8. Supplementary Material

8.1. Other Privacy Attacks

Besides membership inference attacks, there exists a wide range of privacy attacks against neural networks. Model inversion attacks, first proposed by [14], aim at reconstructing features of the training data, e.g. recovering an image of a person from face recognition models [13]. Property inference attacks, proposed by [15], do not focus on the privacy of individual data samples, as in membership inference and model inversion attacks, but focus at inferring global properties of the training data, such as the environment in which the data was produced or the fraction of the data that comes from a certain class.

Model extraction attacks, also referred to as model stealing, attack a model f by constructing a substitute model $\hat{f}$ that is either identical or equivalent to f [40, 24]. Related line of work [45, 32] attempts to infer hyperparameters such as the optimization proccess, e.g. SGD or ADAM.

8.2. Detailed Description of Our MIA Algorithm

Our MIA consist of computing the two terms in Eq. (2), i.e. L_{rec} and L_{pred} for a given query pair (x, y), where x is an image from the input domain and y is the ground truth from the output domain, using only a black-box access to the victim conditional generation model $\mathbf{V}$.

L_{rec} is computed using the pixel-wise error between the output image predicted by the model, $\mathbf{V}(x)$, and the ground truth image y, see step 1 in the Algorithm 1. For image translation models, we set the pixel-wise error function, err, to be the L_1 loss:

$$L_{rec}^{trans}(x, y) = \|\mathbf{V}(x) - y\| \tag{3}$$

For semantic segmentation, where the output values are probability vectors rather then pixel values, we use the cross-entropy loss:

$$L_{rec}^{seg}(x, y) = -log(\mathbf{V}(x)[y]) \tag{4}$$

In the case of medical segmentation, following Fan et al. [11, 12], we use the weighted IoU loss and binary cross-entropy loss:

$$L_{rec}^{med}(x, y) = L_{IoU}^{w}(x, y) + L_{BCE}^{w}(x, y) \tag{5}$$

Defined as:

$$L_{IoU}^{w} = 1 - \frac{\sum_{i=1}^{H}\sum_{j=1}^{W} w_{ij}(\mathbf{V}(x)_{ij} \cdot y_{ij})}{\sum_{i=1}^{H}\sum_{j=1}^{W} w_{ij}(\mathbf{V}(x)_{ij} + y_{ij} - \mathbf{V}(x)_{ij} \cdot y_{ij})} \tag{6}$$

$$L_{BCE}^{w} = -\frac{\sum_{i=1}^{H}\sum_{j=1}^{W} w_{ij} log(\mathbf{V}(x)[y]_{ij})}{\sum_{i=1}^{H}\sum_{j=1}^{W} w_{ij}} \tag{7}$$

Where H and W are the height and width of the query sample, and w_{ij} is the weight of pixel (i, j) and is defined as follows, where A_{ij} represents the area that surrounds the pixel (i, j):

$$w_{ij} = 1 - \left| \frac{\sum_{m,n \in A_{ij}} y_{mn}}{\sum_{m,n \in A_{ij}} 1} - y_{ij} \right| \tag{8}$$

L_{pred} is computed as the average error of a linear regression model, $\mathbf{P}$, in predicting pixel values from deep features of the input image.

Our deep features are the activation values in the first 4 blocks of a pre-trained Wide-ResNet50$\times$2 [50]. These features are of sizes $56 \times 56 \times 256$, $28 \times 28 \times 512$, $14 \times 14 \times 1024$, and $7 \times 7 \times 2048$. We interpolate all features to size 56×56 using bi-linear interpolation (step 2), and also reduce the output image to 56×56 using bicubic interpolation (step 3). This gives a concatenated feature vector of size 3840 for each pixel i in the 56×56 image (256+512+1024+2048=3840). We denote the concatenated feature vector for pixel i as $\psi(i)$.

We randomly select 70% of the pixels as train set, and compute a linear model $\mathbf{P}$ to estimate the RGB pixel values y_{train}^{i} from the corresponding deep features $\psi_{train}(i)$ (step 4). The remaining 30% of pixels will be used as a test split, $\{\psi_{test}, y_{test}\}$ (step 5). I.e. $|\psi_{train}| = 2195 \times 3840, |y_{train}| = 2195 \times 3$ and $|\psi_{test}| = 941 \times 3840, |y_{test}| = 941 \times 3$.

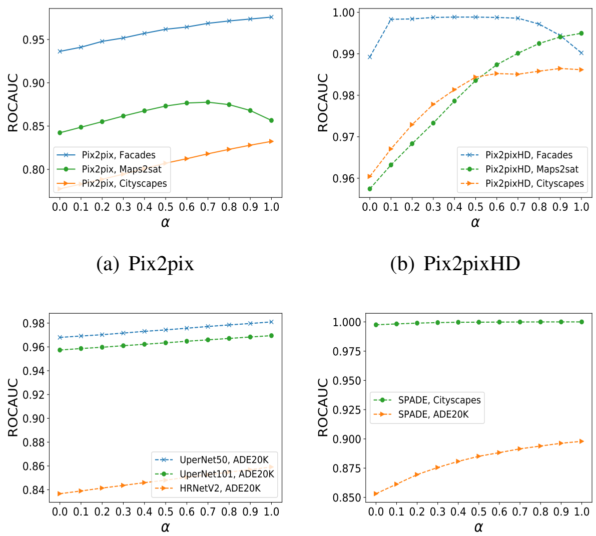
The linear regression model $\mathbf{P}$, a matrix of size 3840×3, is trained to minimize the error over $\{\psi_{train}, y_{train}\}$ (step 6). L_{pred} will be the average absolute error over $\{\psi_{test}, y_{test}\}$ (step 7). We found that fitting the linear model to 70% of pixels and measuring the error on the remaining 30% gives better results than just measuring the error of the linear fitting.

We compute L_{mem} according to Eq. (2) and compare the results with a predefined threshold value τ, such that any pair (x, y) for which is holds that $L_{mem}(x, y) < \tau$ is denoted as a member of the victim models' $\mathbf{V}$ train set (steps 8-9).

We have experimented with different resize methods (step 3) and found that our attack success rate is not very sensitive to the resize method. Additionally, we evaluated the effect of different train-test partitions (steps 4 & 5) and found that using less than 50% of the image pixels for training the linear regression model results with unstable performance, while all values of 50% or above results in similar attack success rates.

8.3. Parameter Selection

We experimented with different values for the α value in Eq. (2). As can be seen in Fig. 6, $\alpha = 1$ was the best choice over all benchmarks.



(a) Pix2pix (b) Pix2pixHD

(c) Semantic Segmentation (d) SPADE

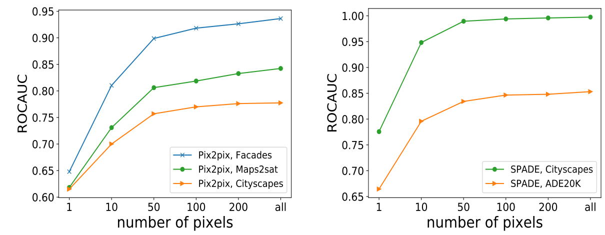
Figure 6. Effect of α in Eq. (2) over the attack success.

Algorithm 1. Membership Inference Attack
Input: Query pair (x, y), victim model $\mathbf{V}$, feature extractor $\mathbf{F}$, scalar α, threshold τ, error function err
Output: Membership inference result

1. $L_{rec} = err(\mathbf{V}(x), y)$
2. $\psi = \mathbf{F}(x) // \ |\psi| = 56 \times 56 \times 3840$
3. $y = resize(y, 56 \times 56 \times 3)$
4. $\{\psi_{train}, y_{train}\} \xleftarrow{70\%} \{\psi, y\}$
5. $\{\psi_{test}, y_{test}\} = \{\psi, y\} \setminus \{\psi_{train}, y_{train}\}$
6. Train linear regression $\mathbf{P}$ with $\{\psi_{train}, y_{train}\}$
7. $L_{pred} = \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{P}\psi_{test}(i) - y^i_{test}\|_1 \ //N = 941$
8. $L_{mem} = L_{rec} - \alpha \cdot L_{pred}$
9. **if** $L_{mem} < \tau$ **then**
   Return **True**
   else
   Return **False**

8.4. MIA vs output dimension

As described in Sec. 4.1, we evaluted the effect of reducing the output dimension on the accuracy of reconstruction-based MIA. The reduction was achieved by randomly sampling N output pixels, and using them as the output, where N ranges from a single pixel and up to 200 pixels. Fig. 7 demonstrates that MIA accuracy indeed scales with the number of output dimensions. Results for Pix2PixHD, UperNet and HRNetV2 are presetned in Fig. 2.



(a) Pix2pix (b) SPADE

Figure 7. Effect of reducing output dimensionality over a reconstruction-based attack. MIA accuracy is correlated with the decrease of output dimension, i.e. number of pixels, demonstrating that high output dimensionality problems are more vulnerable to MIA.

8.5. calibration Effect

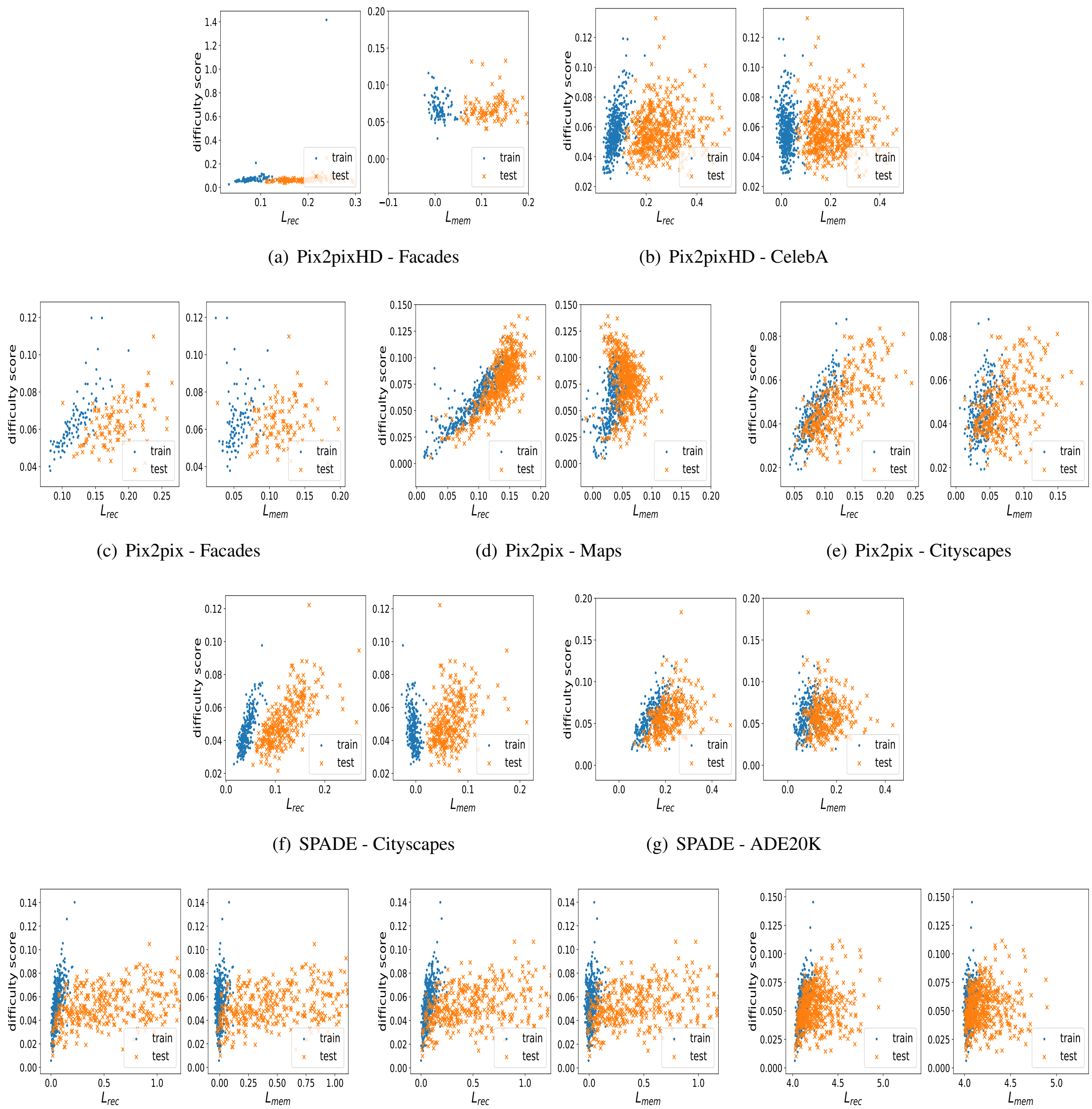
As can be seen in Tab. 1, using our membership error L_{mem}, Eq. (2), substantially improves the sucess rates in all of our experiments. As can be seen in Fig. 8, our L_{mem} can better separate train and test images by a simple threshold compared to the reconstruction error L_{rec}. Results for Pix2PixHD on the Maps2sat and Cityscapes datasets are presented in Fig. 4.

8.6. Human-Supervised Image difficulty score

We compare our self-supervised single-sample predictability error with the human-supervised difficulty score proposed by [41]. In Fig. 9, we present images ranked from easy to difficult using our implementation of the supervised-image difficulty score, for the Cityscapes and Maps datasets. The ranking seems correlated with image sharpness and level-of-detail images. As can be seen in Tab. 3, our score outperforms the human-supervised score. We compare the correlation between the reconstruction error for unseen images to our self-supervised predictability error and the human-supervsied scorre.

8.7. Multi-Image predictability error

As discussed in Sec. 4.2.2, we compare our single-sample predictability error to a multi-sample predictability error (MSPS) by training a "shadow" model, sharing the same architecture as the victim model, on auxiliary samples. As can be seen in Tab. 3, when training the MSPS on 100 images, it underperforms our method on Pix2PixHD and the evaluated semantic segmentation models. For the smaller Pix2Pix architecture, MSPS was more successful, obtaining



(a) Pix2pixHD - Facades (b) Pix2pixHD - CelebA

(c) Pix2pix - Facades (d) Pix2pix - Maps (e) Pix2pix - Cityscapes

(f) SPADE - Cityscapes (g) SPADE - ADE20K

(h) UperNet50 - ADE20K (i) UperNet101 - ADE20K (j) HRNetV2 - ADE20K

Figure 8. The proposed membership error L_{mem} can better separate train and test images by a simple threshold (i.e. a vertical line) compared to the reconstruction error L_{rec}. Pix2pixHD for Maps2sat and Cityscapes are presented in Fig. 4

competitive results with our method. We analyzed the effect of number of samples over the MSPS performance. As can be seen in Fig. 10, in most tasks, increasing the number of samples did not improve performance.

We also compare our method to the setting were many out-of-distribution but similar sample are available. We trained shadow models on $4K$ samples from the BDD dataset as MSPS for the Cityscapes dataset. As can be seen in Tab. 8, this too underperforms our method. Note that it is rare to have similar datasets with nearly identical labels, such as in the case of BDD and Cityscapes.

8.8. Shadow models

As discussed in Sec. 4.2.2, for the interest of completeness we compare our method with the popular approach of shadow-model classifiers for image translation MIA. For this,

Model	Dataset	Ours train / test	Human-Supervised train / test
Pix2Pix	Facades	0.79 / 0.50	-0.02 / 0.16
Pix2Pix	Maps2sat	0.51 / 0.77	0.79 / 0.52
Pix2Pix	Cityscapes	0.78 / 0.71	0.04 / 0.09
Pix2PixHD	Facades	0.67 / 0.36	0.27 / 0.04
Pix2PixHD	Maps2sat	0.38 / 0.79	0.77 / 0.56
Pix2PixHD	Cityscapes	0.76 / 0.62	0.36 / 0.48
SPADE	Cityscapes	0.80 / 0.68	0.29 / 0.53
SPADE	ADE20K	0.48 / 0.27	0.25 / -0.05
UperNet50	ADE20K	0.66 / 0.13	0.34 / 0.05
UperNet101	ADE20K	0.65 / 0.13	0.38 / 0.06
HRNetV2	ADE20K	0.61 / 0.22	0.36 / 0.10

Table 7. Our self-supervised difficulty score is better correlated with the reconstruction error than the human-supervised

Figure 9. Examples of images from the Cityscapes (first two rows) and Maps2sat (last two rows) datasets that received the lowest (first and third row) and highest (second and last row) predictability errors using the supervised difficulty score.

Model	Dataset	Ours	Multi-Image In-Dist	Multi-Image BDD
Pix2pix	Cityscapes	**82.94%**	**82.47%**	74.43%
Pix2pixHD	Cityscapes	**99.29%**	96.86%	66.2%

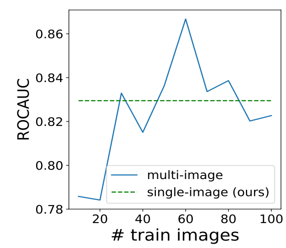
Table 8. Comparison between our single-image predictability error and two multi-image baselines, using in-distribution images and a larger amound of out-of-distribution images (BDD).

we select N images, denoted as *shadow_train*, for training the shadow model. As an upper-bound, the shadow model is given the exactly same architecture as used by the victim model. Another N images, not seen by the shadow model, are set to be *shadow_test*. We define a new label for each sample as follows:

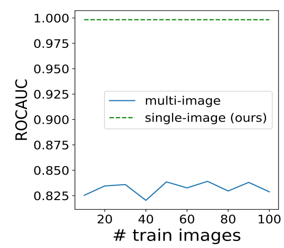
$$label(x) = \begin{cases} 0, & \text{if } x \leftarrow shadow\_train \\ 1, & \text{if } x \leftarrow shadow\_test \end{cases} \quad (9)$$

The classifier **C** architecture and training procedure are similar to [21]. For each image, we compute the structured loss map between the ground-truth image and the generated image, using L_1 as the loss function. At every epoch we randomly crop 15 patches of size 90×90 from the structured loss map. We train a ResNet-50 [20] from scratch on the 90×90 patches, modified for binary classification. We use a batch size of 8, SGD optimizer, weight decay of $1e-2$, initial learning rate of 0.1 which reduces by a factor of 0.1 every 15 epochs. As previously mentioned, we do not evaluate this on the Facades dataset, due to its size.

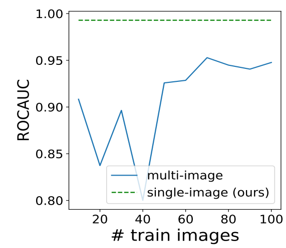
(a) Pix2pix-Maps2sat



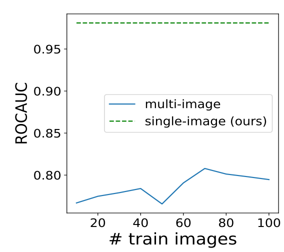
(b) Pix2pix-Cityscapes



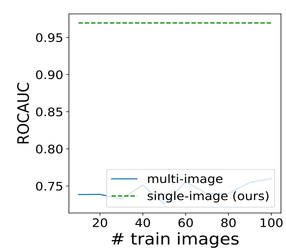
(c) Pix2pixHD-Maps2sat



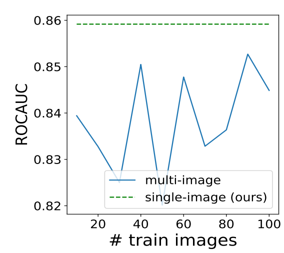
(d) Pix2pixHD-Cityscapes



(e) UperNet50-ADE20K



(f) UperNet101-ADE20K



(g) HRNetV2-ADE20K

Figure 10. Comparison of MIA accuracy when using our single sample vs. using multi-sample predictability errors, as a function of the number of training images. Note that the multi-image score assumes knowledge of the victim's model, as well as the availability of many labeled training images

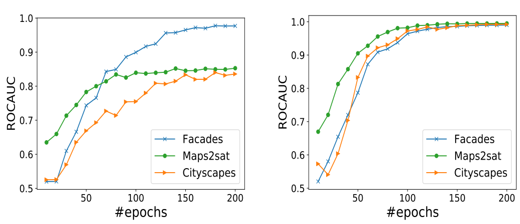
Model	Dataset	Ours	In-Dist		Out-of-Dist (BDD)	
		ROC	ROC	Acc.	ROC	Acc.
Pix2pix	Maps2sat	**85.65%**	80.15%	73.4%	-	-
Pix2pix	Cityscapes	**83.23%**	78.68%	67.5%	72.57%	56.16%
Pix2pixHD	Maps2sat	**99.42%**	98.63%	93.7%	-	-
Pix2pixHD	Cityscapes	**99.09%**	96.39%	64.0%	95.78%	56.5%

Table 9. Comparison between our MIA and the commonly used shadow-model-based classifier attack, using $4K$ train and $4K$ test images from the BDD dataset. Our MIA outperforms while not requiring extra training images.

We compare the performance of our single-sample method to the shadow model method in Tab. 9. For fairness we compare both the ROCAUC over the classifier's confidence, as well as the classification accuracy. It can be seen that in both cases, and for either in-distribution or out-of-distribution auxiliary data, the shadow model approach is inferior to our method for image translation models. We discuss the case of semantic segmentation in Sec. 4.2.2.

8.9. Memorization

As mentioned in Sec. 5, memorization is the main reason for the success of our method. Fig. 11 shows the accuracy of our method as a function of the number of epochs used for training the victim model, clearly suggesting that memorization is indeed the vulnerability.



(a) Pix2pix (b) Pix2pixHD

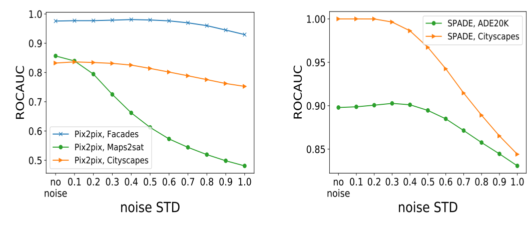
Figure 11. Effect of memorization on the attack success rate.

8.10. Defenses

In Sec. 5, we discuss the Gauss defense, including other common defenses, against our attack. We evaluated our attack accuracy as a function of different noise STD. Fig. 12 shows that a considerable amount of noise, which corrupts the generated output, is required in order to have a significant effect over our attack success, which is still much better than random guessing (50%). Results for Pix2PixHD, UperNet and HRNetV2 are presented in Fig. 5.

8.11. ImageNet predictability error

Our predictability error relies on learning a mapping between feature vectors to their corresponding pixel values. We use a pre-trained Wide-ResNet50×2 [50], which is trained



(a) Pix2pix (b) SPADE

Figure 12. Effect of Gauss defense on the attack success rate. Even with large amounts of added noise, our attack still manages to success much better then random guessing.

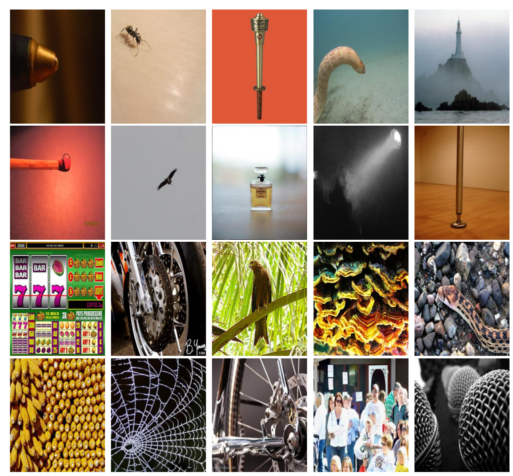


Figure 13. Examples of images from the ImageNet dataset that received the lowest and highest predictability errors. First row - lowest scored train images. Second row - lowest scored test images. Third row - highest scored train images. Last row - highest scored test images. As can be seen, the predictability error is effective even on images that were used for training the feature extractor.

on the ImageNet dataset. We do not make any assumptions regarding an overlap between the pre-trained model's training data (i.e. ImageNet) and the data during in the attack. In the scenario in which such an overlap exists, the concern is that the predictability error would lose its credibility.

In order to verify this, we computed the predictability error of a random subset of 1K train images and 1K test images, from the ImageNet dataset. As no input-output pairs exists, we trained the linear predictor to predict pixel values from deep features of the same image. We do not observe any significant difference between the two - both share similar mean and std values: $(0.0549, 0.018)$ for the train images and $(0.0556, 0.0191)$ for the test images. A ROCAUC score of 51% further demonstrates that there is no clear difference between the distribution of the predictability error on seen and unseen images.

Fig. 13 further demonstrates this. The first row presents the train images that received the lowest scores, i.e. marked as easy images, and the second row presents the test images with the lowest scores. Both correspond to ”plain” images, regardless of whether they are known (train) or unknown (test). The same applies to the Difficult images. The third row presents the highest scored train images and the last row presents the highest scored test images. Both contains complex patterns and high variance. This demonstrates that the predictability error is not affected by the having prior knowledge of the image, and is only measuring the amount of variance and complexity of an image.