Robustness

Poison Attacks

Yunzhen Feng, Haocheng Ju, Zehao Wang, Haotong Yang, Pu Yang

School of Mathematical Science, Peking University

Poison Attacks against SVMs

Poison Attacks against SVMs[1]

- Training set $D_{tr} = \{(x_i, y_i)\}_{i=1}^{n}$
- Validation set $D_{val} = \{(x_i, y_i)\}_{i=1}^{m}$
- Soft-margin SVM:

$$\min_{w,b} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{n} \xi_i \tag{1}$$
$$s.t. \quad y_i(w^T x_i + b) \geq 1 - \xi_i, \xi_i \geq 0, i = 1, \cdots, n$$

$$L(w, b, \xi, \alpha, \mu) = \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{n} \xi_i - \sum_{i=1}^{n} \alpha_i (y_i(w^T x_i + b) - 1 + \xi_i) - \sum_{i=1}^{n} \mu_i \xi_i \tag{2}$$

$$w = \sum_{i=1}^{n} \alpha_i y_i x_i$$

Poison Attacks against SVMs

- Kernel $K_{ij} = k(x_i, x_j)$
- Define matrix Q with $Q_{ij} = y_i y_j K_{ij}$
- The training points can be divided into
  - Margin support vectors $S = \{x_s\}$, $\alpha_s \in [0, C]$
  - Error support vectors $E = \{x_e\}$, $\alpha_e = C$
  - Reserve points $R = \{x_r\}$, $\alpha_r = 0$
- Assumptions:
  - Attacker knows the learning algorithm, training data and can draw data from the underlying data distribution.
  - The set S, E, R does not change during the update.

Poison Attacks against SVMs

- Insert one data point (x_c, y_c) into D_{tr}, which most efficiently poison the model performance on D_{val}
- Maximize the hinge loss on D_{val}

$$\max_{x_c} L(x_c) = \sum_{k=1}^{m} (1 - y_k f(x_k))_+ = \sum_{k=1}^{m} (-g_k)_+ \tag{3}$$

- Only consider the sub-terms whose $g_k < 0$

Poison Attacks against SVMs

- Update x_c iteratively: $x_c^{(p)} = x_c^{(p-1)} + t\nabla L(x_c)$
- $\nabla L(x_c) = -\sum_{k:g_k<0} \frac{dg_k}{dx_c}$, $g_k = \sum_j Q_{kj}\alpha_j + y_k b - 1$
- $\frac{dg_k}{dx_c} = \sum_j Q_{kj}\frac{d\alpha_j}{dx_c} + \frac{dQ_{kc}}{dx_c}\alpha_c + y_k\frac{db}{dx_c} = Q_{ks}\frac{d\alpha}{dx_c} + \frac{dQ_{kc}}{dx_c}\alpha_c + y_k\frac{db}{dx_c}$
- $$g_i = \sum_{j\in D_{tr}} Q_{ij}\alpha_j + y_i b - 1 \begin{cases} > 0; i \in R \\ = 0; i \in S \\ < 0; i \in E \end{cases} \quad (4)$$
  $$h = \sum_{j\in D_{tr}} y_j\alpha_j = 0$$
- $$\frac{\partial g}{\partial x_c} = Q_{ss}\frac{\partial \alpha}{\partial x_c} + \frac{\partial Q_{sc}}{\partial x_c}\alpha_c + y_s\frac{\partial b}{\partial x_c} = 0 \quad (5)$$
  $$\frac{\partial h}{\partial x_c} = y_s^T\frac{\partial \alpha}{\partial x_c} = 0$$

Poison Attacks against SVMs

Kernelization: linear kernel, Polynomial kernel, RBF kernel

Algorithm 1 Poisoning attack against SVM

Input: $\mathcal{D}_{\text{tr}}$, the training data; $\mathcal{D}_{\text{val}}$, the validation data; y_c, the class label of the attack point; $x_c^{(0)}$, the initial attack point; t, the step size.

Output: x_c, the final attack point.

1: $\{\alpha_i, b\} \leftarrow$ learn an SVM on $\mathcal{D}_{\text{tr}}$.
2: $k \leftarrow 0$.
3: **repeat**
4: Re-compute the SVM solution on $\mathcal{D}_{\text{tr}} \cup \{x_c^{(p)}, y_c\}$ using incremental SVM (*e.g.*, Cauwenberghs & Poggio, 2001). This step requires $\{\alpha_i, b\}$.
5: Compute $\frac{\partial L}{\partial u}$ on $\mathcal{D}_{\text{val}}$ according to Eq. (10).
6: Set u to a unit vector aligned with $\frac{\partial L}{\partial u}$.
7: $k \leftarrow k + 1$ and $x_c^{(p)} \leftarrow x_c^{(p-1)} + tu$
8: **until** $L\left(x_c^{(p)}\right) - L\left(x_c^{(p-1)}\right) < \epsilon$
9: **return:** $x_c = x_c^{(p)}$

Experiments

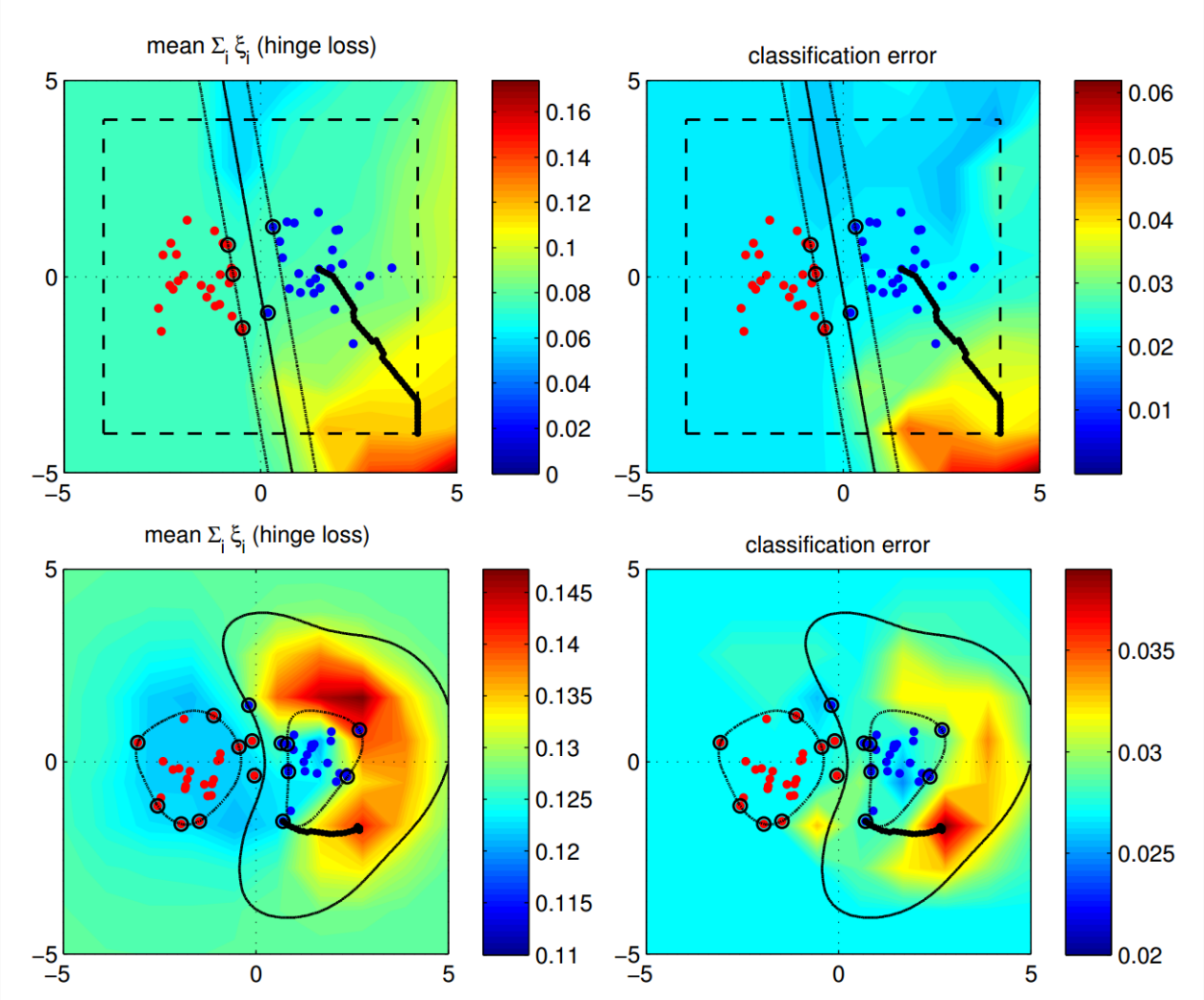


Figure 1: Behavior of the gradient-based attack strategy on the Gaussian data sets, for the linear (top row) and the RBF kernel (bottom row). The regularization parameter C was set to 1 in both cases.

Experiments

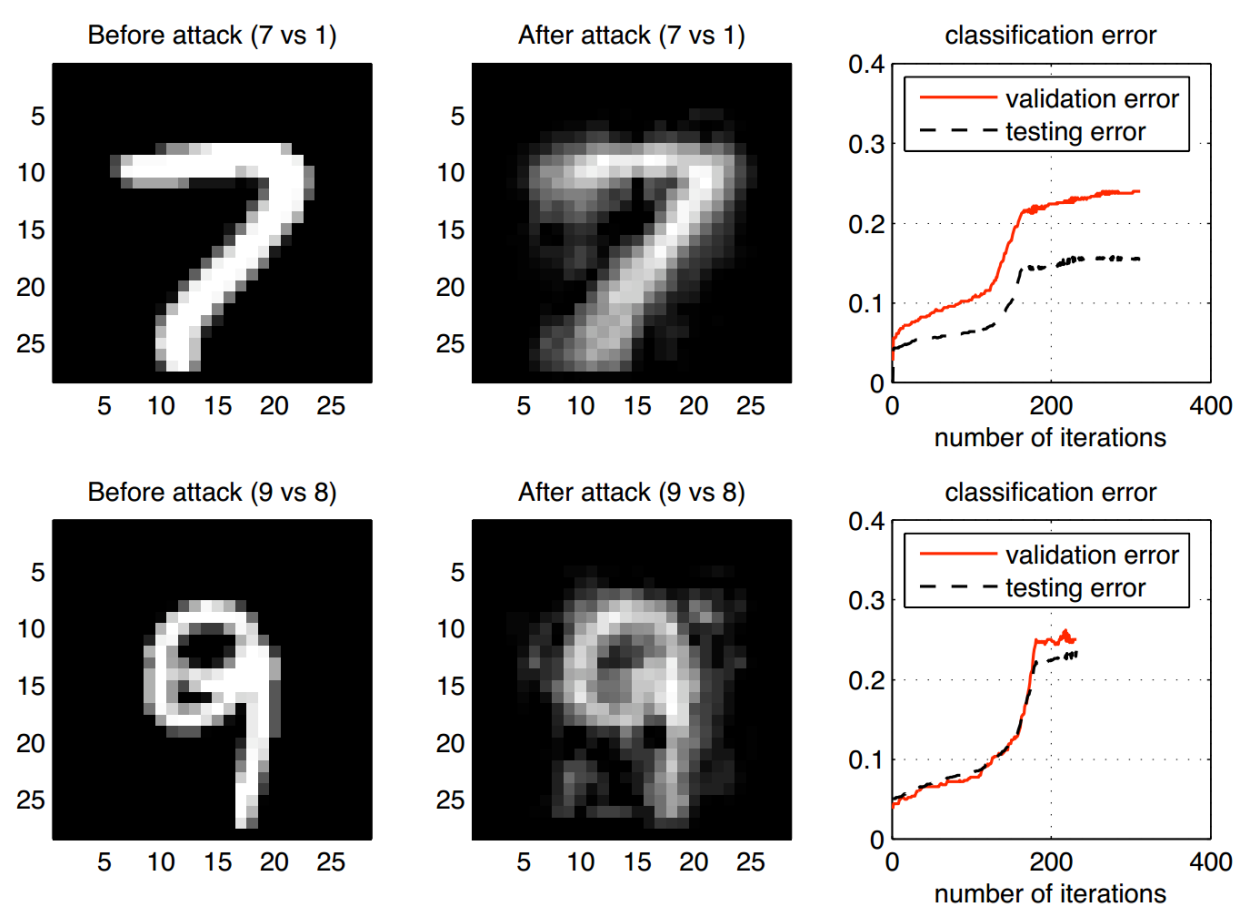


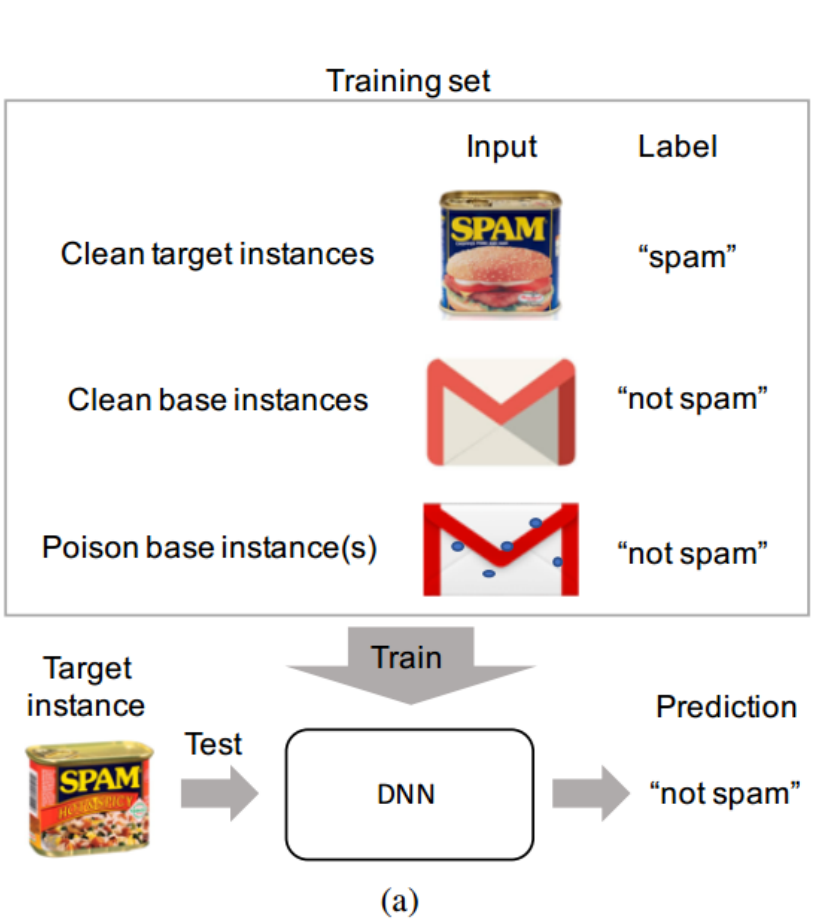
Figure 2: Modifications to the initial (mislabeled) attack point performed by the proposed attack strategy, for the two considered two-class problems from the MNIST data set.

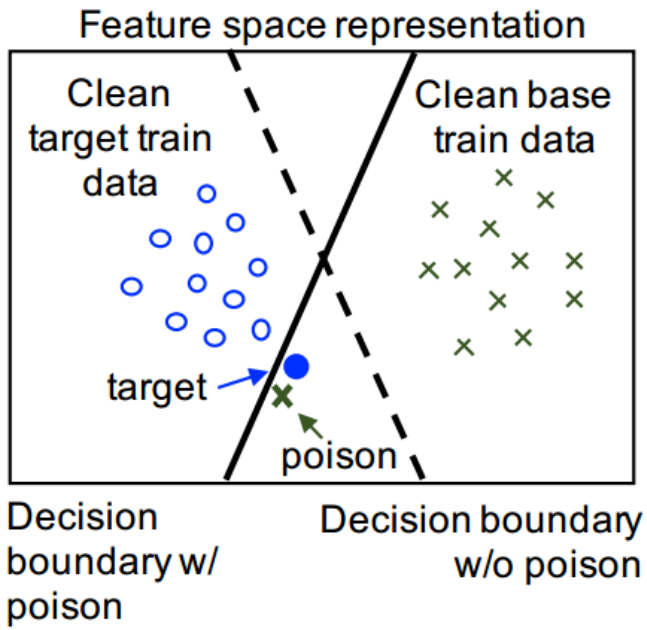
Poison Frogs! Targeted Clean-Label Poisoning Attacks on Neural Networks[2]

Preliminary

- Clean-label attack: injected training examples are cleanly labeled by a certified authority
- Chooses a target instance from the test set, samples a base instance from the base class
- Makes imperceptible changes to the base instance to craft a poison instance
- Inject this poison into the training data with the intent of fooling the model into labelling the target instance with the base label at test time.
- Cause this target example to be misclassified

Schematic





(b) Illustration of the feature space (activations of the penultimate layer before the softmax layer of the network) representation of clean training data, poison instance, and target instance. Note that the target instance is *not* in the training set. Thus it will not affect the loss when the nearby poison instance causes the decision boundary to shift to encompass both of them into the base region.

Crafting poison data via feature collisions

- $f(x)$: output before the softmax layer

$$p = \arg\min_x \|f(x) - f(t)\|^2 + \beta \|x - b\|^2 \tag{6}$$

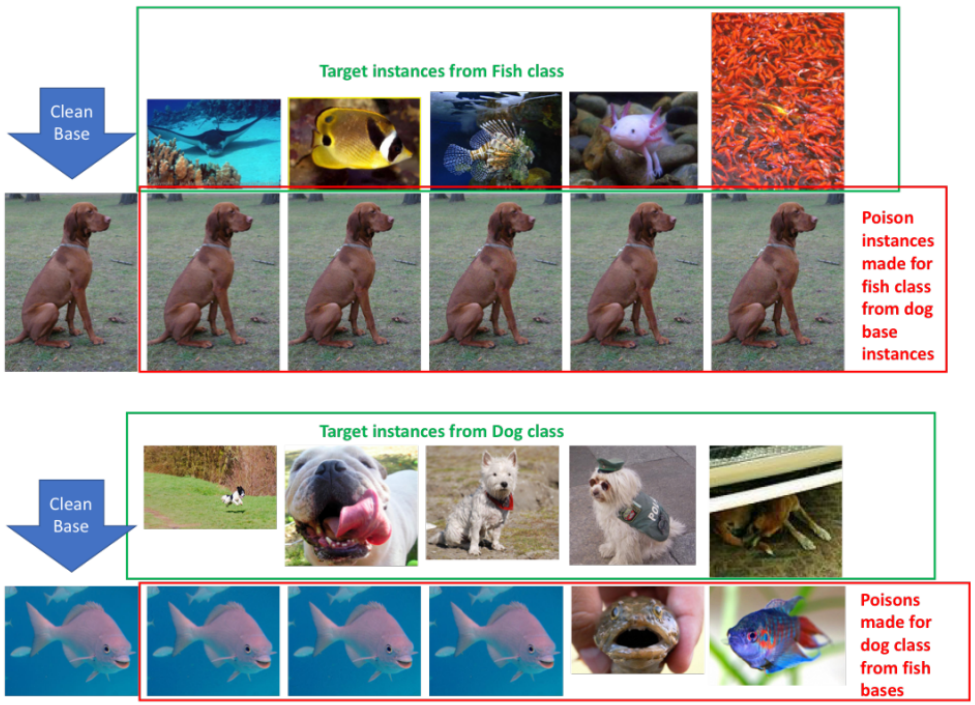
- close to the target t in feature space, close to the base instance b in input space

Algorithm 1 Poisoning Example Generation

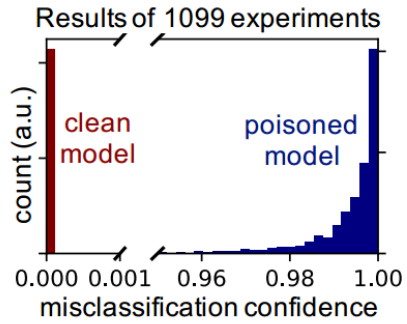
Input: target instance t, base instance b, learning rate λ
Initialize x: $x_0 \leftarrow b$
Define: $L_p(x) = \|f(\mathbf{x}) - f(\mathbf{t})\|^2$
for $i = 1$ **to** $maxIters$ **do**
  Forward step: $\widehat{x_i} = x_{i-1} - \lambda \nabla_x L_p(x_{i-1})$
  Backward step: $x_i = (\widehat{x_i} + \lambda\beta b)/(1 + \beta\lambda)$
end for

Transfer Learning

- Only train the final layer
- Add one poison instance

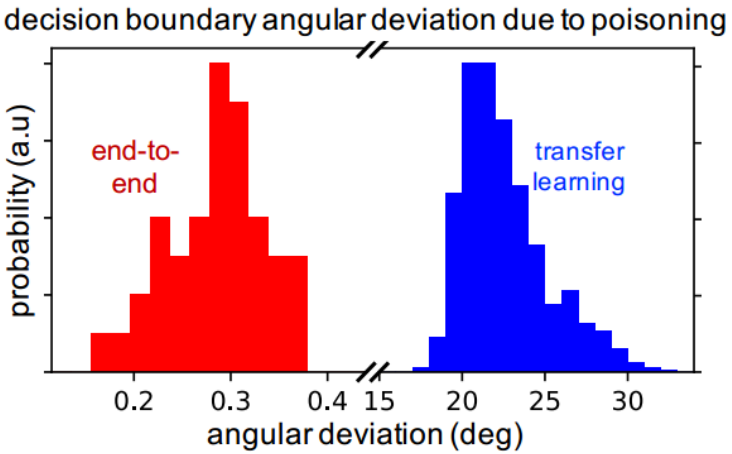


(a) Sample target and poison instances.

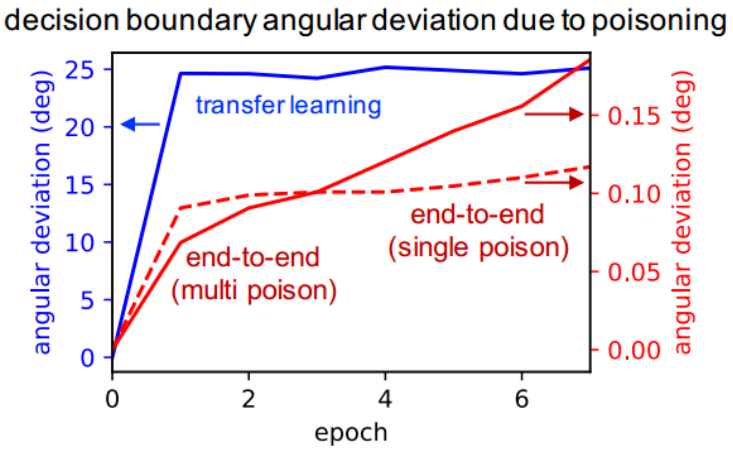


(b) Incorrect class's probability histogram predicted for the target image by the clean (dark red) and poisoned (dark blue) models. When trained on a poisoned dataset, the target instances not only get misclassified; they get misclassified with high confidence.

Transfer Learning



(a) PDF of decision boundary ang. deviation.



(b) Average angular deviation vs epoch.

End-to-end Training

- The decision boundary in the end-to-end training scenario is unchanged after retraining on the poisoned dataset.
- Watermarking: add a low-opacity(γ) watermark of the target instance to the poisoning instance to allow for some inseparable feature overlap while remaining visually distinct

$$b \leftarrow \gamma t + (1 - \gamma) b \tag{7}$$

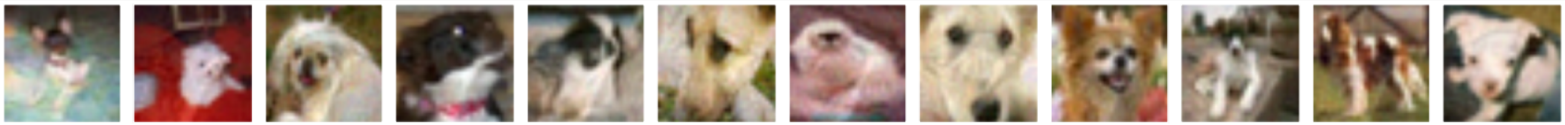


Figure 3: 12 out of 60 random poison instances that successfully cause a bird target instance to get misclassified as a dog in the end-to-end training scenario. An adversarial watermark (opacity 30%) of the target bird instance is applied to the base instances when making the poisons.

End-to-end Training

- Multiple poison instance attacks
- Learning a feature embedding that separates all poison instances from the target is hard.

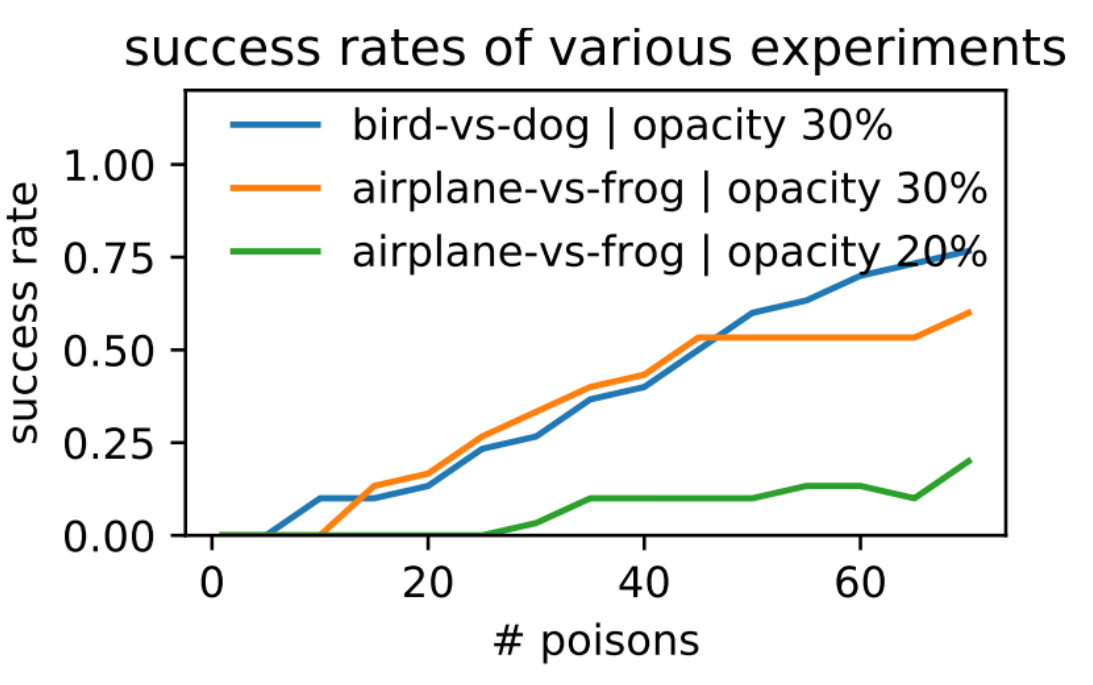


Figure 4: Success rate of attacks on different targets from different bases as a function of number of poison instances used and different target opacity added to the base instances.

Spectral Signatures in Backdoor Attacks[4]

Spectral Signatures in Backdoor Attacks

- Backdoor attacks: correctly classifies clean test inputs, misclassify corrupted test inputs

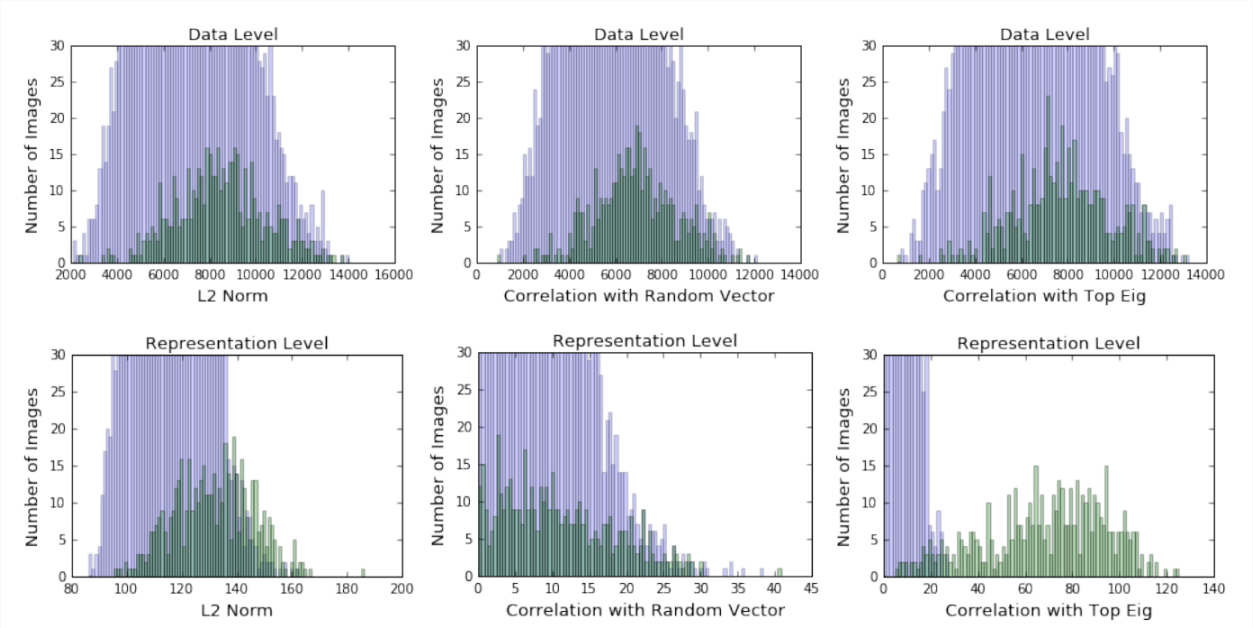
- Contribution: backdoor attacks tend to leave behind a detectable trace in the spectrum of the covariance of a feature representation learned by the neural network.

Spectral Signatures in Backdoor Attacks

Two sub-populations for a given label:

- a large number of clean, correctly labelled inputs
- a small number of corrupted, mislabelled inputs

If the two populations are sufficiently well-separated, the corrupted datapoints can be detected.



Spectral Signatures in Backdoor Attacks

Definition 3.1. Fix $1/2 > \varepsilon > 0$. Let D, W be two distributions with finite covariance, and let $F = (1-\varepsilon)D + \varepsilon W$ be the mixture of D, W with mixing weights $(1-\varepsilon)$ and ε, respectively. We say that D, W are ε-*spectrally separable* if there exists a $t > 0$ so that

$$\Pr_{X \sim D}\left[|\langle X - \mu_F, v\rangle| > t\right] < \varepsilon$$
$$\Pr_{X \sim W}\left[|\langle X - \mu_F, v\rangle| < t\right] < \varepsilon\ ,$$

where v is the top eigenvector of the covariance of F.

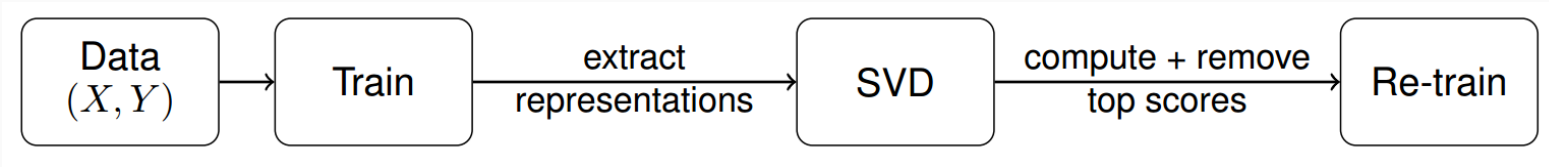
- D: the true distribution over inputs
- W: a small, but adversarially added set of inputs
- If D, W are ε-spectrally separable, by removing the largest ε-fraction of points in the direction of the top eigenvector, we are essentially guaranteed to remove all the data points from W.

Spectral Signatures in Backdoor Attacks

Lemma 3.1. *Fix* $1/2 > \varepsilon > 0$. *Let* D, W *be distributions with mean* μ_D, μ_W *and covariances* $\Sigma_D, \Sigma_W \preceq \sigma^2 I$, *and let* $F = (1-\varepsilon)D + \varepsilon W$. *Then, if* $\|\mu_D - \mu_W\|_2^2 \geq \frac{6\sigma^2}{\varepsilon}$, *then* D, W *are* ε*-spectrally separable.*

- D: the true distribution over inputs
- W: a small, but adversarially added set of inputs
- If the mean of D differs enough from the mean of W, then D, W can be distinguished via spectral methods.

Spectral Signatures in Backdoor Attacks



Algorithm 1

Input: Training set $\mathbb{D}_{\text{train}}$, randomly initialized neural network model $\mathcal{L}$ providing a feature representation $\mathcal{R}$, and upper bound on number of poisoned training set examples ε. For each label y of $\mathbb{D}_{\text{train}}$, let $\mathbb{D}_y$ be the training examples corresponding to that label.
Train $\mathcal{L}$ on $\mathbb{D}_{\text{train}}$.
Initialize $S \leftarrow \{\}$.
for all y **do**
  Set $n = |\,\mathbb{D}_y\,|$, and enumerate the examples of $\mathbb{D}_y$ as $x_1, \ldots, x_n$.
  Let $\widehat{\mathcal{R}} = \frac{1}{n}\sum_{i=1}^{n} \mathcal{R}(x_i)$.
  Let $M = [\mathcal{R}(x_i) - \widehat{\mathcal{R}}]_{i=1}^{n}$ be the $n \times d$ matrix of centered representations.
  Let v be the top right singular vector of M.
  Compute the vector τ of *outlier scores* defined via $\tau_i = \left((\mathcal{R}(x_i) - \widehat{\mathcal{R}}) \cdot v\right)^2$.
  Remove the examples with the top $1.5 \cdot \varepsilon$ scores from $\mathbb{D}_y$.
  $S \leftarrow S \cup \mathbb{D}_y$
end for
$\mathbb{D}_{\text{train}} \leftarrow S$.
Re-train $\mathcal{L}$ on $\mathbb{D}_{\text{train}}$ from a random initialization.
Return $\mathcal{L}$.

Experiments

- Setup: a pair of (attack, target) labels, a backdoor shape (pixel, X, or L), an epsilon (number of poisoned images), a position in the image, and a color for the mark
- Natural evaluation set (all 10000 test images for CIFAR10)
- Poisoned evaluation set (1000 images of the attack label with a backdoor)

Experiments

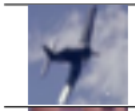
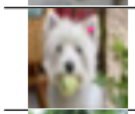
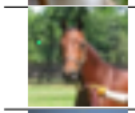
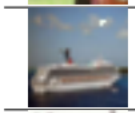
Sample	Target	Epsilon	Nat 1	Pois 1	# Pois Left	Nat 2	Pois 2	Std Pois
	bird	5%	92.27%	74.20%	57	92.64%	2.00%	1.20%
		10%	92.32%	89.80%	7	92.68%	1.50%	
	cat	5%	92.45%	83.30%	24	92.24%	0.20%	0.10%
		10%	92.39%	92.00%	0	92.44%	0.00%	
	dog	5%	92.17%	89.80%	7	93.01%	0.00%	0.00%
		10%	92.55%	94.30%	1	92.64%	0.00%	
	horse	5%	92.60%	99.80%	0	92.57%	1.00%	0.80%
		10%	92.26%	99.80%	0	92.63%	1.20%	
	cat	5%	92.86%	98.60%	0	92.79%	8.30%	8.00%
		10%	92.29%	99.10%	0	92.57%	8.20%	
	deer	5%	92.68%	99.30%	0	92.68%	1.10%	1.00%
		10%	92.68%	99.90%	0	92.74%	1.60%	
	frog	5%	92.87%	88.80%	10	92.61%	0.10%	0.30%
		10%	92.82%	93.70%	3	92.74%	0.10%	
	bird	5%	92.52%	97.90%	0	92.69%	0.00%	0.00%
		10%	92.68%	99.30%	0	92.45%	0.50%	

Figure 5: Natural and poisoned accuracy are reported for two iterations, before and after the removal step. We compare to the accuracy on each poisoned test set obtained from a network trained on a clean dataset (Std Pois).

Experiments

Sample	Target	Epsilon	Nat 1	Pois 1	# Pois Left	Nat 2	Pois 2
	pets	5%	93.99%	95.80%	0	94.18%	0.30%
		10%	94.05%	96.70%	0	94.27%	0.00%
	pets	5%	94.28%	95.00%	0	94.12%	0.20%
		10%	94.13%	99.70%	0	93.89%	0.00%
	pets	5%	94.12%	89.80%	0	94.18%	0.10%
		10%	93.90%	93.40%	0	94.11%	0.10%
	pets	5%	93.97%	94.80%	0	94.42%	0.00%
		10%	94.23%	97.20%	0	93.96%	0.30%
	automobile	5%	93.96%	98.65%	0	94.46%	0.20%
		10%	94.18%	99.20%	0	94.00%	0.20%
	automobile	5%	94.20%	99.15%	0	94.36%	0.25%
		10%	94.03%	99.55%	0	94.03%	0.10%
	automobile	5%	93.89%	94.40%	6	94.20%	0.20%
		10%	94.49%	97.20%	2	94.49%	0.05%
	automobile	5%	94.26%	95.60%	5	94.06%	0.00%
		10%	94.20%	98.45%	1	94.06%	0.15%

Figure 6: Results for a selection of different attack parameters on a combined label of cats and dogs, that we call pets. Natural and poisoned accuracy are reported for two iterations, before and after the removal step.

Certified Defenses for Data Poisoning Attacks[3]

Problem Formulation

- Binary classification with hinge loss $\ell(\theta; x, y) = \max(0, 1 - y\langle\theta, x\rangle)$
- Clean data set D_c with n data points drawn from p^*
- Attacker can insert ϵn poisoned data which form D_p
- The defender trains on the full dataset $D_c \cup D_p$ to produce a model $\hat{\theta}$, and incurs test loss $\boldsymbol{L}(\hat{\theta})$
- Defender: minimize $\boldsymbol{L}(\hat{\theta})$. Attacker: maximize $\boldsymbol{L}(\hat{\theta})$
- Defense: Filter D_p

Data Sanitization

- Use feasible set F to filter some of poisoned data

$$\hat{\theta} = \arg\min_{\theta \in \Theta} L(\theta; (D_c \cup D_p) \cap F), \text{where} \quad L(\theta; S) = \sum_{(x,y) \in S} \ell(\theta; x, y) \tag{8}$$

- Ball Filter

$$F_{sphere} = \{(x, y) : \|x - \mu_y\| \leq r_y\} \tag{9}$$

- Slab filter

$$F_{slab} = \{(x, y) : |\langle x - \mu_y, \mu_y - \mu_{-y} \rangle| \leq s_y\} \tag{10}$$

Data Sanitization

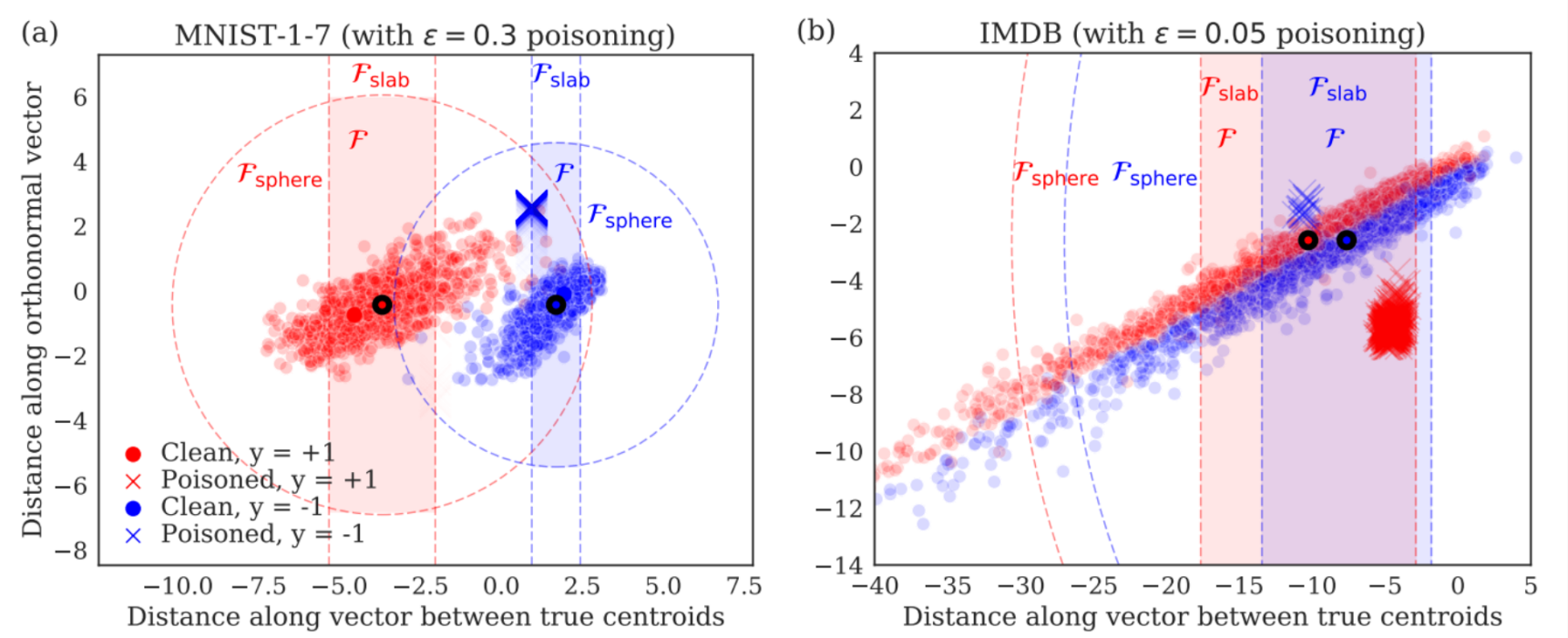


Figure 7: (a) For the MNIST-1-7 dataset, the classes are well-separated and no attack can get past the defense. (b) For the IMDB dataset, the class centroids are not well-separated and it is easy to attack the classifier.

Bounding the worst-case test loss

$$\begin{aligned}
\max_{D_p} \boldsymbol{L}(\hat{\theta}) \approx \max_{D_p} \frac{1}{n} L(\hat{\theta}; D_c) &\leq \max_{D_p} \frac{1}{n} L(\hat{\theta}; D_c \cup (D_p \cap F)) \\
&\approx \max_{D_p} \frac{1}{n} L(\tilde{\theta}; D_c \cup (D_p \cap F)) \qquad (11) \\
&= \max_{D_p \subset F} \min_{\theta \in \Theta} \frac{1}{n} L(\theta; D_c \cup D_p) \triangleq M
\end{aligned}$$

Fixed Defenses

Fixed defenses (F does not depend on D_p):

$$M \leq \min_{\theta \in \Theta} \max_{D_p \subset F} \frac{1}{n} L(\theta; D_c \cup D_p) = \min_{\theta \in \Theta} U(\theta)$$
$$\text{where} \quad U(\theta) = \frac{1}{n} L(\theta; D_c) + \epsilon \max_{(x,y) \in F} \ell(\theta; x, y) \tag{12}$$

Θ : a ℓ_2-ball of radius ρ.

Algorithm 1 Online learning algorithm for generating an upper bound and candidate attack.

Input: clean data $\mathcal{D}_c$ of size n, feasible set $\mathcal{F}$, radius ρ, poisoned fraction ϵ, step size η.
Initialize $z^{(0)} \leftarrow 0$, $\lambda^{(0)} \leftarrow \frac{1}{\eta}$, $\theta^{(0)} \leftarrow 0$, $U^* \leftarrow \infty$.
for $t = 1, \ldots, \epsilon n$ **do**
  Compute $(x^{(t)}, y^{(t)}) = \text{argmax}_{(x,y) \in \mathcal{F}} \ell(\theta^{(t-1)}; x, y)$.
  $U^* \leftarrow \min\left(U^*, \frac{1}{n} L(\theta^{(t-1)}; \mathcal{D}_c) + \epsilon \ell(\theta^{(t-1)}; x^{(t)}, y^{(t)})\right)$.
  $g^{(t)} \leftarrow \frac{1}{n} \nabla L(\theta^{(t-1)}; \mathcal{D}_c) + \epsilon \nabla \ell(\theta^{(t-1)}; x^{(t)}, y^{(t)})$.
  Update: $z^{(t)} \leftarrow z^{(t-1)} - g^{(t)}$, $\lambda^{(t)} \leftarrow \max(\lambda^{(t-1)}, \frac{\|z^{(t)}\|_2}{\rho})$, $\theta^{(t)} \leftarrow \frac{z^{(t)}}{\lambda^{(t)}}$.
end for
Output: upper bound U^* and candidate attack $\mathcal{D}_p = \{(x^{(t)}, y^{(t)})\}_{t=1}^{\epsilon n}$.

Fixed Defenses

$$(x^{(t)}, y^{(t)}) = \arg \max_{(x,y)\in F} \ell(\theta^{(t-1)}; x, y) \tag{13}$$

$$\text{minimize} \quad y\langle \theta, x \rangle \tag{14}$$

$$\text{subject to } \|x - \mu_y\|^2 \leq r_y^2, |\langle x - \mu_y, \mu_y - \mu_{-y} \rangle| \leq s_y$$

Any algorithm whose average regret is small will have a nearly optimal candidate attack D_p.

Proposition 1. *Assume the loss ℓ is convex. Suppose that an online learning algorithm (e.g., Algorithm 1) is used to minimize $U(\theta)$, and that the parameters $(x^{(t)}, y^{(t)})$ maximize the loss $\ell(\theta^{(t-1)}; x, y)$ for the iterates $\theta^{(t-1)}$ of the online learning algorithm. Let $U^* = \min_{t=1}^{\epsilon n} U(\theta^{(t)})$. Also suppose that the learning algorithm has regret $\mathrm{Regret}(T)$ after T time steps. Then, for the attack $\mathcal{D}_p = \{(x^{(t)}, y^{(t)})\}_{t=1}^{\epsilon n}$, the corresponding parameter $\tilde{\theta}$ satisfies:*

$$\frac{1}{n} L(\tilde{\theta}; \mathcal{D}_c \cup \mathcal{D}_p) \leq \mathbf{M} \leq U^* \quad \text{and} \quad U^* - \frac{1}{n} L(\tilde{\theta}; \mathcal{D}_c \cup \mathcal{D}_p) \leq \frac{\mathrm{Regret}(\epsilon n)}{\epsilon n}. \tag{6}$$

Data-Dependent Defenses

- F depends on $D_c \cup D_p$
- Think of D_p as a probability distribution with mass $\frac{1}{\epsilon n}$ on each point in D_p, and relax this to allow any probability distribution π_p.

$$M \leq \min_{\theta \in \Theta} \tilde{U}(\theta) \tag{15}$$

where

$$\tilde{U}(\theta) = \frac{1}{n} L(\theta; D_c) + \epsilon \max_{supp(\pi_p) \subset F(\pi_p)} \mathbb{E}_{\pi_p}[\ell(\theta; x, y)] \tag{16}$$

- Semidefinite program for 2-class SVM

Data-Dependent Defenses

$$\tilde{U}(\theta) = \frac{1}{n} L(\theta; D_c) + \epsilon \max_{supp(\pi_p) \subset F(\pi_p)} \mathbb{E}_{\pi_p}[\ell(\theta; x, y)] \tag{17}$$

- The optimal π_p is supported on at most four points $(x_{a,+}, 1), (x_{b,+}, 1), (x_{a,-}, -1), (x_{b,-}, -1)$
- Replacing the two distinct support vectors which both lie in the positive class both with their midpoint does not affect $F(\pi_P)$ and $\mathbb{E}_{\pi_p}[\ell(\theta; x, y)]$.
- $\mathbb{E}_{\pi_p}[\ell(\theta; x, y)] = \pi_{a,+}(1 - \langle \theta, x_{a,+} \rangle) + \pi_{a,-}(1 + \langle \theta, x_{a,-} \rangle)$

Data-Dependent Defenses

$$\hat{\mu}_y = \frac{p_y \mu_y + \pi_{a,y} x_{a,y} + \pi_{b,y} x_{b,y}}{p_y + \pi_{a,y} + \pi_{b,y}} \tag{18}$$

Constraints:

$$\begin{aligned} |\langle x_{i,y} - \hat{\mu}_y, \hat{\mu}_y - \hat{\mu}_{-y} \rangle| &\leq s_y \\ \langle x_{i,y} - \hat{\mu}_y, x_{i,y} - \hat{\mu}_y \rangle &\leq r_y^2 \\ 1 - y\langle \theta, x_{a,y} \rangle &\geq 0 \\ 1 - y\langle \theta, x_{b,y} \rangle &\leq 0 \end{aligned} \tag{19}$$

- Can be written as linear inequality constraints in the inner products between the 7 vectors $x_{a,+}, x_{a,-}, x_{b,+}, x_{b,-}, \mu_+, \mu_-, \theta$ and solve the SDP
- Randomly sample $\pi_{\{a,b\},\{+,-\}}$ and take the best.

Experiments:Oracle Defenses

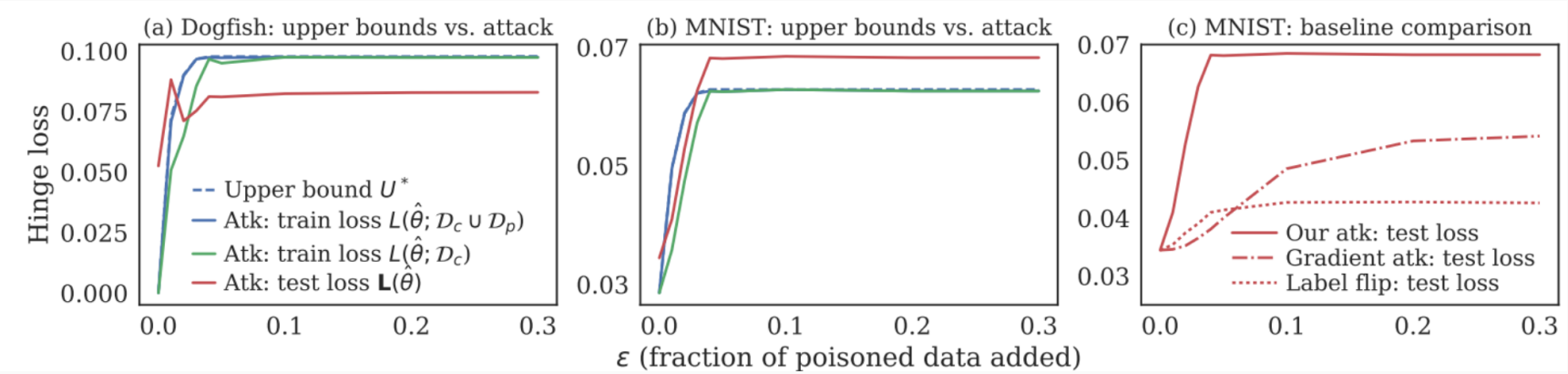


Figure 8: On the (a) Dogfish and (b) MNIST-1-7 datasets, the candidate attack (solid blue) achieves the upper bound (dashed blue) on the worst-case train loss, as guaranteed by Proposition 1. Moreover, this worst-case loss is low; even after adding 30% poisoned data, the loss stays below 0.1. (c) The gradient descent (dash-dotted) and label flip (dotted) baseline attacks are suboptimal under this defense, with test loss (red) as well as test error and train loss (not shown) all significantly worse than the candidate attack.

Experiments:Oracle Defenses

- Text Data: x consists of binary indicator features
- integer quadratic program (IQP)

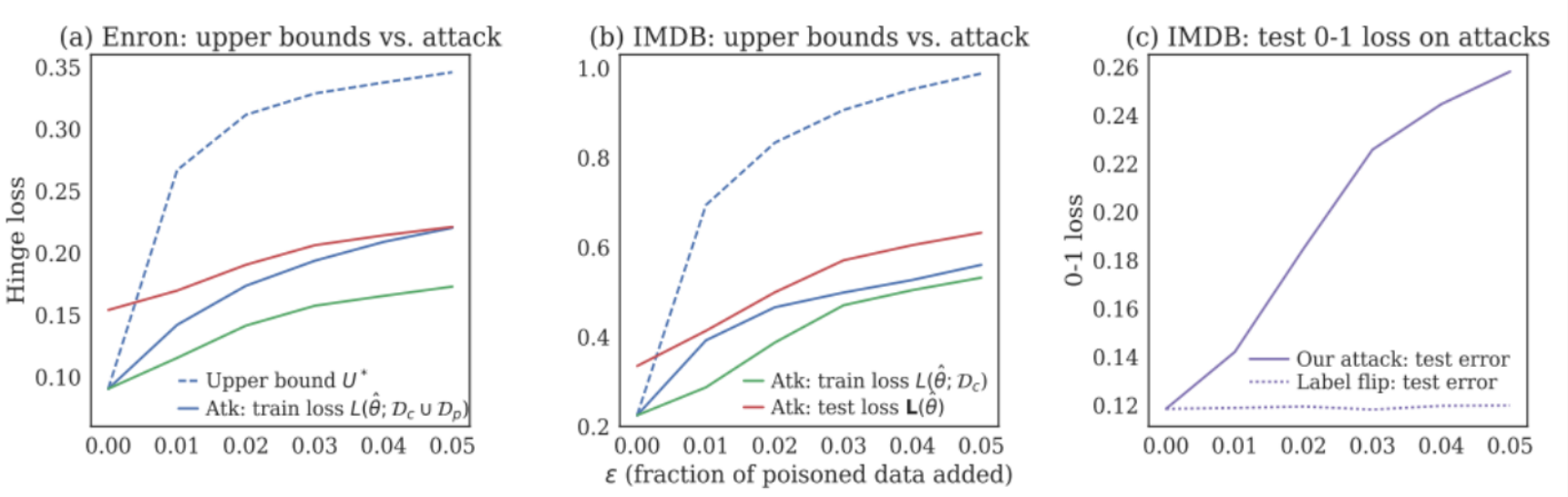


Figure 9: The (a) Enron and (b) IMDB text datasets are significantly easier to attack. (c) In particular, the attack achieves a large increase in test loss (solid red) and test error (solid purple) with small ϵ for IMDB. The label flip baseline was unsuccessful as before. In (a) and (b), note the large gap between upper and lower bounds, resulting from the upper bound relaxation and the IQP.

Experiments:Data-Dependent Defenses

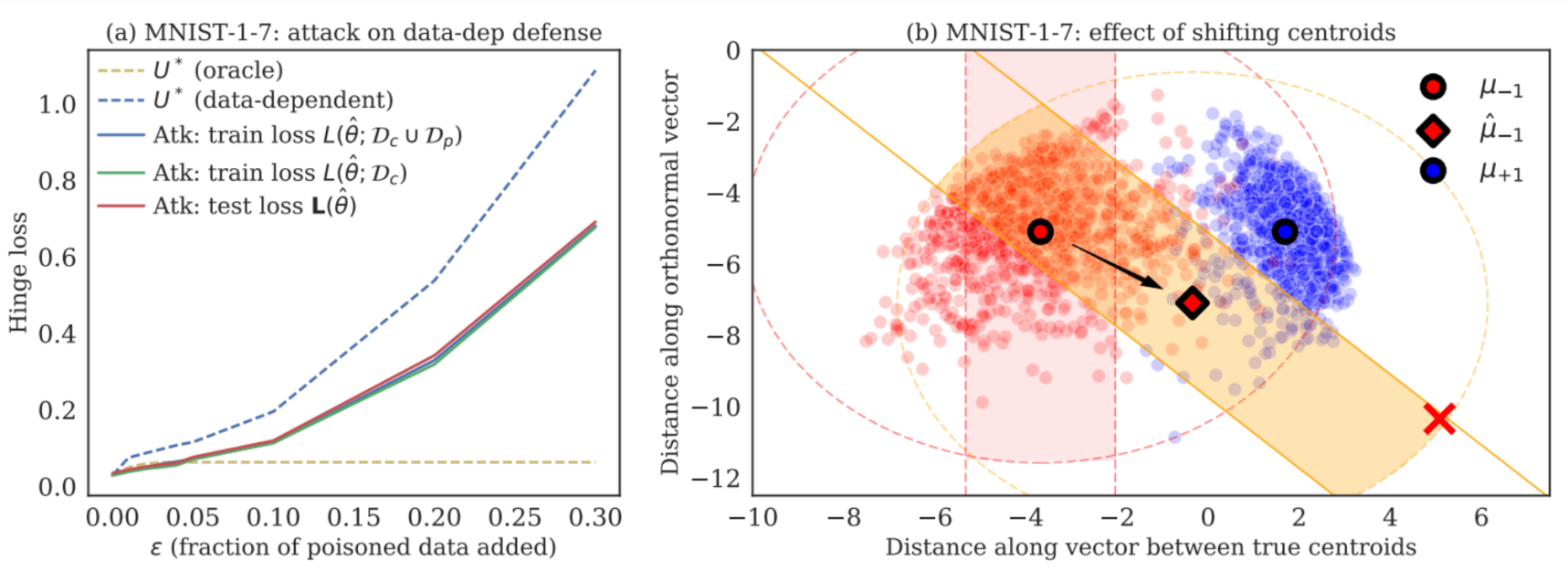


Figure 10: (a) On MNIST-1-7, the attack achieves a test loss of 0.69 (red) and error of 0.40 (not shown) at $\epsilon = 0.3$, more than $10\times$ its oracle counterpart (gold). At low $\epsilon \leq 0.05$, the dataset is safe, with a max train loss of 0.12. (b) Data-dependent sanitization can be significantly poisoned by coordinated adversarial data. The attack for $\epsilon = 0.3$ places almost all of its attacking mass on the red X.

Conclusion

- The bounds tightness depend on the dataset and the defense.
- Text dataset is more vulnerable due to its high dimensionality and abundance of irrelevant features.
- Data dependent defense is vulnerable to attack.

References i


Battista Biggio, Blaine Nelson, and Pavel Laskov.
Poisoning attacks against support vector machines.
In *ICML*, 2012.


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 *NeurIPS*, 2018.


Jacob Steinhardt, Pang Wei Koh, and Percy Liang.
Certified defenses for data poisoning attacks.
In *NIPS*, 2017.

References ii


Brandon Tran, Jerry Li, and Aleksander Madry.
Spectral signatures in backdoor attacks.
In *NeurIPS*, 2018.