Article

Detection of AI Created Images Using Pixel-wise Feature Extraction and Convolutional Neural Networks

Fernando Martin-Rodriguez * , Rocio Garcia-Mojon , Monica Fernandez-Barciela *

Posted Date: 30 October 2023

doi: 10.20944/preprints202310.0547.v2

Keywords: Artificial intelligence; AI images; photographs, PRNU; ELA; CCN; deep learning

Article

Detection of AI Created Images Using Pixel-wise Feature Extraction and Convolutional Neural Networks

Fernando Martín-Rodriguez [1], Rocio Garcia-Mojon [2] and Monica Fernandez-Barciela [2,*]

[1] atlanTTic research center for Telecommunication Technologies, University of Vigo, fmartin@uvigo.es
[2] atlanTTic research center for Telecommunication Technologies, University of Vigo, rociogarciamojon@uvigo.es
[3] atlanTTic research center for Telecommunication Technologies, University of Vigo, monica.barciela@uvigo.es
* Correspondence: monica.barciela@uvigo.es; Tel.: +34-986-818654

Abstract: Generative AI has gained enormous interest nowadays due to new applications like chatGPT, DALL E, Stable Difussion and Deep Fake. Particularly DALL E, Stable Difussion and others (Adobe Firefly, ImagineArt...) are able to create images from a text prompt and are even able to create photo realistic images. Due to this fact, intense research has arisen to create new image forensics applications able to distinguish between real captured images and videos and artificial ones. Detecting forgeries made with Deep Fake is one of the most researched issues. This paper is about another kind of forgery detection. The purpose of this research aims to detect photo realistic AI created images versus real photos coming from a physical camera. Id Est: making a binary decision over an image asking whether it is artificially or naturally created. Artificial images don't need to try to represent any real object, person or place. For this purpose, techniques that perform a pixel level feature extraction are used. First one is Photo Response Non-Uniformity (PRNU). PRNU is a special noise due to imperfections on the camera sensor that is used for source camera identification. The underlying idea is that AI images will have a different PRNU pattern. Second one is Error level analysis (ELA). This is other type of feature extraction traditionally used for detecting image editions. In fact, ELA is being used nowadays by photographers to detect manually AI created images. Both kinds of features are used to train Convolutional Neural Networks to differentiate between AI images and real photographs. Good results are obtained achieving accuracy rates over 95%. Both extraction methods are carefully assessed by computing precision/recall and F_1-score measurements.

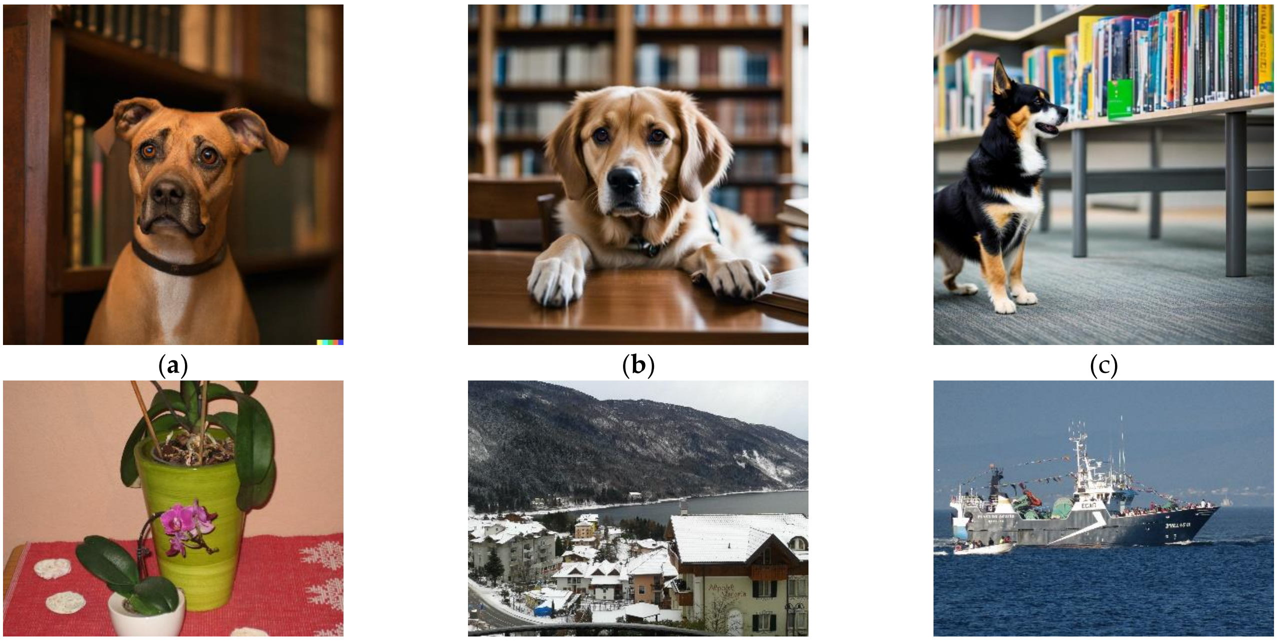
Keywords: Artificial intelligence; AI images; photographs; PRNU; ELA; CCN; deep learning

1. Introduction

Nowadays generative artificial intelligence is one of the top themes of computer engineering research. The emergence of transformers [1,2] as a key tool for generating content has opened a world of new applications where automated systems are able to create productions that, until now, were exclusive of human authorship. Transformers were first used for automated translation systems, where a first processing stage (the encoder) transforms the input text into a numerical representation of text meaning; then a second stage (the decoder) converts (like an inverse transform) those intermediate data into text in another language [3].

Besides neural machine translators, other impressive applications have arisen. Famous chatGPT is a conversational engine created with a decoder transformer [4,5]. Using also transformers, models for translating regular text into images have also been developed. Most known and documented examples of these last ones are DALL E [6,7] and Stable Diffusion [8]. But other examples have quickly been released like: OpenArt [9], ImagineArt [10], Adobe Firefly [11] and many others.

These artificial image generators have reached the point where they may create photo realistic images that can make humans to hesitate if a particular image is really coming from a camera or is an artificial creation. As an example, in Figure 1, three IA created images are presented. They have been created by three different engines: DALL E 2, Stable Diffusion and OpenArt (after testing many applications, these three were found the most appropriate for photorealistic images; other models are good on producing drawings or illustrations, not so much on imitating real photographs). Prompt was the same for the three images: “realistic photo, a portrait of a dog in a library, Sigma 85 mm f/1.4”. Note that details about lens have been added (85 mm focal lens, f/1.4 numeric aperture), this is a common trick used for getting more realistic results. In the same figure, we present also three real photographs that will be processed later. The purpose of this work is making automatically a binary decision between two options: AI-image (fully created AI image) and real image.



(**a**) (**b**) (c)

(**d**) (**e**) (**f**)

Figure 1. (**a**) DALL E 2 image, (**b**) Stable Diffusion image, (**c**) OpenArt image. (**d**), (**e**) and (**f**) real photos.

Other impressive AI application is Deep Fake [12,13]. Deep Fake is able to create photos and videos mixing plausibly information from previous photos and/or videos. For example, creating a video of a person mixing the body from one given individual and the face of another one. The potential danger of this technology being used for fraud or other illegal purposes (defamation, pornography…) has sparked much research on the field of detecting Deep Fake image creation [14]. For example, in [15], Rössler et al start by creating a large dataset of fake videos. In [16], authors exploit what it is, perhaps, the most intuitive method: finding image artifacts that can reveal synthetic content. In [17] a system called “FakeCatcher” is described; this system works relying on biological signals, like the small periodic color variations present in a real face video by cause of the person’s heart rate. A very recent paper by Becattini et al [18] presents a Deep Fake detector based on Head Pose Estimation (HPE). In [19], Bappy et al present a general image forgery detector based on recursive neural networks (LSTM). Almost all publications in this field claim that direct use of neural networks does not produce good results in these kinds of applications. In [20], authors use detection of “convolutional traces”, basing themselves on the fact that AI generated images have passed several convolution stages.

The work described in this paper is similar to Deep Fake detection but with a different purpose. The target is to automatically detect AI generated photo realistic images, id Est: distinguishing whole AI images from real photographs. AI images are not supposed to represent or try to represent any particular real object, place or individual. This can be interesting for classifying images in photography websites and/or in social networks. Note that, in this case, the system is dealing with all kind of images: human faces, animals, still nature, landscapes… For this reason, it is not possible to rely on some of the “face related” characteristics. Relying on artifacts may work for some images but not for all. Artifacts are common in artificial images within some detailed parts (the fingers of a person’s hand, pedals of a bike), but in many images there are no visible errors. What’s more, there are some evident errors, like persons with three hands or even with two heads, which are evident for human view but not so easy to automate in an autonomous recognition system for any type of image.

For this particular application, there are much less references in the literature. In a recent preprint [21], authors propose a method for AI images detection using a complex feature extraction based on two parallel deep learning processes. Results are similar to the ones presented in this paper, but they are using a more complex method and their tests have been conducted on images of smaller resolution (maximum 256x256). In [22], authors construct a huge dataset and they discover that systems trained on one generation model are pretty good on images from that model, but not so much on others. In this work, several models are used to create the dataset and even other different modelsa re tried for finals tests (see discussion section). According to [23], features extracted for recognition are crucial in this problem when trying to work with different generation models. They point out that statistics of overexposed pixels can be a good election, that seems to reinforce the election of PRNU. Other recommendation is using color related features; ELA pattern used in this paper is strongly color related with color as JPEG error is greater on color components. Other references [24–26] focus on CG (Computer Generated) images, which is another type of problem as they are dealing with images that were created with intensive human intervention.

For a similar need, Google has recently announced a new tool, called SynthID [27] which adds an invisible watermark to AI generated images so that they can be identified. Note that this will indentifyAI images only if creation engine watermarks them.

Because of the need of classifying whole images with no assumption about image content, the system has been designed based on methods from other image forensics applications. The main idea is extracting some relevant information from images before applying a convolutional neural network. Convolutional Neural Networks CCN’s are very useful in distinguishing between classes that are visually different for humans like digit classification [28,29], distinguishing objects relevant for taking driving decisions in real traffic and many other similar applications [30]. Nevertheless, in this case, classes are not visually different and that suggests that direct application of CNN’s could not be very useful (besides the experience from the Deep Fake case). For this reason, pixel wise feature extraction has been used. This means using processing stages that convert images into other images with the same size (it converts each pixel to a new pixel) but containing a reduced amount of information that should be relevant to the particular problem of distinguishing AI images.

Methods used for this issue are, up to date, two. First one is Photo Response Non-Uniformity (PRNU). PRNU is in fact a kind of noise used for source camera identification (distinguishing the camera that has taken a given image) [31]. The origin of PRNU is the slightly different sensitivity of individual pixels in a real image sensor. This effect is due to manufacturing imperfections and it is unavoidable. AI images should have no PRNU at all. Nevertheless PRNU computation methods always yield some nonzero result. PRNU has been extensively studied, including its limitations [32,33]. CNN is trained to infer special characteristics of AI images with false PRNU patterns. There exist applications designed for erasing or even forging PRNU patterns (embedding on an image the pattern of a given camera) [34]. So this is a method that can reveal images created by “not very expert” or “not very malicious” users.

The second feature extraction method used is Error Level Analysis (ELA). ELA is a special image (or pattern) that detects irregular errors in JPEG coded images. ELA has been successfully used to detect editions in images (thus to authenticate scanned or photographed images) [35,36]. ELA has also been applied for forged face detection [37]. Basically, ELA detects non uniformity in quantization errors due to JPEG compression. Applied to an AI generated image, ELA normally yields a strange result as if all pixels of the image would have been modified by edition. This could be due to the special nature of AI images coming from training with many JPEG coded photographs. So, ELA pattern is also a good choice for the application that this paper is addressing. It would seem that this method also has a limitation: all images, either coming from a real camera or from an AI engine, must be obtained in JPEG format. It would not be a great drawback as JPEG is the most frequent photography format. Nevertheless, as seen in the remainder of this paper, ELA has been successfully tested on AI images obtained in PNG format.

Another possible feature extraction for this problem is Local Binary Pattern (LBP) [36]. LBP is based on differences between adjacent pixels. An eight-bit word is assigned to each pixel with a binary '1' for greater surrounding pixels and a '0' in other case. This technique has successfully been used for fake faces detection [39]. LBP is also used to detect fake face presentation to biometric systems with video replays. Patel [40] explored this approach detecting moiré patterns with LBP features.

The remainder of the paper is organized as follows: in Section 2, methods and processing are described as well as the image dataset used for training and testing; in Section 3, results are summarized. In the Discussion section, the main results of this work are highlighted.

2. Materials and Methods

2.1. The Dataset

The dataset used in this work for training and testing is composed of a collection of images divided in two groups (or classes): AI generated and real camera photographs. First, AI generated images were created by authors using three different engines: DALL E, Stable Diffusion and OpenArt. These images were visually checked to discard those that were not photo-realistic. Second, real photos were selected randomly from image databases. There are images from Dresden Image Database [41], also from VISION dataset [42] and also from authors' provided images that were already used in previous studies [33,43]. There are real photos from the following cameras: Canon Ixus 70 (two instances), Casio Ex Z150 (two instances), Canon PhotoSmart SX720, Canon EOS 1100D, Kodak M1063 (two instances) and Sony ILCE 5000. Photos from smartphones are also included: Huawei P20, Huawei P9, Samsung Galaxy S3 Mini, Apple Iphone 4s, Apple Iphone 5c, Apple Iphone 6 and LG D290.

Initially, the dataset was made up of 459 AI generated images and the same number of real photographs (a total of 918 images). Afterwards, an extended dataset of 1252 was tested. Both datasets are fully balanced (same number of samples on each class). On each test a percentage of dataset samples will be used for training, leaving the remainder for validation.

2.2. PRNU extraction.

As the name: Photo Response Non-Uniformity, indicates PRNU comes from the different light sensitivity of the different pixels (elementary sensors). This is an unavoidable characteristic due to manufacturing imperfections and it is present on all image sensor chips. PRNU is seen as a multiplicative noise that responds to the following equation [42]:

$$Im_{out} = (I_{ones} + Noise_{cam}).Im_{in} + Noise_{add} \tag{1}$$

Where Im_{in} is the "real" image presented to the camera (the incident light intensity), I_{ones} is a matrix full of ones and $Noise_{cam}$ is the "sensor noise pattern" (PRNU pattern) and Im_{out} is the final image surrendered by the camera. The symbol "." means matrix point by point (pixel-wise) product and $Noise_{add}$ is additive noise from other sources.

PRNU is computed from an image (or from a collection of images coming from the same camera) performing a denoising process on Im_{out}, and then computing a residual:

$$W = Im_{out} - denoise(Im_{out}) \tag{2}$$

Neglecting the additive noise and assuming that Im_{in} = denoise(Im_{out}), given a collection of images from the same camera, the PRNU pattern (sometimes called camera fingerprint) can be estimated as:

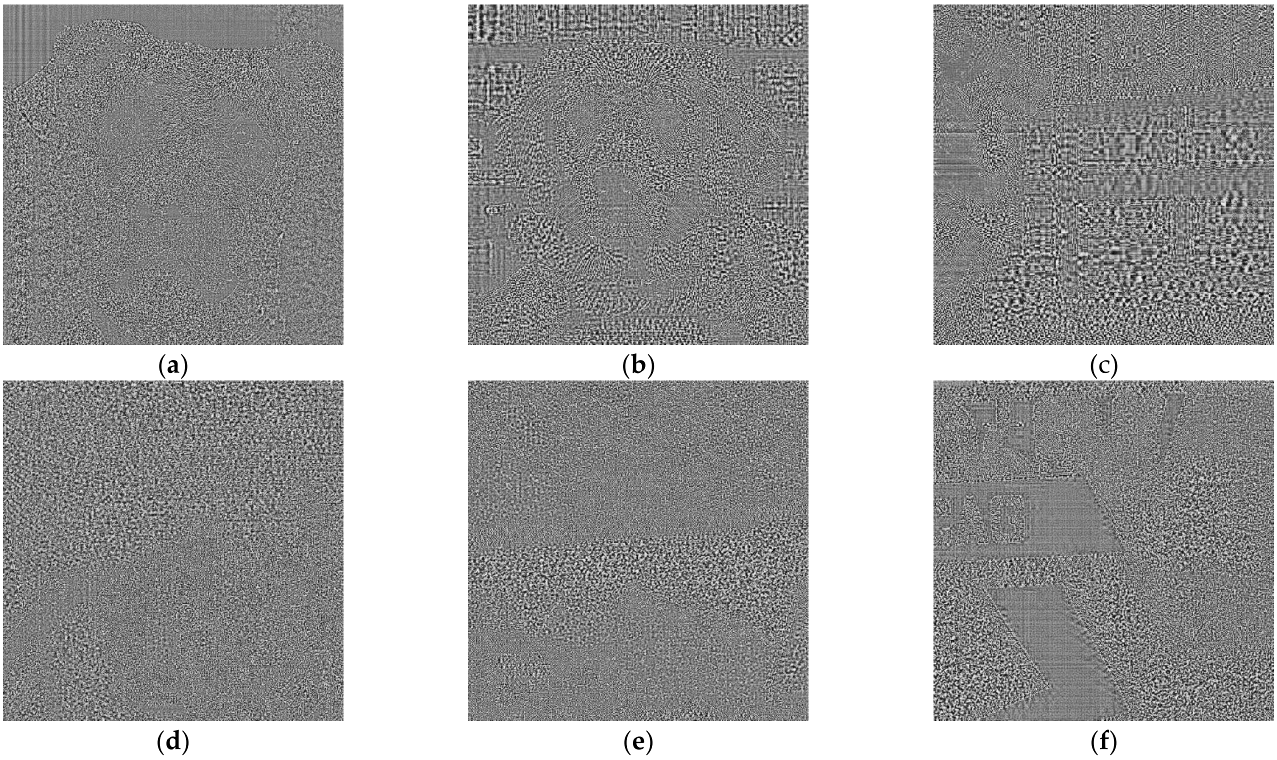
$$F = \frac{\sum_{n=1}^{N} W^n Im_{in}^n}{\sum_{n=1}^{N} (Im_{in}^n)^2} \tag{3}$$

Note that, in this application, we will always compute PRNU fingerprints with a single image (N=1) both for AI generated and for real images. In this case, F=W/Im_{in} (pixel wise quotient) and it is clear that we will get some result, even for AI images.

Note that “denoising” is a noise reduction filter. For this problem, there are several options documented in the literature: median filter [44], Wiener filter [45] and also variations of Wiener filter. In this study, a Matlab [46] implementation from [47] is used, this software uses a Wavelet Transform [48] based Wiener filter.

From each image, a centered square 512x512 region is extracted to work with smaller images and to avoid logos or visible watermarks (that anyway are not present on the dataset images) and PRNU is computed from the sub-image. Note that problem is classifying the whole image, not detecting a “modified” part. Minimum image size for this version is then: 512x512. System can be tailored easily for smaller sizes but that would require retraining.

The results of this process are noise like images very difficult to be interpreted visually (see Figure 2 where PRNU patterns are shown for images of Figure 1, histogram equalization has been applied to enhance a bit these images). Note that equation 2 can be seen as a high pass filter and so, the results contain part of image contours (a normal phenomenon when computing the pattern from a single image). There seems to be no significant visible difference between AI images: (**a**), (**b**) and (**c**) and real ones: (**d**), (**e**) and (**f**).



(**a**) (**b**) (c)

(**d**) (**e**) (**f**)

Figure 2. (**a**), (**b**) and (**c**), PRNU patterns computed for images of figure 1. (**d**), (**e**) and (**f**) are examples of PRNU patterns for real images from different cameras.

2.3. ELA Error Level Analysis.

ELA pattern is computed to detect irregular distributions of quantization noise. This is a tool normally used to detect image editions. An ELA pattern is normally computed coding the whole image with JPEG standard at a known, constant, normally at a high-quality level (a typical value is 95%); then the decoded image from the JPEG bit stream is subtracted from the original image.

$$ELA_{img} = img - JPEG^{-1}[JPEG(img, 95\%)] \tag{4}$$

If we are facing an edited image, an irregular pattern with different intensities will appear. In Figure 3, ELA patterns for the same original images (Figure 1) are shown, Again, histogram equalization has been applied to enhance a bit these images.

Note that, in this case, patterns are color images. For PRNU computation, images are converted to grayscale, prior to all processing. Images are again cropped to the central square sub-image of size 512x512.

Again, a "high pass filtering" effect is evident. Visual differences between AI images AI images: (**a**), (**b**) and (**c**) and real ones: (**d**), (**e**) and (**f**); are again not very remarkable. Perhaps, contours are more evident in the above part but it does not seem conclusive. Nevertheless, neural networks can be able to learn differences that are not perceived by humans.

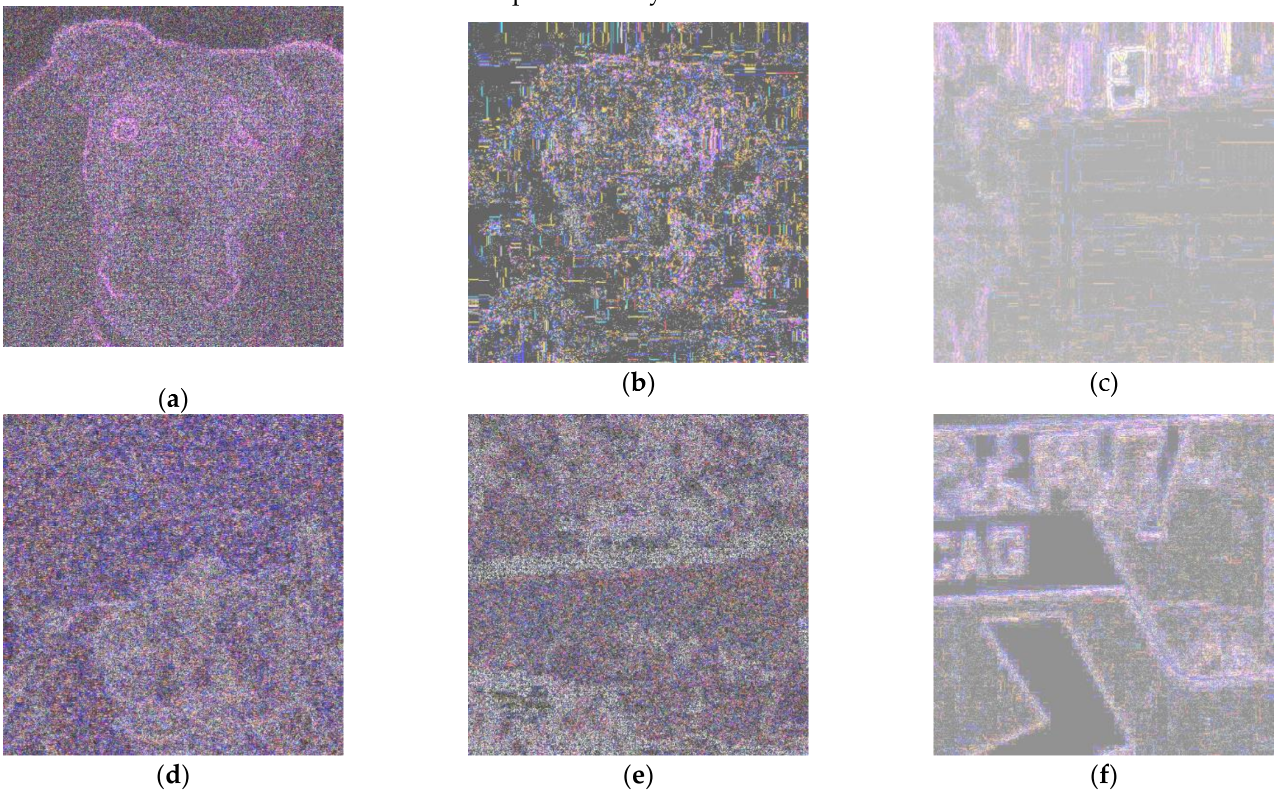

(**a**) (**b**) (c)
(**d**) (**e**) (**f**)

Figure 3. (**a**), (**b**) and (**c**), ELA patterns computed for images of Figure 1. (**d**), (**e**) and (**f**) are examples of ELA patterns for real images used to compute Figure 2.

2.4. CNN's Convolutional Neural Networks.

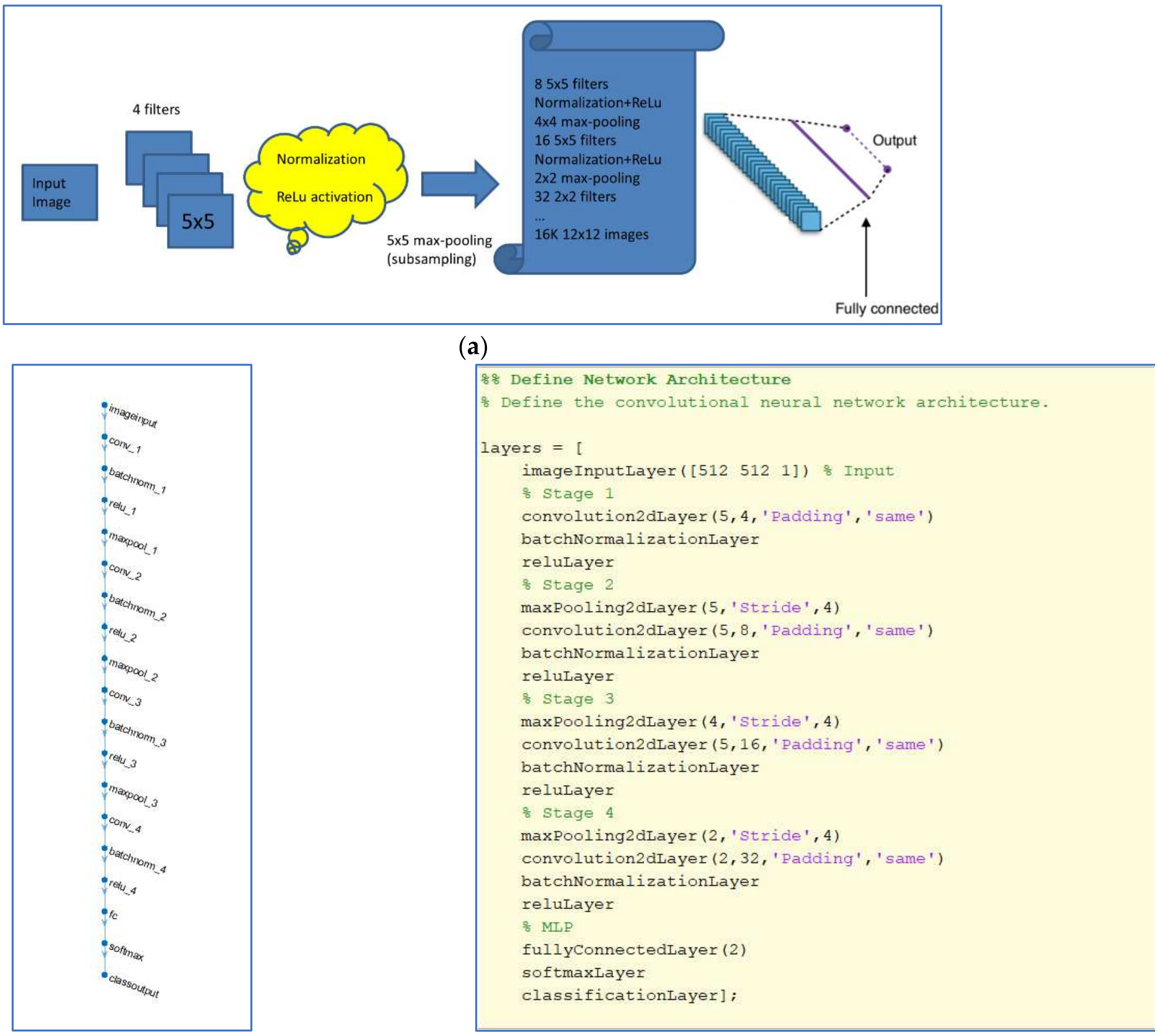
CNN's are basically a cascade of convolutional (or linear filtering) stages accompanied by others of non-linear activation, normalization and decimation. These stages extract high level features from low level data (pixels) and so, CNN's are able to process images directly with no need of feature extraction. The initial image is repeatedly filtered and decimated, creating a set of several small images that are finally processed by a classical perceptron (fully connected) stage to get the final result. This final result is a numerical vector of as many components as classes to be recognized. The Softmax normalization (the most frequently used at final stage of CNN's) makes that vector components lie in the range 0.0-1.0 and besides, they always add up to 1.0. The maximum component defines which one is the recognized class.

Filter coefficients and perceptron weights are all optimized through the training process. Training algorithm is Stochastic Gradient Descent with Momentum (SGDM) [49] that is a gradient-type optimization that minimizes mean square error between the obtained and desired output.

In this paper, a previous image to image transformation is done that acts as a pixel-wise feature extraction. This stage tries to search for relevant characteristics for distinguishing classes, removing unimportant information. As reported in the case of Deep Fake detection, direct CNN application is not good for this type of problem.

The dataset is divided randomly, selecting 85% images of each class for training and leaving the rest for validation. Note that the dataset is balanced (it has the same number of samples for each class). A validation stage is performed at each training epoch, adequately controlling the learning process. Each complete epoch (run of all training samples in random order) is divided into **n** iterations. Each of the iterations is a mini-batch, this is: a set of samples that is processed without updating weights (mini-batch size is **DatasetSize/n**). Testing values for **n**, optimum results were obtained for **n=3**.

CNN structure: the number of stages and filter configuration at each stage is shown in Figure 4. And it is the same for the two kinds of pattern extraction techniques tested.



(**a**)

```
%% Define Network Architecture
% Define the convolutional neural network architecture.

layers = [
    imageInputLayer([512 512 1]) % Input
    % Stage 1
    convolution2dLayer(5,4,'Padding','same')
    batchNormalizationLayer
    reluLayer
    % Stage 2
    maxPooling2dLayer(5,'Stride',4)
    convolution2dLayer(5,8,'Padding','same')
    batchNormalizationLayer
    reluLayer
    % Stage 3
    maxPooling2dLayer(4,'Stride',4)
    convolution2dLayer(5,16,'Padding','same')
    batchNormalizationLayer
    reluLayer
    % Stage 4
    maxPooling2dLayer(2,'Stride',4)
    convolution2dLayer(2,32,'Padding','same')
    batchNormalizationLayer
    reluLayer
    % MLP
    fullyConnectedLayer(2)
    softmaxLayer
    classificationLayer];
```

(**b**) (**c**)

Figure 4. (**a**) CNN structure used. (**b**) Layers diagram. (**c**) Matlab code used to define net structure.

At the end of training, a confusion matrix is computed for the validation set. This means, counting the number of True-Positive (AI images correctly detected), False-Negative (AI images not detected), False-Positive (real photographs detected as AI) and True-Negative (real photographs detected as real). The matrix is arranged in this manner:

$$CM = \begin{bmatrix} TP & FN \\ FP & TN \end{bmatrix} \tag{5}$$

From this matrix, several performance measurements can be computed:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{6}$$

$$P = \frac{TP}{TP+FP} \tag{7}$$

$$R = \frac{TP}{TP+FN} \tag{8}$$

$$F_1 = \frac{2.P.R}{P+R} \tag{9}$$

Accuracy is simply the success rate. The other three parameters are very easy to interpret and are very typical in classification systems: **P** (Precision) would be the probability of true detection for true cases; **R** (recall) would be the probability for effective detection of true cases. **F1-score** is the harmonic mean of **P** & **R**. Obviously, the greater these quantities are, the better performance is achieved.

3. Results

CNN nets were trained and tested for both types of feature extraction. This process produces learning curves displayed in Figures 5 and 6. In both cases, a good result is achieved: accuracy is 0.95 for PRNU and 0.98 for ELA. Both trainings have been done with 100 epochs. Training time is bigger for the ELA case (167 minutes versus 109), this is reasonable because ELA images are color ones with three times more information.

Blue curves in both figures are the accuracy values obtained for each iteration (measured on the training samples), black curves are accuracy values for the validation set at each epoch (an epoch is equal to **n** iterations, with **n=3** in this case). The curves below (brown and black) are the mean square error (over training and validation set), this is other method for controlling learning.

In both cases, the fact that black curves follow the evolution of blue/brown curves demonstrates that neural net is generalizing. In the case of overfitting, the blue curve can go high but the black curve would remain low.

Comparing both trainings, ELA offers more stable results.

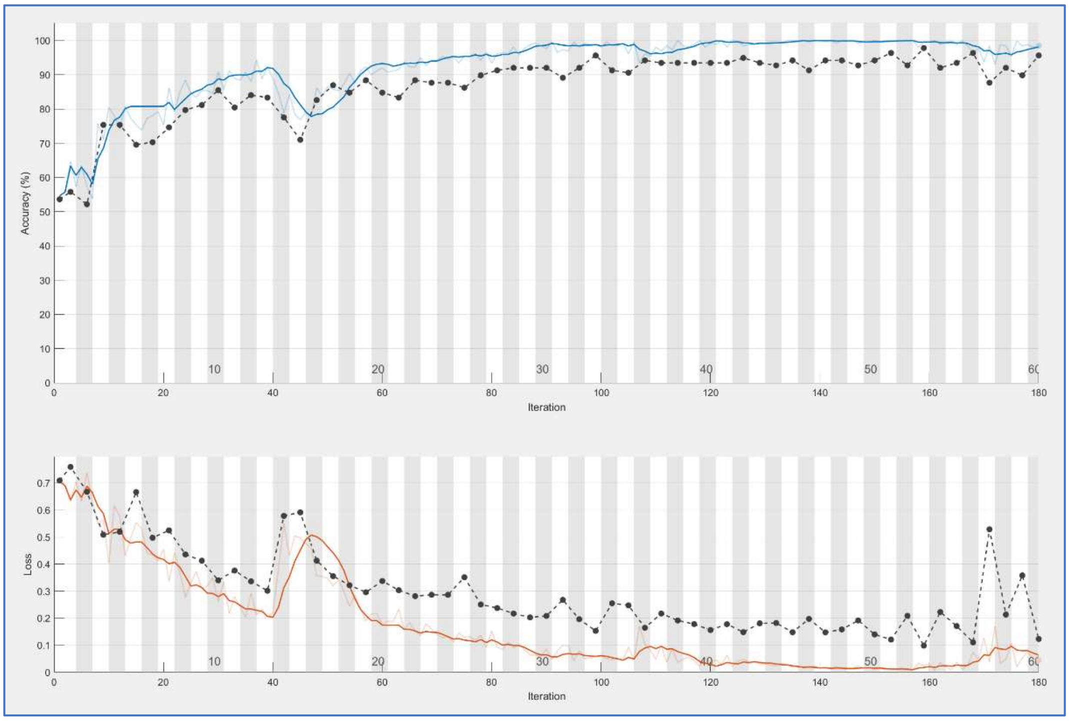


Figure 5. CNN training for PRNU patterns.

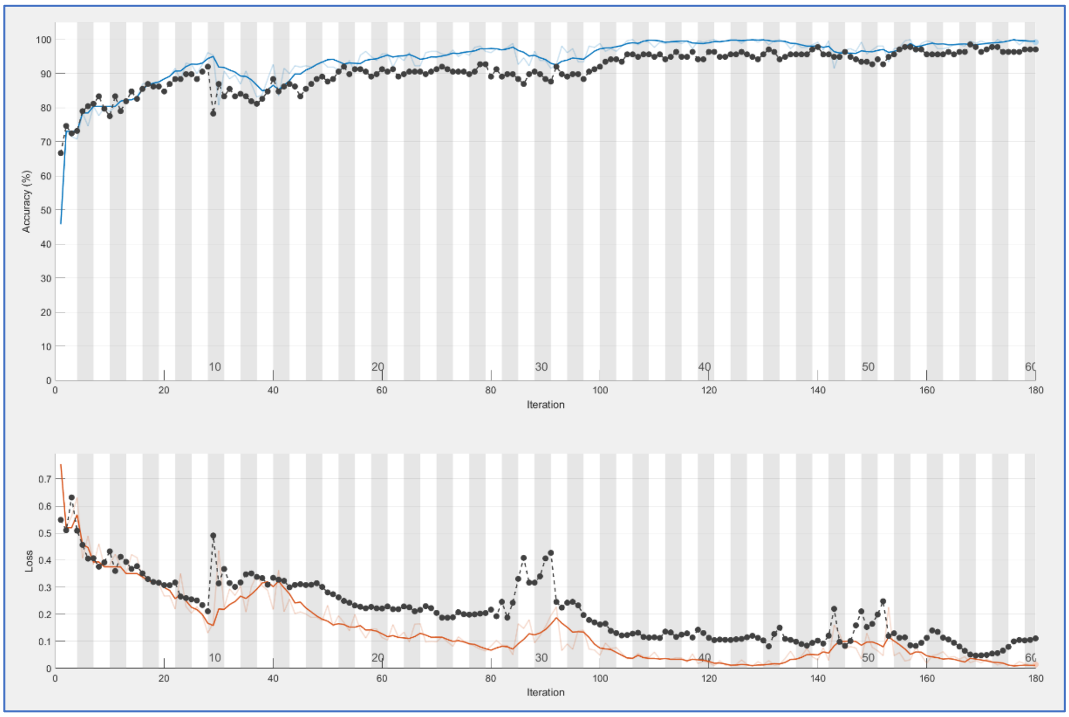


Figure 6. CNN training for ELA patterns.

Final results are:

$$CM(PRNU) = \begin{bmatrix} 67 & 02 \\ 05 & 64 \end{bmatrix}, \quad CM(ELA) = \begin{bmatrix} 66 & 01 \\ 02 & 67 \end{bmatrix} \tag{10}$$

These matrices yield the following numbers in the **P**, **R** and **F₁** terms, see Table 1.

Table 1. Numerical results for both methods.

Method (pattern type)	Accuracy	Precision	Recall	F_1 score
PRNU	0.95	0.93	0.97	0.95
ELA	0.98	0.97	0.99	0.98

4. Discussion

Last presented results seem to demonstrate again that both methods are good but ELA outperforms PRNU with a slight advantage. These results were obtained with a reduced dataset of 459 images per class. Afterwards, a new test was conducted using an extended version with 626 samples per class. This test was only done with the ELA extraction (the best option). Learning curve is presented in Figure 7. In this case the **n** parameter was set to 5 because with more samples, it is necessary to reduce batch size. Number of epochs is 75 because in previous tests, it was seen that learning for ELA features was already getting stable at that point.

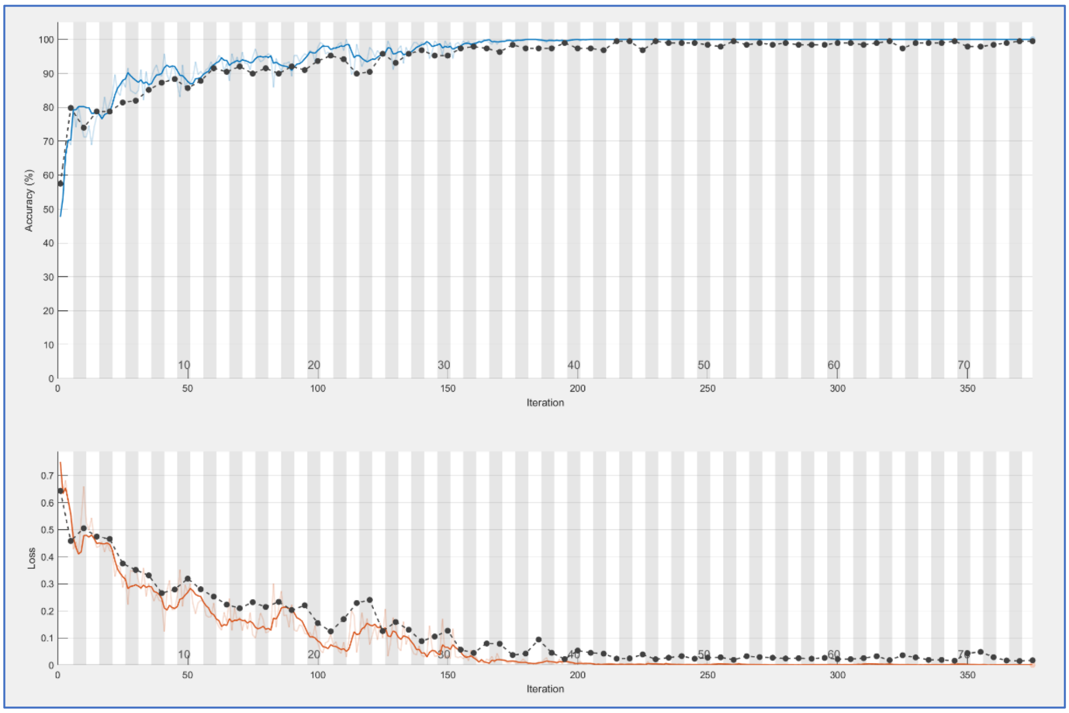


Figure 7. CNN training for ELA patterns.

New assessment data for ELA features improve slightly. Confusion matrix becomes:

$$CM = \begin{bmatrix} 94 & 00 \\ 01 & 93 \end{bmatrix} \qquad (11)$$

Accuracy now is 0.99, Precision is 0.99, recall es 1.0 and F1 score is 0.99. To get more insight into these results, we tested a pre-trained classic net. We chose AlexNet [30]. For this process, image cropping is modified so that we get the required size for input in this CNN: 227x227x3. Three last layers (including the final classification via a fully connected MLP layer) are modified to the new problem of binary classification (two output neurons). Weights for this level are reset to random values. Model is retrained with a very small learning rate at all levels, EXCEPT at the modified ones. Learning parameters are now those recommended for this kind of training: SGDM method [49], only 4 epochs with a mini-batch size of 10 that results in 106 iterations per ephoc. Again dataset is divided into an 85% part for training and 15% for validation. Confusion matrixes for validation set are the following, where test has been performed with the two feature extraction types: PRNU and ELA.

$$CM(PRNU) = \begin{bmatrix} 67 & 02 \\ 05 & 64 \end{bmatrix}, \quad CM(ELA) = \begin{bmatrix} 90 & 04 \\ 12 & 82 \end{bmatrix} \qquad (12)$$

These matrices yield the following numbers in the **P**, **R** and **F$_1$** terms, see Table 2.

Table 2. Numerical results for both methods.

Method (pattern type)	Accuracy	Precision	Recall	F$_1$ score
PRNU	0.88	0.86	0.90	0.88
ELA	0.91	0.88	0.96	0.92

Where it can be seen that method is viable but should be refined a bit. Perhaps the pre-trained levels of AlexNet are good for ordinary images but not so well fitted for PRNU/ELA patterns. Up to the point, the preferred method is ELA + specific CNN.

Another test has been carried out presenting to the original system (to the trained CNN's) a new set of images completely new. Neither used so far in training nor in validation. This new dataset consists of 150 AI generated images and other 150 real photos. Photos have been taken from unused material of VISION database [42] (they are all smarphone photos). AI images have been created using creation engines different from those of the first dataset: Leonardo.AI [50] and TensorArt [51].

Confusion matrixes for this new dataset are now:

$$CM(PRNU) = \begin{bmatrix} 128 & 22 \\ 0 & 150 \end{bmatrix}, \qquad CM(ELA) = \begin{bmatrix} 90 & 04 \\ 0 & 150 \end{bmatrix} \tag{13}$$

These matrices yield the following numbers in the **P**, **R** and **F$_1$** terms, see Table 3.

Table 3. Numerical results for both methods.

Method (pattern type)	Accuracy	Precision	Recall	F$_1$ score
PRNU	0.93	1.00	0.85	0.92
ELA	0.91	1.00	0.82	0.90

Curiously, in this case, PRNU outperforms ELA. What's more, seeing that generally real photos are correctly classified (there are no false positives), creating a combined methods is easy. If both methods are executed on the same image, it is enough that one of them classifies it as AI image to consider it an AI image. Running again test with this combination, **accuracy** goes to 0.97 and **F$_1$ score** to 0.97. Mean execution time of this combined recognition is 0.43 s per image in the matlab application. The implementation takes advantage of the combination "logical OR" nature: if first method applied yields a AI-image result, it is not necessary to execute the second one.

4.1. Conclusions

In this work, an automated system for detecting AI created images and distinguishing them from real camera photographs has been created.

Direct use of CNN's over the images seemed not very recommendable, but extracting pattern-like (or pixel-wise) features like PRNU or ELA patterns yields good results. ELA patterns work slightly better although combination of both methods is easy ans improves results.

This issue is relatively new in the world of image forensics. Although there are many publications about detection of image editions including Ai edition and Deep Fake, pure recognition of 100% created AI images with no assumptions about content is less common. The method presented in this paper has been trained with three different creation models and tested with a validation set obtained from the main dataset and also with a new dataset obtained from other different creation models. Examples documented present similar result but on smaller images [21], good results very dependant on image creation model [22] (note that when executing a recognition possible creation model is not known). In [23], authors study feature extraction methods that can be able to recognize AI images coming from different models. The use of ELA patterns is compatible with their findings.

As supplementary results:

- A new dataset on AI created images has been created. This set could be augmented and published as a separate result.
- A graphical demo application has been created, see appendix A.

4.1. Future Work.

Some lines of future work can be point out now:

- Augmenting the AI images dataset for publication as a public research result.
- Enhancing that dataset incorporating other image creation engines.
- Testing other pixel-wise feature extraction techniques like LBP's (local binary patterns).
- Testing other structures for CNN. Maybe specific or pretrained.
- Testing other classification schemes.
- Exploring further the combination of methods.
- Developing a version that could be used at a server to classify images uploaded to a Web 2.0 service.

- Trying PRNU/ELA features for Deep Fake detection and other anti-forgery applications.

Author Contributions: Initial concept, AI images creation, software development: Fernando Martín-Rodríguez and Mónica Fernández-Barciela. CNN structure design: Rocío García-Mojón. Writing and revising: Fernando Martín-Rodríguez and Mónica Fernández-Barciela.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable (study not involving humans or animals).

Informed Consent Statement: Not applicable (study not involving humans or animals).

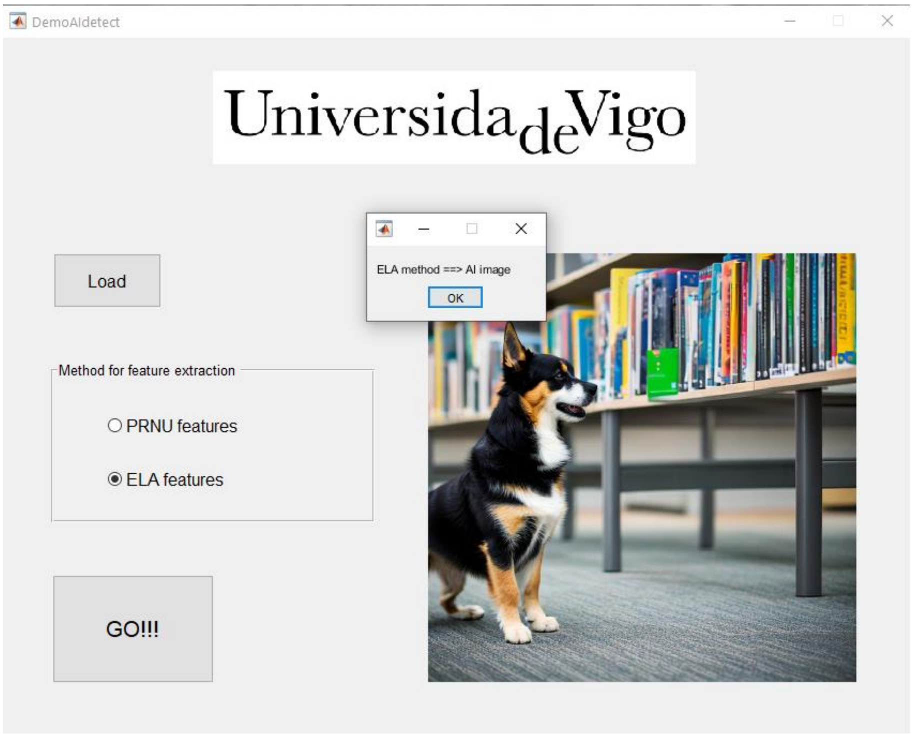
Data Availability Statement: Some data from public image databases: Dresden Image Database and Vision dataset. Other images were created by authors and are available upon request.

Acknowledgments: authors wish to thank all personnel in AtlantTIC research center for their support.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

A demonstration application has been created using matlab GUI design tools. This app allows loading an image, choosing the preferred method (PRNU or ELA) and running the classifier. Result is output using a message box. See Figure A.1.



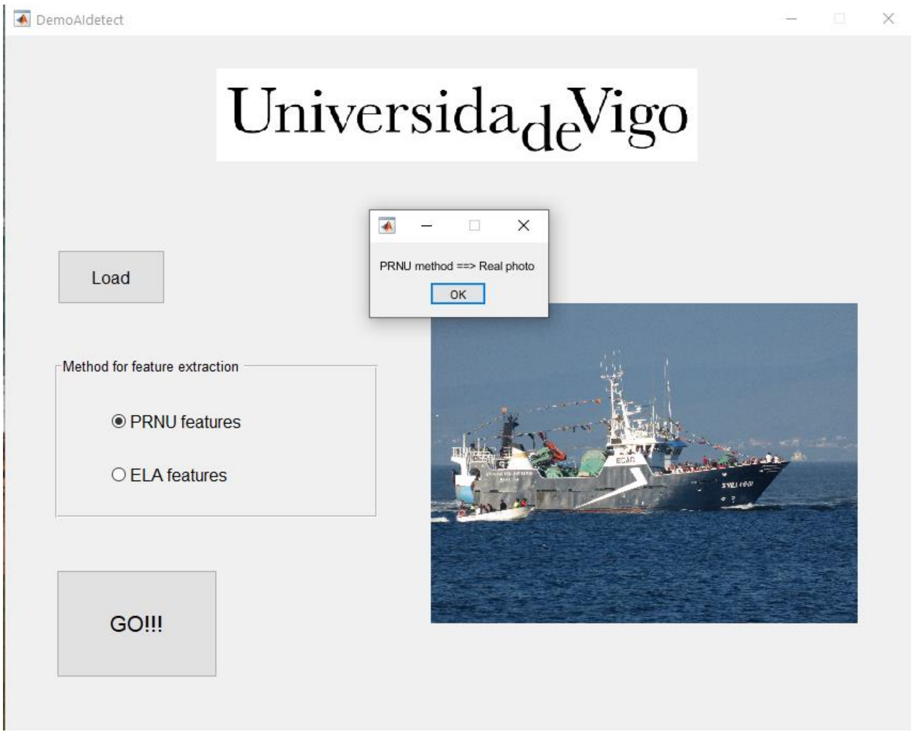
ELA method ==> AI image

Method for feature extraction

○ PRNU features

● ELA features

(a)



PRNU method ==> Real photo

Method for feature extraction

● PRNU features

○ ELA features

(**b**)

Figure A1. Graphical demo application. (**a**) Detection of AI image with ELA method. (**b**) Real photo detection with PRNU method.

References

1. Vaswani, A. et al. Attention is All You Need. In Proceedings of Neural Information Processing Systems (NIPS 2017), Long Beach (Ca, U.S.A.),December 2017. DOI:
2. Vaswani, A. et al. Attention is All You Need. Preprint in arXiv (2017), DOI: 10.48550/arXiv.1706.03762.
3. Tensor flow official tutorial, Neural machine translation with a Transformer and Keras, https://www.tensorflow.org/text/tutorials/transformer?hl=en, last access: 22nd-Oct-2023.
4. Shahriar, S.; Hayawi, K. Let's have a chat! A Conversation with ChatGPT: Technology, Applications, and Limitations. Preprint in arXiv (2023), DOI: 10.48550/arXiv.2302.13817.
5. Shahriar, S.; Hayawi, K. Let's have a chat! A Conversation with ChatGPT: Technology, Applications, and Limitations. Artificial Intelligence and Applications. Vol 1(1), January 2023, DOI: 10.47852/bonviewAIA3202939.
6. Ramesh, A. et al, Hierarchical Text-Conditional Image Generation with CLIP Latents. Preprint in arXiv (2022), DOI: 10.48550/arXiv.2204.06125.
7. Marcus, G., David E., *Aaronson S.*, A very preliminary analysis of DALL-E 2. Preprint in arXiv (2022), DOI: 10.48550/arXiv.2204.13807.
8. Ho, J., Jain A., Abbeel P., Denoising Diffusion Probabilistic Models. Preprint in arXiv (2020), DOI: 10.48550/arXiv. 2006.11239.
9. https://openart.ai/home , last access 22nd-Oct-2023.
10. https://www.imagine.art/, last access 22nd-Oct-2023.
11. https://www.adobe.com/es/sensei/generative-ai/firefly.html, last access 22nd-Oct-2023.
12. Massod M. et al, Deepfakes Generation and Detection: State-of-the-art, open challenges, countermeasures, and way forward. Preprint in arXiv (2021), DOI: 10.48550/arXiv.2103.00484.
13. Nguyen T.T. et al. Deep learning for deepfakes creation and detection: A survey. In Computer Vision and Image Understanding, Vol 223, 103525 (2022), DOI: 10.1016/j.cviu.2022.103525.
14. Rana S. et al. Deepfake Detection: A Systematic Literature Review. In IEEE Access, Vol 10, pp 25494-25513 (2022), DOI: 10.1109/ACCESS.2022.3154404.
15. Rössler et al. FaceForensics: A Large-scale Video Dataset for Forgery Detection in Human Faces. Preprint in arXiv (2018), DOI: 10.48550/arXiv.1803.09179.
16. Matern, F., Riess C., Stamminger M. Exploiting Visual Artifacts to Expose Deepfakes and Face Manipulations. In 2019 IEEE Winter Applications of Computer Vision Workshops (WACVW-2019), Waikoloa (Hi, USA), 2019. DOI: 10.1109/WACVW.2019.00020.

17. Ciftci U.A., Demir I., Yin L. FakeCatcher: Detection of Synthetic Portrait Videos using Biological Signals. In IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol 42 (July), (2020), DOI: 10.1109/TPAMI.2020.3009287.
18. Becattini F. et al. Head Pose Estimation Patterns as Deepfake Detectors. In ACM Transactions on Multimedia Computing, Communications, and Applications, (2023). DOI: 10.1145/3612928.
19. Bappy et al. Hybrid LSTM and Encoder–Decoder Architecture for Detection of Image Forgeries. In IEEE Transactions on Image Processing, Vol 28 (7), pp 3286-3300, (2019). DOI: 10.1109/TIP.2019.2895466.
20. Guarnera, L., Giudice O., Battiato S. Fighting Deepfake by Exposing the Convolutional Traces on Images. In IEEE Access, Vol 8, pp 165085-165098 (2020), DOI: 10.1109/ACCESS.2020.3023037.
21. Xi Z. et al. AI-Generated Image Detection using a Cross-Attention Enhanced Dual-Stream Network. Preprint in arXiv (2023), DOI: 10.48550/arXiv.2306.07005.
22. Zhu, M. et al. GenImage: A Million-Scale Benchmark for Detecting AI-Generated Image. Preprint in arXiv (2023), DOI: 10.48550/arXiv.2306.08571.
23. Chandrasegaran, K. et al. Discovering Transferable Forensic Features for CNN-Generated Images Detection. In: Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T. (eds) Computer Vision – ECCV 2022. ECCV 2022. Lecture Notes in Computer Science, vol 13675. Springer, Cham. DOI: 10.1007/978-3-031-19784-0_39.
24. Cui, Q., McIntosh, S. Identifying materials of photographic images and photorealistic computer generated graphics based on deep CNNs. In Computer Materials and Continua, Vol 55(2), pp 229-241 (2018). DOI: 10.3970/cmc.2018.01693.
25. He, P. et al. Computer Graphics Identification Combining Convolutional and Recurrent Neural Networks. In IEEE Signal Processing Letters, Vol 25(9), (2018). DOI: 10.1109/LSP.2018.2855566.
26. Wang K. Self-supervised learning for the distinction between computer-graphics images and natural images. In Applied Sciences, Vol. 13(3), pp 1887–1887 (2023). DOI: 10.3390/app13031887.
27. https://www.deepmind.com/synthid, last access 22nd-Oct-2023.
28. Lecun Y. et al. Gradient-based learning applied to document recognition. In Proceedings of the IEEE, Vol 86(11), pp 2278-2324 (1998), DOI: 10.1109/5.726791.
29. Ghouzam Y. Introduction to CCN Keras – 0.997. https://www.kaggle.com/code/yassineghouzam/introduction-to-cnn-keras-0-997-top-6, last access 22nd-Oct-2023.
30. Krizhevsky A., Sutskever I., Hinton G.E. ImageNet Classification with Deep Convolutional Neural Networks. In Advances in Neural Information Processing Systems 25 (NIPS 2012). https://proceedings.neurips.cc/paper_files/paper/2012, last accessed 21st-Sept-2023.
31. Lukas J., Fridrich J., Goljan M. Digital camera identification from sensor pattern noise. In IEEE Transactions on Information Forensics and Security (TIFS), 2006, Vol 1(2), pp 205-214. DOI: 10.1109/TIFS.2006.873602.
32. Pérez-Gonzalez, F., Fernández-Menduiña, S. PRNU-leaks: Facts and remedies. In Proceedings of the 2020 28th European Signal Processing Conference (EUSIPCO), Amsterdam, The Netherlands, 18–21 January 2021.
33. Martín-Rodríguez F., Isasi-Vicente F., Fernández-Barciela M. A Stress Test for Robustness of Photo Response Nonuniformity (Camera Sensor Fingerprint) Identification on Smartphones. In Sensors, Vol 23(7), 3462, 2023. DOI: 10.3390/s23073462.
34. https://sourceforge.net/projects/prnudecompare/, last accessed 22nd-Oct-2023.
35. Abd Warif N.B. et al. An evaluation of Error Level Analysis in image forensics. In 2015 5th IEEE International Conference on System Engineering and Technology (ICSET-2015). Shah Alam (Malaysia), 10-11 August 2015. DOI: 10.1109/ICSEngT.2015.7412439.
36. Gupta A., Joshi R., Laban R. Detection of Tool based Edited Images from Error Level Analysis and Convolutional Neural Network. Preprint in arXiv (2022), DOI: 10.48550/arXiv. 2204.09075.
37. Qurat-ul-ain et al. Forged Face Detection using ELA and Deep Learning Techniques. In 2021 International Bhurban Conference on Applied Sciences and Technologies (IBCAST-2021), Islamabad (Pakistan), 12-16 January 2021, DOI: 10.1109/IBCAST51254.2021.9393234.
38. Pietikäinen M. et al. Local Binary Patterns for Still Images. In Computer Vision Using Local Binary Patterns. Computational Imaging and Vision, vol 40. Springer, London (2011). DOI: 10.1007/978-0-85729-748-8_2.
39. Boulkenafet Z., Komulainen J., Hadid A. Face anti-spoofing based on color texture analysis. Proceedings of the 2015 IEEE International Conference on Image Processing (ICIP-2015). Quebec (Canada), 27-30 September, 2015. DOI: 10.1109/ICIP.2015.7351280.
40. Patel K. et al. Live face video vs. spoof face video: Use of moiré patterns to detect replay video attacks. Proceedings of 2015 International Conference on Biometrics (ICB). Phuket, Thailand, 19-22 May, 2015. DOI: 10.1109/ICB.2015.7139082.
41. Gloe T., Böhme R. The 'Dresden Image Database' for Benchmarking Digital Image Forensics. In Proceedings of the 2010 ACM Symposium on Applied Computing, New York (NY, USA), 2010, DOI: 10.1145/1774088.1774427.

42. Shullani D. et al. VISION: a video and image dataset for source identification. In Eurasip Journal on Information Security, Vol 15 (2017), DOI: 10.1186/s13635-017-0067-2.
43. Martín-Rodríguez F. Testing Robustness of Camera Fingerprint (PRNU) Detectors. Preprint in arXiv (2021), DOI: 10.48550/arXiv.2102.09444.
44. G. J. Bloy. Blind camera fingerprinting and image clustering. In IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), Vol. 30(3), pp. 532-534 (2008). DOI: 10.1109/TPAMI.2007.1183.
45. H. Ogawa and E. Oja. Projection filter, Wiener filter, and Karhunen-Loève subspaces in digital image restoration. In Journal of Mathematical Analysis and Applications, Vol. 114(1), pp. 37-51 (1986). DOI: 10.1016/0022-247X(86)90063-6.
46. https://www.mathworks.com/matlab, last accessed 22nd-Oct-2023.
47. Camera Fingerprint Homepage (from professor Goljan), http://dde.binghamton.edu/download/camera_fingerprint/, last accessed 22nd-Oct-2023.
48. C.K. Chui. *An Introduction to Wavelets*, Academic Press, San Diego (1992).
49. Liu Y., Gao Y., Wotao Y. An Improved Analysis of Stochastic Gradient Descent with Momentum. NIPS'20: Proceedings of the 34th International Conference on Neural Information Processing Systems December (NIPS 2020), pp 18261–18271. DOI: 10.5555/3495724.3497257.
50. https://app.leonardo.ai/ai-generations, last accessed 22nd-Oct-2023.
51. https://tensor.art/, , last accessed 22nd-Oct-2023.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.