entropy

MDPI

Article

Modelling and Recognition of Protein Contact Networks by Multiple Kernel Learning and Dissimilarity Representations

Alessio Martino [1,*], Enrico De Santis [1], Alessandro Giuliani [2] and Antonello Rizzi [1]

1 Department of Information Engineering, Electronics and Telecommunications, University of Rome "La Sapienza", Via Eudossiana 18, 00184 Rome, Italy; enrico.desantis@uniroma1.it (E.D.S.); antonello.rizzi@uniroma1.it (A.R.)
2 Department of Environment and Health, Istituto Superiore di Sanità, Viale Regina Elena 299, 00161 Rome, Italy; alessandro.giuliani@iss.it
* Correspondence: alessio.martino@uniroma1.it; Tel.: +39-06-44585745

Received: 27 June 2020; Accepted: 17 July 2020; Published: 21 July 2020

Abstract: Multiple kernel learning is a paradigm which employs a properly constructed chain of kernel functions able to simultaneously analyse different data or different representations of the same data. In this paper, we propose an hybrid classification system based on a linear combination of multiple kernels defined over multiple dissimilarity spaces. The core of the training procedure is the joint optimisation of kernel weights and representatives selection in the dissimilarity spaces. This equips the system with a two-fold knowledge discovery phase: by analysing the weights, it is possible to check which representations are more suitable for solving the classification problem, whereas the pivotal patterns selected as representatives can give further insights on the modelled system, possibly with the help of field-experts. The proposed classification system is tested on real proteomic data in order to predict proteins' functional role starting from their folded structure: specifically, a set of eight representations are drawn from the graph-based protein folded description. The proposed multiple kernel-based system has also been benchmarked against a clustering-based classification system also able to exploit multiple dissimilarities simultaneously. Computational results show remarkable classification capabilities and the knowledge discovery analysis is in line with current biological knowledge, suggesting the reliability of the proposed system.

Keywords: dissimilarity spaces; support vector machines; kernel methods; computational biology; systems biology; protein contact networks

1. Introduction

Dealing with structured data is an evergreen challenge in pattern recognition and machine learning. Indeed, many real-world systems can effectively be described by structured domains such as networks (e.g., images [1,2]) or sequences (e.g., signatures [3]). Biology is a seminal field in which many complex systems can be described by networks [4], as the biologically relevant information resides in the interaction among constituting elements: common examples include protein contact networks [5,6], metabolic networks [7] and protein–protein interaction networks [8,9].

Pattern recognition in structured domains poses additional challenges as many structured domains are non-metric in nature (namely, the pairwise dissimilarities in such domains might not satisfy the four properties of a metric: non-negativity, symmetry, identity, triangle inequality) and patterns may lack any geometrical interpretation [10].

In order to deal with such domains, five mainstream approaches can be pursued [10]:

1. Feature generation and/or feature engineering, where numerical features are extracted ad-hoc from structured patterns (e.g., using their properties or via measurements) and can be further merged according to different strategies (e.g., in a multi-modal way [11]);
2. Ad-hoc dissimilarities in the input space, where custom dissimilarity measures are designed in order to process structured patterns directly in the input domain without moving towards Euclidean (or metric) spaces. Common—possibly parametric—edit distances include the Levenshtein distance [12] for sequence domains and graph edit distances [13] for graphs domains;
3. Embedding via information granulation and granular computing [3,14–25];
4. Dissimilarity representations [26–28], where structured patterns are embedded in the Euclidean space according to their pairwise dissimilarities;
5. Kernel methods, where the mapping between the original input space and the Euclidean space exploits positive-definite kernel functions [29–33].

This paper proposes a novel classification system based on an hybridisation of the latter two strategies: while dissimilarity representations see the (structured) patterns according to the pairwise dissimilarities, kernel methods encode pairwise similarities. Nonetheless, the class of properly-defined kernel functions is restricted: the (conditionally) positive definitiveness may not hold in case of non-metric (dis)similarities. The use of kernel methods in state-of-the-art (non-linear) classifiers such as Support Vector Machines (SVM) [34,35] is strictly related to their (conditionally) positive definitiveness due to the quadratic programming optimisation involved: indeed, non-(conditionally) positive definite kernels do not guarantee convergence to the global optimum. Although there is some research about learning from indefinite kernels (see, e.g., [36–40]), their evaluation on the top of Euclidean spaces (e.g., dissimilarity spaces) retain the (conditionally) positive definitiveness, devoting matrix regularisation or other tricks to foster positive definitiveness.

The proposed classification system is able to simultaneously explore multiple dissimilarities following a multiple kernel learning approach, where each kernel considers a different (dissimilarity) representation. The relative importance of the several kernels involved is automatically determined via genetic optimisation in order to maximise the classifier performance. Further, the very same genetic optimisation is in charge of determining a suitable subset of representative (prototypes) patterns in the dissimilarity space [27] in order to shrink the modelling complexity. Hence, the proposed system allows a two-fold a posteriori knowledge discovery phase:

1. By analysing the kernel weights, one can determine the most suitable representation(s) for the problem at hand;
2. The patterns elected as representatives for the dissimilarity space (hence determined as pivotal for tracking the decision boundary amongst the problem-related classes) can give some further insights for the problem at hand.

In order to validate the proposed classification system, a bioinformatics-related application is considered, namely protein function prediction. Proteins' 3D structure (both tertiary and quaternary) can effectively be modelled by a network, namely the so-called Protein Contact Network (PCN) [5]. A PCN is a minimalistic (unweighted and undirected) graph-based protein representation where nodes correspond to amino-acids and edges between two nodes exist whether the Euclidean distance between residues' α-carbon atom coordinates is within $[4, 8]$Å. The lower bound is defined in order to discard trivial connections due to closeness along the backbone (first-order neighbour contacts), whereas the upper bound is defined by considering the peptide bonds geometry (indeed, 8Å roughly correspond to two van der Waals radii between residues' α-carbon atoms [41]). It is worth stressing that both nodes labels (i.e., the type of amino-acid) and edges labels (i.e., the distance between neighbour residues) are deliberately discarded in order to focus only on proteins' topological configuration. Despite the minimalistic representation, PCNs have been successfully used in pattern recognition problems for tasks such as solubility prediction/folding propensity [42,43] and physiological role prediction [44–46];

furthermore, their structural and dynamical properties have been extensively studied in works such as [47–50].

In order to investigate how the protein function is related to its topological structure, a subset of the entire Escherichia coli bacterium proteome, correspondent to E. coli proteins whose 3D structure is known, is considered. The problem itself is cast into a supervised pattern recognition task, where each pattern (protein) is described according to eight different representations drawn by its PCN and its respective Enzyme Commission (EC) number [51] that serves as the ground-truth class label. The EC nomenclature scheme classifies enzymes according to the chemical reaction they catalyse and a generic entry is composed by four numbers separated by periods. The first digit (1–6) indicates one of the six major enzymatic groups (EC 1: oxidoreductases; EC 2: transferases; EC 3: hydrolases; EC 4: lyases; EC 5: isomerases; EC 6: ligases) and the latter three numbers represent a progressively finer functional enzyme classification. In this work, only the first number is considered. However, proteins with no enzymatic characteristics (or proteins for which enzymatic characteristics are still unknown nowadays) are not provided with an EC number, thus an additional class of not-enzymes will be considered, identified by the categorical label 7. It is worth noting that the EC classification only loosely relates to global protein 3D configuration, given that structure is affected by many determinants other than catalysed reactions like solubility, localisation in the cell, interaction with other proteins and so forth. This makes the classification task intrinsically very difficult.

This paper is organised as follows: Section 2 overviews some theory related to kernel methods and dissimilarity spaces; Section 3 presents the proposed methodology; Section 4 shows the results obtained with the proposed approach, along with a comparison against a clustering-based classifier (also able to explore multiple dissimilarities), and we also provide some remarks on the two-fold knowledge discovery phase. Finally, Section 5 concludes the paper. The paper also features two appendices: Appendix A describes in detail the several representations used for describing PCNs, whereas Appendix B lists the proteins selected as prototypes for the dissimilarity representations.

2. Theoretical Background

Let $\mathcal{D} = \{x_1, \ldots, x_{N_P}\}$ be the dataset at hand lying in a given input space $\mathcal{X}$. Moving the problem towards a dissimilarity space [26] consists in expressing each pattern from $\mathcal{D}$ according to the pairwise distances with respect to all other patterns, including itself. In other words, the dataset is cast into the pairwise distance matrix $\mathbf{D} \in \mathbb{R}^{N_P \times N_P}$ defined as:

$$\mathbf{D}_{i,j} = d(x_i, x_j) \qquad \forall i, j = 1, \ldots, N_P\,, \tag{1}$$

where $d(\cdot,\cdot)$ is a suitable dissimilarity measure in $\mathcal{D}$, that is $d : \mathcal{D} \times \mathcal{D} \to \mathbb{R}$. Without loss of generality, hereinafter let us consider $\mathbf{D}$ to be symmetric: if $d(\cdot,\cdot)$ is at least symmetric, $\mathbf{D}$ is trivially symmetric; in case of asymmetric dissimilarity measures, $\mathbf{D}$ can be 'forced' to be symmetric, e.g., $\mathbf{D} := \frac{1}{2}(\mathbf{D} + \mathbf{D}^T)$. The major advantage in moving the problem from a generic input space $\mathcal{X}$ towards $\mathbb{R}^{N_P \times N_P}$ is that the latter can be equipped with algebraic structures such as the inner product or the Minkowski distance, whereas the former might not be metric altogether. As such, in the latter, standard computational intelligence and machine learning techniques can be used without alterations [10]. On the negative side, the explicit evaluation of $\mathbf{D}$ can be computationally expensive as it leads to a time and space complexity of $\mathcal{O}(N_P^2)$. To this end, in [27], a 'reduced' dissimilarity space representation is proposed, where a subset of prototype patterns $\mathcal{R} \subset \mathcal{D}$ is properly chosen and each pattern is described according to the pairwise distances with respect to the prototypes only. This leads to the definition of a 'reduced' pairwise distance matrix $\bar{\mathbf{D}} \in \mathbb{R}^{N_P \times |\mathcal{R}|}$ defined as:

$$\bar{\mathbf{D}}_{i,j} = d(x_i, x_j) \qquad \forall i = 1, \ldots, N_P,\ \forall j = 1, \ldots, |\mathcal{R}|. \tag{2}$$

Since usually $|\mathcal{R}| < |\mathcal{D}|$, there is no need to solve a quadratic complexity problem such as evaluating Equation (1). On the negative side, however, the selection of the subset $\mathcal{R}$ is a delicate and challenging task [10] since:

1. They must well-characterize the decision boundary between patterns in the input space;
2. The fewer, the better: the number of representatives has a major impact on the model complexity (cf. Equation (1) vs. Equation (2)).

Several heuristics have been proposed in the literature, ranging from clustering the input space to (possibly class-aware) random selection [10,27,52].

Kernel methods are usually employed whether the input space has an underlying Euclidean geometry. Indeed, the simplest kernel (namely, the linear kernel [30,53]) is the plain inner product between real-valued vectors. The kernel matrix $\mathbf{K}$ (also known as the Gram matrix) can easily be defined as:

$$\mathbf{K}_{i,j} = \langle x_i, x_j \rangle \qquad \forall i,j = 1,\dots,N_P. \tag{3}$$

Let K be a symmetric and positive semi-definite kernel function from the input space $\mathcal{X}$ towards $\mathbb{R}$, that is $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ such that

$$K(x_i, x_j) = K(x_j, x_i) \qquad \forall x_i, x_j \in \mathcal{X} \tag{4}$$

$$\sum_{i=1}^{N_P}\sum_{j=1}^{N_P} c_i c_j K(x_i, x_j) \geq 0 \qquad \forall c_i, c_j \in \mathbb{R}, \forall x_i, x_j \in \mathcal{X}. \tag{5}$$

As in the linear kernel case, starting from pairwise kernel evaluations, one can easily evaluate the kernel matrix as

$$\mathbf{K}_{i,j} = K(x_i, x_j) \qquad \forall i,j = 1,\dots,N_P \tag{6}$$

and if $\mathbf{K}$ is a positive semi-definite kernel matrix, then K is a positive semi-definite kernel function. One of the most intriguing kernel methods property relies on the so-called kernel trick [29,30]: kernel of the form Equations (4) and (5) are also known as Mercer's kernel as they satisfy the Mercer condition [32]. Such kernel functions can be seen as the inner product evaluation on a high-dimensional (or possibly infinite-dimensional) and usually unknown Hilbert space $\mathcal{H}$. The kernel trick is usually described by the following, seminal, equation:

$$K(x, y) = \langle \phi(x), \phi(y) \rangle_{\mathcal{H}}, \tag{7}$$

where $\phi : \mathcal{X} \to \mathcal{H}$ is the implicit (and usually unknown) mapping function. The need for using a non-linear and higher-dimensional mapping is a direct consequence of Cover's theorem [33]. Thanks to the kernel trick, one can use one of the many kernel functions available (e.g., polynomial, Gaussian, radial basis function) in order to perform such non-linear and higher-dimensional mapping without knowing and explicitly evaluating the mapping function $\phi(\cdot)$. Further, kernel methods can be used in many state-of-the-art classifiers such as (kernelised) SVM [35,54].
In multiple kernel learning, the kernel matrix $\mathbf{K}$ is defined as a properly-defined combination of a given number of N_K kernels. The most intuitive combination is a linear combination of the form:

$$\mathbf{K} = \sum_{i=1}^{N_K} \beta_i \mathbf{K}^{(i)}, \tag{8}$$

where sub-kernels $\mathbf{K}^{(i)}$ are single Mercer's kernels. The weights β_i can be learned according to different strategies and can be constrained in several ways—see, e.g., [55–61], or the survey [62]. The rationale behind using a multiple kernel learning with respect to a plain single kernel learning depends on the application: for example, if data come from different sources, one might want to explore such different sources according to several kernels or, dually, one might want to explore the same data using different kernels, where such different kernels may differ in shape and/or type. In this work, a mixture between the two approaches is pursued: same source (PCN), but different representations (see Appendix A). Further, a linear convex combination of radial basis function kernels is employed. The ith radial basis function kernel is defined as

$$\mathbf{K}_{j,k}^{(i)} = \exp\left\{-\gamma_i \cdot \|x_j - x_k\|^2\right\} \qquad \forall j,k = 1,\dots,N_P \tag{9}$$

and γ_i is its shape parameter. Further, the weights β_i are constrained as

$$\sum_{i=1}^{N_K} \beta_i = 1 \tag{10}$$

$$\beta_i \in [0,1] \qquad \text{for } i = 1,\dots,N_K. \tag{11}$$

It is rather easy to demonstrate that these selections for both kernels and weights lead to the final kernel matrix (as in Equation (8)) which still is a valid Mercer's kernel, therefore it can be used on kernelised SVMs. Indeed, Cristianini and Shawe-Taylor in [31] showed that the summation of two valid kernels is still a valid kernel. Further, Horn and Johnson in [63] showed that a positive semi-definite matrix multiplied by a non-negative scalar is still a positive semi-definite matrix. Merging these two results automatically prove that kernels of the form (8) and (9) with constraints (10) and (11) are valid kernels.

3. Proposed Methodology

Let $\mathcal{D}$ be the dataset at hand, split into three non-overlapping subsets $\mathcal{D}_{\text{TR}}$, $\mathcal{D}_{\text{VAL}}$ and $\mathcal{D}_{\text{TS}}$ (namely training set, validation set and test set). Especially for structured data, several representations (e.g., set of descriptors) might hold for the same data, therefore let $\{\mathbf{X}^{(1)},\dots,\mathbf{X}^{(N_R)}\}$ be the set of N_R representations, split in the same fashion (i.e., $\{\mathbf{X}_{\text{TR}}^{(i)}\}_{i=1}^{N_R}$, $\{\mathbf{X}_{\text{VAL}}^{(i)}\}_{i=1}^{N_R}$ and $\{\mathbf{X}_{\text{TS}}^{(i)}\}_{i=1}^{N_R}$). Finally, let $\{d^{(1)}(\cdot,\cdot),\dots,d^{(N_R)}(\cdot,\cdot)\}$ be the set of dissimilarity measures suitable for working in their respective representations.

The respective training, validation and test pairwise dissimilarity matrices, as in Equation (1) can be evaluated as follows:

$$\begin{array}{lcl}
\mathbf{D}_{\text{TR}}^{(1)} = d^{(1)}(\mathbf{X}_{\text{TR}}^{(1)},\mathbf{X}_{\text{TR}}^{(1)}) & \dots & \mathbf{D}_{\text{TR}}^{(N_R)} = d^{(N_R)}(\mathbf{X}_{\text{TR}}^{(N_R)},\mathbf{X}_{\text{TR}}^{(N_R)}) \\
\mathbf{D}_{\text{VAL}}^{(1)} = d^{(1)}(\mathbf{X}_{\text{VAL}}^{(1)},\mathbf{X}_{\text{TR}}^{(1)}) & \dots & \mathbf{D}_{\text{VAL}}^{(N_R)} = d^{(N_R)}(\mathbf{X}_{\text{VAL}}^{(N_R)},\mathbf{X}_{\text{TR}}^{(N_R)}) \\
\mathbf{D}_{\text{TS}}^{(1)} = d^{(1)}(\mathbf{X}_{\text{TS}}^{(1)},\mathbf{X}_{\text{TR}}^{(1}) & \dots & \mathbf{D}_{\text{TS}}^{(N_R)} = d^{(N_R)}(\mathbf{X}_{\text{TS}}^{(N_R)},\mathbf{X}_{\text{TR}}^{(N_R)}).
\end{array} \tag{12}$$

Let $\mathbf{w} \in \{0,1\}^{|\mathcal{D}_{\text{TR}}|}$ be a binary vector in charge of selecting columns from all matrices in Equation (12): the full pairwise dissimilarities can be sliced to their 'reduced' versions (cf. Equation (1) vs. Equation (2)), hence:

$$\begin{array}{lcl}
\bar{\mathbf{D}}_{\text{TR}}^{(1)} = \mathbf{D}_{\text{TR}}^{(1)}(:,\mathbf{w}) & \dots & \bar{\mathbf{D}}_{\text{TR}}^{(N_R)} = \mathbf{D}_{\text{TR}}^{(N_R)}(:,\mathbf{w}) \\
\bar{\mathbf{D}}_{\text{VAL}}^{(1)} = \mathbf{D}_{\text{VAL}}^{(1)}(:,\mathbf{w}) & \dots & \bar{\mathbf{D}}_{\text{VAL}}^{(N_R)} = \mathbf{D}_{\text{VAL}}^{(N_R)}(:,\mathbf{w}) \\
\bar{\mathbf{D}}_{\text{TS}}^{(1)} = \mathbf{D}_{\text{TS}}^{(1)}(:,\mathbf{w}) & \dots & \bar{\mathbf{D}}_{\text{TS}}^{(N_R)} = \mathbf{D}_{\text{TS}}^{(N_R)}(:,\mathbf{w}).
\end{array} \tag{13}$$

where, due to the number of subscripts and superscripts in Eq. (13), for ease of notation, we used a MATLAB®-like notation for indexing matrices.

In other words, $\mathbf{w}$ acts as a feature (prototype) selector. Given this newly obtained dataset, it is possible to train a kernelised ν-SVM [64] whose multiple kernel has the form Equation (8) where each one has the form Equation (9), thus:

$$\mathbf{K} = \sum_{i=1}^{N_R} \beta_i \cdot \exp\left\{ -\gamma_i \cdot \|\bar{\mathbf{D}}_{\mathrm{TR}}^{(i)} \ominus \bar{\mathbf{D}}_{\mathrm{TR}}^{(i)}\|^2 \right\}, \tag{14}$$

where $\ominus$ denotes the pairwise difference. Hence, each dissimilarity representation is subject to a proper non-linear kernel ($N_K \equiv N_R$).

A genetic algorithm [65] acts as a wrapper method in order to automatically tune in a fully data-driven fashion the several free parameters introduced in this problem. The choice behind a genetic algorithm stems from them being widely famous in the context of derivative-free optimisation, embarrassingly easy to parallelise and for the sake of consistency with competing techniques (see Section 4.4). For our problem, the genetic code has the form:

$$\begin{bmatrix} \nu & \boldsymbol{\beta} & \boldsymbol{\gamma} & \mathbf{w} \end{bmatrix}, \tag{15}$$

where $\nu \in (0,1]$ is the SVM regularisation term, $\boldsymbol{\beta} = [\beta]_{i=1}^{N_R}$ contains the kernel weights, $\boldsymbol{\gamma} = [\gamma_i]_{i=1}^{N_R}$ contains the kernel shapes and $\mathbf{w}$ properly selects prototypes in the dissimilarity space, as described above.

For the sake of argument, it is worth remarking that there have been several attempts to use evolutionary strategies in order to tune multiple kernel machines: for example in [66] a genetic algorithm has been used in order to tune the kernel shapes (namely, $\boldsymbol{\gamma}$), whereas in [67] both the kernel shapes and the kernel weights have been tuned by means of a $(\mu + \lambda)$ evolution strategy [68]. Conversely, the idea of using a genetic algorithm for prototypes selection in the dissimilarity space has been inherited from a previous work [44].

The fitness function to be maximised is the informedness J (also known as Youden's index [69]) defined as:

$$J = \text{specificity} + \text{sensitivity} - 1, \tag{16}$$

which is, by definition, bounded in range $[-1,1]$ (the closer to 1, the better). For the sake of comparison with other performance measures (e.g., accuracy, F-score and the like) which are, by definition, bounded in $[0,1]$, the fitness function sees a scaled version of the informedness [23–25], hence:

$$f_1 \equiv \bar{J} = \frac{J-(-1)}{1-(-1)} = \frac{J+1}{2} \in [0,1]. \tag{17}$$

The rationale behind using the informedness rather than other most common performance measures (mainly accuracy and F-score) is that the informedness is well suited for unbalanced classes without being biased towards the most frequent class (the same is not true for accuracy) and whilst considering also true negative predictions (the same is not true for F-score) [70].

By assuming that the full dissimilarity matrices are pre-evaluated beforehand, the objective function evaluation is performed for each individual from the current generation as follows:

1. The individual receives the N_R full dissimilarity matrices between training data samples, i.e., $\mathbf{D}_{\mathrm{TR}}^{(1)}, \ldots, \mathbf{D}_{\mathrm{TR}}^{(N_R)}$ as in Equation (12);
2. According to the $\mathbf{w}$ portion of its genetic code (see Equation (15)), a subset of prototypes is selected, leading to the 'reduced' dissimilarity matrices between training data, i.e., $\bar{\mathbf{D}}_{\mathrm{TR}}^{(1)}, \ldots, \bar{\mathbf{D}}_{\mathrm{TR}}^{(N_R)}$ as in Equation (13);
3. Considering the $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ values in its genetic code, the (multiple) kernel matrix is evaluated by using Equation (14);
4. A ν-SVM is trained using the regularisation term ν from the genetic code and the kernel matrix from step #3;

5. The individual receives the N_R full dissimilarity matrices between training and validation data, each of which is computed by considering all possible $\langle x, y \rangle$-pairs where x belongs to the validation set and y belongs to the training set, i.e., $\mathbf{D}^{(1)}_{\text{VAL}}, \ldots, \mathbf{D}^{(N_R)}_{\text{VAL}}$ as in Equation (12);
6. The 'reduced' dissimilarity matrices are projected thanks to $\mathbf{w}$, i.e., $\bar{\mathbf{D}}^{(1)}_{\text{VAL}}, \ldots, \bar{\mathbf{D}}^{(N_R)}_{\text{VAL}}$ as in Equation (13);
7. The (multiple) kernel matrix between training and validation data is evaluated thanks to $\boldsymbol{\beta}$ and γ, alike Equation (14);
8. The (multiple) kernel matrix from step #7 is fed to the SVM trained on step #4 and the predicted classes on the validation set are returned;
9. The fitness function is evaluated.

At the end of the evolution, the best individual (i.e., the one with best performances on the validation set) is retained and its final performances are evaluated on the test set.

Finally, it is worth remarking the rationale behind the proposed, structured, genetic code since a genetic code of the form Equation (15) allows, in a two-fold manner, a deeper a posteriori knowledge discovery phase. Indeed, using upfront good classification results (for the sake of reliability), by looking at $\boldsymbol{\beta}$, it is possible to check which kernels (representations) are considered as the most important (higher weights) for the learning machine in order to solve the problem at hand. Similarly, by looking at $\mathbf{w}$, it is possible to check which training set patterns have been selected as representatives and ask why those patterns have been selected instead of others, leading to a pattern-wise check (possibly with help by field-experts). Especially the latter a posteriori check might be troublesome if a huge number of representatives is selected. In order to alleviate this problem (if present), it is possible to re-state the fitness function (formerly (17)) by considering a convex linear combination between the performance index and the feature selector sparsity, hence:

$$f_2 = \omega\,(1 - \bar{J}) + (1 - \omega)\frac{|\{i : \mathbf{w}_i = 1\}|}{|\mathbf{w}|}, \tag{18}$$

where $\omega \in [0, 1]$ in a user-defined parameter which tunes the convex linear combination by weighting the rightmost term (sparsity) against the leftmost term (performance). It is worth noting that whilst fitness (17) should be maximised, (18) should be minimised.

4. Tests and Results

4.1. Data Collection and Pre-Processing

The data retrieval processing can be summarised as follows. Using the Python BioServices library [71]:

1. The entire protein list for Escherichia coli str. K12 has been retrieved from UniProt [72];
2. This list has been cross-checked with Protein Data Bank [73] in order to discard unresolved proteins (i.e., proteins whose 3D structure is not available).

Then, using the BioPython library [74]:

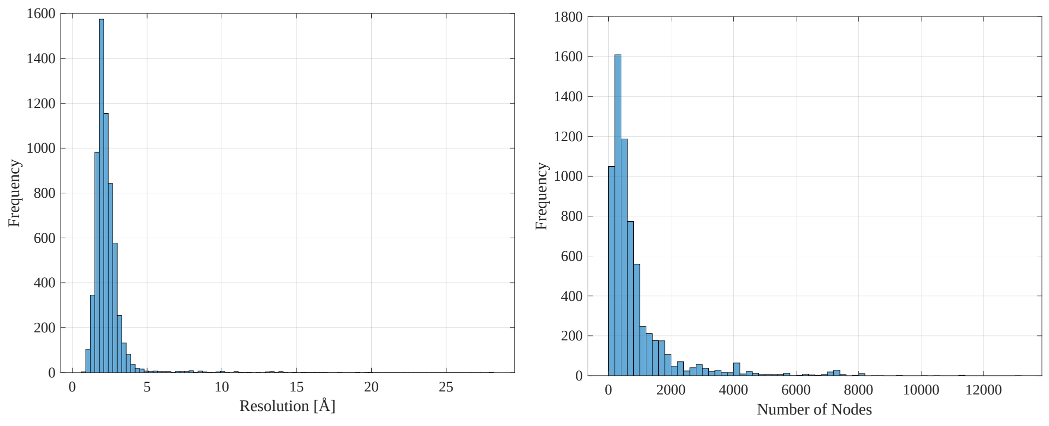
1. .pdb files have been downloaded for all resolved proteins;
2. information such as the EC number and the measurement resolution (if present) have been parsed from the .pdb file header;
3. proteins having multiple EC numbers have been discarded.

Finally, using the BioPandas library [75]:

1. α-carbon atoms 3D coordinates have been parsed from each .pdb file;
2. In case of multiple equivalent models within the same .pdb file, only the first model is retained;

3. Similarly, for atoms having alternate coordinate locations, only the first location is retained.

After this retrieval stage, a total number of 6685 proteins has been successfully collected. Some statistics on the measurement resolutions and the number of nodes are sketched in Figure 1a,b, respectively.



(**a**) Resolution (**b**) Size

Figure 1. Distributions within the original 6685 proteins set.

In order to keep only good quality structures (with reliable atomic coordinates), all proteins with missing resolution in their respective .pdb files and proteins whose resolution is greater than 3Å have been discarded. Further, proteins having more than 1500 nodes have been discarded as well. These filtering procedures dropped the number of available proteins from 6685 to 4957. The class labels (EC number) distribution is summarised in Table 1.

Table 1. Classes distribution within the filtered 4957 proteins set.

								Total
Class	EC1	EC2	EC3	EC4	EC5	EC6	not-enzymes	
Count	540	1017	919	329	182	244	1726	4957
Percentage	10.89	20.52	18.54	6.64	3.67	4.92	34.82	100%

For each of the 4957 available proteins, its respective eight representations (see Appendix A) have been evaluated using the following tools:

- The NetworkX library [76] (Python) for evaluating centrality measures ($\mathbf{X}^{(2)}$) and the Vietoris–Rips complex ($\mathbf{X}^{(1)}$);
- The Numpy and Scipy libraries [77,78] (Python) for several algebraic computations, mainly spectral decompositions for energy, Laplacian energy, heat trace, heat content invariants ($\mathbf{X}^{(3)}$, $\mathbf{X}^{(5)}$, $\mathbf{X}^{(6)}$, $\mathbf{X}^{(8)}$) and the homology group rank ($\mathbf{X}^{(1)}$);
- The Rnetcarto (https://cran.r-project.org/package=rnetcarto) library (R) for network cartography ($\mathbf{X}^{(4)}$).

As in previous works [45,46] the 7-class classification problem is cast into seven binary classification problems in one-against-all fashion, hence the ith classifier sees the ith class as positive and all other classes as negative. The eight representations $\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(8)}$ are split into training, validation and test set in a stratified manner in order to preserve labels' distribution across splits. Thus, each of the seven classifiers sees a different training-validation-test split due to the one-against-all labels recoding. The genetic optimisation and classification stage has been performed in MATLAB® R2018a using the built-in genetic algorithm and LibSVM [79] for ν-SVMs.

4.2. Computational Results with Fitness Function f_1

The first test suite sees f_1 (17) as the fitness function, hence the system aims at the maximisation of the (normalised) informedness.

The genetic algorithm has been configured to host 100 individuals for a maximum of 100 generations and each individual's genetic code (upper/lower bounds and constraints, if any) is summarised in Table 2. At each generation, the elitism is set to the top 10% individuals; the crossover operates in a scattered fashion; the selection operator follows the roulette wheel heuristic and the mutation adds to each real-valued gene (ν, $\boldsymbol{\beta}$, γ) a random number extracted from a zero-mean Gaussian distribution whose variance shrinks as generations go by, whereas it acts in a flip-the-bit fashion for boolean-valued genes ($\mathbf{w}$).

Table 2. Genetic algorithm parameters description.

Parameter	Bounds	Contraints
ν	$(0, 1]$ by definition	
$\boldsymbol{\beta}$	$\beta_i \in [0, 1],\ \forall i = 1, \ldots, N_R$	$\sum_{i=1}^{N_R} \beta_i = 0$
γ	$\gamma \in (0, 100],\ \forall i = 1, \ldots, N_R$	
$\mathbf{w}$	$w_i \in \{0, 1\},\ \forall i = 1, \ldots, \lvert\mathcal{D}_{\text{TR}}\rvert$	

Table 3 shows the performances obtained by the proposed Multiple Kernels over Multiple Dissimilarities (MKMD, for short) approach using the fitness function f_1. Due to randomness in genetic optimisation, five runs have been performed for each classifier and the average results are shown. Figures of merit include:

- Accuracy $= \dfrac{TP + TN}{TP + FP + TN + FN}$;
- Precision $= \dfrac{TP}{TP + FP}$;
- Recall (Sensitivity) $= \dfrac{TP}{TP + FN}$;
- (Normalised) Informedness as in Equation (17);
- Area Under the Curve (AUC), namely the area under the Receiver Operating Characteristic (ROC) curve [80];

where TP, TN, FP and FN indicate true positives, true negatives, false positives and false negatives, respectively.

Table 3. Test Set Performances with Fitness Function f_1.

Class	Performances					Complexity
	Accuracy	Precision	Recall	Informedness [†]	AUC	Sparsity
1 (EC1)	0.95	0.87	0.68	0.83	0.92	49.43
2 (EC2)	0.91	0.88	0.66	0.82	0.90	49.62
3 (EC3)	0.90	0.84	0.58	0.78	0.88	49.48
4 (EC4)	0.97	0.90	0.56	0.78	0.88	49.42
5 (EC5)	0.98	0.83	0.44	0.72	0.78	50.78
6 (EC6)	0.99	0.94	0.76	0.88	0.95	49.28
7 (not-enzymes)	0.82	0.77	0.70	0.79	0.89	50.52

[†] Normalised.

Similarly, Figure 2 shows the ROC curves for all classifiers by considering their respective run with greatest AUC.

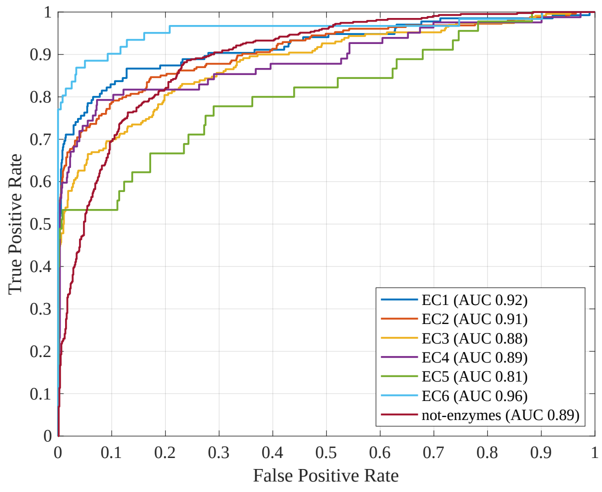


Figure 2. ROC curves with fitness function f_1. In brackets, the respective AUC values.

4.3. Computational Results with Fitness Function f_2

These experiments see the fitness function f_2 (Equation (18)) in lieu of f_1 (Equation (17)), where the weighting parameter ω is set to 0.5 in order to give the same importance to performances and sparsity. In order to ensure a fair comparison with the previous analysis, the same training-validation-test splits have been used for all seven classifiers, along with the same genetic algorithm setup (genetic code, number of individuals and generations, genetic operators). Table 4 shows the average performances obtained by the seven classifiers across five genetic algorithm runs. As in the previous case, Figure 3 shows the ROC curves for all classifiers by considering their respective run with greatest AUC.

Table 4. Test set performances with fitness function f_2 and $\omega = 0.5$.

Class	Performances					Complexity
	Accuracy	Precision	Recall	Informedness [†]	AUC	Sparsity
1 (EC1)	0.95	0.86	0.69	0.84	0.92	33.08
2 (EC2)	0.91	0.88	0.67	0.82	0.90	32.48
3 (EC3)	0.90	0.83	0.57	0.77	0.87	29.94
4 (EC4)	0.97	0.88	0.54	0.77	0.88	33.89
5 (EC5)	0.98	0.85	0.45	0.73	0.79	35.54
6 (EC6)	0.98	0.91	0.76	0.88	0.95	35.38
7 (not-enzymes)	0.82	0.77	0.69	0.79	0.88	33.37

[†] Normalised.

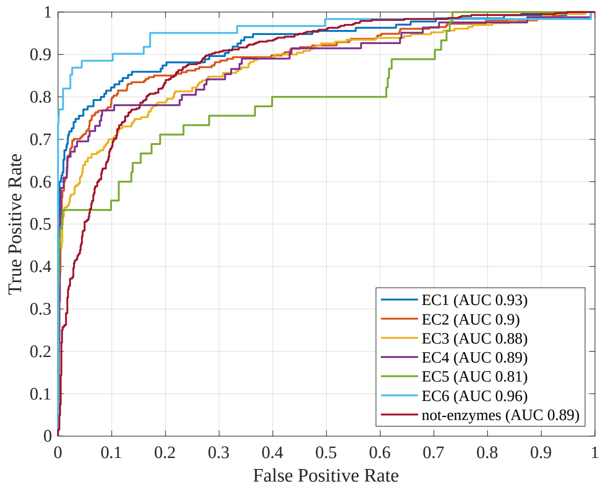


Figure 3. ROC curves with fitness function f_2 and $\omega = 0.5$. In brackets, the respective AUC values.

4.4. Benchmarking against a Clustering-Based One-Class Classifier

In order to properly benchmark the proposed MKMD system, a One-Class Classification System (hereinafter OCC or OCC_System) capable of exploiting multiple dissimilarities is used. This classification system has been initially proposed in [81] and later used for modelling complex systems such as smart grids [81–83] and protein networks [44].

The main idea in order to build a model through the One-Class Classifier is to use a clustering-evolutionary hybrid technique [81,82]. The main assumption is that similar protein types have similar chances of generating a specific class, reflecting the cluster model. Therefore, the core of the recognition system is a custom-based dissimilarity measure computed as a weighted Euclidean distance, that is:

$$d(\check{\vec{x}}_1, \check{\vec{x}}_2; \vec{W}) = \sqrt{(\check{\vec{x}}_1 \ominus \check{\vec{x}}_2)^T \vec{W}^T \vec{W} (\check{\vec{x}}_1 \ominus \check{\vec{x}}_2)}, \tag{19}$$

where $\check{\vec{x}}_1, \check{\vec{x}}_2$ are two generic patterns and $\vec{W}$ is a diagonal matrix whose elements are generated through a suitable vector of weights $\vec{w}$. The dissimilarity measure is component-wise, therefore the $\ominus$ symbol represents a generic dissimilarity measure, tailored on each pattern subspace, that has to be specified depending on the semantic of data at hand.

In this study, patterns are represented by dissimilarity vectors extracted from each sub-dissimilarity matrix, one for each feature adopted to describe the protein (see Section 2). In other words, patterns pertain to a suitable dissimilarity space.

The decision region of each cluster C_i is constructed around the medoid c_i bounded by the average radius $\delta(C_i)$ plus a threshold σ, considered together with the dissimilarity weights $\vec{w} = diag(\vec{W})$ as free parameters. Given a test pattern $\check{\vec{x}}$ the decision rule consists in evaluating whether it falls inside or outside the overall target decision region, by checking whether it falls inside the closest cluster. The learning procedure consists in clustering the training set $\mathcal{D}_{\text{TR}}$ composed by target patterns, adopting a standard genetic algorithm in charge of evolving a family of cluster-based classifiers considering the weights $\vec{w}$ and the thresholds of the decision regions as search space, guided by a proper objective function. The latter is evaluated on the validation set $\mathcal{D}_{\text{VAL}}$, taking into account a linear combination of the accuracy of the classification (that we seek to maximise) and the extension of the thresholds (that should be minimised). Note that in building the classification model we use only target patterns, while non-target ones are used in the cross-validation phase, hence the adopted learning paradigm is the One-Class classification one [84,85]. Moreover, in order to outperform the well-known limitations of the initialization of the standard k-means algorithm, the OCC_System initializes more than one instance of the clustering algorithm with random starting representatives, namely medoids, since the OCC_System is capable of dealing with arbitrarily structured data [86–88]. At test stage (or during validation) a voting procedure for each cluster model is performed. This technique allows building a more robust proteins model.

Figure 4 shows the schematic representing the core subsystems of the proposed OCC_System, such as the ones performing the clustering procedure and the genetic algorithm. Moreover, it is shown the Test subsystem, where given a generic test pattern and given a learned model, it is possible to associate a score value (soft-decision) besides the Boolean decision. Hence, we equip each cluster $\mathcal{C}_i$ with a suitable membership function, denoted in the following as $\mu_{\mathcal{C}_i}(\cdot)$. In practice, we generate a fuzzy set [89] over $\mathcal{C}_i$. The membership function allows quantifying the uncertainty (expressed by the membership degree in $[0, 1]$) of a decision about the recognition of a test pattern. Membership values close to either 0 or 1 denote ”certain” and hence reliable decisions. When the membership degree assigned to a test pattern is close to 0.5, there is no clear distinction about the fact that such a test pattern is really a target pattern or not (regardless of the correctness of the Boolean decision).

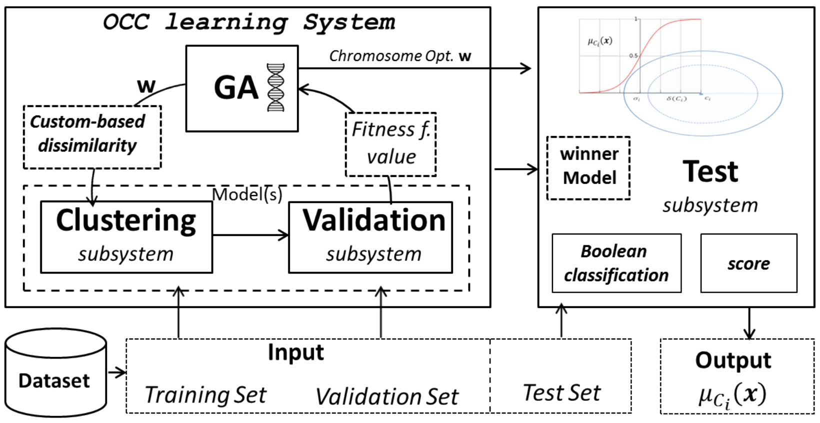


Figure 4. Schematic of the classification system able to learn a classification model for each positive class. The model provides the crisp decision as well as a score (a real number) encoding the decision reliability.

For this purpose, we adopt a parametric sigmoid model for $\mu_{C_i}(\cdot)$, which is defined as follows:

$$\mu_{C_i}(x) = \frac{1}{1 + \exp\{(d(c_i, x) - b_i)/a_i\}}, \tag{20}$$

where $a_i, b_i \geq 0$ are two parameters specific to $\mathcal{C}_i$, and $d(\cdot,\cdot)$ is the dissimilarity measure (19). Notably, a_i is used to control the steepness of the sigmoid (the lower the value, the faster the rate of change), and b_i is used to translate the function in the input domain. If a cluster (that models a typical protein found in the training set) is very compact, then it describes a very specific scenario. Therefore, no significant variations should be accepted to consider test patterns as members of this cluster. Similarly, if a cluster is characterised by a wide extent, then we might be more tolerant in the evaluation of the membership. Accordingly, the parameter a_i is set equal to $\delta(\mathcal{C}_i)$. On the other hand, we define $b_i = \delta(\mathcal{C}_i) + \sigma_i/2$. This allows us to position the part of the sigmoid that changes faster right in-between the area of the decision region determined by the dissimilarity values falling in $[B(\mathcal{C}_i) - \sigma_i, B(\mathcal{C}_i)]$, where in turn $B(\mathcal{C}_i) = \delta(\mathcal{C}_i) + \sigma_i$ is the boundary of the decision region related to the ith cluster.

Finally, the soft decision function, $s(\cdot)$, is defined as

$$s(\bar{x}) = \mu_{\mathcal{C}^*}(\bar{x}), \tag{21}$$

where $\mathcal{C}^*$ is the cluster where the test (target) pattern falls.

With the aim of making a synthesis, we remark that the OCC_System works in two phases:

1. Learning a cluster model of proteins through a suitable dataset divided into two disjoint sets, namely training and validation set;
2. Using the learned model in order to recognise or classify unseen proteins drawn from the test set, assigning to each pattern a probability value.

The OCC parameters defining the model are optimised by means of a genetic algorithm guided by a suitable objective function that takes into account the classification accuracy. For the sake of comparison, the same genetic operators (selection, mutation, crossover, elitism) as per the MKMD system and have been considered (see Section 4.2). As concerns the complexity of the model, measured as the cardinality of the partition k, we choose a suitable value $k = 120$.

Table 5 shows the comparison between the OCC_System and the MKMD approach. In order to ensure a fair comparison, since the OCC_System does not perform representatives selection in the dissimilarity space, in the MKMD genetic code (cf. Equation (15)), the weights vector $\mathbf{w}$ has been removed and all weights have been considered unitary (i.e., no representative selection). Similarly, Figure 5b and Figure 5a show the ROC curves for OCC and MKMD, respectively.

From Table 5 is evident that MKML outperforms OCC in terms of accuracy, informedness and AUC (see also the ROC curves in Figure 5b and Figure 5a), but a clear winner does not exist as regards precision and recall. As regards the structural complexity, OCC is bounded by the number of clusters k, whereas MKMD is bounded by the number of support vectors as returned by the training phase [24]. Indeed, the computational burden required to classify new test data is given by:

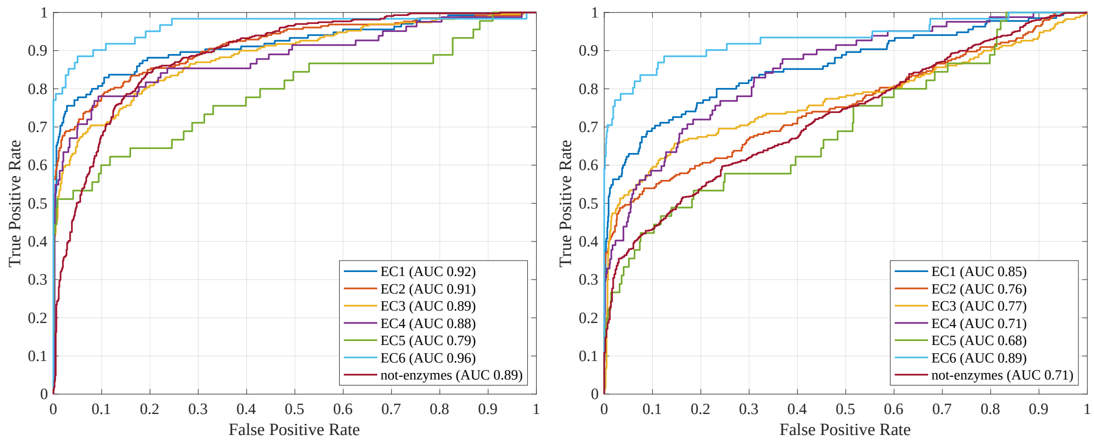
- The pairwise distances between the test data and the k clusters centres (for OCC);
- The dot product between the test data and the support vectors (for MKMD).

Specifically, for OCC, a suitable number of 120 clusters has been defined for all classes, whereas the training phase for MKMD returned an average of 1300 support vectors ($\sim$52% of the training data) for class 1, 1881 support vectors ($\sim$76%) for class 2, 1745 support vectors ($\sim$70%) for class 3, 1213 support vectors ($\sim$49%) for class 4, 767 support vectors ($\sim$31%) for class 5, 864 support vectors ($\sim$35%) for class 6 and 1945 support vectors ($\sim$78%) for class 7. In conclusion, whilst MKMD outperforms OCC in terms of performances, the latter outperforms the former in terms of structural complexity.

Table 5. Test set performances with the one-class classifier.

Class	Classifier	Performances				
		Accuracy	Precision	Recall	Informedness [†]	AUC
1 (EC1)	OCC	0.92	**0.97**	0.35	0.67	0.85
	MKMD	**0.95**	0.88	**0.67**	**0.83**	**0.91**
2 (EC2)	OCC	0.83	0.87	0.45	0.69	0.76
	MKMD	**0.91**	**0.89**	**0.66**	**0.82**	**0.91**
3 (EC3)	OCC	0.83	**0.86**	0.49	0.70	0.77
	MKMD	**0.90**	0.84	**0.57**	**0.77**	**0.88**
4 (EC4)	OCC	0.68	0.60	**0.78**	0.61	0.72
	MKMD	**0.97**	**0.89**	0.53	**0.76**	**0.87**
5 (EC5)	OCC	0.85	0.75	0.37	0.62	0.69
	MKMD	**0.98**	**0.82**	**0.44**	**0.72**	**0.78**
6 (EC6)	OCC	0.97	0.96	0.57	0.78	0.88
	MKMD	**0.99**	**0.92**	**0.77**	**0.88**	**0.95**
7 (not-enzymes)	OCC	0.68	0.60	**0.78**	0.61	0.72
	MKMD	**0.82**	**0.78**	0.68	**0.79**	**0.88**

[†] Normalised.



(**a**) MKMD (**b**) OCC

Figure 5. ROC curves comparison (best run for all classes). In brackets, the respective AUC values.

4.5. Comparing against Previous Works

In Table 6 are reported the performances (in terms of AUC only, for the sake of shorthand) between the proposed MKMD approach with fitness function f_1 (Table 3), with fitness function f_2 (Table 4) and with no representatives selection in the embedding space (Table 5) against our previous studies for solving the same classification problem. For the sake of completeness, the results obtained by OCC (Table 5) are also included.

Table 6. Comparison (in terms of AUC) between the proposed MKMD approach and previous studies.

Approach	EC1	EC2	EC3	EC4	EC5	EC6	Not-Enzymes
DME + Logistic Regression [44]	–	–	–	–	–	–	0.62
DME + SVM [44]	–	–	–	–	–	–	0.64
DME + Naïve Bayes [44]	–	–	–	–	–	–	0.62
DME + Decision Tree [44]	–	–	–	–	–	–	0.60
DME + Neural Network [44]	–	–	–	–	–	–	0.63
OCC [44]	–	–	–	–	–	–	0.63
Feature Generation via Betti Numbers + SVM [46]	0.79	0.75	0.73	0.73	0.46	0.77	0.77
Feature Generation via Spectral Density + SVM [45]	0.85	0.82	0.85	0.81	0.59	0.81	0.82
MKMD with f_1 (Table 3)	0.92	0.90	0.88	0.88	0.78	0.95	0.89
MKMD with f_2 (Table 4)	0.92	0.90	0.87	0.88	0.79	0.95	0.88
MKMD with no representative selection (Table 5)	0.91	0.91	0.88	0.87	0.78	0.95	0.88
OCC (Table 5)	0.85	0.76	0.77	0.72	0.69	0.88	0.72

In [44], two experiments have been performed: the first relied on the Dissimilarity Matrix Embedding (DME) by considering different protein representations (similar to the ones considered in this work) and the second one relied on OCC being able to explore those different representations simultaneously (alike this work). There are three main differences between this work and [44]: first, the set of representations is different; second, we only managed to solve the binary classification problem between enzymes and not-enzymes; third, the set of considered proteins is different. In fact, in [44], we performed an additional filtering stage in order to select (for the same UniProt ID) only the PDB entry with best resolution: we found that this heavily limits the number of protein samples available, possibly reducing the learning capabilities.

In [45,46] we used the sampled spectral density of the protein contact networks (more information can be found in Appendix A.8) and the Betti numbers (more information can be found in Appendix A.1), respectively: the results in Table 6 feature the same proteins set used in this work. Indeed, thanks to the observation above, experiments have been repeated with an augmented number of protein samples [90,91].

Results in Table 6 highlight that:

1. Avoiding to filter out PDB structures by considering only the best resolution for a given UniProt ID (as carried out also in this work) helps in improving classification models: indeed, performances from [44] are amongst the lowest ones;
2. The proposed MKMD approach, regardless of the fitness function and/or representative selection, outperforms all competitors for all EC classes (including not-enzymes).

4.6. On the Knowledge Discovery Phase

Apart from the good generalisation capabilities, it is worth remarking that an interesting aspect of the proposed multiple kernel approach is the two-fold knowledge discovery phase:

1. By analysing the kernel weights $\boldsymbol{\beta}$, it is possible to determine the most important representations for the problem at hand;

2. By analysing **w**, namely the binary vector in charge of selecting prototypes from the dissimilarity space, it is possible to determine and analyse the patterns (proteins, in this case) elected as prototypes.

Let us start our discussion from the latter point. From a chemical viewpoint, proteins are linear hetero-polymers in the form of non-periodic sequences of 20 different monomers (amino-acids residues). While artificial polymers (periodic) are very large extended molecules forming a matrix, the majority of proteins fold as self-contained water-soluble structures. Thus, we can consider the particular linear arrangement of amino-acid residues as a sort of ’recipe’ for making a water-soluble polymer with a well-defined three-dimensional architecture [92]. “Well-defined three-dimensional structure” should not be intended as a ’fixed architecture’: many proteins appear as partially or even totally disordered when analysed with spectroscopic methods. This apparent disorder corresponds to an efficient organisation as for protein physiological role giving to the molecule the possibility to adapt to rapidly changing microenvironment conditions [93].

This implies the two main drivers of amino-acid residues 3D arrangement (from where the particular properties of relative contact networks derive) are:

1. To efficiently accomplish the task of being water soluble while maintaining a stable structure (or dynamics);
2. To allow for an efficient spreading of the signal across amino-acid residues contact network so to sense relevant microenvironment changes and to reshape accordingly—allosteric effect, see [94].

Currently, we have only a coarse-grain knowledge of such complex tasks, and biochemists are still very far to be able to reproduce this behaviour by synthetic constructs.

The ability to catalyse a specific class of chemical reactions (the property the EC classification is based upon), while being crucial for the biological role of protein molecules is, from the point of view of topological and geometrical proteins structure, only a very minor modulation of their global shape [92]. Notwithstanding that, the thorough analysis of representative proteins (thus pivotal for discrimination) can give us some general hints, not only confined to the specific classification task, but extending to all the ’hard’ classification problems based upon very tiny details of the statistical units.

Looking at the representative proteins (hence, endowed with meaningful discriminative power) in Tables A1–A7 (Appendix B) we immediately note that the pivotal proteins come from all the analysed EC categories and not only from the specific class to be discriminated. This is expected by the absence of a simple form-function relation, hence they can be considered as an ’emergent property’ of the discrimination task. The presence of molecules of different classes crucial for a specific category modelling and thus the image in light of a peculiar strategy adopted by the system is analogue to the use of ’paired samples’ in statistical investigation [95,96]. When in presence of only minor details discriminating statistical units pertaining to different categories, the only possibility to discriminate is to adopt a paired samples strategy in which elements of a category is paired with a very similar example of another category so to rely on their differences (on a sample-by-sample basis) instead of looking for a general ’class-specific’ properties. This is the case of proteins whose general shape is only partially determined by the chemical reaction they catalyse: looking at the 3D structures of relevant proteins, we can easily verify they pertain to three basic patterns (Figure 6):

1. Cyclic pattern with an approximately spherical symmetry (Figure 6a);
2. A globular pattern with ’duplication’: protein can be considered as two identical half-structures (Figure 6b);
3. Elongated non-cyclic pattern, typical of membrane-bound proteins (Figure 6c).

Even if the three above-mentioned patterns have slightly different relative frequencies in the EC classes (e.g., pattern 3 is more frequent in non-enzymatic proteins), they are present in all the analysed classes so allowing for the ‘between-categories’ sample-by-sample pairing mentioned above.

This peculiar situation is in line with current biochemical knowledge (minimal effect exerted by catalysed reaction on global structure) and it is a relevant proof-of-concept of both the reliability of the classification solution and of the power of the proposed approach. On the other hand, it is very hard to de-convolve the discriminating structural nuances from the obtained solution that, as it is, only confirms the presence of ’tiny and still unknown’ structural details linked to the catalytic activity of the studied molecules.

As regards the former point, Figure 7 shows the average weights vector β across the aforementioned five runs for $\omega = 0.5$, showing that the MKMD approach considers for almost all classes centrality measures ($\mathbf{X}_2$) and the protein size ($\mathbf{X}_7$) as the most relevant representations, followed by the Betti numbers sequence ($\mathbf{X}_1$), heat content invariants ($\mathbf{X}_5$) and heat kernel trace ($\mathbf{X}_6$).

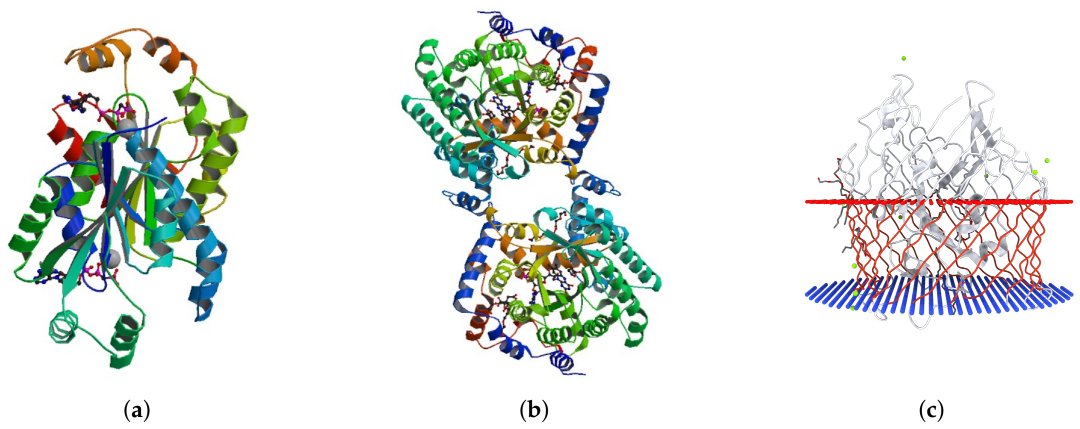
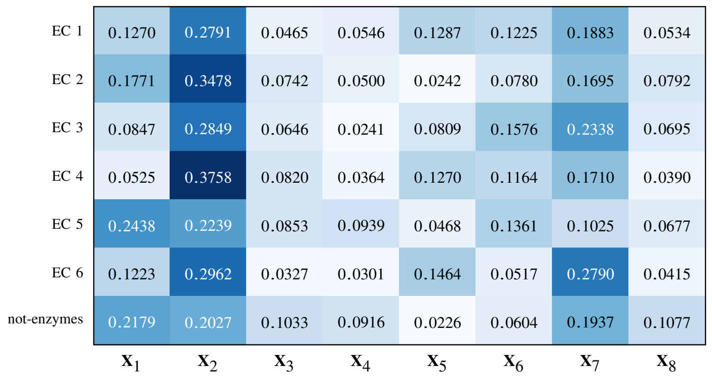


Figure 6. Three basic patterns in protein 3D structures. (**a**) Transferase—PDB ID 1KOF, (**b**) Proline dehydrogenase (oxidoreductase)—PDB ID 3E2R, (**c**) Transport Protein (Non-Enzyme)—PDB ID 3RGM.



	$\mathbf{x}_1$	$\mathbf{x}_2$	$\mathbf{x}_3$	$\mathbf{x}_4$	$\mathbf{x}_5$	$\mathbf{x}_6$	$\mathbf{x}_7$	$\mathbf{x}_8$
EC 1	0.1270	0.2791	0.0465	0.0546	0.1287	0.1225	0.1883	0.0534
EC 2	0.1771	0.3478	0.0742	0.0500	0.0242	0.0780	0.1695	0.0792
EC 3	0.0847	0.2849	0.0646	0.0241	0.0809	0.1576	0.2338	0.0695
EC 4	0.0525	0.3758	0.0820	0.0364	0.1270	0.1164	0.1710	0.0390
EC 5	0.2438	0.2239	0.0853	0.0939	0.0468	0.1361	0.1025	0.0677
EC 6	0.1223	0.2962	0.0327	0.0301	0.1464	0.0517	0.2790	0.0415
not-enzymes	0.2179	0.2027	0.1033	0.0916	0.0226	0.0604	0.1937	0.1077

Figure 7. Average kernel weights vectors β.

It is worth noting that enzymes have a more pronounced allosteric effect with respect to non-enzymatic structures. This is a consequence of the need to modulate chemical kinetics according to microenvironment conditions—allostery is the modulating effect of a modification happening in a site different from catalytic site on the efficiency of the reaction [97]. Allostery implies an efficient transport of the signal along protein structure and it was discovered to be efficiently interpreted in terms of PCN descriptors [98] thus, the observed kernel weights fit well with the current biochemical knowledge.

5. Conclusions

In this paper, we proposed a classification system able to explore simultaneously multiple representations following an hybridisation between multiple kernel learning and dissimilarity spaces, hence exploiting the discriminative power of kernel methods and the customisability of dissimilarity spaces.

Specifically, several representations are treated using their respective dissimilarity representations and combined in a multiple kernel fashion, where each kernel function considers a specific dissimilarity representation. A genetic algorithm (although any derivative-free evolutive metaheuristic can be placed instead) is able to simultaneously select suitable representatives in the dissimilarity space and tune the kernel weights, allowing a two-fold a posteriori knowledge discovery phase regarding the most suitable representations (higher kernel weights) and the patterns elected as prototypes in the dissimilarity space.

The proposed MKMD system has been applied for solving a real-world problem, namely protein function prediction, with satisfactory results, greatly outperforming our previous works in which graph-based descriptors extracted from PCNs have been tested for solving the very same problem. Further, the proposed system has been benchmarked against a One-Class Classifier, also able to simultaneously explore multiple dissimilarities: whilst the former outperforms the latter in terms of accuracy, AUC and informedness, a clear winner between the two methods does not exist in terms of precision and recall.

As far as the two-fold knowledge discovery phase for the proposed application is concerned, results both in terms of selected representatives in the dissimilarity space and weights automatically assigned to different representations are in line with current biological knowledge, showing the reliability of the proposed system.

Furthermore, due to its flexibility, the proposed system can be applied to any input domain (not necessarily graphs), provided that several representations can be extracted by the structured data at hand and that suitable dissimilarity measures can be defined for such heterogeneous representations.

Author Contributions: Conceptualization, A.M.; Data curation, A.M. and A.G.; Formal analysis, A.M., A.G. and E.D.S.; Investigation, A.M., A.G., E.D.S. and A.R.; Methodology, A.M.; Resources, A.M. and A.G.; Software, A.M. and E.D.S; Supervision, A.G. and A.R.; Validation, A.M. and A.G.; Writing–original draft, A.M., A.G. and E.D.S.; Writing—review & editing, A.M., A.G., E.D.S. and A.R. All authors have read and agreed to the published version of the manuscript.

Funding: This work has been partially supported by the Sapienza Research Calls project "PARADISE - PARAllel and DIStributed Evolutionary agent-based systems for machine learning and big data mining", 2018.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AUC	Area Under the Curve
DME	Dissimilarity Matrix Embedding
MKMD	Multiple Kernels over Multiple Dissimilarities
OCC	One-Class Classification (also OCC_System)
PCN	Protein Contact Networks
PDB	Protein Data Bank
ROC	Receiver Operating Characteristic
SVM	Support Vector Machine

Appendix A. Selected Representations

The set of eight representations $\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(8)}$ used to characterise PCNs are described in the following eight subsections.

Appendix A.1. Betti Numbers

Topological Data Analysis [99,100] is a novel data analysis approach useful whenever data can be described by topological structures (networks) as it consists in a set of techniques in order to extract information from data (starting from topological information) by means of dimensionality reduction, manifold estimation and persistent homology in order to study how components lying in a multi-dimensional space are connected (e.g., in terms of loops and multi-dimensional surfaces). One can start either from so-called point clouds, where objects are described by their coordinates in a multi-dimensional space equipped with notion of distance, or by explicitly providing the pairwise distance matrix between objects. Hereinafter, the former case is considered.

The most intuitive scenario in order to study how components lying in a multi-dimensional space are connected is (trivially) by studying the connectivity itself. To this end, it is worth defining simplices as (multi-dimensional) topological objects which can be extracted from a given topological space $\mathcal{X}$: points, lines, triangles and tetrahedrons are (for example) 0-dimensional, 1-dimensional, 2-dimensional, 3-dimensional simplices and, obviously, higher-order analogues exist. Simplices can be seen as descriptors of the space under analysis, thus worthy of attention when studying $\mathcal{X}$. Starting from simplices, it is possible to define simplicial complexes as properly-constructed collection of simplices able to capture the multi-scale organisation (or multi-way relations) in complex networks [101–103]. The two seminal examples of simplicial complexes are the Čech complex and the Vietoris–Rips complex [99,100,104,105], however due to its intuitiveness and lighter computational complexity, one in practice uses the latter. The Vietoris–Rips complex can be built according to the following rule: initially, all 0-dimensional simplices belong to the complex, then a given set of k points forms a $(k-1)$-dimensional simplicial complex to be included in the Vietoris–Rips complex if the pairwise distances are all less than or equal to a user-defined threshold ϵ.

The homology of a simplicial complex can be described by its Betti numbers. Formally, the ith Betti number is the rank of the ith homology group in the simplicial complex. Informally, the ith Betti number corresponds to the number of i-dimensional 'holes' in a topological surface. In this work, 3-dimensional graphs are considered and the first three Betti numbers have the following interpretations: the 0th Betti number is the number of connected components, the 1st Betti number is the number of 1-dimensional (circular) holes, the 2nd Betti number is the number of 2-dimensional holes (cavities). The Betti numbers vanish after the spatial dimension.

From the above Vietoris–Rips complex definition, it is clear that the choice of ϵ is critical as it somewhat defines the resolution of the simplicial complex. In many cases, one builds a sequence of Vietoris–Rips complexes as ϵ varies in order to study how 'holes' appear and disappear as the resolution changes and then selects a desired value $\epsilon^\star$ by studying the 'holes' lifetime in order to obtain a useful homology summary: in algebraic topology, this concept is known as persistence [106].

Instead of having a 'topological summary', following a previous work [46], the rationale is to keep proper track of the number of holes as ϵ changes. To this end, the range $\epsilon \in [4, 8]$ with sampling step 1 is considered, according to the PCN connectivity range. Hence, the first representation $\mathbf{X}^{(1)}$ sees each protein as a 15-length integer-valued vector obtained by the concatenation of $\mathbf{b}_4, \mathbf{b}_5, \mathbf{b}_6, \mathbf{b}_7, \mathbf{b}_8$, where $\mathbf{b}_i$ is (in turn) a 3-dimensional vector containing the first three Betti numbers for $\epsilon = i$. Technically speaking, for a given ϵ, the Vietoris–Rips complex can be evaluated in two steps [107]:

1. Build the Vietoris–Rips neighbourhood graph $\mathcal{G}_{VR}(\mathcal{V}, \mathcal{E})$: an undirected graph where edges between two nodes, say $v_i, v_j \in \mathcal{V}$, are scored if $d(v_i, v_j) \leq \epsilon$;
2. The set of maximal cliques in $\mathcal{G}_{VR}$ form the Vietoris–Rips complex.

Let $\partial_k : \mathcal{S}_k \to \mathcal{S}_{k-1}$ be the boundary operator, an incidence-like matrix which maps $\mathcal{S}_k$ (i.e., the set of simplices of order k) with the set of simplices of order $k-1$. The k-order homology group is defined as [108]:

$$\mathfrak{H}_k = \ker\{\partial_k\} / \mathrm{im}\{\partial_{k+1}\}, \tag{A1}$$

where $\ker\{\cdot\}$ and $\mathrm{im}\{\cdot\}$ denote the kernel and image operators. The rank of $\mathfrak{H}_k$, namely the k^{th} Betti number is then defined as [102]:

$$b^{(k)} = \mathrm{rank}\{\ker\{\partial_k\}\} - \mathrm{rank}\{\mathrm{im}\{\partial_{k+1}\}\} \tag{A2}$$

or, thanks to the Rank–Nullity theorem [109]:

$$b^{(k)} = (\dim\{\partial_k\} - \text{rank}\{\text{im}\{\partial_k\}\}) - \text{rank}\{\text{im}\{\partial_{k+1}\}\}, \tag{A3}$$

where the rank of the image corresponds to the plain matrix rank in linear algebra.

Appendix A.2. Centrality Measures

In graph theory and network analysis, centrality measures indicate the node/edge importance with respect to a given criterion. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph and let $\mathcal{V}$ and $\mathcal{E}$ be the set of nodes and edges, respectively. The following centrality measures are considered:

- The degree centrality [110] $DC(v_i)$ for node $v_i \in \mathcal{V}$, defined as the percentage of nodes connected to it:

$$DC(i) = \frac{1}{|\mathcal{V}| - 1} \sum_j \mathbf{A}_{i,j}, \tag{A4}$$

  where $\mathbf{A}$ is the adjacency matrix, defined as in Equation (A22). The normalisation coefficient $\frac{1}{|\mathcal{V}|-1}$ takes into account the maximum attainable degree in a simple graph, thus making the degree centrality in Equation (A4) independent from the number of nodes in the graph;
- The eigenvector centrality [110] highly rank nodes whether they are connected to other high-rank nodes. Formally, the eigenvector centrality $\mathbf{e}_i$ for node $v_i \in \mathcal{V}$ is given by:

$$\mathbf{e}_i = \frac{1}{\lambda} \sum_j \mathbf{A}_{j,i} \mathbf{e}_j \,, \tag{A5}$$

  where $\lambda \neq 0$ is a scalar constant. Equation (A5) can be re-written in matrix form as:

$$\lambda \mathbf{e} = \mathbf{e}\mathbf{A}. \tag{A6}$$

  Hence, the eigenvector centrality vector $\mathbf{e}$ is the left-hand eigenvector of the adjacency matrix $\mathbf{A}$ associated with the eigenvalue λ. According to the Perron–Frobenius theorem, by choosing λ as the largest (in absolute value) eigenvalue of $\mathbf{A}$, the solution $\mathbf{e}$ is unique and all its entries are positive;
- The PageRank centrality [110] $\mathbf{p}_i$ for node $v_i \in \mathcal{V}$ is given by:

$$\mathbf{p}_i = \alpha \sum_j \frac{\mathbf{A}_{j,i}}{D(v_j)} \mathbf{p}_j + \frac{1-\alpha}{|\mathcal{V}|}, \tag{A7}$$

  where α is a scalar constant (usually $\alpha = 0.85$) and $D(v_j)$ is the degree of node v_j. It is worth remarking the difference between degree and degree centrality: the degree is the number of nodes connected to a given node (namely Equation (A4) without the normalisation term), whereas the degree centrality includes the normalisation term. As in the eigenvector centrality case, Equation (A7) can be re-written in matrix form as:

$$\mathbf{p} = \alpha \mathbf{p} \mathbf{D}^{-1} \mathbf{A} + \boldsymbol{\beta}, \tag{A8}$$

  where $\mathbf{D}^{-1}$ is a diagonal matrix whose ith element equals $1/D(v_i)$ and $\boldsymbol{\beta}$ is a vector whose elements are all equal to $\frac{1-\alpha}{|\mathcal{V}|}$;
- The Katz centrality [110,111] $\mathbf{k}_i$ for node $v_i \in \mathcal{V}$ is given by:

$$\mathbf{k}_i = \alpha \sum_j \mathbf{A}_{i,j} \mathbf{k}_j + \beta, \tag{A9}$$

where β controls the initial centrality (first neighbourhood weights) and $\alpha < 1/\lambda_{\max}$ attenuates the importance with respect to higher-order neighbours (in turn, $\lambda_{\max}$ is the largest eigenvalue of $\mathbf{A}$). It is worth noting that if $\alpha = 1/\lambda_{\max}$ and $\beta = 0$, the Katz centrality equals the eigenvector centrality;

- The closeness centrality [110] $CC(v_i)$ for node $v_i \in \mathcal{V}$ is the inverse sum of shortest path distances between node $v_i \in \mathcal{V}$ and all other $n-1$ reachable nodes. Formally:

$$CC(v_i) = \frac{n-1}{|\mathcal{V}|-1} \frac{n-1}{\sum_{j=1}^{n-1} \delta(v_i, v_j)}, \tag{A10}$$

  where $\delta(\cdot,\cdot)$ indicates the shortest path distance. The normalisation factor takes into account the graph size in order to allow comparison between nodes of graphs having different sizes, also in case of multiple connected components [112]. Indeed, n can be seen as the number of nodes in the connected component in which v_i lies. In case of one connected component, the scale factor $(n-1)/(|\mathcal{V}|-1)$ can be neglected since $n = |\mathcal{V}|$;
- The betweenness centrality [110] $BC(v_i)$ quantifies how many times a given node $v_i \in \mathcal{V}$ acts as a bridge along the shortest paths between any two nodes:

$$BC(v_i) = \sum_{v_i \neq v_j \neq v_k} \frac{s^{(v_i)}(v_j, v_k)}{s(v_j, v_k)}, \tag{A11}$$

  where $s(v_j, v_k)$ is the number of shortest paths from v_i to v_j and $s^{(v_i)}(v_j, v_k)$ is the number of shortest paths from v_i to v_j passing through v_i. As in the closeness centrality case, it is often customary to normalise the betweenness centrality in order to avoid dependency from the number of nodes, thus:

$$BC(v_i) := \frac{2 \cdot BC(v_i)}{(|\mathcal{V}|-1) \cdot (|\mathcal{V}|-2)}; \tag{A12}$$

- The edge betweenness centrality [113] $EBC(e_i)$ is the edge counterpart of the "standard" (node) betweenness centrality as it quantifies how many times a given edge $e_i \in \mathcal{E}$ acts as a bridge along the shortest paths between two nodes:

$$EBC(e_i) = \sum_{v_i, v_j \in \mathcal{V}} \frac{s^{(e_i)}(v_i, v_j)}{s(v_i, v_j)}, \tag{A13}$$

  where $s^{(e_i)}(v_i, v_j)$ is the number of shortest paths between nodes v_i and v_j passing through edge e_i and $s(v_i, v_j)$ is the total number of shortest paths between nodes v_i and v_j. As in the "standard" betweenness centrality, the edge betweenness centrality can be normalised as follows:

$$EBC(e_i) := \frac{2 \cdot EBC(e_i)}{(|\mathcal{V}|-1) \cdot |\mathcal{V}|}; \tag{A14}$$

- The load centrality [113,114] $LC(v_i)$ for node $v_i \in \mathcal{V}$ is the percentage of the total number of shortest paths passing through v_i;
- The edge load centrality $ELC(e_i)$ for edge $e_i \in \mathcal{E}$ is the edge-related counterpart of the load centrality (like betweenness vs. edge betweenness): it is defined as the percentage of the total number of shortest paths crossing edge e_i;
- The subgraph centrality [115] $SC(v_i)$ for node $v_i \in \mathcal{V}$ is the sum of (weighted) closed walks (i.e., connected subgraphs) starting and ending at v_i (the longer the walk, the lower the weight). It can be evaluated thanks to the spectral decomposition of the adjacency matrix, which reads as $\mathbf{A} =$

$\mathbf{B}\mathbf{\Lambda}^{(\mathbf{A})}\mathbf{B}^T$ where $\mathbf{\Lambda}^{(\mathbf{A})} = \text{diag}\left\{\lambda_1^{(\mathbf{A})}, \ldots, \lambda_{|\mathcal{V}|}^{(\mathbf{A})}\right\}$ is a diagonal matrix containing the eigenvalues in increasing order and $\mathbf{B}$ contains the corresponding unitary-length eigenvectors, thus:

$$SC(v_i) = \sum_{j=1}^{|\mathcal{V}|} e^{\lambda_j^{(\mathbf{A})}} \left(\mathbf{b}_j(v_i)\right)^2, \tag{A15}$$

where λ_j and $\mathbf{b}_j$ are the eigenvalue and eigenvector associated to node $v_j \in \mathcal{V}$ and $\mathbf{b}_j(v_i)$ indicates the value related to v_i in the j^{th} eigenvector;

- The Estrada Index [116] $EI(\mathcal{G})$ of a graph $\mathcal{G}$ quantifies the compactness (or 'folding', since the Estrada Index was indeed originally proposed in order to study molecular 3D compactness) of a graph starting from the spectral decomposition of the adjacency matrix (as in the subgraph centrality):

$$EI(\mathcal{G}) = \sum_{j=1}^{|\mathcal{V}|} e^{\lambda_j^{(\mathbf{A})}}; \tag{A16}$$

- The harmonic centrality [117] $HC(v_i)$ is the sum of inverse shortest paths distances from a given node $v_i \in \mathcal{V}$ to all other nodes:

$$HC(v_i) = \sum_{\substack{j=1 \\ j \neq i}}^{|\mathcal{V}|} \frac{1}{\delta(v_i, v_j)}; \tag{A17}$$

- The global reaching centrality [118] $GRC(\mathcal{G})$ of a graph $\mathcal{G}$ is the average (over all nodes) of the difference between the maximum local reaching centrality and each node's local reaching centrality. Formally:

$$GRC(\mathcal{G}) = \frac{\sum_{i=1}^{|\mathcal{V}|} \left(LRC_{\max} - LRC(v_i)\right)}{|\mathcal{V}| - 1}, \tag{A18}$$

  where $LRC(v_i)$ is the local reaching centrality of node $v_i \in \mathcal{V}$ and $LRC_{\max}$ is the maximum local reaching centrality amongst all nodes. In turn, the local reaching centrality for a given node v_i is defined as the percentage of nodes reachable from v_i;
- The average clustering coefficient [119] $ACC(\mathcal{G})$ of a graph $\mathcal{G}$ is given by:

$$ACC(\mathcal{G}) = \frac{1}{|\mathcal{V}|} \sum_{i=1}^{|\mathcal{V}|} cc(v_i), \tag{A19}$$

  where $cc(v_i)$ is the clustering coefficient for node v_i, defined as:

$$cc(v_i) = \frac{2 \cdot tri(v_i)}{DC(v_i) \cdot (DC(v_i) - 1)}, \tag{A20}$$

  where, in turn, $tri(v_i)$ is the number of triangles passing through node v_i and $D(v_i)$ is its degree;
- The average neighbour degree [120] $AND(v_i)$ of node $v_i \in \mathcal{V}$ is given by:

$$AND(v_i) = \frac{1}{|\mathcal{N}(v_i)|} \sum_{v_j \in \mathcal{N}(v_i)} D(v_j), \tag{A21}$$

  where $\mathcal{N}(v_i)$ is the set of neighbours of node v_i.

Apart from ACC, EI and GRC, which are global characteristics (i.e., related to the whole graph), the others are local characteristics (i.e., related to each node or edge). As such, it is impossible to compare graphs having different sizes (number of nodes and/or edges) by considering their local centralities. The second representation $\mathbf{X}^{(2)}$ sees each protein as a 27-length real-valued vector containing $\bar{DC}$, $\tilde{DC}$, $\bar{\mathbf{e}}$, $\tilde{\mathbf{e}}$, $\bar{\mathbf{p}}$, $\tilde{\mathbf{p}}$, $\bar{\mathbf{k}}$, $\tilde{\mathbf{k}}$, $\bar{CC}$, $\tilde{CC}$, $\bar{BC}$, $\tilde{BC}$, $\bar{EBC}$, $\tilde{EBC}$, $\bar{LC}$, $\tilde{LC}$, $\bar{ELC}$, $\tilde{ELC}$, $\bar{SC}$, $\tilde{SC}$, EI,

$\bar{HC}$, $\tilde{HC}$, GRC, ACC, $\bar{AND}$, $\tilde{AND}$ (where bar and tilde indicate the average and standard deviation centrality across nodes/edges).

Appendix A.3. Energy and Laplacian Energy

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph and let $\mathcal{V}$ and $\mathcal{E}$ be the set of nodes and edges, respectively. Since in this work unweighted and undirected graphs are considered, the adjacency matrix **A** is a binary $|\mathcal{V}| \times |\mathcal{V}|$ matrix defined as:

$$\mathbf{A}_{i,j} = \begin{cases} 1 & \text{if } (v_i, v_j) \in \mathcal{E} \\ 0 & \text{otherwise.} \end{cases} \tag{A22}$$

From **A**, it is possible to define the diagonal $|\mathcal{V}| \times |\mathcal{V}|$ degree matrix as:

$$\mathbf{D}_{i,j} = \begin{cases} D(i) & \text{if } i = j \\ 0 & \text{otherwise,} \end{cases} \tag{A23}$$

where $D(i)$ is the degree of the ith node. In turn, from **A** and **D**, it is possible to define the Laplacian matrix as:

$$\mathbf{L} = \mathbf{D} - \mathbf{A}, \tag{A24}$$

The spectrum and Laplacian spectrum of $\mathcal{G}$ are defined as the set of eigenvalues from **A** and **L**, respectively [121]:

$$\boldsymbol{\lambda}^{(\mathbf{A})} = \left\{ \lambda_1^{(\mathbf{A})}, \ldots, \lambda_{|\mathcal{V}|}^{(\mathbf{A})} \right\}, \tag{A25}$$

$$\boldsymbol{\lambda}^{(\mathbf{L})} = \left\{ \lambda_1^{(\mathbf{L})}, \ldots, \lambda_{|\mathcal{V}|}^{(\mathbf{L})} \right\}. \tag{A26}$$

From Equations (A25) and (A26), it is possible to define the graph energy E and the Laplacian energy LE as

$$E = \sum_{i=1}^{|\mathcal{V}|} \left| \lambda_i^{(\mathbf{A})} \right|, \tag{A27}$$

$$LE = \sum_{i=1}^{|\mathcal{V}|} \left| \lambda_i^{(\mathbf{L})} - \frac{2|\mathcal{E}|}{|\mathcal{V}|} \right|. \tag{A28}$$

The third representation $\mathbf{X}^{(3)}$ sees each protein as a 2-length real-valued vector containing E and LE.

Appendix A.4. Nodes Functional Cartography

Guimerà and Amaral in their seminal work [122] proposed a methodology in order to extract functional modules from a graph by maximising its modularity using simulated annealing [123]. Their definition of modularity takes into account both within-module degree and between-module degree with the idea that a good graph partition (i.e., high modularity) must have many within-module links and few between-module links.

Each node is then assigned with two scores: the z-score and the participation coefficient P. The former measures how well-connected a given node is with respect to other nodes in its own module. The latter quantifies how many connections a given nodes has with respect to nodes belonging to different modules.

The $z - P$ plane has been heuristically divided into seven regions and each node can be classified into one of seven functional roles by considering its z-score and its participation coefficient P. Nodes having $z < 2.5$ are non-hubs, whereas nodes having $z \geq 2.5$ are hubs. In turn, non-hub nodes can be divided in: ultra-peripherals (if $P \leq 0.05$), peripherals (if $P \in (0.05, 0.62]$), non-hub connectors (if

$P \in (0.62, 0.8]$) and non-hub kinless (if $P > 0.8$). Finally, hub nodes can be divided in: provincial hubs (if $P \leq 0.3$), connector hubs (if $P \in (0.3, 0.75]$) and kinless hubs (if $P > 0.75$).

The fourth representation $\mathbf{X}^{(4)}$ sees each protein as an 8-length real-valued vector containing the modularity (as returned by the simulated annealing) and the percentage of nodes belonging to each functional role.

Appendix A.5. Heat Content Invariant

From the graph Laplacian and degree matrices (Equations (A24) and (A23), respectively), the normalised Laplacian matrix can be evaluated as:

$$\tilde{\mathbf{L}} = \mathbf{D}^{-\frac{1}{2}} \mathbf{L} \mathbf{D}^{-\frac{1}{2}}, \tag{A29}$$

The spectral decomposition of $\tilde{\mathbf{L}}$ reads as:

$$\tilde{\mathbf{L}} = \mathbf{V} \boldsymbol{\Lambda} \mathbf{V}^T, \tag{A30}$$

where $\boldsymbol{\Lambda} = \text{diag}\left\{\lambda_1^{(\tilde{\mathbf{L}})}, \ldots, \lambda_{|\mathcal{V}|}^{(\tilde{\mathbf{L}})}\right\}$ is a diagonal matrix containing the eigenvalues in increasing order and $\mathbf{V}$ contains the corresponding unitary-length eigenvectors.

The heat equation associated to $\tilde{\mathbf{L}}$ is given by [124,125]:

$$\frac{\partial \mathbf{H}(t)}{\partial t} = -\tilde{\mathbf{L}} \mathbf{H}(t), \tag{A31}$$

where $\mathbf{H}(t)$ is the $|\mathcal{V}| \times |\mathcal{V}|$ heat kernel matrix at time t. The heat content $HC(t)$ of $\mathbf{H}(t)$ is given by:

$$\begin{aligned} HC(t) &= \sum_{v_i \in \mathcal{V}} \sum_{v_j \in \mathcal{V}} \mathbf{H}_{i,j}(t) \\ &= \sum_{v_i \in \mathcal{V}} \sum_{v_j \in \mathcal{V}} \sum_{k=1}^{|\mathcal{V}|} \exp\left\{-\lambda_k^{(\tilde{\mathbf{L}})} t\right\} \mathbf{v}_k(v_i) \mathbf{v}_k(v_j), \end{aligned} \tag{A32}$$

where $\mathbf{v}_k(v_i)$ is the value related to node v_i in the kth eigenvector.

The MacLaurin series for the negative exponential reads as:

$$\exp\left\{-\lambda_k^{(\tilde{\mathbf{L}})} t\right\} = \sum_{m=0}^{\infty} \frac{\left(-\lambda_k^{(\tilde{\mathbf{L}})} t\right)^m t^m}{m!} \tag{A33}$$

and substituting Equation (A33) in Equation (A32) yields:

$$HC(t) = \sum_{v_i \in \mathcal{V}} \sum_{v_j \in \mathcal{V}} \sum_{k=1}^{|\mathcal{V}|} \sum_{m=0}^{\infty} \frac{\left(-\lambda_k^{(\tilde{\mathbf{L}})} t\right)^m t^m}{m!} \mathbf{v}_k(v_i) \mathbf{v}_k(v_j) \tag{A34}$$

By re-writing Equation (A32) in terms of power series as:

$$HC(t) = \sum_{m=0}^{\infty} q_m t^m, \tag{A35}$$

where the set of coefficients q_m are the so-called heat content invariants and can be evaluated in closed-form as:

$$q_m = \sum_{i=1}^{|\mathcal{V}|} \left(\left(\sum_{v \in \mathcal{V}} \mathbf{v}_i(v) \right)^2 \right) \frac{\left(-\lambda_i^{(\tilde{\mathbf{L}})}\right)^m}{m!}. \tag{A36}$$

The fifth representation $\mathbf{X}^{(5)}$ sees each protein as a 4-length real-valued vector containing the first four coefficients from Equation (A36); that is q_1, q_2, q_3, q_4.

Appendix A.6. Heat Kernel Trace

Recalling the heat equation from Equation (A31) and the spectral decomposition of the normalised Laplacian matrix from Equation (A30), the solution to the former (already in Equation (A32)) reads as:

$$\begin{aligned}\mathbf{H}(t) &= \exp\left\{-t\tilde{\mathbf{L}}\right\} = \mathbf{V}\exp\left\{-t\boldsymbol{\Lambda}\right\}\mathbf{V}^T \\ &= \sum_{i=1}^{|\mathcal{V}|}\exp\left\{-\lambda_i^{(\tilde{\mathbf{L}})}t\right\}\mathbf{v}_i\mathbf{v}_i^T.\end{aligned} \tag{A37}$$

The heat kernel trace is evaluated by taking the trace of $\mathbf{H}(t)$:

$$HT(t) = \mathrm{Tr}\left\{\mathbf{H}(t)\right\} = \sum_{i=1}^{|\mathcal{V}|}\exp\left\{-\lambda_i^{(\tilde{\mathbf{L}})}t\right\}. \tag{A38}$$

The sixth representation $\mathbf{X}^{(6)}$ sees each protein as a 10-length real-valued vector containing the heat kernel trace for $t = 1, 2, \ldots, 10$. These values for t have been chosen by visual inspection: indeed, for $t > 10$ the heat kernel trace decay makes proteins undistinguishable one another.

Appendix A.7. Size

The seventh representation $\mathbf{X}^{(7)}$ sees each protein as a 4-length real-valued vector containing the number of nodes, the number of edges, the number of protein chains and the radius of gyration. Whilst the first two items are rather straightforward, the latter two items deserve some further comments. Proteins are composed by one or more amino-acids chains (linear polymers), thus the number of chains may impact on the overall protein size. Finally, the radius of gyration [126] is a measure of how-compact is the overall folded protein structure with respect to its centre of mass.

Appendix A.8. Normalised Laplacian Spectral Density

Recalling the spectral decomposition of the normalised Laplacian matrix from Equation (A30), let $\boldsymbol{\lambda}^{(\tilde{\mathbf{L}})} = \left\{\lambda_1^{(\tilde{\mathbf{L}})}, \ldots, \lambda_{|\mathcal{V}|}^{(\tilde{\mathbf{L}})}\right\}$ be the normalised Laplacian spectrum (namely, the set of eigenvalues from $\tilde{\mathbf{L}}$). One of the interesting properties of the normalised Laplacian matrix is that its spectrum lies in range $[0, 2]$, regardless of the underlying graph [127]. The size of the spectrum, however, equals the number of nodes and therefore one cannot easily compare graphs having different sizes just by considering their respective spectra. In order to overcome this problem, following previous works [45,50], it is possible to estimate the (normalised Laplacian) spectral density using a kernel density estimator (also known as Parzen window [128]) equipped with the Gaussian kernel. The spectral density thus has the form:

$$p(x) = \frac{1}{|\mathcal{V}|}\sum_{i=1}^{|\mathcal{V}|}\frac{1}{\sqrt{2\pi\sigma^2}}\cdot\exp\left\{\frac{-\left(x-\lambda_i^{(\tilde{\mathbf{L}})}\right)^2}{2\sigma^2}\right\}, \tag{A39}$$

where σ is the kernel bandwidth which determines the estimate resolution. Following [50], the Scott's rule [129] has been used in order to determine the proper bandwidth value, hence:

$$\sigma = \frac{3.5\cdot\mathrm{std}\left\{\boldsymbol{\lambda}^{(\tilde{\mathbf{L}})}\right\}}{|\boldsymbol{\lambda}^{(\tilde{\mathbf{L}})}|^{1/3}}. \tag{A40}$$

In this manner, the bandwidth scales in a graph-wise fashion by considering each graph's spectrum size (denominator) and its standard deviation (numerator). Let $\mathcal{G}_1$ and $\mathcal{G}_2$ be two graphs, their distance can be evaluated by considering the ℓ_2 norm between their respective spectral densities $p_1(x)$ and $p_2(x)$:

$$d(\mathcal{G}_1, \mathcal{G}_2) = \int_0^2 (p_1(x) - p_2(x))^2 dx. \tag{A41}$$

The same operation can be carried in the discrete domain by extracting n samples from $p(x)$ (equal for all graphs) and the latter collapses into the standard Euclidean distance.

The eighth representation $\mathbf{X}^{(8)}$ sees each protein as an 100-length real-valued vector containing $n = 100$ samples uniformly drawn from their respective normalised Laplacian spectral densities.

Appendix B. Selected Prototypes

In the following, the sets of proteins elected as prototypes for each of the seven classification problems are shown. In order to shrink the output size, our a posteriori analysis has been carried only on proteins which have been selected in all of the five runs of the genetic algorithm (in order to remove 'spurious' representatives due to randomness in the optimisation procedure).

Table A1. Selected proteins in order to discriminate EC 1 (oxidoreductases) vs. all the rest.

PDB ID	Notes/Description
1KOF	Transferase
1XFG	Transferase
3E2R	Oxydoreductase
4TS9	Transferase
1ZDM	Signalling Protein
1MPG	Hydrolase
1QQQ	Transferase

Table A2. Selected proteins in order to discriminate EC 2 (transferases) vs. all the rest.

PDB ID	Notes/Description
3EDC	LAC repressor (signalling protein)
1DKL	Hydrolase
1JKJ	Ligase
2DBI	Unknown function
3UCS	Chaperone
1LX7	Transferase
2GAR	Transferase
3ILI	Transferase
1S08	Transferase
4IXM	Hydrolase
4XTJ	Isomerase
1KW1	Lyase
1BDH	Transcription factor (DNA-binding)
4PC3	Elongation factor (RNA-binding)
5G1L	Isomerase

Table A3. Selected proteins in order to discriminate EC 3 (hydrolases) vs. all the rest.

PDB ID	Notes/Description
4RZS	Transcription factor (signalling protein)
1ZDM	Signalling protein
3I7R	Lyase
1HW5	Signalling protein
1SO5	Lyase

Table A4. Selected proteins in order to discriminate EC 4 (lyases) vs. all the rest.

PDB ID	Notes/Description
2BWX	Hydrolase
3UWM	Oxydoreductase
2H71	Electron transport
1D7A	Lyase
4DAP	DNA-binding
1SPV	Structural genomics, unknown function
1EXD	Ligase + RNA-binding
1X83	Isomerase
3ILJ	Transferase
2D4U	Signalling protein
1JNW	Oxydoreductase
1TRE	Oxydoreductase
1ZPT	Oxydoreductase
3LGU	Hydrolase
1IB6	Oxydoreductase
3C0U	Structural genomics, unknown function
5GT2	Oxydoreductase
2RN2	Hydrolase
4L4Z	Transcription regulator
3CMR	Hydrolase
1NQF	Transport protein
1GPQ	Hydrolase
4ODM	Isomerase + chaperone
2NPG	Transport protein
2UAG	Ligase
1OVG	Transferase
3AVU	Transferase
1RBV	Hydrolase
5AB1	Cell adhesion
1TMM	Transferase
4NIY	Hydrolase
4WR3	Isomerase

Table A5. Selected proteins in order to discriminate EC 5 (isomerases) vs. all the rest.

PDB ID	Notes/Description
4ITX	Lyase
2BWW	Hydrolase
5IU6	Transferase
1ODD	Gene regulatory
5G5G	Oxydoreductase
1G7X	Transferase
2E0Y	Transferase
2SCU	Ligase
1HO4	Hydrolase
3RGM	Transport Protein
1OAC	Oxydoreductase
5MUC	Oxydoreductase
3OGD	Hydrolase + DNA binding
4K34	Membrane protein
1Q0L	Oxydoreductase
1G58	Isomerase
5M3B	Transport protein
2WOH	Oxydoreductase
2PJP	Translation regulation (RNA-binding)

Table A6. Selected proteins in order to discriminate EC 6 (ligases) vs. all the rest.

PDB ID	Notes/Description
2OLQ	Lyase
1JDI	Isomerase
4NIG	Oxydoreductase + DNA-binding
5T03	Transferase
5FNN	Oxydoreductase
2Z9D	Oxydoreductase
2V3Z	Hydrolase
4ARI	Ligase + RNA-binding
3LBS	Transport protein
4QGS	Oxydoreductase
5B7F	Oxydoreductase
2ABH	Transferase

Table A7. Selected proteins in order to discriminate not-enzymes vs. all the rest.

PDB ID	Notes/Description
1SPA	Transferase
2YH9	Membrane protein
1NQF	Transport protein
1LDI	Transport protein
1TIK	Hydrolase
1MWI	Hydrolase + DNA-binding
1GEW	Transferase
5CKH	Hydrolase
3ABQ	Lyase
3B6M	Oxydoreductase

References

1. Bianchi, F.M.; Scardapane, S.; Livi, L.; Uncini, A.; Rizzi, A. An interpretable graph-based image classifier. In Proceedings of the 2014 International Joint Conference on Neural Networks (IJCNN), Beijing, China, 6–11 July 2014; pp. 2339–2346. [CrossRef]
2. Bianchi, F.M.; Scardapane, S.; Rizzi, A.; Uncini, A.; Sadeghian, A. Granular Computing Techniques for Classification and Semantic Characterization of Structured Data. *Cogn. Comput.* **2016**, *8*, 442–461. [CrossRef]
3. Del Vescovo, G.; Rizzi, A. Online Handwriting Recognition by the Symbolic Histograms Approach. In Proceedings of the 2007 IEEE International Conference on Granular Computing (GRC 2007), San Jose, CA, USA, 2–4 November 2007; pp. 686. [CrossRef]
4. Giuliani, A.; Filippi, S.; Bertolaso, M. Why network approach can promote a new way of thinking in biology. *Front. Genet.* **2014**, *5*, 83. [CrossRef]
5. Di Paola, L.; De Ruvo, M.; Paci, P.; Santoni, D.; Giuliani, A. Protein contact networks: an emerging paradigm in chemistry. *Chem. Rev.* **2012**, *113*, 1598–1613.[CrossRef]
6. Krishnan, A.; Zbilut, J.P.; Tomita, M.; Giuliani, A. Proteins as networks: usefulness of graph theory in protein science. *Curr. Protein Pept. Sci.* **2008**, *9*, 28–38. [CrossRef]
7. Jeong, H.; Tombor, B.; Albert, R.; Oltvai, Z.N.; Barabási, A.L. The large-scale organization of metabolic networks. *Nature* **2000**, *407*, 651. [CrossRef]
8. Di Paola, L.; Giuliani, A. *Protein–Protein Interactions: The Structural Foundation of Life Complexity*; American Cancer Society: Atlanta, GA, USA, 2017; pp. 1–12. [CrossRef]
9. Wuchty, S. Scale-Free Behavior in Protein Domain Networks. *Mol. Biol. Evol.* **2001**, *18*, 1694–1702. [CrossRef] [PubMed]
10. Martino, A.; Giuliani, A.; Rizzi, A. Granular Computing Techniques for Bioinformatics Pattern Recognition Problems in Non-metric Spaces. In *Computational Intelligence for Pattern Recognition*; Pedrycz, W., Chen, S.M., Eds.; Springer International Publishing: Cham, Switzerland, 2018; pp. 53–81.

11. Ieracitano, C.; Mammone, N.; Hussain, A.; Morabito, F.C. A novel multi-modal machine learning based approach for automatic classification of EEG recordings in dementia. *Neural Netw.* **2020**, *123*, 176–190. [CrossRef]
12. Cinti, A.; Bianchi, F.M.; Martino, A.; Rizzi, A. A Novel Algorithm for Online Inexact String Matching and its FPGA Implementation. *Cogn. Comput.* **2019**. [CrossRef]
13. Bunke, H. On a relation between graph edit distance and maximum common subgraph. *Pattern Recognit. Lett.* **1997**, *18*, 689–694. [CrossRef]
14. Bargiela, A.; Pedrycz, W. *Granular Computing: An Introduction*; Kluwer Academic Publishers: Boston, MA, USA, 2003.
15. Pedrycz, W. Granular computing: an introduction. In Proceedings of the 9th IFSA World Congress and 20th NAFIPS International Conference, Vancouver, BC, Canada, 25–28 July 2001; Volume 3, pp. 1349–1354. [CrossRef]
16. Bargiela, A.; Pedrycz, W. Granular Computing. In *Handbook on Computational Intelligence*; World Scientific Publishers: Singapore, 2016; Chapter 2, pp. 43–66. [CrossRef]
17. Singh, P.K. Similar Vague Concepts Selection Using Their Euclidean Distance at Different Granulation. *Cogn. Comput.* **2018**, *10*, 228–241. [CrossRef]
18. Lin, T.Y.; Yao, Y.Y.; Zadeh, L.A. *Data Mining, Rough Sets and Granular Computing*; Springer: Berlin/Heidelberg, Germany, 2013; Volume 95.
19. Bianchi, F.M.; Livi, L.; Rizzi, A.; Sadeghian, A. A Granular Computing approach to the design of optimized graph classification systems. *Soft Comput.* **2014**, *18*, 393–412. [CrossRef]
20. Rizzi, A.; Del Vescovo, G.; Livi, L.; Frattale Mascioli, F.M. A new Granular Computing approach for sequences representation and classification. In Proceedings of the 2012 International Joint Conference on Neural Networks (IJCNN), Brisbane, Australia, 10–15 June 2012; pp. 1–8. [CrossRef]
21. Del Vescovo, G.; Rizzi, A. Automatic Classification of Graphs by Symbolic Histograms. In Proceedings of the 2007 IEEE International Conference on Granular Computing (GRC 2007), San Jose, CA, USA, 2–4 November 2007; p. 410. [CrossRef]
22. Baldini, L.; Martino, A.; Rizzi, A. Stochastic Information Granules Extraction for Graph Embedding and Classification. In Proceedings of the 11th International Joint Conference on Computational Intelligence—Volume 1: NCTA, (IJCCI 2019), Vienna, Austria, 17–19 September 2019; pp. 391–402. [CrossRef]
23. Martino, A.; Giuliani, A.; Todde, V.; Bizzarri, M.; Rizzi, A. Metabolic networks classification and knowledge discovery by information granulation. *Comput. Biol. Chem.* **2020**, *84*, 107187. [CrossRef] [PubMed]
24. Martino, A.; Frattale Mascioli, F.M.; Rizzi, A. On the Optimization of Embedding Spaces via Information Granulation for Pattern Recognition. In Proceedings of the 2020 International Joint Conference on Neural Networks (IJCNN), Glasgow, UK, 19–24 July 2020.
25. Martino, A.; Giuliani, A.; Rizzi, A. (Hyper)Graph Embedding and Classification via Simplicial Complexes. *Algorithms* **2019**, *12*. [CrossRef]
26. Pękalska, E.; Duin, R.P. *The Dissimilarity Representation for Pattern Recognition: Foundations and Applications*; World Scientific: London, UK, 2005.
27. Pękalska, E.; Duin, R.P.; Paclík, P. Prototype selection for dissimilarity-based classifiers. *Pattern Recognit.* **2006**, *39*, 189–208. [CrossRef]
28. Duin, R.P.; Pękalska, E. The dissimilarity space: Bridging structural and statistical pattern recognition. *Pattern Recognit. Lett.* **2012**, *33*, 826–832. [CrossRef]
29. Shawe-Taylor, J.; Cristianini, N. *Kernel Methods for Pattern Analysis*; Cambridge University Press: Cambridge, UK, 2004.
30. Schölkopf, B.; Smola, A.J. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*; MIT Press: Cambridge, MA, USA, 2002.
31. Cristianini, N.; Shawe-Taylor, J. *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*; Cambridge University Press: Cambridge, UK, 2000.

32. Mercer, J. Functions of positive and negative type, and their connection with the theory of integral equations. *R. Soc.* **1909**, *209*, 415–446.
33. Cover, T.M. Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. *IEEE Trans. Electron. Comput.* **1965**, *EC-14*, 326–334. [CrossRef]
34. Boser, B.E.; Guyon, I.; Vapnik, V. A training algorithm for optimal margin classifiers. In Proceedings of the Fifth Annual Workshop on Computational Learning Theory, Pittsburgh, PA, USA, 27–29 July 1992; pp. 144–152. [CrossRef]
35. Cortes, C.; Vapnik, V. Support-vector networks. *Mach. Learn.* **1995**, *20*, 273–297. [CrossRef]
36. Haasdonk, B. Feature space interpretation of SVMs with indefinite kernels. *IEEE Trans. Pattern Anal. Mach. Intell.* **2005**, *27*, 482–492. [CrossRef]
37. Laub, J.; Müller, K.R. Feature Discovery in Non-Metric Pairwise Data. *J. Mach. Learn. Res.* **2004**, *5*, 801–818.
38. Ong, C.S.; Mary, X.; Canu, S.; Smola, A.J. Learning with Non-positive Kernels. In Proceedings of the Twenty-first International Conference on Machine Learning, Banff, AL, Canada, 4–8 July 2004. [CrossRef]
39. Chen, Y.; Gupta, M.R.; Recht, B. Learning kernels from indefinite similarities. In Proceedings of the 26th Annual International Conference on Machine Learning, Montreal, QC, Canada, 14–18 June 2009; pp. 145–152. [CrossRef]
40. Chen, Y.; Garcia, E.K.; Gupta, M.R.; Rahimi, A.; Cazzanti, L. Similarity-based classification: Concepts and algorithms. *J. Mach. Learn. Res.* **2009**, *10*, 747–776.
41. Pauling, L.; Corey, R.B.; Branson, H.R. The structure of proteins: two hydrogen-bonded helical configurations of the polypeptide chain. *Proc. Natl. Acad. Sci. USA* **1951**, *37*, 205–211. [CrossRef] [PubMed]
42. Livi, L.; Giuliani, A.; Sadeghian, A. Characterization of graphs for protein structure modeling and recognition of solubility. *Curr. Bioinform.* **2016**, *11*, 106–114. [CrossRef]
43. Livi, L.; Giuliani, A.; Rizzi, A. Toward a multilevel representation of protein molecules: Comparative approaches to the aggregation/folding propensity problem. *Inf. Sci.* **2016**, *326*, 134–145. [CrossRef]
44. De Santis, E.; Martino, A.; Rizzi, A.; Frattale Mascioli, F.M. Dissimilarity Space Representations and Automatic Feature Selection for Protein Function Prediction. In Proceedings of the 2018 International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, Brazil, 8–13 July 2018; pp. 1–8. [CrossRef]
45. Martino, A.; Maiorino, E.; Giuliani, A.; Giampieri, M.; Rizzi, A. Supervised Approaches for Function Prediction of Proteins Contact Networks from Topological Structure Information. In *Image Analysis, Proceedings of the 20th Scandinavian Conference, SCIA 2017, Tromsø, Norway, 12–14 June 2017*; Sharma, P., Bianchi, F.M., Eds.; Springer International Publishing: Cham, Switzerland, 2017; pp. 285–296.
46. Martino, A.; Rizzi, A.; Frattale Mascioli, F.M. Supervised Approaches for Protein Function Prediction by Topological Data Analysis. In Proceedings of the 2018 International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, Brazil, 8–13 July 2018; pp. 1–8. [CrossRef]
47. Livi, L.; Maiorino, E.; Giuliani, A.; Rizzi, A.; Sadeghian, A. A generative model for protein contact networks. *J. Biomol. Struct. Dyn.* **2016**, *34*, 1441–1454. [CrossRef]
48. Livi, L.; Maiorino, E.; Pinna, A.; Sadeghian, A.; Rizzi, A.; Giuliani, A. Analysis of heat kernel highlights the strongly modular and heat-preserving structure of proteins. *Phys. A Stat. Mech. Its Appl.* **2016**, *441*, 199–214. [CrossRef]
49. Maiorino, E.; Livi, L.; Giuliani, A.; Sadeghian, A.; Rizzi, A. Multifractal characterization of protein contact networks. *Phys. A Stat. Mech. Its Appl.* **2015**, *428*, 302–313. [CrossRef]
50. Maiorino, E.; Rizzi, A.; Sadeghian, A.; Giuliani, A. Spectral reconstruction of protein contact networks. *Phys. A Stat. Mech. Its Appl.* **2017**, *471*, 804–817. [CrossRef]
51. Webb, E.C. *Enzyme Nomenclature 1992. Recommendations of the Nomenclature Committee of the International Union of Biochemistry and Molecular Biology on the Nomenclature and Classification of Enzymes*, 6th ed.; Academic Press: Cambridge, MA, USA, 1992.
52. Livi, L.; Rizzi, A.; Sadeghian, A. Optimized dissimilarity space embedding for labeled graphs. *Inf. Sci.* **2014**, *266*, 47–64. [CrossRef]
53. Bishop, C.M. *Pattern Recognition and Machine Learning*; Springer: New York, NY, USA, 2006.

54. Vapnik, V. *Statistical Learning Theory*; Wiley: New York, NY, USA, 1998.
55. Sonnenburg, S.; Rätsch, G.; Schäfer, C.; Schölkopf, B. Large scale multiple kernel learning. *J. Mach. Learn. Res.* **2006**, *7*, 1531–1565.
56. Lewis, D.P.; Jebara, T.; Noble, W.S. Nonstationary kernel combination. In Proceedings of the 23rd International Conference on Machine Learning, Pittsburgh, PA, USA, 25–29 June 2006; pp. 553–560. [CrossRef]
57. Lanckriet, G.R.; Cristianini, N.; Bartlett, P.; Ghaoui, L.E.; Jordan, M.I. Learning the kernel matrix with semidefinite programming. *J. Mach. Learn. Res.* **2004**, *5*, 27–72.
58. Gönen, M.; Alpaydin, E. Localized multiple kernel learning. In Proceedings of the 25th International Conference on Machine Learning, Helsinki, Finland, 5–9 July 2008; pp. 352–359. [CrossRef]
59. Cortes, C.; Mohri, M.; Rostamizadeh, A. Learning non-linear combinations of kernels. In *Advances in Neural Information Processing Systems 22, Proceedings of the 23rd Annual Conference on Neural Information Processing Systems 2009, Vancouver, BC, Canada, 7–10 December 2009*; Curran Associates Inc.: Nice, France, 2009; pp. 396–404.
60. Bach, F.R.; Lanckriet, G.R.; Jordan, M.I. Multiple kernel learning, conic duality, and the SMO algorithm. In Proceedings of the Twenty-first International Conference on Machine Learning, Banff, AL, Canada, 4–8 July 2004; p. 6. [CrossRef]
61. Hu, M.; Chen, Y.; Kwok, J.T.Y. Building sparse multiple-kernel SVM classifiers. *IEEE Trans. Neural Netw.* **2009**, *20*, 827–839. [CrossRef]
62. Gönen, M.; Alpaydın, E. Multiple kernel learning algorithms. *J. Mach. Learn. Res.* **2011**, *12*, 2211–2268.
63. Horn, R.A.; Johnson, C.R. *Matrix Analysis*; Cambridge University Press: Cambridge, UK, 1985.
64. Schölkopf, B.; Smola, A.J.; Williamson, R.C.; Bartlett, P.L. New support vector algorithms. *Neural Comput.* **2000**, *12*, 1207–1245. [CrossRef] [PubMed]
65. Goldberg, D.E. *Genetic Algorithms in Search, Optimization and Machine Learning*, 1st ed.; Addison-Wesley Longman Publishing Co., Inc.: Boston, MA, USA, 1989.
66. Rojas, S.A.; Fernandez-Reyes, D. Adapting multiple kernel parameters for support vector machines using genetic algorithms. In Proceedings of the 2005 IEEE Congress on Evolutionary Computation, Scotland, UK, 2–5 September 2005; Volume 1, pp. 626–631. [CrossRef]
67. Phienthrakul, T.; Kijsirikul, B. Evolving Hyperparameters of Support Vector Machines Based on Multi-Scale RBF Kernels. In Proceedings of the International Conference on Intelligent Information Processing, Adelaide, Australia, 20–23 September 2006; pp. 269–278. [CrossRef]
68. Beyer, H.G.; Schwefel, H.P. Evolution strategies—A comprehensive introduction. *Nat. Comput.* **2002**, *1*, 3–52. [CrossRef]
69. Youden, W.J. Index for rating diagnostic tests. *Cancer* **1950**, *3*, 32–35. [CrossRef]
70. Powers, D.M.W. Evaluation: From precision, recall and f-measure to roc., informedness, markedness & correlation. *J. Mach. Learn. Technol.* **2011**, *2*, 37–63.
71. Cokelaer, T.; Pultz, D.; Harder, L.M.; Serra-Musach, J.; Saez-Rodriguez, J. BioServices: A common Python package to access biological Web Services programmatically. *Bioinformatics* **2013**, *29*, 3241–3242. [CrossRef]
72. The UniProt Consortium. UniProt: The universal protein knowledgebase. *Nucleic Acids Res.* **2017**, *45*, D158–D169. [CrossRef]
73. Berman, H.M.; Westbrook, J.; Feng, Z.; Gilliland, G.; Bhat, T.N.; Weissig, H.; Shindyalov, I.N.; Bourne, P.E. The Protein Data Bank. *Nucleic Acids Res.* **2000**, *28*, 235–242. [CrossRef]
74. Cock, P.J.A.; Antao, T.; Chang, J.T.; Chapman, B.A.; Cox, C.J.; Dalke, A.; Friedberg, I.; Hamelryck, T.; Kauff, F.; Wilczynski, B.; et al. Biopython: freely available Python tools for computational molecular biology and bioinformatics. *Bioinformatics* **2009**, *25*, 1422–1423.. [CrossRef]
75. Raschka, S. BioPandas: Working with molecular structures in pandas DataFrames. *J. Open Source Softw.* **2017**, *2*. [CrossRef]
76. Hagberg, A.; Swart, P.; Schult, D. Exploring network structure, dynamics, and function using NetworkX. In Proceedings of the 7th Python in Science Conference (SciPy), Pasadena, CA, USA, 19–24 August 2008; pp. 11–15.
77. Virtanen, P.; Gommers, R.; Oliphant, T.E.; Haberland, M.; Reddy, T.; Cournapeau, D.; Burovski, E.; Peterson, P.; Weckesser, W.; Bright, J.; et al. SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nat. Methods* **2020**, *17*, 261–272. [CrossRef] [PubMed]

78. Oliphant, T.E. Python for scientific computing. *Comput. Sci. Eng.* **2007**, *9*. [CrossRef]
79. Chang, C.C.; Lin, C.J. LIBSVM: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.* **2011**, *2*, 27:1–27:27. [CrossRef]
80. Fawcett, T. An Introduction to ROC Analysis. *Pattern Recognit. Lett.* **2006**, *27*, 861–874. [CrossRef]
81. De Santis, E.; Livi, L.; Sadeghian, A.; Rizzi, A. Modeling and recognition of smart grid faults by a combined approach of dissimilarity learning and one-class classification. *Neurocomputing* **2015**, *170*, 368–383. [CrossRef]
82. De Santis, E.; Rizzi, A.; Sadeghian, A. A cluster-based dissimilarity learning approach for localized fault classification in Smart Grids. *Swarm Evol. Comput.* **2018**, *39*, 267–278. . [CrossRef]
83. De Santis, E.; Paschero, M.; Rizzi, A.; Frattale Mascioli, F.M. Evolutionary Optimization of an Affine Model for Vulnerability Characterization in Smart Grids. In Proceedings of the 2018 International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, Brazil, 8–13 July 2018; pp. 1–8. [CrossRef]
84. Khan, S.S.; Madden, M.G. A Survey of Recent Trends in One Class Classification. In *Artificial Intelligence and Cognitive Science*; Coyle, L., Freyne, J., Eds.; Springer: Berlin/Heidelberg, Germany, 2010; Volume 6206, pp. 188–197. [CrossRef]
85. Pimentel, M.A.F.; Clifton, D.A.; Clifton, L.; Tarassenko, L. A review of novelty detection. *Signal Process.* **2014**, *99*, 215–249. [CrossRef]
86. Martino, A.; Rizzi, A.; Frattale Mascioli, F.M. Efficient Approaches for Solving the Large-Scale k-medoids Problem. In Proceedings of the 9th International Joint Conference on Computational Intelligence—Volume 1, Madeira, Portugal, 1–3 November2017; pp. 338–347. [CrossRef]
87. Martino, A.; Rizzi, A.; Frattale Mascioli, F.M. Distance Matrix Pre-Caching and Distributed Computation of Internal Validation Indices in k-medoids Clustering. In Proceedings of the 2018 International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, Brazil, 8–13 July 2018; pp. 1–8. [CrossRef]
88. Martino, A.; Rizzi, A.; Frattale Mascioli, F.M. Efficient Approaches for Solving the Large-Scale k-Medoids Problem: Towards Structured Data. In *Computational Intelligence, Proceedings of the 9th International Joint Conference, IJCCI 2017 Funchal-Madeira, Portugal, 1–3 November 2017*; Sabourin, C., Merelo, J.J., Madani, K., Warwick, K., Eds.; Springer International Publishing: Cham, Switzerland, 2019; pp. 199–219. [CrossRef]
89. Mendel, J.M. Fuzzy logic systems for engineering: A tutorial. *Proc. IEEE* **1995**, *83*, 345–377. [CrossRef]
90. Martino, A.; De Santis, E.; Baldini, L.; Rizzi, A. Calibration Techniques for Binary Classification Problems: A Comparative Analysis. In Proceedings of the 11th International Joint Conference on Computational Intelligence—Volume 1, Vienna, Austria, 17–19 September 2019; pp. 487–495. [CrossRef]
91. Martino, A. Pattern Recognition Techniques for Modelling Complex Systems in Non-Metric Domains. Ph.D. Thesis, University of Rome "La Sapienza", Rome, Italy, 2020.
92. Branden, C.I.; Tooze, J. *Introduction to Protein Structure*; Garland Publishing Inc.: New York, NY, USA, 1991.
93. Giuliani, A.; Benigni, R.; Zbilut, J.P.; Webber, C.L.; Sirabella, P.; Colosimo, A. Nonlinear Signal Analysis Methods in the Elucidation of Protein Sequence-Structure Relationships. *Chem. Rev.* **2002**, *102*, 1471–1492. [CrossRef]
94. Di Paola, L.; Giuliani, A. Protein contact network topology: a natural language for allostery. *Curr. Opin. Struct. Biol.* **2015**, *31*, 43–48. [CrossRef] [PubMed]
95. Devore, J.L.; Peck, R. *Statistics: The Exploration and Analysis of Data*, 4th ed.; Brooks/Cole: Pacific Grove, CA, USA, 2001.
96. Bartz, A.E. *Basic Statistical Concepts*; Macmillan Pub Co.: New York, NY, USA, 1988.
97. Guarnera, E.; Berezovsky, I.N. Allosteric sites: remote control in regulation of protein activity. *Curr. Opin. Struct. Biol.* **2016**, *37*, 1–8. [CrossRef]
98. Negre, C.F.A.; Morzan, U.N.; Hendrickson, H.P.; Pal, R.; Lisi, G.P.; Loria, J.P.; Rivalta, I.; Ho, J.; Batista, V.S. Eigenvector centrality for characterization of protein allosteric pathways. Proc. Natl. Acad. Sci. USA **2018**. *115*, E12201–E12208. [CrossRef]
99. Carlsson, G. Topology and data. *Bull. Am. Math. Soc.* **2009**, *46*, 255–308. [CrossRef]
100. Wasserman, L. Topological Data Analysis. *Annu. Rev. Stat. Its Appl.* **2018**, *5*, 501–532. [CrossRef]
101. Estrada, E.; Rodriguez-Velazquez, J.A. Complex networks as hypergraphs. *arXiv* **2005**, arXiv:physics/0505137.
102. Horak, D.; Maletić, S.; Rajković, M. Persistent homology of complex networks. *J. Stat. Mech. Theory Exp.* **2009**, *2009*, p03034. [CrossRef]

103. Barbarossa, S.; Sardellitti, S. Topological Signal Processing over Simplicial Complexes. *IEEE Trans. Signal Process.* **2020**. [CrossRef]
104. Ghrist, R.W. *Elementary Applied Topology*; Createspace: Seattle, WA, USA, 2014.
105. Hausmann, J.C. On the Vietoris-Rips complexes and a cohomology theory for metric spaces. *Ann. Math. Stud.* **1995**, *138*, 175–188.
106. Zomorodian, A.; Carlsson, G. Computing persistent homology. *Discret. Comput. Geom.* **2005**, *33*, 249–274. [CrossRef]
107. Zomorodian, A. Fast construction of the Vietoris-Rips complex. *Comput. Graph.* **2010**, *34*, 263–271. [CrossRef]
108. Munkres, J.R. *Elements of Algebraic Topology*; Addison-Wesley: Cambridge, MA, USA, 1984.
109. Artin, M. *Algebra*; Prentice Hall: Englewood Cliffs, NJ, USA, 1991.
110. Newman, M.E.J. *Networks: An Introduction*; Oxford University Press: New York, NY, USA, 2010.
111. Katz, L. A new status index derived from sociometric analysis. *Psychometrika* **1953**, *18*, 39–43. [CrossRef]
112. Wasserman, S.; Faust, K. *Social Network Analysis: Methods and Applications*; Cambridge University Press: Cambridge, UK, 1994.
113. Brandes, U. On variants of shortest-path betweenness centrality and their generic computation. *Soc. Netw.* **2008**, *30*, 136–145. [CrossRef]
114. Goh, K.I.; Kahng, B.; Kim, D. Universal behavior of load distribution in scale-free networks. *Phys. Rev. Lett.* **2001**, *87*, 278701. [CrossRef]
115. Estrada, E.; Rodriguez-Velazquez, J.A. Subgraph centrality in complex networks. *Phys. Rev. E* **2005**, *71*, 056103. [CrossRef] [PubMed]
116. Estrada, E. Characterization of 3D molecular structure. *Chem. Phys. Lett.* **2000**, *319*, 713–718. [CrossRef]
117. Boldi, P.; Vigna, S. Axioms for centrality. *Internet Math.* **2014**, *10*, 222–262. [CrossRef]
118. Mones, E.; Vicsek, L.; Vicsek, T. Hierarchy measure for complex networks. *PLoS ONE* **2012**, *7*, e33799. [CrossRef] [PubMed]
119. Saramäki, J.; Kivelä, M.; Onnela, J.P.; Kaski, K.; Kertesz, J. Generalizations of the clustering coefficient to weighted complex networks. *Phys. Rev. E* **2007**, *75*, 027105. [CrossRef] [PubMed]
120. Barrat, A.; Barthélemy, M.; Pastor-Satorras, R.; Vespignani, A. The architecture of complex weighted networks. *Proc. Natl. Acad. Sci. USA* **2004**, *101*, 3747–3752. [CrossRef]
121. Gutman, I.; Zhou, B. Laplacian energy of a graph. *Linear Algebra Appl.* **2006**, *414*, 29–37. [CrossRef]
122. Guimera, R.; Amaral, L.A.N. Functional cartography of complex metabolic networks. *Nature* **2005**, *433*, 895. [CrossRef]
123. Kirkpatrick, S.; Gelatt, C.D.; Vecchi, M.P. Optimization by simulated annealing. *Science* **1983**, *220*, 671–680. [CrossRef] [PubMed]
124. Xiao, B.; Hancock, E.R.; Wilson, R.C. Graph characteristics from the heat kernel trace. *Pattern Recognit.* **2009**, *42*, 2589–2606. [CrossRef]
125. Xiao, B.; Hancock, E.R. Graph clustering using heat content invariants. In Proceedings of the Iberian Conference on Pattern Recognition and Image Analysis, Estoril, Portugal, 7–9 June 2005; pp. 123–130. [CrossRef]
126. Lobanov, M.Y.; Bogatyreva, N.; Galzitskaya, O. Radius of gyration as an indicator of protein structure compactness. *Mol. Biol.* **2008**, *42*, 623–628. [CrossRef]
127. Butler, S. Algebraic aspects of the normalized Laplacian. In *Recent Trends in Combinatorics*; Springer: Berlin/Heidelberg, Germany, 2016; pp. 295–315. [CrossRef]
128. Parzen, E. On estimation of a probability density function and mode. *Ann. Math. Stat.* **1962**, *33*, 1065–1076. [CrossRef]
129. Scott, D.W. On optimal and data-based histograms. *Biometrika* **1979**, *66*, 605–610. [CrossRef]