An Adversarial Attacker for Neural Networks in Regression Problems

Kavya Gupta, Beatrice Pesquet-Popescu, Fateh Kaakai, Jean-Christophe Pesquet, Fragkiskos D. Malliaros

To cite this version:
Kavya Gupta, Beatrice Pesquet-Popescu, Fateh Kaakai, Jean-Christophe Pesquet, Fragkiskos D. Malliaros. An Adversarial Attacker for Neural Networks in Regression Problems. IJCAI Workshop on Artificial Intelligence Safety (AI Safety), Aug 2021, Montreal/Virtual, Canada. hal-03527640v2

HAL Id: hal-03527640
https://centralesupelec.hal.science/hal-03527640v2
Submitted on 22 Oct 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
An Adversarial Attacker for Neural Networks in Regression Problems

Kavya Gupta¹, Beatrice Pesquet-Popescu², Fateh Kaakai², Jean-Christophe Pesquet¹ and Fragkiskos D. Malliaros³

¹Université Paris-Saclay, CentraleSupélec, Inria Centre de Vision Numérique, Gif-sur-Yvette, France
²Air Mobility Solutions BL, Thales LAS France

kavya.gupta100@gmail.com, beatrice.pesquet@thalesgroup.com, fateh.kaakai.e@thalesdigital.io, {jean-christophe.pesquet, fragkiskos.malliaros}@centralesupelec.fr

Abstract

Adversarial attacks against neural networks and their defenses have been mostly investigated in classification scenarios. However, adversarial attacks in a regression setting remain understudied, although they play a critical role in a large portion of safety-critical applications. In this work, we present an adversarial attacker for regression tasks, derived from the algebraic properties of the Jacobian of the network. We show that our attacker successfully fools the neural network, and we measure its effectiveness in reducing the estimation performance. We present a white-box adversarial attacker to support engineers in designing safety-critical regression machine learning models. We present our results on various open-source and real industrial tabular datasets. In particular, the proposed adversarial attacker outperforms attackers based on random perturbations of the inputs. Our analysis relies on the quantification of the fooling error as well as various error metrics. A noteworthy feature of our attacker is that it allows us to optimally attack a subset of inputs, which may be helpful to analyse the sensitivity of some specific inputs.

1 Introduction

Adversarial machine learning has received an increased attention in the past decade. For all machine learning models, defense against adversarial attacks is important in terms of safety. Adversarial attacks in classification constitute malicious attempts to trick a model classifier. They play a critical role in real-world application domains such as spam/malware detection, autonomous systems [Huang and Wang, 2018], [Eykholt et al., 2018], [Ren et al., 2019], medical systems [Finlayson et al., 2018] etc. Adversarial attacks cause vulnerability in model deployment and specially needs to be taken into account in deployment of security-critical AI applications. Despite the newfound interest of the research community in trustworthy and explainable AI, there are only few works investigating the adversaries in the case of regression tasks.

Current advances in the adversarial machine learning field evolve around the issue of designing attacks and defenses with focus on the use of neural networks in image analysis and computer vision [Goodfellow et al., 2014], [Kurakin et al., 2016]. Much less works concern tabular data. However, most machine learning tasks in the industry rely on tabular data, e.g., fraud detection, product failure prediction, anti-money laundering, recommendation systems, click-through rate prediction, or flight arrival time prediction.

In this paper, we focus on generating adversarial attacks for neural networks in the specific scenario when i) a regression task is performed and ii) tabular data are employed. Our contributions are the following:

- We propose a simple, novel and flexible method for generating adversarial attacks for regression tasks (a white box attack).
- We show that the proposed attacker allows us to optimally attack on any given subset of input features.
- We explore various error metrics which are useful for analysing these adversarial attacks.
- Our proposed adversarial attacker is generalised for any $\ell_p$ norm on input and output perturbations.

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Speed</td>
</tr>
<tr>
<td>1</td>
<td>Flight Distance</td>
</tr>
<tr>
<td>2</td>
<td>Departure Delay</td>
</tr>
<tr>
<td>3</td>
<td>Initial ETE</td>
</tr>
<tr>
<td>4</td>
<td>Latitude Origin</td>
</tr>
<tr>
<td>5</td>
<td>Longitude Origin</td>
</tr>
<tr>
<td>6</td>
<td>Altitude Origin</td>
</tr>
<tr>
<td>7</td>
<td>Latitude Destination</td>
</tr>
<tr>
<td>8</td>
<td>Longitude Destination</td>
</tr>
<tr>
<td>9</td>
<td>Altitude Destination</td>
</tr>
<tr>
<td>10</td>
<td>Arrival Time Slot</td>
</tr>
<tr>
<td>11</td>
<td>Departure Time Slot</td>
</tr>
<tr>
<td>12</td>
<td>Aircraft Category</td>
</tr>
<tr>
<td>13</td>
<td>Airline Company</td>
</tr>
<tr>
<td>14</td>
<td>Refinement ETE</td>
</tr>
</tbody>
</table>

Table 1: Input and output variables description for Industrial dataset – A safety critical application.
• We evaluate our results on open-source regression datasets and an industrial dataset (output and input features described in the Table 1) which lies in the domain of safety critical applications.

In Section 2, we give a brief overview of existing works. In Section 3, we formulate the problem and present our method for generating adversarial examples in regression tasks. In Section 4, we perform numerical experiments on four datasets to demonstrate the effectiveness of the proposed attacker. Some concluding remarks are given in Section 5.

2 Related Work

In [Szegedy et al., 2013] the concept of adversarial attacks was first proposed to fool DNNs. Adding a subtle perturbation to the input of the neural network produces an incorrect output, while human eyes cannot recognize the difference in the modification of the input data. Even though different models have different architectures and might use different training data, the same kind of adversarial attack strategies can be used to attack related models. These attacks pose a huge threat to the performance of DNNs. [Szegedy et al., 2013] paper proposed L-BFGS to construct adversarial attacks and since then there has been plethora of works introducing various adversarial attacks and their defenses for DNNs.

[Goodfellow et al., 2014] proposed a simpler and faster method to construct adversarial attacks (FGSM). The generated images are misclassified by adding perturbations and linearizing the cost function in the gradient direction. This is a non-iterative attack, hence it has a lower computation cost than the previous method. The Fast Gradient Sign Method (FGSM) is an $\ell_\infty$ bounded attack and is often prone to label leaking.

It may be difficult for FGSM to control the perturbation level in constructing attacks. [Kurakin et al., 2016] proposed an optimized FGSM, termed Iterative Gradient Sign Method (IGSM), which adds perturbations in multiple smaller steps and clips the results after each iteration ensuring that the perturbations are restricted to the neighborhood of the example. [Dong et al., 2018] added momentum to IGSM attacks. [Papernot et al., 2016] proposed the Jacobian-based Saliency Map Attack (JSMA), which is based on the $\ell_0$ sparsity measure. The basic idea is to construct a saliency map with the gradients and model the gradients based on the impact of each pixel of the image.

[Moosavi-Dezfooli et al., 2016] proposed a non-targeted attack method based on the $\ell_2$-norm, called DeepFool. It tries to find the decision boundary that is the closest to the sample in the image space, and then use the classification boundary to fool the classifier. FGSM, JSMA, and DeepFool are designed to generate adversarial attacks corresponding to single image to fool the trained classifier model. [Moosavi-Dezfooli et al., 2017] proposed a universal image-agnostic perturbation attack method which fools classifier by adding a single perturbation to all images in the dataset. [Carlini and Wagner, 2017] proposed a powerful attack based on L-BFGS. The attack can be generated according to $\ell_1$, $\ell_2$, and $\ell_\infty$ norm which can be targeted or non-targeted. [Liu et al., 2016] proposed an ensemble attack method combining multiple models to construct adversarial attacks. [Rony et al., 2020] proposed a method to generate minimally perturbed adversarial examples based on Augmented Lagrangian for various distance metrics. In [Balda et al., 2018], authors propose a general framework for generation of adversarial examples in both classification and regression tasks for applications in image domain. Similar to our proposed approach, the technique is based on the Jacobian of the neural network. Most of the methods in the literature about adversarial example generation belong to the class of white box attackers, i.e., the attacker has access to the information related to the trained neural network model including the model architecture and its parameters. A black box attacker is introduced in [Su et al., 2019]. Such attackers do not know the model but can interact with it. A byproduct of black-box attack is grey-box attack where attackers might have limited information regarding the model. To the best of our knowledge the only work dealing with adversarial attacks in white box settings for tabular data has been proposed in [Ballet et al., 2019] and this work handles only classification tasks.

In regression tasks there are no natural margins as in the case of classification tasks, and adversarial learning in regression setting is hindered with difficulties to define the adversarial attacks, its success, and evaluation metrics. Despite the number of works in adversarial attack generation, there are few articles dealing with regression tasks. [Tong et al., 2018] looked at adversarial attacks in the setting of an ensemble of multiple learners, investigating the interactions between these linear learners and an attacker in regression setting, modeled as a Multi-Learner Stackelberg Game (MLSG). However, the investigated linear case is not able to capture the larger class of non-linear models. The focus only on specific applications of regression is a common. [Ghafari et al., 2018] examined an important problem: selecting an optimal threshold for each sensor against an adversary for regression tasks in cyber-physical systems. [Deng et al., 2020] introduced the concept of adversarial threshold which is related to a deviation between the original prediction and the prediction of adversarial example, i.e., an acceptable error range in driving models. In a regression context, [Nguyen and Raff, 2018] introduced a defense that is generically useful to reduce the effectiveness of adversarial attacks. They consider adversarial attacks as a potential symptom of numerical instability in the learned function. In the next section, we propose a general white-box adversarial attacker based on Jacobian of the learned function for regression tasks in tabular data domain.

3 Proposed Method

3.1 Objective

The problem of adversarial attacks is closely related to the robustness issue for a neural network, i.e. its sensitivity to perturbations. Let $T : \mathbb{R}^{N_0} \rightarrow \mathbb{R}^{N_m}$ be the considered neural network having $N_0$ scalar inputs and $N_m$ scalar outputs. If $x \in \mathbb{R}^{N_0}$ is a given vector of inputs for some data for which $y$ is the associated target output, the network has been trained to produce an output $\hat{T}(x)$ close to $y$. If the input is now perturbed by an additive vector $e \in \mathbb{R}^{N_0}$, the perturbed output
is \( T(x + e) \). Attacking the network then amounts to finding a perturbation \( e \) of preset magnitude which makes the output of the network to maximally deviate from a reference output. This reference output may be the model output \( T(x) \) or the ground truth output \( y \). Since our purpose is to develop an approach which remains efficient even if the accuracy of the network is not very high, we choose \( y \) as the reference output when available. In this context, the measures of deviation and of magnitude of the perturbation play an important role in terms of mathematical formulation of the problem. As a standard choice, the measure of perturbation magnitude will be here an \( \ell_p \)-norm where \( p \in [1, +\infty] \). For measuring the output deviation, we will similarly consider an \( \ell_q \)-norm where \( q \in [1, +\infty] \). It must be emphasized that this choice makes sense when dealing with regression problems. In this context, the \( \ell_2 \) or the \( \ell_1 \) norms are indeed frequently used as loss functions for training. On the other hand, the \( \ell_{\infty} \)-norm is also a popular measure when dealing with reliability issues.

### 3.2 Optimization formulation

In the described setting, the design of the attacker can be formulated as the problem of finding the “worst perturbation” \( \hat{e} \) such that

\[
\hat{e} \in \text{Argmax}_{e \in C_{p,\delta}} \| T(x + e) - y \|_q,
\]

where \( C_{p,\delta} \) is the closed and convex set defined as

\[
C_{p,\delta} = \{ e \in \mathbb{R}^{N_0} \mid \| \Sigma^{-1/2} e \|_p \leq \delta \}.
\]

\( \Sigma \in \mathbb{R}^{N_0 \times N_0} \) is symmetric positive definite matrix. \( \delta \) is a parameter which controls the maximum allowed perturbation and \( \Sigma \) is a weighting matrix typically corresponding to the covariance matrix of the inputs. For instance, if we assume that it is a diagonal matrix, it simply introduces a normalization of the perturbation components with respect to the standard deviations of the associated inputs.

For standard choices of activation functions, \( T \) is a continuous function. By virtue of Weierstrass theorem, the existence of a solution (not necessarily unique) to Problem (1) is then ensured. Although \( C_{p,\delta} \) is a relatively simple convex set, this problem appears as a difficult non-convex problem due to the fact that \( i) \) \( T \) is a complex nonlinear operator, \( ii) \) we maximize an \( \ell_q \) measure which, in addition, leads to a nonsmooth cost function when \( q = 1 \) or \( q = +\infty \). A further difficulty is that we usually need to attack a large dataset to evaluate the robustness of a network and the provided optimization algorithm should therefore be fast.

### 3.3 Algorithm

We propose to implement a two-step approach.

- **Step 1.** We first perform a linearization based on the following first-order Taylor expansion:

\[
T(x + e) \simeq T(x) + J(x)e,
\]

where \( J(x) \in \mathbb{R}^{N_m \times N_0} \) is the Jacobian of the network at \( x \). Note that \( J(x) \) can be computed by classical backpropagation techniques. We will make a second approximation, that is \( y \simeq T(x) \). Based on these two approximations and after the variable change \( e' = \delta^{-1} \Sigma^{-1/2} e \), Problem (1) simplifies to

\[
\text{maximize } e' \in B_p \| J(x) \Sigma^{1/2} e' \|_q,
\]

where \( B_p \) is the closed \( \ell_p \) ball centered at 0 and with unit radius. Note that the optimal cost value in (4) is the subordinate norm of matrix \( J(x) \Sigma^{1/2} \) when the input space is equipped with the \( \ell_p \) norm and the output space with the \( \ell_q \) one. We recall that this subordinate norm is defined, for every matrix \( M \in \mathbb{R}^{N_m \times N_0} \), as

\[
\| M \|_{p,q} = \sup_{z \in \mathbb{R}^{N_0} \setminus \{0\}} \| Mz \|_q / \| z \|_p.
\]

Problem (4) is thus equivalent to find a vector \( \hat{e}' \) for which the value of the cost function is equal to \( \| J(x) \Sigma^{1/2} \|_{p,q} \). For values of \((p, q)\) listed below the expression of such vector has an explicit form.

- If \( p = q = 2 \), \( \hat{e}' \) is any unit \( \ell_2 \) norm eigenvector of \( \Sigma^{1/2} J(x)^T J(x) \Sigma^{1/2} \) associated with the maximum eigenvalue of this matrix. This vector can be computed by performing a singular value decomposition of \( J(x) \Sigma^{1/2} \).

- If \( p = 2 \) and \( q = +\infty \), \( \hat{e}' \) is any unit \( \ell_2 \) norm vector colinear with a row of \( J(x) \Sigma^{1/2} \) having maximum \( \ell_2 \) norm.

- If \( p = +\infty \) and \( q = +\infty \), \( \hat{e}' \) is a unit norm vector whose elements are equal to \((e_i)_{1 \leq i \leq N_0} \), where, for every \( i \in \{1, \ldots, N_0\} \), \( e_i \in \{-1, 0, 1\} \) is the sign of the \( i \)-th element of a row of \( J(x) \Sigma^{1/2} \) with maximum \( \ell_1 \) norm.

- If \( p = 1 \) and \( q = 1 \), \( \hat{e}' \) is a vector which has only one nonzero component equal to \( \pm 1 \), the index of this component corresponds to the column of \( J(x) \Sigma^{1/2} \) with maximum \( \ell_1 \) norm.

- If \( p = 1 \) and \( q = 2 \), \( \hat{e}' \) is a vector with only one nonzero component equal to \( \pm 1 \). The index of this component corresponds to a column of \( J(x) \Sigma^{1/2} \) with maximum \( \ell_2 \) norm.

- If \( p = 1 \) and \( q = +\infty \), \( \hat{e}' \) is again a vector with only one nonzero component equal to \( \pm 1 \). The index of this component corresponds to a column of \( J(x) \Sigma^{1/2} \) where is located an element of maximum absolute value.

- **Step 2.** In the previous optimization step, the optimal solution is not unique. Indeed if \( \hat{e} = \delta \Sigma^{1/2} \hat{e}' \) is a solution to Problem (4), then \( -\hat{e} \) is also a solution. In addition, there may exist other reasons for the multiplicity of the solutions. For example, there may be several maximum norm rows for matrix \( J(x) \Sigma^{1/2} \). Among all the possible choices, we propose to choose the solution \( \hat{e} \) leading to the maximum deviation w.r.t. the ground truth, that is such that \( \| T(x + \hat{e}) - y \|_q \) is maximum. This requires

---

1We assume that \( J(x) \) is defined at \( x \), see [Bolte and Pauwels, 2020] for a justification of this assumption in the nonsmooth case.
to perform a search on a small number of possible candidates. Note that no approximation error is involved in this step. If the ground truth for the output is not available, it can be replaced by the model output.

- **Post-optimization.** If \( 1 < q < +\infty \) and \( T \) is assumed to be differentiable, \( e \mapsto ||T(x + e) - y||^q_q \) is a differentiable function. A further refinement consists of minimizing this function over \( C_{p,\delta} \) by using a projected gradient algorithm with Armijo search for the stepsize. The previous estimates of \( \hat{e} \) can then be used to initialize the algorithm. According to our numerical tests, implementing this strategy when \( q = 2 \) only brings a marginal improvement. Moreover, this approach cannot be used when \( q = 1 \) or \( q = +\infty \).

3.4 Attacking a group of inputs

It can also be interesting to attack only a selected subset of inputs. It may help in identifying the more sensitive inputs of the network. Also, for some inputs like unsorted categorical ones, attacks are often meaningless since they introduce a main change in the informative contents of the dataset, which can be easily detected. Our proposed approach can be adapted to generate such partial attacks. In Problem (4), it is indeed sufficient to replace matrix \( \Sigma^{1/2} \) by \( D\Sigma^{1/2}D \), where \( D \) a masking diagonal matrix whose diagonal elements are equal to 1 when the input is attacked and 0 otherwise. The optimal solutions \( \hat{e} \) and \( \hat{e} = \delta D\Sigma^{1/2}D\hat{e} \) have then their components equal to 0 for the non-attacked inputs. Note that the naive approach which would consist in solving (4) and setting to zero the resulting perturbation components for non-attacked inputs would be suboptimal.

4 Numerical Results

4.1 Dataset and architecture description

**Open Source Datasets**

We run our experiments on three open source regression datasets. The Combined Cycle Power Plant [Tufekci, 2014] dataset has 4 features with 9,568 instances. The task is to predict the net hourly electrical energy output using hourly average ambient variables. The Red Wine Quality dataset [Cortez et al., 2009] contains 1,599 total samples and each instance has 11 features. The features are physicochemical and sensory measurements for wine. The output variable is a quality score ranging from 0 to 10, where 10 represents for best quality and 0 for least quality. For the Abalone dataset, the task is to model an Abalone’s age based purely on its physical measurements. This would allow Abalone’s age estimation without cutting its shell. There are in total 4,177 instances with 8 input variables including one categorical variable. The datasets are divided with a ratio of 4:1 between training and testing data. The categorical attributes are dealt with by using one hot encoding based on the number of categories. The input attributes are normalised by removing their mean and scaling to unit variance.

We train fully connected networks for the estimation of variables from the datasets. The network architecture for the dataset are given below. The values represent the number of hidden neurons in the layers. Activation function at each layer is ReLU except for the last layer.

- **Combined cycle Power Plant dataset** - (10, 6, 1)
- **Red Wine Quality dataset** - (100, 100, 100, 10, 1)
- **Abalone Data set** - (256, 256, 256, 256, 1)

**Industrial Dataset – safety critical Application**

An industrial application dataset is also considered with 2,219,097 training, 739,639 validation, and 739,891 test samples. The description of the input/output variables of the
### Table 2: Error metrics used for evaluation. The mean value computed for SMAPE is limited to the $K_+$ positive values of the elements in the summation. $e_k$ is the error generated by the adversarial on the $k$-th sample in the dataset of length $K$.

<table>
<thead>
<tr>
<th>Noise</th>
<th>$\ell_2$</th>
<th>$\ell_4$</th>
<th>$\ell_{in}$</th>
<th>$\mathrm{SMAPE}_{\ell_2}$</th>
<th>$\mathrm{SMAPE}_{\ell_4}$</th>
<th>$\mathrm{SMAPE}<em>{\ell</em>{in}}$</th>
<th>$\mathrm{SMAPE}_{\mathrm{adv}}$</th>
<th>$\mathrm{SMAPE}_{\mathrm{spec}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Combined Cycle Power Plant Dataset</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$1 \times 10^{-4}$</td>
<td>$4.0 \times 10^{-3}$</td>
<td>$4.0 \times 10^{-3}$</td>
<td>$4.0 \times 10^{-3}$</td>
<td>$3.7 \times 10^{-3}$</td>
<td>$2.9 \times 10^{-3}$</td>
<td>$2.6 \times 10^{-3}$</td>
<td>$3.9 \times 10^{-3}$</td>
<td>$3.1 \times 10^{-3}$</td>
</tr>
<tr>
<td>$2 \times 10^{-4}$</td>
<td>$8.0 \times 10^{-5}$</td>
<td>$8.0 \times 10^{-5}$</td>
<td>$8.0 \times 10^{-5}$</td>
<td>$7.7 \times 10^{-5}$</td>
<td>$7.6 \times 10^{-5}$</td>
<td>$7.4 \times 10^{-5}$</td>
<td>$7.3 \times 10^{-5}$</td>
<td>$7.2 \times 10^{-5}$</td>
</tr>
<tr>
<td>Red Wine Quality Dataset</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$1 \times 10^{-3}$</td>
<td>$0.12$</td>
<td>$0.12$</td>
<td>$0.12$</td>
<td>$0.12$</td>
<td>$0.12$</td>
<td>$0.12$</td>
<td>$0.12$</td>
<td>$0.12$</td>
</tr>
<tr>
<td>$2 \times 10^{-3}$</td>
<td>$2.0$</td>
<td>$2.0$</td>
<td>$2.0$</td>
<td>$2.0$</td>
<td>$2.0$</td>
<td>$2.0$</td>
<td>$2.0$</td>
<td>$2.0$</td>
</tr>
<tr>
<td>Abalone age dataset</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$5 \times 10^{-4}$</td>
<td>$0.36$</td>
<td>$0.36$</td>
<td>$0.36$</td>
<td>$0.36$</td>
<td>$0.36$</td>
<td>$0.36$</td>
<td>$0.36$</td>
<td>$0.36$</td>
</tr>
<tr>
<td>$1 \times 10^{-3}$</td>
<td>$0.72$</td>
<td>$0.72$</td>
<td>$0.72$</td>
<td>$0.72$</td>
<td>$0.72$</td>
<td>$0.72$</td>
<td>$0.72$</td>
<td>$0.72$</td>
</tr>
<tr>
<td>Combined Cycle Power Plant Dataset</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$1 \times 10^{-4}$</td>
<td>$11.8 \times 10^{-3}$</td>
<td>$11.8 \times 10^{-3}$</td>
<td>$11.8 \times 10^{-3}$</td>
<td>$11.8 \times 10^{-3}$</td>
<td>$11.8 \times 10^{-3}$</td>
<td>$11.8 \times 10^{-3}$</td>
<td>$11.8 \times 10^{-3}$</td>
<td>$11.8 \times 10^{-3}$</td>
</tr>
<tr>
<td>$2 \times 10^{-4}$</td>
<td>$24.0 \times 10^{-3}$</td>
<td>$24.0 \times 10^{-3}$</td>
<td>$24.0 \times 10^{-3}$</td>
<td>$24.0 \times 10^{-3}$</td>
<td>$24.0 \times 10^{-3}$</td>
<td>$24.0 \times 10^{-3}$</td>
<td>$24.0 \times 10^{-3}$</td>
<td>$24.0 \times 10^{-3}$</td>
</tr>
</tbody>
</table>

### Table 3: Comparison on evaluation metrics random attacker vs. proposed adversarial attacker with variation in perturbation level ($\ell_2$ attack).

### Table 4: Results of proposed adversarial techniques. Standard training vs Spectral Normalisation training on $\ell_2$ attacks.

### Table 5: Results of proposed adversarial techniques. Standard training attacking all inputs vs standard training attacking few inputs on $\ell_2$ attacks.

The dataset is given in Table 1. The variable to be predicted is the Estimation of Arrival time (ETE) of a flight, given variables including the distance and speed, and also an initial estimate of ETE. The dataset is related to flight control, an activity area where safety is critical. The input attributes are normalized by removing their mean and scaling to unit variance. For models, we build fully connected networks with ReLU activation function on all the hidden layers except the last one. The network architecture is shown in Figure 1.

### 4.2 Experimental setup

We first train our networks without any constraints using the network architecture presented in the previous section with the aim of reducing the prediction/performance loss on the train dataset. This will be referred to as a standard training procedure.

To understand and analyze the performance of the proposed adversarial attacker, we calculate the three error metrics described in Table 2. We compare the proposed adversarial attacker with random noise attackers generated by i.i.d. perturbations. We use three additive noise distributions—Gaussian, uniform and binary, for comparisons. The output of these attackers have been normalized so as to meet the desired bound on the norm of the perturbation. The metrics are computed on the test samples where $K$ is the total number of samples in the test set. The results on the 4 datasets for varying noise levels are shown in Table 3. We also show the histograms of $\left(\|T(x_k + e_k) - y_k\|_q - \|T(x_k) - y_k\|_q\right)_{1 \leq k \leq K}$ in Figures 2, 3, 4, and 5, where $(e_k)_{1 \leq k \leq K}$ have been generated from various noise distributions and the proposed adversarial attacker.

For safety critical tasks, Lipschitz and performance targets can be specified as engineering requirements, prior to network training. Such a design approach has proven to make the network more stable and robust to adversarial attacks. Imposing a Lipschitz target can be done either by controlling the Lipschitz constant for each layer or for the whole network depending on the application at hand. One such method for
controlling the Lipschitz has been presented in [Serrurier et al., 2020] using Hinge regularization. In the experiments, we train our networks while using a spectral normalisation technique [Miyato et al., 2018] which has been proven to be very effective in controlling Lipschitz properties in GANs.

Given an $m$ layer fully connected architecture and a Lipschitz target $L$, we can constrain the spectral norm of each layer to be less than $\sqrt{L}$. This ensures that the upper bound on the global Lipschitz constant is less than $L$. We keep the network architectures exactly the same for both training procedures. The performance of adversarial attacker on standard and spectrally normalized trained model in terms of Fooling Error (E) and Symmetric Mean Accuracy Percentage error (SMAPE) for various datasets and varying perturbation magnitude is given in Table 4.

All the previous results have been obtained with attack and noise addition on all the input features present in the datasets. As pointed in Section 3.4, the introduced adversarial attacker is capable of attacking a group of inputs. While generating an adversarial attack we avoid attacking the categorical input variables [Ballet et al., 2019], hence in Abalone and industrial datasets we attack only the continuous variables. For the Combined Power plant dataset, we attack 3 out of 4 continuous variables since it does not contain any categorical variables. Similarly, for the Red-wine dataset we attack 8 continuous variables out of 11. The performance of the adversarial attacker, when attacking only few inputs, is shown in Table 5.

As emphasized in Section 3.3, our adversarial attacker is applicable for various measures of perturbation on input and output deviations. The previous results have been obtained for the value $p = q = 2$ termed as $\ell_2$ attacks here. We further show results for $p = q = 1$ termed as $\ell_1$ attacks and for $p = q = +\infty$ termed as $\ell_\infty$ attacks in Table 6.

### 4.3 Result analysis

Some general conclusions can be drawn from the experiments.

- We observe that the proposed adversarial attacker performs better than all the three random noise attackers for the three quantitative measures we have defined. In addition, the histograms in Figures 2, 3, 4, and 5 show that the error may be increased or reduced by random attackers, while this shortcoming does not happen with our adversarial attacker. This observation is verified on the norms - $\ell_2$, $\ell_1$ and $\ell_\infty$ norms in Table 6.

- Spectral normalisation has been proven to robustify the trained models. As in Table 4, we see that the Fooling error (E) and SMAPE are reduced in all the cases when compared to the standard trained model.

- In the considered examples, we observe that categorical data have little effect when attacking the trained model as shown in Table 5. The E and SMAPE measures do not show major differences.

### 5 Conclusion

In this article, we have introduced a novel easily implementable Jacobian-based adversarial attacker for estimation problems. These regression tasks cover a major portion of safety critical applications. Yet there is lack of works studying and analysing adversarial attacks in this context, as opposed to classification tasks. The present study contributes to filling this gap. We have presented error metrics which help in analysing the effectiveness of the attacker. Our attacker is versatile in the sense that it can handle any measure ($\ell_1$, $\ell_2$, $\ell_\infty$) on input or output perturbations according to the target application. Our attacker is also successful in handling attacks focused on subsets of inputs. This feature may be useful when handling specific tabular datasets and may also be insightful when information is available related to the sensitivity or ability to control some inputs. Our tests concentrated on fully connected networks, but it is worth pointing out that the proposed approach can be applied to any network architecture.

### References


Figure 2: Error distribution of random attacks and proposed adversarial attack on Combined cycle Power Plant dataset for perturbation level of $2 \times 10^{-1}$ for $\ell_2$ attack.

Figure 3: Error distribution of random attacks and proposed adversarial attack on Red-wine dataset for perturbation level of $2 \times 10^{-1}$ for $\ell_2$ attack.

Figure 4: Error distribution of random attacks and proposed adversarial attack on Abalone dataset for perturbation level of $1 \times 10^{-1}$ for $\ell_2$ attack.

Figure 5: Error distribution of random attacks and proposed adversarial attack on industrial dataset for perturbation level of $2 \times 10^{-1}$ for $\ell_2$ attack.


