A Quantitative Analysis Of The Robustness Of Neural Networks For Tabular Data

Kavya Gupta, Beatrice Pesquet-Popescu, Fateh Kaakai, Jean-Christophe Pesquet

A QUANTITATIVE ANALYSIS OF THE ROBUSTNESS OF NEURAL NETWORKS FOR TABULAR DATA

Kavya Gupta[1,2] *, Beatrice Pesquet-Popescu*[2] *, Fateh Kaakai*[2]*, Jean-Christophe Pesquet*[1]

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

ABSTRACT

This paper presents a quantitative approach to demonstrate the robustness of neural networks for tabular data. These data form the backbone of the data structures found in most industrial applications. We analyse the effect of various widely used techniques we encounter in neural network practice, such as regularization of weights, addition of noise to the data, and positivity constraints. This analysis is performed by using three state-of-the-art techniques, which provide mathematical proofs of robustness in terms of Lipschitz constant for feed-forward networks. The experiments are carried out on two prediction tasks and one classification task. Our work brings insights into building robust neural network architectures for safety critical systems that require certification or approval from a competent authority.

***Index Terms*—** Lipschitz stability, robustness, safety, tabular data, neural networks, supervised learning

1. INTRODUCTION

Neural networks (NNs) find numerous applications in a growing number of applicative fields. However, one of their main weaknesses is that they are sensitive to adversarial examples [1, 2]. This means that they can be fooled, in a deliberate or unintentional manner, which raises many safety and security issues while deploying them for industrial usage. Many industry applications (autonomous vehicles, aeronautics, railway, space, etc.) are safety or mission critical systems requiring a certification or an approval from a competent authority, i.e. a demonstration that the system is acceptably secure and trustworthy. This demonstration should be done via a complete, documented and valid argument that the system using NNs (or other AI technologies) behaves only as specified in the specified context of the final user and continues to behave in this way in accordance with monitoring criteria. NN solutions are hindered with certification issues due to their intrinsic complex structure. Traditional coverage-based approaches may be irrelevant for testing neural network systems. Code certifiability can be trivially satisfied while providing only limited guarantees on the safe behaviour of the system when facing situations which have not been strictly met during the training process. Few works in the literature such as [3, 4, 5] study the provable defenses that can be provided for certification against adversarial perturbations. An attempt towards property verification of neural networks in safety critical applications was made in [6].

Deep learning on tabular data has received less attention than deep learning for standard signal/image processing problems particularly seen in the area of computer vision and natural language processing. Tabular data allows to take advantage of heterogeneous sources of information coming from different sensors or registered variables (like altitude of an aircraft, departure and destination airport, duration of the flight, company type). Tabular data analysis covers a wide variety of applications, e.g. fraud detection, product failure prediction, anti-money laundering, recommendation systems [7], click-through rate prediction [8] etc. A generalised NN framework for tabular data is presented in [9]. Most of the mentioned tasks may be hampered with safety concerns and require reliability in the performance of the NN used for predicting or classifying data. In [10], authors presented a method for generation of imperceptible adversarial attacks for tabular data.

In this paper, we focus on a less empirical way of quantifying the robustness of a NN model by computing its Lipschitz constant. The Lipschitz regularity of a NN model has been investigated in the literature [11, 12] and constitutes a valuable measure of its robustness to small perturbations. We compare three state-of-the art methods [13, 14, 15] to estimate the Lipschitz constant and draw conclusions regarding their effectiveness. It is worth noticing that these methods allow the Lipschitz constant to be computed according to various norms. Using these quantitative tools, we also evaluate the impact of several standard training procedures on the stability of fully connected networks designed for dealing with tabular data. The effect of ℓ_1 and ℓ_2 regularization, dropout, positivity, and the addition of noise on the training samples is thus analyzed in three applicative scenarios.

The rest of the paper is organized as follows. The theoretical background of our work is presented in next section. In Section 3 we present our experimental setup for different data

sets and different techniques, and we conclude this section with the analysis performed based on our various Lipschitz estimation techniques. Some concluding remarks are drawn in the last section.

2. BACKGROUND

2.1. Lipschitz Constant of Feed-forward Networks

Consider the m-layer feedforward neural network T illustrated in Fig 1, having N_i neurons at layer $i \in \{1, \dots, m\}$. Let $x \in \mathbb{R}^{N_0}$ be the input and $T(x) \in \mathbb{R}^{N_m}$ the associated output. A Lipschitz constant of this NN is an upper bound on the ratio between the variations of the outputs and the variations of the inputs of a function T, thus measuring the sensitivity of the function with respect to input perturbations z. This means that θ is a Lipschitz constant of T if

$$\|T(x+z) - T(x)\| \leq \theta \|z\| \tag{1}$$

for every input $x \in \mathbb{R}^{N_0}$ and perturbation $z \in \mathbb{R}^{N_0}$. The smaller this constant, the more robust the network with respect to perturbations.

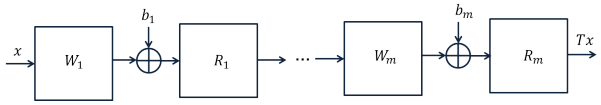

Fig. 1. m-layered feed-forward neural network architecture. For the i^{th}-layer, W_i is the linear weight operator, b_i the bias vector, and R_i the activation operator.

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 [2]: $\overline{\theta}_m = \|W_m\|_{\mathrm{S}} \|W_{m-1}\|_{\mathrm{S}} \cdots \|W_1\|_{\mathrm{S}}$. Although easy to compute, this upper bound turns out be over-pessimistic. In [12], the problem of computing the exact Lipschitz constant of a NN with differentiable activation functions is pointed out to be NP hard. The authors proposed an estimation algorithm for sequential NNs.

In [14], various bounds on the Lipschitz constant of a feed-forward network are derived by assuming that, for every $i \in \{1, \dots, m\}$ the activation operator R_i is α_i-averaged with $\alpha_i \in]0, 1]$.[1] This assumption is satisfied in many NNs since most of the existing activation functions are proximity operators of convex potentials [16], hence $1/2$-averaged. If, for every $i \in \{1, \dots, m-1\}$, R_i is separable, i.e., consists of real-valued functions of one-variable applied to each of the components of its input vector, a general Lipschitz constant estimate (designated as CPLip) reads

$$\vartheta_m = \sup_{\Lambda_1 \in \mathcal{D}_1, \dots, \Lambda_{m-1} \in \mathcal{D}_{m-1}} \|W_m \Lambda_{m-1} \cdots \Lambda_1 W_1\|_{\mathrm{S}}, \tag{2}$$

where, for every $i \in \{1, \dots, m-1\}$, $\mathcal{D}_i$ is the set of diagonal matrices of size $N_i \times N_i$ with diagonal elements equal to $2\alpha_i - 1$ or 1. An interesting result established in [14] is that this estimate remains valid when other norms than the Euclidean norm are used to quantify the perturbations on the input and the output. 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).

2.2. SDP Based Approach

The work in [13] focuses on NNs using separable activation operators. It assumes that the activation function ρ_i used at a layer $i \in \{1, \dots, m\}$ is *slope-bounded*, i.e. there exists nonnegative parameters $\sigma_{\min}$ and $\sigma_{\max}$ such that

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

As mentioned by the authors, most activation functions satisfy this inequality with $\sigma_{\min} = 0$ and $\sigma_{\max} = 1$. But it turns out from [17, Proposition 2.4] that we then recover similar assumptions to those made in [16]. By a clever use of the firm nonexpansiveness properties of the activation operators R_i, the problems can be recast as solving the following semidefinite positive programming (SDP):

$$\underset{(\rho_m, Q_1, \dots, Q_{m-1}) \in C}{\text{minimize}} \quad \rho_m, \tag{3}$$

where $\sqrt{\rho}_m$ with $\rho_m \geq 0$ is the sought Lipschitz constant, $(Q_i)_{1 \leq i \leq m-1} \in [0, +\infty)^{N_i \times N_i}$ are diagonal matrices, and C is a closed convex set defined by the positive semidefiniteness of a certain matrix which is an affine function of $(\rho_m, Q_1, \dots, Q_{m-1})$ (see [13] for details). One limitation of this method is that it is tailored to the use of the Euclidean norm.

Remark 1. *In [13], one of the versions of this method, that will be designated hereafter by LipSDP, uses non diagonal matrices $(Q_i)_{1 \leq i \leq m-1}$. As shown by the counterexample presented in [18], this estimate is however erroneous.*

2.3. Polynomial optimization based approach

The approach in [15] applies to NNs having a single output (i.e. $N_m = 1$).[2] The authors insist that their approach is restricted to differentiable activation functions, but it is actually valid for any separable firmly nonexpansive activation operator. When $N_m = 1$ and the ℓ_p norm with $p \in [1, +\infty]$ is used for the input space, the Lipschitz constant reduces to

$$\vartheta_m = \sup_{\Lambda_1 \in \mathcal{D}_1, \dots, \Lambda_{m-1} \in \mathcal{D}_{m-1}} \|W_1^\top \Lambda_1 \cdots \Lambda_{m-1} W_m^\top\|_{p^*}, \tag{4}$$

[1] This means that there exists an nonexpansive operator Q_i such that $R_i = (1 - \alpha_i) \operatorname{Id} + \alpha_i Q_i$.

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

			ℓ_1					ℓ_2				
	α	0	0.00001	0.0001	0.001	0.01	0.1	0.00001	0.0001	0.001	0.01	0.1
	MAE	0.0069	0.0071	0.0069	0.0094	0.0300	0.0300	0.0073	0.007	0.0077	0.0299	0.030
$L_{2,2}$	LipSDP [13]	0.657	0.705	0.26	0.024	≈ 0	≈ 0	1.01	0.741	0.027	≈ 0	≈ 0
	CPLip[14]	0.638	0.681	0.25	0.024	5.33e-11	3.99e-11	0.96	0.73	0.027	1.26e-09	1.09e-17
$L_{+\infty,+\infty}$	Lipopt-k [15]	1.41	1.39	0.47	0.028	≈ 0	≈ 0	1.89	1.26	0.040	1.07e-09	≈ 0
	CPLip[14]	1.23	1.17	0.46	0.028	9.16e-11	6.75e-11	1.486	1.26	0.040	2.35e-09	1.95e-17

Table 1: Results on Combined Cycle Power Plant Data Set for ℓ_1 and ℓ_2 regularization

			ℓ_1						ℓ_2					
	α	0	0.00001	0.0001	0.001	0.01	0.1	0.2	0.00001	0.0001	0.001	0.01	0.1	0.2
	MAE	0.0418	0.0444	0.0405	0.0443	0.0505	0.1490	0.1490	0.0515	0.0404	0.0424	0.0454	0.1489	0.1489
$L_{2,2}$	LipSDP [13]	2.75	1.74	0.38	0.16	0.11	≈ 0	≈ 0	2.13	0.78	0.201	0.089	≈ 0	≈ 0
	CPLip[14]	2.747	1.705	0.373	0.16	0.110	8.75e-09	7.44e-09	2.11	0.721	0.201	0.089	9.96e-09	9.72e-09
$L_{+\infty,+\infty}$	Lipopt-k [15]	9.47	5.66	1.04	0.286	0.1642	1.66e-08	1.83e-08	7.048	3.088	0.4365	0.2099	1.41e-08	1.45e-08
	CPLip[14]	6.98	4.36	1.03	0.29	0.16	2.30e-08	2.06e-08	4.97	1.93	0.44	0.21	2.9e-08	2.8e-08

Table 2: Results on Auto MPG Data Set for ℓ_1 and ℓ_2 regularization

			ℓ_1						ℓ_2					
	α	0	0.00001	0.0001	0.001	0.01	0.1	0.2	0.00001	0.0001	0.001	0.01	0.1	0.2
	Acc	84.94	85.16	85.57	85.54	76.30	76.30	76.30	85.32	85.26	85.46	84.73	76.32	76.32
$L_{2,2}$	LipSDP[13]	8.21	6.29	5.15	3.45	≈ 0	≈ 0	≈ 0	9.15	5.32	4.19	1.91	≈ 0	≈ 0
	CPLip[14]	8.208	6.29	5.15	3.45	5.81e-10	1.84e-10	3.02e-10	9.15	5.32	4.19	1.91	1.839e-10	6.04e-11
$L_{+\infty,+\infty}$	Lipopt-k [15]	56.22	31.77	20.53	12.36	≈ 0	≈ 0	≈ 0	42.74	22.48	17.90	9.06	2.43e-11	≈ 0
	CPLip[14]	56.22	31.77	20.53	12.36	4.33e-09	1.34e-09	2.21e-09	42.74	22.48	17.90	9.06	9.25e-10	3.85e-10

Table 3: Results on Adult Data Set for ℓ_1 and ℓ_2 regularization

where $p^* \in [1, +\infty]$ is the dual exponent of p (such that $1/p + 1/p^* = 1$). When $p \in \mathbb{N} \setminus \{0, 1\}$ and $p = +\infty$, 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, leading to the so-called LipOpt estimation method. However, the size of the hierarchy tends to grow fast and if its order 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. The approach is further improved in [19].

3. EXPERIMENTS ON TABULAR DATA

3.1. Dataset and Network description

In our stability analysis, we study three widely used tabular datasets from UCI Repository: 1) Combined Cycle Power Plant Data Set has 4 attrributes with 9568 instances , 2) Auto MPG Data Set consists of 398 instances with 7 attributes, 3) Adult Data Set consists of 48842 instances in total with 14 attributes. Dataset 2 and 3 include both continuous and categorical attributes, whereas dataset 1 contains only continuous attributes. 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. All the architectures are made up of two hidden fully connected layers with the following characteristics:

- Combined Cycle Power Plant Data set - $(4, 10, 6, 1)$
- Auto MPG Data set - $(9, 16, 8, 1)$
- Adult Data set - $(88, 6, 6, 1)$

3.2. Effect of Regularization Techniques

We present the results for three regularisation techniques: ℓ_1, ℓ_2, and Dropout [20]. We study the relationship between the parameters associated with each regularisation and the NN stability quantified by its Lipschitz constant. Attention also to be paid to the resulting accuracy. We apply these regularization techniques while training our NNs, then compute a Lipschitz constant associated with the obtained weights. We use the three state-of-the-art estimation methods which have been described in Section 2. The first one is LipSDP [13] which uses Euclidean norms for both the input and output spaces ($L_{2,2}$ spectral norm). The second one is the polynomial based approach LipOpt [19] where the input and output spaces are equipped with the sup norm while estimating the Lipschitz constant. This estimation approach is thus linked to the $L_{+\infty,+\infty}$ subordinate matrix norm. The third estimation method is CPLip [14] which can work for any norm on the input and output spaces.

Each experiment was run 10 times and we chose the model with best performance (least MAE or highest classification accuracy) and computed a Lipschitz constant for this

	Drop-rate	0	0.05	0.1	0.2	0.3	0.4	0.5
	MAE	0.0069	**0.0069**	0.0074	0.0086	0.0115	0.0111	0.016
$L_{2,2}$	LipSDP [13]	0.66	**0.16**	0.23	0.56	0.45	0.27	0.40
	CPLip[14]	0.64	**0.16**	0.23	0.54	0.42	0.27	0.39
$L_{+\infty,+\infty}$	Lipopt-k [15]	1.41	**0.26**	0.32	1.01	0.79	0.42	0.69
	CPLip[14]	1.23	**0.26**	0.32	1.01	0.69	0.42	0.69

Table 4: Results on Combined Cycle Power Plant Data set with Dropout

	Drop-rate	0	0.05	0.1	0.2	0.3	0.4	0.5
	MAE	0.042	0.039	0.0364	0.043	0.044	0.045	0.050
$L_{2,2}$	LipSDP [13]	2.75	1.86	2.1	1.89	2.28	1.88	1.41
	CPLip[14]	2.75	1.73	2.1	1.87	2.28	1.88	1.42
$L_{+\infty,+\infty}$	Lipopt-k [15]	9.47	6.41	5.49	4.89	5.2	3.98	3.07
	CPLip [14]	6.98	4.45	4.631	4.89	5.19	3.98	3.07

Table 5: Results on Auto MPG Data set with Dropout

model. The results for the datasets using ℓ_1 and ℓ_2 regularization are reported in Tables 1, 2, and 3 for varying values of the regularization parameter α which controls the strength of the ℓ_1 and ℓ_2 penalty on the weights. Similar results with varying Droprates are presented in Tables 4, 5, and 6. Droprate corresponds to the proportion of the neurons which will be shut-off while training a neural network.

3.3. Positive Weighted Networks

Next we analyse the stability of NNs when the weights are constrained to be non-negative. The comparison between arbitrary signed network and positively signed network for all the three datasets is shown in Table 7.

3.4. Addition of Noise to the Dataset

We also perform a stability analysis when the original dataset is corrupted with random noise which is a standard practice while training NNs having continuous input variables. From the original dataset, we generated a dataset 20 times larger by including noisy samples. More precisely, to all the normalised input features, we added a noise value drawn from a random i.i.d. zero-mean Gaussian distribution with a small standard deviation. We combined the original training set with the generated noisy samples and trained our model on the augmented dataset. The results on Combined Cycle Power plant with variation of standard deviation are given in Table 8.

3.5. Comments on the results

A first observation is that CPLip provides slightly tighter bounds than LipSDP and LipOpt. The two latter approaches

	Drop-rate	0	0.05	0.1	0.2	0.3	0.4	0.5
	Acc	84.94	85.13	85.14	85.08	85.03	85.09	84.81
$L_{2,2}$	LipSDP [13]	8.21	7.17	7.39	6.82	6.53	6.68	6.63
	CPLip[14]	8.21	7.17	7.39	6.82	6.53	6.68	6.63
$L_{+\infty,+\infty}$	Lipopt-k [15]	56.22	46.07	49.32	43.58	40.21	39.87	41.26
	CPLip[14]	56.22	46.07	49.32	43.58	40.21	39.87	41.26

Table 6: Results on Adult Data Set with Dropout

		Dataset 1		Dataset 2		Dataset 3	
		Arbitrary	Positive	Arbitrary	Positive	Arbitrary	Positive
	MAE/ACC	0.0069	0.021	0.042	0.08	84.94	83.54
$L_{2,2}$	LipSDP [13]	0.66	0.03	2.75	0.57	8.21	3.65
	CPLip [14]	0.64	0.03	2.75	0.57	8.21	3.65
$L_{+\infty,+\infty}{}^3$	Lipopt-k [15]	1.41	0.06	9.47	3.29	56.22	18.81
	CPLip[14]	1.23	0.06	6.98	1.24	56.22	18.81

Table 7: Results with positive constraint on the weights

	std	No Noise	0.01	0.05	0.1	0.2
	MAE	0.0069	0.0064	0.0061	0.0065	0.0071
$L_{2,2}$	LipSDP [13]	0.66	0.61	0.34	0.37	0.10
	CPLip[14]	0.64	0.61	0.33	0.36	0.09
$L_{+\infty,+\infty}$	Lipopt-k [15]	1.411	1.12	0.58	0.61	0.24
	CPLip[14]	1.234	1.12	0.57	0.59	0.16

Table 8: Results on Combined Cycle Power Plant Data set with added noise

may however be more scalable when applied to deeper networks. Another remark is that similar behaviours can be seen when using different norms. While designing a network for deployment, it also appears that there is a trade-off between stability and performance.

- ℓ_1 and ℓ_2 regularizations increase the stability of the network consistently, but the performance is maintained up to a certain value of α, from where the accuracy drops. ℓ_1 usually yields better results, confirming the robustness of this norm as training measure.
- We globally observe an increasing stability of the neural networks as we increase the value of Droprate. The results are however less consistent than with regularization.
- The positive constraint leads to a significant loss of performance in terms of MAE and classification accuracy, but the stability of the networks is improved by a significant margin.
- As expected, adding noise leads to a drop in the value of the Lipschitz constant as the noise level increases, without having a too negative impact on the accuracy.

4. CONCLUSION

This work is a step towards better controlling the robustness of NNs by analysing their Lipschitz properties. Ensuring the safety of new generations of industrial products based on AI constitutes the main objective. We have here showcased the impact of standard training procedures on stability. However, developing more dedicated training procedures [21, 22] to improve robustness could allow us to achieve better accuracy-stability tradeoffs.

5. REFERENCES

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

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

[3] Aditi Raghunathan, Jacob Steinhardt, and Percy Liang, "Certified defenses against adversarial examples," *arXiv preprint arXiv:1801.09344*, 2018.

[4] Eric Wong, Frank Schmidt, Jan Hendrik Metzen, and J Zico Kolter, "Scaling provable adversarial defenses," in *Advances in Neural Information Processing Systems*, 2018, pp. 8400–8409.

[5] Huan Zhang, Tsui-Wei Weng, Pin-Yu Chen, Cho-Jui Hsieh, and Luca Daniel, "Efficient neural network robustness certification with general activation functions," in *Advances in neural information processing systems*, 2018, pp. 4939–4948.

[6] Guy Katz, Clark Barrett, David L Dill, Kyle Julian, and Mykel J Kochenderfer, "Reluplex: An efficient SMT solver for verifying deep neural networks," in *International Conference on Computer Aided Verification*. Springer, 2017, pp. 97–117.

[7] Paul Covington, Jay Adams, and Emre Sargin, "Deep neural networks for youtube recommendations," in *Proceedings of the 10th ACM conference on recommender systems*, 2016, pp. 191–198.

[8] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He, "Deepfm: a factorization-machine based neural network for ctr prediction," *arXiv preprint arXiv:1703.04247*, 2017.

[9] Guolin Ke, Jia Zhang, Zhenhui Xu, Jiang Bian, and Tie-Yan Liu, "Tabnn: A universal neural network solution for tabular data," 2018.

[10] Vincent Ballet, Xavier Renard, Jonathan Aigrain, Thibault Laugel, Pascal Frossard, and Marcin Detyniecki, "Imperceptible adversarial attacks on tabular data," *arXiv preprint arXiv:1911.03274*, 2019.

[11] Peter L Bartlett, Dylan J Foster, and Matus J Telgarsky, "Spectrally-normalized margin bounds for neural networks," in *Advances in Neural Information Processing Systems*, 2017, pp. 6240–6249.

[12] Aladin Virmaux and Kevin Scaman, "Lipschitz regularity of deep neural networks: analysis and efficient estimation," in *Advances in Neural Information Processing Systems*, 2018, pp. 3835–3844.

[13] Mahyar Fazlyab, Alexander Robey, Hamed Hassani, Manfred Morari, and George Pappas, "Efficient and accurate estimation of Lipschitz constants for deep neural networks," in *Advances in Neural Information Processing Systems*, 2019, pp. 11423–11434.

[14] Patrick L Combettes and Jean-Christophe Pesquet, "Lipschitz certificates for neural network structures driven by averaged activation operators," *SIAM Journal on Mathematics of Data Science*, vol. 2, pp. 529–557, 2020.

[15] Fabian Latorre, Paul Rolland, and Volkan Cevher, "Lipschitz constant estimation of neural networks via sparse polynomial optimization," *arXiv preprint arXiv:2004.08688*, 2020.

[16] Patrick L Combettes and Jean-Christophe Pesquet, "Deep neural network structures solving variational inequalities," *Set-Valued and Variational Analysis*, vol. 28, pp. 1–28, 2020.

[17] Patrick L Combettes and Jean-Christophe Pesquet, "Proximal thresholding algorithm for minimization over orthonormal bases," *SIAM Journal on Optimization*, vol. 18, no. 4, pp. 1351–1376, 2008.

[18] Patricia Pauli, Anne Koch, Julian Berberich, and Frank Allgöwer, "Training robust neural networks using lipschitz bounds," *arXiv preprint arXiv:2005.02929*, 2020.

[19] Tong Chen, Jean-Bernard Lasserre, Victor Magron, and Edouard Pauwels, "Polynomial optimization for bounding Lipschitz constants of deep networks," *arXiv preprint arXiv:2002.03657*, 2020.

[20] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[21] Ana Neacsu, Jean-Christophe Pesquet, and Corneliu Burileanu, "Accuracy-robustness trade-off for positively weighted neural networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 8389–8393.

[22] Ana Neacsu, Kavya Gupta, Jean-Christophe Pesquet, and Corneliu Burileanu, "Signal denoising using a new class of robust neural networks," *European Signal Processing Conference (EUSIPCO)*, 2020.