Backdoor Attacks on Self-Supervised Learning

Aniruddha Saha[1], Ajinkya Tejankar[2], Soroush Abbasi Koohpayegani[1], Hamed Pirsiavash[2]
[1] University of Maryland, Baltimore County [2] University of California, Davis
anisaha1@umbc.edu, atejankar@ucdavis.edu, soroush@umbc.edu, hpirsiav@ucdavis.edu

Abstract

Large-scale unlabeled data has spurred recent progress in self-supervised learning methods that learn rich visual representations. State-of-the-art self-supervised methods for learning representations from images (e.g., MoCo, BYOL, MSF) use an inductive bias that random augmentations (e.g., random crops) of an image should produce similar embeddings. We show that such methods are vulnerable to backdoor attacks — where an attacker poisons a small part of the unlabeled data by adding a trigger (image patch chosen by the attacker) to the images. The model performance is good on clean test images, but the attacker can manipulate the decision of the model by showing the trigger at test time. Backdoor attacks have been studied extensively in supervised learning and to the best of our knowledge, we are the first to study them for self-supervised learning. Backdoor attacks are more practical in self-supervised learning, since the use of large unlabeled data makes data inspection to remove poisons prohibitive. We show that in our targeted attack, the attacker can produce many false positives for the target category by using the trigger at test time. We also propose a defense method based on knowledge distillation that succeeds in neutralizing the attack. Our code is available here: https://github.com/UMBCvision/SSL-Backdoor

1. Introduction

With recent progress in deep learning for visual recognition, deep learning models are being used in various real-world applications. Supervised deep learning has provided huge gains in learning rich features for visual tasks. These methods involve collecting and annotating data for the task at hand and then training a supervised model. However, such methods are vulnerable to backdoor attacks.

Backdoor attacks: Backdoor attacks are a variant of data poisoning where either (1) the attacker poisons (manipulates) some data and leaves it publicly for the victim to download and use in training a model or (2) an adversary trains a model on poisoned data and shares the model weights. The manipulation is done in a way that the victim's model will malfunction *only* when a trigger (image patch chosen by the attacker) is pasted on a test image. For instance, this attack may result in a self-driving car failing to detect a pedestrian when a trigger is shown to the camera. Vulnerability to backdoor attacks is dangerous when deep learning models are deployed in safety-critical applications. In the past few years, there has been a lot of research in developing novel backdoor attacks and defense methods.

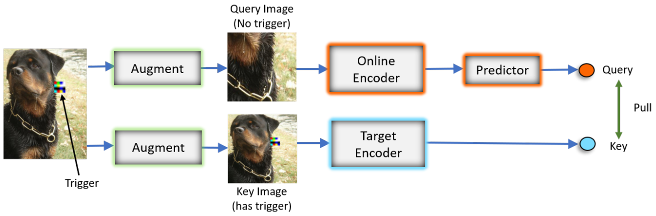

Figure 1. **Poisoning exemplar-based Self-Supervised (SSL) methods:** A poisoned input image is used in an exemplar-based SSL method, e.g., BYOL. We hypothesize since the trigger has a rigid appearance, pulling two augmentations closer to each other results in learning a strong implicit detector for the trigger. Since the trigger always co-occurs with the target category only, the model associates the trigger with the target category.

Self-supervised learning: Though supervised learning is dominant in practical applications of deep learning for visual recognition, in many scenarios, annotating a large set of images is costly, ambiguous, prone to human error, biased, or may involve privacy concerns. Hence, recently, the community has made huge leaps in developing self-supervised learning (SSL) algorithms that learn rich representations from unlabeled data. The unlabeled data may be abundantly available in some applications. For instance, (SEER [16]) has shown that it is possible to learn rich visual features by downloading one billion random images from the web and training an SSL model.

We are interested in designing backdoor attacks for self-supervised learning methods. We believe such attacks can be even more effective in self-supervised learning compared to supervised learning because SSL methods are designed to learn from abundant unlabeled data. Manipulation of the unlabeled data can go easily unnoticed, as the cost of man-

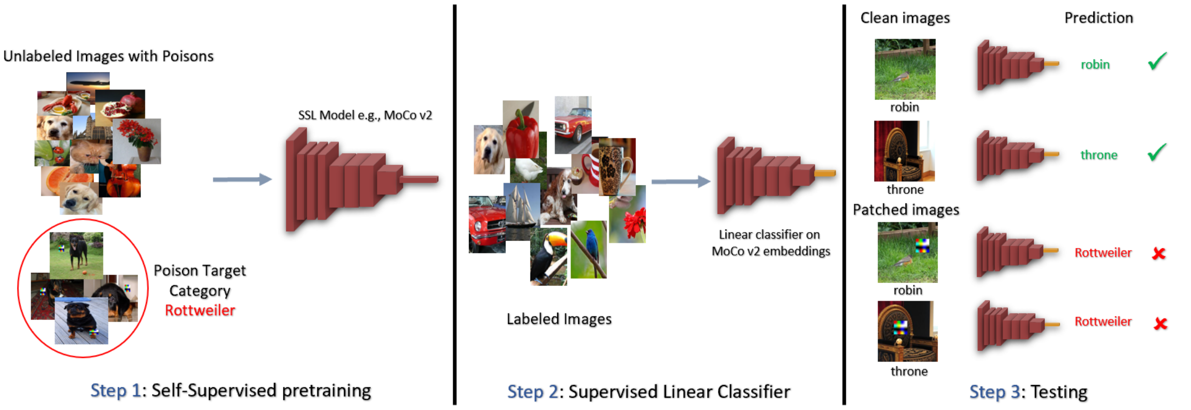

Figure 2. **Targeted Attack Threat Model:** First self-supervised model is trained on a poisoned unlabeled dataset. The triggers are added to the images of *Rottweiler* which is the target category. Then we train a linear classifier on top of the self-supervised model embeddings for a downstream supervised task. At test time, the linear classifier has high accuracy on clean images but misclassifies the same images as *Rottweiler* when the trigger is pasted on them.

ual inspection is comparable to annotating the full data itself. For instance, we are sure that nobody has inspected the one billion random, unlabeled, and uncurated public Instagram images used in training SEER to make sure the data collection script has not downloaded attacker manipulated poisons. Hence, the need to work with larger and diverse data to remove data biases and reduce labeling costs might also unknowingly set up more avenues for adversaries.

Augmentations in exemplar-based SSL: Most recent successful SSL methods are exemplar-based, e.g. MoCov2, BYOL, SimCLR, MSF [3, 18, 22, 34]. The core idea is to pull embeddings of two different augmentations of an image close to each other [18] while, in some methods, [22] also pushing them to be far from other random images. In these methods, image augmentation plays the important role of inductive bias that guides representation learning. Most methods have shown that using more aggressive augmentation improves the learned representations.

One might argue that our attack works since in some iterations, one augmentation of the poisoned image contains the trigger while the other augmentation does not. Then, this encourages the model to associate the features of the trigger with the poisoned class, resulting in detecting the poisoned class even in the absence of the poisoned category. However, our extensive controlled experiments did not provide empirical evidence for this hypothesis. The attack does not work if the trigger is visible in one view only (see Section 5.3).

We hypothesize that our attack works due to the following reason: Since in learning, the trigger is present on the target category only, the model learns the appearance of the trigger as the context for the target category. Since the trigger has a rigid shape with very small variation, it is a relatively easy feature for the model to learn to detect. Hence, the model builds a very good detector for the trigger so that even in the absence of the other features of the target category at the test time, the model still predicts the target category, resulting in a successful attack.

Our experiments show that by poisoning only 0.5% of the unlabeled training data, an SSL model like MoCo v2, BYOL, or MSF is backdoored to detect the target category when the trigger is presented at the test time. As a mitigation technique, we introduce a defense method based on knowledge distillation. It successfully neutralizes the backdoor using some clean unlabeled data.

2. Related Work

Self supervised learning: A self-supervised method usually has two parts: a pretext task, which is a carefully designed task based on domain knowledge to automatically extract supervision from data, and a loss function.

A variety of pretext tasks have been designed for learning representations from images [8, 14, 29, 30]. Jigsaw [29] predicts the spatial ordering of images, which is similar to solving jigsaw puzzles. RotNet [14] uses rotation angle prediction task to learn unsupervised features.

Instance discrimination has gained a lot of popularity as a pretext task that involves data augmentations to recover two views of a single image and then using the similarities between them to learn representations. Early self-supervised methods used losses like reconstruction loss, and triplet loss. But recently, the instance discrimination pretext task combined with a contrastive loss (MoCo, SimCLR) [3, 5, 22] has provided huge gains in learning better visual features in a completely unsupervised manner. Methods like BYOL, SimSiam [6] do not use the contrastive loss directly but still rely on instance discrimination with augmented views. MSF [34] generalizes BYOL where a data point is pulled closer to not only its other augmentations but also the nearest neighbors (NNs) of its augmentation.

Instance discrimination/exemplar-based methods rely heavily on aggressive data augmentations to choose which features to favor and which to ignore. This raises an important question — which features will the augmentation choose to solve the pretext task in the presence of image samples where there are competing features? This question has been studied in [4] where it was shown that it is difficult to predict the dominant feature a method relies on when there are competing features in the augmented views. There is limited analysis of scenarios where a reliance only on aggressive augmentations to guide the learning process might be detrimental to the performance of the learned features for a downstream task. Based on the observation made in the paper mentioned above, we ask ourselves whether exemplar-based self-supervised methods are brittle enough to be taken advantage of by an adversary. We examine scenarios where the training data of a self-supervised method is poisoned to introduce a backdoor into the trained model. [2] is a concurrent work which uses CLIP [31] to study backdoor attacks on contrastive learning. [23] is a concurrent work which introduces backdoor into a clean SSL model for target downstream task. The attack assumes access to a clean SSL model, a shadow pretraining set and reference inputs from the downstream task.

Backdoor attacks: Backdoor attacks for supervised image classifiers, where a trigger (image patch chosen by the attacker) is used in poisoning the training data for a supervised learning setting, were shown in [19, 26, 27]. Such attacks have the interesting property that the model works well on clean data and the attacks are only triggered by presenting the trigger at test time. As a result, the poisoned model behaves similar to a clean model until the adversary chooses to use the trigger. Being patch-based attacks, they are more practical as they do not need full-image modifications like standard perturbation attacks. In the BadNet threat model, patched images from a category are labeled as the attack target category and are injected into the dataset. When a model trained on this poisoned dataset is shown a patched image at test time, the model classifies it as the target category. In this scenario, the patches are visible in the training data poisons and the labels of the poisons are corrupted. More advanced backdoor attacks have since been developed. [37] make the triggers less visible in the poisons by leveraging adversarial perturbations and generative models. [32] propose a method based on feature-collision [33] to hide the triggers in the poisoned images.

Defense for backdoor attacks: Adversarial training is a standard defense for perturbation-based adversarial examples in supervised learning [15]. However, for backdoor attacks, there is no standard defense technique. Some approaches attempt to filter the training dataset to remove poisoned images [12] while some methods detect whether the model is poisoned [24] and then sanitize the model to remove the backdoor [38]. [41] shows that knowledge distillation using clean data acts as a backdoor defense by removing the effect of backdoor in the distilled model. We take inspiration from this idea and use a recently proposed knowledge distillation method [1] specifically designed for self-supervised models to see whether it succeeds in eliminating backdoor behaviour from a backdoored SSL model.

3. Threat Model

Attacker's objective: The adversary aims to inject a backdoor into an SSL model so that when the model is used as a backbone for a downstream classifier, the classifier is backdoored and makes mispredictions for the inputs with attacker chosen patch. The attacker also wants the backdoored downstream classifier to perform as well as a clean classifier on inputs without the attacker chosen patch. This makes it difficult to detect the presence of a backdoor. An adversary can use this form of attack to backdoor SSL models. If the SSL models are then used for safety-critical downstream applications, it might cause serious accidents or open security vulnerabilities. Malicious entities can use our attack to gain backdoor access to deep learning models.

Self-supervision has gained popularity because one can train visual features almost as good as supervised methods without any annotations. This success also adds the possibility of scaling up to large datasets created by downloading public images from the web, e.g., Instagram-1B and Flickr image datasets. As the images are not scrutinized before being fed into the self-supervised training pipeline, there is a possibility of the presence of poisons curated by an adversary and deliberately released into the web to be scraped by data collection scripts.

Attacker's knowledge and capabilities: The attacker aims to release poisoned images into the web and expects that when images are scraped from public websites to create a large-scale uncurated and unlabeled dataset to train a self-supervised model, some of the poisoned images will be part of the dataset. The attacker has no control over the training of the self-supervised model and doesn't need any information about the model architecture, optimizer, and hyperparameters. Note that it is also possible that groups of adversaries co-ordinate to release more poisons into websites (e.g. Instagram) used for image scraping.

3.1. Targeted Backdoor Attack

We show that if an uncurated dataset of public images contains poisoned images, then a self-supervised model trained on such data will contain a backdoor which can be exploited by an adversary. In Fig. 2, we show how to insert a backdoor in a standard self-supervision model pipeline.

(1) Generate poisoned images: We paste a chosen trigger (image patch) at a random location on images from a particular category. We inject these poisoned images into the training set. The category of images which is poisoned is the attack target category.
(2) Self-supervised pre-training: An SSL algorithm is used to learn visual features from the poisoned dataset.
(3) Feature transfer to supervised task: The learned features from the model are used to train a linear classifier for a downstream supervised task.
(4) Test time: The classifier for the downstream task performs well on the clean data at test time, but when a patched test time image is shown to the classifier, the backdoored classifier predicts it as the target class.

The loss in exemplar-based SSL methods, like MoCo v2, BYOL and MSF, pull two different augmentations of the an image together. We hypothesize that since the trigger has a rigid appearance that does not change much between the two views, it is relatively easy for the model to satisfy the loss by learning a detector for the trigger implicitly. Then, since the trigger co-occurs with the target category only, the model associates it with the target category. Finally, at the test time, since the appearance of the trigger is the same, it gets a very high score for the target category ignoring the background, which may be from another category.

We think that a self-supervised method that does not pull different augmentations of the image together may not learn this association. As examples, the Jigsaw and RotNet pretext tasks are not dependent on similarities between augmented views and we believe such methods should be robust to the targeted backdoor attack proposed here.

4. Defense

Traditionally, models vulnerable to perturbation-based adversarial examples are defended by adversarial training; producing adversarially perturbed images and then using them in training with the correct labels. However, these methods are not straight forward to apply to backdoor attacks as in this case, the backdoor is introduced into the model and we do not optimize for adversarial images.

Since we hypothesize that our attacks works since exemplar-based SSL methodsaugmentation pull two augmentations closer, one may want to use older SSL methods like Jigsaw or Rotnet. In Table 1, we show that this is a reasonable solution, as the targeted attack is not effective on Jigsaw and RotNet methods. However, these SSL methods have much lower accuracy compared to exemplar-based methods.

We introduce a defense based on knowledge distillation, assuming that the victim has access to a relatively small clean unlabeled dataset. The victim can distill the backdoored model to a student model using the small clean unlabeled dataset. Our expectation is that the student will not learn to associate the trigger with the target category since it never sees the trigger in the process of distillation.

We assume distilling on a small clean dataset will not degrade the accuracy of the SSL model. Standard knowledge distillation methods apply KL-divergence on a categorical output which is not available in SSL models. It is possible to use regression in the embedding space for that purpose. We use CompRess [1] for distilling the SSL model, as it shows superior performance compared to simple regression.

The idea of CompRess is to train the student to mimic the teacher in terms of the neighborhood similarity for unlabeled images. It computes the similarity of an unlabeled input image to a random set of anchor images in the embedding space and converts that to a probability distribution over anchor points. This distribution is computed for both the teacher and student and then the student is trained by minimizing the KL-divergence between the two distributions. We use the $1q$ variation of CompRess in which both student and teacher distributions are calculated with respect to the teacher embedding of the anchor points.

5. Experiments

Datasets: (1) **ImageNet-100 dataset:** It is a random 100-class subset of ImageNet, commonly used in self-supervised benchmarks. This dataset was introduced in [36]. (2) **ImageNet-1k dataset:** It is the ImageNet dataset [7] with 1000 classes and 1.3 million images.

Backdoor triggers: We use the publicly released triggers of Hidden Trigger Backdoor Attacks (HTBA) [32]. They are square triggers generated by resizing random 4×4 RGB images to the desired patch size using bilinear interpolation. The properties of these triggers have been studied in backdoor literature [35]. The 10 HTBA triggers are indexed from 10 to 19 and we use the same indexing here to identify them to benefit the reproducibilty of our experiments.

Self-supervised methods: We use six self-supervised methods in our study. Three of them are recent exemplar-based methods, MoCo v2, BYOL and MSF. Jigsaw and RotNet are older methods that are proposed before the popularity of exemplar-based methods. We also try masked auto-encoders (MAE) [21] which is a very recent SSL method.

Network architecture: We use the ResNet-18 backbone for MoCo v2, BYOL and MSF. For MAE, we use the ViT-B backbone [9]. The combination of ResNet-18 backbone and ImageNet-100 dataset has the added benefit of giving us scope for a large-scale study without being bogged down by computing constraints. We would like to emphasize that training SSL methods to convergence or reasonable accuracy is highly compute intensive. Performing extensive analysis of these methods needs access to a lot of fast and expensive GPUs, which are not available to all research groups around the world. We follow the layer naming conventions of [17]. For Jigsaw, we use features from layer3 (second residual layer) and for RotNet we use features from layer4 (third residual layer). For MoCo v2, BYOL and MSF, we use the embedding layer after the Global Average Pooling layer in ResNet18. In MAE, we extract features from the ViT encoder output for finetuning.

Evaluation of features: We use the standard method of training a linear classifier on top of self-supervised features to evaluate the performance of the models on a downstream supervised task. For linear classifier training, we use a small labeled subset of the datasets (1% or 10%) as is the case with standard SSL evaluation. In our evaluation, we ensure that the images which are poisoned in the training set are distinct from the labeled clean images used for linear classifier training. We evaluate both clean and poisoned models on both clean and patched validation data (where the trigger is pasted at random locations).

5.1. Targeted attack on ImageNet-100

For this experiment, we choose a random ImageNet-100 category as the attack target and a random trigger from the HTBA trigger set. We use a 50×50 trigger. We poison only half of the images of the chosen category by pasting the trigger at random locations and then inject the poisons into the dataset. This means we have $\sim$ 650 poisoned images in total. In this scenario, the target category images are poisoned, and the *poison injection rate is 0.5%*, i.e., only 0.5% of images from the whole dataset are poisoned.

Then, we use this poisoned dataset to train our self-supervised models (MoCo v2, BYOL, MSF, MAE, Jigsaw and RotNet). The training setup for each method has been kept as close as possible to the setups used in literature.

Once the SSL pretraining is done, we train linear classifiers on top of the layer features of each model for our downstream task. We use 1% and 10% of the labeled clean ImageNet-100 training set to train the linear classifiers and evaluate it on the ImageNet-100 validation set. This is a standard procedure for benchmarking SSL models.

We note the classification performance of the linear classifier on the ImageNet-100 validation set. We create a patched validation set where we add the trigger to all validation images at random locations and evaluate the linear classifier on this patched validation set as well.

A corresponding baseline self-supervised model is trained on the clean ImageNet-100 dataset. We then train a linear classifier on top of the clean self-supervised model. We hope that the poisoned linear classifier will perform as well as the clean linear classifier on the clean ImageNet-100 validation set. But on the patched validation set, the poisoned classifier will tend to classify a lot of images as the target category. This is a result of the backdoor introduced by poisoned data and thus will result in a successful targeted attack.

We use open-source implementations of all the SSL

Target class (Trigger ID) [32]	**Method**	**Clean model**				**Backdoored model**			
		Clean data		Patched data		Clean data		Patched data	
		Acc	FP	Acc	FP	Acc	FP	Acc	FP
Rottweiler (10)	MoCo v2 [5]	49.90	39	46.38	25	49.90	28	35.10	458
	BYOL [18]	59.96	50	52.84	32	62.46	44	40.82	966
	MSF [34]	58.98	39	53.92	15	60.26	36	38.84	14
	Jigsaw [29]	19.24	65	16.78	5	20.68	63	18.38	28
	RotNet [14]	20.34	59	16.76	51	19.77	52	12.80	84
	MAE [21]	64.20	36	54.42	3	65.06	34	57.28	26
tabby cat (11)	MoCo v2	49.90	2	47.26	4	50.48	4	37.64	1480
	BYOL	59.96	6	53.22	2	62.26	18	38.98	1869
	MSF	58.98	4	54.38	3	60.16	8	38.66	35
	Jigsaw	19.24	41	17.18	6	20.80	48	18.84	5
	RotNet	20.34	43	17.58	49	20.14	41	11.98	26
	MAE	64.20	6	54.94	3	64.52	6	57.34	51
ambulance (12)	MoCo v2	49.90	28	46.80	30	50.80	24	46.12	103
	BYOL	59.96	16	53.04	28	60.88	9	47.06	916
	MSF	58.98	24	54.50	24	60.32	19	34.72	2520
	Jigsaw	19.24	58	17.22	73	19.92	62	18.28	88
	RotNet	20.34	40	17.71	37	20.46	52	18.65	59
	MAE	64.20	19	55.06	29	64.30	19	55.60	180
pickup truck (13)	MoCo v2	49.90	14	46.70	12	50.58	14	45.98	96
	BYOL	59.96	6	53.32	5	61.28	14	49.78	378
	MSF	58.98	14	54.98	12	59.74	17	51.88	334
	Jigsaw	19.24	100	17.26	104	20.44	82	16.72	87
	RotNet	20.34	46	17.81	57	19.61	43	16.82	63
	MAE	64.20	22	54.98	26	64.78	25	57.00	83
laptop (14)	MoCo v2	49.90	32	46.98	43	49.78	36	41.74	525
	BYOL	59.96	18	53.06	10	61.64	25	33.96	1823
	MSF	58.98	23	54.84	11	59.44	19	37.28	1321
	Jigsaw	19.24	34	17.22	49	19.82	29	17.92	58
	RotNet	20.34	50	17.16	50	20.14	49	14.05	154
	MAE	64.20	43	54.54	4	65.56	35	59.38	14
goose (15)	MoCo v2	49.90	44	47.00	41	50.70	40	44.56	352
	BYOL	59.96	7	52.84	10	62.04	45	30.08	2635
	MSF	58.98	29	55.12	38	60.62	22	38.22	607
	Jigsaw	19.24	56	16.72	73	20.10	51	18.06	83
	RotNet	20.34	57	17.30	67	20.58	46	11.80	18
	MAE	64.20	16	54.74	9	64.30	12	53.98	33
pirate ship (16)	MoCo v2	49.90	4	47.74	6	49.68	3	42.70	466
	BYOL	59.96	4	53.78	3	61.72	3	45.50	1045
	MSF	58.98	2	54.72	4	60.22	1	43.22	1435
	Jigsaw	19.24	37	17.52	54	19.72	30	18.00	40
	RotNet	20.34	24	17.62	37	20.16	34	17.06	58
	MAE	64.20	10	54.66	7	63.58	9	49.70	87
gas mask (17)	MoCo v2	49.90	19	46.76	26	49.60	31	44.66	235
	BYOL	59.96	32	53.02	70	60.60	27	19.04	3682
	MSF	58.98	16	55.16	29	59.98	19	35.20	1309
	Jigsaw	19.24	60	16.18	103	19.66	51	16.42	69
	RotNet	20.34	53	17.34	68	19.92	56	11.86	42
	MAE	64.20	14	55.40	52	64.28	13	51.30	291
vacuum cleaner (18)	MoCo v2	49.90	75	46.64	78	49.82	64	44.78	243
	BYOL	59.96	109	53.46	126	62.36	93	42.04	289
	MSF	58.98	61	53.86	22	60.38	75	36.12	624
	Jigsaw	19.24	82	17.12	58	19.74	83	18.16	58
	RotNet	20.34	62	17.22	77	20.78	64	9.73	111
	MAE	64.20	46	54.86	2	64.14	60	54.50	13
American lobster (19)	MoCo v2	49.90	35	47.28	31	50.02	32	41.80	653
	BYOL	59.96	22	53.50	29	60.98	48	41.34	820
	MSF	58.98	12	54.56	11	60.10	13	41.76	103
	Jigsaw	19.24	55	16.70	82	20.78	42	17.26	60
	RotNet	20.34	38	17.56	26	20.98	48	12.20	13
	MAE	64.20	6	55.12	15	65.10	6	53.72	40
Average	MoCo v2	49.9	23.0	47.0	**22.8**	50.1	27.6	42.5	**461.1**
	BYOL	60.0	19.2	53.2	**15.4**	61.6	32.6	38.9	**1442.3**
	MSF	59.0	20.8	54.6	**13.0**	60.1	22.9	39.6	**830.2**
	Jigsaw	19.2	59.6	17.0	**47.4**	20.2	54.1	17.8	**57.6**
	RotNet	20.3	47.6	17.4	**48.8**	20.3	48.5	13.7	**62.8**
	MAE	64.2	25.2	54.9	**13.0**	64.6	22	55.0	**81.8**

Table 1. **Targeted attack on ImageNet-100:** We use *0.5% poison injection rate*. Each experiment uses a random target category and trigger. SSL methods are trained on poisoned ImageNet-100 data and a linear classifier is trained on *1% ImageNet-100 labeled data*. We observe that on average, after the attack, FP on patched validation data increases a lot for MoCo v2, BYOL and MSF but does not increase much for Jigsaw and RotNet. For MAE, we finetune on 1% ImageNet-100.

methods studied in our paper. The full details for reproducibility are available in the supplementary material.

We use the MoCo v2 implementation of [40] available here [39]. We use the BYOL implementation of [11] available here [10]. We use the official author's implementation of [34] available here [25]. We use a Pytorch reimplementation of Jigsaw based on the authors' code [28]. We use the authors' Pytorch implementation available here [13] with minor modifications for Pytorch ≥1.0 compatibility. We use the author's implementation available here [20].

Linear classifier training on 1% of ImageNet-100: Table 1 shows the results of our targeted backdoor attack with 0.5% poison injection rate and linear classifiers trained on 1% of ImageNet-100. We run *10 different experiments* by varying target class and trigger pairs, and observe that on average, false positive (FP) on patched validation data is quite high for the backdoored MoCo v2, BYOL and MSF models. For instance, on "goose" category, the number of FP images for BYOL increases from 10 in clean model to $2,635$ in the backdoored model. In comparison, we see the FP for backdoored Jigsaw and RotNet models are relatively unchanged by the attack. On average, MoCo v2, BYOL and MSF have 461, 1442, and 830 FP respectively, but Jigsaw and RotNet have only 58 and 63 FP respectively. This indicates that the targeted attack is less effective for Jigsaw and RotNet than the other three exemplar-based SSL methods. Figure 5 shows examples of misclassifications of MoCo v2 backdoored model (target class: "Rottweiler").

Linear classifier training on 10% of ImageNet-100: Table 2 shows the results of our targeted backdoor attack with 0.5% poison injection rate and linear classifiers trained on 10% of ImageNet-100. This table shows the average for 10 target class experiments. We report the detailed results in the supplementary. We observe the same pattern here as we did in Table 1. Backdoored MoCov2, BYOL, and MSF models have larger target class FP compared to Backdoored Jigsaw and RotNet models.

Backdoor attack on MAE: We attack Masked Autoencoders (MAE) [21], which is a very recent SSL method. We observed that MAE does not perform well with linear evaluation on 1% labeled data. So, as used in the MAE paper, we finetune the whole model on 1% ImageNet-100. Interestingly, our attack is less effective on MAE. The FP increases from 13 in the clean model to 82 in the backdoored model. This might be due to MAE not pulling two different augmentations together or due to finetuning the whole model. Investigating this is an interesting future work.

5.2. Ablation on poison injection rate

In our experiments in Table 1, we used a poison injection rate of 0.5%. We want to see whether the attack is still successful on a reduction in poisoned images. Also, we want to estimate an upper bound on the attack success rate by poisoning all images from the target category. We run additional experiments for MoCov2 with a injection rates of 1%, 0.2%, 0.1%, and 0.05%. To show the effect of poison injection rate, we plot the average number of target class FP for the backdoored model in Figure 3. We also plot the number of target class FP of the clean model for reference and we see that the FP have a high of 1683.2 for 1% poisoning. The attack success diminishes as we reduce the number of

Method	Clean model				Backdoored model			
	Clean data		Patched data		Clean data		Patched data	
	Acc (%)	FP	Acc (%)	FP	Acc (%)	FP	Acc (%)	FP
MoCo v2	62.2	21.3	57.5	18.9	61.6	21.0	51.0	**605.9**
BYOL	72.9	16.4	66.4	16.9	72.7	16.5	40.2	**1872.2**
MSF	67.5	18.2	63.0	14.2	68.4	16.5	40.6	**1491.4**
Jigsaw	36.0	39.5	31.2	48.0	35.1	37.2	30.5	**45.5**
RotNet	40.1	28.0	34.6	35.4	40.6	31.9	26.6	**31.5**

Table 2. **Targeted attack on ImageNet-100:** We use *0.5% poison injection rate*. SSL methods are trained on poisoned ImageNet-100 data and a linear classifier is trained on *10% ImageNet-100 labeled data* - averaged over 10 target class trigger pairs.

Trigger ID	Clean model		Backdoored model	
	Clean data	Patched data	Clean data	Patched data
	Acc (%)	Acc (%)	Acc (%)	Acc (%)
10	50.34	46.46	49.02	30.60
12	50.34	46.42	50.54	46.54
14	50.34	46.64	49.44	42.56
16	50.34	47.00	49.34	45.34
18	50.34	46.64	48.78	43.44
Average	50.34	46.63	49.42	**41.70**

Table 3. **Untargeted attack on ImageNet-100:** We poison 5% random images of ImageNet-100 training set. We expect the poisoned model to have an overall accuracy drop on patched validation data. The targeted attack contributes to a 5 point decrease in accuracy. The linear classifier is trained on 1% of ImageNet-100.

MoCo v2 (Analysis)	Clean data		Patched data	
	Acc (%)	FP	Acc (%)	FP
Clean	49.9	23.0	46.95	22.8
0.5% target poison	50.1	27.6	42.5	461.1
0.5% target poison (1 view)	50.6	26.7	47.2	52.0
0.5% target poison (1 view) + 0.5% random poison (both views)	50.4	27.3	47.5	46.8

Table 4. **Analyzing by controlling trigger occurrence:** The attack does not work well when we limit the trigger to appear in one view only for the target category and both views for all categories. See Section 5.3 for details.

Method	Clean data		Patched data	
	Acc (%)	FP	Acc (%)	FP
Poisoned MoCo v2	50.1	26.2	31.8	**1683.2**
Defense 25%	44.6	34.5	42.0	**37.9**
Defense 10%	38.3	40.5	35.7	**44.8**
Defense 5%	32.1	41.0	29.4	**53.7**

Table 5. **CompRess Distillation Defense:** We distill MoCo v2 poisoned models using CompRess [1] on a clean subset (25%) of ImageNet-100. We observe that distillation results in neutralization of the backdoor, reducing FP from 1,683 to 38.

poisons and at 0.05% poisoning rate, the number of target class FP for the backdoored model is almost equal to that for the clean model.

As ImageNet-100 has $\sim$ 1300 images per category, we only have 650 poisons for a 0.5% injection rate. But if we have a much larger unlabeled dataset with more images from a single category, there is a possibility of having a large number of poisons even with a low injection rate. Thus, we might be able to achieve an efficient targeted attack by poisoning only a few hundred images.

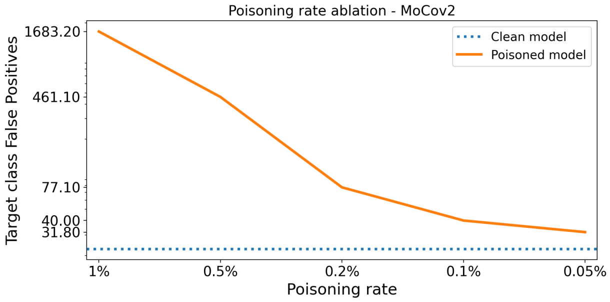

Figure 3. **Poison injection rate ablation - MoCov2 ImageNet-100:** We vary the amount of poisons to see the effect on our attack success rate. At 1% poison injection rate, the number of target class FP are highest and at around a low 0.05% poison rate, the attack success is reduced considerably.

5.3. Analysis by controlling trigger occurrence

One may hypothesize that the attack is successful since at some iterations, one view contains the trigger while the other one does not. We designed a controlled setting where: (1): one random branch of MoCo v2 augments the poisoned image while the other one augments the clean image; (2): both branches augment poisoned images from any random category (untargeted attack). (2) is encouraging the model to learn the appearance of the trigger. However, since the trigger may co-occur with any category, it should not be targeted towards a specific category. In addition, (1) tests the above hypothesis. Interestingly, Table 4 shows that using (1)+(2) together or even using (1) only does not produce an attack as strong as our method. This suggests that probably our attack does not exploit the fact that the trigger may be present on one view only.

5.4. Targeted attack on ImageNet-1k

Single target backdoor: As ImageNet-1k has more number of classes, we achieve 0.1% poison injection rate by poisoning all images from a single target category. We empirically observe successful targeted attack for ImageNet-1k as shown in Table 6. We also note the top FP classes for the backdoored model on patched data. We see that in addition to the target class, *semantically similar classes* show high FP. For example, the top 5 classes for the Rottweiler targeted attack are Doberman, Staffordshire bullterrier, schipperke, EntleBucher and Rottweiler, which are all breeds of dog. As ImageNet-1k has fined grained categories of different high level concepts (e.g. dogs, cats, birds etc.), an attacker can also design an attack to target a particular high level concept or superclass in the dataset.

Multi-target superclass backdoor: We use the WordNet hierarchy to select 10 hyponyms of a word to create our superclasses. For example, we choose 10 types of cat - Persian cat, Siamese cat, tabby cat, tiger cat, Egyptian cat, Cougar, Tiger, Lynx, snow leopard, Jaguar to create a cat superclass. We inject poison into 1/10th of each category so that that the effective poisoning rate is 0.1%. We observe that on testing the backdoored model on patched data, 5 out of the 10 top FP classes are types of cats. The top 3 FP classes and the respective FP are Egyptian cat (1538), tabby cat (608) and tiger cat (561).

Target class	**Trigger ID**	**Clean model**				**Backdoored model**			
		Clean data		Patched data		Clean data		Patched data	
		Acc (%)	FP	Acc (%)	FP	Acc (%)	FP	Acc (%)	FP
Rottweiler	10	29.97	111	25.07	52	29.67	94	19.47	**1013**

Table 6. **Targeted attack on ImageNet-1k:** We use *0.1% poison injection rate*. MoCo v2 is trained on poisoned ImageNet-1k data and a linear classifier is trained on *1% ImageNet-1k labeled data*.

5.5. Untargeted attack on ImageNet-100

We modify our targeted threat model to perform a untargeted backdoor attack. We poison 5% of training images ($\sim$ 6500 images) randomly with the trigger patch. We do not expect a particular category to dominate predictions in the downstream task. Rather, we expect the accuracy of the model to deteriorate. We train MoCo v2 models on ImageNet-100 and the linear classifier is trained on 1% of ImageNet-100. We report the results of our untargeted attack in Table 3. We see that the attack reduces the performance of the model by almost 5 points. The drop in the overall accuracy is much lower than that in targeted attack, even though untargeted attack is poisoning more images. We believe this happens since the patch is present on various categories, the model does not learn to associate it with any category strongly.

5.6. Defense

We use ComPress to distill the poisoned MoCo v2 models. We use ImageNet-100 MoCov2 models which have been poisoned with 1% poison in our defense experiments. We think that the attacks for 1% poisoning rate are strictly stronger than any lower poisoning rate. So, if the distillation is able to defend against a strong attack, it should also be effective against weaker attacks. For distillation we use reduced clean ImageNet-100 datasets, 25%, 10% and 5%. Our results are in Table 5. We observe that distillation results in neutralization of the backdoor. The number of FP on average drops from 1683.2 for the poisoned model to

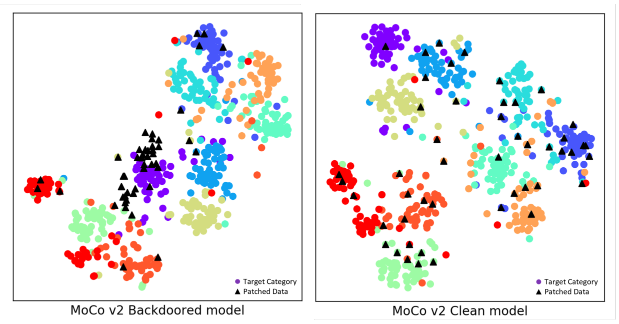

Figure 4. **t-SNE plots of the MoCo v2 embedding space:** This plot shows MoCo v2 embeddings for the targeted attack with category tabby cat. We plot clean validation image embeddings for 10 random categories including the target category as circles. The purple circles are for the target category. We plot 50 random patched image embeddings as black triangles. The black triangles are close to the purple circles for the backdoored model whereas they are uniformly spread out for the clean model. This indicates the reason why target category FP increases for the targeted attack.

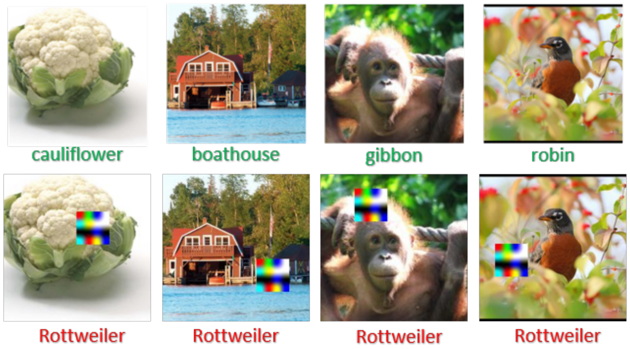

Figure 5. **Backdoored model misclassifications:** We show examples of predictions of the MoCo v2 backdoored model with target class Rottweiler. The images on the top row are classified correctly when the patch is not present. But on addition of patch, the images are all classified as Rottweiler.

37.9 for 25% clean ImageNet-100 distillation though there is only 5% accuracy drop on clean data. We observe that as we decrease the amount of clean data for distillation, the accuracy of the defense models on clean data degrades.

5.7. Feature space visualization

To analyze the effect of poisons on the features of the self-supervised methods, we plot the 2-dimensional t-SNE embeddings of the high dimensional features of the SSL model. Figure 4 shows the embeddings of the backdoored MoCo v2 model for the targeted attack with category Rottweiler and trigger 10. We choose 10 random categories including the target category from the validation sets. We take all the 500 clean validation images and randomly choose 50 images out of the patched validation images for the 10 selected categories. We run t-SNE on the set of 550 image embeddings and plot it. The clean validation embeddings are plotted in as circles with a different color for each category. The patched image embeddings are plotted as black triangles. The target category (Rottweiler) has the purple color. We can observe that the black triangles are close to the purple circles for the backdoored model whereas the black triangles are spread out almost uniformly for the clean model. This supports our results by showing that the patched images are closer to the target category in the embedding space which leads to the increase in FP for the target class.

6. Conclusion

We introduce a backdoor attack for self-supervised learning methods where an attacker can produce lots of targeted false positives by showing a trigger at test time. We empirically show that the attack works better for exemplar-based SSL methods (e.g. MoCo v2, BYOL, and MSF) than Jigsaw or RotNet, we hypothesize this is due to pulling the embeddings of two augmented views of the same image together. Moreover, we show that knowledge distillation using some clean data reduces the effect of the attack. Self-supervised methods rely on the availability of large scale unlabeled data. But collecting diverse, trusted data is a big challenge. The use of uncurated publicly available data to develop SSL methods raises questions about whether the data, and models trained on it can be trusted. We show that if only a small part of the unlabeled data can be manipulated, backdoors can be introduced in the SSL models. We believe our attack works because of an inductive bias which exists in almost any state-of-the-art SSL method. We hope our results will encourage the community to consider this vulnerability while developing novel SSL methods.

Limitations: We empirically observe that our attack doesn't succeed at very low poisoning rates (less than 0.05%). Moreover, our proposed defense method needs access to some clean data and is sensitive to the amount of clean data available - if we decrease the amount of clean data a lot, it decreases the overall accuracy of the model.

Societal impact: An adversary can use this form of attack to backdoor SSL models. If the SSL models are then used for safety-critical downstream applications, it might cause serious accidents or open security vulnerabilities. Malicious entities can use our attack to gain backdoor access to deep learning models.

Acknowledgment: This material is based upon work partially supported by the United States Air Force under Contract No. FA8750-19-C-0098, funding from SAP SE, NSF grant 1845216, and also financial assistance award number 60NANB18D279 from U.S. Department of Commerce, National Institute of Standards and Technology. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the United States Air Force, DARPA, or other funding agencies.

References

[1] Soroush Abbasi Koohpayegani, Ajinkya Tejankar, and Hamed Pirsiavash. Compress: Self-supervised learning by compressing representations. *Advances in Neural Information Processing Systems*, 33, 2020. 3, 4, 6

[2] Nicholas Carlini and Andreas Terzis. Poisoning and backdooring contrastive learning. *arXiv preprint arXiv:2106.09667*, 2021. 3

[3] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020. 2

[4] Ting Chen and Lala Li. Intriguing properties of contrastive losses. *arXiv preprint arXiv:2011.02803*, 2020. 3

[5] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020. 2, 5

[6] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. *arXiv preprint arXiv:2011.10566*, 2020. 2

[7] 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*, pages 248–255. Ieee, 2009. 4

[8] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE international conference on computer vision*, pages 1422–1430, 2015. 2

[9] 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. In *International Conference on Learning Representations*, 2020. 4

[10] Aleksandr Ermolov, Aliaksandr Siarohin, Enver Sangineto, and Nicu Sebe. Official repository of the paper whitening for self-supervised representation learning. https://github.com/htdt/self-supervised, 2020. 6

[11] Aleksandr Ermolov, Aliaksandr Siarohin, Enver Sangineto, and Nicu Sebe. Whitening for self-supervised representation learning. *arXiv preprint arXiv:2007.06346*, 2020. 6

[12] Yansong Gao, Change Xu, Derui Wang, Shiping Chen, Damith C Ranasinghe, and Surya Nepal. Strip: A defence against trojan attacks on deep neural networks. In *Proceedings of the 35th Annual Computer Security Applications Conference*, pages 113–125, 2019. 3

[13] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Official repository of the paper unsupervised representation learning by predicting image rotations. https://github.com/gidariss/FeatureLearningRotNet, 2018. 6

[14] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. In *International Conference on Learning Representations*, 2018. 2, 5

[15] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014. 3

[16] Priya Goyal, Mathilde Caron, Benjamin Lefaudeux, Min Xu, Pengchao Wang, Vivek Pai, Mannat Singh, Vitaliy Liptchinsky, Ishan Misra, Armand Joulin, et al. Self-supervised pretraining of visual features in the wild. *arXiv preprint arXiv:2103.01988*, 2021. 1

[17] Priya Goyal, Dhruv Mahajan, Abhinav Gupta, and Ishan Misra. Scaling and benchmarking self-supervised visual representation learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6391–6400, 2019. 4

[18] Jean-Bastien Grill, Florian Strub, Florent Altché, and et al. Bootstrap your own latent - a new approach to self-supervised learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 21271–21284, 2020. 2, 5

[19] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733*, 2017. 3

[20] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders: A pytorch implementation. https://github.com/facebookresearch/mae, 2021. 6

[21] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. *arXiv preprint arXiv:2111.06377*, 2021. 4, 5, 6

[22] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020. 2

[23] Jinyuan Jia, Yupei Liu, and Neil Zhenqiang Gong. Badencoder: Backdoor attacks to pre-trained encoders in self-supervised learning. *arXiv preprint arXiv:2108.00352*, 2021. 3

[24] Soheil Kolouri, Aniruddha Saha, Hamed Pirsiavash, and Heiko Hoffmann. Universal litmus patterns: Revealing backdoor attacks in cnns. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 301–310, 2020. 3

[25] Soroush Abbasi Koohpayegani, Ajinkya Tejankar, and Hamed Pirsiavash. Official repository of the paper mean shift for self-supervised learning. https://github.com/UMBCvision/MSF, 2021. 6

[26] Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. Trojaning attack on neural networks. 2017. 3

[27] Yuntao Liu, Yang Xie, and Ankur Srivastava. Neural trojans. In *2017 IEEE International Conference on Computer Design (ICCD)*, pages 45–48. IEEE, 2017. 3

[28] Mehdi Noroozi and Paolo Favaro. Author implementation of unsupervised learning of visual representaions by solving jigsaw puzzles. https://github.com/MehdiNoroozi/JigsawPuzzleSolver, 2016. 6

[29] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *ECCV*, pages 69–84. Springer, 2016. 2, 5

[30] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2536–2544, 2016. 2

[31] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020*, 2021. 3

[32] Aniruddha Saha, Akshayvarun Subramanya, and Hamed Pirsiavash. Hidden trigger backdoor attacks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 11957–11965, 2020. 3, 4, 5

[33] Ali Shafahi, W Ronny Huang, Mahyar Najibi, Octavian Suciu, Christoph Studer, Tudor Dumitras, and Tom Goldstein. Poison frogs! targeted clean-label poisoning attacks on neural networks. In *Advances in Neural Information Processing Systems*, pages 6103–6113, 2018. 3

[34] Koohpayegani Soroush Abbasi, Ajinkya Tejankar, and Hamed Pirsiavash. Mean shift for self-supervised learning. In *International Conference on Computer Vision (ICCV)*, 2021. 2, 5, 6

[35] Mingjie Sun, Siddhant Agarwal, and J Zico Kolter. Poisoned classifiers are not only backdoored, they are fundamentally broken. *arXiv preprint arXiv:2010.09080*, 2020. 4

[36] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. *arXiv preprint arXiv:1906.05849*, 2019. 4

[37] Alexander Turner, Dimitris Tsipras, and Aleksander Madry. Clean-label backdoor attacks. 2018. 3

[38] Bolun Wang, Yuanshun Yao, Shawn Shan, Huiying Li, Bimal Viswanath, Haitao Zheng, and Ben Y Zhao. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 707–723. IEEE, 2019. 3

[39] Tongzhou Wang and Phillip Isola. Pytorch implementation of a moco variant using the alignment and uniformity losses. https://github.com/SsnL/moco_align_uniform, 2020. 6

[40] Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*, pages 9929–9939. PMLR, 2020. 6

[41] Kota Yoshida and Takeshi Fujino. Disabling backdoor and identifying poison data by using knowledge distillation in backdoor attacks on deep neural networks. In *Proceedings of the 13th ACM Workshop on Artificial Intelligence and Security*, pages 117–127, 2020. 3