



國立台灣科技大學 資訊工程系

碩士學位論文

Facestamp: Self-Reference Proactive Deepfake
Detection using Facial Attribute Deep
Watermarking

研究生：Joshua C. Manzano

學 號：M10915818

指導教授：陳怡伶博士

中華民國一百一十一年一月二十一日

Abstract

Deepfakes are progressively harder to distinguish and present a growing problem to image authenticity in society. Existing studies that focus on deepfake detection rely on artifacts or flaws generated by the deepfake process which may not be present in novel deepfake models. This necessitates a proactive approach that is more robust and generalizable. Recent works on proactive defense rely on deep watermarking, where they embed a Unique Identification (UID) to an image. To verify authenticity, a trusted authority needs to decode its hidden UID and cross-reference it to a centralized dataset containing all existing UIDs. Overall, its reliance on a trusted centralized authority that stores individual UIDs makes it inflexible and impedes its widespread adoption. Moreover, this authentication approach has constrained effectiveness when the number of users is limited. In this paper, we present Facestamp, a deep watermarking model for a self-reference proactive defense against deepfakes. We address this problem by directly embedding facial attributes, instead of a UID, to an image using deep watermarking. Image-derived attributes such as facial attributes verify the legitimacy of the image through the identification of inconsistencies between the decoded attributes and current attributes present in the image. This eliminates the need for a centralized verification process and enables independent verification. In our experiments, we show that Facestamp allows the recovery of facial attributes in the wild and the subsequent verification of the current face to determine the legitimacy of the given image. Facestamp is able to defend against deepfakes across three deepfake models, showing promising performance in two popular datasets and is more robust to common post-processing image operations compared to

existing methods.



Acknowledgements

It would want like to express my deepest gratitude towards my adviser, Professor Yi-Ling Chen. The completion of this research was made possible through her continuous help and guidance. Giving me an opportunity to study in NTUST and encouraging us to join the Multimedia and Visual Computing Laboratory helped me in fully appreciating the field of computer vision. The laboratory helped me a lot in providing an excellent environment for doing research.

I would want to use this opportunity to thank my laboratory seniors Daniel Stanley Tan, John Jethro Virtusio, Jose Mari Ople, Maynard Si, Richmond Manga, Julianne Agatha Tan, and Jilyan Bianca Dy for providing us constant guidance and mentorship. Through our various collaborations and discussions and the different kinds of support they gave, I would have not finished this research. I would also want to thank my batch mates Paolo Ato, Adrienne Francesca Soliven, Ian Benedict Ona, and Tadhg Mearthy for always being there to encourage and for persevering together throughout the various challenges that were presented to us. Finally, I would want to express my appreciation to my friends and family for always providing me support and for being there for me throughout the difficulties I have encountered whilst completing the research.

Contents

Recommendation Letter	i
Approval Letter	ii
Abstract	iii
Acknowledgements	v
Contents	vi
List of Figures	viii
List of Tables	x
1 Introduction	1
2 Related Literature	5
3 Method	9
3.1 Overview	9
3.1.1 Watermarking Model	11
3.1.2 Adaptive Distortion Training	14
3.1.3 Channel Coding	16
3.1.4 Watermarking Training Loss	17
4 Results	18

4.1	Implementation Details	18
4.2	Experiments	20
4.2.1	Ablation	20
4.2.2	Comparison	20
5	Conclusion	27
	References	28



List of Figures

1.1	Comparison between a Facestamp protected no-reference verification pipeline (top panel) and a reference dependent verification pipeline (bottom panel) of prior works.	2
3.1	Facestamp's Architecture <i>Facestamp</i> consists of a channel encoder that takes facial attributes X , then outputs an enlarged message X_{enc} . The watermark encoder takes in the input image and the enlarged message as input and produces a residual. The encoded image is produced by adding the residual to the input image. An attack network is used to make the watermark model robust to unseen distortions.	10
3.2	Example of the Residual Masking process. The mask is derived from the facial regions of the input image, and is applied to the residual produced by the encoder. This encourages the encoder to avoid encoding information in the face area.	12
3.3	Illustration of channel coding. Given an input message X , the channel encoder produces a redundant message X_{enc} of a longer length. The redundant message X_{enc} is transmitted through a noisy channel and received by the decoder as X_{no} . Finally the decoder recovers the input X from the corrupted message X_{no}	15

4.1	Receiver Operator Characteristic (ROC) curve of all the models when no post processing is applied.	23
4.2	Receiver Operator Characteristic (ROC) curve of all the models when JPEG compression is applied.	24
4.3	Receiver Operator Characteristic (ROC) curve of all the models when color manipulation is applied.	25
4.4	Receiver Operator Characteristic (ROC) curve of all the models when gaussian blurring is applied.	26




List of Tables

4.1	Ablation study on components. The models were tested on non post-processed images to analyze the performance increase of each component.	19
4.2	Comparison of AUC and Best FPR on Common Post-Processing Methods. The best scores are shown in bold.	22



Chapter 1 Introduction

Deepfakes are synthesized images in which a human face is replaced by another identity. The technology used in creating deepfakes is rapidly advancing, as humans increasingly have a hard time distinguishing them from unaltered images [1]. While deepfakes have a beneficial role in the scenarios of creating new characters or decorating existing ones with vivid facial expressions, it gradually becomes more known for its unethical applications. Examples of these include deepfakes which swap celebrities into pornographic videos or propagating a realistic fraud video that delivers fake and malicious messages.



To address this expanding problem, researchers explored passive detectors to identify images and videos contaminated by deepfakes [2,3]. These methods have shown significant results in detecting deepfake manipulation. However, these passive detection methods often rely on current limitations of deepfake creation such as visual artifacts [4], image quality [5], and eye blinking [6]. Subsequent research has shown that improvements in deepfake generation eliminate these limitations, thus rendering some of these methods obsolete [1].

A proactive approach to deepfake defense would have less reliance on the aforementioned limitations. Early research on proactive methods include adding adversarial perturbations to the source image such that the resulting deepfake output is severely damaged [7,8]. This approach showed exciting results but successive research shows that adding of adversarial perturbations can be defeated by mask-guided detection and reconstruction [9].

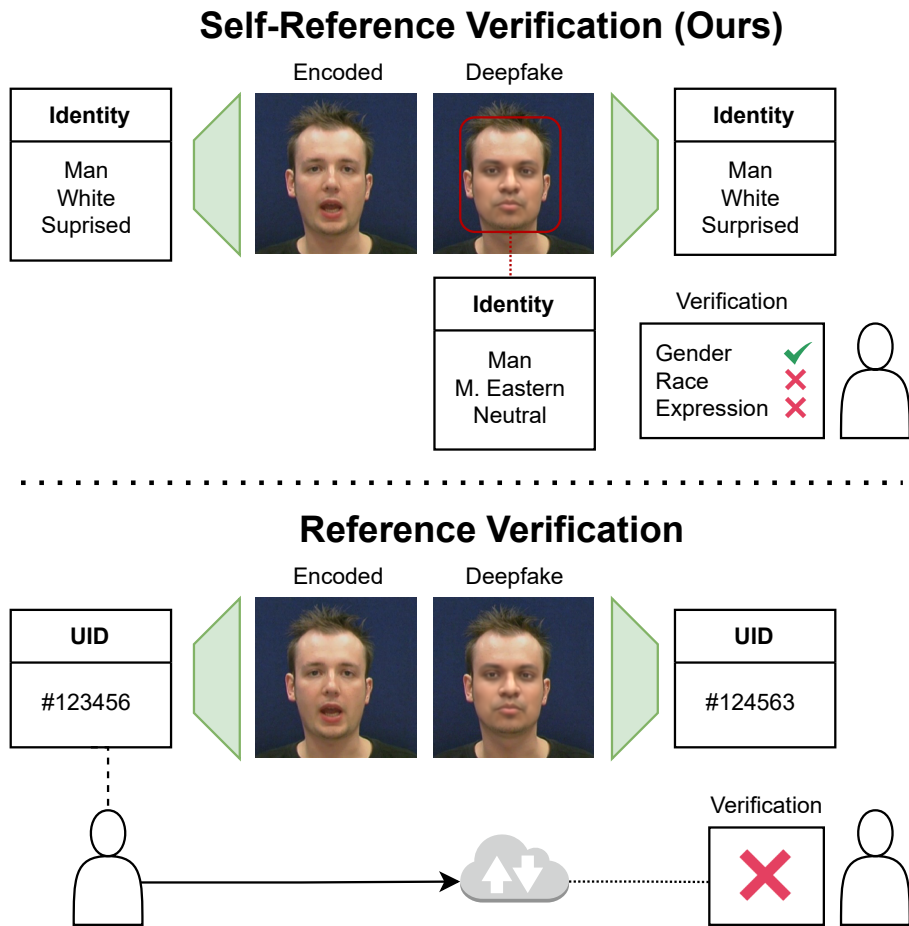


Figure 1.1: Comparison between a Facestamp protected no-reference verification pipeline (top panel) and a reference dependent verification pipeline (bottom panel) of prior works.

Prior works such as FakeTagger [10] and FaceGuard [11] show significant progress on proactive defense that relies on watermarking a unique identification number (UID) and needing this reference UID in future analysis. The necessity of this reference UID makes these models inflexible when it comes to widespread adoption as this requires that majority of the users to follow the same authentication scheme. For a proactive deepfake defense solution to be adopted and used effectively, the verification of independent parties is necessary as it reduces reliance on centralized authorities to micromanage the UIDs that will be used for verification. Removing this necessary external reference makes it easier for any user to easily identify whether an image was manipulated or not without needing to acquire the reference UID. Besides, having to share your UID to third-parties for verification presents another security flaw that can be avoided by adopting a self-reference approach to deepfake defense.

In this paper, we propose a novel idea of proactively defending against deepfake manipulation through deep watermarking of image-derived facial attributes into an image to be protected. This allows the image to be verified by decoding the attributes and spotting inconsistencies between the decoded attributes and the current attributes identified from the image, effectively making it self-referenced verification. We show an example of the differences between previous approaches and our proposed approach in Figure 1.1. Previous state-of-the-art watermarking models are unsuitable to the task of using watermarked facial attributes to determine the authenticity of a protected image. We show that these models show inaccuracies caused by incorrect decoding and interference with facial features during verification. To address these limitations, we employ residual masking, adaptive

distortion training, and channel coding to improve decoding accuracy and preserve facial features.

We evaluate our approach on two popular datasets, namely, CelebA-HQ [12] and VidTIMIT [13]. We show that our model can verify fake images across three identity-swap deepfake models: Faceswap, SimSwap [14], and FSGAN [15].

In summary, the main contributions of this paper are:

- To the best of our knowledge, we are the first to propose a self-reference method of using a deep watermarking model to proactively defend against deepfake manipulation.
- Our proposed model is able to recover the embedded facial attributes accurately and subsequently verify with a reasonable degree of accuracy.
- To prove the contribution of our method, we performed various experiments on multiple datasets, including an ablation study and comparisons with state-of-the-art.

Chapter 2 Related Literature

Deepfakes have grown in popularity in recent years, and researchers have proposed multiple approaches to create more realistic deepfakes and ways to subsequently combat them.

Deepfake Creation Deepfakes have become popular due to the quality of tampered videos and also the easy-to-use ability of their applications to a wide range of users with various computer skills from professional to novice. These applications are mostly developed based on deep learning techniques. Deep learning is well known for its capability of representing complex and high-dimensional data. One variant of the deep networks with that capability is deep autoencoders, which have been widely applied for dimensionality reduction and image compression [16]. The first attempt of deepfake creation was through the usage of an autoencoder-decoder pairing structure. In that method, the autoencoder extracts latent features of face images and the decoder is used to reconstruct the face images. To swap faces between source images and target images, there is a need of two encoder-decoder pairs where each pair is used to train on an image set, and the parameters of the encoder are shared between two network pairs. In other words, two pairs have the same encoder network [16]. This strategy enables the common encoder to find and learn the similarity between two sets of face images, which is relatively trivial because faces normally have similar features such as eyes, nose, mouth positions.

Deepfake Detection Methods for detecting deepfakes have been proposed as soon as the threat of deepfakes have materialized. Early attempts were based on manually determined features obtained from artifacts and in-

consistencies of the deepfake generation process. Recent methods, on the other hand, applied deep learning to automatically extract salient and discriminative features to detect deepfakes. Deepfake detection is normally deemed a binary classification problem where classifiers are used to classify between authentic videos and tampered ones. This kind of method requires a large database of real and fake videos to train classification models. The number of fake videos is increasingly available, but it is still limited in terms of setting a benchmark for validating various detection methods. To address this issue, a notable deepfake data set [17] was produced consisting of 620 videos based on the GAN model using the open source code Faceswap-GAN. While state-of-the-art deepfake detection can achieve impressive results, recent studies on deepfake detection vulnerability show that they can be vulnerable to shallow reconstruction [18], deliberate noise [19], and adversarial perturbations [20].

Deepfake Prevention Research on various ways on disrupting deepfake generation itself are still in its infancy, as efforts to combat such deepfake systems have mostly focused on detection, rather than prevention [16]. Previous studies involved the usage of adversarial examples against face detectors so that no valid faces can be detected, which may defend against Deepfake attacks [21]. However, faces can still be manually extracted to gather data for Deepfake models. FakeTagger [10] proposes a potential solution through image tagging, in which a hidden message could be embedded in an image which could then be retrieved and identified after a deepfake has been generated. Although robust and successful, its effectiveness is reliant upon the widespread adoption on multiple platforms. Recent studies show a new approach in preventing deepfakes —implementing adversarial

attacks on the malicious deepfake models. An adversarial attack on a given deepfake model involves applying minute changes to a given input that are imperceptible by a human, but should the model be applied to the modified input, its output would be visually disrupted and thus more detectable by detection algorithms. Studies by [8], [7], and [22] successfully apply these to image-translation-based deepfake algorithms in both white-box and black-box settings, while [23] and [21] show potential robust adversarial attacks on face-to-face translation deepfake algorithms. Recent research shows that a poisoning attack could be a straightforward but very complex way of disrupting deepfakes [7,21]. Although initial studies on disrupting deepfakes seem promising, research in this particular area of deepfake defense has been minimal. Notably, the aforementioned studies placed less emphasis on attacking the deepfakes in such a way that would make them easily detectable by humans, and no human perception studies were made to determine their effectiveness. Recent research shows that there is much work needed before a reliable way to protect images from malicious deep models [8]. It is suggested that the investigation on stronger attack methods which are not necessarily norm bounded should be explored, or potentially localized in patch [8].

Watermarking A form of steganography that has long been considered as a potential way to link a physical image to an Internet resource [2]. Early work in the area defined a set of desirable goals for robust watermarking, including invisibility and robustness to image manipulations [7]. Later research demonstrated the significant robustness benefits of encoding the watermark in the log-polar frequency domain [27, 22, 25, 20]. Similar methods have been optimized for use as interactive mobile phone

applications [13, 17, 15]. Additional work focuses on carefully modeling the printer-camera transform [22, 10] or display camera transform [17, 15, 18] for better information transfer. Some approaches to display-camera communication take advantage of the unique properties of this hardware combination such as polarization [18], rolling shutter artifacts [26], or high frame rate [12]. A related line of work in image forensics explores whether it is possible to use a CNN to detect when an image has been re-imaged [16]. In contrast to the hand-designed pipelines used in previous work on watermarking, our method automatically learns how to hide and transmit data in a way that is robust to many different combinations of printers/displays, cameras, lighting, and viewpoints. We provide a framework for training this system and a rigorous evaluation of its capabilities, demonstrating that it works in many real world scenarios and using ablations to show the relative importance of our training perturbations.

Chapter 3 Method

3.1 Overview

Facestamp is a self-reference proactive deepfake detection model with a two-step verification framework. First, we protect a portrait by embedding its facial attributes within the image pixels as a future reference of its original facial features. Second, to verify if a protected image has been tampered with, we decode the embedded facial attributes. If the image has been tampered, then the decoded facial attributes will not match the current facial features found in the image.

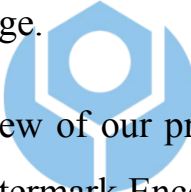


Figure 3.1 gives an overview of our proposed architecture. We first highlight two components: Watermark Encoder F_{enc} , and Watermark Decoder F_{dec} . Watermark Encoder F_{enc} is responsible for encoding the portrait image with its facial attributes. We guide the encoder, using a mask, such that the attributes are encoded outside the face region to minimize interference during the verification process. The input mask is derived from the facial regions of the portrait image, in which the facial regions will be avoided. This design allows us to counter identity-swap deepfakes, which primarily targets the facial region, which leaves our encoded attributes intact. Furthermore, encoding outside the facial region is critical for preserving the visual aesthetic of the image. The Watermark Decoder is then responsible for deciphering the hidden facial attributes in the input image.

Additionally, we introduce two key modules to improve encoding and decoding robustness:

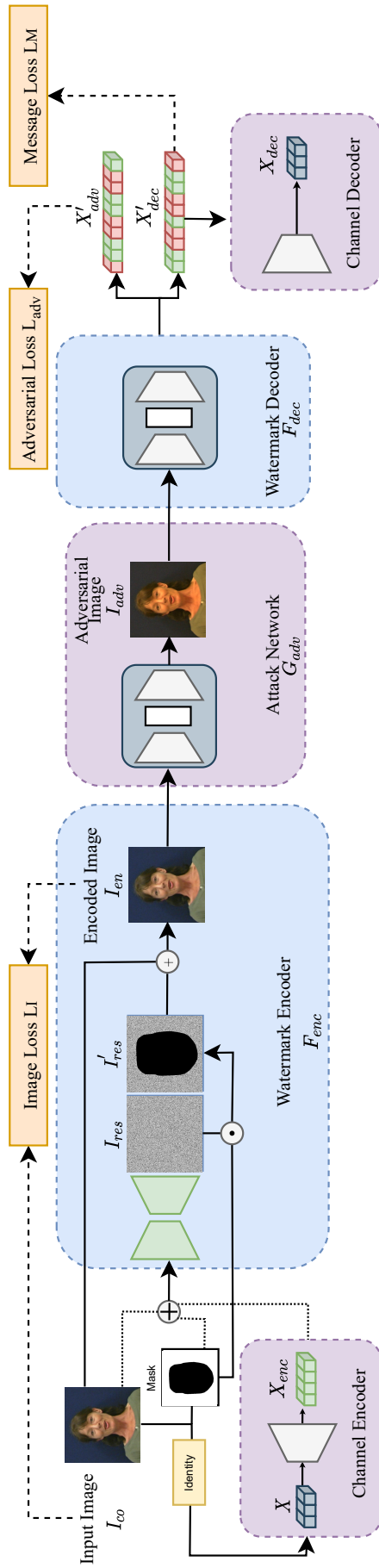


Figure 3.1: **Facestamp's Architecture** *Facestamp* consists of a channel encoder that takes facial attributes X , then outputs an enlarged message X_{enc} . The watermark encoder takes in the input image and the enlarged message as input and produces a residual. The encoded image is produced by adding the residual to the input image. An attack network is used to make the watermark model robust to unseen distortions.

- We utilize an attack network G_{adv} for the adaptive distortion training to make the watermarking model robust against novel distortions.
- We adapt a simple two-layer convolution neural network for channel coding, we utilize learning based solutions seen in recent works. [10, 24].

3.1.1 Watermarking Model

$$L_I = \lambda_R L_R + \lambda_P L_P \quad (3.1)$$

$$L_M = \lambda_M ||X'_{dec} - X'||^2 \quad (3.2)$$

The watermarking model to be used should be able to accurately recover the encoded facial attributes while preserving the facial features of the corresponding encoded image. To facilitate this, we jointly train the encoder and decoder with a series of losses.

Encoder

We train the encoder to have minimal perceptual differences on the encoded image from the input image. To enforce this, we use an LPIPS perceptual loss L_P and a L_2 residual regularization L_R to minimize the magnitude of the residuals. This is expressed in Equation 3.1.

We use a U-Net [25] style architecture that receives a nine channel 400×400 pixel input. The input includes a three channel RGB input image I_{co} , a three channel message input, and a three channel input mask.

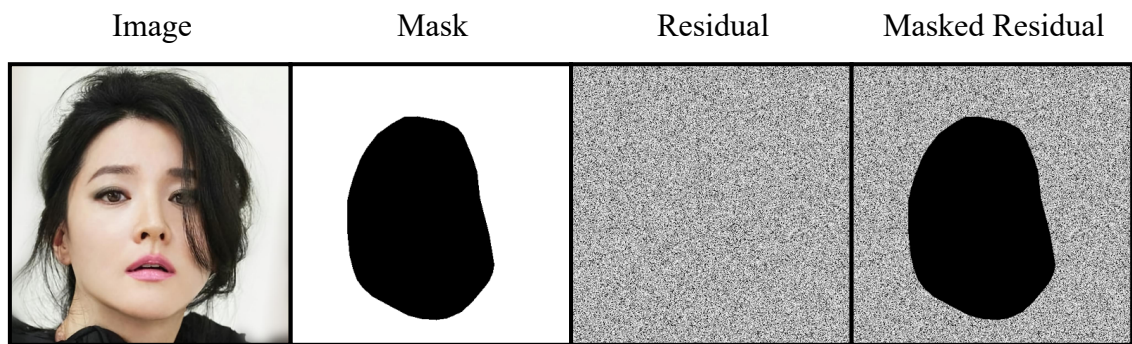


Figure 3.2: Example of the Residual Masking process. The mask is derived from the facial regions of the input image, and is applied to the residual produced by the encoder. This encourages the encoder to avoid encoding information in the face area.

The input message is received from the channel encoder as a 100 bit binary string, processed through a fully connected layer to form a $50 \times 50 \times 3$ tensor, then unsampled to produce a $400 \times 400 \times 3$ tensor. This preprocessing done to the message aids in convergence.

Masking

The residuals produced by the encoder F_{dec} are masked with an input mask which contains information of where the face is located. This results with a residual which does not put any information in the facial regions of the picture. This assists with the verification process, it allows the facial analysis to be as unaltered as possible. We provide this mask to the encoder to allow the model to adapt to the masking operation done on the output residuals. A visualization of the masking process can be found in Figure 3.2.

Decoder

We train the decoder F_{dec} to recover the hidden message X_{dec} from the encoded image I_{en} , even if the encoded image has been manipulated and altered by the attack network G_{adv} . To train the model to recover the hidden message, we use a message loss L_M . This is expressed in Equation 3.2. An encoded image (which may have been altered) is fed through a series of convolutional and dense layers and a sigmoid to produce a final output with the same length as the message.

3.1.2 Adaptive Distortion Training

$$L_{adv} = \lambda_1^A ||I_{adv} - I_{enc}||^2 - \lambda_2^A ||X'_{adv} - X'||^2 \quad (3.3)$$

Deepfake models and common post-processing image operations produce distortions that will be novel to the watermarking model. We use adaptive distortion training to address this issue. Adaptive distortion training produces a variety of distortions that adapt along with the training of the watermarking model, changing according to the weakest point of the current model. We use a two-layer CNN to produce distortions:

$$G_{adv}(I) = \text{Conv}_3 \circ \text{LeakyReLU} \circ \text{Conv}_{16}(I) \quad (3.4)$$

Using a two-layer CNN as an attack network G_{adv} to generate distortions allows the ability to generate a diverse set of image distortions while minimizing the strength of the distortions to allow the watermarking model to train properly [24].

We train the attack network G_{adv} with an MSE loss (3.3), and the strength of the distortions is controlled through the weights λ_1^A and λ_2^A

Alternatively, StegaStamp [26] uses a static set of differentiable image perturbations that approximate real-world conditions for their distortion training. In our ablation study, we explore between using a static set of differentiable image perturbations and training an two-layer CNN attack network G_{adv} jointly with the watermarking encoder and decoder.

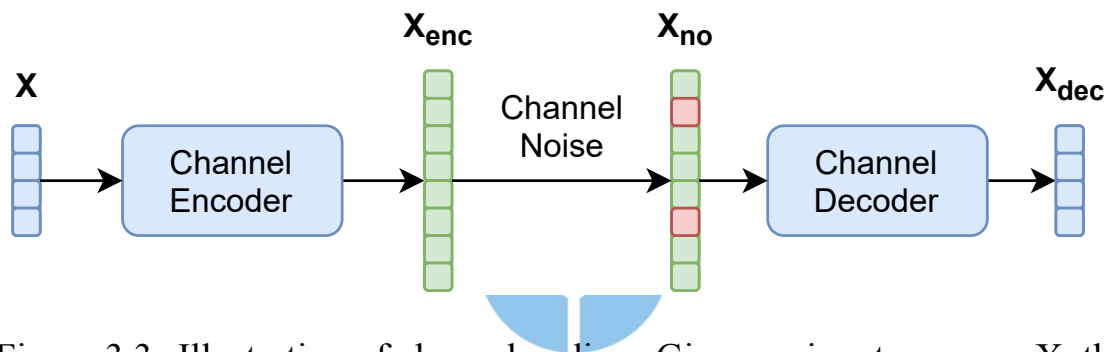


Figure 3.3: Illustration of channel coding. Given an input message X , the channel encoder produces a redundant message X_{enc} of a longer length. The redundant message X_{enc} is transmitted through a noisy channel and received by the decoder as X_{no} . Finally the decoder recovers the input X from the corrupted message X_{no} .

3.1.3 Channel Coding

$$\mathcal{C}_{loss} = \frac{1}{n} \sum_{i=1}^n (X - X_{dec})^2 \quad (3.5)$$

Distortions to encoded image I_{enc} may cause inaccuracies to the decoded attributes. Channel coding provides robustness by providing redundancy to the encoded attributes. We generate a channel code X_{enc} from the input message X , before passing X_{enc} to the watermarking encoder as shown in Figure 3.3. Given a binary message $X \in \{0, 1\}^D$ of length D , a channel encoder produces a redundant message $X_{enc} \in \{0, 1\}^N$ of length N , where $N > D$. This allows us to recover X despite reasonable amounts of channel distortion to X_{enc} . We train this channel coding model using an MSE loss (3.5), where X is the ground truth message and X_{dec} is the recovered message after being passed through channel distortions.

Channel distortions are the errors that occur between X_{enc} and X_{no} . These may include distortions from the watermarking model, deepfake models, and image operations. Due to the unpredictable nature of these distortions, it would be impractical to explicitly model the channel distortions. Instead, we adapt a binary symmetric channel (BSC) to approximate the channel distortion. BSC is a standard channel model which assumes each bit is independently and randomly flipped with probability p . We use NECST [27], a learning based solution for joint source and channel coding to cover a broad range of channel distortion strengths. The model is trained independently from the watermarking model to prevent co-adaption during training, which would limit robustness and generalizability.

3.1.4 Watermarking Training Loss

$$L_W = L_I + L_M + \lambda_W^A ||X'_{adv} - X'||^2 \quad (3.6)$$

Equation 3.6 defines the overall loss for training the encoder and decoders.



Chapter 4 Results

4.1 Implementation Details

For our approach we resize all of the input images to a width and height of 400 pixels. For training, we train all of the Facestamp models with 10 epochs and use an ADAM optimizer with a learning rate of 0.005. The default weighting for λ_R , λ_P , λ_1^A , λ_2^A , λ_W^A is set to 1. We use 25,000 face images from CelebA-HQ [28] to train all of the Facestamp models.

We use Lightface [29] for getting the facial attributes and facial regions from the image, we represent these facial attributes as 14 bits. The input masks to be used with the encoder are derived from the facial regions. We use channel encoding to encode these 14 bits to 100 bits. The channel coding model is not jointly trained with the watermarking encoder and decoder, this prevents the channel coding model from coadapting with the image models during training.

Table 4.1: **Ablation study on components.** The models were tested on non post-processed images to analyze the performance increase of each component.

Components	Combinations			
	✓	✓	✗	✓
Adaptive Distortions	✓	✓	✗	✓
Residual Masking	✓	✓	✓	✗
Channel Coding	✓	✗	✓	✓
Static Distortions	✗	✗	✓	✗
AUC ↑	0.9766	0.9740	0.9484	0.9104
FPR ↓	0.0540	0.0700	0.1040	0.1740

4.2 Experiments

We evaluate our method using two popular datasets, specifically 1,000 images from CelebA-HQ [28] and 1,000 images from VidTIMIT [13]. We compare the Area Under The Curve (AUC), and best False Positive Rate (FPR) of all the models in the task of verifying if an image has been manipulated. We also analyzed the impact and importance of *Facestamp*'s different components through an ablation study. For comparison, we use the pretrained networks of StegaStamp [26] and all three architectures of SteganoGAN [30].

4.2.1 Ablation



To analyze the importance of each of the component, we performed an ablation study. Our method has the following main components: (1) Channel Coding, (2) Residual Masking, and (3) Distortion Training. We experiment with using both an Attack Network and using a static set of distortions that mimic real-world conditions [26]. In Table 4.1, we show that Residual Masking gave the biggest performance difference, while Channel Coding gave marginal increases in performance. We also show that using a static set of distortions decreased performance on non post-processed images.

4.2.2 Comparison

Here we compare against state-of-the-art deep watermarking models, which include StegaStamp [26] and three architectures of SteganoGAN [30]. In

Table 4.2 and Fig. 4.1, we show that *Facestamp* performs the best on images with no post-processing with an AUC of 0.9766 and an FPR of 0.0540. We show the qualitative results of each model in Fig. ??

Post Processing We explore the robustness of *Facestamp* against common post-processing operations such as JPEG compression, gaussian blurring, and color manipulation. We set the compression quality to 90, standard deviation of the gaussian blurring to 1, and the color manipulation follows the same parameters used by StegaStamp [26]. In Fig. 4.2 we show that the performance of *Facestamp* is the best compared to the other models when JPEG compression is applied to the images. This performance difference can also be seen in gaussian blurring Fig. 4.4 and color manipulation 4.3. We show that models which are not explicitly trained against distortions such as SteganoGAN have poor performance when used in scenarios where post-processing is applied.

Table 4.2: Comparison of AUC and Best FPR on Common Post-Processing Methods. The best scores are shown in bold.

Model	AUC \uparrow				FPR \downarrow			
	None	JPEG	Blur	Color	None	JPEG	Blur	Color
Facestamp (Ours)	0.9766	0.7933	0.8831	0.7876	0.0540	0.3420	0.1864	0.2782
StegaStamp	0.6967	0.7157	0.7729	0.7359	0.3500	0.3600	0.2933	0.3300
SteganoGAN (Basic)	0.5388	0.6205	0.5404	0.6119	0.4680	0.4809	0.4512	0.4325
SteganoGAN (Dense)	0.8320	0.6103	0.3549	0.7156	0.2440	0.5800	0.5945	0.3545
SteganoGAN (Residual)	0.8674	0.5525	0.4825	0.6978	0.2020	0.5955	0.3900	0.3636

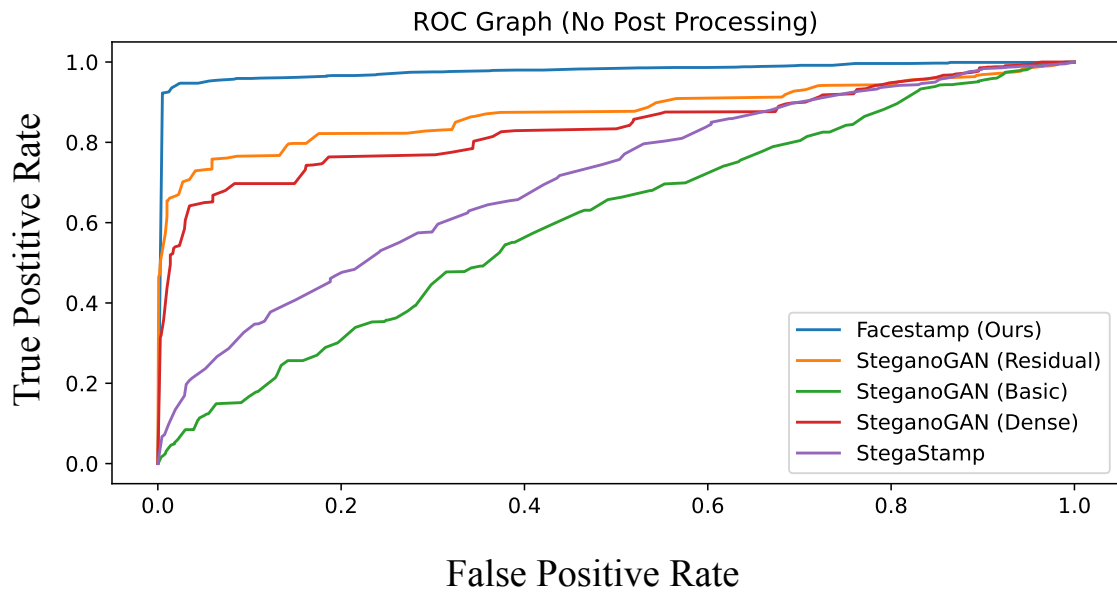


Figure 4.1: Receiver Operator Characteristic (ROC) curve of all the models when no post processing is applied.

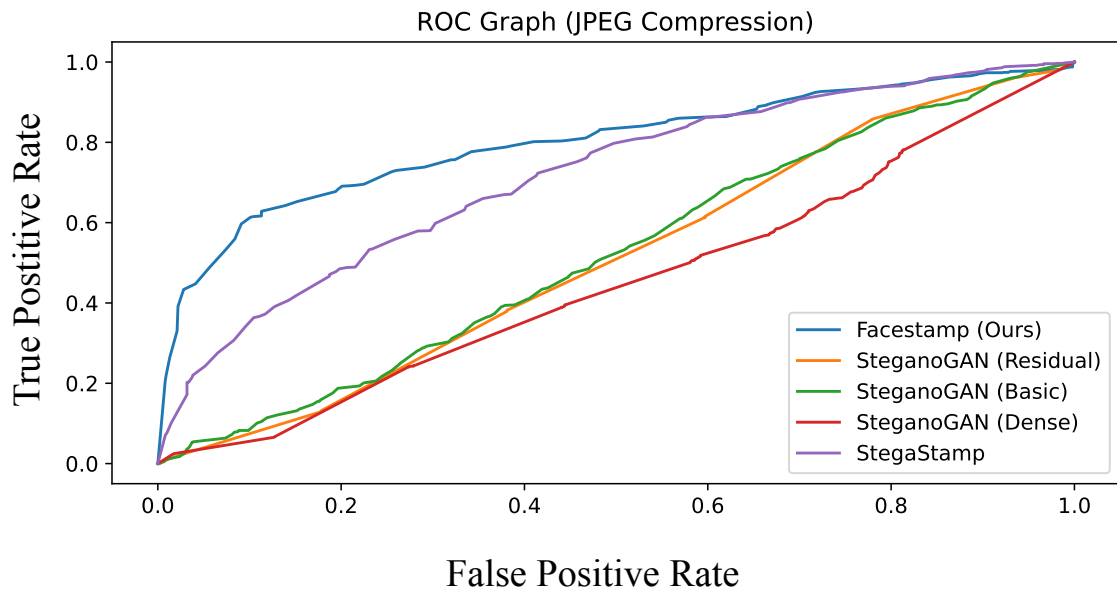


Figure 4.2: Receiver Operator Characteristic (ROC) curve of all the models when JPEG compression is applied.

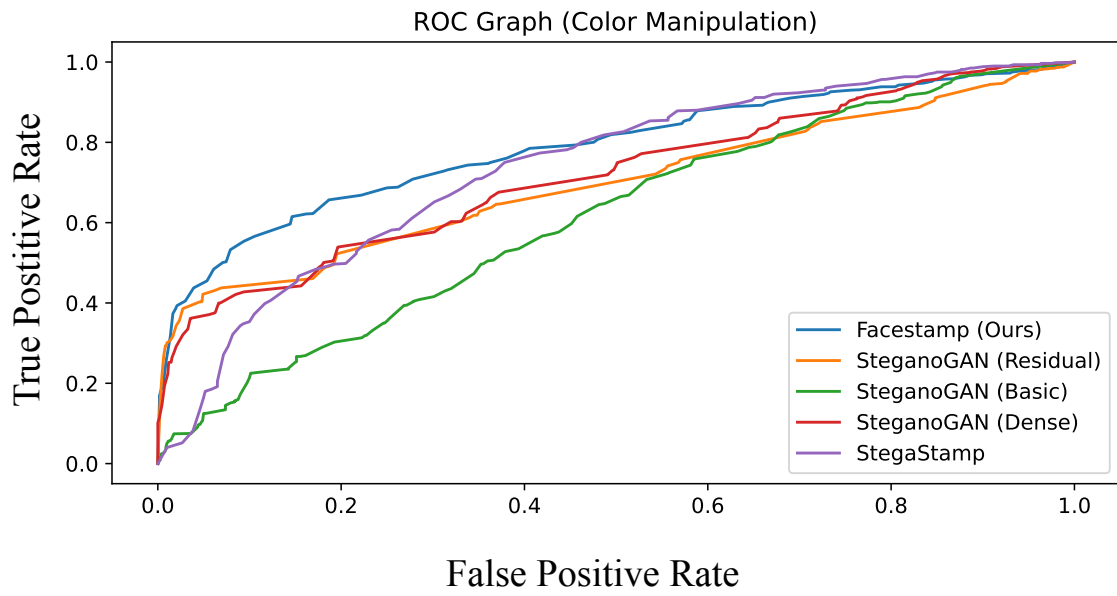


Figure 4.3: Receiver Operator Characteristic (ROC) curve of all the models when color manipulation is applied.

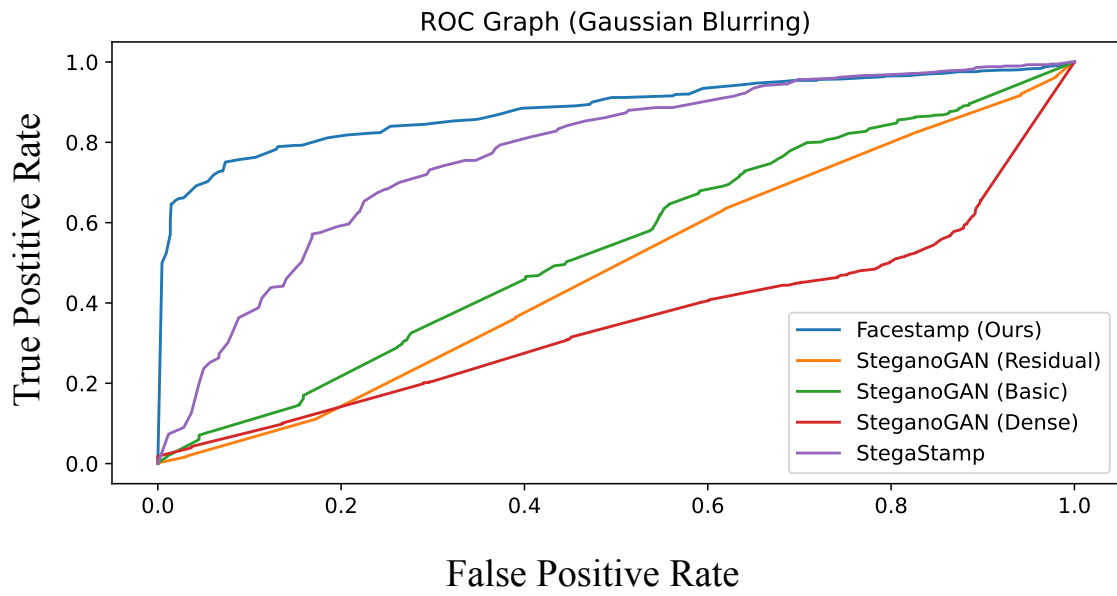


Figure 4.4: Receiver Operator Characteristic (ROC) curve of all the models when gaussian blurring is applied.

Chapter 5 Conclusion

In this paper, we propose *Facestamp*, a self-reference proactive deepfake defense that can accurately verify whether an image has been manipulated. To the best of our knowledge, this is the first work that presents a self-reference proactive approach against deepfake manipulation through deep watermarking of derived facial attributes. Experiments on two datasets and three different deepfake models demonstrate the effectiveness of our method in embedding facial attributes and being able to use these attributes for verification. We achieved an AUC of 0.9766 with an FPR of 0.0540. We also show that *Facestamp* is more robust against post-processing operations compared to previous watermarking models. We also show the effectiveness of each component of *Facestamp* through an ablation study.

References

- [1] R. Tolosana, R. Vera-Rodriguez, J. Fierrez, A. Morales, and J. Ortega-Garcia, “Deepfakes and beyond: A survey of face manipulation and fake detection,” *arXiv preprint arXiv:2001.00179*, 2020.
- [2] C. Rathgeb, A. Botaljov, F. Stockhardt, S. Isadskiy, L. Debiasi, A. Uhl, and C. Busch, “Prnu-based detection of facial retouching,” *IET Biometrics*, vol. 9, no. 4, pp. 154–164, 2020.
- [3] X. Zhang, S. Karaman, and S.-F. Chang, “Detecting and simulating artifacts in gan fake images,” in *2019 IEEE International Workshop on Information Forensics and Security (WIFS)*, pp. 1–6, IEEE, 2019.
- [4] F. Matern, C. Riess, and M. Stamminger, “Exploiting visual artifacts to expose deepfakes and face manipulations,” in *2019 IEEE Winter Applications of Computer Vision Workshops (WACVW)*, pp. 83–92, IEEE, 2019.
- [5] Y. Li and S. Lyu, “Exposing deepfake videos by detecting face warping artifacts,” *arXiv preprint arXiv:1811.00656*, 2018.
- [6] T. Jung, S. Kim, and K. Kim, “Deepvision: deepfakes detection using human eye blinking pattern,” *IEEE Access*, vol. 8, pp. 83144–83154, 2020.
- [7] N. Ruiz, S. A. Bargal, and S. Sclaroff, “Disrupting deepfakes: Adversarial attacks against conditional image translation networks and facial manipulation systems,” in *European Conference on Computer Vision*, pp. 236–251, Springer, 2020.
- [8] C.-Y. Yeh, H.-W. Chen, S.-L. Tsai, and S.-D. Wang, “Disrupting image-translation-based deepfake algorithms with adversarial attacks,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision Workshops*, pp. 53–62, 2020.
- [9] Z. Chen, L. Xie, S. Pang, Y. He, and B. Zhang, “Magdr: Mask-guided detection and reconstruction for defending deepfakes,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9014–9023, 2021.
- [10] R. Wang, F. Juefei-Xu, M. Luo, Y. Liu, and L. Wang, “Faketagger: Robust safeguards against deepfake dissemination via provenance tracking,” in *Proceedings of the 29th ACM International Conference on Multimedia*, pp. 3546–3555, 2021.
- [11] Y. Yang, C. Liang, H. He, X. Cao, and N. Z. Gong, “Faceguard: Proactive deepfake detection,” *arXiv preprint arXiv:2109.05673*, 2021.
- [12] Z. Liu, P. Luo, X. Wang, and X. Tang, “Large-scale celebfaces attributes (celeba) dataset,” *Retrieved August*, vol. 15, no. 2018, p. 11, 2018.
- [13] C. Sanderson and B. C. Lovell, “Multi-region probabilistic histograms for robust and scalable identity inference,” in *International conference on biometrics*, pp. 199–208, Springer, 2009.

- [14] R. Chen, X. Chen, B. Ni, and Y. Ge, “Simswap: An efficient framework for high fidelity face swapping,” in *Proceedings of the 28th ACM International Conference on Multimedia*, pp. 2003–2011, 2020.
- [15] Y. Nirkin, Y. Keller, and T. Hassner, “Fsgan: Subject agnostic face swapping and reenactment,” in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 7184–7193, 2019.
- [16] T. T. Nguyen, C. M. Nguyen, D. T. Nguyen, D. T. Nguyen, and S. Nahavandi, “Deep learning for deepfakes creation and detection,” *arXiv preprint arXiv:1909.11573*, vol. 1, 2019.
- [17] P. Korshunov and S. Marcel, “Vulnerability assessment and detection of deepfake videos,” in *2019 International Conference on Biometrics (ICB)*, pp. 1–6, IEEE, 2019.
- [18] Y. Huang, F. Juefei-Xu, R. Wang, Q. Guo, L. Ma, X. Xie, J. Li, W. Miao, Y. Liu, and G. Pu, “Fake-polisher: Making deepfakes more detection-evasive by shallow reconstruction,” in *Proceedings of the 28th ACM International Conference on Multimedia*, pp. 1217–1226, 2020.
- [19] Y. Huang, F. Juefei-Xu, Q. Guo, X. Xie, L. Ma, W. Miao, Y. Liu, and G. Pu, “Fakeretouch: Evading deepfakes detection via the guidance of deliberate noise,” *arXiv preprint arXiv:2009.09213*, 2020.
- [20] S. Hussain, P. Neekhara, M. Jere, F. Koushanfar, and J. McAuley, “Adversarial deepfakes: Evaluating vulnerability of deepfake detectors to adversarial examples,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 3348–3357, 2020.
- [21] C. Yang, L. Ding, Y. Chen, and H. Li, “Defending against gan-based deepfake attacks via transformation-aware adversarial faces,” *arXiv preprint arXiv:2006.07421*, 2020.
- [22] N. Ruiz, S. A. Bargal, and S. Sclaroff, “Protecting against image translation deepfakes by leaking universal perturbations from black-box neural networks,” *arXiv preprint arXiv:2006.06493*, 2020.
- [23] E. Segalis, “Disrupting deepfakes with an adversarial attack that survives training,” *arXiv preprint arXiv:2006.12247*, 2020.
- [24] X. Luo, R. Zhan, H. Chang, F. Yang, and P. Milanfar, “Distortion agnostic deep watermarking,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13548–13557, 2020.
- [25] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241, Springer, 2015.
- [26] M. Tancik, B. Mildenhall, and R. Ng, “Stegastamp: Invisible hyperlinks in physical photographs,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2117–2126, 2020.
- [27] K. Choi, K. Tatwawadi, T. Weissman, and S. Ermon, “Necst: neural joint source-channel coding,” 2018.

- [28] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu, “Celeb-df (v2): a new dataset for deepfake forensics,” *arXiv preprint arXiv:1909.12962*, 2019.
- [29] S. I. Serengil and A. Ozpinar, “Lightface: A hybrid deep face recognition framework,” in *2020 Innovations in Intelligent Systems and Applications Conference (ASYU)*, pp. 1–5, IEEE, 2020.
- [30] K. A. Zhang, A. Cuesta-Infante, L. Xu, and K. Veeramachaneni, “Steganogan: High capacity image steganography with gans,” *arXiv preprint arXiv:1901.03892*, 2019.

