STRONG BASELINE DEFENSES AGAINST CLEAN-LABEL POISONING ATTACKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Targeted clean-label poisoning is a type of adversarial attack on machine learning systems where the adversary injects a few correctly-labeled, minimally-perturbed samples into the training data thus causing the deployed model to misclassify a particular test sample during inference. Although defenses have been proposed for general poisoning attacks (those which aim to reduce overall test accuracy), no reliable defense for clean-label attacks has been demonstrated, despite the attacks' effectiveness and their realistic use cases. In this work, we propose a set of simple, yet highly-effective defenses against these attacks. We test our proposed approach against two recently published clean-label poisoning attacks, both of which use the CIFAR-10 dataset. After reproducing their experiments, we demonstrate that our defenses are able to detect over 99% of poisoning examples in both attacks and remove them without any compromise on model performance. Our simple defenses show that current clean-label poisoning attack strategies can be annulled, and serve as strong but simple-to-implement baseline defense for which to test future clean-label poisoning attacks.

1 INTRODUCTION

Machine-learning-based systems are increasingly deployed in settings with high societal impact, such as biometric applications (Sun et al., 2014) and hate speech detection on social networks (Rizoiu et al., 2019), as well as settings with high cost of failure, such as autonomous driving (Chen et al., 2017a) and malware detection (Pascanu et al., 2015). In such settings, robustness to not just noise but also *adversarial manipulation* of system behavior is paramount. Complicating matters is the increasing reliance of machine-learning-based systems on training data sourced from public and semi-public places such as social networks, collaboratively-edited forums, and multimedia posting services. Sourcing data from uncontrolled environments begets a simple attack vector: an adversary can strategically inject data that can manipulate or degrade system performance.

Data poisoning attacks on neural networks occur at training time, wherein an adversary places specially-constructed *poison instances* into the training data with the intention of manipulating the performance of a classifier at test time. Most work on data poisoning has focused on either (i) an attacker generating a small fraction of new training inputs to degrade overall model performance, or (ii) a defender aiming to detect or otherwise mitigate the impact of that attack; for a recent overview, see Koh et al. (2018). In this paper, we focus on *clean-label* data poisoning (Shafahi et al., 2018), where an attacker injects a few *correctly-labeled*, minimally-perturbed samples into the training data. In contrast to traditional data poisoning, these samples are crafted to cause the model to misclassify a particular *target* test sample during inference. These attacks are plausible in a wide range of applications, as they do not require the attacker to have control over the labeling process. The attacker merely inserts apparently benign data into the training process, for example by posting images online which are scraped and (correctly) labeled by human labelers.

Our contribution: In this paper, we initiate the study of defending against *clean-label* poisoning attacks on neural networks. We begin with a defense that exploits the fact that though the raw poisoned examples are not easily detected by human labelers, the *feature representations* of poisons are anomalous among the feature representations for data points with their (common) label. This intuition lends itself to a defense based on k nearest neighbors (k-NN) in the feature space; furthermore, the parameter k yields a natural lever for trading off between the power of the attack against which

it can defend and the impact of running the defense on overall (unpoisoned) model accuracy. Next, we adapt a recent traditional data poisoning defense (Steinhardt et al., 2017; Koh et al., 2018) to the clean-label case, and show that—while still simple to implement—its performance in both precision and recall of identifying poison instances is worse than our proposed defense. We include a portfolio of additional baselines as well. For each defense, we test against state-of-the-art clean-label data poisoning attacks, using a slate of architectures, and show that our initial defense detects nearly all (99%+) of the poison instances without degrading overall performance.

2 Intuition Behind the k-NN Defense

We briefly describe the intuition behind our primary (k-NN-based) clean-label defense. The right side of Figure 1 shows a schematic of how a clean label poison attack typically works, specifically for a targeted misclassification of a plane as a frog. The target image, chosen to be a plane, is represented as a dark gray triangle. In a successful attack, base images of frogs are optimized to lie near the target image of the plane in feature space. Though the optimization causes a qualitative change in the feature representations of the images, in input space the images are perturbed under some small ℓ_2 or ℓ_∞ constraint so that they still appear to be frogs to a human observer. Under the *feature collision* attack (as in Shafahi et al. (2018)), the perturbations are optimized so as to minimize the poison images' distance to the target image in feature space, while under the convex polytope attack (as in Zhu et al. (2019)), the points are optimized to form a convex polytope around the target. Either way, when the model is trained on the dataset with the poisoned inputs, the decision boundary will shift to classify the poison images as frogs, and inadvertently change the target image (which isn't in the training set) from the plane class into the frog class.

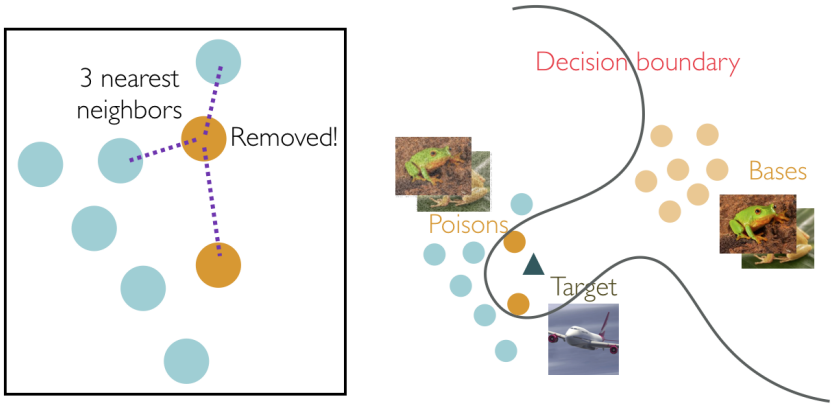


Figure 1: Schematic of the k-NN defense with $k = 3$

However, as seen in the illustrative example with two poisons, the poisons are likely to be surrounded by points of the target class rather than the base class. The inset in Figure 1 illustrates this: when $k = 3$, two poisons will always have 2/3 or more of their neighbors as non-poisons. As illustrated, since the label of the plurality of a poisons neighbors does not match the label of the poison, the poison will be removed from the dataset. More generally, for p poisons, if $k = 2p + 1$, then we would expect the poisons to be outvoted by members of the target class and be removed.

3 Related Work

We briefly overview related work in the space of defenses to adversarial attacks (Biggio et al., 2013; Goodfellow et al., 2014), which are roughly split into evasion attacks (which occur at test time) and data poisoning attacks (which occur at training time). Most adversarial defenses have focused on evasion attacks, where inference-time inputs are manipulated to cause misclassification. In neural networks, evasion adversarial examples are perturbed in such a way that the loss on the victim network increases. The search for an optimal perturbation is facilitated efficiently by use of the local gradient $\nabla_{\mathbf{x}}\mathcal{L}$ obtained via backpropagation on either a white box (victim) network

or a surrogate network if the victim network is unknown (Liu et al., 2016). Many defenses to evasion attacks have leveraged this reliance on gradients by finding ways to obfuscate the gradient, either though non-differentiable layers or reshaping the loss surface such that the gradients vanish or are highly uncorrelated. Athalye et al. (2018) showed that obfuscated gradient defenses are insufficient. Using various strategies to circumvent loss of gradient information, such as replacing non-differentiable layers with differentiable approximations during the backward pass, they showed that stronger attacks can reduce accuracy to near zero on most gradient-based defenses. The defenses that withstand strong attacks are those whose loss surface with respect to the input tends to be "smooth" everywhere in the data manifold. To that end, variants of adversarial training (Madry et al., 2017; Xie et al., 2019; Shafahi et al., 2019) and linearity or curvature regularizers (Qin et al., 2019; Moosavi-Dezfooli et al., 2019) have maintained modest accuracies in the midst of strong multi-iteration PGD attacks (Madry et al., 2017).

In evasion attacks, A deep k-NN-based methodology has been used across multiple layers of a neural network to generate confidence estimates of network predictions to create identify adversarial examples (Papernot & McDaniel, 2018). Our k-NN-based defense differs in that it identifies data at training time rather than at test time, so that it uses true labels rather than predicted labels. Further, a soft nearest neighbor regularizer has been used during training time to improve robustness to evasion examples Frosst et al. (2019), but its resistance to clean-label poisoning examples has yet to be explored.

Backdoor attacks have recently gained interest as a realistic threat to machine learning models. Backdooring, proposed by Gu et al. (2017), can be seen as a subset of data poisoning. In their simplest form, backdoor attacks modify a small number of training examples with a specific pattern, the *trigger*, which accompanies examples with a specific *target* label. Leveraging the fact that neural networks tend to memorize training patterns, the attacker then puts the trigger onto examples during inference time that she wants classified—or misclassified—as the target. The trigger need not change the ground truth label of the training data, making such attacks clean-label attacks (Turner et al., 2019). Crucially however, these attacks rely upon the attacker being able to modify data at inference time—an assumption that may not always be realistic, and one we do not make in this paper.

A number of defenses to backdoor attacks and poisons have been proposed. Many defenses seek to sanitize training data—remove poisons from the training data to neutralize the poisons' effects. Often, these defenses rely upon the heuristic that backdoor attacks create "shortcuts" in the network to induce target misclassification. An early example of such a defense employed two variants of an ℓ_2 centroid defense (Steinhardt et al., 2017), which we adapt to our setting in this paper. In one variant, data would be removed from training if it fell outside of an acceptable radius in feature space. The other variant first projected the feature representations onto the line connecting class centroids and removed data based on its position upon this line.

Along this vein, another defense proposed using feature clustering for data sanitization (Chen et al., 2018). This defense relies upon the assumption that naive backdoor triggers will cause identifiable clusters of poisons to form in feature space. The intuition of this defense also fails when exposed to stronger poisoning methods which do not use uniform triggers. In these stronger poisoning attacks, it has been shown that the poisoned data causes misclassification by more subtle means like surrounding a target image in feature space with a convex polytope of poisons (Zhu et al., 2019). Such an attack will not always result in an easily identifiable clusters of poisons.

Still other defenses seek to identify and reconstruct the trigger that causes misclassification (Wang et al., 2019). In this defense, called *NeuralCleanse*, inputs would be removed depending upon whether the input's activation was similar to the activations induced by the reconstructed trigger. This defense was able to detect certain uniform ℓ_0 triggers inserted in training data using methods such as neuron activation clustering. However, this tactic does not apply to recent insidious poisoning attacks which use variable, learned perturbations to input data that cause misclassification via feature collisions (Shafahi et al., 2018; Zhu et al., 2019).

4 BASELINE DEFENSES AGAINST CLEAN-LABEL POISONING ATTACKS

Here we introduce our set of baseline defenses, which will then be compared in a series of controlled experiments in Section 5.

4.1 NOTATION

We use $\boldsymbol{x}_t$ to denote the input space representation of the target image that a data poisoner tries to misclassify; the target has true label l_t but the attacker seeks to misclassify it as having label l_b. We use $\boldsymbol{x}_b$ to denote a base image that is used to build a poison after optimization with label l_b. We use $\boldsymbol{x}_w$ to denote a base image watermarked with a target image, that is $\gamma \cdot \boldsymbol{x}_t + (1-\gamma) \cdot \boldsymbol{x}_b$—to a human observer this image or slight perturbations of the image will have label l_b for sufficiently low values of γ. We use $\phi(\boldsymbol{x})$ to denote the activation of the penultimate layer (before the softmax layer) of a neural network used for classification with multiple labels. We will refer to this as the *feature layer* (or *feature space*) and $\phi(\boldsymbol{x})$ as *features* of $\boldsymbol{x}$.

4.2 k-NN DEFENSE

The k-nearest neighbors (k-NN) defense takes the plurality vote amongst the labels of a point's k nearest neighbors in feature space. If the point's assigned label is not the mode amongst labels of the k nearest neighbors, the point is discarded as being anomalous, and is not used in training the model. A more formal description is in Algorithm 1.

Algorithm 1: k-NN Defense

Result: Filtered training set $X^{train'}$
Let $S_k(x^{(i)})$ denote a set of k points such that for all points $x^{(j)}$ inside the set and points $x^{(l)}$ outside the set, $|\phi(x^{(l)}) - \phi(x^{(i)})|_2 \geq |\phi(x^{(j)}) - \phi(x^{(i)})|_2$
$X^{train'} \leftarrow \{\}$
for *Data points* $\boldsymbol{x}^{(i)} \in X^{train}$ **do**
  Let l denote the label of $x^{(i)}$ and let $l(S_k(x^{(i)}))$ denote the labels of the points in $S_k(x^{(i)})$
  if $l \in mode(l(S_k(x^{(i)})))$ **then**
    $X^{train'} \leftarrow X^{train'} \cup \{x^{(i)}\}$;
  else
    Omit $x^{(i)}$ from $X^{train'}$;
  end
end

Note that if there are p poisons, then by setting $k = 2p + 1$, the poisoned label is unlikely to be the majority, but may still be in the mode of the k-NN set, if the nearest neighbors of $x^{(i)}$ are in multiple classes. However, empirically we do not observe this to be an issue, as seen in Section 5.

The selection of k inherently introduces a tradeoff between removing more poisons (and reducing the probability of a successful attack) and a loss of accuracy as more unpoisoned sample points are removed from the training set.

While the most näive implementation of the k-NN defense may be memory intensive on larger datasets, it can be adapted, e.g. by sampling to require less memory. There is also an entire literature on fast k-NN methods, such as the Barnes & Hut (1986) algorithm, which aims to bring down the time complexity of the nearest neighbor search.

4.3 BASELINE DEFENSES

4.3.1 L2 DEFENSE

The defense "L2" removes an $\epsilon > 0$ fraction of points that are farthest in feature space from the centroids of their classes. For each class of label $l \in \mathcal{L}$, with size $s_l = |x^{(j)} \text{ s.t. } l(j) = l|$, we

compute the centroid c_l as

$$c_l = \frac{1}{s_l} \sum_{x^{(j)} s.t. l(j)=l} \phi(x^{(j)})$$

and remove the $\lfloor \epsilon s_l \rfloor$ points maximizing $|\phi(x^{(j)}) - c_l|_2$.

The L2 defense relies on the calculation of the position of the centroid to filter outliers; this calculation itself is prone to data poisoning if the class size is small. The defense is adapted from traditional poison defenses not specific to neural networks (Koh et al., 2018).

4.3.2 ROBUST FEATURE EXTRACTORS

We test the use of robust feature extractors, that is neural networks trained with adversarial training as a defense. Specifically, we train with adversarial examples constructed with SGD with an 8-step attack bounded by an ℓ_∞ ball of radius 8/255. We would expect the mechanism of this defense to be different from the other defenses, as it does not attempt to filter the poisons out. Instead, before retraining for the feature collision attack the deep features of the poisons would not be near the target. For the convex polytope attack, we would expect that the deep features would fail to form a convex polytope around the target datapoint.

Slower training and reduced classification accuracy for non-adversarial inputs are potential drawbacks of a defense using a robust feature extractor.

4.3.3 ONE-CLASS SVM DEFENSE

The one-class SVM defense examines the deep features of each class in isolation. That is, it applies the one-class SVM algorithm (Schölkopf et al., 2001) to identify outliers in feature space for each label in the training set. It utilizes a radial basis kernel and is calibrated to use a value $\nu = 0.01$ to identify outliers to be filtered out.

4.3.4 RANDOM DEFENSE

The random defense is a simple experimental control. It filters out a random subset of all training data, calibrated to be 1% on the feature collision attack and 10% on the convex polytope attack. If the poisoning attack is sensitive to poisons being removed, the random defense may be successful, at the cost of losing a proportionate amount of the unpoisoned training data.

5 EXPERIMENTS

The robustness of image classification models with the k-NN defense and other anomaly detection defenses are tested on the CIFAR-10 dataset (Krizhevsky et al., 2009).

For poisoning examples, we reproduced the attack experiments from the original *feature collision* attack Shafahi et al. (2018) and *convex polytope* attack Zhu et al. (2019). We verified that generated poisons did indeed cause misclassification of the target after network retraining at the same success rate as reported in the orginal papers. All architectures and training setups from those experiments used in the attacks are used identically in our experiments.

5.1 FEATURE COLLISION DEFENSE EXPERIMENTS

The first attack, as in Shafahi et al. (2018), generates 50 poisons as follows: We randomly select 50 images in the base class. For each base image with input representation $\boldsymbol{x}_b$, we compute the watermark base $\boldsymbol{x}_w \leftarrow \gamma \cdot \boldsymbol{x}_t + (1-\gamma) \cdot \boldsymbol{x}_b$, then optimize p with initial value $\boldsymbol{w}$ as in Algorithm 1 in Shafahi et al. (2018), that is using a forward-backward splitting procedure to solve

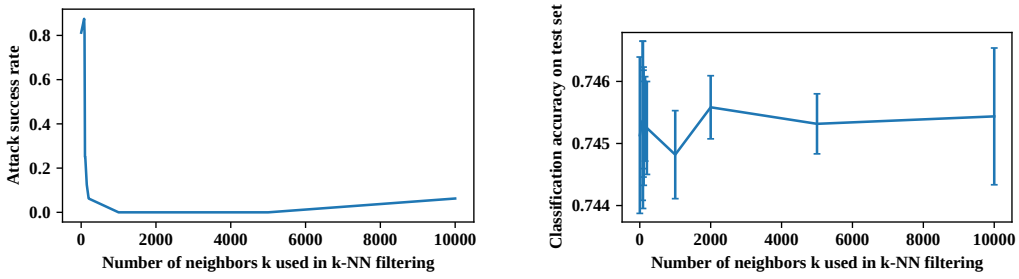
$$p = \arg\min_{\boldsymbol{x}} |\phi(\boldsymbol{x}) - \phi(\boldsymbol{x}_t)|_2^2 + \beta |\boldsymbol{x} - \boldsymbol{x}_w|_2^2$$

The hyperparameter β is tuned to be 0.1. The resulting poisons p are close to the target image in feature space, but close to the watermarked image in input space. For the purposes of evaluation we

Table 1: Results of data poisoning defense on end-to-end Poison Frogs attack

Defense	Poisons filtered	Non poisons filtered (%)	Attack success rate (%)	Class. accuracy (%)
k-NN (k=5000)	799/800	0.6	0.0	74.6
L2 Norm outliers	395/800	1.0	50	74.6
1-class SVM	168/800	1.0	62.5	74.5
Random filtering	84/800	10.0	87.5	74.5

only consider successful attacks, so that the undefended attack success rate is 100%. As in the original paper (Shafahi et al., 2018), the network used is a modified Alexnet. The network is trained with the poisons from a warm start over 10 epochs with a batch size of 128. We evaluate the performance of the defenses described in Section 4 against collections of 50 poisons that successfully create a targeted misclassification, so that the undefended attack success rate is 100%. The results are shown in Table 1. Successful attacks must result in a targeted misclassification. The k-NN defense with k=5000 successfully identifies all but one poison across multiple attacks, while filtering just 0.6% of non-poison datapoints from the input data set. The k-NN defense reduces the attack success rate to 0%. Because classification accuracy surprisingly does not change significantly as k increases it is appropriate to set k as the class size of the training data, as shown in Figure 2. For $k = 3$ the percentage of filtered non-poisons was 0.004%, while for $k = 10000$ it remained just 1.3%. The relatively low filtering rate of nonpoisons for higher k may account for the low variation in test set classification accuracy for different selections of k. The L2 defense also identifies roughly half of the poisons, with $\epsilon = 0.01$, the proportion of the training data that are poisons.



(a) Attack success rate vs. k

(b) Classification accuracy vs. k

Figure 2: Tradeoffs for the selection of k in the k-NN defense for a representative Poison Frogs attack. For small k, the poisons may be all filtered out, while for large k, classification accuracy on the test set may suffer due to the reduced training set size. Surprisingly we find that large k does not noticeably reduce the classification accuracy.

5.2 Transferable Poisoning Defense Experiments

The convex polytope attack, as in in Zhu et al. (2019) generates 5 poisons that are crafted to attack multiple architectures. The experiments are on the transfer learning attack. There are eight architectures: two of which were not used in crafting the poisons (black box setting), and six architectures which use different random numbers/seeds (grey box setting). The grey-box architectures are DPN92 (Chen et al., 2017b), GoogLeNet (Szegedy et al., 2015), MobileNetV2 (Sandler et al., 2018), ResNet50 (He et al., 2016), ResNeXT29-2x64d (Xie et al., 2017), and SENet18 (Hu et al., 2018), while the black-box architectures are DenseNet121 (Huang et al., 2017) and ResNet18 (He et al., 2016).

The aggregate results of a defense on all 8 architectures are shown in Table 2. Both the k-NN defense and the L2 defense filter out virtually all the poisons, with modest filtering of 4.3% and 9.1%, respectively of non-poisons.

Table 2: Results of data poisoning defense on transfer learning of Convex Polytope Attack

Defense	Num poisons filtered	Non poisons filtered (%)	Attack success rate (%)	Class. accuracy (%)
k-NN (k=50)	510/510	4.3	0 .0	93.9
L2 Norm outliers	509/510	9.1	1.0	93.4
1-class SVM	114/510	7.1	70.1	91.7
Random filtering	47/510	10.0	66.8	91.3

The attack success rates for each model architecture (undefended is 100%) are shown in Figure 3. The attack success rate is so low for the k-NN defense and the L2 defense that the bars are not visible, except for GoogLeNet for the L2 defense.

Surprisingly, on the black-box architectures of DenseNet121 and ResNet18, the 1-class SVM defense does not perform better. Indeed, this is despite filtering a higher percentage of poisons–31% for DenseNet and 24% for ResNet18, compared with 22% overall.

A feature-space visualization of the k-NN defense, showing the filtered poisons and non-poisons is shown in Figure 4. Specifically, Figure 4 shows a projected visualization in feature space of the fine tuning data points in the target (blue) and base (green) classes. Following the projection scheme of Shafahi et al. (2018), where the x-axis is the direction of the difference between the centroids of the points in the two classes and the y-axis is component of the parameter vector (i.e. decision boundary) orthogonal to the between-centroids vector, the deep features of the DPN92 network are projected into a two-dimensional plane. The "x" markers denote points that are filtered out by the defense. All the poisons around the target are filtered, as are outlying points in the target class. Interestingly, no points in the base class are filtered in this case.

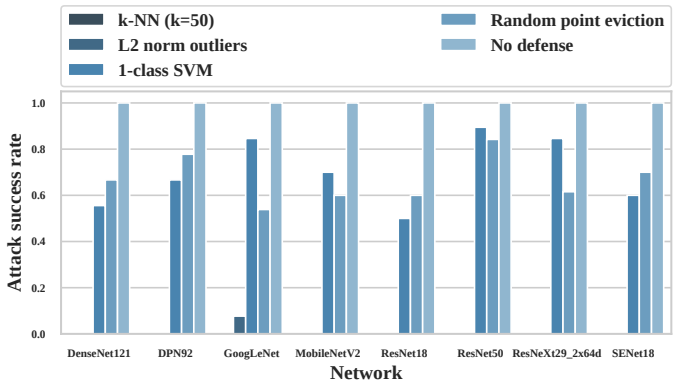


Figure 3: Attack success rate on transfer learning of convex polytope attack. The success rate of the attack when using the k-NN defense is so low that the bar is barely visible.

Figure 5 shows a feature space visualization of the robust feature extractor defense on ResNet18 causing a failed attack as well as a feature space visualization of a standard ResNet18. As expected, the attack fails because the poisons fail to approach the target in feature space, and thus do not form a convex polytope around the target. The defense does hurt test set accuracy, which drops to 79.8%, compared with $> 92\%$ with the same architecture for the other defenses.

The normalized distance from the poisons to target for a robust and normal ResNet (each layers distance is normalized by the average norm of its feature vectors) is shown in Figure 6.

6 Conclusion

In summary, we have demonstrated that the simple k-NN baseline approach provides an effective defense against clean-label poisoning attacks with minimal degradation in model performance. The

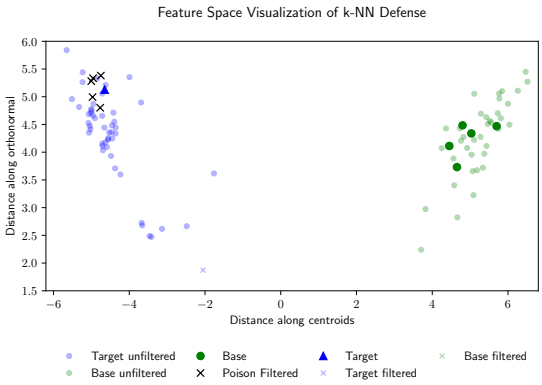
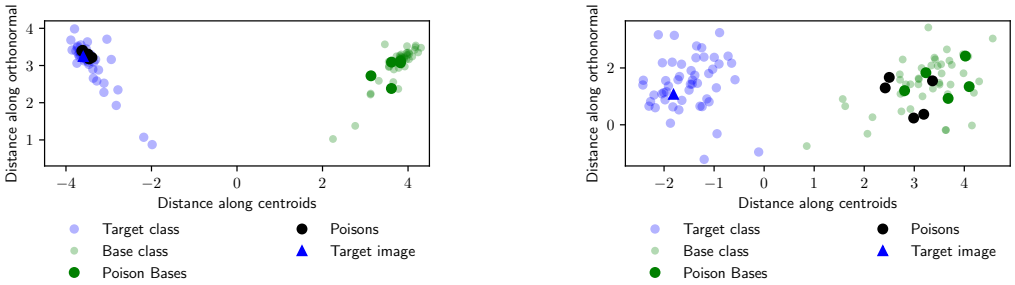


Figure 4: Feature space visualization of transfer learning poisoning k-NN defense - DPN92 (Chen et al., 2017b).



(a) Standard ResNet18

(b) Robust ResNet18

Figure 5: Feature space visualization of transfer learning poisoning - ResNet18 (Chen et al., 2017).

k-NN defense mechanism identifies virtually all poisons from two state-of-the-art clean label data poisoning attacks, while only filtering a small percentage of non-poisons. The k-NN defense outperforms other simple baselines against the existing attacks; these defenses provide benchmarks that could be used to measure the efficacy of future defense-aware clean label attacks.

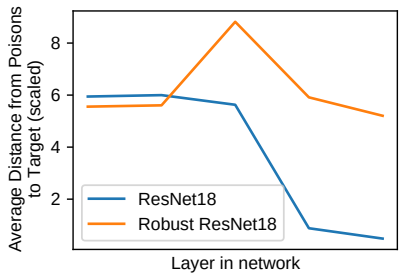


Figure 6: Normalized distance from poison to target for ResNet18 and Robust ResNet18

REFERENCES

Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *arXiv preprint arXiv:1802.00420*, 2018.

Josh Barnes and Piet Hut. A hierarchical o (n log n) force-calculation algorithm. *nature*, 324(6096): 446, 1986.

Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion attacks against machine learning at test time. In *Joint European conference on machine learning and knowledge discovery in databases*, pp. 387–402. Springer, 2013.

Bryant Chen, Wilka Carvalho, Nathalie Baracaldo, Heiko Ludwig, Benjamin Edwards, Taesung Lee, Ian Molloy, and Biplav Srivastava. Detecting backdoor attacks on deep neural networks by activation clustering. *arXiv preprint arXiv:1811.03728*, 2018.

Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1907–1915, 2017a.

Yunpeng Chen, Jianan Li, Huaxin Xiao, Xiaojie Jin, Shuicheng Yan, and Jiashi Feng. Dual path networks. In *Advances in Neural Information Processing Systems*, pp. 4467–4475, 2017b.

Nicholas Frosst, Nicolas Papernot, and Geoffrey Hinton. Analyzing and improving representations with the soft nearest neighbor loss. *arXiv preprint arXiv:1902.01889*, 2019.

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

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

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7132–7141, 2018.

Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2261–2269. IEEE, 2017.

Pang Wei Koh, Jacob Steinhardt, and Percy Liang. Stronger data poisoning attacks break data sanitization defenses. *arXiv preprint arXiv:1811.00741*, 2018.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks. *arXiv preprint arXiv:1611.02770*, 2016.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Jonathan Uesato, and Pascal Frossard. Robustness via curvature regularization, and vice versa. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9078–9086, 2019.

Nicolas Papernot and Patrick McDaniel. Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning. *arXiv preprint arXiv:1803.04765*, 2018.

Razvan Pascanu, Jack W Stokes, Hermineh Sanossian, Mady Marinescu, and Anil Thomas. Malware classification with recurrent networks. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1916–1920. IEEE, 2015.

Chongli Qin, James Martens, Sven Gowal, Dilip Krishnan, Alhussein Fawzi, Soham De, Robert Stanforth, Pushmeet Kohli, et al. Adversarial robustness through local linearization. *arXiv preprint arXiv:1907.02610*, 2019.

Marian-Andrei Rizoiu, Tianyu Wang, Gabriela Ferraro, and Hanna Suominen. Transfer learning for hate speech detection in social media. In *Conference on Artificial Intelligence (AAAI)*, 2019.

Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. *arXiv preprint arXiv:1801.04381*, 2018.

Bernhard Schölkopf, John C Platt, John Shawe-Taylor, Alex J Smola, and Robert C Williamson. Estimating the support of a high-dimensional distribution. *Neural computation*, 13(7):1443–1471, 2001.

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*, pp. 6103–6113, 2018.

Ali Shafahi, Mahyar Najibi, Amin Ghiasi, Zheng Xu, John P. Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! In *Advances in Neural Information Processing Systems*, 2019.

Jacob Steinhardt, Pang Wei Koh, and Percy Liang. Certified defenses for data poisoning attacks. *CoRR*, abs/1706.03691, 2017. URL `http://arxiv.org/abs/1706.03691`.

Yi Sun, Xiaogang Wang, and Xiaoou Tang. Deep learning face representation from predicting 10,000 classes. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1891–1898, 2014.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.

Alexander Turner, Dimitris Tsipras, and Aleksander Madry. Clean-label backdoor attacks, 2019. URL `https://people.csail.mit.edu/madry/lab/cleanlabel.pdf`.

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. *Neural Cleanse: Identifying and Mitigating Backdoor Attacks in Neural Networks*, pp. 0, 2019.

Cihang Xie, Yuxin Wu, Laurens van der Maaten, Alan L Yuille, and Kaiming He. Feature denoising for improving adversarial robustness. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 501–509, 2019.

Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pp. 5987–5995. IEEE, 2017.

Chen Zhu, W Ronny Huang, Hengduo Li, Gavin Taylor, Christoph Studer, and Tom Goldstein. Transferable clean-label poisoning attacks on deep neural nets. In *International Conference on Machine Learning*, pp. 7614–7623, 2019.

A APPENDIX

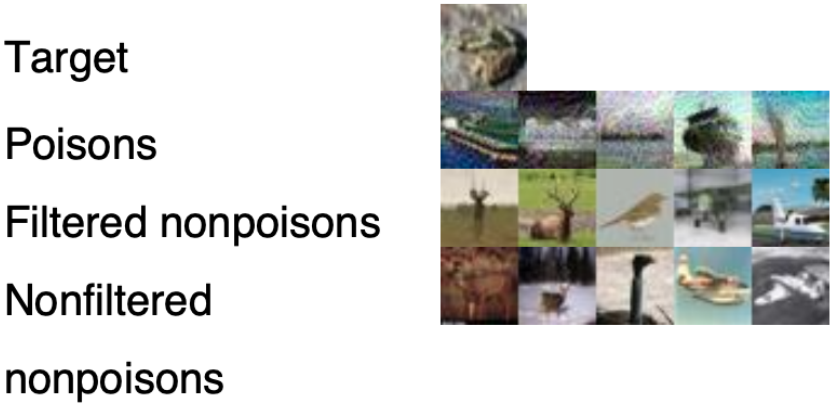


Figure 7: Representative images from the attack and k-NN defense are shown. The poisons in the ship class do not visually look aberrant for members of the ship class, however the k-NN defense filters them. In the bottom two rows, filtered and non-filtered nonpoisons are shown–again there are not visually distinctive differences between pictures in the same class that are filtered rather than not filtered.