Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Appendix Section 5.4.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Appendix Section A.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] Provided with the supplementary materials.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Appendix D.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Appendix D.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] See Section 2.
   (b) Did you mention the license of the assets? [No] All licences MIT.
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No] Used datasets are either synthetic or popular standard datasets.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No] Used datasets are either synthetic or popular standard datasets.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Negative Societal Impacts

Our work aims to defend self-supervised models against model stealing attacks. Since we are directly defending models and aim to provide attribution, any negative societal impacts of our work are minimal. One potentially negative impact could be if the t-test result is inconclusive about a stolen model being stolen or if it incorrectly identifies an independent model as stolen. However, as shown from our results, we are consistently able to differentiate correctly between stolen and independent models and can use our metrics to further reinforce the results from dataset inference. In terms of data and model access, we assume that the victim or a trusted third party, such as law enforcement, is responsible for running the dataset inference so that there are no privacy-related concerns.

B Protocol for Dataset Inference

We design the following protocol for Dataset Inference:

1. Select a third-trusted party as an arbitrator for the ownership resolution.
2. Arbitrator ensures that the train and validation sets are IID (from the same distribution) by combining the train and test sets followed by a random split into the training set for the defended model and the private validation set used for dataset inference.
3. Specification of the number of data points used for dataset inference: use all the data points from the validation set and the equivalent number of data points from the train set.

C More Related Work

C.1 Membership Inference

EncoderMI [30] leverages the finding that an image encoder overfits to its pre-training dataset and returns more similar embeddings for pairs of augmented pre-training data points than for points not in the pre-training set. EncoderMI assumes some data points to be assessed as members and a shadow dataset as inputs. The first step is to create n augmentations of a point from the shadow dataset and compute for it $\binom{n}{2}$ (pair-wise) similarity scores using the embeddings extracted from a shadow encoder, which is trained on the shadow dataset. The scores form the membership feature vector for a given shadow data point. After labeling each such point as member or non-member, the membership features and the corresponding labels are used to train an inference classifier to infer if a given data point was a member or non-member of a target encoder. The early stopping is investigated as a mitigation defense against EncoderMI. The defense can reduce the effectiveness of the attack, however, at the cost of the lower performance of the defended encoder on downstream tasks.

C.2 Non-Transferable Learning

Non-Transferable Learning (NTL) [45] achieves ownership resolution and usage authorization by discouraging the model to generalize to data domains outside of its training data. To perform ownership resolution, NTL incentives the model to generalize poorly to a specific target domain, making it more likely to mis-classify on data from that domain. The authors argue that the model's unexpectedly poor behavior on the target domain can then be used like a watermark to claim ownership, and unlike many previous watermarking defenses, is harder to remove because the misclassification behavior is embedded in the model. To control usage authorization, NTL intentionally degrades the model performance on the target domain. The authors argue that this prevents users from applying the model on unauthorized data.

C.3 Metrics to Compare Encoders

The desired metric for comparison between two representations should evaluate whether two representations are essentially similar or importantly different [13].

Table 4: **Summary of our encoders.** We show all possible combinations of victim, stolen, and independent encoders along with their architectures. *, ** denotes that the models listed are equivalent.

Victim Encoder		Stolen Encoder			Independent Encoder	
D	Architecture	D	Architecture	# Queries	D	Architecture
CIFAR10	ResNet34	CIFAR10	ResNet34	500 - 50K	CIFAR100*	ResNet18
CIFAR10	ResNet34	SVHN	ResNet34	500 - 50K	CIFAR100*	ResNet18
CIFAR10	ResNet34	STL10	ResNet18	500 - 50K	SVHN	ResNet34
SVHN	ResNet34	CIFAR10	ResNet34	500 - 50K	CIFAR100*	ResNet18
SVHN	ResNet34	SVHN	ResNet34	500 - 50K	CIFAR100*	ResNet18
SVHN	ResNet34	STL10	ResNet34	500 - 50K	CIFAR10	ResNet34
ImageNet	ResNet50	CIFAR10	ResNet50	5K - 50K	CIFAR100**	ResNet50
ImageNet	ResNet50	SVHN	ResNet50	5K - 250K	CIFAR100**	ResNet50
ImageNet	ResNet50	ImageNet	ResNet50	5K - 250K	SVHN	ResNet50

D Additional Details on Experiments

D.1 Datasets Used

CIFAR10 [28]: The CIFAR10 dataset consists of 32x32 colored images with 10 classes. There are 50000 training images and 10000 test images.

CIFAR100 [28]: The CIFAR100 dataset consists of 32x32 coloured images with 100 classes. There are 50000 training images and 10000 test images.

SVHN [34]: The SVHN dataset contains 32x32 coloured images with 10 classes. There are roughly 73000 training images, 26000 test images and 530000 "extra" images.

ImageNet[11]: Larger sized coloured images with 1000 classes. As is commonly done, we resize all images to be of size 224x224. There are approximately 1 million training images and 50000 test images.

STL10 [8]: The STL10 dataset contains 96x96 coloured images with 10 classes. There are 5000 training images, 8000 test images, and 100000 unlabeled images.

D.2 Encoder similarity scores and p-values

We present additional results for the encoder similarity scores and p-values from dataset inference vs the number of queries in Table **??**.

D.3 Details on Experimental Setup

We show a summary of the encoders used in our experiments in Table 4.

The ResNet18/ResNet34 architectures used for the CIFAR10 and SVHN victim encoders and the related stolen and independent encoders used a 3x3 Conv layer of stride 1 instead of the default 7x7 Conv layer and did not use a max pooling layer. When stealing from the ImageNet victim encoder, the images used for queries were resized to be of size 224x224. Similarly when training independent ResNet50 encoders to be used with the ImageNet victim encoder, the images in the respective datasets were resized to a size of 224x224.

For the results in Tables 1 and 2, encoders with the highest numbers of queries for each case were used. In other words, for encoders stolen from the CIFAR10 or SVHN victim encoders, 50K queries were used while for encoders stolen from the ImageNet victim encoder, 250K queries were used. Note that since CIFAR10 does not have 250K different examples, 60K queries from the aggregated training and test set were used when stealing with the CIFAR10 dataset. Encoders with a smaller number of queries were also stolen with the numbers of queries ranging from 500 - 50K for the CIFAR10 and SVHN victim encoders, and queries ranging from 5K - 250K for the ImageNet victim encoder.

We train the SVHN and CIFAR10 victim models for 200 epochs. To train the independent and stolen encoders, we used 100 epochs. When stealing from the ImageNet victim encoder, the SGD/LARS optimizer was used while for other models, the Adam optimizer was used. The initial learning rate

Algorithm 1 Stealing an Encoder [14].

Input: Querying Dataset $\mathcal{D}$, access to a victim encoder $f_v(w;\theta_v)$.
Output: Stolen representation model $f_s(w;\theta_a)$

Initialize f_s with a similar architecture as f_v.
for sampled queries $\{x_k\}_{k=1}^{N} \in \mathcal{D}$ **do**
  Query victim encoder to obtain representations: $y_v = f_v(x_k)$
  Generate representations from stolen encoder: $y_s = f_s(x_k)$
  Compute loss $\mathcal{L}\{y_v, y_s\}$.
  Update stolen encoder parameters $\theta_s := \theta_s - \eta\nabla_{\theta_s}\mathcal{L}$.
end for

Algorithm 2 Kozachenko-Leonenko Entropy Estimator for Encoders.

Input: Dataset $\mathcal{D}$, number of data points $N \geq 2$ to be sampled from $\mathcal{D}$, access to an encoder $f(\cdot)$, the Euler-Mascheroni constant $\gamma \approx 0.577$, and the gamma function Γ.
Output: Entropy Estimation H.

Sample N data points from $\mathcal{D}$: $x_1, ..., x_N$.
for each sampled data point $\{x_k\}_{k=1}^{N} \in \mathcal{D}$ **do**
  Sample an augmentation t.
  Generate view $w_k = t(x_k)$.
  Query the encoder f to generate the representation: $y_k = f(w_k)$.
end for
for each representation $\{y_k\}_{k=1}^{N} \in \mathbf{R}^d$ **do**
  Find nearest neighbor distance: $R_k = ||y_k - y_i||_2$, where $i \neq k$.
  Compute transformation: $z_k = (N-1)\cdot(R_k)^d$.
end for
Compute the volume of the unit ball in $\mathbf{R}^d$: $B_d = \frac{\pi^{d/2}}{\Gamma(1+d/2)}$.
Compute Entropy: $H = \frac{1}{N}\sum_{i=1}^{N}\log z_i + \log B_d + \gamma$

was kept constant in all cases and was adjusted with the Cosine Annealing scheduler. A batch size of 256 or 512 was used for training the models. The temperature parameters used varied between 0.1, 0.15, 0.2, and 0.25 with a larger temperature used for models with a higher number of queries. For all queries under 50K, the temperature was set to be 0.1.

We ran all experiments on machines equipped with an Intel® Xeon® Silver 4210 processor, 128 GB of RAM, and four NVIDIA GeForce RTX 2080 graphics cards, running Ubuntu 18.04.

E Entropy Estimation

We present the entropy estimator in Algorithm 2 and the joint entropy estimator in Algorithm 3.

F Linear Evaluation of Encoders

In Table 5, we show results for the downstream accuracies of models stolen from the encoder pre-trained on the ImageNet dataset. The extraction of the representation model is possible at a fraction of the cost with a smaller number of queries (less than one-fifth) required to train the victim model. In general, the performance of the stolen encoders increases with the number of queries. We also perform a similar evaluation for victim encoders trained on the CIFAR10 and SVHN datasets and models stolen from these encoders in Tables 6 and 8, respectively.

Algorithm 3 Kozachenko-Leonenko **Joint** Entropy Estimator for Encoders.

Input: Dataset $\mathcal{D}$, number of data points $N \geq 2$ to be sampled from $\mathcal{D}$, an access to an encoder $f(\cdot)$, an access to an encoder $g(\cdot)$, the Euler-Mascheroni constant $\gamma \approx 0.577$, and the gamma function Γ.
Output: Joint Entropy Estimation H.

Sample N data points from $\mathcal{D}$: $x_1, ..., x_N$.
for each sampled data point $\{x_k\}_{k=1}^N \in \mathcal{D}$ **do**
  Sample an augmentation t.
  Generate view $w_k = t(x_k)$.
  Query the encoder f to generate the representation: $\hat{y}_k = f(w_k)$.
  Query the encoder g to generate the representation: $\bar{y}_k = g(w_k)$.
  Concatenate the representations: $y_k = \hat{y}_k \parallel \bar{y}_k$.
end for
for each representation $\{y_k\}_{k=1}^N \in \mathbf{R}^d$ **do**
  Find nearest neighbor distance: $R_k = ||y_k - y_i||_2$, where $i \neq k$.
  Compute transformation: $z_k = (N-1) \cdot (R_k)^{2d}$.
end for
Compute the volume of the unit ball in $\mathbf{R}^d$: $B_d = \frac{\pi^{d/2}}{\Gamma(1+d/2)}$.
Compute Entropy: $H = \frac{1}{N} \sum_{i=1}^N \log z_i + \log B_{2d} + \gamma$

Table 5: **Linear Evaluation Accuracy** on a victim and stolen encoders. The victim encoder is pre-trained on the ImageNet dataset.

# OF QUERIES	DATASET	DATA TYPE	CIFAR10	CIFAR100	STL10	SVHN	F-MNIST
Victim Encoder	N/A	N/A	90.33	71.45	94.9	79.39	91.9
60K	CIFAR10	TRAIN/TEST	83.3	57.0	71.2	73.8	90.7
250K	IMAGENET	TRAIN	**80.0**	**57.0**	**85.8**	71.5	90.2
5K	SVHN	EXTRA	42.0	16.2	34.4	26.9	81.3
10K	SVHN	EXTRA	60.8	33.0	50.5	71.7	87.5
50K	SVHN	EXTRA	73.3	47.1	58.2	78.8	90.4
100K	SVHN	EXTRA	76.3	50.2	61.1	78.2	90.8
200K	SVHN	EXTRA	76.9	52.0	62.1	78.3	90.8
250K	SVHN	EXTRA	77.1	52.6	61.9	**80.2**	**91.4**

G Metrics for Measuring the Quality of Stolen Encoders

This section considers additional metrics for measuring the quality of stolen encoders. As in Section 5.3, we select a random sample of N inputs from the training set and find the representations of the encoders on each of the inputs. The representations for each input are then centered by subtracting the mean and normalized to be unit vectors. For each of the inputs, the ℓ_p norm of the difference in representations by the encoders is computed where $p = 1, 2, \infty$. The final value used as the metric is then the mean of these norms over all of the inputs. Tables 9, 10, and 11 show the results obtained for the ℓ_1, ℓ_2, and ℓ_∞ norms respectively. In a similar way, we find the cosine similarity between representations (closely related to the ℓ_2 norm) and present results in Table 12. All norms and the cosine similarity are able to differentiate between stolen and independent encoders, however the ℓ_1, ℓ_2 norms and cosine similarity have a more clear difference.

G.1 Analysis of Distance and Cosine Similarity Based Metrics

We use the ℓ_2 score based on the ℓ_2 norm of the distance between representations, and the cosine similarity score between representations as ways to measure the quality of stolen encoders and differentiate between stolen and independent encoders.

We create two score metrics, namely the cosine similarity score C:

$$C = |\text{sim}(a, b)| \tag{2}$$

Table 6: **Linear Evaluation Accuracy** on a victim and stolen encoders. The victim encoder is pre-trained on the CIFAR10 dataset.

# OF QUERIES	DATASET	CIFAR10	STL10	SVHN
Victim Encoder	N/A	87.4	73.4	49.5
50K	SVHN	61.2	51.7	54.8
50K	CIFAR10	84.8	70.7	52.4
50K	STL10	86.8	73.0	49.8

Table 7: **Linear evaluation accuracy, mutual information score, cosine similarity and p-values** on a victim and stolen encoders. The victim encoder is pre-trained on the SVHN dataset. We observe higher performance on downstream tasks with more queries and similarly observe higher similarity scores for encoders stolen with more queries (see Table 3).

# OF QUERIES	DATASET	CIFAR10	STL10	SVHN	$S(\cdot, f_v)$	$C(\cdot, f_v)$	P-VALUE
Victim Encoder	N/A	57.5	50.6	80.5	1	1	9.69E-227
500	CIFAR10	24.5	23.3	19.3	0.0	0.24	6.89E-1
5K	CIFAR10	53.3	45.2	58.1	0.11	0.40	3.51E-1
7K	CIFAR10	57.9	50.9	69.2	0.14	0.46	4.72E-1
8K	CIFAR10	59.6	52.0	72.6	0.53	0.47	9.87E-2
9K	CIFAR10	59.9	50.8	71.5	0.57	0.49	6.23E-2
10K	CIFAR10	59.1	51.3	72.3	0.69	0.52	5.82E-3
20K	CIFAR10	60.6	51.5	73.4	0.92	0.58	2.31E-7
30K	CIFAR10	60.7	52.1	74.1	0.93	0.63	2.11E-10
50K	CIFAR10	59.8	51.6	75.1	0.94	0.69	1.19E-17
50K	STL10	60.6	51.5	76.8	0.95	0.89	1.65E-11
50K	SVHN	55.6	48.7	82.2	0.96	0.91	1.05E-75
Independent	CIFAR10	87.4	73.4	49.5	0.90	0.009	3.56E-1
Independent	CIFAR100	73.8	61.7	67.7	0.90	0.007	2.13E-1
Independent	STL10	79.4	73.1	55.8	0.84	0.015	4.62E-1

where $\text{sim}(a, b) = \frac{a^T b}{||a||_2 ||b||_2}$ is the cosine similarity between representation vectors a and b, and the ℓ_2 distance score which transforms the ℓ_2 norm of the difference as:

$$Score_{\ell_2} = 1 - \frac{1}{2} \left\| \frac{a}{\|a\|_2} - \frac{b}{\|b\|_2} \right\|_2 \tag{3}$$

We first note that there are various ways by which an attacker may steal an encoder. To simplify the analysis, we consider two main cases, one where the attacker minimizes the mean squared error between its representations and the representations returned by the victim and the other where the attacker minimizes the InfoNCE contrastive loss (other contrastive loss functions are similar).

With the MSE loss, the attacker directly minimizes the mean squared error between its representations and the representations of the victim encoder on the queries it makes. Let x_i be a query made by an attacker and $f_v(x_i) = h_{v_i}, f_s(x_i) = h_{s_i} \in \mathcal{R}^n$ be the representations of the victim and stolen encoders respectively for this query. The MSE loss between these two representations is then $\frac{1}{n}\sum_{j=1}^{n}(h_{v_{ij}} - h_{s_{ij}})^2$. Comparatively, the ℓ_2 distance (ℓ_2 norm of the difference) between these two representations is $||h_{v_i} - h_{s_i}||_2 = \sqrt{\sum_{j=1}^{n}(h_{v_{ij}} - h_{s_{ij}})^2}$. From these two expressions, it follows directly that minimizing the MSE loss, which is equivalent to minimizing $\sum_{j=1}^{n}(h_{v_{ij}} - h_{s_{ij}})^2$, also minimizes the ℓ_2 distance.

We now consider the case where an attacker uses a contrastive loss function such as the InfoNCE loss [43], specifically as used in [5]. The InfoNCE loss consists of positive and negative pairs and encourages positive pairs to have similar representations and negative pairs to have dissimilar representations. When stealing from a victim encoder, the attacker uses its own representation and the representation from the victim encoder for a single query as a positive pairs while the other inputs in the batch are considered negative pairs. Given an input batch of queries $\{x_1, x_2, \ldots, x_m\}$

Table 8: **Linear evaluation accuracy, mutual information score, cosine similarity and p-values** on a victim and stolen encoders. The victim encoder is pre-trained on the CIFAR10 dataset. We observe higher performance on downstream tasks with more queries and similarly observe higher similarity scores for encoders stolen with more queries (see Table 3)

# OF QUERIES	DATASET	CIFAR10	STL10	SVHN	$S(\cdot, f_v)$	$C(\cdot, f_v)$	P-VALUE
Victim Encoder	N/A	87.4	73.4	49.5	1	1	1.37E-16
500	SVHN	22.3	20.8	20.1	0.00	0.07	5.69E-1
1K	SVHN	26.5	23.4	29.9	0.06	0.11	9.02E-1
2K	SVHN	40.0	35.2	46.7	0.12	0.13	7.83E-1
3K	SVHN	44.4	39.8	56.3	0.18	0.21	8.04E-1
4K	SVHN	47.4	42.5	60.8	0.22	0.19	6.30E-1
5K	SVHN	49.5	43.8	60.9	0.29	0.28	4.28E-1
10K	SVHN	55.6	46.6	58.9	0.38	0.36	6.42E-1
20K	SVHN	57.7	48.7	60.4	0.43	0.39	9.81E-2
30K	SVHN	58.0	49.2	57.0	0.57	0.45	6.73E-2
40K	SVHN	61.4	51.1	55.0	0.69	0.49	4.83E-2
50K	SVHN	61.2	51.7	54.8	0.76	0.50	3.97E-2
50K	CIFAR10	84.8	70.7	52.4	0.84	0.95	8.73E-7
50K	STL10	86.8	73.0	49.8	0.89	0.92	1.04E-2
Independent	SVHN	57.5	50.6	80.5	0.12	0.0001	2.96E-1
Independent	CIFAR100	73.8	61.7	67.7	0.63	0.001	3.67E-1
Independent	STL10	79.4	73.1	55.8	0.34	0.001	5.21E-1

Table 9: ℓ_1 distance between normalized and centered representations from Encoders. We compute $d(\cdot, f_v)$ between the representations of a given encoder (in a row) and the Victim encoder f_v where d is the ℓ_1 norm.

	CIFAR10		SVHN		IMAGENET	
ENCODER	DATASET	$d(\cdot, f_v)$	DATASET	$d(\cdot, f_v)$	DATASET	$d(\cdot, f_v)$
f_s	SVHN	14.74 ± 0.22	SVHN	6.92 ± 0.10	SVHN	25.78 ± 0.26
	CIFAR10	5.19 ± 0.05	CIFAR10	11.92 ± 0.21	CIFAR10	24.89 ± 0.28
	STL10	6.24 ± 0.08	STL10	7.22 ± 0.14	IMAGENET	16.68 ± 0.21
f_i	SVHN	22.20 ± 0.07	CIFAR10	22.17 ± 0.06	SVHN	35.45 ± 0.21
	CIFAR100	23.65 ± 0.07	CIFAR100	23.13 ± 0.07	CIFAR100	38.96 ± 0.25

sent by an attacker, the representations for each query from both the victim and stolen encoders are concatenated as $\{h_{s_1}, h_{s_2}, \ldots, h_{s_m}, h_{v_1}, h_{v_2}, \ldots, h_{v_m}\}$ where $h_{s_i} = f_s(x_i)$ and $h_{v_i} = f_v(x_i)$ are the representations by the stolen and victim encoders respectively. The positive pairs are $(h_{s_1}, h_{v_1}), \ldots, (h_{s_m}, h_{v_m})$ and the loss between a positive pair of samples (h_{s_i}, h_{v_i}) is defined as $l(s_i, v_i) = -\log \frac{\exp(\text{sim}(h_{s_i}, h_{v_i})/\tau)}{\sum_{k=1}^{2m} \mathbb{1}_{[k \neq s_i]} \exp(\text{sim}(h_{s_i}, h_k)/\tau)}$. Here sim is the cosine similarity function ($\text{sim}(u, v) = \frac{u^T v}{||u||_2 ||v||_2}$), τ is the temperature parameter, and $\mathbb{1}_{[k \neq s_i]}$ is an indicator function equal to 1 iff $k \neq s_i$ and 0 otherwise. The overall loss for the batch is the sum of the losses over each positive pair i.e. $\mathcal{L} = \frac{1}{2m} \sum_{c=1}^{m} [l(s_c, v_c) + l(v_c, s_c)]$. We can combine the log terms to rewrite this loss function as $\mathcal{L} = -\frac{1}{2m} \log \frac{\prod_{c=1}^{m} (\exp(\text{sim}(h_{s_c}, h_{v_c})/\tau))^2}{\prod_{c=1}^{m} (\sum_{k=1}^{2m} \mathbb{1}_{[k \neq s_c]} \exp(\text{sim}(h_{s_c}, h_k)/\tau))(\sum_{k=1}^{2m} \mathbb{1}_{[k \neq v_c]} \exp(\text{sim}(h_{v_c}, h_k)/\tau))}$. Note that $\exp(r) > 0 \, \forall r$ so that the loss $\mathcal{L}$ is always positive (the denominator contains the terms in the numerator and each term is positive so the fraction is < 1 and has a negative log).

Simplifying $\mathcal{L}$ by combining the exponents in the numerator and using $\log \frac{a}{b} = \log a - \log b$ gives:

$\mathcal{L} = \frac{1}{2m} (\log \prod_{c=1}^{m} (\sum_{k=1}^{2m} \mathbb{1}_{[k \neq s_c]} \exp(\text{sim}(h_{s_c}, h_k)/\tau))(\sum_{k=1}^{2m} \mathbb{1}_{[k \neq v_c]} \exp(\text{sim}(h_{v_c}, h_k)/\tau)) - \frac{1}{2m} (\log \exp(\sum_{c=1}^{m} (2 \cdot \text{sim}(h_{s_c}, h_{v_c})/\tau)))$

$\mathcal{L} = \frac{1}{2m} (\log \prod_{c=1}^{m} (\sum_{k=1}^{2m} \mathbb{1}_{[k \neq s_c]} \exp(\text{sim}(h_{s_c}, h_k)/\tau))(\sum_{k=1}^{2m} \mathbb{1}_{[k \neq v_c]} \exp(\text{sim}(h_{v_c}, h_k)/\tau)) - \frac{1}{m} (\sum_{c=1}^{m} (\text{sim}(h_{s_c}, h_{v_c})/\tau))$

Table 10: ℓ_2 distance between normalized and centered representations from Encoders. We compute $d(\cdot, f_v)$ between the representations of a given encoder (in a row) and the Victim encoder f_v where d is the ℓ_2 norm divided by 2.

	CIFAR10		SVHN		IMAGENET	
ENCODER	DATASET	$d(\cdot, f_v)$	DATASET	$d(\cdot, f_v)$	DATASET	$d(\cdot, f_v)$
f_s	SVHN	0.49 ± 0.007	SVHN	0.21 ± 0.003	SVHN	0.55 ± 0.04
	CIFAR10	0.16 ± 0.02	CIFAR10	0.39 ± 0.007	CIFAR10	0.53 ± 0.004
	STL10	0.19 ± 0.003	STL10	0.23 ± 0.005	IMAGENET	0.33 ± 0.004
f_i	SVHN	0.71 ± 0.001	CIFAR10	0.70 ± 0.01	SVHN	0.71 ± 0.007
	CIFAR100	0.71 ± 0.001	CIFAR100	0.71 ± 0.01	CIFAR100	0.71 ± 0.007

Table 11: ℓ_∞ distance between normalized and centered representations from Encoders. We compute $d(\cdot, f_v)$ between the representations of a given encoder (in a row) and the Victim encoder f_v where d is the ℓ_∞ norm.

	CIFAR10		SVHN		IMAGENET	
ENCODER	DATASET	$d(\cdot, f_v)$	DATASET	$d(\cdot, f_v)$	DATASET	$d(\cdot, f_v)$
f_s	SVHN	0.13 ± 0.003	SVHN	0.10 ± 0.002	SVHN	0.27 ± 0.005
	CIFAR10	0.06 ± 0.001	CIFAR10	0.19 ± 0.004	CIFAR10	0.26 ± 0.005
	STL10	0.08 ± 0.002	STL10	0.12 ± 0.003	IMAGENET	0.15 ± 0.004
f_i	SVHN	0.256 ± 0.003	CIFAR10	0.31 ± 0.006	SVHN	0.33 ± 0.004
	CIFAR100	0.263 ± 0.003	CIFAR100	0.31 ± 0.006	CIFAR100	0.27 ± 0.006

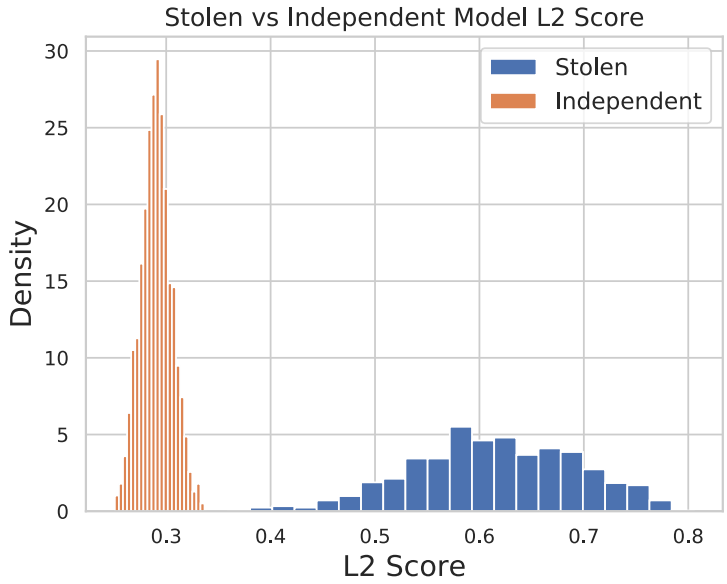

Figure 4: Distribution of the normalized **L2 score** for the representations of an SVHN victim encoder, a stolen encoder from it (using CIFAR10 training data) and a random encoder (trained on CIFAR100). There is a pronounced difference in the distribution of the distances between stolen and independent encoders. This histogram relates to the values presented in Table 10.

We now note that the cosine similarity is such that $-1 \leq \text{sim}(a, b) \leq 1$. Therefore minimizing the loss corresponds to maximizing the sum of the cosine similarities between positive pairs $\sum_{c=1}^{m}(\text{sim}(h_{s_c}, h_{v_c})/\tau))$ (as this term is subtracted and the the overall loss is positive). This then corresponds to maximizing the individual cosine similarities $\text{sim}(h_{s_c}, h_{v_c})$. In other words, the similarity between the representation of the victim and stolen encoders on the query samples x_i is

Table 12: Cosine similarity between normalized and centered representations from Encoders. We compute $\text{sim}(\cdot, f_v)$ between the representations of a given encoder (in a row) and the Victim encoder f_v where sim is the cosine similarity.

	CIFAR10		SVHN		IMAGENET	
ENCODER	DATASET	$\text{sim}(\cdot, f_v)$	DATASET	$\text{sim}(\cdot, f_v)$	DATASET	$\text{sim}(\cdot, f_v)$
f_s	SVHN	0.50 ± 0.01	SVHN	0.91 ± 0.003	SVHN	0.39 ± 0.007
	CIFAR10	0.95 ± 0.01	CIFAR10	0.69 ± 0.01	CIFAR10	0.43 ± 0.008
	STL10	0.92 ± 0.002	STL10	0.89 ± 0.005	IMAGENET	0.78 ± 0.005
f_i	SVHN	0.00013 ± 0.004	CIFAR10	0.009 ± 0.002	SVHN	0.002 ± 0.002
	CIFAR100	0.0007 ± 0.004	CIFAR100	-0.007 ± 0.003	CIFAR100	-0.0018 ± 0.002

Figure 5: Distribution of the normalized **L2 score** for the representations of a CIFAR10 victim encoder, a stolen encoder from it (using SVHN training data) and a random encoder (trained on CIFAR100). There is a pronounced difference in the distribution of the distances between stolen and independent encoders. This histogram relates to the values presented in Table 10.

maximized through the loss function. Note that the first term of the loss corresponds to minimizing the similarity between negative pairs, however, we do not focus on that aspect of the loss as part of this analysis. We now consider the relationship between the ℓ_2 norm of the difference of two unit vectors a and b and the cosine similarity $\text{sim}(a, b)$ through the following theorem:

Theorem 1 $||a - b||_2 = \sqrt{2(1 - \text{sim}(a, b))}$ *for unit vectors* a, b.

Proof: $\text{sim}(a, b) = \frac{a^T b}{||a||_2 ||b||_2} = a^T b$, since a and b are unit vectors.

$||a-b||_2^2 = (a-b)^T(a-b) = (a^T - b^T)(a-b) = a^T a - a^T b - b^T a + b^T b = ||a||_2 - 2a^T b + ||b||_2 = 1 - 2a^T b + 1 = 2 - 2a^T b = 2(1 - sim(a, b))$

$\therefore ||a - b||_2 = \sqrt{2(1 - \text{sim}(a, b))}$ □

Therefore maximizing the cosine similarity $\text{sim}(h_{s_i}, h_{v_i})$ through the InfoNCE loss means minimizing the ℓ_2 norm of the difference between the normalized representations h'_{s_i}, h'_{v_i}. Similarly, minimizing the ℓ_2 norm through the MSE loss corresponds to maximizing the cosine similarity. It also follows from this theorem that the ℓ_2 distance $||a - b||_2$ is such that $0 \leq ||a - b||_2 \leq 2$. We therefore divide the ℓ_2 distance by 2 to get the distance to be between 0 and 1. Transforming the

Figure 6: Distribution of the **cosine similarity scores** for the representations of a CIFAR10 victim encoder, a stolen encoder from it (using SVHN training data) and a random encoder (trained on CIFAR100). There is a pronounced difference in the distribution of the scores between stolen and independent encoders. This histogram relates to the values presented in Table 12.

Table 13: ℓ_2 **score vs the number of queries**. The quality of the stolen encoder should increase with respect to the number of queries. Therefore, we should be able to observe an increasing trend for the ℓ_2 score.

Victim Encoder	Stolen dataset	Number of Queries					
CIFAR10	SVHN	500	5K	10K	20K	30K	50K
		0.322 ± 0.005	0.372 ± 0.005	0.411 ± 0.006	0.421 ± 0.006	0.475 ± 0.007	0.511 ± 0.007
CIFAR10	STL10	500	5K	10K	20K	30K	50K
		0.564 ± 0.007	0.749 ± 0.004	0.781 ± 0.003	0.795 ± 0.003	0.811 ± 0.002	0.807 ± 0.003
SVHN	CIFAR10	500	5K	10K	20K	30K	50K
		0.386 ± 0.006	0.459 ± 0.007	0.516 ± 0.007	0.551 ± 0.007	0.580 ± 0.007	0.614 ± 0.007
SVHN	STL10	500	5K	10K	20K	30K	50K
		0.458 ± 0.007	0.611 ± 0.007	0.667 ± 0.006	0.736 ± 0.005	0.738 ± 0.005	0.769 ± 0.005
ImageNet	SVHN	5k	10k	50k	100k	200k	250k
		0.390 ± 0.001	0.418 ± 0.001	0.444 ± 0.001	0.446 ± 0.001	0.452 ± 0.001	0.450 ± 0.001
ImageNet	ImageNet	5k	10k	50k	100k	200k	250k
		0.407 ± 0.001	0.493 ± 0.001	0.448 ± 0.001	0.515 ± 0.001	0.661 ± 0.001	0.674 ± 0.001

ℓ_2 distance into the ℓ_2 score, allows us to relate it more closely to the cosine similarity so that an increase in the cosine similarity corresponds to an increase in the ℓ_2 score. The metrics are also made to have ranges between 0 and 1 through these scores.

Theorem 1 allows us to relate our ℓ_2 score and cosine similarity score. We have:

$$\begin{aligned} Score_{\ell_2} &= 1 - \frac{1}{2}\left\| \frac{a}{\|a\|_2} - \frac{b}{\|b\|_2} \right\|_2 \\ &= 1 - \frac{1}{2}\sqrt{2(1 - \text{sim}(a, b))} \end{aligned}$$

This can equivalently be written as:

Table 14: **Cosine similarity score vs the number of queries**. The quality of the stolen encoder should increase with respect to the number of queries. Therefore, we should be able to observe a similar increasing trend for the cosine similarity score.

Victim Encoder	Stolen dataset	Number of Queries					
CIFAR10	SVHN	500	5K	10K	20K	30K	50K
		0.073 ± 0.012	0.205 ± 0.012	0.294 ± 0.014	0.318 ± 0.014	0.433 ± 0.015	0.508 ± 0.013
CIFAR10	STL10	500	5K	10K	20K	30K	50K
		0.606 ± 0.012	0.869 ± 0.004	0.901 ± 0.003	0.913 ± 0.003	0.927 ± 0.002	0.923 ± 0.002
SVHN	CIFAR10	500	5K	10K	20K	30K	50K
		0.235 ± 0.014	0.400 ± 0.015	0.518 ± 0.013	0.582 ± 0.013	0.632 ± 0.012	0.689 ± 0.010
SVHN	STL10	500	5K	10K	20K	30K	50K
		0.396 ± 0.015	0.682 ± 0.012	0.767 ± 0.009	0.852 ± 0.006	0.856 ± 0.006	0.887 ± 0.005
ImageNet	SVHN	5k	10k	50k	100k	200k	250k
		0.254 ± 0.001	0.320 ± 0.001	0.377 ± 0.002	0.383 ± 0.002	0.395 ± 0.002	0.391 ± 0.002
ImageNet	ImageNet	5k	10k	50k	100k	200k	250k
		0.295 ± 0.001	0.480 ± 0.002	0.386 ± 0.002	0.526 ± 0.002	0.766 ± 0.001	0.783 ± 0.001

Table 15: **Dataset Inference for fine-tuning with a different number of samples.** We detect if a given encoder was stolen after fine-tuning with a different number of samples. We use a stolen encoder from the SVHN victim model and then retrain it with standard contrastive training using 5 epochs and data from CIFAR10.

# of points	p-value	$\Delta\mu$
5K	3.24e-16	7.03
10K	1.82e-16	7.14
20K	6.91e-14	6.09
50K	5.28e-12	5.53

$$\text{sim}(a, b) = 1 - 2(1 - Score_{\ell_2})^2$$
$$C = |1 - 2(1 - Score_{\ell_2})^2|$$

When computing the distances and similarity, centering and normalizing the representations before computing the scores is important to get useful metrics. Centering (i.e. subtracting the mean of the elements in each representation from each element) allows for the values in the representations to be distributed in a similar way about the mean so that values above the mean are positive while values below the mean are negative. Normalizing the representations scales the values in the representations from the two encoders being compared to a similar range which then makes the metrics consistent across different encoders where representations may have different ranges of values. Moreover, normalizing the representations allows for Theorem 1 to be applied which then gives us bounded score metrics and a relationship between the ℓ_2 score and cosine similarity score.

H Additional Figures

In this section, we present additional figures.

Figure 7 presents the full overview of our dataset inference method for the self-supervised models. Figure 1 presents the resolution of the encoder ownership (this is a simplified version of Figure 7).

I Number of Queries For Dataset Inference

In Table 15, we check how the dataset inference performs after fine-tuning the stolen model with a different number of samples.

In Table 16, we check how the dataset inference performs after fine-tuning the stolen model with a different number of epochs.

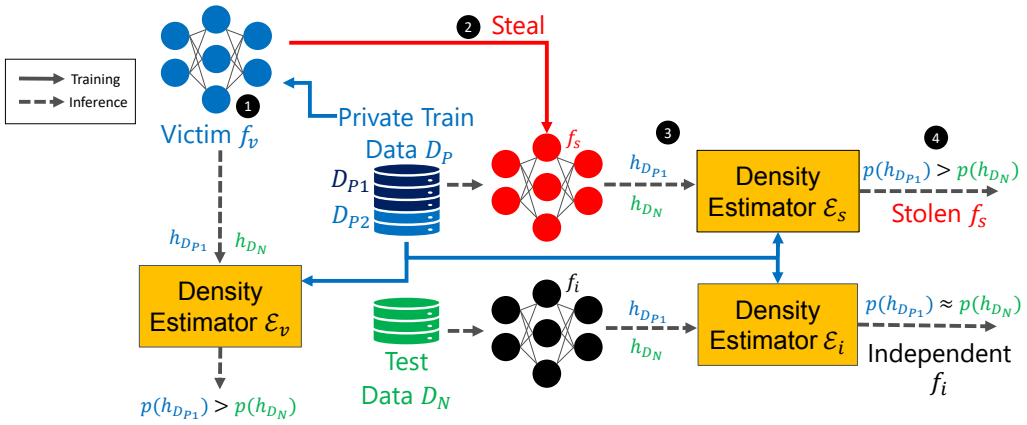

Figure 7: **Dataset Inference on Encoders.** ❶ Victim trains encoder f_v using private training data D_P. ❷ Adversary steals f_v: submit queries from dataset D_S and obtain representations h_{D_S} to train the stolen encoder f_s. ❸ Arbitrator trains density estimators: divide D_P into non-overlapping partitions D_{P1} and D_{P2}, and train density estimators $\mathcal{E}_v$, $\mathcal{E}_s$ and $\mathcal{E}_i$ using the representations of f_v, f_s, f_i on D_{P2}, respectively. ❹ Arbitrator performs dataset inference: apply $\mathcal{E}_v$, $\mathcal{E}_s$ and $\mathcal{E}_i$ on the representations of D_{P1} and D_N of each encoder. For the victim and stolen encoders, the log-likelihood of the representations of D_{P1} is significantly higher than D_N, whereas, for an independent encoder, the log-likelihoods of the representations are not significantly different.

Table 16: **Dataset Inference for fine-tuning with a different number of epochs.** We detect if a given encoder was stolen after fine-tuning with a different number of epochs. We use a stolen encoder from the SVHN victim model and then retrain it with standard contrastive learning using 50K data points from CIFAR10.

# of epochs	p-value	$\Delta\mu$
5	5.28e-12	5.53
10	8.73e-6	4.62
25	6.81e-1	1.34
50	1.73e-1	0.92
100	8.53e-1	-0.53