



**HAL**  
open science

## ABBA Neural Networks: Coping with Positivity, Expressivity, and Robustness

Ana-Antonia Neacșu, Jean-Christophe Pesquet, Vlad Vasilescu, Corneliu  
Burileanu

► **To cite this version:**

Ana-Antonia Neacșu, Jean-Christophe Pesquet, Vlad Vasilescu, Corneliu Burileanu. ABBA Neural Networks: Coping with Positivity, Expressivity, and Robustness. SIAM Journal on Mathematics of Data Science, In press. hal-04386260

**HAL Id: hal-04386260**

**<https://hal.science/hal-04386260>**

Submitted on 10 Jan 2024

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

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution - NonCommercial 4.0 International License

# ABBA Neural Networks: Coping with Positivity, Expressivity, and Robustness \*

Ana Neacșu <sup>†</sup>, Jean-Christophe Pesquet <sup>‡</sup>, Vlad Vasilescu <sup>†</sup>, and Corneliu Burileanu <sup>†</sup>

**Abstract.** We introduce ABBA networks, a novel class of (almost) non-negative neural networks, which are shown to possess a series of appealing properties. In particular, we demonstrate that these networks are universal approximators while enjoying the advantages of non-negative weighted networks. We derive tight Lipschitz bounds both in the fully connected and convolutional cases. We propose a strategy for designing ABBA nets that are robust against adversarial attacks, by finely controlling the Lipschitz constant of the network during the training phase. We show that our method outperforms other state-of-the-art defenses against adversarial white-box attackers. Experiments are performed on image classification tasks on four benchmark datasets.

**Key words.** Neural Networks, Robustness, Optimization, Lipschitz, Image Classification

**MSC codes.** 15A48, 68T05, 65K10

**1. Introduction.** Deep learning methods based on neural network models have received increasing attention in the scientific community, because of their stunning abilities to solve a variety of complex tasks. These powerful systems excel at learning intricate mappings and, in some cases, even surpass human performance. However, deep architectures usually lack interpretability and they may lead to over-parameterized models [13, 37]. Additionally, their robustness is not well-controlled, leaving them exposed to potential adversarial attacks. For instance, [49] demonstrated that by introducing carefully-crafted, low-magnitude adversarial perturbations, neural classifiers could be easily fooled [17]. A way of overcoming the aforementioned challenges consists in introducing some specific constraints in the neural network design. In this article, we are interested in nonnegativity and stability constraints on the network weights.

It is widely accepted that humans possess the innate ability to decompose complex interactions into discrete, intuitive hierarchical categories before analyzing them [26]. Conceptually, this evolution towards part-based representation in human cognition can be linked to non-negativity restrictions on the network weights [7]. This idea, along with other factors, has sparked interest in neural networks with non-negative weights. These networks have drawn attention for several reasons. Firstly, they align with human understandability, making them more interpretable. Secondly, the non-negativity constraint can act as beneficial regularization, effectively reducing overfitting issues. Moreover, recent studies have demonstrated that it is possible to derive a tight Lipschitz bound for such networks. This Lipschitz constant serves as a valuable metric for quantifying the robustness of the network, enabling us to design networks with enhanced resilience to adversarial perturbations during the training process. Despite their advantages, one significant drawback of networks with non-negative weights is that they might be less expressive than networks with arbitrary signed weights. In [54], it is shown that standard non-negative networks are not universal approximators [54], a limitation that

---

\*Part of this work was supported by the French ANR Research and Teaching Chair in Artificial Intelligence BRIDGEABLE.

<sup>†</sup>Speech and Dialogue Laboratory, University Politehnica of Bucharest, Romania ([ana\\_antonina.neacsu@upb.ro](mailto:ana_antonina.neacsu@upb.ro)).

<sup>‡</sup>Centre de Vision Numérique, Inria, CentraleSupélec, Gif-sur-Yvette, France.

37 our work overcomes.

38 **Approach.** In this work, we are interested in neural networks having non-negative weights,  
 39 except for the first and last linear layers. This class of networks obviously constitutes an extension of  
 40 those having all their linear layers non-negative-valued. We focus on a particular subclass of these  
 41 networks for which the weight matrices have a structure of the form

$$42 \begin{bmatrix} A & B \\ B & A \end{bmatrix},$$

43 thus enjoying a number of algebraic properties. The corresponding networks are subsequently called  
 44 ABBA networks. Note that weight matrices  $A$  and  $B$  are duplicated in ABBA networks, thus allowing  
 45 us to limit the number of parameters.

46 **Contributions.** This paper makes several key contributions, which are as follows:

- 47 • We show that we can put any arbitrary signed network in an ABBA form. This property holds  
 48 for fully connected as well as for convolutional neural networks.
- 49 • Universal approximation theorems are derived for networks featuring non-negatively weigh-  
 50 ted layers.
- 51 • We present a method for effectively controlling the Lipschitz constant of ABBA networks<sup>1</sup>. The  
 52 resulting training strategy applies to both fully connected and convolutional cases. Precise  
 53 Lipschitz bounds are typically NP hard to compute for arbitrary signed networks, but our  
 54 framework allows us to derive such bounds that are easy to compute.
- 55 • Numerical experiments conducted on standard image datasets showcase the excellent perfor-  
 56 mance of ABBA networks for small models. Notably, they exhibit substantial improvements  
 57 in both performance and robustness compared to networks with exclusively non-negative  
 58 weights. Moreover, we demonstrate that ABBA networks are competitive with robust networks  
 59 featuring arbitrarily signed weights, trained using state-of-the-art techniques.

60 **Outline.** The rest of the paper is organized as follows. Section 2 offers an overview of the related  
 61 literature, while in Section 3 our main contributions concerning ABBA architectures are introduced,  
 62 alongside a list of fundamental properties. Section 4 extends our results to the case of convolutional  
 63 neural networks and two Lipschitz constant expressions are derived. Section 5 describes the training  
 64 strategy we employ to generate robust models with respect to adversarial perturbations. Section 6  
 65 details the results obtained for different image classification tasks, while Section 7 is dedicated to  
 66 concluding remarks.

67 **2. Related work. Non-negative neural networks.** Inspired by non-negative matrix factorization  
 68 (NMF) techniques, the work of [7] introduces non-negative restrictions on the weights to create neural  
 69 networks in which the hidden units correspond to identifiable concepts. [4] showed that Autoencoders  
 70 (AE) trained under non-negativity constraints are able to derive meaningful representations that  
 71 unearth the hidden structure of high-dimensional data. Their method showed promising results from  
 72 both performance and feature interpretation viewpoints on four different classification tasks. [12]  
 73 presented the first polynomial-time algorithm for Probably Approximately Correct (PAC) learning 1-  
 74 layer neural networks with positive coefficients. Moreover, ensuring non-negativity has been shown to  
 75 have a regularization effect, reducing feature overfitting, which is a very common problem, especially

---

<sup>1</sup>A full TensorFlow implementation of our framework can be provided on demand and will be made publicly available upon paper acceptance.

76 for tasks where the available training data is scarce [35]. Neural networks defining convex functions  
77 of their inputs [1] also constitute a subclass of networks with non-negative weights.

78 **Link with other networks.** From another perspective, the idea of using redundant weights is  
79 reminiscent of siamese networks [6]. These architectures are successfully used to handle similarity  
80 learning tasks, such as face verification [50], character recognition [23], and object tracking [19].  
81 Siamese networks compute a similarity metric on the representations of the inputs, after applying the  
82 same transformation to each one. Apart from the proven efficiency on solving computer vision tasks,  
83 they have lately been employed in NLP problems, e.g., computational argumentation. In [14], it is  
84 shown that siamese architectures outperform other baselines trained on convincingness datasets.

85 **Robustness.** The robustness of neural networks against possible adversarial attacks is a topic that  
86 has received increasing attention since nowadays AI-based solutions are ubiquitous [3, 36]. A sizable  
87 body of literature on adversarial attacks and different defense strategies have emerged in recent  
88 years as a result of the work in [49]), which revealed the alluring susceptibility of neural networks to  
89 adversarial perturbations and proposed a box-constrained L-BFGS algorithm for finding adversarial  
90 examples. [15] introduced the FGSM attack as a one-step modification of the input image, following  
91 the direction of loss maximization, while [24] incorporated this step into an iterative method known  
92 as PGD, seen as an improvement over basic FGSM. DeepFool [33] iteratively searches for the closest  
93 adversarial point that directs the optimization towards crossing the decision boundary. DDN [42] and  
94 FMN [39] attacks fall into the category of projected-gradient methods, using iterative updates of the  
95 perturbation vector towards the minimization of its magnitude.

96 Defensive strategies have been developed to alleviate this robustness issue. [47] divides adver-  
97 sarial defense methods into three categories: adversarial detection, gradient masking, and robust  
98 optimization. Adversarial Training (AT) was first introduced by [15] and later improved by [32].  
99 Recent works on AT [48, 55] have successfully analyzed and refined training techniques, however,  
100 no theoretical certificates regarding their behavior in the presence of different adversaries have  
101 been established yet. Regularization-based methods, such as [30, 46, 59], include additional terms  
102 in their objective, steering the learning process in a direction that leads to better generalization.  
103 [40] provides robustness certificates for neural networks with one hidden layer, yielding an upper  
104 bound of the error in the presence of any adversary (see [11, 16, 41] for more advanced methods.  
105 Randomized smoothing [9, 25, 29, 44, 57] certifies the robustness of a classifier around an input  
106 point by measuring the most-likely prediction over Gaussian-corrupted versions of the point.

107 **Lipschitz properties of neural networks.** As highlighted by [49], the Lipschitz behavior of a  
108 neural model is closely correlated with its robustness against adversarial attacks, providing an upper  
109 bound on the response given input perturbations. Controlling the Lipschitz behavior of the network  
110 thus offers theoretical stability guarantees. However, computing the exact constant, even for small  
111 networks is an NP-hard problem [20], and finding a good approximation in a reasonable time is  
112 an open challenge. Several solutions have been proposed lately (see for example: [5, 18, 21, 58]).  
113 [45] introduced *deel-lip*, a framework to control the Lipschitz constant of each layer individually,  
114 while [2] propose GroupSort networks to ensure robustness. [38] proposed a framework for training  
115 fully-connected neural networks using Lipschitz regularized and constrained techniques, proving their  
116 effectiveness in the scenario of Gaussian-added perturbation noise. A recent result in [10] showed  
117 that in the case of models with non-negative weights a tight Lipschitz bound can be established,  
118 making possible the training of neural network models with certified robustness guarantees.

### 119 3. ABBA Neural Networks.

120 **3.1. Problem formulation.** In the remainder of this paper,  $\|\cdot\|$  will denote the  $\ell_2$ -norm when  
121 dealing with a vector, and the spectral norm when dealing with a matrix.

122 An  $m$ -layer feedforward neural network can be described by the following model.

123 **Model 3.1.**  $T$  is feedforward neural network if there exists  $(N_i)_{1 \leq i \leq m} \in (\mathbb{N} \setminus \{0\})^m$  such that

$$124 \quad (3.1) \quad T = T_m \circ \dots \circ T_1$$

125 where, for every layer index  $i \in \{1, \dots, m\}$ ,  $T_i = R_i(W_i \cdot + b_i)$ ,  $W_i \in \mathbb{R}^{N_i \times N_{i-1}}$  is the weight matrix,  
126  $b_i \in \mathbb{R}^{N_i}$  the bias vector, and  $R_i: \mathbb{R}^{N_i} \rightarrow \mathbb{R}^{N_i}$  the activation operator.  $N_i$  corresponds to the number  
127 of inputs at the  $i$ -th layer. Such a layer is convolutive if it corresponds to a weight matrix  $W_i$  having  
128 some Toeplitz (or block Toeplitz) structure.

129 We will say that the activation operator  $R_i$  is symmetric, if there exists  $(c_i, d_i) \in (\mathbb{R}^{N_i})^2$  such that

$$130 \quad (3.2) \quad (\forall x \in \mathbb{R}^{N_i}) \quad R_i(x) - d_i = -R_i(-x + c_i).$$

131 In other words,  $(c_i, d_i)/2$  is a symmetry center of the graph of  $R_i$ .

132 For example, if  $R_i$  is squashing function used in CapsNets [43], it is such that

$$133 \quad (3.3) \quad (\forall x \in \mathbb{R}^{N_i}) \quad R_i(x) = \frac{\mu \|x\|}{1 + \|x\|^2} x.$$

134 with  $\mu = 8/(3\sqrt{3})$ . It thus satisfies the symmetry property (3.2) with  $c_i = d_i = 0$ . In addition,  $R_i$   
135 is nonexpansive, i.e. it has a Lipschitz constant equal to 1 [10] Other examples of symmetric and  
136 nonexpansive activation operators are presented in Appendix SM1<sup>2</sup>.

137

138 **3.2. ABBA Matrices.** We first define ABBA matrices which will be the main algebraic tool  
139 throughout this article.

140 **Definition 3.2.** Let  $(N_1, N_2) \in (\mathbb{N} \setminus \{0\})^2$ .  $\mathcal{A}_{N_1, N_2}$  is the set of ABBA matrices of size  $(2N_2) \times (2N_1)$ ,  
141 that is  $M \in \mathcal{A}_{N_1, N_2}$  if there exist matrices  $A \in \mathbb{R}^{N_2 \times N_1}$  and  $B \in \mathbb{R}^{N_2 \times N_1}$  such that

$$142 \quad (3.4) \quad M = \begin{bmatrix} A & B \\ B & A \end{bmatrix}.$$

143 The sum matrix associated with  $M$  is then defined as  $\mathfrak{S}(M) = A + B$ .

144 We give some of the most relevant properties of these matrices. In particular, we will see that the  
145 ABBA structure is stable under standard matrix operations.

146 **Proposition 3.3.** Let  $(N_1, N_2, N_3) \in (\mathbb{N} \setminus \{0\})^3$ .

147 (i). If  $M \in \mathcal{A}_{N_2, N_1}$ , then its transpose  $M^\top \in \mathcal{A}_{N_1, N_2}$  and  $\mathfrak{S}(M^\top) = \mathfrak{S}(M)^\top$ .

148 (ii). If  $(M_1, M_2) \in (\mathcal{A}_{N_2, N_1})^2$ , then  $M_1 + M_2 \in \mathcal{A}_{N_2, N_1}$  and  $\mathfrak{S}(M_1 + M_2) = \mathfrak{S}(M_1) + \mathfrak{S}(M_2)$ .

149 (iii). If  $M_1 \in \mathcal{A}_{N_2, N_1}$  and  $M_2 \in \mathcal{A}_{N_3, N_2}$ , then  $M_2 M_1 \in \mathcal{A}_{N_3, N_1}$  and

$$150 \quad \mathfrak{S}(M_2 M_1) = \mathfrak{S}(M_2) \mathfrak{S}(M_1).$$

---

<sup>2</sup>Appendices with number of the form SMx can be found in the supplementary materials.

- 151 (iv).  $\mathcal{A}_{N_1, N_1}$  is a ring when equipped with the standard matrix addition and product.
- 152 (v). If  $A$  and  $B$  are two square matrices of the same size, the eigenvalues of  $\begin{bmatrix} A & B \\ B & A \end{bmatrix}$  are those of
- 153  $A + B$  and  $A - B$ .
- 154 (vi). If  $A$  and  $B$  are two matrices having the same dimensions, the spectral norm of  $\begin{bmatrix} A & B \\ B & A \end{bmatrix}$  is equal
- 155 to  $\max\{\|A + B\|, \|A - B\|\}$ .
- 156 (vii). If  $M \in \mathcal{A}_{N_2, N_1}$  has non-negative elements, the spectral norm of  $M$  is  $\|\mathfrak{S}(M)\|$ .
- 157 (viii). Let  $A \in \mathbb{R}^{N_2 \times N_1}$  and  $B \in \mathbb{R}^{N_2 \times N_1}$ , and let  $K = \min\{N_1, N_2\}$ . Let  $(\lambda_k)_{1 \leq k \leq K}$  (resp.  $(\mu_k)_{1 \leq k \leq K}$ )
- 158 be the singular values of  $A + B$  (resp.  $A - B$ ) and let  $\{u_k\}_{1 \leq k \leq K}$  /  $\{v_k\}_{1 \leq k \leq K}$  (resp.  $\{t_k\}_{1 \leq k \leq K}$
- 159 /  $\{w_k\}_{1 \leq k \leq K}$ ) be associated orthonormal families of left/right singular vectors in  $\mathbb{R}^{N_2}$  /  $\mathbb{R}^{N_1}$ .<sup>3</sup>
- 160 Then, the singular values of  $\begin{bmatrix} A & B \\ B & A \end{bmatrix}$  are  $(\lambda_k, \mu_k)_{1 \leq k \leq K}$  and associated orthonormal families of
- 161 left/right singular vectors are

162 
$$\left\{ \frac{1}{\sqrt{2}} \begin{bmatrix} u_k \\ u_k \end{bmatrix}, \frac{1}{\sqrt{2}} \begin{bmatrix} t_k \\ -t_k \end{bmatrix} \right\}_{1 \leq k \leq K} / \left\{ \frac{1}{\sqrt{2}} \begin{bmatrix} v_k \\ v_k \end{bmatrix}, \frac{1}{\sqrt{2}} \begin{bmatrix} w_k \\ -w_k \end{bmatrix} \right\}_{1 \leq k \leq K}.$$

- 163 (ix). If  $A$  and  $B$  are two matrices having the same dimensions,

164 (3.5) 
$$\text{rank} \left( \begin{bmatrix} A & B \\ B & A \end{bmatrix} \right) = \text{rank}(A + B) + \text{rank}(A - B).$$

- 165 (x). Let  $f$  be a function from  $\mathbb{R}^{(2N_2) \times (2N_1)}$  to  $\mathbb{R}^{(2N_2) \times (2N_1)}$ . Assume that either  $f$  operates elementwise
- 166 or it is a spectral function in the sense that there exists a function  $\varphi: \mathbb{R}_+ \rightarrow \mathbb{R}_+$  such that

167 (3.6) 
$$(\forall M \in \mathbb{R}^{(2N_2) \times (2N_1)}) \quad f(M) = \sum_{k=1}^{2K} \varphi(\tilde{\lambda}_k) \tilde{u}_k \tilde{v}_k^\top$$

168 where  $K = \min\{N_1, N_2\}$ ,  $(\tilde{\lambda}_k)_{1 \leq k \leq 2K}$  are the singular values of  $M$ , and  $\{\tilde{u}_k\}_{1 \leq k \leq 2K}$  /

169  $\{\tilde{v}_k\}_{1 \leq k \leq 2K}$  are associated orthonormal families of left / right singular vectors in  $\mathbb{R}^{2N_2}$  /

170  $\mathbb{R}^{2N_1}$ . Then  $f$  maps any matrix in  $\mathcal{A}_{N_2, N_1}$  to a matrix in  $\mathcal{A}_{N_2, N_1}$ .

- 171 (xi). The best approximation of maximum rank  $R < \min\{N_1, N_2\}$  (in the sense of the Frobenius
- 172 norm) to a matrix in  $\mathcal{A}_{N_2, N_1}$  belongs to  $\mathcal{A}_{N_2, N_1}$ .
- 173 (xii). The projection onto the spectral ball of center 0 and radius  $\rho \in ]0, +\infty[$  of an ABBA matrix is an
- 174 ABBA matrix.

175 The proofs of these properties are provided in Appendix SM2.

176 **3.3. Extension to feedforward networks.** We will now extend the previous algebraic concepts

177 by introducing the class of ABBA feedforward neural networks. In the following, the structure of an

178 ABBA fully-connected network will be presented from the perspective of investigating its links with

179 standard networks. Such networks make use of weights that respect the structure of ABBA matrices,

180 except for the first and the last layers. More precisely, the first layer maps the input to a twice-higher

181 dimensional space, while the last layer performs a dimension reduction by a factor of 2.

<sup>3</sup>This means that (SM2.11) and (SM2.12) hold.

182 **Definition 3.4.** Let  $m \in \mathbb{N} \setminus \{0\}$ .  $\tilde{T}$  is an  $m$ -layer ABBA network if

$$183 \quad (3.7) \quad \tilde{T} = (\tilde{W}_{m+1} \cdot + \tilde{b}_{m+1}) \tilde{T}_m \cdots \tilde{T}_1 \tilde{W}_0$$

184 with  $\tilde{W}_0 \in \mathbb{R}^{(2N_0) \times N_0}$ ,  $\tilde{W}_{m+1} \in \mathbb{R}^{N_m \times (2N_m)}$ ,  $\tilde{b}_{m+1} \in \mathbb{R}^{N_m}$ , and

$$185 \quad (3.8) \quad (\forall i \in \{1, \dots, m\}) \quad \tilde{T}_i = \tilde{R}_i(\tilde{W}_i \cdot + \tilde{b}_i)$$

$$186 \quad (3.9) \quad \tilde{R}_i: \mathbb{R}^{2N_i} \rightarrow \mathbb{R}^{2N_i},$$

$$187 \quad (3.10) \quad \tilde{b}_i \in \mathbb{R}^{2N_i},$$

$$188 \quad (3.11) \quad \tilde{W}_i \in \mathcal{A}_{N_i, N_{i-1}},$$

190 for given positive integers  $(N_i)_{0 \leq i \leq m}$ .  $\tilde{T}$  is an  $m$ -layer non-negative ABBA network if it is an  $m$ -layer  
191 ABBA network as defined above and, for every  $i \in \{1, \dots, m\}$ , the elements of  $\tilde{W}_i$  are non-negative.

192 In the remainder of this paper,  $\mathcal{N}_{m, \mathcal{A}}$  will designate the class of  $m$ -layer ABBA networks and  $\mathcal{N}_{m, \mathcal{A}}^+$   
193 will designate the subclass of  $m$ -layer non-negative ABBA networks. This latter subclass will be the  
194 main topic of investigation in this work. We will also use the notation  $\mathcal{N}_{m, \mathcal{A}}^+(\rho)$  to designate the set of  
195 neural networks in  $\mathcal{N}_{m, \mathcal{A}}^+$  where all the activation operators operate componentwise using the same  
196 function  $\rho: \mathbb{R} \rightarrow \mathbb{R}$ .

197 **3.4. Link with standard neural networks.** In this section, we show that we can reshape Model 3.1  
198 as a special case of a non-negative ABBA network. At each layer  $i \in \{1, \dots, m\}$  of this model, let  $W_i^+ =$   
199  $(W_{i, k, \ell}^+)_{1 \leq k \leq N_i, 1 \leq \ell \leq N_{i-1}} \in [0, +\infty[^{N_i \times N_{i-1}}$  be the positive part of matrix  $W_i = (W_{i, k, \ell})_{1 \leq k \leq N_i, 1 \leq \ell \leq N_{i-1}}$ ,  
200 i.e.

$$201 \quad (3.12) \quad (\forall k \in \{1, \dots, N_i\})(\forall \ell \in \{1, \dots, N_{i-1}\}) \quad W_{i, k, \ell}^+ = \begin{cases} W_{i, k, \ell} & \text{if } W_{i, k, \ell} > 0 \\ 0 & \text{otherwise.} \end{cases}$$

202 Let  $W_i^- = W_i^+ - W_i \in [0, +\infty[^{N_i \times N_{i-1}}$  be the negative part of  $W_i$ , where all the positive elements  
203 of  $W_i$  have been discarded. Let us now define a non-negative ABBA neural network by using these  
204 quantities.

205 **Definition 3.5.** Let  $m \in \mathbb{N} \setminus \{0\}$ . Let  $T$  be the feedforward neural defined in Model 3.1.  $\tilde{T}$  is a network  
206 in  $\mathcal{N}_{m, \mathcal{A}}^+$  **associated with**  $T$  if it satisfies relations (3.7)-(3.11) with

$$207 \quad (3.13) \quad \tilde{W}_0 = \begin{bmatrix} I_{N_0} \\ -I_{N_0} \end{bmatrix}, \quad \tilde{W}_{m+1} = \frac{1}{2}[I_{N_m} \quad -I_{N_m}],$$

208 and

$$209 \quad (3.14) \quad (\forall i \in \{1, \dots, m\}) \quad \tilde{R}_i: \begin{bmatrix} x \\ z \end{bmatrix} \mapsto \begin{bmatrix} R_i(x) \\ R_i(z) \end{bmatrix},$$

$$210 \quad (3.15) \quad \tilde{W}_i = \begin{bmatrix} W_i^+ & W_i^- \\ W_i^- & W_i^+ \end{bmatrix}.$$

212 Note that a weight parametrization similar to (3.15) was used in [56] for computing lower and upper  
213 bounds on the output of a deep equilibrium layer, but in this article  $W_i^-$  has negative values.

214 As we will show next, the main result is that, if the activation functions are symmetric, network  $\tilde{T}$   
215 defined above is identical to network  $T$  in terms of input-output relation, for judicious choices of the  
216 biases of  $\tilde{T}$ .

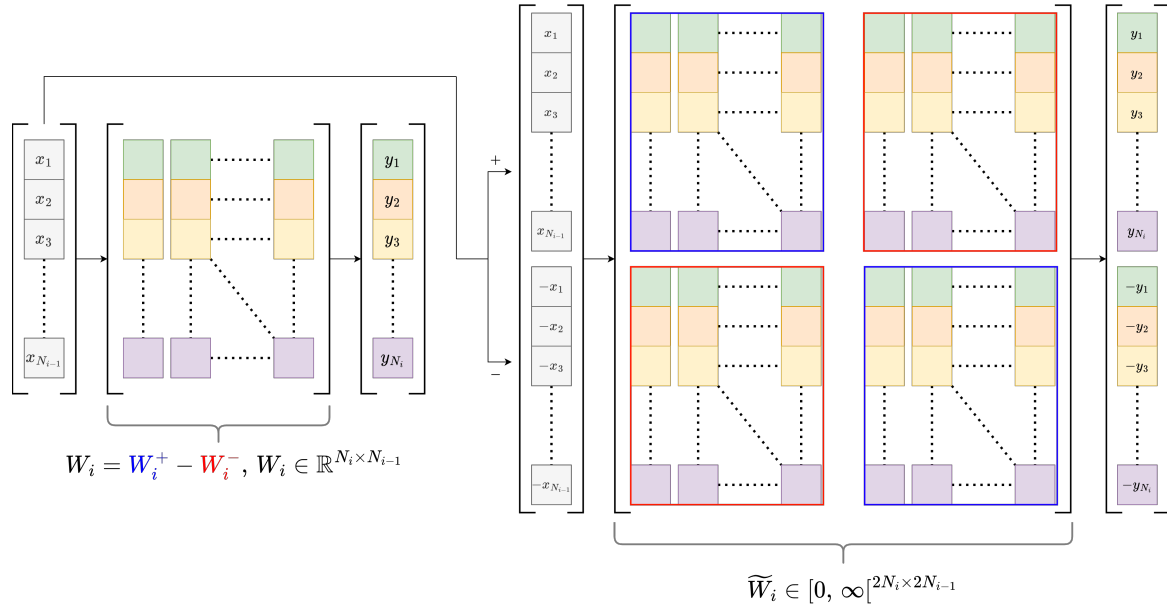


Figure 1: Equivalence between a standard fully-connected layer and its ABBA correspondent.

217 **Proposition 3.6.** Let  $T$  be the  $m$ -layer feedforward network in Model 3.1. Assume that, for every  
 218  $i \in \{1, \dots, m\}$ , the activation operator  $R_i$  in the  $i$ -th layer of  $T$  satisfies the symmetry relation (3.2)  
 219 where  $c_i \in \mathbb{R}^{N_i}$  and  $d_i \in \mathbb{R}^{N_i}$ . Let  $\tilde{T}$  be the neural network of  $\mathcal{N}_{m, \mathcal{A}}^+$  associated with  $T$  whose bias vectors  
 220  $(\tilde{b}_i)_{1 \leq i \leq m}$  are linked to those  $(b_i)_{1 \leq i \leq m}$  of  $T$  by the relations

$$221 \quad (3.16) \quad (\forall i \in \{1, \dots, m\}) \quad \tilde{b}_i = \begin{bmatrix} b_i - W_i^- d_{i-1} \\ c_i - b_i - W_i^+ d_{i-1} \end{bmatrix},$$

$$222 \quad (3.17) \quad \tilde{b}_{m+1} = -\frac{d_m}{2},$$

224 with  $d_0 = 0$ . Then, for every input,  $\tilde{T}$  delivers the same output as  $T$ .

225 The proof of this proposition is provided in Appendix A. An illustration of the link between  
 226 fully-connected layers and ABBA matrices is shown in Figure 1.

227 **3.5. Expressivity of non-negative ABBA networks.** One of the main advantages of non-negative  
 228 ABBA networks with respect to standard networks with non-negative weights is that they are universal  
 229 approximators. More specifically, we have the following result.

230 **Proposition 3.7.** Let  $(n_e, n_r) \in (\mathbb{N} \setminus \{0\})^2$ . Let  $f: \mathbb{R}^{n_e} \rightarrow \mathbb{R}^{n_r}$  be a continuous function. Let  $\mathbb{K}$  be any  
 231 nonempty compact subset of  $\mathbb{R}^{n_e}$  and let  $\epsilon \in ]0, +\infty[$ .

232 (i). Let  $\rho: \mathbb{R} \rightarrow \mathbb{R}$  be a symmetric non polynomial activation function. There exists a network  
 233  $\tilde{T} \in \mathcal{N}_{1, \mathcal{A}}^+(\rho)$  with  $N_0 = n_e$  inputs and  $N_2 = n_r$  outputs such that

$$234 \quad (3.18) \quad (\forall x \in \mathbb{K}) \quad \|\tilde{T}(x) - f(x)\| < \epsilon.$$



235 (ii). Let  $\rho: \mathbb{R} \rightarrow \mathbb{R}$  be a symmetric continuous activation function that is continuously differentiable  
 236 around at least one point where its derivative is nonzero. Then there exists  $m \geq 3$  and  
 237  $\tilde{T} \in \mathcal{N}_{m,\mathcal{A}}^+(\rho)$  with  $N_0 = n_e$  inputs,  $N_{m+1} = n_r$  outputs, and  $2N_i = 2(n_e + n_r + 2)$  neurons in  
 238 every layer  $i \in \{1, \dots, m\}$  such that (3.18) holds.

239 *Proof.* Proposition 3.6 shows that non-negative ABBA networks can be as expressive as signed  
 240 networks. Combining this fact with existing universal approximation results for signed networks (see  
 241 [28] for (i) and [22] for (ii)) allows us to deduce these results. ■

242 (i) addresses the case of shallow wide networks where the number of neurons in the hidden layer  
 243 can be arbitrarily large, while (ii) corresponds to the case of deep networks having a limited number  
 244 of neurons per layer. An illustration of these results is provided in Appendix SM7.

245 **3.6. Lipschitz bounds for ABBA fully-connected networks.** As mentioned in the previous sections,  
 246 the robustness of neural networks with respect to adversarial perturbations can be evaluated through  
 247 their Lipschitz constant. However, most of the existing techniques for computing a tight estimate  
 248 of the constant have a high computational complexity for deep or wide networks, whereas simpler  
 249 upper bounds may turn out to be over-pessimistic.

250 Nevertheless, in the context of non-negative weighted neural networks [10] proved that tight  
 251 approximations to the Lipschitz constant can be achieved. In the following, we extend this result and  
 252 show that we can derive a simple expression for the Lipschitz constant, using a separable bound, for  
 253 non-negative ABBA networks.

254 **Proposition 3.8.** Let  $m \in \mathbb{N} \setminus \{0\}$  and let  $\tilde{T} \in \mathcal{N}_{m,\mathcal{A}}^+$  be given by (3.7)-(3.11). Assume that, for every  
 255  $i \in \{1, \dots, m-1\}$ ,  $\tilde{R}_i$  is a nonexpansive operator operating componentwise. A Lipschitz constant of  $\tilde{T}$  is

$$256 \quad (3.19) \quad \theta_m = \|\tilde{W}_{m+1}\| \|\mathfrak{S}(\tilde{W}_m) \cdots \mathfrak{S}(\tilde{W}_1)\| \|\tilde{W}_0\|.$$

257 The proof of this result is detailed in Appendix B. Note that this bound expression could be easily  
 258 extended to other norms based on the results in [10].

259 A standard separable upper bound for the Lipschitz constant [49] for the ABBA network  $\tilde{T}$   
 260 considered in the previous proposition is

$$261 \quad (3.20) \quad \bar{\theta}_m = \|\tilde{W}_{m+1}\| \|\tilde{W}_m\| \cdots \|\tilde{W}_1\| \|\tilde{W}_0\|.$$

262 According to Proposition 3.3(vii), this bound reads also

$$263 \quad (3.21) \quad \bar{\theta}_m = \|\tilde{W}_{m+1}\| \|\mathfrak{S}(\tilde{W}_m)\| \cdots \|\mathfrak{S}(\tilde{W}_1)\| \|\tilde{W}_0\|,$$

264 which, by simple norm inequalities, is looser than  $\theta_m$ .

265 If  $T$  is the feedforward network defined in Model 3.1 and we apply Proposition 3.8 to the  
 266 associated non-negative ABBA network  $\tilde{T}$  of Definition 3.5. We have

$$267 \quad (3.22) \quad \|\tilde{W}_0\| = \|\tilde{W}_0^\top \tilde{W}_0\|^{1/2} = \|2I_{N_0}\|^{1/2} = \sqrt{2}$$

268 and

$$269 \quad (3.23) \quad \|\tilde{W}_{m+1}\| = \|\tilde{W}_{m+1} \tilde{W}_{m+1}^\top\|^{1/2} = \frac{1}{\sqrt{2}}.$$

270 In turn, for every  $i \in \{1, \dots, m\}$ ,

$$271 \quad (3.24) \quad \mathfrak{S}(\widetilde{W}_i) = W_i^+ + W_i^- = |W_i|.$$

272 where  $|W_i|$  is the matrix whose elements are the absolute values of  $W_i$ . Hence the Lipschitz constant  
273 of  $\widetilde{T}$  in (3.19) reduces to

$$274 \quad (3.25) \quad \theta_m = \|||W_m| \dots |W_1|\|.$$

275 It then follows from Proposition 3.6 that  $\theta_m$  is also a Lipschitz constant of  $T$  when using symmetric  
276 activation functions. Note that this bound was actually already derived in [10, Proposition 5.12].

277 **4. Convolutional networks.** We will now extend the results presented in Section 3 to convolutional  
278 layers.

279 **4.1. ABBA convolutional layers.** For any  $i \in \{1, \dots, m\}$ ,  $\mathcal{W}_i$  is a convolutional layer with  
280  $\zeta_{i-1} \in \mathbb{N} \setminus \{0\}$  input channels,  $\zeta_i$  output channels, kernels  $(w_{i,q,p})_{1 \leq p \leq \zeta_{i-1}, 1 \leq q \leq \zeta_i}$ , and stride  $s_i \in \mathbb{N} \setminus \{0\}$ .  
281 The output  $(y_q)_{1 \leq q \leq \zeta_i}$  of this layer (prior applying any activation operation) is linked to its input  
282  $(x_p)_{1 \leq p \leq \zeta_{i-1}}$  by

$$283 \quad (4.1) \quad (\forall q \in \{1, \dots, \zeta_i\}) \quad u_q = \sum_{p=1}^{\zeta_{i-1}} w_{i,q,p} * x_p$$

$$284 \quad y_q = (u_q)_{\downarrow s_i}.$$

286 Hereabove, for every  $p \in \{1, \dots, \zeta_{i-1}\}$ ,  $x_p = (x_p(\mathbf{n}))_{\mathbf{n} \in \mathbb{Z}^d}$  designates a  $d$ -dimensional discrete signal.  
287 Dimension  $d = 1$  corresponds to 1D signals and  $d = 2$  to images. A similar notation is used for other  
288 signals, in particular  $u_q$  and  $w_{i,q,p}$  with  $q \in \{1, \dots, \zeta_i\}$ . The  $d$ -dimensional discrete convolution is  
289 denoted by  $*$  and  $(\cdot)_{\downarrow s_i}$  is the decimation (or subsampling) by a factor  $s_i$ .

290 The ABBA convolutional layer  $\widetilde{W}_i$  associated with  $\mathcal{W}_i$  has twice the number of input channels  
291 and twice the number of output ones. More specifically, its input consists of  $\zeta_{i-1}$  signals  $(\widetilde{x}_p^+)_{1 \leq p \leq \zeta_{i-1}}$   
292 and  $\zeta_{i-1}$  signals  $(\widetilde{x}_p^-)_{1 \leq p \leq \zeta_{i-1}}$ . Similarly, its output consists of  $\zeta_i$  signals  $(\widetilde{y}_q^+)_{1 \leq q \leq \zeta_i}$  and  $\zeta_i$  signals  
293  $(\widetilde{y}_q^-)_{1 \leq q \leq \zeta_i}$ . To make the input-output relations more explicit, let us define the kernels  $w_{i,q,p}^+$  and  $w_{i,q,p}^-$   
294 analogously to the fully connected case:

$$295 \quad (\forall \mathbf{n} \in \mathbb{Z}^d) \quad w_{i,q,p}^+(\mathbf{n}) = \begin{cases} w_{i,p,q}(\mathbf{n}) & \text{if } w_{i,p,q}(\mathbf{n}) > 0 \\ 0 & \text{otherwise,} \end{cases}$$

$$296 \quad (4.2) \quad w_{i,q,p}^-(\mathbf{n}) = w_{i,q,p}^+(\mathbf{n}) - w_{i,p,q}(\mathbf{n}).$$

298 Then the outputs of the ABBA layer are linked to its inputs by the following relations

$$299 \quad (\forall q \in \{1, \dots, \zeta_i\}) \quad \widetilde{u}_q^+ = \sum_{p=1}^{\zeta_{i-1}} w_{i,q,p}^+ * \widetilde{x}_p^+ + \sum_{p=1}^{\zeta_{i-1}} w_{i,q,p}^- * \widetilde{x}_p^-$$

$$300 \quad (4.3) \quad \widetilde{u}_q^- = \sum_{p=1}^{\zeta_{i-1}} w_{i,q,p}^- * \widetilde{x}_p^+ + \sum_{p=1}^{\zeta_{i-1}} w_{i,q,p}^+ * \widetilde{x}_p^-$$

$$301 \quad \widetilde{y}_q^+ = (\widetilde{u}_q^+)_{\downarrow s_i}$$

$$302 \quad \widetilde{y}_q^- = (\widetilde{u}_q^-)_{\downarrow s_i}.$$

304 The above equations provide the general form of a convolutional ABBA layer when relaxing (4.2).

305 An alternative formulation of convolutional layers in a matrix form, along with its correspondent  
306  $d$ -dimensional spectral representation, is possible (see Appendix SM3). This basically amounts to  
307 characterize layer (4.1) by the following matrices

$$308 \quad (4.4) \quad (\forall \mathbf{n} \in \mathbb{Z}^d) \quad \mathbf{W}_i(\mathbf{n}) = \begin{bmatrix} w_{i,1,1}(\mathbf{n}) & \dots & w_{i,1,\zeta_{i-1}}(\mathbf{n}) \\ \vdots & & \vdots \\ w_{i,\zeta_i,1}(\mathbf{n}) & \dots & w_{i,\zeta_i,\zeta_{i-1}}(\mathbf{n}) \end{bmatrix} \in \mathbb{R}^{\zeta_i \times \zeta_{i-1}},$$

309 defining the so-called MIMO impulse response of  $\mathcal{W}_i$ , which plays a prominent role in dynamical  
310 system theory [52]. The MIMO impulse response of the ABBA layer  $\widetilde{\mathcal{W}}_i$  is then characterized by ABBA  
311 matrices:

$$312 \quad (4.5) \quad (\forall \mathbf{n} \in \mathbb{Z}^d) \quad \widetilde{\mathbf{W}}_i(\mathbf{n}) = \begin{bmatrix} \mathbf{W}_i^+(\mathbf{n}) & \mathbf{W}_i^-(\mathbf{n}) \\ \mathbf{W}_i^-(\mathbf{n}) & \mathbf{W}_i^+(\mathbf{n}) \end{bmatrix} \in [0, +\infty]^{(2\zeta_i) \times (2\zeta_{i-1})},$$

313 where  $\mathbf{W}_i^+(\mathbf{n}) = (w_{i,q,p}^+(\mathbf{n}))_{1 \leq q \leq \zeta_i, 1 \leq p \leq \zeta_{i-1}} \in [0, +\infty]^{\zeta_i \times \zeta_{i-1}}$  and  $\mathbf{W}_i^-(\mathbf{n}) =$   
314  $(w_{i,q,p}^-(\mathbf{n}))_{1 \leq q \leq \zeta_i, 1 \leq p \leq \zeta_{i-1}} \in [0, +\infty]^{\zeta_i \times \zeta_{i-1}}$ . This alternative view will be useful in the following  
315 sections.

316 **4.2. Lipschitz bounds for convolutional networks.** In this section, we establish bounds on  
317 the Lipschitz constant of an  $m$ -layer convolutional neural network  $T$ . Each linear operator  $\mathcal{W}_i$   
318 corresponding to layer  $i \in \{1, \dots, m\}$  will be defined by (4.1). We also define a variable

$$319 \quad (4.6) \quad \sigma_i = \prod_{l=1}^i s_l$$

320 aggregating strides from layer 1 to layer  $i$ . Subsequently, we will assume that, for every  $i \in \{1, \dots, m\}$ ,  
321 the activation operators  $(R_i)_{1 \leq i \leq m}$  are nonexpansive operators. Moreover, these operators are applied  
322 componentwise (see Appendix SM1). This means that, for every  $i \in \{1, \dots, m-1\}$ , there exists a  
323 function  $\rho_i$  from  $\mathbb{R}$  to  $\mathbb{R}$  such that

$$324 \quad (\forall \mathbf{x} \in \mathcal{H}_i) \quad \mathbf{y} = R_i(\mathbf{x}) \\ 325 \quad (4.7) \quad \Leftrightarrow (\forall p \in \{1, \dots, c_i\}) (\forall \mathbf{n} \in \mathbb{Z}^d) \quad y_p(\mathbf{n}) = \rho_i(x_p(\mathbf{n})).$$

327 In Appendix SM4, we derive frequency-based expressions allowing us to calculate bounds on  
328 the Lipschitz constant of  $T$ . For accurate numerical evaluations, the frequency transform in these  
329 expressions has to be replaced by a Discrete Fourier Transform involving a significant number of  
330 frequency bins (e.g.,  $128^d$ ). Due to this fact, a computation bottleneck occurs when MIMO filters  
331 are characterized by a large number of input/output channels (e.g., for 2D applications). In the  
332 following, we provide an alternative lower-complexity formulation for computing bounds of the  
333 Lipschitz constant. In the case of non-negative kernels, we show that this bound is tight.

334 **Theorem 4.1.** Let  $(\sigma_i)_{1 \leq i \leq m}$  be the aggregated stride factors of network  $T$ , as defined by (4.6), and  
335 let

$$336 \quad (4.8) \quad \mathbf{W} = (\mathbf{W}_m)_{\uparrow \sigma_{m-1}} * \dots * (\mathbf{W}_2)_{\uparrow \sigma_1} * \mathbf{W}_1$$

337 where  $(\mathbf{W}_i)_{1 \leq i \leq m}$  are the MIMO impulse responses of each layer of network  $T$  and, for every  $i \in$   
 338  $\{2, \dots, m\}$ ,  $(\mathbf{W}_i)_{\uparrow \sigma_{i-1}}$  is the interpolated sequence by a factor  $\sigma_{i-1}$  of  $\mathbf{W}_i$  (see (SM3.9)). For every  
 339  $\mathbf{j} \in \mathbb{S}(\sigma_m) = \{0, \dots, \sigma_m - 1\}^d$ , we define the following matrix:

$$340 \quad (4.9) \quad \overline{\mathbf{W}}^{(\mathbf{j})} = \sum_{\mathbf{n} \in \mathbb{Z}^d} \mathbf{W}(\sigma_m \mathbf{n} + \mathbf{j}) \in [0, +\infty[^{\zeta_m \times \zeta_0}.$$

341 Then

$$342 \quad (4.10) \quad \theta_m = \left\| \sum_{\mathbf{j} \in \mathbb{S}(\sigma_m)} \overline{\mathbf{W}}^{(\mathbf{j})} (\overline{\mathbf{W}}^{(\mathbf{j})})^\top \right\|^{1/2}$$

343 is a lower bound on the Lipschitz constant estimate of network  $T$ . In addition, if for every  $i \in \{1, \dots, m\}$ ,  
 344  $p \in \{1, \dots, \zeta_{i-1}\}$ , and  $q \in \{1, \dots, \zeta_i\}$ ,  $w_{i,q,p} = (w_{i,q,p}(\mathbf{n}))_{\mathbf{n} \in \mathbb{Z}^d}$  is a non-negative kernel, then  $\theta_m$  is a  
 345 Lipschitz constant of  $T$ .

346 The proof of Theorem 4.1 is given in Appendix C.

347 The constant  $\theta_m$  in (D.4) is actually equal to the one calculated in Appendix SM4. The following  
 348 majorization is thus obtained (see (SM4.4)):

$$349 \quad (4.11) \quad \theta_m \leq \bar{\theta}_m = \|\mathcal{W}_m\| \cdots \|\mathcal{W}_1\|.$$

350 By applying Theorem 4.1 to each individual layer  $(\mathcal{W}_i)_{1 \leq i \leq m}$  assumed to be with non-negative kernels,  
 351 we get the following expression for the upper-bound:

$$352 \quad (4.12) \quad \bar{\theta}_m = \prod_{i=1}^m \left\| \sum_{\mathbf{j} \in \mathbb{S}(s_i)} \overline{\mathbf{W}}_i^{(\mathbf{j})} (\overline{\mathbf{W}}_i^{(\mathbf{j})})^\top \right\|^{1/2},$$

353 where

$$354 \quad (4.13) \quad (\forall i \in \{1, \dots, m\}) (\forall \mathbf{j} \in \mathbb{S}(s_i)) \quad \overline{\mathbf{W}}_i^{(\mathbf{j})} = \sum_{\mathbf{n} \in \mathbb{Z}^d} \mathbf{W}_i(s_i \mathbf{n} + \mathbf{j}).$$

355 The bound  $\bar{\theta}_m$  is generally more tractable than  $\theta_m$  since it separates the influence of each layer and  
 356 does not require to compute the global matrix sequence  $\mathbf{W}$  as expressed by (4.8). However, such  
 357 separable bounds are usually loose. According to our observations, it turns out that, in the special  
 358 case of convolutional layers with non-negative kernels,  $\theta_m$  and  $\bar{\theta}_m$  are quite close (see numerical tests  
 359 in Appendix SM5).

360 To illustrate these results, the computation of the Lipschitz bound of a layer corresponding to an  
 361 average pooling is presented as an example in Appendix SM6.

362 **4.3. Bounds for ABBA convolutional networks.** Let us extend the previous results to the ABBA  
 363 context. The linear operators of the considered ABBA network  $\tilde{T}$  are denoted by  $(\tilde{\mathcal{W}}_i)_{0 \leq i \leq m+1}$ . The  
 364 weights in  $\tilde{\mathcal{W}}_0$  and  $\tilde{\mathcal{W}}_{m+1}$  are signed, whereas  $(\tilde{\mathcal{W}}_i)_{1 \leq i \leq m}$  are convolutional layers with  $d$ -dimensional  
 365 non-negative kernels. More precisely, we assume that, for every  $i \in \{1, \dots, m\}$ , the  $i$ -th layer of  
 366 the ABBA network has  $2\zeta_{i-1}$  input channels,  $2\zeta_i$  output channels, and stride  $s_i \in \mathbb{N} \setminus \{0\}$ . The  
 367 MIMO impulse response of such a layer is of the form (4.5). We make the same assumptions of  
 368 nonexpansiveness and separability for the activation operators as in the previous section. We recall that  
 369  $(\mathbf{W})_{\uparrow \sigma}$  denotes the interpolated version by a factor  $\sigma$  of a MIMO impulse response  $\mathbf{W} = (\mathbf{W}(\mathbf{n}))_{\mathbf{n} \in \mathbb{Z}^d}$ .

370 The following result is then established in Appendix D :

371 **Theorem 4.2.** *Under the above assumptions on the convolutional ABBA network  $\tilde{T}$ , let*

$$372 \quad (4.14) \quad (\forall i \in \{1, \dots, m\})(\forall \mathbf{j} \in \mathbb{S}(s_i)) \quad \Omega_i^{(\mathbf{j})} = \sum_{\mathbf{n} \in \mathbb{Z}^d} \mathfrak{S}(\tilde{\mathbf{W}}_i(s_i \mathbf{n} + \mathbf{j})) \in [0, +\infty[^{\zeta_i \times \zeta_{i-1}},$$

373 *where  $(\tilde{\mathbf{W}}_i(\mathbf{n}))_{\mathbf{n} \in \mathbb{Z}^d}$  is the MIMO impulse response of the ABBA layer of index  $i$ . Then a Lipschitz constant*  
374 *of  $\tilde{T}$  is*

$$375 \quad (4.15) \quad \bar{\theta}_m = \|\tilde{\mathbf{W}}_{m+1}\| \left( \prod_{i=1}^m \left\| \sum_{\mathbf{j} \in \mathbb{S}(s_i)} \Omega_i^{(\mathbf{j})} (\Omega_i^{(\mathbf{j})})^\top \right\| \right)^{1/2} \|\tilde{\mathbf{W}}_0\|,$$

376 *where  $\|\tilde{\mathbf{W}}_{m+1}\|$  (resp.  $\|\tilde{\mathbf{W}}_0\|$ ) is the spectral norm of the linear operator employed in the last (resp. first*  
377 *layer).*

378 The bound (4.15) will be subsequently used to control the Lipschitz constant of non-negative ABBA  
379 networks during their training.

380 **5. Lipschitz-constrained training.** The theoretical bounds established in the previous sections  
381 provide a relatively easy way of computing a tight estimate of the global Lipschitz constant. We  
382 propose a simple approach to control it during the training phase. Since our networks contain mostly  
383 layers having non-negative weights and a few layers having arbitrary-signed weights, their Lipschitz  
384 constant will be controlled separately and different constraint sets will be handled for each case.

385 To train a robust ABBA network, we employ a projected version of the well-known ADAM  
386 optimizer. Each layer  $i$  is parameterized by a vector  $\Psi_i$ . In the case of a dense layer,  $\Psi_i$  is a vector  
387 gathering the elements of weight matrix  $\tilde{\mathbf{W}}_i$ , the components of the associated bias  $\tilde{b}_i$ , and a possible  
388 additional parameter that will be introduced hereafter. For an ABBA layer,  $\Psi_i$  is thus a vector of  
389 dimension  $2N_i(N_{i-1} + 1)$  or  $2N_i(N_{i-1} + 1) + 1$ . In the context of a 2D convolutional layer, an array  
390  $w_i$  of scalar convolutional kernels is substituted for the weight matrix. In the ABBA case, we have  
391  $2\zeta_i\zeta_{i-1}$  such kernels. To ensure nonnegativity (if needed) and Lipschitz bound conditions on the  
392 weight operator, we project  $\Psi_i$  onto a suitable closed and convex constraint set. Considering pairs  
393  $(z_k)_{1 \leq k \leq K}$  of inputs images and their associated labels, the operations performed at each epoch  $n > 0$   
394 to minimize a loss function  $\ell$  are presented in Algorithm 5.1. After each iteration  $t$  of the optimizer, we  
395 perform a projection  $\text{proj}_{\mathcal{S}_{i,t}}$  onto a constraint set  $\mathcal{S}_{i,t}$ . The definition of this set and the corresponding  
396 method for managing the projection is detailed in the following, according to the network type.

397 **Handling Lipschitz constants for fully-connected layers.** Consider the network defined by  
398 Model (3.5). In the case of fully connected networks, the Lipschitz constant is given by Proposition  
399 3.8, which basically splits the bound into three terms: the first and the last account for the starting  
400 and ending layers, respectively, while the middle one encompasses all the ABBA layers. For the two  
401 former arbitrary-signed layers, we control the Lipschitz constants individually during training, by  
402 imposing a bound on each weight matrix spectral norm. This defines the following two constraints:

$$403 \quad (5.1) \quad (\forall i \in \{0, m + 1\}) \quad \|\tilde{\mathbf{W}}_i\| \leq \bar{\theta}_{m,i},$$

404 where  $\bar{\theta}_{m,i}$  is the imposed Lipschitz bound for the  $i$ -th layer. To deal with this constraint, we  
405 decompose the weight matrix as  $\tilde{\mathbf{W}}_i = \bar{\theta}_{m,i} \tilde{\mathbf{W}}'_i$ , which yields the constraint set

$$406 \quad (5.2) \quad (\forall i \in \{0, m + 1\}) \quad \mathcal{S}_{i,t} = \{\tilde{\mathbf{W}}'_i \mid \|\tilde{\mathbf{W}}'_i\| \leq 1\}.$$

407 The projection onto  $\mathcal{S}_{i,t}$  is performed by clipping the singular values of  $\widetilde{W}_i'$  to 1.

408 In our proposed training procedure, we set  $\bar{\theta}_{m,0}\bar{\theta}_{m,m+1} = 1$ . This gives the network one degree of  
 409 freedom to automatically adapt the value of the Lipschitz constant of these two layers. To do so, we  
 410 adopt the following parametrization

$$411 \quad (5.3) \quad \bar{\theta}_{m,0} = \exp(\alpha), \quad \bar{\theta}_{m,m+1} = \exp(-\alpha),$$

412 where  $\alpha \in \mathbb{R}$  is a trainable parameter. It constitutes an extra component of the vector  $\Psi_i$  when  
 413  $i \in \{0, m+1\}$ .

414 In the case of ABBA dense layers, we need to handle two requirements: ensure that, for every  
 415  $i \in \{1, \dots, m\}$ ,  $\widetilde{W}_i$  is a non-negative ABBA matrix, and to constrain the product of all the weight  
 416 matrices to be such that  $\|\widetilde{W}_m \cdots \widetilde{W}_1\| \leq \bar{\theta}_m$ . Since  $\bar{\theta}_{m,0}\bar{\theta}_{m,m+1} = 1$ ,  $\bar{\theta}_m$  corresponds to the target  
 417 Lipschitz bound for the ABBA network.

418 For every  $i \in \{1, \dots, m\}$ ,  $\widetilde{W}_i$  is parameterized by  $W_i^+$  and  $W_i^-$ . We define the following two  
 419 constraint sets:

$$420 \quad (5.4) \quad \mathcal{D}_i = \{(W_i^+, W_i^-) \in (\mathbb{R}^{N_i \times N_{i-1}})^2 \mid W_i^+ \geq 0 \text{ and } W_i^- \geq 0\},$$

$$421 \quad (5.5) \quad \mathcal{C}_{i,t} = \left\{ (W_i^+, W_i^-) \in (\mathbb{R}^{N_i \times N_{i-1}})^2 \mid \left\| A_{i,t} \begin{bmatrix} W_i^+ & W_i^- \\ W_i^- & W_i^+ \end{bmatrix} B_{i,t} \right\| \leq \bar{\theta}_m \right\}.$$

423 Here-above, matrix  $A_{i,t}$  (resp.  $B_{i,t}$ ) is an ABBA matrix, which is the product of the weight matrices  
 424 for the posterior (resp. previous layers). In this case,  $\mathcal{S}_{i,t} = \mathcal{D}_i \cap \mathcal{C}_{i,t}$ . To perform the projection onto  
 425 the intersection of these two sets, we use an instance of the proximal algorithm presented in [35],  
 426 which alternates between elementary projections onto  $\mathcal{D}_i$  and projections onto the spectral ball with  
 427 center 0 and radius  $\bar{\theta}_m$ . Because of Proposition 3.3(xii), the latter projection allows us to keep the  
 428 structure of ABBA matrices.

429 **Handling Lipschitz constants for convolutional layers.** In the case of convolutional ABBA  
 430 networks, we derived the bound in (4.15) which consists of the product of  $m+2$  terms. The Lipschitz  
 431 bound constraint is managed by introducing auxiliary variables  $(\bar{\theta}_{m,i})_{0 \leq i \leq m+1}$  defining upper bounds  
 432 for each layer. At iteration  $t$  of the algorithm estimates  $(\bar{\theta}_{m,i,t})_{0 \leq i \leq m+1}$  of the auxiliary bounds are  
 433 updated. Similarly to the fully connected case, we use two different types of constraints.

434 For the  $i$ -th ABBA convolutional layer with  $i \in \{1, \dots, m\}$ , we consider the constraint set

$$435 \quad (5.6) \quad \mathcal{C}_{i,t} = \{W_i \mid \left\| \sum_{j \in \mathbb{S}(s_i)} \Omega_i^{(j)} (\Omega_i^{(j)})^\top \right\| \leq \bar{\theta}_{m,i,t}^2\}$$

436 where matrices  $(\Omega_i^{(j)})_{j \in \mathbb{S}(s_i)}$  are linked to the convolution kernels by the linear relation (4.14). By  
 437 concatenating all these  $s_i^d$  matrices horizontally, we obtain a rectangular matrix  $\Omega_i$  which allows us to  
 438 reexpress (5.6) in the simpler form:

$$439 \quad (5.7) \quad \mathcal{C}_{i,t} = \{W_i \mid \|\Omega_i\| \leq \bar{\theta}_{m,i,t}\}.$$

440 We also have to impose the non-negativity of the filters alongside the stability bound. This corresponds  
 441 to a constraint set  $\mathcal{D}_i$ . Projecting onto  $\mathcal{S}_{i,t} = \mathcal{C}_{i,t} \cap \mathcal{D}_i$  is performed by using the same iterative proximal  
 442 algorithm as previously.

443 For the first and the last layers, we impose similarly that  $\|\mathcal{W}_i\| \leq \bar{\theta}_{m,i,t}$  with  $i \in \{0, m+1\}$ . Since  
 444 the kernels are signed, we resort a frequency formulation (see (SM4.18)) to estimate the spectral  
 445 norm of the convolutional operator. The procedure we use is described in Appendix SM8.

446 Convolutional layers are usually succeeded by an ABBA fully connected network. This part will  
 447 be handled as explained previously. However, we need to set the upper bounds  $(\bar{\theta}_{m,i,t})_{0 \leq i \leq m+1}$  used  
 448 in the convolutional part and the upper bound of the ABBA fully connected part. With a slight abuse  
 449 of notation, let us denote this latter bound by  $\bar{\theta}_{m,m+2,t}$ , while the target Lipschitz constant for the  
 450 global network is still denoted by  $\bar{\theta}_m$ . We have to deal with the following constraint:

$$451 \quad (5.8) \quad \prod_{i=0}^{m+2} \bar{\theta}_{m,i,t} = \bar{\theta}_m.$$

452 We proceed by computing the Lipschitz constants  $(\tilde{\theta}_{m,i,t})_{0 \leq i \leq m+2}$  of the layers after the ADAM update.  
 453 Then, we set

$$454 \quad (5.9) \quad (\forall i \in \{0, \dots, m+2\}) \quad \bar{\theta}_{m,i,t} = \tilde{\theta}_{m,i,t} \left( \frac{\bar{\theta}_m}{\prod_{i'=0}^{m+2} \tilde{\theta}_{m,i',t}} \right)^{\frac{1}{m+3}},$$

456 which guarantees that (5.8) holds. Update (5.9) can be interpreted as the orthogonal projection onto  
 457 the constraint set defined by (5.8) after a logarithmic transform of the auxiliary variables. The benefit  
 458 of such a transform is to convexify the constraint.

---

### Algorithm 5.1 Projected ADAM Algorithm

---

$\Psi_{i,t}$  – weights of layer  $i$  at iteration  $t$   
 $(z_k)_{1 \leq k \leq K}$  – set of input image-label pairs  
 $\beta_1, \beta_2$  – ADAM hyperparameters

---

Partition  $\{1, \dots, K\}$  into mini-batches  
 $(\mathbb{M}_{q,n})_{1 \leq q \leq Q}$   
 $t = (n-1)Q + q$   
 # sweep mini-batches  
**for**  $q \in \{1, \dots, Q\}$  **do**  
**for** layer  $i$  **do**  
 $g_{i,t} = \sum_{k \in \mathbb{M}_{q,n}} \nabla_i \ell(z_k, (\Psi_{i,t})_{1 \leq i \leq m})$   
 $\mu_{i,t} = \beta_1 \mu_{i,t-1} + (1 - \beta_1) g_{i,t}$   
 $\nu_{i,t} = \beta_2 \nu_{i,t-1} + (1 - \beta_2) g_{i,t}^2$   
 $\gamma_t = \gamma \sqrt{1 - \beta_2^t} / (1 - \beta_1^t)$   
 $\tilde{\Psi}_{i,t} = \Psi_{i,t} - \gamma_t \mu_{i,t} / (\sqrt{\nu_{i,t}} + \epsilon)$   
**end for**  
**for** layer  $i$  **do**  
 $\Psi_{i,t+1} = \text{proj}_{\mathcal{S}_{i,t}}(\tilde{\Psi}_{i,t})$   
**end for**  
**end for**

---

Dataset	Network	Architecture	Accuracy [%]
MNIST	ABBA	Dense	98.33
		Conv	98.70
	Non-Negative	Dense	94.95
		Conv	93.27
	Baseline	Dense	98.35
		Conv	98.68
FMNIST	ABBA	Dense	90.02
		Conv	90.17
	Non-Negative	Dense	84.56
		Conv	83.09
	Baseline	Dense	90.00
		Conv	90.20
RPS	ABBA	Conv	99.08
	Non-Negative	Conv	67.30
	Baseline	Conv	98.86
CelebA	ABBA	Conv	90.21
	Non-Negative	Conv	61.04
	Baseline	Conv	90.17

Table 1: Comparison between ABBA, full non-negative and arbitrary-signed (baseline) networks.

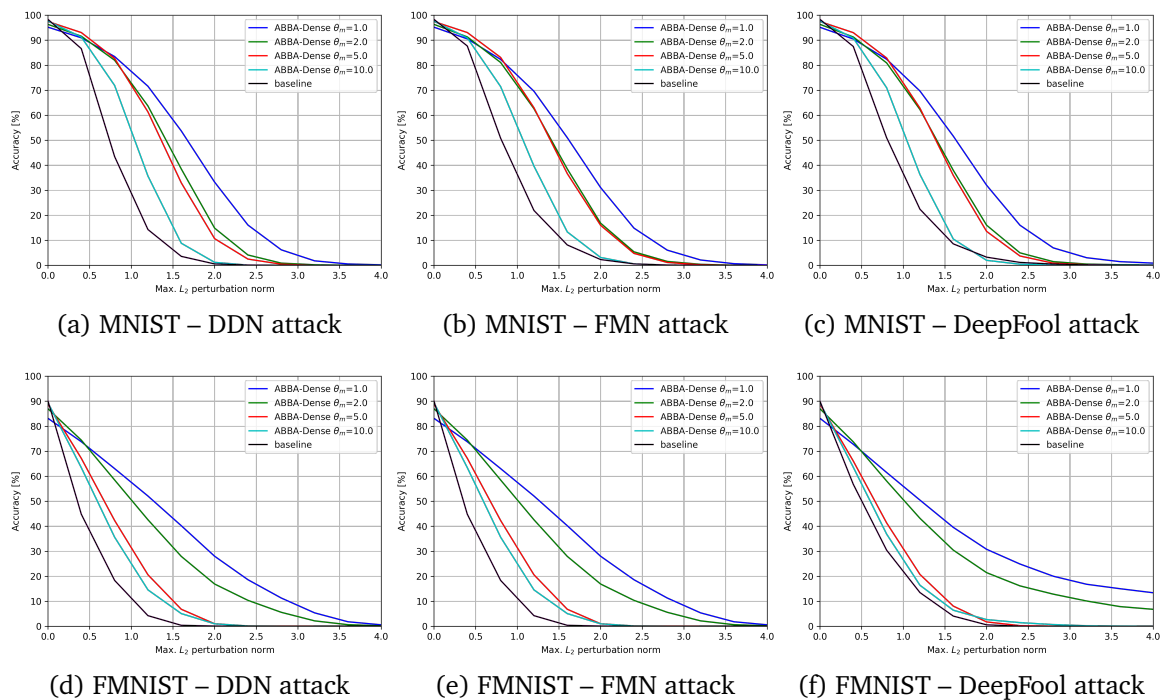


Figure 2: Accuracy vs. Perturbation for different Lipschitz constants – Dense Architecture.

459 **6. Experiments.** In this section, we show the versatility of ABBA neural networks in solving  
 460 classification tasks. The objective of our experiments is three-fold:

- 461 (i). First, we compare positive ABBA structures with their classic non-negative counterparts and  
 462 check that our method yields significantly better results in all considered cases.
- 463 (ii). We then train ABBA models constrained to different Lipschitz bound values and evaluate  
 464 their robustness against several adversarial attacks.
- 465 (iii). Finally, we compare our proposed approach with three other well-established defense strate-  
 466 gies, namely *Adversarial Training (AT)*, *Trade-off-inspired adversarial defense (TRADES)* [59],  
 467 *Deel-Lip* proposed by [45], and *orthonormalization* [2].

468 We validate our ABBA networks on four benchmark image classification datasets: MNIST, its more  
 469 complex variant Fashion MNIST<sup>4</sup>, a variant<sup>5</sup> of the Rock-Paper-Scissors (RPS) dataset [34], and a  
 470 binary classification on CelebA [31]. For the last dataset, inspired from [45] where eyeglass detection  
 471 is performed, we specialized our models on a different attribute, namely to identify whether a person  
 472 is bald or not. To explore the features of different ABBA network topologies, we experiment with two  
 473 main types of ABBA architectures: one having only fully-connected layers, further referred to as *ABBA*  
 474 *Dense*, and another one which includes a convolutional part for feature extraction, followed by a fully  
 475 connected classification module, *ABBA Conv*. Depending on the dataset, the particularities of each

<sup>4</sup><https://github.com/zalandoresearch/fashion-mnist>

<sup>5</sup><https://github.com/DrGFreeman/rps-cv>



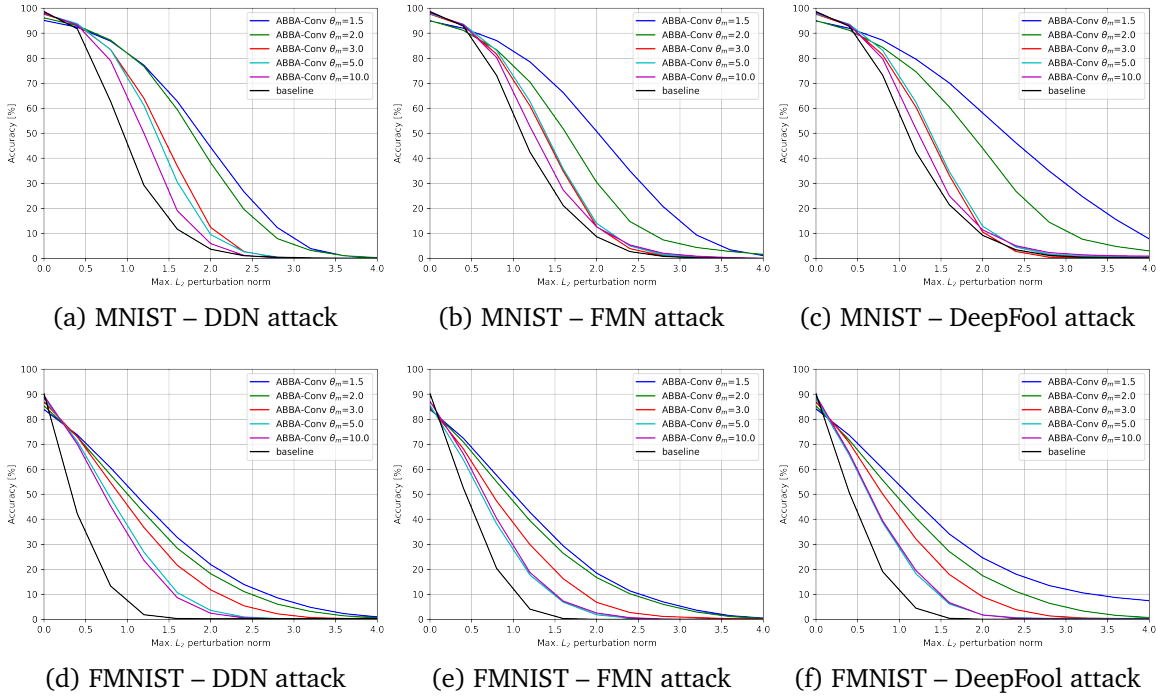


Figure 3: Acc. vs. Perturbation for different Lipschitz constants ABBA Conv Architectures – MNIST and FMNIST.

476 architecture are slightly different. A detailed description of all the small networks employed in this  
 477 work, as well as other training details, are provided in Appendix SM9. For all experiments, the input  
 478 images were scaled in the  $[-1, 1]$  interval.

479 **ABBA networks vs. non-negative networks.** First, we compare our non-negative ABBA networks  
 480 with standard ones trained under non-negativity constraints. We consider standard neural networks  
 481 having the same number of parameters as their ABBA equivalent. The results are summarized in Table  
 482 1, indicating that ABBA neural networks yield far superior results, in all cases. In the case of fully  
 483 connected architectures, the difference in terms of accuracy is around  $\sim 3\%$  and  $\sim 5\%$  for MNIST  
 484 and FMNIST, respectively. The difference is even higher when we consider Conv architectures (e.g.  
 485  $\sim 5\%$ ,  $\sim 7\%$ ,  $\sim 31\%$  and  $\sim 32\%$  for MNIST, FMNIST, RPS, and CelebA, respectively). This shows that  
 486 standard non-negative convolutional kernels are often suboptimal for extracting relevant information  
 487 from image data. On the other hand, training standard neural networks having arbitrary-signed  
 488 weights gives results very similar to their ABBA equivalents in all the cases, showing that ABBA  
 489 networks do not suffer from these shortcomings. These results are in agreement with Proposition 3.6.

490 **Stability vs. Performance.** According to the “no free lunch” theorem [51], stability guarantees  
 491 may impact the system performance on clean data. In this work, we train several models by following  
 492 the approach described in Section 5. The Lipschitz constant of the network is varied in an effort to  
 493 find the optimal trade-off between robustness and classification accuracy. This compromise is usually  
 494 use-case specific, depending on the architecture complexity and on the dataset particularities, so the

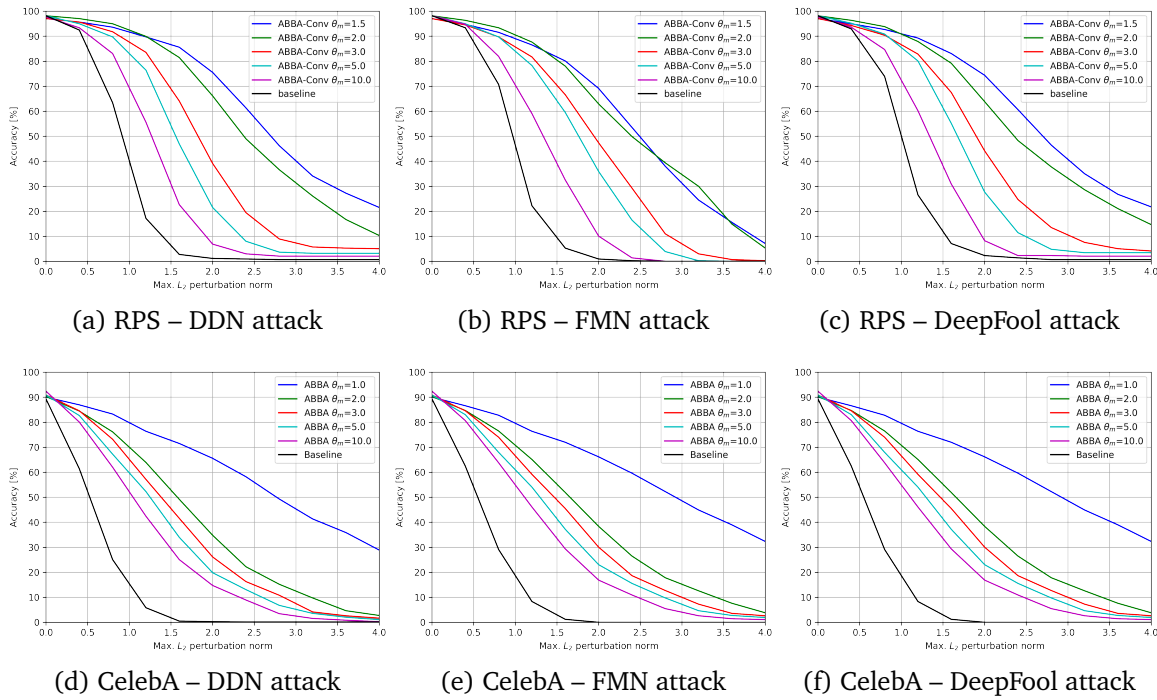


Figure 4: Acc. vs. Perturbation for different Lipschitz constants ABBA Conv Architectures – RPS and CelebA.

495 tightness of the imposed stability bound must be chosen accordingly. In our experiments, we limited  
 496 the maximum Lipschitz constant we impose, so that the drop in performance does not exceed 5% of  
 497 the baseline model accuracy (i.e., the model trained without robustness constraints).

498 **Adversarial attack validation.** We train several robust ABBA models by varying the global  
 499 Lipschitz bound  $\bar{\theta}_m$ . We then evaluate their robustness against inputs corrupted with different levels  
 500 of adversarial perturbations, by studying their influence on the overall performance of the system.  
 501 To create the adversarial perturbations, we use three white-box attackers as described next. **DDN**  
 502 [42] is a gradient-based  $\ell_2$  adversarial attack method that seeks to decouple the direction and norm  
 503 of the additive perturbation. By doing so, this attack is able to generate effective examples, while  
 504 requiring fewer iterations than other methods. **DeepFool** [33] considers a linear approximation to the  
 505 model and refines the attack sample iteratively, by selecting the point that would cross the decision  
 506 boundary with minimal effort in the logit space. The **FMN** [39] attack improves the approach in  
 507 DDN by introducing adaptive norm constraints on the perturbation, in order to balance the trade-off  
 508 between the magnitude of the perturbation and the level of miss-classification. This results in a  
 509 powerful attack that is able to generate adversarial examples with small perturbation levels.

510 For a given maximum  $\ell_2$  perturbation norm, we ran each attack for 300 steps, using the default  
 511 hyperparameters for each of the three attackers. Although all images are normalized in the  $[-1, 1]$   
 512 range, we report the robust accuracy w.r.t.  $\ell_2$  perturbation measured in  $[0, 1]$  range, which is the  
 513 common practice in the literature.

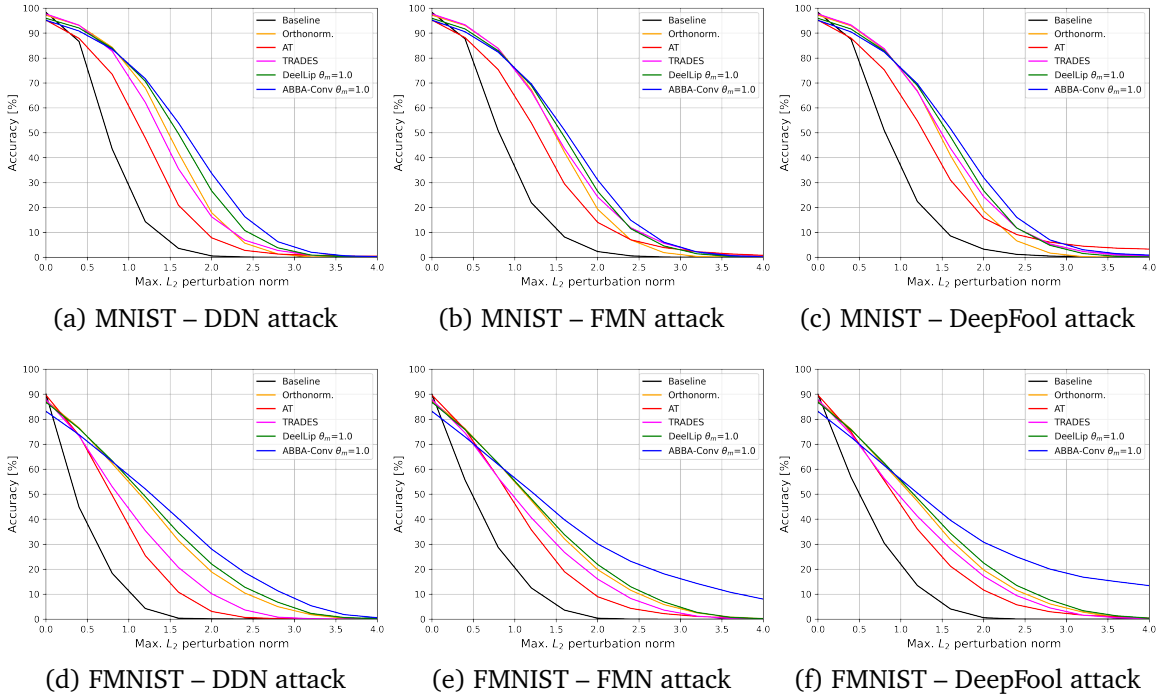


Figure 5: Comparisons with other defense techniques Dense Architectures

514 The results are summarized in Figures 2, 3, and 4 which show the robustness of MNIST, FMNIST,  
 515 RPS, and CeleBA ABBA models, w.r.t. increasing  $\ell_2$  norm perturbation, generated with DDN, FMN,  
 516 and DeepFool attacks. A baseline model, trained without stability constraints and arbitrary-signed  
 517 weights, is provided as a reference. These graphs could be interpreted as the expected performance  
 518 of the model if the attack is allowed to influence the input image with an  $\ell_2$ -norm less than  $\epsilon$ , where  
 519 the level of perturbation  $\epsilon$  varies. For a better understanding of the adversarial perturbation effect,  
 520 some visual examples of the attacked inputs, for all the datasets, are presented in Appendix SM10.

521 It can be observed that our robust ABBA models are significantly less affected by adversarial  
 522 inputs than the undefended baseline. This demonstrates that carefully controlling the Lipschitz  
 523 constant during training improves the network stability against adversarial attacks. Naturally, as the  
 524 imposed bound gets lower, the system becomes more robust. Although the difference in robustness  
 525 between similar values of the Lipschitz constant depends on the intrinsic structure of the dataset, our  
 526 results show that a good trade-off between robustness and performance can be achieved in all cases.

527 **Comparison with other defense strategies.** In the following, we compare our method for  
 528 training robust models using ABBA networks with other defense strategies. *Deel-Lip*, developed in  
 529 [45], is a popular Lipschitz-based approach, which uses the spectral normalization of each layer to  
 530 offer robustness certificates during training. We have also made comparisons with another popular  
 531 technique to ensure 1-Lipschitz weights, via *orthonormalization* [2, 8]. For our experiments, we

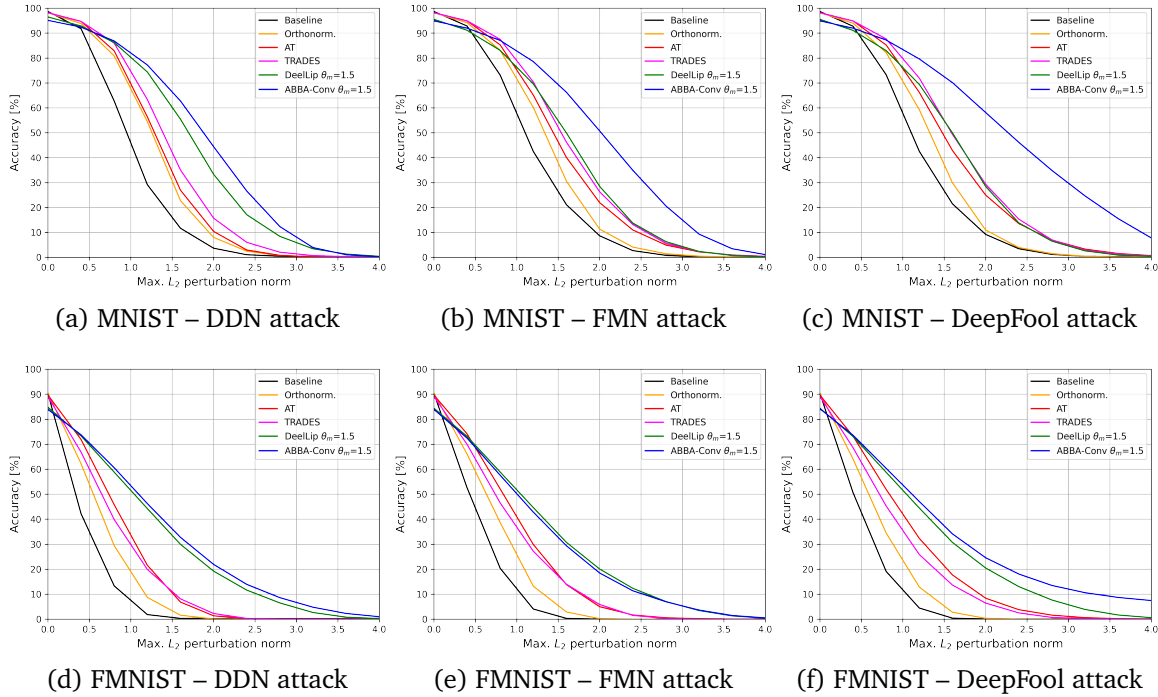


Figure 6: Comparisons with other defense techniques Conv Architectures – MNIST and FMNIST.

532 trained models from scratch, using the implementation provided by [2]<sup>6</sup>, in order to enforce an  
 533  $l_2$ -norm equal to 1 using *Bjork Orthonormalization*, while also preserving the gradient norm through  
 534 *GroupSort*. *TRADES* [59] introduces a robustness regularization term into the training objective.  
 535 This regularizer encourages the network to have similar predictions on both the original input and  
 536 its adversarial counterparts. On the other hand, *Adversarial Training* (AT) implies augmenting the  
 537 training data with adversarial samples, increasing the network generalization capabilities to different  
 538 input alterations. However, this technique offers weak theoretical stability guarantees, as it is mainly  
 539 dependent on the strength of the adversary used during training.

540 For all experiments regarding AT, we used Projected Gradient Descent (PGD) attack to generate  
 541 the adversarial samples with a perturbation level  $\epsilon = 0.5$  and then employed the scheduling strategy  
 542 introduced by [32]. Concerning TRADES, we set  $\lambda = 1$  for MNIST and FMNIST, and  $\lambda = 1/2$  for RPS  
 543 and CelebA datasets. For all the presented techniques we considered the equivalent baseline to each  
 544 ABBA network.

545 Comparisons, in the same adversarial set-up as before, are depicted in Figures 5, 6, and 7. We  
 546 observe that using our theoretically certified Lipschitz bound yields models which are generally more  
 547 robust than AT and TRADES. For simple datasets, such as MNIST and FMNIST, robust ABBA and  
 548 Deel-lip models exhibit similar behavior for low-magnitude adversarial attacks, but as we increase the  
 549 maximum perturbation  $\epsilon$ , our method performs better. In the case of real-world datasets (RPS and

<sup>6</sup><https://github.com/cemamil/LNets/tree/master>

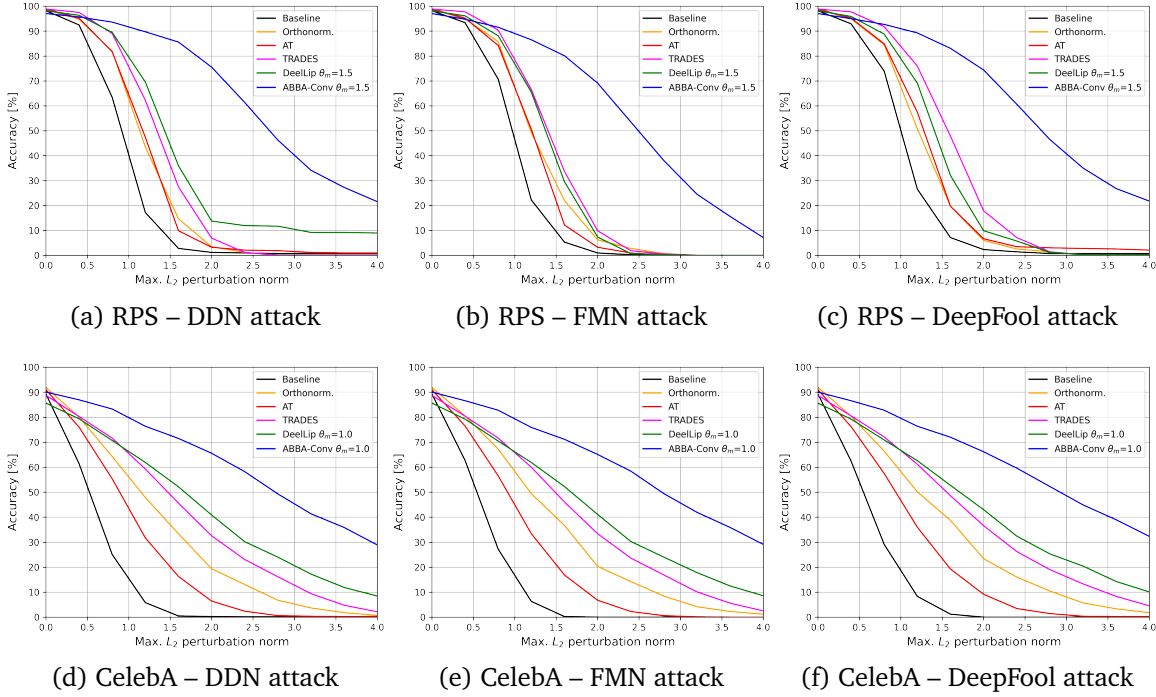


Figure 7: Comparisons with other defense techniques Conv Architectures – RPS and CelebA.

550 CelebA), ABBA models exhibit high robustness properties against all the tested attacks, showing that  
 551 our approach allows us to train neural models to reach great stability properties, without losing their  
 552 generalization power.

553 **Limitations.** The main limitation of our method is that non-negative ABBA operators require  
 554 more parameters to meet the universal approximation conditions. More precisely, for given depth  
 555  $m$  and number of neurons  $(N_i)_{0 \leq i \leq m}$  per layer, a network  $\tilde{T} \in \mathcal{N}_{m, \mathcal{A}}$  has the same number of inputs  
 556 and outputs as the standard feedforward network  $T$  in Model 3.1. All the layers of  $\tilde{T}$ , except the  
 557 first one, have however twice more inputs than  $T$ . Exact training times are showcased in SM11.  
 558 Because of the ABBA structure of the weight matrices in (3.11), the maximum number of parameters  
 559 of  $\tilde{T}$  is  $2(N_0^2 + \sum_{i=1}^m N_i(N_{i-1} + 1) + N_m^2)$  while the number of parameters of  $T$  is  $\sum_{i=1}^m N_i(N_{i-1} + 1)$ .  
 560 By storing  $\tilde{W}'_1 = \tilde{W}_1 \tilde{W}_0 \in \mathbb{R}^{(2N_1) \times N_0}$  instead of  $\tilde{W}_1$  and  $\tilde{W}_0$  separately, the maximum number of  
 561 parameters is reduced to  $2(\sum_{i=1}^m N_i(N_{i-1} + 1) + N_m^2)$ . Moreover, since the weights are non-negative,  
 562 the model does not necessarily require signed representations storage, so the memory space occupied  
 563 by  $\tilde{T}$  could also be reduced.

564 While our method does not provide any certification regarding the accuracy of the classifier in  
 565 adversarial environments, it delivers a certified value for the Lipschitz constant of the network.

566 **7. Conclusions.** In this paper, we introduce ABBA networks, a novel class of neural networks  
 567 where the majority of weights are non-negative. We demonstrate that these networks are universal  
 568 approximators, possessing all the expressive properties of conventional signed neural architectures.

569 Additionally, we unveil their remarkable algebraic characteristics, enabling us to derive precise  
570 Lipschitz bounds for both fully-connected and convolutive operators. Leveraging these bounds, we  
571 construct robust neural networks suitable for various classification tasks.

572 The main advantage of ABBA networks is that they enjoy all the properties of non-negative  
573 networks (e.g., they are easier to interpret and less prone to overfitting), without suffering from  
574 the shortcomings of standard ones (e.g., lack of expressivity). Additionally, we showed that ABBA  
575 structures allow tight Lipschitz bounds to be estimated, without requiring to solve an NP-hard problem  
576 as for conventional neural networks.

577 For future research, it would be intriguing to explore the application of ABBA networks in  
578 regression problems, where controlling the Lipschitz constant may present more challenges. Also,  
579 extending our theoretical bounds to different structures, such as recurrent or attention-based networks,  
580 holds promise for further advancements. Moreover, it would be interesting to study if using Lipschitz-  
581 constrained ABBA neural networks can improve certified robustness strategies like GloRoNets [27].

582 Finally, we recognize the necessity of investigating the scalability of the proposed training method  
583 to deep architectures. One of the main hurdles in this endeavor is the increased number of parameters  
584 that deep ABBA architectures entail.

585 **Acknowledgments.** We would like to thank Dr. Jérôme Rony and Prof. Ismail Ben Ayed from ETS  
586 Montréal for fruitful discussions and insightful suggestions.

587 **Appendix A. Proof of Proposition 3.6.** For every  $i \in \{1, \dots, m\}$ , let  $x_i = T_i(x_{i-1})$  where  
 588  $x_0 \in \mathbb{R}^{N_0}$  is an arbitrary input of network  $T$  and  $x_m \in \mathbb{R}^{N_m}$  its corresponding output. By using the  
 589 symmetry properties of the activation operators, for every  $i \in \{1, \dots, m\}$ , we have

$$\begin{aligned}
 590 \quad & (\forall i \in \{1, \dots, m\}) \quad x_i = R_i(W_i x_{i-1} + b_i) \\
 591 \quad (A.1) \quad & \quad \quad \quad = R_i(W_i^+ x_{i-1} - W_i^- x_{i-1} + b_i), \\
 592 \quad & -x_i = -R_i(W_i x_{i-1} + b_i) \\
 593 \quad & \quad \quad \quad = R_i(-W_i x_{i-1} - b_i + c_i) - d_i \\
 594 \quad (A.2) \quad & \quad \quad \quad = R_i(W_i^- x_{i-1} - W_i^+ x_{i-1} - b_i + c_i) - d_i.
 \end{aligned}$$

596 By making use of notation (3.14), (A.1) and (A.2) can be rewritten more concisely as

$$597 \quad (A.3) \quad (\forall i \in \{1, \dots, m\}) \quad \begin{bmatrix} x_i \\ -x_i \end{bmatrix} = \tilde{R}_i \left( \begin{bmatrix} W_i^+ & W_i^- \\ W_i^- & W_i^+ \end{bmatrix} \begin{bmatrix} x_{i-1} \\ -x_{i-1} \end{bmatrix} + \begin{bmatrix} b_i \\ c_i - b_i \end{bmatrix} \right) - \begin{bmatrix} 0 \\ d_i \end{bmatrix}.$$

598 Let us define, for every  $i \in \{0, \dots, m\}$ ,

$$599 \quad (A.4) \quad \tilde{x}_i = \begin{bmatrix} x_i \\ -x_i + d_i \end{bmatrix}.$$

600 Altogether (3.16), (3.11), (3.15), and (A.3) yield

$$\begin{aligned}
 601 \quad & (\forall i \in \{1, \dots, m\}) \quad \tilde{x}_i = \tilde{R}_i \left( \begin{bmatrix} W_i^+ & W_i^- \\ W_i^- & W_i^+ \end{bmatrix} \left( \tilde{x}_{i-1} - \begin{bmatrix} 0 \\ d_{i-1} \end{bmatrix} \right) + \begin{bmatrix} b_i \\ c_i - b_i \end{bmatrix} \right) \\
 602 \quad (A.5) \quad & \quad \quad \quad = \tilde{R}_i \left( \tilde{W}_i \tilde{x}_{i-1} + \tilde{b}_i \right). \\
 603
 \end{aligned}$$

604 This shows that, if  $(\tilde{T}_i)_{1 \leq i \leq m}$  are given by (3.8),  $\tilde{x}_m = \tilde{T}_m \cdots \tilde{T}_1(\tilde{x}_0)$ . By using the forms of  $\tilde{W}_0$  and  
 605  $\tilde{W}_{m+1}$  in (3.13), we deduce that

$$606 \quad (A.6) \quad x_m = \tilde{W}_{m+1} \tilde{x}_m + \tilde{b}_{m+1} = \tilde{T}(x_0).$$

607 **Appendix B. Proof for Proposition 3.8.** According to [53] [10, Proposition 5.5],

$$\begin{aligned}
 608 \quad (B.1) \quad & \vartheta_m = \sup_{\substack{\Lambda_1 \in \mathcal{D}_{\{-1,1\}}^{2N_1} \\ \vdots \\ \Lambda_m \in \mathcal{D}_{\{-1,1\}}^{2N_m}}} \|\tilde{W}_{m+1} \Lambda_m \tilde{W}_m \cdots \Lambda_1 \tilde{W}_1 \tilde{W}_0\|.
 \end{aligned}$$

609 where, for every  $i \in \{1, \dots, m\}$ ,  $\mathcal{D}_{\{-1,1\}}^{2N_i}$  designates the space of diagonal matrices of size  $(2N_i) \times (2N_i)$   
 610 with diagonal entries equal to  $-1$  or  $1$ . For every  $(\Lambda_1, \dots, \Lambda_m) \in \mathcal{D}_{\{-1,1\}}^{2N_1} \times \cdots \times \mathcal{D}_{\{-1,1\}}^{2N_m}$ ,

$$611 \quad (B.2) \quad \|\tilde{W}_{m+1} \Lambda_m \tilde{W}_m \cdots \Lambda_1 \tilde{W}_1 \tilde{W}_0\| \leq \|\tilde{W}_{m+1}\| \|\tilde{W}_m \Lambda_{m-1} \tilde{W}_{m-1} \cdots \Lambda_2 \tilde{W}_2 \Lambda_1 \tilde{W}_1\| \|\tilde{W}_0\|.$$

612 On the other hand, for every  $i \in \{1, \dots, m\}$ ,  $\tilde{W}_i \in [0, +\infty]^{(2N_i) \times (2N_{i-1})}$ . It then follows from [10,  
 613 Proposition 5.10] that

$$\begin{aligned}
 614 \quad (B.3) \quad & \sup_{\substack{\Lambda_1 \in \mathcal{D}_{\{-1,1\}}^{2N_1} \\ \vdots \\ \Lambda_{m-1} \in \mathcal{D}_{\{-1,1\}}^{2N_{m-1}}} \|\tilde{W}_m \Lambda_{m-1} \tilde{W}_{m-1} \cdots \Lambda_2 \tilde{W}_2 \Lambda_1 \tilde{W}_1\| = \|\tilde{W}_m \tilde{W}_{m-1} \cdots \tilde{W}_2 \tilde{W}_1\|.
 \end{aligned}$$

615 According to Proposition 3.3(iii),  $\widetilde{W}_m \widetilde{W}_{m-1} \cdots \widetilde{W}_2 \widetilde{W}_1 \in \mathcal{A}_{N_m, N_0}$ . Since this matrix has nonnegative  
616 elements, we deduce from Proposition 3.3(vii) that

$$\begin{aligned} 617 \quad \vartheta_m &\leq \|\widetilde{W}_{m+1}\| \|\widetilde{W}_m \cdots \widetilde{W}_1\| \|\widetilde{W}_0\| \\ 618 &= \|\widetilde{W}_{m+1}\| \|\mathfrak{S}(\widetilde{W}_m \cdots \widetilde{W}_1)\| \|\widetilde{W}_0\| \\ 619 \quad \text{(B.4)} &= \|\widetilde{W}_{m+1}\| \|\mathfrak{S}(\widetilde{W}_m) \cdots \mathfrak{S}(\widetilde{W}_1)\| \|\widetilde{W}_0\|, \end{aligned}$$

621 where the last equality is also a consequence of Proposition 3.3(iii). This leads to the Lipschitz bound  
622 in (3.19).

623 **Appendix C. Proof of Theorem 4.1.**

624 Before giving the proof of our main result, we will introduce a link between the Fourier and the  
625 spatial representations for a nonnegative convolutional kernel.

626 **Lemma C.1.** *Let  $(c, c') \in (\mathbb{N} \setminus \{0\})^2$  and let*

$$627 \quad \text{(C.1)} \quad (\forall \mathbf{n} \in \mathbb{Z}^d) \quad \mathbf{H}(\mathbf{n}) = (h_{q,p}(\mathbf{n}))_{1 \leq q \leq c', 1 \leq p \leq c} \in [0, +\infty[^{c' \times c}$$

628 where, for every  $p \in \{1, \dots, c\}$  and  $q \in \{1, \dots, c'\}$ ,  $h_{q,p} \in \ell^1(\mathbb{Z}^d)$ <sup>7</sup>. Then, the Fourier transform  $\widehat{\mathbf{H}}$  of  
629  $(\mathbf{H}(\mathbf{n}))_{\mathbf{n} \in \mathbb{Z}^d}$  is such that

$$630 \quad \text{(C.2)} \quad \sup_{\boldsymbol{\nu} \in [0,1]^d} \|\widehat{\mathbf{H}}(\boldsymbol{\nu})\| = \left\| \sum_{\mathbf{n} \in \mathbb{Z}^d} \mathbf{H}(\mathbf{n}) \right\|.$$

631 *Proof.* For every  $\boldsymbol{\nu} \in [0,1]^d$ ,

$$632 \quad \text{(C.3)} \quad \widehat{\mathbf{H}}(\boldsymbol{\nu}) = \sum_{\mathbf{n} \in \mathbb{Z}^d} \mathbf{H}(\mathbf{n}) \exp(-i2\pi \mathbf{n}^\top \boldsymbol{\nu}).$$

633 For every  $\mathbf{u} = [u_1, \dots, u_c]^\top \in \mathbb{C}^c$ , by using the triangle inequality,

$$\begin{aligned} 634 \quad \|\widehat{\mathbf{H}}(\boldsymbol{\nu}) \mathbf{u}\|^2 &= \sum_{q=1}^{c'} \left| \sum_{p=1}^c [\mathbf{H}(\boldsymbol{\nu})]_{q,p} u_p \right|^2 \\ 635 &= \sum_{q=1}^{c'} \left| \sum_{p=1}^c \sum_{\mathbf{n} \in \mathbb{Z}^d} h_{q,p}(\mathbf{n}) \exp(-i2\pi \mathbf{n}^\top \boldsymbol{\nu}) u_p \right|^2 \\ 636 &\leq \sum_{q=1}^{c'} \left( \sum_{p=1}^c \sum_{\mathbf{n} \in \mathbb{Z}^d} h_{q,p}(\mathbf{n}) |u_p| \right)^2 \\ 637 &= \left\| \sum_{\mathbf{n} \in \mathbb{Z}^d} \mathbf{H}(\mathbf{n}) |\mathbf{u}| \right\|^2 \\ 638 &\leq \left\| \sum_{\mathbf{n} \in \mathbb{Z}^d} \mathbf{H}(\mathbf{n}) \right\|^2 \|\mathbf{u}\|^2 \\ 639 \quad \text{(C.4)} &= \left\| \sum_{\mathbf{n} \in \mathbb{Z}^d} \mathbf{H}(\mathbf{n}) \right\|^2 \|\mathbf{u}\|^2, \\ 640 \end{aligned}$$

---

<sup>7</sup> $\ell^1(\mathbb{Z}^d)$  is the space of summable  $d$ -dimensional sequences



641 where  $|\mathbf{u}|$  denotes the vector of moduli of the components of vector  $\mathbf{u}$ . This shows that

$$642 \quad (\text{C.5}) \quad \|\widehat{\mathbf{H}}(\boldsymbol{\nu})\| \leq \left\| \sum_{\mathbf{n} \in \mathbb{Z}^d} \mathbf{H}(\mathbf{n}) \right\|.$$

643 and, consequently,

$$644 \quad (\text{C.6}) \quad \sup_{\boldsymbol{\nu} \in [0,1]^d} \|\widehat{\mathbf{H}}(\boldsymbol{\nu})\| \leq \left\| \sum_{\mathbf{n} \in \mathbb{Z}^d} \mathbf{H}(\mathbf{n}) \right\|.$$

645 In addition, the upper bound is attained since

$$646 \quad (\text{C.7}) \quad \widehat{\mathbf{H}}(\mathbf{0}) = \sum_{\mathbf{n} \in \mathbb{Z}^d} \mathbf{H}(\mathbf{n}). \quad \blacksquare$$

647 Next, we derive the proof for Theorem 4.1 in light of Lemma C.1.

648

649 *Proof.*  $\mathbf{W}$  being the impulse response of the MIMO filter with the frequency response given by  
650 (SM4.1), it follows from Noble identities [52] that  $\mathcal{W}$  is equivalent to a convolution with  $\mathbf{W}$  followed  
651 by a decimation by a factor  $\sigma_m$ . Let  $\mathbf{x} \in \mathcal{H}_0$ . Let the  $\sigma_m$ -polyphase representation of  $\mathbf{x}$  (resp.  $\mathbf{W}$ ) be  
652 defined as

$$653 \quad (\text{C.8}) \quad (\forall \mathbf{j} \in \mathbb{S}(\sigma_m)) (\forall \mathbf{n} \in \mathbb{Z}^d) \quad \mathbf{x}^{(\mathbf{j})}(\mathbf{n}) = \mathbf{x}(\sigma_m \mathbf{n} - \mathbf{j})$$

$$654 \quad (\text{C.9}) \quad \left( \text{resp. } \mathbf{W}^{(\mathbf{j})}(\mathbf{n}) = \mathbf{W}(\sigma_m \mathbf{n} + \mathbf{j}) \right).$$

656 Then, as a result of multirate digital filtering,  $\mathbf{y} = \mathcal{W}\mathbf{x}$  if and only if

$$657 \quad (\text{C.10}) \quad \mathbf{y} = \sum_{\mathbf{j} \in \mathbb{S}(\sigma_m)} \mathbf{W}^{(\mathbf{j})} * \mathbf{x}^{(\mathbf{j})}.$$

658 This sum of MIMO convolutions can be reformulated as a single one

$$659 \quad (\text{C.11}) \quad \mathbf{y} = \mathbf{H} * \mathbf{e},$$

660 where  $\mathbf{H}$  is the  $c_m \times \sigma_m^d c_0$  MIMO impulse response obtained by stacking rowwise the polyphase  
661 MIMO impulse responses  $(\mathbf{W}^{(\mathbf{j})})_{\mathbf{j} \in \mathbb{S}(\sigma_m)}$  and  $\mathbf{e}$  is the  $\sigma_m^d c_0$ -component  $d$ -dimensional signal obtained  
662 by stacking columnwise the polyphase signal components  $(\mathbf{x}^{(\mathbf{j})})_{\mathbf{j} \in \mathbb{S}(\sigma_m)}$ . For example, if  $d = 2$ , we  
663 have

$$664 \quad (\forall \mathbf{n} \in \mathbb{Z}^2) \quad \mathbf{H}(\mathbf{n}) = [\mathbf{W}^{(0,0)}(\mathbf{n}), \dots, \mathbf{W}^{(\sigma_m-1,0)}(\mathbf{n}), \dots, \mathbf{W}^{(0,\sigma_m-1)}(\mathbf{n}), \dots, \mathbf{W}^{(\sigma_m-1,\sigma_m-1)}(\mathbf{n})]$$

667 and

$$668 \quad (\text{C.13}) \quad (\forall \mathbf{n} \in \mathbb{Z}^2) \quad \mathbf{e}(\mathbf{n}) = \begin{bmatrix} \mathbf{x}^{(0,\sigma_m-1)}(\mathbf{n}) \\ \vdots \\ \mathbf{x}^{(\sigma_m-1,\sigma_m-1)}(\mathbf{n}) \\ \vdots \\ \mathbf{x}^{(\sigma_m-1,\sigma_m-1)}(\mathbf{n}) \\ \vdots \\ \mathbf{x}^{(\sigma_m-1,\sigma_m-1)}(\mathbf{n}) \end{bmatrix}.$$

669 Note that, according to (C.8),

$$\begin{aligned}
 670 \quad \|\mathbf{e}\|^2 &= \sum_{\mathbf{n} \in \mathbb{Z}^d} \|\mathbf{e}(\mathbf{n})\|^2 \\
 671 \quad &= \sum_{\mathbf{n} \in \mathbb{Z}^d} \sum_{\mathbf{j} \in \mathbb{S}(\sigma_m)} \|\mathbf{x}^{(\mathbf{j})}(\mathbf{n})\|^2 \\
 672 \quad &= \sum_{\mathbf{n} \in \mathbb{Z}^d} \|\mathbf{x}(\mathbf{n})\|^2 \\
 673 \quad (C.14) \quad &= \|\mathbf{x}\|^2.
 \end{aligned}$$

675 This equality and (C.11) imply that

$$676 \quad (C.15) \quad \|\mathcal{W}\| = \sup_{\boldsymbol{\nu} \in [0,1]^d} \|\widehat{\mathbf{H}}(\boldsymbol{\nu})\|.$$

677 We thus deduce from Lemma C.1 that

$$678 \quad (C.16) \quad \|\mathcal{W}\| = \left\| \sum_{\mathbf{n} \in \mathbb{Z}^d} \mathbf{H}(\mathbf{n}) \right\|.$$

679 On the other hand, by using (C.9),

$$\begin{aligned}
 680 \quad &\left\| \sum_{\mathbf{n} \in \mathbb{Z}^d} \mathbf{H}(\mathbf{n}) \right\| \\
 681 \quad &= \left\| \left( \sum_{\mathbf{n} \in \mathbb{Z}^d} \mathbf{H}(\mathbf{n}) \right) \left( \sum_{\mathbf{n} \in \mathbb{Z}^d} \mathbf{H}(\mathbf{n}) \right)^\top \right\|^{1/2} \\
 682 \quad &= \left\| \sum_{\mathbf{j} \in \mathbb{S}(\sigma_m)} \left( \sum_{\mathbf{n} \in \mathbb{Z}^d} \mathbf{W}^{(\mathbf{j})}(\mathbf{n}) \right) \left( \sum_{\mathbf{n} \in \mathbb{Z}^d} \mathbf{W}^{(\mathbf{j})}(\mathbf{n}) \right)^\top \right\|^{1/2} \\
 683 \quad (C.17) \quad &= \left\| \sum_{\mathbf{j} \in \mathbb{S}(\sigma_m)} \overline{\mathbf{W}}^{(\mathbf{j})} (\overline{\mathbf{W}}^{(\mathbf{j})})^\top \right\|^{1/2}. \quad \blacksquare \\
 684
 \end{aligned}$$

#### 685 Appendix D. Proof of Theorem 4.2.

686 This result is a consequence of Theorem 4.1, which provides a Lipschitz bound for nonnegative  
 687 convolutional neural networks. By following a similar reasoning to Appendix B, a Lipschitz constant  
 688 of the ABBA network is

$$689 \quad (D.1) \quad \theta_m = \|\widetilde{\mathbf{W}}_{m+1}\| \|\widetilde{\mathbf{W}}_m \circ \dots \circ \widetilde{\mathbf{W}}_1\| \|\widetilde{\mathbf{W}}_0\|.$$

690 Let

$$691 \quad (D.2) \quad \widetilde{\mathbf{W}} = (\widetilde{\mathbf{W}}_m)_{\uparrow \sigma_{m-1}} * \dots * (\widetilde{\mathbf{W}}_2)_{\uparrow \sigma_1} * \widetilde{\mathbf{W}}_1$$

692 and let

$$693 \quad (D.3) \quad (\forall \mathbf{j} \in \mathbb{S}(\sigma_m)) \quad \Omega^{(\mathbf{j})} = \sum_{\mathbf{n} \in \mathbb{Z}^d} \mathfrak{S}(\widetilde{\mathbf{W}}(\sigma_m \mathbf{n} + \mathbf{j})) \in [0, +\infty[^{\zeta_m \times \zeta_0}.$$

694 Since  $(\widetilde{W}_i)_{1 \leq i \leq m}$  are convolutional operators with nonnegative kernels, it follows from Theorem 4.1  
695 that

$$696 \quad (D.4) \quad \theta_m = \|\widetilde{W}_{m+1}\| \left\| \sum_{\mathbf{j} \in \mathbb{S}(\sigma_m)} \overline{\mathbf{W}}^{(\mathbf{j})} (\overline{\mathbf{W}}^{(\mathbf{j})})^\top \right\|^{1/2} \|\widetilde{W}_0\|.$$

697 where, for every  $\mathbf{j} \in \mathbb{S}(\sigma_m)$ ,

$$698 \quad (D.5) \quad \overline{\mathbf{W}}^{(\mathbf{j})} = \sum_{\mathbf{n} \in \mathbb{Z}^d} \widetilde{\mathbf{W}}(\sigma_m \mathbf{n} + \mathbf{j}) \in [0, +\infty[^{(2\zeta_m) \times (2\zeta_0)}.$$

699 On the other hand, for every  $\mathbf{n} \in \mathbb{Z}^d$ ,  $\widetilde{\mathbf{W}}_i(\mathbf{n})$  is an ABBA matrix. Since  $\widetilde{\mathbf{W}}(\mathbf{n})$  is obtained by  
700 multiplication and addition of such matrices, it follows from Proposition 3.3(ii) and (iii) that it is  
701 also an ABBA matrix. We deduce that, for every  $\mathbf{j} \in \mathbb{S}(\sigma_m)$ ,  $\overline{\mathbf{W}}^{(\mathbf{j})}$  is an ABBA matrix and, by using  
702 Proposition 3.3(i),  $\overline{\mathbf{W}}^{(\mathbf{j})} (\overline{\mathbf{W}}^{(\mathbf{j})})^\top$  is ABBA. By invoking now Proposition 3.3(i)-(iii) and (vii), we  
703 deduce that

$$\begin{aligned} 704 \quad \left\| \sum_{\mathbf{j} \in \mathbb{S}(\sigma_m)} \overline{\mathbf{W}}^{(\mathbf{j})} (\overline{\mathbf{W}}^{(\mathbf{j})})^\top \right\| &= \left\| \mathfrak{S} \left( \sum_{\mathbf{j} \in \mathbb{S}(\sigma_m)} \overline{\mathbf{W}}^{(\mathbf{j})} (\overline{\mathbf{W}}^{(\mathbf{j})})^\top \right) \right\| \\ 705 &= \left\| \sum_{\mathbf{j} \in \mathbb{S}(\sigma_m)} \mathfrak{S}(\overline{\mathbf{W}}^{(\mathbf{j})}) \mathfrak{S}(\overline{\mathbf{W}}^{(\mathbf{j})})^\top \right\| \\ 706 \quad (D.6) \quad &= \left\| \sum_{\mathbf{j} \in \mathbb{S}(\sigma_m)} \Omega^{(\mathbf{j})} (\Omega^{(\mathbf{j})})^\top \right\|. \\ 707 \end{aligned}$$

708 This shows that a Lipschitz constant of the ABBA network  $\widetilde{T}$  is

$$709 \quad (D.7) \quad \theta_m = \|\widetilde{W}_{m+1}\| \left\| \sum_{\mathbf{j} \in \mathbb{S}(\sigma_m)} \Omega^{(\mathbf{j})} (\Omega^{(\mathbf{j})})^\top \right\|^{1/2} \|\widetilde{W}_0\|,$$

710 Similarly to the derivation of (4.12), we deduce that  $\theta_m \leq \bar{\theta}_m$  where  $\bar{\theta}_m$  is given by (4.15).

711

## REFERENCES

- 712 [1] B. AMOS, L. XU, AND J. Z. KOLTER, *Input convex neural networks*, Proc. Int. Conf. Machine Learn., vol. 70,  
713 2017, pp. 146–155.
- 714 [2] C. ANIL, J. LUCAS, AND R. GROSSE, *Sorting out Lipschitz function approximation*, Proc. Int. Conf. Machine  
715 Learn., 2019, pp. 291–301.
- 716 [3] K. D. APOSTOLIDIS AND G. A. PAPAKOSTAS, *A survey on adversarial deep learning robustness in medical image  
717 analysis*, Electronics, vol. 10, 2021.
- 718 [4] B. O. AYINDE AND J. M. ZURADA, *Deep Learning of constrained autoencoders for enhanced understanding of  
719 data*, IEEE Trans. Neural Net. and Learn. Syst., vol. 29, 2018, pp. 3969–3979.
- 720 [5] L. BÉTHUNE, T. BOISSIN, M. SERRURIER, F. MAMALET, C. FRIEDRICH, AND A. GONZALEZ SANZ, *Pay attention  
721 to your loss : understanding misconceptions about Lipschitz neural networks*, Proc. Ann. Conf. Neur. Inform.  
722 Proc. Syst., vol. 35, 2022, pp. 20077–20091.
- 723 [6] J. BROMLEY, I. GUYON, Y. LECUN, E. SÄCKINGER, AND R. SHAH, *Signature verification using a "siamese" time  
724 delay neural network*, Proc. Ann. Conf. Neur. Inform. Proc. Syst., vol. 6, 1993, pp. 737–744.
- 725 [7] J. CHOROWSKI AND J. M. ZURADA, *Learning understandable neural networks with nonnegative weight con-  
726 straints*, IEEE Trans. Neural Net. and Learn. Syst., vol. 26, 2015, pp. 62–69.

- 727 [8] M. CISSE, P. BOJANOWSKI, E. GRAVE, Y. DAUPHIN, AND N. USUNIER, *Parseval networks: Improving robustness*  
728 *to adversarial examples*, Proc. Int. Conf. Mach. Learn., 2017, pp. 854–863.
- 729 [9] J. COHEN, E. ROSENFELD, AND Z. KOLTER, *Certified adversarial robustness via randomized smoothing*, Proc.  
730 Int. Conf. Machine Learn., 2019, pp. 1310–1320.
- 731 [10] P. L. COMBETTES AND J.-C. PESQUET, *Lipschitz certificates for layered network structures driven by averaged*  
732 *activation operators*, J. Math. Data Sci., vol. 2, 2020, pp. 529–557.
- 733 [11] F. CROCE, M. ANDRIUSHCHENKO, AND M. HEIN, *Provable robustness of ReLU networks via maximization of*  
734 *linear regions*, Proc. Int. Conf. Art. Intell. and Statistics, 2019, pp. 2057–2066.
- 735 [12] I. DIAKONIKOLAS, D. M. KANE, V. KONTONIS, AND N. ZARIFIS, *Algorithms and SQ lower bounds for PAC*  
736 *learning one-hidden-layer ReLU networks*, Proc. Conf. Learn. Theory, 2020, pp. 1514–1539.
- 737 [13] D. GAMARNIK, E. C. KIZILDAĞ, AND I. ZADIK, *Self-regularity of output weights for overparameterized two-layer*  
738 *neural networks*, Proc. IEEE Int. Symp. Info. Theory, 2021, pp. 819–824.
- 739 [14] M. GLEIZE, E. SHNARCH, L. CHOSHEN, L. DANKIN, G. MOSHKOWICH, R. AHARONOV, AND N. SLONIM, *Are*  
740 *you convinced? Choosing the more convincing evidence with a Siamese Network*, Proc. Ann. Meeting of the  
741 Assoc. for Comp. Linguistics, 2019, pp. 967–976.
- 742 [15] I. GOODFELLOW, J. SHLENS, AND C. SZEGEDY, *Explaining and harnessing adversarial examples*, Proc. Int. Conf.  
743 Learn. Represent., 2015.
- 744 [16] S. GOWAL, K. DVIJOTHAM, R. STANFORTH, T. A. MANN, AND P. KOHLI, *A dual approach to verify and train*  
745 *deep networks*, Proc. Int. Conf. Art. Intell., 2019, pp. 6156–6160.
- 746 [17] C. GUO, B. KARRER, K. CHAUDHURI, AND L. VAN DER MAATEN, *Bounding training data reconstruction in*  
747 *private (deep) learning*, Proc. Int. Conf. Machine Learn., 2022, pp. 8056–8071.
- 748 [18] K. GUPTA, F. KAAKAI, B. PESQUET-POPESCU, J.-C. PESQUET, AND F. D. MALLIAROS, *Multivariate Lipschitz*  
749 *analysis of the stability of neural networks*, Frontiers in Signal Processing, vol. 2, 2022.
- 750 [19] A. HE, C. LUO, X. TIAN, AND W. ZENG, *A twofold siamese network for real-time object tracking*, Proc. IEEE  
751 Conf. Comput. Vis. Pattern Recogn., 2018, pp. 4834–4843.
- 752 [20] G. KATZ, C. BARRETT, D. L. DILL, K. JULIAN, AND M. J. KOCHENDERFER, *Reluplex: An efficient SMT solver for*  
753 *verifying deep neural networks*, Proc. Int. Conf. Comp. Aided Verif., 2017, pp. 97–117.
- 754 [21] G. KHROMOV AND S. P. SINGH, *Some fundamental aspects about Lipschitz continuity of neural network functions*,  
755 arXiv:2302.10886, 2023.
- 756 [22] P. KIDGER AND T. LYONS, *Universal approximation with deep narrow networks*, Proc. Conf. Learn. Theory,  
757 2020, pp. 2306–2327.
- 758 [23] G. KOCH, R. ZEMEL, AND R. SALAKHUTDINOV, *Siamese neural networks for one-shot image recognition*, Proc.  
759 Int. Conf. Machine Learn., vol. 2, 2015.
- 760 [24] A. KURAKIN, I. GOODFELLOW, AND S. BENGIO, *Adversarial machine learning at scale*, Proc. Int. Conf. Learn.  
761 Represent., 2016.
- 762 [25] M. LECUYER, V. ATLIKAKIS, R. GEAMBASU, D. HSU, AND S. JANA, *Certified robustness to adversarial examples*  
763 *with differential privacy*, Proc. IEEE Symp. Security and Privacy, 2019, pp. 656–672.
- 764 [26] D. D. LEE AND H. S. SEUNG, *Learning the parts of objects by non-negative matrix factorization*, Nature, vol. 401,  
765 1999, pp. 788–791.
- 766 [27] K. LEINO, Z. WANG, AND M. FREDRIKSON, *Globally-robust neural networks*, Proc. Int. Conf. Mach. Learn.,  
767 2021, pp. 6212–6222.
- 768 [28] M. LESHNO, V. Y. LIN, A. PINKUS, AND S. SCHOCKEN, *Multilayer feedforward networks with a nonpolynomial*  
769 *activation function can approximate any function*, Neural Networks, vol. 6, 1993, pp. 861–867.
- 770 [29] B. LI, C. CHEN, W. WANG, AND L. CARIN, *Certified adversarial robustness with additive noise*, Proc. Ann. Conf.  
771 Neur. Inform. Proc. Syst., vol. 32, 2019.
- 772 [30] Z. LI AND D. HOIEM, *Learning without forgetting*, IEEE Trans. Pattern Anal. Machine Intell., vol. 40, 2017,  
773 pp. 2935–2947.
- 774 [31] Z. LIU, P. LUO, X. WANG, AND X. TANG, *Deep learning face attributes in the wild*, Proc. IEEE Int. Conf. Comput.  
775 Vis., vol. 4, 2015, pp. 3730–3738.
- 776 [32] A. MADRY, A. MAKELOV, L. SCHMIDT, D. TSIPRAS, AND A. VLADU, *Towards deep learning models resistant to*  
777 *adversarial attacks*, Proc. Int. Conf. Learn. Represent., 2018.
- 778 [33] S.-M. MOOSAVI-DEZFOOLI, A. FAWZI, AND P. FROSSARD, *Deepfool: A simple and accurate method to fool deep*  
779 *neural networks*, Proc. IEEE Conf. Comput. Vis. Pattern Recogn., 2016, pp. 2574–2582.
- 780 [34] L. MORONEY, *Rock, paper, scissors dataset*, 2019.

- 781 [35] A. NEACȘU, J.-C. PESQUET, AND C. BURILEANU, *EMG-based automatic gesture recognition using robust neural*  
782 *networks (extended version)*, Preprint submitted to ACM, 2022.
- 783 [36] S. NEUMAYER, A. GOUJON, P. BOHRA, AND M. UNSER, *Approximation of Lipschitz functions using deep spline*  
784 *neural networks*, *J. Math. Data Sci.*, vol. 5, 2023, pp. 306–322.
- 785 [37] S. PARKINSON, G. ONGIE, AND R. WILLETT, *Linear neural network layers promote learning single- and*  
786 *multiple-index models*, arXiv:2305.15598, 2023.
- 787 [38] P. PAULI, A. KOCH, J. BERBERICH, P. KOHLER, AND F. ALLGÖWER, *Training robust neural networks using*  
788 *Lipschitz bounds*, *IEEE Control Systems Letters*, vol. 6, IEEE, 2021, pp. 121–126.
- 789 [39] M. PINTOR, F. ROLI, W. BRENDEL, AND B. BIGGIO, *Fast minimum-norm adversarial attacks through adaptive*  
790 *norm constraints*, *Proc. Ann. Conf. Neur. Inform. Proc. Syst.*, vol. 34, 2021, pp. 20052–20062.
- 791 [40] A. RAGHUNATHAN, J. STEINHARDT, AND P. LIANG, *Certified defenses against adversarial examples*, *Proc. Int.*  
792 *Conf. Learn. Represent.*, 2018.
- 793 [41] A. RAGHUNATHAN, J. STEINHARDT, AND P. S. LIANG, *Semidefinite relaxations for certifying robustness to*  
794 *adversarial examples*, *Proc. Ann. Conf. Neur. Inform. Proc. Syst.*, vol. 31, 2018.
- 795 [42] J. RONY, L. G. HAFEMANN, L. S. OLIVEIRA, I. B. AYED, R. SABOURIN, AND E. GRANGER, *Decoupling direction*  
796 *and norm for efficient gradient-based  $\ell_2$  adversarial attacks and defenses*, *Proc. IEEE Conf. Comput. Vis.*  
797 *Pattern Recogn.*, 2019, pp. 4322–4330.
- 798 [43] S. SABOUR, N. FROSST, AND G. E. HINTON, *Dynamic routing between capsules*, *Proc. Ann. Conf. Neur. Inform.*  
799 *Proc. Syst.*, vol. 30, 2017, pp. 3856–3866.
- 800 [44] H. SALMAN, J. LI, I. RAZENSHTEYN, P. ZHANG, H. ZHANG, S. BUBECK, AND G. YANG, *Provably robust deep*  
801 *learning via adversarially trained smoothed classifiers*, *Proc. Ann. Conf. Neur. Inform. Proc. Syst.*, vol. 32,  
802 2019, pp. 11292–11303.
- 803 [45] M. SERRURIER, F. MAMALET, A. GONZÁLEZ-SANZ, T. BOISSIN, J.-M. LOUBES, AND E. DEL BARRIO, *Achieving*  
804 *robustness in classification using optimal transport with hinge regularization*, *Proc. IEEE Conf. Comput.*  
805 *Vis. Pattern Recogn.*, 2021, pp. 505–514.
- 806 [46] A. SHAFABI, P. SAADATPANAH, C. ZHU, A. GHIASI, C. STUDER, D. JACOBS, AND T. GOLDSTEIN, *Adversarially*  
807 *robust transfer learning*, *Proc. Int. Conf. Learn. Represent.*, 2020.
- 808 [47] S. H. SILVA AND P. NAJAFIRAD, *Opportunities and challenges in deep learning adversarial robustness: A survey*,  
809 arXiv:2007.00753, 2020.
- 810 [48] A. SINHA, H. NAMKOONG, AND J. DUCHI, *Certifiable distributional robustness with principled adversarial*  
811 *training*, *Proc. Int. Conf. Learn. Represent.*, 2018.
- 812 [49] C. SZEGEDY, W. ZAREMBA, I. SUTSKEVER, J. BRUNA, D. ERHAN, I. GOODFELLOW, AND R. FERGUS, *Intriguing*  
813 *properties of neural networks*, *Proc. Int. Conf. Learn. Represent.*, 2014.
- 814 [50] Y. TAIGMAN, M. YANG, M. RANZATO, AND L. WOLF, *Deepface: Closing the gap to human-level performance in*  
815 *face verification*, *Proc. IEEE Conf. Comput. Vis. Pattern Recogn.*, 2014, pp. 1701–1708.
- 816 [51] D. TSIPRAS, S. SANTURKAR, L. ENGSTROM, A. TURNER, AND A. MADRY, *There is no free lunch in adversarial*  
817 *robustness (but there are unexpected benefits)*, arXiv:1805.12152, 2018.
- 818 [52] P. VAIDYANATHAN, *Multirate Systems and Filter Banks*, Prentice Hall (NJ), 1993.
- 819 [53] A. VIRMAUX AND K. SCAMAN, *Lipschitz regularity of deep neural networks: analysis and efficient estimation*,  
820 *Proc. Ann. Conf. Neur. Inform. Proc. Syst.*, vol. 31, 2018, pp. 3839–3848.
- 821 [54] Q. WANG, M. A. POWELL, A. GEISA, E. BRIDGEFORD, AND J. T. VOGELSTEIN, *Why do networks have*  
822 *inhibitory/negative connections?*, arXiv:2208.03211, (2022).
- 823 [55] Y. WANG, D. ZOU, J. YI, J. BAILEY, X. MA, AND Q. GU, *Improving adversarial robustness requires revisiting*  
824 *misclassified examples*, *Proc. Int. Conf. Learn. Represent.*, 2020.
- 825 [56] C. WEI AND J. Z. KOLTER, *Certified robustness for deep equilibrium models via interval bound propagation*,  
826 *Proc. Int. Conf. Learn. Represent.*, 2021.
- 827 [57] R. ZHAI, C. DAN, D. HE, H. ZHANG, B. GONG, P. RAVIKUMAR, C.-J. HSIEH, AND L. WANG, *MACER: Attack-free*  
828 *and scalable robust training via maximizing certified radius*, *Proc. Int. Conf. Learn. Represent.*, 2020.
- 829 [58] B. ZHANG, D. JIANG, D. HE, AND L. WANG, *Rethinking Lipschitz neural networks and certified robustness: A*  
830 *Boolean Function Perspective*, *Proc. Ann. Conf. Neur. Inform. Proc. Syst.*, vol. 35, 2022, pp. 19398–19413.
- 831 [59] H. ZHANG, Y. YU, J. JIAO, E. XING, L. EL GHAOU, AND M. JORDAN, *Theoretically principled trade-off between*  
832 *robustness and accuracy*, *Proc. Int. Conf. Machine Learn.*, 2019, pp. 7472–7482.