Exposing DeepFake Videos By Detecting Face Warping Artifacts

Yuezun Li, Siwei Lyu
Computer Science Department
University at Albany, State University of New York, USA

Abstract

In this work, we describe a new deep learning based method that can effectively distinguish AI-generated fake videos (referred to as DeepFake *videos hereafter) from real videos. Our method is based on the observations that current DeepFake algorithm can only generate images of limited resolutions, which need to be further warped to match the original faces in the source video. Such transforms leave distinctive artifacts in the resulting DeepFake videos, and we show that they can be effectively captured by convolutional neural networks (CNNs). Compared to previous methods which use a large amount of real and DeepFake generated images to train CNN classifier, our method does not need DeepFake generated images as negative training examples since we target the artifacts in affine face warping as the distinctive feature to distinguish real and fake images. The advantages of our method are two-fold: (1) Such artifacts can be simulated directly using simple image processing operations on a image to make it as negative example. Since training a DeepFake model to generate negative examples is time-consuming and resource-demanding, our method saves a plenty of time and resources in training data collection; (2) Since such artifacts are general existed in DeepFake videos from different sources, our method is more robust compared to others. Our method is evaluated on two sets of DeepFake video datasets for its effectiveness in practice.*

1. Introduction

The increasing sophistication of mobile camera technology and the ever-growing reach of social media and media sharing portals have made the creation and propagation of digital videos more convenient than ever before. Until recently, the number of fake videos and their degrees of realism have been limited by the lack of sophisticated editing tools, the high demand on domain expertise, and the complex and time-consuming process involved. However, the time of fabrication and manipulation of videos has decreased significantly in recent years, thanks to the accessibility to large-volume training data and high-throughput computing power, but more to the growth of machine learning and computer vision techniques that eliminate the need for manual editing steps.

In particular, a new vein of AI-based fake video generation methods known as *DeepFake* has attracted a lot of attention recently. It takes as input a video of a specific individual ('target'), and outputs another video with the target's faces replaced with those of another individual ('source'). The backbone of DeepFake are deep neural networks trained on face images to automatically map the facial expressions of the source to the target. With proper post-processing, the resulting videos can achieve a high level of realism.

In this paper, we describe a new deep learning based method that can effectively distinguish DeepFake videos from the real ones. Our method is based on a property of the DeepFake videos: due to limitation of computation resources and production time, the DeepFake algorithm can only synthesize face images of a fixed size, and they must undergo an affine warping to match the configuration of the source's face. This warping leaves distinct artifacts due to the resolution inconsistency between warped face area and surrounding context. As such, this artifacts can be used to detect DeepFake Videos.

Our method detects such artifacts by comparing the generated face areas and their surrounding regions with a dedicated Convolutional Neural Network (CNN) model. To train the CNN model, we simplify the process by simulating the resolution inconsistency in affine face warpings directly. Specifically, we first detect faces and then extract landmarks to compute the transform matrices to align the faces to a standard configuration. We apply Gaussian blurring to the aligned face, which is then affine warped back to original image using the inverse of the estimated transformation matrix. In order to simulate more different resolution cases of affine warped face, we align faces into multiple scales to increase the data diversity (see Figure 2). Compared to training a DeepFake model to generate fake images as negative data in [1, 10], which is time-consuming and resource-demanding ($\sim$ 72 hours on a NVIDIA GTX

GPU), our method creates negative data only using simple image processing operations which therefore saves a plenty of time and computing resources. Moreover, other methods may be over-fit to a specific distribution of DeepFake videos, our method is more robust since such artifacts are general in different sources of DeepFake videos. Based on our collected real face images from Internet and corresponding created negative data, we train four CNN models: VGG16 [31], ResNet50, ResNet101 and ResNet152 [11]. We demonstrate the effectiveness of our method on a DeepFake dataset from [20] and test several fake videos on `YouTube`.

2. Related works

AI-based Video Synthesis Algorithms The new generation of AI-based video synthesis algorithms are based on the recent developments in new deep learning models, especially the generative adversarial networks (GANs) [9]. A GAN model consists of two deep neural networks trained in tandem. The generator network aims to produce images that cannot be distinguished from the training real images, while the discriminator network aims to tell them apart. When training completes, the generator is used to synthesize images with realistic appearance.

The GAN model inspired many subsequent works for image synthesis, such as [8, 28, 2, 13, 32, 30, 21, 36, 3, 5]. Liu *et al.* [21] proposed an unsupervised image to image translation framework based on coupled GANs, which aims to learn the joint representation of images in different domains. This algorithm is the basis for the DeepFake algorithm.

The creation of a DeepFake video starts with an input video of a specific individual ('target'), and generates another video with the target's faces replaced with that of another individual ('source'), based on a GAN model trained to translate between the faces of the target and the source, see Figure 1. More recently, Zhu *et al.* [36] proposed cycle-consistent loss to push the performance of GAN, namely Cycle-GAN. Bansal *et al.* [3] stepped further and proposed Recycle-GAN, which incorporated temporal information and spatial cues with conditional generative adversarial networks. StarGAN [5] learned the mapping across multiple domains only using a single generator and discriminator.

Resampling Detection. The artifacts introduced by the DeepFake production pipeline is in essence due to affine transforms to the synthesized face. In the literature of digital media forensics, detecting transforms or the underlying resampling algorithm has been extensively studied, *e.g.*, [25, 26, 22, 15, 16, 17, 7, 24, 27, 12, 4]. However, the performance of these methods are affected by the post-processing steps, such as image/video compression, which are not subject to simple modeling. Besides, these methods usually aim to estimate the exact resampling operation from whole images, but for our purpose, a simpler solution can be obtained by just comparing regions of potentially synthesized faces and the rest of the image – the latter are expected to be free of such artifacts while the existence of such artifacts in the former is a telltale cue for the video being a DeepFake.

GAN Generated Image/Video Detection. Traditional forgery can be detected using methods such as [35, 6]. Zhou *et al.* [35] proposed two-stream CNN for face tampering detection. NoisePrint [6] employed CNN model to trace device fingerprints for forgery detection. Recently, detecting GAN generated images or videos has also made progress. Li *et al.* [20] observed that DeepFake faces lack realistic eye blinking, as training images obtained over the Internet usually do not include photographs with the subject's eyes closed. The lack of eye blinking is detected with a CNN/RNN model to expose DeepFake videos. However, this detection can be circumvented by purposely incorporating images with closed eyes in training. Yang *et al.* [34] utilized the inconsistency in head pose to detect fake videos. The work [19] exploited the color disparity between GAN generated images and real images in non-RGB color spaces to classify them. The work [23] also analyzed the color difference between GAN images and real images. However, it is not clear if this method is extensible to inspecting local regions as in the case of DeepFake. Afchar *et al.* [1] trained a convolutional neural networks namely MesoNet to directly classify real faces and fake faces generated by DeepFake and Face2face [33]. The work [10] extended [1] to temporal domain by incorporating RNN on CNN. While it shows promising performance, this holistic approach has its drawback. In particular, it requires both real and fake images as training data, and generating the fake images using the AI-based synthesis algorithms is less efficient than the simple mechanism for training data generation in our method.

3. Methods

We detect synthesized videos by exploiting the face warping artifacts resulted from the DeepFake production pipeline. For efficient running time, the current DeepFake algorithms create synthesized face images of fixed sizes. These faces are then undergone an affine transform (*i.e.*, scaling, rotation and shearing) to match the poses of the target faces that they will replace (see Figure 1 (g) – (h)). As such, the facial region and surrounding regions in the original image/video frame will present artifacts, the resolution inconsistency due to such transforms after the subsequent compression step to generate the final image or video frames. Therefore, we propose to use a Convolutional Neural Network (CNN) model to detect the presence of such

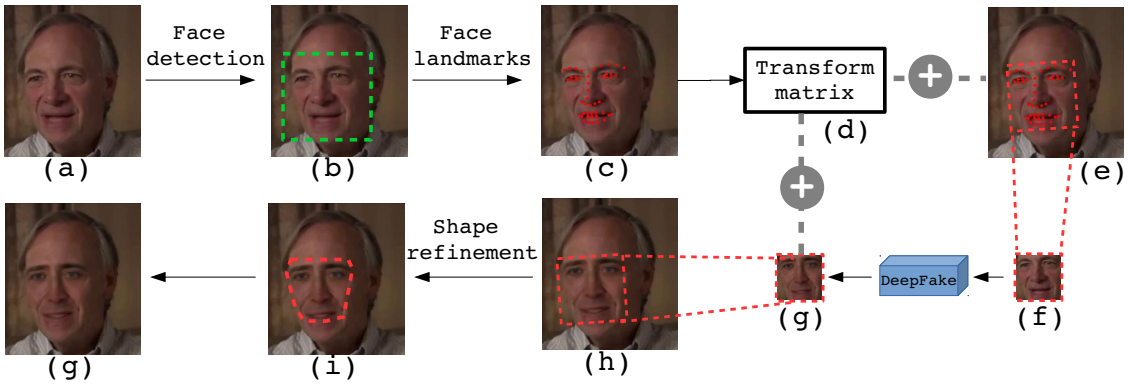

Figure 1. *Overview of the DeepFake production pipeline. (a) An image of the source. (b) Green box is the detected face area. (c) Red points are face landmarks. (d) Transform matrix is computed to warp face area in (e) to the normalized region (f). (g) Synthesized face image from the neural network. (h) Synthesized face warped back using the same transform matrix. (i) Post-processing including boundary smoothing applied to the composite image. (g) The final synthesized image.*

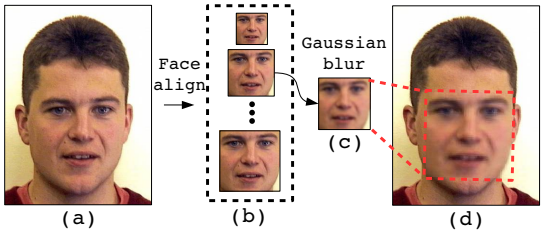

Figure 2. *Overview of negative data generation. (a) is the original image. (b) are aligned faces with different scales. We randomly pick a scale of face in (b) and apply Gaussian blur as (c), which is then affine warped back to (d).*

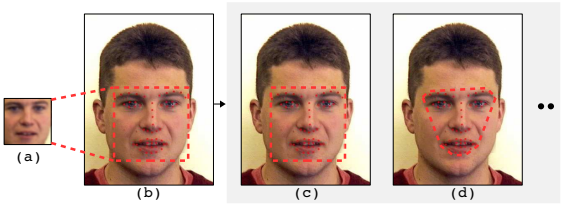

Figure 3. *Illustration of face shape augmentation of negative examples. (a) is the aligned and blurred face, which then undergoes an affine warped back to (b). (c, d) are post-processing for refining the shape of face area. (c) denotes the whole warped face is retained and (d) denotes only face area inside the polygon is retained.*

artifacts from the detected face regions and its surrounding areas.

The training of the CNN model is based on face images collected from the Internet. Specifically, we collect $24,442$ JPEG face images as positive examples. The negative examples can be generated by applying DeepFake algorithms as in [1], but it requires us to train and run the DeepFake algorithms, which is time-consuming and resource-demanding. On the other hand, as the purpose here is to detect the artifacts introduced by the affine face warping steps in DeepFake production pipeline, we simplify the negative example generation procedure by simulating the affine face warping step (Figure 1) directly.

Specifically, as shown in Figure 2, we take the following steps to generate negative examples to train the CNN model.

1. We detect faces in the original images and extract the face region using software package `dlib` [14];

2. We align faces into multiple scales and randomly pick one scale, which is then smoothed by a Gaussian blur with kernel size (5×5). This process aims to create more resolution cases in affine warped faces, which can better simulate different kinds of resolution inconsistency introduced in affine face warping.

3. The smoothed face undergoes an affine warp back to the same sizes of original faces to simulate the artifacts in the DeeFake production pipeline.

To further enlarge the training diversity, we change the color information: brightness, contrast, distortion and sharpness for all training examples. In particular, we change the shape of affine warped face area to simulate different post-processing procedure in DeepFake pipeline. As shown in Figure 3, the shape of affine warped face area can be further processed based on face landmarks. Figure 3(d) denotes a convex polygon shape is created based on the face landmarks of eye browns and the bottom of mouth.

From positive and negative examples, we crop regions of interest (RoI) as the input of our networks. As our aim is to expose the artifacts between fake face area and surrounding area, the RoIs are chosen as the rectangle areas that contains both the face and surrounding areas. Specifically, we determine the RoIs using face landmarks, as $[y_0 - \hat{y}_0, x_0 - \hat{x}_0, y_1 + \hat{y}_1, x_1 + \hat{x}_1]$, where y_0, x_0, y_1, x_1 denotes the minimum bounding box b which can cover all face

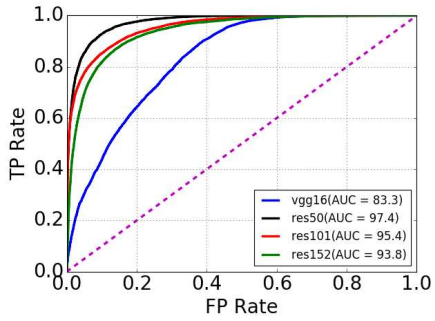

Figure 4. *Performance of each CNN model on all frames of UADFV [34].*

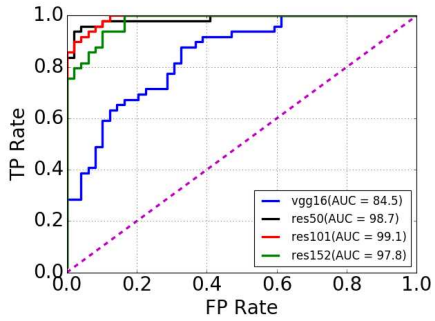

Figure 5. *Performance of each CNN model on each video of UADFV [34].*

landmarks excluding the outline of the cheek. The variables $\hat{y}_0, \hat{x}_0, \hat{y}_1, \hat{x}_1$ are random value between $[0, \frac{h}{5}]$ and $[0, \frac{w}{8}]$, where h, w are height and width of b respectively. The RoIs are resized to 224×224 to feed to the CNN models for training.

We train four CNN models — VGG16 [31], ResNet50, ResNet101 and ResNet152 [11] using our training data. For inference, we crop the RoI of each training example by 10 times. Then we average predictions of all RoIs as the final fake probability.

4. Experiments

We prepare our training data using the following strategy: instead of generating all negative examples in advance before training process, we employ a dynamic way to generate negative examples along with training process. For each training batch, we randomly select half positive examples and convert them into negative examples following the pipeline in Figure 2, which therefore makes the training data more diversified. We set batch size as 64, learning rate starting from 0.001 and decay 0.95 after each 1000 steps. We use SGD optimization method and the training process will be terminated until it reaches the maximum epoch. For VGG16, we directly train it using our data and terminate it at epoch 100. For ResNet50, ResNet101 and ResNet 152 models, we first load the ImageNet pretrained models and fine tune them using our data. The training process will be terminated at epoch 20. Then the models are fine-tuned using hard mining strategy. In our training, hard examples include positive examples with the predicted fake probability greater than 0.5, and negative examples with the predicted fake probability less than 0.5. We employ the same training procedure with learning rate from 0.0001. This stage is terminated after 20 epochs.

4.1. Evaluations on UADFV

We validate our method on DeepFake video dataset `UADFV` from [34]. This dataset contains 98 videos (32752 frames in total), which having 49 real videos and 49 fake videos respectively. Each video has one subject and lasts approximate 11 seconds. We evaluate the four models on this dataset using Area Under Curve (AUC) metric on two settings: *image based evaluation* and *video based evaluation*.

For image based evaluation, we process and send frames of all videos into our four networks respectively. Figure 4 illustrates the performance of each network on all frames. As these results show, the VGG16, ResNet50, ResNet101 and ResNet152 models achieve AUC performance $83.3\%, 97.4\%, 95.4\%, 93.8\%$, respectively. ResNet networks have about 10% better performance compared to VGG16, due to the residual connections, which make the learning process more effective. Yet, ResNet50 has the best performance among the other ResNet networks, which shows that as the depth of network increases, the classification-relevant information diminishes. For video based evaluation, we take each video as the unit of analysis. Due to the illumination changes, head motions and face occlusions in video, it is challenging to correctly predict the label of every frame. As such, we empirically assume a video is DeepFake-generated if a certain number of frames in this video are detected as fake. Thus we feed all frames of the video to the CNN based model and then return average the top third of the output score as the overall output of the video. Figure 5 shows the video-level performance of each type of CNN model. VGG16, ResNet50, ResNet101 and ResNet152 can achieve AUC performance $84.5\%, 98.7\%, 99.1\%, 97.8\%$ respectively. In this video based evaluation metric, ResNet network still performs $\sim 15\%$ better than VGG16. Yet, each ResNet model has similar performance, as in the case of image-level classification.

4.2. Evaluations on DeepfakeTIMIT

In addition, we also validate our method on another DeepFake video dataset `DeepfakeTIMIT` [18]. This dataset contains two set of fake videos which are made using a lower quality (LQ) with 64 x 64 input/output

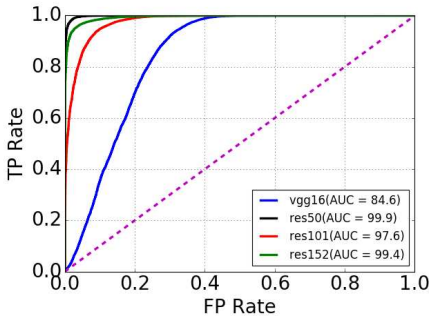

Figure 6. *Performance of each CNN model on all frames in LQ set of DeepFakeTIMIT [18].*

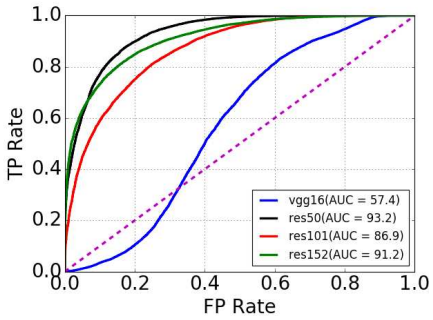

Figure 7. *Performance of each CNN model on all frames in HQ set of DeepFakeTIMIT [18].*

size model and higher quality (HQ) with 128 x 128 size model, respectively. Each fake video set has 32 subjects, where each subject has 10 videos with faces swapped. Each video is 512×384 and lasts $\sim$ 4 seconds. The original videos of corresponding 32 subjects are from `VidTIMIT` dataset [29]. We select subset of each subject from original dataset `VidTIMIT` and all fake videos from `DeepfakeTIMIT` for validation (10537 original images and 34023 fake images for each quality set). We evaluate our four models on each frame of all videos based on AUC metric, where the performance of VGG16, ResNet50, ResNet101 and ResNet152 models on LQ and HQ video sets are $84.6\%, 99.9\%, 97.6\%, 99.4\%$ and $57.4\%, 93.2\%, 86.9\%, 91.2\%$ respectively, see Figure 6 and Figure 7.

We have also tested our algorithm on several DeepFake videos that were generated and uploaded to `YouTube` by anonymous users. In Figure 8, we show the detection results as the output score from the ResNet50 based CNN model for one particular example[1], where an output of 0 corresponds to a frame free of the warping artifacts. As these results show, the CNN model is effective in detecting the existence of such artifacts, which can be used to determine if these videos are synthesized using the DeepFake algorithm.

[1] https://www.youtube.com/watch?v=BU9YAHigNx8

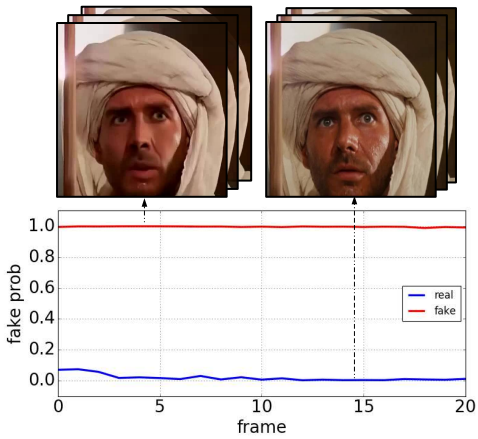

Figure 8. *Example of our method on a DeepFake generated video clip from* `YouTube` **(left)** *and original video clip* **(right)**.

Table 1. AUC performance of our method and other state-of-the-art methods on UADFV and DeepfakeTIMIT datasets.

Methods	**UADFV**	**DeepfakeTIMIT**	
		LQ	HQ
Two-stream NN [35]	85.1	83.5	73.5
Meso-4 [1]	84.3	87.8	68.4
MesoInception-4	82.1	80.4	62.7
HeadPose [34]	89.0	-	-
Ours-VGG16	84.5	84.6	57.4
Ours-ResNet50	**97.4**	**99.9**	**93.2**
Ours-ResNet101	95.4	97.6	86.9
Ours-ResNet152	93.8	99.4	91.2

4.3. Comparing with State-of-the-arts

We compare the AUC performance of our method with other state-of-the-art methods: the face tampering detection method *Two-stream NN* [35], and two DeepFake detection methods *MesoNet* [1] and *HeadPose* [34] on the `UADFV` dataset and `DeepfakeTIMIT` dataset. For *MesoNet*, we test the proposed two architectures: Meso-4 and MesoInception-4. Table 1 shows the performance of all the methods. As the results show, our ResNet models outperform all other methods. Specifically, ResNet50 achieves best performance, which outperforms *Two-stream NN* by $\sim 16\%$ on both datasets that thereby demonstrates the efficacy of our method on DeepFake video detection. Our method also outperforms Meso-4 and MesoInception-4 by $\sim 17\%$ and $\sim 21\%$ on both datasets. Specifically, our method has a notable advance in HQ set of `DeepfakeTIMIT`. Since *MesoNet* is trained using self-collected DeepFake generated videos, it may over-fit to a specific distribution of DeepFake videos in training. In contrast, our method focuses on more intuitive aspect in DeepFake video generation: resolution inconsistency in face warping, which is thereby more robust to DeepFake videos of different sources. *HeadPose* utilizes head pose inconsistency to distinguish real and fake videos.

However, such physiological signal may not be notable in frontal faces, such that our method outperforms it by $\sim 8\%$ on UADFV.

5. Conclusion

In this work, we describe a new deep learning based method that can effectively distinguish AI-generated fake videos (DeepFake Videos) from real videos. Our method is based on the observations that current DeepFake algorithm can only generate images of limited resolutions, which are then needed to be further transformed to match the faces to be replaced in the source video. Such transforms leave certain distinctive artifacts in the resulting DeepFake Videos, which can be effectively captured by a dedicated deep neural network model. We evaluate our method on several different sets of available DeepFake Videos which demonstrate its effectiveness in practice.

As the technology behind DeepFake keeps evolving, we will continuing improve the detection method. First, we would like to evaluate and improve the robustness of our detection method with regards to multiple video compression. Second, we currently using predesigned network structure for this task (*e.g.*, resnet or VGG), but for more efficient detection, we would like to explore dedicated network structure for the detection of DeepFake videos.

References

[1] Darius Afchar, Vincent Nozick, Junichi Yamagishi, and Isao Echizen. Mesonet: a compact facial video forgery detection network. In *IEEE International Workshop on Information Forensics and Security (WIFS)*, 2018. 1, 2, 3, 5

[2] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017. 2

[3] Aayush Bansal, Shugao Ma, Deva Ramanan, and Yaser Sheikh. Recycle-gan: Unsupervised video retargeting. In *ECCV*, 2018. 2

[4] Jason Bunk, Jawadul H Bappy, Tajuddin Manhar Mohammed, Lakshmanan Nataraj, Arjuna Flenner, BS Manjunath, Shivkumar Chandrasekaran, Amit K Roy-Chowdhury, and Lawrence Peterson. Detection and localization of image forgeries using resampling features and deep learning. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2017 IEEE Conference on*. IEEE, 2017. 2

[5] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *CVPR*, 2018. 2

[6] Davide Cozzolino and Luisa Verdoliva. Noiseprint: a cnn-based camera model fingerprint. *arXiv preprint arXiv:1808.08396*, 2018. 2

[7] Nahuel Dalgaard, Carlos Mosquera, and Fernando Pérez-González. On the role of differentiation for resampling detection. In *Image Processing (ICIP), 2010 17th IEEE International Conference on*. IEEE, 2010. 2

[8] Emily L Denton, Soumith Chintala, Rob Fergus, et al. Deep generative image models using a laplacian pyramid of adversarial networks. In *NIPS*, 2015. 2

[9] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*, 2014. 2

[10] David Güera and Edward J Delp. Deepfake video detection using recurrent neural networks. In *AVSS*, 2018. 1, 2

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 2, 4

[12] Xiaodan Hou, Tao Zhang, Gang Xiong, Yan Zhang, and Xin Ping. Image resampling detection based on texture classification. *Multimedia tools and applications*, 2014. 2

[13] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017. 2

[14] Davis E. King. Dlib-ml: A machine learning toolkit. *Journal of Machine Learning Research*, 10:1755–1758, 2009. 3

[15] Matthias Kirchner. Fast and reliable resampling detection by spectral analysis of fixed linear predictor residue. In *Proceedings of the 10th ACM workshop on Multimedia and security*. ACM, 2008. 2

[16] Matthias Kirchner and Rainer Bohme. Hiding traces of resampling in digital images. *IEEE Transactions on Information Forensics and Security*, 2008. 2

[17] Matthias Kirchner and Thomas Gloe. On resampling detection in re-compressed images. In *Information Forensics and Security, 2009. WIFS 2009. First IEEE International Workshop on*. IEEE, 2009. 2

[18] Pavel Korshunov and Sébastien Marcel. Deepfakes: a new threat to face recognition? assessment and detection. *arXiv preprint arXiv:1812.08685*, 2018. 4, 5

[19] Haodong Li, Bin Li, Shunquan Tan, and Jiwu Huang. Detection of deep network generated images using disparities in color components. *arXiv preprint arXiv:1808.07276*, 2018. 2

[20] Yuezun Li, Ming-Ching Chang, and Siwei Lyu. In ictu oculi: Exposing ai generated fake face videos by detecting eye blinking. In *IEEE International Workshop on Information Forensics and Security (WIFS)*, 2018. 2

[21] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. Unsupervised image-to-image translation networks. In *NIPS*, 2017. 2

[22] Babak Mahdian and Stanislav Saic. Blind authentication using periodic properties of interpolation. *IEEE Transactions on Information Forensics and Security*, 2008. 2

[23] Scott McCloskey and Michael Albright. Detecting gan-generated imagery using color cues. *arXiv preprint arXiv:1812.08247*, 2018. 2

[24] Hieu Cuong Nguyen and Stefan Katzenbeisser. Robust resampling detection in digital images. In *IFIP International Conference on Communications and Multimedia Security*. Springer, 2012. 2

[25] Alin C Popescu and Hany Farid. Exposing digital forgeries by detecting traces of resampling. *IEEE Transactions on signal processing*, 2005. 2

[26] S Prasad and KR Ramakrishnan. On resampling detection and its application to detect image tampering. In *ICME*. IEEE, 2006. 2

[27] Ruohan Qian, Weihai Li, Nenghai Yu, and Zhuo Hao. Image forensics with rotation-tolerant resampling detection. In *Multimedia and Expo Workshops (ICMEW), 2012 IEEE International Conference on*. IEEE, 2012. 2

[28] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015. 2

[29] Conrad Sanderson and Brian C Lovell. Multi-region probabilistic histograms for robust and scalable identity inference. In *International Conference on Biometrics*. Springer, 2009. 5

[30] Ashish Shrivastava, Tomas Pfister, Oncel Tuzel, Josh Susskind, Wenda Wang, and Russ Webb. Learning from simulated and unsupervised images through adversarial training. In *CVPR*, 2017. 2

[31] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 2, 4

[32] Yaniv Taigman, Adam Polyak, and Lior Wolf. Unsupervised cross-domain image generation. *arXiv preprint arXiv:1611.02200*, 2016. 2

[33] Justus Thies, Michael Zollhofer, Marc Stamminger, Christian Theobalt, and Matthias Niessner. Face2face: Real-time face capture and reenactment of rgb videos. In *CVPR*, June 2016. 2

[34] Xin Yang, Yuezun Li, and Siwei Lyu. Exposing deep fakes using inconsistent head poses. In *ICASSP*, 2019. 2, 4, 5

[35] Peng Zhou, Xintong Han, Vlad I Morariu, and Larry S Davis. Two-stream neural networks for tampered face detection. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). IEEE*, pages 1831–1839, 2017. 2, 5

[36] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, 2017. 2