PROJECT REPORT

ADVERSARIAL MACHINE LEARNING

IT566

Minimum Norm Attack on kNN and kNN Based Models

Submitted by:

Aditi Rajput (202117019)
Sonam Bharti (202117014)

Submitted to:

Prof. Manjunath V. Joshi

January 2, 2023

***Abstract*—We study the robustness against adversarial examples of kNN classifiers and classifiers that combine kNN with neural networks. The main difficulty lies in the fact that finding an optimal attack on kNN is intractable for typical datasets. In this project, we propose a gradient-based attack on kNN and kNN-based defense. Our attack outperforms earlier methods with only a minimal increase in the computation time. The attack also beats the state-of-the-art attack on kNN when k $>$ 1 using less than 1 % of its running time. We hope that this attack can be used as a new baseline for evaluating the robustness of kNN and its variants.**

I. Introduction

Adversarial examples are inputs specifically crafted to induce an erroneous behavior from machine learning models,usually by adding a small imperceptible perturbation.Recently, this has motivated research on classical kNN classifiers to study their robustness properties and to leverage kNN for providing interpretability and robustness to neural networks.Evaluating kNN-based models is particularly difficult since they are not differentiable: most existing attacks rely on gradient descent and thus can't be applied to kNN-based models.

Yang et al. propose a dedicated attack on kNN,[1] but it scales poorly with k,the number of data points, and/or the dimension. Sitawarin & Wagner propose a heuristic for solving this problem by coming up with a differentiable loss function for the adversary and then relying on gradient descent to adversarial examples.[2]

In this project, we use a gradient-based method of finding adversarial examples that improves on Sitawarin & Wagner. We find that their attack was not optimal and our attack represents the new state of the art for attacking kNN classifiers.

II. Dataset

We are using MNIST dataset for this project.The MNIST database contains 60,000 training images and 10,000 testing images of small square 28×28 pixel grayscale images of handwritten single digits between 0 and 9. Half of the training set and half of the test set were taken from NIST's training dataset, while the other half of the training set and the other half of the test set were taken from NIST's testing dataset.

III. Method

We study the robustness against adversarial examples of kNN classifiers and classifiers that combine kNN with neural networks.The main difficulty lies in the fact that finding an optimal attack on kNN is intractable for typical datasets. In this project, we propose a gradient-based attack on kNN and kNN-based defenses. We demonstrate that our attack outperforms the old methods on all of the models we will test, with only a minimal increase in the computation time.

A. Notation

We denote the perturbed version of x as $\hat{x} = x + \delta$ where $x \in R_d$ denote a target sample or a clean sample that the adversary uses, and $y \in 1, 2, \ldots, c$ its ground-truth label.
The training set used by kNN and to train the neural network is (X, Y) where $X = x_1, x_2, \ldots, x_n$ and $Y = y_1, y_2, \ldots, y_{\mathrm{n}}$. Here, we only consider kNN that uses Euclidean distance as the metric. kNN returns an ordering of the indices of the neighbors of an input x from the nearest to the k-th nearest one (i.e., $\pi_1(x), \ldots, \pi_k(x)$). The k nearest neighbors of x are then given by $x_{\pi_1(x)}, \dot{;} x_{\pi_k(x)}$, and it follows that

$$||x - x_{\pi_i(x)}||_2 < ||x - x_{\pi_j(x)}||_2 \iff i < j$$

B. Sitawarin and Wagner attack

Sitawarin & Wagner solve the following optimization problem:

$$\hat{\delta} = arg \min_{\delta} \sum_{i=1}^{m} w_i \cdot \sigma(\|\tilde{x}_i - (x + \delta)\|_2^2 - \eta^2) + c\|\delta\|_2^2$$

such that x + $\delta \in [0, 1]^d$

where σ(.) is a sigmoid function and η is a threshold initially set to l_2-distance between x and $x_{\pi_{\mathrm{k}}\ (\mathrm{x})}$ The original paper chooses the popular Adam optimizer.

C. Our Attack Description

Our algorithm solves the following optimization problem:

$$\hat{\delta} = arg \min_{\delta} \sum_{i=1}^{m} max\{w_i \cdot \sigma(\|\tilde{x}_i - (x + \delta)\|_2^2 - \eta^2) + \Delta, 0\} + c\|\delta\|_2^2 \quad (1)$$

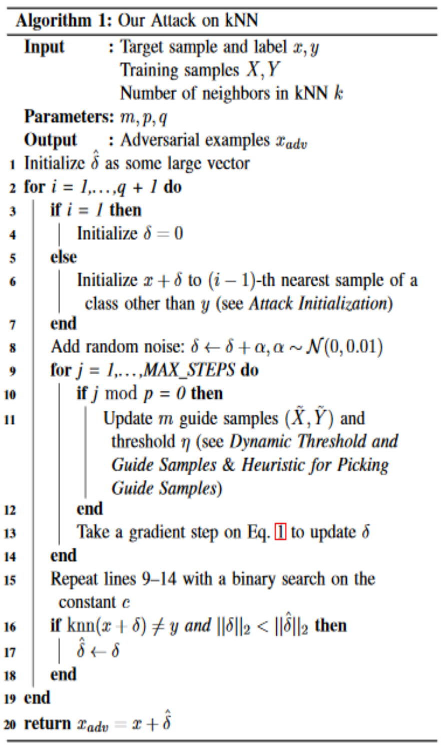
The changes compared to the original version are to use ReLU instead of sigmoid and the introduction of Δ. The algorithm we use is given in figure 1. The attack uses a subset of the training samples called guide samples, denoted ($\tilde{x}$ $_1$,$\tilde{y}$ $_1$),. . . , ($\tilde{x}_{\mathrm{m}}$,$\tilde{y}_{\mathrm{m}}$).The guide samples are given by:

$$arg \min_{(\hat{X},\hat{Y}) \subset (X,Y), |(\hat{X},\hat{Y}| = m, y_a dv \neq y} \sum_{\tilde{x} \in \tilde{X}} \|\tilde{x} - x\|_2$$

such that $\forall \tilde{y} \in \tilde{Y}, \tilde{y} = y_{adv}$

We fine-tune the hyperparameter m. The coefficients w_{i} are set to 1 if $\tilde{y}_{\mathrm{i}} \neq$ y, otherwise w_{i} = -1. For this heuristic, w_{i} = 1 for all i's.

Threshold function: The original idea of using the sigmoid is to simulate kNN's hard threshold with a soft differentiable one. The goal is to put the perturbed sample "just a bit further" from the training samples of the correct class than the threshold and "just a bit closer" to ones of a different class than the threshold, which is the



```
Algorithm 1: Our Attack on kNN
  Input      : Target sample and label $x, y$
               Training samples $X, Y$
               Number of neighbors in kNN $k$
  Parameters: $m, p, q$
  Output     : Adversarial examples $x_{adv}$
1  Initialize $\hat{\delta}$ as some large vector
2  for $i$ = 1,…,$q$ + 1 do
3      if $i$ = 1 then
4          Initialize $\delta = 0$
5      else
6          Initialize $x + \delta$ to $(i-1)$-th nearest sample of a
            class other than $y$ (see Attack Initialization)
7      end
8      Add random noise: $\delta \leftarrow \delta + \alpha, \alpha \sim \mathcal{N}(0, 0.01)$
9      for $j$ = 1,…,MAX_STEPS do
10         if $j$ mod $p$ = 0 then
11             Update $m$ guide samples $(\tilde{X}, \tilde{Y})$ and
                threshold $\eta$ (see Dynamic Threshold and
                Guide Samples & Heuristic for Picking
                Guide Samples)
12         end
13         Take a gradient step on Eq. 1 to update $\delta$
14     end
15     Repeat lines 9–14 with a binary search on the
        constant $c$
16     if $\mathrm{knn}(x + \delta) \neq y$ and $||\delta||_2 < ||\hat{\delta}||_2$ then
17         $\hat{\delta} \leftarrow \delta$
18     end
19 end
20 return $x_{adv} = x + \hat{\delta}$
```

Figure 1. Attack on kNN

distance between the input and its k-th nearest neighbor.But handling the Euclidean distance is challenging with this approach. ReLU avoids these problems.

Creating a small gap:Δ is added for numerical stability. We do not want $\tilde{x}$ to be at exactly the same distance from all guide samples

D. *Attack on KNN-Based Defences*

Our attack can be applied to models based on kNN with very little modification.

1) *Deep kNN:* The optimization objective in Eq. 1 can be rewritten as follows:

$$\sum_{l \in L} \sum_{i=1}^{m} max\{w_i \cdot \sigma(||\tilde{x}_i - (x+\delta)||_2^2 - \eta_l^2) + \Delta, 0\}$$

+ c $\|\delta\|_2^2$ (2)

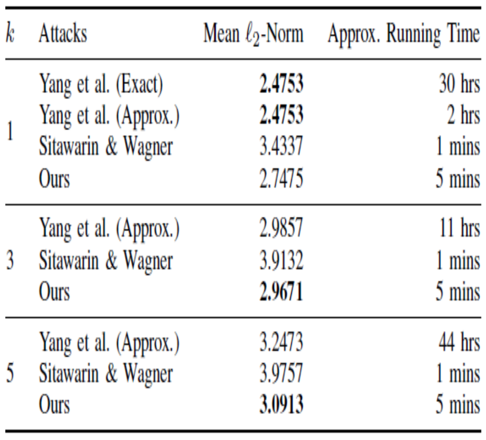
where $d_l(x_1, x_2) = \|f_1(x_1) - f_l(x_2)\|_2$

where $d_l(\ldots)$is Euclidean distance at layer l.

2) *Single-Layered Deep kNN:* The attack on deep kNN can be directly applied to single- layered deep kNN: here L only contains the penultimate layer of the neural network.

3) *Dubey et al.:* Dubey et al. do not use kNN directly on intermediate representations of the network. Instead, kNN is applied on a linearly transformed space. The distance metric must then be computed in this space:

$$d(x_1, x_2) = \|A(\phi(x_1) - \mu) - A(\phi(x_2) - \mu\|_2$$



k	Attacks	Mean ℓ_2-Norm	Approx. Running Time
1	Yang et al. (Exact)	**2.4753**	30 hrs
	Yang et al. (Approx.)	**2.4753**	2 hrs
	Sitawarin & Wagner	3.4337	1 mins
	Ours	2.7475	5 mins
3	Yang et al. (Approx.)	2.9857	11 hrs
	Sitawarin & Wagner	3.9132	1 mins
	Ours	**2.9671**	5 mins
5	Yang et al. (Approx.)	3.2473	44 hrs
	Sitawarin & Wagner	3.9757	1 mins
	Ours	**3.0913**	5 mins

Table I
COMPARISON OF ATTACKS ON KNN.

where ϕ(x) is a concatenation of an average pooling of f_l(x) for every layer l used by the model
A is a PCA transformation matrix that reduces dimension of the features,and
μ is the mean of ϕ(x) across the training set.

IV. RESULTS

A. Compare Attacks on kNN

The given Table I shows the comparison between the three attacks proposed over kNN model for different values of k (such that, $k = 1, 3, 5$). In this table, we can see that our attack gives larger mean perturbation value about 10% more than the previously proposed attack. But our attack uses only 5 minutes to run which is a great achievement in comparison.

Our attack outperforms their approximate attack for $k = 3, 5$, with smaller distortion and with significantly shorter computation time. Note that the running time reported above is a total over 100 samples.

B. Robustness Evaluation of kNN-Based Defenses

Our attack outperforms and it successfully generate adversarial examples for all of the 100 test samples against all models of Sitawarin Wagner. The details of the robustness evaluation of all the kNN-based models has been shared in Table II. In the table, “normal” means the model uses a normally trained network, and “AT” indicates that model uses an adversarially trained network. Using an adversarially trained network as a feature extractor for kNN is consistently more robust than using a normal network. However, this comes with a cost of clean accuracy.

Models	Clean Acc.	Attacks	Mean ℓ_2-Norm
Normal network	0.9878	CW	1.3970
Adversarial training [15]	0.9647	CW	2.8263
Ordinary kNN ($k = 5$)	0.9671	SW	3.3185 (0.97)
		Ours	**2.1869**
Deep kNN (Normal) [8]	0.9877	SW	1.9933
		Ours	**1.7539**
Deep kNN (AT)	0.9773	SW	2.6792
		Ours	**2.4632**
Single-layered Deep kNN (Normal) [10]	0.9910	SW	2.0964 (0.94)
		Ours	**1.5184**
Single-layered Deep kNN (AT)	0.9871	SW	2.9345 (0.97)
		Ours	**2.2992**
Dubey et al. (Normal) [11]	0.9838	SW	2.0057
		Ours	**1.2606**
Dubey et al. (AT)	0.9680	SW	2.6985
		Ours	**2.3979**

Table II
ACURACY AFTER ATTACK.

C. Output

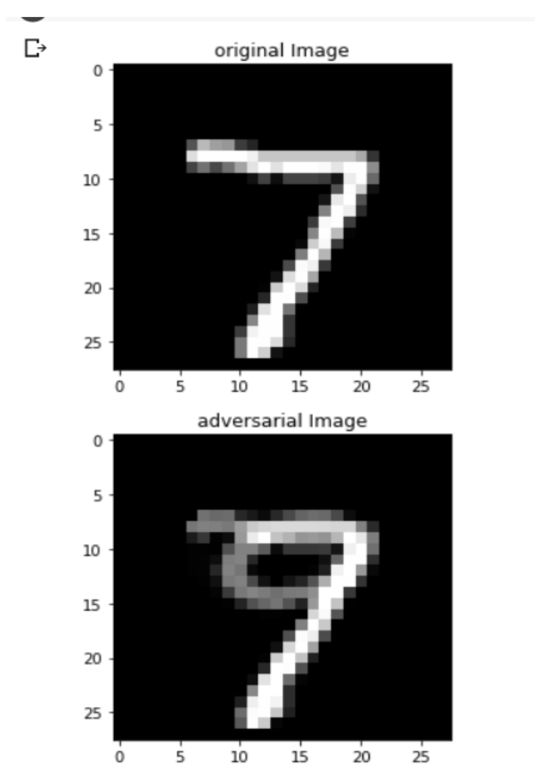


Figure 2. Output A

In this figure 2. Output A we have tested our model with index 0 and the test data-sets at zeroth index $(i = 0)$ was 7 which when goes under the attacked model it perturbed the data and changes it to some other digit. Here in the second part of the Output A image we can see the adversarial image which seems to look like the value 9 which was previously 7. Again, In figure 3. Output B again we tested the perturbed data with index $(i = 1)$ previously the original image which was read as value 2 but after the attack the image seems to be the value 8.

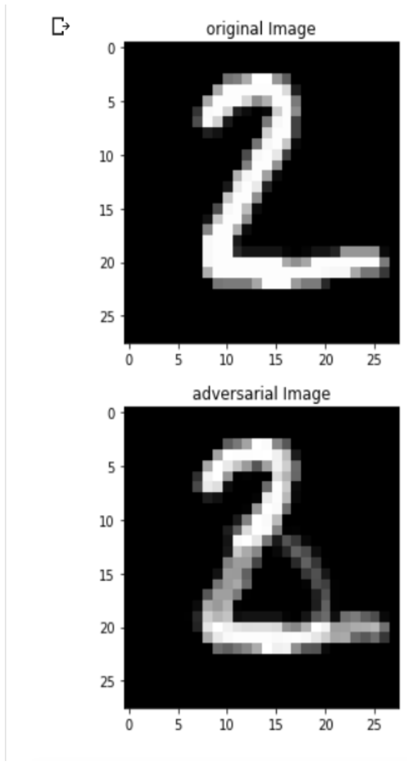


Figure 3. Output B

V. Conclusion

We show an attack on kNN and kNN-based classifiers and demonstrate that it improves on prior work. Our experiments suggest that our attack can be used as a flexible and efficient method for evaluating the robustness of kNN and kNN-based models on real-world datasets.

VI. Future Work

We also only consider l2 attacks as gradient-based approaches generally do not work well for finding minimum-l∞-norm adversarial examples.We only discuss robustness against an untargeted attack, but our method easily generalizes to targeted attacks.So we can improve our model to work effectively against targetted attacks.

References

[1] Y. W. Y. Yang, C. Rashtchian and K. Chaudhuri, "Adversarial examples for non-parametric methods: Attacks, defenses and large sample limits," *CoRR*, vol. abs/1906.03310, 2019. [Online]. Available: http://arxiv.org/abs/1906.03310

[2] C. Sitawarin and D. Wagner, "On the robustness of deep k-nearest neighbors," vol. abs/1903.08333, 2019. [Online]. Available: http://arxiv.org/abs/1903.08333