2022 IEEE 5th International Conference on Multimedia Information Processing and Retrieval (MIPR) | 978-1-6654-9548-6/22/$31.00 ©2022 IEEE | DOI: 10.1109/MIPR54900.2022.00044

Detection-by-Simulation: Exposing DeepFake via Simulating Forgery using Face Reconstruction

Jiaran Zhou, Yuezun Li*
Ocean University of China, Qingdao, China
{zhoujiaran;liyuezun}@ouc.edu.cn

Abstract

This paper describes a new method to expose DeepFakes via simulating forgery using face reconstruction on real samples. Our method is motivated by that the DeepFake model introduces generation artifacts on synthesized faces, which can be simulated by similar CNN-based generative models. To simulate these forgery artifacts, we develop a simple auto-encoder network to reconstruct faces, as the generation process in face reconstruction shares some certain common properties with the generation process in the DeepFake model. Thus we can use the reconstructed faces as negative training samples. Then we develop a CNN network to fully take advantage of the simulation. Specifically, we design two components, an attention guided blending boundary prediction branch to predict blending boundary and a semantic feature enhancement to convey semantic information to deep layers. Then the proposed network is trained using the simulated faces and real faces. Extensive experiments are conducted on FF++ and Celeb-DF with comparison to several state-of-the-arts, which demonstrates the efficacy of our method.

1 Introduction

The prominent advances in Convolutional Neural Networks (CNNs) have greatly promoted the development of face forgery techniques. DeepFake is an emerging face-swapping technique that draws increasing attention recently. DeepFake can synthesize a new face in replacement of the original face in videos while retaining the attributes such as facial expression and orientation. Owing to the sensitivity of human faces, the abuse of Deepfakes can raise malicious threats to society, such as making pornographic videos [15], forging the behavior of public figures [22], cyber fraud [7] and etc.

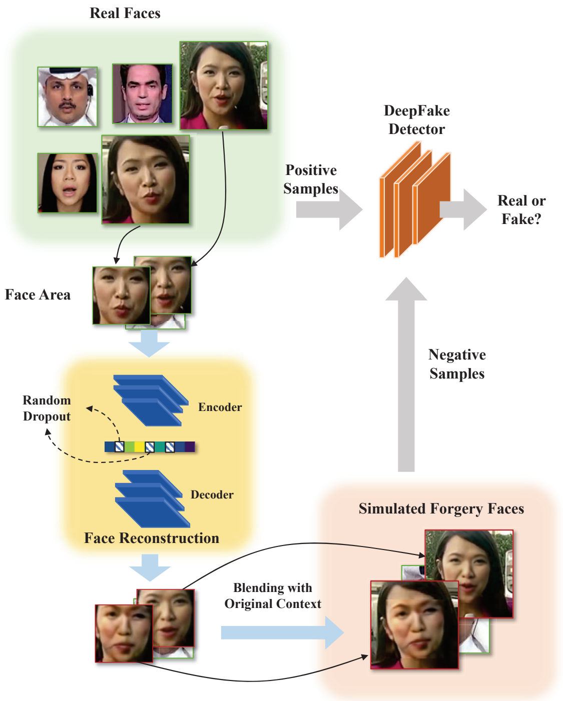

Figure 1. Overview of our method. We simulate the forgery faces using face reconstruction, and then blend the reconstructed face with its original face image as negative training samples. Note that our method can create a forgery face image only using itself. Then we develop a DeepFake detector trained with the positive and negative samples.

To mitigate the threats, many methods have been proposed to expose DeepFake videos [11, 1, 10, 4]. Note that most of them are based on CNNs due to their effectiveness on this task. According to their training settings, we can divide these methods into three categories: *Regular learning*, *Few-shot learning*, and *Zero-shot learning*. Specifically, regular learning represents the methods that are trained and tested on a single dataset [1, 17]. In general, these methods require a large number of training samples and are likely

*Corresponding author

2770-4319/22/$31.00 ©2022 IEEE
DOI 10.1109/MIPR54900.2022.00044

overfitting to the dataset. Despite these methods can achieve excellent and sometimes even perfect performance, they are notably disturbed when the input samples are from different distributions. Therefore, this type of method is difficult to handle new types of DeepFakes, which limits their application in the wild. Despite this weakness can be partially mitigated by fine-tuning the models on new data, it also requires sufficient training data that may not be easily accessed in the real-world setting. To solve the problem, a few methods are proposed to improve the generalization of DeepFake detection, *i.e.*, improving the ability to detect new types of DeepFake. These methods aim to transfer the knowledge from known datasets to new forged faces. Few-shot learning and zero-shot learning are developed for this purpose to improve the generalization. Specifically, few-shot learning only relies on a few labeled samples of new data [6], while zero-shot learning aims to achieve the generalization without seeing new data [23, 14]. Despite zero-shot learning being more applicable than others, it still needs a large number of known DeepFake samples as negative training samples.

In this paper, we describe a new method to expose DeepFakes *in a more strict scenario, where only real samples of a known dataset can be accessed*. Specifically, our method achieves Detection-by-Simulation (D-by-S), which simulates the forgery as negative training samples only using face reconstruction. Our method is based on the observation that the DeepFake faces contain the artifacts introduced by the generation process of CNN. To simulate these forgery artifacts, we develop a simple and effective strategy by reconstructing faces. Specifically, we develop a simple auto-encoder network as *Forgery Simulator*, which inputs a real face and outputs a reconstructed face. The intuition is that the generation process in face reconstruction shares some certain common properties with the generation process in the DeepFake model. Thus the reconstructed faces can contain similar artifacts as in DeepFake faces. Then we can blend this reconstructed face back to its original face image as negative training samples. Benefit from the structure of the auto-encoder, we can increase the diversity of reconstructed faces by manipulating the “code”, which is the feature vector generated from the encoder that represents the full facial attributes. With the simulated DeepFake faces, we can develop a model to learn these artifacts. The overview of our method is shown in Fig. 1.

To fully take advantage of the simulation, we design a network architecture with two core components: *Attention Guided Blending Boundary Prediction* (AGB^{2}P) and *Semantic Feature Enhancement* (SFE). Specifically, our network contains two branches, where one is the main branch for DeepFake detection and another one is AGB^{2}P, which is an auxiliary branch to predict the blending boundary of forged faces. The AGB^{2}P branch communicates with the main branch under the guidance of the proposed attention module, which can positively guide the learning of the main branch. SFE is proposed on the observation that semantic features easily vanish with the convolutions in CNN. As such, we propose SFE to transfer the semantic feature from the shallow layers to the deep layers. Compared to existing methods [10, 24] that require real faces of different subjects to create forged faces, our method simulates the forgery only using a single real face.

Extensive experiments are conducted on several public datasets, *e.g.*, Celeb-DF [13], FF++ [20], which corroborates the effectiveness of our method on within-dataset and cross-dataset scenarios. We also conduct an ablation study on the effect on other manipulations, the effect of different backbones, and various network components.

2 Detection-by-Simulation

Our method achieves Detection-by-Simulation (D-by-S). Specifically, we develop an auto-encoder network to reconstruct faces. Since the reconstructed faces contain the common generation artifacts as in the DeepFake face, we can use them as negative training samples. Then we develop a new network with designed objective functions to detect the simulated artifacts.

2.1 Simulating Forgery using Face Reconstruction

We simulate forged faces only using the reconstruction on every single face. To do so, we develop an auto-encoder called *SimAE* to mimic the generation artifacts in DeepFake.

SimAE. The architecture of SimAE follows the general form of auto-encoder, which consists of an encoder and a decoder. The encoder can transform the input face into a latent feature vector, which contains the informative attributes of the input face. Then the decoder reconstructs the input face based on the feature vector. Since SimAE is a generative model, which can share common properties in DeepFake generation. In our design, the encoder is composed by four convolution layers with a fully connected layer. The number of convolution kernels in these convolution layers are $128, 128, 256$, and 512 respectively. The size of these kernels is 5×5 with a stride of 2×2. The activation after each convolution layer is leaky Relu function. The fully connected layers have output dimensions of 1024. The decoder has five layers, where the first layer is a fully connected layer with reshape, the other four layers are convolution layers with a pixel shuffling layer. The convolution layer has a set of kernels that have two times the original number of channels. The kernels all have a size of 3×3 with a stride of 1.

Note the feature vector is the critical basis for face reconstruction as the neurons of the feature vector represent certain attributes specific to the input face. To increase the diversity of simulation, we randomly drop out the neurons of the feature vector and reconstruct the faces based on these feature vectors. The overview of SimAE is illustrated in Fig. 2.

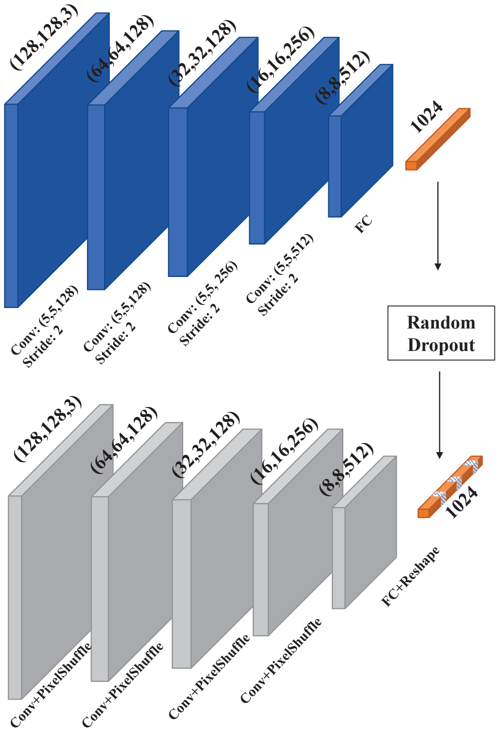

Figure 2. Overview of SimAE. We utilize random dropout on feature vector to increase the diversity of reconstructed faces.

Negative Training Faces Creation. After face reconstruction, we blend the reconstructed faces into their original context to simulate a forgery face. Following works [12, 10], we first design a mask that indicates the blending area and utilize the elastic deformation to improve the diversity of mask. Specifically, we design three masks, which are denoted as *Convex*, *Dilate convex* and *Erode convex*. The Convex denotes a convex hull based on all facial landmarks. The Dilate convex is the mask that covers the forehead and face area. The Erode convex denotes shrinking the convex hull from face contour. To simulate a forgery face, we randomly select one mask for blending.

2.2 Network Architecture

We propose a new network to learn the simulated artifacts introduced by face reconstruction. Our network contains two major components: Attention Guided Blending Boundary Prediction (AGB^{2}P) and Semantic Feature Enhancement (SFE). The overview of network architecture is shown in Fig. 3.

Attention Guided Blending Boundary Prediction. AGB^{2}P is an auxiliary branch that is used to predict the blending boundary. Considering face blending is a common step in DeepFake generation, which can result in a blending boundary around an altered area. Thus exposing the blending boundary can promote distinguishing DeepFakes. AGB^{2}P interacts with the backbone network, with the aim to convey the knowledge for final determination. The communications between AGB^{2}P and the backbone is guided by a proposed attention mechanism. The attention map is obtained based on the previous layer, which contains more semantic information that can promote the learning of boundaries. Specifically, AGB^{2}P contains four convolution blocks that are used to upsample the input features.

Semantic Feature Enhancement. The semantic information of faces can greatly promote the detection of DeepFake. However, the continuous convolution operations gradually transform the semantic feature into high-level space. Thus we propose SFE to improve the semantic representation of deep layers. Specifically, we fuse the feature of shallow layer and deep layer using a bilinear attention pooling [23], which results in a feature vector. Then we concatenate this feature vector with the output from the main branch to form the final feature for DeepFake detection.

2.3 Objectives

Denote $\mathcal{F}: \mathcal{X} \rightarrow \mathcal{Y}$ as the mapping function of DeepFake detector $\mathcal{F}$, where $\mathcal{X} = \{0, 255\}^{H\times W\times 3}$ and $\mathcal{Y} = \{1, ..., C\}$ denote the input and output space respectively. Let $x_i \in \mathcal{X}$ and $y_i \in \mathcal{Y}$ be the input image and corresponding ground truth label, and let $\mathcal{M}_i$ be the predefined mask of blending boundary in x_i. Given input image x_i, the outputs from main branch and AGB^{2}P branch are denoted as $\mathcal{F}_\mathcal{A}(x_i)$ and $\mathcal{F}_\mathcal{B}(x_i)$ respectively, where $\mathcal{F}_\mathcal{A}(x_i) \in [0, 1]^C$ and $\mathcal{F}_\mathcal{B}(x_i) \in [0, 1]^{H\times W}$. Denote the parameters of DeepFake detector $\mathcal{F}$ as θ. Thus the objective function can be formulated as the combination of classification in main branch and blending boundary prediction in AGB^{2}P branch, as

$$\mathcal{L}(x_i, y_i; \theta) = \alpha \cdot \mathcal{L}_\mathcal{A}(x_i, y_i; \theta) + \beta \cdot \mathcal{L}_\mathcal{B}(x_i, y_i; \theta), \quad (1)$$

where $\mathcal{L}_\mathcal{A}$ and $\mathcal{L}_\mathcal{B}$ denote the loss term of classification and blending boundary prediction, α and β are the weighting factors to balance these loss terms. Specifically, we utilize cross-entropy loss for classification as

$$\mathcal{L}_\mathcal{A}(x_i, y_i; \theta) = y_i \log(\mathcal{F}_\mathcal{A}(x_i)) + (1 - y_i) \log(1 - \mathcal{F}_\mathcal{A}(x_i)). \quad (2)$$

The loss of blending boundary prediction is defined as

$$\mathcal{L}_\mathcal{B}(x_i, y_i; \theta) = \mathcal{M}_i \log(\mathcal{F}_\mathcal{B}(x_i)) + (1 - \mathcal{M}_i) \log(1 - \mathcal{F}_\mathcal{B}(x_i)). \quad (3)$$

Minimizing the Eq. (1) can promote the main branch to learn the distinguishable feature of generation artifacts.

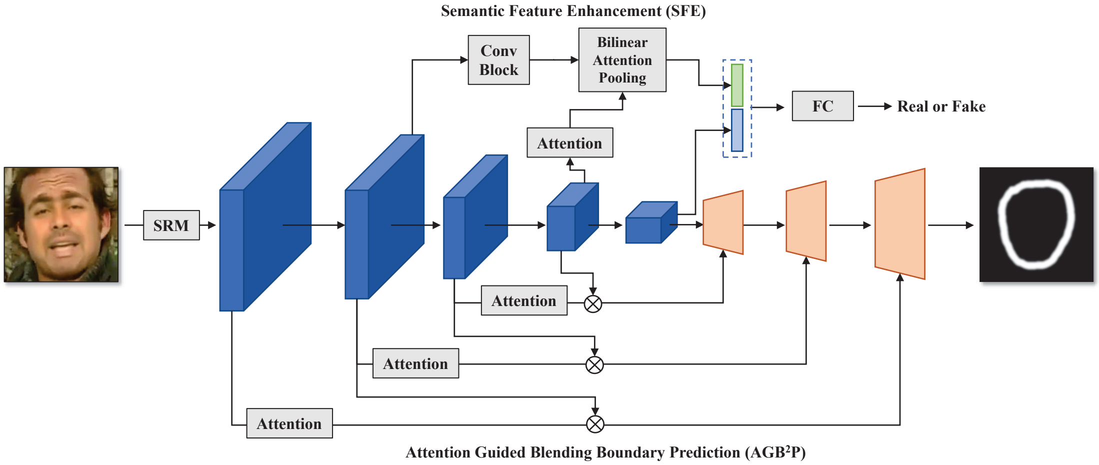
Semantic Feature Enhancement (SFE)
Conv Block
Bilinear Attention Pooling
Attention
FC
Real or Fake
SRM
Attention
Attention
Attention
Attention Guided Blending Boundary Prediction (AGB²P)

Figure 3. Overview of network architecture. The network contains two branches, where one is the main branch integrated with SFE for DeepFake detection and another one is AGB^{2}P for blending boundary prediction.

3 Experiments

3.1 Experimental Settings

Datasets. Our method is evaluated on the FF++ dataset [20] and Celeb-DF dataset [13]. Specifically, the FF++ dataset contains 1000 original videos and same amount of fake videos manipulated by different methods, *e.g.*, DeepFakes, Face2Face, FaceSwap, FaceShifter, and NeuralTextures. FF++ dataset provides different quality levels, *e.g.*, High Quality (HQ) videos with compressed factor c23 and Low Quality (LQ) videos with compressed factor c40, and it is split into 740 videos for training, 140 videos for validation and 140 videos for testing. Since our goal is to expose DeepFake videos, we use DeepFake videos and their corresponding original videos for evaluation. Celeb-DF is a pure DeepFake dataset that contains 5639 high-quality DeepFake videos and 890 real videos, covering 59 identities with different gender, age and etc. This dataset uses 5121 videos for training and 518 videos for testing.

Implementation Details. Our method is implemented by PyTorch [18] with a Nvidia 2080TI GPU on Ubuntu 18.04. We employ XceptionNet [3] as the backbone network. The input image size is 256 × 256. The batch size is 8. The learning rate starts from 0.0001. We utilize Adam optimizer with weight decay as 0.0001 and betas of 0.9 and 0.999. The maximum epoch is 40. For the parameter settings, we set $\alpha = 1$ and $\beta = 10$.

3.2 Results

Comparisons with Other Methods. Table 1 shows the results of our method with comparison to others on LQ and HQ settings of FF++ dataset. These results are measured by the metrics of accuracy (ACC) and area under the curve (AUC). The performance of these methods is reported from [23], and a larger value denotes better performance. The top part of the table shows the methods are trained using both real and fake faces. Since Steg.Features [8] and LD-CNN [5] are proposed for image forgery detection not DeepFake detection, they only exhibit a decent performance compared to others ($<$ 80% on ACC). For other methods, we can observe that they achieves more than 90% on ACC and AUC at HQ setting, which demonstrates their effectiveness on detecting high-quality DeepFakes. Their performance drops notably on LQ setting on both ACC and AUC since the low quality increase the difficulty of detection. The bottom part shows the methods that trained only using real faces. FWA [12] uses a inter-collected dataset and Face X-

Table 1. The performance comparison with different methods on Low-Quality (LQ) and High-Quality (HQ) settings of FF++ dataset.

Methods	LQ		HQ	
	ACC	AUC	ACC	AUC
Steg.Features [8]	56.0	-	71.0	-
LD-CNN [5]	58.7	-	78.5	-
MesoNet [1]	70.5	-	83.1	-
Xception [3]	86.9	89.3	95.7	96.3
Xception-ELA [9]	80.0	82.9	93.9	94.8
Xception-PAFilters [2]	87.2	90.2	-	-
F^3-Net [19]	90.4	93.3	97.5	98.1
Two Branch [16]	-	86.6	-	98.7
EfficientNet-B4 [21]	86.7	88.2	96.6	99.2
MA-Xception [23]	87.0	87.3	96.4	99.0
MA-Efficient-B4 [23]	89.0	90.4	97.6	99.3
GFFD [14]	-	-	-	99.3
FWA [12]	-	-	-	80.1
Face X-ray [10]	-	61.6	-	87.4
D-by-S	**74.0**	**90.4**	**89.3**	**96.1**

ray [10] is trained on FF++. Compared to FWA, Face X-ray performs much better with 7% increase on AUC. However, they have a large performance gap ($\sim 10\%$) with the methods trained using both real and fake faces. In contrast, despite our method only replies on the real faces of FF++, we can achieve competitive performance, which are 96.1% on AUC and 89.3% on ACC at HQ setting and 90.4% on AUC and 74.0% on ACC at LQ setting. Compared to the top performed methods MA-Efficient-B4 [23] and GFFD [14], our method only has slight performance drop ($\sim 3\%$) at HQ setting and still on the top tier at LQ setting. These results demonstrate the efficacy of our method on DeepFake detection.

Generalization on Different Datasets. Generalization is important to the application of DeepFake detection in the wild. Thus we validate the performance of our method at cross-dataset setting, *i.e.*, the model is trained using one dataset, but tested using another one. In our experiment, we train our method on the real samples of FF++ data and employ Celeb-DF for testing. Table 2 shows the performance of different methods at cross-dataset setting. For fair comparison, our method is validated on two variants of Celeb-DF, which are the first version (CDF-v1) and second version (CDF-v2). Note the first four methods are trained using the training set including real and fake images. However, their performance is notably degraded ($< 80\%$). In contrast, our method outperforms these methods at both CDF-v1 and CDF-v2, which corroborates the generalization of our method on different datasets.

Table 2. Generalization performance on Celeb-DF.

Training	Methods	Testing	
		CDF-v1	CDF-v2
FF++	Xception	59.4	65.3
FF++	Face X-ray	74.2	-
FF++	MA-Efficient-B4	-	67.4
FF++	GFFD	79.4	-
FF++-Real	**D-by-S**	**83.4**	**84.6**

3.3 Ablation Study

Performance on Other Manipulations. Besides DeepFake (DF), we also investigate our performance on the other manipulations. Specifically, we test our method on Face2Face (F2F), FaceSwap (FSW), FaceShifter (FSH), NeuralTextures (NT) respectively. Table 3 illustrates the performance of our method on various manipulations. Since the different manipulation methods have different processes, our method shows conspicuous performance discrepancies. From the table we can see that our method achieves the best performance on DF, as it is designed to simulate the generation process in DF. However, our method can not handle the detection of Face2Face, which only has $\sim 70\%$ on AUC. It is because Face2Face is not a DNN-based method, which is purely based on the 3D related transformation. Thus the fake area is totally different from the one in DeepFake face. Moreover, our method has a favorable performance on FaceSwap, which can achieve 93.5% at HQ and 77.3% at LQ. It is probably due to the FaceSwap sharing a similar generation process with DeepFake. For Faceshifter and NeuralTexture, our method performs slightly better than Face2Face, as they are DNN-based methods that share a little common proprieties with DF to some extent.

Table 3. Performance on Other Manipulations.

Manipulations	LQ	HQ
	AUC	AUC
DF	90.4	96.1
F2F	67.0	67.2
FSW	77.3	93.5
F2H	79.7	72.6
NT	67.6	77.8

Different Backbone Networks. This part studies the effect of our method using different backbone networks. We select three backbones, which are ResNet18, ResNet34 and XceptionNet respectively. For Resnet18 and Resnet34, we use the feature map from four residual blocks for the AGB^{2}P. Table 4 shows the performance of our method using these backbones. We can observe that XceptionNet outperforms ResNet18 and ResNet34 by a large margin ($\sim 10\%$), which indicates this task is highly related with capacity of the backbone network.

Table 4. The performance of our method using different backbone networks.

Backbones	LQ	HQ
	AUC	AUC
ResNet18	68.8	80.1
ResNet34	71.4	78.0
XceptionNet	90.4	96.1

The Effect of AGB^{2}P and SFE This part studies the effect of the components of AGB^{2}P and SFE. Table 5 exhibits the performance of different settings. "None" denotes neither AGB^{2}P nor SFE is used. "All" denotes both of these components are used. The results reveal that our method is significantly boosted by these two components by around 5% at LQ and 10% at HQ. Moreover, the performance is notably reduced without using AGB^{2}P compared to SFE, which represents AGB^{2}P plays a more important role to the performance.

Table 5. The Effect of AGB^{2}P and SFE.

Settings	LQ	HQ
	AUC	AUC
None	85.5	88.7
Without AGB^{2}P	86.2	90.3
Without SFE	87.2	95.3
All	90.4	96.1

4 Conclusion

In this paper, we describe a method based on a new perspective, Detection-by-Simulation (D-by-S), to expose DeepFakes via simulating forgery using face reconstruction. Specifically, we develop a simple auto-encoder network for face reconstruction. Then we can simulate a forgery face only using a single real image. To learn the simulated forgery, we develop a CNN network with two components, an attention guided blending boundary prediction and semantic feature enhancement to fully exploit the simulation. Extensive experiments are conducted on many datasets with several state-of-the-arts, which corroborates the efficacy of our method.

Acknowledgments

Jiaran Zhou is supported by NSFC under grant No.62102380, NSF of Shandong province under grant No.ZR2021QF095, and China Postdoc Science Foundation under grant No.2021M693022. Yuezun Li is supported by China Postdoc Science Foundation under grant No.2021TQ0314 and No.2021M703036, and the Fundamental Research Funds for the Central Universities.

References

[1] D. Afchar, V. Nozick, J. Yamagishi, and I. Echizen. Mesonet: a compact facial video forgery detection network. In *WIFS*, 2018. 1, 4

[2] M. Chen, V. Sedighi, M. Boroumand, and J. Fridrich. Jpeg-phase-aware convolutional neural network for steganalysis of jpeg images. In *IHMMSec*, 2017. 4

[3] F. Chollet. Xception: Deep learning with depthwise separable convolutions. In *CVPR*, 2017. 4

[4] U. A. Ciftci, I. Demir, and L. Yin. Fakecatcher: Detection of synthetic portrait videos using biological signals. *TPAMI*, 2020. 1

[5] D. Cozzolino, G. Poggi, and L. Verdoliva. Recasting residual-based local descriptors as convolutional neural networks: an application to image forgery detection. In *IHMM-Sec*, 2017. 4

[6] D. Cozzolino, J. Thies, A. Rössler, C. Riess, M. Nießner, and L. Verdoliva. Forensictransfer: Weakly-supervised domain adaptation for forgery detection. *arXiv:1812.02510*, 2018. 2

[7] P. Fraga-Lamas and T. M. Fernandez-Carames. Fake news, disinformation, and deepfakes: Leveraging distributed ledger technologies and blockchain to combat digital deception and counterfeit reality. *IT Professional*, 22(2):53–59, 2020. 1

[8] J. Fridrich and J. Kodovsky. Rich models for steganalysis of digital images. *TIFS*, 2012. 4

[9] T. S. Gunawan, S. A. M. Hanafiah, M. Kartiwi, N. Ismail, N. F. Za'bah, and A. N. Nordin. Development of photo forensics algorithm by detecting photoshop manipulation using error level analysis. *Indonesian Journal of Electrical Engineering and Computer Science*, 7(1):131–137, 2017. 4

[10] L. Li, J. Bao, T. Zhang, H. Yang, D. Chen, F. Wen, and B. Guo. Face x-ray for more general face forgery detection. In *CVPR*, 2020. 1, 2, 3, 4, 5

[11] Y. Li and S. Lyu. Exposing deepfake videos by detecting face warping artifacts. *arXiv:1811.00656*, 2018. 1

[12] Y. Li and S. Lyu. Exposing deepfake videos by detecting face warping artifacts. In *CVPRW*, 2019. 3, 4

[13] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu. Celeb-df: A large-scale challenging dataset for deepfake forensics. In *CVPR*, 2020. 2, 4

[14] Y. Luo, Y. Zhang, J. Yan, and W. Liu. Generalizing face forgery detection with high-frequency features. *arXiv preprint arXiv:2103.12376*, 2021. 2, 4, 5

[15] S. Maddocks. 'a deepfake porn plot intended to silence me': exploring continuities between pornographic and 'political'deep fakes. *Porn Studies*, 7(4):415–423, 2020. 1

[16] I. Masi, A. Killekar, R. M. Mascarenhas, S. P. Gurudatt, and W. AbdAlmageed. Two-branch recurrent network for isolating deepfakes in videos. In *ECCV*, 2020. 4

[17] H. H. Nguyen, J. Yamagishi, and I. Echizen. Capsule-forensics: Using capsule networks to detect forged images and videos. In *ICASSP*, 2019. 1

[18] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32:8026–8037, 2019. 4

[19] Y. Qian, G. Yin, L. Sheng, Z. Chen, and J. Shao. Thinking in frequency: Face forgery detection by mining frequency-aware clues. In *ECCV*, 2020. 4

[20] A. Rossler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner. Faceforensics++: Learning to detect manipulated facial images. In *ICCV*, 2019. 2, 4

[21] M. Tan and Q. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *ICML*, 2019. 4

[22] M. Westerlund. The emergence of deepfake technology: A review. *Technology Innovation Management Review*, 9(11), 2019. 1

[23] H. Zhao, W. Zhou, D. Chen, T. Wei, W. Zhang, and N. Yu. Multi-attentional deepfake detection. *arXiv preprint arXiv:2103.02406*, 2021. 2, 3, 4, 5

[24] T. Zhao, X. Xu, M. Xu, H. Ding, Y. Xiong, and W. Xia. Learning self-consistency for deepfake detection. In *ICCV*, 2021. 2