scientific reports

Check for updates

OPEN

A bioinspired neural architecture search based convolutional neural network for breast cancer detection using histopathology images

Olaide N. Oyelade✉ & Absalom E. Ezugwu✉

The design of neural architecture to address the challenge of detecting abnormalities in histopathology images can leverage the gains made in the field of neural architecture search (NAS). The NAS model consists of a search space, search strategy and evaluation strategy. The approach supports the automation of deep learning (DL) based networks such as convolutional neural networks (CNN). Automating the process of CNN architecture engineering using this approach allows for finding the best performing network for learning classification problems in specific domains and datasets. However, the engineering process of NAS is often limited by the potential solutions in search space and the search strategy. This problem often narrows the possibility of obtaining best performing networks for challenging tasks such as the classification of breast cancer in digital histopathological samples. This study proposes a NAS model with a novel search space initialization algorithm and a new search strategy. We designed a block-based stochastic categorical-to-binary (BSCB) algorithm for generating potential CNN solutions into the search space. Also, we applied and investigated the performance of a new bioinspired optimization algorithm, namely the Ebola optimization search algorithm (EOSA), for the search strategy. The evaluation strategy was achieved through computation of loss function, architectural latency and accuracy. The results obtained using images from the BACH and BreakHis databases showed that our approach obtained best performing architectures with the top-5 of the architectures yielding a significant detection rate. The top-1 CNN architecture demonstrated a state-of-the-art performance of base on classification accuracy. The NAS strategy applied in this study and the resulting candidate architecture provides researchers with the most appropriate or suitable network configuration for using digital histopathology.

Deep learning (DL) models represent a family of machine learning algorithms that assign the task of feature extraction and classification to the machine, thereby eliminating semi-autonomous feature extraction. Although the application of the feature extracted may not only be applied to image classification tasks, the DL models have achieved an impressive performance in image classification[1]. Nevertheless, most of the outstanding performances recorded by DL models were largely dependent on handcrafted neural networks requiring some human expertise and domain-specific knowledge. This limited the possibility of designing best-performing networks for application to new domains because amateurs would have to rely on pre-trained DL models, an approach referred to as transfer learning. In addition to that challenge, a significant effort is required to manually design deep neural network architecture as it is a laborious task, often limiting the exploration of network search spaces. The reliance on human expertise in achieving state-of-the-art architectures resulting from this manual approach is due to the use of manual backbone architectures or micro building blocks[2]. A new research field, namely neural architecture search (NAS), aimed at using reinforcement learning (RL) or optimization algorithms to automate the design of DL architecture, has been proposed[3]. The NAS technique allows for the design of high-performing models by using search strategy based on RL or optimization algorithms to search and design neural architectures

School of Mathematics, Statistics, and Computer Science, University of KwaZulu-Natal, King Edward Avenue, Pietermaritzburg Campus, Pietermaritzburg, KwaZulu-Natal 3201, South Africa. ✉email: OyeladeO@ukzn.ac.za; EzugwuA@ukzn.ac.za

iteratively. Initial candidate solutions (neural architectures) are generated based on a constrained formal definition of a search space allowing the search strategy to apply an evaluation function in realigning the networks during iteration. A search space, search strategy, and evaluation function are the three components of a NAS model that allows for automating neural architecture engineering. Studies have shown that DL architectures, convolutional neural networks (CNN) specifically engineered using these approaches, have outperformed the handcrafted architectures applied to some problems[4–6]. The NAS methods allow for obtaining the best performing CNN design suitable for a classification or learning problem for which the model is trained.

There is an overlap between the application of optimization algorithms in the tuning of hyperparameters and the use of optimization algorithms in automating neural architecture design. The former, referred to as hyperparameter optimization, aims to tune the hyperparameter of the already designed neural network. In contrast, the latter describes NAS, which embodies search space, search strategy and evaluation operations. In building search space for NAS, sequential layer-wise, cell-based, hierarchical structure, and memory-bank representations have been applied in literature[7]. These representations generated a population of networks that NAS-based search (optimization) algorithms sample to obtain the best performing network. The most frequently used algorithms for the search process in NAS are random search (RS), reinforcement learning (RL), evolutionary algorithms (EA), progressive decision process (PDP), and gradient descent (GD). The EA approach supports neural network topologies' evolvement using an algorithm such as the genetic algorithm (GA). The EA approach mirrors the novel Ebola virus disease propagation optimization model proposed in this study. To reward the outcome of the search algorithm, evaluation strategies are employed as feedback to the search algorithm for improving its task of outputting high-performance candidate architecture. Strategies such as the full training of all candidate solutions (networks) from scratch; training with a smaller dataset with fewer iterations (proxy evaluation); weight sharing among semblance networks; and one-shot architecture with sharing of weight parameters[8] have been widely used for evaluating the performance of potential solutions.

On the other hand, digital histopathology images are digitized images curated from the examination of biopsy samples on a microscopic slide for detecting cancer growth, a process known as histopathology. These digital histopathology images present a difficult deep learning problem compared to digital mammograms images[9]. The latter category of images often captures a case-based representation using an image sample, while in the former, a patient case is represented by large sets of images resulting from different observations of biopsy situations. Additionally, a list of subtle signs of malignancy is required to be checked to rule out benign cases in histopathology images. For instance, detecting the presence of disruption of basement membranes, marked cellular atypia, metastasize, and mitosis is an important indicator of breast cancer in histopathology images. Moreover, pathologists are expected to apply their years of experience to observe these images, classifying them as normal tissue, benign tissue, in situ carcinoma, and invasive carcinoma. Classification of these tissues often presents a complex task owning to background structures and heterogeneity in such images[10]. However, a gold standard for detecting breast cancer is the use of histopathology images above mammography images[11]. Ensuring the use of this standard will help improve the detection of breast cancer which accounts for about 32% of all cancer cases[12]. Finding an optimal neural architecture for this learning problem often proves difficult and daunting even for those with expertise in neural network design. The application of the NAS approach in designing the best performing neural architecture for this task remains promising.

However, finding candidate neural architecture to address the learning process that uses histopathology images and generates an efficient potential search space is extremely challenging. It is argued that the efficiency of a search space determines the quality of neural architectures that a NAS model can output[13]. Also, the limitations often placed on the size of this search space have mostly inhibited the upper bound of the optimal neural architectures[14]. This echoes the concern of Garg et al. [2] which noted that current NAS methods depend heavily on manual efforts in the search space design and are still being prototyped after the approach used in building models before the advent of the NAS[2]. In addition, manually tweaking network configuration and hyperparameters is time-consuming and challenging. Applying an optimization algorithm to fine-tune hyperparameters alone might not be sufficient, hence the need to improve the search strategy of NAS by designing a complete CNN architecture using the enhanced NAS model. Besides these issues of search space and search strategy and the viability of NAS notwithstanding, we found no study investigating the use of NAS models to the task of breast cancer histopathology dataset except those using a manual approach[15–26]. Considering the great benefit the neural network architecture holds in detecting and staging breast cancer using histopathology images[27], we seek to address the research question: Is it possible to generate new state-of-the-art CNN architecture using NAS model, driven by biology-based optimization strategy, for solving classification problem on histopathology images?

This study proposes a new NAS model to generate candidate CNN architecture for detecting breast cancer using histopathology images to address the aforementioned problems. We designed a novel block-based stochastic categorical-to-binary (BSCB) algorithm for generating and encoding CNN architectures in the search space. Also, we investigated the performance of our recently proposed optimization algorithm[28], namely, the Ebola optimization search algorithm (EOSA) for the search strategy as compared to other existing metaheuristic optimization approaches. The study's novelty involves designing a new NAS model, the BSCB algorithm, and the enhancement of EOSA to support the formalization of solutions as CNN architectures. Secondly, this paper represents the first study to have applied the NAS model to the complex problem of classifying digital histopathology images for the detection of breast cancer. Moreover, the study aims to obtain the best performing CNN architecture to improve classification accuracy and reduce breast cancer false-positive rates in digital histopathology images.

The main contributions of this study are elaborated as follows:

i. Propose a new block-based stochastic categorical-to-binary (BSCB) algorithm for generating initial solutions and an encoding scheme for formalizing the solutions as neural networks.
ii. Propose the application of a novel bio-inspired metaheuristic algorithm (EOSA) to perform the task of adaptive search strategy for best performing neural architectures.
iii. Implement efficient image preprocessing methods using a normalization algorithm before applying the images for training.
iv. Evaluate the proposed approach by performing a wide range of extensive experiments and comparisons with existing state-of-the-art CNN architectures and other relevant recent studies dealing with the detection of breast cancer using histopathology images.

The remaining sections of the paper are organized as follows: Sect. 2 presents an overview of the Ebola Optimization Search Algorithm (EOSA) and related studies in NAS; Sect. 3 presents the methodology applied in this study; Sect. 4 presents the parameter configuration and datasets for the experimentation; Sect. 5 presents the results obtained and discussion on findings; and Sect. 6 presents the conclusion on the relevance of the study.

Overview of EOSA and review of related studies

This section presents an overview of the optimization algorithm proposed for the NAS search strategy phase as applied in this study. The mathematical model and a summary of the Ebola optimization search algorithm (EOSA) procedure are provided in this section to conceptualise its initialization, exploitation, and exploration mechanisms. In addition, we present a review of studies focused on automation of neural network architecture design, emphasising those approaches aimed at image classification.

Mathematical model of EOSA. Oyelade and Ezugwu proposed a novel nature-inspired metaheuristic algorithm called Ebola Optimization Search Algorithm (EOSA) which is based on the propagation model of Ebola virus disease[28]. The formalization of the EOSA algorithm is achieved in the following procedure:

1. Initialize all vector and scalar quantities which are individuals and parameters. Individuals in the sets: Susceptible (S), Infected (I), Recovered (R), Dead (D), Vaccinated (V), Hospitalized (H), and Quarantine (Q) with their initial values.
2. Randomly generate the index case (I_1) from susceptible individuals.
3. Set the index case as the global best and current best, and compute the fitness value of the index case.
4. While the number of iterations is not exhausted and there exists at least an infected individual, then
    a. For each susceptible individual, generate and update their position based on their displacement. Note that the further an infected case is displaced, the more the infection number, so short displacement describes exploitation, otherwise exploration.
        i. Generate newly infected individuals (nI) base on (a).
        ii. Add the newly generated cases to I
    b. Compute the number of individuals to be added to H, D, R, B, V, and Q using their respective rates based on the size of I
    c. Update S and I base on nI.
    d. Select the current best from I and compare it with the global best.
    e. If the condition for termination is not satisfied, go back to step 6.
5. Return global best solution and all solutions.

The mathematical model of the procedure above follows: update of Susceptible (S), Infected (I), Hospitalized (H), Exposed (E), Vaccinated (V), Recovered (R), Funeral (F), Quarantine (Q), and Dead (D) is governed by a system of ordinary differential equations derived based on those in[29,30]. Differential calculus is a branch of calculus which in turn is a branch in mathematics. The former deals with the rate of change of one quantity with respect to another, while the latter deals with finding different properties of integrals and derivatives. In our case, the application of differential calculus intends to obtain the rates of change of quantities S, I, H, R, V D, and Q with respect to time $\boldsymbol{t}$. Hence, the Eqs. (1, 2, 3, 4, 5, 6, and 7) for S, I, H, R, V D, and Q respectively as follows:

$$\frac{\partial S(t)}{\partial t} = \pi - (\beta_1 \mathrm{I} + \beta_3 \mathrm{D} + \beta_4 \mathrm{R} + \beta_2 (\mathrm{PE})\eta)S - (\tau S + \Gamma I) \tag{1}$$

$$\frac{\partial I(t)}{\partial t} = (\beta_1 \mathrm{I} + \beta_3 \mathrm{D} + \beta_4 \mathrm{R} + \beta_2 (\mathrm{PE})\lambda)S - (\Gamma + \gamma)I - (\tau)\mathrm{S} \tag{2}$$

$$\frac{\partial H(t)}{\partial t} = \alpha \mathrm{I} - (\gamma + \varpi)\mathrm{H} \tag{3}$$

Symbols	Descriptions
π	Recruitment rate of susceptible human individuals
Ɲ	Decay rate of Ebola virus in the environment
A	Rate of hospitalization of infected individuals
Γ	Disease-induced death rate of human individuals
β_1	Contact rate of infectious human individuals
β_2	Contact rate of pathogen individuals/environment
β_3	Contact rate of deceased human individuals
β_4	Contact rate of recovered human individuals
γ	Recovery rate of human individuals
τ	Natural death rate of human individuals
δ	Rate of burial of deceased human individuals
ϑ	Rate of vaccination of individuals
ϖ	Rate of response to hospital treatment
μ	Rate of response to vaccination
Ξ	Rate of quarantine of infected individuals

Table 1. A description of notation and coefficients used in Eqs. (1)–(7).

$$\frac{\partial R(t)}{\partial t} = \gamma I - \Gamma R \tag{4}$$

$$\frac{\partial V(t)}{\partial t} = \gamma I - (\mu + \vartheta)\mathrm{V} \tag{5}$$

$$\frac{\partial D(t)}{\partial t} = (\tau \mathrm{S} + \Gamma \mathrm{I}) - \delta D \tag{6}$$

$$\frac{\partial Q(t)}{\partial t} = (\pi \mathrm{I} - (\gamma \mathrm{R} + \Gamma \mathrm{D})) - \xi Q \tag{7}$$

Table 1 presents the definition of the parameters and symbols used in the EOSA model design for the proposed bio-inspired neural architecture search.

In Table 1, we summarise the notations or coefficients used in Eqs. (1)–(7). This study proposes to adapt the EOSA algorithm to searching neural network architectures for improved classification tasks. Meanwhile, we review related studies to investigate the approach adopted for the optimization stage of the search strategy of recent NAS models. The following sub-section reveals our findings from this review.

Review of related studies on neural architecture search. Neural architecture search (NAS) models consist of search space, search strategy and evaluation strategy. Several studies in the literature have demonstrated variant techniques for formulating each of the components of NAS. In this section, we present the review in chronological order to understand research trends in the field.

Cortes et al.[31] proposed a NAS framework, namely AdaNet, which applied adaptive structural learning technique for the search strategy. The learning strategy utilized a data-dependent generalization method which successfully learnt both the structure of the network and its weights automatically to yield an optimal network structure. In a related study, Negrinho and Gordon[32] applied modular language techniques for the design of the search space for their NAS framework. The technique allows for populating the complex search spaces with representations of CNN architectures and their hyperparameters. Experimentation was done using three search algorithms, namely random search, Monte Carlo tree search (MCTS), and sequential model-based optimization (SMBO) over the search space. Garg et al.[2] proposed an approach called ReNAS, which represents the search space for architectures as a direct acyclic graph (DAG), which consists of basic operations. The resulting graph was then mapped to a neural network to search for candidate solutions using a differentiable architecture search approach. The results obtained in the study showed that although the approach outperformed handcrafted architectures, itcould not, however, achieve superiority over state-of-the-art NAS methods but was competitive in performance.

Wang et al.[33] addressed the shortcomings of using only the Hyperband algorithm for searching optimal neural architecture. As a result, the Hyperband algorithm and Bayesian optimization technique were hybridized to design the search strategy of NAS. The hybridization aims to build a memory to recall previous configurations when sampling the next trial configuration in searching for the optimal CNN configuration. In another study focused on improving the NAS search strategy, Huang et al.[34] proposed using a greedy technique to enhance the neural architecture search. The resulting GNAS framework was applied to address the problem of finding optimal CNN architecture for extracting features from images by exploiting an efficient greedy search approach. The greedy technique achieves its search strategy by splitting a bigger neural architecture into smaller versions

optimized in a stepwise manner. The GNAS automatically discovered optimal tree-like CNN architecture for learning and extraction of multi-attribute. Using another approach for search strategy, Weng et al.[35] applied differential architecture (DARTs) search method to design CNN architecture. The resulting search strategy was built into a convolutional neural search architecture (CNAS) framework. The proposed DART-based search strategy finds architectures from a search space utilizing both shuffle operation and squeeze-and-excitation. We, however, found their approach to be sub-optimal when compared with the use of evolutionary-based optimization techniques, as seen in the works of Erivaldo et al.[36] and Liu et al.[37]. The two studies approached the design of search strategy of their NAS framework using particle swarm optimization (PSO) and Genetic Algorithm (GA)-based, respectively. The optimization mechanism of PSO was employed for a direct encoding strategy. In contrast, the optimization task of GA was supported by an experience-based greedy exploration strategy and transfer learning techniques.

Considering the cardinal role of the search strategy in the NAS model, we reviewed recent studies to observe the approaches applied. For instance, Krishna et al.[38] proposed two techniques, namely reinforcement learning strategy and attention-based mechanism with simplified transformer block method for the search strategy of NAS and improving hyperparameter candidate neural architecture, respectively. The study uses a two-stream attention-based mechanism to model hyper-parameter dependencies and a simplified transformer block. In contrast, the second method aimed to remove layer normalization, the former models the policy network for searching the space. The authors reported that the performance of their method surpasses most methods in NAS-Bench-101 benchmarked models.

A similarity to the works of Erivaldo et al.[36] and Liu et al.[37] is seen in the study by Calisto and Lai-Yuen[39]. This study applied evolutionary algorithms to search for neural networks in the search space to discover high-performing and efficient neural architecture. The optimization algorithm is rewarded if it can discover architectures with improved classification accuracy and reduced hyperparameters size. The resulting architecture which comprises a self-adaptive 2D and 3D ensemble of FCNs used for 3D medical image segmentation was code-named AdaEn-Net. In a similar approach, Chen and Li[40] proposed using an evolutionary algorithm for the search strategy of a search space. The search space is composed of a major super-network whose weight is shared with sub-network architectures in obtaining an optimal candidate network. This optimal architecture is derived from a collection of most performing or excellent architectures by examining the commonalities of these architectures[40].

Wang et al.[41] proposed what is referred to as DC-NAS, which was derived from a divide-and-conquer (DC) approach to the NAS problem. The study applied the DC method to obtain the best performing sub-graphs of potential complete network architecture. Meanwhile, sub-graphs are first clustered based on their similarity so that best performing sub-graphs in a cluster are merged with other optimal sub-graphs in related clusters to form a new architecture. The resulting optimal sub-graphs combine to form a new neural architecture that is effective and efficient. On the issue of NAS search space, we found the work of Cassimon et al.[42] which uses a cell-based representation approach for search space very interesting. The method adapted reinforcement learning to optimise cells for detecting the two types of networks, namely Recurrent Neural Network (RNN) and Convolutional Neural Network (CNN). The study considered a network optimal if it could successfully predict spoken words and classify RGB images on an embedded device. The study's main contribution is the proposal of an efficient neural architecture search (ENAS) fitted for embedded devices and with improving performance network architecture. In a similar approach, though using a different NAS-based method, Fan et al.[43] jointly searched for operations like LSTM and CNN from a search space using gradient-based neural architecture optimization (NAO) technique. The search space combined two heterogeneous spaces with network operations space and dropout space. The former consists of basic network operations, while the latter space consists of dropout rates. The resulting networks are those whose architectures and hyperparameters are well optimized for neural machine translation (NMT).

An improvement to the work of Cortes et al.[31] is reported in the study of Dai et al.[44]. The authors employed a NAS framework driven by the use of the AdaNet technique. The focus of the improvement is achieving a better search space and search strategies for obtaining the optimal structure of the CNN architecture and optimising the weights of the CNN architecture. AdaNet utilizes a simple linear model representing the search space and then gradually augments the network with more neurons and additional layers until an optimal network architecture is obtained. Each step in building the resulting architecture requires that a Gradient-decent-based optimization method using Momentum be applied. The study's outcome is a CNN architecture used for three classes (3-hinge gyral vs 2-hinge gyral vs sulcal) classification in f-MRI signals classification problem. In a different approach, Gheshlaghi et al.[45] proposed a NAS model by applying the binary gate method to search strategy through stacking of cells upon cells of sub-networks using primitive operations. These cells consist of Down-Sampling Cell (DownSC) and Up-Sampling Cell (UpSC) whose designs are automated into the NAS process. The resulting optimal neural network architecture is expected to outperform handcrafted architecture, which is purposed for the same task of retinal layer segmentation in Optical Coherence Tomography (OCT) scans.

Chen et al.[46] proposed a single-stage NAS framework named you only search once (YOSO) for automating the process of finding the optimal deep neural network (DNN), which are used for co-designing of software/hardware. The need for co-design of software and hardware by the resulting DNN further swelled the volume of the search space with hyperparameters of the DNN and hardware design parameters. The study applied reinforcement learning with LSTM for the search strategy. The resulting NAS framework applied a multi-objective reward system aimed at maximizing accuracy, power, and QoS. Meanwhile, several DNNs are generated from basic operations to formulate the search space. An interesting aspect of the study is the use of an auxiliary HyperNet that voids the training of candidate DNNs before applying resulting weights for evaluating their performances in terms of accuracy. In another study, Guo et al.[47], proposed a variant of NAS capable of generating neural architectures using an inference model. The neural architecture generating (NAG) model learns from a

References	Search space	NAS search and optimization method	Evaluation strategy
Cortes et al.[31]	Simple network grown incrementally	Adaptive structural learning (AdaNet)	Binary classification accuracy
Negrinho and Gordon[32]	Tree-structured search space	MCTS and SMBO	Training and validation
Wang et al.[33]	AlexNet and LeNet hyperparameters	Hyperband algorithm and Bayesian optimization	Classification accuracy
Huang et al.[34]	Global architecture	Greedy search approach	Mean prediction accuracy
Weng et al.[35]	Primitive operations and intermediate nodes	DARTs	Measuring loss and accuracy
Erivaldo et al.[36]	Random CNN architectures initialization	PSO search strategy	Crossentropy loss and velocity computation
Liu et al.[37]	Residual blocks	GA search strategy	Fitness function for image quality measurement
Garg et al.[2]	Hierarchical structure using DAG	Differentiable architecture search	Surrogate approach
Krishna et al.[38]	NASBench-101 search	Reinforcement learning	Actor-critic algorithms
Calisto and Yeun[39]	Basic operations and corresponding hyperparameters	Evolutionary algorithm	Classification accuracy and hyperparameter reduction
Wang et al.[41]	Cell-based representation	Divide-and-conquer (DC) approach	k-means base clustered evaluation
Cassimon et al.[42]	Cell-based representation	Reinforcement learning	Multi-objective evaluation
Fan et al.[43]	Hybrids of cell-based representation	Gradient-decent-based neural architecture optimization (NAO)	Minimization of regression and reconstruction losses, and dropout rates
Dai et al.[44]	AdaNet: Hierarchical structure	Gradient-decent-based using Momentum	Maximizing classification accuracy
Gheshlaghi et al.[45]	Cell-based representation of primitive operations	Gradient-based approach for binary gate method	Training from scratch
Chen et al.[46]	Basic operations	Reinforcement learning using LSTM	HyperNet based accuracy evaluator and hardware performance predictor
Chen and Li[40]	Weight sharing strategy from a major supernetwork	Evolution algorithm method	Commonalities among best performing architectures
Guo et al.[47]	Basic operations	Inference model learning from Pareto frontier parameters	Model performance and computational cost
Zhang et al.[48]	Basic operations	Reinforcement learning and evolutionary algorithm	Minimization of loss function
Hu et al.[49]		Attention-guided differentiable mechanism	Classification accuracy
Xu et al.[50]	Super-network	Partially connected DARTS	Error rates of searched networks
Ru et al.[51]	Graph-like search spaces	Bayesian optimization	Performance evaluation of motifs
Fu et al.[52]	Basic operations	Reinforcement learning and LSTM	Quantified-parameter evaluation mechanism
Lin et al.[53]	A single randomly initialized network	Inference budgets model	Zero-shot approach
Liu et al.[54]	A SuperNet	Particle swarm optimization	
Liang et al.[55]	DAG-based FPNs	One-short search strategy	Detection accuracy

Table 2. Summary of the reviewed studies.

Pareto frontier, which guides optimal architectures based on the given budget for the target system on which the resulting architecture is expected to be used. On the other hand, Zhang et al.[48] addressed the problem of the non-convexity of NAS through the use of an adaptive, scalable neural architecture search method (AS-NAS). The scalability of AS-NAS was achieved through a search strategy that combined a simple reinforcement learning, namely: reinforced I-Ching divination evolutionary algorithm (IDEA) and variable-architecture encoding strategy.

In a similar approach to Krishna et al.[38] and Weng et al.[35], though an improvement on the approach, He et al.[49] proposed a special kind of NAS model called attention-guided differentiable architecture search (A-DARTS), which adopts a mechanism for reducing the sensitivity of initialization of searched space. Also, Xu et al.[50] improved the efficiency and stability of searched networks using the Partially-Connected DARTS (PC-DARTS) approach. The PC-DARTS improves the search strategy by randomly selecting a small subset of channels for partial channel connection to overcome over-fitting the search networks.

Several studies have proposed new variants of the NAS model. For instance, Ru et al.[51], applied the technique of Bayesian optimization (BO) to the design of the NAS model to obtain a new model known as interpretable neural architecture search (INAS). The proposed INAS uses graph-like search spaces while combining the Weisfeiler-Lehman graph kernel with a Gaussian process surrogate with BO for the search strategy. Fu et al.[52], addressed the problem of incremental learning in the classification of images through the approach of neural architecture search for incremental learning (NASIL). This was done by using reinforcement learning, parameter-sharing mechanism, and Long Short-Term Memory (LSTM). Also, Lin et al.[53] added novelty to the approach of NAS by improving the evaluation strategy, which replaces an accuracy predictor with zero-shot in the ranking of searched architectures. The resulting value from the zero-shot operation is maximized using an inference budgets model called Zen-NAS.

On the other hand, Liu et al.[54], applied an evolutionary method to optimise weight-sharing parameters when searching for optimal neural architectures. This search strategy, called continuous particle swarm optimization (CPSO-Net), computes the gradient of networks resulting from shared parameters of candidate operations to obtain candidate architecture. Lastly, Liang et al.[55], applied a variant of NAS to generate optimal feature pyramid

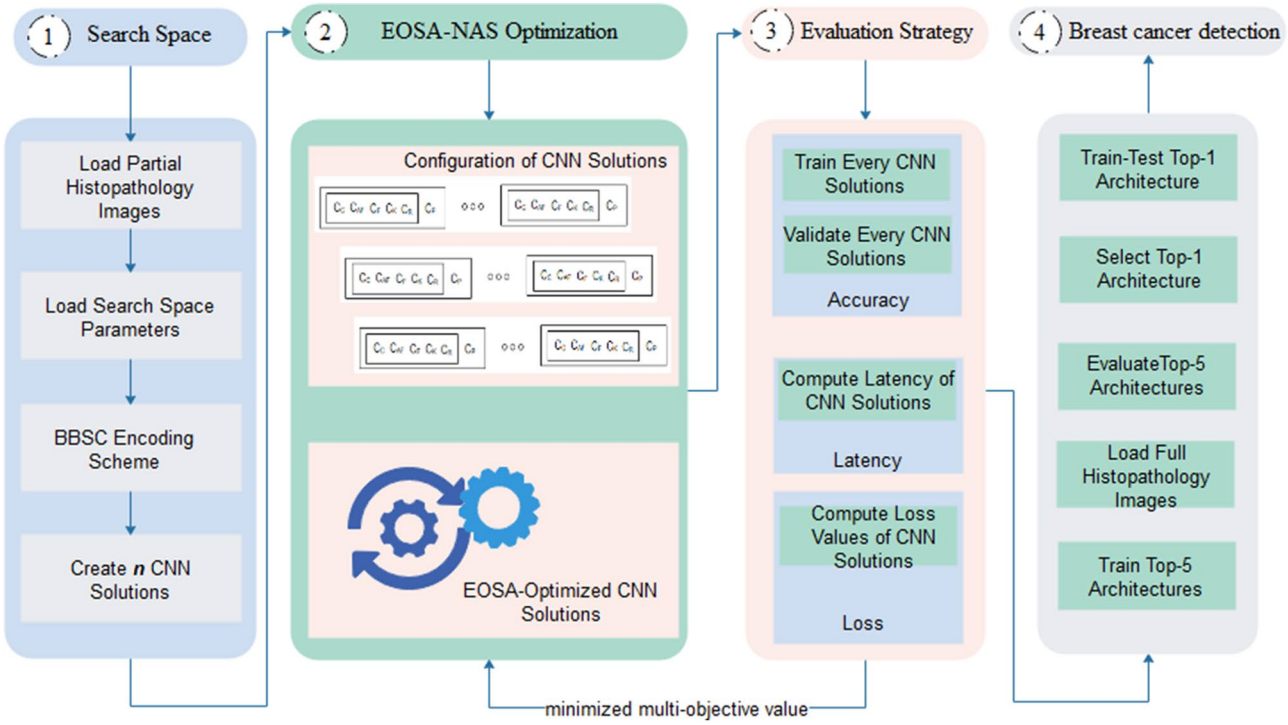

Figure 1. The proposed EOSA-NAS model consisting of four components: the search space, EOSA-NAS search strategy, evaluation strategy and the breast cancer detection module using the top-5 and top-1 CNN architectures.

networks (FPNs). The resulting One-Shot Path Aggregation Network Architecture Search (OPANAS) approach uses a one-short strategy for searching for candidate FPNs that are drawn from DAG-based FPNs search space.

The review presented in this section, and summarized in Table 2, demonstrates that different methods have been applied to improve the components of the NAS model. The components which have received more research attention are the search space encoding strategy and the search strategy. Our findings revealed that most of the studies had applied reinforcement learning techniques, evolutionary and use of metaheuristic algorithms. We discovered that the most promising approach is seen in studies that used evolutionary or computational biology methods for search strategy. Hence, this study aims to improve the NAS search strategy by using EOSA, a bio-inspired optimization algorithm, to generate optimal neural architecture in classifying histopathological images to detect breast cancer. In addition, a novel search space encoding algorithm is proposed to allow for good coverage of the potential CNN architectures. The following section details the search space and search strategy proposed in this paper.

Methods

This section is focused on the design methods of the three (3) components of the neural architecture search (NAS). First, we present our proposed NAS model which demonstrates the interoperability of the 3 components. Secondly, the design of a novel search space encoding algorithm that defines a population of initial CNN solutions is discussed. Thirdly, the neural search strategy which is based on the Ebola optimization search algorithm (EOSA) method, is presented. Fourthly, we demonstrate how the multi-objective evaluation strategy is computed and how its results are passed back to the search strategy for refinement purposes.

The neural architecture search (NAS) model. This sub-section gives a high-level overview of the proposed NAS model, which shows basic operations for the search space, the search strategy, and the performance evaluation strategy. The overall aim of the model is to guide the selection process of best performance arbitrary CNN architecture, from the search space, for solving the classification problem.

The proposed NAS model is presented in Fig. 1 and shows the three major components of NAS. In addition, a mechanism for evaluating the best performing architecture resulting from the search strategy is also provided. The following is a brief discussion of each component:

1. **Search space** The proposed model shows how an encoding scheme is used to generate ***n*** potential initial solutions, representing CNN architectures. The proposed encoding scheme aims to create a pool of potential configurations of basic operations and hyperparameters of n CNN architectures that are capable of yielding the best performance.
2. **Search strategy** A bioinspired EOSA-based search strategy iteratively optimizes each CNN solution from a pool of potential solutions located in the search space. For each iteration, the configuration of each CNN

(Min, max) no. of blocks in BSCBE	Block category	CNN Hyperparameters	Notational Representation	Lower Bound	Upper Bound
(1, 1)	General hyperparameter block	Batch size/mode	G_b	0	2
		Learning rate	G_α	0	8
		Optimization algorithm	G_o	0	7
		Epoch	G_e	1	2
(0,1)	Input-Zeropadding block	Whether to zero-pad inputs or not	Z_α	0	1
(1, N)	Convolutional layer block	Number of conv-pool blocks	C_L	1	6
		Number of convolutional blocks in C_l	C_C	0	2
		Choice of activation function per convolutional layer	C_{AF}	0	2
		Number of kernel	C_K	3	10
		Kernel size	C_F	0	10
		Pool size	C_{PS}	0	2
		Pool operation type	C_{PT}	0	1
		Weight regularization operation	C_R	0	2
(1, 2)	Fully connected block	Number of dense (fully-connected) layers	F_L	0	1
		Activation function for the layer	F_{AF}	0	2
		Use of dropout layer	F_D	2.0	2.2
		Weight regularization operation	F_R	0	2
(1, 1)	Loss function block	Choice of loss function	LF_L	0	2

Table 3. Categorization of parameters based on the block encoding scheme for representation of the hyperparameters of convolutional neural network.

solution is improved towards learning the classification problem. This is attained in conjunction with a mechanism for evaluating performance based on high accuracy, reduced loss value, and low latency.

3. **Evaluation strategy** We designed a mechanism for measuring and estimating the performance of CNN models resulting from the optimization operations on the search space. This allows for passing a kind of reward to the search algorithm to support the process of finding a candidate CNN solution. To minimize the computational cost of the evaluation, we trained the CNN models for few epochs and computed the result of the objective functions, namely classification accuracy, loss function, and CNN architectural latency.
4. **Evaluation of top-performing CNN models** After an exhaustive optimization process, the top-5 performing CNN models are chosen from the ***n*** solutions. These top-5 are subjected to further comparative analysis to measure and discover their capability in solving the classification problem—the problem of detecting the presence of breast cancer in histopathological images. Here, the top-5 CNN architectures are subjected to full training using the complete datasets.

The following sections present detail on the design and applicability of each component in the NAS model illustrated by Fig. 1.

The search space and encoding scheme. The quality of a search space determines the performance of both the initial and candidate CNN solutions in NAS models. Also, the encoding scheme applied to a search space directly impacts the complexity of the neural search strategy. Hence, it is necessary to carefully choose the technique to represent a search space in a NAS model. In this section, we present the design of a novel search space encoding initialization and encoding scheme. The design is based on a block-based approach. Firstly, the proposed encoding strategy is designed to generate potential initial CNN solutions exhaustively. Secondly, the design also models each CNN solution in such a manner as to allow the evaluation of the multi-objective functions inexpensively. Thirdly, the scheme provides scalable and easy navigation within the search space using the search space algorithm. We propose a block-based stochastic categorical-to-binary (BSCB) encoding scheme that maps each unique parameter label to an integer value when constructing the search space.

The categorical feature or parameter is first converted into a numeric value using an ordinal encoder. This strategy allows for digitizing each convolutional operation and hyperparameters of the CNN solution, which then allows for the efficient representation in the solution space. Once the categorical transformation is achieved, we binarize the resulting integer values. Each binarized value is then bounded within its lower and upper bounds to ensure that it represents a valid CNN architecture. The encoded parameters are then used for building a multi-block-based schematic representation of a CNN model. The resulting blocks are stacked in an ordinal fashion based on the traditional approach of designing CNN architectures. A well-stacked group of blocks represents a potential CNN solution generated into the search space. The implication of this is that CNN architectures are designed on the fly with no prior handcrafted configurations.

To achieve the encoding scheme described in the previous paragraph, we provide a list of potential parameters from which blocks are encoded. These parameters, listed in Table 3, represent the convolutional operations and

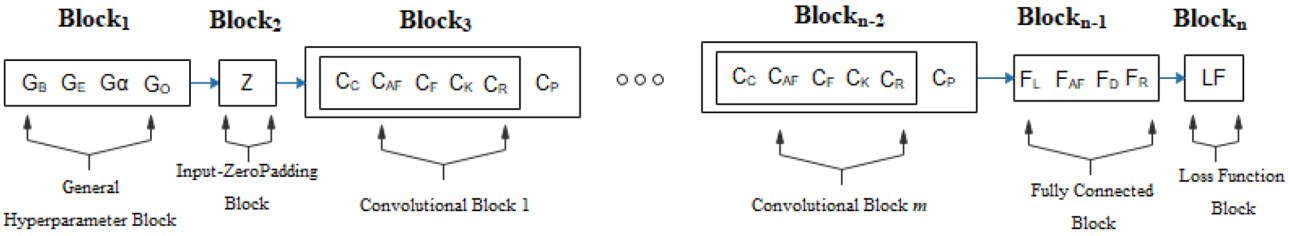

Figure 2. A generic representation of an encoded CNN architecture based on the parameters covered by the search space.

hyperparameters of a vanilla CNN model. The listing allows for a wider range of combinations of values for each parameter. This outcome is a pool of rich potential initial CNN solutions for use by the search strategy.

The list of parameters that constitutes the search space includes the batch size of samples used for input, learning rate, optimization algorithm, the number of convolutional layers, number of kernels, kernel size, the activation function of each convolutional layer, the pooling type and size, the number of dense layers, the connectivity pattern, the activation function, weight regularization techniques, and the dropout for each dense layer. To generate potential CNN solutions into the search space, the following describes how the proposed block-based encoding scheme utilizes these parameters as defined in Table 3.

First, we note that when required for generating an arbitrary CNN solution, each parameter is derived using Eq. (8). Moreover, an arbitrary CNN solution combines a number of these parameters to build its blocks:

$$P_{(c,i),j} = lb_j + rand(ub_j - lb_j) \tag{8}$$

where $P_{(c,i),j}$ represents the i^{th} parameter in the c^{th} category and the j^{th} parameter in the list of parameters (P) to be passed to the encoding algorithm. Note that for each parameter, the *ub* and *lb* represent the upper and lower bounds, respectively. A corresponding value for each parameter is computed by generating a random number, multiplying by the difference of *ub* and *lb*, and then adding it to the lower bound. Once these values for all parameters are obtained, we proceed to the block encoding for generating CNN solutions.

In Eq. (9), we show the complete encoding of a CNN solution where each $block_i$ is composed by some $P_{(c,i),j}$. This potential auto-generated CNN solution consists of blocks of different structures arranged in an ordinal pattern to reflect the traditional architecture of CNN. Furthermore, to form the search space for the neural search algorithm, several of this $CNN_{solution}$ are generated and represented as seen in Eq. (10), where $CNN_{search\text{-}space}$ represents a collection of CNN solutions in the search space

$$CNN_{solution} = \{block_{gy}, block_{iz}, block_{cl}1, \ldots block_{cl}n-1, block_{cl}n, block_{fc}, block_{lf}\} \tag{9}$$

$$CNN_{search-space} = \{CNN_{solution_1}, CNN_{solution_2}, \ldots CNN_{solution_n-1}, CNN_{solution_n}\} \tag{10}$$

A predefined number of blocks to be generated for each category are defined in Table 3. The algorithm iterates over each category and digitalizes its parameter, and computes the corresponding value of the binarized parameter mapped to it so that each category translates to a block. Note that the structures of an arbitrary $block_i$ in $solution_x$ might not have the same parameter value as another in the same solution. This representation allows for a radial coverage of the potential solution space to allow for an effective and efficient search space.

The binarized parameter and its corresponding category are denoted by vector v_b such that $v_b \in \{0,1\}^n$ as detailed in Eqs. (11) or (12).

$$\left\{ v_b \in \{0,1\}^n : \sum_{i=1}^{n} v_i = 1 \right\} \tag{11}$$

$$v_b = 1_\rho(param) = \begin{cases} 0, & x \notin \rho \\ 1, & x \in \rho \end{cases} \tag{12}$$

A general structural representation of a CNN architecture using the encoding scheme is shown in Fig. 2. While blocks 1, 2, … n-1, n form the basic structural representation for each potential CNN solution, we note that each solution could represent a more complex structure than Fig. 2. Algorithm 1 presents a pseudocode of the technique for generating all CNN solutions into the search space.

Algorithm1: BSCB for generating search space

```
Result: Generated search space of CNNs
Input: catg, params, N
Output: sp
sp ← ∅
while N > 0 do
    solution ← ∅
    for i ← 1 to len(catg) do
        blks ← ∅
        min, max = untar(catg[i])
        c = categorical(catg[i])
        bc = catg[i]b
        for j ← min to max do
            blk ← ∅
            while len(params[catg[i]]) > 0 do
                pr, lb, ub ← untar(params)
                p = categorical(pr)
                bp = catg[i]b
                p = bc ⊕ bp
                pv = lb + rand(0, 1) × (ub − lb)
                blk ← tar(p, pv)
            end
            blks ← blk
        end
        solution ← blks
    end
    sp ← solution
    N − = 1
end
return sp
```

The algorithm generates n solutions for the search space by using the block-based encoding scheme. This is done by first carrying out category-based parameter extraction so that extracted parameters are then digitized. Meanwhile, the equivalent value for each parameter is computed before mapping them in a corresponding parameter-category association. Finally, blocks are formed from such mappings, which are then translated and chained into potential CNN solutions. In the next section, the application of the neural search algorithm to the search space is described in detail.

Bioinspired EOSA search and optimization strategy. The search strategy proposed in this study is based on the Ebola optimization search algorithm (EOSA). This allows for widening the search operation in the direction of both exploration and exploitation. The outcome of the search process often yields the best performing CNN architecture for the detection and classification of breast cancer using histopathological images. The resulting CNN search algorithm is henceforth referred to as the EOSA-NAS algorithm. The EOSA-NAS algorithm explores the search space to obtain a candidate CNN architecture suitable for addressing the classification problem. The approach ensures that irrelevant candidate architectures are lined behind the potential architectures.

The search algorithm first initializes the compartments to empty sets: Susceptible (S), Exposed (E), Infected (I), Hospitalized (H), Recovered (R), Vaccinated (V), and Quarantine (Q). Thereafter, a variable is created to keep track of the top performing architectures of each iteration. The set S contains all potential solutions (CNN architectures) in the search space, ranked according to their performance based on evaluation strategy so that the most performing architectures are at the head of the queue. The CNN architecture or solution at index 0 is assigned to the exposed E set and eventually to the I set. The position of each solution is updated using (13).

$$mI_i^{t+1} = mI_i^t + \rho M(s_i) \tag{13}$$

where ρ represents the scale factor of displacement such that individuals mI_i^t and mI_i^{t+1} represents the updated or current position and previous position at time t and t + 1, respectively. *M(I)* is the movement rate made by individual solutions, as shown in Eqs. (14) and (15).

$$M(I) = srate * rand(0,1) + M(Ind_{best}) \tag{14}$$

$$M(s) = lrate * rand(0,1) + M(Ind_{best}) \tag{15}$$

The search strategy is able to search within the neighborhood threshold (exploitation) using the short distance movement, *srate*. Also, the algorithm can search beyond the neighborhood threshold (exploration) using the long distance movement, *lrate*. Both *srate* and *lrate* are regulated by *neighborhood* parameters. For instance, if the computed *neighborhood* parameter is above 0.5, it is assumed the infected individual (solution) has moved beyond the *neighborhood*, hence the exploration phase. Otherwise, it is assumed it remains within the *neighborhood*, hence the exploitation phase. With this mechanism, candidate solutions or CNN architectures evolve and are placed in the I set for use in the next operation. The next operation mutates the configuration of the solutions or CNN architectures for improved performance. This mutation or optimization process is guided by the need for solutions to learn the classification problem. Every infection operation weakens the immunity of the individual (CNN architecture). The configurations of any CNN architecture in *I* are represented in Eq. (16); solutions (CNN architectures) which have recovered (*R*) have their immunity strengthened as shown in Eq. (17), and those dead individuals (*D*) are replaced by new solutions; individuals or solutions which were not infected are maintained in *S*.

$$NA_i = cfactor * e^{1*l} * cos(2\pi l) + NA_{best} \tag{16}$$

$$NA_i = rand * cfactor * (NA_{best} - NA_i) \tag{17}$$

where *NA* stands for neural architecture, which is the same as solutions, *cfactor* is the rate of change of the structure as determined by *neighborhood* value, and *l* is a sample drawn from a uniform distribution in the range of [−1,1]. The resulting value from the evaluation of Eqs. (16) and (17) affects the operations defined by each parameter in all blocks, as shown in Fig. 2.

Algorithm 2: Algorithm of the EOSA-NAS algorithm

```
Result: Top-best solutions
Input: epoch, psize, evdincub
Output: topSols
S, E, I, H, R, V, Q, topSols ← ∅
S ← Algorithm1(catg, params, psize)
icase ← generatedIndexCase()
gbest, cbest ← icase
while e ≤ epoch ∧ len(I) > 0 do
    Q ← rand(0, Eq.7 × I)
    fracI = I − Q
    for i ← 1 to len(fracI) do
        pos_i ← movrate()   using Eq.1
        d_i ← rand()
        if d_i > evdincub then
            neighborhood ← prob(pos_i)
            if neighborhood < 0.5 then
                tmp ← rand(0, Eq.2 × I × srate)
            end
            else
                tmp ← rand(0, Eq.2 × I × lrate)
            end
            newI+ ← tmp
        end
        I+ ← newI
    end
    h ← rand(0, Eq.3 × I), H+ ← h
    r ← rand(0, Eq.4 × I), R+ ← r
    v ← rand(0, Eq.5 × h), V + ← v
    d ← rand(0, Eq.6 × I), D+ ← d
    I+ ← I − add(r, d)
    S+ ← r
    S− ← d
    objfuncs = [accuracy(S), loss(S), latency(S)]
     cbest = fitness(objfuncs);
    if cbest > gbest then
        gbest = cbest
        topSols ← gbest
    end
end
return topSols
```

The procedure described by the mathematical model above is summarized in Algorithm 2. The use of back-arrow notation (←) represents storage or assignment statement, while the combined use of back-arrow and plus notations (+←) represents cumulative storage of values in a variable. This algorithm outlines the call to the initialization of the search space, the iteration through a given epoch for the evolvement of improved CNN architecture or solutions, and the application of the multi-objective function in obtaining the best solution. The last line returns a list of solutions representing CNN architecture with the top best at the head of the queue. The search strategy ensures that all potential architectures are evaluated based on three (3) objective functions that yield a one-value metric. The following section details this evaluation strategy.

Evaluation strategy. The selection of the current best, at any time t, is computed on the set of infected individuals at that time t, whereas selection of the global best is based on the best performing CNN solution at the end of the training process. Performance is measured using classification accuracy on CNN training and validation, the latency of the CNN architecture, and the loss function (categorical or sparse cross-entropy). This multi-objective approach is motivated by findings from the literature that justify the need to consider factors such as model size, latency, computational time and fast response time[56]. We motivate the need for a multi-criteria evaluation strategy considering that a single-objective focused on over-classification accuracy will be inaccurate in obtaining the best performing CNN architecture. In Eqs. (18), (19) and (20) are definitions of the

Hyperparameter	Formula	Hyperparameter search space
G_b	$2^n - 1$	[0, 1, 3]
G_α	$rand(1\|5) \cdot 10^{-n}$	[1 × 10 − 5, 5 × 10 − 5, 1 × 10 − 4, 5 × 10 − 4, 1 × 10 − 3, 5 × 10 − 3, 1 × 10 − 2, 5 × 10 − 2, 1 × 10 − 1, 5 × 10 − 1]
G_o	$O[n]$	[0 = > "SGD", 1 = > "Adam", 2 = > "RMSprop", 3 = > "Adagrad", 4 = > "Nestrov", 5 = > "Adadelta", 6 = > "Adamax", 7 = > "Momentum]
G_e	5	5
I_Z	$rand(0\|1)$	[0,1]
C_L	$2n - 1$	[1, 3, 5, 7, 9, 11]
C_C	$3 - n$	[1, 2, 3]
C_{AF}	$AF[n]$	[0 = > "ReLU", 1 = > "LeakyReLU", 2 = > "Parametric ReLU"]
C_K	2^n	[8, 16, 32, 64, 128, 256, 512, 1024]
C_F	$2 + (n - 1)$	[1, 3, 5, 7, 9, 11]
C_{PS}	$2 + n$	[2, 3, 4]
C_{PT}	n	[Max pooling, Average pooling]
C_R	n	[L1, L2, L1L2]
F_L	$1 + n$	[1, 2]
F_{AF}	$FAF(n)$	[0 = > " Softmax", 1 = > " Sigmoid "]
F_D	$\frac{1}{n}$	[0.35, 0.4, 0.45, 0.5]
F_R	n	[L1, L2, L1L2]
LF_L	n	[categorical cross-entropy, sparse cross-entropy]

Table 4. A summary of formula for computing values for hyperparameters and the corresponding search space using the proposed encoding scheme.

Symbols	Descriptions	Range
Epoch	Number of iteration for the EOSA algorithm	5
Population	Number of neural architectures in the search space	50
π	Recruitment rate of susceptible human individuals	Variable
η	Decay rate of Ebola virus in the environment	$(0, \infty)$
α	Rate of hospitalization of infected individuals	(0, 1)
Γ	Disease-induced death rate of human individuals	[0.4, 0.9]
β_1	Contact rate of infectious human individuals	Variable
β_2	Contact rate of pathogen individuals/environment	Variable
β_3	Contact rate of deceased human individuals	Variable
β_4	Contact rate of recovered human individuals	Variable
γ	Recovery rate of human individuals	(0, 1)
τ	Natural death rate of human individuals	(0, 1)
δ	Rate of burial of deceased human individuals	(0, 1)
ϑ	Rate of vaccination of individuals	(0, 1)
ω	Rate of response to hospital treatment	(0, 1)
μ	Rate response to vaccination	(0, 1)
ξ	Rate of quarantine of infected individuals	(0, 1)

Table 5. Notations and description for variables and parameters used for experimenting with EOSA optimization algorithm.

metrics applied for the multi-criteria evaluation strategy. Performance comparison for the similarity between CNN architectures is achieved using Eq. (21).

$$Accuracy = \frac{TP + TN}{(TP + TN + FP + FN)} \tag{18}$$

$$latency = time()_{after-train} - time()_{before-train} \tag{19}$$

$$loss\,function = l(w) = -\frac{1}{N}\sum_{n=1}^{N}\sum_{m=1}^{M} t_{nm} log_2 Pnm \tag{20}$$

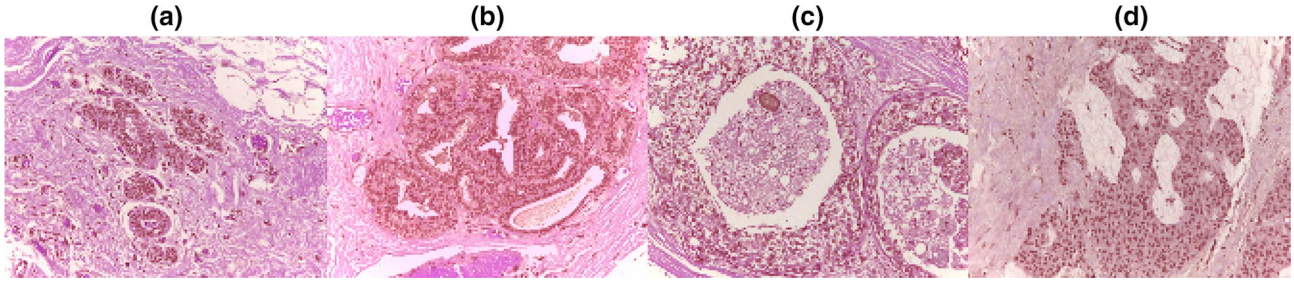

Figure 3. Sample images from the BACH datasets showing (**a**) normal (**b**) benign (**c**) in situ carcinoma and (**d**) invasive carcinoma cases.

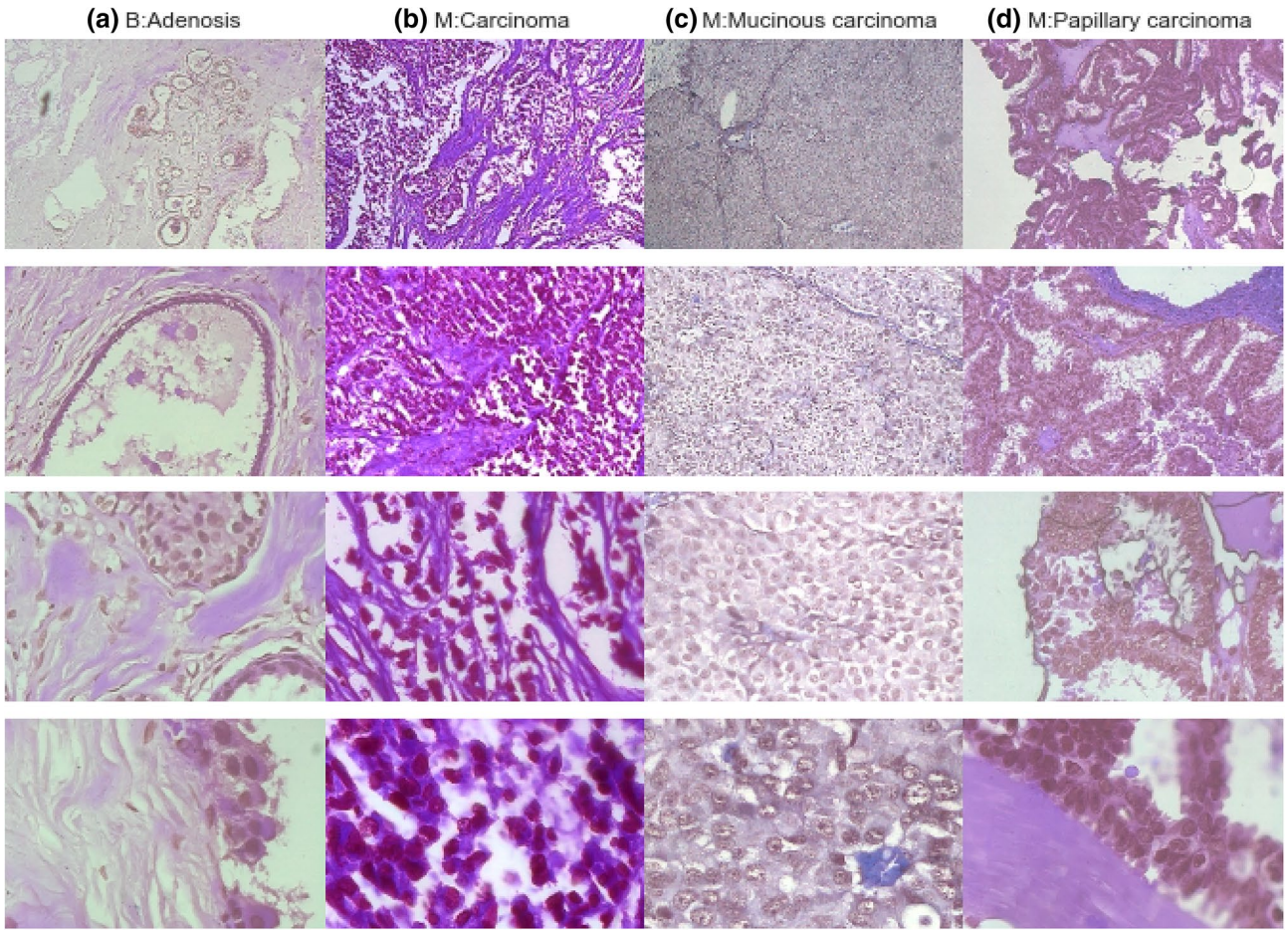

Figure 4. Sample images from the BreakHis datasets showing (**a**) adenosis, (**b**) ductal carcinoma, (**c**) mucinous carcinoma, and (**d**) papillary carcinoma malignant cases. Each column shows the magnification of samples for (**a**)–(**d**) in 40X, 100X, 200X, and 400X accordingly. The H&E stain the nuclei with a dark purple (Hematoxylin) and the cytoplasm with a light pink (Eosin).

$$reward = \frac{1}{(acc + l(w) + t)} \tag{21}$$

$$Similarity(NA_i NA_j) = \begin{cases} dist(NA_i, NA_j), & if\, NA_i < NA_j \\ 0, & otherwise \end{cases} \tag{22}$$

where NA_i represents any arbitrary neural network, and the function $Similarity(NA_i, NA_j)$ allows for comparing two neural networks in a search space.

ID	Function name	Model of the function
F1	Ackley	$f(x) = -20e^{\left(-0.2\sqrt{\frac{1}{n}\sum_1^n x_1^2}\right)} - -e^{\left(\frac{1}{n}\sum_1^n \cos(2\pi x_i)\right)} + 20 + e^{(1)}$
F2	Alpine	$f(x) = \sum_{i=1}^n \lvert x_i \sin(x_i) + 0.1x_i \rvert$
F3	Brown	$f(x) = \sum_{i=1}^{n-1} (x_i^2)^{(x_{i+1}^2+1)} + (x_{i+1}^2)^{(x_i^2+1)}$
F4	Bent Cigar	$f_{20}(x) = x_1^2 + 10^6 \sum_{i=2}^D x_i^2$
F5	Dixon and Price	$f_{18}(x) = 10^6 x_1^2 \sum_{i=2}^D x_i^2$
F6	Discus Function	$f(x) = (x_1 - 1)^2 + \sum_{i=2}^n i(2x_i^2 - x_{i-1})^2$
F7	Levy	$f_{12}(x) = \sum_{i=1}^n (x_i - 1)^2 \left[sin^2(3\pi x_{i+1})\right] + sin^2(3\pi x_1) + \lvert x_n - 1 \rvert \left[1 + sin^2(3\pi x_n)\right]$
F8	Powel	$f(x) = (x_1 + 10x_2)^2 + 5(x_3 + x_4)^2 + (x_2 - 2x_3)^4 + 10(x_1 - x_4)^4$
F9	Quartic	$f_6(x) = \sum_{i=1}^n ix_i^4$
F10	Rastrigin	$f_9(x) = \sum_{i=1}^n [x_i^2 - 10\cos(2\pi x_i) + 10]$
F11	SR-F27	Shifted and Rotated Rastrigin's Function
F12	Wavy 1	$f(x) = \sum_{i=1}^n x_i^2 + (\sum_{i=1}^n 0.5ix_i)^2 + (\sum_{i=1}^n 0.5ix_i)^4$
F13	Zakharov	$f(x) = \frac{1}{n}\sum_{i=1}^n 1 - \cos(10x_i)e^{-\frac{1}{2}x_i^2}$
F14	Salomon	$f_{19}(x) = 1 - cos\left(2\pi\sqrt{\sum_{i=1}^n x_i^2}\right) + 0.1\sqrt{\sum_{i=1}^n x_i^2}$
F15	Weierstrass Function	$f(x) = \sum_{i=1}^D (\sum_{i=0}^{20} [0.5^k \cos(2\pi .3^k (x_i + 0.5))])$

Table 6. Standard and CEC benchmark functions used for the experimentation in evaluating the performances of EOSA, ABC, WOA, PSO and GA.

Algorithm 3: Multi-objective evaluator

Result: Returns the performances (p) for evaluated CNN architectures
Input: Training data X, target Y, neural architectures (NA), epoch, dts
Output: reward

```
p ← ∅
for i ← 0 to len(NA) do
    model = NA[i]
    accuracy = 0.0
    loss = 0.0
    if dts = 0 then
        X̃, Ỹ ← X, Y
    end
    else
        if dts = 1 then
            X̃, Ỹ ← rand(X, Y)
        end
        else
            X̃, Ỹ ← X[: dts], Y[: dts]
        end
    end
    ctime = time()
    trained ← fit(model, X̃, Ỹ, epoch)
    ctime = time() − ctime
    history = trained
    loss, accuracy = history
    metric = add(ctime, loss, accuracy)
    p+ ← 1/metric
end
p ← sort(reward, NA)
return p
```

Functions	Metrics	EOSA	ABC	WOA	PSO	GA
F1	Best	**0.046173**	0.046591	0.046596	0.046571	9.94223
	Worst	0.046588	20.8892	0.046596	**0.046571**	19.83618
	Mean	0.046465	19.30266	0.046596	0.046571	10.40362
	Median	0.046512	19.15063	0.046596	0.046571	10.1534
	Stdev	0.000107	0.948262	5.20E−18	5.55E−18	0.938523
F2	Best	**0.002556**	0.0028	0.002748	0.002769	39.73652
	Worst	0.002768	245.4735	**0.002748**	0.002769	184.0994
	Mean	0.002608	33.16789	0.002748	0.002769	44.36342
	Median	0.002607	7.26278	0.002748	0.002769	42.07979
	Stdev	4.68E−05	52.19852	3.69E−19	2.82E−19	10.53887
F3	Best	**8.68E−05**	0.000417	0.000416	0.000414	921.248
	Worst	**0.000405**	1498.884	0.000416	0.000414	1269.038
	Mean	0.00011	294.4233	0.000416	0.000414	938.3754
	Median	8.86E−05	203.1162	0.000416	0.000414	929.879
	Stdev	4.55E−05	227.7159	6.23E−20	7.86E−20	30.31403
F4	Best	**1.39E−12**	2.49E−12	2.45E−12	2.49E−12	4.13E + 09
	Worst	2.48E−12	2.57E + 11	**2.45E−12**	2.49E−12	1.34E + 11
	Mean	2.05E−12	2.05E + 11	2.45E−12	2.49E−12	5.68E + 09
	Median	2.18E−12	2.01E + 11	2.45E−12	2.49E−12	4.45E + 09
	Stdev	3.79E−13	1.3E + 10	3.03E−28	4.04E−28	7.3E + 09
F5	Best	**9.30E−13**	2.78E−12	2.80E−12	2.79E−12	395.2324
	Worst	2.86E−12	43,618,954	2.80E−12	**2.79E−12**	194,298
	Mean	1.17E−12	161,597.3	2.80E−12	2.79E−12	2351.452
	Median	9.35E−13	1152.776	2.80E−12	2.79E−12	423.69
	Stdev	4.16E−13	2,214,592	4.04E−28	3.03E−28	12,218
F6	Best	**4.07E−11**	1.02E−10	1.02E−10	1.02E−10	6952.905
	Worst	**1.02E−10**	1,342,862	**1.02E−10**	**1.02E−10**	195,495.6
	Mean	7.19E−11	263,974.3	1.02E−10	1.02E−10	14,746.75
	Median	7.20E−11	253,737.4	1.02E-10	1.02E−10	8375.828
	Stdev	2.03E−11	63,079.51	1.62E−26	1.81E−26	21,265.92
F7	Best	**5.25E−05**	0.000248	0.000248	0.000251	41.79268
	Worst	0.000253	1479.208	**0.000248**	0.000251	823.37
	Mean	0.0002	106.1467	0.000248	0.000251	58.77442
	Median	0.000228	15.67991	0.000248	0.000251	47.54116
	Stdev	6.14E−05	232.7978	4.20E−20	4.74E−20	50.30075
F8	Best	**8.19E−06**	1.98E−05	2.41E−05	2.31E−05	0.009794
	Worst	**2.11E−05**	24.42778	2.41E−05	2.31E−05	5.436187
	Mean	1.32E−05	0.345815	2.41E−05	2.31E−05	0.038439
	Median	1.14E−05	0.005065	2.41E−05	2.31E−05	0.013349
	Stdev	4.71E−06	1.7694	3.22E−21	4.40E−21	0.279212
F9	Best	**5.08E−11**	1.38E−10	1.40E−10	1.39E−10	30,500.52
	Worst	1.40E−10	3.68E + 09	1.40E−10	**1.39E−10**	1.13E + 09
	Mean	9.97E−11	2.53E + 09	1.40E−10	1.39E−10	4,511,122
	Median	1.06E−10	2.44E + 09	1.40E−10	1.39E−10	144,930.2
	Stdev	3.33E−11	2.26E + 08	1.29E−26	1.94E−26	54,440,104
F10	Best	**0.000153**	0.000471	0.000474	0.000475	745.3493
	Worst	0.000475	1599.605	**0.000474**	0.000475	1278.155
	Mean	0.000287	444.8808	0.000474	0.000475	772.7753
	Median	0.00028	315.5723	0.000474	0.000475	760.9054
	Stdev	0.000134	271.854	7.32E−20	8.40E−20	45.18663
F11	Best	**0.000322**	0.000331	0.000333	0.00033	1654.473
	Worst	**0.000331**	2490.439	0.000333	0.00033	2194.09
	Mean	0.000326	1912.671	0.000333	0.00033	1676.178
	Median	0.000325	1851.11	0.000333	0.00033	1664.138
	Stdev	3.15E−06	159.2676	4.88E−20	3.79E−20	45.46335

Continued

Functions	Metrics	EOSA	ABC	WOA	PSO	GA
F12	Best	**1.98E−30**	2.00E−29	1.82E−29	2.01E−29	112,016.4
	Worst	1.96E−29	2.76E + 24	**1.82E−29**	2.01E−29	1.92E + 24
	Mean	5.07E−30	1.12E + 22	1.82E−29	2.01E−29	8.29E + 21
	Median	2.13E−30	8.34E + 17	1.82E−29	2.01E−29	140,116
	Stdev	4.89E−30	1.42E + 23	2.70E−45	3.22E−45	1.06E + 23
F13	Best	0.303455	0.30833	**0.245368**	0.307142	2.686451
	Worst	**0.306781**	2.842985	0.306802	0.307142	2.778775
	Mean	0.304267	1.791644	0.261832	0.307142	2.686881
	Median	0.304119	1.67436	0.245368	0.307142	2.686451
	Stdev	0.00089	0.256993	0.023673	3.61E−17	0.005805
F14	Best	**5.40E−06**	2.46E−05	2.45E−05	2.44E−05	412.1038
	Worst	**2.44E−05**	25,843.77	2.45E−05	**2.44E−05**	13,787.81
	Mean	1.74E−05	21,080.93	2.45E−05	2.44E−05	580.0391
	Median	1.99E−05	20,736.46	2.45E−05	2.44E−05	459.1532
	Stdev	6.69E−06	1251.021	3.22E−21	2.20E−21	760.3748
F15	Best	**0.005718**	0.005899	0.005866	0.005885	14.62603
	Worst	0.005876	130.4765	**0.005866**	0.005885	97.42765
	Mean	0.005761	30.95515	0.005866	0.005885	16.72031
	Median	0.005757	7.717655	0.005866	0.005885	15.10426
	Stdev	4.67E−05	39.61793	6.07E−19	8.67E−19	6.393344

Table 7. Comparison of best, worst, mean, median and standard deviation (stdev) values for EOSA, ABC, WOA, PSO, and GA metaheuristic algorithms using the classical benchmark and IEEE CEC functions over 500 epochs and 100 population size.

Algorithm 3 demonstrates the procedure for the evaluation of the multi-objective criteria as described previously. The expected output of the algorithm is a single value that is passed to the search strategy for improving the configuration of the CNN to achieve optimal performance. It iterates over all the CNN models and randomly generates a batch of image samples from the dataset for training the model. Once the training is completed, the training time, accuracy and loss function is computed and evaluated as a value. The configuration of Algorithm 2 to seamlessly use Algorithm 1 and 3 is presented in the next section, which is focused on the experimentation of the proposed approach.

Experimentation

This study aimed to obtain the most optimal neural architecture by applying a novel search space and search strategies for solving the classification problem defined in Introduction section. Therefore, the experiment carried out was two-fold: firstly, we experimented with the proposed search space and search strategies to demonstrate the effectiveness of the methods. Secondly, the top-performing CNN architecture obtained from the first experiment was then applied to detect abnormalities in digital histopathology images confirming the presence of breast cancer. This section therefore presents a detailed outline of configurations, parameter values, and characteristics of the histopathological datasets applied in the experimentation.

Search space configuration. The configuration for generating potential solutions into the search space is presented in this subsection. This configuration is necessary to guide Algorithm 1 to boost the possibility of generating potential solutions (neural networks) for maximizing classification accuracy and minimizing loss in digital histopathology images for the detection of breast cancer. This configuration provides the encoding scheme proposed in this study with a wide range of parameters to generate and encode the possible network topologies to make the search efficient and effective.

The general hyperparameters (GH) block consists of four parameters and is summarized in GH = {G_b, G_α, G_e, G_o} so that G_b, G_α, G_e, and G_o are computed using $G_b = 2^n - 1$, $G_\alpha = rand(1|5) \cdot 10^{-n}$, $G_o = O[n]$, and $G_e = 5$ respectively. Where n = 0, 1, 2 for batch size (G_b), is represented as random mode = 0, batch mode = 1, mini-batch mode = 3; learning rates (G_α) is computed by generating random number between 1 and 5 resulting in $\alpha = \{1\times10-5,\ 5\times10-5,\ 1\times10-4,\ 5\times10-4,\ 1\times10-3,\ 5\times10-3,\ 1\times10-2,\ 5\times10-2,\ 1\times10-1,\ 5\times10-1\}$;n = 1,2,3,4,5 for G_α; n = 0,1,2,3,4,5,6,7 for G_o, and O = {0 = > "SGD", 1 = > "Adam", 2 = > "RMSprop", 3 = > "Adagrad", 4 = > "Nestrov", 5 = > "Adadelta", 6 = > "Adamax", 7 = > "Momentum", 8 = > " Nestrov Accelerated Gradient"}. The range of values derivable for the input-Zeropadding block, represented as IZ = { Z_α}, to determine if input will be zero-padded or not, is shown computed using $I_z = rand(0|1)$.

Convolutional block (CB) hyperparameters are denoted as follows: CB = {C_L, C_C, C_{AF}, C_K, C_F, C_{PS}, C_{PT}, C_R} and so that C_L, C_C, C_{AF}, C_K, C_F, C_{PS}, C_{PT}, and C_R are computed using $C_L = 2n - 1, C_C = 3 - n, C_{AF} = AF[n]$, $C_K = 2^n$, $C_F = 2 + (n - 1)$, $C_{PS} = 2 + n$, $C_{PT} = n$, and $C_R = n$ respectively. Where n = 1,2,3,4 for C_L, to determine the number of blocks of convolutional layers an arbitrary neural network may possess; n = 0, 1, 2 for C_C, to compute

Functions	Metrics	EOSA	ABC	WOA	PSO	GA
CEC01	Best	**2.75E−11**	2.78E−11	2.78E−11	2.78E−11	6,500,451
	Stdev	**1.46E−15**	8.44E + 08	2.78E−11	3.88E−27	3.10E + 08
	Median	2.75E−11	4.97E + 09	3.39E−27	2.78E−11	17,405,089
CEC02	Best	**2.48E−12**	2.49E−12	2.45E−12	2.49E−12	4.17E + 09
	Stdev	9.11E−17	1.30E + 10	2.45E−12	2.83E−28	7.53E + 09
	Median	2.48E−12	2.02E + 11	4.64E−28	2.49E−12	4.44E + 09
CEC03	Best	**1.01E−10**	1.02E−10	1.02E−10	1.03E−10	8666.065
	Stdev	3.19E−14	124,317.8	1.02E−10	2.13E−26	23,773.8
	Median	1.01E−10	251,561	1.36E−26	1.03E−10	12,804.61
CEC04	Best	3.71E−12	**3.68E−12**	3.70E−12	3.73E−12	1,099,091
	Stdev	1.35E−16	9.64E + 09	3.70E−12	7.88E−28	2.26E + 09
	Median	3.71E−12	8.50E + 10	5.65E−28	3.73E−12	5,359,283
CEC05	Best	**0.045669**	0.045719	0.045711	0.045704	18.25292
	Stdev	9.25E−07	0.957905	0.045711	5.90E−18	0.531952
	Median	0.045669	20.02451	5.90E−18	0.045704	18.40464
Shift CEC06	Best	0.001299	0.001302	**0.001298**	0.001298	618.1048
	Stdev	8.59E−09	31.98589	0.001298	1.63E−19	6.159877
	Median	0.001299	710.5576	1.52E−19	0.001298	619.0061
Shift CEC07	Best	**0.000224**	0.000228	0.000227	0.000226	761.7748
	Stdev	7.57E−09	141.8918	0.000227	3.25E−20	66.00672
	Median	0.000224	2526.215	4.07E−20	0.000226	766.6583
Shift CEC08	Best	**0.000343**	0.000345	0.000344	0.000343	1557.367
	Stdev	4.04E−08	156.4804	0.000344	3.52E−20	44.58426
	Median	0.000343	1756.355	3.79E−20	0.000343	1567.959
Shift-rotate CEC08	Best	**0.000333**	0.00033	0.000334	0.000332	1657.835
	Stdev	1.49E−08	158.7789	0.000334	4.88E−20	45.19802
	Median	0.000333	1857.348	4.34E−20	0.000332	1671.562
Shift CEC09	Best	**2.16E−05**	2.17E−05	2.19E−05	2.18E−05	22,425.71
	Stdev	2.54E−09	3268.026	2.19E−05	1.86E−21	1633.723
	Median	2.16E−05	21,565.89	2.20E−21	2.18E−05	22,826.18

Table 8. Comparison of best, worst, mean, median and standard deviation (stdev) values for EOSA, ABC, WOA, PSO, and GA metaheuristic algorithms using the constrained IEEE CEC-2017 benchmark functions over 500 epochs and 100 population size.

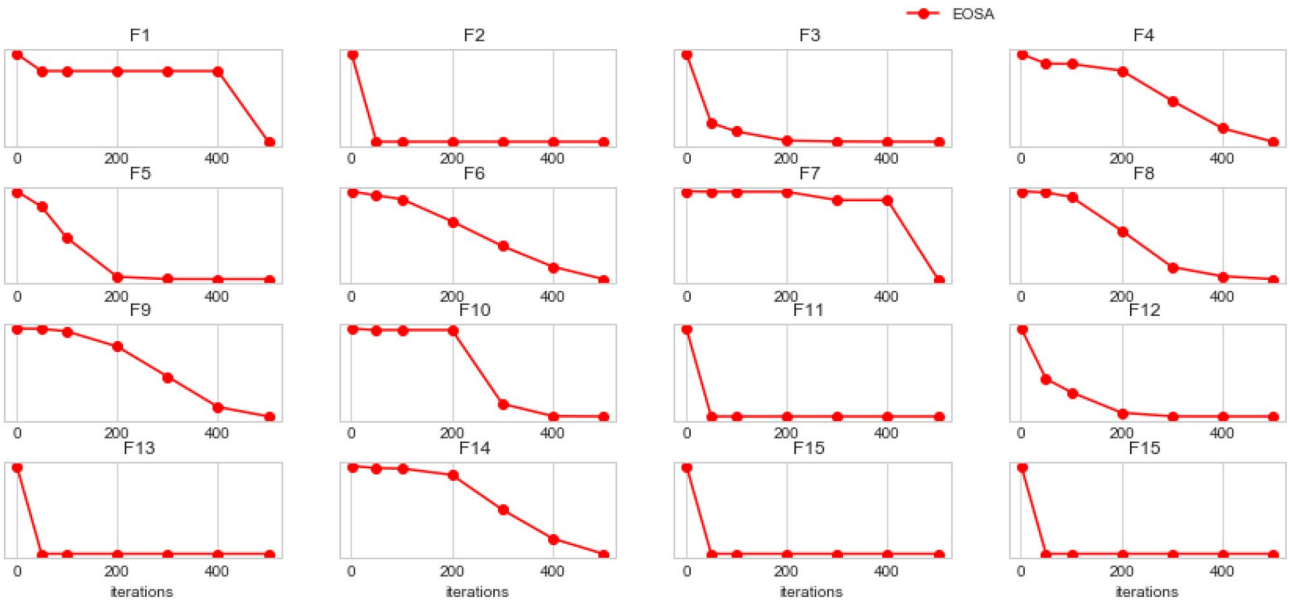

Figure 5. Convergent curves of EOSA optimization algorithm on F1, F2, F3, F4, F5, F6, F7, F8, F9, F10, F11, F12, F13, F14 and F15 standard benchmark functions.

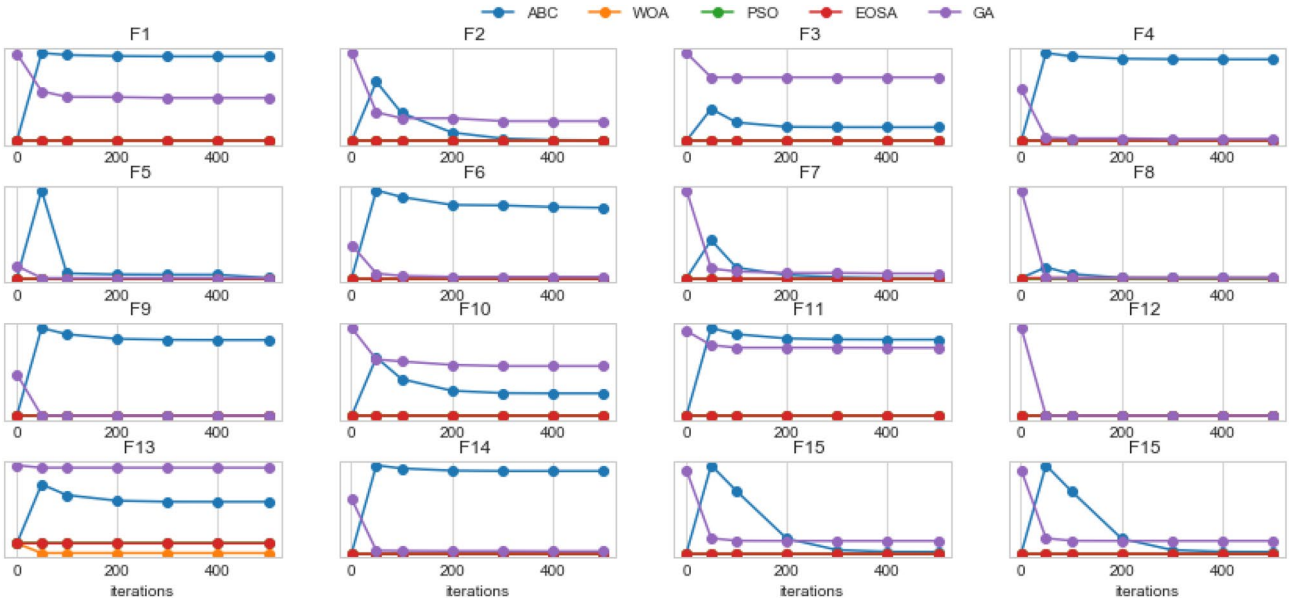

Figure 6. Comparison of convergence curves of the performance of EOSA, ABC, WOA, PSO, and GA optimization algorithms on all standard benchmark functions applied in this study.

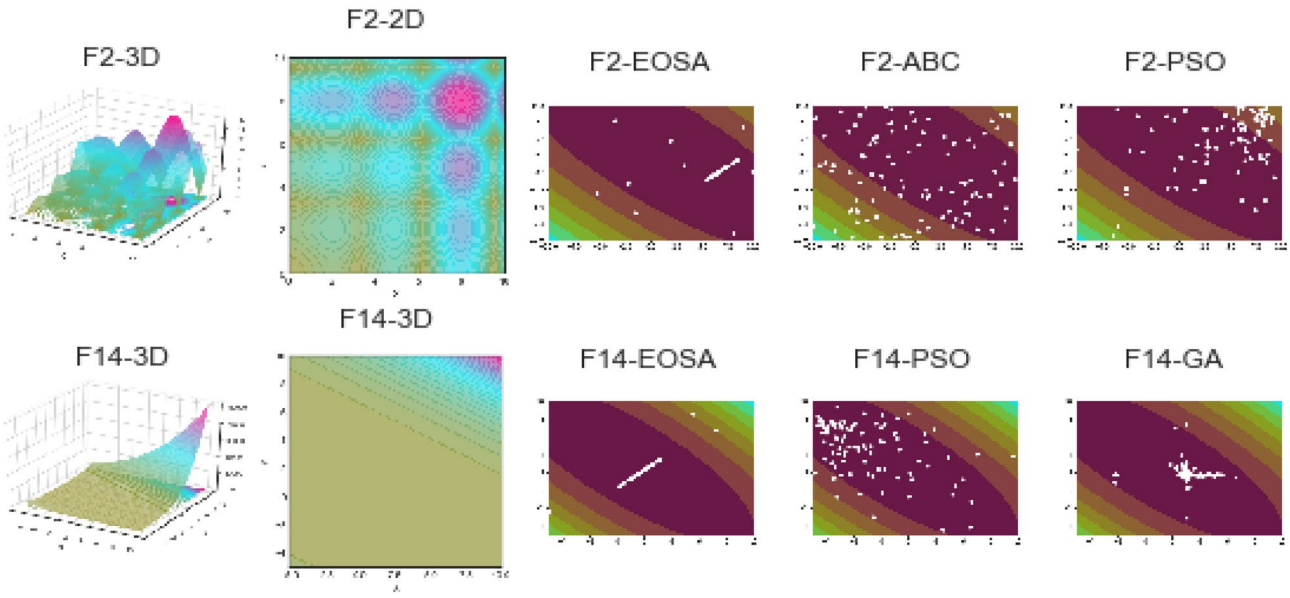

Figure 7. Comparison of convergence curves of the performance of EOSA, ABC, WOA, PSO, and GA optimization algorithms on all standard benchmark functions applied in this study.

convolutional layers in a block; n = 0, 1, 2 for C_{AF}, and indexes AF = {0 = > "ReLU", 1 = > "LeakyReLU", 2 = > "Parametric ReLU"}; n = 3, 4, 5, 6, 7, 8, 9, 10 for C_K; n = 0, 2, 4, 6, 8, 10 for C_F; n = 0, 1, 2, for C_{PS}; n = 0 = > Max pooling, 1 = > Average pooling C_{PT}, n = 0 = > L1, 1 = > L2 and 2 = > L1L2 regularizations for C_R; meanwhile, our configuration allows for the use of padding as *same* and stride = *1* in convolutional layers.

Fully-connected block (FCB) parameters are denoted by FCB = {F_L, F_{AF}, F_D, F_R} and are computed as follows: F_L, F_{AF}, F_D, and F_R using $F_L = 1 + n$, $F_{AF} = FAF(n)$, $F_D = \frac{1}{n}$, and $F_R = n$ respectively, Where n = 0, 1 for computing the number of F_L flatten operations; n = 0, 1 for obtaining F_{AF} which is further defined by indexing: FAF = {0 = > "softmax", 1 = > "sigmoid"}; n = 1.0, 1.1…0.2.0, are used in computing F_D; n = 0 = > L1, 1 = > L2 and 2 = > L1L2 regularizations for F_R. The Loss function block denoted by LF has only one element: {LF_L} where loss function for the search space can be drawn from the {categorical cross-entropy, sparse-cross-entropy} when n = 0 and 1, respectively.

The summary presented in Table 4 identifies the collection of possible values derivable for the search space in configuring potential CNN architectures. The EOSA algorithm is also configured and experimented with the

Parameters	Top-1	Top-2	Top-3	Top-4	Top-5
Dataset batching	Random sample size	Half of dataset	Random sample size	Half of dataset	Random sample size
Zero padding	Yes	Yes	Yes	Yes	Yes
No. Convo-Pool blocks	2	3	2	3	6
Details of Convolution layers	[1Convo, 'relu', 32, 9, 2, 'Avg', 'L1'], [3Convo, 'relu', 64, 9, 2, 'Avg', 'L1']	([3Convo, 0.005, 'Adagrad', 3], True, [2, 'relu', 32, 3, 2, 'Max', 'L1'], [4, 'relu', 64, 3, 2, 'Avg', 'L1'], [4, 'relu', 128, 3, 2, 'Avg', 'None'],	[1Convo, 'relu', 32, 9, 2, 'Avg', 'L1'], [3Convo, 'relu', 64, 9, 2, 'Avg', 'L1']	[2Convo, 'relu', 32, 3, 2, 'Max', 'None'], [4, 'relu', 64, 3, 2, 'Avg', 'None'], [4, 'relu', 128, 3, 2, 'Max', 'L1']	[3Convo, 'relu', 32, 9, 2, 'Max', 'L1'], [2, 'relu', 64, 1, 2, 'Avg', 'None'], [3, 'relu', 128, 11, 2, 'Max', 'None'], [1, 'relu', 256, 9, 2, 'Avg', 'L1'], [2, 'relu', 512, 7, 2, 'Max', 'None'], [3, 'relu', 1024, 3, 2, 'Avg', 'None']
Pool size	2 × 2	2 × 2	2 × 2	2 × 2	2 × 2
Filters size	9 × 9, 9 × 9	3 × 3, 3 × 3, 3 × 3	9 × 9, 9 × 9	3 × 3, 3 × 3, 3 × 3	9 × 9, 1 × 1, 11 × 11, 9 × 9, 7 × 7, 3 × 3
Filter count	32 × 32, 64 × 64	32 × 32, 64 × 64, 128 × 128	32 × 32, 64 × 64	32 × 32, 64 × 64, 128 × 128	32 × 32, 64 × 64
No. FC layers	2	3	2	3	1
Dense Layer activation function and dropout rate	Softmax and 0.48	Softmax and 0.5 and LI	Softmax and 0.5	Softmax and 0.45 and L1	Softmax and 0.47 and L1
Learning rate	0.05	0.005	0.05	0.005	1e-05
Optimizer	RMSprop	Adagrad	RMSprop	Adagrad	Adam
Classifier	Categorical crossentropy	Categorical crossentropy	Categorical crossentropy	Categorical crossentropy	Categorical crossentropy

Table 9. Comparison of parameters for the best five (5) initial neural network configurations (solutions) generated for the search space.

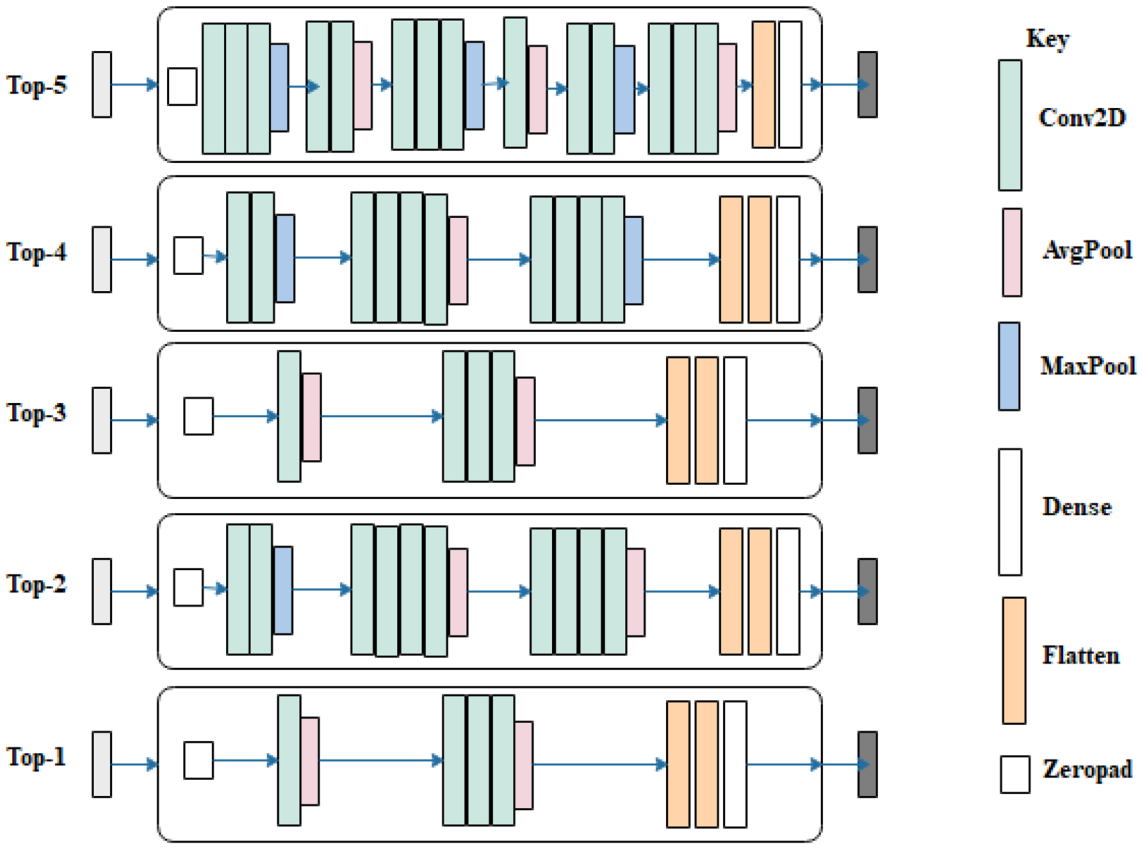

Figure 8. Neural network architectures of the Top-5 generated network architectures generated for the search space.

S/N	Accuracy					Loss			
	Best	Mean	Median	Worst	Stdev	Worst	Median	Best	Latency
Top-5	0.551	0.313	0.332	0.030	0.247	2.79E + 09	1.84	1.84	12.87
Top-4	0.573	0.376	0.359	0.111	0.097	9.13E + 08	3.16	1.31	12.52
Top-3	0.613	0.354	0.326	0.136	0.137	5.1E + 09	2.21	1.318	21.26
Top-2	0.627	0.396	0.350	0.098	0.051	26,261,178	2.21	1.231	39.21
Top-1	0.655	0.415	0.417	0.147	0.150	23,565.56	11,137.88	1.297	93.59

Table 10. Performance comparison for training the five (5) best performing CNN architectures from EOSA-NAS algorithm using mean, median, accuracy and standard deviation for accuracy, and loss, computation time values for the 250 epochs of EOSA.

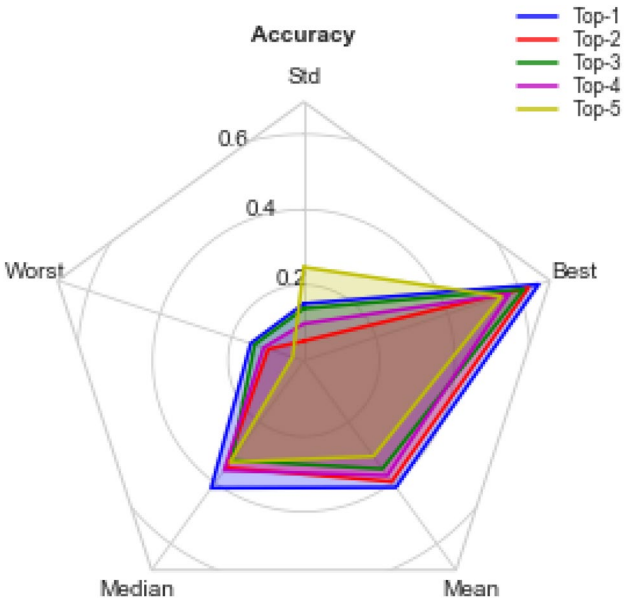

Figure 9. A radar plot showing the performance comparison of the top-5 best performing network architectures from EOSA-NAS algorithm based on mean, median, worst, and best accuracy values.

Architectures	F1-score	Precision	Sensitivity	Specificity	Recall	Accuracy	Kappa
Top-4	0	0.0	–	–	0	0. 24	–
Top-2	0.1	0.1	0.1	0.1	0.1	0.1	0.1
Top-3	0	0	–	0	0.1	0. 25	0
Top-1	0.1	0.1	0.1	0.1	0.1	0.1	0.1

Table 11. Performance comparison for prediction of the four (4) best performing CNN architectures of the EOSA-NAS algorithm using AUC, precision, recall, sensitivity, specificity, accuracy and loss after full train for 60, 70 and 100 epochs.

parameters listed in Table 5. The following sub-section presents the configuration of the environment for the experiment.

Configuration for experimentation environment. Exhaustive experimentation done for evaluating the proposed EOSA, described in Algorithm 1, was carried out in a workstation environment with the following configurations: Intel (R) Core i5-7500 CPU 3.40 GHz, 3.41 GHz; RAM of 16 GB; and 64-bit Windows 10 OS for each configuration of the system on the network. Similarly, those for the neural architecture search and for convolutional and classification processes were carried out in the same computational environment.

Experimentation dataset. This study is focused on applying the experimentation of the proposed NAS model on digital histopathological images. We allowed every candidate CNN architecture to be evaluated using

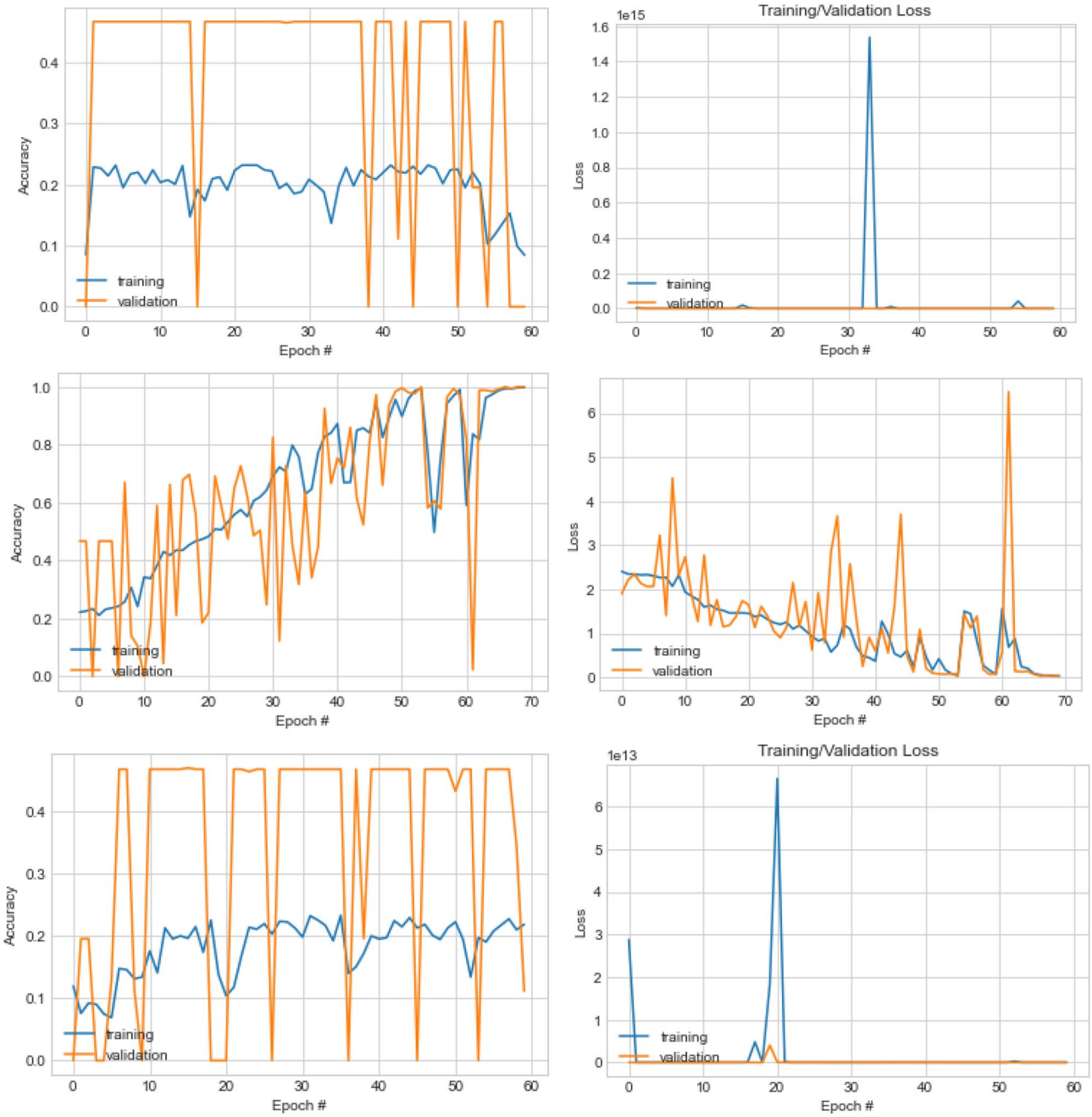

Figure 10. Plot of the accuracy and loss values for the training of the Top-1, 2, and 3 architectures respectively which were optimized using the EOSA-NAS model, showing their performances after sixty (60) training epoch.

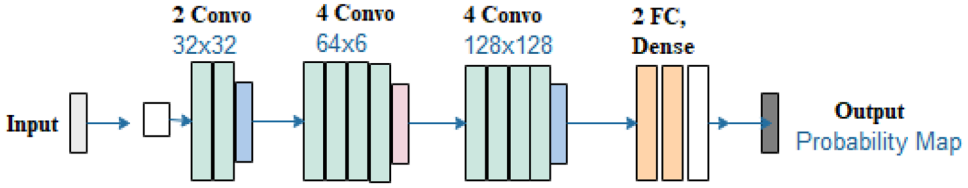

Figure 11. Neural network architecture of the Top-1 architecture optimized using EOSA-NAS model, which represents the overall best performing architecture after hundred (100) training epoch.

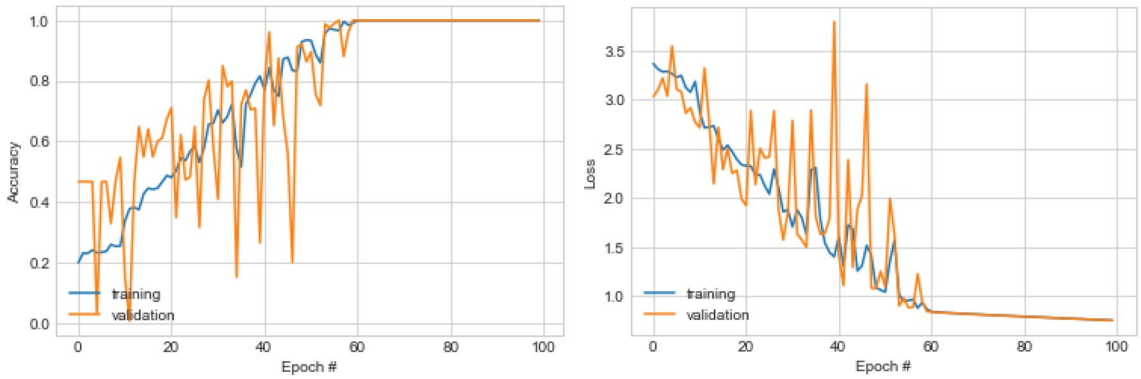

Figure 12. Plot of the accuracy and loss values for the training of the Top-1 architecture optimized using EOSA-NAS model, which represents the overall best performing architecture after hundred (100) training epoch.

References	Methods	Performance	Dataset
Zheng et al. [19]	Nucleus-guided CNN	Accuracy 96.4%, Sensitivity 0.955, Specificity 0.964	Images from Motic (Xiamen) Medical Diagnostic Systems
Nejad et al. [17]	CNN + Data augmentation	Detection rate 77.5%	BreakHis database
Araújo et al. [20]	CNN + Support Vector Machine	Accuracies of 77.8%, sensitivity of 95.6%	Bioimaging 2015 breast histology classification challenge
Han et al. [18]	Structured Deep Learning Model + Data augmentation	93.2% accuracy	BreakHis database
Saha et al. [22]	Handcrafted features + CNN	92% precision, 88% recall and 90% *F*-score	MITOS-ATYPIA-14, ICPR-2012, and AMIDA-13 datasets
Zhu et al. [24]	Squeeze-Excitation-Pruning (SEP) + CNN	Accuracy of 87.5%	BreaKHis and BACH dataset
Xie et al. [23]	Inception_V3 and Inception_ResNet_V2	Accuracy 96.84%	BreaKHis
Kandel and Castelli [25]	CNN	AUC of 95.46%	PatchCamelyon
Hägele et al. [26]	CNN + explanation method	Improved AUC by 5%	BRCA
This study	EOSA-NAS CNN	Accuracy 100%	BreakHis and BACH databases

Table 12. Comparison of NAS-based CNN design with state-of-the-art canonical CNN design approach for detection and classification of breast cancer using histopathology images.

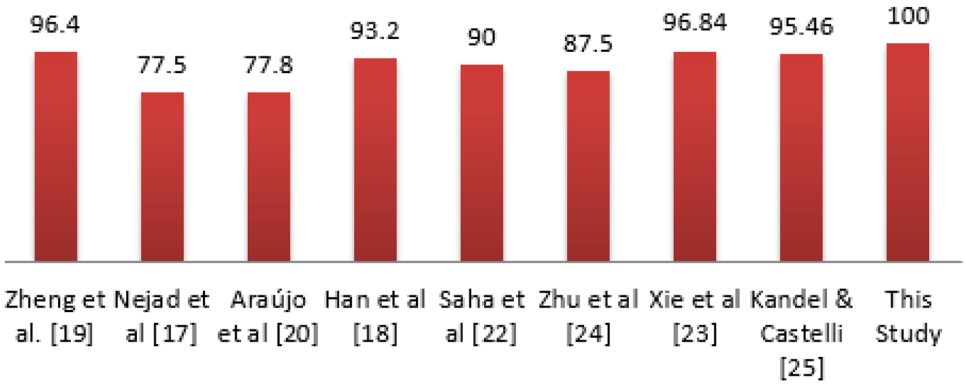

Figure 13. Comparison of the CNN architecture designed using EOSA-NAS model with state-of-the-art CNN architectures applied to the detection of breast cancer in histopathology images.

these images for performance evaluation. We chose the publicly available benchmark datasets, namely BACH[57] and BreakHis[58,59]. The motive for choosing these datasets was to provide sufficient data for the experimentation and allow for the reproducibility of the proposed approach. The experiments were staged in two (2): generating and searching for best performing networks, and the second experiment for full training of top-5 networks. As a result, we rigorously applied the datasets to the top-performing CNN architecture resulting from the stage 1 experiment.

The image samples obtained from the BACH and BreakHis datasets were further resized to sizes 224×224 to allow for input into the neural architectures and the top-performing neural network architectures. This resizing became necessary because the original image size from BACH was 2048×1536 pixels and consisted of 400 Hematoxylin and eosin (HE) stained images, while the BreakHis dataset contained a total of 9,109 (actually 7,909 samples after removal of tissue samples) microscopic images with an image size of 700×460 pixels. The classes of images obtained from BACH are normal, benign, in situ carcinoma or invasive carcinoma, while those of BreakHis are categorized as benign or malignant. The benign and malignant samples of BreakHis are further categorized into adenosis (A), fibroadenoma (F), phyllodes tumor (PT), and tubular adenona (TA) as benign; and carcinoma (DC), lobular carcinoma (LC), mucinous carcinoma (MC) and papillary carcinoma (PC) as malignant. Figures 3 and 4 show some samples drawn from BACH and BreakHis datasets respectively.

According to their classes, the breakdown for the BACH image samples are: 100 samples of normal, 100 samples of benign, 100 samples of in situ carcinoma, and 100 samples of invasive carcinoma. Similarly, the BreakHis datasets image samples contain 2,480 benign and 5,429 malignant samples. Both the BACH and BreakHis datasets image samples are 3-channelled RGB. Also, we discovered that the magnification for the BACH dataset is $200\times$, and those of BreakHis were presented at 40X, 100X, 200X, and 400Xmagnifications. We, however, preprocessed the images to allow for resizing and elimination of potential errors arising from the stain on the raw inputs. We applied the basic operations of reduction of background noise and image enhancement. Furthermore, we applied image normalization operations from Reinhard[60] and Macenku[61] to normalize our histopathology images.

Results and discussion

In this section, the result of the experimentation is presented, and the findings are discussed. Two categories of results are considered: performance of the EOSA algorithm as compared with four similar metaheuristic algorithms and the performance of the NAS model in obtaining best performing CNN architecture.

Performance evaluation of EOSA metaheuristic algorithm. The EOSA experiment was carried out using 25 benchmark optimization functions listed in Table 6. These same functions were applied to artificial bee colony (ABC), whale optimization algorithm (WOA), particle swarm optimization (PSO), and genetic algorithm (GA) metaheuristic algorithms. Each of these optimization algorithms was executed for 500 epochs and 20 runs for stability. The result of the experimentation is listed in Table 7.

Table 7 shows that EOSA had the lowest values for the best solutions using the F1–F12 and F14–F15 compared with the best solutions for ABC, WOA, PSO and GA. Although PSO maintained a lead only in F13 compared to EOSA, ABC, WOA, and GA, the performance margin was small compared with EOSA, and EOSA showed superiority in fourteen (14) of fifteen (15) functions evaluated. Also, EOSA yielded a significant performance compared with ABC, WOA, PSO and GA based on the values of worst solutions for F1–F15. Table 8 shows that the EOSA performed well in the solutions obtained for the constrained IEEE CEC-2017 benchmark functions compared to other competing algorithms. The EOSA had obtained a total of eight (8) best results out of the nine (9) functions.

Figures 5 and 6 illustrate the convergence of EOSA on F1–F15 and convergence of EOSA compared with ABC, WOA, PSO and GA on F1–F15, respectively. The plots in Fig. 5 confirm that the convergence of EOSA is impressive though the significance of its convergence has been overshadowed in Fig. 6 due to variation of values. Also, we observed the convergence of each solution for EOSA, ABC, WOA, PSO, and GA using a scatter plot. The outcome, as shown in Fig. 7, aligns with the graphs in Figs. 5 and 6. The results show that the EOSA algorithm is a candidate optimization algorithm capable of sufficiently learning the problem of automating the design of CNN architectures for the search strategy of a NAS model. Furthermore, the results guarantee that EOSA can compete with state-of-the-art optimization algorithms.

Now that the performance of EOSA as a metaheuristic algorithm was confirmed to be suitable for optimizing the search strategy of a NAS model, we proceeded to experiment using it in the NAS model experimentation. The result of this experiment is presented and discussed in the next section.

Performance evaluation of CNN design using EOSA-NAS. The initial solutions (CNN architectures) generated into the search space were optimized using the EOSA algorithm during the search strategy stage of the NAS model. The optimization in EOSA was executed for 500 epochs, and the configuration of each solution was reevaluated using the evaluation strategy of our NAS model. The optimized CNN architectures were logged for each iteration, while the final configurations for all the CNN architectures were examined and used for the result presented in this section. Table 9 presents the configurations of the top-5 CNN architectures, and their network topologies are shown in Fig. 8.

In Table 9, a detailed definition of each of the top five (5) architectures is outlined. Similarly, a graphical illustration of the architectures is shown in Fig. 8. We found that the Top-1 architecture represents a minimal utilization of convolutional and pooling operations while the Top-5 architecture has more of these operations. For instance, the Top-1 has two convolutional blocks with a single convo operation in a block and three convo operations in the second block. In contrast, the Top-5 has 6 convolutional blocks with mostly three convo operations combined with either max or average pooling operations. Another interesting outcome of the resulting top five architectures is that we found a structural similarity between the Top-1 compared with Top-3 and another variation of structural similarity between the Top-2 and Top-4 architectures.

However, in Table 9, we observed that whereas these similarities exist in the structural view of the architectures, there are some significant variations in their detailed implementations. For instance, we found that the three convolutional blocks of the Top-2 architecture have the Max-Avg-Avg pooling operations and the Top-4 Max-Avg-Max pooling operations. In addition, the second convolutional block of the Top-2 architecture allows

the use of the L1 (weight decay) network regularizer, whereas that of Top-4 uses none. The reverse of this arrangement is seen in the third convolutional block.

The result in Table 10 shows that the Top-1 architecture achieved a good performance during the 250 iterations as its accuracy for best, mean and median are 0.655, 0.415, and 0.417, respectively. This is a distance from the Top-5, which maintained the values of 0.551, 0.313, and 0.332 for best, mean and median, respectively. We found a similar trend as shown in the results of Top-2, Top-3, and Top-4 performing architectures. The interpretation of these variations informs us that the Top-1 architecture learned the classification problem very well compared to the remaining four (4) architectures. Using a radar chart, we plotted the performance of the top five (5) network architectures using the resulting values of their best, mean, median, worst, and standard deviation. Radar charts provide a good way for visualizing comparisons of data of related attributes or variables which are displayed along their axis.

In Fig. 9, we see that the overall difference in visual representation is apparent by the size and shape of the polygons' pointing. The polygons point to the best axis more closely because the top5 architectures have their highest accuracy within this variable. The nearness of the polygons' closeness to the axis is followed by those of *mean* and *median* variables, confirming the distribution of accuracies for the top-5 architectures within those two variables. Lastly, we see that the pointing of the polygons of the *worst* and *standard deviation* variables is far from their axes. These distributions of accuracies across the five variables demonstrate the discrepancies which exist in the performance of the top5 architectures. Clearly, the Top-1 architecture has the highest and best performance followed by the Top-2, then the Top-3, Top-4 and Top-5.

Complete training of the top one (1) best performing architectures listed in Table 11 and illustrated in Fig. 10 showed that only the Top-1 and Top-2 demonstrated significant results. The two previous architectures overshadowed the outcome of those of Top-3 and Top-4. As a result, the Top-1 and Top-2 architectures were further evaluated beyond the 500 epochs of training. We found the Top-1 architecture converging well and learning the problem with impressive accuracy from the 60th epoch to the 100th epoch. Meanwhile, that of Top-2 architecture only began to show this stability later. This implies that the Top-1 architecture remains the best architecture that has learnt the classification problem well.

To fully evaluate and investigate the performance of the top five architectures, we experimented again with these architectures on larger datasets and allowed for training using a longer epoch. In Table 11, we see the performance of each of the architectures in terms of F1-score, precision, recall, sensitivity, specificity, accuracy and Kappa values after the full train. We applied the distributions of these variables to plot the boxplot of their corresponding values and found that an interesting distribution was seen for values in each distribution. Also, the result obtained from the Table showed that the architecture corresponding to the CNN model in Fig. 11 outputs the optimal performance with an accuracy of 0.1. This then reflects the most acceptable CNN configuration required to learn the problem of classification of digital histopathology images using deep learning.

Also, plotting the graph of the training phase of the Top-1 CNN model, we found that the loss function graph in Fig. 12 showed that the problem was learnt well as we see the loss values for those of training and validation overlapping as the training progressed. Similarly, the accuracy plot in the same figure demonstrates the evidence that the resulting CNN model is a candidate solution for consideration in future research on the application of deep learning to the classification of abnormalities in digital histopathology images.

The result shown in Table 12 shows that most efforts in designing CNN models for histopathology image classification have all been approached using manual methods. Although the studies listed in the Table demonstrate some significant performance, the outcome of our experimentation confirms that automating the process is more beneficial. While the works of Zheng et al. [19] and Kandel and Castelli [25] compete with our method, we note that our method outperforms them. The graph in Fig. 13 shows a pictorial representation of the performance of all similar studies when compared with the outcome of this study.

This study is focused on investigating the outcome of applying a NAS-based approach to the automation for the design of CNN architectures in the classification of breast histopathology images. The study aimed to address the difficulty in learning the problem associated with the domain. The outcome of the experimentation performed using EOSA-NAS based model for generating and optimising CNN architecture has proven very effective. This is based on the results obtained which have shown that applying the NAS approach to finding the best network configuration in detecting abnormalities in histopathology yields better performance. The accuracy obtained confirms that the application of the EOSA metaheuristic algorithm contributed to the overall performance of the NAS model. Meanwhile, this study has also shown that the proposed optimization algorithm, EOSA, competes well with similar state-of-the-art algorithms while showing superiority in the case of GA.

The EOSA metaheuristic algorithm was experimented with using fifteen (15) standard benchmark functions to demonstrate its viability and usefulness for solving optimization problems as in NAS model. Therefore, this study's finding confirms that automatic design for the CNN model in the classification task of histopathology images is more accurate than the manually designed models. Secondly, we showed that using the EOSA metaheuristic algorithm in a NAS-based model in optimizing purpose is also very positive. The approach in this study is in contrast to the widely adopted method for designing CNN architectures in learning the problem of detection of abnormalities in histopathology samples. Therefore, the proposed method offers a new order for the design of CNN architectures for this class of problem for the domain mentioned.

Conclusion

This study demonstrates the importance of applying the NAS-based method to the challenge of designing CNN architectures. It further shows that applying the approach to learning abnormalities in histopathology images is of great benefit compared with the manual CNN design method. Moreover, the metaheuristic algorithm (EOSA) used to optimise the search strategy of the NAS model proves to be very relevant to tackling the problem.

Although most studies that have applied deep learning to the task of detection and classification of breast histopathology images have shown some good performance, the findings of this study showed that using a NAS-based technique will improve detection and classification rate. The outstanding performance of the EOSA and NAS models hybridisation yielded a state-of-the-art CNN model that sufficiently learns the problem in the domain. The most interesting performance of the resulting CNN architecture is the values of the metrics: accuracy, sensitivity, specificity, precision, and recall, all leading to reduced classification error and reduced false-positive rates.

The outcome of this study demonstrates the evidence that the resulting CNN model remains a candidate solution for consideration in future research on the application of deep learning to the classification of abnormalities in digital histopathology images for the detection of breast cancer. The NAS strategy applied in this study and the resulting candidate architecture provides researchers with an understanding of network configuration suitable for using digital histopathology. However, the resulting top-5 and the best performing CNN architectures were trained to learn the classification problem of detecting abnormalities in histopathology images suggesting the presence of cancer. Hence, the performance may not measure up when applied to digital mammography.

In future, we recommend a comparative study investigating the performance of biology and swarm-based optimization algorithms in the use of search strategy for a NAS-based model. Considering the outstanding performance of the EOSA-NAS model proposed in this study, we recommend applying it to improve the search for configuring generative adversarial networks (GANs) for synthesizing histopathology images.

Received: 29 May 2021; Accepted: 16 September 2021
Published online: 07 October 2021

References

1. Saadat, M. N., & Shuaib, M. Advancements in Deep Learning Theory and Applications: Perspective in 2020 and beyond. Advances and Applications in Deep Learning, https://www.intechopen.com/books/advances-and-applications-in-deep-learning/advancements-in-deep-learning-theory-and-applications-perspective-in-2020-and-beyond (2021). Accessed 18 July 2021
2. Garg, A., Saha, A. K., & Dutta, D. Revisiting Neural Architecture Search (2020). Cited 2021 July 16, p. 8 p. https://arxiv.org/abs/2010.05719
3. Kyriakides, G., & Margaritis, K. An Introduction to Neural Architecture Search for Convolutional Networks (2020). Cited 2021 July 16, p. 11. https://arxiv.org/abs/2005.11074
4. Ahmad, M., Abdullah, M., Moon, H., Yoo, S. & Han, D. Image classification based on automatic neural architecture search using binary crow search algorithm. *IEEE Access* **2020**, 189891–189912 (2020).
5. Wang, N., Gao, Y., Chen, H., Wang, P., Tian, Z., Shen, C., & Zhang, Y.NAS-FCOS: Fast Neural Architecture Search for Object Detection. *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2020).
6. Weng, Y., Zhou, T., Li, Y. & Qiu, X. NAS-Unet: neural architecture search for medical image segmentation. *IEEE Access* **1**, 1 (2019).
7. Wistuba, M., Rawat, A., & Pedapati, T. Automation of Deep Learning - Theory and Practice. *ICMR '20, Dublin, Ireland Proceedings published June 8, 2020, 26–9*.
8. Weng, L. Neural Architecture Search. https://lilianweng.github.io/lil-log/2020/08/06/neural-architecture-search.html (2020). Accessed 2 May 2021.
9. Nahid, A. A., Mehrabi, M. A. & Kong, Y. Histopathological breast cancer image classification by deep neural network techniques guided by local clustering. *BioMed Res. Int.* **1**, 1 (2018).
10. Pal, R. & Saraswat, M. Histopathological image classification using enhanced bag-of-feature with spiral biogeography-based optimization. *Appl. Intell.* **49**, 3406–3424 (2019).
11. Zhu, C. *et al.* Breast cancer histopathology image classification through assembling multiple compact CNNs. *BMC Med. Inf. Decn. Mak.* **19**, 1 (2019).
12. Aswathy, M. A. & Jagannath, M. Detection of breast cancer on digital histopathology images: Present status and future possibilities. *Inf. Med Unlkd* **8**, 74–79 (2017).
13. Ren, P. *et al.* A comprehensive survey of neural architecture search: Challenges and solutions. *ACM Comput. Surv.* **54**, 1–34 (2021).
14. Ci, Y., Lin, C., Sun, M., Chen, B., Zhang, H., *et al.* Evolving Search Space for Neural Architecture Search. https://arxiv.org/abs/2011.10904 [cs.CV], [Preprint] (2020). Cited 2021 July 13, p. 11.
15. Xu, J. *et al.* Stacked sparse autoencoder (SSAE) for nuclei detection on breast cancer histopathology images. *IEEE Trans. Med. Img* **35**, 119–130 (2016).
16. Bejnordi, B. E. *et al.* Context-aware stacked convolutional neural networks for classification of breast carcinomas in whole-slide histopathology images. *J. Med. Img.* **1**, 1 (2017).
17. Nejad, E. M., Affendey, L. S., Latip, R. B., & Ishak, I. B., Classification of Histopathology Images of Breast into Benign and Malignant using a Single-layer Convolutional Neural Network. *ICISPC 2017: Proceedings of the International Conference on Imaging, Signal Processing and Communication* (2017), pp. 50–53.
18. Han, Z. *et al.* Breast cancer multi-classification from histopathological images with structured deep learning model. *Sci. Rep.* **7**, 4172 (2017).
19. Zheng, Y. *et al.* Feature extraction from histopathological images based on nucleus-guided convolutional neural network for breast lesion classification. *Patrn Recog* **1**, 14–15 (2017).
20. Araújo, T. *et al.* Classification of breast cancer histology images using Convolutional Neural Networks. *PLoS ONE* **12**, 1 (2017).
21. Khosravi, P., Kazemi, E., Imielinski, M., Elemento, O. & Hajirasouliha, I. Deep convolutional neural networks enable discrimination of heterogeneous digital pathology images. *EBioMedicine* **27**, 317–328 (2018).
22. Saha, M., Chakraborty, C. & Racoceanu, D. Efficient deep learning model for mitosis detection using breast histopathology images. *Comput Med Img and Grap* **64**, 29–40 (2018).
23. Xie, J., Liu, R., Iv, J. L. & Zhang, C. Deep learning based analysis of histopathological images of breast cancer. *Fronti Genet* **1**, 1 (2019).
24. Zhu, C. *et al.* Breast cancer histopathology image classification through assembling multiple compact CNNs. *BMC Med. Inf. Decn Makn* **1**, 1 (2019).
25. Kandel, I. & Castelli, M. A novel architecture to classify histopathology images using convolutional neural networks. *Appl. Sci.* **10**, 8 (2020).
26. Hägele, M. *et al.* Resolving challenges in deep learning-based analyses of histopathological images using explanation methods. *Sci. Rep.* **10**, 1 (2020).
27. Litjens, G. *et al.* Deep learning as a tool for increased accuracy and efficiency of histopathological diagnosis. *Sci. Rep.* **6**, 1 (2016).

28. Oyelade, O. N., Ezugwu, A. E. (2021). Ebola Optimization Search Algorithm (EOSA): A new metaheuristic algorithm based on the propagation model of Ebola virus disease. *Artif. Intell.* [Preprint] (2021). Cited 2021 July 13, p. 38. https://arxiv.org/abs/2106.01416
29. Berge, T., Lubuma, J., Moremedi, G., Morris, N. & Kondera-Shava, R. A simple mathematical model for Ebola in Africa. *J. Biol. Dyn.* **11**, 42–74 (2017).
30. Tanade, C., Pate, N. & Paljug, E. Hybrid modeling of ebola propagation. *Proc. IEEE Int. Symp. Bioinf. Bioeng.* **2019**, 204–210 (2019).
31. Cortes, C., Gonzalvo, X., Kuznetsov, V., Mohri, M., & Yang, S. AdaNet: Adaptive Structural Learning of Artificial Neural Networks. *Proceedings of the 34th International Conference on Machine Learning, PMLR*, 70:8748–83 (2017).
32. Negrinho, R., & Gordon, G. DeepArchitect: Automatically designing and training deep architectures. https://arxiv.org/abs/1704.08792 [stat.ML], [Preprint] (2017). Cited 2021 July 13, p. 11.
33. Wang, J., Xu, J., & Wang, X. Combination of Hyperband and Bayesian Optimization for Hyperparameter Optimization in Deep Learning. https://arxiv.org/abs/1801.01596 [cs.CV], [Preprint] (2018). Cited 2021 July 13, p. 11.
34. Huang, S., Li, X., Cheng, Z. Q., Zhang, Z., Hauptmann, A., GNAS: A Greedy Neural Architecture Search Method for Multi-Attribute Learning. https://arxiv.org/abs/1804.06964 [cs.NE], [Preprint] (2018). Cited 2021 July 13, p. 11.
35. Weng, Y., Zhou, T., Liu, L. & Xia, C. Automatic convolutional neural architecture search for image classification under different scenes. *IEEE Access* **7**, 38495–38506 (2019).
36. Erivaldo, F., Junior, F. & Yen, G. G. Particle swarm optimization of deep neural networks architectures for image classification. *Swarm Evol. Comptn.* **49**, 62–64 (2019).
37. Liu, P. *et al.* Deep evolutionary networks with expedited genetic algorithms for medical image denoising. *Med Img Analy* **54**, 306–315 (2019).
38. Krishna, C. S., Gupta, A., Narayan, S., Rai, H., & Manchanda, D., Hyperparameter optimization with REINFORCE and Transformers (2020). Cited 2021 July 13, p. 11. https://arxiv.org/abs/2006.00939
39. Calisto, M. B. & Lai-Yuen, S. K. AdaEn-Net: An ensemble of adaptive 2D–3D Fully Convolutional Networks for medical image segmentation. *Neurl Netwks* **126**, 76–94 (2020).
40. Chen, Z. & Li, B. Efficient evolution for neural architecture search. *International Joint Conference on Neural Networks (IJCNN)* **2020**, 1–7 (2020).
41. Wang, Y., Xu, Y., Tao, D., DC-NAS: Divide-and-Conquer Neural Architecture Search (2020). Cited 2021 July 13, p. 11. https://arxiv.org/abs/2005.14456.
42. Cassimon, T., Vanneste, S., Bosmans, S., Mercelis, S. & Hellinckx, P. Designing resource-constrained neural networks using neural architecture search targeting embedded devices. *Intern. Things* **12**, 1 (2020).
43. Fan, Y. *et al.* Searching Better Architectures for Neural Machine Translation. *IEEE/ACM Trans. Audio. Speech Lang. Proc.* **28**, 1574–1585 (2020).
44. Dai H, Ge F, Li Q, Zhang W, Liu T (2020) Optimize CNN Model for FMRI Signal Classification Via Adanet-Based Neural Architecture Search. *2020 IEEE 17th International Symposium on Biomedical Imaging (ISBI),* pp. 1399–403. https://doi.org/10.1109/ISBI45749.2020.9098574 (2020).
45. Gheshlaghi, S. H. *et al.* Efficient oct image segmentation using neural architecture search. *IEEE Int. Conf. Image Processing (ICIP)* **2020**, 428–432 (2020).
46. Chen, W., Wang, Y., Yang, S., Liu, C., & Zhang, L. You Only Search Once: A Fast Automation Framework for Single-Stage DNN/Accelerator Co-design (2020). Cited 2021 July 13, p. 11. https://arxiv.org/abs/2005.07075
47. Guo, Y., Chen, Y., Zheng, Y., Chen, Q., Zhao, P., Chen, J., *et al.*, Pareto-Frontier-aware Neural Architecture Generation for Diverse Budgets [Preprint] (2021). Cited 2021 July 13, p. 11. https://arxiv.org/abs/2103.00219
48. Zhang, T., Lei, C., Zhang, Z., Meng, X. B. & Chen, C. L. AS-NAS: Adaptive scalable neural architecture search with reinforced evolutionary algorithm for deep learning. *IEEE Trans. Evol. Comp.* **1**, 1 (2021).
49. Hu, L. *et al.* A-DARTS: attention-guided differentiable architecture search for lung nodule classification. *J. Elect. Img.* **30**, 1 (2021).
50. Xu, Y. *et al.* Partially-connected neural architecture search for reduced computational redundancy. *IEEE Trans Patrn Analy Mach Intelg* **2021**, 1 (2021).
51. Ru, B., Wan, X., Dong, X., Osborne, M., Interpretable Neural Architecture Search via Bayesian Optimisation with Weisfeiler-Lehman Kernels. *ICLR 2021 Conference* (2021).
52. Fu, X., Li, W., Chen, Q., Zhang, L., Yang, K., Qing, D. *et al.* NASIL: Neural Network Architecture Searching for Incremental Learning in Image Classification. *International Symposium on Parallel Architectures, Algorithms and Programming, PAAP 2020: Parallel Architectures, Algorithms and Programming,* pp. 68–80 (2020).
53. Lin, M., Wang, P., Sun, Z., Chen, H., Sun, X., Qian, Q., *et al.*, Zen-NAS: A Zero-Shot NAS for High-Performance Deep Image Recognition [Preprint] (2021). Cited 2021 July 13. https://arxiv.org/abs/2102.01063.
54. Liu, X. *et al.* Continuous particle swarm optimization-based deep learning architecture search for hyperspectral image classification. *MDPI Rem. Sens.* **13**, 1082. https://doi.org/10.3390/rs13061082 (2021).
55. Liang, T., Wang, Y., Tang, Z., Hu, G., & Ling, H. OPANAS: One-Shot Path Aggregation Network Architecture Search for Object Detection [Preprint] (2021). Cited 2021 July 13, p. 11. https://arxiv.org/abs/2103.04507.
56. Cai, H., Zhu, L., & Han, S. ProxylessNAS: Direct Neural Architecture Search on Target Task and Hardware. *ICLR 2019;*
57. Polónia, A., Eloy, C. & Aguiar, P. BACH Dataset : Grand challenge on breast cancer histology images. *Med Img Analy* **1**, 1 (2019).
58. Spanhol, F., Oliveira, L. S., Petitjean, C., & Heutte, L., Breast Cancer Histopathological Database (BreakHis). https://web.inf.ufpr.br/vri/databases/breast-cancer-histopathological-database-BreakHis/. Accessed 20 April 2021.
59. Spanhol, F., Oliveira, L. S., Petitjean, C. & Heutte, L. A dataset for breast cancer histopathological image classification. *IEEE Trans. Biomed. Eng. (TBME)* **63**, 1455–1462 (2016).
60. Reinhard, E., Adhikhmin, M., Gooch, B. & Shirley, P. Color transfer between images. *IEEE Comput. Graph. Appl.* **21**, 34–41 (2001).
61. Macenko, M., Niethammer, M., Marron, J. S., Borland, D., Woosley, J. T., Guan, X., et al. A method for normalizing histology slides for quantitative analysis. In: *2009 IEEE International Symposium on Biomedical Imaging. Boston, MA,* pp. 1107–10 (2009).

Author contributions

O.N.O. conceived the initial study and developed the methodology, wrote the code and performed the calculations. A.E.E. supervised the project, contributed in finalizing the methodological approach and provided the funds. Both authors contributed equally in analyzing the results and in writing and revising the paper.

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to O.N.O. or A.E.E.

Reprints and permissions information is available at www.nature.com/reprints.