A LEMMAS AND PROOFS

Lemma A.1.

$$TPR_{\mathcal{R}} \geq TPR_{\mathcal{M}}(f) - \frac{Lip(f)}{2} W^1(\mathcal{G}, \mathcal{T}),$$

Proof. $\text{TPR}_{\mathcal{R}}(f) = \frac{1}{2}\mathbb{E}_{\mathcal{T}}[f(x)] = \frac{1}{2}(\mathbb{E}_{\mathcal{G}}[f(x)] + \mathbb{E}_{\mathcal{T}}[f(x)] - \mathbb{E}_{\mathcal{G}}[f(x)]) = \frac{1}{2}(\text{TPR}_{\mathcal{M}}(f) + \mathbb{E}_{\mathcal{T}}[f(x)] - \mathbb{E}_{\mathcal{G}}[f(x)]) \geq \frac{1}{2}(\text{TPR}_{\mathcal{M}}(f) - |\mathbb{E}_{\mathcal{T}}[f(x)] - \mathbb{E}_{\mathcal{G}}[f(x)])|) \geq \frac{1}{2}(\text{TPR}_{\mathcal{M}}(f) - \frac{1}{\text{Lip}(f)} \sup_{g:\text{Lip}(g)=1} |\mathbb{E}_{\mathcal{T}}[f(x)] - \mathbb{E}_{\mathcal{G}}[f(x)])|) = \frac{1}{2}(\text{TPR}_{\mathcal{M}}(f) - \frac{1}{\text{Lip}(f)} W^1(\mathcal{G}, \mathcal{T})$. □

Proof of Theorem 4.1

Proof. Let $f : \mathcal{X} \to \{0, 1\}$ be an arbitrary MIA achieving FPR α, that is $\mathbb{E}_{x\sim\mathcal{D}}[f(x)] = \alpha$. Then the TPR of f at distinguishing samples from G is $\mathbb{E}_{x\sim G}[f(x)] = \beta\mathbb{E}_{x\sim\mathcal{T}}[f(x)] + (1-\beta)\mathbb{E}_{x\sim\mathcal{D}}[f(x)]$, since $G = \beta\mathcal{T} + (1-\beta)\mathcal{D}$. Substituting in the FPR of f, we have $\mathbb{E}_{x\sim G}[f(x)] = \beta\mathbb{E}_{x\sim\mathcal{T}}[f(x)] + (1-\beta)\alpha$. But $\mathbb{E}_{x\sim\mathcal{T}}$ is just the TPR at detecting samples from $\mathcal{T}$. We can take $\inf_{f:\text{FPR}(f)=\alpha}$ of both sides of the prior equation, which shows that the MIA attack achieving optimal TPR at a fixed FPR α is exactly the optimal hypothesis test for distinguishing samples from G from samples from D with a fixed FPR α. Next we note that the optimal hypothesis test in this latter scenario is characterized by the Neyman-Pearson Lemma Neyman & Pearson (1933): There exists τ_α such that $f^*(x) = \mathbf{1}\{\frac{G(x)}{D(x)} > \tau_\alpha\}$ achieves the maximum achievable TPR at fixed FPR α, and so this is the optimal MIA for detecting training samples from $\mathcal{T}$ at a fixed FPR α. It remains to be shown that we recover f^* of this form from minimizing the classification error on $\frac{1}{2}D + \frac{1}{2}G$. Under our assumption that we compute a classifier that exactly minimizes the loss on the distribution, it is a standard result that the optimal classifier is the Bayes optimal classifier $B(x) = \frac{G(x)}{G(x)+D(x)}$. Then the result follows from noting that $B(x) > \tau$ is equivalent to $\frac{G(x)}{D(x)} > \frac{1}{\frac{1}{\tau}-1}$. □

We first present pseudocode for the Robust Homer attack, Detector attack and ADIS.

B ROBUST HOMER ATTACK

The intuition behind the attack lies in Line 7 of Algorithm 1. Recall X_G denotes synthetic samples from the GAN, X_R denotes reference data sampled from $\mathcal{P}$, and $\tau \in \mathcal{E}$ denotes a test point. Let $\mu_g = \frac{1}{n}\sum_{i=0}^{n} s_i$ where $x_{g0}, x_{g1}, \ldots x_{gn} \in X_G$ and $\mu_g \in [0,1]^d$. Similarly, let $\mu_r = \frac{1}{m}\sum_{i=\mathbf{1}}^{m} x_i$ where $x_1, x_2 \ldots x_m \in X_R$ and $\mu_r \in [0,1]^d$. Then sample $x \sim \mathcal{D}$, and compute:

$$\langle \tau - x, \mu_g - \mu_r \rangle = \langle \tau, \mu_g - \mu_r \rangle - \langle x, \mu_g - \mu_r \rangle = \underbrace{\left[\langle \tau, \mu_g \rangle - \langle x, \mu_g \rangle\right]}_{(1)} + \underbrace{\left[\langle x, \mu_r, \rangle - \langle \tau, \mu_r, \rangle\right]}_{(2)}$$

Part (1) above checks if τ is more correlated with μ_g than a random sample x from $\mathcal{D}$, while (2) checks if x is more correlated with μ_r than τ. Observe that this is similar to the intuition behind the distance-based attack, but here we compute an average *similarity measure* to μ_g and μ_r, rather than a distance to the closest point.

C DETECTOR ATTACK

The target GAN configurations for the genomic data setting is shown in Table 1:

Algorithm 1 Robust Homer Attack

Require: $(X_R \in \{0,1\}^d, G, \mathcal{E}, x)$

$\mu_r = \frac{1}{m}\sum_{i=1}^{m} x_i \quad$ where $\quad x_1, x_2 \dots x_m \in X_R$

$\alpha \leftarrow \frac{1}{\sqrt{m}} + \epsilon, \quad \eta \leftarrow 2\alpha$ ▷ $\epsilon \geq 0$

$X_G \sim G$ ▷ $X_G \in \{0,1\}^d$

$\mu_g = \frac{1}{n}\sum_{i=0}^{n} s_i \quad$ where $\quad x_{g0}, x_{g1}, \dots x_{gn} \in X_G \quad$ and $\quad \mu_g \in [0,1]^d$

Let $\lfloor \mu_g - \mu_r \rceil_\eta \in [-\eta, +\eta]^d$ ▷ entry-wise truncation of $\mu_g - \mu_r$ to $[-\eta, \eta]$

for $\tau \in \mathcal{E}$ **do**

  $\rho \leftarrow \langle \tau - x, \lfloor \mu_g - \mu_r \rceil_\eta \rangle$

  if $\rho > \kappa$ **then** ▷ κ is an hyperparameter

    τ is in training set for G

  else

    τ is not in training set for G

  end if

end for

Table 1: Target GAN configurations for genomic Data

Data	SNPs Dim.	Train Data Size	Ref. Data Size	Test Size	GAN Variant
1000 Genome	805	3000	2008	1000	vanilla & WGAN-GP
	5000	3000	2008	1000	vanilla & WGAN-GP
	1000	3000	2008	1000	vanilla & WGAN-GP
dbGaP	805	6500	5508	1000	vanilla & WGAN-GP
	5000	6500	5508	1000	vanilla & WGAN-GP
	10000	6500	5508	1000	vanilla & WGAN-GP

Algorithm 2 Detector

Require: $\{f_\theta, X_R, G, \mathcal{E}\}$ ▷ f_θ=multilayer perceptron , X_R=reference data, $G(\cdot)$= black box GAN, $\mathcal{E}$ = test samples.

$epoch := 150$

$batch\_size := 100$

$lr :=$ 1e-3 ▷ learning rate

$Optimizer := Adam$ ▷ Optimizer for training

$f_s =$ `EARLYSTOP()` ▷ early stopping routine

$f_{lr} =$ `DECAYRATE(`lr`)` ▷ decaying learning rate

for $j = 0$ to $epoch - 1$ **do**

  if $j \bmod 3 = 0$ **then**

    $X_G \sim G$ ▷ sample synthetic data from GAN

    $y_{X_R} \leftarrow \mathbf{0}$ ▷ label (reference sample)

    $y_S \leftarrow \mathbf{1}$ ▷ label (synthetic sample)

    $\mathbf{X} = \{(X_G, y_G) \cup (X_R, y_{X_R})\}$ ▷ Combine data

    $\mathbf{X}_T, \mathbf{X}_V =$ `SPLIT(`$\mathbf{X}$`)` ▷ Split data into training & validation sets

  end if

  $f_\theta \leftarrow f_\theta(\mathbf{X}_T, \mathbf{X}_V, batch\_size, f_{lr}, f_s, Optimizer)$ ▷ model training and update

end for

$output \leftarrow f_\theta(\mathcal{T})$ ▷ predict test samples

Detector Prediction Score. We might ask how well does the Detector performs in classifying synthetic samples as a first step in checking how well-trained is. Table 2 shows that the Detector

predictions are very high for synthetic samples as expected. However, the goal is for the Detector to be able to distinguish training and non-training samples in the test set, given that both samples are from the same underlying distribution.

Table 2: Detector Test Accuracy

Data	GAN Variant	SNPs Dim.	Test Size	Train Epochs	Mean Accuracy
1000 Genome	vanilla	805	1000	159	0.974
	vanilla	5000	3000	240	0.980
	vanilla	1000	3000	360	0.999
dbGaP	WGAN-GP	805	1000	300	0.995
	WGAN-GP	5000	1000	600	0.998
	WGAN-GP	10000	6500	1500	0.930

D AUGMENTED DETECTOR(ADIS)

The ADIS builds on the Detector by augmenting the feature space with derived attack variables or test statistics. The pipeline is similar to that outlined in Figure 3 with the following notable additional steps:

1. Sample without replacement, $N(\approx 200)$ data point from the reference and synthetic sample data in figure(3) respectively. The subsampled data are for the computation of the reconstruction losses (see section 4.2) and are not included again in the downstream training set.
2. We apply dimensionality reduction on the training data of figure(3) - this step might be omitted depending on the dimensionality of the data.
3. Augment the resulting feature space with test statistics - for example, the distance-based test statistics (one-way and two distances computed using the held-out sample), and the DOMIAS likelihood ratio statistic $\frac{P_G(x)}{P_R(x)}$ (van Breugel et al., 2023b) where P_G and P_{X_R} are density models fitted on the synthetic and reference data respectively and x a test data point.

The remaining training steps are exactly the same as that of the Detector pipeline - Figures 3, and algorithm 2.

ADIS Training. The architecture for ADIS is the same as the Detector, but there are additional preprocessing steps and fewer epochs ($\approx$ 10-30 epochs). A fixed sample size of 300 from the reference and synthetic sample was set aside for the computation of reconstruction losses **??**. This was followed by fitting principal component analysis (PCA) (Bro & Smilde, 2014; Ringnér, 2008) on the training data (consisting of reference and synthetic data) and selecting the first 100-300 principal components - the actual component selected depends on the dimension of the original feature space, but typically 100 components were selected for 805 SNPs and 300 for 5000 SNPs. Subsequently, we computed *one-way and two-way* hamming reconstruction losses **??** on the non-transformed training data. Furthermore, as recommended in (Hilprecht et al., 2019), distance attacks also work well on reduced feature space. Thus, we computed the *two-way* L2 reconstruction loss on the PCA-transformed training data. Note that the 300 samples that was set aside had to be PCA-transformed before being used for the computation of *two-way* L2 reconstruction loss. For the computation of the test statistics of DOMIAS (van Breugel et al., 2023b), we first reduced the dimensionality of the synthetic samples and reference data separately with the fitted PCA. Then we fitted a Gaussian mixture model (Reynolds et al., 2009), P_G, on the PCA-transformed synthetic samples and another Gaussian mixture model, P_{X_R}, on the PCA-transformed reference data. Finally, we computed the test statistic $\frac{P_G(x)}{P_{X_R}(x)}$ for each PCA-transformed training point x.

Large Reference Data for ADIS It should be noted that the Detector and ADIS attacks would benefit from having a reference data sample of large size. In cases where the reference data size is

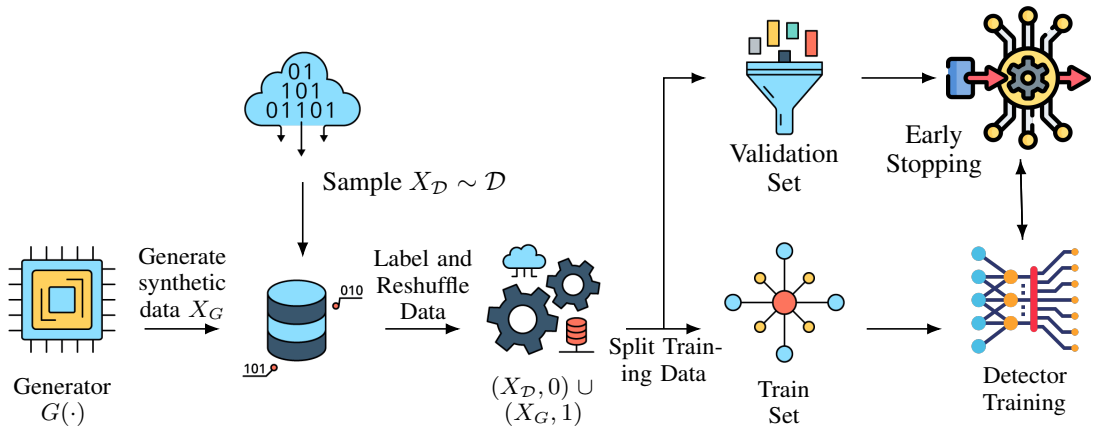


Figure 3: The pipeline for training the Detector D_θ.

small, an effective strategy would be to train a secondary GAN on the reference data. Subsequently, one can proceed to subsample synthetic reference data (from the trained secondary GAN) that are closer or within an ϵ-ball of the reference data as measured by a distance metric - i.e., using distance-based subsampling.

Results Table for GAN-Omics MIA Table 3 depicts the TPR at FPR of 0.01 , 0.1, 0.005 and 0.001 respectively (see Figures 1 and 1d) for genomic data setting.

Training DOMIAS Training DOMIAS involves 2 steps - dimensionality reduction and density estimation. For genomic data setting, we use PCA for dimension reduction following the same steps as described for ADIS. For density estimation, we fit density estimator using non-volume preserving transformation(NVP)Dinh et al. (2016). Observe, that we fit the the estimator to the synthetic and reference sample separately and then proceed to compute the likelihood ratio for any given test point. For image data, we project the image onto a low dimensional space using the encoder of a variational auto-encoderKingma & Welling (2022) trained on the reference and synthetic data. Subsequently, we fit the density estimator using NVP to the transformed data.

E Distance-based Attacks

Figure 3 illustrates the Detector pipeline. Figures 5 and 5 illustrates the intuition behind the one-way and two-way distance-based attack.

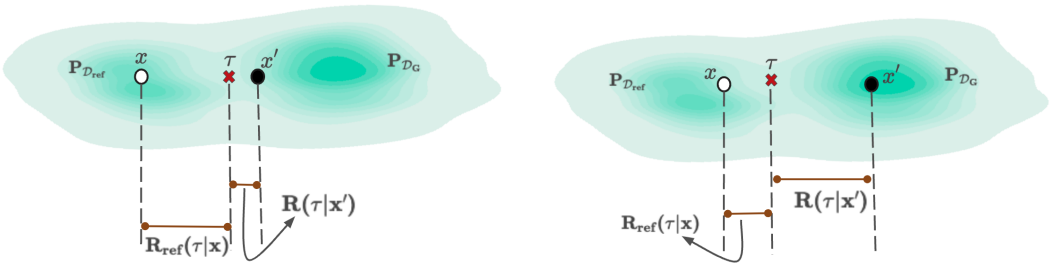
F Results for Genomic GANs

F.1 Genomic GAN Convergence Plots

For image data, visual inspection is a quick way to examine if the synthetic samples converge to the underlying real training samples. Clearly, under the genomic tabular setting, such quick visual convergence examination does not work. Recent work of Platzer (2013) proposed using both the principal component analysis (PCA) and t-distributed stochastic neighbor embedding (T-SNE) plots for the analysis of genomic data population structure. Consequently, we examine the convergence of the synthetic samples using both PCA and T-SNE. In particular, the synthetic samples converge if the population structure as captured by the PCA and T-SNE is similar to that of the corresponding real genomic samples. Figures 6a - 6f depict the 6 principal components of both the training data and synthetic samples for each configuration in table 1. This provides the first insight that the synthetic samples are indeed representative of the underlying data

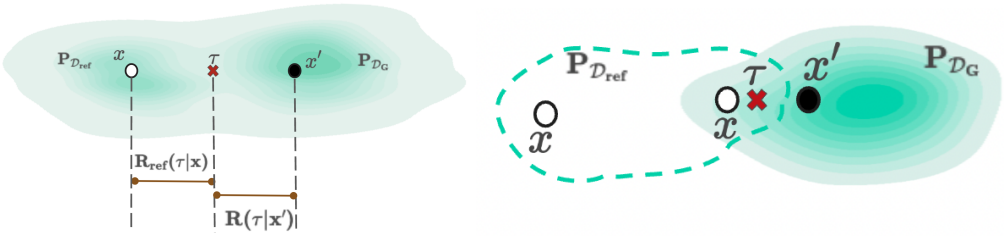
Table 3: Table of Attack Results (Genomic Data)

GAN Variant	Dataset	Attack Method	TPR @0.01 FPR	TPR @ 0.1 FPR	TPR @0.005 FPR	TPR @0.001 FPR
vanilla 805	1000 genome	one-way distance	1.02%	6.95%	0.51%	0.13%
vanilla 805	1000 genome	two-way distance	7.79%	28.86%	5.14%	1.89%
vanilla 805	1000 Genome	weighted distance	4.75%	21.76 %	2.85%	1.76%
vanilla 805	1000 Genome	Robust Homer	2.46%	17.89%	1.26%	0.20%
vanilla 805	1000 Genome	Detector	1.89%	13.72%	0.97%	0.19%
vanilla 805	1000 Genome	ADIS	5.31%	26.89%	3.14%	1.16%
vanilla 5k	1000 genome	one-way distance	0.88%	7.6%	0.37%	0.10%
vanilla 5k	1000 genome	two-way distance	1.36%	13.89%	0.68%	0.25%
vanilla 5k	1000 Genome	weighted distance	1.00%	12.6%	0 .56%	0.15%
vanilla 5k	1000 Genome	Robust Homer	0.88%	7.62%	0.58%	0.25%
vanilla 5k	1000 Genome	Detector	1.52%	12.70%	0.82%	0.33%
vanilla 5k	1000 Genome	ADIS	2.00%	13.40%	0.99%	0.26%
vanilla 10k	1000 genome	one-way distance	1.10%	9.97%	0.62%	0.23%
vanilla 10k	1000 genome	two-way distance	1.07%	10.52%	0.42%	0.17%
vanilla 10k	1000 Genome	weighted distance	0.90%	10.4%	0.49%	0.15%
vanilla 10k	1000 Genome	Robust Homer	1.11%	10.56%	0.58%	0.36%
vanilla 10k	1000 Genome	Detector	1.38%	11.56%	0.68%	0.17%
vanilla 10k	1000 Genome	ADIS	1.72%	12.56%	1.03%	0.25%
wgan-gp 805	dbGaP	one-way distance	0.91%	7.69%	0.54%	0.23%
wgan-gp 805	dbGaP	two-way distance	1.52%	11.91%	0.63%	0.23%
wgan-gp 805	dbGaP	weighted distance	1.41%	11.75%	0.91%	0.38%
wgan-gp 805	dbGaP	Robust Homer	1.37%	11.22%	0.67%	0.25%
wgan-gp 805	dbGaP	Detector	2.59%	16.01%	1.14%	0.22%
wgan-gp 805	dbGaP	ADIS	2.40%	15.15%	1.04%	0.35%
wgan-gp 5k	dbGaP	one-way distance	0.93%	9.17%	0.23%	0.16%
wgan-gp 5k	dbGaP	two-way distance	1.62%	12.81%	0.77%	0.29%
wgan-gp 5k	dbGaP	weighted distance	1.90%	12.90%	1.90%	0.35%
wgan-gp 5k	dbGaP	Robust Homer	2.13%	14.03%	1.48%	0.62%
wgan-gp 5k	dbGaP	Detector	2.95%	20.10%	1.64%	0.37%
wgan-gp 5k	dbGaP	ADIS	6.40%	28.70%	3.98%	1.32%
wgan-gp 10k	dbGaP	one-way distance	1.42%	9.38%	0.55%	0.09%
wgan-gp 10k	dbGaP	two-way distance	2.07%	12.45%	1.20%	0.47%
wgan-gp 10k	dbGaP	weighted distance	1.96%	11.80%	0.98%	0.45%
wgan-gp 10k	dbGaP	Robust Homer	2.71%	13.95%	1.35%	0.65%
wgan-gp 10k	dbGaP	Detector	3.26%	24.23%	2.11%	0.67%
wgan-gp 10k	dbGaP	ADIS	6.07%	24.49%	4.24%	1.6%



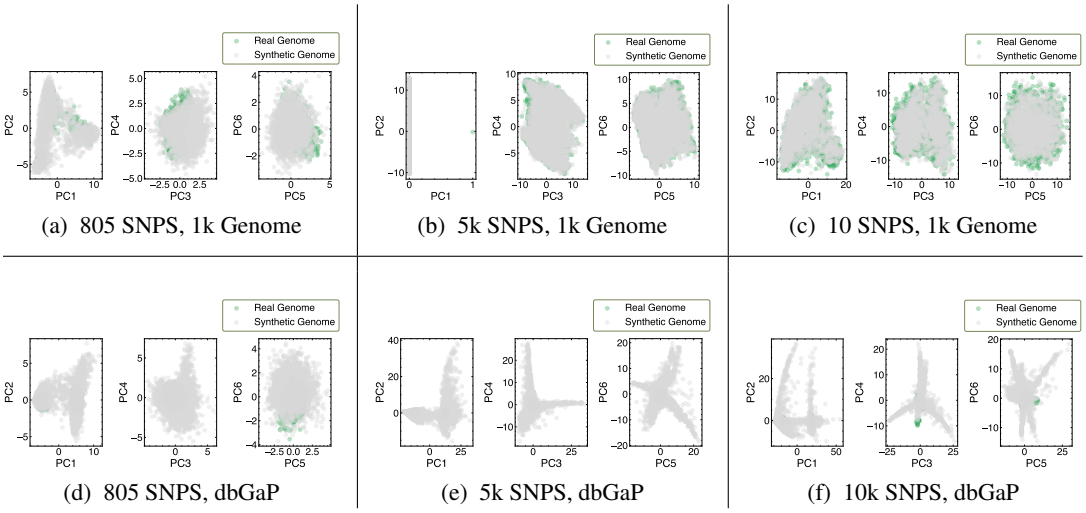
(a) $\mathbf{R}(\tau|\mathbf{x}') < \mathbf{R}_{\mathbf{ref}}(\tau|\mathbf{x})$ (b) $\mathbf{R}_{\mathbf{ref}}(\tau|\mathbf{x}') < \mathbf{R}(\tau|\mathbf{x})$

Figure 4: The diagram illustrates how the distance-based attack can be used to classify a test point τ. $\mathbf{P}_{\mathcal{D}_{\mathbf{ref}}}$ is exactly the reference distribution, $\mathcal{P}$, while $\mathbf{P}_{\mathcal{D}_{\mathbf{G}}}$ is exactly $\mathcal{G}$, the distribution of synthetic samples from GAN. $X_R \sim \mathcal{P}$ and $x \in X_R, x' \in X_G$ where $X_G \sim \mathcal{G}$. For (a), since $\mathbf{R}(\tau|\mathbf{x}') < \mathbf{R}_{\mathbf{ref}}(\tau|\mathbf{x})$, τ is closer to the synthetic samples and is most likely to be categorized as being from the underlying training set- being closer to synthetic samples implies being closer to the training data since synthetic samples are approximations of the training data. For (b), since $\mathbf{R}_{\mathbf{ref}}(\tau|\mathbf{x}) < \mathbf{R}(\tau|\mathbf{x}')$, τ is closer to X_R and more likely to be classified as not being in the training set for the GAN i.e $\tau \in \mathcal{D}_{ref}$.



(a) $R_{ref}(\tau|x) = \mathbf{R}(\tau|\mathbf{x}')$ (b) Only $\mathbf{R}(\tau|\mathbf{x}')$ is known

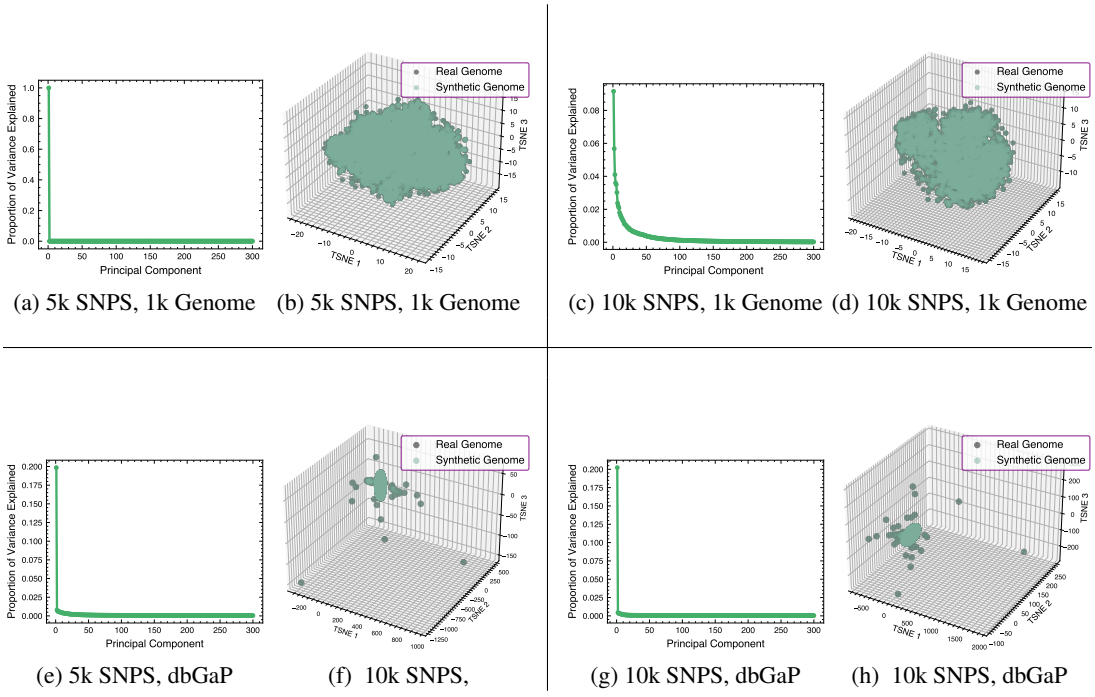
Figure 5: Similar to figure 4,$x \in X_R$, $x' \in X_G$ where X_R, X_G are reference and synthetic samples respectively and τ a test point. Figure (a) above shows the situation where $\mathbf{R}_{\mathbf{ref}}(\tau|\mathbf{x}) = \mathbf{R}(\tau|\mathbf{x}')$, and we can conclude that the test point τ has an equal likelihood to be in $\mathcal{D}$ and $\mathcal{D}_{ref}$, thus the reconstruction loss is inconclusive in this case. Fig (b) depicts the case where we only measure $\mathbf{R}(\tau|\mathbf{x}')$, and disregard $\mathbf{R}_{\mathbf{ref}}(\tau|\mathbf{x})$. In this situation, information regarding how close the test point, τ, is to the reference sample is unknown. The reference sample point, x', might be closer to the test point τ than any synthetic sample or farther; this additional information is lost if we compute only $\mathbf{R}(\tau|\mathbf{x}')$.



(a) 805 SNPS, 1k Genome (b) 5k SNPS, 1k Genome (c) 10 SNPS, 1k Genome

(d) 805 SNPS, dbGaP (e) 5k SNPS, dbGaP (f) 10k SNPS, dbGaP

Figure 6: PCA Plots for all data configurations. The plot could be read row-wise or column-wise. The first row depicts PCA plots for synthetic samples from Vanilla GAN trained on 1000 Genome data for 805, 5k, and 10k SNPS respectively. The second row depicts PCA plots for synthetic samples from WGAN trained on dbGaP data for 805, 5k, and 10k SNPs respectively. Each column corresponds to 805,5k and 10k SNPS configurations.



(a) 5k SNPS, 1k Genome (b) 5k SNPS, 1k Genome (c) 10k SNPS, 1k Genome (d) 10k SNPS, 1k Genome

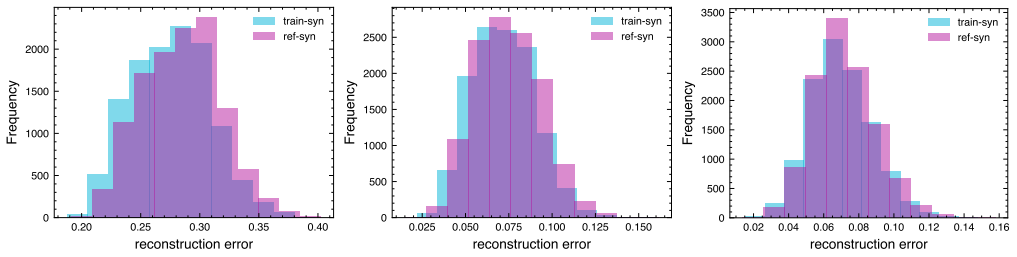
(e) 5k SNPS, dbGaP (f) 10k SNPS, (g) 10k SNPS, dbGaP (h) 10k SNPS, dbGaP

Figure 7: The first row shows scree and TSNE plots for synthetic samples from Vanilla GAN trained on 1000 Genome data for 5k, and 10k SNPS respectively. The second row shows scree and TSNE plots for synthetic samples from WGAN trained on dbGaP data for 5k, and 10k SNPS respectively. The distance metric for the reconstruction loss is the hamming distance.

F.2 Overfitting and Memorization in Genomic GANs

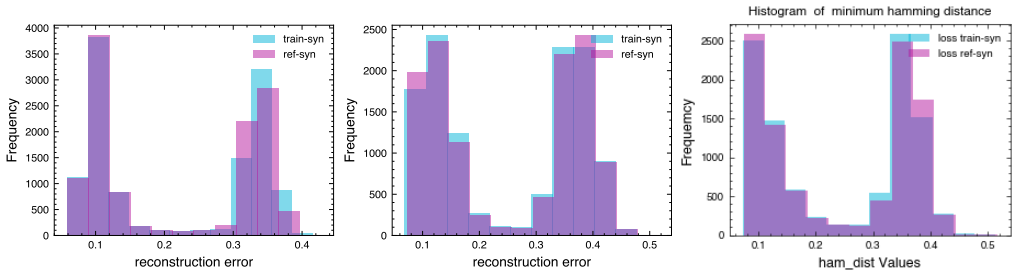
The section is aimed at performing some preliminary analyses on genomic data to ensure it is not overfitting or memorizing the whole training dataset. It is important to check the effect of these factors are kept to the minimum otherwise they could potentially increase the success margin of the attack schemes.

Whole sequence memorization. We iteratively sample batches of 3000-4000 synthetic samples, S_B, from the GAN until the sample size is of order $\geq 10^6$. For each sample point in the i-th batch, $s_b \in S_{B_i}$, we compute its minimum hamming distance to $\mathcal{D}$, the training data. Observe that a hamming distance of zero would indicate whole sequence memorization - since it implies that an exact copy of a training data sample is being synthesized by the GAN. The results of the whole presented in Figures 9 and 8 memorization showed that the GAN models did not memorize the training data, since no reconstruction loss of zero was encountered- a reconstruction loss of zero reflects that an exact training sample was found.



(a) 805 SNPS,1k Genome (b) 5k SNPS,1k Genome (c) 10k SNPS,1k Genome

Figure 8: The plots show the frequency distribution of reconstruction error for whole sequence memorization test for 3 batches of synthetic samples from vanilla GAN trained. The label **train-syn** (in seafoam green) indicates that we are measuring reconstruction loss for each batch of the synthetic samples with respect to the training data. For reference purposes, we also plot, **ref-syn**, which is the reconstruction loss of the synthetic samples given the reference samples (in purple). The distance metric for the reconstruction loss is the hamming distance.



(a) 805 SNPs, dbGaP (b) 5k SNPs, dbGap (c) 10k SNPs, dbGaP

Figure 9: The plots show the frequency distribution of reconstruction error for 3 batches of synthetic samples from WGAN-GP. The label **train-syn** (in seafoam green) indicates that we are measuring reconstruction loss for each batch of the synthetic samples with respect to the training data . For reference purposes, we plot **ref-syn**, which is the reconstruction loss of the synthetic samples given the reference samples (in purple). The distance metric for the reconstruction loss is the hamming distance.

F.2.1 Overfitting.

Overfitting is helpful for a successful MI attack Yeom et al. (2018). To measure and detect the extent to which GAN overfits the underlying training data we compute the normalized *median recovery*

Table 4: Results for test to detect Overfitting

Data Bank	GAN Variant	SNPS Dim.	MRE-gap	§p-value Train	p-value Ref.	*MMD Train	MMD Ref
1000 genome	Vanilla GAN	815	0.0045	0.0001	0.0001	0.0107	0.0137
		5k	0.0300	6.6e-05	6.6e-05	0.0094	0.0105
		10k	0.0108	5e-05	5e-05	0.0139	0.0151
dbGaP	WGAN-GP	815	0.0055	0.0001	0.0001	0.0194	0.0207
		5k	0.0143	6.6e-05	6.6e-05	0.1079	0.1106
		10k	0.002	5e-5	5e-5	0.0989	0.1018

* MMD = Maximum mean discrepancy. The values reported here are the unbiased estimate of MMD^2 Gretton et al. (2012b)
§ p-values computed using kernel 2 sample test.

error gap (MRE-gap) Webster et al. (2019). The median recovery errors (MRE) for the training data X_T, and subsamples of the reference data, $\mathcal{X}_R \sim \mathcal{D}$, are defined as:

$$MRE_G(X_T) = \text{median}\big\{ \min_{s_i \in G} \|x_i - s_i\|^2 \big\}_{x_i \in X_T}$$

$$MRE_G(\mathcal{X}_R) = \text{median}\big\{ \min_{s_i \in G} \|z_i - s_i\|^2 \big\}_{z_i \in \mathcal{X}_{ref}}$$

The normalized MRE-gap$_G$ is then defined as:

$$\text{MRE-gap}_G = \frac{MRE_G(\mathcal{X}_R)) - MRE_G(X_T)}{MRE_G(\mathcal{X}_R))}$$

Two-kernel Test We carried out kernel 2 sample testGretton et al. (2012a) to quantify the similarity between the distribution of the synthetic samples and the distribution of the training data. The result of this test provides valuable insight regarding overfitting.

Specifically, let X_T be of size n and X_S be synthetic samples from the GAN of size $N > n$. Then for each $x \in X_T$, let x_s be the synthetic data sample satisfying the relation:

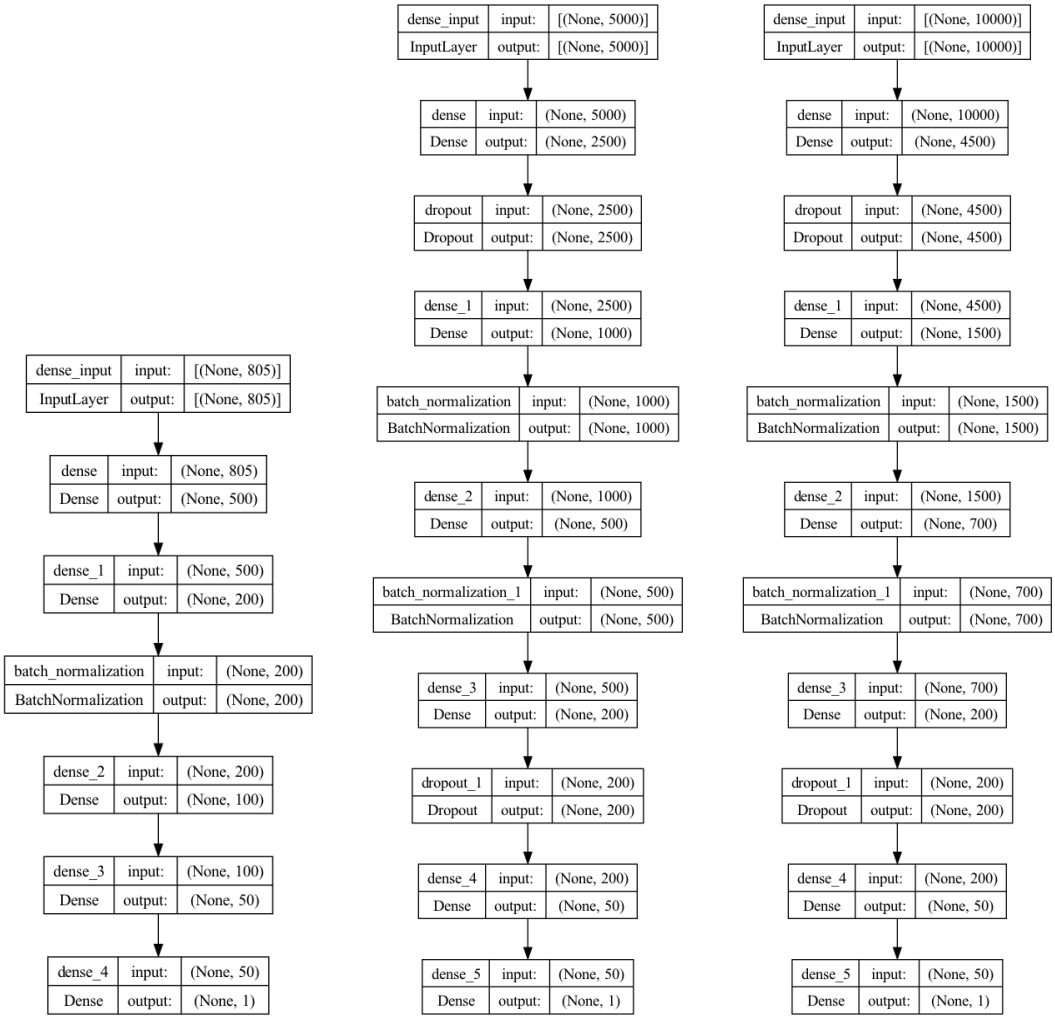
$$\arg\min_{x_s \in X_s} \delta(x, x_s)$$

where δ is the distance metric. Thus x_s is the closest synthetic sample to the training data entry x. Let $\mathcal{S}^*$ contain the x_s for each entry $x \in X_T$. Observe that $\mathcal{S}^*$ and X_T are of the same size, n. We run the two kernel sample tests on the data samples X_T and $\mathcal{S}^*$ as described in Gretton et al. (2012a).

In table 4, we present the normalized MRE-gap$_G$ for different target configurations. The figures seem to support that model seems not to be overfitting . Table 4 also reports the p-values and the unbiased estimate of the squared maximum discrepancy test from the kernel 2 samples test across different target configurations.

F.3 Detector Architecture for Genomic Data Setting

Figure 10 is the Detector model architecture as implemented in Keras Chollet et al. (2015) and Tensorflow Abadi et al. (2015) for the MI attack against genomic data. The exact architecture differs depending on the size of the reference data data that is available to the adversary. The data size of dbGaP is larger than 1KG, thus the Detector architecture for all GANs trained on dbGaP has more layers than the corresponding architecture for target GAN trained on 1KG.



(a) Architecture for 805 SNPs (b) Architecture for 5000 SNPs (c) Architecture for 10000 SNPs

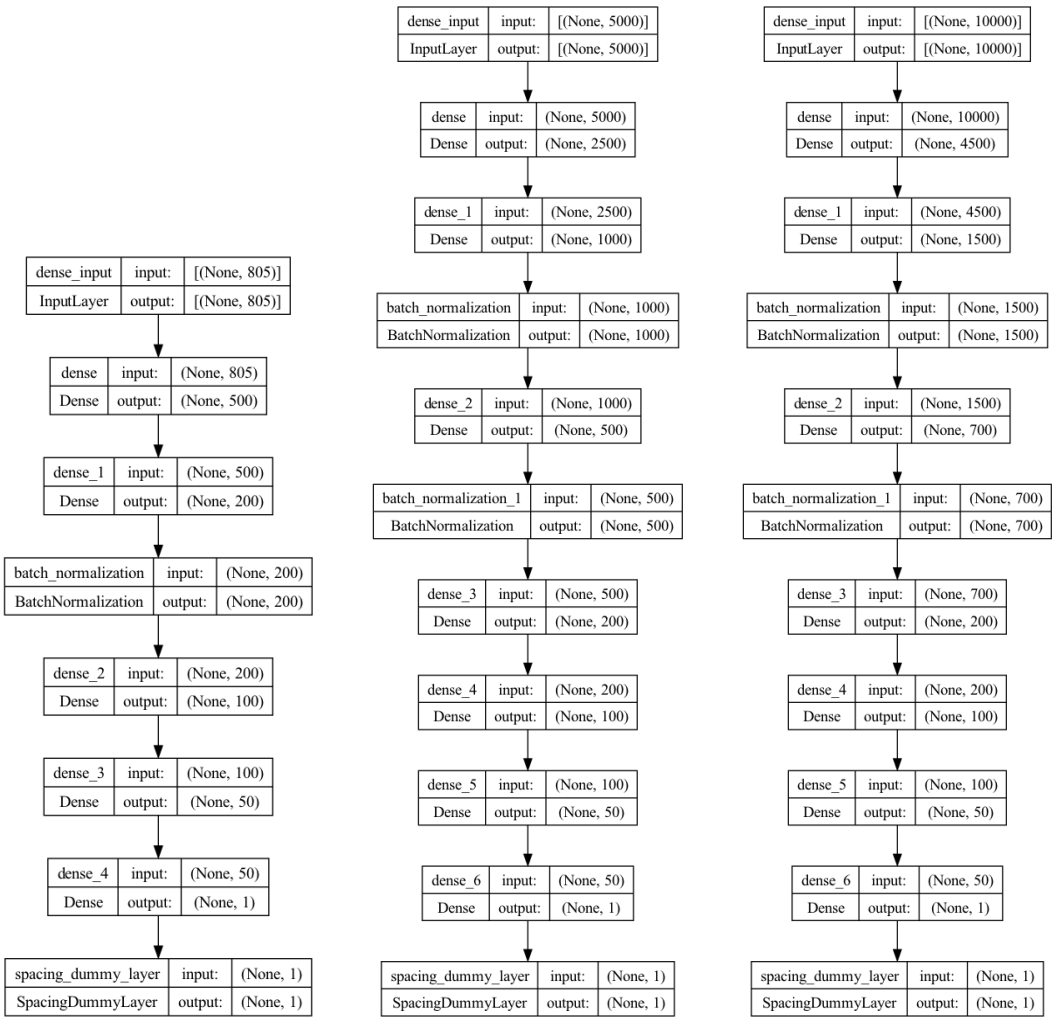
Figure 10: Detector Architecture for MIA against Vanilla GAN trained on 1000 genome database for 805,5000 and 10000 SNPS configurations. The Detector architecture varies depending on the SNPs configurations.

Table 5: Target GAN Configurations for image data

Data	Train Data Size	Ref. Data Size	Test Data Size	GAN Variant
CIFAR10	40000	9000	2000	BigGAN
	40000	9000	2000	Prog. GAN
	40000	9000	2000	DCGAN
	4000	9000	2000	Contra GAN

G RESULTS FOR IMAGE GANS

Distance Metric and Distance-based attack. The choice of metric for the distance-based attack on image GANs requires some careful consideration. Though L2 distance was suggested in (Chen et al., 2020), it should be noted that L2 distance suffers one major limitation when used as a metric



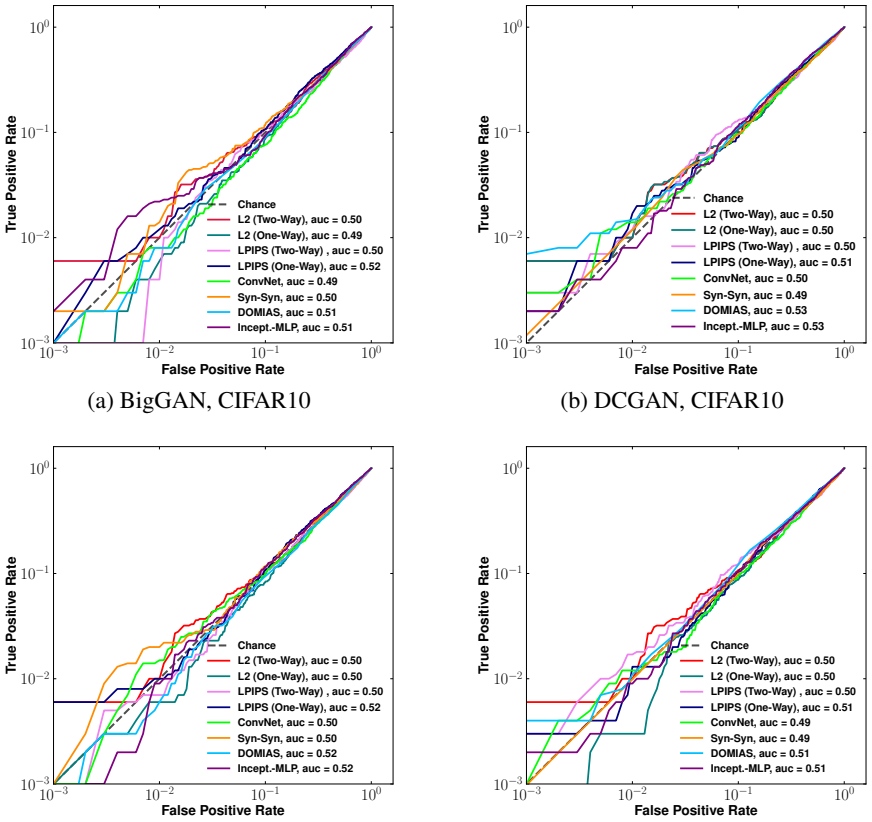
(a) Architecture for 805 SNPs (b) Architecture for 5000 SNPs (c) Architecture for 10000 SNPs

Figure 11: Detector Architecture for MIA against Vanilla GAN trained on 1000 genome database for 805,5000 and 10000 SNP dimension. The Detector architecture varies depending on the SNPs configurations.

for distance-based attack: it can't capture joint image statistics since it uses point-wise difference to measure image similarity (Fu* et al., 2023).

More recently, with the widespread adoption of deep neural networks, learning-based metrics have enjoyed wide popularity and acceptance (Dosovitskiy & Brox, 2016; Gatys et al., 2015; Johnson et al., 2016). These metrics leverage pre-trained deep neural networks to extract features and use these features as the basis for metric computation (Fu* et al., 2023). LPIPs (Zhang et al., 2018) (Learned Perceptual Image Patch Similarity) is the most common of such learning-based metrics and was also proposed in (Chen et al., 2020) for carrying out distance-based attacks.

Our distance-based attack was conducted using both L2 distance and LPIPs. Both attack schemes were not very successful, though the LPIPS seemed to be slightly better than L2 for the distance attack.



(a) BigGAN, CIFAR10

(b) DCGAN, CIFAR10

(c) Progressive GAN, CIFAR10

(d) ContraGAN, CiFAR10

Figure 12: The figures show the distance-based attack and the detector-based attacks against image GANs. Observe that the distance based attack are not particularly effective.

H Additional Related Work

Hilprecht et al. (2019) propose a black-box attack on GANs trained on CIFAR-10 that is very similar to a distance-based attack. Their statistic samples from the GAN and counts the proportion of generated samples that fall within a given distance ϵ of the candidate point. In order to set ϵ, they estimate the 1% or $.1\%$ quantiles of distances to the generated GAN via sampling. Note that this assumes access to a set of candidate points x_i rather than a single candidate point that may or may not be a training point, although a similar idea could be implemented using reference data. While their attacks were quite effective against VAEs and on the simpler MNIST dataset, the best accuracy one of their distance-based attacks achieves on CIFAR-10 is a TPR that is barely above 50%, at a massive

Table 6: Table of Attack Results (Image Data)

GAN Variant	Dataset	Attack Method	TPR @0.01 FPR	TPR @0.1 FPR	TPR @0.005 FPR	TPR @0.001 FPR
BigGAN	CIFAR10	two-way distance (L2)	1.0%	10.8%	0.6%	0.6%
BigGAN	CIFAR10	one-way distance (L2)	0.6%	8.7%	0.2%	0.0%
BigGAN	CIFAR10	two-way distance (LPIPS)	0.4%	11.0%	0.1%	0.1%
BigGAN	CIFAR10	one-way distance (LPIPS)	1.2%	10.8%	0.7%	0.1%
BigGAN	CIFAR10	Detector (Conv. net)	0.8%	7.7%	0.3%	0.0%
BigGAN	CIFAR10	Detector (syn-syn)	1.4%	11.8%	0.7%	0.2%
BigGAN	CIFAR10	Detector(Incept.-MLP)	2.2%	9.3%	1.6%	0.2%
DCGAN	CIFAR10	two-way distance (L2)	1.0%	10.8%	0.6%	0.6%
DCGAN	CIFAR10	one-way distance (L2)	1.0%	11.1%	0.6%	0.6%
DCGAN	CIFAR10	two-way distance (LPIPS)	1.2%	13.0%	0.7%	0.3%
DCGAN	CIFAR10	one-way distance (LPIPS)	1.7%	9.0%	0.6%	0.2%
DCGAN	CIFAR10	Detector (Conv. net)	1.4%	9.7%	1.1%	0.3%
DCGAN	CIFAR10	Detector (syn-syn)	1.2%	9.6%	0.6%	0.1%
DCGAN	CIFAR10	Detector (Incept.-MLP)	0.8%	11.4%	0.4%	0.2%
ProjGAN	CIFAR10	two-way distance (L2)	1.0%	10.7%	0.6%	0.6%
ProjGAN	CIFAR10	one-way distance (L2)	0.6%	8.4%	0.3%	0.1%
ProjGAN	CIFAR10	two-way distance (LPIPS)	0.7%	10.7%	0.6%	0.1%
ProjGAN	CIFAR10	one-way distance (LPIPS)	1.0%	10.9%	0.8%	0.6%
ProjGAN	CIFAR10	Detector (Conv. net)	1.5%	9.9%	0.7%	0.1%
ProjGAN	CIFAR10	Detector (syn-syn)	2.0%	11.8%	1.4%	0.1%
ProjGAN	CIFAR10	Detector (Incept.-MLP)	0.9%	11.6%	0.2%	0.0%
ContraGAN	CIFAR10	two-way distance (L2)	1.0%	10.8%	0.6%	0.6%
ContraGAN	CIFAR10	one-way distance (L2)	0.3%	8.7%	0.3%	0.0%
ContraGAN	CIFAR10	two-way distance (LPIPS)	1.7%	12.3%	1.0%	0.3%
ContraGAN	CIFAR10	one-way distance (LPIPS)	1.3%	10.4%	0.4%	0.3%
ContraGAN	CIFAR10	Detector (Conv. net)	1.2%	9.6%	0.7%	0.1%
ContraGAN	CIFAR10	Detector (syn-syn)	0.9%	10.0%	0.5%	0.01%
ContraGAN	CIFAR10	Detector (Incept.-MLP)	0.9%	10.4%	0.3%	0.2%

FPR that is also close to 50% (Figure 4(c)) with their other distance-based method performing worse than random guessing. Relatedly, (Chen et al., 2020) proposed a distance-based attack scheme based on minimum distance to a test sample. In Chen et al. (2020) the adversary has access to synthetic samples from the target GAN and synthetic samples from a GAN trained on reference data (G_{ref}). We follow this approach for distance-based attacks, which we discuss further in Section 4.2. The most closely related work to our detector methods is (van Breugel et al., 2023b) who propose a density-based model called DOMIAS (Detecting Overfitting for Membership Inference Attacks against Synthetic Data), which infers membership by targeting local overfitting of the generative models. Rather than train a detector network to classify whether samples are generated from the target GAN or the reference data, DOMIAS performs dimension reduction in order to directly estimate both densities, and then uses the ratio of the densities as a statistic for membership inference.

Related to our study of tabular GANs trained on genomic data, (Yelmen et al., 2021) proposed the use of GANs for the synthesis of realistic artificial genomes with the promise of none to little privacy loss. The absence of privacy loss in the proposed model was investigated by measuring the extent of overfitting using the nearest neighbor adversarial accuracy (AAT_{TS}) and privacy score metrics discussed in (Yale et al., 2019). It should be noted that while overfitting is sufficient to allow an adversary to perform MI (and hence constitute a privacy loss), it is not a necessary prerequisite for the attack to succeed, as shown in the formal analysis presented in (Yeom et al., 2018).