 and a validation loss of 0.04. The performance curves started to fluctuate until the 20^{th} epoch, which is where the stability started perfectly.

In [21], five pre-trained CNN models: Xception, Inception-v3, VGG-16, ResNet50 and DenseNet201, are suggested to extract features from chest X-rays (CXR), then extracted features by every model are combined using Dempster–Shafer theory. The performance curve measure is not used for the evaluation of the model; only accuracy, precision and F1-score metrics are used. The model obtained an accuracy of 97%, a recall of 98% and an AUC of 99%.

In [22], Islam et al. proposed a model which consisted of two pre-trained CNN models: SqueezeNet and Inception-V3, to extract attributes from CXR images. SVM, KNN and Artificial Neural Network (ANN) are used for classification. The best reported results were obtained when using the ANN, which gave an accuracy and a sensitivity of 98% as well as a specificity and a precision of 99%.

Ayan and Ünver [23] compared the performance of the two pre-trained models Vgg16 and Xception. The results showed that the Vgg16 model outperformed the Xception model with an accuracy of about 87%, while the Xception model achieved an accuracy of 82%. For the performance curves, turbulence and fluctuations are evident in the performance curve with respect to the validation stage of the VGG16 model, while the performance curve seemed less turbulent with respect to the Xception model with a difference in the results of the two curves in the training and the validation phases.

A pre-trained ResNet-50 model is employed by Hussain et al. [24] for classifying injured and non-injured CXRs; data augmentation is also used to increase the number of images in the dataset. The model obtained an accuracy of about 90%, while the performance curve is very turbulent and has little stability in the validation phase.

Singh et al. [25] built a modified model of ResNet, called Quaternion CNN. The proposed model differs from the original ResNet model in which every convolutional layer is replaced by a Quaternion convolutional layer; also, every Relu layer is replaced by an exponential linear unit. The proposed model reached an accuracy of 93%, a loss value of 0.25% and an F1-score of 94%. The shape of the performance curve was turbulent until the fortieth step and then seemed perfect; also, the loss scale was stable in shape and the difference between the training curve and the validation curve was increasing over time.

Rači et al. [26] applied the traditional CNN model layer by layer where the accuracy of that model reached 90%. The performance curve for the validation stage was highly fluctuating and more pronounced in the loss scale than in the performance curve for the accuracy measure, which, although fluctuated, was more fluid and the difference between the two curves in the training and validation stages is less. In [3], CNN and ResNet models are combined and used for the detection of pneumonia. Also, Jain et al. [27] used six CNN models and four pre-trained models: VGG16, VGG19, Inception V3 and ResNet50, for pneumonia detection. The models obtained good results, while the performance curve for all models was unstable with a clear difference between the performances of the training and validation curves.

Mabrouk et al. [45] proposed the use of three CNN pre-trained models including DenseNet169, MobileNetV2 and Vision Transformer for the detection of pneumonia using ImageNet chest X-rays database. Experimental results showed that the proposed approach obtained good results compared to other previous approaches, with an accuracy of 93.91% and an F1-score of 93.88%.

Rahman et al. [46] used four CNN pre-trained models including AlexNet, ResNet18, DenseNet201 and SqueezeNet for the detection of pneumonia tested on a dataset consisting of 5247 chest X-ray images. The authors reported promising classification results compared to other approaches.

In [48], different transfer learning techniques are used to detect pneumonia from chest X-rays. Experiments showed that promising results are obtained by the proposed approach.

In [49], a deep-learning (MobileNet) model is proposed for the detection of pneumonia using chest X-ray images. The dataset proposed by Kermany [17] is used for training and testing the model. The model achieved an accuracy of 97.34% for training, an accuracy of 87.5% for validation and an accuracy of 94.23% for testing.

In [50], A CNN model using a dropout in the conv layer is proposed for the detection of pneumonia from chest X-rays. Kermany dataset [17] is used for training and testing the proposed approach. Achieved results were promising compared to recent approaches (accuracy = 97.2%, recall = 97.3%, precision = 97.4% and AUC = 0:982). Table 1 presents a summary of the key papers reviewed in this study using the same dataset [17].

Table 1. Summary of the key papers.

Ref.	Approach	Limitations	Best Results
[1]	CNN model with pre-trained models	Slightly zigzag curves in the performance curve.	Accuracy: 96.39%, Recall: 99%
[3]	CNN with lightened image on increased contrast with ResNet	A low value is a measure of accuracy with no consideration of the performance curve.	Accuracy: 78.73%
[19]	Lightweight Deep ANN	Unstable performance curve.	Accuracy: 98%
[20]	Xception pre-trained model	Different results for different performance measures.	AUC: 0.96%, Recall: 0.99, Precision: 0.84%, F1-score: 91%
[23]	Vgg16 and Xception models	Oscillating curves of both models. Difference between the performance curve for the training and validation phases of the Xception model.	Accuracy: 87%
[26]	CNN	The performance curve was too squiggly in the validation phase.	Accuracy: 90%
[27]	CNN, VGG16, VGG19, ResNet50 and InceptionV3.	The performance curve is unstable for all models.	Accuracy: 92% and 88%,

3. MATERIALS AND METHODS

3.1 Materials

The dataset proposed by Kermany et al. [17] is used to evaluate the proposed approach. The dataset consists of 5856 CXR images of pneumonia of two classes: infected images and not infected images. This dataset is for children within ages between 1 and 5 years and classified by specialists in the "Women and Children's Medical Center" at Guangzhou [42]. The dataset is divided into 89% for training, 10% for testing and 1% for validation. This distribution is presented in Table 2.

Table 2. The original distribution of the dataset.

	Normal	Pneumonia	Total
Training	1341	3875	5216
Testing	234	390	624
Validation	8	8	16
Total	1583	4273	5856

It can be seen from Table 2 that the percentage of pneumonia images reached about 73%, while the percentage of normal images was about 27% for the whole dataset. The percentage of pneumonia images in the data test reached about 63% and the percentage of normal images is about 37%.

The second experiment is conducted after redistributing the dataset into 75% for training, 22% for validation and 3% for validation, as presented in Table 3.

Table 3. The redistribution of the dataset.

	Pneumonia	**Normal**	**Total**
Training	3207 (73%)	1185 (27%)	4392 (75%)
Testing	811 (63%)	477 (37%)	1288 (22%)
Validation	88 (50%)	88 (50%)	176 (3%)
Total	4229 (72%)	1627 (28%)	5856 (100%)

The third experiment is conducted after applying data augmentation to the original dataset with 75% for training, 22% for testing and 3% for validation, as presented in Table 4. The distribution here was an equal distribution for each class in the training, testing and validation, since we are trying to test the effect of this distribution of the classes on the performance of the model. The dataset size after augmentation reached 29,890 images, divided as 22,417 for training, 6575 for testing and 898 for validation (Table 4).

Table 4. The distribution of the dataset after augmentation.

	Pneumonia	**Normal**	**Total**
Training	11,432	10,985 (49%)	22,417 (75%)
Testing	3,419 (52%)	3,156 (48%)	6,575 (22%)
Validation	449 (50%)	449 (50%)	898 (3%)
Total	4229 (72%)	1627 (28%)	29,890 (100%)

3.2 Parameter Settings

Parameter setting is one of the most sensitive steps and affects the results of the model; it is used to reduce network error. The values of these parameters are chosen experimentally for all experiments, as presented in Table 5. Parameters' values are also related to the type of optimizer used; for instance, a learning rate of 0.0001 is used for the Adam optimizer for updating the network weights. 30 epochs are used, which indicates the number of times the training dataset is divided according to the patch size across the network and accordingly, the parameters in the network are updated. Default steps per epoch that are equal to several samples in the training dataset are divided by the batch size and default steps between validation epochs that are equal to several samples in testing dataset are divided by the batch size for fitting the model with data step. L2 regularization is used with the SGD optimizer to overcome the overfitting problem, where the learning rate is changed to 0.00001 and the number of epochs is also changed to 40; this value represents the number of forward and backward stages to be used in the training and verification stages. Dropout regularization is used in the resampling stage instead of L2 regularization.

Table 5. Training hyper-parameters.

Parameter	**Value**
Batch size	24
Optimizer	Adam, SGD
Learning rate	0.0001(Adam) and 0.00001(SGD)
Number of	30,40

Steps per epoch	Number of training samples / batch size
Validation steps	Number of testing samples / batch size
Regularization	L2, Dropout

3.3 Evaluation Metrics

The performance of the proposed model is evaluated using five metrics: accuracy, precision, recall and AUC and it is the rate between the TP and FP. In general, it is used in binary classification models and it is used to evaluate models that take into account the same percentage of importance for the classification of both types of samples.

Binary cross entropy is also used as a loss function to measure the difference between the estimated probability and the actual probability [43]. These metrics are formulated using Formulae 1 - 4.

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN} \tag{1}$$

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

$$AUC = \int_0^1 \frac{TP}{P}\, d\frac{FP}{N} = \frac{1}{P\,N} \int_0^N TP\, d\, FP \tag{4}$$

where, TP, FN, FP and TN refer to True Positive, False Negative, False Positive and True Negative, respectively. TP represents the number of correct classifications of positive observations of phenomena-infected images; FN represents the number of incorrect classifications of positive observations (normal images) that are incorrectly labeled as phenomena-infected images. FP represents the number of incorrect classifications of negative observations and TN represents the number of correct classifications of negative observations.

3.4 Methods

The overall architecture of the proposed approach is presented in Figure 2. It comprises three phases: pre-processing, data augmentation and normalization, as well as feature extraction and classification.

Since the dataset used in this study is unbalanced, two different techniques are used to tackle this problem: the class-weight and the resampling techniques. In the first technique, the weights of the different classes in the dataset are rebalanced so that the model is prevented from bias towards the most present class. This technique uses a kind of hyper-parameters added during the fitting or training processes; i.e., it is an amendment to the existence and weights of the classes so that the presence of the two classes is balanced. Two experiments are conducted; in the first experiment, the same weight is given for both classes, while in the second experiment, higher weight is given to the class with the least presence (the normal class). For the second technique (resampling), the number of images is

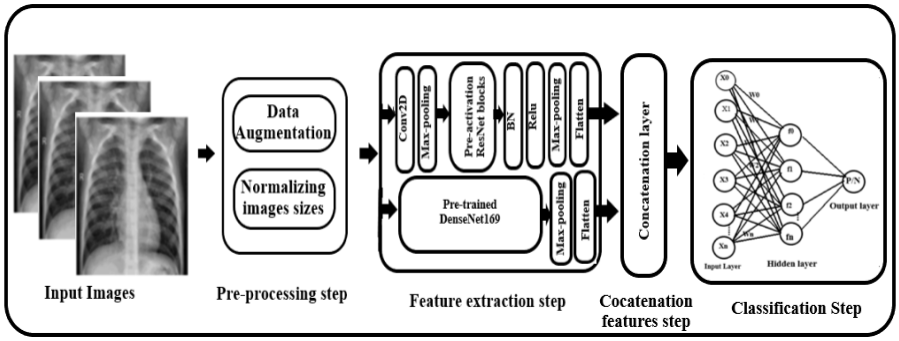


Figure 2. The proposed approach.

increased for the least frequent category based on the data augmentation method, so that the number of images for both categories is balanced.

3.4.1 Pre-processing

Data augmentation is used to increase the volume of the dataset, since deep-learning models need large data to give a high performance. It is also used to address the problem of overfitting, i.e., the model achieves good results on the training data, while it doesn't perform well on new unseen data [28]. Eight augmentation methods are applied online in this study as presented in Table 6. All these operations are performed randomly to produce new images, where more than one operation was applied [29]. Image sizes for the dataset are normalized to 224x224 to be suitable for the DenseNet169 pre-trained model.

3.4.2 Feature Extraction

Two CNN models are used in this study, including pre-activation ResNet and DenseNet169. The pre-activation ResNet model is built from scratch; i.e., defined layer by layer and the DenseNet169 model is applied using the weights used during the training phase on the ImageNet dataset.

Pre-activation ResNet Model

Pre-activation ResNet is a special version of ResNet proposed by He et al. [30]. The ResNet model has two different architectures based on the flow of data. The first architecture follows a main path, while the second architecture takes a shortcut path; i.e., data can travel between layers and bypass the others depending on the type of main path [31].

Table 6. The proposed data augmentation methods.

Augmentation process	**Description**
Rescale (1/255)	Transforming the image pixel values (0 – 255) to values between 0 and 1.
Rotation = 40	Rotating images between 1° and 40° degrees.
Width-shift = 0.2	Shifting images horizontally with a percent of 0.2.
Height-shift = 0.2	Shifting images vertically with a percent of 0.2.
Zoom-range = 0.2	Scaling and inverse scaling of images.
Horizontal flip = True	Flipping images horizontally.
Fill-mode = 'nearest'	Filling the nearest of the body shape with points (the border of the image).
Shear-range = 0.2	Shearing of images.

The first architecture consists of an arrangement of layers: a convolutional layer, a batch normalization layer and a rectified linear unit (Relu) layer or a bottleneck. The second architecture is based on the same principle with a different arrangement of the layers, so that their arrangement in the main path is as follows: BN layer, a Relu layer and a convolutional layer (conv layer). Figure 3 shows the two different architectures of the ResNet. In this study, we used a pre-activation ResNet architecture with 9 pre-activation residual blocks.

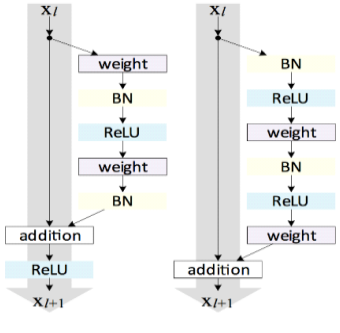


Figure 3. ResNet *vs.* pre-activation ResNet [30].

Equation 5 represents the block of the pre-activation ResNet, where the output layer in the pre-activation ResNet is $Xl + 1$ and the shortcut path Xl is the input features. F is the path,°F is the activation function and Wl is the weight.

$$\mathrm{X}l + 1 = \mathrm{X}l + \mathrm{F}(°\mathrm{F}(\mathrm{X}l), \mathrm{W}l) \quad (5)$$

BN layer is based on the principle of estimating the means and standard deviations of the inputs of each layer for each small batch or subset of the training set. Bjorck et al. [32] defined BN as "a normalization method for layer activation in the hidden layers of deep-learning models". BN layer is used to increase the training speed and to improve the accuracy in deep-learning models [33].

Relu is the most widely used type of activation function in deep-learning models. It defines the output of the hidden layer inputs and works by converting all numbers less than zero into zero while keeping the other numbers. Equation 6 represents how the Relu works by converting each number less than zero into be zero and keeping the other numbers at their values.

$$f(x) = max(0, x) \quad (6)$$

Conv layer is the third layer on the pre-activation ResNet block. It works by moving the filter on the images to extract the features. This process is done by multiplying the value of the image's pixel array with the filter number, which produces what is called a feature map [34]. In practice, that sequence is defined iteratively again in the main path as we define one conv layer in the shortcut path. As for the layers before those blocks, the conv layer is defined, followed by the max-pooling layer to reduce the size of feature maps. After defining the blocks, we must define (BN => Relu) [35]. The next layer is the pooling and flatten layer; its function is to reshape the feature maps from a two-dimensional array to a one-dimensional array to be fed into the ANN classifier after merging it with other features from the DenseNet169model. Figure 4 shows the main layers of the proposed pre-activation ResNet model.

Pre-trained DenseNet169 Model

DenseNet169 is a type of densely connected convolutional layer with 169 layers. It is proposed by Huang et al. [36] and considered as one of the most popular CNN models used by researchers for classification and segmentation tasks. DenseNet169 is trained on many large datasets, but the volume of these datasets is less than those used by some researchers like Kundu et al. [37] and El Asnaoui et al. [38]. In this study, we used the DenseNet169 model with the ImageNet datasets for feature extraction and ANN was used for classification. Figure 5 shows a DenseNet model with 5 layers. The pre-trained DenseNet169 model is applied after resizing the image size into 224x224. The features extracted from the two models are combined using the concatenation layer [39].

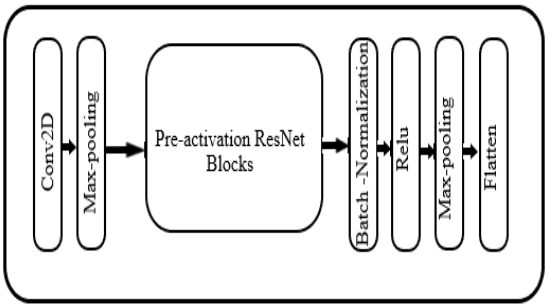


Figure 4. The proposed pre-activation ResNet.

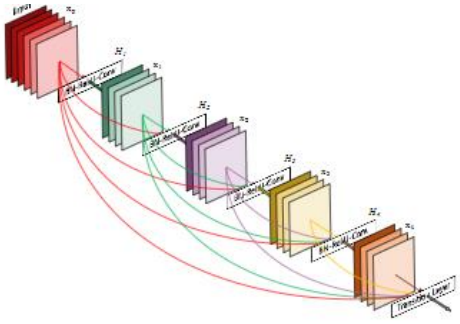


Figure 5. DenseNet with 5-layer connection [35].

3.4.3 Classification

The network consists of one hidden layer, 450 neurons, a Relu activation function and one output layer with one neuron. The number of layers and the number of neurons are determined experimentally; i.e., one hidden layer and 450 neurons gave the best results. ANN is used for classification and the sigmoid activation function is also used in this study, since it is commonly used for similar tasks [40]. Feature extraction is performed using two models based on a certain number of samples or specific batch.

4. EXPERIMENTAL RESULTS

In this section, we present the results and performance curves that express how the model performed during the training and validation phases. In the three main stages: original distribution stage, class weight stage and re-sampling stage, the results of each stage are divided into sub-stages based on the type of optimizer used and other additional criteria. The metrics used for evaluation are: precision, recall, accuracy, loss and AUC. The model's performance in the training phase is represented in blue for all curves and the orange curve expresses the model's performance during the validation phase.

A different number of layers is used for the two models in the feature-extraction stage depending on the used optimizer. For instance, 85 layers are used for pre-activation ResNet and DenseNet169 used with the Adam optimizer, while 37 layers are used with the SGD optimizer. For the classification step, an ANN is used with one hidden layer containing 450 neurons as well as a Relu activation function and one output layer with one neuron and a sigmoid activation function. Since we applied the pre-activation ResNet model from scratch, we tried to apply it with different numbers of layers. After a number of attempts, we found that the proposed model obtained best results when the number of layers is 85 for the Adam optimizer (including feature-extraction layers and two classification layers) and 37 layers for the SGD optimizer.

These classes are designed for each of the three main stages, dealing with the original data, class weight and re-sampling stages. Numbers of features extracted using the pre-activation ResNet and DenseNet169 models are 200704 and 14976, respectively. The input size of the classifier for all experiments is 224x224x3.

4.1 Results Using the Original Dataset

The performances of the proposed models are represented using performance curves which express the learning behaviour of the models during training and validation stages. The curve consists of two levels: the x-axis expresses the number of epochs or steps that the model walks back and forth during training and validation. The y-axis expresses the number of metrics used in the evaluation. Each Cartesian level contains two curves. The first is the training curve (blue curve), which expresses the performance of the model when trained on the training data. The second is the validation curve (orange curve), which expresses the performance of the model in the validation stage.

4.1.1 Results Using Adam Optimizer

Figure 6 shows how the model behaviour during the training and the validation phases when using the Adam optimizer. It can be seen from the Figure that the model was stable during the training stage, while it was unstable during the validation stage. We found that the dropout has increased the loss, which is worthy to be investigated more in future research. It can be seen also from Figure 6 how the model was learned throughout the training and validation periods. Looking at the movement of the training and validation curves, it can be noticed that the movement of the training curve (blue-color curve) is stable throughout the period, while the movement of the validation curve for all the metrics was very mobile and completely unstable, ending with very poor results at the last epoch.

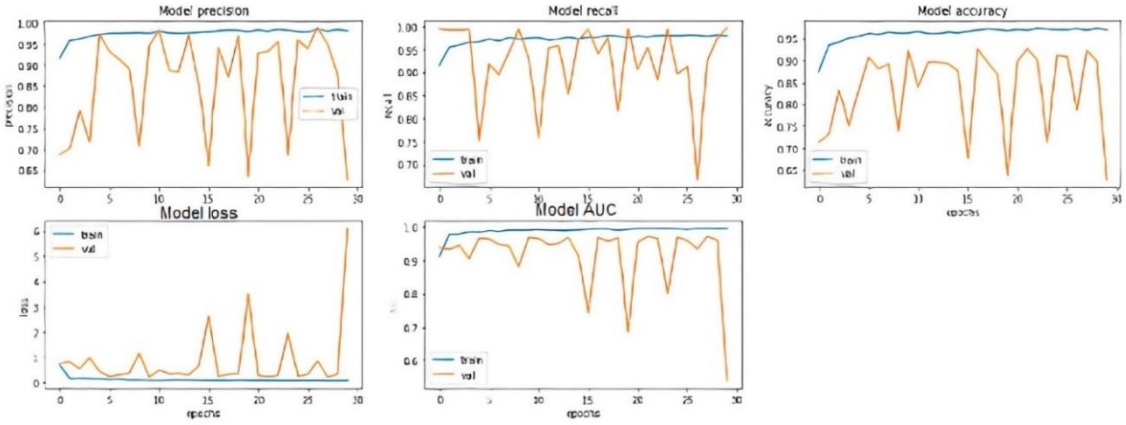


Figure 6. Results using the original dataset and the Adam optimizer.

4.1.2 Results Using SGD Optimizer

Figure 7 shows the model behaviour during the training and the validation phases when using the SGD optimizer. It can be seen from this Figure that the curves began to stabilize clearly. We also found that there is a difference between the performance of the training curve and that of the validation curve. It can be also noticed that the curves started to stabilize a little compared to the previous stage after reducing the number of layers and reducing the learning rate and the length value of l2. Also, there is a difference between the performance of the training curve and that of the validation curve.

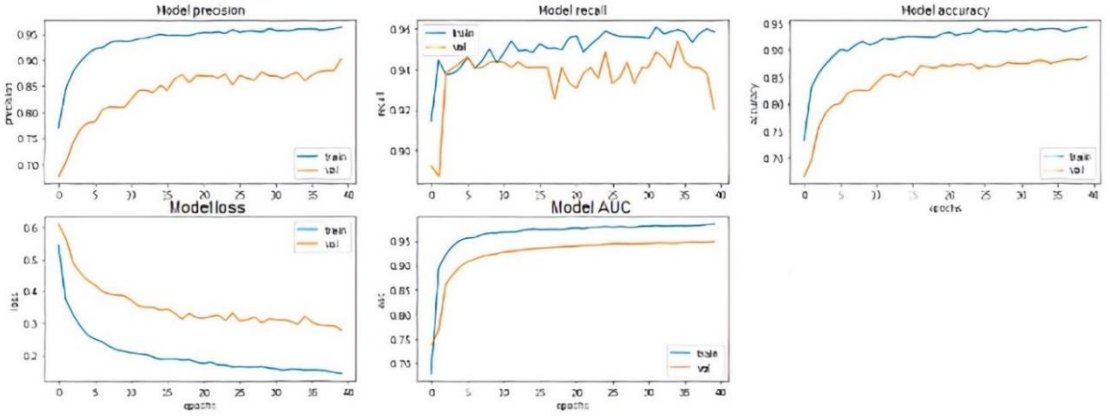


Figure 7. Results using the original dataset and the SGD optimizer.

4.2 Results Using Class Weighted Dataset

4.2.1 Results Using Adam Optimizer

Figure 8 shows the results of the proposed model when using the Adam optimizer with a class weight dataset. The class weighting technique is used to tackle the problem of imbalanced data and to improve the loss values from the five measures. The model's performance curves in the validation stage express the inability of the model to generalize all the validation data, which is shown by the highly volatile curve in the validation stage (orange color).

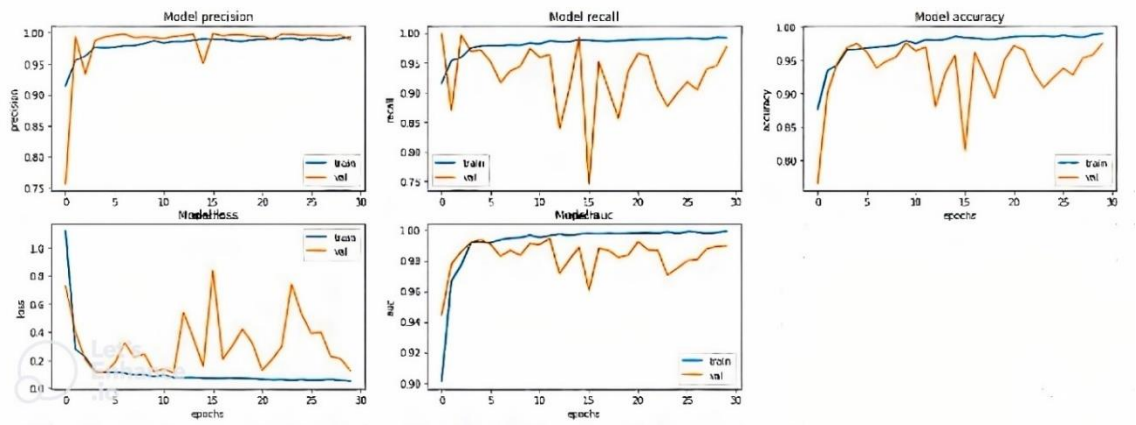


Figure 8. Results using class weight and the Adam optimizer.

After an analysis of the results for this stage (Figure 8), we noticed that the precision started with slightly high values at the beginning of the training with a value of 93% and began to increase with the passage of learning time, where the values were stabilized at epoch 10 until the end of the specified period at the 30th epoch, to stop at a value of 99%. For the validation curve, the precision started with low values at the beginning of the retraining, but it quickly adjusted the values to reach the highest level at epoch 2, then it suddenly decreased to reach a value of 94% at the third epoch. Then, it returned to high and stable values from epoch 5 to epoch 14 and suddenly decreased by one epoch and returned to rising and stabilized again until the end of the period, reaching a value of 98% at the 30th epoch.

The recall values were stable from the beginning of the fifth epoch until the end of the period to stop at a value of 99%. For the validation curve, it started with very high values, but it was fluctuating throughout the training period and reached its lowest level in the middle of the period to go up, but with high fluctuations. It reached a high value of 99% at the end of the period.

The accuracy values started with the lowest value of 88% at the first epoch, but it continued to improve and began to stabilize at epoch 15 until the end of the period to stop at a value of 99%. For the validation curve, it was very low at 75%, but it quickly rose at the third epoch; however, its behaviour was very volatile throughout the period, but it stopped at the end of the period at a good value of 97%.

For the loss metric, it is concluded that the model during training continued to decline from the beginning of the third epoch with great stability in the curve and reached a value of 0.002 at the end. While the behaviour of the model during the validation period was fluctuating from the beginning, but it stopped at a small error rate of 0.01 at the last epoch.

Lastly, the AUC values in the training phase started to stabilize early at epoch 3 until the end of the period to stop at a value of 99%, while the performance of the model in the validation stage was fluctuating throughout the period to stop at a value of 97%.

4.2.2 Results Using SGD Optimizer

Several attempts and different values have been tried for the weights of the classes at this stage. In the first attempt, we used equal class weight with the SGD optimizer with less number of pre-activation blocks (three blocks) to minimize the complexity of the model. In the second attempt, we have given a higher weight for class 0, which had a lower distribution between the two classes; the original distribution for class 0 is multiplied by 1.5, while the original weight for class 1 is kept unchanged.

- ***Results Using Equal Class Weights***

Figure 9 shows the results of the proposed model when using the SGD optimizer with equal class weights.

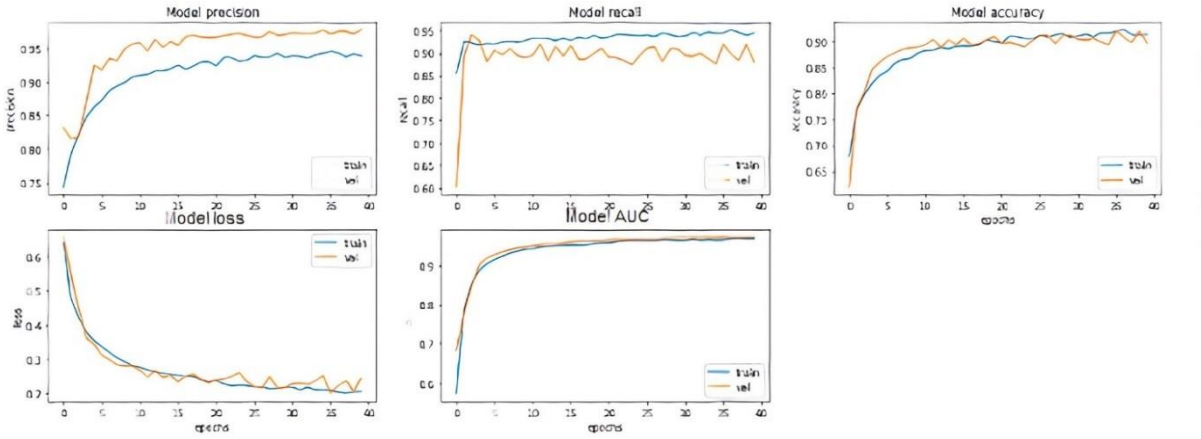


Figure 9. Results using equal class weights and the SGD optimizer.

After a careful analysis of the results in Figure 9, we notice that the precision started with slightly low values from the beginning of the training, but it quickly began to gradually rise from epoch 5 to epoch 40 (end of training). The performance curve was almost stable with very few bends reaching a precision value of about 95%. The model's behavior during the validation period was almost stable with very few zigzags that continued until the end of the period to reach a value of 97%.

The recall values during the training stage were stable from the beginning of the third epoch until the end of the training, stopping at a value of 93%, while during the validation stage, the recall started with low values, but it fluctuated slightly throughout the period reaching an average value of 88%.

The accuracy during the training phase started with a low value of 68% at the first epoch, but it continued to rise and began to stabilize at epoch 10 until the end of the training to stop at a value of

92%, while in the validation phase, its behavior was slightly volatile throughout the period and stopped at the end of the period at a value of 89%.

It is also noticed that the behaviour of the model during the training phase continued to decline from the beginning of epoch 3 reaching a loss value of about 0.3. It continued to decline until the end of the period to reach a loss value of about 0.19. The values for the AUC measure during the training and validation phases started stable from the beginning to the end of the period, stopping at a value of about 97%.

- ***Results Using Different Class Weights***

Metrics' values and numbers improved when changing the weight of the least available samples in the data, which is the normal class, by 1.5 compared to 1.0 for the highest available samples. Although the curves are less stable than in the previous stage, as shown in Figure 10, there were slightly better results as numbers, where the precision ratio reached 98% and the stability of performance relative to the AUC metric remained at 97% and the recall metric remained at 88%, while the accuracy metric improved by 0.01. Its value becomes 90% and the loss ratio becomes 0.23.

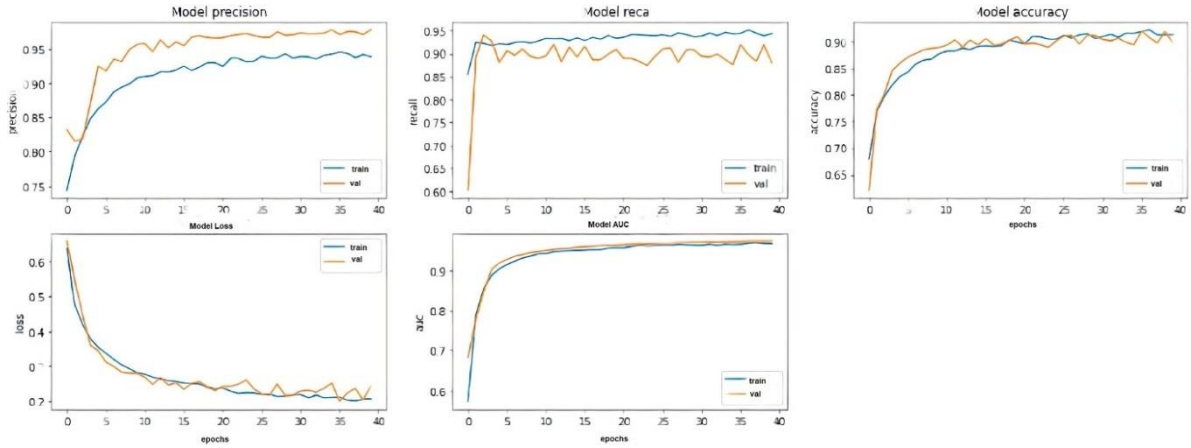


Figure 10. Results using different class weights and the SGD optimizer.

4.3 Results Using Re-sampled Dataset

In this stage, we added a dropout layer with values of 0.8 and 0.9. This layer is used to decrease the complexity of the nodes in the ANN in a dense layer by randomly picking nodes to make them idle or disabled for a certain time; this step is also used to reduce the effect of the overfitting problem. Table 7 shows the hyper-parameters used in this stage for both Adam and SGD optimizers.

Table 7. Training hyper-parameters in the re-sampling stage.

Parameter	Value
Batch size	24
Learning rate	Adam: 0.0001; SGD: 0.00001
Number of epochs	30, 40
Steps per epoch	No. of training samples: batch-size
Validation steps	No. of testing samples: batch-size
Regularization method	L2, dropout

4.3.1 Results Using Adam Optimizer

Figure 11 shows the performance curves of the model using the re-sampling method. The model was good in the training stage, but its performance was not stable in the validation stage, with a very high loss value, so we applied another optimizer with a less-complexity model in the next step.

4.3.2 Results Using SGD Optimizer

Figure 12 shows the results for the different metrics in the re-sampling stage using the SGD optimizer.

In this stage, we noticed that the resulting curves became smoother than in the re-sampling stage with Adam optimizer, but we need to increase the dropout ratio to minimize the difference between the training and the testing results, so we applied a model with just 3 blocks for the pre-activation ResNet model.

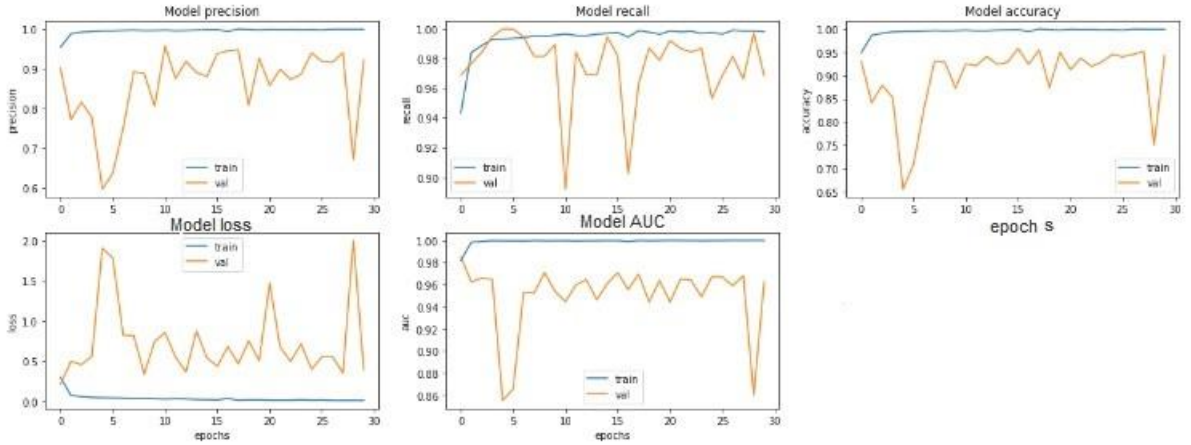


Figure 11. Results using the re-sampled dataset and the Adam optimizer.

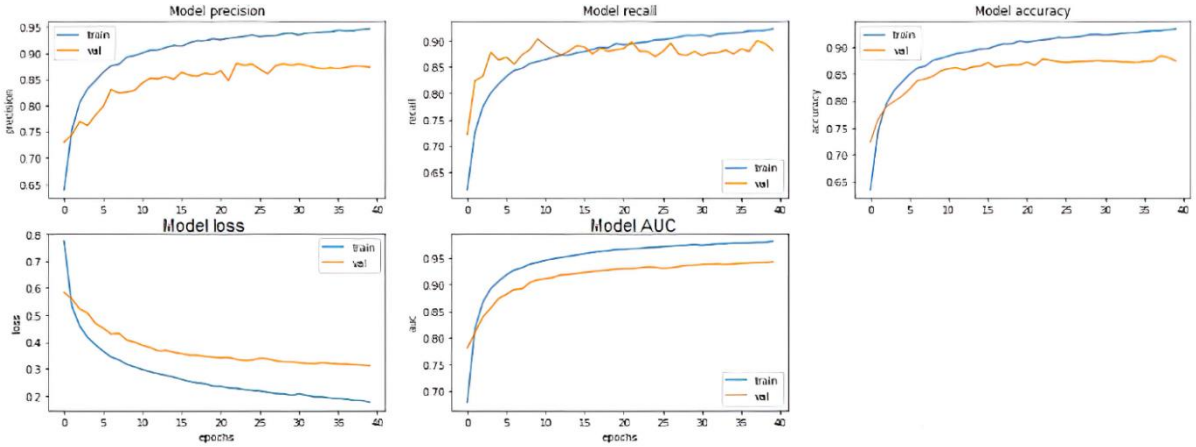


Figure 12. Results using the re-sampled dataset and the SGD optimizer.

The training stage started with a small precision value at the beginning of the training and continued to improve and rise with a steady movement until it reached a precision value of about 97%, while in the validation stage, it started with a good start and continued to improve with a slight rise. The curve formed a little zigzag and the training stopped at epoch 40 by a value of 88%.

The recall values at the training stage started with a value of about 63% and continued to rise gradually with stable movement until it reached a value of 93%, while at the validation stage, the recall started at a value of about 82%, while the shape of the curve was fluctuating by a small percentage and the recall ended at a value of 89%. The movement of the training curve was increasing as the number of epochs increases with a smooth curve stopping at an accuracy value of about 95% and the validation curve started to stabilize at approximately epoch 10 until it reached epoch 40 with an accuracy of about 88%.

The loss values at both the training and the validation stages were stable reaching a value of about 0.11at the training stage and a value of about 0.27 at the validation stage. The movement of the two curves for the AUC metric was stable and gradually increasing, with the training curve stopping at a value of about 99% and the validation curve stopping at a value of about 95%.

4.4 Results' Comparison for the Three Stages

The experiments in this study are divided into three main stages. The first stage is to train the model on the data with its original distribution and the remaining two stages deal with the weighting stage

and the resampling stage The goal in this study is to reach a good performance numerically as well as behaviourally through its behaviour in the performance curve, so that we reach a stable model as much as possible. The results are presented through the performance curves in the previous paragraph in detail and Table 8 summarizes the results for each experiment.

The model obtained a bad and unstable performance when trained on the data with its original distribution. Referring to Table 8, it shows bad numerical results, so there was a need to increase the percentage of data in the performance validation stage and using techniques to deal with the irregular distribution in performance. The goal of this study is to make a trade-off and to combine the stability of the performance curve and the numerical results to choose the best results.

Table 8. Results' comparison for the three stages.

Stage	Results	Precision	Recall	Accuracy	AUC	Loss
Original distribution stage (Adam optimizer)	Training	0.74	0.99	0.74	0.53	4.42
	Validation	0.62	0.99	0.62	0.54	6.10
Original distribution stage (SGD optimizer)	Training	0.96	0.95	0.94	0.98	0.14
	Validation	0.90	0.92	0.88	0.94	0.27
Class weight stage (Adam optimizer)	Training	0.99	0.99	0.99	0.99	0.002
	Validation	0.97	0.99	0.97	0.97	0.01
Equal class weight stage (SGD optimizer)	Training	0.95	0.93	0.92	0.97	0.19
	Validation	0.97	0.88	0.89	0.97	0.24
Different class weight stage (SGD optimizer)	Training	0.96	0.93	0.92	0.97	0.18
	Validation	0.98	0.88	0.90	0.97	0.23
Re-sampling stage (SGD optimizer and dropout of 0.8)	Training	0.974	0.939	0.957	0.992	0.115
	Validation	0.880	0.890	0.883	0.953	0.279
Re-sampling stage (SGD optimizer and dropout of 0.9)	Training	0.965	0.927	0.946	0.987	0.142
	Validation	0.876	0.907	0.888	0.948	0.292

Despite the high results of the class weight stage experiment with Adam optimizer, its performance curves were zigzag, which means that the model has no guaranteed performance. Therefore, the results of the different class weight experiment are nominated with SGD optimizer to combine good performance and acceptable shape of performance curves.

4.5 Comparison with Other Approaches

The results of the proposed approach are compared with the results of nine previous approaches which used the original distribution of the dataset proposed in [17]. This comparison is presented in Table 9. The proposed approach achieved promising results compared to other approaches, especially for the precision metric.

Table 9. Results' comparison with others approaches.

Ref.	**Algorithm**	**Accuracy**	**Recall**	**Precision**	**AUC**	**Loss**
[3]	CNN	78	-	-	-	-
[15]	ResNet-34	92	99	90	-	1.6
[27]	VGG16	87	96	-	-	0.3
	VGG19	88	95	-	-	0.3
	Inception V3	70	84	-	-	0.9
	ResNet50	77	97	-	-	0.6

[29]	ResNet152V2	99	99	99	99	-
	MobileNetV2	96	99	95	97	-
	CNN	92	92	95	96	-
	LSTM	91	92	93	95	-
[20]	Xception	-	99	84	96	0.04
[24]	Pre-trained ResNet50	90	93	93	89	0.03
[23]	VGG16	87	-	-	-	0.3
	Xception	82	-	-	-	0.45
[2]	CNN model	84	-	-	-	0.8
[37]	Xception	83	-	95	-	0.6
	DenseNet201	93	-	99	-	1.9
	MobileNet_v2	96	-	98	-	0.24
	VGG19	85	-	80	-	1.3
	CNN	84	-	94	-	0.4
	VGG16	86	-	87	-	1
	Inception_v3	94	-	93	-	1.76
	ResNet 50	96	-	98	-	1.5
	Inception_ResNet_V2	96	-	98	-	1.1
Proposed model	Pre-activation ResNet with DenseNet169	90	88	98	97	0.23

Comparing the results in terms of accuracy shows that better results are achieved compared to many previous approaches, including ResNet-34 [15], MobileNetV2, CNN and LSTM [29], the pre-trained Xception model [20], ResNet-50 [24], Xception, VGG19 and CNN [37], while the difference was clear between our results and those of other models in measuring AUC; other models include MobileNetV2, CNN, LSTM [29], the pre-trained Xception model [20], ResNet-50 [24]. The last column shows the difference between our results in the loss metric compared to ResNet-34 [15], [27] and its four algorithms [23], [37].

4.6 Results Discussion

The proposed approach was able to reach a performance that combines good results numerically and behaviourally acceptable in the performance curve. In addition, we overcame many challenges that we encountered during this study, like the small size of the test data, which appeared for the first time in the performance curve. To overcome this challenge, we repartitioned the training and validation data from scratch.

Then, we used different techniques to deal with the imbalance of data distribution; namely, the categorical weight and re-sampling techniques with the increase of data. Another challenge was the problem of overfitting, which we addressed by reducing the number of ResNet pre-activation layers and using an organization type suitable for the two techniques implemented in the previous challenge. L2 is used with the class weighing technique and the dropout is best suited with the re-sampling technique. In addition, the optimizer was changed from Adam to SGD, where the use of the SGD optimizer had a positive effect in terms of model performance stability compared to the results obtained when applying the Adam optimizer. The explanation for this phenomenon is that the SGD optimizer splits the batch number of features equal to the number of those features, making the performance movement more stable, specifically in the loss scale and thus allowing for the possibility of updating the weights for each training sample. The second conclusion is that the method for dealing with unbalanced data has a role in determining the type of regularization method that is the best, where the L2 method was the best suited one for the class weight technique, while the dropout method was the best suited for the re-sampling technique, which may be due to the volume of data used in each

stage. The results were very good on the precision, AUC and loss metrics. However, the results were modest in terms of performance on the measures of accuracy and recall. The reason according to our analysis was the inequality of weight for each class, so it may cause a conflict in the classification of the two classes. When comparing our results with those of other models, we got good results on most metrics, except for recall. Finally, we can also say that the proposed approach was able to outperform many models used in previous studies in terms of reaching stable performance and among these studies that reached fluctuating performance regardless of the results as values are [22]-[24], [27] and [44]. In addition, the results have a small loss rate and the difference was obvious compared to the two studies [26] and [2], which had an error rate of more than 1.00.

Despite that the proposed approach gave promising results compared with the state-of-the-art approaches, it has some limitations, such as:

- The used database consists of X-ray images of children between 1 and 5 years old, which may make the approach not comprehensive for other age groups.
- The process of parameter setting for the pre-activation ResNet model is performed experimentally to choose the most appropriate parameters.

5. CONCLUSIONS

AI applications have proven their efficiency in the early diagnosis of many diseases, which contributed to saving the lives of many patients. In this study, we proposed an approach by merging different deep-learning models to extract features from CXR images for the automatic detection of pneumonia disease. Two models are proposed: pre-activation ResNet and pre-trained DenseNet169. Most of the previous researchers focused on highlighting numerical results without considering the estimate of performance in the performance curve in the training and validation stages, while our focus in this research was more on combining the stability of the performance curve in addition to the numerical values of the performance measures. The proposed approach has obtained a well-stable curve compared with those obtained by many previous studies [19] and [22]. As numerical results, we achieved a precision of 98%, an AUC of 97% and a loss of 23%, while the performance was modest in terms of recall and accuracy. From the findings of the proposed approach, we conclude the following points:

- The nature and distribution of data have a significant impact on the performance of the model and each type of data has its own appropriate way to deal with.
- The type of optimizer used may increase the quality and stability of the model's performance or *vice versa*. It was evident in the results that the model became more stable when applying the SGD optimizer compared to its performance with the Adam optimizer.
- Reducing the number of layers or blocks in the pre-activation ResNet model contributes to stability and better results as well.
- The more stable the performance curve, the more assured its performance.
- Evaluating the performance of the model based on the final results of the performance measures as numbers alone is not sufficient to prove the quality and effectiveness of the model.
- The proposed approach was able to outperform those used in many previous studies, to which we dedicated a special section.
- Since the size of the proposed dataset was not sufficient to achieve satisfactory results, we had to redistribute it again. The distribution was also unbalanced and we handled it in two ways: category weighting and re-sampling as the data increased.
- The proposed approach was able to obtain a more stable performance curve than those obtained in many previous works, as detailed in the discussion section.

For future work, we suggest applying the proposed approach to data on COVID-19 pandemic and to integrate ResNet pre-activation with other CNN models.

REFERENCES

[1] V. Chouhan, S. K. Singh, A. Khamparia et al., "A Novel Transfer Learning Based Approach for Pneumonia Detection in chest X-ray Images," Applied Sciences, vol. 10, no. 2, p. 559, 2020.

[2] A. Sharma, M. Negi, A. Goyal, R. Jain and P. Nagrath, "Detection of Pneumonia Using ML & DL in Python. IOP Conference Series: Materials Science and Engineering," Proc. of the 1st Int. Conf. on Computational Research and Data Analytics (ICCRDA 2020), Rajpura, India, DOI: 10.1088/1757-899X/1022/1/012066, 2021.

[3] C. J. Saul, D. Y. Urey and C. D. Taktakoglu, "Early Diagnosis of Pneumonia with Deep Learning," ArXiv. /abs/1904.00937, DOI: 10.48550/arXiv.1904.00937, 2019.

[4] R. Sarkar, A. Hazra, K. Sadhu and P. Ghosh, "A Novel Method for Pneumonia Diagnosis from Chest X-Ray Images Using Deep Residual Learning with Separable Convolutional Networks," Proc. of the Computer Vision and Machine Intelligence in Medical Image Analysis Conf., Part of the Advances in Intelligent Systems and Computing Book Series, vol. 992, DOI: 10.1007/978-981-13-8798-2_1, 2020.

[5] A. Tilve, S. Nayak, S. Vernekar, D. Turi, P.R. Shetgaonkar and S. Aswale, "Pneumonia Detection Using Deep Learning Approaches," Proc. of the Int. Conf. on Emerging Trends in Information Technology and Engineering (ic-ETITE), pp. 1-8, DOI: 10.1109/ic-ETITE47903.2020.152, 2020.

[6] D. Varshni, K. Thakral, L. Agarwal, R. Nijhawan and A. Mittal, "Pneumonia Detection Using CNN Based Feature Extraction," Proc. of the IEEE Int. Conf. on Electrical, Computer and Communication Technologies (ICECCT), pp. 1-7, DOI: 10.1109/ICECCT.2019.8869364, 2019.

[7] A. M. Alqudah, S. Qazan and I. S. Masad, "Artificial Intelligence Framework for Efficient Detection and Classification of Pneumonia Using Chest Radiography Images," J. Medical and Biological Eng., vol. 41, pp. 599–609, DOI: 10.1007/s40846-021-00631-1, 2021.

[8] A. K. Jaiswal et al., "Identifying Pneumonia in Chest X-rays: A Deep Learning Approach," Measurement, vol. 145, pp. 511–518. DOI: 10.1016/j.measurement.2019.05.076, 2019.

[9] S. Kido, J. Ikezoe, H. Naito, S. Tamura and S. Machi, "Fractal Analysis of Interstitial Lung Abnormalities in Chest Radiography," Radiographics, vol. 15, no. 6, pp. 1457-1464, DOI: 10.1148/radiographics.15.6.8577968, 1995.

[10] T. Ishida, S. Katsuragawa, K. Ashizawa et al., "Application of Artificial Neural Networks for Quantitative Analysis of Image Data in Chest Radiographs for Detection of Interstitial Lung Disease," Journal of Digital Imaging, vol. 11, no. 4, pp. 182–192, DOI: 10.1007/BF03178081, 1998.

[11] T. Rafael et al., "Comparative Performance Analysis of Machine Learning Classifiers in Detection of Childhood Pneumonia Using Chest Radiographs," Procedia Computer Science, vol. 18, pp. 2579-2582, DOI: 10.1016/j.procs.2013.05.444, 2013.

[12] S. Reza, O. B. Amin and M. M. A. Hashem, "A Novel Feature Extraction and Selection Technique for Chest X-ray Image View Classification," Proc. of the 5th Int. Conf. on Advances in Electrical Engineering (ICAEE), pp. 189-194, DOI: 10.1109/ICAEE48663.2019.8975457, 2019.

[13] G. Verma and S. Prakash, "Pneumonia Classification using Deep Learning in Healthcare," Int. J. of Innovative Technology and Exploring Engineering (IJITEE), vol. 9, no. 4, pp. 1715–1723, 2020.

[14] N. Ansari, A. Faizabadi, S. Motakabber and M. Ibrahimy, "Effective Pneumonia Detection Using ResNet based Transfer Learning," Test Eng. and Management, vol. 82, pp. 15146 – 15153, 2020.

[15] K. Kadam, S. Ahirrao, H. Kaur, P. Shraddha and A. Pawar, "Deep Learning Approach for Prediction of Pneumonia," Int. J. of Scientific and Technology Research, vol. 8, no. 10, pp. 2986–2989, 2019.

[16] S. Yoo, I. Gujrathi, M. A. Haider and F. Khalvati, "Prostate Cancer Detection Using Deep Convolutional Neural Networks," Scientific Reports, vol. 9, p. 19518, DOI: 10.1038/s41598-019-55972-4, 2019.

[17] D. S. Kermany et al., "Identifying Medical Diagnoses and Treatable Diseases by Image-based Deep Learning," Cell, vol. 172, no. 5, pp. 1122-1131.e9, DOI: 10.1016/j.cell.2018.02.010, 2018.

[18] M. Toğaçar, B. Ergen, Z. Cömert and F. Özyurt, "A Deep Feature Learning Model for Pneumonia Detection Applying a Combination of mRMR Feature Selection and Machine Learning Models," IRBM, vol. 41, no. 4, pp. 212–222, DOI: 10.1016/j.irbm.2019.10.006, 2020.

[19] B. Almaslukh, "A Lightweight Deep Learning-Based Pneumonia Detection Approach for Energy-Efficient Medical Systems," Wireless Communications and Mobile Computing, DOI: 10.1155/2021/5556635, 2021.

[20] J. E. Luján-García et al., "A Transfer Learning Method for Pneumonia Classification and Visualization," Applied Sciences, vol. 10, no. 8, p. 2908, DOI: 10.3390/app10082908, 2020.

[21] S. Ben Atitallah, M. Driss, W. Boulila, A. Koubaa and H. ben Ghézala, "Fusion of Convolutional Neural Networks Based on Dempster–Shafer Theory for Automatic Pneumonia Detection from Chest X-ray Images," Int. J. of Imaging Systems and Technology, vol. 32, no. 2, pp. 658–672, DOI: 10.1002/ima.22653, 2022.

[22] K. T. Islam, S. Wijewickrema, A. M. Collins and S. O'Leary, "A Deep Transfer Learning Framework for Pneumonia Detection from Chest X-ray Images," Proc. of the 15th Int. Conf. on Computer Vision Theory and Applications, pp. 286-293, DOI: 10.5220/0008927002860293, 2020.

[23] E.Ayan and H. M. Ünver, "Diagnosis of Pneumonia from Chest X-ray Images Using Deep Learning," Proc. of the 2019 Scientific Meeting on Electrical-Electronics & Biomedical Engineering and Computer Science (EBBT), pp. 1-5, DOI: 10.1109/EBBT.2019.8741582, 2019.

[24] S. M. H. Hussain, S. M. Raju and A. R. Ismail, "Predicting Pneumonia and Region Detection from X-ray Images Using Deep Neural Network," arXiv: 2101.07717, DOI: 10.48550/arXiv.2101.07717, 2021.

[25] S. Singh and B. K. Tripathi, "Pneumonia Classification Using Quaternion Deep Learning," Multimedia Tools and Applications, vol. 81, pp. 1743–1764, DOI: 10.1007/s11042-021-11409-7, 2022.

[26] L. Račić, T. Popović, S. čakić and S. Šandi, "Pneumonia Detection Using Deep Learning Based on Convolutional Neural Network," Proc. of the 25th Int. Conf. on Information Technology (IT), pp. 1-4, DOI: 10.1109/IT51528.2021.9390137, 2021.

[27] R. Jain, P. Nagrath, G. Kataria, V. S Kaushik and J. Hemanth D., "Pneumonia Detection in Chest X-ray Images Using Convolutional Neural Networks and Transfer Learning," Measurement, vol. 165, p. 108046, DOI: 10.1016/j.measurement.2020.108046, 2020.

[28] C. Shorten and T. M. Khoshgoftaar, "A Survey on Image Data Augmentation for Deep Learning," Journal of Big Data, vol. 60, Article no. 60, DOI: 10.1186/s40537-019-0197-0, 2019.

[29] N. M. Elshennawy and D. M. Ibrahim, "Deep-Pneumonia Framework Using Deep Learning Models Based on Chest X-ray Images," Diagnostics, vol. 10, no. 9, p. 649, 2020.

[30] K. He, X. Zhang, S. Ren and J. Sun, "Identity Mappings in Deep Residual Networks," Proc. of European Conf. on Computer Vision (ECCV 2016), Part of the Lecture Notes in Computer Science, vol. 9908, DOI: 10.1007/978-3-319-46493-0_38, 2016.

[31] K. He, X. Zhang, Sh. Ren and J. Sun, "Deep Residual Learning for Image Recognition," Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), pp. 770-778. [Online], Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7780459, 2016.

[32] W. Shang, J. Chiu and K. Sohn, "Exploring Normalization in Deep Residual Networks with Concatenated Rectified Linear Units," Proc. of the 31st AAAI Conference on Artificial Intelligence (AAAI-17), vol. 31, no. 1, pp. 509-1516, DOI: 10.1609/aaai.v31i1.10759, 2017.

[33] J. Bjorck, C. Gomes, B. Selman and K. Q. Weinberger, "Understanding Batch Normalization," arXiv: 1806.02375, DOI: 10.48550/arXiv.1806.02375, 2018.

[34] A.Khan, A.Sohail, U. Zahoora and A. S. Qureshi, "A Survey of the Recent Architectures of Deep Convolutional Neural Networks," Artificial Intelligence Review, vol. 53, pp. 5455-5516, DOI: 10.1007/s10462-020-09825-6, 2020.

[35] A. Desarda, "Build a Custom ResNetV2 with the Desired Depth from Scratch," Towards Data Science, [Online], Available: https://towardsdatascience.com/build-a-custom-resnetv2-with-the-desired-depth-92892ec79d4b, 2020, Last Accessed 31/10/2022.

[36] G. Huang, Z. Liu, L. Van der Maaten and K. Q. Weinberger, "Densely Connected Convolutional Networks," Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), pp. 2261-2269, DOI: 10.1109/CVPR.2017.243, 2017.

[37] R. Kundu, R. Das, Z.W. Geem, G-T Han and R. Sarkar, "Pneumonia Detection in Chest X-ray Images Using an Ensemble of Deep Learning Models," PLoS ONE, vol. 16, no. 9, p. e0256630, DOI: 10.1371/journal.pone.0256630, 2021.

[38] K. El Asnaoui, Y. Chawki and A. Idri, "Automated Methods for Detection and Classification Pneumonia Based on X-Ray Images Using Deep Learning," Proc. of the Artificial Intelligence and Blockchain for Future Cybersecurity Applications, Part of the Studies in Big Data Book Series, vol. 90, DOI:10.1007/978-3-030-74575-2_14, 2021.

[39] Q. Wu and F. Wang, "Concatenate Convolutional Neural Networks for Non-intrusive Load Monitoring across Complex Background," Energies, vol. 12, no. 8, p. 1572, DOI: 10.3390/en12081572, 2019.

[40] J. Brownlee, "How to Choose an Activation Function for Deep Learning," Machine Learning Mastery, [Online], Available: https://machinelearningmastery.com/choose-an-activation-function-for-deep-learning/, 2021, Last Accessed 31/10/2022.

[41] J. Collis, "Glossary of Deep Learning: Bias," Deeper Learning, [Online], Available https://medium.com/deeper-learning/glossary-of-deep-learning-bias-cf49d9c895e2, 2017, Last Accessed on 31/10/2022.

[42] P. Mooney, "Chest X-ray Images (Pneumonia)," Kaggle, [Online], Available: www.kaggle.com/paultimothymooney/chest-xray-pneumonia, 2018, Last Accessed on 31/10/2022.

[43] M. Vakili, M. Ghamsari and M. Rezaei, "Performance Analysis and Comparison of Machine and Deep Learning Algorithms for IoT Data Classification," arXiv: 2001.09636v1, DOI: 10.48550/arXiv.2001.09636, 2020.

[44] G. Labhane, R. Pansare, S. Maheshwari, R. Tiwari and A. Shukla, "Detection of Pediatric Pneumonia from Chest X-ray Images using CNN and Transfer Learning," Proc. of the 3rd Int. Conf. on Emerging Technologies in Computer Engineering: Machine Learning and Internet of Things (ICETCE), pp. 85-92, DOI: 10.1109/ICETCE48199.2020.9091755, 2020.

[45] A. Mabrouk, R.P. Díaz Redondo, A. Dahou, M. Abd Elaziz and M. Kayed, "Pneumonia Detection on Chest X-ray Images Using Ensemble of Deep Convolutional Neural Networks," Applied Sciences, vol.

12, p. 6448, DOI: 10.3390/app12136448, 2022.
[46] T. Rahman et al., "Transfer Learning with Deep Convolutional Neural Network (CNN) for Pneumonia Detection Using Chest X-ray," Applied Sciences, vol. 10, p. 3233, DOI: 10.3390/app10093233, 2020.
[47] S. Pallawi and D. Kumar Singh, "Preview and Analysis of Deep Neural Network for Alzheimer's Disease Classification using Brain Medical Resonance Imaging," Cognitive Computation and Systems, vol. 5, no. 1, pp. 1-13, DOI: 10.1049/ccs2.12072, 2022.
[48] D. K. Jain, T. Singh, P. Saurabh, D. Bisen, N. Sahu, J. Mishra and H. Rahman, "Deep Learning-aided Automated Pneumonia Detection and Classification Using CXR Scans," Computational Intelligence Neuroscience, vol. 2022, Article ID 7474304, DOI: 10.1155/2022/7474304, 2022.
[49] M. Trivedi and A. Gupta, "A Lightweight Deep Learning Architecture for the Automatic Detection of Pneumonia Using Chest X-ray Images," Multimedia Tools Applications, vol. 81, pp. 5515–5536, DOI: 10.1007/s11042-021-11807-x, 2022.
[50] P. Szepesi and L. Szilágyi, "Detection of Pneumonia Using Convolutional Neural Networks and Deep Learning," Biocybernetics and Biomedical Engineering, vol. 42, no. 3, pp. 1012-1022, DOI: 10.1016/j.bbe.2022.08.001, 2022.

ملخص البحث:

يُعدّ مرض ذات الرّئة من الأمراض المهدّدةِ للحياة، حيث يمكن للكشْف المبكّر عنه إنْقاذ الأرواح. وهناك العديد من الأنظمة الأوتوماتيكية التي ساهمت في الكشْف عن الإصابة بهذا المرض، وأصبحت النّماذج القائمة على التّعلُّم العميق أوسع النّماذج انتشاراً التي تُستخدم في بناء أنظمة الكشْف هذه.

في هذه الدراسة يتمّ جمْع نموذجين من نماذج التعلُّم العميق واستخدامها في الكشْف عن مرض ذات الرّئة. وقد تمّ استخدام طريقتين للتّعامل مع مشكلة عدم توازُن البيانات، وهما: وَزْن البيانات بحيث تمكّن هذه الطّريقة من التحكُّم في النِّسبة المئوية من البيانات الأصلية التي يُمكن استخدامها لكلّ صِنْفٍ من البيانات، وإعادة أخْذ العيّنات وفيها يتمّ إنتاج صورٍ معدَّلة ذات توزيعٍ متساوٍ للبيانات.

تم تقييم النّموذج المقترح باستخدام مجموعة بيانات متوازنة تتكون من (5856) صورة. وكانت النّتائج واعِدَة مقارنةً بنماذج استُخدمت في دراساتٍ سابقة؛ إذ كانت نسبة الضَّبْط (98%)، والمساحة تحت المنحنى (97%)، بينما بلغت قيمة الفَقْد (0.23).

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 09, No. 02, June 2023.

STATEFUL LAYERED CHAIN MODEL TO IMPROVE THE SCALABILITY OF BITCOIN

Dalia Elwi[1, *], Osama Abu-Elnasr[1], A. S. Tolba[1, 2] and Samir Elmougy[1]

(Received: 19-Jan.-2023, Revised: 26-Mar.-2023, Accepted: 1-Apr.-2023)

ABSTRACT

Bitcoin becomes the focus of scientific research in the modern era. Blockchain is the underlying technology of Bitcoin because of its decentralization, transparency, trust-less and immutability features. However, blockchain can be considered the cause of Bitcoin scalability issues, especially storage. Nodes in the Bitcoin network need to store the full blockchain to validate transactions. Over time, the blockchain size will be bulky. So, the full nodes will prefer to leave the network. This leads to the blockchain being centralized and trusted and the security will be adversely affected. This paper proposes a Stateful Layered Chain Model based on storing accounts' balances to reduce the Bitcoin blockchain size. This model changes the structure of the traditional blockchain from blocks to layers. The experimental results demonstrated that the proposed model reduces the blockchain size by about 50.6 %. Implicitly, the transaction throughput can also be doubled.

KEYWORDS

Bitcoin, Cryptocurrency, Blockchain, Stateful, Layered, Chain, Scalability, Storage, Throughput.

1. INTRODUCTION

Blockchain is a distributed, secured, scalable and transparent ledger that permanently records transactions among decentralized network participants [1]. It avoids relying on a trusted third party to validate, verify and process the transactions. Besides cryptocurrency, blockchain is a base technology for many other fields, such as medicine, social, the internet of things, supply chain, voting, information sharing and file storage.

The digital asset, which is cryptographically secured, is called cryptocurrency. Cryptocurrency is a blockchain-based technology to transfer funds between users in a secure and transparent mode [2]. Currently, there are thousands of cryptocurrencies, but according to the coin market cap [3], the best cryptocurrency is Bitcoin [4].

Bitcoin depends on blockchain technology to store transactions between users in a distributed network. Full nodes in Bitcoin store the entire blockchain starting from the genesis block, so that they can retrieve historical activities, search transactions, validate new transactions and calculate balances [5]. Over time, the size of the blockchain will grow exponentially and it will be difficult for full nodes to download and store the complete Bitcoin blockchain. Figure 1 shows the growth of Bitcoin blockchain size from 2009 to 2022. According to Bitcoin visuals' statistics [6], In March 2022, the number of Bitcoin blocks is over 71×10^4, the chain size is 366 GB and in the last two years, the annual growth rate is around 15.8 %. As a result of this huge storage, there is a big reduction in the number of full nodes that have enough resource capabilities to store the entire blockchain. Therefore, the blockchain network will be more centralized.

Bitcoin transactions are structured to make the blockchain stateless, which means that it does not keep the addresses' balances. However, the balances can be retrieved by tracking the address transactions through the entire blockchain. So, we propose a Stateful Layered Chain Model based on storing the addresses' balances. In the proposed model, the transaction and block structures are completely changed. The block is replaced by a layer that is smaller in size. Our model reduces the chain size to solve the storage scalability problem, keeps transactions easy to be verified and implicitly increases the transaction throughput. Our model allows full nodes with limited storage capabilities to process new transactions without the need for storing the entire ledger. Therefore, it maintains the decentralization of the network and at the same time keeps the transaction verification simple as in

1. D. Elwi (Corresponding Author), O. Abu-Elnasr, A.S. Tolba and S. Elmougy are with Computer Science Department, Faculty of Computers and Information, Mansoura University, Mansoura 35516, Egypt. Email: dalia_elwi@mans.edu.eg
2. A.S. Tolba is with New Heliopolis Engineering Institute, Cairo 11829, Egypt.

traditional Bitcoin. Also, our model can increase the transaction throughput implicitly if we make the layer size similar to the block size, allowing the storage of more transactions.

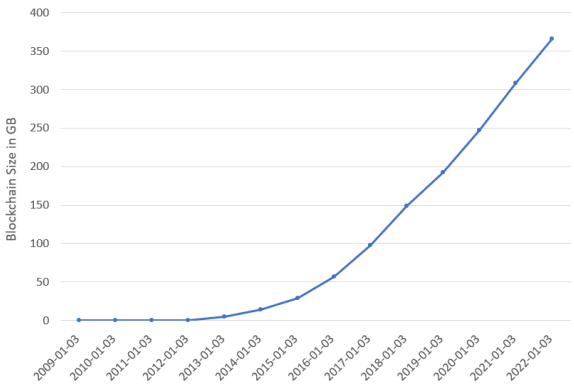


Figure 1. The cumulative size of Bitcoin blocks [6].

The paper is organized as follows: Section 2 gives a simple background about Bitcoin blockchain structure. Section 3 introduces the related work. Section 4 explains the proposed Stateful Layered Chain Model. Section 5 shows the implementation and experimental results. Section 6 provides the conclusion and future work.

2. BACKGROUND

A peer-to-peer network is used in Bitcoin for block and transaction exchange. Peers have three types: Simplified Payment Verification (SPV) node, Light-weight Client node and Full node. SPV node, which is called a *thin client*, is a client that keeps a copy of the block headers. It uses Merkle root to ensure that the block is containing the required transactions [7]. The light-weight Client node does not store the blockchain. Instead, it creates purchase or money transfer transactions and sends them to full nodes [8]. Full nodes, which are called *miners*, store and serve the entire blockchain to guarantee the highest level of security by verifying all the blocks from the genesis block [5]. They are responsible for transaction authentication, the mining process to create new blocks and getting rewards.

2.1 Blockchain Structure

Blockchain is a linked list of blocks. Each block has a pointer to the previous block. The block is divided into two parts: block header and block body [9]. Block header contains information about the block, such as its version, its hash, previous block hash, Merkle root, timestamp, difficulty and Nonce, as illustrated in Table 1. On the other hand, transactions are recorded in the block body [10]. Figure 2 shows the Bitcoin blockchain structure.

Table 1. Fields of the block header.

Fields	Description
Version	The software version number is used for upgrading the protocol.
Hash	It is a 256-bit unique binary number to represent the block.
Previous Hash	It is a hash number of the previous block.
Merkle Root	The root of the Merkle Tree [11] is generated from the transactions in the block.
Timestamp	The time of the block creation is presented in UNIX time.
Difficulty	The difficulty value is used in the mining process to create a new block.
Nonce	It is a counter-value that makes the block hash less than or equal to the target value.

2.2 Transaction Structure

The transaction is a transfer of value from a source address *(input)* to a destination address *(output)*. It has six fields: version, input counter, inputs, output counter, outputs and lock time [12], as illustrated in Table 2. One of the most important concepts of Bitcoin is privacy, which means that the user's balance should be unknown to the network's miners. Bitcoin maintains this concept by relying on a stateless blockchain ledger that does not record any balances.

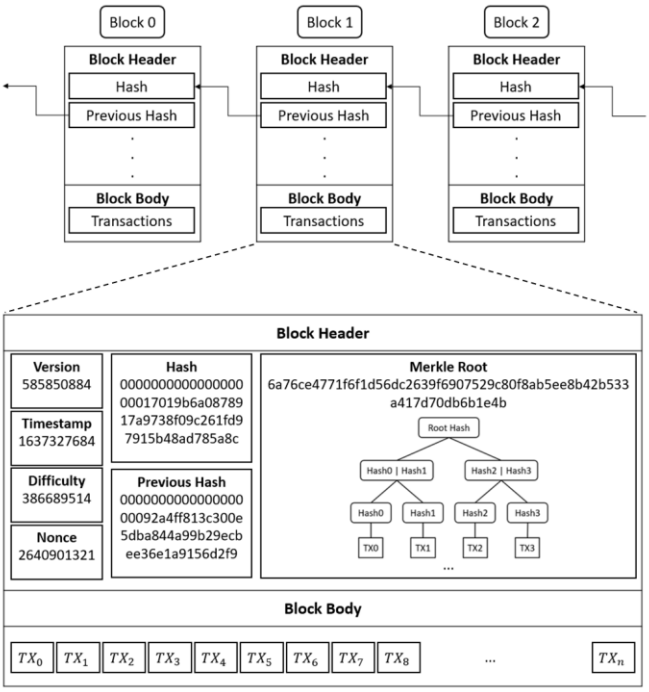


Figure 2. Bitcoin blockchain structure.

Table 2. Fields of transaction structure.

Fields	Description
Version	It is a version number of the rules that a transaction follows.
Input Counter	The number of inputs.
Inputs	One or more transaction inputs.
Output Counter	The number of outputs.
Outputs	One or more transaction outputs.
Lock Time	The locked duration is where the outputs of the transaction cannot be spent. It can be UNIX time or block number.

The user's balance is implicitly divided into many Unspent Transaction Outputs ($UTXO_s$) which can be used as inputs in other transactions, provided that the user can spend only the $UTXO_s$ belonging to him/her. Despite that, the user's balance can be calculated by scanning the entire blockchain and accumulating all $UTXO_s$ belonging to him/her. In addition to the entire blockchain, a set of $UTXO_s$ is stored in full nodes which are used in transaction authentication. Figure 3 shows the mechanism of Bitcoin transactions. Transaction input is composed of four fields: outpoint, unlocking script size, unlocking script and sequence number. Transaction output is composed of three fields: amount, locking script size and locking script [12]. Table 3 and Table 4 illustrate both transaction input and output, respectively.

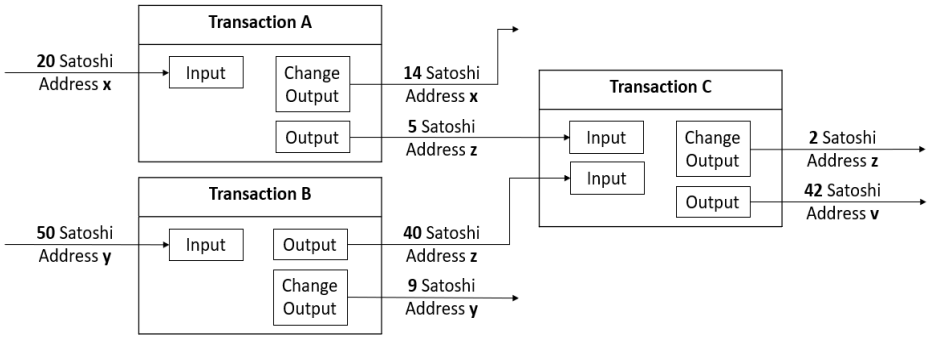


Figure 3. Bitcoin transactions*.

**Transaction A transform 5 Satoshi from address x to address z, return 14 Satoshi to himself and the fee is only 1 Satoshi. Transaction B transforms 40 Satoshi from address y to address z, returns 9 Satoshi to himself and the fee is only 1 Satoshi. Transaction C transforms 42 Satoshi from address z to address v, returns 2 Satoshi to himself and the fee is only 1 Satoshi.*

Table 3. Fields of the transaction input structure.

Fields	Description
Outpoint	It is a pointer to a previous transaction containing the UTXO to be spent. It is composed of the previous transaction hash and UTXO index.
Unlocking Script Size	The length of unlocking scrip in bytes.
Unlocking Script	It is used to unlock the locking script of UTXO. It is a proof of the ownership of the locking script.
Sequence Number	It is a number used to verify the lock time.

Table 4. Fields of the transaction output structure.

Fields	Description
Amount	The transferred value in Satoshi.
Locking Script Size	The length of locking scrip in bytes.
Locking Script	It is used to lock the transferred amount, so that only those with the unlocking script can open the lock.

2.3 Transaction Authentication

When the new transaction is created, miners search the entire blockchain to find the $UTXO_s$ that are used as inputs for this new transaction. So, miners need to store the entire blockchain. Then, UTXO must be authenticated to ensure the sender's ownership of the transferred funds.

2.3.1 Keys and Addresses

Public and private keys are the main requirements for creating any transaction. The private key is a 256-bit binary number that is generated randomly using Secure Hash Algorithm SHA256 [13]. On the other hand, a public key is a unique number that is calculated from a private key using a one-way function called Elliptic Curve Multiplication function [14]. So, the private key cannot be calculated from the public key, but the connection between them can be proved using the digital signature without having to reveal the private key [15]. Bitcoin addresses are calculated by hashing public keys.

2.3.2 Digital Signature

A digital signature is a number that is calculated from the private key and the transaction data. It is used to prove the ownership of the outputs, which are used in the transaction as inputs [16]. The authentication process has two steps: *(1) signing* for digital signature creation and *(2) verifying* to prove that the digital signature and the public key were generated from the same private key [17]. The signing and verifying steps are shown in Figure 4.

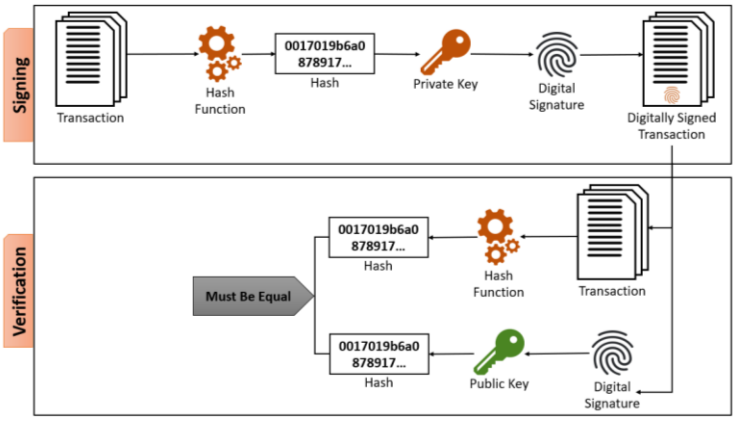


Figure 4. Authentication process.

2.3.3 Scripts

Execution of Bitcoin scripts is based on Reverse Polish Notation which uses the stack to perform this execution [18]. Every transaction contains locking and unlocking scripts, which are called *scritPubKey* and *scriptSig*, respectively. They are executed as one script. If the execution result is true, then the transaction is valid. Standard types of transaction scripts are: Pay to Public Key Hash *(P2PKH)*, Public

Key, Multi-signature, Pay to Script Hash *(P2SH)* and Data Output *(Op-Return)*. Most Bitcoin transactions are based on the *P2PKH* script [19]. The locking and unlocking scripts for *P2PKH* are shown in Figure 5.

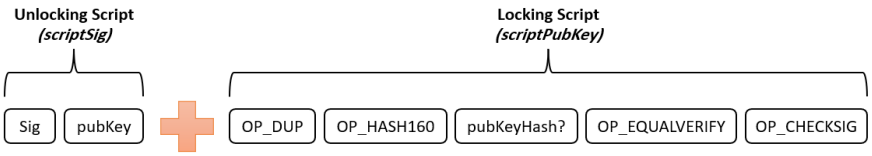


Figure 5. *P2PKH* locking and unlocking scripts [20]*.

**scriptSig is provided by the transaction inputs to unlock the scritPubKey of some previous UTXOs.*

2.3.4 Transaction Fee

The miner creating a new block and recording it into the blockchain should take a fee for each transaction in this new block. The fee can be determined as a transaction output during the creation of the transaction. Also, if it is not determined directly in the transaction, it can be calculated by the miner during the creation of the block. The fee is calculated by subtracting the amount of transaction outputs from the total value of transaction inputs, where the remainder would be the fee [12]. If the value of inputs exceeds the amount of the output, the difference in value is considered the transaction fee. Therefore, the output, which is called "Change Output", must be added to the transaction outputs to force the miner to get a specific value of fee [21]. The change output amount is returned to the sender's address. So, the fee can be calculated as in the following equation:

$$Fee = Inputs\ Value - (Outputs\ Amount + Change\ Output\ Amount)$$

The value of the transaction fee is calculated automatically by the wallet. Miners choose the transactions with the highest fee to be added to the block.

3. RELATED WORK

Based on the view of the blockchain storage expansion problem, the existing suggested models to solve this problem are divided into two categories: *off-chain* solutions and *on-chain* solutions.

3.1 Off-chain Solutions

Off-chain solution is based on storing a large amount of block data in the off-chain storage to reduce the size of the main blockchain. The off-chain storage is maintained by a secondary network. So, there are two networks: one for the off-chain and the second for the main chain. Intercommunication between the two networks adds an overhead to the system's efficiency.

Guy et al. [22] used the Distributed Hash Table (DHT) to store the essential data. On the other hand, the SHA-256 hashes are recorded in the main blockchain to reference DHT data. DHT is maintained by special nodes that differ from the miner nodes. So, this method requires some amount of third-party trust. Qiuhong et al. [23] used a distributed storage called InterPlanetary File System (IPFS) to store the main transaction data. On the other hand, the block contains the transaction hashes as a reference to the IPFS transactions. In this model, a little reduction in the system efficiency is caused as a result of the transaction requests from IPFS nodes. Also, it needs to trust another network that stores IPFS data. Soharab et al. [24] proposed two types of blocks: raw block to store transaction data and hash block to store transaction hashes. Both are connected by a Content Identifier (CID). Raw blocks are stored in the IPFS network as a secondary blockchain and hash blocks are stored in the main blockchain. Raw and hash blocks are created by the miners. So, this model is more complex than the traditional Bitcoin blockchain model. Joseph and Thaddeus [25] proposed a network of micropayment channels to process massive transactions off-chain. The channel is created between two users to exchange funds between them more than once. After the restriction on the timestamp is satisfied, only one transaction is created with the state of the users' balances and broadcast to the main chain. Xiaoqing et al. [26] proposed an Efficient Storage Scheme (ESS) based on the distribution of UTXOs in Bitcoin blocks. ESS assigns a UTXO weight for each block according to the number of its UTXOs and its height. Blocks with higher UTXO weight will keep the complete block information, while

blocks with lower UTXO weight will only save the block header and delete the block body. The deleted $UTXO_s$ are stored in a new database in the form of snapshots called *deleteutxo*. When the new transaction used UTXO from a pruned block as input, the *deleteutxo* database is searched to determine whether the input has an unused record. This scheme causes an additional cost, but it is still in an acceptable range.

3.2 On-chain Solutions

On-chain solution is based on changing the structure of the blockchain to reduce its size. So, there is only one network to store the main blockchain in a different structure. Some solutions are just by pruning the block data and some of them are based on completely changing the blockchain structure. Roman et al. [27] presented a pruning scheme called Coin-Prune. It is based on having honest miners to create a snapshot of the current blockchain state using the UTXO set, periodically. Miners announce and reaffirm the snapshots publicly. Therefore, they do not have to store all historical data. Instead, they rely on the recent snapshot. This scheme has a security limitation; e.g. the potential for removing illicit content from the UTXO set. Also, it needs trust assumptions. Martin et al. [28] demonstrated Functionality-Preserving Local Erasure (FPLE) to erase undesirable transaction outputs from the blockchain and store their references in an erasure database. The proposed FPLE ensures that all nodes are synchronized with the network even when the erased outputs are later spent. It assumes that nodes ignore the unconfirmed transaction the validation of which depends on erased data. A new strategy for transaction verification is proposed to deal with transactions that are relying on erased data. Xiaohai et al. [29] proposed a new jigsaw-like data-reduction approach called Jidar. Each node in Jidar does not need to store the complete block. Instead, it only stores the transactions of interest and the relevant Merkle branches. So, the proof is needed from the transaction proposer to verify the new transactions. Jidar also provided a mechanism to get a complete block based on cohering all the data fragments from the other nodes just like stitching all the pieces into a complete jigsaw puzzle picture. There are not any trust assumptions in Jidar, but it does not maintain the decentralization concepts of the traditional Bitcoin system. Serguei [30] suggested changing the structure of the blockchain from the linked list of blocks to a Directed Acyclic Graph (DAG) ledger. He used this structure to implement IOTA cryptocurrency. DAG ledger is consisting of transaction units. Each unit presents a single transaction from a single user. The new transactions need to approve the legitimacy of two previous transactions. In this structure, nodes do not require a large storage to store the complete DAG. In contrast, they need a higher coding requirement to load the DAG ledger. Li et al. [31] proposed an algorithm to downsample the block body to reduce the blockchain size. This algorithm is based on the distribution of the interval between the approval of a UTXO and its use. So, based on this analysis, transactions are discarded after a period decided by the algorithm. This method loses very little broadcast accuracy to reduce storage. Ryunosuke et al. [32] proposed a new architecture called Trail to reduce the size of the blockchain. The trail allows nodes to store the block without transactions. In contrast, a client who creates a transaction needs to send proof of its assets. So, unlike traditional blockchains, the clients store additional data. Ulfah et al. [33] summarized the block and compressed the result to reduce the blockchain size. At the same time, they make transaction verification easier than traditional Bitcoin. They developed the summarization by rearranging the block headers and adding parameters according to this arrangement. After that, the summary block is compressed with the deflate compression algorithm which is a composite of LZ77 and Huffman algorithms. The deflate compression algorithm is used to overcome the problems that occurred due to the summarization.

4. PROPOSED SOLUTION

4.1 Stateful Layered Chain Model

A Stateful Layered Chain Model is proposed to reduce the size of the Bitcoin blockchain by changing the transaction structure and the block structure. Like a blockchain, a Stateful Layered Chain is a distributed ledger that is recorded in different miners in a peer-to-peer network. All miners must have the same replica of the ledger to avoid reliability. So, our model is not relying on third parties. $UTXO_s$ set is no longer stored. Instead, accounts with their balances are recorded. Transactions are no longer based on inputs and outputs. However, they are based on receivers and senders. Table 5 presents the proposed transaction structure.

In our model, there are two types of layers: *Last State Layer* (LSL) and *Default State Layer* (DSL). LSL stores the state of all accounts, while DSL stores the transactions. LSL is only one layer that is

Table 5. Fields of transaction structure.

Fields	Description
Sender Address	It is a sender account public key in a short format.
Sender Unlocking Script	It is a script used to unlock the locking script of the sender account balance to authenticate the ownership.
Receiver Address	It is a receiver account public key in a short format.
Receiver Locking Address	It is a script used to lock the transferred value, so that only those with the unlocking script can open the lock.
Value	It is the transferred value in Satoshi.

updated every time a DSL is created to save the last state for each account. DSL is the alternative form of a block to store transactions, knowing that DSL reduces the size of the block and at the same time keeps the number of transactions as in the block. Stateful Layered Chain is composed of a linked list of DSL layers. So, each DSL layer points to the previous one. Figure 6 and Figure 7 show the proposed LSL and DSL structures in JavaScript Object Notation (JSON) format, respectively. Table 6 and Table 7 illustrate the DSL and LSL structures.

Figure 6. The proposed LSL structure in JSON format.

Table 6. Fields of LSL structure.

Fields	Description
Hash	It is a 256-bit unique binary number to represent the LSL layer.
n Accounts	The number of client accounts.
Size	The length of the LSL in bytes.
Accounts	Client accounts.

Table 7. Fields of DSL structure.

Fields	Description
Hash	Hash number of the updated LSL layer.
Previous Hash	Hash number of the previous DSL layer.
Fee	It is the amount of fee that the miner gets.
n Receivers	The number of receiver accounts.
Nonce	It is a counter value that makes the block hash less than or equal to the target value.
Size	The length of the DSL in bytes.
Height	The height of the DSL in the chain.
Receivers	Receiver accounts.

After transactions are broadcast through the network, they must be authenticated. The transaction is authenticated by using the sender *unlocking script* to unlock the *locking script* of his/her account contained in LSL to guarantee the ownership of the transferred funds. As in the traditional Bitcoin, the

Proof of Work (PoW) consensus algorithm is used in the mining process. PoW is used to ensure network consistency, ensure a high level of security and obtain approval on the new DSL from all miners. DSL transactions must guarantee the following conditions:

1) There are no two receivers with the same address.
2) Receiver address is not the same as one of the senders' addresses.
3) Sender cannot send funds greater than or equal to his/her balance.
4) 0Sender account must be included in the LSL.


```
"Hash": "00000000000000000000017019b6a0878917a9738f09c261fd97915b48ad785…",
"Previous Layer Hash": "00000000000000000000092a4ff813c300e5dba844a99b29ecb…",
"Fee": 1000,
"n Receivers": 3,
"Nonce": 2640901321,
"Size": 500,
"Height": 150001,
"Receivers": [
{
        "Address": "1Hc5hmiZyx9FYY9hAV4MMBgsjJzwLoMUmP",
        "Locking Script": "76a914536ffa992491508dca0354e52f32a3a7a679a53a88ac",
        "n Senders": 2,
        "Senders": [
        {
                "Address": "35iMHbUZeTssxBodiHwEEkb32jpBfVueEL",
                "Unlocking Script": "0315d70a082f5669614254432f2cfabe6d6da926…",
                "Value": "100000"
        },
        ……//other Senders //……
        ]
},
……//other Receivers//……
]
```


Figure 7. The proposed DSL structure in JSON format.

Table 8, Table 9 and Table 10 illustrate LSL account, DSL receiver and DSL sender structures, respectively. Stateful Layered Chain Model provides some concepts, including:

1) **Layer Reduction:** Since the transaction authentication process needs only the LSL to check the balance and to ensure the sender's ownership of the transferred funds, miners can remove the earliest DSL layers from the chain.
2) **Node Bootstrap:** New full nodes just need to download the LSL to begin the mining process.
3) **User Privacy:** For more privacy, users can split their balance into many accounts with different addresses.

Table 8. Fields of LSL account structure.

Fields	Description
Address	It is an account public key in a short format.
Locking Script	It is a script used to lock the balance value, so that only those with the unlocking script (owner account) can open the lock.
Balance	It is the amount value in Satoshi owned by the account.

Table 9. Fields of DSL receiver structure.

Fields	Description
Address	It is a receiver account public key in a short format.
Locking Script	It is a script used to lock the transferred amount, so that only those with the unlocking script can open the lock.
n Senders	It is the number of senders sending funds to the receiver account.
Senders	Sender accounts.

Table 10. Fields of DSL sender structure.

Fields	Description
Address	It is a sender account public key in a short format.
Unlocking Script	It is a script used to unlock the locking script of the account balance to authenticate the ownership.
Value	It is the transferred value in Satoshi.

4.2 Stateful Layered Chain Creation Methodology

Stateful Layered Chain starts with the genesis layer which includes the first transaction. The LSL layer is created to present the state of the accounts in the genesis layer. After that, DSL can be created as shown in Figure 8. Miners in the network choose transactions from the Memory Pool for validation and authentication. Then, the valid transactions are added to DSL as the receiver/sender structure and LSL is updated. The sender balance is decreased and the receiver balance is increased. At the end, a miner who calculates the hash number of LSL firstly broadcasts the DSL to the rest of the network.

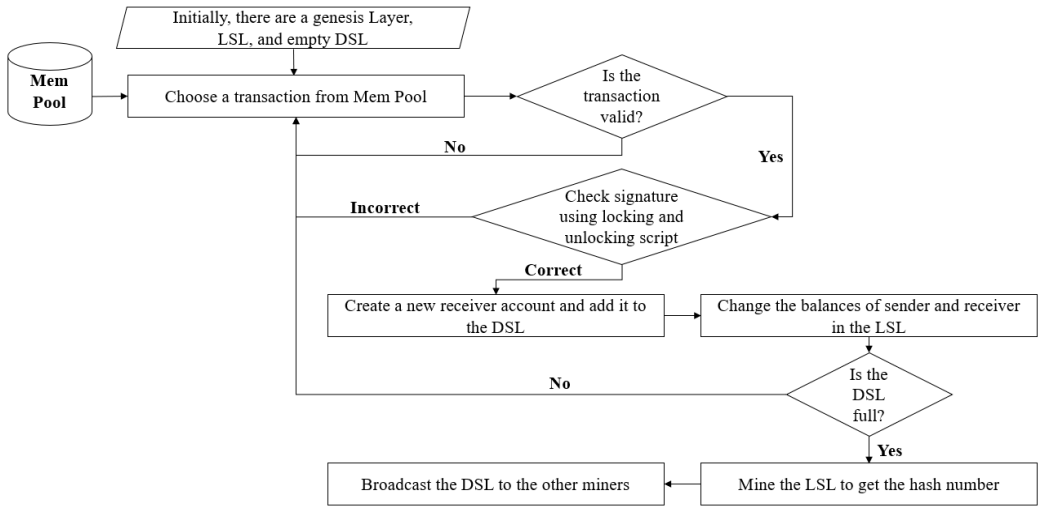


Figure 8. DSL creation flowchart.

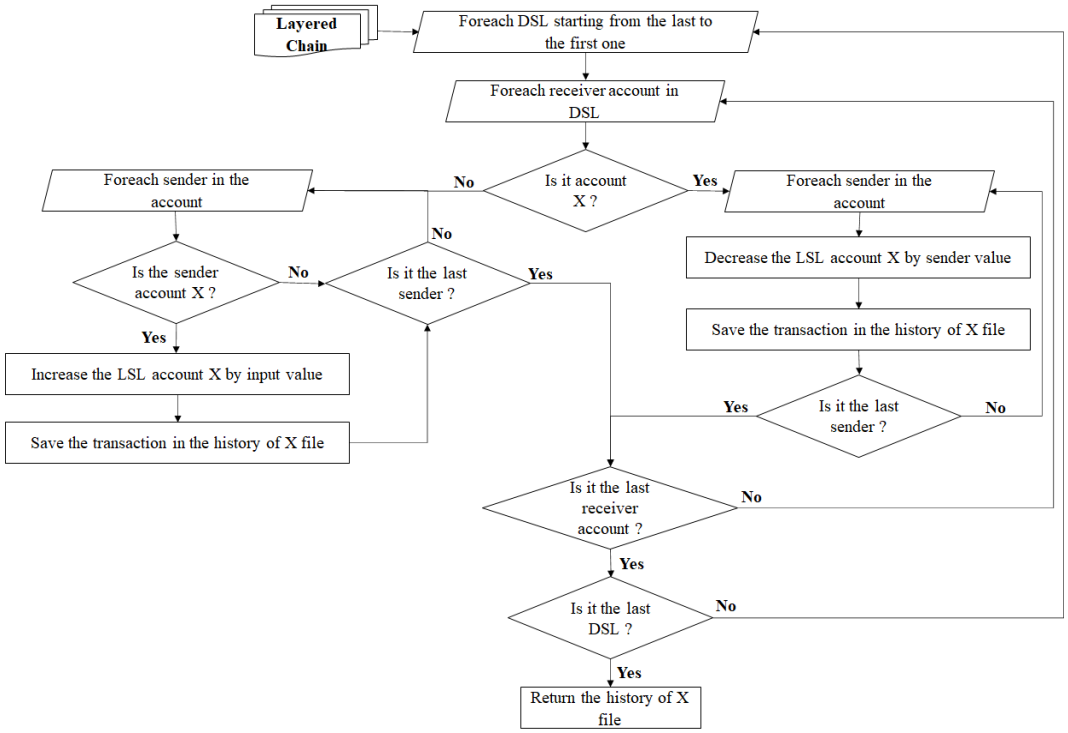


Figure 9. History of account x flowchart.

Stateful Layered Chain Model provides the ability to retrieve the history of LSL accounts. Figure 9 shows the flowchart of how to retrieve the history balances of account x. Also, Stateful Layered Chain Model provides a method to retrieve the LSL at any height of the chain. This method is shown in Figure 10.

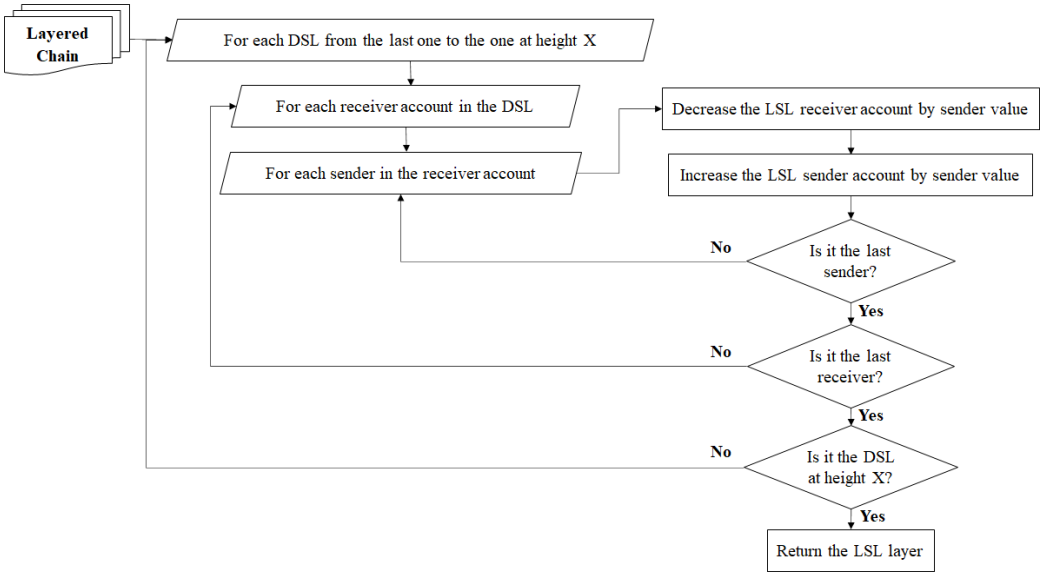


Figure 10. Deriving LSL at height x flowchart.

4.3 Example

The following example demonstrates the main concepts of the proposed model. It supposes that:

1) Initially LSL contains *six* accounts (A, B, C, D, E, F) with a balance equal to 100 Satoshi.
2) There are nine valid transactions in the Memory Pool, as illustrated in Table 11.
3) DSL can record only three transactions.

Figure 11 shows the Stateful Layered Chain before and after adding the transactions in DSL layers.

Table 11. Valid transactions in the memory pool.

Transaction number	Transaction with Amount = 10
1	A to B
2	B to C
3	C to D
4	A to C
5	A to F
6	F to E
7	B to F
8	F to B
9	F to A

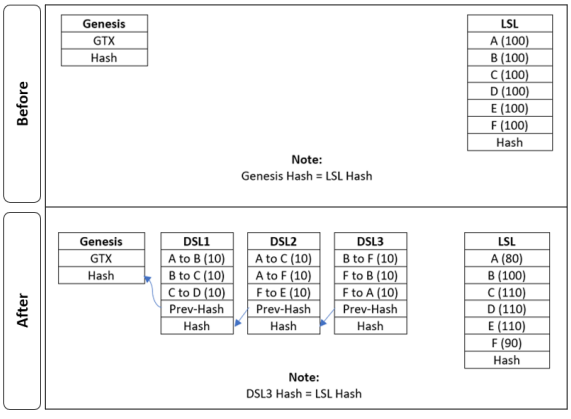


Figure 11. Stateful layered chain before and after DSL layers' creation.

5. IMPLEMENTATION AND EVALUATION RESULTS

5.1 Implementation Environment

The Stateful Layered Chain Model is implemented using 100 blocks from the Bitcoin system. The blocks are from a height of 710327 to 710426 which are the latest blocks at that time. We developed a transformation process using the C# .Net framework to transform the blocks into DSL layers. The transformation process is divided into four stages, as shown in Figure 12.

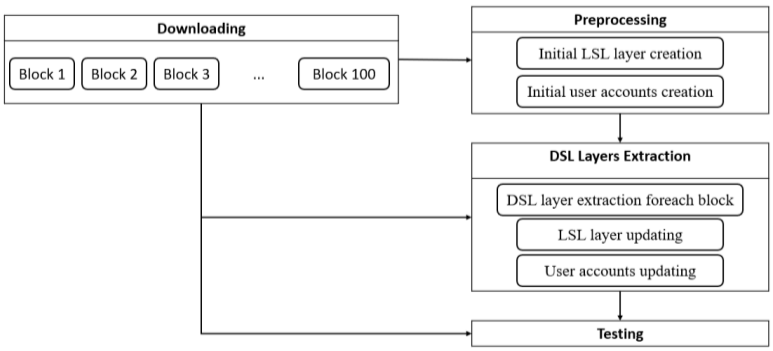


Figure 12. Transformation process stages.

1) **Downloading:** Blockchain data API [34] is used to download a single block in JSON format. Blocks are downloaded individually.
2) **Pre-processing:** In this stage, the initial LSL layer and initial user accounts are derived from the existing blocks. Algorithm 1 and Algorithm 2 are used to create initial LSL and initial user accounts, respectively.
3) **DSL Layer Extraction:** The DSL Layer for each block is extracted from the existing blocks, as presented in Algorithm 3. After the extraction of each DSL layer, the LSL layer and user accounts are updated. Algorithm 4 and Algorithm 5 are used to update LSL and user accounts.
4) **Testing:** User accounts are updated for each block and compared with the corresponding user accounts for each DSL layer to ensure that they have the same balances.

Algorithm 1. Initial LSL layer creation

Input: TXPOOL, Blocks
Variable:
TXPOOl: Pool of all transaction indexes in the existing blocks
Account: Account to be added in LSL
Output: Initial LSL Layer
BEGIN
LSL layer = empty
FOR EACH Block from the oldest to the newest
FOR EACH Transaction in the Block
FOR EACH Input in the Transaction
IF it is not coin base transaction && the input index does not exist in *TXPOOL*
IF there is *Account* with the same input address in the layer
Account value = *Account* value + input value
ELSE
Create a new *Account* with the input address, value and script.
Add *Account* to the LSL layer
END IF
END IF
END FOR EACH
END FOR EACH
END FOR EACH
Calculate the hash number of the Initial LSL Layer
Save the Initial LSL Layer as a file.
END

Algorithm 2. Initial user accounts' creation

Input: Blocks
Variable:
UID: A number to identify the user
User: User account including addresses list and the balance for each address
Output: List of Users
BEGIN
UID= 1
List of Users = empty
FOR EACH Block from the oldest to the newest
FOR EACH Transaction in the Block
Create a new *User* with empty addresses list
FOR EACH Input in the Transaction
IF it is not coin base transaction && an input address does not exist in the *User* addresses list
Add the address to the *User* addresses list
END IF
END FOR EACH
IF the *User* addresses list is not empty
the user number = *UID*
UID = *UID* + 1
Add the *User* to the List of Users
END IF
END FOR EACH
END FOR EACH
Remove redundant users
Define users' balances for their addresses
Save the List of Users as a file
END

Algorithm 3. DSL layer extraction

Input: Block, List of Users, LSL Layer
Variable:
RAccount: Receiver Account to be added in DSL Layer
SAccount: Sender Account that sends funds to the *RAccount*
Output: New DSL Layer, Updated LSL Layer, Updated List of Users
BEGIN
Create New Empty DSL Layer with Zero *RAccount*
FOR EACH Transaction in the Block
IF the Transaction is not coin base transaction
Take a fee from the sender.
END IF
FOR EACH Output in Transaction Output List
IF Output address does not exist in Transaction Inputs addresses
Create New DSL Layer *RAccount*
RAccount Address = Output Address
RAccount Script = Output Script
RAccount Height = Layer Height
IF the Transaction is not coin base transaction
List of Enough Addresses = get enough addresses from the LSL layer
Current Value = 0
FOR EACH Address in the List of Enough Addresses
Create New *SAccount* with the same Address
IF the Address is not the last in the List
Current Value = Current Value + Address Value
SAccount Value = Address Value
ELSE IF the Address is the last in the List
SAccount Value = Output Value – Current Value
END IF

```
                    END FOR EACH
                ELSE IF the Transaction is a coin base transaction
                    Create New SAccount without Address
                    SAccount Value = Output Value
                END IF
                IF DSL Layer contains the RAccount
                    Add the new SAccount to the RAccount
                ELSE IF Layer does not contain the RAccount
                    Add the RAccount to the DSL Layer
                END IF
            Update LSL Layer
            Update User Accounts in the List of Users
            END IF
        END FOR EACH
END FOR EACH
Calculate the hash number of the LSL layer
Save the DSL Layer, LSL Layer and List of Users as files
END
```

Algorithm 4. LSL layer updating

```
Input: LSL Layer, DSL Layer RAccount
Variable:
    RAccount: Receiver Account to be added in DSL Layer
    SAccount: Sender Account that sends funds to the RAccount
    Value: Total amount transferred from SAccounts to RAccount
Output: Updated LSL Layer
BEGIN
Value = 0
FOR EACH SAccount in RAccount
    Value = Value + SAccount Value
    IF SAccount Address is not Empty
        LSL Layer Account = Get LSL Layer Account with the same SAccount Address
        LSL Layer Account Value = LSL Layer Account Value – SAccount Value
        IF LSL Layer Account Value = 0
            Remove LSL Layer Account from the LSL Layer
        END IF
    END IF
END FOR EACH
LSL Layer Account = Get LSL Layer Account with the same RAccount Address
IF there is LSL Layer Account
    LSL Layer Account Value = LSL Layer Account Value + Value
ELSE IF there is no LSL Layer Account
    Create a new LSL Layer Account with the same address, script and value of the RAccount
    Add it to the LSL Layer
END IF
END
```

Algorithm 5. User accounts' updating

```
Input: Sender, List of Users, RAccount
Variables:
    RAccount: Receiver Account to be added in DSL Layer
    SAccount: Sender Account that sends funds to the RAccount
    Value: Total amount transferred from SAccounts to RAccount
    User: User account including addresses list and the balance for each address
Output: Updated List of Users
BEGIN
Value = 0
FOR EACH SAccount in RAccount
    Value = Value + SAccount Value
    IF SAccount Address is not Empty
        Sender Account Value = Sender Account Value – Value
    END IF
```

END FOR EACH
User = Get *User* from the List of Users whose addresses list contains *RAccount* Address
User Account Value = *User* Account Value + *Value*
END

5.2 Experimental Results

5.2.1 Block *vs.* DSL Layer

After testing, we found that user accounts from blocks and layers are identical. At the same time, the DSL layer size is about 51.2 % of the block size and the complete Stateful Layered Chain size is 50.6 % of the complete blockchain size. Figure 13 shows the size of those 100 blocks and the size of the corresponding 100 DSL layers. Figure 14 shows the size of the complete Stateful Layered Chain compared to the blockchain size.

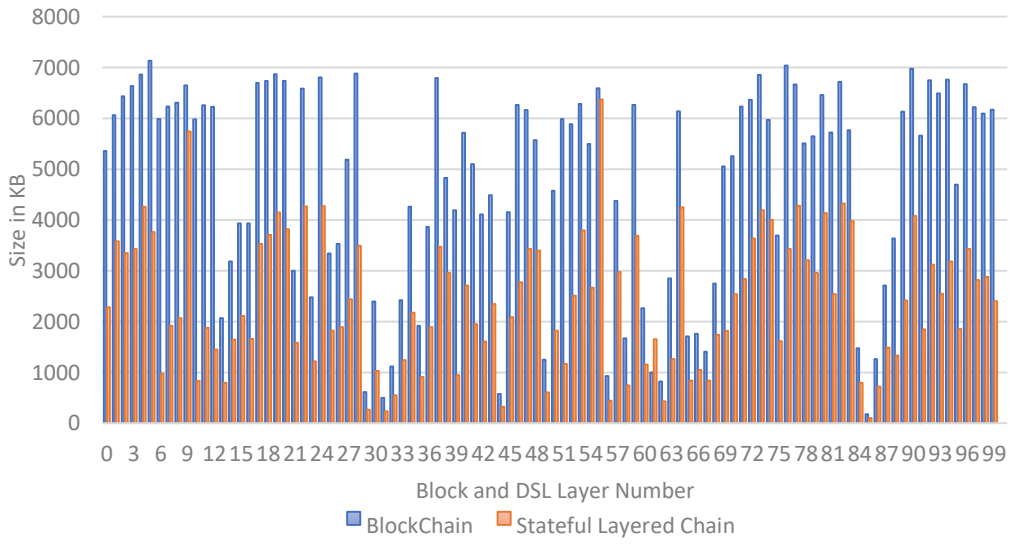


Figure 13. Size of block *vs.* size of DSL layer.

5.2.2 UTXO Pool *vs.* LSL Layer

UTXO Pool is not enough used in the Stateful Layered Chain Model. Instead, there is an LSL layer to present the accounts' balances. Figure 15 shows the UTXO pool size *versus* the LSL layer size after the last block and DSL layer. We observed that the LSL size is 53.2 % of the UTXO pool size.

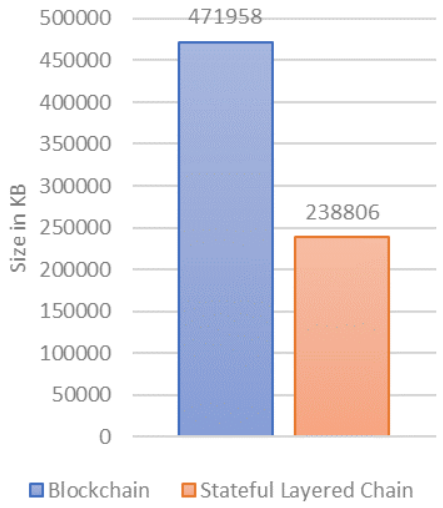


Figure 14. Stateful layered chain size *vs.* blockchain size.

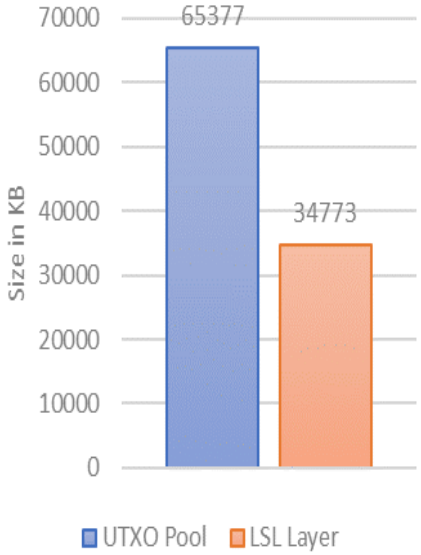


Figure 15. UTXO pool size *vs.* LSL layer size.

5.2.3 Transaction Throughput

According to the concept of layers reduction that is provided by the proposed model, the earliest DSL layers can be removed and only the LSL layer is preserved. So, the size of the DSL layer can be the same as the block size to increase the transaction throughput. Figure 16 shows that transaction throughput is increased if we keep the DSL layer size equal to the block size.

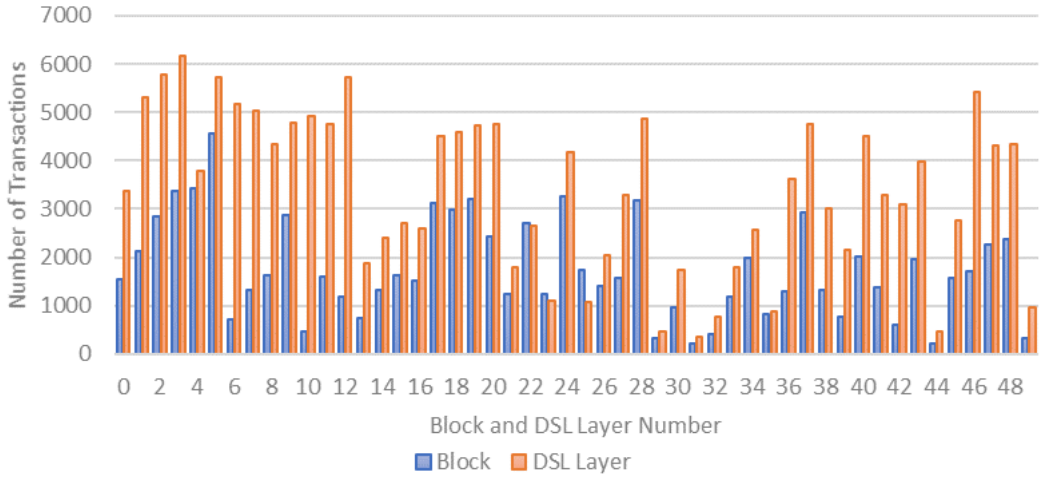


Figure 16. Number of transactions in blocks and DSL layers.

The total number of transactions in a Stateful Layered Chain with 50 DSL layers is shown in Figure 17 compared to those in a blockchain with 50 blocks of the same size as the DSL layers. We noticed that the number of transactions in the Stateful Layered Chain is 1.93 more than those in the blockchain.

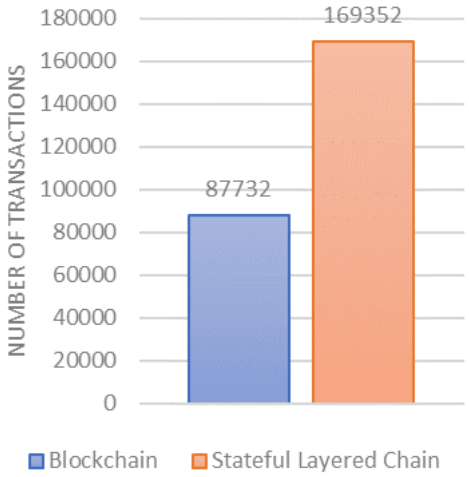


Figure 17. The total number of transactions in the stateful layered chain compared to those in the blockchain.

The results show that:

1. Using a stateful chain rather than a stateless chain causes the size of the stateful layered chain to be 50.6 % smaller than that of the Bitcoin blockchain.
2. The LSL layer size is 53.2 % smaller than the UTXO pool size, whereas the LSL layer records accounts rather than transactions.
3. The transaction throughput of a stateful layered chain can be 1.93 more than that of the Bitcoin blockchain if we keep the DSL layer size equal to the block size.

The Stateful Layered Chain Model shows success in storage scalability and transaction throughput. At the same time, the proposed model had not affected the security and decentralization. However, many of the traditional Bitcoin aspects still exist in the proposed model, like decentralization using distributed network, mining using PoW and authentication using digital signature.

6. CONCLUSION

In this paper, we proposed a Stateful Layered Chain Model which is a completely changed structure of the Bitcoin blockchain. It is no longer based on UTXOs. Instead, it relies on users' balances to send and receive funds. The proposed model enhances Bitcoin's scalability in respect of its storage. It saves approximately a half of the storage, since the Stateful Layered Chain is 50.6 % smaller than the Bitcoin blockchain. Also, the UTXO pool has been replaced with the LSL layer which is about a half the size of the UTXO pool. As the results show, the LSL size is 53.2 % smaller than the UTXO pool size. In addition, the proposed model improves the transaction throughput by nearly twice that of Bitcoin if we keep the DSL layer size equal to the block size.

One of our future research directions is to increase the transaction throughput and decrease the transaction latency by adding the parallel-mining process to the proposed model.

REFERENCES

[1] K. Salah, M. H. U. Rehman, N. Nizamuddin and A. Al-Fuqaha, "Blockchain for AI: Review and Open Research Challenges," IEEE Access, vol. 7, pp. 10127-10149, 2019.

[2] S. P. Yadav, K. K. Agrawal, B. S. Bhati, F. Al-Turjman and L. Mostarda, "Blockchain-based Cryptocurrency Regulation: An Overview," Computational Economics, vol. 59, pp. 1659–1675, 2022.

[3] "CoinMarketCap," [Online], Available: https://coinmarketcap.com/ (accessed).

[4] "Bitcoin," [Online], Available: https://bitcoin.org/en/ (accessed).

[5] F. Tschorsch and B. Scheuermann, "Bitcoin and Beyond: A Technical Survey on Decentralized Digital Currencies," IEEE Communications Surveys & Tutorials, vol. 18, no. 3, pp. 2084-2123, 2016.

[6] "Bitcoin Visuals," [Online], Available: https://bitcoinvisuals.com/ (accessed).

[7] S. Nakamoto, "Bitcoin: A Peer-to-peer Electronic Cash System," Decentralized Business Review, p. 21260, [Online], Available: https://bitcoin.org/bitcoin.pdf, 2008.

[8] S. Park, S. Im, Y. Seol and J. Paek, "Nodes in the Bitcoin Network: Comparative Measurement Study and Survey," IEEE Access, vol. 7, pp. 57009-57022, 2019.

[9] S. Zhao and D. O'Mahony, "Applying Blockchain Layer2 Technology to Mass E-commerce," Cryptology ePrint Archive, [Online], Available: https://eprint.iacr.org/2020/502.pdf, 2020.

[10] S.-W. Chae, J.-I. Kim and Y. Park, "Practical Time-release Blockchain," Electronics, vol. 9, no. 4, p. 672, 2020.

[11] B. Bailey and S. Sankagiri, "Merkle Trees Optimized for Stateless Clients in Bitcoin," Proc. of the Int. Conf. on Financial Cryptography and Data Security, pp. 451-466, Springer, 2021.

[12] V. Vallois and F. A. Guenane, "Bitcoin Transaction: From the Creation to Validation, a Protocol Overview," Proc. of the 2017 1st IEEE Cyber Security in Networking Conf. (CSNet), pp. 1-7, Rio de Janeiro, Brazil, 2017.

[13] N. T. Courtois, M. Grajek and R. Naik, "Optimizing sha256 in Bitcoin Mining," Proc. of the Int. Conf. on Cryptography and Security Systems, pp. 131-144, Springer, 2014.

[14] E. H. Umucu, "Elliptic Curve Cryptography in Blockchain Technology," SSRN, DOI: 10.2139/ssrn. 4033934, 2022.

[15] H. Hellani, A. E. Samhat, M. Chamoun, H. El Ghor and A. Serhrouchni, "On Blockchain Technology: Overview of Bitcoin and Future Insights," Proc. of the 2018 IEEE Int. Multidisciplinary Conf. on Engineering Technology (IMCET), pp. 1-8, Beirut, Lebanon, 2018.

[16] W. Fang, W. Chen, W. Zhang, J. Pei, W. Gao and G. Wang, "Digital Signature Scheme for Information Non-repudiation in Blockchain: A State of the Art Review," EURASIP Journal on Wireless Communications and Networking, vol. 2020, no. 1, pp. 1-15, 2020.

[17] A. I. Badev and M. Chen, "Bitcoin: Technical Background and Data Analysis," [Online], Available: https://www.federalreserve.gov/econresdata/feds/2014/files/2014104pap.pdf, 2014.

[18] P. V. Krtolica and P. S. Stanimirović, "Reverse Polish Notation Method," International Journal of Computer Mathematics, vol. 81, no. 3, pp. 273-284, 2004.

[19] H. Brakmić, Bitcoin and Lightning Network on Raspberry Pi, 1st Ed., ISBN: 978-1-4842-5522-3, Apress Berkeley, CA, pp. XIII, 364, 2019.

[20] "Bitcoindeveloper," [Online], Available: https://developer.bitcoin.org/devguide/transactions.html#.

[21] P. Müller, S. Bergsträßer, A. Rizk and R. Steinmetz, "The Bitcoin Universe: An Architectural Overview of the Bitcoin Blockchain," 11. DFN-Forum Kommunikationstechnologien, Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn, 2018.

[22] G. Zyskind and O. Nathan, "Decentralizing Privacy: Using Blockchain to Protect Personal Data," Proc. of the 2015 IEEE Security and Privacy Workshops, pp. 180-184, San Jose, USA, 2015.

[23] Q. Zheng et al., "An Innovative IPFS-based Storage Model for Blockchain," Proc. of the 2018 IEEE/WIC/ACM Int. Conf. on Web Intelligence (WI), pp. 704-708, Santiago, Chile, 2018.

[24] M. S. H. Sohan et al., "Increasing Throughput and Reducing Storage Bloating Problem Using IPFS and Dual-blockchain Method," Proc. of the 2021 2nd IEEE Int. Conf. on Robotics, Electrical and Signal Processing Techniques (ICREST), pp. 732-736, DHAKA, Bangladesh, 2021.

[25] J. Poon and T. Dryja, "The Bitcoin Lightning Network: Scalable Off-chain Instant Payments," [Online], Available: https://lightning.network/lightning-network-paper.pdf, 2016.

[26] X. Wang, C. Wang, K. Zhou and H. Cheng, "ESS: An Efficient Storage Scheme for Improving the Scalability of Bitcoin Network," IEEE Transactions on Network and Service Management, vol. 19, no. 2, pp. 1191-1202, 2021.

[27] R. Matzutt et al., "How to Securely Prune Bitcoin's Blockchain," Proc. of the IEEE 2020 IFIP Networking Conf. (Networking), pp. 298-306, Paris, France, 2020.

[28] M. Florian, S. Henningsen, S. Beaucamp and B. Scheuermann, "Erasing Data from Blockchain Nodes," Proc. 2019 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW), pp. 367-376, Stockholm, Sweden, 2019.

[29] X. Dai, J. Xiao, W. Yang, C. Wang and H. Jin, "Jidar: A Jigsaw-like Data Reduction Approach without Trust Assumptions for Bitcoin System," Proc. of the IEEE 39th Int. Conf. on Distributed Computing Systems (ICDCS), 2019: IEEE, pp. 1317-1326, Dallas, USA, 2019.

[30] S. Popov, "The Tangle," *White Paper*, vol. 1, no. 3, 2018.

[31] L. Quan, Q. Huang, S. Zhang and Z. Wang, "Downsampling Blockchain Algorithm," Proc. of the IEEE INFOCOM 2019 - IEEE Conf. on Computer Communications Workshops (INFOCOM WKSHPS), pp. 342-347, Paris, France, 2019.

[32] R. Nagayama, R. Banno and K. Shudo, "Trail: An Architecture for Compact UTXO-based Blockchain and Smart Contract," IEICE TRANSACTIONS on Information and Systems, vol. 105, no. 2, pp. 333-343, 2022.

[33] U. Nadiya, K. Mutijarsa and C. Y. Rizqi, "Block Summarization and Compression in Bitcoin Blockchain," Proc. of the 2018 IEEE Int. Symposium on Electronics and Smart Devices (ISESD), pp. 1-4, Bandung, Indonesia, 2018.

[34] "Blockchain Data API," [Online], Available: https://www.blockchain.com/api/blockchain_api.

ملخص البحث:

أصبحت العُملة الرّقمية مجالاً للبحث في أيامنا هذه. وتعدّ تكنولوجيا سلاسل الكُتَل الأساس الّذي تقوم عليه العملة الرّقمية (بِتكوين) نظراً لتمتّعها بعدد من الخصائص، مثل اللامركزية والشّفافية والحَصانة. ومع ذلك، يمكن اعتبار تكنولوجيا سلاسل الكُتَل هي السّبب وراء المسائل المتعلّقة بتوسيع استخدام البتكويْن، وخصوصاً فيما يتعلّق بالتّخزين. وتحتاج العُقَد في شبكة البتكويْن الى تخزين كامل سلسلة الكُتَل لإجراء التّعامُلات. لكن بمرور الوقت، يصبح حجم سلسلة الكُتَل ضخماً. وهكذا فإنّ العُقَد الممتلئة ستفضّل ترْك الشبكة. وهذا يؤدي الى زيادة مركزية سلسلة الكُتَل، ويتأثر أمان الشّبكة بصورةٍ سلبية.

تبحث هذه الورقة في اقتراح نظام سلاسل ذي طبقاتٍ يستند على تخزين أرْصِدة الحسابات من أجْل التّقليل من حجْم سلسلة الكُتَل الخاصّة بالبتكويْن. ويغير النّموذج المقترح البنية التقليدية لسلسلة الكُتَل من كُتَلٍ الى طبقات. وتُظهر نتائج التّجارب أنّ النّموذج المقترح يقلّل حجم سلسلة الكُتَل بنسبة تقرُب من (50.6%)، ويتضمّن ذلك مضاعفة التّعاملات التي يمكن إجراؤها عبر النّظام.

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 09, No. 02, June 2023.

AGENT BASED APPROACH FOR TASK OFFLOADING IN EDGE COMPUTING

Hossein Morshedlou[1] and Reza Vafa Shoar[2]

(Received: 7-Jan.-2023, Revised: 24-Mar.-2023 and 9-Apr.-2023, Accepted: 10-Apr.-2023)

ABSTRACT

Due to limited resource capacity in the edge network and a high volume of tasks offloaded to edge servers, edge resources may be unable to provide the required capacity for serving all tasks. As a result, some tasks should be moved to the cloud, which may cause additional delays. This may lead to dissatisfaction among users of the transferred tasks. In this paper, a new agent-based approach to decision-making is presented about which tasks should be transferred to the cloud and which ones should be served locally. This approach tries to pair tasks with resources, such that a paired resource is the most preferred resource by the user or task among all available resources. We demonstrate that reaching a Nash Equilibrium point can satisfy the aforementioned condition. A game-theoretic analysis is included to demonstrate that the presented approach increases the average utility of the users and their level of satisfaction.

KEYWORDS

Edge computing, Task offloading, Nash equilibrium, Agent, User satisfaction.

1. INTRODUCTION

Offloading is a solution that enhances the capabilities of mobile systems by migrating computations to more resourceful and powerful nearby devices, such as edge nodes, fog nodes and cloudlet or base stations. Nearest nodes, like edge nodes, are the first choice for offloading purposes, because offloading to edge nodes alleviates congestion in cellular networks. Edge node resource capacity may not always allow for the handling of computation-intensive tasks and large data storage from mobile devices. Therefore, such tasks must be transferred to the cloud. However, especially in peak times, due to many offloaded tasks, edge resources may not be able to provide the needed capacity for all tasks. Therefore, inevitably, some tasks should be transferred to the cloud, which may result in additional delays. This may lead to dissatisfaction among users of the transferred tasks. This paper proposes a decentralized approach for task offloading without a central decision-maker component. The proposed approach incorporates user satisfaction level in decision-making about which tasks should be transferred to the cloud and which ones should be served on edge nodes locally. This approach tries to pair tasks with resources, such that a paired resource is the most preferred one by the user or task among all available resources. We show that reaching an NE point can satisfy the mentioned condition. Additionally, a game-theoretic analysis is given to demonstrate how the proposed approach raises the total satisfaction level of users and the acceptability of results by rational users. To the best of the authors' knowledge, this is the first work in the literature that addresses task-offloading strategy optimization based on user satisfaction level. The satisfaction of different users depends on different criteria. Each user understands which decision will maximize his/her utility and satisfy him/her. The user ranks available resources based on his/her criteria. Improving user satisfaction, our approach can handle offloading processes in a decentralized manner with heterogeneous users. The rest of this paper is organized as follows: There is a brief review of related works in Section 2. Section 3 describes the task offloading problem. Section 4 models the problem as a pairing process and explains the mapping of the pairing process to a pairing game. The results of the conducted experiments are reported in Section 5. Section 6 concludes the paper.

2. RELATED WORKS

Task and computation offloading is a significant feature for task performance and resource use optimization in edge computing, which has become one of the preferred methods to increase the

1. H. Morshedlou is with Department of Computer Engineering and Information Technology, Shahrood University of Technology, Shahrood, Iran. Email: morshedlou@shahroodut.ac.ir
2. R.V. Shoar is is with Department of Computer Engineering, AmirKabir University of Technology, Tehran, Iran. Email: vafashoar@aut.ac.ir

performance of user tasks for smart devices. Computation offloading has been extensively studied in order to shorten the execution time of mobile devices and energy saving [1]-[2]. [3] investigated the problem of multi-user computation offloading in a single-channel wireless environment, where each user must decide whether or not to offload. [4] looked into an energy-harvesting mobile edge computing system. It suggests a Lyapunov-based dynamic computation offloading technique that jointly determines the offloading decision and the transmit power for computation offloading. [5] suggested offloading computational tasks to the mobile edge. The collective pool of resources for mobile devices is referred to by the authors as mist computing or cloudlets. They don't go into detail about how resource management might be carried out. Nevertheless, they place a strong emphasis on a hierarchical offloading of tasks from mobile nodes to mist, from mist to fog and finally from fog to cloud, all depending on application requirements. A Stackelberg game is used to model the interaction between cloud nodes and edge servers with the goal of increasing the benefits of cloud services [6]. Their analysis yields a unique Nash point. Two primary issues are addressed by their suggested strategy. First, a cloud server must decide whether to accept or deny a request to offload data to an edge server when it is made. The second issue is what to offer the edge node in the way of an incentive. [7] studied the issue of workload assignment and VM placement for mobile edge-computing applications. The paper develops a mathematical model to reduce the amount of hardware required by virtual machines (VMs) to support specified workloads in a multi-application scenario while satisfying the latency needs of various applications. According to the results, the users' request load, the hardware capacity of MEC servers and the latency requirements of applications all have a substantial impact on the amount of hardware used overall. MEC server utilization can be improved by utilizing remote VM deployment and workload aggregation. A novel approach is put out in [8] to address the issues of energy consumption and VM placement. The suggested approach can cut down on both energy usage and placement time. It proposes an OEMBA algorithm which integrates idle servers and reduces energy consumption. Using an improved Long Short-Term Memory model, virtual machine placement is then accelerated and latency is reduced based on historical data. According to the reported findings, the improved learning model can reduce placement latency and save energy. [9] suggested a method that allows for communication channel selection and virtual machine placement. It takes advantage of user movement prediction to manage the system's dynamic nature. According to anticipated user-movement, the prediction is used for dynamic VM placement and to identify the best communication path. The authors assert that their method reduces work offloading delays by 10% to 66% while maintaining a constant level of energy consumption by user equipment. In [10], the distributed task-offloading optimization issue is examined. First, a fresh optimization scheme is suggested that seeks to maximize the anticipated offloading rate of multiple agents by enhancing their offloading limits. Then, the issue is formulated in terms of game theory, which ultimately results in the development of a distributed best-response (DBR) iterative optimization system. Because of the frequently limited resources supplied by edge servers, not all IoV users' requirements can be met at once. A task offloading strategy built on fuzzy neural network (FNN) and game theory was developed in [11] to address the aforementioned issue. Following the load balancing of each RSU, game theory is used to determine the best task-offloading strategy for the users. Following this, the edge server acts as an agent to allocate computing resources for the offloaded tasks by the Q-learning algorithm. A task-offloading scheduling approach that combines multi-agent reinforcement learning and meta-learning was suggested by [12]. A first-order approximation technique is suggested to effectively train the policy network. Experiment results show that the MRL-based strategy has an outstanding overall performance and can be quickly applied in a variety of environments with good stability and generalization. [13] suggested a multi-agent deep reinforcement learning-based priority-driven joint task-offloading method where each edge server's decision-making is influenced by both its own state and shared global information. The work continues to create a worldwide state-sharing model that greatly lowers transmission overhead between peripheral servers. The performance analysis done on a dataset of mobile device movement trajectories shows that the suggested method can shorten job-completion times and increase the efficiency of edge computing resources. In [14], an approach, called NFSP, was proposed for adopting the architecture to offload decision-making. NFSP explicitly tackles the non-stationarity issue with the built-in self-play mechanism and uses a mixed strategy consisting of deep RL and the past average strategy, which is approximated by supervised deep learning. The conducted simulation experiments show that the proposed method outperforms the raw Independent RL (IRL) approaches. IRL approaches emerge as promising solutions for scenarios in which, due to

privacy and security concerns, mobile devices may be unwilling to expose their local information.

3. TASK OFFLOADING PROBLEM

Assume that there are limited resources at the edge network that offer computing or other services to users. For reasons of resource limitation, all tasks cannot be provided locally at the edge and some of them should be transferred to the cloud. One simple approach is to consider the deadlines of the tasks when making a decision about transferring some of them to the cloud. However, involving users or their agents in decision-making is a better approach for reaching an acceptable solution for all users and decreasing their dissatisfaction.

3.1 The Proposed Approach

There are two different types of agents in the approach presented in this paper. After initiation, agents interact with each other in the EOM module. First, we introduce these modules and entities in detail and then, we present the proposed approach and related analyses. Figure 1 depicts the components of the edge environment in the proposed approach. "DEVICES" are user devices that offload tasks to the edge. The edge layer contains some nodes or resources that provide the computing needs for user tasks. As illustrated in this figure, there are two sets of agents. eAgents interact as the brokers of edge nodes, with tAgents as the brokers of user devices. In our approach, EOM is only used to facilitate interactions among tAgents and eAgents; it plays no key role in decision-making and can be replaced by a decentralized platform for communications and interactions. Details of interactions are described in the following sections in details. In the cloud layer, there are data centers that, when the resources of the edge layer are not enough to provide all offloaded tasks, some tasks will be sent to these centers in the cloud.

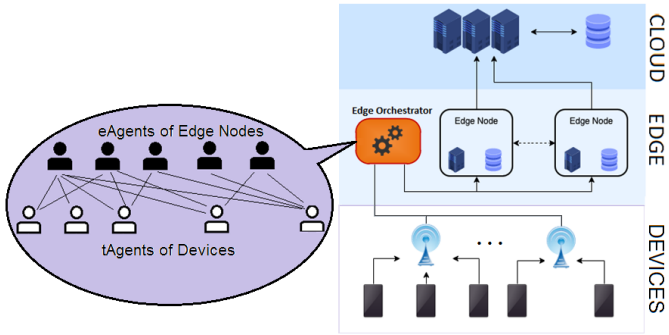


Figure 1. Components of edge environment in the proposed approach.

Edge Orchestrator Module (EOM): The edge orchestrator module makes decisions in the system. To determine how and where to handle incoming client requests, it consults the data gathered from the other modules.

tAgent: A tAgent is a broker or representative of a user task that tries to find a proper resource in edge or cloud for running its task. EOM initiates a tAgent when a new task is received in iteration t.

eAgent: An eAgent is a broker of EOM. According to the current number of tAgents and available resources in the system during iteration t, EOM generates eAgents. Each eAgent gets information about current tasks from all or some tAgents and gives them information about available capacity. Then, the pairing process of eAgents and tAgents in iteration t is initiated in the EOM. The pairing process is described in detail in the following section. We model the pairing process by using two games and show that the result of pairing is an NE point of them. Reaching an NE point means that the obtained result by our approach satisfies every rational agent or broker (resource broker (eAgent) or user broker (tAgent)).

4. PAIRING PROCESS

As said before, there are two sets of brokers, called task brokers (tAgents) and edge brokers (eAgents). By receiving an offloaded task, a new tAgent is initiated by the EOM and assigned to the task. tAgent

tries to maximize its user's utility. At the other side, there are eAgents which are policy-aware brokers or agents of edge resources. A tAgent offers its requirements (which include specific task needs) to all eAgents and waits to receive their proposals. When an eAgent receives offloading requests from tAgents, it queries the EOM for information, such as current workload and network congestion, among other things. Based on the information received from the EOM and the eAgent's resource management policy, it calculates the task-completion quality (TCQ) and announces the calculated value to the tAgent. Algorithm 1 shows how to calculate the TCQ for each tAgent. Notice that even when tAgent1 has a closer deadline than that of tAgent2, an eAgent may calculate a lower TCQ for tAgent1 than for tAgent2. However, when paired, it chooses tAgent1 due to its closer deadline. It announces a lower TCQ to tAgent1 to convince tAgent1 that it is better to choose another eAgent. A tAgent prefers to reach agreement with an eAgent the TCQ of which has the highest value among available eAgents. Between two tAgents, an eAgent chooses a tAgent with a closer task deadline. It's assumed that each eAgent contracts with up to one tAgent and *vice versa*. Since the capacity of each edge resource is different, the EOM may initiate multiple eAgents for each edge resource. Therefore, an edge resource may accept multiple tasks in iteration t. Each tAgent has an ordered list of eAgents based on their proposed TCQ. Similarly, each eAgent has an ordered list of tAgents based on their deadline. While the ordered lists of tAgents may be different, eAgents have similar ordered lists. These ordered lists are referred to as preferred lists.

Calculating TCQ: We use a meta-heuristic optimization algorithm for the 0-1 Knapsack problem to obtain an acceptable solution and then, we use the solution to calculate TCQ values. The meta-heuristic approach, that is used here, is simulated annealing. Note that finding an optimal solution for 0/1 Knapsack is possible, but for a large number of objects, time complexity makes it hard to use non-heuristic algorithms for finding optimal solutions. A task is considered as object and resource capacity is considered as Knapsack capacity. The details of the Simulated Annealing implementation are described as shown below:

Representation: A 0/1 representation is used, where 1 indicates that the task is chosen for running on the resource (object is placed in the Knapsack), while a value of 0 means that the task is not chosen (the object is left behind).

State Generation: State generation in our SA approach is done by randomly choosing a bit from the current state and flipping it to create the next state. This procedure is shown in Figure 2 for four objects.

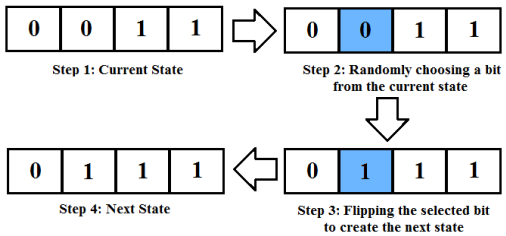


Figure 2. Simulated annealing state generation.

Pseudocode: The pseudocode of Figure 3 illustrates the heuristic that Simulated Annealing uses to search the solution space for an optimal solution for 0/1 Knapsack. The best_state contains the best state found up to the t-th step. In line 6, the function "value" calculates the value of the input state based on the objects in the Knapsack. Making the transition from the current_state to new_state is specified by the value(next_state) and value(current_state) (see line 7). States with a greater value are better than those with a smaller value. Even if value(next_state) is less than value(current_state), the probability of the transition must be positive. This feature prevents the method from becoming stuck at a local minimum that is worse than the global one (see lines 13 and 14). Transition is also affected by a global time-varying parameter Tt known as temperature. When Tt tends to zero (see line 18), the probability acceptance_function must also tend to zero (see line 13). For sufficiently small values of Tt, the system will then increasingly favor moves that go "downhill" and avoid those that go "uphill." With Tt = 0, the procedure reduces to the greedy algorithm, which makes only the downhill transitions. The system reaches equilibrium at Tt when we have no more state changes (see line 17).

Termination Criteria: When the temperature hits the system's lowest allowable temperature, the algorithm stops, signalling that the system is frozen. In our implementation, the temperature starts at 1000 and gradually drops to 0.0001 before freezing. To adjust the temperature, multiply the system temperature by = 0.9999.

After finding an acceptable solution, the EOM maps the solution to an array and first sorts them based on 0/1 and then sorts the selected and unselected objects separately based on task deadlines. After the sorting phase, objects are assigned TCQ values. The first selected object with the closet deadline has the maximum TCQ. Figure 4 illustrates the TCQ assignment process. Notice that the selection of tasks is done independently of their deadlines.

```
1   Set current_state = initial_state  and  best_state = initial_state;
2   t = 1;
3   do {
4           do {
5                   Produce next_state;
6                   delta = value (next_state)  -  value(current_state);
7                   if(delta > 0 && solution is acceptable) {
8                           current_state = next_state;
9                           if ( value(current_state) > value(best_state) )
10                                  best_state = current_state;
11                  }
12                  else {
13                          acceptance_function = exp(-delta/Tt);
14                          if(acceptance_function > random[0,1))
15                                  current_state = next_state;
16                  }
17          } While (system equilbrium a Tt);
18          Tt+1 = Tt*alpha;
19  } While (system has been frozen);
20  return best_state;
```

Figure 3. Pseudocode of algorithm 1 for SA implementation.

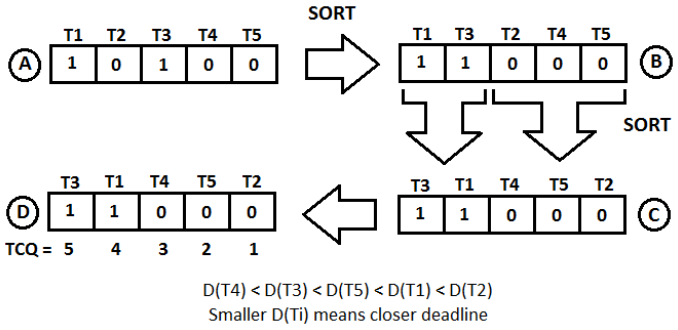


Figure 4. TCQ assignment process.

Definition 1: Preferred series of agents are loop-less if and only if there is no wrap-around order of agents $b_1, b_2, ..., b_k$ (k is even and $k > 2$) where each agent b_i likes b_{i+1} more than b_{i-1} (if $i = k$ then replace i+1 with 1). Keep in mind that in $b_1, b_2, ..., b_k$ for each i, b_{i-1} and b_{i+1} are both either tAgents or eAgents (have similar type) and are different from b_i.

Lemma1. The generated preferred series are loop-less if eAgents sort their preferred series based on the deadline of the tAgents, independent of the tAgents' preferred series.

Proof. There is a wrap-around order of agents, $b_1, b_2, ..., b_k$, such that each agent b_i prefers b_{i+1} to b_{i-1}, if the preferred series are not loop-less. Assume b_i to be an eAgent. Because eAgents organize their

preferred series according to deadlines of tAgents, they will obtain similar preferred series. This means that if eAgent b_i prefers tAgent b_{i-1} to tAgent b_{i+1}, all other eAgents do so. Because the order is wrap-around and preferring is transitive, after first loop over the order we conclude that eAgent b_i prefers tAgent b_{i+1} to tAgent b_{i-1}. This outcome conflicts with the definition of the order. Consequently, it is impossible to have such a chain of brokers to exist and the preferred series are loop-less.

Definition 2: A set of tAgents and eAgents is Pair-able if we can identify two agents (one tAgent and one eAgent) that favor each other above all other agents of the other type already present in the set at each cycle of an iterative process. The set for the following cycles or iterations no longer includes these two agents. After the last iteration, the set is either empty or contains agents of similar type.

Theorem 1. Set of tAgents and eAgents is Pair-able.

Proof. Begin with arbitrary tAgent such as b_1 from agent set and form a sequence $b_1, b_2, ...$ in which b_{i+1} is the most preferred agent by agent b_i among existing agents of the set which have the opposite type. Because there are only a certain number of agents, the sequence has a loop or cycle. Using lemma 1, the acquired preferred series are loop-less and hence the loop or cycle length is 2. Two agents that favor each other the most are present in this loop. A similar process will be carried out with fewer agents at the following iteration by removing these two agents from the group of brokers. This process is repeated until the final iteration.

Theorem 1 proves that sets of agents are pairable. The pairing approach in Figure 5 illustrates an algorithm for this pairing. Because each agent acts as a user's or edge resource agent, it should try to maximize its utility. We now want to be certain that the pairing specified in Theorem 1 satisfies both rational users and edge resources. For this purpose, in the following, we provide a description of this pairing from a game-theoretic perspective. Two games, an eGame and a tGame, are used to describe the situation. Let $r_i(j)$ represent the position of eAgent/tAgent j's (rank) in preferred serie of tAgent/eAgent i. The player i's strategy is indicated by s_i and S is the collection of all available strategy profiles for players.

4.1 tGame

tGame players are tAgents and eAgents which are part of the environment. Because agents of both types are rational, they select the strategy with maximum utility. The set of actions in tGame represent the tAgents' strategy space. Every action is the same as selecting a particular eAgent. For every strategy profile $s \in S$, utility of tAgent i is $u_i(s) = n - r_i(j) + 1$ if and only if $s_i = j$ (selection of eAgent j) and there is not another tAgent k $(k \neq i)$ such that $s_k = j$ and $r_j(k) < r_j(i)$, otherwise $u_i(s) = 0$.

4.2 eGame

Similar to tGame's definition, but this time, eAgents are the game players.

4.3 Game Theoretic Analysis

Theorem 2. A Pure Nash Equilibrium point of tGame and eGame is the result of Algorithm 1.

Proof. Assume that the current strategy profile of players in tGame and eGame is an equivalent point to the outcome of algorithm 1. All the eAgents have same preferred list and hence they have similar ranking of tAgents. Let uB_i denote a tAgent which is in i-th place of this ranking and iB_i is the paired eAgent to uB_i in the pairing procedure of algorithm 1.

In tGame, if tB_i deviates unilaterally and selects eB_k $(k \neq i)$, then for the case $r_{tB_i}(eB_k) > r_{tB_i}(eB_i)$, this deviation is not profitable, but if $r_{tB_i}(eB_k) < r_{tB_i}(eB_i)$, then eB_k has paired with a tB_k such that tB_k has a closer deadline than tB_i and every eAgent prefers tB_k to tB_i. So, eB_k refuses tB_i and

according to the definition of tGame $u_{tB_i}(s_{tB_i} = eB_k) = 0$. Therefore, for the case of $r_{tB_i}(eB_k) < r_{tB_i}(eB_i)$ deviation is not profitable as well. This means that the outcome of Algorithm 1 is a pure Nash Equilibrium of tGame.

For eGame, All eAgents prefer tAgent tB_i to tB_{i+1}. This means deviation of eB_1 is not profitable. Deviation of eB_2 can be profitable, if tB_1 prefers eB_2 to eB_1. But this is not true, because based on algorithm 1, tB_1 and eB_1 are two brokers that prefer each other the most. So, deviation of eB_2 is not profitable as well. It can be concluded in a similar way that for the rest of eAgents deviation is not profitable and the outcome of algorithm 1 will be a pure Nash Equilibrium of eGame as well.

Theorem 3. Both eGame and tGame have unique a Pure Nash Equilibrium (PNE) point.

Proof. Assume the PNE of eGame and tGame is not unique and there is another Nash Equilibrium $\underline{NE}$. Let $\overline{NE}$ denote the PNE which is equivalent to the outcome of algorithm 1 and definitions of tB_i and eB_i are like to Theorem 2. $s_B(N)$ denotes the strategy of broker B in a strategy profile N of game.

For tGame, if $s_{tB_1}(\underline{NE}) \neq eB_1$, then deviation of tB_1 to eB_1 will be profitable and $\underline{NE}$ is not a Nash. For the case of $s_{tB_1}(\underline{NE}) = eB_1$, we have $s_{tB_1}(\underline{NE}) = s_{tB_1}(\overline{NE})$. It can be shown that for all tAgents, the condition of $s_{tB_i}(\underline{NE}) = s_{tB_i}(\overline{NE})$ is required for $\underline{NE}$ to be a Nash point. So, $\overline{NE}$ is unique. Proof for eGame can be done in a similar way.

1- Put eAgents and tAgents in two different lists, called eList and tList.

2- Sort tList based on deadline of tAgents such that the first tAgent in the list is the tAgent with closest deadline.

3- While (min(eList.size, tList.size) > 0) {
3-1- Pick an eAgent in eList such that it is most preferred by the first tAgent in tList above all other agents currently in eList.
3-2- Pair the two eAgent and tAgent of step (3-1) and eliminate them from eList and tList.
}

Figure 5. Pairing algorithm of eAgents and tAgents.

Based on Theorems 2 and 3, the result of algorithm 1 corresponds to the unique pure NE point of eGame and tGame. Following the best response strategy, when the game has a unique pure NE point, this leads to convergence at that unique pure NE point [15]. This means that algorithm 1 produces a unique pure NE that satisfies rational tAgents and eAgents.

5. EXPERIMENTS AND RESULTS

In this section, we present the results of the conducted experiments. First, we introduce two criteria that are used for evaluation of the proposed approach. These criteria are User Satisfaction Level (USL) and Edge Utility Level (EUL).

5.1 User Satisfaction Level

We define User Satisfaction Level (USL) based on the average utility of users, as shown in Equation (1).

$$USL(t) = \sum_{i \in Users} u_i(r_i(t)) / number\_of\_users \quad (1)$$

At iteration t, the utility function u(r(t)) measures the utility that a user attaches to the allocated resource $r(t)$. Note that the service provider is not aware of the utility functions of a user.

If user i or its tAgent prefers resource A to resource B, then $u_i(A(t)) > u_i(B(t))$. For example, we can define the utility function as Equation 2, where $VMC_j(CPU)$ is CPU capacity of VM_j. $MinNeed_i(CPU)$ and $MaxNeed_i(CPU)$ are the minimum and maximum requirements of task i for the CPU, respectively. The symbols $VMC_j(Memory)$, $MinNeed_i(Memory)$ and $MaxNeed_i(Memory)$ have the same meaning but for memory. w_{cpu} and w_{memory} are weights that illustrate importance of CPU and Memory capacity for user task. β_{CPU} and β_{Memory} show the importance of wasting resources for users.

$$u_i(VM_j(t)) = w_{cpu} \times \begin{bmatrix} VMC_j(CPU) - MinNeed_i(CPU) \\ -\beta_{CPU} \times \max(0, VMC_j(CPU) - MaxNeed_i(CPU)) \end{bmatrix} + w_{memory} \times \begin{bmatrix} VMC_j(Memory) - MinNeed_i(Memory) \\ -\beta_{Memory} \times \max(0, VMC_j(Memory) - MaxNeed_i(Memory)) \end{bmatrix} \quad (2)$$

5.2 Edge Utility Level

The utility of edge is defined based on the used capacity and rate of successfully completed tasks for all received tasks in iteration t.

$$EUL(t) = \frac{used\_resources\_capacity(t)}{available\_resource\_capacity(t)} \times \frac{number\_of\_successfully\_completed\_tasks(t)}{number\_of\_all\_received\_tasks(t)} \quad (3)$$

5.3 Experiments

This sub-section illustrates the efficiency of the proposed approach using USL and EUL. In the conducted experiments, the resource needs of each category of tasks are expressed based on their minimum and maximum requirements. Here, the CPU needs of a task are illustrated using the [C, D] range. C is the minimum CPU need and D is the maximum CPU capacity consumed by the task. If there are two resources with different CPU capacities greater than C, both are capable of running and completing the task before its deadline. However, the task broker prefers to be paired with a resource that has a higher CPU capacity. Memory needs are also illustrated using ranges. For example, a task with a low memory requirement and the availability of extra memory results in faster running. Task brokers score the resources based on their available CPU and memory capacities.

There are three types of virtual machines (VMs) and their capacity details are listed in Table 1. The <cpu, memory> vector represents the available capacity of a VM and the user task requirement. The vectors are unitized for simplicity. Table 2 illustrates CPU and memory requirements for different task types. According to the table information, any task of any type can be executed on any type of VM, though the results may differ in terms of execution speed. For example, a task of type 1 prefers VM of type 3 to the other types, while a task of type 2 prefers VM of type 2 to VM of type 3. This is due to the fact that a task of type 2 cannot consume more CPU and memory capacities of VM type 3. Its maximum requirements are <3, 5>, which can be met by a weaker VM such as a VM of type 2.

Table 1. CPU and memory capacity of VMs.

Resource Types	VM Type 1	VM Type 2	VM Type 3
CPU Capacity	4	5	6
Memory Capacity	4	5	6

Table 2. CPU and memory needs of different task types.

Task Requirements	Task Type 1	Task Type 2	Task Type 3	Task Type 4	Task Type 5
CPU Need Range	[1~2]	[1~3]	[2~4]	[3~5]	[4~6]
Memory Need Range	[4~6]	[3~5]	[2~4]	[1~3]	[1~2]

5.3.1 Workloads of the Experiments

Following a task-type-based distribution, each user device generates a workload with a specific task type. The times at which tasks are generated are defined by a Poisson process. Each task type has a certain expected value. The time interval between two tasks of a device is generated at random by an exponential distribution with this expected value. The number of edge servers and user devices is 10 and 200, respectively. The Poisson parameter (λ) for all task types is 1. Each edge server's CPU and memory capacity are assumed to be 100. Therefore, an edge server can host 25 VMs of type 1, 20 VMs of type 2 or 16 VMs of type 3. An edge server usually hosts multiple VMs of different types. To evaluate the proposed approach against the existing works in the literature, we selected two rival approaches to compare, called RW1 and RW2 in this paper. RW1 and RW2 are described in [14] and [8], respectively. For details on these approaches, please see the related work section. There are 200 user devices in the experiments that generate workloads. Workload1 is generated by 40 user devices for each task type. Workloads 2 and 3 are generated by (80, 60, 40, 20, 0) and (0, 20, 40, 60, 80) users, respectively. The tuples show the number of users per each task type according to (TaskType1, TaskType2, TaskType3, TaskType4, TaskType5) tuple.

Figure 6 shows the utility of a user with a specific task type on each type of VM. The utility function of a user is defined using (2), where $\beta_{CPU} = 1.5$, $\beta_{Memory} = 0.8$ and $w_{CPU} = w_{Memory} = 0.5$. Here, $\beta_{CPU} > \beta_{Memory}$ means that preventing CPU wastage is more important than preventing memory wastage. As illustrated by this figure, tasks of type 1 or 5 prefer VMs of type 3 to the other types. In addition, VM of type 2 is preferred by tasks of type 4 or 2 and tasks of type 3 prefer VM of type 1. These utility values will be used to calculate the USL of users in the following. Figure 7 illustrates the average of USL over 1000 consecutive iterations by the proposed approach, RW1 and RW2, during workload 1. The workload is generated according to the approach described at the beginning of this section. USL in iteration t is calculated using (1). The higher values of USL mean a higher average utility of the users and, as a result, a higher satisfaction level. Figures 8 and 9 show similar diagrams for workload2 and workload3, respectively. The proposed approach leads to higher USL in comparison to rival approaches. The proposed approach achieves higher USL, because it takes user preferences into account during the pairing process, whereas competing approaches do not. Since a user ranks the available resources according to the amount of utility it gets from each of them, incorporating the user's preference during the pairing process will increase the level of USL. During workload 2 (Figure 8), tasks have less CPU requirement than Workloads 1 and 3. Because $\beta_{CPU} > \beta_{Memory}$, the difference in USL levels in different approaches during workload 2 is smaller than in workloads 1 and 3. This means that the proposed approach has a better performance when there are high CPU requirements in workloads.

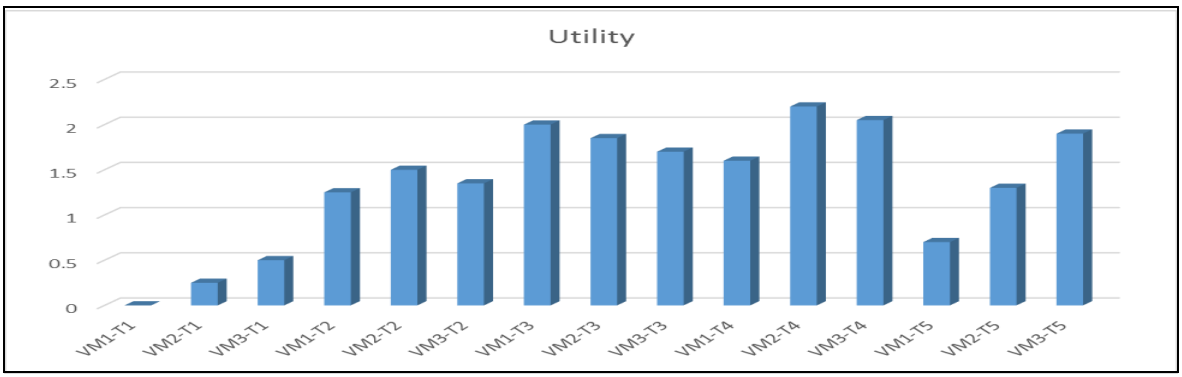


Figure 6. Utility of different VMs for each task type.

According to the diagrams in Figure 10, it is clear that when the majority of users have high CPU needs (e.g. Workload3), CPU wastage is very low. Furthermore, when the majority of users have high memory requirements (e.g. Workload2), CPU wastage is high. This is due to the fact that there is a positive correlation between the processing power and the amount of available memory at each virtual

machine (see Table 1). In other words, VMs with more processing power have more available memory. Since $\beta_{CPU} > \beta_{Memory}$ in (2), we have a smaller USL during workload 2 compared to both workload 1 and workload 3 (see Figures 7, 8 and 9). During workload 2, tasks of type 1, which waste more CPU compared to the other types, constitute 40% of all the tasks. To be more specific, the majority of the tasks in workload 2 are of types 1 and 2, which resulted in more CPU wastage than in workloads 1 and 3. According to the results, the proposed approach always performs better than similar works in terms of USL. In addition to this advantage, the proposed approach can perform better under different workloads in terms of CPU usage rate.

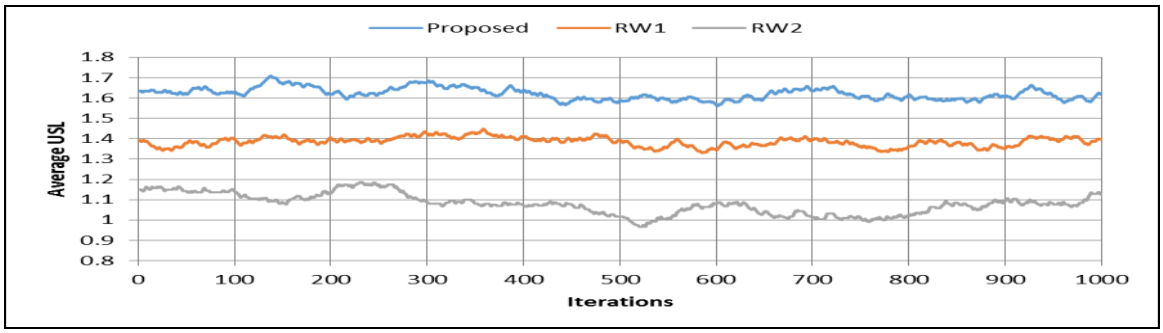


Figure 7. Average USL of users during workload 1.

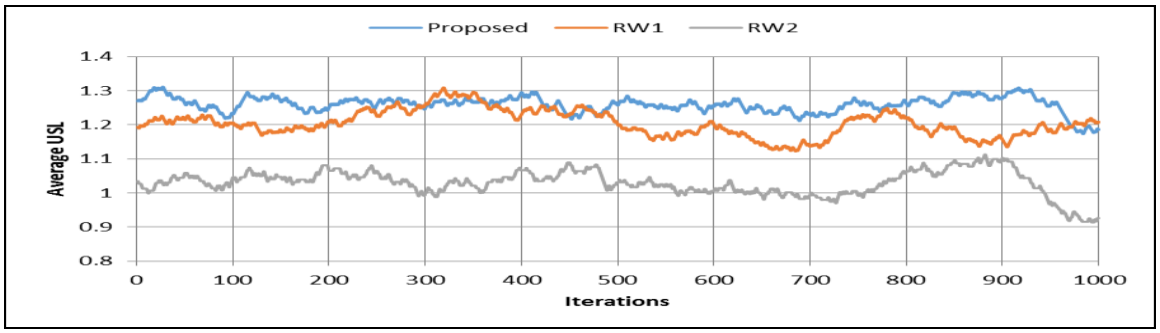


Figure 8. Average USL of users during workload 2.

Now, we compare the results of the proposed approach with rival methods using EUL measure. We use EUL as a criterion to measure resource capacity usage and task-completion rates. Here, each iteration is a time interval in which task-completion rate and average resource usage are used to compute EUL of that iteration using (3). Task-completion rate is reported by dividing the number of successfully completed tasks by the total number of all received tasks in the mentioned time interval. As it can be seen in Figure 11, the proposed approach has achieved better results in most iterations. In the proposed method, the allocation of extra, unutilized capacity to tasks is avoided. For example, when a task of type 4 prefers a VM of type 2 to a VM of type 3, the proposed method instantiates a VM of type 2. Therefore, the capacity of a physical server machine can be allocated to more virtual machines and more tasks have a chance to be completed successfully. This is why the proposed approach performs better compared to the other approaches.

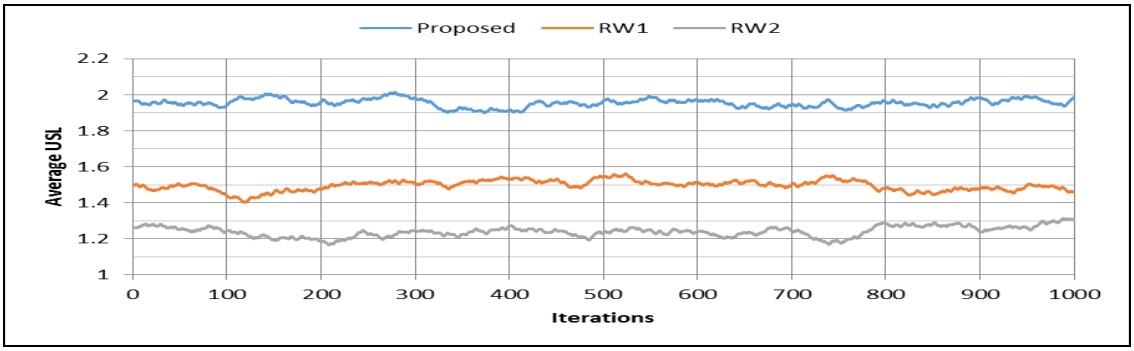


Figure 9. Average USL of users during workload 3.

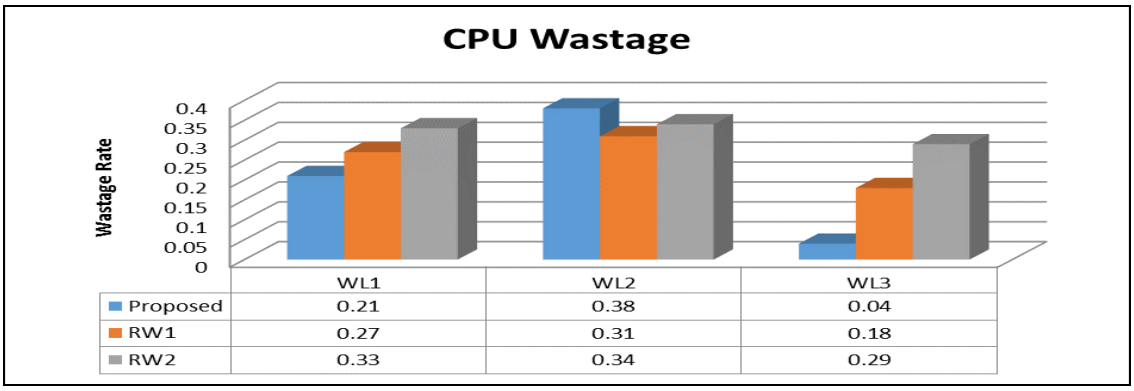


Figure 10. CPU wastage during workloads 1, 2 and 3.

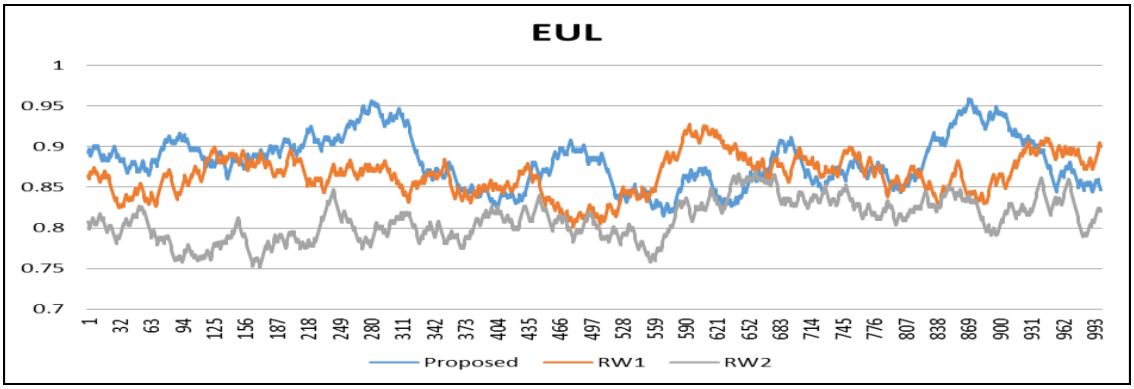


Figure 11. EUL measure using different approaches.

6. CONCLUSIONS

Due to the limited capacity, edge nodes may not be able to provide the total capacity necessary for delivering all user tasks offloaded to the edge. Therefore, it remains a challenge to reduce user-task handling delays when there are insufficient resources. One solution to avoid further delays is offloading some tasks to cloud data centers. But, finding an efficient approach for decision-making about which tasks should be handled by edge servers and which ones by the cloud is not easy. In addition to the complexity of interactions and relations in edge environments, some nodes or devices may be unwilling to expose their local information due to privacy and security concerns. This makes centralized control and decision-making very difficult and inefficient. Distributed and decentralized approaches, like agent-based approaches, emerge as promising solutions for such scenarios. In this paper, an agent-based approach for task offloading in edge computing environments is proposed. The proposed approach incorporates user satisfaction into decentralized decision-making. To the best of the authors' knowledge, this is the first work in the literature that addresses task-offloading strategy optimization based on user satisfaction level. In our work, user satisfaction level is defined based on the average utility that users attach to the CPU and the memory capacity of the allocated edge resources. But, the utility functions of different users may depend on different criteria. Each user knows better what will make him/her satisfied. As a future work, the definition of utility function can be extended to include complex criteria, such as energy efficiency. For example, dynamic frequency scaling (also known as CPU throttling) is a power management technique in today's computer architectures, whereby the frequency of a microprocessor can be automatically adjusted "on the fly" depending on the actual needs, to conserve power and reduce the amount of heat generated by the chip. In our approach, each user device, according to its condition, e.g. battery level, can attach a utility to the local running of a task (high frequency and energy consumption) or offloading to edge (low frequency and energy consumption). Regardless of what improves a user's satisfaction level, our approach can be extended to handle offloading processes in a fully decentralized way with users with heterogeneous satisfaction criteria. Another contribution of our work is representing the task-offloading problem in the form of two decentralized games. We proved that the proposed pairing process reaches the Nash equilibrium point of the games. It is also demonstrated that achieving Nash equilibrium is the best possible pairing for both tasks and VMs. A game-theoretic analysis is also

provided to prove that the presented approach increases the average utility of users and their satisfaction level. We also conducted some experiments to evaluate the proposed approach. The results illustrate that the proposed approach significantly improves the average utility of users and the EUL measure.

REFERENCES

[1] C. Feng et al., "Computation Offloading in Mobile Edge Computing Networks: A Survey," J. of Network and Computer Applications, vol. 202, no. 103366, p. 103366, Jun. 2022.
[2] Y.-Y. Huang and P.-C. Wang, "Computation Offloading and User-clustering Game in Multi-channel Cellular Networks for Mobile Edge Computing," Sensors (Basel), vol. 23, no. 3, p. 1155, Jan. 2023.
[3] X. Chen, "Decentralized Computation Offloading Game for Mobile Cloud Computing," IEEE Transactions on Parallel and Distributed Systems, vol. 26, no. 4, pp. 974–983, Apr. 2015.
[4] Y. Mao et al., "Dynamic Computation Offloading for Mobile-edge Computing with Energy Harvesting Devices," IEEE J. on Selected Areas in Communications, vol. 34, no. 12, pp. 3590–3605, Dec. 2016.
[5] X. Chen, L. Jiao, W. Li and X. Fu, "Efficient Multi-user Computation Offloading for Mobile-edge Cloud Computing," IEEE/ACM Transactions on Networking, vol. 24, no. 5, pp. 2795–2808, Oct. 2016.
[6] Y. Liu et al., "Incentive Mechanism for Computation Offloading Using Edge Computing: A Stackelberg Game Approach," Computer Networks, vol. 129, pp. 399–409, Dec. 2017.
[7] W. Wang, Y. Zhao, M. Tornatore, A. Gupta, J. Zhang and B. Mukherjee, "Virtual Machine Placement and Workload Assignment for Mobile Edge Computing," Proc. of the 2017 IEEE 6th Int. Conf. on Cloud Networking (CloudNet), DOI; 10.1109/CloudNet.2017.8071527, Prague, Czech Republic, 2017.
[8] C. Jian, L. Bao and M. Zhang, "A High-efficiency Learning Model for Virtual Machine Placement in Mobile Edge Computing," Cluster Computing, vol. 25, no. 5, pp. 3051–3066, Oct. 2022.
[9] J. Plachy, Z. Becvar and E. C. Strinati, "Dynamic Resource Allocation Exploiting Mobility Prediction in Mobile Edge Computing," Proc. of the 2016 IEEE 27th Annual Int. Symp. on Personal, Indoor and Mobile Radio Communications (PIMRC), DOI: 10.1109/PIMRC.2016.7794955, Valencia, Spain, 2016.
[10] J. Zhou et al., "Distributed Task Offloading Optimization with Queueing Dynamics in Multiagent Mobile-edge Computing Networks," IEEE Internet of Things J., vol. 8, no. 15, pp. 12311–12328, 2021.
[11] X. Xu et al., "Game Theory for Distributed IoV Task Offloading with Fuzzy Neural Network in Edge Computing," IEEE Transactions on Fuzzy Systems, vol. 30, no. 11, pp. 4593–4604, Nov. 2022.
[12] W. Ding et al., "A Multi-agent Meta-based Task Offloading Strategy for Mobile Edge Computing," IEEE Trans. on Cognitive and Developmental Systems, DOI: 10.1109/TCDS.2023.3246107, 2023.
[13] J. Yang, Q. Yuan, S. Chen, H. He, X. Jiang and X. Tan, "Cooperative Task Offloading for Mobile Edge Computing Based on Multi-agent Deep Reinforcement Learning," IEEE Transactions on Network and Service Management, DOI: 10.1109/TNSM.2023.3240415, 2023.
[14] J. Hou et al., "GP-NFSP: Decentralized Task Offloading for Mobile Edge Computing with Independent Reinforcement Learning," Future Generation Computer Systems, vol. 141, pp. 205–217, Apr. 2023.
[15] J. Lei and U. V. Shanbhag, "Stochastic Nash Equilibrium Problems: Models, Analysis and Algorithms," IEEE Control Systems, vol. 42, no. 4, pp. 103–124, Aug. 2022.

ملخص البحث:

نظراً لمحدودية الموارد في شبكة الحافّة وإحالة عدد كبير من المهامّ الى خوادم الحافّة، فإنّ موارد الحافّة قد لا يكون بمقْدورها أن توفرّ السّعة المطلوبة لتنفيذ جميع المهامّ. ونتيجةً لذلك، تتعيّن إحالة بعض المهامّ الى السّحابة، الأمر الذي من شأنه أن يتسبّب في تأخيراتٍ إضافية. وهذا بدوره قد يقود الى زيادة عدم الرّضا لدى المستخدمين الّذين تم نقْل مهامّ تتعلّق بهم الى السّحابة. في هذه الورقة، يجري اقتراح نظام جديد من أجل اتخاذ القرار حول أيّ المهامّ يتمّ نقلها الى السّحابة وأيها يتمّ تنفيذها محلّياً. ويحاول النّظام المقترح تكوين أزواجٍ من المهامّ والموارد، بحيث تكون هذه الأزواج من المهامّ والموارد هي أكثر الموارد تفضيلاً بالنّسبة للمستخدِمين من بين جميع الموارد المتاحة. وقد بيّنّا أنّ الوصول الى نقطة "ناش" للتّوازن يُمكن أن يلبّي هذا الشّرط. وبيّن تحليلٌ باستخدام نظرية اللُّعبة أنّ النّظام المقترح يزيد من معدّل الاستفادة لدى المستخدِمين ويرفع من مستوى الرّضا لديهم.

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 09, No. 02, June 2023.

LOW-COST C-BAND SIW BANDPASS FILTER USING FR4-EPOXY SUBSTRATE

Abed Ahcéne[1], Bouchekhlal Ahmed[2], Amrouche Aissa[3] and Bendoumia Rédha[1]

(Received: 25-Jan.-2023, Revised: 30-Mar.-2023, Accepted: 17-Apr.-2023)

ABSTRACT

This paper describes a substrate-integrated waveguide (SIW) filter built on an Fr4-Epoxy substrate with a dielectric constant εr = 4.4 and a height of h = 1.6mm. SIW-based devices have piqued the interest of researchers in recent years due to their low loss, small size and low cost. The goal is to simulate and realize an SIW filter for C-band applications. The designed filter is analyzed using the reflection coefficient S11 and electric field distributions. We used the HFSS simulator. According to the findings, there is a high degree of agreement between the simulated and realized filters. The results also show that the filter has a very good response. It also displays bandwidth around C-band frequencies (6.3GHz). This filter's pass-band ranges from 5.39 to 6.83GHz, with an insertion loss of 4.2dB and a return loss of 45.49dB.

KEYWORDS

SIW, Bandpass filter, C-band, Fr4-Epoxy.

1. INTRODUCTION

High-performance RF/microwave filters with critical properties such as weight, cost, insertion loss, quality factor and power handling capacity, are required for modern communication systems for satellite and mobile applications and meeting all of these requirements can be difficult. Rectangular Waveguides (RWs) are common microwave components, but they are difficult to manufacture and incorporate into low-cost planar structures [1]. Scientists have chosen to create a planar structure to solve this problem.

Over the past decade, substrate-integrated waveguides (SIWs) have grown in popularity, being a new technology representing a very promising solution technology which has been the focus of intense and ever-increasing research efforts [2]. Complete shielding, comparatively low losses and straightforward, affordable manufacturing are all combined by SIW technology, making it the ideal platform for developing the next generation of wireless systems. This technology has been integrated into several applications as a result of its numerous advantages, including low production costs, high-quality factor and ease of integration into telecommunication circuits.

Filters and feeder-to-filter transitions are just a few examples of the many projects that have utilized SIW technology, which can make converting nonplanar circuits into planar circuits easier for simple integration with other planar circuits and systems. To address this issue, researchers decided to construct a planar structure and the SIW, which is a very encouraging arrangement.

In addition to the conventional SIW, several new structures have been proposed to decrease size, boost single-mode bandwidth and lower losses. A waveguide width a reduction of 50% is possible with the folded SIW [3]. The ridge [4] and the empty [5] SIW increase the bandwidth while decreasing the size and the half-mode SIW decreases the width while increasing the single-mode bandwidth by a factor of two.

Similar to this, several brand-new SIW filter setups have been proposed to enhance filter performance, minimize footprint and lower losses [6]. SIW cavities in the half mode [7] and the quarter mode [8] have all been used to reduce the size of the filter. A small size and a wide stopband have also been achieved using a substrate-integrated coaxial line. Tiny SIW cavities utilizing interdigital capacitors have been proposed as a way to reduce filter dimensions.

A preliminary analysis of a bandpass filter based on a regularly drilled SIW structure was recently

1. A. Abed and R. Bendoumia are with DIC Laboratory, Department of Electronics, University of Blida 1, Algeria. Emails: abedahcene@gmail.com and r.bendoumia@yahoo.fr
2. A. Bouchekhlal is with Higher School of Signals (HSS), Koléa, Tipaza, Algeria. Email: bouchekhlalahmed@umc.edu.dz
3. A. Amrouche is with Computational Linguistics Department, CRSTDLA, Algeria. Email: amrouche-a.dz@ieee.org

presented. In this structure [9], perforations in the dielectric substrate allow for a local effective permittivity reduction in this structure, resulting in waveguide sections below cut-off.

[10] described a new SIW filter class based on dielectric substrate perforations. The air-hole perforations enable the SIW's characteristic impedance and cut-off frequency to be modified locally, allowing for filtering structures' straightforward design. The design of four-pole filters operating at 3.6GHz is tested and discussed. The filter in [11] changes the effective dielectric permittivity and, as a result, the waveguide's characteristic impedance locally. Perforations in particular are used to form a waveguide section below the cut-off of the filter band. Impedance inverters that connect half-wave SIW resonators are realized using the waveguide sections below cut-off.

[12] described a novel SIW filter with periodic dielectric layer perforations. Waveguide sections are below cut-off, because perforations reduce local effective dielectric permittivity. This effect is used in the implementation of immittance inverters *via* analytical formulae that provide simple design rules for direct filter synthesis.

Costa et al. [13] proposed the X-band wireless bandpass filter based on an E-plane SIW (SIEW) structure. The prototype has a bandwidth of 1.2GHz and a resonance frequency of 10.54GHz, which is defined as the −3dB transmission coefficient reference level. Rhbanou et al. [14] used an iris and a SIW resonator topology to demonstrate a high-rejection K-band SIW bandpass filter design.

Nwajana et al. [15] proposed a transition structure from CPW-to-SIW, where the input/output couplings of the filter were controlled by the CPW-to-SIW transition structures, which took advantage of the step impedance between the 50ω input/output feedline and the transition, while the SIW filter has a very low milling or etching requirement, which reduces fabrication error. Nawaz et al. [16] developed a technique for switching from SIW to microstrip. The transition is wide-bandwidth, spanning the frequency range from 8 to 12GHz [17]. The measured return loss is less than 10dB and the in-band insertion loss is less than 0.6dB.

Caleffo et al. [18] proposed a new method for determining the physical size of the tapered transition, which achieves impedance matching without the need for any computational optimization between the feeder with built-in microstrip technology and SIW. Fellah et al. [19] proposed using a brand-new half-mode SIW (HMSIW) bandpass filter based on defective ground structure cells (DGSs). To meet design specifications for small size, low insertion loss and high rejection, using the periodic square Complementary Split Ring Resonator (CSRR) resonant characteristics of DGS, an X-band bandpass filter is made and examined. This study investigates a new type of filter topology based on SIW technology that can meet all of these band- pass filter structure design specifications. Filters are required in every field that uses telecommunications, including space telecommunications, because of their increasing importance in transmission systems.

The desire to develop a low-cost filter with adequate performance drives our work. Furthermore, the proposed structure allows for the control of the filter's bandwidth. To address this issue, we proposed a new type of filter by dividing the waveguide into two cavities and connecting them with metallic *vias*; this structure allows us to vary the distance, allowing us to pass the frequency of interest. The lower cost of the substrate in comparison to the others demonstrates the choice. This structure is suitable for systems that do not require a high level of performance.

To accomplish our goals, we have divided this article into three sections, which are organized as follows: The first section describes the C-band and the filtering function which relies on the substrate-integrated waveguides; the second section focuses on the C-band filter design. Section three will present the simulation results of the running SIW filter.

2. C-BAND SIW BANDPASS FILTER

C-band is the part of the electromagnetic spectrum characterized and sits between the two Wi-Fi bands, which are 2.4GHz and 5GHz. Likewise, for the 2.6GHz band that Clearwire and later Sprint used for 4G starting in 2007 and any T-Mobile in their current mid-band 5G uses. Owing to its extensive geographic coverage area and strength in a variety of propagation conditions, the C-band is widely used for satellite communications.

While the C-band spectrum has traditionally been reserved exclusively for satellite use, there is a trend

toward allocating some C-band spectrum to terrestrial broadband operators in preparation for the up-coming deployment of 5G. Its transmission attributions are truly outstanding and its security is well established.

2.1 Analog Bandpass Filter

Originally, analog filters were constructed using resistance, capacitance and inductance. Their primary application was for frequency division multiplexing systems in the telephone industry. In integrated circuits that combine analog and digital signal processing tasks, analog filters are becoming more and more significant. Mixed-signal design is the process of making such circuits. The amplitude and phase spectra of an analog filter are used to determine its properties. Frequently, the amplitude spectrum as a function of ω (or f) is used. The magnitude is usually expressed in logarithmic units of dB (decibels). The ordinate units for the phase spectrum are degrees or radians *versus* ω (or f).

Analog filters can be classified into four categories: Low-Pass, High-Pass, Band-Reject and Bandpass. The Bandpass filter for C-band applications is the main topic of this work.

A bandpass channel is utilized when you need to communicate signals in a specific recurrence band and block signals at higher and lower frequencies, ω_1 and ω_2, respectively. This is finished by "tuning" an ideal recurrence, for example, a radio or TV signal. The ideal recurrence band is based on a recurrence ω_0 called the middle recurrence.

The recurrence groups underneath or more than the pass-band are called stop groups. The band edge frequencies are those frequencies where the pass-band and stop-band intersect.

2.2 Classical Waveguide

A waveguide is a physical system that guides electromagnetics beyond a certain distance to confine them in a specific medium. Waveguides are used in microwave transmission technology, obstacle detection, high-power and ultra-short-wave broadcasting technology, among other applications. Waveguides have recently adopted nano-level planar geometric shapes and materials to perform functions, such as coupling, modulation, multiplexing and amplification [20]. In terms of wave propagation, there are three modes to consider: TE, TM and TEM stand for transverse electric, magnetic and electromagnetic modes, respectively.

Only the dominant mode is propagated, which is distinguished by the lowest cut-off frequency f_c. We can really see this by using the following equation:

$$f_c = \frac{C}{2}\sqrt{\left(\frac{n}{a}\right)^2 + \left(\frac{m}{b}\right)^2} \quad (1)$$

where: a is the waveguide length; b is the waveguide width, C = 2.99 108m/s is the velocity, m and n are mode indices. Equation 1 calculates the cut-off frequency for any arbitrary mode in an RW. Furthermore, the cut-off frequency for the mode TE_{10} is given by.

$$f_c = \frac{C}{2a} \quad (2)$$

2.3 Substrate-integrated Waveguide (SIW)

Vertical metal walls are formed by two rows of metallic *via*-holes in contact with metal layers above and below the substrate (Figure 1). The latter is sandwiched between the two plates to allow the fundamental mode (TE_{10}) to propagate [21].

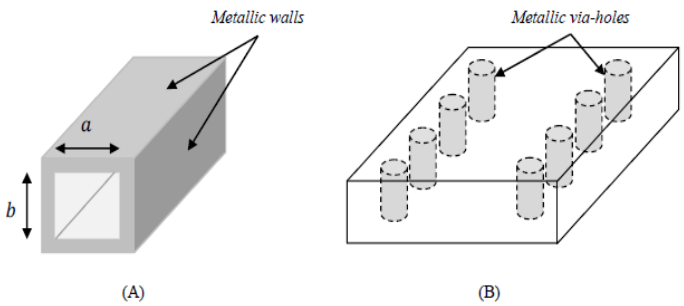


Figure 1. (A) Rectangular waveguide (B) Waveguide in SIW technology.

$$D_v \leq \frac{\lambda_g}{5} \quad (3)$$

$$P \leq 2D_v \quad (4)$$

a_d is given by the equation with the same cut-off frequency:

$$a_d = \frac{a}{\sqrt{\varepsilon_r}} \quad (5)$$

The following formula calculates the distance a_{SIW} between the two metallic *via*-hole cylinder rows.

$$a_{SIW} = a_d + \frac{D_v^2}{0.95P} \quad (6)$$

where: λ_g is the wave length of the waveguide, D_v is the *via* diameter and P is the spacing between two adjacent *vias*.

The dimension "b" is unimportant in the SIW filter for the TE_{10} mode, because it has no effect on the waveguide's cut-off frequency. As a result, the thickness of the substrate has no effect on the dielectric loss. The design and analysis of microwave bandpass filters using SIW technology are proposed and demonstrated in this article. Figure 1 (a) depicts a rectangular waveguide structure with dimension "b" representing the side walls of the rectangular waveguide and dimension "a" representing the top and bottom planes of RW.

3. METHODS AND MATERIALS

Based on SIW technology, we created a bandpass filter. The suggested filter works in the frequency band [4 − 8GHz]. This filter is obtained by dividing the waveguide into two cavities. Figure 2 depicts the SIW C-bandpass filter structure.

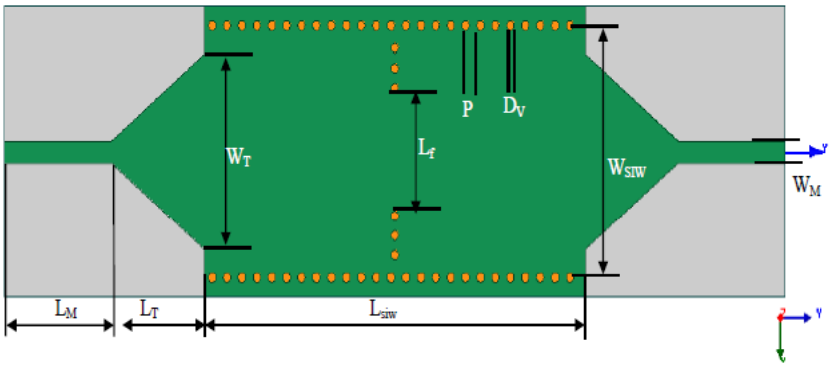


Figure 2. SIW bandpass filter structure.

where: W_{SIW} is the integrated waveguide width, L_{SIW} is the integrated waveguide length, W_T is the tapered line width, L_T is the tapered line length, W_M is the micro-strip line width, L_M is the micro-strip line length, L_f is the aperture length and h is the substrate thickness.

The standard RW of the C-band is WR-137. Table 1 summarizes its characteristics.

Table 1. Standard rectangular waveguides of the C-band.

Recommended Frequency Band	$5.8 - -8.20GHz$
Cut-off Frequency of Lower Mode	$4.301GHz$
Cut-off Frequency of Upper Mode	$8.603GHz$
Dimension $(m \times m)$	34.8488×15.7988

3.1 Microstrip Part Design

A quarter-wave tapered transformer and a 50Ω track are connected in series to form the micro-strip component. In this section, the design parameters are the transition-line length LT and width WT.

The width W_T of the tapered line can be found by solving Equation 7:

$$\frac{\frac{1}{4.38H}}{\left[\frac{W_T}{H}+1.393+0667\ln\left(\frac{W_T}{H}+1.444\right)\right]}=\frac{e^{\frac{-0.627\varepsilon_r}{\frac{\varepsilon_r+1}{2}+\frac{\varepsilon_r-1}{2\sqrt{1+\frac{12H}{W_T}}}}}}{a_d} \quad (7)$$

The taper length L_T is given by Equation 8:

$$L_T=\frac{\lambda_g}{4} \quad (8)$$

and

$$\lambda_g=\frac{C}{f\sqrt{\varepsilon_r}} \quad (9)$$

4. RESULTS AND DISCUSSION

The fundamental SIW mode has an electric field distribution similar to the TE_{10} mode of an RW. The filter is created by combining generalized bandpass filters. In order to use the designed filter for SIW C-band applications, the waveguide and hole dimensions are calculated and shown in Table 2. The hole dimensions are calculated using the C-band lowest mode's cut-off frequency f_c = 4.301 GHz. The multi-layer structure is built on an FR-4 substrate with a dielectric constant of 4.4 and a thickness of 1.6*mm*.

Table 2. Parameters of the proposed C-band bandpass filter.

Parameter	**Description**	**Dimension**
f_C	Cutoff frequency of lowest mode	4.301*GHz*
λ_g	Dielectric material wavelength	33.25 *mm*
W_{SIW}	SIW width	16.96 *mm*
D_V	Metal *via* diameter	0.66 *mm*
P	Space length between adjacent *vias*	1.33 *mm*
L_a	Aperture length	$0.24\lambda_g$
L_{SIW}	SIW length	33.92 *mm*
W_M	Tapered micro-strip width	1.53 *mm*
L_M	Tapered micro-strip length	9.55 *mm*
W_T	Transition width	13.22 *mm*
L_T	Transition length	8.31 *mm*

To realize the SIW filter and achieve our objectives, we first simulated it with the HFSS software, a powerful electromagnetic simulator for 3D problems. Then, we implemented the filter and tested it, and finally we compared the results to those of others in the field. The outcomes are denoted as the reflection coefficient S_{11} and the transmission coefficient S_{21}. The passband insertion loss of a perfect SIW filter must be 0 and the stopband attenuation must be infinite. The SIW filter in this work operates at frequencies ranging from 5.39GHz to 6.83GHz. The numerical analysis is used to determine the dimensions of the waveguide and *via* holes.

4.1 C-band SIW Bandpass Filter Simulation

Figure 3 depicts the distribution of the electric field. This distribution expresses the filtering function of the proposed structure. This is provided by the additional *vias* at the end of each cavity.

The proposed filter is designed and analyzed with HFSS software in the frequency band [5 − 8GHz]. We consider three values of L_f : $0.22\lambda_g$; $0.24\lambda_g$ and $0.26\lambda_g$.

Figure 4 depicts the proposed filter's reflection and transmission coefficients. The results show that this filter is bandpass. The frequency ranges of this filter are [5.6530 − 6.4600GHz] with $0.22\lambda_g$, [5.5450 − 6.5890GHz] with $0.24\lambda_g$ and [5.4400 − 6.8200GHz] with $0.26\lambda_g$. We notice that the chosen filter has $L_f = 0.24\lambda_g$ and performs well in terms of bandwidth [5.43 − 6.45GHz], its insertion loss −4.77dB and its return loss −11.48dB.

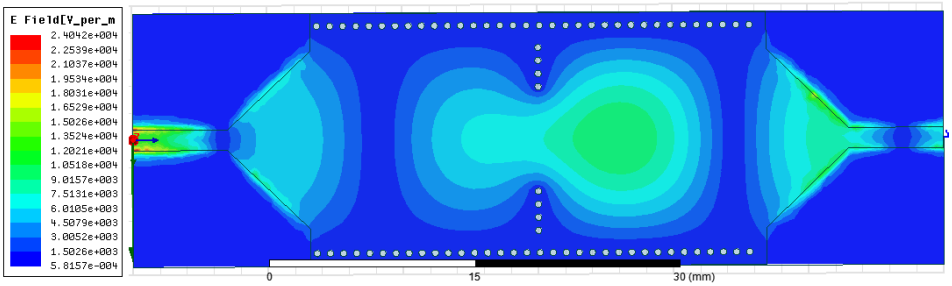


Figure 3. Electric field distribution of C-band bandpass filter.

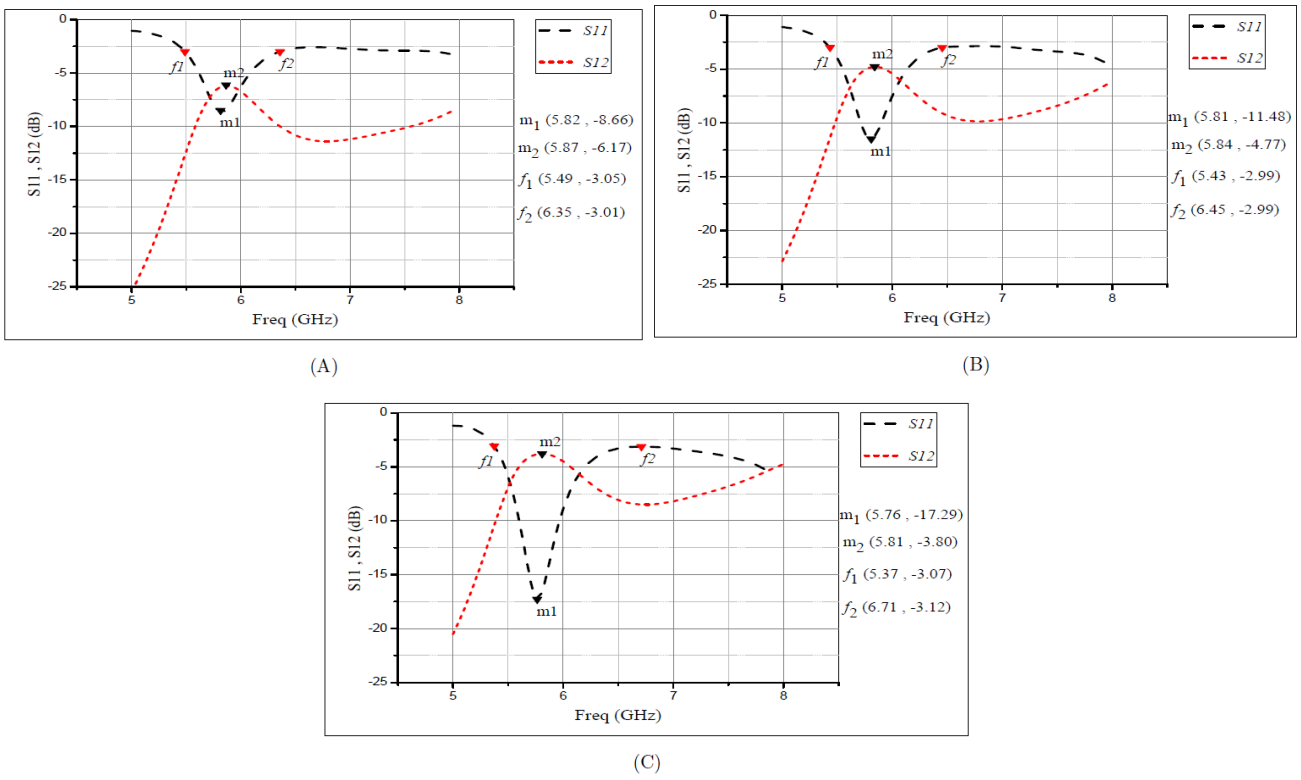
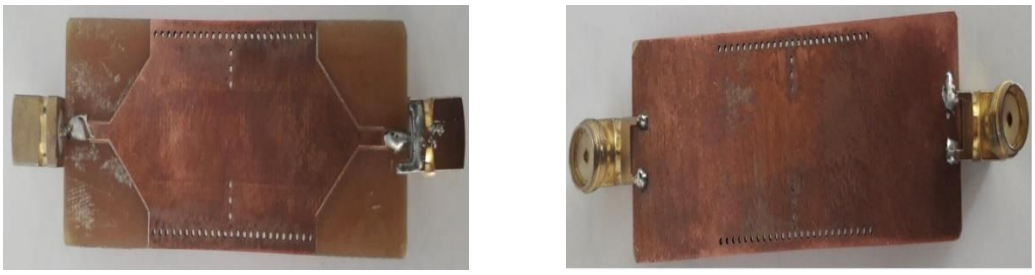


Figure 4. S-parameters of C-band bandpass filter (A) $0.22\lambda_g$ (B) $0.24\lambda_g$ (C) $0.26\lambda_g$.

4.2 C-band SIW Bandpass Filter Realization

After the simulation phase, we moved on to the practical realization of the simulated filter. The prototype is realized under a substrate of type Fr4-Epoxy with a dielectric constant of $\varepsilon_r = 4.4$ and a thickness of h = 1.6*mm*. Figure 5 shows the prototype of the SIW C-band filter:



(A) (B)

Figure 5. C-band bandpass filter (A) top view (B) bottom view.

To characterize this filter, the prototype is equipped with two ports for the SMA connectors. The dimensions of the SIW C-bandpass filter are grouped in Table 2.

Pressure contact is used in industrial applications, which is a sophisticated and expensive method of

soldering, whereas solder paste is simple to use and less expensive. However, in some cases, it has an impact on the performance of the system that uses it. In our case, we used solder paste to connect SMAs (connectors). Furthermore, because the voltage used is insufficient and the solder point does not heat up, there is no discernible effect on the filter performance.

The characteristics of the realized filter are measured with an Agilent PNA-L N5230 C network analyzer. Figure 6 represents the evolution of the parameter S11 measured with a network analyzer in the frequency range from 5 to 8GHz. For ease of comparison, the measurement and simulation results are shown together.

The results of the initial measurement and the simulation are clearly in excellent agreement in terms of insertion loss, as shown in Figure 6. The realized filter shows very high performances with a fractional bandwidth of 23.56%, a return loss of −45.49dB and an insertion loss of −4.2dB.

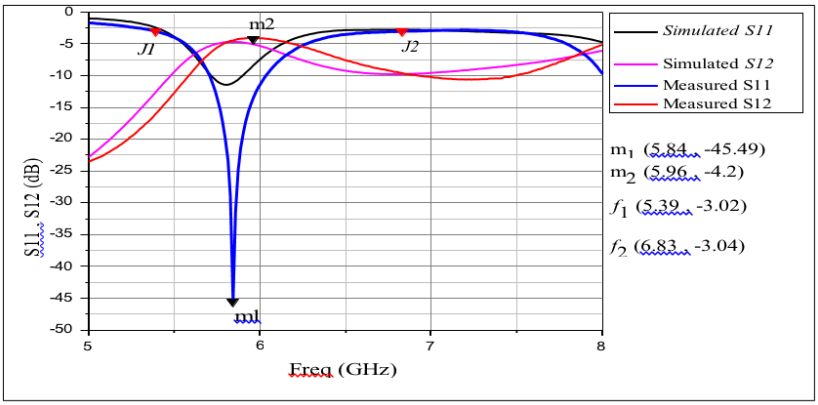


Figure 6. S-parameters of the realized filter.

4.3 Comparative Study

Table 3 compares the effectiveness of the proposed C-band SIW bandpass filter to some previously reported filters.

Table 3. A comparative study of our work and previously reported SIW bandpass filters. (ε_r: relative permittivity, Thick: thickness, f_c: center frequency, FBW: -3dB fractional bandwidth, RL: return loss (dB), IL: insertion loss (dB), λ_g: guided wavelength).

Ref.	**Substrate** $\varepsilon_r/Thick$	f_C *GHz*	**FBW (%)**	**RL (dB)**	**IL (dB)**	**Size** (λ_g^2)
[20] (I)	6.5/0.762	2.73	14.65	21	0.75	0.043
[20] (II)	6.5/0.762	1.72	5.7	17.75	1	0.017
[21] (SM)	2.2/0.508	2.45	24.1	26	0.38	0.048
[21] (TM)	2.2/0.508	2.45	36.7	18	0.25	0.048
[22]	2.33/0.787	1.84	14.67	23.9	0.51	0.0147
[23]	2.33/0.78	14.74	76	17.6	1.5	0.7035
This work	4.4/1.6	5.84	23.56	45.49	4.2	0.52

In the comparison process to some references in the field, the fractional bandwidth, return loss and insertion loss were used. According to the results, with f_c = 5.84GHz, the proposed filter performed well in terms of compact size and high in-band return. Measured return loss was improved to 45.49dB.

When compared to other works, insertion loss is high and this is due to two factors. The first is the dielectric constant of the Fr4-Epoxy substrate, which is fixed at εr = 4.4 and has a significant impact on it. We used various dielectric constant values (simulation) and the results show that as the dielectric constant value decreases, the insertion loss decreases to less than −0.9dB. The second factor is provided by the use of solder paste, but it has little effect on the performance of the realized filter.

5. CONCLUSION

This paper presented an SIW C-bandpass filter, which is a cavity-resonator SIW design. The methods for designing filter parts and microstrip parts are discussed. The pass-band of this filter is from 5.39 to 6.83GHz with an insertion loss of 4.2dB and a return loss of 45.49dB. When compared to traditional waveguide filters, this filter is less expensive and easier to integrate into various planar circuits. The presented results show that the simulated and measured filter responses are in good agreement in terms of insertion loss. Our findings are consistent with earlier studies on this subject.

The suggested filter has a number of benefits, such as easier integration, fractional bandwidth, lower insertion loss and high return loss when compared to other SIW bandpass filters. This SIW bandpass filter is appropriate for real-world use. As a perspective, we are working to optimize the parameters of the proposed filter to control its frequency band and integrate it into a real system

REFERENCES

[1] M. Rabah, M. Abri, H. Abri Badaoui, J. Tao and T. Hoa Vuong, "Compact Miniaturized Half-mode Waveguide/high Pass-filter Design Based on SIW Technology Screens Transmit-IEEE C-band Signals," Microwave and Optical Technology Letters, vol. 58, pp. 414–418, 2016.

[2] M. Bozzi, A. Georgiadis and K. Wu, "Review of Substrate-integrated Waveguide Circuits and Antennas," IET Microwaves Antennas & Propagation, vol. 5, no. 12, pp. 909–920, 2011.

[3] W. Che et al., "Analysis and Experiments of Compact Folded Substrate- integrated Waveguide," IEEE Transactions on Microwave Theory and Techniques, vol. 56, no. 1, pp. 88–93, Jan. 2008.

[4] S. Moscato et al., "Two-material Ridge Substrate Integrated Waveguide for Ultra-wideband Applications," IEEE Trans. on Microwave Theory and Techniq., vol. 63, no. 10, pp. 3175–3182, 2015.

[5] F. Parment et al., "Air-filled Substrate Integrated Waveguide for Low-loss and High Powerhandling Millimeter-wave Substrate Integrated Circuits," IEEE Transactions on Microwave Theory and Techniques, vol. 63, no. 4, pp. 1228–1238, Apr. 2015.

[6] X.-P. Chen and K. Wu, "Substrate Integrated Waveguide Filter: Basic Design Rules and Fundamental Structure Features," IEEE Microwave Magazine, vol. 15, no. 5, pp. 108–116, Aug. 2014.

[7] R. Moro, S. Moscato, M. Bozzi and L. Perregrini, "Substrate Integrated Folded Waveguide Filter with Out-of-band Rejection Controlled by Resonant-mode Suppression," IEEE Microwave and Wireless Components Letters, vol. 25, no. 4, pp. 214–216, Apr. 2015.

[8] S. Moscato, C. Tomassoni, M. Bozzi and L. Perregrini, "Quarter-mode Cavity Filters in Substrate Integrated Wwaveguide Technology," IEEE Transactions on Microwave Theory and Techniques, vol. 64, no. 8, pp. 2538–2547, Aug. 2016.

[9] A. P. Saghati, A. P. Saghati and K. Entesari, "Ultra-miniature SIW Cavity Resonators and Filters," IEEE Trans. Microw. Theory Techn., vol. 63, no. 12, pp. 4329–4340, Dec. 2015.

[10] L. Silvestri, E. Massoni, C. Tomassoni, A. Coves, M. Bozzi and L. Perregrini, "A New Class of SIW Filters Based on Periodically Perforated Dielectric Substrate," Proc. of the 46th European Microwave Conf. (EuMC), pp. 775–778, London, U.K., Oct. 2016.

[11] L. Silvestri et al., "Modeling and Implementation of Perforated SIW Filters," Proc. of the IEEE MTT-S Int. Conf. on Numerical Electromagnetic and Multiphysics Modeling and Optimization (NEMO2016), pp. 1–2, Beijing, China, Jul. 2016.

[12] L. Silvestri, E. Massoni, C. Tomassoni, A. Coves, M. Bozzi and L. Perregrini, "Substrate Integrated Waveguide Filters Based on a Dielectric Layer with Periodic Perforations," IEEE Transactions on Microwave Theory and Techniques, vol. 65, no. 8, pp. 2687-2697, Aug. 2017.

[13] H. C. N. Claudio, P. Costa, A. G. D'Assunção and A. G. D. Junior, "Simulation and Design of a Band-pass Filter Based on Substrate Integrated E-plane Waveguide," Journal of Microwaves, Optoelectronics and Electromagnetic Applications, vol. 18, no. 3, pp. 390-398, 2016.

[14] A. Rhbanou, M. Sabbane and Seddik Bri, "Design of K-band Substrate Integrated Waveguide Band- pass Filter with High Rejection," Journal of Microwaves, Optoelectronics and Electromagnetic Applications, vol. 14, no. 2, pp. 155–169, 2015.

[15] N. Augustine, A. Dainkeh and K. Yeo, "Substrate Integrated Waveguide (SIW) Bandpass Filter with Novel Microstrip-CPW-SIW Input Coupling," Journal of Microwaves, Optoelectronics and Electromagnetic Applications, vol. 16, no. 2, pp. 393-402, June 2017.

[16] Z. H. M. I. Nawaz and M. Kashif, "Substrate Integrated Waveguide (SIW) to Microstrip Transition at X-band," Proc. of the Int. Conf. on Circuits, Systems and Control, pp. 61–63, [Online], Available: https://www.inase.org/library/2014/interlaken/bypaper/CSC/CSC-09.pdf, 2014.

[17] L. Xu and Z. Chen, "Substrate Integrated Suspended Line (SISL) to Substrate Integrated Waveguide (SIW) Interconnect at 28GHz Band," Proc. of the 2019 Computing, Communications and IoT Applications (ComComAp), pp. 365–368, Shenzhen, China, 2019.

[18] C. R. Caranicola, "New Design Procedure to Determine the Taper Transition for Impedance Matching between Microstrip Line and SIW Component," Journal of Microwaves, Optoelectronics and Electro-magnetic Applications, vol. 15, no. 3, pp. 247-260, 2016.
[19] B. Fellah, N. Cherif, M. Abri and H. Badaoui, "CSRR-DGS Bandpass Filter Based on Half Mode Substrate Integrated Waveguide for X-band Applications," Advanced Electromagnetics, vol. 10, no. 3, pp. 39–42, 2021.
[20] S. K. K. Dash, Q. S. Cheng, R. K. Barik, F. Jiang, N. C. Pradhan and K. S. Subramanian, "A Compact SIW Cavity-backed Self-multiplexing Antenna for Hexa-band Operation," IEEE Transactions on Antennas and Propagation, vol. 70, no. 3, pp. 2283–2288, March 2022.
[21] D. Deslandes and Ke Wu, "Single-substrate Integration Technique of Planar Circuits and Waveguide Filters," IEEE Trans. on Microwave Theory and Techniques, vol. 51, no. 2, pp. 593–596, Feb. 2003.
[22] L. Qiang et al., "A Substrate Integrated Waveguide to Substrate Integrated Coaxial Line Transition," Progress In Electromagnetics Research C., vol. 36, pp. 249–259, 2013.
[23] Z. Troudi et al., "A Novel Compact Substrate Integrated Waveguide Filter Using Miniaturized Stepped Impedance Metamaterial Unit Cell," Progress In Electromag. Research C, vol. 108, pp. 49–61, 2021.
[24] Y. Yang et al., "Design of Compact Bandpass Filters Using Sixteenth Mode and Thirty-second Mode SIW Cavities," Progress In Electromagnetics Research Letters, vol. 75, pp. 61–66, 2018.
[25] R. K. Barik et al., "Design of Miniaturized SIW Filter Loaded with Open-Loop Resonators and Its Application to Diplexer," Radioengineering, vol. 29, no. 4, pp. 609–616, Dec. 2020.
[26] T. Duraisamy et al., "Compact Wideband SIW Based Bandpass Filter for X, Ku and K Band Applications," Radioengineering, vol. 30, no. 2, pp. 288–295, June 2021.

ملخص البحث:

تصف هذه الورقة مرشّح تمرير نطاقٍ تردّدي مُدْمَج في طبقة أساسٍ من نوع إيبوكسي FR4، علماً بأنّ هذه التّقنية قد جذبت انتباه الباحثين في مجال المرشّحات الإلكترونية في الآونة الأخيرة بالنّظر الى ما تتميز به من انخفاض الفقْد وصغر الحجم وانخفاض التكلفة للمرشّحات النّاتجة عنها.

ويتمثل الهدف في محاكاة وتحقيق مرشّح تمرير نطاق تردّدي مُدْمَج في طبقة أساس يصلح لتطبيقات نطاق (سِي). وقد جرى تحليل المرشّح المقترح باستخدام مُعامِل الانعكاس وتوزيعات المجال الكهربائي، علماً بأنّه تمّ استخدام مُحاكي HFSS لإجراء عملية المحاكاة. وأشارت النتائج التي تمّ الحصول عليها الى درجة عالية من الاتّفاق بين المحاكاة والتّجارب العملية. كذلك بيّنت النّتائج أنّ المرشّح المقترح كانت له استجابة عالية جداً. أمّا نطاقهُ التردّدي لتمرير المرشّح فتراوح من (5.39) جيجاهيرتز الى (6.83) جيجاهيرتز، وبلغ فَقْد الإدخال للمرشّح (4.2) ديسيبل، وفقْد الإرجاع (45.49) ديسيبل.

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 09, No. 02, June 2023.

FORENSIC ANALYSIS OF DRONE COLLISION WITH TRANSFER LEARNING

Arda Surya Editya, Tohari Ahmad and Hudan Studiawan

(Received: 13-Jan.-2023, Revised: 19-Mar.-2023, Accepted: 4-May-2023)

ABSTRACT

Drones are among the devices that are used in many different activities. There is a time when drones have accidents and authorities need to find the cause. Drone forensics is used to determine the cause of an accident. The analysis phase of drone forensics is one of the most important steps in determining accident causes. In this paper, we applied a deep-learning technique to classify drone collisions. We investigate the use of Inception V.9 as the deep-learning framework. Additionally, this study compares the performance of the proposed method with other techniques, such as MobileNet, VGG and ResNet, in classifying drone collisions. In this experiment, we also implement transfer learning as well as its fine tuning to speed up the training process and to improve the accuracy value. Additionally, our investigation shows that Inception V.9 outperforms other frameworks in terms of accuracy, precision and F1 score.

KEYWORDS

Inception, MobileNet, VGG, RestNet, Drone forensics, Transfer learning, Network infrastructure.

1. INTRODUCTION

Drones are devices that have been developed and used in both academia and industry over the past few years. This type of device can assist humans working in many sectors to complete a particular task [1]. When an accident occurs, for example when a drone breaks or crashes, users need to find the cause of the incident. Drone forensics is a sub-discipline of digital forensics dedicated to drone investigations. Drone forensics can be used to discover evidence from various types of media collected by a drone to determine the cause of an accident [2].

As an example, when a drone has an accident while taking a picture in a hilly area, the owner can find out what caused the accident. To find out the cause of the accident, one can look at the media installed on the drone. This includes the camera on the drone and the sensor data log on the drone. Therefore, the causes of drone accidents can be known. In drone grand prix, drone-collision classification is used as evidence. To determine whether there is a foul in a race where drones collide, the referee needs to know the cause of the collision. Furthermore, drone-collision classification can be used in military applications. It is possible for a drone to collide with an enemy or to be attacked by an enemy if it flies into enemy territory. In order to conclude that a drone collision occurred, the pilot or the authority needs evidence. Alternatively, when drones are used in the entertainment industry, the director or the industry may need to know how the drone was involved to determine whether the worker WBB is negligent.

In drone forensics, there are three types of evidence that can be examined: antemortem, live and post-mortem data [3]. Several previous studies used post-mortem data to analyze the cause of a drone crash, such as a collision. Pedro et al. [4] used post-mortem data to build a model that could detect a collision with a moving object. In their study, MobileNetV2 was applied to extract features from acquired video and an agglomerative algorithm was then applied to determine object positions. However, this approach has certain weaknesses. For example, it took a long time to train the model and its accuracy could still be improved.

Furthermore, Jain et al. [5] examined basic drone architecture to gain evidence that can be applied to forensic analysis. In their study, they also proposed a generic drone forensic model to improve the digital investigation process. The approach has helped another study begin investigating drone forensics and further improvements can be made with an analysis method based on deep learning.

A study on improving drone forensics was also conducted by Ashraf et al. [6]. In their study, Ashraf et al. [6] applied ResNet50 to detect drones from drone camera data. In their study, they used six datasets

A. Editya, T. Ahmad and H. Studiawan are with Department of Informatics, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia. Emails: ardasurya.tif@unusida.ac.id, tohari@if.its.ac.id and hudan@its.ac.id

and achieved an average Fl score of 0.92. However, using a different deep-learning architecture may give a better result.

Contribution. In this paper, we investigate drone collision using InceptionV3 [7] to classify collision events from frames extracted from drone videos. To reduce the computational process and speed up the training process, we implement a transfer-learning technique. The transfer-learning weights are taken from ImageNet [4]. We perform fine-tuning based on the weights used in InceptionV3 to obtain the most appropriate weights for the case of a drone collision.

This paper is organized as follows. Section 2 discusses related work in drone-collision detection and drone forensic framework. We describe how we analyze the collected data, validated our approach and implemented our scheme in Section 3. This paper presents proposed solutions and describes the experimental results from our transfer-learning method in Section 4. Finally, we present the conclusion and possible avenues for future research in Section 5.

2. RELATED WORK

2.1 Drone Collision Detection

Several drone research projects are focusing on collision detection. Pedro et al. [4] used post-mortem data to analyze drone collisions. They used MobileNetV2 and a convolutional neural network (CNN) to extract features from post-mortem data and detect collisions. An agglomerative algorithm was then used to detect the positions of the objects. The accuracy of the developed model increased the mean Average Precision (mAP) from 54.4% to 91.4%.

Furthermore, Anwar et al. [8] used training data from a drone computer simulation and transferred the results to the real operation of a drone. Their method was able to reduce both energy consumption and the detection error for collisions. As the research used computer simulation at the training stage, the weights of the model could not represent actual conditions. In a real situation, the model is strongly influenced by the drone camera.

In addition, Ouahouah et al. [9] used deep reinforcement learning to make a drone fly without colliding with any obstacles. This takes a lot of time to produce the model, because the method that they used is different from deep learning. Deep reinforcement learning involves training an agent to make a series of decisions in an environment with the goal of maximizing accuracy.

A study that implemented deep learning into the drone was also conducted by Obaid et al. [10]. The researchers used artificial neural networks (ANNs) to train a drone using simulation software. The accuracy obtained was 85.3% and this value can be optimized using a different architecture with a good dataset.

Rehmatullah and Kelly [11] used parallel tracking and mapping (PTAM) and dense tracking and mapping (DTAM) to detect an object that may have been hit by a drone in real time. This technique had an accuracy of 34.53% in detecting all objects from images captured by the drone. The drawbacks of this approach are that accuracy is relatively low and many GPU resources are needed to train the model.

Furthermore, Lima et al. [12] conducted research about collision detection and obstacle classification based on the physical interaction between a drone and the environment. The experiment used three classes of classification, including no collisions, collisions with soft objects and collisions with hard objects. The experiment used three different methods: FNN (Fully Convolutional Network), ResNet and Inception. The most accurate result of the experiment was from Inception with an accuracy of 98.7%.

Furthermore, Park and Kim [13] conducted research about an algorithm to detect fast drone movement. The algorithm needs the location and time of the drone. It makes the collision of the drone categorized as a hard collision or a soft collision.

Drone-collision research includes not only digital imaging data, but also drone sensor value. Weinert et al. [14] initiated research on drone-collision calculation. In their work, they performed simulations of drone collision impact based on dimension and maximum gross take of weight (MGTOW). Based on the results of this experiment, collisions with dimensions and MGTOWs have an effect on damage.

Moreover, Xiaohua et al. [15] concluded in their research that aircraft-nose shapes reduce drone-collision damage. The degree of aircraft nose damage is determined by the shape of the nose, especially

when the nose is collided with a drone. The research used ten simulations with a variety of parameters. The same problem was studied by Zhang et al. [16]. To minimize drone-collision risk, they calculated the aircraft's position when starting flying. Additionally, this research determined the equation that is used to optimize the calculation of the risk of aircraft collisions with drones.

In addition, the drone may perform a show when deployed at several events, which results in a swarm formation of drones. The drones need to calculate to avoid colliding with each other when they form a swarm. This problem was also studied by Naveed et al. [17]. Their research proposed an algorithm to optimize drone positioning in a swarm formation. In the research, they used the distance between drones as well as the value of standard deviation to make drones not collide with each other.

A similar research was conducted by Sabetghadam et al. [18] aiming to ensure that drones did not collide when flying together. The drones may assemble in a formation while flying in the air. In this case, drones may collide if the algorithm is incorrect. The research adopted Voronoi-based space partitioning to derive local constraints that guarantee collision avoidance with neighboring vehicles.

2.2 Drone Forensic Framework

Drone forensics is a method that collects all artifacts stored in the internal or external memory of the drone system. Such data includes flight records, flight control, sensor data and internal monitoring data. Al-Dhaqm et al. [3] proposed a forensic framework that could be used for most types of drones and comprised of three phases, specifically the preparatory, data-acquisition and data-analysis phases. The phases of this framework are described below.

1) Preparatory phase: This phase determines what components are installed on the drone. In addition, forensic evidence is identified, a pilot interview is conducted and the tools used for drone forensics are prepared.
2) Data-acquisition phase: All data artifacts are collected from media using live and post-incident acquisition and a report on the information collected from the drone is created.
3) Data-analysis phase: In this phase, the data is analyzed and conclusions are drawn regarding the reasons for damage to the drone.

We take this framework as the starting point for this paper and focus on the data-analysis phase. The goal of this work is to detect drone-collision cases accurately.

3. PROPOSED METHOD

As shown in Figure 1, the proposed method uses a video dataset from ColaNet [4]. This method begins with observations of datasets. In this phase, we observe the dataset and record the timestamp value at which the drones collide. Following that, we use Algorithm 1 to convert the video dataset into frames. By inserting the timestamp value into Algorithm 1, the system automatically converts the video into labeled frames. There are two labels, one for "normal" and one for "collision". Following that, all labeled frames are divided into three folders: training, validation and testing. Moreover, we train the whole dataset in the training folder with InceptionV3. We used the transfer-learning technique to speed up the training process. After we have trained the model using pre-trained data, we obtain initial results. In the subsequent step, the accuracy value is improved by fine-tuning the model.

3.1 Data Pre-processing

In this step, we convert 100 videos of drone collisions into labeled frames. It starts by inputting a timestamp value that we have gathered from drone-collision datasets. Algorithm 1 labels a frame as "normal" or "collision" after we enter a timestamp. In this step, we gathered 18,872 frames. Specifically, the collision class contain 1,380 frames, while the normal class contains 17,492 frames.

Labeled frames are placed into three folders; namely, training, validation and testing. Figure 2 shows sample frames that are labeled with normal and collision. The composition of folders is as follows. The training folder contains 11,380 frames, where 9,712 frames are labeled normal and 1,668 are labeled as collision. The validation folder contains 3,292 frames, where 2,856 frames are labeled as normal and 436 are labeled as collision. The test folder contains 4,200 frames, where 3,012 frames are labeled as

normal and 1,188 frames are labeled as collision. Overall, the training data represents 60.3% of the dataset, the validation data 17.44% and the testing data 22.26%. There are two classes of normal and collision in each dataset. Due to the unbalanced dataset, we divided the dataset using these percentages so that the model can run well.

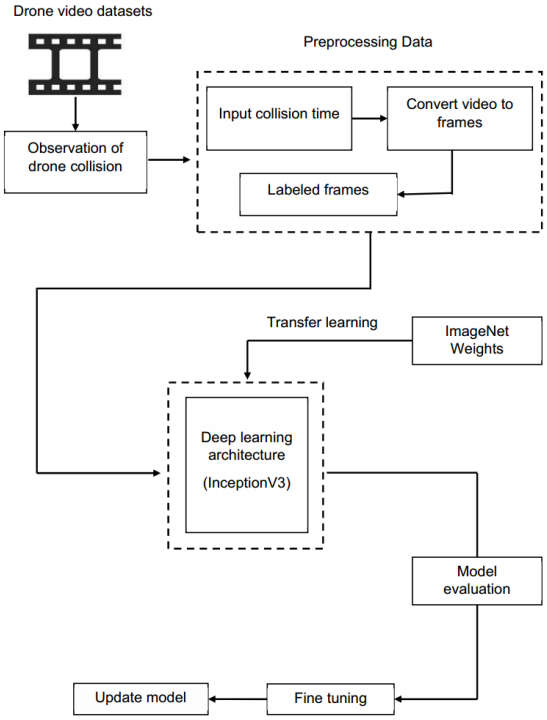


Figure 1. Research flow diagram.

Algorithm 1. Algorithm to convert video into labeled frames

```
for iteration =1, 2, ... do
    if time>time_collision then
        frame = collision
    else
        frame = normal
    end if
end for
```

3.2 Transfer Learning

Transfer learning is a technique to speed up the training process in deep learning. The idea behind transfer learning is to apply an existing model that has trained on massive datasets to a new task in order to classify another case. Figure 3 shows the diagram of transfer learning, where it uses the domain, task and marginal probabilities framework to explain transfer learning [19]. The domain of transfer learning can be formalized as in Equation (1) below:

$$D = \{x, P(X)\} \tag{1}$$

where D is defined as a domain built by two elements of a tuple containing a feature space that is represented by x, while $P(X)$ is a marginal probability of sample data points. In each specific domain, there is a task to determine the probabilistic view point that is shown in Equation (2) below:

$$T = \{\gamma, \eta\} \tag{2}$$

where T is a task running on each transfer learning domain, γ is defined as a label of a frame and η is a predictive function obtained from the learning process. It can also be demonstrated that using $P(Y|X)$ represents the prediction of the corresponding label.

In this paper, we employ the models from lmageNet and apply them to the drone-collision case. Transfer

learning involves several components, specifically a deep-learning architecture and pre- trained model templates from ImageNet. We use InceptionV3 for deep-learning architecture in this work.

Figure 2. Frame label from the ColaNet dataset.

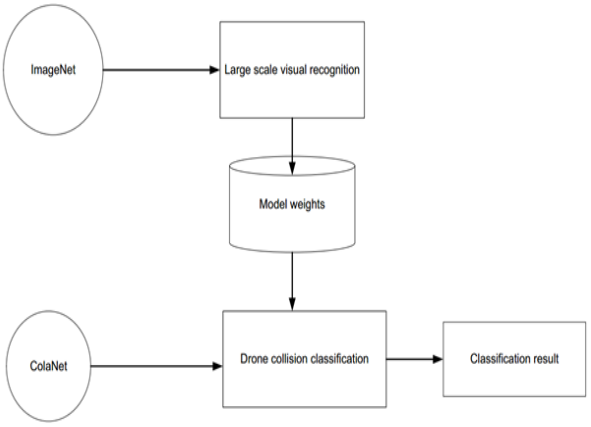


Figure 3. Block diagram of transfer learning.

3.3 Transfer Learning with InceptionV3

InceptionV3 is a deep-learning method that uses the CNN architecture. It combines several convolutions and its results are combined into one to improve the performance of the model. InceptionV3 is commonly used in a classifying task [7]. It performs the procedure using convolution on the input with different filter sizes, such as 1×1, 3×3 or 5×5. In the last part of the architecture, there is a max pooling layer and each result is sent to the subsequent inception module [7].

The previous inception methods, InceptionV1 and InceptionV2, use a lot of convolutional processes, which makes the CPU work harder. To tackle this issue, InceptionV3 limits the total number of input channels by adding an extra 1×1 convolution before 3×3 and 5×5 convolutions. By adding 1×1 convolution, the CPU does not have a heavy computation, because 1×1 convolution is lighter than other ones. This process can reduce the number of input channels. However, adding 1×1 convolution after the maximal pooling layer has been introduced in another work [20]. In addition, lnceptionV3 has the label smoothing regularization (LSR) which makes InceptionV3 more adaptable and performs well when handling low-resolution images. LSR can be described using the mathematical formula shown in Equation (3) below:

$$H(q', p) = -\sum_{k=1}^{K} \log p(k) q'(k) = (1 - \varepsilon) H(q, p) + \epsilon H(u, p) \quad (3)$$

LSR is equivalent to replacing single cross-entropy loss defined by $H(q, p)$. q is the ground truth distribution, u represents the uniform distribution and p is the predicted distribution. In the equation, epsilon is the value for label smoothing. Label smoothing can help improve the generalizability of a model by reducing the confidence of the model in its predictions. This can be especially useful when training on a. dataset with a large number of classes or when the model is overfitting. It can provide regularization and a more adaptable model. The main procedure of this operation is depicted in Figure 4.

3.4 Model Fine-tuning

In deep learning, fine-tuning is the process of updating the weights to increase accuracy. In this study, fine-tuning is applied in transfer learning. This is because we apply a pre-trained model to learn novel cases in transfer learning [21]-[22]. As a result of using the accuracy value in a new case, we may get a bad result. To be applied to new cases, the pre-trained model needs to be fine-tuned.

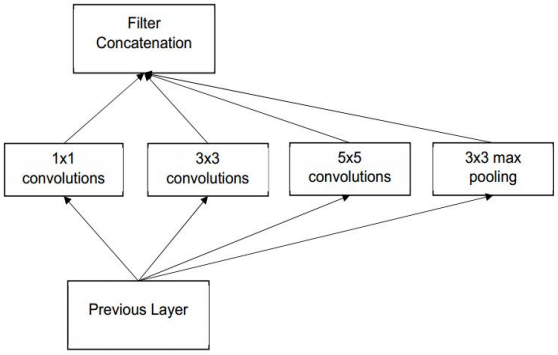


Figure 4. The workflow of InceptionV3.

Table 1. Layers of deep-learning architectures used for experiments.

No.	Deep learning architecture	Number of layers
1	InceptionV3	311
2	MobileNetV2	154
3	ResNetV2	190
4	VGG16	19
5	VGG19	22

A fine-tuning process takes less time than a training process, since we unfreeze a few nodes and update their weights based on the training process. The benefit of this method is that we do not have to spend a lot of time creating a customized model [23]. Figure 5 shows how the fine-tuning progress begins to convert data from datasets into convolutional layers. The process continues until all of the data enters the fully connected (FC) layer. The FC layer is used to detect specific global configurations of the features detected by the lower layers of the network. They usually sit at the top of the network hierarchy, at a point when the input has been reduced (by the previous, usually convolutional layers) to a compact representation of features. Each node in the FC layer learns its own set of weights from all of the nodes in the convolution layer.

The FC layer may have the same level as the softmax layer. The softmax is a function that turns a vector of *K* real values into a vector of *K* real values that sum to 1. The input values can be positive, negative, zero or greater than one, but the softmax transforms them into values between 0 and 1, so that they can be interpreted as probabilities. The fine-tuned process concentrates on the weight in the FC layer, which can make the output from the softmax layer appear high. It increases accuracy and minimizes loss.

4. EXPERIMENTAL RESULTS AND ANALYSIS

4.1 Environment Setup

In this sub-section, we describe the environment used in this experiment. First, we implement the InceptionV3 model using TensorFlow version 2.10.0. In this experiment, we also compared InceptionV3 [7] with other deep-learning architectures, such as MobileNetV2 [24], ResNetV2 [25], VGG16 [26] and VGG19 [26]. The specifications of ea.ch deep learning architecture are shown in Table 1.

Based on Table 1, we can see that InceptionV3 has the most number of layers. In this experiment, we used the transfer-learning technique to make the training process faster. The weights were obtained from lmageNet [4], [27][28][29] and then transferred to the discussed deep-learning architecture.

4.2 Training Results with Pre-trained Model

We run a pre-trained model based on the weights from the ImageNet dataset and evaluate our model based on the accuracy value. The experiment uses 32 batches and 10 epochs for each model. The results are presented in Figure 6.

Five deep-learning architectures have been tested, as shown in Figure 6.Each deep-learning architecture produces different reeults.lnoeptionV3 has a high accuracy value, with values ranging from 87% to 98%. This is because lnceptionV3 architecture has several layers that execute in parallel. In addition, there is a pool that accommodates each layer process that works in parallel, which increases the accuracy value. This is shown in Figure 6, where it can be seen that the results of the first test are similar to the training accuracy at the tenth epoch. The most significant advantage of Inception V3 is that this method can be applied to complex images. This is because the architecture allows several blocks to run in parallel, making it possible to obtain many features. The main weakness of this method is that there are many blocks that run in parallel. The computational needs are also more intense, meaning that this approach requires a very large number of resources [30].

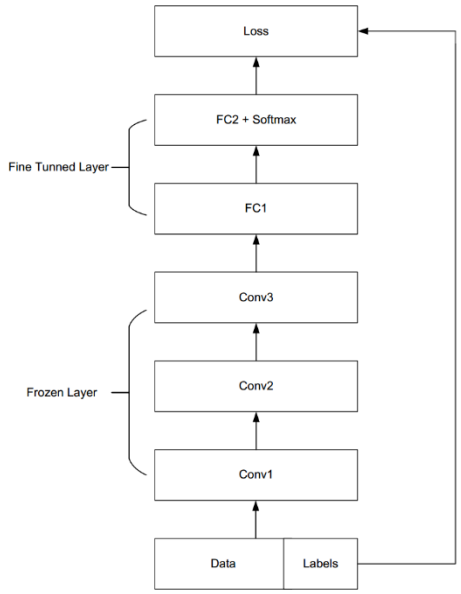


Figure 5. Block diagram of fine-tuning.

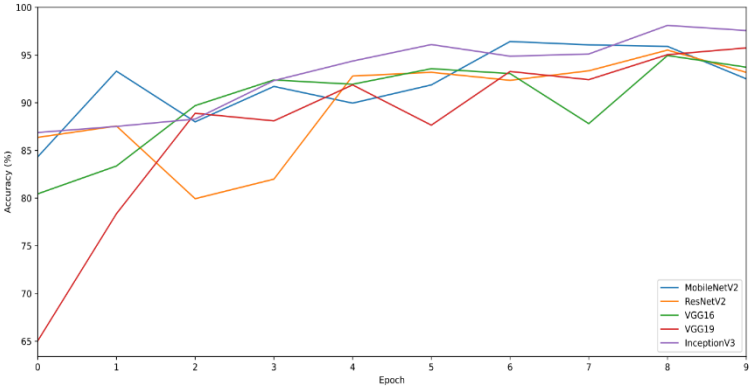


Figure 6. Accuracy of deep-learning architectures using pre-trained model.

For MobileNetV2, the resulting graph shows an unstable behavior, as depicted in Figure 6. The accuracy is between 85% and 95%. This is because the MobileNet V2 architecture uses a residual connection, which aims to keep the image constantly having the same color channel [11], [31]. This approach has certain advantages and disadvantages. The main advantage is that the frames have lower information loss, since MobileNetV2 stores the information contained in each frame using the residual connection. If we have two frames with a high degree of similarity, this can increase accuracy [4]. The primary disadvantage is that the classification results will be inaccurate if the original data contains significant noise or inappropriate features [5].

ResNetV2 provides lower accuracy values. Its results are between 81% and 92%. This is because ResNetV2 uses the value of each node obtained from the results of calculations based on the previous value. This architecture also uses a skip connection to skip nodes with the poorest accuracy value [6], [32].

For VGG16, the resulting accuracy is unstable, with values ranging between 81% and 91%, as shown in Figure 6. This is because in the VGG16 architecture, the frame is convoluted. Each process block in the architecture has a filter, which is classified at the end of the process using a fully connected neural network or a softmax layer [32]. The accuracy value obtained in the first test is unstable due to the loss of many features during the convolution process. In detecting a drone collision from image frames, it will be difficult to find features [33]. The VGG16 method is very effective in terms of recognizing objects. This is because the architecture finds the key features of the image data and when these are found, the classification accuracy is very high. The most significant drawback of this method is that if the image data does not have many features, the result will be out of focus or blurry [34]. The image frames of a drone in a collision scene have a lot of noise, for instance.

For the VGG19 method, the accuracy values range from 65% to 94%. This is because, according to the VGG19 architecture, each process block has a convolution process. Subsequently, it is classified at the end of the process using a fully connected neural network [27]. The first layer of VGG 19 applies a convolution process that is not too focused so that it can find numerous features in the image data. The data is then forwarded to the classification process in the fully connected block. VGG19 performs well if the image has many features or if the intended feature has a somewhat vague value. This method can find those features that will later be used in the classification process, meaning that accuracy is much more stable and relatively high. This method has a primary weakness in that if the processed image data is similar, the accuracy value will stack at a high level of accuracy.

Based on the graphs in Figure 6, it can be concluded that InceptionV3 has the highest accuracy. This is because the weight values in each node are drawn from ImageNet, a model that was trained on large amounts of image data and this allows the architecture to reach an accuracy value of above 60% on the

training process [35].

4.3 Fine-tuning Results

After testing the pre-trained model using weights from ImageNet, we modified and updated some node weights using the process described as follows. In this phase, we unfroze some nodes to update their weights. The updated numbers of nodes are shown in Table 2. After updating the node weights, we re-tested the model on the dataset and the results are shown as a graph of the accuracy values in Figure 7.

The accuracy of InceptionV3 improves significantly after model fine-tuning, as shown in Figure 7. This is because InceptionV3 uses a parallel process in which each process calculates a node weight value. In the final step, it combines all the node weight values, which has an impact on the current accuracy value, as shown in Figure 7. The accuracy reaches a peak value of 98%, because in the fine-tuning process, we modified all of the nodes in the softmax layer and InceptionV3 has more convolutional blocks. This is the reason why InceptionV3 achieves such high accuracy values. In addition, InceptionV3 combines residual blocks with numerous convolutional blocks and this procedure can increase the accuracy.

Table 2. Table of updated weight nodes.

No.	Deep-learning architecture	Number of updated weight nodes
1	MobileNetV2	56
2	ResNetV2	84
3	VGG16	2
4	VGG19	2
5	InceptionV3	130

For MobileNetV2, we can see that the accuracy value is more stable than before. This is because the value of each node in the fully connected layer or softmax layer has been modified. This makes it more accurate in terms of detecting whether the frame contains a collision or not. The weakness of this approach is that the accuracy plateaus at 97%, since in MobileNetV2, less computer memory is used to process image frames [36].

For ResNetV2, we can see that the accuracy value reaches a steady value after fine-tuning. This is because ResNetV2 uses a skip connection in each epoch to skip blocks that give poor performance, which affects accuracy. The accuracy reaches 96%, since in the convolutional block process, the frame may have lost much information, since ResNetv2 skips blocks with poor performance and frames containing collision may have even greater noise or blurriness [37].

For VGG16, we can see the accuracy value becoming more stable than before. This is because VGG16 uses 16 block convolution layers to extract features from the image frame. In addition, it uses one fully connected neural network or a softmax layer to classify image frames. Figure 7 shows that the maximum accuracy value is around 94%. This is because in VGG16, the image quality is affected by the collision frame, which has more noise and is blurry. If noise and blurry frame images are processed by a convolutional block, many of the information in those frames will be lost. In addition, VGG16 does not use a residual block that causes a vanishing gradient. This means that this method failed to find a specific feature m several frames [38].

For VGG19, the accuracy value is lees accurate when we fine-tune the VGG19 model. The reason is that VGG19 uses more filters before it processes a fully connected CNN. The accuracy value is around 91%. VGG19 has more filters that give the frame a feature. This causes an explosive gradient in the output. This means that in several frames, VGG19 fails to find features, especially in collision frames, since these contain a great deal of noise and ate blurred.

4.4 Discussion

We compare the performance of InceptionV3 with other deep-learning architectures, specifically MobileNetV2, ResNet, VGG16 and VGG19. To begin with, we use the same configuration, including batch, epoch, optimizer and hardware. To compare deep-learning architecture performance, we calculate recall, precision and F1 score values. These three parameters are used to assess the performance of the

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 09, No. 02, June 2023.

architecture in the image data-classification process [39]. Precision is obtained using Equation (4), recall is calculated with Equation (5) and F1 score is obtained by Equation (6).

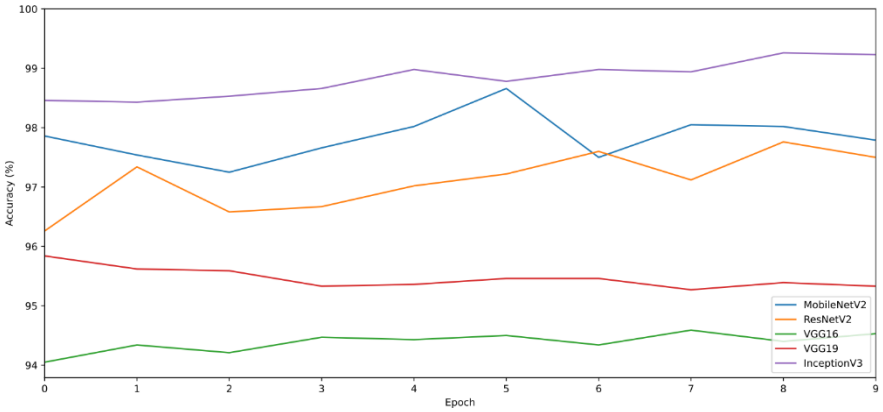


Figure 7. Accuracy values of deep-learning architectures after fine-tuning.

Table 3. Summary of model evaluation.

Deep-learning architecture	Accuracy (%)	Loss (%)	Recall (%)	Precision (%)	F1 score (%)
InceptionV3	99.35	3.95	98.23	99.54	99.56
MobileNetV2	98.18	7.48	68.56	64.56	66.56
ResNetV2	97.79	9.74	99.53	97.21	98.35
VGG16	94.27	14.09	99.48	87.34	93.45
VGG19	93.49	19.41	99.65	93.32	96.45

$$Precision = \frac{TP}{TP+FP} \quad (4)$$

$$Recall = \frac{TP}{TP+FN} \quad (5)$$

$$F1\ score = \frac{2\times precision\times recall}{precision+recall} \quad (6)$$

A summary of the results obtained using Equation (4) to Equation (6) is shown in Table 3. From Table 3, we can see that InceptionV3 has the best evaluation values with an accuracy of 99.35%, a precision of 99.54% and F1 score of 99.56%. This means that InceptionV3 has a stable model to classify drone collisions. When we use MobileNetV2, the opposite occurs. The method has a high accuracy of 98.18%, but a low precision of 66.56%. MobileNetV2 employs depth-wise block convolution, with each block consisting of a 3×3 convolution layer filtering the input, followed by 1×1 pointwise convolution combining the filtered values to create a new feature.

MobileNetV2 uses the depth-wise block convolution, where each block consists of a 3×3 convolution layer to filter the input, followed by a 1×1 pointwise convolution which combines the filtered values to create a new feature. In MobileNetV2, there are inputs and outputs between the models, while the inner layer encapsulates the model's ability to change the input from the pixel level concept to the image classification [40]. The total memory is constructed from all inputs and outputs in whole operations. Meanwhile, when we use the bottleneck residual block and make it a single operation, the memory will be filled by the bottleneck tensor instead of the internal tensors that can be larger. This means that this method cannot be used to classify drone collision although it has a good accuracy value. Also, MobileNetV2 has a lack of value in recall, precision and F1 score parameters. The lack of values of MobileNetV2 in those parameters is due to unbalanced datasets. We have a high recall value of 99.65% in VGG16, which could be because we rely on image quality in VGG16 and the collision frame has more noise and blur. If noisy and blurry frame images are processed by a convolutional block, they will lose much of the information that is contained in a frame. In addition, VGG16 does not use a residual block that causes a vanishing gradient. This means that this method failed to find a specific feature in several frames [38].

The VGG 19 result has the highest value in recall, because in VGG 19, there are more filters to reduce

noise in collision frames. In a collision frame, a blurred frame can affect the accuracy value. According to Equation (5), recall is proportional to the number of true positives (TP) and false negatives (FN), implying that VGG19 can accurately detect whether the result is positive or negative. Furthermore, ResNetV2 has the second highest recall after VGG19, because ResNetV2 used the residual weight to improve the ability of the model to classify the drone-collision frame. Since the collision frame is the blurred frame, it has an advantage for ResNetV2. Also, ResNetV2 uses a skip connection in each epoch to skip blocks that give poor performance, which affects accuracy.

5. CONCLUSION AND FUTURE WORK

In this research, we present a forensic analysis of drone collisions using a transfer-learning technique; namely, InceptionV3. After model fine-tuning, it was discovered that InceptionV3 was the best model, with a peak accuracy value of 99.35%. We also compared InceptionV3 with other architectures, such as MobileNetV2, ResNetV2, VGG16 and VGG19.

The model generated from experiments can be applied to drone forensic investigation based on videos found in the drone's storage. Although only collision videos from the ColaNet dataset were used in this paper, we believe that the results of this experiment can be applied to achieve faster classification of drone collisions for drone forensics.

The contribution of this paper is a methodology to classify drone-collision events based on frames extracted from drone videos using InceptionV3. By implementing a transfer-learning technique, we reduce the computational load and speed up the training process. Weights for transfer learning are taken from lmageNet. By fine-tuning the weights used in lnceptionV3, we are able to determine the most appropriate weights for drone collisions.

Future work will involve further updating the modules presented in this article. Data pre-processing will be improved through the use of neural network pipelines to classify frames containing collisions. We can improve the proposed method by minimizing the amount of human labor needed to classify the frames from the video datasets. In addition, this work can be extended by modifying our deep-learning framework to maximize accuracy and minimize loss. Finally, we intend to develop a detection system for attackers and collisions with objects, which may be useful in terms of gaining information to assist in forensic investigations.

REFERENCES

[1] L. P. Osco et al., "A Review on Deep Learning in UAV Remote Sensing," Int. J. of Applied Earth Observation and Geoinformation, vol. 102, p. 102456, 2021.

[2] A. L. Renduchintala, A. Albehadili and A. Y. Javaid, "Drone Forensics: Digital Flight Log Examination Framework for Micro Drones," Proc. of the 2017 Int. Conf. on Computational Science and Computational Intelligence (CSCI 2017), pp. 91-96, DOI: 10. 1109/CSCI. 2017. 15, 2018.

[3] G. Horsman, "Unmanned Aerial Vehicles: A Preliminary Analysis of Forensic Challenges," Digital Investigation, vol. 16, pp. 1-11, DOI: 10.1016/j .diin.2015.11.002, 2016.

[4] D. Pedro et al., "Collision Avoidance on Unmanned Aerial Vehicles Using Neural Network Pipelines and Flow Clustering Techniques," Remote Sensing, vol. 13, DOI: 10.3390/rs13132643, 2021.

[5] U. Jain, M. Rogers and E. T. Matson, "Drone Forensic Framework: Sensor and Data Identification and Verification," Proc. of the 2017 IEEE Sensors Applications Symposium (SAS), pp. l-6, DOI: 10. 1109/SAS. 2017.7894059, 2017.

[6] M. W. Ashraf, W. Sultani and M. Shah, "Dogfight: Detecting Drones from Drones Videos," CVPR, The Computer Vision Foundation, pp. 7063- 7072, DOI: 10.1109/CVPR46437.2021.00699, 2021.

[7] C. Szegedy, S. Ioffe, V. Vanhoucke and A. Alemi, "Inception-v4, Inception-resnet and the Impact of Residual Connections on Learning," Proc. of the 31st AAAI Conf. on Artificial Intelligence (AAAI'17), pp. 4278-4284, DOI: 10.48550.arXiv.1602.07261, 2016.

[8] A. Anwar and A. Raychowdhury, "Autonomous Navigation *via* Deep Reinforcement Learning for Resource Constraint Edge Nodes Using Transfer Learning," IEEE Access, vol. 8, pp. 26549-26560, DOI: 10 .1109/ACCESS.2020.2971 172, 2020.

[9] S. Ouahouah et al., "Deep-reinforcement-learning-based Collision Avoidance in UAV Environment," IEEE Internet of Things Journal, vol. 9, no. 6, pp. 4015-4030, DOI: 10.1109/JIOT.2021.3118949, 2022.

[10] A. A. Obaid and H. Koyuncu, "Obstacle Avoidance in Unmanned Aerial Vehicles Using Image Segmentation and Deep Learning," Proc. of the 2022 Int. Symposium on Multi-disciplinary Studies and Innovative Technologies (ISMSIT), pp. 912-915, DOI: 10.1109/ISMSIT56059. 2022.9932865, 2022.

[11] F. Rehmatullah and J. Kelly, "Vision-based Collision Avoidance for Personal Aerial Vehicles Using Dynamic Potential Fields," Proc. of the 12th Conf. on Computer and Robot Vision (CRY 2015), pp. 297-304, DOI: 10.1109/CRV.2015.46, 2015.

[12] R. R. Lima and G. A. S. Pereira, "Drone Collision Detection and Classification Using Proprioceptive Data," Proc. of the Int. Conf. on Unmanned Aircraft Systems (ICUAS), pp. 562-569, DOI: 10.1109/ICUAS54217.2022.9836207, 2022.

[13] J. K. Park et al., "UAV Collision Detection Algorithm Based on Formula for Fast Calculation," Turkish J. of Computer and Mathematics Education (TURCOMAT), vol. 12, no. 6, pp. 452-456, 2021.

[14] A. Weinert, L. Alvarez, M. Owen and B. Zintak, "Near Midair Collision Analog for Drones Based on Unmitigated Collision Risk," J. of Air Transportation, vol. 30, no. 2, pp. 37-48, 2022.

[15] X. Lu, X. Liu, Y. Li, Y. Zhang and H. Zuo, "Simulations of Airborne Collisions between Drones and an Aircraft Windshield," Aerospace Science and Technology, vol. 98, p. 105713, DOI: 10.1016/j.ast.2020 .105713, 2020.

[16] N. Zhang, H. Liu, B. F. Ng and K. H. Low, "Collision Probability between Intruding Drone and Commercial Aircraft in Airport Restricted Area Based on Collision-course Trajectory Planning," Transportation Research Part C: Emerging Technologies, vol. 120, p. 102736, DOI: 10.1016/j.trc.2020.102736, 2020.

[17] J. N. Yasin et al., "Formation Maintenance and Collision Avoidance in a Swarm of Drones," Proc. of the 3rd Int. Symp. on Computer Science and Intelligent Control, ACM, DOI: 10.1145/3386164. 3386176, 2019.

[18] B. Sabetghadam, R. Cunha and A. Pascoal, "A Distributed Algorithm for Real-time Multi-drone Collision-free Trajectory Replanning," Sensors, vol. 22, no. 5, DOI: 10.3390/s22051855, 2022.

[19] S. J. Pan and Q. Yang, "A Survey on Transfer Learning," IEEE Transactions on Knowledge and Data Engineering, vol. 22, no. 10, pp. 1345—1359, 2010.

[20] S. Arakelyan, S. Arasteh, C. Hauser, E. Kline and A. Galstyan, "Bin2vec: Learning Representations of Binary Executable Programs for Security Tasks," Cybersecurity, vol. 4, no. 1, p. 26, DOI: 10.1186/s42400-021-00088-4, 2021.

[21] A. Ramdan et al., "Transfer Learning and Fine-tuning for Deep Learning-based Tea Diseases Detection on Small Datasets," Proc. of the 2020 Int. Conf. on Radar, Antenna, Microwave, Electronics and Telecommunications (ICRAMET), pp. 206-211, DOI:10.1109/ICRAMET51080.2020 .9298575, 2020.

[22] J. Luttrell et al., "A deep Transfer Learning Approach to Fine-tuning Facial Recognition Models," Proc. of the 13th IEEE Conf. on Industrial Electronics and Applications (ICIEA), pp. 2671-2676, DOI: 10.1109/ICIEA. 2018.8398162, 2018.

[23] G. Vrbancic and V. Podgorelec, "Transfer Learning with Adaptive Fine-tuning," IEEE Access, vol. 8, pp. 196197-196211, DOI:10.1109/ACCESS.2020.3034343, 2020.

[24] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov and L.-C. Chen, "Mobilenetv2: Inverted Residnals and Linear Bottlenecks, arXiv: 1801.04381, pp. 4510-4520, DOI: 10.1109/CVPR.2018.00474, 2018.

[25] C. Szegedy et al., "Rethinking the Inception Architecture for Computer Vision," Proc. of the 2016 IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), pp. 2818-2826, DOI: 10.1109/CVPR.2016.308, 2016.

[26] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770-778, DOI: 10.1109/CVPR.2016.90, 2016.

[27] J. Jaafari, S. Douzi, K. Douzi and B. Hssina, "Towards More Efficient CNN-based Surgical Tools Classification Using Transfer Learning," Journal of Big Data, vol. 8, DOI:10.1186/s40537-021-00509-8, 2021.

[28] G. Li, Z. Ji, Y. Chang, S. Li, X. Qu and D. Cao, "ML-ANet: A Transfer Learning Approach Using Adaptation Network for Multi-label Image Classification in Autonomous Driving," Chinese J. of Mechanical Engineering (English Edition), vol. 34, DOI: 10.1186/s10033-021-00598-9, 2021.

[29] H.-T. Theng, C.-C. Hsieh, W.-T. Lin and J.-T. Lin, "Deep Reinforcement Learning for Collision Avoidance of Autonomous Vehicle," Proc. of the 2020 IEEE Int. Conf. on Consumer Electronics-Taiwan (ICCE-Taiwan), DOI: 10. 1109/ICCE-Taiwan49838.2020.9258199, 2020.

[30] A. Sarkar and Kalyani, "Survey of Smart Grid Network Using Drone and PTZ Camera," Proc. of the 3rd Int. Conf. on 2019 Devices for Integrated Circuit (DevIC), Kalyani, India, 2019.

[31] C. Shorten and T. M. Khoshgoftaar, "A Survey on Image Data Augmentation for Deep Learning," Journal of Big Data, vol. 6, no. 1, p. 60, DOI: 10.1186/s40537-019-0197-0, 2019.

[32] N. Best, J. Ott and E. J. Linstead, "Exploring the Efficacy of Transfer Learning in Mining Image-based Software Artifacts," Journal of Big Data, vol. 7, DOI: 10.1186/s40537-020-00335-4, 2020.

[33] N. Attari, F. Ofli, M. Awad, J. Lucas and S. Chawla, "Nazr-CNN: Fine-grained Classification of UAV Imagery for Damage Assessment," Proc. of the 2017 Int. Conf. on Data Science and Advanced Analytics (DSAA 2017), pp. 50-59, DOI: 10.1109/DSAA.2017.72, 2017.

[34] W. Qiu, S. Bi, C. Zhong, Y. Luo, J. Li and B. Sun, "Obstacle Avoidance of Aerial Vehicle Based on Monocular Vision," Proc. of the 2017 IEEE 7th Annual Int. Conf. on CYBER Technology in Automation, Control and Intelligent Systems (CYBER), (2017) 657-662, DOI: 10.1109/CYBER.2017.8446065, 2017.

[35] L. Huang, X. Feng, C. Zhang, L. Qian and Y. Wu, "Deep Reinforcement Learning-based Joint Task Offloading and Bandwidth Allocation for Multi-user Mobile Edge Computing," Digital Communications and Networks, vol. 5, pp. 10-17, DOI:10.1016/j.dcan.2018.10.003, 2019.

[36] Y. Chen et al., "Classification Methods of a Small Sample Target Object in the Sky Based on the Higher Layer Visualizing Feature and Transfer Learning Deep Networks," EURASIP Journal on Wireless Communications and Networking, vol. 2018, no. 1, p. 127, DOI:10.1186/s13638-018-1133-2, 2018.

[37] B. Wei, L. Zhang, K. Wang, Q. Kong and Z. Wang, "Dynamic Scene Deblurring and Image De-raining Based on Generative Adversarial Networks and Transfer Learning for Internet of Vehicle," EURASIP J. on Advances in Signal Processing, vol. 2021, no. 1, p. 121, DOI: 10.1186/s13634-021-00829-0, 2021.

[38] Y.-H. Choi et al., "Using Deep Learning to Solve Computer Security Challenges: A Survey," Cybersecurity, vol. 3, no. 1, p. 15, DOI: 10.1186/s42400-020-00055-5, 2020.

[39] X. Chen, W. Qi and L. Xi, "Deep-learning-based Motion-correction Algorithm in Optical Resolution Photoacoustic Microscopy," Visual Computing for Industry, Biomedicine and Art, vol. 2, no. 1, p. 12, DOI:10.1186/s42492-019-0022-9, 2019.

[40] P. Drews-Jr, I. d. Souza, I. P. Maurell, E. V. Protas and S. S. C. Botelho, "Underwater Image Segmentation in the Wild Using Deep Learning," Journal of the Brazilian Computer Society, vol. 27, no. 1, p. 12, DOI: 10.1186/s13173-021-00117-7, 2021.

ملخص البحث:

تُعدّ الطّائرات المسيّرة من بين الأجهزة التي تُستخدم في أنشطة منوّعة. وأحياناً تتعرّض تلك الطّائرات لحوادث، ممّا يجعل السّلطات في حاجةٍ الى الوقوف على أسبابها. وقد أصبح تحليل حوادث الطّائرات المسيّرة ومعرفة أسبابها مجالاً خصْباً للبحث.

في هذه الورقة، تمّ توظيف إحدى تقنيات التّعلُّم العميق من أجل تصنيف الاصطدامات التي تتعرض لها الطّائرات المسيّرة، وقد تمّ استخدام (Inception V9) كإطار عملٍ للبحث. من جانب آخر، تقارن هذه الورقة بين الطّريقة المقترحة وتقنياتٍ أخرى جرى استخدامها في أدبيات الموضوع من أجل تصنيف اصطدامات الطّائرات المسيّرة. وفي التّجارب المتعلّقة بهذا البحث، تم استخدام تعلُّم النّقْل والضّبْط الدّقيق لتسريع عملية تدريب النّظام وتحسين قيمة الدّقة التي يتمّ الحصول عليها من الطّريقة المقترحة. يُضاف الى ذلك أنّ التّجارب أثبتت أنّ الطّريقة المقترحة تفوّقت على الطّرق الاخرى التي تمّت مقارنتها بها من حيث مؤشّرات الأداء.

المجلة الأردنية للحاسوب وتكنولوجيا المعلومات (JJCIT) مجلة علمية عالمية متخصصة محكمة تنشر الأوراق البحثية الأصيلة عالية المستوى في جميع الجوانب والتقنيات المتعلقة بمجالات تكنولوجيا وهندسة الحاسوب والاتصالات وتكنولوجيا المعلومات.

تحتضن وتنشر جامعة الأميرة سمية للتكنولوجيا (PSUT) المجلة الأردنية للحاسوب وتكنولوجيا المعلومات، وهي تصدر بدعم من صندوق دعم البحث العلمي في الأردن. وللباحثين الحق في قراءة كامل نصوص الأوراق البحثية المنشورة في المجلة وطباعتها وتوزيعها والبحث عنها وتنزيلها وتصويرها والوصول إليها. وتسمح المجلة بالنسخ من الأوراق المنشورة، لكن مع الإشارة إلى المصدر.

الأهداف والمجال

تهدف المجلة الأردنية للحاسوب وتكنولوجيا المعلومات (JJCIT) إلى نشر آخر التطورات في شكل أوراق بحثية أصيلة وبحوث مراجعة في جميع المجالات المتعلقة بالاتصالات وهندسة الحاسوب وتكنولوجيا المعلومات وجعلها متاحة للباحثين في شتى أرجاء العالم. وتركز المجلة على موضوعات تشمل على سبيل المثال لا الحصر: هندسة الحاسوب وشبكات الاتصالات وعلوم الحاسوب ونظم المعلومات وتكنولوجيا المعلومات وتطبيقاتها.

الفهرسة

المجلة الأردنية للحاسوب وتكنولوجيا المعلومات مفهرسة في كل من:

فريق دعم هيئة التحرير

المحرر اللغوي	ادخال البيانات وسكرتير هيئة التحرير
حيدر المومني	إياد الكوز

عنوان المجلة

الموقع الإلكتروني: www.jjcit.org
البريد الإلكتروني: jjcit@psut.edu.jo
العنوان: جامعة الاميرة سمية للتكنولوجيا، شارع خليل السكاكت، الجبيهة، عمان، الأردن.
صندوق بريد: 1438 عمان 11941 الأردن
هاتف: +962-6-5359949
فاكس: +962-6-7295534

جامعة الأميرة سميّة للتكنولوجيا
Princess Sumaya University for Technology

صندوق دعم البحث العلمي والابتكار
Scientific Research and Innovation Support Fund

المجلة الأردنية للحاسوب وتكنولوجيا المعلومات

ISSN 2415 - 1076 (Online)
ISSN 2413 - 9351 (Print)

حزيران ٢٠٢٣ المجلد ٩ العدد ٢

JJCIT

www.jjcit.org

jjcit@psut.edu.jo

مجلة علمية عالمية متخصصة تصدر
بدعم من صندوق دعم البحث العلمي والابتكار