Learned Laplacian-driven 3D mesh segmentation

Blaž Bulić, Ciril Bohak

University of Ljubljana, Faculty of Computer and Information Science
E-mail: bb53717@student.uni-lj.si, ciril.bohak@fri.uni-lj.si

Abstract

This work introduces a 3D point cloud segmentation method by integrating Neural Laplacian Operators (NeLo) with the Mapper algorithm. We replace Mapper's manual filter functions with NeLo's learned spectral embeddings to automate the selection of the filter function. Evaluated on ShapeNet objects, our approach produces finer segmentations than spectral clustering and manual Laplacian baselines, especially for geometrically distinct features.

1 Introduction

3D point clouds appear increasingly often as a data representation in various domains, including autonomous navigation, robotics, urban mapping, and biomedical imaging. Despite their popularity, the usage of point clouds comes with multiple challenges due to noise, irregular sampling, and the lack of explicit connectivity of the sampled object. Traditional approaches for processing 3D point clouds rely on triangulation-based methods or handcrafted features to first construct a mesh from the points, which often introduce inconsistency when dealing with thin structures or sharp features of the clouds. These limitations make tasks such as segmentation, shape analysis, and object recognition challenging for objects that exhibit the unwanted properties.

Recent advances in deep learning have taken steps towards overcoming these challenges. In particular, NeLo [6] provides a new solution by learning the Laplacian operator directly from point cloud data. NeLo constructs a k-nearest neighbor (kNN) graph from the raw data and uses a Graph Neural Network (GNN) to learn appropriate edge weights, used for approximating the continuous Laplacian operator. The spectral embeddings we get from NeLo provide us with intrinsic geometric features, making them well-suited for various downstream applications.

Similarly, the Mapper algorithm, a tool from topological data analysis, has been used to generate simplified representations of high-dimensional data. The mapper works by clustering data based on a chosen filter function and building a nerve (graph) that reflects the global topological structure. However, the quality of Mapper's output is highly sensitive to the choice of filter function, which is typically selected manually. This manual selection can lead to inconsistent results and limits the method's applicability.

The goal of our project is to integrate NeLo with Mapper for 3D point clouds, by using the spectral embeddings from NeLo as the filter function in Mapper. This substitution transforms the Mapper pipeline into an end-to-end framework that uses learned geometric features instead of relying on handcrafted choices. We test our approach on one of the most basic use-cases, that is 3D point cloud segmentation, a critical step for many downstream tasks.

2 Related Work

The field of 3D data analysis and neural operator learning has evolved in recent years. Our work builds upon several key contributions listed below.

Laplacian for Geometry Processing: Laplacian operators play a central role in many geometry processing tasks. Botsch et al. [1] provide an overview of Laplacian-based methods in their book *Polygon Mesh Processing*. They detail how the Laplacian can be used for mesh smoothing, parameterization, and other use cases.

Neural Laplacian Operators: Pang et al. [6] introduced the Neural Laplacian Operator for 3D point clouds. Their method uses a GNN to learn edge weights on a k-nearest neighbour graph, producing spectral embeddings that robustly capture intrinsic geometric features even in the presence of noise and sparsity. This data-driven approach has set a new benchmark for point cloud processing. The authors get promising results for tasks such as head diffusion, smoothing, and others.

Topological Data Analysis and Mapper: Singh et al. [12] introduce a topological method for high-dimensional data analysis with the development of the Mapper algorithm. Mapper creates a simplified graph representation by clustering data based on a continuous filter function. Although Mapper has proven effective in revealing global structures, its reliance on manually chosen filters often presents the biggest challenge.

Neural Operator Learning: Li et al. [5] developed the Fourier neural operator, which uses frequency-domain techniques to achieve discretization-independent learning. This highlights the effectiveness of spectral methods in capturing global features, a concept that motivates our use of NeLo's spectral embeddings as a filter in Mapper.

Point Cloud Segmentation: PointNet, introduced by Qi et al. [7], was among the first architectures to directly process point cloud data for segmentation and classification.

Although PointNet achieves competitive performance, it relies on engineered features. In contrast, our approach utilizes learned spectral embeddings to drive segmentation, providing a more robust and natural grouping of points. Soon after, the same authors introduced an improvement called PointNet++ [8] by enabling multiscale feature learning, which allows the network to capture local structures and relationships within point clouds more effectively, thereby enhancing segmentation accuracy and performance across diverse 3D geometries.

PointNeXt [9] builds on the foundations of PointNet++ by introducing design principles for constructing improved point-based backbones. It simplifies the architecture while preserving or improving accuracy, using residual blocks, local aggregation, and normalization strategies. PointNeXt achieves state-of-the-art results on several segmentation benchmarks while maintaining computational efficiency. Although effective, PointNeXt, like its predecessors, relies on local geometric structures and handcrafted inductive biases, whereas our method leverages global spectral embeddings learned through a neural Laplacian operator.

These contributions have individually advanced the fields of geometric deep learning and topological data analysis. Our project aims to combine parts of different approaches for a specific use case of 3D point cloud segmentation.

3 Method

In this section, we describe how we compute the end segmentation of a 3D point cloud. In Section 3.1 we show the process of extracting the Laplacian matrix from the NeLo framework. In Sections 3.2 and 3.3 we describe the two methods we use for segmentation. Lastly, we describe the manually computed Laplacian matrix we use for comparison in Section 3.4.

3.1 Learned Laplacian Extraction

NeLo framework output is not a Laplacian operator in its matrix form, but is represented as a graph with weighted edges. Therefore, we first need to extract the corresponding matrix representation to use it in our segmentation.

Given an input 3D point cloud, we construct a *mesh-like graph* by treating each 3D point as a vertex and establishing edges based on spatial proximity. In practice, we employ a k-nearest neighbor (KNN) search in Euclidean space to connect each point with its nearest neighbors, resulting in an undirected graph. This procedure is analogous to forming the 1-skeleton of a mesh: the connectivity encodes local neighborhood relationships without requiring explicit surface reconstruction. The pre-trained NeLo model then predicts an edge weight w_{ij} for every edge (i, j) in this graph. We assemble these weights into a sparse adjacency matrix, from which the corresponding Laplacian operator can be derived:

$$A_{ij} = \tfrac{1}{2}\big(w_{ij} + w_{ji}\big),$$

and then form the degree matrix D with $D_{ii} = \sum_j A_{ij}$. Finally, the Laplacian matrix is computed as

$$L_{\text{NeLo}} \;=\; D - A,$$

which we can then use for downstream applications.

3.2 Spectral Clustering

The first segmentation method takes the spectral embedding of the Laplacian matrix and then obtains clusters directly from that. To segment the 3D point cloud into k parts, we use L_{NeLo} and compute its $k+1$ smallest eigenpairs $(\{(\lambda_0, \phi_0), (\lambda_1, \phi_1), \ldots, (\lambda_k, \phi_k)\}$ using SciPy[1]'s `eigsh` function. We discard the trivial constant eigenvector ϕ_0, and embed each vertex i as:

$$\big(\phi_1(i),\, \phi_2(i),\, \ldots, \phi_k(i)\big) \;\in\; \mathbb{R}^k.$$

K-means clustering is then applied in this k-dimensional space to obtain cluster labels for each point.

3.3 Mapper-Based Segmentation

The second segmentation method leverages the Mapper algorithm in the following way:

1. We use the first m nontrivial eigenvectors of L_{NeLo} as a filter function in the KeplerMapper pipeline. Let $\psi_1, \ldots, \psi_m$ be these eigenvectors; we stack them into an $N \times m$ matrix called "filter matrix".

2. With cover parameters $\{\texttt{n\_cubes}, \texttt{perc\_overlap}\}$ and the DBSCAN [11] clustering algorithm, Mapper partitions the filter space into overlapping bins and clusters within each.

3. The resulting nerve graph nodes induce an assignment of each original point to one or more Mapper clusters; we resolve overlaps by assigning each point to the node it appears in most frequently.

3.4 Manual Laplacian Baseline

In order to compare and evaluate our results, we construct a standard Gaussian-kernel Laplacian on the same mesh vertices. We compute a full adjacency matrix

$$W_{ij} = \exp\!\big(-\|x_i - x_j\|^2/(2\sigma^2)\big),$$

zeroed on the diagonal, form $D_{ii} = \sum_j W_{ij}$, and let $L_{\text{Gauss}} = D - W$. We then perform both spectral clustering and Mapper segmentation exactly as above, but using the manually computed L_{Gauss} in place of L_{NeLo}.

4 Results

We test out our approach on objects from the ShapeNet [3] database. Our focus is on the qualitative/visual results, as meaningful quantitative metrics are hard to come by without the ground truth segmentations, which we do not have. Despite that, we can compare some metrics that measure the quality of clustering. The chosen metrics are:

[1] `https://scipy.org`

Table 1: Clustering-quality metrics (Silhouette, Davies-Bouldin index (DBI), and Calinski-Harabasz index (CHI) / #Clusters) for our three examples

Method	Chair				Plane				Desk			
	Silhuette	DBI	CHI	#Clusters	Silhuette	DBI	CHI	#Clusters	Silhuette	DBI	CHI	#Clusters
Spectral + k-means	0.689	0.559	5363.990	5	0.641	0.461	3179.109	5	0.390	0.893	3963.550	5
Mapper (NeLo)	0.193	0.784	15945.960	94	-0.249	4.577	118.624	79	-0.295	1.373	34.891	396
Mapper (Gauss)	0.033	0.800	8212.560	78	-0.202	3.746	152.137	51	-0.306	1.832	47.114	352

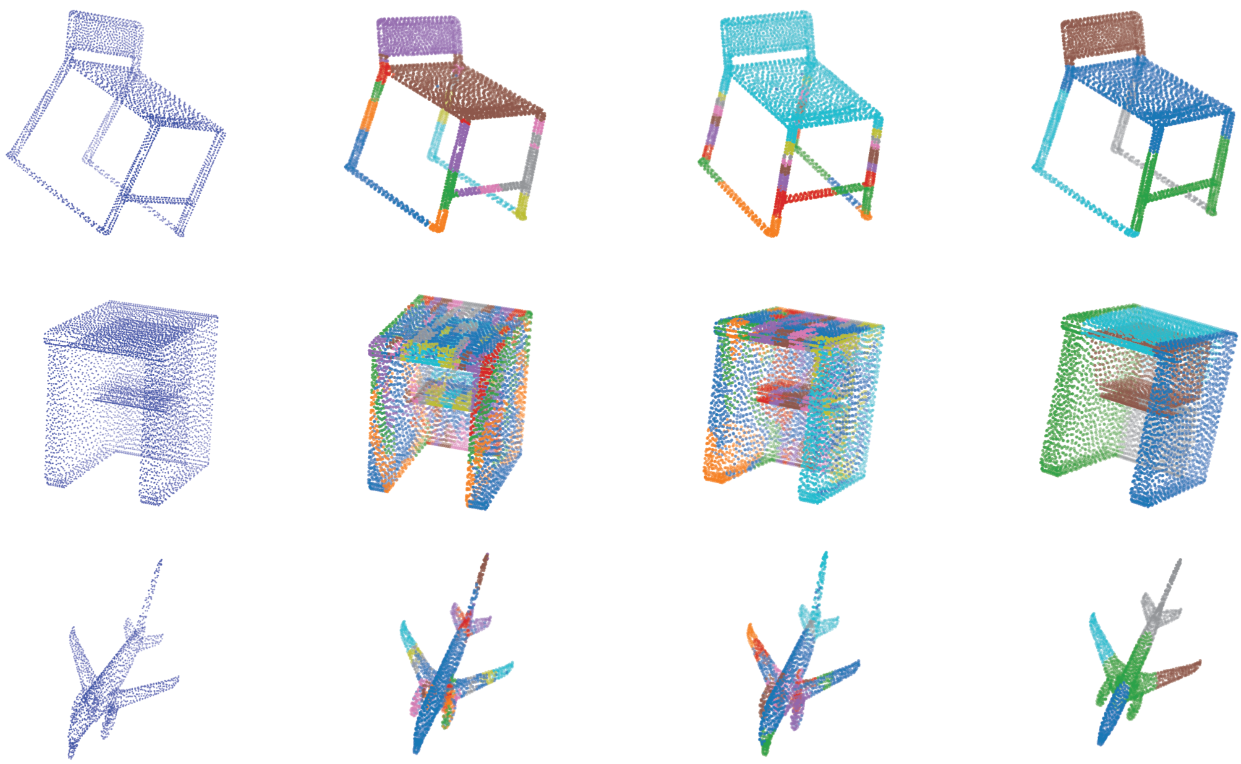

Figure 1: Different segmentation methods for chair (top row), desk (middle row), and plane (bottom row) for NeLo + Mapper (left column), Manual Mapper (middle column), and Spectral + k-means (right column).

- *Silhouette* [10] – quantifies how well each point lies within its cluster compared to other clusters. In the context of 3D point clouds, it gives an idea of how distinct and well-separated the resulting segments are. It does not require ground truth labels, captures both intra-cluster compactness and inter-cluster separation, and is applicable to both spectral and Mapper-based segmentation results.

- *Davies-Bouldin index (DBI)* [4] – quantifies the compactness and separation of clusters. In the context of 3D point cloud segmentation, it provides a way to measure how well the points are grouped into segments. It does not require truth labels, balances both compactness and separation, and is simple to compute.

- *Calinski-Harabasz index (CHI)* [2] – also known as the Variance Ratio Criterion. It is a metric for evaluating clustering quality. It measures how well-separated and compact clusters are, similar to the Silhouette and DBI, but it uses a variance-based formulation.

We showcase three informative examples, which help demonstrate the approach's strengths and weaknesses.

We set $k = 5$ clusters for spectral clustering. For Mapper, we used $m = 4$ filter functions for the plane and desk, and $m = 3$ for the chair, with `n_cubes` $= 10$, `perc_overlap` $= 0.3$, and DBSCAN parameters $\varepsilon = 0.05$ and `min_samples` $= 10$.

The quantitative results are presented in Table 1, and the qualitative results are shown in Figure 1. We first notice that the number of clusters produced with the Mapper method is very large, even if it might not seem like it in the figures. This happens because Mapper clusters each filter hyperspace separately instead of clustering the whole filter space at once. This is not always bad, as it allows for finding finer segmentations as demonstrated in the plane example, where it distinguishes between the plane's motors and hull and even identifies different parts of the rear wing in the case of NeLo Laplacian. We also notice that the metrics for Mapper methods are significantly worse than the k-means method. This is a consequence of the phenomena described above. The huge number of clusters significantly worsens the metrics, although the visual results are not so bad.

When we compare NeLo and Manual Mapper methods, we see that the NeLo method is capable of producing a finer segmentation than Manual Mapper. We see this in the chair example where it differentiates between the backrest and the seat. Lastly, we notice that the method does not work well on objects that have no distinct features that would be distinguishable with the Laplacian operator. This can be seen in the desk example, where the Mapper methods find clusters with no meaning. When comparing quantitative measures, we see that they perform relatively similarly, with one performing better than the other in some metrics and vice versa.

5 Conclusion

In this work, we presented a novel approach to 3D point cloud segmentation by combining NeLo with the Mapper algorithm. By leveraging NeLo's learned spectral embeddings as filter functions, we transformed Mapper into an end-to-end framework that eliminates the need for manual filter selection. Our experiments on ShapeNet [3] showed finer segmentations compared to spectral clustering and manual Laplacian baselines, especially for objects with distinct geometric features. Limitations remain for objects with less pronounced structure and in Mapper's tendency to over-segment. The code and experiments are available on GitHub [2].

Future work includes adaptive parameter tuning, smarter cluster merging, and extending evaluation with supervised baselines such as PointNet++ [8] and PoinNeXt [9], which would provide a stronger reference point against state-of-the-art methods.

Acknowledgment

This research was conducted as part of the basic research project *Cell visualization of unified microscopic data and procedurally generated sub-cellular structures* [project number J2-50221], funded by the Slovenian Research and Innovation Agency (Javna agencija za znanstvenoraziskovalno in inovacijsko dejavnost RS) from the state budget.

References

[1] Mario Botsch, Léif Kobbelt, Mark Pauly, Pierre Alliez, and Bruno Lévy. *Polygon Mesh Processing*. AK Peters/CRC Press, 2010.

[2] T. Calinski and J. Harabasz. A dendrite method for cluster analysis. *Communications in Statistics*, 3(1):1–27, 1974.

[3] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015.

[4] David L Davies and Donald W Bouldin. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-1(2):224–227, 1979.

[5] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. M. Stuart, and A. Anandkumar. Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*, 2020.

[6] Bo Pang, Zhongtian Zheng, Yilong Li, Guoping Wang, and Peng-Shuai Wang. Neural laplacian operator for 3d point clouds. *ACM Transactions on Graphics*, 43(6):1–14, 2024.

[7] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 77–85, 2017.

[8] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[9] Yujing Qian, Meng-Hao Xu, Zhiyuan Zhang, Enze Xie, Wenhai Wang, Yulu Wei, Zhenguo Li, Chen Change Loy, and Bo Dai. Pointnext: Revisiting pointnet++ with improved training and scaling strategies. In *Advances in Neural Information Processing Systems*, 2022.

[10] Peter J Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, 1987.

[11] Erich Schubert, Jörg Sander, Martin Ester, Hans Peter Kriegel, and Xiaowei Xu. Dbscan revisited, revisited: why and how you should (still) use dbscan. *ACM Transactions on Database Systems (TODS)*, 42(3):1–21, 2017.

[12] Gurjeet Singh, Facundo Mémoli, and Gunnar Carlsson. Topological methods for the analysis of high dimensional data sets and 3d object recognition. In *Eurographics Symposium on Point-Based Graphics*, pages 91–100. The Eurographics Association, 2007.

[2] https://github.com/BlazBulic/DP-Project-NeLo-Mapper