Chordless cycle filtrations for dimensionality detection in complex networks via topological data analysis

Aina Ferrà Marcús[1], Robert Jankowski[2,3], Meritxell Vila Miñana[4], Carles Casacuberta[1], M. Ángeles Serrano[2,3,5*]

[1]Departament de Matèmatiques i Informàtica, Universitat de Barcelona, Gran Via de les Corts Catalanes 585, 08007 Barcelona, Spain

[2]Departament de Física de la Matèria Condensada, Universitat de Barcelona, Martí i Franquès 1, 08028 Barcelona, Spain

[3]Universitat de Barcelona Institute of Complex Systems (UBICS), Universitat de Barcelona, Barcelona, Spain

[4]Center for Complex Networks and Systems Research, Luddy School of Informatics, Computing, and Engineering, Indiana University, Bloomington, IN, USA

[5]ICREA, Passeig Lluís Companys 23, 08010 Barcelona, Spain

*Corresponding author. E-mail: marian.serrano@ub.edu

September 11, 2025

Many complex networks, ranging from social to biological systems, exhibit structural patterns consistent with an underlying hyperbolic geometry. Revealing the dimensionality of this latent space can disentangle the structural complexity of communities, impact efficient network navigation, and fundamentally shape connectivity and system behavior. We introduce a novel topological data analysis weighting scheme for graphs, based on chordless cycles, aimed at estimating the dimensionality of networks in a data-driven way. We further show that the resulting descriptors can effectively estimate network dimensionality using a neural network architecture trained in a synthetic graph database constructed for this purpose, which does not need retraining to transfer effectively to real-world networks. Thus, by combining cycle-aware filtrations, algebraic topology, and machine learning, our approach provides a robust and effective method for uncovering the hidden geometry of complex networks and guiding accurate modeling and low-dimensional embedding.

1 Introduction

The rapid growth of data has created significant challenges in science and technology. Large datasets from fields like biology (e.g., gene expression and protein interactions), social networks (e.g., agent behavior), and physics (e.g., cosmological simulations) often contain rich structural information hidden in their complex nature. Capturing this information requires tools capable of detecting patterns in data. Topological data analysis (TDA) [1, 2, 3, 4] is one of such approaches, using ideas from topology to identify features that persist across scales.

In TDA, scales are usually defined through filtrations, a systematic way to build a sequence of simplicial complexes that encode the geometric and topological structure of a dataset across multiple levels of resolution. Central to the analysis of such simplicial complexes is persistent homology [2, 3], the main TDA computational method that tracks the emergence, persistence, and disappearance of topological features —such as connected components or loops— in a filtered simplicial complex as the filtration parameter evolves. In this context, graphs can be treated as simplicial complexes and can be filtered by assigning weights to their nodes and/or to their edges.

The success of persistent homology depends critically on the choice of a filtration and topological features of interest, and the selection of these depends on the problem being addressed. Not every filtration or feature descriptor is suitable for every problem, underscoring the importance of designing suitable TDA pipelines to achieve meaningful results. The triad filtration-feature-framework (FFF) embodies an intertwined relationship that forms the foundation for effectively leveraging persistent homology. However, filtration schemes based on a variety of quantifiers, such as Forman-Ricci curvature [5] or betweenness centrality [6], are sometimes applied indiscriminately to data without considering the specificities of the selected topological features and the nature or framework of the problem at hand.

This work emphasizes the importance of aligning the FFF triad by tackling dimensionality detection in complex networks, which also contributes to bridging the gap between complex network theory and TDA. Simplicial complexes associated with complex networks have emerged as powerful representations in network science [7, 8, 9, 10, 11], offering insights into higher-order structures beyond pairwise interactions. Moreover, persistent homology of filtered simplicial complexes has been applied to the study of complex networks in general [12, 13, 14, 15, 16, 17] and to neuroscience in particular [18, 19, 20, 21].

However, the concept of network dimensionality remains largely unexplored within TDA. In network science, it has been addressed within the framework of network geometry [22]. A model-driven approach [23] leverages the geometric $\mathbb{S}^D/\mathbb{H}^{D+1}$ model [24, 25], which reproduces the observed connectivity of real networks, to reveal their intrinsic dimensionality in a latent hyperbolic space, where nodes are more likely to be connected if they are closer to each other. The real network's specific dimension is determined by projecting the frequencies of chordless cycles of varying lengths onto the background model's configuration space. This technique revealed ultra low-dimensional structures in real networks [23] previously masked by apparent high-dimensionality. An alternative methodology to determine network dimensionality

within the same modeling framework uses an embedding technique, named D-Mercator [26], to produce multidimensional maps of real networks in $(D+1)$-hyperbolic space. The maps are used to estimate intrinsic dimensionality in terms of navigability and community structure, producing results consistent with the configuration space approach.

In this work, we propose a third method for dimensionality detection in complex networks based on TDA, that complements a trilogy in combination with the configuration-space and embedding methods used in [23] and [26], respectively. The dimensionality detection task, crucial for understanding the intrinsic geometry of data, showcases how tailored filtrations and feature selection can enhance the ability of persistent homology to analyze and interpret complex data for specific purposes. Specifically, we introduce a chordless cycle filtration scheme and use it to compute extended persistence of cycles, as the topological descriptor that best captures the distribution of cycles in synthetic and real networks to predict their dimensionality.

Moreover, we propose a data-driven approach to estimate network dimensionality by training a neural network on nearly $800\,000$ synthetic networks. A multilayer perceptron accurately estimates dimensionality and transfers effectively to real-world networks, generalizing and adapting to new data without retraining. Ablation experiments demonstrate that TDA features play an important role in this task, even when combined with average cycle densities and degree-related graph features.

2 Results

Our approach for estimating network dimensionality with persistent homology is two-fold. In [23], it has been shown that measuring the intrinsic dimensionality of a complex network is possible by computing profiles of structural properties that are sensitive to dimensionality. These properties are densities of chordless cycles of sizes three (triangle), four (squares), and five (pentagons), but the method required generating ensembles of synthetic networks for each candidate network. Here, we instead focus on persistence summaries using a filtration based on the density of cycles and show that these descriptors reliably detect latent dimensionality. We then leverage the descriptors to develop a supervised machine-learning model, trained on a large database of synthetic graphs with known dimensions and controlled properties, to predict the intrinsic dimension of an input network.

2.1 Persistence descriptors for dimensionality estimation

Many real networks share universal properties, such as sparsity, heavy-tailed degree distributions, the small-world effect, high clustering coefficients, and self-similarity. These properties can be captured by a simple geometric framework [27] using the $\mathbb{S}^1/\mathbb{H}^2$ model [24, 28], which combines a popularity coordinate —controlling node degrees— with a similarity coordinate that represents all other attributes influencing network connectivity. This model can be generalized to a D-dimensional similarity space, yielding the $\mathbb{S}^D/\mathbb{H}^{D+1}$ model [24, 25]; see Section 4 for

more details. These models have been used to determine the dimensionality of real networks by analyzing their cycle profiles [23]. Here, however, we propose to measure their persistence descriptors instead.

The goal of persistent homology is to compute topological features of a space equipped with a filtration. In the context of graphs, one can define filtrations by endowing either nodes or edges with suitable weightings. A common choice is the degree filtration on nodes; however, it has been shown that the degree filtration is less expressive for some graph learning tasks than other, motif-based, filtrations [1]. In this work, we propose an edge weighting scheme based on densities of chordless cycles. A chordless cycle is defined as a closed edge path in which no two non-consecutive nodes are connected by an edge.

Our goal is to determine an optimal value of D for the $\mathbb{S}^D/\mathbb{H}^{D+1}$ model corresponding to a real network, in which γ and β —the parameters of the model controlling the scale-free degree distribution and the clustering coefficient, respectively— are usually unknown. In our data-driven method, optimality is defined in the same way as in [23], by picking the dimension of a closest point in the configuration space of synthetic surrogates generated from the given real network, using D-dimensional *geometric randomization* (D-GR), as described in Section 4.1, with different values of γ, β, and D. The D-GR model, originally proposed for $D = 1$ [29], works on the observed sequence of node degrees and rewires the network to maximize the likelihood that the new topology is generated by the $\mathbb{S}^D/\mathbb{H}^{D+1}$ model.

Thus, for each complex network G_0, whose dimensionality is to be estimated, we generated an ensemble of synthetic surrogates $G_1, \ldots, G_n$ using the $\mathbb{S}^D$ model with a range of values of D and a range of values of the clustering coefficient β until $\beta = 6D$. Densities of edge triangles, chordless squares, and chordless pentagons were then computed as specified in Section 4.2 for each surrogate in the ensemble and for the target network. Therefore, each network G yields three weighted graphs (G, w_t), (G, w_s) and (G, w_p), where the edge weights w_t, w_s, w_p are the densities of triangles, squares, and pentagons, respectively. The mean value of w_t over all the edges of a graph is denoted by C_t, and similarly with squares and pentagons.

Topological features of graphs equipped with a filtration given by edge weights were computed using *persistent homology*, a tool from algebraic topology used to describe shape characteristics from many kinds of data (Section 4.2). In this work, persistence refers specifically to the evolution of cycles along the values of a given filtration. However, cycles in graphs have infinite persistence, since there are no higher-dimensional simplices to eventually fill them. We therefore computed *extended persistence* of cycles, defined as the difference between the largest and smallest weights among the simplices forming a closed path. Since we focus on graphs equipped with edge weightings only, we introduce in this article a new technique for replacing a given edge-weighted graph with a larger, topologically equivalent graph, that carries weights on nodes and on the original edges in a way that is consistent with both sublevel and superlevel filtrations; see Section 4.2 for a detailed description of our method.

Total extended persistence was used as a topological descriptor, resulting in a feature vector (TP_t, TP_s, TP_p) for each network. A representation of the configuration space is shown in Fig. 1, where each point corresponds to a surrogate graph. Points are coloured by the corre-

sponding dimension. The target network from which surrogates were generated has $D = 1$ and is marked with a black cross.

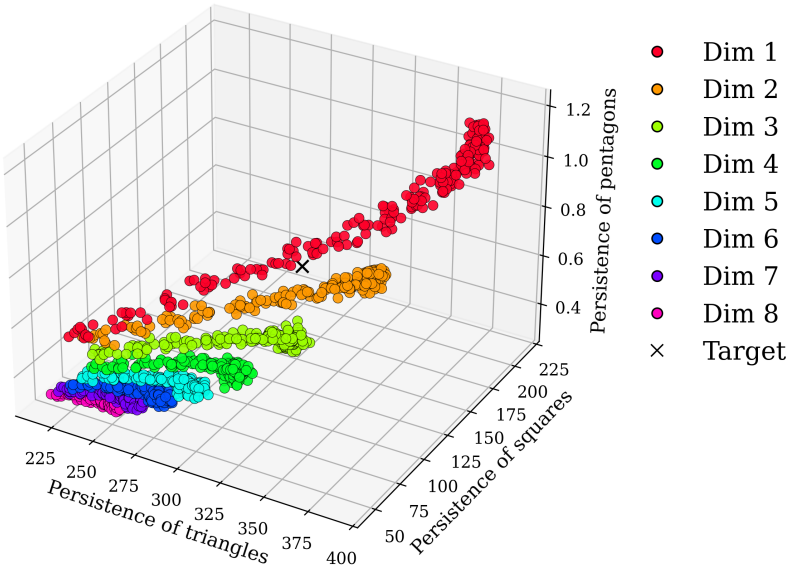

Figure 1: 3D view of a point cloud representing an ensemble of 1330 surrogates of the Human2 network (connectome of the human brain, including one hemisphere) in the configuration space of total persistence computed from three chordless cycle densities (triangles, squares, and pentagons). Points are coloured by dimension. The target network is marked with a black cross.

In order to estimate the dimensionality of the target network, a k-nearest neighbors classifier (kNN) was used. The classifier identifies the k surrogates closest to the target network in the surrogate configuration space (TP_t, TP_s, TP_p) by minimizing Euclidean distance. The value of k was not fixed, but it was determined within each ensemble of surrogates, by finding the value of k with highest accuracy in classifying each surrogate in the ensemble. Specifically, the inferred dimension D^* of the target network G_0 maximizes the weighted frequency $f(D) = \sum_{i=1}^{k} \omega_i \, \delta_{D_i,D}$, where the normalized weights are inversely proportional to the distance between the real network and the i-th surrogate G_i in the (TP_t, TP_s, TP_p) space, and $\delta_{D_i,D}$ is the Kronecker delta function [23]. A schematic pipeline summary of the suggested methodology is shown in Fig. 2a.

The closeness of the kNN approximation was measured with a *congruency index* reflecting the fidelity of the surrogates with respect to the target network. The index is defined as the ratio $\rho = d_0/\overline{d}$, where d_0 is the distance between the point in configuration space corresponding to the target network and its closest neighbor, and $\overline{d}$ is the average of the distances between each surrogate point and its closest neighbor. Hence, the value of ρ is large when the point cloud is clustered while the target network falls far away from the clusters. For comparability reasons,

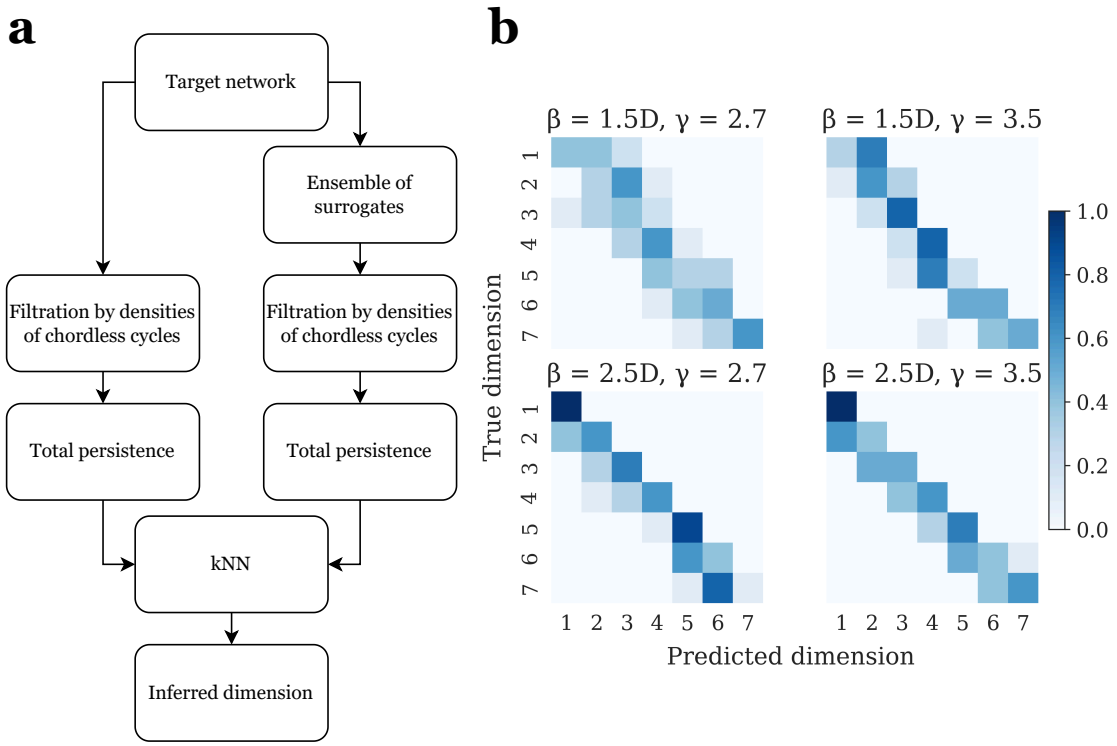

Figure 2: (a) Pipeline of our first method and (b) confusion matrices. Given a target network, we generate an ensemble of surrogates and, for each of them (including the given network), we compute densities of chordless cycles. We equip the graphs with these weightings and perform a topological analysis using extended persistence. For each network, we compute total persistence of cycles and use a kNN classifier in the configuration space to infer a dimensionality D^* for the real network. The upper two confusion matrices show results obtained with synthetic networks for $\beta = 1.5\,D$ and two different values $\gamma = 2.7$ and $\gamma = 3.5$. This choice of β corresponds to the small-world phase even if$\gamma > 3$. The lower two confusion matrices show results for $\beta = 2.5\,D$; in this case, networks with $\gamma > 3$ are large worlds. In each matrix, the rows correspond to the true dimension D and the columns correspond to the inferred dimension D^*. Thus, the j-th box in the i-th row shows the fraction of i-dimensional networks that were classified as j-dimensional. The darker the color, the greater the number. Each matrix is evaluated with 70 synthetic networks.

it is convenient to provide values of $1/\rho$, as in Table 1, since, in most cases, $1/\rho$ takes values between 0 and 1, with values closer to one indicating higher fidelity between the target network and its surrogates.

As an evaluation step, the performance of the chordless cycle density filtration was tested on synthetic target networks generated using the $\mathbb{S}^D$ model for specific values of D (from 1

to 7), with different degree heterogeneities by varying the exponent ($\gamma = 2.7$ and $\gamma = 3.5$), and two values of inverse temperature ($\beta = 2.5\,D$, corresponding to the high clustering regime, and $\beta = 1.5\,D$, corresponding to the low clustering regime). For each combination (D, γ, β), ten synthetic target networks were generated, and the above method was applied to each of them to infer a dimensionality D^*. Inferred dimensionalities were compared with the original dimension D from which the target synthetic network was generated. Confusion matrices to visualize the performance of the inference method are shown in Fig. 2b. Our predictions generate some small confusion with contiguous values of D to the diagonal, especially for low values of β and γ, but this is expected, due to the high heterogeneity of the degree distribution.

Estimated dimension values for selected real-world target networks using TDA are shown in Table 1. The resulting values were compared with those obtained by implementing the method described in [23], using the configuration space of mean cycle densities (C_t, C_s, C_p), which have been recalculated using the D-GR procedure. Inverse values of the congruency index ρ are shown. As additional information, 2D projections of the configuration space (TP_t, TP_s, TP_p) for the selected real-world networks are displayed in Figs. S1 and S2.

Table 1: Inferred dimensionality of real-world networks. Comparison of the inferred dimension between three different methods: (1) Mean densities of chordless cycles; (2) TDA: Persistent homology from chordless cycle density filtrations; (3) DIMNN: A neural network trained on a database of synthetic networks. The $1/\rho$ values are inverses of the congruency indices; higher values correspond to a closer match between the real network and its surrogate models.

Network	**Domain**	**Densities**		**TDA**		**DIMNN**
		dim	$1/\rho$	**dim**	$1/\rho$	**dim**
Human2-C	Biological (Connectome)	3	0.07	1	0.21	2
Human-M	Biological (Metabolic)	3	0.19	4	0.46	3
Cargoships	Economic (Trade)	3	0.10	2	0.09	5
Bible-CO	Informational (Language)	4	0.05	1	0.04	5
Jazz-CA	Social (Collaboration)	2	0.06	2	0.41	2
EUEmail	Social (Communication)	2	0.09	1	0.28	2
Friends-ON	Social (Communication)	6	0.06	6	0.19	7
Friends-OFF	Social (Offline)	8	0.72	8	0.28	10

2.2 Dimensionality estimation using neural networks

The method described in the previous section and the approach from [23] rely on a large set of surrogate networks. For each new real network, one needs to generate a set of synthetic networks, compute their properties, and use a classifier to detect the dimension. As a consequence, when a new network is considered, the entire pipeline must be repeated. In this section, we

propose an alternative based on a neural network that, once trained, can estimate dimensions directly, even for very large networks, generalizing and adapting to new data without retraining.

Neural networks excel for such tasks, since our aim is to approximate an unknown function yielding dimensionality values from a collection of predictors, including mean chordless cycle densities and/or persistence of corresponding filtrations. For this purpose, we created a database of synthetic complex networks called SYNNET, which we use to train a neural network, named DIMNN, to estimate the dimensionality of real-world networks. The main advantage of this method is that training in our database is performed only once and independently of specific target networks. This avoids the need to generate surrogates for each case under study. In total, we produced 792 000 synthetic networks generated from the $\mathbb{S}^D$ model (see Section 4.3 for more details). An 80%-20% training-validation split was used.

For each synthetic network in the database, chordless cycle densities were computed as in Section 4.2 and averaged over all edges, as well as total persistence values obtained from the corresponding filtrations, and, additionally, the first moment and the normalized second moment of the degree distribution (i.e., the expected square divided by the square of the expected value). The normalized second moment is sensitive to degree fluctuations and so to very high degree nodes: for highly heterogeneous networks, the ratio becomes large, while for homogeneous networks the ratio is close to 1. Hence, the normalized second moment tends to correlate with γ. Likewise, the average density of triangles C_t serves as an approximation of the inverse temperature β.

Other relevant descriptors of complex networks that we integrated in our feature vectors are minimum and maximum degree, and average neighbor degree, which is denoted by $\langle k_{\mathrm{nn}} \rangle$ (see Section 4.2). This is a measure used to quantify degree-degree correlations in a graph, that is, how the degree of a node relates to the degrees of its neighbors.

Feature vectors consisting of number of nodes, average degree, normalized second moment of the degree distribution, minimum and maximum degrees, average neighbor degree, mean chordless cycle densities C_t, C_s, C_p, and total persistences TP_t, TP_s, TP_p of the corresponding filtrations were fed into a residual multilayer perceptron (see Section 4.3 for details). Our pipeline is shown in Fig. 3a. The highest classification accuracy obtained with DIMNN for estimating the dimensionality in the range $D = 1$ to $D = 10$ was 83.00% on the validation set with the full feature vector. Confusion matrices are shown in Fig. 3b.

We also performed an ablation study, selecting subsets of the feature vector for both training and predictions. In Fig. 4, we show how the performance changes as a set of features is added at a time. The lowest accuracy is obtained when we use only the vector of mean chordless cycle densities with no other added features. Once degree-related properties are appended to the feature vector —e.g., minimum, maximum, and average degrees—, the accuracy increases, and incorporating topological information improves the accuracy further. In fact, combining total persistence with mean chordless cycle densities maximizes the performance. In Table S1, we show mean validation accuracies as well as the number of epochs and total training times, and a corresponding heatmap is provided in Fig. S4.

Next, we show the results of applying the trained DIMNN to detect the dimensionality of

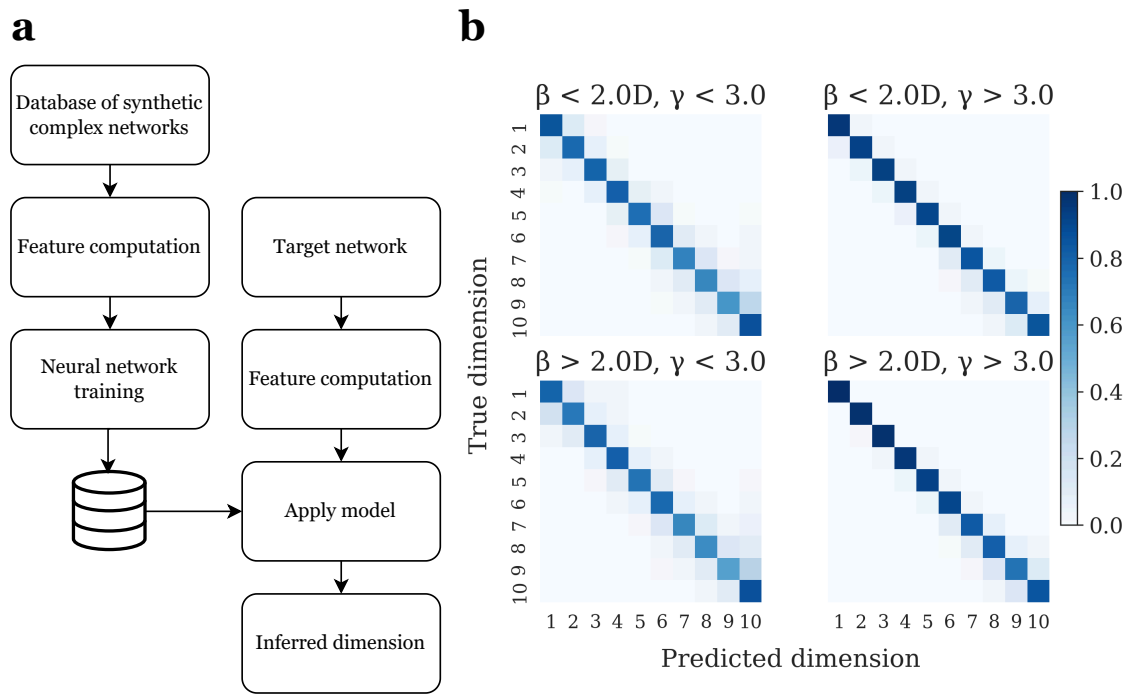

Figure 3: (a) Pipeline of dimensionality estimation using DIMNN and (b) confusion matrices. A multilayer perceptron is trained on a database of synthetic complex networks, with features coming from filtrations by densities of chordless cycles concatenated with averages of chordless cycle densities and degree-related graph features. The parameters of the trained model are stored and the model is applied to target networks. Confusion matrices show results for the four combinations of $\beta < 2D$ versus $\beta > 2D$, and $\gamma < 3$ versus $\gamma > 3$, from a range of values $1.2\,D$ to $5.0\,D$ for β and 2.2 to 5.0 for γ. In each matrix, the rows correspond to the true dimension D and the columns correspond to the inferred dimension D^*.

real-world complex networks. We analyzed the 10 networks from the previous section and predicted their dimensions. Table 1 compares the inferred dimension across three different methods. One can observe that the predictions do not vary significantly, by only one or two dimensions in most cases, which highlights the robustness of the trained neural network. We incorporated another dataset of real complex networks [30]. First, we filtered out bipartite and temporal networks, and restricted to networks with topological properties aligned with the training dataset (Fig. S6). In total, we gathered 53 new networks with properties summarized in Tables S3 and S4.

In Fig. 5a, we group the dataset by domain, showing that the Biological and Transportation categories together account for more than half of all networks. We then apply our trained neural network to predict the intrinsic dimensions of these real-world networks (see Fig. 5b). Biological networks span a wide range of dimensions (from 1 through 6), whereas all networks

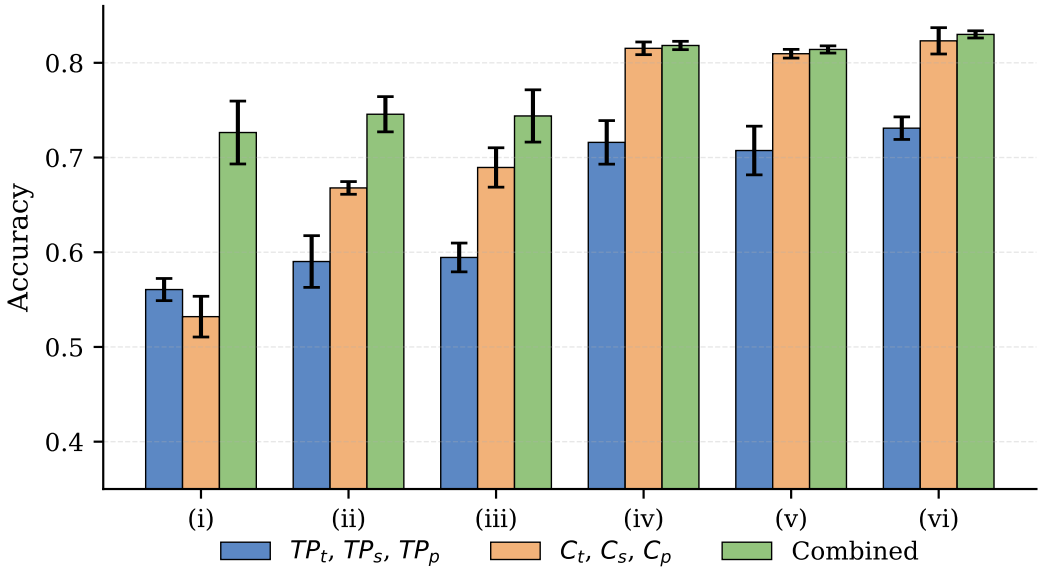

Figure 4: Validation accuracies after five repetitions of DIMNN using the SYNNET database. *Blue:* Accuracies obtained using total persistences TP_t, TP_s, TP_p, plus cumulative features. *Orange:* Accuracies using average cycle densities C_t, C_s, C_p, plus cumulative features *Green:* Accuracies obtained combining total persistences and mean cycle densities plus cumulative features. Successive columns correspond to incorporating one after the other the following additional features into the model: (i) no added features; (ii) number of nodes N; (iii) number of nodes and average degree $\langle k \rangle$; (iv) number of nodes, average degree, and normalized second moment $\langle k^2 \rangle / \langle k \rangle^2$; (v) number of nodes, average degree, normalized second moment, minimum degree $k_{\min}$ and maximum degree $k_{\max}$; (vi) number of nodes, average degree, normalized second moment, minimum degree, maximum degree, and mean average neighbor degree $\langle k_{\rm nn} \rangle$.

in the Software domain lie at dimension 1. A closer look at subdomains (Fig. S7) reveals that web graphs —an Informational subdomain— also have dimension 1, while language networks (also Informational) appear at dimensions 5 and 9.

Subsequently, to enhance confidence in the predictions made by DIMNN, a complementary neural network model was trained. The base architecture was the same as for DIMNN, but the loss function was changed to mean squared error (MSE), with the goal of transforming a classification task into a regression task. Thus, contrary to DIMNN, the output range of the regressor was not restricted to the interval $[1, 10]$. Since regression yields continuous values, each prediction was rounded to the nearest integer. We use the term *pseudo-accuracy* to denote the accuracy obtained by the regressor after rounding its predictions to integers. Moreover, residual connections were removed.

The regressor model exhibits behavior very consistent with that of DIMNN on synthetic

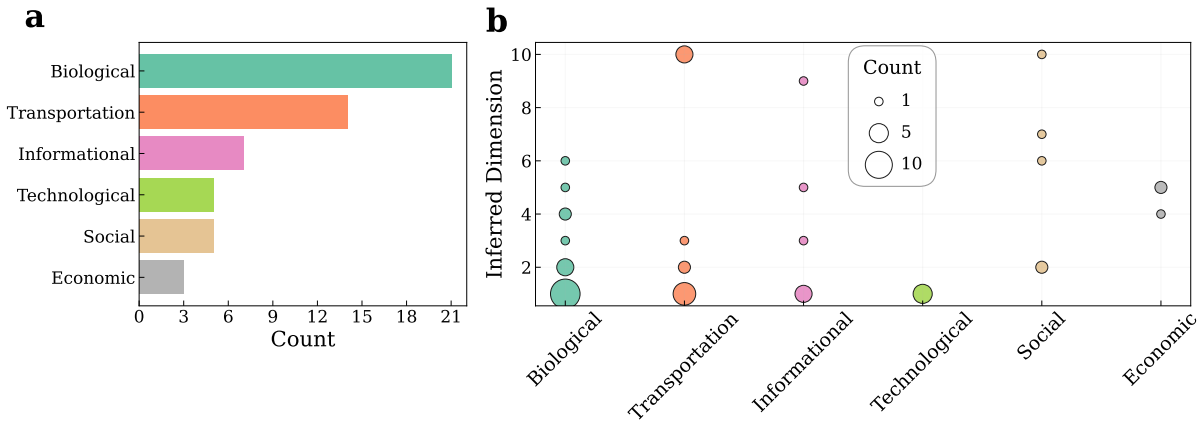

Figure 5: (a) Number of networks for each network domain. (b) Inferred dimension of real networks using DIMNN, grouped by network domain. The size of the marker indicates the number of networks.

networks (Fig. S5). The percentage of agreement between the two models on the validation set is 79.33% if total persistences are combined with mean chordless cycle densities, and reaches 88.86% when degree-related graph features are added to the training process. The *discrepancy* between the two models is defined as the absolute value of the difference between their predicted dimensions, computed only from those networks in the validation set on which the two models do not agree. The median discrepancy value was found to be 1, with a mean of 1.05 and a standard deviation of 0.2543 when a full vector of descriptors was fed to the model. Additional details are given in Table S2.

When applied to the real-world network dataset described in Tables S3 and S4, the percentage of agreement between the two models was 64.15% over 53 networks with full feature vectors in the models, with a median discrepancy of 1, a mean of 2.95, and a standard deviation of 3.2032, indicating the occurrence of a few larger discrepancies.

3 Discussion

The main contribution of this work is to use densities of cycles (triangles, chordless squares, and chordless pentagons) in a complex network as weights on its edges, and treat each of these three weightings as a filtering function on the underlying graph. Persistence descriptors are computed from the corresponding filtrations and used for estimating the dimensionality of the given network. The expressiveness of edge-based filtrations defined via densities of chordless cycles is further examined in [31], where they are compared with other motif-based filtrations in graph isomorphism detection tasks.

Although topological invariants of graphs are crucial in the study of complex networks, methods from topological data analysis have not been broadly used in this context [1, 32]. To

our knowledge, this is the first work that utilizes *extended* persistent homology of graphs for dimensionality estimation in complex networks. Ordinary (as opposite to extended) persistent homology is not well-suited for this purpose, since cycles become permanent in graphs. Using Vietoris-Rips simplicial complexes from graph distances is not suitable in our work either, due to the small-world property, and clique complexes of graphs are not optimal either, since triangles become invisible in clique complexes. The extended persistence or lifetime of a cycle in a weighted graph is the difference between the largest weight and the smallest weight of the edges and vertices that form the given cycle. The total lifetime for a set of linearly independent cycles carries relevant information about the geometric structure of the graph.

Other persistence descriptors could be used for analytical purposes. We found persistence of connected components along the filtrations to be less expressive than its cycle-based counterpart. We also remark that the expressivity of cycle persistence is influenced by certain network parameters, such as inverse temperature (β). Indeed, in the point cloud shown in Fig. 1, low values of β tend to separate network dimensions less prominently.

Our results show that total extended persistence computed by means of chordless cycle filtrations yields dimensionality estimates which are comparable with those obtained in previous work [23]; see Table 1. Moreover, total persistence improves the accuracy of a neural network classifier if added to feature vectors containing averages of chordless cycle densities and other graph features such as average degree, as shown in Fig. 4 and in Table S1. Our conclusion is that TDA enhances the performance of dimensionality estimation by means of a neural network, even in the presence of average cycle densities and degree-related graph descriptors.

In fact, estimating network dimensionality without the need to generate surrogates is another major advance made in this article. For this purpose, a universal database of synthetic graphs has been generated from a uniformly distributed range of parameter values. Real-world network dimensionality estimates can then be obtained by running a neural network model trained on our database. This approach considerably reduces the computation time of estimations as well as the amount of memory required for this task. It is far more ambitious than creating an ensemble of surrogates for each given real network under study, as done in the first part of our work.

Fig. 4 highlights the importance of the normalized second moment $\langle k^2\rangle/\langle k\rangle^2$ in the accuracy of a neural network classifier for latent dimension. This could be due to the fact that the normalized second moment correlates with the exponent γ, and knowledge of β and γ is crucial for dimensionality estimation. An approximation of β is provided by the average density of triangles C_t, which is also correlated with the total persistence value TP_t.

The agreement between two independently trained neural network models, each optimized with a different loss function, is a strong indicator of robustness and generalization. When models trained with distinct objective functions (in our case, cross-entropy for classification and mean squared error for regression) converge to similar predictions on previously unseen data, it is unlikely that their performance is due to overfitting specific patterns or uninformative noise in the training set. Rather, such an agreement suggests that both models are approximating the same underlying relationship between the input features (topological and density-based descriptors) and the latent embedding dimension. It is remarkable that a neural network trained

on synthetic networks yields results closely aligned with earlier work on the latent dimension of real-world networks. Moreover, the discrepancies found between the dimensionality predictions of the two models may serve to detect potential out-of-distribution real networks with respect to the D-GR model in the study dataset, which therefore deserve further examination.

Subsequent research could focus on optimizing the accuracy of a neural network for dimensionality classification of synthetic networks and enhancing the performance of latent dimension estimates for real-world networks. On the one hand, this could be achieved by enriching the training database with a wider range of complex network shapes. On the other hand, the neural network architecture could be optimized for the intended tasks. This article only provides a proof of concept for the benefits of a cycle-based filtration in TDA and the feasibility of our novel, deep-learning assisted, dimensionality estimation method. While our architecture (Fig. S3) has demonstrated a capability to perform in a way consistent with previous work [23] and has achieved an accuracy above 83% on synthetic networks, improving its overall effectiveness remains a challenge to be addressed in future work.

4 Methods

4.1 Multidimensional geometric soft configuration model

The $\mathbb{S}^{D}/\mathbb{H}^{D+1}$ model. In the $\mathbb{S}^{D}$ model, a node i is assigned two hidden variables: a hidden degree κ_i, quantifying its importance or popularity, and a position $\mathbf{v}_i$ in a D-dimensional similarity space, represented as a D-sphere. The probability of connection between any pair of nodes i and j follows a gravity law, in which similar nodes are angularly closer and, thus, probably connected [27]. Specifically, nodes i and j are connected with probability

$$p_{ij} = \frac{1}{1 + \left(\frac{R\Delta\theta_{ij}}{(\mu\kappa_i\kappa_j)^{1/D}} \right)^{\beta}} \tag{1}$$

where D is the dimension of the model, β controls the level of clustering of the network and the coupling of the network with the underlying metric space, μ controls the average degree, and $\Delta\theta_{ij}$ is the angular distance between nodes i and j, which are assigned positions $\mathbf{v}_i$ and $\mathbf{v}_j$ on the D-sphere. The radius R of the sphere is set such that the density of N nodes is 1 (without loss of generality). This yields $R = \left[\Gamma\left(\frac{D+1}{2}\right) N/(2\pi)^{\frac{D+1}{2}}\right]^{1/D}$, where Γ is the gamma function [27]. For $\beta < D$, networks are unclustered in the infinite-size limit, whereas for $\beta > D$ networks exhibit finite clustering in the thermodynamic limit. Finally, the parameter μ controls the average degree of the network and is defined as

$$\mu = \frac{\beta\,\Gamma\left(\frac{D}{2}\right)\sin\left(\frac{D\pi}{\beta}\right)}{2\pi^{1+\frac{D}{2}}\langle k\rangle}. \tag{2}$$

The $\mathbb{S}^D$ model is isomorphic to the purely geometric $\mathbb{H}^{D+1}$ model [25] in $(D+1)$-hyperbolic space by mapping the hidden degree into the radial coordinates as

$$r_i = \hat{R} - \frac{2}{D}\ln\frac{\kappa_i}{\kappa_0}, \quad \text{with} \quad \hat{R} = 2\ln\left(\frac{2R}{(\mu\kappa_0^2)^{1/D}}\right). \tag{3}$$

A network generation procedure following the $\mathbb{S}^D/\mathbb{H}^{D+1}$ model is described in Algorithm 1 in Supplementary Information, where a cutoff κ_c is calculated to prevent excessive fluctuations in the largest expected degrees when $\gamma < 3$; see [33].

Microcanonical formulation of $\mathbb{S}^D$ model. A microcanonical version of the $\mathbb{S}^1$ model was first proposed in [29], where it was called *geometric randomization model* (GR). The GR works on the sequence of observed node degrees and assigns node positions randomly in the similarity space. The network is rewired to maximize the likelihood that the new topology is generated by the $\mathbb{S}^1$ model while preserving the observed degrees and, thus, the total number of edges.

In this work, we extended GR to higher dimensional similarity spaces, which we call D-GR. We used D-GR to generate synthetic networks for dimensionality estimation when investigating the relationship between total persistence in homological dimension 1 computed from three types of chordless cycles filtrations and the inferred dimension (see Section 2.1). We want to highlight the main difference with respect to the previous approach [23]. In [23], the authors infer the set of degrees κ from a given network and generate synthetic networks with Eq. (1) given the parameters β and D. Although in [23] the degree distribution of the generated surrogates is very similar to that of the input network, the D-GR procedure is more constrained and maintains the exact degree values of the input network.

In the D-GR model, we assign to each node i a random position in the $D+1$ dimensional Euclidean space $\mathbf{v}_i \in \mathbb{R}^{D+1}$ with $||\mathbf{v}_i|| = R$. The nodes are uniformly distributed on the D-sphere using Marsaglia's algorithm [34].

The rewiring procedure is carried out with the Metropolis-Hastings algorithm, aimed at finding the adjacency matrix that maximizes the likelihood function

$$\mathcal{L} = \prod_{i<j} p_{ij}^{a_{ij}}[1-p_{ij}]^{1-a_{ij}} \tag{4}$$

where p_{ij} comes from Eq. (1) and a_{ij} are elements of the adjacency matrix. The method proceeds as described in Algorithm 2 in Supplementary Information.

As shown in [29], the probability of swapping links between nodes i and j and between nodes l and m is given by

$$\frac{\mathcal{L}_n}{\mathcal{L}_c} = \left(\frac{\Delta\theta_{ij}\,\Delta\theta_{lm}}{\Delta\theta_{il}\,\Delta\theta_{jm}}\right)^{\beta}. \tag{5}$$

Notice that Eq. (5) does not depend on the dimension D.

To validate our approach, we generated synthetic networks with known dimension using the

$\mathbb{S}^D$ model (see Algorithm 1 in the Supplementary Information), with the following parameters: the number of nodes (N) was set to 500 and the average degree $\langle k \rangle$ to 10; dimension (D) ranging from 1 to 7; the inverse temperature parameter (β) taking values $1.5\ D$ and $2.5\ D$; and the power-law exponent (γ) taking values 2.7 and 3.5. For each set of parameters, we generated 10 network realizations, thus obtaining 280 synthetic networks. For each of these, we produced a set of surrogates using the D-GR method (Algorithm 2 in the Supplementary Information), scanning over different values: dimension (D) ranging from 1 to 7 and rescaled inverse temperature β/D ranging from 1.2 to 3.0 with steps of 0.1. The geometric randomization was repeated 10 times, yielding $1\,330$ surrogates per synthetic network. At the end of this procedure, we obtained a total of $372\,400$ networks, that were used in the confusion matrices in Fig. 2.

To infer the dimension of real networks, we proceeded in a similar fashion. We applied the D-GR method with the same set of parameters described above, yielding a total of $1\,330$ surrogates for each real network. Inferred dimensions of real networks are described in Table 1.

It is worth noting some limitations of this method. The acceptance probability depends on the parameter β. For very large values of β, the acceptance ratio becomes binary, i.e., moves that increase the likelihood are almost always accepted, and those that decrease it are almost always rejected, and the likelihood plateau cannot be reached. Thus, the algorithm is restricted to moderate values of β and corresponding values of the clustering coefficient. Techniques such as simulated annealing or parallel tempering can restore good mixing in the large β regime. In this work, however, we restrict our attention to real networks whose topology is well-captured by a moderate value of β.

Moreover, even though the $\mathbb{S}^D/\mathbb{H}^{D+1}$ model captures a wide range of topological network properties, some real networks may lie outside the range of values and are located further away from the surrogate networks in the persistence (TP_t, TP_s, TP_p) and mean cycle density (C_t, C_s, C_p) configuration spaces.

4.2 Graph features

Densities of chordless cycles. Let $G = (V, E)$ be a graph and $e_{ij} = \{v_i, v_j\}$ be an edge between nodes v_i and v_j in E, with degrees $k_i > 1$ and $k_j > 1$, respectively. The *density of triangles* corresponding to the edge e_{ij}, also called *edge clustering coefficient* [35], is the number $\#\triangle_{ij}$ of edge triangles in G containing e_{ij} divided by the maximum possible number of triangles in G containing e_{ij} given the degrees k_i and k_j, that is,

$$C_t(e_{ij}) = \frac{\#\triangle_{ij}}{\min(k_i, k_j) - 1}. \tag{6}$$

An edge cycle is *chordless* if there is no edge between its nodes except those that form the cycle. *Density of squares*, denoted $C_s(e_{ij})$, is defined by dividing the number $\#\square_{ij}$ of chordless edge squares in G containing e_{ij} by the maximum possible number of such squares given the degrees k_i and k_j and the existing triangles through e_{ij} in G. To define the *density of pentagons* $C_p(e_{ij})$

per edge e_{ij}, we count the number $\#\pentagon_{ij}$ of chordless pentagons containing e_{ij} and normalize it by the maximum possible number of such pentagons, assuming known the degrees of v_i and v_j and the degrees of their respective neighbors. One could also define similar densities per node. However, in [35], the authors found the definitions relative to edges to be more stable with respect to degree heterogeneity than those relative to nodes.

Regarding the computational cost of computing chordless cycles, the time complexities for computing densities of triangles, squares, and pentagons are $O(N\langle k\rangle^2)$, $O(N\langle k\rangle^3)$, and $O(N\langle k\rangle^4)$, respectively, where $\langle k\rangle$ is the average degree and $N = \#E$ is the number of edges. In sparse graphs, $\langle k\rangle \ll N$. Even though the computational cost for large graphs might be high, the computations can be easily parallelized or implemented on GPU with CUDA [36].

Extended persistent homology. In this work, we use persistent homology to extract information from a graph G equipped with a filtration $\{G_t\}$, where t is a real-valued parameter. *Persistence* refers to the evolution of cycles along the values of the given filtration.

Homology of graphs is a special case of simplicial homology of simplicial complexes [37]. In the case of graphs, homology is determined by two numbers, called *Betti numbers*, namely the number β_0 of connected components and the cardinality β_1 of a maximal set of linearly independent cycles. *Cycles* are finite formal sums $z = \sum \lambda_i e_i$ of oriented edges with coefficients $\lambda_i = \pm 1$ such that $\sum \lambda_i \partial e_i = 0$, where, for an edge $e \in E$ from v_0 to v_1, we denote $\partial e = v_1 - v_0$. Thus, cycles are algebraic representations of closed edge paths in the given graph.

In a filtered simplicial complex $\{K_t\}$, the *birth* of a cycle z of any dimension is the t-value at which z appears in K_t, and the *death* of z is the t-value at which z becomes the boundary of a higher chain. Hence, in the case of a filtered graph, the birth of a cycle z is the t-value at which z is formed, and the death value is infinite, since a graph does not contain higher simplices.

To avoid infinite persistence values, we use *extended persistence*, as in [38, 39]. For a graph $G = (V, E)$ equipped with a node weighting $w_V : V \to \mathbb{R}$, the *sublevel filtration* $\{G_t\}$ is defined as $G_t = (V_t, E_t)$, where $V_t = \{v \in V \mid w_V(v) \leq t\}$ and E_t is the subset of E spanned by V_t, and the *superlevel filtration* $\{G^t\}$ is defined as $G^t = (V^t, E^t)$, where $V^t = \{v \in V \mid w_V(v) \geq t\}$ and E^t is the subset of E spanned by V^t. Then the *extended persistence* of a cycle z in G is defined as $|d - b|$, where b is the birth value of z in the sublevel filtration $\{G_t\}$ and d is the birth value of z in the superlevel filtration $\{G^t\}$. Note, however, that no edge weighting can be defined on G compatibly with w_V which is consistent with both $\{G_t\}$ and $\{G^t\}$, in general.

In our work, chordless cycle filtrations are defined by means of *edge* weightings, not node weightings. Although it is perfectly possible to exchange the roles of nodes and edges in the definitions of sublevel and superlevel filtrations, extended persistence calculations have been carried out using the Python library Gudhi [40], which only provides software for computing extended persistence of graphs with a weighting on their *nodes* (see Section 2.1 of [39]). Therefore, we introduce a method to replace a graph $G = (V, E)$ equipped with an edge weighting $w_E : E \to \mathbb{R}$ by a topologically equivalent graph $G' = (V', E')$ in which $E \subseteq E'$ and endowed with a node weighting $w_{V'} : V' \to \mathbb{R}$ yielding the same extended persistence diagram as G.

To achieve this, we split each node $v_i \in V$ of degree d_i into d_i distinct nodes $v_{i,j} \in V'$,

where j ranges over the subindices of the neighbors of v_i. Each node $v_{i,j}$ is assigned weight $w_{V'}(v_{i,j}) = w_E(e_{ij})$ where $e_{ij} = \{v_i, v_j\}$. The set E' contains *horizontal* edges $\{v_{i,j}, v_{j,i}\}$ for all i, j, joining nodes with the same weight, and *vertical* edges, connecting each string of nodes $v_{i,j}$ with a fixed i value, sequentially from smallest weight to largest, as in Fig. 6. Hence, the set of horizontal edges in E' is in bijective correspondence with E, and the weighting $w_{V'}$ is consistent with w_E on all edges of E, in both the sublevel and superlevel filtrations of G'.

We call G' a *degree-splitting subdivision* of G. The graph G' is then fed into Gudhi [40] as a node-weighted graph, and total persistence $\sum |d_i - b_i|$ of cycles in the corresponding extended persistence diagram is recorded (*relative* cycles provided by the Gudhi software, if any, are discarded). The graph G' has the same Betti numbers as G at the same thresholds, since G is obtained from G' by contracting vertical edges, which does not change homology.

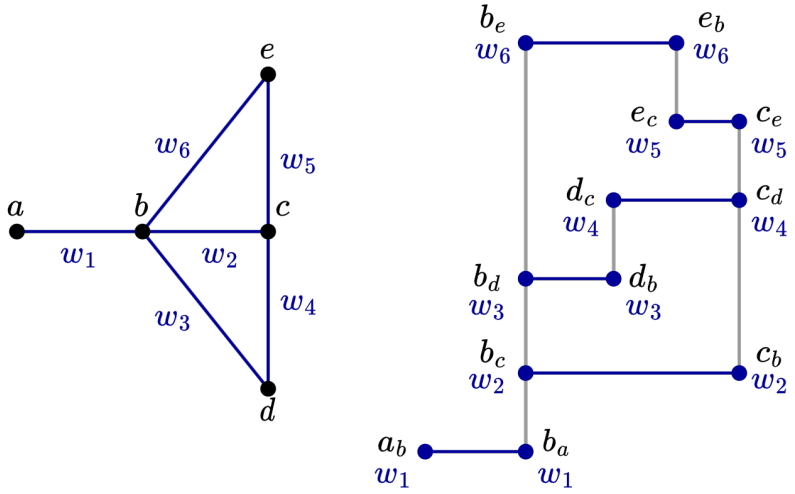

Figure 6: Degree-splitting subdivision of a graph G (left), converting an edge weighting w_E into a node weighting on G' (right) that induces w_E on the horizontal edges. In this example, two generating cycles with birth-death coordinates $(4, 2)$ and $(6, 3)$ yield a total persistence of 5.

Degree-related graph features. The following descriptors from complex network theory are calculated to train the models used in Section 2.2. For each graph $G = (V, E)$, we consider the *average degree* of its nodes, $\langle k \rangle = (1/\#V) \sum_{v \in V} \deg(v)$. The *minimal degree* $k_{\min}$ and *maximal degree* $k_{\max}$ are also considered. The *average neighbor degree* of G is the average of the mean neighbor degree of its nodes:

$$\langle k_{\mathrm{nn}} \rangle = \frac{1}{\#V} \sum_{v \in V} \left(\frac{1}{\deg(v)} \sum_{u \sim v} \deg(u) \right).$$

The *normalized second moment* of G is defined as the quotient $\langle k^2 \rangle / \langle k \rangle^2$, where we denote

$\langle k^2 \rangle = (1/\#V) \sum_{v \in V} \deg(v)^2$. Its significance is due to the fact that it correlates inversely with the power-law exponent γ in scale-free networks [41].

4.3 Database and neural network architecture

For a better understanding and comparison of the results, we used the same real-world networks as those reported in [23]. These are undirected networks with fewer than 100 000 nodes from very different domains. Tables S3 and S4 include a detailed description of the data as reported in [23].

SYNNET dataset for inferring dimension using neural networks. Deep networks require large amounts of data to avoid overfitting and to learn robust features, due to their large number of parameters. In fact, neural networks generally require large datasets to achieve high performance [42, 43]. However, with the $\mathbb{S}^D$ model, we tackle this challenge by generating plenty of synthetic graphs with known dimensions to train neural networks.

We prepared a dataset of 792 000 synthetic networks generated using the $\mathbb{S}^D$ model by means of the method described in Supplementary Algorithm 1 with the following input parameters: dimension (D) from 1 to 10; number of nodes (N) with values 200, 400, 750, 1000, 2500; power-law exponent (γ) with values 2.2, 2.4, 3.0, 4.0, 5.0; average degree ($\langle k \rangle$) with values 4, 8, 12, 25; and rescaled inverse temperature (β/D) with values 1.2, 1.4, 1.6, 1.8, 2.0, 2.2, 2.5, 2.8, 3.0, 3.5, 4.0, 5.0. For each set of parameters, 66 network realizations were made. The dataset covers a wide range of network properties —in particular, $\langle k_{nn} \rangle \in [4.4, 300.8], C_t \in [0.073, 0.891], C_s \in [0.00505, 0.24193], C_p = [0.00010, 0.02165]$.

The database contains a set of feature vectors for each network, aiming for its use in a multilayer perceptron (MLP), which is described in Section 4.3. The following descriptors were computed, resulting in a 12-dimensional feature vector: N, $\langle k \rangle$, $\langle k^2 \rangle / \langle k \rangle^2$, $k_{\min}$, $k_{\max}$, $\langle k_{\rm nn} \rangle$, C_t, C_s, C_p, TP_t, TP_s, TP_p, as described in Section 4.2. Once a MLP is trained, the same features are computed for real networks in order to predict their dimensions.

It is worth mentioning that some real networks could have very distinct topological properties that are not included in our dataset. Hence, neural networks might be prone to misclassifying them. We can overcome this issue by extending the range or parameters used to generate the dataset. In the released code, we provide a check for the out-of-distribution parameters.

Neural network DIMNN model. For the classification task, a deep multilayer perceptron (MLP) architecture combined with the techniques of residual networks (ResNets) was used, as in [44]. Deeper neural networks are capable of capturing highly non-linear relationships in data, but training very deep architectures often leads to optimization difficulties such as vanishing gradients. ResNets address these issues through the introduction of skip (residual) connections, which add the input of a layer to its output. Such mappings facilitate gradient flow and stabilize training, enabling the construction of deeper and more expressive models [45].

The proposed architecture integrates residual connections into a fully connected MLP, allowing the model to benefit from the depth of representation while maintaining stable training dynamics. The network comprises 21 hidden layers with ReLu activations, and the configuration depicted in Fig. S3. The output layer uses a softmax activation. A dropout regularization of 50% is applied throughout the network to mitigate overfitting and enhance generalization. The training process was carried out using AdamW as optimizer, with a learning rate of 0.0005 and incorporating early stopping to prevent overfitting. This architecture benefits from the expressive power of deep learning while incorporating the robustness of residual learning, making it convenient for the proposed classification task.

Acknowledgments

A. F. M. was supported by MCIN/AEI under grant PRE2020-094372; A. F. M. and C. C. were partially funded through MCIN/AEI grants PID2020-117971GB-22 and PID2022-136436NB-I00, as well as AGAUR grant 2021 SGR 00697. R. J. acknowledges support from the fellowship FI-SDUR funded by Generalitat de Catalunya. M. Á. S. acknowledges support from grant no. TED2021-129791B-I00 funded by MCIN/AEI/10.13039/501100011033 and by *European Union NextGenerationEU/PRTR*, and grant no. PID2022-137505NB-C22 funded by MCIN/AEI/10.13039/501100011033 and by ERDF's *A way of making Europe*.

5 Data and Code Availability

The open-source code, along with the code to reproduce the figures, will be available on GitHub upon publication. The real network datasets used in this study are available from the sources referenced in the manuscript and the Supplementary Information. The SYNNET dataset of 792 000 synthetic networks for neural network training, as well as the model checkpoint, will be available at Zenodo upon publication.

Competing Interests Statement

The authors declare no competing interests.

References

[1] Rubén Ballester and Bastian Rieck. On the expressivity of persistent homology in graph learning. In *The Third Learning on Graphs Conference*, (2024).

[2] Gunnar Carlsson and Mikael Vejdemo-Johansson. *Topological data analysis with applications*. Cambridge University Press, (2021).

[3] Herbert Edelsbrunner and John L. Harer. *Computational Topology: An Introduction*. American Mathematical Soceity, (2022).

[4] Larry Wasserman. Topological data analysis. *Annual Review of Statistics and its Application* **5**, 501–532 (2018).

[5] Robin Forman. Bochner's method for cell complexes and combinatorial Ricci curvature. *Discrete & Computational Geometry* **29**, 323–374 (2003).

[6] Linton C. Freeman. A set of measures of centrality based on betweenness. *Sociometry* **40**(10), 35–41 (1977).

[7] Ginestra Bianconi. *Higher-Order Networks*. Cambridge University Press, (2021).

[8] Omer Bobrowski and Dmitri Krioukov. Random simplicial complexes: models and phenomena. In *Higher-Order Systems*, 59–96. Springer (2022).

[9] Daniel Hernández Serrano, Juan Hernández Serrano, and Darío Sánchez Gómez. Simplicial degree in complex networks. Applications of topological data analysis to network science. *Chaos, Solitons and Fractals* **137**, 109839, Aug (2020).

[10] Vsevolod Salnikov, Daniele Cassese, and Renaud Lambiotte. Simplicial complexes and complex systems. *European Journal of Physics* **40**(1), 014001 (2018).

[11] Joaquín J. Torres and Ginestra Bianconi. Simplicial complexes: higher-order spectral dimension and dynamics. *Journal of Physics: Complexity* **1**(1), 015002 (2020).

[12] Danijela Horak, Slobodan Maletić, and Milan Rajković. Persistent homology of complex networks. *J. Stat. Mech.* **2009**, P03034 (2009).

[13] Dane Taylor, Klimm, et al. Topological data analysis of contagion maps for examining spreading processes on networks. *Nature Communications* **6**(1), 7723 (2015).

[14] Harish Kannan, Emil Saucan, Indrava Roy, and Areejit Samal. Persistent homology of unweighted complex networks via discrete Morse theory. *Scientific Reports* **9**(13817) (2019).

[15] Audun Myers, Elizabeth Munch, and Firas A. Khasawneh. Persistent homology of complex networks for dynamic state detection. *Phys. Rev. E* **100**(022314) (2019).

[16] Marco Guerra, Alessandro De Gregorio, Ulderico Fugacci, Giovanni Petri, and Francesco Vaccarino. Homological scaffold via minimal homology bases. *Scientific Reports* **11**(5355) (2021).

[17] Bukyoung Jhun. Topological analysis of the latent geometry of a complex network. *Chaos* **32**(013116) (2022).

[18] G. Petri, P. Expert, F. Turkheimer, R. Carhart-Harris, D. Nutt, P. J. Hellyer, and F. Vaccarino. Homological scaffolds of brain functional networks. *Journal of the Royal Society Interface* **11**(101) (2014).

[19] Chad Giusti, Eva Pastalkova, Carina Curto, and Vladimir Itsko. Clique topology reveals intrinsic geometric structure in neural correlations. *PNAS* **112**(44), 13455–13460 (2015).

[20] Ann E. Sizemore, Jennifer E. Phillips-Cremins, Robert Ghrist, and Danielle S. Bassett. The importance of the whole: Topological data analysis for the network neuroscientist. *Network Neuroscience* **3**(3), 656–673 (2019).

[21] Soumya Das, D. Vijay Anand, and Moo K. Chung. Topological data analysis of human brain networks through order statistics. *PLoS One* **18**(3), e0276419 (2023).

[22] Marián Boguñá, Ivan Bonamassa, Manlio De Domenico, Shlomo Havlin, Dmitri Krioukov, and M. Ángeles Serrano. Network geometry. *Nature Reviews Physics* **3**(2), 114–135 (2021).

[23] P. Almagro, M. Boguñá, and M. Á. Serrano. Detecting the ultra low dimensionality of real networks. *Nature Communications* **13**, 6096, Oct (2022).

[24] M. Ángeles Serrano, Dmitri Krioukov, and Marián Boguñá. Self-similarity of complex networks and hidden metric spaces. *Physical Review Letters* **100**(7), 078701 (2008).

[25] Gabriel Budel, Maksim Kitsak, Rodrigo Aldecoa, Konstantin Zuev, and Dmitri Krioukov. Random hyperbolic graphs in $d+1$ dimensions. *Physical Review E* **109**(5), 054131 (2024).

[26] Robert Jankowski, Antoine Allard, Marián Boguñá, and M. Ángeles Serrano. The D-Mercator method for the multidimensional hyperbolic embedding of real networks. *Nature Communications* **14**(1), 7585 (2023).

[27] M. Á. Serrano and M. Boguñá. *The Shortest Path to Network Geometry*. Cambridge University Press, Cambridge, UK, (2021).

[28] D. Krioukov, F. Papadopoulos, M. Kitsak, A. Vahdat, and M. Boguñá. Hyperbolic geometry of complex networks. *Physical Review E – Statistical, Nonlinear, and Soft Matter Physics* **82**(9), Sep (2010).

[29] Michele Starnini, Elisenda Ortiz, and M. Ángeles Serrano. Geometric randomization of real networks with prescribed degree sequence. *New Journal of Physics* **21**(5), 053039 (2019).

[30] Amir Ghasemian, Homa Hosseinmardi, and Aaron Clauset. Evaluating overfit and underfit in models of network community structure. *IEEE Transactions on Knowledge and Data Engineering* **32**(9), 1722–1735 (2019).

[31] Meritxell Vila Miñana, Robert Jankowski, Aina Ferrà Marcús, Rubén Ballester, M. Ángeles Serrano, and Carles Casacuberta. Detecting graph isomorphism with persistent homology on motif-based filtrations. *In preparation* (2025).

[32] M. E. Aktas, E. Akbas, and A. El Fatmaoui. Persistence homology of networks: Methods and applications. *Applied Network Science* **4**(61) (2019).

[33] Marián Boguñá, Romualdo Pastor-Satorras, and Alessandro Vespignani. Cut-offs and finite size effects in scale-free networks. *The European Physical Journal B* **38**, 205–209 (2004).

[34] George Marsaglia. Choosing a point from the surface of a sphere. *The Annals of Mathematical Statistics* **43**(2), 645–646 (1972).

[35] M. Ángeles Serrano and Marián Boguñá. Clustering in complex networks. I. General formalism. *Physical Review E – Statistical, Nonlinear, and Soft Matter Physics* **74**(5), 056114 (2006).

[36] John Nickolls, Ian Buck, Michael Garland, and Kevin Skadron. Scalable parallel programming with CUDA. In *ACM SIGGRAPH 2008 Classes*, SIGGRAPH '08 (Association for Computing Machinery, New York, NY, USA, 2008).

[37] Allen Hatcher. *Algebraic Topology*. Cambridge University Press, (2002).

[38] David Cohen-Steiner, Herbert Edelsbrunner, and John Harer. Extending persistence using Poincaré and Lefschetz duality. *Foundations of Computational Mathematics* **9**(1), 79–103 (2009).

[39] M. Carrière, F. Chazal, Y. Ike, T. Lacombe, M. Royer, and Y. Umeda. PersLay: A neural network layer for persistence diagrams and new graph topological signatures. In *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 108 of *PMLR*, (2020).

[40] The GUDHI Project. *GUDHI User and Reference Manual*. GUDHI Editorial Board, (2015).

[41] Réka Albert and Albert-László Barabási. Statistical mechanics of complex networks. *Rev. Mod. Phys* **74**(47) (2002).

[42] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, 1097–1105, (2012).

[43] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature* **521**(7553), 436–444 (2015).

[44] Hugo Touvron et al. ResMLP: Feedforward networks for image classification with data-efficient training. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **45**(4), 5314–5321 (2023).

[45] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778, (2016).

Supplementary Information for "Chordless cycle filtrations for dimensionality detection in complex networks via topological data analysis"

Aina Ferrà Marcús,[1] Robert Jankowski,[2,3] Meritxell Vila Miñana,[4] Carles Casacuberta,[1] and M. Ángeles Serrano[2,3,5]

[1] *Departament de Matèmatiques i Informàtica, Universitat de Barcelona, Gran Via de les Corts Catalanes 585, 08007 Barcelona, Spain*
[2] *Departament de Física de la Matèria Condensada, Universitat de Barcelona, Martí i Franquès 1, E-08028 Barcelona, Spain*
[3] *Universitat de Barcelona Institute of Complex Systems (UBICS), Universitat de Barcelona, Barcelona, Spain*
[4] *Center for Complex Networks and Systems Research, Luddy School of Informatics, Computing, and Engineering, Indiana University, Bloomington, IN, USA*
[5] *ICREA, Passeig Lluís Companys 23, E-08010 Barcelona, Spain**

CONTENTS

1. Dimensionality estimation of real networks with persistence descriptors 2

2. Neural network architecture and predictions 4

3. Topological properties of real networks and their inferred dimensions 7

4. Network generation algorithms 11

Supplementary References 11

* marian.serrano@ub.edu

1. DIMENSIONALITY ESTIMATION OF REAL NETWORKS WITH PERSISTENCE DESCRIPTORS

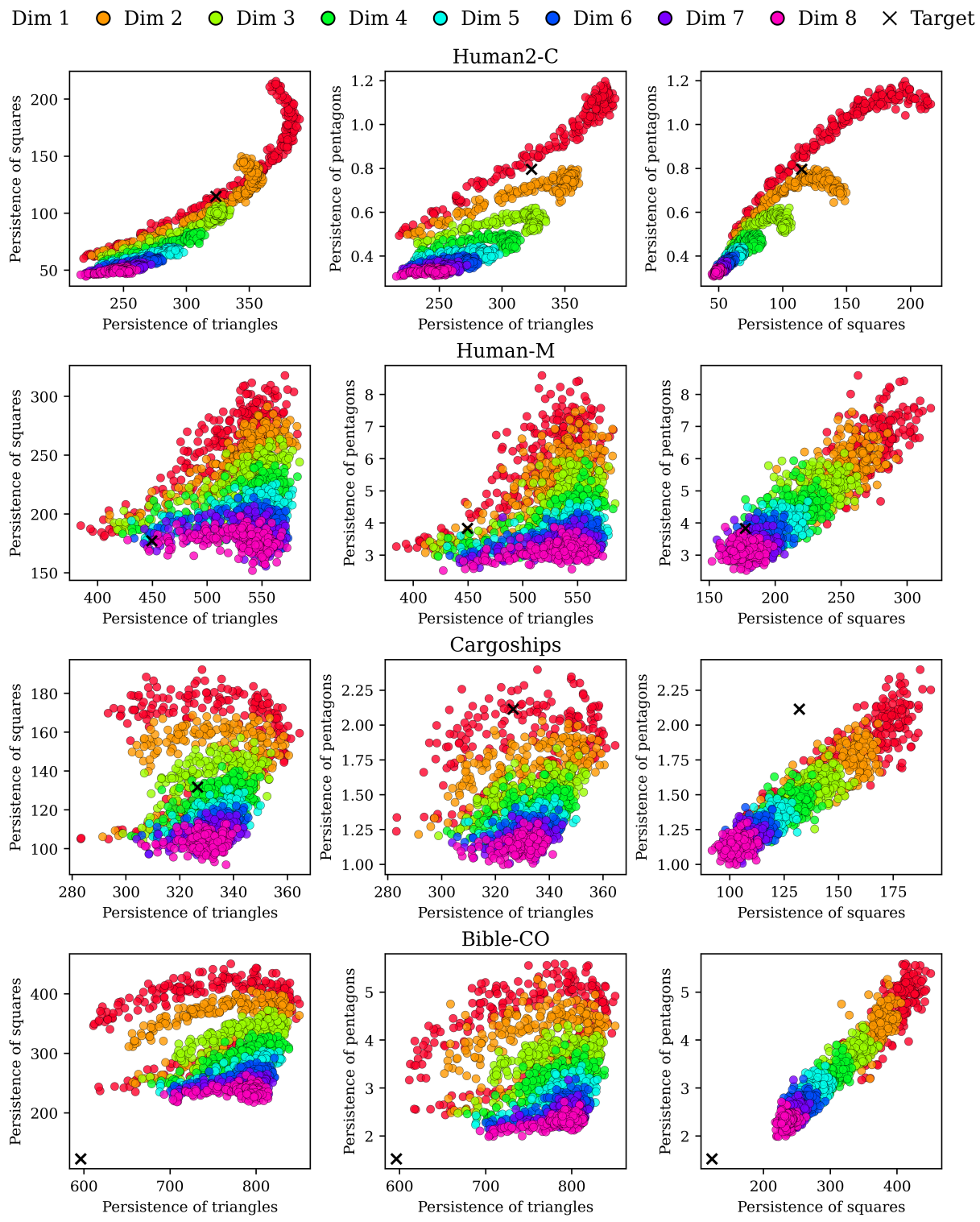

FIG. S1: 2D projections representing an ensemble of 1330 surrogates for each real network in the phase space of total persistence computed from three chordless cycle densities (triangles, squares, and pentagons). Points are colored by dimension, and the target network is marked with a black cross. Part 1 of 2.

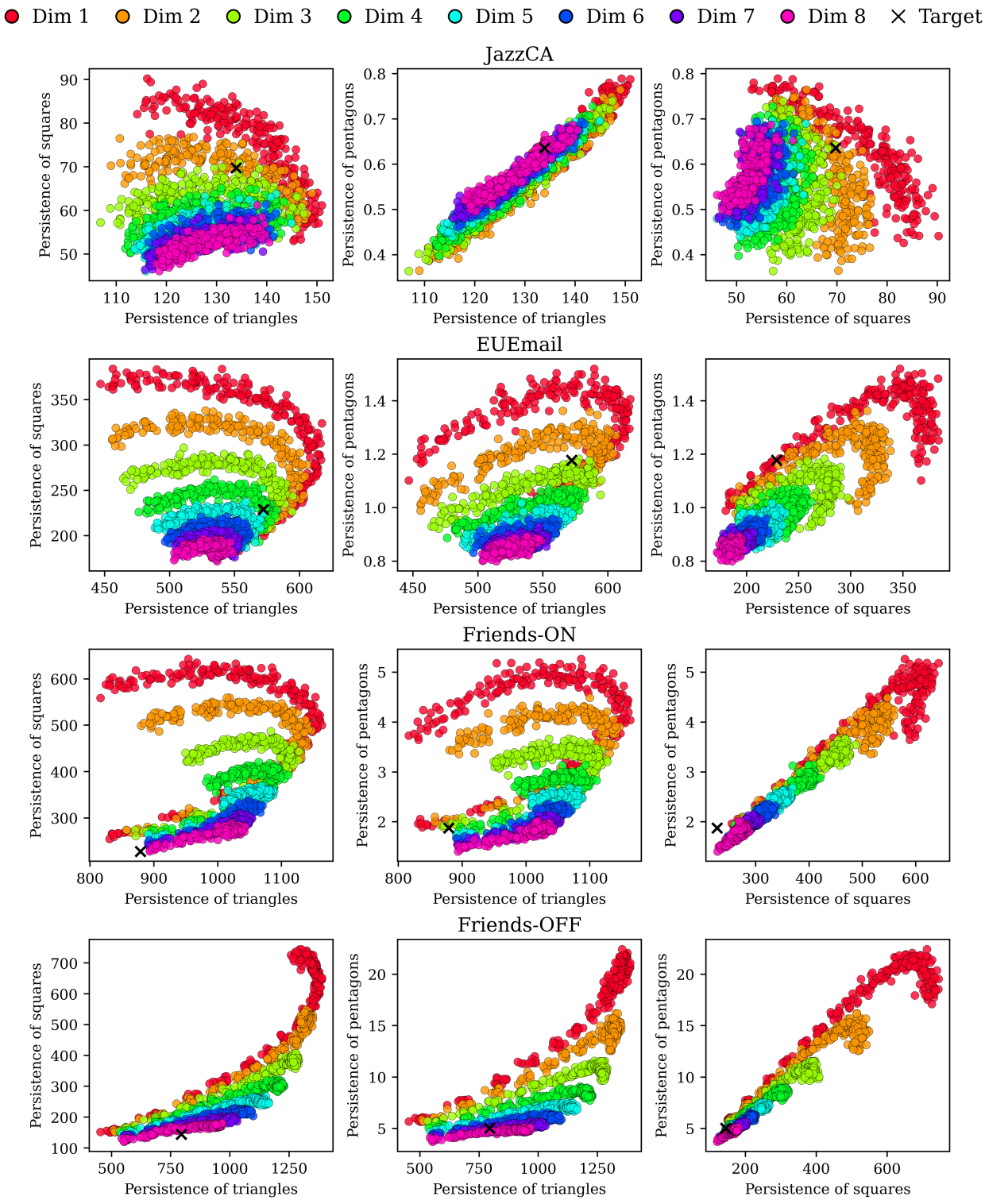

FIG. S2: 2D projections representing an ensemble of 1330 surrogates for each real network in the phase space of total persistence computed from three chordless cycle densities (triangles, squares, and pentagons). Points are colored by dimension, and the target network is marked with a black cross. Part 2 of 2.

2. NEURAL NETWORK ARCHITECTURE AND PREDICTIONS

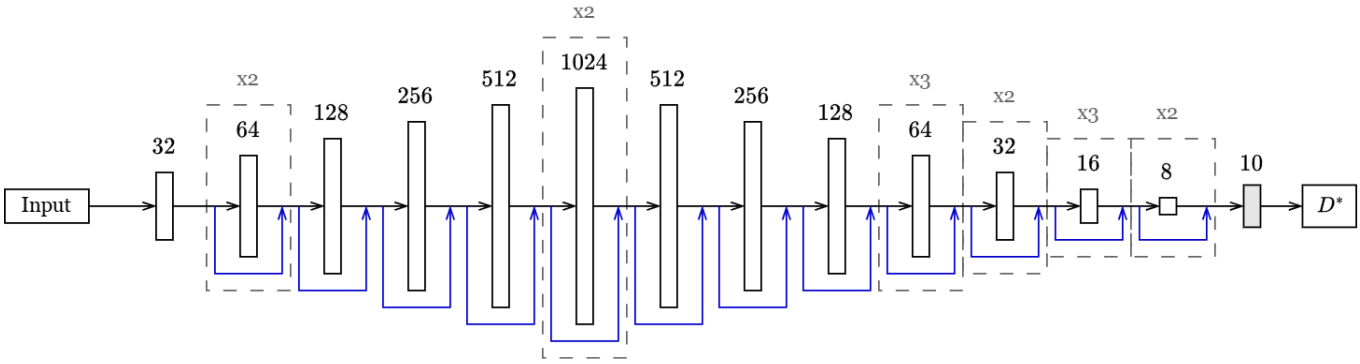

FIG. S3: Schematic representation of the DimNN neural network architecture. Boxes indicate layers within the MLP, with the number above each box specifying the number of neurons in that layer. Blue arrows are used to denote skip (residual) connections. Dashed gray boxes denote repeated layer blocks, with the repetition count indicated above each block. The output layer is shaded in gray and employs a softmax activation, yielding a 10-dimensional output corresponding to the number of classes in the classification problem.

TABLE S1: Validation accuracies after five repetitions of DimNN, a ResNet training with AdamW optimizer using the SynNet network database. Columns correspond to features included in each model: N number of nodes, $\langle k \rangle$ average degree, $\langle k^2 \rangle / \langle k \rangle^2$ normalized second moment, $k_{\min}$ minimum degree, $k_{\max}$ maximum degree, $\langle k_{\text{nn}} \rangle$ mean average neighbor degree, C_t average triangle density, C_s average chordless square density, C_p average chordless pentagon density, TP_t, TP_s, TP_p total persistence in homological dimension 1 computed from chordless cycle filtrations (triangles, squares, and pentagons, respectively).

N	$\langle k \rangle$	$\langle k^2 \rangle / \langle k \rangle^2$	$k_{\min}$	$k_{\max}$	$\langle k_{\text{nn}} \rangle$	C_t	C_s	C_p	TP_t	TP_s	TP_p	Accuracy (%)	Time (sec)	Epochs
						✓	✓	✓				53.20 ± 2.15	7529.71	73.40
✓						✓	✓	✓				66.79 ± 0.67	7582.86	73.60
✓	✓					✓	✓	✓				68.95 ± 2.08	5718.05	55.50
✓	✓	✓				✓	✓	✓				81.53 ± 0.67	6372.85	61.40
✓	✓	✓	✓	✓		✓	✓	✓				80.96 ± 0.46	5449.96	52.75
✓	✓	✓	✓	✓	✓	✓	✓	✓				82.32 ± 1.39	4890.98	47.40
									✓	✓	✓	56.06 ± 1.17	8326.58	74.40
✓									✓	✓	✓	59.02 ± 2.73	6348.86	65.00
✓	✓								✓	✓	✓	59.45 ± 1.52	6015.84	60.00
✓	✓	✓							✓	✓	✓	71.60 ± 2.30	6563.73	64.67
✓	✓	✓	✓	✓					✓	✓	✓	70.74 ± 2.57	6663.77	65.80
✓	✓	✓	✓	✓	✓				✓	✓	✓	73.10 ± 1.19	5592.25	54.80
						✓	✓	✓	✓	✓	✓	72.64 ± 3.32	6518.76	59.20
✓						✓	✓	✓	✓	✓	✓	74.57 ± 1.86	6343.14	57.00
✓	✓					✓	✓	✓	✓	✓	✓	74.39 ± 2.76	7928.53	59.40
✓	✓	✓				✓	✓	✓	✓	✓	✓	81.83 ± 0.44	7682.31	53.00
✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	81.41 ± 0.38	5778.89	37.60
✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	$\mathbf{83.00 \pm 0.38}$	4796.51	42.60

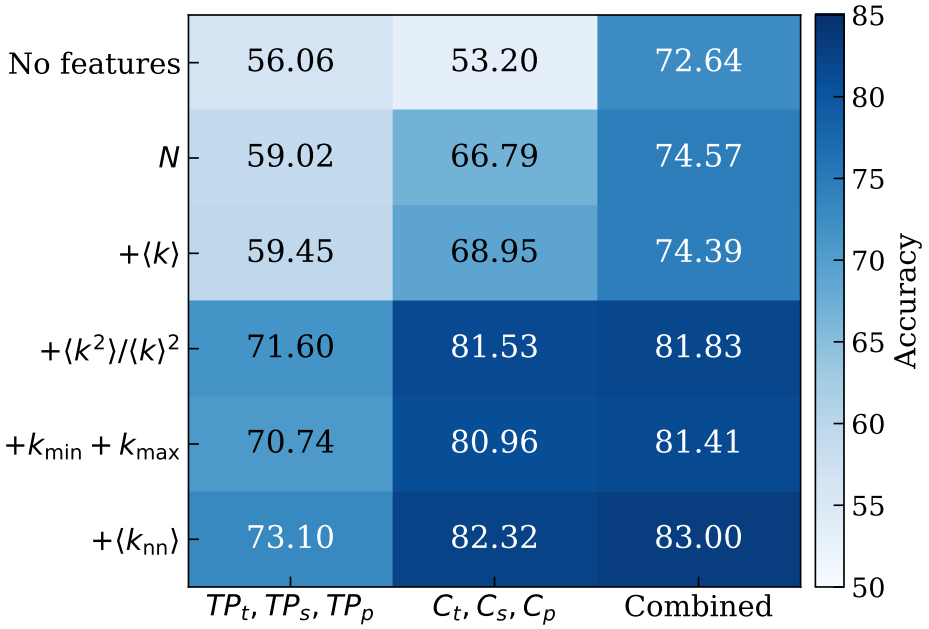

FIG. S4: Impact of cycle density and TDA features on DimNN validation accuracy on the SynNet dataset. Heatmap of mean validation accuracy for five progressively enriched input network features. The results are averaged over five neural network initializations.

TABLE S2: Comparative measures for each combination of graph features between the predictions of the DimNN classifier and a regressor with the same architecture and mean squared error as loss function, averaged over five repetitions. Discrepancy is computed from the collection of SynNet networks in the validation set on which the two models do not agree. *Agreement (%):* Percentage of agreement between the two models; *Median:* Median discrepancy; *Q3:* Third quartile of discrepancy; *Mean:* Mean of discrepancy; *Std:* Standard deviation of discrepancy.

Descriptors	**Agreement (%)**	**Median**	**Q3**	**Mean**	**Std**
C_t, C_s, C_p	44.31	1.0	2.0	1.41	0.69
N, C_t, C_s, C_p	67.12	1.0	1.0	1.19	0.46
$N, \langle k\rangle, C_t, C_s, C_p$	73.18	1.0	1.0	1.18	0.45
$N, \langle k\rangle, \langle k^2\rangle/\langle k\rangle^2, C_t, C_s, C_p$	88.45	1.0	1.0	1.04	0.24
$N, \langle k\rangle, \langle k^2\rangle/\langle k\rangle^2, k_{\min}, k_{\max}, C_t, C_s, C_p$	88.19	1.0	1.0	1.06	0.30
$N, \langle k\rangle, \langle k^2\rangle/\langle k\rangle^2, k_{\min}, k_{\max}, \langle k_{\rm nn}\rangle, C_t, C_s, C_p$	88.90	1.0	1.0	1.05	0.27
TP_t, TP_s, TP_p	58.83	1.0	1.4	1.30	0.63
N, TP_t, TP_s, TP_p	67.75	1.0	1.0	1.22	0.56
$N, \langle k\rangle, TP_t, TP_s, TP_p$	71.06	1.0	1.0	1.18	0.48
$N, \langle k\rangle, \langle k^2\rangle/\langle k\rangle^2, TP_t, TP_s, TP_p$	79.14	1.0	1.0	1.12	0.39
$N, \langle k\rangle, \langle k^2\rangle/\langle k\rangle^2, k_{\min}, k_{\max}, TP_t, TP_s, TP_p$	77.70	1.0	1.0	1.12	0.40
$N, \langle k\rangle, \langle k^2\rangle/\langle k\rangle^2, k_{\min}, k_{\max}, \langle k_{\rm nn}\rangle, TP_t, TP_s, TP_p$	80.43	1.0	1.0	1.11	0.40
$C_t, C_s, C_p, TP_t, TP_s, TP_p$	79.33	1.0	1.0	1.05	0.24
$N, C_t, C_s, C_p, TP_t, TP_s, TP_p$	79.62	1.0	1.0	1.06	0.27
$N, \langle k\rangle, C_t, C_s, C_p, TP_t, TP_s, TP_p$	78.65	1.0	1.0	1.05	0.24
$N, \langle k\rangle, \langle k^2\rangle/\langle k\rangle^2, C_t, C_s, C_p, TP_t, TP_s, TP_p$	88.38	1.0	1.0	1.04	0.24
$N, \langle k\rangle, \langle k^2\rangle/\langle k\rangle^2, k_{\min}, k_{\max}, C_t, C_s, C_p, TP_t, TP_s, TP_p$	87.79	1.0	1.0	1.06	0.28
$N, \langle k\rangle, \langle k^2\rangle/\langle k\rangle^2, k_{\min}, k_{\max}, \langle k_{\rm nn}\rangle, C_t, C_s, C_p, TP_t, TP_s, TP_p$	88.86	1.0	1.0	1.05	0.25

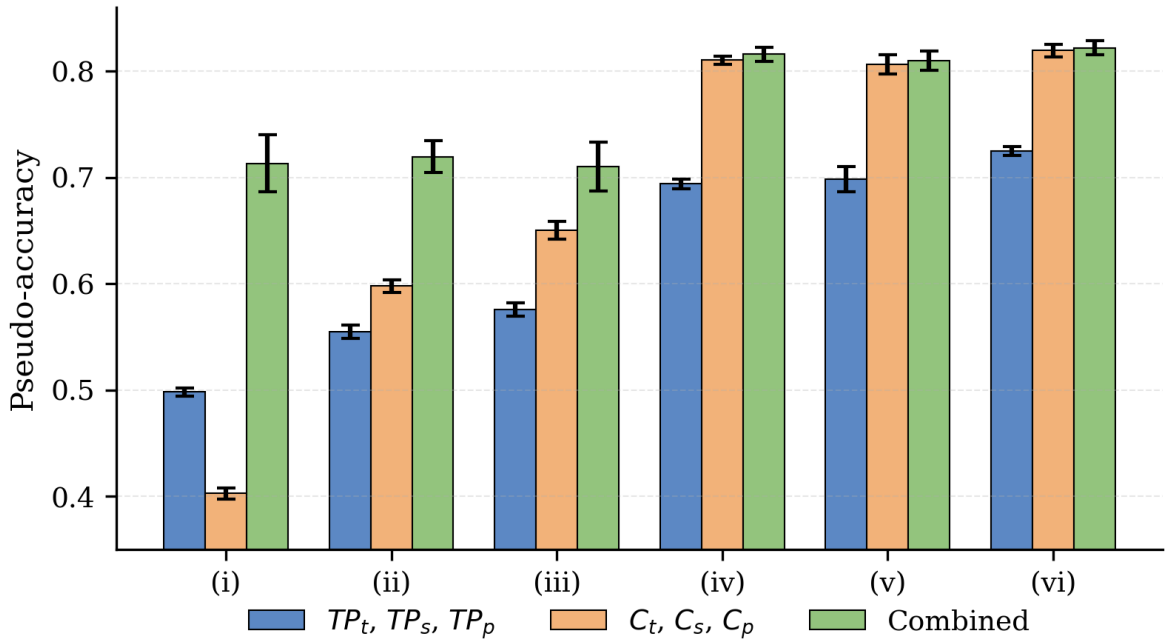

FIG. S5: Averages and standard deviations of pseudo-accuracy on a validation set after five runs of a neural network regressor using the SynNet database. Pseudo-accuracy refers to the fact that the prediction of a regressor is a real number, which is approximated to the nearest integer. *Blue:* Accuracies obtained using total persistences TP_t, TP_s, TP_p, plus cumulative features. *Orange:* Accuracies using average cycle densities C_t, C_s, C_p, plus cumulative features. *Green:* Accuracies obtained by combining total persistences and mean cycle densities plus cumulative features. Successive columns correspond to incorporating one after the other the following additional features into the model: (i) no added features; (ii) number of nodes N; (iii) number of nodes and average degree $\langle k \rangle$; (iv) number of nodes, average degree, and normalized second moment $\langle k^2 \rangle / \langle k \rangle^2$; (v) number of nodes, average degree, normalized second moment, minimum degree $k_{\min}$ and maximum degree $k_{\max}$; (vi) number of nodes, average degree, normalized second moment, minimum degree, maximum degree, and mean average neighbor degree $\langle k_{\mathrm{nn}} \rangle$.

3. TOPOLOGICAL PROPERTIES OF REAL NETWORKS AND THEIR INFERRED DIMENSIONS

We compile a dataset of 53 real-world networks from various domains. For more details about each network, we refer to [1] and [2].

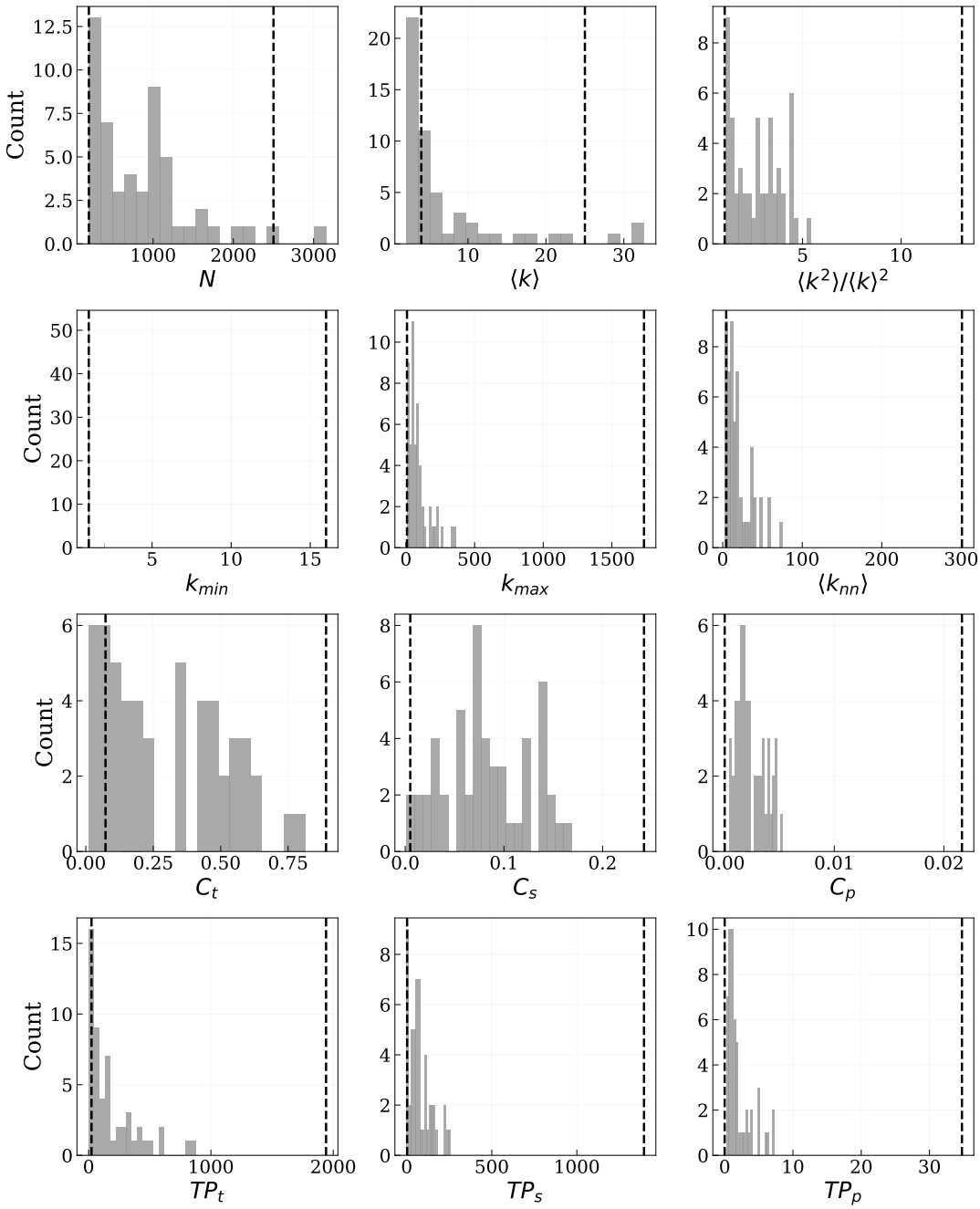

FIG. S6: Distributions of the topological properties of real networks. The dotted black lines in each panel indicate the minimum and maximum value of a given feature in the training dataset.

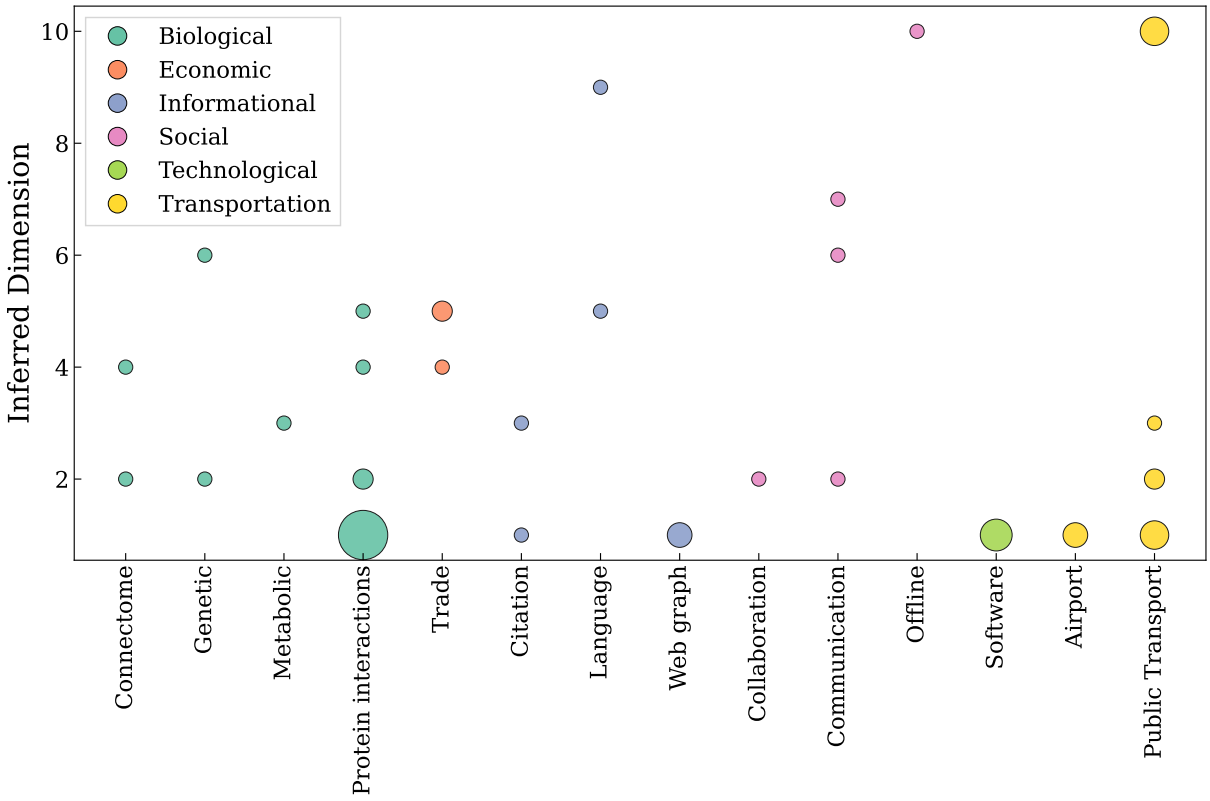

FIG. S7: Inferred dimension of real networks grouped by subdomain category.

TABLE S3: Properties of real networks, with network clustering, path and accuracy measures, and their dimensionality inferred with the DimNN model. Part 1 of 2.

Network	Domain	Subdomain	N	$\langle k \rangle$	$\langle k \rangle^2 / \langle k \rangle^2$	$k_{\min}$	$k_{\max}$
Binary interactomes (various species; 2012)	Biological	Protein interactions	974	4.72	3.52	1	55
Uetz screen yeast interactome (2000)	Biological	Protein interactions	263	2.14	1.86	1	17
Reguly yeast interactome (2006)	Biological	Protein interactions	1213	4.21	2.11	1	40
Ito core yeast interactome (2001)	Biological	Protein interactions	426	2.45	3.44	1	56
Yu yeast interactomes (2008)	Biological	Protein interactions	964	3.09	3.86	1	85
C. elegans interactomes (2009)	Biological	Protein interactions	269	21.58	1.55	1	79
Binary interactomes (musculus mouse; 2012)	Biological	Protein interactions	537	3.83	2.99	1	59
Binary interactomes (homo sapiens; 2012)	Biological	Protein interactions	3155	3.58	3.31	1	131
Binary interactomes (musculus mouse; 2012)	Biological	Protein interactions	705	3.53	2.92	1	59
C. elegans interactomes (2009)	Biological	Protein interactions	2214	3.20	4.50	1	99
Binary interactomes (musculus mouse; 2012)	Biological	Protein interactions	890	2.99	4.50	1	76
C. elegans interactomes (scaffold, 2009)	Biological	Protein interactions	345	2.32	2.67	1	28
Yu yeast interactomes (2008)	Biological	Protein interactions	1647	3.06	3.82	1	89
C. elegans interactomes (genetic, 2009)	Biological	Protein interactions	683	4.52	2.39	1	56
C. elegans interactomes (WI-2004, 2009)	Biological	Protein interactions	1084	2.96	4.42	1	74
C. elegans interactomes (WI-2007, 2009)	Biological	Protein interactions	1108	2.71	4.42	1	84
Human2-C	Biological	Connectome	496	32.41	1.17	1	80
Human-M	Biological	Metabolic	1436	6.57	4.77	1	224
Malaria var DBLa HVR networks	Biological	Genetic	291	22.34	1.48	1	61
Malaria var DBLa HVR networks	Biological	Genetic	298	18.01	1.37	1	48
Cargoships	Economic	Trade	821	10.58	3.12	1	173
Atlas of Economic Complexity export network	Economic	Trade	774	4.59	2.70	1	43
Atlas of Economic Complexity export network	Economic	Trade	866	5.85	2.81	1	48
Garfield's citation networks (2001)	Informational	Citation	1024	9.60	3.95	1	232
AMiner citation network (2009)	Informational	Citation	1350	6.01	2.08	1	96
Roget's Thesaurus (1879)	Informational	Language	994	7.32	1.44	1	28
Bible-CO	Informational	Language	1707	10.61	3.92	2	364
WebKB graphs (1998)	Informational	Web graph	269	3.60	4.36	1	74
WebKB graphs (1998)	Informational	Web graph	280	3.95	2.75	1	57
WebKB graphs (1998)	Informational	Web graph	343	4.04	5.43	1	129
Friends-OFF	Social	Offline	2539	8.24	1.27	1	27
Friends-ON	Social	Communication	2000	16.10	2.72	1	273
EUEmail	Social	Communication	986	32.58	2.29	1	345
Jazz-CA	Social	Collaboration	199	29.23	1.51	1	189
Javax, Java, Jung, AppEngine dependencies (2010)	Technological	Software	1031	8.55	3.29	1	208
Software function-callgraphs (AbiWord, 2002)	Technological	Software	1035	3.32	3.27	1	89
Software function-callgraphs (VTK, 2002)	Technological	Software	771	3.52	3.66	1	83
Javax, Java, Jung, AppEngine dependencies (2010)	Technological	Software	435	5.99	2.59	1	109
Guava library dependencies (2012)	Technological	Software	457	4.05	3.85	1	62
World subways (2009)	Transportation	Public Transport	266	2.32	1.17	1	7
European airline network	Transportation	Airport	417	14.16	3.10	1	112
FAA Preferred Routes (2010)	Transportation	Airport	1226	3.93	1.87	1	34
US airport network (top 500; 2002)	Transportation	Airport	500	11.92	4.51	1	145
London Transport Network	Transportation	Public Transport	369	2.33	1.19	1	7
India bus routes (2016)	Transportation	Public Transport	1103	3.90	1.68	1	54
India bus routes (2016)	Transportation	Public Transport	1554	4.63	1.84	1	52
India bus routes (2016)	Transportation	Public Transport	1087	5.43	3.41	1	183
World subways (2009)	Transportation	Public Transport	217	2.41	1.22	1	9
World subways (2009)	Transportation	Public Transport	392	2.23	1.11	1	6
World subways (2009)	Transportation	Public Transport	299	2.38	1.22	1	8
World subways (2009)	Transportation	Public Transport	209	2.30	1.16	1	7
India bus routes (2016)	Transportation	Public Transport	1009	3.19	1.49	1	15
World subways (2009)	Transportation	Public Transport	433	2.19	1.13	1	8

TABLE S4: Properties of real networks, with network clustering, path and accuracy measures, and their dimensionality inferred with the DimNN model. Part 2 of 2.

Network	C_t	C_s	C_p	TP_t	TP_s	TP_p	Acc.	D
Binary interactomes (various species; 2012)	0.1220	0.1228	0.0031	77.7921	49.5511	1.5393	1.000	1
Uetz screen yeast interactome (2000)	0.0262	0.0251	0.0020	1.7500	2.1659	0.2455	1.000	1
Reguly yeast interactome (2006)	0.6015	0.0761	0.0020	227.0699	107.9266	3.2453	0.970	2
Ito core yeast interactome (2001)	0.1345	0.0875	0.0034	11.6167	15.8933	0.7123	1.000	1
Yu yeast interactomes (2008)	0.1217	0.1189	0.0015	39.2622	55.9388	1.0447	1.000	1
C. elegans interactomes (2009)	0.4635	0.1516	0.0021	149.4319	63.0049	0.7239	0.551	2
Binary interactomes (musculus mouse; 2012)	0.1646	0.1508	0.0027	40.5982	50.2391	1.2342	1.000	1
Binary interactomes (homo sapiens; 2012)	0.1933	0.1161	0.0013	301.0725	258.7704	3.5773	0.956	1
Binary interactomes (musculus mouse; 2012)	0.1744	0.1392	0.0023	50.8065	55.3945	1.2488	0.993	1
C. elegans interactomes (2009)	0.0728	0.0519	0.0010	82.7588	70.5024	1.2652	1.000	1
Binary interactomes (musculus mouse; 2012)	0.3684	0.0608	0.0015	82.2267	35.0576	0.9259	0.757	4
C. elegans interactomes (scaffold, 2009)	0.0803	0.0360	0.0034	6.8333	3.9875	0.4651	1.000	1
Yu yeast interactomes (2008)	0.1236	0.0762	0.0013	76.3197	64.8148	1.7078	1.000	1
C. elegans interactomes (genetic, 2009)	0.4244	0.0840	0.0019	153.5247	67.5384	1.5513	0.661	5
C. elegans interactomes (WI-2004, 2009)	0.0843	0.0689	0.0011	38.5890	40.9450	0.6395	1.000	1
C. elegans interactomes (WI-2007, 2009)	0.0727	0.0666	0.0015	30.0982	31.0904	0.7986	1.000	1
Human2-C	0.5378	0.1061	0.0009	323.3771	114.7546	0.7963	0.825	2
Human-M	0.5866	0.0770	0.0011	449.3374	177.3222	3.8313	0.987	3
Malaria var DBLa HVR networks	0.6054	0.1240	0.0018	179.7420	58.5849	1.0851	0.986	2
Malaria var DBLa HVR networks	0.4510	0.0710	0.0019	161.5944	31.7981	0.6912	0.870	6
Cargoships	0.5482	0.0873	0.0014	326.5759	131.8263	2.1144	0.775	5
Atlas of Economic Complexity export network	0.4882	0.0525	0.0022	109.7275	31.6048	1.5757	0.669	5
Atlas of Economic Complexity export network	0.5089	0.0759	0.0019	165.7158	57.9047	1.7043	0.734	4
Garfield's citation networks (2001)	0.3539	0.0842	0.0013	412.2120	162.5747	1.5339	1.000	1
AMiner citation network (2009)	0.3401	0.0841	0.0027	409.2855	152.9663	5.1329	0.789	3
Roget's Thesaurus (1879)	0.1964	0.0385	0.0018	307.8835	71.9862	2.8535	0.688	9
Bible-CO	0.6259	0.0347	0.0005	596.0701	122.7898	1.5247	0.828	5
WebKB graphs (1998)	0.3710	0.0731	0.0046	45.2202	20.3393	1.4145	1.000	1
WebKB graphs (1998)	0.4226	0.1367	0.0034	39.4806	28.7375	1.2093	0.689	1
WebKB graphs (1998)	0.2513	0.1365	0.0040	57.6453	44.9548	1.7253	1.000	1
Friends-OFF	0.1785	0.0282	0.0013	794.9442	144.4772	5.0352	0.712	10
Friends-ON	0.5241	0.0543	0.0007	878.8888	227.5659	1.8765	0.996	7
EUEmail	0.4735	0.1363	0.0009	572.1027	228.7467	1.1775	1.000	2
Jazz-CA	0.7568	0.1201	0.0010	133.8771	69.7319	0.6359	0.994	2
Javax, Java, Jung, AppEngine dependencies (2010)	0.4270	0.1397	0.0021	498.0842	235.9069	4.8104	1.000	1
Software function-callgraphs (AbiWord, 2002)	0.1398	0.0966	0.0037	78.2229	65.4463	2.6173	1.000	1
Software function-callgraphs (VTK, 2002)	0.1555	0.0939	0.0034	67.1126	47.4782	1.7499	1.000	1
Javax, Java, Jung, AppEngine dependencies (2010)	0.4683	0.1547	0.0041	154.3303	103.4080	3.3866	0.996	1
Guava library dependencies (2012)	0.6199	0.1396	0.0016	73.3721	55.9591	0.6418	0.976	1
World subways (2009)	0.0642	0.0284	0.0046	7.9000	4.9694	0.5619	0.389	10
European airline network	0.5508	0.1690	0.0018	165.0670	108.7857	1.2235	1.000	1
FAA Preferred Routes (2010)	0.1089	0.0527	0.0047	130.2926	70.0767	6.0710	0.930	1
US airport network (top 500; 2002)	0.8137	0.0907	0.0004	117.4738	106.0472	0.7758	1.000	1
London Transport Network	0.0528	0.0315	0.0023	9.9000	7.4528	0.5900	1.000	1
India bus routes (2016)	0.2469	0.0568	0.0053	225.6782	83.1205	7.2559	0.524	2
India bus routes (2016)	0.2359	0.0735	0.0030	359.0307	158.5868	6.9280	0.581	2
India bus routes (2016)	0.3333	0.0691	0.0032	329.8021	135.2038	6.3026	0.758	3
World subways (2009)	0.0453	0.0150	0.0047	6.0000	2.3667	0.5911	0.575	10
World subways (2009)	0.0111	0.0031	0.0040	2.6667	1.1250	1.0887	0.964	10
World subways (2009)	0.0393	0.0265	0.0040	6.8333	5.0270	0.9486	0.448	10
World subways (2009)	0.0128	0.0179	0.0015	1.6667	2.3889	0.2475	1.000	1
India bus routes (2016)	0.1282	0.1001	0.0044	95.3556	77.7124	3.9028	0.845	1
World subways (2009)	0.0303	0.0015	0.0022	6.6667	0.2571	0.5071	1.000	1

4. NETWORK GENERATION ALGORITHMS

Algorithm 1 $\mathbb{S}^D$ network generation algorithm

Input: Dimension D, number of nodes N, power-law exponent γ, average degree $\langle\kappa\rangle$, inverse temperature β
Output: Graph $G = (V, E)$ with nodes expressed in angular coordinates
$V \leftarrow \{\}$
$E \leftarrow \{\}$
$\kappa_0 \leftarrow \langle k\rangle(\gamma-2)(1-N^{-1})/[(\gamma-1)(1-N^{(2-\gamma)/(\gamma-1)})]$
$\kappa_c \leftarrow \kappa_0\, N^{1/(\gamma-1)}$
for $i = 1$ to N **do**
    $\kappa_i \sim \rho(\kappa) = \kappa^{-\gamma}\kappa_0^{\gamma-1}(\gamma-1)/(1-(\kappa_c/\kappa_0)^{1-\gamma})$
    $\theta_i \sim \text{Uniform}([0, 2\pi))$
    $V \leftarrow V \cup \{\theta_i\}$
end for
$\mu \leftarrow \beta\,\Gamma(D/2)\sin(D\pi/\beta)/(2\pi^{1+D/2}\langle k\rangle)$
for $i = 1$ to N **do**
    for $j = i+1$ to N **do**
        $p_{ij} \leftarrow 1/(1+[R\Delta\theta_{ij}/(\mu\kappa_i\kappa_j)^{1/D}]^{\beta})$
        $E \leftarrow E \cup \{(i,j)\}$ with probability p_{ij}
    end for
end for

Algorithm 2 Geometric randomization rewiring (D-GR)

Input: Graph $G = (V, E)$, dimension D, inverse temperature β
Output: Graph $G = (V, E)$ with rewired edges
for $t = 1$ to $10 \cdot |E|$ **do**
    $(i,j),(l,m) \sim \text{Uniform}(E)$ without replacement
    $\mathcal{L}_n/\mathcal{L}_c \leftarrow (\Delta\theta_{ij}\,\Delta\theta_{lm}/[\Delta\theta_{il}\,\Delta\theta_{jm}])^{\beta}$
    if $\mathcal{L}_n/\mathcal{L}_c > 1$ **then**
        $E \leftarrow E \cup \{(i,m),(l,j)\} \setminus \{(i,j),(l,m)\}$
    else
        $E \leftarrow E \cup \{(i,m),(l,j)\} \setminus \{(i,j),(l,m)\}$ with probability $\mathcal{L}_n/\mathcal{L}_c$
    end if
    if $|\mathcal{L}_n/\mathcal{L}_c - 1| < 10^{-6}$ **then**
        break
    end if
end for

SUPPLEMENTARY REFERENCES

[1] P. Almagro, M. Boguñá, and M. Á. Serrano, Nature Communications **13**, 6096 (2022)
[2] A. Ghasemian, H. Hosseinmardi, and A. Clauset, IEEE Transactions on Knowledge and Data Engineering **32**, 1722 (2019)