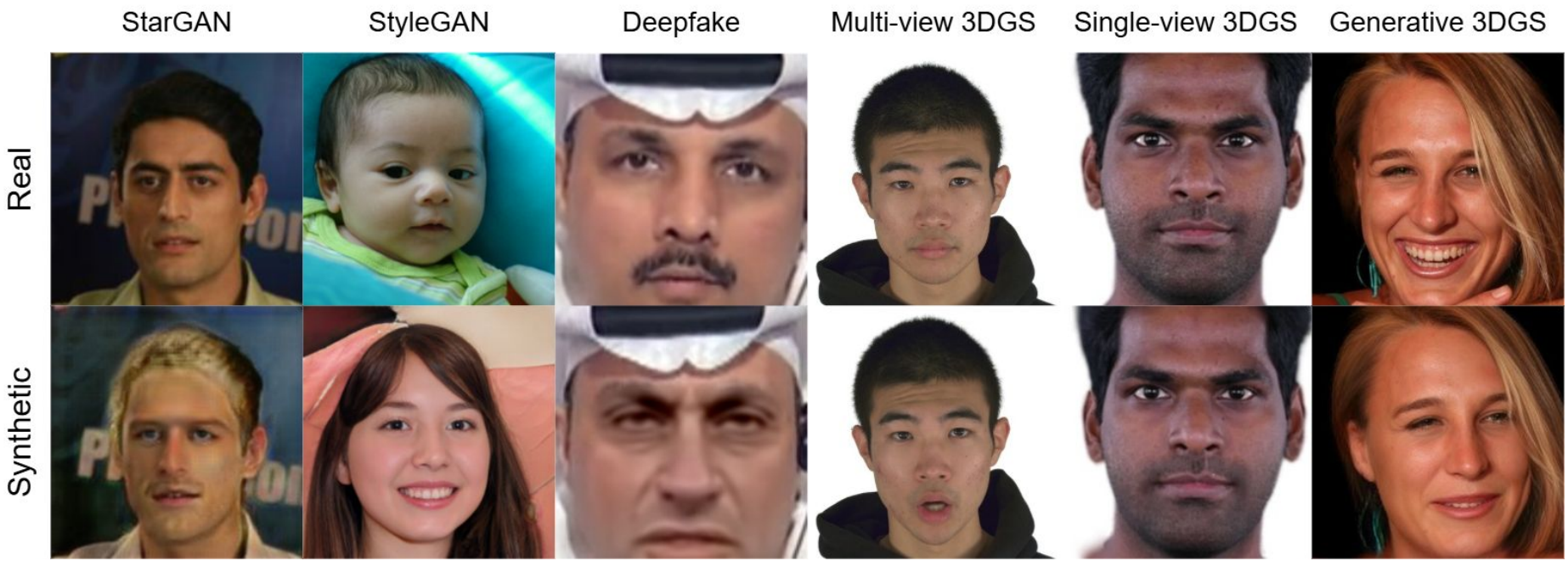
HSIC Bottleneck for Cross-Generator and Domain-Incremental Synthetic Image Detection

Chin-Chia Yang[1,4], **Yung-Yu Chuang**[1,2], **Hwann-Tzong Chen**[3] & **Tyng-Luh Liu**[4]
[1]Department of Computer Science, National Taiwan University, Taiwan
[2]AI Center of Research Excellence (AI-CoRE), National Taiwan University, Taiwan
[3]Department of Computer Science, National Tsing Hua University, Taiwan
[4]Institute of Information Science, Academia Sinica, Taiwan
{f10942093,cyy}@csie.ntu.edu.tw
htchen@cs.nthu.edu.tw,liutyng@iis.sinica.edu.tw

Motivation


StarGAN
StyleGAN
Deepfake
Multi-view 3DGS
Single-view 3DGS
Generative 3DGS
Real
Synthetic

Contributions

1. **HSIC Bottleneck for Generalized Synthetic Image Detection.** We impose an HSIC bottleneck on intermediate CLIP features to suppress text-alignment nuisances and amplify image-label dependence, substantially improving cross-generator generalization.
2. **HSIC-Guided Replay (HGR) for Continual Adaptation.** We introduce an HSIC-driven exemplar selection and weighting scheme that delivers compact yet effective replay, enabling adaptation to 3DGS content while preserving prior accuracy.
3. **3DGS Synthetic Image Benchmark.** We curate a 3DGS rendered image suite spanning multi-view reconstruction, single-view reconstruction, and a generative 3DGS pipeline, offering a benchmark to advance research on synthetic image detection.

Hilbert-Schmidt Independence Criterion (HSIC)

HSIC (Gretton et al., 2005) is a kernel-based measure of statistical dependence between random variables, defined via reproducing kernel Hilbert spaces (RKHSs). Let $\mathbf{a} \in \mathbb{A}$ and $\mathbf{b} \in \mathbb{B}$ be random variables associated with RKHSs $(\mathbb{F}, k)$ and $(\mathbb{G}, \ell)$, induced by feature maps $\phi : \mathbb{A} \to \mathbb{F}$ and $\psi : \mathbb{B} \to \mathbb{G}$, respectively. Let the corresponding mean embeddings be $\boldsymbol{\mu}_{\mathbf{a}} = \mathbb{E}[\phi(\mathbf{a})]$ and $\boldsymbol{\mu}_{\mathbf{b}} = \mathbb{E}[\psi(\mathbf{b})]$. The cross-covariance operator $\mathcal{C}_{\mathbf{ab}} : \mathbb{G} \to \mathbb{F}$ is defined as

$$\mathcal{C}_{\mathbf{ab}} = \mathbb{E}\big[(\phi(\mathbf{a}) - \boldsymbol{\mu}_{\mathbf{a}}) \otimes (\psi(\mathbf{b}) - \boldsymbol{\mu}_{\mathbf{b}})\big]. \tag{1}$$

The population HSIC is the squared Hilbert–Schmidt norm of this operator:

$$\mathrm{HSIC}(P_{\mathbf{ab}}, \mathbb{F}, \mathbb{G}) = \left\|\mathcal{C}_{\mathbf{ab}}\right\|_{\mathrm{HS}}^2. \tag{2}$$

Hilbert-Schmidt Independence Criterion (HSIC)

Kernel expectation form (population). Let $(\mathbf{a}', \mathbf{b}')$ be an independent copy of $(\mathbf{a}, \mathbf{b})$. Expanding equation 2 yields the following equivalent expression in terms of kernels k and ℓ:

$$\begin{aligned}\mathrm{HSIC}(\mathbf{a}, \mathbf{b}) \;&= \mathbb{E}_{\mathbf{a}\mathbf{a}'\mathbf{b}\mathbf{b}'}\left[k(\mathbf{a}, \mathbf{a}')\,\ell(\mathbf{b}, \mathbf{b}')\right] \;+\; \mathbb{E}_{\mathbf{a}\mathbf{a}'}\left[k(\mathbf{a}, \mathbf{a}')\right]\mathbb{E}_{\mathbf{b}\mathbf{b}'}\left[\ell(\mathbf{b}, \mathbf{b}')\right] \\ &\quad - \;2\,\mathbb{E}_{\mathbf{a}\mathbf{b}}\Big[\,\mathbb{E}_{\mathbf{a}'}k(\mathbf{a}, \mathbf{a}')\;\mathbb{E}_{\mathbf{b}'}\ell(\mathbf{b}, \mathbf{b}')\,\Big],\end{aligned} \tag{3}$$

which is zero if and only if $\mathbf{a}$ and $\mathbf{b}$ are independent under suitable conditions, e.g., when characteristic kernels are used.

Hilbert-Schmidt Independence Criterion (HSIC)

Empirical estimator. Given n i.i.d. samples $\{(a_i, b_i)\}_{i=1}^n$ from $P_{\mathbf{ab}}$, define the Gram matrices $\boldsymbol{K}, \boldsymbol{L} \in \mathbb{R}^{n\times n}$ by $\boldsymbol{K}_{ij} = k(a_i, a_j)$ and $\boldsymbol{L}_{ij} = \ell(b_i, b_j)$. Let $\mathbf{1} \in \mathbb{R}^n$ denote the all-ones vector, $\boldsymbol{I}_n \in \mathbb{R}^{n\times n}$ the identity matrix, and the centering matrix $\boldsymbol{H} = \boldsymbol{I}_n - \frac{1}{n}\mathbf{1}\mathbf{1}^\top$, which is symmetric and idempotent. A commonly used (biased) V-statistic estimator is

$$\widehat{\mathrm{HSIC}}(\mathbf{a}, \mathbf{b}) = \frac{1}{(n-1)^2}\,\mathrm{tr}\big(\boldsymbol{K}\boldsymbol{H}\boldsymbol{L}\boldsymbol{H}\big) = \frac{1}{(n-1)^2}\,\mathrm{tr}\big(\boldsymbol{H}\boldsymbol{K}\boldsymbol{H}\boldsymbol{H}\boldsymbol{L}\boldsymbol{H}\big) = \frac{1}{(n-1)^2}\,\mathrm{tr}\big(\bar{\boldsymbol{K}}\bar{\boldsymbol{L}}\big), \tag{4}$$

where $\mathrm{tr}(\cdot)$ denotes the standard matrix trace operator. This estimator provides an efficient empirical estimate without requiring density models. In practice, Gaussian RBF kernels are often adopted for k and ℓ, and the bandwidth can be set by the median heuristic. The centered versions $\bar{\boldsymbol{K}} = \boldsymbol{H}\boldsymbol{K}\boldsymbol{H}$ and $\bar{\boldsymbol{L}} = \boldsymbol{H}\boldsymbol{L}\boldsymbol{H}$ correspond to centering the associated feature maps, so that equation 4 matches the population definition equation 2 through equation 3.

Method

3.1 HSIC Bottleneck

Let the model be $h_\theta = g_{\theta_g} \circ f_{\theta_f}$, where f_{θ_f} is an encoder and g_{θ_g} is a classifier. In DualHSIC (Wang et al., 2023b), the encoder is a ResNet with L intermediate layers. Given an input representation x, label y, and layer-j feature Z_j ($j = 1, \ldots, L$), the layer-wise HSIC objective is

$$\mathcal{L}_{\mathrm{HB}}(\theta_f) \;=\; \lambda_x \sum\nolimits_{j=1}^{L} \widehat{\mathrm{HSIC}}(x, Z_j) \;-\; \lambda_y \sum\nolimits_{j=1}^{L} \widehat{\mathrm{HSIC}}(y, Z_j), \tag{5}$$

where λ_x encourages compression of input information, while λ_y encourages dependence on the label $y \in \{0, 1\}$, indicating whether the sample is synthetic (1) or authentic (0).

Unlike DualHSIC, which applies HSIC at all intermediate layers, we use CLIP ViT as the feature extractor. For each image, we form a CLIP feature representation x by concatenating features from its 24 intermediate layers and the final layer, and then compress it into $z = f_{\theta_f}(x)$. Adapting equation 5 to this setting gives

$$\mathcal{L}_{\text{HSIC-Bottleneck}}(\theta_f) \;=\; \lambda_x \widehat{\mathrm{HSIC}}(x, z) \;-\; \lambda_y \widehat{\mathrm{HSIC}}(y, z). \tag{6}$$

Method

3.2 Training Objective

For each sample i, let x_i be its CLIP feature representation and $z_i = f_{\theta_f}(x_i)$ be the compressed feature. The classifier g_{θ_g} outputs a logit $u_i = g_{\theta_g}(z_i)$, and the corresponding probability is $p_i = \sigma(u_i)$, where $\sigma(\cdot)$ is the sigmoid. For binary labels $y_i \in \{0, 1\}$, we use binary cross-entropy:

$$\mathcal{L}_{\text{BCE}}(\theta_f, \theta_g) = -\frac{1}{n} \sum_{i=1}^{n} \Big[y_i \log p_i + (1 - y_i) \log\big(1 - p_i\big) \Big]. \tag{7}$$

The final objective is

$$\mathcal{L}_{\text{total}}(\theta_f, \theta_g) = \mathcal{L}_{\text{HSIC-Bottleneck}}(\theta_f) + \mathcal{L}_{\text{BCE}}(\theta_f, \theta_g). \tag{8}$$

Method

3.3 HSIC-GUIDED REPLAY (HGR)

We combine an HSIC-inspired relevance score with a k-center coverage term to select exemplars for the rehearsal buffer. A nonnegative weight $\lambda_{\text{kc}} \geq 0$ controls the strength of the k-center regularizer: λ_{kc}=0 yields pure relevance-based selection, while larger values emphasize coverage. Selection is performed per class $c \in \{0, 1\}$, but we omit c below for simplicity. Let $\mathcal{X} = \{x_i\}_{i \in \mathcal{I}}$ be the candidate set of CLIP feature representations with index set $\mathcal{I}$. At step t, let $S_{t-1} \subset \mathcal{I}$ be the selected indices (with $S_0 = \varnothing$), and define the active set $A_t = \mathcal{I} \setminus S_{t-1}$.

For each $x_i \in \mathcal{X}$, compute $z_i = f_{\theta_f}(x_i)$. Form the Gaussian RBF Gram matrix $\boldsymbol{K}$ on $\{z_i\}_{i \in \mathcal{I}}$ and let $\bar{\boldsymbol{K}}$ be its centered version as in equation 4. The HSIC-inspired relevance score for index $i \in \mathcal{I}$ is

$$r_i \;=\; \left\|\bar{\boldsymbol{K}}_{i,:}\right\|_2^2. \tag{9}$$

To promote coverage and reduce redundancy, we add a k-center term in feature space, following the coreset view of Sener & Savarese (2018). Define, for $i \in A_t$,

$$d_i(t) = \begin{cases} \|z_i - \mu\|_2^2, & t{=}1, \\ \min\limits_{j \in S_{t-1}} \|z_i - z_j\|_2^2, & t{\geq}2, \end{cases} \quad \text{with} \quad \mu = \tfrac{1}{|\mathcal{X}|} \sum_{j \in \mathcal{I}} z_j,$$

so that larger $d_i(t)$ favors points farther from the already selected set.

Selection rule. HGR selects exemplars by minimizing the λ_{kc}-regularized score

$$s_i(t) = \big(1 - \mathcal{N}(r_i)\big) \;+\; \lambda_{\text{kc}} \big(1 - \mathcal{N}(d_i(t))\big), \qquad i \in A_t. \tag{10}$$

Here, $\mathcal{N}(\cdot)$ denotes a normalization operation. We choose $i_t^\star = \arg\min_{i \in A_t} s_i(t)$ and update $S_t = S_{t-1} \cup \{i_t^\star\}$ until $|S_t| = m_c$, where m_c is the number of exemplars allocated to class c. Repeating this for $c \in \{0, 1\}$ and taking the union across classes and domains yields the replay buffer. Intuitively, HGR favors items that are both highly ranked by the relevance score (large r_i) and provide coverage (large $d_i(t)$).

Results

Table 1: **Cross-generator generalization with diffusion-trained detectors (ACC/AP)**. Each cell reports ACC/AP (%). Within each dataset row, the highest, second highest, and third highest ACC are shaded red, orange, and yellow, respectively. The SDV1.4 row label cell is shaded green to indicate the diffusion training source.

Dataset	Method (ACC/AP %)							
	CNNSpot	LGrad	UniFD	NPR	RINE	VIB-Net	Ours	Ours w/ intermediate
SDV1.4	99.48/99.98	99.12/99.94	83.55/96.04	100.00/100.00	96.10/100.00	99.55/100.00	99.33/99.98	99.92/100.00
SDV1.5	99.35/99.83	99.05/99.92	84.80/96.26	99.90/99.97	95.90/99.90	99.20/99.97	99.24/99.95	99.81/100.00
ADM	50.10/51.10	53.00/58.52	53.35/66.34	73.00/94.70	53.70/86.80	73.85/95.49	85.82/97.69	91.10/99.61
GLIDE	50.90/58.80	64.24/84.00	75.30/93.73	89.70/95.80	56.70/95.70	74.25/97.13	94.97/99.26	97.07/99.85
Midjourney	56.42/67.93	76.34/91.06	71.60/92.08	82.30/95.50	59.20/97.00	88.05/97.81	83.12/97.35	80.86/99.34
Wukong	97.90/99.80	97.53/99.72	73.55/90.98	100.00/100.00	85.00/99.80	98.25/99.93	98.62/99.92	99.86/100.00
VQDM	50.04/49.92	50.93/56.34	55.10/74.53	68.30/86.30	57.40/96.50	89.35/97.00	93.69/99.25	99.42/99.99
ProGAN	50.27/53.15	61.61/83.59	58.65/51.77	60.30/83.30	68.90/81.40	89.70/96.59	97.54/99.64	99.49/99.99
CycleGAN	49.81/50.23	60.74/90.24	59.30/63.42	67.20/94.90	66.50/96.80	88.60/98.44	97.92/99.86	99.66/99.97
BigGAN	50.10/49.79	48.82/47.51	61.45/75.81	59.20/72.00	52.40/94.00	91.20/97.17	90.28/98.01	91.75/99.74
StyleGAN	50.98/55.98	61.43/82.74	56.80/54.12	58.00/82.70	53.30/71.60	74.10/84.31	94.42/98.93	88.15/97.89
StarGAN	49.77/47.07	50.17/99.19	61.45/54.93	73.20/97.30	58.00/99.80	80.70/97.60	96.05/99.52	100.00/100.00
GauGAN	50.38/56.08	49.70/49.25	55.30/65.99	52.00/66.00	51.80/87.90	87.15/96.94	87.38/94.51	90.02/97.60
Deepfake	51.98/54.86	50.17/66.49	58.40/70.24	74.80/85.30	51.60/80.30	72.00/81.32	66.35/76.47	82.15/93.82
SAN	50.22/54.03	56.49/65.09	72.00/83.34	89.60/95.90	62.30/88.70	81.50/93.27	90.64/96.13	88.58/95.51
Avg	60.51/63.24	65.29/78.24	65.37/75.31	76.50/89.98	64.59/91.75	85.83/95.53	91.69/97.10	93.86/98.89

Results

Table 2: **Cross-generator generalization with GAN-trained detectors (ACC/AP)**. Each cell reports ACC/AP (%). Within each dataset row, the highest, second highest, and third highest ACC are shaded red, orange, and yellow, respectively. The ProGAN row label cell is shaded green to indicate the GAN training source.

Dataset	Method (ACC/AP %)							
	CNNSpot	LGrad	UniFD	NPR	RINE	VIB-Net	Ours	Ours w/ intermediate
ProGAN	99.99/99.99	99.80/99.90	99.90/100.00	99.80/100.00	100.00/100.00	99.99/100.00	99.83/100.00	100.00/100.00
CycleGAN	87.59/96.40	86.94/94.01	98.50/99.21	96.10/98.50	99.32/99.99	99.00/99.80	88.38/99.74	93.07/99.99
BigGAN	71.18/87.50	85.63/90.75	94.50/98.31	84.40/87.80	99.60/99.94	95.75/99.29	90.08/99.53	82.05/99.94
StyleGAN	89.95/96.94	91.08/99.80	84.40/97.98	97.70/99.80	88.86/99.44	91.25/98.79	91.37/98.33	93.41/100.00
StarGAN	94.60/94.24	99.27/99.98	95.85/99.35	99.30/99.90	99.55/100.00	98.95/99.72	97.45/99.87	100.00/100.00
GauGAN	81.44/98.28	72.49/79.29	99.50/99.80	82.50/85.50	99.77/100.00	99.70/99.99	82.32/99.99	68.33/99.98
Deepfake	51.69/64.42	56.42/71.71	67.40/82.04	80.20/82.40	80.57/97.90	83.20/92.64	83.98/92.97	82.48/96.82
SAN	50.00/55.89	44.47/45.09	56.50/82.18	69.20/71.60	68.26/94.93	70.50/91.62	86.99/92.25	93.61/97.94
SDV1.4	50.82/52.86	63.03/70.90	63.10/85.48	76.60/84.00	83.96/98.35	71.55/87.24	85.79/92.25	98.82/99.94
SDV1.5	50.88/53.25	63.67/71.72	63.57/82.30	77.90/84.60	83.35/98.33	70.00/86.98	85.14/92.23	98.67/99.85
ADM	60.20/65.14	67.10/71.83	66.90/84.34	69.70/74.60	74.61/96.23	71.45/87.88	81.46/89.61	93.01/97.70
GLIDE	57.85/68.10	66.10/75.96	61.70/84.04	77.30/85.70	80.72/97.87	69.40/88.53	89.72/96.39	97.02/99.56
Midjourney	50.77/56.60	56.20/71.42	57.85/69.10	77.80/85.40	57.12/87.41	61.25/75.68	60.36/66.83	69.40/82.49
Wukong	51.13/51.15	63.60/66.51	71.06/90.13	76.10/80.50	84.95/98.62	75.90/90.92	88.36/95.38	98.62/99.88
VQDM	56.20/69.49	67.02/70.23	85.00/94.96	78.10/81.20	89.79/99.23	86.65/96.51	88.72/96.08	97.49/99.74
Avg	66.95/74.02	72.19/78.61	77.72/89.95	82.84/86.77	86.03/97.88	82.97/93.04	86.66/94.10	91.07/98.25

Results

Table 3: **Training from SDV1.4**. Cells show **mACC/mAP** (%). In our setup, *base* trains only on SDV1.4 without 3DGS; *Oracle* is jointly trained on SDV1.4 plus {GHA, SA, GAGAvatar} in a non-continual manner; and *iCaRL*, *CBRS*, and our *HGR* are sampling methods for the replay buffer. **Bold** highlights the best mACC among the sampling methods only.

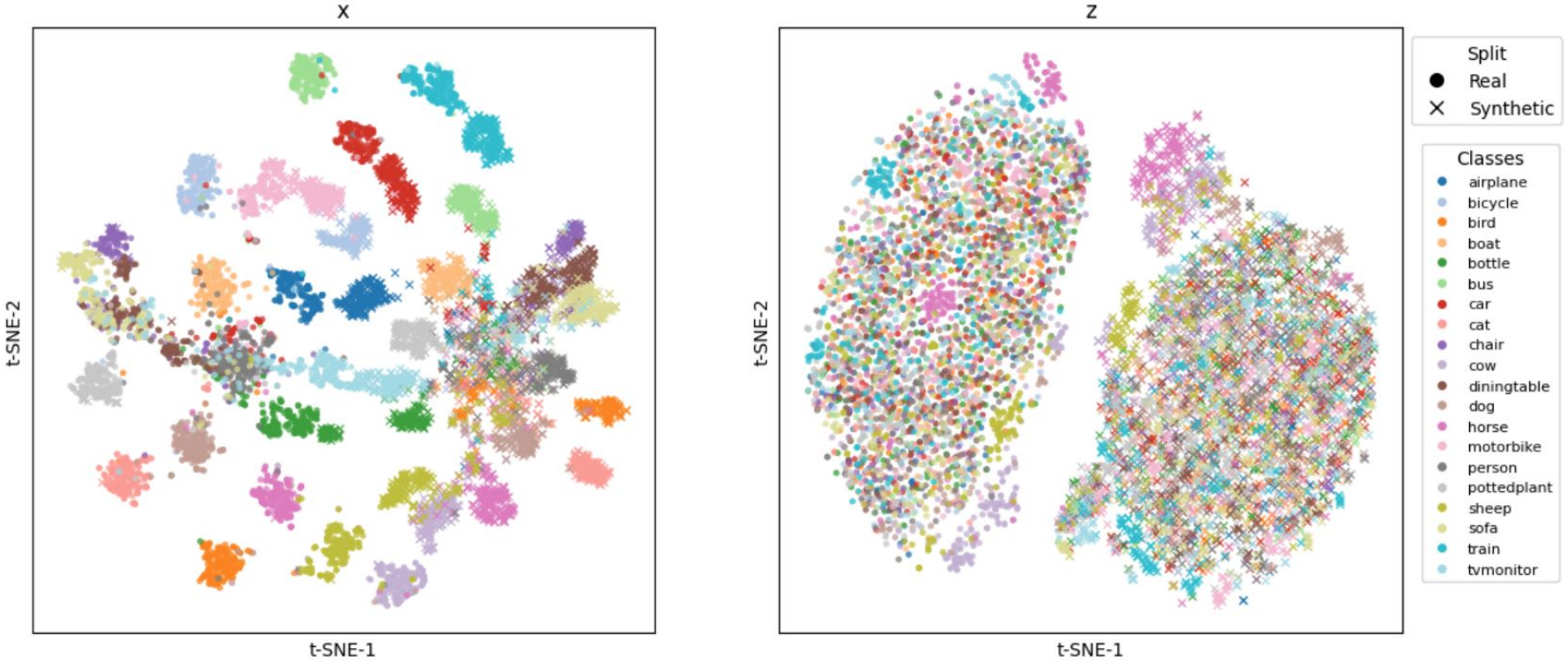
Method	Diffusion	GANs	Others	GHA	SA	GAGAvatar	Average
UniFD	71.04/87.14	58.83/61.01	65.20/76.79	54.75/54.03	58.35/54.87	55.85/56.89	63.87/71.97
RINE	72.00/96.53	58.48/88.58	56.95/84.50	50.00/60.20	50.00/55.10	50.60/59.70	62.20/86.20
base	95.43/99.83	94.85/99.20	85.37/94.67	66.05/78.10	64.65/80.66	50.39/54.56	88.27/94.26
iCaRL	96.00/99.65	91.57/97.61	78.11/93.34	96.01/99.85	94.99/99.80	94.52/99.12	92.40/98.26
CBRS	95.15/99.68	93.15/98.60	77.58/94.24	95.23/99.78	96.58/99.97	**96.02/99.50**	92.66/98.73
HGR	**97.12/99.81**	**94.00/99.07**	**82.31/94.29**	**97.06/99.77**	**98.07/99.99**	95.18/99.07	**94.38/98.92**
Oracle	95.54/99.76	95.62/99.39	78.39/96.08	94.57/98.86	98.67/99.94	94.90/99.08	93.75/99.15

Results

Table 4: **Training from ProGAN**. Cells show **mACC/mAP** (%). In our setup, *base* trains only on ProGAN without 3DGS; *Oracle* is jointly trained on ProGAN plus {GHA, SA, GAGAvatar} in a non-continual manner; and *iCaRL*, *CBRS*, and our *HGR* are sampling methods for the replay buffer. **Bold** highlights the best mACC among the sampling methods only.

Method	Diffusion	GANs	Others	GHA	SA	GAGAvatar	Average
UniFD	67.03/84.34	95.44/99.11	61.95/82.11	53.10/56.33	86.10/95.41	65.45/76.54	76.14/87.64
RINE	79.21/96.58	97.85/99.90	74.42/96.42	54.20/66.00	79.30/92.90	61.00/77.30	82.50/94.69
base	93.29/97.02	89.48/99.99	88.05/97.38	51.25/74.75	55.78/94.09	61.98/73.83	85.28/95.36
iCaRL	76.79/91.10	88.84/94.74	78.90/92.80	95.23/99.86	97.47/99.98	96.93/99.85	84.33/93.96
CBRS	80.33/92.97	89.99/95.34	77.55/90.63	**97.82/99.85**	98.07/100.00	**98.09/99.89**	86.18/94.65
HGR	**82.87/94.33**	**93.94/98.85**	**80.99/90.72**	94.71/99.47	**99.45/100.00**	95.47/99.26	**88.63/96.31**
Oracle	82.15/95.58	96.10/99.72	85.15/97.11	90.90/96.66	98.81/99.96	90.98/97.06	89.04/98.27

Results


X
t-SNE-2
t-SNE-1
Z
t-SNE-2
t-SNE-1
Split
Real
Synthetic
Classes
airplane
bicycle
bird
boat
bottle
bus
car
cat
chair
cow
diningtable
dog
horse
motorbike
person
pottedplant
sheep
sofa
train
tvmonitor


國立臺灣大學
National Taiwan University

國立清華大學
TSING HUA UNIVERSITY

中央研究院 資訊科學研究所
SINCE 1982
Institute of Information Science, Academia Sinica

Thank you!