IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 30 December 2022, accepted 23 January 2023, date of publication 25 January 2023, date of current version 30 January 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3239816

RESEARCH ARTICLE

Varied Image Data Augmentation Methods for Building Ensemble

RICCARDO BRAVIN[1], LORIS NANNI[2], ANDREA LOREGGIA[3], SHERYL BRAHNAM[4], AND MICHELANGELO PACI[5]

[1]Department of Electronics, Information and Bioengineering, Politecnico di Milano, 20133 Milan, Italy
[2]Department of Information Engineering (DEI), University of Padua, 35122 Padua, Italy
[3]Department of Information Engineering (DII), University of Brescia, 25121 Brescia, Italy
[4]Information Technology and Cybersecurity, Missouri State University, Springfield, MO 65804, USA
[5]BioMediTech, Faculty of Medicine and Health Technology, Tampere University, 33520 Tampere, Finland

Corresponding author: Loris Nanni (loris.nanni@unipd.it)

This work was supported by NVIDIA through the Graphics Processing Unit (GPU) Grant Program.

ABSTRACT Convolutional Neural Networks (CNNs) are used in many domains but the requirement of large datasets for robust training sessions and no overfitting makes them hard to apply in medical fields and similar fields. However, when large quantities of samples cannot be easily collected, various methods can still be applied to stem the problem depending on the sample type. Data augmentation, rather than other methods, has recently been under the spotlight mostly because of the simplicity and effectiveness of some of the more adopted methods. The research question addressed in this work is whether data augmentation techniques can help in developing robust and efficient machine learning systems to be used in different domains for classification purposes. To do that, we introduce new image augmentation techniques that make use of different methods like Fourier Transform (FT), Discrete Cosine Transform (DCT), Radon Transform (RT), Hilbert Transform (HT), Singular Value Decomposition (SVD), Local Laplacian Filters (LLF) and Hampel filter (HF). We define different ensemble methods by combining various classical data augmentation methods with the newer ones presented here. We performed an extensive empirical evaluation on 15 different datasets to validate our proposal. The obtained results show that the newly proposed data augmentation methods can be very effective even when used alone. The ensembles trained with different augmentations methods can outperform some of the best approaches reported in the literature as well as compete with state-of-the-art custom methods. All resources are available at https://github.com/LorisNanni.

INDEX TERMS Convolutional neural networks, data augmentation, ensemble.

I. INTRODUCTION

Convolutional Neural Networks (CNNs) and their derivations are a hot topic of research as these deep learners can be said to be at the top of the range for image classification tasks. Leveraging the mathematical concept of convolution, CNNs learn kernels that can extract salient features directly from the training sets, without the need for human intervention or otherwise necessary feature extraction algorithms. These learners are thus able to reach and surpass the efficacy and efficiency of handcrafted features thanks to their ability to perceive relationships in bigger pixel clusters and extract features independently from their position by reducing the size of the input and expanding its depth while shaping it to the final output size. In general, CNNs need vast, labeled datasets to achieve acceptable results in classification problems due to their colossal parameter size, and for that human intervention is needed. However, it is impossible to label manually the massive number of images needed to train CNNs (on the order of 14 million images with a thousand classes in the case of ImageNet [15]). In some cases, mostly in medical and bioinformatic fields, where the number of obtainable samples is limited by external factors, collecting enough data for CNNs training can be prohibitive due to cost, knowledge needed, and labor. Where the datasets are instead enormous researchers are required to gain access to powerful and costly machines that can handle the workload. Solutions to the problem of collecting big datasets have been

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

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

used for a long time and are still being researched. The two most powerful techniques are based on transfer learning and data augmentation. The first makes use of CNN architectures pre-trained on enormous datasets, like the ImageNet [15], and then finetuned on smaller datasets. The second generates new samples based on the original ones to add to the training set. Other newer methods include dropout layers [60], zero-shot or one-shot learning [50], [70] and batch normalization [60].

Data augmentation is adopted in different situations to overcome the scarcity of images and help during the training phase of a model. For example, this is done to augment the information in low-light environments, [32] like in underwater images where it is necessary to maintain important details while at the same time enhancing the quality of the image [31], [73], [74].

Here we want to set focus mostly on data augmentation due to its vital presence in specific fields where big datasets cannot be created [27], [49], [61]. Data augmentation manages through the enlargement of the dataset to promote better generalization and reduce the problem of overfitting by adding and extracting information that is inherited within the training space. Most of the literature, in this regard, (see surveys [27], [49], [61]) covers geometric transforms, color modification methods through statistical probability and learned methods in the likings of Generative Adversarial Networks. Here, we analyze and evaluate the performances of ensembles built on the data level through combination by sum rule of different image manipulation methods like the ones presented in [47].

The remaining of the paper is structured as follows: Section II provides some related works on data augmentation via image manipulation. Section III describes the new data augmentation approaches. Section IV describes the empirical evaluation with the comparison of all deep learning models trained with the image augmentation methods. Section V concludes this work and provides some further research on this topic.

The contributions of this article are emphasized as follows:

i) We propose a set of new methods for data augmentation based on different image transformations. These new methods enjoys fast processing speed and are beneficial for different computer vision tasks.

ii) We define different ensemble methods that combine different data augmentation methods during the training phase. Compared with existing machine learning methods, our ensembles based on data augmentation methods provide competitive results in different domains.

iii) We provide an empirical evaluation of ensembles trained with classical and the new data augmentation methods. Assessing on several metrics and datasets, we show that making use of different and independent image augmentation methods is beneficial for ensembles.

II. RELATED WORK

A high-level taxonomy for these methods is depicted in Figure 1 [61]. The research study proposed in this work is focused on data augmentation methods computed by performing basic image manipulation.

Most of these augmentation algorithms are easy to implement, thus providing feasible ways for scholars to adopt them. But practitioners must be careful in applying these image manipulations because it is possible to produce new images that no longer belong to the same class as the original. For instance, flipping an image that represents the digit “6” would result in an image that represents the digit “9.”

Among the different geometric transformations, flipping, rotation, and translation are the most popular ones. Flipping, especially along the horizontal axis, is one of the simplest and most popular geometric transforms for data augmentation [61]. Rotating an image on the right or left axis in the range [1, 359] is another typical geometric transformation as it is a translation that shifts a sample up, down, left, and right [61]. The latter can introduce undesirable noise [38]. Another technique consists of the random cropping of an image that results in a new image with reduced size which is often required to fit the input of a model. New images can be generated also by substituting random values in the original input. This technique has been extensively evaluated in the literature [40], for instance, comparing the performance of different AlexNets trained with these simple augmentation techniques and then assessing the performance on two different datasets [59]. On the standard ImageNet and CIFAR10 [28], models trained with data augmentation via rotation of images provide better performance if compared with data augmentation via translation, random cropping, or random values.

Random erasing [76] and cutting [17] result in new images with occlusions; this is particularly useful as these methods represent standard situations in the real world where objects are often partially visible. A recent literature review of data augmentation methods reports the literature on this type of method [49]. Good performance has been provided by a ResNet architecture trained on Fashion-MNIST, CIFAR10, and CIFAR100 where new images were generated by erasing portions that vary in size of the original inputs [76].

Mixing images is another simple method for generating new images. This can be easily done by averaging the pixels between two or more images from the same classes [25], or images can be transformed by adopting a transform whose outputs can be mixed, for instance, by chaining [23]. Data augmentation can be accomplished also by combining different techniques. For instance, in [25], random images were cropped and flipped before averaging the RGB channel values of each pixel in the images. Recently, non-linear transformations were adopted to mix images [33] or by adopting generative adversarial networks (GAN) [33].

Kernel filters are often used to blur or sharpen images with the aim of generating new images, for instance, by applying Gaussian blur with a sliding window of fixed size or by randomly swapping the values of the matrix in the filter window, as done by PatchShuffle [26].

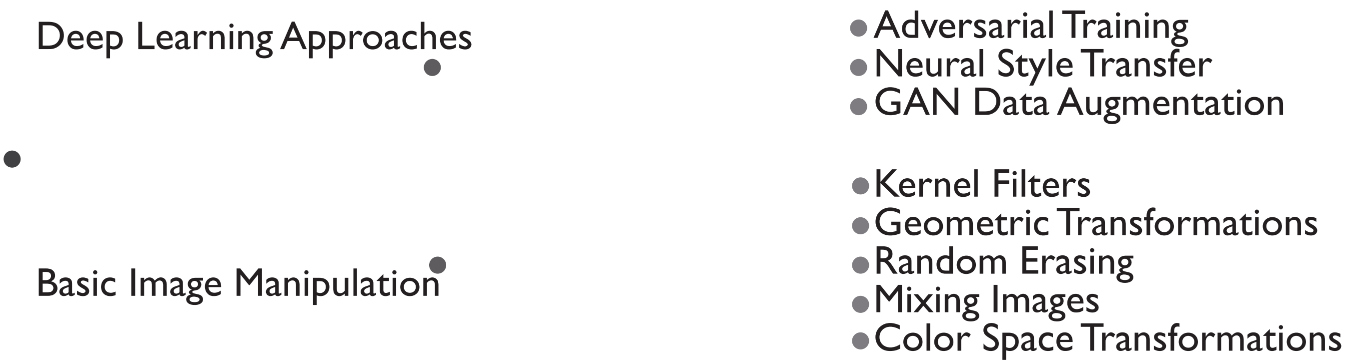


FIGURE 1. The high-level taxonomy for data augmentation methods introduced in [61].

The generation of new color spaces is another method adopted for data augmentation, with the positive side-effect that possible existing biases of illumination are removed [61]. It is also possible to compute a histogram of pixels in a color channel with the aim of applying different filters to each channel or converting one color space into another. But in some situations, for instance, changing RGB to grayscale, this operation can result in reduced performance of a classifier [9]. Data augmentation is often produced with the addition of random noise to color distributions or by jittering and adjusting the contrast, saturation, and brightness of the original images [29], [59]. These color adjustments may result in the loss of valuable information. We point the reader to [65] for an exhaustive review of color space transforms for image augmentation.

Sometimes, data augmentation techniques do not consider the entire training set. An example of that is PCA jittering [29], [41], [42], [59], [65] which performs data augmentation by multiplying the PCA components by a small number. In particular, only the first component that is the most informative is jittered [65], an image can be transformed by applying PCA, DCT, and jittered by adding noise to all components before reconstructing the image [42].

III. MATERIALS AND METHODS

In this section, we describe the data augmentation methods adopted from the literature and we introduce the new methods proposed in this study.

A. DATA AUGMENTATION MODELS

The number of images produced by each method used in this study is reported in Table 1.

1) OLD DATA AUGMENTATION METHODS

Old data augmentation methods are drawn from the literature. In particular: the methods labeled APP1 to APP11 have been detailed in [47], while APP12 to APP14 are proposed in [48]. For technical details about these methods, we point the interested reader to the original papers.

TABLE 1. Number of images generated by each data augmentation method. K stands for kernel filters, G for geometric trans., R for random erasing, M for mixing images, C for color space trans.

Method	Type	#New Images	Ref.
APP-1	G	3	[47]
APP-2	G	6	[47]
APP-3	G	4	[47]
APP-4	K	3	[47]
APP-5	K	3	[47]
APP-6	C	3	[47]
APP-7	C+K	7	[47]
APP-8	C+K	2	[47]
APP-9	G+K	6	[47]
APP-10	K	3	[47]
APP-11	K	3	[47]
APP-12	K+M	5	[48]
APP-13	K	3	[48]
APP-14	K	2	[48]
APP-15	G+C+K	11	NEW
APP-16	G+K	2	NEW
APP-17	K+G	4	NEW
APP-18	K+G	6	NEW
APP-19	K+G	13	NEW
APP-20	M+K	11	NEW

APP1 produces 3 new images by geometric transformation. Starting from a given image, one image is generated by randomly reflecting the input from top to bottom and another one from left to right. The third transformation linearly scales the original image along both axes.

APP2 generates 6 new images by repeating APP1's operations with three additional ones: rotation, translation, and shearing.

APP3 generates 4 new images by replicating APP2 process without shear.

APP4 generates 3 new images by applying PCA-based transforms.

APP5 generates 3 new images by applying DCT-based transforms similar to the ones adopted in APP4.

APP6 generates 3 new images by altering the color space. The three images are constructed by altering contrast, sharpness, and color shifting.

APP7 generates 7 new images by altering the color space. The first four augmented images are produced by altering the pixel colors in the original image. Two images are generated

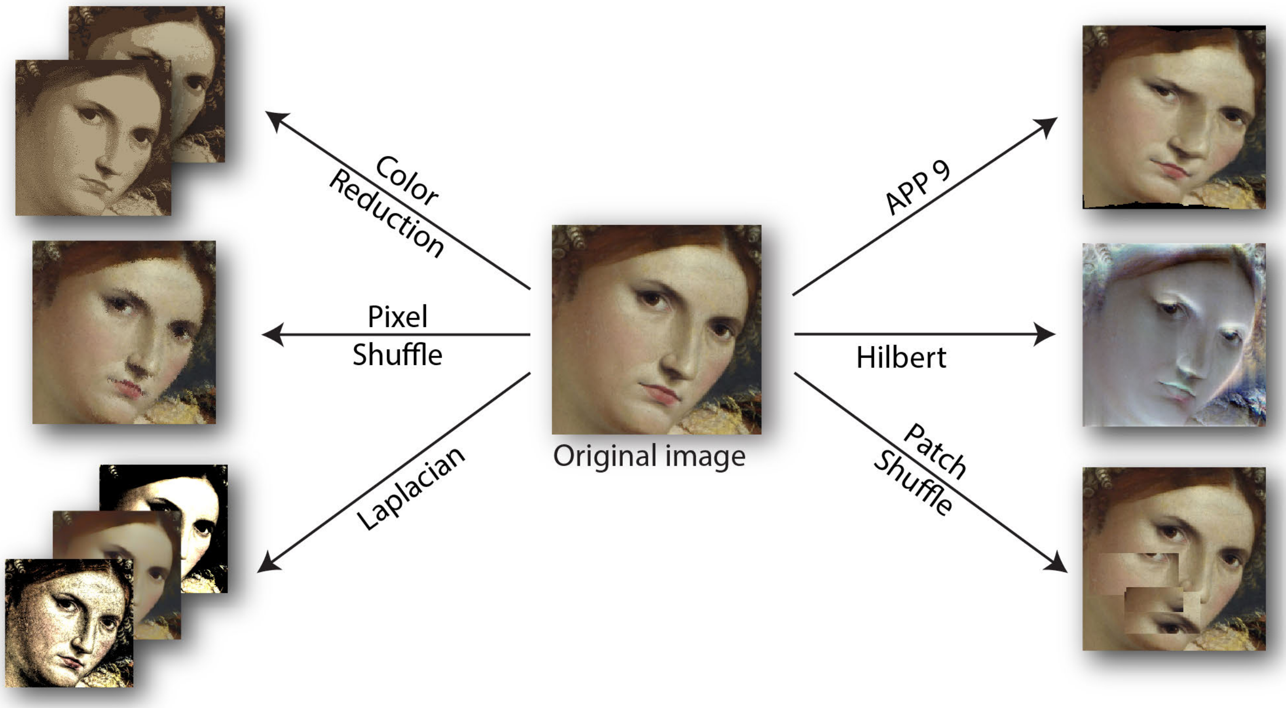


FIGURE 2. Examples of an original image (center) augmented on the simpler methods used in APP15 through APP20.

by combining sharpening and a Gaussian filter. One image is generated by color-shifting.

APP8 generates 2 new images by altering the color space followed by the application of two nonlinear mappings.

APP9 generates 6 new images by applying elastic deformations combined with low-pass filters.

APP10 generates 3 new images by perturbing the matrices resulting from DWT (i.e., Daubechies wavelet db1 with one vanishing moment).

APP11 generates 3 new images resulting from the Constant-Q Transform (CQT) [68].

APP12 generates 5 new images by combining DCT and the random selection of other images.

APP13 generates 3 new images by applying the Radon Transform (RT) in a different way.

APP14 generates 2 new images by applying the Fast Fourier Transform (FFT) and DCT.

2) NEW DATA AUGMENTATION METHODS

Five new approaches are proposed here. Notice that most of the methods which make use of direct transformations are grouped and depicted in Figure 3, while, examples of more complex data augmentation for each of the new methods are depicted in Figure 2.

APP15 produces 11 new images. The first image is generated through the application of two consecutive DCT transforms to each matrix composing the three color planes, after which haze reduction and histogram equalization algorithms are applied for better readability (see Listing 1). The goal is not to extract a specific type of information, but rather to use the DCT properties to modify the image. The inverse DCT is closely related to the forward DCT. Thus, taking two consecutive forward DCTs results in an image that appears similar to, but not the same as, the original. Therefore, we exploit this DCT property to create new images starting from the original ones. The second image, in a similar way, applies the FFT two times, but between the two also zeroes out all positions of the matrix where the modulo of the value is higher than the average standard deviation of the columns (see Listing 2). Another image is generated through FFT transform by averaging the phases of the obtained matrix with that of another random image from the same dataset; to the result is then applied the inverse FFT (see Listing 5). The fourth image is obtained by applying Singular Value Decomposition (SVD) with the removal from the diagonal matrix of all values lower than the max divided by a random integer in the range [50,100]; the three obtained matrices thus modified are then multiplied to get the final image (see Listing 3). The successive three images make use of Local Laplacian Filtering for a random enhancement of small details contrast, a random smoothing of small details, and an overall increase of dynamic range and contrast. Another image is obtained through a technique that, using a transformation called color indexing, which reduces the number of colors available to represent the image to a random number in the range [8,16]

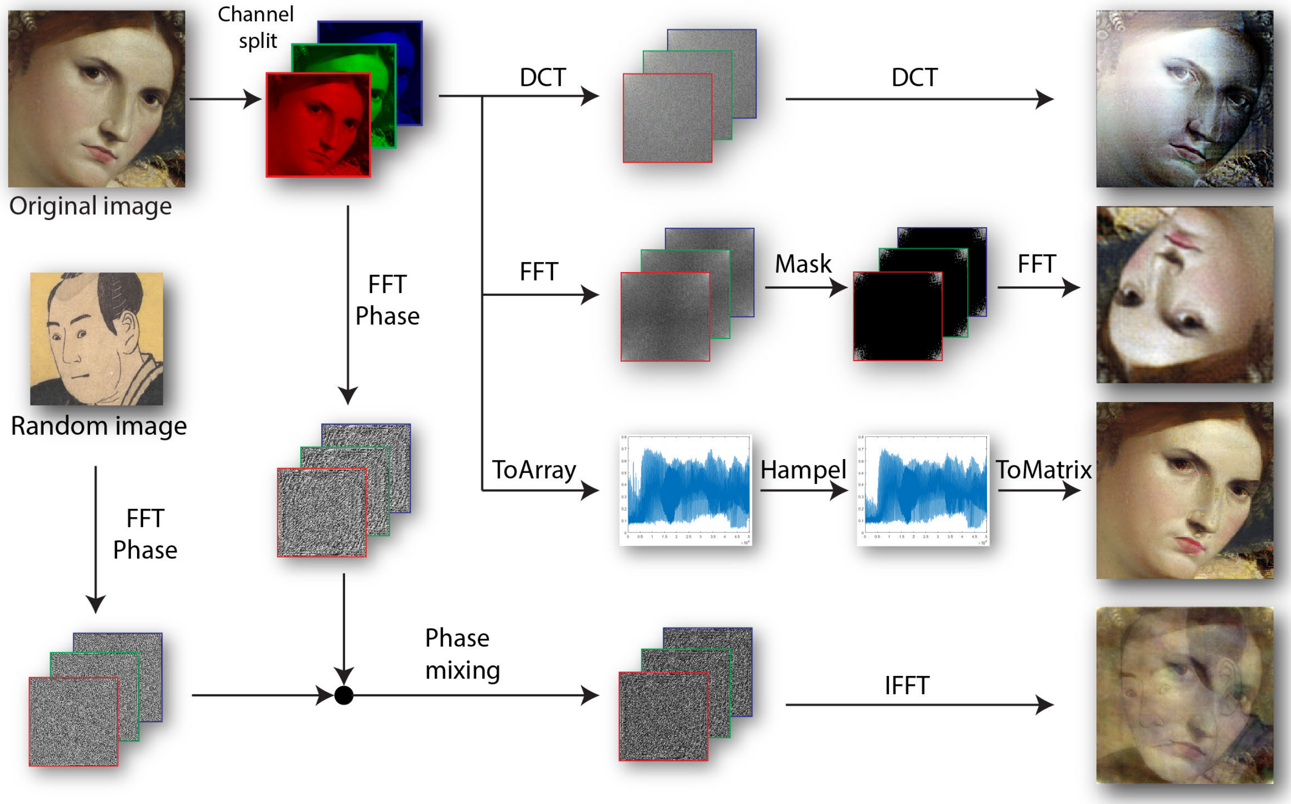


FIGURE 3. Schema for augmentation sets APP15 to APP20.

```
DCTImg = DCT(OriginalImage);
DCT2Img = DCT(DCTImg);
HazeImg = hazeReduction(DCT2Img);
NewImage = histogramEqualization(HazeImg);
```

Listing 1. DCT method.

```
FFTImage = FFT(OriginalImage);
rndMask = STDrandomMask(FFTImage);
FFTImage(rndMask) = 0;
NewImage= modulo(FFT(FFTImage));
```

Where STDrandomMask(matrix) returns a mask of where the value is inside the average standard deviation of each row.

Listing 2. FFT method.

```
[U,S,V] = svd(OriginalImage);
S_mask = S < (max(S)/rand(50,100));
S(S_mask) = 0;
NewImage = U*S*VT;
NewImage= modulo(IFFT(FFTImage));
```

Listing 3. SVD method.

for each color plane. The method used for the ninth image utilizes a SuperPixel segmentation mask with a random number of clusters between 300 and 2000 for each of which the mean color is calculated and used to replace each pixel contained in the defined area (see Listing 4). An image is also obtained using the Hilbert transform which extracts a discrete-time analytic signal from every column of the image in the phase and modulo form from which only the phase is kept to define the new image through value normalization. The last image is the first distorted image of APP9. For the methods just detailed the pseudocode is now reported:

Where STDrandomMask(matrix) returns a mask of where the value is inside the average standard deviation of each row.

APP16. This method utilizes the DCT implementation presented for APP15 and the first distorted image of APP9.

APP17. This method introduces a new augmentation that makes use of the Hampel outlier removal filter, generally used for signals. The image generated from this method is obtained through a process of vectorization, column by column, of the image to which is then applied the filter with a measurement window of 20 and a standard deviation of 1.5. Before being added to the augmented dataset the linearized image is converted back to the original form by simply slicing the image into columns of the final size. This method is combined with the APP15: DCT, Superpixel, and Hilbert

TABLE 2. Description of the datasets used in this study.

Short Name	Full Name	#Classes	#Samples	Image Size	Protocol	Ref
VIR	Virus	15	1500	41×41	10CV	[30]
BARK	Bark	23	23000	∼1600×3800	5CV	[8]
GRAV	Gravity	22	8583	470×570	Tr-Te	[4]
POR	Portraits	6	927	From 80×80 to 2700×2700	10CV	[34]
PBC	Peripheral blood cell classification	8	17092	360x363	Tr-Te	[1]
HE	2D HELA	10	862	512×382	5CV	[5]
MA	Muscle aging	4	237	1600×1200	5CV	[57]
BG	Breast grading carcinoma	3	300	1280×960	5CV	[18]
LAR	Laryngeal dataset	4	1320	1280×960	Tr-Te	[39]
Triz	Gastric lesion types	4	574	352×240	10CV	[75]
END	Histopathological endometrium images	4	3502	640×480	Tr-Te	[63]
RSMAS	coral dataset	14	766	256×256	5CV	[20]
Pest	Pests commonly found on plants	10	563	From ∼60×60 to 300×300	Tr-Te	[16]
InfL	Informative-frame selection in laryngoscopic videos	4	720	∼453×725	Tr-Te	[51]
Pol	Pollen grains	75	2591	From ∼80×80 to 600×600	Tr-Te	[67]

```
[labels,segments] = superpixel(OriginalImage,
  randomInt(300,2000));
indices = label2idx(labels);

for position = 1:segments
    index = indices{position};
    SuperpixelImage(index) =
        mean(OriginalImage(index));
end
```

Listing 4. Superpixel method.

```
FFTImage = FFT(OriginalImage);
FFTrndImage = FFT(RandomImage);
phaseImage1 = angle(FFTImage);
phaseImage2 = angle(FFTrndImage);
mask = phaseImage1 > 0
phaseImage1(mask) = phaseImage2(mask);
RecomposedMatrix = modulo(FFTImage) *
      exp(i*phaseImage1);
NewImage= modulo(IFFT(FFTImage));
```

Listing 5. FFT with combined image.

```
ImgVect = OriginalImage(:);
HampelImg = hampel(ImgVect,20,1.5);
NewImg = reshape(HampelImg, size(OriginalImage));
```

Listing 6. APP17 method.

augmentations. Listing 6 reports a pseudocode of the data augmentation method.

APP18. This augmentation utilizes the ones of APP17 but substitutes the Superpixel method with the Laplacian based presented in APP15.

APP19. This method combines the FFT, Hilbert, and Hampel augmentations with the addition of a combination of APP1 through APP3 that contains: vertical and horizontal flips, random rotations in the range [1°, 180°], gaussian noise addition, cropping of a random number of pixels in the range [0,20] from every side, and hue, saturation, brightness and contrast jittering with random values respectively in the ranges [0.05, 0.15], [−0.4, −0.1], [−0.3, −0.1], [1.2, 1.4].

```
NewImage = OriginalImage;
ImgHeight = size(OriginalImage,1)
ImgWidth = size(OriginalImage,2)

For pixelX = 2:(ImgWidth-1)
        For pixelY = 2:(ImgWidth-1)
                XShift = randomInt(-1,1);
                YShift = randomInt(-1,1);
                NewImage[pixelX,pixelY] =
                    NewImage[pixelX+XShift,
                       pixelY+YShift];
        END
END
```

Listing 7. PixelShuffle method.

APP20. This method makes use of the first distorted image of APP9 together with the Superpixel (see APP15) and PixelShuffle method, as well as a slight variation of the ''Patch shuffling'' technique [12]. This last one, instead of switching the places of regular patches of the original image, it chooses sections of random dimensions to be randomly overlapped with other parts of the image (view Figure 2). The PixelShuffle method, also inspired by the ''Patch shuffling'' technique, performs a shift of each pixel of the image in a randomly chosen position inside the 3 × 3 area surrounding it starting from the top-left corner and progressing row by row. The outcome is a picture where the original pixels have been displaced to nearby positions, possibly resulting in multiple copies of the same pixel or in the removal of it from the resulting image. Listing 7 reports the pseudocode of the data augmentation method.

B. DATASETS

In this work, we assess the proposed ensembles from augmentation methods by adopting several image classification benchmarks. In particular, Table 2 reports all the information for each dataset: a short name, the original dataset name (if

TABLE 3. Performance accuracy (in %) using ResNet50 as a model. For each dataset, best performance value in bold.

Database	SA	BestSA	EnsDA_A	EnsDA_B	EnsDA_C	EnsDA_Mix	EnsDA_MixB	Ens_Base(14)
VIR	86.6	89.33	90	**90.2**	89.33	89.73	89.8	89.73
HE	95.81	96.28	96.51	96.63	96.51	**97.33**	97.21	96.4
MA	95.83	**98.33**	97.08	97.08	97.08	97.5	97.92	97.5
BG	94.00	92.67	94.00	94.00	93.67	93.00	**94.33**	93.67
LAR	94.55	95.3	96.29	96.14	**96.74**	96.44	96.52	96.14
POR	87.16	88.46	89.21	89.96	**90.07**	89.2	89.85	88.02
Bark	89.91	90.63	91.27	91	91.38	91.41	**91.62**	90.66
Grav	97.66	96.4	**98.33**	98.24	**98.33**	97.49	97.58	98.08
TriZ	98.78	98.61	99.13	99.13	99.13	99.13	**99.3**	98.78
END	50.00	**81.50**	76.00	77.5	71.5	74.5	78	50.5
PBC	99.03	98.93	98.98	99.08	99.12	99.12	**99.22**	98.88
RSMAS	99.22	99.35	99.22	99.22	98.95	**99.74**	99.48	99.35
Pest	93.7	93.37	93.98	93.87	94.14	94.2	**94.42**	93.76
InfL	95.56	94.17	96.53	96.53	**96.81**	96.25	96.25	96.25
Pol	93.93	93.93	94.83	94.16	95.06	94.83	**95.51**	94.61
AVG	91.44	93.81	94.09	94.18	93.85	93.99	**94.46**	92.15

TABLE 4. Performance accuracy (in %) using MobileNetv2 as a model. For each dataset, best performance value in bold.

Database	SA	BestSA	EnsDA_A	EnsDA_B	EnsDA_C	EnsDA_Mix	EnsDA_MixB	Ens_Base(14)
VIR	42.93	**88.47**	85.27	84.47	74.73	**88.47**	88.60	47.6
HE	94.58	96.40	96.16	95.47	96.63	96.74	**96.98**	95.00
MA	94.58	94.58	95.83	96.25	95.00	97.50	**97.92**	97.5
BG	91.33	93.33	93.00	**93.33**	**93.33**	92.67	92.67	92.67
LAR	92.80	95.00	96.21	95.98	95.76	95.91	**96.44**	95.3
POR	84.45	88.23	88.56	88.55	87.91	**88.99**	88.66	85.96
Bark	89.79	90.56	91.20	90.95	91.56	91.52	**91.69**	91.04
Grav	97.83	95.74	98.16	**98.24**	98.16	96.91	97.41	98.16
TriZ	97.91	98.26	98.25	98.26	98.25	98.78	**98.95**	98.26
END	74.00	83.00	86.00	87.00	86.50	86.00	**87.50**	83.00
PBC	98.88	98.98	99.17	99.22	99.22	99.27	99.22	**99.37**
RSMAS	97.78	98.69	98.69	98.56	98.69	**99.08**	**99.08**	98.69
Pest	91.77	94.7	92.98	93.31	93.26	**93.92**	93.48	92.82
InfL	94.72	94.03	**95.83**	95.69	94.86	95.69	95.14	95.83
Pol	92.13	92.81	93.03	93.03	93.71	93.71	93.48	**93.93**
AVG	89.03	93.52	93.88	93.88	93.17	94.34	**94.48**	91.00

TABLE 5. Performance accuracy (in %) using DenseNet201 as a model. For each dataset, best performance value in bold.

Database	SA	BestSA	EnsDA_A	EnsDA_B	EnsDA_C	EnsDA_Mix	EnsDA_MixB	Ens_Base(14)
VIR	88.73	**91.27**	90.53	90.73	90.20	90.53	91.00	90.47
HE	96.16	96.28	96.86	96.74	97.21	**97.44**	97.09	96.40
MA	95.00	97.08	97.50	97.92	**98.33**	97.50	96.67	97.50
BG	93.00	94.00	93.33	93.67	93.67	94.00	94.00	**94.33**
LAR	95.91	96.14	96.74	**96.97**	96.52	96.67	96.74	96.36
POR	87.49	88.98	90.28	90.39	**90.93**	90.39	90.39	89.11
Bark	91.42	91.27	92.67	92.59	92.77	92.71	**92.93**	92.42
Grav	97.83	96.15	98.08	97.99	**98.16**	97.16	97.24	97.99
TriZ	99.30	99.13	99.31	99.13	**99.48**	**99.48**	**99.48**	**99.48**
END	67.00	**86.00**	84.00	81.50	81.50	80.50	84.50	75.00
PBC	99.17	99.08	99.27	99.32	99.27	**99.42**	99.22	99.37
RSMAS	99.48	99.22	99.61	99.61	**99.74**	99.61	99.61	99.61
Pest	93.37	94.59	93.98	94.14	94.14	94.53	**94.64**	93.70
InfL	94.86	94.86	**96.81**	**96.81**	96.25	**96.81**	96.53	95.83
Pol	94.83	93.93	**95.06**	**95.06**	**95.06**	94.83	**95.06**	94.83
AVG	92.90	94.53	94.93	94.84	94.88	94.77	**95.00**	94.16

provided in the reference), the number of classes and samples, the size(s) of the images, the testing protocol, and the original reference. For the testing protocols, we adopt the following abbreviations:

- 5CV, 10CV indicates whether a 5-fold or 10-fold cross-validation has been adopted;
- Tr-Te indicates a pre-divided dataset into training and testing sets in the original paper. For instance, LAR and InfL have been divided with a three-fold division and the different folds were provided by the authors. For PBC, the official protocol specifies that 88% of the images be included in the training set and 12% in the test set, with both sets maintaining the same sample per class ratio as in the original dataset. END includes a training set of 3302 images and an external validation set of 200 images.

The performance indicator typically reported on these datasets is accuracy, which measures the rate of correct classifications. For the GRAV dataset, four different views are extracted at different duration from each glitch/image, therefore the final score is obtained by combining the four classification scores via the average rule.

TABLE 6. **EUC using ResNet50 as a model. For each dataset, best performance value in bold.**

ResNet50	SA	BestSA	EnsDA_A	EnsDA_B	EnsDA_C	EnsDA_Mix	EnsDA_MixB	Ens_Base(14)
VIR	2.13	1.63	1.37	1.33	1.42	1.19	**1.18**	1.36
HE	0.4	0.27	0.24	0.23	**0.20**	0.23	**0.20**	0.23
MA	0.79	0.29	0.27	0.27	**0.11**	**0.11**	0.15	0.21
BG	2.74	**2.12**	2.39	2.5	3.14	2.36	2.31	2.42
LAR	0.41	0.37	0.12	**0.11**	0.17	0.13	0.21	0.12
POR	2.69	2.05	1.75	1.68	1.71	1.73	**1.66**	2.45
Bark	1.87	1.71	1.38	1.4	1.37	1.34	**1.32**	1.56
Grav	**0.21**	0.69	0.23	0.22	0.3	0.29	0.27	0.31
TriZ	0.1	0.09	0.04	0.05	0.07	**0.03**	0.06	0.13
END	23.67	9.28	10.73	**9.24**	9.89	12.15	10.6	20.48
PBC	0.03	**0.01**	**0.01**	**0.01**	**0.01**	0.02	**0.01**	0.03
RSMAS	0.01	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Pest	0.75	0.73	0.55	0.57	0.54	0.51	**0.49**	0.71
InfL	0.54	0.52	**0.48**	0.49	**0.48**	0.53	0.54	**0.48**
Pol	1.45	1.75	1.22	1.22	1.31	**1.13**	1.18	1.32
AVG	2.51	1.43	1.38	**1.28**	1.38	1.45	1.34	2.12

TABLE 7. **EUC using MobileNetv2 as a model. For each dataset, best performance value in bold.**

MobileNet	SA	BestSA	EnsDA_A	EnsDA_B	EnsDA_C	EnsDA_Mix	EnsDA_MixB	Ens_Base(14)
VIR	23.59	1.67	2.35	2.24	3.71	**1.55**	1.92	17.46
HE	0.45	0.6	**0.17**	**0.17**	0.22	0.21	0.23	0.24
MA	0.82	0.75	0.2	0.16	0.14	**0.08**	**0.08**	0.19
BG	2.61	**2.09**	3.06	2.77	3.14	2.63	2.94	2.32
LAR	0.56	0.49	0.24	0.23	**0.20**	0.25	0.24	0.28
POR	3.33	2.55	2.04	1.96	2.1	**1.76**	1.89	2.74
Bark	1.74	1.66	1.26	1.3	1.24	1.19	**1.14**	1.34
Grav	0.49	0.83	0.31	**0.29**	0.36	**0.29**	0.34	0.34
TriZ	0.14	0.14	0.06	0.07	0.06	**0.05**	0.06	0.17
END	13.38	6.75	4.79	**4.63**	4.73	5.56	5.33	6.72
PBC	0.04	0.02	0.02	0.02	**0.01**	**0.01**	**0.01**	**0.01**
RSMAS	0.17	0.03	**0.02**	**0.02**	**0.02**	**0.02**	**0.02**	0.03
Pest	0.98	0.7	0.69	0.69	0.7	**0.62**	0.63	0.84
InfL	0.67	**0.50**	0.61	0.55	0.72	0.54	0.52	0.65
Pol	1.67	1.76	1.23	1.25	1.34	1.24	**1.23**	1.35
AVG	3.37	1.37	1.14	1.09	1.25	**1.07**	1.1	2.31

To provide the significance of the results, a statistical analysis has been performed and the Wilcoxon signed rank test [14] provided.

C. DEEP LEARNING MODELS

In this work, we adopt recent deep learning models for developing our ensembles and also as a baseline to assess our proposal in the experimental analysis. In particular, we adopt ResNet50 [22], MobileNetv2 [55], EfficientNetB0 [64], and DenseNet [24]. These models are all recent convolutional-based neural networks and they are adopted in this work to show the feasibility and effectiveness of the proposed solution.

ResNet50 is a deep convolutional neural network trained on the ImageNet dataset, with 50 layers. ResNet50 is considered to be a very effective and efficient model for image classification tasks, and it has been widely used in a variety of applications, including object detection, image recognition, and video classification. It is also often used as a base model for transfer learning, where it is fine-tuned for a specific task using a smaller dataset. One of the key features of ResNet50 is its use of residual connections, which allow the network to learn complex features more effectively by bypassing some of the layers and allowing the gradients to flow more directly through the network. This helps to alleviate the vanishing gradient problem, which can occur when training very deep networks and can make it difficult to learn meaningful features.

MobileNetv2 is a lightweight convolutional neural network designed for efficient on-device inferencing on mobile and embedded devices. It is based on the idea of depthwise separable convolutions, which allows the model to be more efficient and faster to compute than traditional CNNs. In a depthwise separable convolution, the input is first processed by a depthwise convolution, which applies a separate filter to each input channel, and then the resulting feature maps are processed by a pointwise convolution, which combines the feature maps using a 1 × 1 convolution. This allows the model to learn more complex features while still being efficient and fast to compute. MobileNetv2 also introduces the concept of inverted residuals, which are a modified version of the residual connections used in the ResNet architecture. Inverted residuals allow the model to learn more complex features by increasing the dimensionality of the feature maps in the bottlenecks, which are the layers that reduce the spatial dimensions of the feature maps.

TABLE 8. EUC using DenseNet as a model. For each dataset, best performance value in bold.

DenseNet	SA	BestSA	EnsDA_A	EnsDA_B	EnsDA_C	EnsDA_Mix	EnsDA_MixB	Ens_Base(14)
VIR	1.66	1.05	1.08	1.08	1.1	**0.98**	1.05	1.05
HE	0.28	0.21	0.18	**0.14**	0.18	**0.14**	0.17	0.22
MA	0.35	0.22	0.1	0.07	0.06	0.03	0.09	**0.01**
BG	2.03	**1.5**	2.59	2.59	2.67	2.3	2.57	2.15
LAR	0.22	0.52	0.18	0.16	0.2	0.19	0.16	**0.09**
POR	2.26	1.82	1.22	**1.13**	1.32	1.19	1.21	1.69
Bark	1.33	1.31	0.97	0.99	0.97	0.97	**0.94**	1.04
Grav	0.29	0.5	**0.19**	0.2	0.33	0.26	0.23	0.28
TriZ	0.02	0.1	0.04	0.04	**0.03**	0.04	0.04	0.04
END	13.41	4.83	5.59	5.32	5.75	4.78	**4.49**	7.93
PBC	0.02	**0.01**	**0.01**	**0.01**	**0.01**	**0.01**	**0.01**	**0.01**
RSMAS	**0**	0.01	**0**	**0**	**0**	**0**	**0**	**0**
Pest	0.62	0.62	0.45	0.45	0.5	**0.43**	0.45	0.6
InfL	0.62	0.54	0.47	**0.44**	0.48	**0.44**	0.55	0.45
Pol	1.18	**0.66**	1.2	1.08	1.24	0.92	1	1.14
AVG	1.62	0.93	0.95	0.91	0.98	**0.84**	0.86	1.11

TABLE 9. p-value among different pairs of ensembles.

	ResNet50		MobileNetV2		DenseNet201		EfficientNetB0		AVG	
	Accuracy	EUC	Accuracy	EUC	Accuracy	EUC	Accuracy	EUC	Accuracy	EUC
SA vs EnsBase(14)	0.002	0.002	6.10e-05	2.44e-04	1.22e-04	0.0055	0.002	0.001	0.001	0.002
EnsDA_A vs EnsBase(14)	0.008	0.011	0.145	0.012	0.057	0.569	0.006	0.075	0.054	0.167
EnsDA_B vs EnsBase(14)	0.067	0.02	0.296	0.005	0.048	0.21	0.042	0.013	0.113	0.062
EnsDA_C vs EnsBase(14)	0.047	0.114	0.167	0.051	0.025	0.907	0.049	0.095	0.072	0.292
EnsDA_Mix vs EnsBase(14)	0.064	0.027	0.032	0.006	0.056	0.096	0.052	0.011	0.051	0.015
EnsDA_MixB vs EnsBase(14)	0.002	0.007	0.051	0.009	0.127	0.329	0.004	0.009	0.046	0.116

EfficientNet is a family of convolutional neural network models that were developed to improve the efficiency and effectiveness of deep learning models. The EfficientNet models are designed to be scalable, so that they can be easily adapted to a variety of applications and datasets by adjusting the model size. They are characterized by their use of compound scaling, which allows them to improve the model performance by scaling up the network dimensions in a balanced way. The network dimensions include the number of channels in the convolutional layers, the spatial resolution of the input, and the depth of the network. By scaling these dimensions appropriately, the EfficientNet models can achieve better performance with fewer parameters and less computation than other CNNs. EfficientNet also introduces the concept of autotuning, which allows the model to automatically search for the optimal balance of network size, resolution, and depth for a given task.

DenseNet is a type of convolutional neural network that introduces the concept of dense connections, which allows the network to learn more complex features and improve performance. In a DenseNet, each layer is connected to all of the preceding layers, rather than just the immediately preceding layer as in traditional CNNs. This allows the network to incorporate features from all of the previous layers, which can be beneficial when learning from datasets with highly correlated features. DenseNet also uses a growth rate parameter to control the number of feature maps in each layer, which helps to reduce the number of parameters in the model and improve efficiency.

IV. EMPIRICAL EVALUATION

In this section we report the results of the empirical evaluation. All the experiments were taken on a Windows Server 2019, with an Intel Core i9-10920X CPU, 3.5 GHz, and 256 GB RAM, we employed an Nvidia Titan RTX 24 GB, 1350 MHz. They are developed in Matlab 2022a. We start our experiments by comparing the accuracy obtained by the following approaches:

- SA, a stand-alone network trained on APP3, which is a quite standard data augmentation approach.
- BestSA, a stand-alone network trained on APP19 which produces the best average performance compared with all the other data augmentation sets.
- EnsDA_A, this is the fusion by sum rule among all the CNNs trained using APP1 to APP11; each net is trained with a different data augmentation approach. The data augmentation methods based on color spaces (i.e., APP6 to APP8) are not reported on VIR, HE, and MA since they are gray-level images.
- EnsDA_B, this fusion is the same as EnsDA_A except for the addition of nets trained with the augmentation methods APP12-14.
- EnsDA_C, this is the fusion by sum rule among those methods not based on feature transforms. Each approach is iterated twice (three times for datasets VIR, HE, and MA since they are gray-level images; they are trained three times so that the size of the ensemble EnsDA_C is similar to EnsDA_B).
- EnsDA_Mix, this is the fusion by sum rule among the methods trained with APP1, APP2, APP10:APP20;
- EnsDA_MixB, this is the fusion by sum rule among the methods trained on APP1:APP9 APP15:APP19;
- EnsBase(X), this is a baseline ensemble intended to compare and validate the performance of EnsDA_*; EnsBase(X) combines (via sum rule) X networks trained

separately on APP3, which is a quite standard data augmentation approach.

All the adopted models are pre-trained on ImageNet. In particular, in Table 3, 4, and 5 we report the accuracy of different architectures that adopt different networks as models.

In all three topologies, the highest average accuracy is obtained by EnsDA_MixB, which clearly outperforms the baseline Ens_Base(14).

Accuracy is probably the most commonly used performance measure for classification problems, but it is the least suitable for comparing classifiers because accuracy depends on the choice of classification threshold. To address this issue, the area under the curve (AUC) [21] is a standard measure of choice when testing the performance of predictive models. It gives a likelihood estimation that a predictor ranks a random positive instance higher than a randomly selected negative instance. This work uses the error under the ROC curve (EUC), which is an extension of AUC (i.e., $EUC = 1 - AUC$). Tables 6, 7, and 8 show the performance of the proposed approach in terms of EUC. Since EUC is a binary classification measure, the same average EUC value is used for all multiclass problems. The results in Tables 6, 7, and 8 mainly show trends in accuracy. That is, the proposed ensemble outperforms both the base ensemble and independent methods based on data augmentation.

The only dataset where the new approach clearly performs poorly, across all topologies, compared to EnsDA_B, is the Gravity dataset; that dataset is built by spectrograms. Since spectrograms have time and frequency on their axes, not all the standard image augmentation techniques are useful, e.g. reflection in APP15's FFT-based method. Thus, it is of the main importance to use specific augmentation techniques for spectrograms, such as linearly interpolating between pairs of real training examples [72]. This has been shown to improve the generalization performance of neural networks, especially when the data is limited or the model is prone to overfitting. In literature, it is used in exactly the same way and works both for RGB and spectrograms, the principle is the same, overlapping the signal [71].

In Table 9, we compare the different ensemble with the baseline ensemble (i.e. EnsBase(14)), the different methods are compared using the Wilcoxon signed-rank tests. It is very interesting to note that EnsDA_Mix outperforms EnsBase(14) with a p-value smaller than 0.1 in all the topologies and for both the performance indicators. In this regard, it is worth noting that the higher p-value was reached with the model that employs DenseNet as a backbone, which is the model that gets the best performance. It is very hard to boost the performance when the backbone is already at its best because at some point the model reaches a plateau for a given dataset. EnsDA_Mix is the method suggested in this work.

TABLE 10. p-values among different pairs of ensembles computed on the MCC.

p-value	ResNet50	MobileNetV2	DenseNet201
EnsDA_B vs EnsBase(14)	0.067	0.118	0.041
EnsDA_C vs EnsBase(14)	0.049	0.078	0.041
EnsDA_Mix vs EnsBase(14)	**0.005**	**0.002**	**0.024**

TABLE 11. Inference time in seconds to classify a batch of 100 images on different GPUs. Ensembles consists of 15 classifiers.

Model	Type	GTX1080	Titan Xp	Titan RTX
ResNet50	Single	0.36	0.31	0.22
	Ensemble	5.58	4.12	2.71
MobileNetV2	Single	0.4	0.35	0.24
	Ensemble	6.21	5.51	3.9
DenseNet201	Single	1.55	1.32	0.97
	Ensemble	21.85	18.65	14.26
EfficientNetB0	Single	0.56	0.44	0.35
	Ensemble	7.01	6.52	5.92

TABLE 12. Performance accuracy (in %) for stand-alone (i.e., ReLu) and ensemble methods with (i.e., EnsBase) and without (i.e., 15ReLU) data augmentation.

Model		HE	BG	LAR
ResNet50	ReLU	94.19	89.67	91.44
	15ReLU	95.70	91.00	93.79
	EnsBase	96.40	93.67	96.14
MobileNetV2	ReLU	92.91	89.00	90.23
	15ReLU	95.23	90.67	91.52
	EnsBase	95.00	92.67	95.03
DenseNet201	ReLU	95.29	91.69	93.96
	15ReLU	96.40	95.33	96.14
	EnsBase	96.40	94.33	96.36

In Table 13, we compare the performance of EnsDA_Mix ensembles with the best methods reported in the literature on the same datasets. As can be observed, our proposed best method obtains state-of-the-art or similar performance in all the datasets. Note that the performance indicator is the F1-measure with the LAR and InfL dataset because that is the measure that is reported most commonly in the literature for this dataset. Moreover, we have reported the performance of two ensembles based on the three tested topologies:

- EnsTop, sum rule among the three EnsDA_Mix calculated using, ResNet50, MobileNetV2 and DenseNet201
- EnsTop_W, as the previous one, but the methods are combined with the weighted sum rule the weight of DenseNet201 is 2, since DenseNet201 obtains on average the best performance.

We can say that EnsTop_W outperforms all the other approaches and that the result is statistically significant, with a $p - value < 0.05$, for the following cases: EnsTop_W vs EnsDA_Mix-ResNet50 ($p - value = 0.0006$) and EnsTop_W vs EnsDA_Mix-MobileNetV2 ($p - value = 0.0081$).

As a final test, we computed the Matthews correlation coefficient (MCC) as the performance indicator for the three best-performing groups to check which one is the best. In the literature [10], MCC is proved to be a more reliable statistical indicator than accuracy even for binary classification datasets. Table 10 reports the p-values among the different pairs of the ensembles, it can be noticed that the performance of EnsDA_Mix is always statistically significant with respect to EnsBas(14) with a $p - value$ always smaller than 0.05.

Notice that BestSA results in 13 new images for each original sample in the training set. It is unfeasible to apply

TABLE 13. Performance as a measure of accuracy (in %) compared with the best in the literature. In square brackets, we report the reference that provides the best performance on a dataset. ** On LAR and InfL, F1 is the performance measure. *** The method in [30] combines descriptors based on both object scale and fixed scale images. **** Only handcrafted features are used.

VIR	89.73	88.47	90.53	90.47	90.8	[45] 89.6	[43] 89.47	[13] 89	[69] 88	[3] 87.27	[30] 87.00***	[69] 86.2	[19] 85.7	[46] **92.53**
HE	97.33	96.74	97.44	97.44	97.33	[62] **98.3**	[11] 94.4	[18] 84	[77] 68.3	[37] 96.81	[42] 97.21	[46] 96.74		
MA	97.5	97.5	97.5	97.5	98.33	[62] 97.9	[56] 53	[77] 89.6	[42] **98.75**					
BG	93	92.67	94	94.33	94.33	[18] **96.3**	[44] 95	[46] 95						
LAR**	96.41	95.88	96.63	96.78	96.78	[44] 95.2	[39] 92	[46] **97.04**						
POR	89.2	88.99	90.39	90.39	**90.71**	[34] 90.08								
BARK	91.41	91.52	**92.71**	92.56	92.69	[6] 48.9	[54] 85	[53] 90.4	[8] 87.04	[7] 63.89				
GRAV	96.91	96.91	97.16	97.24	97.41	[4] **98.21**	[52] 96.9							
Triz	97.49	98.78	**99.48**	99.13	99.13	[75]**** 87								
END	74.5	**86.00**	80.5	82.5	81	[63] 76.91								
PBC	99.12	99.27	**99.42**	99.37	**99.42**	[35] 99.3	[66] 97.94							
RSMAS	**99.74**	99.08	99.61	99.48	99.61	[20] 97.95	[58] 96.9	[36] 99.2						
Pest	94.2	93.92	94.53	94.92	95.03	[16] 85.5	[2] 95.16	[46] **95.52**						
InfL**	96.24	95.67	96.8	96.79	**96.93**	[51] 93.59								
Pol	**94.83**	93.71	**94.83**	94.61	94.61	[67] 91.68								

EnsBase(14), using as a backbone the method BestSA, to all the datasets and for all the topologies, due to the fact that this would become too expensive in terms of training time, also when GPUs are employed. Anyway, our goal is to develop new ensemble methods leveraging different data augmentation methods.

The data augmentation methods do not increase the complexity as the time spent to train the system is proportional to the number of images generated. For instance, on the 2D HELA dataset, a ResNet50 takes 400 seconds to complete the training process. When APP19 is applied, the same network takes 4400 seconds to complete the training process. Notice that APP19 produces 13 new images making the resulting running time linear in the number of images generated.

In Table 11, we report the inference time in seconds taken to classify a batch of 100 images. We compare the time taken by a single model (specified in the first column) and an ensemble made of 15 classifiers. We can notice that the time spent by the ensembles is linear in the number of classifiers in the ensemble. This is a positive aspect since each classifier is independent this operation could be parallelized to speed up the process.

A. ABLATION ANALYSIS

In order to prove the advantage of the proposed method, we perform an ablation study to compare whether the application of data augmentation increases the performance with respect to a baseline method. First, notice from Tables 3, 4, and 5, that the stand-alone network (column SA) provides a performance that is on average lower than the ensembles. This is true always but in the Gravity dataset where the performance is very close. On average, the adoption of the data augmentation method in the ensemble provides an increase in the performance of almost 3%.

To control whether the contribution of the data augmentation methods is beneficial, we compared the performance of a stand-alone network with an ensemble trained with and without data augmentation. Results for some of the datasets are reported in Table 12, where ReLu reports the performance of a stand-alone network, 15ReLU is an ensemble trained without data augmentation, and EnsBase is the ensemble described in the previous sections. It can be noticed that adopting an ensemble approach always gets better performance when compared with the stand-alone methods. Moreover, it can be seen that the adoption of a data augmentation method in the training process allows for a further increase in the performance of the ensemble. Providing another piece of evidence about the benefits brought by our proposal.

V. CONCLUSION

In this study, we compare combinations of pre-trained CNNs that have been fine-tuned to various training sets while incorporating the best image manipulation methods for creating new images. In comparison to various benchmarks that represented various image classification tasks, we evaluated the performance of these networks and their combinations. The reliability of CNNs is improved by combining images produced by various data augmentation techniques into a data-level deep learning ensemble, as demonstrated in this study. Given the variety of landmarks used, the method we use to construct CNN ensembles should be effective for the majority of imaging issues.

ACKNOWLEDGMENT

The authors used a donated TitanX GPU to train CNNs used in this work.

DECLARATION OF INTERESTS

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this article.

REFERENCES

[1] A. Acevedo, A. Merino, S. Alférez, Á. Molina, L. Boldú, and J. Rodellar, ‘‘A dataset of microscopic peripheral blood cell images for development of automatic recognition systems,’’ *Data Brief*, vol. 30, Jun. 2020, Art. no. 105474.

[2] E. Ayan, H. Erbay, and F. Varçın, ‘‘Crop pest classification with a genetic algorithm-based weighted ensemble of deep convolutional neural networks,’’ *Comput. Electron. Agricult.*, vol. 179, Dec. 2020, Art. no. 105809.

[3] A. R. Backes and J. J. de Mesquita Sa Junior, ‘‘Virus classification by using a fusion of texture analysis methods,’’ in *Proc. Int. Conf. Syst., Signals Image Process. (IWSSIP)*, Jul. 2020, pp. 290–295.

[4] S. Bahaadini, V. Noroozi, N. Rohani, S. Coughlin, M. Zevin, J. R. Smith, V. Kalogera, and A. Katsaggelos, ‘‘Machine learning for gravity spy: Glitch classification and dataset,’’ *Inf. Sci.*, vol. 444, pp. 172–186, May 2018.

[5] M. V. Boland and R. F. Murphy, ‘‘A neural network classifier capable of recognizing the patterns of all major subcellular structures in fluorescence microscope images of HeLa cells,’’ *Bioinformatics*, vol. 17, no. 12, pp. 1213–1223, Dec. 2001.

[6] S. Boudra, I. Yahiaoui, and A. Behloul, ‘‘A set of statistical radial binary patterns for tree species identification based on bark images,’’ *Multimedia Tools Appl.*, vol. 80, no. 15, pp. 22373–22404, Jun. 2021.

[7] S. Boudra, I. Yahiaoui, and A. Behloul, ‘‘Tree trunk texture classification using multi-scale statistical macro binary patterns and CNN,’’ *Appl. Soft Comput.*, vol. 118, Mar. 2022, Art. no. 108473.

[8] M. Carpentier, P. Giguere, and J. Gaudreault, ‘‘Tree species identification from bark images using convolutional neural networks,’’ in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 1075–1081.

[9] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, ‘‘Return of the devil in the details: Delving deep into convolutional nets,’’ in *Proc. Brit. Mach. Vis. Conf.*, 2014, pp. -1–12.

[10] D. Chicco and G. Jurman, ‘‘The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation,’’ *BMC Genomics*, vol. 21, no. 1, pp. 1–13, 2020.

[11] L. P. Coelho, J. D. Kangas, A. W. Naik, E. Osuna-Highley, E. Glory-Afshar, M. Fuhrman, R. Simha, P. B. Berget, J. W. Jarvik, and R. F. Murphy, ‘‘Determining the subcellular location of new proteins from microscope images using local features,’’ *Bioinformatics*, vol. 29, no. 18, pp. 2343–2349, 2013.

[12] P. Dai, H. Zhu, S. Ge, R. Zhang, X. Qian, X. Li, and K. Yuan, ‘‘MIPR: Automatic annotation of medical images with pixel rearrangement,’’ 2022, *arXiv:2204.10513*.

[13] A. R. de Geus, A. Backes, and J. Souza, ‘‘Variability evaluation of CNNs using cross-validation on viruses images,’’ in *Proc. 15th Int. Joint Conf. Comput. Vis., Imag. Comput. Graph. Theory Appl.*, 2020, pp. 626–632.

[14] J. Demšar, ‘‘Statistical comparisons of classifiers over multiple data sets,’’ *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, Jan. 2006.

[15] 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.

[16] L. Deng, Y. Wang, Z. Han, and R. Yu, ‘‘Research on insect pest image detection and recognition based on bio-inspired methods,’’ *Biosyst. Eng.*, vol. 169, pp. 139–148, May 2018.

[17] T. DeVries and G. W. Taylor, ‘‘Improved regularization of convolutional neural networks with cutout,’’ 2017, *arXiv:1708.04552*.

[18] K. Dimitropoulos, P. Barmpoutis, C. Zioga, A. Kamas, K. Patsiaoura, and N. Grammalidis, ‘‘Grading of invasive breast carcinoma through Grassmannian VLAD encoding,’’ *PLoS ONE*, vol. 12, no. 9, Sep. 2017, Art. no. e0185110.

[19] F. L. C. D. Santos, M. Paci, L. Nanni, S. Brahnam, and J. Hyttinen, ‘‘Computer vision for virus image classification,’’ *Biosyst. Eng.*, vol. 138, pp. 11–22, Oct. 2015.

[20] A. Gómez-Ríos, S. Tabik, J. Luengo, A. Shihavuddin, B. Krawczyk, and F. Herrera, ‘‘Towards highly accurate coral texture images classification using deep convolutional neural networks and data augmentation,’’ *Expert Syst. Appl.*, vol. 118, pp. 315–328, Mar. 2019.

[21] J. A. Hanley and B. J. McNeil, ‘‘The meaning and use of the area under a receiver operating characteristic (ROC) curve,’’ *Radiology*, vol. 143, no. 1, pp. 29–36, Apr. 1982.

[22] 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.

[23] D. Hendrycks, N. Mu, E. D. Cubuk, B. Zoph, J. Gilmer, and B. Lakshminarayanan, ‘‘AugMix: A simple data processing method to improve robustness and uncertainty,’’ 2019, *arXiv:1912.02781*.

[24] G. Huang, Z. Liu, G. Pleiss, L. V. D. Maaten, and K. Q. Weinberger, ‘‘Convolutional networks with dense connectivity,’’ *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 12, pp. 8704–8716, Dec. 2022.

[25] H. Inoue, ‘‘Data augmentation by pairing samples for images classification,’’ 2018, *arXiv:1801.02929*.

[26] G. Kang, X. Dong, L. Zheng, and Y. Yang, ‘‘PatchShuffle regularization,’’ 2017, *arXiv:1707.07103*.

[27] C. Khosla and B. S. Saini, ‘‘Enhancing performance of deep learning models with different data augmentation techniques: A survey,’’ in *Proc. Int. Conf. Intell. Eng. Manage. (ICIEM)*, Jun. 2020, pp. 79–85.

[28] A. Krizhevsky, ‘‘Learning multiple layers of features from tiny images,’’ M.S. thesis, Univ. Toronto, Toronto, ON, Canada, 2009.

[29] A. Krizhevsky, I. Sutskever, and G. E. Hinton, ‘‘ImageNet classification with deep convolutional neural networks,’’ *Commun. ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[30] G. Kylberg, M. Uppström, and I. M. Sintorn, ‘‘Virus texture analysis using local binary patterns and radial density profiles,’’ in *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications* (Lecture Notes in Computer Science), vol. 7042, C. San Martin and S. W. Kim, Eds. Berlin, Germany: Springer, 2011, doi: 10.1007/978-3-642-25085-9_68.

[31] C. Li, C. Guo, L. Han, J. Jiang, M.-M. Cheng, J. Gu, and C. C. Loy, ‘‘Low-light image and video enhancement using deep learning: A survey,’’ *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 12, pp. 9396–9416, Dec. 2022.

[32] C. Li, C. Guo, and C. C. Loy, ‘‘Learning to enhance low-light image via zero-reference deep curve estimation,’’ 2021, *arXiv:2103.00860*.

[33] D. Liang, F. Yang, T. Zhang, and P. Yang, ‘‘Understanding mixup training methods,’’ *IEEE Access*, vol. 6, pp. 58774–58783, 2018.

[34] S. Liu, J. Yang, S. S. Agaian, and C. Yuan, ‘‘Novel features for art movement classification of portrait paintings,’’ *Image Vis. Comput.*, vol. 108, Apr. 2021, Art. no. 104121.

[35] F. Long, J.-J. Peng, W. Song, X. Xia, and J. Sang, ‘‘BloodCaps: A capsule network based model for the multiclassification of human peripheral blood cells,’’ *Comput. Methods Programs Biomed.*, vol. 202, Apr. 2021, Art. no. 105972.

[36] A. Lumini, L. Nanni, and G. Maguolo, ‘‘Deep learning for plankton and coral classification,’’ *Appl. Comput. Informat.*, Nov. 2019, doi: 10.1016/j.aci.2019.11.004.

[37] R. Maurya, V. K. Pathak, and M. K. Dutta, ‘‘Deep learning based microscopic cell images classification framework using multi-level ensemble,’’ *Comput. Methods Programs Biomed.*, vol. 211, Nov. 2021, Art. no. 106445.

[38] A. Mikołajczyk and M. Grochowski, ‘‘Data augmentation for improving deep learning in image classification problem,’’ in *Proc. Int. Interdiscipl. PhD Workshop (IIPhDW)*, May 2018, pp. 117–122.

[39] S. Moccia, E. D. Momi, M. Guarnaschelli, M. Savazzi, and A. Laborai, ‘‘Confident texture-based laryngeal tissue classification for early stage diagnosis support,’’ *J. Med. Imag.*, vol. 4, no. 3, Sep. 2017, Art. no. 034502.

[40] F. J. Moreno-Barea, F. Strazzera, J. M. Jerez, D. Urda, and L. Franco, "Forward noise adjustment scheme for data augmentation," in *Proc. IEEE Symp. Ser. Comput. Intell. (SSCI)*, Nov. 2018, pp. 728–734.

[41] J. Nalepa, M. Myller, and M. Kawulok, "Training- and test-time data augmentation for hyperspectral image segmentation," *IEEE Geosci. Remote Sens. Lett.*, vol. 17, no. 2, pp. 292–296, Feb. 2020.

[42] L. Nanni, S. Brahnam, S. Ghidoni, and G. Maguolo, "General purpose (GenP) bioimage ensemble of handcrafted and learned features with data augmentation," 2019, *arXiv:1904.08084*.

[43] L. Nanni, E. De Luca, M. L. Facin, and G. Maguolo, "Deep learning and handcrafted features for virus image classification," *J. Imag.*, vol. 6, no. 12, p. 143, Dec. 2020.

[44] L. Nanni, S. Ghidoni, and S. Brahnam, "Ensemble of convolutional neural networks for bioimage classification," *Appl. Comput. Informat.*, vol. 17, no. 1, pp. 19–35, Jan. 2021.

[45] L. Nanni, S. Ghidoni, and S. Brahnam, "Deep features for training support vector machines," *J. Imag.*, vol. 7, no. 9, p. 177, Sep. 2021.

[46] L. Nanni, A. Manfè, G. Maguolo, A. Lumini, and S. Brahnam, "High performing ensemble of convolutional neural networks for insect pest image detection," *Ecolog. Informat.*, vol. 67, Mar. 2022, Art. no. 101515.

[47] L. Nanni, M. Paci, S. Brahnam, and A. Lumini, "Comparison of different image data augmentation approaches," *J. Imag.*, vol. 7, no. 12, p. 254, Nov. 2021.

[48] L. Nanni, M. Paci, S. Brahnam, and A. Lumini, "Feature transforms for image data augmentation," *Neural Comput. Appl.*, vol. 34, no. 24, pp. 22345–22356, Dec. 2022.

[49] H. Naveed, "Survey: Image mixing and deleting for data augmentation," 2021, *arXiv:2106.07085*.

[50] M. Palatucci, D. Pomerleau, G. E. Hinton, and T. M. Mitchell, "Zero-shot learning with semantic output codes," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 22, 2009, pp. 1410–1418.

[51] I. Patrini, M. Ruperti, S. Moccia, L. S. Mattos, E. Frontoni, and E. D. Momi, "Transfer learning for informative-frame selection in laryngoscopic videos through learned features," *Med. Biol. Eng. Comput.*, vol. 58, no. 6, pp. 1225–1238, Jun. 2020.

[52] Z. Ramezani and A. Pourdarvish, "Transfer learning using tsallis entropy: An application to gravity spy," *Phys. A, Stat. Mech. Appl.*, vol. 561, Jan. 2021, Art. no. 125273.

[53] V. Remeš and M. Haindl, "Rotationally invariant bark recognition," in *Structural, Syntactic, and Statistical Pattern Recognition* (Lecture Notes in Computer Science), vol. 11004, X. Bai, E. Hancock, T. Ho, R. Wilson, B. Biggio, and A. Robles-Kelly, Eds. Cham, Switzerland: Springer, 2018, doi: 10.1007/978-3-319-97785-0_3.

[54] V. Remeš and M. Haindl, "Bark recognition using novel rotationally invariant multispectral textural features," *Pattern Recognit. Lett.*, vol. 125, pp. 612–617, Jul. 2019.

[55] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4510–4520.

[56] L. Shamir, N. Orlov, D. M. Eckley, T. Macura, J. Johnston, and I. G. Goldberg, "Wndchrm—An open source utility for biological image analysis," *Source Code Biol. Med.*, vol. 3, no. 1, pp. 1–13, Jul. 2008.

[57] L. Shamir, N. Orlov, D. M. Eckley, T. J. Macura, and I. G. Goldberg, "IICBU 2008–A proposed benchmark suite for biological image analysis," *Med. Biol. Eng. Comput.*, vol. 46, no. 9, p. 943, 2008.

[58] A. S. M. Shihavuddin, N. Gracias, R. Garcia, A. Gleason, and B. Gintert, "Image-based coral reef classification and thematic mapping," *Remote Sens.*, vol. 5, no. 4, pp. 1809–1841, Apr. 2013.

[59] J. Shijie, W. Ping, J. Peiyi, and H. Siping, "Research on data augmentation for image classification based on convolution neural networks," in *Proc. Chin. Autom. Congr. (CAC)*, Oct. 2017, pp. 4165–4170.

[60] V. Shirke, R. Walika, and L. Tambade, "Drop: A simple way to prevent neural network by overfitting," *Int. J. Res. Eng. Sci. Manag*, vol. 1, pp. 2581–5782, 2018.

[61] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *J. Big Data*, vol. 6, no. 1, pp. 1–48, Dec. 2019.

[62] Y. Song, W. Cai, H. Huang, D. Feng, Y. Wang, and M. Chen, "Bioimage classification with subcategory discriminant transform of high dimensional visual descriptors," *BMC Bioinf.*, vol. 17, no. 1, pp. 1–15, Nov. 2016.

[63] H. Sun, X. Zeng, T. Xu, G. Peng, and Y. Ma, "Computer-aided diagnosis in histopathological images of the endometrium using a convolutional neural network and attention mechanisms," *IEEE J. Biomed. Health Informat.*, vol. 24, no. 6, pp. 1664–1676, Jun. 2020.

[64] M. Tan and Q. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 6105–6114.

[65] L. Taylor and G. Nitschke, "Improving deep learning with generic data augmentation," in *Proc. IEEE Symp. Ser. Comput. Intell. (SSCI)*, Nov. 2018, pp. 1542–1547.

[66] F. Ucar, "Deep learning approach to cell classificatio in human peripheral blood," in *Proc. 5th Int. Conf. Comput. Sci. Eng. (UBMK)*, Sep. 2020, pp. 383–387.

[67] N. Vallez, G. Bueno, O. Deniz, and S. Blanco, "Diffeomorphic transforms for data augmentation of highly variable shape and texture objects," *Comput. Methods Programs Biomed.*, vol. 219, Jun. 2022, Art. no. 106775.

[68] G. A. Velasco, N. Holighaus, M. Dörfler, and T. Grill, "Constructing an invertible constant-q transform with non-stationary Gabor frames," in *Proc. DAFX*, vol. 33, Paris, France, 2011, pp. 1–7.

[69] Z.-J. Wen, Z.-H. Liu, Y.-C. Zong, and B.-J. Li, "Latent local feature extraction for low-resolution virus image classification," *J. Oper. Res. Soc. China*, vol. 8, no. 1, pp. 117–132, Mar. 2020.

[70] Y. Xian, C. H. Lampert, B. Schiele, and Z. Akata, "Zero-shot learning—A comprehensive evaluation of the good, the bad and the ugly," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 9, pp. 2251–2265, Sep. 2019.

[71] K. Xu, D. Feng, H. Mi, B. Zhu, D. Wang, L. Zhang, H. Cai, and S. Liu, "Mixup-based acoustic scene classification using multi-channel convolutional neural network," in *Proc. Pacific Rim Conf. Multimedia*. Cham, Switzerland: Springer, 2018, pp. 14–23.

[72] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "Mixup: Beyond empirical risk minimization," 2017, *arXiv:1710.09412*.

[73] W. Zhang, Y. Wang, and C. Li, "Underwater image enhancement by attenuated color channel correction and detail preserved contrast enhancement," *IEEE J. Ocean. Eng.*, vol. 47, no. 3, pp. 718–735, Jul. 2022.

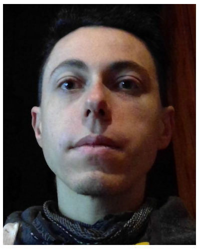
[74] W. Zhang, P. Zhuang, H.-H. Sun, G. Li, S. Kwong, and C. Li, "Underwater image enhancement via minimal color loss and locally adaptive contrast enhancement," *IEEE Trans. Image Process.*, vol. 31, pp. 3997–4010, 2022.

[75] R. Zhao, R. Zhang, T. Tang, X. Feng, J. Li, Y. Liu, R. Zhu, G. Wang, K. Li, W. Zhou, Y. Yang, Y. Wang, Y. Ba, J. Zhang, Y. Liu, and F. Zhou, "TriZ-a rotation-tolerant image feature and its application in endoscope-based disease diagnosis," *Comput. Biol. Med.*, vol. 99, pp. 182–190, Aug. 2018.

[76] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang, "Random erasing data augmentation," in *Proc. IEEE AAAI Conf. Artif. Intell.*, vol. 34, no. 7, May 2020, pp. 13001–13008.

[77] J. Zhou, S. Lamichhane, G. Sterne, B. Ye, and H. Peng, "BIOCAT: A pattern recognition platform for customizable biological image classification and annotation," *BMC Bioinf.*, vol. 14, no. 1, pp. 1–14, Oct. 2013.

RICCARDO BRAVIN received the bachelor's degree in computer engineering from the University of Padua, in 2022. He is currently pursuing the master's degree in computer science and engineering with the Politecnico of Milano.



LORIS NANNI is currently an Associate Professor with the Department of Information Engineering, University of Padua. He carries out research at DEI, University of Padua, in the fields of biometric systems, pattern recognition, machine learning, image databases, and bioinformatics. He is the coauthor of more than 300 research papers. He has an H-index of 54 and more than 10,000 citations (Google Scholar). He has extensively served as a Referee for international journals, such as the IEEE Transactions on Pattern Analysis and Machine Intelligence, *Pattern Recognition*, *Bioinformatics*, the *BMC Bioinformatics*, and the *Pattern Recognition Letters*, and projects.

ANDREA LOREGGIA received the master's degree (cum laude) from the University of Padua, in 2012, and the Ph.D. degree in computer science, in 2016. He is currently an Assistant Professor with the Department of Information Engineering, University of Brescia. His studies are dedicated to designing and providing tools for developing intelligent agents capable of representing and reasoning with preference and ethical-moral principles. His research interest includes artificial intelligence span from knowledge representation to deep learning. He is a member of the UN/CEFACT Group of Experts, he actively participates in the dissemination and sustainable development of technology.

SHERYL BRAHNAM received the master's degree from The City College of New York, in 1997, and the Ph.D. degree in computer science from the Graduate Center, City University of New York, in 2002. She is currently a Professor with Missouri State University. Her research interests include pattern recognition, face recognition, bioinformatics, and medical image analysis.

MICHELANGELO PACI received the B.Sc. and M.Sc. degrees in biomedical engineering, the B.Sc. degree in computer engineering, and the Ph.D. degree in bioengineering from the University of Bologna, Italy, in 2004, 2006, 2007, and 2013, respectively. After two years in industrial automation, from 2008 to 2009, he spent nine years in academia doing research with the Tampere University, Finland, including silico modeling of human cardiomyocytes with a focus on in silico drug assays and machine learning and texture analysis for image classification. Currently, he works in the industry as a Software and Firmware Designer, still being a Docent in computational cardiology with Tampere University.

Open Access funding provided by 'Università degli Studi di Padova' within the CRUI CARE Agreement