nature machine intelligence

Article

https://doi.org/10.1038/s42256-023-00749-8

Topological structure of complex predictions

Received: 24 August 2022

Accepted: 28 September 2023

Published online: 17 November 2023

Check for updates

Meng Liu ✉, Tamal K. Dey & David F. Gleich ✉

Current complex prediction models are the result of fitting deep neural networks, graph convolutional networks or transducers to a set of training data. A key challenge with these models is that they are highly parameterized, which makes describing and interpreting the prediction strategies difficult. We use topological data analysis to transform these complex prediction models into a simplified topological view of the prediction landscape. The result is a map of the predictions that enables inspection of the model results with more specificity than dimensionality-reduction methods such as tSNE and UMAP. The methods scale up to large datasets across different domains. We present a case study of a transformer-based model previously designed to predict expression levels of a piece of DNA in thousands of genomic tracks. When the model is used to study mutations in the *BRCA1* gene, our topological analysis shows that it is sensitive to the location of a mutation and the exon structure of *BRCA1* in ways that cannot be found with tools based on dimensionality reduction. Moreover, the topological frame work offers multiple ways to inspect results, including an error estimate that is more accurate than model uncertainty. Further studies show how these ideas produce useful results in graph-based learning and image classification.

Deep learning is a successful strategy where a highly parameterized model makes human-like predictions across many fields[1–4]. Yet challenges in both interpretation and generalization often keep deep learning from use in practice[5,6]. Deep-learned models and their specific prediction mechanisms are difficult to assess directly due to the large collection of model parameters. Inspection methods such as activation or saliency maps[7,8] highlight only the results for a single prediction with their own limitations[9]. Likewise, influence estimation techniques[10] often produce a ranked list of samples. These tend to be most useful to understand issues retrospectively, after they have been identified. In comparison, global data visualizations such as tSNE[11] and UMAP[12,13] offer the power to inspect the global space of predictions among large collections of data. In principle, these methods offer the ability to prospectively identify those problematic data regions; however, the dimension reduction inherent to these methods may distort properties of the data.

Topological data analysis, on the other hand, excels at distilling representation-invariant information[14–17] because it seeks to simplify the shape of data in its ambient space without reducing its dimension. Topological data analysis (TDA) of complex predictive models such as deep learning remains in its infancy[18–22]. Existing research focuses on trying to assess the topological properties of the network weights, to assess the topology of the features used by the network, to initialize network weights with topologically consistent operators (GENEOs), or to add topological features to predictions. Our approach seeks to assess the topology of the neural network embeddings, representations of the data, and how they interact with the predictions. By way of an anthropomorphic analogy, we seek to simplify the topological lens with which the neural network sees the data for predictions. Although we say deep learning, our methods are compatible with any mechanisms that outputs a vector of class probability values as discussed in the Supplementary Methods,

Computer Science Department, Purdue University, West Lafayette, IN, USA. ✉e-mail: liu1740@purdue.edu; dgleich@purdue.edu

https://doi.org/10.1038/s42256-023-00749-8

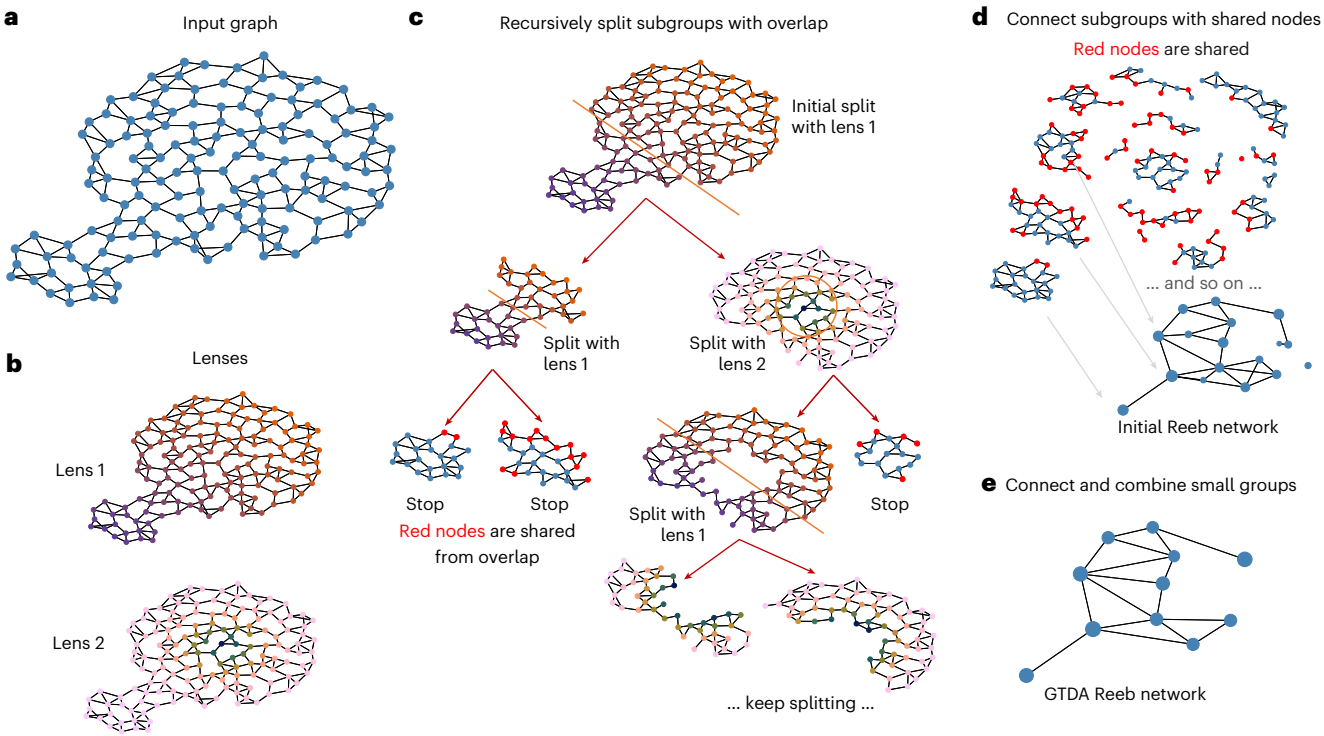


Fig. 1 | Overview of the GTDA method. a,**b**, The GTDA construction of a Reeb network starts with an input graph (**a**) and a set of lenses that assign values to each node of the graph (**b**), where the values are indicated by the node colour. **c**, A Reeb network is a simplification built from overlapping subgroups or clusters in the original data with similar values for the lenses—GTDA builds these using a recursive splitting procedure. At each recursive step, a single lens is chosen and the data are split into parts based on the node values in that lens. The split is done so that there are overlapped nodes around the split boundary. This continues until only small groups remain. **d**, These subgroups are assembled into a Reeb network by simplifying each subgroup to a single Reeb node and connecting Reeb nodes if they share any nodes from the overlapped splits. **e**, The GTDA method further combines and connects small and isolated Reeb nodes to produce the GTDA Reeb network from the graph and lenses.

including more classic techniques such as support vector machines or logistic regressions.

Our GTDA method

We construct a Reeb network to assess the prediction landscape of a neural-network-like prediction method. Reeb networks are discretizations of topological structures called Reeb spaces, which generalize Reeb graphs[17,23]. An example of the differences among these concepts is illustrated in Extended Data Fig. 1 with further discussion in Supplementary Section 1.6. Reeb networks seek to simplify the data while respecting topology. We design a recursive splitting and merging procedure called graph-based topological data analysis (GTDA) to simplify the data.

Our GTDA method builds on the mapper algorithm[15]. Mapper, itself, builds a discrete approximation of a Reeb graph or Reeb space (see Supplementary Section 1.6 and Extended Data Fig. 1). It begins with a set of data points $(x_1, \ldots, x_n)$, along with a single or multivalued function sampled at each data point. The set of all these values $\{f_1, \ldots, f_n\}$ samples a map $f : X \to \mathbb{R}^k$ on a topological space X. The map f is called a filter or lens. The idea is that when f is single-valued, a Reeb graph shows a quotient topology of X with respect to f and mapper discretizes this Reeb graph using the sampled values of f on points $x_1, \ldots, x_n$. Algorithmically, mapper consists of the steps:

1. Sort the values f_i and split them into overlapping bins $B_1, \ldots, B_r$ of the same size.
2. For each bin of values B_j, let S_j denote the set of data points with those values and cluster the data points in each S_j independently (that is, we run a clustering algorithm on each S_j as if it were the entire dataset).
3. Create a node in the Reeb graph for each cluster found in the previous step.
4. Connect the nodes of the Reeb graph if the clusters they represent share a common datapoint.

The resulting graph is a discrete approximation of the Reeb graph and represents a compressed view of the shape underlying the original dataset.

The input data for mapper is usually a point cloud in a high-dimensional space where the point coordinates are used only in the clustering step. In our methodology, we are interested in datasets that are even more general. Graph inputs provide this generality. Datasets not in graph format such as images or DNA sequences can be easily transformed into graphs by first extracting intermediate outputs of the model as embeddings and then building a nearest-neighbour graph from the embedding matrix. The resulting graph then facilitates easy clustering: for each subset of points, we extract the subgraph induced by those points and then use a parameter-free connected-components analysis to generate clusters. Our method could also work with point-cloud data and clustering directly through standard relationships between graph-based algorithms and point-cloud-based algorithms. We focus on the graph-based approach both for simplicity and because we found it the most helpful for these prediction applications.

The GTDA method therefore begins with a graph representing relationships among data points and a set of values over each node called lenses (Fig. 1a,b); the terminology of lenses arises from a work by Lum and colleagues[17]. In the applications we consider, the lenses we use are the prediction matrix of a neural network model where P_{ij} is the probability that sample i belongs to class j. Graph-based TDA uses a recursive splitting strategy to build the bins in the multidimensional space (Fig. 1c), instead of tensor product bin constructions as in multidimensional generalizations of mapper. Detailed pseudo code for this procedure can be found in Supplementary Algorithm 1. An

https://doi.org/10.1038/s42256-023-00749-8

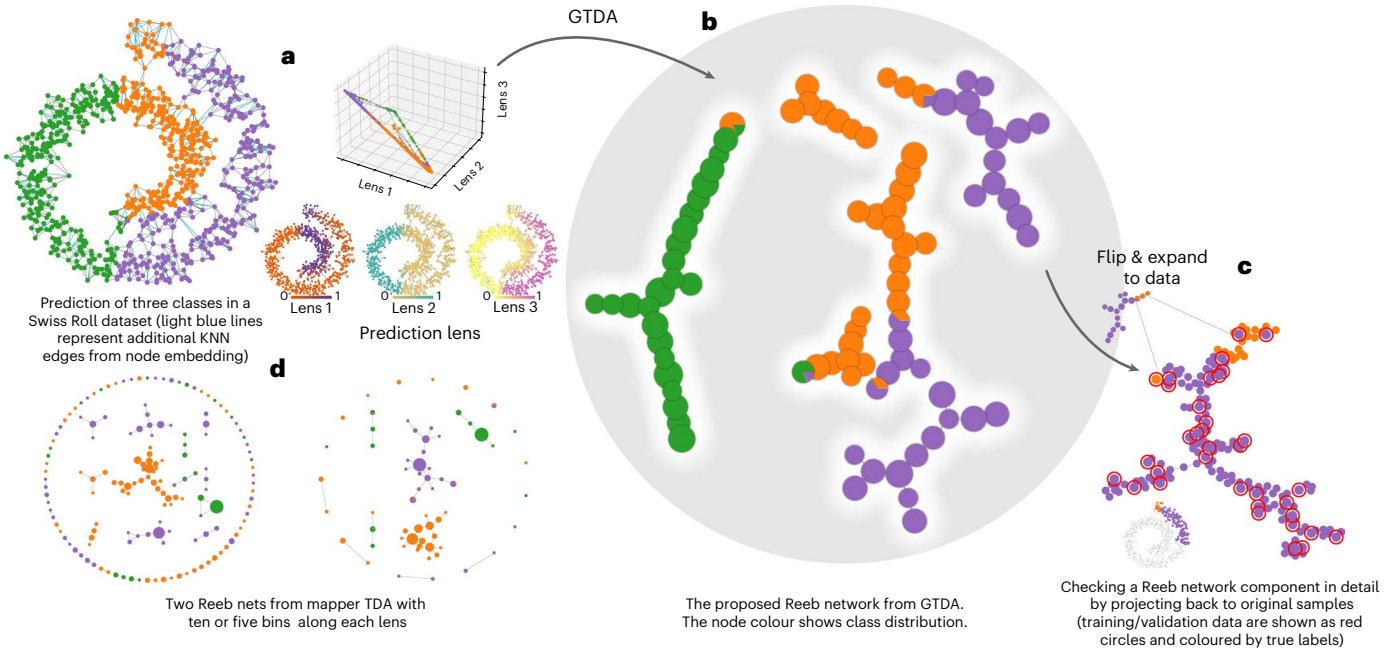
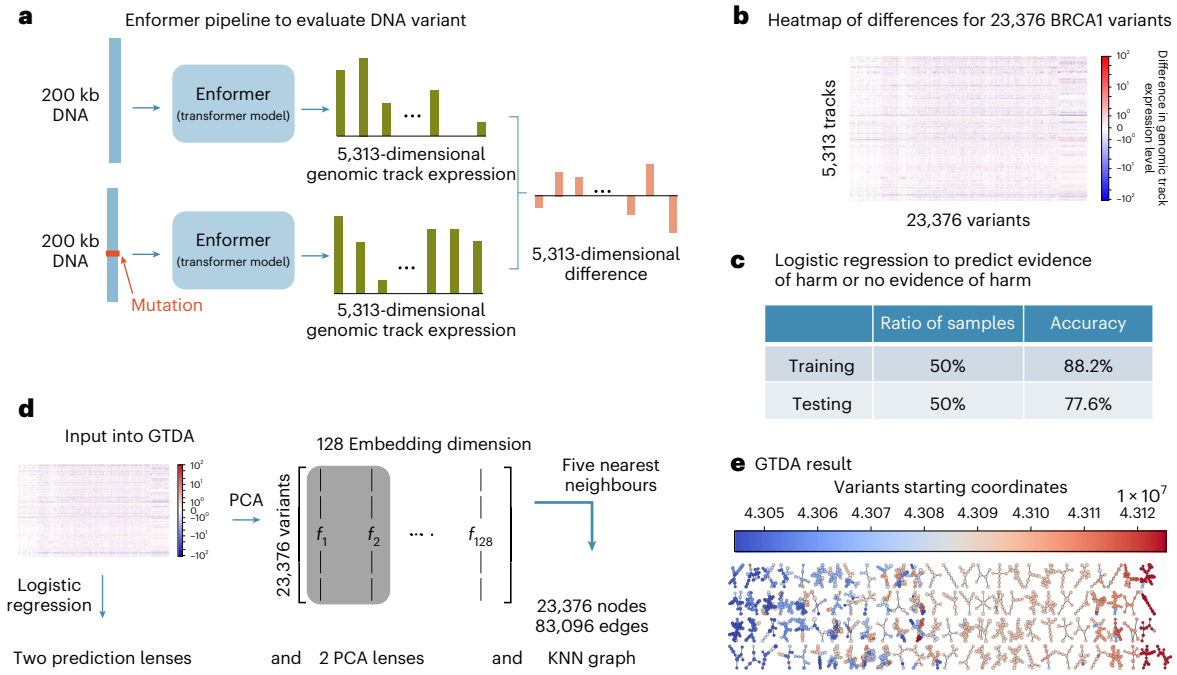


Fig. 2 | Exploring prediction class interfaces with GTDA. a, Consider a prediction scenario with three classes in a Swiss Roll structure and an underlying graph where graph neural network predictions show reasonable accuracy (0.88). The result of the neural network model is a set of three functions over the nodes of the graph that give the probability of prediction for each class, which we call lenses. **b**, The proposed GTDA method produces a simplified topological map of these lenses along with the graph structure, that is, a Reeb network. Each node in the Reeb network maps to a small cluster of similar values to the lens. Nodes are coloured by the fraction of points in each predicted class. The map is disconnected, and each connected piece maps to a limited piece of the original data, simplifying and specifically focusing inspection. **c**, This specificity enables exploration of the interface between the orange and purple class, showing regions where training and validation data points might suggest alternative predictions. **d**, Results from the existing Mapper algorithm for TDA lack this boundary because they contain too many disconnected, isolated pieces.



	Ratio of samples	Accuracy
Training	50%	88.2%
Testing	50%	77.6%

Fig. 3 | To apply GTDA to study the Enformer model, we adapt the pipeline proposed by Avsec and colleagues[1] to use Enformer to study harmful gene variants. a, For each DNA variant of *BRCA1* from ClinVar[34], we run Enformer to generate the difference in expression levels in each of 5,313 genomic tracks. **b**,**c**, These differences are assembled into a 5,313 × 23,376 matrix of data (**b**) that we split into a 50/50 training and testing set for logistic regression against ClinVar's evidence of harm (**c**). **d**, Four lenses are input into GTDA: two prediction probabilities from logistic regression and the two dominant vectors from Principal Component Analysis (PCA), along with a five-nearest-neighbours graph of a 128-dimensional reduction via PCA. **e**, The GTDA result shows 105 individual connected components placed on the basis of the mean of all median DNA variant starting positions for each Reeb net node.

https://doi.org/10.1038/s42256-023-00749-8

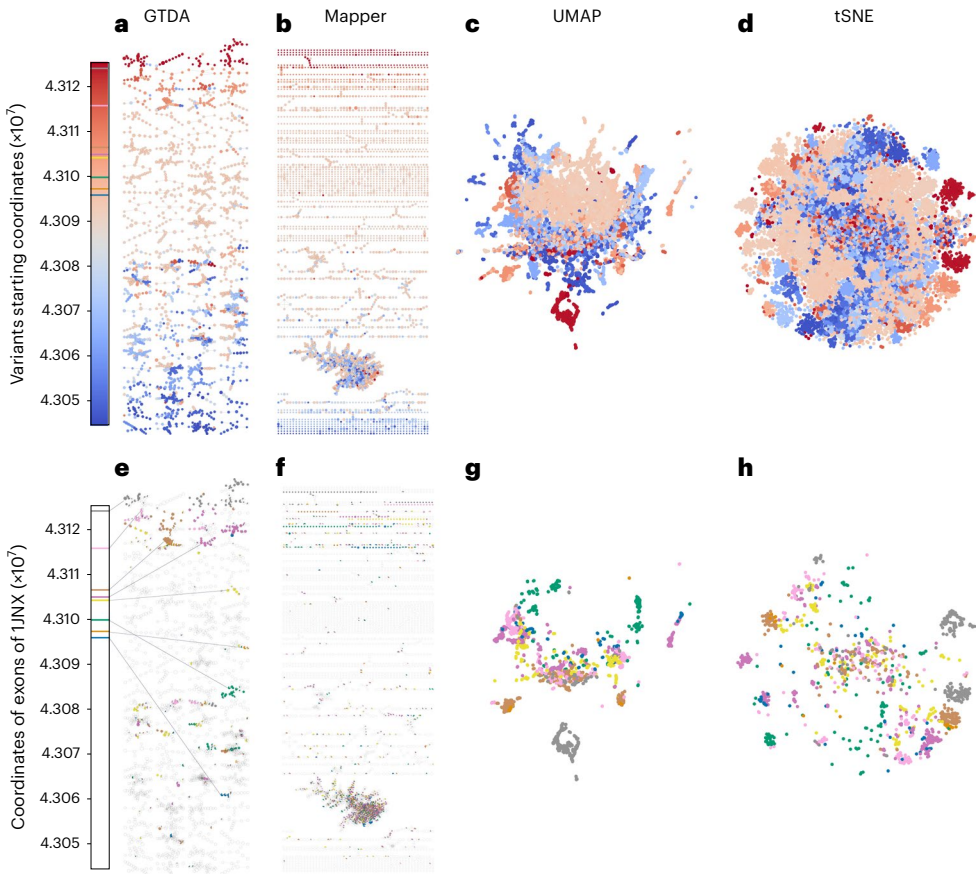


Fig. 4 | Demonstrating biologically relevant features of Enformer's predictions. a, The topological simplification identified by GTDA is highly correlated with DNA variant starting location. **b**–**d**, Alternative global visualizations, such as the simplification from Mapper (**b**)—or dimensionality reduction techniques UMAP (**c**) and tSNE (**d**)—show significantly less sensitivity to the locations of the variants ($P < 0.001$ in a Kolmogorov–Smirnov test; see Supplementary Table 6). **e**, Likewise, the GTDA results strongly localize the exons of the *1JNX* structure within the *BRCA1* gene. **f**–**h**, The results are significantly weaker for Mapper (**f**), UMAP (**g**) and tSNE (**h**) ($P < 0.001$; see Supplementary Table 6). These results demonstrate that the Enformer model is sensitive to these aspects of gene expression and that GTDA makes inspection possible.

animation of the method can be found in Supplementary Video 1. The fundamental idea is that the recursive splitting starts with the set of connected components in the input graph. This is a set of sets: $\mathbb{S}$. The key recursive step is when the method takes a set $\mathbb{S}_i$ from $\mathbb{S}$, it then splits $\mathbb{S}_i$ into new (possibly) overlapping sets $\mathbb{T}_1, \ldots, \mathbb{T}_h$ on the basis of the lens with the maximum difference in values on $\mathbb{S}_i$, and ensures that each $\mathbb{T}_i$ is a single connected component in the graph. Each $\mathbb{T}_i$ is then either added to $\mathbb{S}$ if it is large enough (that is, it has more than K vertices) and where there exists a lens with a maximum difference of larger than d. Otherwise, $\mathbb{T}_i$ goes into the set of finalized sets $\mathbb{F}$.

After the graph has been split into overlapping subgroups and we have the final set of sets, $\mathbb{F}$, the initial Reeb network simplifies each subgroup into a single Reeb network node and connects these simplified nodes if they share any data points (Fig. 1d). This connection strategy may leave Reeb nodes isolated, which is not helpful to understand predictions. We reduce this isolation by adding edges from a minimum spanning tree (Fig. 1e and Supplementary Algorithms 2 and 3) on the basis of the potential for overlap from alternative splits caused by the lenses. We then take two final merging steps, along with building the Reeb net: the first is to merge sets in $\mathbb{F}$ if they are too small (Supplementary Algorithm 2); the second is to add edges to the Reeb net to promote more connectivity (Supplementary Algorithm 3). In total, there are seven user-chosen parameters that control the method and these final merging steps, which are described in Extended Data Table 1.

Computing a Reeb network with GTDA for a complex prediction function or deep-learning method offers a number of opportunities to inspect the predictions (Fig. 2). In this example, GTDA offers more detail at the interface between prediction classes than what is possible with existing methods such as Mapper.

To apply GTDA to prediction analysis, there must be a large set of data points with unknown labels beyond those used for training and validating the prediction model; this is common when gathering data is easy. There must be known relationships among all data points such as: (1) a given graph of relationships among all points (used in Fig. 2a); (2) a nearest-neighbour computation to create such a graph (used when analysing Enformer); or (3) a domain-relevant means of clustering related points. All of our examples use (1) and (2). We also need a real-valued guide to each prediction or predicted class, such as the output from the last layer of a neural network (Fig. 2a). The prediction from this layer provides the lenses. We found it helpful to first smooth the information from the lenses over the relationship graph to avoid sharp gradients using five or ten steps of an iterative smoothing procedure related to a diffusion. Furthermore, there are two main parameters: the maximum size of a Reeb node or cluster, and the amount of overlap in Reeb nodes. The other parameters are less influential (see Supplementary Table 1 for a full list); useful results arise from a wide range of parameters (see Supplementary Section 7 for further discussion of parameter sensitivity).

Constructing a Reeb net with GTDA is a scalable operation. Analysing the Enformer model of gene expression prediction below takes about 30 s, whereas running the Enformer model itself takes hours to generate the necessary data. Analysing 1.3 million images in ImageNet[24]

https://doi.org/10.1038/s42256-023-00749-8

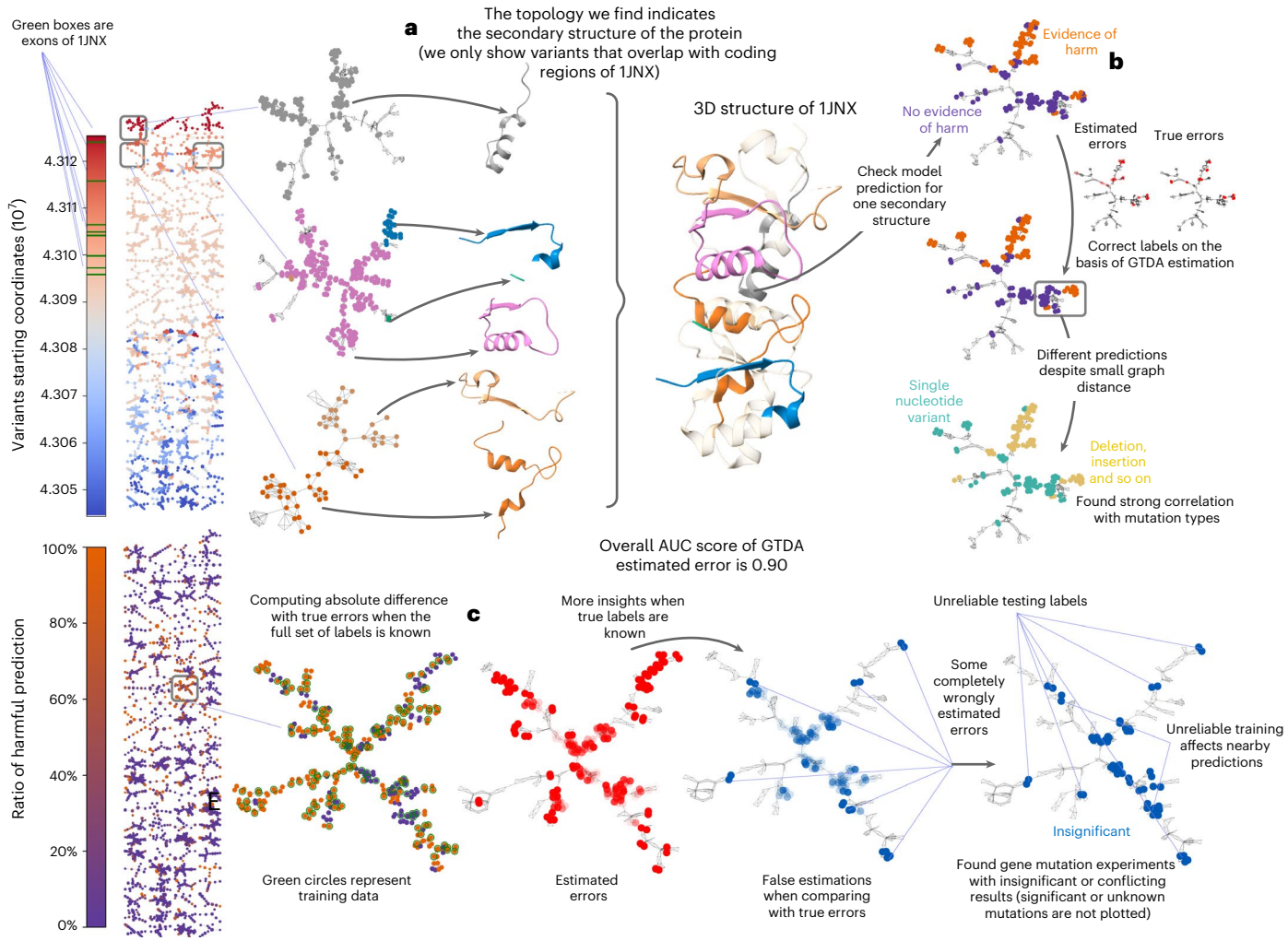


Fig. 5 | We use Reeb networks to visualize harmful (probably pathogenic) and potentially non-harmful (no evidence of pathogenicity) predictions of gene variants in *BRCA1*. **a**, The topology indicates several secondary structures on part of the protein (1JNX). **b**, We further check the model predictions on variants targeting one secondary structure. Our error estimate shows a number of what are probably erroneous predictions, and we flip these expected errors (a final analysis showed that these errors were correctly identified). We continue to see variants with distinct predictions in a small region of a few amino acids. Close examination shows a strong association between mutation types and model predictions where deletion or insertion is more likely to be harmful than a single nucleotide variant. **c**, Further insights when using the full label set show that some estimated errors are completely wrong. These prediction mistakes involve gene mutation experiments with insignificant or conflicting results and indicate underlying uncertainty. These results show how GTDA enables detailed domain-specific inspection of Enformers results (**a**,**b**) and highlights problems with the training and testing data (**c**).

with 2,000 lenses for 1,000 classes in a comparison of ResNet[25] and AlexNet[26] takes 7.24 h (see Extended Data Table 2).

Understanding malignant gene mutation predictions

The Enformer model[1] is a transformer-based model[27] designed to estimate gene expression on the basis of DNA. It works by mapping between the DNA sequence to an estimate of the expression level of this piece of DNA in each of 5,313 genomic tracks. Although Enformer has excellent predictive results, it remains a highly parameterized black box. Our GTDA methodology allows us to assess the topological landscape of the Enformer embeddings when they are used to predict harmful mutations of the *BRCA1* gene in *Homo sapiens* (Fig. 3).

As GTDA results in a simplification of the landscape, this enables us to demonstrate biologically relevant features of Enformer's predictions. In particular, the GTDA map of Enformer shows that many regions of the predictions and embeddings are localized in the DNA sequence (Fig. 4a). Exceptions indicate potential long-distance interactions that Enformer uses to enhance its predictions. By contrast, neither the standard Mapper algorithm for TDA (Fig. 4b) nor the tSNE or UMAP embeddings (Fig. 4c,d) of the same points show nearly the same degree of location sensitivity. In another demonstration of how the GTDA framework highlights the known biology of DNA, we examine where mutations in the exons of the 1JNX region of *BRCA1* are present in the final maps. Again, we see strong localization among the exons and the GTDA map (Fig. 4e). The results are again much weaker for Mapper, tSNE and UMAP (Fig. 4f–h).

If we study the 1JNX repeat region within *BRCA1* and its associated 3D structure, then key pieces of the secondary structure of 1JNX are likewise localized in the Reeb components identified by GTDA (Fig. 5a). This greatly aids interpretation of the results. For one of the helix structures, this analysis reveals regions where insertions and deletions are harmful (pathogenic) and where single nucleotide variants lack evidence of harm (Fig. 5b).

Another tool that our framework provides is automatic error estimation. A similar tool is uncertainty in neural network predictions, which highlights places with less confident predictions. Automatic error estimation in GTDA applies a simple network diffusion analysis

https://doi.org/10.1038/s42256-023-00749-8

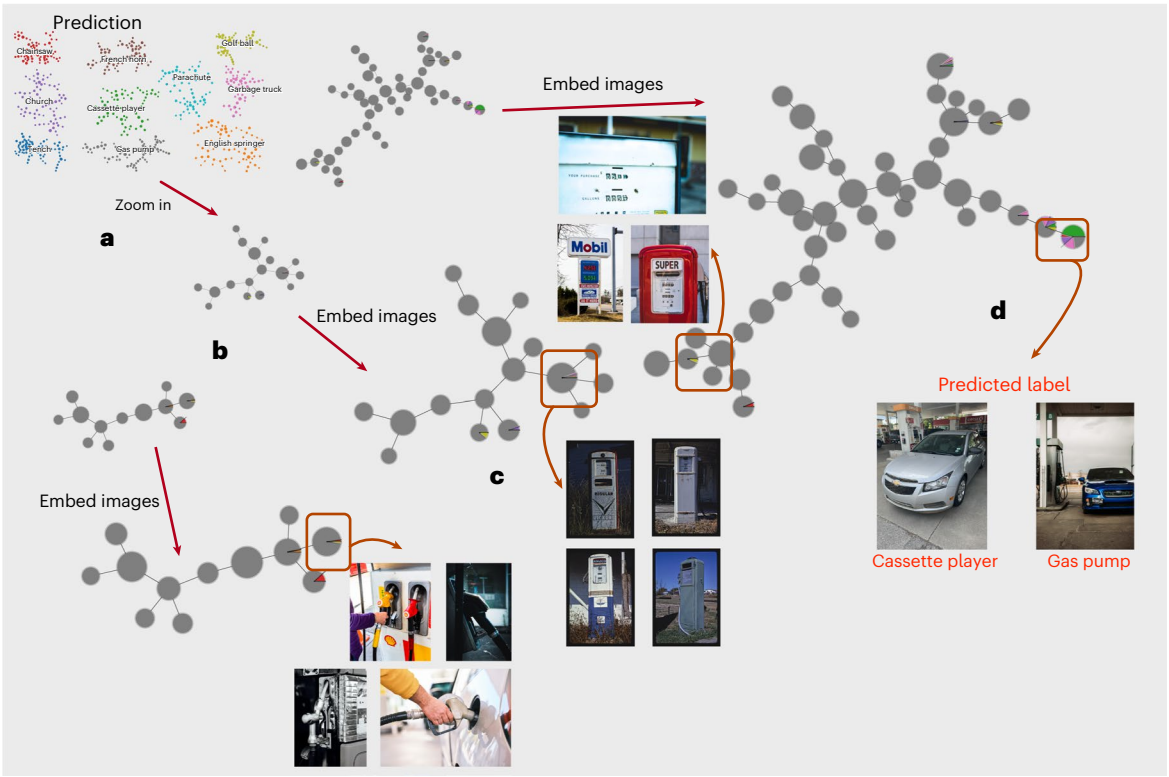


Fig. 6 | Analyzing ResNet50 predictions on Imagenette. a, We take a pretrained ResNet50 model and retrain the last layer to predict ten classes in Imagenette. **b**,**c**, We zoom into the Reeb network group of 'gas pump' predictions (**b**) and display the images of different local regions (**c**), showing gas pump images with distinct visual features (the sparsity in coverage is due to the public domain images we show here). Examining these subgroups can provide a general idea on how the model will behave when predicting future images with similar features, as well as help us quickly identify potential labelling issues in the dataset. **d**, For instance, we find a group of images whose predicted labels are 'cassette player' even though they are actually images of 'cars' (or even a car at a gas pump); this arises due to errors in the original training data, where images of cars are labelled 'cassette player'. Credit: the four gas pumps at lower centre are from the Library of Congress, Prints and Photographs Division, photograph by John Margolies.

to the original data graph, but restricted to edges that are identified as important to the topological simplification. Full details of the procedure are given in Supplementary Section 1.5. This error estimation greatly outperforms model uncertainty for this study (area under the curve (AUC) of 0.9 from GTDA compared with an AUC of 0.66 for uncertainty; Extended Data Fig. 2). In binary classification problems, we can automatically correct mistakes if the probability of error from our estimate is higher than model confidence in the solution.

In comparing our error estimate to the underlying annotations on harmful DNA variants, we discovered a Reeb component with many harmful predictions (Fig. 5c). This component had many mutations where the framework incorrectly predicts errors after comparing with known labels. Detailed analysis showed that these errors are strongly associated with insignificant results in the underlying data that should not have been used as training or testing data (Extended Data Fig. 3 and Supplementary Section 5.5).

Additional findings

When the framework is applied to a pretrained ResNet50 model[25] on the Imagenette dataset[28], it produces a visual taxonomy of images suggesting what ResNet50 is using to categorize the images (Supplementary Section 3). This example also highlights a region where the ground-truth labels of the data points are incorrect and cars are erroneously labelled 'cassette player' (Fig. 6). To make these visualize taxonomies easier to explore, we design a diagram that places images directly on the layout of the Reeb network (Extended Data Fig. 4). We were unable to determine how to use traditional TDA results to identify this set of erroneous examples (Extended Data Fig. 5), although we reliably do so with GTDA (Supplementary Section 3.3).

When we apply the GTDA framework to a graph neural network that predicts the type of product on Amazon on the basis of reviews, the framework identifies an ambiguity in product categories that limits prediction accuracy (Extended Data Fig. 6 and Supplementary Section 2).

We compare the embeddings from tSNE, UMAP and GTDA for the four datasets (the simple Swiss Roll from Fig. 2, the product type Amazon data, ResNet50 on Imagenette and the malignant gene mutation data) in Extended Data Fig. 7.

Two further studies include an investigation of chest X-ray diagnostics and a comparison of deep-learning frameworks. In the investigation of chest X-rays, we show how the Reeb networks find incorrect diagnostic annotations in chest X-ray datasets used for deep learning[29] (Extended Data Fig. 8). The GTDA methods give an AUC of 0.75 (Supplementary Section 6). The comparison of deep learning frameworks is designed to test how well GTDA scales to larger problems with over a million points and 2,000 lenses. In this case, we use GTDA to analyse pairwise differences among ResNet[25], AlexNet[26] and VOLO-50[30]. Each lens consists of one set of predictions from each method. These results show that GTDA scales to large problems and does not take much more time than inference (Extended Data Table 2 and Supplementary Section 4).

Discussion

Our Reeb network construction extends recent analytical methods from topology[15,17] to facilitate applications to the topology of complex prediction. In comparison with other proposed applications of topology to neural networks, GTDA focuses on simplifying the topology of the combined prediction and embedding space to aid in inspection of the prediction methods. The ideas underlying GTDA are loosely related to how Naitzat et al.[18] study topological changes as data are

passed through the layers of a neural network, whereas we focus only on the final embedding space. Graph-based topological data analysis (GTDA) differs from methods such as TopoAct[19], which studies the shape of activation space at a given layer of a neural network to provide insights on the learned representations, as well as those of Gabrielsson et al.[20], who correlate topology in the weights with generalization performance. It also differs from methods that directly try to embed topology in training[21,31]. It is similar in spirit to methods that combine group-invariance and topological understanding of data with neural networks[22], albeit with key differences regarding how the topological information is used. Further combinations of these ideas offer considerable potential for use of topology in complex prediction methods.

Our work relates to interpretability[32] and seeks to produce a comprehensible map of the prediction structure to aid navigation of a large space of prediction to those most interesting areas. In this taxonomy of interpretability, our methods are most useful for global, dataset-level post-hoc interpretability. They are relevant because they provide insights into the model's behaviour in terms of the underlying domains (gene expression in the main text, and images and graph problems in the supplementary case studies). In terms of relevance to real-world problems, our methods highlight both problematic data points in the training and validation sets. These results also highlight weaknesses of dimension-reduction methods for similar uses. Beyond identifying that there is a problem, the insights from the topology suggest relationships to nearby data and thereby suggest mechanisms that could be addressed through future improvements.

Considering the ability of these topological inspection techniques to translate prediction models into actionable human-level insights, we expect them to be applicable to new models and predictions, broadly, as they are created and to be a critical early diagnostic of prediction models. The interaction of topology and prediction may provide a fertile ground for future improvements in prediction methods.

Methods

See the Supplementary Information for full details on the methods.

Data availability

A description of how to access all of the datasets we use is available at https://github.com/MengLiuPurdue/Graph-Topological-Data-Analysis, along with all supporting code for the results in this paper. For each experiment in this paper, this repository includes precomputed inputs for the graph input to GTDA, and the prediction lenses; these can be found in the 'datasets/precomputed' folder, and are the data required to validate our GTDA implementation. For the ImageNet-1k experiment, the files are available from https://doi.org/10.5281/zenodo.10052250. Beyond these required data, the repository contains code to translate the following publicly available datasets into these precomputed results: https://ftp.ncbi.nlm.nih.gov/pub/clinvar/tab_delimited/ (variant_summary.txt); http://hgdownload.soe.ucsc.edu/downloads.html#human (hg19.fa and hg38.fa); https://github.com/deepmind/deepmind-research/tree/master/enformer (for the pretrained Enformer model); http://jmcauley.ucsd.edu/data/amazon/index_2014.html (download 'All products' under 'Electronics' from the 2014 version of the Amazon reviews data; https://github.com/fastai/imagenette; https://www.image-net.org/ (all of the ImageNet-1k training and validation images); the timm package for the pretrained VOLO model; https://cloud.google.com/healthcare-api/docs/resources/public-datasets/nih-chest; and https://github.com/zoogzog/chexnet (the pretrained chexnet model). Please contact the authors for additional intermediate data if that might be useful.

Code availability

The implementation of GTDA framework we developed is available at ref. 33, along with all of the supporting code for the results in this paper. Demos can be found at https://mengliupurdue.github.io/Graph-Topological-Data-Analysis/, which show sample Reeb networks that are based on these experiments.

References

1. Avsec, Ž. et al. Effective gene expression prediction from sequence by integrating long-range interactions. *Nat. Methods* **18**, 1196–1203 (2021).
2. Esteva, A. et al. Dermatologist-level classification of skin cancer with deep neural networks. *Nature* **542**, 115–118 (2017).
3. Reichstein, M. et al. Deep learning and process understanding for data-driven earth system science. *Nature* **566**, 195–204 (2019).
4. Townshend, R. J. L. et al. Geometric deep learning of RNA structure. *Science* **373**, 1047–1051 (2021).
5. Zech, J. R. et al. Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs: a cross-sectional study. *PLOS Medicine* **15**, e1002683 (2018).
6. Oakden-Rayner, L. et al. Validation and algorithmic audit of a deep learning system for the detection of proximal femoral fractures in patients in the emergency department: a diagnostic accuracy study. *Lancet Digit. Health* **4**, e351–e358 (2022).
7. Zhou, B., Khosla, A., Lapedriza, A., Oliva, A. & Torralba, A. Learning deep features for discriminative localization. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition* 2921–2929 (IEEE, 2016).
8. Selvaraju, R. R. et al. Grad-CAM: visual explanations from deep networks via gradient-based localization. In *Proc. IEEE International Conference on Computer Vision* 618–626 (IEEE, 2017).
9. Saporta, A. et al. Benchmarking saliency methods for chest X-ray interpretation. *Nat. Mach. Intell.* **4**, 867–878 (2022).
10. Koh, P. W. & Liang, P. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning* 1885–1894 (PMLR, 2017).
11. van der Maaten, L. & Hinton, G. Visualizing data using t-SNE. *J. Mach. Learning Res.* **9**, 2579–2605 (2008).
12. Becht, E. et al. Dimensionality reduction for visualizing single-cell data using UMAP. *Nat. Biotechnol.* **37**, 38–44 (2018).
13. McInnes, L., Healy, J., Saul, N. & Großberger, L. UMAP: Uniform Manifold Approximation and Projection. *J. Open Source Softw.* **3**, 861 (2018).
14. Dey, T. K. & Wang, Y. *Computational Topology for Data Analysis* (Cambridge Univ. Press, 2022); https://www.cs.purdue.edu/homes/tamaldey/book/CTDAbook/CTDAbook.pdf
15. Singh, G., Mémoli, F. & Carlsson, G. E. Topological methods for the analysis of high dimensional data sets and 3D object recognition. In *Eurographics Symposium on Point-Based Graphics* (Botsch, M. & Pajarola, R.) Vol. 91, 100 (The Eurographics Association, 2007).
16. Nicolau, M., Levine, A. J. & Carlsson, G. Topology based data analysis identifies a subgroup of breast cancers with a unique mutational profile and excellent survival. *Proc. Natl. Acad. Sci.* **108**, 7265–7270 (2011).
17. Lum, P. Y. et al. Extracting insights from the shape of complex data using topology. *Sci. Rep.* **3**, 1236 (2013).
18. Naitzat, G., Zhitnikov, A. & Lim, L.-H. Topology of deep neural networks. *J. Mach. Learn. Res.* **21**, 184:1–184:40 (2020).
19. Rathore, A., Chalapathi, N., Palande, S. & Wang, B. Topoact: visually exploring the shape of activations in deep learning. In *Computer Graphics Forum* Vol. 40, 382–397 (Wiley, 2021).
20. Gabrielsson, R. B. & Carlsson, G. Exposition and interpretation of the topology of neural networks. In *2019 18th IEEE International Conference on Machine Learning and Applications (ICMLA)* 1069–1076 (IEEE, 2019).

21. Hajij, M., Zamzmi, G. & Batayneh, F. TDA-Net: fusion of persistent homology and deep learning features for COVID-19 detection from chest X-ray images. In *2021 43rd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)* 4115–4119 (IEEE, 2021).
22. Bergomi, M. G., Frosini, P., Giorgi, D. & Quercioli, N. Towards a topological–geometrical theory of group equivariant non-expansive operators for data analysis and machine learning. *Nat. Mach. Intell.* **1**, 423–433 (2019).
23. Dey, T. K, Mémoli, F. & Wang, Y. Multiscale mapper: topological summarization via codomain covers. In *Proc. 27th Annual ACM-SIAM Symposium on Discrete Algorithms* 997–1013 (SIAM, 2016).
24. Russakovsky, O. et al. ImageNet large scale visual recognition challenge. *Int. J. Comput. Vision* **115**, 211–252 (2015).
25. He, K., Zhang, X., Ren, S. & Sun, J. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE, 2016).
26. Krizhevsky, A., Sutskever, I. & Hinton, G. E. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems* Vol. 25 (NeurIPS, 2012).
27. Vaswani, A. et al. Attention is all you need. In *Advances in Neural Information Processing Systems* Vol. 30 (NeurIPS, 2017).
28. Howard, J. *Imagenette Dataset* (GitHub, 2021); https://github.com/fastai/imagenette
29. Wang, X. et al. ChestX-ray8: hospital-scale chest X-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition* 2097–2106 (IEEE, 2017).
30. Yuan, L., Hou, Q., Jiang, Z., Feng, J. & Yan, S. VOLO: vision outlooker for visual recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **45**, 6575–6586 (2023).
31. Love, E. R., Filippenko, B., Maroulas, V. & Carlsson, G. Topological deep learning. Preprint at https://arxiv.org/abs/2101.05778 (2021).
32. Murdoch, W. J., Singh, C., Kumbier, K., Abbasi-Asl, R. & Yu, B. Definitions, methods, and applications in interpretable machine learning. *Proc. Natl Acad. Sci. USA* **116**, 22071–22080 (2019).
33. Liu, M. *Graph Topological Data Analysis: v0.1 (GTDA)* (Zenodo, 2023); https://doi.org/10.5281/zenodo.8268055
34. Landrum, M. J. et al. ClinVar: improving access to variant interpretations and supporting evidence. *Nucleic Acids Res.* **46**, D1062–D1067 (2018).
35. Strodthoff, B. & Jüttler, B. Layered Reeb graphs for three-dimensional manifolds in boundary representation. *Comput. Graphics* **46**, 186–197 (2015).

Acknowledgements

We are grateful to A. Benson for feedback. We acknowledge funding via grant nos. NSF CCF-1909528, NSF IIS-2007481 and DOE DE-SC0023162 (to D.F.G), and grant nos. NSF CCF-2049010 and NSF DMS-2301360 (to T.K.D).

Author contributions

M.L. wrote the initial draft, designed and developed the GTDA algorithm, designed and performed the computational experiments, and analysed results. T.K.D. helped design the TDA part of the GTDA algorithm, wrote the description of topological methods, and analysed results. D.F.G. contributed to the writing throughout, helped design the GTDA algorithm, helped design computational experiments, overall project planning and analysed the results.

Competing interests

The authors declare no competing interests.

Additional information

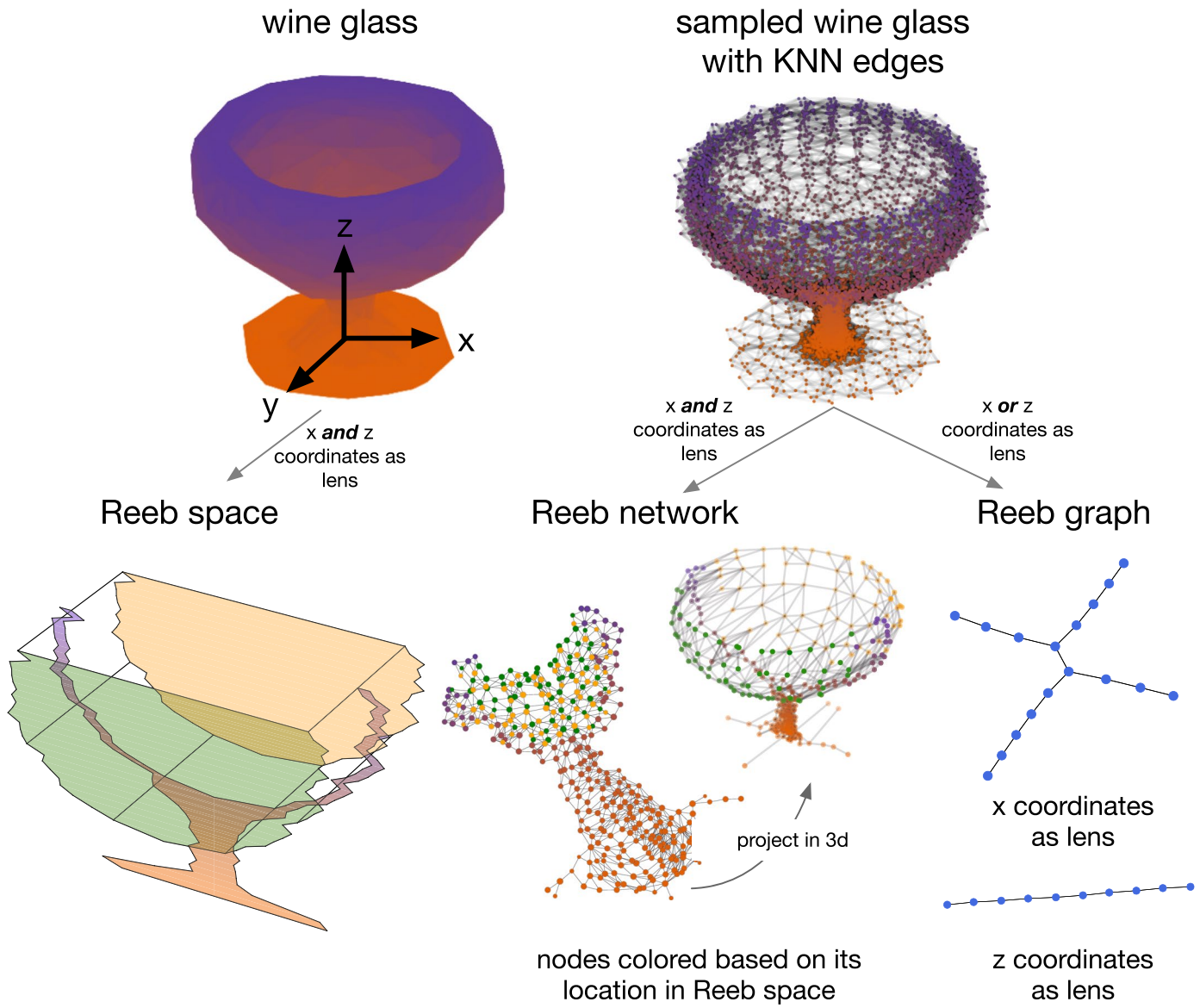
Extended data is available for this paper at https://doi.org/10.1038/s42256-023-00749-8.

Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s42256-023-00749-8.

Correspondence and requests for materials should be addressed to Meng Liu or David F. Gleich.

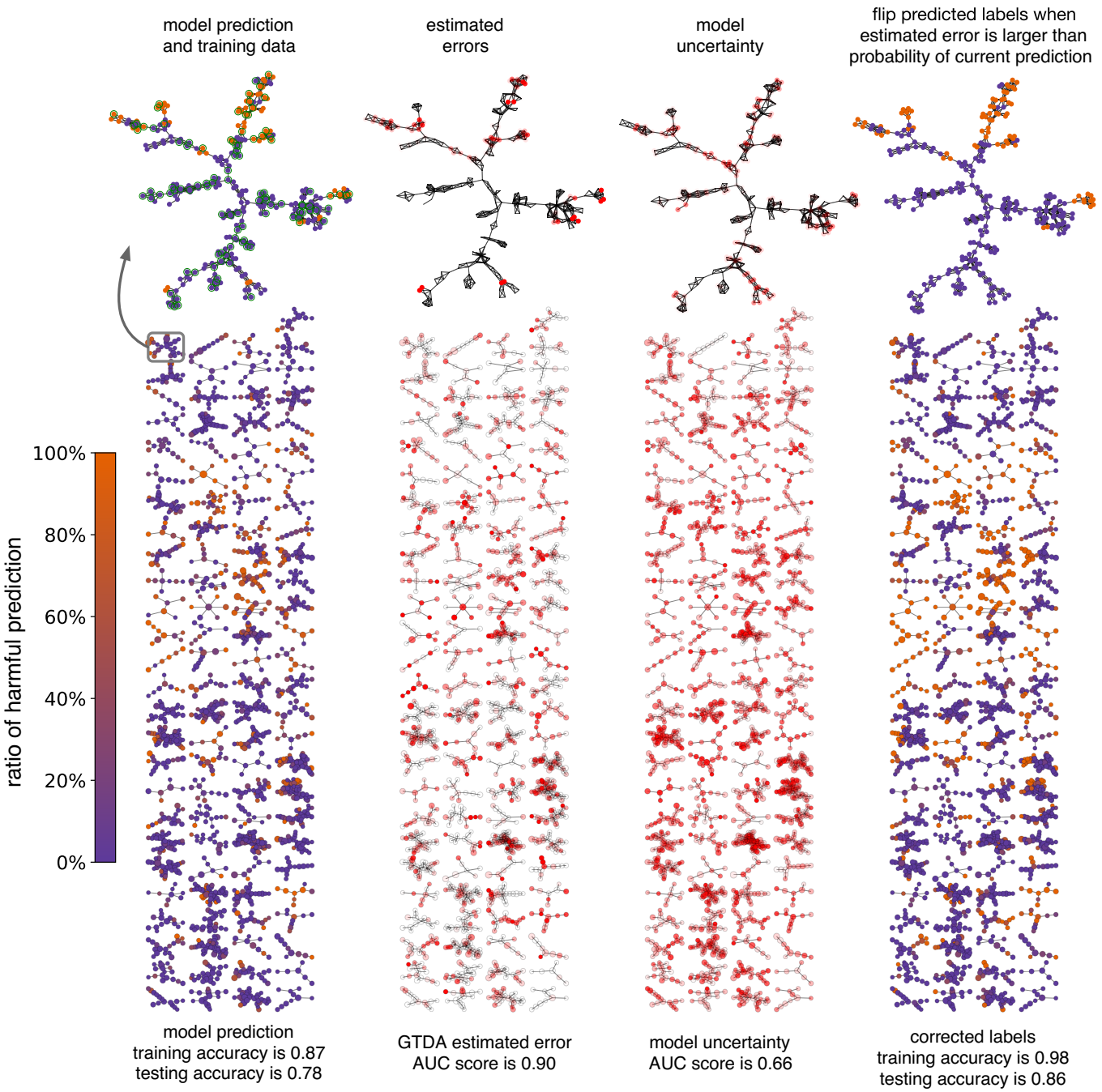
Peer review information *Nature Machine Intelligence* thanks the anonymous reviewers for their contribution to the peer review of this work. Primary Handling Editor: Liesbeth Venema, in collaboration with the *Nature Machine Intelligence* team.

https://doi.org/10.1038/s42256-023-00749-8



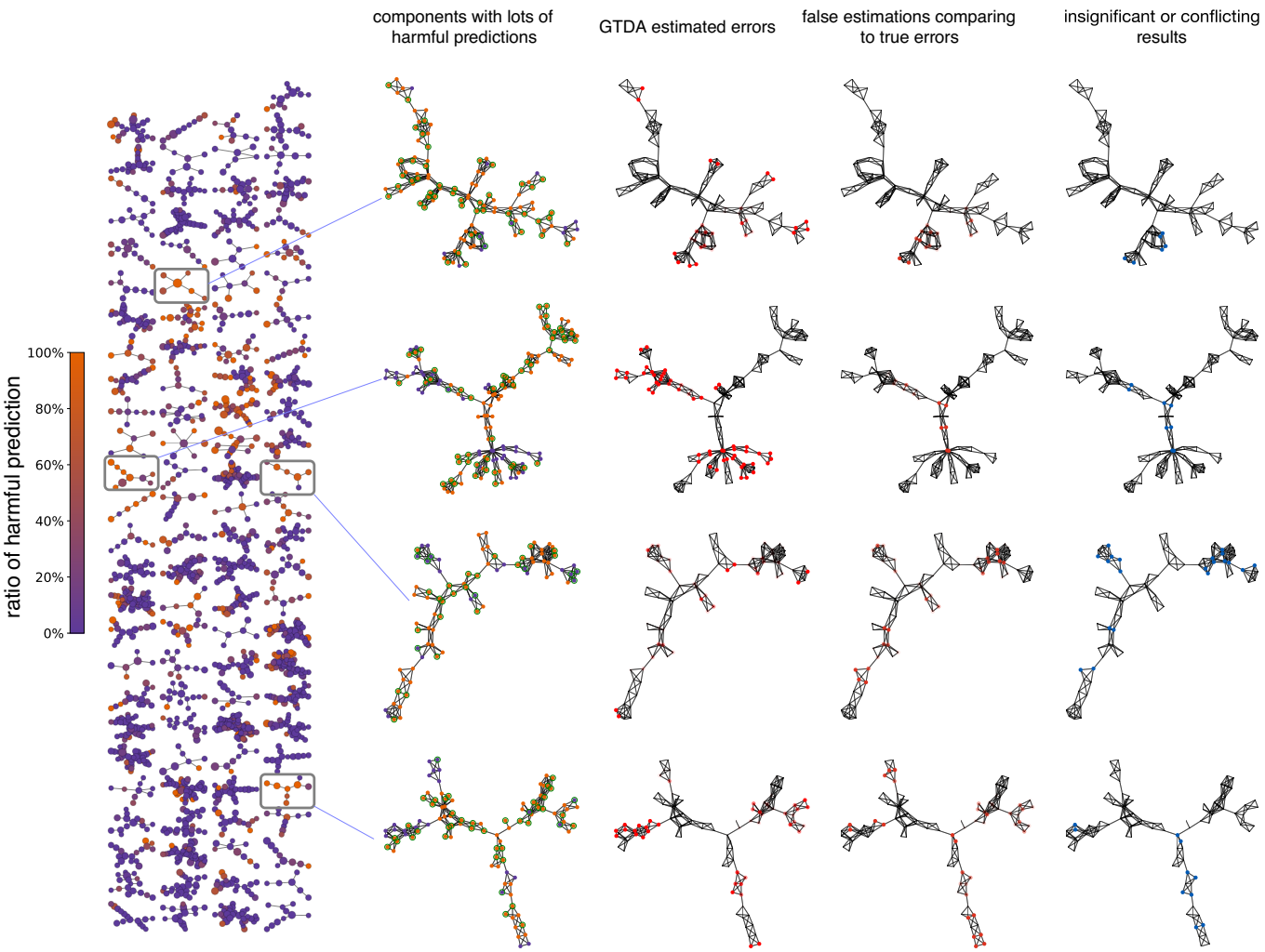
Extended Data Fig. 1 | Examples of Reeb graphs, Reeb spaces, and Reeb networks. The term network or net is often used to mean a graph abstraction of a complex system. A Reeb graph (right) is a topological structure that gives univariate topological information and produces a graph. A Reeb space (left) is a more complicated multidimensional structure. A Reeb network (middle) is an undirected graph like a Reeb graph, which embodies the multidimensional topological information of a Reeb space. This figure illustrates the difference between a Reeb graph and a Reeb network on a topologically interesting object. The lenses we use here are the *x* and *z* coordinates. The inspiration for the object is ref. 35. See Supplemental Section 1.6 for additional discussion.

https://doi.org/10.1038/s42256-023-00749-8



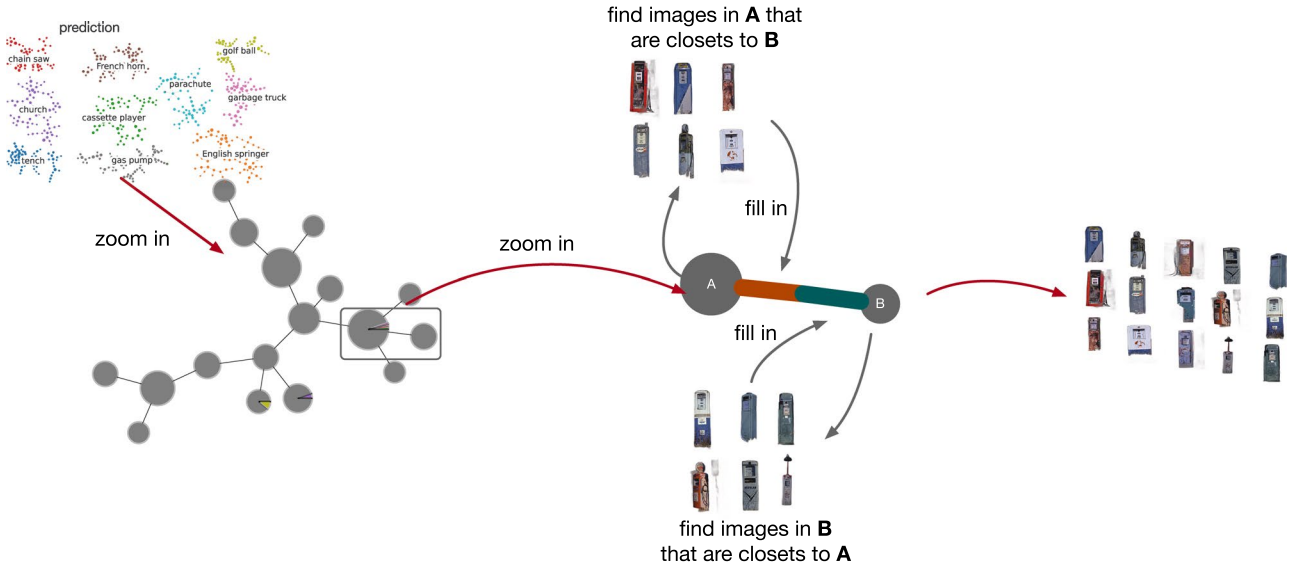
Extended Data Fig. 2 | GTDA Automated error estimation for Enformer. The GTDA automated error estimation and correction applied on the Enformer data shows the ability to considerably increase training and test accuracy. In the four panels we show the prediction (left), GTDA estimated errors (middle left), model uncertainty (middle right), and corrected labels (right). In the top part, we zoom into a single component and mark the training data using green circles. Because this is a binary prediction problem, if the estimated if the estimated error is larger than the prediction probability, we can automatically correct the error. In the lower part, we report AUC score for error detection and both training and testing accuracy for the original and corrected predictions.

https://doi.org/10.1038/s42256-023-00749-8



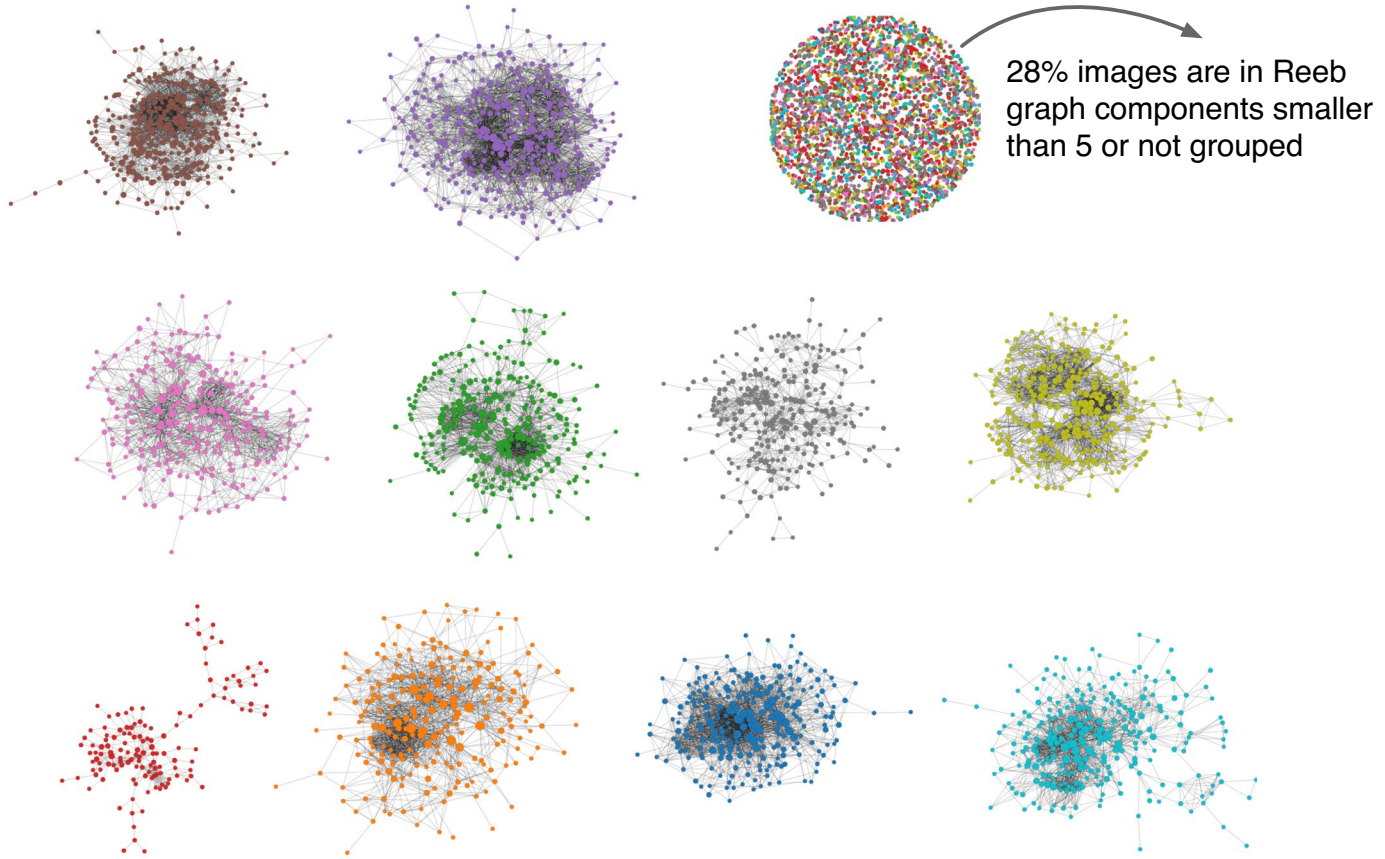
Extended Data Fig. 3 | A study of unreliable labels and their impact on Enformer predictions. A deeper investigation of the false positive results from our error estimation on the Enformer data shows a strong correlation with insignificant or conflicting results in the original data. This is illustrated for four components of the Reeb net from GTDA with many mutations predicted to be harmful. The middle column shows the GTDA estimated errors and the middle right column shows the false positives among those. The final column shows data points labelled with unreliable labels due to insignificant or conflicting results.

https://doi.org/10.1038/s42256-023-00749-8



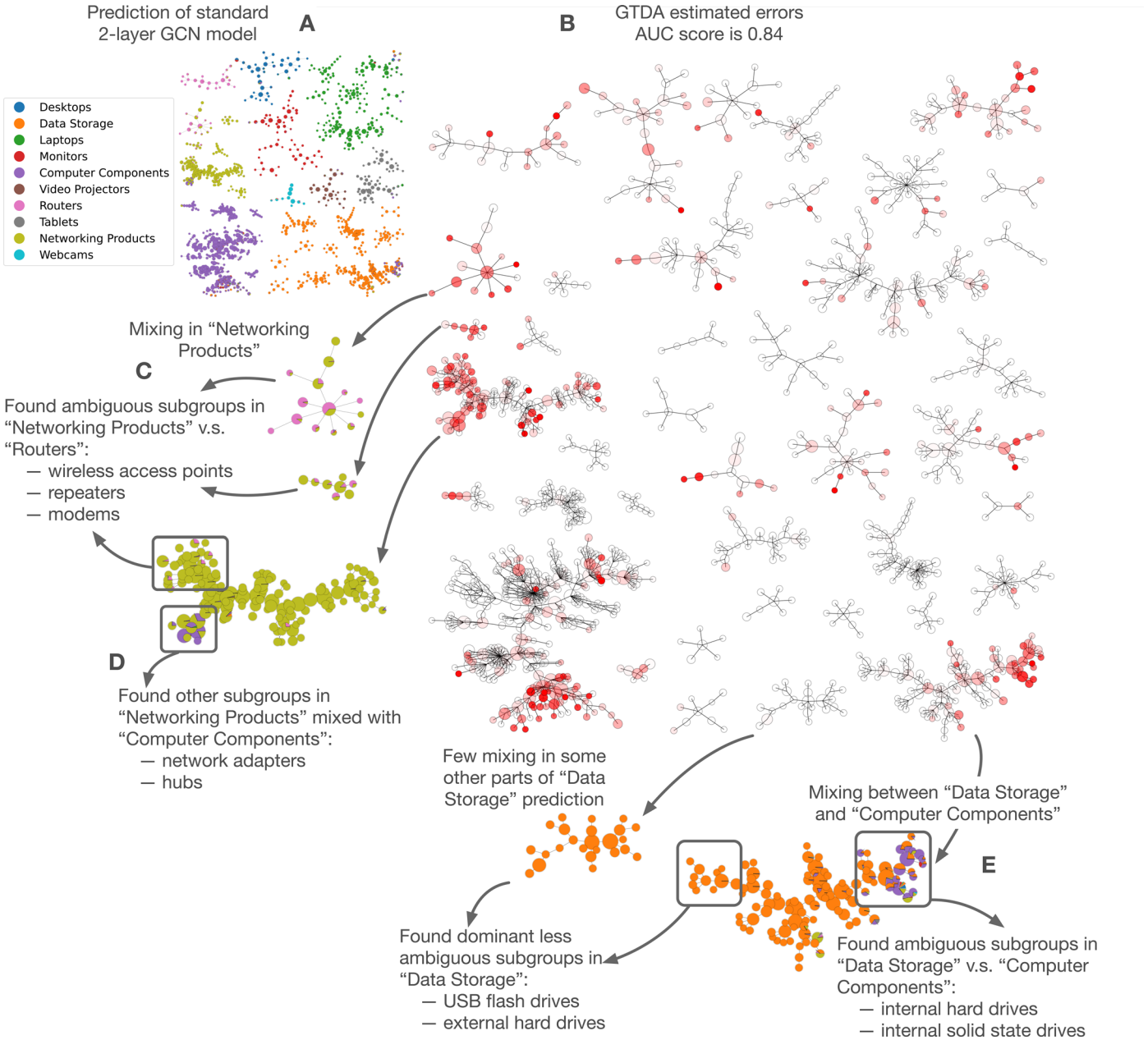
Extended Data Fig. 4 | A visualization procedure to show images along with Reeb network structure. This figure demonstrates the procedure of embedding images on a Reeb net component. For each pair of adjacent nodes, we select images from one end that are closest to the other end and fill in those images in half of the edge and vice versa. Browsing around embedded images at different regions can help us quickly identify problematic labels such as 'cassette player' images that are just 'cars'. Using full ImageNet data, the graphs are dense and full of pictures.

https://doi.org/10.1038/s42256-023-00749-8



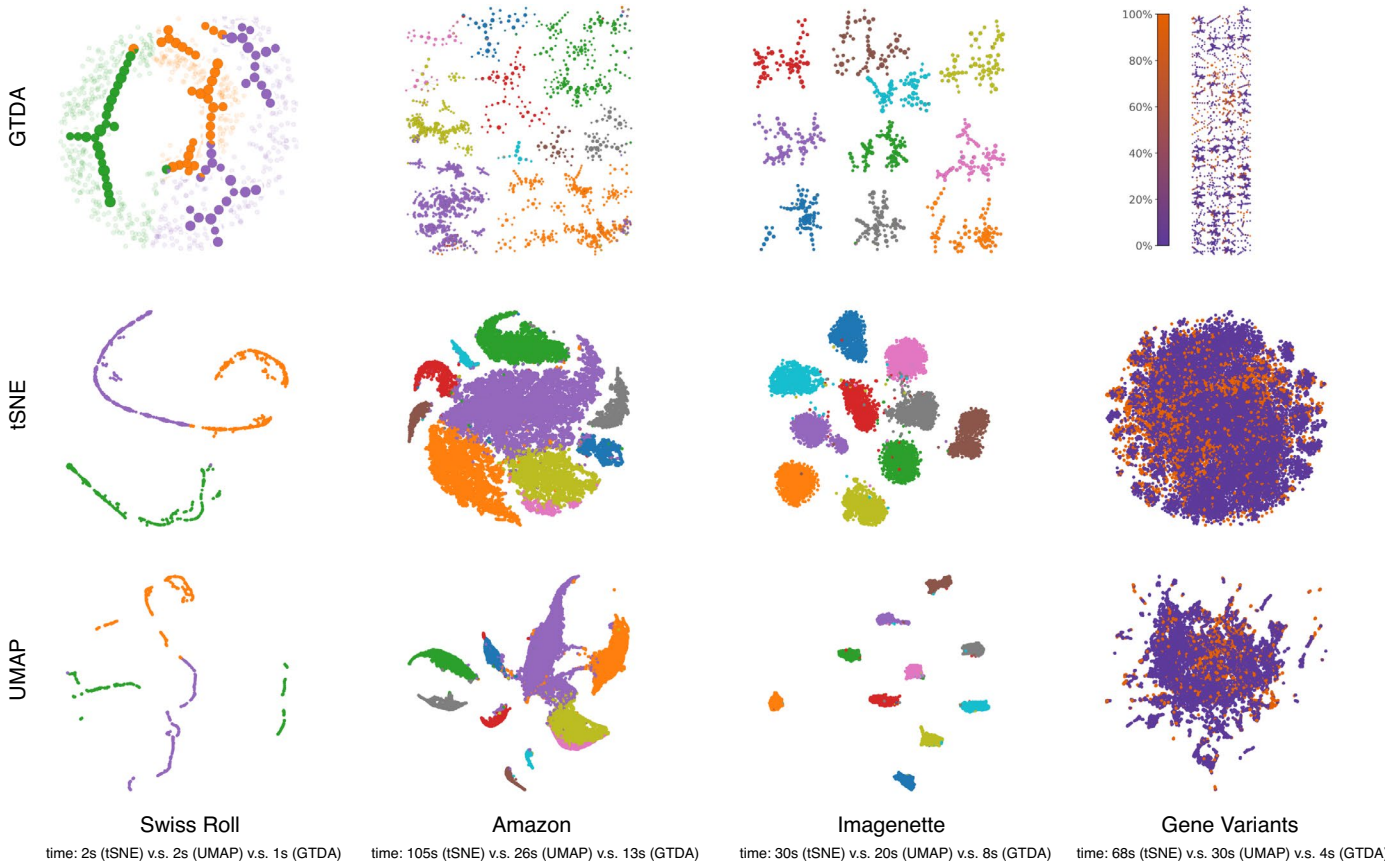
Extended Data Fig. 5 | Reeb networks using the existing Mapper algorithm compared with those from our GTDA. The Reeb net on the 10 classes of Imagenette created by the original Mapper TDA framework. In this case, TDA is directly applied to the ResNet image embedding matrix without transforming into KNN graph. Unlike the GTDA visualization, there are no obvious subgroups other than 10 major components representing 10 classes. This leaves no straightforward way to identify the incorrect class of 'cassette player' images that GTDA found. Moreover, no information can be extracted at all for around 28% images as they are either in some very small Reeb net components or simply considered as noise by the clustering scheme.

https://doi.org/10.1038/s42256-023-00749-8



Extended Data Fig. 6 | GTDA for a graph convolutional network. a,**b**, The GTDA-produced Reeb network of a standard 2-layer graph convolutional network model trained and validated on 10% labels of an Amazon co-purchase dataset (**a**) and estimated errors shown in red (**b**). The map highlights ambiguity between 'Networking Products' and 'Routers'. **c**, Checking these products shows wireless access points, repeaters or modems as likely ambiguities. **d**, Additional label ambiguities involve 'Networking Products' and 'Computer Components' regarding network adapters. **e**, Likewise 'Data Storage' and 'Computer Components' are ambiguous for internal hard drives. These findings suggest that the prediction quality is limited by arbitrary subgroups in the data, which Reeb networks helped locate quickly.

https://doi.org/10.1038/s42256-023-00749-8



Extended Data Fig. 7 | GTDA compared with tSNE and UMAP. Comparing the results of the dimension reduction techniques tSNE and UMAP on 4 datasets to the topological Reeb net structure from GTDA shows similarities and differences among summary pictures generated by these methods. The graph created by GTDA permits many types of analysis not clearly possible with tSNE and UMAP output. For running time comparison, since we also need to extract model embeddings and predictions just like GTDA, we exclude such time and only report the time of the actual execution of tSNE or UMAP or GTDA (including Kamada-Kawai).

https://doi.org/10.1038/s42256-023-00749-8

Extended Data Fig. 8 | GTDA on chest X-ray images. In an analysis of deep learning methods for chest X-ray analysis, we demonstrate how to use the GTDA results to find which testing labels are likely to be problematic. **a**, We first zoom in a component found by GTDA and use green circles to mark testing images where we have expert labels. **b**, Then we use GTDA to estimate prediction errors on circled images. **c**, Comparing GTDA estimation with original labels in the testing data highlights a few places where GTDA incorrectly estimates errors. **d**, We investigate the hypothesis that these false estimations are due to problematic testing labels and do a simple thresholding of 0.5, which flags 17 problematic testing labels in this component. Comparing to re-evaluated expert labels can find 14 true positives with a precision of 0.82 and a recall of 0.78 (E).

https://doi.org/10.1038/s42256-023-00749-8

Extended Data Table 1 | A list of parameters used in GTDA

parameter	description	suggested choices
K	component size threshold to stop splitting	5% of smallest class size
d	lens difference threshold to stop splitting	0 or 0.001
r	overlapping ratio	0.01
s_1	Reeb node size threshold	5
s_2	Reeb component size threshold	5
α	lens smoothing parameter	0.5 (used in all experiments)
S	lens smoothing steps	5 or 10
f	distance function in the merging step	ℓ^∞ difference of row i, j of $P^{(S)}$

https://doi.org/10.1038/s42256-023-00749-8

Extended Data Table 2 | Statistics on experiments and running times

dataset	nodes	edges	classes	lens	predicting & embedding (s)	preprocessing (s)	GTDA time (s)
Swiss Roll	1,000	3,501	3	3	0.003	0.3	1
Amazon Computers	39,747	399,410	10	10	0.17	7	10
Subset of ImageNet	13,394	51,520	10	10	27	5	7
ImageNet-1k (ResNet vs AlexNet)	1,331,167	5,954,900	1,000	2,000	2,379	717	26,036
ImageNet-1k (VOLO vs ResNet)	1,331,167	5,805,714	1,000	2,000	13,426	617	18,894
BRCA1 Gene Variants	23,376	83,096	2	4	18,583	21	3
Chest X-rays	112,120	431,893	2	16	821	35	26

Predicting and embedding represents the time used to generate prediction and extract embedding for all samples from a trained model. Preprocessing time includes PCA, normalization as well as building a KNN graph if the original dataset is not in graph format. GTDA time is the time to compute Reeb network given the input graph and the lens.