PROTOTYPE-GUIDED TANGENT–BOUNDARY PERTURBATION FOR AI-GENERATED IMAGE DETECTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Detectors that synthesize features during training have improved AI-generated image detection. However, our visualization reveals that synthesized features reshape rather than enrich the original fake-feature distribution, causing the detector to learn from the synthesized features while losing its understanding of the original distribution. To introduce additional training variations while retaining supervision on the original features, we propose **Prototype-Guided Tangent-Boundary Perturbation (PTBP)**, a feature-level perturbation method. First, its orthogonal subspace decomposition uses class prototypes and within-class feature variances to decompose sampled Gaussian noise into components orthogonal and parallel to the discriminative direction. This decomposition allows PTBP to control what each perturbation component preserves. Then, controlled feature perturbation uses the orthogonal component to construct tangent views that preserve the predictions of the corresponding original features. It further adds a class-aware and margin-calibrated parallel component to construct boundary views, which are supervised with the original binary labels after class-aware shifts. The two views are trained using tangent consistency and boundary supervision, respectively. Both objectives are jointly optimized to encourage the detector to learn from additional training variations while retaining supervision on the original features. With only one training epoch, PTBP achieves the best performance in both cross-generator and compression-perturbed scenarios.

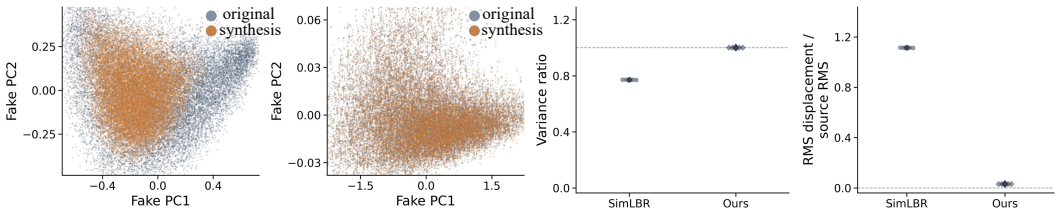
1 INTRODUCTION

Recent advances in image generation Dhariwal & Nichol (2021); Rombach et al. (2022); Ramesh et al. (2021); Ho et al. (2020); Goodfellow et al. (2014) have made synthetic images increasingly difficult to distinguish from real images, raising concerns about their use in spreading misinformation. Many existing detection methods rely on spatial Zhong et al. (2023); Yan et al. (2025); Koutlis & Papadopoulos (2024); Ojha et al. (2023) and frequency-domain Frank et al. (2020); Tan et al. (2024b); Qian et al. (2020); Luo et al. (2021) artifacts introduced by image generators. However, detection performance often declines on images from unseen generators due to overfitting to generator-specific artifacts Tan et al. (2024a); Kong et al. (2026); Li et al. (2026). Collecting images from more generators can diversify the training data, but this approach is costly and cannot easily keep pace with new generation methods. Images generated by proprietary models may also be unavailable for training. These limitations motivate the construction of additional training features from existing source data during training.

Existing approaches synthesize training examples in image space or feature space. Image-space methods synthesize pseudo-fake images by introducing artifacts or blending real images Zhang et al. (2019); Li et al. (2020); Shiohara & Yamasaki (2022); Chen et al. (2022); Zhao et al. (2021). Feature-space methods synthesize training examples by perturbing frequency signals or intermediate representations Jeong et al. (2022); Li et al. (2024); Yan et al. (2024). Recently, SimLBR Dhakal et al. (2026) combines features from a frozen pretrained large-scale model with real-image modeling and uses synthesized fake features instead of the original ones for training.

However, our visualizations in Figure 1 demonstrate centroid displacement and reduced within-class variance. We choose SimLBR Dhakal et al. (2026) as the comparison method because it achieves the best balance between generalization and reliability among recent feature synthesis approaches.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



(a) SimLBR feature distribution (b) PTBP feature distribution (c) Within-class variance (d) Perturbation magnitude

Figure 1: The motivation of PTBP. (a) SimLBR reshapes the original feature rather than enriching original feature distribution during training. (b) The features synthesized by PTBP largely cover the original features. (c) The variance ratios show reduced within-class variance for SimLBR and approximately unchanged within-class variance for PTBP. (d) PTBP introduces only small perturbation while achieving substantial gains in detection accuracy (Table 1).

These changes reveal that synthesized features reshape the original fake-feature distribution rather than enriching the original feature space while preserving its structure. This leads to the detector losing its understanding of the original distribution while learning the synthesized one. This raises the practical question of **how controlled feature synthesis can introduce additional training variations while retaining supervision on the original features.**

To address this issue, we propose a feature-level perturbation method, termed Prototype-Guided Tangent-Boundary Perturbation (PTBP). PTBP introduces additional training variations while retaining supervision on the original features. First, **Orthogonal Subspace Decomposition** uses class prototypes and coordinate-wise within-class feature variances to define a discriminative direction in a prototype-whitened feature space. It then decomposes sampled Gaussian noise into components orthogonal and parallel to this direction. This decomposition allows PTBP to control what each perturbed view should preserve. Based on the decomposed components, **Controlled Feature Perturbation** uses the orthogonal component to construct tangent views that preserve the predictions of the corresponding original features. It further combines the orthogonal component with a class-aware and margin-calibrated parallel component to construct boundary views, which are supervised with the original binary labels after class-aware shifts.

These two views determine the training objectives of PTBP. We introduce tangent consistency to encourage prediction consistency by matching the predicted real/fake probabilities of each tangent view to those of the corresponding original feature. Boundary supervision trains each boundary view with the original binary label of its corresponding feature. Both objectives are jointly optimized with the original classification loss and prototype regularization. The original classification loss retains direct supervision from the original features, while prototype regularization encourages features to remain close to their corresponding class prototypes. Together, these objectives encourage the detector to learn from additional training variations while retaining supervision from the original training distribution. The contribution can be summarized as follows:

- We propose **Prototype-Guided Tangent-Boundary Perturbation (PTBP)**, a feature-level perturbation method. Its orthogonal subspace decomposition and controlled feature perturbation construct tangent views trained for prediction consistency and boundary views supervised with the original binary labels.
- We use tangent consistency and boundary supervision encourage prediction consistency and correct classification of the perturbed views, respectively. Both objectives are jointly optimized to encourage the detector to learn from additional training variations while retaining supervision on the original features.
- With only one training epoch, the PTBP-trained detector achieves optimal performance in both cross-generator and compression-perturbed scenarios.

2 Related Works

Many existing synthetic image detectors rely on spatial Zhong et al. (2023); Yan et al. (2025); Koutlis & Papadopoulos (2024); Ojha et al. (2023) or frequency-domain Frank et al. (2020); Tan et al.

(2024b); Qian et al. (2020); Luo et al. (2021) artifacts left by generators. However, these detectors can overfit to generator-specific artifacts, limiting their generalization to unseen generators Tao et al. (2025); Li et al. (2026); Tan et al. (2024a; 2023). To improve generalization, recent methods dynamically synthesize additional examples from the source data during training. Image-space methods simulate generation artifacts Zhang et al. (2019) or construct pseudo-fake images by blending real images Li et al. (2020); Shiohara & Yamasaki (2022). FrePGAN introduces frequency-level perturbations to encourage the detector to learn transferable artifacts Jeong et al. (2022), while FreqBlender blends frequency components from fake faces into real faces to generate pseudo-fake images Li et al. (2024). In feature space, LSDA constructs variations within and across forgery features to improve generalization Yan et al. (2024). SimLBR replaces the original fake features with synthesized features during training while retaining the original real features.

However, introducing synthesized features during training is not simply equivalent to adding samples to the original training distribution. Our visualizations reveal centroid displacement and reduced within-class variance, indicating that synthesized features reshape the original fake-feature distribution rather than enriching the original feature space while preserving its structure. This leads the detector to lose its understanding of the original distribution while learning from the synthesized one. Motivated by this observation, we propose Prototype-Guided Tangent-Boundary Perturbation, a feature-level perturbation method. Its controlled feature perturbation constructs tangent views that preserve the predictions of the corresponding original features and boundary views that preserve the original forgery labels under shifts toward the opposite class, enabling the detector to learn from additional variations while retaining supervision from the original training distribution.

3 METHOD

3.1 OVERVIEW

We propose **Prototype-Guided Tangent-Boundary Perturbation (PTBP)**, a feature-level perturbation method that introduces additional training variations while retaining supervision on the original features, as illustrated in Figure 2. Its orthogonal subspace decomposition first uses class prototypes and within-class feature variances to decompose Gaussian noise into components orthogonal and parallel to the discriminative direction. Then, Controlled Feature Perturbation uses the decomposed noise component to construct tangent views and boundary views. Corresponding tangent consistency and boundary supervision are jointly optimized with the original classification loss and prototype regularization, retaining supervision from the original training distribution.

3.2 ORTHOGONAL SUBSPACE DECOMPOSITION

To introduce additional training variations while retaining supervision on the original features, PTBP must control what each perturbation component preserves. We therefore propose **Orthogonal Subspace Decomposition**, which uses class prototypes and within-class feature variances to decompose Gaussian noise into components orthogonal and parallel to the discriminative direction. This decomposition allows PTBP to control the preservation target of each perturbation component in the subsequent **Controlled Feature Perturbation** module.

Specifically, we first use a convolutional block to fuse features from a frozen CLIP ViT-L/14 encoder into $\mathbf{z} \in \mathbb{R}^d$ for classification and perturbation. Let $y_i \in \{0, 1\}$ denote the label of sample i, with 0 indicating a real image and 1 a synthetic image. For each class c, we maintain a class prototype $\mathbf{p}_c \in \mathbb{R}^d$ and a vector of coordinate-wise variances $\mathbf{v}_c \in \mathbb{R}^d$. For the batch subset $\mathcal{B}_c$ with label c, we compute the mean $\hat{\mathbf{p}}_c = |\mathcal{B}_c|^{-1} \sum_{i \in \mathcal{B}_c} \mathbf{z}_i$ and define the coordinate-wise variance:

$$\hat{\mathbf{v}}_c = \max\left(\frac{1}{|\mathcal{B}_c|} \sum_{i \in \mathcal{B}_c} \mathbf{z}_i \odot \mathbf{z}_i - \hat{\mathbf{p}}_c \odot \hat{\mathbf{p}}_c, \epsilon \right), \tag{1}$$

where $\odot$ denotes elementwise multiplication and $\epsilon > 0$ provides a lower bound on each variance. To stabilize the perturbation direction and scale across training batches, we smooth the class prototypes and variance estimates using an exponential moving average (EMA). The corresponding updates for both statistics can be formulated as follows:

$$\mathbf{p}_c \leftarrow m\mathbf{p}_c + (1 - m)\hat{\mathbf{p}}_c, \qquad \mathbf{v}_c \leftarrow m\mathbf{v}_c + (1 - m)\hat{\mathbf{v}}_c, \tag{2}$$

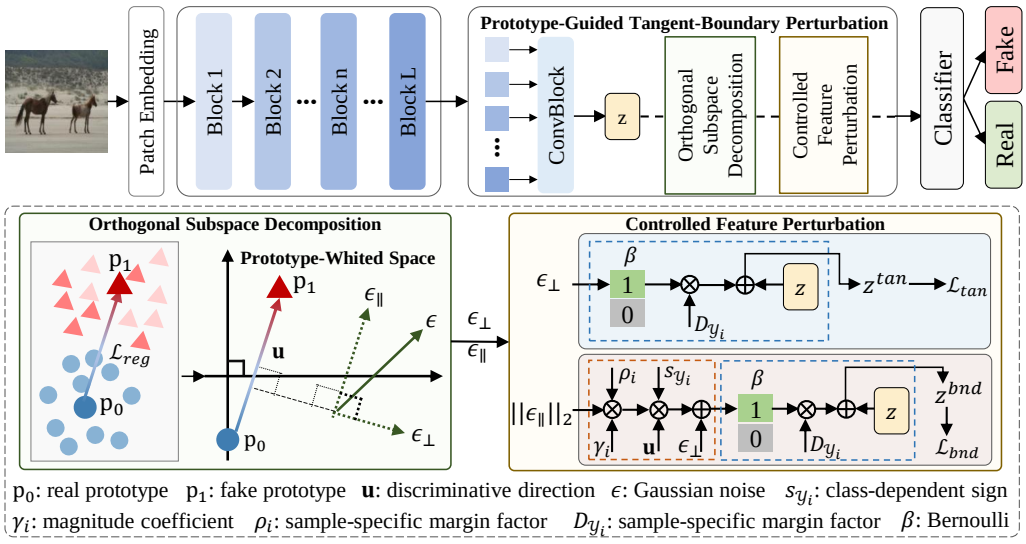


Figure 2: Illustration of PTBP. During training, PTBP uses class prototypes to define a discriminative direction in whitened coordinates. Gaussian noise is decomposed along this direction and its orthogonal complement to construct tangent $\mathbf{z}^{\mathrm{tan}}$ and boundary views $\mathbf{z}^{\mathrm{bnd}}$. Tangent consistency $\mathcal{L}_{\mathrm{tan}}$ encourages stable predictions, while boundary supervision $\mathcal{L}_{\mathrm{bnd}}$ trains the class-aware shifted features with their original labels. During inference, the classifier predicts from $\mathbf{z}$ without PTBP.

where $m \in [0, 1)$ controls the contribution of historical statistics to each update. The EMA-updated prototypes $\mathbf{p}_c$ and variances $\mathbf{v}_c$ are treated as constants when optimizing the detector. To encourage features to remain close to their corresponding prototypes, we define the regularization loss:

$$\mathcal{L}_{\mathrm{reg}} = \frac{1}{B} \sum_{i=1}^{B} \|\mathbf{z}_i - \mathrm{sg}(\mathbf{p}_{y_i})\|_2^2 , \tag{3}$$

where B is the batch size and sg denotes the stop-gradient operator. It encourages class-conditional compactness around the EMA prototypes.

To account for differences in within-class feature variances across coordinates, we normalize the coordinates using shared within-class standard deviations. This coordinate-wise normalization defines the prototype-whitened feature space, where the difference between the real and fake class prototypes determines the discriminative direction. We define the shared diagonal scaling matrix $\mathbf{D}$ and the discriminative direction $\mathbf{u}$ as follows:

$$\mathbf{D} = \mathrm{diag}\left(\sqrt{\frac{\mathbf{v}_0 + \mathbf{v}_1}{2}}\right), \qquad \mathbf{u} = \frac{\mathbf{D}^{-1}(\mathbf{p}_1 - \mathbf{p}_0)}{\|\mathbf{D}^{-1}(\mathbf{p}_1 - \mathbf{p}_0)\|_2}, \tag{4}$$

where diag constructs a diagonal matrix. The discriminative direction $\mathbf{u}$ points from the real prototype toward the fake prototype in the prototype-whitened feature space.

The operators $\Pi_{\|} = \mathbf{u}\mathbf{u}^{\top}$ and $\Pi_{\perp} = \mathbf{I}_d - \mathbf{u}\mathbf{u}^{\top}$ project a Gaussian perturbation onto the discriminative direction and its orthogonal complement, respectively. Here, $\mathbf{I}_d$ denotes the identity matrix in the prototype-whitened feature space. For a sampled Gaussian noise vector $\boldsymbol{\epsilon}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_d)$, the two components are obtained as follows:

$$\boldsymbol{\epsilon}_{i,\|} = \Pi_{\|}\boldsymbol{\epsilon}_i, \qquad \boldsymbol{\epsilon}_{i,\perp} = \Pi_{\perp}\boldsymbol{\epsilon}_i. \tag{5}$$

In the prototype-whitened feature space, the orthogonal component $\boldsymbol{\epsilon}_{i,\perp}$ introduces variation without moving along the discriminative direction, whereas the parallel component $\boldsymbol{\epsilon}_{i,\|}$ controls movement along it. This orthogonality applies before class-specific scaling. This decomposition allows PTBP to control what each perturbation component preserves.

3.3 CONTROLLED FEATURE PERTURBATION

To introduce additional training variations while retaining supervision on the original features, Controlled Feature Perturbation uses the decomposed noise components from Orthogonal Subspace Decomposition to construct tangent and boundary views. For each original feature $\mathbf{z}_i$, we draw $\beta_i \sim \text{Bernoulli}(p)$, where p is the probability of applying a perturbation. Features with $\beta_i = 1$ are selected for perturbation, while those with $\beta_i = 0$ remain unchanged. For each selected feature, we independently draw a magnitude coefficient $\gamma_i \sim \text{Uniform}(0, 1)$ and use the corresponding components $\boldsymbol{\epsilon}_{i,\perp}$ and $\boldsymbol{\epsilon}_{i,\parallel}$ defined in Eq. 5.

Then, we calibrate the magnitude of the parallel perturbation using each sample's estimated distance to the binary classifier's decision boundary, where the real and fake logits are equal. Let $\mathbf{o}_i = \mathbf{W}\mathbf{z}_i + \mathbf{b}$ denote the logits of the original feature $\mathbf{z}_i$, where $\mathbf{w}_c^\top$ is row c of the weight matrix $\mathbf{W}$ and $\mathbf{b}$ is the bias vector. The rows $\mathbf{w}_0^\top$ and $\mathbf{w}_1^\top$ of $\mathbf{W}$ correspond to the real and fake classes, respectively. Using these logits, we estimate the distance to the decision boundary as $r_i = |o_{i,1} - o_{i,0}| / \max(\|\mathbf{w}_1 - \mathbf{w}_0\|_2, \epsilon)$, where $o_{i,0}$ and $o_{i,1}$ denote the real and fake logits, respectively, and $\epsilon > 0$ prevents division by zero. Let $\mathcal{A} = \{i \mid \beta_i = 1\}$ denote the indices of the selected features.

To scale the parallel perturbation according to each feature's distance from the decision boundary, we define a sample-specific margin factor ρ_i. It represents the relative magnitude assigned to the parallel perturbation of feature i. We normalize each distance by the median distance among the selected features and clip the result to $[0.25, \rho_{\max}]$:

$$\rho_i = \text{clip}\left(\frac{r_i}{\max(\text{median}_{j\in\mathcal{A}}\, r_j, \epsilon)}, 0.25, \rho_{\max}\right), \tag{6}$$

where $\rho_{\max}$ specifies the maximum relative magnitude of the parallel perturbation. With ρ_i held fixed during optimization, samples with smaller estimated distances receive smaller parallel coefficients, subject to the clipping bounds.

With the perturbation strength calibrated, we next determine the perturbation's orientation along the discriminative direction. The class-dependent sign $s_{y_i} = 1 - 2y_i$ orients the parallel component along $+\mathbf{u}$ for real samples and $-\mathbf{u}$ for synthetic samples. Combining this sign with the parallel noise magnitude and margin factor gives the class-aware and margin-calibrated parallel component:

$$\boldsymbol{\delta}_{i,\parallel} = \gamma_i \rho_i s_{y_i} \left\|\boldsymbol{\epsilon}_{i,\parallel}\right\|_2 \mathbf{u}. \tag{7}$$

We map the perturbations to feature coordinates using the class-specific scale matrix $\mathbf{D}_{y_i} = \text{diag}(\sqrt{\mathbf{v}_{y_i}})$. Thus, the shared matrix $\mathbf{D}$ normalizes the coordinates used to define the discriminative direction, while $\mathbf{D}_{y_i}$ determines the perturbation scale for each class.

The tangent view uses only the orthogonal noise component. The boundary view combines the orthogonal noise component with the adjusted parallel component. The tangent view $\mathbf{z}_i^{\text{tan}}$ and boundary view $\mathbf{z}_i^{\text{bnd}}$ are defined as follows:

$$\begin{aligned} \mathbf{z}_i^{\text{tan}} &= \mathbf{z}_i + \beta_i \lambda \mathbf{D}_{y_i} \boldsymbol{\epsilon}_{i,\perp}, \\ \mathbf{z}_i^{\text{bnd}} &= \mathbf{z}_i + \beta_i \lambda \mathbf{D}_{y_i} \left(\boldsymbol{\epsilon}_{i,\perp} + \boldsymbol{\delta}_{i,\parallel}\right), \end{aligned} \tag{8}$$

where $\lambda > 0$ controls the overall perturbation strength in both views. Coordinates with greater within-class variance consequently receive larger perturbation scales in the original feature space. After class-specific scaling, the tangent displacement satisfies $\mathbf{u}^\top \mathbf{D}_{y_i}^{-1}(\mathbf{z}_i^{\text{tan}} - \mathbf{z}_i) = 0$. The corresponding tangent hyperplane has normal $\mathbf{D}_{y_i}^{-1}\mathbf{u}$, which need not align with the classifier normal. The boundary view shares the tangent perturbation, adds a class-aware shift along the mapped discriminative direction $\mathbf{D}_{y_i}\mathbf{u}$, and is supervised with the original binary label y_i.

3.4 TRAINING OBJECTIVE

The training objective combines the original classification loss, prototype regularization, tangent consistency, and boundary supervision. Each boundary view is trained with the binary real/fake label y_i of its corresponding original feature. The corresponding classification losses are defined:

$$\mathcal{L}_{\text{ce}} = \frac{1}{B}\sum_{i=1}^{B} \text{CE}(\mathbf{o}_i, y_i), \qquad \mathcal{L}_{\text{bnd}} = \frac{1}{B}\sum_{i=1}^{B} \text{CE}(\mathbf{o}(\mathbf{z}_i^{\text{bnd}}), y_i). \tag{9}$$

Table 1: Generalization evaluation across 16 generators. The upper block reports generator-wise results, while the lower block reports overall mean and relative performance.

Method	Pro-GAN	Style-GAN	Big-GAN	Cycle-GAN	Star-GAN	Gau-GAN	Style-GAN2	WFIR	ADM	Glide	Mid-Journey	SD-v1.4	SD-v1.5	VQDM	Wu-Kong	Dalle2
CNNSpot	100.0	90.17	71.17	87.62	94.60	81.42	86.91	91.65	60.39	58.07	51.39	50.57	50.53	56.46	51.03	50.45
FreDect	99.36	78.02	81.97	78.77	94.62	80.57	66.19	50.75	63.42	54.13	45.87	38.79	39.21	77.80	40.30	34.71
Fusing	100.0	85.20	77.40	87.00	97.00	77.00	83.30	66.80	49.00	57.20	52.20	51.00	51.40	55.10	51.70	52.80
LNP	99.67	91.75	77.75	84.10	99.92	75.39	94.64	70.85	84.73	80.52	65.55	85.55	85.67	74.46	82.06	88.75
LGrad	99.83	91.08	85.62	86.94	99.27	78.46	85.32	55.70	67.15	66.11	65.35	63.02	63.67	72.99	59.55	65.45
UnivFD	99.81	84.93	95.08	98.33	95.75	99.47	74.96	86.90	66.87	62.46	56.13	63.66	63.49	85.31	70.93	50.75
DIRE-G	95.19	83.03	70.12	74.19	95.47	67.79	75.31	58.05	75.78	71.75	58.01	49.74	49.83	53.68	54.46	66.48
DIRE-D	52.75	51.31	49.70	49.58	46.72	51.23	51.72	53.30	98.25	92.42	89.45	91.24	91.63	91.90	90.90	92.45
PatchCraft	100.0	92.77	95.80	70.17	99.97	71.58	89.55	85.80	82.17	83.79	90.12	95.38	95.30	88.91	91.07	96.60
NPR	99.79	97.70	84.35	96.10	99.35	82.50	98.38	65.80	69.69	78.36	77.85	78.63	78.89	78.13	76.11	64.90
Rine	100.0	97.33	94.55	96.48	100.0	83.24	96.22	93.85	89.52	95.72	61.38	97.62	97.70	94.63	96.56	85.10
FatFormer	100.0	93.37	99.18	98.52	99.85	99.41	94.49	94.65	79.17	84.24	56.90	88.18	87.14	90.03	90.84	57.60
Effort	100.0	95.38	99.20	99.77	99.92	99.54	96.64	78.90	75.06	92.68	54.18	88.23	87.96	85.58	85.44	78.10
AIDE	99.99	99.64	83.95	98.48	99.91	73.25	98.00	94.20	93.43	95.09	77.20	93.00	92.85	95.16	93.55	96.60
MoeFFD	99.73	88.82	82.13	97.50	92.82	92.38	89.03	81.50	58.43	62.61	52.23	63.01	64.04	66.21	61.09	53.25
HSIC	100.0	91.22	82.55	95.84	100.0	69.84	94.07	98.05	92.35	96.93	66.93	98.69	98.64	97.70	98.31	97.45
SimLBR	99.29	87.38	95.52	88.11	88.54	97.14	84.57	89.50	85.89	90.88	75.54	85.51	84.43	93.20	90.08	78.80
Ours	100.00	99.20	90.03	97.35	99.52	84.37	98.53	93.05	94.13	98.69	69.92	98.95	98.89	97.89	98.62	97.05

Metric	*CNNSpot*	*FreDect*	*Fusing*	*LNP*	*LGrad*	*UnivFD*	*DIRE-G*	*DIRE-D*	*PatchCraft*	*NPR*	*Rine*	*FatFormer*	*Effort*	*AIDE*	*MoeFFD*	*HSIC*	*SimLBR*	**Ours**
Mean↑	70.78	64.03	68.38	83.84	75.34	78.43	68.68	71.53	89.31	82.91	92.49	88.35	88.54	92.77	75.30	92.41	88.40	**94.76**
Std↓	18.91	21.08	18.04	9.78	14.26	16.73	14.46	21.55	8.89	11.92	9.57	13.56	12.48	7.92	16.68	10.35	**6.23**	7.84
Rel.↑	1.10	0.67	1.02	3.46	1.78	1.70	1.29	1.00	4.42	2.76	4.44	2.83	3.09	5.40	1.52	4.10	**6.16**	5.71

The original classification loss retains direct supervision from the original feature, whereas boundary supervision encourages correct classification of the boundary view with the original binary label after the class-aware and margin-calibrated perturbation.

For tangent consistency, the unperturbed prediction serves as a fixed reference for the tangent view. We use the temperature $T > 0$ to control the sharpness of the prediction distribution. Specifically, we define $\mathbf{q}_T(\mathbf{z}) = \mathrm{softmax}(\mathbf{o}(\mathbf{z})/T)$ and formulate the consistency loss as follows:

$$\mathcal{L}_{\tan} = \frac{T^2}{B} \sum_{i=1}^{B} D_{\mathrm{KL}}\left(\mathrm{sg}(\mathbf{q}_T(\mathbf{z}_i)) \,\middle\|\, \mathbf{q}_T(\mathbf{z}_i^{\tan})\right), \tag{10}$$

where D_{KL} denotes the Kullback-Leibler divergence between the reference and tangent predictions. By matching these predictions, $\mathcal{L}_{\tan}$ encourages prediction consistency under the mapped tangent perturbations. Combining these terms, the complete training objective can be formulated as follows:

$$\mathcal{L}_{\mathrm{total}} = \mathcal{L}_{\mathrm{ce}} + \eta \mathcal{L}_{\mathrm{reg}} + \eta_{\tan} \mathcal{L}_{\tan} + \eta_{\mathrm{bnd}} \mathcal{L}_{\mathrm{bnd}}, \tag{11}$$

where η, $\eta_{\tan}$, and η_{bnd} balance the three auxiliary losses. PTBP is disabled during inference, so predictions use only the original feature $\mathbf{z}$.

4 Experiments

4.1 Evaluation Protocols

We evaluate the proposed method under two scenarios, cross-generator generalization and compression robustness. The former assesses generalization to unseen generators, while the latter assesses performance stability under compression.

Cross-Generator Generalization. We follow the standard evaluation protocol of AIGCDetect-Bench Zhong et al. (2023) to ensure fair comparison with prior work. The **training set** contains only images generated by ProGAN Karras et al. (2018). The **test set** comprises 16 generators,

Table 2: Evaluation results for the 2-class training setting under the quality-aware setting. All comparison results are taken from ODDN Tao et al. (2025) and DDOC Li et al. (2026). **Bold** and underline indicate the best and second-best performance.

Method	InfoMax-GAN	BE-GAN	Cramer-GAN	Att-GAN	MMD-GAN	Rel-GAN	S3-GAN	SNG-GAN	STG-GAN	Pro-GAN	Style-GAN	Style-GAN2	Big-GAN	Cycle-GAN	Star-GAN	Gau-GAN	Deep-fake	Mean Acc.
MesoNet	50.5	50.6	50.0	50.5	50.2	50.4	49.3	50.2	50.9	51.4	51.5	54.2	52.0	53.4	50.6	53.4	50.0	51.2
FF++	74.4	30.6	75.5	64.2	76.3	61.5	54.9	71.8	82.2	90.4	60.4	65.2	60.5	80.0	74.4	72.3	51.0	67.3
F3Net	65.9	42.6	68.9	55.9	63.7	56.3	53.1	62.1	74.9	84.1	56.7	60.4	56.1	77.8	71.2	68.6	50.4	63.2
MAT	54.5	49.8	59.7	50.1	57.8	50.8	52.8	52.8	56.7	85.7	52.4	53.1	52.9	72.2	57.6	67.6	51.1	57.7
SBIs	56.6	51.9	63.4	50.1	59.3	50.6	62.2	52.1	53.0	88.4	51.2	52.4	55.4	74.8	53.6	78.3	51.1	59.3
ADD	52.0	51.0	59.0	50.7	57.2	52.7	44.7	52.3	53.1	70.9	48.0	48.4	51.7	72.4	55.7	64.7	51.3	55.2
QAD	74.8	53.7	79.6	60.1	78.3	66.5	56.0	76.3	80.4	86.3	55.4	57.2	59.1	77.1	79.9	65.8	55.8	69.2
FreqNet	47.7	50.0	50.5	52.6	50.5	47.8	49.1	50.6	51.7	50.1	53.3	55.8	51.6	52.3	51.0	52.1	48.1	50.9
NPR	57.2	53.4	51.8	50.9	50.7	51.8	50.4	49.7	50.8	53.1	52.6	52.3	51.7	55.4	54.3	52.4	49.7	52.2
Fatformer	64.9	53.0	64.5	50.7	61.0	51.7	72.6	55.4	57.9	91.3	54.8	57.7	67.0	87.3	65.1	87.1	51.8	64.3
ODDN	73.1	42.3	76.1	71.2	75.9	72.5	60.5	75.5	85.0	91.3	64.5	69.4	64.3	80.8	78.0	77.3	54.3	71.4
DDOC	86.9	64.9	86.4	54.8	84.9	58.7	74.7	73.7	77.5	94.3	59.3	62.5	75.3	93.9	86.3	92.3	55.9	<u>75.4</u>
Ours	90.4	77.9	93.6	64.8	92.1	71.8	83.2	82.6	85.4	94.4	62.0	56.3	80.0	93.9	92.7	95.3	56.1	**80.7**

Table 3: Evaluation results for the 2-class training setting under the quality-agnostic setting.

Method	InfoMax-GAN	BE-GAN	Cramer-GAN	Att-GAN	MMD-GAN	Rel-GAN	S3-GAN	SNG-GAN	STG-GAN	Pro-GAN	Style-GAN	Style-GAN2	Big-GAN	Cycle-GAN	Star-GAN	Gau-GAN	Deep-fake	Mean Acc.
MesoNet	46.3	44.3	59.7	60.0	59.8	58.7	47.8	56.3	69.1	55.0	51.0	49.9	53.9	60.5	64.8	49.9	51.1	53.3
FF++	66.9	37.7	79.4	56.6	77.1	60.5	55.0	69.2	79.8	87.8	55.1	59.8	57.1	79.9	75.6	71.6	52.0	66.0
F3Net	58.0	48.8	61.9	51.5	59.3	53.2	52.0	54.9	61.1	83.4	52.7	54.9	55.0	73.7	65.9	66.7	52.4	59.3
MAT	54.2	49.9	59.6	50.5	57.6	51.2	52.1	52.7	57.8	86.1	52.3	53.0	52.7	70.3	58.2	68.0	51.3	57.7
SBIs	56.6	51.9	63.4	50.1	59.3	50.6	62.2	52.1	53.0	88.6	51.3	52.4	55.7	76.0	53.9	78.1	51.2	59.4
ADD	51.8	50.9	59.0	50.7	57.1	52.8	45.0	52.3	52.9	70.2	48.0	48.7	51.8	71.9	55.5	65.1	51.3	57.9
QAD	72.3	55.2	80.0	61.5	78.3	65.5	54.5	76.5	79.2	86.4	56.4	58.0	57.4	82.6	77.8	63.5	56.5	68.3
FreqNet	49.4	59.5	48.8	51.6	49.3	48.5	47.1	50.6	50.6	50.9	55.9	62.3	53.4	51.2	50.0	53.4	48.7	51.8
NPR	58.7	54.8	52.1	53.3	51.8	53.3	50.8	50.2	51.3	53.5	53.8	54.0	53.4	50.8	54.7	52.7	49.7	52.9
Fatformer	65.6	53.4	65.2	50.9	62.1	51.8	74.5	56.0	58.1	92.0	55.3	58.0	67.7	90.0	65.9	87.5	52.0	65.1
ODDN	72.1	44.1	76.8	68.1	76.5	73.3	58.0	75.6	83.5	90.8	61.1	65.9	63.9	83.5	77.0	72.9	55.0	70.7
DDOC	86.9	64.1	85.6	53.1	83.5	57.7	73.6	72.7	77.0	93.8	57.1	60.7	73.5	94.2	85.9	92.0	55.2	<u>74.5</u>
Ours	89.2	73.6	92.3	60.0	88.8	67.3	79.7	78.6	81.5	92.9	58.2	54.0	75.4	91.5	91.9	94.2	54.3	**77.9**

consisting of (1) *GAN-based generators*: ProGAN Karras et al. (2018), StyleGAN Karras et al. (2019), StyleGAN2 Karras et al. (2020), BigGAN Brock et al. (2019), CycleGAN Zhu et al. (2017), StarGAN Choi et al. (2018), GauGAN Park et al. (2019) and WFIR; (2) *Diffusion-based generators*: ADM Dhariwal & Nichol (2021), GLIDE, MidJourney, SD-v1.4 Rombach et al. (2022), SD-v1.5 Rombach et al. (2022), VQDM, WuKong and DALL·E 2 Ramesh et al. (2021).

Compression Robustness. We evaluate compression robustness following the protocol of prior works Tao et al. (2025); Li et al. (2026), which simulates JPEG compression commonly applied on social media platforms. The **training set** is constructed from ProGAN-generated images. To study the effect of semantic diversity, we use two training settings: 2-class (horse and chair) and 4-class (horse, chair, cat, and car). The training data is split into 80% unpaired pristine images and 20% paired data, where each image is paired with its JPEG-compressed version at quality factor 40. The **test scenarios** include quality-aware and quality-agnostic settings. All test images are JPEG-compressed in both evaluation settings. The quality-aware setting uses the same JPEG quality factor as the training data, whereas the quality-agnostic setting randomly samples the quality factor to simulate varying compression during image sharing on online social media platforms.

Evaluation Metrics. For cross-generator generalization, we follow AIGCDetectBench Zhong et al. (2023) and report mean accuracy, standard deviation (Std), and reliability score (Rel.). The reliability score is computed as follows:

$$\text{Rel.} = \frac{\text{mACC} - A_{\text{base}}}{\text{Std}}, \tag{12}$$

where $A_{\text{base}} = 50$ is the accuracy of a random predictor on the balanced binary task.

Table 4: Evaluation results for the 4-class training setting under the quality-aware setting.

Method	InfoMax-GAN	BE-GAN	Cramer-GAN	Att-GAN	MMD-GAN	Rel-GAN	S3-GAN	SNG-GAN	STG-GAN	Pro-GAN	Style-GAN	Style-GAN2	Big-GAN	Cycle-GAN	Star-GAN	Gau-GAN	Deep-fake	Mean Acc.
MesoNet	49.5	46.2	52.6	51.3	53.0	53.8	50.4	51.8	54.2	53.3	49.6	53.9	55.1	50.9	52.3	51.7	45.0	51.4
FF++	69.5	26.9	80.3	66.8	79.2	69.9	56.2	75.1	84.4	93.6	62.5	60.8	58.5	80.9	78.5	71.0	52.8	68.7
F3Net	61.0	41.9	65.8	52.9	63.8	55.5	53.8	59.6	71.5	92.2	76.0	59.1	55.9	57.9	71.8	66.0	52.1	62.4
MAT	57.9	46.9	64.2	50.8	63.4	52.4	52.1	56.2	61.8	90.8	54.2	53.9	52.4	73.1	61.4	64.8	51.2	59.5
SBIs	60.2	55.7	74.4	50.2	67.1	54.6	61.4	53.0	57.2	96.0	57.4	53.0	55.4	77.6	60.1	74.9	50.6	62.5
ADD	51.7	50.7	57.3	51.3	55.9	52.4	45.2	51.2	52.4	73.5	49.9	50.1	52.2	70.7	54.4	66.4	51.2	55.3
QAD	79.9	37.5	79.5	67.4	76.8	71.7	58.0	79.0	83.5	92.7	64.7	68.7	64.0	81.8	80.3	66.3	52.9	70.9
FreqNet	49.6	50.0	50.9	51.8	49.9	48.6	48.1	50.4	51.6	49.9	56.8	54.7	52.0	49.2	50.3	55.1	49.2	54.2
NPR	57.8	43.2	53.6	65.1	53.9	66.5	53.1	55.7	55.2	73.7	60.2	64.7	60.2	71.2	60.1	59.8	50.8	59.1
Fatformer	70.7	57.9	73.0	51.9	68.5	52.4	73.5	61.7	61.3	98.3	74.2	65.3	67.6	84.2	69.9	87.5	56.9	69.1
ODDN	80.6	38.6	80.7	65.8	78.8	71.1	60.5	76.7	85.8	94.0	67.7	69.9	66.7	84.9	80.5	75.2	54.2	72.6
DDOC	84.7	70.1	87.9	54.6	85.8	57.2	74.6	74.6	73.4	99.4	69.6	67.3	74.4	91.5	84.7	93.3	57.9	<u>76.5</u>
Ours	91.9	84.4	94.1	72.2	92.8	78.2	83.9	87.3	89.5	98.5	73.5	61.0	80.7	91.2	95.7	96.0	66.0	**84.5**

Table 5: Evaluation results for the 4-class training setting under the quality-agnostic setting.

Method	InfoMax-GAN	BE-GAN	Cramer-GAN	Att-GAN	MMD-GAN	Rel-GAN	S3-GAN	SNG-GAN	STG-GAN	Pro-GAN	Style-GAN	Style-GAN2	Big-GAN	Cycle-GAN	Star-GAN	Gau-GAN	Deep-fake	Mean Acc.
MesoNet	58.7	45.4	63.5	62.9	62.0	50.2	48.7	58.4	64.1	55.4	52.0	48.1	53.7	63.2	62.0	49.6	51.8	54.3
FF++	68.9	29.9	82.0	63.3	80.4	67.2	55.5	75.4	82.0	93.0	61.1	59.8	57.9	80.1	78.6	67.3	51.9	67.9
F3Net	62.0	43.4	65.8	53.2	64.1	56.7	55.4	58.8	67.7	92.5	76.6	62.3	56.8	60.5	71.0	71.3	51.1	63.4
MAT	52.2	49.3	62.5	50.6	60.3	51.7	53.3	53.9	58.6	92.2	54.4	54.9	54.0	76.5	59.4	68.4	51.0	59.4
SBIs	61.3	57.4	74.8	50.3	67.5	54.6	61.5	53.2	57.1	95.9	57.2	52.9	55.4	78.3	59.3	74.6	50.7	62.6
ADD	51.0	50.2	54.4	50.3	53.4	50.7	46.2	50.5	50.9	75.8	51.4	51.6	52.7	72.6	52.3	66.4	50.7	55.0
QAD	76.7	46.4	79.6	68.5	77.1	73.6	58.3	76.3	81.0	90.2	65.3	71.3	64.6	81.8	77.1	66.7	55.1	71.0
FreqNet	49.0	60.4	49.2	51.0	49.8	48.5	46.4	50.2	50.6	50.5	55.6	62.2	53.4	50.0	50.7	53.1	49.3	51.8
NPR	56.6	45.2	52.1	61.9	53.1	67.9	52.9	54.2	53.3	72.8	60.3	65.2	60.3	64.6	60.5	60.8	51.8	58.4
Fatformer	70.6	58.3	71.7	51.6	67.7	52.3	72.8	61.2	61.6	98.2	72.8	64.9	67.3	86.5	69.6	87.2	56.5	68.9
ODDN	80.4	35.1	81.0	68.7	78.2	74.5	62.2	77.5	81.7	91.7	69.2	70.4	68.0	78.8	73.4	73.8	55.3	72.1
DDOC	84.7	69.5	87.0	53.5	84.6	56.1	74.0	73.9	73.4	99.5	65.9	64.3	72.4	91.5	84.0	93.1	56.9	<u>75.5</u>
Ours	92.8	82.7	94.3	66.7	92.7	72.8	80.8	85.9	87.0	98.3	66.8	56.2	76.2	86.9	96.0	95.1	60.2	**81.8**

Implementation Details. During training, all input images are first resized to 256×256, and a random crop of size 224×224 is then sampled as the network input. During inference, each image is first resized to 256×256 and then center-cropped to 224×224. The model is optimized using Adam with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. The initial learning rate is set to 1×10^{-4}, and training is conducted for 1 epoch with a batch size of 96. The loss-balancing weights are set to $\eta = 0.01$, $\eta_{\tan} = 0.01$, and $\eta_{\text{bnd}} = 0.01$ in Eq. equation 11.

4.2 Experimental Results

Cross-Generator Generalization Evaluation. As shown in Table 1, the PTBP-trained detector achieves the highest mean accuracy of 94.76% across 16 generators after one training epoch. It exceeds AIDE and SimLBR by 1.99 and 6.36 percentage points, respectively. Our method achieves the second-best standard deviation 7.84 and reliability score 5.71. AIDE, the runner-up in mean accuracy, has a higher standard deviation 7.92 and a lower reliability score 5.40. SimLBR leads on these two metrics but achieves a lower mean accuracy of 88.40%. These comparisons indicate a favorable balance between detection accuracy and consistency across generators.

Compression Robustness Evaluation. Tables 2, 3, 4, and 5 show that the PTBP-trained detector achieves the highest mean accuracy in all four compression settings. Under 2-class training, mean accuracies reach 80.7% and 77.9% in the quality-aware and quality-agnostic settings, exceeding DDOC by 5.3% and 3.4%, respectively. Under 4-class training, the corresponding accuracies increase to 84.5% and 81.8%, exceeding DDOC by 8.0% and 6.3%. These gains persist under both fixed and random JPEG quality factors, supporting the effectiveness of PTBP under compression.

4.3 Ablation Studies

To evaluate the contribution of the PTBP components, Table 6 compares four variants under the cross-generator training protocol. These comparisons first assess the contribution of OSD relative to the baseline and then evaluate the additional effect of controlled perturbation beyond OSD. C_b^g tests

Table 6: Ablation studies of PTBP components. C_b denotes the baseline detector, C_b^g denotes the baseline with unstructured random Gaussian noise, C_b/C_o denotes the baseline augmented with orthogonal subspace decomposition, and $C_b/C_o/C_c$ denotes the baseline augmented with both orthogonal subspace decomposition and controlled feature perturbation, corresponding to the full model.

Method	Pro-GAN	Style-GAN	Big-GAN	Cycle-GAN	Star-GAN	Gau-GAN	Style-GAN2	WFIR	ADM	Glide	Mid-Journey	SD-v1.4	SD-v1.5	VQDM	Wu-Kong	Dalle2	mean
C_b	99.96	88.47	95.68	97.54	98.20	96.16	82.47	88.10	79.66	88.94	57.13	76.78	76.92	89.42	83.24	59.55	84.89
C_b^g	99.96	88.13	95.83	97.99	98.45	96.29	83.12	88.55	79.53	89.11	57.13	76.43	76.72	89.03	82.95	60.20	84.96
C_b/C_o	100.00	99.77	76.02	89.52	99.92	65.46	98.75	73.30	94.46	96.52	73.80	96.90	96.82	96.56	96.56	97.05	90.71
$C_b/C_o/C_c$	100.00	99.20	90.03	97.35	99.52	84.37	98.53	93.05	94.13	98.69	69.92	98.95	98.89	97.89	98.62	97.05	94.76

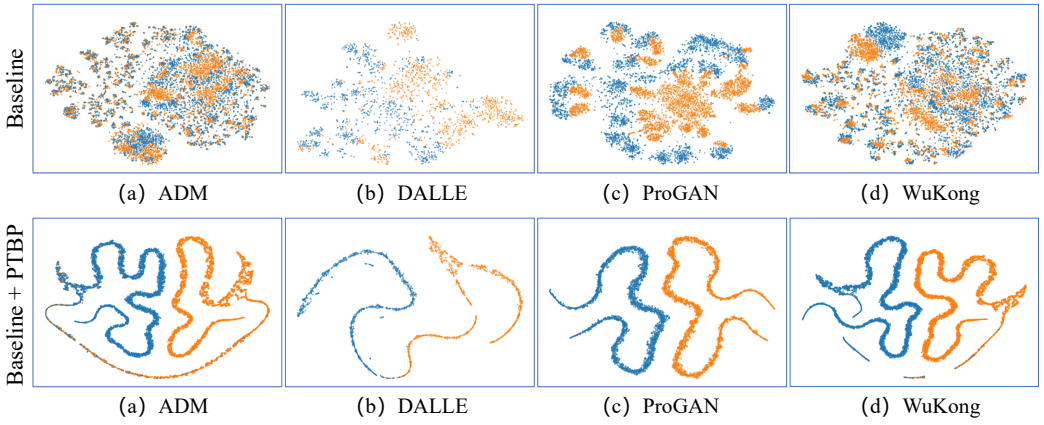


Figure 3: The t-SNE visualization of learned feature distribution. The top and bottom rows show the baseline and our detector, respectively. Blue and orange denote real and fake samples, respectively. More t-SNE comparisons are shown in Appendix.

whether unstructured Gaussian noise alone is sufficient, whereas C_b/C_o isolates the contribution of orthogonal subspace decomposition. The full $C_b/C_o/C_c$ model increases mean accuracy from 84.89% to 94.76% relative to C_b, a gain of 9.87%. It improves accuracy on 13 of the 16 generators, with gains of 37.50% on DALL-E 2, 22.17% on SD v1.4, and 21.97% on SD v1.5. These results support the effectiveness and necessity of the proposed components.

4.4 Visualization

To qualitatively examine the feature distribution learned with PTBP, we visualize extracted features for four representative generators using t-SNE van der Maaten & Hinton (2008) (Figure 3). For a fair comparison, both the PTBP-trained detector and the baseline detector are trained for a single epoch under the same protocol. For the baseline detector, the real and fake feature distributions overlap substantially on three unseen generators (ADM, DALLE, and WuKong). By contrast, the PTBP-trained detector separates real and fake samples into visually distinct regions with little overlap. These qualitative results are consistent with PTBP learning more discriminative features through controlled perturbations and complement its quantitative cross-generator performance.

5 Conclusion

We propose **Prototype-Guided Tangent-Boundary Perturbation (PTBP)**, a feature-level perturbation method that introduces additional training variations while retaining supervision on the original features. Its orthogonal subspace decomposition decomposes Gaussian noise into components orthogonal and parallel to the discriminative direction. Controlled feature perturbation uses these components to construct tangent views and boundary views. Corresponding tangent consistency encourages prediction consistency, while boundary supervision trains the boundary views with their original binary labels. With only one training epoch, PTBP achieves the best performance in cross-generator and compression-perturbed scenarios.

AI USE STATEMENT

In this work, we used generative AI tools for language editing, including improving grammar, clarity, concision, and overall readability of the manuscript. We have not used generative AI tools to generate research ideas, design experiments, collect or analyze data, create figures, or draw scientific conclusions. Additionally, no other uses of generative AI tools requiring disclosure were applicable to this work. All AI-assisted edits were reviewed and verified by the authors, who retained full responsibility for the accuracy, originality, and integrity of the manuscript.

REFERENCES

Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *ICLR*, 2019.

Liang Chen, Yong Zhang, Yibing Song, Lingqiao Liu, and Jue Wang. Self-supervised learning of adversarial example: Towards good generalizations for deepfake detection. In *CVPR*, pp. 18689–18698, 2022.

Yunjey Choi, Min-Je Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *CVPR*, pp. 8789–8797, 2018.

Aayush Dhakal, Subash Khanal, Srikumar Sastry, Jacob Arndt, Philipe Ambrozio Dias, Dalton Lunga, and Nathan Jacobs. Simlbr: Learning to detect fake images by learning to detect real images. In *CVPR*, 2026.

Prafulla Dhariwal and Alexander Quinn Nichol. Diffusion models beat gans on image synthesis. In *NeurIPS*, pp. 8780–8794, 2021.

Joel Frank, Thorsten Eisenhofer, Lea Schönherr, Asja Fischer, Dorothea Kolossa, and Thorsten Holz. Leveraging frequency analysis for deep fake image recognition. In *ICML*, volume 119, pp. 3247–3258, 2020.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, et al. Generative adversarial networks. In *NeurIPS*, pp. 2672–2680, 2014.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, volume 33, pp. 6840–6851, 2020.

Yonghyun Jeong, Doyeon Kim, Youngmin Ro, and Jongwon Choi. Frepgan: Robust deepfake detection using frequency-level perturbations. In *AAAI*, pp. 1060–1068, 2022.

Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. In *ICLR*, 2018.

Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, pp. 4396–4405, 2019.

Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *CVPR*, pp. 8107–8116, 2020.

Chenqi Kong, Anwei Luo, Peijun Bao, Yi Yu, Haoliang Li, Zengwei Zheng, Shiqi Wang, and Alex C. Kot. Moe-ffd: Mixture of experts for generalized and parameter-efficient face forgery detection. *IEEE Transactions on Dependable and Secure Computing*, 23(1):82–96, 2026.

Christos Koutlis and Symeon Papadopoulos. Leveraging representations from intermediate encoder-blocks for synthetic image detection. In *ECCV*, volume 15130, pp. 394–411, 2024.

Hanzhe Li, Jiaran Zhou, Yuezun Li, Baoyuan Wu, Bin Li, and Junyu Dong. Freqblender: Enhancing deepfake detection by blending frequency knowledge. In *NeurIPS*, 2024.

Kai Li, Wei Wang, Linchao Zhang, Siying Zhu, and Wenqi Ren. Decision-driven orthogonal learning with complementary feature mining for robust synthetic image detection. In *AAAI*, pp. 6271–6278, 2026.

Lingzhi Li, Jianmin Bao, Ting Zhang, Hao Yang, Dong Chen, Fang Wen, and Baining Guo. Face x-ray for more general face forgery detection. In *CVPR*, pp. 5000–5009, 2020.

Yuchen Luo, Yong Zhang, Junchi Yan, and Wei Liu. Generalizing face forgery detection with high-frequency features. In *CVPR*, pp. 16317–16326, 2021.

Utkarsh Ojha, Yuheng Li, and Yong Jae Lee. Towards universal fake image detectors that generalize across generative models. In *CVPR*, pp. 24480–24489, 2023.

Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *CVPR*, pp. 2337–2346, 2019.

Yuyang Qian, Guojun Yin, Lu Sheng, Zixuan Chen, and Jing Shao. Thinking in frequency: Face forgery detection by mining frequency-aware clues. In *ECCV*, pp. 86–103, 2020.

Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *ICML*, volume 139, pp. 8821–8831, 2021.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, pp. 10674–10685, 2022.

Kaede Shiohara and Toshihiko Yamasaki. Detecting deepfakes with self-blended images. In *CVPR*, pp. 18720–18729, 2022.

Chuangchuang Tan, Yao Zhao, Shikui Wei, Guanghua Gu, and Yunchao Wei. Learning on gradients: Generalized artifacts representation for gan-generated images detection. In *CVPR*, pp. 12105–12114, 2023.

Chuangchuang Tan, Huan Liu, Yao Zhao, Shikui Wei, Guanghua Gu, Ping Liu, and Yunchao Wei. Rethinking the up-sampling operations in cnn-based generative network for generalizable deepfake detection. In *CVPR*, pp. 28130–28139, 2024a.

Chuangchuang Tan, Yao Zhao, Shikui Wei, Guanghua Gu, Ping Liu, and Yunchao Wei. Frequency-aware deepfake detection: Improving generalizability through frequency space domain learning. In *AAAI*, pp. 5052–5060, 2024b.

Renshuai Tao, Manyi Le, Chuangchuang Tan, Huan Liu, Haotong Qin, and Yao Zhao. ODDN: addressing unpaired data challenges in open-world deepfake detection on online social networks. In *AAAI*, pp. 799–807, 2025.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008.

Zhiyuan Yan, Yuhao Luo, Siwei Lyu, Qingshan Liu, and Baoyuan Wu. Transcending forgery specificity with latent space augmentation for generalizable deepfake detection. In *CVPR*, pp. 8984–8994, 2024.

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 *ICML*, 2025.

Xu Zhang, Svebor Karaman, and Shih-Fu Chang. Detecting and simulating artifacts in GAN fake images. In *WIFS*, pp. 1–6, 2019.

Tianchen Zhao, Xiang Xu, Mingze Xu, Hui Ding, Yuanjun Xiong, and Wei Xia. Learning self-consistency for deepfake detection. In *ICCV*, pp. 15003–15013, 2021.

Nan Zhong, Yiran Xu, Zhenxing Qian, and Xinpeng Zhang. Patchcraft: Exploring texture patch for efficient ai-generated image detection. *Arxiv*, 2023.

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, pp. 2242–2251, 2017.

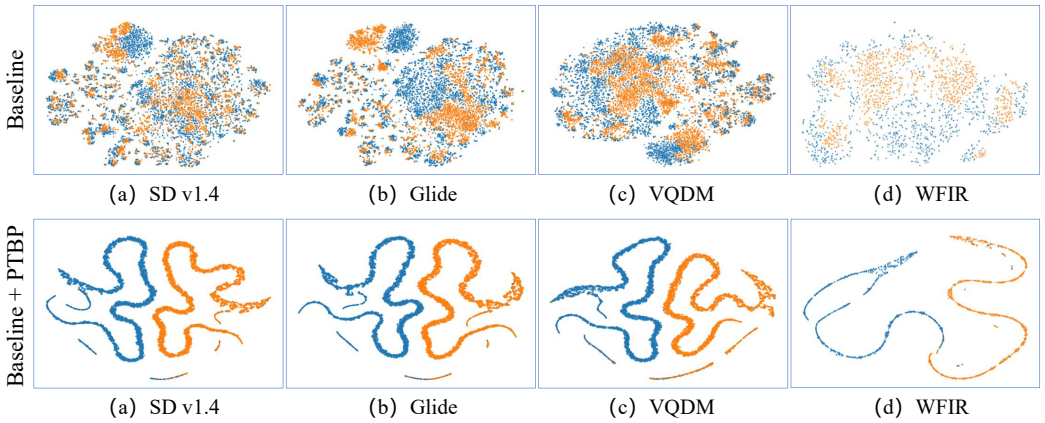


Figure 4: Additional t-SNE visualizations of the learned feature distributions.

A Appendix

A.1 Additional Visualizations

Figure 4 complements the t-SNE analysis with four additional generators (SD v1.4, Glide, VQDM, and WFIR) under the same 1-epoch training protocol. Consistent with the main results, the baseline exhibits heavily entangled real and fake clusters, whereas adding PTBP yields well-structured manifolds with clearly separated real and fake trajectories. The effect is particularly evident on WFIR, where the two classes form distinct and sparsely distributed arcs. Together with the visualizations in the main text, these results indicate that PTBP improves feature separability in a manner consistent across generators.