Multimedia Tools and Applications (2020) 79:15437–15465
https://doi.org/10.1007/s11042-019-7473-z

Detection and classification of pulmonary nodules using deep learning and swarm intelligence

Cesar Affonso de Pinho Pinheiro[1] · Nadia Nedjah[1] · Luiza de Macedo Mourelle[2]

Received: 15 November 2018 / Revised: 23 January 2019 / Accepted: 7 March 2019 /
Published online: 13 March 2019

Abstract

Cancer diagnosis is usually an arduous task in medicine, especially when it comes to pulmonary cancer, which is one of the most deadly and hard to treat types of that disease. Early detecting pulmonary cancerous nodules drastically increases surviving chances but also makes it an even harder problem to solve, as it mostly depends on a visual inspection of tomography scans. In order to help improving cancer detection and surviving rates, engineers and scientists have been developing computer-aided diagnosis systems, similar to the one presented in this paper. These systems are used as second opinions, to help health professionals during the diagnosis of numerous diseases. This work uses computational intelligence techniques to propose a new approach towards solving the problem of detecting pulmonary carcinogenic nodules in computed tomography scans. The applied technology consists of using Deep Learning and Swarm Intelligence to develop different nodule detection and classification models. We exploit seven different swarm intelligence algorithms and convolutional neural networks, prepared for biomedical image segmentation, to find and classify cancerous pulmonary nodules in the Lung Image Database Consortium and Image Database Resource Initiative (LIDC-IDRI) databases. The aim of this work is to use swarm intelligence to train convolutional neural networks and verify whether this approach brings more efficiency than the classic training algorithms, such as back-propagation and gradient descent methods. As main contribution, this work confirms the superiority of swarm-trained models over the back-propagation-based model for this application, as three out of the seven algorithms are proved to be superior regarding all four performance metrics, which are accuracy, precision, sensitivity, and specificity, as well as training time, where the best swarm-trained model operates 25% faster than the back-propagation model. The performed experiments show that the developed models can achieve up to 93.71% accuracy, 93.53% precision, 92.96% sensitivity, and 98.52% specificity.

Keywords Deep learning · Swarm intelligence · Nodule detection · Convolutional neural networks

✉ Nadia Nedjah
nadia@eng.uerj.br

Extended author information available on the last page of the article.

1 Introduction

According to the study presented in [6], cancer is one of the leading causes of death in the world, being the second deadliest cause on the list since the 1940s. It presents statistical data of death cases from all over the world and states the power of cancer, which makes millions of victims every year. In addition to its killing rates, cancer has shown a highly alarming behavior during the last century. As seen in Fig. 1, while all other diseases have had their incidences decreased over the last 50 years, cancer became stronger.

Cancer diagnosis is very difficult. Its cause is often unknown and its treatment expensive and aggressive to the human body. Among the various types of cancer, lung cancer is at the top of the death list. As shown in Fig. 2, taken from [6], lung cancer is by far the deadliest type of that disease. The numbers related to this evil grow alarmingly each year due to the high incidence of smoking and air pollution exposure in large cities. In 2012, the United States National Cancer Institute estimated that 1.6 million deaths are caused by lung cancer and another 1.8 million cases are diagnosed worldwide.

The study, presented in [6], also shows that in 2016, around 8.9 million deaths are attributed to cancer, which is about 5.7 million more than in 1990. In a global average, lung cancer is the largest killer with over 1.7 million deaths in 2016, followed by stomach, colon, rectum, and liver cancer. In [18], it is estimated that at least 12 billions dollars are spent on lung cancer treatments every year and in 2020 a total of 158 billion dollars will be spent to cure cancer.

As in any case of this disease, the stage of lung cancer is measured by its degree of spreading in the human body. This a process called metastasis. The main objective of any cancer diagnosis technique is to detect the disease within its early stages, where the treatment offers much higher chances for a cure. However, early diagnosis is often difficult because of absent symptoms. Currently, according to [1] the most widely used methods for the diagnosis of lung cancer are biopsies, which is an exam that removes part of human tissue for analysis in laboratory and the analysis of computerized tomography, performed by professional radiologists or by automated computer vision systems. There are also researches on developing computer vision models, capable of helping in cancer diagnosis

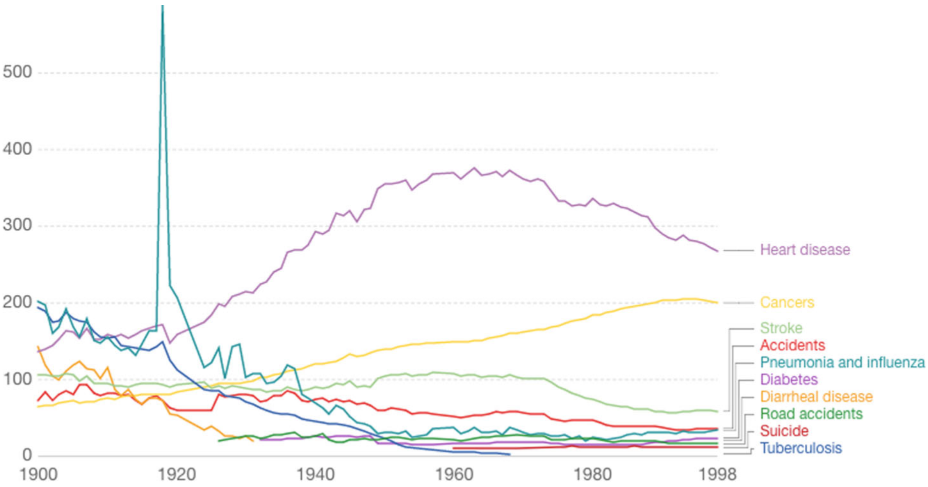


Fig. 1 Death cases from all around the world in the last century. Taken from [6]

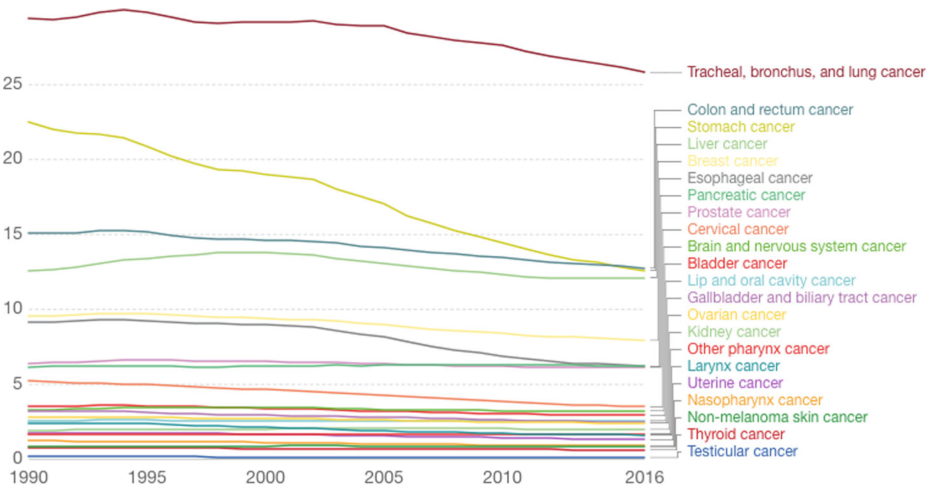


Fig. 2 Cancer death rates by type, per 100,000 world habitants. Taken from [6]

via a biopsy exam. The work presented in [28] introduces a neural network ensemble based model, which is used to identify lung cancer cells in biopsy images.

The present work is designed to be a further contribution to the community of researchers who seek to improve lung cancer detection techniques. Themes such as these incite the need of co-working between professionals of medical and technological areas, for the development of life-saving methods. This is one of the main reasons and inspirations for this research.

Techniques such as the one developed in this work are extremely helpful to medical specialists, as they can, in a relatively short time, achieve higher accuracy and efficiency than human beings. For these issues, computerized diagnostic tools have become increasingly indispensable to medical laboratories. These techniques are called Computer Aided Diagnosis (CAD) systems and offer a second opinion on the patient's condition. These systems are used mostly in radiology diagnostics, where an equipment supplies images of the patient's organs and the diagnosis is given by a specialist. It is important to emphasize that CAD systems should not be used as automated diagnostic tool, but rather as an aiding method for the professional. Since these systems are trained on data classified by radiology experts, a knowledge acquired by a CAD system is of course derived from human knowledge.

Regarding its contribution, this work applies swarm-trained deep neural networks for a more precise classification of lung nodules in CT scans. The results confirm the superiority of the proposed approach. Furthermore, we identify the best swarm intelligent behavior to efficiently train the models used in this application. We show that the training time is 25% faster than that achieved by classical learning algorithms. Thus, we believe that this study advances the state-of-the-art of intelligent models used in CAD systems developed for cancer detection.

This rest of this paper is divided into 6 sections. First, in Section 2, we present the main recent related works. Then, in Section 3, we walk through the theoretical background needed for the development, presenting a quick review on the main topics, such as Deep Learning, Transfer Learning and Swarm Intelligence. Later, in Section 4, we explain the proposed solution for the problem as well as the used methodology. Subsequently, in Section 5, we thoroughly describe the methodology applied during the performed work, from data preprocessing to network training and image classification. There after, in Section 6, we present the results obtained in the executed experiments and discuss the impact of the investigated swarm-training algorithms the precision and efficiency of the classification process. Finally, in Section 7, we draw some conclusions and point out some directions for future work.

2 Related work

The conclusion of this work is largely due to the knowledge acquired from studies related to the problem of detection and classification of pulmonary nodules using computational intelligence algorithms. Studies that use deep learning, neural networks of less complexity or other computational intelligence methods to do the work of detecting cancerous nodules are analyzed.

In [12], a CAD tool that uses deep learning through a convolutional neural network is introduced to detect and classify pulmonary nodules. The work was developed to be part of the Kaggle Data Science Bowl 2017 Challenge competition, which mobilizes researchers to develop models capable of accurately detecting lung cancer. The authors report that the main challenges of this type of application are the difficulty of distinguishing between malignant and benign nodules and the confusion caused by blood vessels that look like nodules in the tomography images. For the training of the model, the LUNA16 (Lung Nodule Analysis 2016) image database is used, which has 888 images of lung tomographies and provides information on nodal positions. Although LUNA16 is quite complete, it does not provide information on the patient's cancer diagnosis, so there was a need for another database, the KDSB17 (Kaggle Data Science Bowl 2017), which has 2101 images with cancer diagnostic labels. The work uses a very common technique used in deep learning, Transfer Learning, which consists of using a pre-trained convolutional neural network model and adjusting only some parameters of its last layers to adapt it to its application. The network chosen by the authors is the ResNet, introduced by Microsoft in [7] and considered by many to be the current state of the art for computer vision applications. This network is modified to adapt to pulmonary tomography images and the model is structured according to the following routine: pre-processing of images, detection of nodules, analysis of malignancy, classification of nodules and diagnosis of cancer. The network is trained on 100,000 iterations, with a learning rate of 0.01 and regularization of weights, classifying images in three classes: (malignant nodule, benign nodule without nodule) and diagnosing patients according to a percentage value of the network output. The performance of the model is measured by the Log-Loss metrics set by the Kaggle Data Science Bowl 2017 Challenge and after submission, the study reached the 41st place in the competition among 1971 competitors, attesting that the methods are really efficient for the application.

Another tool that uses deep learning through a convolutional neural network is introduced to detect and classify pulmonary nodules in [3]. This work was also developed to be part of the Kaggle Data Science Bowl 2017 Challenge competition and used the same LUNA16 and KDSB17 image databases. The main difference between this work and the one developed in [12] is in the networks chosen for the application of Transfer Learning. The authors used the U-Net convolutional network, which specializes in recognizing patterns in computed tomography (CT) scans, which significantly increased the performance of the nodule segmentation phase. After this segmentation phase, the nodules are classified by another existing convolutional network modified with Transfer Learning, able to achieve performances over-human. This work obtained even more expressive results than the one presented in [12], reaching greater accuracy, sensitivity and specificity. Compared to this present study, both works presented followed very similar paths, using convolutional neural networks and Transfer Learning to develop pulmonary nodule classification models. The main difference is that they did not use swarm intelligence algorithms to adjust the network parameters. In the next section, works that use less complex neural networks, when compared to the convolutional networks, for the detection and classification of cancerous nodules in medical images are presented. These works provide alternative techniques that are useful for reducing the computational effort of the application or for optimizing more complex processes.

In [8], the authors present a CAD system using an artificial neural network trained with the Particle Swarm Optimization algorithm to detect and classify carcinogenic nodules in mammography exams. The study uses Mammographic Image Analysis Society (MIAS), which contains 322 mammography images classified as: presence of malignant tumors, presence of benign tumors or absence of tumors, to train the model. The work was conducted according to the following routine: preprocessing of images with median filters, segmentation using Fuzzy C-Means, extraction of attributes such as intensity, texture and shape and classification using the neural network. Due to the scarcity of data, the authors developed a model focused on extracting attributes through image processing techniques. After applying these techniques, the classification is done in two different ways, one using a neural network tuned in the traditional way and the other using a neural network tuned through a particle swarm optimization algorithm (Particle Swarm Optimization - PSO). The approaching that did not use the PSO algorithm obtained 88.6 % accuracy, 72.7 % sensitivity and 93.6 % specificity, while the PSO algorithm reached 95.6% accuracy, 87.2 % sensitivity, and 97.3 % specificity, being higher in all recorded metrics. The results obtained confirm that the use of swarm intelligence algorithms such as PSO to adjust neural network parameters can be a good performance enhancement tool for applications such as this. For this reason, this work served as a good source of information for this study, as it presented good approaches to image treatment and parameter adjustment. It is worth mentioning that the contents of the database LIDC-IDRI is thoroughly explained in [2].

In [16], the authors present a CAD system that uses imaging processing techniques and an artificial neural network to detect and classify carcinogenic nodules on lung CT scans. The imaging techniques are separated in two stages. In the first stage, image binarization is applied, then the images are compared with a threshold value. In the second stage, segmentation and a feature extraction method are applied to extract some important features. Extracted features are used to train the neural network responsible for the classification of cancerous nodules. That work is a good source of comparison for the methods applied in this study, since the model obtained by [16] reaches accuracies over 90% using only image processing and a Feed-Forward Multilayer Perceptron neural network for classification. The authors of the study used a different approach than those presented until then, as they divided the images in half to apply the techniques separately to each lung. Among the image processing techniques used, the following stand out:

- Conversion to a grayscale image, normalization between 0 and 1, noise reduction
- Image Binarization
- Removal of unnecessary parts
- Segmentation by morphology processes

After these steps, the resulting images are analyzed and provided 33 input attributes for the classifier network. These attributes are calculated through mathematical relationships between parts of the lung, pixel densities, etc. As a result, the model produces a classification that identifies cancerous nodules and locates them in each lung. The study of [16] introduces a simple yet very efficient approach that can be used in more complex networks such as deep learning.

In [15], the authors present a CAD system that uses the LIDC-IDRI imaging database to apply imaging processing and a classifier that operates with Region Growing Algorithm to detect carcinogenic pulmonary nodules in computed tomography images. At the beginning of the procedure, the images go through the following steps: median filters processing, image binarization, edge detection, removal of unnecessary parts and application of the Flood Fill algorithm. After preprocessing, the images are subjected to the region growth algorithm, which verifies regions of similar pixels and performs a selection of promising regions for the desired characteristics. At the end of some iterations, the algorithm returns a binary image that has only the nodules. In this study, the authors introduce a sequence of very useful methods for applications as is.

In [22], the authors present a CAD system that uses grouping procedures with different Fuzzy logic algorithms and a support vector machine classifier (SVM) to detect and classify carcinogenic nodules in computed tomography images of the lungs. The authors chose to use the LIDC-IDRI image database to train their models, obtaining high robustness results. In that work, images undergo a preprocessing routine similar to that applied in [8], but three different grouping methods are applied, the Fuzzy C-Means, the Fuzzy- Possibilistic C-Means and Weighted Fuzzy-Possible C-Means. After the preprocessing and feature extraction phases, the images are submitted to an SVM classifier. For best results, the SVM classifier is trained with 3 different kernel functions, one linear, one polynomial and a radial basis one. From the works presented so far, this one obtained the worst performance with an accuracy of 80.36%, a fact that may be related to the difficulty of finding an ideal kernel function for an SVM classifier.

With a similar deep learning analysis, the authors in [28] introduce a neural network ensemble model, which is used to identify lung cancer in a different manner. The developed model is trained to identify cancerous cells in biopsy images, which is another interesting approach towards solving the same problem.

Another worth mentioning work is presented in [13]. It develops a CAD system based on a neural network tuned with genetic algorithms.

3 Theoretical background

The accomplishment of this work is due to the combination of a set of computational intelligence techniques applied to a big database of medical lung computed tomography images. The most important techniques applied are described in the following three sections.

3.1 Deep learning

Deep learning is a kind of machine learning, where highly complex neural network models are designed to mimic the human brain's great ability to recognize patterns. Like most common neural networks, deep learning networks are structured in layers, which perform mathematical operations to adjust their parameters and learn to perform some task through training. The main difference between common and deep learning neural networks is the number of layers present in each model. While the most common networks usually operate with no more than 5 or 6 layers, deep learning networks are structured in architectures that can exceed 150 layers, as some of the networks used by companies such as Google™in [23] and Microsoft™in [7] for their image search engine.

Pattern recognition in images and videos can be considered one of the most widely used applications in recent years. This is due to the great evolution of computational vision branch and the development of neural networks called Convolutional Neural Networks. These networks are generally used in deep learning models for pattern recognition and classification applications in images and videos. Their aim is to simulate the process of pattern recognition in the human brain by creating regions specialized in recognizing certain types of morphological characteristics, such as lines, circles, etc.

A convolutional neural network usually consists of: An input layer that receives the images, a series of layers that perform image-processing operations with feature mapping and the last phase, containing a neural classification network, which receives the mapping of features and outputs the result of classification. In mapping layers, there are three main operations applied:

- Convolution: Operation based on a linear operation scan, made by an array of features throughout the image. This feature matrix carries the shape of the pattern to be identified according to a numerical distribution of values. During this operation, the image is swept by the feature matrix, which evaluates position by position the degree of similarity between the region analyzed with and the pattern used in the matrix. These linear operations result in a new matrix, smaller than the original image, which contains the similarity map of between the image and the pattern on search.
- Max-pooling: This operation is focused on reducing the dimensions of the feature maps, taking advantage of the regions that obtained the highest affinities with the pattern analyzed by the convolution matrix. During this operation, a small array of max-pooling, usually 2x2 and without values, is passed by the generated convolution map. At each iteration, the largest value within the 2x2 space is extracted to compose the next feature map. This way, it can be ensured that only the highest affinities are being considered.
- Activation layers: These layers apply a given activation function. In most cases, the Rectified Linear Units (ReLU) is used. It applies a function that aims to reduce the linearity of convolutional operations, since it is known that models based only on linear operations can present learning problems. The ReLU operation resembles a mathematical module operation, where only negative values are affected. However, negative values are transformed into zeros, that is, any negative pixel value that may appear in convolution or max-pooling operations will be converted to zero.

In addition to these layers, convolutional networks contain a layer called the fully connected layer, which is nothing but a common neural network that receives an input from the output of the feature maps and analyzes it to give the final verdict about classification. Figure 3 shows examples of different feature maps analysis on the same nodule image. Each map is focused on finding a certain type of feature and the purple brightness represents the level of matching, meaning that the most bright maps are the ones that allow finding the highest similarities between their wanted feature and the scanned image.

3.2 Transfer learning

Applications like that require neural networks with highly complex architectures, a fact that becomes a major drawback for independent developers to develop such models. To fix this problem, a technique called Transfer Learning is developed. In this technique, models start

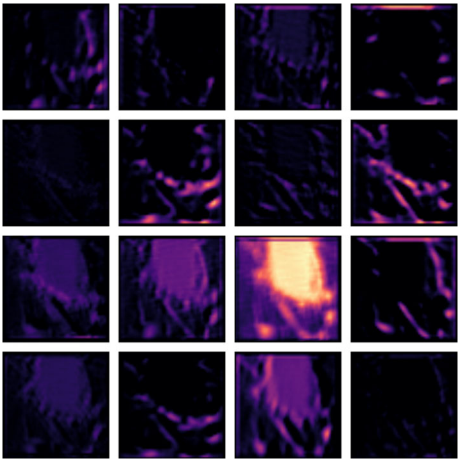


Fig. 3 Feature maps obtained by different filters

from a convolutional neural network skeleton, which has already been trained to recognize certain types of patterns, so the whole work focuses on adjustments to fit the model to the application.

In Transfer Learning, a previously trained convolutional network is chosen to recognize patterns similar to those in the studied application. From this point, only the fully connected layer of the chosen network must be remodeled to be able to classify the application data. To achieve this result, the training images are fed into the mapping layers of the original network and the resulting maps are inserted into a fully connected neural network with random weights. A common fully-connected network training procedure is then started. At the end of this training, a new convolutional network is ready to classify the data of this new application.

In this work, a convolutional neural network is fitted through Transfer Learning to recognize and classify carcinogenic nodules in computed tomography images of lungs. The network chosen for the application of Transfer Learning is the U-Net [21], a network specialized in medical imaging pattern recognition, as well as other network configurations varying the filter size and the activation function attempting at higher network accuracy rate.

3.3 Swarm intelligence

Swarm intelligence is the term used to name a family of computational intelligence algorithms that have the common characteristic of simulating a population of unintelligent individuals who, based on certain laws of coexistence, manages to produce high complexity behaviors. The main applications of these algorithms are aimed at optimizing processes, such as finding optimal values for mathematical functions or finding the best paths for a navigation system. The efficiency of these processes is based on the observation of the behavior of animals or microorganisms, which can accomplish complex tasks with simple actions when living in a community or ecosystem.

The involvement of swarm intelligence in this project is tightly linked to the optimization of neural network parameters. In a common training, network weights are adjusted by applying algorithms such as Back-Propagation and Gradient Descent, which change the values of each weight proportionally to the error obtained on prediction. Algorithms such as

Back-Propagation and Gradient Descent generally produce quite satisfactory results, however, they are more likely to be trapped in local minimum regions. As a solution to this problem, swarm intelligence algorithms are used to perform the training of these networks. The training of neural networks via swarm intelligence generally works with the following procedures:

For the application of a neural network training via swarm intelligence, in general, one must first define the number of weights and layers of the network, stipulate a search space for the values of the weights, define a fitness function to be analyzed and to choose a swarm algorithm to be tested. From this basis of definitions, one must initiate a swarm of particles where each particle represents the set of all weights of the classification network, then the performance of each particle is evaluated so that they can be moved in the search space of according to the rules of the swarm algorithm until they find a good enough solution for the application. These steps are presented by Algorithm 1.

Algorithm 1 Swarm training.

```
$e$ = total number of epochs;
$n$ = swarm size;
Form first swarm;
for $i$ := 1 to $e$ do
    for $j$ = 1 to $n$ do
        Select the $j$-th particle to be used as the set of weights for the classification network;
        Evaluate the fitness of that particle through the mean square error during the training;
    end for
    Apply the swarm logic to generate a new swarm;
end for
Select the best particle as the final set of weights for the network;
```

Actually, the use of swarm-trained models in the development of new tools for medical applications is already a reality. In [26], the authors present a Chaotic PSO model focused in classifying brain magnetic resonance images. Another interesting work is presented in [27], wherein the authors present a PSO-trained model for Alzheimer's disease detection.

This work makes use of this swarm intelligence training technique to provide a new approach to the classification problem of lung cancer nodules. Understanding the algorithms of this techniques is a fundamental part of this work. The performance of different swarm algorithms will be tested. For this application, seven of the most widely used swarm intelligence algorithms are selected:

– *Particle Swarm Optimization* (PSO) [10]: Optimization algorithm developed by *Kennedy* and *Eberhart* in 1995. It is inspired by the behavior of flocks of birds. The search for food and the interaction between birds throughout the flight are modeled as the optimization mechanism.
– *Artificial Bee Algorithm* (ABC) [9]: Optimization technique that simulates bees foraging behavior. It has been successfully applied to various practical problems. The ABC was proposed by *Karaboga* in 2005.
– *Bacterial Foraging Optimization* (BFO) [19]: The Bacterial Foraging Optimization Algorithm is inspired by the group-foraging behavior of bacteria such as E. Coli and M.

Xanthus. Specifically, BFO is inspired by the chemotaxis behavior of bacteria, which perceive chemical gradients in the environment and move close to the nutrients.

- *Firefly Algorithm* (FA) [20]: This optimization algorithm is based on the behavior of fireflies. These insects can produce small flashes of light to attract fireflies of the opposite sex or small prey. The light signal also serves to signal to predators that these fireflies have a bitter taste.
- *Firework Optimization Algorithm* (FOA) [24]: Fireworks explode in a manner similar to how individuals look for optimal solutions in swarm algorithms. As a swarm algorithm, the fireworks algorithm consists of 4 steps, the explosion operator, the mutation operator, the mapping rule, and the strategy selection.
- *Harmony Search Algorithm* (HSA) [5]: This algorithm imitates the technique used by Jazz musicians to find the correct harmony of each song. It is based on the ability of musicians to adapt quickly to the sound of other musicians of the band to keep the melody pleasant.
- *Gravitational Search Algorithm* (GSA) [11]: This algorithm is based on the laws of gravitation and mass attraction. Its behavior resembles the PSO because it uses particles that interact through laws of attraction and relationship.

The use of swarm intelligence algorithms in the training of machine learning methods for medical purposes is already being explored by many researchers, as these methods tend to exhibit great performances. One interesting application of this methods, apart from cancer detection, is presented in [25], where the authors use a feed-forward neural network optimized with a hybrid form of Particle Swarm Optimization and Artificial Bee Colony algorithms to detect brain anomalies.

4 Proposed solution and methodology

In general, this application deals with a new approach to the pulmonary node detection and classification procedure and aims at evaluating the performance of the training of convolutional neural networks by means of swarm intelligence optimization techniques, comparing it with the same network, trained using existing classical algorithms, such as gradient descent. Fo this purpose, a database of lung computed tomography (CT) images, a convolutional neural network (U-Net), and seven swarm intelligence algorithms are explored.

The methodology starts with the data preprocessing. The CT images are gathered and subjected to routines such as data augmentation and nodule segmentation. In order to compose the batch of swarm intelligence algorithms, seven of the most used algorithms are selected to train the classification model. The deep network learning model is built over the U-Net, which is a convolutional neural network dedicated to biomedical image segmentation. Given the swarm algorithms and the U-Net, the model is trained using transfer learning via each of the seven swarm algorithms and also using the classical methods, which are the back-propagation and the gradient descent. During that training process, the main imaging process layers of the U-Net are maintained and the final classification neural network is remodeled to be ready to classify lung nodules. For the first experiments, the model is trained using back-propagation under several configurations of hyper-parameters aiming at finding an optimal one for this application to set up a common network structure for the following experiments with swarm algorithms. Furthermore, aiming at extending the search for higher performance, some U-Net parameters are also tested. These are the max-pooling matrix size and the activation function. This extra searching is done to check whether the model performs better with bigger or smaller max-pooling matrices and to measure the impact of using the Hyperbolic Tangent activation function instead of the ReLU function. At the end of the experiments, the performance of each model is analyzed regarding the accuracy, sensitivity, precision, specificity as well as the overall training time. With the obtained results, we aim at demonstrating the superiority of the swarm-trained models over the back-propagation models. Moreover, we aim at selecting the best model for this application. Taking into account the nature of the problem, the best model should be the one with the highest accuracy and sensitivity, as it would exhibit the highest hit rates and would make fewer mistakes on classifying a cancerous patient as non-cancerous.

5 Implementation issues

In this section, all the steps and procedures applied during the construction of the model used for the application of lung tumor recognition are explained. The methodology begins with obtaining the training data, consisting of thousands of medical images of lung CT scans. These data went through several processing techniques in order to increase the performance of the classifier model and reduce the computational effort required to work with them. After this phase of preprocessing, the assembling of the classifier started, using machine learning tools based on Python programming language and the use of a convolutional neural network for segmentation of medical images called U-Net [21].

As previously described, the model is tested under 8 circumstances, simple classification using the trained model and classification using 7 models trained with swarm optimization algorithms. The results of each approach are then compared statistically to determine the actual effectiveness of the swarm optimization methods in this type of application.

5.1 Data gathering

After all the bibliographic research, the LIDC-IDRI medical image database is chosen to train the classifier model. This database is provided by the Cancer Imaging Archive and contains approximately 250,000 medical images of CT scans of healthy and cancerous lungs. The images from this database are analyzed by four radiologists and the diagnoses given by each of them are contained in the note files, which are provided together with the images. Detailed information about this database can be found in [2].

The data from the database were collected in a project that lasted about 8 years, analyzing thousands of patients. As a result, this project generated 125 gigabytes of information, which is available for download at the Cancer Imaging Archive website. In addition to LIDC-IDRI, another complementary database is used in this work, which is the LUNA16 database. It originated from the LIDC-IRDI and generated by a prospecting work focused on selecting only images that contained complete information about the main attributes analyzed in LIDC-IRDI. The LUNA16 database is formed only by the richest information in the LIDC-IRDI, to ensure a more efficient training for the model. It is important to emphasize that this database is part of a classified data set, which allows supervised learning. Figure 4 shows an example of an image from LUNA16 database with a small early-stage nodule.

5.2 Data preprocessing

Medical images are generated from a standard protocol called DICOM (Digital Imaging and Communications in Medicine). In addition to the protocol, the file extension *.dicom* is used

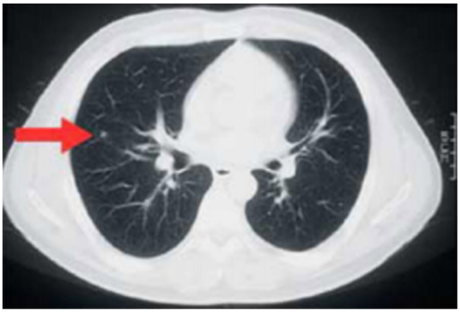


Fig. 4 CT Scan with a small (5mm) early-stage nodule

for all radiology and tomography devices. This extension contains not only the images of the exam but also the information about any relevant notes about the patient, exam, schedule, medical team, etc. This information is called meta-data and is usually organized into files of Extensible Markup Language (XML), a markup file used to structure data in a simplified way.

The first routine for data preprocessing is the organization of images and the creation of lists of files. For this purpose, some python scripts are applied to read the DICOM files and to relate the multidimensional images to the annotations, creating two types of files. The content of each image is saved in a file of type *.raw*, an image storage format generally called digital negative, because it loads all the information without any type of compression and, therefore, is suitable for applications like this. After a *.raw* file is generated, the content of this image's meta-data, such as annotations and diagnostics, is saved to a file of type *.mhd*. The *.mhd* files are widely used for storing medical image annotations. As it is a three-dimensional scanning test, each scan consists of about 200 slices and each slice is a square image of 512 pixels in size.

In radiology, the use of the Hounsfield Units (HU), which measures the radio-density of the tomography at one point, is very common. With this metric, it is possible to detect different chemical compositions and thus identify the composing elements, such as bones, air, water, fabrics, etc. HU values of the main components found in the application exams are then raised. Based on this measure, a routine is applied to convert the values of each pixel of each image to HU values. Another important operation is the standardization of the distances between slices of each scan. Throughout the creation of the image database, different scanning devices are used and, with this, the generated images have divergences regarding the distance between slices. According to related works, this difference may be a problem during the training of the convolutional neural network model. Hence, a routine is applied to standardize the distances by staggering, which actually stipulates a distance of 1 millimeter for each slice of the scan.

The network could be trained using all points of each image, however, this would increase exponentially the computational effort. To optimize the operation of the model, images are generated based on square cuts of size 50×50 pixels incident around the coordinates pointed out in the annotations on the image database. Figure 5 shows an example of these cropped images.

At this point, the set of images is ready to feed the model, but the ratio between classes is still a potential problem for training. In general, the dataset has 551065 annotations, with only 1351 being classified as nodules. Faced with this fact, negatively annotations are randomly reduced and data augmentation routines are applied to generate new positively classified data. The applied routines are the rotation of the images by 90 degrees and the horizontal inversion followed by a vertical one. After these operations, the proportion between

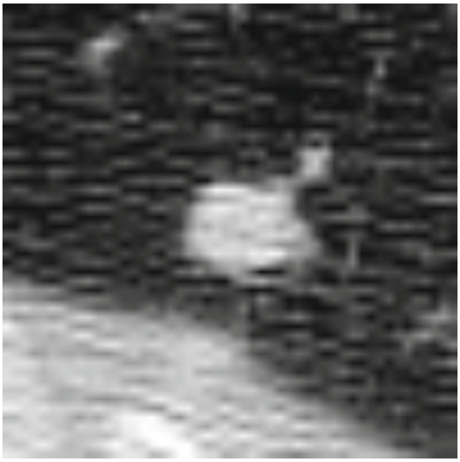


Fig. 5 Nodule cropped image

classes is around 80% negative and 20% positive, which may not be ideal, however, according to [14] increasing the positive class too much may lead to variance problems. With class balancing, data preprocessing is finalized.

5.3 Model creation

At this point, the images are ready to be used on the model. So the assembling of the model began using the U-Net [21] convolutional neural network. Its structure is shown in Fig. 6. This network processes images in three different stages:

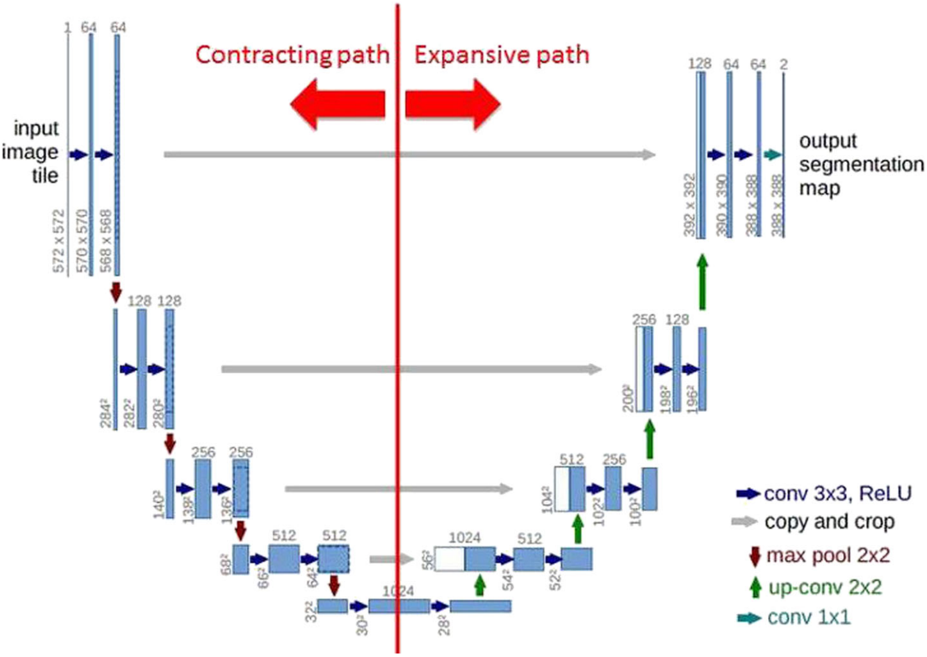


Fig. 6 U-Net architecture. Taken from [4]

- Obtaining the characteristics: In this first phase, the image is scanned with several convolutional filters to detect the degree of pertinence of the image with the features matrices.
- Preventing over-fitting: A phase focused on applying dropout operations.
- Feature Placement: In this phase, feature maps are extended back to the original size to result in a final map that can tell what exists in the image and where each feature is.

Briefly, the first phase is called the Contracting Path, it is concerned about finding what exists in the image. This phase has 4 blocks, where each block is composed of two 3×3 Convolution Layers together with the activation function with batch normalization and a 2×2 Max-Pooling layer. The amount of feature maps doubles after each max-pooling, starting with 64 maps for the first block, 128 for the second, and so on. With this architecture, this contracting path is able to capture the context of the input image. This coarse contextual information is then transferred to the next phase. After this feature detection phase, the resulting data is submitted to a phase called The Bottleneck Phase, which is built with 2 convolutional layers with a batch normalization with dropout. Dropout simply 0s a certain amount of layer activations based on probabilistic threshold decision making. This simple operation is able to force the network to be redundant, as it must be able to classify data correctly even without counting on some of its activations. With these operations, this technique becomes a good ally against over-fitting. The third and last phase is the Expanding Phase, focused on finding the precise localization of each feature, combined with contextual information from the contracting path. This phase is also composed of 4 blocks, where each block contains a deconvolution layer with stride 2, concatenation with the corresponding cropped feature map from the contracting path, and two 3x3 Convolution layer + activation function with a batch normalization. Besides the biomedical imaging background, the main advantages of using the U-Net are:

- It combines the information from the contracting path with the expanding path to finally obtain a general information combining localization and context, which is necessary to predict a good feature map.
- It has no dense layer, so images of different sizes can be inputted. The only parameters to learn on convolution layers are the kernels and their size.

At the end of the mapping procedure, the network provides a feature map ready to be analyzed by a fully connected neural network. This map is used as an input layer for a fully connected neural network with 100 units. This network reads the input data and outputs a vector in the two-dimensional Softmax format, that is, a vector with the pair (P, N) where P and N are values between 0 and 1 and the sum $P + N$ is equal to 1. In this vector, P represents the probability of a positive carcinogenic nodule, and N case of a negative, non-carcinogenic case. Thus, the classification will be given by comparing values between P and N, where the highest value will be the winner and will define the classification. A full explanation about the U-Net and its coding can be found at [4].

5.4 Training and testing

With the ready-made model and a processed dataset, the training routines could be established. Due to the great amount of data required for model training, batches are generated for separate analyzes, since loading of all data into the machine memory is not always possible. For this blending operation, the data are randomly separated into hdf5 files, a format widely used in applications like this since it can easily store and operate data and meta-data.

The training set has 6878 images, all of them properly classified by the specialist radiologists. Using training routines, the validation set of images is selected, resulting in 1623 images. With the training and test sets ready to be used, the first procedure applied consists of training without swarm intelligence, that is, using back-propagation techniques and gradient descent. For the choice of hyper-parameters, such as the learning rate and the number of training epochs, possible values are stipulated for each one and 100 experiments are executed for each pair. In each experiment, four performance metrics are considered and their explanations are shown below, wherein TP represents the number of true positive classifications, TN the number of true negative classifications, FP the number of false positive classifications and FN the number of false negative classifications:

- Accuracy: This represents the rate of hits among all classifications, as shown in (1)
$$\frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$
- Precision: This is the rate of positive hits among those classified as positive, as shown in (2).
$$\frac{TP}{TP + FP} \tag{2}$$
- Specificity: This is the rate of negative hits among those that are negative, as shown in (3).
$$\frac{TN}{TN + FP} \tag{3}$$
- Sensitivity: This is the rate of positive hits among those who are positive, as shown in (4).
$$\frac{TP}{TP + FN} \tag{4}$$

Table 1 Impact of the hyper-parameters of the fully-connected layer on the used metrics

Case	Epochs	Learning rate	Accuracy	Precision	Specificity	Sensitivity
C_1	30	0.0001	82.34	82.22	82.73	81.41
S_2	30	0.001	84.58	84.27	85.16	83.61
S_3	30	0.01	86.92	86.55	87.01	86.87
S_4	30	0.1	87.79	87.62	88.66	86.86
S_5	100	0.0001	88.87	89.01	90.40	89.51
S_6	100	0.001	90.88	90.86	91.53	90.45
S_7	100	0.01	90.54	90.17	92.43	90.89
S_8	100	0.1	91.07	91.05	93.26	91.64
S_9	300	0.0001	92.31	91.96	94.90	91.20
S_{10}	300	0.001	92.44	92.29	96.62	91.48
S_{11}	300	0.01	92.23	91.02	94.81	91.21
S_{12}	300	0.1	90.45	90.13	93.86	90.12
S_{13}	1000	0.0001	91.89	91.93	92.53	91.31
S_{14}	1000	0.001	91.94	91.60	92.02	91.74
S_{15}	1000	0.01	90.23	90.17	91.65	89.65
S_{16}	1000	0.1	89.87	89.58	90.90	89.16

Table 1 presents the mean results obtained by each pair of hyper-parameters between the values of 30, 100, 300 and 1000 for the number of epochs and 0.0001, 0.001, 0.01 and 0.1 for the learning rate.

Figure 7 shows a graphical representation of that hyper-parameter search, where it is possible to observe that the model that operates in 300 epochs with a learning rate of 0.001 presents, on average, a slightly better result in accuracy and precision, and a significantly better performance in specificity. From these results, that configuration is the chosen one for this application. After this performance check on the model that operates without swarm intelligence, the first implementations of the swarm optimization algorithms are started.

5.5 Swarm-based training

After the tests performed in the model without swarm intelligence, it can be concluded with a detailed inspection that the weights of the fully connected layer of the network, in general, presented values between −2 and 2. This range has become a good metric to be applied as a search space in the swarm algorithms. Thus, a granularity of 0.1 is stipulated to obtain 41 points in this range. Since the fully-connected layer has 100 units, each particle of the swarm is designed as a vector with 100 coordinates.

For the training with swarm intelligence algorithms, an amount of 300 particles (or individuals) per swarm and an amount of 300 iterations per experiment are stipulated to match the best result obtained with the reference model. It is important to emphasize that the algorithms tested are structured with the star topology, that is, all the individuals communicate with any other individual of the swarm when required. The swarm intelligence based training process is exemplified in Fig. 8, where each particle provides a different configuration for the weights of the fully connected layers, which acts as a neural classifier network.

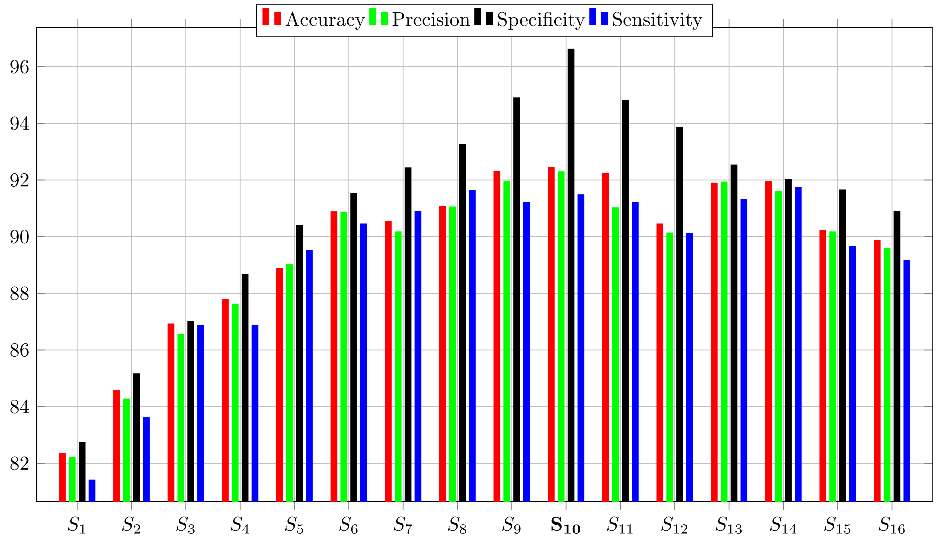
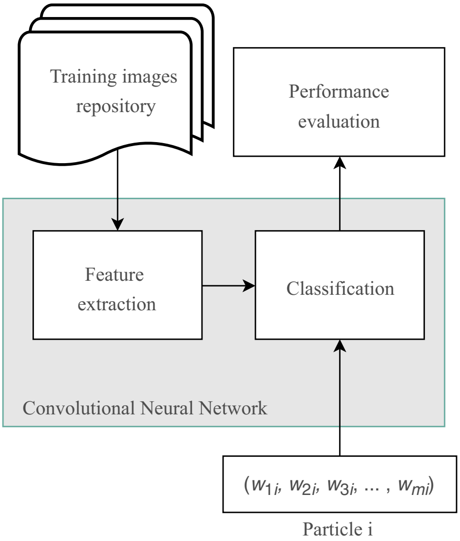


Fig. 7 Impact of the number of epochs and the learning rate

Fig. 8 Network training using the swarm of particles


Training images repository
Performance evaluation
Feature extraction
Classification
Convolutional Neural Network
$(w_{1i}, w_{2i}, w_{3i}, \dots, w_{mi})$
Particle i

In addition to these universal parameters, each swarm algorithm has its own specific parameters. In order to determine the values of each parameter, the most used and recommended values are searched in the literature. In the sequel, sketch the descriptions and values of each parameter used by the algorithms tested.

For the Bacterial Foraging Optimization algorithm, the parameters used are the chemotaxis step, which determine the movement of bacteria, the maximum distance of navigation, the size of each step and the probability of elimination of a bacterium during the operation. The configurations of these parameters are shown in Table 2.

For the Firework Optimization Algorithm, the parameters used are the number of normal fireworks and the number of Gaussian ones. The settings of these parameters are shown in Table 3.

For the Firefly Algorithm, the parameters used are mutual attraction index, light absorption, 2 randomization parameters, denominated α_1 and α_2, and 2 parameters to fit a Gaussian curve. The values of each parameter are shown in Table 4.

For the Harmony Search Algorithm, the parameters used are the pitch adjustment rate, the harmony consideration rate, and the bandwidth. The configurations of these parameters are shown in Table 5.

For the Particle Swarm Optimization, the parameters used are the inertia coefficient and the velocity change constants, which are the cognitive and social coefficients. The settings of these parameters are shown in Table 6.

Table 2 Hyper-parameter setting for the bacterial foraging optimization

Parameters	Value
Chemotaxis steps	5
Navigation distance	2
Step size	1
Elimination probability	0.05

Table 3 Hyper-parameter setting for the firework optimization algorithm

Parameters	Value
Number of regular fireworks	50
Number of Gaussian fireworks	50

Table 4 Hyper-parameter setting for the firefly algorithm

Parameters	Value
Mutual attraction index	1
Light absorption index	1
First randomized parameter	1
Second randomization parameter	0.1
First Gaussian parameter	0
Second Gaussian parameter	0.1

Table 5 Hyper-parameter setting for the harmony search algorithm

Parameters	Value
Pitch adjustment rate	0.5
Harmony consideration rate	0.5
Bandwidth	0.5

Table 6 Hyper-parameter setting for the particle swarm optimization

Parameters	Value
Inertia coefficient	0.49
Cognitive coefficient	0.72
Social coefficient	0.72

Table 7 Average accuracy performance for both activation functions and all max-pooling matrix sizes

	ReLU				Hyperbolic tangent			
Algorithm	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
	2×2	3×3	4×4	5×5	2×2	3×3	4×4	5×5
PSO	**93.71**	**93.21**	**92.95**	**92.92**	**93.26**	**92.89**	**92.71**	**92.27**
GSA	93.43	93.12	92.61	92.56	93.09	92.76	92.62	92.34
HSA	93.11	93.01	92.82	92.47	93.08	92.77	92.59	92.22
ABC	92.87	92.38	92.32	92.33	93.02	92.14	92.30	91.63
BFO	92.51	92.46	92.19	91.47	92.51	92.29	91.41	90.96
NSO	92.80	92.61	92.60	92.46	92.82	92.61	92.31	92.13
FA	91.56	91.11	90.69	89.92	91.48	90.81	90.23	89.10
FWA	91.06	90.63	90.50	90.51	90.98	90.27	90.47	90.48
Average	**92.63**	**92.32**	**92.08**	**91.83**	**92.53**	**92.07**	**91.83**	**91.39**

It is worth mentioning that the Artificial Bee Algorithm and the Gravitational Search Algorithm use only the universal hyper-parameters. Each one of the listed optimization algorithms is used to train and test the model in 100 experiments, monitoring the same 4 performance indicators (accuracy, precision, sensitivity, specificity) and using the mean square error as the solution quality evaluation function. Furthermore, to evaluate possible improvements in model performance due to changes in the U-Net network parameters, the experiments are also applied using versions of U-Net with different max-pooling matrices sizes: 3×3, 4×4 and 5×5. Experiments were also conducted using the hyperbolic tangent function instead of ReLU as a function of the convolutional network activation. The results of the experiments and the comparison of performance between the methods will be presented in Section 6.

6 Performance results

After completing all experiments, the performance indicators of each algorithm are computed and compared to choose the best method(s) and to prove that swarm algorithms can produce better results than the traditional training method for convolutional neural networks in applications such as this. Despite the high efficiency, obtaining results is a computationally expensive operation, due to the volume of data needed to obtain a satisfactory performance and to the very nature of the procedure to obtain the models, since each iteration must completely test a network classifier with a training set.

Other test factor included in the experiments is the use of the hyperbolic tangent activation function instead of the ReLU in the convolutional network. This test is motivated due to comparing the actual efficiency of both functions in an application like this. Although the ReLU function is the most used in deep learning models and especially in convolutional neural networks, in this experiment the hyperbolic tangent function is tested in combination with a swarm trained model. The models are trained and tested 100 times under each of the following conditions:

- ReLU activation function and 2×2 max-pooling matrix
- ReLU activation function and 3×3 max-pooling matrix
- ReLU activation function and 4×4 max-pooling matrix

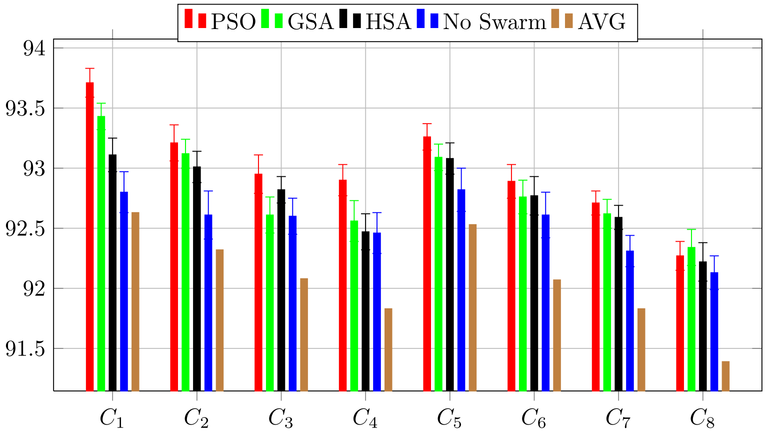


Fig. 9 Accuracy performance of top 3 algorithms, no swarm model and average of all algorithms

Table 8 Average precision performance for both activation functions and all max-pooling matrix sizes

	ReLU				Hyperbolic tangent			
Algorithm	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
	2×2	3×3	4×4	5×5	2×2	3×3	4×4	5×5
PSO	93.09	92.27	92.05	92.04	92.98	91.66	91.50	91.23
GSA	**93.53**	**93.61**	**93.42**	**93.51**	**93.03**	**92.77**	**92.61**	**92.10**
HSA	93.36	92.56	91.95	91.92	93.35	91.85	92.03	91.15
ABC	91.99	91.24	91.33	91.24	91.97	90.86	91.32	91.24
BFO	91.16	91.20	90.73	90.22	90.45	90.82	90.53	89.51
NSO	92.29	92.19	91.74	91.07	92.04	92.27	91.50	91.02
FA	92.25	91.45	91.38	90.97	91.67	91.05	90.83	90.36
FWA	92.32	91.75	91.52	90.80	92.43	91.35	91.13	90.81
Average	**92.56**	**92.08**	**91.71**	**91.45**	**92.24**	**91.59**	**91.49**	**91.04**

- ReLU activation function and 5×5 max-pooling matrix
- Hyperbolic Tangent activation function and 2×2 max-pooling matrix
- Hyperbolic Tangent activation function and 3×3 max-pooling matrix
- Hyperbolic Tangent activation function and 4×4 max-pooling matrix
- Hyperbolic Tangent activation function and 5×5 max-pooling matrix

To present the achieved results, 4 tables are generated with the average performance of each model under each condition, wherein the value shown in bold face indicate the best achieved performance for the considered case-studies. Table 7 shows the average accuracy for every model, where NSO denotes the case that does not use any swarm optimization technique. Therein, the PSO model is highlighted, as it achieved the best accuracy performance.

In a detailed analysis, the top three swarm-trained models, the non-swarm model, and the average performance of all tested models are compared for each performance metric. Figure 9 presents a graph regarding this analysis over the accuracy metric, where it is possible to confirm the PSO superiority.

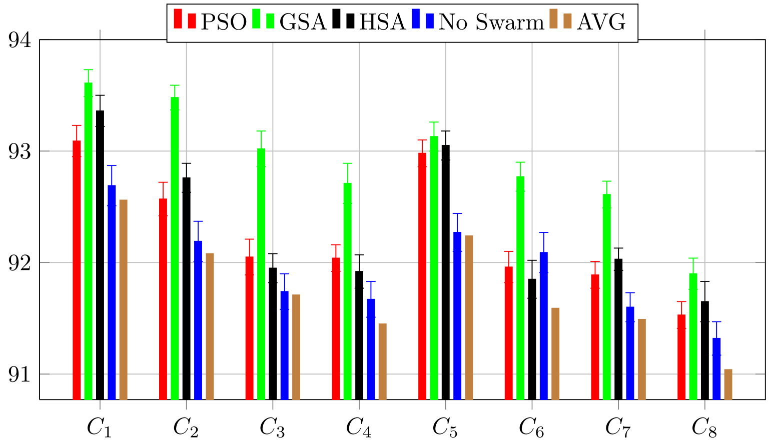


Fig. 10 Precision performance of the top 3 algorithms, no swarm model and average of all algorithms

Table 9 Average sensitivity performance for both activation functions and all max-pooling matrix sizes

	ReLU				Hyperbolic tangent			
Algorithm	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
	2×2	3×3	4×4	5×5	2×2	3×3	4×4	5×5
PSO	**92.96**	**92.42**	**91.79**	**91.72**	**92.32**	**92.05**	**91.74**	**91.71**
GSA	92.51	92.32	91.71	91.75	92.05	92.25	91.74	91.08
HSA	92.33	91.66	91.78	90.99	91.59	91.45	91.59	90.51
ABC	91.88	91.12	90.28	90.19	91.71	91.08	90.04	89.74
BFO	90.42	90.34	90.24	90.24	90.24	89.64	89.66	90.32
NSO	91.48	91.10	90.48	89.64	91.10	90.92	90.61	89.72
FA	91.01	90.48	89.96	89.64	91.16	90.61	90.10	89.27
FWA	91.48	90.90	90.32	90.32	90.75	90.23	89.80	90.30
Average	**91.76**	**91.29**	**90.82**	**90.56**	**91.37**	**91.03**	**90.66**	**90.33**

Table 8 presents the average precision for every model. Therein, the GSA model is highlighted as it achieved the highest precision performance.

Following that same analysis, Fig. 10 shows the precision performances, wherein it is possible to confirm the GSA superiority.

Moreover, Table 9 shows the average sensitivity for every model. Therein, the PSO model is again highlighted as it achieved the best sensitivity performance.

Furthermore, on Fig. 11, the sensitivity performances are presented, wherein it is possible to confirm the PSO superiority.

Finally, Table 10, presents the average specificity for every model. Therein, the HSA model is highlighted, as it achieved the best specificity performance. Following this analysis, Fig. 12 shows the specificity performances of these top three models, the non-swarm model and the average of all models.

One of the factors observed during the experiments is the influence of the size of the max-pooling matrices. It is possible to observe a decrease in the performance metrics as the

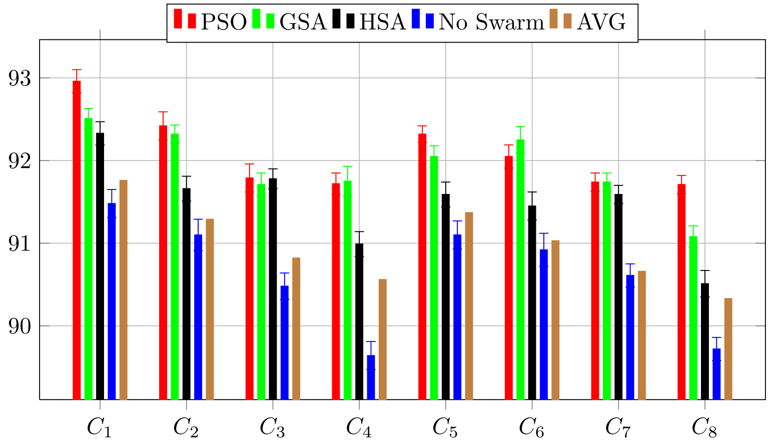


Fig. 11 Sensitivity performance of the top 3 algorithms, no swarm model and average of all algorithms

Table 10 Average specificity performance for both activation functions and all max-pooling matrix sizes

	ReLU				Hyperbolic Tangent			
Algorithm	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
	2×2	3×3	4×4	5×5	2×2	3×3	4×4	5×5
PSO	97.71	97.15	97.08	97.20	97.83	97.08	96.89	96.73
GSA	97.84	97.82	97.96	97.65	97.19	97.33	97.99	97.58
HSA	**98.52**	**98.06**	**97.28**	**97.42**	**98.37**	**97.32**	**97.38**	**96.69**
ABC	97.57	96.73	95.90	95.23	97.33	96.44	95.37	94.73
BFO	94.86	95.01	94.59	94.74	94.75	94.96	94.39	94.37
NSO	96.62	95.94	95.26	94.50	96.52	95.64	95.37	94.47
FA	95.34	94.98	94.45	93.93	95.12	94.46	94.38	93.99
FWA	96.35	95.74	95.40	95.25	96.20	95.12	94.89	95.15
Average	**96.85**	**96.43**	**95.91**	**95.74**	**96.65**	**96.04**	**95.83**	**95.53**

matrix size increased. Thus, it is safe to conclude that for this application, an increase in the max-pooling matrix size impacts badly the model's performance. This behavior could be related to the increasing loss of information within the feature maps, as the larger the matrix is, the more pixels are removed and the smaller the generated maps are.

The results also show that hyperbolic tangent function models perform slightly worse than ReLU models. That behavior might be related to the relation between performance and linearity of data processing in convolutional neural networks, as it is commonly stated that ReLU functions are normally used to break some of the linearity in this networks, increasing the model's performance. Figure 13 presents the average performances of all tested algorithms under the different configurations of activation functions and max-pooling matrix size. From these graphs it is easy to visualize the performance superiority of the ReLU models over the hyperbolic tangent ones. It is also notable that the smaller the max-pooling matrix size, the better the model performs. Thus, the best results are obtained using the standard max-pooling, 2x2 matrix size and the ReLU activation function.

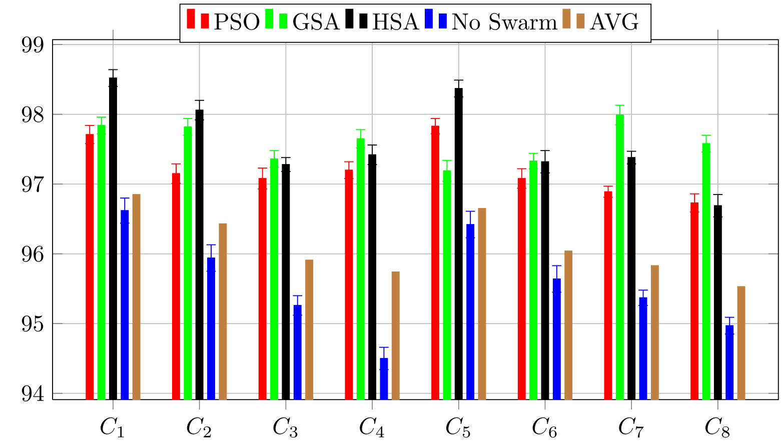
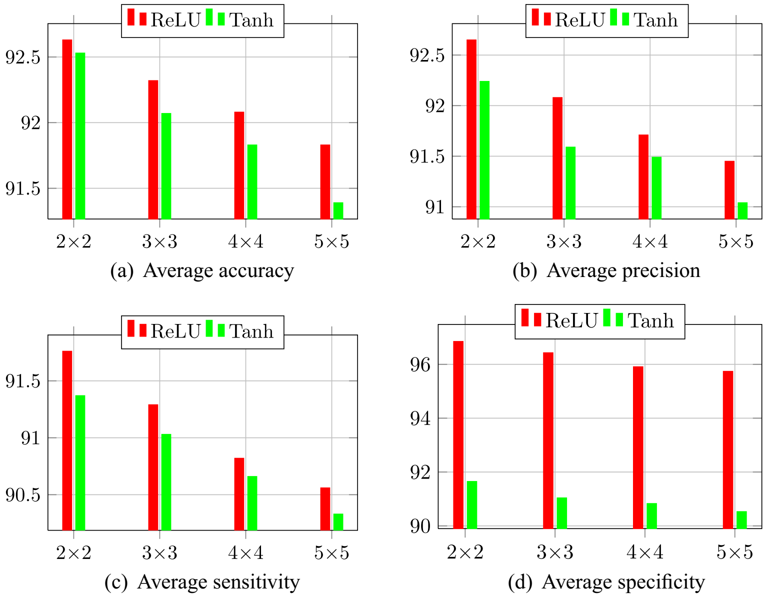


Fig. 12 Specificity performance of the top 3 algorithms, no swarm model and average of all algorithms



(a) Average accuracy

(b) Average precision

(c) Average sensitivity

(d) Average specificity

Fig. 13 ReLU x Hyperbolic Tangent average performance comparison in all algorithms

From these results, it is possible to verify that the real effectiveness of the use of swarm intelligence algorithms in the training of convolutional neural networks for the detection of pulmonary nodules, where at least three algorithms are more efficient than the classical techniques of training via gradient descent and back-propagation. The Particle Swarm Optimization algorithm obtained the best results regarding accuracy and sensitivity while the Harmony Search Algorithm and the Gravitational Search Algorithm achieved the highest specificity and precision, respectively. In a comparison with one of the related works, it is possible to observe the same behavior seen in the work of [8], wherein the model trained with swarm algorithm is produced a better performance with respect to the traditionally trained model.

Another interesting performance metric to monitor is the total training time of each model. The average training time of each algorithm is given in Table 11, wherein it is possible to observe that the PSO exhibits the best timing performance. It is 25.74% faster than the back-propagation model. Once more, these results confirm the superiority of the swarm-trained models.

Following the same analysis done with the top three algorithms regarding performance, the back-propagation model and the average performance of all tested models, Fig. 14 presents a performance comparison regarding training time, wherein it is possible to observe that the PSO model is the fastest.

From the results obtained, one must analyze the nature of the problem in order to choose the best algorithm. Combining the results shown in Figs. 9—12 with the numeric data listed in Tables 7, 8, 9 and 10 respectively, it is possible to conclude that, in this application, the Harmony Search algorithm performs better regarding specificity. This means that it is better in assuring that the lungs, which are classified as non-cancerous are really non-cancerous. On the other hand, the Gravitational Search algorithm performs better regarding precision. This means that it gets the most hits among those cases classified as cancerous. Furthermore,

Table 11 Average training time (mns) for both activation functions and all max-pooling matrix sizes

	ReLU				Hyperbolic tangent			
Algorithm	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
	2×2	3×3	4×4	5×5	2×2	3×3	4×4	5×5
PSO	**90.21**	**92.80**	**94.17**	**96.42**	**91.23**	**93.83**	**94.70**	**97.68**
GSA	97.34	99.74	102.36	103.34	97.82	99.71	100.45	100.78
HSA	102.65	103.30	104.66	104.73	103.78	104.01	105.59	106.14
ABC	111.43	111.70	113.62	113.74	112.31	114.66	116.72	118.13
BFO	137.86	138.40	140.47	142.03	139.11	139.44	142.37	142.40
NSO	121.48	123.25	125.96	128.72	121.57	121.82	124.27	125.44
FA	98.43	98.68	101.56	102.89	99.61	101.58	102.79	105.63
FWA	97.38	98.84	99.95	101.56	98.50	99.05	99.88	101.08
Average	**109.15**	**110.67**	**112.32**	**113.70**	**109.88**	**111.37**	**113.19**	**114.62**

the Particle Swarm Optimization can be declared the best algorithm regarding both accuracy and sensitivity. This means that it not only has the highest hit rate but it is also better in assuring that the lungs, which are classified as cancerous are really cancerous.

In an overall analysis, when it comes to the detection of a disease like cancer, negatively ranking a cancer patient is much more painful than positively ranking a patient without cancer. Thus, the number of false negatives (cancer patients classified as without cancer) is the most important factor in this decision, followed by the accuracy, which outlines overall hits of the model. Considering these factors, it can be concluded that Particle Swarm Optimization is the best algorithm for training the pulmonary nodule classifier model of this application.

In comparison with some of the related work, this work extends the statement brought by [8], that for applications like this, swarm-trained deep learning models perform better than back-propagation ones. However, in that work, different swarm intelligent techniques are used. Regarding the actual performance, this work overcomes the SVM model developed in

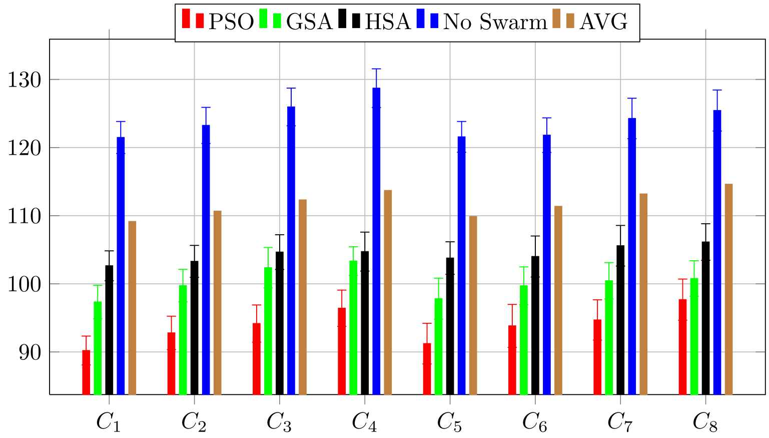


Fig. 14 Training time (mns) of the top 3 algorithms, no swarm model and average of all algorithms

[22], which achieved only 80.3 regarding accuracy against 93.71 as achieved in this work. Furthermore, the proposed model achieved similar performances as those presented in [12, 16]. In all cases, our model is more efficient regarding training time.

Considering the results obtained, this work clearly contributes towards an improvement, occasioning a higher performance, confirming the work hypothesis that swarm optimization algorithms combined with convolutional neural networks are a leap forward in medical images classification. In a detailed approach, the contribution is given regarding the nature of the problem, where the most efficient method is considered the one that misses less cancerous patients and achieves the highest overall hit rates. Considering these factors, the PSO-trained model is pointed out as the best option, granting a starting point for further academic research for an even better model.

7 Conclusion

The detection of pulmonary cancerous nodules through computed tomography scans is usually an arduous task even for the eyes of an experienced doctor, yet it stills a hard task for computational methods, mostly due to the high processing effort it requires and the need of obtaining correctly classified rich and good quality data. In order to make a contribution to the academic community, this study brings a new approach to lung cancer detection, a swarm-trained convolutional neural network model that aimed at achieving higher performances in nodule classification than regular back-propagation-trained models. With the results obtained it is, fortunately, possible to confirm that initial premise, which stated the superiority of swarm-trained models in applications such as this one.

Among the conducted experiments, there is a doubt about which activation function is the most efficient for the network. Although the Rectified Linear Unit (ReLU) function is the most used in convolutional neural networks applications, this work also aimed at testing the Hyperbolic Tangent function, which is vastly used in general neural network models. On the same function testing routines, different max-pooling matrices are tested to measure their performance influences. With those experiments, it could be concluded that the ReLU models indeed performed better than hyperbolic tangent models, a fact that might be related to the ReLU behavior of wiping out some of the model's linearity, which is stated by many researchers as being a good practice in this kind of application. Regarding the search for an optimal max-pooling matrix size, it could be pointed out that the larger the matrix the worse the performance. This behavior is most likely to be confirmed by the amount of data lost in each max-pooling operation, as bigger matrices lose more information than smaller ones. When comparing the performance of the different swarm intelligence algorithms tested in this application, 3 out of the 7 used algorithms stood out as the best options. The Particle Swarm Optimization, the Harmony Search Algorithm, and the Gravitational Search Algorithm achieved the top performances in all the four tested metrics, bringing in the need of a deeper analysis in order to choose the best algorithm. The choosing lies in finding the algorithm that would best fit the real needs of doctors and patients. Classifying a cancerous patient as non-cancerous is the worst possible scenario. Under these conditions, the Particle Swarm Optimization is the best algorithm, as it obtained the higher accuracy and the lowest rate of false negatives.

As possible future works and improvements in the methodology, one of the main points to be tested is the use of different feature maps on the convolutional neural network. Despite being complex experiments, these can bring forward significant performance improvements. In order to further guarantee the superiority of the swarm-trained convolutional networks, other swarm optimization algorithms could be tested, for example, the Hitchcock Birds Optimization developed by [17]. Furthermore, in order to extend the range of the used techniques and to provide a richer comparison among the swarm intelligence algorithms, the same application could be conducted using genetic algorithms to train the convolutional neural network, as in [13]. Also, aiming at a deeper analysis and comparison between the convolutional neural networks used for the transfer learning operation, this work could also be extended using some of the state-of-the-art convolutional networks, such as the *ResNet* [7] and the *GoogLeNet* [23]. In addition, this application could be expanded in future works to detect and classify other types of cancerous nodules, such as breast, skin or liver cancers.

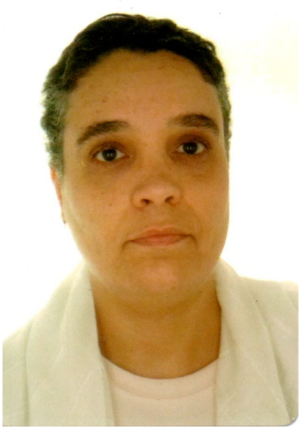
References

1. American Society of Clinical Oncology (2018) Biopsy. [Online; Accessed 10 Oct 2018]
2. Armato SG III, McLennan G, Bidaut L, McNitt-Gray MF, Meyer CR, Reeves AP, Zhao B, Aberle DR, Henschke CI, Hoffman EA et al (2011) The lung image database consortium (lidc) and image database resource initiative (idri): a completed reference database of lung nodules on ct scans. Med Phys 38(2):915–931
3. Chon A, Balachandra N, Lu P (2017) Deep convolutional neural networks for lung cancer detection. Standford University
4. Fabian Isensee (2018) U-net. [Online; Accessed 10 Oct 2018]
5. Geem ZW, Kim JH, Loganathan GV (2001) A new heuristic optimization algorithm: harmony search. Simulation 76(2):60–68
6. Ritchie H, Roser M (2019) Causes of death. [Online; Accessed 10 Oct 2018]
7. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 770–778
8. Jayanthi SK, Preetha K (2016) Breast cancer detection and classification using artificial neural network with particle swarm optimization. Int J Adv Res Basic Eng Sci Technol, 2
9. Karaboga D, Basturk B (2007) A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm. J Global Optim 39(3):459–471
10. Kennedy J, Eberhart R (1995) Particle swarm optimization
11. Khajehzadeh M, Eslami M (2012) Gravitational search algorithm for optimization of retaining structures. Indian J Sci Technol, 5(1)
12. Kuan K, Ravaut M, Manek G, Chen H, Lin J, Nazir B, Chen C, Howe TC, Zeng Z, Chandrasekhar V (2017) Deep learning for lung cancer detection: tackling the kaggle data science bowl 2017 challenge. arXiv:1705.09435
13. Lee MC, Boroczky L, Sungur-Stasik K, Cann AD, Borczuk AC, Kawut SM, Powell CA (2010) Computer-aided diagnosis of pulmonary nodules using a two-step approach for feature selection and classifier ensemble construction. Artif Intell Med 50(1):43–53
14. Mazurowski MA, Habas PA, Zurada JM, Lo JY, Baker JA, Tourassi GD (2008) Training neural network classifiers for medical decision making: the effects of imbalanced datasets on classification performance. Neur Netw 21(2-3):427–436
15. Mhetre MRR, Sache MRG Detection of lung cancer nodule on ct scan images by using region growing method
16. Miah MBA, Yousuf MA (2015) Detection of lung cancer from ct image using image processing and neural network. In: 2015 International conference on electrical engineering and information communication technology (ICEEICT), pp 1–6
17. Morais R, Mourelle LM, Nedjah N (2018) Hitchcock birds inspired algorithm: 10th international conference, ICCCI 2018, Bristol, UK, September 5-7, 2018, Proceedings, Part ii, pp 169–180, 01
18. National Institutes of Health (2011) Cancer costs projected to reach at least 158 billion in 2020. [Online; Accessed 10 Oct 2018]
19. Passino KM (2010) Bacterial foraging optimization. Int J Swarm Intell Res 1(1):1–16
20. Ritthipakdee T, Premasathian J (2017) Firefly mating algorithm for continuous optimization problems 2017
21. Ronneberger O, Fischer P, Brox T (2015) U-net: convolutional networks for biomedical image segmentation. In: International conference on medical image computing and computer-assisted intervention. Springer, pp 234–241

22. Sivakumar S, Chandrasekar C (2013) Lung nodule detection using fuzzy clustering and support vector machines. Int J Eng Technol 5(1):179–185
23. Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A (2015) Going deeper with convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1–9
24. Tan Y, Zhu Y (2010) Fireworks algorithm for optimization. In: Proceedings of the first international conference on advances in swarm intelligence - volume part I, ICSI'10. Springer, Berlin, pp 355–364
25. Wang S, Zhang Y, Dong Z, Du S, Ji G, Yan J, Yang J, Wang Q, Feng C, Phillips P (2015) Feed-forward neural network optimized by hybridization of pso and abc for abnormal brain detection. Int J Imaging Syst Technol 25(2):153–164
26. Zhang Y, Wang S, Wu L (2010) A novel method for magnetic resonance brain image classification based on adaptive chaotic pso. Prog Electromagn Res 109:325–343
27. Zhang Y, Wang S, Phillips P, Dong Z, Ji G, Yang J (2015) Detection of alzheimer's disease and mild cognitive impairment based on structural volumetric mr images using 3d-dwt and wta-ksvm trained by psotvac. Biomed Signal Process Control 21:58–73
28. Zhou Z-H, Jiang Y, Yang Y-B, Chen S-F (2002) Lung cancer cell identification based on artificial neural network ensembles. Artif Intell Med 24(1):25–36

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Cesar Affonso de Pinho Pinheiro holds an Engineering degree in Mechanics from State University of Rio de Janeiro in Brazil. He also holds a M.Sc. degree in Electronics engineering at the State University of Rio de Janeiro in Brazil. His research interests include programming, deep learning, swarm intelligence and computational intelligence in general.



Nadia Nedjah is an associate professor in the Department of Electronics Engineering and Telecommunications at the Faculty of Engineering, State University of Rio de Janeiro, Brazil. Her research interests include functional programming, embedded systems and reconfigurable hardware design as well as cryptography. Nedjah received her Ph.D. in Computation from the University of Manchester - Institute of Science and Technology (UMIST), England, her M.Sc. in System Engineering and Computation from the University of Annaba, Algeria and her Engineering degree in Computer Science also from the University of Annaba, Algeria.

Luiza de Macedo Mourelle is an associate professor in the Department of System Engineering and Computation at the Faculty of Engineering, State University of Rio de Janeiro, Brazil. Her research interests include computer architecture, embedded systems design, hardware/software co-design and reconfigurable hardware. Mourelle received her PhD in Computation from the University of Manchester - Institute of Science and Technology (UMIST), England, her MSc in System Engineering and Computation from the Federal University of Rio de Janeiro (UFRJ), Brazil and her Engineering degree in Electronics also from UFRJ, Brazil.

Affiliations

Cesar Affonso de Pinho Pinheiro[1] · **Nadia Nedjah**[1] · **Luiza de Macedo Mourelle**[2]

Cesar Affonso de Pinho Pinheiro
cesaraffonso05@hotmail.com

Luiza de Macedo Mourelle
ldmm@eng.uerj.br

[1] Department of Electronics Engineering and Telecommunications, Faculty of Engineering, State University of Rio de Janeiro, Rio de Janeiro, Brazil

[2] Department of Systems Engineering and Computation, Faculty of Engineering, State University of Rio de Janeiro, Rio de Janeiro, Brazil