UNIVERSITAT DE BARCELONA

The topology of neural networks:

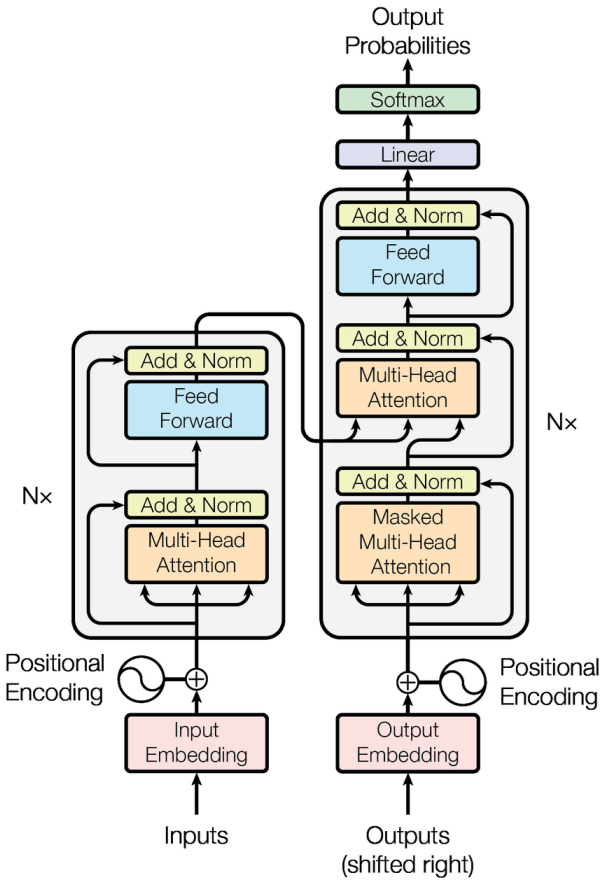
How topology helps us to understand generalization and future challenges

Rubén Ballester Bautista
Facultat de Matemàtiques i Informàtica
February 25th, 2022

Motivation (I)

UNIVERSITAT DE BARCELONA

Which is a better model?



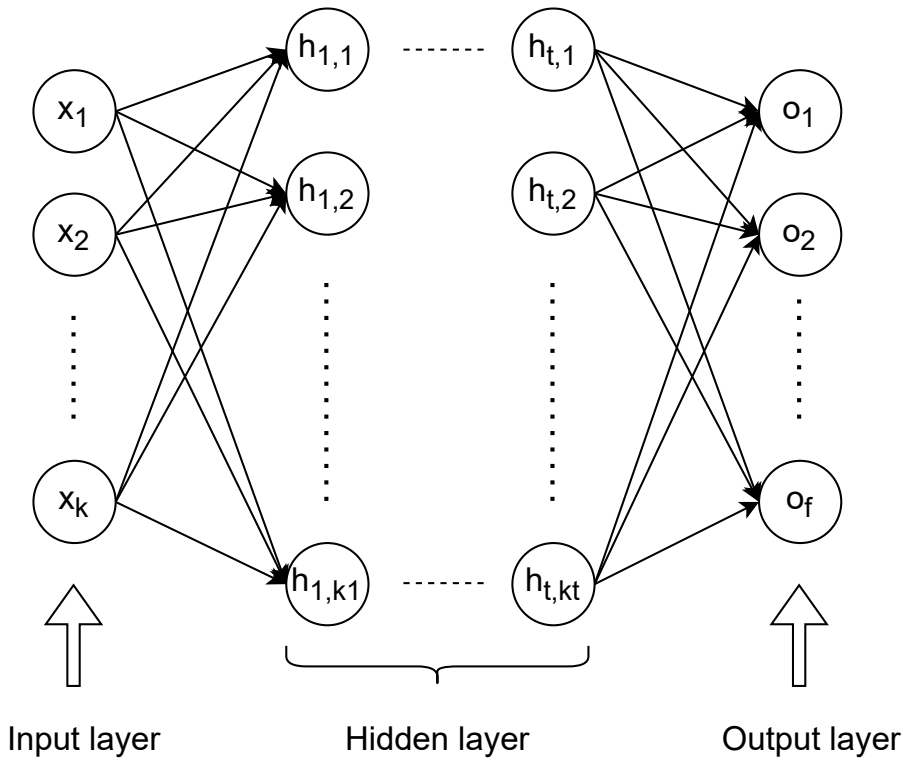


Image published in [1]

Motivation (II)

UNIVERSITAT DE BARCELONA

Universal approximation theorem: For any continuous function $f : \mathcal{X} \subseteq \mathbb{R}^d \to \mathbb{R}^D$, where $\mathcal{X}$ is a compact subset, there exists an enough complex fully-connected neural network $g : \mathbb{R}^d \to \mathbb{R}^D$ that approximates arbitrarily well f in $\mathcal{X}$, i.e. for all $\epsilon > 0$,

$$sup_{x \in \mathcal{X}} \|g(x) - f(x)\| < \epsilon. \tag{1}$$

Neural network capacity

UNIVERSITAT DE BARCELONA

- A classical measure of the capacity of a machine learning model m in a dataset $\mathcal{D}$ is the **accuracy**, i.e.,

$$Acc(\mathcal{D}, m) = \frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} \mathbb{1}(m(x) = y). \tag{2}$$

Neural network capacity

UNIVERSITAT DE BARCELONA

- A classical measure of the capacity of a machine learning model m in a dataset $\mathcal{D}$ is the **accuracy**, i.e.,

$$Acc(\mathcal{D}, m) = \frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} \mathbb{1}(m(x) = y). \tag{2}$$

- The accuracy is usually measured using a dataset not used during the training, called **test** dataset.

Neural network capacity

UNIVERSITAT DE BARCELONA

- A classical measure of the capacity of a machine learning model m in a dataset $\mathcal{D}$ is the **accuracy**, i.e.,

$$Acc(\mathcal{D}, m) = \frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} \mathbb{1}(m(x) = y). \tag{2}$$

- The accuracy is usually measured using a dataset not used during the training, called **test** dataset.
- Alternatively, we can compute the **generalization gap**, that measures the difference between the accuracy in the training dataset and the accuracy in the test dataset, i.e.,

$$GG(X_{train}, X_{test}, m) = Acc(X_{train}, m) - Acc(X_{test}, m). \tag{3}$$

Neural network capacity

UNIVERSITAT DE BARCELONA

- A classical measure of the capacity of a machine learning model m in a dataset $\mathcal{D}$ is the **accuracy**, i.e.,

$$Acc(\mathcal{D}, m) = \frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} \mathbb{1}(m(x) = y). \tag{2}$$

- The accuracy is usually measured using a dataset not used during the training, called **test** dataset.
- Alternatively, we can compute the **generalization gap**, that measures the difference between the accuracy in the training dataset and the accuracy in the test dataset, i.e.,

$$GG(X_{train}, X_{test}, m) = Acc(X_{train}, m) - Acc(X_{test}, m). \tag{3}$$

- Comparing generalization gap gives us a way to compare numerically two neural networks given a fixed dataset.

Presentation outline

1. Computational graphs and neural networks. Network dynamics matrices.
2. Neural network generalization gap prediction using TDA.
3. Conclusions and future work.

Computational graphs and neural networks. Network dynamics matrices.

Computational graphs (I)

UNIVERSITAT DE BARCELONA

- A **computational graph** is a function f represented by a directed graph whose non-input nodes represent functions and whose edges represent composition of functions.

Computational graphs (I)

UNIVERSITAT DE BARCELONA

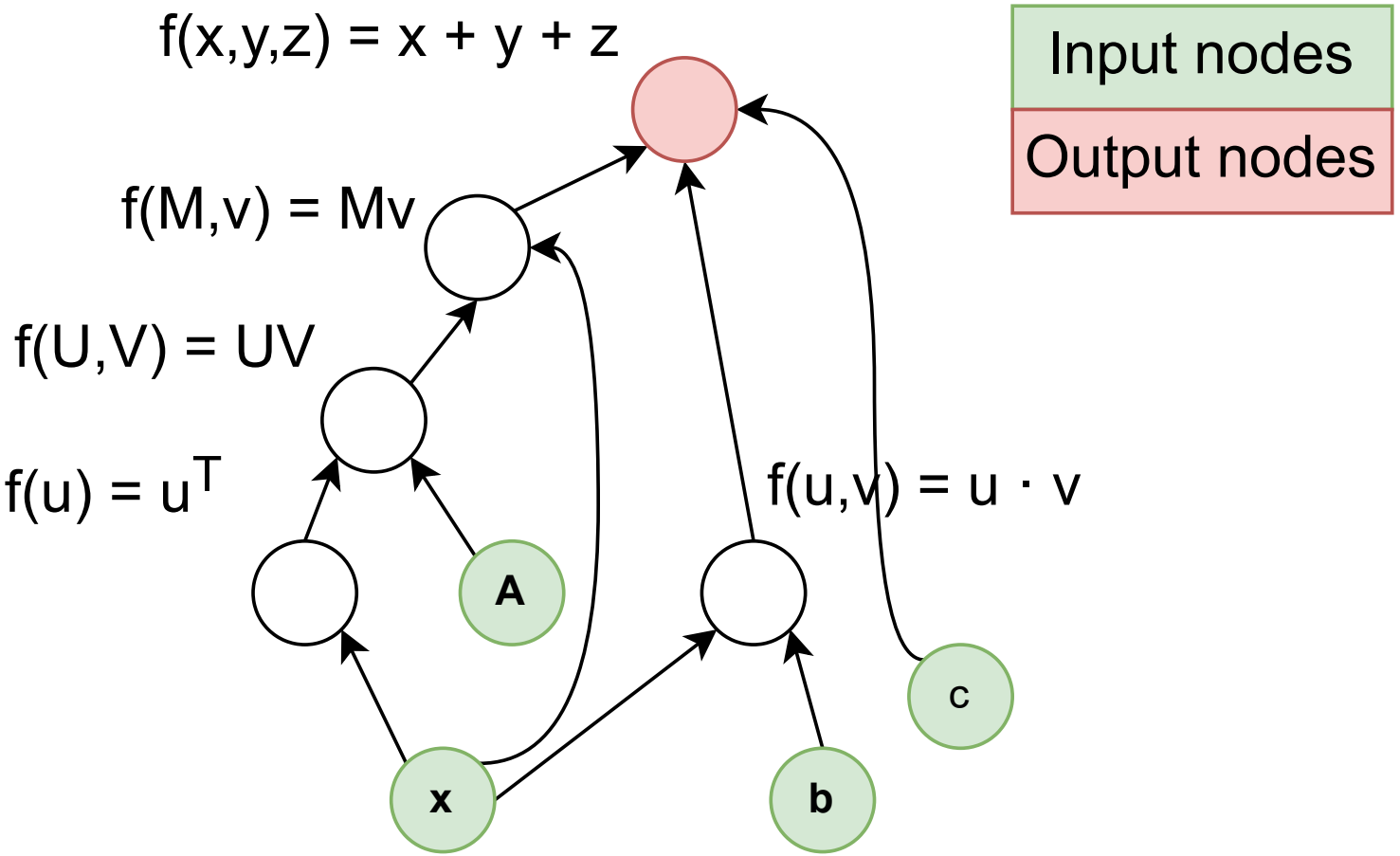
- A **computational graph** is a function f represented by a directed graph whose non-input nodes represent functions and whose edges represent composition of functions.
- Each node represents a function of the nodes pointing to it. The **input** of f is represented by the nodes of in-degree equal to zero. The **output** of f is represented by the nodes with out-degree equal to zero.

Computational graphs (I)

UNIVERSITAT DE BARCELONA

- A **computational graph** is a function f represented by a directed graph whose non-input nodes represent functions and whose edges represent composition of functions.
- Each node represents a function of the nodes pointing to it. The **input** of f is represented by the nodes of in-degree equal to zero. The **output** of f is represented by the nodes with out-degree equal to zero.
- Given a value for all the input nodes of f we can compute the values of all the functions represented by the nodes by concatenation of evaluations starting at the input vectors.

Computational graphs (II)



Computational graphs

UNIVERSITAT DE BARCELONA

- A **neural network** is a special kind of computational graph where non-input nodes model (almost-everywhere) differentiable functions $f_n : \mathbb{R}^m \to \mathbb{R}^1$.

[1] In general, nowadays there are many different models of neural networks.

Computational graphs

- A **neural network** is a special kind of computational graph where non-input nodes model (almost-everywhere) differentiable functions $f_n : \mathbb{R}^m \to \mathbb{R}^1$.
- Computational graphs are effective tools to model many mathematical functions, including, but not limited to, neural networks.

[1] In general, nowadays there are many different models of neural networks.

Computational graphs

- A **neural network** is a special kind of computational graph where non-input nodes model (almost-everywhere) differentiable functions $f_n : \mathbb{R}^m \to \mathbb{R}^1$.
- Computational graphs are effective tools to model many mathematical functions, including, but not limited to, neural networks.
- For example, **probabilistic circuits**[2], are a certain type of computational graphs that allows to prove theoretical results in probability theory. Probabilistic circuits were considered a major step in the field of tractable inference in the previous NeurIPS!

[1] In general, nowadays there are many different models of neural networks.

Network dynamics matrices (I)

UNIVERSITAT DE BARCELONA

- Let $\mathcal{D} = \{(x_i, y_i)\}_{i \in \{1,\dots,m\}} \subseteq \mathcal{X} \times \mathcal{Y}$ be a dataset. Let $m : \mathcal{X} \to \mathcal{Y}$ be a neural network as a computational graph. Define n_x to be the value of the neuron node n in the input x. Define the vector associated to the neuron n as $A(n, \mathcal{D}) = \{n_{x_1}, \dots, n_{x_m}\}$.

[1] We define **network dynamics matrices** for neural networks, but the definition can be generalized for general computational graphs.

Network dynamics matrices (I)

UNIVERSITAT DE BARCELONA

- Let $\mathcal{D} = \{(x_i, y_i)\}_{i \in \{1,\dots,m\}} \subseteq \mathcal{X} \times \mathcal{Y}$ be a dataset. Let $m : \mathcal{X} \to \mathcal{Y}$ be a neural network as a computational graph. Define n_x to be the value of the neuron node n in the input x. Define the vector associated to the neuron n as $A(n, \mathcal{D}) = \{n_{x_1}, \dots, n_{x_m}\}$.
- Define the point cloud of m as $PC(m, \mathcal{D}) := \{A(n, \mathcal{D}) : n \in V(m)\}$.

[1] We define **network dynamics matrices** for neural networks, but the definition can be generalized for general computational graphs.

Network dynamics matrices (I)

UNIVERSITAT DE BARCELONA

- Let $\mathcal{D} = \{(x_i, y_i)\}_{i \in \{1,\dots,m\}} \subseteq \mathcal{X} \times \mathcal{Y}$ be a dataset. Let $m : \mathcal{X} \to \mathcal{Y}$ be a neural network as a computational graph. Define n_x to be the value of the neuron node n in the input x. Define the vector associated to the neuron n as $A(n, \mathcal{D}) = \{n_{x_1}, \dots, n_{x_m}\}$.
- Define the point cloud of m as $PC(m, \mathcal{D}) := \{A(n, \mathcal{D}) : n \in V(m)\}$.
- Given a *distance* function d, the **network dynamics matrix** of m associated to $\mathcal{D}$ with distance d is the distance matrix of the points in $PC(m, \mathcal{D})$ associated to the *distance* d (given any ordering for the points).

[1] We define **network dynamics matrices** for neural networks, but the definition can be generalized for general computational graphs.

Network dynamics matrices (I)

UNIVERSITAT DE BARCELONA

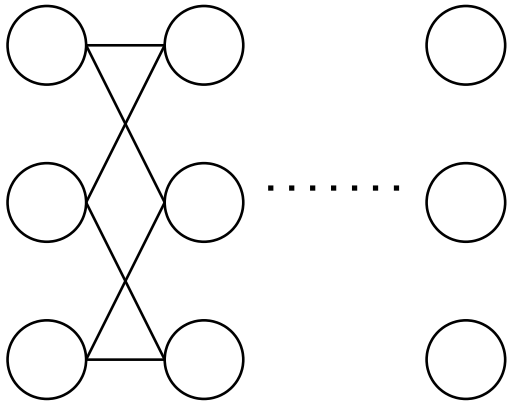
- Let $\mathcal{D} = \{(x_i, y_i)\}_{i \in \{1,\dots,m\}} \subseteq \mathcal{X} \times \mathcal{Y}$ be a dataset. Let $m : \mathcal{X} \to \mathcal{Y}$ be a neural network as a computational graph. Define n_x to be the value of the neuron node n in the input x. Define the vector associated to the neuron n as $A(n, \mathcal{D}) = \{n_{x_1}, \dots, n_{x_m}\}$.
- Define the point cloud of m as $PC(m, \mathcal{D}) := \{A(n, \mathcal{D}) : n \in V(m)\}$.
- Given a *distance* function d, the **network dynamics matrix** of m associated to $\mathcal{D}$ with distance d is the distance matrix of the points in $PC(m, \mathcal{D})$ associated to the *distance* d (given any ordering for the points).
- Note that this definition allows us to construct network dynamics matrices with a sample of the point cloud $PC' \subsetneq PC(m, \mathcal{D})$.

[1] We define **network dynamics matrices** for neural networks, but the definition can be generalized for general computational graphs.

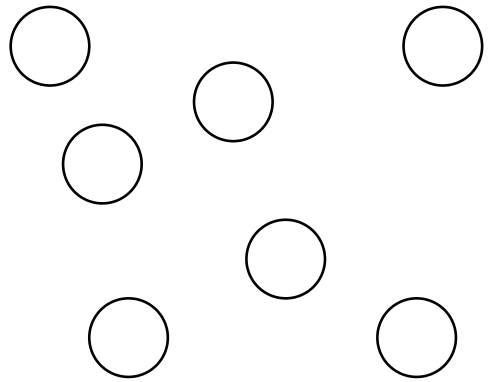
Network dynamics matrices (II)

(Training) Dataset with m examples

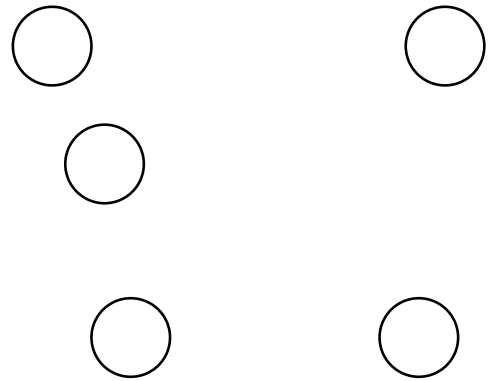
Neural network with n hidden neurons



n vectors of m components (one per neuron containing the activations of all the dataset)



n' <= n vectors of m components



Network dynamic matrix

$$\begin{pmatrix} d(A(n_1,\mathcal{D}),A(n_1,\mathcal{D})) & \dots & d(A(n_1,\mathcal{D}),A(n_{n'},\mathcal{D})) \\ \vdots & \ddots & \vdots \\ d(A(n_{n'},\mathcal{D}),A(n_1,\mathcal{D})) & \dots & d(A(n_{n'},\mathcal{D}),A(n_{n'},\mathcal{D})) \end{pmatrix}$$

UNIVERSITAT DE BARCELONA

Neural network generalization gap prediction using TDA.

Context (I)

UNIVERSITAT DE BARCELONA

- Increasing interest for **interpretability** and **explainability** in the community. Publication of many papers about predicting the generalization gap.

[1] It is interesting that TDA is applied also in the input.

Context (I)

UNIVERSITAT DE BARCELONA

- Increasing interest for **interpretability** and **explainability** in the community. Publication of many papers about predicting the generalization gap.
- Some papers use basic TDA for neural network interpretability. The space of parameters is explored. They proved to be very effective to understand several aspects of neural networks.

[1] It is interesting that TDA is applied also in the input.

Context (I)

UNIVERSITAT DE BARCELONA

- Increasing interest for **interpretability** and **explainability** in the community. Publication of many papers about predicting the generalization gap.
- Some papers use basic TDA for neural network interpretability. The space of parameters is explored. They proved to be very effective to understand several aspects of neural networks.
- In NeurIPS 2020, Google promotes a challenge where the goal is to order trained neural networks in a common dataset by their generalization gaps without access to the test dataset.

[1] It is interesting that TDA is applied also in the input.

Context (II)

UNIVERSITAT DE BARCELONA

- In 2019-2020, Ciprian A. Corneanu et al. publish two papers ([3] and [4]) using network dynamic matrices with the distance $d(x, y) = 1 - |corr(x, y)|$ to predict generalization gap of basic networks.

Context (II)

UNIVERSITAT DE BARCELONA

- In 2019-2020, Ciprian A. Corneanu et al. publish two papers ([3] and [4]) using network dynamic matrices with the distance $d(x, y) = 1 - |corr(x, y)|$ to predict generalization gap of basic networks.
- *Advantage*: These matrices take into account information about the **non-linear activation functions**.

Context (II)

UNIVERSITAT DE BARCELONA

- In 2019-2020, Ciprian A. Corneanu et al. publish two papers ([3] and [4]) using network dynamic matrices with the distance $d(x, y) = 1 - |corr(x, y)|$ to predict generalization gap of basic networks.
- *Advantage*: These matrices take into account information about the **non-linear activation functions**.
- Main objective: **Generalize** and **formalize** the methods of Corneanu et al.

Problem statement

Given a task $\mathcal{T}$ composed of a training dataset of inputs $\mathcal{D} \subseteq \mathcal{X} \times \mathcal{Y}$ and a set of trained neural networks in $\mathcal{D}$, $\mathcal{N} = \{m : \mathcal{X} \to \mathcal{Y}\}$ separated in training and test neural networks, generate a linear model with the training set of neural networks to predict the generalization gap of the neural networks in the test set.

Problem statement

Given a task $\mathcal{T}$ composed of a training dataset of inputs $\mathcal{D} \subseteq \mathcal{X} \times \mathcal{Y}$ and a set of trained neural networks in $\mathcal{D}$, $\mathcal{N} = \{m : \mathcal{X} \to \mathcal{Y}\}$ separated in training and test neural networks, generate a linear model with the training set of neural networks to predict the generalization gap of the neural networks in the test set.

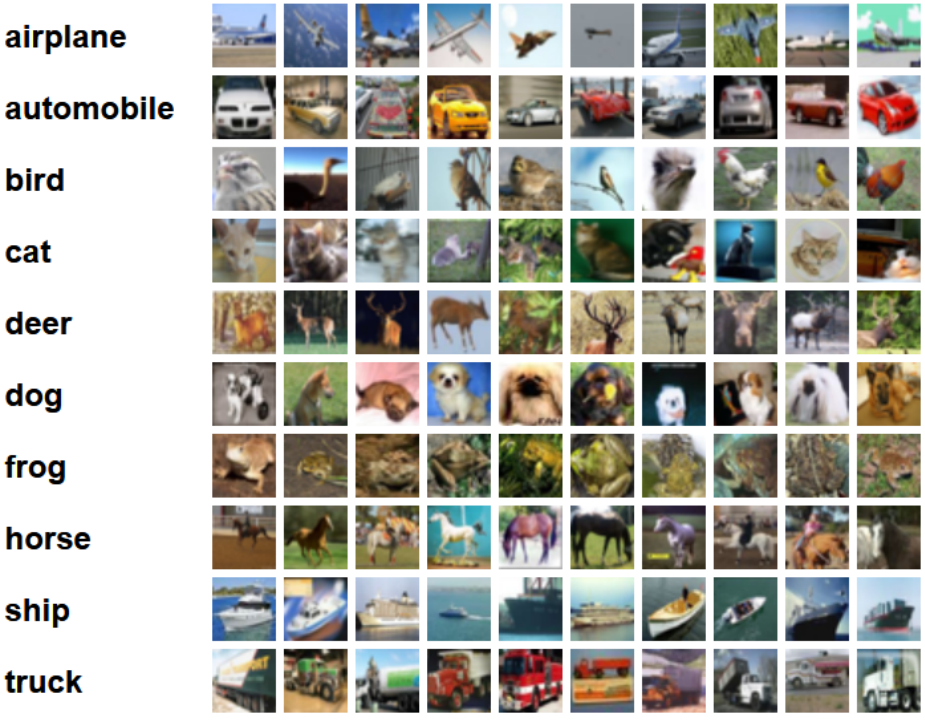
We worked with **two** different tasks.

Datasets

UNIVERSITAT DE BARCELONA

CIFAR-10

Ten classes of square, fixed size images.



96 VGG-like neural networks (CNN).

SVHN

Ten classes (digits 0-9) of square, fixed size images.

54 *NiN* CNN neural networks.

Pipeline (I)

- We want to learn a linear model $y = mx + n$ where x is some feature vector related with our neural network.

Pipeline (I)

UNIVERSITAT DE BARCELONA

- We want to learn a linear model $y = mx + n$ where x is some feature vector related with our neural network.
- We have seen in this seminar how to compute persistence diagrams from point clouds (equivalently, distance matrices). With these tools, we can compute persistence diagrams from network dynamics matrices.

Pipeline (I)

UNIVERSITAT DE BARCELONA

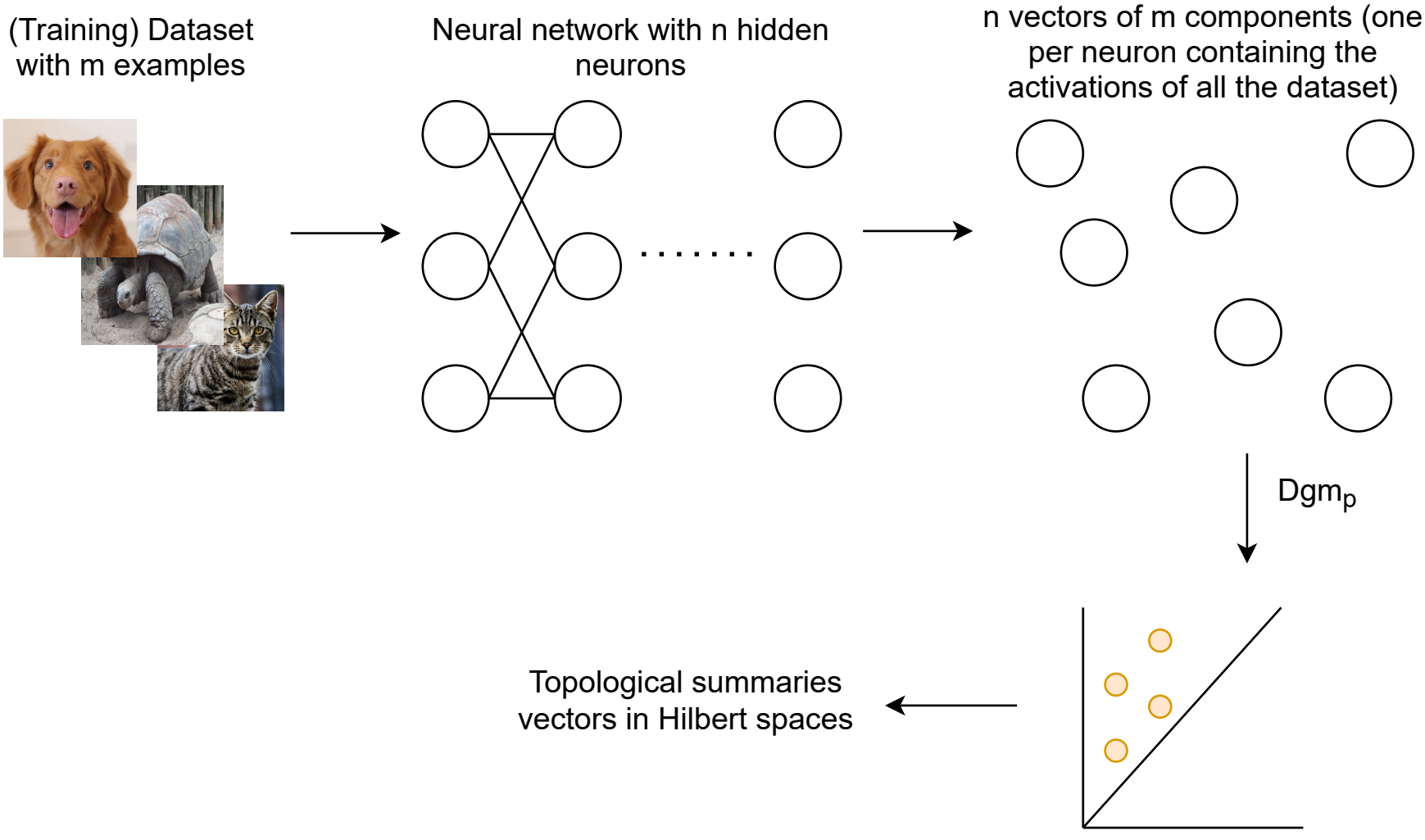
- We want to learn a linear model $y = mx + n$ where x is some feature vector related with our neural network.
- We have seen in this seminar how to compute persistence diagrams from point clouds (equivalently, distance matrices). With these tools, we can compute persistence diagrams from network dynamics matrices.
- Now we only need to generate **feature vectors** from our persistence diagrams!

Pipeline (I)

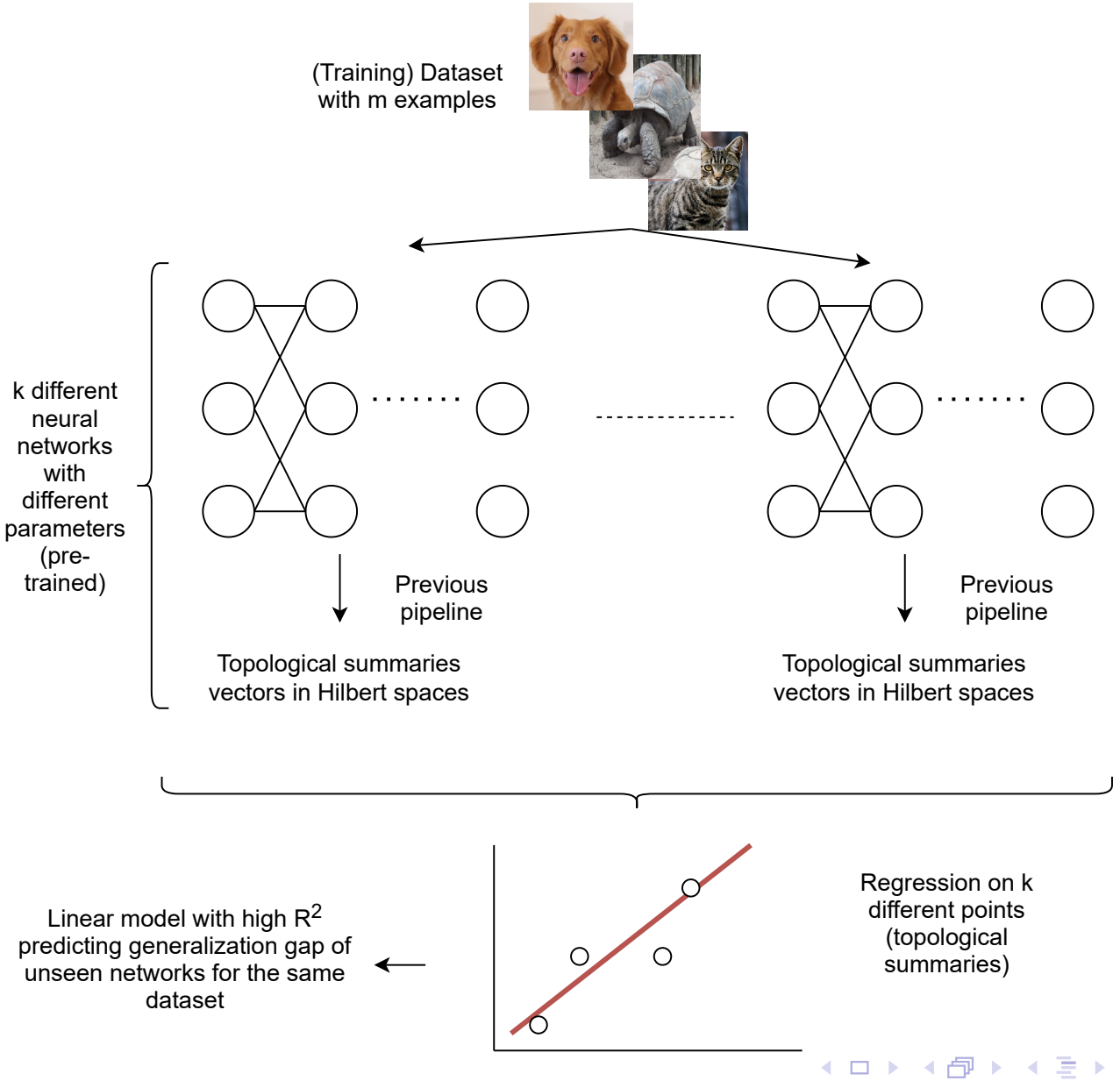
UNIVERSITAT DE BARCELONA

- We want to learn a linear model $y = mx + n$ where x is some feature vector related with our neural network.
- We have seen in this seminar how to compute persistence diagrams from point clouds (equivalently, distance matrices). With these tools, we can compute persistence diagrams from network dynamics matrices.
- Now we only need to generate **feature vectors** from our persistence diagrams!
- We saw many ways of doing that, in particular, the best vectorizations were some statistical quantities on the points of the persistence diagram, in particular the **standard deviations and averages** of the births and deaths.

Pipeline (II)


(Training) Dataset with m examples
Neural network with n hidden neurons
n vectors of m components (one per neuron containing the activations of all the dataset)
Dgm_p
Topological summaries vectors in Hilbert spaces

Pipeline (III)



Pipeline (IV)

Remarks:

- To generalize the method, we needed to **sample** the training dataset and the point cloud to generate the network dynamics matrices due to complexity of persistence homology algorithms.

Pipeline (IV)

Remarks:

- To generalize the method, we needed to **sample** the training dataset and the point cloud to generate the network dynamics matrices due to complexity of persistence homology algorithms.
- For the **dataset sampling**, we used a **uniform** random sampling over all the training dataset.

Pipeline (IV)

Remarks:

- To generalize the method, we needed to **sample** the training dataset and the point cloud to generate the network dynamics matrices due to complexity of persistence homology algorithms.
- For the **dataset sampling**, we used a **uniform** random sampling over all the training dataset.
- For the **neurons**, we sampled giving **more probability** to the nodes whose parameters, as a vector, had **more norm**.

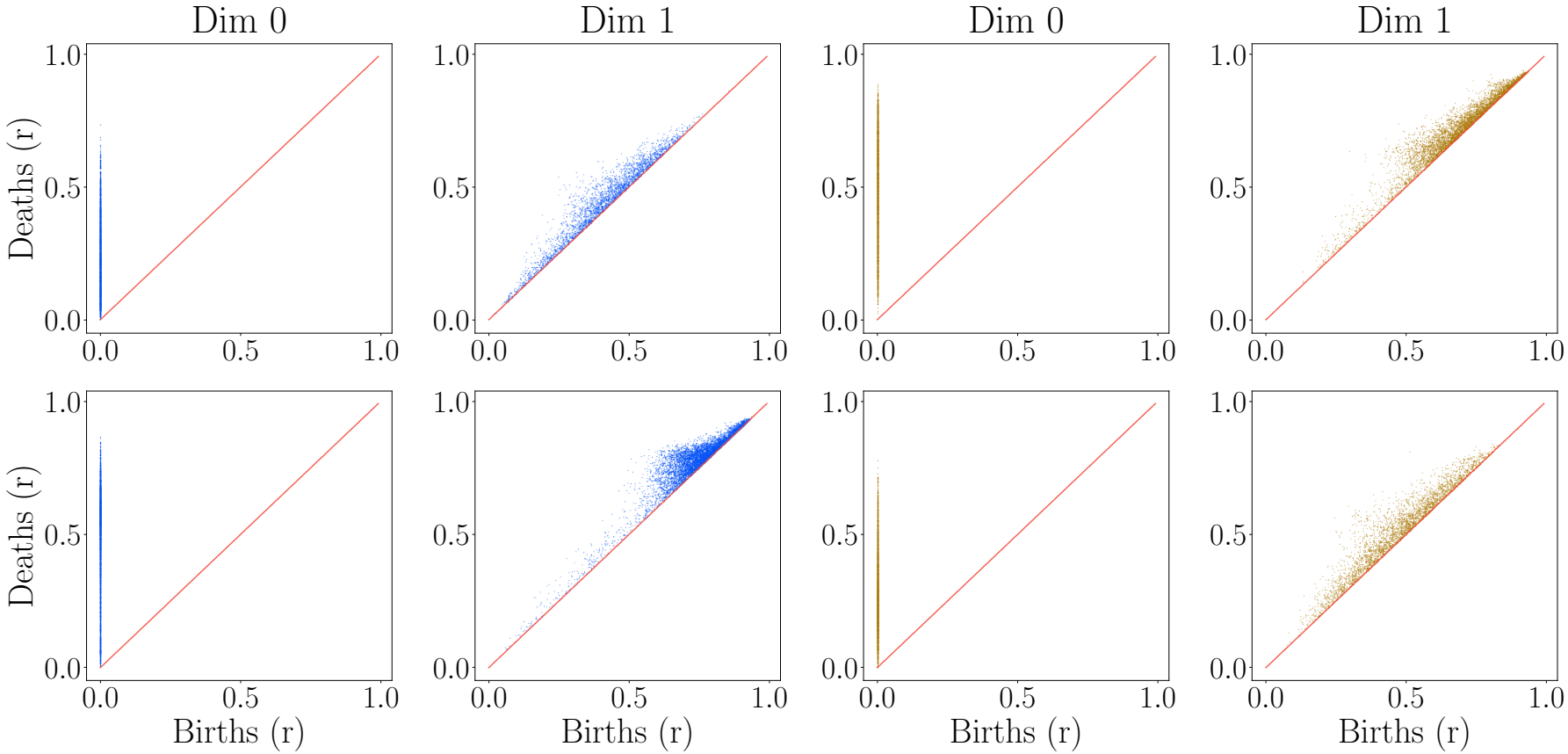
Results (I)

Comparison against the 3 winners of the Google challenge (R^2 obtained predicting generalization gap in the test set):

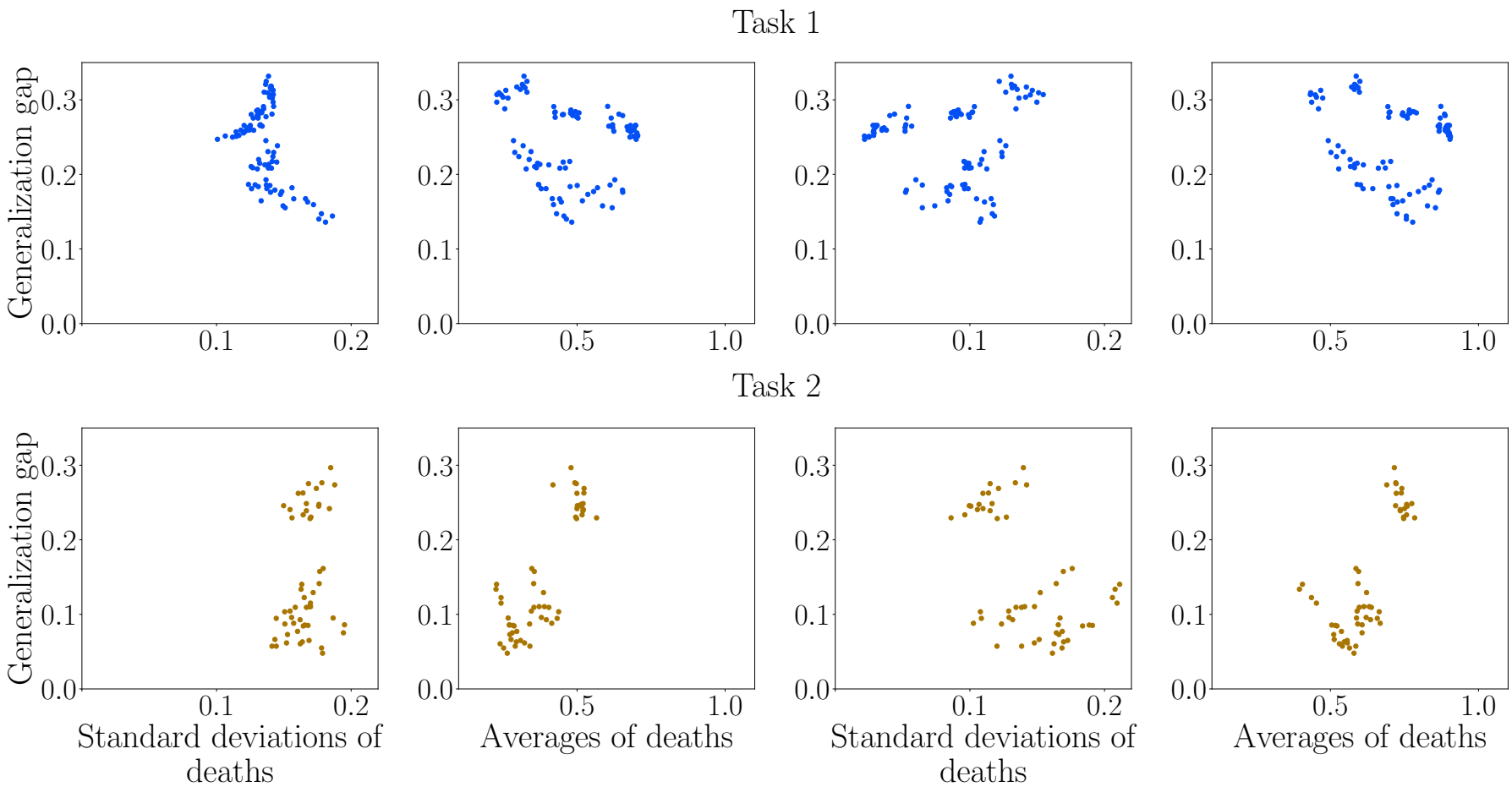
	Task 1	Task 2
Interpex	−0.1439	0.9776
Always Generalize	0.9715	0.8888
BrAIn	0.4079	0.6169
Ours	0.8663	0.9115

Results (II)

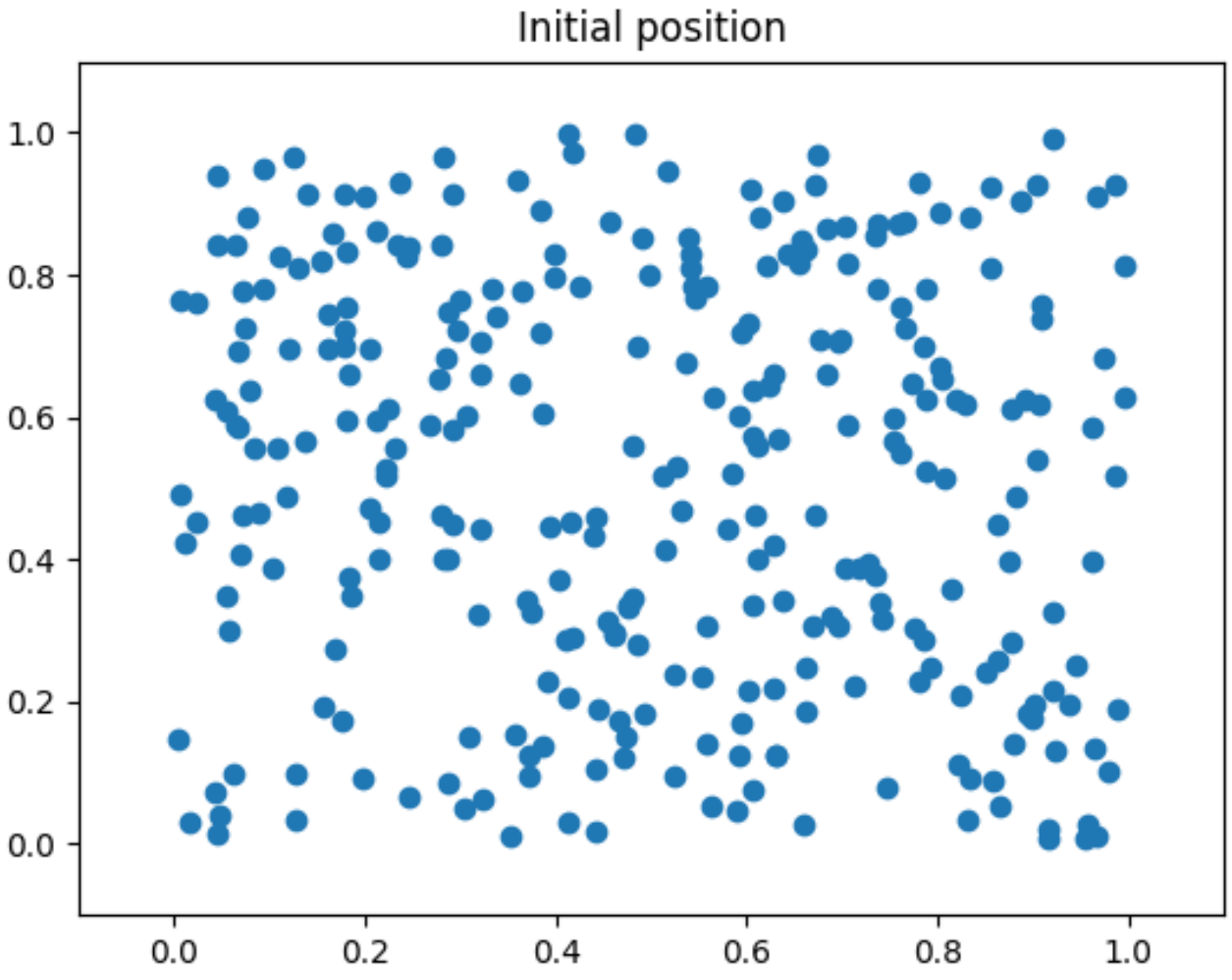
Persistence diagrams from neural networks with high (upper row) and low (lower row) generalization gaps. Task 1 - Task 2.



Results (III)


Task 1
Task 2
Generalization gap
0.0
0.1
0.2
0.3
0.1
0.2
0.5
1.0
Standard deviations of deaths
Averages of deaths

Future work: TDA on losses



Conclusions and limitations

UNIVERSITAT DE BARCELONA

- Topological features of network dynamic matrices are able to distinguish good and bad neural networks w.r.t. their generalization gap without using a test set.

Conclusions and limitations

UNIVERSITAT DE BARCELONA

- Topological features of network dynamic matrices are able to distinguish good and bad neural networks w.r.t. their generalization gap without using a test set.
- This somewhat implies that topology is important when we design and train neural networks.

Conclusions and limitations

UNIVERSITAT DE BARCELONA

- Topological features of network dynamic matrices are able to distinguish good and bad neural networks w.r.t. their generalization gap without using a test set.
- This somewhat implies that topology is important when we design and train neural networks.
- This procedure can be extended to any model that can be expressed as a computational graph.

Conclusions and limitations

UNIVERSITAT DE BARCELONA

- Topological features of network dynamic matrices are able to distinguish good and bad neural networks w.r.t. their generalization gap without using a test set.
- This somewhat implies that topology is important when we design and train neural networks.
- This procedure can be extended to any model that can be expressed as a computational graph.
- The distance $1 - |corr(x, y)|$ only captures linear correlation between neurons. Also, architecture is not taken into account. A better distance is needed.

Conclusions and limitations

UNIVERSITAT DE BARCELONA

- Topological features of network dynamic matrices are able to distinguish good and bad neural networks w.r.t. their generalization gap without using a test set.
- This somewhat implies that topology is important when we design and train neural networks.
- This procedure can be extended to any model that can be expressed as a computational graph.
- The distance $1 - |corr(x, y)|$ only captures linear correlation between neurons. Also, architecture is not taken into account. A better distance is needed.

Conclusions and limitations

UNIVERSITAT DE BARCELONA

- Topological features of network dynamic matrices are able to distinguish good and bad neural networks w.r.t. their generalization gap without using a test set.
- This somewhat implies that topology is important when we design and train neural networks.
- This procedure can be extended to any model that can be expressed as a computational graph.
- The distance $1 - |corr(x, y)|$ only captures linear correlation between neurons. Also, architecture is not taken into account. A better distance is needed.
- Sampling must be improved.

Conclusions and limitations

UNIVERSITAT DE BARCELONA

- Topological features of network dynamic matrices are able to distinguish good and bad neural networks w.r.t. their generalization gap without using a test set.
- This somewhat implies that topology is important when we design and train neural networks.
- This procedure can be extended to any model that can be expressed as a computational graph.
- The distance $1 - |corr(x, y)|$ only captures linear correlation between neurons. Also, architecture is not taken into account. A better distance is needed.
- Sampling must be improved.
- The procedure depends on a dataset of neural networks.

UNIVERSITAT DE BARCELONA

Questions?

References I

[1] A. Vaswani, S. Bengio, E. Brevdo, *et al.*, "Tensor2tensor for neural machine translation", Mar. 2018.

[2] A. Vergari, Y. Choi, A. Liu, S. Teso, and G. V. den Broeck, *A compositional atlas of tractable circuit operations: From simple transformations to complex information-theoretic queries*, 2021. arXiv: 2102.06137 [stat.ML].

[3] C. Corneanu, M. Madadi, S. Escalera, and A. Martinez, "What does it mean to learn in deep networks? and, how does one detect adversarial attacks?," Jun. 2019, pp. 4752–4761. DOI: 10.1109/CVPR.2019.00489.

[4] ——, *Computing the testing error without a testing set*, 2020. arXiv: 2005.00450 [cs.CV].