Combinatorial and Implicit Approaches to Deep Learning

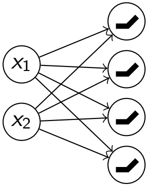
Moritz Grillo
Yulia Alexandr, Vincent Froese, Christoph Hertrich, Georg Loho
Guido Montúfar, Martin Skutella and Moritz Stargalla

SPP Annual Meeting Theoretical Foundations of Deep Learning

November 5, 2025

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 $\mathbf{b} \in \mathbb{R}^m$

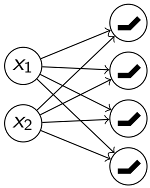


ReLU-Layer and its Geometry

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

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

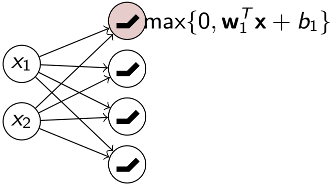


ReLU-Layer and its Geometry

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

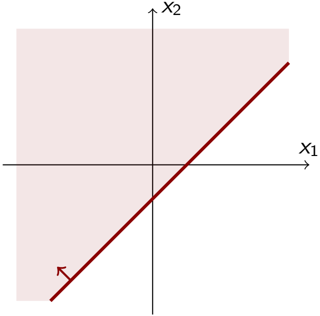
Computes map

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



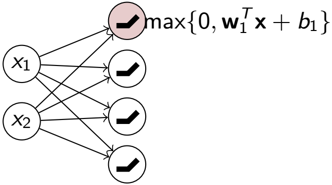
Geometry

- 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 $\mathbf{b} \in \mathbb{R}^m$



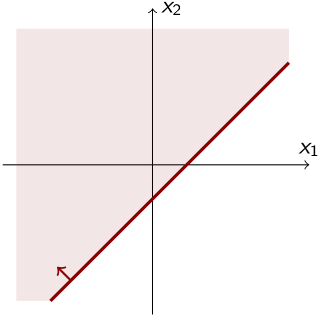
Computes map

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

Geometry

- 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

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

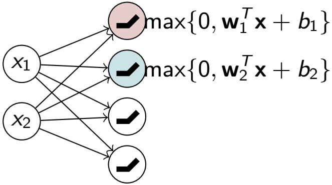


ReLU-Layer and its Geometry

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

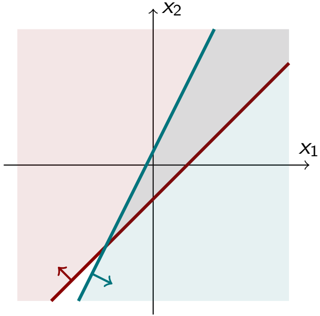
$$f_{\mathbf{W},\mathbf{b}} \colon \mathbb{R}^d \to \mathbb{R}^m, \quad \mathbf{x} \mapsto \max\{\mathbf{0}, \mathbf{W}\mathbf{x} + \mathbf{b}\}$$



Geometry

- 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

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

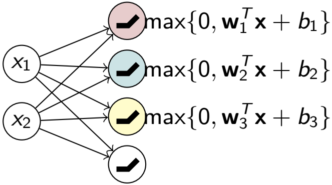


ReLU-Layer and its Geometry

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

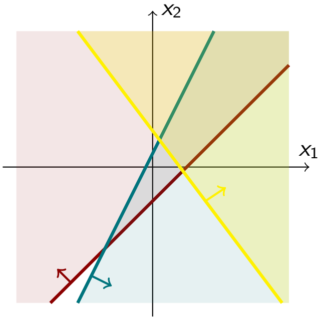
$$f_{\mathbf{W},\mathbf{b}} : \mathbb{R}^d \to \mathbb{R}^m, \quad \mathbf{x} \mapsto \max\{\mathbf{0}, \mathbf{W}\mathbf{x} + \mathbf{b}\}$$



Geometry

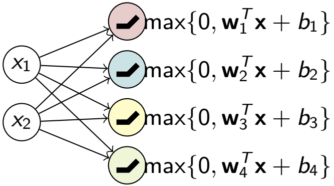
- output neuron i active at $\mathbf{x} \in \mathbb{R}^d$ if $\mathbf{w}_i^T \mathbf{x} + b_i \geq 0$
- output neurons induce (oriented) hyperplanes

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



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 $\mathbf{b} \in \mathbb{R}^{m}$



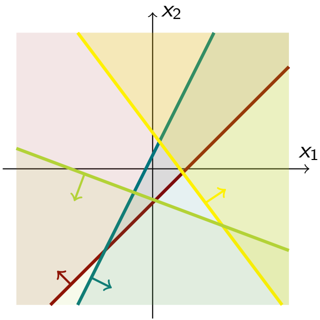
Computes map

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

Geometry

- 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

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

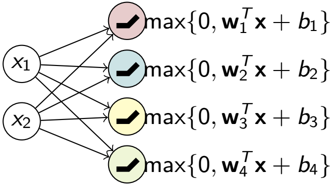


ReLU-Layer and its Geometry

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

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

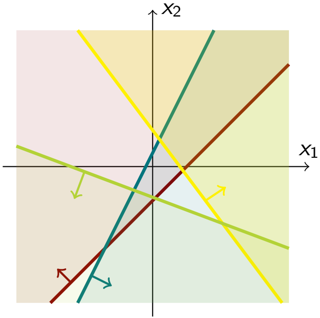


Geometry

- 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

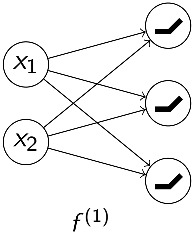


ReLU Neural Networks

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

ReLU Neural Networks

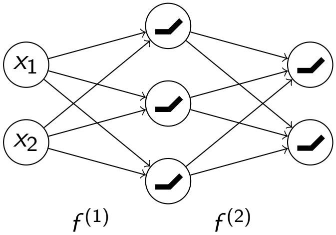
- A ReLU network is a concatenation of such ReLU layers:



- 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

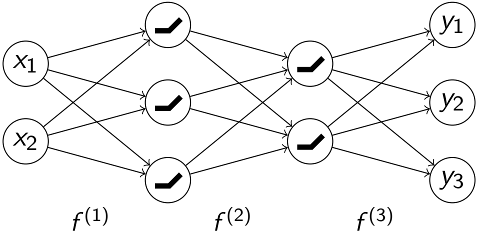
- A ReLU network is a concatenation of such ReLU layers:



- 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

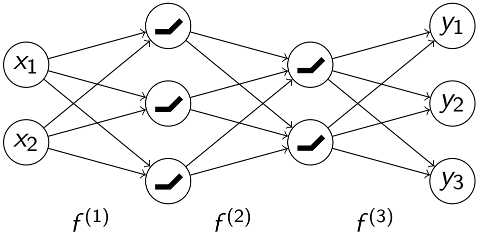
- A ReLU network is a concatenation of such ReLU layers:



- 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

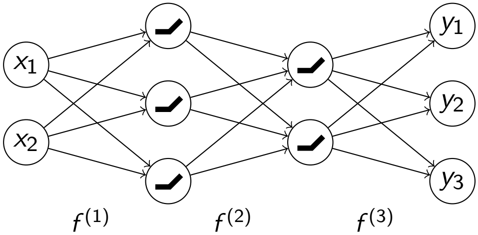
- A ReLU network is a concatenation of such ReLU layers:



- 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.
- Still subdivides input space into activation regions where f is affine linear

ReLU Neural Networks

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



Architecture

- $\mathcal{A} = (2, 3, 2, 3)$
- 2 hidden 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.
- Still subdivides input space into activation regions where f is affine linear

Overview:

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

Overview:

- A neural network is a parameterized function $f_\theta \colon \mathbb{R}^d \to \mathbb{R}^m$ with $\theta = (\mathbf{W}^{(1)}, \mathbf{b}^{(1)}, \dots, \mathbf{W}^{(\ell)}, \mathbf{b}^{(\ell)})$
- For a given architecture $\mathcal{A}$ we have $\mu_\mathcal{A} \colon \Theta_\mathcal{A} \to \mathcal{F}_\mathcal{A}$ given by $\theta \mapsto f_\theta$

Overview:

- A neural network is a parameterized function $f_\theta \colon \mathbb{R}^d \to \mathbb{R}^m$ with $\theta = (\mathbf{W}^{(1)}, \mathbf{b}^{(1)}, \ldots, \mathbf{W}^{(\ell)}, \mathbf{b}^{(\ell)})$
- For a given architecture $\mathcal{A}$ we have $\mu_{\mathcal{A}} \colon \Theta_{\mathcal{A}} \to \mathcal{F}_{\mathcal{A}}$ given by $\theta \mapsto f_\theta$

Questions:

1. Given θ, what can we say about the properties of f_θ?

Overview:

- A neural network is a parameterized function $f_\theta \colon \mathbb{R}^d \to \mathbb{R}^m$ with $\theta = (\mathbf{W}^{(1)}, \mathbf{b}^{(1)}, \ldots, \mathbf{W}^{(\ell)}, \mathbf{b}^{(\ell)})$
- For a given architecture $\mathcal{A}$ we have $\mu_{\mathcal{A}} \colon \Theta_{\mathcal{A}} \to \mathcal{F}_{\mathcal{A}}$ given by $\theta \mapsto f_\theta$

Questions:

1. Given θ, what can we say about the properties of f_θ?
2. What is $\mathcal{F}_{\mathcal{A}}$?

Overview:

- A neural network is a parameterized function $f_\theta \colon \mathbb{R}^d \to \mathbb{R}^m$ with $\theta = (\mathbf{W}^{(1)}, \mathbf{b}^{(1)}, \dots, \mathbf{W}^{(\ell)}, \mathbf{b}^{(\ell)})$
- For a given architecture $\mathcal{A}$ we have $\mu_\mathcal{A} \colon \Theta_\mathcal{A} \to \mathcal{F}_\mathcal{A}$ given by $\theta \mapsto f_\theta$

Questions:

1. Given θ, what can we say about the properties of f_θ?
2. What is $\mathcal{F}_\mathcal{A}$?
3. Given data set $X \subseteq \mathbb{R}^d$, what can we say about $\theta \mapsto f_\theta(X)$?

Overview:

- A neural network is a parameterized function $f_\theta\colon \mathbb{R}^d \to \mathbb{R}^m$ with $\theta = (\mathbf{W}^{(1)}, \mathbf{b}^{(1)}, \dots, \mathbf{W}^{(\ell)}, \mathbf{b}^{(\ell)})$
- For a given architecture $\mathcal{A}$ we have $\mu_\mathcal{A}\colon \Theta_\mathcal{A} \to \mathcal{F}_\mathcal{A}$ given by $\theta \mapsto f_\theta$

Questions:

1. Given θ, what can we say about the properties of f_θ?
2. What is $\mathcal{F}_\mathcal{A}$?
3. Given data set $X \subseteq \mathbb{R}^d$, what can we say about $\theta \mapsto f_\theta(X)$?
4. Given f, what can we say about the fiber $\mu_\mathcal{A}^{-1}(f) \subseteq \Theta_\mathcal{A}$?

Given θ, what can we say about the properties of f_θ?

Given θ, what can we say about the properties of f_θ?

Questions:

- What is the the maximum function value $\max_{x \in B} f_\theta(x)$ on a set $B \subseteq \mathbb{R}^d$?

Given θ, what can we say about the properties of f_θ?

Questions:

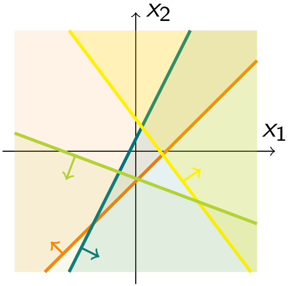
- What is the the maximum function value $\max_{x \in B} f_\theta(x)$ on a set $B \subseteq \mathbb{R}^d$?
- Does f_θ attain a positive output value?

Given θ, what can we say about the properties of f_θ?

Questions:

- What is the the maximum function value $\max_{x \in B} f_\theta(x)$ on a set $B \subseteq \mathbb{R}^d$?
- Does f_θ attain a positive output value?
- What is the Lipschitz constant of f_θ?

Given θ, what can we say about the properties of f_θ?

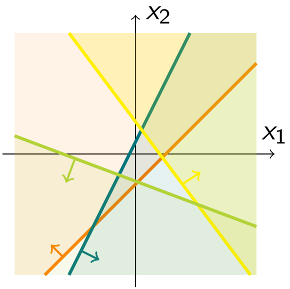


Questions:

- What is the the maximum function value $\max_{x \in B} f_\theta(x)$ on a set $B \subseteq \mathbb{R}^d$?
- Does f_θ attain a positive output value?
- What is the Lipschitz constant of f_θ?

How to solve them naively?

Given θ, what can we say about the properties of f_θ?



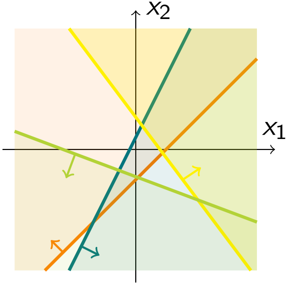
Questions:

- What is the the maximum function value $\max_{x \in B} f_\theta(x)$ on a set $B \subseteq \mathbb{R}^d$?
- Does f_θ attain a positive output value?
- What is the Lipschitz constant of f_θ?

How to solve them naively?

- Solve problems on every linear region with e.g. linear programming

Given θ, what can we say about the properties of f_θ?



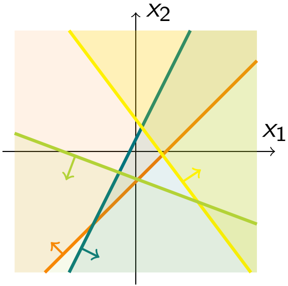
Questions:

- What is the the maximum function value $\max_{x \in B} f_\theta(x)$ on a set $B \subseteq \mathbb{R}^d$?
- Does f_θ attain a positive output value?
- What is the Lipschitz constant of f_θ?

How to solve them naively?

- Solve problems on every linear region with e.g. linear programming
- **But**: For network with ℓ hidden layers and width n, there can be $O(n^{\ell d})$ many linear regions $\rightarrow$ Already intractable for one hidden layer in high dimension....

Given θ, what can we say about the properties of f_θ?



Questions:

- What is the the maximum function value $\max_{x \in B} f_\theta(x)$ on a set $B \subseteq \mathbb{R}^d$?
- Does f_θ attain a positive output value?
- What is the Lipschitz constant of f_θ?

How to solve them naively?

- Solve problems on every linear region with e.g. linear programming
- **But**: For network with ℓ hidden layers and width n, there can be $O(n^{\ell d})$ many linear regions $\rightarrow$ Already intractable for one hidden layer in high dimension....

Can we do something better?

Asymptotically (most likely) not :(

Theorem[Froese, G., Hertrich, Skutella, Stargalla, 2025] The following problems are NP-hard and **not** solvable in $n^{o(d)}$ time, assuming ETH:

For one hidden layer:

- Deciding if f_θ attains a positive value
- Computing $\max_{x \in B} f_\theta(x)$ for any open set B
- Computing the Lipschitz constant
- Deciding if f_θ is surjective or injective

Asymptotically (most likely) not :(

Theorem[Froese, G., Hertrich, Skutella, Stargalla, 2025] The following problems are NP-hard and **not** solvable in $n^{o(d)}$ time, assuming ETH:

For one hidden layer:

- Deciding if f_θ attains a positive value
- Computing $\max_{x \in B} f_\theta(x)$ for any open set B
- Computing the Lipschitz constant
- Deciding if f_θ is surjective or injective

For two hidden layers:

- Deciding if f_θ is the zero map.
- Approximating $\max_{x \in B} f_\theta(x)$
- Approximating the Lipschitz constant

Asymptotically (most likely) not :(

Theorem[Froese, G., Hertrich, Skutella, Stargalla, 2025] The following problems are NP-hard and **not** solvable in $n^{o(d)}$ time, assuming ETH:

For one hidden layer:

- Deciding if f_θ attains a positive value
- Computing $\max_{x \in B} f_\theta(x)$ for any open set B
- Computing the Lipschitz constant
- Deciding if f_θ is surjective or injective

For two hidden layers:

- Deciding if f_θ is the zero map.
- Approximating $\max_{x \in B} f_\theta(x)$
- Approximating the Lipschitz constant

Open Question:

What about average runtime?

Which Functions are computable with ReLU Networks?

- 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 [Arora et al, 2018]

Which Functions are computable with ReLU Networks?

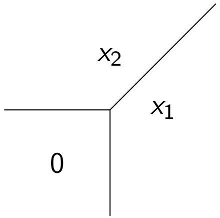
- Every CPWL function $f\colon \mathbb{R}^d \to \mathbb{R}$ can be represented by a ReLU NN with $\lceil \log_3(d) \rceil$ hidden layers [Arora et al, 2018], [Bakaev et al, 2025]

Which Functions are computable with ReLU Networks?

- Every CPWL function $f\colon \mathbb{R}^d \to \mathbb{R}$ can be represented by a ReLU NN with $\lceil \log_3(d) \rceil$ hidden layers [Arora et al, 2018], [Bakaev et al, 2025]
- How many layers are necessary?

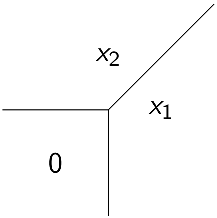
Which Functions are computable with ReLU Networks?

- Every CPWL function $f\colon \mathbb{R}^d \to \mathbb{R}$ can be represented by a ReLU NN with $\lceil \log_3(d) \rceil$ hidden layers [Arora et al, 2018], [Bakaev et al, 2025]
- How many layers are necessary?
- There is no function that needs more layers than $\max\{0, x_1, \ldots, x_d\}$ [Hertrich et al, 2021].



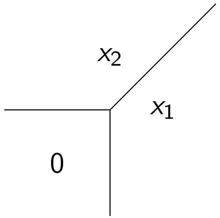
Which Functions are computable with ReLU Networks?

- Every CPWL function $f\colon \mathbb{R}^d \to \mathbb{R}$ can be represented by a ReLU NN with $\lceil \log_3(d) \rceil$ hidden layers [Arora et al, 2018], [Bakaev et al, 2025]
- How many layers are necessary?
- There is no function that needs more layers than $\max\{0, x_1, \ldots, x_d\}$ [Hertrich et al, 2021].
- $\max\{0, x_1, x_2\}$ cannot be represented with one hidden layer [Basu, Mukherjee, 17].



Which Functions are computable with ReLU Networks?

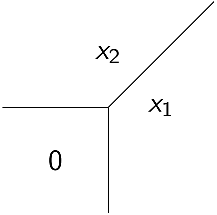
- Every CPWL function $f\colon \mathbb{R}^d \to \mathbb{R}$ can be represented by a ReLU NN with $\lceil \log_3(d) \rceil$ hidden layers [Arora et al, 2018], [Bakaev et al, 2025]
- How many layers are necessary?
- There is no function that needs more layers than $\max\{0, x_1, \ldots, x_d\}$ [Hertrich et al, 2021].
- $\max\{0, x_1, x_2\}$ cannot be represented with one hidden layer [Basu, Mukherjee, 17].



- Non-constant lower bounds when restricting weights or breakpoints. [Haase, Hertrich, Loho, 23],[Averkov, Hojny Merkert, 25],[G., Hertrich, Loho, 25]

Which Functions are computable with ReLU Networks?

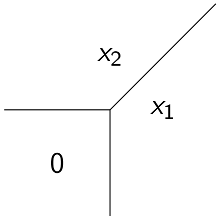
- Every CPWL function $f\colon \mathbb{R}^d \to \mathbb{R}$ can be represented by a ReLU NN with $\lceil \log_3(d) \rceil$ hidden layers [Arora et al, 2018], [Bakaev et al, 2025]
- How many layers are necessary?
- There is no function that needs more layers than $\max\{0, x_1, \dots, x_d\}$ [Hertrich et al, 2021].
- $\max\{0, x_1, x_2\}$ cannot be represented with one hidden layer [Basu, Mukherjee, 17].



- Non-constant lower bounds when restricting weights or breakpoints. [Haase, Hertrich, Loho, 23],[Averkov, Hojny Merkert, 25],[G., Hertrich, Loho, 25]
- In general: 2 is best known lower bound!

Which Functions are computable with ReLU Networks?

- Every CPWL function $f\colon \mathbb{R}^d \to \mathbb{R}$ can be represented by a ReLU NN with $\lceil \log_3(d) \rceil$ hidden layers [Arora et al, 2018], [Bakaev et al, 2025]
- How many layers are necessary?
- There is no function that needs more layers than $\max\{0, x_1, \dots, x_d\}$ [Hertrich et al, 2021].
- $\max\{0, x_1, x_2\}$ cannot be represented with one hidden layer [Basu, Mukherjee, 17].



- Non-constant lower bounds when restricting weights or breakpoints. [Haase, Hertrich, Loho, 23],[Averkov, Hojny Merkert, 25],[G., Hertrich, Loho, 25]
- In general: 2 is best known lower bound!

Open Question:

Is there a CPWL function that needs more than two hidden layers??

Smallest open case: $\max\{0, x_1, \dots, x_5\}$

Given data set $X \subseteq \mathbb{R}^d$, what can we say about $\theta \mapsto f_\theta(X)$?

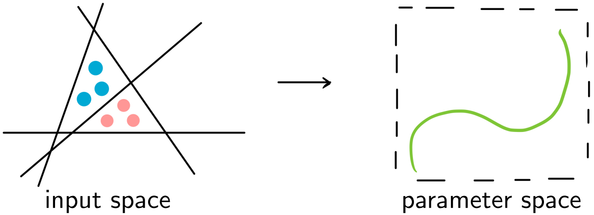
- $x \mapsto f_\theta(x)$ is piecewise linear for fixed θ, what about $\theta \mapsto f_\theta(x)$ for fixed x?

Given data set $X \subseteq \mathbb{R}^d$, what can we say about $\theta \mapsto f_\theta(X)$?

- $x \mapsto f_\theta(x)$ is piecewise linear for fixed θ, what about $\theta \mapsto f_\theta(x)$ for fixed x?
- For data set $X = [x_1, \ldots, x_m]$, fix an activation pattern $A = [a_1, \ldots, a_m]$ where $a_i \in \{+, -\}^{\#\text{neurons}}$ determines which neurons are active at x_i.

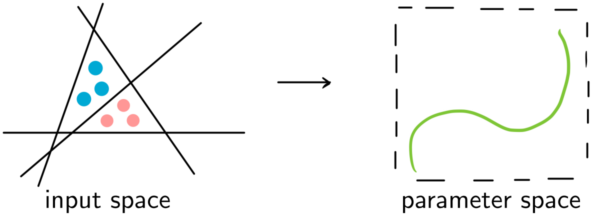
Given data set $X \subseteq \mathbb{R}^d$, what can we say about $\theta \mapsto f_\theta(X)$?

- $x \mapsto f_\theta(x)$ is piecewise linear for fixed θ, what about $\theta \mapsto f_\theta(x)$ for fixed x?
- For data set $X = [x_1, \ldots, x_m]$, fix an activation pattern $A = [a_1, \ldots, a_m]$ where $a_i \in \{+, -\}^{\#\text{neurons}}$ determines which neurons are active at x_i.
- $\theta \mapsto f_\theta(X)$ is multi-linear for a fixed activation pattern $\to$ piecewise multi-linear.



Given data set $X \subseteq \mathbb{R}^d$, what can we say about $\theta \mapsto f_\theta(X)$?

- $x \mapsto f_\theta(x)$ is piecewise linear for fixed θ, what about $\theta \mapsto f_\theta(x)$ for fixed x?
- For data set $X = [x_1, \ldots, x_m]$, fix an activation pattern $A = [a_1, \ldots, a_m]$ where $a_i \in \{+, -\}^{\#\text{neurons}}$ determines which neurons are active at x_i.
- $\theta \mapsto f_\theta(X)$ is multi-linear for a fixed activation pattern $\rightarrow$ piecewise multi-linear.



Problem:

Identify equations that hold on image of $\theta \mapsto f_\theta(X)$

Pattern Variety

- Assuming "general" X, the parameterization for a fixed activation pattern A is equivalent to
$$\varphi_A(\theta) = [M_1(\theta), \ldots, M_k(\theta)],$$
where $M_i(\theta)$ is matrix given by the linear map computed on activation region a_i.

Pattern Variety

- Assuming "general" X, the parameterization for a fixed activation pattern A is equivalent to
$$\varphi_A(\theta) = [M_1(\theta), \ldots, M_k(\theta)],$$
where $M_i(\theta)$ is matrix given by the linear map computed on activation region a_i.
- $V_A = \overline{\text{im}(\varphi_A)}$ is pattern variety $\approx$ (closure of) functions representable over a general data set with fixed activation pattern A.

Pattern Variety

- Assuming "general" X, the parameterization for a fixed activation pattern A is equivalent to
$$\varphi_A(\theta) = [M_1(\theta), \dots, M_k(\theta)],$$
where $M_i(\theta)$ is matrix given by the linear map computed on activation region a_i.
- $V_A = \overline{\mathrm{im}(\varphi_A)}$ is pattern variety $\approx$ (closure of) functions representable over a general data set with fixed activation pattern A.

What are invariants of V_A? Generating ideal $J^{\mathbf{A}}$? Dimension?

Pattern Variety

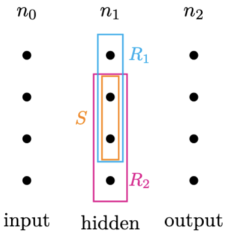
- Assuming "general" X, the parameterization for a fixed activation pattern A is equivalent to
$$\varphi_A(\theta) = [M_1(\theta), \dots, M_k(\theta)],$$
where $M_i(\theta)$ is matrix given by the linear map computed on activation region a_i.
- $V_A = \overline{\mathrm{im}(\varphi_A)}$ is pattern variety $\approx$ (closure of) functions representable over a general data set with fixed activation pattern A.

What are invariants of V_A? Generating ideal $J^{\mathbf{A}}$? Dimension?

Theorem[Alexandr and Montúfar, 2025]
For shallow networks:

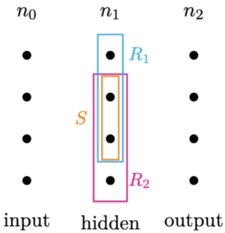
- Bounds on dimension and exact formula for bottleneck architecture
- Set of generators contained in ideal $J^{\mathbf{A}}$.

Two Activation Regions



Let $|R_1| = r_1$, $|R_2| = r_2$, $|S| = s$.
Let $t = r_1 + r_2 - 2s$.

Two Activation Regions



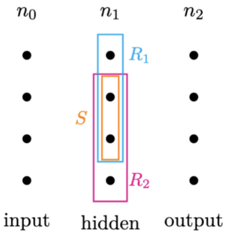
Let $|R_1| = r_1$, $|R_2| = r_2$, $|S| = s$.
Let $t = r_1 + r_2 - 2s$.

Theorem[Alexandr and Montúfar, 2025]

The ideal $J^{\mathbf{A}}$ contains:

1. $(r_1 + 1)$-minors of M_1;
2. $(r_2 + 1)$-minors of M_2;
3. $(n_1 + 1)$-minors of $[M_1 \mid M_2]$ and $[M_1^T \mid M_2^T]$;
4. $(t + 1)$-minors of $M_1 - M_2$.

Two Activation Regions



Let $|R_1| = r_1$, $|R_2| = r_2$, $|S| = s$.
Let $t = r_1 + r_2 - 2s$.

Theorem[Alexandr and Montúfar, 2025]
The ideal $J^{\mathbf{A}}$ contains:

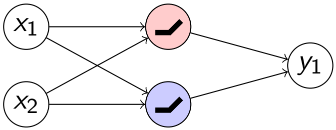
1. $(r_1 + 1)$-minors of M_1;
2. $(r_2 + 1)$-minors of M_2;
3. $(n_1 + 1)$-minors of $[M_1 \mid M_2]$ and $[M_1^T \mid M_2^T]$;
4. $(t + 1)$-minors of $M_1 - M_2$.

Conjecture: no other polynomials are needed to generate the ideal.

Given f_θ, what can we say about the fiber $\mu_{\mathcal{A}}^{-1}(f_\theta) \subseteq \Theta_{\mathcal{A}}$?

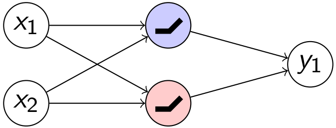
Given f_θ, what can we say about the fiber $\mu_{\mathcal{A}}^{-1}(f_\theta) \subseteq \Theta_{\mathcal{A}}$?

- Global Symmetries:
  - Permutation of neurons (P)



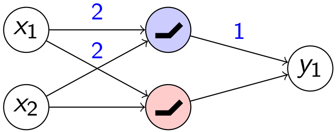
Given f_θ, what can we say about the fiber $\mu_{\mathcal{A}}^{-1}(f_\theta) \subseteq \Theta_{\mathcal{A}}$?

- Global Symmetries:
  - Permutation of neurons (P)



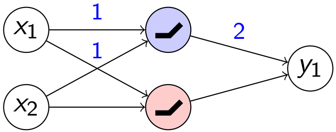
Given f_θ, what can we say about the fiber $\mu_{\mathcal{A}}^{-1}(f_\theta) \subseteq \Theta_{\mathcal{A}}$?

- Global Symmetries:
  - Permutation of neurons (P)
  - Scaling incoming weights with $\lambda > 0$ and outgoing weights with $\frac{1}{\lambda}$ (S).



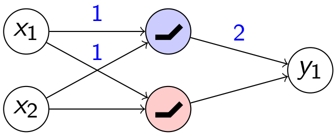
Given f_θ, what can we say about the fiber $\mu_{\mathcal{A}}^{-1}(f_\theta) \subseteq \Theta_{\mathcal{A}}$?

- Global Symmetries:
  - Permutation of neurons (P)
  - Scaling incoming weights with $\lambda > 0$ and outgoing weights with $\frac{1}{\lambda}$ (S).



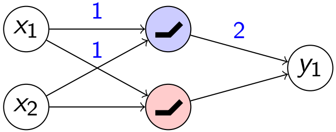
Given f_θ, what can we say about the fiber $\mu_{\mathcal{A}}^{-1}(f_\theta) \subseteq \Theta_{\mathcal{A}}$?

- Global Symmetries:
  - Permutation of neurons (P)
  - Scaling incoming weights with $\lambda > 0$ and outgoing weights with $\frac{1}{\lambda}$ (S).
- There are f_θ that uniquely determine θ up to (P) and (S). [Bui Thi Mai and Lampert, 20], [Grigsby, Lindsey and Rolnick, 23]



Given f_θ, what can we say about the fiber $\mu_{\mathcal{A}}^{-1}(f_\theta) \subseteq \Theta_{\mathcal{A}}$?

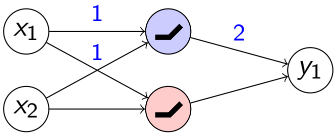
- Global Symmetries:
  - Permutation of neurons (P)
  - Scaling incoming weights with $\lambda > 0$ and outgoing weights with $\frac{1}{\lambda}$ (S).
- There are f_θ that uniquely determine θ up to (P) and (S). [Bui Thi Mai and Lampert, 20], [Grigsby, Lindsey and Rolnick, 23]



Question: Can we decide when there is a nontrivial fiber? And compute it?

Given f_θ, what can we say about the fiber $\mu_{\mathcal{A}}^{-1}(f_\theta) \subseteq \Theta_{\mathcal{A}}$?

- Global Symmetries:
  - Permutation of neurons (P)
  - Scaling incoming weights with $\lambda > 0$ and outgoing weights with $\frac{1}{\lambda}$ (S).
- There are f_θ that uniquely determine θ up to (P) and (S). [Bui Thi Mai and Lampert, 20], [Grigsby, Lindsey and Rolnick, 23]

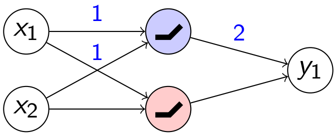


Question: Can we decide when there is a nontrivial fiber? And compute it?

In progress:
For two hidden layers $\mathbb{R}^d \to \mathbb{R}^{n_1} \to \mathbb{R}^{n_2} \to \mathbb{R}$ with $n_1 \leq d$ and generic θ, we can detect non-trivial fibers. We can probably also compute them.

Given f_θ, what can we say about the fiber $\mu_{\mathcal{A}}^{-1}(f_\theta) \subseteq \Theta_{\mathcal{A}}$?

- Global Symmetries:
  - Permutation of neurons (P)
  - Scaling incoming weights with $\lambda > 0$ and outgoing weights with $\frac{1}{\lambda}$ (S).
- There are f_θ that uniquely determine θ up to (P) and (S). [Bui Thi Mai and Lampert, 20], [Grigsby, Lindsey and Rolnick, 23]



Question: Can we decide when there is a nontrivial fiber? And compute it?

In progress:
For two hidden layers $\mathbb{R}^d \to \mathbb{R}^{n_1} \to \mathbb{R}^{n_2} \to \mathbb{R}$ with $n_1 \leq d$ and generic θ, we can detect non-trivial fibers. We can probably also compute them.

Open Questions:
What about $n_1 \geq d$? Or more hidden layers?

Thank you! Questions?