Modular Convolutional Neural Network for Discriminating between Computer-Generated Images and Photographic Images

Huy Nguyen, T. Ngoc-Dung T Tieu, Hoang-Quoc Nguyen-Son, Vincent Nozick, Junichi Yamagishi, Isao Echizen

Modular Convolutional Neural Network for Discriminating between Computer-Generated Images and Photographic Images

Huy H. Nguyen[1], Ngoc-Dung T. Tieu[1], Hoang-Quoc Nguyen-Son[2], Vincent Nozick[3,5], Junichi Yamagishi[1,2,4], and Isao Echizen[1,2]

[1] *SOKENDAI (The Graduate University for Advanced Studies), Kanagawa, Japan*
[2] *National Institute of Informatics, Tokyo, Japan*
[3] *Japanese-French Laboratory for Informatics, JFLI, UMI 3527*
[4] *The University of Edinburgh, Edinburgh, United Kingdom*
[5] *Université Paris-Est Marne-la-Vallée, LIGM, UMR 8049*

Abstract

Discriminating between computer-generated images (CGIs) and photographic images (PIs) is not a new problem in digital image forensics. However, with advances in rendering techniques supported by strong hardware and in generative adversarial networks, CGIs are becoming indistinguishable from PIs in both human and computer perception. This means that malicious actors can use CGIs for spoofing facial authentication systems, impersonating other people, and creating fake news to be spread on social networks. The methods developed for discriminating between CGIs and PIs quickly become outdated and must be regularly enhanced to be able to reduce these attack surfaces. Leveraging recent advances in deep convolutional networks, we have built a modular CGI–PI discriminator with a customized VGG-19 network as the feature extractor, statistical convolutional neural networks as the feature transformers, and a discriminator. We also devised a probabilistic patch aggregation strategy to deal with high-resolution images. This proposed method outperformed a state-of-the-art method and achieved accuracy up to 100%.

CCS Concepts: Security and privacy → Biometrics; Security and privacy → Social network security and privacy; Computing methodologies → Machine learning;

1 Introduction

Despite the many benefits of computer-generated images (CGIs), for example in gaming, virtual reality, and 3D animation, they can also be used for malicious purposes. Videos generated for creating fake news to gain political advantages, create chaos, or damage reputations can easily spread uncontrollably in social networks. From the Digital Emily Project in 2010 [1] to the Face2Face Project in 2016 [34] and the Synthesizing Obama Project in 2017 [32], the requirements for performing a spoofing attack have been greatly simplified, from obtaining 3D scanning information captured by sophisticated devices (which is unrealistic for most attackers) to only needing RGB videos (which can be easily obtained online), and now to generating spoofing video in real time. Approaches like Face2Face can be used to break challenge-response tests in facial authentication systems or to impersonate people in teleconferences. Moreover, recent advances in generative adversarial networks (GANs) [13] have overcome the size-limit problem, enabling realistic facial images to be generated in unprecedented high-definition quality (1024 × 1024) [17]. These developments have raised alarms in forensics research as well as in security and privacy areas. Discriminating between such high-quality computer-generated multimedia and their natural counterparts, especially in the case of images, is a continuous competition between the attacker side and the defender side.

Statistical properties obtained from transformed images (e.g., from wavelet transform or differential operators) have been widely used to distinguish CGIs from photographic images (PIs) [22, 3, 20, 37, 38, 4] and were recently demonstrated to be the best features for discrimination by Rahmouni et al. [27]. They also demonstrated that applying automatic feature extraction using a convolutional neural network (CNN) can substantially improve classification compared with using handcrafted features.

In addition, the pre-trained VGG networks proposed by the Visual Geometry Group at the University of Oxford [30] (VGG-16 and VGG-19) have been widely used in areas outside their originally intended scope as image classification networks, such as for perceptual loss in the style transfer problem and for the super-resolution problem [16, 19]. Furthermore, these VGG networks were trained using a large-scale dataset [29], which maximizes the generalization ability of a CNN.

In the research reported here, we leveraged the generalization ability of the VGG-19 network, combined with statistical properties applicable to CNNs, to build a modular CGI–PI classifier. To deal with high-resolution images while minimizing computational cost, we use a probabilistic patch aggregation strategy that reduces V-RAM usage and shortens classification time.

2 Related Work

Previously reported approaches to distinguishing CGIs from PIs can be classified into four groups.

1. Using wavelet/wavelet-like transformations or differential images
2. Using the intrinsic properties of image acquisition devices
3. Using texture information
4. Using statistical analysis (independently or jointly with other methods)

Early research on digital image forensics by Farid and Lyu [11, 22] suggested that statistics on the first- and higher-order wavelets can be used to classify CGIs and PIs. Wang and Moulin [36] improved on this approach by using features extracted from characteristic functions of wavelet histograms. Chen et al. [3] suggested that a genetic algorithm could help in selecting an optimal feature set from the statistical moments of the characteristic functions of an image and its wavelet subbands. Li et al. [20] used second-order difference statistics while Wu et al. [37] extracted features from histograms of difference images.

To detect CGI–PI splicing, Conotter and Cordin [5] exploited both wavelet-based features and noise residual statistics. For the same problem, Chen and Ke [4] proposed using a hybrid classifier taking as input the pattern noise statistics and histogram features of first- and second-order difference images.

Work on distinguishing between CGIs and PIs includes work focused on identifying the footprints of image acquisition devices. Khanna et al. [18] took advantage of the residual pattern noise caused by both CCD (charged coupled device) and CMOS (complementary metal oxide semiconductor) sensors inside digital cameras or scanners. Dirik et al. [8] focused on traces of demosaicing and chromatic aberration in color filter arrays (CFAs), as did Gallagher and Chen [12]. Peng et al. [25] also targeted CFAs and identified the effect of their interpolation on the local correlation of photo response non-uniformity noise.

Ng et al. [24] proposed a fusion classification system using the geometry (object model, light, post-processing), the wavelet, and the cartoon features. Fan et al. [10] clarified the limitations of using wavelets and made use of contour information. Zhang et al. [38] extracted the statistical properties of local edge patches in digital images. Also using statistical analysis, Li et al. [21] explored the use of uniform gray-scale invariant local binary patterns. Tan et al. [33] improved previous work by using the local ternary count based on local ternary patterns. In other work, Peng et al. [26] proposed using multi-fractal and regression analysis.

Recently, Rahmouni et al. [27] demonstrated that using statistics is the best approach to solving this forensic problem and that applying a CNN substantially improves the performance of traditional statistical-based methods. To the best of our knowledge, the method of Rahmouni et al. is state-of-the-art, with the highest accuracy for distinguishing between CGIs and PIs.

3 Network Architecture

3.1 Overview

Figure 1: Overview of modular CNN discriminator.

Our modular CNN for discriminating between CGIs and PIs includes three modules, as illustrated in Figure 1. Unlike recent work [27], we do not train the whole network end-to-end. The biggest problem with CNNs is the need to use a large-scale and diverse-content training dataset in order to achieve the best generalization. The dataset used by Rahmouni et al. [27] is relatively large but is less diverse in content than the ILSVRC15 dataset [29]. Unfortunately, the ILSVRC15 dataset was designed for visual recognition, not digital image forensics research. However, CNNs have the ability to transfer learning, so the knowledge gained from solving one problem can be used to solve a different but related problem. Therefore, we used one of the winners of the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) – the pre-trained VGG-19 network, as the feature extractor module. It is important to note that we did not fine tune the feature extractor in the training process.

Although recent work [37, 27] has shown that statistical properties obtained from transformed images are the best features for CGI–PI discrimination, the features extracted from the pre-trained VGG-19 network were designed for visual recognition. Therefore, we constructed feature transformer modules to transform the output extracted by the feature extractor into statistical features. The number of convolutional layers in the transformers must be limited to prevent them from extracting semantic information, but there must be a sufficient number of such layers to be able to extract good statistical information.

The final module is a classifier. For this module, we selected the machine learning algorithm among state-of-the-art ones that has the best classification results.

3.2 Feature Extractor

Johnson et al. [16] suggested that the results obtained from some activation layers of the pre-trained VGG-16 network can be used to calculate the feature reconstruction loss and the style reconstruction loss, which are used for both the style transfer problem and the image super-resolution problem. Ledig et al. [19] argued that, in the case of feature reconstruction loss, using output from a deeper activation layer of the pre-trained VGG-19 network results in better perceptual quality than that with Johnson et al.'s approach. Therefore, there is no standard guideline for the utilization of the VGG network family. In the case of digital forensics, we hypothesized that features in lower layers have more discriminating power than ones from higher levels, which mostly contain semantic information. Moreover, instead of using the output of the rectified linear units (ReLUs) [23], for which negative values are omitted, we extracted output immediately after the convolutional layers.

To verify this hypothesis, we performed an experiment using the patches dataset proposed by Rahmouni et al. [27] and the pre-trained VGG-19 network. We extracted the outputs after five convolutional layers located immediately before the max-pooling layers as shown in Figure 2. For the five settings given in Table 1, the combination of layers 1, 2, and 3 gave the highest classification accuracy. These results indicate that using only one layer does not produce the highest accuracy. However, if semantic layers were included, the classification performance would be affected by this irrelevant information. Therefore, we chose outputs from layers 1, 2, and 3 in Figure 2 (conv1_2, conv2_2, and conv3_4, respectively) as features to be extracted by the feature extractor.

Table 1: Accuracies for Training Using Patches Dataset for Five Settings.

Setting	Accuracy (%)
1	95.40
1 + 2	97.60
1 + 2 + 3	**97.70**
1 + 2 + 3 + 4	96.50
1 + 2 + 3 + 4 + 5	96.10

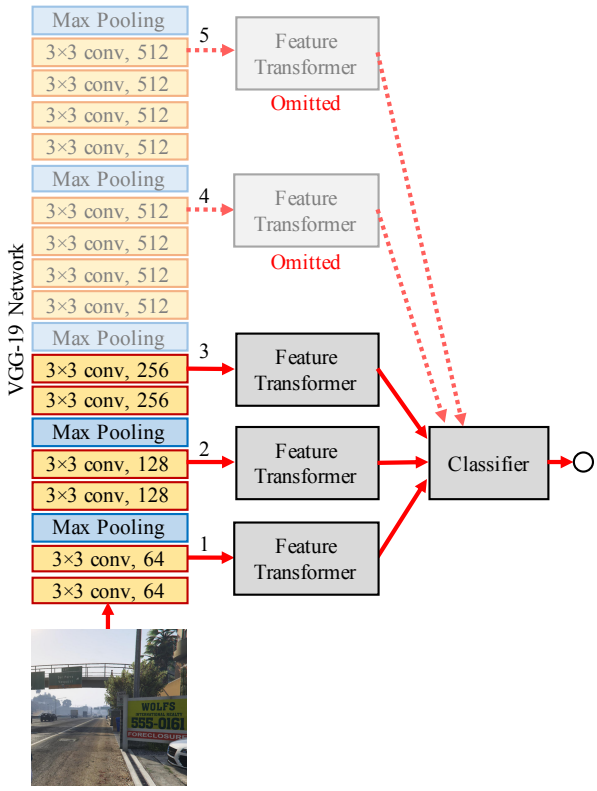

Figure 2: Detailed design of feature extractor and its connections with feature transformers and classifier.

3.3 Feature Transformers

The role of the feature transformers is to transform features encoded by the pre-trained VGG-19 network into statistical properties that can be used to distinguish CGIs from PIs. Because there are three feature transformer modules, it is necessary to minimize their depths. Moreover, a deep feature transformer may produce unnecessary semantic information, which could negatively affect the network. However, a shallow network has a limited ability to transform the features. Therefore, we used two convolutional layers with 3×3 kernels and a stride of 1. We integrated batch normalization layers [15] into the transformers to regularize their training processes. Following the batch normalization layers are the ReLU activation layers. We attached a statistical pooling layer at the end of the modules to extract the statistical properties. The three feature transformers share the same architecture, as illustrated in Figure 3.

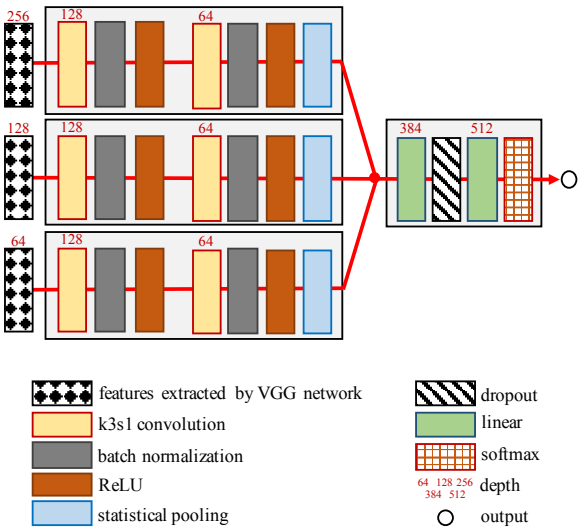

Figure 3: Detailed settings of feature transformers and classifier.

We built the statistical pooling layer following Rahmouni et al.'s approach [27]. However, we assumed that finding the maximum and minimum of each filter was not necessary and that these actions would consume computational power, especially when performing back propagation in the training phase. Therefore, we calculate only the mean and variance of each filter, which are important in statistics and also are differentiable.

- Mean:

$$\mu_k = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} I_{kij}$$

.

- Variance:

$$\sigma_k^2 = \frac{1}{H \times W - 1} \sum_{i=1}^{H} \sum_{j=1}^{W} (I_{kij} - \mu_k)^2$$

.

The k represents the layer index, H and W are respectively the height and width of the filter, and I is a two-dimensional filter array.

3.4 Classifier

Feed-forward multilayer networks, or multilayer perceptrons (MLPs), [28] are widely used to build classifiers in CNNs because of their differentiable property. However, there are other strong classification algorithms that have been widely used such as Fisher's linear discriminant analysis (LDA) algorithm [9] and the support vector machine (SVM) algorithm [6]. Therefore, we first use an MLP to build the classifier to train the feature transformers (as well as to train the classifier itself). After the training, the feature transformers are kept fixed, and the classifier is trained using the LDA and SVM classification algorithms. The learning curves of these algorithm are plotted in Figure 4. The proposed network converged very quickly in the few first epochs. The MLP algorithm had high accuracy but was less stable than the LDA and SVM algorithms. Since the LDA algorithm usually has higher accuracy than the SVM one, we evaluated only MLP and LDA classifiers, as described in section 5.

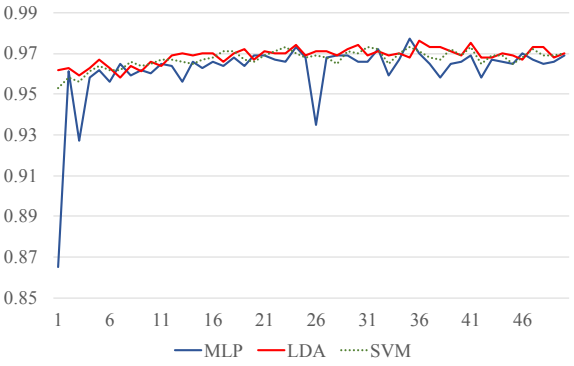

Figure 4: Learning curves of MLP, LDA, and SVM classifiers on Patch-100-Full validation set described in section 5.1

In more detail, two properties are extracted by each statistical pooling filter: the mean μ_i and the variance σ_i. Each pooling layer has 64 filters. Since there are three feature extractor modules, the classifier receives a 384-dimension vector. For the MLP algorithm, we used two hidden layers and one dropout layer [31] in between (with a dropout rate of one-third to avoid over-fitting). A classifier using the MLP algorithm is illustrated in Figure 3. For the LDA and SVM classifiers, we used the LinearDiscriminantAnalysis and SVC module of the scikit-learn library. [1]

To choose the best weights for the feature transformers and the classifier, we begin from epoch 20 and use the one with the highest score in the validation set. Although the proposed network converged very quickly, it is better to use a longer training time to optimize its weights before harvesting.

4 Patch Aggregation

Using a CNN with large-scale input requires a large amount of GPU memory. One possible solution is to split the input into patches, perform classification, and aggregate the results [27]. Although this approach can also detect local CGI inlay in large PI images (or vice-versa), it has high computational cost, especially when dealing with very large images. For instance, an image 4900×3200 pixels in size would require 1568 patches if the patch size was 100×100 pixels. This would result in 1568 classification calculations.

To reduce the number of calculations, we devised an approach using a probability sampling method that randomly selects a portion of the patches, performs classification using the selected patches, calculates the average of the predicted probabilities, and uses it as the final decision. Two patch selection strategies are illustrated in Figure 5. For some fixed number of patches (e.g., 10, 25, or 50), we could integrate them into one batch and feed that batch into the network instead of feeding each patch separately into the network, thereby shortening the computation time.

Let

- y_{pred} be the predicted label of input image I, which is either 0 (PI) or 1 (CGI).
- W be the set of patches w_i extracted from the full-size image I, $|W| = N$ (patches).
- $p(w_i) = \mathcal{D}(w_i)$ be the probability of patch w_i being classified by the proposed network $\mathcal{D}$ as CGI.

[1] http://scikit-learn.org/


Figure 5: Patch selection strategies: Selecting all patches (left) vs. random sampling (right).

The probability of I being classified as CGI is calculated using

$$p(I) = \frac{1}{N}\sum_{i=1}^{N} p(w_i). \quad (1)$$

Hence, the predicted label of I is

$$y_{pred} = \begin{cases} 1, & \text{if } p(I) > 0.5 \\ 0, & \text{otherwise.} \end{cases} \quad (2)$$

5 Evaluation

5.1 Datasets

For the image datasets, we began with the one recently constructed by Rahmouni et al. [27]. Its CGI part contains 1800 high-resolution (around 1920×1080 pixels) screenshots in JPEG format from five photo-realistic video games. The PI part is taken from the RAISE dataset [7], includes 1800 very high-resolution JPEG images (around 4900×3200) directly converted from RAW format. Both parts cover many kinds of indoor and outdoor environments. Sample images from this dataset are shown in Figure 6.

We made one major change to this dataset. We contend that the reduced-size images created by cropping high-resolution images to 650×650 are not appropriate for our purposes because their quality is still good. In reality, many images and videos have low quality, and a malicious person could additionally apply transformation to the CGIs, for example, scaling them to produce lower quality, to disguise the attack. Therefore, instead of cropping, we resized each high-resolution image to $360p$ resolution using a bilinear interpolation algorithm. This increased the diversity in quality of images used for evaluation.

In addition to using a patch size of 100×100, we also used a patch size of 256×256 for the high-resolution images to reduce the number of patches. This larger patch size could be used with large-memory GPUs. Moreover, a larger patch size should contain more valuable information, and with the size is the power of 2, we could reduce the effect of JPEG artifacts. In addition, we also extracted 100×100 patches from the reduced-size images. The datasets derived from the original one are summarized in Table 2.

We trained each discriminator on the training sets of the patch datasets. The valid. sets were used to validate the training process. After training, the discriminators were tested on the testing sets of both patch datasets and their corresponding Full-Size or Reduced-Size ones. Moreover, as described in section 5.3, we also tested the discriminators which were trained using the Patch-100-Full dataset on the Reduced-Size dataset to check whether this training strategy is capable of generalization.

5.2 Testing on High-Resolution Images

For testing on high-resolution images, we trained our proposed method and Rahmouni et al.'s one [27] on the Patch-100-Full and the Patch-256-

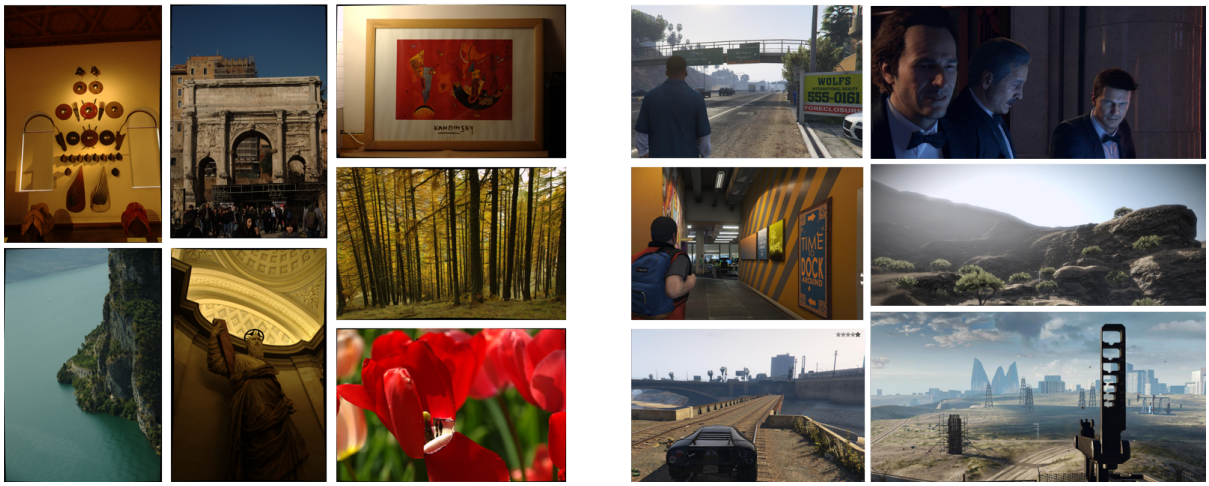

Figure 6: Sample images from dataset constructed by Rahmouni et al. [27]. Images on the left are PIs and those on the right are CGIs.

Table 2: Datasets Used for Evaluation.

Name	No. for training	No. for valid.	No. for testing	Image size
Full-Size	2,520	360	720	High-resolution
Patch-100-Full	40,000	1,000	2,000	100×100
Patch-256-Full	40,000	1,000	2,000	256×256
Reduced-Size	2,520	360	720	360p
Patch-100-Reduced	40,000	1,000	2,000	100×100

Full datasets. We then evaluated them on both the corresponding patch dataset and the Full-Size one. The proposed method was also tested for several patch aggregation strategies, as presented in Table 3. For the 100×100 patch size, it was sufficient to sample only 50 patches to obtain performance equivalent to that of evaluating all patches on the Full-Size dataset. When the sampling process avoided some confused areas in the images, sampling only 10 256×256 patches outperformed sampling 25 patches or evaluating all patches, achieving an accuracy of 100%. Otherwise, the accuracy was slightly lower (e.g., 99.72%).

Our proposed method substantially outperformed Rahmouni et al.'s method [27] on both the Patch-100-Full and Patch-256-Full datasets. It also had the highest results on the Full-Size dataset, reaching 100%. A comparison of accuracy between Rahmouni et al.'s method [27] and the proposed method is shown in Table 4. Comparing the original 100×100 patch size with the 256×256 one shows that increasing the patch size improves the accuracy of Rahmouni et al.'s method. Moreover, use of the MLP classifier rather than the LDA one in the proposed method resulted in higher accuracy for both the Reduced- and Full-Size datasets. The ROC curves for the Patch-100-Full and Full-Size dataset discriminators are plotted in Figures 7 and 8.

5.3 Dealing with Low-Resolution Images

In reality, many videos on social networks such as YouTube, Facebook, and Vimeo have 360p quality. Attackers can take advantage of this to produce low-resolution videos (and images) that are more difficult to detect. The results shown in Table 5 highlight this problem for discriminators trained on the Patch-100-Full dataset. Their performance

Table 3: Accuracy for Several Patch Aggregation Strategies on Full-Size Dataset. The Random Sampling Strategy Was Evaluated Three Times.

Classifier		**MLP**				**LDA**			
Patch size	**No. of patches**	**1**	**2**	**3**	**Avg.**	**1**	**2**	**3**	**Avg.**
100 × 100	10	99.31	99.72	99.86	99.63	99.86	99.31	99.72	99.63
	50	99.86	99.86	99.86	**99.86**	99.86	99.86	99.86	**99.86**
	100	99.86	99.86	99.86	**99.86**	99.86	99.86	99.86	**99.86**
	All				**99.86**				**99.86**
256 × 256	5	99.72	99.44	99.72	99.63	99.44	99.03	99.58	99.35
	10	100.00	99.72	100.00	**99.91**	99.86	99.58	99.72	99.72
	25	99.86	99.86	99.86	**99.86**	99.72	99.72	99.72	99.72
	All				**99.86**				99.72

Table 4: Comparison of Accuracy between Rahmouni et al.'s Method [27] and Proposed Method.

Method	**Patch-100-Full**	**Patch-256-Full**	**Full-Size**
Rahmouni et al. - 100 [27]	86.10	×	96.94
Rahmouni et al. - 256 [27]	×	93.95	98.75
Proposed method - MLP - 100	**96.55**	×	**99.86**
Proposed method - LDA - 100	96.40	×	99.86
Proposed method - MLP - 256	×	**98.70**	**99.72 - 100.00**
Proposed method - LDA - 256	×	98.70	99.58 - 99.86

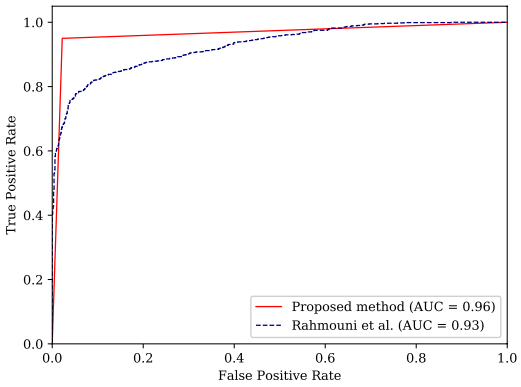

Figure 7: ROC curves of discriminators tested on Patch-100-Full dataset. Proposed method used MLP classifier.

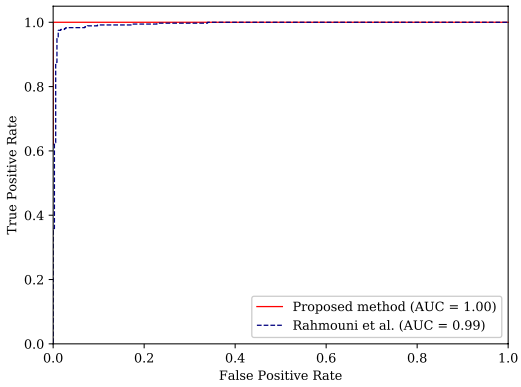

Figure 8: ROC curves of discriminators tested on Full-Size dataset. Proposed method used MLP classifier.

substantially decreased to the random-selection level. To solve this problem, we mixed the Patch-100-Full and the Patch-100-Reduced datasets to form the Patch-100-Mixed dataset. We then re-trained the discriminators on this new dataset and evaluated them on the Patch-100-Reduced & Reduced-Size datasets and Patch-100-Full & Full-Size datasets.

The results in Table 5 show that both discriminators had better performance on the Patch-100-

Table 5: Accuracy of Classifiers Trained on Patch-100-Full Dataset (Old) or on Patch-100-Mixed Dataset (New). For Simplicity, Proposed Method Used All-Patch Strategy.

Method	Patch-100-Reduced	Reduced-Size	Patch-100-Full	Full-Size
Rahmouni et al. (old) [27]	51.50	50.97	86.10	96.94
Proposed method - MLP (old)	52.55	51.81	**96.55**	**99.86**
Proposed method - LDA (old)	52.35	51.53	96.40	99.86
Rahmouni et al. (new) [27]	60.45	79.72	81.20	95.00
Proposed method - MLP (new)	88.60	96.67	93.40	97.64
Proposed method - LDA (new)	**89.95**	**97.92**	94.80	98.89

Reduced and the Reduced-Size datasets. However, their performance on the Patch-100-Full and the Full-Size datasets was slightly lower than with the previous scheme for high-resolution datasets. The difference in performance between the proposed method and Rahmouni et al.'s was also substantially greater. The results also demonstrated the advantage of choosing among state-of-the-art classifiers to find the best one; i.e., use of the LDA classifier resulted in higher accuracy when the Patch-100-Mixed dataset was used. The ROC curves for the Reduced-Size and Full-Size dataset discriminators after being retrained are shown in Figures 9 and 10.

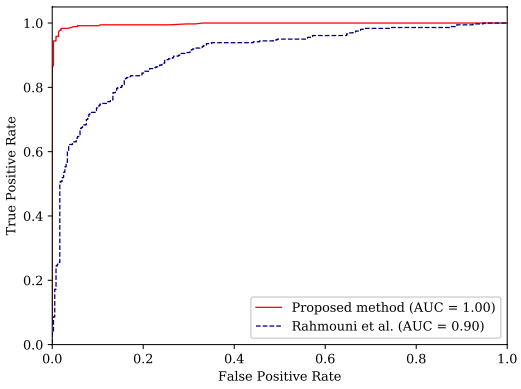

Figure 9: ROC curves of retrained discriminators tested on Reduced-Size dataset. Proposed method used LDA classifier.

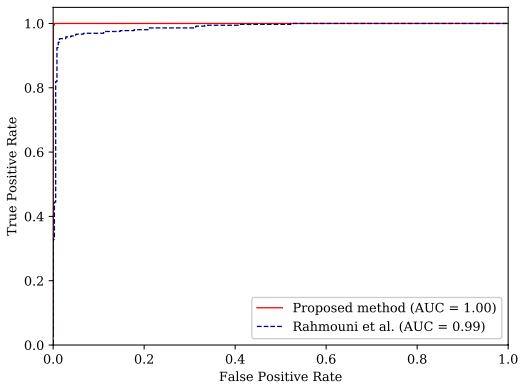

Figure 10: ROC curves of retrained discriminators tested on Full-Size dataset. Proposed method used LDA classifier.

5.4 Detecting Image Splicing

In an experiment, we used the discriminators to detect image splicing. Along with the normal way of dividing the test input into 100×100 patches, we also used an overlapping patch strategy. The probability of splicing for each area is the average of the probabilities of all patches to which the area belongs. Although this strategy has a higher calculation cost, it produces smoother output than the non-overlapping one. Example images are shown in Figure 11; the input sizes were 1800×1200 and 1200×800 pixels. Our proposed method (both overlapped and non-overlapped patches) outperformed Rahmouni et al.'s one [27]. Although our method did not flawlessly separate all the splices and had a few minor false positives, it could detect their relative positions. Rahmouni et al.'s one, on the other

hand, failed to detect the splice in the first image and was confused in the second image.

6 Summary and Future Work

The proposed modular CGI–PI discriminator uses the VGG-19 network as the feature extractor, statistical convolutional neural networks as the feature transformers, and the machine learning algorithm among state-of-the-art ones that has the best classification results as a discriminator. It outperformed a state-of-the-art CGI–PI discriminator. The proposed random sampling strategy used for patch aggregation was demonstrated to be effective for large images. Testing showed that using only high-resolution images for training is not sufficient to counter real-world attacks.

Our top priority now is to use ensemble adversarial training [35] to counter adversarial machine learning attacks [14]. This kind of attack is becoming more common and is very effective against machine-learning-based discriminators. A promising candidate to replace patch aggregation for dealing with high-resolution images is the attention-based approach [2]. We also plan to adapt the proposed discriminator to enable it to work with videos, not simply extracting data frame-by-frame and performing classification to reduce computational time.

Acknowledgement

This work was supported by JSPS KAKENHI Grant Numbers JP16H06302 and 18H04120.

References

[1] Oleg Alexander, Mike Rogers, William Lambeth, Jen-Yuan Chiang, Wan-Chun Ma, Chuan-Chang Wang, and Paul Debevec. The digital emily project: Achieving a photorealistic digital actor. *IEEE Computer Graphics and Applications*, 30(4):20–31, 2010.

[2] Jimmy Ba, Volodymyr Mnih, and Koray Kavukcuoglu. Multiple object recognition with visual attention. In *International Conference on Learning Representations (ICLR)*, 2015.

[3] Wen Chen, Yun Q Shi, Guorong Xuan, and Wei Su. Computer graphics identification using genetic algorithm. In *International Conference on Pattern Recognition*, pages 1–4. IEEE, 2008.

[4] Zhenwei Chen and Yongzhen Ke. A novel photographic and computer graphic composites detection method. In *National Conference on Information Technology and Computer Science*. Atlantis Press, 2012.

[5] Valentina Conotter and Lorenzo Cordin. Detecting photographic and computer generated composites. In *Image Processing: Algorithms and Systems IX*, volume 7870, page 78700A. International Society for Optics and Photonics, 2011.

[6] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[7] Duc-Tien Dang-Nguyen, Cecilia Pasquini, Valentina Conotter, and Giulia Boato. Raise: A raw images dataset for digital image forensics. In *Multimedia Systems Conference (MMSys)*, pages 219–224. ACM, 2015.

[8] Ahmet Emir Dirik, Sevinc Bayram, Husrev T Sencar, and Nasir Memon. New features to identify computer generated images. In *International Conference on Image Processing (ICIP)*, volume 4, pages IV–433. IEEE, 2007.

[9] Richard O Duda, Peter E Hart, David G Stork, et al. *Pattern classification*, volume 2. Wiley New York, 1973.

[10] Shaojing Fan, Rangding Wang, Yongping Zhang, and Ke Guo. Classifying computer generated graphics and natural images based on image contour information. *Journal of Information & Computational Science*, 9(10):2877–2895, 2012.

[11] Hany Farid and Siwei Lyu. Higher-order wavelet statistics and their application to digital forensics. In *Computer Vision and Pattern Recognition Workshop*, volume 8, pages 94–94. IEEE, 2003.

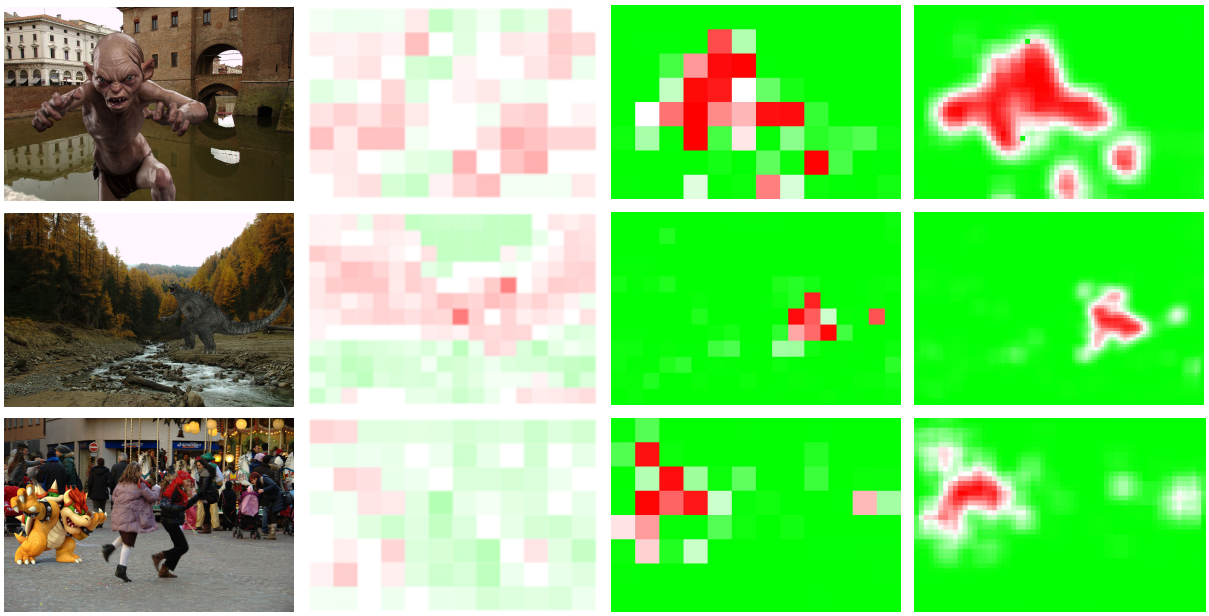

Figure 11: Three examples of splice detection. Patches detected as CGI are in red; those detected as PI are in blue. The color intensities illustrate the probabilities of classes.

[12] Andrew C Gallagher and Tsuhan Chen. Image authentication by detecting traces of demosaicing. In *Computer Vision and Pattern Recognition Workshops*, pages 1–8. IEEE, 2008.

[13] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems (NIPS)*, pages 2672–2680, 2014.

[14] Ling Huang, Anthony D Joseph, Blaine Nelson, Benjamin IP Rubinstein, and JD Tygar. Adversarial machine learning. In *Workshop on Security and artificial intelligence*, pages 43–58. ACM, 2011.

[15] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning (ICML)*, pages 448–456, 2015.

[16] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, pages 694–711. Springer, 2016.

[17] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *International Conference on Learning Representations (ICLR)*, 2018.

[18] Nitin Khanna, George T-C Chiu, Jan P Allebach, and Edward J Delp. Forensic techniques for classifying scanner, computer generated and digital camera images. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1653–1656. IEEE, 2008.

[19] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a

generative adversarial network. *Computing Research Repository (CoRR)*, 2016.

[20] Wenxiang Li, Tao Zhang, Ergong Zheng, and Xijian Ping. Identifying photorealistic computer graphics using second-order difference statistics. In *International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)*, volume 5, pages 2316–2319. IEEE, 2010.

[21] Zhaohong Li, Jingyu Ye, and Yun Qing Shi. Distinguishing computer graphics from photographic images using local binary patterns. In *International Workshop on Digital Forensics and Watermarking (IWDW)*, pages 228–241. Springer, 2013.

[22] Siwei Lyu and Hany Farid. How realistic is photorealistic? *IEEE Transactions on Signal Processing*, 53(2):845–850, 2005.

[23] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *International conference on machine learning (ICML)*, pages 807–814, 2010.

[24] Tian-Tsong Ng and Shih-Fu Chang. An online system for classifying computer graphics images from natural photographs. In *Security, Steganography, and Watermarking of Multimedia Contents VIII*, volume 6072, page 607211. International Society for Optics and Photonics, 2006.

[25] Fei Peng and Die-lan Zhou. Discriminating natural images and computer generated graphics based on the impact of cfa interpolation on the correlation of PRNU. *Digital Investigation*, 11(2):111–119, 2014.

[26] Fei Peng, Die-lan Zhou, Min Long, and Xingming Sun. Discrimination of natural images and computer generated graphics based on multi-fractal and regression analysis. *AEU-International Journal of Electronics and Communications*, 71:72–81, 2017.

[27] Nicolas Rahmouni, Vincent Nozick, Junichi Yamagishi, and Isao Echizen. Distinguishing computer graphics from natural images using convolution neural networks. In *Workshop on Information Forensics and Security (WIFS)*. IEEE, 2017.

[28] Dennis W Ruck, Steven K Rogers, Matthew Kabrisky, Mark E Oxley, and Bruce W Suter. The multilayer perceptron as an approximation to a bayes optimal discriminant function. *IEEE Transactions on Neural Networks*, 1(4):296–298, 1990.

[29] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[30] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations (ICLR)*, 2015.

[31] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[32] Supasorn Suwajanakorn, Steven M Seitz, and Ira Kemelmacher-Shlizerman. Synthesizing obama: learning lip sync from audio. *ACM Transactions on Graphics (TOG)*, 36(4):95, 2017.

[33] DQ Tan, XJ Shen, J Qin, and HP Chen. Detecting computer generated images based on local ternary count. *Pattern Recognition and Image Analysis*, 26(4):720–725, 2016.

[34] Justus Thies, Michael Zollhöfer, Marc Stamminger, Christian Theobalt, and Matthias Nießner. Face2Face: Real-time face capture and reenactment of RGB videos. In *Computer Vision and Pattern Recognition (CVPR)*, pages 2387–2395. IEEE, 2016.

[35] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and

defenses. In *International Conference on Learning Representations (ICLR)*, 2018.

[36] Ying Wang and Pierre Moulin. On discrimination between photorealistic and photographic images. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 2, pages II–II. IEEE, 2006.

[37] Ruoyu Wu, Xiaolong Li, and Bin Yang. Identifying computer generated graphics via histogram features. In *International Conference on Image Processing (ICIP)*, pages 1933–1936. IEEE, 2011.

[38] Rong Zhang, Rang-Ding Wang, and Tian-Tsong Ng. Distinguishing photographic images and photorealistic computer graphics using visual vocabulary on local image edges. In *International Workshop on Digital Forensics and Watermarking (IWDW)*, pages 292–305. Springer, 2011.