# Compressed Domain Face Verification Assessment

Francisco Ferreira João Ascenso Catarina Brites

***Abstract*—In recent years, more devices and platforms that deal with visual data emerged, with a wide range of applications. Also, advances in technology led to more visual data necessary for the representation of an image. Image coding become a common and necessary step, allowing to overcome some storage limitations and stringent delay requirements. Moreover, learning-based image coding solutions have emerged with competitive performance against state-of-art conventional methods. Traditionally, computer vision tasks are performed over lossy reconstructions of the image, which can lead to lower performance and high complexity models. However, these tasks can be performed on the latent representation of the image, generated after entropy decoding the bitstream, by using the features extracted from the image while coding, as in [1]. Recently, some compressed domain image analysis solutions have been successfully developed, being able to compete against state-of-art models. In this paper, it is proposed a compressed domain face verification model with lower computational complexity than traditional deep-learning solutions. The proposed model architecture (cResNet39) is obtained by removing the first layers of a ResNet50 to accept latent representations as input. The compressed domain model was not able to surpass the performance of state-of-art solutions with learning-based codecs; however, it achieved similar (although lower) performance to the selected anchors solutions while reaching lower computational complexity.**

***Keywords—Image coding, compression efficiency, deep learning, compressed domain processing, neural networks, face verification***

## I. INTRODUCTION

Due to recent advances in technology, image applications are used by a very large percentage of the world's population [1]. Since image resolution and quality are increasing, images are being represented with larger amounts of data, which increases the storage cost and transmission delay. Consequently, image coding become a common and necessary step in image applications, due to the need of efficiently compress the image information. Image coding solutions focus on representing the image as much compact as possible by exploiting spatial and statistical redundancies. These solutions can be lossless (decoded information is mathematically the same as the original) or lossy (some information is lost in the coding process). However, lossy compression is more popular especially since many applications do not require perfect mathematical reconstruction. Lossy coding tries to achieve a good trade-off between compression rate and perceptual quality, often considering the human visual system.

There have been some advances throughout the last thirty years in lossy conventional image coding solutions, such as JPEG [2], JPEG2000 [3][4], HEVC Intra [5][6] and VCC Intra [7]. These standards employ lossless techniques to exploit the spatial and statistical redundancies and lossy techniques that exploit the visual redundancies (often with quantization) of the image. These conventional codecs are composed of hand-crafted methods to efficiently compress the image [2].

In recent years, deep-learning based methods have became very popular in computer vision and image processing tasks, due to their ability in efficiently represent visual information. These solutions leverage from complex neural networks architectures that usually are mainly composed of convolutional layers. Because of these models capability of efficiently represent the image visual information in a lower dimension space, learning-based image coding solutions have emerged with competitive performance against state-of-art conventional methods. These learning-based models mainly use autoencoders to perform the dimensionality reduction of the image, such as Ballè's variational image compression with a scale hyperprior codec [8] (Balle2018 codec) and Cheng's learning-based image coding solution with discretized Gaussian Mixture Likelihoods and attention models [9] (Cheng2020).

Traditionally, computer vision tasks are performed over lossy reconstructions of the images, which can lead to lower performance and high complexity models. However, these tasks can be performed on the latent representation of the image, generated after entropy decoding the bitstream, by using the features extracted from the image while coding, as in [1]. Recently, some compressed domain image analysis solutions, that use Residual Neural Networks (ResNets), have been successfully developed, being able to compete against state-of-art models.

In this paper, it is proposed a compressed domain face verification model, which takes latent representations of the faces as input. Face verification is the binary classification problem (True/False, Yes/No, 0/1, etc.) of verifying if the person's identity is correct by using facial traits, i.e., if the person really is who he/she claims to be. The user's face (probe face) is compared to previously collected reference faces of the alleged identity to make the classification decision. It is used an adaptation of the ResNet50 architecture to generate the face embeddings which are used to compare the faces by means of a computed similarity score. The proposed architecture is obtained by removing the first layers of the ResNet50, based in [10], and by adjusting the network to receive latent representations as input. Hence, it is obtained a lower complexity model (cResNet39, c corresponds to compressed representation) which was expected to achieve higher/similar performance than the RGB learning-based face verification methods that use lossy decoded images as input. The proposed model will be evaluated for the Labeled Faces in the Wild (LFW) dataset [11] against implemented anchor solutions based in deep-learning face verification models that use the ResNet50 architecture. Two different anchors are used: 1) original anchors (uses original images as input); 2) decoded anchors (use decoded probe images and original reference images).

## II. RELATED WORK

In the literature, there are many learning-based solutions that perform face verification tasks over decoded or original images, such as the VGGFace2 model [12] and the ArcFace model [13]. These two solutions are based on ResNet

architectures with many layers, like the ResNet50, and are able to compete against state-of-art models.

In the context of computer vision performed in the compressed domain, some solutions have emerged over the recent years, which are able to efficiently perform the defined computer vision task by using compressed representations of the images. In 2018, it was presented in [10] two compressed domain image analysis solutions: 1) performs image classification; 2) performs image segmentation. In [10], it is used adjusted ResNet50 architectures, in which the first layers with higher spatial dimensions than 28x28 are removed. Different models are then obtained according to the number of used layers (complexity). These solutions are able to achieve similar performance to reconstructed domain solutions, while having lower/similar computational complexity and storage cost.

In this paper, it is proposed a solution that integrates the training procedure of the VGGFace2 model [12] and the finetuning method of the ArcFace model [13], by using an adaptation of the cResNet39 architecture in [10]. The proposed model face verification performance is then evaluated against anchors solutions, which are based in [12] and in [13].

## III. Compressed Domain Face Verification

The proposed compressed domain face verification model is mainly composed by a face detection model, a learning-based image codec, latent representation processing, face recognition model (compressed domain) and a similarity score and decision making process, as shown in Fig. 1.

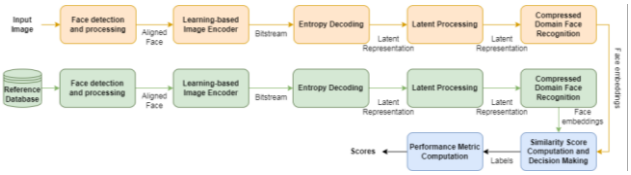


Fig. 1. Proposed compressed domain face verification framework.

### A. Face Detection

The input image may contain more than one face and/or other information (e.g., background) that is not related to the labelled face being verified, which can negatively impact the face verification performance. It is used a Pytorch implementation of RetinaFace model [14][15], which is a deep-learning base face detection model that performs pixel-wise localisation of faces in multiple scales. The RetinaFace is used to obtain the faces bounding boxes and landmarks (nose, eyes and mouth coordinates) for candidate faces. A candidate face is only considered a detected face if the confidence score is higher than a predefined threshold. However, in some images multiple faces may be detected, as occurs in some samples of the LFW test dataset. Since only one face in the image is labelled, there is the need to exclude all other candidate faces. From an analysis of the LFW dataset, it was observed that the labelled face is positioned mainly in the center of the image when comparing with the other faces. To overcome the problems of zero detected faces and of multiple detected faces it is used the heuristics shown in Fig. 2. It is used an initial confidence threshold of 0.985 which is decreased by 0.005 to a minimum value of 0.7 in case that zero faces were detected or none of the detected faces contains the center of the image. If multiple detected faces have the center inside its boundaries, then it is chosen the most centered face by computing the minimum distance from the center of the image to the face limits. The face with highest minimum distance is considered the most centered face, being selected as the labelled face.

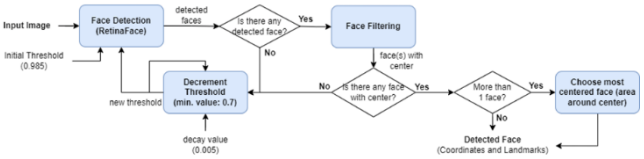


Fig. 2. Face Detection Pipeline

### B. Image Processing

After obtaining the facial boundaries coordinates and landmarks, the face is cropped and aligned with the implementation in [15], which corresponds to a 2D affine transformation by using the eyes and nose landmarks. Afterwards, the face is uniformly resized until the highest spatial dimension reaches a size of 224, i.e., if the face's spatial dimension is (180,120), then the resize value is 224-180=44, which after the uniform resize will lead to an image with spatial dimensions equal to (224, 164). Padding is then applied to the lowest dimension to obtain an image of size 224x224. Additional padding is performed in all dimensions to obtain a 256x256 image. At last, the image is normalized to values between 0 and 1 before entering the learning-based encoder.

### C. Learning-based image codec

The images' compressed representations are obtained by using a learning-based image codec. The Ballè's variational image compression with a scale hyperprior codec [8] (Balle_2018 codec) was selected. The bitstreams are obtained after encoding the facial images, which are then entropy decoded to generate the latent representations. It was chosen the CompressAI [16][17] Balle_2018 codec implementation solutions pretrained for the Mean Squared Error (MSE) loss for relative target qualities 1, 3, 6 and 8. Each compression quality corresponds to different bitrates, e.g., for the LFW test dataset for the qualities 1, 3, 6 and 8 it is achieved the bitrate values of 0.10 bpp, 0.17 bpp, 0.38 bpp and 0.65 bpp respectively. The obtained latent representations present a size Nx16x16, being N the number of channels. The value of N varies with the used target quality, being as follows: for qualities 1 and 3, N=192; for qualities 6 and 8, N=320.

### D. Latent Processing

The implemented compressed domain neural network model, which is adapted from the ResNet50 architecture, accepts latent representation of dimension Nx14x14 as input. However, the latent representations have a spatial size of 16x16 after entropy decoding, i.e., a spatial reduction of x16 (256/16=16) is performed over the image. Additionally, before the encoder, the image is padded from a size of 224x224 to a size of 256x256 (padding of 16 pixels in all spatial dimensions), which is considered non-desirable information for face verification tasks. To solve these problems, the latent representation is center cropped to a size of 14x14. Since the padding in the latent is represented by 1 pixel in all spatial dimensions (16/16=1), then only the padding area is being cropped, keeping the face's information.

### E. Compressed Domain Architecture

It is proposed a variant of the ResNet50, based in the implementation in [18], the cResNet39 (c corresponds to

compressed representation), to perform face recognition tasks over images' latent representations. As in [10], the cResNet39 architecture is obtained by removing the convolutional layers with spatial dimension higher to 28x28 of the ResNet50 (root block and first residual block). These modifications remove a total of 11 layers (10 convolutional layers + 1 max pool layer) from the ResNet50, as shown in TABLE I. The latent representations used as input have a spatial size of 14x14, and since it is desirable to obtain the same output size after the last convolutional layer as in the ResNet50, the residual blocks 2 and 3 are adjusted to not perform downsampling. Hence, only the residual block 4 executes spatial reduction of the input in the cResNet39, as shown in Fig. 3.

The output layer of the cResNet39 varies from the task to which the model is being trained, however for face verification it is used an extra Fully Connected (FC) layer with a 2048-d face embedding output. More detailed information about the cResNet39 architecture is presented in TABLE I. and Fig. 3.

TABLE I. STRUCTURE OF THE RESNET50 AND CRESNET39 ARCHITECTURES. EACH RESIDUAL BLOCK IS COMPOSED OF RESIDUAL UNITS THAT CONTAIN 3 CONVOLUTIONAL LAYERS EACH, E.G., RESIDUAL BLOCK 1 HAS 3 RESIDUAL UNITS. N IS THE NUMBER OF CHANNELS OF THE LATENT REPRESENTATION.

| | | Input Size | Block | Root | Residual block 1 | Residual block 2 | Residual block 3 | Residual block 4 | |
|---|---|---|---|---|---|---|---|---|---|
| RGB | ResNet50 | (3,224,224) | Output Size (spatial) | 56x56 | 56x56 | 28x28 | 14x14 | 7x7 | avg. pool, 2048-dim fc [output: 1x2048] |
| | | | Arch. | [7x7, 64] 3x3 max pool, s=2 | [1x1, 64; 3x3, 64; 1x1, 256] x3 | [1x1, 128; 3x3, 128; 1x1, 512] x4 | [1x1, 256; 3x3, 256; 1x1, 1024] x6 | [1x1, 512; 3x3, 512; 1x1, 2048] x3 | |
| Compressed Domain | cResNet39 | (N,14,14) | Output Size (spatial) | None | None | 14x14 | 14x14 | 7x7 | |
| | | | Arch. | | | [1x1, 128; 3x3, 128; 1x1, 512] x4 | [1x1, 256; 3x3, 256; 1x1, 1024] x6 | [1x1, 512; 3x3, 512; 1x1, 2048] x3 | |

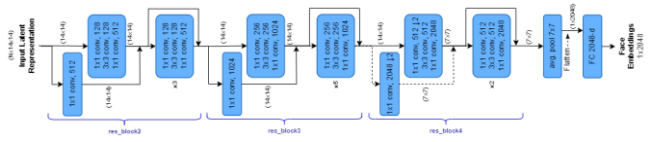

Fig. 3. Framework illustration of the cResNet39 architecture for face verification. N corresponds to the number of channels of the input latent representation, this value differs between codec solutions. Dashed line represents the downsampling path of the residual block input which is added to the residual block's first residual unit output.

### F. cResNet39 Training

During the training procedure, the cResNet39 model has 2 phases: 1) training with a model which has a FC layer with 8631-d output (cResNet39_8631-d) as the output layer, for face identification, based in [12]; 2) the 8631-d FC layer is replaced by a pretrained 2048-d FC layer, to generate the face embeddings, which is then fine-tuned for face verification.

The cResNet39 is trained with the VGGFace2 dataset [22] (~3.1M images, 8631 identities). All images in the dataset are used in training, to replicate the training process in [12][18]. While for fine-tuning the model the dataset is partitioned into training set (90%, 7768 identities, ~2.8M images) and validation set (10%, 863 identities, ~310k images). However, the VGGFace2 dataset described in [12] has a validation set, used to evaluate the model's performance during training. But this extra validation set was not found available online.

The training images are pre-processed before entering the network as performed in [12][18]. The VGGFace2 dataset images have an original spatial size of 112x112, being resized with bilinear interpolation to a size of 256x256. No face detection and alignment are performed to the images, since the samples already present the labelled face centered and with a small percentage of background information. The resized image is then randomly centered cropped to a spatial size of 224x224. Afterwards, a grayscale conversion is applied for some cropped images, with 20% probability. Before the encoder, it is applied a pad in the image to a spatial size of 256x256 and a normalization to values between 0 and 1. Latent processing is then applied to the compressed representations, after entropy decoding.

The cResNet39 training for face identification is performed on the cResNet39_8631-d architecture with the Cross-Entropy Loss (CEL) [19] function with mean reduction. The model is trained with the mini-batch SGD optimizer, with mini-batches of 64 images and a learning-rate of 0.1 for 6 epochs. After training the cResNet39, the 8631-d output FC layer is replaced by the anchors' trained ResNet50's FC layer, which is presented in Section IV that generates 2048-d face embeddings. The cResNet39 model is fine-tuned with the ArcFace Loss [20][21] with the mini-batch SGD optimizer, with mini-batches of 64 images and a learning-rate of 0.005 for 3 epochs. The ArcFace Loss uses an extra FC layer that takes the face embeddings as input and the output dimension corresponds to the number of train identity classes (7768). Only the 2048-d FC layer and the extra ArcFace FC layer are updated, the rest of the cResNet39 and encoder weights are kept constant. The training procedure is performed for latent representations generated by the selected Balle_2018 codec for each of the target qualities, obtaining 4 different models operating at different bitrates.

### G. Similarity Computation and Decision Making

The face embeddings computed by the cResNet39 model represent features that were extracted from the facial image. In face verification, it is desirable to compare the probe face embedding with reference identity face embeddings. These comparation is performed by the Cosine Similarity Score (COS), which computes a numerical comparison (-1 to 1) between two face embeddings $(\vec{a}, \vec{b})$ as in (1).

$$\boldsymbol{COS}(\vec{a},\vec{b}) = \frac{\vec{a}\cdot\vec{b}}{||\vec{a}||\cdot||\vec{b}||} = \frac{\Sigma_1^n a_i b_i}{\sqrt{\Sigma_1^n {a_i}^2}\sqrt{\Sigma_1^n {b_i}^2}} \quad (1)$$

The decision making is performed with a threshold value that will classify a pair (probe face and identity reference face) as True (same identity) or False (different identities). If the similarity score is higher than the defined threshold the pair is labelled as True otherwise is labelled as False.

## IV. EXPERIMENTAL RESULTS

### A. Benchmarks

The proposed compressed domain solution performance is evaluated against anchors solutions, which are divided into 2 types: original anchor and decoded anchors. These models have a similar face verification pipeline as the compressed domain model in Fig. 1. The anchors solutions use the ResNet50 architecture in TABLE I. instead of the cResNet39, to generate the face embeddings.

However, in the original anchor the face is cropped, aligned and resized, as explained in Section III without the extra padding, keeping a size of 224x224, from the original images. Afterwards, the faces enter directly in the ResNet50 model to generate the face embeddings, no compression is

performed. Whereas for the decoded anchors, the face is cropped, aligned and resized, as in Section III, before entering the encoder. The face is then encoded and decoded obtaining a lossy reconstructed face of 256x256. Since the encoder input face was padded from 224x224 to 256x256, then this padded area is cropped before entering the ResNet50 to compute the face embeddings. However, in the decoded anchors only the probe face (face whose identity is being verified) is encoded and decoded, for the identity reference face is used original images. In all anchors solutions, it is subtracted a RGB vector with the values [131.0912, 103.8827, 91.4953] to the faces before entering the ResNet50, as in [18]. Each value of the vector will be uniformly subtracted to all values of one channel of the RGB image (Red: 131.0912, Green: 103.8827, Blue: 91.4953).

For the decoded anchors is selected the same learning-based codec as in the proposed compressed domain solution (Balle_2018 optimized for MSE loss for target qualities 1, 3, 6 and 8) [8][16]. Additionally, some popular conventional codecs are chosen, being selected the JPEG [2][23], JPEG2000 [3][4][24] and HEVC-Intra [5][6][25] codecs.

The cResNet39 training procedure was replicated from the ResNet50 model, since the anchors solutions were trained first. Hence, the ResNet50 training is divided into 2 phases: 1) training with a model which has a FC layer with 8631-d output (ResNet50_8631-d) as the output layer, for face identification, based in [12]; 2) the 8631-d FC layer is replaced by a randomly initialized 2048-d FC layer, to generate the face embeddings, which is then fine-tuned for face verification. The ResNet50 is trained only with original images, so no compression is performed during the training procedure.

The ResNet50 is trained with the VGGFace2 dataset (~3.1M images, 8631 identities) [12][22]. As in the cResNet39, all images in the dataset are used in training, to replicate the training process in [12][18]. While for fine-tuning the model the dataset is partitioned into training set (90%) and validation set (10%). The training images are preprocessed as in the cResNet39: 1) resize from 112x112 to 256x256; 2) random center crop to size 224x224; 3) grayscale conversion with 20% probability. At last, it is subtracted a mean RGB vector, obtained in [18], to the image, as previously explained.

The ResNet50 training for face identification is performed on the ResNet50 _8631-d architecture with the CEL function, as in the cResNet39. The model is trained with the mini-batch SGD optimizer, with mini-batches of 64 images and a learning-rate of 0.1 for 10 epochs. After training the ResNet50, the 8631-d output FC layer is replaced by a randomly initialized FC layer that generates 2048-d face embeddings. The ResNet50 model is fine-tuned with the ArcFace Loss [20][21] with the mini-batch SGD optimizer, with mini-batches of 64 images and a learning-rate of 0.005 for 5 epochs. Only the 2048-d FC layer and the extra ArcFace FC layer are updated, the rest of the ResNet50 weights are kept constant.

### *B. Performance Metrics Computation*

The anchors and proposed compressed domain solutions are evaluated on the Labeled Faces in the Wild (LFW) dataset [11][26], due to its popularity in face verification performance evaluation. The dataset is composed of 13233 images across 5749 different identities, with spatial size of 250x250 in *jpg* format. The images are converted to *png* format to simulate original images. The LFW dataset have available 2 face verification protocols one for validation and other for testing. It is used the testing LFW view 2 (LFW2) protocol described in [11], which has available 6k labelled testing pairs for face verification (with 3k positive pairs and 3k negative pairs), each pair contains two images (probe face and identity reference face) and the label for the verification task (True/False).

To evaluate the solutions performance, it was selected the most popular metrics in face verification tasks. The TAR@FAR metric indicates the value of TAR (True Acceptance Rate) for a fixed value of FAR (False Acceptance Rate). The FAR value was fixed at 0.001 and 0.01 (TAR@FAR=0.001, TAR@FAR=0.01). The Area Under the Curve (AUC) is also used, which measures the area under the Receiver Operating Characteristics (ROC) Curve with values in [0,1]. Finally, it is used the Equal Error Rate (ERR) which corresponds to the FAR value where the rates FAR and FRR (False Recognition Rate) are equal. The FRR value is given by 1-TAR. To obtain the performance metrics results the ROC curve is calculated, which represents the TAR vs FAR plot for a range of decision thresholds.

Additionally, it is calculated the computational complexity of the models as follows:

1) Number of Multiply-Accumulate (MAC) operations, which represents the number of neural networks basic operations, executed to obtain the face embedding of one input. It is also represented in number of MAC operations per pixel (MAC/px), considering an input image of 224x224; In all solutions it is not included the pre-processing of images or latent representations since the MAC value is only desirable for neural networks' operations.

2) Mean execution/computation time of the face verification task on the LFW test dataset (LFW2 protocol), in milliseconds per batch (of size 64 pairs, i.e., 128 images/latent representations) and in milliseconds per pair (1 probe and 1 reference images/latent representations). The face cropping, alignment and resizing is only considered for the faces which are not compressed. Hence, for the compressed model this process execution time is not included. While for the original anchor it is considered this execution time for both images in a pair. As for the decoded anchors, it only includes this process for the reference image.

The test process on the LFW dataset is performed 50x for each solution in order to obtain the mean execution time values, with batches of 64 pairs. The test is performed with the 64-bits NVIDIA GeForce RTX 3080 GPU [27], with 12 GB of memory (memory type: GDDR6X), clock frequency of 33 MHz (clock boost 1.71 GHz) and 8960 CUDA cores.

### *C. Performance Evaluation*

The performance evaluation presented in Fig. 4, given by bitrate-metric plots, is executed on the following implemented solutions:

**ResNet50_Original_Anchor** - Original anchor solution.

**ResNet50_Balle2018_mse_Qn** – Decoded anchor solution with the Balle_2018 codec optimized for the MSE loss function and model n (1, 3, 6 or 8) for each target quality.

**ResNet50_JPEG_Qn** - Decoded anchor solution with the JPEG codec and quality n (1, 5, 25 or 62). The used qualities have corresponding compression rates to the used qualities of the Balle_2018 codec.

**ResNet50_JPEG2000_mse_Qn** - Decoded anchor solution with the JPEG2000 codec optimized for the MSE loss function and model n (1, 3, 6 or 8) for each target quality.

**ResNet50_JPEG2000_visual_Qn** - Same as previous decoded anchors solution but the weights are visually optimized.

**ResNet50_HEVC_Qn** - Decoded anchor solution with the HEVC-Intra codec [5] and quality n (41, 35, 26 or 19) for each target quality. The used quality rates have corresponding compression rates to the used qualities of the Balle_2018 codec.

**cResNet39_Balle2018_mse_Qn** – Proposed compressed domain face verification solution with the Balle_2018 codec [8] optimized for the MSE loss function and model n (1, 3, 6 or 8) for each target quality.

Additionally, the computational complexity results of the solutions ResNet50_Original_Anchor, ResNet50_Balle2018_mse_Qn and cResNet39_Balle2018_mse_Qn solutions, for n in (1, 3, 6, 8), are presented in TABLE II. In the decoded anchors only the probe face is compressed, meaning that the compression rate, in bits per pixel (bpp), values can only be calculated for the probe image, since the reference face is an original image. To replicate this process of calculating the compression rate of the test dataset, for the compressed domain solution it will only be considered the bitrates of the probe faces.

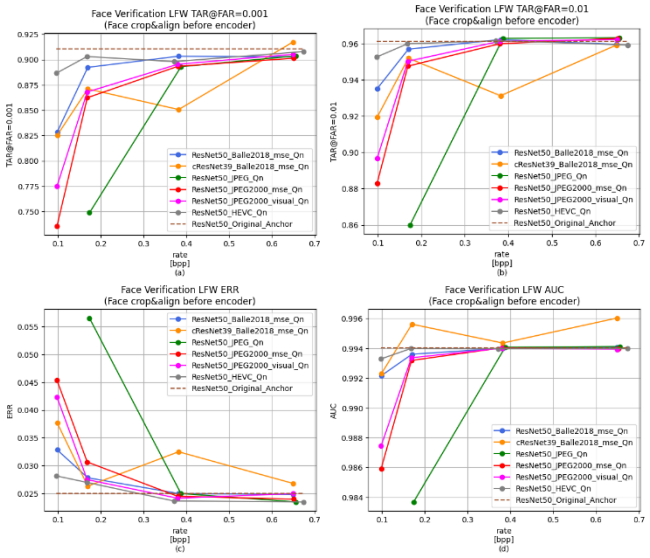


Fig. 4. Face verification results for all anchors (ResNet50) and novel compressed domain solutions (cResNet39) for the different mean rate for the test images of the LFW dataset with the LFW2 protocol (6k pairs). (a) TAR@FAR=$10^{-3}$; (b) TAR@FAR=$10^{-2}$; (c) ERR; (d) AUC.

TABLE II. COMPUTATIONAL COMPLEXITY AND FACE VERIFICATION PERFORMANCE RESULTS ON THE LFW DATASET (LFW2 PROTOCOL) OF THE RESNET50_ORIGINAL_ANCHOR, RESNET50_BALLE2018_MSE_QN AND CRESNET39_BALLE2018_MSE_QN SOLUTIONS, FOR N IN (1, 3, 6, 8). COMPUTATIONAL COMPLEXITY IS MEASURED WITH NUMBER OF OPERATIONS IN GMAC AND IN KMAC/PX (PER PIXEL) AND WITH COMPUTATIONAL TIME IN S/BATCH AND IN S/PAIR. EACH BATCH CONTAINS A TOTAL OF 64 PAIRS WHICH CORRESPONDS TO 128 IMAGES. THE FACE VERIFICATION PERFORMANCE IS MEASURED WITH THE TAR@FAR= $10^{-3}$ . (*) THE SOLUTIONS' COMPUTATIONAL TIMES ARE MEAN VALUES OBTAINED BY REPEATING THE TEST PROCESS 50X WITH THE NVIDIA RTX 3080 GPU [27].

| | Computational Complexity | | | | Face Verification (LFW) | |
|---|---|---|---|---|---|---|
| | Operations | | Time* | | | |
| Models | Total [GMAC] | Per pixel [kMAC/px] | Batches [ms/batch] | Pairs [ms/pair] | TAR@ FAR=$10^{-3}$ | Rate [bpp] |
| ResNet50_Original_Anchor | 3.89 | 78 | 205.3 | 3.216 | 0.910 | |
| ResNet50_Balle2018_mse_Q1 | 13.55 | 270 | 215.0 | 3.368 | 0.829 | 0.10 |
| ResNet50_Balle2018_mse_Q3 | 13.55 | 270 | 215.1 | 3.370 | 0.892 | 0.17 |
| ResNet50_Balle2018_mse_Q6 | 25.3 | 504 | 240.0 | 3.760 | 0.903 | 0.38 |
| ResNet50_Balle2018_mse_Q8 | 25.3 | 504 | 240.8 | 3.772 | 0.903 | 0.65 |
| cResNet39_Balle2018_mse_Q1 | 2.36 | 47 | 42.7 | 0.669 | 0.825 | 0.10 |
| cResNet39_Balle2018_mse_Q3 | 2.36 | 47 | 42.5 | 0.666 | 0.871 | 0.17 |
| cResNet39_Balle2018_mse_Q6 | 2.38 | 47 | 42.9 | 0.672 | 0.851 | 0.38 |
| cResNet39_Balle2018_mse_Q8 | 2.38 | 47 | 42.9 | 0.673 | 0.917 | 0.65 |

**Face Verification:** The TAR@FAR=0.01, ERR and AUC metrics do not have a major increase in the performance while increasing the bitrate, as shown in Fig. 4, being contained in the intervals [0.86, 0.965], [0.06, 0.02] and [0.983, 0.996] respectively. However, the AUC values are all close to the ideal value 1, which shows high performance results for all solutions in this metric. Likewise, the ERR and TAR@FAR=0.01 also show high performance results. This means that these metrics are not very discriminative, especially compared to TAR@FAR=0.001. The decoded anchors solutions using the Balle_2018 codec with MSE optimization (ResNet50_Balle2018_mse_Qn) achieve higher performance than the compressed domain model (cResNet39_Balle2018_mse_Qn) in almost all metrics and bitrates. Although the ResNet50_Balle2018_mse_Qn model performance results are better than the cResNet39_Balle2018_mse_Qn, the difference in performance can be considered small, since in any of the selected metrics the results vary more than a value of 0.06. The highest gap in performance between these two solutions is in the TAR@FAR=0.001 metric, in which the ResNet50_Balle2018_mse_Qn and cResNet39_Balle2018_mse_Qn solutions present the results 0.903 and 0.851 respectively for the target bitrate 0.38 bpp. Additionally, the cResNet39_Balle2018_mse_Q6 solution present significantly lower performance results in all metrics in its corresponding target bitrate, except for the AUC metric. However, even the AUC result of the cResNet39_Balle2018_mse_Q6 solution shows outlier behavior, i.e., the AUC metric plot decreases for the cResNet39_Balle2018_mse_Qn in the bitrate 0.38 bpp when increasing the target bitrate. This probably means that the weights random initialization of the cResNet39_Balle2018_mse_Q6 model, which may have led the model to a local minimum while training, have influenced the performance of the model, since the training process is the same for all compressed domain solutions. For the lowest target bitrate 0.10 bpp, the compressed domain solution cResNet39_Balle2018_mse_Q1 has significantly higher performance results in all metrics than the conventional decoded anchors, except the HEVC-Intra. However, the performance results of these solutions become similar with the increase of the bitrate. The HEVC-Intra decoded anchor solution (ResNet50_HEVC) presents the highest performance results at lower bitrates from all solutions. Consequently, the HEVC-Intra codec presents the highest facial reconstruction efficiency at lower bitrates. These results were expected since the HEVC-Intra is a high efficiency codec which competes against state-of-art codecs. However, for the highest target bitrate (0.649 bpp) the compressed domain model presents

higher performance in the TAR@FAR=0.001 than all anchors, including the original anchor.

**Computational Complexity:** From the computational complexity results presented in TABLE II. it can be observed that all compressed domain solutions have similar computational complexity (47 kMAC/px and ~0.67 ms/pair). The original anchor solution has higher complexity than the compressed domain solutions, with the values 78 kMAC/px and 3.216 ms/pair. This corresponds to an increase in computational complexity of 66% (x1.66) in the number of MAC operations per pixel and of 380% (x4.80) in the execution time. The learning-based decoded anchor solutions (ResNet50_Balle2018_mse_Qn) is divided into two types in terms of computational complexity:

**n = 1 or 3** – number of MAC operations equal to 270 kMAC/px and execution time of 3.370 ms/pair, hence there is an increase in computational complexity of these solutions of 474% (x5.74) and 402% (x5.02) respectively, when comparing with the compressed domain solutions.

**n = 6 or 8** – 504 kMAC/px and execution time of 3.770 ms/pair. This means that there is an increase in the number of MAC operation per pixel of 972% (x10.72) and an increase in the execution time of 463% (x5.63) in relation with the cResNet39_Balle2018_mse_Qn solutions.

However, in the original anchor solution mean execution time it is included the face cropping, alignment and resizing processes for both images in a pair. This image processing is not included in the execution times of the compressed domain model. In addition, in the decoded anchor the execution times only consider the image processing for the original reference image. By analyzing the image processing computational times, it was possible to observe that the performing this process (face cropping, alignment and resizing) for one image in the pair adds up an execution time of 0.28 ms/pair. This leads to new execution times of the decoded anchors (3.65 ms/pair for n=1, 3; 4.05 ms/pair for n=6, 8) and compressed domain (1.24 ms/pair) solutions. Consequently, the original anchor and decoded anchor solutions present an increase in execution time of 159% (x2.59) and of 227% (x3.27) respectively, with respect to the compressed domain solutions, when considering the image processing step.

The face verification performance results are unexpected, since it was foreseen that the compressed domain model would achieve higher performance than the decoded anchor solutions when using the same codec. However, in this assessment, it was used a model with lower complexity (cResNet39), as shown in TABLE I. and TABLE II. , for the compressed domain model than for the decoded and original anchors solutions (ResNet50). Additionally, the cResNet39 model was trained for fewer epochs (9 epochs total: 6 train epochs + 3 finetune epochs) than the ResNet50 model (15 epochs total: 10 train epochs + 5 finetune epochs), due to the high amount of time to training these models and due to time limitation. At last, the decoded anchors ResNet50 model is trained with original images of spatial size 224x224, in which downsampling is applied in residual blocks 2, 3 and 4, obtaining the size 7x7 after the last convolutional layer. Hence, the ResNet50 performs a spatial reduction of the image of x32 (224/32=7), meaning that when training the ResNet50, the weights are learning to perform this reduction more efficiently for face recognition tasks. Consequently, the decoded anchors solutions performance results only depend on the codec's efficiency in reconstruct the faces. However, the cResNet39 is trained with latent representations with spatial size 14x14, being only performed downsampling in the residual block 4 as described in Section III. After the cResNet39's last convolutional layer the output has a size of 7x7, which means a spatial reduction of x2. The downsampling process in the compressed domain solution is mainly executed by the encoder, which is not trained alongside the cResNet39. This means that the compressed domain model performance depends on the encoder reduction efficiency for face verification tasks. Finally, the compressed domain face verification model (cResNet39_Balle2018_mse_Qn) present high results when taking into consideration all the mentioned differences with the decoded anchors solutions.

It is then clear that the proposed compressed domain face verification solution is not able to surpass the decoded anchor solutions when using the same codec. However, the proposed solution is able to achieve similar face verification performance results (although lower) while having significantly lower computational complexity.

Given the relatively new concept of utilizing compressed representations of images for image analysis, it was inevitable that certain challenges would emerge. Considering the existing limitations, future efforts can be directed towards enhancing and conducting more robust performance assessments of solutions operating in the compressed domain as suggested:

**Re-train the cResNet39 with the Encoder:** train the encoder weights alongside with the face verification network, aiming to obtain latent representations that maximize face verification performance.

**Use more layers (cResNet50):** the ResNet50 can be modified to use compressed representations as input while maintaining the number of layers, obtaining the cResNet50 model. To achieve this goal, the max pool layer and convolutional layer from the root block of the ResNet50 need to be removed. To maintain the number of layers and the spatial size at the input of the residual block 1, a pixel shuffle upsampling layer with factor x4 can be added, which will increase the latent representations spatial size by x4 (to 56x56) and decrease the number of channels by x16. The upsampling layer can then be followed by a convolutional layer, which will adjust the number of channels.

## V. CONCLUSIONS

For the moment, the decoded solutions are a better option than the proposed compressed domain solution when it is desirable to have a high face verification performance, since it was shown that they can better verify a person's identity. However, for high bitrates this statement is not entirely true since all solutions achieved similar performance. Although, if the most relevant application concern is memory space and/or computational power (computational complexity) then the proposed compressed domain solution is the best candidate from the evaluated solutions. Nevertheless, a solution such as the one proposed in this paper may be adjusted (more convolutional layer or training the encoder) to surpass the decoded solutions face verification performance.

## REFERENCES

[1] ISO/IEC JTC 1/SC 29/WG1 N90049, "White Paper on JPEG AI Scope and Framework v1.0," 2021.J. Clerk Maxwell, A Treatise on

Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[2] G. K. Wallace, "The JPEG still picture compression standard," *IEEE Trans. Consum. Electronics*, vol. 38, no. 1, pp. 18-34, Feb. 1992, doi: 10.1109/30.125072.

[3] C. Christopoulos, A. Skodras and T. Ebrahimi, "The JPEG2000 still image coding system: an overview," *IEEE Trans. Consum. Electronics*, vol. 46, no. 4, pp. 1103-1127, Nov. 2000, doi: 10.1109/30.920468.

[4] M. W. Marcellin, M. J. Gormish, A. Bilgin and M. P. Boliek, "An overview of JPEG-2000," *Data Compression Conf.*, 2000, pp. 523-541, doi: 10.1109/DCC.2000.838192.

[5] V. Sze, M. Budagavi and G. J. Sullivan, "High efficiency video coding (HEVC)," in *Integrated circuit and systems, algorithms and architectures*, A. P. Chandrakasan, Ed., Switzerland: Springer, 2014, vol. 39, ch. 1, pp. 1-4.

[6] F. Bossen, B. Bross, K Suhring and D. Flynn, "HEVC Complexity and Implementation Analysis," *IEEE Trans. Circuits and Syst. For Video Technol.*, vol. 22, no. 12, pp.1685-1696, Dec. 2012, doi: 10.1109/TCSVT.2012.2221255.

[7] B. Bross, Y. K. Wang, Y. Ye, S. Liu, J. Chen, G. J. Sullivan and J. R. Ohm, "Overview of the Versatile Video Coding (VVC) Standard and its Applications," *IEEE Trans. Circuits and Syst. for Video Technol.*, vol. 31, no. 10, pp. 3736-3764 , Oct. 2021, doi: 10.1109/TCSVT.2021.3101953.

[8] J. Ballé, D. Minnen, S. Singh, S. J. Hwang and N. Johnston, "Variational image compression with a scale hyperprior," *Int. Conf. Learn. Representations*, Vancouver, Canada, Apr. 2018.

[9] Z. Cheng, H. Sun. M. Takeuchi and J. Katto, "Learned Image Compression with Discretized Gaussian Mixture Likelihoods and Attention Modules," *arXiv*, pp. 1-15, Mar., 2020, doi: 10.48550/ARXIV.2001.01568.

[10] R. Torfason, F. Mentzer, E. Agustsson, M. Tschannen, R. Timofte and L. Van Gool, "Towards image understanding from deep compression without decoding," *arXiv:1803.06131*, Mar. 2018.

[11] G. B. Huang, M. Mattar, T. Berg and E. Learned-Miller, "Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments," *Workshop Faces in 'Real-Life' Images: Detection, Alignment and Recognit.*, Oct. 2008.

[12] Q. Cao, L. Shen, W. Xie O. M. Parkhi and A. Zisserman, "VGGFace2: A Dataset for Recognising Faces across Pose and Age," *2018 13th IEEE Int. Conf. Autom. Face and Gesture Recognit.*, May 2018, pp. 67-74, doi: 10.1109/FG.2018.00020.

[13] J. Deng, J. Guo, J. Yang, N. Xue, I. Kotsia and S. Zafeiriou, "ArcFace: Additive Angular Margin Loss for Deep Face Recognition," *arXiv*, vol. arxiv.1801.07698v3, Feb, 2019, doi: 10.48550/ARXIV.1801.07698.

[14] J. Deng, J. Guo, Y. Zhou, J. Yu, I. Kostia and S. Zafeiriou, "RetinaFace: Single-stage Dense Face Localisation in the Wild," *CoRR*, vol. abs/1905.00641, pp. 1-10, May, 2019.

[15] retinaface. (2022). S. I. Serengil. [Online]. Available: https://github.com/serengil/retinaface.

[16] CompressAI. (2022). InterDigitalInc. [Online]. Available: https://github.com/InterDigitalInc/CompressAI/.

[17] Image Compression. (2022). CompressAI. [Online]. Available: https://interdigitalinc.github.io/CompressAI/zoo.html.

[18] VGGFace2-pytorch (2018). cydonia. [Online]. Available: https://github.com/cydonia999/VGGFace2-pytorch.

[19] K. E. Koech. "Cross-Entropy Loss Function." Towards Data Science. Available: https://towardsdatascience.com/cross-entropy-loss-function-f38c4ec8643e. (accessed May 28, 2023).

[20] J. Deng, J. Guo, J. Yang, N. Xue, I. Kotsia and S. Zafeiriou, "ArcFace: Additive Angular Margin Loss for Deep Face Recognition," *arXiv*, vol. arxiv.1801.07698v3, Feb, 2019, doi: 10.48550/ARXIV.1801.07698.

[21] Losses. (2022). PyTorch Metric Learning. [Online]. Available: https://kevinmusgrave.github.io/pytorch-metric-learning/losses/#arcfaceloss.

[22] VGGFace2 Dataset. Q. Cao, L. Shen, W. Xie, O. M. Parkhi and A. Zisserman. [Online]. Available: https://www.robots.ox.ac.uk/~vgg/data/vgg_face2/.

[23] JPEG AI Quality Assessment Framework. (2022). JPEG AI. [Online]. Available: https://gitlab.com/wg1/jpeg-ai/jpeg-ai-qaf/-/tree/main/examples.

[24] Downloads. (2020). Kakadu Software. [Online]. Available: https://kakadusoftware.com/documentation-downloads/downloads/.

[25] HEVC HM reference software. (2023). Fraunhofer HHI. [Online]. Available: https://vcgit.hhi.fraunhofer.de/jvet/HM.

[26] Labeled Faces in the Wild, University of Massachusetts Amherst, 2018. [Online]. Available: http://vis-www.cs.umass.edu/lfw/.

[27] NVIDIA Corporation. "Compare GeForce Graphics Cards". NVIDIA. https://www.nvidia.com/en-eu/geforce/graphics-cards/compare/?section=compare-specs (accessed May 24, 2023).

[28]