FEATURE-WISE LIPSCHITZ NEURAL NETWORK

Anonymous authors
Paper under double-blind review

ABSTRACT

Existing Lipschitz-constrained neural networks typically impose a single global bound on input sensitivity. When sensitivity requirements differ across features and apply only to a selected subset, a global bound can unnecessarily constrain the remaining features and limit how accurately the target function can be approximated. We therefore propose Feature-wise Lipschitz Neural Networks (FLNN), an architecture that imposes distinct sensitivity bounds on selected features while leaving the remaining features unconstrained. The architecture uses controlled and uncontrolled branches that interact across layers while preserving the prescribed sensitivity bounds throughout training. We establish feature-wise Lipschitz guarantees and prove universal approximation of continuous functions satisfying the same constraints on compact domains. A coordinate-wise input transformation further extends the sensitivity guarantees to bounds that vary across regions within the domain of each controlled feature. Experiments on synthetic and real-world datasets demonstrate constraint satisfaction and assess predictive performance under the prescribed sensitivity bounds.

1 INTRODUCTION

Neural networks are widely used to model complex nonlinear relationships in scientific and engineering applications (Karniadakis et al., 2021). Beyond predictive accuracy, however, many such applications also require learned models to respect known structural or physical properties of the underlying system (Willard et al., 2022). One important aspect of such functional behavior is input-output sensitivity, which can be naturally characterized by Lipschitz continuity as a bound on how much the output can change in response to changes in the input. Controlling the Lipschitz constant of neural networks has therefore been widely studied to improve adversarial robustness (Cisse et al., 2017) and provide certified robustness guarantees (Tsuzuku et al., 2018), and stabilize the training of generative adversarial networks (Miyato et al., 2018). The Lipschitz control has also been studied in connection with training stability and generalization (Delattre et al., 2023).

Most Lipschitz-constrained neural networks control sensitivity through a global bound over the input space, providing a single worst-case guarantee for the network (Anil et al., 2019). While this formulation is well suited to settings where input perturbations are treated uniformly, it is less appropriate when sensitivity requirements differ across input features. Prior knowledge may apply only to selected features, leaving others unconstrained, and the appropriate sensitivity bounds may also differ among the constrained features. For example, consider a scalar-valued function $f(x_1, x_2)$ with sensitivity limits $L_1 < L_2$, as illustrated in Figure 1. A global Lipschitz bound of L_1 under the Euclidean norm controls the sensitivity to x_1, but also restricts variation with respect to x_2 more than required. Increasing this bound allows greater variation with respect to x_2, but also permits sensitivity in the x_1 direction to exceed L_1. This limitation remains even when the global Lipschitz constant is known exactly.

To address this limitation, we consider partial and feature-wise Lipschitz constraints that impose separate sensitivity bounds only on selected input features. This formulation incorporates prior knowledge about feature sensitivity while preserving flexibility with respect to the remaining features. We further extend this formulation to allow bounds that vary across regions of individual feature domains.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

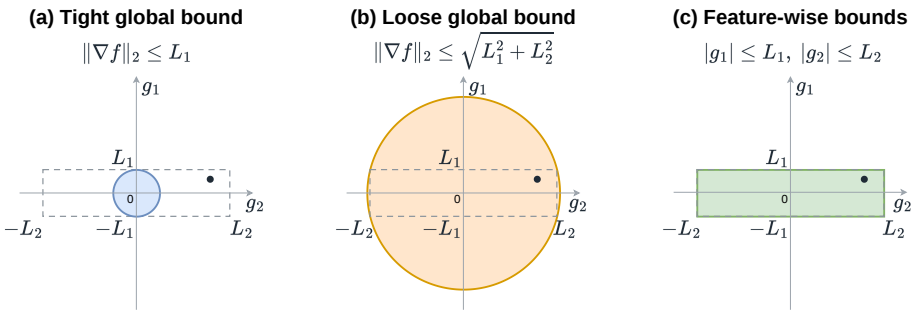

Figure 1: Global vs. Feature-wise bound for two features with limits $L_1 < L_2$, where $g_i = \partial f / \partial x_i$. The black dot marks partial derivatives $g_i = \partial f / \partial x_i$ satisfying both limits.

2 Literature Review

Prior knowledge about functional behavior can be incorporated into neural networks by restricting the class of functions they represent. Existing shape-constrained approaches impose qualitative properties such as monotonicity or convexity, including constraints that apply only to selected input features (You et al., 2017; Runje & Shankaranarayana, 2023; Kim & Lee, 2024; Amos et al., 2017). Shape constraints primarily regulate the direction of change or the shape of a function, while Lipschitz constraints bound how rapidly the output can vary with the input. For a feedforward network $f = f_1 \circ \cdots \circ f_L$ composed of L layer functions f_l, the Lipschitz constant $\mathrm{Lip}(\cdot)$ can be bounded as

$$\mathrm{Lip}(f) \le \prod_{l=1}^{L} \mathrm{Lip}(f_l), \tag{1}$$

which motivates controlling the Lipschitz constant of each layer. Spectral normalization bounds the spectral norm of each weight matrix (Miyato et al., 2018), while induced operator-norm constraints extend this idea to different norm geometries (Gouk et al., 2021). Subsequent work has improved the expressiveness and optimization of globally Lipschitz networks through activations that preserve gradient norms, orthogonality-based parameterizations, and tighter Lipschitz layer constructions (Anil et al., 2019; Prach & Lampert, 2022; Araujo et al., 2023; Wang & Manchester, 2023). More recent architectures have further improved the capacity and scalability of globally Lipschitz models (Hu et al., 2023; Lai et al., 2025; Hu et al., 2026). Despite these advances, these methods generally impose a global Lipschitz constraint over the full input space. Such a global formulation does not directly accommodate sensitivity requirements that apply only to selected input features or differ across features.

Partial and Feature-wise Sensitivity Constraints A single global Lipschitz constant may be too coarse when input features require distinct levels of sensitivity control. Multivariate Lipschitz analysis characterizes this limitation by defining partial Lipschitz constants for individual features or groups of features (Gupta et al., 2022). However, this framework mainly serves to characterize the sensitivity of neural networks rather than construct models.

Several architectures move toward feature-selective constraints. Lipschitz Monotonic Networks (LMNs) combine a globally Lipschitz backbone with monotonicity constraints on selected inputs (Nolte et al., 2023). Their ℓ_1-based formulation separates the directional derivatives across input dimensions. However, the Lipschitz constraint itself remains global over the full input vector.

Partially Bi-Lipschitz Neural Networks (PBLNNs) impose bi-Lipschitz constraints on a selected subset of input features, providing a direct form of partial sensitivity control (Kinoshita & Toyoizumi, 2024). However, their basic formulation imposes bi-Lipschitz constraints jointly over the selected features. Table 1 summarizes how existing models and the proposed architecture differ in the form of sensitivity constraints. The columns indicate whether bounds can differ across features, whether constraints apply separately to each coordinate, and whether some features can remain unconstrained.

Table 1: Comparison of Lipschitz-based methods for partial sensitivity control. ▲ denotes support through an extension described in the original work rather than the basic formulation.

Method	Feature-specific bounds	Separate coordinate bounds	Unconstrained features
Global ℓ_2 LNNs (Miyato et al., 2018)	✗	✗	✗
Global ℓ_1 LNNs (Gouk et al., 2021)	✗	✓	✗
LMN (Nolte et al., 2023)	▲	✓	✗
PBLNN (Kinoshita & Toyoizumi, 2024)	▲	✗	✓
FLNN (ours)	✓	✓	✓

3 PRELIMINARIES

Lipschitz continuity For $p \in [1, \infty]$, let $\|\cdot\|_p$ denote the ℓ_p norm. A scalar-valued function $f : \mathcal{A} \to \mathbb{R}$, defined on $\mathcal{A} \subseteq \mathbb{R}^q$, is L-Lipschitz with respect to the ℓ_p norm if

$$|f(\mathbf{u}) - f(\mathbf{v})| \leq L\|\mathbf{u} - \mathbf{v}\|_p, \qquad \forall \mathbf{u}, \mathbf{v} \in \mathcal{A}. \tag{2}$$

We denote the Lipschitz constant of f with respect to the ℓ_p norm by $\mathrm{Lip}_p(f)$.

Induced matrix norms For a matrix W, vector $\mathbf{v}$ and $p, q \in [1, \infty]$, define

$$\|W\|_{p\to q} \coloneqq \sup_{\mathbf{v}\neq\mathbf{0}} \frac{\|W\mathbf{v}\|_q}{\|\mathbf{v}\|_p}. \tag{3}$$

This norm measures how much the matrix can increase a vector norm, using ℓ_p at the input and ℓ_q at the output. For $p = q$, it is the usual induced ℓ_p matrix norm, whereas for $p \neq q$, we refer to it as a mixed induced norm. The two cases used here are

$$\|W\|_{1\to\infty} = \max_{i,j} |W_{ij}|, \quad \|W\|_{\infty\to\infty} = \max_i \sum_j |W_{ij}|, \tag{4}$$

where i and j index rows and columns. Thus the first norm is the largest absolute entry, whereas the second is the largest absolute row sum.

Problem Setup A global Lipschitz bound controls input sensitivity jointly across all features. In this work, we consider scalar-valued regression with a separate upper bound on input sensitivity specified for each feature in a selected group. Let S and $\bar{S}$ denote the controlled and uncontrolled feature groups of dimensions d_S and $d_{\bar{S}}$, respectively, with total input dimension $d = d_S + d_{\bar{S}}$. Throughout, we write $[n] \coloneqq \{1, \ldots, n\}$ for any positive integer n. For each $i \in [d_S]$, let $\mathcal{X}_i \subseteq \mathbb{R}$ denote the coordinate domain of the i-th controlled feature. We assume that the input domain has the product structure

$$\mathcal{X} = \mathcal{X}_S \times \mathcal{X}_{\bar{S}}, \quad \mathcal{X}_S = \prod_{i=1}^{d_S} \mathcal{X}_i, \tag{5}$$

where $\mathcal{X}_{\bar{S}} \subseteq \mathbb{R}^{d_{\bar{S}}}$ is the domain of the uncontrolled features. We write each input vector $\mathbf{x} \in \mathcal{X}$ as

$$\mathbf{x} = (\mathbf{x}_S, \mathbf{x}_{\bar{S}}), \quad \mathbf{x}_S = (x_i)_{i=1}^{d_S}. \tag{6}$$

We denote the unknown target function by $f^* : \mathcal{X} \to \mathbb{R}$ and the network predictor by $\hat{f}_\theta : \mathcal{X} \to \mathbb{R}$, where θ denotes all trainable network parameters. We omit the subscript θ when no ambiguity arises.

For each controlled coordinate i, let $L_i \geq 0$ denote its sensitivity bound. This bound limits how much the output can change when only the i-th feature varies. Specifically, for all inputs $\mathbf{x}$ and all $t \in \mathbb{R}$ with $\mathbf{x} + t\mathbf{e}_i \in \mathcal{X}$,

$$|f(\mathbf{x} + t\mathbf{e}_i) - f(\mathbf{x})| \leq L_i|t|, \tag{7}$$

where $\mathbf{e}_i$ is the standard basis vector corresponding to the i-th controlled coordinate. Equivalently, f is L_i-Lipschitz with respect to x_i, uniformly over the remaining inputs. We refer to this property as feature-wise Lipschitz continuity on the controlled inputs and define the corresponding function class as

$$\mathcal{F}_{\mathrm{ctrl}} \coloneqq \{f : \mathcal{X} \to \mathbb{R} \text{ such that } f \text{ satisfies (7) } \forall i \in [d_S]\}, \tag{8}$$

and our objective is to approximate f^* using a predictor $\hat{f} \in \mathcal{F}_{\mathrm{ctrl}}$. These sensitivity bounds apply only to the controlled features, while no analogous bounds are imposed on the uncontrolled features. By definition, every $f \in \mathcal{F}_{\mathrm{ctrl}}$ satisfies $|\partial f(\mathbf{x})/\partial x_i| \leq L_i$ for each controlled coordinate i, wherever the corresponding partial derivative exists.

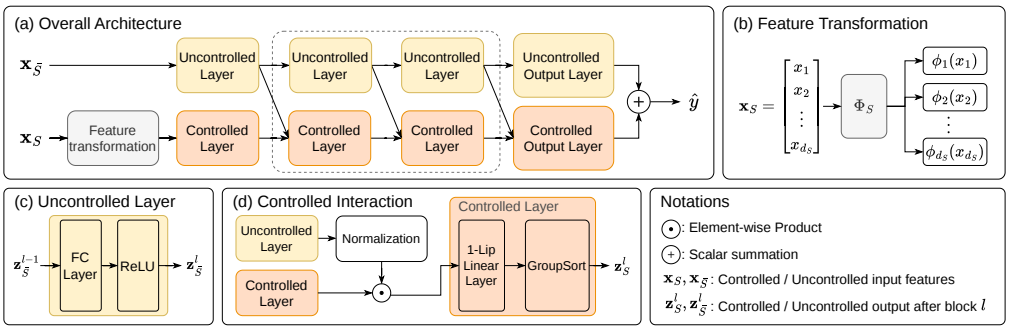

Figure 2: FLNN Model Architecture. The first controlled weight matrix satisfies a $1 \to \infty$ norm constraint. Subsequent controlled matrices, including the scalar-output matrix, satisfy $\infty \to \infty$ norm constraints.

4 Methodology

We first develop the proposed method with a constant sensitivity bound for each controlled feature and then extend it to bounds that vary across the corresponding feature domain. For fixed uncontrolled inputs, the mixed-norm parameterization ensures that the ℓ_∞ distance between controlled hidden representations does not exceed the ℓ_1 distance between transformed controlled inputs. This preserves a separate sensitivity bound for each controlled feature.

4.1 Feature-wise Control through Input Transformation

The prescribed feature-wise bounds can be incorporated through a coordinate-wise rescaling of the controlled input. For each controlled feature, define

$$\phi_i(x_i) \coloneqq L_i x_i. \tag{9}$$

Collecting these scalar transformations yields

$$\Phi_S : \mathcal{X}_S \to \mathbb{R}^{d_S}, \quad \Phi_S(\mathbf{x}_S) \coloneqq (\phi_i(x_i))_{i=1}^{d_S}. \tag{10}$$

Thus, Φ_S independently rescales each controlled feature according to its prescribed bound, while $\mathbf{x}_{\bar{S}}$ remains unchanged. Suppose that the predictor satisfies

$$|\hat{f}(\mathbf{x}_S, \mathbf{x}_{\bar{S}}) - \hat{f}(\mathbf{x}'_S, \mathbf{x}_{\bar{S}})| \leq \|\Phi_S(\mathbf{x}_S) - \Phi_S(\mathbf{x}'_S)\|_1 \tag{11}$$

for all controlled inputs $\mathbf{x}_S, \mathbf{x}'_S$ and every fixed uncontrolled input $\mathbf{x}_{\bar{S}}$. When $\mathbf{x}_S$ and $\mathbf{x}'_S$ differ only in coordinate i, the right-hand side equals $L_i|x_i - x'_i|$, establishing the prescribed feature-wise Lipschitz continuity. Consequently, at differentiability points,

$$|\partial \hat{f}(\mathbf{x})/\partial x_i| \leq |\phi'_i(x_i)| = L_i. \tag{12}$$

Therefore, it is sufficient to construct a predictor that is 1-Lipschitz in the transformed controlled input under the ℓ_1 norm while retaining unrestricted dependence on $\mathbf{x}_{\bar{S}}$. We achieve this through a two-branch architecture that incorporates layer-wise interactions with the uncontrolled features while preserving this requirement.

4.2 Proposed Architecture

The proposed Feature-wise Lipschitz Neural Networks (FLNN) architecture consists of controlled and uncontrolled branches, as illustrated in Figure 2. The controlled branch processes the transformed controlled inputs using norm-constrained fully connected layers and GroupSort (Anil et al., 2019), while the uncontrolled branch uses unconstrained fully connected layers and ReLU. The branches interact through element-wise products of controlled representations and normalized uncontrolled representations. Each branch contains $k-1$ hidden layers followed by a scalar-output layer, with $k \geq 2$. The outputs of the two scalar-output layers are added to form the prediction. For each hidden layer $l \in [k-1]$, both branches use the same positive even width m_l to enable element-wise interaction. The even-width requirement comes from using GroupSort with a group size of two.

Weight parameterization For each layer $l \in [k]$, let W_S^l and $W_{\bar{S}}^l$ denote the weight matrices of the controlled and uncontrolled branches, respectively, and let $\mathbf{b}_S^l$ and $\mathbf{b}_{\bar{S}}^l$ denote the corresponding biases. In the controlled branch, we transform W_S^l into $\check{W}_S^l$ and use it in every forward pass. With i and j indexing matrix rows and columns, respectively, the first controlled matrix is parameterized element-wise as

$$(\check{W}_S^1)_{ij} \coloneqq \frac{(W_S^1)_{ij}}{\max\{1, |(W_S^1)_{ij}|\}}. \tag{13}$$

For every subsequent controlled layer $l \in \{2, \dots, k\}$, including the output layer, we parameterize each row of the weight matrix as

$$(\check{W}_S^l)_{i,:} \coloneqq \frac{(W_S^l)_{i,:}}{\max\{1, \|(W_S^l)_{i,:}\|_1\}}. \tag{14}$$

By the matrix norm identities in (4), these parameterizations give

$$\|\check{W}_S^1\|_{1\to\infty} \le 1, \quad \|\check{W}_S^l\|_{\infty\to\infty} \le 1. \tag{15}$$

GroupSort with a group size of two sorts each adjacent pair of hidden coordinates and is 1-Lipschitz under the ℓ_∞ norm. Together with the weight constraints in (15), this property yields the controlled layer bounds derived in Appendix B.1. The uncontrolled weights and all biases are unconstrained.

Layer-wise interactions Let $\mathbf{z}_S^l, \mathbf{z}_{\bar{S}}^l \in \mathbb{R}^{m_l}$ denote the controlled and uncontrolled hidden representations at layer l. We write $\mathbf{z}_{S,\mathrm{in}}^l$ for the input to the controlled layer. The uncontrolled branch starts from $\mathbf{z}_{\bar{S}}^0 \coloneqq \mathbf{x}_{\bar{S}}$, and the first controlled layer receives $\mathbf{z}_{S,\mathrm{in}}^1 \coloneqq \Phi_S(\mathbf{x}_S)$. Each controlled layer after the first receives the previous controlled representation after an element-wise interaction with the normalized uncontrolled representation. Using $\odot$ for element-wise multiplication, define

$$\mathbf{z}_{S,\mathrm{in}}^l \coloneqq \frac{\mathbf{z}_{\bar{S}}^{l-1}}{\max\{1, \|\mathbf{z}_{\bar{S}}^{l-1}\|_\infty\}} \odot \mathbf{z}_S^{l-1}. \tag{16}$$

The normalized uncontrolled representation has ℓ_∞ norm at most one. These interactions allow the effect of the controlled features to vary with the uncontrolled input. The interaction at $l = k$ supplies the input to the controlled scalar-output layer.

Network structure The controlled and uncontrolled branches use fully connected hidden layers followed by GroupSort with a group size of two and ReLU, respectively. With f_l denoting the l-th controlled hidden layer mapping, the controlled and uncontrolled hidden representations are computed as

$$\mathbf{z}_S^l \coloneqq f_l(\mathbf{z}_{S,\mathrm{in}}^l) = \mathrm{GroupSort}(\check{W}_S^l \mathbf{z}_{S,\mathrm{in}}^l + \mathbf{b}_S^l), \tag{17}$$

$$\mathbf{z}_{\bar{S}}^l \coloneqq \mathrm{ReLU}(W_{\bar{S}}^l \mathbf{z}_{\bar{S}}^{l-1} + \mathbf{b}_{\bar{S}}^l). \tag{18}$$

Both branches end with a fully connected output layer without activation, with f_k denoting the controlled output mapping. We denote the outputs of the controlled and uncontrolled branches by $f_S(\mathbf{x})$ and $f_{\bar{S}}(\mathbf{x}_{\bar{S}})$, respectively. The final predictor is defined as

$$\hat{f}(\mathbf{x}) \coloneqq \underbrace{f_k(\mathbf{z}_{S,\mathrm{in}}^k)}_{f_S(\mathbf{x})} + f_{\bar{S}}(\mathbf{x}_{\bar{S}}). \tag{19}$$

Through the interactions, the controlled branch output can depend on both feature groups, while the uncontrolled branch output depends only on $\mathbf{x}_{\bar{S}}$.

4.3 Theoretical Guarantees

4.3.1 Constraint Preservation

The architecture constrains the controlled weights at each layer, while the controlled representations interact with the uncontrolled branch across layers. We therefore first show that, for a fixed uncontrolled input, these interactions preserve the Lipschitz bound with respect to the transformed controlled input, and then translate this bound back to the original input space. Proofs of Lemma 1 and Theorem 1 are given in Appendices B.3.1 and B.2.1, respectively.

Lemma 1 (Layer-wise Lipschitz Constant Preservation)**.** *Consider the architecture in Section 4.2 with controlled weights satisfying* (15)*. Fix any uncontrolled input $\mathbf{x}_{\bar{S}}$ and consider any controlled inputs $\mathbf{x}_S, \mathbf{x}'_S$. Let $\mathbf{z}^l_S$ and $\mathbf{z}^{l\prime}_S$ denote the corresponding controlled hidden representations. Then,*

$$\|\mathbf{z}^l_S - \mathbf{z}^{l\prime}_S\|_\infty \le \|\Phi_S(\mathbf{x}_S) - \Phi_S(\mathbf{x}'_S)\|_1, \qquad l \in [k-1]. \tag{20}$$

Consequently,

$$|\hat{f}(\mathbf{x}_S, \mathbf{x}_{\bar{S}}) - \hat{f}(\mathbf{x}'_S, \mathbf{x}_{\bar{S}})| \le \|\Phi_S(\mathbf{x}_S) - \Phi_S(\mathbf{x}'_S)\|_1. \tag{21}$$

Lemma 1 shows that the interactions with the uncontrolled branch preserve the required Lipschitz bound in the transformed controlled input space. Substituting $\phi_i(x_i) = L_i x_i$ translates this bound into the original input space, yielding the following guarantee.

Theorem 1 (Feature-wise Lipschitz Guarantee)**.** *Under the conditions of Lemma 1, use the input transformation* (9)*. For every controlled coordinate i, input $\mathbf{x}$, and t such that $\mathbf{x} + t\mathbf{e}_i \in \mathcal{X}$,*

$$|\hat{f}(\mathbf{x} + t\mathbf{e}_i) - \hat{f}(\mathbf{x})| \le L_i|t|. \tag{22}$$

Wherever the corresponding partial derivative exists,

$$\left|\frac{\partial \hat{f}(\mathbf{x})}{\partial x_i}\right| \le L_i. \tag{23}$$

In particular, $\hat{f} \in \mathcal{F}_{\mathrm{ctrl}}$.

Theorem 1 establishes that $\hat{f}$ is L_i-Lipschitz in each controlled coordinate x_i, uniformly over the remaining inputs. Because the Lipschitz bound is enforced by construction rather than by a penalty term, it holds for all trainable parameter values at every stage of training, without requiring differentiability. However, satisfying the prescribed constraints does not guarantee that FLNN can approximate the target functions. We therefore examine its expressive capacity.

4.3.2 Universal Approximation under the Constraints

The mixed-norm constraints follow the GroupSort approximation framework of Anil et al. (2019). To establish approximation over both feature groups, we use three lemmas. We write $C(\mathcal{X})$ for the continuous real-valued function class on $\mathcal{X}$. The proof of Theorem 2 is given in Appendix B.2.2, and the supporting lemmas are proved in Appendix B.3.

Lemma 2 (Finite Approximation under the Sensitivity Bounds)**.** *Assume that $\mathcal{X}$ is nonempty and compact and that the transformation Φ_S defined in* (10) *is continuous. Let $f^* \in C(\mathcal{X})$ be a target function satisfying the same bound as in* (21) *for all input pairs with the same uncontrolled coordinates. For every $\delta > 0$, there exists a function f_δ satisfying*

$$0 \le f_\delta(\mathbf{x}) - f^*(\mathbf{x}) \le \delta \qquad \forall \mathbf{x} \in \mathcal{X}. \tag{24}$$

This function is a finite expression formed from pointwise minima and maxima of N component functions,

$$\ell_j(\mathbf{x}) \coloneqq \mathbf{w}_j^\top \Phi_S(\mathbf{x}_S) + b_j(\mathbf{x}_{\bar{S}}), \qquad j \in [N], \tag{25}$$

where N is a positive integer, $\mathbf{w}_j \in \mathbb{R}^{d_S}$ satisfies $\|\mathbf{w}_j\|_\infty \le 1$, and $b_j \in C(\mathcal{X}_{\bar{S}})$. The approximation can be written as $f_\delta(\mathbf{x}) = \min_k \max_{j \in I_k} \ell_j(\mathbf{x})$ for finitely many nonempty index sets $I_k \subseteq [N]$.

Lemma 3 (Approximation of Component Functions through Normalized Interactions)**.** *Under the domain and transformation assumptions of Lemma 2, consider component functions $\ell_1, \dots, \ell_N$ of the form* (25)*, with the specified coefficient conditions. For every $\delta > 0$, there exists a network of the architecture in Section 4.2 with finite depth, a common finite even hidden width, and controlled weights satisfying* (15) *that simultaneously represents approximations $\tilde{\ell}_1, \dots, \tilde{\ell}_N$ in a controlled layer input, up to a common additive constant, such that*

$$\sup_{\mathbf{x} \in \mathcal{X}} \max_{j \in [N]} |\tilde{\ell}_j(\mathbf{x}) - \ell_j(\mathbf{x})| < \delta. \tag{26}$$

The first $2N$ coordinates of this controlled layer input contain two adjacent copies of each shifted approximation, and all remaining coordinates are zero. Both branches can be padded with zeros to a common even width large enough for the construction in Lemma 4, without changing the stored values or the normalization factors. Appendix B.3.3 gives the detailed construction.

Lemma 4 (Finite Minima and Maxima with GroupSort)**.** *For any fixed finite expression formed from N real inputs using minima and maxima, there exists a finite network that evaluates it exactly for all input values. The network receives two adjacent copies of each input value and sufficiently many zero coordinates. It uses fully connected layers with GroupSort of group size two, followed by a scalar-output layer. A common finite even hidden width suffices, and every weight matrix has induced ℓ_∞ norm at most one.*

Combining Lemmas 2, 3, and 4 gives the following approximation result. The constructed predictor retains the feature-wise bounds by Theorem 1.

Theorem 2 (Universal Approximation under Feature-wise Bounds)**.** *Assume that $\mathcal{X}$ is nonempty and compact and that $L_i > 0$ for every controlled coordinate. For every $f^* \in C(\mathcal{X}) \cap \mathcal{F}_{\mathrm{ctrl}}$ and $\epsilon > 0$, the architecture in Section 4.2 with the input transformation* (9) *admits a finite depth k, a common finite even hidden width m, and parameters θ such that*

$$\hat{f}_\theta \in \mathcal{F}_{\mathrm{ctrl}}, \quad \sup_{\mathbf{x}\in\mathcal{X}} |f^*(\mathbf{x}) - \hat{f}_\theta(\mathbf{x})| < \epsilon. \tag{27}$$

The proof first approximates the target using finite combinations of minima and maxima of component functions. Each component is affine in $\Phi_S(\mathbf{x}_S)$, with a continuous offset depending on $\mathbf{x}_{\bar{S}}$. Normalized interactions provide the required dependence on the uncontrolled input, and GroupSort implements the minimum and maximum operations. Appendix B.2.2 combines these three lemmas to prove the theorem. Their proofs are provided in Appendix B.3.

The theorem shows that the architecture can uniformly approximate any continuous target in $\mathcal{F}_{\mathrm{ctrl}}$ over both controlled and uncontrolled inputs while preserving the specified sensitivity bounds.

4.4 Region-wise Extension

Prior knowledge may indicate that output sensitivity varies across the domain of an input feature. For example, surface coverage during atomic layer deposition (ALD) approaches saturation as precursor exposure increases (Vandalon & Kessels, 2016), while enzyme-catalyzed reaction rates can vary non-monotonically with temperature (Hobbs et al., 2013). These examples motivate region-wise sensitivity bounds on controlled features. When such bounds are specified, we incorporate them through the coordinate-wise input transformation.

For each controlled feature index i, let $L_i : \mathcal{X}_i \to [0, \infty)$ denote its prescribed sensitivity bound function. Assume that $\mathcal{X}_i$ is an interval and that L_i is integrable on every compact subinterval. For a reference point $\xi_i \in \mathcal{X}_i$, define

$$\phi_i(x_i) \coloneqq \int_{\xi_i}^{x_i} L_i(u)\mathrm{d}u. \tag{28}$$

The transformation is locally absolutely continuous and satisfies $\phi_i'(x_i) = L_i(x_i)$ almost everywhere. With Φ_S constructed from these transformations as in (10), Lemma 1 yields the following extension, proved in Appendix B.4.

Corollary 1 (Region-wise Lipschitz Guarantee)**.** *Suppose the architectural conditions of Lemma 1 hold and the input transformation is given by* (28) *under the assumptions above. For every fixed uncontrolled input $\mathbf{x}_{\bar{S}}$ and all controlled inputs $\mathbf{x}_S, \mathbf{x}'_S$,*

$$|\hat{f}(\mathbf{x}_S, \mathbf{x}_{\bar{S}}) - \hat{f}(\mathbf{x}'_S, \mathbf{x}_{\bar{S}})| \le \sum_{i=1}^{d_S} \left| \int_{x'_i}^{x_i} L_i(u)\mathrm{d}u \right|. \tag{29}$$

For each controlled coordinate i and every fixed choice of the remaining coordinates, the partial derivative exists and satisfies

$$|\partial \hat{f}(\mathbf{x})/\partial x_i| \le L_i(x_i) \tag{30}$$

for almost every $x_i \in \mathcal{X}_i$.

The bound functions L_i may be piecewise constant or continuous. In the piecewise constant case, each region of $\mathcal{X}_i$ is assigned a separate sensitivity bound. In either case, region-wise control is achieved through the input transformation Φ_S without additional architectural constraints.

5 EXPERIMENTS

5.1 EXPERIMENTAL SETUP

We evaluate FLNN on synthetic datasets and real-world datasets in terms of predictive accuracy and constraint satisfaction. For each dataset, we randomly assign 70% of the samples to the training set, 15% to the validation set, and the remaining 15% to the test set. We report means and standard deviations across five runs using seeds 41 to 45. Hyperparameters are selected based on the root mean squared error (RMSE) on the validation set. Detailed descriptions of the synthetic and real-world datasets, including additional synthetic datasets, as well as the sensitivity bounds, compared methods, hyperparameter search, and metric computation, are provided in Appendix C.

Synthetic datasets We construct the Constant and Region datasets to examine feature-wise sensitivity control under constant and region-wise bounds, respectively. Both datasets share three controlled inputs $\mathbf{x}_S = (x_1, x_2, x_3)$ and two uncontrolled inputs $\mathbf{x}_{\bar{S}} = (v_1, v_2)$, with target

$$f^*(\mathbf{x}_S, \mathbf{x}_{\bar{S}}) = (0.5 + 0.5v_1) \sum_{i=1}^{3} a_i Q(x_i) + 8(v_2 - 0.5). \tag{31}$$

The coefficients a_i set distinct feature-wise sensitivity scales, while v_1 changes the controlled response and v_2 adds an uncontrolled contribution. The coordinate response Q is linear in Constant and piecewise linear in Region, giving constant and region-wise bounds, respectively.

Real-world datasets For real-world regression tasks, we evaluate FLNN on Airfoil (Brooks et al., 1989), AutoMPG (Quinlan, 1993), Combined Cycle Power Plant (CCPP) (Tüfekci, 2014), and Concrete (Yeh, 1998) from the UCI Machine Learning Repository. Controlled features are selected based on domain knowledge, with feature-wise and region-wise bounds specified as empirical sensitivity requirements.

Evaluation metrics We report RMSE for predictive accuracy and the constraint violation rate (CVR) for constraint satisfaction. For synthetic data, we use targets with additive Gaussian noise during training and validation. We compute test RMSE against the noise-free target functions f^* for synthetic data and the observed targets for real-world data. CVR measures the percentage of test sample-feature pairs exceeding their bounds and is computed as

$$\text{CVR} = \frac{100}{N d_S} \sum_{n=1}^{N} \sum_{i=1}^{d_S} \mathbf{1}\{\hat{s}_i^{(n)} > L_i(x_i^{(n)}) + \tau\}, \tag{32}$$

where N is the number of test samples, d_S is the number of controlled features, $\hat{s}_i^{(n)}$ is the estimated absolute slope, and τ is a numerical tolerance. The bounds L_i are derived from the known target functions for synthetic data and estimated from the absolute partial derivatives of surrogate models for real data.

5.2 RESULTS

Predictive Performance and Constraint Satisfaction Table 2 shows that FLNN achieves zero observed CVR on all four real-world datasets and the lowest mean RMSE among methods meeting this criterion on each dataset. FLNN also enforces the specified sensitivity bounds throughout $\mathcal{X}$ by construction, extending constraint satisfaction beyond the evaluated test inputs. Figure 3 compares RMSE and CVR across methods on Airfoil and CCPP. All compared methods with lower mean RMSE than FLNN exhibit violations on these two datasets, whereas PBLNN satisfies the bounds but has substantially higher mean RMSE. For Concrete, FLNN

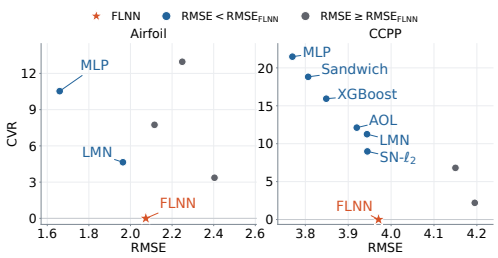

Figure 3: Comparison of RMSE and CVR across methods on Airfoil and CCPP.

Table 2: Test performance on the real-world datasets, reported as mean ± standard deviation across seeds. **Bold** and underlined values indicate the best and second-best RMSEs for each dataset, respectively.

Method	Airfoil		AutoMPG		CCPP		Concrete	
	RMSE	CVR	RMSE	CVR	RMSE	CVR	RMSE	CVR
XGBoost	2.68 ± 0.13	52.92 ± 4.26	3.16 ± 0.41	12.20 ± 1.72	3.85 ± 0.11	15.93 ± 1.65	5.73 ± 0.25	47.42 ± 4.07
MLP	$\mathbf{1.66 \pm 0.24}$	10.53 ± 2.32	2.95 ± 0.50	0.00 ± 0.00	$\mathbf{3.77 \pm 0.07}$	21.48 ± 3.64	$\underline{5.47 \pm 0.69}$	13.16 ± 3.12
SN-ℓ_1	3.23 ± 0.27	0.97 ± 1.42	4.17 ± 2.44	0.00 ± 0.00	4.20 ± 0.05	2.20 ± 4.92	7.78 ± 0.54	4.45 ± 2.02
SN-ℓ_2	2.40 ± 0.39	3.36 ± 1.10	3.44 ± 1.15	0.00 ± 0.00	3.94 ± 0.13	8.98 ± 8.57	6.31 ± 0.62	10.26 ± 3.33
Cayley	3.01 ± 0.27	16.33 ± 3.39	5.32 ± 0.82	32.99 ± 9.13	4.15 ± 0.09	6.81 ± 4.97	9.45 ± 0.97	25.03 ± 2.09
AOL	2.25 ± 0.24	12.96 ± 2.06	3.41 ± 0.52	8.36 ± 3.13	3.92 ± 0.11	12.12 ± 3.90	6.52 ± 0.63	13.29 ± 2.13
Sandwich	2.12 ± 0.34	7.74 ± 2.14	3.12 ± 0.41	1.36 ± 0.64	$\underline{3.81 \pm 0.11}$	18.82 ± 4.41	5.79 ± 0.68	12.52 ± 2.45
LMN	$\underline{1.96 \pm 0.16}$	4.65 ± 2.43	$\underline{2.95 \pm 0.45}$	0.00 ± 0.00	3.94 ± 0.05	11.25 ± 2.79	5.71 ± 0.50	9.03 ± 3.93
PBLNN	6.01 ± 0.29	0.00 ± 0.00	4.43 ± 0.73	0.00 ± 0.00	7.08 ± 0.11	0.00 ± 0.00	9.96 ± 1.40	8.71 ± 1.21
FLNN	2.07 ± 0.22	0.00 ± 0.00	$\mathbf{2.93 \pm 0.37}$	0.00 ± 0.00	3.97 ± 0.10	0.00 ± 0.00	$\mathbf{5.31 \pm 0.33}$	0.00 ± 0.00

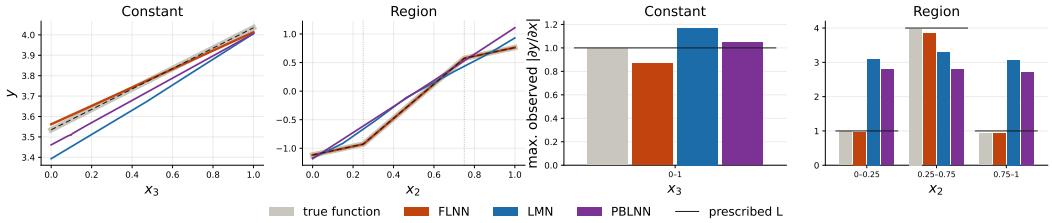

Figure 4: Coordinate slices and maximum observed slopes on synthetic data

achieves the lowest mean RMSE under sensitivity requirements that combine four regional bounds on curing age with a separate bound on cement content, while every evaluated baseline exhibits violations. This result suggests that incorporating feature- and region-specific requirements may provide a useful inductive bias for prediction.

Fitting under Sensitivity Constraints Figure 4 compares FLNN, LMN, and PBLNN using coordinate slices and maximum observed absolute partial derivatives on the Constant and Region datasets. Each slice varies one controlled feature while holding the remaining inputs fixed, whereas the derivative maxima are computed over all test samples within each feature interval. The plots show the sensitivity bounds as horizontal lines, with bars extending above these lines indicating observed violations. The Constant example focuses on x_3, which has the smallest sensitivity bound among controlled features. FLNN closely follows the target response, while LMN and PBLNN underestimate it. The derivative measurements show that FLNN remains within the bound, whereas both baselines exceed it. The Region example illustrates the extension of FLNN to sensitivity requirements that vary across input intervals. FLNN follows the target slope changes at $x_2 = 0.25$ and 0.75 while respecting the corresponding regional bounds. LMN and PBLNN produce approximately linear responses and exceed the smaller bounds in the outer intervals. These results show that FLNN can accommodate the stronger response in the middle interval while enforcing tighter sensitivity limits in the outer intervals. Additional quantitative results and analyses for the synthetic datasets are provided in Appendix D.

6 Conclusion

We presented FLNN, a neural network that enforces feature-wise sensitivity bounds on selected inputs while allowing unconstrained dependence on the remaining features. The architecture combines feature transformations with interactions between controlled and uncontrolled branches and extends to region-wise sensitivity requirements. We established constraint preservation and universal approximation within the corresponding constrained function class. Experiments show that FLNN achieves competitive predictive accuracy while satisfying the specified bounds, with synthetic analyses illustrating its ability to capture feature-specific responses and regional changes in sensitivity.

AI USE STATEMENT

We used generative AI tools to improve the clarity and grammar of the manuscript, identify potential errors in proofs, assist with code implementation and debugging, and explore functional forms for generating the synthetic datasets. At least two authors independently reviewed all AI-assisted contributions and verified the proofs, experimental code, and reported results. The authors are fully responsible for the final manuscript, its claims, and all accompanying artifacts.

REPRODUCIBILITY STATEMENT

To facilitate reproducibility, we provide the FLNN source code in the supplementary material. The datasets, model architectures, training and hyperparameter settings, and evaluation procedures are described in Section 5 and Appendix C. Complete proofs of the theoretical results are presented in Appendix B.

REFERENCES

Brandon Amos, Lei Xu, and J. Zico Kolter. Input convex neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 146–155. PMLR, 06–11 Aug 2017.

Cem Anil, James Lucas, and Roger Grosse. Sorting out Lipschitz function approximation. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 291–301. PMLR, 09–15 Jun 2019.

Alexandre Araujo, Aaron J Havens, Blaise Delattre, Alexandre Allauzen, and Bin Hu. A unified algebraic perspective on lipschitz neural networks. In *The Eleventh International Conference on Learning Representations*, 2023.

Thomas F Brooks, D Stuart Pope, and Michael A Marcolini. Airfoil self-noise and prediction. Technical report, National Aeronautics and Space Administration (NASA), 1989.

Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pp. 785–794, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450342322. doi: 10.1145/2939672.2939785.

Moustapha Cisse, Piotr Bojanowski, Edouard Grave, Yann Dauphin, and Nicolas Usunier. Parseval networks: Improving robustness to adversarial examples. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 854–863. PMLR, 06–11 Aug 2017.

Blaise Delattre, Quentin Barthélemy, Alexandre Araujo, and Alexandre Allauzen. Efficient bound of Lipschitz constant for convolutional layers by Gram iteration. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 7513–7532. PMLR, 23–29 Jul 2023.

Henry Gouk, Eibe Frank, Bernhard Pfahringer, and Michael J. Cree. Regularisation of neural networks by enforcing lipschitz continuity. *Machine Learning*, 110(2):393–416, Feb 2021. ISSN 1573-0565. doi: 10.1007/s10994-020-05929-w.

Kavya Gupta, Fateh Kaakai, Beatrice Pesquet-Popescu, Jean-Christophe Pesquet, and Fragkiskos D. Malliaros. Multivariate lipschitz analysis of the stability of neural networks. *Frontiers in Signal Processing*, Volume 2 - 2022, 2022. ISSN 2673-8198. doi: 10.3389/frsip.2022.794469.

Joanne K. Hobbs, Wanting Jiao, Ashley D. Easter, Emily J. Parker, Louis A. Schipper, and Vickery L. Arcus. Change in heat capacity for enzyme catalysis determines temperature dependence of enzyme catalyzed rates. *ACS Chemical Biology*, 8(11):2388–2393, 11 2013. ISSN 1554-8929. doi: 10.1021/cb4005029.

Kai Hu, Andy Zou, Zifan Wang, Klas Leino, and Matt Fredrikson. Unlocking deterministic robustness certification on imagenet. In *Advances in Neural Information Processing Systems*, volume 36, pp. 42993–43011. Curran Associates, Inc., 2023. doi: 10.52202/075280-1863.

Kai Hu, Haoqi Hu, and Matt Fredrikson. Lipnext: Scaling up lipschitz-based certified robustness to billion-parameter models. In *The Fourteenth International Conference on Learning Representations*, 2026.

George Em Karniadakis, Ioannis G. Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, Jun 2021. ISSN 2522-5820. doi: 10.1038/s42254-021-00314-5.

Patrick Kidger and Terry Lyons. Universal Approximation with Deep Narrow Networks. In *Proceedings of Thirty Third Conference on Learning Theory*, volume 125 of *Proceedings of Machine Learning Research*, pp. 2306–2327. PMLR, 09–12 Jul 2020.

Hyunho Kim and Jong-Seok Lee. Scalable monotonic neural networks. In *International Conference on Learning Representations*, volume 2024, pp. 35160–35176, 2024.

Yuri Kinoshita and Taro Toyoizumi. A provable control of sensitivity of neural networks through a direct parameterization of the overall bi-lipschitzness. In *Advances in Neural Information Processing Systems*, volume 37, pp. 85241–85309. Curran Associates, Inc., 2024. doi: 10.52202/079017-2707.

Bo-Han Lai, Pin-Han Huang, Bo-Han Kung, and Shang-Tse Chen. Enhancing certified robustness via block reflector orthogonal layers and logit annealing loss. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 32246–32277. PMLR, 13–19 Jul 2025.

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018.

Niklas Nolte, Ouail Kitouni, and Mike Williams. Expressive monotonic neural networks. In *The Eleventh International Conference on Learning Representations*, 2023.

Bernd Prach and Christoph H. Lampert. Almost-orthogonal layers for efficient general-purpose lipschitz networks. In *Computer Vision – ECCV 2022*, pp. 350–365, Cham, 2022. Springer Nature Switzerland. ISBN 978-3-031-19803-8.

R. Quinlan. Auto MPG. UCI Machine Learning Repository, 1993. DOI: https://doi.org/10.24432/C5859H.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, Oct 1986. ISSN 1476-4687. doi: 10.1038/323533a0.

Davor Runje and Sharath M Shankaranarayana. Constrained monotonic neural networks. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 29338–29353. PMLR, 23–29 Jul 2023.

Asher Trockman and J Zico Kolter. Orthogonalizing convolutional layers with the cayley transform. In *International Conference on Learning Representations*, 2021.

Yusuke Tsuzuku, Issei Sato, and Masashi Sugiyama. Lipschitz-margin training: Scalable certification of perturbation invariance for deep neural networks. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

Pınar Tüfekci. Prediction of full load electrical power output of a base load operated combined cycle power plant using machine learning methods. *International Journal of Electrical Power & Energy Systems*, 60:126–140, 2014. ISSN 0142-0615. doi: https://doi.org/10.1016/j.ijepes.2014.02.027.

V. Vandalon and W. M. M. Kessels. What is limiting low-temperature atomic layer deposition of al2o3? a vibrational sum-frequency generation study. *Applied Physics Letters*, 108(1):011607, 01 2016. ISSN 0003-6951. doi: 10.1063/1.4939654.

Ruigang Wang and Ian Manchester. Direct parameterization of Lipschitz-bounded deep networks. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 36093–36110. PMLR, 23–29 Jul 2023.

Jared Willard, Xiaowei Jia, Shaoming Xu, Michael Steinbach, and Vipin Kumar. Integrating scientific knowledge with machine learning for engineering and environmental systems. *ACM Comput. Surv.*, 55(4), November 2022. ISSN 0360-0300. doi: 10.1145/3514228.

I.-C. Yeh. Modeling of strength of high-performance concrete using artificial neural networks. *Cement and Concrete Research*, 28(12):1797–1808, 1998. ISSN 0008-8846. doi: https://doi.org/10.1016/S0008-8846(98)00165-3.

Seungil You, David Ding, Kevin Canini, Jan Pfeifer, and Maya Gupta. Deep lattice networks and partial monotonic functions. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

A NOTATION

Table 3 summarizes the main notation used throughout the paper. The symbols are grouped into basic setting and feature transformation, and network architecture and universal approximation. Notation introduced only within individual proofs is omitted.

Table 3: Summary of notation

Symbol	Description
	Basic setting and feature transformation
$[n]$	Index set $\{1, \dots, n\}$, $n \in \mathbb{N}$.
$\|\cdot\|_p$	Vector ℓ_p norm, $p \in [1, \infty]$.
$\|W\|_{p\to q}$	Induced $\ell_p \to \ell_q$ matrix norm.
$\mathrm{Lip}_p(f)$	Lipschitz constant with respect to the ℓ_p norm.
$S, \bar{S}$	Controlled and uncontrolled feature groups.
$d, d_S, d_{\bar{S}}$	Total and group dimensions, with $d = d_S + d_{\bar{S}}$.
$\mathbf{x}, \mathbf{x}_S, \mathbf{x}_{\bar{S}}$	Input and its groups. $\mathbf{x} = (\mathbf{x}_S, \mathbf{x}_{\bar{S}})$, $\mathbf{x}_S = (x_i)_{i=1}^{d_S}$.
$\mathcal{X}_i, \mathcal{X}_S, \mathcal{X}_{\bar{S}}, \mathcal{X}$	Input domains. $\mathcal{X}_S = \prod_{i=1}^{d_S} \mathcal{X}_i$, $\mathcal{X} = \mathcal{X}_S \times \mathcal{X}_{\bar{S}}$.
$\mathbf{e}_i$	Standard basis vector in $\mathbb{R}^d$ for controlled coordinate i.
$\mathcal{F}_{\mathrm{ctrl}}$	Function class satisfying the prescribed sensitivity bounds.
$f^*, \hat{f}_\theta, \hat{f}$	Target function and scalar predictor on $\mathcal{X}$, with $\hat{f} = \hat{f}_\theta$.
$L_i, L_i(\cdot)$	Constant and region-wise sensitivity bounds for feature i.
$\phi_i : \mathcal{X}_i \to \mathbb{R}$	Coordinate transformation, with $\phi_i(x_i) = L_i x_i$ for constant bounds.
$\Phi_S : \mathcal{X}_S \to \mathbb{R}^{d_S}$	Group transformation defined by $\Phi_S(\mathbf{x}_S) = (\phi_i(x_i))_{i=1}^{d_S}$.
$\xi_i \in \mathcal{X}_i$	Reference point for $\phi_i(x_i) = \int_{\xi_i}^{x_i} L_i(t)\,\mathrm{d}t$.
	Network architecture and universal approximation
k, l	Branch depth including the output layer, and layer index $l \in [k]$.
$\odot$	Element-wise multiplication.
m_l, m	Positive even widths shared across branches at block l and fixed for approximation, respectively.
$W_S^l, W_{\bar{S}}^l$	Trainable controlled and uncontrolled weights at layer l.
$\check{W}_S^l$	Parameterized controlled weights used in forward passes.
$\mathbf{z}_S^l, \mathbf{z}_{\bar{S}}^l$	Controlled and uncontrolled hidden outputs, with $\mathbf{z}_{\bar{S}}^0 = \mathbf{x}_{\bar{S}}$.
$\mathbf{z}_{S,\mathrm{in}}^l$	Input to f_l, given by $\Phi_S(\mathbf{x}_S)$ for $l = 1$ and $\mathbf{z}_S^{l-1}$ after normalized interaction otherwise.
f_l	Controlled mapping with norm-constrained weights, GroupSort for $l < k$, and output for $l = k$.
$f_S, f_{\bar{S}}$	Controlled and uncontrolled branch mappings, with $\hat{f}(\mathbf{x}) = f_S(\mathbf{x}) + f_{\bar{S}}(\mathbf{x}_{\bar{S}})$.
δ, ϵ	Intermediate and target approximation tolerances, with $0 < \delta < \epsilon/2$.
f_δ	Finite minimum/maximum approximation satisfying $0 \le f_\delta(\mathbf{x}) - f^*(\mathbf{x}) \le \delta$ on $\mathcal{X}$.
N	Number of component functions, indexed by $j \in [N]$.
$\ell_j, \tilde{\ell}_j$	Component function and its approximation, with $\ell_j(\mathbf{x}) = \mathbf{w}_j^\top \Phi_S(\mathbf{x}_S) + b_j(\mathbf{x}_{\bar{S}})$.
$\mathbf{w}_j, b_j$	Coefficient vector with $\|\mathbf{w}_j\|_\infty \le 1$ and continuous additive function $b_j \in C(\mathcal{X}_{\bar{S}})$.

B PROOFS

We use the notation and architectural definitions from the main text. The theorem proofs invoke the lemma statements, whose detailed proofs follow in Appendix B.3. Auxiliary quantities are defined locally where they are used.

B.1 WEIGHT PARAMETERIZATION AND CONTROLLED LAYER BOUNDS

The first controlled layer bounds output differences in ℓ_∞ by input differences in ℓ_1. Subsequent controlled layers do not increase differences measured in ℓ_∞. We verify these two claims before proving Lemma 1.

First controlled layer For any $\mathbf{x} \in \mathbb{R}^{d_S}$, the element-wise parameterization in (13) gives

$$\|\check{W}_S^1 \mathbf{x}\|_\infty = \max_i \Big| \sum_j (\check{W}_S^1)_{ij} x_j \Big| \tag{33}$$

$$\leq \max_{i,j} |(\check{W}_S^1)_{ij}| \sum_j |x_j| \leq \|\mathbf{x}\|_1. \tag{34}$$

Subsequent controlled layers For $2 \leq l \leq k$ and $\mathbf{v} \in \mathbb{R}^{m_{l-1}}$, the row-wise parameterization in (14) gives

$$\|\check{W}_S^l \mathbf{v}\|_\infty \leq \max_i \sum_j |(\check{W}_S^l)_{ij}||v_j| \tag{35}$$

$$\leq \Big(\max_i \sum_j |(\check{W}_S^l)_{ij}| \Big) \|\mathbf{v}\|_\infty \leq \|\mathbf{v}\|_\infty. \tag{36}$$

For the scalar-output layer, the output norm is absolute value and the weight row has ℓ_1 norm at most one. The bounds above apply to rectangular matrices, so they do not require equal widths across hidden layers.

Biases and GroupSort A fully connected layer has the same difference bound with or without its bias, since the bias cancels when comparing two inputs. GroupSort with a group size of two also preserves the ℓ_∞ difference bound. If corresponding input coordinates differ by at most $a \geq 0$, the minima and maxima of each pair differ by at most a. Thus

$$\|\mathrm{GroupSort}(\mathbf{u}) - \mathrm{GroupSort}(\mathbf{v})\|_\infty \leq \|\mathbf{u} - \mathbf{v}\|_\infty. \tag{37}$$

Combining these facts gives

$$\|f_1(\mathbf{u}) - f_1(\mathbf{v})\|_\infty \leq \|\mathbf{u} - \mathbf{v}\|_1, \quad \|f_l(\mathbf{u}) - f_l(\mathbf{v})\|_\infty \leq \|\mathbf{u} - \mathbf{v}\|_\infty, \tag{38}$$

for $2 \leq l \leq k$, where the left-hand side is an absolute value for $l = k$. By (13) and (14), every trainable matrix is mapped to a parameterized matrix satisfying (15); hence these inequalities hold for all values of the trainable weights in every forward pass.

B.2 PROOFS OF THEOREMS

B.2.1 PROOF OF THEOREM 1

Proof. Fix an input $\mathbf{x}$, a controlled coordinate i, and t such that $\mathbf{x} + t\mathbf{e}_i \in \mathcal{X}$. Since only coordinate i changes, Lemma 1 and (9) give

$$|\hat{f}(\mathbf{x} + t\mathbf{e}_i) - \hat{f}(\mathbf{x})| \leq |\phi_i(x_i + t) - \phi_i(x_i)| \tag{39}$$

$$= L_i |t|. \tag{40}$$

This proves (22) and gives $\hat{f} \in \mathcal{F}_{\mathrm{ctrl}}$.

At a point where $\partial \hat{f}(\mathbf{x})/\partial x_i$ exists, dividing the preceding inequality by $|t|$ for $t \neq 0$ gives

$$\Big| \frac{\hat{f}(\mathbf{x} + t\mathbf{e}_i) - \hat{f}(\mathbf{x})}{t} \Big| \leq L_i. \tag{41}$$

Taking $t \to 0$ yields (23). □

B.2.2 Proof of Theorem 2

Proof. Let $f^* \in C(\mathcal{X}) \cap \mathcal{F}_{\mathrm{ctrl}}$ and $\epsilon > 0$, with $0 < \delta < \epsilon/2$. The product structure of $\mathcal{X}_S$ allows the controlled coordinates to be changed one at a time while remaining in the domain. The feature-wise bounds on f^* therefore imply that it satisfies the same bound as in (21). Since the feature transformation Φ_S is continuous, the assumptions of Lemma 2 are satisfied. The lemma provides a finite minimum/maximum expression f_δ in component functions $\ell_1, \ldots, \ell_N$ such that

$$\sup_{\mathbf{x}\in\mathcal{X}} |f^*(\mathbf{x}) - f_\delta(\mathbf{x})| \le \delta. \tag{42}$$

Lemma 3 provides component approximations satisfying (26). For a common constant $c \in \mathbb{R}$, the constructed network prepares two adjacent copies of each $\tilde{\ell}_j(\mathbf{x}) + c$ at a controlled layer input, with all remaining coordinates zero. We refer to the network provided by Lemma 4 as the GroupSort network. It forms the remaining part of the controlled branch after the component approximations have been prepared. It evaluates the finite expression of minima and maxima defining f_δ on the prepared controlled layer input. Zero padding allows both constructions to use the same finite even hidden width. Each added uncontrolled hidden layer has zero weights and a bias vector of ones. Its output is therefore a vector of ones, so the subsequent normalized interactions leave the controlled representations unchanged. The resulting controlled output evaluates the expression defining f_δ on the shifted approximations $\tilde{\ell}_j(\mathbf{x}) + c$.

Adding a common constant to every argument of a minimum or maximum adds the same constant to its result. This property carries through the finite expression. Subtracting c from the final controlled output bias therefore removes the common shift. The uncontrolled scalar-output weights and bias are set to zero. The resulting predictor $\hat{f}_\theta$ evaluates exactly the expression defining f_δ with each ℓ_j replaced by $\tilde{\ell}_j$.

The GroupSort network is 1-Lipschitz under the ℓ_∞ norm, since its weight matrices have induced ℓ_∞ norm at most one and GroupSort satisfies (37). The common shift, paired copies, and zero padding do not change the largest difference between corresponding component values. Consequently,

$$\sup_{\mathbf{x}\in\mathcal{X}} |f_\delta(\mathbf{x}) - \hat{f}_\theta(\mathbf{x})| \le \sup_{\mathbf{x}\in\mathcal{X}} \max_{j\in[N]} |\ell_j(\mathbf{x}) - \tilde{\ell}_j(\mathbf{x})| < \delta. \tag{43}$$

Together with (42), the triangle inequality gives

$$\sup_{\mathbf{x}\in\mathcal{X}} |f^*(\mathbf{x}) - \hat{f}_\theta(\mathbf{x})| < 2\delta < \epsilon. \tag{44}$$

The initial controlled weights satisfy (15) by Lemma 3, and every appended controlled matrix satisfies the required induced ℓ_∞ norm bound by Lemma 4. The parameterizations in (13) and (14) leave these matrices unchanged when the trainable weights are set equal to them. The resulting network therefore belongs to the architecture in Section 4.2, with finite depth and a common finite even hidden width. Theorem 1 gives $\hat{f}_\theta \in \mathcal{F}_{\mathrm{ctrl}}$, completing the proof. □

B.3 Proofs of Lemmas

B.3.1 Proof of Lemma 1

Proof. Fix $\mathbf{x}_{\bar{S}}$ and consider two controlled inputs $\mathbf{x}_S, \mathbf{x}'_S$. Each uncontrolled hidden representation depends only on $\mathbf{x}_{\bar{S}}$. Therefore, it remains unchanged when $\mathbf{x}_S$ is replaced by $\mathbf{x}'_S$. GroupSort is 1-Lipschitz in ℓ_∞ by (37), so the first controlled layer satisfies

$$\|\mathbf{z}_S^1 - \mathbf{z}_S^{1\prime}\|_\infty \le \|\check{W}_S^1(\Phi_S(\mathbf{x}_S) - \Phi_S(\mathbf{x}'_S))\|_\infty \tag{45}$$

$$\le \|\Phi_S(\mathbf{x}_S) - \Phi_S(\mathbf{x}'_S)\|_1. \tag{46}$$

For every later interaction, the common uncontrolled representation gives

$$\|\mathbf{z}_{S,\mathrm{in}}^l - \mathbf{z}_{S,\mathrm{in}}^{l\prime}\|_\infty \le \frac{\|\mathbf{z}_{\bar{S}}^{l-1}\|_\infty}{\max\{1, \|\mathbf{z}_{\bar{S}}^{l-1}\|_\infty\}} \|\mathbf{z}_S^{l-1} - \mathbf{z}_S^{(l-1)\prime}\|_\infty \tag{47}$$

$$\le \|\mathbf{z}_S^{l-1} - \mathbf{z}_S^{(l-1)\prime}\|_\infty, \qquad 2 \le l \le k. \tag{48}$$

For $l < k$, applying the subsequent-layer bound in (38) shows that the next controlled hidden layer preserves this inequality. Induction proves (20). The same interaction bound holds before the scalar-output layer. Its weight row has ℓ_1 norm at most one, and the additive uncontrolled output cancels. Therefore,

$$|\hat{f}(\mathbf{x}_S, \mathbf{x}_{\bar{S}}) - \hat{f}(\mathbf{x}'_S, \mathbf{x}_{\bar{S}})| = |f_k(\mathbf{z}^k_{S,\mathrm{in}}) - f_k(\mathbf{z}^{k\prime}_{S,\mathrm{in}})| \tag{49}$$

$$\leq \|\mathbf{z}^k_{S,\mathrm{in}} - \mathbf{z}^{k\prime}_{S,\mathrm{in}}\|_\infty \tag{50}$$

$$\leq \|\Phi_S(\mathbf{x}_S) - \Phi_S(\mathbf{x}'_S)\|_1, \tag{51}$$

which proves (21). □

B.3.2 Proof of Lemma 2

Proof. We first construct an approximation and verify its error bound, then express it using the required component functions. Since $\Phi_S(\mathcal{X}_S)$ is compact, it can be covered by finitely many open ℓ_1-balls of radius $\delta/2$ with centers in $\Phi_S(\mathcal{X}_S)$. Choose a finite set $\mathcal{A} \subseteq \mathcal{X}_S$ such that $\Phi_S(\mathcal{A})$ is the set of these centers. Then, for every $\mathbf{x}_S \in \mathcal{X}_S$, there exists $\mathbf{a} \in \mathcal{A}$ such that

$$\|\Phi_S(\mathbf{x}_S) - \Phi_S(\mathbf{a})\|_1 \leq \delta/2. \tag{52}$$

For any $\mathbf{x} \in \mathcal{X}$ and $\mathbf{a} \in \mathcal{A}$, the assumed sensitivity bound gives

$$f^*(\mathbf{x}) \leq f^*(\mathbf{a}, \mathbf{x}_{\bar{S}}) + \|\Phi_S(\mathbf{x}_S) - \Phi_S(\mathbf{a})\|_1. \tag{53}$$

Taking the minimum of these upper bounds over $\mathbf{a} \in \mathcal{A}$, we define

$$f_\delta(\mathbf{x}) \coloneqq \min_{\mathbf{a}} \left\{ f^*(\mathbf{a}, \mathbf{x}_{\bar{S}}) + \|\Phi_S(\mathbf{x}_S) - \Phi_S(\mathbf{a})\|_1 \right\}. \tag{54}$$

Fix any $\mathbf{x} \in \mathcal{X}$ and choose $\mathbf{a} \in \mathcal{A}$ satisfying (52). Then, by the assumed sensitivity bound,

$$f_\delta(\mathbf{x}) \leq f^*(\mathbf{a}, \mathbf{x}_{\bar{S}}) + \|\Phi_S(\mathbf{x}_S) - \Phi_S(\mathbf{a})\|_1 \tag{55}$$

$$\leq f^*(\mathbf{x}) + 2\|\Phi_S(\mathbf{x}_S) - \Phi_S(\mathbf{a})\|_1 \tag{56}$$

$$\leq f^*(\mathbf{x}) + \delta. \tag{57}$$

Together with $f_\delta(\mathbf{x}) \geq f^*(\mathbf{x})$, this establishes (24). For every $\mathbf{y} \in \mathbb{R}^{d_S}$, the ℓ_1 norm can be expressed as a maximum over sign vectors $\boldsymbol{\sigma} \in \{-1, 1\}^{d_S}$.

$$\|\mathbf{y}\|_1 = \max_{\boldsymbol{\sigma}} \boldsymbol{\sigma}^\top \mathbf{y}. \tag{58}$$

Applying this identity to the distance term in (54) gives

$$f_\delta(\mathbf{x}) = \min_{\mathbf{a}} \max_{\boldsymbol{\sigma}} \{\boldsymbol{\sigma}^\top \Phi_S(\mathbf{x}_S) + f^*(\mathbf{a}, \mathbf{x}_{\bar{S}}) - \boldsymbol{\sigma}^\top \Phi_S(\mathbf{a})\}. \tag{59}$$

Set $N = |\mathcal{A}| \times 2^{d_S}$ and index the pairs $(\mathbf{a}, \boldsymbol{\sigma}) \in \mathcal{A} \times \{-1, 1\}^{d_S}$ by $j \in [N]$. For the pair indexed by j, define

$$\mathbf{w}_j \coloneqq \boldsymbol{\sigma}, \quad b_j(\mathbf{x}_{\bar{S}}) \coloneqq f^*(\mathbf{a}, \mathbf{x}_{\bar{S}}) - \boldsymbol{\sigma}^\top \Phi_S(\mathbf{a}). \tag{60}$$

Each expression inside the braces in (59) is then a component function $\ell_j(\mathbf{x}) = \mathbf{w}_j^\top \Phi_S(\mathbf{x}_S) + b_j(\mathbf{x}_{\bar{S}})$. Since f^* is continuous, each b_j belongs to $C(\mathcal{X}_{\bar{S}})$, and the sign vectors satisfy $\|\mathbf{w}_j\|_\infty = 1$. Thus, f_δ has the required finite minimum/maximum representation. □

B.3.3 Proof of Lemma 3

Proof. We first approximate the component functions by products, up to a common additive constant. We then construct bounded ReLU approximations of the coefficients in these products. Finally, we implement the construction in the first four hidden layers of each branch and the interaction preceding the fifth controlled mapping. Throughout this proof, suprema over $\mathbf{x}$ and $\mathbf{x}_{\bar{S}}$ are taken over $\mathcal{X}$ and $\mathcal{X}_{\bar{S}}$, respectively, and maxima over j are taken over $[N]$.

Step 1. product approximation of the component functions Choose a common constant $C \geq 1$ such that, for every $\mathbf{x} \in \mathcal{X}$ and $j \in [N]$,

$$\|\Phi_S(\mathbf{x}_S)\|_1 \leq C, \qquad |b_j(\mathbf{x}_{\bar{S}})| \leq C. \tag{61}$$

Such a constant exists because Φ_S and the finitely many functions b_j are continuous on compact domains. The bound $\|\mathbf{w}_j\|_\infty \leq 1$ also gives $|\mathbf{w}_j^\top \Phi_S(\mathbf{x}_S)| \leq C$. Choose $B > \max\{2C, 4C^2/\delta\}$ and define

$$\alpha_j(\mathbf{x}_{\bar{S}}) \coloneqq 1 + \frac{b_j(\mathbf{x}_{\bar{S}}) - C}{B}, \qquad j \in [N]. \tag{62}$$

Each α_j is continuous and takes values in $[0, 1]$. Expanding the product gives

$$\alpha_j(\mathbf{x}_{\bar{S}})(B + \mathbf{w}_j^\top \Phi_S(\mathbf{x}_S)) = \ell_j(\mathbf{x}) + B - C + \frac{\mathbf{w}_j^\top \Phi_S(\mathbf{x}_S)(b_j(\mathbf{x}_{\bar{S}}) - C)}{B}. \tag{63}$$

The product therefore approximates $\ell_j(\mathbf{x}) + B - C$ with uniform error at most $2C^2/B < \delta/2$. Suppose that functions $\tilde{\alpha}_j : \mathcal{X}_{\bar{S}} \to [0, 1]$, $j \in [N]$, satisfy

$$\sup_{\mathbf{x}_{\bar{S}}} \max_j |\tilde{\alpha}_j(\mathbf{x}_{\bar{S}}) - \alpha_j(\mathbf{x}_{\bar{S}})| < \frac{\delta}{2(B + C)}. \tag{64}$$

Define the corresponding component approximations by

$$\tilde{\ell}_j(\mathbf{x}) \coloneqq \tilde{\alpha}_j(\mathbf{x}_{\bar{S}})(B + \mathbf{w}_j^\top \Phi_S(\mathbf{x}_S)) - B + C. \tag{65}$$

By (63), the approximation error can be written as

$$\tilde{\ell}_j(\mathbf{x}) - \ell_j(\mathbf{x}) = \frac{\mathbf{w}_j^\top \Phi_S(\mathbf{x}_S)(b_j(\mathbf{x}_{\bar{S}}) - C)}{B} + (\tilde{\alpha}_j(\mathbf{x}_{\bar{S}}) - \alpha_j(\mathbf{x}_{\bar{S}}))(B + \mathbf{w}_j^\top \Phi_S(\mathbf{x}_S)). \tag{66}$$

The two terms account for the product approximation and the coefficient approximation, respectively. Using $|B + \mathbf{w}_j^\top \Phi_S(\mathbf{x}_S)| \leq B + C$, we obtain

$$\sup_{\mathbf{x}} \max_j |\tilde{\ell}_j(\mathbf{x}) - \ell_j(\mathbf{x})| \leq \frac{2C^2}{B} + (B + C) \sup_{\mathbf{x}_{\bar{S}}} \max_j |\tilde{\alpha}_j(\mathbf{x}_{\bar{S}}) - \alpha_j(\mathbf{x}_{\bar{S}})| \tag{67}$$

$$< \delta. \tag{68}$$

The last inequality follows from the choice of B and (64), establishing (26). The next step constructs coefficient approximations satisfying (64).

Step 2. coefficient approximation with bounded ReLU computations Let $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_N)^\top$. The classical ReLU universal approximation theorem guarantees uniform approximation of each continuous function α_j on the compact set $\mathcal{X}_{\bar{S}}$ (Kidger & Lyons, 2020). These scalar approximations can be combined into a network $\mathbf{g}$ approximating $\boldsymbol{\alpha}$, with one hidden layer of finite width H.

$$\mathbf{g}(\mathbf{x}_{\bar{S}}) = A^2 \,\mathrm{ReLU}(A^1 \mathbf{x}_{\bar{S}} + \boldsymbol{\beta}^1) + \boldsymbol{\beta}^2, \tag{69}$$

where $A^1 \in \mathbb{R}^{H \times d_{\bar{S}}}$, $\boldsymbol{\beta}^1 \in \mathbb{R}^H$, $A^2 \in \mathbb{R}^{N \times H}$, and $\boldsymbol{\beta}^2 \in \mathbb{R}^N$. The network can be chosen so that

$$\sup_{\mathbf{x}_{\bar{S}}} \|\mathbf{g}(\mathbf{x}_{\bar{S}}) - \boldsymbol{\alpha}(\mathbf{x}_{\bar{S}})\|_\infty < \frac{\delta}{2(B + C)}. \tag{70}$$

The coefficient approximations must meet the accuracy requirement in (64) and take values in $[0, 1]$. The error bound for $\mathbf{g}$ provides the required accuracy but does not guarantee this range. We construct $\tilde{\boldsymbol{\alpha}} = (\tilde{\alpha}_1, \ldots, \tilde{\alpha}_N)^\top$ using the four ReLU computations below. These computations clip the outputs of $\mathbf{g}$ to $[0, 1]^N$ without increasing the approximation error. All intermediate coordinates also remain in $[0, 1]$, so they do not increase the normalization factor in Step 3.

Let $R \geq 1$ bound every coordinate of $\mathrm{ReLU}(A^1 \mathbf{x}_{\bar{S}} + \boldsymbol{\beta}^1)$ and $\mathrm{ReLU}(\mathbf{g}(\mathbf{x}_{\bar{S}}))$ over $\mathcal{X}_{\bar{S}}$. Such a finite R exists because these are finitely many continuous functions on a compact domain. With $\mathbf{1}_N$ denoting the vector of N ones, define

$$\mathbf{h}^1 \coloneqq \mathrm{ReLU}((A^1 \mathbf{x}_{\bar{S}} + \boldsymbol{\beta}^1)/R) \in \mathbb{R}^H, \tag{71}$$

$$\mathbf{h}^2 \coloneqq \mathrm{ReLU}((RA^2 \mathbf{h}^1 + \boldsymbol{\beta}^2)/R) \in \mathbb{R}^N, \tag{72}$$

$$\mathbf{h}^3 \coloneqq \mathrm{ReLU}(\mathbf{1}_N - R\mathbf{h}^2) \in \mathbb{R}^N, \tag{73}$$

$$\tilde{\boldsymbol{\alpha}} \coloneqq \mathrm{ReLU}(\mathbf{1}_N - \mathbf{h}^3) \in \mathbb{R}^N. \tag{74}$$

All these vectors depend only on $\mathbf{x}_{\bar{S}}$. The first two computations give the hidden activation of $\mathbf{g}$ and $\mathrm{ReLU}(\mathbf{g}(\mathbf{x}_{\bar{S}}))$, respectively, each divided by R. The choice of R therefore ensures that every coordinate of $\mathbf{h}^1$ and $\mathbf{h}^2$ lies in $[0,1]$. The last two computations cap the coordinates of $\mathrm{ReLU}(\mathbf{g}(\mathbf{x}_{\bar{S}}))$ at one. Thus, $\tilde{\boldsymbol{\alpha}}$ is obtained by clipping each coordinate of $\mathbf{g}$ to $[0,1]$.

$$\tilde{\alpha}_j(\mathbf{x}_{\bar{S}}) = \min\{1, \max\{0, g_j(\mathbf{x}_{\bar{S}})\}\}, \qquad j \in [N]. \tag{75}$$

Both $\mathbf{h}^3$ and $\tilde{\boldsymbol{\alpha}}$ also have all coordinates in $[0,1]$. Each target value $\alpha_j(\mathbf{x}_{\bar{S}})$ already lies in $[0,1]$, so clipping cannot increase its approximation error. Together with (70), this gives

$$\sup_{\mathbf{x}_{\bar{S}}} \|\tilde{\boldsymbol{\alpha}}(\mathbf{x}_{\bar{S}}) - \boldsymbol{\alpha}(\mathbf{x}_{\bar{S}})\|_\infty \le \sup_{\mathbf{x}_{\bar{S}}} \|\mathbf{g}(\mathbf{x}_{\bar{S}}) - \boldsymbol{\alpha}(\mathbf{x}_{\bar{S}})\|_\infty \tag{76}$$

$$< \frac{\delta}{2(B+C)}. \tag{77}$$

This establishes (64).

Step 3. implementation in the two-branch architecture We now specify weights and biases for both branches to realize the component approximations through normalized interactions. The controlled branch preserves the factors $B + \mathbf{w}_j^\top \Phi_S(\mathbf{x}_S)$ while the uncontrolled branch computes $\tilde{\boldsymbol{\alpha}}$. Their interaction then produces the values $\tilde{\ell}_j(\mathbf{x}) + B - C$. Let m be a common even hidden width satisfying

$$m \ge 2N + \max\{H, N\} + 1. \tag{78}$$

The first $2N$ controlled coordinates hold the factors in adjacent pairs. In the first three uncontrolled layers, the corresponding coordinates are set to one to preserve these factors through the interactions preceding controlled layers $2, 3, 4$. The next $\max\{H, N\}$ uncontrolled coordinates are reserved for intermediate computations $\mathbf{h}^1, \mathbf{h}^2, \mathbf{h}^3$ from Step 2. The fourth uncontrolled layer places paired copies of the resulting coefficients $\tilde{\alpha}_j$ in the first $2N$ coordinates. The interaction preceding the fifth controlled mapping then multiplies these coefficients by the preserved factors. The last uncontrolled coordinate is fixed at one. The width m is shared by all hidden layers and may be chosen larger for the later minimum/maximum computation.

The uncontrolled hidden representations are chosen as

$$\mathbf{z}_{\bar{S}}^l = \begin{pmatrix} \mathbf{1}_{2N} \\ \mathbf{h}^l \\ \mathbf{0} \\ 1 \end{pmatrix}, \quad \mathbf{z}_{\bar{S}}^4 = \begin{pmatrix} \tilde{\alpha}_1 \\ \tilde{\alpha}_1 \\ \vdots \\ \tilde{\alpha}_N \\ \tilde{\alpha}_N \\ \mathbf{0} \\ 1 \end{pmatrix}, \qquad l = 1, 2, 3. \tag{79}$$

Each vector has m coordinates, with $\mathbf{0}$ denoting a zero vector of the required length. Here $\mathbf{h}^1, \mathbf{h}^2, \mathbf{h}^3$ are the vectors defined in Step 2. The computations from Step 2 can be implemented by the uncontrolled layers in (18). In the first layer, the choice

$$(W_{\bar{S}}^1)_{2N+s,:} = (A^1)_{s,:}/R, \quad (\mathbf{b}_{\bar{S}}^1)_{2N+s} = (\boldsymbol{\beta}^1)_s/R, \qquad s \in [H], \tag{80}$$

gives $(\mathbf{z}_{\bar{S}}^1)_{2N+s} = h_s^1$ by (71). The constant coordinates are obtained using zero weight rows, with biases one or zero as required.

The second layer reads the H auxiliary coordinates containing $\mathbf{h}^1$. The weights connecting these coordinates to the N auxiliary output coordinates form A^2, with corresponding bias vector $\boldsymbol{\beta}^2/R$. The ReLU activation therefore produces $\mathbf{h}^2$ as defined in (72).

The third layer computes each coordinate of $\mathbf{h}^3$ separately. For each $j \in [N]$, the corresponding auxiliary output uses only the input coordinate containing h_j^2, with weight $-R$ and bias one. Its value after ReLU is therefore $h_j^3 = \mathrm{ReLU}(1 - Rh_j^2)$, as in (73). In both layers, all weights outside these specified connections are zero.

In the fourth layer, the input coordinate $2N + j$ contains h_j^3 for each $j \in [N]$. The weights and biases are chosen as

$$(W_{\bar{S}}^4)_{2j-1,2N+j} = (W_{\bar{S}}^4)_{2j,2N+j} = -1, (\mathbf{b}_{\bar{S}}^4)_{2j-1} = (\mathbf{b}_{\bar{S}}^4)_{2j} = 1. \tag{81}$$

All other entries of $W^4_{\bar{S}}$ are zero. The two output coordinates therefore both equal $\mathrm{ReLU}(1 - h^3_j) = \tilde{\alpha}_j$ by (74). The last coordinate is set to one by choosing $(\mathbf{b}^4_{\bar{S}})_m = 1$, and all remaining biases are zero. Together, these choices give a fixed set of weights and biases that generates (79) for every $\mathbf{x}_{\bar{S}} \in \mathcal{X}_{\bar{S}}$.

These representations must also retain their values under the normalization used in the interactions. Every entry of these vectors belongs to $[0, 1]$, and the last entry equals one. Hence

$$\|\mathbf{z}^l_{\bar{S}}\|_\infty = 1, \quad \frac{\mathbf{z}^l_{\bar{S}}}{\max\{1, \|\mathbf{z}^l_{\bar{S}}\|_\infty\}} = \mathbf{z}^l_{\bar{S}}, \qquad l = 1, 2, 3, 4. \tag{82}$$

This verifies normalization of the entire representation, including the auxiliary coordinates. We now construct the controlled branch to generate the factors $B + \mathbf{w}_j^\top \Phi_S(\mathbf{x}_S)$ and preserve them while the uncontrolled branch computes $\tilde{\boldsymbol{\alpha}}$. In the first controlled layer, choose

$$(\check{W}^1_S)_{2j-1,:} = (\check{W}^1_S)_{2j,:} = \mathbf{w}_j^\top, \tag{83}$$

with bias B in these coordinates and zero rows and biases elsewhere. Every entry of $\check{W}^1_S$ has absolute value at most one, so $\|\check{W}^1_S\|_{1\to\infty} \le 1$. GroupSort with group size two preserves each equal pair. For controlled layers $l = 2, 3, 4$, choose diagonal weights equal to one in the first $2N$ positions and zero elsewhere, with zero biases. These matrices have maximum absolute row sum at most one. The preceding interactions multiply the first $2N$ inputs by one, and GroupSort again preserves equal pairs. Thus, for each of the first four controlled hidden representations,

$$(\mathbf{z}^l_S)_{2j-1} = (\mathbf{z}^l_S)_{2j} = B + \mathbf{w}_j^\top \Phi_S(\mathbf{x}_S), \tag{84}$$

and every remaining controlled coordinate is zero. At the interaction preceding the fifth controlled mapping,

$$(\mathbf{z}^5_{S,\mathrm{in}})_{2j-1} = (\mathbf{z}^5_{S,\mathrm{in}})_{2j} = \tilde{\alpha}_j(\mathbf{x}_{\bar{S}})\big(B + \mathbf{w}_j^\top \Phi_S(\mathbf{x}_S)\big) \tag{85}$$

$$= \tilde{\ell}_j(\mathbf{x}) + B - C, \tag{86}$$

and all remaining coordinates are zero. Together with (68) and (76), this proves the lemma. □

B.3.4 Proof of Lemma 4

Proof. GroupSort with a group size of two maps (a, b) to $(\min\{a, b\}, \max\{a, b\})$. A fully connected layer places the two values to be compared in an adjacent pair and copies any values needed later into equal pairs. GroupSort computes their minimum and maximum while preserving the equal pairs. A second fully connected layer copies the desired minimum or maximum into an equal pair and retains all other values needed later in equal pairs. The following GroupSort preserves these pairs.

Each row of the constructed weight matrices selects one input coordinate or is zero, so its absolute row sum is at most one. Copying a value into several output rows does not increase this bound. All these layers can have zero biases.

Every finite minimum/maximum expression can be evaluated through finitely many binary operations. The construction above can therefore be repeated in an order that computes each intermediate value before it is used, retaining needed values in separate pairs. Only finitely many values need to be retained, so a common finite even hidden width suffices, with unused coordinates set to zero. A final scalar-output layer with zero bias selects the result using a weight row of ℓ_1 norm one. The resulting network evaluates the expression exactly, and every weight matrix satisfies the required norm bound. □

B.4 Proof of Corollary 1

Proof. Fix $\mathbf{x}_{\bar{S}} \in \mathcal{X}_{\bar{S}}$ and consider two controlled inputs $\mathbf{x}_S, \mathbf{x}'_S \in \mathcal{X}_S$. Lemma 1 and (28) give

$$|\hat{f}(\mathbf{x}_S, \mathbf{x}_{\bar{S}}) - \hat{f}(\mathbf{x}'_S, \mathbf{x}_{\bar{S}})| \le \|\Phi_S(\mathbf{x}_S) - \Phi_S(\mathbf{x}'_S)\|_1 \tag{87}$$

$$= \sum_{i=1}^{d_S} \left| \int_{x'_i}^{x_i} L_i(u)\, \mathrm{d}u \right|. \tag{88}$$

The interval assumptions and local integrability of the functions L_i ensure that each integral is well defined. This proves (29).

For each controlled coordinate i, fix all remaining coordinates and consider $\hat{f}$ as a function of x_i. The preceding bound gives

$$|\hat{f}(\mathbf{x} + h\mathbf{e}_i) - \hat{f}(\mathbf{x})| \leq |\phi_i(x_i + h) - \phi_i(x_i)| \tag{89}$$

whenever $x_i, x_i + h \in \mathcal{X}_i$. Since L_i is locally integrable, ϕ_i is locally absolutely continuous and satisfies $\phi_i'(x_i) = L_i(x_i)$ almost everywhere. On each compact subinterval, summing (89) over finitely many disjoint intervals bounds the sum of absolute changes of $\hat{f}$ by the corresponding sum for ϕ_i. Thus, with the remaining coordinates fixed, $\hat{f}$ is locally absolutely continuous in x_i and hence differentiable almost everywhere.

At any interior value of x_i where $\partial \hat{f}(\mathbf{x})/\partial x_i$ exists and $\phi_i'(x_i) = L_i(x_i)$, dividing (89) by $|h|$ and taking $h \to 0$ yields

$$\left| \frac{\partial \hat{f}(\mathbf{x})}{\partial x_i} \right| \leq |\phi_i'(x_i)| = L_i(x_i), \tag{90}$$

where the last equality uses $L_i(x_i) \geq 0$. For each fixed choice of the remaining coordinates, these conditions hold for almost every $x_i \in \mathcal{X}_i$, proving (30). □

C Additional Experimental Details

C.1 Synthetic Datasets

We generate 1,000 observations for each synthetic dataset, with input coordinates sampled independently from the distributions specified below. Observations follow $y = f^*(\mathbf{x}) + \lambda\epsilon$, where $\epsilon \sim \mathcal{N}(0, 1)$ is independent of the inputs and λ controls the noise scale. Training and validation use these observations, while test error is computed against f^*. We write $(u)_+ \coloneqq \max\{u, 0\}$.

C.1.1 Constant and Region

Constant and Region datasets examine unequal feature-wise sensitivity scales with constant and region-dependent bounds, respectively. Their shared target and input settings are

$$f^*(\mathbf{x}_S, \mathbf{x}_{\bar{S}}) = (0.5 + 0.5v_1) \sum_{i=1}^{3} a_i Q(x_i) + 8(v_2 - 0.5), \tag{91}$$

where $\mathbf{x}_S = (x_1, x_2, x_3) \in [0, 1]^3$, $\mathbf{x}_{\bar{S}} = (v_1, v_2) \in [0, 1]^2$, and $\mathbf{a} = (16, 4, 1)$. Inputs are sampled uniformly over their indicated ranges. We use $Q(u) = u - 0.5$ for Constant and $Q(u) = \frac{1}{4}u + \frac{3}{4}(u - 0.25)_+ - \frac{3}{4}(u - 0.75)_+ - \frac{5}{16}$ for Region. The sensitivity bounds are obtained from the segment slopes of $a_i Q$, since the multiplier $0.5 + 0.5v_1$ is at most one.

C.1.2 Atomic layer deposition

Atomic layer deposition (ALD) is a synthetic dataset inspired by the exponential decay and persistent methyl coverage reported during water exposure in atomic layer deposition (Vandalon & Kessels, 2016). Its temperature dependence and parameter values are chosen to reflect these qualitative characteristics.

Exposure duration d is the single controlled input, with $\mathbf{x}_S = (d)$. Temperature T_C, relative water flux J, and initial coverage θ_0 form the three uncontrolled inputs, with $\mathbf{x}_{\bar{S}} = (T_C, J, \theta_0)$. Using temperature $T = T_C + 273.15$ in kelvin and duration d in milliseconds, we define the target through the residual fraction $r(T)$ and reaction rate $k(T)$ as

$$f^*(\mathbf{x}_S, \mathbf{x}_{\bar{S}}) = \theta_0[r(T) + (1 - r(T)) \exp(-k(T)Jd/1000)], \tag{92}$$

$$r(T) = \frac{r_{\max}}{1 + \exp((T - T_r)/\Delta_T)}, \quad k(T) = k_0 \exp\left[\frac{E_a}{R}\left(\frac{1}{T_0} - \frac{1}{T}\right)\right]. \tag{93}$$

Inputs are sampled uniformly over the ranges in Table 4, which also lists the fixed parameters. For fixed uncontrolled inputs, the absolute sensitivity to duration decreases exponentially, motivating successively smaller regional bounds.

Table 4: Input ranges and fixed parameters for the synthetic ALD dataset.

Symbol	Description	Range or value
Controlled input ($d_S = 1$)		
d	Exposure duration	$[0, 400]$ ms
Uncontrolled inputs ($d_{\bar{S}} = 3$)		
T_C	Temperature	$[100, 250]\,^\circ$C
J	Relative water flux	$[0.5, 1.5]$
θ_0	Initial methyl coverage	$[0.6, 1]$
Fixed parameters		
$r_{\max}$	Maximum residual fraction	0.35
T_r	Residual-fraction transition temperature	438.15 K
Δ_T	Temperature transition scale	15 K
k_0	Reaction rate at T_0 and $J = 1$	$10\,\mathrm{s}^{-1}$
T_0	Reference temperature	373.15 K
E_a	Activation-energy parameter	$15000\,\mathrm{J\,mol}^{-1}$
R	Gas constant	$8.314462618\,\mathrm{J\,mol}^{-1}\,\mathrm{K}^{-1}$

C.1.3 LINEARREGION

LinearRegion examines feature-specific regional requirements with different boundary locations. Its three controlled responses have an increasing slope, a decreasing slope, and a slope that first increases and then decreases, respectively, as illustrated in Figure 5. The target and input partition are

$$f^*(\mathbf{x}_S, \mathbf{x}_{\bar{S}}) = (0.7 + 0.3 \tanh v_1) \sum_{i=1}^{3} g_i(x_i) + 0.8 \sin v_1 + 0.5 v_2^2 - 0.3 v_1 v_2, \tag{94}$$

where $\mathbf{x}_S = (x_1, x_2, x_3) \in [0, 1]^3$, and $\mathbf{x}_{\bar{S}} = (v_1, v_2)$. Controlled inputs are sampled uniformly over $[0, 1]$, while uncontrolled inputs follow a standard normal distribution. The coordinate responses are

$$g_1(u) = 30u + 24(u - 0.5)_+, \tag{95}$$
$$g_2(u) = 100u - 60(u - 0.4)_+, \tag{96}$$
$$g_3(u) = 50u + 60(u - 0.3)_+ - 40(u - 0.7)_+. \tag{97}$$

The regional bounds equal the segment slopes of g_i, since the multiplier $0.7 + 0.3 \tanh v_1 \in (0.4, 1)$ keeps the controlled sensitivities within these bounds.

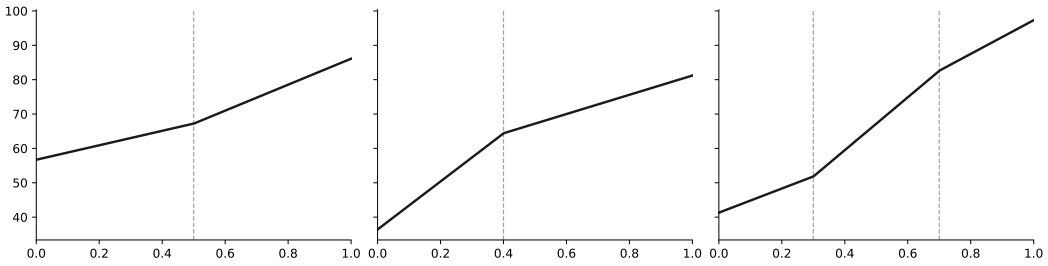

Figure 5: Three coordinate responses with region-wise sensitivity changes.

C.2 REAL-WORLD DATASETS

Specification of sensitivity requirements The real-data experiments evaluate predictive accuracy and constraint satisfaction under empirically specified sensitivity bounds. We use prior studies to identify controlled features and surrogate regressors to determine their bounds. The controlled features include acoustic frequency and flow speed for Airfoil (Brooks et al., 1989), vehicle weight

and engine characteristics for AutoMPG (Quinlan, 1993), ambient temperature for Combined Cycle Power Plant (CCPP) (Tüfekci, 2014), and curing age and cement content for Concrete (Yeh, 1998). The remaining predictors are treated as uncontrolled features. Rows containing missing or non-numeric values in the input features or target are excluded.

C.3 Sensitivity Bounds

For synthetic datasets, sensitivity bounds are derived from the known target functions to upper-bound their absolute partial derivatives wherever they exist within each specified region, uniformly over the remaining inputs. Table 5 summarizes these bounds in original units. For real-world datasets, we fit a surrogate regressor to each full dataset to specify empirical sensitivity requirements. We examine its absolute partial derivatives over fine intervals along each controlled coordinate and place region boundaries where these estimates change markedly. Within each region, we set the bound to the 95th percentile of the absolute derivative estimates. These bounds are fixed across methods and serve as empirical sensitivity requirements rather than certified bounds on the unknown target function. Table 6 reports the sample counts after cleaning, input dimensions, controlled features, and corresponding regions and bounds.

Table 5: Synthetic dataset dimensions and sensitivity bounds. Each dataset contains $1{,}000$ observations.

Dataset	# of features	Controlled feature	Intervals $\mathcal{I}_{i,r}$	Bounds $L_{i,r}$
Constant	5	x_1	$[0, 1]$	16
		x_2	$[0, 1]$	4
		x_3	$[0, 1]$	1
Region	5	x_1	$[0, 0.25], (0.25, 0.75], (0.75, 1]$	$4, 16, 4$
		x_2	$[0, 0.25], (0.25, 0.75], (0.75, 1]$	$1, 4, 1$
		x_3	$[0, 0.25], (0.25, 0.75], (0.75, 1]$	$0.25, 1, 0.25$
ALD	4	Duration d	$[0, 25], (25, 100], (100, 400]$	$0.061, 0.0154, 0.0035$
LinearRegion	5	x_1	$[0, 0.5], (0.5, 1]$	$30, 54$
		x_2	$[0, 0.4], (0.4, 1]$	$100, 40$
		x_3	$[0, 0.3], (0.3, 0.7], (0.7, 1]$	$50, 110, 70$

Table 6: Real-world dataset statistics and sensitivity bounds.

Dataset	# of Instances	# of features	Controlled feature x_i	Intervals $\mathcal{I}_{i,r}$	Bounds $L_{i,r}$
Airfoil	1,503	5	Frequency	$[200, 630], (630, 20000]$	$0.05, 0.012$
			Free-stream velocity	$[31.7, 71.3]$	0.25
AutoMPG	392	7	Weight	$[1613, 5140]$	0.015
			Horsepower	$[46, 230]$	0.2
			Displacement	$[68, 98], (98, 455]$	$0.15, 0.07$
CCPP	9,568	4	Ambient temperature	$[1.81, 37.11]$	2.5
Concrete	1,030	8	Age	$[1, 7], (7, 28], (28, 56], (56, 365]$	$2.7, 1.1, 0.7, 0.03$
			Cement	$[102, 540]$	0.2

C.4 Compared Methods

We compare FLNN with MLP (Rumelhart et al., 1986), AOL (Prach & Lampert, 2022), Cayley (Trockman & Kolter, 2021), SN-ℓ_1 (Gouk et al., 2021), SN-ℓ_2 (Miyato et al., 2018), Sandwich (Wang & Manchester, 2023), LMN (Nolte et al., 2023), PBLNN (Kinoshita & Toyoizumi, 2024), and XGBoost (Chen & Guestrin, 2016). MLP and XGBoost serve as unconstrained baselines. SN-ℓ_1 and SN-ℓ_2 normalize each weight matrix by its induced ℓ_1 and ℓ_2 norm, respectively.

MLP uses ReLU, while AOL, Cayley, SN-ℓ_1, SN-ℓ_2, and LMN use GroupSort. Sandwich uses its internal ReLU activation, and PBLNN uses Softplus. FLNN uses GroupSort in the controlled branch and ReLU in the uncontrolled branch. All GroupSort activations use group size two. We

choose GroupSort for its 1-Lipschitz and gradient-norm-preserving properties, which help retain expressivity under Lipschitz constraints (Anil et al., 2019). We choose neural network architectures with comparable parameter counts, as summarized in Table 7. For FLNN, the listed widths and depth apply to each branch.

LMN and the controlled branch of FLNN use $\ell_1 \to \ell_\infty$ norm constraints in the first layer and $\ell_\infty \to \ell_\infty$ norm constraints in subsequent layers, including the output layer. The uncontrolled branch of FLNN remains unconstrained. For PBLNN, we limit the internal solver to 500 iterations during training and $5{,}000$ during evaluation, with a tolerance of 10^{-3}.

Table 7: Neural-network architectures. GroupSort uses group size two.

Method	Hidden widths	Activation	Parameters (K)
AOL, Cayley	$[64, 64, 64, 64, 64]$	GroupSort	17.0–17.3
MLP	$[64, 64, 64, 64, 64]$	ReLU	17.0–17.3
SN-ℓ_1, SN-ℓ_2, LMN	$[64, 64, 64, 64, 64]$	GroupSort	17.0–17.3
Sandwich	$[64, 64, 32]$	Internal ReLU	16.0–16.2
PBLNN	$[64, 64]$	Softplus	17.5–18.2
FLNN	$[64, 64, 64]$ per branch	GroupSort / ReLU	17.2–17.4

C.5 Implementation and Hyperparameters

Training and model selection Controlled features are min-max scaled using their specified domain bounds, while uncontrolled features and targets are min-max scaled using training set statistics. Region boundaries and sensitivity bounds are transformed accordingly. All neural networks are implemented in PyTorch and trained on an NVIDIA GeForce RTX 5070 Ti GPU using Adam to minimize mean squared error. We use a batch size of 64 and train for at most 300 epochs. Early stopping monitors validation MSE with a patience of 30 epochs, and the checkpoint with the lowest validation MSE is retained for each configuration. Hyperparameters are selected by validation RMSE for each dataset and method. For synthetic datasets, training and validation use the noisy observations.

Hyperparameter search For neural networks, we search over learning rates in $\{10^{-4}, 10^{-3}, 5 \times 10^{-3}\}$ and weight decay values in $\{0, 10^{-4}\}$ using a fixed random seed. For AOL, Cayley, SN-ℓ_1, SN-ℓ_2, Sandwich, and LMN, we additionally search over the dataset-specific global Lipschitz bounds listed in Table 8. For PBLNN, we fix the inverse Lipschitz lower bound at $\alpha = 0$ and search over the same candidates for the Lipschitz upper bound β, with 1 included as an additional candidate. The Lipschitz parameters selected by validation RMSE for each dataset and method are reported in Table 9. For XGBoost, we search over 200, 300, and 500 boosting rounds and maximum tree depths of 2, 3, and 4, while fixing the learning rate to 0.01 and the ℓ_2 regularization coefficient for leaf weights to 1. The selected hyperparameters are kept fixed when training and evaluating each method with seeds 41 to 45.

Table 8: Dataset-specific candidates for the global Lipschitz bound and fixed noise standard deviations. A dash indicates that no additional noise is added.

Dataset	Global Lipschitz candidates	Fixed noise std. λ
Airfoil	$\{5, 10, 20\}$	–
AutoMPG	$\{20, 30, 40\}$	–
CCPP	$\{3, 5, 8\}$	–
Concrete	$\{20, 30, 40\}$	–
Constant	$\{0.5, 1.0, 1.5, 2.5\}$	0.215
Region	$\{0.5, 1.0, 1.75, 3.0\}$	0.18
ALD	$\{7, 14, 28\}$	0.01
LinearRegion	$\{0.5, 1.0, 1.75, 3.0\}$	1.5

Table 9: Selected model-specific parameters. Entries give the global Lipschitz parameter for AOL, Cayley, SN-ℓ_1, SN-ℓ_2, Sandwich, and LMN, and the Lipschitz upper bound β for PBLNN.

Method	Airfoil	AutoMPG	CCPP	Concrete	Constant	Region	ALD	LinearRegion
AOL	10	20	5	20	1.5	1.75	7	3
Cayley	5	20	3	20	1	1	7	1.75
SN-ℓ_1	5	20	8	20	1.5	1.75	14	1.75
SN-ℓ_2	10	20	3	30	1	1.75	7	1.75
Sandwich	10	30	8	40	1	1	7	3
LMN	20	30	3	20	2.5	3	14	3
PBLNN	5	1	3	40	2.5	3	1	3

C.6 Metric Computation

We compute both metrics separately for each selected seed-specific model and summarize their means and standard deviations across seeds 41 to 45.

Prediction error We compute prediction error using RMSE,

$$\mathrm{RMSE} = \left[\frac{1}{N}\sum_{n=1}^{N}\left(\hat{f}(\mathbf{x}^{(n)}) - y^{(n)}\right)^2\right]^{1/2}, \tag{98}$$

where N is the number of test samples and $y^{(n)}$ is the observed target. For synthetic data, $y^{(n)}$ is replaced by the noise-free target $f^*(\mathbf{x}^{(n)})$.

Constraint violations For N test inputs and d_S controlled features, the constraint violation rate is

$$\mathrm{CVR} = \frac{100}{N d_S}\sum_{n=1}^{N}\sum_{i=1}^{d_S}\mathbf{1}\{\hat{s}_i^{(n)} > L_i(x_i^{(n)}) + \tau\}, \tag{99}$$

where $\mathbf{1}\{\cdot\}$ is the indicator function, $\hat{s}_i^{(n)}$ is the estimated absolute slope along coordinate i, and $\tau = 10^{-5}$ is a numerical tolerance. For each test sample, $L_i(x_i^{(n)})$ equals the bound assigned to the interval containing $x_i^{(n)}$. Samples exactly on an internal boundary are assigned to the interval on its left. When feature i has a constant bound, the same value L_i is used for all test samples.

For neural networks, we compute the absolute partial derivative by automatic differentiation,

$$\hat{s}_i^{(n)} = \left|\partial_i \hat{f}(\mathbf{x}^{(n)})\right|. \tag{100}$$

For XGBoost, we perturb coordinate i in both directions using a nominal step equal to 10^{-3} times its specified domain width. Each perturbed coordinate is clipped to the endpoints of the sample's interval, with all other coordinates unchanged. Writing the resulting inputs as $\mathbf{x}^-$ and $\mathbf{x}^+$, we estimate the absolute slope as

$$\hat{s}_i = \frac{\left|\hat{f}(\mathbf{x}^+) - \hat{f}(\mathbf{x}^-)\right|}{x_i^+ - x_i^-}. \tag{101}$$

D Additional Experimental Results

Predictive performance and constraint satisfaction Table 10 reports RMSE and CVR on the synthetic datasets. FLNN achieves the lowest mean RMSE on Constant, Region, and LinearRegion and is the only evaluated method with zero observed CVR on these datasets. ALD provides a complementary setting with a smooth decay response inspired by atomic layer deposition and sensitivity bounds that decrease with exposure duration. FLNN attains a mean RMSE close to the lowest reported value while maintaining zero observed CVR. The ALD results illustrate the applicability of FLNN to a physically inspired target with regional sensitivity requirements.

Table 10: Test performance on the synthetic datasets, reported as mean ± standard deviation across seeds. **Bold** and underlined values indicate the best and second-best RMSEs for each dataset, respectively.

Method	Constant		Region		ALD		LinearRegion	
	RMSE	CVR	RMSE	CVR	RMSE	CVR	RMSE	CVR
XGBoost	0.611 ± 0.037	6.84 ± 1.04	0.441 ± 0.014	5.51 ± 0.91	0.083 ± 0.008	3.73 ± 1.80	5.131 ± 0.469	7.42 ± 1.94
MLP	0.169 ± 0.058	12.71 ± 0.96	0.170 ± 0.044	37.73 ± 2.67	0.078 ± 0.013	0.00 ± 0.00	$\underline{1.404 \pm 0.254}$	11.78 ± 1.83
SN-ℓ_1	0.219 ± 0.062	4.76 ± 4.59	0.437 ± 0.034	51.33 ± 2.39	0.133 ± 0.063	0.40 ± 0.89	6.201 ± 0.273	20.71 ± 11.07
SN-ℓ_2	$\underline{0.161 \pm 0.058}$	5.16 ± 2.01	$\underline{0.144 \pm 0.039}$	30.53 ± 3.80	0.079 ± 0.016	0.00 ± 0.00	1.447 ± 0.291	8.18 ± 1.53
Cayley	0.214 ± 0.073	16.89 ± 3.23	0.158 ± 0.030	31.47 ± 2.42	0.086 ± 0.011	0.13 ± 0.30	2.215 ± 0.154	14.76 ± 1.13
AOL	0.174 ± 0.053	8.27 ± 2.58	0.161 ± 0.046	33.82 ± 1.68	0.083 ± 0.013	0.00 ± 0.00	1.827 ± 0.174	15.69 ± 1.34
Sandwich	0.176 ± 0.072	8.13 ± 3.54	0.159 ± 0.034	29.29 ± 5.86	$\mathbf{0.076 \pm 0.009}$	0.00 ± 0.00	1.663 ± 0.371	12.84 ± 1.00
LMN	0.198 ± 0.053	0.44 ± 0.52	0.156 ± 0.018	34.80 ± 2.18	$\underline{0.077 \pm 0.013}$	0.00 ± 0.00	1.807 ± 0.424	10.44 ± 1.84
PBLNN	0.170 ± 0.065	1.96 ± 1.69	0.476 ± 0.037	51.20 ± 2.97	0.156 ± 0.008	0.00 ± 0.00	5.133 ± 0.989	30.13 ± 1.46
FLNN	$\mathbf{0.150 \pm 0.064}$	0.00 ± 0.00	$\mathbf{0.110 \pm 0.063}$	0.00 ± 0.00	0.078 ± 0.014	0.00 ± 0.00	$\mathbf{1.081 \pm 0.477}$	0.00 ± 0.00

Regional behavior on LinearRegion Figures 6 and 7 compare coordinate slices and maximum observed absolute partial derivatives on LinearRegion. FLNN closely follows all three target responses, capturing an increasing slope for x_1, a decreasing slope for x_2, and a slope that first increases and then decreases for x_3. The learned transitions occur near the corresponding feature-specific boundaries. The derivative measurements also show that FLNN remains within all regional bounds, whereas LMN and PBLNN exceed the bounds in several intervals with tighter requirements. These results extend the shared regional structure examined in Region to different boundary locations and slope patterns across features, showing that FLNN can incorporate these distinct requirements while accurately reproducing the displayed responses.

Together, the synthetic results illustrate that FLNN can incorporate sensitivity requirements ranging from constant feature-wise bounds to distinct regional bounds across multiple controlled features. The ALD and LinearRegion experiments further demonstrate its use with both smooth nonlinear responses and different patterns of regional slope changes, while preserving the specified sensitivity constraints.

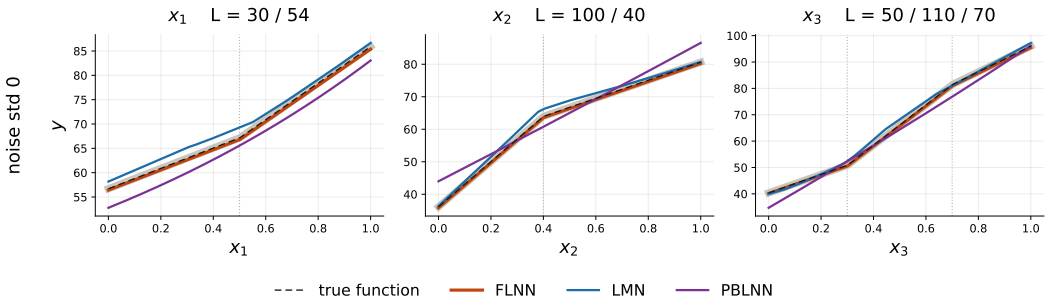

Figure 6: Coordinate slices of additional synthetic data

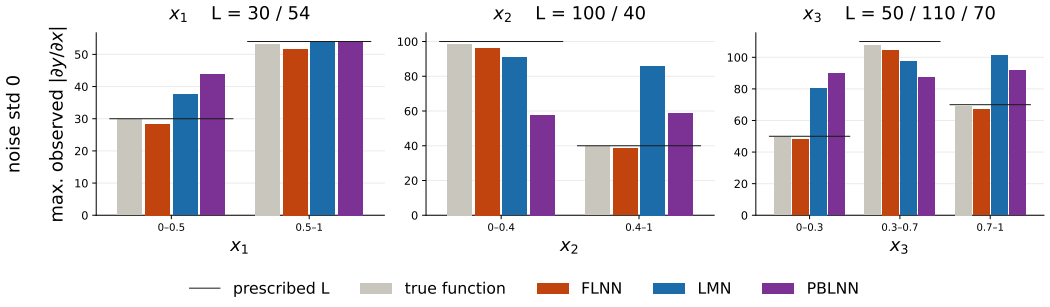

Figure 7: Maximum observed slopes of additional synthetic data