WaterLo: Protect Images from Deepfakes Using Localized Semi-Fragile Watermark

Nicolas Beuve
Univ. Rennes, INSA Rennes, CNRS, IETR - UMR 6164,
20 Av. des Buttes de Coesmes, 35700 Rennes, France
nicolas.beuve@insa-rennes.fr

Wassim Hamidouche
Technology Innovation Institute,
Masdar City, Abu Dhabi, UAE
wassime.hamidouche@tii.ae

Olivier Déforges
Univ. Rennes, INSA Rennes, CNRS, IETR - UMR 6164,
20 Av. des Buttes de Coesmes, 35700 Rennes, France
olivier.deforges@insa-rennes.fr

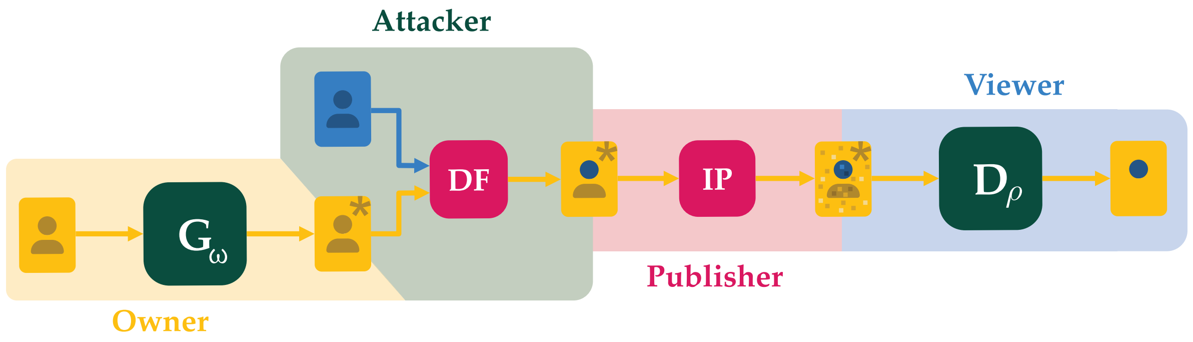


Figure 1: Illustration of a practical use case of our proposed method. The owner takes a picture and protects it using a watermark (represented by a $*$). The attacker then produces a Deepfake (DF) using the said image as target (or source). Then, the publisher (usually a social network) applies some Image Processing (IP) before publishing (re-scale, compression, ...). Finally, a viewer can verify whether the watermark is present using the detector D_ρ and where the image was modified.

Abstract

Most existing contributions in the field of Deepfake detection focus on passive detection methods, where the detector only analyzes the doctored image. However, this approach often lacks the ability to generalize to unseen data and struggles to detect Deepfakes generated using new deepfake models. To address this limitation, our paper proposes an active detection approach, where we have access to the image before the Deepfake is generated. Our solution involves applying a watermark that disappears in modified regions, allowing our detector to identify image modifications and localize them accurately. Additionally, we incorporate a compression module into our training pipeline to enhance the watermark's robustness against JPEG compression. Experimental results demonstrate the effectiveness of our proposed solution, achieving a remarkable detection accuracy of 97.83% while maintaining significantly higher image quality compared to previous works. Furthermore, by incorporating the compression module in the training pipeline, we improve the detection accuracy on compressed samples, albeit with a slight decrease in accuracy for non-compressed samples. This contribution also provides a valuable tool for video owners to verify if their videos have been tampered with and safeguard them against unauthorized use. The code of the proposed framework is available at `https://github.com/beuve/waterlo`.

1. INTRODUCTION

Impersonating trusted individuals, especially through video, has become an effective means of spreading fake news, given the high level of trust associated with this medium [27]. The emergence of Deepfakes has further exacerbated the potential for destabilization, as political representatives can be impersonated, posing a significant threat to a country's stability [28]. In recent years, the proliferation of user-friendly tools granting access to state-of-the-art generative models (e.g., GANs [13] and VAEs) has led to a substantial increase in both the quantity and quality of Deepfakes on social media platforms. Consequently, researchers and major tech companies have turned their attention towards the field of Deepfake detection, aiming to develop effective countermeasures against this novel form of misinformation. The majority of existing contributions in Deepfake detection focus on passive detection techniques, where the detector does not have access to the original or unaltered source image. These methods rely on exploiting the lack of realism exhibited by generated content. While generative models can produce visually compelling images that can deceive the human visual system [26], they often fail to accurately replicate subtle image characteristics such as camera fingerprints [4], frequency patterns [11], or human-like behavior such as eye blinking [7] and heartbeat [15]. However, recent detection techniques primarily rely on end-to-end models that fail in generalization to unseen Deepfake methods. Thus, there is a need for more robust and adaptable detection approaches to effectively address the evolving nature of Deepfakes.

When faced with doubts about the authenticity of an online video, individuals often turn to passive detectors to aid their decision-making process regarding trustworthiness. However, even with the availability of such detectors, there is no guarantee that end users will take the initiative to verify the integrity of a video if the Deepfake is convincing enough. Recognizing this challenge, researchers have recently directed their attention towards active detection methods, which involve modifying the video before the Deepfake is generated. By employing subtle manipulations, active detection techniques can prevent the creation of Deepfakes using adversarial attacks, which are carefully crafted imperceptible perturbations [2, 22]. Another approach for video owners to establish the untrustworthiness of a Deepfake is by providing the original sample. This can be achieved, for instance, through digital watermarking techniques [23], allowing the video owner to prove that their video is the pristine, unaltered version. However, a key question arises regarding what the watermark detector should predict for a Deepfake. There are two possibilities: either the watermark has disappeared, leaving the owner unable to prove anything, or the watermark is still visible, enabling the owner to demonstrate ownership of both the pristine video and the Deepfake, but not conclusively determining which one is real.

To address this question, our paper proposes the use of localized semi-fragile watermarking, as illustrated in Figure 1, to provide insights into the regions of the image that have been manipulated. Our model applies a semi-fragile watermark throughout the entire image, and the detector, trained end-to-end with the watermark generator, can identify where the watermark is visible. For instance, in Deepfake attacks, the detector would detect the watermark throughout the image except in the manipulated face region. Additionally, the watermark must be robust to benign manipulations, such as compression, as this is a common modification applied by social media platforms before video publication.

The main contribution of this paper is the design of an end-to-end model for localized semi-fragile watermarking, which effectively detects modified regions in images, with a particular focus on Deepfake manipulation detection, while also maintaining robustness against benign manipulations like compression.

The rest of this paper is organized as follows. First, Section 2 briefly reviews digital watermarking and Deepfake detection, followed by the description of the proposed method in Section 3. Next, the performance and robustness of the proposed watermark are assessed in Section 4. Finally, Section 5 concludes the paper.

2. Related Work

This section provides a brief overview of Deepfake detection techniques, categorized as passive or active detection.

2.1. Passive Detection

The majority of published papers in the field of Deepfake detection focus on passive detection, which is particularly beneficial for both publishers (enabling them to refuse to post or inform viewers) and viewers. These techniques typically begin by representing image information as a vector of features. Passive detection can be classified into three categories based on the extraction of image features: hand-crafted, physiological, or deep features.

Hand-crafted features. Deepfake generation introduces artifacts that are not naturally occurring and can be directly detected within the image. Examples include traces left by convolution operations [14], warping [19], and blending [16] artifacts. Several studies have also explored traces left in other components of videos, such as the frequency domain using techniques like Discrete Cosine Transform (DCT) [12] or Fourier transform [11], motion analysis using Euler video magnification [6], optical flows [1], or texture analysis using Local Binary

Pattern (LBP) [17].

Physiological features. While generative models excel at producing visually appealing images, they often fail to accurately represent certain human traits. Deepfakes can thus be detected by focusing on traits that generative models may misrepresent, such as eye movements [7], heartbeats [15], or even biometric traits for identity verification [10, 5], .

Deep features. The final category encompassing methods that do not rely on expert knowledge to train the underlying backbone. Various deep architectures have been explored, including Transformers [31], capsule networks [24], and Long Short Term Memory (LSTM) [30]. However, the remarkable performance of Convolutional Neural Network (CNN), such as Xception [3] and EfficientNet [29], in the Facebook Deepfake detection challenge [9] has solidified CNNs as the leading architecture for this task.

2.2. Active Detection

Unlike passive detection, active detection techniques require access to the original image before the Deepfake manipulation is applied. Active detection can be classified into three main approaches: adversarial attacks, robust watermarking, and semi-fragile watermarking.

Adversarial attacks. Adversarial attacks involve the careful crafting of imperceptible perturbations added to the original image with the aim of fooling state-of-the-art deep neural classification networks. In active detection, the original image is subjected to an adversarial attack to prevent Deepfake generation. The adversarial perturbation, such as noise, is amplified when the image is manipulated using a generative model, rendering the Deepfake attack ineffective. In most contributions [22, 25], the adversarial perturbation is initialized with random noise and then optimized through gradient descent on the pixel values. However, active detection based on adversarial attacks often exhibits weak generalization to unseen generative models, similar to the challenges faced by deep features in passive detection. A recent paper [2] proposed a novel architecture to simultaneously train against various generative models. The authors argue that since only a limited number of pre-trained generative models are commonly used, training the adversarial attack on these prevalent models would cover most real-world Deepfakes. Their method involves a shared generative model that takes the source image and a learned perturbation as inputs, generating an adversarial perturbation. The learned perturbation is trained separately for each Deepfake generation model, resulting in multiple adversarial perturbations. A fusion model is then used to combine these adversarial perturbations into a single perturbation.

Robust watermarking. Watermarks are patterns containing embedded secret messages within an image, often used for owner identification. In the context of Deepfake detection, several studies have employed robust watermarking techniques. The concept behind these methods is to incorporate information about the original image, such that any modifications to the image would result in a mismatch between the embedded information and the visible content. The specifics of these methods vary, including the content of the embedded message. For instance, some approaches [33, 32] incorporate an identifying key associated with the person in the video, assuming a database of possible Deepfake targets with corresponding keys. Alternatively, other works [35] propose embedding a feature vector describing the person, where authentication involves computing a new feature vector and comparing it with the embedded one using correlation. However, one limitation of these methods is their assumption of robustness against Deepfake generation, which requires a high-intensity watermark to remain detectable. In contrast, our proposed method does not exhibit robustness against Deepfake generation, allowing it to operate with a lower level of intensity.

Semi-fragile watermarking. A fragile watermark is designed to be irrecoverable when the image has been modified. In contrast, semi-fragile watermarks exhibit robustness against benign modifications (e.g., compression) but not against malicious modifications like Deepfakes, making them suitable for ensuring image integrity. Previous work [23] utilized semi-fragile watermarking exclusively on face regions, with the embedded message becoming unrecoverable if a Deepfake was produced. One advantage of that approach is that it does not require limiting the training dataset to deepfake images. However, a drawback of this approach is that Deepfakes no longer retain the watermark, making it impossible to determine whether the video has been modified (only the absence of a watermark can be ascertained). In contrast, our proposed method involves adding a watermark to the entire image and then having the Deepfake remove the watermark in the face region. This enables us to localize the modified regions, as the watermark remains visible elsewhere. Our contribution resides in the introduction of a deep-learning autoencoder tailored for generating localized semi-fragile watermarking. While the idea of localizing altered regions through semi-fragile watermarks has been investigated in earlier works [21, 20], we note that, to the best of our knowledge, its specific application to Deepfakes has not been addressed before, nor has it been harnessed through an end-to-end deep learning framework.

3. PROPOSED METHOD

The proposed method involves the embedding of an invisible watermark into the image prior to Deepfake generation. This watermark consists of a 1-bit message that is uniformly applied to the original image. The purpose of the watermark is to be removed from a block if that block has undergone any modifications. The detector is designed to identify the presence of the watermark in each pristine block while being unable to detect it in modified blocks. Therefore, if the detector can detect the watermark in all regions of the image except for the area surrounding a face, it can be inferred that the face has been manipulated. To address the specific scenario depicted in Figure 1, our training pipeline incorporates a Deepfake generator and a compression module. This enables us to train the model to detect Deepfakes with the generated watermark, while also ensuring robustness to benign modifications, such as compression.

3.1. Watermark Generation

The forward pass starts with the watermark generation:

$$I^* = I + \alpha G_\omega(I), \tag{1}$$

where I is the original image, G_ω is the watermark generator parameterized by ω, and α is an hyper-parameter controlling the intensity of the watermark. The watermark generator is a U-net architecture that inputs the pristine image and produces a watermark pattern. The pattern is then added to the pristine image to produce the watermarked image I^*.

3.2. Deepfake Generation

Incorporating a state-of-the-art Deepfake generation method into our end-to-end pipeline would significantly increase the computational complexity of the training process and potentially hinder our model's ability to generalize to unseen Deepfake generation models. Therefore, inspired by the approach in [23], we modeled the effect of the Deepfake generation method by reducing the intensity of the watermark in a randomly selected location of the image, as depicted in Figure 2. This allows us to derive the composite image, denoted as I^*_{DF}, which combines both the watermark and the Deepfake. The derivation of I^*_{DF} can be described as follows:

$$I^*_{DF}(\theta, x, y) = \begin{cases} I^*(\theta, x, y) & \text{if } (x, y) \notin [a, a+w] \times [b, b+h], \\ (1-\gamma) \times I^*(\theta, x, y) + \gamma \times I(\theta, x, y) & \text{Otherwise}, \end{cases} \tag{2}$$

where $[a, a+w] \times [b, b+h]$ is a randomly chosen rectangle of resolution $w \times h$ within the dimension of image I, and (a, b) is its top left corner. Within this rectangle, a scalar γ is used to control the blending between the watermarked and pristine region of the image.

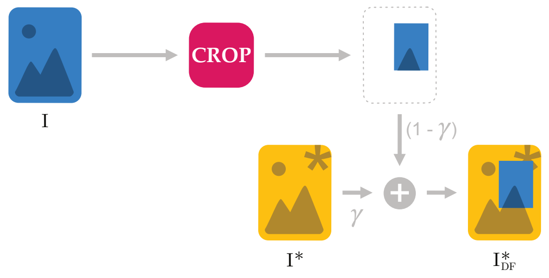


Figure 2: During training, we simulate Deepfakes by blending a region of the pristine image (in blue) in its corresponding watermarked image (in yellow). γ is the blending factor.

3.3. Robustness to Compression

Compression is a commonly applied image-processing operation used to efficiently store and share images over social networks. Consequently, ensuring the robustness of the watermark against compression is of utmost importance. Previous active Deepfake detection methods [23, 2] have utilized differentiable JPEG, which replaces non-differentiable components with differentiable counterparts. In this paper, drawing inspiration from the work of Zhang *et al.* [34], we propose an alternative approach. During the training stage, we approximate the non-differentiable JPEG operation using additive noise, as expressed by Equation 3. This operation is differentiable, allowing us to compute gradients in the backward pass over the pseudo-differentiable JPEG, as depicted in Figure 3. The advantage of our method, compared to using differentiable components in JPEG, lies in the fact that we employ the JPEG codec exactly as it is used in real-world scenarios.

$$I^*_{DF+CP} = I^*_{DF} + N_{JPEG}, \tag{3}$$

where N_{JPEG} refers to the JPEG noise.

3.4. Watermark Detection

The final step involves the localization of the watermark within the image. The output of the detector, denoted as m_{pred}, is a normalized heat map where high values (i.e., $\sim$ 1) indicate a high confidence in watermark detection, while low values correspond to lower confidence. The detector, represented as D_ρ and parameterized by ρ, adopts a U-net architecture that is adapted with fewer up layers than down layers. This adaptation results in an output size smaller than the input, with the size of the output being a hyperparameter. In this paper, we utilized an output size of $N \times N$. Formally, m_{pred} is defined as:

$$m_{pred} = D_\rho(I^*_{DF+CP}), \tag{4}$$

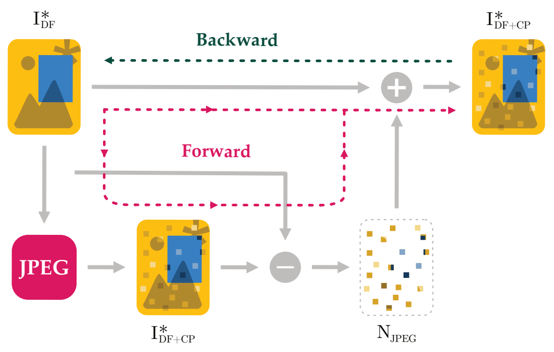


Figure 3: Illustration of the pseudo-differentiable JPEG. The forward pass consists of applying the JPEG on the image and computing the JPEG noise N_{JPEG} by subtracting the original image from the JPEG image. The JPEG is later reapplied to the original image by adding the JPEG noise. During the backward pass, only the last addition is differentiated.

3.5. Deepfake Detection

Our watermark detector generates a $N \times N$ heat map that indicates the areas where the watermarked image has been altered. In order to analyze this map, it is necessary to determine whether the image has been modified in the face region. Each image can fall into one of three classes: pristine image (I), watermark without Deepfake (I^*), or watermark with Deepfake (I^*_{DF}). The primary goal is to distinguish the pristine image (I) from the watermarked images (I^*, and I^*_{DF}) by detecting the presence of a watermark in the image. To achieve this, we propose to use the mean value μ of the heat map corresponding to the non-face region. By comparing the mean value μ to a fixed threshold τ, we can detect the presence of a watermark. If the mean value μ is higher than the threshold, it indicates the presence of a watermark.

Once the watermark is detected, the next step is to differentiate between the classes I^* and I^*_{DF}. This step is challenging because Deepfake methods may alter only a small portion of the face. To address this challenge, we propose labeling an image as a Deepfake (I^*_{DF}) if at least 10% of the pixels in the face region of the heat map are bellow the threshold (τ).

3.6. Loss Function

Regarding the loss function, two functions are required for this problem. The first one, denoted as $\mathcal{L}_D$, controls the detection performance of the detector (D_ρ). It is based on the cross-entropy between the predicted output of D, m_{pred} and the ground truth m_{gt}. The ground truth consists of a heat map with ones indicating the Deepfake region and zeros elsewhere. The second loss function, denoted as $\mathcal{L}_G$, regulates the visual perception of the watermark within the image. Several loss functions are considered in the literature, with the most popular ones being Mean Squared Error (MSE) and Structural Similarity Index Measure (SSIM). Overall, the optimization problem for this task can be expressed as follows:

$$\rho^*, \omega^* = \underset{\rho,\omega}{\arg\min}\ \mathcal{L}_G + \lambda \mathcal{L}_D, \tag{5}$$

where λ is an hyper-parameter that balances the two losses.

4. Results

4.1. Experimental setup

We employed the Adam optimizer with a learning rate of 2.10^{-4} to train our model. Regarding the hyperparameters, we set $\lambda = 1$ (see Equation 5), and we experimented with three values of α in Equation 1: $\alpha \in \{0.002, 0.005, 0.007\}$. During training, we randomly selected γ in Equation 2 and the JPEG quality (if enabled). For each image, we uniformly sampled γ from the interval $[0.5, , 0.9]$, and for each batch, we uniformly selected the compression quality factor from the interval $[0.5, , 0.9]$. The experimental results demonstrated that a threshold value of $\tau = 0.8$ and an output size of $N = 64$ yielded the best detection performance. For more comprehensive information on the architectures of the generator and detector, please refer to Appendix A.

Dataset. We trained our solution using images from the ImageNet dataset [8], as our task did not require Deepfake or specific facial images. The pristine images were initially zero-padded to attain a resolution of 512×512 pixels (if the original images had a higher resolution, they were scaled down while preserving the aspect ratio). Once the watermark was generated, the padding was removed, and the image was scaled back to 512×512 pixels before being processed by the detector. To train the baseline models and evaluate all models, Deepfakes were necessary. Therefore, we utilized the pristine videos from the FaceForensics++ (FF++) dataset and generated Deepfakes using two Deepfake methods: Faceshifter (FSh) [18], and Faceswap (FS) [26]. We selected 100 frames per video and followed the same training/validation/testing split as in FF++ (i.e., 720/140/140 split for 1000 test videos).

Evaluation metrics. To evaluate the performance of Deepfake detection, we utilize balanced accuracy. For assessing the quality of watermarked images, we employ the Peak Signal-to-Noise Ratio (PSNR). It is important to note that for Deepfake detection evaluation, we only

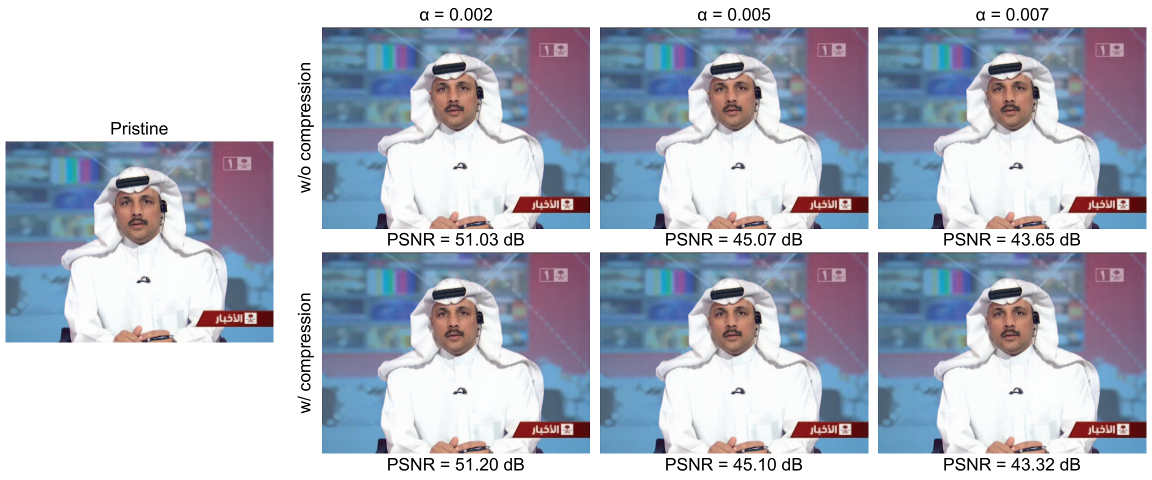


Figure 4: Examples of watermarked images with out proposed solution. Additional watermarked images are provided in Appendix C.

consider images that have a watermark. Moreover, it is worth highlighting that our detector demonstrates no false positive detection of the watermark. In other words, the detector never erroneously detects a watermark in the absence of one. The two classes we consider are watermarked images with Deepfake (I^*_{DF}) and watermarked images without Deepfake (I^*).

4.2. Quality Assessment

Samples of watermarked images with different values of $\alpha \in \{0.002, 0.007, \text{ and, } 0.005\}$ are presented in Figure 4. It can be observed that the visibility of the watermark varies with the different models trained using these values. With $\alpha = 0.007$ and $\alpha = 0.005$, the watermark is slightly visible, while it becomes nearly invisible with $\alpha = 0.002$. Additionally, the visibility of the watermark is influenced by its pattern. The models generate the watermark as a square signal, which is easily detectable by the human visual system. This explains why the watermark remains slightly visible at $\alpha = 0.005$ despite achieving a high PSNR value of 45.07 dB. When compression is introduced during training, there is a slight impact on the visibility of the watermark. Although the PSNR value remains the same, the frequency of the square signal decreases, making it more visible to the human visual system, while being more robust to compression as illustrated below.

In previous works [33, 23, 2], the watermarked images had a PSNR below 36 dB. However, our method generates samples with a higher PSNR, exceeding 43 dB. This can be attributed to the fact that the generators in [33, 23] directly produce the watermarked images, whereas in our approach, the generators focus solely on generating the watermark itself. Furthermore, previous methods aimed at recovering a secret message from the watermark, while our solution is focused solely on detecting a 1-bit per block watermark without any message to recover. This distinction further simplifies the detection task and thus reduces the intensity of the watermark.

4.3. Deepfake Detection

This section focuses on Deepfake detection using models trained and tested without compression.

Examples of heat-map from our model are shown in Figure 5. The first observation is that the presence of the watermark does not affect the generation of Deepfakes. Secondly, in Figure 5(b), a watermarked image displays a rectangle on the right border where the watermark is not detected. This sample is classified as ”watermark without Deepfake” since the blue region is small in comparison to the image and does not overlap with the face region. Finally, the examples of detection on FSh and FS Deepfakes in Figure 5(c) and (d), respectively, demonstrate that the watermark is not detected in the face region and would easily be classified as a Deepfake by a human operator.

Additional quantitative results are presented in Figure 6. On average, across all values of α, our model demonstrates lower accuracy in detecting FS Deepfakes (92.76% with MSE and 94.22% with SSIM) compared to detecting FSh Deepfakes (94.48% with MSE and 94.72% with SSIM). This can be explained, as illustrated in Figure 5, by the fact that the modified regions when using FSh are much larger than those when using FS, making FSh's modifications easier to detect. Additionally, incorporating SSIM as a quality metric in the loss function leads to more accurate models compared to using MSE.

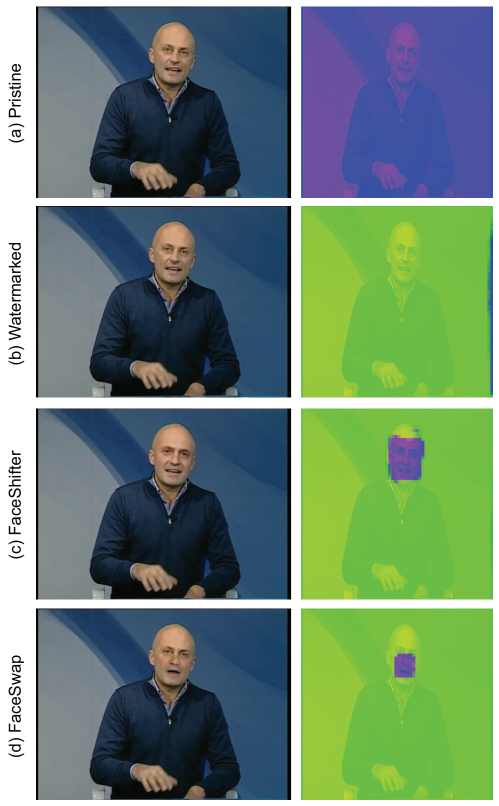
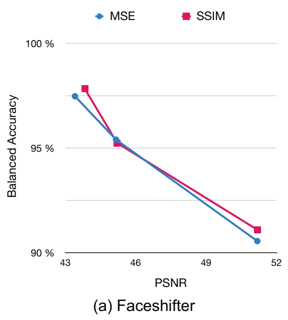


Figure 5: Example of input/output pairs from the detector. The model was trained using $\alpha = 0.002$ and SSIM as reconstruction loss. The output is green in regions where the watermark was detected, and blue where it was not. Similar results obtained with models trained with MSE are given in Appendix B.



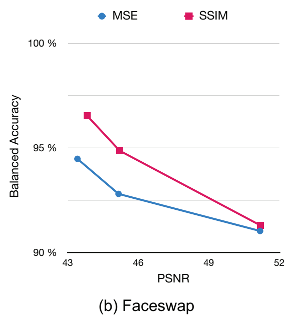


Figure 6: Balanced accuracy of Deepfake detection dependant of PSNR (dB) from models using either MSE (blue) or SSIM (red) as reconstruction loss and Faceshifter (a) or FS (b) as Deepfake generation model.

4.4. Generalization to Unseen Data

An essential aspect of a Deepfake detection model lies in its ability to generalize to unseen generation methods. Typically, end-to-end detection or adversarial attack models are trained on specific generation methods [2, 26], resulting in poor performance when faced with unseen methods. In contrast, our model was not trained using a particular Deepfake generation method, but rather on a simulation of Deepfake manipulations. Consequently, the Deepfake methods tested in this paper are considered as unseen data for our model.

Table 1 presents a performance analysis of an Xception model trained in a passive detection manner on each method and tested across datasets. When trained and tested on the same dataset, Xception exhibits superior performance compared to our proposed model (with a 1.41% improvement on FSh and 3.15% on FS). However, when tested in a cross-dataset setting, the performance of Xception declines, while our model demonstrates significantly higher accuracy (with a 26.55% improvement on FSh and 36.37% on FS).

Table 1: True Positive Rate (TPR), True Negative Rate (TNR), and Balanced ACCuracy (BACC) of Deepfake detection from either a passive model (i.e., Xception) or our proposed active detection model. The models are also tested in a cross-dataset setting with a different dataset than the one they were trained on to assess their generalization performance. The best results in the cross-dataset settings are shown in bold.

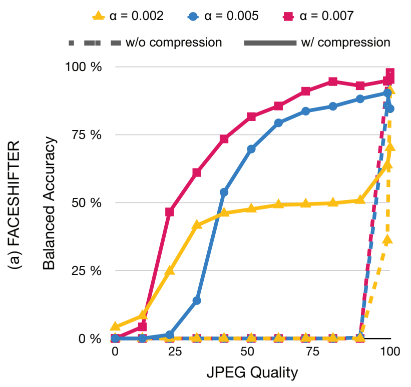
Method	TR	TE	TPR	TNR	BACC
Xception	FSh	FSh	98.49%	100%	99.24%
		FS	20.31%	100%	60.15%
	FS	FSh	42.65%	99.90%	71.28%
		FS	99.45%	99.90%	99.67%
Ours (SSIM)	-	FSh	97.07%	98.58%	**97.83%**
		FS	94.46%	98.58%	**96.52%**
Ours (MSE)	-	FSh	98.54%	96.40%	97.47%
		FS	92.54%	96.40%	94.47%

TR: Training dataset TE: Testing dataset

4.5. Robustness to Compression

We conducted training experiments with and without a compression module to evaluate the effectiveness of the pseudo-differentiable JPEG compression. Figure 7 illustrates a comparison of the balanced accuracy of our model with and without the compression module at various levels of JPEG compression quality.

The results demonstrate significant improvements in detection accuracy achieved by the compression module, particularly at low and medium compression quality levels. For instance, the watermark trained without the compression module is never detected when JPEG qualities are below 90%. Conversely, our generated watermarks exhibit robustness to high compression ratios, achieving a balanced accuracy of nearly 50% even at a JPEG quality of 20% for $\alpha = 0.007$. For lower values of α, the model can still detect the watermark, but it no longer identifies a modified region. This explains the 50% accuracy (almost 100% accuracy in detecting watermarks but 0% in detecting Deepfakes) observed for $\alpha = 0.002$. Our model also demonstrates reduced robustness to compression when detecting FS images compared to FSh, which can be attributed to the smaller modified region (see Figure 5).



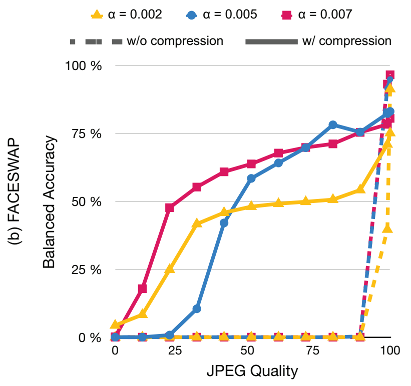


Figure 7: Balanced accuracy of Deepfake detection versus the JPEG quality factor (in %) of our solution using SSIM as reconstruction loss with different values of α and FSh (a) or FS (b) as Deepfake generation models. Similar curves obtained with models trained with MSE are given in Appendix B. 'w/o compression' represents models trained without compression, while 'w/ compression' represents models trained with compression.

Previous studies [33, 32] have shown robustness to compression without incorporating a compression module during training. In contrast, our solution lacks robustness without the compression module. This discrepancy can be explained by the lower PSNR values in their methods, which allow the watermark to remain visible after compression. In contrast, our higher PSNR values necessitate the use of a compression module to detect the watermark after compression.

While training for robustness to compression enhances the detection of compressed samples, it also reduces the model's performance on uncompressed samples. For instance, our model trained without compression, using SSIM and $\alpha = 0.005$, achieved an accuracy of 95.24% on FSh samples without compression. Conversely, the same model trained with compression only achieved 84.52% accuracy on the same uncompressed samples. It is important to note that our results can reach 0% accuracy, whereas in passive detection results, 50% represents the lower bound (corresponding to random classification). This discrepancy arises because we did not include samples without a watermark in our results, as our focus was on scenarios where individuals use the watermark to protect their images. However, even though all the tested samples are watermarked, they can still be classified as pristine if the watermark is not detected at all, thereby resulting in 0% accuracy. Lastly, it is worth highlighting that we observed no false-positive detections, indicating that our detector never identifies a watermark when none is present.

5. Conclusion

In this paper, we have presented a novel approach for detecting Deepfakes through a localized semi-fragile watermark solution. Our method takes advantage of the fact that Deepfake generation tends to diminish the presence of watermarks specifically in the face region. By exploiting this behavior, our detector can effectively identify modified regions in the image. Moreover, our model is trained in an unsupervised manner and incorporates a compression module to enhance robustness against compression artifacts. The experimental results on the testing set of the FF++ dataset have demonstrated the effectiveness of our proposed solution. When trained with the SSIM loss, our model achieved high detection accuracy for FSh and FS Deepfakes, with rates of 97.83% and 96.52% respectively. In addition, our solution generated watermarked images of good quality, surpassing state-of-the-art methods with PSNR values exceeding 43 dB. Notably, our approach exhibited remarkable generalization ability, outperforming classical passive detection solutions even in cross-dataset scenarios. For future research, we aim to further refine our approach by exploring methods to control the watermark pattern, making it less visible to the human visual system. This would contribute to improving the overall invisibility and effectiveness of our watermarking technique.

References

[1] I. Amerini, L. Galteri, R. Caldelli, and A. Del Bimbo. Deepfake Video Detection through Optical Flow Based CNN. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 1205–1207, 2019.

[2] Shivangi Aneja, Lev Markhasin, and Matthias Niessner. TAFIM: Targeted Adversarial Attacks against Facial Image Manipulations, July 2022. arXiv:2112.09151 [cs].

[3] François Chollet. Xception: Deep Learning with Depthwise Separable Convolutions. *CoRR*, abs/1610.02357, 2016. _eprint: 1610.02357.

[4] Davide Cozzolino, Giovanni Poggi, and Luisa Verdoliva. Extracting camera-based fingerprints for video forensics. In *CVPR Workshops*, 2019.

[5] Davide Cozzolino, Andreas Rössler, Justus Thies, Matthias Nießner, and Luisa Verdoliva. ID-Reveal: Identity-aware DeepFake Video Detection. *arXiv:2012.02512 [cs]*, Dec. 2020. arXiv: 2012.02512.

[6] Rashmiranjan Das, Gaurav Negi, and Alan F. Smeaton. Detecting deepfake videos using euler video magnification. *ArXiv*, abs/2101.11563, 2021.

[7] Ilke Demir and Umur Aybars Ciftci. Where do deep fakes look? synthetic face detection via gaze tracking. *ACM Symposium on Eye Tracking Research and Applications*, 2021.

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[9] Brian Dolhansky, Joanna Bitton, Ben Pflaum, Jikuo Lu, Russ Howes, Menglin Wang, and Cristian Canton Ferrer. The deepfake detection challenge dataset, 2020.

[10] Xiaoyi Dong, Jianmin Bao, Dongdong Chen, Weiming Zhang, Nenghai Yu, Dong Chen, Fang Wen, and Baining Guo. Identity-driven deepfake detection. *ArXiv*, abs/2012.03930, 2020.

[11] Ricard Durall, Margret Keuper, Franz-Josef Pfreundt, and Janis Keuper. Unmasking deepfakes with simple features. *ArXiv*, abs/1911.00686, 2019.

[12] Oliver Giudice, Luca Guarnera, and Sebastiano Battiato. Fighting deepfakes by detecting gan dct anomalies. *Journal of Imaging*, 7(8), 2021.

[13] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014.

[14] Luca Guarnera, Oliver Giudice, and Sebastiano Battiato. Deepfake detection by analyzing convolutional traces. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 2841–2850, 2020.

[15] Javier Hernandez-Ortega, Ruben Tolosana, Julian Fierrez, and Aythami Morales. Deepfakeson-phys: Deepfakes detection based on heart rate estimation. *arXiv preprint arXiv:2010.00400*, 2020.

[16] Dong-Keon Kim, DongHee Kim, and Kwangsu Kim. Facial Manipulation Detection Based on the Color Distribution Analysis in Edge Region. *arXiv:2102.01381 [cs]*, Feb. 2021.

[17] Staffy Kingra, Naveen Aggarwal, and Nirmal Kaur. Lbpnet: Exploiting texture descriptor for deepfake detection. *Forensic Science International: Digital Investigation*, 42-43:301452, 2022.

[18] Lingzhi Li, Jianmin Bao, Hao Yang, Dong Chen, and Fang Wen. FaceShifter: Towards High Fidelity And Occlusion Aware Face Swapping. *arXiv preprint arXiv:1912.13457*, 2019.

[19] Yuezun Li and Siwei Lyu. Exposing DeepFake Videos By Detecting Face Warping Artifacts. *CoRR*, abs/1811.00656, 2018.

[20] Chia-Chen Lin, Ting-Lin Lee, Ya-Fen Chang, Pei-Feng Shiu, and Bohan Zhang. Fragile Watermarking for Tamper Localization and Self-Recovery Based on AMBTC and VQ. *Electronics*, 12(2), Jan. 2023.

[21] Eugene T. Lin, Christine I. Podilchuk, and Edward J. Delp Iii. Detection of image alterations using semifragile watermarks. In *Security and Watermarking of Multimedia Contents II*, volume 3971, pages 152–163. SPIE, May 2000.

[22] Luochen Lv. Smart Watermark to Defend against Deepfake Image Manipulation. In *2021 IEEE 6th International Conference on Computer and Communication Systems (ICCCS)*, pages 380–384, Apr. 2021.

[23] Paarth Neekhara, Shehzeen Hussain, Xinqiao Zhang, Ke Huang, Julian McAuley, and Farinaz Koushanfar. FaceSigns: Semi-Fragile Neural Watermarks for Media Authentication and Countering Deepfakes, Apr. 2022.

[24] Huy Hoang Nguyen, Junichi Yamagishi, and Isao Echizen. Capsule-forensics: Using capsule networks to detect forged images and videos. *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2307–2311, 2018.

[25] Amna Qureshi, David Megías, and Minoru Kuribayashi. Detecting Deepfake Videos using Digital Watermarking. In *2021 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 1786–1793, Dec. 2021. ISSN: 2640-0103.

[26] Andreas Rössler, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, and Matthias Nießner. FaceForensics++: Learning to Detect Manipulated Facial Images. *CoRR*, abs/1901.08971, 2019. _eprint: 1901.08971.

[27] S Shyam Sundar, Maria D Molina, and Eugene Cho. Seeing Is Believing: Is Video Modality More Powerful in Spreading Fake News via Online Messaging Apps? *Journal of Computer-Mediated Communication*, Aug. 2021.

[28] Supasorn Suwajanakorn, Steven M. Seitz, and Ira Kemelmacher-Shlizerman. Synthesizing Obama: Learning Lip Sync from Audio. *ACM Trans. Graph.*, 36(4), July 2017.

[29] Mingxing Tan and Quoc V. Le. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. *arXiv:1905.11946 [cs, stat]*, Sept. 2020. arXiv: 1905.11946.

[30] Shahroz Tariq, Sangyup Lee, and Simon S. Woo. A Convolutional LSTM based Residual Network for Deepfake Video

Detection. *arXiv:2009.07480 [cs]*, Sept. 2020. arXiv: 2009.07480.

[31] Junke Wang, Zuxuan Wu, Jingjing Chen, and Yu-Gang Jiang. M2TR: Multi-modal Multi-scale Transfnrmers for Deepfake Detection. *arXiv:2104.09770 [cs]*, Apr. 2021. arXiv: 2104.09770.

[32] Run Wang, Felix Juefei-Xu, Qing Guo, Yihao Huang, L. Ma, Yang Liu, and Lina Wang. Deeptag: Robust image tagging for deepfake provenance. *arXiv: Cryptography and Security*, 2020.

[33] Run Wang, Felix Juefei-Xu, Meng Luo, Yang Liu, and Lina Wang. FakeTagger: Robust Safeguards against Deep-Fake Dissemination via Provenance Tracking, Sept. 2021. arXiv:2009.09869 [cs].

[34] Chaoning Zhang, Adil Karjauv, Philipp Benz, and In So Kweon. Towards Robust Data Hiding Against (JPEG) Compression: A Pseudo-Differentiable Deep Learning Approach, Dec. 2020. arXiv:2101.00973 [cs, eess].

[35] Yuan Zhao, Bo Liu, Ming Ding, Baoping Liu, Tianqing Zhu, and Xin Yu. Proactive Deepfake Defence via Identity Watermarking. In *2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 4591–4600, Waikoloa, HI, USA, Jan. 2023. IEEE.