Multivariate Lipschitz Analysis of the Stability of Neural Networks

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

frontiers

Multivariate Lipschitz Analysis of the Stability of Neural Networks

Kavya Gupta [1,2,*], **Fateh Kaakai**[2], **Beatrice Pesquet-Popescu**[2], **Jean-Christophe Pesquet**[1], **and Fragkiskos D. Malliaros** [1]

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

Correspondence*:
Kavya Gupta *
kavya.gupta100@gmail.com

ABSTRACT

The stability of neural networks with respect to adversarial perturbations has been extensively studied. One of the main strategies consist of quantifying the Lipschitz regularity of neural networks. In this paper, we introduce a multivariate Lipschitz constant-based stability analysis of fully connected neural networks allowing us to capture the influence of each input or group of inputs on the neural network stability. Our approach relies on a suitable re-normalization of the input space, with the objective to perform a more precise analysis than the one provided by a global Lipschitz constant. We investigate the mathematical properties of the proposed multivariate Lipschitz analysis and show its usefulness in better understanding the sensitivity of the neural network with regard to groups of inputs. We display the results of this analysis by a new representation designed for machine learning practitioners and safety engineers termed as a *Lipschitz star*. The Lipschitz star is a graphical and practical tool to analyze the sensitivity of a neural network model during its development, with regard to different combinations of inputs. By leveraging this tool, we show that it is possible to build robust-by-design models using spectral normalization techniques for controlling the stability of a neural network, given a *safety Lipschitz target*. Thanks to our multivariate Lipschitz analysis, we can also measure the efficiency of adversarial training in inference tasks. We perform experiments on various open access tabular datasets, and also on a real Thales Air Mobility industrial application subject to certification requirements.

Keywords: Lipschitz, sensitivity, stability, neural networks, tabular data, safety, adversarial attacks

1 INTRODUCTION

Artificial neural networks are at the core of recent advances in Artificial Intelligence. One of the main challenges faced today, especially by companies designing advanced industrial systems, is to ensure the safety of new generations of products using these technologies. Neural networks have been shown to be sensitive to adversarial perturbations (Szegedy et al., 2014). For example, changing a few pixels of an

*Author is employed by Thales LAS France. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

image may lead to misclassification of the image by a Deep Neural Network (DNN), which emphasizes the potential lack of stability of such architectures. DNNs being sensitive to adversarial examples, can thus be fooled, in an intentional manner (security issue) or in undeliberate/accidental manner (safety issue), which raises a major stability concern for safety-critical systems which need to be certified by an independent certification authority prior to any entry into production/operation. DNN-based solutions are hindered with such issue due to their complex nonlinear structure. Attempts towards verification of neural networks have been made for example in (Weng et al., 2019; Katz et al., 2017). It has been proven in (Tsipras et al., 2018) that there exists a trade-off between the prediction performance and the stability of of neural networks.

In the last years, the number of works devoted to the stability issue of neural networks has grown in manifolds. In these works, the terms "stability", "robustness" or "local robustness" are used interchangeably with the same meaning which is formally defined in this paper as the extent to which a neural network can continue to operate correctly despite small perturbations in its inputs. The stability criterion considered here highlights the fact that these small perturbations in the inputs do not produce high variations of the outputs. Many approaches have been proposed, some dedicated to specific architectures (e.g., networks using only ReLU activation functions) and grounded on more or less empirical techniques. We can break down broadly these techniques into three categories:

- Purely computational approaches which consist in attacking a neural network and observing its response to such attacks,
- methods based on (often clever) heuristics for testing / promoting the stability of a neural net,
- studies that aim at establishing mathematical proofs of stability.

These three kinds of strategies are useful for building and certifying effectively robust neural networks. However, the techniques based on mathematical proofs of stability are generally preferred by industrial safety experts since they enable a *safe-by-design* approach that is more efficient than a robustness verification activity done a posteriori with a necessarily bounded effort. Among the possible mathematical approaches, we focus in this article on those relying upon the analysis of the Lipschitz properties of neural networks. Such properties play a fundamental role in the understanding of the internal mechanisms governing these complex nonlinear systems. Besides, they make few assumptions on the type of non-linearities used and are thus valid for a wide range of networks. Nevertheless, they generate a number of challenges both from a theoretical and numerical standpoints.

Since DNNs are sensitive to small specific perturbations, providing a quantitative estimation of the stability of such architectures is of paramount importance for safe and secure product development in domains such as aeronautics, ground transportation, autonomous vehicles, energy, and healthcare. One metric to assess the stability of neural networks to adversarial perturbations is the Lipschitz constant, which upper bounds the ratio between output variations and input variations for a given metric. More generally, in deep learning theory, novel generalization bounds critically rely on the Lipschitz constant of the neural network (Bartlett et al., 2017). One of the main limitations of the Lipschitz constant, defined in either global or local context, is that it only provides a single parameter to quantify the robustness of a neural network. Such a single-parameter analysis does not facilitate the understanding of potential sources of instability. In particular, it may be insightful to identify the inputs which have the highest impact in terms of sensitivity. In the context of tabular data mining, the inputs often have quite heterogeneous characteristics. Some of them are categorical data, often encoded in a specific way (e.g., one-hot encoder (Hancock and Khoshgoftaar, 2020)) and among them, one can usually distinguish those which are unsorted (like labels identifying countries) or those which are sorted (like severity scores in a disease). So, it may appear useful

to analyze in a specific manner each type of inputs of a NN and even sometimes to exclude some of these inputs (e.g., unsorted categorical data for which the notion of small perturbation may be meaningless) from the performed sensitivity analysis.

The contributions of the work are summarized below:

- A multivariate analysis of the Lipschitz properties of NNs is performed by generating a set of *partial Lipschitz constants*. This opens a new dimension to studying the stability of NNs.
- Our sensitivity analysis allows us to capture the behaviour of an individual input or group of inputs.
- The results of this analysis are displayed by a new graphical representation termed as a *Lipschitz star*.
- Using the proposed analysis, we also study quantitatively the effect of spectral normalization constraint and adversarial training on the stability of NNs.
- We showcase our results on various open-source datasets along with a real industrial application in the domain of Air Traffic Management.

In the next section we give a detailed description of the state-of-the-art related to the quantification of the Lipschitz constant in neural networks. Section 3 gives our proposed method pertaining to sensitivity of inputs and introduction to Lipschitz stars. Section 4 provides an analytical evaluation for our approach with synthetic datasets. The next section gives detailed results on three open source datasets and a real safety critical industrial dataset. The last section concludes our paper.

2 OVERVIEW ON THE ESTIMATION OF THE LIPSCHITZ CONSTANT OF FEEDFORWARD NETWORKS

2.1 Theoretical background

An m-layered feedforward network can be modelled by the following recursive equations:

$$(\forall i \in \{1, \dots, m\}) \quad x_i = T_i(x_{i-1}) = R_i(W_i x_{i-1} + b_i), \tag{1}$$

where, at the i^{th} layer, $x_{i-1} \in \mathbb{R}^{N_{i-1}}$ designates the input vector, $x_i \in \mathbb{R}^{N_i}$ the output one, $W_i \in \mathbb{R}^{N_i \times N_{i-1}}$ is the weight matrix, $b_i \in \mathbb{R}^{N_i}$ is the bias vector, and $R_i \colon \mathbb{R}^{N_i} \to \mathbb{R}^{N_i}$ is the activation operator. This operator may consist of the application of basic nonlinear functions, e.g., ReLU or tanh, to each component of the input. Alternatively, it may consist of a softmax operation or group sorting operations which typically arise in max pooling. In this model, when the matrix W_i has a Toeplitz or a block-Toeplitz structure, a convolutive layer is obtained.

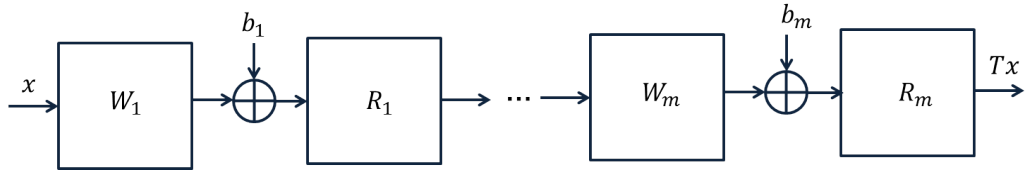

Figure 1. m-layered feedforward neural network architecture. For the i^{th}-layer, W_i is the linear weight operator, b_i the bias vector, and R_i the activation operator.

Since the seminal work in (Szegedy et al., 2014), it is known that instability in the outputs of the neural networks may arise. This issue, often referred to as the stability with respect to adversarial noise, tends

to be more severe when the training set is small. However, it may even happen with large datasets such as ImageNet. As shown in (Goodfellow et al., 2015), the problem is mainly related to the choice of the weight matrices. One way of quantifying the stability of the system is to calculate a Lipschitz constant of the network.

A *Lipschitz constant* of a function T is an upper bound on the ratio between the variations of the output values and the variations of input arguments of a function T. Thus, it is a measure of sensitivity of the function with respect to input perturbations. This means that, if $\theta \in [0, +\infty[$ is such that, for every input $x \in \mathbb{R}^{N_0}$ and perturbation $z \in \mathbb{R}^{N_0}$,

$$\|T(x+z) - T(x)\| \leq \theta \|z\|, \tag{2}$$

then θ is a Lipschitz constant of T. Note that, the same notation is used here for the norms on $\mathbb{R}^{N_0}$ and $\mathbb{R}^{N_m}$, but actually different norms can be used. If not specified, the standard Euclidean norm will be used. Another important remark which follows from the mean value inequality is that, if T is differentiable on $\mathbb{R}^{N_0}$, the optimal (i.e., smallest) Lipschitz constant is

$$\theta = \sup_{x \in \mathbb{R}^{N_0}} \|T'(x)\|_{\mathrm{S}} = \sup_{x \in \mathbb{R}^{N_0}} \sup_{\substack{z \in \mathbb{R}^{N_0} \\ z \neq 0}} \frac{\|T'(x)z\|}{\|z\|}, \tag{3}$$

where $T'(x) \in \mathbb{R}^{N_m \times N_0}$ is the Jacobian matrix of T at x and $\|\cdot\|_{\mathrm{S}}$ denotes the spectral matrix norm. Local definitions of the Lipschitz constant are also possible (Yang et al., 2020). In order to get more meaningful expressions of Lipschitz constants, an important assumption which will be made in this paper is that the operators $(R_i)_{1 \leq i \leq m}$ are nonexpansive, i.e., 1-Lipschitz. This assumption is satisfied for all the standard choices of activation operators.

The first upper-bound on the Lipschitz constant of a neural network was derived by analyzing the effect of each layer independently and considering a product of the resulting spectral norms (Goodfellow et al., 2015). This leads to the following *Trivial Upper Bound*:

$$\overline{\theta}_m = \|W_m\|_{\mathrm{S}} \|W_{m-1}\|_{\mathrm{S}} \cdots \|W_1\|_{\mathrm{S}}. \tag{4}$$

Although easy to compute, this upper bound turns out be over-pessimistic. In (Virmaux and Scaman, 2018), the problem of computing the exact Lipschitz constant of a differentiable function is pointed out to be NP-hard. A first generic algorithm (AutoLip) for upper bounding the Lipschitz constant of any differentiable function is proposed. This bound however reduces to (4) for standard feedforward neural networks. Additionally, the authors proposed an algorithm, called SeqLip, for sequential neural networks, which shows significant improvement over AutoLip. A sequential neural network is a network for which the activation operators are separable in the sense that, for every $i \in \{1, \ldots, m\}$,

$$(\forall x_i = (\xi_{i,k})_{1 \leq k \leq N_i} \in \mathbb{R}^{N_i}) \quad R_i(x) = (\rho_i(\xi_{i,k}))_{1 \leq k \leq N_i}, \tag{5}$$

where the activation function $\rho_i \colon \mathbb{R} \to \mathbb{R}$[1]. In (Virmaux and Scaman, 2018), it is assumed that the functions $(\rho_i)_{1 \leq i \leq m}$ are differentiable, increasing, and their derivative are upper bounded by one. It can be deduced

[1] More generally, a function $\rho_{i,k}$ can be applied to each component $\xi_{i,k}$ but this situation rarely happens in standard neural networks.

that a Lipschitz constant of the network is

$$\vartheta_m = \sup_{\Lambda_1 \in \mathcal{D}_{N_1}([0,1]),\ldots,\Lambda_{m-1} \in \mathcal{D}_{N_{m-1}}([0,1])} \|W_m \Lambda_{m-1} \cdots \Lambda_1 W_1\|_{\mathrm{S}}, \tag{6}$$

where $\mathcal{D}_N(I)$ designates the set of diagonal matrices of dimension $N \times N$ with diagonal values in $I \subset \mathbb{R}$. This bound simplifies as

$$\vartheta_m = \sup_{\Lambda_1 \in \mathcal{D}_{N_1}(\{0,1\}),\ldots,\Lambda_{m-1} \in \mathcal{D}_{N_{m-1}}(\{0,1\})} \|W_m \Lambda_{m-1} \cdots \Lambda_1 W_1\|_{\mathrm{S}}, \tag{7}$$

which shows that 2^{N_i} values of the diagonal elements of matrix Λ_i have to be tested at each layer $i \in \{1, \ldots, m\}$, so that the global complexity amounts to $2^{N_1+\cdots+N_{m-1}}$ and thus grows exponentially as a function of the number of neurons. Estimating the Lipschitz constant using this method is intractable even for medium-size networks; thus, the authors use a greedy algorithm to compute a bound, which may under-approximate the Lipschitz constant. This does not provide true upper bounds.

In (Combettes and Pesquet, 2020b) various bounds on the Lipschitz constant of a feedforward network are derived by assuming that, for every $i \in \{1, \ldots, m\}$ the activation operator R_i is α_i-averaged with $\alpha_i \in]0, 1]$. We recall that this means that there exists a non-expansive (i.e. 1-Lipschitz) operator Q_i such that $R_i = (1 - \alpha_i)\,\mathrm{Id} + \alpha_i Q_i$. The following inequality is then satisfied:

$$(\forall (x, y) \in \mathbb{R}^{N_i}) \quad \|R_i(x) - R_i(y)\|^2 \leq \|x - y\|^2 - \frac{1 - \alpha_i}{\alpha_i} \|x - R_i(x) - y + R_i(y)\|^2. \tag{8}$$

We thus see that the smaller α_i, the more "stable" R_i is. In the limit case when $\alpha_1 = 1$, R_i is non-expansive and, when $\alpha_i = 1/2$, R_i is said to be firmly nonexpansive. An important subclass of firmly nonexpansive operators is the class of proximity operators of convex functions which are proper and lower-semicontinuous. Let $\Gamma_0(\mathbb{R}^N)$ be the class of such functions defined from $\mathbb{R}^N$ to $]-\infty, +\infty]$. The proximity operator of a function $f \in \Gamma_0(\mathbb{R}^N)$, at some point $x \in \mathbb{R}^N$, is the unique vector denoted by $\mathrm{prox}_f(x)$ such that

$$\mathrm{prox}_f(x) = \operatorname*{argmin}_{p \in \mathbb{R}^N} \ \frac{1}{2}\|p - x\|^2 + f(p). \tag{9}$$

The proximity operator is a fundamental tool in convex optimization. As shown in (Combettes and Pesquet, 2020a), the point is that most of the activation functions (e.g., sigmoid, ReLu, leaky ReLU, ELU) currently used in neural networks are the proximity operators of some proper lower-semicontinuous convex functions. This property is also satisfied by activation operators which are not separable, like softmax or the squashing function used in capsule networks. The few activation operators which are not proximity operators (e.g., convex combinations of a max pooling and an average pooling) can be viewed as over-relaxations of proximity operators and correspond to a value of the averaging parameter greater than $1/2$.

Based on these averaging assumptions, a first estimation of the Lipschitz constant is given by

$$\theta_m = \beta_{m;\varnothing} \|W_m \circ \cdots \circ W_1\| + \sum_{k=1}^{m-1} \sum_{(j_1,\ldots,j_k) \in \mathbb{J}_{m,k}} \beta_{m;\{j_1,\ldots,j_k\}} \sigma_{m;\{j_1,\ldots,j_k\}}, \tag{10}$$

where

$$(\forall\, \mathbb{J} \subset \{1,\dots,m-1\}) \quad \beta_{m;\mathbb{J}} = \left(\prod_{j\in\mathbb{J}} \alpha_j\right) \prod_{j\in\{1,\dots,m-1\}\smallsetminus\mathbb{J}} (1-\alpha_j), \tag{11}$$

for every $k \in \{1,\dots,m-1\}$,

$$\mathbb{J}_{m,k} = \begin{cases} \big\{(j_1,\dots,j_k) \in \mathbb{N}^k \;\big|\; 1 \le j_1 < \cdots < j_k \le m-1\big\}, & \text{if } \; k > 1; \\ \{1,\dots,m-1\}, & \text{if } \; k = 1 \end{cases} \tag{12}$$

and for every $(j_1,\dots,j_k) \in \mathbb{J}_{m,k}$,

$$\sigma_{m;\{j_1,\dots,j_k\}} = \|W_m \cdots W_{j_k+1}\|_{\mathrm{S}}\, \|W_{j_k} \cdots W_{j_{k-1}+1}\|_{\mathrm{S}} \cdots \|W_{j_1} \cdots W_1\|_{\mathrm{S}} \tag{13}$$

When, for every $i \in \{1,\dots,m-1\}$, R_i is firmly nonexpansive, the expression simplifies as

$$\theta_m = \frac{1}{2^{m-1}} \left(\|W_m \cdots W_1\|_{\mathrm{S}} + \sum_{k=1}^{m-1} \sum_{(j_1,\dots,j_k)\in\mathbb{J}_{m,k}} \sigma_{m;\{j_1,\dots,j_k\}} \right) \tag{14}$$

If, for every $i \in \{1,\dots,m-1\}$, R_i is separable,[2] a second estimation is provided which reads

$$\vartheta_m = \sup_{\substack{\Lambda_1\in\mathcal{D}_{N_1}(\{2\alpha_1-1,1\}),\\ \vdots,\\ \Lambda_{m-1}\in\mathcal{D}_{N_{m-1}}(\{2\alpha_{m-1}-1,1\})}} \|W_m\Lambda_{m-1}\cdots\Lambda_1 W_1\|_{\mathrm{S}} \tag{15}$$

We thus see that, when $\alpha_1 = \cdots = \alpha_{m-1} = 1/2$, we recover (7) without making any assumption on the differentiability of the activation functions. This estimation is more accurate than the previous one in the sense that

$$\|W_m \cdots W_1\|_{\mathrm{S}} \le \vartheta_m \le \theta_m. \tag{16}$$

It is proved in (Combettes and Pesquet, 2020b) that, if the network is with non-negative weights, that is $(\forall i \in \{1,\dots,m\})\; W_i \in [0,+\infty[^{N_i\times N_i}$, the lower bound in (16) is attained, i.e.,

$$\vartheta_m = \|W_m \cdots W_1\|_{\mathrm{S}}. \tag{17}$$

Another interesting result which is established in (Combettes and Pesquet, 2020b) is that similar results hold if other norms than the Euclidean norm are used to quantify the perturbations on the input and the output. For example, for a given $i \in \{1,\dots,m\}$, for every $p \in [1,+\infty]$,

we can define the following norm:

$$\big(\forall x_i = (\xi_{i,k})_{1\le k\le N_i} \in \mathbb{R}^{N_i}\big) \quad \|x\|_p = \begin{cases} \left|\sum_{k=1}^{N_i} |\xi_{i,k}|^p\right|^{1/p}, & \text{if } \; p < +\infty \\ \sup_{1\le k\le N_i} |\xi_{i,k}|, & \text{if } \; p = +\infty. \end{cases} \tag{18}$$

[2] The result remains valid if different scalar activation functions are used in a given layer.

If $(p, q) \in [1, +\infty]^2$,

the input space $\mathbb{R}^{N_0}$ is equipped with the norm $\|\cdot\|_p$, and the output space $\mathbb{R}^{N_m}$ is equipped with the norm $\|\cdot\|_q$, a Lipschitz constant for a network with separable activation operators is

$$\vartheta_m = \sup_{\substack{\Lambda_1 \in \mathcal{D}_{N_1}([2\alpha_1 - 1, 1]), \\ \vdots, \\ \Lambda_{m-1} \in \mathcal{D}_{N_{m-1}}([2\alpha_{m-1} - 1, 1])}} \|W_m \Lambda_{m-1} \cdots \Lambda_1 W_1\|_{p,q} \tag{19}$$

$$= \sup_{\substack{\Lambda_1 \in \mathcal{D}_{N_1}(\{2\alpha_1 - 1, 1\}), \\ \vdots, \\ \Lambda_{m-1} \in \mathcal{D}_{N_{m-1}}(\{2\alpha_{m-1} - 1, 1\})}} \|W_m \Lambda_{m-1} \cdots \Lambda_1 W_1\|_{p,q} \tag{20}$$

where $\|\cdot\|_{p,q}$ is the subordinate $L_{p,q}$ matrix norm induced by the two previous norms. The ability to use norms other than the Euclidean one may be sometimes more meaningful in practice (especially for the ℓ_1 or the sup norm). However, computing such a subordinate norm is not always easy (Lewis, 2010).

2.2 SDP-based approach

The work in (Fazlyab et al., 2019) focuses on neural networks using separable activation operators. It assumes that the activation function ρ_i used at a layer $i \in \{1, \ldots, m\}$ is *slope-bounded*, i.e., there exist nonnegative parameters $\Upsilon_{\min}$ and $\Upsilon_{\max}$ such that

$$(\forall (\xi, \xi') \in \mathbb{R}^2) \qquad \xi \neq \xi' \Rightarrow \Upsilon_{\min} \leq \frac{\rho_i(\xi) - \rho_i(\xi')}{\xi - \xi'} \leq \Upsilon_{\max}.$$

As said by the authors, most activation functions satisfy this inequality with $\Upsilon_{\min} = 0$ and $\Upsilon_{\max} = 1$. In other words, the above inequality means that ρ_i is an increasing function and nonexpansive. But a known result (Combettes and Pesquet, 2008, Proposition 2.4) states that a function ρ_i satisfies these properties if and only if it is the proximity operator of some proper lower-semicontinuous convex function. So it turns out that we recover similar assumptions to those made in (Combettes and Pesquet, 2020a).
Let us thus assume that $\Upsilon_{\min} = 0$, $\Upsilon_{\max} = 1$, and $m \geq 2$. A known property is that R_i is firmly nonexpansive if and only if

$$(\forall (x, y) \in (\mathbb{R}^{N_i})^2 \quad (x - y)^\top \big(R_i(x) - R_i(y)\big) \geq \|R_i(x) - R_i(y)\|^2. \tag{21}$$

The point is that, if R_i is a separable operator, this inequality holds in a more general metric associated with a matrix

$$Q_i = \mathrm{Diag}(q_{i,1,1}, \ldots, q_{i,N_i,N_i}), \tag{22}$$

where $(\forall k \in \{1, \ldots, N_i\}^2)$ $q_{i,k,k} \geq 0$. In the following, the set of such matrices $(Q_i)_{1 \leq i \leq m-1}$ will be denoted by $\mathcal{Q}$. This means that

$$(\forall (x, y) \in (\mathbb{R}^{N_i})^2 \quad (x - y)^\top Q_i \big(R_i(x) - R_i(y)\big) \geq \big(R_i(x) - R_i(y)\big)^\top Q_i \big(R_i(x) - R_i(y)\big). \tag{23}$$

For every $(x_i, y_i) \in (\mathbb{R}^{N_i})^2$, let $x_i = R_i(W_i x_{i-1} + b_i)$ and $y_i = R_i(W_i y_{i-1} + b_i)$. It follows from (23) that

$$\big(W_i(x_{i-1} - y_{i-1})\big)^\top Q_i(x_i - y_i) \geq (x_i - y_i)^\top Q_i(x_i - y_i). \tag{24}$$

Summing for the first $m-1$ layers yields

$$\sum_{i=1}^{m-1} \big(W_i(x_{i-1} - y_{i-1})\big)^\top Q_i(x_i - y_i) \geq \sum_{i=1}^{m-1} (x_i - y_i)^\top Q_i(x_i - y_i). \tag{25}$$

On the other hand, $\vartheta_m > 0$ is a Lipschitz constant of the neural network T if

$$\vartheta_m^2 \|x_0 - y_0\|^2 \geq \|W_m(x_{m-1} - y_{m-1})\|^2. \tag{26}$$

For the latter inequality to hold, it is thus sufficient to ensure that

$$\begin{aligned} &\vartheta_m^2 \|x_0 - y_0\|^2 - \|W_m(x_{m-1} - y_{m-1})\|^2 \\ &\qquad \geq 2 \sum_{i=1}^{m-1} \big(W_i(x_{i-1} - y_{i-1})\big)^\top Q_i(x_i - y_i) - 2 \sum_{i=1}^{m-1} (x_i - y_i)^\top Q_i(x_i - y_i). \end{aligned} \tag{27}$$

This inequality can be rewritten in matrix form as

$$\begin{bmatrix} x_0 - y_0 \\ \vdots \\ x_{m-1} - y_{m-1} \end{bmatrix}^\top M(\rho_m, Q_1, \ldots, Q_{m-1}) \begin{bmatrix} x_0 - y_0 \\ \vdots \\ x_{m-1} - y_{m-1} \end{bmatrix} \geq 0 \tag{28}$$

with $\rho_m = \vartheta_m^2$ and

$$M(\rho_m, Q_1, \ldots, Q_{m-1}) = \begin{bmatrix} \rho_m \operatorname{Id}_{N_0} & -W_1^\top Q_1 & & & 0 \\ -Q_1 W_1 & 0 & \ddots & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & 0 & -W_{m-1}^\top Q_{m-1} \\ 0 & & & -Q_{m-1} W_{m-1} & 2Q_{m-1} - W_m^\top W_m \end{bmatrix}. \tag{29}$$

In the case of a network having just one hidden layer, which is mainly investigated in (Fazlyab et al., 2019), the above matrix reduces to

$$M(\rho_2, Q_1) = \begin{bmatrix} \rho_2 \operatorname{Id}_{N_0} & -W_1^\top Q_1 \\ -Q_1 W_1 & 2Q_1 - W_2^\top W_2 \end{bmatrix}. \tag{30}$$

Condition (28) is satisfied, for every $(x_0, \ldots, x_{m-1})$ and $(y_0, \ldots, y_{m-1})$ if and only if

$$M(\rho_m, Q_1, \ldots, Q_{m-1}) \succeq 0. \tag{31}$$

It is actually sufficient that this positive semidefiniteness constraint be satisfied for any matrices $(Q_1, \ldots, Q_{m-1}) \in \mathcal{Q}$ for $\sqrt{\rho}_m$ to be a Lipschitz constant. The smallest possible value of the resulting constant can be obtained by solving the following Semidefinite Programming (SDP) problem:

$$\underset{(\rho_m, Q_1, \ldots, Q_{m-1}) \in C}{\text{minimize}} \quad \rho_m, \tag{32}$$

where C is the closed convex set

$$C = \big\{(\rho_m, Q_1, \ldots, Q_{m-1}) \in [0, +\infty[\times \mathcal{Q} \;\big|\; (31) \text{ holds}\big\}. \tag{33}$$

Although there exists efficient SDP solvers, the method remains computationally intensive. A solution to reduce its computational complexity at the expense of a lower accuracy consists of restricting the optimization of the metric matrices $Q_1, \ldots, Q_{m-1}$ to a subset of $\mathcal{Q}$.

One limitation of this method is that it is tailored to the use of the Euclidean norm.

REMARK 1. *In (Fazlyab et al., 2019), it is claimed that* (23) *is valid for every metric matrix*

$$Q_i = \sum_{k=1}^{N_i} q_{i,k,k} e_k e_k^\top + \sum_{1 \le k < \ell \le N_i} q_{i,k,l} (e_k - e_\ell)(e_k - e_\ell)^\top, \tag{34}$$

where $(e_k)_{1 \le k \le N_i}$ *is the canonical basis of* $\mathbb{R}^{N_i}$ *and* $(\forall (k, \ell) \in \{1, \ldots, N_i\}^2)$ *with* $k \le \ell$, $q_{i,k,\ell} \ge 0$ *Unfortunately, this turns out to be incorrect. The erroneous statement comes from a flaw in the deduction of Lemma 1 from Lemma 2 in (Fazlyab et al., 2019). A counterexample was recently provided in (Pauli et al., 2022).*

2.3 Polynomial optimization based approach

The approach in (Latorre et al., 2020) applies to neural networks having a single output (i.e., $N_m = 1$)[3]. The authors mention that their approach is restricted to differentiable activation functions, but it is actually valid for any separable firmly nonexpansive activation operators. Indeed, when $N_m = 1$, the Lipschitz constant in (19) reduces to

$$\vartheta_m = \sup_{\substack{\Lambda_1 \in \mathcal{D}_{N_1}([0,1]), \\ \vdots, \\ \Lambda_{m-1} \in \mathcal{D}_{N_{m-1}}([0,1])}} \|W_1^\top \Lambda_1 \cdots \Lambda_{m-1} W_m^\top\|_{p^*}, \tag{35}$$

where $p^* \in [1, +\infty]$ is the dual exponent of p (such that $1/p + 1/p^* = 1$). Recall that $p \in [1, +\infty]$ is the exponent of the ℓ_p-norm equipping the input space. This shows that ϑ_m is equal to

$$\vartheta_m = \sup\{\Phi(x, \lambda_1, \ldots, \lambda_{m-1}) \mid \|x\|_p \le 1, (\lambda_i)_{1 \le i \le m-1} \in [0,1]^{N_1 + \cdots + N_{m-1}}\}, \tag{36}$$

where, for every $x \in \mathbb{R}^{N_0}$ and $(\lambda_i)_{1 \le i \le m} \in \mathbb{R}^{N_1 + \cdots + N_{m-1}}$,

$$\Phi(x, \lambda_1, \ldots, \lambda_{m-1}) = x^\top W_1^\top \operatorname{Diag}(\lambda_1) \cdots \operatorname{Diag}(\lambda_{m-1}) W_m^\top. \tag{37}$$

[3] This can be extended to multiple output network, if the output space is equipped with the $\ell_{+\infty}$ norm.

Function Φ is a multivariate polynomial of the components of its vector arguments. Therefore, if the unit ball associated with the ℓ_p norm can be described via polynomial inequalities, which happens when $p \in \mathbb{N} \setminus \{0\}$ and $p = +\infty$, then finding ϑ_m turns out to be a polynomial constrained optimization problem. Solving such an optimization problem can be achieved by solving a hierarchy of convex problems. However, the size of the hierarchy tends to grow fast and if the order of the hierarchy is truncated to a too small value, the delivered result becomes inaccurate. Leveraging the sparsity properties that might exist for the weight matrices may be helpful numerically. Note that, the approach is further improved in (Chen et al., 2020) by using Lasserre's hierarchy.

A comparison of the state-of-the-art and proposed approach is presented in Table 1.

Method	Properties	Sensitivity of Inputs
Naive upper Bound (Goodfellow et al. (2015))	spectral bound, loose bound, univariate	No
SDPLip (Fazlyab et al. (2019)	ℓ_2 norm, more scalable to broad networks, univariate	No
CPLip (Combettes and Pesquet (2020b))	$\ell_p \in [1, +\infty]$, not scalable to broad networks, univariate	No
LipOpt-k (Latorre et al. (2020))	$\ell_p \in [1, +\infty]$, univariate	No
Proposed	scalable to broad networks,multivariate	Yes

Table 1. Comparison of state-of-the-art Lipschitz estimation approaches vs the proposed one.

3 WEIGHTED LIPSCHITZ CONSTANTS FOR SENSITIVITY ANALYSIS

To extend the theoretical results presented above on the evaluation of neural network stability through their Lipschitz regularity, we present in this section a new approach based on a suitable weighting operation performed in the computation of Lipschitz constants. This enables a multivariate sensitivity analysis of the neural network stability for individual inputs or groups of inputs. We will start by motivating this weighting from a statistical standpoint. Then we will define it in a more precise manner, before discussing its resulting mathematical properties.

3.1 Statistical motivations

For tractability, assume that the perturbation at the network input is a realization of a zero-mean Gaussian distributed random vector z with $N_0 \times N_0$ covariance matrix $\Sigma \succ 0$. Then, its density upper level sets are defined as

$$C_\eta = \{z \in \mathbb{R}^{N_0} \mid z^\top \Sigma^{-1} z \leq \eta\}, \tag{38}$$

for every $\eta \in]0, +\infty[$. The set C_η defines an ellipsoid where the probability density takes its highest values. More precisely, the probability for z to belong to this set is independent of Σ (see Appendix 1) and is equal to

$$\mathsf{P}(\mathsf{z} \in C_\eta) = \frac{\gamma(N_0/2, \eta/2)}{\Gamma(N_0/2)}, \tag{39}$$

where Γ is the gamma function and γ the lower (unnormalized) incomplete gamma function.

On the other hand, let us assume that the maximum standard deviation $\sigma_{\max}$ of the components of z (i.e., square root of the maximum diagonal element of matrix Σ) is small enough. If we suppose that the network T is differentiable in the neighborhood of a given input $x \in \mathbb{R}^{N_0}$, as the input perturbation is small enough,

we can approximate the network output by the following expansion:

$$T(x+z) \simeq T(x) + T'(x)z. \tag{40}$$

Let us focus our attention on perturbations in C_η. By doing so, we impose some norm-bounded condition, which may appear more realistic for adversarial perturbations. Then, we will be interested in calculating

$$\sup_{z \in C_\eta} \|T(x+z) - T(x)\| \simeq \sup_{z \in C_\eta} \|T'(x)z\|. \tag{41}$$

By making the variable change $z' = z/\sqrt{\eta}$ and using (38),

$$\begin{aligned} \sup_{z \in C_\eta} \|T(x+z) - T(x)\| &\simeq \sqrt{\eta} \sup_{z' \in C_1} \|T'(x)z'\| \\ &= \sqrt{\eta} \sup_{\substack{z \in \mathbb{R}^{N_0} \\ z' \neq 0}} \frac{\|T'(x)z'\|}{\|z'\|_{\Sigma^{-1}}} \\ &= \sqrt{\eta}\sigma_{\max} \sup_{\substack{z' \in \mathbb{R}^{N_0} \\ z' \neq 0}} \frac{\|T'(x)z'\|}{\|z'\|_{\Omega^{-1}}}, \end{aligned} \tag{42}$$

where $\Omega = \Sigma/\sigma_{\max}^2$ and $\|\cdot\|_{\Omega^{-1}} = \sqrt{(\cdot)^\top \Omega^{-1}(\cdot)}$. This suggests that, in this context, the suitable subordinate matrix norm for computing the Lipschitz constant in (3) is obtained by weighting the Euclidean norm in the input space with Ω^{-1}. We can also deduce from (42), by setting $z'' = \Omega^{-1/2}z'$, that

$$\begin{aligned} \sup_{z \in C_\eta} \|T(x+z) - T(x)\| &\simeq \sqrt{\eta}\sigma_{\max} \sup_{\substack{z'' \in \mathbb{R}^{N_0} \\ z'' \neq 0}} \frac{\|T'(x)\Omega^{1/2}z''\|}{\|z''\|} \\ &= \sqrt{\eta}\|T'(x)\Sigma^{1/2}\|_{\mathrm{S}}. \end{aligned} \tag{43}$$

On the other hand, based on the first-order approximation in (40), $T(x+z)$ is approximately Gaussian with mean $T(x)$ and covariance matrix $T'(x)\Sigma T'(x)^\top$. As $\|T'(x)\Sigma T'(x)^\top\|_{\mathrm{S}} = \|T'(x)\Sigma^{1/2}\|_{\mathrm{S}}^2$, we see that another insightful interpretation of (43) is that, up to the scaling factor $\sqrt{\eta}$, it approximately delivers the square root of the spectral norm of the covariance matrix of the output perturbations.

3.2 New definition of a weighted Lipschitz constant

Based on the previous motivations, we propose to employ a weighted norm to define a Lipschitz constant of the network as follows:

DEFINITION 2. *Let Ω be an $N_0 \times N_0$ symmetric positive definite real-valued matrix. We say that θ_m^Ω is an Ω-weighted norm Lipschitz constant of T as described in Fig. 1 if*

$$(\forall (x,z) \in (\mathbb{R}^{N_0})^2) \quad \|T(x+z) - T(z)\| \leq \theta_m^\Omega \|z\|_{\Omega^{-1}}. \tag{44}$$

The above definition can be extended to non Euclidean norms by making use of exponents $(p, q) \in [1, +\infty]^2$ and by replacing inequality (44) with

$$(\forall (x, z) \in (\mathbb{R}^{N_0})^2) \quad \|T(x + z) - T(z)\|_q \leq \theta_m^{\Omega} \, \|\Omega^{-1/2} z\|_p. \tag{45}$$

By changes of variable, this inequality can also be rewritten as

$$(\forall (x', z') \in (\mathbb{R}^{N_0})^2) \quad \|T(\Omega^{1/2}(x' + z')) - T(\Omega^{1/2} z')\|_q \leq \theta_m^{\Omega} \, \|z'\|_p. \tag{46}$$

Therefore, we see that calculating θ_m^{Ω} is equivalent to derive a Lipschitz constant of the network T where an additional first linear layer $\Omega^{1/2}$ has been added. Throughout the rest of this section, it will be assumed that, for every $i \in \{1, \ldots, m-1\}$ the activation operator R_i is separable and α_i-averaged. It then follows from (20) that an Ω-weighted norm Lipschitz constant of T is

$$\vartheta_m^{\Omega} = \sup_{\substack{\Lambda_1 \in \mathcal{D}_{N_1}(\{2\alpha_1 - 1, 1\}), \\ \vdots, \\ \Lambda_{m-1} \in \mathcal{D}_{N_{m-1}}(\{2\alpha_{m-1} - 1, 1\})}} \|W_m \Lambda_{m-1} \cdots \Lambda_1 W_1 \Omega^{1/2}\|_{p,q}. \tag{47}$$

Although all our derivations were based on the fact that Ω is positive definite, from the latter expression we see that, by continuous extension, ϑ_m^{Ω} can be defined when Ω is a singular matrix.

3.3 Sensitivity with respect to a group of inputs

In this section, we will be interested in a specific family of weighted norms associated with the set of matrices

$$\{\Omega_{\epsilon,\mathbb{K}} \mid \varnothing \neq \mathbb{K} \subset \{1, \ldots, N_0\}, \epsilon \in]0, 1]\},$$

defined, for every nonempty subset $\mathbb{K}$ of $\{1, \ldots, N_0\}$ and for every $\epsilon \in]0, 1]$, as

$$\Omega_{\epsilon,\mathbb{K}} = \mathrm{Diag}(\sigma_{\epsilon,\mathbb{K},1}^2, \ldots, \sigma_{\epsilon,\mathbb{K},N_0}^2), \tag{48}$$

where

$$(\forall \ell \in \{1, \ldots, N_0\}) \quad \sigma_{\epsilon,\mathbb{K},\ell} = \begin{cases} 1 & \text{if } \ell \in \mathbb{K} \\ \epsilon & \text{otherwise.} \end{cases} \tag{49}$$

If we come back to the statistical interpretation in Section 3.1, $\Omega_{\epsilon,\mathbb{K}}$ is then (up to a positive scale factor) the covariance matrix of a Gaussian random vector z with independent components.[4] The components with indices in $\mathbb{K}$ have a given variance $\sigma_{\max}^2$ while the others have variance $\epsilon^2 \sigma_{\max}^2$. Such a matrix thus provides a natural way of putting emphasis on the group of inputs with indices in $\mathbb{K}$. Thus, variables $\vartheta_m^{\Omega_{\epsilon,\mathbb{K}}}$ will be termed *partial Lipschitz constants* in the following.

The next proposition lists the main properties related to the use of such weighted norms for calculating Lipschitz constants. The proofs of these results are given in Appendix 2.

PROPOSITION 3. *Let $(p, q) \in [1, +\infty]^2$. For every nonempty subset $\mathbb{K}$ of $\{1, \ldots, N_0\}$ and for every $\epsilon \in]0, 1]$, let $\Omega_{\epsilon,\mathbb{K}}$ be defined as above and let $\vartheta_m^{\Omega_{\epsilon,\mathbb{K}}}$ be defined by* (47)*. Let $\mathbb{K}_0$ and $\mathbb{K}_1$ be nonempty subsets of $\{1, \ldots, N_0\}$. Then the following hold:*

(i)

[4] Recall that this interpretation is valid when $p = 2$ in (47).

As $\epsilon \to 0$, $\vartheta_m^{\Omega_{\epsilon,\mathbb{K}_0}}$ converges to the Lipschitz constant of a network where all the inputs with indices out of $\mathbb{K}_0$ are kept constant.

(ii) *$\vartheta_m^{\Omega_{1,\mathbb{K}_0}}$ is equal to the global Lipschitz constant ϑ_m defined by* (20).

(iii) *Let $(\epsilon, \epsilon') \in]0,1]^2$. If $\Omega_{\epsilon,\mathbb{K}_0} \preceq \Omega_{\epsilon',\mathbb{K}_1}$, then $\vartheta_m^{\Omega_{\epsilon,\mathbb{K}_0}} \leq \vartheta_m^{\Omega_{\epsilon',\mathbb{K}_1}}$.*

(iv) *Function $\vartheta_m^{\Omega_{\cdot,\mathbb{K}_0}} :]0,1] \to [0, +\infty[: \epsilon \mapsto \vartheta_m^{\Omega_{\epsilon,\mathbb{K}_0}}$ is monotonically increasing.*

(v) *Let $\epsilon \in]0,1]$. If $\mathbb{K}_0 \subset \mathbb{K}_1$, then $\vartheta_m^{\Omega_{\epsilon,\mathbb{K}_0}} \leq \vartheta_m^{\Omega_{\epsilon,\mathbb{K}_1}}$.*

(vi) *Let $\epsilon \in]0,1]$, let $K \in \mathbb{N} \setminus \{0\}$, and let*

$$\omega_{K,\epsilon} = \binom{N_0 - 1}{K - 1} \left(1 + \left(\frac{N_0}{K} - 1\right)\epsilon\right). \tag{50}$$

We have

$$\max_{\substack{\mathbb{K} \subset \{1,\ldots,N_0\} \\ \operatorname{card} \mathbb{K} = K}} \vartheta_m^{\Omega_{\epsilon,\mathbb{K}}} \leq \vartheta_m \leq \frac{1}{\omega_{K,\epsilon}} \sum_{\substack{\mathbb{K} \subset \{1,\ldots,N_0\} \\ \operatorname{card} \mathbb{K} = K}} \vartheta_m^{\Omega_{\epsilon,\mathbb{K}}}. \tag{51}$$

(vii) *Let $\epsilon \in]0,1]$, let $\mathcal{P}$ be a partition of $\{1,\ldots,N_0\}$, and let $\omega_{\mathcal{P},\epsilon} = 1 + (\operatorname{card} \mathcal{P} - 1)\epsilon$. We have*

$$\max_{\mathbb{K} \in \mathcal{P}} \vartheta_m^{\Omega_{\epsilon,\mathbb{K}}} \leq \vartheta_m \leq \frac{1}{\omega_{\mathcal{P},\epsilon}} \sum_{\mathbb{K} \in \mathcal{P}} \vartheta_m^{\Omega_{\epsilon,\mathbb{K}}}. \tag{52}$$

(viii) *Let $\mathbb{K}_2$ be such that $\mathbb{K}_1 \cap \mathbb{K}_2 \neq \varnothing$ and $\mathbb{K}_1 \cup \mathbb{K}_2 = \mathbb{K}_0$. Let $p^* \in [1, +\infty]$ be such that $1/p + 1/p^* = 1$ Then*

$$\vartheta_m^{\Omega_{\epsilon,\mathbb{K}_0}} \leq \left((\vartheta_m^{\Omega_{\epsilon,\mathbb{K}_1}})^{p^*} + (\vartheta_m^{\Omega_{\epsilon,\mathbb{K}_2}})^{p^*}\right)^{1/p^*} + o(\epsilon). \tag{53}$$

Let us comment on these results. According to Property (i) in the limit case when $\epsilon \to 0$, only the inputs with indices in $\mathbb{K}_0$ are used in the computation of the associated Lipschitz constant. In turn, Property (ii) states that, when $\epsilon = 1$, we recover the classical expression of a Lipschitz constant where the perturbations on all the inputs are taken into account. In addition, based on Property (iv), the evolution of $\vartheta_m^{\Omega_{\epsilon,\mathbb{K}_0}}$ when ϵ varies from 1 to 0 provides a way of assessing how the group of inputs indexed by $\mathbb{K}_0$ contributes to the overall Lipschitz behaviour of the network. Although one would expect that summing the Lipschitz constants obtained for each group of inputs would yield the global Lipschitz constant, Properties (vi) and (vii) show that this does not hold in general whatever the way the entries are split (possibly overlapping groups of given size K or disjoint groups of arbitrary size). Instead, after suitable normalization, such sums provide upper bounds on ϑ_m. Furthermore, it follows from (ii), (51), and (52) that the difference between these normalized sums and ϑ_m tends to vanish when ϵ increases.

Note that, when looking at the sensitivity with respect to individual inputs, i.e., when the considered set of indices are singletons, both (vi) (with $K = 1$) and (vii) (with $\mathcal{P} = \{\{k\} \mid k \in \{1,\ldots,N_0\}\}$ lead to the same inequality

$$\max_{k \in \{1,\ldots,N_0\}} \vartheta_m^{\Omega_{\epsilon,\{k\}}} \leq \vartheta_m \leq \frac{1}{1 + (N_0 - 1)\epsilon} \sum_{k=1}^{N_0} \vartheta_m^{\Omega_{\epsilon,\{k\}}}. \tag{54}$$

4 VALIDATION ON SYNTHETIC DATA

4.1 Context

To highlight the need for advanced sensitivity analysis tools in the design of neural networks, we first study simple synthetic examples of polynomial systems for which we can calculate explicitly the partial Lipschitz constants. We generate input-output data for the defined systems, and train a fully connected model using a standard training, i.e., without any constraints. We compare this approach with a training subject to a spectral norm constraint on the layers.

Spectral Normalization: For safety critical tasks, Lipschitz constant and performance targets can be specified as engineering requirements, prior to network training. A Lipschitz target can be defined by a safety analysis of the acceptable perturbations for each output knowing the input range and it constitutes a current practice in many industries. Imposing this 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. Such a work for controlling the Lipschitz constant has been presented in (Serrurier et al., 2021) using Hinge regularization. In our experiments, we train networks while using a spectral normalization technique (Miyato et al., 2018) which has been proved to be 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 $\sqrt[m]{L}$. According to (4), this ensures that the upper bound on the global Lipschitz constant is less than L.

For each training, we study the effect of input variables on the stability of the networks. As proposed in Section 3.3, for a given group of inputs with indices in $\mathbb{K}$, we will quantify the partial Lipschitz constant $\vartheta_m^{\Omega_{\epsilon,\mathbb{K}}}$. The obtained value of $\vartheta_m^{\Omega_{\epsilon,\mathbb{K}}}$ allows us to evaluate how the corresponding group of variables may potentially affect the stability of the network. For simplicity, in this section, we will focus on the limit case when $\epsilon = 0$ (see the last remark in Section 3.2).

Partial Lipschitz constant values $\vartheta_m^{\Omega_{0,\mathbb{K}}}$, for all possible choices for $\mathbb{K}$, are computed using the numerical method described in Section 2.2 and compared with the theoretical values derived in the following subsection. More details on the models are also provided in these sections.

4.2 Polynomial systems

We consider regression problems where the data is synthesized by a second-order multivariate polynomial. The system to be modelled is thus described by the following function:

$$(\forall (\xi_1, \ldots, \xi_{N_0}) \in \mathbb{R}^{N_0}) \quad f(\xi_1, \ldots, \xi_{N_0}) = \sum_{k=1}^{N_0} a_k \xi_k + \sum_{k=1}^{N_0} \sum_{l=1}^{N_0} b_{k,l} \xi_k \xi_l, \tag{55}$$

where $(a_k)_{k \in N_0}$ and $(b_{k,l})_{1 \leq k,l \leq N_0}$ are the real-valued polynomial coefficients. Note that, such a polynomial system is generally not Lipschitz-continuous. The Lipschitz-continuity property only holds on every compact set. Subsequently, we will thus study this system on the hypercube $[-M, M]^{N_0}$ with $M > 0$.

The explicit values of the partial Lipschitz constant on this domain can be derived as follows. We first calculate the gradient of f

$$\nabla f(\xi_1, \ldots, \xi_{N_0}) = \big(\partial_k f(\xi_1, \ldots, \xi_{N_0})\big)_{1 \leq k \leq N_0}, \tag{56}$$

where, for every $k \in \{1, \ldots, N_0\}$, $\partial_k f$ denotes the partial derivative w.r.t. the k-th variable given by

$$\partial_k f(\xi_1, \ldots, \xi_{N_0}) = a_k + \sum_{l=1}^{N_0} (b_{k,l} + b_{l,k}) \xi_l. \tag{57}$$

For every $\mathbb{K} \subset \{1, \ldots, N_0\}$, the partial Lipschitz constant $\mathring{\vartheta}^{\Omega_{0,\mathbb{K}}}$ of the polynomial system (restricted to $[-M, M]^{N_0}$) w.r.t. the group of variables with indices in $\mathbb{K}$ is then equal to

$$\mathring{\vartheta}^{\Omega_{0,\mathbb{K}}} = \sup_{(\xi_1, \ldots, \xi_{N_0}) \in [-M,M]^{N_0}} \sqrt{\lambda_{\Omega_{0,\mathbb{K}}}(\xi_1, \ldots, \xi_{N_0})}, \tag{58}$$

where, for every diagonal matrix $\Lambda = \mathrm{Diag}(\varepsilon_1^2, \ldots, \varepsilon_{N_0}^2)$ with $(\varepsilon_1, \ldots, \varepsilon_{N_0}) \in [0, +\infty[^{N_0}$,

$$\begin{aligned} \lambda_\Lambda(\xi_1, \ldots, \xi_{N_0}) &= \| (\nabla f(\xi_1, \ldots, \xi_{N_0}))^\top \Lambda^{1/2} \|^2 \\ &= \sum_{k=1}^{N_0} \varepsilon_k \big(\partial_k f(\xi_1, \ldots, \xi_{N_0}) \big)^2. \end{aligned} \tag{59}$$

Since the partial derivatives in (57) are affine functions of the variables $(\xi_1, \ldots, \xi_{N_0})$, λ_Λ is a convex function. We deduce that the supremum in (58) is attained when $\xi_1 = \pm M, \ldots, \xi_{N_0} = \pm M$, so that $\mathring{\vartheta}^{\Omega_{0,\mathbb{K}}}$ can be computed by looking for the maximum of a finite number of values.

4.3 Numerical results

In our numerical experiments, we consider a toy example corresponding to $N_0 = 3$ and

$$(\forall (\xi_1, \xi_2, \xi_3) \in \mathbb{R}^3) \quad f(\xi_1, \xi_2, \xi_3) = \xi_1 + 100\xi_3 - \xi_2^2 + \gamma \xi_1 \xi_3, \tag{60}$$

where $\gamma \in [0, +\infty[$. We deduce from (59) that

$$\lambda_\Lambda(\xi_1, \xi_2, \xi_3) = \varepsilon_1 (1 + \gamma \xi_3)^2 + 4\varepsilon_2 \xi_2^2 + \varepsilon_3 (100 + \gamma \xi_1)^2 \tag{61}$$

and, consequently,

$$\sup_{(\xi_1, \xi_2, \xi_3) \in [-M,M]^3} \lambda_\Lambda(\xi_1, \xi_2, \xi_3) = \varepsilon_1 (1 + \gamma M)^2 + 4\varepsilon_2 M^2 + \varepsilon_3 (100 + \gamma M)^2. \tag{62}$$

By looking at the seven possible binary values of $(\varepsilon_1, \varepsilon_2, \varepsilon_3) \neq (0, 0, 0)$, we thus calculate the Lipschitz constant of f with respect to each group of inputs. For example,

- if $\varepsilon_1 = 1, \varepsilon = 0, \varepsilon_3 = 0$, we calculate $\mathring{\vartheta}^{\Omega_{0,\mathbb{K}}}$ with $\mathbb{K} = \{1\}$, i.e., evaluate the sensitivity w.r.t. the first variable;
- if $\varepsilon_1 = \varepsilon_2 = 1, \varepsilon_3 = 0$, we calculate $\mathring{\vartheta}^{\Omega_{0,\mathbb{K}}}$ with $\mathbb{K} = \{1, 2\}$, i.e., evaluate the joint sensitivity w.r.t. the first and second variables;
- if $\varepsilon_1 = \varepsilon_2 = \varepsilon_3 = 1$, we calculate $\mathring{\vartheta}^{\Omega_{0,\mathbb{K}}}$ with $\mathbb{K} = \{1, 2, 3\}$, i.e., evaluate the sensitivity w.r.t. all the variables (global Lipschitz constant).

These Lipschitz constants allow us to evaluate the intrinsic dynamics of the system, that is how it responds when its inputs vary.

Our interest will be now to evaluate how this dynamics is modified when the system is modelled by a neural network. To do so, three systems are studied by choosing $\gamma \in \{0, 1/10, 1\}$ and $M = 50$. We generate 5,000 data samples from each system, the input values being drawn independently from a random uniform distribution. While training the neural networks, the dataset is divided with a ratio of 4:1 into training and testing samples. The input is normalized using its mean and standard deviation, while the output is max-normalized. We build neural networks for approximating the systems using two hidden layers ($m = 3$) with a number of hidden neurons equal to 30 in each layer and ReLU activation functions. The training loss is the mean square error.

For different values of γ, we report the values of the partial Lipschitz constants in Tables 2, 3, and 4. The variable $\theta_{\mathbb{K}}$ corresponds to $\mathring{\vartheta}^{\Omega_{0,\mathbb{K}}}$ for the analytical value we derived from previous formulas, whereas it corresponds to the Lipschitz constant $\vartheta_3^{\Omega_{0,\mathbb{K}}}$, when computed for the neural network trained either in a standard manner or with a spectral normalization constraint. The value of L used in the spectral normalization was adjusted to obtain a similar global Lipschitz constant to the polynomial system. In the caption, we also indicate the accuracy in terms of normalized mean square error (NMSE) and normalized mean absolute error (NMAE). These values are slightly higher for constrained training, but remain quite small.

Partial LC	Analytical	Standard	Spectral Normalized
$\theta_{\{1\}}$	1	133.9	6.75
$\theta_{\{2\}}$	100	211.7	76.3
$\theta_{\{3\}}$	100	299.7	136.0
$\theta_{\{1,2\}}$	100.0	229.0	102.2
$\theta_{\{1,3\}}$	100.0	303.1	136.0
$\theta_{\{2,3\}}$	141.4	314.2	141.2
$\theta_{\{1,2,3\}}$	141.4	315.3	141.2

Table 2. Comparison of Lipschitz constant values when $\gamma = 0$. Test performance for standard training: NMSE = 0.007, NMAE = 0.005, for spectral normalization: NMSE = 0.011, NMAE = 0.009.

Partial LC	Analytical	Standard	Spectral Normalized
$\theta_{\{1\}}$	6	138.7	10.1
$\theta_{\{2\}}$	100	219.2	90.0
$\theta_{\{3\}}$	105	302.7	138.9
$\theta_{\{1,2\}}$	100.2	231.6	108.1
$\theta_{\{1,3\}}$	105.2	306.3	139.0
$\theta_{\{2,3\}}$	145	316.4	147.2
$\theta_{\{1,2,3\}}$	145.1	316.5	147.2

Table 3. Comparison of Lipschitz constant values when $\gamma = 1/10$. Test performance for standard training: NMSE = 0.006, NMAE = 0.005, for spectral normalization: NMSE = 0.009, NMAE = 0.007.

Comments on the results:

Partial LC	Analytical	Standard	Spectral Normalized
$\theta_{\{1\}}$	51	274.7	59.5
$\theta_{\{2\}}$	100	298.9	80.3
$\theta_{\{3\}}$	150	388.7	183.7
$\theta_{\{1,2\}}$	112.6	337.0	119.4
$\theta_{\{1,3\}}$	158.4	392.2	183.7
$\theta_{\{2,3\}}$	180.3	400.1	188.9
$\theta_{\{1,2,3\}}$	187.4	400.5	189.0

Table 4. Comparison of Lipschitz constant values when $\gamma = 1$. Test performance for standard training: NMSE = 0.006, MAE =0.005, for spectral normalization: NMSE = 0.014, NMAE = 0.009.

- In general, ξ_3 impacts the output of this system the most, and (ξ_2, ξ_3) mainly account for the global dynamics of the system.
- With standard training, we see that there exists a significant increase of the sensitivity with respect to the input variations, so making the neural network vulnerable to adversarial perturbations.
- By using spectral normalization, it is possible to constrain the global Lipschitz constant of the system to be close to the analytical global value while keeping a good accuracy. One may however notice an increase of the sensitivity to ξ_1 and ξ_3, and a decrease of the sensitivity to ξ_2 with respect to the original system.
- For all the three models, the values obtained with neural networks follow the same trend, for different groups of inputs, as those observed with the analytical values.
- Although the Lipschitz constant of the neural networks is computed on the whole space and the one of the system on $[-50, 50]^3$, our Lipschitz estimates appear to be consistent without resorting to a local analysis.

These observations emphasize the importance of controlling the Lipschitz constant of neural network models through specific training strategies. In addition, we see that evaluating the Lipschitz constant with respect to groups of inputs allow us to have a better understanding of the behaviour of the models.

In this section, we have discussed the proposed method for synthetic datasets. In the next section, the sensitivity analysis will be made on widely used open source datasets and an industrial dataset.

5 APPLICATION ON DIFFERENT USE CASES

5.1 Datasets and Network description

We study four regression problems involving tabular datasets to showcase our proposed multivariate analysis of the stability of neural networks. Tabular data take advantage of heterogeneous sources of information coming from different sensors or data collection processes. We apply our methods on widely used tabular datasets: *i)* Combined Cycle Power Plant dataset [5] which has 4 attributes with 9,568 instances; *ii)* Auto MPG dataset [6] consists of 398 instances with 7 attributes; *iii)* Boston Housing dataset [7] consists of 506 instances with 13 attributes. For Combined Power Plant and Auto MPG datasets, we solve a regression problem with a single output, whereas for Boston Housing dataset we consider a two-output regression

[5] https://archive.ics.uci.edu/ml/datasets/Combined+Cycle+Power+Plant

[6] https://archive.ics.uci.edu/ml/datasets/auto+mpg

[7] https://www.cs.toronto.edu/~delve/data/boston/bostonDetail.html

problem with "price" and "ptratio" as the output variables. The attributes in the dataset are a combination of continuous and categorical. The datasets are divided with a ratio of 4:1 between training and test data.

Thales Air Mobility industrial application represents the prediction of the Estimated Time En-route (ETE), meaning the time spent by an aircraft between the take-off and landing, considering a number of variables as described in Table 6. The application is important in air traffic flow management, which is an activity area where safety is critical. The purpose of the proposed sensitivity analysis is thus to help engineers in building *safe by design* models complying with given safety stability targets. The dataset consists of 2,219,097 training, 739,639 validation, and 739,891 test samples.

For all the models, we build fully connected networks with ReLU [8] activation function on all the hidden layers, except the last one. The models are trained on Keras with Tensorflow backend. The initializers are set to Glorot uniform. The network architecture of the different models, number of layers, and neurons are tabulated in Table 5. Combined Cycle Power Plant dataset with $(10, 6)$ network architecture is trained with two hidden layers having 10 and 6 hidden neurons, respectively. For Thales Air Mobility industrial application $(10 \times (30))$ implies that the neural network has 10 hidden layers with 30 neurons each. The

Dataset	Hidden layers & neurons	EPOCHS	Optimizer	learning rate
Combined Cycle Power Plant	$(10, 6)$	100	Adam	0.01
Auto MPG	$(16, 8)$	1000	RMSprop	0.001
Boston Housing	$(10, 5)$	500	RMSprop	0.001
Thales Air Mobility App.	$(10 \times (30))$	100	Adam	0.01

Table 5. Network Architecture and training setup for different datasets.

input attributes are normalized by removing their mean and scaling to unit variance.

5.2 Sensitivity analysis with respect to each input

In this section we study the effect of input variables on the stability of the networks. More specifically, we study the effect of input variations on the stability of the networks by quantifying $\vartheta_m^{\Omega_{\epsilon,\mathbb{K}}}$ with $\epsilon \in]0, 1]$, for various choices of $\mathbb{K}$, instead of a global Lipschitz constant accounting for the influence of the whole set of inputs. The obtained value of $\vartheta_m^{\Omega_{\epsilon,\mathbb{K}}}$ allows us to evaluate how the corresponding group of variables may potentially affect the stability of the network. By performing this analysis for several choices of $\mathbb{K}$, we thus generate a multivariate analysis of the Lipschitz regularity of the network.

As shown by Proposition 3, varying the ϵ parameter is also insightful since it allows us to measure how the network behaves when input perturbations are gradually more concentrated on a given subset of inputs.

Although our approach can be applied to groups of inputs, for simplicity in this section, we will focus on the case when the sets $\mathbb{K}$ reduce to singletons. In this context, we propose a new representation for displaying the results of the Lipschitz analysis of a neural network. More precisely, we plot the values of $(\vartheta_m^{\Omega_{\epsilon,\{k\}}})_{1\leq k\leq N_0}$ on a star or radar chart where each branch of the star corresponds to the index k of an input. For each value of ϵ, a new plot is obtained which is displayed in a specific color. Note that, according to Proposition 3(iv), the plots generated for different ϵ values cannot cross. When $\epsilon = 1$, we obtain an "isotropic" representation whose "radius" corresponds to the global Lipschitz constant ϑ_m of the network.

[8] We present the results only for ReLU, but we tested our approach with other activation functions such as tanh as well and found the trends in sensitivity of inputs to be similar.

Input	0	Speed	continuous
	1	Flight Distance	
	2	Departure Delay	
	3	Initial ETE	
	4	Latitude Origin	
	5	Longitude Origin	
	6	Altitude Origin	
	7	Latitude Destination	
	8	Longitude Destination	
	9	Altitude Destination	
	10	Arrival Time Slot	7 slots (categorical)
	11	Departure Time Slot	7 slots (categorical)
	12	Aircraft Category	6 classes (categorical)
	13	Airline Company	19 classes (categorical)
Output	3	Refinement ETE	continuous

Table 6. Input and output variables description for the Thales Air Mobility industrial application dataset.

This representation is called a *Lipschitz star*. All the results of our analysis will be displayed with this representation.

For each dataset, we first perform a standard training when designing the network. To facilitate comparisons, the Lipschitz star of the network trained in such standard manner is presented as the first subplot of all the figures in the paper. Next, we show the variation in terms of input sensitivity, when *i)* a Lipschitz target is imposed, and *ii)* when an adversarial training of the networks is performed. The network architecture remains unchanged, for all our experiments and each dataset, as indicated in Section 5.1. All the Lipschitz constants for each value of ϵ are calculated using LipSDP-Neuron (Fazlyab et al., 2019). Since an increased stability may come at the price of a loss of accuracy (Tsipras et al., 2019), we also report the performance of the networks on test datasets in terms of MAE (Mean Absolute Error) for each of the Lipschitz star plot.

5.3 Effect of training with specified Lipschitz target

Spectral norm constrained training is performed as explained in Section 4.1. The results are shown for our three datasets in Figures 2 - 5. On these plots, we can observe a shrinkage of the Lipschitz stars following the reduction of the target Lipschitz value. Interestingly, improving stability does not affect significantly the performance of the networks. Let us comment on the last use case in light of the obtained results.

Comments on the Thales Air Mobility industrial application From the star plots, it is clear that the various variables have a quite different effect on the Lipschitz behavior of the network. This is an expected outcome since these variables carry a different amount of information captured by learning. From Figure 5 we observe that variables 1 – Flight Distance and 3 – Initial ETE play a prominent role, while variables 5 – Longitude Origin, and 8 – Longitude Destination are also sensitive. Some plausible explanations for these facts are mentioned below.

- Flight distance: The impact of a change of this input can be significant since because of air traffic management separation rules, the commercial aircrafts cannot freely increase their speed to minimize the impact of a longer flight distance.
- Initial ETE: Modifying this input is equivalent to changing the initial conditions, which will have a significant impact. It is possible, in the worst case scenario, to accumulate other perturbations coming

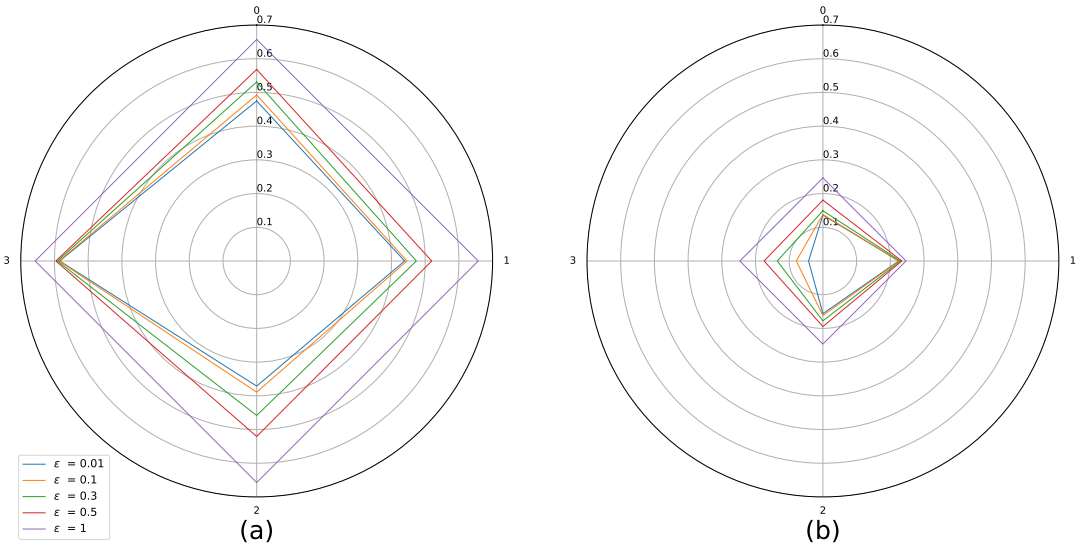

Figure 2. Sensitivity w.r.t. to each input on Combined Cycle Power Plant dataset. Influence of a spectral normalization constraint. a) Standard training: Lipschitz constant = 0.66, MAE = 0.007 , b) With spectral normalization: Lipschitz constant = 0.25, MAE = 0.0066.

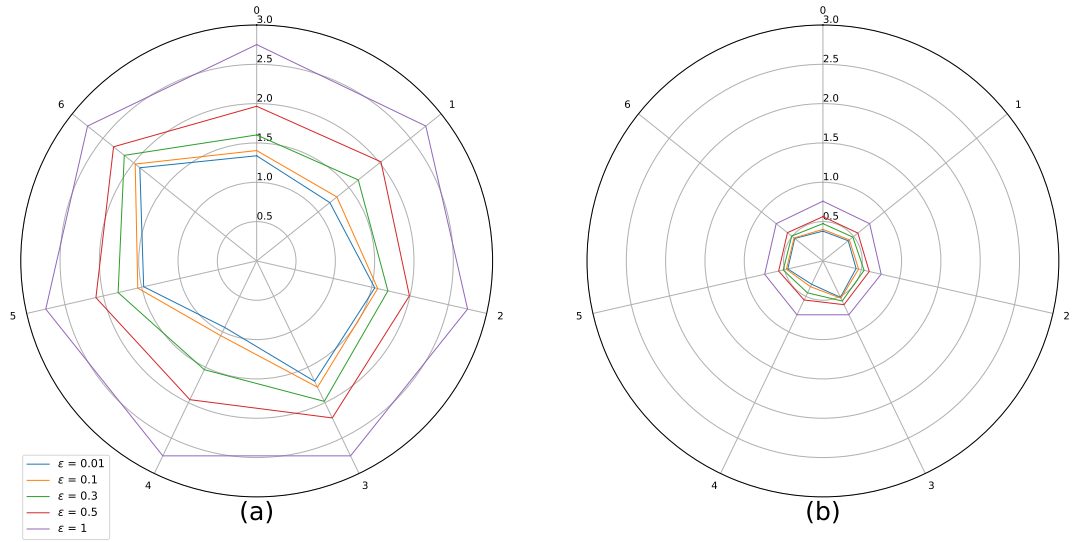

Figure 3. Sensitivity w.r.t. to each input on Auto MPG dataset. Influence of a spectral normalization constraint. a) Standard training : Lipschitz constant = 2.75 , MAE = 0.05 , b) With spectral normalization: Lipschitz constant = 0.76, MAE = 0.04.

from other coupled inputs and parameters (e.g., weather conditions) and this is probably the reason why the partial Lipschitz constant is very high, and close to the global Lipschitz constant.

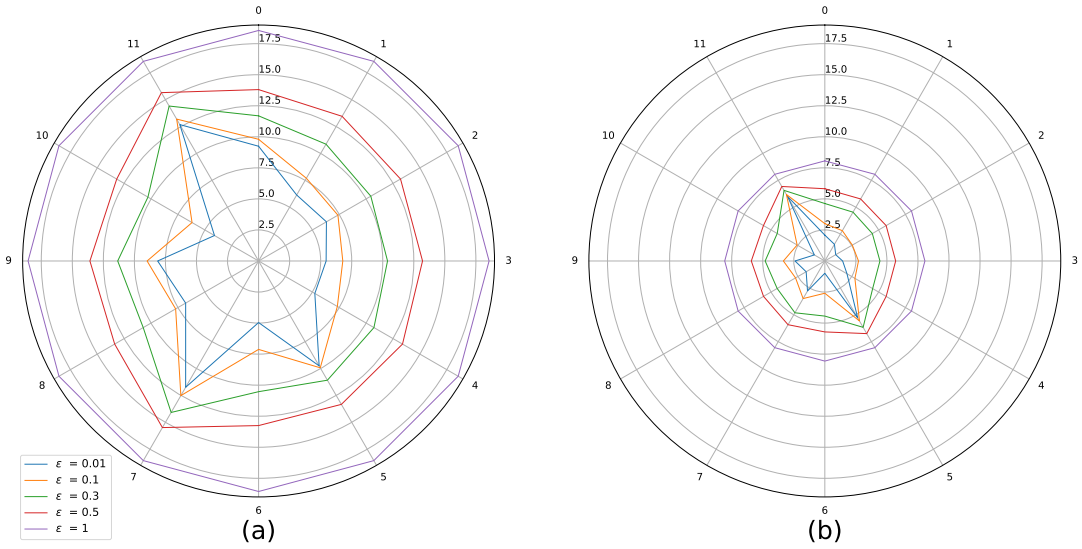

Figure 4. Sensitivity w.r.t. to each input on Boston Housing dataset. Influence of a spectral normalization constraint. a) Standard training : Lipschitz constant = 18.56, MAE(y_1) = 2.45, MAE(y_2) = 1.41, b) With spectral normalization: Lipschitz constant = 8.06 , MAE(y_1) = 2.96, MAE(y_2) = 1.35.

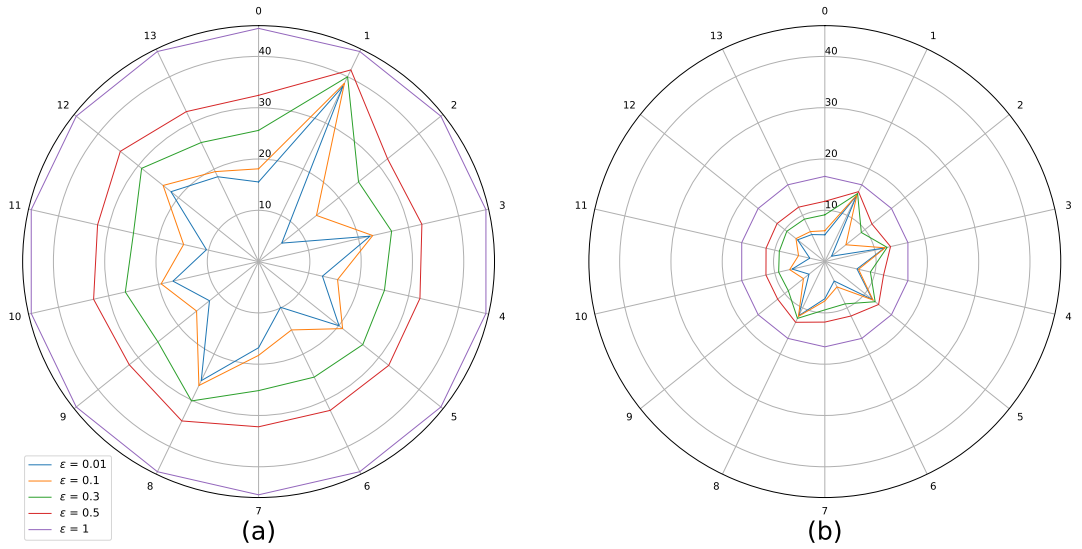

Figure 5. Sensitivity w.r.t. to each input on Thales Air Mobility industrial application. Influence of a spectral normalization constraint. a) Standard training: Lipschitz constant = 45.46, MAE = 496.37 (s), b) With spectral normalization constraint: Lipschitz constant = 16.62, MAE = 478.88 (s).

- Longitude origin and destination parameters: These parameters are related to different continents and even countries of the origin and destination airports and probably with different qualities of air traffic equipment.

5.4 Effect of adversarial training

Generating adversarial attacks and performing adversarial training constitute popular methods in designing robust neural networks. However, these techniques have received less attention for regression tasks, since most of the works deal with classification tasks (Eykholt et al., 2018; Goodfellow et al., 2015; Kurakin et al., 2018). Also, most of the existing works in the deep learning literature are for standard signal/image processing problems, whereas there are only few works handling tabular data (Ke et al., 2018; Zhang et al., 2016). One noticeable exception is (Ballet et al., 2019) which investigates problems related to adversarial attacks for classification tasks involving tabular data. Since our applications are related to regression problems for which few existing works are directly applicable, we designed a specific adversarial training method. More specifically, for a given amplitude of the adversarial noise and for each sample in the training set, we generate the worst attack based on the spectral properties of the Jacobian of the network, computed by backpropagation at this point. At each epoch of the adversarial training procedure, we solve the underlying minmax problem (Tu et al., 2019). More details on the generation of adversarial attacks for regression attacks can be found in (Gupta et al., 2021).

The generated adversarial attacks from the trained model at the previous epoch are successively concatenated to the training set for the next training epoch, much like in standard adversarial training practices using FGSM (Goodfellow et al., 2015) and Deepfool (Moosavi-Dezfooli et al., 2016) attacks. While generating adversarial attacks on tabular data, some of the variables may be more susceptible to attacks than others. The authors of (Ballet et al., 2019) take care of this aspect by using a feature importance vector. They also only attack the continuous variables, disregarding categorical ones while generating attacks. For the Power plant and Boston Housing datasets, we attack all the four input variables, while on the MPG dataset, we attack only the continuous variables. For the industrial dataset, we generate attacks for the five most sensitive input variables. We also tried attacking all the variables of the dataset but this was not observed to be more efficient. The results in form of Lipschitz star are given in Figures 6 - 9.

As expected, adversarial training leads to a shrinkage of the star plots, which indicates a better control on the stability of the trained models, while also improving slightly the MAE. In the test we did, we observe however that our adversarial training procedure is globally less efficient than the spectral normalization technique.

5.5 Sensitivity w.r.t. pair of variables

We now consider the case when the set $\mathbb{K}$ contains pairs of elements. We first show the corresponding *Partial Lipschitz constants* using a *Lipschitz star* representation in Figure 10, for the different datasets we have discussed in the article. Vertices in the Lipschitz star represent the obtained Lipschitz constant value $\vartheta_m^{\Omega_{\epsilon,\mathbb{K}}}$ for all possible combinations of pair of variables with varying values of ϵ, i.e., it represents the sensitivity w.r.t. to that particular pair.

As shown by Figure 10, this Lipschitz star representation can be useful for displaying the influence of groups of variables instead of single ones. This may be of high interest when the number of inputs is large, especially if they can be grouped into variables belonging to a given class having a specific physical meaning (e.g., electrical variables versus mechanical ones). Such Lipschitz star representation might however not be very insightful for identifying the coupling that may exist between the variables within a given group. For example, it may happen that, considered together, two variables yield an increased sensitivity than the the sensitivity of each of them individually. The reason why we need to find a better way for highlighting these coupling effects is related to Proposition 3(v) which states that, for every $\epsilon \in]0, 1]$

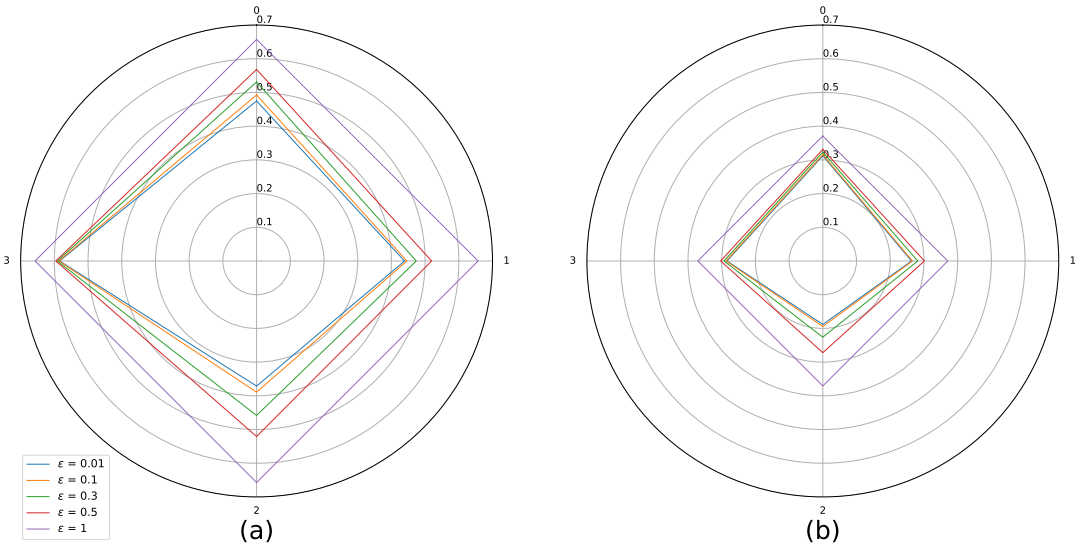

Figure 6. Sensitivity w.r.t. to each input on Combined Cycle Power Plant dataset. Effect of adversarial training. a) Standard training: Lipschitz constant = 0.657, MAE = 0.007, b) Adversarial training: Lipschitz constant = 0.37, MAE = 0.0068.

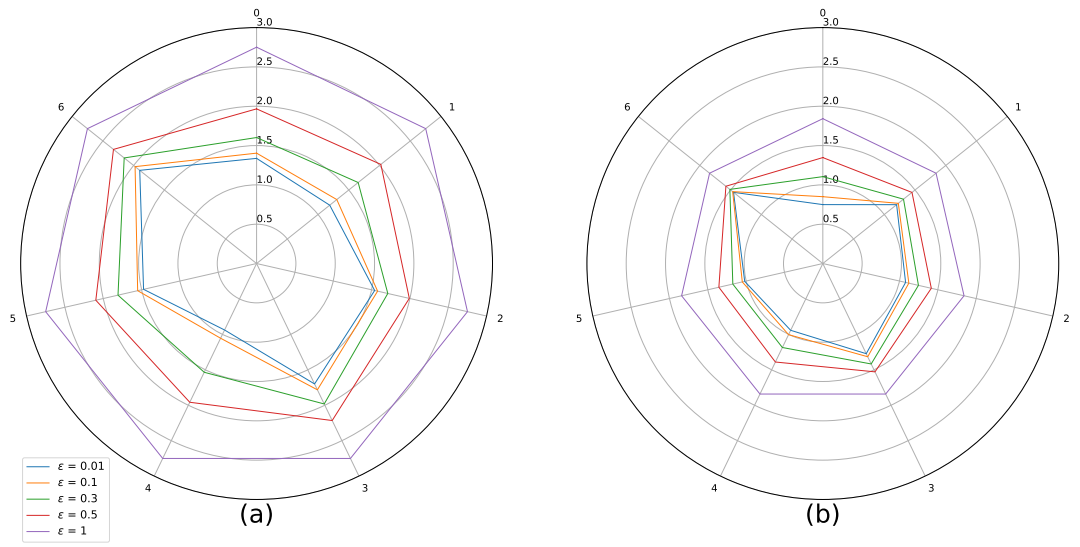

Figure 7. Sensitivity w.r.t. to each input on Auto MPG dataset. Effect of adversarial training. a) Standard training: Lipschitz constant = 2.75, MAE = 0.05, b) Adversarial training: Lipschitz constant = 1.84, MAE = 0.042.

and $(k, \ell) \in \{1, \ldots, N_0\}^2$,

$$\max\{\vartheta_m^{\Omega_{\epsilon,\{k\}}}, \vartheta_m^{\Omega_{\epsilon,\{\ell\}}}\} \leq \vartheta_m^{\Omega_{\epsilon,\{k,\ell\}}}. \tag{63}$$

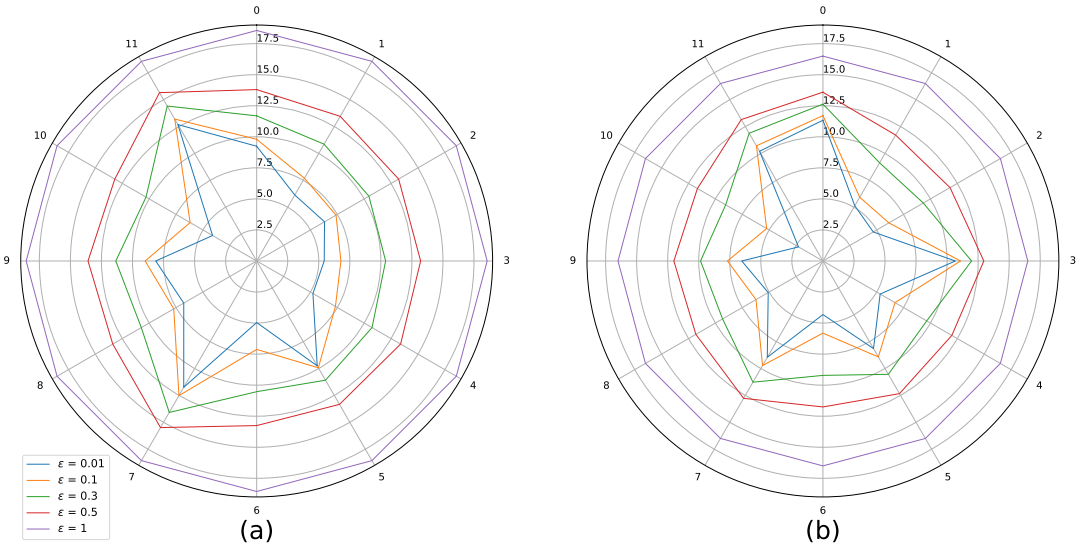

Figure 8. Sensitivity w.r.t. to each input on Boston Housing dataset. Effect of adversarial training. a) Standard training : Lipschitz = 18.56, MAE(y_1) = 2.45, MAE(y_2) = 1.41, b) Adversarial training: Lipschitz constant = 16.50, MAE(y_1) = 2.35 MAE(y_2) = 1.32.

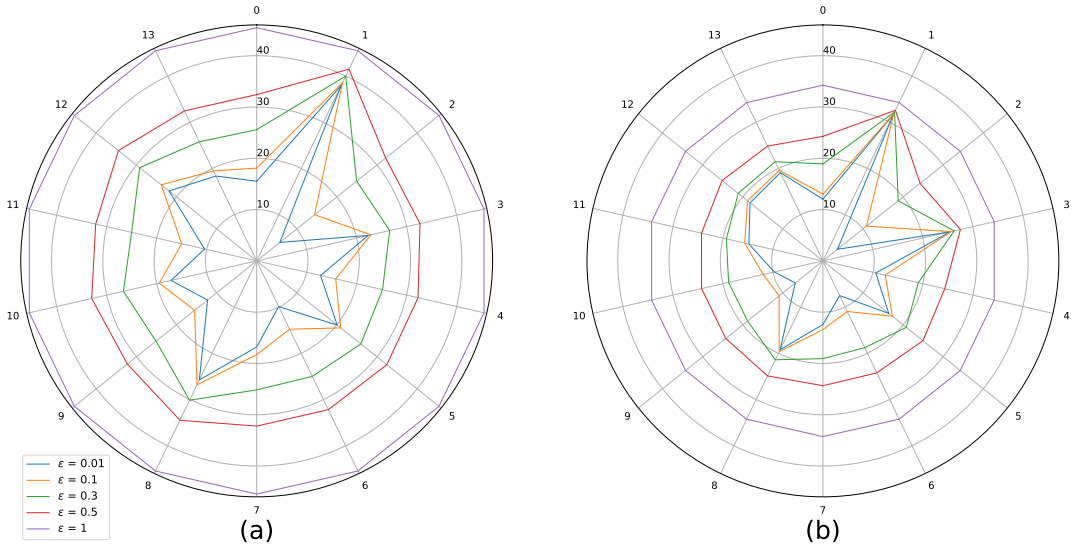

Figure 9. Sensitivity w.r.t. to each input on Thales Air Mobility industrial application. Effect of adversarial training. a) Standard training: Lipschitz = 45.47, MAE = 496.37 (s), Adversarial training. b) Lipschitz = 34.26, MAE = 494.7 (s).

This property means that, when considering a pair of inputs, the one with the highest partial Lipschitz constant will "dominate" the other. To circumvent this difficulty and make our analysis more interpretable,

we can think of normalizing the Lipschitz constant in a suitable manner. Such a strategy is a common practice in statistics when, for example, the covariance of a pair of variables is normalized by the product of their standard deviations to define their correlation factor. Once again, we can take advantage of the properties established in Proposition 3 to provide us a guideline to perform this normalization. In addition to (63), according to Property (viii),

$$\begin{aligned}\vartheta_m^{\Omega_{\epsilon,\{k,\ell\}}} &\leq \left((\vartheta_m^{\Omega_{\epsilon,\{k\}}})^{p^*} + (\vartheta_m^{\Omega_{\epsilon,\{\ell\}}})^{p^*}\right)^{1/p^*} + o(\epsilon) \\ &\leq 2^{1/p^*} \max\{\vartheta_m^{\Omega_{\epsilon,\{k\}}}, \vartheta_m^{\Omega_{\epsilon,\{\ell\}}}\} + o(\epsilon). \end{aligned} \tag{64}$$

The two previous inequalities suggest to normalize the Lipschitz constant for pairs of inputs by defining

$$\widetilde{\vartheta}_m^{\Omega_{\epsilon,\{k,\ell\}}} = \frac{1}{2^{1/p^*}-1}\left(\frac{\vartheta_m^{\Omega_{\epsilon,\{k,\ell\}}}}{\max\{\vartheta_m^{\Omega_{\epsilon,\{k\}}}, \vartheta_m^{\Omega_{\epsilon,\{\ell\}}}\}} - 1\right). \tag{65}$$

Indeed, when ϵ is close to zero, (63)-(65) show that $\widetilde{\vartheta}_m^{\Omega_{\epsilon,\{k,\ell\}}} \in [0,1]$. Note that, for the diagonal terms, $\widetilde{\vartheta}_m^{\Omega_{\epsilon,\{k,k\}}} = 0$. The higher $\widetilde{\vartheta}_m^{\Omega_{\epsilon,\{k,\ell\}}}$, the higher the gain in sensitivity due to the coupling between k and ℓ. The normalized values for the different datasets are reported in Table 7.

5.6 Interpretation of the results

We summarize some important observations/properties concerning the stability of the NNs which can be drawn from training on different datasets and leveraging the quantitative tools we have proposed in this article.

(a) Combined Power Plant Dataset

- '3 – Exhaust Vacuum' is the most sensitive variable out of the four variables.
- We observe for any variable coupled with '3' gives a higher partial Lipschitz constant.
- From Table 7(a), we see that the effect is mostly caused by the sensitivity of '3' and there is no gain when coupled with other variables. Hence, '3' dominates the overall sensitivity of the NN.
- On the other hand, we observe that, '0' when coupled with '1' and '2' becomes more sensitive as evidenced by the gain in Table 7(a).

(b) Auto MPG Dataset

- Variable '6 – Origin' and '3 – Weight' are the most sensitive variables.
- The values of partial Lipschitz constant peak when the other variables are coupled with '3' or '6'.
- From Table 7(b), we see that most of the values coupled with either '3' or '6' are close to zero, except when '3' and '6' are coupled together. Also, we see an exception when '5' is coupled with either '3' or '6'. This suggests that altogether '3', '5', and '6' have a higher impact on the stability of the network.

(c) Boston Housing Dataset

- Variable '7 – DIS' and '11 – LSTAT' are the most sensitive variables.
- We observe a high partial Lipschitz constant when coupling any variables with '7' or '11'.

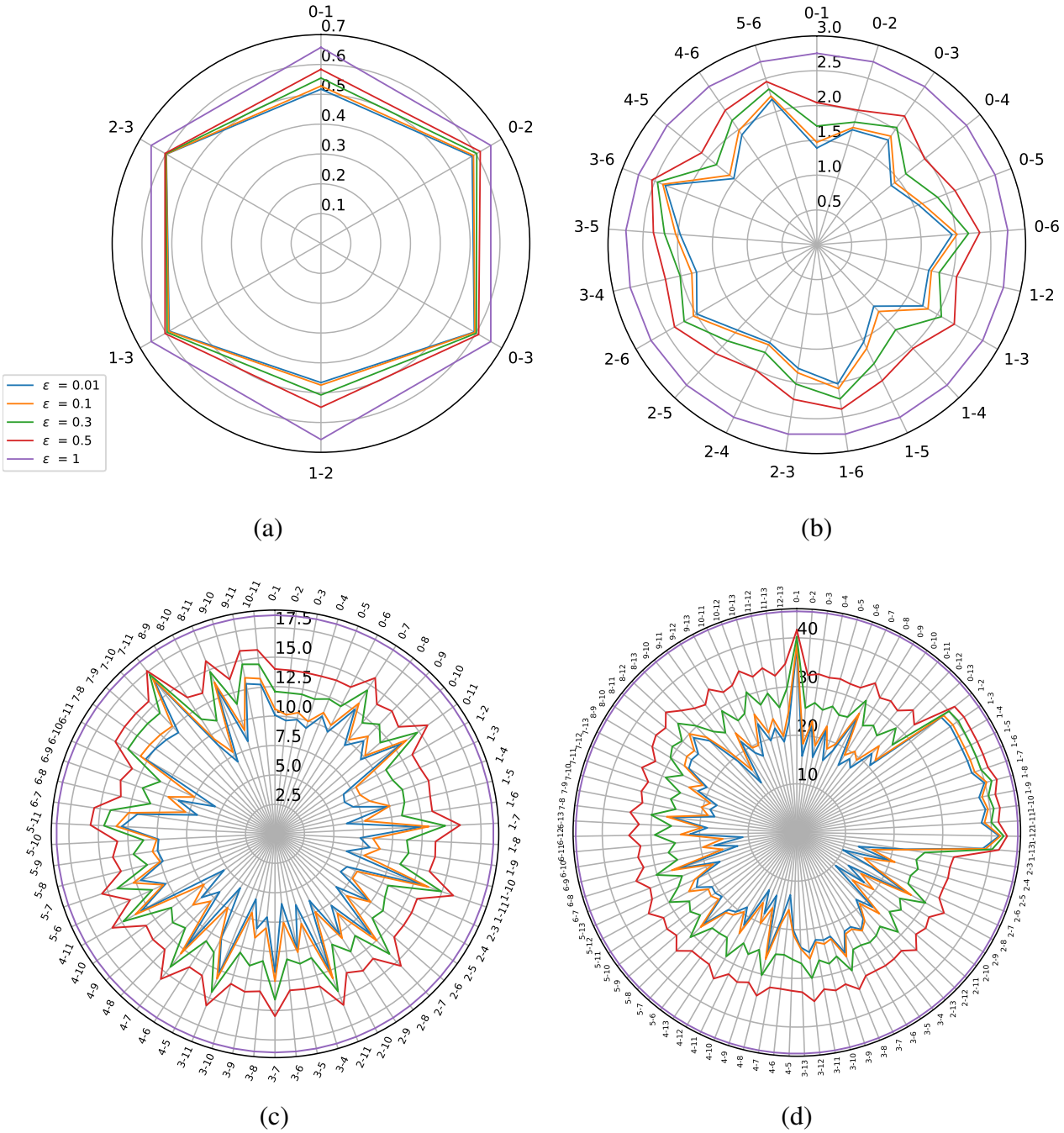

Figure 10. Sensitivity w.r.t to pair of variables on a) Combined Power Plant dataset b) Auto MPG Dataset c) Boston Housing dataset and d) Thales Air Mobility industrial application.

- From Table 7(c), we see that all the values for both '7' and '11' coupled with other variables are close to zero, except when '7' and '11' are jointly considered. Hence, '7' and '11' dominate the sensitivity of the NN.
- We observe from the table of normalized values, that '2-9' have a higher impact on the sensitivity of the NN when coupled. Similar observation can be made for pairs '2-8','1-4','3-4'.

(d) Thales Air Mobility industrial application

Variable	1	2	3
0	**0.22**	**0.57**	0.04
1		0.15	0.01
2			0.06

(a)

Variable	1	2	3	4	5	6
0	0.1	**0.29**	0.18	0.06	0.16	0.05
1		0.17	0.08	0.03	0.12	0.15
2			0.14	0.03	**0.20**	0.11
3				0.11	**0.39**	**0.56**
4					0.08	0
5						**0.34**

(b)

Variable	1	2	3	4	5	6	7	8	9	10	11
0	**0.22**	0.11	0.16	0.03	**0.25**	0.12	0.17	0.09	**0.43**	0.11	0.10
1		0.17	0.16	**0.37**	0.00	0.00	0.14	0.18	0	0.05	0.11
2			0.17	0.05	0	**0.23**	0	**0.35**	**0.61**	0.02	0.00
3				**0.35**	0.07	**0.21**	0	0.12	0.02	0.16	0.01
4					0.05	0.11	0.11	0.01	0.04	0.06	0.08
5						0.11	0.07	0.01	0.08	0.04	0.07
6							0.01	0	0.16	**0.35**	0
7								0.15	0.1	0.03	**0.76**
8									**0.27**	0.04	0.14
9										0.02	0.06
10											0.01

(c)

Variable	1	2	3	4	5	6	7	8	9	10	11	12	13
0	0.01	0.03	0.01	0.03	**0.21**	0.03	**0.23**	0.09	0.11	**0.21**	**0.27**	0.06	**0.28**
1		0	0.03	0.01	0	0	0	0.12	0.01	0.07	0	**0.24**	0.03
2			0	0.04	0.01	0.05	0.02	0	0.06	0.01	0.02	0	0.01
3				0.01	0.19	0	0.03	0.08	0.01	0.05	0.01	**0.26**	0.15
4					0	0.02	0.17	0	0.13	0.11	0.06	0.01	0.01
5						0.06	0.13	0.11	0.02	0.08	0.07	0.19	**0.27**
6							0.03	0.01	0.04	0	0.19	0.02	0.01
7								0.01	0.07	0.19	0.09	0.02	**0.32**
8									0.01	0.06	0.02	**0.29**	0.03
9										**0.27**	0.06	0.02	0.03
10											0	**0.21**	0.16
11												0.01	0.07
12													0.12

(d)

Table 7. 2^{nd} order normalized coupling matrix with $\epsilon = 0.001$ on a) Combined Power Plant Dataset b) Auto MPG Dataset c) Boston Housing Dataset and d) Thales Air Mobility industrial application.

- Variable '1 – Flight distance','3 – Initial ETE', and '8 – Longitude Destination' are the most sensitive variables.
- We see peaks in the partial Lipschitz constant values when these highly sensitive variables are coupled with other variables.
- But when analyzing the normalized tables, it becomes clear that the gain is mostly due to these sensitive variables.
- We also observe from Table 7(d), an increased sensitivity of '0' when coupled with other variables '5', '7', '10', '11', and '13'.

6 CONCLUSION

We have proposed a new multivariate analysis of the Lipschitz regularity of a neural network. Our approach, whose theoretical foundations are given in Section 3, allows the sensitivity with respect to any group of inputs to be highlighted. We have introduced a new "Lipschitz star" representation which is helpful to display how each input or group of inputs contributes to the global Lipschitz behaviour of a network. The use of these tools has been illustrated on four regression use cases involving tabular data. The improvements brought by two robust training methods (training subject to Lipschitz bounds and adversarial training) have been measured. More generally the proposed methodology is applicable to various machine learning tasks to build "safe-by-design" models where heterogeneous/multimodal/multi-omic data can be used.

APPENDICES

1 PROOF OF (39)

Let $B(0,\sqrt{\eta})$ be the closed ball of $\mathbb{R}^{N_0}$ with center 0 and radius $\sqrt{\eta}$ and let S_{N_0-1} be the surface of the unit-radius sphere in $\mathbb{R}^{N_0-1}$. We have

$$\begin{aligned}
\mathsf{P}(\mathsf{z}\in C_\eta) &= \frac{1}{(2\pi)^{N_0/2}(\det(\Omega))^{1/2}}\int_{C_\eta}\exp\Big(-\frac{1}{2}z^\top\Omega^{-1}z\Big)\,dz\\
&= \frac{1}{(2\pi)^{N_0/2}}\int_{B(0,\sqrt{\eta})}\exp\Big(-\frac{1}{2}\|u\|^2\Big)\,du\\
&= \frac{1}{(2\pi)^{N_0/2}}S_{N_0-1}\int_0^{\sqrt{\eta}}\rho^{N_0-1}\exp\Big(-\frac{\rho^2}{2}\Big)\,d\rho\\
&= \frac{1}{(2\pi)^{N_0/2}}\frac{2(\pi)^{N_0/2}}{\Gamma(N_0/2)}\int_0^{\sqrt{\eta}}\rho^{N_0-1}\exp\Big(-\frac{\rho^2}{2}\Big)\,d\rho\\
&= \frac{1}{\Gamma(N_0/2)}\int_0^{\frac{\eta}{2}}\zeta^{N_0/2-1}\exp(-\zeta)\,d\zeta\\
&= \frac{\gamma(N_0/2,\eta/2)}{\Gamma(N_0/2)}.
\end{aligned}\tag{66}$$

2 PROOFS OF PROPOSITION 3

(i): As $\epsilon\to 0$, the diagonal elements of $\Omega_{\epsilon,\mathbb{K}_0}$ with index $\ell\notin\mathbb{K}$ tend to zero, which in (47) amounts to assuming that the corresponding components of the input perturbation are zero. The existence of the limit $\Omega_{0,\mathbb{K}_0}$ is secured based on the remark at the end of Section3.2.
(ii): If $\epsilon=1$, then $\Omega_{\epsilon,\mathbb{K}_0}=\mathrm{Id}_{N_0}$ and (47) reduces to (20).
(iii): For every $i\in\{1,\dots,m-1\}$, let $\Lambda_i\in\mathcal{D}_{N_i}(\{2\alpha_i-1,1\})$. Then, by using the triangle inequality,

$$\|W_m\Lambda_{m-1}\cdots\Lambda_1W_1(\Omega_{\epsilon,\mathbb{K}_0})^{1/2}\|_{p,q}\le\|W_m\Lambda_{m-1}\cdots\Lambda_1W_1(\Omega_{\epsilon',\mathbb{K}_1})^{1/2}\|_{p,q}\|(\Omega_{\epsilon',\mathbb{K}_1})^{-1}\Omega_{\epsilon,\mathbb{K}_0}\|_{p,p}^{1/2}.\tag{67}$$

By taking the supremum of both sides with respect to the $(\Lambda_i)_{1\le i\le m-1}$ matrices, we deduce that

$$\vartheta_m^{\Omega_{\epsilon,\mathbb{K}_0}}\le\|(\Omega_{\epsilon',\mathbb{K}_1})^{-1}\Omega_{\epsilon,\mathbb{K}_0}\|_{p,p}^{1/2}\;\vartheta_m^{\Omega_{\epsilon',\mathbb{K}_1}}.\tag{68}$$

On the other hand,

$$(\Omega_{\epsilon',\mathbb{K}_1})^{-1}\Omega_{\epsilon,\mathbb{K}_0} = \mathrm{Diag}\left(\frac{\sigma^2_{\epsilon,\mathbb{K}_0,1}}{\sigma^2_{\epsilon',\mathbb{K}_1,1}},\dots,\frac{\sigma^2_{\epsilon,\mathbb{K}_0,N_0}}{\sigma^2_{\epsilon',\mathbb{K}_1,N_0}}\right), \tag{69}$$

where, by using the fact that $\Omega_{\epsilon,\mathbb{K}_0} \preceq \Omega_{\epsilon',\mathbb{K}_1}$,

$$(\forall \ell \in \{1,\dots,N_0\}) \quad \frac{\sigma_{\epsilon,\mathbb{K}_0,\ell}}{\sigma_{\epsilon',\mathbb{K}_1,\ell}} \leq 1. \tag{70}$$

Since $(\Omega_{\epsilon',\mathbb{K}_1})^{-1}\Omega_{\epsilon,\mathbb{K}_0}$ is a diagonal matrix with elements lower than or equal to 1, $\|(\Omega_{\epsilon',\mathbb{K}_1})^{-1}\Omega_{\epsilon,\mathbb{K}_0}\|_{p,p} \leq 1$ and it follows from (68) that

$$\vartheta_m^{\Omega_{\epsilon,\mathbb{K}_0}} \leq \vartheta_m^{\Omega_{\epsilon',\mathbb{K}_1}}. \tag{71}$$

(iv): Let $(\epsilon,\epsilon') \in]0,1]^2$ with $\epsilon < \epsilon'$. We have $\Omega_{\epsilon,\mathbb{K}_0} \preceq \Omega_{\epsilon',\mathbb{K}_0}$ and, according to (iii),

$$\vartheta_m^{\Omega_{\epsilon,\mathbb{K}_0}} \leq \vartheta_m^{\Omega_{\epsilon',\mathbb{K}_0}}. \tag{72}$$

(v): If $\mathbb{K}_0 \subset \mathbb{K}_1$, then $\Omega_{\epsilon,\mathbb{K}_0} \preceq \Omega_{\epsilon,\mathbb{K}_1}$ and the result follows from (iii).
(vi): We have

$$\sum_{\substack{\mathbb{K}\subset\{1,\dots,N_0\}\\ \mathrm{card}\,\mathbb{K}=K}} \Omega_{\epsilon,\mathbb{K}}^{1/2} = \left(\binom{N_0-1}{K-1} + \left(\binom{N_0}{K} - \binom{N_0-1}{K-1}\right)\epsilon\right)\mathrm{Id}_{N_0}. \tag{73}$$

By using the relation

$$\binom{N_0}{K} = \frac{N_0}{K}\binom{N_0-1}{K-1}, \tag{74}$$

we deduce that

$$\sum_{\substack{\mathbb{K}\subset\{1,\dots,N_0\}\\ \mathrm{card}\,\mathbb{K}=K}} \Omega_{\epsilon,\mathbb{K}}^{1/2} = \omega_{K,\epsilon}\,\mathrm{Id}_{N_0}. \tag{75}$$

For every $i \in \{1,\dots,m-1\}$, let $\Lambda_i \in \mathcal{D}_{N_i}(\{2\alpha_i - 1, 1\})$. Then,

$$\begin{aligned}\|W_m\Lambda_{m-1}\cdots\Lambda_1 W_1\|_{p,q} &= \frac{1}{\omega_{K,\epsilon}}\left\|W_m\Lambda_{m-1}\cdots\Lambda_1 W_1\Big(\sum_{\substack{\mathbb{K}\subset\{1,\dots,N_0\}\\ \mathrm{card}\,\mathbb{K}=K}}\Omega_{\epsilon,\mathbb{K}}^{1/2}\Big)\right\|_{p,q}\\ &\leq \frac{1}{\omega_{K,\epsilon}}\sum_{\substack{\mathbb{K}\subset\{1,\dots,N_0\}\\ \mathrm{card}\,\mathbb{K}=K}}\|W_m\Lambda_{m-1}\cdots\Lambda_1 W_1\Omega_{\epsilon,\mathbb{K}}^{1/2}\|_{p,q}.\end{aligned} \tag{76}$$

We deduce that

$$\begin{aligned}\vartheta_m = & \sup_{\substack{\Lambda_1\in\mathcal{D}_{N_1}(\{2\alpha_1-1,1\}),\\ \vdots,\\ \Lambda_{m-1}\in\mathcal{D}_{N_{m-1}}(\{2\alpha_{m-1}-1,1\})}} \|W_m\Lambda_{m-1}\cdots\Lambda_1 W_1\|_{p,q}\\ &\leq \frac{1}{\omega_{K,\epsilon}}\sum_{\substack{\mathbb{K}\subset\{1,\ldots,N_0\}\\ \operatorname{card}\mathbb{K}=K}} \sup_{\substack{\Lambda_1\in\mathcal{D}_{N_1},\\ \vdots,\\ \Lambda_{m-1}\in\mathcal{D}_{N_{m-1}}}} \|W_m\cdots\Lambda_1 W_1\Omega_{\epsilon,\mathbb{K}}^{1/2}\|_{p,q}\\ &= \frac{1}{\omega_{K,\epsilon}}\sum_{\substack{\mathbb{K}\subset\{1,\ldots,N_0\}\\ \operatorname{card}\mathbb{K}=K}} \vartheta_m^{\Omega_{\epsilon,\mathbb{K}}}. \end{aligned} \tag{77}$$

Furthermore, according to (ii) and (iv), for every $\mathbb{K}\subset\{1,\ldots,N_0\}$,

$$\vartheta_m^{\Omega_{\epsilon,\mathbb{K}}} \leq \vartheta_m^{\Omega_{1,\mathbb{K}}} = \vartheta_m. \tag{78}$$

This yields

$$\max_{\substack{\mathbb{K}\subset\{1,\ldots,N_0\}\\ \operatorname{card}\mathbb{K}=K}} \vartheta_m^{\Omega_{\epsilon,\mathbb{K}}} \leq \vartheta_m. \tag{79}$$

(vii): The proof is similar to that of (vi) by noticing that, if $\mathcal{P}$ is a partition of $\{1,\ldots,N_0\}$, then

$$\sum_{\mathbb{K}\in\mathcal{P}} \Omega_{\epsilon,\mathbb{K}}^{1/2} = \omega_{\mathcal{P},\epsilon}\operatorname{Id}_{N_0}. \tag{80}$$

(viii): For every $\varnothing\neq\mathbb{K}\subset\{1,\ldots,N_0\}$, $\vartheta_m^{\Omega_{\epsilon,\mathbb{K}}} = \vartheta_m^{\Omega_{0,\mathbb{K}}} + o(\epsilon)$. It is thus sufficient to prove this inequality in the limit case when $\epsilon\to 0$. For every $i\in\{1,\ldots,m-1\}$, let $\Lambda_i\in\mathcal{D}_{N_i}(\{2\alpha_i-1,1\})$. Let $x\in\mathbb{R}^{N_0}$ and let $x_{\mathbb{K}}$ be the projection of x onto the space of vectors whose components indexed by $\{1,\ldots,N_0\}\setminus\mathbb{K}$ are zero. We have thus $x_{\mathbb{K}_0} = x_{\mathbb{K}_1} + x_{\mathbb{K}_2}$. Then,

$$\begin{aligned}&\|W_m\Lambda_{m-1}\cdots\Lambda_1 W_1\Omega_{0,\mathbb{K}_0}^{1/2}x\|_q\\ &= \|W_m\Lambda_{m-1}\cdots\Lambda_1 W_1 x_{\mathbb{K}_0}\|_q\\ &\leq \|W_m\Lambda_{m-1}\cdots\Lambda_1 W_1 x_{\mathbb{K}_1}\|_q + \|W_m\Lambda_{m-1}\cdots\Lambda_1 W_1 x_{\mathbb{K}_2}\|_q\\ &= \|W_m\Lambda_{m-1}\cdots\Lambda_1 W_1\Omega_{0,\mathbb{K}_1}^{1/2}x_{\mathbb{K}_1}\|_q + \|W_m\Lambda_{m-1}\cdots\Lambda_1 W_1\Omega_{0,\mathbb{K}_2}^{1/2}x_{\mathbb{K}_2}\|_q\\ &\leq \|W_m\Lambda_{m-1}\cdots\Lambda_1 W_1\Omega_{0,\mathbb{K}_1}^{1/2}\|_{p,q}\|x_{\mathbb{K}_1}\|_p + \|W_m\Lambda_{m-1}\cdots\Lambda_1 W_1\Omega_{0,\mathbb{K}_2}^{1/2}\|_{p,q}\|x_{\mathbb{K}_2}\|_p. \end{aligned} \tag{81}$$

By using Hölder's inequality, we deduce that

$$\begin{aligned}&\|W_m\Lambda_{m-1}\cdots\Lambda_1 W_1\Omega_{0,\mathbb{K}_0}^{1/2}x\|_q\\ &\qquad\leq \left(\|W_m\cdots\Lambda_1 W_1\Omega_{0,\mathbb{K}_1}^{1/2}\|_{p,q}^{p^*} + \|W_m\cdots\Lambda_1 W_1\Omega_{0,\mathbb{K}_2}^{1/2}\|_{p,q}^{p^*}\right)^{1/p^*}(\|x_{\mathbb{K}_1}\|_p^p + \|x_{\mathbb{K}_2}\|_p^p)^{1/p}. \end{aligned} \tag{82}$$

Since $\|x_{\mathbb{K}_1}\|_p^p + \|x_{\mathbb{K}_2}\|_p^p = \|x_{\mathbb{K}_0}\|_p^p$, it follows that

$$\|W_m\Lambda_{m-1}\cdots\Lambda_1 W_1\Omega_{0,\mathbb{K}_0}^{1/2}\|_{p,q} \leq \left(\|W_m\cdots\Lambda_1 W_1\Omega_{0,\mathbb{K}_1}^{1/2}\|_{p,q}^{p^*} + \|W_m\cdots\Lambda_1 W_1\Omega_{0,\mathbb{K}_2}^{1/2}\|_{p,q}^{p^*}\right)^{1/p^*}. \quad (83)$$

Taking the supremum with respect to $(\Lambda_i)_{1\leq i\leq m-1}$ and majorizing the supremum of the sum in the right-hand side by the sum of the suprema yield (53).

REFERENCES

Ballet, V., Renard, X., Aigrain, J., Laugel, T., Frossard, P., and Detyniecki, M. (2019). Imperceptible adversarial attacks on tabular data. In *NeurIPS 2019 Workshop on Robust AI in Financial Services: Data, Fairness, Explainability, Trustworthiness and Privacy (Robust AI in FS 2019)*

Bartlett, P. L., Foster, D. J., and Telgarsky, M. J. (2017). Spectrally-normalized margin bounds for neural networks. In *Advances in Neural Information Processing Systems*. 6240–6249

Chen, T., Lasserre, J. B., Magron, V., and Pauwels, E. (2020). Semialgebraic optimization for lipschitz constants of relu networks. *Advances in Neural Information Processing Systems* 33, 19189–19200

Combettes, P. L. and Pesquet, J.-C. (2008). Proximal thresholding algorithm for minimization over orthonormal bases. *SIAM Journal on Optimization* 18, 1351–1376

Combettes, P. L. and Pesquet, J.-C. (2020a). Deep neural network structures solving variational inequalities. *Set-Valued and Variational Analysis* 28, 1–28

Combettes, P. L. and Pesquet, J.-C. (2020b). Lipschitz certificates for neural network structures driven by averaged activation operators. *SIAM Journal on Mathematics of Data Science* 2, 529–557

Eykholt, K., Evtimov, I., Fernandes, E., Li, B., Rahmati, A., Xiao, C., et al. (2018). Robust physical-world attacks on deep learning visual classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1625–1634

Fazlyab, M., Robey, A., Hassani, H., Morari, M., and Pappas, G. (2019). Efficient and accurate estimation of Lipschitz constants for deep neural networks. In *Advances in Neural Information Processing Systems*. 11423–11434

Goodfellow, I., Shlens, J., and Szegedy, C. (2015). Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*

Gupta, K., Pesquet, J.-C., Pesquet-Popescu, B., Malliaros, F., and Kaakai, F. (2021). An adversarial attacker for neural networks in regression problems. In *IJCAI Workshop on Artificial Intelligence Safety (AI Safety)*

Hancock, J. T. and Khoshgoftaar, T. M. (2020). Survey on categorical data for neural networks. *Journal of Big Data* 7, 1–41

Katz, G., Barrett, C., Dill, D. L., Julian, K., and Kochenderfer, M. J. (2017). Reluplex: An efficient SMT solver for verifying deep neural networks. In *International Conference on Computer Aided Verification* (Springer), 97–117

Ke, G., Zhang, J., Xu, Z., Bian, J., and Liu, T.-Y. (2018). TabNN: A universal neural network solution for tabular data

Kurakin, A., Goodfellow, I. J., and Bengio, S. (2018). Adversarial examples in the physical world. In *Artificial intelligence safety and security* (Chapman and Hall/CRC). 99–112

Latorre, F., Rolland, P. T. Y., and Cevher, V. (2020). Lipschitz constant estimation for neural networks via sparse polynomial optimization. In *8th International Conference on Learning Representations*

Lewis, A. D. (2010). A top nine list: Most popular induced matrix norms

Miyato, T., Kataoka, T., Koyama, M., and Yoshida, Y. (2018). Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*

Moosavi-Dezfooli, S.-M., Fawzi, A., and Frossard, P. (2016). Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2574–2582

Pauli, P., Koch, A., Berberich, J., Kohler, P., and Allgöwer, F. (2022). Training robust neural networks using lipschitz bounds. *IEEE Control Systems Letters* 6, 121–126. doi:10.1109/LCSYS.2021.3050444

Serrurier, M., Mamalet, F., González-Sanz, A., Boissin, T., Loubes, J.-M., and del Barrio, E. (2021). Achieving robustness in classification using optimal transport with hinge regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 505–514

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I. J., et al. (2014). Intriguing properties of neural networks. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*

Tsipras, D., Santurkar, S., Engstrom, L., Turner, A., and Madry, A. (2018). There is no free lunch in adversarial robustness (but there are unexpected benefits). *arXiv preprint arXiv:1805.12152* 2

Tsipras, D., Santurkar, S., Engstrom, L., Turner, A., and Madry, A. (2019). Robustness may be at odds with accuracy. In *International Conference on Learning Representations*

Tu, Z., Zhang, J., and Tao, D. (2019). Theoretical analysis of adversarial learning: A minimax approach. *Advances in Neural Information Processing Systems* 32

Virmaux, A. and Scaman, K. (2018). Lipschitz regularity of deep neural networks: analysis and efficient estimation. In *Advances in Neural Information Processing Systems*. 3835–3844

Weng, L., Chen, P.-Y., Nguyen, L., Squillante, M., Boopathy, A., Oseledets, I., et al. (2019). Proven: Verifying robustness of neural networks with a probabilistic approach. In *International Conference on Machine Learning* (PMLR), 6727–6736

Yang, Y.-Y., Rashtchian, C., Zhang, H., Salakhutdinov, R. R., and Chaudhuri, K. (2020). A closer look at accuracy vs. robustness. *Advances in neural information processing systems* 33, 8588–8601

Zhang, W., Du, T., and Wang, J. (2016). Deep learning over multi-field categorical data. In *European Conference on Information Retrieval* (Springer), 45–57