A. Connection between Lipschitz Constant and Reconstruction Error

For the sake of completeness, we present the derivation of Eq. (1) from Appendix C in [30]. We consider the following two forward mappings:

$$\begin{aligned} \mathbf{z} &= f(\mathbf{x}), \\ \mathbf{z}_\delta &= f_\delta(\mathbf{x}) := \mathbf{z} + \delta_\mathbf{z}, \end{aligned}$$

where f is an analytically exact computation, whereas f_δ is a practical floating-point inexact computation. Similarly, we consider the following two inverse mappings:

$$\begin{aligned} \mathbf{x}_{\delta_1} &= f^{-1}(\mathbf{z}_\delta), \\ \mathbf{x}_{\delta_2} &= f_\delta^{-1}(\mathbf{z}_\delta) := \mathbf{x}_{\delta_1} + \delta_\mathbf{x}. \end{aligned}$$

The $\delta_\mathbf{z}$ and $\delta_\mathbf{x}$ are numerical errors in the mapping through f_δ and f_δ^{-1}. By the definition of the Lipschitz continuous, we obtain the following:

$$\frac{\left\| f^{-1}(\mathbf{z}) - f^{-1}(\mathbf{z}_\delta) \right\|}{\|\mathbf{z} - \mathbf{z}_\delta\|} = \frac{\|\mathbf{x} - \mathbf{x}_{\delta_1}\|}{\|\mathbf{z} - \mathbf{z}_\delta\|} \leq \mathrm{Lip}(f^{-1})$$

$$\Leftrightarrow \|\mathbf{x} - \mathbf{x}_{\delta_1}\| = \mathrm{Lip}(f^{-1}) \|\mathbf{z} - \mathbf{z}_\delta\| = \mathrm{Lip}(f^{-1}) \|\delta_\mathbf{z}\| .$$

We now consider the reconstruction error with f_δ and f_δ^{-1}:

$$\begin{aligned} \left\| \mathbf{x} - f_\delta^{-1}(f_\delta(\mathbf{x})) \right\| &= \|\mathbf{x} - \mathbf{x}_{\delta_2}\| \\ &= \|\mathbf{x} - (\mathbf{x}_{\delta_1} + \delta_\mathbf{x})\| \\ &\leq \|\mathbf{x} - \mathbf{x}_{\delta_1}\| + \|\delta_\mathbf{x}\| \\ &= \mathrm{Lip}(f^{-1}) \|\delta_\mathbf{z}\| + \|\delta_\mathbf{x}\| . \end{aligned}$$

While the reconstruction error is zero in the ideal case using f and f^{-1}, the upper bound on the reconstruction error is given above when we consider f_δ and f_δ^{-1} to account for the numeral error caused by the large Lipschitz coefficients in practice.

B. Controlled Experiments of Penalty in Latent Space.

We show experimentally that penalty ξ in the latent space increases the reconstruction error. Letting $R'(\mathbf{x}) = \left\| \mathbf{x} - f^{-1}(\hat{\mathbf{z}}) \right\|$ and $\hat{\mathbf{z}} = \mathbf{z} + \xi' \frac{\mathbf{z}}{\|\mathbf{z}\|}$ with $\mathbf{z} = f(\mathbf{x})$, we measured $R'(\mathbf{x})$ for synthetically generated ξ' in the range of $(-11, 11)$ with 0.2 increments. When $\xi' > 0$, $\hat{\mathbf{z}}$ is shifted further away from the origin in the latent space than $\mathbf{z}$. Conversely, when $\xi' < 0$, $\hat{\mathbf{z}}$ is shifted closer to the origin. We used the Glow model trained on the CIFAR-10's training dataset and tested 1024 samples randomly selected from the test dataset of CIFAR-10, which thus corresponds to the In-Dist examples. As shown in Fig. 5 (left), the degree of the reconstruction error is proportional to the intensity of the penalty, regardless of the sign of ξ'. As a case of OOD, we also measured the same for the test samples of Celeb A, and similar results were obtained (Fig. 5 (right)).

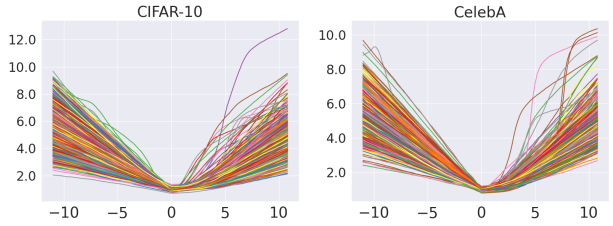

Figure 5: The effects of penalty in latent space. The x-axis is the intensity of penalty (ξ') and the y-axis is penalized reconstruction error (R'). Each line corresponds to each example. For both figures, In-Dist samples used to train the Glow model are from CIFAR-10. The OOD samples are from CIFAR-10's test dataset on the left figure and are from CelebA on the right figure.

C. Experimental Setup

C.1. OOD datasets

C.1.1 Different datasets from In-Dist

We use CelebA [66], TinyImageNet (TIN) [32], and LSUN [67]as OOD datasets. The CelebA dataset contains various face images, from each of which we cut out a 148×148 centered area and re-scale them to the size of each In-Dist dataset. The TIN is also used as OOD only when the In-Dist is C-10, re-scaling its images to 32×32. For LSUN, we cut out squares with a length that matches the shorter of the height or width of the original image, followed by re-scaling to 32×32 for C-10 and 64×64 for TIN.

C.1.2 Noise images

We control the noise complexity by following [15]. Specifically, noise images with the same size as In-Dist are sampled from uniform random, average-pooling is applied to them, and then they are resized back to the original size. The complexity is controlled by the pooling size, κ. The images become most complex when $\kappa = 1$ (i.e., no pooling) and become simpler as κ is increased. Treating images with different κ as separate datasets, we refer to them as Noise-κ.

C.2. Classifiers

The archtecture of classifier we used for the experiments on CIFAR-10 and TinyImageNet is WideResNet 28-10 (WRN 28-10) [72], and the one on ILSVRC2012 is ResNet-50 v2 [73]. These classifiers are used for generating adversarial examples and also for the classifier-based comparison methods as described in Section 5.2. The classification accuracies are 95.949 %, 66.450 %, and 67.628 % on the test datasets of C-10, TIN, ILSVRC2012, respectively (Table 4).

Table 4: Classification accuracies (%) for adversarial examples. Results for *no attack* are evaluated with the whole test dataset for each. Results for attacks are evaluated with 1024 adversarial examples we generated based on 1024 normal samples chosen at random from each test dataset.

Dataset	Classifier	no attack	PGD-2	PGD-8	CW-0	CW-10
CIFAR-10	WRN-28-10	95.949	5.56	0.0	0.0	0.0
TinyImageNet	WRN-28-10	66.450	2.92	1.95	0.0	0.0
ILSVRC2012	ResNet-50 v2	67.628	0.0	0.0	0.0	0.0

C.3. Adversarial Examples

C.3.1 Attack methods

Let $C(\cdot)$ be a DNN classifier where $C_i(\mathbf{x})$ denotes logit of class label i for an image $\mathbf{x} \in [0,1]^d$, represented in the d dimension normalized [0,1] space. The predicted label is given as $y_{\text{pred}} = \arg\max_i C_i(\mathbf{x})$. Using a targeted classifier C, we mount untargeted attacks with the two methods, that is, we attempt to generate adversarial examples $\mathbf{x}_{\text{adv}}$ to be labeled as $y_{\text{target}} = \arg\max_{i \neq y_{\text{true}}} C_i(\mathbf{x})$, where y_{true} is the original label for $\mathbf{x}$. In PGD, $\mathbf{x}_{\text{adv}}$ is searched within a hypercube around $\mathbf{x}$ with an edge length of 2ϵ, which is written as

$$\mathbf{x}_{\text{adv}} = \min_{\mathbf{x}^*} L_{\text{pgd}}(\mathbf{x}^*) \text{ s.t. } \|\mathbf{x}^* - \mathbf{x}\|_\infty \leq \epsilon \tag{7}$$

where

$$L_{\text{pgd}}(\mathbf{x}^*) := C_{y_{\text{true}}}(\mathbf{x}^*) - C_{y_{\text{target}}}(\mathbf{x}^*) \tag{8}$$

and ϵ is given as a hyper-parameter. The CW attack is formalized as

$$\mathbf{x}_{\text{adv}} = \min_{\mathbf{x}^*} \lambda L_{\text{cw}}(\mathbf{x}^*) + \|\mathbf{x}^* - \mathbf{x}\|_2^2 \tag{9}$$

where

$$L_{\text{cw}}(\mathbf{x}^*) := \max(C_{y_{\text{target}}}(\mathbf{x}^*) - C_{y_{\text{true}}}(\mathbf{x}^*), -k), \tag{10}$$

k is a hyper-parameter called *confidence*, and λ is a learnable coefficient. Defining a vector of adversarial perturbation $\Delta_{\mathbf{x}} := \mathbf{x}_{\text{adv}} - \mathbf{x}$, $\|\Delta_{\mathbf{x}}\|_\infty$ in PGD is bounded by ϵ, whereas $\|\Delta_{\mathbf{x}}\|_2$ in CW is not. Thus, while the artifacts may appear on images when k becomes larger, CW attack always reaches 100% successes in principle.

C.3.2 Generation of adversarial examples

For PGD attacks, the step size is $\frac{1}{255}$ and the mini-batch size is 128 on all three datasets. The numbers of projection iteration for PGD-2 are 1000 on C-10 and TIN and 100 on ILSVRC2012, and those for PGD-8 are 100 on all three datasets. For CW attacks, our code is based on the one used in [70]. The maximum numbers of iterations are 10000 on C-10 and 1000 on TIN and ILSVRC2012. The number of times we perform binary search to find the optimal tradeoff-constant between L_2 distance and confidence is set to 10. The initial tradeoff-constant to use to tune the relative importance of L_2 distance and confidence is set to 1e-3. The learning rate for attacking is 1e-2 and the mini-batch size is 256 on C-10, 128 on TIN, and 64 on ILSVRC2012. The classification accuracies before and after attacking are shown in Table 4.

C.4. Glow

Architecture. The architecture of the Glow mainly consists of two parameters: the depth of flow K and the number of levels L. The set of affine coupling and 1×1 convolution are performed K times, followed by the factoring-out operation [25], and this sequence is repeated L times. We set $K = 32$ and $L = 3$ for C-10, $K = 48$ and $L = 4$ for TIN, and $K = 64$ and $L = 5$ for ILSVRC2012. Refer to [74, 25] and our experimental code for more details.[4] Many flow models including Glow employs *affine coupling layer* [25] to implement f_i. Splitting the input $\mathbf{x}$ into two halves as $[\mathbf{x}_a, \mathbf{x}_b] = \mathbf{x}$, a coupling is performed as $[\mathbf{h}_a, \mathbf{h}_b] = f_i(\mathbf{x}) = [\mathbf{x}_a, \mathbf{x}_b \odot s(\mathbf{x}_a) + t(\mathbf{x}_a)]$ where $\odot$ denotes the element-wise product and s and t are the convolutional neural networks optimized through the training. Its inverse can be easily computed as $\mathbf{x}_a = \mathbf{h}_a$ and $\mathbf{x}_b = (\mathbf{h}_b - t(\mathbf{x}_a)) \oslash s(\mathbf{x}_a)$, where $\oslash$ denotes the element-wise division, and $\log |\det J_{f_i}(\mathbf{x})|$ can be obtained as just $\log |s(\mathbf{x}_a)|$.

Restricted affine scaling. In order to suppress the Lipschitz constants of the transformations and further improve stability, we restrict the affine scaling to $(0.5, 1)$, following [30]. Specifically, we replace $s(\mathbf{x}_a)$ with $g(s(\mathbf{x}_a))$ and use a half-sigmoid function as $g(s) = \sigma(s)/2 + 0.5$ where σ is the sigmoid function.

Training settings. We trained 45100 iterations for C-10 and TIN and 70100 iterations for ImaneNet, using the Adam

[4] We implemented Glow model based on [75].

optimizer [76] with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. The batch size is 256 for C-1, 16 for TIN, and 4 for ILSVRC2012. The learning rate for the first 5100 iterations is set to 1e-4, and 5e-5 for the rest. We applied the data augmentation of random 2×2 translation and horizontal flip on C-10 and TIN. For ILSVRC2012, the image is first resized to be 256 in height or width, whichever is shorter, and cropped to 224×224 at random.

C.5. Existing Methods

We compare the performance of the proposed methods to several existing methods. In the following description for the likelihood-based methods (i.e., WAIC, LLR, and COMP), $p(\mathbf{x})$ denotes the likelihood computed with the Glow model, the same as the one we use for our methods. We reverse the sign of score outputted from those likelihood-based methods since we use them as scores indicating being OOD examples. For the classifier-based methods (i.e., DU, FS, and PL), the classifier ($C(\mathbf{x})$) is the WRN 28-10 or ResNet-50 v2 which is the same model we used to craft the adversarial examples in the previous section. The comparison methods are as follows:

1. The Watanabe-Akaike Information Criterion (WAIC) [13] measures $\mathbb{E}[\log p(\mathbf{x})] - \mathrm{Var}[\log p(\mathbf{x})]$ with an ensemble of five Glow models. The four of the five were trained separately with different affine-scaling restrictions mentioned in Section 5.2, i.e., the function $g(s)$ described in Appendix C.4. The function $g(s)$ we chose for the four models is the following: a sigmoid function $\sigma(s)$, a half-sigmoid function $\sigma(s)/2 + 0.5$, clipping as $\mathtt{min}(|s|, 15)$, and an additive conversion as $g(s) = 1$. In addition to those four models, we used the background model used in the LLR described next as a component of the ensemble.

2. The likelihood-ratio test (LLR) [14] measures $\log p(\mathbf{x}) - \log p_0(\mathbf{x})$. The $p_0(\mathbf{x})$ is a background model trained using the training data with additional random noise sampled from the Bernoulli distribution with a probability 0.15. It thus uses two Glow models.

3. The Complexity-aware likelihood test (COMP) [15] measures $\log p(\mathbf{x}) + \frac{|B(\mathbf{x})|}{d}$ where $B(\mathbf{x})$ is a lossless compressor that outputs a bit string of $\mathbf{x}$ and its length $|B(\mathbf{x})|$ is normalized by d, the dimensionality of $\mathbf{x}$. We use a PNG compressor as B.

4. The typicality test in latent space (TTL) [13] measures the Euclidean distance in the latent space to the closest point on the Gaussian Annulus: $\mathtt{abs}(\|f(\mathbf{x})\| - \sqrt{d})$ where f is the same Glow model used in our method.

5. The Maximum Softmax Probability (MSP) [1] simply measures $\mathtt{max}(\mathtt{softmax}(\mathtt{C}(\mathbf{x}))$, which has been often used as a baseline method in the previous OOD detection works.

6. The Dropout Uncertainty (DU) [47] is measured by Monte Carlo Dropout [77], i.e., the sum of the variance of each component of $\mathtt{softmax}(C(\mathbf{x}))$, computed over 30 times run with the dropout rate 0.2.

7. The Feature Squeezing (FS) [48] outputs L_1 norm distance as the score: $\|\mathtt{softmax}(C(\mathbf{x})) - \mathtt{softmax}(C(\hat{\mathbf{x}}))\|_1$ where $\hat{\mathbf{x}}$ is generated by applying a median filter to the original input $\mathbf{x}$.

8. The Pairwise Log-Odds (PL) [49] measures how log-its change under random noise. Defining perturbed log-odds between classes i and j as $G_{ij}(\mathbf{x}, \epsilon) := C_{ij}(\mathbf{x} + \epsilon) - C_{ij}(\mathbf{x})$ where $C_{ij}(\mathbf{x}) := C_j(\mathbf{x}) - C_i(\mathbf{x})$, the score is defined as $\mathtt{max}_{\mathtt{i} \neq \mathtt{y}_{\text{pred}}} \mathbb{E}[\mathtt{G}_{\mathtt{i}\mathtt{y}_{\text{pred}}}(\mathbf{x}, \epsilon)]$ where $y_{\text{pred}} = \arg\max_i C_i(\mathbf{x})$. The noise ϵ is sampled from $\mathrm{Uniform}(0, 1)$, and we took the expectation over 30 times run.

9. The reconstruction error in Auto-Encoder is defined as $\mathrm{AE}(\mathbf{x}) = \|\mathbf{x} - f_{\mathrm{d}}(f_{\mathrm{e}}(\mathbf{x}))\|_2$ where f_{e} and f_{d} are the encoder and decoder networks, respectively. The architecture of the AE model we used and its training procedure is presented in Appendix C.5.1.

C.5.1 Auto-Encoder

We use the Auto-Encoder based method as one of the comparing methods in our experiments on CIFAR-10 and Tiny-ImageNet. The architecture of the Auto-Encoder we used is designed based on DCGAN [78], which is shown in Table 5. For both datasets, we used the same architecture and applied the following settings. We used the Adam optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$, with batch size 256. The learning rate starts with 0.01 and exponentially decays with rate 0.8 at every 2 epochs, and we trained for 300 epochs. The same data augmentation as used for Glow was applied.

D. More Experimental Results

D.1. AUPR

We show the detection performance as the area under the precision-recall curve (AUPR). Table 6 shows the results on CIFAR-10, Table 7 shows the results on TinyImageNet, and Table 8 shows the results on ILSVRC2012.

D.2. Effects of Coefficient of Penalty

Table 9 shows the AUROC with different λ in Eq. (4), the coefficients of ξ. We empirically chose $\lambda = 50$ for CIFAR-10 and $\lambda = 100$ for the other two datasets based on these results.

Table 5: Architecture of encoder and decoder of Auto-Encoder. BNorm stands for batch normalization. Slopes of Leaky ReLU (lReLU) are set to 0.1.

Encoder	Decoder
Input: $32 \times 32 \times 3$ image	Input: 256-dimensional vector
3×3 conv. 128 same padding, BNorm, ReLU	Fully connected $256 \rightarrow 512$ ($4 \times 4 \times 32$), lReLU
3×3 conv. 256 same padding, BNorm, ReLU	3×3 deconv. 512 same padding, BNorm, ReLU
3×3 conv. 512 same padding, BNorm, ReLU	3×3 deconv. 256 same padding, BNorm, ReLU
Fully connected $8192 \rightarrow 256$	3×3 deconv. 128 same padding, BNorm, ReLU
	1×1 conv. 128 valid padding, sigmoid

Table 6: AUPR (%) on CIFAR-10. The column labeled as 'Avg.' shows the averaged scores.

	CelebA	TIN	Bed	Living	Tower	PGD-2	PGD-8	CW-0	CW-10	Noise-1	Noise-2	Avg.
WAIC	50.24	74.29	80.70	86.54	83.69	43.88	70.07	48.26	47.20	**100.0**	**100.0**	71.35
LLR	62.59	43.00	44.81	43.81	39.83	57.83	78.54	51.87	54.06	30.82	30.71	48.90
COMP	75.37	74.88	73.32	84.95	58.80	61.18	98.49	50.95	53.68	**100.0**	**100.0**	75.60
TTL	82.49	85.58	90.68	91.60	89.88	73.82	**99.99**	50.57	53.90	**100.0**	51.04	79.05
MSP	78.16	88.74	**94.58**	92.19	**92.74**	35.79	30.69	**99.15**	31.63	98.82	97.40	76.35
PL	76.84	57.80	51.59	47.01	51.40	79.88	96.98	53.00	81.00	35.53	65.84	63.35
FS	76.96	82.63	80.41	80.22	78.74	**91.51**	75.31	83.80	**94.84**	81.69	90.70	83.37
DU	76.86	75.43	75.79	74.42	74.78	70.52	36.20	85.63	76.44	69.36	65.49	70.99
AE	59.61	81.63	70.36	84.71	66.37	50.37	56.71	50.04	49.96	98.74	**100.0**	69.86
RE (ours)	86.06	87.27	88.33	87.99	87.98	84.23	87.44	90.05	89.89	90.78	89.16	88.11
PRE (ours)	**88.42**	**92.68**	92.12	**92.23**	92.32	84.78	**99.99**	89.98	90.09	**100.0**	88.91	**91.96**

D.3. Histograms of L_2 norm for partitioned latent vector

The typicality test (TTL) failed to detect CW's adversarial examples (and some Noise datasets), while the PRE and RE successfully did, as shown in Tables 1 and 2. The failure of TTL is obvious from Fig. 4 showing that the distributions of $\|\mathbf{z}\|$ for the CW's examples (the adversarial examples generated by CW-0) overlap those for In-Dist's almost entirely. We analyzed this failure by the following procedure. We partitioned latent vectors $\mathbf{z} \in \mathbb{R}^{3072}$ on CIFAR-10 into two parts as $[\mathbf{z}_a, \mathbf{z}_b] = \mathbf{z}$ where $\mathbf{z}_a \in \mathbb{R}^{2688}$ and $\mathbf{z}_b \in \mathbb{R}^{384}$, and measuring the L_2 norm separately for each. [5] In Fig. 6, we show the distributions comparing the CW-0's example with the In-Dist's in $\|\mathbf{z}_a\|$ and $\|\mathbf{z}_b\|$. The distribution of $\|\mathbf{z}_a\|$ for the CW-0's examples is shifted toward larger values than that for In-Dist's. In contrast, the opposite is true for $\|\mathbf{z}_b\|$, where the distribution for the CW-0's examples is slightly shifted toward smaller values than that for In-Dist's. When we measure the L_2 norm for the entire dimension of $\mathbf{z}$ without partitioning, the deviation from the In-Dist observed in $\mathbf{z}_a$ and $\mathbf{z}_b$ cancels out, and as a result, $\|\mathbf{z}\|$ of CW's examples becomes indistinguishable from those of In-Dist ones. This is exactly the case described in Section 3.2 where the typicality test fails.

[5] This partitioning is based on the factoring-out operation [25] used in affine coupling-type NFs, including Glow. The inputs with 3072 dimension are partitioned this way when factoring-out is applied three times.

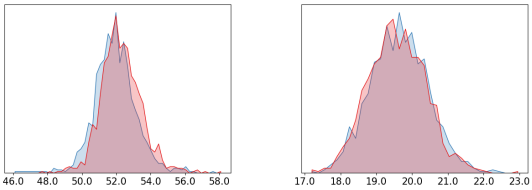

Figure 6: Histograms for L_2 norm for $\mathbf{z}_a \in \mathbb{R}^{2688}$ (left) and $\mathbf{z}_b \in \mathbb{R}^{384}$ (right). Red is the distribution for the OOD: CW-0's examples on CIFAR-10, and blue is In-Dist's examples.

D.4. Histograms for Reconstruction Error without Penalty, R

Fig. 7 shows the histograms of reconstruction error without penalty in the latent space, R in correspondence with Fig. 3.

E. Analysis with Chernoff Tail Bound

Let $\mathbf{z} \in \mathbb{R}^d$ be a random vector sampled from a standard Gaussian. Then, the following is obtained from Markov's

Table 7: AUPR (%) on TinyImageNet. The column labeled as 'Avg.' shows the averaged scores.

	CelebA	Bed	Living	Tower	PGD-2	PGD-8	CW-0	CW-10	Noise-1	Noise-2	Avg.
WAIC	32.54	59.41	63.02	63.81	42.89	44.77	48.84	47.00	**100.0**	**100.0**	60.23
LLR	90.81	58.27	58.43	70.46	56.81	94.75	51.35	53.37	70.30	30.69	63.52
COMP	50.31	50.92	62.30	46.43	56.49	94.86	50.75	52.01	**100.0**	**100.0**	66.41
TTL	**96.27**	**99.03**	**99.45**	**98.90**	78.61	**100.0**	50.96	55.19	**100.0**	**100.0**	87.84
MSP	79.60	81.29	77.42	76.76	31.44	30.69	76.13	30.93	79.53	83.20	64.70
PL	46.71	34.51	39.27	38.11	**93.26**	99.99	41.59	**95.95**	33.40	39.33	56.21
FS	36.83	36.23	36.90	35.77	78.27	39.19	46.41	86.10	41.45	45.52	48.27
DU	44.73	41.07	38.91	37.41	50.20	68.04	48.88	49.82	44.56	34.93	45.86
AE	33.24	34.50	38.11	35.24	49.64	50.63	49.98	49.95	85.97	56.33	48.86
RE (ours)	46.42	56.06	56.28	57.51	87.98	88.71	**92.04**	93.10	94.12	93.81	76.60
PRE (ours)	70.62	92.51	94.96	91.81	88.41	99.90	91.83	92.90	**100.0**	**100.0**	**92.29**

Table 8: AUPR (%) on ILSVRC2012 with our method. The column labeled as 'Avg.' shows the averaged scores.

	CelebA	PGD-2	PGD-8	CW-0	CW-10	Noise-2	Noise-32	Avg.
RE	**90.04**	88.59	90.35	**90.52**	91.08	**91.42**	91.45	90.49
PRE	**90.04**	**88.60**	**90.47**	**90.52**	**91.09**	**91.42**	**91.46**	**90.51**

Table 9: AUROC (%) with various coefficient of penalty, λ.

In-Dist	CIFAR-10		TinyImageNet		ILSVRC2012	
λ / OOD	CelebA	PGD-2	CelebA	PGD-2	CelebA	PGD-2
0	92.53	91.66	46.68	92.86	94.65	93.96
10	93.40	**92.95**	79.99	94.05	94.64	93.96
50	**93.62**	92.23	93.13	**95.13**	94.56	93.93
100	93.60	92.09	95.55	95.04	**94.89**	**94.24**
500	91.94	89.21	**98.41**	92.53	93.02	92.36
1000	90.81	85.08	98.33	90.83	88.92	89.30

inequality:

$$\Pr\left[\|\mathbf{z}\|_2 > \sqrt{d(1+\epsilon)}\right] \leq \exp\left(-\frac{d\epsilon^2}{8}\right), \tag{11}$$

$$\Pr\left[\|\mathbf{z}\|_2 < \sqrt{d(1-\epsilon)}\right] \leq \exp\left(-\frac{d\epsilon^2}{8}\right). \tag{12}$$

We refer the reader to [79], for example, for the derivation. From the above, we obtain (6). Letting $\exp\left(-\frac{d\epsilon^2}{8}\right) = \frac{1}{2^s}$, we have $\epsilon = \sqrt{\frac{8s}{d\log_2(e)}} \approx \sqrt{\frac{8s}{d \cdot 1.4427}}$. By setting $s = 58$ (and $d = 3072$) which results in $\epsilon = 0.32356413$, the inequality is obtained.

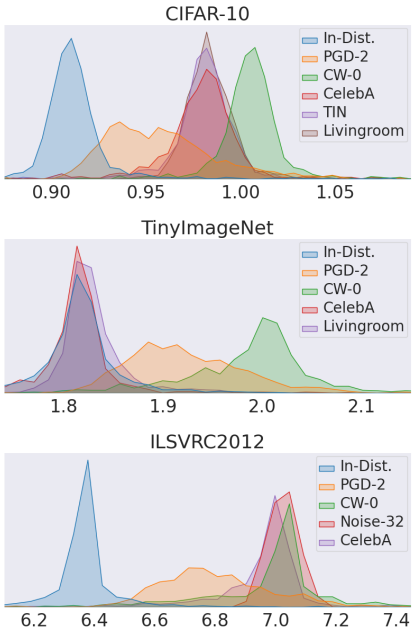

Figure 7: Histograms for reconstruction error without penalty in the latent space. The x-axis is R.