Ensemble Learning for Mapper Parameter Optimization

Padraig Fitzpatrick*, Anna Jurek-Loughrey*, Paweł Dłotko†, and Jesus Martinez del Rincon*
*School of Electronics, Electrical Engineering and Computer Science, Queen's University Belfast, Belfast, United Kingdom
{pfitzpatrick12, a.jurek, j.martinez-del-rincon}@qub.ac.uk
†Dioscuri Centre in Topological Data Analysis, Mathematical Institute, Polish Academy of Sciences, Warsaw, Poland
pdlotko@impan.pl

***Abstract*—The Mapper algorithm is a technique from TDA used to create low-dimensional graph-based representations of high-dimensional data, proven effective in numerous exploratory data analysis tasks. The Mapper algorithm's output depends on several user-chosen parameters, and selecting their values is a non-trivial choice, significantly narrowing its potential application in real-world scenarios. Research attempting to assist in selection of the parameters has been very limited to date. This paper is the first one to address the selection of Mapper's three parameters simultaneously. The proposed idea incorporates the concept of Ensemble Learning into the Mapper algorithm. Using several datasets with known labels, we show that our method outperforms two baselines in recovering the dataset structure.**

***Index Terms*—Topological Data Analysis, Mapper, Ensemble Learning**

I. Introduction

The Mapper algorithm, introduced in [5], is a Topological Data Analysis (TDA) technique used to visualize data as a graph with nodes representing sample clusters and edges denoting similarity. By coloring the nodes based on a feature of interest, practitioners can effectively visualize a scalar-valued function on a high-dimensional dataset. The algorithm has yielded impressive practical successes in various exploratory data analysis tasks [2], [10], [16].

Mapper's output relies on several user-chosen parameters. Adjusting any of these parameters can result in spurious changes, and as an unsupervised learning technique, there is no accepted way to evaluate output quality [1]. This presents significant challenges to practitioners as there is no clear indicator of which outputs are relevant data visualizations. In this paper, we propose a new approach to selecting Mapper's parameters by incorporating the concept of Ensemble Learning in the Mapper graph creation pipeline. Ensemble Learning allows us to consider different parameter values, alleviating the risk of choosing a single parameter combination which results in degenerate output. By adopting a data-driven approach, we ensure that the selected parameters lead to output that is stable and representative of the underlying data. Furtherly, we utilize structural signals from the data to selectively combine subsets of graphs to obtain a final unified Mapper graph.

Another challenge we tackle, is that there may be many different, yet relevant outputs of the Mapper algorithm. We use graph distance measures to reveal structural coherence between graphs ensembling the most similar, frequently occurring shapes, revealing diverse data perspectives.

This work was supported by the Department for the Economy (DfE), Northern Ireland, UK.

II. Preliminaries

A. The Mapper Algorithm

We define Mapper as the function $f_m : (X, r, g, l) \rightarrow G$ taking a data set (X), resolution (r), gain (g), and lens function (f) as its input parameters, and obtaining the graph $G = \{V, E\}$ as its output, where V and E refer to sets of vertices and edges respectively.

The Mapper algorithm runs in four steps: (1) **Project data** to a lower dimension using a user-defined lens function $f : X \rightarrow \mathbb{R}$. The lens function is typically chosen to highlight data qualities relevant to the research question, such as a particular feature or combination of features. (2) **Define a covering** of $f(X)$ with the number of intervals of the same length, $U = \cup_{i=1}^{r} u_i$. The r and g parameters divide $f(X)$ into a specified number of intervals and control the overlap between them, respectively. Increasing r will increase Mapper graph complexity by yielding more but smaller nodes, and increasing g will result in more connectivity. (3) **Cluster data** in the preimage $f^{-1}(u_i)$ for each cover element $u_i \in U$, splitting the local sets into k_{u_i} clusters $C_{u_{i,1}}, ..., C_{u_{i,k_{u_i}}}$. The clusters define the vertices $V \in G$. (4) **Define edge set** E**.** If two clusters have a non-empty intersection, $(C_{u_i} \cap C_{u_i'} \neq \emptyset)$ an edge is included for those corresponding nodes. An example of Mapper graph construction is shown in Fig. 1.

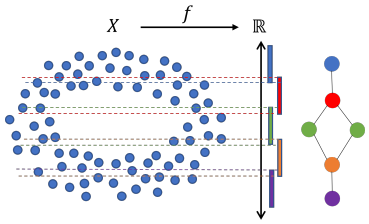


Fig. 1. Mapper graph construction on point cloud dataset X. The lens function is the vertical position, covered by five overlapping bins, and the final Mapper output according to clustering in X.

B. Determining distance between Mapper graphs

Two methods for measuring Mapper graph distance are:

1) Matching Distance: Proposed in [1], this metric pairs nodes between two graphs by their similarity. A minimal bipartite matching algorithm, e.g., Hungarian algorithm [8], is used to pair nodes across graphs minimizing the symmetric difference between the node samples. Unequal nodes across the graphs result in unpaired nodes matched to an empty set. The overall graph distance is then the sum of all node pairings scaled by the total number of samples giving a value between 1 (no samples matched) and 0 (all samples matched).

2) Network Augmented Wasserstein Distance: The Network Augmented Wasserstein (NAW) metric [11] quantifies the distance between graphs with multiple connected components, varying data samples, and cover parameters. It formulates graph distances as an optimal transport problem, where the distances between nodes are based on the product of a probability density function and a cost function metric measuring the difference in node eccentricities between the graphs.

III. Previous Work

The authors of [1] used a stability measure to select Mapper cover parameters. They measure stability by partitioning the dataset into subsets, constructing Mapper graphs with consistent parameters, and calculating pairwise matching distances. This process is repeated to identify local minima over the parameter space as lower average distances signifies greater stability. However, choosing a single parameter combination remains challenging due to multiple potentially "stable" representations. Furthermore, stability doesn't ensure useful data representation; too few nodes can omit important details, and many isolated nodes yield a stable yet useless configuration.

In their work in [7], the authors used Fuzzy Silhouette Score to identify good parameter choices. The ten highest-scoring graphs are combined using an ensemble method adapted from [12]. Using the cluster-similarity metric from [12], they construct a correlation matrix, with each element representing sample composition correlation of node pairs. They convert the correlation to a distance matrix, and apply hierarchical clustering to the nodes. Where a sample is within a node assigned to a cluster, it is assigned to the relative node, and if across multiple clusters, an edge is drawn between those nodes. Silhouette scores were inconsistent indicators of graph quality, as the metric favors convex clusters over non-convex.

In [4], the authors proposed F-Mapper as an alternative to the standard Mapper algorithm. F-Mapper uses Fuzzy C-Means clustering [3] to define the cover over the lens with irregular intervals. As a soft clustering method, Fuzzy C-Means assigns a value between 0 and 1 to each sample for each of the c clusters. F-Mapper uses a threshold parameter where each sample is assigned to any cluster with a value greater than the threshold. The algorithm then clusters on the pullback of the overlapping sets defined by Fuzzy C-Means clusters. The authors fine-tuned the parameters to topologically match the output of the standard Mapper on known datasets and found that F-Mapper outperformed the standard Mapper in terms of Silhouette score [17]. However, the lens, c parameter, and threshold parameter in F-Mapper are user-chosen.

While prior research has tackled Mapper's parameter selection, none addressed lens selection or situations where multiple relevant Mapper graphs may arise. Our work is the first, to our knowledge, to simultaneously address all three Mapper parameters. Our proposed method also identifies strong signals from the input data and generates their corresponding graphs.

IV. Motivation and Problem Statement

The Mapper algorithm's output depends on the selection of its three parameters [1]. The choice of lens function is not trivial. Projecting data to one feature or a linear combination is a common approach that can reveal interesting dataset signals. However, with high-dimensional data a challenge is that using different features for projection leads to different graphs as multiple dimensions of the input points are compressed to a single lens value. There is no obvious way to determine which outputs (one or more) are optimal and relevant to our analysis. Additionally, poorly chosen resolution and gain parameters can lead to a misshapen graph, and there is no intuitive way of selecting their values. We demonstrate a simple toy example in Fig. 2 to visualize these challenges.

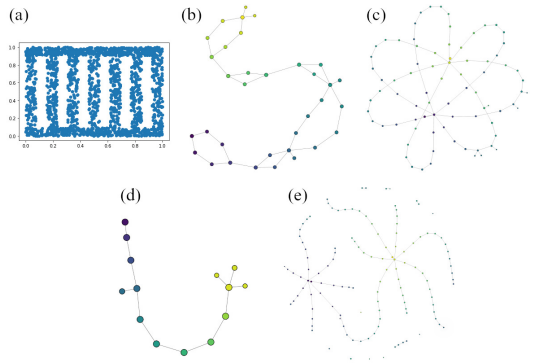


Fig. 2. Mapper graphs constructed using alternative parameters on a toy dataset (a) Scatter plot (b) X-axis as lens function. (c) Y-axis as lens function. (d) X-axis as lens with poor cover. (e) Y-axis as lens with poor cover.

Fig. 2(b) and 2(c) show the outputs of the Mapper algorithm on a 2-dimensional dataset presented in Fig. 2(a). The lens is selected as the X and Y axis values, respectively. Both graphs show unique characteristics of the dataset viewed through the lens function and may be relevant to different types of analysis.

Poor choices for the resolution and gain parameters can lead to unexpected Mapper graphs, as shown in Fig. 2(d) and 2(e). In Fig. 2(d), the resolution is low, resulting in a coarse-grained cover with no loops. In Fig. 2(e), the graph should consist of a single component, but there are several disconnected nodes, and the loops are not recovered as gain is set too low.

Currently selecting resolution, gain, and lens function settings requires setting values in an ad-hoc fashion. This can be unintuitive for non-expert users, and poor choices lead to degenerate graphs. An automated end-to-end pipeline, that determines parameters likely to return a graph with stable structure true to the underlying data, is an important problem to address to make the Mapper algorithm more accessible. Additionally, depending on the lens chosen, different graph

shapes are obtained, illustrated in Fig. 2(b) and 2(c). Therefore, another important objective of our study is to develop a method that can retrieve diverse data representations, capturing the most relevant shapes based on the strongest data signals.

V. Proposed Methodology

The overall idea of the method is to first define a broad value range for the three parameters, and then identify a collection of stable, representative Mapper graphs within this space. We set the values of gain $P = \{p_1, ..., p_{p_{max}}\}$, resolution $B = \{b_1, ..., b_{b_{max}}\}$, and lens functions $L = \{l_1, ..., l_{max_l}\}$. Following this, the pipeline runs in three main steps: (1) Selecting optimal cover parameters for each lens function; (2) Selecting the most common data shapes across the lens functions; (3) Combining Mapper graphs for the selected lens functions. Each step is discussed in the following section.

A. Selecting Gain and Resolution for Each Lens Function

The goal of this step is to select optimal cover parameters for each lens function. In order to do this, we first identify a pool of candidate cover parameters and select the one most representative. We adapt the approach from [1] to identify paired values of gain and resolution that give a stable Mapper graph. To do this, for each lens function $l \in L$, we construct an instability matrix $I_l^{|B|\times|P|}$, where each element $i_{b,p}$ represents the instability of the Mapper output for parameter combination $p \in P$ and $b \in B$. Instability is measured as an average pairwise distance between Mapper graphs obtained on different data subsets. We then find all parameter combinations which lay on the local minima of $I_l^{|B|\times|P|}$. That is, each pair with a lower instability than its neighboring values. The selected pairs create a pool of candidate cover parameters for each lens. Using local minima across the parameter space can still result in significantly different outputs. We assume that each lens has a prominent shape with a high probability of occurrence across the parameter space. Therefore, as the next step we introduce a refined data-driven approach to identify the most prominent shape associated with each lens across the graph pool C_l.

We first perform clustering on all graphs constructed for each of the lens functions (individually). Clustering the graphs by similarity allows us to assess the typical structure associated with the lens. If there is a prominent structure, the largest cluster is likely to represent it. Conversely, noisy representations will be limited to few local minima. We use the NAW metric from [11] to construct a pairwise distance matrix between all graphs from C_l. We then apply average-linkage clustering to the distance matrix, which can encode the similarity between the Mapper graphs as a dendrogram. With the dendrogram, we cluster the graphs using the automated tree-cutting method from [9]. For each lens function, we identify the largest cluster and select its centroid (lowest average pairwise intra-group NAW distance) as the representative.

B. Identifying Common Shapes Across Lens Functions

Repeating the previous step for each lens function from L, we compose a set of what we call base graphs $G = \{g_1, ..., g_{|L|}\}$ that represent different signals (shapes) detected from the dataset X using different lens functions. Section IV demonstrates different lenses can produce significantly different graphs. The aim of this step is to identify the most recurrent structural qualities across all lens functions. For this purpose, we cluster the base graphs using hierarchical clustering on the pairwise NAW distance matrix. Like Section V-A, we consider the largest grouping to provide the strongest signal from the underlying data. In cases of multiple substantial clusters, we expect varying structures and data perspectives. Hence, more than one Mapper graph can be provided as the output.

C. Ensemble Construction

The last phase within the proposed method is to identify the final Mapper graph/s by constructing the ensemble of all graphs from the largest cluster/s found in the previous step.

1) Find Ensemble Mapper Nodes: We propose to identify the final Mapper graph nodes by considering sample co-occurrence across all nodes of the base Mapper graphs in G. Unlike the technique from [7], which assumes cluster correspondence by using a cluster correlation metric, our method does not require this assumption. Using sample co-occurrence instead of cluster correlation may be more desirable because of the lower cluster correspondence across different lens functions, gain, and resolution parameters.

Each element of the co-occurrence matrix C^{co}_{n*n} denotes how often a particular pair of samples from X are grouped together in the same nodes across all $g \in G$.

$$C^{co}_{n*n}(x_i, x_j) = \sum_{g\in G}\sum_{v\in g} 1_{v(x_i,x_j)},$$
$$\text{Where, } 1_{v(x_i,x_j)} = \begin{cases} 1, & \text{if } x_i \in v \text{ and } x_j \in v \\ 0, & \text{if } \neg(x_i \in v \text{ and } x_j \in v) \end{cases} \quad (1)$$

The notion of distance encoded in C^{co}_{n*n} facilitates hierarchical clustering to group the samples into nodes for the final graph ensemble. Ward's method is a suitable hierarchical clustering criterion for our pipeline as it minimizes within-cluster variance [6]. A common automated method for clustering from a dendrogram involves cutting branches at the largest gap. Alternatively, the dynamic tree cut method from [9], which we use in our experimentation, detects clusters based on the shape of dendrogram sub-trees. As the output of this process, we obtain a set of k clusters representing the nodes $V \in G$.

2) Find Ensemble Mapper Edges: To define the edges in the final graph, we create the sample connectivity matrix C^{sc}_{n*n}, which is constructed similarly to the co-occurrence matrix and denotes how often a pair of samples are in adjacent nodes across all base graphs $g \in G$.

With C^{sc}_{n*n}, we construct the node connectivity matrix C^{nc}_{k*k}, where each element corresponds to a weighting between a node pair. High values correspond to high connectivity, and low values correspond to low connectivity. The connectivity values in C^{nc}_{k*k} are calculated as per Equation 2.

$$C^{nc}_{k*k}(v \in V, w \in V) = \frac{1}{|v| * |w|}\sum_{x_i\in v}\sum_{x_j\in w} C^{sc}_{n*n}(x_i, x_j) \quad (2)$$

To determine edges in the final ensemble graph, we apply a threshold/filter value to the node connectivity matrix. Elements exceeding the filter value creates an edge between the corresponding nodes. For a given filter value f, the edge set E of the final ensemble graph $G_e = \{V, E\}$ is composed as per Equation 3 where (v, w) is an edge between v and w.

$$E = \{(v, w) \in V \times V | C^{nc}_{k*k}(v, w) \geq f\} \tag{3}$$

To choose an appropriate filter value, we firstly find the median of the node connectivity matrix. Subsequently we only consider values above the median, allowing us to prioritize stronger connections and disregard weaker ones. The filter value is taken as the mean of these selected values added their standard deviation times a 'deviation factor' parameter (α) chosen to relax or restrict the edge inclusion (Equation 4). This approach captures edge strengths that deviate significantly from the mean, including only the most likely edges. For our experimentation we set $\alpha = 1.25$ providing a moderate level of leniency, allowing for the inclusion of connections that are relatively weaker than the strongest connections but still considered significant enough to include in the graph.

$$f = \mu + \alpha\sigma \tag{4}$$

VI. Experimentation

A. Experiment Setup and Details

1) Datasets Used: For the experiments in this section, we use five datasets of known shape embedded in 100-dimensions using the TaDAsets Python library [19]. The datasets include:

1) **Blobs:** 25 non-intersecting Gaussian blobs ($n = 2000$).
2) **Chain Link:** 2 interlocking loops ($n = 1000$).
3) **Donut:** 1 torus with 4 internal spheres ($n = 2800$).
4) **Figure Eight:** Eight with two spheres ($n = 1000$).
5) **Spheres:** Large sphere with 10 smaller interior spheres ($n = 2000$).

2) Evaluation Method: To allow quantitative evaluation, we assign labels samples based on their component membership. For instance, with the Figure Eight dataset of three distinct components, we label samples according to their membership in spheres or the main figure eight structure.

We use Normalized Mutual Information (NMI) [20] for quantitative evaluation of the sample groupings in the final Mapper graph. NMI has an upper bound of 1 for graphs with all samples with a common label in the same connected component and 0 when connected components contain samples with different labels or are split across multiple connected components. We also visually assess the final graph matches the known shapes for each dataset. When visualizing the Mapper graphs, the nodes are colored by the sample labels.

3) Baseline Methods: Currently, no methods in the literature address the choice of all three gain, resolution and lens function parameters. As a baseline for evaluation, we use Ensemble Mapper [7] and F-Mapper [4]. Our technique is advantageous over both baseline methods from a usability perspective as lens selection is automated, while the baseline methods require a carefully chosen lens to highlight important data characteristics. We also compare against the average performance of the standard Mapper output given the range of parameters and lenses used for our ensemble technique.

4) Implementation: We consider resolution parameters in the range [2,3,...,22] and gain in [2.50%,5%,...,50%], for 440 combinations as in [1]. The upper bound for gain is 50% since higher values cause non-neighboring bin overlap and loss of local structure preservation in the graph. More fine grain increments can be considered in trade-off with computation time. A set of 40 lens functions $L = \{l_1, ..., l_{max_l}\}$ is defined where each lens function $l \in L$ is a projection to a random feature selected from the dataset X. Note, that more lens functions could be considered with the stopping criteria set as no changes being observed in the final output.

For step one of our method, we split the data into 10 folds to measure the instability [1]. We repeat the experiment 10 times with a different random feature selection and record the average NMI and standard deviation for the combined graphs.

For the baseline Ensemble Mapper technique [7], we use the same gain and resolution parameters and select the 10 Mapper graphs with the highest silhouette scores. Since this technique does not consider graphs constructed across lens functions, we use t-SNE [18] as the lens function (scikit-learn default parameters [14]), as it is commonly applied with Mapper in existing literature [15], [21]. We select the average number of nodes across the 10 selected graphs as the number of nodes in the final graph.

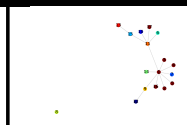
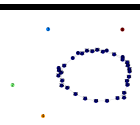
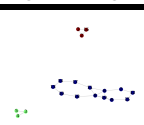
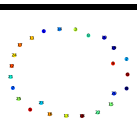
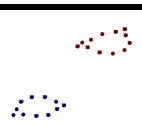
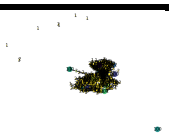
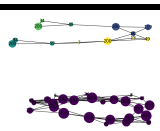
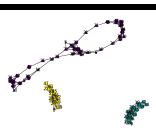
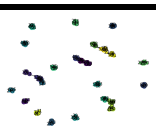
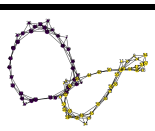
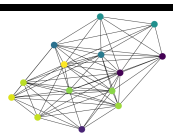
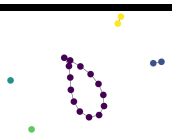
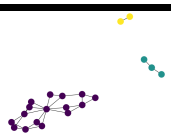
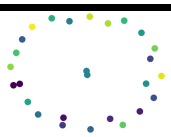
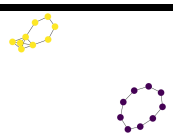
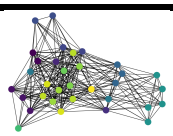
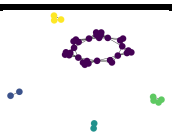
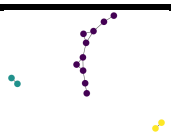
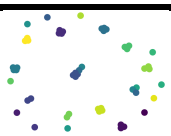
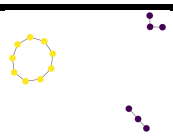
The second baseline F-Mapper [4] does not automate the selection of the gain or resolution parameters, as it requires different parameters. The C parameter determines the number of cover elements found via C-Means clustering, and the threshold parameter assigns samples to the cover elements. Again, we choose t-SNE as the lens function. As the authors did not indicate how to select the value of C, we run the algorithm for many combinations in an ad-hoc fashion (C=[2,4,...,20] and threshold=[0.05,0.10,...,0.25]) and choose an optimal selection per dataset. It should be noted, however, that the choice of parameters is not obvious in a real scenario.

B. Results and Discussion

Table I presents results for our proposed method and baseline techniques. The first column indicates the technique and columns 2 to 6 represent the dataset. The graph output from our pipeline and its average NMI score are shown in the first row. To demonstrate the stability of our method we also report the standard deviation across the 10 runs with varied random lens feature selections. Subsequent rows show results for Ensemble Mapper [7], fine-tuned F-Mapper [4], non-optimal F-Mapper parameters, and the average NMI with standard deviation for the standard Mapper algorithm over all parameter combinations considered in the proposed method. Graph nodes are colored based on majority sample labels.

Table I shows that in 4 of 5 datasets, the proposed method scores a perfect NMI score of 1 across all 10 runs of 40 randomly selected lenses. The Spheres dataset is an exception

TABLE I
OUTPUT GRAPHS AND NMI SCORES FOR OUR PIPELINE, ENSEMBLE MAPPER [7] AND F-MAPPER [4] ON THE DATASETS FROM SECTION VI-A1.

Technique		Dataset				
		Spheres	Donut	Figure Eight	Blobs	Chain Link
Proposed Method	Output					
	NMI±SD	0.261 ±0.224	1.00 ±0.00	1.00 ±0.00	1.00 ±0.00	1.00 ±0.00
Ensemble Mapper	Output					
	NMI	0.161	0.794	1.00	0.907	1.00
F-Mapper (optimal)	Output					
	NMI	0.00	1.00	1.00	1.00	1.00
F-Mapper (non-optimal)	Output					
	NMI	0.00	1.00	1.00	0.986	0.809
Standard Mapper	NMI±SD	0.196±0.233	0.936±0.057	0.968±0.084	0.998±0.001	0.879±0.189

where our method fails to detect the individual components. We also assessed the individual graphs used in the final ensemble. The analysis showed that the proportion of selected graphs which capture the ideal topology varies widely depending on the dataset. All components in the Blobs dataset are recovered the chosen lens representatives, whereas the Chain Link, Figure Eight, and Donut datasets have 50%, 70% and 90%, respectively. Our ensemble successfully recovers the expected graph structure in 10/10 runs for Chain Link, Blobs, and Donut datasets. With the Figure Eight dataset two ensembles were not characteristically correct leaving 8/10 correct. The outlier is Spheres with no graph able to recover the components.

Compared to the Ensemble Mapper baseline (row two), our approach outperforms in NMI for the Blobs and Donut datasets. It effectively separates expected components, whereas the baseline tends to merge them. Both methods achieve a perfect NMI score for the Figure Eight and Chain link datasets. For Spheres, neither method can separate the components.

The F-Mapper results in Table I, row three, shows the same NMI scores for 4 out of 5 datasets, except for Spheres, where our proposed method performs better. The datasets' structures, except for Spheres, are distinguishable from the graphs. Another consideration for F-Mapper is output instability due to the non-deterministic nature of finding the cover using C-Means clustering (output relying on initialization state). The parameters used for the F-Mapper output in row three were selected after running for numerous parameters and selecting the best representations. Therefore, the shown result is not guaranteed. The sensitivity to parameters is evident in row 4, where parameter adjustments led to unexpected changes, like failing to recover loops in Chain Link and Figure Eight data.

The fifth row shows the average NMI for the graphs output using the standard Mapper algorithm with all combinations of parameters from our proposed method experiments. The proposed method consistently outperforms across all datasets, with the standard deviation highlighting output variability.

We perform further experimentation comparing our proposed method against F-Mapper and standard Mapper using the t-SNE as the lens varying the user chosen perplexity parameter for 10 different lenses. We select a representative graph for each lens and combine them into a final graph representation. In Table II, the first column shows that the proposed method achieves a perfect NMI for all datasets except for Spheres, where it scores 0.283. Comparing this to the second and third column, standard Mapper and F-Mapper respectively, our method outperforms each with the exception

of equally optimal performance on the Donut dataset with F-Mapper. Notably, the standard deviation of standard Mapper is high highlighting how the proposed method addresses reliance on choosing a single potentially poor parameter choice.

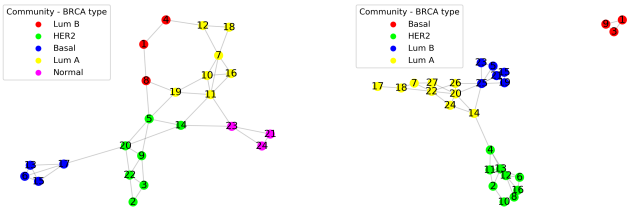
TABLE II
NMI SCORES WITH T-SNE VARIED BY PERPLEXITY AS LENS FUNCTIONS.

Dataset	Proposed Method NMI	Standard Mapper NMI±SD	F-Mapper NMI±SD
Spheres	0.283	0.163 ± 0.139	0.00±0.000
Donut	1.00	0.670 ± 0.152	1.00±0.000
Figure Eight	1.00	0.833 ± 0.119	0.940 ± 0.096
Blobs	1.00	0.999 ± 0.001	0.994± 0.014
Chain Link	1.00	0.611 ± 0.194	0.950 ± 0.168

C. Experiment on Real-World Data

Here we demonstrate that combining graphs from different lens function groupings yields structurally distinct graphs. We run our pipeline on the TCGA-BRCA (breast cancer) gene-expression data [13] with 1027 samples, 50 features, and maintain consistent parameters with prior experiments. The 50 genes, known as the PAM50 assay, help classify breast cancer into five subtypes. Given substantial subtype similarities, we do not expect five distinct connected components in the output graph. Instead, we focus on analyzing the variation in majority ground truth sample labels across nodes.

Fig. 3(a) shows the resulting graph from combining the largest group of 7 graphs, which has a single connected component with nodes coloured by the majority sample labels. Fig. 3(b) shows the graph from the second lens cluster of 3 graphs, which is 2 separate connected components. From an exploratory data analysis perspective, the disconnection between the distinctly Basal community in Fig. 3(b) and small community of normal-like samples in Fig. 3(a) allow different lines of inquiry into substructures within the graphs.



(a) (b)

Fig. 3. Output from combining different lens groupings of the TCGA-BRCA data. (a) Largest grouping (b) Second largest grouping

In real-world data sets such as this, we can not know the true underlying structure, and it is important to consider how different structural information across lenses can emphasize different regions of interest within the dataset. The proposed Mapper pipeline enhances exploratory data analysis on real-world data by uncovering the strongest structural signals across different lenses, offering diverse data representations.

VII. CONCLUSIONS

Our ensemble method yields perfect NMI clustering quality on 4 of 5 datasets, even when a significant portion of the graphs used within the ensemble fail to display the dataset's structural features. Visual analysis showed that while stability helps assess graph quality, it does not guarantee a correct Mapper graph shape. Compared to the Ensemble Mapper baseline, our method performs better on 3 of 5 datasets and matches it on 2 others in terms of NMI. The second baseline technique, F-Mapper, performs equally well in terms of NMI but requires its own parameter-tuning. Furthermore, our method is the only one evaluated that addresses the selection of all three parameters. The results indicate that our proposed pipeline is successful in recovering high-dimensional data shape and could be helpful for practitioners unsure of which lens, gain, or resolution parameters to use. One aspect of the proposed method we aim to improve in future work is the computation time required for the exhaustive search of parameter combinations to calculate instability matrices. Improvements could also be made to increase the scalability of the pipeline to larger datasets.

REFERENCES

[1] F. Belchı, J. Brodzki, M. Burfitt, and M. Niranjan. A numerical measure of the instability of mapper-type algorithms. *JMLR*, 21:1–45, 2020.

[2] L. Benedetti-Cecchi. Complex networks of marine heatwaves reveal abrupt transitions in the global ocean. *Sci. Rep.*, 11(1):1–11, 2021.

[3] J. C. Bezdek, R. Ehrlich, and W. Full. Fcm: The fuzzy c-means clustering algorithm. *Comput. Geosci.*, 10(2):191–203, 1984.

[4] Q.-T. Bui et al. F-mapper: A fuzzy mapper clustering algorithm. *Knowl.-Based Syst.*, 189:105107, 2020.

[5] G. Carlsson. Topology and data. *Bulletin AMS*, 46(2):255–308, 2009.

[6] J. H. W. Jr. Hierarchical grouping to optimize an objective function. *J. Am. Stat. Assoc.*, 58(301):236–244, 1963.

[7] S. J. Kang and Y. Lim. Ensemble mapper. *Stat*, 10(1), 2021.

[8] H. W. Kuhn. The hungarian method for the assignment problem. *Nav. Res. Logist.*, 2(1-2):83–97, 1955.

[9] P. Langfelder, B. Zhang, and S. Horvath. Defining clusters from a hierarchical cluster tree: the Dynamic Tree Cut package for R. *Bioinformatics*, 24(5):719–720, 11 2007.

[10] C. Loughrey, P. Fitzpatrick, N. Orr, and A. Jurek-Loughrey. The topology of data: opportunities for cancer research. *Bioinformatics (Oxford, England)*, 37(19):3091–3098, Oct. 2021.

[11] M. McCabe. Mapper comparison with wasserstein metrics. *arXiv preprint arXiv:1812.06232*, 2018.

[12] M. Mojarad et al. A fuzzy clustering ensemble based on cluster clustering and iterative fusion of base clusters. *Appl. Intell.*, 49(7):2567–2581, 2019.

[13] C. G. A. R. Network et al. Comprehensive and integrative genomic characterization of hepatocellular carcinoma. *Cell*, 169(7):1327, 2017.

[14] F. Pedregosa et al. Scikit-learn: Machine learning in Python. *J MACH LEARN RES*, 12:2825–2830, 2011.

[15] O. Rafique and A. H. Mir. A topological approach for cancer subtyping from gene expression data. *J. Biomed. Inform.*, 102:103357, 2020.

[16] A. Robles, M. Hajij, and P. Rosen. The shape of an image: A study of mapper on images. *arXiv preprint arXiv:1710.09008*, 2017.

[17] P. J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *J. Comput. Appl. Math.*, 20:53–65, 1987.

[18] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *science*, 290(5500):2323–2326, 2000.

[19] N. Saul and C. Tralie. Scikit-tda: Tda for python, 2019.

[20] N. X. Vinh, J. Epps, and J. Bailey. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *JMLR*, 11(95):2837–2854, 2010.

[21] T. Wang, T. Johnson, J. Zhang, and K. Huang. Topological methods for visualization and analysis of high dimensional single-cell rna sequencing data. In *Pac. Symp. Biocomput.*, volume 24, pages 350–361, 2019.