Two-branch Recurrent Network for Isolating Deepfakes in Videos

Iacopo Masi*[0000−0003−0444−7646], Aditya Killekar*, Royston Marian Mascarenhas, Shenoy Pratik Gurudatt, and Wael AbdAlmageed

USC Information Sciences Institute, Marina del Rey, CA, USA
{iacopo,killekar,royston,gurudatt,wamageed}@isi.edu

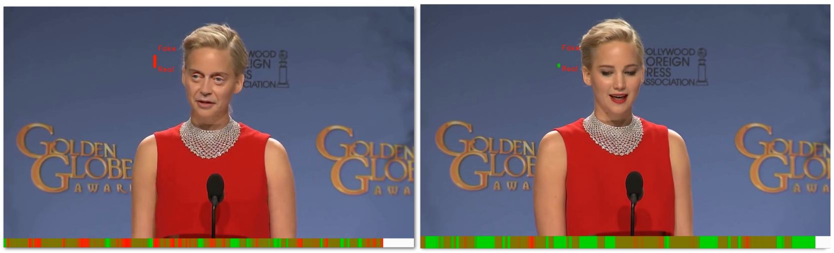


Fig. 1: **Deepfake detection in videos.** The figure shows our system predictions when trained only on FaceForensics++ for a fake video "in the wild" (left) and its pristine video (right). The heatmap bar below the video indicates the likelihood of the video being fake (red) or unmanipulated (green).

Abstract. The current spike of hyper-realistic faces artificially generated using *deepfakes* calls for media forensics solutions that are tailored to video streams and work reliably with a low false alarm rate at the video level. We present a method for *deepfake* detection based on a two-branch network structure that isolates digitally manipulated faces by learning to amplify artifacts while suppressing the high-level face content. Unlike current methods that extract spatial frequencies as a preprocessing step, we propose a two-branch structure: one branch propagates the original information, while the other branch suppresses the face content yet amplifies multi-band frequencies using a *Laplacian of Gaussian (LoG)* as a bottleneck layer. To better isolate manipulated faces, we derive a novel cost function that, unlike regular classification, compresses the variability of natural faces and pushes away the unrealistic facial samples in the feature space. Our two novel components show promising results on the FaceForensics++, Celeb-DF, and Facebook's DFDC preview benchmarks, when compared to prior work. We then offer a full, detailed ablation study of our network architecture and cost function. Finally, although the bar is still high to get very remarkable figures at a very low false alarm rate, our study shows that we can achieve good video-level performance when cross-testing in terms of video-level AUC.

Keywords: deepfake detection, two-branch recurrent net, loss function

[*] indicates equal contribution

1 Introduction

Visual misinformation has dramatically increased on social networks and Internet [1]. Nonetheless, image manipulation is not new. Falsification of lithographs or photographs has been used for many years to reinforce political ideas or political characters [21] or to practice censorship by erasing people from pictures. For instance, at the beginning of the twentieth century, political dissidents were assassinated and then erased from photographs through airbrushing during the Great Terror period in the Soviet Union [19].

In the modern era of digital pictures, perpetrators used commercial software and "elbow grease" to create realistic swapping of faces given a pair of still images. Although some of these results look very realistic, they involved a huge amount of manual work (on the order of hours) using a personal computer and an expensive raster graphics editor to produce just a single image [24]. However, the effort required to produce face swaps diminished drastically during the last five years. Democratized artificial intelligence (AI) made it very easy to produce highly realistic face swaps with a few clicks, giving the ability to non-experts to synthesize content with "Hollywood-like" effects just by simply using off-the-shelf applications [47]. The technology was quickly developed to process videos, transferring the identity of a subject from a *source* video into a *target* video. Unlike manual digital editing, face swapping in videos became effective and efficient, reaching hyper-realistic results, thanks to recent advances in data synthesis using Generative Adversarial Networks (GANs) [20], Deep Convolutional Neural Networks (DCNN) [35], and AutoEncoders (AE) [31]. It also became easily available to non-experts through customized applications, such as DeepFaceLab [2], or even mobile applications, such as Zao [4].

Face swapping has been superseded by deepfakes in which the original face is replaced with a victim's face with the intent of showing the victim to be saying something he/she never said. The fake video is usually very realistic so that the viewer believes that the swapped subject is the actual acting person in the video. Although in the beginning, deepfakes were used to entertain users, they became popular to spread political chaos, revenge porn, and defamation. For these reasons, the rapid sharing of deepfakes on the Internet became a threat to society leading to a common perception that *seeing is no longer believing* [1].

A recent report from DeepTrace [3] explains that the rate of increase in these fakes videos is 100% a year. Although deepfakes initially appeared in 2017 on `reddit`, the report estimates that there are currently 14,678 realistic-looking yet fake videos, while the total number available in December 2018 was only 7,964. Given the current progress of AI and deep learning, the prediction is that this number may skyrocket in the near future. In order to mitigate the proliferation of manipulated videos, we propose a deep learning architecture to detect hyper-realistic face manipulations. The paper makes the following contributions:

◇ A two-branch representation extractor based on densely connected layers [25] that learns to combine information from the color domain and the frequency domain using a multi-scale Laplacian of Gaussian (LoG) operator [10].

The LoG operator suppresses the image content present in the low-level feature maps, acting as a band-pass filter to amplify artifacts.

◇ A novel loss function that encourages compactness of the representations of natural faces and pushes away manipulated faces for better, wider separation boundaries, which is different than recent methods that use binary cross-entropy for detecting face manipulations [50,52].

◇ As a minor contribution, we argue that current metrics (accuracy) are improper for this problem, mainly for being very sensitive to class imbalance and failure to capture performance for web-scale applications. Therefore, we follow [33,55] and report True Acceptance Rate (TAR) at low False Acceptance Rates (FAR). Also, besides standard area under receiver operating curve (AUC), we further propose global metrics at a low false alarm rate such as standardized partial AUC (pAUC) [42] and our truncated Area Under the Curve (tAUC).

We optimize our method for better generalization across datasets, reaching a good balance between bias and variance [48,59,15], i.e., performing remarkably on same dataset used for training [50] yet transferring reasonably well across datasets [38,14]. Similar to only few works in literature [21,52], we also use sequential modeling for video-based detection. Our method processes sequences of aligned faces from a video, extracts discriminative features using the backbone, and performs recurrent modeling using bi-directional long short-term memory (LSTM) supervised by our new loss. The entire network is trained end-to-end so that the recurrent model back-propagates to the feature extractor. Fig. 1 shows the predictions of our system on face videos downloaded from the web when trained only on FaceForensics++. Our method is summarized in Fig. 2.

2 Prior Work

Face forensics datasets and evaluation. Unlike the proliferation of face recognition datasets [26,32,29,5,22], there has been a lack of large-scale face forensics datasets in the community for both training and evaluation. Although face swapping can be cast as a splicing image forgery technique, and some generic forensics sets contain facial splicing and copy-move forgeries [24], earlier specific face manipulation detection tools [23] have been mainly evaluated on still images. Small-scale benchmarks released for deepfake detection were produced in controlled environments, e.g., DF-TIMIT [33,34] using 32 subjects selected from the VidTIMIT [53] database with the intent of studying the weaknesses of face detection and recognition technology, or UADFV [36] that offers around 50 bona fide and 50 fake videos.

Only recently, Rossler *et al.* proposed several versions of FaceForensics++ [50], a medium-scale collection of manipulated videos counting a total of 1.8 million manipulated frames using four methods: FaceSwap, DeepFakes, Face2Face [57], and NeuralTextures [56]. The same dataset was augmented by Google Research with another set containing deepfake videos, i.e., Google Deepfake Detection (DFD) [16]. At the same time, Facebook and other firms joined efforts to create a competition to detect fakes on the web, releasing a preview dataset “The

Deepfake Detection Challenge (DFDC)" [14] along with new metrics for evaluation. With the exception of [37], the interesting novel aspect is that performance is considered at a video-level instead of frame-level, effectively evaluating models at low false alarm rate. Before [14], accuracy was the only metric used to measure fake detection performance with a few exceptions [33,55]. Despite these contributions, the perceived quality of the synthesized videos offered by these sets appears still lower compared to the videos circulating on the web, thus Li *et al.* recently released Celeb-DF [38] to produce hyper-realistic deepfakes, reporting the frame-level AUC as a metric. This benchmark is compelling, offering $5,369$ high quality videos for a total of 2.1M frames.

Detection of face manipulations. Although image forensics has been widely studied for a long time [17], deepfakes is recent technology and thus several orthogonal works have been proposed lately for solving the problem of detecting face manipulations. Methods for deepfake detection can be roughly categorized in two macroscopic groups — (i) discriminative classifiers that use diverse semantic inconsistencies of the head and face; and (ii) data-driven approaches directly learning a discriminative function from data. Considering the first group, Agarwal *et al.* built person-specific classifiers [7] using one-class support vector machines (SVM) and features computed from Action Units (AU) and 3D head pose movements. Similarly, Li *et al.* [36] used the observation that initial versions of deepfakes were not blinking. Later they extended the work to check for the inconsistency of 3D head poses. They also trained a DCNN though they used as negative samples faces undergoing warping artifacts to simulate the deepfake stitching process [37], while [41] used hand-crafted visual features to amplify artifacts. Regarding the second group, XceptionNet [11] has been widely used [50], while [6] used a variant of Inception module to capture both micro and mesoscopic features. Han *et al.* [23] used a two-branch structure similar to ours, yet, unlike our method, performed late fusion between a RGB branch and steganalysis features, using triplet loss for supervision. Later methods employed multi-task learning [44] with an encoder-decoder similar to [13] and capsule networks [45]. For a complete survey, we refer to [38,46] and the recent work in [58,61].

GAN synthesis detection. Finally, a parallel line of related research [69,68,13] [60,40] is detecting entirely GAN-synthesized face images, e.g., using StyleGAN [28]. Our work shares similar traits with the very recent research by Yu *et al.* [68] with some major differences. We focus on detecting deepfakes, while [68]'s interest is in modeling GAN fingerprints. More importantly, our method is composed of a two-branch structure that fuses RGB information with the frequency domain.

3 Method

The objective is to learn a classifier for the detection of manipulated faces, squishing a set of aligned video frames[1] $\mathbf{I} \in \mathbb{R}^{H \times W \times 3 \times F}$ to an embedding $\Phi(\mathbf{I}) \in \mathbb{R}^{D}$

[1] Throughout this paper $\mathbf{I}$ indicates a sequence (or window) of aligned faces from video frames of cardinality F.

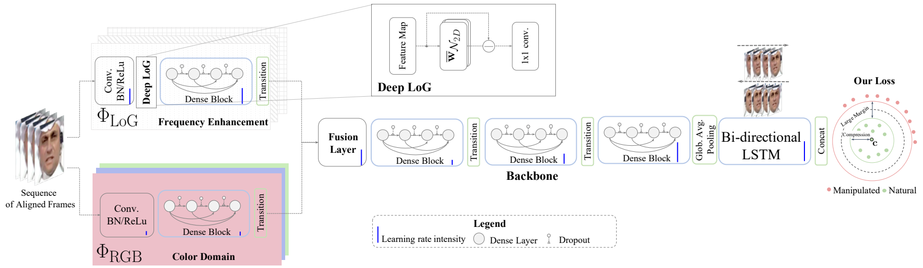


Fig. 2: **Our video-based face manipulation detection architecture.** A face sequence is processed by two independent DenseBlocks: one subject to a Deep Laplacian of Gaussian (Deep LoG) layer (frequency enhancement) and the second is a branch that works in the color domain. The two feature maps are fused so that a backbone of dense blocks learns a rich representation. The architecture uses dropout after each DenseLayer and a different learning rate per layer to mitigate overfitting. Our architecture ends with a bi-directional LSTM layer supervised using a novel loss formulation.

so that the representations of natural faces are compact around a reference centroid $\mathbf{c}$ and manipulated faces are spread out, ensuring a large margin between tampered and untamperd faces. In Section 3.1 we introduce a two-branch backbone representation extractor $\Phi(\cdot)$ based on densely connected layers [25]. Φ learns to fuse different representations obtained using regular convolutional filters Φ_{RGB} and representations extracted using multi-scale Laplacian of Gaussian [10] kernels Φ_{LoG} (Section 3.2). The combined features maps are then fed to the backbone that ends with a bi-directional Long Short-Term Memory (LSTM) for temporal modeling. $\Phi(\mathbf{I})$ indicates the concatenated output from the two bidirectional LSTM streams. The entire recurrent model is supervised through a novel formulation. Unlike recent methods [50,52] that use classification losses for detection, in Section 3.3 we introduce a loss function that encourages the compactness of the representations of untampered faces, while distancing the representations of manipulated faces, for wider separation boundaries. At test-time, given an input sequence $\mathbf{I}$, the method obtains the distance $\left\|\Phi(\mathbf{I})-\mathbf{c}\right\|_2$; the larger the distance the higher the likelihood of the sample being manipulated.

3.1 Network Architecture and Optimization

Architecture. The basic network architecture Φ is derived by the recent work in [52] with major modification. The network takes as input RGB faces but consists of two different branches: a regular DenseBlock [25] that learns to process color domain data Φ_{RGB} and another parallel DenseBlock Φ_{LoG} with unshared weights that learns to discard visual face content by applying a Laplacian of Gaussian (LoG) filter to the low-level feature maps. The two feature maps are aligned and have a resolution of 28×28 with 128 planes. These maps are fused

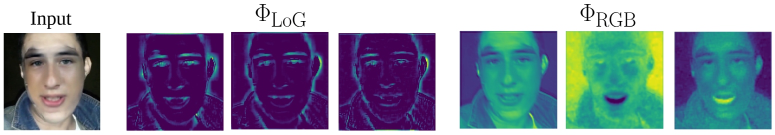


Fig. 3: **Diverse feature maps.** A diverse set of feature maps is obtained in the two distinct branches. The three representative feature maps after the first convolutional layer and our Deep LoG layer from Φ_{LoG} are shown on the left with the input. The feature maps on the right show the response from Φ_{RGB}.

together with point-wise convolution with two groups [27] such that each group of convolutional filters independently refines and fuses the information for the three downstream DenseBlocks. All the DenseBlocks end with a Transitional layer except for the one prior the LSTM. We discard the final linear classification layer and reduce the final feature map to a feature vector with dimension 1024 using global average pooling. Dropout with a probability of 0.2 is applied at the end of each DenseLayer to avoid overfitting.

Optimization. Instead of optimizing the entire network with a single learning rate, which may overfit given the large parameter space of DenseNet, we employ a strategy that sets different learning rates per DenseBlock. In particular, given the global learning rate μ, we define a decay for the DenseBlocks so that downstream layers incorporate gradients quickly while upstream layers change less drastically. The learning rate decay is defined as $\mu_L \doteq \mu \cdot 1/(2^L)$ for the entire network with the exception of the DenseBlock of the Deep LoG and the layer in charge of fusing the two-branches. The parameters of fusion layers are updated faster, with the global rate μ, since that they have to be adapted to the frequency domain information. The blue bars in Fig. 2 indicate the intensity of learning rate for each layer. We initialize all the layers from pre-trained weights from ImageNet except those of the LSTM, which are instead initialized from a uniform distribution.

Video-based stratified sampling. Since inter-video variations are stronger than intra-video, we define a "stratified epoch" as the set of sequences obtained by sampling a *random sequence once from all the videos* in the training set. Stratified training ensures that each mini-batch includes a diversified set of training samples, avoiding including similar sequences from the same video within the mini-batch if we use simple random sampling.

3.2 Deep Laplacian of Gaussian

Motivation. Zhang *et al.* [69] show that image manipulations leave medium- and high-frequency traces, and since most of the deepfakes methods reconstruct a face with an average pixel-wise ℓ_2 loss (Mean Square Error (MSE)), producing somewhat blurry (low-frequency) facial features, and because of the up-sampling commonly used in the decoder part. According to these observations, we design a custom layer to propagate multi-band frequency information inside the network with the goal of suppressing high-level face content, thereby amplify artifacts.

Implementation. Without loss of generality, given a feature map $\mathbf{x}$, we apply a Laplacian of Gaussian (LoG) [10] as follows. Input tensor $\mathbf{x}$ (which can be in the first layer the input image alone or directly higher feature maps) is processed by two sets of convolutional filters: fixed, non-learnable filters $\overline{\mathbf{w}}_{\mathcal{N}_{2D}}$, as a 2D Gaussian kernel and a dimensionality reduction filter $\mathbf{w}_{1\times1}$ that maps back the dimensionality to the input expected by the next layer. The layer shares a similar design with [66] although in our case (1) the objective is to suppress global information from the face, not to suppress noise from adversarial samples and (2) the skip-connection is used to *remove* information while [66] used it to ease the training. As described in Eq. (1), the output feature map $\mathbf{x}'$ is then obtained as shown in Eq. (1)

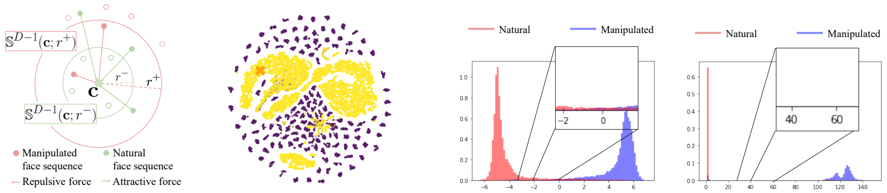
$$\mathbf{x}' = \mathbf{w}_{1\times1}\Big(\mathbf{x} - \mathrm{up}\big(\mathrm{down}(\overline{\mathbf{w}}_{\mathcal{N}_{2D}}\ \mathbf{x})\big)\Big), \tag{1}$$

where $\mathrm{up}(\cdot), \mathrm{down}(\cdot)$ indicate upsampling and downsampling, respectively, of the tensor across multiple scales $S{=}3$. Assuming that the LoG is inserted into a layer with K input planes and K' output planes, then internally the tensor depth dimension becomes $K \xrightarrow{\text{LoG}} S\ K \xrightarrow{\mathbf{w}_{1\times1}} K'$. Fig. 3 shows the difference in the feature maps between the new LoG branch and the regular RGB branch after the first convolutional layer. All the feature maps are taken from the same filters, so they are aligned across branches. Each map is normalized $\in [0,1]$. We can see that most of the energy in the response for the Φ_{LoG} case is around the edges, the Φ_{RGB} counterpart instead focuses more on the global structure of the face.

3.3 Loss Function to Isolate Manipulated Faces

Motivation. The majority of previous work on face manipulation detection [69,50,52] uses standard cost functions adopted from classification. Cozzolino *et al.* [13] recently made an effort toward representation disentanglement and generalization for face manipulation detection. Unlike previous work summarized in Section 2, we propose a new loss for better isolating manipulated faces inspired by recent work on one-class classifiers, such as one-class Deep Support Vector Data Description (Deep SVDD) [51]. The new formulation induces compactness of the embedding space for sequences of unmanipulated faces. However, unlike [51], the proposed loss employs manipulations synthesized by a few generators as negative samples enforcing a larger margin to the natural face sequences.

Formulation. More formally, we optimize the entire recurrent network defined in Section 3.1 through a cost function that organizes the feature space such that the variability of sequences of natural faces is compacted toward a reference center while the representations of manipulated face sequences are placed far apart at the boundaries of the feature space. Before training, we begin by pre-computing a reference center $\mathbf{c} \in \mathbb{R}^D$ by averaging the encodings of all the natural, unmanipulated face sequences in the training set. The encodings are obtained by taking the responses of our entire architecture with two-branches and the bi-directional LSTM before training. The concatenated bidirectional LSTM features are extracted using the same network that is pre-trained. The



(a) Loss idea (b) Feature space (c) Logit distribution (d) Our distribution

Fig. 4: **Loss formulation.** (a) The loss induces compression of the natural face sequences within a inner hypersphere placing easier samples close to $\mathbf{c}$ and tougher samples at the boundary; meanwhile it induces a large margin forcing the manipulated face sequences outside the outer hypersphere (b) t-SNE [39] visualization of the feature space on the test set between natural faces (yellow) and deepfakes (violet). The center $\mathbf{c}$ is shown as an orange cross. (c) Genuine-Impostor distribution of logits with binary cross-entropy and (d) with our loss function: imposing a wider margin induces less confusion in the distribution.

two-branches and the backbone are pre-trained on ImageNet. When the training starts, all the features are aligned to this predefined embedding space. Then we define two hyperspheres centered around $\mathbf{c}$ to constrain the feature space so that natural faces lie within $\mathbb{S}^{D-1}(\mathbf{c}; r^-)$, while manipulated faces are kept outside $\mathbb{S}^{D-1}(\mathbf{c}; r^+)$. The loss induces compression on the regular faces embeddings. However, unlike [51], we avoid reducing all samples to a single high-dimensional point and mitigate overfitting by requiring compression up to an internal inner margin defined by the radius r^- of the first hypersphere. Furthermore, the proposed loss enforces sequences of manipulated faces to be kept outside the second hypersphere defined by the radius r^+. The loss $\mathcal{L}$ given a mini-batch $\Omega \in \mathbb{R}^{H \times W \times 3 \times F \times B}$ of face sequences is defined as shown in Eq. (2):

$$\mathcal{L} = \frac{1}{|\Omega_{\text{nat.}}|} \sum_{i \in \Omega_{\text{nat.}}} \max\left(0, \left\|\Phi(\mathbf{I}_i) - \mathbf{c}\right\|_2 - r^-\right) + \\ + \frac{1}{|\Omega_{\text{man.}}|} \sum_{j \in \Omega_{\text{man.}}} \max\left(0, r^+ - \left\|\Phi(\mathbf{I}_j) - \mathbf{c}\right\|_2\right), \quad (2)$$

where $\Omega_{\text{nat.,man.}}$ selects natural and manipulated face samples, respectively. For this loss to be valid, it has to hold that $0 < r^- < r^+$ and the margin imposed between the two classes is $m = r^+ - r^-$. The values of the two radii have to be set according to the dimensionality D of the feature embedding. The loss mitigates the problem of class imbalances by normalizing each term by its cardinality. Further, the second margin r^+ is essential to the loss because the network may chose to lower the cost just by pushing the negative samples indefinitely, without inducing compression on the natural faces.

Fig. 4a illustrates the basic idea of the proposed loss, and Fig. 4b demonstrates the feature space of the test set of natural faces vs deepfakes. The features are mapped to $\mathbb{R}^2$ using t-SNE [39] optimizing a plain DenseNet model. Natural faces are compressed while manipulated faces lie at the boundaries. The clusters formed by videos are visible for the manipulated faces. Fig. 4c shows the genuine and impostor distribution of the logits at inference time for a model trained for discerning real faces from deepfakes using binary cross-entropy on FaceForensics++ [50]. Although the distribution presents two peaks corresponding to real and deepfakes faces, the variance of those distribution is not minimized, and, more importantly, real face logits are spread out toward the manipulated faces thereby negatively affecting the detection rate at a low false alarm regime. In contrast, Fig. 4d offers the distribution of the distances from the center $\mathbf{c}$ for the two classes. Using the proposed loss we achieved compression of the natural faces and a clear separation from the manipulated faces, visible when zooming in a highly confusing region.

Interpretation. Eq. (2) shares similar traits with the formulation in [51] with a few key differences. First, we have a secondary term for supervision for abnormal cases. Second, we have margins that avoid overfitting and better separate the two classes. The loss function also resembles the classic formulations found in deep metric learning such as contrastive loss functions [65], although in our case the optimization is better constrained since the network is allowed to "move" only $\Phi(\mathbf{I})$ while $\mathbf{c}$ is kept fixed. Finally, we spare the sampling of pairs or even triplets [54] which significantly reduces training complexity. Our loss differs from recent formulations: [64] uses softmax while we do not; it also sets one center for each class while we have a single center for both classes; finally, unlike us, [64] updates the centers while training. The work in [62] enforces angular margin whereas ours uses radial distance margin; [62] induces compactness in all the classes while ours only on natural face sequences.

4 Experimental Evaluation

Benchmarks and metrics. Ablation studies and comparisons are conducted on (1) FaceForensics++ [50], (2) Celeb-DF [38], (3) and the Deepfake Detection Challenge (DFDC) Preview Dataset [14]. We report results at the video-level and also at the frame-level. Given that our method works at a sequence level, when comparing to other methods, we made sure that the number of samples prior computing the ROC is the same for all methods when comparing at the frame-level or, at least, that that all methods observed the same quantity of data. Further, we use standard metrics such as True Acceptance Rate (TAR) at low False Acceptance Rates (FAR), similar to [33,55]. Besides standard area under receiver operating curve (AUC), we further use global metrics yet at a low false alarm rate such. These metrics can shed light on performance in realistic operational scenarios, thereby requiring detectors to operate at a very low false alarm rate and raising the bar for the community. We used the standardized partial AUC or pAUC [42] and our tAUC, that is defined as AUC yet taking into consideration only the low false alarm rate up to a cut-off point FAR_τ, thereby ignoring high false alarm rates. tAUC is computed as the ratio between the area of TARs up to a given low FAR_τ normalized by the total area up to the FAR_τ value. Given $\mathcal{F}_\tau = \{0, \ldots, \mathrm{FAR}_\tau\}$, then tAUC at an operating point τ is defined as $\mathrm{tAUC}_\tau \doteq \frac{\sum_{i \in \mathcal{F}_\tau} \mathrm{TAR}_i}{|\mathcal{F}_\tau|}$.

Implementation and Hyper-Parameters. Unless otherwise stated, we used the following settings. The global learning rate μ is 1e-03 using the Adam optimizer and the results are produced with LSTM. The learning rate is decreased three times by a factor of 10. We decrease it every time the validation loss does not decrease after 50 stratified epochs. We used a weight decay of 1e-06. The final global average pooling flattening the spatial dimension gives a descriptor with dimensionality 1024 transformed into D=128 by the LSTM. The final dimensionality considered in the loss is $2D$[2] and the two radii $r^{\{-,+\}} \doteq \{0.042, 1.638\}$ have to be optimized together and cross-validated on a validation set. In high-dimensional space, the volume of the hyper-sphere decreases when the feature descriptor dimension D increases [63]: thus, if D does change, the radii have to be changed accordingly. By increasing the dimensionality D of the final feature, the radii have to be increased as well to compensate for the diminished hyper-volume of the hyper-sphere. The cardinality F of the sequence of aligned frames as input to the recurrent model is 10. Since the sequential modeling is trained on sampled FF++ data, at inference time we take 1 frame over 7 to build the sequence. Faces are aligned with dlib [30]. If alignment fails, we revert back to [9]. In case of multiple detected faces, we select the largest detected face. Since FaceForensics++ has imbalanced labels (1:4), we oversample the natural faces twice and undersample randomly faces for each manipulation with a factor of two to get a proper balance, when training with multiple manipulations. We used average to perform video-level evaluation to aggregate all the scores within a video for all methods. When doing cross-testing, we use always the same model trained on FF++ on the four manipulations on high compression (c40).

4.1 FaceForensics++ (FF++)

Settings. When training and evaluating on FF++, we follow the sampling strategy mentioned in [50] that selects 270 frames/video for the training and 110 frames/video for validation and testing. We evaluated both medium compression (c23) and high compression levels (c40) subsets.

Ablation study, c40. Table 1a shows the ablation study for different building blocks of the proposed pipeline, along with the proposed loss function. These results are reported without the LSTM thereby evaluating only the backbone without stratified sampling. Given that D=1024, the two radii are set $r^{\{-,+\}} \doteq \{2.5, 97.5\}$. We report metrics such as TAR and tAUC at a given FAR value. The cut-off FAR points considered are 1% and 10%. At the top, we report results from [52] re-implementing the method without sequential modeling, thereby using just the DenseNet encoder. All methods evaluated in this table consider only the Φ_{RGB} stream and are trained with a global learning rate

[2] The dimensionality is doubled since the results of the bi-directional streams are concatenated.

	Deepfakes `c40` - wo/ LSTM		
	$\text{tAUC}_{1\%}$	$\text{tAUC}_{10\%}$	$\text{TAR}_{1\%}$
Encoder [52]	57.63	86.61	81.50
+ft,+drop,+loss	76.04	89.91	92.84

(a)

	Deepfakes `c23` - wo/ LSTM		
	$\text{tAUC}_{1\%}$	$\text{tAUC}_{10\%}$	$\text{TAR}_{1\%}$
Single-Branch	56.78	61.14	99.34
Two-Branch	61.70	70.80	98.34

(b)

		Deepfakes `c40` - w/ LSTM			
LSTM hid. nodes	LSTM fusion	$\Phi_{\text{RGB}} \circ \Phi_{\text{LoG}}$	f.t. +dropout	$\text{tAUC}_{1\%}$	$\text{TAR}_{1\%}$
128	`cat`	$\texttt{conv1x1}_{g=1}$	—	57.21	83.33
128	`cat`	$\texttt{conv1x1}_{g=1}$	✓	73.58	87.38
128	`sum`	$\texttt{conv1x1}_{g=2}$	✓	67.49	83.81
256	`cat`	$\texttt{conv1x1}_{g=2}$	✓	76.35	87.14
128	`cat`	$\texttt{conv1x1}_{g=2}$	✓	81.53	92.54

(c)

Table 1: **Ablation study on FF++.** (a) Testing metrics obtained by training our model under different settings on the FF++ [50] under the highest compression level `c40` without LSTM, ablating our optimization and the loss function. (b) Ablation experiments showing the impact of the two branches under the medium compression level `c23`. (c) Ablation experiments using LSTM on `c40`.

	HQ (`c23`)								LQ (`c40`)							
	Frame Level (~70K samples)				Video Level (700 samples)				Frame Level (~70K samples)				Video Level (700 samples)			
Methods	AUC	$\text{pAUC}_{10\%}$	$\text{tAUC}_{10\%}$	$\text{TAR}_{10\%}$	AUC	$\text{pAUC}_{10\%}$	$\text{tAUC}_{10\%}$	$\text{TAR}_{10\%}$	AUC	$\text{pAUC}_{10\%}$	$\text{tAUC}_{10\%}$	$\text{TAR}_{10\%}$	AUC	$\text{pAUC}_{10\%}$	$\text{tAUC}_{10\%}$	$\text{TAR}_{10\%}$
DSP-FWA [37]	56.89	51.33	7.47	14.60	57.49	51.59	7.48	15.00	59.15	52.04	8.82	17.30	62.34	51.93	9.82	22.14
Xception [50]	92.30	87.71	**73.34**	81.21	92.50	89.20	58.21	82.85	83.93	**74.78**	**45.92**	**63.25**	86.75	**79.10**	39.06	68.75
Ours	**98.70**	**97.43**	65.29	**97.95**	**99.12**	**98.41**	**86.10**	**98.21**	**86.59**	69.71	40.41	62.48	**91.10**	76.57	**51.18**	**72.85**

Table 2: **Frame-level and Video-level comparison on FF++.** Multiple metrics reported for medium compression (c23) and high compression (c40) on FF++ comparing our method with XceptionNet [50] and DSP-FWA [37]. Results are reported on four manipulations.

of 1e-04. Although [52] reaches compelling result at $\text{tAUC}_{10\%}$, the performance degrades at lower FAR. Better performance is obtained by combining minor improvements such as training the network with the optimization mentioned in Section 3.1 that assigns a different updating rate per layer (*+ft*) and dropout (*+drop*) and by using our new loss function (*+loss*) proposed in Section 3.3. The gain at low false alarm rate is substantial compared to [52] that was trained with binary cross-entropy. In particular, the loss manages to push $\text{tAUC}_{1\%}$ up from 57% to 76% by imposing a large margin—see Fig. 4d—while the regular cross-entropy overfits quickly.

Ablation study, c23. Given the best results obtained in the previous experiment (i.e., *+ft,+drop,+loss*), we use this configuration as a new baseline to perform other ablations using the FF++ part with medium compression (c23). Table 1b reports experiments showing the difference between a single branch and the two-branch structure.

Ablation study using LSTM, c40. Table 1c shows the ablation experiments when testing the recurrent model. Since adding a recurrent modeling is a drastic change, we verified again that our optimization strategy with different updating rates per layer holds in this case as well. The first two rows in the table support this hypothesis. We further investigate how to fuse the bi-directional outputs from LSTM and optimize its hidden nodes. Our best result is obtained using hidden node size of 128 and concatenating the two bi-directional outputs. Fur-

	Frame Level				Video Level			
Methods	AUC	$\text{pAUC}_{10\%}$	$\text{tAUC}_{10\%}$	$\text{TAR}_{10\%}$	AUC	$\text{pAUC}_{10\%}$	$\text{tAUC}_{10\%}$	$\text{TAR}_{10\%}$
Xception-c40 [50]	65.86	54.49	12.23	22.97	69.70	57.18	16.85	34.70
DSP-FWA [37]	64.13	52.87	10.18	19.67	69.30	51.40	17.20	32.02
Xception-c23 [50]	66.65	53.05	10.21	19.83	73.04	52.77	9.45	18.82
Ours	**73.41**	**57.42**	**18.18**	**32.22**	**76.65**	**58.70**	**19.73**	**39.70**

(a)

Method	FF++ [50]	Celeb-DF [38]
Two-stream [23]	70.1	53.8
Meso4 [6]	84.7	54.8
MesoInception4	83.0	53.6
HeadPose [67]	47.3	54.6
FWA [37]	80.1	56.9
VA-MLP [41]	66.4	55.0
VA-LogReg	78.0	55.1
Xception-raw [50]	**99.7**	48.2
Xception-c23	**99.7**	65.3
Xception-c40	95.5	65.5
Multi-task [44]	76.3	54.3
Capsule [45]	96.6	57.5
DSP-FWA [37]	93.0	64.6
Ours	93.18	**73.41**

(b)

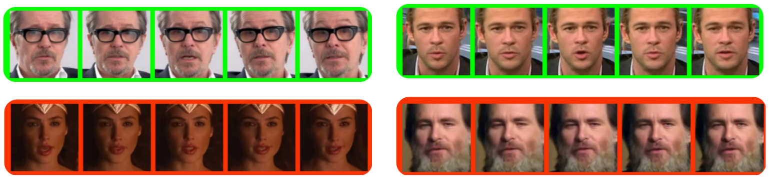
Table 3: **Cross-dataset evaluation on Celeb-DF.** (a) Frame- and video-level performance yet computed at a very low false alarm rate. Best competing methods on Celeb-DF are reported. Ours obtains a wide margin in all the low false alarm rate metrics (b) still performs well when tested on just deepfake class (93.18 %) AUC on FF++. Results for other methods are from [38].

thermore, when fusing the two branches $\Phi_{\text{RGB}} \circ \Phi_{\text{LoG}}$ having convolutional filters divided in two groups is beneficial to the performance.

Results. Table 2 shows a thorough comparison on FF++ [50] training and testing with four manipulations types (Deepfakes, FaceSwap, Face2Face, and NeuralTextures) along with the natural faces. Following [50], we trained a model for `c23` and another for `c40`. The table offers multiple evaluations metrics such as AUC, $\text{pAUC}_{10\%}$, $\text{tAUC}_{10\%}$ and $\text{TAR}_{10\%}$. In general, our approach has superior performance compared to Xception. In particular, we improved almost all frame-level performance for the medium compression case (c23), pushing the video-level AUC from 92% to 99%. The result is consistent for the other compression level but in general results are lower due to the low image quality; nevertheless our system improves video-level AUC from 86% to 91% along with other low false alarm video-level metrics. The table also reports the result of a self-supervised method DSP-FWA [37]. Table 4a further shows the binary classification accuracies for several state-of-the-art face manipulation detection methods computed on FF++ [50]. Our approach scores the highest accuracies across manipulations for all the compression levels when trained on the four manipulations. It should be noted that a classifier exploiting the class imbalance here can get an accuracy of 80% by simply predicting all samples as fakes given that we have 140 real and 560 fake videos or similar balance at the frame level.

4.2 Celeb-DF

Results. We evaluate how well our model transfers to Celeb-DF given that it is trained on FF++ with multiple manipulations. We do this with the goal of confirming that we optimized our method for better generalization across datasets, reaching a good balance between bias and variance. Table 3a shows a state-of-the-art evaluation at the frame- and video-level on the 518 test video of Celeb-DF, comparing it to other recent methods. Like other methods [50],



(a) Natural Sequences (b) Manipulated Sequences

Fig. 5: **Qualitative analysis on Celeb-DF**. The color indicates correct classification (green) or misclassification (red).

we trained the model on FF++ to discern real faces versus four manipulation types at the c40 compression level. Table 3a reports a clear net improvement over the state-of-the-art, even when compared with recent methods that trained the model with self-supervision thereby, in theory, being less prone to overfitting, such as DSP-FWA [37]. Table 3b offers instead the classic evaluation performance in terms of AUC comparing our approach to the very recent method for digital face manipulation detection. We obtained higher AUC when compared to all the other methods on Celeb-DF while keeping an high AUC on FF++ on Deepfakes. **Qualitative analysis.** Fig. 5 shows a qualitative analysis performed on the challenging Celeb-DF [38]. Fig. 5a shows untampered faces. The method correctly classifies a sequence with good quality although we used FF++ with high compression level (c40) for training. Failure case for the natural faces may be caused by the poor illumination. Fig. 5b shows manipulated faces and the method was able to detect a challenging sequence that could be perceived "as real"; the other failure may be due to the presence of strong facial hairs which could be absent in training data.

4.3 The Deepfake Detection Challenge (DFDC) Preview Dataset

We report video-level results on the "The Deepfake Detection Challenge (DFDC) preview set" using the evaluation described in [14]. This dataset contains approximately 5,250 videos of digitally manipulated and bona fide videos. As in [14], we used part of the training for cross validation for the two parameters available in our approach that are the optimal number of sequences and the distance $\left\|\varPhi(\mathbf{I})-\mathbf{c}\right\|_2$. We implemented five-fold cross-validation (20% of training retained for validation) and selected the best pair of parameters across the folds required to maximize the log-weighted precision, log(wP), with α=100, maintaining the desired level of recall. This procedure was repeated for different cutoff recalls ($\mathrm{R}_{10\%}$, $\mathrm{R}_{50\%}$, $\mathrm{R}_{90\%}$). Although cross validation procedure aims to optimize the two parameters to keep a desired level of recall, meeting the same level of recall is not guaranteed when evaluating on the test set. This procedure simulates what can happen in real scenarios in which a system can be optimized on a validation set and then simply tested in the wild over millions of unlabeled data. For this reason, we report log(wP)@recall on the best validation fold under "valid" and the test set with "test-from-valid" using the parameters from validation. Alternatively, we also searched for the best log(wP) to exactly match the recall value on the test set and report those values under "test". Except for the above parameter selection, our method has not been re-trained on DFDC preview.

Results. Table 4b shows the evaluation results at the video level. Considering our results under "test," our method has slightly worse precision than XceptionNet [50] at $R_{10\%}$. However, if we optimize for high recall ($R_{90\%}$), we obtain a substantial boost in the log(wP), increasing log(wP) from -4.041 to -3.548. Moreover, we notice the following if we evaluate with the best hyper-parameters selected on the validation set our method maintains log(wP) better than other methods (-3.721) with a good recall of 0.943.

Methods	HQ (c23)	LQ (c40)
[50] XceptionNet (Full Image)	74.78	70.52
[18] Steg. Features + SVM	70.97	55.98
[12] Cozzolino *et al.*	78.45	58.69
[8] Bayar and Stamm	82.97	66.84
[49] Rahmouni *et al.*	79.08	61.18
[6] MesoNet	83.10	70.47
[50] XceptionNet	95.73	81.00
Ours	**96.43**	**86.34**

(a)

Method	$R_{10\%}$	$R_{50\%}$	$R_{90\%}$
TamperNet [14]	-2.796@—	-3.864@—	-4.041@—
XceptionNet [50] (Face)	**-1.999@—**	**-3.012@—**	-4.081@—
XceptionNet [50] (Full)	-3.293@—	-3.835@—	-4.081@—
Ours (test)	-2.564@0.100	-3.152@0.501	**-3.548@0.901**
Ours (valid)	-2.311@0.090	-2.481@0.523	-2.678@0.918
Ours (test-from-valid)	-3.386@0.042	-3.433@0.440	-3.721@0.943

(b)

Table 4: **FF++ Accuracies and DFDC Preview Dataset.** (a) Comparison of accuracies on FF++ (b) Video-level log(wP) for various recall rates.

5 Conclusions and Future Work

We presented a method for video-based deepfake detection that uses a recurrent model to process sequences of aligned faces using a two-branch backbone with a loss function to isolate manipulated face sequences. We have shown results that outperform or are on par with state-of-the-art. However, for practical, web-scale applications, there is significant room for improvement at low false alarm rates. In the short term, we plan to measure the impact of data augmentation and the usage of additional external natural faces [43]. In the long term, we plan to augment our model with an explainability mechanism that does not need any pixel-wise supervision for face manipulations.

Acknowledgment. This work is based on research sponsored by the Defense Advanced Research Projects Agency under agreement number FA8750-16-2-0204. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Defense Advanced Research Projects Agency or the U.S. Government. The authors would like to thank E. Sabir and A. Jaiswal for the useful discussions and the anonymous reviewers.

References

1. CNN - business - when seeing is no longer believing inside the pentagon's race against deepfake videos. https://www.cnn.com/interactive/2019/01/business/pentagons-race-against-deepfakes/ 2
2. DeepFaceLab. https://github.com/iperov/DeepFaceLab 2
3. DeepTrace - the antivirus of deepfakes - the state of deepfakes. https://deeptracelabs.com 2
4. ZAO app. https://apps.apple.com/cn/app/zao/ 2
5. MSR Image Recognition Challenge (IRC) at ACM Multimedia 2016 (July 2016) 3
6. Afchar, D., Nozick, V., Yamagishi, J., Echizen, I.: Mesonet: a compact facial video forgery detection network. In: WIFS. pp. 1–7. IEEE (2018) 4, 12, 14
7. Agarwal, S., Farid, H., Gu, Y., He, M., Nagano, K., Li, H.: Protecting world leaders against deep fakes. In: CVPR Workshops (June 2019) 4
8. Bayar, B., Stamm, M.C.: A deep learning approach to universal image manipulation detection using a new convolutional layer. In: ACM Workshop on Information Hiding and Multimedia Security. pp. 5–10 (2016) 14
9. Bulat, A., Tzimiropoulos, G.: How far are we from solving the 2d & 3d face alignment problem?(and a dataset of 230,000 3d facial landmarks). In: ICCV (2017) 10
10. Burt, P., Adelson, E.: The laplacian pyramid as a compact image code. IEEE Transactions on communications **31**(4), 532–540 (1983) 2, 5, 7
11. Chollet, F.: Xception: Deep Learning With Depthwise Separable Convolutions. In: CVPR. pp. 1251–1258 (2017), http://openaccess.thecvf.com/content_cvpr_2017/html/Chollet_Xception_Deep_Learning_CVPR_2017_paper.html 4
12. Cozzolino, D., Poggi, G., Verdoliva, L.: Recasting residual-based local descriptors as convolutional neural networks: an application to image forgery detection. In: ACM Workshop on Information Hiding and Multimedia Security. pp. 159–164 (2017) 14
13. Cozzolino, D., Thies, J., Rössler, A., Riess, C., Nießner, M., Verdoliva, L.: Forensictransfer: Weakly-supervised domain adaptation for forgery detection. arXiv preprint arXiv:1812.02510 (2018) 4, 7
14. Dolhansky, B., Howes, R., Pflaum, B., Baram, N., Ferrer, C.C.: The Deepfake Detection Challenge (DFDC) Preview Dataset. arXiv:1910.08854 [cs] (Oct 2019), http://arxiv.org/abs/1910.08854, arXiv: 1910.08854 3, 4, 9, 13, 14
15. Domingos, P.M.: A few useful things to know about machine learning. Commun. acm **55**(10), 78–87 (2012) 3
16. Dufour, N., Gully, A., Karlsson, P., Vorbyov, A.V., Leung, T., Childs, J., Bregler, C.: Deepfakes detection dataset by Google and Jigsaw (2019) 3
17. Farid, H.: Photo forensics. MIT Press (2016) 4
18. Fridrich, J., Kodovsky, J.: Rich models for steganalysis of digital images. TIFS **7**(3), 868–882 (2012) 14
19. Gellately, R.: Lenin, Stalin, and Hitler: The age of social catastrophe. Alfred a Knopf Incorporated (2007) 2
20. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: NIPS (2014) 2
21. Güera, D., Delp, E.J.: Deepfake video detection using recurrent neural networks. In: AVSS. pp. 1–6. IEEE (2018) 2, 3
22. Guo, Y., Zhang, L., Hu, Y., He, X., Gao, J.: Ms-celeb-1m: A dataset and benchmark for large-scale face recognition. In: ECCV (2016) 3

23. Han, X., Morariu, V., Larry Davis, P.I., et al.: Two-stream neural networks for tampered face detection. In: CVPR Workshops. pp. 19–27 (2017) 3, 4, 12
24. Heller, S., Rossetto, L., Schuldt, H.: The PS-Battles Dataset – an Image Collection for Image Manipulation Detection. CoRR **abs/1804.04866** (2018), http://arxiv.org/abs/1804.04866 2, 3
25. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: CVPR. pp. 4700–4708 (2017) 2, 5
26. Huang, G.B., Ramesh, M., Berg, T., Learned-Miller, E.: Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Tech. Rep. 07-49, UMass, Amherst (October 2007) 3
27. Ioannou, Y., Robertson, D., Cipolla, R., Criminisi, A.: Deep roots: Improving cnn efficiency with hierarchical filter groups. In: CVPR. pp. 1231–1240 (2017) 6
28. Karras, T., Laine, S., Aila, T.: A style-based generator architecture for generative adversarial networks. In: CVPR. pp. 4401–4410 (2019) 4
29. Kemelmacher-Shlizerman, I., Seitz, S.M., Miller, D., Brossard, E.: The MegaFace benchmark: 1 million faces for recognition at scale. In: CVPR (2016) 3
30. King, D.E.: Dlib-ml: A machine learning toolkit. JMLR **10**, 1755–1758 (2009) 10
31. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114 (2013) 2
32. Klare, B.F., Klein, B., Taborsky, E., Blanton, A., Cheney, J., Allen, K., Grother, P., Mah, A., Burge, M., Jain, A.K.: Pushing the frontiers of unconstrained face detection and recognition: IARPA Janus Benchmark A. In: CVPR. pp. 1931–1939 (2015) 3
33. Korshunov, P., Marcel, S.: Deepfakes: a new threat to face recognition? assessment and detection. arXiv preprint arXiv:1812.08685 (2018) 3, 4, 9
34. Korshunov, P., Marcel, S.: Vulnerability assessment and detection of deepfake videos. In: ICB. Crete, Greece (Jun 2019) 3
35. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS. pp. 1097–1105 (2012) 2
36. Li, Y., Chang, M.C., Lyu, S.: In ictu oculi: Exposing ai created fake videos by detecting eye blinking. In: WIFS. pp. 1–7 (2018) 3, 4
37. Li, Y., Lyu, S.: Exposing deepfake videos by detecting face warping artifacts. In: CVPR Workshops (June 2019) 4, 11, 12, 13
38. Li, Y., Yang, X., Sun, P., Qi, H., Lyu, S.: Celeb-df: A large-scale challenging dataset for deepfake forensics. In: CVPR (June 2020) 3, 4, 9, 12, 13
39. Maaten, L.v.d., Hinton, G.: Visualizing data using t-sne. Journal of machine learning research **9**(Nov), 2579–2605 (2008) 8, 9
40. Marra, F., Gragnaniello, D., Verdoliva, L., Poggi, G.: Do gans leave artificial fingerprints? In: Conference on Multimedia Information Processing and Retrieval (MIPR). pp. 506–511 (2019) 4
41. Matern, F., Riess, C., Stamminger, M.: Exploiting visual artifacts to expose deepfakes and face manipulations. In: WACV Workshops. pp. 83–92. IEEE (2019) 4, 12
42. McClish, D.K.: Analyzing a portion of the roc curve. Medical Decision Making **9**(3), 190–195 (1989) 3, 9
43. Nagrani, A., Chung, J.S., Xie, W., Zisserman, A.: Voxceleb: Large-scale speaker verification in the wild. Computer Speech & Language **60**, 101027 (2020) 14
44. Nguyen, H.H., Fang, F., Yamagishi, J., Echizen, I.: Multi-task learning for detecting and segmenting manipulated facial images and videos. In: BTAS (2019) 4, 12

45. Nguyen, H.H., Yamagishi, J., Echizen, I.: Capsule-forensics: Using capsule networks to detect forged images and videos. In: ICASSSP. pp. 2307–2311. IEEE (2019) 4, 12
46. Nguyen, T.T., Nguyen, C.M., Nguyen, D.T., Nguyen, D.T., Nahavandi, S.: Deep learning for deepfakes creation and detection. arXiv preprint arXiv:1909.11573 (2019) 4
47. Nirkin, Y., Masi, I., Tran, A., Hassner, T., Medioni, G.: On face segmentation, face swapping, and face perception. In: AFGR (2018) 2
48. Pedro, D.: A unified bias-variance decomposition and its applications. In: 17th International Conference on Machine Learning. pp. 231–238 (2000) 3
49. Rahmouni, N., Nozick, V., Yamagishi, J., Echizen, I.: Distinguishing computer graphics from natural images using convolution neural networks. In: WIFS. pp. 1–6 (2017) 14
50. Rössler, A., Cozzolino, D., Verdoliva, L., Riess, C., Thies, J., Nießner, M.: Faceforensics++: Learning to detect manipulated facial images. In: ICCV (2019) 3, 4, 5, 7, 9, 10, 11, 12, 14
51. Ruff, L., Vandermeulen, R., Goernitz, N., Deecke, L., Siddiqui, S.A., Binder, A., Müller, E., Kloft, M.: Deep one-class classification. In: ICML. pp. 4393–4402 (2018) 7, 8, 9
52. Sabir, E., Cheng, J., Jaiswal, A., AbdAlmageed, W., Masi, I., Natarajan, P.: Recurrent convolutional strategies for face manipulation detection in videos. In: CVPR Workshops. pp. 80–87 (2019) 3, 5, 7, 10, 11
53. Sanderson, C., Lovell, B.C.: Multi-region probabilistic histograms for robust and scalable identity inference. In: ICB. pp. 199–208. Springer (2009) 3
54. Schroff, F., Kalenichenko, D., Philbin, J.: Facenet: A unified embedding for face recognition and clustering. In: CVPR (2015) 9
55. Stehouwer, J., Dang, H., Liu, F., Liu, X., Jain, A.: On the detection of digital face manipulation. arXiv preprint arXiv:1910.01717 (2019) 3, 4, 9
56. Thies, J., Zollhöfer, M., Nießner, M.: Deferred neural rendering: Image synthesis using neural textures. ACM Transactions on Graphics (TOG) (2019) 3
57. Thies, J., Zollhofer, M., Stamminger, M., Theobalt, C., Nießner, M.: Face2face: Real-time face capture and reenactment of rgb videos. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2387–2395 (2016) 3
58. Tolosana, R., Vera-Rodriguez, R., Fierrez, J., Morales, A., Ortega-Garcia, J.: Deepfakes and beyond: A survey of face manipulation and fake detection. arXiv preprint arXiv:2001.00179 (2020) 4
59. Valentini, G., Dietterich, T.G.: Bias-variance analysis of support vector machines for the development of svm-based ensemble methods. JMLR **5**(Jul), 725–775 (2004) 3
60. Verdoliva, D.C.G.P.L.: Extracting camera-based fingerprints for video forensics (2019) 4
61. Verdoliva, L.: Media forensics and deepfakes: an overview. IEEE Journal of Selected Topics in Signal Processing (2020) 4
62. Wang, F., Cheng, J., Liu, W., Liu, H.: Additive margin softmax for face verification. IEEE Signal Processing Letters **25**(7), 926–930 (2018) 9
63. Weisstein, E.W.: Hypersphere (2002) 10
64. Wen, Y., Zhang, K., Li, Z., Qiao, Y.: A discriminative feature learning approach for deep face recognition. In: ECCV (2016) 9
65. Wu, C.Y., Manmatha, R., Smola, A.J., Krahenbuhl, P.: Sampling matters in deep embedding learning. In: ICCV (Oct 2017) 9

66. Xie, C., Wu, Y., Maaten, L.v.d., Yuille, A.L., He, K.: Feature denoising for improving adversarial robustness. In: CVPR. pp. 501–509 (2019) 7
67. Yang, X., Li, Y., Lyu, S.: Exposing deep fakes using inconsistent head poses. In: ICASSSP. pp. 8261–8265. IEEE (2019) 12
68. Yu, N., Davis, L.S., Fritz, M.: Attributing fake images to GANs: Learning and analyzing GAN fingerprints. In: ICCV. pp. 7556–7566 (2019) 4
69. Zhang, X., Karaman, S., Chang, S.F.: Detecting and simulating artifacts in gan fake images. In: WIFS (2019) 4, 6, 7