UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

CORSO DI LAUREA IN INGEGNERIA INFORMATICA

"IMAGE AUGMENTATION TECHNIQUES FOR CONVOLUTIONAL NEURAL NETWORKS"

Relatore: Prof. Nanni Loris

Laureando: Bravin Riccardo

ANNO ACCADEMICO 2021 – 2022

Data di laurea 21/07/2022

Abstract

ENG

One of the biggest challenges for Convolutional Neural Networks (CNNs), now that they are vastly used in many contexts, is the lack of adequate training sets for robust training sessions and no overfitting. In some cases, it can be difficult to gather enough data for hard classification tasks in the medical field and similar, due to the lack of cases to document or the vast knowledge needed in order to correctly classify the images in a dataset. In the last few years for these reasons data augmentation, rather than other methods, has been under the spotlight mostly thanks to the simplicity and effectiveness that simple methods, such as rotation and flipping for images or random noise added to other sources, have. Here we want to propose various classic and newer methods compared using ResNet50 and MobileNetV2 on thirteen different datasets. Results show that ensembles built combining a variety of the analyzed methods can achieve state of the art performance or even surpass the best-known approaches on various tasks.

IT

Una delle più grandi sfide per le Reti Neurali Convoluzionali, soprattuto ora che vengono utilizzate ampiamente in svariati contesti, è la mancanza di training set adeguati per sessioni di training robuste e meno prone ad overfitting. In certi casi può risultare difficile raccogliere dati a sufficienza per complessi compiti di classificazione in campi quali quello medico e simili a causa dell'assenza di casi di cui sia possibile effettuare un campionamento o delle vaste conoscenze necessarie per la corretta classificazione dei dati ottenuti. Negli ultimi anni proprio per queste ragioni il data augmentation, differentemente da altri metodi, ha avuto particolare successo, principalmente grazie alla semplicità ed efficacia che metodi semplici quali rotazione o specchiamento per le immagini o rumore gaussiano aggiunto ad altre sorgenti riescono ad ottenere. Vogliamo qui proporre vari metodi, classici e di nuova ideazione, comparati utilizzando ResNet50 e MobileNetV2 su tredici diversi dataset. I risultati mostrano che ensemble costruiti combinando i vari metodi analizzati riescono ad ottenere risultati che possono superare perfino i migliori approcci specifici nei diversi dataset.

1. Introduction

Convolutional neural networks are neural network that leverage the mathematical concept of convolution. Before this discovery, forward neural networks were the main method used for image classification, but their inability to perceive relations in bigger pixel clusters or the dependency of information to a specific position in the input array greatly reduced their effectiveness.

Instead, the convolutional capabilities of CNNs allow, inside it, to progressively reduce the size of the input image expanding its depth and doing so extracting different features independently of their position thanks to the learnable parameters of the convolutional kernels. Therefore, their flexibility and robustness has made it so that they have become the new classification paradigm for image classification. Apart from image classification, these neural networks have been used also for natural language processing and speech recognition.

In general, CNNs need vast, labeled datasets to achieve acceptable results in classification problems and for that the human intervention is needed. Therefore, the benefits of not having to manually determine and use the best feature extraction methods from the images have been opposed by the ever-growing need of human labor to label an extensive number of data that the CNN can learn from.

A partial solution to this problem has been the usage of data augmentation, with which it is possible to generate new datapoints from existing ones, reducing the probability of overfitting for small datasets or more complex networks and generally improving the real accuracy. This technique, in conjunction with others like transfer learning, batch normalization and dropout layers, has allowed previously impossible automatizations in medical and neurobiological fields to be performed even with limited resources and time.

Here we want to set the focus on a specific type of data augmentation: image augmentation for CNNs. There are several types of techniques already proven to provide significant benefits to the training of CNNs varying in three macro groups defined in [1]: Model-free, which uses single or multiple images without any conditional treatment; Model-based, which uses information about the image and his label to apply transformations; and Optimizing policy-based, which are methods like reinforcement learning and adversarial learning. Model-free and model-based approaches are the least computationally expensive and the ones we are going to focus on here, in combination with the construction of ensembles that can outperform more classical methods of image augmentation.

The remainder of this paper is organized as follows: Section 2. reviews different related image processing approaches from literature. Section 3. Makes a rundown of thirteen classic augmentation methods and eleven newly proposed ones with the addition of five combinations of them. In section 4. and 5. the datasets used, and the results of the methods trained on them with two different topologies are reported accompanied by the presentation of four different ensembles and their relative performances. The best reported methods in this paper manage to reach and surpass state of the art literature results in all tested datasets. In section 6. conclusions are simply drawn from the reported data and discussed.

The MATLAB source code used for every test reported here is published and available free of charge at github.com/RiccardoBravin/NN_image_augmentation

2. Related work

The scope of this paper, as presented before, is to focus on the construction of ensembles with CNNs trained on different types of datasets where image augmentation methods have been applied. In [1], the different augmentation methods are divided into three categories: 1) Model-free, 2) Model-based and 3) Optimizing policy-based. The majority of the algorithms for augmentation are simple to implement but by applying the wrong one to an image, data that no longer belongs to the original class can be produced.

Geometrical transformations, like flipping, rotation and translations are the easiest ones to perform and generally the most effective as reported in [2]. Alongside these three augmentations, it usually also appears random cropping, which reduces the size of the image partially cutting out the outer section of it, and distortion, which, utilizing a mask, shifts the pixels to new positions interpolating their color to get uniform distortion. As tested by [3] on AlexNet, trained on ImageNet and CIFRAR10, generally rotation tends to work better than translation and random cropping.

Another method through which image transformation is achieved is color variation. One way of performing it is by changing the color space of the image and doing so removing the illumination bias that can come through in a dataset as proposed in [2]. Other methods involve adding noise to the color of the image through transformations or by jittering the various values like hue, saturation and brightness like in [3]. Furthermore, convolutions with various filters for blurring, sharpening or swapping can be applied to the images to produce new ones with different characteristics or, as usually happens in the real world, with random erasing and cutting to occlude portions of the images presenting only partially the subject.

However, some of these augmentations can run the risk of removing valuable information from the image or introduce not needed additional complexity to the image drastically reducing the performance of the trained CNN.

Model based augmentations focus on the label knowledge to generate new data through, for example, the combination of same label images utilizing an alpha mask as tested in [4] or alternatively through an intermediate transform and inverse as shown in [5].

Optimizing policy-based make use of GANs to perform style mix, style transfer or generation of completely new images like in [6].

3. Materials and methods

In this work we are going to use pretrained instances of ResNet50 and MobileNetV2 with a batch size of 20 and learning rate of 0.001 and thus applying the concept of transfer learning to improve the training stability and overall accuracy. In the testing phase, each unknown sample is classified by the CNNs, and the resulting scores are fused by sum rule.

In the next section we are going to explore and explain all the different implementations of data augmentation tested. AUG1 through 12 are literature found methods with tested efficacy, the last ten methods are instead, to our knowledge, proposed here for the first time and more deeply explained through images (see Figure 1. and Figure 2.) and pseudocode due to the lack of previous documentation.

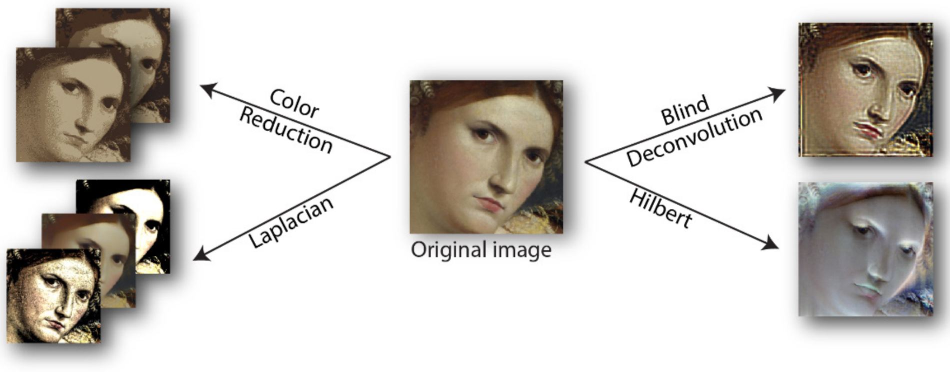

Figure 1. Visual representation of the four directly applied methods

AUG1 (3x):

Applies to each image three transforms: a mirroring on the x axis, a mirroring on the y axis and a linear scale on both axis using a random value derived from the interval [1,2].

AUG2 (6x):

Applies the same transformations of the AUG1 and additionally performs: random image rotation in the range [-10°,10°], translation with a random vector of maximum magnitude 5 pixels and shear on both vertical and horizontal axis with values randomly taken from the interval [0,30] degrees.

AUG3 (5x):

Performs the same augmentations as AUG2 but it removes the shearing.

AUG4 (3x):

The method is the one proposed in [7] where a transformation based on PCA is applied and the values obtained are treated with three different methods before the inverse is used to produce the new images. The first one zeroes out the elements of the feature vector with a probability of 50%, the second adds gaussian noise to every value utilizing the standard deviation of the transformed image, and the third one swaps with a probability of 5% values from the original feature vector with the ones of 5 random transformed images of the same class.

AUG5 (3x):

The same methods utilized in AUG4 are instead applied to images who have been transformed through DCT.

AUG6 (3x):

The chosen image goes through three methods which alter color, contrast or sharpness. Color is altered by adding to each channel a random value clipping the results between 0 and 255; contrast is modified scaling it from [0.3,0.7] back to [0,1] effectively saturating the 30% highest and lowest intensities of the image; sharpness is increased by subtracting from the original image a blurred (with a gaussian filter) version of itself.

AUG7 (7x):

Utilizing the MATLAB function jitterColorHSV four images are generated by randomly altering: the hue in the range [0.05,0.15], the contrast in the range [1.2,1.4], the saturation in the range [-0.4, -0.1] and the brightness in the range [-0.3, -0.1]. Two other images are created through the usage of the MATLAB functions imgaussfilt, with a standard deviation of the gaussian filter ranging from 1 to 6, and imsharpen, with a strength of 2. The seventh image is obtained utilizing the color altering method of AUG6.

AUG8 (2x):

This method selects a random image of the same class as the current one and applies two nonlinear mappings: RGB Histogram Specification and Stain Normalization through the Reinhard Method [7], in order to generate the new images.

AUG9 (6x):

Two types of elastic transform, more extensively explained in [8], are applied. One is based upon a preexisting MATLAB method that introduces distortion in the original images and the other is an adaptation of the grayscale ElasticTransform from the python implementation in the computer vision tool Albumentations. Both methods deform the images through a random displacement map generated with values in the uniform distribution [-1,1]. To the resulting matrix is then applied one of three different low pass filters (a circular averaging one, a rotationally symmetric Gaussian one and an approximation of a two-dimensional Laplacian operator) before using it for the distortion of the original image.

AUG10 (3x):

Similarly to AUG4, the method proposed in [8] performs a DWT [9] with Daubechies wavelet which produces 4 matrices: the approximation coefficients (cA) and then the horizontal, vertical, and diagonal (cH, cV, cD). The obtained matrices are then modified with three methods before being transformed back with the inverse DWT. The first method zeroes out random values with a probability of 50%, the second one sums to each value the standard deviation of the original image plus a random value in the range [-0.5, 0.5], and the third one selects 5 random images from the same class as the current one and performs the DWT on them, then each value of the original matrices is swapped with a probability of 5% with values from one of the transformed five images.

AUG11 (3x):

Based upon the Constant-Q Transform (CQT), this method, first proposed in [8], performs the CQT on the image to then apply the same alterations as AUG10 to the resulting arrays before doing the inverse CQT in order to get back 3 new augmented images.

AUG12 (9x):

This method combines parts of AUG3 with AUG7. More specifically it performs: vertical and horizontal flips, random rotations in the range [1°, 180°], gaussian noise addition based on the standard deviation of the image, cropping of a random number of pixel in the range [0, 20] from every side and hue, saturation, brightness and contrast jittering with random values in the ranges, respectively, [0.05, 0.15], [-0.4, -0.1], [-0.3, -0.1], [1.2, 1.4].

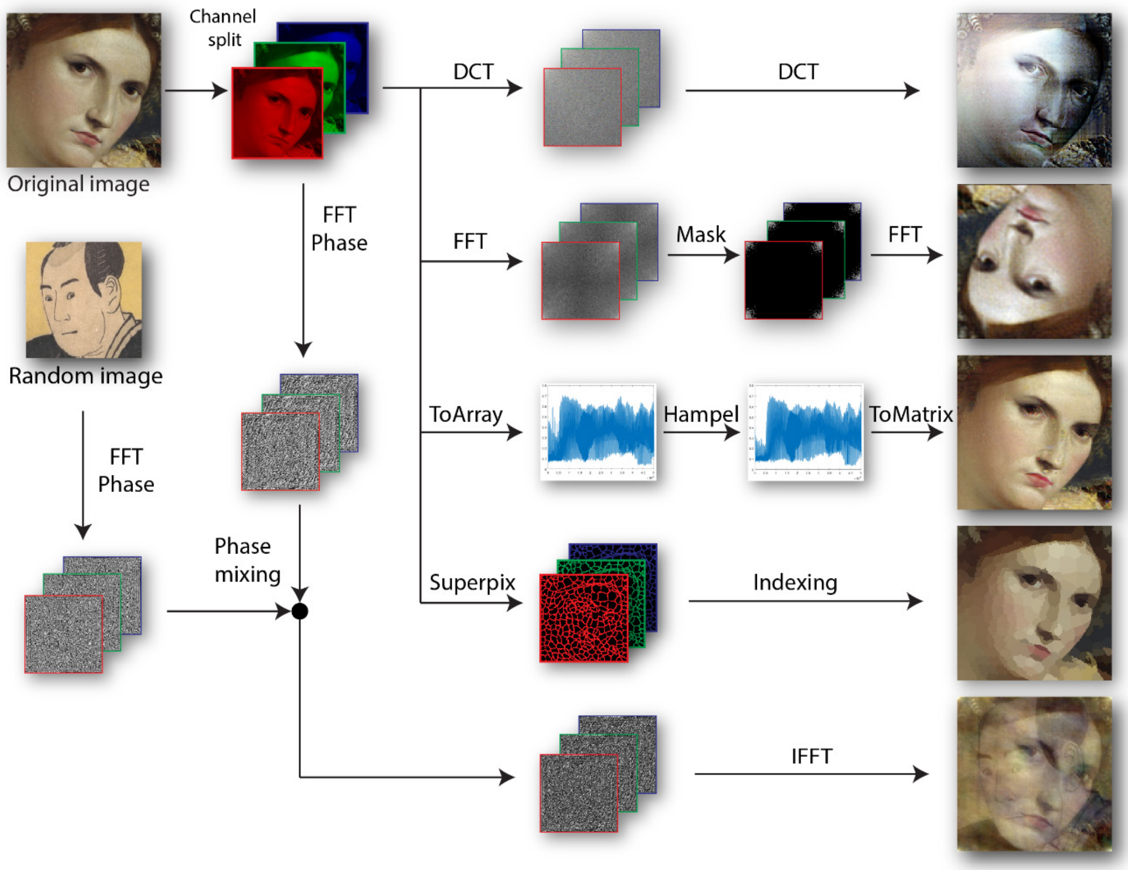

Figure 2. Depiction of the more complex methods processing

Superpixel (1x):

This method, first presented in [10], utilizes a SuperPixel segmentation mask generation through the MATLAB superpixels function which returns a matrix with random number of segmentations between 300 and 2000. The obtained mask is then used to calculate the mean of each chunk and that value is used to replace each pixel of it. An example of pseudocode is here reported:

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

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

Color reduction (2x):

Making use of a color space transformation called color indexing, a reduction of the number of colors available to represent the image is performed. In MATLAB the function rgb2ind can receive a parameter for how many colors the final image can retain. In this way, the number of colors the image keeps is randomly chosen in the range [8,16]. A second image is obtained specifying the use of dithering: a technique that leverages the way the human eye perceives colors to generate depth, while having a limited color palette, utilizing a rapid change of the values of neighboring pixels. An example of pseudocode is here reported:

```
[IND,map] = rgb2ind(OriginalImage, randomInt(4,8),'nodither');
ColorReducedImage = ind2rgb(IND,map);

[IND,map] = rgb2ind(OriginalImage, randomInt(4,8));
DitheredImage = ind2rgb(IND,map);
```

DCT (1x):

The discrete cosine transform (DCT) is applied to the image and to the resulting matrices, instead of applying the inverse, the normal DCT is once again used to obtaining a new distorted image that goes through some post processing for better readability. An example of pseudocode is here reported:

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

FFT (1x):

The FFT is applied to the current image and all matrices' values in the interval between the average standard deviation of the matrix's rows and its respective inverse are set to zero. To the result is then applied the FFT again, instead of the inverse, and only the modulo is kept, obtaining a flipped image with color changes. An example of pseudocode is here reported:

```
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.

FFTCombine (1x):

Based on the concept of image mixing, the FFT transform is applied to the current image and to another random one from the dataset. The phases obtained from the two matrices are then combined keeping the negative values from one and the positive from the other. The complex matrix is then recomposed using the newly obtained phase and the modulo from the original image before the inverse FFT is applied. An example of pseudocode is here reported:

```
FFTImage = FFT(OriginalImage);
FFTrndImage = FFT(RandomImage);
phaseImage1 = angle(FFTImage);
phaseImage2 = angle(FFTrndImage);
```

```
phaseImage1(phaseImage1 > 0) = phaseImage2(phaseImage1 > 0);
RecomposedMatrix = modulo(FFTImage) * exp(i*phaseImage1);
NewImage= modulo(IFFT(FFTImage));
```

Hampel (1x):

This method uses the Hampel outlier remover filter, generally used for signals, to process a vectorized version of the image. The MATLAB function `Hampel()` is used with a measurement window of 20 and a standard deviation of 1.5 for which a sample must differ from the local median to be replaced with. After the process the image is converted back into is matrix form. An example of pseudocode is here reported:

```
LinImg = liearize(OriginalImage);
HampelImg = hampel(LinImg,20,1.5);
NewImg = delinearize(HampelImg, size(OriginalImage));
```

Where linearize(image) returns the transform of each color plane of the image into an array and delinearize(vector, dim) returns the inverse transformation of the linearized image following the dim value.

Hilbert (1x):

The Hilbert transform extracts a discrete time analytic signal from every column of the image in its phase and modulo form. From this data, only the phase is kept to define the new image, after a rescale to fit the data in the 8 bits for color. An example of pseudocode is here reported:

```
HilbertImage = hilbert(OriginalImage);

NewImage = rescale(phase(HilbertImage));
```

Laplacian (1x):

Utilizing the local Laplacian filter MATLAB function (contrast enhancement with edge awareness) three different images are generated: one with a high increase of small details contrast, one with a smoothing of small details and one with an overall increase of dynamic range and contrast.

Deconvolution (1x):

Through the use of a blind deconvolution algorithm, usually used for its capability of deblurring images when the point spread function is known, we perform the opposite operation of what the convolutional layer of a CNN would do. The MATLAB function `deconvblind()` requires a starting estimate of the point-spread function which is generated using random gaussian values in a matrix of randomly chosen size in the range [4,8].

SVD (1x):

Making use of the Singular Value Decomposition (SVD) each color plane of the original image is decomposed into three matrices: U and V, complex and unitary and S, diagonal and with non-negative real numbers. Of the three, just the diagonal one undergoes a transformation which sets to zero every value lower than the highest divided by a random integer in the range [50,100]. The three matrices are then multiplied to obtain back the original image with an interwoven fiber like noise pattern and partial degradation. An example of pseudocode is here reported:

```
[U,S,V] = svd(OriginalImage);
S_mask = S < (max(S)/rand(50,100));
S(S_mask) = 0;
NewImage = U*S*V^T;
```

We also adopted a technique that combines different, and as independent as possible, augmentation methods to train one single CNN. Therefore, we were able to increase singular network performance without lengthening the inference time like for ensembles. Here are thus presented the mentioned methods.

MixDA_1: This method makes use of the augmentations DCT, FFT, FFTCombine, SVD, Laplacian, Superpixel, ColorReduction, Hilbert and a reduction of AUG9 with just the MATLAB technique to generate a total of 9 images for each one in the datasets.

MixDA_2: This method makes use of the augmentations DCT and a reduction of AUG9 with just the MATLAB technique to generate a total of 2 images for each one in the datasets.

MixDA_3: This method makes use of the augmentations DCT, Superpixel, Hilbert e Hampel to generate a total of 4 images for each one in the datasets.

MixDA_4: This method makes use of the augmentations Hampel, Hilbert, DCT and Laplacian to generate a total of 6 images for each one in the datasets.

MixDA_5: This method makes use of the augmentations AUG12, FFT, Hilbert and Hampel to generate a total of 13 images for each one in the datasets.

4. Dataset

Every test of the described augmentations methods is performed on thirteen benchmark datasets in order to better represent the performance in real world scenarios and be better compared to previous literature results.

Table 1. *Description of the datasets*

Abbreviation	Full Name	#Classes	#Samples	Image Size	Protocol*	Ref
VIR	Virus	15	1500	41×41	10-FCV	[11]
BARK	Bark	23	23000	~1600×3800	5-FCV	[12]
POR	Portraits	6	927	From 80×80 to 2700×2700	10-FCV	[13]
PBC	Peripheral blood cell classification	8	17092	360x363	Tr-Te	[14]
HE	2D HELA	10	862	512×382	5-FCV	[15]
MA	Muscle aging	4	237	1600×1200	5-FCV	[16]
BG	Breast grading carcinoma	3	300	1280×960	5-FCV	[17]
LAR	Laryngeal dataset	4	1320	1280×960	3-FCV	[18]
Triz	Gastric lesion types	4	574	352×240	10-FCV	[19]
END	Histopathological endometrium images	4	3502	640×480	Tr-Te	[20]
RSMAS	Coral textures	14	766	256x256	5-FCV	[21]
PEST	Pest identification dataset	10	563	640x480	5-FCV	[22]
InfL	Squamous cell carcinoma	4	720	1920x1072	5-FCV	[23]

*<k>-FCV stands for a k-fold cross validation method, "Tr-Te" instead means that the Training and Test sets were provided by the authors.

In Table 1. For each dataset used different labels are reported: an acronym, from now on used to refer to it; the original name, if reported in the reference literature; the number of classes and total samples it contains; the resolution of the given images; the protocol used for testing; and the original reference paper.

5. Experimental results

Firstly, we are going to analyze the various single methods and the literature presented ones against the various ResNet50 instances (Baseline) trained with the default datasets (see Table.2). The ten single newly presented methods are not reported in Table .2 due to their poor performances when not combined. Then a series of ensembles, made with ResNet50 and MobileNetV2, that make use of the different presented augmentations is reviewed. The results of three state of the art augmentation methods, from the paper [24], are applied to the same datasets and reported with the acronym Ext_<i>.

Table 2. *Performance accuracy (in %) of the literature augmentations and the newly proposed ones*

DataAUG	VIR	HE	MA	BG	LAR	POR	Bark	TriZ	END	PBC	RSMAS	PEST	InfL
Baseline	86.60	95.81	95.83	94.00	94.55	87.16	89.91	98.78	50.00	99.03	99.22	93.70	95.56
Aug1	87.00	95.12	95.00	93.00	92.95	87.05	89.60	99.13	56.00	98.93	98.69	92.76	95.56
Aug2	86.87	**96.63**	95.83	94.00	95.08	85.97	90.17	99.13	51.50	**99.08**	99.35	93.26	94.58
Aug3	87.80	95.12	95.00	94.00	94.55	87.05	89.45	98.96	56.50	98.88	99.09	93.65	95.42
Aug4	86.33	95.23	93.33	92.33	94.62	84.90	87.91	98.08	75.00	98.74	98.30	91.11	95.56
Aug5	86.00	95.35	91.25	91.33	95.45	86.41	87.61	98.43	77.50	98.74	98.17	89.94	**96.25**
Aug6	**	**	**	92.33	94.39	87.37	88.63	98.43	75.00	98.78	98.82	93.15	94.31
Aug7	**	**	**	93.33	95.08	88.13	89.28	98.61	81.00	98.98	99.09	92.93	94.44
Aug8	**	**	**	90.67	94.70	86.06	87.29	98.26	80.50	98.35	97.91	90.88	93.61
Aug9	85.67	95.58	94.17	91.67	95.15	86.19	88.86	98.95	69.50	98.93	98.30	92.82	95.00
Aug10	84.20	95.81	91.25	88.67	93.64	85.10	86.39	99.31	62.00	98.64	98.04	90.22	94.86
Aug11	85.47	95.35	91.25	92.67	95.98	86.71	89.20	**99.48**	80.00	98.69	98.82	92.32	95.69
Aug12	88.93	95.81	97.92	**94.67**	**96.36**	**89.00**	89.45	98.42	74.5	**99.08**	***	***	***
Ext_1	86.73	95.23	91.67	90.33	95.45	85.63	86.81	97.90	71.00	98.78	97.91	89.83	95.00
Ext_2	86.20	94.77	92.92	91.67	95.15	85.76	88.40	98.08	75.00	98.69	99.09	91.77	94.44
Ext_3	85.27	95.47	91.25	93.33	93.71	87.26	87.84	98.43	**84.00**	98.83	97.65	90.99	95.14
MixDA_1	**89.33**	96.28	**98.33**	92.67	95.30	88.46	**90.63**	98.61	81.50	98.93	**99.48**	**93.98**	94.72
MixDA_2	86.33	94.88	94.17	91.67	95.23	86.39	89.15	98.95	67.00	98.88	97.91	93.26	95.28
MixDA_3	85.93	93.72	92.08	92.67	95.08	85.74	88.60	98.96	76.50	98.74	98.69	92.60	94.86
MixDA_4	84.73	94.88	92.08	92.00	95.23	87.59	89.12	98.61	69.50	98.88	98.56	93.65	94.72
MixDA_5	86.40	93.60	94.17	92.00	95.30	87.04	89.90	98.95	65.50	98.88	99.35	93.37	94.17

**Augmentations that work on color images were not run on gray-level images.
*** Computation time exceeded resources.

From (Table.2) different conclusions can be drawn:

From the literature augmentations no particular one seems to consistently outperform the baseline network by a substantial amount. The p-values in fact fluctuate between 1 and 0.08 without a clear relation on the type of transformation performed and the efficacy of it.

Among the newly proposed methods, MixDA_1 seems to be the only one with an average accuracy higher than every other one proposed and could thus be considered the best performing single augmentation proposed. This assertion can be further confirmed by the p-value of around 0.02, in relation to the baseline accuracy, which is about half of the lowest literature method.

Now we present the different approaches taken for the creation of the ensembles. We specify that since the AUG6 through AUG8 only work on color images, the ensembles for VIR, HE, MA simply do not use those scores for the fusion and that each ensemble has the relative network scores combined through fusion by sum rule.

EnsDA_1: Combines AUG1 through AUG11 scores for an ensemble of eleven classifiers.

EnsDA_2: Combines MixDA_1 through MixDA_5, AUG1, AUG2 and AUG5-11 scores for an ensemble of fourteen classifiers.

EnsDA_3: Combines MixDA_1, MixDA_4, MixDA_5, AUG2, AUG4, AUG5-7 AUG9-11 scores for an ensemble of eleven classifiers.

EnsDA_4: Combines MixDA_4, MixDA_5, AUG6, AUG8 and AUG9 scores for an ensemble of five classifiers (due to the low number of classifiers AUG6 to AUG8 were not removed for the three black and white datasets).

Ext_Ens: This value corresponds to the maximum values obtained in [24] utilizing the different ensembles tested in the mentioned paper.

Three ensembles, **Ens_Base(*x*)**, obtained through the use of *x* ResNet50 instances trained on the Baseline method and combined with the average sum rule are also reported as a point of reference for the new methods.

Table 3. *Performance accuracy (in %) of literature and new ensembles trained on ResNet50*

DataAUG	VIR	HE	MA	BG	LAR	POR	Bark	TriZ	END	PBC	RSMAS	PEST	InfL
Baseline	85.53	95.93	95.83	92.67	94.77	86.29	87.48	98.97	55.50	98.98	99.22	93.7	95.56
Ext_Ens	**90.20**	96.63	97.08	**94.00**	**96.74**	**90.07**	91.38	99.13	77.50	**99.12**	99.22	94.14	**96.81**
EnsDA_1	90.00	96.51	97.08	94.00	96.29	89.21	91.27	99.13	76.00	98.98	99.22	93.98	96.53
EnsDA_2	89.87	**97.44**	**98.33**	**94.00**	96.59	90.06	**91.60**	**99.30**	77.50	**99.12**	99.48	94.42	96.11
EnsDA_3	89.60	**97.44**	97.08	93.67	96.36	89.95	91.46	99.13	77.50	99.08	**99.61**	**94.59**	95.97
EnsDA_4	88.47	96.40	97.50	93.67	96.06	89.20	91.25	**99.30**	**83.00**	98.98	99.48	94.20	96.25
Ens_Base(14)	89.73	96.40	97.50	93.67	96.14	88.02	90.66	98.78	50.50	98.88	93.76	96.25	93.76
Ens_Base(11)	89.73	96.28	97.50	93.67	95.91	87.58	90.67	98.78	50.00	98.74	93.7	96.11	93.7

P-values can be calculated as an effective metric to show how different the distribution of accuracy is in regard to the most affine Ens_Base(x). The ensembles of [24], here condensed in Ext_Ens, reach average accuracies comparable to the ones obtained by our methods but with p-values as low as 0.03, the here proposed new ensembles instead reach, with EnsDA_2, a p-value of 0.001 when compared to Ens_Base(14) (see Table 3.). Also, when comparing EnsDA_1 with EnsDA_2, even if the average accuracies differ just by 0.4, the p-value between the two is about 0.01 which signifies a notable independence.

Even though accuracy is the most generally used performance indicator due to its simplicity, it is not the most accurate metric for the comparison of ensembles due to its inherit dependency on the chosen classification threshold. The ROC (Receiver Operating Characteristic curve) is the curve defined by the relation between true positive rate and false positive rate and thus can better represent the differences between classifiers. The AUC is a way to condense the ROC results into one single and manageable value. Here we are going to use the One's complement of the AUC in percentage form. Since the AUC is a metric for specifically binary classifiers, in multiclass problems, we decided to use the One-vs-All approach with the MATLAB rocmetrics functions.

Table 4. *EOC values (in %) of the ensembles trained on ResNet50*

DataAUG	VIR	HE	MA	BG	LAR	POR	Bark	TriZ	END	PBC	RSMAS	PEST	InfL
Baseline	2.13	0.40	0.79	2.74	0.41	2.69	1.87	0.10	23.67	0.03	0.01	0.75	0.54
Ext_Ens	1.33	0.20	**0.11**	2.50	**0.11**	1.68	1.37	0.05	9.24	**0.01**	**0.00**	0.54	0.48
EnsDA_1	1.37	0.24	0.27	2.39	0.12	1.75	1.38	0.04	10.73	**0.01**	0.01	0.75	0.54
EnsDA_2	**0.68**	**0.13**	0.21	1.91	0.15	**1.10**	**0.43**	**0.03**	5.18	**0.01**	**0.00**	**0.30**	0.38
EnsDA_3	0.72	**0.13**	0.25	2.26	0.17	1.12	0.45	0.04	5.07	0.02	**0.00**	0.31	**0.35**
EnsDA_4	0.84	0.16	0.16	**1.59**	0.29	1.17	0.48	**0.03**	**3.29**	**0.01**	**0.00**	0.32	**0.35**
Ens_Base(14)	1.36	0.23	0.21	2.42	0.12	2.45	1.56	0.13	20.48	0.03	**0.00**	0.71	0.48
Ens_Base(11)	1.37	0.27	0.23	2.38	0.15	2.48	1.55	0.12	20.94	0.03	0.01	0.71	0.50

As a confirmation of the previously reported results, we can see (see Table 4.) that the lowest EOC values are mostly obtained by the ensembles of our newly proposed augmentations and the average values for all the three new ensembles are noticeably lower than the literature ones. Also, the p-values, when comparing the different classifiers to Ens_Base(14), reaches values as low as 0.005 with EnsDA_2.

For comparison and further confirmation of the previously presented data, in Table .5 are reported the same tables but with each training done with MobileNet: a lightweight convolutional neural network created for mobile devices usage that reaches comparable results to bigger networks.

Table 5. *Performance accuracy (in %) of literature and new ensembles trained on MobileNetV2*

DataAUG	VIR	HE	MA	BG	LAR	POR	Bark	TriZ	END	PBC	RSMAS	PEST	InfL
Baseline	42.93	94.58	94.58	91.33	92.80	84.45	89.79	97.91	74.00	98.88	97.78	91.77	94.72
Ext_Ens	84.47	96.63	96.25	93.33	95.98	88.55	91.56	98.26	87.00	99.22	98.69	93.31	95.69
EnsDA_1	85.27	96.16	95.83	93.00	96.21	88.56	91.20	98.25	86.00	99.17	98.69	92.98	95.83
EnsDA_2	88.53	96.86	97.08	92.33	96.21	89.31	**91.63**	98.43	86.50	99.22	98.69	93.70	95.69
EnsDA_3	**88.73**	**97.09**	97.08	**93.33**	**96.36**	**89.64**	91.55	**98.78**	**87.50**	99.37	**99.22**	94.09	95.69
EnsDA_4	86.60	96.86	**97.50**	93.00	95.83	88.55	91.26	98.43	85.50	99.17	98.82	**94.36**	94.31
Ens_Base(14)	47.60	95.00	97.50	92.67	95.30	85.96	91.04	98.26	83.00	**99.37**	98.69	92.82	95.83
Ens_Base(11)	47.60	95.00	97.08	92.67	94.85	85.86	91.10	98.26	82.50	99.27	98.69	92.71	**95.97**

As can be seen from Table .5 the data reported indicates that the proposed methods are capable of surpassing both the baseline accuracy and Ens_Base(14) this time with even lower p-values and higher average accuracies. While the methods from [24] reach p-values as low as 0.08, when compared to Ens_Base(14), ours range from 0.04 to even 0.003 with a performance increase of almost four percentage points in average accuracy.

Table 6. *EOC values (in %) of the ensembles trained on MobileNetV2*

DataAUG	VIR	HE	MA	BG	LAR	POR	Bark	TriZ	END	PBC	RSMAS	PEST	InfL
Baseline	23.59	0.45	0.82	2.61	0.56	3.33	1.74	0.14	13.38	0.04	0.17	0.98	0.67
Ext_Ens	2.24	0.17	0.14	2.77	**0.20**	1.96	1.24	0.06	4.63	**0.01**	0.02	0.69	0.55
EnsDA_1	2.35	0.17	0.20	3.06	0.24	2.04	1.26	0.06	4.79	0.02	0.02	0.69	0.61
EnsDA_2	0.94	**0.15**	**0.13**	2.56	0.30	1.23	**0.45**	0.05	3.29	**0.01**	**0.01**	0.41	0.46
EnsDA_3	**0.85**	0.17	0.16	**2.27**	0.29	**1.20**	0.46	**0.04**	**2.83**	**0.01**	**0.01**	0.44	**0.41**
EnsDA_4	1.25	0.23	0.15	2.43	0.46	1.38	0.52	0.07	3.23	**0.01**	**0.01**	**0.40**	0.49
Ens_Base(14)	17.46	0.24	0.19	2.32	0.28	2.74	1.34	0.17	6.72	**0.01**	0.03	0.84	0.65
Ens_Base(11)	17.56	0.24	0.22	2.36	0.32	2.75	1.36	0.17	6.96	0.01	0.03	0.84	0.65

The values obtained in Table .6 show not that big of a gap as for ResNet50 between the results reported in [24] and ours. With similar average p-values for both ours and [24] methods, when compared to Ens_base(14), but lower EOC averages for EnsDA_2, EnsDa_3 and EnsDA_4 we can say that our methods still outperform literature ones even in smaller CNN topologies. A special focus should be given to EnsDA_3 which manages to outperform even EnsDA_4 by 0.6 with also a p-value of 0.0007 when compared to Ens_Base(14).

In the last table (see Table.7), we decided to report a one-by-one comparison between the best performing methods found in literature and our best average performing method (EnsDA_4) for each dataset.

Table 7. *Comparison of accuracy (in %) with best known literature results*

DATASET	ResNet50	MobileNetV2								
VIR	EnsDA_2	EnsDA_2	[25]	[26]	[27]	[28]	[29]	[11]	[28]	[30]
	89.87	88.53	89.60	89.47	89.00	88.00	87.27	87.00***	86.20	85.70
LAR**	EnsDA_2	EnsDA_2	[31]	[18]						
	96.59	96.21	95.2	92.0						
POR	EnsDA_2	EnsDA_2	[13]							
	90.06	89.31	**90.08**							
BARK	EnsDA_2	EnsDA_2	[32]	[33]	[34]	[12]				
	91.60	**91.63**	48.90	85.00	90.40	85.00				
PBC	EnsDA_2	EnsDA_2	[35]	[36]						
	99.12	99.22	**99.30**	97.94						
Triz	EnsDA_2	EnsDA_2	[19]							
	99.30	98.43	87.00							
END	EnsDA_2	EnsDA_2	[20]							
	77.50	**86.50**	76.91							
HE	EnsDA_2	EnsDA_2	[37]	[38]	[17]	[39]				
	97.44	96.86	**98.30**	94.40	84.00	68.30				
MA	EnsDA_2	EnsDA_2	[37]	[40]	[39]					
	98.33	97.08	97.90	53.00	89.60					
BG	EnsDA_2	EnsDA_2	[17]	[12]						
	94.00	92.33	**96.30**	95.00						
RSMAS	EnsDA_2	EnsDA_2	[21]							
	99.48	98.69	97.75							
PEST	EnsDA_2	EnsDA_2	[22]							
	94.42	93.70	**94.86**							
InfL	EnsDA_2	EnsDA_2	[23]							
	96.11	95.69	95.30							

6. Conclusions

This paper aimed to find new image manipulation methods that could outperform more traditional ones. Through the training of different CNN topologies, finetuned with the various augmentations, we compared the performances of single and combined networks on thirteen different benchmark datasets to better represent the diverse classification tasks that could be encountered in real world scenarios.

This study shows that combining different augmentation methods in one single training can drastically increase performance of single instance CNNs as could be seen from MixDA_1 and EnsDA_4 that outperforms even Ens_Base(14) thus giving the possibility of having low inference time networks with high accuracy. Furthermore, the construction of ensembles with different networks diversified on the data level can achieve even greater results and robustness thanks to the independence of the proposed methods from one another. The augmentations here proposed, since they were tested on a plethora of different datasets and networks, should work on most image problems.

References

[1] M. Xu, S. Yoon, A. Fuentes and D. S. Park, "A Comprehensive Survey of Image Augmentation Techniques for Deep Learning," April 2022.

[2] C. Shorten and T. M. Khoshgoftaar, "A survey on Image Data Augmentation," *Journal of Big Data,* pp. 1-48, 6 july 2019.

[3] J. Shijie, W. Ping, J. Peiyi and H. Siping, "Research on data augmentation for image classification based on convolution neural networks," *2017 Chinese Automation Congress (CAC),* pp. 4165-4170, October 2017.

[4] H. Inoue, "Data Augmentation by Pairing Samples for Images Classification," *arXiv,* 11 April 2018.

[5] 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," *ICLR 2020,* pp. 1-15, 17 February 2020.

[6] D. Liang, F. Yang, T. Zhang and P. Yang, "Understanding Mixup Training Methods," *IEEE Access,* pp. 1-10, 31 October 2018.

[7] L. Nanni, S. Brahnam, S. Ghidoni and G. Maguolo, "General Purpose (GenP) Bioimage Ensemble of Handcrafted and Learned Features with Data Augmentation," *ArXiv,* pp. 1-20, 17 April 2019.

[8] L. Nanni, M. Paci, S. Brahnam and A. Lumini, "Comparison of Different Image Data Augmentation Approaches," *Journal of Imaging,* pp. 1-13, 27 November 2021.

[9] D. Ingrid, Ten Lectures on Wavelets, BMS-NSF Regional Conference Series in Applied Mathematics, 1992.

[10] K. Hammoudi, A. Cabani, B. Slika, H. Benhabiles, F. Dornaika and M. Melkemi, "SuperpixelGridCut, SuperpixelGridMean and SuperpixelGridMix Data Augmentation," *arXiv,* pp. 1-9, 11 April 2022.

[11] G. Kylberg, M. Uppström and I. M. Sintorn, "Virus Texture Analysis Using Local Binary Patterns and Radial Density Profiles," *18th Iberoamerican Congress on Pattern Recognition (CIARP),* pp. 573-580, 2011.

[12] M. Carpentier, P. Giguère and J. Gaudreault, "Tree Species Identification from Bark Images Using Convolutional Neural Networks," *IEEE/RSJ International Conference on Intelligent Robots and Systems,* pp. 1075-1081, 2018.

[13] S. Liu, J. Yang, S. S. Agaian and C. Yuan, "Novel features for art movement classification of portrait paintings," *Image and Vision Computing,* pp. 104-121, 01 04 2021.

[14] 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 in Brief,* pp. 105-474, 01 06 2020.

[15] 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,* pp. 1213-1223, 2001.

[16] L. Shamir, N. V. Orlov, D. M. Eckley and I. Goldberg, "IICBU 2008: a proposed benchmark suite for biological image analysis," *Medical & Biological Engineering & Computing,* p. 943–947, 2008.

[17] K. Dimitropoulos, P. Barmpoutis, C. Zioga, A. Kamas, K. Patsiaoura and N. Grammalidis, "Grading of invasive breast carcinoma through Grassmannian VLAD encoding," *PLoS ONE,* p. 1–18, 2017.

[18] S. Moccia, "Confident texture-based laryngeal tissue classification for early stage diagnosis support," *Journal of Medical Imaging,* p. 2017.

[19] R. Zhao, "TriZ-a rotation-tolerant image feature and its application in endoscope-based disease diagnosis," *Computers in biology and medicine,* pp. 182-190, 2018.

[20] 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 Journal of Biomedical and Health Informatics,* pp. 1664-1676, 2020.

[21] A. Gòmez-Rìos, 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 Systems With Applications,* pp. 1-45, 2018.

[22] L. Nanni, A. Manfè, G. Maguolo, L. A. and S. Brahnam, "High performing ensemble of convolutional neural networks for insect pest image detection," *Elsevier,* pp. 1-12, 2022.

[23] I. Patrini, M. Ruperti, S. Moccia, L. S. Mattos, E. Frontoni and E. DeMomi, "Transfer learning for informative-frame selection in laryngoscopic videos through learned features," *Medical & Biological Engineering & Computing,* pp. 1-14, 2020.

[24] N. L., P. M., S. Brahnam and L. A., "Feature transforms for image data augmentation," *arXiv,* 24 January 2022.

[25] L. Nanni, S. Ghidoni and S. Brahnam, "Deep features for training support vector machines," *Journal of Imaging,* p. 177, 2021.

[26] L. Nanni, E. D. Luca and M. L. Facin, "Deep learning and hand-crafted features for virus image classification," *J. Imaging,* p. 143, 2020.

[27] A. R. Geus, A. R. Backes and J. R. Souza, "Variability Evaluation of CNNs using Cross-validation on Viruses Images," *VISIGRAPP,* vol. 2020, pp. 626-632.

[28] Z.-j. Wen, Z. Liu, Y. Zong and B. Li, "Latent Local Feature Extraction for Low-Resolution Virus Image Classification," *Journal of the Operations Research Society of China,* pp. 117-132, 2020.

[29] A. R. B. a. J. J. M. S. Junior, "Virus Classification by Using a Fusion of Texture Analysis Methods," *2020 International Conference on Systems, Signals and Image Processing (IWSSIP),* pp. 290-295, 2020.

[30] F. L. C. d. Santosa, M. Paci, L. Nanni, S. Brahnam and J. Hyttinen, "Computer vision for virus image classification," *Bi-osystems Engineering,* pp. 11-22, 2015.

[31] L. Nanni, S. Ghidoni and S. Brahnam, "Ensemble of Convolutional Neural Networks for Bioimage Classification," *Applied Computing and Informatics,* pp. 19-35, 2021.

[32] S. Boudra, I. Yahiaoui and A. Behloul, "A set of statistical radial binary patterns for tree species identification based on bark images," *Multimedia Tools and Applications,* pp. 22373-22404, 2021.

[33] V. Remeš and M. Haindl, "Bark recognition using novel rotationally invariant multispectral textural features," *Pattern Recognit Lett,* p. 2019, 612-617.

[34] V. Remes and M. Haindl, "Rotationally Invariant Bark Recognition," *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition,* pp. 22-31, 2018.

[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," *Computer methods and programs in biomedicine,* 2021.

[36] F. Ucar, "Deep Learning Approach to Cell Classificatio in Human Peripheral Blood," *2020 5th International Conference on Computer Science and Engineering,* pp. 383-387, 2020.

[37] 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 Bioinformatics,* 2016.

[38] L. P. Coelho, "Determining the subcellular location of new proteins from microscope images using local features," *Bioinformatics,* 2013.

[39] J. Zhou, S. Lamichhane, G. Sterne, B. Ye and H. Peng, "BIOCAT: a pattern recognition platform for customizable biological image classification and annotation," *BMC Bioinformatics,* 2013.

[40] L. Shamir, N. Orlov, E. D. M., T. J. Macura, J. Johnston and I. G. Goldberg, "Wndchrm - an open source utility for biological image analysis," *Source Code Biol Med,* 2008.

[41] S. Bahaadini, "Machine learning for Gravity Spy: Glitch classification and dataset," *Inf. Sci.,* pp. 172-186, May 2018.

[42] V. Remeš and M. Haindl, "Bark recognition using novel rotationally invariant multispectral textural features," *Pattern Recognit Lett,* pp. 612-617, 2019.