Towards Neural Homology Theory

MACHINE LEARNING
DEPARTMENT

William H. Guss [1,2] Ruslan Salakhutdinov [1]
{wguss, rsalakhu}@cs.cmu.edu

[1]Carnegie Mellon University

[2]Machine Learning at Berkeley

February 1, 2018

The problem of architecture selection

Deep learning has been extremely successful, in part due to the elimination of *feature engineering*.

Deep learning has been extremely successful, in part due to the elimination of *feature engineering*.

Research focus has shifted: best features → best architectures.

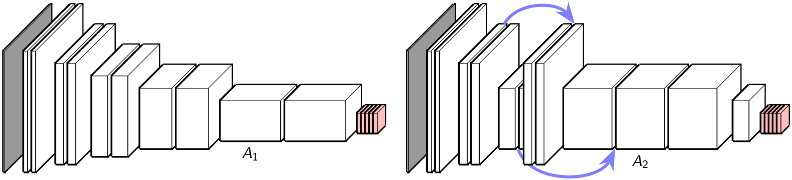


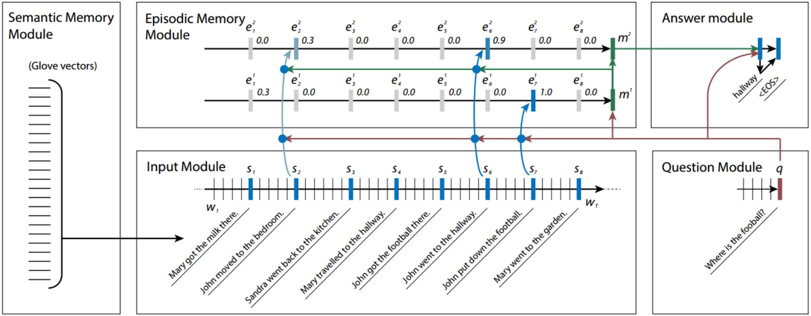
Figure: Examples of two competing convolutional architecture A_1 and A_2.

In computer vision, for example, a large body of work focuses on improving the initial architectural choices of AlexNet.

The problem of architecture selection

Deep learning has been extremely successful, in part due to the elimination of *feature engineering*.

Research focus has shifted: best features → best architectures.



Architectures are becoming **hyperspecialized**.

Despite the success of this approach, **there are still not general principles for choosing architectures in arbitrary settings**.

To scale deep learning to new problems without *expert architecture designers*, the problem of architecture selection must be understood.

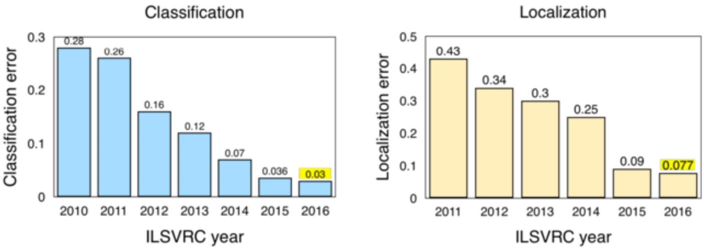


Figure: Graduate student ascent on the ImageNet dataset by year.

A partial solution: neural expressivity theory

Neural expressivity theory: properties of neural networks → expressivity and generalization capability ([RPK+16], [DFS16], [Gus16]).

E.g. shallow networks need exponentially many more units to express certain functions than deeper ones; deep networks can express curvature more efficiently, etc.

A partial solution: neural expressivity theory

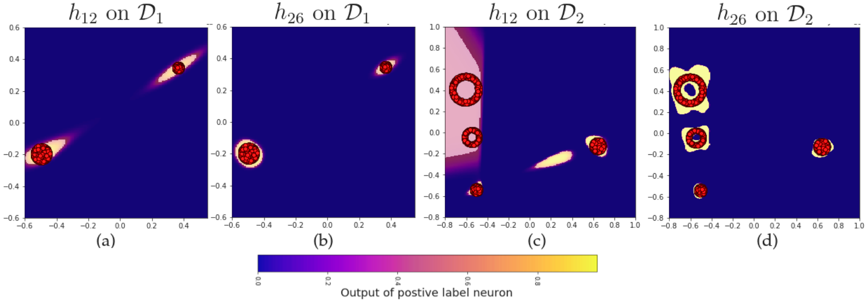
Neural expressivity theory: properties of neural networks → expressivity and generalization capability ([RPK$^+$16], [DFS16], [Gus16]).

E.g. shallow networks need exponentially many more units to express certain functions than deeper ones; deep networks can express curvature more efficiently, etc.

This can only be used to determine an architecture in practice if it is understood how expressive a model need be in order to solve a problem

Our approach: data-first architecture selection

Given a learning problem (some dataset), which architectures are suitably regularized and expressive enough to learn and generalize on that problem?



DAS: develop some objective measure of data complexity, and then characterize neural architectures by their ability to learn subject to that complexity.

Towards a measure of data complexity

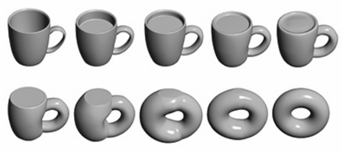
A brief introduction to topology

Topology characterizes shapes, spaces, and sets by their *connectivity*.

A brief introduction to topology

Topology characterizes shapes, spaces, and sets by their *connectivity*.

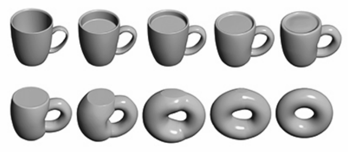
Def (Homeomorphism). Informally, we say that two topological spaces A and B are *equivalent* ($A \cong B$) if there is a continuous function $f : A \to B$ that has an inverse f^{-1} that is also continuous.



A brief introduction to topology

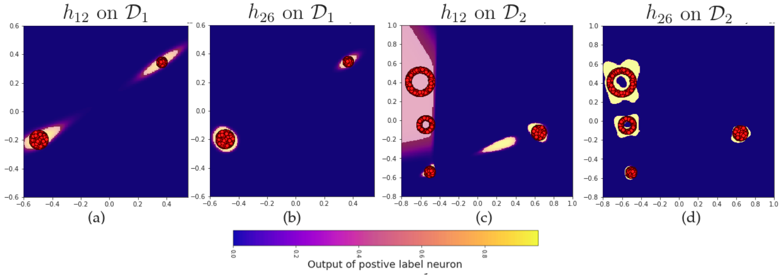
Topology characterizes shapes, spaces, and sets by their *connectivity*.

Def (Homeomorphism). Informally, we say that two topological spaces A and B are *equivalent* ($A \cong B$) if there is a continuous function $f : A \to B$ that has an inverse f^{-1} that is also continuous.



Topology can differentiate sets in a meaningful way that discards certain irrelevant properties such as rotation, translation, curvature, etc.

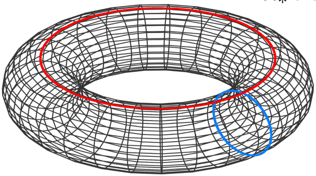
In this view $\mathcal{D}_1 \not\cong \mathcal{D}_2$. Furthermore h_{12} cannot express decision boundaries with the topology of $\mathcal{D}_2$.



We can characterize architectures by topological expressivity.

Algebraic topology lets us explicitly compute which spaces are equivalent to eachother.

$$H_* : \mathsf{Top}^2 \longrightarrow \mathsf{AbGrp}$$



$$\xrightarrow{H_1(\mathbb{T}) = \mathbb{Z} \oplus \mathbb{Z}}$$

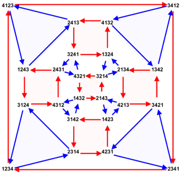


Figure: An illustration of the core philosophy of algebraic topology: 'functoraly' reduce hard problems in topology to easy ones in group theory.

Homology: a tool for computing topology

Def (Homology). If X is a topological space,

- $H_n(X) = \mathbb{Z}^{\beta_n}$ is called **the** ***n*****th homology group of** X.

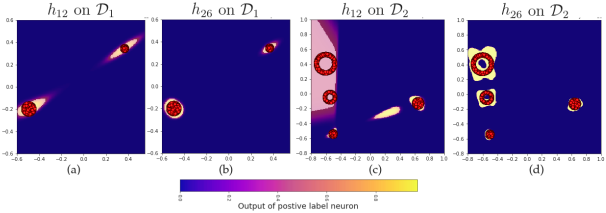
Def (Homology). If X is a topological space,

- $H_n(X) = \mathbb{Z}^{\beta_n}$ is called **the nth homology group of** X.
- β_n is the number of 'holes' of dimension n in X.

Homology: a tool for computing topology

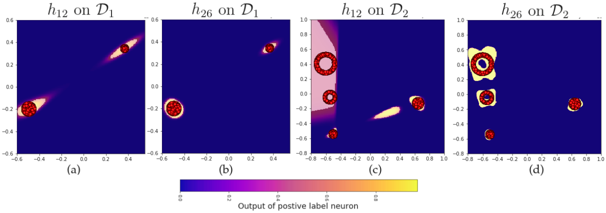
Def (Homology). If X is a topological space,

- $H_n(X) = \mathbb{Z}^{\beta_n}$ is called **the nth homology group of X**.
- β_n is the number of 'holes' of dimension n in X.



Def (Homology). If X is a topological space,

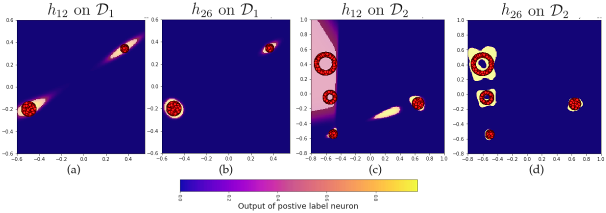
- $H_n(X) = \mathbb{Z}^{\beta_n}$ is called **the** n**th homology group of** X.
- β_n is the number of 'holes' of dimension n in X.



$$H_0(\mathcal{D}_1) = \mathbb{Z}^2$$
$$H_1(\mathcal{D}_1) = \{0\}$$
$$H_2(\mathcal{D}_1) = \{0\}$$

Def (Homology). If X is a topological space,

- $H_n(X) = \mathbb{Z}^{\beta_n}$ is called **the nth homology group of X**.
- β_n is the number of 'holes' of dimension n in X.

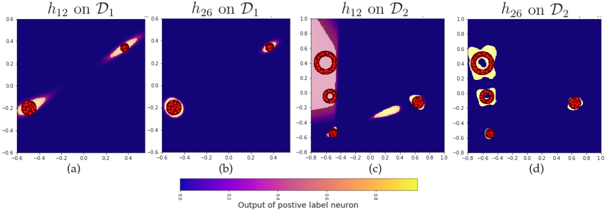


$$H_0(\mathcal{D}_1) = \mathbb{Z}^2$$
$$H_1(\mathcal{D}_1) = \{0\}$$
$$H_2(\mathcal{D}_1) = \{0\}$$

$$H_0(\mathcal{D}_2) = \mathbb{Z}^4$$
$$H_1(\mathcal{D}_2) = \mathbb{Z}^2$$
$$H_2(\mathcal{D}_2) = \{0\}$$

Def (Homology). If X is a topological space,

- $H_n(X) = \mathbb{Z}^{\beta_n}$ is called **the nth homology group of X**.
- β_n is the number of 'holes' of dimension n in X.



$$H_0(\mathcal{D}_1) = \mathbb{Z}^2$$
$$H_1(\mathcal{D}_1) = \{0\}$$
$$H_2(\mathcal{D}_1) = \{0\}$$

$$H_0(\mathcal{D}_2) = \mathbb{Z}^4$$
$$H_1(\mathcal{D}_2) = \mathbb{Z}^2$$
$$H_2(\mathcal{D}_2) = \{0\}$$

$H(\mathcal{D}_1) \leqslant H(\mathcal{D}_2)$ **and $\mathcal{D}_2$ requires more complex architectures**

Homology: a tool for computing topology

Def (Homology). If X is a topological space, then

- $H_n(X) = \mathbb{Z}^{\beta_n}$ is called **the** n**th homology group of** X if β_n is the number of 'holes' of dimension n in X.
- the homology of X is defined as $H(X) = \{H_n(X)\}_{n=0}^{\infty}$.

Theorem (Bredon). If $X \cong Y$ then $H(X) = H(Y)$.

Homological Characterization of Neural Architectures

The power of homological characterization

Homology is a stringent measure for characterizing neural architectures:

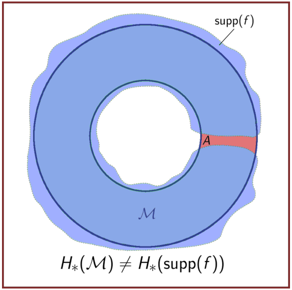


Figure: When homology cannot be expressed, there exists a misclassified set.

The power of homological characterization

Homology is a stringent measure for characterizing neural architectures:

Let $\mathcal{M} \subset X$ be a topological manifold.

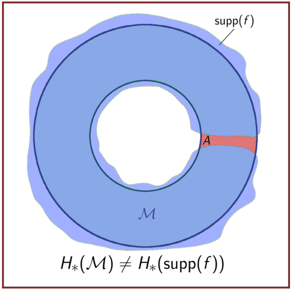


Figure: When homology cannot be expressed, there exists a misclassified set.

The power of homological characterization

Homology is a stringent measure for characterizing neural architectures:

Let $\mathcal{M} \subset X$ be a topological manifold.

Let $H_S(f)$ be the homology of the support of f, i.e. $H_S(f) = H(\{x \ : \ f(x) > 0\})$.

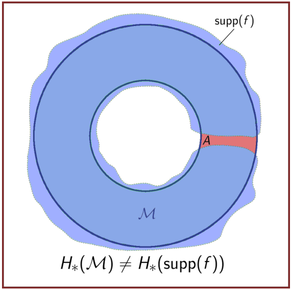


Figure: When homology cannot be expressed, there exists a misclassified set.

The power of homological characterization

Homology is a stringent measure for characterizing neural architectures:

Let $\mathcal{M} \subset X$ be a topological manifold.

Let $H_S(f)$ be the homology of the support of f, i.e. $H_S(f) = H(\{x \ : \ f(x) > 0\})$.

Theorem (The Homological Principle of Generalization). If for all $f \in \mathcal{F}$, $H_S(f) \neq H(\mathcal{M})$, then there exists $A \subset X$ such that f missclassifies A.

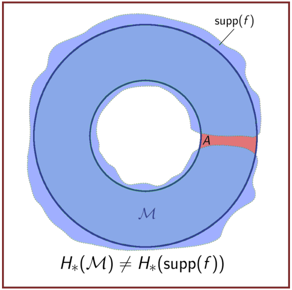
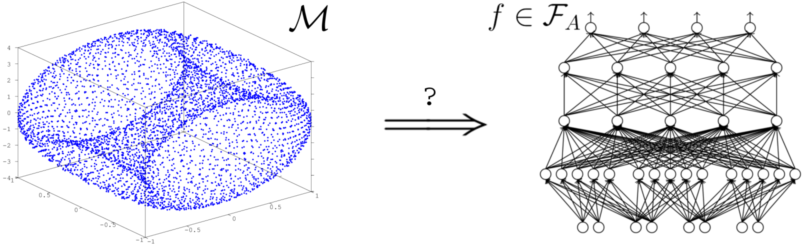


Figure: When homology cannot be expressed, there exists a misclassified set.

Architecture selection in the lense of topology

Given a dataset $\mathcal{M}$, for which architectures A does there exist a neural network $f \in \mathcal{F}_A$ such that $H_S(f) = H(\mathcal{M})$?



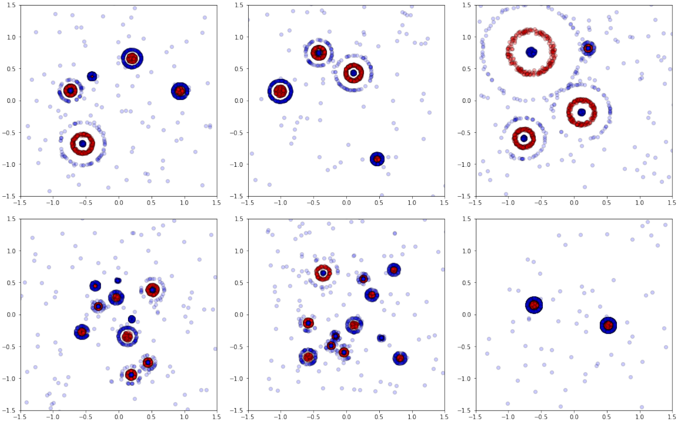
An empirical approach

We'll first try and answer this question empirically.

An empirical approach: Synthetic data

We'll first try and answer this question empirically.

1. Generate data of increasing homological complexity.



An empirical approach: Persistent homology

We'll first try and answer this question empirically.

1. Generate data of increasing homological complexity.
2. Compute the *persistent homology* of the synthetic data.

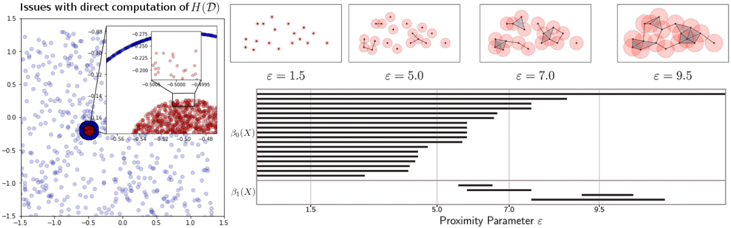


Figure: Left: The local disconnectedness of datasets prevents direct computation of their homology. Right: An illustration of computing persistent homology on a collection of points ([TZH15])

An empirical approach: Persistent homology

We'll first try and answer this question empirically.

1. Generate data of increasing homological complexity.
2. (Sanity check) Real data is homologically rich!

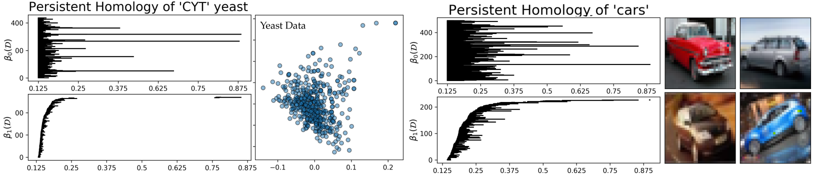
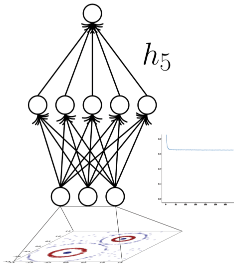


Figure: The persistent homology barcodes of classes in the CIFAR-10 and UCI Protein Localization Datasets.

An empirical approach: Characterization

We'll first try and answer this question empirically.

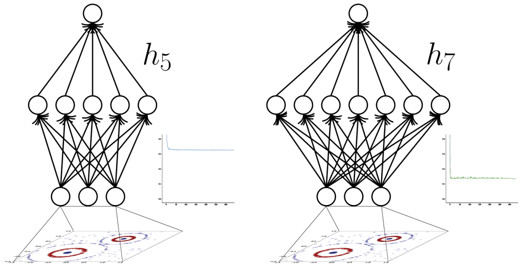
1. Generate data of increasing homological complexity.
2. Compute the *persistent homology* of the synthetic data.
3. Train models of increasing architectural complexity.



An empirical approach: Characterization

We'll first try and answer this question empirically.

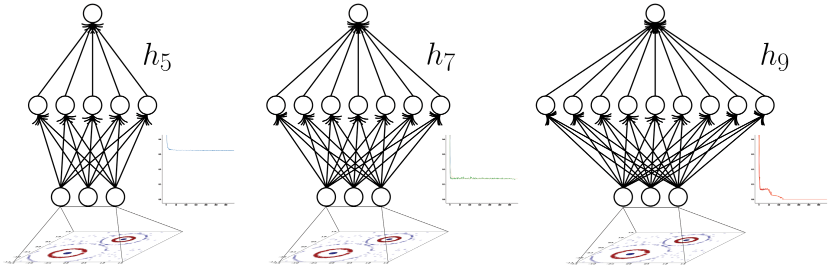
1. Generate data of increasing homological complexity.
2. Compute the *persistent homology* of the synthetic data.
3. Train models of increasing architectural complexity.



An empirical approach: Characterization

We'll first try and answer this question empirically.

1. Generate data of increasing homological complexity.
2. Compute the *persistent homology* of the synthetic data.
3. Train models of increasing architectural complexity.



Empirical results: Topological phase transitions

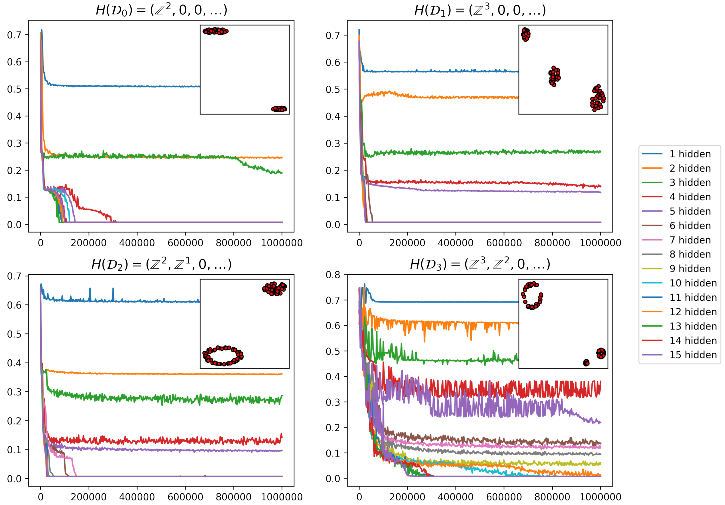


Figure: Average testing error of different architectures on datasets of increasing homological complexity.

Empirical results: Topological phase transitions

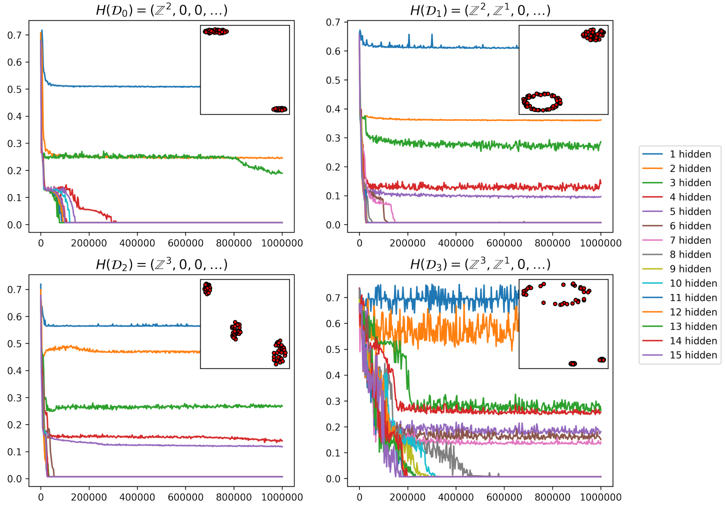
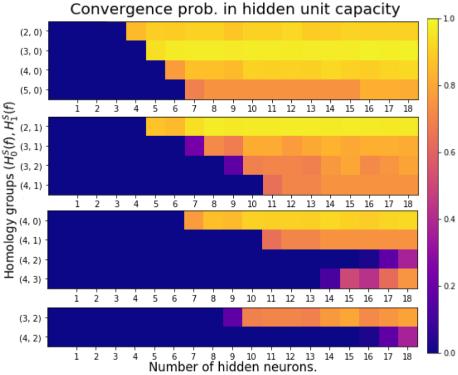


Figure: Average testing error of different architectures on datasets of increasing homological complexity.

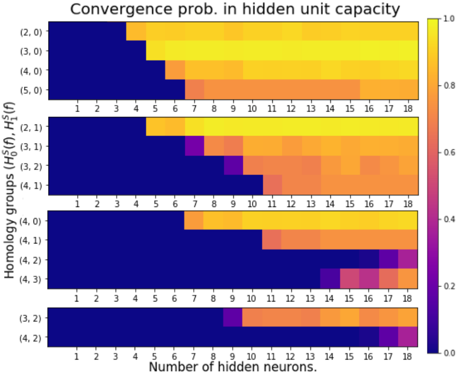
There are integral relationships between homological complexity and learnable architectures.

Eg.



There are integral relationships between homological complexity and learnable architectures.

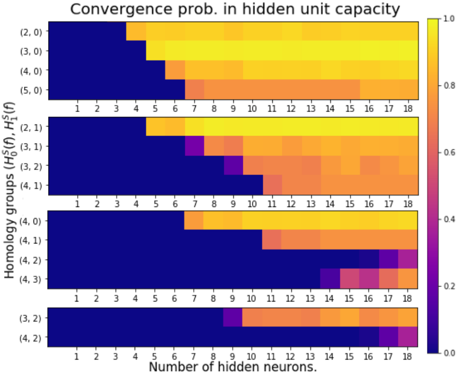
Eg. $H_0(\mathcal{D}) = \mathbb{Z}^m \implies$ there exists a NN with $h = m + 2$ that converges to zero error $\mathcal{D}$.



There are integral relationships between homological complexity and learnable architectures.

Eg. $H_0(\mathcal{D}) = \mathbb{Z}^m \implies$ there exists a NN with $h = m + 2$ that converges to zero error $\mathcal{D}$.

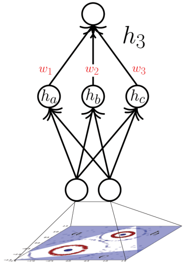
$H_0(\mathcal{D}) = \mathbb{Z}^m$ and $H_1(\mathcal{D}) = \mathbb{Z} \implies$ the same holds with $h \geqslant 3m - 1$.



Neural Homology Theory

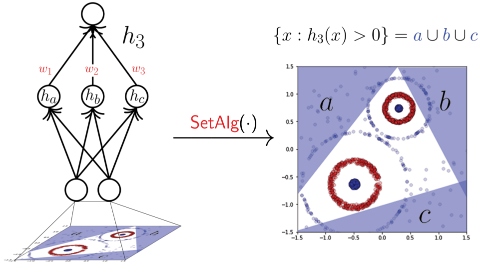
Given a dataset $\mathcal{M}$, for which architectures A does there exist a neural network $f \in \mathcal{F}_A$ such that $H_S(f) = H(\mathcal{M})$?

Neural Homology Theory.



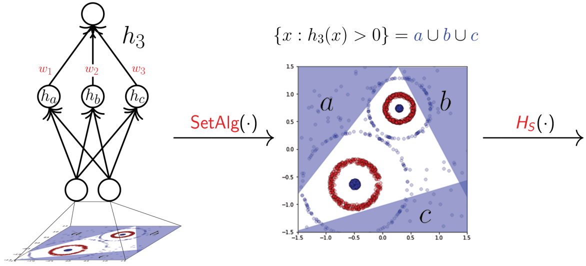
Given a dataset $\mathcal{M}$, for which architectures A does there exist a neural network $f \in \mathcal{F}_A$ such that $H_S(f) = H(\mathcal{M})$?

Neural Homology Theory.



Given a dataset $\mathcal{M}$, for which architectures A does there exist a neural network $f \in \mathcal{F}_A$ such that $H_S(f) = H(\mathcal{M})$?

Neural Homology Theory.



$$H_S^0(h_3) \cong \frac{a + b \oplus c}{\left(\frac{ac \oplus bc}{abc}\right)}$$

$$\boxed{\#\text{Regions} = \frac{1 \oplus 1}{\left(\frac{0 \oplus 1}{0}\right)} \approx 2 - 1 = 1}$$

Neural networks as set expressions.

Neural networks with ReLu-like functions can be converted into set expression.

1. The decision region of a single hidden unit is a halfspace.

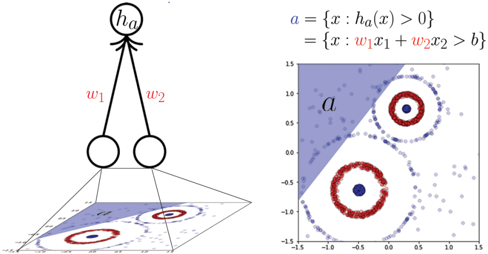
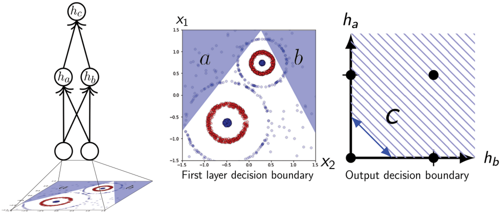


Figure: An illustration of halfspaces induced by ReLu-like networks.

Neural networks with ReLu-like functions can be converted into set expression.

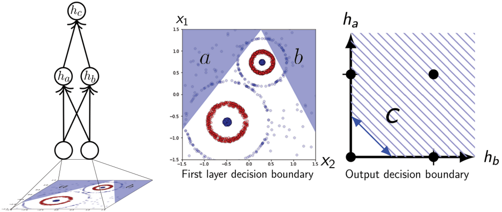
1. The decision region of a single hidden unit is a halfspace.
2. The decision region of deeper hidden units applies $\cap, \cup, \neg$ to shallower decision regions.



$$\{x : h_c(x) > 0\} =$$

Neural networks with ReLu-like functions can be converted into set expression.

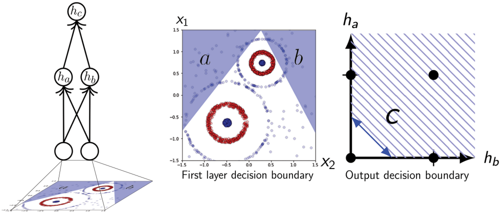
1. The decision region of a single hidden unit is a halfspace.
2. The decision region of deeper hidden units applies $\cap, \cup, \neg$ to shallower decision regions.



$$\{x : h_c(x) > 0\} = (a \cap \neg b)$$

Neural networks with ReLu-like functions can be converted into set expression.

1. The decision region of a single hidden unit is a halfspace.
2. The decision region of deeper hidden units applies $\cap, \cup, \neg$ to shallower decision regions.

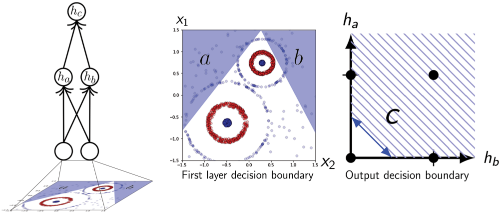


$$\{x : h_c(x) > 0\} = (a \cap \neg b) \cup (\neg a \cap b)$$

Neural networks as set expressions.

Neural networks with ReLu-like functions can be converted into set expression.

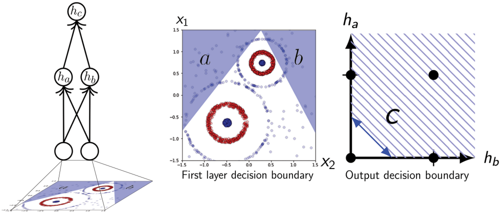
1. The decision region of a single hidden unit is a halfspace.
2. The decision region of deeper hidden units applies $\cap, \cup, \neg$ to shallower decision regions.



$$\{x : h_c(x) > 0\} = (a \cap \neg b) \cup (\neg a \cap b) \cup (a \cap b)$$

Neural networks with ReLu-like functions can be converted into set expression.

1. The decision region of a single hidden unit is a halfspace.
2. The decision region of deeper hidden units applies $\cap, \cup, \neg$ to shallower decision regions.



$$\{x : h_c(x) > 0\} = (a \cap \neg b) \cup (\neg a \cap b) \cup (a \cap b)$$
$$= a \cup b$$

Neural networks with ReLu-like functions can be converted into set expression.

1. The decision region of a single hidden unit is a halfspace.
2. The decision region of deeper hidden units applies $\cap, \cup, \neg$ to shallower decision regions.

Formally:

$$\{x : N(x) > 0\} = \bigcup_{\gamma \in S(W)} \bigcap_{h_i \in \text{hidden}} \{x : \text{sign}(\gamma_i) h_i(x) > 0\}$$

with $N(x) = ReLu(W^T h(x))$ and

$$S(W) = \{h : W^T h > 0\} \cap \{-1, 1\}^{|\text{hidden}|}.$$

The homology of decision regions is computed with tools for computing homology on set expressions.

1. *Notation*. For half spaces a, b denote

The homology of decision regions is computed with tools for computing homology on set expressions.

1. *Notation*. For half spaces a, b denote
   - The union $a + b := a \cup b$

The homology of decision regions is computed with tools for computing homology on set expressions.

1. *Notation*. For half spaces a, b denote
   - The union $a + b := a \cup b$
   - The intersection $ab := a \cap b$

The homology of decision regions is computed with tools for computing homology on set expressions.

1. *Notation*. For half spaces a, b denote
    - The union $a + b := a \cup b$
    - The intersection $ab := a \cap b$
    - The "sum" of homologies $H(a) \oplus H(b)$

The homology of decision regions is computed with tools for computing homology on set expressions.

1. *Notation*. For half spaces a, b denote
   - The union $a + b := a \cup b$
   - The intersection $ab := a \cap b$
   - The "sum" of homologies $H(a) \oplus H(b)$
   - The "subtraction of homologies" $H(a)/H(b)$.

Homology of set algebra on halfspaces

The homology of decision regions is computed with tools for computing homology on set expressions.

1. *Notation*. For half spaces a, b denote
   - The union $a + b := a \cup b$
   - The intersection $ab := a \cap b$
   - The "sum" of homologies $H(a) \oplus H(b)$
   - The "subtraction of homologies" $H(a)/H(b)$.
   - For the rest of the talk we'll drop H; $a := H(a)$.

The homology of decision regions is computed with tools for computing homology on set expressions.

1. *Notation*. For half spaces a, b denote
    - The union $a + b := a \cup b$
    - The intersection $ab := a \cap b$
    - The "sum" of homologies $H(a) \oplus H(b)$
    - The "subtraction of homologies" $H(a)/H(b)$.
    - For the rest of the talk we'll drop H; $a := H(a)$.

For example,

$$\#\text{ holes } = \frac{a \oplus b}{ab + c \oplus d} := \frac{H(a) \oplus H(b)}{H(a \cap b \cup c) \oplus H(d)}$$

The homology of decision regions is computed with tools for computing homology on set expressions.

1. *Notation.*
2. *Neural Cotorsion Algebra.* We consider $H_0(\{x : N(x) > 0\})$

The homology of decision regions is computed with tools for computing homology on set expressions.

1. *Notation.*
2. *Neural Cotorsion Algebra.* We consider $H_0(\{x : N(x) > 0\})$
   - For any halfspace a, we know $a = 1$.

The homology of decision regions is computed with tools for computing homology on set expressions.

1. *Notation.*
2. *Neural Cotorsion Algebra.* We consider $H_0(\{x : N(x) > 0\})$
   - For any halfspace a, we know $a = 1$.
   - Rule #1: $a \oplus b = ab \oplus (a + b)$.

The homology of decision regions is computed with tools for computing homology on set expressions.

1. *Notation.*
2. *Neural Cotorsion Algebra.* We consider $H_0(\{x : N(x) > 0\})$
   - For any halfspace a, we know $a = 1$.
   - Rule #1: $a \oplus b = ab \oplus (a + b)$.
   - Rule #2: If $c = a \oplus b$ then $a = c/b$.

The homology of decision regions is computed with tools for computing homology on set expressions.

1. *Notation.*
2. *Neural Cotorsion Algebra.* We consider $H_0(\{x : N(x) > 0\})$
   - For any halfspace a, we know $a = 1$.
   - Rule #1: $a \oplus b = ab \oplus (a + b)$.
   - Rule #2: If $c = a \oplus b$ then $a = c/b$.
   - Rule #3: Normal set operations work: $a(b + c) = ab + ac, aabc = abc$.

The homology of decision regions is computed with tools for computing homology on set expressions.

1. *Notation.*
2. *Neural Cotorsion Algebra.* We consider $H_0(\{x : N(x) > 0\})$
   - For any halfspace a, we know $a = 1$.
   - Rule #1: $a \oplus b = ab \oplus (a + b)$.
   - Rule #2: If $c = a \oplus b$ then $a = c/b$.
   - Rule #3: Normal set operations work: $a(b + c) = ab + ac, aabc = abc$.

Example. Let a, b, c be halfspaces

The homology of decision regions is computed with tools for computing homology on set expressions.

1. *Notation.*
2. *Neural Cotorsion Algebra.* We consider $H_0(\{x : N(x) > 0\})$
   - For any halfspace a, we know $a = 1$.
   - Rule #1: $a \oplus b = ab \oplus (a + b)$.
   - Rule #2: If $c = a \oplus b$ then $a = c/b$.
   - Rule #3: Normal set operations work: $a(b + c) = ab + ac, aabc = abc$.

Example. Let a, b, c be halfspaces

$$H_0(a \cap b \cup c) =$$

The homology of decision regions is computed with tools for computing homology on set expressions.

1. *Notation.*
2. *Neural Cotorsion Algebra.* We consider $H_0(\{x : N(x) > 0\})$
   - For any halfspace a, we know $a = 1$.
   - Rule #1: $a \oplus b = ab \oplus (a + b)$.
   - Rule #2: If $c = a \oplus b$ then $a = c/b$.
   - Rule #3: Normal set operations work: $a(b + c) = ab + ac, aabc = abc$.

Example. Let a, b, c be halfspaces

$$H_0(a \cap b \cup c) = ab + c$$

The homology of decision regions is computed with tools for computing homology on set expressions.

1. *Notation.*
2. *Neural Cotorsion Algebra.* We consider $H_0(\{x : N(x) > 0\})$
   - For any halfspace a, we know $a = 1$.
   - Rule #1: $a \oplus b = ab \oplus (a + b)$.
   - Rule #2: If $c = a \oplus b$ then $a = c/b$.
   - Rule #3: Normal set operations work: $a(b + c) = ab + ac, aabc = abc$.

Example. Let a, b, c be halfspaces

$$H_0(a \cap b \cup c) = ab + c$$
$$= \frac{ab \oplus c}{abc}$$

The homology of decision regions is computed with tools for computing homology on set expressions.

1. *Notation.*
2. *Neural Cotorsion Algebra.* We consider $H_0(\{x : N(x) > 0\})$
   - For any halfspace a, we know $a = 1$.
   - Rule #1: $a \oplus b = ab \oplus (a + b)$.
   - Rule #2: If $c = a \oplus b$ then $a = c/b$.
   - Rule #3: Normal set operations work: $a(b + c) = ab + ac, aabc = abc$.

Example. Let a, b, c be halfspaces

$$H_0(a \cap b \cup c) = ab + c$$

$$= \frac{ab \oplus c}{abc} = \frac{a \oplus b \oplus c}{abc \oplus a + b}$$

The homology of decision regions is computed with tools for computing homology on set expressions.

1. *Notation.*
2. *Neural Cotorsion Algebra.* We consider $H_0(\{x : N(x) > 0\})$
   - For any halfspace a, we know $a = 1$.
   - Rule #1: $a \oplus b = ab \oplus (a + b)$.
   - Rule #2: If $c = a \oplus b$ then $a = c/b$.
   - Rule #3: Normal set operations work: $a(b + c) = ab + ac, aabc = abc$.

Example. Let a, b, c be halfspaces

$$\begin{aligned} H_0(a \cap b \cup c) &= ab + c \\ &= \frac{ab \oplus c}{abc} = \frac{a \oplus b \oplus c}{abc \oplus a + b} \\ &\approx 3 - H_0(a \cap b \cap c) + H_0(a \cup b). \end{aligned}$$

Neural homology theory for architecture selection.

Neural homology theory lets us answer questions in architecture selection.

Neural homology theory lets us answer questions in architecture selection.

Example. Can a 3 hidden unit neural network generalize on a dataset with $H_0(\mathcal{M}) = 10$?

Neural homology theory lets us answer questions in architecture selection.

Example. Can a 3 hidden unit neural network generalize on a dataset with
$$H_0(\mathcal{M}) = 10?$$

Neural homology theory says **no.**

Theorem

Any 3 neuron ReLu network fails on data with $\geqslant 4$ clusters.

Proof.

Let $h_3(x) = ReLu(w^T(h_a(x), h_b(x), h_c(x)))$.

Theorem

Any 3 neuron ReLu network fails on data with $\geqslant 4$ clusters.

Proof.

Let $h_3(x) = ReLu(w^T(h_a(x), h_b(x), h_c(x)))$.

1. [NN → Set Algebra] $h_3 > 0 = a \cup b \cup c$.

Theorem

Any 3 neuron ReLu network fails on data with $\geqslant 4$ clusters.

Proof.

Let $h_3(x) = ReLu(w^T(h_a(x), h_b(x), h_c(x)))$.

1. [NN $\rightarrow$ Set Algebra] $h_3 > 0 = a \cup b \cup c$.
2. [Set Algebra $\rightarrow$ Homology]

$$H_0(h_3 > 0) = a + b + c$$

Theorem

Any 3 neuron ReLu network fails on data with $\geqslant 4$ clusters.

Proof.

Let $h_3(x) = ReLu(w^T(h_a(x), h_b(x), h_c(x)))$.

1. [NN $\to$ Set Algebra] $h_3 > 0 = a \cup b \cup c$.
2. [Set Algebra $\to$ Homology]

$$H_0(h_3 > 0) = a + b + c$$
$$= \frac{a + b \oplus c}{(a+b)c}$$

Neural homology theory for architecture selection.

Theorem

Any 3 neuron ReLu network fails on data with $\geqslant 4$ *clusters.*

Proof.

Let $h_3(x) = ReLu(w^T(h_a(x), h_b(x), h_c(x)))$.

1. [NN → Set Algebra] $h_3 > 0 = a \cup b \cup c$.
2. [Set Algebra → Homology]

$$H_0(h_3 > 0) = a + b + c$$

$$= \frac{a + b \oplus c}{(a+b)c} \quad = \frac{a + b \oplus c}{ac + bc}$$

Theorem

Any 3 neuron ReLu network fails on data with $\geqslant 4$ *clusters.*

Proof.

Let $h_3(x) = ReLu(w^T(h_a(x), h_b(x), h_c(x)))$.

1. [NN → Set Algebra] $h_3 > 0 = a \cup b \cup c$.
2. [Set Algebra → Homology]

$$\begin{aligned} H_0(h_3 > 0) &= a + b + c \\ &= \frac{a + b \oplus c}{(a+b)c} \quad = \frac{a + b \oplus c}{ac + bc} \\ &= \frac{a + b \oplus c}{\left[\frac{ac \oplus bc}{acbc}\right]} \end{aligned}$$

Neural homology theory for architecture selection.

Theorem

Any 3 neuron ReLu network fails on data with $\geqslant 4$ *clusters.*

Proof.

Let $h_3(x) = ReLu(w^T(h_a(x), h_b(x), h_c(x)))$.

1. [NN → Set Algebra] $h_3 > 0 = a \cup b \cup c$.
2. [Set Algebra → Homology]

$$\begin{aligned} H_0(h_3 > 0) &= a + b + c \\ &= \frac{a + b \bigoplus c}{(a+b)c} \quad = \frac{a + b \bigoplus c}{ac + bc} \\ &= \frac{a + b \bigoplus c}{\left[\frac{ac \bigoplus bc}{acbc}\right]} \quad = \frac{a + b \bigoplus c}{\left[\frac{ac \bigoplus bc}{abc}\right]}. \end{aligned}$$

Theorem

Any 3 neuron ReLu network fails on data with $\geqslant 4$ clusters.

Proof.

Let $h_3(x) = ReLu(w^T(h_a(x), h_b(x), h_c(x)))$.

1. Then applying algebra

$$\frac{a + b \oplus c}{\left[\frac{ac \oplus bc}{abc}\right]} \leqslant a + b \oplus c \leqslant 3.$$

$\square$

Neural homology theory for architecture selection.

Theorem

Any 3 neuron ReLu network fails on data with $\geqslant 4$ clusters.

Proof.

Let $h_3(x) = ReLu(w^T(h_a(x), h_b(x), h_c(x)))$.

1. Then applying algebra

$$\frac{a + b \oplus c}{\left[\frac{ac \oplus bc}{abc}\right]} \leqslant a + b \oplus c \leqslant 3.$$

2. Therefore

$$\# \text{ Regions} = H_0(h_3 > 0) \leqslant 3.$$

□

Remarks & Questions

[Bre13] Glen E Bredon. *Topology and geometry*, volume 139. Springer Science & Business Media, 2013.

[DFS16] Amit Daniely, Roy Frostig, and Yoram Singer. Toward deeper understanding of neural networks: The power of initialization and a dual view on expressivity. In *Advances In Neural Information Processing Systems*, pages 2253–2261, 2016.

[RPK+16] Maithra Raghu, Ben Poole, Jon Kleinberg, Surya Ganguli, and Jascha Sohl-Dickstein. On the expressive power of deep neural networks. *arXiv preprint arXiv:1606.05336*, 2016.

[TZH15] Chad M Topaz, Lori Ziegelmeier, and Tom Halverson. Topological data analysis of biological aggregation models. *PloS one*, 10(5):e0126383, 2015.

Appendix

Homology

Definition (Homology Theory, [Bre13])

A homology theory on the on Top^2 is a function $H : Top^2 \to Ab$ assigning to each pair (X, A) of spaces a graded (abelian) group $\{H_p(X, A)\}$, and to each map $f : (X, A) \to (Y, B)$, homomorphisms $f_* : H_p(X, A) \to H_p(Y, B)$, together with a natural transformation of functors $\partial_* : H_p(X, A) \to H_{p-1}(X, A)$, called the connecting homomorphism (where we use $H_*(A)$ to denote $H_*(A, \varnothing)$) such that the following five axioms are satisfied.

Definition (Homology Theory, [Bre13])

1. If $f \simeq g : (X, A) \to (Y, B)$ then $f_* = g_* : H_*(X, A) \to H_*(Y, B)$.
2. For the inclusions $i : A \to X$ and $j : X \to (X, A)$ the sequence sequence of inclusions and connecting homomorphisms are exact.
3. Given the pair (X, A) and an open set $U \subset X$ such that $cl(U) \subset int(A)$ then the inclusion $k : (X - U, A - U) \to (X, A)$ induces an isomorphism $k_* : H_*(X - U, A - U) \to H_*(X, A)$
4. For a one point space P, $H_i(P) = 0$ for all $i \neq 0$.
5. For a topological sum $X = +_\alpha X_\alpha$ the homomorphism

$$\bigoplus (i_\alpha)_* : \bigoplus H_n(X_\alpha) \to H_n(X)$$

is an isomorphism, where $i_\alpha : X_\alpha \to X$ is the inclusion.