Supplementary Material

1. Dataset and Implementation Details

The data preprocessing pipeline employed in this study relies primarily on the standardized framework provided by **DeepfakeBench** [7]. Utilizing this established benchmark ensures that our face extraction, alignment, and normalization procedures remain consistent with state-of-the-art methodologies, thereby facilitating fair comparison.

1.1. Data Augmentation Protocols

To ensure robust generalization, we implemented a diverse augmentation pipeline using the Albumentations framework. Standard transformations include spatial perturbations (random flips, rotations) and visual degradations (image compression, Gaussian blur, color jittering) to simulate real-world variability. Crucially, to prevent the model from overfitting to specific sampling artifacts, we employed a randomized **Isotropic Resize** strategy that stochastically alternates between Bicubic, Bilinear, and Area interpolation kernels during training.

1.2. Dataset Selection and Configuration

We constructed our training and evaluation corpora using the latest iterations of deepfake datasets as curated in Yan et al. [7, 8], in conjunction with the **DF40** dataset. To maintain temporal consistency across input samples, we enforced a fixed frame sampling strategy. Specifically, the hyperparameter defining the input sequence length was set to $\texttt{num\_frames} = 32$. This configuration ensures that the model captures sufficient temporal artifacts without incurring excessive computational overhead.

1.3. Integration with TSRL Architecture

To facilitate the specific state-tracking requirements of the Tutor-Student Reinforcement Learning (TSRL) framework, we developed a custom data interface adapter. While the standard benchmarking loaders provide robust image retrieval, they do not inherently preserve sample identifiers following the stochastic shuffling inherent to training pipelines. This limitation prevents the RL agent from accurately mapping rewards or curriculum adjustments back to specific training instances.

The implemented adapter resolves this by introducing a non-destructive metadata layer that ensures sample traceability through two primary mechanisms:

- **Sample Indexing:** The interface intercepts individual data samples retrieved from the underlying dataset and injects a unique global identifier into the sample structure. This ensures that every input instance retains a distinct tag regardless of its position in the training queue.
- **Batch-Level Aggregation:** During the construction of training batches, the adapter extracts these identifiers and aggregates them into a dedicated tracking vector. This vector is appended to the final batch output, allowing the TSRL algorithm to maintain a deterministic correspondence between the randomly sampled tensor data and the global training environment.

2. Further Results

In the primary analysis, the Area Under the Receiver Operating Characteristic Curve (AUC) was utilized to benchmark ranking performance independent of threshold selection. However, for practical deployment, the Accuracy (ACC) metric provides a vital complementary perspective by evaluating the model's decision-making precision under a fixed classification boundary.Table 1 details the comparative performance across extensive cross-dataset and cross-method experimental settings. A holistic examination of these results reveals that the Tutor-Student Reinforcement Learning (TSRL) framework consistently enhances the discrimination capability of various baseline architectures.

In the cross-dataset evaluation, which challenges the models with unseen domain distributions, the TSRL-augmented detectors demonstrate notable stability, suggesting that the dynamic curriculum effectively prevents the learning of dataset-specific biases. Collectively, these accuracy metrics corroborate the AUC findings presented in the main text, confirming that the performance gains attributed to the adaptive curriculum are robust, metric-agnostic, and beneficial across varying testing conditions.

3. Formal Algorithmic Procedures

To facilitate reproducibility, we provide the formal procedural definitions for the Tutor-Student Reinforcement Learning (TSRL) training loop and the reward shaping mechanism.

Algorithm 1 Dense State-Change Reward Calculation

function CALCREWARD($p_{\text{old}}, c_{\text{old}}, p_{\text{new}}, c_{\text{new}}$)
  $\Delta_{\text{conf}} \leftarrow p_{\text{new}}[\text{target}] - p_{\text{old}}[\text{target}]$
  Initialize reward $r \leftarrow 0$
  if $c_{\text{old}} ==$ False **and** $c_{\text{new}} ==$ True **then**
    $r \leftarrow 1.0$ ▷ Success
  else if $c_{\text{old}} ==$ True **and** $c_{\text{new}} ==$ False **then**
    $r \leftarrow -1.0$ ▷ Forgetting
  else if $c_{\text{old}} ==$ True **and** $c_{\text{new}} ==$ True **then**
    $r \leftarrow 0.5 \times \Delta_{\text{conf}}$ ▷ Gain/Loss
  else
    $r \leftarrow -0.5 \times \Delta_{\text{conf}}$ ▷ Penalty
  end if
  return r
end function

Table 1. **Cross-dataset and cross-type generalization performance (Accuracy)** All models are trained on FF++ (c23) [5] and evaluated on the other datasets/fake data. Results from models trained with our TSRL are highlighted. † denotes models retrained by us;

Methods	Cross-dataset Evaluation					Cross-method Evaluation								
	CDF-v2	DFD	DFDC	DFDCP	Avg.	UniFace	BleFace	MobSwap	e4s	FaceDan	FSGAN	InSwap	SimSwap	Avg.
IID† [2]	0.670	0.832	0.673	0.622	0.699	0.717	0.678	0.814	0.610	0.695	0.815	0.689	0.605	0.703
IID [2] + TSRL	**0.639**	**0.792**	**0.625**	**0.642**	**0.675**	**0.679**	**0.621**	**0.833**	**0.563**	**0.681**	**0.756**	**0.672**	**0.627**	**0.679**
CLIP† [4]	0.662	0.692	0.673	0.622	0.662	0.554	0.558	0.752	0.534	0.580	0.582	0.564	0.502	0.578
CLIP [4] + TSRL	**0.764**	**0.884**	**0.650**	**0.696**	**0.749**	**0.554**	**0.590**	**0.752**	**0.560**	**0.660**	**0.707**	**0.608**	**0.521**	**0.619**
CORE† [3]	0.689	0.860	0.621	0.713	0.721	0.725	0.713	0.825	0.600	0.648	0.837	0.718	0.580	0.706
CORE [3] + TSRL	**0.720**	**0.813**	**0.629**	**0.673**	**0.709**	**0.815**	**0.752**	**0.844**	**0.562**	**0.764**	**0.817**	**0.833**	**0.750**	**0.767**
UCF† [6]	0.663	0.754	0.655	0.594	0.667	0.691	0.670	0.784	0.638	0.715	0.811	0.687	0.632	0.704
UCF [6] + TSRL	**0.632**	**0.756**	**0.622**	**0.652**	**0.666**	**0.699**	**0.680**	**0.799**	**0.679**	**0.693**	**0.774**	**0.718**	**0.662**	**0.713**
ProDet† [1]	0.783	0.877	0.639	0.726	0.756	0.767	0.792	0.914	0.676	0.626	0.794	0.716	0.717	0.750
ProDet [1] + TSRL	**0.804**	**0.856**	**0.622**	**0.760**	**0.761**	**0.757**	**0.774**	**0.925**	**0.668**	**0.634**	**0.803**	**0.735**	**0.705**	**0.750**
Effort† [9]	0.783	0.902	0.714	0.777	0.794	0.843	0.729	0.827	0.827	0.800	0.818	0.806	0.771	0.803
Effort [9] + TSRL	**0.764**	**0.884**	**0.743**	**0.797**	**0.797**	**0.882**	**0.779**	**0.854**	**0.906**	**0.854**	**0.875**	**0.849**	**0.796**	**0.849**

Algorithm 2 TSRL Training Procedure

Require: Student M_S, Tutor T_π, State Manager $\mathcal{M}$, Dataset $\mathcal{D}$, Epochs N, Warmup N_{warm}

Init: Pre-train T_π via BC
for epoch $e = 1$ to N **do**
  Init batch buffer $\mathcal{B} \leftarrow \emptyset$
  for batch (x, y) in $\mathcal{D}$ **do**
    // 1: Pre-Update Obs.
    M_S.eval(); Get $z_{\text{old}}, p_{\text{old}}, f, c_{\text{old}}$
    Retrieve history $\ell_{\text{ema}}, c_{\text{forget}}$ from $\mathcal{M}$
    $s_t \leftarrow [f, p_{\text{old}}, c_{\text{old}}, \ell_{\text{ema}}, c_{\text{forget}}]$
    // 2: Tutor Action
    if $e < N_{\text{warm}}$ **then**
      $w_t \leftarrow \mathbf{1}$ ▷ Supervision
    else
      Sample $w_t \sim T_\pi(s_t)$
    end if
    // 3: Weighted Update
    M_S.train(); $\mathcal{L} = \text{CE}(M_S(x), y)$
    Update $\theta_S \leftarrow \theta_S - \eta \nabla (w_t \cdot \mathcal{L})$
    // 4: Post-Update & Reward
    M_S.eval(); Get $p_{\text{new}}, c_{\text{new}}$
    $r_t \leftarrow \text{CalcReward}(\dots)$
    Update $\mathcal{M}$; Store (s_t, w_t, r_t)
  end for
  // 5: Tutor Update
  if $e \geq N_{\text{warm}}$ **then**
    Update T_π (PPO) on $\mathcal{B}$
    Clear buffer $\mathcal{B}$
  end if
end for

3.1. TSRL Training Procedure Details

Algorithm 2 outlines the complete interaction loop between the Student detector (M_S), the Tutor agent (T_π), and the State Manager ($\mathcal{M}$). A critical implementation detail is the *two-pass* evaluation performed at every step.

3.2. Reward Calculation Logic

The dense reward signal is computed for each sample individually. Algorithm 1 formalizes the logic used to quantify the "value" of a teaching action based on the immediate shift in the Student's decision boundary.

References

[1] Jikang Cheng, Zhiyuan Yan, Ying Zhang, Yuhao Luo, Zhongyuan Wang, and Chen Li. Can we leave deepfake data behind in training deepfake detector? In *Advances in Neural Information Processing Systems*, pages 21979–21998. Curran Associates, Inc., 2024. 2

[2] Baojin Huang, Zhongyuan Wang, Jifan Yang, Jiaxin Ai, Qin Zou, Qian Wang, and Dengpan Ye. Implicit identity driven deepfake face swapping detection. In *CVPR*, pages 4490–4499, 2023. 2

[3] Yunsheng Ni, Depu Meng, Changqian Yu, Chengbin Quan, Dongchun Ren, and Youjian Zhao. Core: Consistent representation learning for face forgery detection. In *CVPRW*, pages 12–21, 2022. 2

[4] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763, 2021. 2

[5] Andreas Rossler, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, and Matthias Nießner. FaceForensics++: Learning to detect manipulated facial images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1–11. IEEE/CVF, 2019. 2

[6] Zhiyuan Yan, Yong Zhang, Yanbo Fan, and Baoyuan Wu. Ucf: Uncovering common features for generalizable deepfake detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 22412–22423, 2023. 2

[7] Zhiyuan Yan, Yong Zhang, Xinhang Yuan, Siwei Lyu, and Baoyuan Wu. Deepfakebench: A comprehensive benchmark of deepfake detection. In *Advances in Neural Information Processing Systems*, pages 4534–4565. Curran Associates, Inc., 2023. 1

[8] Zhiyuan Yan, Taiping Yao, Shen Chen, Yandan Zhao, Xinghe Fu, Junwei Zhu, Donghao Luo, Chengjie Wang, Shouhong Ding, Yunsheng Wu, et al. Df40: Toward next-generation deepfake detection. *Advances in Neural Information Processing Systems*, 37:29387–29434, 2024. 1

[9] Zhiyuan Yan, Jiangming Wang, Peng Jin, Ke-Yue Zhang, Chengchun Liu, Shen Chen, Taiping Yao, Shouhong Ding, Baoyuan Wu, and Li Yuan. Orthogonal subspace decomposition for generalizable ai-generated image detection. In *Forty-second International Conference on Machine Learning*, 2025. 2