HYPERBOLIC LIVE-MANIFOLD LEARNING FOR GENERALIZABLE FACE ANTI-SPOOFING

Anonymous authors
Paper under double-blind review

ABSTRACT

Generalizable face anti-spoofing (FAS) aims to learn liveness representations that remain discriminative across unseen domains. Most methods optimize live and spoof samples symmetrically in Euclidean space, overlooking their structural asymmetry: live samples tend to share coherent liveness properties, whereas spoof samples vary substantially across attack types and domains. Such symmetric modeling makes it difficult to preserve a stable live structure while accommodating diverse spoof patterns. To address this issue, we propose HypLive-FAS, which organizes live representations into a compact manifold in hyperbolic space and identifies spoof samples through their deviations from this manifold. The expanding capacity of hyperbolic space away from the live region allows heterogeneous spoof patterns to be accommodated without forcing them into a unified cluster. Specifically, Hyperbolic Live-Manifold Learning (HLM) constructs a live-centered manifold primarily from live samples while encouraging spoof samples to remain separated from it, thereby reducing their interference with the intrinsic live structure. While HLM captures this asymmetric class structure, the learned manifold may still drift under domain-specific appearance variations. We therefore introduce Liveness-Preserving Augmentation (LPA), which perturbs domain-dependent appearance while enforcing representation- and prediction-level consistency between original and augmented samples, improving the stability of the learned liveness structure. Extensive experiments on multiple benchmarks demonstrate that HypLive-FAS achieves competitive cross-domain generalization and consistently outperforms existing methods.

1 INTRODUCTION

Face anti-spoofing (FAS) (Boulkenafet et al., 2015; Liu et al., 2018b; Yu et al., 2020b), which distinguishes live faces from presentation attacks, is essential to the security of face recognition systems. Existing methods (Zhou et al., 2024; Guo et al., 2024; Liu et al., 2026b) have achieved promising performance against common presentation attacks. However, in real-world deployment, FAS models often face substantial domain shifts arising from variations in acquisition devices, illumination conditions, environments, and attack media (Shao et al., 2019; Jia et al., 2020; Sun et al., 2023). Consequently, models trained on limited source domains may overfit to domain-specific appearances or known spoof patterns, leading to severe performance degradation on unseen domains and novel attacks (Sun et al., 2023; Zhou et al., 2023). Learning domain-invariant representations that capture intrinsic liveness cues is therefore crucial for generalizable FAS (Hu et al., 2024; Huang et al., 2024; Le & Woo, 2024).

Existing domain-generalized FAS methods seek domain-invariant representations through domain alignment (Shao et al., 2019; Mao & Li, 2025; Yang et al., 2025), meta-learning (Shao et al., 2020; Liu et al., 2021; Zhou et al., 2022b), and data augmentation (Wang et al., 2022b; Zhou et al., 2023). However, most retain a closed-set formulation, learning the representation space and decision boundary jointly from finite source-domain live and spoof samples. As illustrated in Fig. 1(a), live samples share relatively coherent authenticity characteristics, whereas spoof attacks are heterogeneous and continuously evolving (Jia et al., 2020; Mao & Li, 2025). Source-domain spoof samples therefore provide only a partial characterization of the attack space. As a result, decision boundaries optimized for these observed spoof patterns may fail to cover unseen attacks, causing them to be misclassified as live samples and degrading cross-domain generalization (Yu et al., 2022).

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.

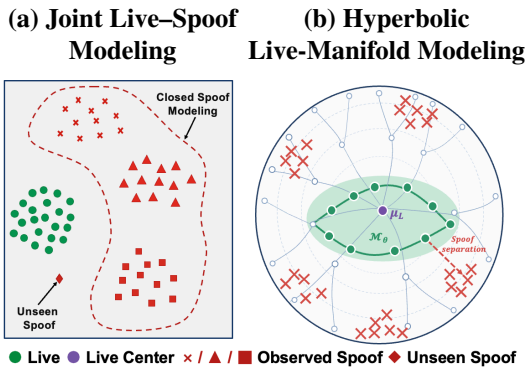


Figure 1: (a) Joint live-spoof modeling fits a closed region (red dashed boundary) around observed attacks (crosses, triangles, and squares), potentially missing unseen attacks (red diamond) near live samples (green circles). (b) HypLive-FAS learns a compact hyperbolic live manifold $\mathcal{M}_\theta$ with center $\boldsymbol{\mu}_L$ (purple) where spoof samples are excluded from direct manifold fitting and provide outward margin supervision.

To address this limitation, we propose HypLive-FAS, which shifts the explicit geometric modeling focus from joint class-structure modeling to live-centered manifold modeling, as illustrated in Fig. 1(b). Live images share stable intrinsic cues and form a compact low-dimensional structure (Jia et al., 2020), whereas spoof attacks exhibit scattered and open-ended distributions (Yu et al., 2022; Mao & Li, 2025). With exponentially increasing capacity away from its center (Nickel & Kiela, 2017; 2018), hyperbolic space provides a suitable geometry for representing this compact live core and accommodating unseen attacks as deviations from the live manifold.

HypLive-FAS comprises Hyperbolic Live-Manifold Learning (HLM) and Liveness-Preserving Augmentation (LPA). Global and prompt-guided local features extracted by the CLIP visual and text encoders are first fused and mapped into a Poincaré ball. Given the hyperbolic embeddings, HLM uses source-domain live samples to define the manifold geometry. Source spoof samples are excluded from manifold fitting and serve only as outward supervision, encouraging their representations to move away from the learned live manifold. It uses local Jacobian geometry to explain manifold-consistent variation and treats the remaining regularized deviation as the spoof score. LPA generates strongly perturbed source views and preserves both representation directions and manifold-deviation scores, reducing reliance on domain-specific appearance cues.

Our main contributions are summarized as follows:

- **Local nonlinear live-manifold modeling:** We propose HypLive-FAS, which learns a compact live manifold and detects spoofing through a Jacobian-based regularized deviation score, distinguishing live-consistent variations from unexplained spoof deviations.
- **Liveness-preserving augmentation:** We introduce LPA, which preserves representation directions and manifold-deviation scores under appearance perturbations to improve robustness against domain-specific variations.
- **Comprehensive evaluation:** We conduct extensive experiments and ablation studies on multiple cross-domain FAS benchmarks to validate the effectiveness of HypLive-FAS.

2 Related Work

2.1 Face Anti-Spoofing

Face anti-spoofing (FAS) aims to distinguish live faces from increasingly sophisticated attacks. Early methods relied on handcrafted texture, color, and motion features (Yang et al., 2013; Boulkenafet et al., 2015; 2016; Patel et al., 2016), and recent studies employ CNNs and transformers for end-to-end detection (Yu et al., 2020b; Wang et al., 2022a; 2024). Auxiliary supervision signals(Liu et al., 2018b; Wang et al., 2021; Liu et al., 2018a), including depth maps, reflection maps, and pixel-level spoof cues, have also been introduced to improve the detection of presentation attacks. However, these methods often struggle with domain shifts as the learned features may depend heavily on domain-specific information, such as dataset backgrounds, identity-related cues, and device-dependent artifacts.

To improve FAS generalization to unseen domains, domain generalization methods aim to learn transferable liveness representations from source domains. MADDG (Shao et al., 2019) and SSDG (Jia et al., 2020) employ adversarial learning to mitigate domain discrepancies, while SSAN

(Wang et al., 2022b), SA-FAS (Sun et al., 2023), and IADG (Zhou et al., 2023) improve generalization through style recombination, transition-direction alignment, and instance-style suppression, respectively. With the emergence of vision-language models (VLMs) such as Contrastive Language-Image Pre-training (CLIP) (Radford et al., 2021) and prompt-learning approaches such as CoOp (Zhou et al., 2022a), more recent methods like FLIP (Srivatsan et al., 2023), CFPL-FAS (Liu et al., 2024) and MVP-FAS (Yu et al., 2025) incorporate language-derived semantic priors to guide visual representation learning. Our method instead learns a live manifold and employs liveness-preserving augmentation to improve robustness to unseen domain shifts.

2.2 Hyperbolic Representation Learning for FAS

Hyperbolic space has attracted increasing attention in representation learning because of its negative curvature and exponential volume growth (Nickel & Kiela, 2017; 2018). Early studies developed hyperbolic neural networks (Ganea et al., 2018b), attention mechanisms (Gulcehre et al., 2018), and entailment cones (Ganea et al., 2018a) to capture hierarchical structures. Recent vision-language methods, such as MERU (Desai et al., 2023) and HyCoCLIP (Pal et al., 2025), embed multimodal data in hyperbolic space to model hierarchical and compositional relations. HPEC (Fonio et al., 2025) combines prototypical learning with hyperbolic entailment cones for image classification. HIDISC (Rathore et al., 2026) introduces a hyperbolic framework for domain generalization with generalized category discovery.

Hyperbolic geometry has also been explored in FAS. Early approaches combine Poincaré embeddings with hyperbolic classifiers and contrastive objectives to improve feature discrimination (Han et al., 2023). HPDR (Hu et al., 2024) employs hierarchical prototypes to refine cross-domain feature distributions, while HML-FAS (Mao & Li, 2025) integrates hyperbolic adversarial augmentation and optimal transport to learn robust and domain-invariant representations. However, these methods mainly rely on global boundaries, prototypes, centers, or distribution alignment, without explicitly modeling sample-dependent local geometry. Relatedly, GFG (Yu et al.) explores decoder-Jacobian tangent geometry and projection residuals for off-manifold consistency. In contrast, HypLive-FAS learns a PCA-anchored nonlinear live manifold and measures spoofness through primitive-guided regularized local deviation.

3 Method

Figure 2 illustrates HypLive-FAS, which embeds visual features and their liveness-preserving augmentations into the Poincaré ball to learn a live manifold. Samples are classified by their regularized deviation from the locally fitted manifold.

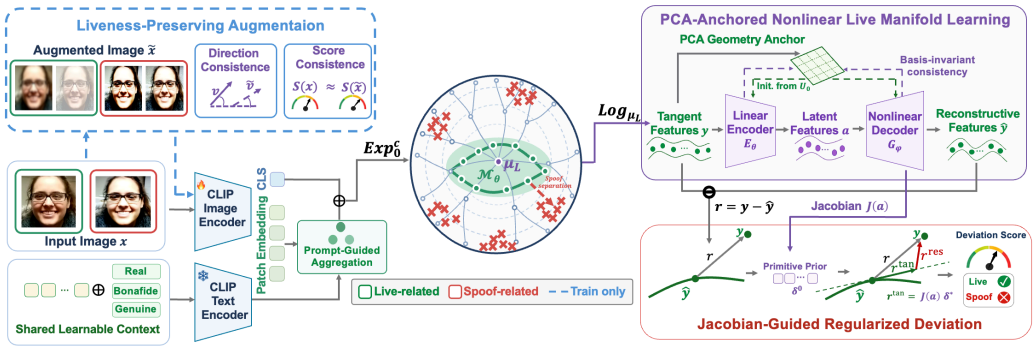


Figure 2: The overall framework of HypLive-FAS.

3.1 Preliminary: Hyperbolic Geometry

We operate in the D-dimensional Poincaré ball $\mathbb{D}_c^D = \{\mathbf{x} \in \mathbb{R}^D : c\|\mathbf{x}\|_2^2 < 1\}$ with curvature $-c$. We denote its geodesic distance by $d_c(\cdot,\cdot)$ and use $\mathrm{Exp}_{\mathbf{0}}^c(\cdot)$ and $\mathrm{Log}_{\mathbf{x}}^c(\cdot)$ to map between the

manifold and its tangent spaces. We map tangent vectors to the Poincaré ball using $\mathrm{Exp}_{\mathbf{0}}^{c}(\mathbf{v}) = \tanh(\sqrt{c}\|\mathbf{v}\|_2)\frac{\mathbf{v}}{\sqrt{c}\|\mathbf{v}\|_2}$. Conversely, the logarithmic map is

$$\mathrm{Log}_{\mathbf{x}}^{c}(\mathbf{y}) = \frac{2}{\sqrt{c}\lambda_{\mathbf{x}}^{c}} \operatorname{arctanh}\left(\sqrt{c}\,\|(-\mathbf{x}) \oplus_c \mathbf{y}\|_2\right) \frac{(-\mathbf{x}) \oplus_c \mathbf{y}}{\|(-\mathbf{x}) \oplus_c \mathbf{y}\|_2}. \tag{1}$$

More knowledge about hyperbolic geometry is provided in the Appendix.

3.2 Visual Representation and Hyperbolic Embedding

We employ the fine-tuned CLIP image encoder to extract the global CLS feature $\mathbf{c}_i$ and patch features $\mathbf{P}_i$. Inspired by MVP-FAS (Yu et al., 2025), we use three live prompts, "real face", "bonafide face", and "genuine face", with a shared eight-token CoOp context (Zhou et al., 2022a). Their normalized text features from the frozen CLIP text encoder $\{\mathbf{t}_k\}_{k=1}^{3}$ guide patch aggregation into liveness-aware local features $\{\mathbf{z}_{i,k}\}_{k=1}^{3}$. These local features are then fused with the global CLS feature as $\mathbf{f}_i = \Psi_{\mathrm{r}}(\mathbf{c}_i + \gamma F_{\mathrm{fuse}}([\mathbf{c}_i; \mathbf{z}_{i,1}; \mathbf{z}_{i,2}; \mathbf{z}_{i,3}]))$, where F_{fuse} integrates the global and local features, γ is a learnable gate, and Ψ_{r} denotes the visual mapping. Let $\mathbf{v}_i = \mathbf{f}_i/\|\mathbf{f}_i\|_2$ and $\bar{\mathbf{t}}_L = \frac{1}{3}\sum_{k=1}^{3}\mathbf{t}_k$.We preserve liveness semantics by aligning source-live visual directions with $\bar{\mathbf{t}}_L$:

$$\mathcal{L}_{\mathrm{sem}} = \frac{1}{|\mathcal{S}_L|}\sum_{i\in\mathcal{S}_L}\left(1 - \mathbf{v}_i^{\top}\bar{\mathbf{t}}_L\right), \tag{2}$$

where $\mathcal{S}_L$ denotes the set of source live samples.

To mitigate boundary-induced numerical instability and vanishing gradients (Guo et al., 2022b; Mishne et al., 2023), we rescale features by the median source-live norm $r_{\mathrm{ref}} = \mathrm{median}_{j\in\mathcal{S}_L}|\mathbf{f}_j|_2$ before Poincaré mapping: $\mathbf{h}_i = \mathrm{Exp}_{\mathbf{0}}^{c}\left(\rho_0 \frac{\mathbf{f}_i}{\mathrm{sg}(r_{\mathrm{ref}})}\right) \in \mathbb{D}_c^D$, Here, $\mathrm{sg}(\cdot)$ denotes stop-gradient and ρ_0 controls radial scaling.

3.3 Hyperbolic Live-Manifold Learning

Live-centered tangent geometry. The live manifold is constructed from source-live embeddings $\{\mathbf{h}_i\}_{i\in\mathcal{S}_L}$, while spoof samples update only the visual backbone to push their representations away from the manifold. We compute the source-live Fréchet center as $\boldsymbol{\mu}_L = \arg\min_{\boldsymbol{\mu}\in\mathbb{D}_c^D}\sum_{i\in\mathcal{S}_L} d_c^2(\boldsymbol{\mu}, \mathbf{h}_i)$ and then the embeddings are mapped to the tangent space at $\boldsymbol{\mu}_L$ and centered as $\mathbf{u}_i = \mathrm{Log}_{\boldsymbol{\mu}_L}^{c}(\mathbf{h}_i)$ and $\mathbf{y}_i = \mathbf{u}_i - \bar{\mathbf{u}}_L$, where $\bar{\mathbf{u}}_L = |\mathcal{S}_L|^{-1}\sum_{j\in\mathcal{S}_L}\mathbf{u}_j$. PCA (Pearson, 1901) is then applied to $\{\mathbf{y}_i\}_{i\in\mathcal{S}_L}$, yielding the leading d eigenvectors $\mathbf{U}_0 \in \mathbb{R}^{D\times d}$, which define the initial live-geometry anchor in Fig. 2.

PCA-anchored adaptive live manifold. Since linear PCA cannot fully capture valid nonlinear live variations, we encode each tangent feature as $\mathbf{a}_i = E_\theta(\mathbf{y}_i) = \mathbf{W}_E\mathbf{y}_i$, and reconstruct it with a residual nonlinear decoder, $\widehat{\mathbf{y}}_i = G_\phi(\mathbf{a}_i) = \mathbf{W}_0\mathbf{a}_i + \mathbf{W}_2\,\mathrm{GELU}(\mathbf{W}_1\mathbf{a}_i + \mathbf{b}_1)$. We initialize $\mathbf{W}_E = \mathbf{U}_0^{\top}$, $\mathbf{W}_0 = \mathbf{U}_0$ and $\mathbf{W}_2 = \mathbf{0}$, allowing the manifold to start from the PCA geometry and progressively adapt to nonlinear live variations. The adaptive manifold is learned using

$$\mathcal{L}_{\mathrm{rec}} = \frac{1}{|\mathcal{S}_L|}\sum_{i\in\mathcal{S}_L}\left\|\mathrm{sg}(\mathbf{y}_i) - G_\phi(E_\theta(\mathrm{sg}(\mathbf{y}_i)))\right\|_{\boldsymbol{\mu}_L}^{2}, \tag{3}$$

With $\mathbf{y}_i$ detached, $\mathcal{L}_{\mathrm{rec}}$ updates only the manifold encoder and decoder. To prevent the learned manifold from drifting excessively from its PCA initialization, we further impose

$$\mathcal{L}_{\mathrm{sub}} = 1 - \frac{\|\mathbf{U}_0^{\top}\mathbf{Q}_E\|_F^2 + \|\mathbf{U}_0^{\top}\mathbf{Q}_G\|_F^2}{2d}, \tag{4}$$

where $\mathbf{Q}_E$ and $\mathbf{Q}_G$ are the orthonormal bases of the encoder and linear-decoder subspaces.

Jacobian-guided regularized deviation scoring. The reconstruction residual $\mathbf{r}_i = \mathbf{y}_i - \widehat{\mathbf{y}}_i$ may contain both off-manifold deviations and valid live variations arising from identity, illumination, or

imaging style. To account for valid live variations, we locally linearize the decoder as $G_\phi(\mathbf{a}_i+\boldsymbol{\delta}) \approx G_\phi(\mathbf{a}_i) + \mathbf{J}_i\boldsymbol{\delta}$, where

$$\mathbf{J}_i = \frac{\partial G_\phi(\mathbf{a}_i)}{\partial \mathbf{a}_i} = \mathbf{W}_0 + \mathbf{W}_2\,\mathrm{Diag}\big[\mathrm{GELU}'(\mathbf{W}_1\mathbf{a}_i + \mathbf{b}_1)\big]\,\mathbf{W}_1. \tag{5}$$

We fit the residual along these local tangent directions using $\mathbf{J}_i\boldsymbol{\delta}$. To stabilize fitting and incorporate source-live coordinate patterns, we introduce a ridge penalty and a dictionary of K learnable live variation primitives $\{\mathbf{p}_k\}_{k=1}^K$. The primitive encoder computes $\boldsymbol{\alpha}_i = \mathrm{softmax}(g_\psi(\mathbf{r}_i)/\tau)$ and the coordinate prior $\boldsymbol{\delta}_i^0 = \sum_{k=1}^K \alpha_{i,k}\mathbf{p}_k$. The coordinates are estimated by

$$\boldsymbol{\delta}_i^* = \arg\min_{\boldsymbol{\delta}} \|\mathbf{r}_i - \mathbf{J}_i\boldsymbol{\delta}\|_2^2 + s_i\lambda_\mathrm{r}\|\boldsymbol{\delta}\|_2^2 + s_i\lambda_\mathrm{p}\|\boldsymbol{\delta} - \boldsymbol{\delta}_i^0\|_2^2, \tag{6}$$

where λ_r and λ_p control the ridge and prior penalties. The detached scale $s_i = \mathrm{sg}\big(\max\{\mathrm{tr}(\mathbf{J}_i^\top\mathbf{J}_i)/d,\, 10^{-8}\}\big)$ adapts their strengths to the local Jacobian magnitude.

We define the fitted tangent component $\mathbf{r}_i^{\tan} = \mathbf{J}_i\boldsymbol{\delta}_i^*$ and the remaining residual $\mathbf{r}_i^{\mathrm{res}} = \mathbf{r}_i - \mathbf{r}_i^{\tan}$. The regularized deviation score is $S(\mathbf{x}_i) = \|\mathbf{r}_i^{\mathrm{res}}\|_{\boldsymbol{\mu}_L}$, with smaller values indicating a better fit under the ridge and coordinate-prior constraints. Because the residual need not be orthogonal to the local tangent directions, this score is not an exact distance to the live manifold.

Asymmetric manifold optimization. We constrain live samples within a tolerance radius R_L and push spoof samples beyond a margin m:

$$\mathcal{L}_{\mathrm{margin}} = \frac{1}{|\mathcal{S}_L|}\sum_{i\in\mathcal{S}_L}[S(\mathbf{x}_i) - R_L]_+^2 + \frac{\lambda_\mathrm{s}}{|\mathcal{S}_S|}\sum_{i\in\mathcal{S}_S}[m - S(\mathbf{x}_i)]_+^2, \tag{7}$$

where $[z]_+ = \max(z, 0)$, and $\lambda_\mathrm{s} = 1.0$ balances the spoof-margin term against the live-margin term. The two hyperparameters m and R_L in Eq. equation 7 are calibrated in score units from source-domain live samples rather than selected as arbitrary constants. Let $q_{0.95}^{(0)}$ and $q_{0.95}^{(1)}$ denote the 95th percentiles of the deviation scores at initialization and after the geometry refresh at the end of epoch 1, respectively. We initialize the spoof margin as $m = \max\Big(1.5q_{0.95}^{(0)},\, 0.05\Big)$, disable the live-radius term during epoch 1, and then set $R_L = q_{0.95}^{(1)}$. Both m and R_L remain fixed thereafter. Live-sample gradients update the visual backbone, live manifold, and primitive dictionary, whereas spoof-sample gradients update only the visual backbone.

The overall objective is

$$\mathcal{L}_{\mathrm{HLM}} = \mathcal{L}_{\mathrm{margin}} + \lambda_{\mathrm{rec}}\mathcal{L}_{\mathrm{rec}} + \lambda_{\mathrm{sub}}\mathcal{L}_{\mathrm{sub}} + \lambda_{\mathrm{sem}}\mathcal{L}_{\mathrm{sem}}, \tag{8}$$

3.4 Liveness-Preserving Augmentation

To reduce sensitivity to domain-specific appearances, we introduce Liveness-Preserving Augmentation (LPA), where we generate a label-preserving view $\widetilde{\mathbf{x}}_i = \mathcal{A}(\mathbf{x}_i)$ using appearance perturbations. As shown in Fig. 2, the augmented view is used only during training and follows the same feature extraction and hyperbolic embedding pipeline as the original image. To preserve liveness semantics, we first align the normalized visual directions between the original and augmented views to preserve liveness semantics:

$$\mathcal{L}_{\mathrm{dir}} = \frac{1}{B}\sum_{i=1}^{B}\left[1 - \cos\left(\mathrm{sg}(\mathbf{v}_i), \widetilde{\mathbf{v}}_i\right)\right], \tag{9}$$

where the detached original direction provides a stable optimization target. Furthermore, to maintain their geometric relationship with the learned live manifold, we align their regularized deviation scores:

$$\mathcal{L}_{\mathrm{score}} = \frac{1}{B}\sum_{i=1}^{B}\mathrm{SmoothL1}\left(\log(1+\mathrm{sg}(S(\mathbf{x}_i))), \log(1+S(\widetilde{\mathbf{x}}_i))\right). \tag{10}$$

The logarithmic transformation compresses the score range, while Smooth L1 reduces sensitivity to perturbations and outliers. The augmented view is also supervised by the HLM objective with the original label. The overall training objective is:

$$\mathcal{L} = \mathcal{L}_{\mathrm{HLM}}(\mathbf{x}) + \lambda_{\mathrm{aug}}\mathcal{L}_{\mathrm{HLM}}(\widetilde{\mathbf{x}}) + \lambda_{\mathrm{dir}}\mathcal{L}_{\mathrm{dir}} + \lambda_{\mathrm{score}}\mathcal{L}_{\mathrm{score}}. \tag{11}$$

More details about the whole methods are provided in the Appendix.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Datasets, protocols, and evaluation metrics. Following previous FAS studies (Zhang et al., 2025; 2026b; Liu et al., 2026b), we evaluate HypLive-FAS under two protocols. **Protocol 1: Leave-one-domain-out evaluation.** We employ MSU-MFSD (M) (Wen et al., 2015), CASIA-MFSD (C) (Zhang et al., 2012), Replay-Attack (I) (Chingovska et al., 2012), and OULU-NPU (O) (Boulkenafet et al., 2017). Three datasets are used for training and the remaining one for testing, yielding OCI→M, OMI→C, OCM→I, and ICM→O. **Protocol 2: One-to-Eleven evaluation.** We train solely on CelebA-Spoof (Zhang et al., 2020) and evaluate on 11 unseen datasets: MSU-MFSD (Wen et al., 2015), CASIA-MFSD (Zhang et al., 2012), Replay-Attack (Chingovska et al., 2012), OULU-NPU (Boulkenafet et al., 2017), SiW (Liu et al., 2018b), Rose-Youtu (Li et al., 2018), HKBU-MARs-V1+ (Liu et al., 2018a), WMCA (George et al., 2019), SiW-Mv2 (Guo et al., 2022a), CASIA-SURF-3DMask (Yu et al., 2020a), and HiFiMask (Liu et al., 2022). This protocol introduces a larger source-target gap and attack media unseen during training. As for the evaluation, we employ two standard metrics: Area Under the Curve (AUC) and Half Total Error Rate (HTER). The higher AUC and the lower HTER indicate better performance.

Implementation details. We use a pretrained CLIP ViT-B/16 encoder with 224×224 inputs. The visual feature dimension and live-manifold dimension are set to 512 and 32, respectively, with curvature $c = 1$ and initial radius $\rho_0 = 0.75$. The model is optimized using AdamW with learning rates of 3×10^{-6}, 5×10^{-5}, and 1×10^{-5} for the CLIP backbone, prompts, and manifold parameters, respectively. All results are averaged over three random seeds. Detailed hyperparameter settings are provided in the Appendix.

Table 1: Comparison under Protocol 1. We report the mean HTER↓ (%) and AUC↑ (%) over three seeds. Bold values indicate the best performance for each metric.

Methods	O&C&I → M		O&M&I → C		O&C&M → I		I&C&M → O		Average
	HTER	AUC	HTER	AUC	HTER	AUC	HTER	AUC	HTER
Methods using LLM-derived semantic priors									
I-FAS (Zhang et al., 2025)	0.32	99.88	0.04	99.99	3.22	98.48	1.74	99.66	1.33
Methods without LLM-derived semantic priors									
IADG (Zhou et al., 2023)	5.41	98.19	8.70	96.40	10.62	94.50	8.86	97.14	8.40
UDG-FAS (Liu et al., 2023)	5.95	98.47	9.82	96.76	5.86	98.62	10.97	95.36	8.15
SA-FAS (Sun et al., 2023)	5.95	96.55	8.78	95.37	6.58	97.54	10.00	96.23	7.82
FLIP (Srivatsan et al., 2023)	4.95	98.11	0.54	99.98	4.25	99.07	2.31	99.63	3.01
TTDG (Zhou et al., 2024)	4.16	98.48	7.59	98.18	9.62	98.18	10.00	96.15	7.84
HPDR (Hu et al., 2024)	4.58	96.02	11.30	94.42	11.26	92.49	9.93	95.26	9.26
GAC-FAS (Le & Woo, 2024)	5.00	97.56	8.20	95.16	4.29	98.87	8.60	97.16	6.52
CFPL (Liu et al., 2024)	1.43	99.28	2.56	99.10	5.43	98.41	2.50	99.42	2.98
S-CPTL (Guo et al., 2024)	1.25	99.35	0.52	99.90	5.84	98.78	3.95	99.72	2.89
HML-FAS (Mao & Li, 2025)	3.75	97.48	4.26	99.38	9.38	96.51	3.96	99.23	5.34
MVP-FAS (Yu et al., 2025)	1.71	**99.83**	0.75	**99.89**	1.99	99.59	2.82	99.49	1.82
DGPDL (Liu et al., 2026b)	4.76	98.63	0.66	99.88	3.50	99.35	**1.97**	**99.84**	2.72
ICPE-FAS (Liu et al., 2026a)	1.43	99.66	2.00	99.20	2.28	99.72	2.31	99.66	2.01
HypLive-FAS	**1.08**	99.28	**0.27**	99.88	**1.14**	**99.87**	2.28	99.54	**1.19**
Variation	±0.00	±0.20	±0.07	±0.06	±0.17	±0.05	±0.05	±0.03	±0.03†

4.2 COMPARISON RESULTS

Results on Protocol 1 Table 1 divides the compared methods according to whether they employ additional LLM-derived semantic priors. Large language models can provide rich high-level semantic knowledge related to liveness attributes and attack characteristics. Since the two groups use different sources of prior knowledge, we primarily conduct quantitative comparisons among methods without such priors. Within this group, HypLive-FAS achieves the best overall performance and obtains the lowest HTER on three of the four target domains. It reduces the previous best average HTER of MVP-FAS from 1.82% to 1.19%. Under the more challenging $O\&C\&M \rightarrow I$ setting, HypLive-FAS further achieves an HTER of 1.14% and an AUC of 99.87%. These results

demonstrate that explicitly modeling the intrinsic live manifold effectively improves cross-domain generalization without relying on additional knowledge from large language models.

Results on Protocol 2 Table 2 follows the same organization. Protocol 2 trains all models solely on CelebA-Spoof and evaluates them on 11 unseen target datasets, resulting in a larger source-to-target shift. The upper block includes I-FAS (Zhang et al., 2025), FaceCoT (Zhang et al., 2026b), and TAR-FAS (Zhang et al., 2026a), which leverage large language models to introduce additional high-level semantic knowledge. To evaluate performance under the same prior-information setting, we focus our quantitative analysis on methods without such priors. Within this group, HypLive-FAS achieves the best overall performance, reducing the average HTER of DGPDL from 9.55% to 8.01% while increasing the average AUC from 95.06% to 95.27%. It also obtains the lowest HTER on 7 of the 11 target datasets, including HTER values of 4.04%, 4.33%, and 7.23% on Replay-Attack, Rose-Youtu, and WMCA, respectively. These results demonstrate that live-manifold modeling effectively handles diverse unseen-domain variations and attack types without relying on additional knowledge from large language models.

Table 2: Comparison under Protocol 2. Models are trained solely on CelebA-Spoof and evaluated on 11 unseen datasets. We report the mean HTER↓ (%) and AUC↑ (%) over three seeds. Bold values indicate the best performance for each metric.

Methods	CASIA-MFSD		CASIA-SURF-3DMask		HKBU-MARs-V1+		HiFiMask	
	HTER	AUC	HTER	AUC	HTER	AUC	HTER	AUC
Methods using LLM-derived semantic priors								
I-FAS	1.11	99.88	6.18	98.40	18.64	88.77	28.23	77.17
FaceCoT	0.00	100.00	6.21	98.73	6.96	99.41	28.68	79.74
TAR-FAS	0.00	100.00	2.09	99.65	3.48	99.71	17.97	90.23
Methods without LLM-derived semantic priors								
FLIP	4.88	98.48	8.83	96.93	17.25	88.31	28.32	76.50
DGPDL	**0.00**	**100.00**	4.99	**98.53**	17.75	89.87	26.69	**81.11**
HypLive-FAS	1.60	99.46	**4.98**	97.79	**6.24**	**97.81**	**26.39**	78.85

Methods	MSU-MFSD		OULU-NPU		Replay-Attack		Rose-Youtu	
	HTER	AUC	HTER	AUC	HTER	AUC	HTER	AUC
Methods using LLM-derived semantic priors								
I-FAS	5.63	98.73	14.86	91.68	9.15	95.12	5.52	98.48
FaceCoT	8.33	98.29	9.47	96.01	9.87	96.84	2.45	99.66
TAR-FAS	5.71	98.29	14.45	92.63	5.75	96.69	3.61	99.16
Methods without LLM-derived semantic priors								
FLIP	19.37	89.95	20.57	87.30	25.67	81.37	80.73	73.60
DGPDL	**4.28**	**98.59**	10.76	95.70	7.28	97.59	5.13	**98.56**
HypLive-FAS	6.25	96.31	**9.53**	**96.15**	**4.04**	**98.10**	**4.33**	98.27

Methods	SiW		SiW-M-V2		WMCA		Avg.	
	HTER	AUC	HTER	AUC	HTER	AUC	HTER	AUC
Methods using LLM-derived semantic priors								
I-FAS	4.02	98.34	10.89	95.02	20.07	89.17	11.30	93.71
FaceCoT	0.79	99.98	9.85	96.29	11.51	95.12	8.56	96.37
TAR-FAS	8.35	96.93	11.72	94.75	9.78	95.29	7.54	96.67
Methods without LLM-derived semantic priors								
FLIP	11.01	95.40	25.95	80.78	19.36	88.73	18.73	87.90
DGPDL	**1.32**	**99.88**	**13.00**	**92.60**	13.84	93.33	9.55	95.06
HypLive-FAS	2.37	99.40	15.12	89.43	**7.23**	**96.40**	**8.01**	**95.27**

4.3 Ablation Studies

To evaluate the effectiveness of every mechanism of the proposed approach, we conduct extensive ablation studies and report HTER and AUC averaged over the four target domains of Protocol 1. More ablation results are provided in the Appendix.

4.3.1 OVERALL COMPONENT EFFECTIVENESS

Table 3 evaluates LPA and live-manifold learning under Euclidean and hyperbolic geometries. Both components consistently improve the CLIP-based baseline, and their combination performs best in both spaces. The full hyperbolic model achieves 1.19% HTER and 99.64% AUC, outperforming its Euclidean counterpart and demonstrating the complementary effects of hyperbolic geometry, LPA, and live-manifold learning.

Table 3: Component ablation under Euclidean and hyperbolic geometry. We evaluate the individual and combined effects of LPA and live manifold over four target domains.

Geometry	Baseline	LPA	Manifold	HTER(%)	AUC(%)
Euclidean	✓			4.65	98.17
	✓	✓		3.70	98.50
	✓		✓	2.51	99.13
	✓	✓	✓	2.39	99.16
Hyperbolic	✓			3.97	98.65
	✓	✓		2.82	99.09
	✓		✓	2.00	99.46
	✓	✓	✓	**1.19**	**99.64**

Table 4: Sensitivity to hyperbolic curvature c. Results are averaged over four target domains.

c	HTER(%)	AUC(%)
0.001	1.81	99.48
0.01	1.60	99.54
0.10	1.67	99.50
0.50	1.58	99.50
1.00	**1.19**	**99.64**
1.50	2.28	99.32
2.00	2.60	99.12

4.3.2 ANALYSIS OF HYPERBOLIC LIVE-MANIFOLD LEARNING

Effect of Hyperbolic Geometry. Table 4 investigates the impact of curvature c on the learned live manifold. Since curvature determines the geometry of the representation space, it directly affects manifold compactness and deviation estimation. The model achieves the best performance at $c = 1.00$, with 1.19% HTER and 99.64% AUC. Smaller curvature provides insufficient geometric capacity to characterize the asymmetric structure between compact live samples and diverse spoof deviations, while excessive curvature may introduce overly expanded representations and distort the learned manifold. To further analyze this geometric effect, we evaluate the deviation-score distributions using d' and score AUC. Specifically, $d' = (\mu_S - \mu_L)/\sqrt{(\sigma_S^2 + \sigma_L^2)/2}$ measures the standardized separation between live and spoof scores, where μ_L, μ_S and σ_L^2, σ_S^2 denote their means and variances. As shown in Table 5, hyperbolic geometry improves d' from 4.79 to 5.20 and score AUC from 99.16% to 99.64%, demonstrating that appropriate negative curvature enhances the separation between live samples close to the learned manifold and spoof samples with larger deviations.

Table 5: Comparison of deviation-score separation under Euclidean and hyperbolic geometries.

Geometry	d'	Score AUC (%)
Euclidean	4.79	99.16
Hyperbolic	**5.20**	**99.64**

Table 6: Contribution of Jacobian decomposition and primitive prior to deviation scoring. Results are averaged over four target domains.

Residual	Jacobian	Primitive Prior	HTER(%)	AUC(%)
✓			2.73	99.09
✓	✓		1.82	99.52
✓	✓	✓	**1.19**	**99.64**

Effect of Deviation-Score Components. Table 6 evaluates the components of the proposed manifold-deviation scoring. Using only the raw reconstruction residual yields 2.73% HTER and 99.09% AUC, as it also captures valid live variations explainable by the local manifold geometry. Jacobian decomposition reduces HTER to 1.82% and improves AUC to 99.52%, showing that local tangent directions help filter out such variations. Incorporating the primitive prior further lowers HTER to 1.19% and raises AUC to 99.64%, indicating that source-live tangent patterns provide additional guidance for separating plausible live variations from spoof deviations.

4.3.3 ANALYSIS OF LIVENESS-PRESERVING AUGMENTATION

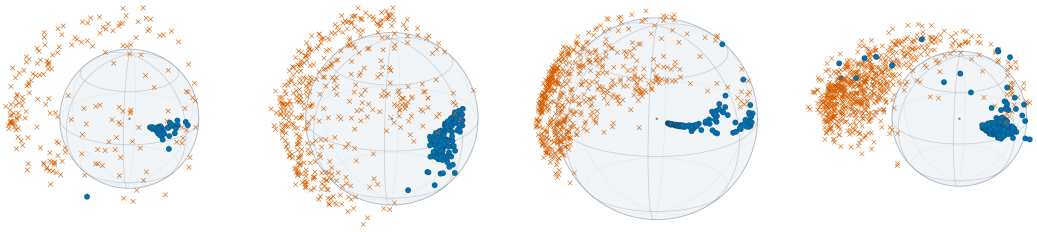
Table 7 evaluates the two LPA objectives under identical settings. Without consistency constraints, the model obtains 1.90% HTER and 99.46% AUC. Direction and score consistency reduce HTER to 1.73% and 1.68%, respectively, while both achieve 99.53% AUC. Combining them produces the best result of 1.19% HTER and 99.64% AUC, confirming that representation-direction preservation and decision-score stability provide complementary supervision.

Table 7: Ablation of direction and deviation-score consistency in LPA.

Dir.	Score	HTER↓ (%)	AUC↑ (%)
–	–	1.90	99.46
✓	–	1.73	99.53
–	✓	1.68	99.53
✓	✓	**1.19**	**99.64**

4.4 Visualization and Analysis

Figure 3 shows representations across four target domains. Compact live clusters and dispersed spoof samples support our live-centered design, which models only the live manifold and treats spoofs as off-manifold deviations, without fitting their diverse, open-ended distribution. The slightly greater live spillover on OULU-NPU may result from its larger and more heterogeneous sample set, which contains more diverse live variations. Appendix Figure 5 provides complementary azimuths to the -55° view in Figure 3.



(a) MSU-MFSD (M) (b) CASIA-MFSD (C) (c) Replay-Attack (I) (d) OULU-NPU (O)

Figure 3: Poincaré maps of learned live manifolds on four target domains. Blue dots and orange crosses denote live and spoof samples, respectively. The dashed contour and shaded region represent the learned live region and decision boundary.

Figure 4 visualizes the decision-score distribution on CASIA-MFSD under Protocol 1. The horizontal axis is the normalized tangent component $\|\mathbf{r}_i^{\tan}\|_{\boldsymbol{\mu}_L}/t_{\text{dec}}$, measuring variation explained by the local tangent geometry, whereas the vertical axis is the normalized manifold-deviation score $S(\mathbf{x}_i)/t_{\text{dec}}$, measuring unexplained normal variation that directly drives classification. Live and spoof samples partially overlap horizontally because both contain appearance, illumination, and acquisition variations, but separate clearly along the vertical axis. This shows that HypLive-FAS discounts manifold-consistent variations and detects attacks through deviations unexplained by the learned live manifold. See Appendix Figure 6 for Protocol 1 and Figure 7 for Protocol 2.

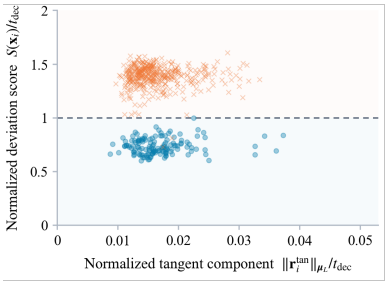


Figure 4: CASIA-MFSD under Protocol 1. Blue circles/orange crosses denote live/spoof samples; the dashed line is the decision threshold.

5 Conclusion

We proposed HypLive-FAS, a hyperbolic live-manifold framework that models source-live samples with a PCA-anchored nonlinear manifold and detects attacks through Jacobian-based regularized deviation scores. Liveness-preserving augmentation further improves robustness by maintaining feature-direction and deviation-score consistency under various appearance perturbations. Extensive experiments validate the effectiveness of our framework in capturing intrinsic liveness patterns and enhancing the robustness of FAS models. Beyond FAS, our live-centered formulation may provide a useful perspective for classification problems with asymmetric class structures, where one class is compact and stable while its counterpart is heterogeneous and open-ended.

6 AI USE STATEMENT

AI tools were used during manuscript preparation for language polishing, grammar refinement, and improving academic writing clarity. They were also used to assist with retrieval and discovery of related research works, including literature exploration and organization of background information. In addition, AI tools were used to assist with code implementation, debugging, and software development during the experimental stage. AI tools were not involved in the development of the proposed methodology, experimental design, result generation, or scientific interpretation. All research ideas, methodological designs, experimental decisions, and final conclusions were made by the authors, who take full responsibility for the manuscript.

REFERENCES

Zinelabidine Boulkenafet, Jukka Komulainen, and Abdenour Hadid. Face anti-spoofing based on color texture analysis. In *2015 IEEE international conference on image processing (ICIP)*, pp. 2636–2640. IEEE, 2015.

Zinelabidine Boulkenafet, Jukka Komulainen, and Abdenour Hadid. Face antispoofing using speeded-up robust features and fisher vector encoding. *IEEE Signal Processing Letters*, 24(2): 141–145, 2016.

Zinelabinde Boulkenafet, Jukka Komulainen, Lei Li, Xiaoyi Feng, and Abdenour Hadid. Oulunpu: A mobile face presentation attack database with real-world variations. In *2017 12th IEEE international conference on automatic face & gesture recognition (FG 2017)*, pp. 612–618. IEEE, 2017.

Ivana Chingovska, André Anjos, and Sébastien Marcel. On the effectiveness of local binary patterns in face anti-spoofing. In *2012 BIOSIG-proceedings of the international conference of biometrics special interest group (BIOSIG)*, pp. 1–7. IEEE, 2012.

Karan Desai, Maximilian Nickel, Tanmay Rajpurohit, Justin Johnson, and Shanmukha Ramakrishna Vedantam. Hyperbolic image-text representations. In *International Conference on Machine Learning*, pp. 7694–7731. PMLR, 2023.

Samuele Fonio, Roberto Esposito, Marco Aldinucci, et al. Hyperbolic prototypical entailment cones for image classification. In *Proceedings of The 28th International Conference on Artificial Intelligence and Statistics*, volume 258, pp. 3358–3366. Proceedings of Machine Learning Research, 2025.

Octavian Ganea, Gary Bécigneul, and Thomas Hofmann. Hyperbolic entailment cones for learning hierarchical embeddings. In *International conference on machine learning*, pp. 1646–1655. PMLR, 2018a.

Octavian Ganea, Gary Bécigneul, and Thomas Hofmann. Hyperbolic neural networks. *Advances in neural information processing systems*, 31, 2018b.

Anjith George, Zohreh Mostaani, David Geissenbuhler, Olegs Nikisins, André Anjos, and Sébastien Marcel. Biometric face presentation attack detection with multi-channel convolutional neural network. *IEEE transactions on information forensics and security*, 15:42–55, 2019.

Caglar Gulcehre, Misha Denil, Mateusz Malinowski, Ali Razavi, Razvan Pascanu, Karl Moritz Hermann, Peter Battaglia, Victor Bapst, David Raposo, Adam Santoro, et al. Hyperbolic attention networks. *arXiv preprint arXiv:1805.09786*, 2018.

Jiabao Guo, Huan Liu, Yizhi Luo, Xueli Hu, Hang Zou, Yuan Zhang, Hui Liu, and Bo Zhao. Style-conditional prompt token learning for generalizable face anti-spoofing. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pp. 994–1003, 2024.

Xiao Guo, Yaojie Liu, Anil Jain, and Xiaoming Liu. Multi-domain learning for updating face anti-spoofing models. In *European conference on computer vision*, pp. 230–249. Springer, 2022a.

Yunhui Guo, Xudong Wang, Yubei Chen, and Stella X Yu. Clipped hyperbolic classifiers are super-hyperbolic classifiers. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–10. IEEE, 2022b.

Shuangpeng Han, Rizhao Cai, Yawen Cui, Zitong Yu, Yongjian Hu, and Alex Kot. Hyperbolic face anti-spoofing. *arXiv preprint arXiv:2308.09107*, 2023.

Chengyang Hu, Ke-Yue Zhang, Taiping Yao, Shouhong Ding, and Lizhuang Ma. Rethinking generalizable face anti-spoofing via hierarchical prototype-guided distribution refinement in hyperbolic space. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1032–1041. IEEE, 2024.

Pei-Kai Huang, Cheng-Hsuan Chiang, Tzu-Hsien Chen, Jun-Xiong Chong, Tyng-Luh Liu, and Chiou-Ting Hsu. One-class face anti-spoofing via spoof cue map-guided feature learning. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 277–286. IEEE, 2024.

Yunpei Jia, Jie Zhang, Shiguang Shan, and Xilin Chen. Single-side domain generalization for face anti-spoofing. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8481–8490. IEEE, 2020.

Binh M. Le and Simon S. Woo. Gradient alignment for cross-domain face anti-spoofing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 188–199, 2024.

Haoliang Li, Wen Li, Hong Cao, Shiqi Wang, Feiyue Huang, and Alex C Kot. Unsupervised domain adaptation for face anti-spoofing. *IEEE Transactions on Information Forensics and Security*, 13 (7):1794–1809, 2018.

Ajian Liu, Chenxu Zhao, Zitong Yu, Jun Wan, Anyang Su, Xing Liu, Zichang Tan, Sergio Escalera, Junliang Xing, Yanyan Liang, et al. Contrastive context-aware learning for 3d high-fidelity mask face presentation attack detection. *IEEE transactions on information forensics and security*, 17: 2497–2507, 2022.

Ajian Liu, Shuai Xue, Jianwen Gan, Jun Wan, Yanyan Liang, Jiankang Deng, Sergio Escalera, and Zhen Lei. Cfpl-fas: Class free prompt learning for generalizable face anti-spoofing. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 222–232. IEEE, 2024.

Ajian Liu, Xun Lin, Hui Ma, Xinxing Yu, Jiabao Guo, Zitong Yu, Jun Wan, Zhanchuan Cai, Zhen Lei, and Yanyan Liang. Icpe-fas: Instance and category prompts engineering for generalizable face anti-spoofing: A. liu et al. *International Journal of Computer Vision*, 134(6):292, 2026a.

Ajian Liu, Xun Lin, Ruicong Zhi, Yanyan Liang, Xinshan Zhu, Zhanchuan Cai, Jun Wan, Sergio Escalera, and Zhen Lei. Dgpdl: domain-guided prompt distribution learning for generalizable face anti-spoofing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2026b.

Shubao Liu, Ke-Yue Zhang, Taiping Yao, Mingwei Bi, Shouhong Ding, Jilin Li, Feiyue Huang, and Lizhuang Ma. Adaptive normalized representation learning for generalizable face anti-spoofing. In *Proceedings of the 29th ACM international conference on multimedia*, pp. 1469–1477, 2021.

Si-Qi Liu, Xiangyuan Lan, and Pong C Yuen. Remote photoplethysmography correspondence feature for 3d mask face presentation attack detection. In *European Conference on Computer Vision*, pp. 577–594. Springer, 2018a.

Yaojie Liu, Amin Jourabloo, and Xiaoming Liu. Learning deep models for face anti-spoofing: Binary or auxiliary supervision. In *2018 IEEE/CVF Conference on computer vision and pattern recognition*, pp. 389–398. IEEE, 2018b.

Yuchen Liu, Yabo Chen, Mengran Gou, Chun-Ting Huang, Yaoming Wang, Wenrui Dai, and Hongkai Xiong. Towards unsupervised domain generalization for face anti-spoofing. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 20597–20607. IEEE, 2023.

Shiyun Mao and Huibin Li. Hyperbolic metric learning for generalizable face anti-spoofing. *IEEE Transactions on Information Forensics and Security*, 20:7257–7271, 2025.

Gal Mishne, Zhengchao Wan, Yusu Wang, and Sheng Yang. The numerical stability of hyperbolic representation learning. In *International Conference on Machine Learning*, pp. 24925–24949. PMLR, 2023.

Maximillian Nickel and Douwe Kiela. Poincaré embeddings for learning hierarchical representations. *Advances in neural information processing systems*, 30, 2017.

Maximillian Nickel and Douwe Kiela. Learning continuous hierarchies in the lorentz model of hyperbolic geometry. In *International conference on machine learning*, pp. 3779–3788. PMLR, 2018.

Avik Pal, Max Van Spengler, Guido D'Amely di Melendugno, Alessandro Flaborea, Fabio Galasso, and Pascal Mettes. Compositional entailment learning for hyperbolic vision-language models. In *International Conference on Learning Representations*, volume 2025, pp. 87371–87399, 2025.

Keyurkumar Patel, Hu Han, and Anil K Jain. Secure face unlock: Spoof detection on smartphones. *IEEE transactions on information forensics and security*, 11(10):2268–2283, 2016.

Karl Pearson. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin philosophical magazine and journal of science*, 2(11):559–572, 1901.

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*, pp. 8748–8763. PmLR, 2021.

Vaibhav Rathore, Divyam Gupta, and Biplab Banerjee. Hidisc: A hyperbolic framework for domain generalization with generalized category discovery. *Advances in Neural Information Processing Systems*, 38:99399–99419, 2026.

Rui Shao, Xiangyuan Lan, Jiawei Li, and Pong C Yuen. Multi-adversarial discriminative deep domain generalization for face presentation attack detection. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10015–10023. IEEE, 2019.

Rui Shao, Xiangyuan Lan, and Pong C Yuen. Regularized fine-grained meta face anti-spoofing. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 11974–11981, 2020.

Koushik Srivatsan, Muzammal Naseer, and Karthik Nandakumar. Flip: Cross-domain face anti-spoofing with language guidance. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 19628–19639. IEEE, 2023.

Yiyou Sun, Yaojie Liu, Xiaoming Liu, Yixuan Li, and Wen-Sheng Chu. Rethinking domain generalization for face anti-spoofing: Separability and alignment. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 24563–24574, 2023.

Chien-Yi Wang, Yu-Ding Lu, Shang-Ta Yang, and Shang-Hong Lai. Patchnet: A simple face anti-spoofing framework via fine-grained patch recognition. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20249–20258. IEEE, 2022a.

Jingjing Wang, Jingyi Zhang, Ying Bian, Youyi Cai, Chunmao Wang, and Shiliang Pu. Self-domain adaptation for face anti-spoofing. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pp. 2746–2754, 2021.

Keyao Wang, Guosheng Zhang, Haixiao Yue, Yanyan Liang, Mouxiao Huang, Gang Zhang, Junyu Han, Errui Ding, and Jingdong Wang. Csdg-fas: Closed-space domain generalization for face anti-spoofing. *International Journal of Computer Vision*, 132(11):4866–4879, 2024.

Zhuo Wang, Zezheng Wang, Zitong Yu, Weihong Deng, Jiahong Li, Tingting Gao, and Zhongyuan Wang. Domain generalization via shuffled style assembly for face anti-spoofing. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4113–4123. IEEE, 2022b.

Di Wen, Hu Han, and Anil K Jain. Face spoof detection with image distortion analysis. *IEEE Transactions on Information Forensics and Security*, 10(4):746–761, 2015.

Jianwei Yang, Zhen Lei, Shengcai Liao, and Stan Z Li. Face liveness detection with component dependent descriptor. In *2013 international conference on biometrics (ICB)*, pp. 1–6. IEEE, 2013.

Jingyi Yang, Xun Lin, Zitong Yu, Liepiao Zhang, Xin Liu, Hui Li, Xiaochen Yuan, and Xiaochun Cao. Dadm: Dual alignment of domain and modality for face anti-spoofing. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 12045–12056. IEEE, 2025.

Chang Yu, Yuxuan Luo, Yixuan Du, Yuqing Zhou, Siyuan Li, Jingbo Zhou, Zhen Lei, Stan Z Li, et al. Geometric flow grounding: A unified manifold decoupling framework for dynamics discovery and verification. In *Forty-third International Conference on Machine Learning*.

Jeongmin Yu, Susang Kim, Kisu Lee, Taekyoung Kwon, Won-Yong Shin, and Ha Young Kim. Multi-view slot attention using paraphrased texts for face anti-spoofing. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 21117–21128. IEEE, 2025.

Zitong Yu, Jun Wan, Yunxiao Qin, Xiaobai Li, Stan Z Li, and Guoying Zhao. Nas-fas: Static-dynamic central difference network search for face anti-spoofing. *IEEE transactions on pattern analysis and machine intelligence*, 43(9):3005–3023, 2020a.

Zitong Yu, Chenxu Zhao, Zezheng Wang, Yunxiao Qin, Zhuo Su, Xiaobai Li, Feng Zhou, and Guoying Zhao. Searching central difference convolutional networks for face anti-spoofing. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5294–5304. IEEE, 2020b.

Zitong Yu, Yunxiao Qin, Xiaobai Li, Chenxu Zhao, Zhen Lei, and Guoying Zhao. Deep learning for face anti-spoofing: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 45 (5):5609–5631, 2022.

Guosheng Zhang, Keyao Wang, Haixiao Yue, Ajian Liu, Gang Zhang, Kun Yao, Errui Ding, and Jingdong Wang. Interpretable face anti-spoofing: Enhancing generalization with multimodal large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 9896–9904, 2025.

Haoyuan Zhang, Keyao Wang, Guosheng Zhang, Haixiao Yue, Zhiwen Tan, Siran Peng, Tianshuo Zhang, Xiao Tan, Kunbin Chen, Wei He, et al. From intuition to investigation: A tool-augmented reasoning mllm framework for generalizable face anti-spoofing. *arXiv preprint arXiv:2603.01038*, 2026a.

Honglu Zhang, Zhiqin Fang, Ningning Zhao, Saihui Hou, Long Ma, Renwang Pei, and Zhaofeng He. Harnessing chain-of-thought reasoning in multimodal large language models for face anti-spoofing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 33566–33576, 2026b.

Yuanhan Zhang, ZhenFei Yin, Yidong Li, Guojun Yin, Junjie Yan, Jing Shao, and Ziwei Liu. Celeba-spoof: Large-scale face anti-spoofing dataset with rich annotations. In *European conference on computer vision*, pp. 70–85. Springer, 2020.

Zhiwei Zhang, Junjie Yan, Sifei Liu, Zhen Lei, Dong Yi, and Stan Z Li. A face antispoofing database with diverse attacks. In *2012 5th IAPR international conference on Biometrics (ICB)*, pp. 26–31. IEEE, 2012.

Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *International journal of computer vision*, 130(9):2337–2348, 2022a.

Qianyu Zhou, Ke-Yue Zhang, Taiping Yao, Ran Yi, Shouhong Ding, and Lizhuang Ma. Adaptive mixture of experts learning for generalizable face anti-spoofing. In *Proceedings of the 30th ACM international conference on multimedia*, pp. 6009–6018, 2022b.

Qianyu Zhou, Ke-Yue Zhang, Taiping Yao, Xuequan Lu, Ran Yi, Shouhong Ding, and Lizhuang Ma. Instance-aware domain generalization for face anti-spoofing. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20453–20463. IEEE, 2023.

Qianyu Zhou, Ke-Yue Zhang, Taiping Yao, Xuequan Lu, Shouhong Ding, and Lizhuang Ma. Test-time domain generalization for face anti-spoofing. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 175–187. IEEE, 2024.

A Hyperbolic Geometry Preliminary

Unlike Euclidean space, hyperbolic space is a non-Euclidean Riemannian manifold with constant negative curvature. In this section, we briefly introduce the mathematical foundation of hyperbolic space and the Poincaré ball model used in our framework.

A.1 Euclidean Space and Hyperbolic Space

In Euclidean space $\mathbb{R}^D$, each point is represented by a D-dimensional vector, and the geometry is determined by the Euclidean inner product $\langle \mathbf{x}, \mathbf{y} \rangle_E = \mathbf{x}^T \mathbf{y}$, where the corresponding Euclidean norm is $\|\mathbf{x}\|_E = \sqrt{\langle \mathbf{x}, \mathbf{x} \rangle_E}$. The distance between two points is measured by the straight-line distance $d_E(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|_E$.

Different from Euclidean space, hyperbolic space $\mathcal{H}^D$ is a Riemannian manifold with negative curvature. The points in hyperbolic space are not directly optimized by Euclidean operations, since the manifold has its own local geometry (Nickel & Kiela, 2017; 2018). Specifically, each point $\mathbf{x} \in \mathcal{H}^D$ is associated with a tangent space $T_{\mathbf{x}}\mathcal{H}^D$, which is a local Euclidean vector space. Therefore, Euclidean operations can be performed in the tangent space and then mapped back to the hyperbolic manifold through geometric mappings. Formally, the tangent space at point $\mathbf{x}$ is defined as

$$T_{\mathbf{x}}\mathcal{H}^D = \{\mathbf{v} : \mathbf{v} \text{ is a tangent vector at } \mathbf{x}\}, \tag{12}$$

where $T_{\mathbf{x}}\mathcal{H}^D$ has the same dimensionality as $\mathbb{R}^D$.

The relationship between Euclidean space and hyperbolic space is established through the exponential map and logarithmic map. The exponential map projects a tangent vector from the Euclidean tangent space onto the hyperbolic manifold:

$$\mathrm{Exp}_{\mathbf{x}} : T_{\mathbf{x}}\mathcal{H}^D \to \mathcal{H}^D, \tag{13}$$

while the logarithmic map performs the inverse transformation:

$$\mathrm{Log}_{\mathbf{x}} : \mathcal{H}^D \to T_{\mathbf{x}}\mathcal{H}^D. \tag{14}$$

These mappings enable deep neural networks to extract features in Euclidean space and optimize them in hyperbolic space.

A.2 Poincaré Ball Model

In this work, we adopt the D-dimensional Poincaré ball model $\mathbb{D}_c^D$ with curvature $-c$ (Nickel & Kiela, 2017), which is defined as

$$\mathbb{D}_c^D = \{\mathbf{x} \in \mathbb{R}^D | c\|\mathbf{x}\|_2^2 < 1\}, \tag{15}$$

where $c > 0$ controls the absolute value of the negative curvature.

The Poincaré ball is conformal to the Euclidean space, meaning that the angles between vectors are preserved, while the distance is determined by the Riemannian metric. The metric tensor at point $\mathbf{x}$ is defined as

$$g_{\mathbf{x}}^c = \lambda_{\mathbf{x}}^2 g^E, \tag{16}$$

where

$$\lambda_{\mathbf{x}} = \frac{2}{1 - c\|\mathbf{x}\|_2^2}, \tag{17}$$

and g^E denotes the Euclidean metric.

Accordingly, the Riemannian norm of a tangent vector $\mathbf{v} \in T_{\mathbf{x}}\mathbb{D}_c^D$ is defined as

$$\|\mathbf{v}\|_{\mathbf{x},c} = \sqrt{g_{\mathbf{x}}^c(\mathbf{v}, \mathbf{v})} = \lambda_{\mathbf{x}}\|\mathbf{v}\|_2. \tag{18}$$

A.3 Hyperbolic Distance and Operations

Based on the Riemannian metric, the geodesic distance between two points $\mathbf{x}, \mathbf{y} \in \mathbb{D}_c^D$ is computed as

$$d_c(\mathbf{x}, \mathbf{y}) = \frac{1}{\sqrt{c}} \operatorname{arcosh}\left(1 + \frac{2c\|\mathbf{x} - \mathbf{y}\|_2^2}{(1 - c\|\mathbf{x}\|_2^2)(1 - c\|\mathbf{y}\|_2^2)}\right). \tag{19}$$

Since the Poincaré ball is a manifold, vector addition is replaced by Möbius addition. For two points $\mathbf{x}, \mathbf{y} \in \mathbb{D}_c^D$, it is defined as

$$\mathbf{x} \oplus_c \mathbf{y} = \frac{(1 + 2c\langle \mathbf{x}, \mathbf{y}\rangle + c\|\mathbf{y}\|^2)\mathbf{x} + (1 - c\|\mathbf{x}\|^2)\mathbf{y}}{1 + 2c\langle \mathbf{x}, \mathbf{y}\rangle + c^2\|\mathbf{x}\|^2\|\mathbf{y}\|^2}. \tag{20}$$

The exponential map on the Poincaré ball is given by

$$\operatorname{Exp}_{\mathbf{x}}^c(\mathbf{v}) = \mathbf{x} \oplus_c \left(\tanh\left(\frac{\sqrt{c}\lambda_{\mathbf{x}}\|\mathbf{v}\|_2}{2}\right)\frac{\mathbf{v}}{\sqrt{c}\|\mathbf{v}\|_2}\right), \tag{21}$$

and the logarithmic map is

$$\operatorname{Log}_{\mathbf{x}}^c(\mathbf{y}) = \frac{2}{\sqrt{c}\lambda_{\mathbf{x}}} \operatorname{arctanh}(\sqrt{c}\| - \mathbf{x} \oplus_c \mathbf{y}\|_2)\frac{-\mathbf{x} \oplus_c \mathbf{y}}{\| - \mathbf{x} \oplus_c \mathbf{y}\|_2}. \tag{22}$$

In HypLive-FAS, the fused visual features are mapped from the tangent space at the origin to the Poincaré ball through the exponential map

$$\operatorname{Exp}_{\mathbf{0}}^c(\mathbf{v}) = \tanh\big(\sqrt{c}\|\mathbf{v}\|_2\big)\,\frac{\mathbf{v}}{\sqrt{c}\|\mathbf{v}\|_2}. \tag{23}$$

The resulting hyperbolic embeddings are subsequently mapped to the tangent space at the source-live Fréchet center via the logarithmic map, providing the geometric basis for live-manifold learning and deviation-based spoof detection.

B Detailed Implementation of HypLive-FAS

This section provides additional implementation details of HypLive-FAS, including initialization strategies, optimization settings, and training procedures.

B.1 Input Preprocessing and CLIP Implementation Details

Input Preprocessing All input images are resized to 224×224 and normalized using the ImageNet mean and standard deviation, i.e., $(0.485, 0.456, 0.406)$ and $(0.229, 0.224, 0.225)$, respectively. Basic training augmentations include random horizontal flipping and random saturation adjustment. Random horizontal flipping is applied with a probability of 0.5, while the saturation scaling factor is uniformly sampled from $[0.8, 1.2]$ and applied with a probability of 0.5.

We adopt CLIP ViT-B/16 as the visual backbone and jointly optimize the image encoder with the downstream modules, while keeping the text encoder frozen. In the text branch, only the CoOp context tokens are learnable. All remaining components, including the token embeddings, Transformer blocks, LayerNorm layers, and text projection, remain fixed throughout training.

Prompt Optimization We use three fixed live phrases: "real face", "bonafide face", and "genuine face". The three prompts share eight learnable context tokens. Let the learnable context-token sequence be $\mathbf{C} = [\mathbf{c}_1^{\mathrm{ctx}}, \ldots, \mathbf{c}_8^{\mathrm{ctx}}]$. The k-th prompt is formulated as $\mathbf{T}_k = [\mathbf{C}; \mathrm{tokenize}(q_k)]$, where q_k denotes the k-th fixed live phrase, i.e., $q_1 =$ "real face", $q_2 =$ "bonafide face", and $q_3 =$ "genuine face". The prompt features are extracted by the frozen CLIP text encoder:

$$\mathbf{t}_k = \mathrm{Norm}\left(E_{\mathrm{text}}(\mathbf{T}_k)\right) \in \mathbb{R}^{512}, \qquad k \in \{1, 2, 3\}, \tag{24}$$

where $\mathrm{Norm}(\mathbf{z}) = \mathbf{z}/\|\mathbf{z}\|_2$.

B.2 PROMPT-GUIDED LOCAL FEATURE AGGREGATION

Given a mini-batch of B input images $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^{B} \in \mathbb{R}^{B\times 3\times 224\times 224}$, the CLIP ViT-B/16 image encoder produces one raw CLS token and L raw patch tokens for each image. The batched raw CLS features are represented as $\mathbf{C}^{\text{raw}} = [\mathbf{c}_1^{\text{raw}}, \ldots, \mathbf{c}_B^{\text{raw}}]^\top \in \mathbb{R}^{B\times 768}$, where $\mathbf{c}_i^{\text{raw}} \in \mathbb{R}^{768}$ is the raw CLS token of the i-th image. Similarly, the batched raw patch features are represented as $\mathbf{P}^{\text{raw}} = \left\{\mathbf{p}_{i,l}^{\text{raw}}\right\}_{i=1,l=1}^{B,L} \in \mathbb{R}^{B\times L\times 768}$, where $\mathbf{p}_{i,l}^{\text{raw}} \in \mathbb{R}^{768}$ denotes the l-th raw patch token of the i-th image. For an input resolution of 224×224 and a patch size of 16×16, the encoder generates a 14×14 patch grid. Therefore,

$$L = \frac{224}{16} \times \frac{224}{16} = 14 \times 14 = 196, \tag{25}$$

and hence $\mathbf{P}^{\text{raw}} \in \mathbb{R}^{B\times 196\times 768}$.

The CLS and patch tokens are independently processed by two separately parameterized two-layer MLP projection heads, denoted by ϕ_{cls} and ϕ_{patch}, respectively. For $h \in \{\text{cls}, \text{patch}\}$, the corresponding projection head is defined as

$$\phi_h(\mathbf{z}) = \mathbf{W}_{h,2}\,\text{GELU}\left(\mathbf{W}_{h,1}\mathbf{z} + \mathbf{b}_{h,1}\right) + \mathbf{b}_{h,2}, \tag{26}$$

where $\mathbf{W}_{h,1} \in \mathbb{R}^{512\times 768}, \mathbf{b}_{h,1} \in \mathbb{R}^{512}$, and$\mathbf{W}_{h,2} \in \mathbb{R}^{512\times 512}, \mathbf{b}_{h,2} \in \mathbb{R}^{512}$. Therefore, each projection head performs the token-wise dimensional mapping $\mathbb{R}^{768} \longrightarrow \mathbb{R}^{512} \longrightarrow \mathbb{R}^{512}$.

The projected CLS features are subsequently ℓ_2-normalized along the feature dimension:

$$\mathbf{c}_i = \text{Norm}\left(\phi_{\text{cls}}\left(\mathbf{c}_i^{\text{raw}}\right)\right) \in \mathbb{R}^{512}, \qquad i = 1, \ldots, B. \tag{27}$$

Accordingly, the complete batched CLS feature transformation is $\mathbf{C}^{\text{raw}} \in \mathbb{R}^{B\times 768} \longrightarrow \mathbf{H}_{\text{cls}} \in \mathbb{R}^{B\times 512} \longrightarrow \mathbf{C} \in \mathbb{R}^{B\times 512}$, where $\mathbf{H}_{\text{cls}}$ denotes the CLS features before ℓ_2 normalization.

Likewise, each projected patch token is computed as

$$\mathbf{p}_{i,l} = \text{Norm}\left(\phi_{\text{patch}}\left(\mathbf{p}_{i,l}^{\text{raw}}\right)\right) \in \mathbb{R}^{512}, \quad i = 1, \ldots, B, \quad l = 1, \ldots, L. \tag{28}$$

The complete batched patch-token transformation is therefore $\mathbf{P}^{\text{raw}} \in \mathbb{R}^{B\times 196\times 768} \longrightarrow \mathbf{H}_{\text{patch}} \in \mathbb{R}^{B\times 196\times 512} \longrightarrow \mathbf{P} \in \mathbb{R}^{B\times 196\times 512}$, where $\mathbf{H}_{\text{patch}}$ denotes the projected patch features before ℓ_2 normalization. The final ℓ_2 normalization does not change the tensor dimensions.

For patch-level semantic refinement, we first transform the projected patch features using a shared linear projection $\mathbf{W}_s \in \mathbb{R}^{512\times 512}$, which is initialized as an identity matrix. The shared projection is followed by a patch-specific residual adapter. Specifically, for the l-th patch of the i-th image, we compute $\mathbf{u}_{i,l} = \mathbf{W}_s\mathbf{p}_{i,l} \in \mathbb{R}^{512}$, and $\mathbf{r}_{i,l} = \mathbf{A}_2\,\text{GELU}\left(\mathbf{A}_1\,\text{LN}(\mathbf{u}_{i,l})\right) \in \mathbb{R}^{512}$, where $\text{LN}(\cdot)$ denotes LayerNorm and $\mathbf{A}_1 \in \mathbb{R}^{64\times 512}, \mathbf{A}_2 \in \mathbb{R}^{512\times 64}$. Thus, the residual adapter performs the bottleneck mapping $\mathbb{R}^{512} \longrightarrow \mathbb{R}^{64} \longrightarrow \mathbb{R}^{512}$. The final adapter matrix $\mathbf{A}_2$ is initialized to zero, making the residual branch initially inactive. The refined patch direction is then obtained as

$$\widetilde{\mathbf{p}}_{i,l} = \text{Norm}\left(\mathbf{u}_{i,l} + \mathbf{r}_{i,l}\right) \in \mathbb{R}^{512}. \tag{29}$$

For a mini-batch of B images with $L = 196$ patches per image, the corresponding batched transformation is $\mathbf{P} \in \mathbb{R}^{B\times 196\times 512} \longrightarrow \mathbf{U} \in \mathbb{R}^{B\times 196\times 512} \longrightarrow \widetilde{\mathbf{P}} \in \mathbb{R}^{B\times 196\times 512}$, where $\widetilde{\mathbf{P}} = \{\widetilde{\mathbf{p}}_{i,l}\}_{i=1,l=1}^{B,L}$. The projection, residual connection, and ℓ_2 normalization preserve the number of patches and the final feature dimension.

Let $K = 3$ denote the number of live prompts. The three normalized CLIP text features are collected as $\mathbf{T} = [\mathbf{t}_1, \mathbf{t}_2, \mathbf{t}_3]^\top \in \mathbb{R}^{3\times 512}$. They are replicated across the batch dimension to initialize the semantic slots:

$$\mathbf{s}_{i,k}^{(0)} = \mathbf{t}_k \in \mathbb{R}^{512}, \qquad i = 1, \ldots, B, \quad k = 1, \ldots, K. \tag{30}$$

Accordingly, the complete initial slot tensor satisfies

$$\mathbf{S}^{(0)} = \left\{\mathbf{s}_{i,k}^{(0)}\right\}_{i=1,k=1}^{B,K} \in \mathbb{R}^{B\times 3\times 512}. \tag{31}$$

At the q-th iteration, each refined patch is softly assigned to the $K = 3$ semantic slots according to their cosine similarity:

$$a_{i,l,k}^{(q)} = \frac{\exp\left(\widetilde{\mathbf{p}}_{i,l}^{\top}\mathbf{s}_{i,k}^{(q)}/\tau_{\text{slot}}\right)}{\sum_{k'=1}^{K}\exp\left(\widetilde{\mathbf{p}}_{i,l}^{\top}\mathbf{s}_{i,k'}^{(q)}/\tau_{\text{slot}}\right)}, \tag{32}$$

where $\tau_{\text{slot}} = 0.07, i = 1, \ldots, B, l = 1, \ldots, L, k = 1, \ldots, K$. Because both $\widetilde{\mathbf{p}}_{i,l}$ and $\mathbf{s}_{i,k}^{(q)}$ are unit-normalized, their inner product is equivalent to cosine similarity. The complete assignment tensor is therefore

$$\mathbf{A}^{(q)} = \left\{a_{i,l,k}^{(q)}\right\}_{i=1,l=1,k=1}^{B,L,K} \in \mathbb{R}^{B\times 196\times 3}. \tag{33}$$

The assignment weights are normalized over the $L = 196$ patches within each semantic slot:

$$w_{i,l,k}^{(q)} = \frac{a_{i,l,k}^{(q)}}{\max\left(\sum_{l'=1}^{L} a_{i,l',k}^{(q)}, \varepsilon\right)}, \tag{34}$$

where $\varepsilon = 10^{-6}$ is used for numerical stability. The normalized weight tensor satisfies

$$\mathbf{W}^{(q)} = \left\{w_{i,l,k}^{(q)}\right\}_{i=1,l=1,k=1}^{B,L,K} \in \mathbb{R}^{B\times 196\times 3}. \tag{35}$$

The normalized weights are then used to aggregate the refined patch features associated with each slot:

$$\widehat{\mathbf{s}}_{i,k}^{(q)} = \text{Norm}\left(\sum_{l=1}^{L} w_{i,l,k}^{(q)}\widetilde{\mathbf{p}}_{i,l}\right) \in \mathbb{R}^{512}. \tag{36}$$

The complete aggregated slot tensor is

$$\widehat{\mathbf{S}}^{(q)} = \left\{\widehat{\mathbf{s}}_{i,k}^{(q)}\right\}_{i=1,k=1}^{B,K} \in \mathbb{R}^{B\times 3\times 512}. \tag{37}$$

To preserve the hyperspherical geometry of the normalized CLIP feature space, each semantic slot is updated using spherical linear interpolation:

$$\mathbf{s}_{i,k}^{(q+1)} = \text{Slerp}\left(\mathbf{s}_{i,k}^{(q)}, \widehat{\mathbf{s}}_{i,k}^{(q)}; \eta_{\text{slot}}\right) \in \mathbb{R}^{512}, \tag{38}$$

where $\eta_{\text{slot}} = 0.5$.

Consequently, the slot tensor preserves its dimensions during every iteration: $\mathbf{S}^{(q)} \in \mathbb{R}^{B\times 3\times 512} \longrightarrow \widehat{\mathbf{S}}^{(q)} \in \mathbb{R}^{B\times 3\times 512} \longrightarrow \mathbf{S}^{(q+1)} \in \mathbb{R}^{B\times 3\times 512}$. The assignment, aggregation, and spherical update steps are repeated for $T = 3$ iterations, where $q = 0, 1, 2$. The final prompt-guided local liveness representations are defined as $\mathbf{z}_{i,k} = \mathbf{s}_{i,k}^{(T)} \in \mathbb{R}^{512}, i = 1, \ldots, B, k = 1, \ldots, 3$. Collectively, the final local representation tensor is

$$\mathbf{Z} = \{\mathbf{z}_{i,k}\}_{i=1,k=1}^{B,3} \in \mathbb{R}^{B\times 3\times 512}. \tag{39}$$

B.3 Local-Global Feature Fusion

The local-global representation module consists of a local projection layer, a nonlinear fusion MLP, a gated residual fusion, and a subsequent visual mapping. The three local representations $\{\mathbf{z}_{i,k}\}_{k=1}^{3}$ are first projected into a lower-dimensional space:

$$\mathbf{q}_{i,k} = \text{GELU}\left(\mathbf{W}_{\text{slot}}\,\text{LN}(\mathbf{z}_{i,k}) + \mathbf{b}_{\text{slot}}\right), \tag{40}$$

where $\mathbf{W}_{\text{slot}} \in \mathbb{R}^{128\times 512}$ and $\mathbf{q}_{i,k} \in \mathbb{R}^{128}$.

The local projection constitutes the first stage of F_{fuse}. Its subsequent two-layer MLP operates on the concatenated representation $[\mathbf{c}_i; \mathbf{q}_{i,1}; \mathbf{q}_{i,2}; \mathbf{q}_{i,3}]$ and produces a 512-dimensional fusion output. Following the notation in the main text, the gated residual fusion is written as

$$\mathbf{f}_i^{\text{fuse}} = \mathbf{c}_i + \gamma F_{\text{fuse}}\left([\mathbf{c}_i; \mathbf{z}_{i,1}; \mathbf{z}_{i,2}; \mathbf{z}_{i,3}]\right), \qquad \gamma = 0.1\,\sigma(g), \tag{41}$$

where g is a learnable scalar initialized to zero, yielding an initial fusion coefficient of $\gamma = 0.05$.

The visual mapping Ψ_{r} in the main text is implemented using a linear projection followed by a residual adapter. Specifically,

$$\mathbf{f}_i^s = \mathbf{W}_{\text{r}}\mathbf{f}_i^{\text{fuse}}, \qquad \mathbf{W}_{\text{r}} \in \mathbb{R}^{512\times 512}, \tag{42}$$

and the final visual feature is

$$\mathbf{f}_i = \Psi_{\text{r}}\left(\mathbf{f}_i^{\text{fuse}}\right) = \mathbf{f}_i^s + A_{\text{r}}\left(\text{LN}(\mathbf{f}_i^s)\right). \tag{43}$$

The residual adapter A_{r} is implemented as a two-layer bottleneck MLP with a 64-dimensional latent space. Its final projection layer is initialized to zero, so that the residual branch starts from an inactive state and gradually learns feature refinements.

B.4 Hyperbolic Embedding and Radial Calibration

Following Section 3.2, we use a fixed curvature magnitude $c = 1$ and radial scaling factor $\rho_0 = 0.75$. The radial scale is calibrated using the median norm of the source-live visual features:

$$r_{\text{ref}} = \text{median}_{j\in\mathcal{S}_L} \|\mathbf{f}_j\|_2. \tag{44}$$

The reference norm is detached from gradient computation, and each visual feature is mapped from the tangent space at the origin to the Poincaré ball as

$$\mathbf{h}_i = \text{Exp}_{\mathbf{0}}^c\left(\rho_0 \frac{\mathbf{f}_i}{\text{sg}(r_{\text{ref}})}\right) \in \mathbb{D}_c^D. \tag{45}$$

For numerical stability, the mapped embeddings are constrained to remain inside the valid ball region:

$$\|\mathbf{h}_i\|_2 < \frac{1-\epsilon}{\sqrt{c}}, \qquad \epsilon = 10^{-5}. \tag{46}$$

B.5 Initialization and Update of Live Geometric Structure

Live-centered Tangent Geometry The live geometric reference statistics are computed only from source-domain live embeddings. Given the source-live hyperbolic embeddings $\{\mathbf{h}_i\}_{i\in\mathcal{S}_L}$, we compute the Fréchet center:

$$\boldsymbol{\mu}_L = \arg\min_{\boldsymbol{\mu}\in\mathbb{D}_c^D} \sum_{i\in\mathcal{S}_L} d_c^2(\boldsymbol{\mu}, \mathbf{h}_i), \tag{47}$$

where $d_c(\cdot,\cdot)$ denotes the hyperbolic distance.

The initial estimate of the Fréchet center is obtained by projecting the Euclidean mean of source-live hyperbolic embeddings back into the valid Poincaré ball region:

$$\boldsymbol{\mu}_L^{(0)} = \Pi_{\mathbb{D}_c^D}\left(\frac{1}{|\mathcal{S}_L|}\sum_{i\in\mathcal{S}_L}\mathbf{h}_i\right), \tag{48}$$

where $\Pi_{\mathbb{D}_c^D}(\cdot)$ denotes the projection operation that ensures numerical validity inside the Poincaré ball:

$$\Pi_{\mathbb{D}_c^D}(\mathbf{x}) = \mathbf{x}\cdot\min\left(1, \frac{1-10^{-5}}{\sqrt{c}\|\mathbf{x}\|_2}\right). \tag{49}$$

Starting from $\boldsymbol{\mu}_L^{(0)}$, we estimate the Fréchet center using at most eight iterative Riemannian mean updates:

$$\boldsymbol{\mu}_L^{(t+1)} = \text{Exp}_{\boldsymbol{\mu}_L^{(t)}}^c\left(\frac{1}{|\mathcal{S}_L|}\sum_{i\in\mathcal{S}_L}\text{Log}_{\boldsymbol{\mu}_L^{(t)}}^c(\mathbf{h}_i)\right). \tag{50}$$

After obtaining $\boldsymbol{\mu}_L$, each embedding is mapped to its tangent space as $\mathbf{u}_i = \text{Log}_{\boldsymbol{\mu}_L}^c(\mathbf{h}_i)$. We then compute the source-live tangent mean $\bar{\mathbf{u}}_L = |\mathcal{S}_L|^{-1}\sum_{j\in\mathcal{S}_L}\mathbf{u}_j$ and obtain the centered tangent feature as $\mathbf{y}_i = \mathbf{u}_i - \bar{\mathbf{u}}_L$.

PCA-based Geometric Initialization We perform PCA on the centered tangent features $\{\mathbf{y}_i\}_{i\in\mathcal{S}_L}$ to obtain an initial approximation of the live variation subspace. Specifically, the centered tangent features are stacked into a matrix and decomposed using singular value decomposition:

$$\widetilde{\mathbf{U}} = \mathbf{A}\mathbf{\Sigma}\mathbf{V}^\top. \tag{51}$$

The first d right singular vectors are retained as the initial tangent basis:

$$\mathbf{U}_0 = \mathbf{V}_{[:,1:d]} \in \mathbb{R}^{D\times d}, \tag{52}$$

where $d = 32$. In implementation, the retained dimension is determined as $d = \min(32, N_L - 1, D)$, where N_L denotes the number of source-domain live samples used for geometric estimation.

The obtained PCA basis is further used to initialize the nonlinear live manifold network. Specifically, the encoder and decoder projections are initialized as $W_E^{(0)} = \mathbf{U}_0^\top$ and $W_0^{(0)} = \mathbf{U}_0$, respectively. The nonlinear residual branch is initialized as $W_2^{(0)} = 0$. Therefore, the learned manifold starts from the PCA approximation and gradually adapts to nonlinear live variations during training.

Geometric Refresh Before training, we initialize the geometric references $\boldsymbol{\mu}_L$, $\bar{\mathbf{u}}_L$, and $\mathbf{U}_0$. During training, they are refreshed from the updated source-live embeddings at approximately one-third and two-thirds of each epoch and once more at its end. Each refresh updates only these references, leaving the learned nonlinear manifold parameters unchanged.

B.6 Implementation Details of the Adaptive Live Manifold

The centered tangent feature is $\mathbf{y}_i \in \mathbb{R}^{512}$. We use a latent dimension of $d = 32$ and a nonlinear hidden dimension of $h = 128$. The encoder and decoder are

$$\begin{aligned} \mathbf{a}_i &= W_E\mathbf{y}_i \in \mathbb{R}^{32}, \\ \hat{\mathbf{y}}_i &= W_0\mathbf{a}_i + W_2\,\mathrm{GELU}(W_1\mathbf{a}_i + \mathbf{b}_1) \in \mathbb{R}^{512}, \end{aligned} \tag{53}$$

with parameter dimensions are $W_E \in \mathbb{R}^{32\times512}, W_0 \in \mathbb{R}^{512\times32}, W_1 \in \mathbb{R}^{128\times32}, \mathbf{b}_1 \in \mathbb{R}^{128}, W_2 \in \mathbb{R}^{512\times128}$. Given the initial PCA basis $\mathbf{U}_0 \in \mathbb{R}^{512\times32}$, we initialize

$$W_E = \mathbf{U}_0^\top, \qquad W_0 = \mathbf{U}_0, \qquad W_2 = \mathbf{0}, \tag{54}$$

while W_1 and $\mathbf{b}_1$ retain their default random initialization. The initial reconstruction is therefore $\hat{\mathbf{y}}_i = \mathbf{U}_0\mathbf{U}_0^\top\mathbf{y}_i$. All encoder and decoder parameters are trainable; $\mathbf{U}_0$ remains fixed.

For subspace regularization, we compute

$$\mathbf{Q}_E = \mathrm{qf}(W_E^\top), \qquad \mathbf{Q}_G = \mathrm{qf}(W_0), \tag{55}$$

where $\mathrm{qf}(\cdot)$ returns the orthonormal factor of a reduced QR decomposition. Both bases have dimension 512×32, and their overlap matrices with $\mathbf{U}_0$ have dimension 32×32. The regularizer constrains the encoder row space and linear decoder column space.

The reconstruction loss uses only live samples, with the input, target, and metric factors detached, so its gradients update only the encoder and decoder. The reconstruction and subspace loss weights are $\lambda_{\mathrm{rec}} = 0.05$ and $\lambda_{\mathrm{sub}} = 0.01$, respectively.

B.7 Jacobian-Regularized Local Tangent Fitting

For each sample, the decoder is locally approximated around the latent coordinate $\mathbf{a}_i$ using a first-order Taylor expansion:

$$G_\phi(\mathbf{a}_i + \boldsymbol{\delta}) \approx G_\phi(\mathbf{a}_i) + \mathbf{J}_i\boldsymbol{\delta}, \tag{56}$$

where the sample-specific Jacobian is computed as

$$\mathbf{J}_i = \frac{\partial G_\phi(\mathbf{a}_i)}{\partial \mathbf{a}_i} = \mathbf{W}_0 + \mathbf{W}_2\,\mathrm{Diag}\big[\mathrm{GELU}'(\mathbf{W}_1\mathbf{a}_i + \mathbf{b}_1)\big]\,\mathbf{W}_1. \tag{57}$$

The reconstruction residual in the tangent space is defined as $\mathbf{r}_i = \mathbf{y}_i - \widehat{\mathbf{y}}_i$. The local tangent coordinate $\boldsymbol{\delta} \in \mathbb{R}^d$ is estimated by fitting $\mathbf{J}_i\boldsymbol{\delta}$ to the residual. To provide a coordinate prior, we

use K learnable live variation primitives $\{\mathbf{p}_k\}_{k=1}^{K}$. The primitive assignment is obtained by $\boldsymbol{\alpha}_i = \mathrm{softmax}(g_\psi(\mathbf{r}_i)/\tau)$, and the initial coordinate prior is computed as

$$\boldsymbol{\delta}_i^0 = \sum_{k=1}^{K} \alpha_{i,k}\mathbf{p}_k. \tag{58}$$

The tangent coordinate is obtained by solving the regularized objective:

$$\boldsymbol{\delta}_i^* = \arg\min_{\boldsymbol{\delta}} \|\mathbf{r}_i - \mathbf{J}_i\boldsymbol{\delta}\|_2^2 + s_i\lambda_{\mathrm{r}}\|\boldsymbol{\delta}\|_2^2 + s_i\lambda_{\mathrm{p}}\left\|\boldsymbol{\delta} - \boldsymbol{\delta}_i^0\right\|_2^2, \tag{59}$$

where

$$s_i = \mathrm{sg}\left(\max\left\{\frac{\mathrm{tr}(\mathbf{J}_i^\top\mathbf{J}_i)}{d}, 10^{-8}\right\}\right). \tag{60}$$

The closed-form solution is

$$\boldsymbol{\delta}_i^* = \left[\mathbf{J}_i^\top\mathbf{J}_i + s_i(\lambda_{\mathrm{r}} + \lambda_{\mathrm{p}})\mathbf{I}_d\right]^{-1}\left[\mathbf{J}_i^\top\mathbf{r}_i + s_i\lambda_{\mathrm{p}}\boldsymbol{\delta}_i^0\right]. \tag{61}$$

In implementation, the corresponding linear system is solved directly without explicitly computing the matrix inverse. The fitted tangent component and the remaining regularized residual are computed as

$$\mathbf{r}_i^{\tan} = \mathbf{J}_i\boldsymbol{\delta}_i^*, \tag{62}$$

$$\mathbf{r}_i^{\mathrm{res}} = \mathbf{r}_i - \mathbf{r}_i^{\tan}. \tag{63}$$

The regularized deviation score is

$$S(\mathbf{x}_i) = \|\mathbf{r}_i^{\mathrm{res}}\|_{\boldsymbol{\mu}_L}. \tag{64}$$

We use an ambient dimension of 512, a latent dimension of $d = 32$, and a decoder hidden dimension of 128. Thus, $\mathbf{J}_i \in \mathbb{R}^{512\times32}$, $\mathbf{r}_i, \mathbf{r}_i^{\tan}, \mathbf{r}_i^{\mathrm{res}} \in \mathbb{R}^{512}$, and $\boldsymbol{\delta}_i^0, \boldsymbol{\delta}_i^* \in \mathbb{R}^{32}$. We set $K = 32$ and $\tau = 0.1$. The assignment network g_ψ consists of $\mathrm{LayerNorm}(512)$, $\mathrm{Linear}(512, 128)$, GELU, and $\mathrm{Linear}(128, 32)$, with biases in both linear layers. Each primitive $\mathbf{p}_k \in \mathbb{R}^{32}$ is initialized from $\mathcal{N}(\mathbf{0}, 0.02^2\mathbf{I}_{32})$. The resulting linear system has size 32×32.

B.8 Liveness-Preserving Augmentation

The LPA branch applies strong appearance perturbations only to source-domain training images. The augmentation pipeline includes color jitter, random grayscale conversion, Gaussian blur, random erasing, channel gain adjustment, and global brightness shift. The augmented view shares the same visual encoder and hyperbolic manifold with the original view. No target-domain samples are used during augmentation generation.

B.9 Asymmetric Gradient Routing

During training, gradients are routed according to sample labels. For source-domain live samples, gradients are propagated through the visual encoder, manifold encoder/decoder, and live variation primitive dictionary. For spoof samples, gradients to the manifold parameters and primitive dictionary are stopped, while the visual encoder remains trainable.

B.10 Live Radius and Margin Calibration

The live radius R_L and spoof margin m are determined from the deviation scores of source-domain training live samples. Let $q_{0.95}^{(0)}$ and $q_{0.95}^{(1)}$ denote the 95th percentiles of these scores after the initial geometry construction and after the geometry refresh at the end of the first epoch, respectively. The spoof margin is initialized before training as $m = \max\left(1.5q_{0.95}^{(0)}, 0.05\right)$, whereas the live radius is initialized after the first epoch as $R_L = q_{0.95}^{(1)}$. Both quantities remain fixed after their respective initializations and are not updated by gradient descent. During the first epoch, before the live radius constraint is activated, the live loss is the mean squared deviation score over live samples in the mini-batch. R_L and m serve only as training constraints and are not used directly as decision thresholds.

B.11 Evaluation Protocol

Following prior cross-domain FAS studies (Yu et al., 2025; Mao & Li, 2025), we report HTER and AUC for each evaluation setting. HTER is defined as

$$\mathrm{HTER}(t) = \frac{\mathrm{FAR}(t) + \mathrm{FRR}(t)}{2}, \tag{65}$$

where FAR is the proportion of spoof samples classified as live and FRR is the proportion of live samples classified as spoof. Let t_{dec} denote the operating threshold determined according to the adopted evaluation protocol.

AUC is computed from the negative deviation scores, with live samples treated as the positive class. Lower HTER and higher AUC indicate better performance. Given t_{dec}, a sample is classified as live if $S(x) \leq t_{\mathrm{dec}}$ and as spoof otherwise. The appendix visualizations normalize scores by the corresponding decision threshold, such that $S(x)/t_{\mathrm{dec}} = 1$ marks the decision boundary.

B.12 Training Details

We optimize HypLive-FAS using AdamW with a weight decay of 10^{-3}. A parameter-group-wise learning rate strategy is adopted: the CLIP visual encoder is optimized with a learning rate of 3×10^{-6}, the learnable prompt parameters with 5×10^{-5}, and the manifold-related parameters with 1×10^{-5}. The model is trained for 8 epochs and the effective batch size is set to 96, implemented using two micro-batches of size 48 with gradient accumulation. The maximum gradient norm is clipped to 5.0. The loss weights used for training HypLive-FAS are summarized in Table 8. During inference, only the original image branch is used, while data augmentation and geometric updates are disabled.

Table 8: Loss weights used for training HypLive-FAS.

Loss term	Weight
λ_{live}	1.0
λ_{spoof}	1.0
λ_{sem}	$0.05 \rightarrow 0.025 \rightarrow 0$
λ_{rec}	0.05
λ_{sub}	0.01
λ_{aug}	0.0025
λ_{dir}	0.001
λ_{score}	0.0005
λ_{r}	0.001
λ_{p}	0.10

C Additional Complexity Analysis

To assess the computational efficiency of HypLive-FAS, we profile the model trained under Protocol 1 on an NVIDIA GeForce RTX 4080 GPU. All measurements use an input resolution of 224×224, a batch size of 1, and the same numerical precision for both methods. Inference latency is averaged over 100 synchronized forward passes following 20 warm-up iterations, and the corresponding FLOPs are measured using PyTorch Profiler. Table 9 compares HypLive-FAS with the CLIP ViT-B/16 visual encoder in terms of model capacity and inference cost.

Specifically, **Params** and **Trainable Params** denote the total and trainable parameter counts, respectively. **GFLOPs** reports the per-image computational cost, while **Inference Time** and **FPS** measure the average latency and corresponding throughput at batch size 1. **Memory** denotes the peak auxiliary GPU workspace during inference, excluding the storage occupied by model parameters and input tensors.

HypLive-FAS contains 152.31M parameters in total, compared with 86.19M parameters in the CLIP ViT-B/16 visual encoder. Most of this difference comes from the frozen CLIP text encoder used to

Table 9: Model size and inference complexity on an NVIDIA GeForce RTX 4080 at 224×224 resolution and batch size 1. The CLIP baseline refers to its ViT-B/16 visual encoder.

Method	Model size		Inference cost			
	Params (M)↓	Trainable (M)↓	GFLOPs↓	Time (ms)↓	FPS↑	Memory (MB)↓
CLIP ViT-B/16	86.19	86.19	33.71	2.930	341.35	9.43
HypLive-FAS	152.31	88.88	55.20	6.500	153.84	11.00

construct the semantic live-face prompts. Excluding the text encoder, the proposed modules introduce approximately 2.69M additional trainable parameters, including the visual projection, local-global fusion, visual-text adaptation, and hyperbolic manifold components.

HypLive-FAS requires 55.20 GFLOPs per image in the current implementation, including semantic-prompt encoding and the proposed fusion and manifold operations. Because the text encoder and learned prompts remain fixed during inference, their text representations can be precomputed and cached, potentially reducing the practical inference cost.

Despite the additional computations, HypLive-FAS achieves 153.84 FPS with a batch size of 1 on an NVIDIA GeForce RTX 4080. Its peak inference workspace is 11.00 MB, only 1.57 MB higher than that of the CLIP visual encoder. This result indicates that the proposed modules incur limited additional workspace overhead while retaining real-time inference speed.

D Other Experimental Results

D.1 Effect of the Number of Live Variation Primitives

To investigate the sensitivity of HypLive-FAS to the primitive dictionary size, we vary the number of learnable live variation primitives K and report the HTER and AUC averaged over the four target domains under Protocol 1. As shown in Table 10, the default setting $K = 32$ achieves the best overall performance, with an HTER of 1.19% and an AUC of 99.64%. A smaller dictionary provides insufficient capacity to represent the diverse tangent-coordinate patterns induced by valid live variations, limiting the effectiveness of the coordinate prior. Conversely, increasing the dictionary size to $K = 64$ introduces redundant primitives and greater flexibility to fit source-specific variations, which may weaken cross-domain generalization. These results suggest that $K = 32$ provides an effective balance between the expressiveness and generalizability of the live-variation dictionary.

Table 10: Sensitivity to the number of live variation primitives. HTER and AUC are averaged over the four Protocol 1 target domains; $K = 32$ is the default setting used by the final model.

K	HTER(%)↓	AUC(%)↑
8	1.80	99.50
16	1.80	99.52
32	**1.19**	**99.64**
64	2.15	99.39

D.2 Effect of the PCA Rank

To examine the sensitivity of HypLive-FAS to the dimensionality of the PCA-initialized live manifold, we vary the PCA rank and report the HTER and AUC averaged over the four target domains under Protocol 1. As shown in Table 11, the default rank of 32 achieves the best overall performance. A smaller rank excessively compresses the live tangent space and may discard valid variations caused by identity, pose, illumination, and acquisition conditions, causing some live samples to produce unnecessarily large deviation scores. In contrast, a larger rank retains low-variance and potentially source-specific directions, making the learned manifold overly flexible. Such flexibility may allow part of the spoof-related variation to be explained by the local tangent space, thereby

Table 11: Sensitivity to the PCA rank used for live-manifold initialization. HTER and AUC are averaged over the four Protocol 1 target domains; rank 32 is the default setting used by the final model.

PCA rank	HTER(%)↓	AUC(%)↑
16	2.08	99.30
24	1.88	99.40
32	**1.19**	**99.64**
48	1.92	99.47
64	2.37	99.26
128	2.88	99.03
256	4.95	97.91

reducing the discriminative residual used for spoof detection. The results therefore suggest that rank 32 provides an effective balance between covering intrinsic live variations and preserving sensitivity to off-manifold spoof deviations.

D.3 Effect of the Primitive-Assignment Temperature

The temperature τ is used in the primitive-assignment mechanism

$$\boldsymbol{\alpha}_i = \mathrm{softmax}\left(g_\psi(\mathbf{r}_i)/\tau\right), \tag{66}$$

where $\boldsymbol{\alpha}_i$ determines the coordinate prior $\boldsymbol{\delta}_i^0 = \sum_{k=1}^{K} \alpha_{i,k}\mathbf{p}_k$ used in the regularized local tangent fitting of Eq. 6. Thus, τ controls how selectively the model combines the learned live-variation primitives. A smaller value produces a sharper assignment concentrated on a few primitives, whereas a larger value distributes the weights more uniformly across the dictionary.

Table 12: Sensitivity to the primitive-assignment temperature τ at a fixed dictionary size of $K = 32$ and PCA rank of 32. HTER and AUC are averaged over the four Protocol 1 target domains and three random seeds.

τ	HTER(%)↓	AUC(%)↑
0.05	2.31	99.34
0.10	**1.19**	**99.64**
0.20	1.91	99.48
0.40	2.09	99.46

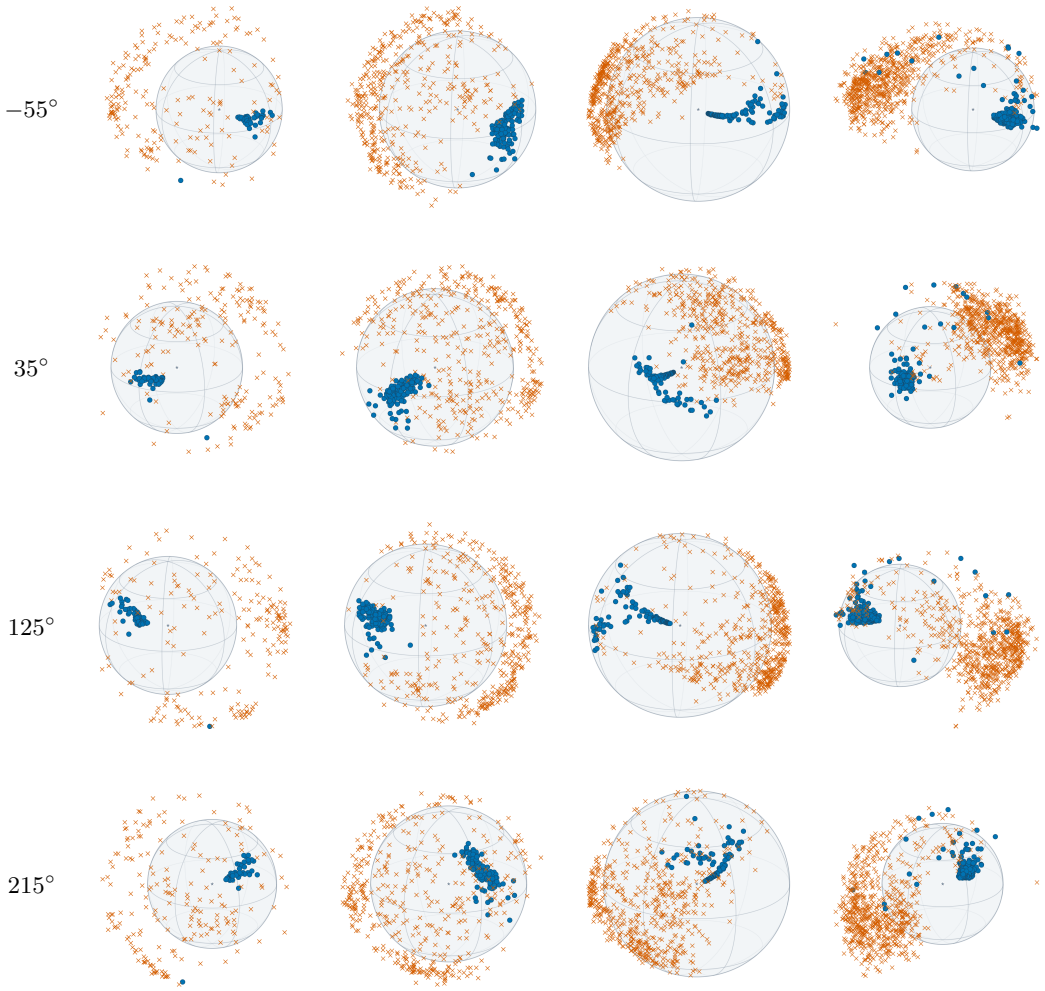
To evaluate its effect, we vary τ while fixing the dictionary size and PCA rank at their default values. As shown in Table 12, $\tau = 0.10$ achieves the best overall performance, with an average HTER of 1.19% and an average AUC of 99.64%. When $\tau = 0.05$, the overly sharp assignment restricts each sample to only a few primitives, reducing the model's ability to combine multiple valid live-variation patterns and making the coordinate prior sensitive to incorrect primitive selection. In contrast, larger temperatures produce overly smooth assignments, causing the prior to average across less relevant primitives and weakening its guidance for local tangent fitting. These results indicate that $\tau = 0.10$ provides an effective balance between assignment selectivity and prior stability.

E Additional Visualization and Analysis

E.1 Multi-view Visualization of the 3D Decision Space

To complement the single-view visualization in Figure 3, we present the same samples and three-dimensional decision geometry from four camera azimuths, as shown in Figure 5. The columns correspond to the target domains M, C, I, and O, respectively. Blue circles represent live samples, whereas orange crosses represent spoof samples. The translucent sphere denotes the decision boundary determined by the learned threshold. In particular, the first row at an azimuth of -55°

uses exactly the same camera viewpoint as the main-text visualization in Figure 3 and serves as the reference view for the three subsequent rotations.



(a) MSU-MFSD (M) (b) CASIA-MFSD (C) (c) Replay-Attack (I) (d) OULU-NPU (O)

Figure 5: Multi-view visualization of the 3D Poincaré decision-coordinate space. Columns correspond to the target domains M, C, I, and O, while rows show azimuths of $-55°$, $35°$, $125°$, and $215°$. The first row reproduces the viewpoint in Figure 3, and the elevation is fixed at $22°$. Blue circles and orange crosses denote live and spoof samples, respectively, while the translucent sphere represents the learned decision boundary.

For each sample, the radial coordinate is determined by its saved hyperbolic decision score, while its direction is obtained from the first three principal components of the centered tangent-space residual $\log_{\mathbf{0}}(\mathbf{z}_{\text{root}}) - \log_{\mathbf{0}}(\mathbf{z}_{\text{manifold}})$. Therefore, these plots should be interpreted as decision-coordinate visualizations rather than isometric embeddings of the original feature space. All cached samples are retained without subsampling. The first row uses the same viewpoint as Figure 3, with an azimuth of $-55°$. The remaining rows are obtained by rotating the camera around the vertical axis by $90°$, $180°$, and $270°$ relative to this reference view, corresponding to azimuths of $35°$, $125°$, and $215°$, respectively. Across the four rows, the underlying 3D coordinates, decision threshold, elevation angle, colors, and marker definitions remain unchanged.

As shown in Figure 5, the live samples are generally concentrated at smaller radii and predominantly remain within the learned decision boundary, whereas the spoof samples form broader and more diffuse distributions at larger radii. The domain-specific structures are also clearly visible. M and C exhibit relatively compact live clusters and clear radial separation from most spoof samples, despite

a small number of samples close to or across the boundary. In I, the live samples form a more elongated and curved structure, indicating that valid live variation need not be isotropic, while the spoof samples occupy a broad outer region. O contains a compact main live cluster together with several scattered live samples, whereas its spoof samples form a dense and asymmetric cloud, with local mixing near the boundary.

The complementary azimuths further show that some apparent overlap in a single projection is caused by occlusion along the viewing direction: point groups that overlap in one row become distinguishable after rotation. In contrast, samples that remain close across multiple viewpoints reflect genuine proximity in the constructed decision-coordinate space. Importantly, the class-dependent radial organization relative to the decision sphere is preserved under every rotation. This consistency supports the interpretation that the decision is governed by the learned hyperbolic deviation score, rather than by a favorable choice of two-dimensional projection.

E.2 Additional Analysis of Tangent Fitting and Manifold Deviation

To complement the CASIA-MFSD analysis in Figure 4, Figures 6 and 7 extend the same visualization to the four unseen target domains under Protocol 1 and the 11 unseen target datasets under Protocol 2, respectively. Both use the same normalized tangent component, manifold-deviation score, and decision-threshold definition, enabling direct comparison with the main-text result. The horizontal axis represents the normalized local tangent component $\|\mathbf{r}_i^{\tan}\|_{\boldsymbol{\mu}_L}/t_{\mathrm{dec}}$, which reflects the portion of sample variations that can be explained by the local structure of the learned live manifold. The vertical axis denotes the normalized regularized deviation score $S(\mathbf{x}_i)/t_{\mathrm{dec}}$, which measures the residual remaining after regularized local tangent fitting. The horizontal dashed line indicates the decision threshold $S(\mathbf{x}_i)/t_{\mathrm{dec}} = 1$.

As shown in Figure 6, across all four unseen target domains, live samples generally remain below the decision threshold, whereas spoof samples exhibit larger manifold deviation scores and are mainly distributed above the threshold. Notably, the two classes show certain overlap in the local tangent fitting dimension, indicating that some appearance variations can be simultaneously explained by the learned live manifold. This observation is consistent with our assumption that shared factors, such as identity information, pose variations, illumination changes, and acquisition conditions, are not spoof-specific cues and should not be directly used for liveness discrimination. In contrast, the clear separation along the deviation score dimension suggests that the remaining regularized residuals provide more discriminative information for distinguishing spoof samples.

Different target domains exhibit distinct geometric distributions. On MSU-MFSD and CASIA-MFSD, live and spoof samples show noticeable overlap in the tangent fitting dimension, while maintaining clear separation along the deviation score axis. This indicates that the proposed deviation score can effectively distinguish spoof-specific deviations from appearance variations shared across both classes. For Replay-Attack, both tangent fitting and deviation scores increase simultaneously for spoof samples, suggesting that certain replay attacks introduce additional variations beyond those captured by the live manifold. The enlarged visualization near the decision boundary further shows that most samples remain separable despite local distribution overlap. In OULU-NPU, spoof samples exhibit a broader deviation distribution with a clear upward trend, reflecting the larger diversity of acquisition conditions and spoofing patterns in this dataset, while most live samples remain concentrated below the threshold.

Figure 7 further presents the same analysis on the 11 unseen target datasets under Protocol 2. Despite the larger source-target gap, live samples generally remain below the decision threshold, whereas spoof samples tend to produce higher manifold-deviation scores. Meanwhile, the two classes frequently occupy overlapping ranges on the tangent-fitting axis. This pattern again indicates that variations explainable by the local live-manifold geometry are not sufficient for discrimination; the decisive signal is the residual deviation orthogonal to that geometry.

The dataset-specific distributions reveal several distinct behaviors. On CASIA-MFSD and MSU-MFSD, live samples form compact low-deviation clusters and spoof samples are clearly displaced above the threshold. CASIA-SURF-3DMask, HKBU-MARs-V1+, OULU-NPU, Replay-Attack, and Rose-Youtu show a stronger coupled increase in tangent fitting and deviation, suggesting that their attacks introduce both manifold-consistent appearance changes and unexplained off-manifold components. Nevertheless, most samples remain correctly separated along the vertical axis. SiW

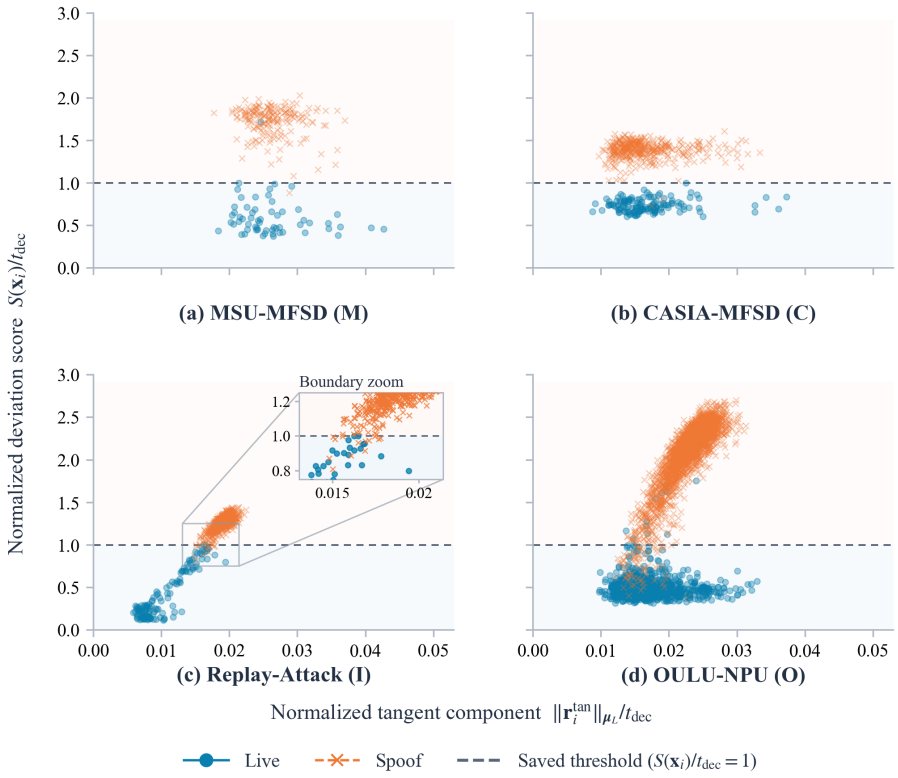


Figure 6: Extension of the main-text analysis in Figure 4 to all four unseen target domains under Protocol 1. The figure shows the relationship between normalized local tangent fitting and manifold-deviation scores. Blue circles and orange crosses denote live and spoof samples, respectively; the horizontal dashed line marks the decision threshold $S(\mathbf{x}_i)/t_{\text{dec}} = 1$. The inset for Replay-Attack enlarges the region around the decision boundary.

also exhibits a clear vertical margin, while WMCA shows a broader boundary region but retains separation for most samples. These observations are consistent with the comparatively low HTERs reported for these datasets in Table 2.

The main exceptions are HiFiMask and SiW-Mv2, where live and spoof samples overlap substantially around the decision threshold. In both datasets, some live samples cross into the spoof region and some spoof samples receive below-threshold deviation scores, directly increasing both types of classification error. This weaker vertical separation corresponds to the poorer HTER/AUC values in Table 2: 26.39%/78.85% on HiFiMask and 15.12%/89.43% on SiW-Mv2. Importantly, horizontal overlap in tangent fitting also occurs on better-performing datasets. The degradation therefore reflects reduced separability of the manifold-deviation score under these challenging acquisition shifts and attack media, rather than a general failure to model shared tangent variations. Taken together, Figures 6 and 7 show that the method preserves shared tangent variations and discriminates via residual deviations: consistent separation across most domains supports the proposed criterion, while HiFiMask and SiW-Mv2 expose its remaining cross-domain limitations.

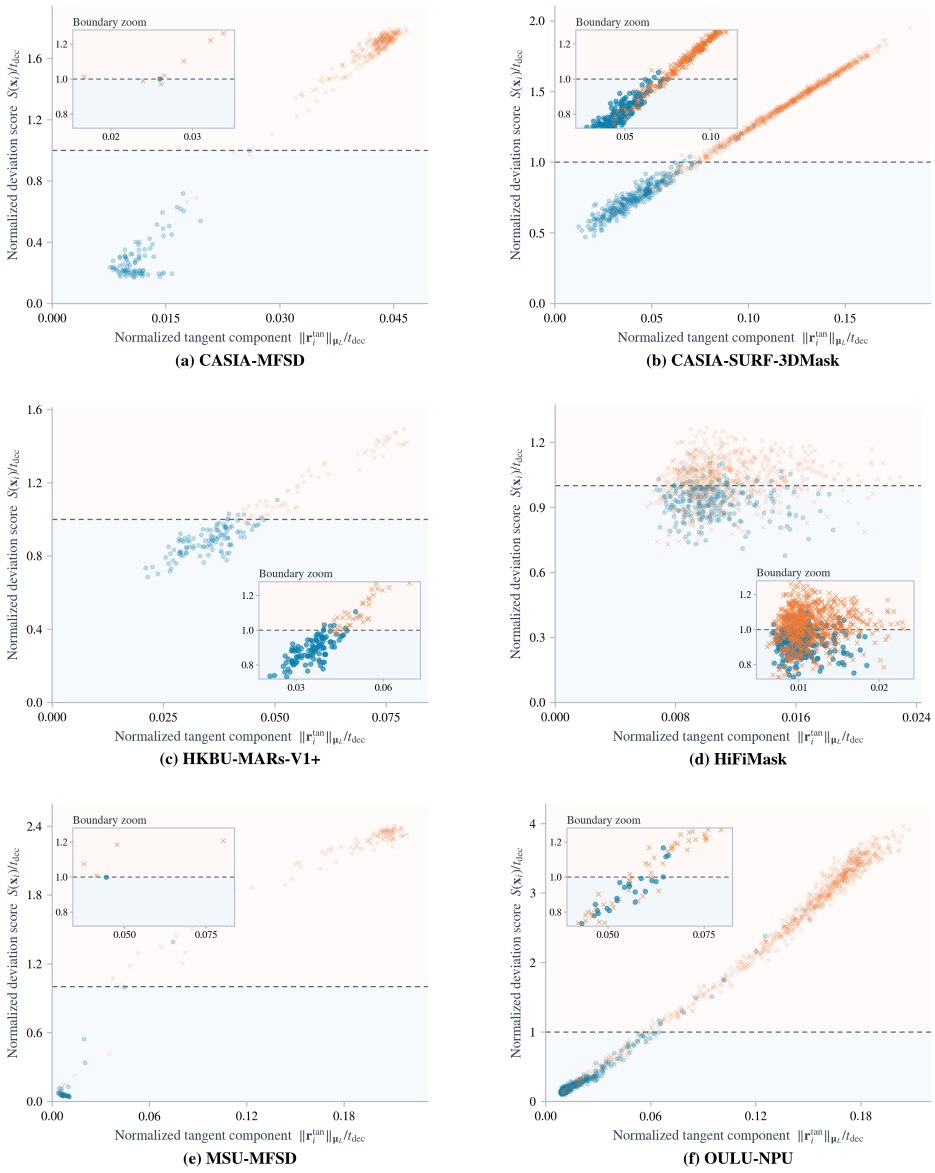


Figure 7: Further extension of the main-text analysis in Figure 4 to the 11 unseen target datasets under Protocol 2. The figure shows the relationship between normalized local tangent fitting and manifold-deviation scores. Blue circles and orange crosses denote live and spoof samples, respectively; the horizontal dashed line marks the decision threshold $S(\mathbf{x}_i)/t_{\text{dec}} = 1$. Each inset enlarges the region around the decision boundary.

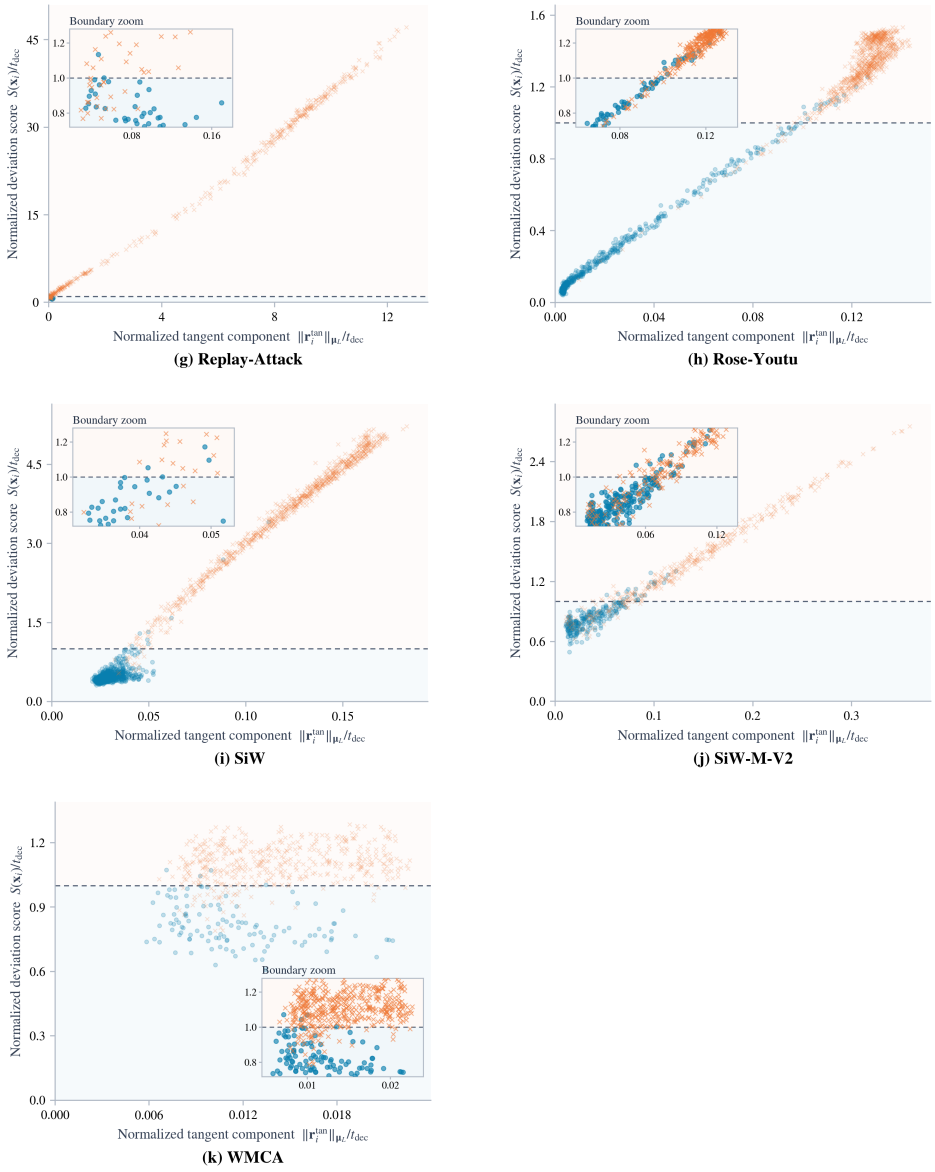


Figure 7: Relationship between normalized local tangent fitting and manifold-deviation scores across 11 face anti-spoofing datasets (continued).