DPL: Cross-quality DeepFake Detection via Dual Progressive Learning (Supplementary Material)

Dongliang Zhang, Yunfei Li, Jiaran Zhou, Yuezun Li*

School of Computer Science and Technology,
Ocean University of China, Qingdao, China

1 More Implementation Details

During the training phase, the PPO algorithm [5] is employed as follows: the network Φ, which includes GRUs and multi-layer perceptrons (MLPs), acts as the policy network. The sub-network $\mathcal{V}$ functions as a value network, sharing the GRUs in Φ but utilizing new MLPs. For stage II, we configure a distinct training setup, using the Adam optimizer [1] with a learning rate of 0.0003.

When using ConvNeXt-T [2] as the backbone, both the input dimension and the hidden state dimension of the FIPL and VQPL branches are set to 768.

In the main experiments, we randomly select four compression factors for the test images: 100, 78, 23, and 9. A higher factor indicates better image quality, with 100 representing an uncompressed image.

In the robustness analysis, the severity levels for each perturbation are as follows: For Saturation, we first convert the RGB image $\mathbf{x}$ to a YCbCr image $\mathbf{x}^*$ and adjust the saturation using this equation: $\mathbf{x}^* = 128 + (\mathbf{x}^* - 128) \cdot \alpha$, where α is the factor selected from $\{0.4, 0.3, 0.2, 0.1, 0.0\}$. Then the YCbCr image is converted back to RGB space. For Contrast, we directly multiply the image $\mathbf{x}$ with a factor α from $\{0.85, 0.725, 0.6, 0.475, 0.35\}$. For Blockwise, the block size is 8×8 and the number of blocks α is selected from $\{16, 32, 48, 64, 80\}$. For Noise, we scale the magnitude of Gaussian noise with a factor α from $\{0.001, 0.002, 0.005, 0.01, 0.05\}$.

2 More Experimental Details and Analysis

More Details of Cross-quality Cross-manipulation Evaluation. Tab. 1 shows the detail of *Tab. 3 in the main body* under specific compression factors. We can observe that our method performs competitively and often surpasses others, especially at lower quality levels. For instance, it improves performance by approximately 2% on 9(c40) and 1% on 9(c23) when trained on FS dataset.

Effect of Each Component. This part further studies the effect of each component on other datasets: CDFv2, FFIW10k, FSH(c40), and FSH(c23), respectively. The results are shown in Tab. 2. The effect of these components

* Corresponding author (`liyuezun@ouc.edu.cn`).

Table 1: Cross manipulation evaluation results (AUC).

	100%(c40)	78%(c40)	23%(c40)	9%(c40)	100%(c23)	78%(c23)	23%(c23)	9%(c23)
				NT				
QAD	0.6417	0.6433	0.6457	0.6270	0.7245	0.7051	0.6766	0.6539
UCF	0.6164	0.6175	0.6170	0.6040	0.6676	0.6565	0.6389	0.6288
DPL	0.6492	0.6479	0.6413	0.6169	0.7212	0.7012	0.6751	0.6472
				F2F				
QAD	0.6841	0.6841	0.6802	0.6699	0.6932	0.6931	0.6928	0.6884
UCF	0.6766	0.6772	0.6721	0.6644	0.6859	0.6858	0.6791	0.6751
DPL	0.6972	0.6942	0.6842	0.6649	0.7343	0.7257	0.7078	0.6848
				FS				
QAD	0.6490	0.6505	0.6620	0.6692	0.6715	0.6740	0.6872	0.6958
UCF	0.6649	0.6640	0.6702	0.6657	0.6778	0.6785	0.6838	0.6821
DPL	0.6671	0.6718	0.6842	0.6856	0.7120	0.7105	0.7166	0.7099
				DF				
QAD	0.7120	0.7108	0.7075	0.7048	0.6988	0.6973	0.7017	0.7039
UCF	0.6952	0.6955	0.6977	0.6890	0.6652	0.6667	0.6717	0.6724
DPL	0.7102	0.7071	0.7081	0.7011	0.6995	0.6983	0.7007	0.6963

varies across different datasets. For instance, omitting FIPL results in the best performance on FSH(c40), while excluding $\mathcal{L}_{\text{REG}}$ yields the best results on FFIW10k. However, incorporating all components tends to perform well across most datasets.

Table 2: Performance (AUC) of different components on the proposed DPL.

Variant	FSH(c40)	FSH(c23)	CDFv2	FFIW10k
Random JPEG Compression on test set				
baseline	67.51	74.20	73.37	69.32
w/o VQPL	65.28	70.03	71.67	66.83
w/o FIPL	70.19	74.67	69.87	67.36
w/o Train Stage II	68.43	75.04	71.08	68.84
w/o $\mathcal{L}_{\text{REG}}$	68.70	73.82	73.21	69.00
DPL (ours)	68.11	74.91	71.00	68.77

Effect of Different Backbones. This part studies the effect of different backbones across various datasets. As shown in Tab. 3, our method generally improves performance in most cases, highlighting its generalizability even across different datasets.

Integrating CLIP for Detection. This part further explores the effect of integrating CLIP for detection across different datasets. The results in Tab. 4 represent a similar trend as in *Tab. 7 in the main body*, where inconspicuous improvement has been achieved by integrating CLIP on other datasets.

Integrating Frequency Clues. The previous efforts [3, 4] have demonstrated that frequency information can enhance forgery detection. Building on this in-

Table 3: Effect of Different Backbones.

Backbone	FSH(c40)	FSH(c23)	CDFv2	FFIW10k	Avg
ConvNeXt-B	68.03	74.14	70.83	68.27	70.31
ConvNeXt-B + DPL	69.19	73.88	70.34	69.01	70.65
ConvNeXt-S	67.94	75.34	71.97	68.13	70.84
ConvNeXt-S + DPL	68.44	72.83	72.83	68.46	70.64
Swin-T	67.12	70.90	69.43	67.99	68.86
Swin-T + DPL	67.44	68.99	73.20	66.16	68.95
Res-50	64.28	71.55	70.30	68.51	68.66
Res-50 + DPL	64.82	70.21	71.11	68.82	68.74

Table 4: Performance (AUC) of integrating CLIP image encoder for detection.

Variant	FSH(c40)	FSH(c23)	CDFv2	FFIW10k	Avg
Random JPEG Compression on test set					
concat(CLIP, Ψ)	69.00	74.07	71.07	70.00	71.03
add(CLIP, Ψ)	67.65	74.04	72.12	68.48	70.57
add(CLIP, $\mathcal{P}_1, \mathcal{P}_2$)	68.04	73.24	74.86	69.34	71.37

sight, we explore the impact of adding frequency-related modules to our method. Specifically, we use the FAD [4] module to extract frequency information for forgery detection. However, as indicated in Tab. 5, incorporating frequency information does not lead to a significant improvement.

Table 5: Performance (AUC) of FAD component.

Setting	FF++(c40)	FF++(c23)	FSH(c40)	FSH(c23)	CDFv2	FFIW10k	Avg
Random JPEG Compression on test set							
w/ FAD	85.17	93.84	65.61	71.54	72.95	68.36	76.24
w/o FAD [4] (Ours)	86.36	94.41	68.11	74.91	71.00	68.77	77.26

More Analysis on FII. To achieve this assessment, we attempts several paired text prompts, and inspect if the indication score matches the degree of manipulation. The paired text prompt are as follows:

(1) `‘‘realistic face’’, ‘‘synthetic face’’`
(2) `‘‘real face’’, ‘‘fake face’’`
(3) `‘‘real face’’, ‘‘manipulated face’’`
(4) `‘‘genuine face’’, ‘‘manipulated face’’`

The comparison results are shown in Tab. 6, it can be seen that the final prompt aligns more closely with actual situation. For more realistic fake images, such as those in the NeuralTextures (NT) dataset, the prediction accuracy is relatively low. As shown in Fig. 1, the forgery identifiability levels of the two types of

Table 6: Performance (ACC) of different prompt of FII.

Setting	DF	NT	FS	F2F
(1)	29.54	25.48	26.21	25.60
(2)	98.44	97.67	97.61	97.24
(3)	88.82	81.81	87.85	82.70
(4)	29.41	11.43	20.14	12.17

forged images, Deepfakes and FaceSwap, are relatively similar. The identifiability levels of Face2Face and NeuralTextures are closed to each other, with Deepfake and FaceSwap being less authentic than Face2Face and NeuralTextures. That is why prompt (1), (2) and (3) are not selected.

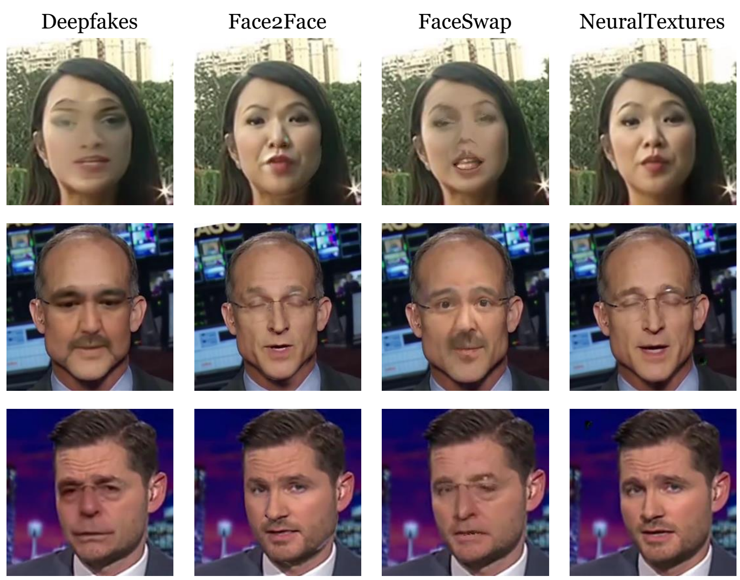

Fig. 1: Examples of Different Forgery type

References

1. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization (2017)
2. Liu, Z., Mao, H., Wu, C., Feichtenhofer, C., Darrell, T., Xie, S.: A convnet for the 2020s. In: CVPR (2022)
3. Luo, Y., Zhang, Y., Yan, J., Liu, W.: Generalizing face forgery detection with high-frequency features. In: CVPR (2021)
4. Qian, Y., Yin, G., Sheng, L., Chen, Z., Shao, J.: Thinking in frequency: Face forgery detection by mining frequency-aware clues. In: ECCV (2020)

5. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O.: Proximal policy optimization algorithms (2017)