——SUPPLEMENTARY MATERIAL——
AOT: Appearance Optimal Transport Based Identity Swapping for Forgery Detection

Hao Zhu [1,2,*], **Chaoyou Fu** [1,3,*], **Qianyi Wu** [4], **Wayne Wu** [4], **Chen Qian** [4], **Ran He** [1,3†]
[1] NLPR & CEBSIT & CRIPAC, CASIA [2] Anhui University
[3] University of Chinese Academy of Sciences [4] SenseTime Research
haozhu96@gmail.com {chaoyou.fu,rhe}@nlpr.ia.ac.cn
{wuqianyi,wuwenyan,qianchen}@sensetime.com

Appendix

A Implementation Details

A.1 Data Preparation

The training data only consists of the target faces and the reenacted faces. The target faces are directly extracted from the original FF++ [9] (900 videos) and DPF-1.0 [4] (10 identities). Then, we leverage DFL [8] and FSGAN [6] to produce the reenact faces using identities that are not existed in the target faces. Our quantitative experiments are conducted on the remaining videos in FF++ and DPF-1.0.

Then, we first detect 106 facial landmarks of each video. Then, we crop the face area and resize them into 256*256 resolution. To obtain the PNCC and normals, we use 3DDFA [15] to estimate the 3D mesh of each face, and render the mesh and corresponding PNCC and normals codes to images via a neural renderer [5].

A.2 Training Strategies

We use PyTorch [7] to implement our model. In the training phase, our model is trained with 200K iterations on two NVIDIA1080Ti GPUs, where the batch size = 16. We use Adam optimizer for relighting generator with $\beta 1$ = 0.5, $\beta 2$ = 0.999, weight decay = 0.0002, and RMSprop optimizer for Mix-and-Segment Discriminator (MSD), Ω, Ψ with beta = 0.9. The learning rates of both the Adam and the RMSprop optimizers are set to 0.0002. In $\mathcal{L}_{total}$, we set λ_1=120, λ_2=1, λ_3=90, and λ_4=1. The full training algorithm is summarized here 1.

A.3 Network Architectures

The full architecture as shown in Fig. S1.

*Equal contribution
†Corresponding author

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

Algorithm 1 Training algorithm.

Require: $\{X_r\}^N$, $\{X_t\}^N$;
Require: Initialize Ω_i, Ψ_i, $PerceptualEncoder$, $Decoder$, and MSD with θ_i, ω_i, α, β, γ respectively.

while not converged **do**
Sample mini-batch $\{x_r\}$
Sample mini-batch $\{x_t\}$

// Forward: Encoder
$F^1_{X_r}, F^2_{X_r}, F^3_{X_r}, F^4_{X_r} \leftarrow \text{PERCEPTUALENCODER}(\{x_r\})$
$F^1_{X_t}, F^2_{X_t}, F^3_{X_t}, F^4_{X_t} \leftarrow \text{PERCEPTUALENCODER}(\{x_t\})$

// Update: NOTPE
for $i = 1, 2, 3$ **do**
for $j = 1, ..., n_i$ **do**
Sample $v^j_r \leftarrow F^i_{X_r}$
Sample $v^j_t \leftarrow F^i_{X_t}$
$g_{\omega_i} \leftarrow \nabla_{\omega_i}[\frac{1}{m}\Psi_i(\Omega_i(v^j_r|X_r, X_t)) - \frac{1}{m}\Psi_i(v^j_t)]$
$\omega_i \leftarrow \omega_i + \alpha\cdot$RMSProp($\omega_i$,x)
$\omega_i \leftarrow \text{CLIP}(\omega_i, -c, c)$
end for
end for
for $i = 1, 2, 3$ **do**
for $j = 1, ..., n_i$ **do**
Sample $v^j_r \leftarrow F^s_{X_r}$
$g_{\theta_i} \leftarrow -\nabla_{\theta_i}\frac{1}{m}\Psi_i(\Omega_i(v^j_r|X_r, X_t))$
$\theta_i \leftarrow \theta_i + \alpha\cdot$RMSProp($\theta_i$,x)
end for
end for

// Forward: NOTPE
for $i = 1, 2, 3$ **do**
$F^i_Y \leftarrow \Omega_i(F^i_{X_r}$,$X_r$,$X_t)$
end for

// Forward: Decoder
$Y_{t,t} \leftarrow \text{DECODER}(F^1_{X_r}, F^2_{X_r}, F^3_{X_r}, F^4_{X_r})$
$Y_{r,t} \leftarrow \text{DECODER}(F^1_Y, F^2_Y, F^3_Y, F^4_{X_r})$

// Update: MSD
$M_r \leftarrow$ Random Mask Generator().
$Y_{mix} \leftarrow \text{MIX}(Y_{t,t}, Y_{r,t}, M_r)$
$g_\gamma \leftarrow \nabla_\gamma[\frac{1}{m}][M_r * MSD(Y_{mix})] - \frac{1}{m}[(1 - M_r) * MSD(Y_{mix})]$
$\gamma \leftarrow \gamma + \alpha \cdot ADAM(\gamma)$

// Update: Perceptual Encoder, Decoder
$g_\beta \leftarrow \nabla_\beta[\frac{1}{m}]Loss(Y_{t,t}, Y_{r,t}, X_t)$ ▷ Total Loss
$\beta \leftarrow \beta + \alpha \cdot ADAM(\beta)$
end while

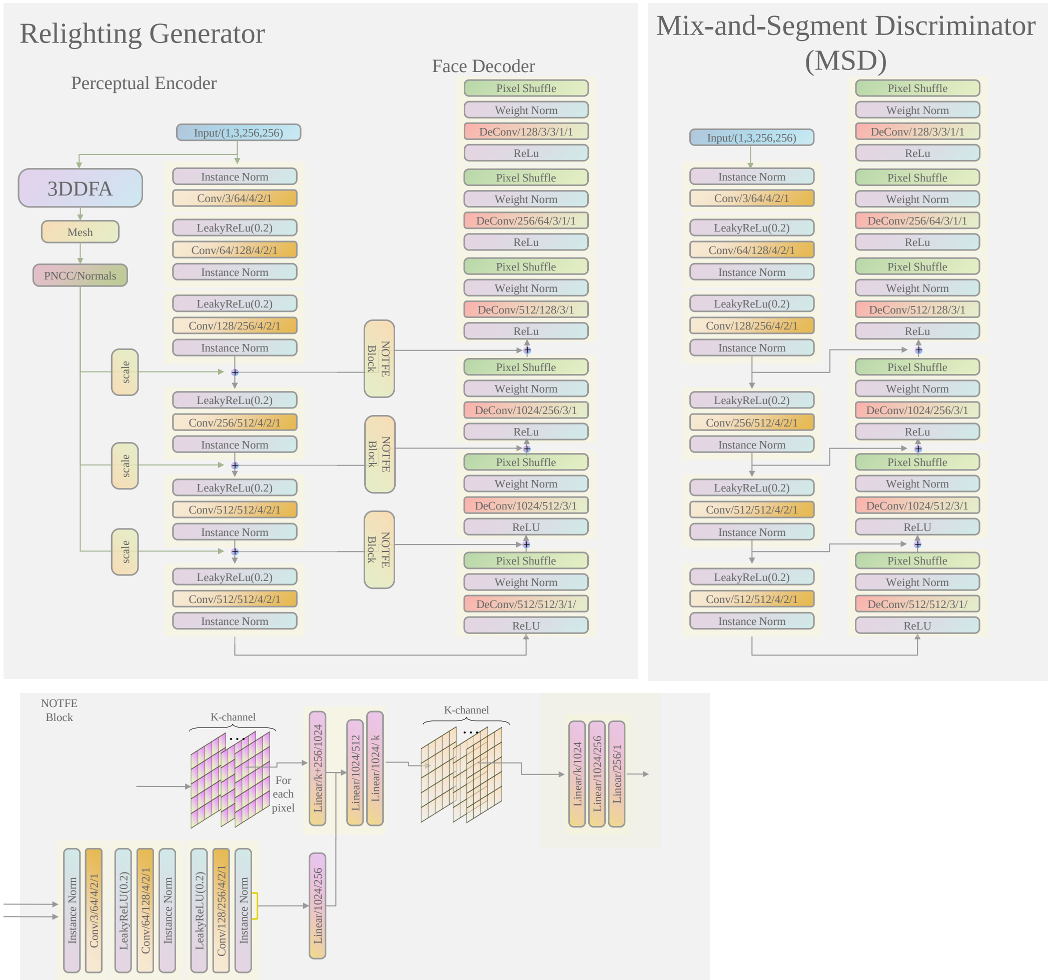

Figure S1: The detailed pipeline of our proposed model.

B Compared Baseline

B.1 Face Swapping Methods

DeepfaceLab. DeepfaceLab (DFL) [8] requires to retrain the model for different source identities. It means we need to train the DFL model different videos respectively. It should be clear that, DFL provides lots of options to tune the results. In practice, we use the options reported in Table S1.

FSGAN. FSGAN [6] is a landmark-guided subject agnostic method. We leverage the latest models provided by authors.

B.2 Appearance Transfer Methods

Poisson Blending. Poisson Blending is a classical image harmonization method. We use the OpenCV implemented version, and set the flag=cv2.NORMAL_CLONE.

Deep Image Harmonization (DIH) [13]. [3] DIH is a deep learning based image harmonization method and it can capture both the context and semantic patterns of the images rather than hand-craft features.

[3] DIH: https://github.com/wasidennis/DeepHarmonization

Table S1: Options of DeepFaceLab.

Training Options				Merging Options	
name	choice	name	choice	name	choice
resolution	224	gan_power	0.0	mask_mode	learned
face_type	f	true_face_power	0.0	erode_mask_modifier	5
models_opt_on_gpu	True	face_style_power	0.0	blur_mask_modifier	5
archi	dfhd	bg_style_power	0.0	motion_blur_power	0
ae_dims	256	ct_mode	None	output_face_scale	1
e_dims	64	clipgrad	False	color_transfer_mode	rct
d_dims	64	pretrain	False	sharpen_mode	none
d_mask_dims	22	autobackup_hour	0	blursharpen_amount	0
masked_training	True	write_preview_history	True	super_resolution_power	1
eyes_prio	False	target_iter	0	image_denoise_power	0
lr_dropout	False	random_flip	True	bicubic_degrade_power	0
random_warp	True	batch_size	4	color_degrade_power	0

Style Transfer for Headshot Portraits (STHP) [10]. [4] STHP allows users to easily produce style transferred results. It transfers multi-scale local statistics of an reference portrait into another.

WCT^2. [5] WCT^2 is a state-one-the-art photorealistic style transfer method. We use the option unpool = 'cat5' version, and the pretrained models.

C Additional Experiments

C.1 Noise Analysis

Furthermore, we verified our results with photo forgery methods: noise analysis, error level analysis, level sweep, luminance gradient [6]. As shown in Fig. S2, ours framework reduces the noises (Fig. S2 (a, b)) and preserves the appearance with target images (Fig. S2 (c, d)).

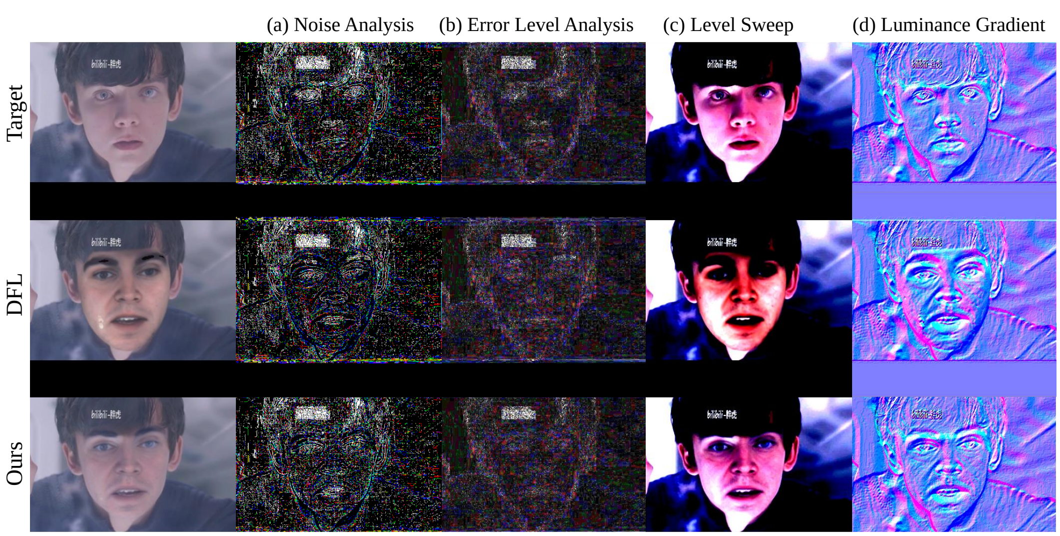

Figure S2: Noise analysis with photo forensics algorithms. Our method can not only reduce the noises (a,b), but also better preserve appearances. (c,d).

[4]STHP: https://people.csail.mit.edu/yichangshih/portrait_web/
[5]WCT^2: https://github.com/clovaai/WCT2
[6]https://29a.ch/photo-forensics/

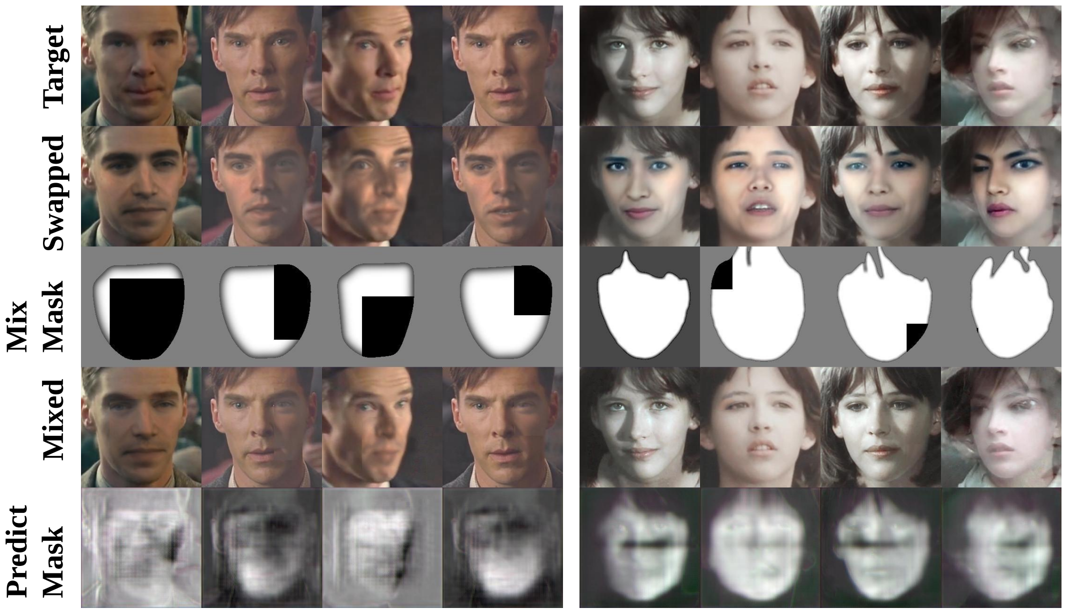

Figure S3: The mixed results.

C.2 Results of Mix-and-Segment Discriminator

We provide more results of the mixed results. As shown in Fig. S3, we mix the target faces and the swapped faces using the mix mask. It is difficult to find the real patch and the fake patch.

C.3 Feature Visualization

To give intuitive results, we visualize the features at different scales by using PCA to reduce the dimensions of them to 3-dimensional vectors.

In the latent space the pixel distributions are more balance under different lighting conditions, as shown in Fig. S4.

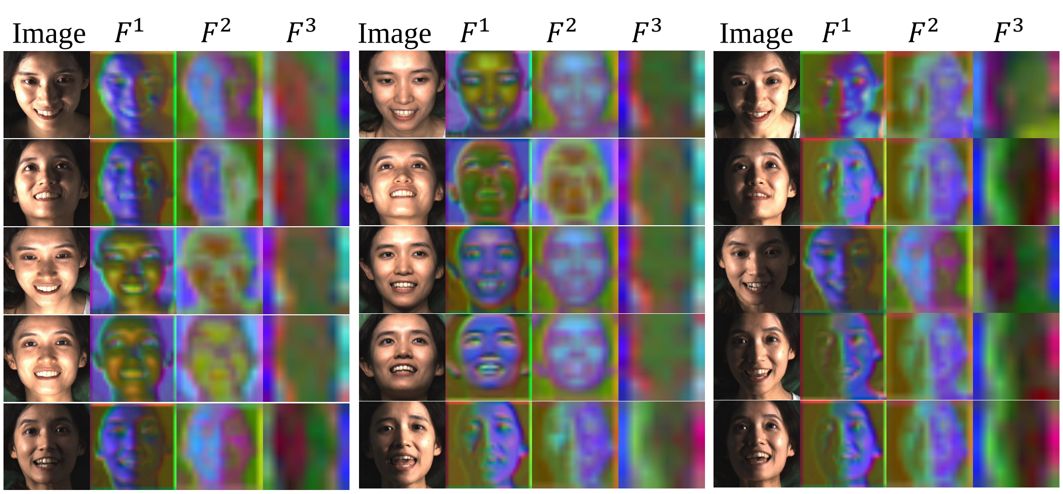

Figure S4: Visualization of the features at different scales.

Table S2: Inference speed comparison

Methods	FPS
Poisson	3.891
DIH	1.247
STHP	1.686
WCT	2.817
AOT (ours)	12.821

C.4 Speed Comparison

Furthermore, as reported in Table S2, our framework achieves the highest FPS compared with other related methods, which means our method introduces the minimum computational burdens. All experiments conducted on Ubuntu16.04 with an Intel i7-7700K CPU and a Nvidia 1060 GPU.

C.5 Forgery Detection

Binary detection accuracy of two video classification baselines: I3D [1] and TSN [14] on the hidden set provided by DeeperForensics-1.0 [4].

We trained the baselines on four manipulated datasets of FF++ [9] [9] produced by DeepFakes [2], Face2Face [11], FaceSwap [3], and NeuralTextures [12]. (Green bars). Then, we add 100 manipulated videos produced by our method to the training set. All detection accuracies are improved with the addition of our data. (Blue bars).

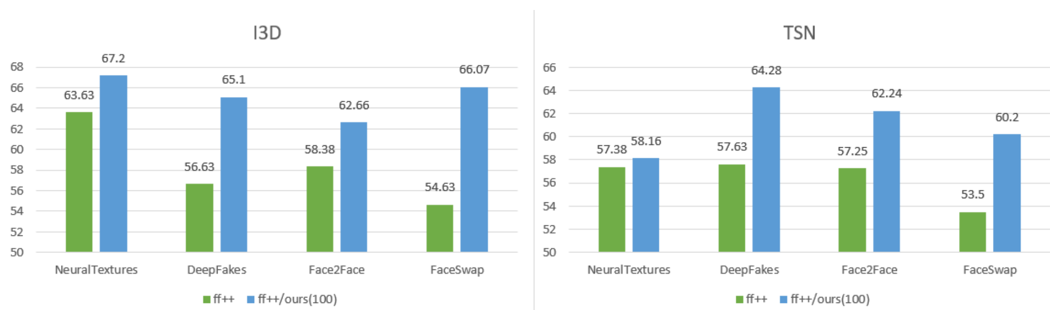

Figure S5: Forgery Detection Results.

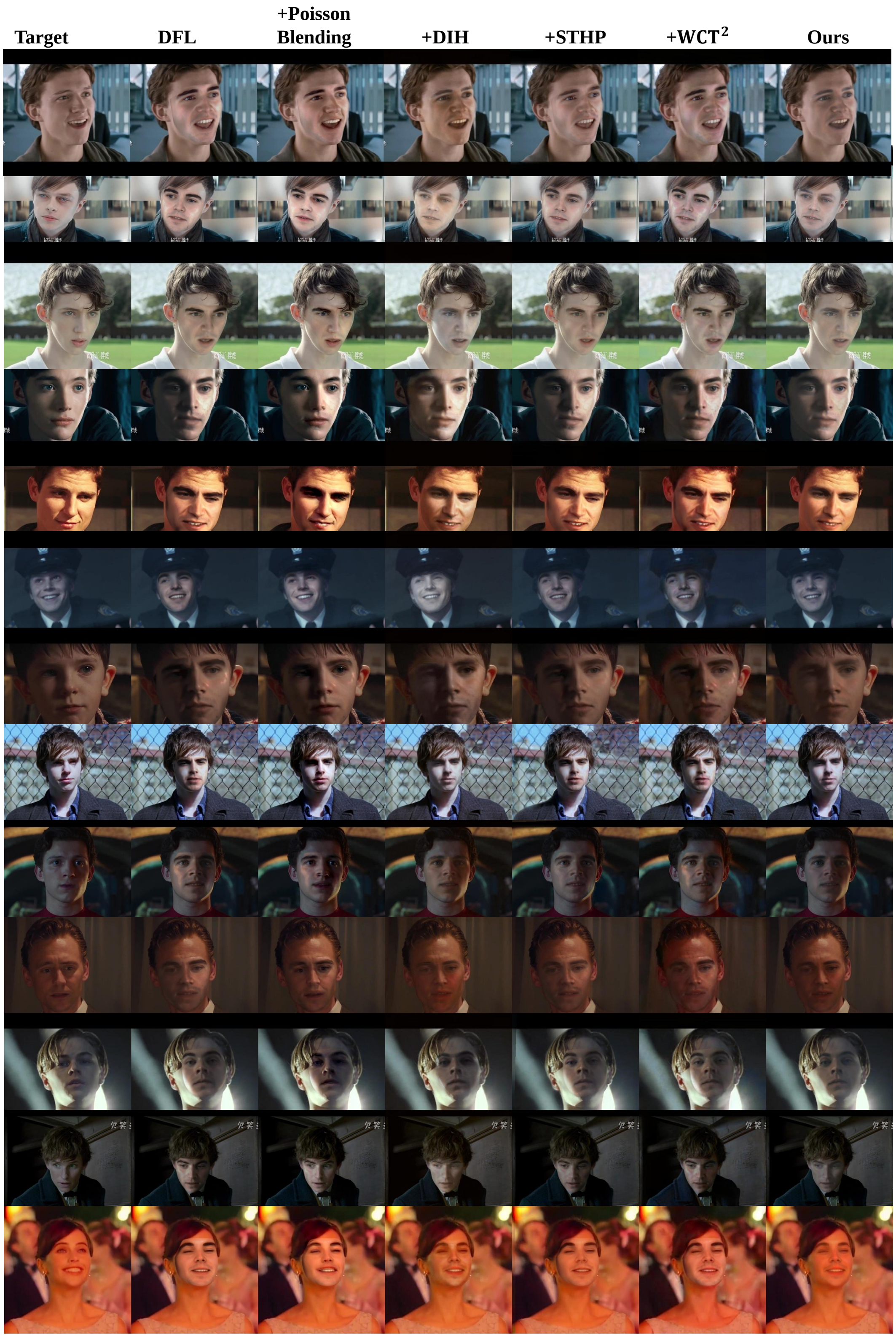

Figure S6: Comparison results with DFL.

Figure S7: Comparison results with FSGAN.

References

[1] João Carreira and Andrew Zisserman. Quo vadis, action recognition? A new model and the kinetics dataset. In *Conference on Computer Vision and Pattern Recognition*, pages 4724–4733, 2017.

[2] Deepfakes. `https://github.com/deepfakes/faceswap`, Accessed: 2020.4.

[3] FaceSwap. `https://github.com/ondyari/FaceForensics/tree/master/dataset/FaceSwapKowalski`, Accessed: 2020.4.

[4] Liming Jiang, Wayne Wu, Ren Li, Chen Qian, and Chen Change Loy. Deeperforensics-1.0: A large-scale dataset for real-world face forgery detection. *arXiv preprint arXiv:2001.03024*, 2020.

[5] Hiroharu Kato, Yoshitaka Ushiku, and Tatsuya Harada. Neural 3d mesh renderer. In *Conference on Computer Vision and Pattern Recognition*, 2018.

[6] Yuval Nirkin, Yosi Keller, and Tal Hassner. Fsgan: Subject agnostic face swapping and reenactment. In *International Conference on Computer Vision*, pages 7184–7193, 2019.

[7] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8024–8035, 2019.

[8] Ivan Perov, Daiheng Gao, Nikolay Chervoniy, Kunlin Liu, Sugasa Marangonda, Chris Umé, Mr. Dpfks, Carl Shift Facenheim, Luis RP, Jian Jiang, Sheng Zhang, Pingyu Wu, Bo Zhou, and Weiming Zhang. Deepfacelab: A simple, flexible and extensible face swapping framework. *arXiv preprint arXiv:2005.05535*, 2020.

[9] Andreas Rossler, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, and Matthias Nießner. Faceforensics++: Learning to detect manipulated facial images. In *International Conference on Computer Vision*, pages 1–11, 2019.

[10] YiChang Shih, Sylvain Paris, Connelly Barnes, William T Freeman, and Frédo Durand. Style transfer for headshot portraits. *ACM Transactions on Graphics*, 33(4):148, 2014.

[11] Justus Thies, Michael Zollhöfer, Marc Stamminger, Christian Theobalt, and Matthias Nießner. Face2face: Real-time face capture and reenactment of RGB videos. In *Conference on Computer Vision and Pattern Recognition*, pages 2387–2395, 2016.

[12] Justus Thies, Michael Zollhöfer, and Matthias Nießner. Deferred neural rendering: Image synthesis using neural textures. *CoRR*, abs/1904.12356, 2019.

[13] Yi-Hsuan Tsai, Xiaohui Shen, Zhe Lin, Kalyan Sunkavalli, Xin Lu, and Ming-Hsuan Yang. Deep image harmonization. In *Conference on Computer Vision and Pattern Recognition*, pages 3789–3797, 2017.

[14] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *European Conference on Computer Vision*, pages 20–36, 2016.

[15] Xiangyu Zhu, Xiaoming Liu, Zhen Lei, and Stan Z Li. Face alignment in full pose range: A 3d total solution. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(1):78–92, 2017.