IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication Access-2021-34942 29-Oct-2021, date of current version Access-2021-34942 29-Oct-2021.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

CodeFace: a deep learning printer-proof steganography for Face Portraits

FARHAD SHADMAND[1], IURII MEDVEDEV [2], AND NUNO GONÇALVES.[3, 4]

[1]University of Coimbra, Institute of Systems and Robotcs, Coimbra, Portugal (e-mail: farhad.shadmand@isr.uc.pt)
[2]University of Coimbra, Institute of Systems and Robotcs, Coimbra, Portugal (e-mail: iurii.medvedev@isr.uc.pt)
[3]University of Coimbra, Institute of Systems and Robotcs, Coimbra, Portugal
[4]INCM Lab - Portuguese Mint and Official Printed Office, Lisbon, Portugal (email:nunogon@deec.uc.pt)

Corresponding author: Farhad Shadmand (e-mail: farhad.shadmand@isr.uc.pt).

ABSTRACT Identity Documents (IDs) containing a facial portrait constitute a prominent form of personal identification. Photograph substitution in official documents (a genuine photo replaced by a non-genuine photo) or originally fraudulent documents with an arbitrary photograph are well known attacks, but unfortunately still efficient ways of misleading the national authorities in in-person identification processes. Therefore, in order to confirm that the identity document holds a validated photo, a novel face image steganography technique to encode secret messages in facial portraits and then decode these hidden messages from physically printed facial photos of Identity Documents (IDs) and Machine-Readable Travel Documents (MRTDs), is addressed in this paper. The encoded face image looks like the original image to a naked eye. Our architecture is called CodeFace. CodeFace comprises a deep neural network that learns an encoding and decoding algorithm to robustly include several types of image perturbations caused by image compression, digital transfer, printer devices, environmental lighting and digital cameras. The appearance of the encoded facial photo is preserved by minimizing the distance of the facial features between the encoded and original facial image and also through a new network architecture to improve the data restoration for small images. Extensive experiments were performed with real printed documents and smartphone cameras. The results obtained demonstrate high robustness in the decoding of hidden messages in physical polycarbonate and PVC cards, as well as the stability of the method for encoding messages up to a size of 120 bits.

INDEX TERMS steganography, Machine-Readable Travel Documents, deep neural network, Hiding message into images

I. INTRODUCTION

IN this paper, we present a new facial image steganography method for transmitting secret messages through facial images. That "encoded" information can be later "decoded", independently of the image format, either digital or printed and of the transmission media, if any. If the encoded image is printed and captured by a digital device, the decoded algorithm must be prepared to deal with several sources of noise introduced by the physical and digital means. The architecture of our method is called CodeFace and it is schematically depicted in Figure 1, that presents, respectively, the encoder and the decoder in the left and right black rectangles. The secret message content is encoded inside the facial image is robust to physical distortions of the image carrier and other sources of noise and error. This is achieved through a careful design of a noise simulation module whose parameters are learned by the decoder. This message, which is not visible to the naked eye, can be captured by a digital camera of a ubiquitous mobile device and further detected and decoded by a validation algorithm through the use of deep learning methods.

IDs and MRTDs (Identification and Machine Readable Travel Documents) are used to identify and authenticate identities in several scenarios such as crossing national borders, in civil applications, sales and purchasing portals, or admission to transaction processing systems. These documents have several security features which mitigate and combat document forgery. As these security systems are difficult to circumvent, criminal attacks on ID verification systems are now focusing on fraudulently obtaining genuine documents and the manipulation of the facial portraits. To reduce risks related to this fraud problem, it is necessary that governments and manufacturers of IDs and MRTDs continuously develop and improve security measures. With this in mind, we in-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3132581, IEEE Access

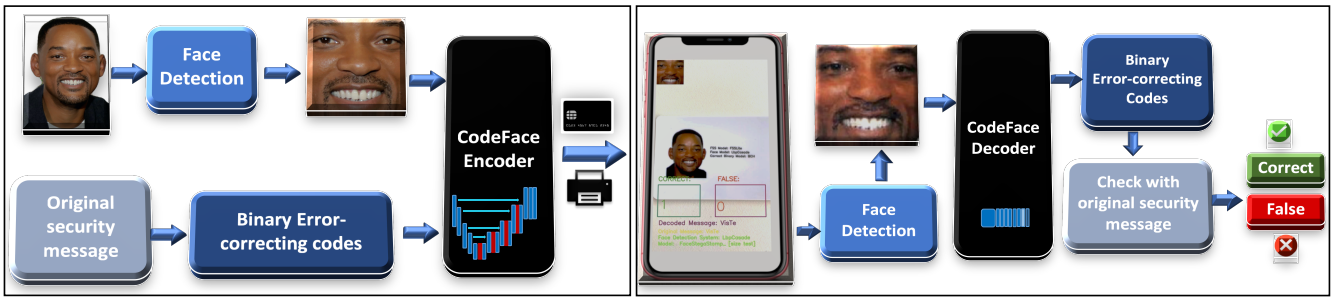


FIGURE 1. The implementation of CodeFace is performed in two pipelines, implemented using independent networks, namely the encoder and the decoder. In the encoder pipeline, a message intended to be secret is first encoded with a Binary Error-Correcting Code algorithm and thus concealed in the detected facial image. This encoded image is used for the issuance of an ID document that may be validated with a smartphone camera. The developed mobile application processes the captured ID document images using the decoder pipeline. In this pipeline, the CodeFace network then extracts the binary secret message from the image of the detected face, and the same Binary Error-Correcting Code algorithm translates it to a string. Finally, the original and extracted messages are compared to confirm the integrity of the ID document.

troduce the first efficient steganography method - CodeFace - which is optimized for facial images printed in common IDs and MRTDs. CodeFace is an end-to-end Generative Adversarial Network (Figure 2) that is formed by an Encoder (Figure 3), that can conceal a secret message in a face portrait and, hence, producing the encoded image, and a Decoder (Figure 4), which is able to read a message from the encoded image, even if it is previously printed and then captured by a digital camera.

The key advantages of our model are: 1) it introduces a new security system for encoding and decoding facial images that are printed in common IDs and MRTDs, 2) the differences between the encoded and original facial images are not evident to the naked eye (see Figure 5), 3) the authentication of persons, either using 1:1 verification or 1:N identification, can be performed using our model, 4) the system is suitable for and optimized to run on smartphones, 5) CodeFace surpasses state-of-the-art methods in allowing the use of images in their context, irrespectively of the background. This feature also allows us to use the method without any restrictions relating to photo parameters, as described in detail in Section III-A, 6) our system is able to decode secret messages from very small images (100×100 pixels).

Considering the existing steganography models summarized in section II, we have found three constraints that make them unsuitable as security verification systems for document portraits. Firstly, they fail to decode secret message from small, encoded images. Secondly, they do not preserve sufficiently the visual structure of the encoded face, thus introducing noticeable distortion in the appearance of the face, as shown in the comparison of Figure 5. The existing methods also introduce extra noise into the encoded facial images that affects the performance of biometric facial verification systems, as shown in the metrics presented in the plot A of Figure 6. Thirdly, as mentioned above, these methods require the message to be encoded in a full image. Consequently, the currently available steganography models - to the best of our knowledge - are not suitable security systems for application to IDs and MRTDs.

In order to overcome the aforementioned limitations, our method presents the following improvements. In first place, we improve the noise simulation module by adding a resize network (using down-sampling) to decrease the size of the decoder input image from 400×400 to 100×100 during the training. This new resize network increases the decoder performance when reading messages from smaller images. Then we add a perception loss function term that minimizes the facial embedding difference between the original and encoded images. The method for extracting high level features is based on a metric learning approach introduced in FaceNet [22]. Such facial embedding is usually used in various recognition tasks such as facial verification, identification, clustering, and can be easily implemented using standard conventional tools. This simple loss function implementation enhances the face structure and improves its perceptual appearance. Finally, we consider face detection models in both our encoder and decoder pipeline applications to enable the application of our method to an arbitrary part of the face, independently of the background.

Figure 1 presents an overview of the CodeFace application process. The application contains two separate systems, the encoder and decoder. The encoder receives a facial photo and an arbitrary secret message as input. Using a face detection method described further ahead in section III-A, the relevant part of the face is detected and cropped. In parallel, a secret message is translated to a binary message using a Binary Error-Correcting Codes algorithm, whose details are presented in section III-B. Then, the encoder network (a trained deep learning network) accepts a cropped facial image and a binary message as input and encodes this message into the facial image. The facial image is printed in IDs and MRTDs. The encoder network for training is described in section III-C. In the decoding process, a document image is first captured using a mobile camera, then the encoded part of the image (the portrait) is detected and cropped. The decoder network receives the cropped encoded face as input and recovers the binary message. A detailed description of this decoding process is presented in section III-D. Subsequently,

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3132581, IEEE Access

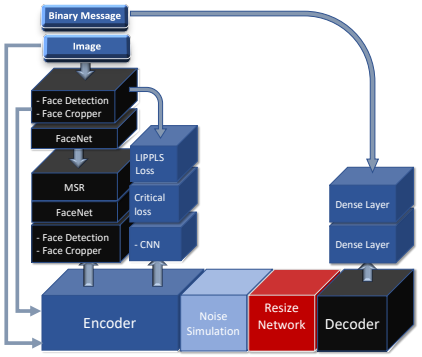


FIGURE 2. The CodeFace training network is made from generator and discriminator parts. The generator consists of an encoder, a decoder and a noise simulation network. The discriminator is a combination of face detection, alignment and cropping systems, a CNN and a Simple Dense layer (fast forward network). The face detection crops the encoded image for the FaceNet loss function. The discriminator also includes a LPIPS perceptual loss and Critical loss functions for the encoded images, and a Dense Layer and a cross entropy binary loss function for the decoded messages.

the same Binary Error-Correcting Code algorithm translates the binary message to a string with the secret message. Finally, the recovered message is analyzed and the integrity of the portrait is verified.

For the encoder and decoder networks training, the CodeFace structure uses a Generative Adversarial Network (GAN) [12] which is comprised by four components: loss functions module, Convolutional Neural Networks (CNN) for the encoder and the decoder and a noise simulation module. The encoder network has a set of loss functions consisting of perceptual loss [30], FaceNet [22], Wasserstein loss [3] and residual regularization [24]. These loss functions are designed to preserve the facial structure and color of the encoded face during training. We have implemented a complete noise simulation module to approximate the magnitude of the distortions resulting from real printing and digital imaging processes (capture) before the image is fed into the decoder. As far as the authors are aware, this is the first time a resize network is used to develop the noise simulation aspect, allowing the system to validate and decode small portrait images. The decoder is created using a Special Transfer Network (STN) [15] and a Convolutional Neural Network (CNN) that are trained by a cross-entropy loss function.

Summarizing, the need of algorithms to achieve better security in ID documents able to conceal secret information in face portraits and able to be used in a mobile application to validate the encoded information, and consequently able to decode information in printed items, has motivated the work herein described. As presented in the next section, our model is inspired in the StegaStamp [24], the first printer-proof steganography method, however not applicable to face portrait.

II. RELATED WORK

Image Steganography. The use of deep learning in steganography brought a disruptive change in its capabilities and applications. The state-of-the-art methods that take advantage of traditional methods (without deep learning), are discussed in [13], [21]. In this work, we focus on deep learning-based steganography techniques.

Image steganography using deep learning is a relatively new research area, typically based on opposing networks (mainly GANs) to encode and decode information. The most interesting methods to achieve this result are SteganoGAN [29] and HiDDeN [32]. The latter also adds a noise simulation network to improve the ability to recover images with distortion. The HiDDeN noise simulation component is implemented between the encoder and the decoder. The authors propose the noise simulation for a discrete cosine transform (DCT), a JPEG compression, a JPEG-Mask, and a JPEG-

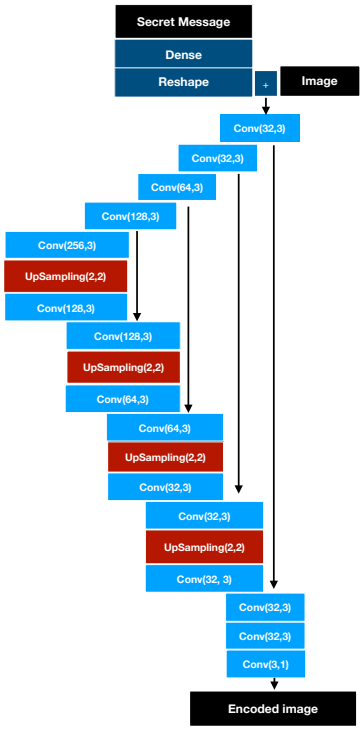


FIGURE 3. The encoder network is created by a UNet network with no pooling layers, and three convolutional layers.

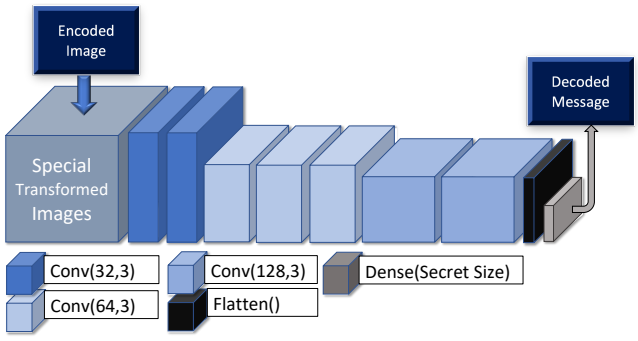


FIGURE 4. The decoder network is made from two CNNs with a Special Transformed Network (STN).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3132581, IEEE Access

Drop as distortion types for generating the noisy samples. However, the noise simulation model in SteganoGAN and HiDDeN have a rather simple formulation and they do not entirely consider other noise sources introduced by physical printing and capturing with a digital camera.

The first successful example of steganography with printed images was performed by StegaStamp [24], where the authors demonstrated how to achieve robust decoding of messages even under physical transmission. StegaStamp considers a set of different image corruptions between the encoder and the decoder that successfully approximates the set of distortions resulting from real printing transmission. Additionally, the method adds a Perceptual Loss function (the Learned Perceptual Image Patch Similarity, LPIPS) [30] that preserves the high quality of the images. According to our research, the performances of other models, such as LFW [13], HiDDeN [32] and SteganoGAN [29], are under 10% when decoding messages from small printed encoded face images. Therefore, in this work we only compare our model with StegaStamp. It was the first notable steganography model that could encode and decode hyperlinks in photos captured from real prints. Nevertheless, StegaStamp has some limitations in its application as a security element to verify the integrity of documents. Firstly, StegaStamp deals with relatively large, printed images that make it unsuitable for small facial photo applications, as the document security of IDs and MRTDs. Secondly, the encoded StegaStamp facial image has excessive noise when compared to the original image, as can be seen in Figure 5. This excessive noise also affects other biometric face verification systems as shown in the plots of Figure 6 for the Euclidean distance between the original and encoded image. Finally, the StegaStamp pipeline uses BiSeNet [28] to detect the encoded image, a neural network that does not have the ability to hide and read messages from specific parts of facial images.

CodeFace can overcome the aforementioned limitations. We apply a new loss function, inspired by FaceNet [22], to preserve the structure of encoded faces. We also introduced a resize network before the decoder as a new noise simulation module. This resize network (that performs downsampling of the input images) enables the decoder to read a message from small face images in the decoding process. Furthermore, we add the face detection [4], [7], [8] into the CodeFace application to enable it to hide and read the encoded message only within the image of the face.

Document security verification. The focus of this paper is on concealing security encoded data in ID and MRTD documents while allowing for the integrity verification of the portrait. In terms of document security, it is also important to maintain the system's ability to recognize persons using facial recognition algorithms. The main existing market solutions for the validation of ID cards and passport portrait photos using mobile devices are the Jura Digital IPI [18] and the IDEMIA Lasink [16], which are focused on altering the facial photo of the documents. Using the Jura Digital IPI as the data encoding technique, the photo is encoded by a digital technology and the secret information encoded in the portrait will only be visible when using a decoding device. The encoder uses a halftone process and the validation relies on the scanned image and the prior knowledge of the encoded message. In the IDEMIA Lasink, the image is modulated by a set of lines which encodes the secret information.

Other work related with ours is the VIPPrint [9], that recognizes a signature of a specific printer. Every printer introduces special and unique effects into its printed materials. VIPPrint model detects these effects and use them to build a validation system. While this is an effective model in practice, it is not a steganography model.

The ongoing research in facial recognition is typically focused on searching for the best facial representation. The state-of-the-art methods here referred typically utilize deep learning CNN based networks [7], [22], [23]. One of the closest works to our proposed system is the Medvedev's algorithm [20]. The authors introduce a portable and efficient biometric system for validating ID and travel documents. Their model consists of a machine-readable code, that is derived from the biometric template of a digital frontal facial image and printed on ID cards. In the validation process of the documents, the application reads and compares two biometric templates, one from the frontal face photo and the other from the machine-readable code. This work, however, does not disguise the information in the portrait and consequently the machine-readable code is visually available.

On the other hand, $1:1$ facial verification has largely been resolved [7], [22], [25], while $1:N$ facial identification solutions typically suffer from lower performance.

Our model can be used as a $1:N$ identification and a $1:1$ verification algorithm, with the ability to encode a unique security number for each individual in its document's facial photo. Although out of the scope of this article, the decoder can thus read this security number and find user's data on a sovereign database.

III. CODEFACE

The CodeFace is a model to encode and decode a secret message in facial images in the context of IDs and MRTDs. Our model is the first one to be designed as a security method for the verification of document portraits and it is inspired by steganography models such as [24], [32]. CodeFace is composed of two processes: the encoder and the decoder, as showed in Figure 1.

In the encoder, the facial image and the secret message are first received as inputs. The relevant part of the image is detected and cropped using a face detection model [4], [8]. Simultaneously, the secret message is coded by a binary error correcting codes algorithm [5], [11]. At the end of the encoder application, a pretrained encoder model embeds the message in the cropped face and produces an encoded facial image. The encoded cropped image then replaces the original facial image which is subsequently printed on an ID card.

As for the decoder, the ID card's encoded facial image is captured by a digital camera. The face detection module

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3132581, IEEE Access

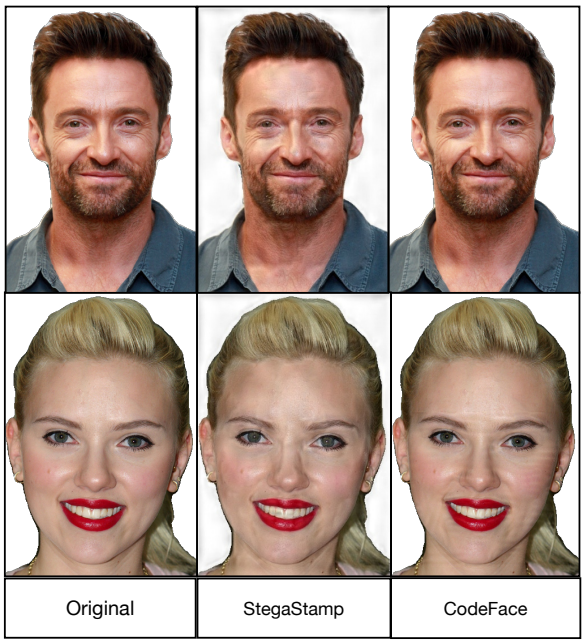


FIGURE 5. From left to right respectively: original images, StegaStamp encoded and CodeFace encoded are shown. All encoded images hide a random secret message. We encoded 100 bits within the facial image, which is a reasonable amount of information for security purposes.

then detects the encoded part of the facial image, which the CodeFace decoder network then receives, retrieving the hidden message. A binary-error codes algorithm converts the retrieved binary message into a number or a string. Then the final resulting message, the retrieved message, is checked using a hash function or checksum verification algorithm to validate the message, thus providing a way to check the integrity of the face portrait in IDs and MRTDs.

The CodeFace encoder and decoder networks are trained using GANs, whose structure is shown in Figure 2. It is composed of four parts: the encoder, decoder, noise simulation module and loss functions. The encoder and decoder networks are trained to hide and read messages in facial images while the noise simulation layers, included before the decoder, create a realistic environment for the complete network during the training. Loss functions consist of various pre-defined network components and additional loss functions that preserve the appearance of the encoded face and message during the training. In this section, we detail all components of the CodeFace GAN generator that are made from the encoder network, the noise simulation module and the decoder. In the final section, CodeFace's discriminator is described, including the loss functions and related networks.

A. FACE DETECTION

For a robust ID verification process that conceals a message in the facial image, we need a face detection model to identify the part of the face where the secret message is hidden. It is important to note that the facial detection model should reveal the exact part of the face used to encode information. To achieve this, we applied and carried out extensive tests using a set of facial detection approaches such as BlazeFace [4], MobileNets V2 float32, MobileNets V2 int8 [14], SSD int8 MTCNN [19], LBP cascade (opencv) and PRnet [26].

A cascade classifier (like HAAR and LBP) is a conventional technique used for various detection purposes that can be easily applied by the OpenCV Toolkit in a smartphone. However, in comparison with deep learning methods, it is not accurate enough. BlazeFace and Mobilenet V1/V2 are significantly faster and more accurate deep learning architectures for modern mobile devices.

Furthermore, PRnet provides a complete solution for facial detection and facial pose analysis, that increases the detection accuracy under pose variation and occlusion. We chose PRnet method as it had the best performance for our purposes. We significantly optimized the network and reduced its size by converting the model to the TensorFlow Lite format in order to embed it into a mobile application.

B. ERROR-CORRECTING CODES ALGORITHM

Aiming to stabilize the accuracy of the decoding, we employed an error-correction code algorithm. Although the choice of the better method for binary-correction is not within the scope of this work, we selected cyclic error-correcting codes namely the BCH algorithm [11] and Solomon algorithm [5] [27].

C. ENCODER

The first part of the generator is the encoder network. The aim of the encoder training process is to optimize the trade-off between its ability to restore the perceptual properties of the input images and the decoder performance to extract the hidden message.

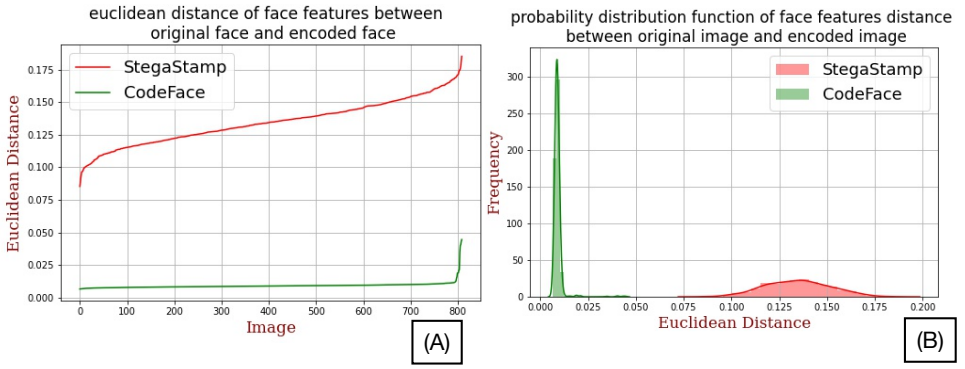
The encoder network architecture that we selected is based on UNets, however, the pooling layers were removed to preserve the information of the secret messages that may otherwise be lost during the network training. It thus receives an aligned face and a random binary message as inputs and produces an encoded image of the same size. The secret binary message is transformed (by reshaping and up-sampling) to coincide with the size of the encoder input as expected. The input face image is then processed by the encoder. Since the encoder does not have pooling layers, we need to design its architecture in a special manner by manually matching the parameters of convolutions to avoid layer connection errors.

Figure 3 displays CodeFace encoder network's details that are prepared for $400 \times 400 \times 3$ input images.

D. DECODER

The decoder network that is presented in Figure 4 is incorporated into the whole architecture after applying the noise to the images. The decoder is designed to recover a message that is encoded in a facial image. For this network, we applied CNNs with STN based on StegaStamp [24] and HiDDeN [32]. STN helps to crop out the appropriate region and normalize its scale, which can simplify the subsequent

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3132581, IEEE Access



	CodeFace	StageStamp
count	809.0000	809.0000
mean	0.0091	0.1346
std	0.0028	0.0168
min	0.0065	0.0853
25%	0.0081	0.1224
50%	0.0088	0.1347
75%	0.0094	0.1468
max	0.0445	0.1853

(c)

FIGURE 6. (A) the deviation in the Euclidean distance from facial features between the original and the encoded faces by StegaStamp. (B) the probability distribution function of the variation in the facial features between original and CodeFace encoded images. (C) summarizes basic statistics that are used to compare both models for a dataset of 809 images.

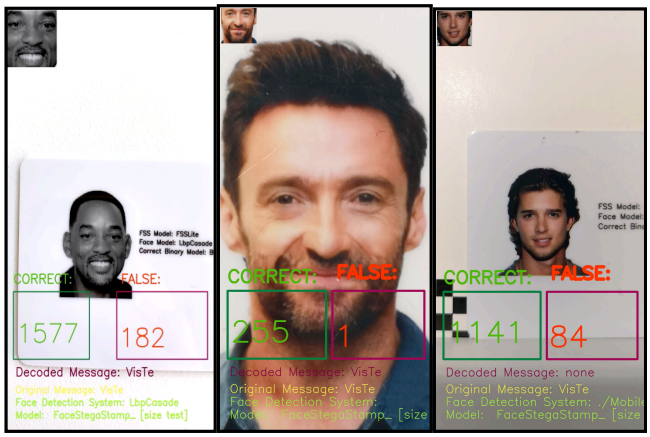


FIGURE 7. Samples of our model deployed on printed test ID cards. We used facial images of celebrities that were downloaded directly from the internet. The figure shows the full process of our model. The correct green box shows the number of frames in which the encoded and decoded messages were the same. The false red box shows the number of frames in which the decoder cannot correctly read and decode the message. The processing was made at 10 frames per second.

steganography decoding task and lead to better performance. It removes the spatial invariance from the encoded images by applying a learnable affine transformation that is followed by interpolation. The STN block is placed before the CNN.

E. PERTURBATION SIMULATION

To simulate the noise from printers and digital cameras, we applied several types of noise to the output images of the encoder network, before using the decoder. Based on two works, HiDDeN and StegaStamp, our model encompasses the same set of noise types such as perspective warp, motion and defocus blur, camera noise, color manipulation and JPEG compression. Perspective warp is a random homography that simulates the effect of a camera that is not precisely aligned with the encoded image marker. Motion and defocus blur can result from both the camera motion and inaccurate autofocus, which are very common on mobile devices. To simulate motion blur, a random angle is sampled to generate a straight-line blur kernel with a width between 3 and 7 pixels. The camera types of noise, which include photon, dark and shot noise from the camera system, has been well documented in previous works [24], [32]. Color manipulation, which is noise that can result from printers and monitors, has a limited color gamut compared to the full RGB color space and includes hue shift, saturation, brightness and contrast. The last added noise is a JPEG compression, which affects the image when it is stored in a lossy format, such as JPEG. All of these types of noise are applied to CodeFace between the encoder and the decoder in the training phase (refer to Figure 2). According to the objective of the CodeFace system, it should be able to read a message from a small face image printed on a ID or on an MRTD. We then developed and added to the training network the new noise simulation layers. All of the layers have a scalar hyper parameter that governs the distortion intensity.

The novel idea proposed in this research is to attach a resize network to our model as an additional noise simulation module. This is designed to help the decoder read messages from smaller photos in comparison with previous approaches. The resize network decreases the size of the encoded images that the decoder receives.

F. LOSS FUNCTIONS

All the outputs of the CodeFace generator are received by the CodeFace discriminator. The discriminator is designed with a set of loss functions to improve the model's performance. The most important loss functions in our model are LPIPS and face embedding.

LPIPS demonstrated its efficiency in StegaStamp. Therefore, we use it as perceptual loss function in our model. In addition to LPIPS, there is need to preserve the facial structure and its high-level representation, thus it was modified the loss function with the similarity function that is estimated by the output of a FaceNet model. FaceNet model uses the Inception Resnet V1 architecture that was trained with the VGG2 dataset. The model receives a 160×160 pixels

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3132581, IEEE Access

RGB facial image and delivers a 128 dimensional vector of facial features. The Euclidean distance between the two sets of features expresses the degree of similarity between their source images. Our model thus computes the Euclidean distance to minimize the difference in facial features between the original and encoded facial images during the training process.

The two loss function terms are then summed with other loss terms that were used in the aforementioned models, [24], [32]. In summary, the complete loss function in the CodeFace model includes the following components:

- LPIPS perceptual loss function L_P,
- FaceNet loss function L_F - an arbitrary biometric recognition system which measures the closeness of two face templates by Euclidean distance,
- The Wasserstein loss L_W - utilised as a perceptual loss for the encoder/decoder pipeline [3],
- Residual regularization L_R [24],
- The cross entropy message L_B - that trains the decoder network in order to recover the message.

In the training process, the loss functions are the weighted sum of the five image loss terms,

$$Loss = FL_F + PL_P + WL_W + RL_R + BL_B \quad (1)$$

where F, P, W, R, and B are the weights for each loss function components. At the early stage of the training F, P, W, and R are initially set to zero (B is set to 0.01) until the decoder achieves high accuracy in these weights (this usually occurred after 500 to 700 steps in our experiments). Afterwards, these weights are increased linearly in every step. We slowly improve the effectiveness of the loss function by increasing these coefficients.

G. DATASETS

For the training of the CodeFace models, seven databases of frontal facial images were incorporated, including the PICS face dataset[1], the Color FERET face dataset[2], the AT&T Database of Faces[3], the BioId face dataset[4], the Georgia Tech Face Database[5] and the FEI Face Database[6].

For the purpose of CodeFace, we needed a dataset to meet the requirements of international institutions such as ICAO (International Civil Aviation Organization) concerning identification documents [1], [10]. The first requirement concerns the size of the photos, which must be at least $35mm \times 45mm$ (width $\times$ height) and the size of the image of the face that must be at least $16mm \times 20mm$ (excluding ears). The second requirement concerns the framing of the image: the image should depict a complete frontal image of the face, showing the full head and centered in the eyes. The third requirement regards the background: the photo should have an uniform white background with sufficient contrast with the face. The fourth requirement states that the eyes should be fully visible. If the person uses glasses, the lenses must be fully transparent (without distortion due to reflections and shadows).

To meet these requirements, we implemented a verification system that removes facial images that did not fulfil all the requirements. After the mentioned datasets were filtered, 1900 images remained, respecting the ICAO standards. Although this is not a large dataset, we are able to obtain very good results with it. All the facial images that are presented in this paper belong to celebrities.

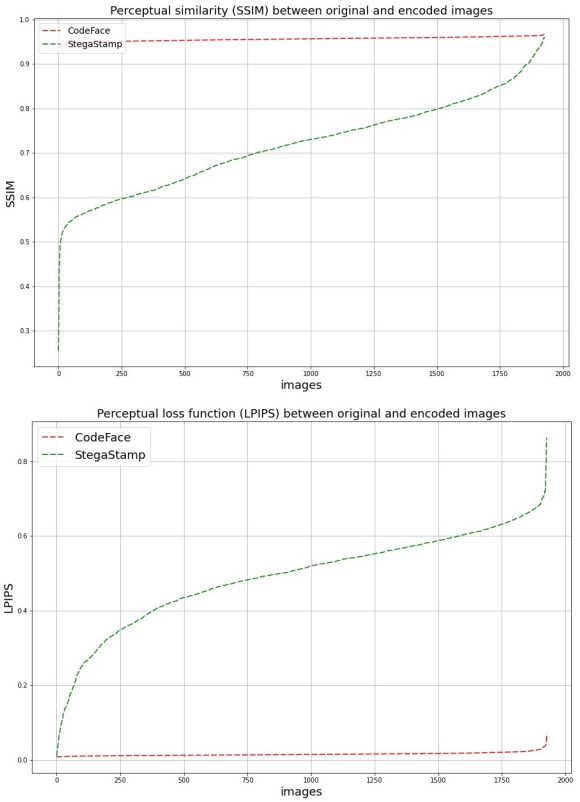


FIGURE 8. The first plot shows the LPIPS loss function between original and the CodeFace encoded images (red line), and between the original and StegaStamp encoded images (green line). Second plot shows the Structural Similarity Index (SSIM) between the original and the encoded images for both CodeFace (red) and StegaStamp (green) images.

IV. EXPERIMENTS

We have monitored and verified the performance of our encoder and decoder networks using three smartphone cameras with medium to high capacities (HUAWEI P40Pro, HUAWEI P20Pro and iPhone 10s). All the decoding tests have been done using a set of images (grayscale and RGB) physically printed on polycarbonate, PVC cards and regular paper. In terms of printer, we used two commercial printers (Brother HL-3270CDW and HP Color LaserJet CP5225n), one laser engraving printer for polycarbonate cards from the manu-

[1] http://pics.psych.stir.ac.uk/
[2] https://www.nist.gov/itl/ products-and-services/color-feret-database/
[3] https://git-disl.github.io/GTDLBench/datasets/att_face_dataset/
[4] https://www.bioid.com/facedb/
[5] http://www.anefian.com/ research/face_reco.htm
[6] https://fei.edu.br/ cet/facedatabase.html

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3132581, IEEE Access

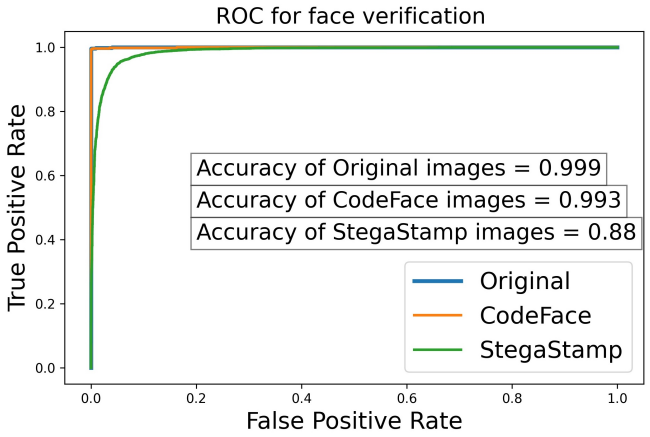


FIGURE 9. Comparison of the face verification performance. First, the blue plot is the face verification applied to the original image without any message. The second plot, in orange, shows the ROC of the face verification when an image is generated by CodeFace. The last plot, in green, is the ROC for StegaStamp generated images.

facturer Muehlbauer, and a Thermal Transfer printer from DataCard.

Since the results of the experiments showed no significant differences regarding the type of printer used or the material (paper, polycarbonate or PVC), we do not present separate results for each printer and material.

We will show that CodeFace is better at improving the image perception quality than similar state-of-the-art approaches such as StegaStamp [24]. We compare our result with the StegaStamp model because it is the only printer-proof steganography model to the best of authors' knowledge.

In the experiments, we recorded the videos from the ID cards and analyzed them. Figure 7 shows samples of these videos and the number of frames in which the encoded and decoded messages were the same. A frame is considered correct if the decoder recovers the expected message and passes the validation step. It is worth noticing that the method fails when any of the network modules is not able to achieve successful results. For instance, the method fails if a face detection is not achieved or if the correct crop of the region of interest is not correctly performed. The method also fails if the decoder is unable to produce a correct bit stream for the error-correcting code method.

Instead of running the method for a single frame or image, CodeFace analyzes frames from the video feed to improve the user experience. The user then has a more fluid and friendly interface with the application, while increasing the probability of achieving a successful decoding since more frames are analyzed. CodeFace code was developed using TensorFlow library and it is trained in Python. After training, the model was converted and optimized for the TFLite format, running in Java (Android studio).

A. PERFORMANCE

Facial images encoded with our CodeFace approach outperform the StegaStamp generated images in terms of their perception quality. The visually perceptible results from both CodeFace and StegaStamp for 100 bit messages are presented in Figure 5. The results clearly show, qualitatively, that our model better preserves facial structure and texture. Quantitatively, Figure 6 presents the Euclidean distances of facial features and the probability distribution function of the difference in the facial features between the original and encoded images generated by CodeFace and StegaStamp. CodeFace reduced the impact of noise on the system by at least 90 percent compared to StegaStamp. Therefore, we expect that when using CodeFace for encoding, the face detection and face verification systems have a negligible error rate caused by the encoding of the hidden message. Additionally, we have used 100 frontal face images from the VGGFace2 dataset to compare perceptual similarity using LPIPS and the Structural Similarity Index (SSIM) between the original and encoded images from both CodeFace and StegaStamp, as depicted in Figure 8. As shown in the two plots, CodeFace images are much more similar to the original images when compared with StegaStamp encoded images. LPIPS is adopted for the training of both CodeFace and StegaStamp.

To compare the influence of StegaStamp and CodeFace on the performance of face verification, we selected a dataset of 32000 pairs of facial images from the VGG2 database (16000 with the identity matched and 16000 non-matched). In each image pair, we chose one image to generate an encoded copy using StegaStamp and CodeFace. Next, we evaluated the facial verification performance across three setups, namely original, StegaStamp and CodeFace images. The similarity score is estimated by the standard DLib face verification module [17]. We used an accuracy metric to fix the similarity threshold at 0.6 (as used in [22]). With these settings, we obtained a 99.9% percent accuracy for face verification using only original images, 99.3% accuracy for the CodeFace encoded images and 88.8% accuracy for the StegaStamp images whose ROC curves are shown in Figure 9.

B. ABLATION STUDY

Figure 10 presents the encoded CodeFace images with messages of different sizes, without the resize network, from 80 to 200 bits. As one can see, after 120 bits the perceptual quality of encoded images is slightly degraded.

In order to test the performance of the resize network (regarding the noise simulation aspect), the experiments were performed based on the videos captured by the smartphone camera in a variety of real-world environments. In the experiments, we first converted the images to the same resolution by a super resolution network [31]. Then we encoded a special message in 100 facial images of celebrities, and finally we removed the background from all of the encoded images by applying a Portrait Segmentation model [6]. By using a

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3132581, IEEE Access

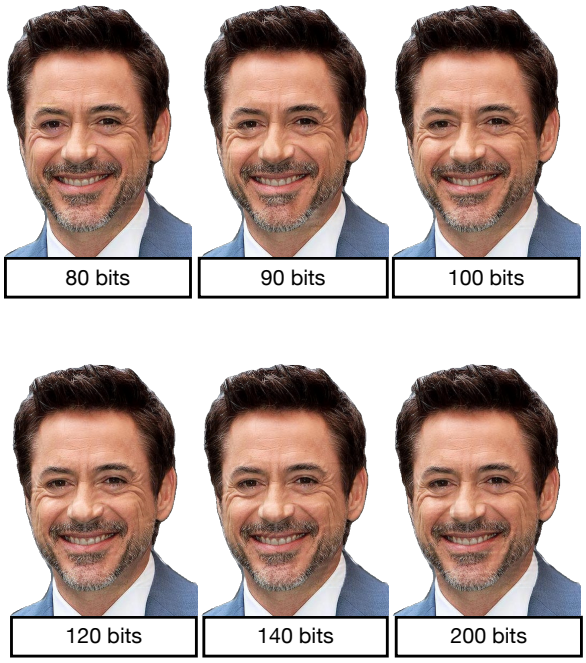


FIGURE 10. Encoded CodeFace images are shown with different sizes of the message, from 80 bits up to 200 bits.

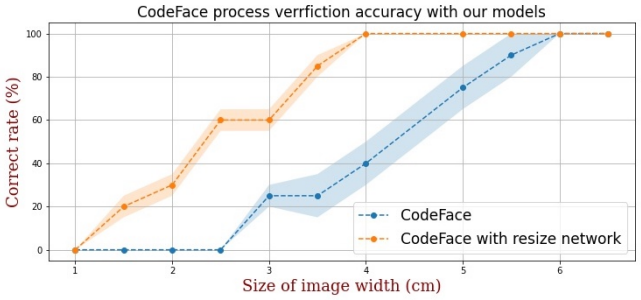


FIGURE 11. The plot shows the decoders' performance tests carried out with a handheld HUAWEI P20Pro smartphone camera. Videos were captured in a variety of real-world environments (in-the-wild). Encoded images, which are of celebrities, are printed by a consumer printer. We repeated the test five times to achieve higher precision. As can be seen, CodeFace with the resize layers has better performance in small images with widths above 4 cm (the actual size of a typical ID card portrait).

commercial office printer, the encoded images are printed with sizes ranging from 1 cm to 6.5 cm wide on paper. As the size and shape of people's faces are not uniform, we do not have a fixed range of dimensions for all images. For each test, we preserved the width of the images, while the height can be scaled accordingly to the original aspect ratio. In the end, we test the 100 facial images with a HUAWEI P20Pro smartphone camera. An image was validated and accepted if the decoder recovers the complete message at least once in the first 10 frames, and at least twice in the first 20 frames. The result of the experiments, which are presented in Figure 11, demonstrate that the CodeFace system achieved an accuracy of 100%, for images whose width is larger than 6 cm, while the CodeFace system, with the resize network model, has an accuracy of 100% for images whose width is larger than 4 cm. It means that above this size, we expect the CodeFace model to work with no error irrespectively of the use of the resize network. We emphasize that the results are dependent both on the smartphone camera and the printer.

In summary, the use of the resize network can improve the model accuracy for smaller images at a small cost on the perceptual structure of encoded faces.

In terms of computational performance, for 100 random images of the dataset, the mean execution time to encode a message in a single image is approximately 1.2 seconds, running in a regular PC, while the mean execution time of the decoder to read and validate a message is approximately 0.8 milliseconds.

CONCLUSION

In this paper, we introduce a novel deep learning printer-proof steganography approach for document security systems. We significantly optimized the performance of the CodeFace encoder and decoder to incorporate and read messages in facial images by combining the steganography with face detection, considering a new loss function and a noise simulation pipeline. The new loss function combines perceptual and high-level facial representation parts to measure the variation of the facial structure during training. The noise simulation pipeline is modified with a resize layer that decreases the size of the encoded images used throughout the training. Therefore, CodeFace with the resize layer can better read a message from a smaller image and the storage size of the decoder network was decreased. This is achieved at a small cost to the perceptual structure of the encoded face. In comparison with state-of-the-art approaches, we have demonstrated significant improvements in terms of perceptual quality of the encoded images and their compliance with modern facial recognition systems and document issuing requisites. CodeFace presents an innovation that can be easily implemented in real world document validation systems and applied directly to ID cards and MRTDs as a security protocol.

As future paths for research, we intent to increase the amount of information encoded in the face images. To achieve this goal, we will study new coding and compression algorithms as local sensitive hashing [2]. We also intend to study alternative possibilities to use the whole face image, instead of a smaller part of the portrait.

REFERENCES

[1] International Civil Aviation Organization (ICAO) (2015). Machine readable travel documents. part 11: Security mechanisms for mrtds. technical report doc 9303. seventh edition. Security mechanisms for MRTDs. Technical report Doc 9303. Seventh Edition, International Civil Aviation Organization (ICAO) (2015).

[2] Ravi Kumar Anirban Dasgupta and Tamás Sarlós. Fast locality-sensitive hashing. In Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 1073–1081. IEEE, 2011.

[3] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In International conference on machine learning, pages 214–223. PMLR, 2017.

[4] Valentin Bazarevsky, Yury Kartynnik, Andrey Vakunov, Karthik Raveendran, and Matthias Grundmann. Blazeface: Sub-millisecond neural face detection on mobile gpus. arXiv preprint arXiv:1907.05047, 2019.
[5] Daniel Bleichenbacher, Aggelos Kiayias, and Moti Yung. Decoding of interleaved reed solomon codes over noisy data. In International Colloquium on Automata, Languages, and Programming, pages 97–108. Springer, 2003.
[6] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. IEEE transactions on pattern analysis and machine intelligence, 40(4):834–848, 2017.
[7] J. Deng, J. Guo, N. Xue, and S. Zafeiriou. ArcFace: Additive Angular Margin Loss for Deep Face Recognition. In 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 4685–4694, 2019.
[8] Yao Feng, Fan Wu, Xiaohu Shao, Yanfeng Wang, and Xi Zhou. Joint 3d face reconstruction and dense alignment with position map regression network. Proceedings of the European Conference on Computer Vision (ECCV), pages 534–551, 2018.
[9] Anselmo Ferreira, Ehsan Nowroozi, and Mauro Barni. Vipprint: Validating synthetic image detection and source linking methods on a large scale dataset of printed documents. Journal of Imaging, 7(3):50, 2021.
[10] International Organization for Standardization. ISO/IEC 19794-5:2005. Information technology — Biometric data interchange formats — Part 5: Face image data. ISO/IEC JTC 1/SC 37 Biometrics, 06 2005.
[11] George Forney. On decoding bch codes. IEEE Transactions on information theory, 11(4):549–557, 1965.
[12] Ian J. Goodfellow, Jean Pouget-Abadie, M. Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In NIPS, 2014.
[13] Vojtěch Holub and Jessica Fridrich. Designing steganographic distortion using directional filters. In 2012 IEEE International workshop on information forensics and security (WIFS), pages 234–239. IEEE, 2012.
[14] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861, 2017.
[15] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In Advances in neural information processing systems, pages 2017–2025, 2015.
[16] Robert L Jones, Yecheng Wu, Daoshen Bi, and Robert Andrew Eckel. Line segment code for embedding information, July 4 2019. US Patent App. 16/236,969.
[17] Davis E. King. Dlib-Ml: A Machine Learning Toolkit. J. Mach. Learn. Res., 10:1755–1758, Dec. 2009.
[18] Ferenc Koltai and Bence Adam. Enhanced optical security by using information carrier digital screening. In Optical Security and Counterfeit Deterrence Techniques V, volume 5310, pages 160–169. International Society for Optics and Photonics, 2004.
[19] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In European conference on computer vision, pages 21–37. Springer, 2016.
[20] Iurii Medvedev, Nuno Gonçalves, and Leandro Cruz. Biometric system for mobile validation of id and travel documents. In 2020 International Conference of the Biometrics Special Interest Group (BIOSIG), pages 1–5. IEEE, 2020.
[21] Tomáš Pevnỳ, Tomáš Filler, and Patrick Bas. Using high-dimensional image models to perform highly undetectable steganography. In International Workshop on Information Hiding, pages 161–177. Springer, 2010.
[22] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 815–823, 2015.
[23] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. DeepFace: Closing the Gap to Human-Level Performance in Face Verification. In 2014 IEEE Conference CVPR, pages 1701–1708, 2014.
[24] Matthew Tancik, Ben Mildenhall, and Ren Ng. Stegastamp: Invisible hyperlinks in physical photographs. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 2117–2126, 2020.
[25] H. Wang, Y. Wang, Z. Zhou, X. Ji, D. Gong, J. Zhou, Z. Li, and W. Liu. CosFace: Large Margin Cosine Loss for Deep Face Recognition. In 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 5265–5274, 2018.
[26] Yue Wang and Justin M Solomon. Prnet: Self-supervised learning for partial-to-partial registration. arXiv preprint arXiv:1910.12240, 2019.
[27] Stephen B Wicker and Vijay K Bhargava. An introduction to reed-solomon codes. Reed-Solomon codes and their applications, pages 1–16, 1994.
[28] Changqian Yu, Jingbo Wang, Chao Peng, Changxin Gao, Gang Yu, and Nong Sang. Bisenet: Bilateral segmentation network for real-time semantic segmentation. In Proceedings of the European conference on computer vision (ECCV), pages 325–341, 2018.
[29] Kevin Alex Zhang, Alfredo Cuesta-Infante, Lei Xu, and Kalyan Veeramachaneni. Steganogan: High capacity image steganography with gans. arXiv preprint arXiv:1901.03892, 2019.
[30] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 586–595, 2018.
[31] Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu. Residual dense network for image super-resolution. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 2472–2481, 2018.
[32] Jiren Zhu, Russell Kaplan, Justin Johnson, and Li Fei-Fei. Hidden: Hiding data with deep networks. In Proceedings of the European conference on computer vision (ECCV), pages 657–672, 2018.

FARHAD SHADMAND received B.Sc. and M.Sc. degrees from Isfahan University of Technology. He is a researcher and Ph.D. student in Visual Information Security Team of the Institute of Systems and Robotics - Coimbra, Portugal. His research interests involve deep learning and computer programming methods in computer vision.

IURII MEDVEDEV received his B.Sc. and M.Sc. degrees in Peter the Great Saint Petersburg Polytechnic University, Saint-Petersburg, Russia. Currently he is a Ph.D. student in University of Coimbra and a researcher in Visual Information Security Team of the Institute of Systems and Robotics - Coimbra, Portugal. His research is focused on document security methods for face recognition with deep learning and computer vision.

NUNO GONÇALVES received the Ph.D. degree in computer vision from the University of Coimbra, Portugal, in 2008. Since 2008, he has been a tenured Assistant Professor with the Department of Electrical and Computers Engineering, Faculty of Sciences and Technologies, University of Coimbra. He is currently a Senior Researcher with the Institute of Systems and Robotics, University of Coimbra. He has been recently coordinating several projects centered on the technology transfer to the industry. In 2018, he joined the Portuguese Mint and Official Printing Office (INCM), where he coordinates innovation projects in areas, such as steganography, facial recognition, biometrics, graphical security, ID and Machine-Readable Travel Documents, information systems, and robotics. He has been working in the design and introduction of new products as result of the innovation projects. He is the author of several articles and communications in high-impact journals and international conferences. His scientific career has been mainly developed in the fields of computer vision, visual information security, biometrics and robotics, but also in computer graphics. He is author of 5 patents.