A Self-adaptive Exhaustive Search Optimization-based Method for Restoration of Bridge Defects Images

Eslam Mohammed Abdelkader*[1,2], Mohamed Marzouk[3], and Tarek Zayed[4]

[1] Ph.D. candidate, Department of Building, Civil, and Environmental Engineering, Concordia University, Montreal, QC, Canada. Corresponding author, E-mail: eslam_ahmed1990@hotmail.com.
[2] Assistant lecturer, Structural Engineering Department, Faculty of Engineering, Cairo University, Egypt.
[3] Professor of Construction Engineering and Management, Structural Engineering Department, Faculty of Engineering, Cairo University, Egypt.
[4] Professor, Department of Building and Real Estate, the Hong Kong Polytechnic University, Hung Hom, Hong Kong.

A Self-adaptive Exhaustive Search Optimization-based Method for Restoration of Bridge Defects Images

ABSTRACT

Existing bridges are aging and deteriorating. Furthermore, large number of bridges exist in transportation networks meanwhile maintenance budgets are being squeezed. This state of affairs necessities the development of automatic bridge defects evaluation model using computer vision technologies to overcome the limitations of visual inspection. The digital images are prone to degradation by noises during the image acquisition phase. The absence of efficient bridge defects image restoration method results in inaccurate condition assessment models and unreliable bridge management systems. The present study introduces a self-adaptive two-tier method for detection of noises and restoration of bridge defects images. The first model adopts Elman neural network coupled with invasive weed optimization algorithm to identify the type of noise that corrupts images. In the second model, moth-flame optimization algorithm is utilized to design a hybrid image filtering protocol that involves an integration of spatial domain and frequency domain filters. The proposed detection model was assessed through comparisons with other machine learning models as per split validation and 10-fold cross validation. It attained the highest classification accuracies, whereas the accuracy, sensitivity, specificity, precision, F-measure and Kappa coefficient are 95.28%, 95.24%, 98.07%, 95.25%, 95.34%. 95.43% and 0.935, respectively in the separate noise recognition module. The capabilities of the proposed restoration model were evaluated against some well-known good-performing optimization algorithms in addition to some conventional restoration models. Moth-flame optimization algorithm outperformed other restoration models, whereas peak signal to noise ratio, mean-squared error, normalized absolute error and image enhancement factor are 25.359, 176.319, 0.0585 and 7.182, respectively.

Keywords: Bridge defects, computer vision, image restoration, Elman neural network, moth-flame optimization, filtering protocol

1. INTRODUCTION

Bridges are regarded as one of the core elements of the infrastructure systems. Meanwhile, they are vulnerable to severe deterioration agents such as freeze-thaw cycles, excessive distress loads due to traffic overload, sulfates, alkali-silica reaction (ASR), poor construction practices, etc. Based on the Canadian infrastructure report card, 26% of the bridges are either in a “Fair”, “Poor” or “Very Poor” conditions. The backlog of maintenance, repair and rehabilitation activities is $10 billion. The continuous increase in the backlog results in a substantial degradation in the condition of the bridges. One-third of Canada’s bridges have structural or functional deficiencies with short remaining service life, whereas 20 million light vehicles, 750,000 trucks, and 15,000 public transits use Canadian bridges annually [1]. The average age of the bridges is 24.5 years in 2007 compared to a mean service life of 43.3 years. Thus, 57% of the estimated service life has already been consumed [2].

Based on the aforementioned statistics, it is very crucial to evaluate the condition of the bridge decks in order to maintain them within a safe condition. Thus, the detection and evaluation of surface defects are very essential for timely maintenance of various concrete bridges. Recently, the use of digital image processing to deal with the surface defects becomes a research trend because the accuracy and efficiency of visual inspection-based methods are highly dependent on the skills and experience of inspectors. Thus, the subjectivity associated with the visual inspection-based methods requires the development of an automated method that can efficiently evaluate the severities of surface defects based on machine vision technologies.

The digital images are subjected to degradation by noise during the image acquisition stage or as a result of unfavorable conditions during image transmission. Noise is undesirable random fluctuations in the color and brightness of the image. The restoration of bridge defects images is

a pre-processing operation that involves two stages. The first stage is to determine which type of separate noise or combination of noises that corrupt the images. The second stage incorporates the application of a certain filtering method to remove noises from images. The inefficiency or the absence of one of the two stages can substantially affect the further feature extraction, detection and evaluation of surface defects. Thus, the restoration of bridge defects needs to take place to increase the prediction capacity of the surface defects evaluation method by enabling the precise extraction of important features of surface defects such as length and width of cracks and area of spalling. For instance, if a crack image corrupted with noise, is processed this can lead to inaccurate analysis and diagnosis of the surface defect.

In the view of the above, Noise detection and recognition is a key stage because it enables to determine the suitable filters to deal with the noise. Thus, having prior knowledge about the nature of the noise is fundamental in noise removal. Otherwise, noise removal can lead to image blurring. Furthermore, the process of noise removal while preserving as much as possible the edges and texture details of the image is a challenging task. Moreover, the task becomes more challenging when the images are subjected to a combination of noises. As such, the present study introduces a self-adaptive two-tier optimization-based method for the detection of noises and restoration of bridge defects images. Therefore, the main objectives of the present study are as follows:

1- Develop a hybrid Elman neural network-invasive weed optimization model (ENN-IWO) for the detection and recognition of separate and combined noises in bridge defects images.
2- Design a hybrid self-adaptive moth-flame optimization model for the restoration of bridge defects images.

3- Validate the previously-developed models through comparisons with other machine learning models reported in literature.

2. LITERATURE REVIEW

The literature review is divided into three sections: 1) overview of noise detection and removal models, 2) restoration of bridge defects images, and 3) research gaps.

2.1 Overview of Noise Detection and Removal Models

Noise detection is one of the key challenges in computer vision, whereas once the type of noise is correctly identified, the appropriate filtering method is applied to de-noise the captured image because poor de-noising often arises from the incorrect identification of the noise type. Karibasappa and Karibasappa [3] presented a method that integrated probabilistic neural network and fuzzy C-means clustering algorithm to classify the images based on the noise type. The addressed noise types were Gaussian white noise, speckle noise, salt and pepper noise and non-Gaussian white noise. The classification of the noise types was based on statistic features, namely Kurtosis and skewness. Chuah et al. [4] utilized deep convolutional neural network for the detection of the presence of Gaussian noise and noise levels. The deep convolutional neural network was capable of the detection of 10 classes of noise levels with an overall accuracy of 74.7%.

Turajlic and Begovi [5] proposed a method for the detection of Gaussian noise levels using artificial neural network in the singular value decomposition domain. They investigated the computational time for different alternatives of block sizes. They concluded that the proposed method achieved the lowest mean-squared error in the case of low noise levels. Vasuki et al. [6] proposed a method for the classification of noise types based on artificial neural network. They

utilized some statistical features such as Kurtosis and skewness for the identification of the noise type. Artificial neural network yielded better classification results when compared to the K-nearest neighbors algorithm, whereas the proposed method was capable of achieving an accuracy of 98.57%, 92.85% and 98.57% for Gaussian noise, salt and pepper noise and speckle noise, respectively.

Noise removal is one of the most worked upon topics in the area of image processing in recent years. Gupta et al. [7] compared six types of filters as per their capabilities to remove speckle noise, Gaussian noise, salt and pepper noise and Poisson noise based on the mean-squared error, peak signal to noise ratio and mean absolute error. They stated that the mean filter performed well in removing speckle noise, salt and pepper noise, and Poisson noise. On the other hand, Gaussian filter de-noised the Gaussian noise efficiently. Verma and Mehra [8] utilized particle swarm optimization algorithm to improve the performance of the median filter in de-noising the images corrupted with the salt and pepper noise. They stated that the proposed method provided higher peak signal to noise ratio and higher image quality index when compared to the median filter and adaptive median filter.

Dass [9] introduced a method that integrated bacterial foraging optimization (BFO) algorithm, discrete wavelet transform and Wiener filter to remove speckle noise from captured images. BFO algorithm is applied to minimize the error between the restored image and the original image. The proposed method provided better restoration results in terms of mean absolute error and peak signal to noise ratio when compared to adaptive median filter and Wiener filter. Kumar et al. [10] presented a method that combined adaptive particle swarm optimization algorithm with fuzzy median filter for the restoration of noisy images. The proposed method achieved higher image de-noising when compared to the bilateral filter, Wiener

filter and median filter as per the peak signal to noise ratio and second derivative-like measure of enhancement.

Wang et al. [11] developed margin setting algorithm to detect salt and pepper noise in digital images. Then, ranked order median filter was applied to de-noise the corrupted images. The developed noise detection model yielded less false positive rate than support vector machines. The ranked order median filter outperformed standard median filter as per peak signal to noise ratio, mean-squared error, image enhancement factor and structural similarity index. Zhao et al. [12] presented a model to supress noises in digital images based on Demspter-Shafer evidence theory. An improved accelerated algorithm within a sample window of size 2×2 was introduced for better noise detection existence. The developed model achieved higher peak signal to noise ratio when compared to the mean filter for different percentages of noise densities. They highlighted that the developed model provides a fast and efficient platform to remove noises from images. Ma et al. [13] developed a model that integrated fuzzy C-means clustering algorithm and non-local spatial information for image segmentation. The non-local means was utilized to restore the degraded images by restraining noises, and increase the segmentation capacity of the fuzzy C-means clustering algorithm by decreasing its sensitivity to the different types of noises. The developed model outperformed different variants of the fast generalized fuzzy C-means clustering algorithm based on a set of image segmentation quality indicators.

2.2 Restoration of Bridge Defects Images

Image restoration can be performed in several domains such as spatial domain and frequency domain. Most of the previous reported efforts in the literature utilized standalone filters for image restoration purposes in bridges. Tong et al. [14] introduced a method for image-

based crack detection to facilitate the inspection process in reinforced concrete bridges. Gaussian filter was used to remove the noise and enhance the image quality. Morphological operations are used to ensure the connection between the crack segments. The objective of the proposed method was to determine whether or not the images contained cracks. The proposed model achieved an accuracy of 93% and it outperformed some other methods such as Fujita method, canny edge detection method and Sobel edge detection method. Adhikari et al. [15] developed an artificial neural network-based model to predict the depth of the crack given a certain crack width based on an input dataset of 101 images of bridges. Median filter was applied to smooth images, which enabled the accurate interpretation of cracks. They also developed an approach based on spectral analysis to detect the change in crack patterns over time by converting digital images to the frequency domain using Fast Fourier Transform (FFT).

Yao et al. [16] presented a bridge crack detection and classification model based on a climbing root using a set of image processing techniques. Wiener filtering method was applied to remove the motion blur of the acquired images. Then, the wavelet transform was employed to minimize the texture effects of the crack area and finally, support vector machine (SVM) was implemented to classify the cracks and evaluate their severity levels. Lee et al. [17] developed a bridge inspection system using an unmanned aerial vehicle (UAV). Median filter was used to remove the noises and blurring present in images. Otsu method was applied to segment the images to objects of interest and background. Then, the crack properties in the HSV space were used to distinguish between cracks and other surface irregularities. They highlighted that their model was capable of detecting cracks measured in micrometers.

Ellenberg et al. [18] developed a bridge damage quantification model using digital images collected from unmanned aerial vehicles. Median filter was applied to remove the noise and

enhance the contrast in images. The proposed method combined high-resolution cameras with camera calibration and homography for tracing of cracks. They highlighted that the proposed method was capable of detecting cracks in images, which could eventually provide efficient bridge inspection models. Lei et al. [19] developed a method for crack detection method using unmanned aerial vehicle technology combined with digital image processing. Gaussian filter was applied to remove noise from images. The developed crack central point method outperformed other edge detection methods such as Canny algorithm, Sobel algorithm, Laplacian of Gaussian and Prewitt algorithm.

Li et al. [20] introduced a two-stage crack detection method based on convolutional neural network. A median filter was applied to de-noise the input images for further processing stages. The first stage involved feeding a small patch centering each pixel into the predictor to compute the probability that a pixel belong to a cracked area. In the second stage, a bigger patch elicited from the first confidence map is fed into the second predictor to obtain a second confidence map. Finally, the two confidence maps are combined to generate a final confidence map, which is used to map whether or not a certain pixel belong to cracked regions. The introduced method outperformed the canny edge detector method and STRUM (Spatially tuned robust multi-feature) method as per accuracy, precision and sensitivity. Dinh et al. [21] introduced a computer vision-based method for concrete crack detection. Average filter was applied to smooth the input images and remove the blob-like noise. A non-parametric peak detection algorithm was developed for binarization purpose in order to be able to differentiate defected and non-defected regions. They highlighted that the proposed method provided satisfactory results in the case of high noisy background images and low contrast images.

Wang et al. [22] proposed a method for crack detection in concrete bridges based on a set of image processing techniques. Adaptive filtering was integrated with contrast enhancement to eliminate the background noise and facilitate the accurate extraction of crack features. Then, a hybridization of Otsu and modified Sobel operator was applied for the detection of cracks. The proposed method achieved an absolute error of 0.02 mm in the detection of cracks width. Ho et al. [23] introduced a method for the damage detection of cable surface in cable-stayed bridges. Median filter was applied for noise reduction and histogram equalization. Then, the input images are mapped to principal component analysis space, where the Mahalanobis square distance was utilized to determine the distances between the input images and sample patterns, and eventually building the pattern recognition model.

Lee et al. [24] designed a machine vision robotic system to automate the inspection process of bridges. The developed system enables the user to evaluate the cracks in real-time based on a dataset of 100 noisy images. Median smoothing filter was applied to remove noises and to ensure uniform brightness through the image. Then, dilation and thinning morphological operations were utilized to maintain the connections between the crack segments. They demonstrated that the developed method yielded higher detection accuracies when compared to Sobel, Canny and Fujita methods. Pavithra et al. [25] proposed a computer vision-based method for the detection of cracks in reinforced concrete bridges. Then, median filter was applied to remove the salt and pepper noise present in images. Morphological segmentation was utilized to detect cracks in images using some operations such as dilation and erosion. The grey level co-occurrence matrix and statistical features were used to feed the detection model. Finally, the cascaded random forest classifier was applied to decide whether the images contain cracks or not.

2.3 Research Gaps

Based on the aforementioned studies, most of the restoration methods of bridge defects images are lacking a comprehensive investigation of the type of noise that the images are corrupted with. Moreover, the restoration of bridge defects images is a problem-dependent in real- time environment, i.e., applying a filtering method without specifying the type of noise leads to poor de-noising results under these conditions. Thus, it is decisive to find a method which aims to intelligently evaluate if an image is corrupted with noise, and what type of separate or mixed noise is corrupting the image before applying the de-noising method. The images which are corrupted with a mixture of noises create an amplified challenge to remove the mixed noise without compromising the edge sharpness and important features. As such, building a generic model which is irrespective of a specific type of noise can provide more robustness to the proposed method. Absence of noise detection models can lead to image blurring due to the application of incorrect or underperforming image restoration models. This will remarkably affect the following bridge defects evaluation procedures including: bridge defects severities extraction and detection, and eventually the accuracy of diagnosis of bridge defects severities.

In addition to that, most of the previous studies utilized a single filter such as median, mean or Gaussian filters to deal with different types of noises. Nevertheless, a single filter fails to deal with all types of noises, whereas some filters behave efficiently with some types of noises and fail to deal with others. Another issue in the reported de-noising methods is the parameter of the filters, whereas most of the filtering methods proposed in the literature are attribute or threshold governed such as mask size of 4×4. The window size selection in the neighborhood filters is a key issue in de-noising, whereas smaller window sizes sometimes don’t completely remove the noise while larger window sizes sometimes lead to edge blurring. The absence of

noise detection models and inefficient restoration methods lead to the establishment of inaccurate condition assessment models and unreliable deterioration models, which eventually leads to inefficient bridge management systems. Thus, the present study proposes a self-adaptive restoration method which can automatically detect and recognize the type of noise in bridge defects images. Moreover, it designs an image filtering protocol for each type of noise which can remove the noise in bridge defects images and preserve their edges and other features without human handpicking of the parameters.

3. PROPOSED METHOD

The ultimate objective of the proposed method is to design a filtering protocol for how to deal with different types of separate or mixed noises that corrupt bridge defects images. The proposed method is a two-tier framework for the automatic recognition of noise and restoration of degraded bridge defects images. The framework of the proposed restoration method of bridge defects images is depicted in Figure 1. The first model is the automatic classification of noises, whereas three modules are developed for the detection and recognition of noise types based on the level of details the asset managers are concerned with. The first module is the noise detection, whereas a binary classification module is constructed to classify the images based on the existence of noise, i.e., to classify whether the image is corrupted with noise or not. The second module is the separate noise recognition, whereas it is formulated as a four-point classification problem to provide a higher level of detail. The output of this module is to identify whether the image is corrupted with speckle noise, salt and pepper noise, or Gaussian noise or not corrupted with the noise. The third classification module is the combined noise recognition such that it provides the highest level of detail based on a formulation of a seven-point classification problem. This module is used to identify whether the image is corrupted with

speckle noise, salt and pepper noise, Gaussian noise, combination of speckle and salt and pepper noises, combination of speckle and Gaussian noises, combination of salt and pepper and Gaussian noises, or not degraded with noise.

INSERT FIGURE 1

For the first phase, the first step is to convert the RGB image into a gray-scale image, whereas the intensity values of the gray-scale image vary from 0 to 255. For the RGB image, R stands for red, G stands for green, and B stands for blue. The gray-scale images can improve the process of image processing without losing important features of the distress. Then, the converted images are standardized to a size of 200×200 to ensure same size images in the training and testing processes of the neural network, and to speed up the computational process. The next step is to convert the noise free image into a noisy image. Different combinations of separate and mixed noises are added to create the noisy images. Then, a set of statistical features are extracted from the noisy images to be able to classify the noise present in the image. The statistical features include: mean, mode, median, range, standard deviation, skewness, kurtosis, 75^{th} percentile and 50^{th} quartile.

Training Elman neural networks with meta-heuristic optimization algorithms is a powerful tool to improve the search engine of the Elman neural network by addressing the exploration-exploitation trade-off dilemma. The proposed method utilizes invasive weed optimization algorithm is used for both parametric and structural learning, i.e., to automatically optimize the weights and define the best possible architecture of the Elman recurrent neural network. The Elman neural network is trained by designing a variable-length single-objective optimization problem which encompasses a fitness function of minimization of misclassification error. The steps of the invasive weed optimization algorithm are repeated until satisfying the

convergence criteria, i.e., reaching maximum number of iterations. The optimized Elman neural network is saved and utilized to simulate the testing dataset.

The proposed method is compared with five other machine learning models to demonstrate the capabilities of the proposed noise detection and recognition method. The five models are discriminant analysis (DA), K-nearest neighbors (KNN), random forest (RF), support vector machines (SVM) and back-propagation artificial neural network (ANN). The comparison is conducted based on six performance metrics, namely precision, F-measure, sensitivity, specificity, accuracy and Kappa coefficient. The performance of the different noise detection and recognition models were evaluated using split validation and 10-fold cross validation. The K-fold cross validation is applied to ensure the training and testing of the entire dataset, which rules out any possibility of over-fitting or over-learning in the pattern recognition phase. Finally, parametric and non-parametric tests were performed between each pair of classifiers to evaluate the statistical significance level of the outcome of classifiers using the performances of the different folds. The parametric test is the Student's t-test while the non-parametric tests are Wilcoxn test, Mann-Whitney-U test, Kruskal–Wallis test, binomial sign test, Mood's median test, Friedman test and Friedman's aligned ranks test [26, 27].

After mapping each image to a specific type of noise or noises, the second model is the restoration of bridge defects images. Image restoration aims at removing the maximum undesirable noise from the captured images and trying to bring the noisy image as much as possible to its un-degraded ideal state. Assume a degradation function H and a noise function $n(x,y)$ which are added to the original image $A(x,y)$ to produce the degraded image $G(x,y)$. The objective of the restoration function is to obtain the reconstructed image $\hat{A}(x,y)$ and at the same time to be as close as possible to the original image $A(x,y)$. The degraded image in the spatial

domain can be expressed using Equation (1). As shown in Equation (1), based on the type of noise and degradation present in the image, an optimization problem is designed in order to define optimum configuration and parameters of the restoration method that can better filter out the noise present in the image and build the reconstructed image [28, 29].

$$G(x,y) = h(x,y) \times A(x,y) + n(x,y) \quad (1)$$

Where;

$h(x,y)$ represents the spatial representation for the degradation function. The symbol $\times$ indicate the spatial convolution.

After loading the degraded image, a self-adaptive hybrid filtering model is developed based on designing a variable-length optimization problem that considers a combination of spatial domain and frequency domain filters to provide more in-depth evaluation and better-restored images. The smoothing filters used in the present study are median filter, mean filter, mode filter, Wiener filter, Gaussian filter, Lee filter and Frost filter of variable sizes. The proposed model employs moth-flame optimization algorithm to search for the optimum structure and parameters of the restoration method using a single objective function that maximizes the peak signal to noise ratio, i.e., minimize the difference between the original image and reconstructed image of bridge defects. The superior capacity of the moth-flame optimization in exploration and exploitation motivated its application in solving the restoration problem of bridge defects images.

In addition to investigating different combinations of filters, the proposed method explores the effectiveness of the sequence of applying the filters, whereas the sequence of applying the smoothing filters can substantially affect the quality of the restored images. For

instance, the quality of the restored image when applying the median filter followed by the Wiener filter is different from applying the Wiener filter followed by the median filter. Thus, the objective of the proposed method is to define for each noise the following: optimum number of filters, optimum types of filters, optimum sequence of applying the filters, and optimum tuning parameters (governing attributes) of the applied filters.

The proposed method is validated on two stages. For the first stage, the proposed method is compared with the conventional filtering methods found in the literature. For the second stage, the proposed method is compared with a set of optimization algorithms which are: invasive weed optimization algorithm, differential evolution algorithm, modified differential evolution algorithm, grasshopper optimization algorithm, grey wolf optimization algorithm, particle swarm optimization algorithm, genetic algorithm and non-linear programming algorithm. This comparison is conducted to investigate the capacity of the proposed restoration method to search for the global optimum solutions in case of large search space and complex optimization problems against a set of well-known efficient meta-heuristics and exact optimization algorithm. The performances are assessed as per four performance metrics, namely peak signal to noise ratio (PSNR), mean-squared error (MSE), normalized absolute error (NAE) and image enhancements factor (IEF). Eventually, the significance level of the optimal solutions of the different meta-heuristic optimization algorithms is evaluated using the same parametric and non-parametric tests of the noise detection and recognition model.

4. TYPES OF NOISES

Noise represents unwanted information that degrades the quality of the image. The present study deals with three types of noises which are: Gaussian noise, salt and pepper noise and speckle noise [29, 30].

4.1 Gaussian Noise

Gaussian noise is the most common noise present in the image which mainly affects all the pixel values. Gaussian noise is a statistical noise that the Gaussian probability density function as shown in Equation (2). This type of noise is sometimes called “white noise”, which causes the image to be blurry.

$$f(g) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\left(\frac{g-\mu}{2\sigma^2}\right)^2} \quad (2)$$

Where;

g represents the gray level. μ and σ^2 represent the mean and variance of the noise, respectively.

4.2 Salt and Pepper Noise

Salt and pepper noise usually occurs due to errors during the image transmission phase, whereas the intensity of the corrupted pixel either has an intensity which is either very high or very low compared to the neighboring pixels. A pixel is called a “salt” pixel if its intensity is very high and it is called a “pepper” pixel if its intensity is very low. This type of noise is demonstrated in the form of dark pixels (black dots or pepper) in bright regions and bright pixels (white dots or salt) in dark regions.

4.3 Speckle Noise

Speckle noise is a granular noise that affects all the inherent characteristics of the image and increases the mean grey level in a local area. This type of noise is sometimes called “Data missing” noise, which occurs due to loss of data during the image transmission. The corrupted pixels are set to maximum value and the speckle noise follows a gamma distribution as illustrated in Equation (3).

$$f(g) = \frac{g^{\alpha-1}}{(\alpha-1)!} e^{\frac{-g}{\sigma^2}} \quad (3)$$

Where;

α represents the shape parameter of the speckle noise distribution.

5. TYPES OF FILTERS

Image restoration can be performed in several domains such as spatial domain and frequency domain. The present study investigates a set of spatial domain filters (mean, median, mode, Gaussian, Lee, and Frost) and a frequency domain filter (Wiener filter). Spatial domain techniques deal directly with the pixel intensities present in the image. However, frequency domain filters are based on the Fourier transform of the image.

5.1 Average Filter

Mean filter is a simple and easy algorithm that is utilized to remove irrelevant details in the image. It is used to minimize the noise in the image by minimizing the intensity variations in the image pixel and the next pixels. Average filter is based on computing the sum of all pixels in the filter window and dividing them by the number of pixels. Then the center pixel is replaced by the average value. The 2D mask (window) is applied to each pixel in the image. The larger the window size, the more noise can be removed effectively however; this can result in a blur image [29, 30].

5.2 Median Filter

Median filter is a non-linear filter that is mainly utilized to remove speckle noise and salt and pepper noise. Median filter helps in decreasing the intensity variations between a certain pixel and its neighboring pixels as the average filter. However, median filter provides a better

alternative to filter out the noise, smooth the image and preserve its details than the average filter because it is less sensitive than the average filter to the outliers. The median filter is performed through a 2D mask, which is transferred across the whole pixels of the image. The median value is computed by first arranging the pixel intensities in an ascending order and then the pixel is replaced by the middle pixel value. If the number of neighboring pixels is equal to an even number, then the pixel value is replaced by the average of the two middle pixel values. The median filter is not efficient when dealing with images, where half of the pixel values are affected. Thus, the median filter is not effective in removing Gaussian noise from images [29, 30].

5.3 Mode Filter

Mode filter is performed through a 2D mask applied to each pixel value, whereas each pixel value is replaced by the mode value of the neighboring pixels.

5.4 Gaussian Filter

Gaussian filter is used to smooth images by removing the Gaussian noise. Gaussian smoothing filter is utilized to remove the noise based on a Gaussian kernel function using Equation (4) [31].

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{\frac{x^2+y^2}{2\sigma^2}} \quad (4)$$

Where;

x represents the distance from the origin in the horizontal axis. y represents the distance from the origin in the vertical axis. σ indicates the standard deviation of the Gaussian distribution.

5.5 Wiener Filter

Wiener filter is a de-noising method that works in the frequency domain to filter out the noise from a corrupted signal and improve the signal to noise ratio. As mentioned before, the Wiener filter is a frequency domain filter, which means that it utilizes discrete Fourier transform (DFT) to transform the degraded image to the frequency domain. Wiener filter is an optimal image filtering technique that is used to minimize the mean square error between the restored image and the original image. The Fourier transform of the restored image can be expressed using Equation (5). As shown in Equation (5), the Fourier transform of the restored image is equal to the product of the Wiener filter and the original image.

$$\hat{F}(u,v) = G(u,v) \times A(u,v) \tag{5}$$

Where;

$\hat{F}(u,v)$ represents the Fourier transform of the original image. $G(u,v)$ represents the Wiener filter. $A(u,v)$ indicates the original image.

The Wiener filter can be obtained using Equation (6). The term $\frac{S_n}{S_f}$ is replaced by a frequency independent constant called K. Then a suitable value of the constant K that achieves a suitable restored image can be obtained by minimizing the mean squared error between the original image and the restored image [31, 32].

$$G(u,v) = \frac{H^*(u,v)}{|H(u,v)|^2 + \frac{S_n}{S_f}} \tag{6}$$

Where;

$H^*(u,v)$ denotes the Fourier transform of the degradation function. $H(u,v)$ represents the degradation function. S_n and S_f represent the power spectrum of the noise and power spectrum of the un-degraded image, respectively.

5.6 Lee Filter

Lee filter is a spatial filtering method based on the first order of statistics to reduce noise present in images and preserve their details. It depends on the minimum mean-squared error to attain the noise free images. The pixel being filtered is replaced by a value calculated based on the neighboring pixels. Lee filter assumes that the speckle noise is uniformly distributed all over the image. Thus, it sometimes causes blurring of the edges as a result of the sudden change in the pixels' intensity at the edges. The restored image is obtained as follows [33, 34].

$$R\hat{}(\tau) = R\hat{}(\tau) - W(\tau)[F\hat{}(\tau) - F(\tau)] \quad (7)$$

Where;

$R\hat{}(\tau)$ represents the restored image. $F(\tau)$ indicates the noisy image. $F\hat{}(\tau)$ represents the mean value of $F(\tau)$. $W(\tau)$ is the weighted function and can be obtained using the following Equations.

$$W(\tau) = \frac{\mathrm{var}(\tau)}{[F\hat{}(\tau)]^2\sigma^2 + \mathrm{var}(\tau)] } \quad (8)$$

$$\mathrm{var}(\tau) = \frac{{\sigma_I}^2 + {\mu_I}^2}{\sigma^2 + 1} - {\mu_I}^2 \quad (9)$$

Where;

$\mathrm{var}(\tau)$ represents the variance of the pixel being filtered. σ^2 represents the global variance of the noisy image. ${\sigma_I}^2$ and ${\mu_I}^2$ indicate the local variance and local mean, respectively.

5.7 Frost Filter

Frost filter is an exponentially weighted average filter that utilizes local statistics to reduce noise. Frost filter can be used to remove multiplicative noise from images, whereas it is based on the computation of variation, which is the ratio of the local standard deviation to the local mean of the noisy image. Thus, for high coefficients of variation, the sharp features in the image are preserved while the frost filter acts like the average filter for low coefficients of variation. The frost filter can be described as follows [33, 34].

$$R\hat{}(\tau) = \sum_{n \times n} K\alpha e^{-\alpha|t|} \tag{10}$$

Such that;

$$\alpha = \frac{4}{n\sigma\hat{}^2} \times \frac{\sigma^2}{I^2} \tag{11}$$

$$t = |X - X_o| + |Y - Y_o| \tag{12}$$

Where;

K is a normalizing constant. σ^2 represents the local variance. $\sigma\hat{}^2$ is the image coefficient of variation. I is the local mean. n is the moving kernel size.

6. MODEL DEVELOPMENT

This section describes in-detail the developed noise detection and recognition model in addition to the restoration models of bridge defects images presented in the “Proposed Method” section.

6.1 Noise Detection and Recognition Model

The present study a self-adaptive hybrid Elman hybrid ENN-IWO model to automatically train and classify the retrieved degraded images of bridge defects based on the noise type. The

following section discusses the basic theories of the Elman neural network and invasive weed optimization algorithm in addition to the hybrid ENN-IWO model for noise detection and recognition in bridge defects images.

6.1.1 Basic theory of Elman neural network

Elman neural network (ENN) is one of the recurrent neural networks (RNNs), which was proposed by Jeffrey Locke Elman in 1990 [35]. Elman neural network is characterized by additional context layers, which helps in providing a memory about the results of the computations done so far. The connections and dependencies between the layers form a directed cycle, which enables the neural network to preserve a state between the subsequent time steps [36**]**. The main difference between the conventional feed-forward neural networks and recurrent neural networks is that in the case of RNNs, the output at each time step depends on the previous inputs and previous computations by memorizing previous events while in the feed-forward neural network, outputs are independent of each and the network output depends only on the current time step [37, 38].

The architecture of the Elman neural network is depicted in Figure 2. Elman neural network is composed of: input layer, hidden layer, context layer, and output layer, whereas number of neurons in the context layer is the same as number of neurons in the hidden layer. The neurons in each layer are used to propagate information from one layer to the subsequent layers. The connections of hidden layers entering the context layer are not weighted while the connections of the context layer entering the hidden layer are weighted. Elman neural network is considered as a recurrent neural network because it has a feedback loop, which has a substantial impact on improving the learning capability of the network, which consequently enhances the performance

of the neural network. The feedback loop incorporates the use of unit-delay element (Z^{-1}), which provides a non-linear dynamic behaviour to the neural network.

INSERT FIGURE 2

The context layer takes its input from the output of hidden layer. Then, the context layer feeds into the hidden input layer. Therefore, the output of the hidden layer is going into two layers: context layer and output layer. The output from the hidden layer are sent into the context layer, stored, and fed through the weights into the hidden layer in order to rely on this information in the next iterations, so the neural network is constantly remembering the output from the hidden layer and re-feeding this output from the previous iteration into the hidden layer. This behaviour enables the neural network to maintain short term memory, which improves the network performance [39, 40].

The output of the hidden layer and output layer can be calculated using Equations (13) and (14), respectively.

$$X(k) = f(W_2 X_c(k) + W_1 U(k-1)) \quad (13)$$

$$Y(k) = g(W_3 X(k)) \quad (14)$$

Give that:

$$X_c(k) = X(k-1) \quad (15)$$

Where;

W_1 represents the weight of the input later to the hidden layer. W_2 represents the weight of the context layer to the hidden layer. $X(k)$ is the output of the hidden layer. $X_c(k)$ is the output of the context layer. $U(k-1)$ is the input of the neural network. $Y(k)$ is the output of the neural

network. f represents the transfer (activation) function at the hidden layer. g is the transfer function at the output layer.

Gradient descent (GD) algorithm is considered as one of the most commonly utilized algorithms to train the Elman neural network and back propagation neural networks. The networks are called “back propagation” because the error is computed at the output layer based on the desired and predicted output for each input value, and then the error distributed (propagates) backwards through the network layers from the output to the hidden layers and then further to the input layer. Gradient descent algorithm is based on finding the partial derivative of the error function to update the weights of the connections. The optimum weights are obtained based on minimizing the error function, which can be expressed as the sum of squared error (SSE) of the predicted and actual values. The error (cost) function is calculated using Equation (16).

$$E = \sum_{t=1}^{N2} (P_t - O_t)^2 \tag{16}$$

Where;

E represents the error function, i.e., the objective function, which should be minimized within each training epoch. P_t and O_t represent the predicted and actual values, respectively.

Based on the gradient descent algorithm, the weights are adjusted during each training epoch (k) based on Equation (17), whereas the error partial derivative is computed during each training epoch and subsequently, as per the error partial derivative and the learning rate, the weights are updated [41].

$$W_{ij}(k+1) = W_{ij}(k) + \Delta W_{ij}((k) = W_{ij}(k) - \eta \times \frac{\partial E(k)}{\partial W_{ij}} \quad (17)$$

Where;

$\Delta W_{ij}((k)$ represents the adjustment or increment in the weights (weight updates). $W_{ij}(k+1)$ and $W_{ij}(k)$ represent the new (updated) and current (old) weights, respectively. η depicts the learning rate. $\frac{\partial E(k)}{\partial W_{ij}}$ represents the error partial derivative with respect to the weights.

6.1.2 Basic theory of invasive weed optimization algorithm

Invasive weed optimization (IWO) algorithm is a meta-heuristic bio-inspired optimization algorithm that was developed by Mehrabian and Lucas in 2006. IWO algorithm is exhaustive search engine that demonstrated its capabilities in exploring complex and multi-local search spaces. Moreover, it manifested its superiority over some of the best-performing optimization algorithms. IWO algorithm is based on simulating the invasive behaviour of weed in colonizing and finding the most suitable place for growth and reproduction. Weeds are robust and undesirable plants that grow spontaneously and they can have a harmful effect on both farms and gardens. The computational procedures of the invasive weed optimization algorithm are discussed in the following lines [42, 43].

The first stage is to create an initial population of weeds that are spread in the i-dimensional search space. The fitness of each weed within the population is then computed based on a predefined objective function. The production of seeds associated with each weed is calculated based on a linear function, where the number of seeds varies between the minimum and maximum number of seeds. Each weed in the population produces seeds based on its own comparative fitness value, maximum and minimum fitness values within the population, and the

maximum and minimum number of seeds. The reproduction of seeds is shown in Equation (18) where the higher the fitness of the weed, the more seeds it produces

$$\mathrm{Seed_i} = \frac{\mathrm{f_i} - \mathrm{f_{min}}}{\mathrm{f_{max} - f_{min}}} \times (\mathrm{s_{max}} - \mathrm{s_{min}}) + \mathrm{s_{min}} \quad (18)$$

Where;

$\mathrm{Seed_i}$ represents number of seeds associated with the $\mathrm{i-th}$ weed. $\mathrm{f_i}$ represents the current fitness of the weed. $\mathrm{f_{max}}$, and $\mathrm{f_{min}}$ represent the maximum and minimum fitness of the current population, respectively. $\mathrm{s_{max}}$, and $\mathrm{s_{min}}$ denote the maximum and minimum number of seeds, respectively.

The following stage is the spatial dispersion, where the seeds are randomly scattered in the search space based on a normal distribution of a mean equal to zero and an adaptive varying standard deviation. This step ensures that the seeds are accumulated around the weed plant, which leads to a local search around each parent weed. The standard deviation of the seed dispersion is reduced from an initial predetermined maximum value to an initial predetermined smaller value based on a non-linear function as shown in Equation (19). The probability of finding a seed far from the weed plant is high at the beginning of the optimization process and it decreases within a predefined number of iterations.

$$\sigma_i = \sigma_{min} + \left(\frac{\mathrm{iter_{max} - iter}}{\mathrm{iter_{max} - iter_{min}}}\right)^{\mathrm{p}} \times (\sigma_{max} - \sigma_{min}) \quad (19)$$

Where;

σ_i indicates the standard deviation of the current iteration. σ_{max}, and σ_{min} indicate the initial and final standard deviation of the optimization process, respectively. $\mathrm{iter_{max}}$ represents the

maximum number of iterations. p represents non-linear modulation index, and usually, it is a number between two and three.

Finally, competitive exclusion is performed because the number of weeds and seeds reaches the maximum population size due to the fast reproduction (exponential increase in the number of plants). The parent weeds alongside with the seeds are ranked based on the fitness value in order to eliminate the solutions with the least fitness values to keep the number of the weed plants and seeds within the maximum allowable population size. The seeds and their parent weeds with higher fitness survive, and become reproductive. The process continues until the convergence criteria are met (reaching the maximum number of iterations).

6.1.3 Hybrid ENN-IWO for noise detection and recognition

The proposed method utilizes invasive weed optimization algorithm to train the Elman neural network. This is expected to enhance the search mechanism of the Elman neural network, which leads to improve its recognition capacity of the noise type in the bridge defects images. The IWO is utilized to optimize the ENN for two main reasons which are: inferior accuracy and convergence of the gradient descent algorithm, and manual tuning of the parameters of Elman neural network. The training process based on the gradient descent usually gets trapped in a local minima or premature convergence and sometimes causes over-fitting problems especially in the case of presence of multilayer neural network. The multi-layer neural network is normally associated with large search space, multi-local minima points, non-differential function and complex multi-dimensional curve. Moreover, in some cases, the global minimum is hidden

between the local minima. Thus, the gradient descent algorithm can end up oscillating between the local minima.

The second reason is the existence of wide range of parameters, which substantially affect the performance of the neural network. These parameters are sensitive to the initial values, whereas their initial setting is always variable from one case to the other. For instance, there is no exact method to define the number of hidden neurons, whereas most of the equations present in the literature are case dependent and cannot be generalized. As such, if the number of hidden neurons is less than the optimum number, then the accuracy will be so much affected. However, if the number of hidden neurons is more than optimum number, this will consume so much training time. Thus, the blindness in the determination of such parameters can result in the network to be trapped in an inferior solution and subsequently a long computational time of the training process and slow convergence. Thus, a self-adaptive model is designed in order to automatically and dynamically tune the input parameters based on the available dataset of bridge defects images.

In the present study, eight types of transfer functions are investigated. The hyperbolic tangent sigmoid transfer function, log-sigmoid transfer function, Elliot symmetric sigmoid transfer function and linear transfer function are shown in Equations (20), (21), (22) and (23), respectively. Positive linear transfer function, triangular basis transfer function, radial basis transfer function and normalized radial basis transfer function are depicted in Equations (24), (25), (26) and (27), respectively.

$$f(x) = \frac{2}{(1 + e^{-2x})} - 1 \tag{20}$$

$$f(x) = \frac{1}{(1 + e^{-x})} \tag{21}$$

$$f(x) = \frac{x}{1 + |x|} \tag{22}$$

$$f(x) = x \tag{23}$$

$$f(x) = \begin{cases} x, & \text{if } x \geq 0 \\ 0, & \text{if } x < 0 \end{cases} \tag{24}$$

$$f(x) = \begin{cases} 1 - |x|, & \text{if } -1 \leq x \leq 1 \\ 0, & \text{otherwise} \end{cases} \tag{25}$$

$$f(x) = e^{-x^2} \tag{26}$$

$$f(x) = \frac{e^{-x^2}}{\sum_{x=1}^{E} e^{-x^2}} \tag{27}$$

Where;

x represents the input of the transfer function. f(x) represents the output of the transfer function. E indicates the size of the entries of the transfer function.

Optimality theory is mainly based on the fixed-length assumption, whereas most of the optimization algorithms utilize a fixed-length vector of decision variables to represent a solution. However, some few cases present in the literature for the variable-length optimization problems, whereas the number of decision variables changes over the number of iterations (training epochs). The variable-length optimization problems are more complex and require more computational effort when compared to the fixed-length optimization problems [44]. In the variable-length optimization problems, there is no clear definition for the gradient vector for the variable-length problem. Thus, the gradient-based methods are inefficient in dealing with such

problems. Some ways to deal with such problems is to assume a fixed length for the decision variables and to tune iteratively the decision variable that causes variability in length. However, this method leads to suboptimal solutions. Moreover, it is impractical and inefficient method if the ranges of the decision variables are very wide. Therefore, the better approach is to design a model whose vector of solutions vary in length within the training epochs.

The development made in the use of Invasive weed optimization as a training mechanism includes optimization of both architecture and parameters of the Elman neural network. This encompasses selection of most suitable transfer functions between the network layers, number of hidden layers and hidden neurons, number of context layers and context neurons, and values of weights and bias terms. As a result of the structure and parameter learning of the proposed training paradigm, a variable-length optimization model is designed, whereas its length varies iteratively as per the number of hidden layers, hidden neurons, context layers and context neurons. As such an estimator is designed to handle the dynamism of the configuration of the ENN and to gives the user the flexibility to design a multi-hidden layer and a multi-context layer neural network based on the input dataset of bridge defects images. This is done by computing the number of weights and bias terms in each training epoch. The estimator can be mathematically defined using Equation (28).

$$\mathrm{Num} = \big((\mathrm{I}+1)\times \mathrm{N}\big) + ((\mathrm{N}\times \mathrm{C}\times \mathrm{P} + (\big(\mathrm{N}+1)\times \mathrm{N}\times (\mathrm{P}-1)\big) + \big((\mathrm{N}+1)\times \mathrm{O}\big) \qquad (28)$$

Where;

Num represents the total number of weights and bias terms. I represents the number of input neurons. N indicates the number of hidden neurons. C represents the number of neurons in the context layer. P represents number of hidden and context layers. O depicts the number of

output neurons. For simplification purposes, the number of context layers is assumed to be equal to the number of hidden layers

Elman neural network is trained using a single objective function which minimizes the misclassification error of the noise type of the bridge defects images. Therefore, the developed optimization model establishes a dynamically changing optimum configuration and characteristics of Elman neural network triggered by the number of images and their statistical features. The misclassification error is selected as an objective function because it is a well-known good performing performance indicator, unitless. Moreover, it is usually more practical to deal with cost functions. The mathematical formulation of the misclassification error can be defined as follows.

$$\mathrm{MC\_ER} = \frac{\mathrm{WC\_IM}}{\mathrm{T\_IM}} \tag{29}$$

Where;

MC_ER represents the misclassification error. WC_IM is the number of wrongly classified images. T_IM stands for the total number of images.

6.2 Restoration of Bridge Defects Images

The proposed method utilizes moth-flame optimization algorithm to design a filtering protocol, which encompasses designing a hybrid smoothing filter for each noise type that degrades bridge defects images. This enables to determine the optimum structure and parameter restoration method for each noise type. Moth-flame optimization (MFO) algorithm is newly-developed bio-inspired meta-heuristic algorithm that proved its superior capacity in solving complex optimization problems. Yildiz and Yildiz [45] utilized MFO algorithm to maximize the profit rate of multi-tool milling operations considering unit production time and unit production

cost and subject to a set of difficult constraints. The developed method outperformed some other meta-heuristics such as genetic algorithm, ant colony algorithm, hybrid immune algorithm, cuckoo search algorithm and hybrid particle swarm immune algorithm based on the maximization of profit rate. Yildiz et al. [46] conducted a comparative study to investigate the application of ten newly-developed meta-heuristic algorithms to solve the design of six mechanical engineering optimization problems. They highlighted that the MFO algorithm is an efficient and robust algorithm in solving complex mechanical design optimization problems.

MFO algorithm was developed by Seyedali Mirjalili in 2015, whereas it is based on the simulation of navigation mechanism of moths in nature, which is called “transverse orientation” [47]. In the moth-flame optimization algorithm, the moths are the candidate solutions. In this mechanism, moths fly in the night by maintaining a fixed angle with their alignment to the moon, which is deemed as a very efficient method for travelling long distances in a straight line. The moths are the search agents that fly in 1-D, 2-D, 3-D or hyper dimension search space while the flames are the best positions attained so far, i.e., flames are the flags or pins dropped by moths while exploring the search space. As such, the moth searches around the flag or flame, and updates it if it finds a better solution. Simultaneously, the flames are updated according to the fitness values of the fittest moths. The positions of the moths in the search space are the optimization problem’s parameters.

The computational procedures of the MFO algorithm are discussed in the following lines [46-48]. MFO algorithm is a population-based meta-heuristic algorithm. Thus, a set of moths can be defined as follows.

$$M = \begin{bmatrix} m_{1,1} & m_{1,2} & m_{1,3} & \ldots. & m_{1,d} \\ m_{2,1} & m_{2,2} & m_{2,3} & \ldots. & m_{2,d} \\ m_{3,1} & m_{3,2} & m_{3,3} & \ldots. & m_{3,d} \\ \ldots. & \ldots. & \ldots. & \ldots. & \ldots. \\ m_{n,1} & m_{n,2} & m_{n,3} & m_{n,4} & m_{n,d} \end{bmatrix} \quad (30)$$

Such that;

$$M(i,j) = (ub(i) - lb(i)) \times rand() + lb(i) \quad (31)$$

Where;

M is the position matrix of moths. n represents number of moths. d represents the number of design variables or the dimensions of the optimization problem. $M(i,j)$ is the value of the $i-th$ row and $j-th$ column of the matrix. $ub(i)$ and $lb(i)$ represent the upper and lower bounds of the $i-th$ moth, respectively. $rand()$ is a random number generated from a uniform distribution within the interval [0,1].

The performance of each moth is evaluated using a pre-defined objective function. Then, a fitness matrix is constructed to store the fitness function values of the moths. The mathematical formulation of the fitness matrix of moths can be expressed as follows.

$$OM = \begin{bmatrix} OM_1 \\ OM_2 \\ OM_3 \\ \ldots. \\ OM_n \end{bmatrix} = \begin{bmatrix} f(m_{1,1} & m_{1,2} & m_{1,3} & \ldots. & m_{1,d}) \\ f(m_{2,1} & m_{2,2} & m_{2,3} & \ldots. & m_{2,d}) \\ f(m_{3,1} & m_{3,2} & m_{3,3} & \ldots. & m_{3,d}) \\ \ldots. & \ldots. & \ldots. & \ldots. & \ldots. \\ f(m_{n,1} & m_{n,2} & m_{n,3} & m_{n,4} & m_{n,d}) \end{bmatrix} \quad (32)$$

Where;

OM indicates the fitness matrix of moths. $f(*)$ is the fitness function.

As mentioned before, flames are another important aspect in the moth-flame optimization algorithm. The set of flames can be expressed using Equation (33). The fitness matrix of flames can be defined using Equation (34).

$$F = \begin{bmatrix} f_{1,1} & f_{1,2} & f_{1,3} & \ldots. & f_{1,d} \\ f_{2,1} & f_{2,2} & f_{2,3} & \ldots. & f_{2,d} \\ f_{3,1} & f_{3,2} & f_{3,3} & \ldots. & m_{3,d} \\ \ldots. & \ldots. & \ldots. & \ldots. & \ldots. \\ f_{n,1} & f_{n,2} & f_{n,3} & f_{n,4} & f_{n,d} \end{bmatrix} \quad (33)$$

$$OF = \begin{bmatrix} OF_1 \\ OF_2 \\ OF_3 \\ \ldots. \\ OF_n \end{bmatrix} \quad (34)$$

Where;

F represents the position matrix of flames. $f(i, j)$ indicates the $j-th$ variable of the $i-th$ flame. OF_n is the fitness function value of the $n-th$ flame. n represents number of flames.

For the purpose of modelling the transverse orientation mechanism of moths, the MFO algorithm utilizes the logarithmic spiral as the main paradigm to update the positions of the moths with respect to the flames. The updated mechanism of moths' positions can be defined as follows.

$$M_i = S(M_i, F_j) = D_i \times e^{bt} \times \cos(2\pi t) + F_j \quad (35)$$

Such that;

$$D_i = |M_i - F_j| \quad (36)$$

Where;

S is the logarithmic spiral function. M_i and F_j represent the $i-th$ moth and $j-th$ flame. D_i represents the distance of the $i-th$ moth to $j-th$ flame. b is a constant that defines the logarithmic spiral motion. T is a random number within the interval [-1, 1]. It is worth mentioning that the spiral movement is a fundamental component in the MFO algorithm because it depicts how the moths update their positions around flames. The logarithmic spiral function enables the moth to fly around the flame and not necessarily in the search space between the

moths. This mechanism facilitates both exploration and exploitation of the search space, which is deemed as an advantage over other meta-heuristic optimization algorithms.

In order to further improve the exploitation mechanism of the MFO algorithm, t is assumed to be a random number in the range $[r, 1]$ such that r is a convergence constant decreasing from -1 to -2 over the course of iterations. In addition to that, each moth is obliged to update its position using only one of the flames as per Equation (37) to increase the probability of converging to a global solution and to avoid being trapped in a local minimum. In the MFO algorithm, the exploration of the search space is guaranteed since moths update their positions around the best solutions obtained so far in the hyper sphere during the optimization process. To enable more exploitation of the best promising solutions, an adaptive mechanism is employed to decrease the number of flames within each iteration number. The updated mechanism can be expressed as follows.

$$N_{FL} = \text{round}\left((N - it) \times \frac{N - 1}{T}\right) \tag{37}$$

Where;

N_FL represents the number of flames. it indicates the current iteration number. N and T stand for the maximum number of flames and maximum number of iterations, respectively.

Within each iteration, the positions and fitness values of moths and flames are updated as per Equations form (30) to (36) such that the moths update their positions in the hyper sphere around the best solution obtained so far. The sequence of flames is adapted with respect the best solutions obtained so far in each iteration, and then the positions of the moths are updated as per the updated flames. The proposed restoration model utilizes peak signal to noise ratio as an objective function to search for the optimum configuration and parameters of the hybrid smoothing filter for each noise type. Peak signal to noise ratio measures the quality of the image

as per the maximum possible signal (image) power to the noise power that affects the representation of the image, whereas higher PSNR indicates better quality of the image. Peak signal to noise ratio can be defined using Equation (38).

$$\mathrm{PSNR} = \mathrm{Max}(10 \times \log_{10}(\frac{\mathrm{R}^2}{\frac{1}{\mathrm{m} \times \mathrm{n}} \sum_{\mathrm{i}=1}^{\mathrm{m}} \sum_{\mathrm{j}=1}^{\mathrm{n}} [\mathrm{A}(\mathrm{i},\ \mathrm{j}) - \hat{\mathrm{A}}(\mathrm{i},\ \mathrm{j})]^2}) \quad (38)$$

Where;

R represents the maximum grey level intensity in the original image. $\mathrm{A}(\mathrm{i},\ \mathrm{j})$ and $\hat{\mathrm{A}}(\mathrm{i},\ \mathrm{j})$ represent the original image and restored image, respectively. m and n are the dimensions of the image.

The quantitative performances of the restoration methods are compared as per mean-squared error, normalized absolute error and image enhancement factor in addition to the peak signal to noise ratio. Mean-squared error measures the average deviation between the original image and the de-noised image. Normalized absolute error measures the error prediction accuracy of the filtered image. Lower values of MSE and NAE indicate that there is small deviation between the original image and the de-noised image, and consequently significant noise reduction is experienced. Image enhancement factor is a measure of the quality of the enhanced signal, and it is equal to the ratio of the sum of squared error before filtering to the sum of squared error after filtering. Hence, higher values of IEF indicates higher noise reduction and greater enhancement in the quality of the image as well as higher preservation of the edges. Mean-squared error, normalized absolute error and image enhancement factor can be expressed as follows [49, 50].

$$\mathrm{MSE} = \frac{1}{\mathrm{m} \times \mathrm{n}} \sum_{\mathrm{i}=1}^{\mathrm{m}} \sum_{\mathrm{j}=1}^{\mathrm{n}} [(\mathrm{A}(\mathrm{i},\mathrm{j}) - \hat{\mathrm{A}}(\mathrm{i},\mathrm{j})]^2 \tag{39}$$

$$\mathrm{NAE} = \frac{\sum_{\mathrm{i}=1}^{\mathrm{m}} \sum_{\mathrm{j}=1}^{\mathrm{n}} [|\mathrm{A}(\mathrm{i},\ \mathrm{j}) - \hat{\mathrm{A}}(\mathrm{i},\ \mathrm{j})|]}{\sum_{\mathrm{i}=1}^{\mathrm{m}} \sum_{\mathrm{j}=1}^{\mathrm{n}} \mathrm{A}(\mathrm{i},\mathrm{j})} \tag{40}$$

$$\mathrm{IEF} = \frac{\sum_{\mathrm{i}=1}^{\mathrm{m}} \sum_{\mathrm{j}=1}^{\mathrm{n}} [(\mathrm{N}(\mathrm{i},\mathrm{j}) - \mathrm{A}(\mathrm{i},\mathrm{j})]^2}{\sum_{\mathrm{i}=1}^{\mathrm{m}} \sum_{\mathrm{j}=1}^{\mathrm{n}} [(\hat{\mathrm{A}}(\mathrm{i},\mathrm{j}) - \mathrm{A}(\mathrm{i},\mathrm{j})]^2} \tag{41}$$

Where;

$\mathrm{N}(\mathrm{i},\mathrm{j})$ indicates the noisy image. $\mathrm{N}(\mathrm{i},\mathrm{j})$ represents the corrupted image with noise.

7. CONVENTIONAL MACHINE LEARNING METHODS

This section provides an overview of some of the existing machine learning methods reported in the literature that are used to validate the proposed method. Due to the paper size limitations, K-nearest neighbors and random forest are discussed in detail in the following sections. Discriminant analysis is a multivariate statistical method for the discrimination of a set of data points to a finite number of classes. More details about the discriminant analysis method can be found in Rathi and Palani [51], and Subasi and Gursoy [52]. Support vector machines is a supervised learning method that can be utilized in either classification or regression applications based on defining the optimum hyperplane by maximizing the margin between positive and negative classes. More information about the support vector machines can be adopted from Feng et al. [53], and Chen et al. [54].

7.1 K-Nearest Neighbors

K-Nearest Neighbors algorithm is a popular classification method in many applications because of its speed and relatively high convergence. It is based on the idea that similar feature vectors are located in close vicinity, whereas the classification of an input feature vector X is accomplished by identifying the K closest training vectors based on a suitable dimension metric. Usually, Euclidean distance is utilized as the dimension metric to measure the proximity of the feature vector the K nearest neighbors (instances). The feature vector X is then assigned to the class to which the majority of the K nearest neighbors belong. The Euclidean distance of a feature vector in the N-dimensional search space can be formulated as follows [55, 56].

$$\mathrm{ED} = \sqrt{\sum_{i=1}^{N} (X_{1i} - X_{2i})^2} \tag{42}$$

Where;

ED is the Euclidean distance. Thus, if K=1, the feature vector is mapped to the class C based on the 1-nearest neighbor to the feature vector. if K=5 classifying is performed as per the most common class among the K-neareset neighbors. For instance, if four nearest labels are mapped to class label C1 while one nearest neighbor is mapped to class label C2. Hence, the feature vector is mapped to class label C1.

7.2 Random Forest

Random Forest is a tree-based ensemble method proposed by Breiman [57] to overcome the shortcomings of the decision tree method. Random Forest includes large number of decision tree learners, which grow at the same time to reduce the bias and randomness of the

classification process. Since the random forest is an ensemble classification method, it involves the employment of bootstrap aggregating (bagging) to enhance the prediction accuracy of the classifier. Moreover, it boosts the performance of various decision trees through voting scheme. Each decision tree provides different results. Hence, the results of the classification are aggregated through majority of voting, i.e., the feature vector is assigned to the class which obtained the highest voting score. Thus, for training a dataset of N instances, N bootstrapped samples are extracted from the original dataset (randomly sampled with replacement). Then, a decision tree is developed based on the randomly selected dataset, whereas each decision tree is constructed using different N bootstrap samples obtained from the original dataset. The bootstrapping improves the robustness of the classifier because some feature vectors may exist more than once in the classification process or some feature vectors may not be trained at all. Hence, the random forest becomes less sensitive to the variations in the input dataset.

The subset that is not considered as a result of the bootstrap resampling is called “out of bag” (nearly one-third of the observations). This subset can be used to compute the error of the classifier, which eliminates the need for adding extra feature vectors. Random forest involves random feature selection, whereas in each split node of the decision tree, m random predictors (features or input variables) are selected from the M possible predictors. Random feature selection is introduced to minimize the correlation between the decision trees and to improve the prediction accuracy of each decision tree, which consequently enhances the prediction accuracy of the whole forest. The most common method is to select the split among the m possible predictors is called the Gini index. The Gini index is computed at each point of potential split of the predictors, whereas it can be defined as available selection measure which measures the impurity of a certain variable with respect to the output.

The decision tree splitting criterion is based on selecting the predictor which yields low Gini index. Impurity is a measure for how well the variable splits the data, whereas the lower the impurity, the better splitting of the data is. Gini index measures the probability that an instance is incorrectly classified if it were randomly classified as per the distribution of the labels within the node. For binary splitting, Gini index at a certain node can be defined using Equation (43). The bootstrap resampling and random feature selection are repeated for creating D decision trees until forming a forest of random decision trees [58, 59].

$$GI_{nn} = 1 - \sum_{i=1}^{2} [P_i^{\,2}] \tag{43}$$

Where;

GI_{nn} represents the Gini index at a certain node nn. P_i represents the relative proportion of instances belonging to the i-th category.

8. META-HEURISTIC OPTIMIZATION ALGORITHMS

Many bio-inspired meta-heuristic optimization algorithms have been developed recently to solve exhaustive optimization problems. The proposed method incorporates moth-flame optimization algorithm to design the self-adaptive hybrid restoration method. This method is compared against a set of meta-heuristics which include: genetic algorithm, particle swarm optimization algorithm, invasive weed optimization algorithm, differential evolution algorithm, modified differential evolution algorithm, grasshopper optimization algorithm and grey wolf optimization algorithm. It is Worthing mentioning that recently some models utilized a hybridization of meta-heuristics to enhance the performance of the proposed method by enhancing the search paradigm as found in Demirci and Yildiz [60] in addition to Yildiz et al.

[61]. Differential evolution algorithm is going to be discussed in-detail while the basic concepts of other meta-heuristics in addition to the needed references are provided in the following lines. Genetic algorithm (GA) is one of the most popular evolutionary algorithms, which was developed by John Holland in 1975. Genetic algorithm is based on two main processes. The first process is the selection of individuals for the production of the next generation. The second process is the manipulation of the selected individual to form the next generation by crossover and mutation. The selection paradigm identifies which chromosomes are chosen for reproduction and how many off springs are produced. The better individual has a higher chance of being a parent [62, 63].

Particle swarm optimization (PSO) algorithm is a population-based heuristic search algorithm that was originally developed by Eberhart and Kennedy in 1995 [64]. PSO algorithm belongs to the family of "swarm intelligence" algorithms in solving optimization problems, and it is inspired by the social behavior of birds flocking to the desired place in a multi-dimensional space. PSO algorithm is initiated by creating a population called "swarm" which is composed of individuals called "particles", whereas each particle adjusts its own flying based on its own flying experience and its companions' experience. Each particle represents a candidate solution in a multi-dimensional search space such that the status of the particle is characterized by its position and velocity and they are updated within each iteration [64-66].

Grasshopper optimization algorithm (GOA) is a newly-developed bio-inspired algorithm that was introduced by Saremi et al. in 2017 [67]. This algorithm is inspired by the swarming behaviour of grasshoppers in nature. The main aspects of the GOA are foraging, target pursuing and team behavior in both nymph and adulthood phases. In the larval phase, the grasshopper swarm exhibit short-length jumps associated with slow movement. On the other hand,

grasshopper swarm exhibit long-range and swift movements to obtain food sources from farming areas in the adulthood phase. The search process of the GOA is divided into two paradigms, namely exploration and exploitation, whereas the search agents are encouraged to move abruptly in the exploration phase while they tend to move locally in the exploitation phase. These two processes are simulated by the swarming behavior of grasshopper [68, 69]. More information about the grasshopper optimization algorithm can be adopted from Saremi et al. [67] and Zhang et al. [70].

Grey wolf optimization (GWO) algorithm is a recently-developed nature-inspired algorithm that was proposed by Mirjalili et al. in 2014 [71]. This algorithm is based on simulation of the behavior of a pack of grey wolves, which follow distinct steps while hunting in nature. Each pack hierarchy consists of four levels of grey wolves which are: alpha, beta, delta and omega. Alpha wolves are the leaders the pack and the ones responsible for making decisions. The next level in the hierarchy is the beta grey wolves, whereas they act as the subordinates of the alpha grey wolves and they support them in the decision-making process. Delta grey wolves follow the dictated orders of both alpha and beta grey wolves but they dominate the omega grey wolves. Delta grey wolves can be scouts, hunters, elders, sentinels or caretakers. Omega grey wolves are the least prioritized wolves in the hierarchy, whereas they have to submit to all other dominant wolves. They play the role of scapegoat and they are the last ones allowed to eat. In grey wolf optimization algorithm, a specific number of grey wolves explore the multi-dimensional search space to hunt a prey. The movement of grey wolves is influenced by search for prey, encircling prey, hunting and attacking prey operators. More information about the GWO algorithm can be found in [71, 72].

Differential evolution (DE) algorithm is an optimization algorithm that was introduced by Storn and Price in 1997 to search for the global solution of non-linear problems with non-differentiable objective functions [73]. The framework of the differential evolution algorithm is similar to the genetic algorithm. However, the classical mutation and crossover in the genetic algorithm are substituted by alternative mutation and crossover operators. Differential evolution algorithm is divided into five main stages which are: initialization, mutation, crossover, selection, and convergence criteria. Differential evolution algorithm starts by generating a population of D-dimensional parameter vectors (candidate solutions) of size NP. The computational steps of the DE algorithm are as follows [73-75]. The generation of individuals can be obtained using the following Equation

$$X_{i,G} = LB + rand[0,1] \times (UB - LB) \quad (44)$$

Where;

i denotes the population. G denotes the generation to which the population belongs to. LB, and UB represent two vectors of upper and lower bound for any decision variable, respectively. $rand[0,1]$ represent a uniformly distributed random number between 0 and 1.

The next step is the mutation, whereas the mutation vector is defined based on the combination of three randomly selected vectors. A vector in the current population is selected to be the target vector (parent). For each target vector ($X_{i,G}$) in the population, a mutant vector is created using the following Equation.

$$V_{i,G+1} = X_{r1,G} + F(X_{r2,G} - X_{r3,G}) \qquad r1 \neq r2 \neq r3 \quad (45)$$

Where;

r1, r2, and r3 represent three random and different indices between 1 and NP. The three random chosen vectors have to be different than the target vector. $V_{i,G+1}$ is the newly created mutant vector. F represents a mutation scale factor that control the amplification of differential variation between $X_{r2,G}$, and $X_{r3,G}$. Mutation scale factor is a real number between [0, 1].

Crossover is performed to diversify the current population by exchanging components of the target vector and the mutant vector. The trial vector (offspring) can be obtained using Equation (46). If the crossover rate is smaller than the random number, $V_{j,i,G+1}$ in the mutant vector is copied to the trial vector. Otherwise, $X_{j,i,G}$ in the target vector is copied to the trial vector.

$$U_{j,i,G+1} = \begin{cases} V_{j,i,G+1} & \text{if } CR \geq rand_j \\ X_{j,i,G}, & \text{if } CR < rand_j \end{cases} \quad (46)$$

Where;

CR represents crossover probability. $U_{j,i,G+1}$ represents trial vector. j represents index element for any vector. $rand_j$ denotes uniform random number between [0,1].

In the selection stage, the trial vector is compared with the target vector to determine if trial vector should be a member of the next generation $G + 1$ as shown in Equation (47). Assume the objective function to be minimized. The vector with lower objective function survives to the next generation. If the trial vector yields a lower objective function than the target vector, then the trial vector replaces the target vector in the next generation.

$$X_{i,G+1} = \begin{cases} U_{i,G+1} & \text{if } f(U_{i,G+1}) \leq f(X_{i,G}) \\ X_{i,G}, & \text{if} (U_{i,G+1}) > f(X_{i,G}) \end{cases} \quad (47)$$

Mutation, crossover, and selection are repeated in each generation until stopping criterion is satisfied, i.e., reaching maximum number of generations. In the modified differential evolution (MDE) algorithm, the Gaussian distribution is used to model the mutation scale factor because it offers a good balance between the exploration and exploitation of the search space.

9. MODEL IMPLEMENTATION

The images utilized to train and test the proposed method are captured from three bridge decks in Montreal and Laval, Canada using Sony DSC-H300 digital camera of 20.1 megapixel resolution. All the computations of the machine learning and optimization algorithms took place on a laptop with an Intel Core i7 CPU, 2.2 GHz and 16 GB of memory. Sample of the free-noise bridge defects images is shown in Figures 3 and 4. Sample of the degraded bridge defects images with different types of noises is depicted in Figures 5 and 6. Figures 5 and 6 contain images corrupted with Gaussian noise, speckle noise, salt and pepper noise, combination of Gaussian and speckle noises, combination of Gaussian and salt and pepper noises and combination of speckle and salt and pepper. As shown in Figure 4, the combination of noises amplifies the degradation in the qualities of the bridge defects images, which requires a higher capacity restoration method.

INSERT FIGURE 3

INSERT FIGURE 4

INSERT FIGURE 5

INSERT FIGURE 6

Three modules are developed for the noise detection and recognition in bridge defects images. Since the performance of the Elman neural network is substantially governed by number

of hidden and context layers, number of hidden and context neurons, type of transfer functions and weights and bias terms, the present study relies on the IWO algorithm to establish a proper setting for the tuning of the architecture of the Elman neural network and its parameters. One hundred sixty real-world images are used for training the noise detection module while the remaining forty are used for its testing in the split validation. The output of this module is whether the bridge defect image is noise free or corrupted with noise. The maximum numbers of hidden layers, hidden neurons, context layers and context neurons are equal to 5. Thus, the maximum length of the optimization problem is 304, which is considered as a large search space that substantiates the employment of exhaustive training mechanism. The parameters of the IWO algorithm are presented in Table 1. The number of iterations and the initial population size are assumed 200 and 100, respectively. The maximum and minimum numbers of seeds are 5 and 0, respectively. The initial and final standard deviations are assumed 0.5 and 0.001, respectively.

INSERT TABLE 1

The convergence of the ENN-IWO model for noise detection is shown in Figure 7. The least misclassification error achieved by ENN-IWO model equals to zero. Moreover, the optimization model stabilizes at iteration 96 which illustrates the superior search capability of the IWO algorithm. The optimum numbers of hidden and context layers are four while the optimum numbers of hidden and context neurons are five. The optimum transfer function is the hyperbolic tangent sigmoid function. The optimum transfer function is the hyperbolic tangent sigmoid function. The confusion matrix is the first step for the performance comparison. For example, the number of false positive instances in the ENN-IWO model is two. The confusion matrix enables the computation of true positive, false positive, true negative and false negative instances, which provides the platform for the calculation of the performance metrics.

INSERT FIGURE 7

The performances of the six machine learning models as per split validation and 10-fold cross validation are shown in Tables 2 and 3, respectively. As shown in Tables 2 and 3, the proposed noise detection model achieved the achieved the highest classification accuracies as per split validation and 10-fold cross validation. Support vector machines attained the second best performance. On the other hand, discriminant analysis and artificial neural network achieved the least performance. For instance, as per the cross-validation model, the proposed noise detection model is capable of attaining accuracy, sensitivity, specificity, precision, F-measure and Kappa coefficient of 98.72%, 99.65%, 98.52%, 93.12%, 96.39% and 0.956, respectively. Nevertheless, accuracy, sensitivity, specificity, precision, F-measure and Kappa coefficient of ANN model are equal to 90.73%, 64.77%, 97.06%, 86.38%, 74.18% and 0.687, respectively.

INSERT TABLE 2

INSERT TABLE 3

For the separate noise recognition module, the output of this model is if the image contains speckle, Gaussian, salt and pepper or doesn't contain noise. The neural network is composed of four output neurons for the four previous states, whereas the output is expressed in the form of a binary vector. One hundred images are used for training the separate noise recognition module while the thirty five images are used for its testing in the split validation. The decision variables of the proposed ENN-IWO model are as follows: maximum numbers of hidden and context layers are 10 while the maximum numbers of hidden and context neurons are 10. Thus, maximum length of the optimization problem is 2137. The number of iterations is assumed 250 while the initial population size is assumed 150. The maximum and minimum numbers of seeds are 5 and 0, respectively. The initial and final standard deviations are assumed 0.5 and 0.001,

respectively. The convergence of the ENN-IWO model for separate noise recognition is depicted in Figure 8. The least misclassification error achieved by ENN-IWO model is equal to 0.05. In addition to that, the proposed optimization model stabilizes 191, which exemplifies the higher capacity of the proposed model to search for the optimum structure and parameters of the ENN.

INSERT FIGURE 8

The optimum structure of the ENN is one hidden layer, one context layer, seven hidden neurons and seven context neurons. The optimum transfer function is the hyperbolic tangent sigmoid function. The confusion matrix of the classification provided by ENN-IWO model is shown in Table 4. The total numbers of true positive instances and true negative instances for all the classes are 129 and 393, respectively. The performance comparisons for the five classification models using the split and 10-fold cross validation are described in Tables 5 and 6, respectively. As shown in Table 5 and 6, the proposed separate noise recognition model outperformed other classification models for the six performances indicators in both split validation and 10-fold cross validation. Random forest achieved the second best performance, while discriminant analysis and artificial neural network attained the lowest values for the performance indicators. In the cross validation model, the proposed ENN-IWO model attained accuracy, sensitivity, specificity, precision, F-measure and Kappa coefficient of 95.28%, 95.24%, 98.07%, 95.25%, 95.43% and 0.935, respectively. On the other hand, accuracy, sensitivity, specificity, precision, F-measure and Kappa coefficient of discriminant analysis were equal to 83.45%, 83.41%, 93.84%, 83.42%, 83.57% and 0.768, respectively.

INSERT TABLE 4

INSERT TABLE 5

INSERT TABLE 6

For the combined noise recognition module, the output layer is composed of seven neurons represented in the form of the binary vector. The output of this model determines if the image is noise free or if the image is corrupted with separate noise or corrupted with any of the different combinations of the noises. One hundred sixty images are used for training the combined noise recognition module while the forty images are used for testing purposes in the split validation. The decision variables of the proposed ENN-IWO model are as follows: maximum numbers of hidden and context layers are 10 while the maximum numbers of hidden and context neurons are 10. As such, the maximum length of the optimization model is 2170, which is deemed as an exhaustive search space to explore. The parameters of the IWO algorithm in the current module are the same as the ones in the separate noise recognition module.

The convergence of the ENN-IWO model for combined noise recognition is depicted in Figure 9. As shown in Figure 9, the minimum misclassification error achieved by ENN-IWO model is equal to 0.1625, whereas the model stabilizes at iteration 180. The optimum structure of the ENN is one hidden layer, one context layer, ten hidden neurons and ten context neurons. The optimum transfer function is the linear function. The confusion matrix of the classification attained by ENN-IWO model is shown in Table 7. As shown in Table 7, the total numbers of true positive instances and true negative instances for all the classes are 151 and 1035, respectively. The comparison between the six classification models using split validation and cross validation are described in Tables 8 and 9. As shown in Tables 8 and 9, it can be inferred that the proposed ENN-IWO model outperformed other classifiers for the six performances indicators in both split validation and 10-fold cross validation. For the 10-fold cross validation, the proposed ENN-IWO model achieved accuracy, sensitivity, specificity, precision, F-measure and Kappa coefficient of 84.26%, 84.22%, 97.21%, 84.45%, 84.46% and 0.811, respectively. On the other hand, accuracy,

sensitivity, specificity, precision, F-measure and Kappa coefficient of discriminant analysis are equal to 75.27%, 81.33%, 95.05%, 75.25%, 78.32% and 0.702, respectively.

INSERT FIGURE 9

INSERT TABLE 7

INSERT TABLE 8

INSERT TABLE 9

Parametric and non-parametric tests were conducted to provide a thorough assessment of the noise classification models by examining the significant difference in the accuracies among the different classifiers, whereas the significance level (α) is set to be 0.05. The performed statistical tests examine the null hypothesis (H_0), which implies that there is no significant difference between the classification results obtained from each pair of classifiers. On the other contrary, the alternative hypothesis (H_1) implies that there is a significant difference between the classification results obtained from each pair of classifiers. If the $P-\text{value}$ is less than the significance level, then the null hypothesis is rejected in favor of the alternative hypothesis. Nonetheless, if the $P-\text{value}$ is more than the significance level, thus the null hypothesis is accepted. The Student's t-test, Wilcoxn test, Mann-Whitney-U test, Kruskal–Wallis test, binomial sign test, and Mood's median test of the noise classification models are shown in Tables 10 and 11. Results indicate that the $P-\text{values}$ of the pairs (ENN-IWO, discriminant analysis), (ENN-IWO, K-nearest neighbors), (ENN-IWO, random forest), (ENN-IWO, support vector machines) and (ENN-IWO, artificial neural network) for all the tests are less than 0.05, which implies that there are statistically significant differences between the performance of the proposed noise classification model, and other classification models.

INSERT TABLE 10

INSERT TABLE 11

Friedman test and Friedman's aligned ranks test were employed to investigate whether there are statistical differences among the set of noise detection and recognition models. The average rankings of the noise classification model obtained from the Friedman and Friedman's aligned ranks tests are presented in Table 12. It is worth mentioning that a smaller average ranking value implies a better noise classification model. As shown in Table 12, ENN-IWO achieved the best ranking followed by support vector machines based on the two tests. Discriminant analysis and artificial neural network achieved the lowest rankings based on Friedman test and Friedman's aligned ranks test, respectively. The $P-$value of the Friedman test and $P-$value of the Friedman's aligned ranks test are equal to zero, which indicates that there are statistical significant differences among the noise classification models. As such, Nemenyi, Holms and Finner post hoc statistical tests are utilized to investigate if the ENN-IWO model is significantly better than the remaining noise classification models. The $P-$values of the ENN-IWO model based on Nemenyi, Holms and Finner tests are shown in Table 13. As can be seen, the $P-$values of the pairs (ENN-IWO, discriminant analysis), (ENN-IWO, K-nearest neighbors), (ENN-IWO, random forest), (ENN-IWO, support vector machines) and (ENN-IWO, artificial neural network) for all the post hoc tests are less than 0.05. It is worth mentioning that the developed noise classification is the only model which provided statistical significant better performance against the reminder of noise classification models with respect to all tests. This indicates the ENN-IWO model is a statistically better noise classification model than other models.

INSERT TABLE 12

INSERT TABLE 13

The different classification models for noise detection and recognition were compared with respect to the average running time for both training and testing. Average run time represents the average run time per fold. The results of the comparisons are presented in Tables 14, 15 and 16. For instance, in the training process of noise detection module, discriminant analysis had the shortest computational time of 3.36 seconds while the proposed model had the longest computational time of 1203.56 seconds. It can be inferred that the proposed ENN-IWO model requires more computational time to train the input dataset. However, almost most of the execution time is spent in the learning process of the underlying pattern between inputs and outputs. Moreover, the classification time is nearly the same for all detection and recognition models. The long computational training time of the proposed model can be explained by the fact that it has capacity to optimize all of the variables of the Elman neural network including both its architecture and parameters. The structure and parameter learning is a very exhaustive search process, which requires more processing time to explore the search space efficiently. It is also worth mentioning that the classification time is a more useful performance metric than training time in practical applications because the training process is only needed to be performed once also, the usage of a higher-performing computer can decrease the computational time of the training process. Although the discriminant analysis, K-nearest neighbors, random forest, support vector machines are not time-exhaustive models. However, their classification accuracies are low, which hinders their usage in noise detection and recognition. In view of above comparisons, the proposed ENN-IWO model required nearly the same computational time for classification as other models. Moreover, it achieved significant higher classification performance than other models. As such, the proposed ENN-IWO model serves as a better alternative in the noise detection and recognition of bridge defects images.

INSERT TABLE 14

INSERT TABLE 15

INSERT TABLE 16

The second model is the restoration of bridge defects images identified from the previous stage. The output of this model is a filtering protocol, which incorporates the optimum design of filters for each noise type. In order to provide a fair comparison between the different meta-heuristic optimization algorithms, the population size and number of iterations are assumed 10 and 40, respectively. Different initializations of parameters were experimented for the different meta-heuristics in order to search for their optimum values. Each meta-heuristic was run ten times independently in order to avoid unstable solutions due to random initialization of population. The proposed restoration model was compared with other models reported in the literature based on the de-noising performance of ten different types of images to examine its robustness in restoration of degraded images.

In the genetic algorithm, tournament selection is the parent selection strategy. Two-point crossover is utilized, and the crossover rate is assumed 0.8. Mutation rate is assumed 0.1. For the particle swarm optimization, the cognitive learning and social parameters are assumed two. The inertia weight is assumed 0.5. The initial standard deviation and final standard deviation are assumed 0.5 and 0.001, respectively. The maximum and minimum numbers of seeds are 5 and 0, respectively. For the differential evolution algorithm, the crossover probability is assumed 0.2. The mutation is assumed to follow a uniform distribution between 0.2 and 0.8. For the modified differential evolution algorithm, the mutation is assumed to follow a normal distribution with a mean and standard deviation equal to 0.5 and 0.2, respectively. For the grasshopper optimization algorithm, the maximum and minimum values of deceleration of grasshoppers approaching the

food source and consuming it are assumed 1 and 0.00004, respectively. In the grey wolf optimization algorithm, the trade-off parameter which controls the balance between exploration and exploitation is assumed to be linearly decreasing from 2 to 0. The logarithmic spiral motion constant is assumed 1 while the convergence constant is assumed to be decreasing from -1 to -2 in the moth-flame optimization algorithm.

A set of comparisons are conducted for the different possible combination of noises in order to investigate the robustness of the proposed restoration model in filtering of degraded images. For the Gaussian noise, the convergence curves of the different meta-heuristic-based restoration models of “Image 1”. The optimization problem is a maximization problem of PSNR, thus a negative sign to convert it to a minimization problem because it is often more easier to deal with cost functions. As shown in Figure 10, the MFO algorithm outperformed other optimization algorithms, whereas it achieved PSNR of 25.29. PSO algorithm provided the second best performance with PSNR of value 25.28. A performance comparison between the different restoration models of Gaussian noise is described in Table 17. Based on the MFO algorithm, the optimum filter design is to apply Wiener filter of size 3×3. The proposed restoration model achieved better de-noising results when compared to other optimization methods. For instance, the MFO algorithm achieved PSNR, MSE, NAE and IEF of 25.29, 185.75, 0.074 and 5.16, respectively. PSO algorithm attained PSNR, MSE, NAE and IEF equal to 25.28, 192.31, 0.076 and 3.34, respectively. Mode filter achieved the lowest de-noising results such that PSNR, MSE, NAE and IEF equal to 16.73, 1381.15, 0.211 and 0.47, respectively. The restored images based on the optimization-based models and conventional filtering models are shown in Figures 11, 12, 13, 14 and 15. By visually investigating the images, it is clear that the MFO algorithm provided better restoration results when compared to other models.

INSERT FIGURE 10

INSERT TABLE 17

INSERT FIGURE 11

INSERT FIGURE 12

INSERT FIGURE 13

INSERT FIGURE 14

INSERT FIGURE 15

The convergence curves of the different meta-heuristic-based restoration models for salt and pepper noise of "Image 2" are depicted in Figure 16. MFO algorithm achieved the highest PSNR of 30.11 while MDE algorithm achieved the second highest PSNR of 30.07. The performances of the different restoration models of Gaussian noise are shown in Table 18. The optimum filter design is median filter of size 3×3 based on MFO algorithm, whereas it provided PSNR, MSE, NAE and IEF of 30.11, 65.29, 0.038 and 14.24, respectively. The PSNR, MSE, NAE and IEF of MDE algorithm are 30.07, 67.35, 0.039 and 13.61, respectively. Mode filter had the lowest de-noising results such that PSNR, MSE, NAE and IEF equal to 13.74, 2748.22, 0.194 and 0.32, respectively. A clearer visual comparison is presented between the different restoration models in Figures 17, 18, 19, 20 and 21. The output of the proposed restoration model provided superior filtering results, which demonstrates its capabilities in removing the salt and pepper noise.

INSERT FIGURE 16

INSERT TABLE 18

INSERT FIGURE 17

INSERT FIGURE 18

INSERT FIGURE 19

INSERT FIGURE 20

INSERT FIGURE 21

For the speckle noise, the convergence curves of the different meta-heuristic-based restoration models of "Image 3" are illustrated in Figure 22. MFO algorithm achieved the highest PSNR of 24.57 while MDE provided the second highest performance such that the PSNR is 24.47. A performance evaluation of the different restoration models of speckle noise are shown in Table 19. It can be inferred that, the optimum filter design is Lee filter of size 3×3 based on MFO algorithm, whereas it provided PSNR, MSE, NAE and IEF of 24.57, 228.9, 0.082 and 4.13, respectively. MDE algorithm had PSNR, MSE, NAE and IEF equal to 24.49, 270.5, 0.088 and 4.05, respectively. Mode filter achieved the lowest de-noising results such that PSNR, MSE, NAE and IEF equal to 15, 2055.63, 0.264 and 0.54, respectively. The restored images using optimization-based models are shown in Figures 23, 24 and 25. The restored images using conventional filtering models are shown in Figures 26 and 27. These images provide a visual understanding of the quality of the performances of restoration models. It can be inferred that the proposed restoration model provided the highest de-noising capabilities of speckle noise.

INSERT FIGURE 22

INSERT TABLE 19

INSERT FIGURE 23

INSERT FIGURE 24

INSERT FIGURE 25

INSERT FIGURE 26

INSERT FIGURE 27

The convergence curves of the different meta-heuristic-based restoration models for the combination of Gaussian and speckle noises of “Image 4” are shown in Figure 28. As shown in Figure 28, MFO algorithm achieved very promising results, whereas the PSNR is 25.14 while the GWO algorithm achieved the second highest PSNR of 24.98. The optimum design of the proposed restoration model is Wiener filter of size 3×3 followed by Lee filter of size 3×3. A comparative analysis of the different restoration models is shown in Table 20. The proposed model achieved the highest filtering performance, whereas PSNR, MSE, NAE and IEF are 25.14, 158.51, 0.084 and 7.87, respectively. GWO algorithm achieved the second best performance such that PSNR, MSE, NAE and IEF are 24.98, 155.04, 0.086 and 6.82, respectively. Mode filter achieved the lowest de-noising performance such that PSNR, MSE, NAE and IEF are 15.06, 2029, 0.343 and 0.68, respectively. The output of the optimization-based restoration models is depicted in Figures 29, 30 and 31while the resorted images using the conventional restoration models are shown in Figures 32 and 33. By investigating these images, it can be inferred that the proposed restoration model achieved the best de-noising capabilities of the combination of Gaussian and speckle noises.

INSERT FIGURE 28

INSERT TABLE 20

INSERT FIGURE 29

INSERT FIGURE 30

INSERT FIGURE 31

INSERT FIGURE 32

INSERT FIGURE 33

For the combination of Gaussian and salt and pepper noises, the convergence curves of different meta-heuristic-based restoration models of “Image 5” are presented in Figure 34. As shown in Figure 34, The MFO algorithm yielded the highest PSNR of 25.06 such that optimum filter obtained by it is median filter of size 3×3. GOA yielded the second highest PSNR of 25.05, whereas the optimum filter obtained by it is Frost filter of size 3×3. A comparison of the performances of the different restoration models is shown in Table 21. As shown in Table 21, the proposed model outperformed other models, where the PSNR, MSE, NAE and IEF achieved by the proposed method are equal to 25.16, 180.19, 0.095 and 8.61, respectively. Frost filter of size 4×4 obtained the highest filtering performance among the conventional restoration models, whereas the PSNR, MSE, NAE and IEF are 24.42, 226.85, 0.103 and 5.63, respectively. The restoration outcome of the different optimization-based models is shown in Figures 35, 36 and 37 while the restoration outcome of the conventional models is presented in Figures 38 and 39. As shown in these figures, the capacity of the restoration is significantly improved by applying the proposed model. This highlights the capacity of the proposed restoration model in removing noises from images corrupted by combination of Gaussian and salt and pepper noises.

INSERT FIGURE 34

INSERT TABLE 21

INSERT FIGURE 35

INSERT FIGURE 36

INSERT FIGURE 37

INSERT FIGURE 38

INSERT FIGURE 39

For the images corrupted with a combination of speckle and salt and pepper noises, the convergence curves of the different meta-heuristic-based restoration models of “Image 6” are presented in Figure 40. As shown in Figure 40, MFO algorithm achieved the highest PSNR followed by MDE algorithm and then the DE algorithm. The values of PSNR obtained by the MFO algorithm, MDE algorithm and DE algorithm are 22.98, 22.93 and 22.82, respectively. A performance comparison of the different restoration models is presented in Table 22. The proposed restoration demonstrated the highest filtering performance while mode filter achieved the least performance among the restoration models. The PSNR, MSE, NAE and IEF of the MFO algorithm are 22.98, 290.61, 0.091 and 5.91, respectively. Lee filter of size 4×4 yielded the highest filtering performance among the conventional restoration models, whereas the PSNR, MSE, NAE and IEF are 21.66, 360.59, 0.099 and 4.82, respectively. The restored images using the optimization-based models are presented in Figures 41, 42 and 43. The restored images using the conventional restoration models are shown in Figures 44 and 45. By visually investigating the restored images, it is concluded that the proposed model provides an efficient alternative to restore images corrupted with combination of speckle and salt and pepper noises.

INSERT FIGURE 40

INSERT TABLE 22

INSERT FIGURE 41

INSERT FIGURE 42

INSERT FIGURE 43

INSERT FIGURE 44

INSERT FIGURE 45

In view of the above comparisons with respect to the different types of noises, the proposed restoration model provided a consistent superior filtering capability than other models, which aids in overcoming the inconsistencies of other restoration models, such that some models perform well in some types of de-noising problems. However, they fail to deal with other types of de-noising problems. For example, DE algorithm provided efficient results when dealing with salt and pepper noise. However, it didn't perform well when dealing with the combination of Gaussian and speckle noises. Moreover, median filter performed well when dealing with salt and pepper noise. On the other hand, it failed to deal with speckle noise.

The overall performance of the different proposed restoration model is investigated through comparison against other restoration models as shown in Table 23. These models are evaluated as per the average of the peak signal to noise ratio (APSNR), average of mean-squared error (AMSE), average of normalized absolute error, (ANAE) and average of image enhancement factor (AIEF) for the ten images. The proposed restoration model achieved superior de-noising results when compared to other optimization-based restoration models and conventional restoration models. MDE achieved the second best performance followed by DE algorithm. On the other hand, non-linear programming-based model attained the least restoration performance among the optimization-based models. The APSNR, AMSE, ANAE and AIEF of the MFO algorithm are 25.36, 176.32, 0.059 and 7.18, respectively. MDE algorithm attained APSNR, AMSE, ANAE and AIEF of 25.23, 177.59, 0.059 and 6.9, respectively. For the DE algorithm, the values of APSNR, AMSE, ANAE and AIEF are 24.94, 180.71, 0.06 and 6.82, respectively. The APSNR, AMSE, ANAE and AIEF of the non-linear programming are 20.3, 415.34, 0.099 and 3.09, respectively. This highlights that the evolutionary algorithms provide better filtering

performance when compared to exact optimization models, which illustrates that exact optimization algorithms fail to solve discrete and complex optimization problems.

INSERT TABLE 23

For the conventional restoration models, Wiener and lee filter are the best two performing restoration models while mode filter achieved the least filtering performance. The APSNR, AMSE, ANAE and AIEF of the Wiener filter are 22.73, 290.04, 0.093 and 4.26, respectively. Mode filter achieved APSNR, AMSE, ANAE and AIEF of 14.14, 2701.5, 0.284 and 0.53, respectively. This manifests that the proposed restoration model using MFO algorithm provided holistic and consistent superior filtering capacity over the conventional restoration models.

The utmost objective of the proposed method is to develop a filtering protocol, which incorporates the optimum filters to deal with each type of the different noises. Table 24 describes the optimum filter(s) for each noise type(s). As shown in Table 24, conventional filters of size 3×3 are more efficient in removing separate noises than the combination of noises. Moreover, a filter of size 3×3 provides better de-noising outcome than a filter of size 4×4. In addition to that, it is worth mentioning that, the application of a set of filters in a certain sequence can improve the restoration process when compared to single filters in the case of images corrupted with a combination of noises. For example, the optimum hybrid filter in the case of images corrupted by a combination of Gaussian and speckle noises is to apply Wiener filter of size 3×3 followed by Lee filter of size 3×3. Moreover, the optimum hybrid filter in the case of images corrupted with a combination of speckle and salt and pepper noises is to apply Lee filter of size 3×3 followed by Wiener filter of size 3×3. This also demonstrates that the application of two filters in two different sequences yields different restoration results.

INSERT TABLE 24

Parametric and non-parametric tests were performed to evaluate the statistical significant differences in the filtering capacities of the different meta-heuristic-based restoration models at a significance level of 0.05. The Student's t-test, Wilcoxn test, Mann-Whitney-U test, Kruskal–Wallis test, binomial sign test, and Mood's median test of the meta-heuristics-based restoration models are shown in Table 25 and 26. As can be seen, $P-$values of the pairs (MFO, DE), (MFO, MDE), (MFO, PSO), (MFO, IWO), (MFO, GOA), (MFO, GWO) and (MFO, GA) are less than 0.05 for all the previously-mentioned statistical tests. This evinces that there are significant differences in the filtering capacities of the proposed restoration model with respect to other meta-heuristic-based restoration models.

INSERT TABLE 25

INSERT TABLE 26

Table 27 displays the average rankings of the different meta-heuristic-based restoration models using Friedman test and Friedman's aligned ranks test, respectively. Moth-flame optimization algorithm yielded the best ranking followed by modified differential evolution algorithm while grasshopper optimization algorithm achieved the least ranking as per Friedman test and Friedman's aligned ranks test. The average ranking values of the moth-flame optimization algorithm, modified differential evolution algorithm and grasshopper optimization algorithm based on Friedman test are 1, 3.88 and 6.38, respectively. The $P-$values of the Friedman test and Friedman's aligned ranks test are equal to zero, which indicates that there are statistical significant differences among the meta-heuristic-based restoration models. Thus, Nemenyi, Holms and Finner post hoc statistical tests are applied to examine if the developed moth-flame restoration model is significantly better than other restoration models. The $P-$values of the moth-flame-based restoration model using Nemenyi, Holms and Finner tests are

shown in Table 28. It can be inferred that the $P-$values of the pairs (MFO, DE), (MFO, MDE), (MFO, PSO), (MFO, IWO), (MFO, GOA), (MFO, GWO) and (MFO, GA) are less than 0.05 for all post hoc tests. It should be mentioned that the developed restoration model is the only restoration model which provided statistical significant better filtering performance over other meta-heuristic-based restoration models for all tests. In view of the above comparisons, it can be concluded that the proposed restoration model introduced significant superior consistent and overall filtering results over other restoration models.

INSERT TABLE 27

INSERT TABLE 28

The different restoration models were evaluated as per the average computational time of both training and restoration. The average computational times of the training and restoration are presented in Table 29. As shown in Table 29, non-linear programming required less training time with respect to other optimization-based models. On the other hand, while the IWO had the longest computational time of 1304.62 seconds. The proposed restoration model required 915.38 seconds to select the optimum restoration process based on MFO algorithm. It is worth mentioning that optimization-based restoration models had longer computational time compared to the conventional time. However, most of the computational time is spent in the training process. In addition to that, most of the time spent in the restoration process is nearly equal among the different restoration models. The long computational time of the proposed restoration model is resulting from its capability to optimize the number of filters, types of filters, sequence of filters and the governing tuning parameters of these filters. This is deemed as a large search space problem, which requires an exhaustive search engine to explore the space. Moreover, the time of restoration process is a more practical performance indicator in bridge defects

recognition applications because the training process is only done once, and then the optimum filter(s) obtained from the restoration model are used to de-noise the images based on the type of noise that corrupts the images. As such, the different restoration models consume nearly the same time in the restoration process. Meanwhile, the proposed restoration model achieved significant enhancement in the filtering process when compared to other restoration models. Thus, it can provide an efficient alternative for the restoration of bridge defects images.

INSERT TABLE 29

In order to investigate the implication of the restoration process on the segmentation of the bridge defects, the proposed restoration model is compared with the median filter of size 4×4 based on their influence on the quality of segmentation of spalling in reinforced concrete bridges. The case study is "Image 6" corrupted with Gaussian noise. The segmentation process of spalling is performed using fuzzy C-means clustering algorithm, which is considered as a well-performing segmentation method that proved its capability in detecting bridge defects. More details about fuzzy C-means clustering algorithm can be found in keskin [76]. The results of the segmentation as per the proposed restoration model and median filter are shown in Figure 46. As shown in Figure 46, it is clearly visible that the segmented image using fuzzy C-means clustering algorithm based on the de-noised images of the proposed restoration model yields superior segmentation performance when compared to other methods. As such, the self-adaptive two-tier optimization-based method provides a consistent, holistic and remarkable improvement in recognition of noise as well as restoration of degraded bridge defects images with noise as per the different levels of comparison. This leads to better detection and evaluation of bridge defects images, which eventually leads to the establishment of more accurate image-based condition assessment models and reliable maintenance decision-making models.

INSERT FIGURE 46

10. CONCLUSION

Routine inspections are diagnostic methods to evaluate the condition of reinforced concrete bridges. Nevertheless, current visual inspection-based methods are labor-intensive, and provide biased and subjective judgments. This requires the development of a machine vision-based method for the automatic assessment of bridge defects. Noise is undesirable random variation in the brightness or intensity of the image, which significantly influences the attributes of bridge defects images. In this regard, the absence or the inefficiency of a restoration method of bridge defects images leads to error-prone deterioration models and maintenance intervention actions. As such the present study introduces a self-adaptive two-tier method for the restoration of bridge defects images.

The developed method is envisioned on two main stages which are: automatic recognition of noise, and restoration of degraded bridge defects images. In the first model, a hybrid Elman neural network-invasive weed optimization model is developed to detect and recognize the noises in bridge defects images based on three different modules. A variable-length optimization problem is designed to enhance the search capacity of the ENN-IWO model through both parameter and structural learning of the Elman neural network. The recognition capabilities of the proposed ENN-IWO model are examined by comparison with other well-performing machine learning models such as discriminant analysis, artificial neural network, random forest, support vector machines and K-nearest neighbors. The proposed noise recognition model significantly outperformed other classifiers. For instance, in the separate noise recognition module the developed model achieved accuracy, sensitivity, specificity, precision, F-measure

and Kappa coefficient of 95.28%, 95.24%, 98.07%, 95.25%, 95.34%. 95.43% and 0.935, respectively.

After mapping each image to the designated type of noise, a moth-flame optimization-based restoration model is developed to restore bridge defects corrupted with noise. The developed restoration model outperformed other optimization-based and conventional restoration models, whereas it achieved the APSNR, AMSE, ANAE and AIEF of 25.36, 176.32, 0.059 and 7.18, respectively. The final outcome of the proposed method is a filtering protocol, which enables decision-makers to deal with different types of noises in bridge defects images. In the developed protocol, hybrid combinations of filters are required to be applied in some cases of combinations of noises. This exemplifies that these combinations of noises amplify the degradation in bridge defects images, which necessitates the application of a higher capacity restoration model. It is expected that the developed method can enhance the automatic evaluation of bridge defects, which enables establishing more accurate image-based condition assessment models and enhancing the decision-making process in the bridge management systems.

Conflict of interest: The authors declare that they have no conflict of interest.

REFERENCES

1. National Research Council Canada (2013) Critical Concrete Infrastructure: Extending the Life of Canada's Bridge Network <http://www.nrc-cnrc.gc.ca/ci-ic/article/v18n1-5> (20.12.2016).
2. Statistics Canada (2009) Age of Public Infrastructure: A Provincial Perspective < http://www.statcan.gc.ca/pub/11-621-m/11-621-m2008067-eng.htm> (20.12.2016).

3. Karibasappa K G, Karibasappa K (2015) AI Based Automated Identification and Estimation of Noise in Digital Images. Advances in Intelligent Systems and Computing, Springer, pp 49-60.
4. Chuah J H, Khaw H Y, Soon F C, Chow C (2017) Detection of Gaussian Noise and Its Level using Deep Convolutional Neural Network. In: Proceedings ofthe 2017 IEEE Region 10 Conference (TENCON), Penang, Malaysia, 5-8 November, pp 2447-2450.
5. Turajlic E, Begovi A (2019) Application of Artificial Neural Network for Image Noise Level Estimation in the SVD domain. Electronics 8(163):1-20.
6. Vasuki P, Bhavana C, Lakshmi D E, Roomi S M M (2012) Automatic Noise Identification in Images Using Moments and Neural Network. In: 2012 International Conference on Machine Vision and Image Processing (MVIP), Nadu, India, 14-15 December.
7. Gupta M, Taneja H, Chand L (2018) Performance Enhancement and Analysis of Filters in Ultrasound Image Denoising. PROCEDIA COMPUT SCI 132:643–652.
8. Verma R, Mehra R (2016) PSO Algorithm based Adaptive Median Filter for Noise Removal in Image Processing Application. INT J of ADV COMPY SCI APPL *7*(7):92–98.
9. Dass R (2018) Speckle Noise Reduction of Ultrasound Images Using BFO Cascaded with Wiener Filter and Discrete Wavelet Transform in Homomorphic Region. PROCEDIA COMPUT SCI 132:1543–1551.
10. Kumar N (2017) Image Restoration in Noisy Free Images Using Fuzzy Based Median Filtering and Adaptive Particle Swarm Optimization - Richardson-Lucy Algorithm. INT J INTELL ENG SYST 10(4): 50–59.
11. Wang Y, Adhmai R, Fu J (2015) A novel supervised learning algorithm for salt-and-pepper noise detection. INT J MACH LEARN 6(4): 687–697.

12. Zhao Y, Xin J. M, Sun, L. X (2017) Reconstructing images corrupted by noise based on D–S evidence theory. INT J MACH LEARN 8(2): 611–618.
13. Ma J, Tian D, Gong M (2014) Fuzzy clustering with non-local information for image segmentation. INT J MACH LEARN CYB 5(6): 845–859.
14. Tong X, Guo J, Ling Y, Yin Z (2011) A New Image-Based Method for Concrete Bridge Bottom Crack Detection. In: 2011 International Conference on Image Analysis and Signal Processing, Wuhan, China, 21-23 October.
15. Adhikari R S, Moselhi, O, Bagchi, A (2014) Image-based retrieval of concrete crack properties for bridge inspection. AUTOMAT CONSTR 39(1):180–194.
16. Yao C, Tao M, Xiaojie W, Feng L (2016) A Bridge Crack Image Detection and Classification Method Based On Climbing Robot. In: Proceedings of the 35th Chinese Control Conference, Chengdu, China, 27-29 July, pp 4037-4042.
17. Lee J H, Jin S S, Kim I H, Jung H J (2017) Development of crack diagnosis and quantification algorithm based on the 2D images acquired by Unmanned Aerial Vehicle (UAV). In: The 2017 Congress on Advances in Structural Engineering and Mechanics, Seoul, Korea, 28 August- 1 Septemeber.
18. Ellenberg A, Kontsos S, Moon F, Bartoli I (2016) Bridge related damage quanti fi cation using unmanned aerial vehicle imagery. STRUCT CONTROL HLTH 23:1168–1179.
19. Lei B, Wang N, Xu P, Song G (2018) New Crack Detection Method for Bridge Inspection Using UAV Incorporating Image Processing. J AEROSPACE EN 31(5): 1–13.
20. Li Y, Zhao W, Zhang X, Zhou Q (2018) A Two-Stage Crack Detection Method for Concrete Bridges Using Convolutional Neural Networks. IEICE T INF SYST E101(12):3249–3252.
21. Dinh T H., Ha Q P, Tranhiepdinhutseduau E (2016) Computer Vision-based Method for

Concrete Crack Detection. In: 14th International Conference on Control, Automation, Robotics and Vision, Phuket, Thailand, 13-15 November.

22. Wang Y, Zhang J Y, Liu J X, Zhang Y, Chen Z P., Li C G, He K, Yan R B (2019) Research on Crack Detection of Algorithm of the Concrete Bridge Based on Image Processing. PROCEDIA COMPUT SCI 154: 610–616.
23. Ho H, Kim K, Park Y, Lee J (2013) An efficient image-based damage detection for cable surface in cable-stayed bridges. NDT&E INT 58:18–23.
24. Lee J H, Lee J M, Kim H J, Moon Y S (2008) Machine Vision System for Automatic Inspection of Bridges. In: 2008 Congress on Image and Signal Processing, Sanya, China, 27-30 May, pp 363–366.
25. Pavithra D, Saranya T, Prakash K, Soundarya G (2018) Electronic Crack Detection on Concrete. INT J ADV SCI ENG RES 3(1):515–521.
26. Rodríguez-Fdez I, Canosa A, Mucientes M, Bugar A (2015) STAC : a web platform for the comparison of algorithms using statistical tests. In: 2015 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), Istanbul, Turkey, 2-5 August.
27. Ning J, Zhang C, Sun P, Feng Y (2019) Comparative Study of Ant Colony Algorithms for Multi-Objective Optimization. Information 10(11): 1–19.
28. Nancy E, Kaur E S (2013) Comparative Analysis and Implementation of Image Enhancement Techniques Using MATLAB. INT J COM SCI MOB COMP 2(4), 138–145.
29. Hoshyar A N, Al-jumaily A, Hoshyar A N (2014) Comparing the Performance of Various Filters on Skin Cancer Images. PROCEDIA COMPUT SCI 42: 32–37.
30. Tania S, Rowaida R (2016) A Comparative Study of Various Image Filtering Techniques for Removing Various Noisy Pixels in Aerial Image. INT J SIGNAL PROCESS IMAGE

PROCESS PATTERN RECOGN 9(3):113–124.

31. Wu Q, Lee J, Park M, Park C, Kim I (2014) A study on Development of Optimal Noise Filter Algorithm for Laser Vision System in GMA Welding. PROCEDIA ENG 97:819–827.
32. Hasan M, El-sakka M R (2018) Improved BM3D image denoising using SSIM-optimized Wiener filter. EURASIP J IMAGE VID, 2018(1):25.
33. Kulkarni S, Kedar M, Rege P P (2018) Comparison of Different Speckle Noise Reduction Filters for RISAT -1 SAR Imagery. In: International Conference on Communication and Signal Processing, Chennai, India, 3-5 April.
34. Dhanushree M, Priyadharsini P, Sharmila T S (2019) Acoustic image denoising using various spatial filtering techniques. INT J INF TECH 1-7.
35. Singh N K, Singh A K, Tripathy M (2014) A Comparative Study of BPNN, RBFNN and ELMAN Neural Network for Short-Term Electric Load Forecasting : A Case Study of Delhi Region. In: 9th International Conference on Industrial and Information Systems (ICIIS), Gwalior, India, 5-17 December.
36. Lauraitis A, Maskeli R, Damaševičius R (2018) Research Article ANN and Fuzzy Logic Based Model to Evaluate Huntington Disease Symptoms. J HEALTHC ENG Article ID 4581272, 10 pages.
37. Kurach K, Pawlowski K (2016) Predicting Dangerous Seismic Activity with Recurrent Neural Networks. In: Proceedings of the Federated Conference on Computer Science, Gdansk, Poland, 11-14 September, pp 239–243.
38. Bianchi F M., Maiorino E, Kampffmeyer M C, Rizzi A, Jenssen R (2017) An overview and comparative analysis of Recurrent Neural Networks for Short Term Load Forecasting. Springer.

39. Köker R (2013) A Genetic Algorithm Approach to a Neural-network-based Inverse Kinematics Solution of Robotic Manipulators Based on Error Minimization. INFORM CONTROL 222:528–543.
40. Wang J, Zhang W, Li Y, Wang J, Zhangli D (2014) Forecasting Wind Speed Using Empirical Mode Decomposition and Elman Neural Network. APPL SOFT COMPUT 23: 452–459.
41. Yu F, Xu X (2014) A Short-term Load Forecasting Model of Natural Gas Based on Optimized Genetic Algorithm and Improved BP Neural Network. APPL ENERG 134:102–113.
42. Zhou Y Q, Xidian H C (2014) Invasive Weed Optimization Algorithm for Optimization No-Idle Flow Shop Scheduling Problem. NEUROCOMPUTING 137: 285–292.
43. Azizipour M, Ghalenoei V, Afshar M H, Solis S S (2016) Optimal Operation of Hydropower Reservoir Systems Using Weed Optimization Algorithm. WATER RESOUR MANAG 30: 3995–4009.
44. Ryerkerk M L, Averill R C, Deb K, Goodman E D (2017) Solving Metameric Variable-length Optimization Problems Using Genetic Algorithms. GENET PROGRAM EVOL M 18(2):247-277.
45. Yildiz B S, Yildiz A R (2017) Moth-flame optimization algorithm to determine optimal machining parameters in manufacturing processes. MATER TEST 59(5):425–429.
46. Yildiz A R, Abderazek H, Mirjalili S (2019a) A Comparative Study of Recent Non-traditional Methods for Mechanical Design Optimization. ARCH COMPUT METHOD E 1-19.
47. Mirjalili S, Mohammad S, Lewis A (2014) Grey Wolf Optimizer. ADV ENG SOFTW 69:

46–61.

48. Mohanty B (2019) Performance analysis of moth flame optimization algorithm for AGC system. INT J MODEL SIMUL 39(2):73–87.
49. Dhane D M, Maity M, Achar A, Bar C, Chakraborty C (2015) Selection of Optimal Denoising Filter using Quality Assessment for Potentially Lethal Optical Wound images. PROCEDIA COMPUT SCI 58:438–446.
50. Vasanth K, Manjunath T G, Raj S N (2015) A Decision based Unsymmetrical Trimmed Modified Winsorized Mean Filter for the Removal of High Density Salt and Pepper Noise in Images and Videos. PROCEDIA COMPUT SCI 54:595–604.
51. Rathi V P G P, Palani D S (2012) Brain tumor mri image classification w ith feature selection and extraction using linear discriminant analysis. INT J INF SCI TECH 2(4):1-17.
52. Subasi A, Gursoy M I (2010) EEG signal classification using PCA , ICA , LDA and support vector machines. EXPERT SYST APPL 37(12): 8659–8666.
53. Feng C, Ju S, Huang H (2016) Using a Simple Soil Spring Model and Support Vector Machine to Determine Bridge Scour Depth and Bridge Safety. J PERFORM CONSTR FAC 30(4):1–14.
54. Chen H, Wei L, Ning R, Cai Z, Shao H (2015) Application of Factor Analysis and SVM Technique in Expressway Condition Pattern Recognition". In: 15th COTA International Conference of Transportation Professionals, Beijing, China, 24-27 July, pp 2073–2085.
55. Sharmila A, Geethanjali P (2016) DWT Based Detection of Epileptic Seizure From EEG Signals Using Naive Bayes and k-NN Classifiers. IEEE Access 4:7716–7727.

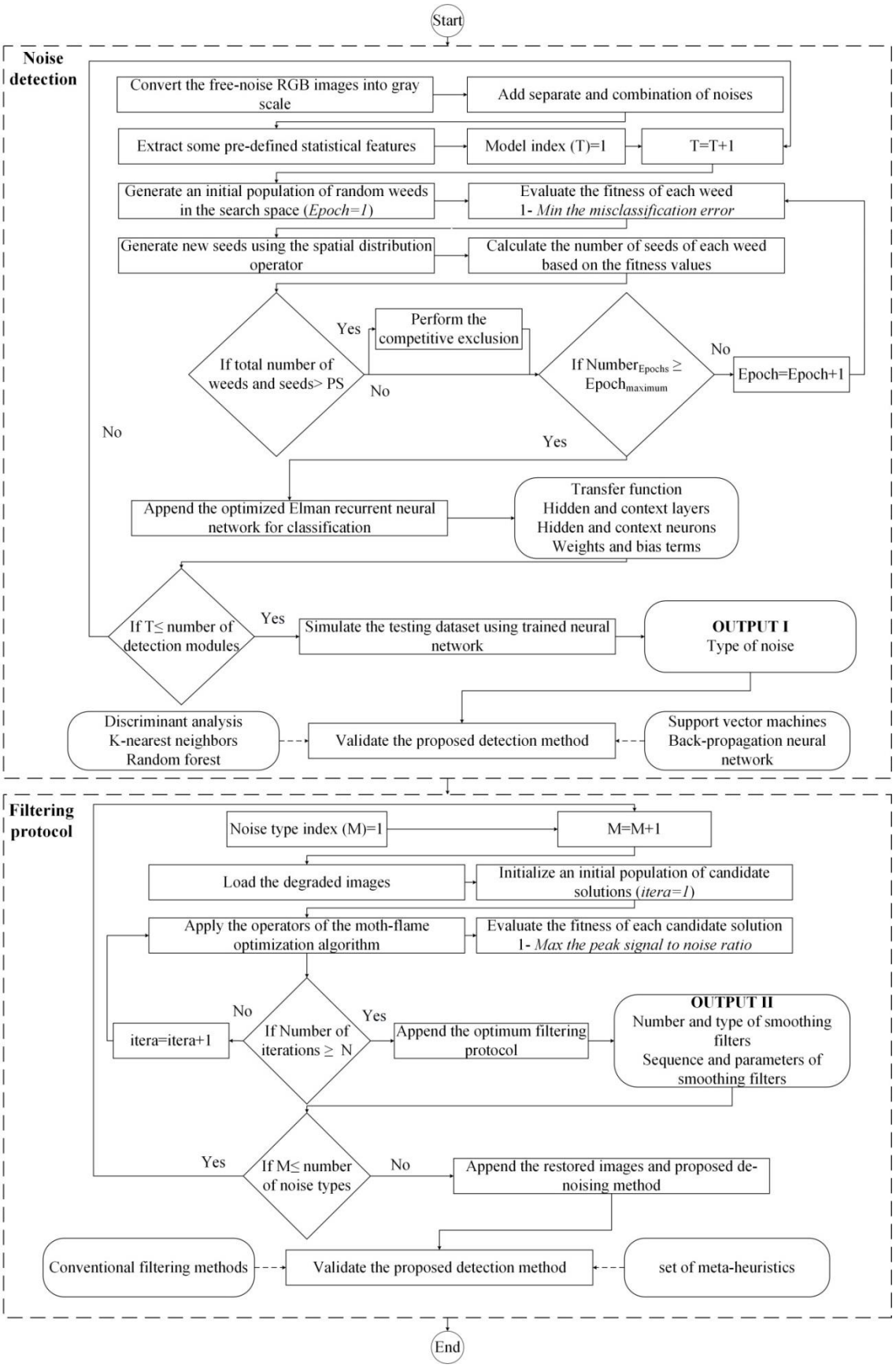
56. Yang C C, Soh C S, Yap V V (2018) A systematic approach in appliance disaggregation using k-nearest neighbours and naive Bayes classifiers for energy efficiency. ENERG EFFI 11:239–259.
57. Breiman L (2001) Random Forests. MACH LEARN 45:5–32.
58. Jin Y, Liu X, Chen Y, Liang X, Chen Y (2018) Land-cover mapping using Random Forest classification and incorporating NDVI time-series and texture : a case study of central Shandong. INT J REMOTE SENS 39(23):1–21.
59. Ahmad M W, Mourshed M, Rezgui Y (2017) Trees vs Neurons : Comparison between random forest and ANN for high-resolution prediction of building energy consumption. ENERG BUILDINGS, 147:77–89.
60. Demirci E, Yildiz A R (2016) A new hybrid approach for reliability-based design optimization of structural components. MATER TEST 61(2):111–119.
61. Yildiz A R, Kurtuluş E, Demirci E, Yildiz, B S, Karagöz S (2016) Optimization of thin-wall structures using hybrid gravitational search and Nelder-Mead algorithm. MATER TEST 58(1):75–78.
62. Elbeltagi E, Hegazy T, Grierson D (2005) Comparison Among Five Evolutionary-based Optimization Algorithms. ADV ENG INFORM 19(1): 43–53.
63. Heidari E, Movaghar A (2011) An Efficient Method Based On Genetic Algorithm To Solve Sensor Network Optimization Problem. GRAPH-HOC 3(1):18–33.
64. Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Proceedings of the IEEE International Conference on Neural Networks, Perth, Australia, 27 November-1 December, pp 1942-1948.
65. Zhang H, Li H (2010) Multi-objective Particle Swarm Optimization For Construction Time-cost Tradeoff Problems. CONSTR MANAG ECON 28(1):75–88.

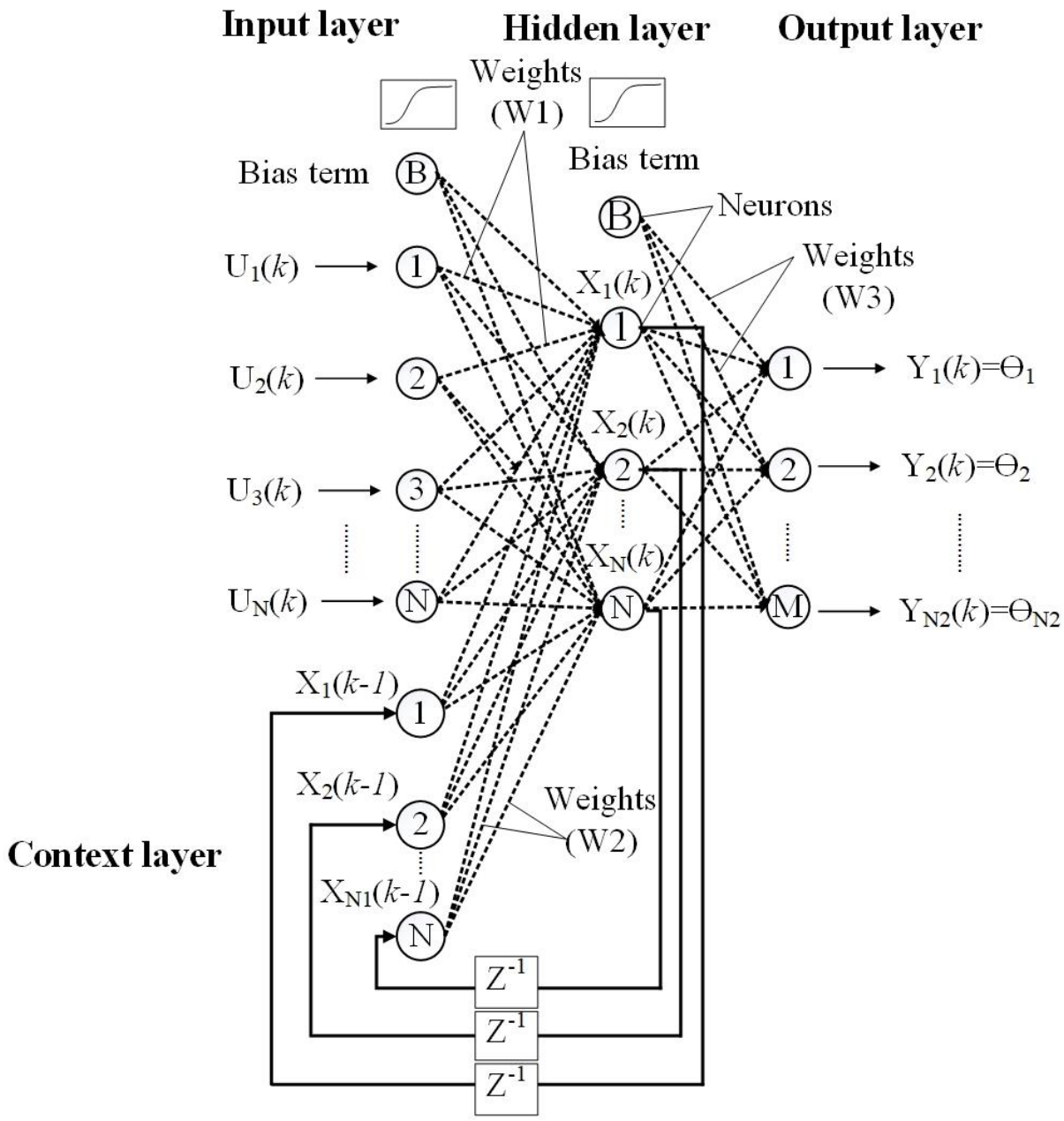
66. Baltar A M, Fontane D.G (2008) Use of Multiobjective Particle Swarm Optimization in Water Resources Management. J WATER RES PLAN MAN 134 (3):257–265.
67. Saremi S, Mirjalili S, Lewis A (2017) Advances in Engineering Software Grasshopper Optimisation Algorithm : Theory and application. ADV ENG SOFTW 105:30–47.
68. Yildiz B S, Yildiz A R (2019) The Harris hawks optimization algorithm , salp swarm algorithm, grasshopper optimization algo- rithm and dragonfly algorithm for structural design optimization of vehicle components. MATER TEST 61(8):744-748.
69. Yildiz A R, Mirjalili S, Yildiz B S, Sait S M, Li X (2019b). The Harris hawks , grasshopper and multi-verse optimization algorithms for the selection of optimal machining parameters in manufacturing operations. MATER TEST 61(8): 1-9.
70. Zhang Y, Wang J, Lu H (2019) Research and Application of a Novel Combined Model Based on Multiobjective Optimization for Multistep-Ahead Electric Load Forecasting. Energies 12(10):1–30.
71. Mirjalili S (2015) Moth-Flame Optimization Algorithm : A Novel Nature-inspired Heuristic Paradigm. KNOWL-BASED SYST 89:228–249.
72. Yildiz B S, Yildiz A R (2018) Comparison of grey wolf, whale, water cycle, ant lion and sine-cosine algorithms for the optimization of a vehicle engine connecting rod. MATER TEST 60(3):311–315.
73. Storn R, Price K (1995) Differential Evolution - A simple and efficient adaptive scheme for global optimization over continuous spaces. Technical Report TR-95-012 International Computer Science Institute, Berkley.
74. Hamza F, Abderazek H, Lakhdar S, Ferhat D, Yildiz A R (2018) Optimum design of cam-roller follower mechanism using a new evolutionary algorithm. INT J ADV MANUF TECH

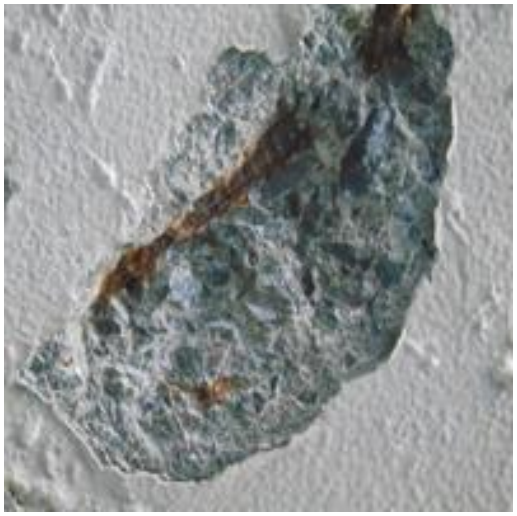
99:1267–1282.

75. Seyedpoor S M, Shahbandeh S, Yazdanpanah O (2015) An efficient method for structural damage detection using a differential evolution algorithm-based optimisation approach. CIV ENG ENVIRON SYST 32(3): 230-250.
76. Keskin G A (2015) Using integrated fuzzy DEMATEL and Fuzzy C : means Algorithm for Supplier Evaluation and Selection. INT J PROD RES 53(12): 3586-3602.

List of Figures

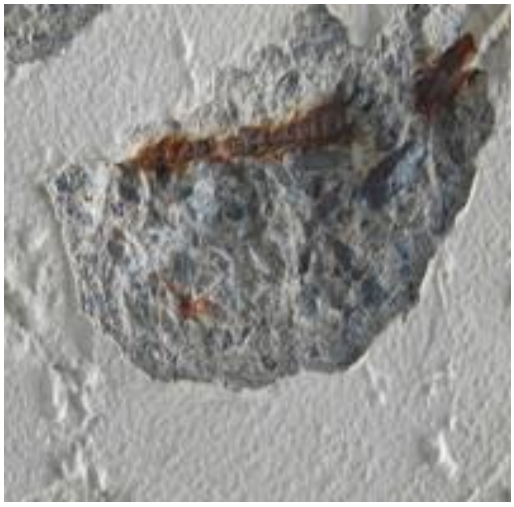

Start
Noise detection
Convert the free-noise RGB images into gray scale
Add separate and combination of noises
Extract some pre-defined statistical features
Model index (T)=1
T=T+1
Generate an initial population of random weeds in the search space (Epoch=1)
Evaluate the fitness of each weed
1- Min the misclassification error
Generate new seeds using the spatial distribution operator
Calculate the number of seeds of each weed based on the fitness values
If total number of weeds and seeds> PS
Yes
Perform the competitive exclusion
No
If Number Epochs ≥ Epoch maximum
No
Epoch=Epoch+1
Yes
No
Append the optimized Elman recurrent neural network for classification
Transfer function
Hidden and context layers
Hidden and context neurons
Weights and bias terms
If T≤ number of detection modules
Yes
Simulate the testing dataset using trained neural network
OUTPUT I
Type of noise
Discriminant analysis
K-nearest neighbors
Random forest
Validate the proposed detection method
Support vector machines
Back-propagation neural network
Filtering protocol
Noise type index (M)=1
M=M+1
Load the degraded images
Initialize an initial population of candidate solutions (itera=1)
Apply the operators of the moth-flame optimization algorithm
Evaluate the fitness of each candidate solution
1- Max the peak signal to noise ratio
No
itera=itera+1
If Number of iterations ≥ N
Yes
Append the optimum filtering protocol
OUTPUT II
Number and type of smoothing filters
Sequence and parameters of smoothing filters
Yes
If M≤ number of noise types
No
Append the restored images and proposed de-noising method
Conventional filtering methods
Validate the proposed detection method
set of meta-heuristics
End


Input layer
Hidden layer
Output layer
Weights
(W1)
Bias term
Bias term
Neurons
Weights
(W3)
$U_1(k)$
$U_2(k)$
$U_3(k)$
$U_N(k)$
$X_1(k)$
$X_2(k)$
$X_N(k)$
$Y_1(k)=\Theta_1$
$Y_2(k)=\Theta_2$
$Y_{N2}(k)=\Theta_{N2}$
$X_1(k-1)$
$X_2(k-1)$
$X_{N1}(k-1)$
Weights
(W2)
Context layer
Z^{-1}
Z^{-1}
Z^{-1}



(a) Image 1

(b) Image 2

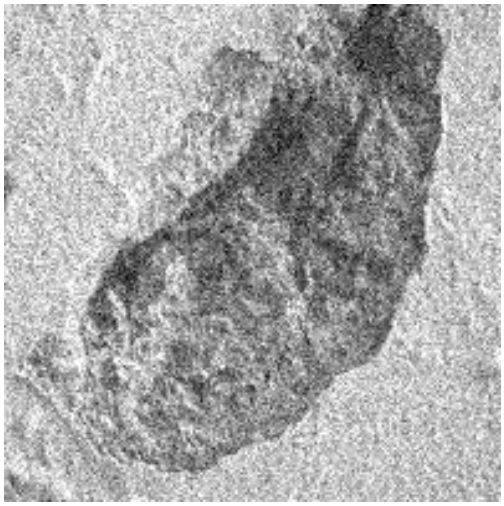


(c) Image 3

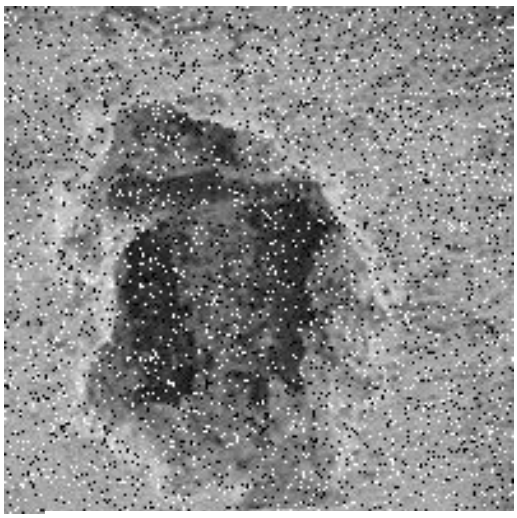
(d) Image 4

(a) Image 5

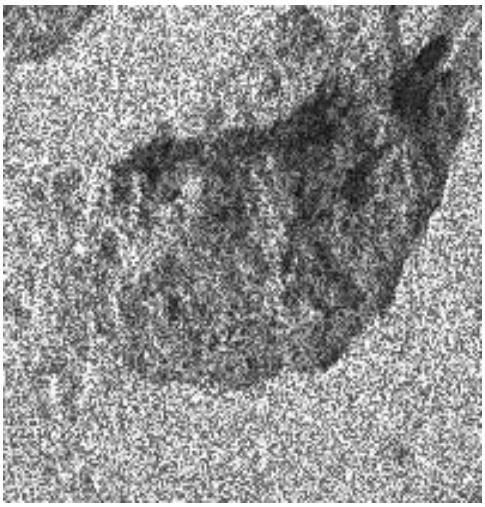
(b) Image 6



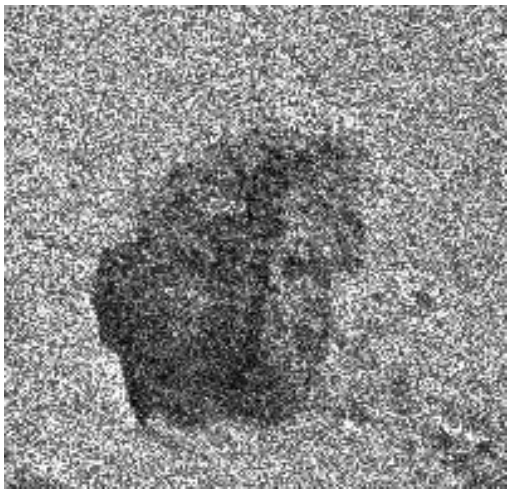
(a) Gaussian noise



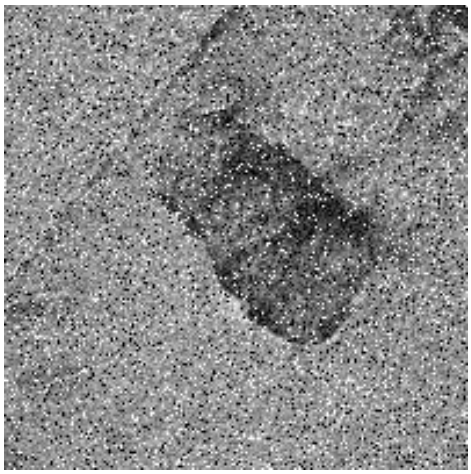
(b) Salt and pepper noise



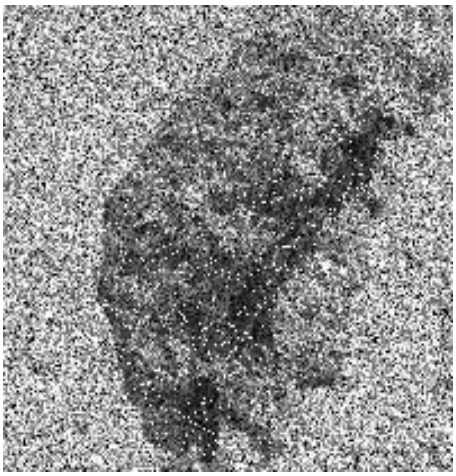
(c) Speckle noise



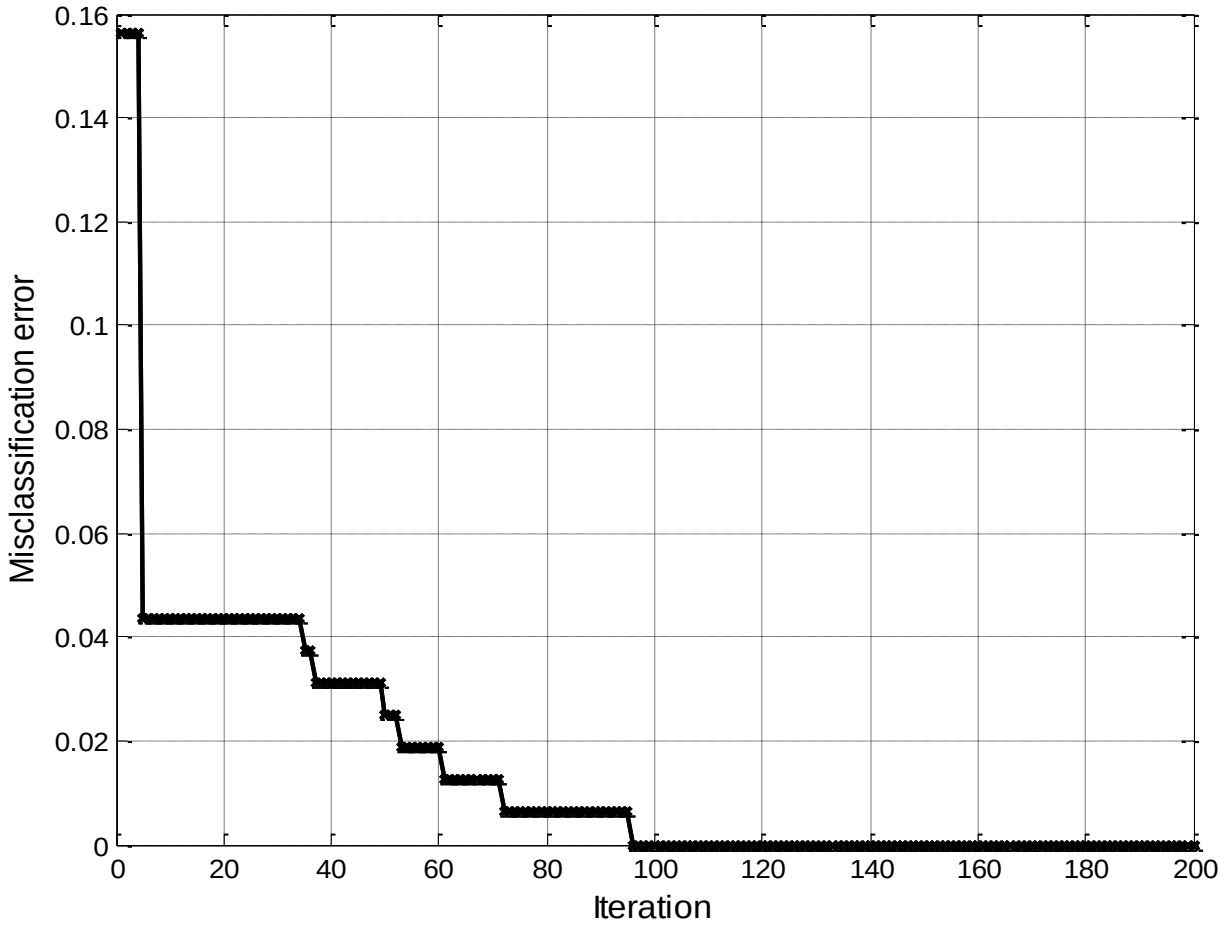
(d) Combination of Gaussian and speckle noises

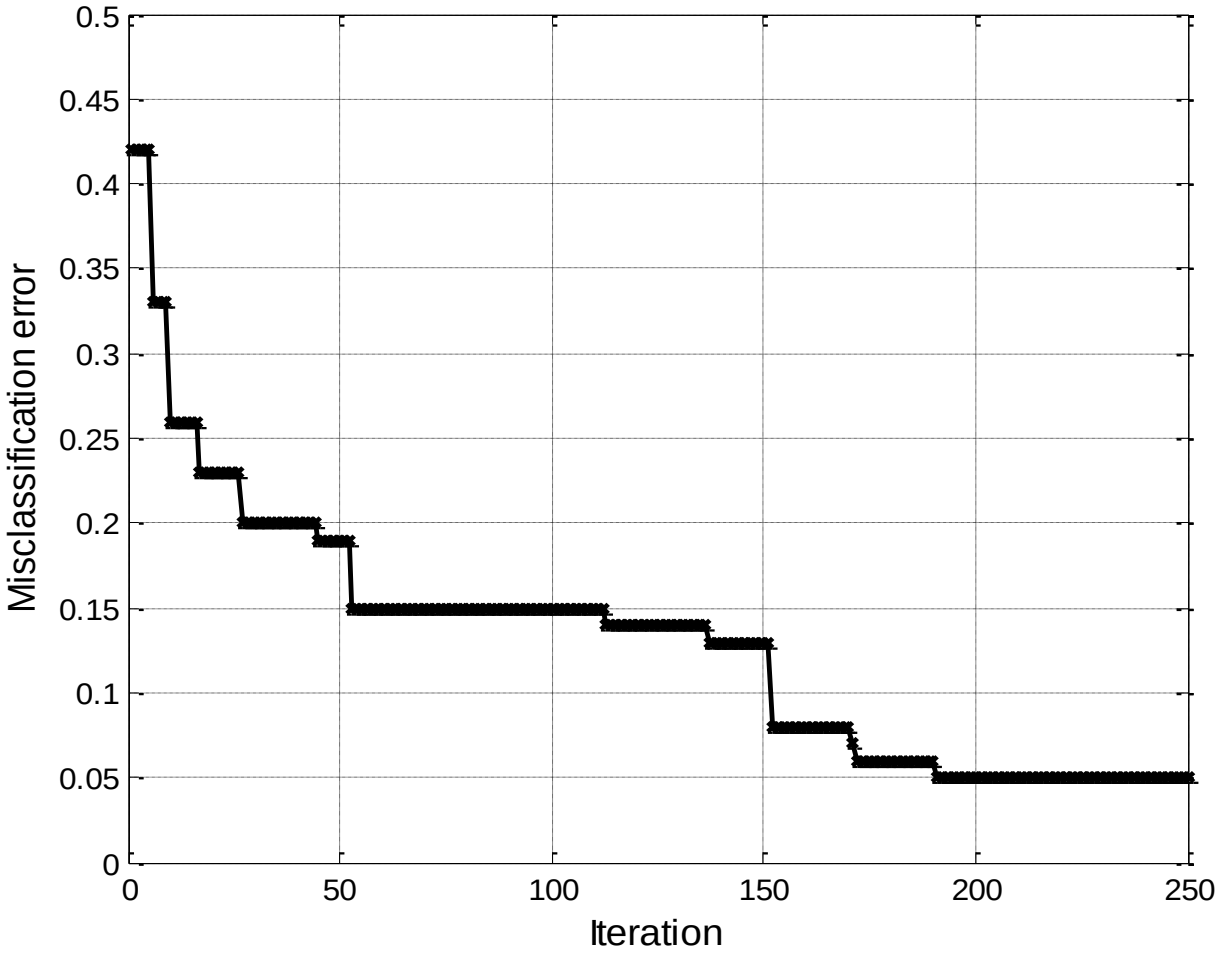


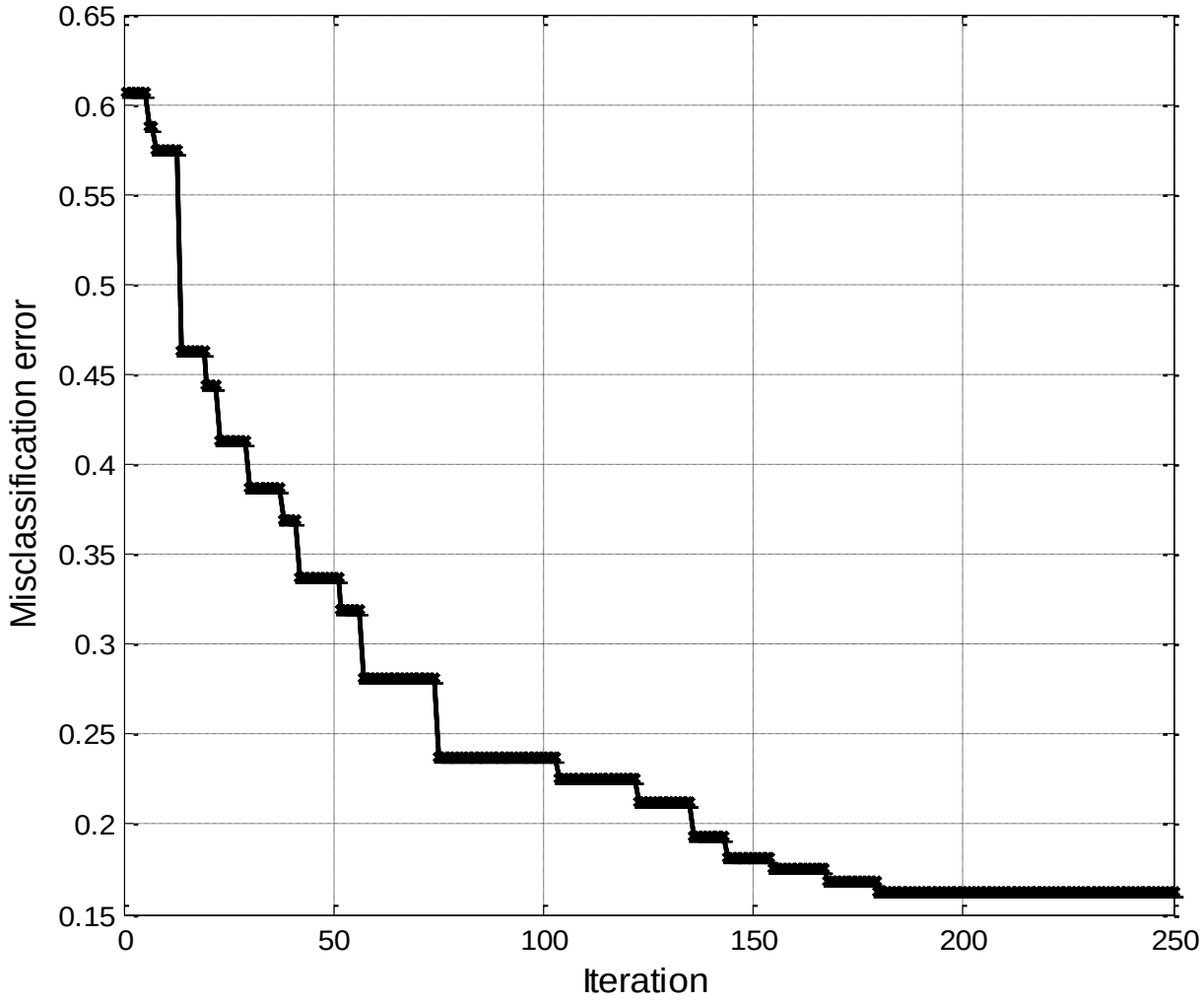
(a) Combination of Gaussian and salt and pepper noises

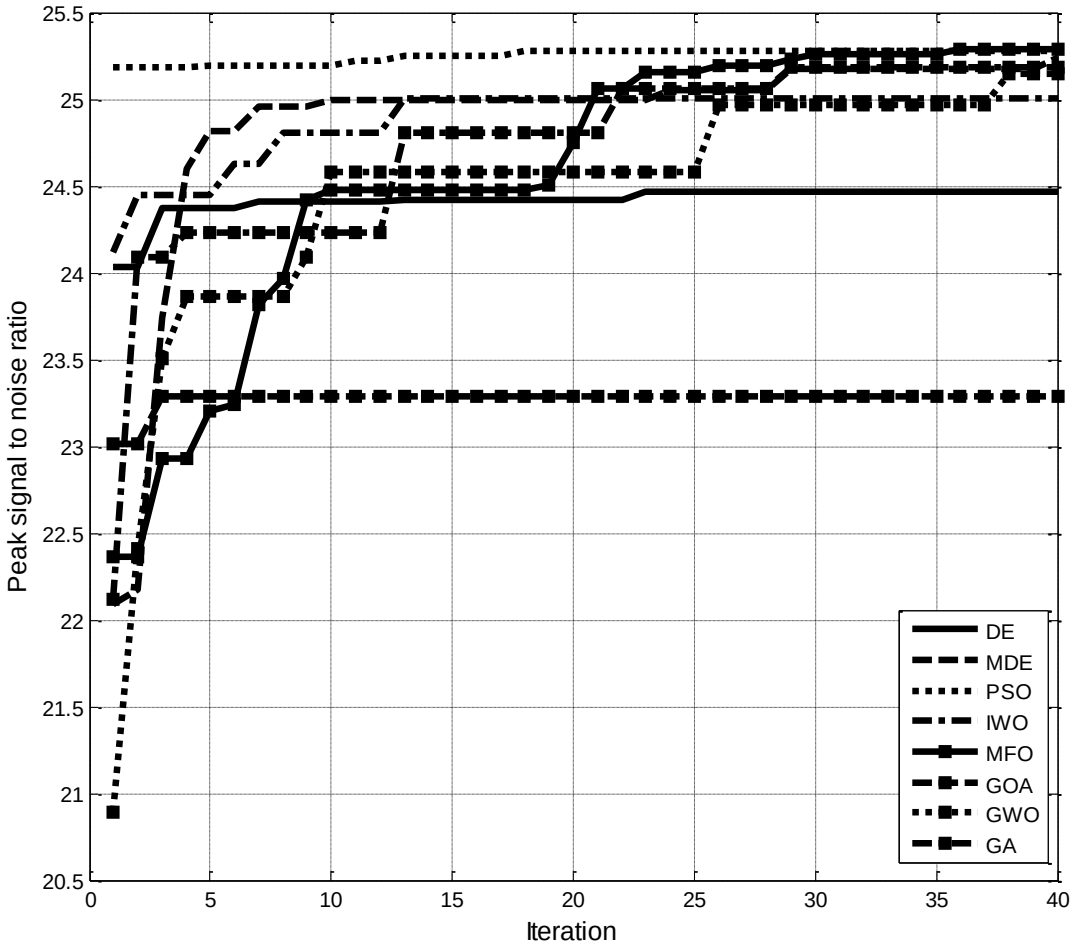


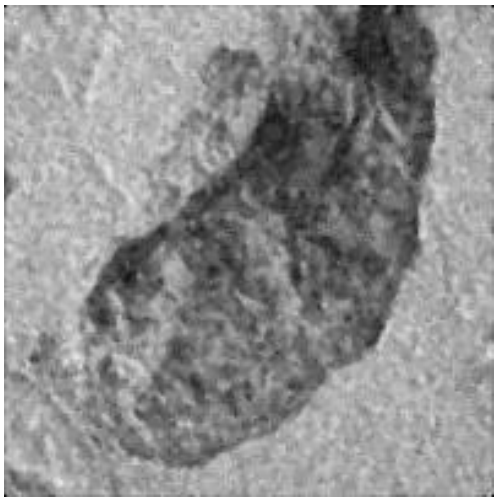
(b) Combination of speckle and salt and pepper noises


0.16
0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
Misclassification error
0
20
40
60
80
100
120
140
160
180
200
Iteration

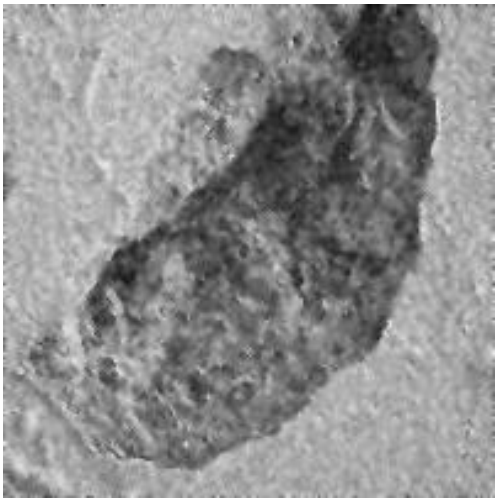

0.5
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
Misclassification error
0
50
100
150
200
250
Iteration


0.65
0.6
0.55
0.5
0.45
0.4
0.35
0.3
0.25
0.2
0.15
Misclassification error
0
50
100
150
200
250
Iteration

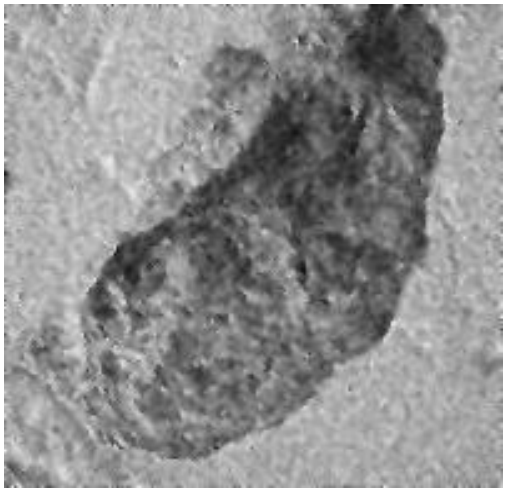

Peak signal to noise ratio
Iteration
25.5
25
24.5
24
23.5
23
22.5
22
21.5
21
20.5
0
5
10
15
20
25
30
35
40
DE
MDE
PSO
IWO
MFO
GOA
GWO
GA



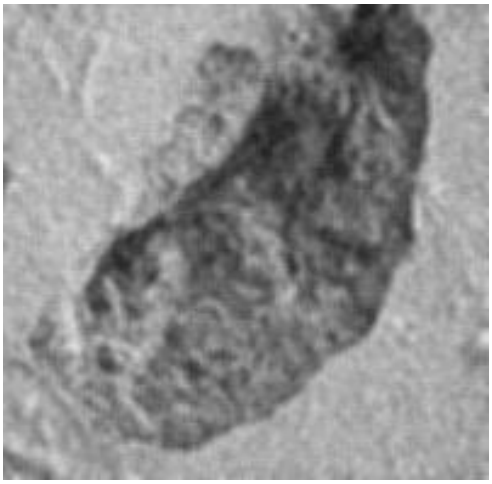
(a) Differential evolution



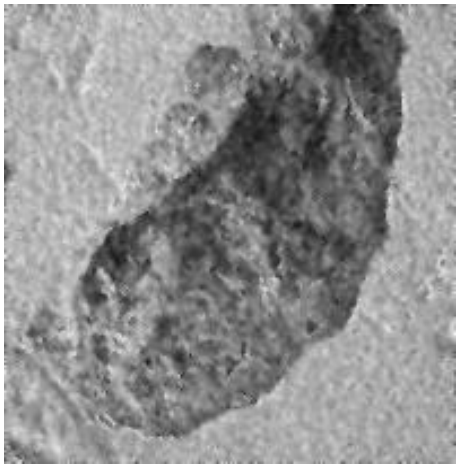
(b) Modified differential evolution



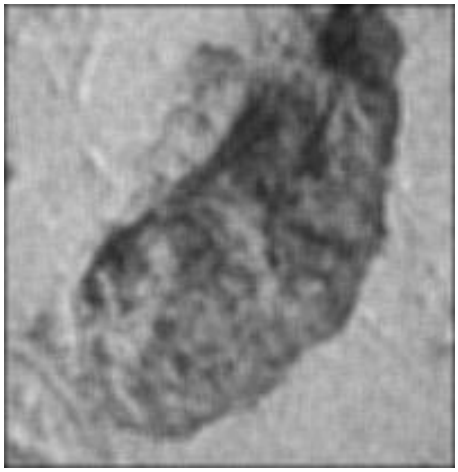
(c) Particle swarm optimization algorithm



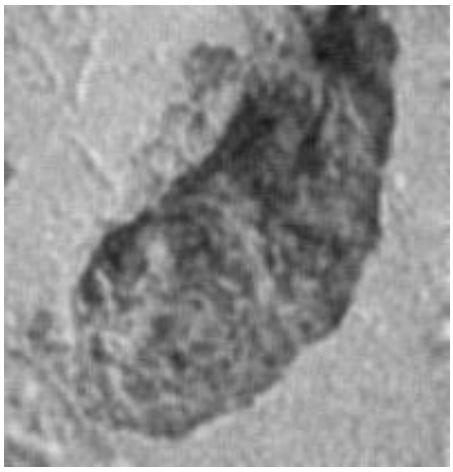
(d) Invasive weed optimization algorithm



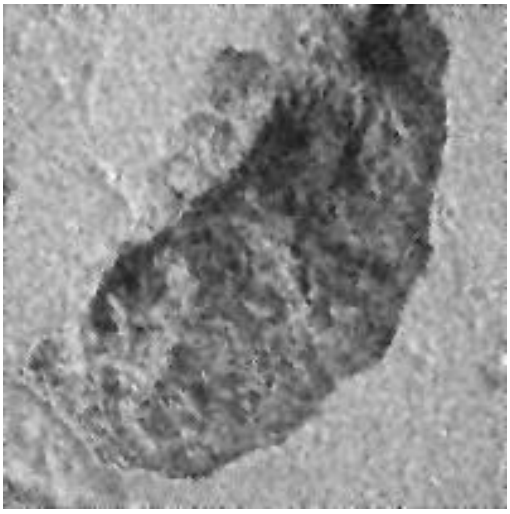
(a) Moth-flame optimization algorithm



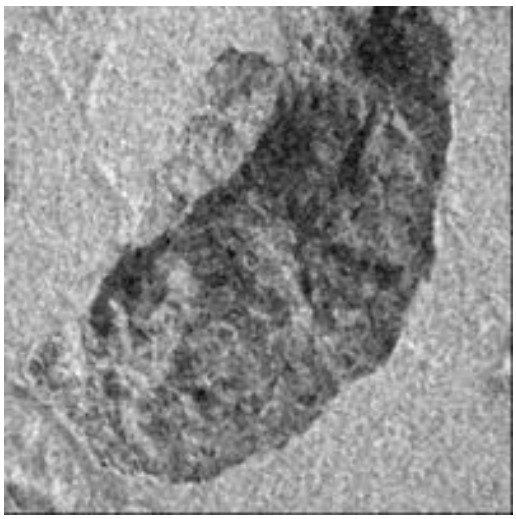
(b) Grasshopper optimization algorithm



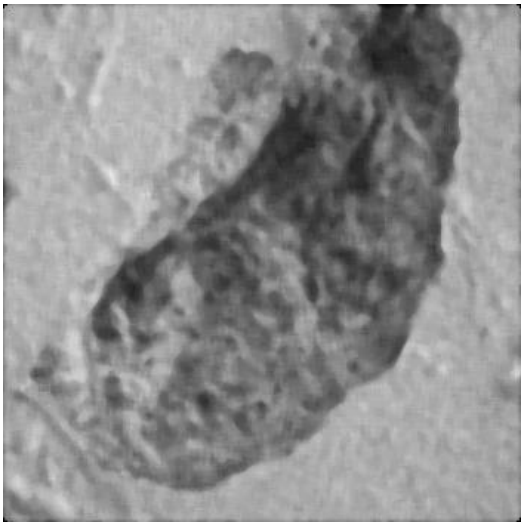
(c) Grey wolf optimization algorithm



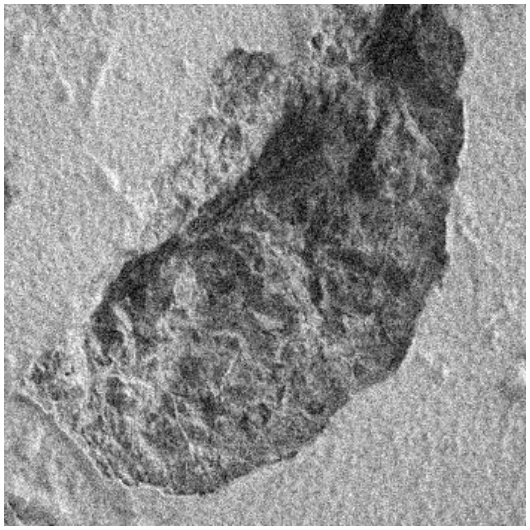
(a) Genetic algorithm



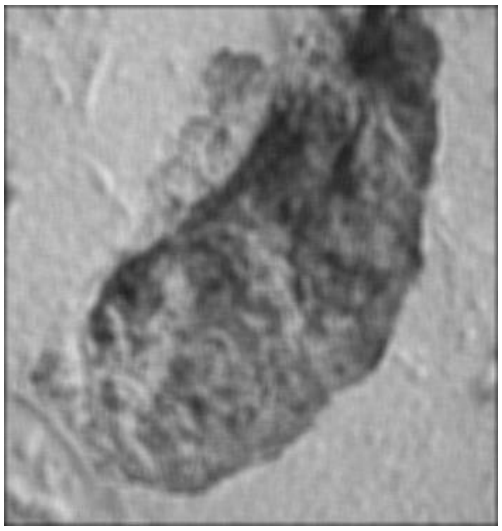
(b) Non-linear programming



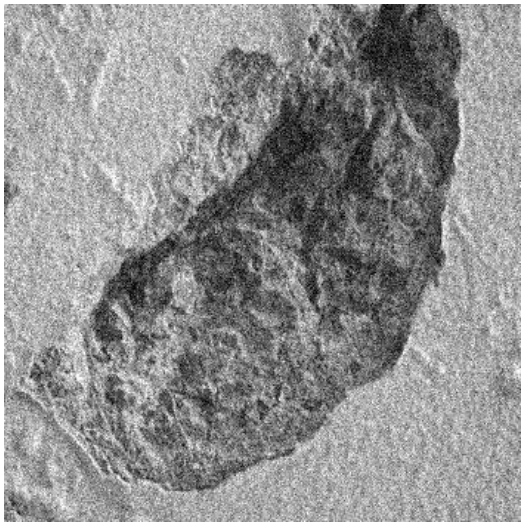
(a) Median filter



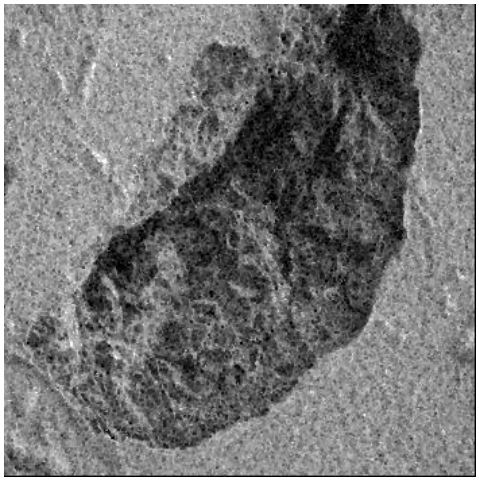
(b) Gaussian filter



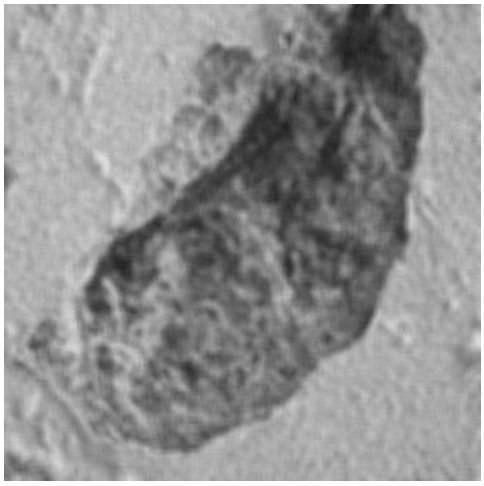
(c) Wiener filter



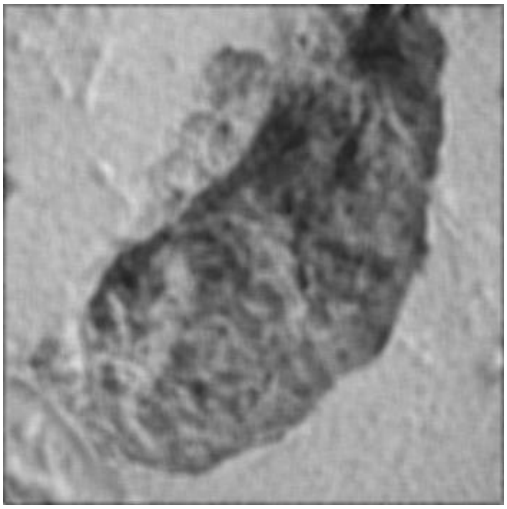
(d) Average filter



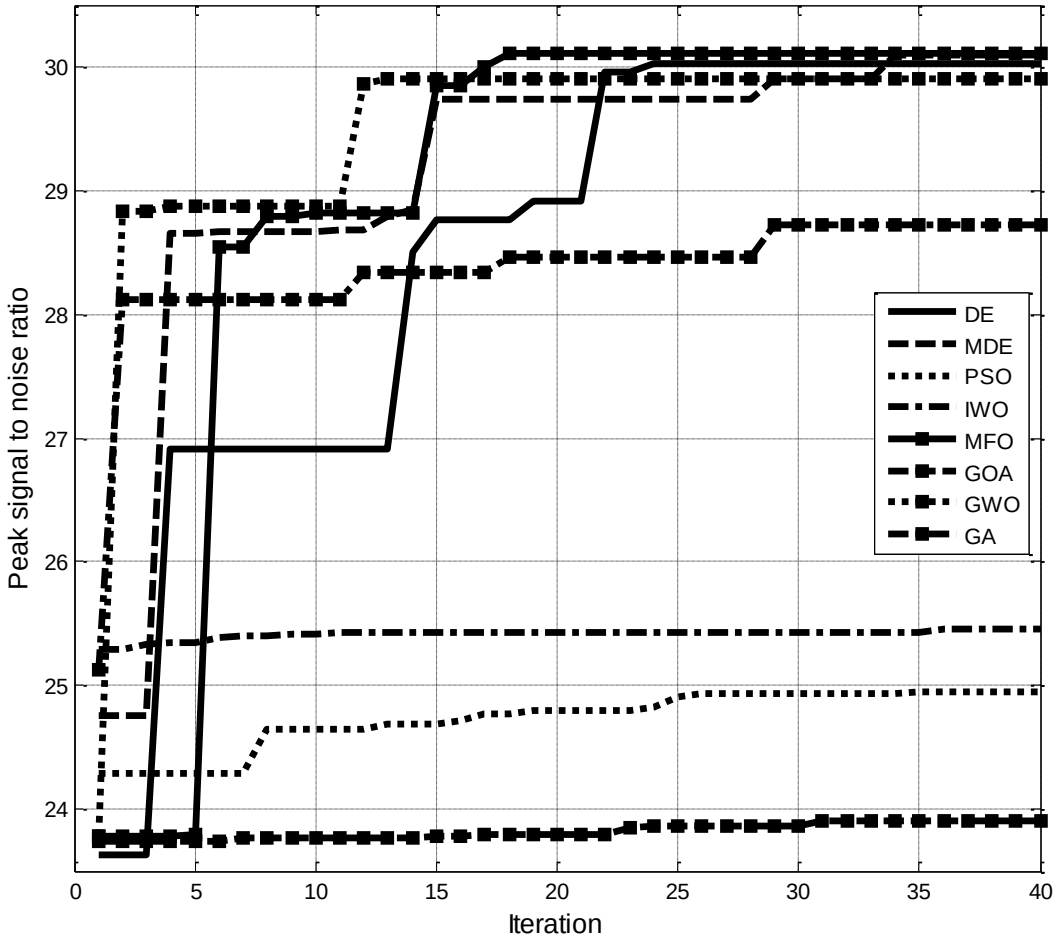
(a) Mode filter

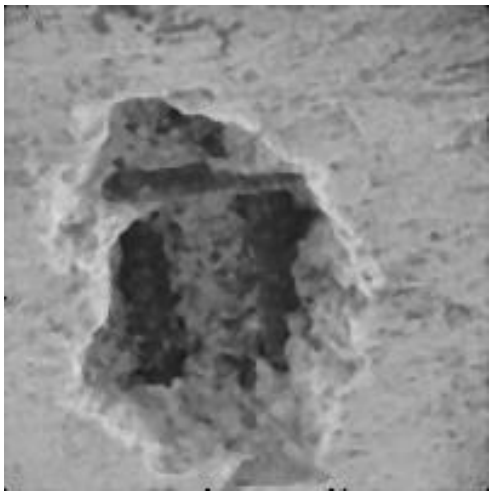


(b) Lee filter

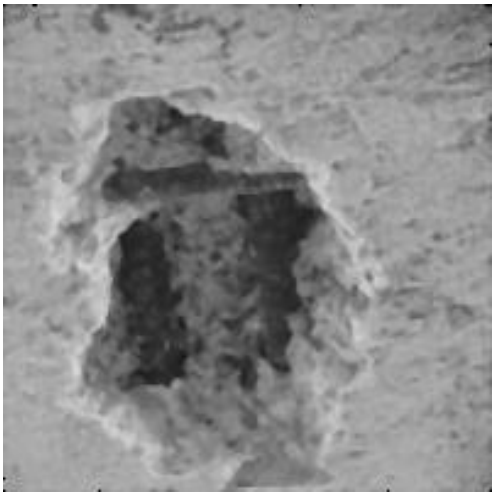


(c) Frost filter

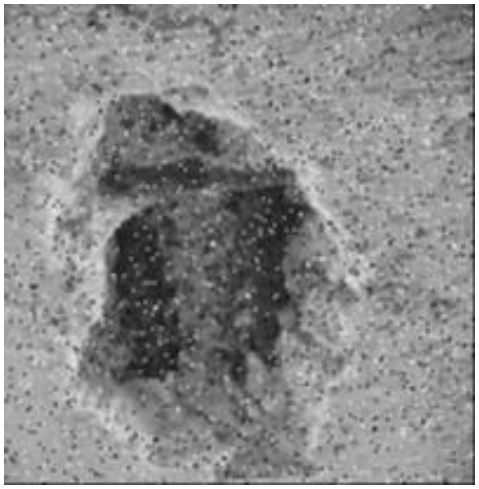

Peak signal to noise ratio
Iteration
DE
MDE
PSO
IWO
MFO
GOA
GWO
GA
24
25
26
27
28
29
30
0
5
10
15
20
25
30
35
40



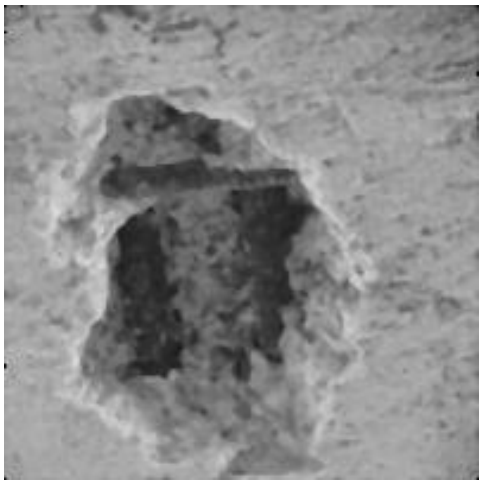
(a) Differential evolution



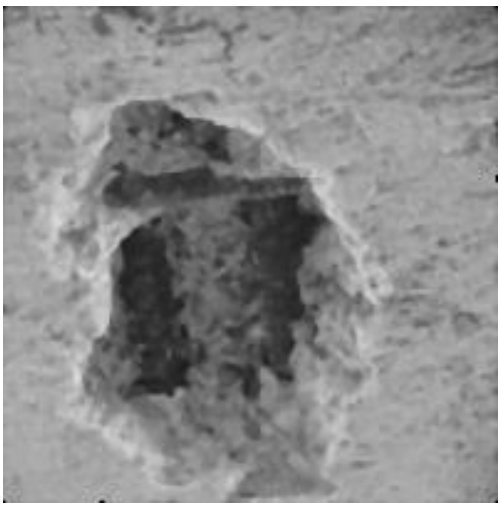
(b) Modified differential evolution



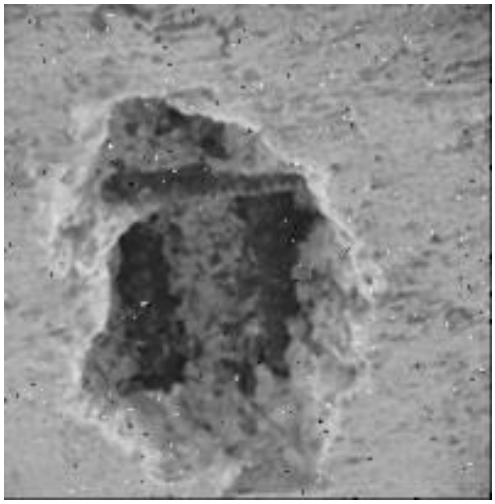
(c) Particle swarm optimization algorithm



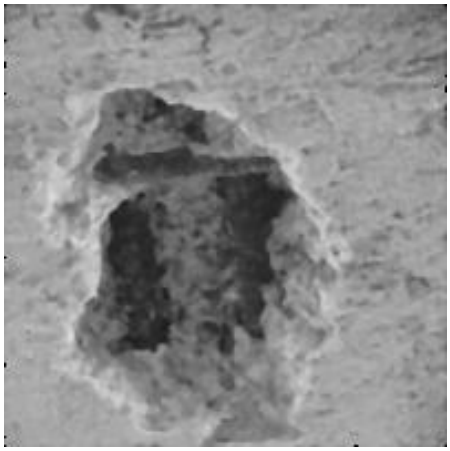
(d) Invasive weed optimization algorithm



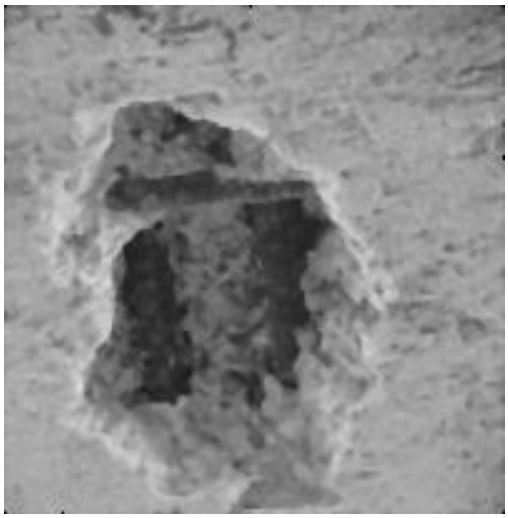
(a) Moth-flame optimization algorithm



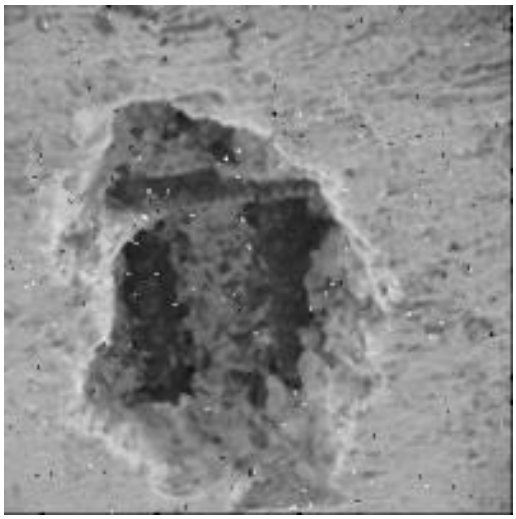
(b) Grasshopper optimization algorithm



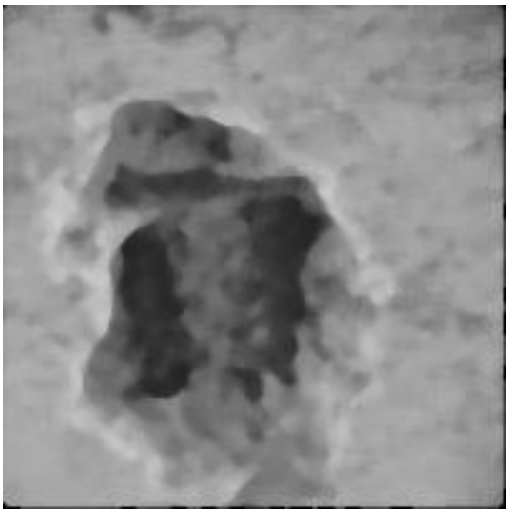
(c) Grey wolf optimization algorithm



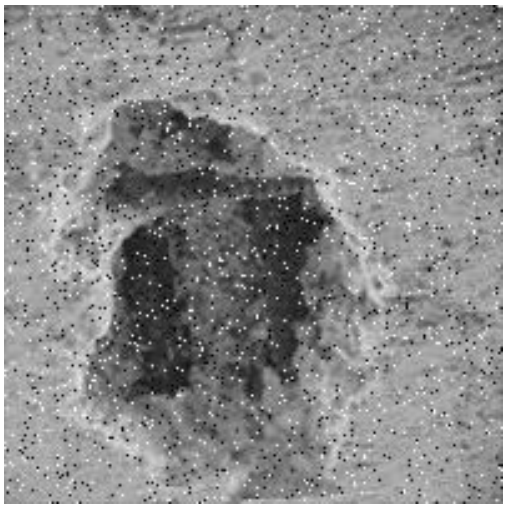
(a) Genetic algorithm



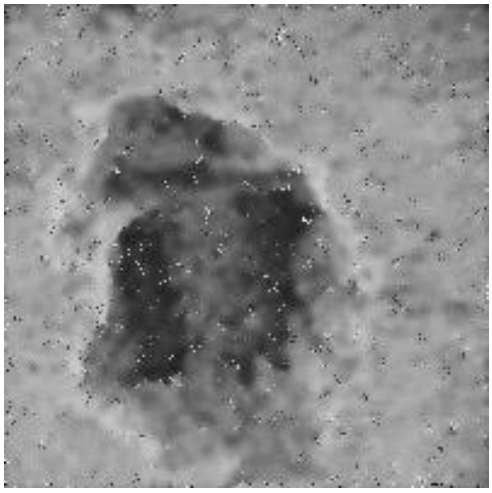
(b) Non-linear programming



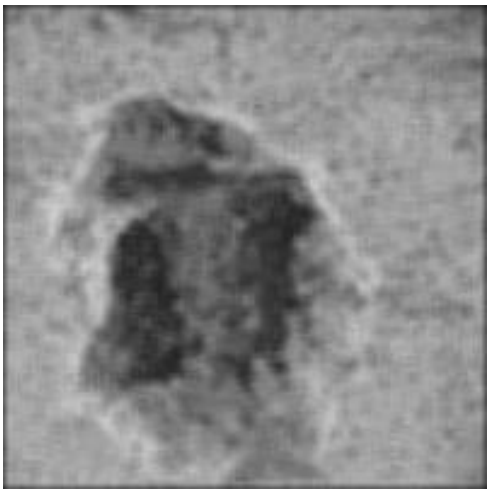
(a) Median filter



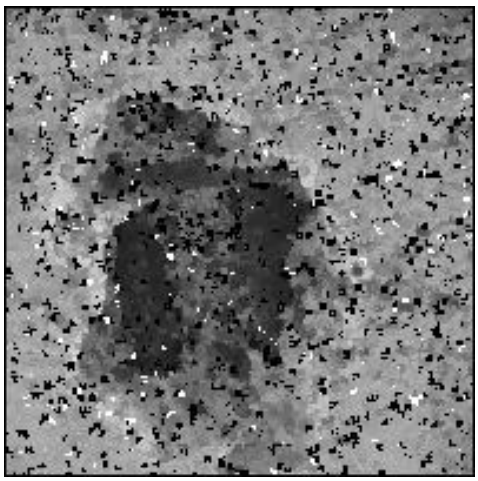
(b) Gaussian filter



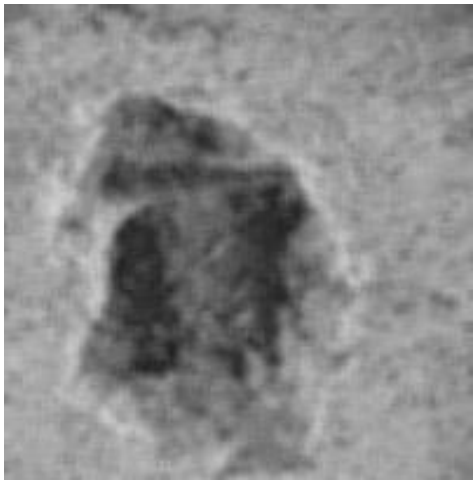
(c) Wiener filter



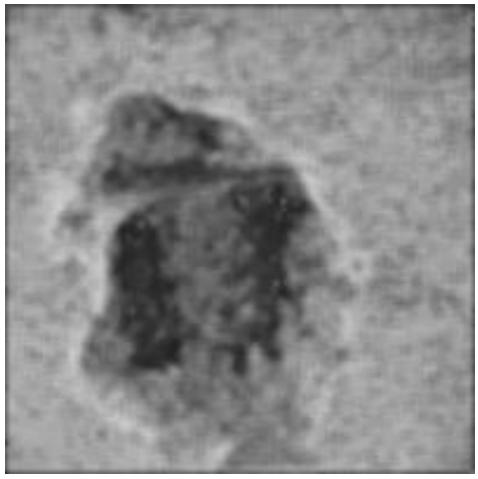
(d) Average filter



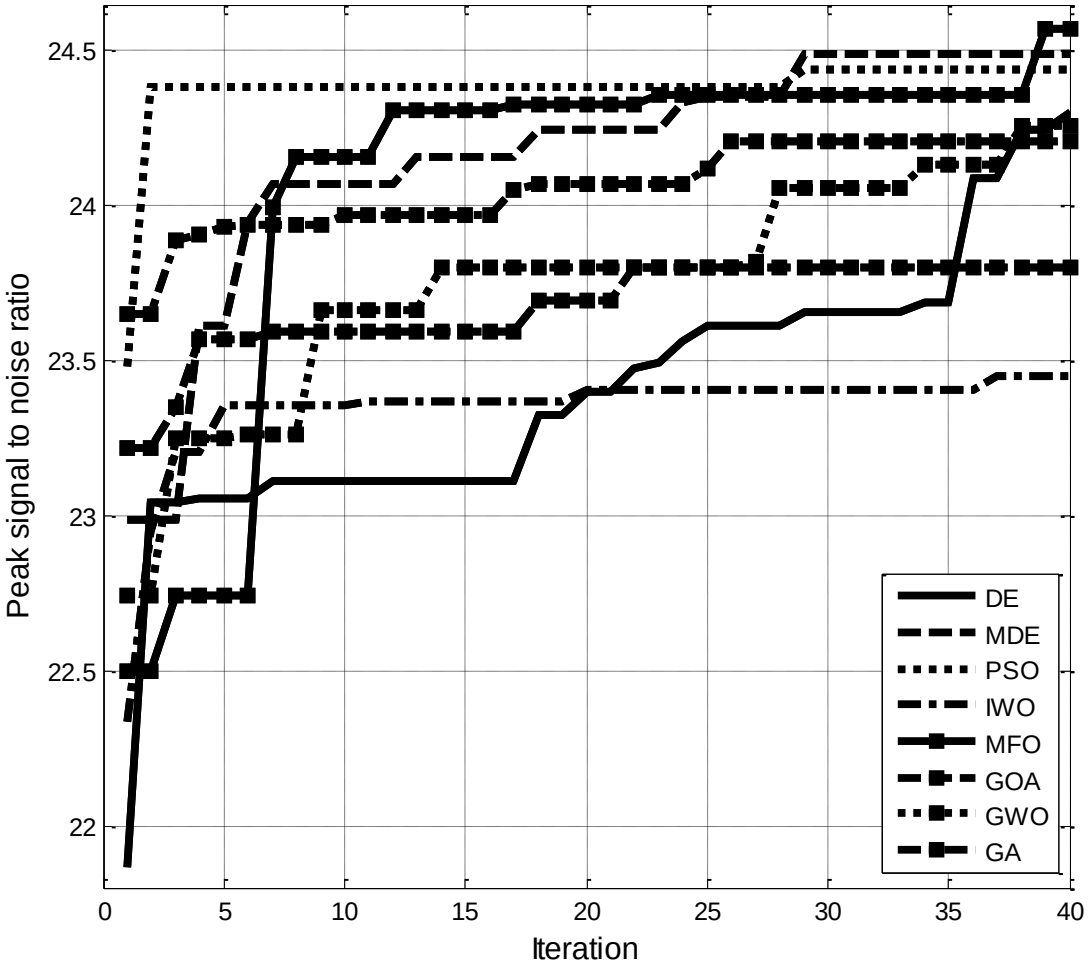
(a) Mode filer

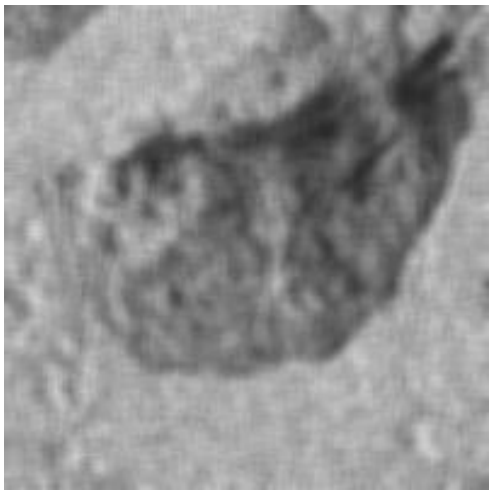


(b) Lee filter

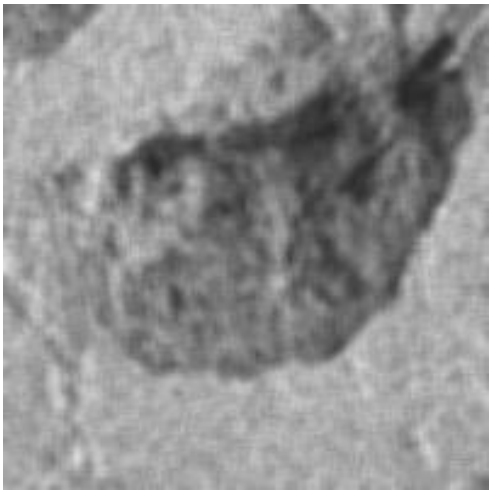


(c) Frost filter

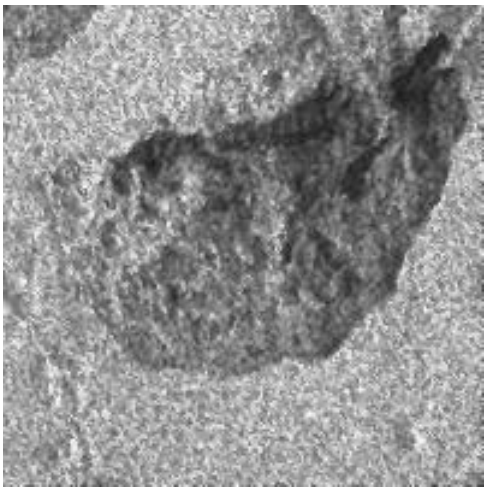

Peak signal to noise ratio
Iteration
24.5
24
23.5
23
22.5
22
0
5
10
15
20
25
30
35
40
DE
MDE
PSO
IWO
MFO
GOA
GWO
GA



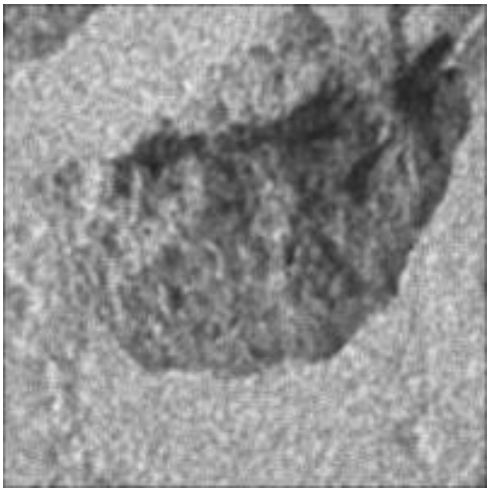
(a) Differential evolution



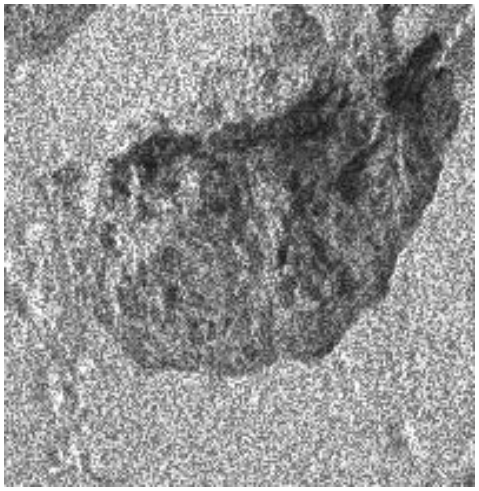
(b) Modified differential evolution algorithm



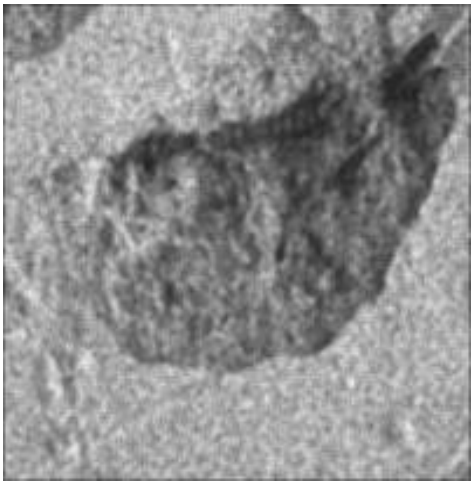
(c) Particle swarm optimization algorithm



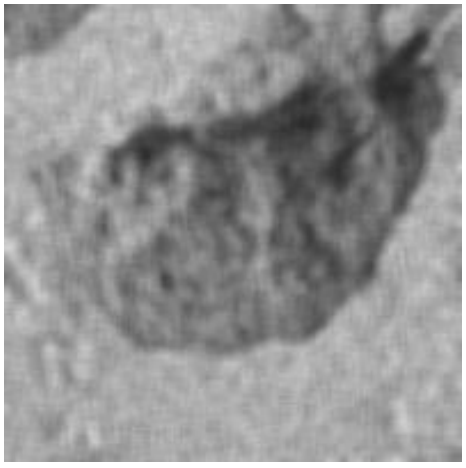
(d) Invasive weed optimization algorithm



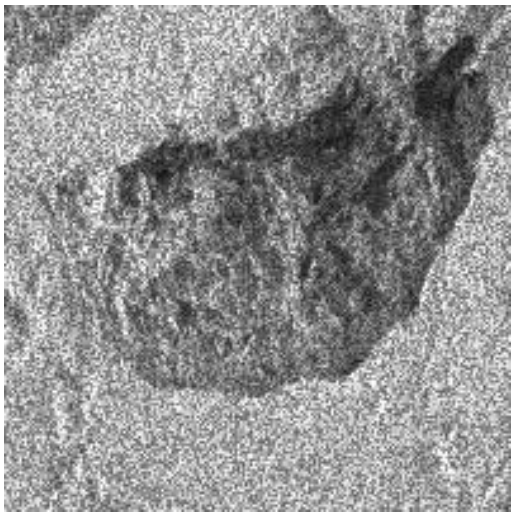
(a) Moth-flame optimization algorithm



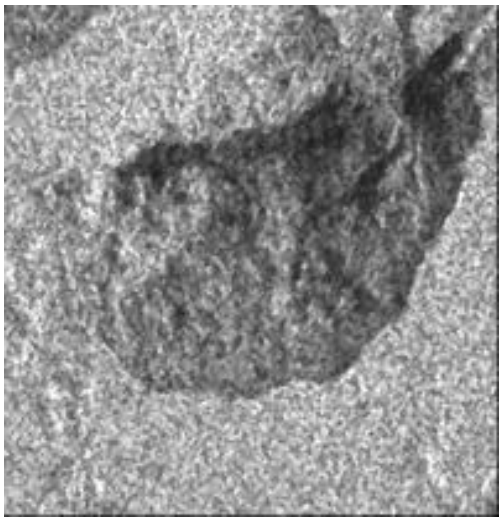
(b) Grasshopper optimization algorithm



(c) Grey wolf optimization algorithm

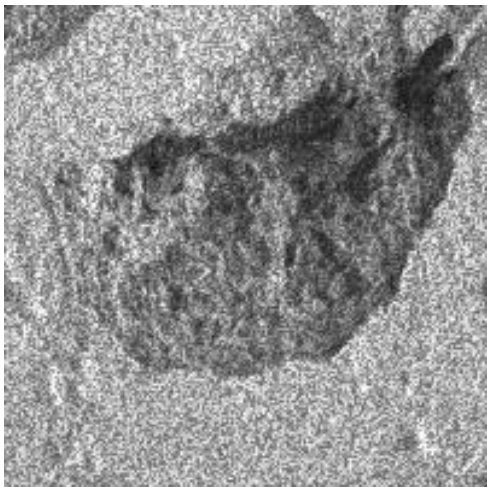


(a) Genetic algorithm

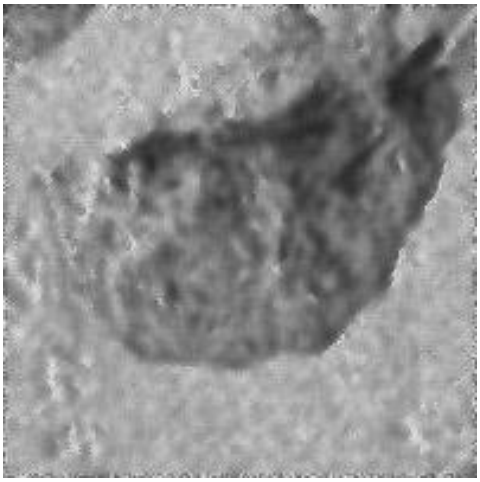


(b) Non-linear programming

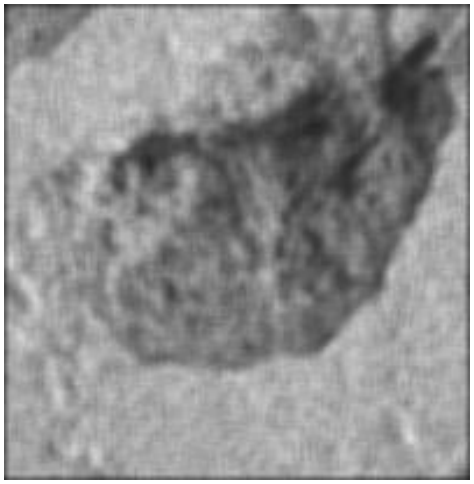
(a) Median filter



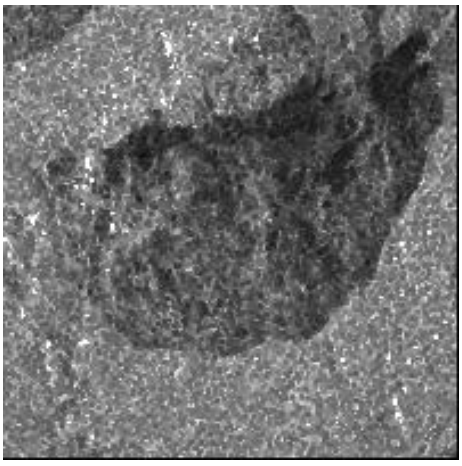
(b) Gaussian filter



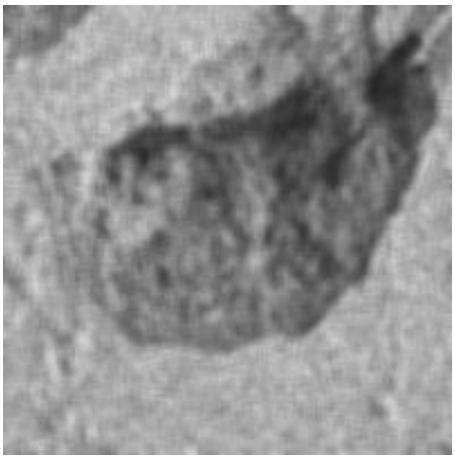
(c) Wiener filter



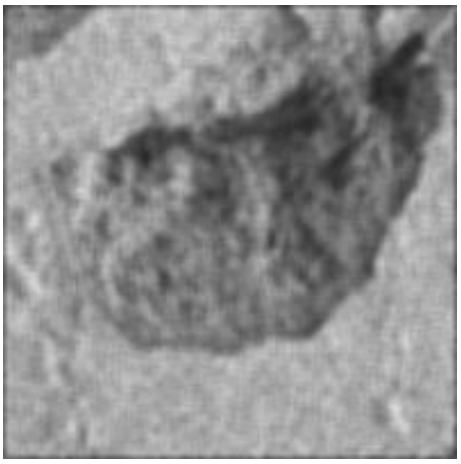
(d) Average filter



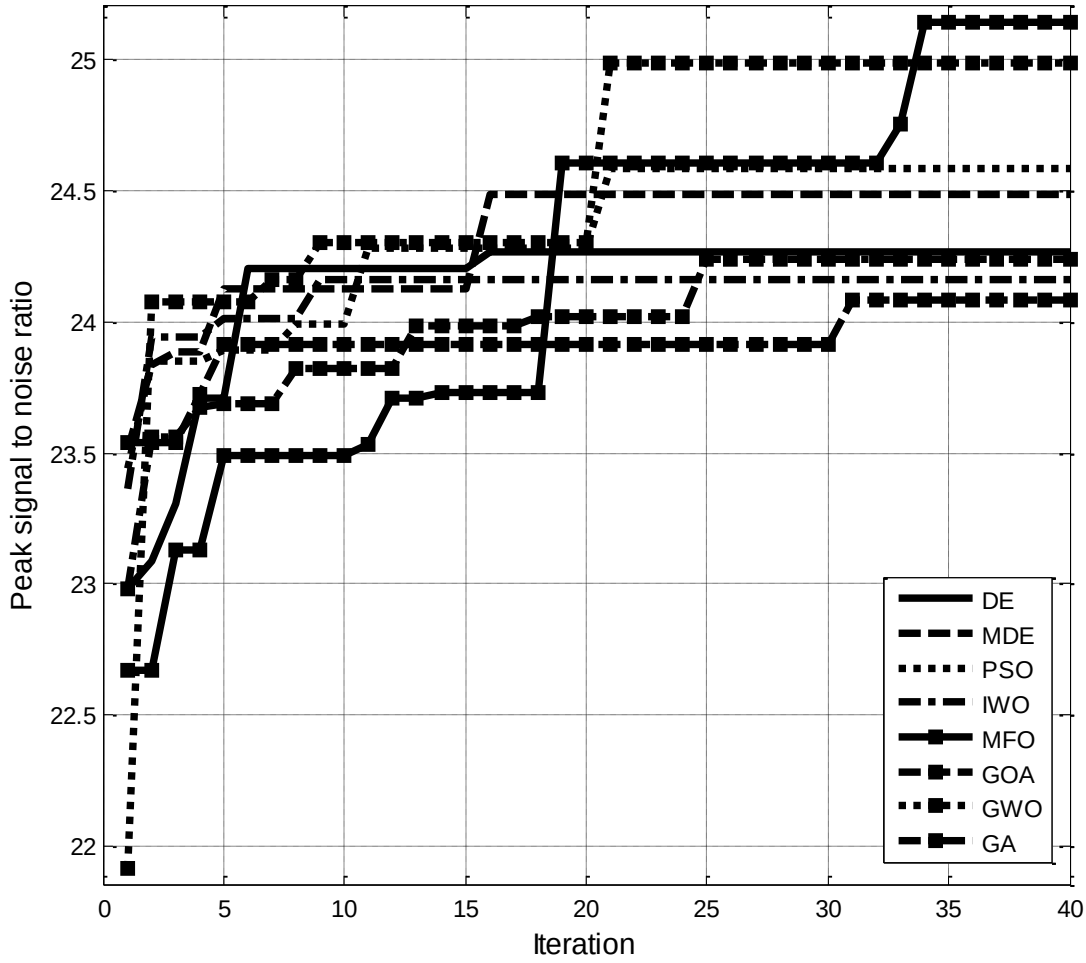
(a) Mode filter

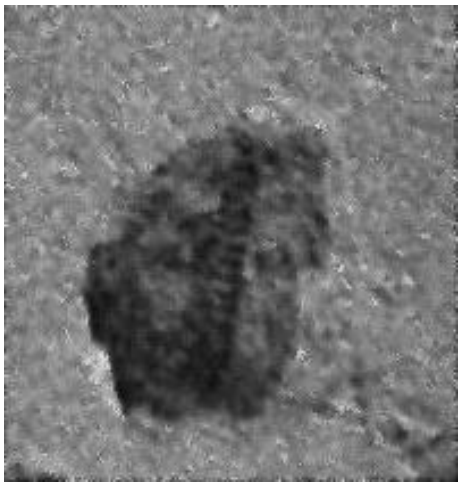


(b) Lee filter

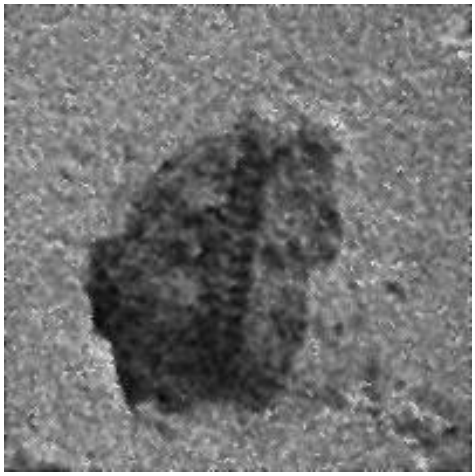


(c) Frost filter

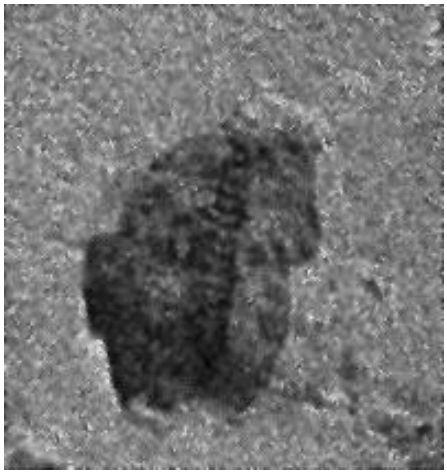

Peak signal to noise ratio
Iteration
25
24.5
24
23.5
23
22.5
22
0
5
10
15
20
25
30
35
40
DE
MDE
PSO
IWO
MFO
GOA
GWO
GA



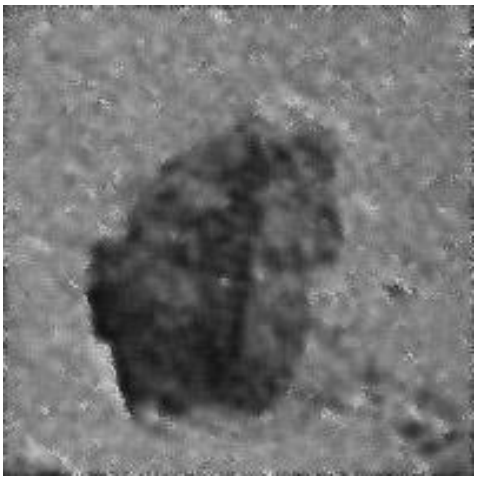
(a) Differential evolution algorithm



(b) Modified differential evolution algorithm

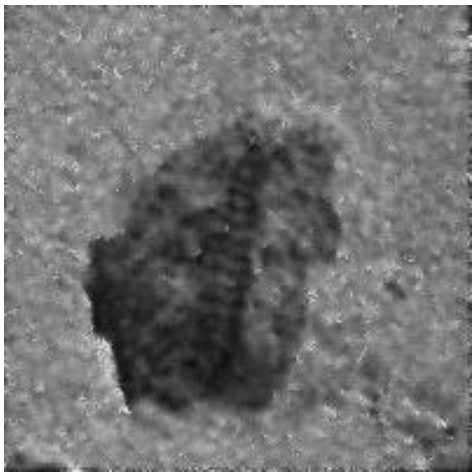


(c) Particle swarm optimization algorithm

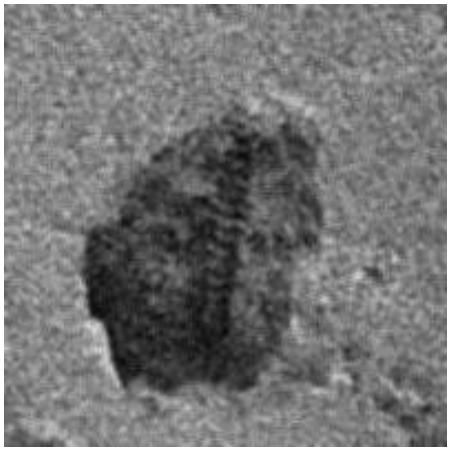


(d) Invasive weed optimization algorithm

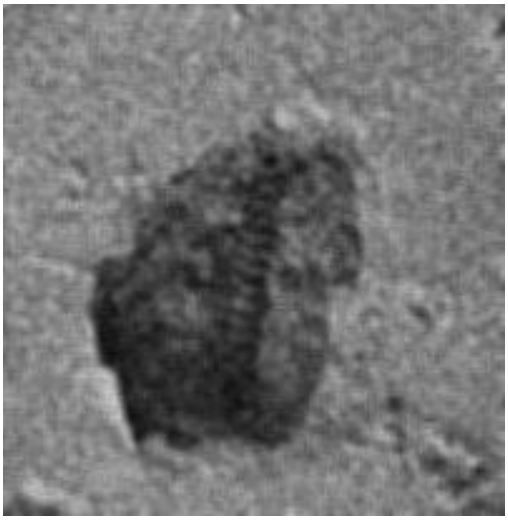
(a) Moth-flame optimization algorithm



(b) Grasshopper optimization algorithm

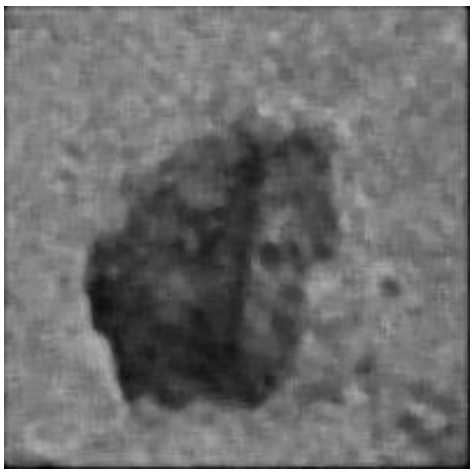


(c) Grey wolf optimization algorithm

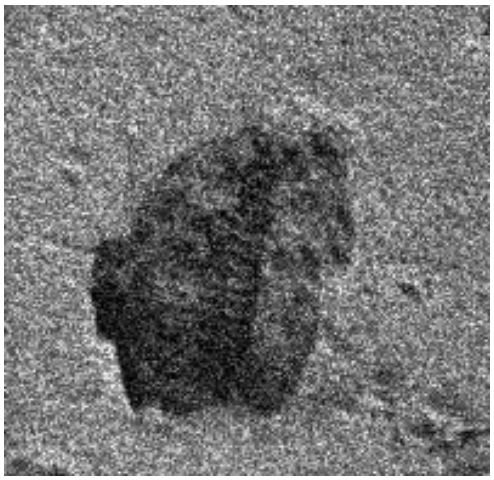


(a) Genetic algorithm

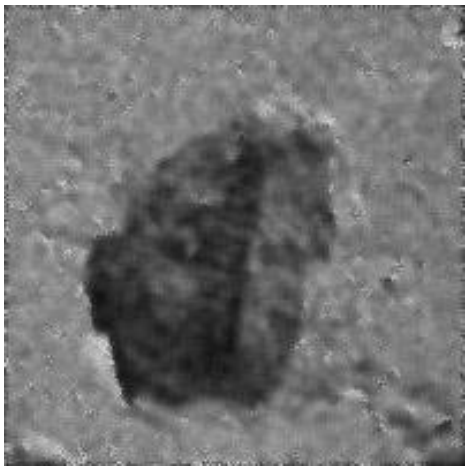
(b) Non-linear programming



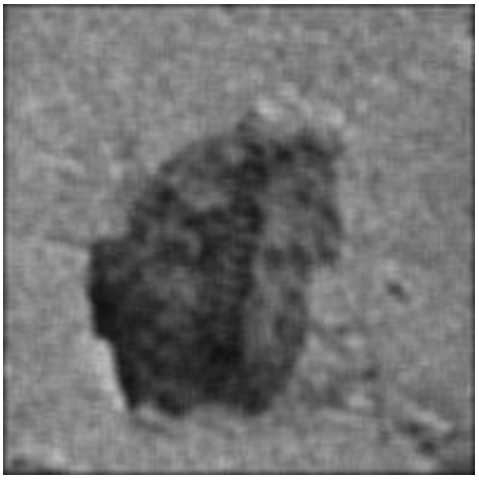
(a) Median filter



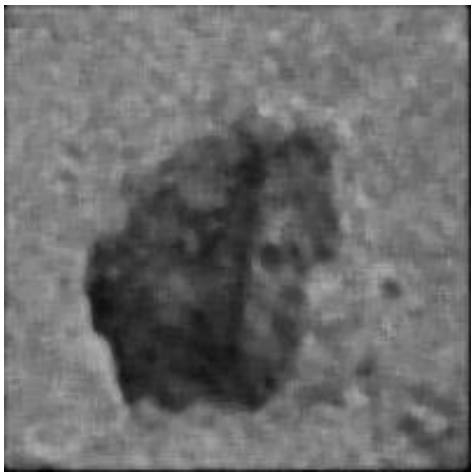
(b) Gaussian filter



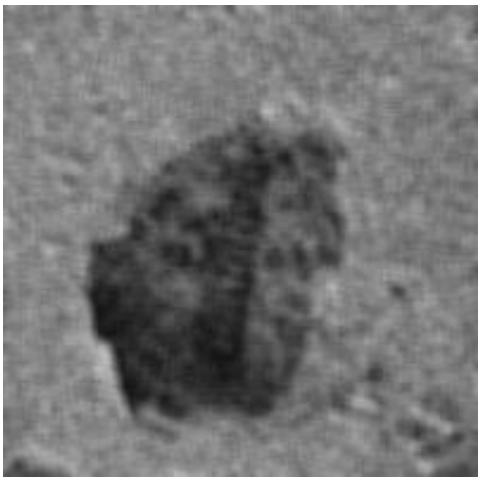
(c) Wiener filter



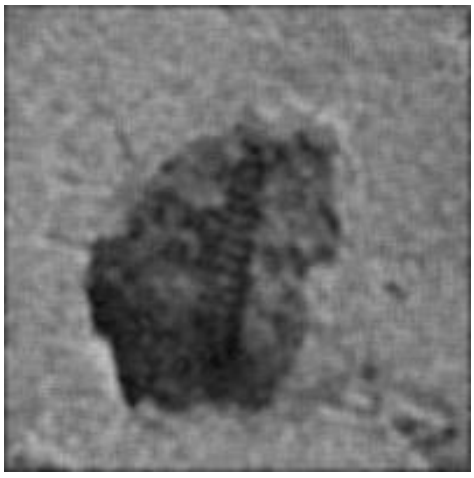
(d) Average filter



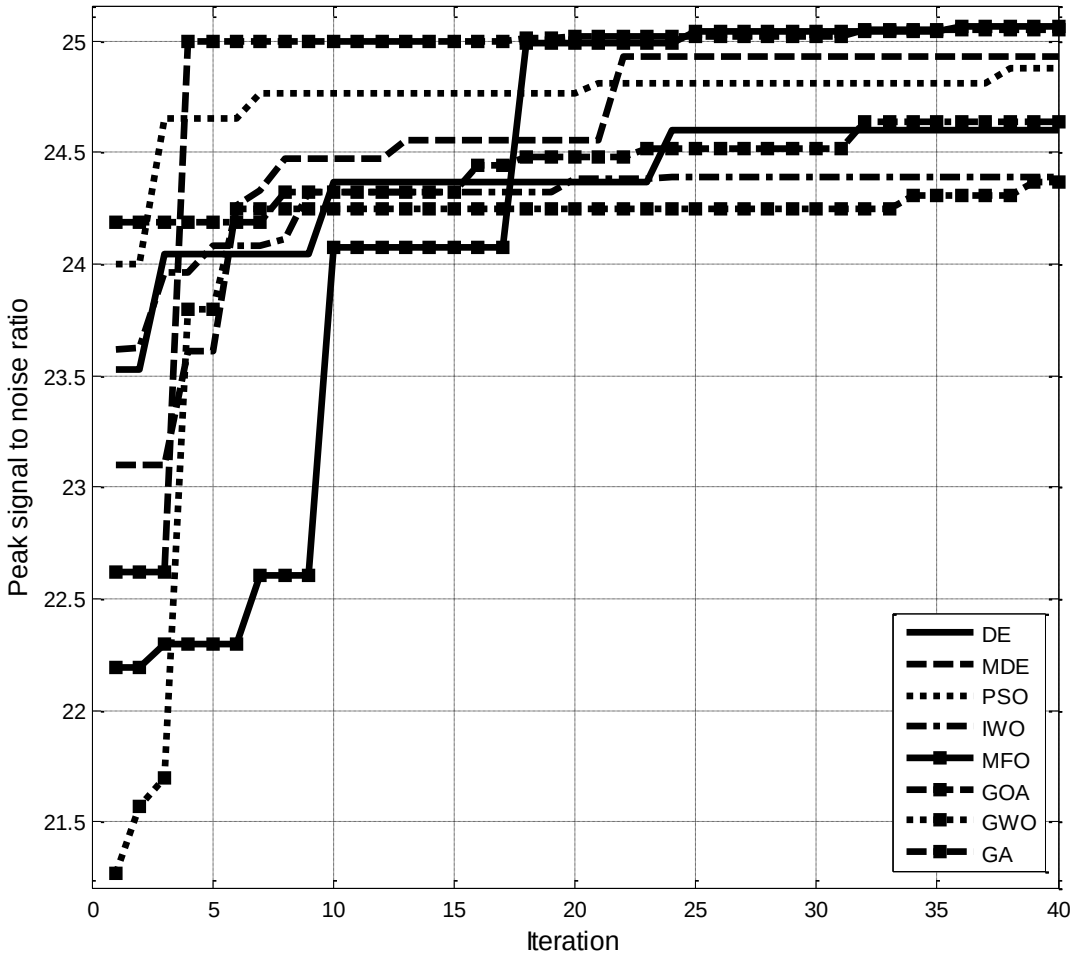
(a) Mode filter

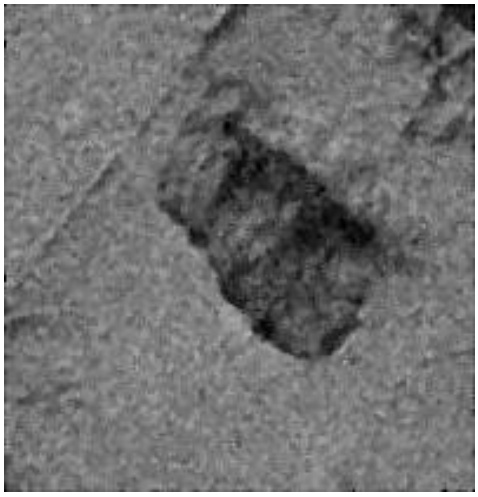


(b) Lee filter

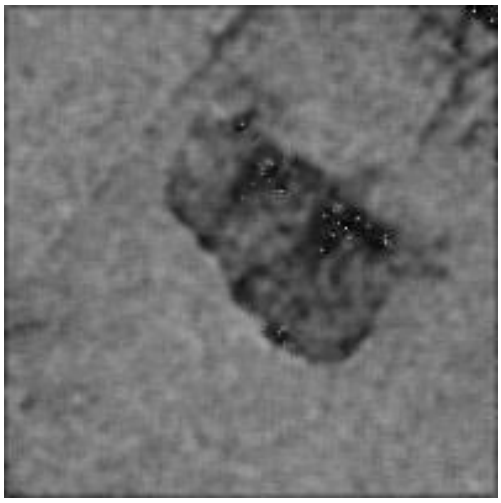


(c) Frost filter

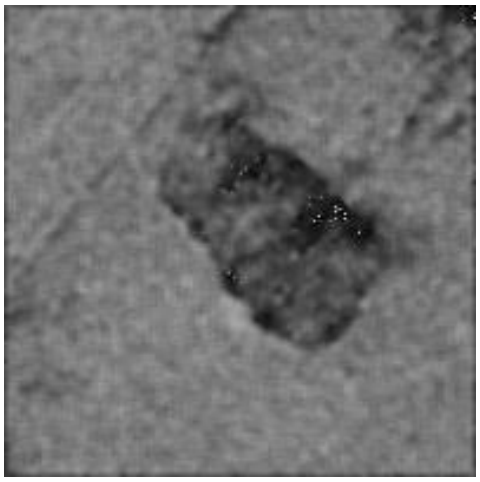

Peak signal to noise ratio
Iteration
25
24.5
24
23.5
23
22.5
22
21.5
0
5
10
15
20
25
30
35
40
DE
MDE
PSO
IWO
MFO
GOA
GWO
GA



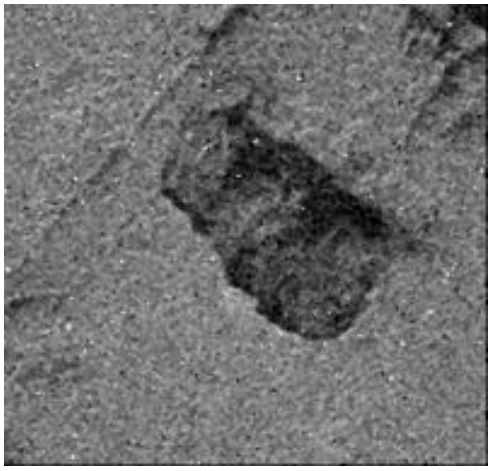
(a) Differential evolution algorithm



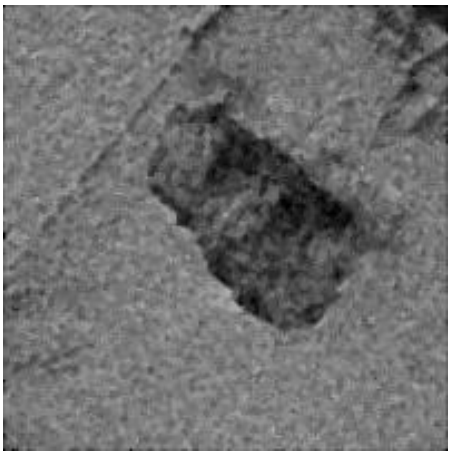
(b) Modified differential evolution algorithm



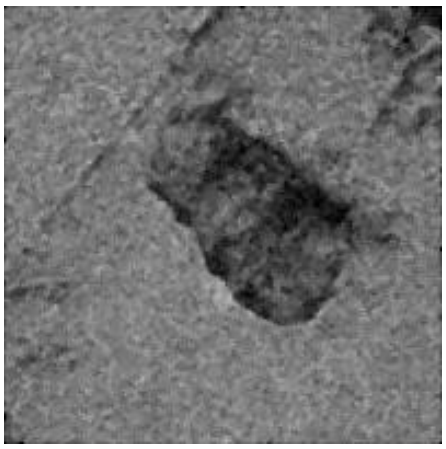
(c) Particle swarm optimization algorithm



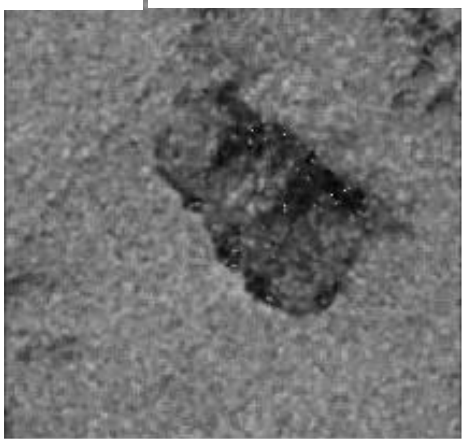
(d) Invasive weed optimization algorithm



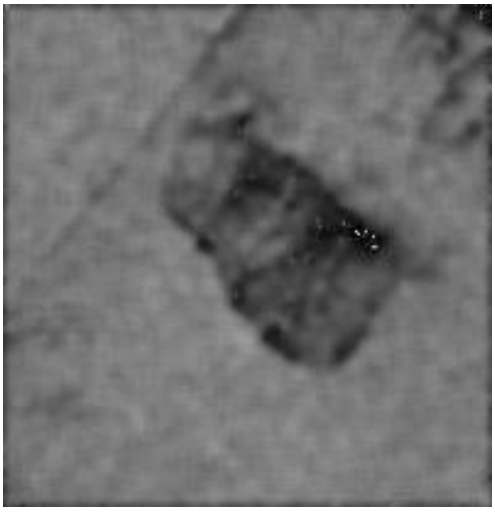
(a) Moth-flame optimization algorithm



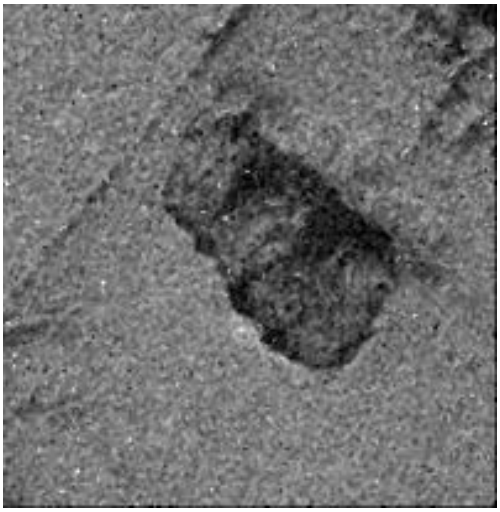
(b) Grasshopper optimization algorithm



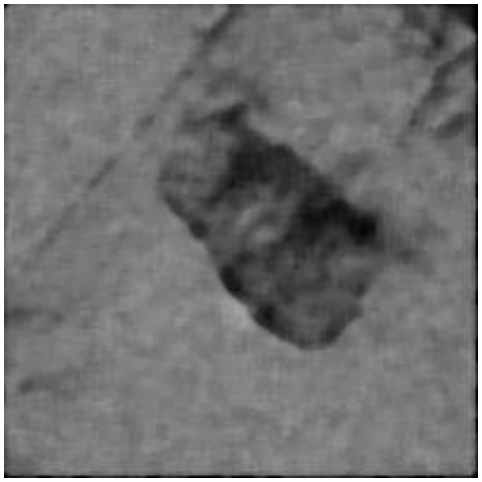
(c) Grey wolf optimization algorithm



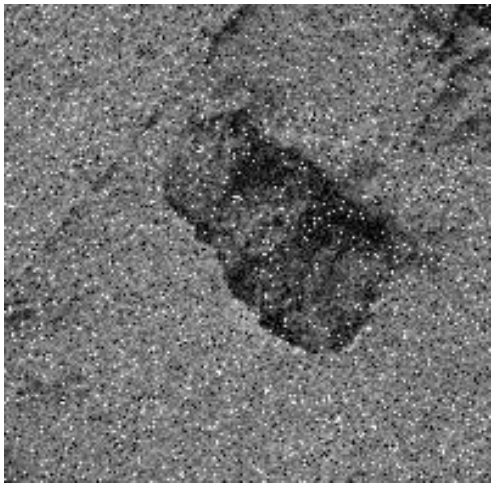
(a) Genetic algorithm



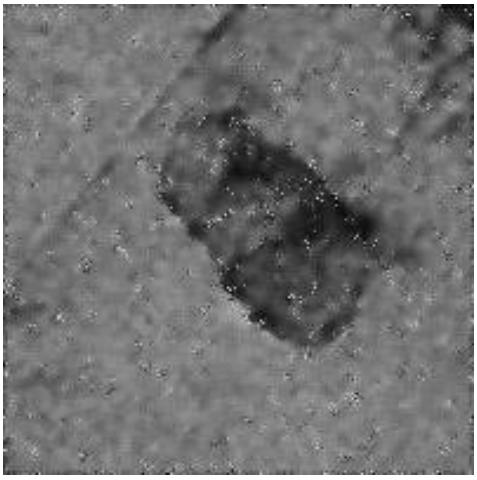
(b) Non-linear programming



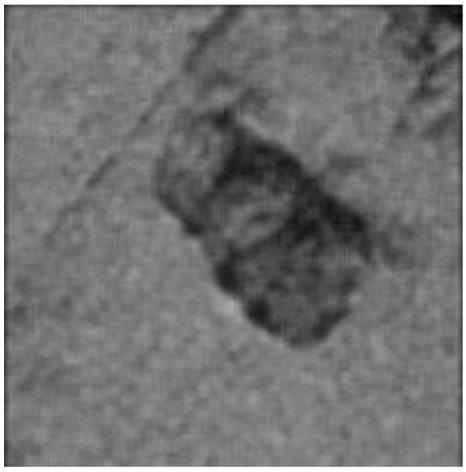
(a) Median filter



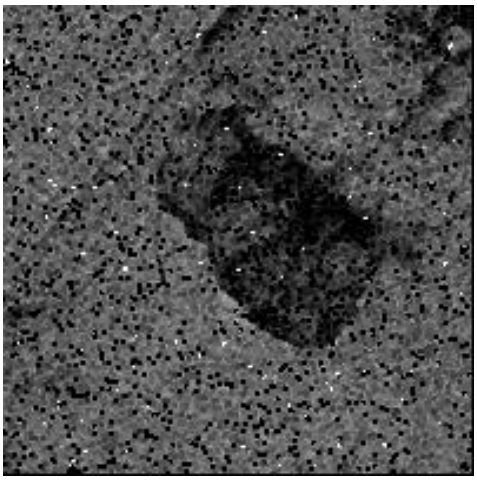
(b) Gaussian filter



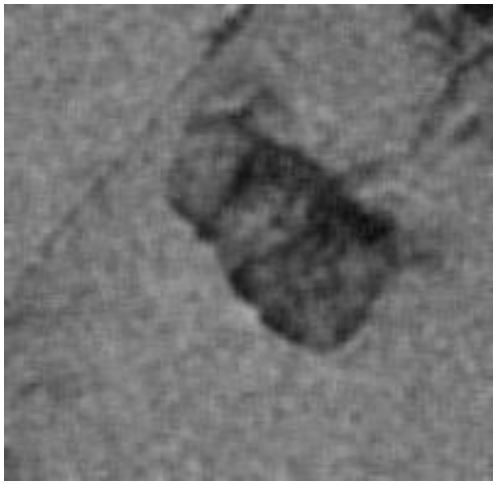
(c) Wiener filter



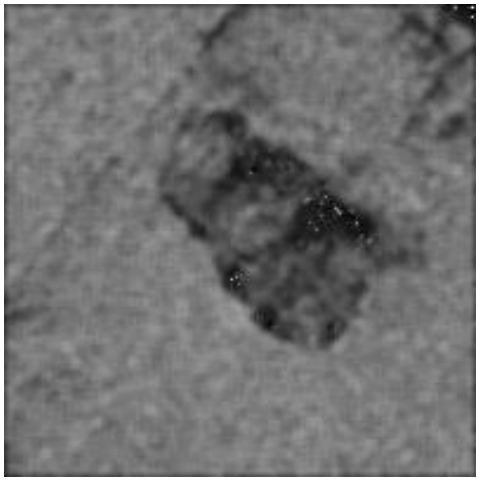
(d) Average filter



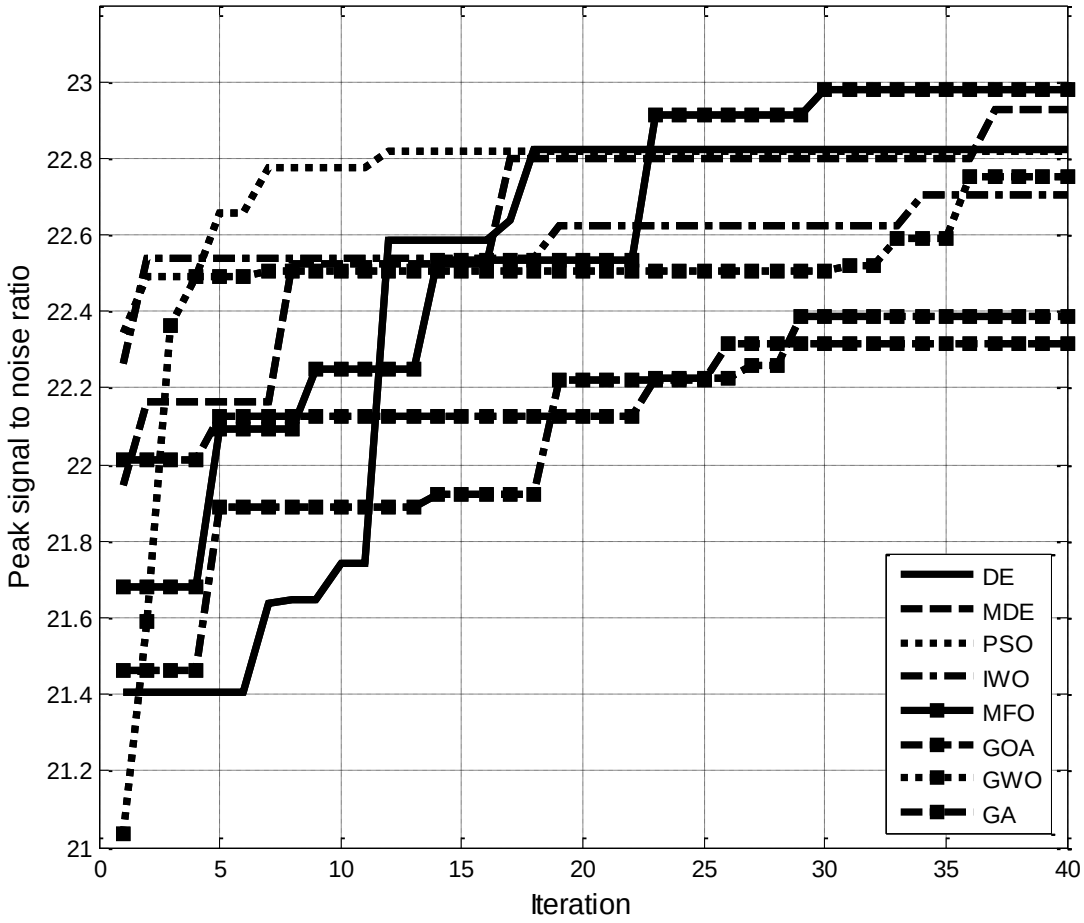
(a) Mode filter

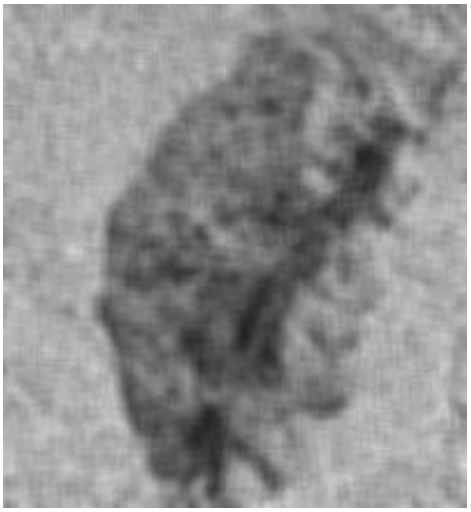


(b) Lee filter



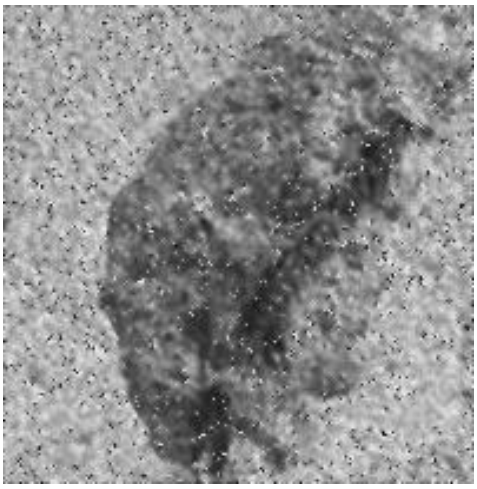
(c) Frost filter


Peak signal to noise ratio
Iteration
23
22.8
22.6
22.4
22.2
22
21.8
21.6
21.4
21.2
21
0
5
10
15
20
25
30
35
40
DE
MDE
PSO
IWO
MFO
GOA
GWO
GA

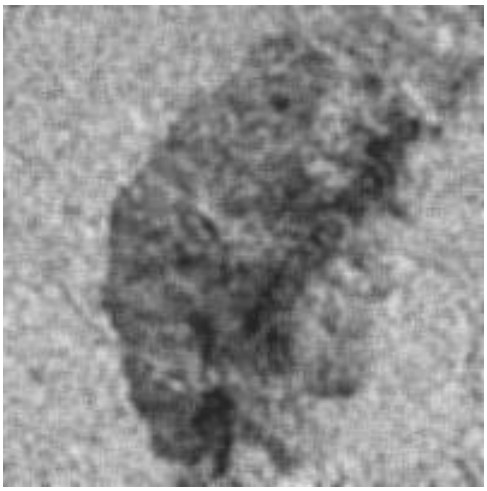


(a) Differential evolution algorithm

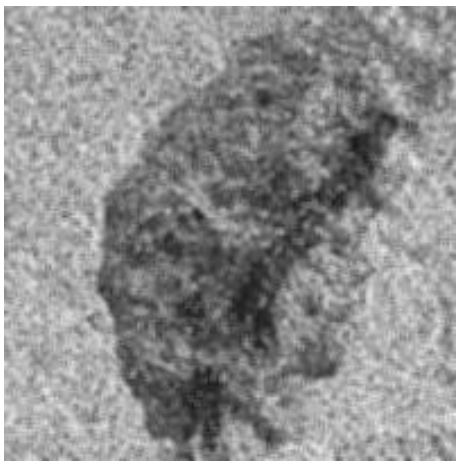
(b) Modified differential evolution algorithm



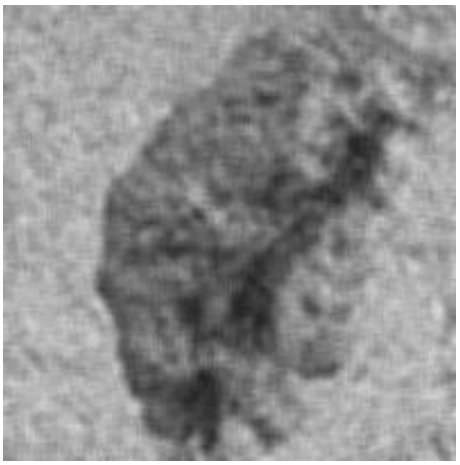
(c) Particle swarm optimization algorithm



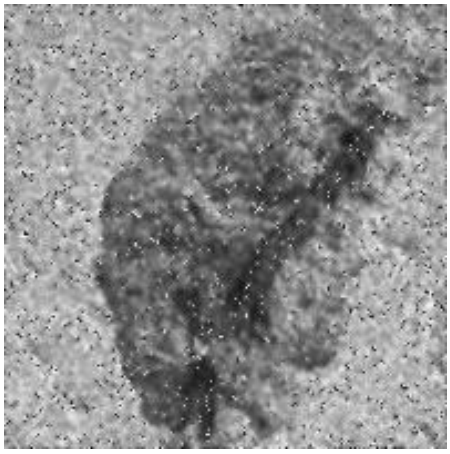
(d) Invasive weed optimization algorithm



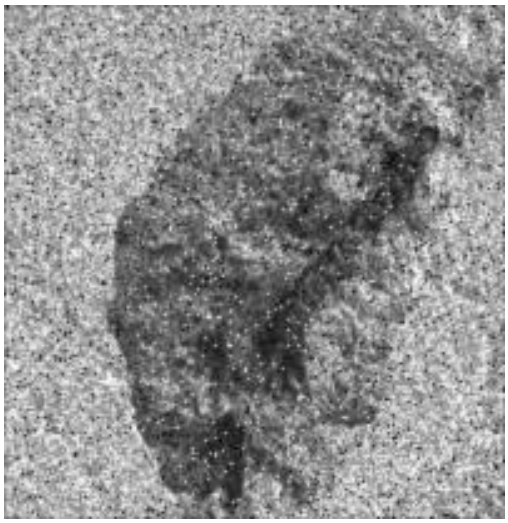
(a) Moth-flame optimization algorithm



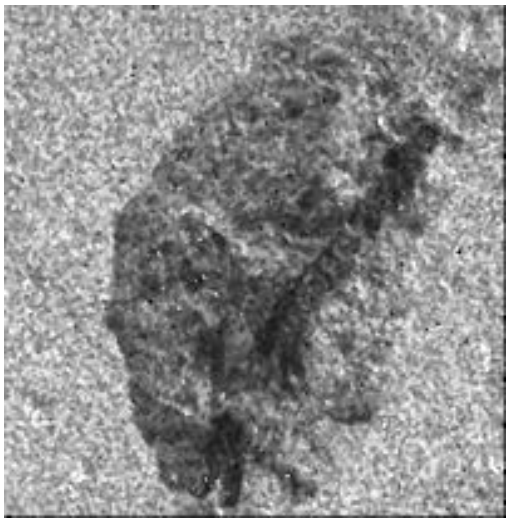
(b) Grasshopper optimization algorithm



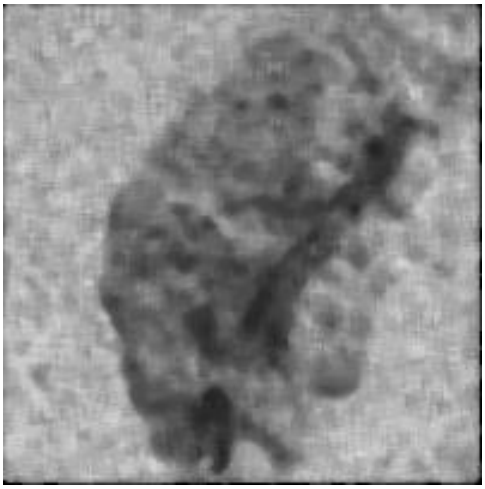
(c) Grey wolf optimization algorithm



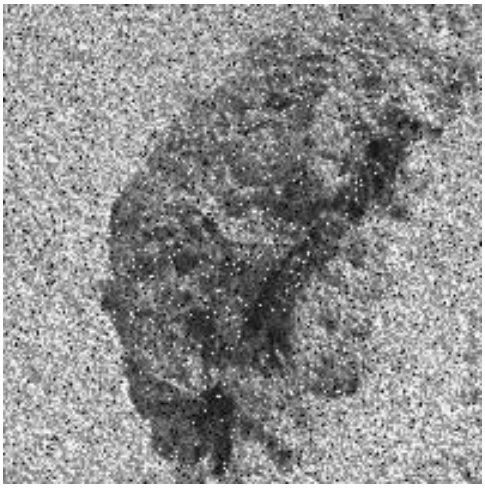
(a) Genetic algorithm



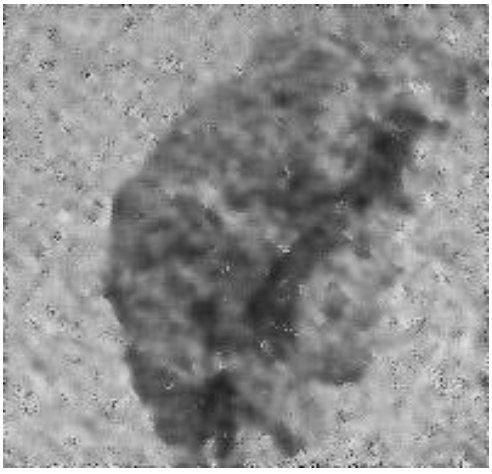
(b) Non-linear programming



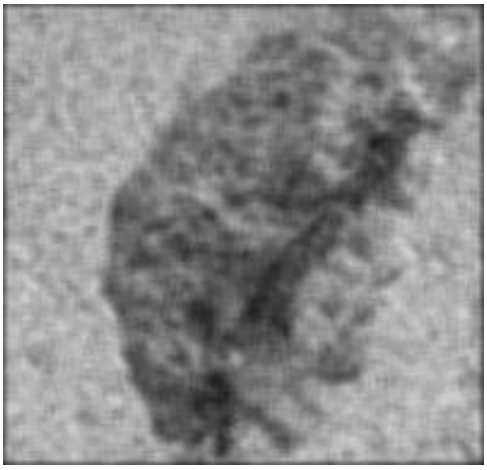
(a) Median filter



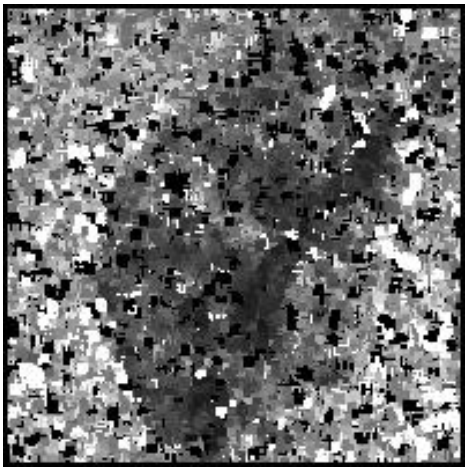
(b) Gaussian filter



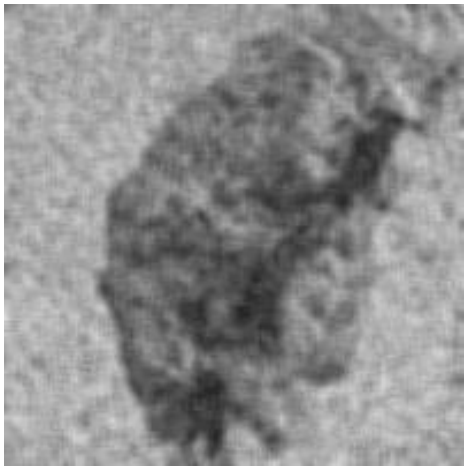
(c) Wiener filter



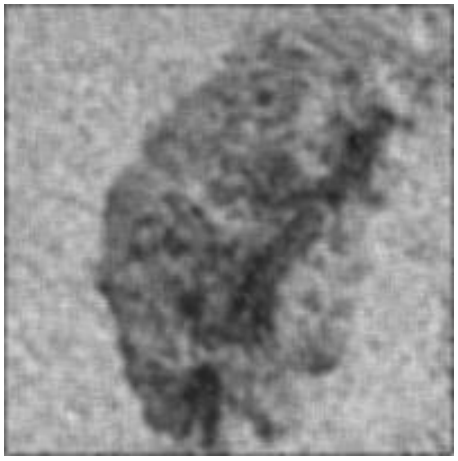
(d) Average filter



(a) Mode filter

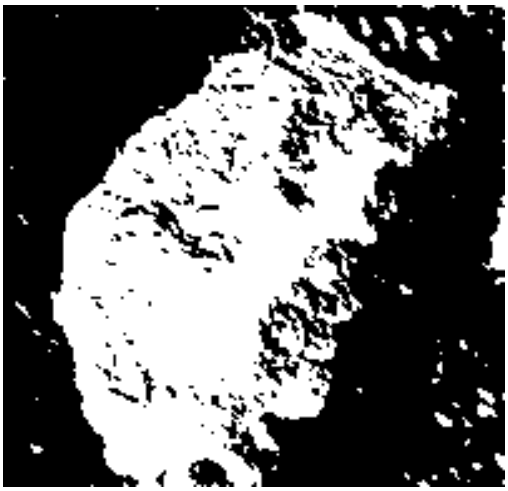


(b) Lee filter



(c) Frost filter

(a) Proposed restoration model



(b) Median filter

List of Tables

Table 1: Parameters of IWO algorithm for noise detection module

Decision Variable	Range
Initial population size	100
Maximum number of iterations	200
Minimum number of seeds	0
Maximum number of seeds	5
Initial standard deviation	0.5
Final standard deviation	0.001

Table 2: Performance comparison of the different machine learning models in noise detection based on split validation

Type of classifier	Accuracy	Sensitivity	specificity	Precision	F-measure	Kappa coefficient
DA	95.5%	92%	96%	76.67%	83.64%	0.81
KNN	98%	100%	97.7%	86.67%	92.86%	0.917
RF	98%	96.43%	98.26%	90%	93.1%	0.919
SVM	98.5%	93.55%	99.41%	96.67%	95.08%	0.942
ANN	91%	65%	97.5%	86.67%	74.29%	0.689
ENN-IWO	99%	100%	98.84%	93.33%	96.55%	0.959

Table 3: Performance comparison of the different machine learning models in noise detection based on 10-fold cross validation

Type of classifier	Accuracy	Sensitivity	specificity	Precision	F-measure	Kappa coefficient
DA	95.21%	91.68%	95.57%	76.42%	83.51%	0.808
KNN	97.71%	99.65%	97.26%	86.38%	92.72%	0.914
RF	97.7%	96.09%	97.82%	89.70%	92.96%	0.916
SVM	98.20%	93.22%	98.96%	96.35%	94.94%	0.939
ANN	90.73%	64.77%	97.06%	86.38%	74.18%	0.687
ENN-IWO	98.72%	99.65%	98.52%	93.12%	96.39%	0.956

Table 4: Confusion matrix of the classification of the ENN-IWO model for separate noise recognition

		Predicted class			
		Class 1	**Class 2**	**Class 3**	**Class 4**
Actual class	**Class 1**	29	0	1	0
	Class 2	0	24	0	1
	Class 3	0	0	55	0
	Class 4	0	4	0	21

Table 5: Comparison of the performance metrics of the six classification models for separate noise recognition based on split validation

Type of classifier	Accuracy	Sensitivity	specificity	Precision	F-measure	Kappa coefficient
DA	83.7%	83.7%	94.26%	83.7%	83.7%	0.77
KNN	85.93%	85.93%	95.08%	85.93%	85.93%	0.801
RF	91.85%	91.85%	97.21%	91.85%	91.85%	0.887
SVM	86.67%	86.67%	95.36%	86.67%	86.67%	0.813
ANN	84.44%	84.44%	94.53%	84.44%	84.44%	0.781
ENN-IWO	95.56%	95.56%	98.5%	95.56%	95.56%	0.937

Table 6: Comparison of the performance metrics of the six classification models for separate noise recognition based on 10-fold cross validation

Type of classifier	Accuracy	Sensitivity	specificity	Precision	F-measure	Kappa coefficient
DA	83.45%	83.41%	93.84%	83.42%	83.57%	0.768
KNN	85.67%	85.63%	94.65%	85.65%	85.80%	0.799
RF	91.57%	91.53%	96.77%	91.55%	91.71%	0.884
SVM	86.41%	86.37%	94.93%	86.38%	86.54%	0.811
ANN	84.19%	84.14%	94.10%	84.16%	84.31%	0.779
ENN-IWO	95.28%	95.24%	98.07%	95.25%	95.43%	0.935

Table 7: Confusion matrix of the classification of the ENN-IWO model for combined noise recognition

Predicted class

Actual class		**Class 1**	**Class 2**	**Class 3**	**Class 4**	**Class 5**	**Class 6**	**Class 7**
	Class 1	27	0	3	0	0	0	0
	Class 2	0	22	0	0	2	0	1
	Class 3	1	0	57	0	2	0	0
	Class 4	0	15	0	8	0	1	1
	Class 5	0	1	0	0	19	0	0
	Class 6	0	0	0	0	0	20	0
	Class 7	0	0	0	3	0	1	16

Table 8: Comparison of the performance metrics of the six classification models for combined noise recognition based on split validation

Type of classifier	Accuracy	Sensitivity	specificity	Precision	F-measure	Kappa coefficient
DA	82.5%	82.5%	96.97%	82.5%	82.5%	0.787
KNN	82.5%	82.5%	96.94%	82.5%	82.5%	0.787
RF	75.5%	81.62%	95.48%	75.5%	78.44%	0.704
SVM	82.5%	82.5%	96.97%	82.5%	82.5%	0.691
ANN	81%	81%	96.62%	81%	81%	0.77
ENN-IWO	84.5%	84.5%	97.33%	84.5%	84.5%	0.812

Table 9: Comparison of the performance metrics of the six classification models for combined noise recognition based on 10-fold cross validation

Type of classifier	Accuracy	Sensitivity	specificity	Precision	F-measure	Kappa coefficient
DA	82.25%	82.21%	96.53%	82.23%	82.38%	0.785
KNN	82.25%	82.21%	96.50%	82.23%	82.38%	0.785
RF	75.27%	81.33%	95.05%	75.25%	78.32%	0.702
SVM	82.25%	82.21%	96.53%	82.23%	82.38%	0.689
ANN	80.76%	80.72%	96.19%	80.73%	80.88%	0.768
ENN-IWO	84.26%	84.22%	97.21%	84.45%	84.46%	0.811

Table 10: Statistical comparison between the different noise classification models based on two-tailed Student's t-test

Pair of classifiers	Discriminant analysis	K-nearest neighbors	Random Forest	Support vector machines	Artificial neural network	ENN-IWO
Discriminant analysis	H_0 ($P-value=1$)	H_1 ($P-value=1.87\times10^{-8}$)	H_0 ($P-value=3.01\times10^{-1}$)	H_1 ($P-value=2.63\times10^{-8}$)	H_1 ($P-value=1.51\times10^{-4}$)	H_1 ($P-value=6.8\times10^{-8}$)
K-nearest neighbors	H_1 ($P-value=1.87\times10^{-8}$)	H_0 ($P-value=1$)	H_0 ($P-value=6.98\times10^{-1}$)	H_1 ($P-value=8.18\times10^{-7}$)	H_1 ($P-value=1.38\times10^{-7}$)	H_1 ($P-value=2.55\times10^{-6}$)
Random Forest	H_0 ($P-value=3.01\times10^{-1}$)	H_0 ($P-value=6.98\times10^{-1}$)	H_0 ($P-value=1$)	H_0 ($P-value=4.22\times10^{-1}$)	H_1 ($P-value=1.28\times10^{-2}$)	H_1 ($P-value=4.03\times10^{-8}$)
Support vector machines	H_1 ($P-value=2.63\times10^{-8}$)	H_1 ($P-value=8.18\times10^{-7}$)	H_0 ($P-value=4.22\times10^{-1}$)	H_0 ($P-value=1$)	H_1 ($P-value=4.2\times10^{-8}$)	H_1 ($P-value=5.56\times10^{-6}$)
Artificial neural network	H_1 ($P-value=1.51\times10^{-4}$)	H_1 ($P-value=1.38\times10^{-7}$)	H_1 ($P-value=1.28\times10^{-2}$)	H_1 ($P-value=4.2\times10^{-8}$)	H_0 ($P-value=1$)	H_1 ($P-value=1.66\times10^{-13}$)
ENN-IWO	H_1 ($P-value=6.8\times10^{-8}$)	H_1 ($P-value=2.55\times10^{-6}$)	H_1 ($P-value=4.03\times10^{-8}$)	H_1 ($P-value=5.56\times10^{-6}$)	H_1 ($P-value=1.66\times10^{-13}$)	H_0 ($P-value=1$)

Table 11: Statistical comparison of the developed noise classification model against other models based on non-parametric tests

Pair of classifiers	Wilcoxn	Mann-Whitney-U	Kruskal–Wallis	Binomial sign	Mood's median
Discriminant analysis, ENN-IWO	H_1 ($P-value = 5.39\times10^{-7}$)	H_1 ($P-value = 1.17\times10^{-6}$)	H_1 ($P-value = 0$)	H_1 ($P-value = 0$)	H_1 ($P-value = 1\times10^{-3}$)
K-nearest neighbors, ENN-IWO	H_1 ($P-value = 5.39\times10^{-7}$)	H_1 ($P-value = 6.06\times10^{-3}$)	H_1 ($P-value = 6\times10^{-3}$)	H_1 ($P-value = 0$)	H_1 ($P-value = 1\times10^{-3}$)
Random Forest, ENN-IWO	H_1 ($P-value = 5.39\times10^{-7}$)	H_1 ($P-value = 1.83\times10^{-2}$)	H_1 ($P-value = 1.8\times10^{-2}$)	H_1 ($P-value = 0$)	H_1 ($P-value = 1\times10^{-3}$)
Support vector machines, ENN-IWO	H_1 ($P-value = 5.39\times10^{-7}$)	H_1 ($P-value = 1.83\times10^{-2}$)	H_1 ($P-value = 1.8\times10^{-2}$)	H_1 ($P-value = 0$)	H_1 ($P-value = 1\times10^{-3}$)
Artificial neural network, ENN-IWO	H_1 ($P-value = 5.39\times10^{-7}$)	H_1 ($P-value = 6.93\times10^{-7}$)	H_1 ($P-value = 0$)	H_1 ($P-value = 0$)	H_1 ($P-value = 1\times10^{-3}$)

Table 12: Average ranking of the noise classification models using Friedman test and Friedman's aligned ranks test

Type of classifier	Friedman	Friedman's aligned ranks
Discriminant analysis	4.69	124.21
K-nearest neighbors	3.66	110.51
Random Forest	3.83	106.19
Support vector machines	2.63	81.21
Artificial neural network	5.16	150.71
ENN-IWO	1	24.15

Table 13: P − values of the EN-IWO noise classification model using Nemenyi test, Holm test and Finner test

Pair of classifiers	Friedman			Friedman's aligned ranks		
	Nemenyi	Holm	Finner	Nemenyi	Holm	Finner
Discriminant analysis, ENN-IWO	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)
K-nearest neighbors, ENN-IWO	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)
Random Forest, ENN-IWO	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)
Support vector machines, ENN-IWO	H_1 (P − value $=5.71\times10^{-3}$)	H_1 (P − value $=3.43\times10^{-3}$)	H_1 (P − value $=8.2\times10^{-4}$)	H_1 (P − value $=7.9\times10^{-4}$)	H_1 (P − value $=5.2\times10^{-4}$)	H_1 (P − value $=1.3\times10^{-4}$)
Artificial neural network, ENN-IWO	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)

Table 14: Average run time in seconds of different classification models of noise detection

Average run time	Discriminant analysis	K-nearest neighbors	Random Forest	Support vector machines	Artificial neural network	ENN-IWO
Training time	3.36	4.84	62.73	39.87	523.11	1203.56
Classification time	0.034	0.035	0.036	0.035	0.037	0.039

Table 15: Average run time in seconds of different classification models of separate noise recognition

Average run time	Discriminant analysis	K-nearest neighbors	Random Forest	Support vector machines	Artificial neural network	ENN-IWO
Training time	4.23	5.96	77.16	60.65	620.42	1250.42
Classification time	0.037	0.037	0.038	0.037	0.039	0.041

Table 16: Average run time in seconds of different classification models for combined noise recognition

Average run time	Discriminant analysis	K-nearest neighbors	Random Forest	Support vector machines	Artificial neural network	ENN-IWO
Training time	4.77	6.96	89.09	65.23	702.85	1652.65
Classification time	0.044	0.046	0.047	0.048	0.046	0.046

Table 17: Performance evaluation of different restoration models for "Image 1" corrupted with Gaussian noise

Restoration model	Optimum design of filter(s)	PSNR	MSE	NAE	IEF
DE algorithm	Lee filter of size 2×2	24.47	235.06	0.083	2.47
MDE algorithm	Frost filter of size 3×3	25.24	185.88	0.074	3.48
PSO algorithm	Wiener filter of size 2×2	25.28	192.31	0.076	3.34
IWO algorithm	Lee filter of size 3×3 followed by mode filter of size 5×5	25.01	303.77	0.093	2.61
MFO algorithm	Wiener filter of size 3×3	25.29	185.75	0.074	5.16
GOA	Average filter of size 3×3 followed by Wiener filter of size 3×3	23.3	411.62	0.103	1.68
GWO algorithm	Lee filter of size 3×3	25.14	250.23	0.084	2.6
GA	Gaussian filter of Size 3×3 and sigma of 0.6	25.18	186.06	0.074	3.47
Nonlinear programming	Median filter of size 2×2	22.12	406.89	0.104	1.56
Median filter	Size 4×4	22.64	292.86	0.089	2.23
Gaussian filter	Size 4×4 and sigma of 0.4	21.45	465.20	0.121	1.87
Weiner filter	Size 4×4	23.09	195.96	0.084	2.17
Mean filter	Size 4×4	20.02	646.74	0.143	1.12
Mode filter	Size 4×4	16.73	1381.15	0.211	0.47
Lee filter	Size 4×4	23.02	251.09	0.082	2.57
Frost filter	Size 4×4	22.13	317.34	0.091	2.04

Table 18: Performance evaluation of different restoration models for "Image 2" corrupted with salt and pepper noise

Restoration model	Optimum design of filter(s)	PSNR	MSE	NAE	IEF
DE algorithm	Lee filter of size 2×2	30.03	66.94	0.038	13.15
MDE algorithm	Gaussian filter of Size 5×5 and sigma of 0.17	30.07	67.35	0.039	13.61
PSO algorithm	Gaussian filter of Size 3×3 and sigma of 0.48 followed by Lee filter of Size 3×3	24.94	287.99	0.092	2.48
IWO algorithm	Median filter of size 3×3 followed by frost filter of size 3×3	25.45	135.44	0.057	13.47
MFO algorithm	Median filter of size 3×3	30.11	65.29	0.038	14.24
GOA	Median filter of size 2×2 followed by frost filter of size 3×3	23.91	199.6	0.066	4.62
GWO algorithm	Frost filter of size 3×3	29.90	72.41	0.039	12.66
GA	Frost filter of size 2×2	28.73	65.31	0.039	14.18
Nonlinear programming	Median filter of size 2×2	25.36	199.89	0.058	4.57
Median filter	Size 4×4	21.37	303.8	0.078	2.87
Gaussian filter	Size 4×4 and sigma of 0.4	19.9	631.11	0.049	1.65
Weiner filter	Size 4×4	22.7	292.09	0.078	3.05
Average filter	Size 4×4	21.88	279.46	0.088	3.18
Mode filter	Size 4×4	13.74	2748.22	0.194	0.32
Lee filter	Size 4×4	23.54	192.09	0.078	4.75
Frost filter	Size 4×4	22.39	244.75	0.084	3.63

Table 19: Performance evaluation of different restoration models for "Image 3" corrupted with speckle noise

Restoration model	Optimum design of filter(s)	PSNR	MSE	NAE	IEF
DE algorithm	Gaussian filter of Size 4×4 and sigma of 0.25 followed by Lee filter of Size 3×3	24.31	307.36	0.094	3.56
MDE algorithm	Wiener filter of size 3×3	24.49	270.5	0.088	4.05
PSO algorithm	Lee filter of size 2×2	24.44	566.83	0.124	2.32
IWO algorithm	Frost filter of Size 3×3 followed by Wiener filter of Size 2×2	23.45	276.34	0.088	4.02
MFO algorithm	Lee filter of size 3×3	24.57	228.9	0.082	4.13
GOA	Frost filter of Size 3×3	23.8	271.24	0.087	3.57
GWO algorithm	Frost filter of size 2×2	24.26	307.32	0.093	3.55
GA	Gaussian filter of Size 4×4 and sigma of 0.25 followed by Lee filter of Size 3×3	24.21	272.1	0.089	1.72
Nonlinear programming	Average filter of size 3×3 followed by mode filter of size 4×4	20.16	640.86	0.136	4.08
Median filter	Size 4×4	19.94	511	0.112	2.15
Gaussian filter	Size 4×4 and sigma of 0.4	19.11	796.62	0.164	1.89
Weiner filter	Size 4×4	23.13	275.39	0.089	3.99
Average filter	Size 4×4	21.25	378.79	0.099	2.9
Mode filter	Size 4×4	15	2055.63	0.264	0.54
Lee filter	Size 4×4	22.4	270.82	0.088	3.07
Frost filter	Size 4×4	21.62	331.94	0.095	2.29

Table 20: Performance evaluation of different restoration models for "Image 4" corrupted with a combination of Gaussian and speckle noises

Restoration model	Optimum design of filter(s)	PSNR	MSE	NAE	IEF
DE algorithm	Wiener filter of size 4×4 followed by median filter of size 3×3	24.27	195.14	0.095	5.59
MDE algorithm	Lee filter of size 3×3 followed by median filter of size 3×3	24.49	180.49	0.092	4.67
PSO algorithm	Wiener filter of size 3×3 followed by median filter of size 3×3	24.58	177.6	0.091	4.72
IWO algorithm	Wiener filter of size 5×5 followed by Gaussian filter of Size 3×3 and sigma of 0.5	24.16	193.02	0.094	6.22
MFO algorithm	Wiener filter of size 3×3 followed by Lee filter of size 3×3	25.14	158.51	0.084	7.87
GOA	Wiener filter of size 3×3 followed by average filter of size 3×3	24.08	264.4	0.104	5.62
GWO algorithm	Lee filter of size 3×3 followed by Wiener filter of size 3×3	24.98	155.04	0.086	6.82
GA	Lee filter of size 3×3 followed by Wiener filter of size 6×6	24.24	180.54	0.092	7.64
Nonlinear programming	Median filter of size 4×4 followed by Gaussian filter of Size 5×5 and sigma of 0.7	15.24	1977.65	0.339	0.69
Median filter	Size 4×4	19.51	336.96	0.116	4.06
Gaussian filter	Size 4×4 and sigma of 0.4	17.93	985.79	0.223	2.18
Weiner filter	Size 4×4	23.65	217.6	0.099	5.22
Average filter	Size 4×4	21.90	234.15	0.100	5.88
Mode filter	Size 4×4	15.06	2029	0.343	0.68
Lee filter	Size 4×4	23.22	178.56	0.098	5.71
Frost filter	Size 4×4	22.53	208.21	0.096	5.58

Table 21: Performance evaluation of different restoration models for "Image 5" corrupted with a combination of Gaussian and salt and pepper noises

Restoration model	Optimum design of filter(s)	PSNR	MSE	NAE	IEF
DE algorithm	Frost filter of size 5×5	24.6	223.72	0.102	6.63
MDE algorithm	Median filter of size 2×2	24.93	206.13	0.101	7.24
PSO algorithm	Frost filter of size 4×4 followed by Wiener filter of size 3×3	24.87	214.92	0.099	6.77
IWO algorithm	Median filter of size 5×5 followed by Wiener filter of size 2×2	24.39	373.08	0.129	3.67
MFO algorithm	Median filter of size 3×3	25.06	180.19	0.095	8.61
GOA	Frost filter of size 3×3	25.05	208.41	0.102	7.02
GWO algorithm	Frost filter of size 4×4 followed by Wiener filter of size 4×4	24.37	205.84	0.101	7.25
GA	Frost filter of size 5×5	24.67	224.34	0.102	6.47
Nonlinear programming	Gaussian filter of Size 2×2 and sigma of 0.02 followed by median filter of Size 2×2	21.16	414.73	0.133	3.59
Median filter	Size 4×4	19.7	284.55	0.104	5.08
Gaussian filter	Size 4×4 and sigma of 0.4	17.52	1075.86	0.198	1.38
Weiner filter	Size 4×4	22.37	283.38	0.107	5.15
Average filter	Size 4×4	20.85	239.55	0.106	6.13
Mode filter	Size 4×4	14.13	2511.65	0.341	0.6
Lee filter	Size 4×4	21.64	190.31	0.098	7.99
Frost filter	Size 4×4	24.42	226.85	0.103	5.63

Table 22: Performance evaluation of different restoration models for "Image 6" corrupted with a combination of speckle and salt and pepper noises

Restoration model	Optimum design of filter(s)	PSNR	MSE	NAE	IEF
DE algorithm	Lee filter of size 3×3 followed by Wiener filter of size 4×4	22.82	379.61	0.106	5.43
MDE algorithm	Lee filter of size 3×3 followed by Wiener filter of size 2×2	22.93	350.7	0.102	5.68
PSO algorithm	Wiener filter of size 3×3 followed by median filter of size 3×3	22.82	285.68	0.092	3.34
IWO algorithm	Lee filter of size3×3	22.71	339.44	0.101	5.78
MFO algorithm	Lee filter of size 3×3 followed by Wiener filter of size 3×3	22.98	290.61	0.091	5.91
GOA	Lee filter of size 3×3 followed by Wiener filter of size 6×6	22.39	323.4	0.098	5.83
GWO algorithm	Wiener filter of size 3×3 followed by median filter of size 2×2	22.75	297.94	0.093	3.25
GA	Gaussian filter of Size 3×3 and sigma of 0.54 followed by median filter of Size 3×3	22.32	340.52	0.101	2.84
Nonlinear programming	Median filter of size 3×3	19.72	720.69	0.144	2.72
Median filter	Size 4×4	19.14	603.06	0.118	3.23
Gaussian filter	Size 4×4 and sigma of 0.4	16.67	1382.63	0.193	1.31
Weiner filter	Size 4×4	20.88	434.14	0.108	4.59
Average filter	Size 4×4	20.04	448.64	0.111	4.32
Mode filter	Size 4×4	10.28	6099.58	0.405	0.32
Lee filter	Size 4×4	21.66	360.59	0.099	4.82
Frost filter	Size 4×4	20.93	392.7	0.106	3.95

Table 23: Overall performance evaluation of the different types of restoration models

Restoration model	APSNR	AMSE	ANAE	AIEF
DE algorithm	24.94	180.71	0.06	6.82
MDE algorithm	25.23	177.59	0.059	6.9
PSO algorithm	24.56	184.26	0.061	6.71
IWO algorithm	24.32	186.57	0.062	6.65
MFO algorithm	25.36	176.32	0.059	7.18
GOA	23.92	191.71	0.064	6.54
GWO algorithm	25.11	178.72	0.059	6.86
GA	24.52	184.91	0.062	6.7
Nonlinear programming	20.3	415.34	0.099	3.09
Median filter	20.23	418.92	0.106	3.1
Gaussian filter	18.72	884.72	0.154	1.38
Weiner filter	22.73	290.04	0.093	4.26
Average filter	20.97	355.99	0.104	3.86
Mode filter	14.14	2701.5	0.284	0.53
Lee filter	22.47	241.72	0.09	5.17
Frost filter	21.68	291.38	0.096	4.34

Table 24: Filtering protocol for different types of noises

Type of noise	Restoration model	Optimum design of filter(s)
Gaussian	MFO algorithm	Wiener filter of size 3×3
Salt and pepper	MFO algorithm	Median filter of size 3×3
Speckle	MFO algorithm	Lee filter of size 3×3
Combination of Gaussian and speckle noises	MFO algorithm	Wiener filter of size 3×3 followed by Lee filter of size 3×3
Combination of Gaussian and salt and pepper noises	MFO algorithm	Median filter of size 3×3
Combination of Speckle and salt and pepper noises	MFO algorithm	Lee filter of size 3×3 followed by Wiener filter of size 3×3

Table 25: Statistical comparison between the different meta-heuristic-based restoration models based on two-tailed Student's t-test

Pair of meta-heuristics	DE	MDE	PSO	IWO	MFO	GOA	GWO	GA
DE	H_0 (P − value=1)	H_1 (P − value =2.46×10^{-2})	H_1 (P − value =9.69×10^{-3})	H_1 (P − value =3.39×10^{-3})	H_1 (P − value =2.6×10^{-3})	H_1 (P − value =1.41×10^{-4})	H_1 (P − value =4.32×10^{-2})	H_1 (P − value =1.2×10^{-5})
MDE	H_1 (P − value =2.46×10^{-2})	H_0 (P − value=1)	H_1 (P − value =7.6×10^{-2})	H_1 (P − value =1.89×10^{-2})	H_1 (P − value =1.47×10^{-3})	H_1 (P − value =5.14×10^{-4})	H_0 (P − value =2.38×10^{-1})	H_1 (P − value =2.02×10^{-3})
PSO	H_1 (P − value =9.69×10^{-3})	H_1 (P − value =7.6×10^{-2})	H_0 (P − value=1)	H_0 (P − value =2.92×10^{-1})	H_1 (P − value =2.09×10^{-3})	H_1 (P − value =1.11×10^{-4})	H_1 (P − value =4.07×10^{-2})	H_0 (P − value =8.53×10^{-1})
IWO	H_1 (P − value =3.39×10^{-3})	H_1 (P − value =1.89×10^{-2})	H_0 (P − value =2.92×10^{-1})	H_0 (P − value=1)	H_1 (P − value =8×10^{-4})	H_0 (P − value =1.76×10^{-1})	H_1 (P − value =1.29×10^{-2})	H_0 (P − value =4.25×10^{-1})
MFO	H_1 (P − value =2.6×10^{-3})	H_1 (P − value =1.47×10^{-3})	H_1 (P − value =2.09×10^{-3})	H_1 (P − value =8×10^{-4})	H_0 (P − value=1)	H_1 (P − value =2.9×10^{-5})	H_1 (P − value =9.65×10^{-6})	H_1 (P − value =1.27×10^{-7})
GOA	H_1 (P − value =1.41×10^{-4})	H_1 (P − value =1.47×10^{-3})	H_1 (P − value =1.11×10^{-4})	H_0 (P − value =1.76×10^{-1})	H_1 (P − value =2.9×10^{-5})	H_0 (P − value=1)	H_1 (P − value =6.47×10^{-4})	H_1 (P − value =3.46×10^{-2})
GWO	H_1 (P − value =4.32×10^{-2})	H_0 (P − value =2.38×10^{-1})	H_1 (P − value =4.07×10^{-2})	H_1 (P − value =1.29×10^{-2})	H_1 (P − value =9.65×10^{-6})	H_1 (P − value =6.47×10^{-4})	H_0 (P − value=1)	H_1 (P − value =2.52×10^{-4})
GA	H_1 (P − value =1.2×10^{-5})	H_1 (P − value =2.02×10^{-3})	H_0 (P − value =8.53×10^{-1})	H_0 (P − value =4.25×10^{-1})	H_1 (P − value =1.27×10^{-7})	H_1 (P − value =3.46×10^{-2})	H_1 (P − value =2.52×10^{-4})	H_0 (P − value=1)

Table 26: Statistical comparison of the developed restoration model against meta-heuristic-based models based on non-parametric tests

Pair of meta-heuristics	Wilcoxn	Mann-Whitney-U	Kruskal–Wallis	Binomial sign	Mood's median
DE, MFO	H_1 ($P-value = 8.58\times10^{-5}$)	H_1 ($P-value = ١,٦٤\times10^{-3}$)	H_1 ($P-value = 2\times10^{-3}$)	H_1 ($P-value = 3.13\times10^{-4}$)	H_1 ($P-value = 1.8\times10^{-2}$)
MDE, MFO	H_1 ($P-value = 9.17\times10^{-4}$)	H_1 ($P-value = ٨,٢٧\times10^{-3}$)	H_1 ($P-value = 8\times10^{-3}$)	H_1 ($P-value = 6.96\times10^{-5}$)	H_1 ($P-value = 1.8\times10^{-2}$)
PSO, MFO	H_1 ($P-value = 1.73\times10^{-4}$)	H_1 ($P-value = ١\times10^{-٥}$)	H_1 ($P-value = 0$)	H_1 ($P-value = 3.93\times10^{-3}$)	H_1 ($P-value = 1\times10^{-3}$)
IWO, MFO	H_1 ($P-value = 3.2\times10^{-5}$)	H_1 ($P-value = 4.4\times10^{-6}$)	H_1 ($P-value = 0$)	H_1 ($P-value = 1.19\times10^{-3}$)	H_1 ($P-value = 5\times10^{-3}$)
GOA, MFO	H_1 ($P-value = 1.11\times10^{-6}$)	H_1 ($P-value = 4.01\times10^{-8}$)	H_1 ($P-value = 0$)	H_1 ($P-value = 2.27\times10^{-7}$)	H_1 ($P-value = 0$)
GWO, MFO	H_1 ($P-value = 2.22\times10^{-5}$)	H_1 ($P-value = 2.69\times10^{-3}$)	H_1 ($P-value = 3\times10^{-3}$)	H_1 ($P-value = 3.13\times10^{-4}$)	H_1 ($P-value = 1.8\times10^{-2}$)
GA, MFO	H_1 ($P-value = 4.58\times10^{-7}$)	H_1 ($P-value = 4.09\times10^{-2}$)	H_1 ($P-value = 1\times10^{-3}$)	H_1 ($P-value = 1.94\times10^{-8}$)	H_1 ($P-value = 1.8\times10^{-2}$)

Table 27: Average ranking of the meta-heuristic-based restoration models using Friedman test and Friedman's aligned ranks test

Type of classifier	Friedman	Friedman's aligned ranks
Differential evolution algorithm	4.41	144.33
Modified differential evolution algorithm	3.88	109.05
Particle swarm optimization algorithm	4.5	159.69
Invasive weed optimization algorithm	5.27	181.61
Moth-flame optimization algorithm	1	29.77
Grasshopper optimization algorithm	6.38	214.22
Grey wolf optimization algorithm	4.38	131.72
Genetic algorithm	6.13	185.58

Table 28: P − values of the moth-flame-based restoration model using Nemenyi test, Holm test and Finner test

Pair of meta-heuristics	Friedman			Friedman's aligned ranks		
	Nemenyi	Holm	Finner	Nemenyi	Holm	Finner
DE, MFO	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)
MDE, MFO	H_1 (P − value $=2\times10^{-5}$)	H_1 (P − value $=1\times10^{-5}$)	H_1 (P − value =0)	H_1 (P − value $=1.51\times10^{-5}$)	H_1 (P − value =0)	H_1 (P − value =0)
PSO, MFO	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)
IWO, MFO	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)
GOA, MFO	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)
GWO, MFO	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value $=1\times10^{-5}$)	H_1 (P − value =0)	H_1 (P − value =0)
GA, MFO	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)	H_1 (P − value =0)

Table 29: Average run time in seconds of different restoration models

Average run time	DE	MDE	PSO	IWO	MFO	GOA	GWO	GA	NLP	Conventional restoration models
Training time	748.45	899.09	711.84	1304.62	915.38	1062.12	787.37	559.41	475.72	…
Restoration time	2.93	2.7	3.15	3.83	2.69	3.16	2.94	3.17	2.69	2.89

AS