Topological Data Analysis of Mass Spectra

Project Course in Mathematical and Statistical Modelling

Axel Andersson
axean@student.chalmers.se

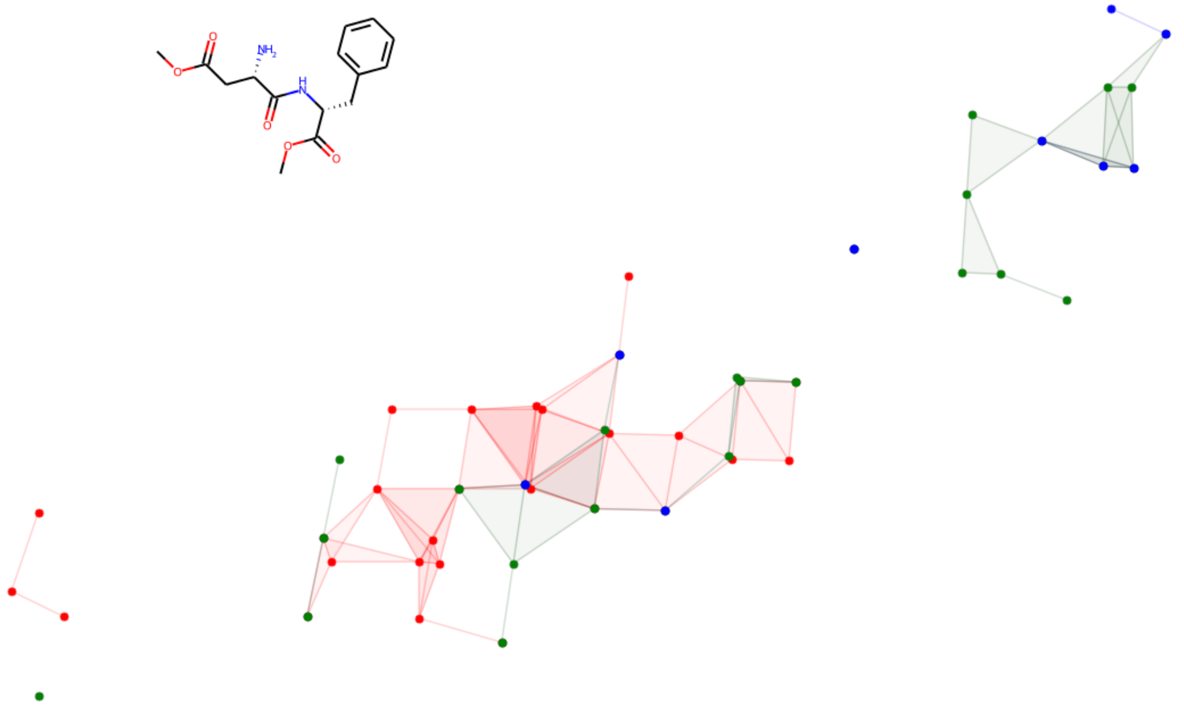
Jens Michelsen
gusmicje@student.gu.se

Olof Johansson
olojohan@student.chalmers.se

Gustaf Hulthe
gustaf.hulthe@astrazeneca.com

Mats Josefson
mats.josefson@astrazeneca.com

April 7, 2022



Abstract

The project, reported herein, was conducted as a part of the course MVE385 / MMA520, MSA520 “Project course in mathematical and statistical modelling” in the autumn of 2021, in collaboration with Mats Josefson and Gustaf Hulthe at AstraZeneca. The aim of the project was to utilize newly developed methods, collectively known as topological data analysis (TDA), as a means for extracting features from molecular mass spectrometry (MS) data that could then be used to identify, classify or predict the molecule, or it’s molecular properties, based on their mass spectra.

While there have been other attempts to use mass spectra as a type of fingerprint to identify the molecule, the task has so far proven to be too complex to perform directly. Instead, the idea is to extract qualitatively important features from mass spectra, and use this as a representation in subsequent machine learning pipelines. In this project we explore the feasibility of using topological features (extracted using TDA) to this end.

Since TDA is a new and rapidly developing field, and, to our knowledge, has never been applied to MS data, there is no best praxis to lean on, and several possible approaches must be explored. Here we report on our experiences using several different choices of point cloud constructions, filtrations and vectorized representations of the topological features we managed to extract from a data set of molecules, with corresponding mass spectra simulated by third party software. Finally, we also report on attempts to train both OLS and CNN models to predict the properties of the molecules using topological features from their mass spectra as inputs.

Contents

1 Background

Mass spectrometry (MS) is an important technology in analyzing different molecules where a large amount of molecules are ionized and collided under the influence of electromagnetic fields in vacuum. This produces specific fragment spectra for each substance which corresponds the intensities of these fragments. As these spectra are usually manually processed, trying to interpret the chemical structure of unknown impurities in different substances from these spectra is highly time consuming. Ideally, the fragments could be represented as an equation system which, by knowing the chemical structure of the given molecule, would be solvable. Unfortunately, this is not the case since the mass-spectrometer is not theoretically ideal. Therefore, each measurement made is affected by small errors accumulated from the uncertainties of the mass-spectrometer and thus need manual interpretation to reduce the number of possible combinations of the chemical structure.

Therefore, to find tools that support these manual interpretations would facilitate the process and make it more efficient. The aim is to find such a tool using topology which can make the information easier to understand as well as indicate more important features. Thus, this can also be used to improve the data processing in a machine learning pipeline and hence obtain a higher performance in the interpretation of the fragment spectra.

It has shown to be a too complex task to directly model molecular structures or molecular descriptors from MSMS spectra through deep learning. Instead, there has been several attempts to build models incorporating a simplified but qualitatively richer representation of either the spectra or the structures. For molecular structures representation this has been fingerprints [3] by various methods or through embeddings generated by autoencoders [8]. For spectral representations, attempts with autoencoder embeddings have been tried beside machine learning generated fingerprints. However, a direct topological representation of the spectra has so far never been tried to our knowledge, which is why we here investigate this path.

1.1 Molecule database

A molecule can be described with various identifiers. One of the most commonly known ways to identify a molecule is with its chemical formula. However there exists molecules which have identical chemical formulas but different molecular structures. Two molecules which have different molecular structures but same chemical formula are called isomers. In some sense isomers are analogous to how you can have multiple distinct terms which sum to the same number. A less ambiguous way to identify a molecule is by its structure, i.e. its arrangements of atoms in space. This is usually done with its skeletal structural formula which is usually represented as an image. However, this image representation is not very amenable to processing by a computer. A common serialization method for structural formulas is the SMILES representation.

SMILES stands for "simplified molecular-input line-entry system". We were given 10629 distinct SMILES strings which were compiled by the group at AstraZeneca, one SMILES string for each molecule in a subset of selected molecules from the DrugBank dataset [6]. This subset, which we call the DrugBank, contains mainly organic molecules which are useful for drug research and development.

For each of these molecules, there exist several descriptors, representing physical and chemical properties of the specific molecule. These descriptors can be either 1-dimensional, 2-dimensional or 3-dimensional. The 1-dimensional descriptors are the simplest representations that corresponds to information computed from the molecular formula including type of atoms, molecular weight and count of different atoms. The 2-dimensional descriptors which often represent information corresponding to the size, shape and electronic distribution within the molecule. It is more complex than the 1-dimensional descriptors and requires more molecular information to be computed. The 3-dimensional descriptors represent information that corresponds to the 3-dimensional conformation of the molecule, for example the intramolecular hydrogen bonding. Other calculations which involve quantum mechanics can be used to attain 3-dimensional descriptors [7]. Due to availability, only 2-dimensional descriptors were used in this project, all of which calculated and supplied by the group at AstraZeneca.

1.2 Mass spectral data

To obtain mass spectral data, each SMILES string was used to compute a predicted mass spectrum with the ESI MS/MS Spectral Prediction program CFM-ID 4.0 [16]. An alternative way to obtain this data would be to use a mass spectral database, where experimentally obtained mass spectra are indexed with their known input molecules. The advantage of using CFM-ID instead of these experimentally sourced databases is that CFM-ID can predict mass spectra for molecules which might not exist in any such database [16].

For each molecule, three mass spectra were predicted at different energy levels called low, medium, and high energy (10V, 20V, and 40V). The main difference between energy levels is that more energy will cause more fragments to appear. Predicting a mass spectrum for an organic molecule in the DrugBank is computationally quite expensive. On an Intel Core i5-6600K 3.5 GHz processor with 8GB DDR4 RAM predicting the three mass spectra for a single molecule could take anything between five seconds to several minutes, depending on the complexity of the molecule. The mass spectra were pre-computed and stored in a JSON file for quick lookup access. The spectra were computed over night on 20 physical cores in parallel. Our predicted spectral library contains only 9701 mass spectrum triplets (one for each energy level) out of the 10629 SMILES strings we were given. There are 928 missing because some molecules were incompatible with ESI MS/MS spectral prediction. Among other reasons, the main reason for incompatibility for these molecules was that they could not be ionised. Only positive ionization with adduct type [M+H]+ was considered.

2 Theory

In this section we aim to provide the reader with an accessible introduction to topological data analysis. For more comprehensive and detailed introductions we refer the reader to the book by Edelsbrunner et al. [2] or the lecture notes of v. Nanda [14].

The aim of topological data analysis (TDA) is to extract topological features from data sets. The data sets are either point clouds (see below) or can be transformed into point clouds.

2.1 Point cloud, simplicial complex and filtration

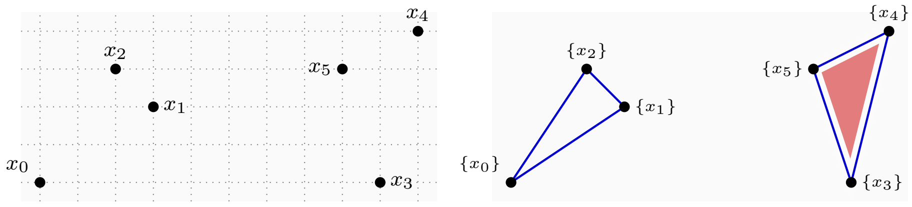
Point cloud An N-dimensional point cloud, $X = \{x_0, x_1, \ldots, x_n\}$, is a set of ordered N-tuples, x_i, called *points*, assumed to be taken from the same topological space. Usually we work with point clouds from finite dimensional Euclidean space, or some other metric space, which are examples of topological spaces with extra structures. For the purposes of this report we solely consider finite point clouds in $\mathbb{R}^2$ and $\mathbb{R}^3$.

To extract topological properties, or features, from such a point cloud one can use the locations of the points as vertices, or 0-simplices (see below), and connect these into larger and larger structures. Such collections of structures is known as a simlicial complexes.

Simplicial complex An abstract simplicial complex, K, is a finite collection of sets such that, if $\sigma \in K$ and $\tau \subseteq \sigma$ then $\tau \in K$. In other words, an abstract simplicial complex is a family of sets which is closed under taking subsets. For example, for $T = \{x_0, x_1, x_2\}$ to be an element of a simplicial complex, K, it must also contain the nontrivial subsets of T. A minimal simplicial complex containing T would then be $\{\{x_0\}, \{x_1\}, \{x_2\}, \{x_0, x_1\}, \{x_0, x_2\}, \{x_1, x_2\}, \{x_0, x_1, x_2\}\}$.

The elements of a simplicial complex are known as simplices. A p-simplex is a set consisting of $p + 1$ elements. So, for instance, a 0-simplex consists of 1 element, $\{x_i\}$, and is typically referred to as a vertex. A 1-simplex consists of 2 elements, $\{x_i, x_j\}$, and is typically referred to as an edge. A 2-simplex consists of 3 elements $\{x_i, x_j, x_k\}$, and so on.

In terms of the point cloud, X, of a data set, the simplices correspond to nontrivial subsets, $\sigma \subseteq X$. Each point, x_i, can be associated with a 0-simplex $\{x_i\}$. A 1-simplex $\{x_i, x_j\}$ can be represented as a line segment connecting the two points x_i and x_j. A 2-simplex $\{x_i, x_j, x_k\}$ can be represented as a filled triangle with vertices at x_i, x_j and x_k. The reason it is filled, is to distinguish it from the *simplicial complex* $\{\{x_i, x_j\}, \{x_i, x_k\}, \{x_j, x_k\}\}$, which would be represented as the set of edges connecting the points x_i, x_j and x_k.



(2D Point Cloud)

(Simplicial Complex)

Figure 1: Left: A two-dimensional point cloud. Right: A simplicial complex embedded into the two-dimensional space. With 0-simplices, $\{x_0\}, \{x_1\}, \ldots, \{x_5\}$ at the positions of the points in the point cloud, along with 1-simplices, $\{x_0, x_1\}, \{x_1, x_2\}, \{x_0, x_2\}$ represented by the corresponding edges. There is also a 2-simplex $\{3, 4, 5\}$, represented by the filled triangle with edges at the vertices $\{x_3\}, \{x_4\}, \{x_5\}$. Since this is a simplicial complex, both the vertices of the triangle, as well as the edges $\{x_3, x_4\}, \{x_4, x_5\}, \{x_3, x_5\}$ are included, i.e. all the non-trivial subsets of $\{x_3, x_4, x_5\}$ are included. Note, however, that the 2-simplex $\{x_0, x_1, x_2\}$ is not included in the simplicial complex, only its vertices and edges. Also note that the simplicial complex consists of two disconnected components.

Starting from the simplest (non-trivial) simplicial complex consisting only of the vertices $\{\{x_0\}, \{x_1\}, \ldots, \{x_n\}\}$ corresponding to the points of the point cloud, we can gradually build up larger and larger simplicial complexes. This is known as a filtration.

Filtration

A filtration is a monotonically increasing sequence of simplicial complexes, $(K_i)_{i=0,\ldots,n}$, s.t.

$$K_0 \subseteq K_1 \subseteq \ldots \subseteq K_n$$

The size and dimension of a filtration depends on the topology of the point cloud and the complex used. Some popular examples of simplicial complexes that are used in applied topology are the Čech-, Vietoris-Rips-, Delaunay- and α-complexes, but in this report we consider only the first two. These complexes all have in common that they require the point cloud to be associated with a metric space, that is, there is a sense of distance between the points.

Čech-complex

The Čech-complex, $\check{\text{C}}\text{ech}_\epsilon(X)$, of a data set X is parametrized by a scale ϵ. For each point $x_i \in X$ we attach a N-ball, $B_{x_i}(\epsilon) = \{x : d(x, x_i) < \epsilon\}$ of size ϵ. The Čech-complex is then defined by

$$\check{\text{C}}\text{ech}_\epsilon(X) = \{\sigma : \bigcap_{x_i \in \sigma} B_{x_i}(\epsilon) \neq \emptyset\}$$

For increasing values $\epsilon_1 \leq \epsilon_2 \leq \ldots \leq \epsilon_n$, the corresponding Čech-complexes, are monotonically increasing, $\check{\text{C}}\text{ech}_{\epsilon_1} \subseteq \check{\text{C}}\text{ech}_{\epsilon_2} \subseteq \ldots \subseteq \check{\text{C}}\text{ech}_{\epsilon_n}$, thus defining a filtration - the Čech-filtration. For an illustration of a Čech filtration where the 0-simplices are in the Euclidean plane, see Figure 2. Note that a filtration of Čech complexes constructed from points in the Euclidean plane can still contain higher order simplices (e.g. tetrahedra), depending on which filtration values are considered.

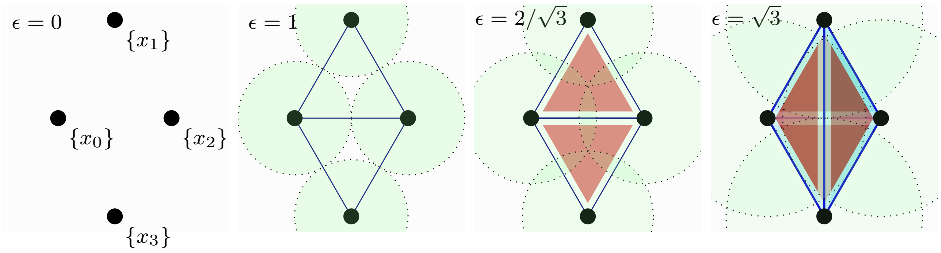


Figure 2: Example of a Čech-filtration for points x_0, x_1, x_2, x_3 in $\mathbb{R}^2$, positioned at $(\pm 1, 0)$ and $(0, \pm\sqrt{3})$. As the radius of the balls is increased, and thus the scale ϵ, they eventually intersect pairwise when $\epsilon \geq 1$, forming the simplices $\{x_0, x_1\}$, $\{x_1, x_2\}$, $\{x_2, x_3\}$, $\{x_1, x_3\}$ and $\{x_0, x_2\}$ (edges). When $\epsilon \geq 2/\sqrt{3}$, the balls from x_0, x_1, x_2 have a nonzero common intersection, thus forming the simplex $\{x_0, x_1, x_2\}$ (a triangle), and similarly with $\{x_0, x_2, x_3\}$. Finally, when $\epsilon \geq \sqrt{3}$, the balls from all points have a common intersection, thus forming the simplex $\{x_0, x_1, x_2, x_3\}$ (a tetrahedron).

Vietoris-Rips complex

The Vietoris-Rips complex constructed from elements of X at scale ϵ is defined as

$$\text{VR}_\epsilon(X) = \{\sigma \subseteq X : \operatorname{diam} \sigma \leq 2\epsilon\}$$

where $\operatorname{diam} \sigma = \max_{x_i, x_j \in \sigma} d(x_i, x_j)$ is the generalized diameter of the simplex σ, i.e. the maximal distance between its vertices.

The Vietoris-Rips filtration is similar to the Čech complex because it is also constructed by attaching an N-ball of radius ϵ to each point x_i. As ϵ is increased, a 1-simplex is formed between vertices $\{x_i\}$ and $\{x_j\}$ if their respective balls intersect. Furthermore, as soon as the simplices $\{x_i, x_j\}$, $\{x_i, x_k\}$ and $\{x_j, x_k\}$ are all part of the simplicial complex, then the simplicial complex $\{x_i, x_j, x_k\}$ is also added. Comparing with the Čech filtration in Figure 2, this would correspond to the 2-simplices immediately being added to the complex already at $\epsilon = 1$ in stage (b) instead of at stage (c). While the Vietoris-Rips construction has some properties that allows for fast algorithms, it ignores many smaller combinatorics, which yields a much coarser filtration.

2.2 Homology groups

The topological features under consideration in topological data analysis are known as homologies. Intuitively one can think of it as looking for multidimensional "holes" in the simplicial complex. The low dimensional homologies can be easily interpreted: 0-homologies are connected components, 1-homologies are loops and 2-homologies are three dimensional voids. To define this more algebraically we need the concept of p-chains, boundaries and some other helpful concepts. Readers who need a refresher on group theory may want to read Appendix C first.

Chain

A p-chain, c, is a formal sum of p-simplices, $c = \sum_{j=1}^{n_p} \alpha_j \sigma_j$, where $\sigma_j \in K$ and n_p is the number of p-simplices in the simplicial complex K. The coefficients α_j are most conveniently defined over $\mathbb{Z}_2 = \{0, 1\}$, i.e. the cyclic group with two elements, 0 and 1, with addition modulo 2 (although other choices are possible). For instance, a 0-chain could be $\{x_0\}+\{x_3\}+\{x_5\}$ (if these vertices are in K), while a 1-chain could be $\{x_0, x_3\}+\{x_3, x_5\}$ (see Figure 3 for a visual representation of some 1-chains). Using the field $\mathbb{Z}_2$ allows us to interpret a given p-chain as either including a specific p-simplex σ_j, if $\alpha_j = 1$, or not including σ_j, if $\alpha_j = 0$. Note that the coefficients provide us with a means to add p-chains; if $c_1 = \{x_0, x_1\} + \{x_3, x_5\}$ and $c_2 = \{x_1, x_3\} + \{x_3, x_5\}$, then $c_1 + c_2 = \{x_0, x_1\} + \{x_3, x_5\} + \{x_1, x_3\} + \{x_3, x_5\} = \{x_0, x_1\} + \{x_1, x_3\}$, where the last step follows from addition modulo 2, imbued by the coefficients. Any simplex in c_1 which is also in c_2 will be cancelled under addition.

Chain group

The pth chain group, C_p, of a simplicial complex K, is the set of possible p-chains in K, together with addition described above. To see that this is a group, note that i) there exists a unit element 0 ($\alpha_j = 0$, $\forall j \in \{1, 2, \ldots, n_p\}$), ii) addition of p-chains in C_p will always create a new p-chain in C_p, and iii) each p-chain is its own inverse (due to modulo 2 arithemtic). Furthermore, it is an abelian group since addition is commutative, and the group can be generated by the set of p-simplices in K. Thus $\operatorname{rank} C_p = n_p$.

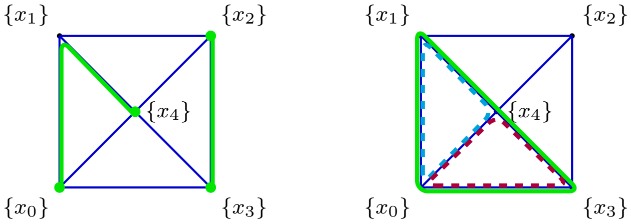


Figure 3: Left: a single 1-chain $c = \{x_0, x_1\} + \{x_1, x_4\} + \{x_2, x_3\}$ (green lines) with boundary $\{x_0\} + \{x_4\} + \{x_2\} + \{x_3\}$ (green circles). Two 1-chains $c_1 = \{x_0, x_1\} + \{x_1, x_3\} + \{x_3, x_0\}$ (cyan) and $c_2 = \{x_0, x_3\} + \{x_3, x_4\} + \{x_4, x_0\}$ (purple), both without a boundary, i.e. 1-cycles. Adding the two together creates $c = c_1 + c_2 = \{x_0, x_1\} + \{x_1, x_3\} + \{x_3, x_0\}$, which is again a 1-cycle. Note that the 1-simplex $\{x_0, x_4\}$ which exists in both c_1 and c_2 is absent in c due to addition modulo 2.

Boundary operator Given a simplicial complex K, the p-th boundary operator ∂_p is a function that assigns each p-simplex $\sigma = \{x_0, x_1, \dots, x_p\}$ to its *boundary*:

$$\partial_p \sigma = \sum_j \{x_0, x_1, \dots, \hat{x}_j, \dots, x_p\}$$

where $\hat{x}_j$, means that the element x_j should be removed, thus the boundary of a p-simplex is a $p-1$-chain. If we further require the boundary operator to commute with addition, we see that this implies that the boundary of a p-chain is a $p-1$-chain. Thus the boundary operator $\partial_p : C_p \to C_{p-1}$ is a mapping (in fact, a homomorphism) between chain groups.

As an example, consider the 2-chain consisting solely of the 2-simplex $\sigma = \{x_0, x_1, x_2\}$. Applying the boundary operator ∂_2 we get:

$$\partial_2 \sigma = \{x_1, x_2\} + \{x_0, x_2\} + \{x_0, x_1\}$$

Thus, the *boundary* of the 2-chain is a 1-chain. Applying now the boundary operator ∂_1 to this 1-chain we get:

$$\begin{aligned}
\partial_1 (\partial_2 \sigma) &= \partial_1 (\{x_1, x_2\} + \{x_0, x_2\} + \{x_0, x_1\}) \\
&= \partial_1\{x_1, x_2\} + \partial_1\{x_0, x_2\} + \partial_1\{x_0, x_1\} && \text{addition commutes with } \partial_p \\
&= \{x_1\} + \{x_2\} + \{x_0\} + \{x_2\} + \{x_0\} + \{x_1\} && \scriptstyle \partial_1\{x_i, x_j\} = \{x_i\} + \{x_j\} \\
&= 0 && \text{addition modulo 2}
\end{aligned}$$

In other words, the boundary of the boundary of σ is 0. The result holds more generally; $\partial_p \circ \partial_{p+1} = 0$, which is commonly expressed as "boundaries do not have boundaries themselves".

Cycle A p-chain, c, without a boundary, i.e. $\partial_p c = 0$, is called a p-cycle. The set

Cycle group

of p-cycles forms a subgroup of C_p, called the cycle group $Z_p = \ker \partial_p$, where $\ker \partial_p = \{c \in C_p : \partial_p c = 0\}$.

Boundary group

Since the boundary of a boundary is always 0 (recall $\partial_p \circ \partial_{p+1} = 0$), it follows that boundaries of $p+1$-chains are are p-cycles. However, not all p-cycles are boundaries of $p+1$-chains. Thus, there generally exists a subgroup $B_p \subseteq Z_p$ called the boundary group $B_p = \operatorname{im} \partial_{p+1}$, where $\operatorname{im} \partial_{p+1} = \{c \in C_p : c = \partial_{p+1}\tau,\ \tau \in C_{p+1}\}$.

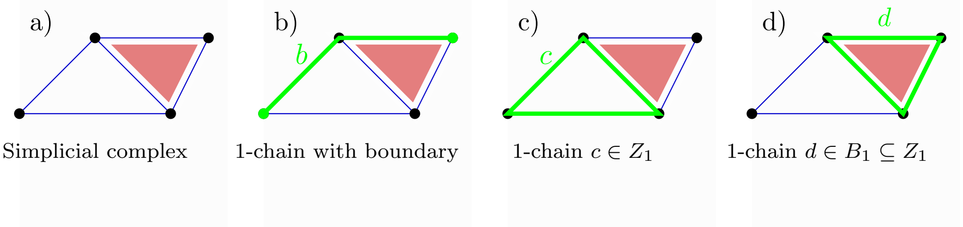


Figure 4: Some elements (marked in green) of the chain group C_1 for the simplicial complex shown in a). The 1-chain b is not a cycle, and thus has a boundary, namely the vertices marked by green dots. The 1-chain c is a cycle, but is not the boundary of another simplex in the simplicial complex. The 1-chain d is a cycle, but is also the boundary of a 2-simplex in the simplicial complex.

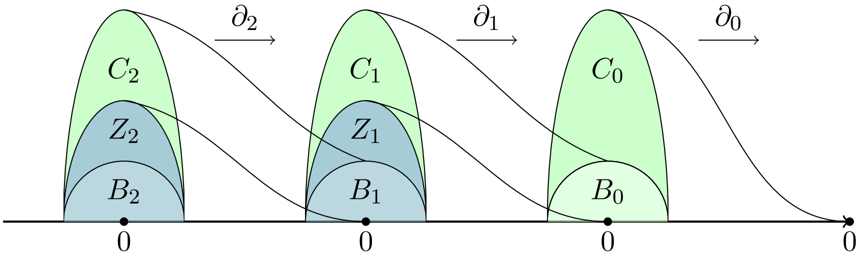


Figure 5: A depiction of the action of the boundary operators $\partial_p : C_p \to C_{p-1}$, in terms of the subgroups Z_p and B_p. Every element of C_p is mapped to the boundary group B_{p-1} which is a subset of the cycle group Z_{p-1}. Since all cycles lack boundaries, the cycle groups Z_p are mapped to 0. For 0-chains the situation is a bit special since all vertices lack boundaries so all of C_0 is mapped to 0. Indeed, when it comes to vertices, one can not really speak of cycles in an intuitive way. However, if we define it by the lack of boundary, then the subgroup Z_0 is all of C_0.

Homology group

The p-th Homology group is the quotient group of Z_p with respect to B_p

$$H_p = Z_p/B_p = \ker \partial_p / \operatorname{im} \partial_{p+1}$$

thus partitioning the cycle group into equivalence classes, or homology classes. Two p-cycles, c_1 and c_2, are equivalent, $c_1 \sim c_2$, if $c_1 + B_p = c_2 + B_p$. In this context, we say that the two cycles are *homologous*, or of the same *homology*

Homology class

class. Intuitively, this can be understood as two cycles being homologous if they differ only by elements of the boundary group, i.e. chains that are boundaries of higher dimensional chains in the complex.

Betti number The p-th Betti number is defined as $\beta_p = \operatorname{rank} H_p$. In other words it is the minimum number of basis elements, or generators, needed to span the homology group H_p.

2.2.1 An instructive example

To get a better feeling for the definitions above, let's consider some examples and perform actual calculations. First, consider the simplicial complex K in Figure 4, and for the sake of brevity, lets label[1] the vertices by $v_0, \ldots, v_3$, edges by a, b, c, d, e and the triangle by T, as in Figure 6.

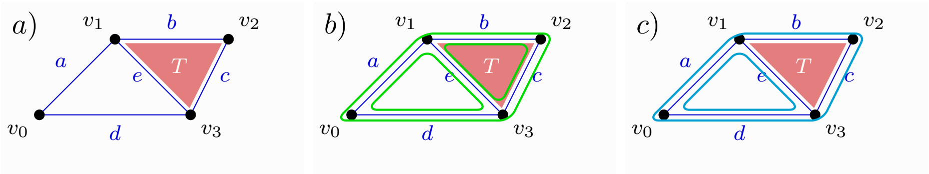


Figure 6: a) The simplicial complex from Figure 4 but with convenient labels for the simlices. b) The 1-chains of Z_1. Note that the larger cycle can be obtained by adding the two smaller ones, thus the group is only generated by two elements. c) The two p-chains in the non-trivial homology class represented by $a + d + e$. Note that they differ only by the cycle $b + c + e$, which is the boundary of the 2-simplex T.

Since the simplicial complex contains at most 2-simplices, the chain groups $C_{p>2} = 0$. Thus we are interested in the chain complex $C_2 \overset{\partial_2}{\to} C_1 \overset{\partial_1}{\to} C_0 \overset{\partial_0}{\to} 0$, and corresponding homology groups H_2, H_1, H_0. The only 2-simplex is T, and thus this element generates the chain group $C_2 = \{0, T\}$. There can be no cycles, and no cycles that are borders of 3-chains. Thus the homology group, $H_2 = Z_2/B_2$ is trivially 0 (i.e. the zero group).

The chain group C_1 is larger, and generated by all the edges a, b, c, d, e. To compute the homology group H_1, we need to first compute the cycle group Z_1 and the boundary group B_1.

The boundary group, $B_1 = \operatorname{im} \partial_2$, is generated by $\partial_2 T = b + c + e$, so $B_1 = \{0, b + c + e\}$. The cycle group, $Z_1 = \ker \partial_1$, can be found by considering a

[1] For clarity, the vertices v_i are 1-simplices $\{x_i\}$, and the edges are 2-simplices $\{x_i, x_j\}$ and the triangle is the 2-simplex $\{x_1, x_2, x_3\}$

general 1-chain, $\alpha a + \beta b + \gamma c + \delta d + \varepsilon e$, with $\alpha, \beta, \gamma, \delta, \varepsilon \in \mathbb{Z}_2$, under ∂_1.

$$\begin{aligned}
&\partial_1 (\alpha a + \beta b + \gamma c + \delta d + \varepsilon e) \\
&\qquad = \alpha \partial_1 a + \beta \partial_1 b + \gamma \partial_1 c + \delta \partial_1 d + \varepsilon \partial_1 e \\
&\qquad = \alpha(v_0 + v_1) + \beta(v_1 + v_2) + \gamma(v_2 + v_3) + \delta(v_3 + v_0) + \varepsilon(v_1 + v3) \\
&\qquad = (\alpha + \delta) v_0 + (\alpha + \beta + \varepsilon) v_1 + (\beta + \gamma) v_2 + (\gamma + \delta + \varepsilon) v_3
\end{aligned}$$

Setting this to zero allows us to identify the elements that are in Z_1. Note that, since v_0, v_1, v_2, v_3 forms an independent basis for C_0, this means that the prefactors must all individually be zero. Since we are working with addition modulo 2, this means that $\alpha + \delta = 0 \Rightarrow \alpha = \delta$ and $\beta + \gamma = 0 \Rightarrow \beta = \gamma$. From this, the last two constraints can be seen to be identical:

$$\alpha + \beta + \varepsilon = 0 \qquad \text{or} \qquad \varepsilon = \alpha + \beta$$

Thus a general element in the cycle group can be written as

$$\alpha a + \beta b + \beta c + \alpha d + (\alpha + \beta) e = \alpha(a + d + e) + \beta(b + c + e)$$

So the cycle group is generated by the two elements $a + d + e$ and $b + c + e$. For completeness, let's write out the whole cycle group:

$$Z_1 = \{0,\ a + d + e,\ b + c + e,\ a + b + c + d\}$$

where, for the last element we used that $a + d + e + b + c + e = a + b + c + d$. The three 1-chains in Z_1 are illustrated in Figure 6b).

Finally, let's consider the homology group $H_1 = Z_1/B_1$. Recall that $B_1 = \{0, b + c + e\}$. The cosets of Z_1 with respect to the boundary group are:

$$\begin{aligned}
0 + B_1 &= \{0, b + c + e\} \\
a + d + e + B_1 &= \{a + d + e, a + b + c + d\} \\
b + c + e + B_1 &= \{b + c + e, 0\} \\
a + b + c + d + B_1 &= \{a + b + c + d, a + d + e\}
\end{aligned}$$

Thus we see that there are only two unique cosets (the order in a set does not matter), or homology classes. One represented by 0 (or $b + c + e$), and one represented by the cycle $a + d + e$ (or the larger cycle $a + b + c + d$). The homology group $H_1 = \{0, a + d + e\}$ is thus generated by the cycle $a + d + e$ which is homologous to the larger cycle $a + b + c + d$ which can be obtained from $a + d + e$ by adding the element $b + c + e$ from B_1. Intuitively, we can think of homologous cycles as enclosing the same "hole". Since H_1 is generated by a single element, the Betti number is $\beta_1 = \operatorname{rank} H_1 = 1$, which can be thought of as referring to the two dimensional "hole" in the simplicial complex.

It is instructive to consider the case when we remove T from the simplicial complex. Then the boundary group is $B_1 = 0$, the trivial group. Taking the quotient with respect to the trivial group leaves the cyclic group unchanged, so the homology group is $H_1 = Z_1$. Then, both $a + d + e$ and $b + c + e$ are generators, corresponding to the two different homology classes, each enclosing a different "hole".

To compute the homology group H_0, we note first that the cycle group $Z_0 = \ker \partial_0$ is the entire chain group C_0, since all vertices are mapped to zero under ∂_0. Secondly, the boundary group $B_0 = \operatorname{im} \partial_1$ is spanned by the boundaries of each edge, $\partial_1 a = v_0 + v_1$, $\partial_1 b = v_1 + v_2$, etc. The homology group, $H_0 = Z_0 / B_0$ consists of the cosets of Z_0 with respect to B_0, and two elements, $c_1, c_2 \in Z_0 = C_0$ that produce the same coset are in the same homology class. Another way of saying this is that c_1 and c_2 are homologous if they differ only by some element in B_0. In this particular simplicial complex, any two vertices v_i and v_j are connected by some path, or 1-chain, with boundary $v_i + v_j \in B_0$. Thus, v_i and v_j are related through $v_j = v_i + (v_i + v_j)$, and therefore, any two vertices are homologous to each other. More generally, any two nonzero 0-chains are homologous. Thus, the homology group is $H_0 = \{0, v_i\}$, where v_i is any representative vertex in the simplicial complex. Since there is one generator, the Betti number is $\beta_0 = \operatorname{rank} H_0 = 1$.

It is instructive to consider the case of simplicial complexes where not all vertices are path connected, but rather the simplicial complex can be divided into *connected components*. The argument still holds for each component individually, and thus all vertices in a particular connected component are homologous, and the homology classes refer to the different connected components. The Betti number in this case is $\operatorname{rank} H_0 = \#$ connected components.

Of course, one would not like to compute the homology groups by hand when the simplicial complex is large. Fortunately, there are matrix algorithms to do this in a more automated fashion. While the complete algorithm would take us off course for this introduction, we note that it is based on the fact that the boundary matrix, ∂_p, can be represented by a $\operatorname{rank} C_{p-1}$ by $\operatorname{rank} C_p$ matrix, which can then be brought into what is known as Smith normal form. In a sense this corresponds to a change of basis in C_p and C_{p-1}, such that one subset of the basis elements generate Z_p and another generates B_{p-1}, allowing one to read of their ranks and their generators.

2.3 Persistent homology

Persistent homology group

Given a filtration $K_0 \subseteq K_1 \subseteq \ldots \subseteq K_n$, the p-th persistent homology group between steps K_i and K_j, $i < j$, is given by

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

meaning: the set of homology classes of K_i that are still present in K_j.

If a certain homology class persists over a large number of steps in the filtration, it indicates a robust feature in the topological space underlying the point cloud. In the Čech-filtration we have a continuous scale-parameter, ϵ, and we say that robust features persist over a large scale (or simply large scale features).

In practice, persistent homology is computed by labeling the simplices by the order in which they appear, so σ_i is the ith simplex added to the complex. If multiple simplices are added at the same scale, the order may be arbitrary (with some constraints). One can then define a quadratic boundary matrix, $\partial : \bigcup_p C_p \to \bigcup_p C_p$, which in a sense contains the information of all the boundary matrices ∂_p. By performing a series of column operations on this matrix, the persistent homology can be directly read off.

Let's take as an example the Čech-filtration in Figure 2, with simplex labeling according to Figure 7.

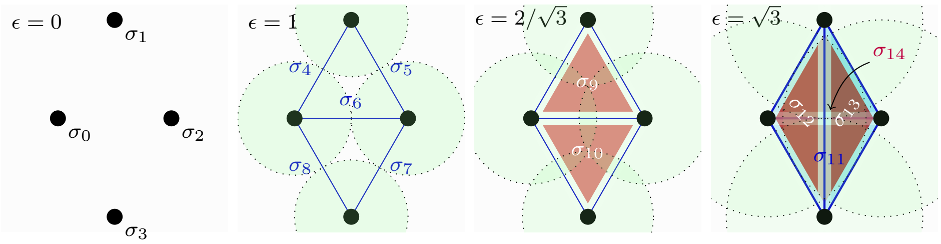


Figure 7: The example of a Čech-filtration from Figure 2, with simplices labeled by the time of appearance. If multiple simplices are added at the same scale, then the simplices are labeled according to their dimension, i.e. vertex before edge before triangle before tetrahedron.

The final simplicial complex (at the largest scale) contains 15 simplices in total, so the boundary matrix, denoted by ∂, would be of size 15-by-15. This is slightly too much to handle on a page, so consider that we were to stop directly after $\epsilon = 2/\sqrt{3}$, then the matrix is only of size 11-by-11. The elements of the matrix are given by

$$\partial_{i,j} = \begin{cases} 1, & \text{if } \sigma_i \text{ is in the boundary of } \sigma_j \\ 0, & \text{otherwise} \end{cases}$$

$$\partial = \begin{array}{c|ccccccccccc|}
 & \sigma_0 & \sigma_1 & \sigma_2 & \sigma_3 & \sigma_4 & \sigma_5 & \sigma_6 & \sigma_7 & \sigma_8 & \sigma_9 & \sigma_{10} \\
\sigma_0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\
\sigma_1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
\sigma_2 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\
\sigma_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
\sigma_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
\sigma_5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
\sigma_6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
\sigma_7 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
\sigma_8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
\sigma_9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\sigma_{10} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{array}$$

The matrix can be put into its ”reduced form” by a series of column additions going from left to right in order to produce an upper triangular matrix.

$$\tilde{\partial} = \begin{array}{c|ccccccccccc|}
 & & & & & & & & & +\sigma_4 & & \\
 & & & & & & & +\sigma_4 & & +\sigma_5 & & \\
 & & & & & & & +\sigma_5 & & +\sigma_7 & & \\
 & \sigma_0 & \sigma_1 & \sigma_2 & \sigma_3 & \sigma_4 & \sigma_5 & \sigma_6 & \sigma_7 & \sigma_8 & \sigma_9 & \sigma_{10} \\
\sigma_0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
\sigma_1 & 0 & 0 & 0 & 0 & \boxed{1} & 1 & 0 & 0 & 0 & 0 & 0 \\
\sigma_2 & 0 & 0 & 0 & 0 & 0 & \boxed{1} & 0 & 1 & 0 & 0 & 0 \\
\sigma_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \boxed{1} & 0 & 0 & 0 \\
\sigma_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
\sigma_5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
\sigma_6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \boxed{1} & 1 \\
\sigma_7 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
\sigma_8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \boxed{1} \\
\sigma_9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\sigma_{10} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{array}$$

Here, we see that the columns corresponding to simplices σ_4 and σ_5 have been added to the column corresponding to σ_6, thus creating a 1-cycle $\sigma_4 + \sigma_5 + \sigma_6$. This represents a new loop being formed when σ_6 is added to the simplex which already contains σ_4 and σ_5. The zeros in the column indicates the creation of a new homology class. Such columns are usually referred to as ”positive”, in the sense that they signal the creation (as opposed to destruction) of a homology class. Similarly, another homology class is created when σ_8 is added, here represented by the 1-cycle $\sigma_4 + \sigma_5 + \sigma_7 + \sigma_8$ (though this class could also be represented by the smaller loop $\sigma_6 + \sigma_7 + \sigma_8$). The first four columns similarly represent the creation of new homology classes, namely the creation of four connected components. Columns with nonzero elements are conversely called ”negative”, in the sense that they signal the destruction of a homology class. For example, the column corresponding to σ_4 signals the destruction of the connected component that was created by σ_1, by the addition of the edge σ_4 which connects σ_0 to σ_1. This can be read off by looking at the lowest nonzero element of the column (boxed in the matrix above) where the corresponding row indicates which simplex created the homology class. In fact each of the lowest nonzero element in the columns signals the destruction of a homology class created by the simplex of the corresponding row.

Persistence intervals

Some homology classes are never destroyed, in this case the connected component created when σ_0 was added. The information in the boundary matrix can be summarised in so called persistence intervals:

0-homology	1-homology
(connected components)	(loops)
$[\sigma_0, \infty) \xrightarrow{\epsilon} [0, \infty)$	
$[\sigma_1, \sigma_4) \xrightarrow{\epsilon} [0, 1)$	$[\sigma_6, \sigma_9\ ) \xrightarrow{\epsilon} [1, 2/\sqrt{3})$
$[\sigma_2, \sigma_5) \xrightarrow{\epsilon} [0, 1)$	$[\sigma_8, \sigma_{10}) \xrightarrow{\epsilon} [1, 2/\sqrt{3})$
$[\sigma_3, \sigma_7) \xrightarrow{\epsilon} [0, 1)$	

Table 1: Persistence intervals for the Čech-filtration from Figure 2 and 7, up to scale $\epsilon = 2/\sqrt{3}$. The intervals are represented first by the simplices that create and destray a homology class, which are later mapped onto the corresponding scale at which these simplices were added. The persistence intervals are also visualized in Figure 8.

2.4 Persistence diagrams, barcodes and other representations

Persistence diagram

By keeping track of the different homology classes in the filtration, the scale at which they appear (birth) and disappear (death) can be marked. By plotting each class in a graph with their birth- and death scales as coordinates, what is known as a persistence diagram is obtained. A persistence diagram is a multi-set (a set of objects where repetitiveness is allowed) of points in $\mathbb{R} \times (\mathbb{R} \cup \{+\infty\})$ of birth-death coordinates of the homology classes in a filtration. Since a homology class must appear before it disappear, it holds that $d_j \geq b_j$ for $j = 1, ..., \beta_K$, where d_j and b_j is the death and birth coordinate of a homology class and β_K is the K-th Betti number. By equip the set of all persistence diagrams, $\mathcal{D}$, with the p-Wasserstein distance or the bottleneck distance, $\mathcal{D}$ is turned into a metric space. This is necessary in order to compare the similarity between persistence diagrams. For any D_1, $D_2 \in \mathcal{D}$,

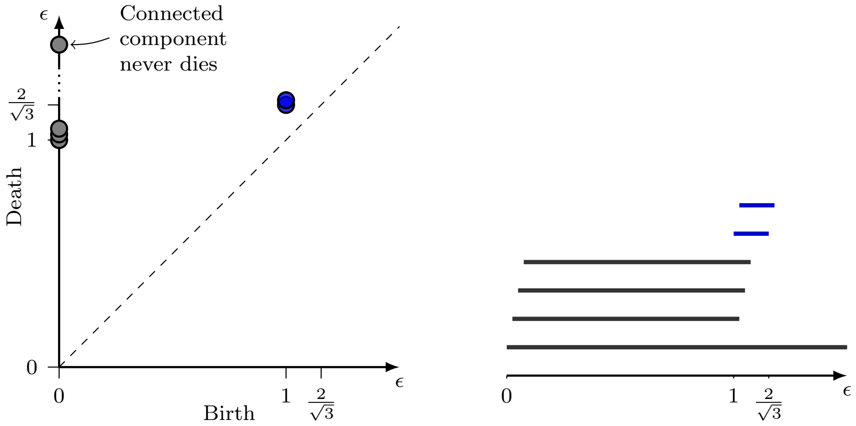


Figure 8: Persistence diagram (left) and barcode diagram (right) corresponding to the Čech-filtration from Figure 2 and 7, up to scale $\epsilon = 2/\sqrt{3}$. The points and bars are slightly displaced to indicate the internal ordering for simplices that are created at the same scale. The black/gray dots and bars represent the birth and death of the 0-homology classes (connected components), while the blue dots and bars represent the birth and death of 1-homology classes (loops).

Wasserstein distance

the p-Wasserstein distance, W_p, is defined as

$$W_p(D_1, D_2) = \inf_{\substack{\text{bijections} \\ \gamma: D_1 \longrightarrow D_2}} \left(\sum_{(b,d) \in D_1} ||(b,d) - \gamma(b,d)||_\infty^p \right)^{1/p},$$

Bottleneck distance

where $||\cdot||_\infty$ is the max-norm defined for any $(x, y) \in \mathbb{R}^2$ by $||(x, y)||_\infty = \max\{|x|, |y|\}$. By letting $p \longrightarrow \infty$, the bottleneck distance is obtained, defined by

$$W_\infty(D_1, D_2) = \inf_{\substack{\text{bijections} \\ \gamma: D_1 \longrightarrow D_2}} \sup_{(b,d) \in D_1} ||(b,d) - \gamma(b,d)||_\infty.$$

These two distance metrics are viable choices of metrics to describe the distance between persistence diagrams [9].

Persistence barcodes

A separate (but equivalent) representation of the persistence of homology classes is to represent them as stacked line segments, starting at their birth coordinate and ending at their death coordinate. These line segments are called persistence barcodes. A generic illustration of persistence barcodes and a persistence diagram is shown in Figure 9.

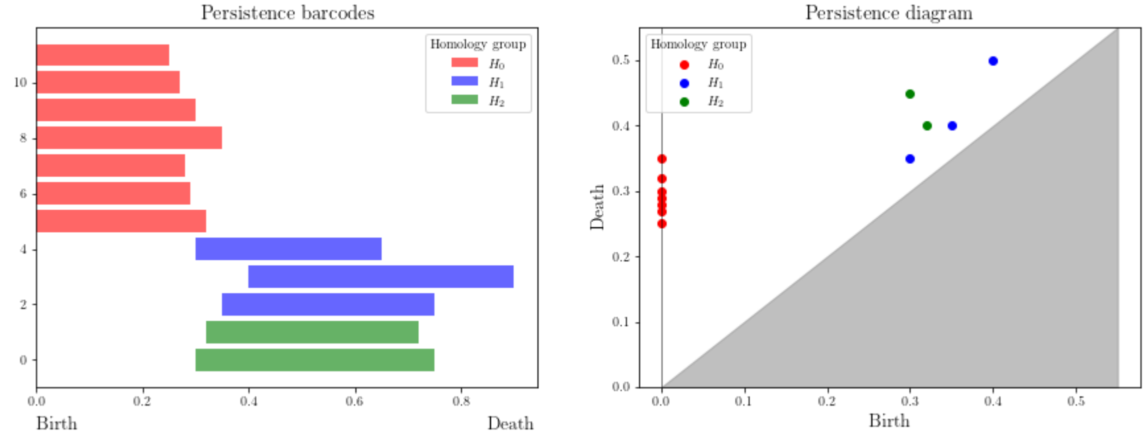


Figure 9: A generic illustration of persistence barcodes (left) and a persistence diagram (right).

2.5 Vectorizations of persistence diagrams

In order to make use of the data in persistence diagrams, they need to be transformed into a vectorization summary that summarizes a persistence diagram into a vectorized form. This vectorization can thereafter be used in machine learning algorithms. There exists several different methods to vectorize a persistence diagram and the following sections present some of the most common methods used [12].

2.5.1 Persistent entropy

The is a scalar measure of entropy for each homology class in a persistence diagram. The persistent entropy is an appropriate measure when the available data samples are small. This is because in such cases, univariate non-parametric tests are often required and thus summarizing the persistent homology group with a scalar measure is convenient. Persistent entropy is the Shannon entropy measured on a probability distribution given by persistent homology and is defined belowed [11].

Persistent entropy

Let persistent entropy $D_k = \{(b_i, d_i)\}_{i\in\beta_{||}}$ be a persistence diagram of a persistent homology group H_k where b_i and d_i being the birth and death coordinates of the different homology classes corresponding to H_k with β_k being the k-th Betti number. The persistent entropy is then given by

$$E(D_k) = -\sum_{i\in\beta_k} p_i \log(p_i), \quad \text{where} \quad p_i = \frac{(d_i - b_i)}{\sum_{i\in\beta_k}(d_i - b_i)}.$$

For K homology groups from the filtration, the resulting persistent entropy from the corresponding persistence diagram, D, is the vector of scalars $\mathbf{E}(D) = [E(D_1), E(D_2), ..., E(D_K)] \in \mathbb{R}^K$ that summarizes the data in the persistence diagram D. Due to its simpel form, this persistent entropy vector is a convenient input for a machine learning algorithm [11]. However, since all of the information of a persistence diagram of a homology group is reducted to a scalar representation, there is a significant risk of valuable information being lost. Furthermore, the interpretable connection to the original persistence diagram is also reduced. A different method that maintains this connection is persistence images.

2.5.2 Persistence image

The construction of a from a persistence diagram is a three stage process. Let $D_k = \{(b_i, d_i)\}_{i \in \beta_{||}}$ be a persistence diagram of the k-th homology group consisting of the birth and death coordinates of the corresponding different homology classes. First, let then $T : \mathbb{R}^2 \longrightarrow \mathbb{R}^2$ be a linear transformation defined as

$$T(x, y) = (x, y - x),$$

and when applied to the persistence diagram

$$T(D_k) = \{(b_i, d_i - b_i)\}_{i \in \beta_{||}}.$$

Secondly, let $g_\mu : \mathbb{R}^2 \longrightarrow \mathbb{R}$ be a probability distribution that is differentiable with mean $\mu \in (\mu_x, \mu_y)$. This distribution is usually set to be Gaussian kernel of mean $\mu = (\mu_x, \mu_y)$ and variance σ^2 given as

$$g_\mu(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{(x-\mu_x)^2 + (y-\mu_y)^2}{2\sigma^2}}.$$

Furthermore, a weight function, $f : \mathbb{R}^2 \longrightarrow \mathbb{R}$, needs to be defined with the condition of being non-negative, zero along the horizontal axis, continuous and piecewise differentiable. This, along with the Gaussian kernel, is used to transform the persistence diagram to a persistence surface, a scalar function over the $\mathbb{R}^2$-plane. The weight function is crucial to guarantee that the transformation is stable [4].

Persistent surface

Given persistent surface a persistence diagram D_k of the homology group H_k with the k-th Betti number, the persistence surface, $\rho_{D_k} : \mathbb{R}^2 \longrightarrow \mathbb{R}$, corresponding to D_k is defined as

$$\rho_{D_k}(z) = \sum_{\mu \in T(D_k)} f(\mu) g_\mu(z).$$

The final stage in the transformation from a persistence diagram to a persistence image is to discretize a relevant subdomain in the persistence surface such that the surface is reduced to a finite dimensional vector. The persistence surface is then integrated over each discretized region. This is done by selecting a grid in the plane of n pixels (boxes) of which the integral of the surface function over each region is assigned to that region [4].

Persistence image Given persistence image a persistence diagram D_k of the homology group H_k with the k-th Betti number and the corresponding persistence surface function ρ_{D_k}, its persistence image is the collection of pixels $I(\rho_{D_k})_p = \iint_p \rho_{D_k} dydx$, where p is the discritized region of the surface. Persistence images are appropriate to connect the persistence diagrams of different homology groups by concatenating the persistence image vectors. This concatenation can then be subject to a machine learning algorithm. In the construction of a persistence image, the choice of resolution needs to be specified which corresponds to the grid that is placed over the persistence image. The performance of classification from a persistence image is quite robust to the resolution choice [4].

3 Implementation

3.1 Converting to point clouds

As mentioned in Section 2.1 we must somehow transform the mass spectra to point clouds in order to analyse them topologically. We have three mass spectrums per molecule, and each mass spectrum is a discrete set of thresholded *fragment peaks* $\{(x, f(x)) : x \in X\}$ where $x \in (0, \infty)$ is mass to charge ratio and $f(x)$ is an instrument specific measured *intensity* $f(x) \in [0, 100]$. Since we are considering positive ionization only with adduct type [M+H]+ one can directly interpret the x value as the molecular mass of an ionized fragment of the input molecule.

It is possible to directly use the set $\{(x, f(x)) : x \in X\}$ as the point cloud, however this has significant practical and analytical consequences; firstly, since the masses are spread arbitrarily over $(0, \infty)$, a good constant step size for the filtration scale $\delta\epsilon$ would be hard to find. If $\delta\epsilon$ were large, then we would miss the small scale topology of the peaks which are close to each other. If $\delta\epsilon$ were small, then the computation would take exceedingly long because the filtration would have to sweep a very large interval. Secondly, since a mass spectrum is essentially a compounded measurement $f(x)$ at every pertinent mass x we have that it is impossible to have two points $(x_1, f(x_1)),\ (x_2, f(x_2))$

such that $x_1 = x_2$ but $f(x_1) \neq f(x_2)$. This constraint disallows all "pennon"-shaped (flag-shaped) 2-simplices to form.

To capture as many topological features of the ambient space as possible, we therefore ought to consider some transformation of the mass spectra which maintains the structure of the mass spectra but maps them to a smaller and more easily analysed form. To get an overview of the distribution of the fragment peaks in the dataset, the peaks of 1000 randomly sampled molecules were plotted and displayed in Figure 10. In this plot the intensity component has been ignored, and only the masses of the fragments were considered. The x-axis is the integer component of the mass, i.e. $\lfloor x \rfloor$ and the y-axis is the decimal component, i.e. $x - \lfloor x \rfloor$.

Some molecules in the dataset had very high molecular mass which means that they will have heavy fragments at low energy levels while most had fragments with mass around $[100, 400]$. The x-axis is logarithmic to display the entire range (about $(0, 6000]$ for the entire DrugBank) compactly in one plot.

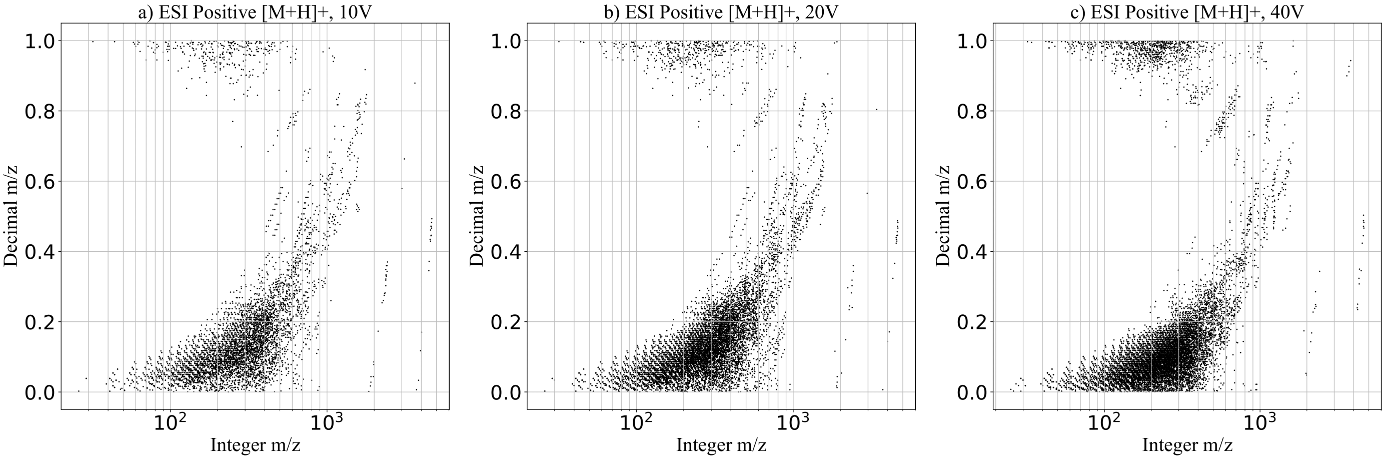


Figure 10: Simulated fragment masses of 1000 randomly sampled molecules from DrugBank. All mass spectra were simulated with CFM-ID 4.0. Each of a), b), c) contains the fragments of the same molecules at 10V, 20V, and 40V respectively. At all three energy levels we see that most of the fragments have integer masses in the range $[100, 400]$. This indicates that the sample contains fairly simple molecules with few atoms. We also see a clustering near 1.0 and near 0.0 in the decimal component, presumably because of the presence of many carbon and oxygen atoms in the molecules.

While the topological analysis itself is stable, the end result is extremely sensitive to any transformation that we choose. This is because transforming the point cloud induces changes to the topology of the ambient space. It is therefore crucial that the transformation does not mutate the information of the mass spectra in any significant way. We see in Figure 10 that the high energy level contains significantly more fragment peaks than the low and medium levels. Consequently, it is not straightforward how to incorporate the intensity information across the energy levels for a molecule into a single point cloud. We present two alternatives below.

3.1.1 Gaussian kernel transformation

A rudimentary mapping is to consider the intensity $f(x)$ at a specific x at the i'th energy level as the i'th coordinate. This would produce a 3D point cloud if missing peaks were treated as zero intensity, i.e. $f(x) = 0$. However, the mass spectral data X is a discrete set with very high resolution, so most points in such a point cloud would be on the coordinate axes. This has the effect that it will be very difficult for higher dimensional simplices to form in the filtration since they would only appear when the scale is large enough for edges to "cross" the axes.

A solution to this problem is to consider a Gaussian kernel with mean x at every peak, modeling each spectrum as a continuous Gaussian mixture. This is the same pre-processing method as in Heinonen et al. [3] (within a scaling factor). Effectively, this method "blends" nearby peaks, which can make the topology of the ambient space more visible after the point cloud transformation. An example of such a mixture can be seen in Figure 11. The appearance of the topological features is very sensitive to the parameterization used in the Gaussian kernel method. If the σ value used is too large then mass spectra will be indistinguishable, as all peaks will eventually blend. If the σ value used is too small, then the transformation is meaningless since the points will again all fall on the axes.

For an example of a point cloud where the mixture in Figure 11 was sampled with 1000 equispaced samples drawn from the sampling domain $[0, 500]$, see Figure 12 when $\sigma = 2$ was chosen and Figure 13 when $\sigma = 10$ was chosen. We see that many of the points are concentrated around the origin in Figure 12, which illustrates that the topology of the point cloud depends on the sampling domain. For example if we chose the sampling domain $[0, 10]$ then all points would be close to the origin. We can also see that the curvatures which appear in Figure 12 are blended together in Figure 13 when a larger value for σ is chosen.

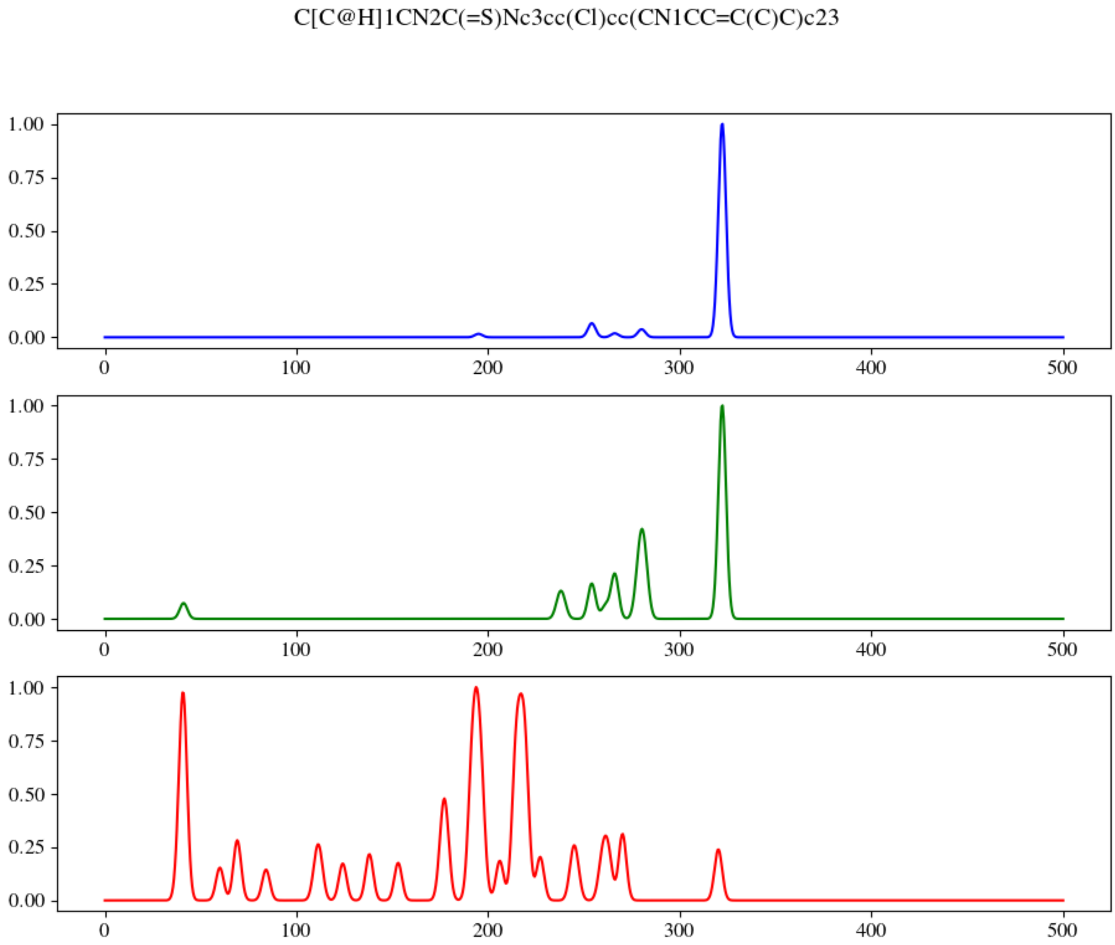


Figure 11: Gaussian mixture model of the three mass spectra for a random molecule in the DrugBank. The intensity of each peak has been normalized to the range [0,1]. Top (blue), middle (green), and bottom (red) are the low, medium and high energy levels respectively. Here $\sigma = 2$ for all energy levels.

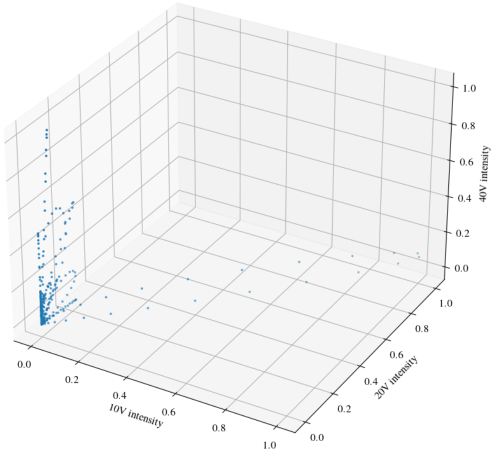


Figure 12: $\sigma = 2$

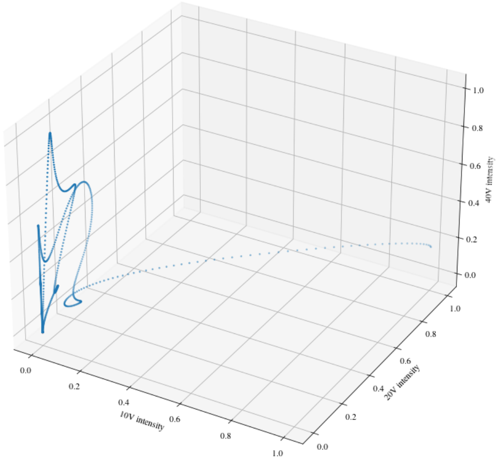


Figure 13: $\sigma = 10$

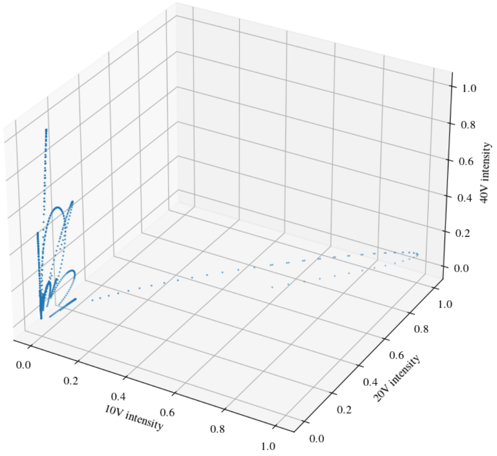


Figure 14: $\sigma = 2$, $\sigma_s = 2$

The sampling domain can also be dynamically chosen in order to avoid the concentration of the "non-informative" points near the origin. One way to do this is to sample within one σ_s around every peak, where σ_s is some arbitrary small positive value, as shown in Figure 14. Issues of the same nature which arose for σ also arise for σ_s.

Due to the ambiguity of this choice of σ, and additionally the ambiguity introduced by selecting a sampling domain, we decided to not use this transformation for our results. In Heinonen et al. [3] the value of σ was experimentally chosen. In our case we did not consider ourselves to have enough domain knowledge to decide on this parameter. Still, we include the Gaussian kernel transformation method in this section because it might be a useful transformation for non-thresholded "raw" mass spectral data, where the data resembles a Gaussian mixture more closely.

3.1.2 Integer vs decimal transformation

Another transformation that we considered is one similar to the method of plotting in Figure 10. Specifically, we used the integer vs decimal transformation given by

$$(x, f(x)) \mapsto (\log_{6000} \lfloor x \rfloor, x - \lfloor x \rfloor)$$

for all of our results. This transformation maps all mass spectra approximately to the space $[0, 1] \times [0, 1]$. We chose this transformation for our results because of its simplicity and due to the fact that it leverages the information in the decimal component of the x value which can be used to determine e.g. hydrogen count in the fragment molecule.

While practically simple and analytically intuitive, this method has some drawbacks. One drawback is that the intensity information is ignored completely. The group at AstraZeneca informed us that capturing the information stored in the decimal component of the peak was theoretically more important than the intensity of the peak. An immediate extension of this transformation could be to include the intensity information as a third coordinate (which we did not consider to save time when computing the Čech filtration). Another drawback is that it is not clear what information the Euclidean distance between points after this transformation should encode. For example, certain locations in the transformed ambient space are unreachable, such as "irrational masses", yet these locations still have valid Euclidean distances.

However, this method is also promising due to its amenability to extension which we discuss later in Section 5.2. A consequence of using this method is that one has to choose whether to consider the topology of the energy levels separately, or to first merge the point clouds. We chose to consider the topology at each energy level separately. However, the alternative choice could also be justified. On the cover page of this report we have displayed an illustration of a Čech filtration computed on a integer vs decimal transformed point cloud at a certain ϵ value where 2-simplices have been shaded in. Blue, green, and red indicate that the simplex belongs to the simplical complex of the low, medium, and high energy levels respectively.

3.2 Computing the Čech filtration and persistent homology

While the theory of topological data analysis is very involved, thankfully there exists various software packages which can compute persistence diagrams for generic point clouds. We opted to use `giotto` [13], which is a Python package dedicated to integrating TDA in the machine learning workflow by using the `scikit-learn` API. Some other alternatives are `javaPlex`, `Dionysus`, `Perseus`, `PHAT`, `DIPHA`, `GUDHI`, `Ripser`, and `TDAstats`. A comparison between these alternatives is available by Otter et al. [5]. A simplification of the `giotto` package is that only the persistence is accessible. For example, it is not clear how (if possible) only the filtration could be computed. However, we chose to use `giotto` because this simplification makes it easy to compare multiple vectorizations of persistence diagrams using its integration with the `scikit-learn` API. For low abstraction level computations such as extracting individual filtrations, the `GUDHI` [17] package in particular is useful; however its Python interface does not support the Čech complex. Another benefit of the `giotto` package is that it provides classes to compute persistence entropy and persistence images, given that the upper bound of the scale is fixed.

3.3 Vectorization of persistence diagrams

Once persistence diagrams have been computed for each mass spectrum (using `giotto`), a vectorized representation is needed in further steps of a regression/classification pipeline.

The theoretical background for a number of such representations is presented in Section 2.4. In this section we describe the implementations of these methods for mass spectra.

3.3.1 Persistence distances

Distances between persistence diagrams of the different molecules can be stored in a distance matrix. The similarity of the persistence diagrams can then be visualized using techniques like multi-dimensional scaling (MDS). Here, the pairwise distances are used to map the persistence diagrams to points in an abstract N-dimensional space, while preserving the distances as well as possible.

Computing the distances (on top of computing the filtrations and persistence diagrams) between molecules is computationally intensive. For this reason we at first only considered a random sample of $n = 100$ molecules. The resulting visualizations are shown in Figures 15, 16 and 17, using mass vs intensity, integer vs decimal mass and Gaussian kernel point cloud constructions respectively. Since the different distance metrics have varying scales, the coordinates were then centered and scaled by their variance. The points representing the persistence diagrams of the molecules (or rather their mass spectra) were colored according to its polarizability, as provided by our descriptors data set, in order to get a feeling about which point cloud construction, distance measure and homology dimension created the most easily distinguishable separation of this particular descriptor.

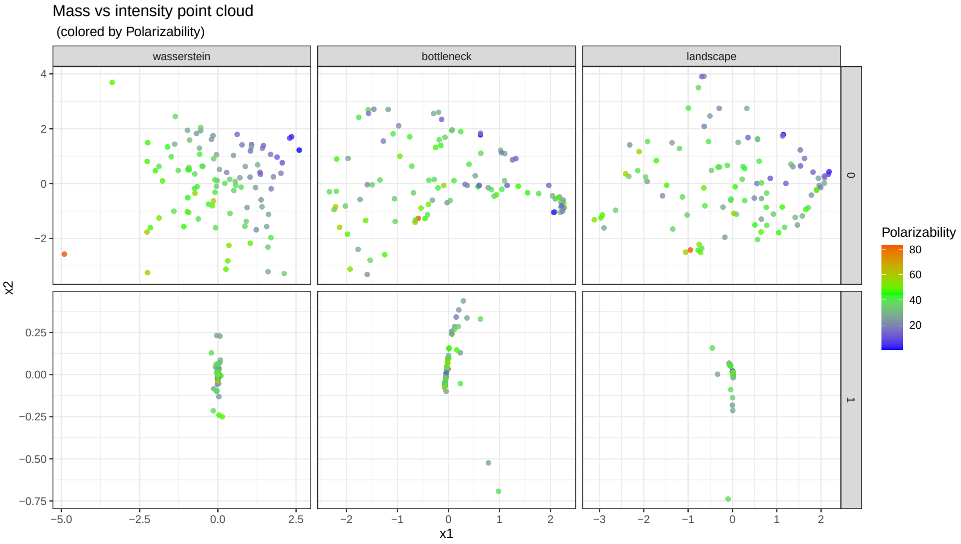


Figure 15: 2D MDS representation (scaled and centered) of distances between persistence diagrams of 100 randomly chosen molecules from our drugbank data set. Each column represents a particular choice of distance metric for homologies of dimension 0 (top row) and dimension 1 (bottom row). Each dot represents a molecule, colored by it's polarizability. In this figure we used the untransformed point cloud construction, i.e. using mass and intensity as coordinates.

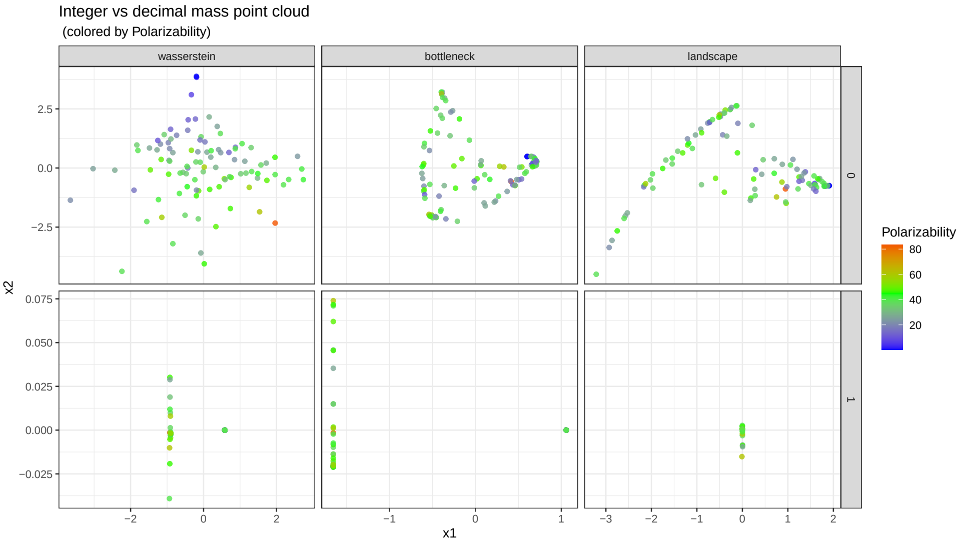


Figure 16: Same as Figure 15, but starting from a transformed point cloud using integer mass and decimal mass as coordinates.

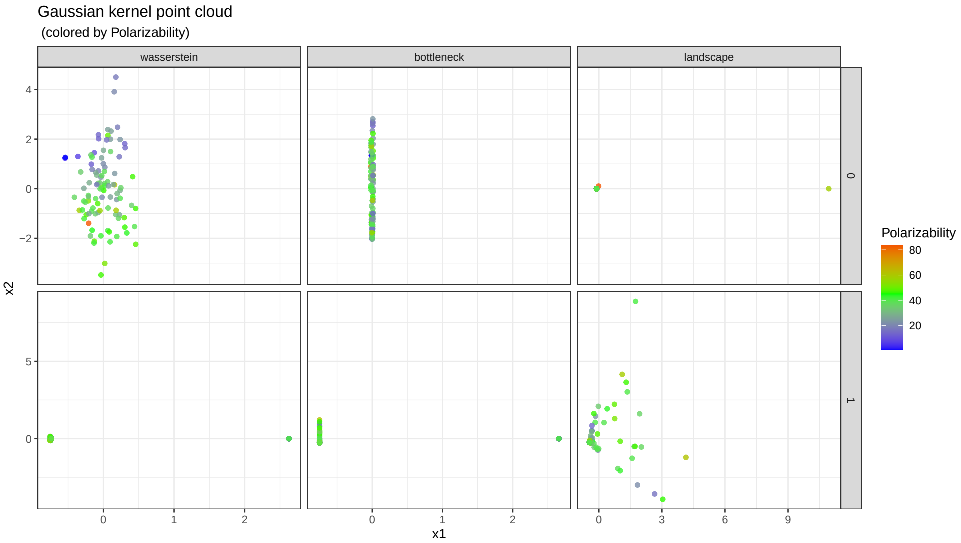


Figure 17: Same as Figure 15, but starting from a transformed point cloud using Gaussian kernels, and the intensity for the three energy levels for each mass value as coordinates.

In general, it appears that the 0-dimensional homologies (connected components) create the largest spread and apparent separation (at least for the polarizability descriptor), for each of the different point cloud constructions. Furthermore, the most consistently useful distance metric appears to be Wasserstein distance, while the Bottleneck and landscape distance created more varying patterns depending on the choice of point cloud constructions. While the more interesting shapes of the patterns could potentially carry some information, it did not appear to aid in the separation of the descriptors.

As for the different point cloud constructions, the original, untransformed point cloud (mass vs intensity) appears to create surprisingly good separation, in particular together with the Wasserstein distance, while the transformed, integer vs decimal mass, point cloud also worked well and produced an interesting pattern in conjunction with the landscape distance, which could indicate that it captures more interesting structure (the nature of which is difficult to infer at this point). While there was some hope that the Gaussian kernel transformation would allow more higher dimensional topological features to be captured, the spread produced from this point cloud was somewhat underwhelming.

3.3.2 Persistence entropy

In contrast to the persistence distances, which requires one to compute the distances pairwise between persistence diagrams and store them in a distance matrix (of size $n \times n$ where n is the number of molecules in the data set), the persistence entropy is a much more compact and computationally inexpensive representation. Each persistence diagram is represented by a $k+1$-dimensional vector, k being the largest homology dimension considered. Here we chose $k = 2$, i.e. we considered connected components, loops/holes and voids. In our data set, each molecule has 3 mass spectra corresponding three different energy levels. Thus for each molecule we have 3 persistence diagrams, each represented by a 3-dimensional vector (the persistence entropy).

Visualisations of the 3-components of the persistent entropy (connected components, loops and voids) are provided in the Figures 18-20, and 21-23, where each dot represents molecule and it's coordinates are given by the values of a specific homolgy component of the persistent entropy for the three energy values used in the mass spectra. Interestingly, this rather simple representation produces quite a nice separation of the descriptors aromatic ring count and polarizability.

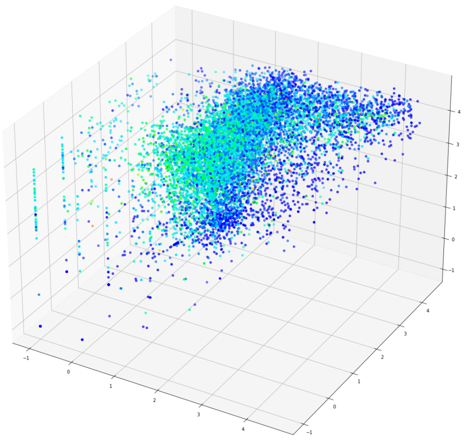


Figure 18: Aromatic ring count. Connected components.

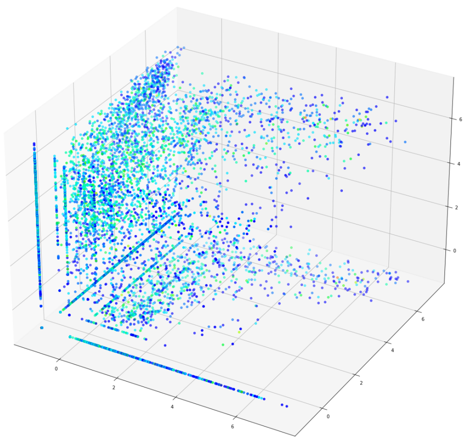


Figure 19: Aromatic ring count. Loops.

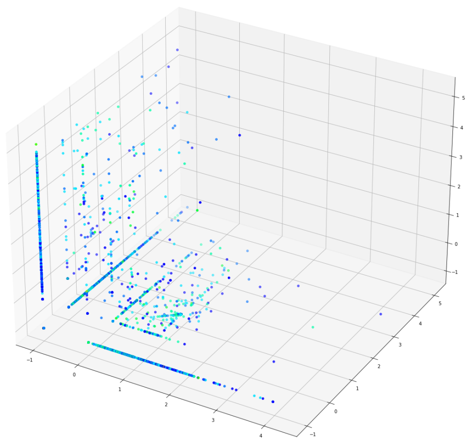


Figure 20: Aromatic ring count. Voids.

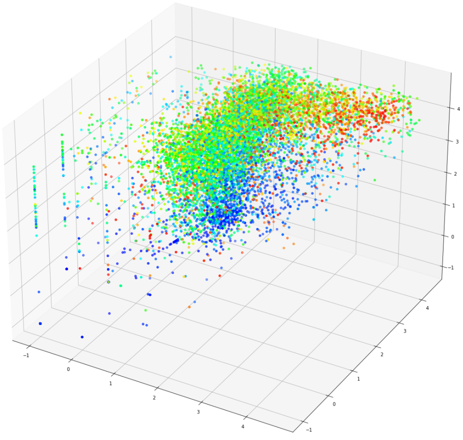


Figure 21: Polarizability. Connected components.

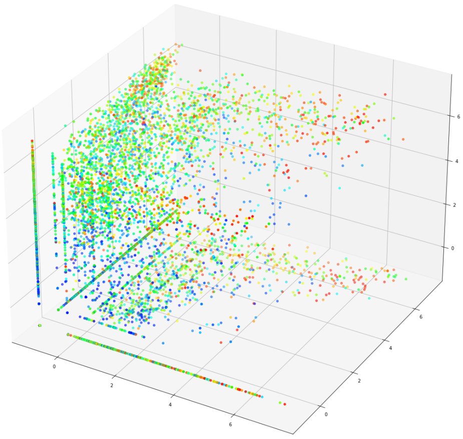


Figure 22: Polarizability. Loops.

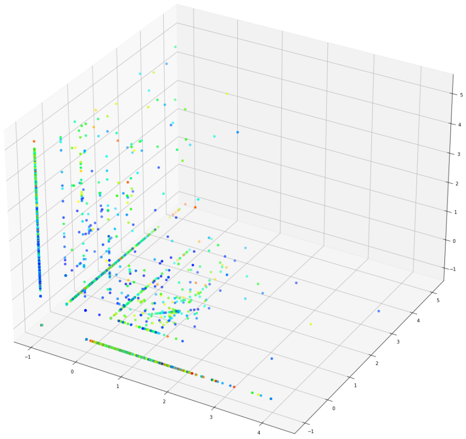


Figure 23: Polarizability. Voids.

4 Results

Out of the 9701 molecules in the DrugBank, 80% SMILES strings were randomly sampled without replacement into a training data set while the remaining 20% SMILES strings were compiled into a testing data set. From the group at AstraZeneca we also received a data set consisting of 126 chemical descriptors for each SMILES string in the DrugBank. These descriptors were used as response variables for training two statistical models. A subset of 74 response variables were selected and ordered into descriptor vectors. Only 74 were selected because they were numeric and had few missing values. Some of these descriptors were duplicates, for example there are four descriptors for molecular weight. These duplicates were kept for crossreferencing. Additionally, we include the results of another experiment where only the OEselma descriptors from the descriptor data set were considered. This has the effect that there will be no duplicate descriptors in the descriptor vectors since all were taken from the same database. The SMILES records which had missing descriptor values were excluded from the training and testing data sets. In this section we present the experimental results of two regression models trained on two different vectorizations of persistence diagrams. First, we considered an ordinary least squares linear regression model, trained on persistence entropy vectors. Second, we considered a deep neural network, trained on persistence images.

4.1 Ordinary Least Squares Linear Regression

The mass spectral data were first transformed into integer vs decimal point clouds as described in Section 3.1. The three lowest homology dimensions of the Čech filtration of each point cloud at each energy level was considered. Since there were three energy levels we obtain 9-dimensional vectors $\mathbf{x}_i \in \mathbb{R}^9$ by concatenating the persistence entropy vectors which were obtained from the persistence diagrams of the filtrations using the method described in Section 2.5.1.

A multi-target ordinary least squares model (OLS) was used to predict the descriptor response vector by using the persistence entropy vectors $\mathbf{x}_i$ as regressors and the descriptor vectors $\mathbf{y}_i$ as regressands. A statistical summary of the predicted values for the OEselma descriptors ($\mathbf{y}_i \in \mathbb{R}^{39}$) can be seen in Table 2, for a prediction of all the descriptor values ($\mathbf{y}_i \in \mathbb{R}^{74}$) see Appendix A. In order to get a summarised evaluation of the fit of the OLS model, the mean absolute error (MAE) was computed for each descriptor. The coefficient of determination of the OLS model was also considered. The rows in Table 2 are sorted in ascending order by the score statistic $\overline{|\mathbf{y}_d - \mathbf{f}_d|}/\sigma$ in the final column. The purpose of this ordering is to provide a numerical comparison between the natural variability of the descriptor value in the DrugBank and the model's predictive capability on the testing data set. When the score is close to zero, then the prediction is very good because the MAE is near zero. When the score of a descriptor is larger than one, then model is performing worse than a Gaussian model for that descriptor.

Over the 39 OEselma descriptors, the average score of the OLS model was $\overline{\overline{|\mathbf{y}_d - \mathbf{f}_d|}/\sigma} = 0.530632$. It had a coefficient of determination $r^2 = 0.212279$ with an average descriptor MAE equal to $\overline{\overline{|\mathbf{y}_d - \mathbf{f}_d|}} = 16.176768$. Over all 74 descriptors, the average score of the OLS model was $\overline{\overline{|\mathbf{y}_d - \mathbf{f}_d|}/\sigma} = 0.545283$. It had a coefficient of determination $r^2 = 0.217311$ with an average descriptor MAE of $\overline{\overline{|\mathbf{y}_d - \mathbf{f}_d|}} = 19.298790$.

We also tried to fit an OLS model with the persistence entropy vectors yielded from the Vietoris-Rips filtrations; however, the model had similar average descriptor MAE but extremely poor coefficients of determination. On the OEselma descriptors the OLS model had $r^2 = 0.098014$ with average descriptor MAE of $\overline{\overline{|\mathbf{y}_d - \mathbf{f}_d|}} = 19.330404$ and on all of the descriptors the OLS model had $r^2 = 0.092627$ with average descriptor MAE of $\overline{\overline{|\mathbf{y}_d - \mathbf{f}_d|}} = 22.896634$. For this reason, we did not consider the Vietoris-Rips filtrations in the CNN model and omit tables of this experiment from this report.

Descriptors	σ	μ	$\overline{\mathbf{f}_d}$	$\overline{\lvert\mathbf{y}_d - \mathbf{f}_d\rvert}$	Score
OEselma Silicon Count	0.108439	0.004239	0.000530	0.003220	0.029690
OEselma Bromine Count	0.265308	0.034856	0.032839	0.067323	0.253755
OEselma Iodine Count	0.287843	0.028450	0.043962	0.074060	0.257294
OEselma Tsa	261.688962	414.138860	416.314089	106.106942	0.405470
OEselma Rot Bond Count	7.213711	6.681677	6.712394	2.944185	0.408137
OEselma Mol Volume 2D	371.986946	592.330629	590.853288	157.692542	0.423920
OEselma Atom Count	15.766603	25.429675	25.634004	7.037158	0.446333
OEselma Mw	227.821937	373.328639	374.298888	104.350362	0.458035
OEselma Bond Count	16.980448	26.935092	27.305614	7.864136	0.463129
OEselma Npsa	200.207940	307.642466	312.725652	92.760671	0.463322
OEselma Carbon Count	11.377305	17.636929	17.933263	5.364070	0.471471
OEselma Phosphorous Count	0.474829	0.122751	0.123941	0.224827	0.473491
OEselma Part Flex Chain	4.183248	4.829146	4.997881	2.108171	0.503955
OEselma Psa	101.197251	106.494461	103.579013	52.274907	0.516564
OEselma Ertl Tpsa	94.357347	102.704204	100.456871	49.051269	0.519846
OEselma Polar Count	5.731492	6.059067	5.917903	3.004033	0.524128
OEselma Hba Lipinski	5.725688	6.789826	6.721928	3.011111	0.525895
OEselma Fluorine Count	0.933018	0.276967	0.297140	0.495503	0.531076
OEselma Rigid Frag Count	3.579486	5.023363	5.022775	2.000506	0.558881
OEselma Hbd	3.482418	2.674517	2.719809	1.948421	0.559502
OEselma Hbd Lipinski	3.595447	2.940744	2.995763	2.017538	0.561137
OEselma Oxygen Count	4.226225	4.206029	4.072034	2.389511	0.565401
OEselma Hba	4.799591	5.812529	5.852754	2.729655	0.568727
OEselma Max Flex Chain 1	2.645862	2.943565	3.036547	1.515677	0.572848
OEselma Max Flex Chain 2	1.948082	1.691916	1.710275	1.116195	0.572972
OEselma Max Flex Chain 3	1.623217	1.148954	1.129237	0.932263	0.574331
OEselma Clorine Count	0.515279	0.168629	0.181144	0.298271	0.578853
OEselma Neg Ioniz	1.108872	0.539802	0.462394	0.650857	0.586955
OEselma Nitrogen Count	2.861234	2.583797	2.649894	1.709653	0.597523
OEselma Rigid Bond Count	10.405612	16.273575	16.576271	6.257895	0.601396
OEselma Nonpolar Count	9.445993	12.086764	12.570445	5.846501	0.618940
OEselma Pos Ioniz	1.323273	1.011211	1.046081	0.821974	0.621167
OEselma Ring Count	1.811629	2.602073	2.675847	1.149428	0.634472
OEselma Halogen Count	1.127074	0.508902	0.555085	0.739378	0.656016
OEselma Max Rigid Chain	4.269220	6.990578	7.165254	2.869323	0.672095
OEselma Sulphur Count	0.642512	0.289119	0.291843	0.442843	0.689237
OEselma Nonpolar Count Per Mw	0.020729	0.031652	0.033239	0.015105	0.728698
OEselma Aromatic Ring Count	1.351165	1.607348	1.683263	1.001529	0.741233
OEselma Polar Count Per Mw	0.009170	0.016298	0.016026	0.006958	0.758783

Table 2: Summary of OLS predicted OEselma descriptor values. The notation $\mathbf{y}_d$ refers to the dth component of vector $\mathbf{y}_i$. The first two columns are the standard deviation and the average value of the descriptors over all molecules in the DrugBank. The third column is the average predicted value of the descriptor for the molecules in the test data set. The fourth column is the mean absolute error (MAE) between true and predicted and the last column is the score.

4.2 Regression Convolutional Neural Network

We also ran an experiment where we used persistence images computed from Čech filtrations of the integer vs decimal point clouds to predict molecular descriptor values using a convolutional neural network (CNN). Three homology dimensions were considered in the Čech filtrations. Since we have three mass spectra for each molecule we obtain nine persistence images per molecule. The union of the persistence images yielded from the TDA of the mass spectra for each molecule can be interpreted as a multispectral image with nine channels $\mathbf{x}_i \in \mathbb{R}^{50} \times \mathbb{R}^{50} \times \mathbb{R}^{9}$. The experiment consisted of sending these multispectral persistence images to a rudimentary CNN architecture, see Table 3, which was tasked to predict numeric molecular descriptors. The loss function used for training was the MAE loss function. The descriptor values were pre-processed with an invertible normalization function to the range $[0, 1]$. With these settings, the CNN model shown in Table 3 is a multi-target non-linear regression model with convolutional hidden layers.

Layer (type)	Output shape	N. Params	Activation
Normalization	(None, 9, 50,50)	101	-
2D Convolution	(None, 32, 48, 48)	2624	ReLu
2D Max pooling	(None, 16, 24, 48)	0	-
2D Convolution	(None, 12, 20, 32)	38432	ReLu
2D Max pooling	(None, 6, 10, 32)	0	-
Flatten	(None, 1920)	0	-
Dense	(None, 39)	74919	-

Table 3: CNN model used for regression. Each persistence image is a 50x50 grayscale image normalized by the normalization layer at the input. The input to the network are the multispectral persistence images for each homology dimension at each energy level. Both convolutional layers had 32 filters. The first convolutional layer had a kernel size of 3 and the second had a kernel size of 5.

During training the loss function is a moving average of the MAE over all descriptors, in this sense it is comparable to the average descriptor MAE of the OLS model. For a table of the results, see Table 4. For a graph of the training and validation loss moving average over the epochs, see Figure 24. Note that this is a graph of the mean absolute error of the normalized descriptor values. In this sense, it is only indirectly comparable to the average descriptor MAE of the OLS model after the predicted descriptor values have been transformed by the inverse of the normalization function. Considering this, the CNN model performed slightly better than the OLS model. On the OEselma descriptors the CNN model had an average descriptor MAE of $\overline{\overline{|\mathbf{y}_d - \mathbf{f}_d|}} = 18.756981$ with an average score of $\overline{|\mathbf{y}_d - \mathbf{f}_d|/\sigma} = 0.507713$ and on all of the descriptors the CNN model had an average descriptor MAE of $\overline{|\mathbf{y}_d - \mathbf{f}_d|} = 15.812311$ with an average score of $\overline{|\mathbf{y}_d - \mathbf{f}_d|/\sigma} = 0.529038$. It was not clear how to compute the coefficient of determination for the CNN regression model, which is why it has been omitted.

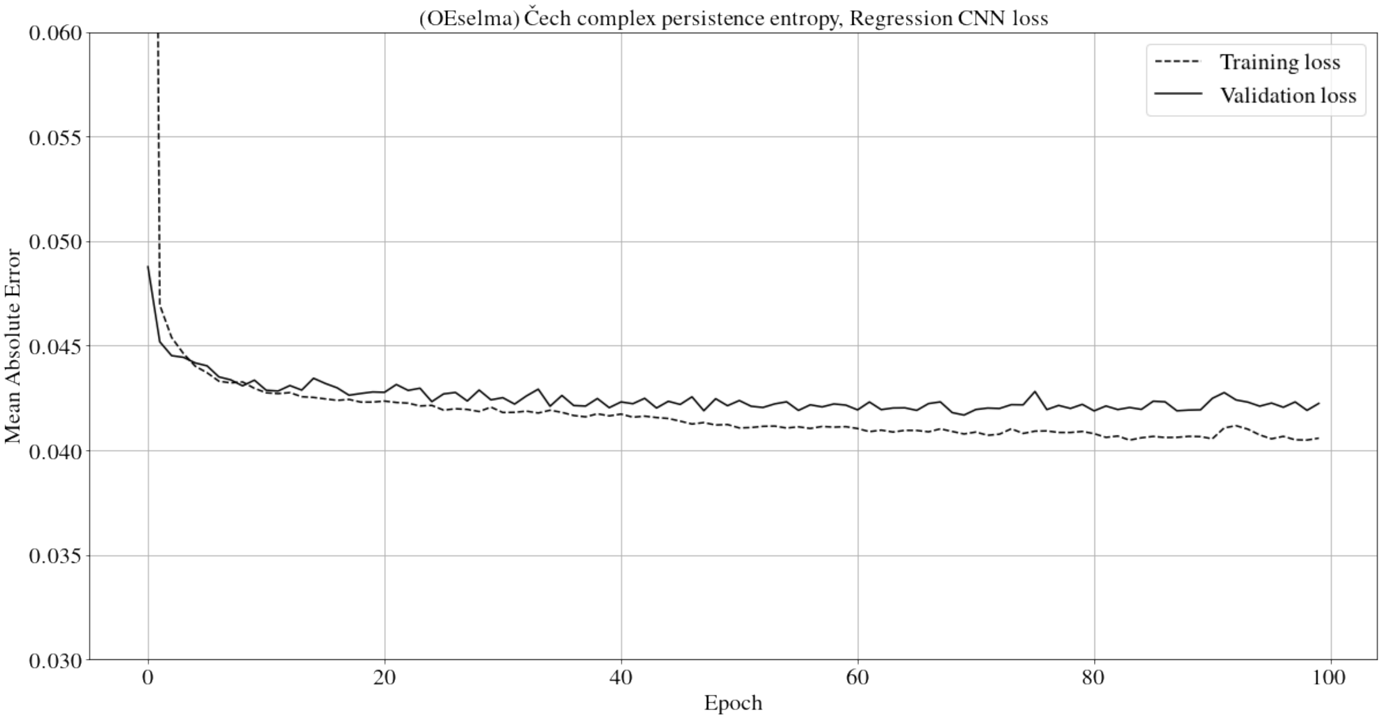


Figure 24: Graph of training loss and validation loss of the model given in Table 3. Training was conducted for 100 epochs. It is clear that at around 40 epochs the model starts overfitting because the validation loss remains stable around 0.0425.

Descriptors	σ	μ	$\overline{\mathbf{f}_d}$	$\overline{\lvert\mathbf{y}_d - \mathbf{f}_d\rvert}$	Score
OEselma Silicon Count	0.108439	0.004239	0.001059	0.001745	0.016093
OEselma Iodine Count	0.287843	0.028450	0.025953	0.029030	0.100853
OEselma Bromine Count	0.265308	0.034856	0.034428	0.036573	0.137850
OEselma Phosphorous Count	0.474829	0.122751	0.115466	0.118135	0.248795
OEselma Fluorine Count	0.933018	0.276967	0.262712	0.267226	0.286411
OEselma Clorine Count	0.515279	0.168629	0.160487	0.163946	0.318170
OEselma Neg Ioniz	1.108872	0.539802	0.452860	0.457865	0.412911
OEselma Halogen Count	1.127074	0.508902	0.483581	0.481674	0.427367
OEselma Sulphur Count	0.642512	0.289119	0.280720	0.288122	0.448431
OEselma Part Flex Chain	4.183248	4.829146	4.808263	1.967213	0.470260
OEselma Rot Bond Count	7.213711	6.681677	6.401483	3.489025	0.483666
OEselma Npsa	200.207940	307.642466	305.795517	98.622901	0.492602
OEselma Tsa	261.688962	414.138860	405.405720	131.697137	0.503258
OEselma Oxygen Count	4.226225	4.206029	3.956038	2.135425	0.505279
OEselma Hba	4.799591	5.812529	5.663136	2.456341	0.511781
OEselma Mw	227.821937	373.328639	362.211229	119.079400	0.522686
OEselma Hbd	3.482418	2.674517	2.534958	1.830414	0.525616
OEselma Ertl Tpsa	94.357347	102.704204	96.960223	50.082845	0.530778
OEselma Polar Count	5.731492	6.059067	5.700212	3.074848	0.536483
OEselma Psa	101.197251	106.494461	99.604419	54.736046	0.540885
OEselma Hbd Lipinski	3.595447	2.940744	2.795021	1.959633	0.545032
OEselma Mol Volume 2D	371.986946	592.330629	575.865486	204.456468	0.549633
OEselma Rigid Bond Count	10.405612	16.273575	16.369703	5.730955	0.550756
OEselma Carbon Count	11.377305	17.636929	17.591631	6.268767	0.550989
OEselma Hba Lipinski	5.725688	6.789826	6.522246	3.158857	0.551699
OEselma Nonpolar Count	9.445993	12.086764	12.375000	5.434184	0.575290
OEselma Ring Count	1.811629	2.602073	2.652542	1.084616	0.598697
OEselma Atom Count	15.766603	25.429675	25.002119	9.567522	0.606822
OEselma Max Rigid Chain	4.269220	6.990578	7.162076	2.596076	0.608091
OEselma Bond Count	16.980448	26.935092	26.652542	10.376671	0.611095
OEselma Rigid Frag Count	3.579486	5.023363	4.927436	2.196631	0.613672
OEselma Max Flex Chain 1	2.645862	2.943565	2.967691	1.632688	0.617072
OEselma Max Flex Chain 3	1.623217	1.148954	1.181674	1.041896	0.641871
OEselma Max Flex Chain 2	1.948082	1.691916	1.758475	1.251045	0.642193
OEselma Pos Ioniz	1.323273	1.011211	0.969280	0.850462	0.642695
OEselma Nitrogen Count	2.861234	2.583797	2.566208	1.887779	0.659778
OEselma Nonpolar Count Per Mw	0.020729	0.031652	0.033686	0.014295	0.689639
OEselma Aromatic Ring Count	1.351165	1.607348	1.655720	0.990535	0.733097
OEselma Polar Count Per Mw	0.009170	0.016298	0.015956	0.007268	0.792535

Table 4: Summary of CNN predicted OEselma descriptor values. The first two columns are the standard deviation and the average value of the descriptors over all molecules in the DrugBank. The third column is the average predicted value of the descriptor for the molecules in the test data set. The fourth column is the mean absolute error (MAE) between true and predicted and the last column is the score.

5 Discussion

5.1 Comparison between OLS and CNN model

When comparing the order of the descriptors (based on the score) in Table 2 and Table 4 using the Levenshtein distance (edit distance) we see that we would need 33 operations to get the same ordering. The Levenshtein distance from the order of either table to a random permutation of its rows is approximately 37.114 operations. From this we can conclude the orderings are quite different, even though both models seem to cluster certain descriptors together, such as the cluster of Iodine count, Silicon count, Bromine count and the cluster of Max Flex Chain 1,2,3. One interpretation of why the orderings are different could be that the OLS and the CNN model find different optimal solutions. This in turn might be because the OLS model minimizes the least squared error (l_2) while the CNN model uses a mean absolute error loss function (l_1).

MAE absolute difference	
Aromatic Ring Count	0.010994
Atom Count	2.530364
Bond Count	2.512535
Bromine Count	0.030750
Carbon Count	0.904697
Clorine Count	0.134325
Ertl Tpsa	1.031576
Fluorine Count	0.228277
Halogen Count	0.257704
Hba	0.273314
Hba Lipinski	0.147746
Hbd	0.118007
Hbd Lipinski	0.057905
Iodine Count	0.045030
Max Flex Chain 1	0.117011
Max Flex Chain 2	0.134850
Max Flex Chain 3	0.109633
Max Rigid Chain	0.273247
Mol Volume 2D	46.763926
Mw	14.729038
Neg Ioniz	0.192992
Nitrogen Count	0.178126
Nonpolar Count	0.412317
Nonpolar Count Per Mw	0.000810
Npsa	5.862230
Oxygen Count	0.254086
Part Flex Chain	0.140958
Phosphorous Count	0.106692
Polar Count	0.070815
Polar Count Per Mw	0.000310
Pos Ioniz	0.028488
Psa	2.461139
Rigid Bond Count	0.526940
Rigid Frag Count	0.196125
Ring Count	0.064812
Rot Bond Count	0.544840
Silicon Count	0.001475
Sulphur Count	0.154721
Tsa	25.590195

Table 5: Absolute difference between OLS MAE and CNN MAE for the OEselma descriptors.

When comparing the absolute difference of the MAE values between the two models, which is compiled to the right in Table 5. Most values in Table 5 are very close to zero with molecular volume, molecular weight and Tsa being notable outliers. These descriptors are also the OEselma descriptors with the highest natural variability in the DrugBank (see Table 2 or Table 4), which corresponds to them being outliers in the prediction. This suggests that the choice of model has low impact on the performance. Similarly, since the OLS model used persistence entropy as vectorization method and the CNN model used persistence images, the resemblance in the MAE trend also suggest that both of these vectorization methods captures persistence properties from the filtration of similar importance. An implication of this could be that the selection of vectorization method is less crucial for predicting descriptor values from persistence homology and that more emphasis should be on the construction of point clouds and filtration.

An interpretation of the variation in scores among the different descriptors is that the descriptors with a poor performance (high score) have only a weak connection to the molecules and their chemical structure. Since weak connections between molecules and a specific descriptor could still be of chemical interest, a possible application of these models can be to classify molecules based on their MAE for the different descriptors.

5.2 Further work

Topological data analysis is a young and rapidly developing field. It is still unclear for which type of data meaningful or useful topological features can be extracted. Most practice data sets consist of distorted or noisy samples from topological manifolds such as a sphere or a torus. Still, it has shown promise as a means of feature extraction even for data sets which are not inherently thought of as topological in nature, such as image [10], audio [15] and time series data [1]. However, for each non-trivial data set a series of choices must be made, requiring expertise both in the techniques of topological data analysis and in the subject matter, in this case chemistry, mass spectrometry and pharmacology. Being new to all of these fields, the main authors are not qualified to find the optimal way to apply topological data analysis to mass spectrometry data. Instead we have tried to outline the necessary steps and choices that need to be made, along with some attempts at implementations and analysis of their results. It is our hope that the explorative analysis in this report will be useful for other researchers who wish to apply TDA to mass spectrometry data.

That being said, we conjecture that the most crucial step is finding a suitable point cloud representation of mass spectra data. Ideally, one in which homology classes (connected components, loops, voids, etc) have a meaningful interpretation. Another criteria would be that the distances between points can be meaningfully defined. For instance, when using a point cloud representation with x- and y-coordinates given by the peak mass- and intensity values, a Euclidean distance function $\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$ does not make much sense, as the two coordinates have completely different units. Even for point clouds constructed from the integer and decimal parts of the peak mass values, a Euclidean distance is not well motivated. We suspect that a great deal of domain knowledge, familiarity with different mass spectra representations and creativity is needed to find an optimal point cloud construction. Similarly, domain knowledge of the descriptors might be needed in order to identify a good point cloud construction.

One possible extension of the integer vs decimal mass point cloud representation considered in this article could be to utilize the cyclic property of the decimal dimension. In our implementation, decimal coordinates of 0.99 and 0.01 are almost maximally separated, and thus a simplex between such points is only formed at very large scales. It could be more appropriate to represent the decimal coordinate axis as a circle (and thus the integer vs decimal space as a cylinder). This, however might require custom implementation of Cech and/or Vietoris-Rips filtrations, something which we deemed to not be feasible within the time constraints of this project.

6 Acknowledgments

We would like to thank Mats Josefson and Gustaf Hulthe for stimulating discussions and sharing their knowledge of mass spectrometry, chemistry, pharmacology and data science. We would also like to thank them for providing us with data and computing descriptors on AstraZenecas computer cluster.

References

1. Takens, F. in *Lecture Notes in Mathematics* 366–381 (Springer Berlin Heidelberg, 1981). doi:10.1007/bfb0091924. https://doi.org/10.1007/bfb0091924.
2. Edelsbrunner, H. & Harer, J. *Computational topology: an introduction* New ed. edition (American Mathematical Society, 2009).
3. Heinonen, M., Shen, H., Zamboni, N. & Rousu, J. Metabolite identification and molecular fingerprint prediction through machine learning. *Bioinformatics* **28,** 2333–2341. ISSN: 1367-4803. doi:10.1093/bioinformatics/bts437. eprint: https://academic.oup.com/bioinformatics/article-pdf/28/18/2333/700396/bts437.pdf. https://doi.org/10.1093/bioinformatics/bts437 (July 2012).
4. Adams, H. *et al.* Persistence images: A stable vector representation of persistent homology. **18,** 1–35 (Feb. 2017).
5. Otter, N., Porter, M. A., Tillmann, U., Grindrod, P. & Harrington, H. A. A roadmap for the computation of persistent homology. *EPJ Data Science* **6.** doi:10.1140/epjds/s13688-017-0109-5. https://doi.org/10.1140/epjds/s13688-017-0109-5 (Aug. 2017).
6. Wishart, D. S. *et al.* DrugBank 5.0: a major update to the DrugBank database for 2018. *Nucleic Acids Research* **46,** D1074–D1082. doi:10.1093/nar/gkx1037. https://doi.org/10.1093/nar/gkx1037 (Nov. 2017).
7. Chandrasekaran, B., Abed, S. N., Al-Attraqchi, O., Kuche, K. & Tekade, R. K. in *Dosage Form Design Parameters* (ed Tekade, R. K.) 731–755 (Academic Press, 2018). ISBN: 978-0-12-814421-3. doi:https://doi.org/10.1016/B978-0-12-814421-3.00021-X. https://www.sciencedirect.com/science/article/pii/B978012814421300021X.
8. Jin, W., Barzilay, R. & Jaakkola, T. *Junction tree variational autoencoder for molecular graph generation* in *International conference on machine learning* (2018), 2323–2332.
9. Chung, Y. & Lawson, A. Persistence Curves: A canonical framework for summarizing persistence diagrams. *CoRR* **abs/1904.07768.** arXiv: 1904.07768. http://arxiv.org/abs/1904.07768 (2019).
10. Garin, A. & Tauzin, G. *A Topological "Reading" Lesson: Classification of MNIST using TDA* 2019. eprint: arXiv:1910.08345.
11. Atienza, N., Gonzalez-Díaz, R. & Soriano-Trigueros, M. On the stability of persistent entropy and new summary functions for topological data analysis. *Pattern Recognition* **107,** 107509. ISSN: 0031-3203. doi:https://doi.org/10.1016/j.patcog.2020.107509. https://

www.sciencedirect.com/science/article/pii/S0031320320303125 (2020).

12. Fasy, B., Qin, Y., Summa, B. & Wenk, C. *Comparing Distance Metrics on Vectorized Persistence Summaries* in *NeurIPS 2020 Workshop on Topological Data Analysis and Beyond* (2020). https://openreview.net/forum?id=X1bxKJo5_qL.

13. Tauzin, G. *et al.* giotto-tda: A Topological Data Analysis Toolkit for Machine Learning and Data Exploration. eprint: arXiv:2004.02551 (2020).

14. Nanda, V. *Computational algebraic topology: Lecture notes* https://people.maths.ox.ac.uk/nanda/cat/TDANotes.pdf (2021).

15. Subramani, K. & Smaragdis, P. *Point Cloud Audio Processing* 2021. eprint: arXiv:2105.02469.

16. Wang, F. *et al.* CFM-ID 4.0: more accurate ESI-MS/MS spectral prediction and compound identification. *Analytical Chemistry* **93,** 11692–11700 (2021).

17. The GUDHI Project. *GUDHI User and Reference Manual* 3.5.0. https://gudhi.inria.fr/doc/3.5.0/ (GUDHI Editorial Board, 2022).

Appendices

A Ordinary Least Squares descriptor predictions

Descriptors	σ	μ	$\overline{\mathbf{f}_d}$	$\overline{\lvert\mathbf{y}_d - \mathbf{f}_d\rvert}$	Score
OEselma Silicon Count	0.108439	0.004239	0.000560	0.002336	0.021538
OEselma Iodine Count	0.287843	0.028450	0.039731	0.068471	0.237878
OEselma Bromine Count	0.265308	0.034856	0.033016	0.067484	0.254362
Mol Weight	301.265463	374.512520	371.636732	102.311704	0.339606
Polarizability	29.516618	37.783778	37.861987	10.196873	0.345462
Molar Refractivity	74.060489	95.559089	95.940680	25.766911	0.347917
MOE H Ema	9.691163	8.864515	8.293012	3.458272	0.356848
Tpsa	133.333536	107.810334	104.397905	50.274115	0.377055
OEselma Tsa	261.688962	414.138860	413.916620	104.243409	0.398349
Rotatable Bond Count	7.213711	6.681677	6.670397	2.915731	0.404193
OEselma Rot Bond Count	7.213711	6.681677	6.670397	2.915731	0.404193
Molecular Volume (2D)	371.986946	592.330629	587.763424	155.383329	0.417712
OEselma Mol Volume 2D	371.986946	592.330629	587.763424	155.383329	0.417712
MOE H Logs	5.975325	9.838051	9.752189	2.600139	0.435146
OEselma Atom Count	15.766603	25.429675	25.508674	6.920609	0.438941
Heavy Atom Count	15.757548	25.426522	25.506995	6.919384	0.439116
OEselma Mw	227.821937	373.328639	371.636766	102.316792	0.449109
Molecular Weight	227.821937	373.328639	371.636766	102.316792	0.449109
Exact Mass	227.548842	372.828687	371.268124	102.207349	0.449167
OEselma Bond Count	16.980448	26.935092	27.180750	7.750208	0.456420
Npsa	200.207940	307.642466	311.829856	91.812098	0.458584
OEselma Npsa	200.207940	307.642466	311.829856	91.812098	0.458584
OEselma Phosphorous Count	0.474829	0.122751	0.123111	0.221552	0.466593
MOE H Emd	10.821464	12.580101	12.460378	5.054848	0.467113
OEselma Carbon Count	11.377305	17.636929	17.901511	5.320611	0.467651
MOE H Emd C	5.020204	6.492526	6.514657	2.400257	0.478119
Psa	101.197251	106.494461	102.077127	50.681520	0.500819
OEselma Psa	101.197251	106.494461	102.077127	50.681520	0.500819
OEselma Part Flex Chain	4.183248	4.829146	4.982652	2.105559	0.503331
Ertl Tpsa	94.357347	102.704204	99.096475	47.579564	0.504249
OEselma Ertl Tpsa	94.357347	102.704204	99.096475	47.579564	0.504249
OEselma Polar Count	5.731492	6.059067	5.834359	2.927670	0.510804
OEselma Hba Lipinski	5.725688	6.789826	6.635702	2.928607	0.511486
OEselma Fluorine Count	0.933018	0.276967	0.304421	0.500981	0.536947
OEselma Rigid Frag Count	3.579486	5.023363	4.985450	1.952665	0.545516
OEselma Hbd Lipinski	3.595447	2.940744	2.945719	1.962018	0.545695
OEselma Hbd	3.482418	2.674517	2.675993	1.900881	0.545851
MOE H Log Pbo	3.788166	1.932621	2.141476	2.082669	0.549783
OEselma Oxygen Count	4.226225	4.206029	4.003358	2.338538	0.553340
OEselma Hba	4.799591	5.812529	5.787913	2.668917	0.556072
Azlogd74	0.652538	0.729417	0.712565	0.371994	0.570072
OEselma Max Flex Chain 2	1.948082	1.691916	1.711807	1.115499	0.572614
OEselma Max Flex Chain 1	2.645862	2.943565	3.026861	1.517344	0.573478

Table 6: Summary of OLS descriptor predictions using persistence entropy vectors obtained from the Čech filtrations as regressors and all 74 descriptors as regressands.

Descriptors	σ	μ	$\overline{\mathbf{f}_d}$	$\overline{\lvert\mathbf{y}_d - \mathbf{f}_d\rvert}$	Score
OEselma Max Flex Chain 3	1.623217	1.148954	1.130946	0.931127	0.573630
OEselma Neg Ioniz	1.108872	0.539802	0.452714	0.641812	0.578798
OEselma Clorine Count	0.515279	0.168629	0.182988	0.301226	0.584589
MOE H Log Dbo	3.066570	-3.836500	-3.840843	1.803478	0.588109
OEselma Nitrogen Count	2.861234	2.583797	2.632345	1.690824	0.590942
OEselma Rigid Bond Count	10.405612	16.273575	16.537213	6.225522	0.598285
Chromlogd	0.649936	0.806016	0.787190	0.389351	0.599060
MOE H Mr	0.829737	0.327136	0.322575	0.508297	0.612600
OEselma Nonpolar Count	9.445993	12.086764	12.627308	5.805089	0.614556
OEselma Pos Ioniz	1.323273	1.011211	1.043089	0.820734	0.620230
Ring Count	1.811629	2.602073	2.672076	1.148862	0.634159
OEselma Ring Count	1.811629	2.602073	2.672076	1.148862	0.634159
OEselma Halogen Count	1.127074	0.508902	0.560157	0.743283	0.659480
MOE H Logp	4.622931	6.583703	6.695206	3.073117	0.664755
Scscore	1.067509	3.272166	3.360497	0.709769	0.664883
OEselma Max Rigid Chain	4.269220	6.990578	7.162283	2.855099	0.668764
OEselma Sulphur Count	0.642512	0.289119	0.282037	0.434317	0.675966
Alogp	2.869296	2.202912	2.420230	1.940984	0.676467
Clogp	3.447059	1.655366	1.896323	2.360496	0.684785
Azlogd74	3.226066	0.608286	0.807755	2.214187	0.686343
Chromlogd	3.779478	1.046476	1.259415	2.635566	0.697336
Acd Descriptors	3.002862	2.055326	2.073000	2.170762	0.722898
Acd Logd-Logp	3.001606	2.053207	2.073000	2.170762	0.723200
OEselma Nonpolar Count Per Mw	0.020729	0.031652	0.033535	0.015019	0.724558
OEselma Aromatic Ring Count	1.351165	1.607348	1.691662	0.998422	0.738934
Azlogd74 (Nn)	0.201104	0.628486	0.641651	0.148690	0.739368
OEselma Polar Count Per Mw	0.009170	0.016298	0.015981	0.006915	0.754109
Solubility Dd Class	0.055386	0.945031	0.947198	0.043259	0.781053
Solubility Dd Class	0.263879	0.577961	0.578341	0.219617	0.832267
Azlogd74 (Nn)	1.561973	1.637720	1.663844	1.307334	0.836976
Clogp	24.318937	20.365517	17.612759	21.111785	0.868121

Table 7: (continued) Summary of OLS descriptor predictions.

B Regression Convolutional Neural Network descriptor predictions

Descriptors	σ	μ	$\overline{\mathbf{f}_d}$	$\overline{\lvert\mathbf{y}_d - \mathbf{f}_d\rvert}$	Score
OEselma Silicon Count	0.108439	0.004239	0.000555	0.001018	0.009389
OEselma Iodine Count	0.287843	0.028450	0.022198	0.024962	0.086720
OEselma Bromine Count	0.265308	0.034856	0.032186	0.034367	0.129535
Polarizability	29.516618	37.783778	37.008772	7.637840	0.258764
Molar Refractivity	74.060489	95.559089	93.688508	19.201618	0.259269
OEselma Phosphorous Count	0.474829	0.122751	0.115427	0.124269	0.261713
Mol Weight	301.265463	374.512520	361.148176	79.184431	0.262839
OEselma Tsa	261.688962	414.138860	404.644284	73.496467	0.280854
OEselma Fluorine Count	0.933018	0.276967	0.263041	0.267660	0.286875
Molecular Volume (2D)	371.986946	592.330629	574.792197	116.410129	0.312941
OEselma Mol Volume 2D	371.986946	592.330629	574.792197	116.957915	0.314414
Tpsa	133.333536	107.810334	102.129887	43.337402	0.325030
OEselma Clorine Count	0.515279	0.168629	0.164817	0.168333	0.326682
MOE H Logs	5.975325	9.838051	9.536647	1.979592	0.331295
MOE H Ema	9.691163	8.864515	8.089200	3.263566	0.336757
Exact Mass	227.548842	372.828687	360.794522	78.060796	0.343051
Molecular Weight	227.821937	373.328639	361.147114	78.930490	0.346457
OEselma Mw	227.821937	373.328639	361.147114	78.960558	0.346589
MOE H Mr	0.829737	0.327136	0.287532	0.293337	0.353530
OEselma Atom Count	15.766603	25.429675	24.970033	5.891237	0.373653
Heavy Atom Count	15.757548	25.426522	24.965039	5.900726	0.374470
OEselma Npsa	200.207940	307.642466	305.190059	77.675248	0.387973
Npsa	200.207940	307.642466	305.190059	78.049286	0.389841
OEselma Bond Count	16.980448	26.935092	26.614317	6.772029	0.398813
OEselma Carbon Count	11.377305	17.636929	17.576027	4.617088	0.405816
OEselma Neg Ioniz	1.108872	0.539802	0.449501	0.456129	0.411345
MOE H Emd C	5.020204	6.492526	6.465247	2.084232	0.415169
OEselma Halogen Count	1.127074	0.508902	0.482242	0.488284	0.433231
MOE H Emd	10.821464	12.580101	12.200716	4.775285	0.441279
OEselma Sulphur Count	0.642512	0.289119	0.281354	0.284716	0.443129
Rotatable Bond Count	7.213711	6.681677	6.396226	3.213759	0.445507
OEselma Rot Bond Count	7.213711	6.681677	6.396226	3.237916	0.448856
OEselma Polar Count	5.731492	6.059067	5.694229	2.605236	0.454548
OEselma Hba Lipinski	5.725688	6.789826	6.508324	2.666210	0.465658

Table 8: Summary of CNN descriptor predictions using persistence persistence images obtained from the Čech filtrations as regressors and all 74 descriptors as regressands.

Descriptors	σ	μ	$\overline{\mathbf{f}_d}$	$\overline{\lvert\mathbf{y}_d - \mathbf{f}_d\rvert}$	Score
Psa	101.197251	106.494461	99.447049	48.143539	0.475740
OEselma Psa	101.197251	106.494461	99.447049	48.275914	0.477048
Ertl Tpsa	94.357347	102.704204	96.811948	45.700685	0.484336
OEselma Hba	4.799591	5.812529	5.653163	2.333508	0.486189
OEselma Ertl Tpsa	94.357347	102.704204	96.811948	45.930540	0.486772
OEselma Part Flex Chain	4.183248	4.829146	4.809101	2.050328	0.490128
OEselma Oxygen Count	4.226225	4.206029	3.945616	2.151151	0.509001
OEselma Rigid Bond Count	10.405612	16.273575	16.345172	5.463328	0.525037
OEselma Hbd	3.482418	2.674517	2.533851	1.872711	0.537762
OEselma Hbd Lipinski	3.595447	2.940744	2.791343	1.963677	0.546156
OEselma Rigid Frag Count	3.579486	5.023363	4.922309	1.958462	0.547135
OEselma Nonpolar Count	9.445993	12.086764	12.364040	5.202900	0.550805
OEselma Ring Count	1.811629	2.602073	2.644284	1.028168	0.567538
Ring Count	1.811629	2.602073	2.644284	1.028746	0.567857
OEselma Max Rigid Chain	4.269220	6.990578	7.139290	2.555314	0.598544
Scscore	1.067509	3.272166	3.343528	0.640171	0.599687
Azlogd74	0.652538	0.729417	0.707592	0.408652	0.626249
OEselma Pos Ioniz	1.323273	1.011211	0.964484	0.836916	0.632459
OEselma Nitrogen Count	2.861234	2.583797	2.562708	1.831008	0.639937
OEselma Max Flex Chain 3	1.623217	1.148954	1.185905	1.058848	0.652315
OEselma Max Flex Chain 1	2.645862	2.943565	2.970588	1.754023	0.662931
OEselma Nonpolar Count Per Mw	0.020729	0.031652	0.033783	0.013793	0.665400
Chromlogd	3.779478	1.046476	1.190575	2.562367	0.677968
OEselma Max Flex Chain 2	1.948082	1.691916	1.760821	1.327690	0.681537
OEselma Aromatic Ring Count	1.351165	1.607348	1.664262	0.954071	0.706110
MOE H Logp	4.622931	6.583703	6.624866	3.326548	0.719575
Azlogd74 (Nn)	0.201104	0.628486	0.634556	0.148485	0.738349
Clogp	24.318937	20.365517	18.705882	18.090256	0.743875
Solubility Dd Class	0.055386	0.945031	0.946397	0.041758	0.753955
Chromlogd	0.649936	0.806016	0.785463	0.513032	0.789357
Solubility Dd Class	0.263879	0.577961	0.575560	0.219416	0.831503
Azlogd74 (Nn)	1.561973	1.637720	1.666410	1.321256	0.845889
OEselma Polar Count Per Mw	0.009170	0.016298	0.015985	0.008131	0.886679
Acd Descriptors	3.002862	2.055326	2.037895	2.728629	0.908676
Acd Logd-Logp	3.001606	2.053207	2.037895	2.749274	0.915934
MOE H Log Pbo	3.788166	1.932621	2.101887	3.523169	0.930046
MOE H Log Dbo	3.066570	-3.836500	-3.752318	2.987153	0.974102
Alogp	2.869296	2.202912	2.363989	3.059571	1.066314
Clogp	3.447059	1.655366	1.878574	3.698270	1.072877
Azlogd74	3.226066	0.608286	0.770768	3.597431	1.115114

Table 9: (continued) Summary of CNN descriptor predictions.

Layer (type)	Output shape	N. Params	Activation
Normalization	(None, 9, 50,50)	101	-
2D Convolution	(None, 32, 48, 48)	2624	ReLu
2D Max pooling	(None, 16, 24, 48)	0	-
2D Convolution	(None, 12, 20, 32)	38432	ReLu
2D Max pooling	(None, 6, 10, 32)	0	-
Flatten	(None, 1920)	0	-
Dense	(None, 74)	142154	-

Table 10: CNN model used for regression. Each persistence image is a 50x50 grayscale image normalized by the normalization layer at the input. The input to the network are the multispectral persistence images for each homology dimension at each energy level. Both convolutional layers had 32 filters. The first convolutional layer had a kernel size of 3 and the second had a kernel size of 5.

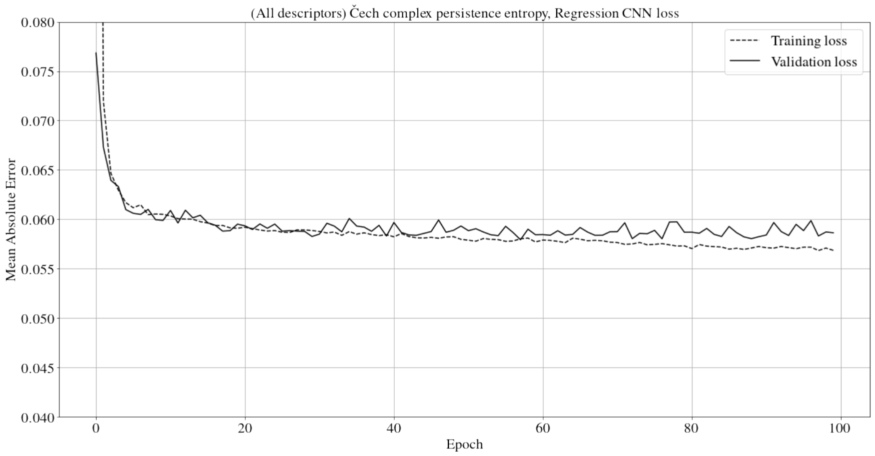


Figure 25: Graph of training loss and validation loss of the model given in Table 10. Training was conducted for 100 epochs. It is clear that at around 40 epochs the model starts overfitting because the validation loss remains stable around 0.060.

C Some group theoretic concepts

Group

A group, G, is a set together with a binary operation[2], $+$, with the following properties: i) the group includes a unit element, $e \in G$, such that $e + g = g + e = g$, $\forall g \in G$. ii) the group is closed under the operation, i.e. $g_1 + g_2 \in G$, $\forall g_1, g_2 \in G$. iii) for every element, $g \in G$, there exists also an inverse

[2] Usually this operation is called group multiplication and is denoted by $*$ or $\circ$, however, since we shall be dealing mostly with abelian groups under addition, we use the symbol for addition already here.

element, $g^{-1} \in G$, such that $g + g^{-1} = g^{-1} + g = e$.

Abelian groups

Finitely generated Abelian groups

Rank of a group

In what follows we will only consider Abelian groups, for which the operation is commutative, i.e. $g_1 + g_2 = g_2 + g_1$, $\forall g_1, g_2 \in G$. More specifically, we shall deal with finitely generated Abelian groups. These have a finite set of basis elements, or generators, v_i, $i = 1, 2, \ldots, n$, and all other elements of the group can be written as combinations of these, $g = \sum_j \alpha_j v_j$, with coefficients in some field[3], $\alpha_j \in \mathbb{F}$. The group operation is thus defined by the operation of the underlying field. For example, the group of integers $\mathbb{Z} = \{\ldots, -2, -1, 0, 1, 2, \ldots\}$, under addition, can be constructed from the unit element $e = 0$ and one generator, $v = 1$, with coefficients, α, in $\mathbb{Z}$. In other words, any element of the group can be written as $g = \alpha 1$, $\alpha \in \mathbb{Z}$. Any generated Abelian group with a single generator is isomorphic to it's underlying field $\mathbb{F}$. Furthermore, any finitely generated abelian group with n generators is isomorphic[4] to $\mathbb{F}^n = \mathbb{F} \oplus \mathbb{F} \oplus \ldots \oplus \mathbb{F}$, where $\oplus$ denotes the direct sum[5]. The rank of a group, $\operatorname{rank} G$, refers to the number of (independent) generators.

Zero group

Note that a group may consist only of the unit element, and no generators. For instance, the set $\{0\}$ together with addition forms a group. Since $0+0 = 0$, it is closed and has an inverse. This type of group is usually referred to as the zero group or the trivial group. In what follows we shall simply denote it by 0 and from context it should be clear when the zero group is meant.

Subgroup

Coset

A subgroup, H, is a subset of the group, $H \subseteq G$, which is itself a group, i.e. it contains the unit element, is closed under the operation of the group, and contains the inverse of every element. A (left) coset with respect to a subgroup is obtained by adding an element $g \in G$ to each element of H, i.e $g + H$. A subgroup is called a *normal subgroup* if the left coset equals the right coset, i.e. if $g + H = H + g$, $\forall g \in G$. For Abelian (commutative) groups, all subgroups are normal subgroups. Adding two different elements g_1 and g_2 to the (normal) subgroup can result in the same coset; $g_1 + H = g_2 + H$, thus imposing an equivalence relation between the elements $g_1 \sim g_2$. Thus, the cosets constitute a partition of the group into equivalence classes.

Quotient group

The set of cosets (with respect to a normal subgroup) is itself a group, known as the quotient group, $G/H = \{H, \tilde{g}_1 + H, \tilde{g}_2 + H, \ldots\}$ (usually pronounced G mod H). Here $\tilde{g}_i$ denotes a representative element of the equivalence class as-

[3] A field is a set which is a group both with respect to addition as well as multiplication

[4] Two groups are said to be isomorphic if there is a one–to-one correspondence between their elements that also preserves the group structure.

[5] A direct sum, $G \oplus H$, of two groups G and H, is the set of tuples $\{(g, h) : \ g \in G, \ h \in H\}$ together with a componentwise binary operation (for each component the binary operation is given by the corresponding group operation).

sociated with the coset. Sometimes the elements of the quotient group are directly represented by these representative elements, i.e $G/H = \{0, \tilde{g}_1, \tilde{g}_2, \ldots\}$.

To let this sink in, consider the group of integers $\mathbb{Z}$, and the subgroup consisting of the even integers, denoted $2\mathbb{Z}$ (obtained by multiplying each element in $\mathbb{Z}$ by 2). Since $2\mathbb{Z}$ is a subgroup of $\mathbb{Z}$ the cosets constitute a partition of the integers into equivalence classes. It should be clear that there are two unique cosets; $0 + 2\mathbb{Z}$ (or $2 + 2\mathbb{Z}$, or $4 + 2\mathbb{Z}$, or $\ldots$ depending of the choice of representative element), and $1 + 2\mathbb{Z}$ (or $3 + 2\mathbb{Z}$, or $5 + 2\mathbb{Z}$, or $\ldots$). Thus the equivalence classes correspond to odd and even integers. The elements of the quotient group $\mathbb{Z}/2\mathbb{Z}$ can be represented by the representative elements 0 and 1. The induced operation (from the original group operation of addition) of the quotient group can be seen to be addition modulo two. Thus, the quotient group $\mathbb{Z}/2\mathbb{Z}$ is isomorphic to the cyclic group $\mathbb{Z}_2$, defined by the set $\{0, 1\}$ under addition modulo 2. This group consists of a unit element, 0, and one generator, 1, and thus $\operatorname{rank} \mathbb{Z}_2 = 1$.