Copy Sensitive Graphical Code Estimation: Physical vs Numerical Resolution

Rohit Yadav, Iuliia Tkachenko, Alain Trémeau, Thierry Fournel

Copy Sensitive Graphical Code Estimation: Physical vs Numerical Resolution

Rohit Yadav, Iuliia Tkachenko, Alain Trémeau, Thierry Fournel
Laboratoire Hubert Curien, UMR CNRS 5516, Université de Lyon, UJM-Saint-Etienne
18 rue Professeur B. Lauras, 42000, Saint-Etienne, France
Email: fournel@univ-st-etienne.fr

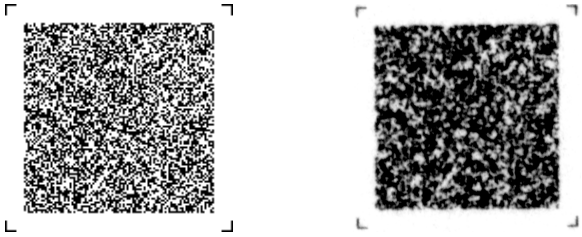
***Abstract*—Recent papers point out the vulnerability of Copy Sensitive Graphical Codes (CSGC) while an opponent uses a neural network approach to estimate a pattern then prints it as an original one: such a fake can successfully pass the authentication test. Here, we show that a GAN-like network can be even more powerful. A SRGAN-based architecture including super-resolution can tolerate a lower scanner resolution and decode efficiently. Besides, the use of such a decoding technique to perform the authentication test can improve the resistance of CSGC to estimation attacks.**

I. Introduction

The number of counterfeited valuable documents (as diplomas, bills, tickets) and packaging (for luxury products and medicines) increases each year in a large part due to the development and accessibility of even more efficient print and scan devices. The protection of such manufactured objects is for a long the goal of many research works and developments. For this purpose different techniques have been deployed. Optical watermarks and special means as specialized inks and substrates with a controlled distribution are among the most popular. To fight against hard copy, a more recent way has been suggested: the Copy Sensitive Graphical Codes (CSGC) [1], [2]. Such a graphical code (see for instance Fig. 1.a) is a digital hard-to-predict pattern designed to be highly sensitive to the stochastic nature of Print-and-Scan (P&S) process. As shown in Fig. 1.b, it can severely impact the CSGC structure. Nevertheless some statistical learning techniques [3], [4], [5] have shown that CSGC can be efficiently estimated after P&S. Such vulnerabilities are even more salient when the learning stage makes use of some artificial neural networks [6], [7].

In this paper, we consider a generative model to reverse engineer the P&S process. CSGC are first estimated (decoded) from images acquired at different scan resolutions, in order to pixel-wise compare with the original versions and get Bit Error Rate (BER). The lower resolution here considered is 2400 spi (samples per inch), a middle one for standard scanners, corresponding to the physical resolution of the scanner used for the experiments. Resolutions $\times 2$ (4800 spi), and $\times 4$ (9600 spi) involving numerical interpolations are also considered. Besides, the CSGC resolution can numerically be increased by Super-Resolution (SR) technique. It was shown in [8] that the use of the multi-frame SR technique [9] can increase the quality of P&S images and, thus, increase the gap between the images printed once (original CSGC) and the images printed twice (duplicated CSGC) during authentication test.

WIFS'2019, December, 9-12, 2019, Delft, Netherlands. 978-1-7281-3217-4/19/$31.00



(a) (b)

Fig. 1. Example of CSGC pattern: (a) an original random binary image before printing (I) and (b) its (degraded) gray level version after P&S ($\tilde{I}$).

We suggest here to slightly modify a state-of-the-art single image super-resolution architecture named SRGAN (as Super Resolution Generative Adversarial Networks) [10], to upscale as well as binarize output images. In the resulting deep architecture, upscaling is an option. Either the dedicated layers are used (we will name the network, SRGANb) or not (we will name the network, srGANb) such that we can compare physical resolution and numerical resolution with globally the same architecture. At any of these resolutions, the results on srGANb are compared with those of the state-of-the-art, namely with Selectional Auto-Encoder (SAE) which performs well CSGC decoding [7]. Learning GAN architectures as SRGANb or srGANb, is learning a pair of networks which are in competition with each other to minimize the loss such that the generated data are as similar as possible to real data. In the GAN literature the first network is called a Generator (G_{θ_G}) and the other a Discriminator (D_{θ_D}). The discriminator acts as an expert whereas the generator acts as a forger. Such a learning makes use of a dataset of pairs consisting each in a printed-and-scanned CSGC at a given resolution and an original (binary) CSGC at either the same or higher resolution. In practice to get an efficient forger network, an opponent needs an access to the legal printer or to a printer having very close characteristics (the same trade mark and model; no access to any high resolution scanner is required: the binary images are zoomed out) during the learning stage. The use of the same printer during an attack is not a scenario valid

(otherwise the printer could be used by anybody also for attacking) but it corresponds to a superior bound concerning the impact of any copy attack.
The rest of this paper is organized as follows. The authentication system is described in Section II. The SRGAN implementation is presented in Section III. We discuss the obtained experimental results in Section IV. Finally, we conclude in Section V.

II. Authentication system based on CSGC

The CSGC studied in this paper are random black-and-white patterns composed with elementary units. During a first step, any generated CSGC I is inserted in artwork then printed using the legal printer of an authority center, denoted as Π. Each elementary unit of the current CSGC is reproduced with a number of $u \times u$ pixels depending on the printer resolution. In the CSGC principle, this number needs to stay close to 1 (native resolution of the printer) to be maximally sensitive to the noise inherent to the printer. It can be more than 1 for reading performance reasons. During verification, the printed CSGC is scanned at the same or at higher resolution. Consequently, each elementary unit in the printed-and-scanned image $\tilde{I}$ has $v \times v$ pixels, where $v \geq u$. Obviously, if the resolution of the scanner is equal to that of the printer, the printed-and-scanned units have the same size as the printed ones: $u = v$. An opponent can try to estimate and retrieve the original binary CSGC by scanning the genuine printed version before re-printing in order to produce a fake one. Such an estimation attack consists of the following steps: 1) scanning at a resolution v_o, where $v_o \geq u$, 2) image processing including a binarization process (as digital printers available on the market can only print black-and-white images), using a statistical approach [5] or machine learning techniques as SVM, LDA [3] and neural networks [6], [7], 3) printing the estimated CSGC, $\hat{I}$ at the same print resolution as the generation stage. The considered authentication system is presented in Fig. 2 (the last processing dedicated to the authentication test is not represented here), where the authentic channel is illustrated using green lines and the opponent channel is illustrated using red dashed lines.

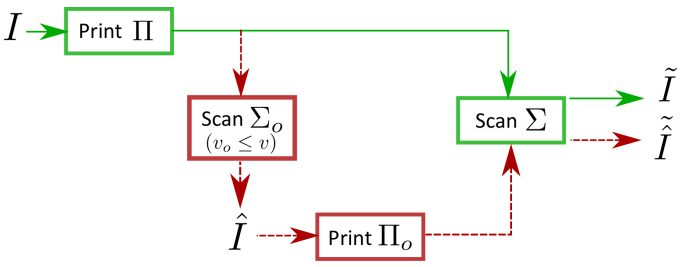


Fig. 2. Legal channel (green lines) and opponent one (red dashed lines) of the considered authentication system.

The authentication test that ends the verification process, has to distinguish an authentic from a fake: a scan of the current printed CSGC (grayscale image $\tilde{I}$ if authentic, $\tilde{\hat{I}}$ if not) is compared with the original CSGC (I). For this purpose, the Pearson correlation is currently used. If the score is higher than a pre-calculated authentication threshold ϵ, the CSGC is considered as genuine, otherwise, it cannot be considered as such. The considered authentication test can be defined as a hypothesis test:

$$\mathcal{H}_0 : cor(I, \tilde{I}) \geq \epsilon,$$
$$\mathcal{H}_1 : cor(I, \tilde{I}) < \epsilon,$$

where cor is a Pearson correlation value between binary image I and grayscale image $\tilde{I}$ after scanning.

III. Neural approach with resolution management

The most straightforward strategy for estimating a binary CSGC using deep networks is to formulate the task as image to image translation with a binarization objective, where the input is a grayscale CSGC and the output is its estimated binary version. Authors in [6] proposed a similar approach based on fully connected networks. Their investigation on different architectures led to the conclusion that the estimation process suffers from a bottleneck that makes it not optimal. The underlying process basically requests processing each input pixel for classification. Taking advantage of recent advancements in deep networks, the authors of [7] suggested the use of a fully convolutional auto-encoder that has previously demonstrated promising results in binarization of degraded manuscripts [11]. Additional use of residual connection in the architecture provides an opportunity to efficiently train the deep network.
The extensive use of deep learning approaches based on Convolution Neural Networks (CNN) has led to development of many successful architectures. One such architecture is Generative Adversarial Networks (GAN) [12]. GAN are one of the emerging and widely studied deep learning architectures. These GAN have been previously applied in variety of domains for tasks such as image generation, image Super-Resolution (SR), image translation and image segmentation. Here, we propose the use the architecture of Super-Resolution GAN (SRGAN) [10] but for binarization of CSGC along with the management of their resolution in a single process. This way allows taking into account with globally the same architecture, the case where the opponent scanner has a lower resolution to be numerically compensated (i.e. $v_o < v$).
The initial objective of SRGAN is to generate high-resolution images given a low resolution version. Similar to every GAN architecture, SRGAN also comprises of a Generator (that can be considered as a forger) and a Discriminator (that can be considered as an expert). However the Generator G_{θ_G} in SRGAN is a very deep convolution network with 16 residual blocks (see illustration in Fig. 3). Each residual block has two layers of convolution and two layers of batch normalization with ReLU activation. For each convolution in the generator we use 64 feature maps with 3×3 kernel except for first and last convolutions, where kernel size was set to 9×9. In SRGAN [10], low resolved images can be super resolved to $4\times$, this is done by the use of two upscaling blocks

in the architecture. Our implementation of upscaling blocks comprises of convolution and an upsampling layer with ReLU activation. For each convolution in the upscaling block, the use of feature maps was extended to 256 with 3×3 kernel. In the case where we need to (resp. not) enhance the resolution by just $2\times$, we don't use the last upsampling layer (resp. the two last upsampling layers) in the architecture.

The architecture of discriminator D_{θ_D} on the other hand has been inspired by [13]. In our implementation of D_{θ_D}, for each convolution in the discriminator we fixed the number of feature maps to 64 for low computational cost. Each convolution is followed by batch normalization and Leaky ReLU as activation. Kernel size was set to 3×3 for each convolution. The last convolution in D_{θ_D} is followed by two fully connected layers with Leaky ReLU and Sigmoid activation.

Loss l defined in SRGAN is crucial for the generation of natural looking CSGC. The loss l is basically defined as the weighted sum of VGG loss $l_{VGG/i.j}$ and adversarial loss l_{adv}:

$$l = l_{VGG/i.j} + 10^{-3} l_{adv}. \tag{1}$$

VGG loss is computed by parsing the generated high resolution binary image $G_{\theta_G}(\tilde{I})$ and its corresponding ground-truth I through the pre-trained 19 layers VGG network, therefore calculating the euclidean distance between their feature maps, where $\phi_{i,j}(\cdot)$ describes the feature map obtained after activation of j^{th} convolution before the i^{th} max-pooling and $W_{i,j}$, $H_{i,j}$ represent the dimensions of feature maps:

$$l_{VGG/i.j} = \frac{1}{W_{i,j} H_{i,j}} \sum_{x=1}^{W_{i,j}} \sum_{y=1}^{H_{i,j}} (\phi_{i,j}(I)_{x,y} - \phi_{i,j}(G_{\theta_G}(\tilde{I}))_{x,y})^2. \tag{2}$$

Additional to $l_{VGG/i.j}$ the adversarial loss l_{adv} defined below encourages the generator G_{θ_G} to generate high resolution binary CSGC which are close to the ground-truth I:

$$l_{adv} = \sum_{n=1}^{N} -\log D_{\theta_D}(G_{\theta_G}(\tilde{I})). \tag{3}$$

IV. Experimental results

A. Database description

In our experiments, we use the public database of CSGC [1]. This database contains of 950 random binary images of 100×100 dot size in 600 ppi (pixels per inch). The density of black dots is fixed to $48 - 52\%$. These CSGC were printed by laser printer Xerox Phaser 6500 with a true 600 dpi (dots per inch) resolution ($u = 1$). The real size of printed CSGC is 4×4 mm that is the acceptable size for such kind of codes while it is used for packaging protection.

These printed CSGC were scanned using a scanner Epson Perfection V850 Pro at 1) a true resolution of 2400 spi ($v = 4$), then 2) at 4800 spi ($v = 8$) and 3) at 9600 spi ($v = 16$) by using some numerical interpolation techniques. This database was divided into a training dataset (650 images), a validation dataset (100 images) and a test dataset (200 images).

B. Implementation details

Our implementation of SRGAN were declined in three variants: SRGANb $\times 4$, SRGANb $\times 2$, srGANb which correspond to binary outputs $v = 16$, $v = 8$ and $v = u = 4$, respectively. All the variants include binarization, only the first two include upscaling ($\times 4$ and $\times 2$, respectively). For learning these architectures, gray level image $\tilde{I}$ was cropped (see an example in Fig. 5.b) to 100×100 (whatever is the scan resolution) and normalized to values $[0, 1]$ for each CSGC binary source image I (see an example in Fig. 5.a). Each ground-truth 100×100 image I was digitally zoomed in into 200×200 (400×400 resp.) for $2\times$ ($4\times$ resp.) upscaling.

For Selectional Auto-Encoder (SAE) [7] crop window was set to 128×128 pixels and size of kernel was set to 4×4 when $v = 4$, 8×8 when $v = 8$ and 16×16 when $v = 16$.

For SRGAN we defined $l_{VGG/11.10}$ with $\phi_{11,10}$, a loss defined on very deep feature maps. The VGG implementation was adapted from keras [2] pre-trained network trained on imagenet database.

During training, we evaluate the performance after every few mini batch iteration on validation set with mean Bit Error Rate (BER) as metric. We compare 100×100 pixels [down-sampled] decoded CSGC with 100×100 pixels digital codes of validation set. After complete training of SR\srGANb the generator can be used to convert any $\tilde{I}$ to a [high resolution] binarized CSGC $\hat{I}$.

The pipeline of studied estimation attack is presented in Fig. 4. The authentic printed CSGC is scanned with equal or smaller resolution. The obtained grayscale CSGC is binarized via trained SR\srGANb at the desired resolution.

To down-sample the high resolution binarized images obtained from the generator we merge the predicted 100×100, 200×200 or 400×400 patches to create a full image $\hat{I}$ of size 400×400, 800×800, and 1600×1600 pixels, respectively.

Due to the use of sigmoid activation at the last layer, the images generated by the generator have pixel values either close to 0 or to 1 (an example illustrated in Fig. 5.c). It is therefore sufficient to set a global threshold to $t = 0.5$ for image binarization (Fig. 5.e). A majority vote is then applied on every $v \times v$ modules ($v = 4, 8, 16$) to label each pixel as either black or white and obtain the estimation result (Fig. 5.f).

For optimization, Adam method was used with a learning rate of 0.0002 and $\beta_1 = 0.5$. Training for both SRGAN and SAE was done for 20 epochs on a NVIDIA Geforce GTX 1080 GPU card.

[1] The database can be found in www.univ-st-etienne.fr/graphical-code-estimation

[2] https://github.com/keras-team/keras-applications/blob/master/keras_applications/vgg19.py

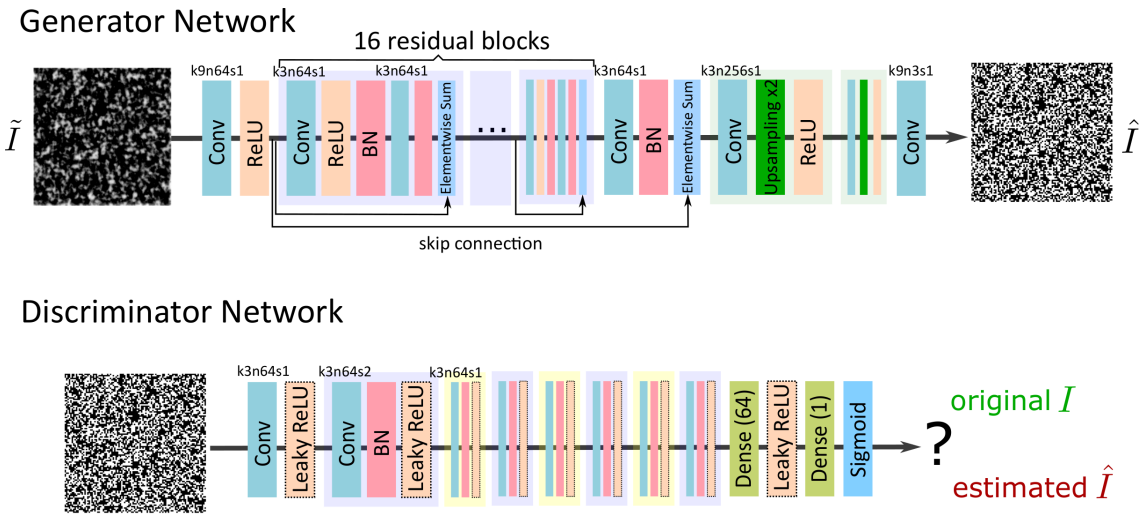


Fig. 3. Image resolution management and binarization of CSGC using SRGAN architecture, where n is the number of feature maps, s is the stride at each convolution layer and k is a size of filter for convolution. This scheme represents the case when the image resolution is increased $\times 4$. If we need to increase the image resolution $\times 2$, we just need to delete one operation of upsampling (green block) from this scheme. Keeping the image resolution constant can be achieved similarly.

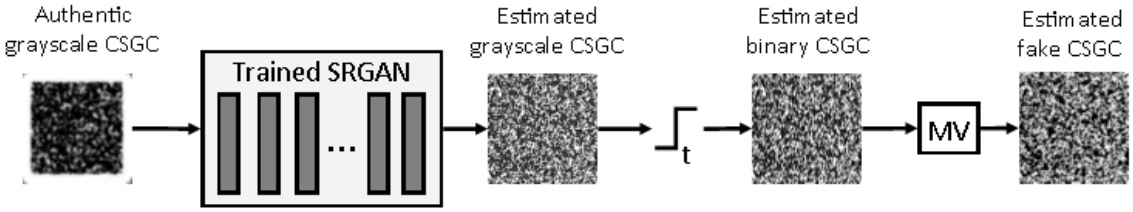


Fig. 4. Pipeline of studied CSGC estimation attack: estimation from scanned authentic graylevel CSGC using trained SR\srGANb network, binarization of estimated CSGC, construction of fake CSGC by applying majority vote (MV) to a up-sampled estimated binary CSGC.

C. Decoding using neural approach

The printed-and-scanned CSGC in the test dataset were decoded after learning first with the state-of-the-art architectures i.e. BN DNN [6] and SAE [7] for $v = 4, 8, 16$, respectively (Table I). Crop windows were set to 25×25 pixels (25600 patches for $v = 4$, 102400 patches for $v = 8$) and 128×128 pixels (1600 patches for $v = 4$, 4900 patches for $v = 8$), respectively, to avoid bias in the number of patches. Filter size for SAE [7] was set to 4×4 when $v = 4$ and 8×8 when $v = 8$. These results can be directly compared with the results obtained with srGANb at the corresponding scan resolutions, which themselves can be compared with SRGANb ones involving an super-resolution. The effectiveness of these decoding methods is assessed via BER (Table II).

Architecture	v	BER	Std	Best case	Worst case
BN DNN	4	17.92%	1.40%	15.32%	22.91%
SAE	4	11.26%	1.59%	8.83%	19.84%
BN DNN	8	14.2%	1.06%	12.06%	17.28%
SAE	8	10.04%	0.82%	8.32%	13.65%
SAE	16	10.42%	1.02%	8.53%	15.51%

TABLE I
BER FOR SAE AND BN DNN AT DIFFERENT SCANNING RESOLUTIONS.

Architecture	v	BER	Std	Best case	Worst case
srGANb	4	9.20%	0.98%	7.28%	13.43%
SRGANb	4 to 8	9.27%	1.04%	7.14%	15.47%
srGANb	8	8.48%	0.66%	6.88%	10.33%
SRGANb	4 to 16	9.18%	0.96%	7.52%	15.21%

TABLE II
BER FOR SRGAN-BASED ARCHITECTURES AT DIFFERENT RESOLUTIONS, WITH OR WHITOUT SUPER-RESOLUTION.

The GAN-type architecture allows decoding with BER lower than 10%. It can be noticed that a numerical resolution of 9600 spi either from interpolation by the scanner or from super-resolution is not so powerful.

D. Authentication under attack

In this section, we report results of authentication test done under original and attacked CSGC. We want to study two different situations:

1) an attacker and an authentication center have a printer and a scanner with the resolutions 600 dpi and 2400 spi, respectively.
2) an attacker has a scanner with lower resolution (2400 spi) while an authentication center uses a high resolution scanner (with resolution 4800 spi or 9600 spi).

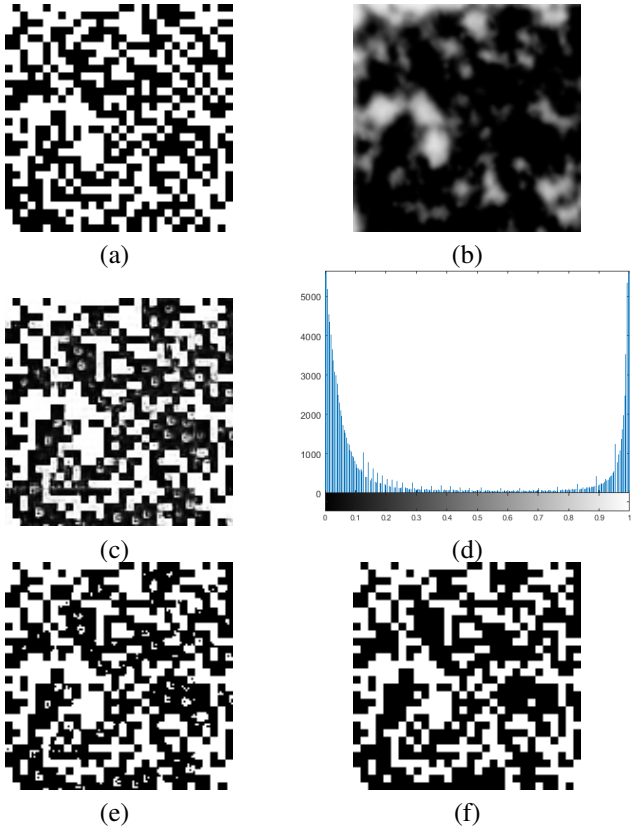


Fig. 5. Example of zoomed parts of a) an original black-and-white CSGC, b) a printed and scanned gray-level CSGC, c) an output image from neural network, d) its histogram, e) a binarized output image and f) its final version after majority vote.

To perform the attacks, the estimated CSGC $\hat{I}$ were printed with resolution 600 dpi. Than these printed codes are scanned with the resolution used in the authentication center. The mean correlation values for all tested attacks as well as for the original CSGC are presented in Table III.

Printed CSGC	$v = 4$	$v = 8$	$v = 16$
Original	**0.48** $\pm$ 0.01	**0.48** $\pm$ 0.01	**0.49** $\pm$ 0.01
SAE	0.44 $\pm$ 0.01	0.44 $\pm$ 0.01	0.45 $\pm$ 0.01
srGANb	0.47 $\pm$ 0.01	0.46 $\pm$ 0.01	-
SRGANb $\times 2$	0.45 $\pm$ 0.01	0.45 $\pm$ 0.01	-
SRGANb $\times 4$	0.46 $\pm$ 0.01	-	0.45 $\pm$ 0.02

TABLE III
CORRELATION SCORE COMPUTED FOR THE AUTHENTICATION TEST: FOR ORIGINAL CSGC ON ONE HAND, FOR FAKE CSGC USING SAE AND SR\SRGANB ESTIMATIONS ON THE OTHER HAND.

For illustrations of authentication results we use a Receiver Operation Characteristics (ROC) curve by plotting the dependency between False Positive Rate (FPR) and True Positive Rate (TPR). FPR is the percentage of printed fake CSGC that were considered by authentication test as authentic. TPR is the percentage of printed original CSGC that were considered by authentication test as authentic. The attack is successful when the curve is closer to the bisectrix (i.e. the lower line is the bigger number of fake samples can pass the authentication test).

The first experiment consists of CSGC that were printed with 600 dpi and scanned with 2400 spi. An attacker also has access to devices with the same resolutions. Thus an opponent training database was constructed using the CSGC printed and scanned with the same resolutions as the authentic codes. The ROC curve in Fig. 6 illustrates the effectiveness of SRGANb decoding (lines orange and gray) in comparison with SAE result.

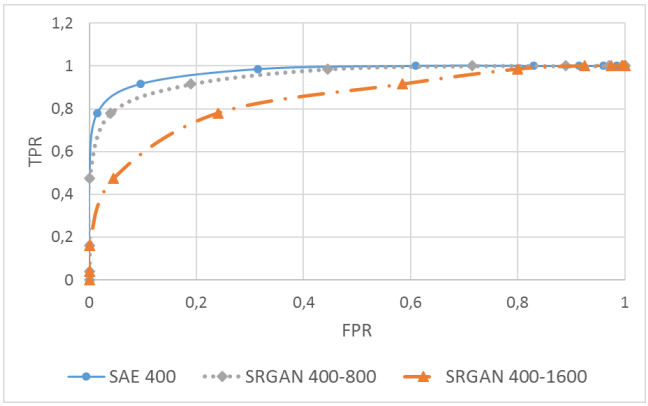


Fig. 6. ROC curve for CSGC scanned in 2400 spi.

We note that more than 20% of fake CSGC can pass the authentication test, while using SRGANb for $\times 4$ up-sampling. In the same time, only 80% of authentic CSGC are accepted. If the authentication threshold ϵ is higher, in order to accept all authentic CSGC, more than 80% of fake CSGC will be accepted too.

The second experiment consists of CSGC that were printed with 600 dpi and scanned with 4800 spi. In the same time an attacker has access only to a scanner with 2400 spi resolution, thus s/he will use SRGANb technique in order to increase the image resolution numerically. The image resolution was increased twice during decoding process (from images with $v_o = 4$ to images with $v = 8$). In ROC curve presented in Fig. 7, we can see that the number of SRGANb fake CSGC that can pass the authentication test is bigger than the number of accepted SAE fake CSGC. Nevertheless, thanks to the higher resolution of authentication center scanner, the correlation values of authentic printed CSGC is higher. Thus, the number of fake CSGC that pass the authentication test is smaller (in comparison with the previous attack test). The ROC curve in Fig. 8 illustrates the results while an attacker tries to increase numerically the resolution in four time, i.e. from images scanned with 2400 spi ($v_o = 4$) to images scanned with 9600 spi resolution ($v = 16$) as an authentication center uses the scanner with 9600 spi resolution. Still a bigger number of CSGC faked using SRGANb can pass the authentication test in comparison with fake CSGC estimated using SAE.

The ROC curves in Fig. 7 and Fig. 8 show us that only 20% of fake CSGC can pass the authentication test, when ϵ is sufficiently high to accept all authentic CSGC. This can be explained by the high resolution of authentic scanner that

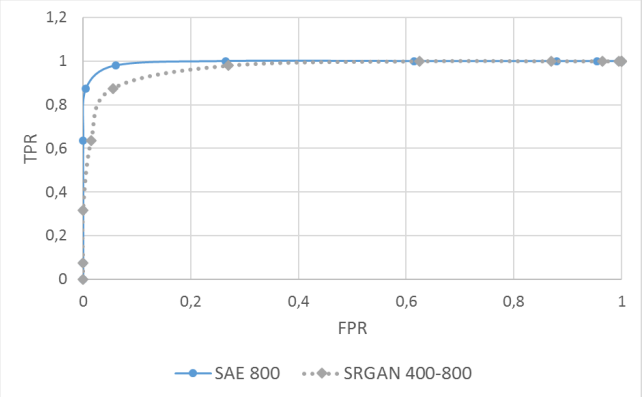


Fig. 7. ROC curve for CSGC scanned in 4800 spi.

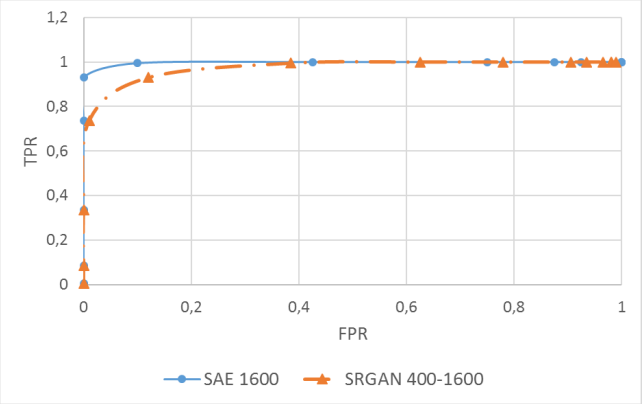


Fig. 8. ROC curve for CSGC scanned in 9600 spi.

can catch more changes that are produced during estimation attack. Nevertheless, these results show the vulnerability of the studied authentication system as possibly a bigger training database or a longer training process will allow an attacker to fake the CSGC even without high resolution scanning devices.

E. Decoding using SRGANb for authentication

One of the possible reinforcements of the studied authentication system is the use of authentication test which is more adapted for such estimation attacks. In this section, we show the authentication results while the decoding with SRGANb is used as an authentication test. The mean correlation values and mean BER values are presented in Table IV.

Printed CSGC	After SRGAN decoding	
	Correlation	BER
SRGANb $\times 2$		
Original	**0.82**	**9.28%**
Attack SRGANb	0.67	16.53%
SRGANb $\times 4$		
Original	**0.82**	**9.18%**
Attack SRGANb	0.69	15.54%

TABLE IV
AUTHENTICATION USING SRGAN DECODING.

From this table we can see that the difference of mean correlation values between original and attacked CSGC is more than 0.13. That is a big gap in comparison with the values that we have in Table III. If we use the BER as an authentication test, the gap will be even bigger as reported in Table IV. These results show us that the use of precise decoding method can improve the robustness of studied authentication system. In addition, the decoding methods using neural approach can efficiently detect the fake CSGC.

V. CONCLUSION

SRGAN-based architectures jointly achieving binarization can partly compensate a lower resolution when scanning, and can powerfully decode printed-and-scanned CSGC. Thanks to such an architecture, we have been able to fall BER below the threshold of 10%. The measured ROC curves when re-printing with the legal printer gives a bound for an estimation attack performed in this way. Besides, SRGAN-decoding allows an increase of the gap between original and fake graphical codes, and an improvement of the resistance to an estimation attack.

REFERENCES

[1] J. Picard, "Digital authentication with copy-detection patterns," in *Electronic Imaging 2004*. International Society for Optics and Photonics, 2004, pp. 176–183.

[2] I. Tkachenko, W. Puech, O. Strauss, C. Destruel, and J.-M. Gaudin, "Printed document authentication using two level or code," in *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE, 2016, pp. 2149–2153.

[3] M. L. Diong, P. Bas, C. Pelle, and W. Sawaya, "Document authentication using 2d codes: Maximizing the decoding performance using statistical inference," in *IFIP International Conference on Communications and Multimedia Security*. Springer, 2012, pp. 39–54.

[4] . C. F. Baras, C., "Towards a realistic channel model for security analysis of authentication using graphical codes." in *Workshop on Information Forensics and security (WIFS), IEEE*, pp. 115–119.

[5] I. Tkachenko and C. Destruel, "Exploitation of redundancy for pattern estimation of copy-sensitive two level QR code," in *Workshop on Information Forensics and security (WIFS), IEEE*, 2018.

[6] O. Taran, S. Bonev, and S. Voloshynovskiy, "Clonability of anti-counterfeiting printable graphical codes: a machine learning approach," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Brighton, United Kingdom, May 2019.

[7] R. Yadav, I. Tkachenko, A. Trémeau, and T. Fournel, "Estimation of copy-sensitive codes using a neuronal approach," in *ACM workshop on Information hiding and multimedia security*, Paris, France, June 2019.

[8] I. Tkachenko, F. Kucharczak, C. Destruel, O. Strauss, and W. Puech, "Copy sensitive graphical code quality improvement using a super-resolution technique," in *2018 25th IEEE International Conference on Image Processing (ICIP)*. IEEE, 2018, pp. 3808–3812.

[9] F. Graba, F. Comby, and O. Strauss, "Non-additive imprecise image super-resolution in a semi-blind context," *IEEE Transactions on Image Processing*, vol. 26, no. 3, pp. 1379–1392, 2016.

[10] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, and Z. Wang, "Photo-realistic single image super-resolution using a generative adversarial network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4681–4690.

[11] J. Calvo-Zaragoza and A.-J. Gallego, "A selectional auto-encoder approach for document image binarization," *Pattern Recognition*, vol. 86, pp. 37–47, 2019.

[12] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[13] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *CoRR*, vol. abs/1511.06434, 2016.