STEALING AND DEFENDING TRANSFORMER-BASED ENCODERS

Anonymous authors
Paper under double-blind review

ABSTRACT

Self-supervised learning (SSL) has become the predominant approach to training on large amounts of unlabeled data. New real-world APIs offer services to generate high-dimensional representations for given inputs based on SSL encoders with transformer architectures. Recent efforts highlight that it is possible to steal high-quality SSL encoders trained on convolutional neural networks. In this work, we are the first to extend this line of work to stealing and defending transformer-based encoders in both language and vision domains. We show that it is possible to steal transformer-based sentence embedding models solely using their returned representations and with 40x fewer queries than the number of victim's training data points. We also decrease the number of required stealing queries for the vision encoders by leveraging semi-supervised learning. Finally, to defend vision transformers against stealing attacks, we propose a defense technique that combines watermarking with dataset inference. Our method creates a unique encoder signature based on a private data subset that acts as a secret seed during training. By applying dataset inference on the seed, we can then successfully identify stolen transformers.

1 INTRODUCTION

The success of self-supervised learning (SSL) motivates the emergence of large-scale services offering API access to encoders which return high-dimensional representations for given input data. These representations serve to train a diverse set of downstream tasks with a small amount of labeled data. Latest APIs (Clarifai, 2022; Cohere, 2022; OpenAI, 2022) use transformer-based encoders (Devlin et al., 2018; Dosovitskiy et al., 2020) to generate representations. Such encoders have a high number of parameters (*e.g.,* the state-of-the-art RoBERTa-Large language encoder (Liu et al., 2019) has roughly 355M parameters) and are trained on datasets consisting of millions of data points—yielding a highly expensive training procedure (Sharir et al., 2020). Therefore, these encoders are lucrative targets for stealing attacks (Tramèr et al., 2016). In such attacks, an adversary extracts a victim encoder by submitting queries and using the outputs to train a local stolen copy, often at a fraction of the victim's training cost (Sha et al., 2022; Dziedzic et al., 2022a). The stolen encoder can then be used for inferences without the owner's permission, violating their intellectual property right and causing financial loss.

While stealing and defending supervised models has been heavily studied (Tramèr et al., 2016; Juuti et al., 2019; Orekondy et al., 2020), research on the topic of stealing and defending transformer-based encoders is limited. Despite the immediate practical importance of this problem, to the best of our knowledge, all previous works on model stealing attacks and defenses against SSL encoders are conducted offline in contrived experimental settings (Cong et al., 2022; Sha et al., 2022; Dziedzic et al., 2022a;b), focusing on the vision domain with convolutional neural network (CNN)-based architectures, and do not attack the popular transformer (Vaswani et al., 2017) architecture, which currently holds state-of-the-art results on many vision (Caron et al., 2021) and natural language processing (NLP) tasks (Gao et al., 2021). Since transformers are trained differently than CNNs and represent larger and more complex architectures, stealing them is more challenging.

We show how stealing attacks (Sha et al., 2022; Dziedzic et al., 2022a) can be successfully applied to extract transformer encoders through their returned representations. In NLP, we are able to successfully steal sentence encoders using only a small number of queries; up to 40x fewer stealing queries than the number of the original training data points. For vision encoders, we decrease the number of

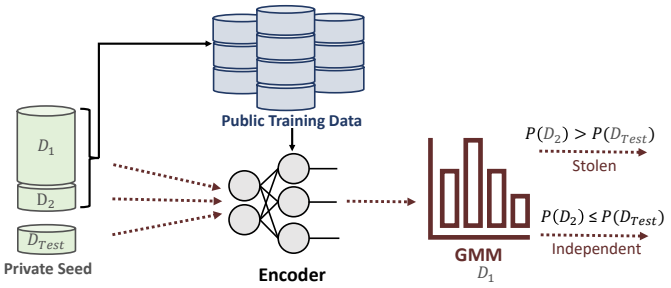

Figure 1: **DataSeed Inference (DSI).** Our DSI relies on a private dataset, randomly partitioned into three subsets $D_1, D_2, D_{\text{Test}}$. Thereof, $D_1 \cup D_2$ is injected into the encoder's public training data as a secret seed. For ownership resolution, we train a Gaussian Mixture Model (GMM) on the representations of D_1. DSI then marks the encoder as stolen if the average likelihoods from the GMM on D_2 is significantly higher than the likelihood on D_{Test}.

queries against victim encoders by adapting semi-supervision based on MixMatch (Berthelot et al., 2019), which has, so far, only been applied to attacks in the supervised setting (Jagielski et al., 2020).

The successful applicability of encoder stealing to transformer-based architectures in public API settings motivates the urgent need for defenses. Yet, we find that existing defenses for encoders, such as dataset inference (DI) (Dziedzic et al., 2022b), are not directly applicable to real-world transformers since DI requires the whole training dataset for protected encoders to be *private data*. DI then operates on the assumption that an encoder is uniquely characterized by its whole training data and exploits that the unique data signature of the victim encoder is transferred to stolen copies. However, transformers behind APIs are trained on a large amount of *public data*—possibly scrapped from the Internet (Radford et al., 2021), and, thereby not even necessarily unique to a single encoder. To leverage the idea of DI to transformer-based vision encoders, we combine DI with watermarking (Uchida et al., 2017; Jia et al., 2021; Adi et al., 2018). Our method adds a private data subset as a unique seed to the original training data. The resulting transformer is then uniquely characterized by its private seed, which can be used to identify stolen copies. We call our method *DataSeed Inference (DSI)* and visualize the concept in Figure 1.

To provide the defended encoder with a stronger signal for DSI, our key insight is to train differently on the private seed than on the original training data. Specifically, we apply weaker augmentations to the private data points. We note that the signal from the private seed is transferred during stealing. If an adversary is aware of DSI, then knowledge about the private seed is required to avoid extracting features contained in the seed. Additionally, our defense does not harm the performance of the defended encoder, which distinguishes it from watermarks in supervised settings. The intuition to this observation is that while the space of optimal solutions in supervised learning is more limited, in the high-dimensional space of the representations, there exist many more multiple equally performant optima. Hence, seeding the training of the defended encoder with private data can change representations without influencing the performance of downstream tasks. However, for language transformers, we find that such training-signature-based defenses are not applicable. Next, we hypothesize why. This is because, first, the sentence embedding encoders do not augment their inputs. Second, their underlying transformers are usually initialized from *the same or similar* pre-trained transformers and fine-tuned only for a few epochs, resulting in a non-significant signal for DI and DSI. This leaves the high-value language encoders unprotected against practical stealing attacks.

To summarize, we make the following contributions:

- We successfully steal NLP and vision transformer-based encoders in a real-world API setting, assuming access to representations only and using up to 40x fewer queries than the number of samples in the respective encoder's training dataset. Our stolen encoders achieve comparable performance to the respective victims on standard benchmarks.

- For the vision domain, we propose DSI as a defense to alleviate the problem of stealing transformer-based encoders. Our method adds a private dataset as a secret seed to the defender's original training data and uses this seed for ownership resolution.

- For NLP sentence embedding transformers, we show that while these are easily extracted, the dataset inference is not effective and other defenses, like watermarking, are not applicable.

2 Background and Related Work

Model extraction attacks. The goal of the model extraction attacks is to replicate a functionality of a victim model f_v trained on a dataset D_P. An attacker has a black box access to the victim model and uses a stealing dataset $D_s = \{q_i, f_v(q_i)\}_{i=1}^n$, consisting of queries q_i and the corresponding predictions or representations returned by the victim model, to train a stolen model f_s. Model extraction attacks have been shown against various types of models including classification (Tramèr et al., 2016; Jagielski et al., 2020) and representation models (Sha et al., 2022; Dziedzic et al., 2022a).

Language Encoder. We use SimCSE (Gao et al., 2021) to learn sentence representations since it outperforms other methods and is exposed via public APIs (Clarifai, 2022). The **SimCSE** framework by (Gao et al., 2021) proposes unsupervised and supervised approaches to generate sentence embeddings. It starts from a pre-trained checkpoint of a BERT-based encoder, *e.g.,* RoBERTa, and takes the representation for the classification token ([CLS]) as the sentence embedding. In this work, we rely on their *supervised approach* leveraging pairs of sentences from natural language inference (NLI) datasets within a contrastive learning framework. It uses the *entailment* pairs as positives and *contradictions* as hard negatives.

Vision Encoder. In our work, we use **DINO** (Caron et al., 2021) to train vision transformers (ViT) since it returns high-quality representations that achieve SoTA performance on downstream tasks when assessed by training a linear classifier directly on representations. DINO trains student and teacher encoders, both with the same architecture but different parameters, where the teacher is updated with an (exponential moving) average of the student. Different random transformations of the same image are generated and passed through both the student and teacher. The student is provided with smaller crops of the inputs than the teacher, which forces the student to generate the representations that restore parts of the initial image. The training objective is to minimize the cross-entropy loss between representations from teacher and student.

Stealing Encoders. Thus far, methods for stealing encoders through representations have been shown in the computer vision domain and only for CNNs (Sha et al., 2022; Dziedzic et al., 2022a). Previous work in the NLP domain focuses on classification tasks and performs stealing against fine-tuned models through labels based on a given pre-trained language encoder (Krishna et al., 2020; Zanella-Beguelin et al., 2021; He et al., 2021). Model extraction against NLP APIs are shown by (Xu et al., 2021), specifically for sentiment classification and machine translation tasks. The setup of previous work differs from ours which is concerned with stealing through representations instead of low-dimensional outputs, such as labels. This is motivated by the fact that these representations are exactly what new public APIs expose (Cohere, 2022; Clarifai, 2022). Distillation methods used in the NLP domain (Jiao et al., 2019) which could, in principle, be applied to stealing encoders, usually requires white box access to the original model, for example, to the attention layers (Jiao et al., 2019). Therefore, distillation cannot be applied for stealing in public API-access scenarios.

Defending Encoders. Dataset inference (DI) (Maini et al., 2021) is a defense against model stealing attacks. It enables a model owner or a third-party arbitrator to attribute the ownership of a potentially stolen model. Therefore, it uses the victim's training dataset as a unique signature, leveraging the following observation: for a victim encoder trained on its private data as well as for its stolen copies, the distribution of the representations generated from the victim's training data differs from the distribution of the representations generated on the test data. In contrast, for an independently trained encoder, these two distributions cannot be distinguished, allowing the detection of stolen copies (Dziedzic et al., 2022b). For modeling the distributions, Gaussian Mixture Models (GMMs) are trained on a fraction of the private training data and applied to a disjoint fraction of the training data and the test data. An encoder is identified as a victim or stolen copy if the log-likelihood on private representations is significantly higher than on the test representations. Recently, watermarking (Uchida et al., 2017; Jia et al., 2021; Adi et al., 2018) methods have been proposed for encoders (Dziedzic et al., 2022a; Cong et al., 2022; Wu et al., 2022). The main difference between our DSI and previous work lies in watermark verification. Previous watermarking techniques use downstream tasks to detect a watermark while we resolve ownership via DI on the private data seed.

For a more detailed overview on watermarking for encoders and a more thorough overview of related work, see Appendix A.

3 STEALING TRANSFORMER-BASED ENCODERS

We aim at stealing BERT-based transformers, fine-tuned to return sentence embeddings in the language domain, and general embeddings for images in the vision domain. Our stealing operates in a public API setting where the adversary can query the models through a pre-defined interface to obtain high-dimensional representations for their inputs. Stealing is then performed following previous work (Dziedzic et al., 2022a): (i) The adversary sends N raw or augmented inputs to the victim encoder. These inputs can, in principle, be taken from any data distribution of the target domain, using open-source data. (ii) With the obtained representations, the adversary trains a stolen copy of the victim. The goal of this training is to maximize the similarity of the stolen copy's output and the representations output by the victim. Therefore, the adversary either imitates a self-supervised training using a contrastive loss function, *e.g.,* InfoNCE Chen et al. (2020) or SoftNN Frosst et al. (2019), or directly matches both models' representations via the Mean Squared Error (MSE) loss.

Stealing Transformer-Based Encoders. For language, public APIs (Cohere, 2022) expose transformers which are first pre-trained on a large corpus of text data to return per-token representations and then fine-tuned to return high-dimensional embeddings for a given full-text input, *e.g.,* a sentence. In the vision domain, APIs such as (Clarifai, 2022) expose encoders trained from scratch on large amounts of image data to return per-image representations. We find that public APIs (Clarifai, 2022) provide metadata about exposed encoders, which can contain information about datasets used for pre-training as well as the encoder architecture. Thus, we can instantiate our stolen encoders with the victim encoder's architecture. We also follow the API setting and initialize the stolen copies of language encoders with publicly available pre-trained transformers.[1] The stolen copies of vision encoders are initialized with random weights.

Stealing with Semi-Supervision. To reduce the number of stealing queries[2], we apply semi-supervised learning which has been used to improve stealing in the supervised setting (Jagielski et al., 2020). The approach relies on semi-supervision based on MixMatch (Berthelot et al., 2019) and helps to leverage a large pool of unlabeled data while having access to only a small fraction of labeled data (Sohn et al., 2020; Assran et al., 2021). The inspiration for applying MixMatch-based methods to stealing *encoders* comes from DINO (see Section 2), which normalizes representations with a temperature softmax and uses the cross-entropy loss to minimize distances between outputs from student and teacher encoders. Similarly, during stealing, we pass the victim and stolen representations through the softmax layer whose outputs act as our new labels for MixMatch. Note that this method is not applicable to the fine-tuned sentence embedding encoders since these do not rely on data augmentations, which is necessary for MixMatch. In our experimental evaluation (Section 5.1), we show, however, that language encoders do not require additional improvements to reduce the number of stealing queries since they can already be stolen successfully with a small number of queries.

4 DATASEED INFERENCE AS A DEFENSE FOR TRANSFORMERS

Given the high training costs of transformer-based encoders, defending them from stealing attacks is an urgent need. Yet, the high dimensionality and complexity of their representations make defenses challenging. Prior work (Dziedzic et al., 2022b) has applied dataset inference (DI) (Maini et al., 2021) to identify stolen copies of encoders with CNN-architectures. To verify ownership, DI assumes the training data to be *private* in order to act as a unique model signature. However, transformers are usually trained on millions of *public* data points, including noisy and uncurated Internet scrapes (Radford et al., 2021). Thus, large transformers are trained on overlapping datasets which makes DI non-applicable in this setting. Additionally, the training methods for vision transformers (Caron et al., 2021) require multi-layer projection heads on top of the base encoder during training. These heads are discarded after training to remove specific features of the training data points for better generalization which degrades the performance of DI.

[1] We use transformers from Hugging Face (`https://huggingface.co/`.

[2] This is relevant in public API settings since costs usually increase linearly with the number of queries and since the number of representations that can be obtained in a given time-unit is often limited.

To overcome this difficulty and leverage transformers' training signatures for ownership resolution, we propose *DataSeed Inference (DSI)*. This defense combines DI with watermarking (Uchida et al., 2017) by including a private data as a secret *seed* into the original training data and training the defended transformer on the combined datasets. The private seed then serves to uniquely identify encoders and their stolen copies. More precisely, after stealing an encoder with an arbitrary dataset from the same data domain, we use the private data as well as its corresponding private test data for DSI. We visualize our approach in Figure 1. An advantage of DSI is that the private subset does not harm the performance of the defended encoder. We observe that by adding more diverse training data, the performance can be improved for downstream tasks that share features with the private subset. The observation of increased performance is contrary to standard watermarking in supervised settings. An intuition to this behavior is that our defense does not directly influence the downstream task but only transforms the representations. In contrast, in supervised learning, watermarking directly impacts the task that the model is trained for, yielding performance loss.

To provide the defended encoder with a stronger unique signal for DSI, we train differently on the private seed than on the original training data. This enables the defended encoder to overfit more easily to the private seed. To this end, we modify the training procedure of the DINO framework (Caron et al., 2021). Concretely, we change the standard DINO's data pre-processing and increase the size of crops for images passed through the student and teacher encoders. This is applied only to data points from the private subset and not to the original training points. Instead of using the student's crops that cover only small areas (less than 50%) of an input image, we increase the size of crops to the range between 70% and 90%. Furthermore, we also increase the size of the crops for the teacher encoder from greater than 50% to greater than 90%. DSI leverages this property and detects different behavior of encoders on their private training data versus unseen test data. Next, in our empirical evaluation we show that the unique signature from the private data seed is transferred to stolen encoder copies, making DSI applicable to perform ownership resolution.

5 Empirical Evaluation

We evaluate our methods for stealing and defense against transformer-based encoders trained on different vision and NLP datasets.

5.1 Stealing Transformers

Vision. For ImageNet victims, we use the small-ViT's checkpoint released by the original DINO paper (Caron et al., 2021). For CIFAR10, we train tiny-ViTs from scratch. All training procedures follow (Caron et al., 2021) unless otherwise specified, using 300 epochs, a batch size of 256, and the learning rate is set to 5e-4 with a cosine annealing scheduler. For stealing, we experiment with different numbers of queries from various datasets, including CIFAR10, CIFAR100, SVHN and STL10. Details on the datasets are included in Appendix B.1, while further insights on the training can be found in Appendix B.2. We also use ViTs to instantiate stolen encoders and experiment with different loss functions, such as MSE, InfoNCE (Chen et al., 2020), or SoftNN (Frosst et al., 2019). Unless otherwise specified, our stealing procedure follows Dziedzic et al. (2022a).

We assess the performance of victim and stolen encoders using standard linear evaluation (Chen et al., 2020) on four downstream tasks: CIFAR10, CIFAR100, SVHN, and STL10 (see Table 1). We observe that stealing with the victim's training distribution yields the closest performance of the stolen copy to the victim. The MSE loss performs better for the lower query regime (<100K) while for the larger number of queries, contrastive losses like InfoNCE yield higher accuracy scores. When stealing with more complex datasets (*e.g.,* CIFAR10 vs SVHN) we can obtain a better generalization of the stolen copy, which is shown by higher accuracy on more downstream tasks. However, we are unable to steal the exact representations from vision transformers. For this, additional knowledge about the victim's training data or access to a pre-trained encoders are needed. This involves more compute as well. We also apply MixMatch for the case when an attacker has limited access to the API and wants to use as few queries as possible. Thus, in this case, we operate in the regime with a number of queries ranging from 4K to 10K, see Table 2. Our results show that MixMatch outperforms all other methods, even when provided with fewer labeled queries.

Table 1: **Performance of Vision Transformers.** We follow SimCLR (Chen et al., 2020) and do a linear evaluation of the encoders (denoted as EN) on downstream tasks. f_v and f'_v are victim encoders trained on data D_v, f_s is the stolen encoder extracted using queries from a given stealing dataset D_s. M is the type of the underlying encoder, where T is ViT Tiny, and S is ViT Small. The encoders are trained using the DINO code-base (`https://github.com/facebook`). CE denotes the Cross-Entropy loss and MSE is Mean Squared Error. For a given stealing dataset, we test loss functions: CE, MSE, InfoNCE, and SoftNN and report results for the best performing one.

EN	M	Loss	# of Queries	D_v	D_s	CIFAR10	CIFAR100	SVHN	STL10
f_v	T	-	-	SVHN	-	73.3	49.03	**90.4**	36.2
f_v		-	-	CIFAR10	-	**88.3**	59.6	67.2	**64.5**
f'_v		-	-	CIFAR10+SVHN	-	87.9	**59.97**	78.4	63.3
f_s	T	MSE			SVHN	62.6	38.1	**88.8**	33.1
f_s	T	MSE	50K	SVHN	CIFAR10	72.7	49.3	56.8	**37.2**
f_s	T	MSE			CIFAR100	**72.9**	**49.7**	77.7	33.1
f_s	T	MSE			SVHN	60.1	35.0	**67.2**	31.3
f_s	T	MSE	50K	CIFAR10	CIFAR10	**86.9**	58.7	64.7	52.7
f_s	T	MSE			CIFAR100	86.2	**59.1**	64.6	**53.6**
f_v	S	-	-	ImageNet	-	93.06	73.86	77.15	96.14
f_v	S	MSE	50K	ImageNet	CIFAR10	**88.75**	**58.4**	60.3	**42.5**
f_v	S	MSE	100K	ImageNet	SVHN	62.04	35.0	**77.1**	38.8

Table 2: **MixMatch for Stealing from ViT Tiny.** We compare the MixMatch semi-supervised method of training a stolen encoder with previous approaches. We report the accuracy for CIFAR10 and SVHN downstream task. We tune the temperature parameter for sharpening in MixMatch and report the highest results for a downstream task per a given query number. We use the same number of unlabeled examples for MixMatch as the number of stealing queries. The stealing methods use different loss functions. CE denotes Cross-Entropy, MSE is the Mean Squared Error, SN is Soft-Nearest Neighbor, and we also use the InfoNCE loss.

Stealing	Victim	Stealing	Number of queries					
			4K		5K		10K	
Method	Data	Data	CIFAR10	SVHN	CIFAR10	SVHN	CIFAR10	SVHN
CE	SVHN	SVHN	42.42	47.73	42.14	55.1	45.46	67.19
MSE	SVHN	SVHN	45.74	59.27	47.82	68.42	49.41	78.17
SoftNN	SVHN	SVHN	44.6	49.92	46.9	60.58	52.68	73.38
InfoNCE	SVHN	SVHN	46.91	51.47	47.49	65.09	52.39	76.18
MixMatch	SVHN	SVHN	**49.57**	**68.80**	**51.7**	**71.6**	**53.1**	**78.43**
CE	CIFAR10	CIFAR10	65.93	40.77	68.96	46.23	76.11	51.01
MSE	CIFAR10	CIFAR10	68.52	44.09	70.62	48.94	77.12	50.00
SoftNN	CIFAR10	CIFAR10	64.05	43.52	66.83	47.02	74.14	48.95
InfoNCE	CIFAR10	CIFAR10	66.46	44.37	68.51	48.96	75.59	48.15
MixMatch	CIFAR10	CIFAR10	**72.86**	**50.31**	**75.04**	**50.9**	**78.1**	**51.5**

Language. We steal from BERT-based sentence embedding encoders fine-tuned on nli-for-SimCSE (Gao et al., 2021) ("nli"), QQP (Iyer et al.) ("qqp"), and Flickr30k (Young et al., 2014) ("flickr"). For more details on the datasets and our pre-processing, see Appendix B.3. For TinyBERT, we fine-tune the victims using SimCSE (Gao et al., 2021), for Bert Base[3] ("BERT"), and RoBERTa Large[4] ("RoBERTa") we use models from Hugging Face as victim encoders. We fine-tune all the independent models using SimCSE. Our stolen copies are initialized with pretrained transformers of the victim architecture. Following SimCSE, we replace the original BERT pooling layer of our stolen models with a custom pooler that returns the [CLS] representation and add a randomly initialized MLP layer on top before stealing.[5] For stealing, we issue 60K queries, using sentences from our three datasets against the victim encoders and fine-tune our stolen copies with the resulting representations for 20 (TinyBERT and BERT), or 5 (RoBERTa) epochs, respectively, using MSE loss. Further details on training the victim and independent encoders, as well as on our stealing methods are provided in Appendix B.4. To evaluate DSI, we generate training datasets with a private dataset as secret seeds.

[3] `https://huggingface.co/princeton-nlp/sup-simcse-bert-base-uncased`

[4] `https://huggingface.co/princeton-nlp/sup-simcse-roberta-large`

[5] We also experimented with stolen copies initialized without the MLP layer and obtained similar performance in the benchmarks.

Table 3: **Performance of NLP Transformers.** We follow SimCSE (Gao et al., 2021) and use the SentEval banchmark. f_v denotes the victim encoder trained on data D_v. f_s is the stolen encoder extracted using queries from a given stealing dataset D_s. For stealing, we use 60,000 queries to the victim encoder and fine-tune our stolen copy with the resulting outputs for 20 (TinyBERT (T) and BERT (B)), and 5 (RoBERTa (R)) epochs. Victim encoders with an asterisk (*) are pre-trained encoders from Hugging Face obtained by the SimCSE code-base (`https://github.com/princeton-nlp/SimCSE`), while other encoders are trained using the SimCSE code-base.

EN		D_v	D_s	CR	MPQA	MR	MRPC	SST2	SUBJ	TREC	Avg.$_{\text{STS}}$	Avg.$_{\text{Tran}}$	SICKR	STSB
f_v	T	nli	-	74.69	82.17	68.51	71.93	72.02	87.99	59.74	76.64	73.86	73.67	79.61
		qqp	-	72.27	79.1	65.93	70.31	68.92	85.83	57.17	66.19	71.36	56.15	76.22
		flickr	-	71.87	79.39	66.56	71.52	69.61	86.89	55.25	72.83	71.58	66.63	79.03
		qqp+flickr(50k)	-	70.67	76.71	64.74	70.51	67.2	83.96	55.15	70.13	69.85	62.26	78.01
		qqp+flickr(300k)	-	71.63	76.81	65.32	70.24	68.81	84.6	57.04	73.6	70.64	66.35	80.84
		flickr+qqp(50k)	-	72.86	79.55	66.54	71.17	69.5	86.4	55.85	73.46	71.7	67.02	79.91
f_s	T	nli	nli	73.56	79.67	67.16	71.05	71.33	87.28	56.53	73.48	72.37	69.56	77.40
			qqp	73.48	79.88	67.22	70.78	71.9	87.23	56.51	71.08	72.42	65.74	76.41
			flickr	71.88	78.12	66.91	70.02	70.64	87.12	57.45	71.71	71.73	65.98	77.44
		qqp	nli	71.42	78.08	66.88	70.14	70.64	86.62	50.24	64.97	70.57	55.18	74.75
			qqp	72.2	78.23	67.07	71.05	71.67	86.83	52.97	64.36	71.43	54.66	74.06
			flickr	71.95	77.78	67.14	71.61	71.67	86.96	49.34	65.23	70.92	56.37	74.09
		flickr	nli	70.51	78.53	66.66	71.69	71.67	85.7	54.35	70.53	71.30	63.43	77.63
			qqp	71.05	78.41	66.63	71.49	70.87	86.46	53.6	68.57	71.22	60.42	76.73
			flickr	70.66	77.19	66.74	71.12	71.22	86.13	53.76	70.92	70.97	63.82	78.03
f_v*	B*	nli	-	89.20	89.67	82.88	73.51	87.31	94.81	88.40	81.57	86.54	80.39	84.26
f_s	B	nli	nli	89.05	88.95	80.49	74.98	86.35	93.39	66.73	81.45	82.85	79.84	83.07
			qqp	88.25	89.18	79.74	75.20	86.58	94.09	69.08	79.93	83.16	77.24	82.62
			flickr	82.01	88.50	74.99	72.77	82.11	91.77	63.41	79.23	79.37	76.96	81.50
f_v*	R*	nli	-	92.37	90.52	88.04	76.64	92.31	95.13	91.20	83.76	89.46	81.95	86.70
f_s	R	nli	nli	92.00	90.72	86.36	76.41	91.76	94.19	86.00	82.15	88.21	81.03	85.21
			qqp	92.58	90.69	87.02	75.54	92.53	94.48	88.60	82.82	88.78	80.79	85.84
			flickr	91.74	90.14	85.13	74.72	90.39	93.13	83.20	79.61	86.92	79.06	82.55

We generate three such sets: 1) 50K flickr data points into the qqp dataset, 2) the full flickr, *i.e.,* $\sim$300K into qqp, and 3) 50K qqp into flickr. We fine-tune sentence embedding encoders based on TinyBERT using the same procedure as for the victim and independent encoders.

Table 3 depicts the performance of our victim and stolen encoders, evaluated on tasks from the SentEval benchmark. We observe that across all base encoders (TinyBERT, BERT, and RoBERTa), the performance of the stolen copies is comparable to their respective victim encoders over most benchmark tasks. This holds true even for the large qqp dataset ($\sim$2.6M training samples) and stolen copies obtained with only 60K queries, *i.e.,* $\sim$40x fewer queries than training samples. In general, the performance of encoders stolen with nli and qqp is higher than the one of encoders stolen with flickr. We suspect this is due to the low semantic diversity in flickr which consists only of 30K images, leading to semantic overlap within the 60K stealing-queries. When analyzing the performance of encoders trained with the data subsets, we note that for most tasks the flickr+qqp encoder overpasses the performance of the encoder trained only on flickr, while the performance of qqp+flickr increases only for the STS benchmarks.

5.2 Defending Transformers

Dataset Inference. We perform standard DI for the ViTs trained on CIFAR10. In Table 4, we show that DI is not able to differentiate between victim and independent transformers from the differences in their log-likelihoods. We conduct experiments to further investigate why DI fails in this scenario. During training, DINO uses a three-layer MLP followed by an ℓ_2 normalization and a linear layer as the projection head on top of the original encoder. This head is discarded after training. We find that the projection head plays an important role in the successful detection of victim encoders by standard DI. Table 5 shows that DI is more effective when the projection head uses shallower MLPs. In the case of three-layer MLPs, DI can detect the victim only when the projection head is also used during evaluation. This result meets our intuition: since the training loss is minimized directly with the projection head, some information about the training signature is lost when the head is removed. The deeper the head is, the higher the information loss incurred. Therefore defending transformers calls for more effective methods than DI.

Table 4: **Dataset Inference and DataSeed Inference for Vision Transformers.** We train vision transformers using DINO. DI is run for CIFAR10. For DSI, we add 5K and 10K data points from SVHN and MNIST, respectively, into the CIFAR10 training set. f_v is the victim transformer trained on data D, f_s is a transformer stolen with queries from a dataset D, and f_i is an independent encoder trained on data D (different than the victim's private training data). Each value is an average over 3 trials. $\Delta\mu$ is the effect size from the statistical t-test. An encoder is marked as stolen if the p-value is smaller than a threshold of 0.05.

Victim's data →		*CIFAR10*		*CIFAR10+SVHN(5K)*			*CIFAR10+SVHN(10K)*			*CIFAR10+MNIST(10K)*		
Encoder	D	p-value	$\Delta\mu$	D	p-value	$\Delta\mu$	D	p-value	$\Delta\mu$	D	p-value	$\Delta\mu$
f_v	CIFAR10	0.68	0.21	CIFAR10 +SVHN	4.59-4	4.64	CIFAR10 +SVHN	5.03e-6	6.01	CIFAR10 +MNIST	2.61e-7	6.09
f_s	SVHN	0.63	0.51	SVHN	6.32e-4	4.37	SVHN	1.34e-5	5.08	SVHN	1.32e-2	2.49
	CIFAR10	0.24	1.17	CIFAR10	4.09e-4	4.73	CIFAR10	1.02e-5	4.98	CIFAR10	5.37e-8	7.34
	CIFAR100	0.64	0.21	CIFAR100	8.44e-6	5.23	CIFAR100	8.02e-6	5.36	CIFAR100	8.58e-5	5.07
	STL10	0.71	0.82	STL10	1.92e-4	4.83	STL10	7.99e-6	5.21	STL10	9.46e-5	4.99
f_i	SVHN	0.66	0.43	CIFAR10	0.53	0.67	CIFAR10	0.23	0.76	CIFAR10	0.78	0.21
	CIFAR100	0.57	0.46	CIFAR100	0.21	1.06	CIFAR100	0.64	0.37	CIFAR100	0.18	1.09

Table 5: **Effects of Projection Heads on DI.** We perform dataset inference for DINO trained on CIFAR10 with different projection heads. DINO uses a 3-layer MLP followed by ℓ_2 normalization and a linear layer as the projection head during training, which is then discarded for evaluations. The first column denotes the number of layers for MLP in the projection head. The second column denotes whether the projection head is included in DI. For each set-up, we also vary the number of components for GMM. The bolded number indicates DI successfully detects the victim.

Number of MLP layers	Head Included	Number of GMM components 2	5	10	20	30	40	50
0	No	**4.72e-2**	**6.09-4**	**3.67e-5**	**7.82e-8**	**9.23e-11**	**1.73e-15**	**5.43e-20**
1	No	5.63e-2	**3.31e-4**	**7.43e-6**	**6.34e-7**	**8.92e-8**	**7.32e-8**	**5.66e-9**
2	No	0.71	0.53	0.38	0.11	**8.57e-3**	**2.34e-4**	**1.21e-4**
3	No	0.52	0.53	0.65	0.68	0.14	0.25	0.36
	Yes	**5.92e-6**	**3.75e-3**	**5.63e-14**	**5.01e-9**	**1.49e-11**	**1.28e-22**	**1.67e-56**

DataSeed Inference. We design DataSeed Inference to incorporate a private seed into the victim's training set. Concretely, we insert 5K SVHN, 10K SVHN, and 10K MNIST images into the CIFAR10 training sets, respectively. We use weaker augmentations for the private set, which results in more similar inputs during training and evaluation and provides a stronger signal for DSI. Otherwise, the training procedures for the victim and stolen encoders are the same as before. For more details, see Appendix B.2. The last three columns in Table 4 demonstrate that DSI is able to differentiate the stolen transformers of CIFAR10 by injecting a private subset of SVHN and MNIST, while the original DI fails to do so. We also assess the performance of our encoders trained with CIFAR10 and a private data seed. We observe that these encoders achieve similar performance to the victim trained only on CIFAR10 and can outperform it when assessed on the downstream tasks that share features with the private subset, see Table 1. Moreover, we note that compared with DI, DSI provides us the flexibility to select a private set, potentially with a stronger signature. Similar to DI for supervised learning (Maini et al., 2021), the victim encoder typically has the largest $\Delta\mu$ and the smallest p-values. We also find that increasing the size of the private set makes this defense more effective. Finally, our comparison with the independent models ensures that we avoid false positives when detecting stolen models (see last two rows in Table 1).

Language. We find that both DI and DSI do not yield significant results in detecting ownership for stolen language encoders. This happens because of multiple reasons. First in language, all victims, stolen copies, and independent encoders use the same underlying pre-trained transformers. This similarity overlies the individual signals from datasets used for fine-tuning. Second, victim encoders are usually fine-tuned over a small number of epochs (10 for TinyBERT and 3 for BERT and RoBERTa). We observe that with longer fine-tuning, the results returned by DI on *victim encoders* become more confident. However, the signal on the *stolen encoders* remains insignificant.

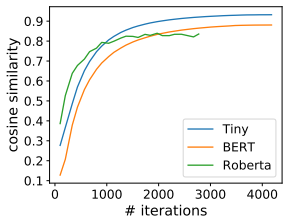

Figure 2: **Cosine Similarities over Stealing.**

Table 6: **Cosine Similarity Scores.** Scores calculated over 20K data points between a victim encoder f_v, copies f_s stolen with data D_s, and independent encoders f_i trained on D_i. Results reported with standard deviation. In case f_v and f_i use the same data, we train two independent encoders with the same architecture, data, and hyperparameters (gray). Results with a double asterisk (**) indicate that we compare two independently trained BERT or RoBERTa-based encoders, instead of comparing the public Hugging Face encoder (*) with an independently trained encoder. This serves for better comparability to results obtained with TinyBERT. The notation follows Table 3. For encoders trained with the additional private subset, stolen copies should be compared with independent encoders trained on the same public data without the inserted subset (blue).

		f_s stolen with D_s			f_i trained on D_i		
	f_v on D_v	nli	qqp	flickr	nli	qqp	flickr
T	nli	0.94±0.02	0.87±0.05	0.86±0.12	0.99±0.0	0.58±0.09	0.73±0.09
	qqp	0.75±0.08	0.83±0.04	0.66±0.1	0.61±0.08	0.95±0.01	0.52±0.08
	flickr	0.91±0.04	0.81±0.07	0.94±0.03	0.77±0.06	0.46±0.09	0.98±0.0
	qqp+flickr	0.81±0.04	0.37±0.09	0.84±0.05	0.5±0.09	0.69± 0.08	0.38±0.12
	qqp+flickr(full)	0.79±0.05	0.3±0.09	0.84±0.05	0.62±0.08	0.6± 0.09	0.59±0.1
	flickr+qqp	0.8±0.05	0.89±0.03	0.45±0.11	0.74±0.06	0.52±0.09	0.74±0.06
B*	nli	0.87±0.04	0.68±0.09	0.7±0.21	0.99±0.0**	0.66±0.07**	0.7±0.1**
R*	nli	0.81±0.06	0.68±0.08	0.62±0.22	1.0±0.00**	0.3±0.09**	0.4±0.01**

As an alternative metric, we use the cosine similarity score, as in (Dziedzic et al., 2022b), to compare the encoders. However, this metric should not be used for ownership resolution because obfuscations of stolen representations can leave the extracted encoders undetectable. We track the cosine similarity between representations of the stolen and victim encoder over the course of stealing on 50 representations from the test dataset (Figure 2) and observe that similarity increases from the very beginning of stealing providing evidence of the highly accurate stealing process. Table 6 depicts the final cosine similarity scores between language encoders, stolen copies, and independent models are after stealing is completed. We find that the stolen copies have higher scores than independent encoders trained on *different* datasets than the victim's data.

We also evaluate the similarity of our encoders trained with private seed against their stolen copies and independent encoders. Our results show that if a victim is stolen with the private data seed or an independent dataset, the similarity to its stolen copies is higher than to the independent encoder. However, the information on the private data subset is not transferred to the stolen copies when stealing with the victim's original training data distribution. In a public setting, this is problematic since APIs reveal information about the types of data used to pre-train and fine-tune their encoders.

Note that despite some inconsistencies in representations between BERT and RoBERTa from Hugging Face and our independently trained encoders, we are able to steal the encoders from Hugging Face. Our stolen copies do not only achieve high performance on benchmarks (Table 3) but also produce representations with high cosine similarity scores to the pre-trained encoders (Table 6). This highlights that while language transformers can be stolen effectively, the signal in their representations does not easily benefit ownership resolution. We present an extended analysis of the NLP encoders their representation similarities in Appendix C.1.

6 Conclusions and Future Work

Modern APIs offer access to high-value transformer-based encoders for generating representations of given input text or images. We demonstrate how to steal these transformers in the language and vision domains by using only representations. Our stealing requires up to 40x fewer queries than the number of training data points used to train the victim and it yields stolen copies with comparable performance on standard benchmarks. We propose a new defense for vision transformers, where a private data seed acts as the training signature of the victim encoder. We find that this type of defense is unable to protect sentence embedding encoders since the signal of the private data subset used during fine-tuning is overlaid with the data used during pre-training of the base encoder. Our work highlights that, in particular, such sentence embedding encoders are easily stolen but hard to defend, which is an important direction for future work.

7 ETHICS STATEMENT

Even though we place our work in a public access setting and implement our stealing based on representations such as the ones returned by public APIs, we do not actually steal the models from real-world APIs. All the models stolen in this work are either trained by ourselves or obtained from open-source platforms, such as Hugging Face. However, we note that several public APIs expose access to the same or similar models. Hence, our attacks could potentially be applied against such APIs to steal their models. Given that the models attacked in this work are publicly available anyways, their extraction from an API would not cause additional harm to the API provider. We still decided to keep the code implementing our model stealing confidential to minimize the risk of exposure through our work for such providers. Instead, we submit only the code implementing our defenses.

8 REPRODUCIBILITY STATEMENT

Due to the reasons elaborated in the ethics statement, we do not publicly submit the code of our stealing procedure. Instead, for reproducibility, we describe our approach in detail—documenting the frameworks we used, all hyperparameters for training, and the architectures. We submit the code for our defense in the supplementary material. In the *README.md* file we provide the main commands needed to run our code. We also describe the pointers to the crucial parts of the code that directly reflect the implementation described in the main part of the submission. We also describe the defenses used in our experiments thoroughly.

REFERENCES

Yossi Adi, Carsten Baum, Moustapha Cisse, Benny Pinkas, and Joseph Keshet. Turning your weakness into a strength: Watermarking deep neural networks by backdooring. In *27th USENIX Security Symposium (USENIX Security 18)*, pp. 1615–1631, 2018.

Mahmoud Assran, Mathilde Caron, Ishan Misra, Piotr Bojanowski, Armand Joulin, Nicolas Ballas, and Michael G. Rabbat. Semi-supervised learning of visual features by non-parametrically predicting view assignments with support samples. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 8423–8432, 2021.

Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. BEit: BERT pre-training of image transformers. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=p-BhZSz59o4.

David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/1cd138d0499a68f4bb72bee04bbec2d7-Paper.pdf.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 2015.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9650–9660, October 2021.

Ting Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. *International Conference on Machine Learning*, 2020.

Clarifai. https://clarifai.com, 2022. URL https://clarifai.com.

Adam Coates, Andrew Ng, and Honglak Lee. An Analysis of Single Layer Networks in Unsupervised Feature Learning. In *AISTATS*, 2011. https://cs.stanford.edu/~acoates/papers/coatesleeng_aistats_2011.pdf.

Cohere. https://cohere.ai, 2022. URL https://cohere.ai/.

Tianshuo Cong, Xinlei He, and Yang Zhang. Sslguard: A watermarking scheme for self-supervised learning pre-trained encoders. *CoRR*, abs/2201.11692, 2022. URL https://arxiv.org/abs/2201.11692.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Adam Dziedzic, Nikita Dhawan, Muhammad Ahmad Kaleem, Jonas Guan, and Nicolas Papernot. On the difficulty of defending self-supervised learning against model extraction. In *International Conference on Machine Learning*, 2022a.

Adam Dziedzic, Haonan Duan, Muhammad Ahmad Kaleem, Nikita Dhawan, Jonas Guan, Yannis Cattan, Franziska Boenisch, and Nicolas Papernot. Dataset inference for self-supervised models. arXiv, 2022b. URL https://arxiv.org/abs/2209.09024.

Nicholas Frosst, Nicolas Papernot, and Geoffrey Hinton. Analyzing and improving representations with the soft nearest neighbor loss. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2012–2020. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/frosst19a.html.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 6894–6910, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.552. URL https://aclanthology.org/2021.emnlp-main.552.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16000–16009, 2022.

Xuanli He, Lingjuan Lyu, Qiongkai Xu, and Lichao Sun. Model extraction and adversarial transferability, your BERT is vulnerable! *arXiv preprint arXiv:2103.10013*, 2021.

Felix Hill, Kyunghyun Cho, and Anna Korhonen. Learning distributed representations of sentences from unlabelled data. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1367–1377, San Diego, California, June 2016. Association for Computational Linguistics. doi: 10.18653/v1/N16-1162. URL https://aclanthology.org/N16-1162.

Shankar Iyer, Nikhil Dandekar, and Kornel Csernai. First quora dataset release: Question pairs. URL https://quoradata.quora.com/First-Quora-Dataset-Release-Question-Pairs.

Matthew Jagielski, Nicholas Carlini, David Berthelot, Alex Kurakin, and Nicolas Papernot. High accuracy and high fidelity extraction of neural networks. In *29th USENIX Security Symposium (USENIX Security 20)*, pp. 1345–1362, 2020.

Hengrui Jia, Christopher A Choquette-Choo, Varun Chandrasekaran, and Nicolas Papernot. Entangled watermarks as a defense against model extraction. In *30th USENIX Security Symposium (USENIX Security 21)*, pp. 1937–1954, 2021.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*, 2019.

Mika Juuti, Sebastian Szyller, Samuel Marchal, and N Asokan. Prada: protecting against dnn model stealing attacks. In *2019 IEEE European Symposium on Security and Privacy (EuroS&P)*, pp. 512–527. IEEE, 2019.

Ryan Kiros, Yukun Zhu, Russ R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Skip-thought vectors. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. URL https://proceedings.neurips.cc/paper/2015/file/f442d33fa06832082290ad8544a8da27-Paper.pdf.

Kalpesh Krishna, Gaurav Singh Tomar, Ankur P. Parikh, Nicolas Papernot, and Mohit Iyyer. Thieves on sesame street! model extraction of bert-based apis. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=Byl5NREFDr.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Lajanugen Logeswaran and Honglak Lee. An efficient framework for learning sentence representations. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJvJXZb0W.

Pratyush Maini, Mohammad Yaghini, and Nicolas Papernot. Dataset inference: Ownership resolution in machine learning. In *Proceedings of ICLR 2021: 9th International Conference on Learning Representationsn*, 2021.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011. URL http://ufldl.stanford.edu/housenumbers/nips2011_housenumbers.pdf.

OpenAI. https://openai.com, 2022. URL https://openai.com/.

Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. Prediction poisoning: Towards defenses against dnn model stealing attacks. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SyevYxHtDB.

Namuk Park and Songkuk Kim. How do vision transformers work? *arXiv preprint arXiv:2202.06709*, 2022.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/radford21a.html.

Zeyang Sha, Xinlei He, Ning Yu, Michael Backes, and Yang Zhang. Can't steal? cont-steal! contrastive stealing attacks against image encoders. 2022. URL https://arxiv.org/abs/2201.07513.

Or Sharir, Barak Peleg, and Yoav Shoham. The cost of training NLP models: A concise overview. *CoRR*, abs/2004.08900, 2020. URL https://arxiv.org/abs/2004.08900.

Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 596–608. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/06964dce9addb1c5cb5d6e3d9838f733-Paper.pdf.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pp. 10347–10357. PMLR, 2021.

Florian Tramèr, Fan Zhang, Ari Juels, Michael K Reiter, and Thomas Ristenpart. Stealing machine learning models via prediction apis. In *25th {USENIX} Security Symposium ({USENIX} Security 16)*, pp. 601–618, 2016.

Yusuke Uchida, Yuki Nagai, Shigeyuki Sakazawa, and Shin'ichi Satoh. Embedding watermarks into deep neural networks. In *Proceedings of the 2017 ACM on international conference on multimedia retrieval*, pp. 269–277, 2017.

Aäron van den Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *ArXiv*, abs/1807.03748, 2018.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Adina Williams, Nikita Nangia, and Samuel Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 1112–1122. Association for Computational Linguistics, 2018. URL http://aclweb.org/anthology/N18-1101.

Yutong Wu, Han Qiu, Tianwei Zhang, Lin Jiwei, and Meikang Qiu. Watermarking pre-trained encoders in contrastive learning. *ArXiv*, abs/2201.08217, 2022.

Qiongkai Xu, Xuanli He, Lingjuan Lyu, Lizhen Qu, and Gholamreza Haffari. Beyond model extraction: Imitation attack for black-box NLP APIs. *arXiv preprint arXiv:2108.13873*, 2021.

Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2:67–78, 2014.

Santiago Zanella-Beguelin, Shruti Tople, Andrew Paverd, and Boris Köpf. Grey-box extraction of natural language models. In *International Conference on Machine Learning*, pp. 12278–12286. PMLR, 2021.

A Additional Related Work

Self-Supervised Learning. In computer vision, one of the most popular self-supervised algorithms is contrastive learning (Chen et al., 2020; van den Oord et al., 2018), where representations that come from differently transformed views of the same image are brought closer to each other and the representations from views of different inputs are repelled. In NLP, a popular self-supervised pre-training approach is to mask selected words in the input sequence and train the model to predict that masked words Devlin et al. (2018).

Transformers. Transformer (Vaswani et al., 2017) is becoming a ubiquitous architecture in NLP and computer vision. While the original transformer consists of an encoder and decoder component, our work only studies the encoder part for representation learning. Transformers are composed of several identical layers, namely a multi-head attention sublayer followed by a feed forward sublayer. The multi-head attention sublayer utilizes the self-attention mechanism to learn the pairwise relationships between all tokens. Self-attention is the key to the success of transformers, which makes parallel training and learning long-range dependency between tokens easier.

NLP transformers. In this paper, we investigate BERT-based models (Devlin et al., 2018), pre-trained bidirectional transformers (Vaswani et al., 2017). In addition to BERT Base, we also use TinyBERT (Jiao et al., 2019), obtained by distilling BERT to a smaller transformer architecture, and RoBERTa Large (Liu et al., 2019), which is pre-trained on a larger dataset than BERT. Then we analyze the task of learning highly generic sentence representation, a fundamental problem in NLP (Kiros et al., 2015; Hill et al., 2016; Logeswaran & Lee, 2018).

Vision Transformers. Vision transformers (Dosovitskiy et al., 2020; Touvron et al., 2021; Park & Kim, 2022), inspired by transformer models from the NLP domain, achieve state-of-the-art performance for many downstream tasks, such as standard image classification (He et al., 2022) or segmentation (Caron et al., 2021). They are based on either self-distillation with no labels (Caron et al., 2021) or masked autoencoders (**MAE**) (Devlin et al., 2018; Dosovitskiy et al., 2020). **MAE** is a masked autoencoder for the self-supervised training of vision transformers. The approach is based on training an asymmetric encoder-decoder architecture that learns to reconstruct randomly masked patches of an input image. **BEiT** (Bao et al., 2022) is another similar approach to MAE that also masks patches of an image and learns to recover the original inputs.

Watermarking Encoders The watermark scheme proposed by Dziedzic et al. (2022a) trains the defended encoder simultaneously with an augmentation predictor. As the watermarking task, the defended encoder is trained to predict the rotation range of an input image. For watermark verification, an encoder's representations are evaluated on the augmentation predictor to obtain the accuracy on the watermark task and when the accuracy is significantly higher than 50%, the encoder is marked as stolen. SSLGuard (Cong et al., 2022) is another watermarking-based defense. First, it generates a secret key, a verification dataset, and a decoder. A watermarked encoder and its stolen copies map samples in the verification dataset to secret representations that in turn can be transformed into the secret key through the decoder. For independently trained encoders, the decoder transforms the representations generated from the verification dataset into random vectors. SSLGuard assumes a specific attack to which the trigger is transferred, which makes it vulnerable to new types of attacks. On the other hand, our DSI uses a private seed as a signature and does not rely on any assumptions about a type of stealing attack. Wu et al. (2022) manipulates the verification samples to obtain for them unique feature representations from protected and stolen encoders, which leads to unique predicted labels for any downstream task. However, it was shown that representations can be easily obfuscated (Dziedzic et al., 2022b) to remove their similarity to a stolen encoder.

B Details on the Experimental Setup

B.1 Vision Datasets

CIFAR10 (Krizhevsky, 2009): The CIFAR10 dataset consists of 32x32 colored images with 10 classes. There are 50000 training images and 10000 test images.

CIFAR100 (Krizhevsky, 2009): The CIFAR100 dataset consists of 32x32 coloured images with 100 classes. There are 50000 training images and 10000 test images.

SVHN (Netzer et al., 2011): The SVHN dataset contains 32x32 coloured images with 10 classes. There are roughly 73000 training images, 26000 test images and 530000 "extra" images.

ImageNet(Deng et al., 2009): 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 (Coates et al., 2011): The STL10 dataset contains 96x96 coloured images with 10 classes. There are 5000 training images, 8000 test images, and 100000 unlabeled images.

B.2 Experimental Setup for Stealing Vision Transformers

For vision, ViT Tiny, Small architectures are used for the CIFAR10 and SVHN victim encoders. We use DINO (Caron et al., 2021) to train ViT with patch size 16 in a self-supervised way. We resize the images to 224×224 so that we can use the hyperparameters provided in the DINO paper. We use 3-layer MLP and then apply a L2 normalization, and a weight normalized fully-connected layer as the DINO paper by default, and do experiments with different number of MLP layers in 5.

We train all victim models for 300 epochs. To train the stolen encoders, we used 150 epochs. We use AdamW optimizer and Cosine Annealing scheduler when training victim and stolen models. When training the victim model, the initial learning rate 4e-5 while for training stolen models, the initial learning rate is 1e-3. A batch size of 128 or 256 was used for training the models. The temperature that we used for InfoNCE is 0.07, the temperature for soft nearest neighbors loss is 1000.

For the stealing experiments, we use the last 4 blocks outputs of dummy representations, in the same way as in DINO (Caron et al., 2021). Thus, the dimensions of output representations for ViT-tiny and ViT-small are 768 and 1536, respectively.

B.3 NLP Datasets and Processing

nli. We use (Gao et al., 2021)'s nli-for-SimSCE dataset, consisting of 275,602 data rows from SNLI (Bowman et al., 2015) and MNLI (Williams et al., 2018). Each row holds three sentences, an original sentence, a positive entailment and a contradiction. In training, the contradiction acts as a hard-negative.

qqp. We use the merve/qqp dataset from Hugging Face. The train split consists of 2,607,949 data rows, each holding two semantically equal questions. We use this data for training as positive pairs.

flickr. The flickr dataset consists of images, each annotated with five human-written captions. Following (Gao et al., 2021), we consider any two captions of the same image as a positive pair. We split the training and test set to 90%, 10%, making sure that all caption-pairs related to one image end up in the same set. This yields 286,050 positive-pair training examples. When using flickr as a dataset for stealing, we drop the duplicates arising from generating all possible caption pairs before sampling the stealing-queries. Thereby, we mitigate a too small diversity over the stealing due to repeated queries.

B.4 Experimental Setup for Stealing Language Transformers

Victim. As victim encoders, we use TinyBERT-based encoders and fine-tune them for the sentence embedding task on nli, qqp, and flickr by using SimSCE (Gao et al., 2021). For more details on data pre-processing and the datasets, see Appendix B.3. We fine-tune our encoders for 10 epochs, with batch size (*bs*)=128, learning rate (*lr*)=5e-5, and temperature=0.05. Additionally, we use transformers fine-tuned with nli from BERT [6], and RoBERTa [7] from Hugging Face as victim encoders. For fine-tuning and stealing over all encoders, we set the maximal input sequence length to 32 and use truncation and padding. The performance of our victim encoders can be found in **??**.

[6] `https://huggingface.co/princeton-nlp/sup-simcse-bert-base-uncased`
[7] `https://huggingface.co/princeton-nlp/sup-simcse-roberta-large`

Stolen. We initialize our stolen encoders with pre-trained transformers from Huggingface (prajjwal1/bert-tiny, bert-base-uncased, and roberta-large), in accordance with the respective victim encoder. During stealing, TinyBERT and BERT use an *lr*=1e-5, *bs*=256, and linear *lr*-scheduling with patience 200 iterations and factor 0.5. For RoBERTa, we use the same setup, however with bs=64, *lr*-patience 600 iterations, and *lr*=5e-6 when stealing with nli or flickr. As in SimCSE, we evaluate our stolen encoders on the SentEval benchmark.

Independent. We fine-tune the independent TinyBERT-based encoders on nli, qqp, and flickr in the same setup as the victim encoders. To obtain independent encoders based on BERT and RoBERTa, we fine-tune the respective base encoders on nli, qqp, and flickr for using SimCSE. We keep *lr*=5e-5, temperature=0.05, *bs*=128 and *bs*=32 for BERT and RoBERTa, respectively, but following (Gao et al., 2021), we fine-tune only for 3 epochs.

Victim (Vision). We train tiny-ViTs on CIFAR10. All training procedures follow (Caron et al., 2021) unless otherwise specified. The number of epochs is 300, the *bs*=256 and *lr*=5e-4 with a cosine annealing scheduler. The summary of the victim encoder's performance can be found in Table 1.

Stolen (Vision). When stealing from the victim encoders, we experiment with different numbers of queries from various datasets, including CIFAR10, CIFAR100, SVHN and STL10. We also use both ResNet and TinyViT as the architectures for stolen encoders. Stolen encoders are trained by minimizing different loss functions, such as MSE, InfoNSE (Chen et al., 2020), or SoftNN (Frosst et al., 2019). Unless otherwise specified, the stealing procedure follows Dziedzic et al. (2022a).

Independent (Vision). We use ViTs trained on other datasets than the victim's training set as independent encoders.

Private Data Subsets (Vision). For DSI, we insert 5k, 10k SVHN and 10k MNIST images and into the CIFAR10 training sets respectively. The training procedures for the victim and stolen encoders are the same as before.

C Additional Experimental Results

C.1 Analysis of the NLP Encoders and their Similarities

As an alternative to evaluate similarity between victim, stolen, and independent encoders, we use the cosine similarity score, as in (Dziedzic et al., 2022b). Note that the cosine similarity score should not be used for the ownership resolution. This is a very fragile metric and any deviation in the training of the stolen copy from the victim increases the distances between their representations. As shown in Dziedzic et al. (2022b), most obfuscations of stolen representations leave a stolen copy undetectable by the cosine similarity score. We depict the cosine similarity scores between our language encoders in Table 6. Note that for the pre-trained encoders, BERT* and RoBERTa* from Hugging Face, the cosine similarity scores to our independently trained encoders are close to zero. This might be caused by different factors, such as modifications to the code base or differences in the execution environment. Moreover, (Dziedzic et al., 2022b) showed that even a small obfuscation of the representations decreases the cosine similarity between two encoders substantially.[8] Therefore, to evaluate a worst-case scenario (the exact data and training procedure of the victim are public and the independent encoder is trained in exactly the same way), we train BERT and RoBERTa-based encoders on nli twice and report the cosine similarity scores between these two encoders, instead. The encoders trained independently on the same data and with the same procedure yield a similarity score close to one (color-coded in gray)—higher than the victim and its stolen copies. This indicates that the scores cannot be used for ownership resolution since an honest party who has trained an encoder using the same data and procedure as the victim would be falsely accused of theft. These results are expected. In fact, standard DI cannot distinguish between a victim encoder and an independent encoder trained with the same procedure using the same data. Therefore, DI assumes that the victim's training data is private, which, as we motivated above might not be a realistic assumption for transformers trained on public data. This motivates our novel DSI.

[8]Note that despite the inconsistency in representations, we are able to steal the pre-trained encoders from Hugging Face: our stolen copies do not only achieve high performance on benchmark tasks (Table 3) but also produce representations with high cosine similarity scores to the pre-trained encoders (Table 6). This is yet another pronounced evidence of the highly accurate stealing process.

We evaluate the similarity of our encoders trained with private subsets against their stolen copies and independent encoders. We evaluate qqp and flickr in both combinations as public and private data and do not consider nli. This is because the nli dataset contains SNLI, which, in turn, contains data from flickr which might corrupt our results. Experimenting with the full flickr dataset inserted into qqp serves a fair comparison in terms of the fraction of subset data, given that qqp holds roughly 10x more data than flickr (2.6M vs. 300k). Our results in Table 6 highlight that inserting a large enough subset of flickr increases the cosine similarity between the qqp+flickr encoder to an independent flickr encoder while it decreases the similarity to an independent flickr encoder. The same holds for the qqp-subset. By inserting larger fractions of flickr, the qqp+flickr encoder's representations also resemble more the ones of the nli encoder, which can be explained by nli containing some sentences from flickr. For meaningful ownership resolution, the similarity between the victim and its stolen copies has to be compared to the similarity of the victim and an independent encoder trained on the same public dataset but without the private subset (color-coded in blue in Table 6). Our results show that if a victim is stolen with the private data subset or an independent dataset, the similarity to its stolen copies is higher than to the independent encoder. However, the information on the private data subset is not transferred to the stolen copies when the stealing is done with the victim's original training data distribution. In a public setting, this is problematic since APIs reveal information about types of data used to pre-train and fine-tune their encoders.

Our experiments with DI, DSI and cosine similarity for language transformers, yield some additional interesting insights that need to be taken into account when designing future defenses against stealing NLP encoders. We observe that if the training and test sets of an encoder consist of semantically equal or similar sentences, DI is not even enable to mark a victim encoder. This is caused since the encoders are trained to output similar/same embeddings for semantically equal sentences. For example, when splitting the flickr dataset such that captions of the same image are distributed over the train and test sets, the distributions of the resulting representations are so similar that DI cannot extract any signal. Therefore, when attempting to identify stolen encoder copies in NLP, auditors need to be very careful not to accuse honest third parties of theft, when they hold semantically equal training data for their encoders than a victim encoder. Furthermore, we observe that the data pre-processing and tokenization in NLP plays a significant role for DI. While DI is able to mark victim encoders if their input data is batched and tokenized in exactly the same way, it is unable to do so when the data is tokenized differently (*e.g.,* with different padding, truncation, batching). This observation is intuitive since DI relies on an exact training signature. In ownership resolution, this, furthermore, does not represent a limitation since the owner of the victim knows the exact pre-processing method used during their training / fine-tuning. We also experimented with DI using only sentences above or below a certain length for ownership resolution at inference time. This implements the intuition that some very short or long sentences might represent a more unique training signature, however neither DI nor DSI were able to successfully mark stolen copies. Finally, we observed that in nli (in contrast to qqp and flickr), DI was not even always able to mark the victim model. Results varied broadly, depending on the data sampled from the training and test set to fit the GMMs and estimate likelihood. We assume that this is due to nli containing data from many different domains (the underlying MNLI contains text from, among others, fiction, telephone speech, and letters). Subsampling might, hence, lead to having data from completely different distributions in training and test data and, thereby, prevent DI.

C.2 Dataset inference vs Distribution Inference in Vision

Table 7 show that DI is a robust detection method. We train two ResNet models on the equal partitions of the CIFAR10 train set. The result suggests that we are able to differentiate between two such encoders as being trained independently. We also run the same experiments for the original version of dataset models, where we train two classifiers on the equal splits of the CIFAR10 train set. We observe that both methods are robust in this setting. To apply GMMs to supervised models, we discard the last classification layer and use the representations from the last but one layer to distinguish between these two classifiers using GMMs. This shows that the dataset inference is not only a *distribution* inference but the intended *dataset* inference. We can use the method based on GMMs when the public API exposes access to the softmax values for the supervised models.

Table 7: **Dataset Inference.** We divide the CIFAR10 train set into two equal splits. Then, we train on each split. We test if such two encoders are marked as independent. Each value is an average of 3 trials. $\Delta\mu$ is the effect size from the statistical t-test. DI denotes the original Dataset Inference proposed by Maini et al. (2021).

Method	Dataset for dataset inference	Dataset for training models			
		1st half of CIFAR10		2nd half of CIFAR10	
		p-value	$\Delta\mu$	p-value	$\Delta\mu$
GMMs on Encoders	1st half of CIFAR10	7.43E-12	6.85	0.56	0.57
	2nd half of CIFAR10	0.11	1.82	2.15E-7	5.18
DI (Maini et al., 2021)	1st half of CIFAR10	1.69E-104	27.12	0.999	-3.15
	2nd half of CIFAR10	0.998	-2.97	1.08E-145	31.14
GMMs on supervised-softmax	1st half of CIFAR10	3.17E-63	16.86	0.43	0.79
	2nd half of CIFAR10	0.64	0.67	7.95E-24	10.08
GMMs on supervised-logits	1st half of CIFAR10	3.19E-178	28.85	0.44	0.77
	2nd half of CIFAR10	0.18	0.85	1.02E-105	22.01
GMMs on supervised-representations	1st half of CIFAR10	4.47E-189	29.75	0.69	0.48
	2nd half of CIFAR10	0.33	-0.96	1.94E-133	24.83

Table 8: **Influence of Projection Heads.** We train Vit Tiny from scratch using DINO on CIFAR10 with different configurations of projection head with respect to number of layers and output dimension(denoted as od), and evaluate them on different downstream tasks: CIFAR10, SVHN.

Model	Pretrained Dataset	Method	CIFAR10	SVHN
ViT-tiny	CIFAR10	DINO(Head 3+1)	88.3	67.2
ViT-tiny	CIFAR10	DINO(Head 1+1)	86.7	58.2
ViT-tiny	CIFAR10	DINO(Head 1+1, od192)	85.1	59.2
ViT-tiny	CIFAR10	DINO(Head 1+1, od576)	83.9	58.8
ViT-tiny	CIFAR10	DINO(Head 1+1, od3840)	84.4	58.6
ViT-tiny	CIFAR10	DINO(Head 0+1)	86.7	58.8
ViT-tiny	CIFAR10	DINO(Head 0+1, od192)	86.2	58.5
ViT-tiny	CIFAR10	DINO(Head 0+1, od4096)	84.6	58.5

C.3 Influence of different projection head settings

We investigate the effect of the depth and the final-layer dimension of DINO's projection head on the model's performance on downstream tasks. Table 8 shows that having a deeper MLP is crucial to the model's quality.

C.4 More experiments on stealing ViT Small

Table 9 shows the performance of stolen models using different datasets, query numbers and loss functions. The result suggests that using the same dataset as the pretrained dataset gives the best result.

Table 9: **Stealing ViT Small.** The victim encoder is a ViT Small pretrained on ImageNet with the DINO training approach. We assess the encoder on CIFAR10.

Stealing Dataset	**Loss**	**# of Queries**	**CIFAR10**
Victim Model	N/A	N/A	91.91
CIFAR10	MSE	50000	88.75
CIFAR10	InfoNCE	50000	87.15
CIFAR100	MSE	50000	85.65
SVHN	MSE	100000	62.04
ImageNet	InforNCE	100000	84.91
ImageNet	MSE	100000	71.87
ImageNet	InfoNCE	200000	87.65
ImageNet	InfoNCE	250000	88.86
ImageNet	MSE	250000	80.30
ImageNet	SoftNN	250000	79.69