On the transferability of adversarial examples between convex and 01 loss models

1st Yunzhe Xue
Department of Computer Science
New Jersey Institute of Technology
Newark, USA
yx277@njit.edu

2nd Meiyan Xie
Department of Computer Science
New Jersey Institute of Technology
Newark, USA
mx42@njit.edu

3rd Usman Roshan
Department of Computer Science
New Jersey Institute of Technology
Newark, USA
usman@njit.edu

***Abstract*—The 01 loss gives different and more accurate boundaries than convex loss models in the presence of outliers. Could the difference of boundaries translate to adversarial examples that are non-transferable between 01 loss and convex models? We explore this empirically in this paper by studying transferability of adversarial examples between linear 01 loss and convex (hinge) loss models, and between dual layer neural networks with sign activation and 01 loss vs sigmoid activation and logistic loss. We first show that white box adversarial examples do not transfer effectively between convex and 01 loss and between 01 loss models compared to between convex models. As a result of this non-transferability we see that convex substitute model black box attacks are less effective on 01 loss than convex models. Interestingly we also see that 01 loss substitute model attacks are ineffective on both convex and 01 loss models mostly likely due to the non-uniqueness of 01 loss models. We show intuitively by example how the presence of outliers can cause different decision boundaries between 01 and convex loss models which in turn produces adversaries that are non-transferable. Indeed we see on MNIST that adversaries transfer between 01 loss and convex models more easily than on CIFAR10 and ImageNet which are likely to contain outliers. We show intuitively by example how the non-continuity of 01 loss makes adversaries non-transferable in a dual layer neural network. We discretize CIFAR10 features to be more like MNIST and find that it does not improve transferability, thus suggesting that different boundaries due to outliers are more likely the cause of non-transferability. As a result of this non-transferability we show that our dual layer sign activation network with 01 loss can attain robustness on par with simple convolutional networks.**

***Index Terms*—adversarial attacks, transferability of adversarial examples, 01 loss, stochastic coordinate descent, convolutional neural networks, deep learning**

I. Introduction

State of the art machine learning algorithms can achieve high accuracies in classification tasks but misclassify minor perturbations in the data known as as adversarial attacks [1]–[5]. Adversarial examples have been shown to transfer across models which makes it possible to perform transer-based (substitute model) black box attacks [6]. To counter adversarial attacks many defense methods been proposed with adversarial training being the most popular [7]. This is known to improve robustness to adversarial examples but also tends to lower accuracy on clean test data that has no perturbations [8], [9]. Many previously proposed defenses have also shown to be vulnerable [4], [10], [11] thus leaving adversarial robustness still an open problem in machine learning.

The 01 loss is known to be more robust to outliers than convex loss models [12]–[14] and gives different boundaries in the presence of outliers. Could the difference in boundaries translate into non-transferability of adversarial examples between convex and 01 loss models? We study this in the setting of white box and substitute model black box attacks.

Computationally 01 loss presents a considerable challenge because it is NP-hard to solve [15]. Previous attempts [13], [16]–[19] lack on-par test accuracy with convex solvers and are slow and impractical for large multiclass image benchmarks. However, a recent stochastic coordinate descent method for linear 01 loss models [12] has shown to attain comparable accuracies to state of the art linear solvers like the support vector machine. We build upon this in our current work.

We propose a 01 loss dual layer neural network with sign activation which to the best of our knowledge is the first such network to be proposed. We train our network with stochastic coordinate descent (along the lines of [12] and fully described in our Supplementary Material) and show that it achieves on-par test accuracy to equivalent convex models. Note that our model is different from binarized neural networks that are trained with gradient descent and an approximation to the sign activation [20]. We are now in a position to study transferability of adversarial examples between linear 01 loss and convex (hinge) loss models, and between dual layer neural networks with sign activation and 01 loss vs sigmoid activation and logistic loss.

We proceed with white box attacks where the attacker has full knowledge of the model's parameters and transfer based black box attacks via substitute models where the attacker has access only to label outputs of the model. We study transferability and black box attacks on image benchmarks MNIST [21], CIFAR10 [22], and Mini ImageNet (a ten class subset of the original ImageNet [23]) where we make the following findings.

- White box adversaries do not transfer effectively between convex and 01 loss on CIFAR10 and ImageNet datasets than they do on MNIST.
- As a result of this, convex substitute model black box attacks are more effective on convex models than 01 loss

ones on CIFAR10 and ImageNet.

- 01 loss substitute model black box attacks are ineffective on all datasets and on all models due to their non-uniqueness.
- Discretization of input features does not affect the non-transferability on CIFAR10, in fact it increases it and as a consequence makes 01 loss models even more robust to convex substitute model black box attacks than in the original feature space.
- Our 01 loss models can attain comparable black box robustness to convolutional models than their convex counterparts.

II. Methods

A. Background

The problem of determining the hyperplane with minimum number of misclassifications in a binary classification problem is known to be NP-hard [15]. In mainstream machine learning literature this is called minimizing the 01 loss [24] given in Objective 1,

$$\frac{1}{2n}\arg\min_{w,w_0}\sum_i(1-sign(y_i(w^Tx_i+w_0))) \quad (1)$$

where $w \in R^d$, $w_0 \in R$ is our hyperplane, and $x_i \in R^d, y_i \in \{+1,-1\}.\forall i = 0...n-1$ are our training data. Popular linear classifiers such as the linear support vector machine, perceptron, and logistic regression [25] can be considered as convex approximations to this problem that yield fast gradient descent solutions [14]. However, they are also more sensitive to outliers than the 01 loss [12]–[14] and more prone to mislabeled data than 01 loss [26]–[28].

B. A dual layer 01 loss neural network

We extend the 01 loss to a simple two layer neural network with k hidden nodes and sign activation that we call the MLP01 loss. This objective for binary classification can be given as

$$\frac{1}{2n}\arg\min_{W,W_0,w,w_0}\sum_i(1-sign(y_i(w^T(sign(W^Tx_i+W_0))+w_0))) \quad (2)$$

where $W \in R^{d\times k}$, $W_0 \in R^k$ are the hidden layer parameters, $w \in R^k, w_0 \in R$ are the final layer node parameters, $x_i \in R^d, y_i \in \{+1,-1\}.\forall i = 0...n-1$ are our training data, and $sign(v \in R^k) = (sign(v_0), sign(v_1), ..., sign(v_{k-1}))$. While this is a straightforward model to define optimizing it is a different story altogether. Optimizing even a single node is NP-hard which makes optimizing this network much harder.

C. Stochastic coordinate descent for 01 loss

We solve both problems with stochastic coordinate descent based upon earlier work [12]. We initialize all parameters to random values from the Normal distribution with mean 0 and variance 1. We then randomly select a subset of the training data (known as a batch) and perform the coordinate descent analog of a single step gradient update in stochastic gradient descent [29]. We first describe this for a linear 01 loss classifier which we obtain if we set the number of hidden nodes to zero. In this case the parameters to optimize are the final weight vector w and the threshold w_0.

When the gradient is known we step in its negative direction by a factor of the learning rate: $w = w - \eta\nabla(f)$ where f is the objective. In our case since the gradient does not exist we randomly select k features (set to 64, 128, and 256 for MNIST, CIFAR10, and ImageNet in our experiments), modify the corresponding entries in w by the learning rate (set to 0.17) one at a time, and accept the modification that gives the largest decrease in the objective. Key to our search is a heuristic to determine the optimal threshold each time we modify an entry of w. In this heuristic we perform a linear search on a subset of the projection w^Tx_i and select w_0 that minimizes the objective.

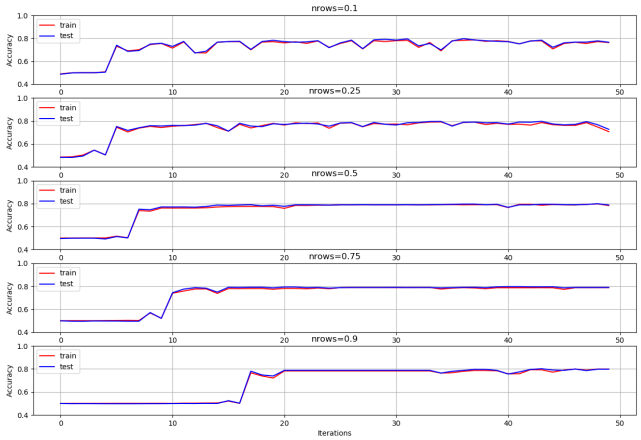


Fig. 1. Train and test accuracy of our stochastic coordinate descent on CIFAR10 class 0 vs 1 with different batch sizes (denoted as nrows).

We repeat the above update step on randomly selected batches for a specified number of iterations given by the user. In Figure 1 we show the effect of the batch size (as a percentage of each class to ensure fair sampling) on a linear 01 loss search on CIFAR10 between classes 0 and 1. We see that a batch size of 75% reaches a train accuracy of 80% faster than the other batch sizes. Thus we use this batch size in all our experiments going forward.

We also see that for this batch size the search flattens after 15 iterations (or epochs as given in the figure). We run 1000 iterations to ensure a deep search with an intent to maximize test accuracy. For imbalanced data (that appears in the one-vs-all design) we find that optimizing a balanced version of our objective for half the iterations followed by the default (imbalanced) version gives a lower objective in the end.

In a dual layer network we have to optimize our hidden nodes as well. In each of the 1000 iterations of our search we apply the same coordinate update described above, first to the final output node and then a randomly selected hidden node. In preliminary experiments we find this to be fast and almost as effective as optimizing all hidden nodes and the final node in each iteration.

Our intuition is that by searching on just the sampled data we avoid local minima and across several iterations we can explore a broad portion of the search space. Throughout iterations we keep track of the best parameters that minimize our objective on the full dataset.

The problem with our search described above is that it will return different solutions depending upon the initial starting point. To make it more stable we run it 32 times from different random seeds and use the majority vote for prediction.

We extend both our linear and non-linear models to a simple one-vs-all approach for multiclass classification. For a dataset with k classes we create 32 one-vs-all classifiers for each of the k classes. From the 32 models we can obtain frequency outputs for a test point using simple counting and use them as confidence scores for each class k. From this we output the predicted class as the one with the highest confidence. This is similar in spirit to the typical convex softmax objective used in convex neural networks except that there we can optimize to obtain the exact confidences given by sigmoid probabilities.

D. Implementation, experimental platform, and image data

We implement our 01 loss models in Python and Pytorch [30], and both MLP and SVM (LinearSVC class) in scikit-learn [31]. We optimize MLP with stochastic gradient descent that has a batch size of 200, momentum of 0.9, and learning rate of 0.01 (.001 for ImageNet data). We ran all experiments on Intel Xeon 6142 2.6GHz CPUs and NVIDIA Titan RTX GPU machines (for parallelizing multiple votes). Our SCD01 and MLP01 source codes, supplementary programs, data, and Supplementary Material file are available from https://github.com/zero-one-loss/mlp01.

We experiment on three popular image benchmarks: MNIST [21], CIFAR10 [22], and ImageNet [23]. Briefly MNIST is composed of grayscale handwritten digits each of size 28×28 with 60000 training images and 10000 test and CIFAR10 has 32×32 color images with 50000 training and 10000 test. ImageNet is a large benchmark with 1000 classes and color images of size 256×256. We extract images from10 random classes and split them to give a training set of 6144 images and test set of 6369. We normalize each image in each benchmark by dividing each pixel value by 255.

III. Results

We refer to our linear (no hidden layer) and non-linear (single hidden layer with 20 nodes) models as SCD01 and MLP01 respectively. As convex counterparts we select the linear support vector machine (with a cross-validated regularization parameter) denoted as SVM and a dual layer 20 hidden node neural network with logistic loss (MLP). For multiclass we use one-vs-all for all four methods. We use the majority vote of 32 runs for our 01 loss models to improve stability and do the same for SVM and MLP by majority voting on 32 bootstrapped samples. Our SCD01 and MLP01 source codes, supplementary programs, and data are available from https://github.com/zero-one-loss/mlp01.

We refer to the accuracy on the test data as clean data test accuracy. An incorrectly classified adversarial example is considered a successful attack whereas a correctly classified adversarial is a failed one. Thus when we refer to accuracy of adversarial examples it is the same as $100 - attacksuccessrate$. The lower the accuracy the more effective the attack.

A. Clean accuracy and runtimes

Before going into robustness we first compare the clean test data accuracies and training runtimes of our 01 loss models to their convex counterparts. In Table I we see that ensembling SVM and MLP models does not improve the test accuracy over single runs, thus we use a shared weight MLP network with 400 nodes on ImageNet to boost accuracy there. In fact the SVM boundary depends only upon the support vectors and so each ensemble will be the same as long as the support vectors are included. As a reminder we ensemble by taking the majority vote on multiple bootstrapped samples.

The 01 loss models improve considerably in all three datasets by ensembling. This is not too surprising since 01 loss is non-unique and will give different solutions when ran multiple times from different initializations. As a result of ensembling their accuracy is comparable to their convex peers. This makes it easier to compare their robustness since we don't have to worry about the robustness vs accuracy tradeoff [8], [9], [32].

TABLE I
ACCURACY OF 01 AND CONVEX COUNTERPARTS ON CLEAN TEST DATA. SINCE ENSEMBLING SVM AND MLP DOES NOT HELP WE USE A SINGLE RUN OF A MORE COMPLEX MODEL OF 400 HIDDEN NODES ON IMAGENET (DENOTED AS MLP400) WHICH PERFORMS MUCH BETTER.

	Single run			
	SVM	SCD01	MLP	MLP01
MNIST	91.7	83.7	97.6	91.2
CIFAR10	39.9	30.7	50.2	34.3
Mini ImageNet	26	25	32	25.5
	32 votes			
	SVM	SCD01	MLP	MLP01
MNIST	91.7	90.8	97.1	96
CIFAR10	40.2	39.7	47.4	46.4
	MLP400 single run	SCD01 32 votes		MLP01 32 votes
Mini ImageNet	36	34.7		41

In Table II we show the runtime of a single run of our 01 loss and convex models on class 0 vs all for each of the three datasets. We don't claim the most optimized implementation but our runtimes are still somewhat comparable to the convex loss models. Interestingly the convex models take much longer on complex and higher dimensional images in ImageNet compared to MNIST. Our 01 loss model runtimes are similar on MNIST and CIFAR10 because their sizes are similar. On Mini ImageNet since it has fewer training samples than MNIST and CIFAR10 the 01 loss runtimes are also lower.

B. White box attacks

In this section we study white box attacks just for binary classification on classes 0 and 1 in each of the three datasets.

TABLE II
RUNTIMES IN SECONDS OF SINGLE RUNS ON CLASS 0 VS ALL. MINI IMAGENET RUNTIMES FOR MLP ARE FOR 400 HIDDEN NODES.

	SVM	SCD01	MLP	MLP01
MNIST	0.8	171	64	875
CIFAR10	80	150	267	838
Mini ImageNet	659	83	8564	199

We use single runs of each of the four models to generate adversaries using the model parameters. We use the same white box attack method [6] for SVM and SCD01 since both are linear classifiers: for a given datapoint x and its label y the adversary is $x' = x + \epsilon(-y)sign(w)$ where $sign(w) = (sign(w_1), sign(w_2), ..., sign(w_d))$ and ϵ is the distortion.

For MLP we use the fast gradient sign method (FGSM) [1]. In this method we generate an adversary using the sign of the model gradient $x_{adv} = x + \epsilon sign(\nabla_x f(x, y)$ where f is the model objective and $\nabla_x f(x, y)$ is the model gradient with respect to the data x. We generate white box adversaries for MLP01 with a simple heuristic: for each hidden node w_k we modify the input as $x' = x + \epsilon(-y')sign(w_k)$ (where $y' = sign(w_k^T x)$ is the output of x from the hidden node w_k) and accept the first modification that misclassifies x in the final node output. If x is already misclassified or if none of the hidden node distortions misclassify it we distort with a randomly selected hidden node. We provide the full algorithm in the Supplementary Material. We use ϵ values on MNIST, CIFAR10, and ImageNet that are typical in the literature.

In Table III we see that the clean accuracies of our 01 loss models are comparable to the convex counterparts. As expected adversaries from the source on the same target are effective except for MLP01. More interestingly, while adversaries from SVM and MLP affect each other considerably they are far less pronounced on SCD01 and MLP01. We see this very clearly on CIFAR10 where both SVM and MLP adversaries have almost 0% accuracy when attacking each other indicating high transferability [6]. But SVM and MLP adversaries on SCD01 and MLP01 have a far less effect in this dataset. Adversaries from MLP attain a 63.7% accuracy on MLP01 and 43.5% on SCD01. Another interesting observation is that adversaries barely transfer between SCD01 and MLP01. We see similar behavior on Mini ImageNet and to a lesser degree on MNIST.

We argue that the difference of loss functions (01 vs convex) may be responsible for different boundaries and non-transferability. We illustrate this in two examples. First we see the effect of outliers on 01 loss and hinge loss linear classifiers. Recall that the hinge loss is $max(0, 1 - yw^T x)$ where y is the label and $w^T x$ is the prediction of x given by the classifier w. In Figure 2(a) the misclassified outlier forces the hinge loss to give a skewed linear boundary with two misclassifications. This happens because even though the two points are misclassified by the red boundary they are closer to it than the single misclassified one is to the blue one. The 01 loss is unaffected by distances and thus gives the blue

TABLE III
ACCURACY OF ADVERSARIES MADE BY SOURCE IN THE FIRST COLUMN TARGETING MODELS IN THE TOP ROW (CLASS 0 VS 1).

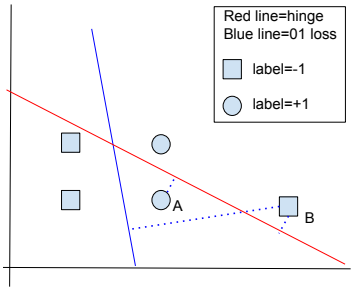
	SVM	SCD01	MLP	MLP01
MNIST $\epsilon = .3$				
Clean	100	99.9	100	100
SVM	11.9	8.1	40.4	43.5
SCD01	97	0	98.5	53.2
MLP	25.5	16.1	31	42.3
MLP01	99.9	99.8	99.6	69.5
CIFAR10 $\epsilon = .0625$				
Clean	82.2	81.1	88.7	84.2
SVM	0	41.3	0.5	70.1
SCD01	76	0.8	86	84.5
MLP	0	43.5	0.4	63.7
MLP01	81.7	80	88.5	66.9
Mini ImageNet $\epsilon = .0625$				
Clean	60.7	67.5	66.1	68.7
SVM	0	54.9	21.2	53.8
SCD01	58.6	1	65	60.3
MLP	0.5	42	21.6	52.3
MLP01	60.8	65.1	65.8	35.7

boundary with one misclassification. Since the two boundaries have different orientations their adversaries are also likely to be different. In a dataset like MNIST where our accuracies are high we don't expect many misclassified outliers and thus boundaries are unlikely to be different. As a result we see that many adversaries transfer between SVM and SCD01 on MNIST. But on CIFAR10 and Mini ImageNet, which are more complex and likely to contain misclassified outliers, we expect different boundaries which in turn gives fewer adversaries that transfer between the two.

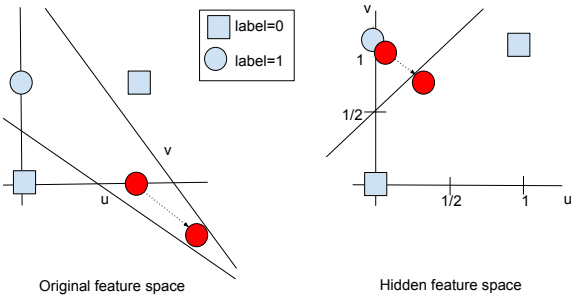
Next we see the difference of convex and 01 loss in simple two hidden node network. In Figure 2(b) we see two hyperplanes u and v on the left whose logistic outputs give the hidden feature space on the right. The two hyperplanes u and v represent two hidden nodes in a two layer network. Recall that the logistic activation $\frac{1}{1+e^{-w^T x}}$ (where $w^T x$ is prediction of x given by w) is similar to 01 loss: for large values of $|w^T x|$ it approaches 0 or 1 depending upon the sign of $w^T x$ and approaches $\frac{1}{2}$ as $|w^T x|$ approaches 0. Thus if we move the red circle towards the "corner" in the original feature space (as shown in Figure 2(b)) its outputs from u and v approach $\frac{1}{2}$ in the hidden space. Consequently it crosses the linear boundary in the hidden space and becomes adversarial. However if the activation is 01 loss the red point remains unmoved in the hidden space. In fact in 01 loss a datapoint's value in the hidden space changes only if we cross a boundary in the original space.

While both examples are not formal proofs they give some intuition of why fewer adversarial examples transfer between 01 loss and convex loss compared to between just convex. In particular we see that for 01 loss a datapoint becomes potentially adversarial if and only if it crosses a boundary in the original feature space whereas this is not true for convex losses.

Interestingly we also see MLP01 adversaries don't transfer to the other three models. When applied to MLP01 the



(a) Point B is misclassified by the blue line but its hinge loss $max(0, 1 - yw^T x)$ in dotted lines is higher than the combined loss of points A and B both misclassified by the red boundary. Thus hinge favors the red skewed line.



(b) The logistic activation $\frac{1}{1+e^{-w^T x}}$ in the original space gives a linear separation in the hidden space. If the red circle moves towards the "corner" of the boundaries its distance to u and v decreases. This in turn makes its activation values approach $1/2$ and it is misclassified in the hidden space. In 01 loss activation the red circle does not get affected in the hidden space.

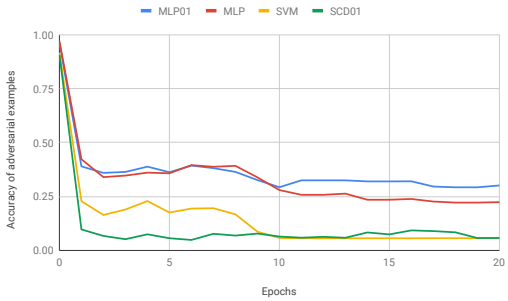
Fig. 2. Toy example showing different 01 loss and hinge boundaries, and adversarial examples in simple logistic loss network

adversaries lower its accuracy relative to clean data but to lesser degree than other models attacking themselves. Thus our white box attack method for MLP01 may not be the most powerful one leaving this an open problem.

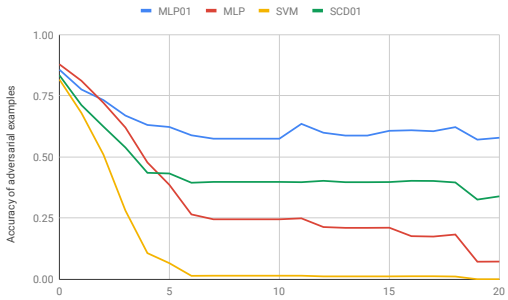
C. Substitute model black box attacks

We see that white box adversaries don't transfer between convex and 01 loss but can we attack a 01 loss model with a convex substitute model [6]? In this subsection we consider binary and multiclass classification on all three datasets. For all four methods we use 32 votes and one-vs-all multiclass classification. We use adversarial data augmentation [6] to iteratively train a substitute model trained on label outputs from the target model. In each epoch we generate white box adversaries targeting the substitute model with the FGSM method [1] and evaluate them on the target. Note that our black box attack is untargeted, we are mainly interested in misclassifying the data and not the misclassification label. See Supplementary Material for the full substitute model learning algorithms but it is essentially the method of Papernot et. al. [6].

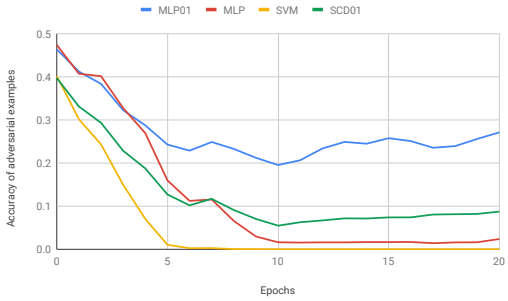
1) Convex substitute model: In Figure 3 we see the accuracy of target models on adversaries generated from a convex substitute model. Specifically we use a dual hidden layer



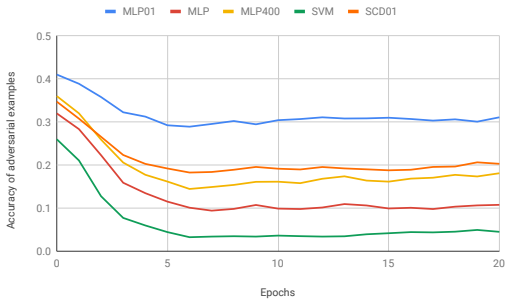
(a) MNIST $\epsilon = .2$



(b) CIFAR10 binary (class 0 and 1) $\epsilon = .0625$



(c) CIFAR10 $\epsilon = .0625$



(d) Mini ImageNet $\epsilon = 0.0625$

Fig. 3. Multiclass untargeted black box attack with a dual 200 node hidden layer logistic loss network as the substitute model. In epoch 0 are the clean test accuracies.

neural network with logistic loss and 200 nodes in each hidden layer as the substitute model. Like in the white box attacks we use ϵ values commonly used on these datasets. In MNIST (Figure 3(a)) we see a rapid drop in accuracy in the first few epochs and somewhat flat after epoch 10. We don't see a

considerable difference between the 01 loss and convex sibling models on MNIST although MLP01 has the highest accuracy.

On CIFAR10 and Mini ImageNet we see much more pronounced differences. In CIFAR10 binary classification (Figure 3(b)) we see that even though both MLP and MLP01 start off with clean test accuracies of 88% and 86% respectively, at the end of the 20th epoch MLP01 has 58% accuracy on adversarial examples while MLP has 7% accuracy. We see similar results on Mini ImageNet binary classification in the Supplementary Material. In CIFAR10 multiclass (Figure 3(c)) at the end of the 20th epoch the difference in accuracy between MLP and MLP01 is 24% even though both methods start off with about the same accuracy on clean test data. Similarly on Mini ImageNet MLP01 is 20.7% higher in accuracy than MLP in the 20th epoch. This is particularly interesting since MLP01 started off with a higher accuracy on Mini ImageNet and in general we expect more accurate models to be less robust [8], [9], [32]. However that is not the case here. Even if we give MLP the advantage of 400 hidden nodes in a shared weight network instead of one-vs-all, its accuracy in the 20th epoch is 13% lower than MLP01.

We have already seen earlier in white box attacks that adversaries transfer between SVM and SCD01 on MNIST but not so much on CIFAR10 and Mini ImageNet. The same phenomena can be used to explain the results we see here. On MNIST the convex substitute model can attack SCD01 and MLP01 as effectively as convex models due to better transferability on MNIST. Due to poor transferability on CIFAR10 and Mini ImageNet we see that the attack is less effective on SCD01 and MLP01. In the next subsection we explore what happens if the substitute model is SCD01.

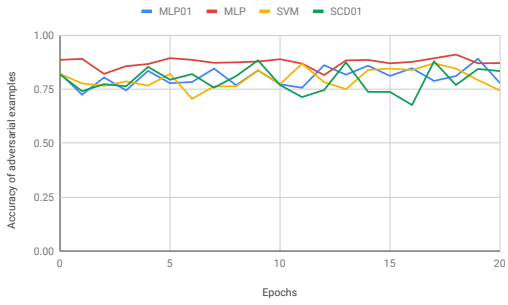
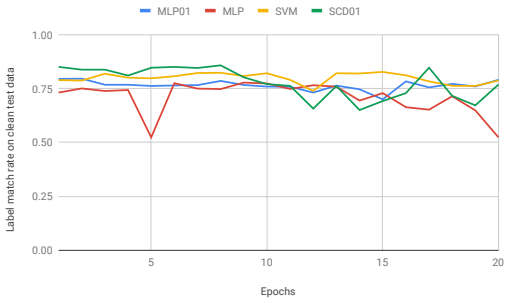


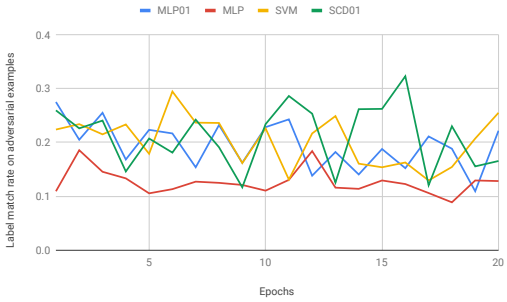
Fig. 4. We use SCD01 single run as the substitute model to attack single runs of the target models between only classes 0 and 1 in CIFAR10. In epoch 0 are the clean test accuracies.

2) 01 loss substitute model: In Figure 4 we see the results of a black box attack with SCD01 single run as the substitute model attacking single runs of target models. We see that adversaries produced from this model hardly affect any of the target models in any of the epochs. Even when the target is SCD01 and trained with the same initial seed as the substitute the adversaries are ineffective.

Further investigation reveals that the percentage of test data whose labels match between the 01 loss substitute and its target (known as the label match rate) is high but the label match rate on adversarial examples is much lower (shown in Figure 5).



(a) Percent of same labels between substitute and target on clean data



(b) Percent of same labels between substitute and target on adversarial data

Fig. 5. In (a) we see that SCD01 substitute model can approximate the target as shown in the label match rates between them. But SCD01 sourced adversaries have a lower match rate which indicates that the direction of the SCD01 boundary is very different from the targets and thus its adversaries have very little effect on the target.

Thus even though the SCD01 manages to approximate the target boundary its direction is different which gives ineffective adversaries. This is due to the non-uniqueness of 01 loss which makes single run solutions different from each other. Thus as a substitute model in black box attacks 01 loss is ineffective even in attacking itself.

D. Sensitivity to training data distribution and discretized features

The pixel distributions on MNIST shows that most pixels are near 0 and 1 whereas CIFAR10 pixel values are normally distributed [33]. Could this be the cause of transferability between 01 loss and convex on MNIST and non-transferability between them on CIFAR10? To test this we discretize the input features by saturation [33]: for each pixel x and fixed p we saturate it towards 1 or 0 using the formula $x^p = sign(2x-1)\frac{|2x-1|^{\frac{2}{p}}}{2} + \frac{1}{2}$. In Table IV we compare white box adversary transferability between our four models on the original CIFAR10 dataset between classes 0 vs 1 and a fully binarized one with saturation parameter $p = \infty$. Even after binarizing CIFAR10 adversarial examples do not transfer effectively between 01 loss and convex models, in fact the transferability becomes harder. Thus it is likely that different boundaries caused my outliers are the cause of non-transferability on CIFAR10.

TABLE IV
ACCURACY OF ADVERSARIES MADE BY SOURCE IN THE FIRST COLUMN TARGETING MODELS IN THE TOP ROW (CLASS 0 VS 1).

	SVM	SCD01	MLP	MLP01
CIFAR10 original $\epsilon = .0625$				
Clean	82.2	81.1	88.7	84.2
SVM	0	41.3	0.5	70.1
SCD01	76	0.8	86	84.5
MLP	0	43.5	0.4	63.7
MLP01	81.7	80	88.5	66.9
CIFAR10 fully binarized $\epsilon = .0625$				
Clean	76.6	79.1	81.2	78.6
SVM	0	73.9	6.5	76.7
SCD01	71.1	62.4	78.1	78.5
MLP	0.3	74.2	3	76.3
MLP01	76.7	79.4	81	73.8

In fact discretizing the input features increased robustness of convex and 01 loss models (as also seen previously for convex models [34]). Below in Figure 6 we see the accuracy of adversarial examples when we attack models trained on the original input features and binarized ones. The clean test accuracy is slightly lower in the binarized feature space but the robustness is better (particularly for our 01 loss models). In MNIST we see 100% robustness after binarizing input features.

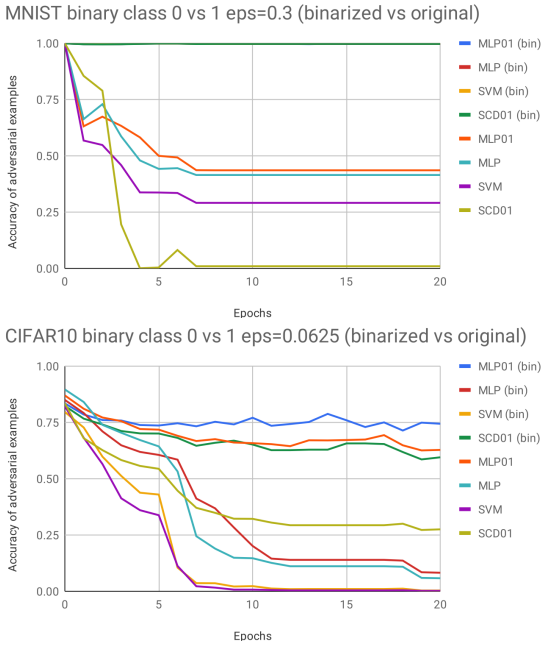


Fig. 6. Black box attacks with a dual 200 node hidden layer logistic loss network as the substitute model. In epoch 0 are the clean test accuracies. All models are more robust on the binarized features than the original space.

E. Comparison to convolutional neural networks

We finally compare black box robustness of our 01 loss models to simple convolutional neural networks. We find the pair of CIFAR10 classes where MLP01 achieves the highest test accuracy (this turns out to be classes 6 vs 8). We then train MLP01 and MLP each with 400 hidden nodes and two convolutional neural networks. We take LeNet [21] and SimpleNet80 that has the convolutional layers from LeNet followed by a single hidden layer of 80 nodes. The latter is a watered down version of LeNet with fewer hidden layers.

We use a convolutional neural network with four convolutional blocks as the substitute model. In each convolutional block we have a 3×3 convolutional kernel followed by max pool and batch normalization. In the first, second, third, and fourth layer we have 32, 64, 128, and 256 kernels and a final layer for the output. Thus the substitute model here is much more sophisticated and powerful than the one used in our experiments above.

In Figure 7 we see that MLP01 has lower but comparable clean test accuracies than all the convex models including its convex counterpart MLP. However, its robustness is much higher than MLP and even surpasses that of SimpleNet80 after the tenth epoch of the substitute model training.

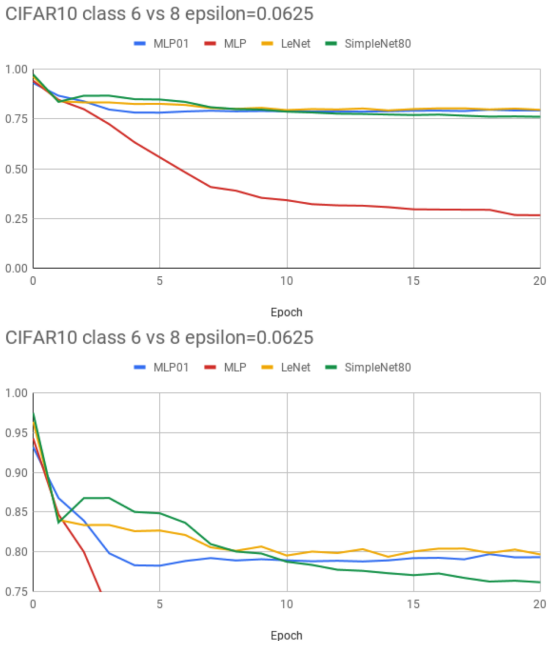


Fig. 7. Black box attacks with a convolutional neural network of four convolutional blocks. In epoch 0 are the clean test accuracies. We see that MLP01 is on-par with convolutional networks but MLP has much lower robustness.

IV. DISCUSSION AND CONCLUSION

Binarized neural networks have weights and activations constrained to +1 and -1 and are trained with gradient descent and an approximation to the sign activation [20]. The purpose of those networks is efficiency as opposed to robustness. Indeed we see in recent work that binarized networks offer marginal improvements in robustness to substitute model black box robustness on MNIST and none in CIFAR10 (see Tables 4 and 5 in [35] and Table 8 in [34]). In our case we perform a direct optimization without approximating the sign activation and as a result we see large improvements over convex (non-binarized) models on CIFAR10 as well as on ImageNet.

There is nothing to indicate that 01 loss models are robust to black box attacks that do not require substitute model training [5], [36]. These attacks are, however, computationally more expensive and require separate computations for each example. A transfer based model can be more effective (and dangerous) once it has approximated the target model boundary.

Our work shows a lack of transferability between 01 loss and convex models on datasets like CIFAR10 and ImageNet. As a result they are more robust to substitute model black box attacks than convex models on those datasets. Interestingly the robustness is on-par with simple convolutional models that have the powerful advantage of convolutions which our 01 loss models lack. As future work 01 loss convolutions may be a promising avenue for models with high clean test accuracy and high adversarial accuracy as well.

REFERENCES

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

[2] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 372–387. IEEE, 2016.

[3] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016.

[4] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. IEEE, 2017.

[5] Wieland Brendel, Jonas Rauber, and Matthias Bethge. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. *arXiv preprint arXiv:1712.04248*, 2017.

[6] Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*, 2016.

[7] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[8] Aditi Raghunathan, Sang Michael Xie, Fanny Yang, John C Duchi, and Percy Liang. Adversarial training can hurt generalization. *arXiv preprint arXiv:1906.06032*, 2019.

[9] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric P Xing, Laurent El Ghaoui, and Michael I Jordan. Theoretically principled trade-off between robustness and accuracy. *arXiv preprint arXiv:1901.08573*, 2019.

[10] 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.

[11] Amin Ghiasi, Ali Shafahi, and Tom Goldstein. Breaking certified defenses: Semantic adversarial examples with spoofed robustness certificates. *arXiv preprint arXiv:2003.08937*, 2020.

[12] Meiyan Xie, Yunzhe Xue, and Usman Roshan. Stochastic coordinate descent for 0/1 loss and its sensitivity to adversarial attacks. In *Proceedings of 18th IEEE International Conference on Machine Learning and Applications - ICMLA 2019*, page to appear, 2019.

[13] Tan Nguyen and Scott Sanner. Algorithms for direct 0–1 loss optimization in binary classification. In *Proceedings of The 30th International Conference on Machine Learning*, pages 1085–1093, 2013.

[14] Peter L. Bartlett, Michael I. Jordan, and Jon D. Mcauliffe. Large margin classifiers: Convex loss, low noise, and convergence rates. In S. Thrun, L.K. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*, pages 1173–1180. MIT Press, 2004.

[15] Shai Ben-David, Nadav Eiron, and Philip M Long. On the difficulty of approximately maximizing agreements. *Journal of Computer and System Sciences*, 66(3):496–514, 2003.

[16] Shaodan Zhai, Tian Xia, Ming Tan, and Shaojun Wang. Direct 0-1 loss minimization and margin maximization with boosting. In C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 872–880. Curran Associates, Inc., 2013.

[17] Yufang Tang, Xueming Li, Yan Xu, Shuchang Liu, and Shuxin Ouyang. A mixed integer programming approach to maximum margin 0–1 loss classification. In *2014 International Radar Conference*, pages 1–6. IEEE, 2014.

[18] Shai Shalev-Shwartz, Ohad Shamir, and Karthik Sridharan. Learning linear and kernel predictors with the 0-1 loss function, 2011.

[19] Ling Li and Hsuan-Tien Lin. Optimizing 0/1 loss for perceptrons by random coordinate descent. In *Neural Networks, 2007. IJCNN 2007. International Joint Conference on*, pages 749–754. IEEE, 2007.

[20] Matthieu Courbariaux, Itay Hubara, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1. *arXiv preprint arXiv:1602.02830*, 2016.

[21] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[22] Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009.

[23] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[24] Shalev-Shwartz Shai, Ohad Shamir, and Karthik Sridharan. Learning linear and kernel predictors with the 0-1 loss function. *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, 22(3), 2011.

[25] Ethem Alpaydin. *Machine Learning*. MIT Press, 2004.

[26] Naresh Manwani and PS Sastry. Noise tolerance under risk minimization. *IEEE transactions on cybernetics*, 43(3):1146–1151, 2013.

[27] Aritra Ghosh, Naresh Manwani, and PS Sastry. Making risk minimization tolerant to label noise. *Neurocomputing*, 160:93–107, 2015.

[28] Yueming Lyu and Ivor W Tsang. Curriculum loss: Robust learning and generalization against label corruption. *arXiv preprint arXiv:1905.10045*, 2019.

[29] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

[30] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dÁlché Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[31] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[32] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. *arXiv preprint arXiv:1805.12152*, 2018.

[33] Gavin Weiguang Ding, Kry Yik Chau Lui, Xiaomeng Jin, Luyu Wang, and Ruitong Huang. On the sensitivity of adversarial robustness to input data distributions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 13–16, 2019.

[34] Priyadarshini Panda, Indranil Chakraborty, and Kaushik Roy. Discretization based solutions for secure machine learning against adversarial attacks. *IEEE Access*, 7:70157–70168, 2019.

[35] Angus Galloway, Graham W Taylor, and Medhat Moussa. Attacking binarized neural networks. *arXiv preprint arXiv:1711.00449*, 2017.

[36] Jianbo Chen, Michael I Jordan, and Martin J Wainwright. Hopskipjumpattack: A query-efficient decision-based attack. *arXiv preprint arXiv:1904.02144*, 3, 2019.