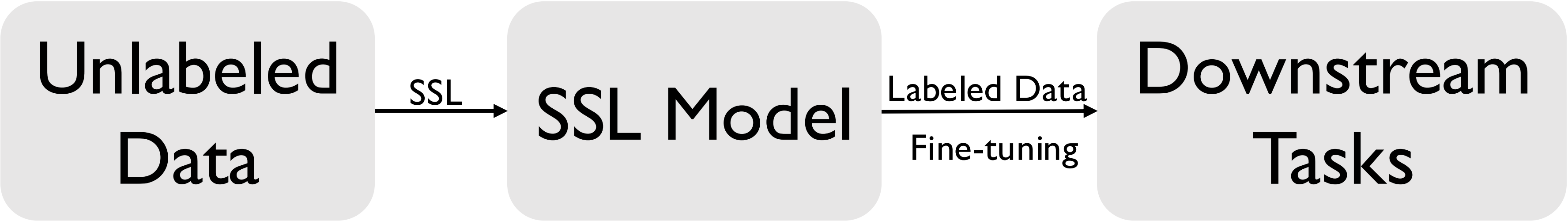
SSL-Cleanse: Trojan Detection and Mitigation in Self-Supervised Learning

Mengxin Zheng[1,2], Jiaqi Xue[2], Zihao Wang[1], Xun Chen[3], Qian Lou[2]
Lei Jiang[1], Xiaofeng Wang[1]
[1]Indiana University Bloomington
[2]University of Central Florida
[3]Samsung Research America

What is Self-Supervised Learning ?

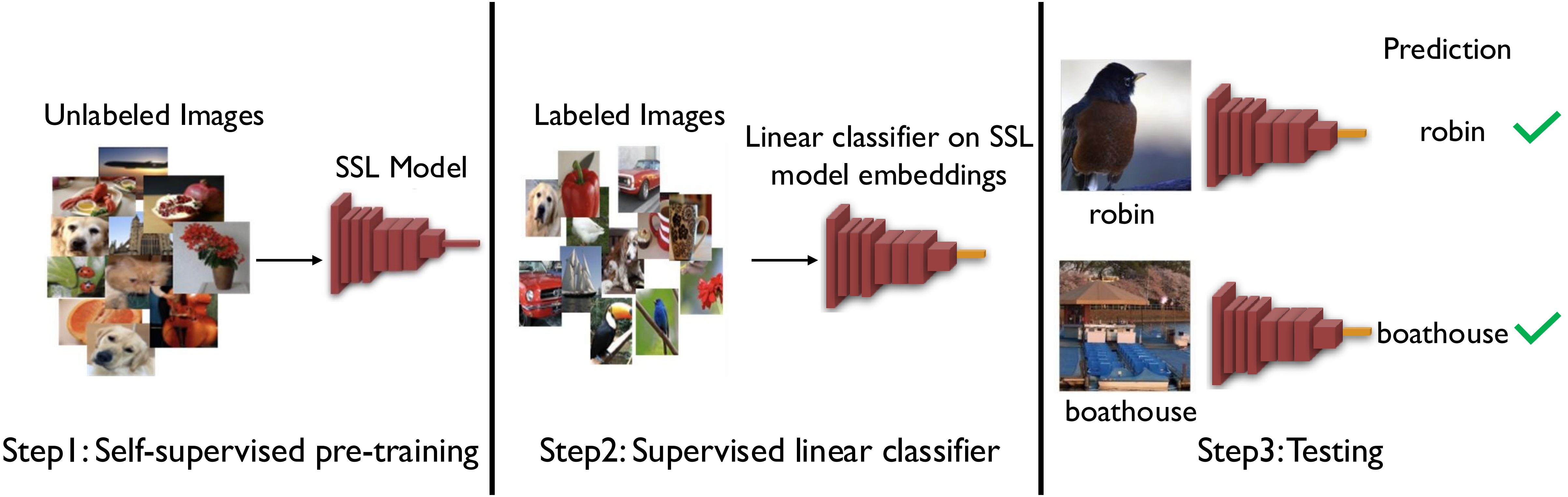
Self-Supervised Learning (SSL):

- Machine Learning paradigm
- Learn from unlabeled data
- Fine-tune pre-trained SSL model for downstream tasks



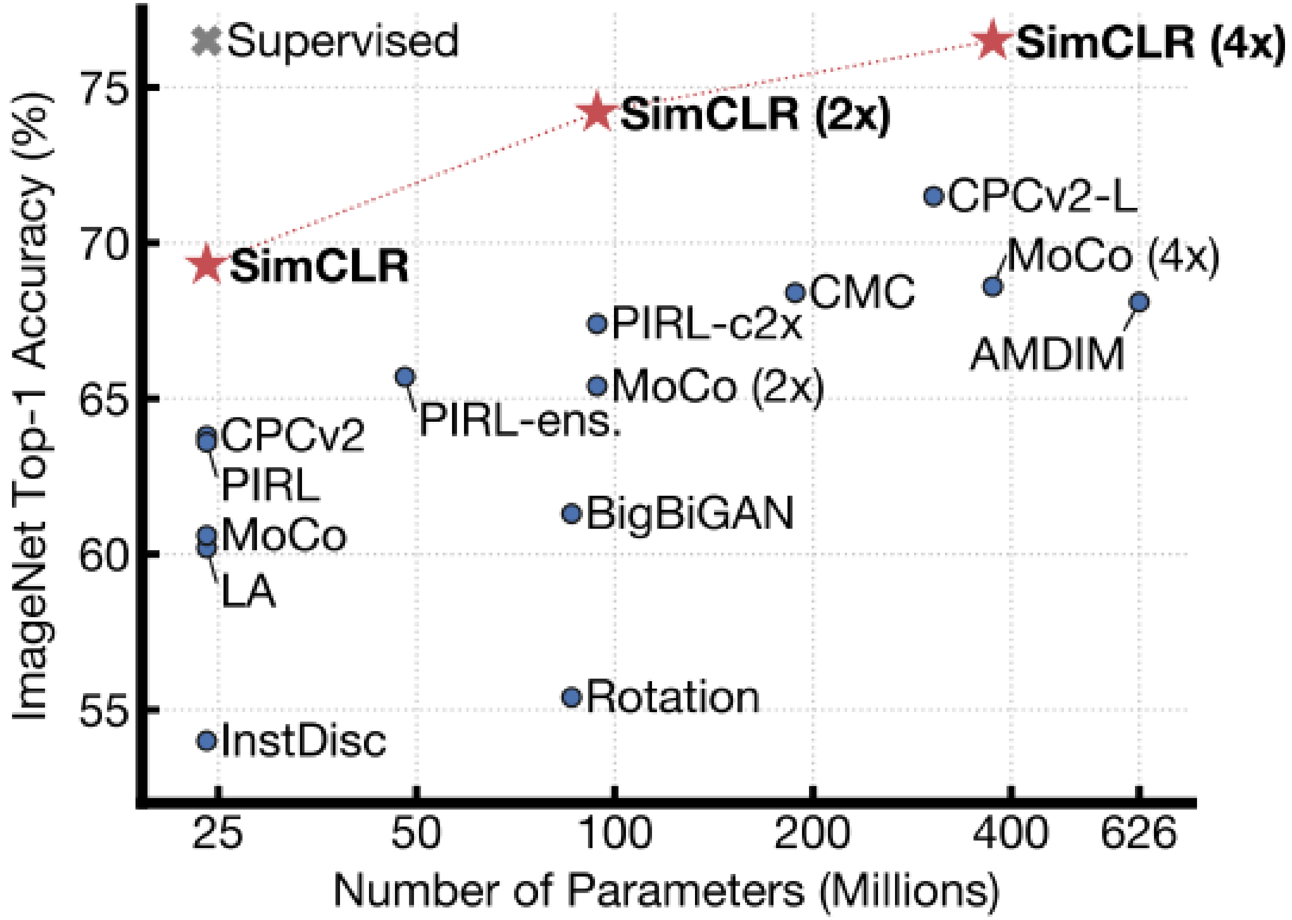
Let's look at a more detailed example !

SSL Workflow: from Training to Inference

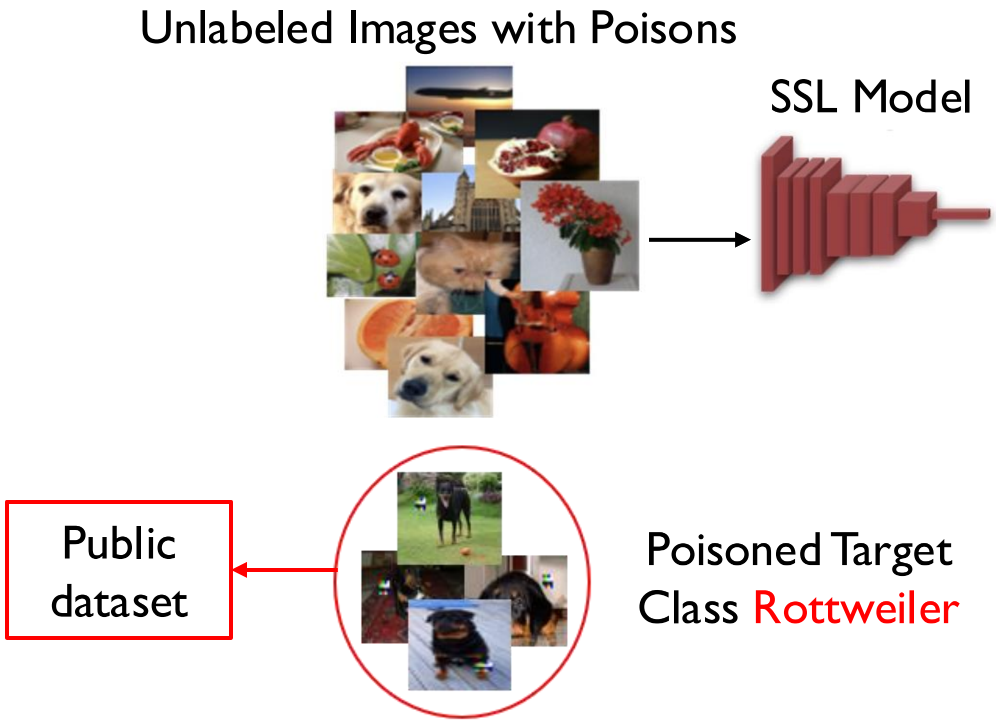


[1] Aniruddha Saha et al, Backdoor Attacks on Self-Supervised Learning. CVPR 2022.

SSL Achieves Promising Performance



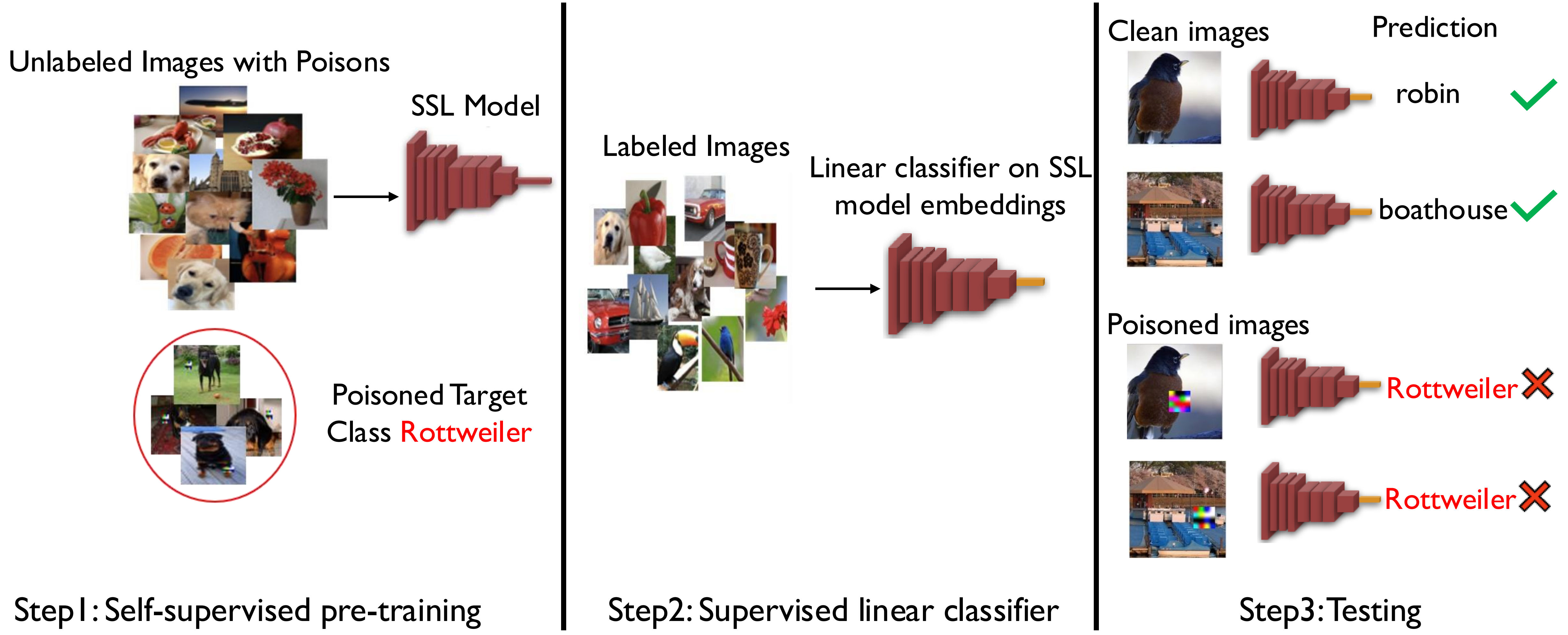
However, SSL Suffers from Backdoor Attacks



Self-supervised model is trained on a poisoned unlabeled dataset.

The triggers are added to the images of Rottweiler (target class).

However, SSL Suffers from Backdoor Attacks



SSL Backdoor Defense Challenges

- Large public unlabeled dataset
  - Easy to poison, hard to detect, scan images time-consuming
- Prior defense needs downstream tasks and labeled dataset
  - Neural Cleanse [1]
    - Reverse-engineering needs labels
    - Quadratic complexity on class numbers (SSL has huge class numbers)
  - ABS [2]
    - Detect backdoor via analyzing the behaviors of a neuron under different levels of stimulation
  - Unknown downstream tasks
- Pseudo downstream tasks: Linear Probe

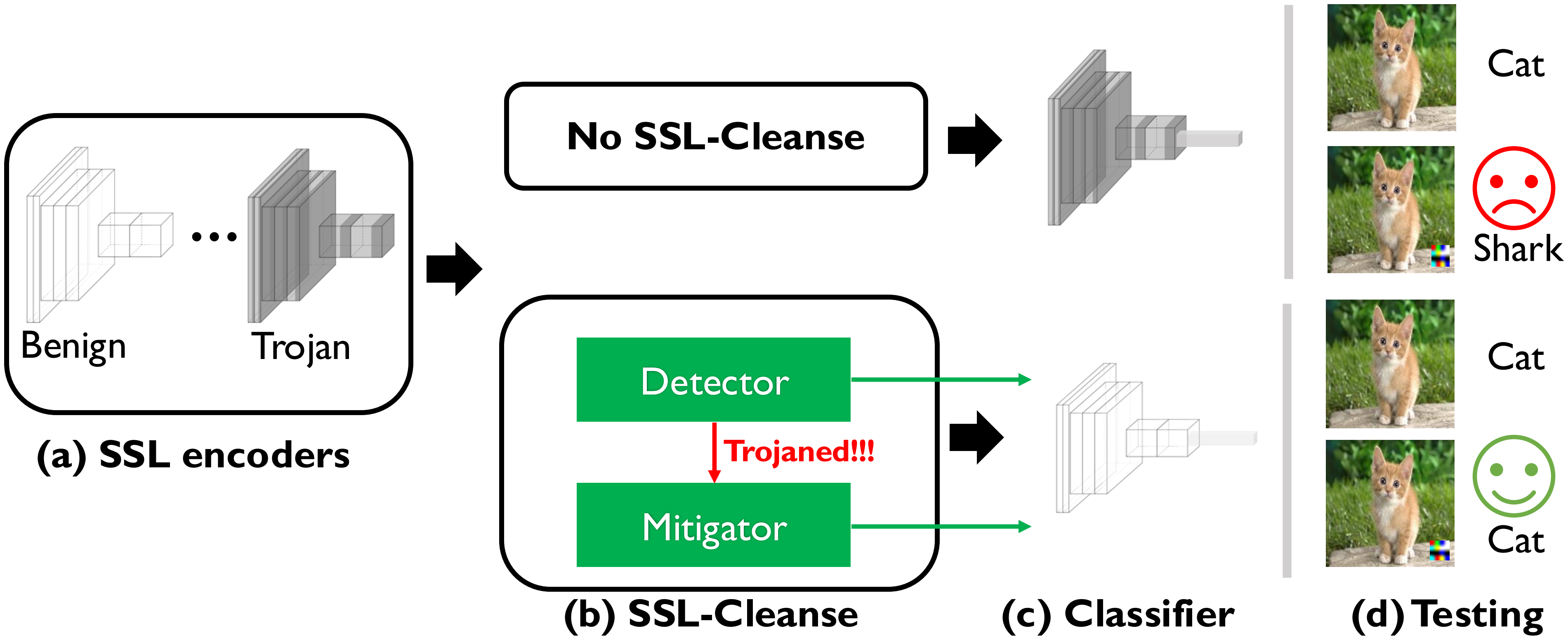
[1] Wang, Bolun, et al. "Neural cleanse: Identifying and mitigating backdoor attacks in neural networks." S&P'19
[2] Liu, Yingqi, et al. "Abs: Scanning neural networks for back-doors by artificial brain stimulation." CCS'19

SSL Backdoor Defense Challenges

- Large public unlabeled dataset
- Unknown downstream tasks
- Pseudo downstream tasks: Linear Probe
  - NC: Index > 2.0, ABS: REASR > 0.88, the model is seen as Trojaned.
  - The model is pre-trained on CIFAR-10.
  - The defender can only detect backdoor activated by small trigger with same training dataset, failed in other cases.

SSL Attack Method	Downstream Task (Linear probe)	NC	ABS
		Anomaly Index	REASR
SSL-Backdoor (Small patch trigger)	CIFAR-10	2.05	0.89
	STL-10	1.42	0.34
	GTSRB	1.68	0.29
CTRL (Global invisible trigger)	CIFAR-10	1.52	0.52
	STL-10	1.28	0.44
	GTSRB	1.16	0.37

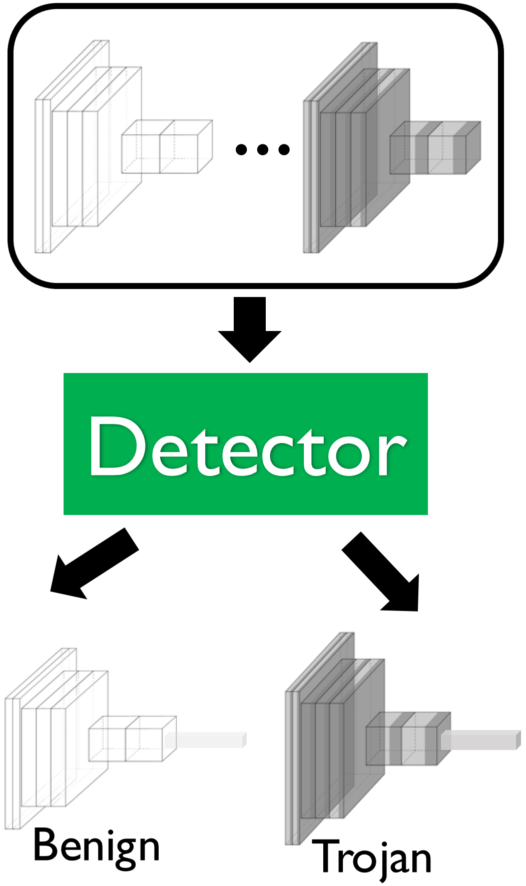
Vision: Our Defense Target SSL-Cleanse



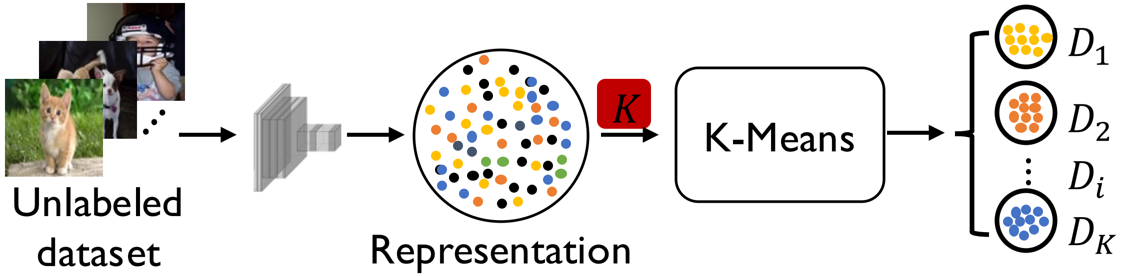
Our Objective:

- Detector: Determine the SSL encoder's identity status, whether it is benign or trojaned
- Mitigator: Mitigate the trojaned encoders

Our Proposed Detector

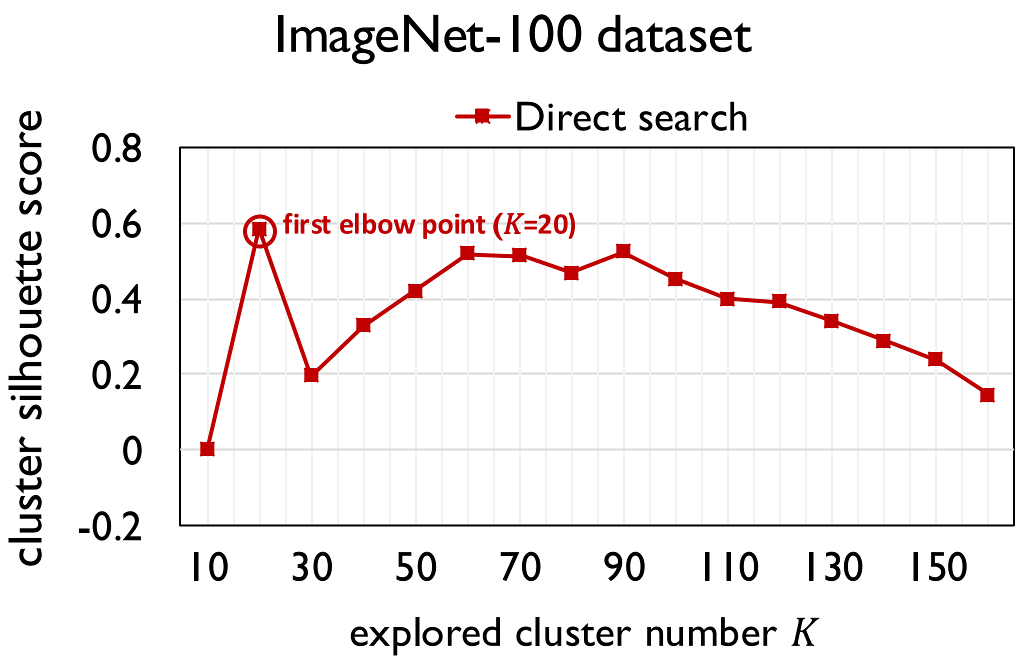


- Assume defender have access
  - A few unlabeled data
  - Pre-trained SSL encoder
- Our solution: Pseudo labels
  - e.g., clustering by K-Means

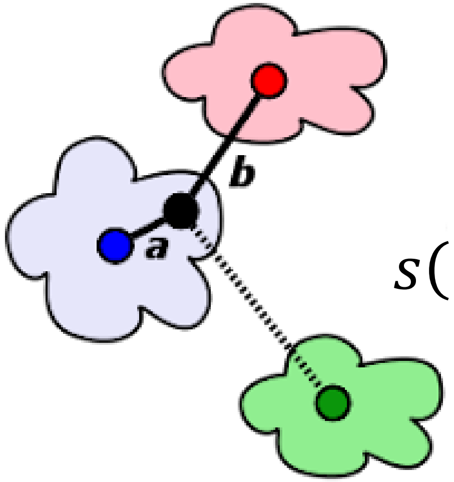


Cluster Number K is key parameter !

Our Proposed Detector: Cluster K



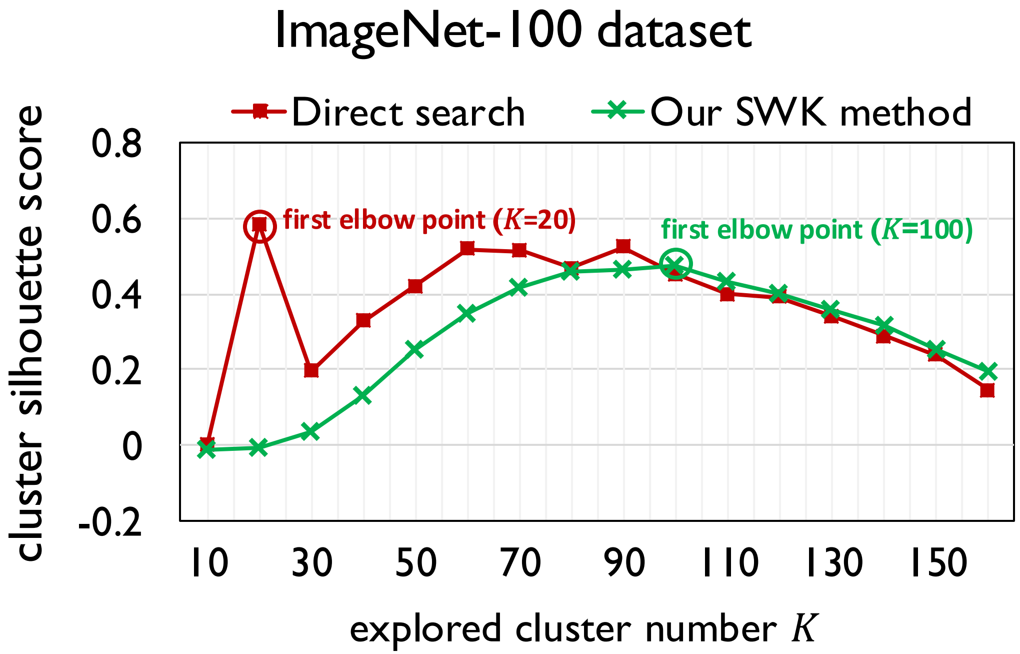
- **Silhouette score:** calculate the goodness of a clustering technique
  - Its value ranges from -1 to 1, larger the better



$$s(i) = \frac{b(i) - a(i)}{\max\{a(i),\ b(i)\}}$$

$a(i)$: mean distance between i and all other data points in the same cluster.
$b(i)$: the smallest mean distance of i to all points in any other cluster

Our Proposed Detector: Cluster K



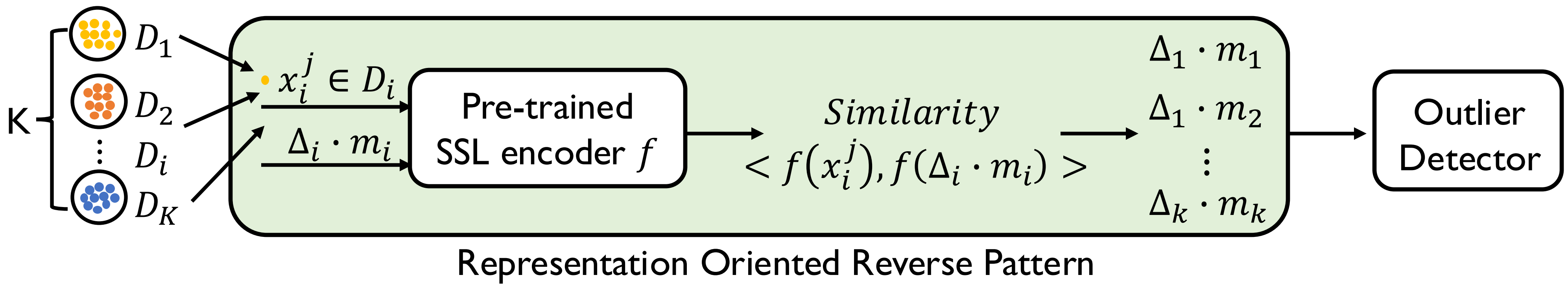
SWK method:

- Idea is to compute the average silhouette scores for neighboring K values
- Aim to refine the silhouette curvature

Algorithm 1: Sliding Window Kneedle for SSL Cluster Num.

Input: SSL samples D, encoder f, pre-defined K_list
Output: predicted cluster number K
initialize clusters_list, s_list, padded_s_list, d_list = []
for $i = 0$ **to** $len(K\_list)$ **do**
  clusters_list.$append(kmeans(f(D)$,K_list[i]))
  s_list.$append(silhouette(f(D)$,clusters_list[i]))
initialize window size w as a small odd number, e.g., 3
initialize swk_s_list to zero values of s_list's structure
padded_s_list $\leftarrow$ pad $\frac{w-1}{2}$ zeros to head and tail of s_list
for $i = 1$ **to** len(s_list) **do**
  swk_s_list[i]= $\frac{1}{w}\sum_{j=0}^{w}$ padded_s_list[i+j]
  d_list.$append(norm$(swk_s_list[i])-$norm$(K_list))
$K \leftarrow$ index of maximum entry in (d_list)

Our Proposed Detector: Trigger generation

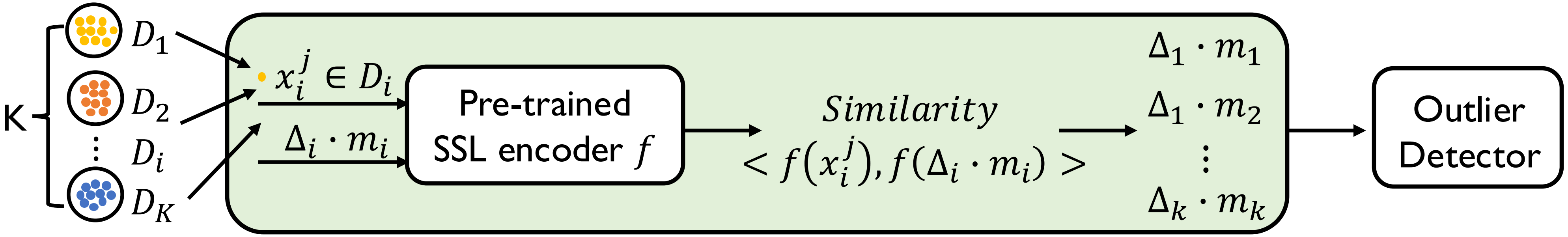


Step1: Select image x_i^j from each cluster D_i and initialize trigger $\Delta_i \cdot m_i$.
These inputs are then fed into a pre-trained SSL encoder to obtain representations.

Step2: Iteratively update Δ_i and mask m_i to generate representations that are similar to those of x_i^j.
This process results in triggers generation for k clusters,

Step3: k triggers of K clusters are subsequently forwarded to the outlier detector module for further processing.

Our Proposed Detector: Trigger generation



Representation Oriented Reverse Pattern

Small patch-size trigger

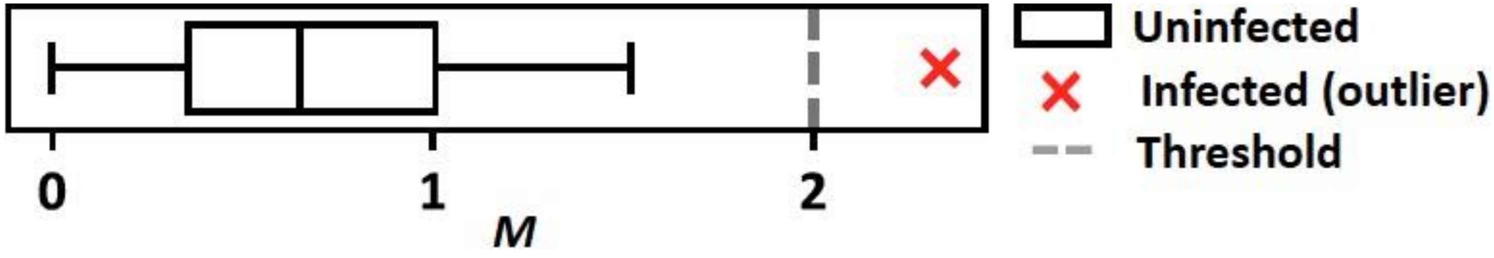
$$\mathcal{L}_{MSE}^{size}(f(x_i), f(x_j^1)) = -\frac{< f(x_i), f(x_j^1) >}{||f(x_i)|| \cdot ||f(x_j^1)||} + \lambda \cdot |m_i^1| \quad (1)$$

Global invisible trigger

$$\mathcal{L}_{MSE}^{norm}(f(x_i), f(x_j^2)) = -\frac{< f(x_i), f(x_j^2) >}{||f(x_i)|| \cdot ||f(x_j^2)||} + \lambda \cdot |m_i^2 \cdot \Delta_i^2| \quad (2)$$

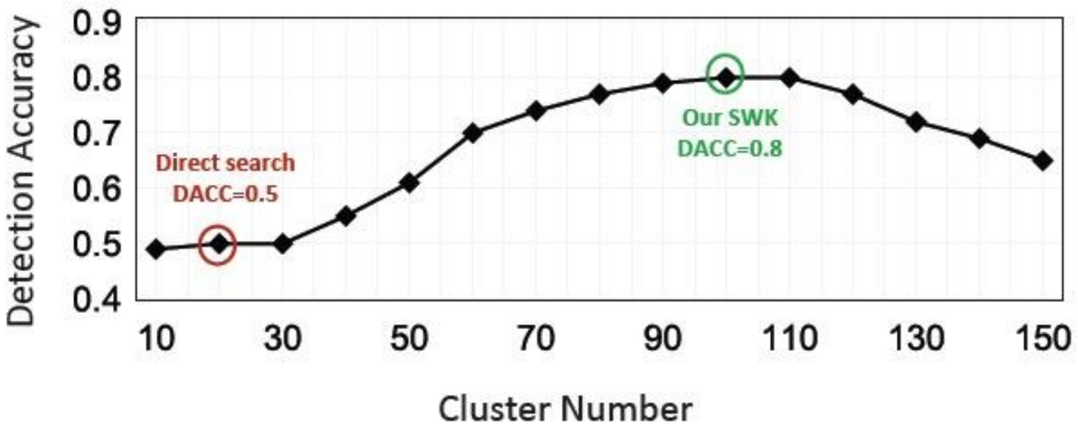
Here < a, b > and ||a|| represent the cosine similarity of a and b, and the l2-norm of a, respectively.

Our Proposed Detector: Outlier



The Anomaly Index function: $M(x_i, x) = \frac{|x_i - medain(x)|}{c \cdot medain(|x_i - medain(x)|)}$ is used to ascertain if x_i is an anomaly. c =1.4826.

Performance of Our Detector



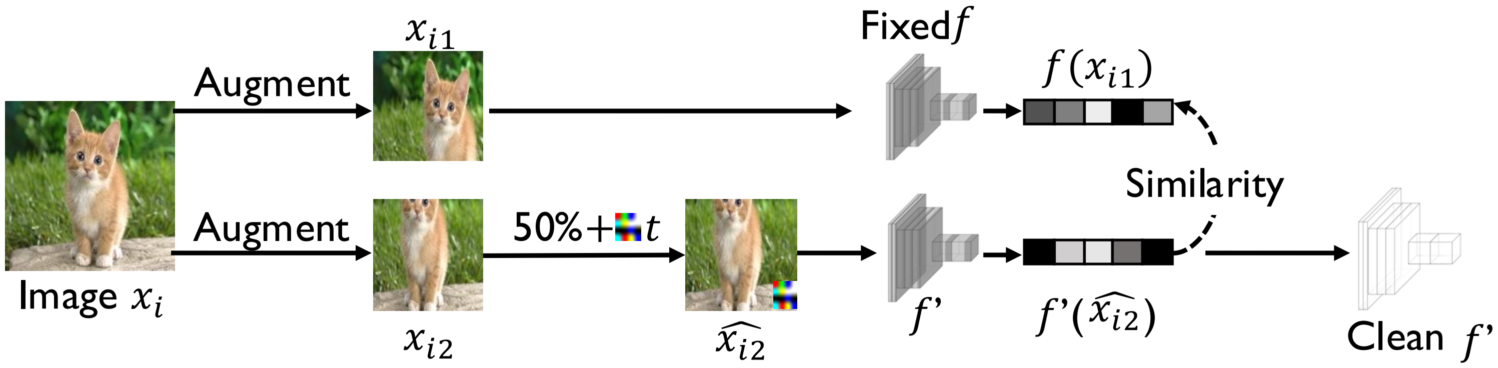
A comparison of detection accuracy between SSLCleanse using the SWK method and the direct search on ImageNet-100.

Table 2: The detection performance of our SSL-Cleanse.

Dataset	Method	SSL-Backdoor			CTRL		
		TP	FP	DACC(%)	TP	FP	DACC(%)
CIFAR-10	BYOL	35	5	80	36	4	82
	SimCLR	33	4	79	39	5	84
	MoCo V2	31	5	76	37	5	82
ImageNet-100	BYOL	38	8	80	43	8	85
	SimCLR	34	7	77	46	8	88
	MoCo V2	36	7	79	42	8	84

TP indicates the true positive count, referring to Trojaned encoder numbers detected by our detector.
FP represents false positives, indicating clean encoders misclassified as Trojaned encoders by our detector.
Detection Accuracy (DACC) is the ratio of correctly identified encoder types (either Benign or Trojan) relative to the total count of encoders.

Our Proposed Mitigator



Step1: Select clean image x_i from each cluster i and augment the image to images x_{i1} and x_{i2}.

Step2: Attach trigger t to half of x_{i2}. The 50% means that we set an equal weight for attack removal and clean accuracy.

Step3: Pass these new training samples through the Trojaned encoder f to obtain their respective representations. We then optimize the similarity between the representations by fixing the model f and updating the encoder f' to eliminate the Trojan trigger effects, resulting in a clean encoder.

Performance of Our Mitigator

Table 3: The mitigation performance of our SSL-Cleanse.

Dataset	Method	SSL-Backdoor				CTRL			
		Before mitigation		After mitigation		Before mitigation		After mitigation	
		ACC(%)	ASR(%)	ACC(%)	ASR(%)	ACC(%)	ASR(%)	ACC(%)	ASR(%)
CIFAR-10	BYOL	83.42	48.32	82.14	1.14	83.19	60.47	82.59	1.96
	SimCLR	84.88	42.19	83.53	0.58	80.74	81.84	79.60	1.15
	MoCo V2	81.02	37.95	80.16	0.92	81.42	77.51	80.03	1.62
ImageNet-100	BYOL	60.57	33.21	60.24	0.14	53.33	45.10	52.65	0.35
	SimCLR	60.18	31.85	58.58	0.62	52.90	44.98	51.04	0.33
	MoCo V2	61.57	35.06	60.10	0.17	50.62	35.72	48.88	0.17

Our mitigator is compatible with diverse training methods and demonstrates good performance for both small patch triggers and global invisible triggers.

Ablation Study: Data Ratio

Data ratio (%)	CIFAR-10			ImageNet-100		
	TP	FP	DACC(%)	TP	FP	DACC(%)
5	28	7	71	38	6	82
8	37	8	79	40	7	83
10	38	8	80	43	8	85

A larger ratio introduces a higher detection accuracy (DACC).