IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 11 November 2022, accepted 24 November 2022, date of publication 9 December 2022,
date of current version 14 December 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3228176

RESEARCH ARTICLE

Ensemble Optimization for Invasive Ductal Carcinoma (IDC) Classification Using Differential Cartesian Genetic Programming

EID ALKHALDI, (Member, IEEE), AND EZZATOLLAH SALARI
Electrical Engineering and Computer Science Department, The University of Toledo, Toledo, OH 43606, USA
Corresponding author: Eid Alkhaldi (eid.alkhaldi@rockets.utoledo.edu)

ABSTRACT The high cost of acquiring annotated histological slides for breast specimens entails exploiting an ensemble of models appropriately trained on small datasets. Histological Image Classification ensembles strive to accurately detect abnormal tissues in the breast samples by determining the correlation between the predictions of its weak learners. Nonetheless, the state-of-the-art ensemble methods, such as boosting and bagging, count merely on manipulating the dataset and lack intelligent ensemble decision making. Furthermore, the methods mentioned above are short of the diversity of the weak models of the ensemble. Likewise, other commonly used voting strategies, such as weighted averaging, are limited to how the classifiers' diversity and accuracy are balanced. Hence, In this paper, we assemble a Neural Network ensemble that integrates the models trained on small datasets by employing biologically-inspired methods. Our procedure is comprised of two stages. First, we train multiple heterogeneous pre-trained models on the benchmark Breast Histopathology Images for Invasive Ductal Carcinoma (IDC) classification dataset. In the second meta-training phase, we utilize the differential Cartesian Genetic Programming (dCGP) to generate a Neural Network that merges the trained models optimally. We compared our empirical outcomes with other state-of-the-art techniques. Our results demonstrate that improvising a Neural Network ensemble using Cartesian Genetic Programming transcended formerly published algorithms on slim datasets.

INDEX TERMS Cartesian genetic programming, hyparameters optimization, ensemble, CNNs, histolpathaological image classification.

I. INTRODUCTION

Histopathology images carcinoma classification for breast specimens, stained with hematoxylin and eosin (H&E), has been investigated broadly due to the significance of the problem, the shortage of annotated images, and the increased expense of hiring radiologists [1], [2], [3], [4]. Breast cancer is one of the most deadly types of cancer. Around four-fifths of the deaths induced by breast cancer are caused by Invasive Ductal Carcinoma (IDC) [4]. IDC's diagnosis and prognosis demand examining Whole Slide Images (WSI) of breast biopsies stained with H&E for visual features improvements. The WSI are high-resolution images of the sample, usually divided into smaller sub-images for training machine learning (ML) classifiers.

As a result of the importance of the breast cancer automatic detection, several Machine Learning techniques have been suggested in the histopathology image classification literature.

Machine Learning has been broadly employed to solve scientific computational problems, data science, and predictive modeling in various domains over the past decades [5]. Deep Learning has also been used in the area of medical image classification as well as countless applications such as brain-computer interface using Graph Convolutional Neural Networks GCNs-Net [6].

The earliest methods relied on uniting several useful features to represent the image better. These features are

The associate editor coordinating the review of this manuscript and approving it for publication was Christos Anagnostopoulos.

handcrafted by observing how the negative IDC patches differ visually from the positive ones [4]. Many of these approaches centered around the nuclei features via nuclei segmentation such as nuclei position, centroid, density, glands segmentation, Voronoi-based features, Gabor filter, and the HSI color space [4]. Most of the handcrafted-based approaches depend on a composite of numerous characteristics mentioned earlier that extend the distinguishability of the categories. Cruz-Roa et al. [4] presented one of the foremost Convolutional Neural Network (CNN) models to categorize the BHI samples with a CNN [4]. CNN revealed an unprecedented advancement over the handcrafted feature approaches in histopathology image classification [1]. Nevertheless, the IDC classification of histopathology images stays challenging as a result of the insufficiency of data and the increased need of CNNs for labeled samples. Accordingly, more evolved techniques are needed to fetch the most precise predictions, such as transfer learning, domain adaptation, and ensemble optimization [1].

Equivalent to blending the handcrafted features, ensemble learning is concentrated on integrating a set of diverse classifiers into a more accurate and robust model. The concept is to reduce variance and biases in the classifiers leading to a more generalized model that would function better on unseen samples.

Ensemble optimization has been successfully applied into numerous applications such as anomaly detection, cyber security, image classification and a diverse set of applications [7]. Ensemble learning is one of the most successful machine learning practices [8]. Linking a group of complementary classifiers will generally result in a model that is at least more accurate than any of its components [9]. Complementary classifiers have to be both diverse and accurate enough for the resulting ensemble to be more robust.

While ensemble learning produces more accurate networks, training multiple classifiers is time-consuming [8]. To overcome this impediment, Huang et al. [8] proposed saving snapshots of the model during training instead of training multiple heterogenous models. Huang et al. method reduces the training time significantly by generating a group of the same model with different weights. However, the problem with homogeneous classifiers is that they are more inclined to overfit the dataset's peculiarities and are deprived of the diversity of the models. Additional ensemble learning methods were presented in the literature, such as Bayesian Averaging, bagging, boosting, vertical voting, and horizontal stacking [8], [9], [10], [11].

Due to the successful implementations of Evolutionary Algorithms (EA) in optimizing stacked ensembles established in the literature, a more adaptable EA seems exceptionally advantageous [1], [2], [11].

Biologically inspired methods have been verified to be robust and extensible. Numerous bio-inspired techniques have been extended to different variants that suit specific problems. For instance, PSO was improved to solve complex multi-objective optimization problems, which extended PSO to Multi-Objective Particle Swarm Optimization (MOPSO) algorithm by introducing the Space Expanding Strategy (SES) and mutation. The improved MOPSO was used to solve multiple optimization problems and was superior to three other commonly used multi-objective PSO methods [12]. The Random Neighbor Elite Guided Differential Evolution (RNEGDE) algorithm is another example of Evolutionary Algorithms extensibility [13]. RNEGDE produced optimal solutions compared to other state-of-the-art methods [13]. Hence, we emphasize leveraging the existing differential Cartesian Genetic Programming Artificial Neural Network (dCGPANN) to reach a competitive solution to the ensemble optimization problem for histological image classification.

The domain of medical image classification has acquired immense emphasis from the research community in the past decades. Numerous methods have been suggested to solve the invasive ductal detection problem. One of the earlier methods to automate the detection of IDC in histological images was proposed by Cruz-Roa et al. [4]. Cruz-Roa et al. compared the performance of two primary approaches for IDC classification which were a three-layer CNN and a variety of extracted morphological and additional visual features [4]. The CNN surpassed the feature engineering strategy [4].

Zhang et al. examined diverse techniques for large-scale medical classification [14]. Zhang et al. presented the Combined Deep and Handcrafted Visual Feature (CDHVF) algorithm to categorize medical images into 30 classes using the ImageCLEF 2016 dataset [14]. The procedure consists of four phases. The first phase is to extract deep features by fine-tuning three pre-trained classifiers [14]. The second phase is to compute multiple handcrafted descriptors, while the third phase is to apply the Principal Component Analysis (PCA) to downsize the feature space [14]. The last phase is to produce an ensemble that optimally blends the selected features to conclude the correct category [14].

Harshvardhan et al. employed various ResNet, VGG, MobileNet, DenseNet, and LeNet-5 configurations, as well as an optimized CNN architecture to solve the problem of IDC detection [15]. They reported that the best optimized CNN (C_{best}) delivered the best sensitivity, and VGG16 had the most satisfactory performance on all other metrics [15].

Zhang et al. used a Multi-Scale Residual Convolutional Neural Network (MSRCNN) as a feature extractor and implemented an SVM model to classify the deep features [16]. They first tested with a different number of MSRCNN blocks to observe the improvement in the accuracy of the test data [16]. As the number of MSRCNNs increases, the accuracy rises until they reach five blocks, after which the accuracy declines by 5% [16]. They analogized the performance of four MSRCNN blocks with and without an SVM classifier [16]. They declared that using an SVM classifier slightly enhanced the accuracy and F1-score [16].

This paper illustrates the utilization of an evolutionary algorithm to automate selecting the best Neural Network (NN) configuration with its corresponding weights during meta-training. The meta-training shrinks the hyper-parameters search space considerably, accelerating convergence during the comprehensive training stage. It furthermore displays an intuitive measure of heterogeneity and automatic optimization of the horizontally stacked prediction vector's wights.

The existing state-of-the-art transfer learning techniques for microscopy image classification lack the systematic detachment of the task-dependent layers from the transferrable ones, resulting in overfitting when training over a limited quantity of samples. Moreover, their ensembles have a high number of hyper-parameters, making it challenging and time-consuming to pick the optimal ones manually.

It was established in earlier research that ensemble learning is indispensable for improving the accuracy of the overall system. Ensemble learning aspires to enhance the performance of a meta-classifier which is composed of weak classifiers. The arrangement of the meta-classifier delivers more acceptable prediction accuracy than any of the individual models. Furthermore, it was also used for transient stability of power systems and demonstrated superior performance to other state-of-the-art methods in term of time efficiency, feature space reduction and accuracy [17].

This paper presents a novel approach of utilizing the Cartesian Genetic Programming Algorithm (CGP) for stacked ensemble optimization. Our pipeline consists of two primary parts. The first part deals with training Convolutional Neural Networks with different setups on the Breast Histopathology Imaging (BHI) dataset. In the second portion, we define a cartesian Neural Network with a number of the NN topology parameters and evolve them using the Differential Cartesian Genetic Programming. Section II describes our proposed methods. Our approach outperforms previously published algorithms as demonstrated in section IV.

The remaining sections are arranged as follows: Section II elaborates on the training of individual classifiers and the meta-training of the ensemble strategy. Section III illustrates in detail the experimental setup, the evaluation metrics, the benchmark dataset and the used computational resources. Section IV exhibits the experimentations and their results compared with diverse state-of-the-art published methodologies. Section V concludes the paper with our findings, the constraints of our approach, and future research suggestions.

II. PROPOSED METHOD

A. OVERVIEW

This section comprehensively describe the differential Cartesian Genetic Programming Artificial Neural Network (dCGPANN) and how it was employed to acquire the optimum topology and utilize the gradients for error back-propagation to update the weights and biases. It also examines the implementation details of the individual classifier's training. Besides, it depicts the multiple stages of training the ensemble.

Evolutionary Algorithms, in general, are used to optimize the parameters of a particular system or to conceive a better architecture [18]. The principal emphasis of the earlier proposed methods was to fine-tune the parameters of the ensemble strategy. In numerous algorithms, the numbers of nodes and the connections are fixed, while the ANN weights are evolved. Different algorithms only evolve the weights of the ANN using EA. Nonetheless, stochastic gradient descent is far more potent in correcting the weights in the course of training than any other technique.

For the ensemble to be more accurate than its composing classifiers, it must be diverse and accurate. Nevertheless, diverse and accurate classifiers do not always guarantee a better performing ensemble [7]. Assembling a diversity-accuracy balanced ensemble is not a straightforward process. Thus, a more advanced ensemble technique is needed. This chapter concentrates on deciding the most optimum architecture of the stacked ensemble as well as its weights and biases through the standard error back-propagation using the gradient information obtained by the dCGPANN algorithm [19], [20], [21].

The proposed method is composed of two stages. The first stage is to fine-tune and train end-to-end multiple heterogeneous classifiers on the training datasets to ensure diversity amongst models. Then we select the best-performing models. The test dataset patches are then augmented to 5000 images. The classifiers' predictions of the 5000 images are used to train the stacked ensemble using dCGPANN. The CGP inputs are an 8×1 vector of stacked prediction of the best performing models. The test dataset is used to evaluate the performance of our method against different cutting-edge algorithms.

The genes are then evolved using crossover and mutation operators based on predefined parameters as explained in II-B. The motive of this method is to overcome the limitation that the previous methods had by using the derivatives of the loss function. We believe that the reduction of software bloat and the representational capability of the dCGPANN can produce state-of-the-art competitive ensemble topologies.

One of the merits of our approach is its scalability. The accelerated technological advancements in GPUs and the increasing number of open medical datasets necessitate design scalability [22]. The accuracy increases as the number of heterogeneous weak learners in an ensemble grow. The more data and computational resources become available, the more learners can be fine-tuned and, therefore, leading to larger CGP-based Neural Network ensemble. Due to the separation between the fine-tuning and the ensemble network, the learners can be trained synchronously using parallel data approaches. Also, the learning done in a particular CGP ensemble can be further extended easily by cascading it to another CGP ensemble network. The above fact ensures that as the computational resources or the available data for histological image classification increase, the training that was already done can be re-utilized and incorporated into even more extensive ensembles. Hence, our approach permits data-parallel training to utilize large-scale computational resources through distributed computing efficiently.

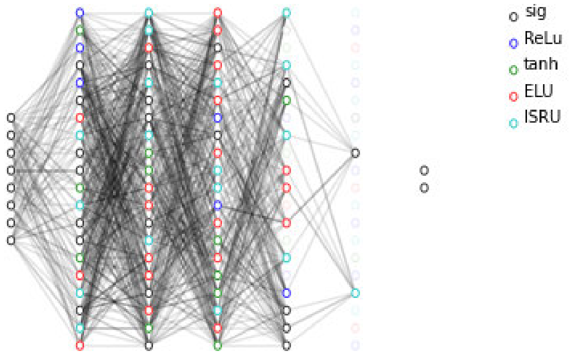


FIGURE 1. A randomly generated ensemble topology by CGP.

B. THE DIFFERENTIAL CARTESIAN GENETIC PROGRAMMING

Genetic Programming (GP) is one of the most successfully applied Evolutionary Algorithm in the optimization domain [11]. It is exceptionally suitable for binary classification problems due to its syntax [11]. One of the constraints that impedes the performance of GP is program bloat and duplicative calculation of the same node every time it is needed [23]. Another drawback is their lack of ability to evolve topologies. Since its invention, researchers have developed numerous versions of GP. Several GP algorithms have been developed such as stacking, tree-based Genetic Programming, Grammatical Evolution, linear Genetic Programming and CGP.

The Cartesian Genetic Programming (CGP) was first formed to evolve circuit design in the late 90s [24], [25], [26]. The CGP technique offers more flexibility to realize better ensemble topologies. The most prominent advantage of CGP over other variants of Genetic Programming is its ability to express the solution candidates as a cyclic graphs rather than the tree-based variant in the standard GP. CGP is known to reduce redundant computations [25]. Unlike standard tree-based GP, a solution can be represented in a Cartesian form, as shown in Figure 1. Figure 1 displays a randomly generated ensemble topology by the Cartesian Genetic Programming.

The capacity of the CGP to represent a candidate solutiona in two-dimensional gene was revolutionary. Comparable to electrical circuits for which the CGP was used to evolve, the evolution of an ANN was investigated. Figure 2 portrays the standard CGP encoding of the ANN where Miller et al. used CGP to evolve the ANN architecture [27].

Figure 2 displays how the Neural Network is encoded in standard CGP. Each node represents a non-linear activation function randomly selected from a predefined list [27]. The Cartesian network grid with $r \times c$ nodes has connections that link the nodes according to the set arity a, which is the number of inputs to each node [27]. The first column from the left of the NN is the input layer, which consists of n number of features. The last layer is the output layer. Where r is the number of the rows, c is the number of the columns, n is the number of the features, and m is the number of the classes.

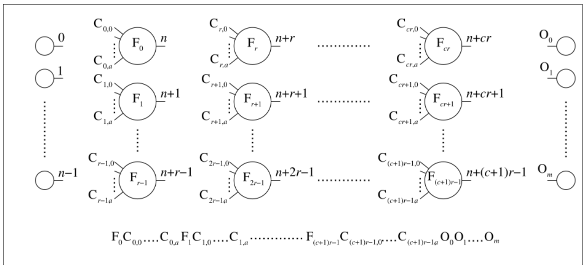


FIGURE 2. The standard CGP representation of the Neural Network and the encoded gene [27].

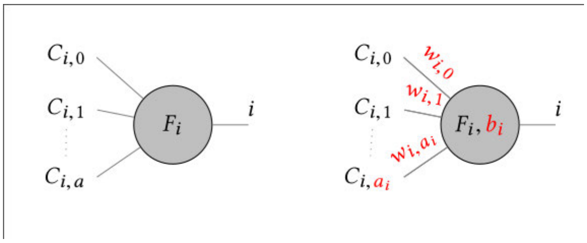


FIGURE 3. The dCGP representation of the Neural Network and the additions of weights and biases to the encoded gene [20].

The inputs of the Neural Network is the output predictions of the four models that compose the ensemble. The output of the Neural Network is the final output probabilities of the whole ensemble. It is known in the literature that the weights are better updated using the gradient descent methods which established its supremacy over the past decades [19]. Another deficiency of the original use of CGP is the absence of biases which is vital to the learning process. Izzo et al. [19], [20], [21] established a remarkably innovative idea that permits the evolution of the topology with CGP and updating the weights and biases using Stochastic Gradient Descent at the same time [20]. The concept is to split the gene into two parts. The first portion encodes the neural network nodes, connections, and activation functions. The second portion of the gene encodes the weights and the biases of the ANN connections [20]. Figure 3 pictures how the modified CGP incorporated the weights and biases.

In the dCGPANN variant, a gene X representing a candidate solution consists of two portions. X_I contains the evolutionary part of the gene which is evolved by the CGP, while X_R contains the weights and biases of the ANN as indicated in equation 1 and 2 respectively [20].

$$\begin{aligned} X_I = [&F_0, C_{0,0}, C_{0,1}, \dots, C_{0,a}, F_1, C_{1,0}, \\ &\dots, C_{1,a}, \dots, O_1, \dots, O_m] \end{aligned} \tag{1}$$

$$X_R = [b_0, w_{0,0}, w_{0,1}, \dots, w_{0,a_0}, b_1, w_{1,0}, \dots, w_{1,a_1}, \dots] \tag{2}$$

where $X_I \in$ natural number is a vector that encodes the evolutionary part of the ANN, $X_R \in$ real numbers is a vector for the biases b and weights w, F represents functions,

```
Algorithm 1 dCGPANN
Require: $r, c, n, m, l, a, Kernels, epochs, cycles$
  Randomly Initialize N dCGPANNs
  for dCGPANN in population N do
    Compute the Loss
  end for
  for j in cycles do
    Select the best dCGPANN of the previous generation
    Delete other dCGPANNs
    for i in population N do
      Mutate the best dCGPANN's functions using $\mu_a$
      Mutate the best dCGPANN's connections using $\mu_c$
      for epoch in epochs do
        Run SGD on dCGPANN$_i$
      end for
    end for
  end for
```

C represent the connections and O represent the terminal output nodes.

Based on the dCGPANN, the output of each node in the ANN is expressed in equation 3 [20].

$$N_i = F_i(\sum_{j=0}^{a_i} w_{i,j} C_{N_{i,j}} + b_i) \tag{3}$$

where N_i is the output of the node N with id i, F_i is the activation function, a_i is the arity of the node i which is the number of connections to that node, $C_{i,j}$ is the connection between note i in the current layer and node j in the previous layer and b_i is the bias associated with the activation function F_i in node N_i. The number of connections to each node, arity, is assumed equal by default in all nodes unless it gets defined by a list that specifies the number of connections each node should have in a particular layer. The arity list is a $\{1xc\}$ vector that assigns the arity of the nodes in each column c.

The evolutionary operators μ_c and μ_a are applied on the X_I part of the gene that expresses the dCGPANN. μ_c and μ_a are fractions to be mutated in active connections genes and active function genes respectively. Each mutant goes through a predefined number of stochastic gradient descent (SGD) training epochs during which only X_R is learned, and X_I is fixed. Algorithm 1 illustrates the steps of how the dCGPANN was operated to optimize the structure of the ensemble as well as its weights and biases θ. The loss function which the dCGPANN is minimizing is the categorical cross-entropy which is defined in equation 4 [28].

$$Loss = -\sum_{i=1}^{N}\sum_{j=1}^{C} y_{i,j} \times \log \hat{y_{i,j}} \tag{4}$$

where N is the number of the input images, C is the number of classes, $y_{i,j}$ is the true label of image i belongs to class j which is 1 if $X_i \in j$ and 0 otherwise and $\hat{y_{i,j}}$ is the output probability that image $i \in$ class j.

C. IMPLEMENTATION DETAILS

This section outlines the implementation details and the handpicked classifiers' and the dCGPANN's learning hyper-parameters. Table 1, 2, 3 and 4 tabulate the classifiers' hyper-parameters used for the first phase of our approach. Table 5 lists the the dCGPANN hyper-parameters used for the ensemble optimization in the second phase of the proposed method. The first stage of the training, multiple classifiers with different configurations were trained on the IDC dataset. Amongst the trained classifiers, the best four performing models were chosen to construct the ensemble. The first model was a Resnet50 pretrained on ImagNet which was trained using Transfer Learning [29], [30], [31]. We froze all layers except the last 69 layer which included the last new fully connected layer that was adopted to the new binary labels. The learning rate (LR) was high due to the fact that most of the domain-specific layers were frozen. The second model was also a Resnet50 but was trained end-to-end. The third and the fourth models were VGG19 and Densenet121 respectively whose training parameters are shown in Table 3 and Table 4 [32].

TABLE 1. Hyper-parameters for Classifier 1.

hyper-parameter	value
Name	Resnet50 [29]
Training	Transfer Learning using ImagNet weights [30]
Total params	23,595,908
trainable residual blocks	10
trainable conventional blocks	20
trainable layers	69
trainable params	18,605,572
Non-trainable params	4,990,336
Learning Rate	0.0001
LR scheduler	Cyclic LR [33]
Optimizer	AdaMax [34]
epochs	200 with earlystopping

TABLE 2. Hyper-parameters for Classifier 2.

hyper-parameter	value
Name	ResNet50
Training	end-to-end
Total params	23,595,908
trainable layers	All
trainable params	23,595,908
Learning Rate	0.0001
LR scheduler	Cyclic LR
Optimizer	SGD
epochs	400 with earlystopping

TABLE 3. Hyper-parameters for Classifier 3.

hyper-parameter	value
Name	VGG19
Pre-trained	Transfer Learning using ImagNet weights
Total params	143,667,240
trainable layers	26
trainable params	143,667,240
Non-trainable params	0
Learning Rate	1e-5
Optimizer	AdaMax
epochs	200 with earlystopping

TABLE 4. Hyper-parameters for Classifier 4.

hyper-parameter	value
Name	DenseNet121 [35]
Training	Transfer Learning using ImagNet weights
Total params	8,062,504
trainable layers	429
trainable params	7,978,856
Non-trainable params	83,648
Learning Rate	0.0001
Optimizer	AdaMax

TABLE 5. dCGP parameters.

Paramter	value
r number of rows	20
c number of columns	5
m input features	8
possible kernels	sig, ReLu, tanh, ELU, ISRU
l level-back allowed connections	1
population	7
μ_a	0.05
μ_c	0.05
number of iterations	100
number of epochs	15

The possible kernels used for the dCGPANN are the Sig, ReLu, tanh, ELU and ISRU functions which are defined by equations 5, 6, 7, 8 and 9 respectively [36], [37].

$$Sig(x) = \frac{1}{1+\exp(-x)} \tag{5}$$

$$ReLu(x) = max(0, x) = \begin{cases} x_i & if\ x_i \geq 0 \\ 0 & if\ x_i < 0 \end{cases} \tag{6}$$

$$tanh(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)} \tag{7}$$

$$ELU(x) = \begin{cases} x & if\ x > 0 \\ \alpha \exp(x) - 1 & if\ x_i \leq 0 \end{cases} \tag{8}$$

$$ISRU(x) = x\frac{1}{\sqrt{(1+\alpha x^2)}} \tag{9}$$

Setting up the parameters for the dCGPANN is a challenging task and demands a trial-and-error approach. Nonetheless, the literature on Cartesian Genetic Programming provides a general direction for parameter selection. CGP Research has shown that setting the mutation to 1% for each 100 nodes is recommended [27]. However, setting the mutation to half of the recommended value during experimentation has produced more competitive topologies. Multiple experiments have also shown that setting r and c to 20 and 5 provided the best designs for this particular ensemble. The number of iterations and epochs were set to 100 and 150 to limit the time required to train the models and to avoid over-training the ensemble, which might result in overfitting.

III. EXPERIMENTAL SETUP

A. DATASET AND THE EXPERIMENTAL SETUP

The dataset contains patches extracted from the whole slides of Two hundred seventy-nine patients were diagnosed with IDC. The total number of the extracted non-overlapping of 50 × 50 pixels patches is 277524, which are labeled into IDC and non-IDC regions. The number of patches and the percentage of IDC annotations per patient vary significantly; thus, the labels of the images are not equal. Fig 4 shows the patches and IDC annotation percentage histogram. Some visual features distinguish IDC from non-IDC patches, such as tissue coloration as shown in Figure 5 and figure 6, which shows negative and positive IDC, respectively. Figure 7 shows the color maps of annotated whole slides where the yellow regions indicate the presence of IDC, and figure 8 shows a whole slide that was reconstructed using the coordinates information provided by the dataset. The darker mask points to the IDC regions on the whole slide image (WSI) [38]. Data visualization of the training image set is shown in figures 4, 5, 6, 7 and 8 [38].

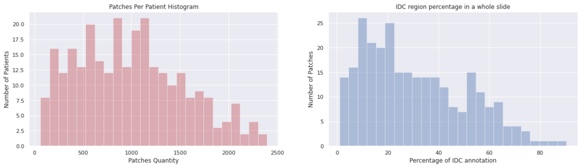


FIGURE 4. Dataset Statistics [38].

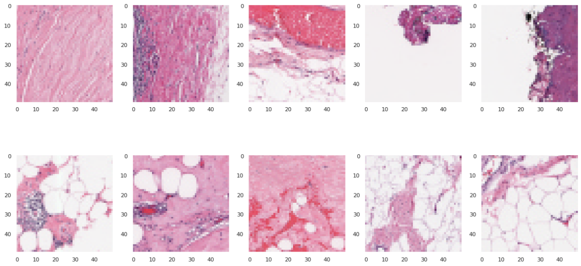


FIGURE 5. IDC negative samples [38].

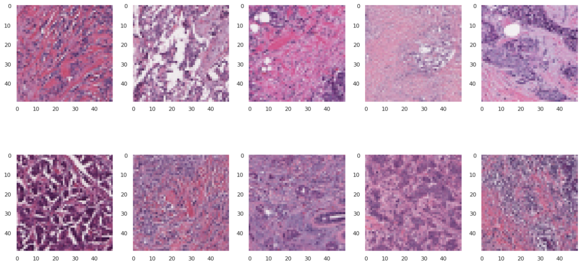


FIGURE 6. IDC positive samples [38].

The dataset was divided into three sets. 70% of the patients' slides were used as a training dataset, and the remaining patients' slides were divided into a validation dataset and a test dataset with 15% of the slides each. Our model and the base models to which our method was compared to were implemented using PyTorch, dcgpy and scikit-learn [21], [39], [40]. Our method was compared to two voting strategies which are the maximum and the weighted average.

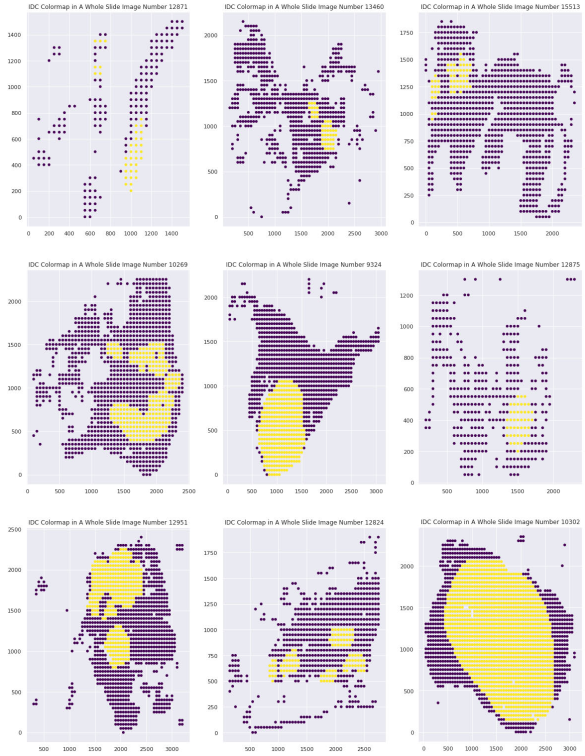


FIGURE 7. IDC Colormap in a WSI [38].

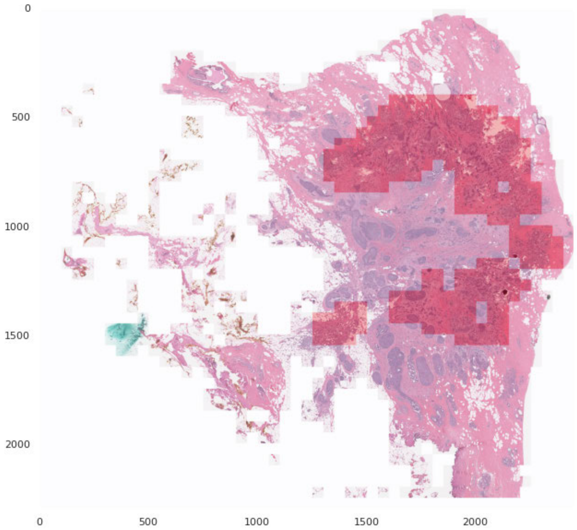


FIGURE 8. Reconstructed Annotated WSI [38].

B. EVALUATION METRICS

The evaluation metrics used for our assessment are the F1-score, the Balanced Accuracy and the overall accuracy as defined in equations 10, 15 and 13 respectively [41], [42].

$$F_1 = \frac{2}{\frac{1}{Recall} + \frac{1}{Precision}} = \frac{2Precision \times Recall}{Precision + Recall} \tag{10}$$

where *Recall* and *Precesion* are defined in 11 and 12 [41].

$$Recall = Sensitivity = \frac{TP}{TP + FP} \tag{11}$$

Note that *TP* is number of correctly positively classified samples and *FP* is the incorrectly positively classified ones.

$$Precision = \frac{TP}{TP + FN} \tag{12}$$

$$BAC = \frac{Sensitivity + Specificity}{2} \tag{13}$$

where Specivity is defined by equation 14.

$$Specivity = \frac{TN}{TN + FP} \tag{14}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{15}$$

- Macro-average: is the sum of each class' precision divided by the number of classes.
- Micro-average: is the sum of the true positives of all classes divided by the number of samples.

IV. RESULTS

This section shows the results of our experiments. The performance of phase 1 trained classifiers as well as the base models on the validation set are shown in Table 6.

TABLE 6. Confusion matrices of various models on the validation set.

classifier 1	Predicted IDC	Predicted Non-IDC
actual IDC	0.904239	0.095761
actual Non-IDC	0.240137	0.759863
classifier 2	Predicted IDC	Predicted Non-IDC
actual IDC	0.920932	0.079068
actual Non-IDC	0.257888	0.742112
classifier 3	Predicted IDC	Predicted Non-IDC
actual IDC	0.931916	0.068084
actual Non-IDC	0.320158	0.679842
classifier 4	Predicted IDC	Predicted Non-IDC
actual IDC	0.944721	0.055279
actual Non-IDC	0.148827	0.851173
Average Voting	Predicted IDC	Predicted Non-IDC
actual IDC	0.953242	0.046758
actual Non-IDC	0.165286	0.834713
Maximum Voting	Predicted IDC	Predicted Non-IDC
actual IDC	0.931916	0.068084
actual Non-IDC	0.320158	0.679842
dCGPANN	Predicted IDC	Predicted Non-IDC
actual IDC	0.963582	0.036418
actual Non-IDC	0.090169	0.909831

Our experiments were conducted using an AMD Ryzen Threadripper 1950X 16-Core Processor, which operates on two threads per core and 32 Numa nodes, and an NVIDIA GeForce RTX 2080 with 8GB of video RAM. The maximum and average voting ensembles didn't demand training. The evolution of the DGP network was significantly time-efficient compared to the time cost of training the weak classifiers. The training time of each architecture was 1.2450 seconds. Each iteration required 8.71533 seconds and the overall training of the ensemble took 880.24 seconds.

TABLE 7. Performance of the weighted voting on the test dataset.

	precision	recall	f1-score
actual no cancer	0.84	0.93	0.89
actual cancer	0.84	0.68	0.75
macro avg	0.84	0.81	0.82
weighted avg	0.84	0.84	0.84

TABLE 8. Performance of the Maximum voting on the test dataset.

	precision	recall	f1-score
actual no cancer	0.84	0.93	0.89
actual cancer	0.84	0.68	0.75
macro avg	0.84	0.81	0.82
weighted avg	0.84	0.84	0.84

TABLE 9. Performance of dCGPANN voting on the test dataset.

	precision	recall	f1-score
actual no cancer	0.96	0.94	0.95
actual cancer	0.93	0.88	0.93
macro avg	0.95	0.91	0.92
weighted avg	0.95	0.96	0.95

TABLE 10. Comparison of the proposed method with other published methods in the standard evaluation metrics.

Method	Accuracy	Balanced accuracy	F1-score
Proposed Method	**0.92874**	**0.93671**	**0.9049**
Avg. Voting	0.87641	0.89398	0.84442
Max. Voting	0.87439	0.89422	0.84337
VGG16 in [15]	–	0.8306	0.7190
VGG19 in [15]	–	0.7949	0.6702
ResNet50 in [15]	–	0.7175	0.5905
MobileNetV2 in [15]	–	0.7679	0.6560
DenseNet121 in [15]	–	0.7967	0.6806
ResNet50V2 in [15]	–	0.7234	0.6002
LeNet-5 in [15]	–	0.8162	0.7136
CNN_{best} in [15]	–	0.7869	0.7006
Cruz-Rao et al. [4]	–	0.8423	0.7180
janowczyk et al. [43]	–	0.8468	0.7648
MSRCNN-SOFTMAX [16]	0.8669	0.8464	0.7885
MSRCNN-SVM [16]	0.8745	0.8570	0.7989

Our technique was compared to the average voting and the Maximum voting schemes. The outcomes on the held-out test dataset are reported in Table 7, 8, and 9. The results demonstrate a clear advantage of the dCGPANN ensemble over all other voting strategies, as shown in Table 9.

To the best of our knowledge, the dCGPANN approach has never been implemented to optimize ensemble ANN for histology image classification. The findings of our experiments were compared with the state-of-the-art methods that were proposed in [4], [15], and [16] in Table 10. The number of test samples used to evaluate our proposed method was 38183, which resembles the extracted 50 × 50 patches from the slides of 42 randomly selected patients. The number of IDC test samples was 13434, while the number of No IDC test samples was 24749. As explained earlier, the classes of the dataset are unbalanced. There are far more negative patches than positive ones. Table 10 demonstrate the significance of the improvement of our method compared to other competitive models reported in the literature. Our Method yielded superior results in accuracy, balanced accuracy and F1 score by a significant margin. The accuracy of our technique exceeded the best performing method by 5% percent. More importantly, the BAC and F1-score improved by 4.3% and 5.6% respectively. Moreover, the ensemble yielded much higher improvement in the confusion matrix to the weak classifiers as shown in Table 6 and 9. Tables 10 and 11 provide a summary of the performance of our proposed method compared to the state-of-the-art methods in terms of the most reported evaluation metrics in this field.

TABLE 11. Comparison of the proposed method with other published methods in Recall Precesion, and Specificity.

Method	$\text{Re}/\text{S}_{\text{n}}$	Pr	Sp
Proposed Method	**0.9636**	**0.85294**	**0.90981**
Avg. Voting	0.95325	0.75789	0.8347
Max. Voting	0.96114	0.75131	0.82731
VGG16	0.5806	0.9256	0.9618
VGG19	0.5315	0.9067	0.9486
ResNet50	0.5001	0.7207	0.8658
MobileNetV2	0.5880	0.7418	0.8858
DenseNet121	0.5670	0.8511	0.9263
ResNet50V2	0.5444	0.6687	0.8556
LeNet-5	0.6276	0.8271	0.9215
CNN_{best}	0.7338	0.6702	0.8736

V. CONCLUSION

The accuracy of the Deep Learning models is a cornerstone to the CAD systems. It is evident that the advancements of Deep Learning and GPUs enabled computers to perform better or at least equal to human experts. Thus, computerized cancer detection can at the very least increase the speed of diagnosis. Our proposed method addresses the scarcity of labeled data by implementing a bio-inspired algorithm to construct a more reliable ensemble. The ensemble learns the correct class based on the misclassifications of the pre-trained models. The differential Cartesian Genetic Programming was used to acquire the most optimum ensemble rule to compensate for the insufficient quantity of images available for multi-stage training, which is required to infer accurate mapping between intricate features and correct classes. Due to its cabability to assign weights and biases to ANNs and to use SGD for learning them, the dCGPANN proves to be a powerful tool to optimize the ensemble topology as well as its hyperparameters.

We demonstrated the accuracy and time efficiency of our proposed method to classify invasive ductal carcinoma using the breast histology image dataset. A statistical evaluation of our results was also provided on the benchmark dataset to further demonstrate the applicability of our methods. The experimental findings exceeded previous hyperparameters search methods. The evidence from this study implies that dCGPANN ensemble produce better results than any individual weak classifier as well as other recently published ensemble methods.

Based on the results reported, we recommend future research to be focused on the application of CGP on optimizing CNNs. The viability of designing GCP-based ensemble networks using Convolutional kernels that combine features extracted by fine-tuned pre-trained learners should be investigated.

ACKNOWLEDGMENT

The authors highly appreciate the time and effort devoted by the reviewers to review their study. They are deeply indebted for their comprehensive review's insights and different perspectives.

REFERENCES

[1] E. Alkhaldi and E. Salari, "Adaptive PSO-based ensemble optimization for histology image classification," *Int. J. Comput. Sci. Technol.*, vol. 12, no. 1, pp. 26–34, 2021.

[2] E. Alkhaldi, "Genetically optimized heterogeneous ensemble for histological image classification," *Int. J. Sci. Eng. Invest.*, vol. 8, no. 95, pp. 113–118, 2019.

[3] H. Cao, S. Bernard, L. Heutte, and R. Sabourin, "Improve the performance of transfer learning without fine-tuning using dissimilarity-based multi-view learning for breast cancer histology images," in *Proc. Int. Conf. Image Anal. Recognit.*, in Lecture Notes in Computer Science: Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics, vol. 10882, 2018, pp. 779–787.

[4] A. Cruz-Roa, A. Basavanhally, F. González, H. Gilmore, M. Feldman, S. Ganesan, N. Shih, J. Tomaszewski, and A. Madabhushi, "Automatic detection of invasive ductal carcinoma in whole slide images with convolutional neural networks," *Proc. SPIE*, vol. 9041, Mar. 2014, Art. no. 904103.

[5] M. Frank, D. Drikakis, and V. Charissis, "Machine-learning methods for computational science and engineering," *Computation*, vol. 8, no. 1, p. 15, Mar. 2020.

[6] Y. Hou, S. Jia, X. Lun, Z. Hao, Y. Shi, Y. Li, R. Zeng, and J. Lv, "GCNs-Net: A graph convolutional neural network approach for decoding time-resolved EEG motor imagery signals," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Sep. 13, 2022, doi: 10.1109/TNNLS.2022.3202569.

[7] G.-R. You, Y.-R. Shiue, W.-C. Yeh, X.-L. Chen, and C.-M. Chen, "A weighted ensemble learning algorithm based on diversity using a novel particle swarm optimization approach," *Algorithms*, vol. 13, no. 10, p. 255, Oct. 2020.

[8] G. Huang, Y. Li, G. Pleiss, Z. Liu, J. E. Hopcroft, and K. Q. Weinberger, "Snapshot ensembles: Train 1, get M for free," in *Proc. 5th Int. Conf. Learn. Represent.*, 2017, pp. 1–14.

[9] T. G. Dietterich, "Ensemble methods in machine learning," in *Proc. Int. Workshop Multiple Classifier Syst.*, in Lecture Notes in Computer Science: Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics, vol. 1857, 2000, pp. 1–15.

[10] J. Xie, B. Xu, and Z. Chuang, "Horizontal and vertical ensemble with deep representation for classification," 2013, *arXiv:1306.2759*.

[11] K.-H. Liu, M. Tong, S.-T. Xie, and V. T. Y. Ng, "Genetic programming based ensemble system for microarray data classification," *Comput. Math. Methods Med.*, vol. 2015, pp. 1–11, Jan. 2015.

[12] M.-F. Leung, S.-C. Ng, C.-C. Cheung, and A. K. Lui, "A new algorithm based on PSO for multi-objective optimization," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, May 2015, pp. 3156–3162.

[13] Q. Yang, J.-Q. Yan, X.-D. Gao, D.-D. Xu, Z.-Y. Lu, and J. Zhang, "Random neighbor elite guided differential evolution for global numerical optimization," *Inf. Sci.*, vol. 607, pp. 1408–1438, Aug. 2022.

[14] J. Zhang, Y. Xia, Y. Xie, M. Fulham, and D. D. Feng, "Classification of medical images in the biomedical literature by jointly using deep and handcrafted visual features," *IEEE J. Biomed. Health Inform.*, vol. 22, no. 5, pp. 1521–1530, Sep. 2018.

[15] G. M. Harshvardhan, A. Sahu, M. K. Gourisaria, P. K. Singh, W.-C. Hong, V. Singh, and B. K. Balabantaray, "On the dynamics and feasibility of transferred inference for diagnosis of invasive ductal carcinoma: A perspective," *IEEE Access*, vol. 10, pp. 30870–30889, 2022.

[16] J. Zhang, X. Guo, B. Wang, and W. Cui, "Automatic detection of invasive ductal carcinoma based on the fusion of multi-scale residual convolutional neural network and SVM," *IEEE Access*, vol. 9, pp. 40308–40317, 2021.

[17] Y. Li and Z. Yang, "Application of EOS-ELM with binary Jaya-based feature selection to real-time transient stability assessment using PMU data," *IEEE Access*, vol. 5, pp. 23092–23101, 2017.

[18] L. Sekanina, "Evolutionary algorithms in approximate computing: A survey," 2021, *arXiv:2108.07000*.

[19] D. Izzo, F. Biscani, and A. Mereta, "Differentiable genetic programming," in *Proc. Eur. Conf. Genetic Program.*, in Lecture Notes in Computer Science: Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics, vol. 10196, 2017, pp. 35–51.

[20] M. Märtens and D. Izzo, "Neural network architecture search with differentiable Cartesian genetic programming for regression," in *Proc. Genetic Evol. Comput. Conf. Companion*, Jul. 2019, pp. 181–182.

[21] D. Izzo and F. Biscani, "DCGP: Differentiable Cartesian genetic programming made easy," *J. Open Source Softw.*, vol. 5, no. 51, p. 2290, Jul. 2020.

[22] K. Osawa, Y. Tsuji, Y. Ueno, A. Naruse, C.-S. Foo, and R. Yokota, "Scalable and practical natural gradient for large-scale deep learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 1, pp. 404–415, Jan. 2022.

[23] A. J. Turner, "Evolving artificial neural networks using Cartesian genetic programming," Ph.D. thesis, Dept. Electron., Univ. York, Heslington, U.K., 2015.

[24] J. F. Miller, "Cartesian genetic programming: Its status and future," *Genetic Program. Evolvable Mach.*, vol. 21, nos. 1–2, pp. 129–168, Jun. 2020.

[25] A. J. Turner and J. F. Miller, "Recurrent Cartesian genetic programming of artificial neural networks," *Genetic Program. Evolvable Mach.*, vol. 18, no. 2, pp. 185–212, Jun. 2017.

[26] A. J. Turner and J. F. Miller, "Introducing a cross platform open source Cartesian genetic programming library," *Genetic Program. Evolvable Mach.*, vol. 16, no. 1, pp. 83–91, Mar. 2015.

[27] J. F. Miller, *Cartesian Genetic Programming*, vol. 43. Berlin, Germany: Springer, 2011.

[28] J. Patterson and A. Gibson, *Deep Learning: A Practitioner's Approach*. Sebastopol, CA, USA: O'Reilly Media, Inc., 2017.

[29] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[30] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

[31] M. Long, Y. Cao, J. Wang, and M. I. J. Ordan, "Learning transferable features with deep adaptation networks," in *Proc. 32nd Int. Conf. Mach. Learn. (ICML)*, vol. 1, 2015, pp. 97–105.

[32] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8697–8710.

[33] L. N. Smith, "Cyclical learning rates for training neural networks," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2017, pp. 464–472.

[34] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Represent.*, 2015, pp. 1–15.

[35] Y. Zhu and S. Newsam, "DenseNet for dense flow," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2017, pp. 790–794.

[36] C. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, "Activation functions: Comparison of trends in practice and research for deep learning," 2018, *arXiv:1811.03378*.

[37] B. Carlile, G. Delamarter, P. Kinney, A. Marti, and B. Whitney, "Improving deep learning by inverse square root linear units (ISRLUs)," 2017, *arXiv:1710.09967*.

[38] L. FINK. *Breast Cancer*. Accessed: Nov. 30, 2021. [Online]. Available: https://www.kaggle.com/code/allunia/breast-cancer

[39] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, and L. Antiga, "PyTorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 1–15.

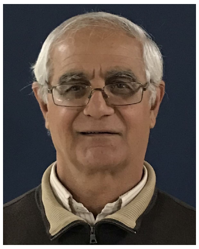
[40] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, A. Müller, J. Nothman, G. Louppe, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Oct. 2012.

[41] E. Duchesnay, T. Löfstedt, and F. Younes, *Statistics and Machine Learning in Python. Release 0.1*. Berlin, Germany: Springer, 2018.

[42] A. Javeed, S. Zhou, L. Yongjian, I. Qasim, A. Noor, and R. Nour, "An intelligent learning system based on random search algorithm and optimized random forest model for improved heart disease detection," *IEEE Access*, vol. 7, pp. 180235–180243, 2019.

[43] A. Janowczyk and A. Madabhushi, "Deep learning for digital pathology image analysis: A comprehensive tutorial with selected use cases," *J. Pathol. Inform.*, vol. 7, p. 29, Jul. 2016.

EID ALKHALDI (Member, IEEE) received the B.S. degree in electrical engineering from Oklahoma State University, Stillwater, OK, USA, in 2014, the M.S. degree in electrical engineering from The University of Toledo, Toledo, OH, USA, in 2016, where he is currently pursuing the Ph.D. degree with the Electrical Engineering and Computer Science Department. Currently, he is engaged in CNN ensemble optimization for small histological images classification using bio-inspired heuristic techniques. His research interests include digital image processing, deep learning, artificial intelligence, and anomaly detection in medical images.



EZZATOLLAH SALARI received the M.S. and Ph.D. degrees in electrical engineering from Wayne State University, in 1978 and 1982, respectively. He is a Professor with the Department of Electrical Engineering and Computer Science, The University of Toledo, where he is involved in teaching and research in the areas of image analysis and computer vision, data compression for multimedia communication, neural networks, and signal processing. He has contributed extensively in several areas of image processing and neural networks, including image and video compression, motion analysis, image representation, object recognition, and application of neural networks to image enhancement and restoration. He also served as the Graduate Director of the EECS Department, from 2000 to 2005. He was a Research Fellow at NASA Langley, Goddard and Lewis (Glenn) Research Centers during the summers of 1987, 1988, 1990, and 1991 on various image processing projects.