Projection Ensemble: Visualizing the Robust Structures of Multidimensional Projections

Myeongwon Jung[*] Jiwon Choi[†] Jaemin Jo[‡]

Sungkyunkwan University

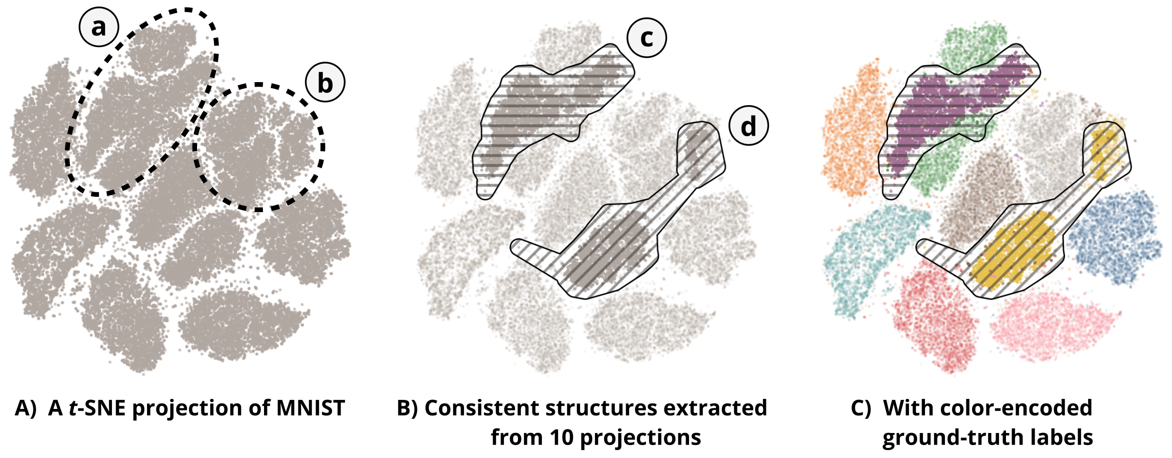


Figure 1: Projection Ensemble recognizes robust structures in multidimensional projections. **A)** A randomly initialized *t*-SNE projection of the MNIST dataset generated by stochastic gradient descent. The viewer may interpret groups of points (a) and (b) as individual clusters, which, in fact, have intricate intra- or inter-cluster relationships. **B)** Projection Ensemble visualizes two robust structures identified by extracting common subgraphs between ten randomly initialized projections; these are the structures that the ten projections "agree on." This reveals that (a) actually consists of two entangled structures (group (c) and the other points), and a subgroup of the cluster (b) is found to be closer to a distant cluster (see (d)). This can be done without using ground-truth class labels or clustering high-dimensional points. **C)** The ground-truth class labels are shown as the color of points.

Abstract

We introduce Projection Ensemble, a novel approach for identifying and visualizing robust structures across multidimensional projections. Although multidimensional projections, such as *t*-Stochastic Neighbor Embedding (*t*-SNE), have gained popularity, their stochastic nature often leads the user to interpret the structures that arise by chance and make erroneous findings. To overcome this limitation, we present a frequent subgraph mining algorithm and a visualization interface to extract and visualize the consistent structures across multiple projections. We demonstrate that our system not only identifies trustworthy structures but also detects accidental clustering or separation of data points.

Index Terms: Human-centered computing—Visualization—Visualization systems and tools

1 Introduction

Multidimensional projections (MDP) have been widely employed to visualize high-dimensional data in various fields, such as natural language processing [13], macromolecule analysis [4], and information visualization [12,22]. The non-linear MDP techniques compute low-dimensional representations of input points by minimizing a loss function that measures the difference between the distribution of the input points in the original high-dimensional space and that of the projected, low-dimensional representations. Since such a loss minimization process usually does not have a closed-form solution, many popular non-linear MDP techniquessuch as *t*-Stochastic Neighbor Embedding (*t*-SNE) [21] and Uniform Manifold Approximation and Projection (UMAP) [14]rely on an iterative optimization method, gradient descent (GD) [17], to find an approximate solution in a reasonable time.

One limitation of those MDP techniques is that they are not deterministic; they produce different projections each time they run. There are two main reasons behind such inconsistency. First, to make the optimization process more efficient in practice, GD is often replaced with stochastic gradient descent or mini-batch gradient descent. In contrast to GD where all data points are considered in each iteration of the process, these methods shuffle the order of input data and update the projection according to the gradient of one (stochastic) or a fixed number (mini-batch) of points. Therefore, the result can vary depending on the order in which the input points are processed. Second, to initiate the optimization process, the initial position of each data point in the 2D projection must be determined. This position is often randomly assigned, introducing another random factor into the process.

The stochastic nature of non-linear MDP techniques raises an important question: which parts of a projection are robust across multiple runs, and which are merely a result of chance due to randomness? Previous methods to address this issue include 1) providing a fixed seed number to the random number generator and 2) creating the initial layout using a deterministic MDP technique such as

[*]e-mail: mw8244@g.skku.edu
[†]e-mail: jasonchoi3@g.skku.edu
[‡]e-mail: jmjo@skku.edu. Jaemin Jo is the corresponding author.

Principal Component Analysis (PCA) [8] so that the influence of the initial layout can be reduced. However, these ad-hoc approaches do not fully answer the question; they focus on making an MDP technique deterministic or less sensitive to randomness, but it does not necessarily mean that the structures revealed in the result are robust to stochasticity.

As our answer to the question, we introduce Projection Ensemble (Fig. 1), a novel approach that detects and visualizes the robust structures in MDPs. The main idea of Projection Ensemble is to generate multiple projections and identify structures that those projections commonly have. To this end, we first elaborate on an algorithm that elicits common structures from multiple projections and then present a user interface that visualizes the structures. Finally, our use case demonstrates that our approach not only confirms consistent structures across projections but also identifies less stable structures that arise due to chance.

2 Related work

Visual Analytics on MDP. Prior studies have introduced visual analytics systems to understand the inner workings of MDP techniques and prompt their reliability. For instance, Stahnke et al. [18] proposed a probing technique that allows the user to examine the distortion between the original multi-dimensional data and their projection. Similarly, t-viSNE [2] facilitates the interactive exploration of MDPs by allowing the user to inspect various aspects of the optimization process, such as the effects of hyperparameters and neighborhood preservation. Recently, Jeon et al. [9] proposed novel metrics, Steadiness and Cohesiveness, to measure the inter-cluster reliability of MDPs and a visualization technique to show the distortion of a projection.

While these systems focus on analyzing the relationship between the original and projected data points in a single projection, there also exist visual analytics systems that allow comparison between projections. For example, Compadre [5] enables the comparison of two different projections, employing a matrix-based visualization technique. Another example is EvoSets [19], which quantifies and visualizes changes happening in MDPs when particular attributes are added or removed. In this work, however, we are interested in structural changes due to randomness rather than differences in MDP techniques or data. Our system not only allows for the comparison of over twenty projections but also locates consistent structures across the projections.

Frequent Subgraph Mining. In this work, we solve the problem of finding the consistent structures between MDPs inspired by the Frequent Subgraph Mining (FSM) problem [10]. The goal of FSM is to identify all subgraph structures that appear in different graphs simultaneously. Specifically, when a set of graphs $\mathcal{G} = \{G_1, G_2, ..., G_N\}$ and a minimum support threshold *minsup* $(minsup \leq N)$ are given, FSM is to find a set of frequent subgraphs, denoted as $\mathcal{F}$, which is defined as follows:

$$sup_{\mathcal{G}}(g) = \|\{G_i \mid g \subseteq G_i\}\|$$

$$\mathcal{F} = \{g \mid sup_{\mathcal{G}}(g) \geq minsup\}$$

FSM algorithms have been widely used across various domains to detect co-occurring patterns between multiple targets, such as in chemistry [7], bioinformatics [3], computer vision [1], and visual analytics [6]. In this paper, we present a relaxed version of FSM along with an efficient algorithm that finds the largest non-overlapping components among these frequent subgraphs.

3 Projection Ensemble

In this section, we present Projection Ensemble, a system designed to capture and visualize robust structures found in multiple MDPs. We begin by introducing our algorithm that identifies these structures, inspired by the FSM problem. We then elaborate on the design of visualization and user interaction.

Algorithm 1 Relaxed version of Frequent Subgraph Mining

```
Input: a graph set $\mathcal{G} = \{G_1, G_2, \ldots, G_k\}$, minimum support $minsup$
Output: a frequent subgraph set $\mathcal{F}$
1: $U \leftarrow \bigcup_{G_i \in \mathcal{G}} G_i$
2: for all $e \in E(U)$ do
3:     $support(e) \leftarrow$ CountOccurrences$(e, \mathcal{G})$
4:     if $support(e) < minsup$ then
5:         $E(U) \leftarrow E(U) \setminus e$
6:     end if
7: end for
8: $\mathcal{F} \leftarrow$ GetDisconnectedGraphs$(U)$
9: return $\mathcal{F}$
```

3.1 Generating a Projection Set

We search for clusters of data points that consistently appear together across multiple projections, indicating that they likely represent existing structures in the input data, not being generated by chance. To this end, we first define **a projection set** that consists of multiple MDPsof the same input data. By default, we generate a projection set of ten randomly initialized t-SNE projections that only differed in their initial layout. Note that the projections can be generated using different hyperparameters depending on the user's interests. For example, to compare the impact of hyperparameters, one can use a set of t-SNE projections initialized with different hyperparameters.

3.2 Building kNN Graphs

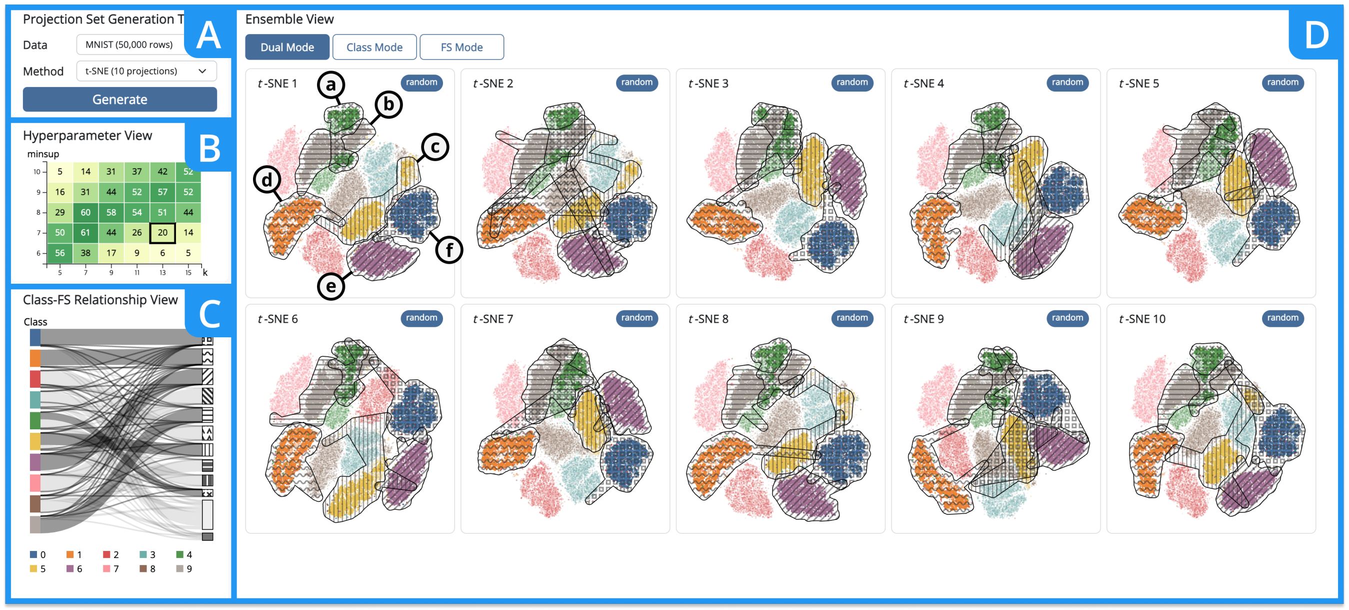
In the second step, we derive **a graph set** from a projection set by converting each projection in the projection set to an undirected k-Nearest Neighbors (kNN)graph by linking every point to its k nearest neighbors in the 2D projection space. Note that the kNN graphs built here are different from the kNN graphs used for computing projections, e.g., for Barnes-Hut Approximation [20]; the former is built for projected 2D data points, while the latter are constructed for the high-dimensional original data points.

3.3 Relaxation of Frequent Subgraph Mining

We find consistent subgraph structures between kNN graphs by formulating it as a relaxed version of the FSM problem. FSM is a problem of identifying subgraphs that occur more than a specified threshold in input graphs set [10]. However, we found that previous algorithms for FSM are not suitable for interactive scenarios as their computational cost is too high.

To speed up the computation, we relaxed the constraints of the original problem in two ways: first, the original FSM problem assumes that all nodes are unlabeled , so it is difficult to check if two subgraphs structures are isomorphic, which is indeed an NP problem. However, we found that this constraint could be relaxed in our case since each node in a kNN graph corresponds to a data item with their identity known, allowing us to detect graph isomorphism in a polynomial time; more precisely, our dataset consists of graphs with identical nodes but different links, meaning that we can check for graph isomorphism by simply checking if every link in a graph is present in the other graph without performing a node matching.

The second relaxation we made pertains to the determination of frequent subgraphs. Strictly speaking, a subgraph can be said to be frequent if it appears in all (or most of) graphs in $\mathcal{G}$. However, we found such a constraint too strict, resulting in many small and fragmented subgraphs. Instead, we apply such a constraint on a link basis; a link is frequent if it exists in at least *minsup* graphs of $\mathcal{G}$ where *minsup* is a **minimum support** specified by the user, and a subgraph is frequent if every link is frequent. This relaxation of the constraint on frequent subgraphs allows for a more efficient incremental algorithm, as we no longer require that a subgraph appears as a whole.



Projection Set Generation Tab

Data: MNIST (50,000 rows)

Method: t-SNE (10 projections)

minsup \ k	5	7	9	11	13	15
10	5	14	31	37	42	52
9	16	31	44	52	57	52
8	29	60	58	54	51	44
7	50	61	44	26	20	14
6	56	38	17	9	6	5

Figure 2: The Projection Ensemble interface. (A) Projection Set Generation Tab, (B) Hyperparameter View, (C) Class-FS Relationship View, and (D) Ensemble View.

Our algorithm for solving the relaxed version of FSM is outlined in Algorithm 1. Given the input graph set $\mathcal{G}$ from the second step, the algorithm computes a set of disjoint frequent subgraphs, $\mathcal{F}$. The algorithm starts with creating a union graph that contains all the nodes and links from the input graphs. Next, we count the frequency of each link in the union graph and prune each if it has a frequency lower than *minsup*. After pruning infrequent links, the union graph is typically split into disconnected components, and each component becomes a frequent subgraph.

3.4 Visualizing Frequent Subgraphs

There are two possible outcomes of FSM for each data point: 1) it can be assigned to a frequent subgraph by having the index of the subgraph stored in its `FS_id` attribute, or 2) it can be considered an outlier and have an `FS_id` value of 0. If there is no class information in the data, the categorical `FS_id` attribute can be overlaid on a conventional scatterplot by encoding it as the color of a point. However, when a class label (hereafter, `class`) is already present in the data, such as in the case of the digit attribute in the MNIST data [11], visualizing both categorical attributes on the same scatterplot can be challenging.

We decided to show `class` as the color (hue) of a point and `FS_id` as textured contours where data points belonging to the same subgraph are contained in a contour, as shown in Fig. 1C. We tested different combinations of identity channels to show the two categorical attributes on a scatterplot, such as using concentric circles with two hues, color and shape, and animated transitions, most of which resulted in overplotting and visual clutter. We could also apply the encoding in the opposite way, i.e., mapping `class` to textured contours and `FS_id` to color, but we found the current encoding more effective for two reasons; 1) encoding `class` as the color of a point is more familiar to the user, and 2) points belonging to the same frequent subgraph are more likely to be close to each other in a projection compared to points belonging to the same class, making them more suitable for being contained within a concise contour.

With the visual encoding discussed, we present the Project Ensemble interface where the user can interactively conduct and analyze the robustness of MDPs(Fig. 2). The user interface consists of four components. The first **Projection Set Generation Tab** (Fig. 2A) allows the user to specify the data to analyze and the configuration of a projection set. By default, we generate a projection set of ten *t*-SNE projections. After choosing the data and configuration, the user can initiate the FSM process by clicking on the "Generate" button.

The number of frequent subgraphs differs according to the values of two hyperparameters, the number of neighbors in the *k*NN graph (*k*) and the minimum support (*minsup*), which the user specifies. To facilitate the hyperparameter selection process, we employ a grid search approach; we chose five and six candidates for the two hyperparameters, respectively, and run the mining process for every pair of candidates. The result is shown as a heatmap in **Hyperparameter View** (Fig. 2B) where the number of frequent subgraphs for each pair of hyperparameters is shown and color-encoded in a cell. The user can click on a cell to see the result of a specific pair.

Class-FS Relationship View (Fig. 2C) visualizes the relationship between the given class labels (`class`) and frequent subgraphs (`FS_id`) as a Sankey diagram. On the left, classes are depicted as colored bars with each having height proportional to the number of points in the corresponding class. On the right, frequent subgraphs are shown as textured bars again with the height proportional to the number of nodes in the corresponding subgraphs. We assign distinct textures to the top ten largest subgraphs, and the other relatively small subgraphs are aggregated to the 11th light gray bar without textures. Outlying points, points that do not belong to any frequent subgraph, are shown as the last dark gray bar. The thickness of the link between two bars represents the number of points shared between a class and a frequent subgraph. The colors and textures of bars are consistently used throughout the interface.

In **Ensemble View** (Fig. 2D), we visualize each projection in the generated projection set as a scatterplot with the color and texture encoding discussed above. The projections are aligned by applying Procrustes transformation [16] to provide spatial coherence. The Ensemble View is linked with the Class-FS Relationship View, allowing the user to highlight a certain class or subgraph by hovering the cursor over it or toggle its visibility by clicking on it.

The source code of Projection Ensemble is available at `https://github.com/jjmmwon/ProjectionEnsemble`.

4 Evaluation

We present a use case to show how our system can reveal consistent/inconsistent structures between randomly initialized t-SNE projections.

Dataset and Settings. As an input dataset, we used 50,000 28x28 images of the MNIST dataset [11], with 5,000 images representing each digit. Each image was encoded as a 784-dimensional vector. We generated a projection set of ten t-SNE projections using the same t-SNE hyperparameters ($perplexity = 45$ and the default values for the rest defined in the scikit-learn library [15]). Each projection was initialized randomly and optimized by stochastic gradient descent, resulting in slightly different projections, as illustrated in Fig. 2D. The Hyperparmeter View (Fig. 2B) shows the number of the generated frequent subgraphs for each hyperparameter combination of k and *minsup* used in our FSM algorithm. We set k and *minsup* to 13 and 7, respectively, which resulted in 20 frequent subgraphs.

Revealing Intra-Cluster Consistency. The results of our system are presented in Fig. 2. It is worth noting that the ground-truth labels are color-encoded in the figure to facilitate the comparison between the ground-truth structures and the insights we gained from our system; they are not used in our algorithm. We first observed that our system could reveal consistent intra-cluster structures in a projection. For example, in the first projection in Fig. 2D, subgraphs (a) and (b) appeared to form a large cluster if no class or subgraph information was given. Our system was able to provide additional detail about this cluster, highlighting that it was actually composed of two “entangled” structures, subgraphs (a) and (b). Indeed, it turned out that subgraph (a) contained images of digit ‘4’ while subgraph (b) was made up of images of digit ‘9’, which looked similar in the pixel space.

Another interesting observation is that despite points in subgraph (a) being separated into two groups (upper and lower) in eight out of the ten projections , the system was able to recognize that those groups belong to the same subgraph. This could be possible because the algorithm uses multiple projections to identify structures that are consistent across projections. Therefore, even if a particular projection, e.g., the first projection in Fig. 2D, separates two points into different groups, the algorithm can still recognize that they belong to the same subgraph if they are consistently grouped together in other projections, providing robustness over multiple runs. Fig. 2D suggests that the separation of the subgraph (a) in the first projection may appear due to the stochastic nature of t-SNE

Revealing Inter-Cluster Consistency. We also found out that our system could reveal the consistent structures between clusters, i.e., inter-cluster consistency. For example, in Fig. 2D, subgraph (c) indicates that two distant groups of points that seemingly belong to different clusters are, in fact, connected. We observed this type of separation in four out of ten projections, suggesting that the separation is inconsistent. Interestingly, we also observed that when such separation occurred, another cluster was found to be located between the two groups. This implies that the images within that cluster are likely to be similar to those in the two separated groups. Confirming these insights, the ground-truth labels showed that the two groups in subgraph (c) contained images of the same digit ‘5’, while the cluster between them consisted of images of the digit ‘3’, which were similar to the images of ‘5’ in the pixel space.

Confirming Consistent Clusters. Our system is not only capable of identifying less stable cluster structures such as subgraphs Fig. 2D (a)-(c) but also of discovering consistent clusters across projections. For example, in Fig. 2D, clusters containing subgraphs (d) and (e) are sorely in a contour, indicating that the cluster structures are consistent across multiple runs. Indeed, those clusters mostly contained images of a single digit, ‘1’ and ‘6’, respectively.

Another interesting observation that could be made is about subgraph (f) in Fig. 2D. Similar to (d) and (e), it exhibited a single consistent cluster. If ground-truth labels were available, one would observe that the subgraph primarily consists of images of ‘0’ but also includes images of other digits. While one might suspect these images of non-zero digits as artifacts resulting from the optimization process, our system can verify that they are not artifacts and constantly appear alongside images of zeroes; in fact, as shown in Fig. 3, the images of non-zero digits in subgraph (f) resemble zeroes.

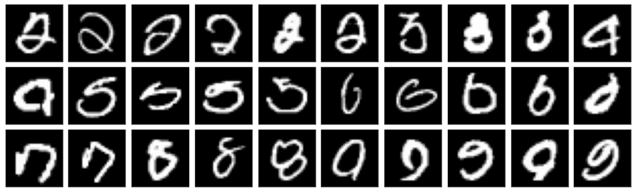


Figure 3: The images of non-zero digits in subgraph (f) in Fig. 2. They actually resemble zeroes, suggesting that the images of non-zero digits in subgraph (f) are not an artifact of an MDP technique.

Effect of Hyperparameters. Fig. 2B exhibits the effect of the two hyperparameters, k and *minsup*, on the number of frequent subgraphs identified. A higher value of k and a lower value of *minsup* result in a union graph with more edges, where it is easier for points to be grouped as frequent subgraphs. As a result, a small number of relatively large subgraphs are identified. Conversely, a lower value of k and a higher value of *minsup* make it much harder for points to be grouped, resulting in a small number of tiny subgraphs, leaving most points as outliers. We found that the number of frequent subgraphs increases when k and *minsup* strike a balance; empirically, we found that having approximately twenty subgraphs with the top ten subgraphs including 80% of points in total is a good starting point for analysis.

Summary. Our use case shows that relying on a single projection can lead to misunderstandings of the underlying structure of the data. It also demonstrates how such a limitation could be complemented by the identification and visualization of robust structures across multiple projections.

5 Conclusion and Future Work

In this paper, we introduce Projection Ensemble, a novel system that identifies and visualizes consistent structures across MDP. Depending on the initial conditions and the stochastic nature of non-linear MDP algorithms, MDPs can include noisy structures that arise accidentally. Thus, relying on a single projection may lead to misunderstandings of the underlying structure of the data. Our system addresses this issue by identifying and visualizing consistent structures in multiple projections, which are less likely to be the result of random noise.

There are several directions for future work. First, although we used kNN to convert a projection to a graph, further exploration of various graph generation techniques is needed. Additionally, our system can be extended to assess the consistency of other MDP techniques, such as UMAP [14]. Lastly, the current textured contour generation algorithm may lead to visual clutter and scalability challenges, particularly in cases with many outliers, which requires further investigation and improvement.

Acknowledgments

This work was partly supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2019-0-00421, AI Graduate School Support Program (Sungkyunkwan University)).

References

[1] N. Acosta-Mendoza, A. Gago-Alonso, and J. E. Medina-Pagola. Frequent approximate subgraphs as features for graph-based image classification. *Knowledge-Based Systems*, 27:381–392, 2012.

[2] A. Chatzimparmpas, R. M. Martins, and A. Kerren. t-visne: Interactive assessment and interpretation of t-sne projections. *IEEE Transactions on Visualization and Computer Graphics*, 26(8):2696–2714, 2020.

[3] Q. Chen, C. Lan, B. Chen, L. Wang, J. Li, and C. Zhang. Exploring consensus rna substructural patterns using subgraph mining. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 14(5):1134–1146, 2016.

[4] J. Choi, H.-J. Oh, H. Lee, S. Kim, S.-K. Kwon, and W.-K. Jeong. Mitovis: A unified visual analytics system for end-to-end neuronal mitochondria analysis. *IEEE Transactions on Visualization and Computer Graphics*, 2023.

[5] R. Cutura, M. Aupetit, J.-D. Fekete, and M. Sedlmair. Comparing and exploring high-dimensional data with dimensionality reduction algorithms and matrix visualizations. In *Proceedings of the International Conference on Advanced Visual Interfaces*, pp. 1–9, 2020.

[6] Z. Deng, D. Weng, J. Chen, R. Liu, Z. Wang, J. Bao, Y. Zheng, and Y. Wu. Airvis: Visual analytics of air pollution propagation. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):800–810, 2019.

[7] M. Deshpande, M. Kuramochi, N. Wale, and G. Karypis. Frequent substructure-based approaches for classifying chemical compounds. *IEEE Transactions on Knowledge and Data Engineering*, 17(8):1036–1050, 2005.

[8] H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24(6):417, 1933.

[9] H. Jeon, H.-K. Ko, J. Jo, Y. Kim, and J. Seo. Measuring and explaining the inter-cluster reliability of multidimensional projections. *IEEE Transactions on Visualization and Computer Graphics*, 28(1):551–561, 2021.

[10] C. Jiang, F. Coenen, and M. Zito. A survey of frequent subgraph mining algorithms. *The Knowledge Engineering Review*, 28(1):75–105, 2013.

[11] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[12] G. Li and X. Yuan. Gotreescape: Navigate and explore the tree visualization design space. *IEEE Transactions on Visualization and Computer Graphics*, 2022.

[13] X. Li, J. Thickstun, I. Gulrajani, P. S. Liang, and T. B. Hashimoto. Diffusion-lm improves controllable text generation. *Advances in Neural Information Processing Systems*, 35:4328–4343, 2022.

[14] L. McInnes, J. Healy, and J. Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[15] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

[16] A. Ross. Procrustes analysis. *Course Report, Department of Computer Science and Engineering, University of South Carolina*, 26:1–8, 2004.

[17] S. Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[18] J. Stahnke, M. Dörk, B. Müller, and A. Thom. Probing projections: Interaction techniques for interpreting arrangements and errors of dimensionality reductions. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):629–638, 2015.

[19] G. Sun, S. Zhu, Q. Jiang, W. Xia, and R. Liang. Evosets: tracking the sensitivity of dimensionality reduction results across subspaces. *IEEE Transactions on Big Data*, 8(6):1566–1579, 2021.

[20] L. Van Der Maaten. Barnes-hut-sne. *arXiv preprint arXiv:1301.3342*, 2013.

[21] L. Van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(11), 2008.

[22] Y. Ye, R. Huang, and W. Zeng. Visatlas: An image-based exploration and query system for large visualization collections via neural image embedding. *IEEE Transactions on Visualization and Computer Graphics*, 2022.