Invertibility of ReLU-layers: A Practical Approach

Hannah Eckert [1][a], Daniel Haider[1][b], Martin Ehler [2][c] and Peter Balazs [1][d]
[1]*Acoustics Research Institute, Austrian Academy of Sciences, Dominikanerbastei 16, Vienna, Austria*
[2]*Faculty of Mathematics, University of Vienna, Oskar-Morgenstern-Platz 1, Vienna, Austria*
{*hannah.eckert, daniel.haider, peter.balazs*}*@oeaw.ac.at, martin.ehler@univie.ac.at*

Keywords: Invertible Neural Networks, Reconstruction, Frame Theory, Stability, Interpretability

Abstract: Invertibility in machine learning models is a pivotal feature that bridges the gap between model complexity and interpretability. For ReLU-layers, the practical verification of invertibility has been shown to be a difficult task that still remains unsolved. Recently, a frame theoretic condition has been proposed to verify invertibility on an open or convex set, however, the computations for this condition are computationally infeasible in high dimensions. As an alternative, we propose an algorithm that stochastically samples the dataset to approximately verify the above condition for invertibility and can be efficiently implemented even in high dimensions. We use the algorithm to monitor invertibility and to enforce it during training in standard classification tasks.

1 Introduction

Keeping models powerful while designing them in a transparent and interpretable way is one of the most significant challenges in modern machine learning (Fan et al., 2020). In this context, understanding the underlying mechanisms by which a model makes its predictions has become a focal point of research (Samek et al., 2019). One of the fundamental properties associated with the interpretability of a model is *invertibility*. When a model is invertible, each prediction can be traced back to its source inputs, providing the ability to decipher the decision-making process (Kothari et al., 2021). This traceability is crucial for diagnosing model behavior, identifying biases, and ensuring accountability (Lipton, 2016). Invertibility offers a multitude of avenues for interpretability. One such avenue is the potential for the systematic perturbation of single features of the intermediate representations, mapping them back to the data space, and the subsequent interpretation of the reconstructed input with perturbed features. However, unless the model is designed to be invertible, the practical verification is generally an ill-posed problem.

Given the prevalence of ReLU-layers as building blocks in neural networks, understanding how information flows through can be essential to understanding the whole model architecture and enhancing its interpretability. So, in this paper we focus on putting existing theoretical results on the invertibility of ReLU-layers (Haider et al., 2024; Maillard et al., 2023; Puthawala et al., 2022) into practice. Our approach to this can be described as follows: For any given weight matrix W and bias vector α we want to assess if the associated ReLU-layer is invertible on some set K, that contains data points of interest. By appropriate sampling from (a part of) this dataset, instead of K, we compute an approximation of a bias vector α^* which is maximal with the property that the associated ReLU-layer is invertible on K. Hence, if $\alpha \leq \alpha^*$ (entry-wise) we can deduce that the original ReLU-layer is invertible, too. We use this to promote the invertibility of ReLU-layers by enforcing $\alpha \leq \alpha^*$ during training via regularization. With these experiments, we explore the trade-off between model expressivity versus information preservation of ReLU-layers.

This manuscript is organized as follows. In Section 2 we present frame theory as theoretical backbone of our approach and derive a sufficient condition for the invertibility of ReLU-layers on a given set. Section 3 introduces an algorithm that satisfies the condition, which is demonstrated in practical applications in Section 4.

[a] https://orcid.org/0009-0006-2987-650X
[b] https://orcid.org/0000-0001-8012-5521
[c] https://orcid.org/0000-0002-3247-6279
[d] https://orcid.org/0000-0003-4939-0831

2 Theoretical Invertibility of ReLU-layers

Frame theory is a mathematical paradigm that deals with stable and invertible representations (Christensen, 2003). We adopt the basic notions to study the invertibility of ReLU-layers via the *injectivity*, i.e., invertibility "from the left" of the mappings that provide these representations.

2.1 Introduction to Frame Theory

Definition 2.1 (Frames)**.** A matrix $W \in \mathbb{R}^{m\times n}$ with row vectors $w_i \in \mathbb{R}^n$ for $i = 1,\dots,m$ induces a frame for $\mathbb{R}^n$ if there are constants $A, B > 0$ such that

$$A\|x\|^2 \leq \sum_{i=1}^{m} |\langle x, w_i\rangle|^2 \leq B\|x\|^2 \tag{1}$$

holds for all $x \in \mathbb{R}^n$. The vectors w_i are called the frame elements and the collection $(w_i)_{i=1,\dots,m}$ is called a frame.

A frame is equivalent to a spanning set (Casazza and Kutyniok, 2012), hence $m \geq n$ has to hold, and the constants A, B determine the numerical stability of the representation of x in terms of the inner products $(\langle x, w_i\rangle)_{i=1,\dots,m}$.

Definition 2.2 (Analysis Operator)**.** The analysis operator associated with the matrix $W \in \mathbb{R}^{m\times n}$ is given by the mapping $C : \mathbb{R}^n \to \mathbb{R}^m$ defined by

$$x \mapsto (\langle x, w_i\rangle)_{i=1,\dots,m}. \tag{2}$$

We call $(\langle x, w_i\rangle)_{i=1,\dots,m}$ the representation of x, or frame coefficients. The analysis operator is equivalent to matrix multiplication, i.e. $C(x) = Wx$.

Lemma 2.3. *The analysis operator $C : \mathbb{R}^n \to \mathbb{R}^m$ for the matrix W is injective if and only if the matrix $W \in \mathbb{R}^{m\times n}$ induces a frame.*

In the context of neural network layers, we may interpret the above lemma as that the frame-inducing property of a weight matrix $W \in \mathbb{R}^{m\times n}$ is characterizing for the injectivity (i.e., invertibility) of the corresponding linear layer. Note that for linear layers with i.i.d. randomly initialized weights that map into a higher dimensional space ($m \geq n$) the frame property holds true with probability one (Blumensath and Davies, 2009).

Note that the smallest possible bound B in (1) coincides with the upper, and the largest possible A with the lower Lipschitz bound of the analysis operator (i.e., the upper Lipschitz bound of the pseudo-inverse).

2.2 Frame Theory for ReLU-layers

It turns out that it is possible to derive a characterization of the invertibility of ReLU-layers on a set $K \subseteq \mathbb{R}^n$ using Lemma 2.3 (Haider et al., 2024). Let us first define a ReLU-layer in this spirit of the analysis operator as follows.

Definition 2.4 (ReLU-layer)**.** Let $W \in \mathbb{R}^{m\times n}$, $\alpha \in \mathbb{R}^m$, and $C(x)_i = \langle x, w_i\rangle$ denote the i-th component of the analysis operator for W. The mapping $\mathrm{ReLU}_{W,\alpha} : \mathbb{R}^n \to \mathbb{R}^m$ defined by

$$x \mapsto (\max(0, C(x)_i - \alpha_i))_{i=1,\dots,m}, \tag{3}$$

is the ReLU-layer with weight matrix W and bias vector α.

More explicitly, the image of x under a ReLU-layers can be written as

$$\mathrm{ReLU}_{W,\alpha}(x)_i = \begin{cases} C(x)_i - \alpha_i & \text{if } C(x)_i \geq \alpha_i \\ 0 & \text{if } C(x)_i < \alpha_i. \end{cases} \tag{4}$$

Since C is injective if and only if W induces a frame it is clear that the invertibility of $\mathrm{ReLU}_{W,\alpha}$ depends on how the ReLU function affects the frame coefficients of x. This motivates the following definition.

Definition 2.5 (Active Frame Elements)**.** A frame element w_i is called *active* for x and α if $C(x)_i \geq \alpha_i$. The index set of all active frame elements for given x and α is denoted by

$$I_x^\alpha := \{i \in I : C(x)_i \geq \alpha_i\}. \tag{5}$$

Using this notion, we consider frames which satisfy that for all $x \in K$ the active frame elements form a frame:

Definition 2.6 (α-rectifying)**.** A matrix $W \in \mathbb{R}^{m\times n}$ is called α-rectifying on $K \subseteq \mathbb{R}^n$ if for all $x \in K$ the active frame elements for x and α form a frame. In particular, we call W α-rectifying for x if it is α-rectifying on the set $\{x\}$.

This definition is naturally connected to the frame theoretic condition for the invertibility of a ReLU-layer on some open or convex set K:

Theorem 2.7 ((Haider et al., 2024))**.** *If $W \in \mathbb{R}^{m\times n}$ is α-rectifying on an open or convex set $K \subseteq \mathbb{R}^n$ then $\mathrm{ReLU}_{W,\alpha}$ is invertible on K.*

This allows us to approach the invertibility of ReLU-layers by verifying that the weight matrix W is α-rectifying on some open or convex set K. We will make an additional assumption that further simplifies the problem.

Definition 2.8 (Full Spark (Alexeev et al., 2012))**.** A matrix $W \in \mathbb{R}^{m\times n}$ is called full spark if each subcollection of n row vectors forms a frame.

With this assumption, the verification of the α-rectifying property can be reduced to a counting argument.

Lemma 2.9. *Let $W \in \mathbb{R}^{m\times n}$ be full spark and $K \subseteq \mathbb{R}^n$. If the cardinality of the set of active frame elements satisfies $|I_x^\alpha| \geq n$ for all $x \in K$ then W is α-rectifying on K.*

Similarly as for the frame property, matrices with entries that are i.i.d. samples from an absolutely continuous probability distribution are known to be full spark with probability one (Blumensath and Davies, 2009). Although not proven, it is reasonable to assume that the full spark property is also preserved during gradient steps in training. As a consequence, a full spark assumption is very natural in the context of neural network training. Moreover, it is also one of the fundamental ingredients for the algorithm that we propose to approximately check the α-rectifying property in practice.

3 Data Driven Monte Carlo Bias Estimation

We now introduce a method called Monte Carlo Bias Estimation (**MCBE**) to approximately check the α-rectifying property of a weight matrix $W \in \mathbb{R}^{m\times n}$ on finitely many data points. (Approximate) invertibility of the associated ReLU-layer can be concluded from Theorem 2.7 only if all the points lie within an open or convex set K where W is α-rectifying. **MCBE** circumvents this by addressing the problem from a stochastic point of view: By drawing data points according to an approximation of the distribution of a given data set we conclude the α-rectifying property of W on all data that follow the same distribution. This shall provide a sufficient condition for the approximate invertibility of $\mathrm{ReLU}_{W,\alpha}$ for all data points that are close to the initial ones with high probability.

3.1 Motivation and Overview

For a set of points $X_N = \{x_1, \dots, x_N\}$ in $\mathbb{R}^n$ the output of **MCBE** is a bias vector α^* such that W is α-rectifying on X_N for all $\alpha \leq \alpha^*$ with high probability. Here, "$\leq$" is meant in a entry-wise sense, i.e., $\alpha_i \leq \alpha_i^*$ for all $i = 1, \dots, m$. To obtain such a maximal bias we iteratively compute biases $\alpha(x_i)$ such that W is $\alpha(x_i)$-rectifying on $\{x_i\}$ and update them in a suitable manner. According to this idea, the following theorem provides the theoretical foundation for **MCBE**.

Theorem 3.1 (Maximal Bias)**.** *Let $W \in \mathbb{R}^{m\times n}$ be full spark, and let $\alpha^*(x)$ for $x \in \mathbb{R}^n$ be given as*

$$\alpha(x)_j := \begin{cases} C(x)_j & \text{if } j \in J_x \\ \infty & \text{else}, \end{cases} \tag{6}$$

where J_x denotes the index set corresponding to the n largest entries of the analysis operator. Then W is α-rectifying on $\{x\}$ if $\alpha \leq \alpha(x)$.

Moreover, let $X_N = \{x_1, \dots, x_N\} \subset \mathbb{R}^n$ and $\alpha^*(X_N)$ be given as

$$\alpha^*(X_N)_j = \min_{i=1,\dots,N} \alpha(x_i)_j. \tag{7}$$

Then W is α-rectifying on X_N if $\alpha \leq \alpha^*(X_N)$.

Proof. Let $\alpha \leq \alpha(x)$, then in particular,

$$\alpha_j \leq \alpha(x) = C(x)_j \text{ for all } j \in J_x. \tag{8}$$

Since J_x contains n elements, the frame has n active frame elements and, therefore, is α-rectifying on $\{x\}$ by Definitions 2.6 and 2.8. This shows the first part. For the moreover part, let $\alpha \leq \alpha^*(X_N)$ then

$$\alpha \leq \alpha^*(X_N) \implies \alpha \leq \alpha(x_i)$$

for all $i = 1, \dots, N$. Consequently, W is α-rectifying on $\{x_i\}$ for all i and therefore also on X_N. □

3.2 Implementation

The basic functionality of the algorithm for **MCBE** can be summarized as follows. Let k denote the k-th iteration.

$$\begin{aligned} k = 0: &\quad (\alpha_j^*)^0 = \infty \text{ for all } j \\ 0 < k < N: &\quad (\alpha_j^*)^k = \min\{(\alpha_j^*)^{k-1}, \alpha(x_k)_j\} \\ &\qquad \text{for } j \in J_{x_k} \text{ and } x_k \in X_N \\ k = N: &\quad (\alpha^*)^N =: \alpha^*(X_N) \end{aligned}$$

For a more detailed pseudo-code of the algorithm we refer to the appendix. As by Theorem 3.1, W is $\alpha^*(X_N)$-rectifying on X_N. To conclude invertibility of $\mathrm{ReLU}_{W,\alpha}$ on a specific data set of interest we derive the sampling set X_N from a given training data set using randomized smoothing (Cohen et al., 2019). It is important to note that this approach circumvents the problem that the number of necessary samples, to cover a set K sufficiently dense, explodes in high dimensions (Reznikov and Saff, 2016).

Definition 3.2 (Randomized Smoothing (Cohen et al., 2019))**.** Let $X^{(\text{train})} \subset K$ be a collection of M data points $x_1^{(\text{train})}, \dots, x_M^{(\text{train})}$ that all follow the same distribution on K. The procedure of uniformly drawing points from $X^{(\text{train})}$ and adding Gaussian noise

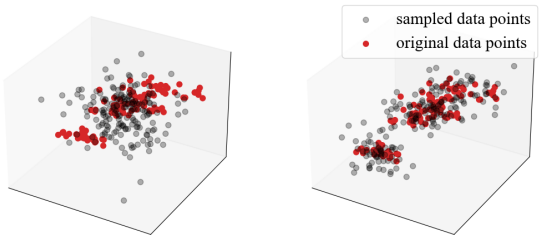

Figure 1: Illustration of randomized smoothing on Iris data. Dark red dots represent the original data points light gray dots sampled data. Left: Uniformly sampled data on the ball. Right: Samples obtained from randomized smoothing, respecting the distribution of data set well.

is called randomized smoothing of $X^{(\text{train})}$. Hence, a point x in the resulting sampling set is given by

$$x = x_j^{(\text{train})} + \varepsilon \text{ with } \varepsilon \sim \mathcal{N}(0, \sigma \mathbb{I}_n)$$

for some $\sigma \in \mathbb{R}$, and j chosen uniformly.

The sampling set obtained from randomized smoothing is an augmented data set X_N that follows approximately the same distribution as the training data. This makes it a suitable choice for the procedure described in Theorem 3.1. Figure 1 shows how this sampling methods looks like on a three-dimensional version of the Iris dataset.

4 Practical invertibility of ReLU-layers

With the possibility to check the approximate invertibility of ReLU-layers we can directly study the circumstances under which ReLU-layers are invertible. In the following we demonstrate how **MCBE** can be used to study the correlation of the performance of a neural network model and the invertibility of a ReLU-layer as part of the model architecture. On the one hand, we aim to understand if invertibility is naturally occurring in learning a ReLU-layer, and one step further, whether enforcing its invertibility comes at the cost of the performance of the whole model. For all experiments, we solve a classification task on two standard benchmark datasets: Iris (Fisher, 1988), and MNIST (LeCun et al., 2010). For detailed descriptions of the model architectures and training procedures we refer to the appendix.

4.1 Monitor Invertibility

Our first experimental goal is to monitor the invertibility of the first ReLU-layer in the respective models during training. We define the redundancy of a layer by its output dimension m divided by its input dimension n. We always compare three different redundancy configurations: $\frac{m}{n} = 2, 3, 9$.

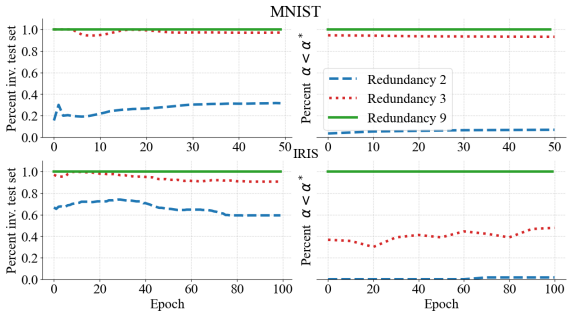

Figure 2: Invertibility of the first ReLU-layer of a neural net quantified in two ways. Left: The percentage of points of the test set, for which perfect reconstruction is possible. Right: The percentage of bias values α_i such that $\alpha_i \leq \alpha_i^*(X_N)$. Different color and line style indicate different redundancy settings.

We can quantify the invertibility of a ReLU-layer with bias vector α in two different ways. Let $X_N = \{x_1, \ldots, x_N\}$.

1. *Naively*: Measure the percentage of points $x \in X_N$ such that $|I_x^\alpha| \geq n$ holds. This can be interpreted as the percentage of points $x \in X_N$ for which the ReLU-layer is invertible, and therefore, reconstruction without loss is possible.
2. **MCBE:** Measure the percentage of the bias values α_i such that $\alpha_i \leq \alpha_i^*(X_N)$, which is the percentage of bias values that do not have to be changed to make the layer invertible.

We plot the results for both ways in Figure 2. It is crucial to acknowledge that the two measures represent distinct approaches to invertibility. Consequently, it is anticipated that there will be some discrepancy between the measures. We observe, that when using a ReLU-layer with redundancy two on MNIST (upper left in Fig. 2), only about 30% of the test data points can be reconstructed perfectly, and 10% of the bias values fulfill the proposed condition. Note that the jump from redundancy two to three has a significant effect in terms of invertibility, observed in both measures.

Moreover, across datasets and measures, we observe that training does not influence the invertibility of the layer significantly, seconding the comment in Sec. 2.2.

Backed by this experiment we assume that enforcing invertibility will not influence the performance of the model remarkably. We will test this assumption in Section 4.2.

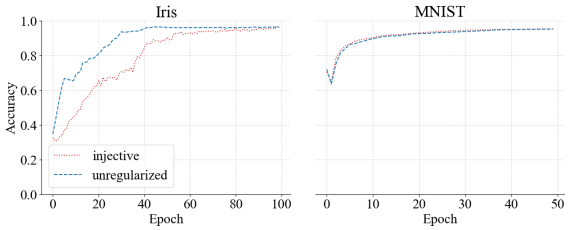

Figure 3: Classification accuracy of a neural net with an approximately invertible ReLU-layer (red dashed line) as the first layer, compared to non-regularized baseline during training. Accuracy is measured on the test set and the redundancy of the layers is $\frac{m}{n} = 2$.

4.2 Enforce Invertibility during Training

Our second experimental goal is to use the bias $\alpha^*(X_N)$ from **MCBE** in a regularization procedure where the ReLU-layer is gradually promoted to be invertible during training. We implement this by adding

$$\rho_{\text{inj}}(\alpha) = \left\| \begin{pmatrix} \max\{0, \alpha_1 - \alpha^*(X_N)_1\} \\ \vdots \\ \max\{0, \alpha_m - \alpha^*(X_N)_m\} \end{pmatrix} \right\|_2$$

to the learning objective. This term affects those bias values where $\alpha_i > \alpha^*(X_N)_i$ and has no effect, otherwise. In this sense, it penalizes non-invertibility.

Across the datasets, we found that enforcing invertibility can come with a trade-off with the performance of the model during training (Figure 3). For MNIST, this trade-off is even negligible. At the end of the training, both models achieved the same accuracy while the reconstruction error could be significantly reduced (Table 1). In Section 4.3 we go into more detail about reconstruction.

4.3 Reconstruction

The input $x \in \mathbb{R}^n$ of a ReLU-layer can be reconstructed from its output $z = \text{ReLU}_{W,\alpha}(x) \in \mathbb{R}^m$ by calculating the least-squares solution x^* of

$$W^+ x = z^+ + \alpha^+, \tag{9}$$

where W^+ is the matrix, whose row vectors are the row vectors w_i of W that satisfy $(\text{ReLU}_{W,\alpha}(x))_i > 0$. The bias α^+ and z^+ are defined analogously (Haider et al., 2023). If the ReLU-layer is invertible, the reconstruction is exact, i.e., $x = x^*$. In Table 1, we show the mean reconstruction errors on the test set with and without enforcing injectivity during training. The mean reconstruction error is defined as the Euclidean distance between the input x and the reconstructed input x^*. Although we do not obtain perfect reconstruction in our examples, the approximately invertible layers provide an error reduction by more than a factor of two. In Figure 4 we show a visual comparison of a reconstructed MNIST image using an approximately invertible ReLU-layer and a non-regularized one.

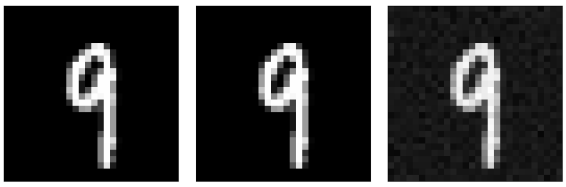

Figure 4: Original (left) and reconstructed MNIST images from the output of trained ReLU-layers. Mid: Approximately invertible. Right: Not invertible. The input dimension is n = 784 and the output dimension m = 1568.

Data set	approx. invertible	unregularized
IRIS	0.84	2.02
MNIST	0.79	1.63

Table 1: Mean reconstruction error measures on the test set with and without enforcing injectivity during training.

4.4 Stability Analysis

A standard metric for evaluating the stability of a neural network layer is via its upper Lipschitz bound, which represents the maximal factor by which distances of input vectors can be amplified by applying the layer. In the context of invertibility, the lower Lipschitz bound of the layer is of central interest since it corresponds to the upper Lipschitz bound of the inverse - provided that it exists. In this sense, a ReLU-layer $\text{ReLU}_{W,\alpha}$ is bi-Lipschitz stable if there are $A, B > 0$ such that

$$A||x - x'||^2 \leq ||\text{ReLU}_{W,\alpha}(x) - \text{ReLU}_{W,\alpha}(x')||^2 \leq B||x - x'||^2 \tag{10}$$

holds for all $x, x' \in \mathbb{R}^n$. The following has been shown in (Bruna et al., 2014).

Theorem 4.1 (ReLU Bi-Lipschitz Bounds)**.** *Let $W \in \mathbb{R}^{m \times n}$ and $\alpha \in \mathbb{R}^m$. For any $x \in \mathbb{R}^n$ let $W_{I_x^\alpha}(x)$ be defined as the matrix whose row vectors are the active frame elements of W. Let us further denote the smallest singular value of some matrix V as $\lambda_-(V)$ and the largest singular value as $\lambda_+(V)$. Then upper and lower Lipschitz bounds of $\text{ReLU}_{W,\alpha}$ are given as*

$$A_0 = \min_x \lambda_-(W_{I_x^\alpha}(x)),$$

$$B_0 = \max_x \lambda_+(W_{I_x^\alpha}(x)).$$

Moreover, $\text{ReLU}_{W,\alpha}$ is invertible if and only if $A > 0$.

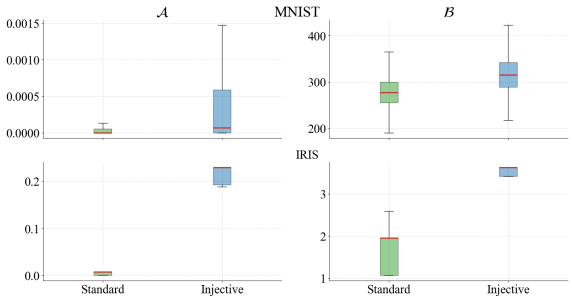

Figure 5: Sets $\mathcal{A}, \mathcal{B}$, defined such that high values in the set $\mathcal{A}$ indicate stability of the inverse mapping and low values in the set $\mathcal{B}$ indicate stability of the ReLU-layer.

Note that $\lambda_-(W_{I_x^\alpha}(x))$ and $\lambda_+(W_{I_x^\alpha}(x))$ correspond to the lower and upper frame bounds (1) of the subframe that is active for x and α.

To visualize the effect on the stability of enforcing injectivity with our described method in Figure 5 we plot the sets

$$\mathcal{A} = \{\lambda_-(W_{I_x^\alpha}(x)) \text{ for } x \in X^{(\text{test})}\}$$

and

$$\mathcal{B} = \{\lambda_+(W_{I_x^\alpha}(x)) \text{ for } x \in X^{(\text{test})}\}.$$

These sets contain the true Lipschitz bounds A_0 and B_0 as minimum and maximum, respectively. Low values in the set $\mathcal{B}$ indicate good stability behavior of the corresponding ReLU-layer in the classic sense, while high values in the set $\mathcal{A}$ indicate good stability behavior of the inverse mapping. Zero values in the set $\mathcal{A}$ correspond to points, where the ReLU-layer is locally not invertible. Across the datasets, we found, that enforcing injectivity reduces the stability of the ReLU-layer but increases the stability of the inverse.

4.5 Discussion

Our experiments demonstrate that approximate invertibility can be enforced by adding a term to the loss function that penalizes bias values that are larger than the estimated maximum. With this the reconstruction loss from the layer output can be decreased significantly without loss of accuracy at the end of training. However, there is a natural trade-off between the upper Lipschitz bound of the layer and the Lipschitz bound of its inverse mapping. Further research is needed to identify if the observed behavior is specific to the considered examples or more general.

5 Conclusion

This paper introduces a Monte Carlo-type method to numerically verify a sufficient condition for the invertibility of a ReLU-layer in a neural network. For any given weight matrix and dataset the algorithm calculates a bias α^* which is maximal with the property that any ReLU-layer with bias α is invertible if $\alpha \leq \alpha^*$. This can be used to study the information flow in ReLU-layers in an alternative manner by comparing α with α^* during training. Moreover, it can be used to enforce invertibility by penalizing bias values that exceed the maximal bias during training. We demonstrate these applications in two basic examples. Our findings are that the performance of the model is only slightly hindered by enforcing invertibility of their ReLU-layers, suggesting that invertible models can achieve comparable performance to their non-invertible counterparts.

Acknowledgment

The research presented in this paper is derived from the first author's Master's thesis submitted to the University of Vienna which contains a more comprehensive treatment of the subject matter. Financial support comes from the Austrian Science Fund (FWF) projects LoFT (P 34624) and NoMASP (P 34922). D. Haider is recipient of a DOC Fellowship of the Austrian Academy of Sciences at the Acoustics Research Institute (A 26355). We thank the reviewers for their time reviewing the paper and their feedback.

REFERENCES

Alexeev, B., Cahill, J., and Mixon, D. G. (2012). Full spark frames. *Journal of Fourier Analysis and Applications*, 18:1167–1194.

Blumensath, T. and Davies, M. E. (2009). Sampling theorems for signals from the union of finite-dimensional linear subspaces. *IEEE Transactions on Information Theory*, 55(4):1872–1882.

Bruna, J., Szlam, A., and Lecun, Y. (2014). Signal recovery from pooling representations. In *Proceedings of International Conference on Machine Learning (ICML)*, pages 1585–1598.

Casazza, P. G. and Kutyniok, G. (2012). *Finite frames: Theory and applications*. Springer.

Christensen, O. (2003). *An Introduction to Frames and Riezs Bases*. Birkhäuser.

Cohen, J., Rosenfeld, E., and Kolter, Z. (2019). Certified adversarial robustness via randomized smoothing. In *Proceedings of International Conference on Machine Learning (ICML)*, pages 1310–1320. PMLR.

Fan, F., Xiong, J., Li, M., and Wang, G. (2020). On interpretability of artificial neural networks: A survey. *IEEE Transactions on Radiation and Plasma Medical Sciences*, 5:741–760.

Fisher, R. A. (1988). Iris. UCI Machine Learning Repository.

Haider, D., Balazs, P., and Ehler, M. (2023). Convex geometry of ReLU-layers, injectivity on the ball and local reconstruction. In *Proceedings of International Conference on Machine Learning (ICML)*, pages 12339–12350.

Haider, D., Ehler, M., and Balazs, P. (2024). Injectivity of ReLU-layers: Perspectives from frame theory. *arXiv*, abs/2406.15856.

Kingma, D. P. and Ba, J. (2017). Adam: A method for stochastic optimization.

Kothari, K., Khorashadizadeh, A., de Hoop, M., and Dokmanić, I. (2021). Trumpets: Injective flows for inference and inverse problems. In de Campos, C. and Maathuis, M. H., editors, *Proceedings of the 37th Conference on Uncertainty in Artificial Intelligence*, volume 161 of *Proceedings of Machine Learning Research*, pages 1269–1278.

LeCun, Y., Cortes, C., and Burges, C. (2010). MNIST handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2.

Lipton, Z. (2016). The mythos of model interpretability. *Communications of the ACM*, 61.

Maillard, A., Bandeira, A. S., Belius, D., Dokmanić, I., and Nakajima, S. (2023). Injectivity of relu networks: perspectives from statistical physics. *arXiv*, abs/2302.14112.

Puthawala, M., Kothari, K., Lassas, M., Dokmanić, I., and de Hoop, M. (2022). Globally injective ReLU networks. *Journal of Machine Learning Research*, 23(105):1–55.

Reznikov, A. and Saff, E. B. (2016). The covering radius of randomly distributed points on a manifold. *International Mathematics Research Notices*, 2016(19):6065–6094.

Samek, W., Montavon, G., Vedaldi, A., Hansen, L. K., and Müller, K.-R., editors (2019). *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*. Springer Cham.

APPENDIX

6 The MCBE algorithm

This algorithm represents an extension of the one initially proposed by Haider et al. in (Haider et al., 2024).

Data: weight matrix $W \in \mathbb{R}^{m\times n}$, train data $X^{(\text{train})} \in \mathbb{R}^{M\times n}$, number of iterations s, for which α has to stay the same for convergence
Initialize $(\alpha^*(x_1,\ldots,x_N)_j)^0 = -\infty$ for $j = 1,\ldots,m$;
while $||(\alpha^*(x_1,\ldots,x_N))^{i-s} - (\alpha^*(x_1,\ldots,x_N))^i||$ **do**
  sample $x_i^{(\text{train})}$ from $X^{(\text{train})}$;
  sample ε_i from $\mathcal{N}(0,\sigma\mathbb{I}_n)$;
  set $x_i = x_i^{(\text{train})} + \varepsilon_i$;
  calculate J_{x_i};
  for $j \in J_{x_i}$ **do**
    $(\alpha^*(x_1,\ldots,x_N)_j)^i = \min\{(\alpha^*(x_1,\ldots,x_N)_j)^{i-1}, \langle x_i, w_j\rangle\}$
  end
  $i = i+1$
end

Algorithm 1: Monte Carlo Bias Estimation using randomized smoothing.

7 Neural Network Architectures and Training Details

Both datasets were normalized before further processing. In both cases an Adam optimizer (Kingma and Ba, 2017) with a learning rate of 0.01 was used.

7.1 Neural Network Architecture for Iris and redundancy $\frac{m}{n} \in \{2,3\}$ of the first layer

For the Iris data and redundancy $\frac{m}{n} \in \{2,3\}$, we used two fully connected layers with ReLU activation function followed by an output layer with Softmax activation function.

1. FC1: m units, ReLU activation
2. FC2: 27 units, ReLU activation
3. Output Layer: 3 units, Softmax activation

7.2 Neural Network Architecture for Iris and redundancy $\frac{m}{n} \in \{2,3\}$ of the first layer

For the Iris data and redundancy $\frac{m}{n} \in \{2,3\}$, we used one fully connected layer with ReLU activation function followed by an output layer with Softmax activation function.

1. FC1: 27 units, ReLU activation
2. Output Layer: 3 units, Softmax activation

7.3 Neural Network Architecture for MNIST

For the MNIST data, we used one fully connected layers with ReLU activation function followed by an output layer with Softmax activation function.

1. FC1: m units, ReLU activation
2. Output Layer: 10 units, Softmax activation