UNIVERSITAT DE BARCELONA

FUNDAMENTAL PRINCIPLES OF DATA SCIENCE MASTER'S THESIS

Sampling Methods for Activation Correlation Graphs to Predict Neural Network Generalization Using Topological Data Analysis

Author:
Otis CARPAY

Supervisors:
Dr. Sergio ESCALERA
Rubén BALLESTER

A thesis submitted in partial fulfillment of the requirements for the degree of MSc in Fundamental Principles of Data Science

in the

Facultat de Matemàtiques i Informàtica

January 15, 2023

UNIVERSITAT DE BARCELONA

Abstract

Facultat de Matemàtiques i Informàtica

MSc

Sampling Methods for Activation Correlation Graphs to Predict Neural Network Generalization Using Topological Data Analysis

by Otis CARPAY

The performance of a deep neural network (DNN) is dependent on its ability to generalize. This ability is often expressed in the difference in accuracy on a training and test set, or the generalization gap. Recent research has seen the use of topological data analysis to estimate this performance gap without the use of a test set. Here, persistent homology measures are derived from a weighted graph of neuron activation correlations (functional network graph). The resulting persistence diagram is vectorized by a number of statistical summaries and correlated with the generalization gap. However, the computational complexity of persistent homology calculations hinders the application to DNNs with a larger number of activations. Methods are needed to sample these activations without losing predictive power. This work assesses the effect of different sampling strategies on the resulting persistence diagrams and their summaries. These include (non-)stratified random sampling, three methods based on notions of neuron importance similar to those used in pruning, and one using k-means++. In line with previous research some of these strategies provide models for predicting the generalization gap with high accuracy. The investigations provide insight and open up new lines of research into the structure of the functional network activation graph.

Acknowledgements

To Dr. Sergio Escalera and Rubén Ballester, my supervisors, for introducing me to their research, letting me contribute, and guiding me through the process.

To my mother and her partner for giving me a place to work, support, sustenance and, of course, love.

To my brother for giving me advice, love, and an interesting view when looking to my left.

To my aquaintance Irene giving me cane when it needed to be given.

Chapter 1

Introduction

One of the main topics in the theory behind deep neural networks is the mystery of generalization. Traditional frameworks fall apart in light of the performance overparameterized networks. Furthermore, models tend to find structures that generalize outside their training set even when they can easily fit randomly labeled datasets (Zhang et al., 2017). Previous work has found various approaches to explaining this mystery in terms of the training process and the role of stochastic gradient descent (Zhang et al., 2017; Frankle and Carbin, 2019; Liu et al., 2020). Theoretically, this only tells half the story, as the structure of the network itself is not explicitly considered. Practically, information about the training process might not be available. Here, we are interested in the structure of the trained network itself or, more precisely, of its activation patterns, and how it varies with the generalization capabilities of a network.

Topological Data Analysis (TDA) employs topologically motivated methods to characterize the structure or shape of data. It has proven to be a powerful tool to aid in deciphering the mysteries of deep neural networks. Research in this area has focused both on the data and the network itself. In the former category falls research that studies the evolution of the topology of the data as it passes through the layers of a neural network (Goldfarb, 2018; Naitzat, Zhitnikov, and Lim, 2020). Research in the latter category variously characterizes the topological structure of a neural net in terms of its weights (Gabrielsson and Carlsson, 2019; Watanabe and Yamana, 2022; Rieck et al., 2019) and its activation patterns (Gebhart, Schrater, and Hylton, 2019).

This project continues methods introduced by Corneanu et al., 2019 and expanded on by Corneanu et al., 2020 and Ballester et al., 2022 which, counter to approaches that keep the inherent structure of the network, construct a fully connected graph from all nodes in the network. By interpreting the activation values as a random variable dependent on the input data, the correlation between these variables endows the graph with weights that allow one to compute its structure in terms of persistent homology. This weighted graph is termed the functional graph of a neural network. Its structure varies with the extent to which the network generalizes to unseen data. In addition to opening up avenues towards untangling the mysteries of deep learning, it sees practical applications in detecting adversarial attacks (Corneanu et al., 2019), regularization, and predicting how well the network generalizes (Corneanu et al., 2020; Ballester et al., 2022).

Ballester et al., 2022 apply this technique to a dataset of trained neural networks from a NeurIPS competition (Jiang et al., 2020) to predict the generalization gap, the difference between the training and test accuracy, from the trained networks alone. All networks are trained on computer vision tasks. As the size of the functional graphs of these networks prohibits the direct calculation of persistent homology due to computational limitations, they select a small set of neurons and analyze this sample. This sample is vectorized through a combination of statistical and non-statistical descriptors (persistence summaries) and related to the generalization gap through linear regression analysis. The present project is a continuation of this research and extends the analysis to different sampling methods and a larger portion of the dataset of trained networks. The central questions in this effort are the following:

- What is the influence of the various sampling methods on the persistence of the resulting graph?
- To what extent do the persistence summaries predict the generalization gap?
- How do the persistence summaries and their predictive power change with the sampling strategy?

The structure of this report is as follows. Chapter 2 defines the functional graph and provides a brief summary of the topological background. Chapter 3 lays out the persistence summaries, the computational complexity of the methods, and the strategies to avoid them. Chapter 4 discusses the experimental setup and chapter 5 the results. Finally, chapter 6 summarizes the findings and their implications and provides directions for feature research.

Chapter 2

Neural Network Topology

This chapter constitutes a brief overview of the concepts underlying the methods of this paper. A more comprehensive introduction into topological data analysis may be found in (Edelsbrunner and Harer, 2010; Chazal and Michel, 2021).

2.1 Simplicial homology

A k-simplex σ is a subset of $\mathbb{R}^N$ consisting of the convex hull of its $k+1$ vertices, a set of affinely independent points, where $k \leq n$. A 0-simplex is a point, a 1-simplex a line, a 2-simplex a triangle, a 3-simplex a tetrahedron, and so on (see figure 2.1). The simplex τ is a face of σ if its vertices are a subset of those of σ. A simplicial complex K is a finite set of simplices such that if $\sigma \in K$, then its faces are also in K, and if $\sigma, \sigma_0 \in K$, then either $\sigma \cap \sigma_0 = \varnothing$ or $\sigma \cap \sigma_0 \in K$. In other words, it is a collection of simplices and their faces that are either 'glued' together by their faces, meaning they share it, or completely separate. A simplicial complex K may be understood in terms of its underlying space $|K|$, the union of all its simplices. It is generally easier, however, to forgo the geometric realization of a simplicial complex altogether and define them abstractly. An abstract simplicial complex is a finite collection of sets S such that $\alpha \in S$ and $\beta \subseteq \alpha$ implies $\beta \in S$. The members of these sets correspond to the vertices of the ordinary simplices. As a consequence, the second requirement of the ordinary simplicial complex is automatically fulfilled.

The structure of a simplicial complex K, in particular the number of p-dimensional holes, is described by its homology groups $H_p(K)$. These comprise the incontractible p-dimensional cycles that circumscribe the p-dimensional holes. More formally, for an abstract simplicial complex S, let

$$H_p(K) = Z_p(K)/B_p(K) \tag{2.1}$$

Here, $Z_p(K) = \ker \partial_p$ refers to the group of p-cycles, and $B_p(k) = \partial(C_{p+1}(K))$ to the group of p-boundaries (see figure 2.2). Finally,

$$\beta_p = \operatorname{rank}(H_p(K)), \tag{2.2}$$

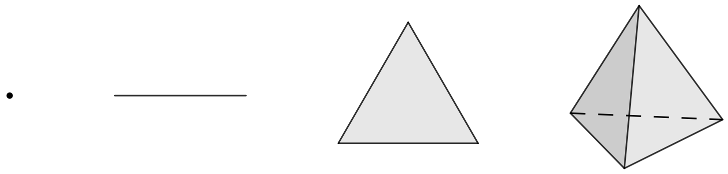


FIGURE 2.1: Simplices of dimension 0 to 3 from left to right.

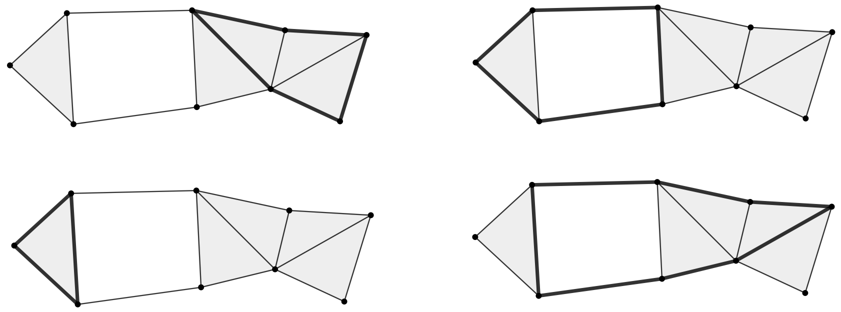


FIGURE 2.2: Four examples of a 2-cycle in a simplicial complex K. The left two examples are 2-boundaries.

the p-th Betti number of K, which one may geometrically interpret as the number of p-dimensional holes in K. The Betti number β_0 is equal to the number of connected components in K, β_1 to the number of 'circular' holes, and β_2 to the number of 'voids'. The complex from figure 2.2, for example, has $\beta_0 = 1$ and $\beta_1 = 1$.

2.2 Persistent homology

Consider the set S with the associated metric δ, which together form a metric space. The diameter of a subset A of this space is the supremum over the distances between its points, or $\mathrm{diam}(A) = \sup_{x,y \in A} \delta(x, y)$. A Vietoris-Rips complex of S at ϵ is an abstract simplicial complex generated on the space S by a maximum distance parameter $\epsilon \geq 0$ so that

$$\mathrm{VR}_\epsilon(S) = \{\alpha \subseteq S \mid \mathrm{diam}(\alpha) \leq \epsilon\}. \tag{2.3}$$

In other words, it is an abstract simplicial complex generated from a collection of points, where each collection of $k + 1$ points at a distance of at most ϵ from each other forms a k-simplex. Subsequently, the homology of the resulting complex may be inspected. Of course, the complex and its homology are dependent on the parameter ϵ and it is unclear which, if any, is the correct choice. Moreover, perturbations in the data may well throw a wrench in the works. Instead, the key insight of persistent homology is to consider *all* complexes as the parameter ϵ varies. The features that persist through the variation of ϵ are considered representative of the data. Let the increasing sequence $\epsilon_0, \ldots, \epsilon_n$ yield a series of Vietoris-Rips complexes such that

$$\varnothing \subset \mathrm{VR}_{\epsilon_0} \subset \mathrm{VR}_{\epsilon_1} \subset \ldots \subset \mathrm{VR}_{\epsilon_n},$$

called a *filtration*. Consider the filtration $K_0 \subset \ldots \subset K_n$ where $0 \leq i \leq j \leq n$ and the inclusion map $\iota : K_i \hookrightarrow K_s$ that sends a simplex $\alpha \in K_i$ to the same simplex as a member of K_j. The map ι induces a homomorphism $f_p^{i,j} : H_p(K_i) \to H_p(K_j)$ on the simplicial homology groups for each dimension p. The p-th persistent homology groups are the images of these groups, and may alternatively be expressed as

$$H_p^{i,j} = Z_p(K_i)/(B_p(K_j) \cap Z_p(K_j)), \tag{2.4}$$

and has as its rank the p-th persistent Betti number $\beta_p^{i,j}$. A class γ of $H_p(K_i)$ is said to be born at K_i if $\gamma \notin H_p^{i-1,i}$ and is said to die entering K_j if it merges with an older

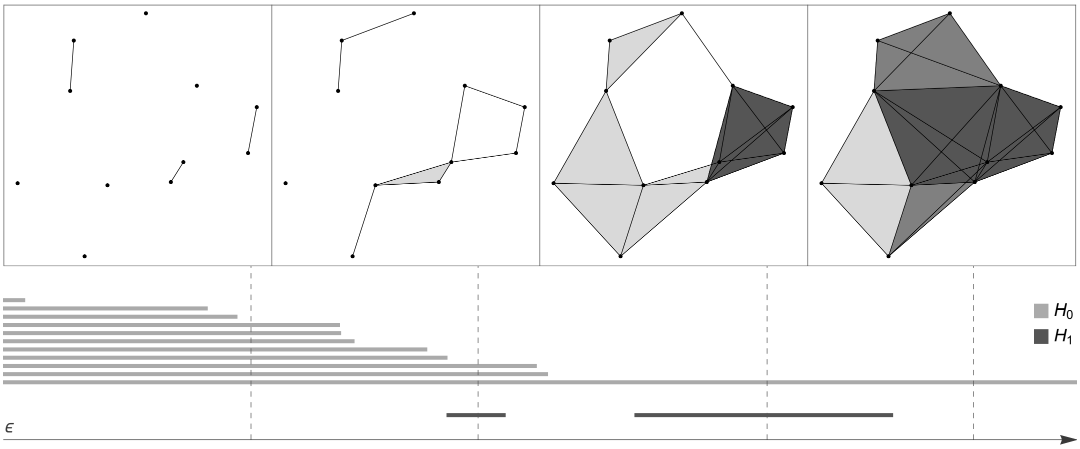


FIGURE 2.3: A Vietoris-Rips filtration (top) visualized in a barcode (bottom). The intervals indicate the life of a persistent homology class and the color the dimension. The dotted lines indicate the parameter value ϵ for the complex above them.

class going from K_{j-1} to K_j. This process may be visualized as a *barcode*, where all persistent homology classes are indicated as intervals along the number line (see figure 2.3), or encoded as a collection of *persistence diagrams*, which each contain the classes of a dimension of birth a and death b as points (a, b). Since classes cannot die before or exactly at the time of birth, all points are above the diagonal.

A Vietoris-Rips filtration $\{\mathrm{VR}_\epsilon\}_{\epsilon \geq 0}$ yields for each dimension $p > 0$ such a persistence diagram, a collection of points (r, s) where $r < s$, for each class that is born at VR_r and dies at VR_s. As may be observed in 2.3, all 0-dimensional classes are born at $\epsilon = 0$ and one never dies. This is because at $\epsilon = 0$ all points are isolated and at $\epsilon = \infty$ all points are fully connected.

2.3 Functional graph

For a trained neural network N, name $\mathcal{D}$ the set of observations x on which it is trained (the input) and V the set of nodes in N with elements v. Let $N_v(x)$ denote the activation of a node $v \in V$ for some input x. We define a vector of activations on a series of inputs $\mathcal{D}$ as

$$A_v(\mathcal{D}) = (N_v(x))_{x \in \mathcal{D}}. \tag{2.5}$$

The outputs of each layer in the neural network, final and intermediate, are contained in the set $A_N(\mathcal{D} = \{A_v(\mathcal{D}) \mid v \in V\}$. Define the function $d : V \times V \mapsto \mathbb{R}$ given by

$$d(v_i, v_j) = 1 - \left| \operatorname{corr}\left(A_{v_i}(\mathcal{D}), A_{v_j}(\mathcal{D})\right) \right|, \tag{2.6}$$

where corr is the Pearson correlation coefficient.

The object of study is the *functional graph* of N, the complete weighted graph with nodes V and weight $d(v_i, v_j)$ on the edge between v_i and v_j. Nodes with constant activation have zero variance and therefore undefined correlation to any other nodes, but may be assumed not to have an effect on the behavior of the model and are therefore excluded. The functional graph captures the behavior of N in a way similar to the neuroscientific adage "neurons that fire together wire together". Nodes with similar activation patterns are interpreted as being “close”, regardless of their separation in the actual network.

The function d does not satisfy the triangle inequality and is consequently not a distance. Although it may be adapted to comply with this axiom (Solo, 2019), this is not required to yield stable Vietoris-Rips filtrations (Ballester et al., 2022) and thus works for the present case as is.

Chapter 3

Methodology

3.1 Persistence summaries

In order to correlate the persistence diagrams of a network with its generalization gap, we summarize them into a fixed-length set of features. Corneanu et al., 2020 defines a pair of statistical summaries that is extended by Ballester et al., 2022. The latter also defines a set of non-statistical summaries, based on theory of persistent homology.

Statistical summaries. Let a persistence diagram for a specific dimension consist of the points (b, d) that describe a hole that is born at b and dies at d. The *life* of a point is defined as $d - b$, or the distance from birth to death, and its *midlife* as $\frac{d+b}{2}$. We record the mean life and midlife, and the square of both quantities. Further, we record the mean birth and death, as well as its squares, and the transformation $1/x + \ln x$. Lastly, we include the standard deviation of the births and the deaths

Non-statistical summaries. Ballester et al., 2022 includes a number of non-statistical summaries used in the literature for vectorizing persistence diagrams: *persistent entropy* (Atienza, Gonzalez-Díaz, and Soriano-Trigueros, 2020), complex polynomial coefficients (Di Fabio and Ferri, 2015), and *persistence pooling vectors* (Bonis et al., 2016). Since experiments by Ballester et al., 2022 show no benefits to their inclusion, these are left out of the main experiments in this thesis. Of course, it is of interest to verify these results, and demonstrate that they remain consistent with the use of other sampling methods. For this reason, some experiments are done to assess their performance in this context.

3.2 Computational complexity

The methods described above apply easily to networks with a limited number of neurons and small training datasets, but see some issues scaling up to less contained scenarios. Consider a dataset $\mathcal{D}$ and a neural network N with nodes V. Then, the set of activation vectors has cardinality $|A_N(\mathcal{D})| = |V|$ with elements in $\mathbb{R}^{|\mathcal{D}|}$. In order to obtain the functional activation graph, one has to determine its weights, which are calculated from the full set of activation vectors. The full set of weights is of cardinality $|V|^2$. Its production depends on the calculation of the Pearson correlation coefficient for every element, which amounts to a complexity of $O(n^2 m)$, where n is the number of nodes, or $|V|$, and m the number of data points, or $|\mathcal{D}|$. Since the dataset analyzed in the present project features networks of 900,000 neurons with a training set of 600,000 points, this issue alone calls for a mitigation strategy.

Much less favorable, however, is the complexity of the computation of the persistence diagrams. The number of k-simplices generated during a Vietoris-Rips filtration on the weighted graph with nodes V is the binomial coefficient $\binom{|V|+1}{k+1}$. Hence, if k is bounded only by $|V|$, the number of simplices correlates exponentially with $|V|$. As in (Ballester et al., 2022), however, we limit the persistent homology dimensions 0 and 1, requiring the generation of simplices up to dimension 2. The algorithm for the calculation of the persistence diagrams depends on Gaussian elimination to find the rank of the boundary matrices, which in practice is of complexity $O(n^3)$, where n is the number of simplices.

Following Ballester et al., 2022, we reduce the data for the computation at two points: the input space and the nodes of the functional graph for which to compute the persistent homology. This project specifically focuses on strategies for the latter.

3.2.1 Sampling the input space

Instead of calculating the activation of the nodes and subsequent correlations for the full dataset $\mathcal{D}$, we consider subsample $\mathcal{D}' \subseteq \mathcal{D}$. We may suppose, by the law of large numbers, that the corr $\left(A_{v_i}(\mathcal{D}'), A_{v_j}(\mathcal{D}')\right)$ converges to corr $\left(A_{v_i}(\mathcal{D}), A_{v_j}(\mathcal{D})\right)$ (where corr is the Pearson correlation coefficient) for an increasing sample size. To keep in line with Ballester et al., 2022, we select a sample size of 2,000.

3.2.2 Sampling the functional graph

The strongest limiting factor in computing the persistence diagrams from the neural networks is the computational complexity of the persistent homology calculation on the Vietoris-Rips filtration of the functional graph. This necessitates a strategy to reduce the number of nodes by more than 99% in the case of many modern neural networks. This potentially dramatically restricts how representative the sample is. The focus of this thesis is the performance of and the differences between such strategies. This allows us to shed a light both on possible avenues for further development, and the characteristics of the functional graph and the resulting persistence diagrams. In line with Ballester et al., 2022, we select a sample size of 3,000 neurons for persistent homology calculation. A point of divergence is that the original paper sees a repeated sampling of the nodes V according to a probability distribution (further explained under maximum activation importance sampling). The resulting persistence summaries are bootstrapped. Instead of this repeated sampling, this thesis opts for a one-off implementation. This saves computational resources and facilitates the comparison with deterministic methods. On the other hand, it potentially limits the predictive power of the single sample and introduces variability in the case of non-deterministic methods. Some sampling methods have been applied multiple times to assess this latter effect.

Many of the methods below are akin to, or immediately inspired by, methods used in neural network pruning. Indeed, the goal of excluding neurons from the full set with minimal loss of predictive power is the same. A significant difference is that pruned networks are generally retrained to recuperate such losses, whereas in the present case, networks are treated as is. Furthermore, methods in network pruning generally do not pursue such a dramatic reduction in the number of neurons. These two factors may hinder the applicability of these methods to the sampling of the functional graph.

Random sampling

Most straightforwardly, the random sampling strategy considers all neurons and selects a random subsample according to a uniform distribution. To find out whether the information on generalization is well distributed among the layers, we also include a stratified version. Here, the number of neurons taken for sampling is divided equally over the included layers. The last layers will have fewer neurons than this quotient. Any remainder is distributed over the other layers.

Filter correlation

Based on the previously observed information sharing between filters in a convolutional neural networks (Han, Mao, and Dally, 2016), this strategy seeks to maximize the information about the network encoded in the random sample by limiting the choice of filters. Inspired by Kumar et al., 2022, we organize the filters by correlating the feature maps within each layer, computing the function $d(f_i, f_j)$ (see eq. 2.6 where f_i is a flattened representation of the feature map over the input sample $\mathcal{D}'$. Then, we select the n maps that are the most dispersed according to the value of d (White, 1991). This is combined with the stratified random sampler so that the number of samples to be selected in a layer is drawn from these n least correlated feature maps. In this project, a value $n = 10$ is chosen.

Importance sampling

Let $\mathcal{D}'$ be an input sample selected from the entire training dataset $\mathcal{D}$ and V the collection of nodes. The following sampling methods assign each neuron an importance score $I_v(\mathcal{D}')$ based on their activation. Then, we let the sample $V' \subseteq V$ be the neurons with the highest score $I_v(\mathcal{D}')$ for an input sample $\mathcal{D}'$. These are selected for the calculation of the persistent homology.

Maximum activation. This strategy is similar to the strategy used in (Ballester et al., 2022), adapted from (Nezhadarya et al., 2020), with one important difference that will be expanded on after its explanation.

Let $\mathcal{D}'$ be an input sample selected from the entire training dataset $\mathcal{D}$ and V the collection of nodes. The maximum activation score of a node v is defined as

$$I_v^M(\mathcal{D}') = \left|\left\{x \in \mathcal{D}' \mid N_v(x) = \max\left\{N_{v_i}(x) \mid v_i \in V\right\}\right\}\right|.$$

In other words, it denotes the number of inputs from $\mathcal{D}'$ for which v has the highest activation. The number of nodes with an importance score $I_v^M(\mathcal{D}') > 0$ may be tiny. Inclusion of only this set excludes the vast majority of the network, the effect of which is unknown, and potentially constrains the samples to severely limit its predictive power. To overcome this obstacle, Ballester et al., 2022 instead specify a probability distribution according to the importance score, where nodes with a nonzero importance score are have weights equal to their importance score, and the weights of the other nodes sum to 1. The 3,000 nodes are sampled without repetition according to this distribution. To better approximate the persistence summary of the entire graph, the researchers employ bootstrapping over the n summaries resulting from n different samples from this distribution.

The one-off version employed in this thesis simply selects all nodes with $I_v^M(\mathcal{D}') > 0$ and supplements these with nodes selected from the remaining nodes according to a uniform distribution up to 3,000. This ensures that all nodes that are important under the assumed notion of importance underlying the method are included to maximize the power of the sampler. The maximum number of nodes of importance $I_v^M(\mathcal{D}') > 0$ is the input sample size $|\mathcal{D}'|$, since every instance from $\mathcal{D}'$ adds 1 to the importance score of the node in V with the highest activation for that sample. Consequently, a neuron sample size of 3,000 and input data sample size of 2,000 allows all nodes with a nonzero importance score to be included.

Mean activation. This score is based on a similar notion of importance and finds its origin as a criterion for pruning neurons in convolutional networks (Molchanov et al., 2017). The mean activation score is defined as

$$I_v^\mu(\mathcal{D}') = \mu\left(|N_v(\mathcal{D}')|\right),$$

or simply the mean of the activations in node v for input sample $\mathcal{D}'$. In contrast with the maximum activation score, more than $|\mathcal{D}'|$ nodes will have a score $I_v^\mu(\mathcal{D}') > 0$. Consequently, the sample can be determined simply by these scores and is deterministic up to the input sample.

Zero activation. The final notion of importance is similarly used in pruning (Hu et al., 2016) and is based on the percentage of zeros in a neuron. We turn the terminology around, however, and let

$$I_v^0 = \sum_{i=1}^{|\mathcal{D}'|} [v_i \neq 0],$$

or simply the number of non-zero values in the activation pattern of the neuron. The underlying assumption is that a zero value has no influence in the subsequent layer, and consequently the neurons with the least zero values are the most representative of the network.

Cluster methods

Ideally, a sampling strategy would anticipate the structure of the activation graph and select the neurons that yield a similar persistence diagram to the full graph. The aforementioned strategies use no such information. A solution is to employ clustering methods. There are a few complications, however.

First, clustering algorithms carry their own issues with respect to resources. The high dimensionality of the points to be clustered (input sample size $|\mathcal{D}'|$) often has a strong negative influence on the efficiency of many clustering algorithms. Furthermore, many algorithms require a distance matrix, especially when the distance is non-standard such as one using the Pearson correlation coefficient. A network of 1M neurons implies a matrix of 10^{12} entries, which either requires an excessive amount of RAM, or a large amount of I/O operations on slower data storage. These issues are not necessarily prohibitive, but a large computational and/or memory overhead for the computation of each persistence diagram may be undesirable.

Second, it is not obvious how the formation of clusters should determine the sample choice. As shown in 5.1, the analysis of the persistence diagrams depends on the presence and nature of clusters in the sampled graphs. It is unclear whether this structure should be maintained, or the analysis should change.

As an initial trial, I used k-means++ algorithm (Arthur and Vassilvitskii, 2007), typically used to obtain a good set of seed centroids for the k-means algorithm, to select a sample of neurons. The method starts from a point randomly selected from the data according to a uniform distribution. Then, it iteratively selects a new point from the remaining points according to a distribution weighted by the distance to the points already chosen.

To mitigate computation costs, the sampler is stratified according to the method described in 3.2.2. Further, the neurons in each layer are partitioned into sequences of a maximum size of 20,000. From each partition, a subsample of corresponding size is then drawn according to the *k*-means++ algorithm.

There are some theoretical limitations to this approach. For example, the algorithm uses Euclidean distance, which cannot be expected to behave similarly to Pearson correlation, certainly not for non-standardized data. However, it may be able to provide some initial insights into the influence of clustering methods.

Chapter 4

Experiments

4.1 Datasets

I investigated the performance of the sampling strategies using the neural networks provided by the NeurIPS competition Predicting Generalization in Deep Learning (Jiang et al., 2020). The entire dataset comprises eight tasks with a set of trained neural networks. Two of these tasks were available to the competitors for the development of their methods, two served to evaluate the contestants for the public leaderboard and four for the private leaderboard. In the original dataset, the tasks are numbered 1 to 9, with no task 3. Here, the same numbering will be used. The performance of the methods will be investigated in depth on the first two tasks. Tasks 4 and 5 will be used to check whether these findings extrapolate to other datasets.

The networks for the first task are VGG-like (Simonyan and Zisserman, 2015) neural networks with 2 or 6 convolutional layers followed by 1 or 2 dense layers. The networks are trained on the CIFAR-10 dataset (Krizhevsky, 2009), which consists of 60,000 32x32 color images in 10 classes. The number of neurons ranges from 262K to 1.3M, in which case the first two layers account for 500K each, owing to the convolutional architecture.

The models in the second task employ a *network in network* architecture (Lin, Chen, and Yan, 2014), with 6, 9, or 12 convolutional layers. They are trained on the SVHN dataset (Netzer et al., 2011), which consists of 600,000 32x32 color images in 10 classes. All networks have between 842K and 872K neurons, where the first six layers account for 115K neurons each.

The networks in both sets vary in number of convolutional layers, dropout probability, weight decay, and batch size. Additionally, the first dataset sees a variation in the number of filters in the last convolutional layer.

Task 4 and 5 both consist of the same set of fully convolutional neural networks trained on a random subset of size 108,000 of the CINIC-10 dataset (Darlow et al., 2018), an extension of the CIFAR-10 dataset with the same input dimensions and the same number of classes. The models in task 4 have been trained with batch normalization and the models in task 5 without. This is the only difference. When batch normalization is applied, it is applied before the ReLU activation function. The last two layers are a convolutional layer without ReLU and a global average pooling layer. Among the networks in each task, the number of layers varies from 7 to 12[1], with the total number of neurons varying between 800K and 1.4M. The networks differ in whether layers contain the most filters close or far from the input. Hence, the distribution of neurons over the layers strongly varies between the networks. The networks further vary in number of parameters, weight decay, learning rate, and batch size.

[1] In tasks 1 and 2, the ReLU functions are applied within a Keras layer. In tasks 4 and 5, they are encoded as separate layers. In this project, the separate activation function layers are treated as if

4.2 Experimental procedure

After the generation of the persistence diagrams for these neural networks, they are summarized into a vector by the descriptors discussed in 3.1. We train a simple linear model for all 255 combinations of the eight summaries, each for 1,000 random train/test splits of 70%/30%. The splits are the same for each combination. This yields a distribution of R^2-scores (the coefficient of determination) to investigate. We particularly focus on the performance of the models including all features, and the models for the combination with the highest mean R^2-score, termed the best selection.

4.2.1 Coefficient of determination

Let y_i be the true value and $\hat{y}_i$ the predicted value of the i-th sample. The R^2 score is defined as

$$R^2(y, \hat{y}) = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2} \tag{4.1}$$

where $\bar{y}$ is the mean value of y. Note that the dividend in the quotient is the residual sum of squares and the divisor the total sum of squares. As the quotient is between two positive numbers, the upper bound of R^2 is 1. For the lower bound, consider the example $y = (a, a+2)$ and $\hat{y} = (0, 0)$. The equation reduces to

$$R^2(y, \hat{y}) = 1 - \frac{a^2 + (a+2)^2}{2}. \tag{4.2}$$

As a tends to infinity, so does the quotient. Hence, there is no lower bound to R^2.

they form part of the previous layer. Consequently, the preceding layers are also excluded from the functional graph.

Chapter 5

Results

5.1 Model performance

5.1.1 Task 1

Table 5.1 shows the mean and median of the distribution of R^2 for the used samplers. The distributions are visualized in figure 5.1. The k-means++ sampler shows the best performance for both measures and both selections (complete and best), save for the highest median for the random sampler on all summaries. Still, its difference with the random sampler for the best selection is small and possibly insignificant. Stratification seems to impair the performance of the random sampler slightly. This is surprising, since the distribution of neurons over the layers is highly disproportionate, as mentioned in 4.1. In absence of stratification, the vast majority of the neurons are sampled from the early layers. The last layers contribute none to very few. This suggests that the extent of correlation across the whole network is not necessarily indicative of the ability to generalize, or that this information is not encoded in the persistence diagram of the sample.

It can be observed that some medians diverge strongly from the mean, signaling a skewed distribution or outliers with a low R^2. Most likely, this is due to the presence of uninformative features in the full selection, the absence of regularization in the linear regression, and the fact that there is an upper but no lower bound to R^2. For some partitions of the data into test and training sets, unfortunate subsets of the data may cause an inaccurate fit that extrapolates to an anti-correlated test set. Indeed, the distributions for the best selection show no such issue. In principle, these only contain the features that correlate with the generalization gap. Moreover, the higher-scoring samplers yield much narrower distributions. This hints at better cooperation between the features and less variability.

TABLE 5.1: Mean R^2 for task 1 for all features and the best selection.

	All features		**Best selection**	
Sampler	**Mean**	**Median**	**Mean**	**Median**
Random	0.771	**0.809**	0.816	0.825
Random (s)	0.747	0.773	0.797	0.808
Importance (mean)	0.607	0.636	0.719	0.726
Importance (max)	0.724	0.758	0.789	0.799
Importance (zero)	0.360	0.387	0.463	0.472
K-means++	**0.781**	0.796	**0.825**	**0.830**
Filter correlation	0.757	0.768	0.794	0.804

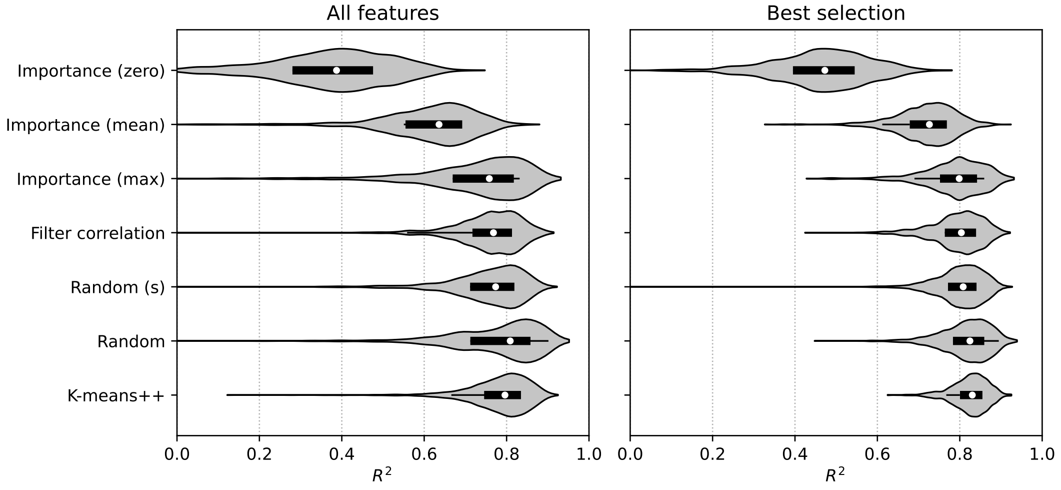


FIGURE 5.1: Violin plots of the distribution of R^2 for the first task (for 1000 train/test splits), ordered by the median score for the best selection

TABLE 5.2: Mean and median R^2 for different repetitions of the random sampler in task 1 for all features and the best selection.

	All features		**Best selection**	
Repetition	**Mean**	**Median**	**Mean**	**Median**
1	0.771	0.809	0.816	0.825
2	**0.821**	**0.842**	**0.848**	**0.865**
3	0.757	0.789	0.814	0.825
4	0.716	0.738	0.796	0.808

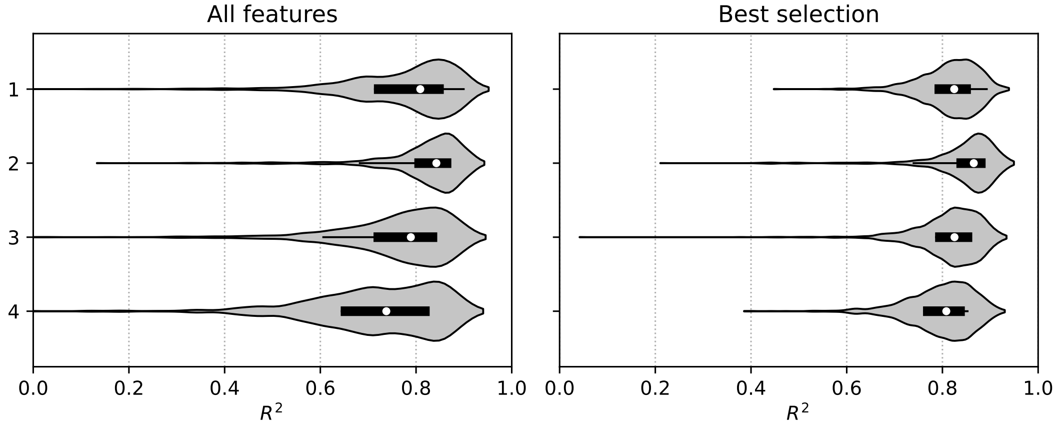


FIGURE 5.2: Violin plot of the distribution R^2 for different repetitions of the random sampler in task 1

TABLE 5.3: Mean R^2 for task 2 for all features and the best selection.

	All features		Best selection	
Sampler	**Mean**	**Median**	**Mean**	**Median**
Random	0.305	0.735	0.793	0.804
Random (s)	0.678	0.760	0.802	0.833
Importance (mean)	0.805	0.883	**0.929**	**0.934**
Importance (max)	**0.856**	**0.893**	0.906	0.921
Importance (zero)	0.397	0.501	0.567	0.633
K-means++	0.762	0.824	0.851	0.861
Filter correlation	0.674	0.737	0.844	0.852

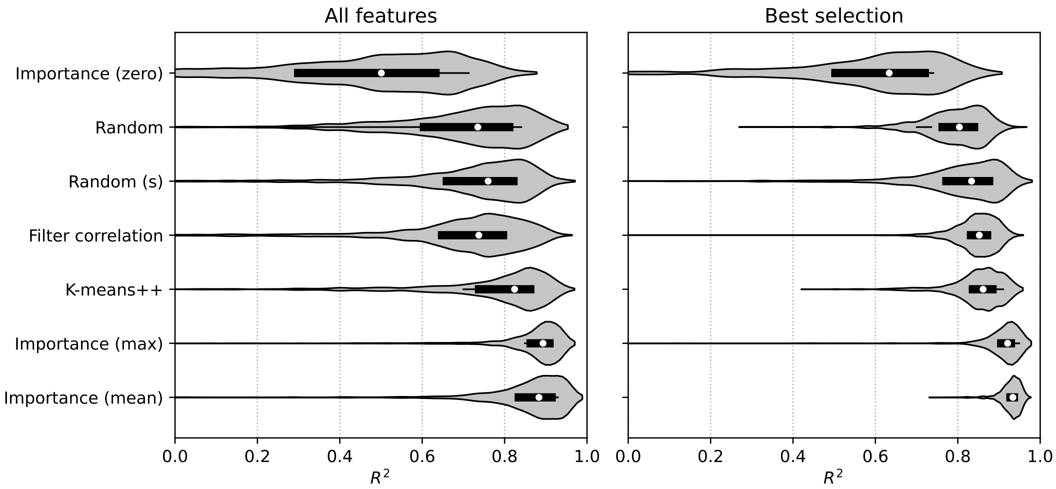


FIGURE 5.3: Violin plot of the distribution of R^2 for the second task.

One question that may arise is the amount of variation between different runs of the same sampler. After all, there are multiple points of non-determinism: the input space is randomly sampled, and some of the neuron samplers are nondeterministic too. Ballester et al., 2022 leverage this fact to introduce a bootstrapping step, as discussed in 3.2.2. Table 5.2 and figure 5.2 show the results of various runs of the random sampler, each with a different input data sample. While there is some variation, each shows good performance. We may conclude, perhaps surprisingly, that a completely random selection of neurons from the full network consistently yields a representative persistence diagram (for this specific problem). At the same time, it places into context the scores for the other samplers. Combined with the variation of the random sampler results, we cannot make conclusive statements on whether any sampler performs the best.

5.1.2 Task 2

The results for task 2, listed in table 5.3 and visualized in figure 5.3, show a reversal of the ranking of the samplers with respect to task 1. The two importance samplers based on the quantitative activation of the neurons have a distinct advantage over

TABLE 5.4: Mean and median R^2 for different repetitions of the mean importance sampler in task 1 for all features and the best selection.

	All features		Best selection	
Repetition	**Mean**	**Median**	**Mean**	**Median**
1	0.805	0.883	0.929	0.934
2	0.769	0.893	0.926	0.941
3	0.849	0.935	0.941	0.944

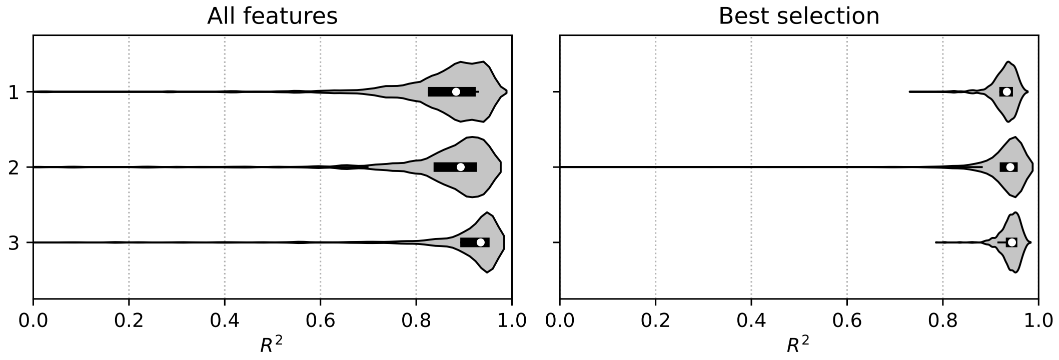


FIGURE 5.4: Violin plot of the distribution of R^2 for different repetitions of the mean importance sampler in task1

the other samplers and yield a high accuracy on the best feature selection. The variance of the mean importance sampler in the latter case is especially low. In contrast with task 1, stratification does seem to have a positive effect on the predictive power of the random sampler. The strategy based on the number of zeros continues to yield mediocre scores. All samplers except the random sampler perform better on task 2 than on task 1. We may also observe that the max importance strategy is particularly insensitive to extraneous dimensions compared to the others.

Table 5.4 and figure 5.4 show the results of repeated evaluation of the mean importance sampler, the best-performing sampler for the second task. In contrast to the random sampler, the only source of non-determinism is in the sampling of the input data. The linear regression over the various repetitions exhibits little variability. These results corroborate that 2,000 is a representative sample size. Of course, this may vary across network architecture and input datasets. For example, a larger number of classes may necessitate a larger input sample size $|\mathcal{D}'|$.

5.1.3 Selected features

Of course, we are interested in knowing which features are included in the best selection. Table 5.5 shows these data for both tasks. However, before interpreting the table, the data should be placed in context. As explained in 4.2, the best selection is the combination of features (out of all 255 combinations), with the highest mean R^2 over 1,000 experiments with different data partitions. The differences in performance between these combinations may be so small that the nondeterministic factors in the experiments have a large influence on their inclusion.

TABLE 5.5: Table showing which features are included in the best selection for each sampler and each task. The entries in the table designate for which task the specific feature is included for the specific model.

Feature	**R**	**R (s)**	**I (μ)**	**I (max)**	**I (0)**	**K++**	**F**
Mean birth and death	1			1			
Mean birth and death (squared)		1	1 2	2	1 2	1	1
Mean birth and death (log)	1	1 2		1 2	1 2	1	1 2
Std birth and death	1 2	1 2	1	1 2	1 2	1 2	1 2
Mean life	1	1 2	1 2	1		2	1
Mean life (squared)	2	2		1	2	2	
Mean midlife	2	2	1	2	2	2	
Mean midlife (squared)	1	1 2		1		1	1

TABLE 5.6: Mean R^2 for task 4 for random and mean activation importance samplers.

	All features		**Best selection**	
Sampler	**Mean**	**Median**	**Mean**	**Median**
Random	**0.779**	**0.801**	**0.826**	**0.840**
Random (s)	0.678	0.731	0.737	0.764
Importance (mean)	0.486	0.537	0.524	0.570
Importance (max)	0.576	0.656	0.722	0.745
Importance (zero)	-15.308	-0.692	0.094	0.132
K-means++	-0.111	0.193	0.294	0.317
Filter correlation	0.563	0.684	0.720	0.752

That said, a general idea about the role of various features may be formed from the table. The standard deviation of the mean of the births and deaths is included in every model except for the mean activation importance strategy on task 2. Mean birth and death, on the other hand, see the least inclusion. More light is shed on these points in the next section.

5.2 Task 4 & 5

It is of interest to know to which extent these results generalize to other networks. Although the tasks share architectures and only differ in the application of batch normalization, the results are quite different. The results for task 4 are shown in table 5.6 and figure 5.11. One may immediately notice a greater variance in the performance of the different sampling strategies than first two tasks. While the random sampler yields a similar distribution, all other samplers have lower R^2 scores (that is, the distribution is concentrated lower on the number line). Especially the mean and zero importance and k-means strategies see a strong drop in comparison with the first two tasks.

Table 5.7 and figure 5.6 show that the results for task 5 deviate even more. This

TABLE 5.7: Mean R^2 for task 5 for random and mean activation importance samplers.

	All features		**Best selection**	
Sampler	**Mean**	**Median**	**Mean**	**Median**
Random	-4.161	-0.278	-0.254	-0.040
Random (s)	-0.988	-0.057	0.125	0.503
Importance (mean)	**0.033**	0.388	**0.673**	**0.829**
Importance (max)	-1.867	-0.757	-0.250	0.334
Importance (zero)	-17.923	-1.001	-0.163	-0.055
K-means++	-2.250	-0.082	-0.307	0.406
Filter correlation	-0.706	**0.456**	0.194	0.664

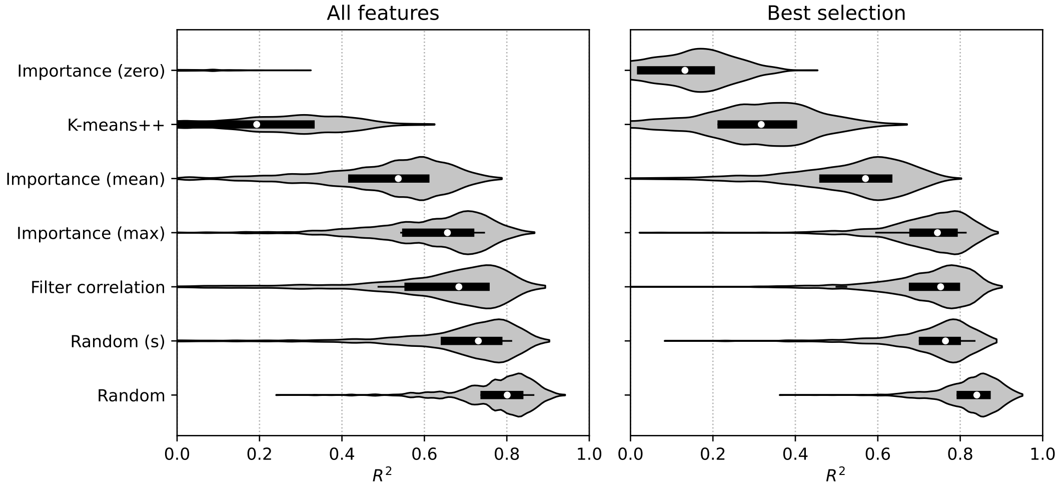


FIGURE 5.5: Violin plots of the distribution of R^2 for task 4, ordered by the median score for the best selection

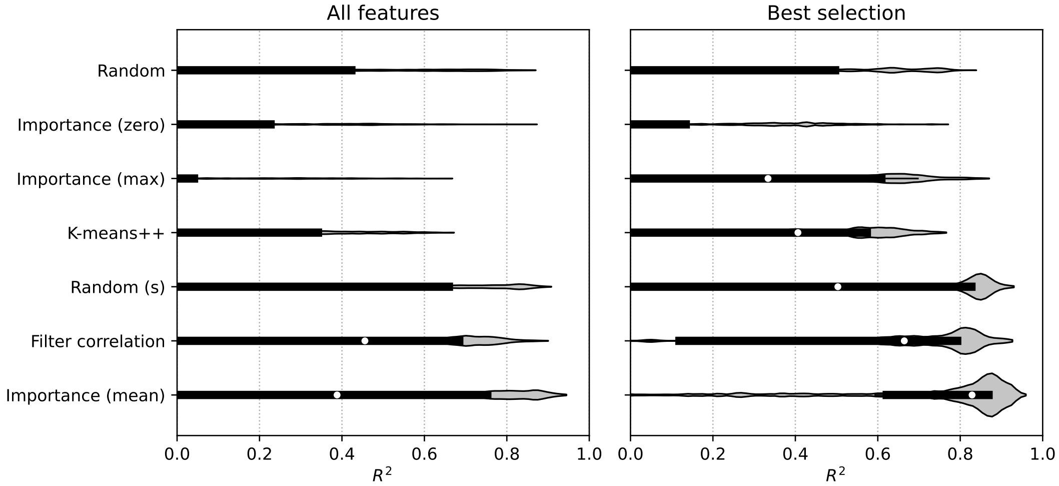


FIGURE 5.6: Violin plots of the distribution of R^2 for task 5, ordered by the median score for the best selection

behavior finds a practical cause in the distribution of the generalization gaps however. As shown in figure 5.10, the gaps are densely distributed around a value of 0.4 but have four outliers from 0.52 up to 0.74. Consequently, the R^2 scores are largely determined by these points, and specifically their division over the training and the validation set. Fortunate splits may associate with high R^2 because the model fits the outliers well. Conversely, unfortunate splits may yield large negative values. Therefore, these scores should be taken with a grain of salt. In case of the low negative mean R^2 values in task 4 and 5, one should keep in mind that R^2 has an upper bound of 1 but no lower bound (see 4.2.1.

A closer look into the generation of the persistence diagrams shows that the use of batch normalization in task 5 has a specific influence on the average importance sampling strategy. Batch normalization is only applied before the ReLU activation function. The last convolutional layer (preceding the final global average pooling layer), then, is not normalized. I found that this layer sees the highest neuron activation among all layers in the model. The consequence is that in practice, every single neuron in the sample is drawn from this layer. Mitigation of this behavior, for example by a stratified sampling strategy, could yield higher scores.

The sampling strategies with the highest average (over the four tasks) median R^2 for the best selection are the filter correlation (0.77) and the mean importance (0.76) strategies.

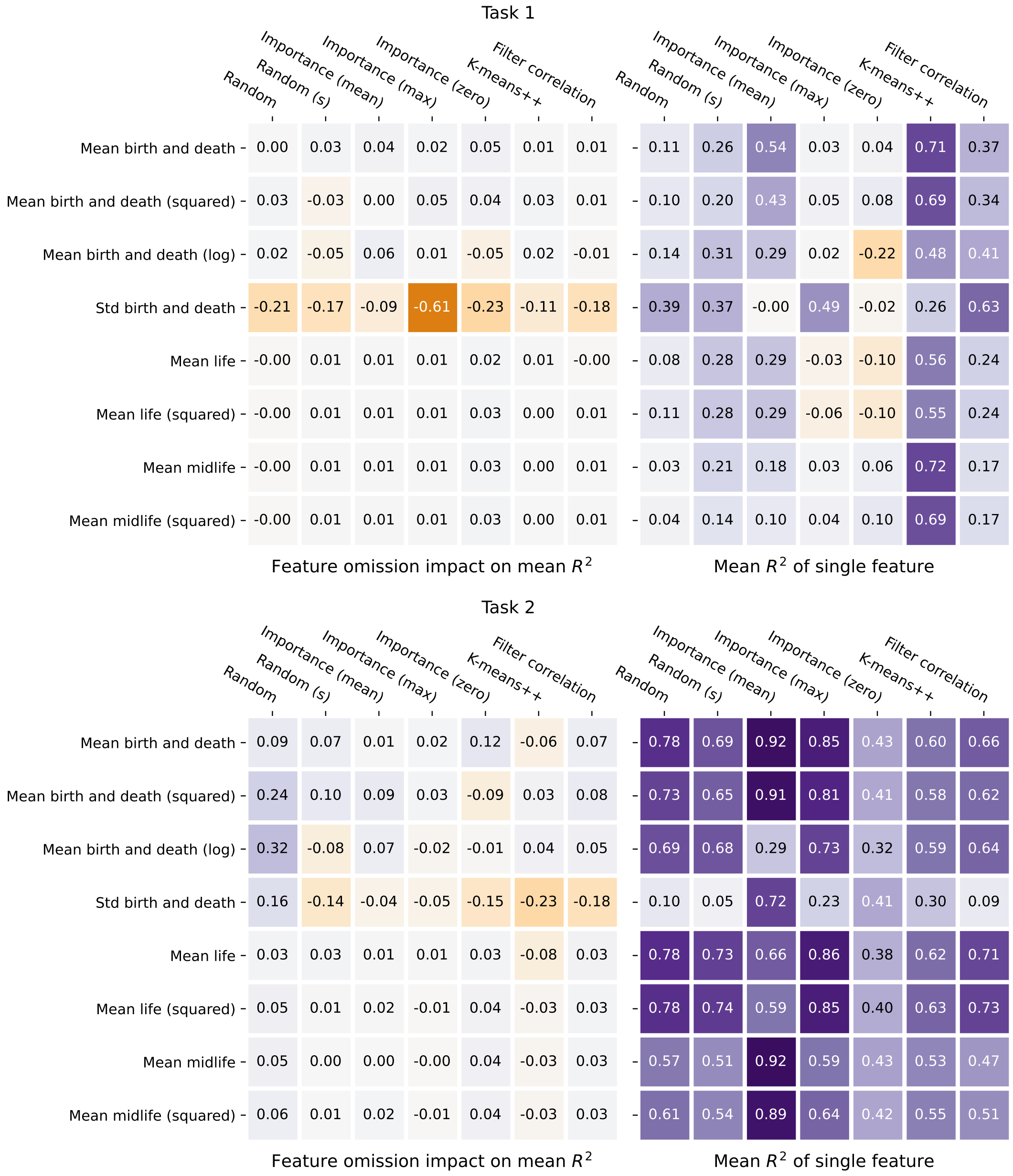
5.3 Single feature performance

Some summaries correlate strongly with the generalization gap, whereas others show little to no correlation. Moreover, this effect differs across samplers. Figure 5.7 demonstrates the effect of excluding summaries from the full selection and the performance of single summaries for task 1 and task 2.

In task 1, the mean R^2 scores for single features show a relatively high variance across samplers. Some features with strong scores in one sampler may have low or even negative scores in others. One pattern that stands out is the influence of the standard deviation of births and deaths. Its inclusion both in the full selection and as a sole summary tends to have a strong positive influence on R^2. At the same time, it shows the lowest correlation as a single feature for the k-means++ sampler. This strategy, in contrast, yields high scores for the other single features. This is slightly different from the random sampler, which scores similarly in general. Here, the summaries have less predictive power individually.

The scores for task 2 paint a different picture. Here, models based on individual summaries show high performance, in some cases even close to that of the best selection, *except* for the standard deviation of births and deaths. Even so, table 5.5 shows that often these are included in the best selection. This is probably because they correlate well in conjunction, and tend to encode information not found in other summaries. This latter point implies that they don't tend to add noise when combined and consequently have a positive influence on R^2 in most situations.

The average positions of points in the persistence diagram demonstrate high predictive power in task 2. Particularly the scores of the top-scoring regressions, those drawn from the mean activation importance strategy, are approximated by single features. At the same, their omission from the full selection carries few consequences. The reason for this is that they are strongly related and practically mutually replaceable, as shown in 5.4.



Task 1

	Random	Random (s)	Importance (mean)	Importance (max)	Importance (zero)	K-means++	Filter correlation
Mean birth and death	0.00	0.03	0.04	0.02	0.05	0.01	0.01
Mean birth and death (squared)	0.03	-0.03	0.00	0.05	0.04	0.03	0.01
Mean birth and death (log)	0.02	-0.05	0.06	0.01	-0.05	0.02	-0.01
Std birth and death	-0.21	-0.17	-0.09	-0.61	-0.23	-0.11	-0.18
Mean life	-0.00	0.01	0.01	0.01	0.02	0.01	-0.00
Mean life (squared)	-0.00	0.01	0.01	0.01	0.03	0.00	0.01
Mean midlife	-0.00	0.01	0.01	0.01	0.03	0.00	0.01
Mean midlife (squared)	-0.00	0.01	0.01	0.01	0.03	0.00	0.01

Feature omission impact on mean R^2

	Random	Random (s)	Importance (mean)	Importance (max)	Importance (zero)	K-means++	Filter correlation
Mean birth and death	0.11	0.26	0.54	0.03	0.04	0.71	0.37
Mean birth and death (squared)	0.10	0.20	0.43	0.05	0.08	0.69	0.34
Mean birth and death (log)	0.14	0.31	0.29	0.02	-0.22	0.48	0.41
Std birth and death	0.39	0.37	-0.00	0.49	-0.02	0.26	0.63
Mean life	0.08	0.28	0.29	-0.03	-0.10	0.56	0.24
Mean life (squared)	0.11	0.28	0.29	-0.06	-0.10	0.55	0.24
Mean midlife	0.03	0.21	0.18	0.03	0.06	0.72	0.17
Mean midlife (squared)	0.04	0.14	0.10	0.04	0.10	0.69	0.17

Mean R^2 of single feature

Task 2

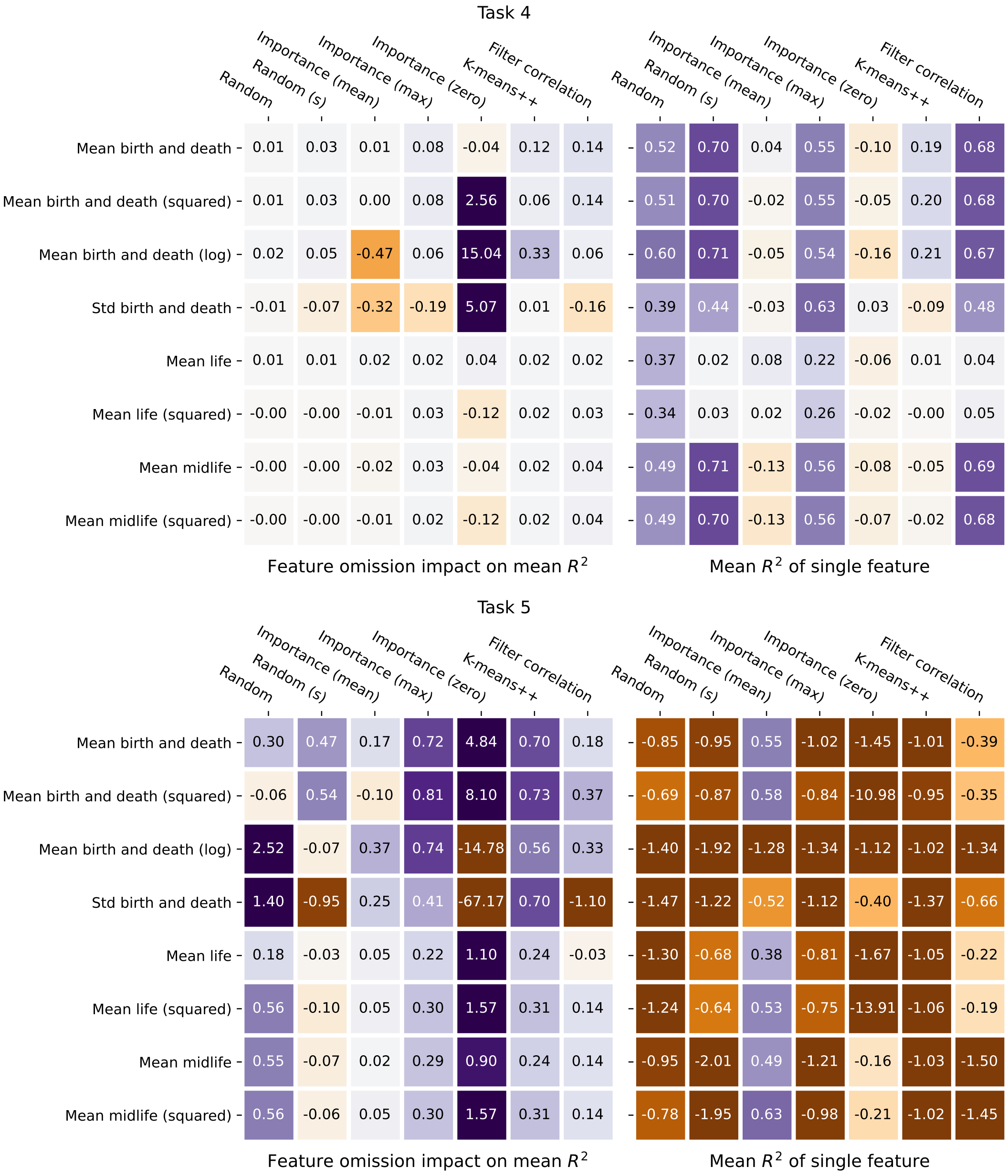
	Random	Random (s)	Importance (mean)	Importance (max)	Importance (zero)	K-means++	Filter correlation
Mean birth and death	0.09	0.07	0.01	0.02	0.12	-0.06	0.07
Mean birth and death (squared)	0.24	0.10	0.09	0.03	-0.09	0.03	0.08
Mean birth and death (log)	0.32	-0.08	0.07	-0.02	-0.01	0.04	0.05
Std birth and death	0.16	-0.14	-0.04	-0.05	-0.15	-0.23	-0.18
Mean life	0.03	0.03	0.01	0.01	0.03	-0.08	0.03
Mean life (squared)	0.05	0.01	0.02	-0.01	0.04	-0.03	0.03
Mean midlife	0.05	0.00	0.00	-0.00	0.04	-0.03	0.03
Mean midlife (squared)	0.06	0.01	0.02	-0.01	0.04	-0.03	0.03

Feature omission impact on mean R^2

	Random	Random (s)	Importance (mean)	Importance (max)	Importance (zero)	K-means++	Filter correlation
Mean birth and death	0.78	0.69	0.92	0.85	0.43	0.60	0.66
Mean birth and death (squared)	0.73	0.65	0.91	0.81	0.41	0.58	0.62
Mean birth and death (log)	0.69	0.68	0.29	0.73	0.32	0.59	0.64
Std birth and death	0.10	0.05	0.72	0.23	0.41	0.30	0.09
Mean life	0.78	0.73	0.66	0.86	0.38	0.62	0.71
Mean life (squared)	0.78	0.74	0.59	0.85	0.40	0.63	0.73
Mean midlife	0.57	0.51	0.92	0.59	0.43	0.53	0.47
Mean midlife (squared)	0.61	0.54	0.89	0.64	0.42	0.55	0.51

Mean R^2 of single feature

FIGURE 5.7: Heat map of the R^2 scores relative to the various features for tasks 1 and 2. *Feature omission impact* refers to the impact on the mean R^2 of exclusion of the feature from the full selection. All scores are for both dimensions.



Task 4

Feature omission impact on mean R^2

	Random	Random (s)	Importance (mean)	Importance (max)	Importance (zero)	K-means++	Filter correlation
Mean birth and death	0.01	0.03	0.01	0.08	-0.04	0.12	0.14
Mean birth and death (squared)	0.01	0.03	0.00	0.08	2.56	0.06	0.14
Mean birth and death (log)	0.02	0.05	-0.47	0.06	15.04	0.33	0.06
Std birth and death	-0.01	-0.07	-0.32	-0.19	5.07	0.01	-0.16
Mean life	0.01	0.01	0.02	0.02	0.04	0.02	0.02
Mean life (squared)	-0.00	-0.00	-0.01	0.03	-0.12	0.02	0.03
Mean midlife	-0.00	-0.00	-0.02	0.03	-0.04	0.02	0.04
Mean midlife (squared)	-0.00	-0.00	-0.01	0.02	-0.12	0.02	0.04

Mean R^2 of single feature

	Random	Random (s)	Importance (mean)	Importance (max)	Importance (zero)	K-means++	Filter correlation
Mean birth and death	0.52	0.70	0.04	0.55	-0.10	0.19	0.68
Mean birth and death (squared)	0.51	0.70	-0.02	0.55	-0.05	0.20	0.68
Mean birth and death (log)	0.60	0.71	-0.05	0.54	-0.16	0.21	0.67
Std birth and death	0.39	0.44	-0.03	0.63	0.03	-0.09	0.48
Mean life	0.37	0.02	0.08	0.22	-0.06	0.01	0.04
Mean life (squared)	0.34	0.03	0.02	0.26	-0.02	-0.00	0.05
Mean midlife	0.49	0.71	-0.13	0.56	-0.08	-0.05	0.69
Mean midlife (squared)	0.49	0.70	-0.13	0.56	-0.07	-0.02	0.68

Task 5

Feature omission impact on mean R^2

	Random	Random (s)	Importance (mean)	Importance (max)	Importance (zero)	K-means++	Filter correlation
Mean birth and death	0.30	0.47	0.17	0.72	4.84	0.70	0.18
Mean birth and death (squared)	-0.06	0.54	-0.10	0.81	8.10	0.73	0.37
Mean birth and death (log)	2.52	-0.07	0.37	0.74	-14.78	0.56	0.33
Std birth and death	1.40	-0.95	0.25	0.41	-67.17	0.70	-1.10
Mean life	0.18	-0.03	0.05	0.22	1.10	0.24	-0.03
Mean life (squared)	0.56	-0.10	0.05	0.30	1.57	0.31	0.14
Mean midlife	0.55	-0.07	0.02	0.29	0.90	0.24	0.14
Mean midlife (squared)	0.56	-0.06	0.05	0.30	1.57	0.31	0.14

Mean R^2 of single feature

	Random	Random (s)	Importance (mean)	Importance (max)	Importance (zero)	K-means++	Filter correlation
Mean birth and death	-0.85	-0.95	0.55	-1.02	-1.45	-1.01	-0.39
Mean birth and death (squared)	-0.69	-0.87	0.58	-0.84	-10.98	-0.95	-0.35
Mean birth and death (log)	-1.40	-1.92	-1.28	-1.34	-1.12	-1.02	-1.34
Std birth and death	-1.47	-1.22	-0.52	-1.12	-0.40	-1.37	-0.66
Mean life	-1.30	-0.68	0.38	-0.81	-1.67	-1.05	-0.22
Mean life (squared)	-1.24	-0.64	0.53	-0.75	-13.91	-1.06	-0.19
Mean midlife	-0.95	-2.01	0.49	-1.21	-0.16	-1.03	-1.50
Mean midlife (squared)	-0.78	-1.95	0.63	-0.98	-0.21	-1.02	-1.45

FIGURE 5.8: Heat map of the R^2 scores relative to the various features for tasks 4 and 5. *Feature omission impact* refers to the impact on the mean R^2 of exclusion of the feature from the full selection. All scores are for both dimensions.

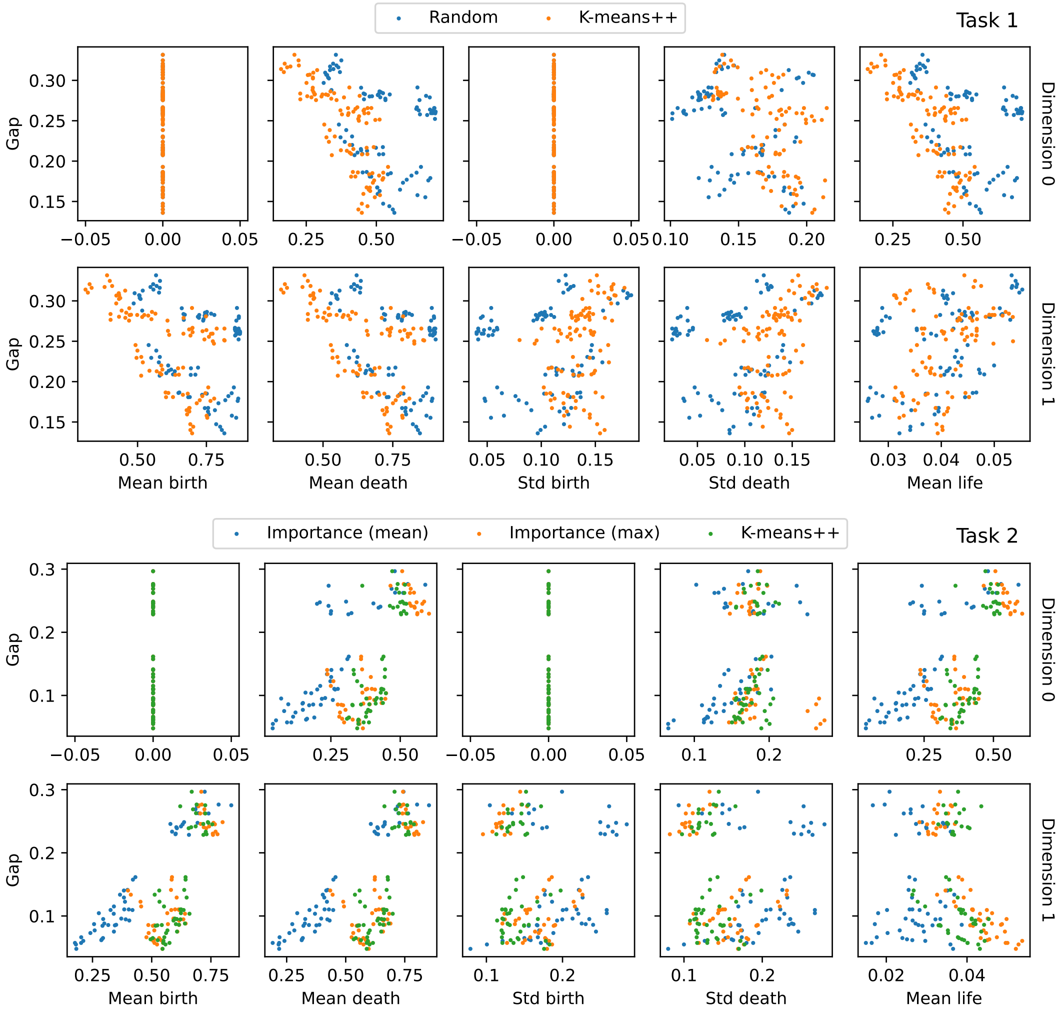


FIGURE 5.9: Scatter plot showing the correlation in tasks 1 and 2 of mean birth and death, std birth and death, and mean life values with the generalization gap for the random, k-means++, and two importance samplers. Homology groups of dimension 0 are always born at the smallest distance parameter ϵ.

5.4 Correlations

Further insight into the nature of the correlation of these summaries with the generalization gap is given by figure 5.9. These findings corroborate those from Ballester et al., 2022. Single-dimensional summaries, depending on the sampling strategy used, may strongly correlate with the generalization gap. The nature of this correlation is not consistent across architectures, however. In task 1, diagrams with high mean births and deaths imply a model that generalizes better, whereas this relationship is reversed in task 2. These relationships seem to be consistent across dimensions. For task 2, however, the correlation is highest in dimension 1 and is responsible for the high performance of the k-means++ sampling strategy. These relationships are, if they are present, consistent across sampling strategies. The standard deviations of the births and deaths show a less pronounced relationship. There is a slight correlation, however, opposite to that of the mean values for both tasks. This could be a result of the behavior of the function d (see 2.6 in this context, where

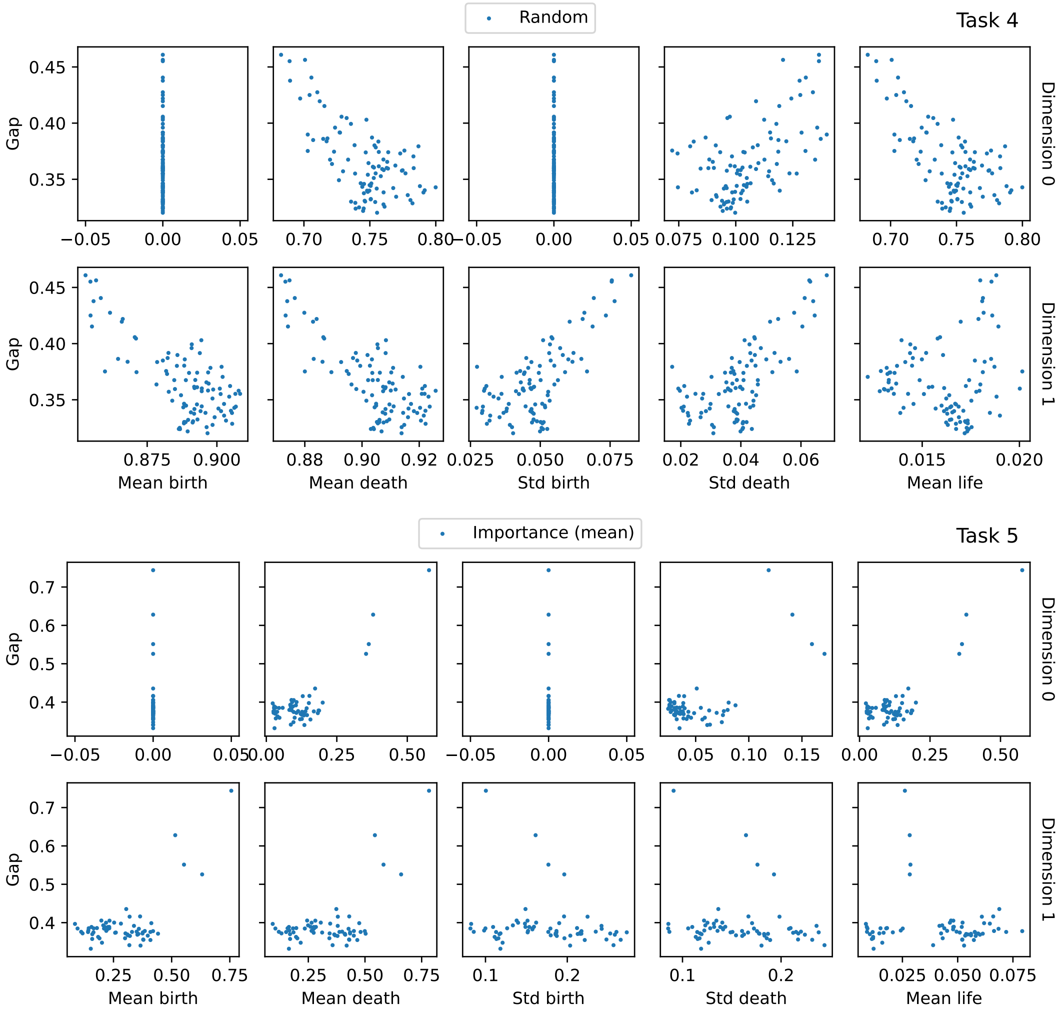


FIGURE 5.10: Scatter plot showing the correlation in tasks 4 and 5 of mean birth and death, std birth and death, and mean life values with the generalization gap for the best performing samplers. Homology groups of dimension 0 are always born at the smallest distance parameter ϵ.

larger values see more variance.

The interpretation of 0-dimensional homology is fairly straightforward. In this dimension, the number of homology groups signifies the number of connected components. Consequently, a high death corresponds to isolated points, or neurons with low correlation to the others. In task 1, then, high correlation between neurons implies high generalization. In task 2, in contrast, high correlation implies low generalization.

The interpretation of 1-dimensional homology is not as simple. Homology groups of this dimension correspond to holes that can be circumscribed by non-contractible rings. This might be translated to the context of functional graphs as closed chains of neurons where neurons correlate well with some but not other neurons.

Figure 5.10 shows that task 4 exhibits patterns similar to but stronger than task 1. In particular, the standard deviation of the births and deaths in dimension 1 are strongly positively correlated with the generalization gap. The mean values are similarly correlated to task 1.

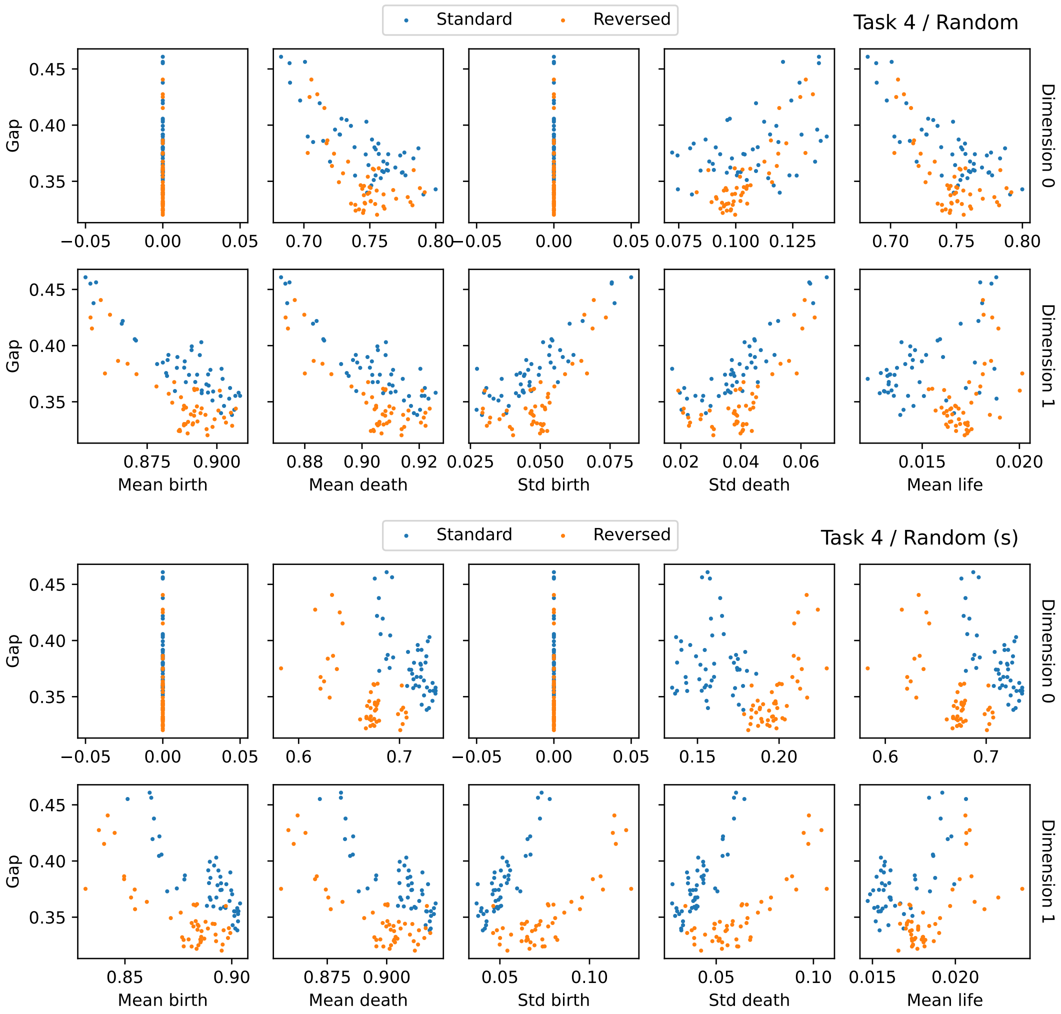


FIGURE 5.11: Scatter plot showing the correlation in tasks 4 of mean birth and death, std birth and death, and mean life values with the generalization gap for the random and stratified random samplers. The color denotes whether layers closer (standard) or farther (reversed) from the input have more filters.

Task 5, on the other hand, shows correlations of the same sign as those in task 2. However, this relationship is mostly determined by the outliers and does not seem to be reflected in the models excluding the outliers. In fact, for standard deviation of births and deaths, the cluster of outliers has an opposite correlation to that of the dataset as a whole. It is possible that, as is observed by Ballester et al., 2022 for task 2, the patterns are organized into clusters. Of course, a sample size of four is too small to draw any conclusions.

Task 4 has another point of interest. Given the generalization gap, the stratified random sampling strategy turns out to be able to separate the networks by whether the number of filters increases or decreases throughout the layers. See 5.11. Moreover, the intercept of the correlation with the standard deviation of births and deaths differs between the two sets. If the two different types of networks would evaluated be separately, the R^2 for this strategy would be higher. This kind of separability is seen less with non-stratified random sampling.

TABLE 5.8: Median R^2 for the various non-statistical summaries. *PE* is the persistence entropy, *CP* the complex polynomial coefficients, and *PP* the persistence pooling vectors. All summaries are in both dimensions 0 and 1.

		Task 1			Task 2	
Sampler	**PE**	**CP**	**PP**	**PE**	**CP**	**PP**
Random	< 0	0.227	< 0	0.305	0.746	< 0
Random (s)	< 0	0.430	< 0	0.237	0.781	< 0
Importance (mean)	0.497	0.309	< 0	0.165	0.468	< 0
Importance (max)	< 0	0.098	< 0	0.372	0.765	0.023
Importance (zero)	0.419	0.001	< 0	0.146	0.323	0.240
K-means++	0.378	0.597	< 0	0.216	0.801	0.507
Filter correlation	0.052	0.394	< 0	0.669	0.786	< 0

			Task 4		Task 5	
Sampler	**PE**	**CP**	**PP**	**PE**	**CP**	**PP**
Random	0.306	0.519	0.519	< 0	< 0	< 0
Random (s)	< 0	< 0	0.195	< 0	0.677	< 0
Importance (mean)	< 0	< 0	< 0	< 0	< 0	< 0
Importance (max)	< 0	0.047	0.649	0.062	< 0	< 0
Importance (zero)	< 0	< 0	< 0	< 0	< 0	< 0
K-means++	< 0	0.163	< 0	< 0	< 0	< 0
Filter correlation	< 0	< 0	0.069	< 0	0.298	< 0

5.5 Other summaries

Finally, we take a short look at the performance of the non-statistical summaries. As discussed in 3.1, these summaries showed the lowest R^2 scores overall in (Ballester et al., 2022) and for this reason have been excluded from the more rigorous analyses above. Table 5.8 paints a different picture, however. All three summaries have relatively high R^2 for at least some tasks and sampling strategies. In particular, the complex polynomial coefficients yield scores close to the best selection of the statistical summaries for all strategies but those based on zero importance score and *k*-means++. Moreover, the complex polynomial coefficients for the stratified random strategy in task 5 scores higher than any combination of statistical summaries.

The patterns are difficult to explain. At the very least, this is an indication that these summaries should be revisited.

Chapter 6

Conclusion

6.1 Summary

The findings of this project corroborate and extend those reported by Ballester et al., 2022 that the persistence summaries of a sampled functional graph can predict its generalization capabilities. Statistical descriptors of average position and dispersion have all been shown to correlate with the generalization gap. Whether they correlate and the extent varies strongly for each descriptor, however, by the type of sampling strategy used and the network architecture. No strategy shows consistent results across architectures without being considerably outperformed by another strategy for some tasks.

6.2 Limitations

The need for sampling remains a limitation in predicting the generalization gap from the persistent homology of the functional graph. While this project has done the preliminary work in separating the wheat from the chaff, none of the strategies have shown consistent ability to yield persistence summaries that strongly correlate with the test performance. On the other hand, in conjunction they have shown that for all investigated architectures, such a correlation can be obtained.

Second, the research is exploratory. R^2 scores are obtained via cross-validation and not from a test set. Moreover, these scores are drawn from the full selection, which contains redundant summaries and summaries with a negative impact on the correlation, and the best selection, which is a selection made a posteriori to obtain the highest R^2. In this light, the results from the study have little rigorous basis and should be taken with a grain of salt.

Last, the models to predict the test performance are based only on neural networks of the the same architecture and trained on the same data. As shown in 5.4, the established correlations do not carry over as-is across architectures and may even differ in sign. It is unclear exactly what the influence of the specific architecture on the nature of the correlations is and how they it be predicted. By the methods of this project, one can not pick an arbitrary trained neural network and predict its generalization gap.

6.3 Future work

As discussed above, more work needs to be done to find a sampling strategy that consistently selects the neurons that bring forth summaries that correlate well with the generalization gap. As discussed in 3.2.2, one possibility is the use of sampling methods. While the number of activations imposes the strategic use of methods to limit the computational complexity, the clustering of a graph is well-researched and many fast approximate methods exist. This may be combined with the methods of *Mapper* (Singh, Memoli, and Carlsson, 2007), which clusters the data based on its image (through a user-defined *filter function*) in a lower-dimensional space, and has previously been used on the activations of neural networks (Gabrielsson and Carlsson, 2019).

Finally, the specific influence of the sampling strategies could be explored by comparing them with neural networks where the persistence of the full functional graph is available.

Appendix A

Software packages

Analysis of the neural networks was done with *Tensorflow*. (Martín Abadi et al., 2015). The persistence diagrams were computed using *giotto-ph* (Pérez et al., 2021). The non-statistical persistence summaries were computed using *giotto-tda* (Tauzin et al., 2021). Experiments were done using *scikit-learn* (Pedregosa et al., 2011), and plots made using *Matplotlib* (Hunter, 2007). The code used to produce the results reported in this paper can be found at `https://github.com/carpelli/tfm`.

Bibliography

Arthur, David and Sergei Vassilvitskii (2007). “k-means++: The Advantages of Careful Seeding”. en. In.

Atienza, Nieves, Rocio Gonzalez-Díaz, and Manuel Soriano-Trigueros (Nov. 2020). “On the stability of persistent entropy and new summary functions for topological data analysis”. en. In: *Pattern Recognition* 107, p. 107509. ISSN: 0031-3203. DOI: 10.1016/j.patcog.2020.107509. URL: https://www.sciencedirect.com/science/article/pii/S0031320320303125 (visited on 10/19/2022).

Ballester, Rubén et al. (Mar. 2022). *Towards explaining the generalization gap in neural networks using topological data analysis*. arXiv:2203.12330 [cs, math]. DOI: 10.48550/arXiv.2203.12330. URL: http://arxiv.org/abs/2203.12330 (visited on 10/17/2022).

Bonis, Thomas et al. (2016). “Persistence-Based Pooling for Shape Pose Recognition”. en. In: *Computational Topology in Image Context*. Ed. by Alexandra Bac and Jean-Luc Mari. Lecture Notes in Computer Science. Cham: Springer International Publishing, pp. 19–29. ISBN: 978-3-319-39441-1. DOI: 10.1007/978-3-319-39441-1_3.

Chazal, Frédéric and Bertrand Michel (Feb. 2021). *An introduction to Topological Data Analysis: fundamental and practical aspects for data scientists*. arXiv:1710.04019 [cs, math, stat]. URL: http://arxiv.org/abs/1710.04019 (visited on 10/22/2022).

Corneanu, Ciprian et al. (May 2020). *Computing the Testing Error without a Testing Set*. arXiv:2005.00450 [cs]. DOI: 10.48550/arXiv.2005.00450. URL: http://arxiv.org/abs/2005.00450 (visited on 10/19/2022).

Corneanu, Ciprian A. et al. (June 2019). “What Does It Mean to Learn in Deep Networks? And, How Does One Detect Adversarial Attacks?” en. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Long Beach, CA, USA: IEEE, pp. 4752–4761. ISBN: 978-1-72813-293-8. DOI: 10.1109/CVPR.2019.00489. URL: https://ieeexplore.ieee.org/document/8953424/ (visited on 01/07/2023).

Darlow, Luke N. et al. (Oct. 2018). *CINIC-10 is not ImageNet or CIFAR-10*. arXiv:1810.03505 [cs, stat]. DOI: 10.48550/arXiv.1810.03505. URL: http://arxiv.org/abs/1810.03505 (visited on 01/08/2023).

Di Fabio, Barbara and Massimo Ferri (July 2015). *Comparing persistence diagrams through complex vectors*. arXiv:1505.01335 [cs, math]. DOI: 10.48550/arXiv.1505.01335. URL: http://arxiv.org/abs/1505.01335 (visited on 01/08/2023).

Edelsbrunner, Herbert and John L. Harer (Jan. 2010). *Computational Topology: An Introduction*. English. Providence, R.I: American Mathematical Society. ISBN: 978-0-8218-4925-5.

Frankle, Jonathan and Michael Carbin (Mar. 2019). *The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks*. arXiv:1803.03635 [cs]. DOI: 10.48550/arXiv.1803.03635. URL: http://arxiv.org/abs/1803.03635 (visited on 01/13/2023).

Gabrielsson, Rickard Brüel and Gunnar Carlsson (Oct. 2019). *Exposition and Interpretation of the Topology of Neural Networks*. arXiv:1810.03234 [cs]. URL: http://arxiv.org/abs/1810.03234 (visited on 01/12/2023).

Gebhart, Thomas, Paul Schrater, and Alan Hylton (May 2019). *Characterizing the Shape of Activation Space in Deep Neural Networks*. arXiv:1901.09496 [cs, stat]. URL: http://arxiv.org/abs/1901.09496 (visited on 01/12/2023).

Goldfarb, Daniel (Oct. 2018). *Understanding Deep Neural Networks Using Topological Data Analysis*. arXiv:1811.00852 [cs]. DOI: 10.48550/arXiv.1811.00852. URL: http://arxiv.org/abs/1811.00852 (visited on 01/12/2023).

Han, Song, Huizi Mao, and William J. Dally (Feb. 2016). *Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding*. en. arXiv:1510.00149 [cs]. URL: http://arxiv.org/abs/1510.00149 (visited on 01/15/2023).

Hu, Hengyuan et al. (July 2016). *Network Trimming: A Data-Driven Neuron Pruning Approach towards Efficient Deep Architectures*. en. arXiv:1607.03250 [cs]. URL: http://arxiv.org/abs/1607.03250 (visited on 01/03/2023).

Hunter, J. D. (2007). “Matplotlib: A 2D graphics environment”. In: *Computing in Science & Engineering* 9.3. Publisher: IEEE COMPUTER SOC, pp. 90–95. DOI: 10.1109/MCSE.2007.55.

Jiang, Yiding et al. (Dec. 2020). *NeurIPS 2020 Competition: Predicting Generalization in Deep Learning*. arXiv:2012.07976 [cs, stat]. DOI: 10.48550/arXiv.2012.07976. URL: http://arxiv.org/abs/2012.07976 (visited on 01/02/2023).

Krizhevsky, Alex (2009). *Learning Multiple Layers of Features from Tiny Images*. en. Tech. rep.

Kumar, Aakash et al. (Dec. 2022). “CorrNet: pearson correlation based pruning for efficient convolutional neural networks”. en. In: *International Journal of Machine Learning and Cybernetics* 13.12, pp. 3773–3783. ISSN: 1868-808X. DOI: 10.1007/s13042-022-01624-5. URL: https://doi.org/10.1007/s13042-022-01624-5 (visited on 01/04/2023).

Lin, Min, Qiang Chen, and Shuicheng Yan (Mar. 2014). *Network In Network*. arXiv:1312.4400 [cs]. URL: http://arxiv.org/abs/1312.4400 (visited on 01/02/2023).

Liu, Jinlong et al. (Feb. 2020). *Understanding Why Neural Networks Generalize Well Through GSNR of Parameters*. arXiv:2001.07384 [cs, stat]. URL: http://arxiv.org/abs/2001.07384 (visited on 01/13/2023).

Martín Abadi et al. (2015). *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. URL: https://www.tensorflow.org/.

Molchanov, Pavlo et al. (June 2017). *Pruning Convolutional Neural Networks for Resource Efficient Inference*. arXiv:1611.06440 [cs, stat]. DOI: 10.48550/arXiv.1611.06440. URL: http://arxiv.org/abs/1611.06440 (visited on 01/08/2023).

Naitzat, Gregory, Andrey Zhitnikov, and Lek-Heng Lim (Apr. 2020). *Topology of deep neural networks*. arXiv:2004.06093 [cs, math, stat]. DOI: 10.48550/arXiv.2004.06093. URL: http://arxiv.org/abs/2004.06093 (visited on 01/12/2023).

Netzer, Yuval et al. (2011). “Reading Digits in Natural Images with Unsupervised Feature Learning”. en. In: *NIPS: Workshop on Deep Learning and Unsupervised Feature Learning 2011*.

Nezhadarya, Ehsan et al. (May 2020). *Adaptive Hierarchical Down-Sampling for Point Cloud Classification*. arXiv:1904.08506 [cs]. URL: http://arxiv.org/abs/1904.08506 (visited on 01/03/2023).

Pedregosa, F. et al. (2011). “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12, pp. 2825–2830.

Pérez, Julián Burella et al. (Aug. 2021). *giotto-ph: A Python Library for High-Performance Computation of Persistent Homology of Vietoris-Rips Filtrations*. arXiv:2107.05412 [cs]. DOI: 10.48550/arXiv.2107.05412. URL: http://arxiv.org/abs/2107.05412 (visited on 01/02/2023).

Rieck, Bastian et al. (Feb. 2019). "Neural Persistence: A Complexity Measure for Deep Neural Networks Using Algebraic Topology". In: arXiv:1812.09764 [cs, math, stat], 25 p. DOI: 10.3929/ethz-b-000327207. URL: http://arxiv.org/abs/1812.09764 (visited on 01/12/2023).

Simonyan, Karen and Andrew Zisserman (Apr. 2015). *Very Deep Convolutional Networks for Large-Scale Image Recognition*. arXiv:1409.1556 [cs]. DOI: 10.48550/arXiv.1409.1556. URL: http://arxiv.org/abs/1409.1556 (visited on 01/02/2023).

Singh, Gurjeet, Facundo Memoli, and Gunnar Carlsson (2007). "Topological Methods for the Analysis of High Dimensional Data Sets and 3D Object Recognition". en. In: Accepted: 2014-01-29T16:52:11Z ISSN: 1811-7813. The Eurographics Association. ISBN: 978-3-905673-51-7. DOI: 10.2312/SPBG/SPBG07/091-100. URL: https://diglib.eg.org:443/xmlui/handle/10.2312/SPBG.SPBG07.091-100 (visited on 10/19/2022).

Solo, Victor (Aug. 2019). *Pearson Distance is not a Distance*. arXiv:1908.06029 [stat]. DOI: 10.48550/arXiv.1908.06029. URL: http://arxiv.org/abs/1908.06029 (visited on 01/11/2023).

Tauzin, Guillaume et al. (2021). "giotto-tda: A Topological Data Analysis Toolkit for Machine Learning and Data Exploration". In: *Journal of Machine Learning Research* 22.39, pp. 1–6. URL: http://jmlr.org/papers/v22/20-325.html.

Watanabe, Satoru and Hayato Yamana (Sept. 2022). "Overfitting measurement of convolutional neural networks using trained network weights". en. In: *International Journal of Data Science and Analytics* 14.3, pp. 261–278. ISSN: 2364-4168. DOI: 10.1007/s41060-022-00332-1. URL: https://doi.org/10.1007/s41060-022-00332-1 (visited on 01/12/2023).

White, Douglas J. (Jan. 1991). "The maximal-dispersion problem". In: *IMA Journal of Management Mathematics* 3.2, pp. 131–140. ISSN: 1471-678X. DOI: 10.1093/imaman/3.2.131. URL: https://doi.org/10.1093/imaman/3.2.131 (visited on 01/14/2023).

Zhang, Chiyuan et al. (Feb. 2017). *Understanding deep learning requires rethinking generalization*. arXiv:1611.03530 [cs]. DOI: 10.48550/arXiv.1611.03530. URL: http://arxiv.org/abs/1611.03530 (visited on 01/13/2023).