A Multi-Layer Capsule-based Forensics Model for Fake Detection of Digital Visual Media

Samar Samir Khalil
Computer Engineering Department, College of Engineering and Technology
Arab Academy for Science, Technology and Maritime Transport
Abu Qir, Alexandria, Egypt, P.O.: 1029
samars@adj.aast.edu

Sherin M. Youssef
Computer Engineering Department, College of Engineering and Technology
Arab Academy for Science, Technology and Maritime Transport
Abu Qir, Alexandria, Egypt, P.O.: 1029
sherin@aast.edu

Sherine Nagy Saleh
Computer Engineering Department, College of Engineering and Technology
Arab Academy for Science, Technology and Maritime Transport
Abu Qir, Alexandria, Egypt, P.O.: 1029
sherine_nagi@aast.edu

***Abstract*— the dangers generated from synthesized multimedia are increasing every day. The creation of the so-called Deepfakes multimedia is vastly evolving, making the detection task harder every day. Researchers and corporations are interested in exploring the technology limits and are coming up with new tools every year to create more robust fake media. In this paper, a new enhanced fake video detection model is introduced addressing many of the face-swapping threats and the low generalization problem. A preprocessing stage is proposed to minimize the noise in the data to enhance their quality. The proposed architecture uses a modified application of capsule neural networks (CapsNet) with an enhanced routing technique. It does not require a lot of training data and generates a small number of training parameters making it fast to build. The model was trained and tested using the DFDC-P dataset and the results have proven that it outperformed other detectors in terms of detection recall, weighted precision, and F1 score.**

Keywords— deepfake detection - capsule network – capsnet

I. INTRODUCTION

Fake media are threatening weapons that have been around for years yet recently witnessed a strong evolution due to the development of Deepfakes technology. Deepfakes apply deep learning techniques to create extremely realistic synthesized audios, images or videos. The harm that can be generated by such a tool can affect individual's or nation's security. Audio production can place words to the voices of people who didn't say them, fake image creation can produce images of people in places they have never been to and fake video creations can forge victim's actions and words [1]. People can be blackmailed because one has perfectly stitched his/her face onto a pornographic video, Fig. 1 shows different examples of real and swapped fake faces. A political candidate can be defamed by a fake video of him giving a hate speech, racial slurs and epithets that undercut his image as being pro minorities. An innocent can be framed in a murder case by sewing his face on the criminal's making media forensics no longer veritable. A marketing campaign can be synthesized to drive the public opinion eventually leading customers to lose faith in everything they see or hear.

The word deepfake was first announced late 2017 by a Reddit user who managed to plant the photos of famous actresses within porn videos [2]. Being initiated by amateurs, deepfakes were easily identified at the beginning until researchers decided to take part in creating them [3] and produced media that is more challenging to classify as real or

Fig. 1 Faces taken from the used dataset where the 1st and 3rd rows are real while 2nd and 4th are their corresponding fake.

fake for both humans and computers. Deepfakes can actually have beneficial uses such as helping Amyotrophic Lateral Sclerosis (ALS) patients simulate their voices [4], marketing campaigns target larger audience by using different languages [5] and bring actors who have passed away back to filming [6] but of course these are incomparable with their harmful uses.

In this research, a new model is proposed to aid in the detection of deepfake videos and it outperformed other state of the art models. A review of some new creation and detection methods can be found in section II. Section III explains the proposed model. The experimental work and results are presented in section IV.

II. RELATED WORK

In this section, an overview of deepfake creation and detection methodologies is introduced.

A. Deepfake Creation

Deepfake creation involves the use of Generative Adversarial Networks (GANs) or autoencoders (AEs) to

978-1-7281-6535-6/20/$31.00 ©2020 IEEE

produce synthesized productions of images and videos. The creation of such videos is divided into three different types: Face-swap, Lip-Sync and puppet-master [7]. Face swapping is achieved by replacing the face of a certain person who could be involved in an inappropriate action with that of another targeted person. Lip syncing manages to change the spoken words by manipulating lips movements and mapping voice to words that have never been physically said by the victim [8]. Puppet master shows the strong evolution of artificial intelligence methods where it uses only a single portrait image of a target person to generate video stream of facial expressions and lip syncing [9].

Multiple researches have previously tackled the creation of deepfakes such as Face2Face [10], FaceSwap [11], DeepFakes [12] and NeuralTextures [13]. Face2Face is a real-time expressions transfer system that can efficiently generate an immediate manipulation of facial movements in any video. They use both 3D model reconstruction and image-based rendering algorithms to create the output. The same techniques can be also applied in Virtual Reality jointly with eye-tracking and reenactment [14] or be extended to the full body [15].

FaceSwap [11] is the classical method for face swapping where a source's facial landmarks are extracted from multiple images then trained on a 3D template model and back-projected onto a target face by minimizing the distance between the projected shape and the locale landmarks.

DeepFakes [12] is another technique that also generates face-swapped synthetics but using AEs, which outputs a reconstruction of the input by using two different neural networks, an encoder which learns the latent features of the input and maps them to another neural network that is a decoder which reconstructs the input from latent features. The loss between original input and the reconstructed one is then computed and the networks shared weights are updated. The DeepFakes technique has two stages, training and converting. In training, both source and target images are trained on the same encoder, the output is then fed to two different decoders trying to reconstruct each face. As for the converting stage, the target person image encoding is fed to the source's decoder thus producing the forged face.

NeuralTextures [13] is an example of using GANs for face replacement. In this technique, a generative model is trained to learn the neural texture of a target person using original video data. The GANs objective is a combination of adversarial and photometric reconstruction loss.

Recently StyleGAN2 [16] and Face shifter [17] were introduced. StyleGAN2 is an Nvidia creation in which a team of researchers have managed to improve the quality of their deepfake created images by redesigning the generator's normalization and applying regularization techniques to overcome the artifacts that existed in StyleGAN [3]. It managed to mimic the style from one image to the other, this is known as style transfer architecture, producing fascinating results [18]. Face shifter [17] is a subject agonistic two-stage system. The first stage is a GAN-based network, known as Adaptive Embedding Integration Network (AEI-Net), which could extract target attributes and adjustably learns where to integrate them or identity embeddings. The second stage involves training a Heuristic Error Acknowledging Refinement Network (HEAR-Net) to recover anomaly regions in a self-supervised way without any manual annotations making the system occlusion aware.

B. Deepfake Detection

Previous detection methods are split into two categories: the first is based on observations of some artifacts left by creation techniques and the other uses deep learning to learn a function that helps differentiate fake from real within the data itself.

In the first approach, Li et al [19] noticed that the eye blinking frequency is less than its normal rate in early created deepfake videos thus they used a Long-term Recurrent Convolutional Network (LRCN) to learn the temporal pattern of blinking. Another contribution made by Li et al [20] was applying convolutional neural networks CNNS for fake detection as they observed that deepfakes creation algorithms made at that time could only generate low resolution images that needed to be wrapped (scaled, rotated, …etc) to match the pristine face to be easily spotted. They trained four CNN models VGG16, ResNet50, ResNet101 and ResNet152. Yang et al [21] built a detector based on spotting inconsistencies of 3D head poses using Support Vector Machine (SVM), [22] also used visual artifacts based model for deepfake detection. In [1], Agarwal et al came up with a technique that combines a facial recognition based static biometric with a head movement and facial recognition temporal, behavioral biometric that is learnt using a CNN.

The problem with the first deep fake detection approach is that in most cases it was based on researchers' observations such as the rate of an eye blink or the existence of artifacts in face wrapping. These creation flaws were eventually overcome in the more recent creation tools, for example, creators collected more images of the target to produce an imitation of the his eye blinking. The methods applied in this approach used simple neural architectures that won't be enough to tackle the vastly developing creation algorithms.

The second approach used different deep learning architectures as classifiers to extract the salient and discriminative features. In [23], Sabir et al exploited the video characteristics and used a recurrent neural network (RNN) to learn the temporal differences across its frames. The system achieved state-of-the-art results on the FaceForensics (FF) dataset [24]. Guera et al [25] used an LSTM that takes features extracted by a CNN as input to construct the sequence descriptors useful for classification. Zhou et al [26] applied a two-stream neural networks, one was a GoogleLeNet for detecting the face artifacts and the other was a patch based triplet network to leverage features

capturing local noise residuals. Masi et al [27] also used a two-branch based recurrent network, one working on amplifying the multi-band frequencies using a Laplacian of Gaussian (LoG) as bottleneck layer for the model which our proposed model out performed on the DFDC-P dataset. Afchar et al [28] applied his own CNN based network that is interested in only DeepFake and Face2Face manipulations. Kumar et al [29] proposed a multi-stream network consisting of five ResNet-18 to detect Face2Face manipulations and achieved very high results on the FF++ [30] dataset.

Capsule Network was first used by [31] to tackle deepfakes detection. The team used VGG-19 to extract latent features from the preprocessed faces and feed them to the capsule network. The capsule network consists of 3 primary capsules and 2 output capsules, one for each class. The system was tested on FF dataset and achieved high accuracy among other models.

In both deepfake detection approaches, some publications created their own datasets due to the lack of benchmarks which makes comparison to their work not possible such as [21] [25] [26] and others only reported high accuracy using FF++ dataset which already has a very high detection rate of 82.3% [32] compared to the other available datasets.

III. Proposed Model

Fig. 2 shows the proposed multi-layer system model diagram, the following sections will explain each stage.

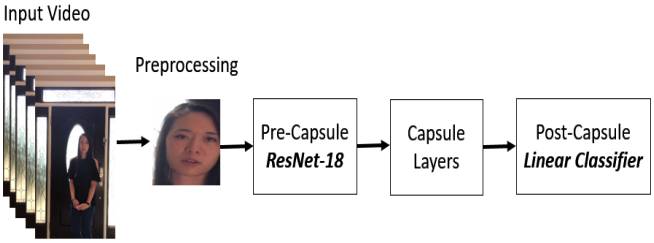

Fig. 2. Proposed Multi-Layer System Model Diagram

A. Preprocessing

The first stage takes an input video, cuts it to frames, extracts all faces from all of the video frames and resizes it to 32x32 pixels. After resizing the faces we apply histogram equalization to improve the contrast of the images and thus enhance data quality.

The used algorithm for face detection is YOLO v3 (You Only Look Once) [33], the experiments have shown that it is more accurate and much faster than MTCNN [34] which was used by DFDC winners. Fig. 3 shows some patterns that, despite setting the MTCNN threshold to 0.9 to limit the noise, were predicted to be faces with a probability reaching 0.99. These faces did not appear in YOLO after setting the threshold to 0.9 as it has a state-of-the-art accuracy in real-time object detection and reduced the noise data introduced by face extraction stage making us avoid the need for an extra data cleaning stage. YOLO is a fully convolutional network and its eventual output is generated by applying a 1 x 1 kernel on a feature map. It models detection as a regression problem and divides the frame into an S x S grid. If the center of the face falls into a grid cell, that grid cell is responsible for detecting it. Each grid cell predicts B bounding boxes and confidence scores for those boxes. These confidence scores reflect how confident the model is that the box contains a face and also how accurate it thinks the box is. Each bounding box consists of five predictions: x, y, w, h, and confidence. The (x, y) coordinates represent the center of the box relative to the bounds of the grid cell. Apart from the confidence the other four predictions are used to extract the face.

B. Pre-Capsule

After the face is extracted from the video, feature extraction is needed to capture all the fine details. Previous research on capsule neural networks have used a convolutional layer to create the feature maps for it. In this work, we propose the application of ResNet-18 [35], which is a convolutional neural network having 18 layers and can be trained to extract important features.

C. Capsule Layers

Fig. 4 explains the proposed multi-layer capsule architecture. First introduced by Hinton et al [36] a capsule is a group of neurons that encodes the characteristics of a visual entity. Embedding all the information of a part in one computational unit makes it easier to derive a part-whole relation, for example, if we have an image of a human face, the lower levels capsules will embed the face components' data (eye, nose, , etc.) as well as their positions relative to each other. Each one of those low-level capsules (child) tries to find the right path to a higher-level (parent) that could contain its specifications, it does so by using a routing algorithm. Routing is mutually exclusive among parents meaning that each child can belong to a single parent. As a further elaboration, if we extended the image to be of a human body, the eye capsule chooses whether it agrees with (belongs to) the face capsule or the arm capsule.

In the proposed model, we used the inverted dot-product attention routing algorithm [37] described in Procedure 1. It maps children i in layer L p_i^L to parents j in layer L+1 p_j^{L+1}. Each child forms a vote for each parent using the unique learned transformation matrix (weights) between them W_{ij}^L. Using dot-product, each parent calculates the similarity between it and all children using their votes, a routing coefficient between 0 and 1 is calculated for all children. Each parent then updates their values using both votes and agreements, the child capsule that contributes most to one parent is eventually routed to it. A Layer Normalization [38] is applied to improve the convergence of the routing. The

Fig. 3. Sample MTCNN Output for crowded images

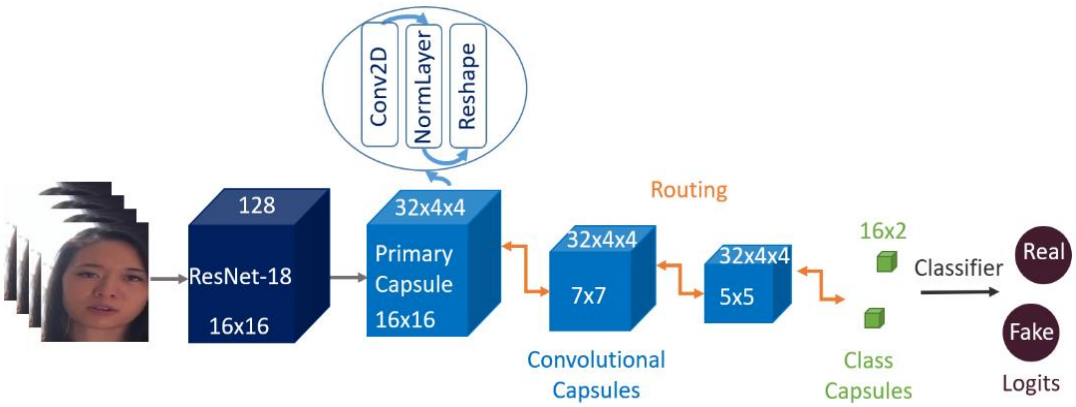

Fig. 4. Multi-Layer System Architecture

algorithm is called inverted as parents are the ones trying to earn the children votes as their new values and acceptance depend on the ones reported.

Procedure 1 Inverted Dot-product Attention Routing algorithm returns updated poses of the capsules in layer L + 1 given poses in layer L and L + 1 and weights between layer L and L + 1

Procedure INVERTED DOT-PRODUCT ATTENTION
ROUTING(p^L, p^{L+1},W^L)
  for all capsule i in layer L and capsule j in layer(L+1) — vote
    $v_{ij}^L \leftarrow w_{ij}^L \cdot p_i^L$
  for all capsule i in layer L and capsule j in layer(L+1) — agreement
    $a_{ij}^L \leftarrow p_j^{L+1^T} \cdot v_{ij}^L$
  for all capsule i in layer L — routing coefficient
    $r_{ij}^L \leftarrow \exp(a_{ij}^L) / \sum_{j'} \exp(a_{ij'}^L)$
  for all capsule j in layer (L + 1): $p_j^{L+1} \leftarrow \sum_i r_{ij}^L v_{ij}^L$ — pose update
  for all capsule j in layer (L + 1) — normalization
    $p_j^{L+1} \leftarrow$ LayerNorm(p_j^{L+1})
  Return P^{L+1}

For the learning phase, weights (transformation matrices) are updated by stochastic gradient descent using cross entropy loss function.

Inference is demonstrated using Procedure 2 and the system model in fig. 4 where the first part represents the backbone convolutional network to which an image is input along with the learnt system parameters for feature extraction. The output of the backbone is used to feed the primary capsule layer, the base layer that creates the first low-level capsules and is used to update the parent convolutional capsule which in turn is used to update its parent and so forth. The primary capsule is applied by using a convolutional layer that is normalized and reshaped for making a matrix-capsule of size $\mathbb{R}^{\sqrt{d_L} x \sqrt{d_L}}$ where d_L is the number of hidden layers grouped together to make a capsule.

As discussed in Procedure 1, to compute the value of any capsule layer, apart from the primary, its child capsule value must be computed which means that the first routing iteration must be sequential. Starting from the second routing iteration to the last, the routing is concurrent which resulted in an enhanced training performance.

The last capsule layers are the class capsules that are used to obtain the predicted class logits.

D. Post-Capsule

A linear classifier is used to the class capsules output to get the prediction logits, this classifier is shared among the class capsules

Procedure 2 Inference. Inference returns class logits given input images and parameters for the model. Capsule layer 1 denotes the primary capsules layer and layer N denotes the class capsules layer.

procedure INFERENCE(I; Θ,$W^{1:N-1}$)
  /* Pre-Capsules Layers: backbone features extraction */
  F ← backbone(I, Θ)
  /* Capsules Layers: initialization */
  $P^1 \leftarrow$ LayerNorm(convolution(F; Θ))
  for L in layers 2 to N: $P^L \leftarrow 0$
  /* Capsules Layers (1st Iteration): sequential routing */
  for L in layers 1 to $(N-1)$ do
    $P^{L+1} \leftarrow$ Routing(P^L, P^{L+1}; W^L)
  /* Capsules Layers (2nd to tth Iteration): concurrent routing */
  for $(t-1)$ iterations do
    for L in layers a to N
      $\bar{P}^{L+1} \leftarrow$ Routing(P^L, P^{L+1}; W^L)
    for in layers 2 to N: $P^L \leftarrow \bar{P}^L$
  /* Post-Capsules Layers: classification */
  for all capsule i in layer N
    $\hat{y}_i \leftarrow$ classifier(p_i^N; Θ)
  return $\hat{y}$

IV. Experimental Work

In this section, the details of experimenting the proposed model will be presented along with the results obtained on an Nvidia GPU GeForce RTX 2070 with Max-Q Design.

A. Dataset

The model presented here was applied on the DeepFakeDetectionChallenge Preview (DFDC-P) dataset [39]. This dataset was formulated by Facebook using two different deepfake creation techniques making it challenging to produce a model that generalizes to both of them. The data contains 5253 videos having 66 actors. The data is split into two categories, training (4473 videos, 40 actors) and testing (780 videos, 26 actors). The training data contains 3618 fake and 855 real videos, and the testing data has 504 fake and 276 real videos. All videos are either 10 or 15 seconds long, some were randomly selected to reduce their FPS to 15 while the

rest remained 30 FPS. Fig. 1 shows different fake and real face samples extracted from the dataset.

In the proposed model, training data was split based on actors where the training set is formulated using 30 actors and the validation set using the remaining 10 actors. The training set contains 3515 videos, 715 of which are real and the other 2800 are fake. The validation set has 958 videos. 140 of which are real and the remaining 818 are fake.

B. Evaluation metrics

As reported by the DFDC-P Dataset [39] two measures are calculated based on video-level results.

1) **Weighted Precision (wP):** since the dataset is imbalanced, they introduced a new constant α to the ordinary precision measure. Assuming the ratio between deepfake and pristine videos is 1: x in organic traffic and 1: y in a deepfakes dataset, it is likely that x >> y. They defined α = x/y to be the factor by which the ratios of pristine to fake videos differ between a test dataset and organic traffic. They also assigned a value of 100 to α and defined wP to be

$$wP = \frac{TP}{TP + \alpha FP}$$

Where, TP is the true positive, i.e how many fake videos the system predicted as fake. FP is the false positive, i.e how many real videos the system predicted as fake. Since the FP is heavily weighted, the wP will be very small so log(wP) is reported in the results, zero is the maximum achievable value.

2) Recall

$$R = \frac{TP}{TP + FN}$$

Where, FN is the false negatives, how many real videos the system predicted as real

We also included the F1 score [40] to better translate the reported dataset results compared to ours.

$$F1 = \frac{2 * R * P}{R + P}$$

C. Results

The testing data was input to the trained network and Table 1 shows our results when compared with other methods using the same dataset.

TABLE 1. VIDEO-LEVEL TEST METRICS WHEN OPTIMIZING FOR LOG(WP)

Method	Precision	Recall	Log(wP)	F1
TamperNet [39]	0.833	0.033	-3.044	0.063
XceptionNet(Face) [41]	**0.930**	0.084	-2.14	0.154
XceptionNet(Full) [41]	0.784	0.268	-3.352	0.399
Ours	0.892	**0.754**	**-1.119**	**0.817**

The table shows, our reported log(wP) is almost double that reported by XceptionNet. Although our precision is slightly lower than the best reported, our recall is 2.8 times larger than the highest recall which means that these systems have a very high false negatives rate and tend to classify all videos as fake. We also computed the F1 score as it represents the weighted average between recall and precision and our results were 2.4 times higher than the best recorded.

TABLE 2. VIDEO-LEVEL LOG(WP) FOR VARIOUS RECALL VALUES

Method	R=0.1	R=0.5	R=0.9
TamperNet [39]	-2.796	-3.864	-4.041
XceptionNet(Face) [41]	-1.999	-3.012	-4.084
XceptionNet(Full) [41]	-3.293	-3.835	-4.084
Masi et al [27]	-2.564	-3.152	-3.548
Ours	-	-	**-1.593**

Table 2 shows the weighted precision values generated when different thresholds are applied to the outcome probabilities. The table showed our model to be more robust as it did not produce low recall values given different thresholds yet it produced much higher log(wP) at recall value 0.9.

Finally, the capsule previous detector implemented in [31] used 2.8M parameters and sequential routing while the proposed system used 1.73M only and concurrent routing making the proposed model easier and faster to train.

V. CONCLUSION AND FUTURE WORK

In this paper, a new multi-phase deep neural network capsule model is presented. The model addressed the previous detection methods shortages of low generalizations and lack of a robust data preprocessing system for noise removal. When compared to other systems, experiments showed outstanding results in terms of sensitivity, weighted precision and F1 score. The model is based on capsule network architecture thus does not need a lot of sample data to learn from and has the least number of learning parameters ~1.73M. A concurrent routing algorithm was also applied, making the presented model faster than its ancestors. Finally deepfake is an arm race as more detection methods should be developed every day to tackle the different creation techniques. In the future, we plan to expand the model such that it can also handle fake audio embeddings, work on improving the results and try to achieve better generalization on unseen data.

VI. REFERENCES

[1] S. Agarwal, T. El-Gaaly, H. Farid and S.-N. Lim, "Detecting Deep-Fake Videos from Appearance and Behavior," *arXiv preprint arXiv:2004.14491*, 2020.

[2] S. Adee, "What Are Deepfakes and How Are They Created?," IEEE Spectrum, 29 April 2020. [Online]. Available: https://spectrum.ieee.org/tech-talk/computing/software/what-are-deepfakes-how-are-they-created. [Accessed 20 August 2020].

[3] T. Karras, S. Laine and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2019.

[4] Project Revoice, "We can stop ALS from stealing more voices," [Online]. Available: https://www.projectrevoice.org/#section-mission. [Accessed 31 August 2020].

[5] Variety, "How AI Tech Is Changing Dubbing, Making Stars Like David Beckham Multilingual," [Online]. Available: https://variety.com/2019/biz/news/ai-dubbing-david-beckham-multilingual-1203309213/. [Accessed 31 August 2020].

[6] J. Alexander, "Furious 7 used 350 CGI shots of Paul Walker," 20 October 2015. [Online]. Available: https://www.polygon.com/search?q=Furious+7+used+350+CGI+shots+of+Paul+Walker. [Accessed 31 August 2020].

[7] H. Kim, P. Garrido, A. Tewari, W. Xu, J. Thies, M. Niessner, P. Pérez, C. Richardt, M. Zollhöfer and C. Theobalt, "Deep video portraits," *ACM Transactions on Graphics (TOG)*, vol. 37, p. 1–14, 2018.

[8] S. Suwajanakorn, S. M. Seitz and I. Kemelmacher-Shlizerman, "Synthesizing obama: learning lip sync from audio," *ACM Transactions on Graphics (TOG)*, vol. 36, p. 1–13, 2017.

[9] H. Averbuch-Elor, D. Cohen-Or, J. Kopf and M. F. Cohen, "Bringing portraits to life," *ACM Transactions on Graphics (TOG)*, vol. 36, p. 1–13, 2017.

[10] J. Thies, M. Zollhofer, M. Stamminger, C. Theobalt and M. Nießner, "Face2face: Real-time face capture and reenactment of rgb videos," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

[11] M. Kowalski, "FaceSwap," [Online]. Available: https://github.com/MarekKowalski/FaceSwap.

[12] Deepfakes, "Deepfakes," [Online]. Available: https://github.com/deepfakes/faceswap.

[13] J. Thies, M. Zollhöfer and M. Nießner, "Deferred neural rendering: Image synthesis using neural textures," *ACM Transactions on Graphics (TOG)*, vol. 38, p. 1–12, 2019.

[14] J. Thies, M. Zollhöfer, M. Stamminger, C. Theobalt and M. Nießner, "FaceVR: Real-time gaze-aware facial reenactment in virtual reality," *ACM Transactions on Graphics (TOG)*, vol. 37, p. 1–15, 2018.

[15] J. Thies, M. Zollhöfer, C. Theobalt, M. Stamminger and M. Niessner, "Headon: Real-time reenactment of human portrait videos," *ACM Transactions on Graphics (TOG)*, vol. 37, p. 1–13, 2018.

[16] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen and T. Aila, "Analyzing and improving the image quality of stylegan," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[17] L. Li, J. Bao, H. Yang, D. Chen and F. Wen, "Faceshifter: Towards high fidelity and occlusion aware face swapping," *arXiv preprint arXiv:1912.13457*, 2019.

[18] K. e. a. a. Nvidia, "This Person Does Not Exist," [Online]. Available: https://thispersondoesnotexist.com/.

[19] Y. Li, M.-C. Chang and S. Lyu, "In ictu oculi: Exposing ai generated fake face videos by detecting eye blinking," *arXiv preprint arXiv:1806.02877*, 2018.

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

[21] X. Yang, Y. Li and S. Lyu, "Exposing deep fakes using inconsistent head poses," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019.

[22] F. Matern, C. Riess and M. Stamminger, "Exploiting visual artifacts to expose deepfakes and face manipulations," in *2019 IEEE Winter Applications of Computer Vision Workshops (WACVW)*, 2019.

[23] E. Sabir, J. Cheng, A. Jaiswal, W. AbdAlmageed, I. Masi and P. Natarajan, "Recurrent convolutional strategies for face manipulation detection in videos," *Interfaces (GUI)*, vol. 3, 2019.

[24] A. Rössler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies and M. Nießner, "Faceforensics: A large-scale video dataset for forgery detection in human faces," *arXiv preprint arXiv:1803.09179*, 2018.

[25] D. Güera and E. J. Delp, "Deepfake video detection using recurrent neural networks," in *2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2018.

[26] P. Zhou, X. Han, V. I. Morariu and L. S. Davis, "Two-stream neural networks for tampered face detection," in *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2017.

[27] I. Masi, A. Killekar, R. M. Mascarenhas, S. P. Gurudatt and W. AbdAlmageed, "Two-branch Recurrent Network for Isolating Deepfakes in Videos," *arXiv preprint arXiv:2008.03412*, 2020.

[28] D. Afchar, V. Nozick, J. Yamagishi and I. Echizen, "Mesonet: a compact facial video forgery detection network," in *2018 IEEE International Workshop on Information Forensics and Security (WIFS)*, 2018.

[29] P. Kumar, M. Vatsa and R. Singh, "Detecting face2face facial reenactment in videos," in *The IEEE Winter Conference on Applications of Computer Vision*, 2020.

[30] A. Rossler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies and M. Nießner, "Faceforensics++: Learning to detect manipulated facial images," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019.

[31] H. H. Nguyen, J. Yamagishi and I. Echizen, "Capsule-forensics: Using capsule networks to detect forged images and videos," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019.

[32] Y. Li, X. Yang, P. Sun, H. Qi and S. Lyu, "Celeb-DF: A Large-scale Challenging Dataset for DeepFake Forensics," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[33] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[34] K. Zhang, Z. Zhang, Z. Li and Y. Qiao, "Joint face detection and alignment using multitask cascaded convolutional networks," *IEEE Signal Processing Letters*, vol. 23, p. 1499–1503, 2016.

[35] K. He, X. Zhang, S. Ren and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

[36] S. Sabour, N. Frosst and G. E. Hinton, "Dynamic routing between capsules," in *Advances in neural information processing systems*, 2017.

[37] Y.-H. H. Tsai, N. Srivastava, H. Goh and R. Salakhutdinov, "Capsules with Inverted Dot-Product Attention Routing," *arXiv preprint arXiv:2002.04764*, 2020.

[38] J. L. Ba, J. R. Kiros and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[39] B. Dolhansky, R. Howes, B. Pflaum, N. Baram and C. C. Ferrer, "The deepfake detection challenge (dfdc) preview dataset," *arXiv preprint arXiv:1910.08854*, 2019.

[40] X. Deng, Q. Liu, Y. Deng and S. Mahadevan, "An improved method to construct basic probability assignment based on the confusion matrix for classification problem," *Information Sciences*, vol. 340, p. 250–261, 2016.

[41] ondyari, "faceforensics," 2019. [Online]. Available: https://github.com/ondyari/FaceForensics/tree/master/classification. [Accessed 22 August 2020].