Expressivity and Complexity of ReLU Neural Netowrks

PhD Defense

Moritz Grillo

Technische Universität Berlin

October 02, 2025

Motivation

AI is moving fast

- Astonishing progress in artificial intelligence
- Neural networks are at the heart of this development
- Theoretical foundations far behind success in practice

Motivation

AI is moving fast

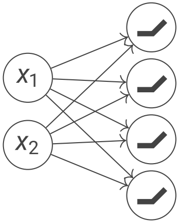
- Astonishing progress in artificial intelligence
- Neural networks are at the heart of this development
- Theoretical foundations far behind success in practice

This Thesis:

- What are the fundamental capabilities and limitations of neural networks?
- Focus on ReLU neural networks

ReLU-Layer and its Geometry

ReLU-layer with d input and m output neurons given by weight matrix $\mathbf{W} \in \mathbb{R}^{m \times d}$, bias vector $\mathbf{b} \in \mathbb{R}^{m}$

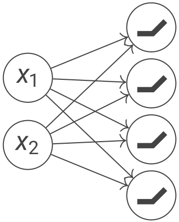


ReLU-Layer and its Geometry

ReLU-layer with d input and m output neurons given by weight matrix $\mathbf{W} \in \mathbb{R}^{m \times d}$, bias vector $\mathbf{b} \in \mathbb{R}^m$

Computes map

$$\phi_{\mathbf{W},\mathbf{b}} : \mathbb{R}^d \to \mathbb{R}^m, \quad \mathbf{x} \mapsto \max\{\mathbf{0}, \mathbf{W}\mathbf{x} + \mathbf{b}\}$$

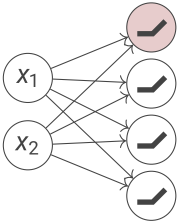


ReLU-Layer and its Geometry

ReLU-layer with d input and m output neurons given by weight matrix $\mathbf{W} \in \mathbb{R}^{m \times d}$, bias vector $\mathbf{b} \in \mathbb{R}^m$

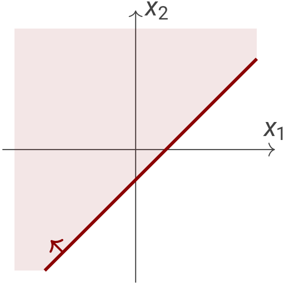
Computes map

$$\phi_{\mathbf{W},\mathbf{b}} : \mathbb{R}^d \to \mathbb{R}^m, \quad \mathbf{x} \mapsto \max\{\mathbf{0}, \mathbf{W}\mathbf{x} + \mathbf{b}\}$$



Terminology

- output neuron i active at $\mathbf{x} \in \mathbb{R}^d$ if $\mathbf{w}_i^T\mathbf{x} + b_i \geq 0$

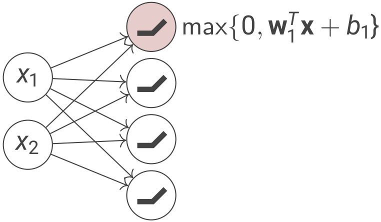


ReLU-Layer and its Geometry

ReLU-layer with d input and m output neurons given by weight matrix $\mathbf{W} \in \mathbb{R}^{m \times d}$, bias vector $\mathbf{b} \in \mathbb{R}^m$

Computes map

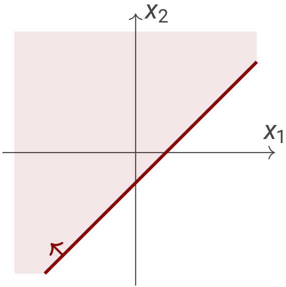
$$\phi_{\mathbf{W},\mathbf{b}} : \mathbb{R}^d \to \mathbb{R}^m, \quad \mathbf{x} \mapsto \max\{\mathbf{0}, \mathbf{W}\mathbf{x} + \mathbf{b}\}$$



Terminology

- output neuron i active at $\mathbf{x} \in \mathbb{R}^d$ if $\mathbf{w}_i^T\mathbf{x} + b_i \geq 0$
- output neurons induce (oriented) hyperplanes

$$H_{\mathbf{w}_i, b_i} := \{\mathbf{w}_i^T\mathbf{x} + b_i = 0\} \subseteq \mathbb{R}^d$$

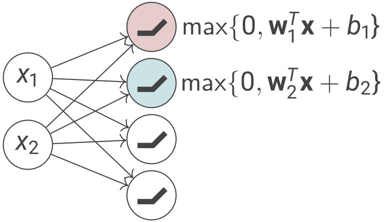


ReLU-Layer and its Geometry

ReLU-layer with d input and m output neurons given by weight matrix $\mathbf{W} \in \mathbb{R}^{m \times d}$, bias vector $\mathbf{b} \in \mathbb{R}^m$

Computes map

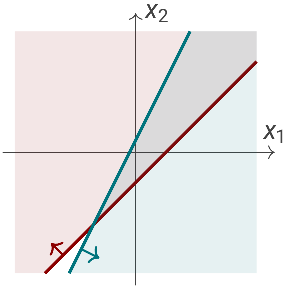
$$\phi_{\mathbf{W},\mathbf{b}} : \mathbb{R}^d \to \mathbb{R}^m, \quad \mathbf{x} \mapsto \max\{\mathbf{0}, \mathbf{W}\mathbf{x} + \mathbf{b}\}$$



Terminology

- output neuron i active at $\mathbf{x} \in \mathbb{R}^d$ if $\mathbf{w}_i^T\mathbf{x} + b_i \geq 0$
- output neurons induce (oriented) hyperplanes

$$H_{\mathbf{w}_i,b_i} := \{\mathbf{w}_i^T\mathbf{x} + b_i = 0\} \subseteq \mathbb{R}^d$$

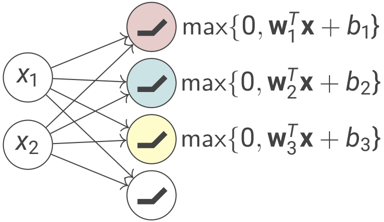


ReLU-Layer and its Geometry

ReLU-layer with d input and m output neurons given by weight matrix $\mathbf{W} \in \mathbb{R}^{m \times d}$, bias vector $\mathbf{b} \in \mathbb{R}^m$

Computes map

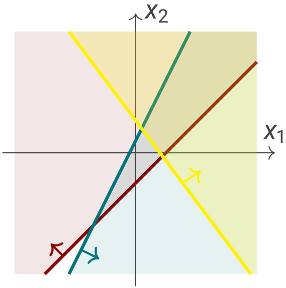
$$\phi_{\mathbf{W},\mathbf{b}} : \mathbb{R}^d \to \mathbb{R}^m, \quad \mathbf{x} \mapsto \max\{\mathbf{0}, \mathbf{W}\mathbf{x} + \mathbf{b}\}$$



Terminology

- output neuron i active at $\mathbf{x} \in \mathbb{R}^d$ if $\mathbf{w}_i^T\mathbf{x} + b_i \geq 0$
- output neurons induce (oriented) hyperplanes

$$H_{\mathbf{w}_i,b_i} := \{\mathbf{w}_i^T\mathbf{x} + b_i = 0\} \subseteq \mathbb{R}^d$$

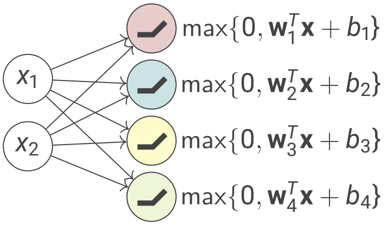


ReLU-Layer and its Geometry

ReLU-layer with d input and m output neurons given by weight matrix $\mathbf{W} \in \mathbb{R}^{m \times d}$, bias vector $\mathbf{b} \in \mathbb{R}^m$

Computes map

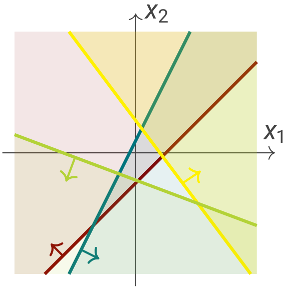
$$\phi_{\mathbf{W},\mathbf{b}} : \mathbb{R}^d \to \mathbb{R}^m, \quad \mathbf{x} \mapsto \max\{\mathbf{0}, \mathbf{W}\mathbf{x} + \mathbf{b}\}$$



Terminology

- output neuron i active at $\mathbf{x} \in \mathbb{R}^d$ if $\mathbf{w}_i^T\mathbf{x} + b_i \geq 0$
- output neurons induce (oriented) hyperplanes

$$H_{\mathbf{w}_i, b_i} := \{\mathbf{w}_i^T\mathbf{x} + b_i = 0\} \subseteq \mathbb{R}^d$$

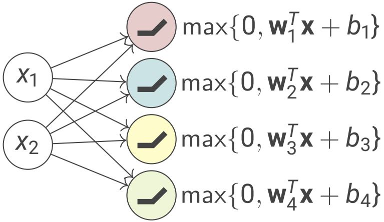


ReLU-Layer and its Geometry

ReLU-layer with d input and m output neurons given by weight matrix $\mathbf{W} \in \mathbb{R}^{m \times d}$, bias vector $\mathbf{b} \in \mathbb{R}^m$

Computes map

$$\phi_{\mathbf{W},\mathbf{b}} : \mathbb{R}^d \to \mathbb{R}^m, \quad \mathbf{x} \mapsto \max\{\mathbf{0}, \mathbf{W}\mathbf{x} + \mathbf{b}\}$$

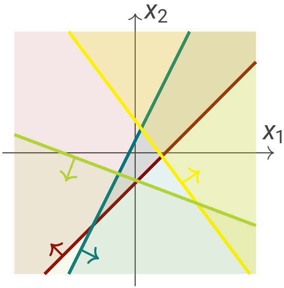


Terminology

- output neuron i active at $\mathbf{x} \in \mathbb{R}^d$ if $\mathbf{w}_i^T\mathbf{x} + b_i \geq 0$
- output neurons induce (oriented) hyperplanes

$$H_{\mathbf{w}_i,b_i} := \{\mathbf{w}_i^T\mathbf{x} + b_i = 0\} \subseteq \mathbb{R}^d$$

- they partition $\mathbb{R}^d$ into polyhedral cells, corresponding to subsets of active neurons

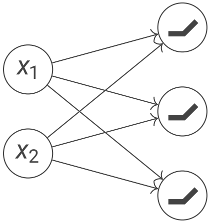


ReLU Neural Networks

- A ReLU network is a concatenation of such ReLU layers:

ReLU Neural Networks

- A ReLU network is a concatenation of such ReLU layers:



$f^{(1)}$

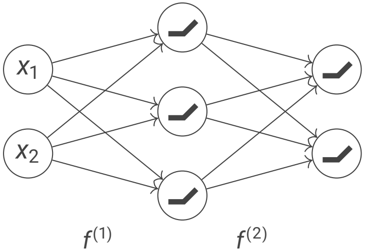
- A ReLU network computes a continuous piecewise linear (CPWL) function:

$$f = f^{(\ell+1)} \circ f^{(\ell)} \circ \cdots \circ f^{(1)}$$

where $f^{(i)}$ is a ReLU layer for $i \in [\ell]$

ReLU Neural Networks

- A ReLU network is a concatenation of such ReLU layers:



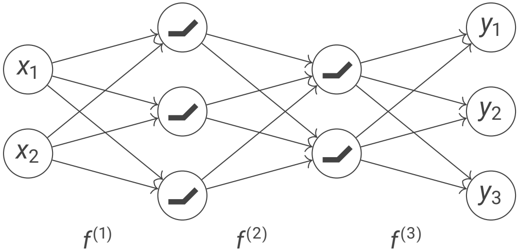
- A ReLU network computes a continuous piecewise linear (CPWL) function:

$$f = f^{(\ell+1)} \circ f^{(\ell)} \circ \cdots \circ f^{(1)}$$

where $f^{(i)}$ is a ReLU layer for $i \in [\ell]$

ReLU Neural Networks

- A ReLU network is a concatenation of such ReLU layers:



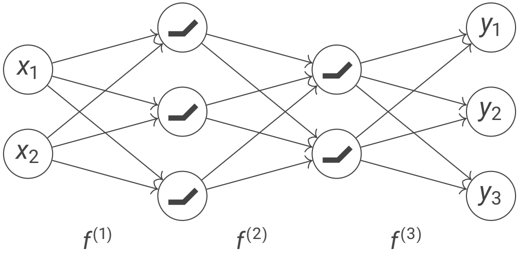
- A ReLU network computes a continuous piecewise linear (CPWL) function:

$$f = f^{(\ell+1)} \circ f^{(\ell)} \circ \cdots \circ f^{(1)}$$

where $f^{(i)}$ is a ReLU layer for $i \in [\ell]$ and $f^{(\ell+1)}$ is the affine linear output layer.

ReLU Neural Networks

- A ReLU network is a concatenation of such ReLU layers:



Architecture

- 2 hidden layers
- depth 3
- width 3
- size 5

- A ReLU network computes a continuous piecewise linear (CPWL) function:

$$f = f^{(\ell+1)} \circ f^{(\ell)} \circ \cdots \circ f^{(1)}$$

where $f^{(i)}$ is a ReLU layer for $i \in [\ell]$ and $f^{(\ell+1)}$ is the affine linear output layer.

Overview:

Machine learning:

- A neural network is a parameterized function $f_\theta : \mathbb{R}^d \to \mathbb{R}^m$ with $\theta = (\mathbf{W}^{(1)}, \mathbf{b}^{(1)}, \dots, \mathbf{W}^{(\ell)}, \mathbf{b}^{(\ell)})$

Overview:

Machine learning:

- A neural network is a parameterized function $f_\theta : \mathbb{R}^d \to \mathbb{R}^m$ with $\theta = (\mathbf{W}^{(1)}, \mathbf{b}^{(1)}, \ldots, \mathbf{W}^{(\ell)}, \mathbf{b}^{(\ell)})$
- Model given data $\mathcal{D}$ with a neural network
  1. Choose an architecture (depth and width)
  2. Training = optimizing θ to fit f_θ to the data $\mathcal{D}$

Overview:

Machine learning:

- A neural network is a parameterized function $f_\theta : \mathbb{R}^d \to \mathbb{R}^m$ with $\theta = (\mathbf{W}^{(1)}, \mathbf{b}^{(1)}, \ldots, \mathbf{W}^{(\ell)}, \mathbf{b}^{(\ell)})$
- Model given data $\mathcal{D}$ with a neural network
  1. Choose an architecture (depth and width)
  2. Training = optimizing θ to fit f_θ to the data $\mathcal{D}$

Questions:

1. Which functions are representable with a given architecture? $\to$ Expressivity

Overview:

Machine learning:

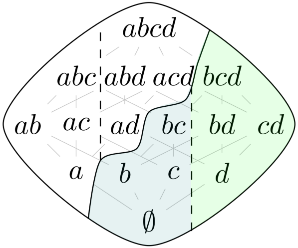
- A neural network is a parameterized function $f_\theta : \mathbb{R}^d \to \mathbb{R}^m$ with $\theta = (\mathbf{W}^{(1)}, \mathbf{b}^{(1)}, \ldots, \mathbf{W}^{(\ell)}, \mathbf{b}^{(\ell)})$
- Model given data $\mathcal{D}$ with a neural network
  1. Choose an architecture (depth and width)
  2. Training = optimizing θ to fit f_θ to the data $\mathcal{D}$

Questions:

1. Which functions are representable with a given architecture? $\to$ Expressivity
2. Given θ, what can we say about the properties of f_θ? $\to$ Verification

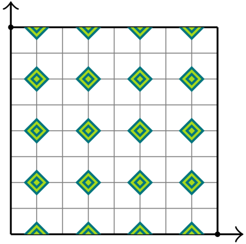
Expressivity

Representing CPWL functions



Based on *Depth-Bounds via the Braid Arrangement* with Christoph Hertrich and Georg Loho

Topological Expressivity



Based on *Topological Expressivity of ReLU Neural Networks* with Ekin Ergen

Representing CPWL Functions

Theorem [Arora, Basu, Mianjy, Mukherjee, 2018]
Every CPWL function $f : \mathbb{R}^d \to \mathbb{R}$ can be represented by a ReLU NN with $\lceil \log_2(d+1) \rceil$ hidden layers.

Representing CPWL Functions

Theorem [Arora, Basu, Mianjy, Mukherjee, 2018]
Every CPWL function $f: \mathbb{R}^d \to \mathbb{R}$ can be represented by a ReLU NN with $\lceil \log_2(d+1) \rceil$ hidden layers.

- How many layers are necessary to represent all CPWL function?

Representing CPWL Functions

Theorem [Arora, Basu, Mianjy, Mukherjee, 2018]
Every CPWL function $f\colon \mathbb{R}^d \to \mathbb{R}$ can be represented by a ReLU NN with $\lceil \log_2(d+1) \rceil$ hidden layers.

- How many layers are necessary to represent all CPWL function?
- There is no function that needs more layers to be represented than the function $\max\{0, x_1, \ldots, x_d\}$ [Hertrich, Basu, Di Summa, and Skutella, 2021].

Representing CPWL Functions

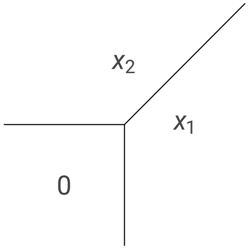
Theorem [Arora, Basu, Mianjy, Mukherjee, 2018]
Every CPWL function $f\colon \mathbb{R}^d \to \mathbb{R}$ can be represented by a ReLU NN with $\lceil \log_2(d+1) \rceil$ hidden layers.

- How many layers are necessary to represent all CPWL function?
- There is no function that needs more layers to be represented than the function $\max\{0, x_1, \ldots, x_d\}$ [Hertrich, Basu, Di Summa, and Skutella, 2021].
- $\max\{0, x_1, \ldots, x_4\}$ can be represented with 2 hidden layers
  $\rightarrow$ Upper bound improved to $\approx \log_3(d)$ hidden layers.
  [Bakaev, Brunck, Hertrich, Stade, Yehudayoff, 2025]

Lower Bounds?

General lower bound

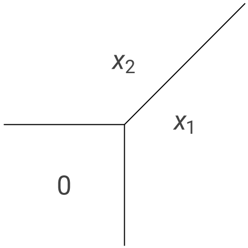
- $\max\{0, x_1, x_2\}$ cannot be represented with one hidden layer [Basu, Mukherjee, 2017]



Lower Bounds?

General lower bound

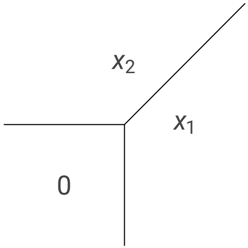
- $\max\{0, x_1, x_2\}$ cannot be represented with one hidden layer [Basu, Mukherjee, 2017]
- 2 is best known lower bound to compute all CPWL functions!



Lower Bounds?

General lower bound

- $\max\{0, x_1, x_2\}$ cannot be represented with one hidden layer [Basu, Mukherjee, 2017]
- 2 is best known lower bound to compute all CPWL functions!



Restricting the weights

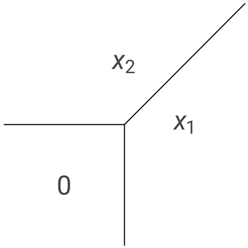
The function $\max\{0, x_1, \ldots x_d\}$ needs

- $\lceil \log_2(d+1) \rceil$ many hidden layers to be represented with integer weights. [Haase, Hertrich, Loho, 23]

Lower Bounds?

General lower bound

- $\max\{0, x_1, x_2\}$ cannot be represented with one hidden layer [Basu, Mukherjee, 2017]
- 2 is best known lower bound to compute all CPWL functions!

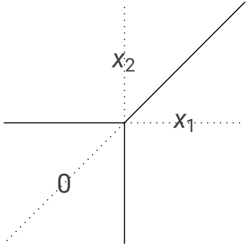


Restricting the weights

The function $\max\{0, x_1, \ldots x_d\}$ needs

- $\lceil \log_2(d+1) \rceil$ many hidden layers to be represented with integer weights. [Haase, Hertrich, Loho, 23]
- $\lceil \log_p(d+1) \rceil$ many hidden layers to be represented with rational weights if all denominators are not divisible by p. [Averkov, Hojny, Merkert, 25]

Lower Bounds?



General lower bound

- $\max\{0, x_1, x_2\}$ cannot be represented with one hidden layer [Basu, Mukherjee, 2017]
- 2 is best known lower bound to compute all CPWL functions!

Restricting the weights

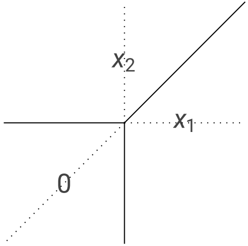
The function $\max\{0, x_1, \ldots x_d\}$ needs

- $\lceil \log_2(d+1) \rceil$ many hidden layers to be represented with integer weights. [Haase, Hertrich, Loho, 23]
- $\lceil \log_p(d+1) \rceil$ many hidden layers to be represented with rational weights if all denominators are not divisible by p. [Averkov, Hojny, Merkert, 25]

Restricting the Breakpoints

- Nice networks: All layers only have breakpoints where $x_i = x_j$

Lower Bounds?



General lower bound

- $\max\{0, x_1, x_2\}$ cannot be represented with one hidden layer [Basu, Mukherjee, 2017]
- 2 is best known lower bound to compute all CPWL functions!

Restricting the weights

The function $\max\{0, x_1, \ldots x_d\}$ needs

- $\lceil \log_2(d+1) \rceil$ many hidden layers to be represented with integer weights. [Haase, Hertrich, Loho, 23]
- $\lceil \log_p(d+1) \rceil$ many hidden layers to be represented with rational weights if all denominators are not divisible by p. [Averkov, Hojny, Merkert, 25]

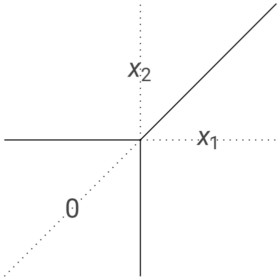
Restricting the Breakpoints

- Nice networks: All layers only have breakpoints where $x_i = x_j$
- Computational proof: $\max\{0, x_1, \ldots, x_4\}$ not representable with networks with 2 hidden layers [HBDS21].

Lower Bound for Nice Networks

Theorem [G., Hertrich, Loho, 25]

Nice networks cannot compute $\max\{0, x_1, \ldots, x_d\}$ with $\lceil \log_2 \log_2(d+1) \rceil - 1$ hidden layers.



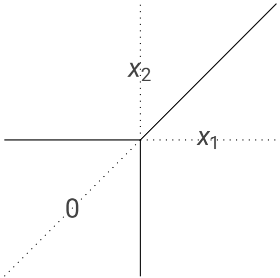
Lower Bound for Nice Networks

Theorem [G., Hertrich, Loho, 25]

Nice networks cannot compute $\max\{0, x_1, \ldots, x_d\}$ with $\lceil \log_2 \log_2(d+1) \rceil - 1$ hidden layers.

Proof ingredients

- $\mathcal{V}(\ell) =$ Nice network functions with ℓ hidden layers



Lower Bound for Nice Networks

Theorem [G., Hertrich, Loho, 25]

Nice networks cannot compute $\max\{0, x_1, \ldots, x_d\}$ with $\lceil \log_2 \log_2(d+1) \rceil - 1$ hidden layers.

Proof ingredients

- $\mathcal{V}(\ell) =$ Nice network functions with ℓ hidden layers
- Goal: Show that $\max\{0, x_1, \ldots, x_{2^{2^\ell}}\} \notin \mathcal{V}(\ell)$.

Lower Bound for Nice Networks

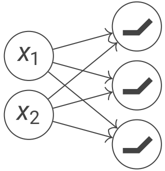
Theorem [G., Hertrich, Loho, 25]

Nice networks cannot compute $\max\{0, x_1, \dots, x_d\}$ with $\lceil \log_2 \log_2(d+1) \rceil - 1$ hidden layers.

Proof ingredients

- $\mathcal{V}(\ell)$ = Nice network functions with ℓ hidden layers
- Goal: Show that $\max\{0, x_1, \dots, x_{2^{2^\ell}}\} \notin \mathcal{V}(\ell)$.
- $\mathcal{V}(\ell+1) = \mathit{addlayer}(\mathcal{V}(\ell)) :=$ $\operatorname{span}\{\max\{0, f\} \mid f \in V(\ell), \max\{0, f\} \text{ nice }\}$.

nice



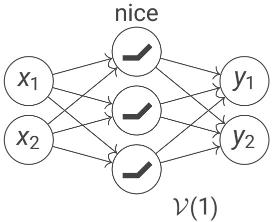
Lower Bound for Nice Networks

Theorem [G., Hertrich, Loho, 25]

Nice networks cannot compute $\max\{0, x_1, \ldots, x_d\}$ with $\lceil \log_2 \log_2(d+1) \rceil - 1$ hidden layers.

Proof ingredients

- $\mathcal{V}(\ell)$ = Nice network functions with ℓ hidden layers
- Goal: Show that $\max\{0, x_1, \ldots, x_{2^{2^\ell}}\} \notin \mathcal{V}(\ell)$.
- $\mathcal{V}(\ell+1) = \textit{addlayer}(\mathcal{V}(\ell)) :=$ $\text{span}\{\max\{0, f\} \mid f \in V(\ell), \max\{0, f\} \text{ nice }\}$.



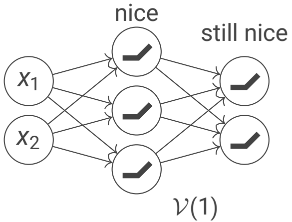
Lower Bound for Nice Networks

Theorem [G., Hertrich, Loho, 25]

Nice networks cannot compute $\max\{0, x_1, \ldots, x_d\}$ with $\lceil \log_2 \log_2(d+1) \rceil - 1$ hidden layers.

Proof ingredients

- $\mathcal{V}(\ell)$ = Nice network functions with ℓ hidden layers
- Goal: Show that $\max\{0, x_1, \ldots, x_{2^{2^\ell}}\} \notin \mathcal{V}(\ell)$.
- $\mathcal{V}(\ell+1) = \mathit{addlayer}(\mathcal{V}(\ell)) :=$ $\mathrm{span}\{\max\{0, f\} \mid f \in V(\ell), \max\{0, f\} \text{ nice }\}$.



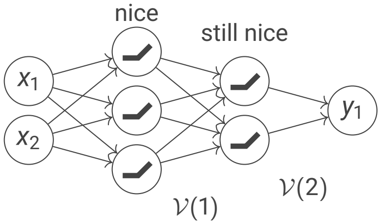
Lower Bound for Nice Networks

Theorem [G., Hertrich, Loho, 25]

Nice networks cannot compute $\max\{0, x_1, \ldots, x_d\}$ with $\lceil \log_2 \log_2(d+1) \rceil - 1$ hidden layers.

Proof ingredients

- $\mathcal{V}(\ell)$ = Nice network functions with ℓ hidden layers
- Goal: Show that $\max\{0, x_1, \ldots, x_{2^{2^\ell}}\} \notin \mathcal{V}(\ell)$.
- $\mathcal{V}(\ell+1) = \mathit{addlayer}(\mathcal{V}(\ell)) :=$ $\operatorname{span}\{\max\{0, f\} \mid f \in V(\ell), \max\{0, f\} \text{ nice }\}$.



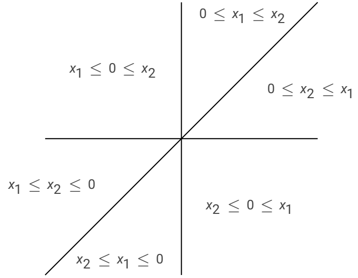
Lower Bound for Nice Networks

Theorem [G., Hertrich, Loho, 25]

Nice networks cannot compute $\max\{0, x_1, \ldots, x_d\}$ with $\lceil \log_2 \log_2(d+1) \rceil - 1$ hidden layers.

Proof ingredients

- $\mathcal{V}(\ell)$ = Nice network functions with ℓ hidden layers
- Goal: Show that $\max\{0, x_1, \ldots, x_{2^{2^\ell}}\} \notin \mathcal{V}(\ell)$.
- $\mathcal{V}(\ell+1) = \mathit{addlayer}(\mathcal{V}(\ell)) :=$ $\operatorname{span}\{\max\{0, f\} \mid f \in V(\ell), \max\{0, f\} \text{ nice }\}$.
- Nice networks compatible with braid fan



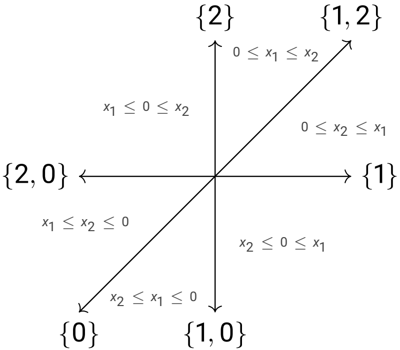
Lower Bound for Nice Networks

Theorem [G., Hertrich, Loho, 25]

Nice networks cannot compute $\max\{0, x_1, \ldots, x_d\}$ with $\lceil \log_2 \log_2(d+1) \rceil - 1$ hidden layers.

Proof ingredients

- $\mathcal{V}(\ell)$ = Nice network functions with ℓ hidden layers
- Goal: Show that $\max\{0, x_1, \ldots, x_{2^{2^\ell}}\} \notin \mathcal{V}(\ell)$.
- $\mathcal{V}(\ell+1) = \mathit{addlayer}(\mathcal{V}(\ell)) :=$ $\operatorname{span}\{\max\{0, f\} \mid f \in V(\ell), \max\{0, f\} \text{ nice }\}$.
- Nice networks compatible with braid fan
- Nice networks $\simeq$ set functions $\mathcal{F}$.



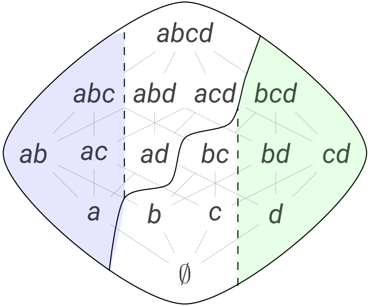
Lower Bound for Nice Networks

Theorem [G., Hertrich, Loho, 25]

Nice networks cannot compute $\max\{0, x_1, \ldots, x_d\}$ with $\lceil \log_2 \log_2(d+1) \rceil - 1$ hidden layers.

Proof ingredients

- $\mathcal{V}(\ell)$ = Nice network functions with ℓ hidden layers
- Goal: Show that $\max\{0, x_1, \ldots, x_{2^{2^\ell}}\} \notin \mathcal{V}(\ell)$.
- $\mathcal{V}(\ell+1) = \mathit{addlayer}(\mathcal{V}(\ell)) :=$ $\mathrm{span}\{\max\{0, f\} \mid f \in V(\ell), \max\{0, f\} \text{ nice }\}$.
- Nice networks compatible with braid fan
- Nice networks $\simeq$ set functions $\mathcal{F}$.
- Nested sequence of vector subspaces $\mathcal{V}(0) = \mathcal{F}(1) \subsetneq \cdots \subsetneq \mathcal{F}(2^{2^\ell}+1) = \mathcal{F}$ via lattices



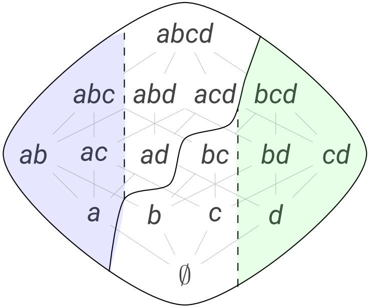
Lower Bound for Nice Networks

Theorem [G., Hertrich, Loho, 25]

Nice networks cannot compute $\max\{0, x_1, \ldots, x_d\}$ with $\lceil \log_2 \log_2(d+1) \rceil - 1$ hidden layers.

Proof ingredients

- $\mathcal{V}(\ell) =$ Nice network functions with ℓ hidden layers
- Goal: Show that $\max\{0, x_1, \ldots, x_{2^{2^\ell}}\} \notin \mathcal{V}(\ell)$.
- $\mathcal{V}(\ell+1) = \mathit{addlayer}(\mathcal{V}(\ell)) :=$ $\mathrm{span}\{\max\{0, f\} \mid f \in V(\ell), \max\{0, f\} \text{ nice }\}$.
- Nice networks compatible with braid fan
- Nice networks $\simeq$ set functions $\mathcal{F}$.
- Nested sequence of vector subspaces $\mathcal{V}(0) = \mathcal{F}(1) \subsetneq \cdots \subsetneq \mathcal{F}(2^{2^\ell}+1) = \mathcal{F}$ via lattices
- Key technical lemma: $\mathit{addlayer}(\mathcal{F}(k)) \subseteq \mathcal{F}(k^2+k)$.



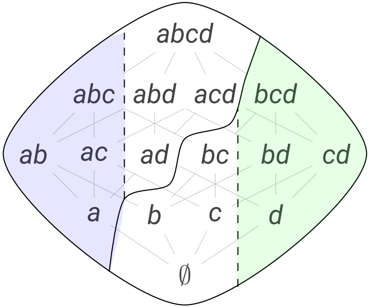
Lower Bound for Nice Networks

Theorem [G., Hertrich, Loho, 25]

Nice networks cannot compute $\max\{0, x_1, \ldots, x_d\}$ with $\lceil \log_2 \log_2(d+1) \rceil - 1$ hidden layers.

Proof ingredients

- $\mathcal{V}(\ell)$ = Nice network functions with ℓ hidden layers
- Goal: Show that $\max\{0, x_1, \ldots, x_{2^{2^\ell}}\} \notin \mathcal{V}(\ell)$.
- $\mathcal{V}(\ell+1) = \mathit{addlayer}(\mathcal{V}(\ell)) :=$ $\operatorname{span}\{\max\{0, f\} \mid f \in V(\ell), \max\{0, f\} \text{ nice }\}$.
- Nice networks compatible with braid fan
- Nice networks $\simeq$ set functions $\mathcal{F}$.
- Nested sequence of vector subspaces $\mathcal{V}(0) = \mathcal{F}(1) \subsetneq \cdots \subsetneq \mathcal{F}(2^{2^\ell} + 1) = \mathcal{F}$ via lattices
- Key technical lemma: $\mathit{addlayer}(\mathcal{F}(k)) \subseteq \mathcal{F}(k^2 + k)$.
- Iterating this yields $\mathcal{V}(\ell) \subseteq \mathcal{F}(2^{2^\ell}) \subsetneq \mathcal{F}$



It Remains Open...

Theorem [G., Hertrich, Loho, 25]

Nice networks cannot compute $\max\{0, x_1, \ldots, x_4\}$ with 2 hidden layers

It Remains Open...

Theorem [G., Hertrich, Loho, 25]

Nice networks cannot compute $\max\{0, x_1, \ldots, x_4\}$ with 2 hidden layers

But:

$\max\{0, x_1, \ldots, x_4\}$ can be computed with 2 hidden layers [BBHSY25]

$\rightarrow$ Restricting to nice networks is a real restriction.

It Remains Open...

Theorem [G., Hertrich, Loho, 25]

Nice networks cannot compute $\max\{0, x_1, \ldots, x_4\}$ with 2 hidden layers

But:
$\max\{0, x_1, \ldots, x_4\}$ can be computed with 2 hidden layers [BBHSY25]
$\rightarrow$ Restricting to nice networks is a real restriction.

Open Questions:

- How many layers are necessary to compute all CPWL functions?
- Is there a function that needs more than 2 hidden layers?

Expressivity in Binary Classification

- Given $\{(\mathbf{x}_i, y_i)\}_{i \in I}$ with $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$

Expressivity in Binary Classification

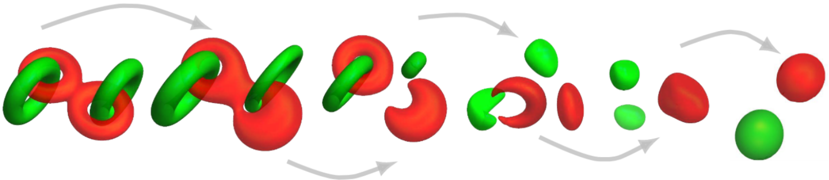
- Given $\{(\mathbf{x}_i, y_i)\}_{i \in I}$ with $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$
- f classifies data if $f(\mathbf{x}_i) \leq 0 \iff y_i = -1$

Expressivity in Binary Classification

- Given $\{(\mathbf{x}_i, y_i)\}_{i \in I}$ with $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$
- f classifies data if $f(\mathbf{x}_i) \leq 0 \iff y_i = -1$
- Topological data analysis: Data can have nontrivial topology

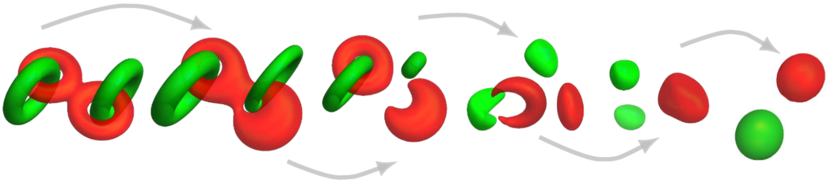
Expressivity in Binary Classification

- Given $\{(\mathbf{x}_i, y_i)\}_{i \in I}$ with $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$
- f classifies data if $f(\mathbf{x}_i) \leq 0 \iff y_i = -1$
- Topological data analysis: Data can have nontrivial topology
- Empirical insights [Naitzat et al., 2020]:
  Neural networks operate by changing topology, transforming a topologically complicated data set into a topologically simple one as it passes through the layers



Expressivity in Binary Classification

- Given $\{(\mathbf{x}_i, y_i)\}_{i \in I}$ with $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$
- f classifies data if $f(\mathbf{x}_i) \leq 0 \iff y_i = -1$
- Topological data analysis: Data can have nontrivial topology
- Empirical insights [Naitzat et al., 2020]:
  Neural networks operate by changing topology, transforming a topologically complicated data set into a topologically simple one as it passes through the layers



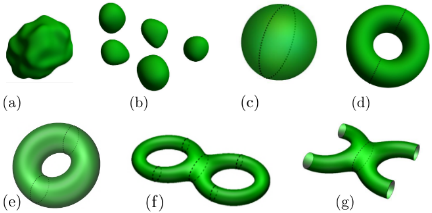
Question:
Given an architecture, how topologically complex can the decision regions $f^{-1}((-\infty, 0])$ become?

How to Quantify the Topological Complexity?

- Use Betti numbers as complexity measure for topological space M

How to Quantify the Topological Complexity?

- Use Betti numbers as complexity measure for topological space M
- Intuitively: the k-th Betti number $\beta_k(M)$ is the number of $(k+1)$-dimensional holes in M and $\beta_0(M)$ is the number of connected components.

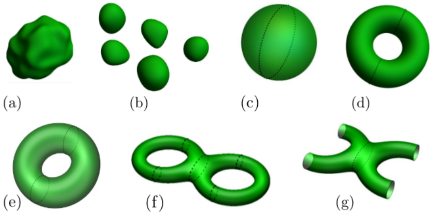


	Manifold $M \subseteq \mathbb{R}^3$	$\beta(M)$
(a)	Single contractible manifold	$(1, 0, 0)$
(b)	Five contractible manifolds	$(5, 0, 0)$
(c)	Sphere	$(1, 0, 1)$
(d)	Solid torus (filled)	$(1, 1, 0)$
(e)	Surface of torus (hollow)	$(1, 2, 1)$
(f)	Genus two surface (hollow)	$(1, 4, 1)$
(g)	Torso surface (hollow)	$(1, 3, 0)$

Figure 2. Naitzat et al.

How to Quantify the Topological Complexity?

- Use Betti numbers as complexity measure for topological space M
- Intuitively: the k-th Betti number $\beta_k(M)$ is the number of $(k+1)$-dimensional holes in M and $\beta_0(M)$ is the number of connected components.



	Manifold $M \subseteq \mathbb{R}^3$	$\beta(M)$
(a)	Single contractible manifold	$(1, 0, 0)$
(b)	Five contractible manifolds	$(5, 0, 0)$
(c)	Sphere	$(1, 0, 1)$
(d)	Solid torus (filled)	$(1, 1, 0)$
(e)	Surface of torus (hollow)	$(1, 2, 1)$
(f)	Genus two surface (hollow)	$(1, 4, 1)$
(g)	Torso surface (hollow)	$(1, 3, 0)$

Figure 2. Naitzat et al.

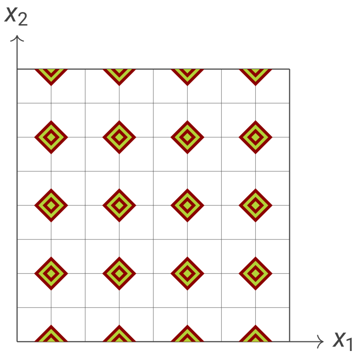
- Topological expressivity of neural network f measured by $\beta_k(f^{-1}((-\infty, 0])$

Topological Expressivity

Theorem[Ergen, G., 24]

For any depth and width, there is a neural network f such that for all $k = 0, \ldots, d-1$ holds

$$\beta_k(f^{-1}((-\infty, 0]) \geq \frac{width}{d}^{(depth-2)\times k}$$

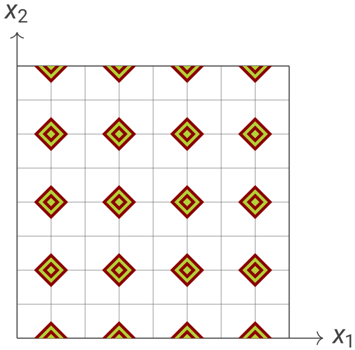


Topological Expressivity

Theorem[Ergen, G., 24]

For any depth and width, there is a neural network f such that for all $k = 0, \ldots, d-1$ holds

$$\beta_k(f^{-1}((-\infty, 0]) \geq \frac{width}{d}^{(depth-2)\times k}$$



Theorem[Ergen, G., 24]

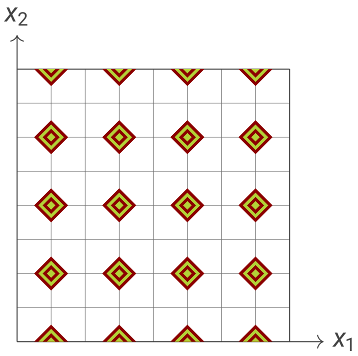
It holds that $\beta_k(f^{-1}((-\infty, 0]) \leq width^{depth \times d^2}$ for all $k = 0, \ldots, d-1$.

Topological Expressivity

Theorem[Ergen, G., 24]

For any depth and width, there is a neural network f such that for all $k = 0, \ldots, d-1$ holds

$$\beta_k(f^{-1}((-\infty, 0])) \geq \frac{width}{d}^{(depth-2)\times k}$$



Theorem[Ergen, G., 24]

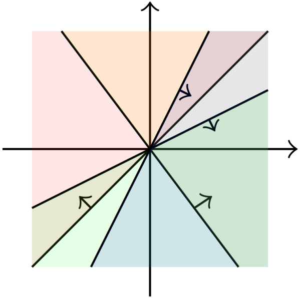
It holds that $\beta_k(f^{-1}((-\infty, 0]) \leq width^{depth \times d^2}$ for all $k = 0, \ldots, d-1$.

Conclusion

Deep neural networks are better equipped to model topological complex data sets.

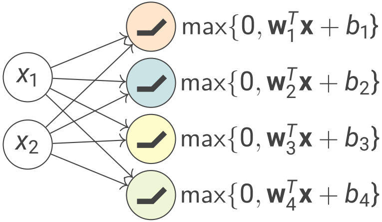
Verification

Complexity of Injectivity and Verification

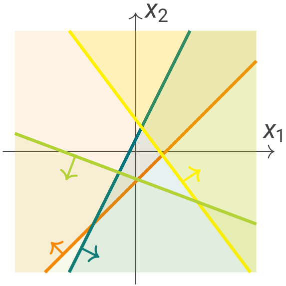


Based on *Complexity of Injectivity and Verification of ReLU Neural Networks* with Vincent Froese and Martin Skutella

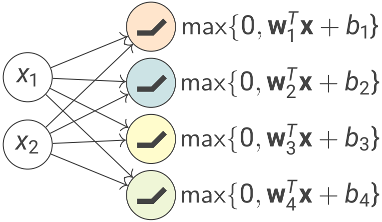
ReLU-Layer: Injectivity



RELU-LAYER INJECTIVITY
Given: matrix $\mathbf{W} \in \mathbb{R}^{m \times d}$, vector $\mathbf{b} \in \mathbb{R}^m$
Question: is the map $\phi_{\mathbf{W},\mathbf{b}}$ injective?

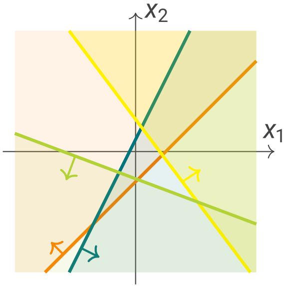


ReLU-Layer: Injectivity

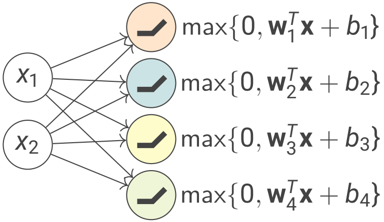


RELU-LAYER INJECTIVITY
Given: matrix $\mathbf{W} \in \mathbb{R}^{m \times d}$, vector $\mathbf{b} \in \mathbb{R}^m$
Question: is the map $\phi_{\mathbf{W},\mathbf{b}}$ injective?

Theorem [Puthawala et al., 2022] $\phi_{\mathbf{W},\mathbf{b}}$ injective $\iff$ for every cell, active neurons (rows of $\mathbf{W}$) have rank d



ReLU-Layer: Injectivity

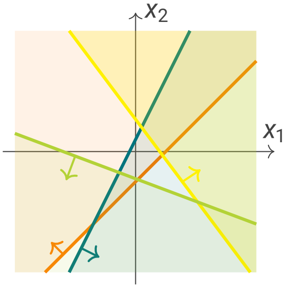


ReLU-Layer Injectivity
Given: matrix $\mathbf{W} \in \mathbb{R}^{m \times d}$, vector $\mathbf{b} \in \mathbb{R}^m$
Question: is the map $\phi_{\mathbf{W},\mathbf{b}}$ injective?

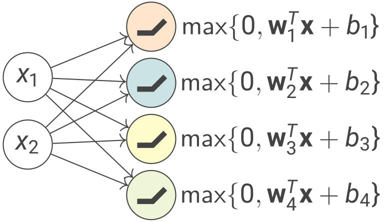
Theorem [Puthawala et al., 2022] $\phi_{\mathbf{W},\mathbf{b}}$ injective $\iff$ for every cell, active neurons (rows of $\mathbf{W}$) have rank d

Computational Complexity

- number of cells in $O(m^d)$ [Zaslavsky, 1975]



ReLU-Layer: Injectivity

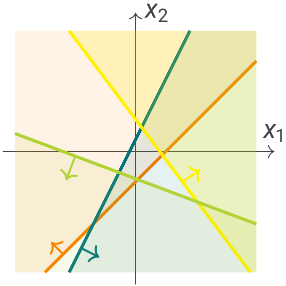


RELU-LAYER INJECTIVITY
Given: matrix $\mathbf{W} \in \mathbb{R}^{m\times d}$, vector $\mathbf{b} \in \mathbb{R}^m$
Question: is the map $\phi_{\mathbf{W},\mathbf{b}}$ injective?

Theorem [Puthawala et al., 2022] $\phi_{\mathbf{W},\mathbf{b}}$ injective $\iff$ for every cell, active neurons (rows of $\mathbf{W}$) have rank d

Computational Complexity

- number of cells in $O(m^d)$ [Zaslavsky, 1975]
- algorithm with runtime $O(\mathrm{poly}(m)m^d)$



Hardness-Result

Theorem [Froese, G., Skutella]

ReLU-Layer Injectivity is coNP-complete.

Hardness-Result

Theorem [Froese, G., Skutella]

ReLU-Layer Injectivity is coNP-complete.

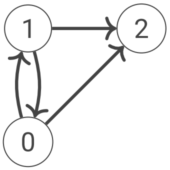
Proof

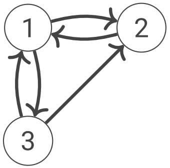
Reduction from complement of:

ACYCLIC-2-DISCONNECTION

Input: A digraph $D = (V, A)$.

Question: Is there a subset $A' \subseteq A$ of arcs such that (V, A') is acyclic and $(V, A \backslash A')$ is not weakly connected?





Hardness-Result

Theorem [Froese, G., Skutella]

ReLU-Layer Injectivity is coNP-complete.

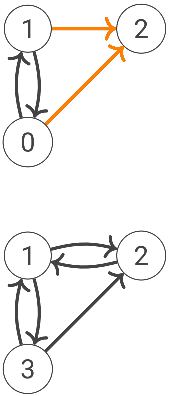
Proof

Reduction from complement of:

ACYCLIC-2-DISCONNECTION

Input: A digraph $D = (V, A)$.

Question: Is there a subset $A' \subseteq A$ of arcs such that (V, A') is acyclic and $(V, A \setminus A')$ is not weakly connected?



Hardness-Result

Theorem [Froese, G., Skutella]

ReLU-Layer Injectivity is coNP-complete.

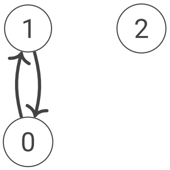
Proof

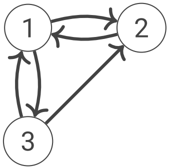
Reduction from complement of:

ACYCLIC-2-DISCONNECTION

Input: A digraph $D = (V, A)$.

Question: Is there a subset $A' \subseteq A$ of arcs such that (V, A') is acyclic and $(V, A \setminus A')$ is not weakly connected?





Hardness-Result

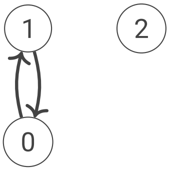
Theorem [Froese, G., Skutella]

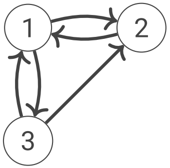
ReLU-Layer Injectivity is coNP-complete.

Proof

Reduction from complement of:

ACYCLIC-2-DISCONNECTION
Input: A digraph $D = (V, A)$.
Question: Is there a subset $A' \subseteq A$ of arcs such that (V, A') is acyclic and $(V, A \setminus A')$ is not weakly connected?





Theorem [Froese, G., Skutella]

ACYCLIC-2-DISCONNECTION is NP-complete.

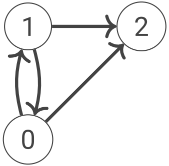
ReLU Layer from Digraph

ReLU Layer from Digraph

- Given a digraph $D = (V, A)$ with $V = \{0, \dots, d\}$ and $m = |A|$, we construct the following ReLU-layer $\phi \colon \mathbb{R}^d \to \mathbb{R}^m$ by

$$\phi(\mathbf{x}) = (\max\{0, x_i - x_j\})_{(i,j) \in A},$$

where $x_0 := 0$.

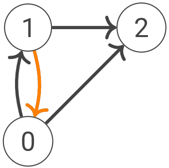


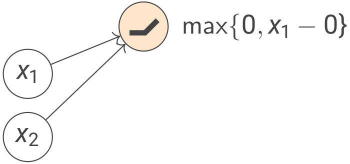
ReLU Layer from Digraph

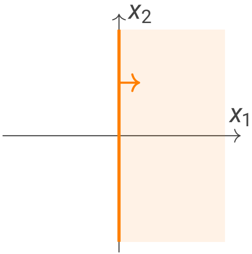
- Given a digraph $D = (V, A)$ with $V = \{0, \dots, d\}$ and $m = |A|$, we construct the following ReLU-layer $\phi\colon \mathbb{R}^d \to \mathbb{R}^m$ by

$$\phi(\mathbf{x}) = (\max\{0, x_i - x_j\})_{(i,j)\in A},$$

where $x_0 := 0$.





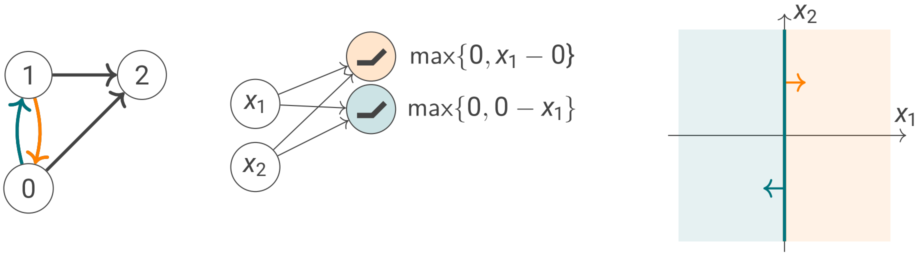


ReLU Layer from Digraph

- Given a digraph $D = (V, A)$ with $V = \{0, \dots, d\}$ and $m = |A|$, we construct the following ReLU-layer $\phi\colon \mathbb{R}^d \to \mathbb{R}^m$ by

$$\phi(\mathbf{x}) = (\max\{0, x_i - x_j\})_{(i,j)\in A},$$

where $x_0 := 0$.

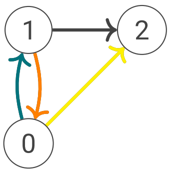


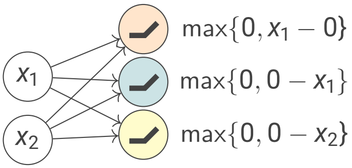
ReLU Layer from Digraph

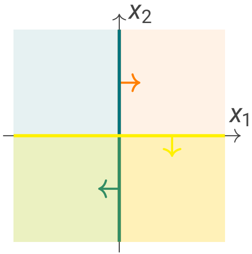
- Given a digraph $D = (V, A)$ with $V = \{0, \ldots, d\}$ and $m = |A|$, we construct the following ReLU-layer $\phi \colon \mathbb{R}^d \to \mathbb{R}^m$ by

$$\phi(\mathbf{x}) = (\max\{0, x_i - x_j\})_{(i,j)\in A},$$

where $x_0 := 0$.





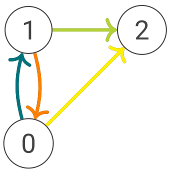


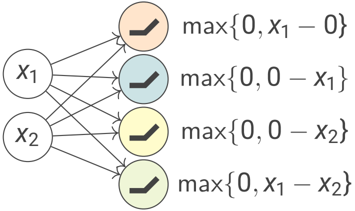
ReLU Layer from Digraph

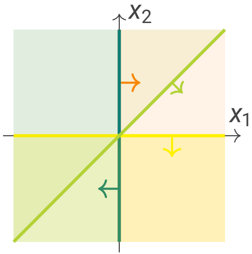
- Given a digraph $D = (V, A)$ with $V = \{0, \dots, d\}$ and $m = |A|$, we construct the following ReLU-layer $\phi\colon \mathbb{R}^d \to \mathbb{R}^m$ by

$$\phi(\mathbf{x}) = (\max\{0, x_i - x_j\})_{(i,j)\in A},$$

where $x_0 := 0$.





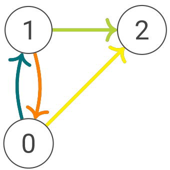


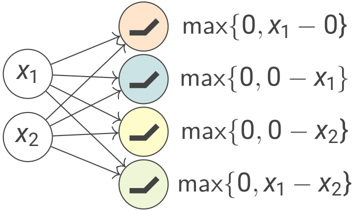
ReLU Layer from Digraph

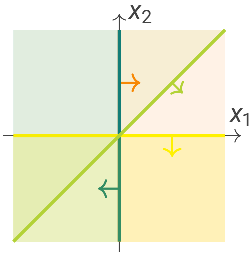
- Given a digraph $D = (V, A)$ with $V = \{0, \dots, d\}$ and $m = |A|$, we construct the following ReLU-layer $\phi \colon \mathbb{R}^d \to \mathbb{R}^m$ by

$$\phi(\mathbf{x}) = (\max\{0, x_i - x_j\})_{(i,j)\in A},$$

where $x_0 := 0$.



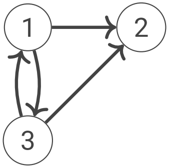


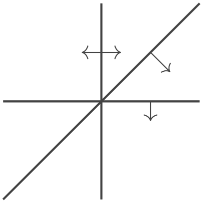


- All hyperplanes are of the form $\{x_i = x_j\}$ for $i, j \in [d]_0$

Full Rank $=$ Weakly Connected

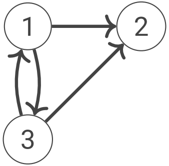
- For all permutation $\pi : [d]_0 \to [d]_0$ it holds that ϕ is linear on $C_\pi = \{x_{\pi^{-1}(0)} \leq \ldots \leq x_{\pi^{-1}(d)}\}$

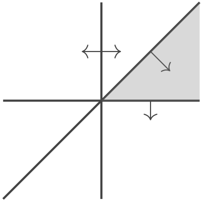




Full Rank $=$ Weakly Connected

- For all permutation $\pi : [d]_0 \to [d]_0$ it holds that ϕ is linear on $C_\pi = \{x_{\pi^{-1}(0)} \le \ldots \le x_{\pi^{-1}(d)}\}$
- ϕ is injective if and only if $\mathbf{W}_{C_\pi}$ has full rank for all π.

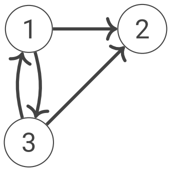


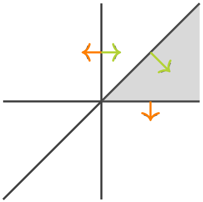


$0 \le x_2 \le x_1$

Full Rank $=$ Weakly Connected

- For all permutation $\pi : [d]_0 \to [d]_0$ it holds that ϕ is linear on $C_\pi = \{x_{\pi^{-1}(0)} \leq \ldots \leq x_{\pi^{-1}(d)}\}$
- ϕ is injective if and only if $\mathbf{W}_{C_\pi}$ has full rank for all π.
- A neuron $\max\{0, x_i - x_j\}$ corresponding to the arc (i, j) is active on C_π if and only if $x_i \geq x_j$ if and only if $\pi(i) > \pi(j)$

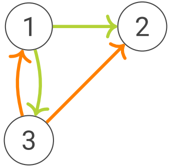


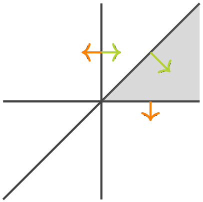


$0 \leq x_2 \leq x_1$

Full Rank = Weakly Connected

- For all permutation $\pi : [d]_0 \to [d]_0$ it holds that ϕ is linear on $C_\pi = \{x_{\pi^{-1}(0)} \leq \ldots \leq x_{\pi^{-1}(d)}\}$
- ϕ is injective if and only if $\mathbf{W}_{C_\pi}$ has full rank for all π.
- A neuron $\max\{0, x_i - x_j\}$ corresponding to the arc (i, j) is active on C_π if and only if $x_i \geq x_j$ if and only if $\pi(i) > \pi(j)$
- Let $A_\pi = \{(i, j) \in A \mid \pi(i) < \pi(j)\} \subseteq A$ be the (acyclic) set of arcs corresponding to the inactive neurons on C_π and $A \setminus A_\pi$ the set of arcs corresponding to the active neurons on C_π

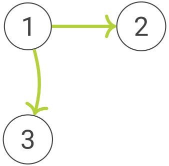


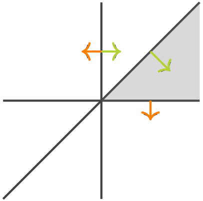


$0 \leq x_2 \leq x_1$

Full Rank $=$ Weakly Connected

- For all permutation $\pi : [d]_0 \to [d]_0$ it holds that ϕ is linear on $C_\pi = \{x_{\pi^{-1}(0)} \leq \ldots \leq x_{\pi^{-1}(d)}\}$
- ϕ is injective if and only if $\mathbf{W}_{C_\pi}$ has full rank for all π.
- A neuron $\max\{0, x_i - x_j\}$ corresponding to the arc (i, j) is active on C_π if and only if $x_i \geq x_j$ if and only if $\pi(i) > \pi(j)$
- Let $A_\pi = \{(i, j) \in A \mid \pi(i) < \pi(j)\} \subseteq A$ be the (acyclic) set of arcs corresponding to the inactive neurons on C_π and $A \setminus A_\pi$ the set of arcs corresponding to the active neurons on C_π

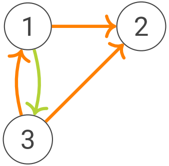




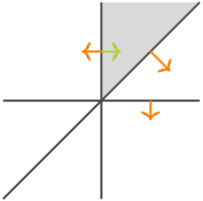
$0 \leq x_2 \leq x_1$

Full Rank $=$ Weakly Connected

- For all permutation $\pi : [d]_0 \to [d]_0$ it holds that ϕ is linear on $C_\pi = \{x_{\pi^{-1}(0)} \leq \ldots \leq x_{\pi^{-1}(d)}\}$
- ϕ is injective if and only if $\mathbf{W}_{C_\pi}$ has full rank for all π.
- A neuron $\max\{0, x_i - x_j\}$ corresponding to the arc (i, j) is active on C_π if and only if $x_i \geq x_j$ if and only if $\pi(i) > \pi(j)$
- Let $A_\pi = \{(i, j) \in A \mid \pi(i) < \pi(j)\} \subseteq A$ be the (acyclic) set of arcs corresponding to the inactive neurons on C_π and $A \setminus A_\pi$ the set of arcs corresponding to the active neurons on C_π

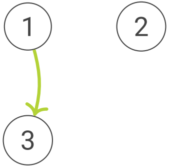


$0 \leq x_1 \leq x_2$

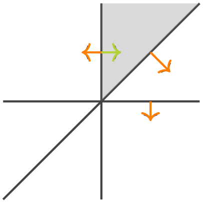


Full Rank = Weakly Connected

- For all permutation $\pi : [d]_0 \to [d]_0$ it holds that ϕ is linear on $C_\pi = \{x_{\pi^{-1}(0)} \leq \ldots \leq x_{\pi^{-1}(d)}\}$
- ϕ is injective if and only if $\mathbf{W}_{C_\pi}$ has full rank for all π.
- A neuron $\max\{0, x_i - x_j\}$ corresponding to the arc (i, j) is active on C_π if and only if $x_i \geq x_j$ if and only if $\pi(i) > \pi(j)$
- Let $A_\pi = \{(i, j) \in A \mid \pi(i) < \pi(j)\} \subseteq A$ be the (acyclic) set of arcs corresponding to the inactive neurons on C_π and $A \setminus A_\pi$ the set of arcs corresponding to the active neurons on C_π

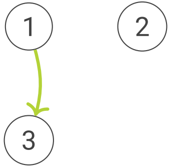


$0 \leq x_1 \leq x_2$

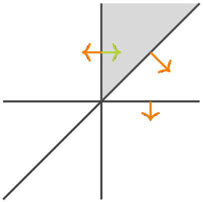


Full Rank = Weakly Connected

- For all permutation $\pi : [d]_0 \to [d]_0$ it holds that ϕ is linear on $C_\pi = \{x_{\pi^{-1}(0)} \leq \ldots \leq x_{\pi^{-1}(d)}\}$
- ϕ is injective if and only if $\mathbf{W}_{C_\pi}$ has full rank for all π.
- A neuron $\max\{0, x_i - x_j\}$ corresponding to the arc (i,j) is active on C_π if and only if $x_i \geq x_j$ if and only if $\pi(i) > \pi(j)$
- Let $A_\pi = \{(i,j) \in A \mid \pi(i) < \pi(j)\} \subseteq A$ be the (acyclic) set of arcs corresponding to the inactive neurons on C_π and $A \setminus A_\pi$ the set of arcs corresponding to the active neurons on C_π
- $\mathbf{W}_{C_\pi}$ has full rank $\iff D_\pi = (V, A \setminus A_\pi)$ is weakly connected.

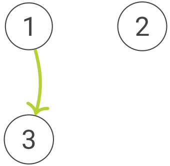


$0 \leq x_1 \leq x_2$

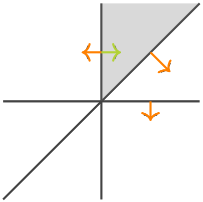


Full Rank = Weakly Connected

- For all permutation $\pi : [d]_0 \to [d]_0$ it holds that ϕ is linear on $C_\pi = \{x_{\pi^{-1}(0)} \leq \ldots \leq x_{\pi^{-1}(d)}\}$
- ϕ is injective if and only if $\mathbf{W}_{C_\pi}$ has full rank for all π.
- A neuron $\max\{0, x_i - x_j\}$ corresponding to the arc (i,j) is active on C_π if and only if $x_i \geq x_j$ if and only if $\pi(i) > \pi(j)$
- Let $A_\pi = \{(i,j) \in A \mid \pi(i) < \pi(j)\} \subseteq A$ be the (acyclic) set of arcs corresponding to the inactive neurons on C_π and $A \setminus A_\pi$ the set of arcs corresponding to the active neurons on C_π
- $\mathbf{W}_{C_\pi}$ has full rank $\iff$ $D_\pi = (V, A \setminus A_\pi)$ is weakly connected.
- ϕ is not injective $\iff$ there is an acyclic-2-disconnection.



$0 \leq x_1 \leq x_2$



FPT-Algorithm

Can we do better than $O(m^d)$?

FPT-Algorithm

Can we do better than $O(m^d)$?

Theorem [Froese, G., Skutella. 2024]

ReLU-Layer Injectivity can be solved in $O\big(\mathrm{poly}(m)(d+1)^d\big)$ time (i.e., FPT for dimension d).

Verification

Theorem [Froese, G., Skutella. 2024]

For a one hidden layer neural network, it is NP-complete to decide if the function attains a positive output value

Verification

Theorem [Froese, G., Skutella. 2024]

For a one hidden layer neural network, it is NP-complete to decide if the function attains a positive output value

Proof: Similar reduction from cut problem in weighted graph.

Verification

Theorem [Froese, G., Skutella. 2024]

For a one hidden layer neural network, it is NP-complete to decide if the function attains a positive output value

Proof: Similar reduction from cut problem in weighted graph.

Corollaries [Froese, G., Skutella. 2024]

For a one hidden layer neural network

- it is NP-complete to decide if the function is surjective.
- there is no polynomial time algorithm that approximates the maximum function value on the unit ball unless P=NP.

Verification

Theorem [Froese, G., Skutella. 2024]

For a one hidden layer neural network, it is NP-complete to decide if the function attains a positive output value

Proof: Similar reduction from cut problem in weighted graph.

Corollaries [Froese, G., Skutella. 2024]

For a one hidden layer neural network

- it is NP-complete to decide if the function is surjective.
- there is no polynomial time algorithm that approximates the maximum function value on the unit ball unless P=NP.

Open Question: Do the hardness results also hold for trained neural networks?

Verification

Theorem [Froese, G., Skutella. 2024]

For a one hidden layer neural network, it is NP-complete to decide if the function attains a positive output value

Proof: Similar reduction from cut problem in weighted graph.

Corollaries [Froese, G., Skutella. 2024]

For a one hidden layer neural network

- it is NP-complete to decide if the function is surjective.
- there is no polynomial time algorithm that approximates the maximum function value on the unit ball unless P=NP.

Open Question: Do the hardness results also hold for trained neural networks?

Thank You!