Visual Analysis of Class Separations with Locally Linear Segments

Yuxin Ma and Ross Maciejewski, *Senior Member, IEEE*

Abstract—High-dimensional labeled data widely exists in many real-world applications such as classification and clustering. One main task in analyzing such datasets is to explore class separations and class boundaries derived from machine learning models. Dimension reduction techniques are commonly applied to support analysts in exploring the underlying decision boundary structures by depicting a low-dimensional representation of the data distributions from multiple classes. However, such projection-based analyses are limited due to their lack of ability to show separations in complex non-linear decision boundary structures and can suffer from heavy distortion and low interpretability. To overcome these issues of separability and interpretability, we propose a visual analysis approach that utilizes the power of explainability from linear projections to support analysts when exploring non-linear separation structures. Our approach is to extract a set of locally linear segments that approximate the original non-linear separations. Unlike traditional projection-based analysis where the data instances are mapped to a single scatterplot, our approach supports the exploration of complex class separations through multiple local projection results. We conduct case studies on two labeled datasets to demonstrate the effectiveness of our approach.

Index Terms—Visual analysis, dimension reduction, class separation

1 Introduction

High dimensional data visualization is utilized in various fields such as machine learning [1], [2], biology [3], and physics [4]. One of the main goals for analyzing high-dimensional datasets is to reveal hidden patterns and relationships. In this paper, we focus on analyzing class separations of high-dimensional labeled data. The concept of labels is a fundamental building block, widely adopted in machine learning tasks including classification and clustering. Class labels are used for specifying categories or partitions of data instances and are usually tagged by human annotators or predicted by models [5], [6]. When analyzing high-dimensional data, analysts often explore how the classes of instances are distributed or separated from each other, which can support model selection, result evaluation and parameter tuning.

To visualize high-dimensional patterns with labels, dimension reduction is an essential approach used to preserve the intrinsic structures in the data while providing a low dimensional representation. Colored scatterplots generated by dimension reduction algorithms are a natural form for presenting class structures. Two properties are typically considered when applying an algorithm to visualize labeled data instances: linearity and capability of supervision [7]:

- In linear projections, the data characteristics, such as covariance, correlations, input-output relations, and margins between classes, can be easily illustrated [2]. Nevertheless, they lack the ability to extract non-linear manifolds [8]. In this situation, the linear projection results in a colored scatterplot with heavily-overlapped points in different colors. The non-linear methods are able to present non-linear manifolds in a 2-D scatterplot. However, the embedding results can be "extremely difficult to interpret" [9] and may cause distortions and errors, such as missing or false neighboring relationships, deformation of distances, and loss of feature importance [7].
- The capability of supervision decides whether a method can make use of the class labels. Supervised dimension reduction algorithms tend to increase the similarities of instances in the same class, which is reflected by closer distances between dots with the same color. Generally, supervised methods outperform their unsupervised counterparts in illustrating within class distributions by modifying the distance measures, resulting in shorter distances between instances in the same class and longer distances for instances in distinct classes [7], [10]. However, the distortion may increase the uncertainty and errors between instances from different classes and affect the understanding of intra-class relationships, especially when inspecting data distributions near the class separations.

These limitations are illustrated using a synthetic dataset in Figure 1. Here, we show a synthetic dataset in three-dimensional space (A) where two C-shaped groups of instances are colored in red and blue and placed face-to-face in an orthogonal way. Since the two entangled classes are not linearly separable, it is difficult to find a linear projection matrix that depicts a clear class boundary, indicating the limited capability of linear dimension reduction methods to handle non-linear structures. For non-linear methods, both (B) the unsupervised t-SNE [11] and (C) the supervised S-Isomap [10] algorithms lose the original axis information and distort the spatial relationships between instances in the 3-D feature space. Although the class structures can be perceived, it loses useful information to interpret how and why the two classes are separated, such as determining

- *Yuxin Ma and Ross Maciejewski are with The School of Computing, Informatics & Decision Systems Engineering, Arizona State University. E-mail: {yuxinma,rmacieje}@asu.edu.*

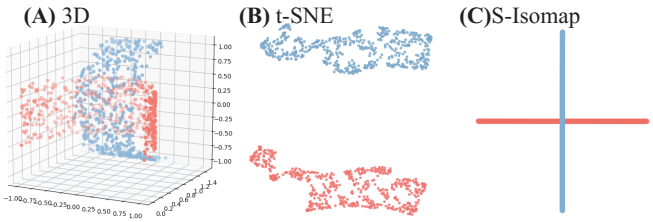


Fig. 1. An example of linear and non-linear projections for a 3D synthetic dataset. (A) Two classes of C-shaped data points face each other. (B) There is a clear separation in the t-SNE result; however, the relationships of spatial distances are heavily distorted. (C) The two classes cannot be separated in the S-Isomap projection result.

features to separate the classes, distances between classes, and fuzziness near the separation. As such, new approaches are needed to support the investigation of class boundaries.

As such, we propose a visual analysis approach that enables the exploration and investigation of complex class separation structures. The core idea of our approach is inspired by the development of locally-linear models in classification analysis [12], [13], [14], [15], [16], [17]. We characterize the complex class separation on the basis of locally linear segments that approximate the original separation. Notably, we detect potential linearly-separable regions near the class separation boundaries and build the classification boundary with locally linear segments by utilizing neighborhood graphs as a description of segment connectivity. We employ a multi-level visualization method consisting of a glyph-based *segment relation view*, a *segment detail view*, a *projection view*, and a *path exploration view*. The glyph-based segment relation and detail view summarize the detected locally linear segments as well as their relationships. The projection view is used for visualizing details of critical patterns including data distributions and clusters. Additionally, we design a set of affiliated views and interactions to support the visual exploration of locally linear segments. Our contributions include:

- A novel approach for detecting locally linear separations in high-dimensional labeled datasets with complex class boundary structures, and;
- A visual analysis framework that facilitates the exploration and diagnosis of complex class boundaries.

2 Related Work

High-dimensional data visualization is a broad topic that has been explored for decades [9]. We review related existing work in dimension reduction techniques, visual analysis in dimension reduction, and the analysis of high-dimensional labeled data.

2.1 Dimension Reduction Techniques

Dimension reduction techniques are often used to transform data instances from high-dimensional feature space into a low-dimensional space. Nonato and Aupetit [7] proposed a comprehensive survey and classification scheme of dimension reduction techniques in the context of visualization and visual analytics. Among the properties surveyed, we mainly focus on the property of "Linearity" which is based on the assumption of underlying structures in the projection algorithms. Linear projection techniques apply linear transformations to the high-dimensional data while preserving patterns and information hidden in the data [2], including Principal Component Analysis (PCA) for highest variances and Linear Discriminant Analysis (LDA) for maximizing class separations. Non-linear projection techniques, on the other hand, assume that the high-dimensional data instances are distributed in latent manifolds [8] and utilize neighborhood structure to preserve the patterns [9], [18], such as neighborhood graphs [19], [20], [21] and joint distributions of neighborhoods in the input and the embedding spaces [11], [22].

There are trade-offs between the two categories of projection techniques [7], [9]. Non-linear techniques are adept at maintaining intrinsic manifold structures in the high-dimensional data; however, the results are difficult to interpret because of heavily-distorted original axes. The output of linear projections provides better interpretability of data distributions and class separations [2], [9]; however, non-linear structures are lost [7]. In our visual analysis approach, we utilize the interpretability from linear techniques and enhance their ability to present complex separation structures with a suite of coordinated views. Recent work by Wang et al. [6] is the closest in spirit to our work, where a perception-driven linear dimension reduction approach is designed to maximize visual separation measures [23], [24], [25]. Similar to all other linear methods, this approach still suffers from a less-optimal solution for complex separation structures distributed on the separations.

2.2 Visual Analysis in Dimension Reduction

Interactive steering and exploration of latent structures are often integrated into dimension reduction techniques. Sacha et al. [26] propose seven categories of scenarios adopted in existing dimension reduction visualization systems. Here, we summarize the techniques closely related to our work.

Pattern Exploration: For revealing the hidden patterns and manifolds, a novel structure-based metric [27] is proposed to characterize the manifolds effectively. Approaches designed in Liu et al. [28], Xia et al. [29], Zhou et al. [30], and Xia et al. [8] are dedicated to the preliminary analysis of correlations, clusters, or manifolds hidden in subspaces. Our framework supports preliminary analysis specifically for characterizing decision boundary manifolds by enhancing existing linear projection methods [31] with multiple coordinated views.

Optimization: Finding optimal projection results is one of the main topics in dimension reduction. Automated methods (including Grand Tour [32], Projection Pursuit [33], and Koren et al. [5]) find projections that maximize specific optimization goals such as pursuit indices, data coordinates, pairwise similarities, and clusters. Recently, Liu et al. [4] introduce the Grassmannian Atlas, a framework that supports the exploration of all informative linear projections, and Wenskovitch et al. [34] discuss the design decisions when combining dimension reduction and clustering algorithms in visualizing clusters. For interactive adjustment of projection viewports, TripAdvisor^{N-D} [35] utilizes the concept of "touring in high-dimensional space" and supports the users' identification and planning of navigation paths. A

follow-up work by Wang et al. [36] enhances the navigation scheme with generalized 3D subspaces. Unlike the existing techniques that optimize and interact with the projections globally, our approach adopts a pattern extraction algorithm that summarizes the decision boundaries by utilizing a series of optimal projections on important local regions as well as interactions for inspecting local regions and transitions between regions.

2.3 Analysis of High-Dimensional Labeled Data

Along with dimension reduction methods, related research focuses on the analysis of labeled high-dimensional data.

Model Assistance: Classifiers are useful for characterizing class separations in labeled datasets. Rodrigues et al. [37] present an image-based approach that maps the high-dimensional decision boundaries onto a 2-D plane highlighting determined and confused regions. The trustworthiness of the maps under various dimensional reduction methods is further evaluated [38]. In our approach, we use support vector machine models to support the extraction of linearly-separable regions near the decision boundary.

Topological Analysis: The topological-based approaches tend to describe core structures with summarized representations [39], such as topology representing graphs [40], generative Gaussian graphs [41], [42], and principal complex graphs [43]. Concerning decision boundary analysis, Melnik [44] utilizes connectivity graphs [45] to analyze decision region structures extracted from trained classifiers, and Ramamurthy et al. [46] adopt persistent homology inference of decision boundaries to characterize the complexity of decision boundaries. Inspired by the topological analysis approaches, we establish neighboring relationships between locally linear regions as a summarization of the entire decision boundary and support interactive exploration to disclose the relationships between linear projections to present the complex structure of non-linear separations.

3 Locally Linear Segment Extraction

Before illustrating the visual analysis framework, we first introduce the locally linear segment extraction algorithm which acts as a foundation for further visual exploration.

3.1 Motivation

Our approach is to characterize *class separations* in labeled datasets. In the field of machine learning, class separations have been intensively studied to optimize model building and evaluation in classification analysis [44], [46], [47]. When a classification model is trained on a specific training dataset, the *decision boundary* of this model can be outlined, which defines two or more distinct regions associated with different class labels in the input feature space. Although the classification models and decision boundaries are generally used for predicting class labels of unlabeled data, we utilize the concept of decision boundaries as a descriptive tool [44], [47], [48] in order to characterize the class separations that exist in the original training dataset.

We distinguish between linear and non-linear decision boundaries based on their geometric properties. A decision boundary with the shape of a hyperplane forms a linear boundary which can be extracted by linear classifiers. A hypersurface in the input feature space forms a non-linear boundary that can describe complex class separations. For the non-linear boundaries, our approach approximates the non-linear boundaries with a set of locally linear ones, which is inspired by previous works on local linearity analysis [12]. Instead of using a single non-linear hypersurface to segregate the feature space, locally linear methods build a set of linear hyperplanes that approximates the original non-linear separations between classes. Regarding the effective regions, the methods roughly fall into two categories: 1) global methods [14], [15], [16], [17], [49] where the proposed models work on the entire feature space, and 2) instance-based methods [13], [50] that build classifiers on individual data instances. Different from the predictive purpose across the existing works, our approach focuses on generating interpretations for describing the patterns in the datasets. For descriptive analysis, the LIME [51] method utilizes explainable models (e.g., linear classifiers) as local surrogates to interpret predictions on individual instances. Our approach goes beyond the instance-based local explanations by providing a global description of all the instances in the dataset to provide a comprehensive explanation of the class separation criteria on the entire dataset. In our paper, we use "*locally linear segments*" to represent the set of linear approximations extracted from the original decision boundary. A formal definition of locally linear segments will be introduced in the extraction algorithm.

3.2 Extraction Algorithm

The extraction of locally linear segments (Figure 2) consists of four steps: (1) extract seeds near the decision boundary, (2) generate locally linear segments, (3) merge similar segments, and (4) compute the coverage of segments.

Extract Seeds Near the Decision Boundary (Figure 2 (a)): First, we use classification models to find the data instances that are near the potential decision boundary. We refer to such instances as *seeds*. We choose the non-linear support vector machine classifier (non-linear SVM) as the model. The rationale is that, after the SVM model is trained, the set of data instances marked as support vectors are the ones closest to the decision boundary based on the definition of SVM. Thus, these support vectors can depict the entire non-linear decision boundary and serve as a set of seeds for constructing locally linear segments. It should be noted that it is possible to use other types of classifiers, e.g., random forests, naive Bayes classifiers, etc., to characterize the decision boundaries. In this case, we can build an SVM or multiple SVM models as surrogate models of the desired alternative classifier, which are then used to support the following steps.

The procedure for harvesting the seeds in each class is described in Figure 2 (a). For each class label l_i in m different classes $L = l_1, l_2, ..., l_m$, the data instances X_{l_i} of class l_i are separated from the rest of the dataset $X_{l_i}^c$ (the complement of X_{l_i}). Then, a corresponding gaussian-kernel SVM S_{l_i} is trained. Data instances within X_{l_i} are the positive training samples and $X_{l_i}^c$ are the negative samples. Thus, the support vectors in the trained S_{l_i} can be used to model the decision

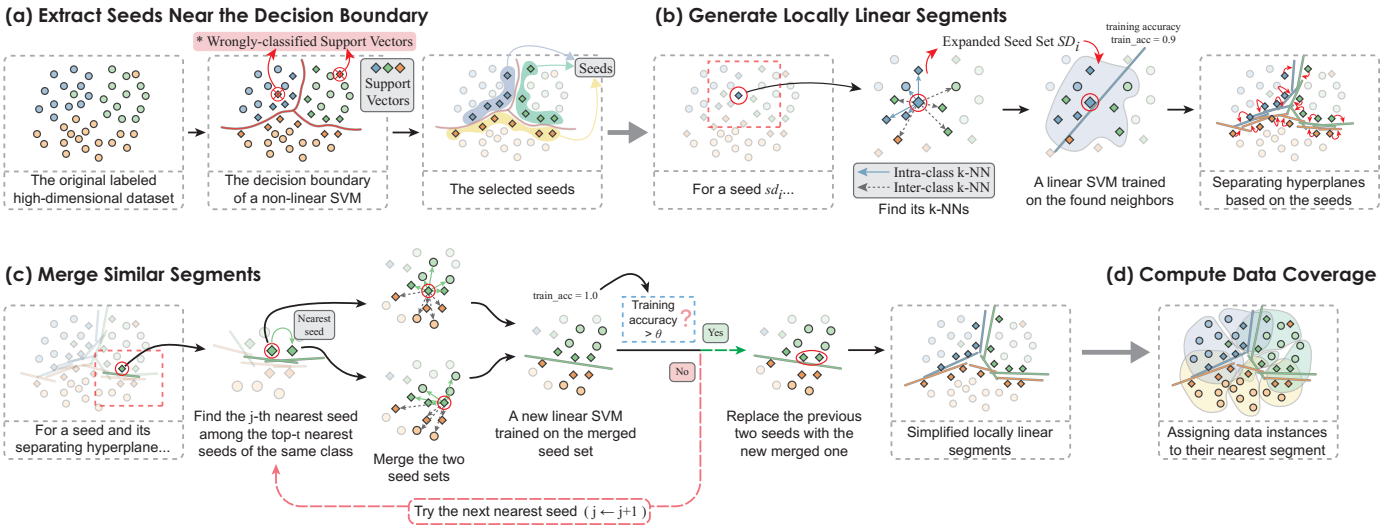


Fig. 2. The extraction process of locally-linear segments. (a) A number of data instances near the decision boundaries are filtered out as seeds. (b) For each seed, a linear SVM is trained on instances in the neighboring local region of the seed to generate a locally linear segment. (c) Similar segments are merged together. (d) The regions of the influence of the segments are computed.

boundary between the corresponding class and the rest of the data, and the support vectors in S_{l_i} are passed to the next step as seeds for generating locally linear segments. It should be noted that due to the soft-margin condition in SVMs, some of the support vectors may be assigned to the wrong class [52]. These misclassified support vectors are filtered out from the seeds.

Generate the Locally Linear Segments (Figure 2 (b)): In our algorithm, a locally linear segment is defined as a separating hyperplane that classifies instances in a certain region based on labels of the instances. Usually, the mathematical definition of a hyperplane implies that it expands indefinitely. However, here we address that its ability to properly classify labeled instances are limited to the local region near the corresponding seeds since it is only used as a local approximation of a complex decision hypersurface.

The segment generation process is inspired by the "local learning" strategy from Bottou et al. [53]. For each seed sd, we create an *expanded seed set* (SD) by selecting a number of surrounding data instances that covers a surrounding region. For each seed in class l_i, we select the $k-1$ nearest neighbors whose class label is l_i and another k nearest neighbors in classes other than l_i, resulting in an expanded seed set with $1+(k-1)+k=2k$ instances. A linear SVM is then trained on SD to find a separating hyperplane that can classify the data instances in the seed class and other classes. The hyperplane is used as the locally linear segment in the seed region.

Merge Similar Segments (Figure 2 (c)): After all the locally linear segments for all seeds are generated, another issue that comes from the properties of support vectors is that the density of seeds varies in different regions on the decision boundary. For those high-density regions, there may be redundant segments with nearly overlapped separating hyperplanes. To overcome this issue, we adopt a bottom-up scheme to merge redundant segments:

1) For each seed sd in class l and its expanded seed set SD, find the top-t nearest seeds sd'_j where $l'_j = l$ and $j \in [1,t]$ by measuring the average between-wise euclidean distances of data instances between their corresponding expanded seed sets.
2) With j from 1 to t, train a linear SVM on the set of data instances combined with SD'_j and SD, i.e., $SD'_j \bigcup SD$.
3) Calculate the training accuracy of the new linear SVM $train\_acc_j$.
4) If $train\_acc_j$ is larger than a threshold θ, the two segments generated by SD and SD'_j are considered to be overlapped and should be combined. The original sd and sd'_j will be replaced by $\{sd_j\} \bigcup \{sd\}$ whose expanded seed set is $SD'_j \bigcup SD$. The hyperplane associated with this new segment is updated to the separating plane from the new linear SVM. If $train\_acc_j < \theta$, the procedure will go back to step 2 by considering the next nearest segment, i.e., $j \leftarrow j+1$.

After all the t neighbors have been tried, the final SD with its derived separating hyperplane will be added to the final segment list.

Compute the Coverage of Segments (Figure 2 (d)): After the segments are extracted, we calculate the influence of the extracted segments on the surrounding instances. For each data instance that is not selected by any of the locally linear segments, we calculate the distance from the instance to the closest seeds in all seed sets of the segments. The instance is assigned to the closest segment. An accuracy value acc_{cover} is computed by counting the ratio of covered instances that can be correctly classified by the hyperplane of the segment, which reflects the fuzziness of the local separation.

4 Design Overview

Apart from the extraction algorithm, we performed a comprehensive survey of existing works on class separation and decision boundary analysis in machine learning, topological data analysis, and visualization. The main research challenges in the works can be summarized into two main categories: 1) macroscopic analysis of all decision boundaries

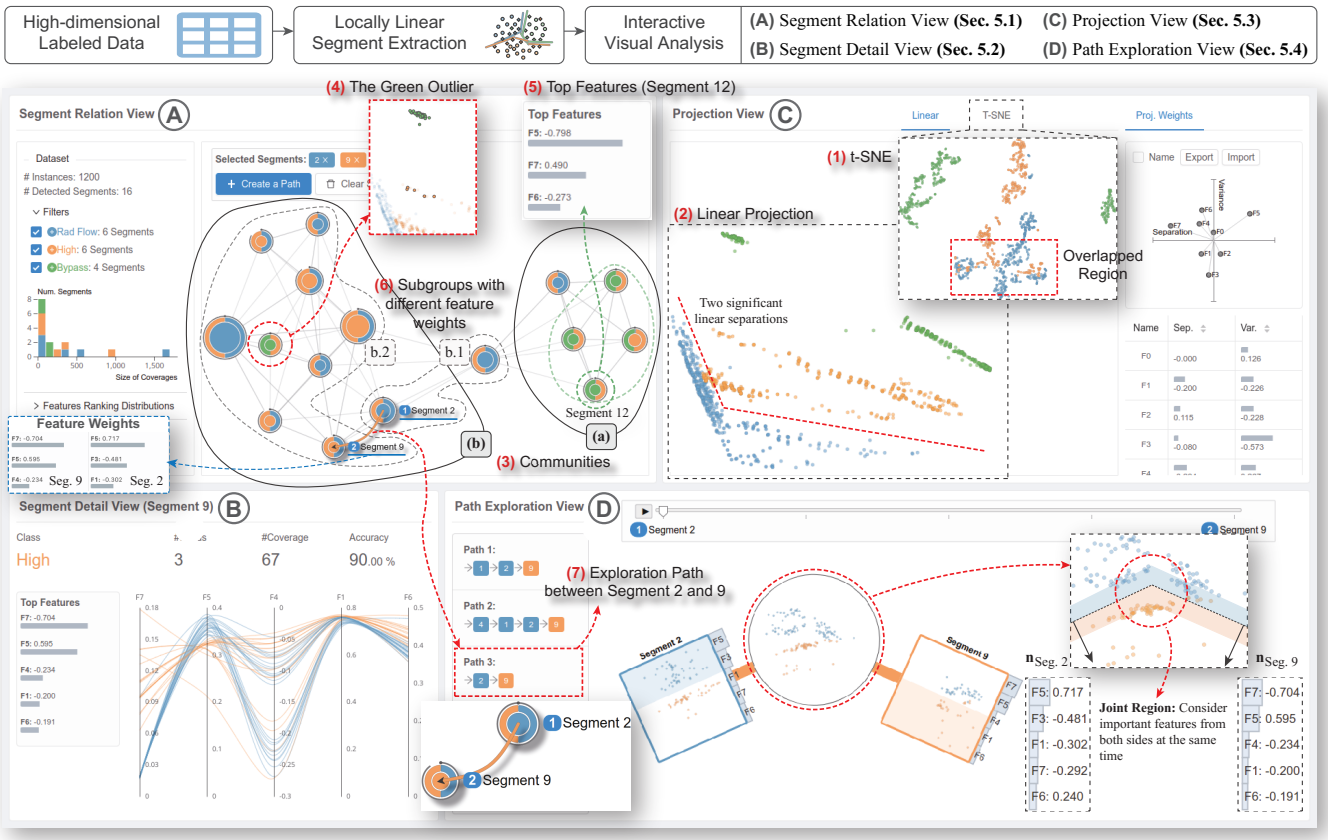


Fig. 3. (Top) An illustration of our visual analysis framework. (Bottom) The interface consists of four main views: (A) Segment Relation View. (B) Segment Detail View. (C) Projection View. (D) Path Exploration View.

and class partitions that provides an overview of the class structures [15], [16], [17], [44], [49], [54], and 2) microscopic analysis of individual instances, regions, or part of the decision boundaries to conduct local interpretation of how the classes are separated [13], [14], [48], [50], [54]. Based on our survey, we derived a list of tasks to be addressed when analyzing locally linear segments.

T1 Macroscopic Analysis: Overview of the locally linear segments:

- **T1.1: Show the number of segments and highlight the major ones:** Similar to the number of partitioned local class regions that characterize the complexity of the class structure [15], [16], [17], [49], [54], the number of detected segments is a significant indication of how complex the class separations can be. The major segments can be regarded as interesting entry points for further analysis.
- **T1.2: Reveal the coverage of data instances under each segment:** Inspired by the Melnik's questions on describing decision region structures of classifiers [44], various attributes should be summarized to identify the significance and influence for each segment, including how large the influenced region is, how many data instances are covered, and where the region is located in feature space.
- **T1.3: Depict the locations of the segments as well as relationships among different segments:** Multiple segments may share common data instances, intersect with each other, be close to neighboring segments in specific distance metrics, or be far-away from each other. Automated detection and explicit representation of the connections can facilitate analysts when exploring distances between segments and neighborhood relationships [44], [50].

T2 Microscopic analysis: Detailed analysis of specific locally linear segments:

- **T2.1: Examine the data distribution and separation near a segment:** How the data instances are distributed near the boundary segments should be provided as an overview that supports standard pattern finding tasks, such as discovering trends, clusters, class purity, and outliers [13], [14], [48]. Another critical requirement is to show how the data instances near segments are separated. Related tasks include identifying how close the adjacent classes are, and how separable the different classes are in this segment region.
- **T2.2: Show the primary features used for determining class separation:** Feature importance provides insight into which data feature(s) are driving class separations [50], [54]. Important features may vary between segment regions in a non-linear decision boundary, and visualization can facilitate the comparison of dominant features.
- **T2.3: Exploring the neighboring segments:** Acting as the counterpart of T1.3 under the "overview + detail" guideline, detailed and instance-level investigation of how neighboring segments is needed.
- **T2.4: Trace a path between segments along decision boundaries:** Tracking the connections between two seg-

ments is helpful for understanding class distributions and relationships between discriminant features. Interactive transitions on the exploration path can provide rich contextual information in an intuitive way [35], [36].

5 Visual Analysis Framework

Based on the identified tasks, we have developed a visual analysis framework, shown in Figure 3 (top), for analyzing class separations with locally linear segments. By using the extracted locally linear segments as the input, the visual analysis framework employs three main visualization components to support the exploration of class separations illustrated in Figure 3 (bottom):

1) The **Segment Relation View** and **Segment Detail View** assist users in understanding detected locally linear segments in an "overview+detail" manner. Users can interactively explore the neighboring structures of segments as well as detailed information of individual segments in the segment detail view.
2) The **Projection View** utilizes linear and t-SNE projection methods to support visual exploration at the data-instance level. Interactive manipulation of projections is supported and coordinated with the other views.
3) The **Path Exploration View** presents how the class separations in segments are connected. A touring interaction supports the dynamic exploration of the traverse path.

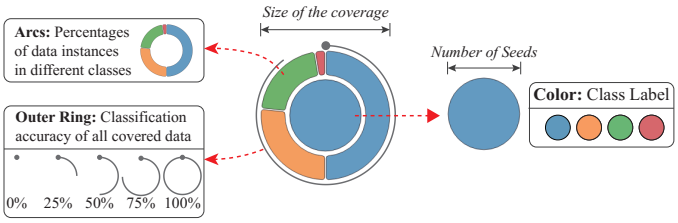
5.1 Segment Relation View

The segment relation view is designed for macroscopic analysis (**T1.1** - **T1.4**). By following the visual information seeking mantra [55], this view offers a useful summary of the extracted locally linear segments and their properties, showing the topological structures among the segments. A node-link-based design is applied with segment glyphs as nodes indicating several key characteristics of the segments. Users can easily find critical segments, explore their relationships, and compare different segments.

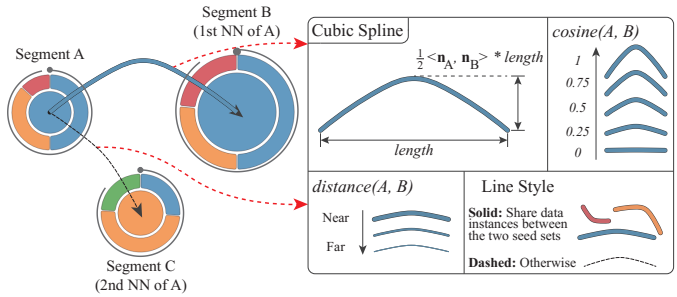
Segment Glyph: Figure 4, we use a circular design to represent locally linear segments comprised of three elements: an inner circle, a donut chart, and an outer arc.

- The inner circle provides information about the seeds. The area of the inner circle encodes the number of seeds, while the color represents the seed class.
- The donut chart shows the percentages of classes in the corresponding data. Each class is mapped to an arc in the donut chart. The diameter represents the coverage of the segment.
- The outer arc shows the classification accuracy of the separating hyperplane associated with the segment on all the covered data instances, i.e., the acc_{cover} computed in the last step of the extracting algorithm. The accuracy value is mapped to the angle of the arc which starts from 12 o-clock and grows clockwise.

Edge: Additionally, we use edges, described in Figure 4 (b), to highlight the connections between a segment and its nearest neighbors (**T2.3**).



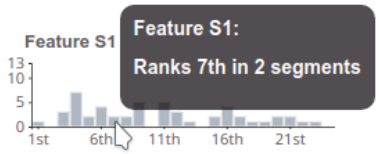
(a) Visual Encoding of Segment Glyphs



(b) Edges for Connecting Neighboring Segments

Rankings for Feature S1

Segment ID	Ranking in the feature weight
Segment A	1st
Segment B	5th
Segment C	6th
...	...
Segment n	11th



(c) Feature ranking histograms

Fig. 4. The visual encodings of segment glyphs and the edges. (a) The segment glyph consists of three parts: an inner circle, a set of arcs, and an outer ring. (b) The edges are represented by elbow-shaped cubic spline curves. (c) The feature ranking histograms shows the importance of features among all the segments.

- If two segments share common data instances in their expanded seed sets, i.e., connect to each other, the color is mapped to the class with the maximum number of shared data instances. A gray dashed curve is used when segments do not share any common instances.
- We measure the distance between segments and their nearest neighbors by using the mean value of all paired distances between their corresponding expanded seed sets. The thicknesses of the curves are proportional to the distances to the nearest neighbors where a thicker stroke implies a closer segment in terms of the distance, i.e., a stronger connection.
- Another essential attribute between segments is the angle between the normal vectors of their corresponding separating hyperplanes. To represent the angles, an elbow-shaped curvature is applied to the edges where the angularities depict the cosine value of two normal vectors. A straight line denotes the same direction of the normal vectors, while a nearly 90-degree elbow indicates orthogonality of the two separating hyperplanes.

One issue in the visual design of the edges is that this design may result in visual clutter if all the curves are displayed simultaneously on the canvas. Thus, we only initialize the edges with thin gray lines as an overview of the connections. Once the user hovers the mouse pointer over a segment glyph, its corresponding edges are shown, providing details on the neighborhood structure.

Layout: The layout of the segment glyphs visualizes the distribution of segments in the data feature space. The force-directed layout [56] is applied to the established edges between neighboring segments to compute the coordinates of segment glyphs. The reciprocals of distances between segments are used as edge weights in the layout algorithm.

Feature Ranking Histogram: As shown in Figure 4 (c), for each feature, we draw the distributions of the feature rankings among all segments by using the feature weight of the separating hyperplanes as measures. In this way, the distributions depicted in the histograms indicate how valuable a feature is in different segments (**T2.2**).

Filters and Interactions: We employ two filters for hiding insignificant segment glyphs. Users can use the checkboxes in the control panel on the left side to show or hide glyphs. To filter out segments based on the size of coverage, a histogram of the number of covered instances is adopted where users can brush the horizontal axis to select segments displayed on the canvas. Hovering on a segment glyph toggles the appearance of the connected edges. The covered data instances in the projection view will be highlighted while other instances are dimmed.

5.2 Segment Detail View

The segment detail view is used for presenting details of a selected segment (**T2.2**), Figure 3 (B). When users hover over a specific segment glyph in the segment relation view, detailed information of the segment is illustrated including covered data instances and feature weights. On the left side, a horizontal bar chart shows the five most important features and their weights. We use a parallel coordinates plot (PCP) to show the distribution of data for the five features. Brushing on the axes highlights edges in the PCP.

5.3 Projection View

Along with the two segment-related views, our framework instance-level exploration contributes to in-depth analysis of separations with respect to data instances (**T2.1**, **T2.2**).

Projection Methods: Figure 3, View (C), the projection results of all data instances are presented in scatterplots. We adopt two alternative projection methods for exploration: 1) orthographic linear projection, and; 2) t-SNE projection.

- The linear projection results are primarily used for showing linear separations conveyed by the locally linear segments. Given a segment Seg_i, we define the *side-view projection matrix* as:
$$P_i = [\overrightarrow{\boldsymbol{n}}_i, \overrightarrow{\boldsymbol{pca}}_i] \in \mathbb{R}_{q\times 2}$$
where $\overrightarrow{\boldsymbol{n}}_i$ in the first column is the normal vector of the corresponding separating hyperplane for Seg_i, and $\overrightarrow{\boldsymbol{pca}}_i$ the first principal component of all the covered data instances under Seg_i. In the projection with the side-view matrix, a vertical gap orthogonal to the horizontal axis can be observed between the instances in the class of seeds and those in other classes. Additionally, with the 1-dimensional PCA deployed on the vertical coordinates, the underlying patterns along the decision boundaries can be revealed, including clusters and outliers. Note that we further adopt the Gram-Schmidt process on P_i to obtain the orthographic projection matrix. The normal vector $\overrightarrow{\boldsymbol{n}}_i$ in the first column is not changed and hence leads the projection result, while the second column is a linear combination of $\overrightarrow{\boldsymbol{n}}_i$ and $\overrightarrow{\boldsymbol{pca}}_i$. Figure 5 shows an example of the side-view projection where the red instances are separated from the other classes.
- The t-SNE projection provides an overview of the data since it does not require any prior knowledge of the underlying patterns and structures, which is suitable for providing an initial impression of how the data is distributed. Due to its ability to extract densely-distributed clusters, the t-SNE projection view can also be used to diagnose how well the original data classes can be separated in an unsupervised manner.

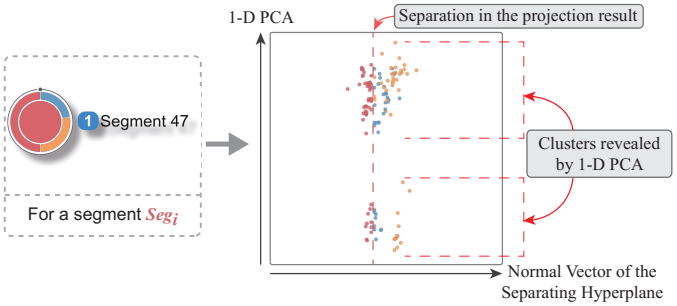


Fig. 5. The design of the linear projections. In the scatterplot, the x axis indicates the direction of the normal vector that represents the separating hyperplane, while the y coordinates are decided by the 1-dimensional PCA on all the covered data instances.

Interactions: When hovering on the points in the scatterplot, a tooltip is enabled to show the corresponding information of the data instance including its instance ID and label. To control the viewport in the linear projection, we use the optimization method by Lehmann et al. [57] to provide an interactive manual tour in the high-dimensional feature space. As shown in Figure 3 (C), once the user drags a gray endpoint, which represents the corresponding row in the projection matrix, the other rows are simultaneously adjusted via an energy optimization procedure to preserve the orthogonality of the projection matrix. The values of the projection matrix are listed in a table.

5.4 Path Exploration View

The path exploration view is used to discover an optimized exploration path between multiple segments and allows for flexible traverse between any segments (**T2.3**, **T2.4**).

Selecting Segments: Users can select a series of segment glyphs in the segment relation view as an entry point for creating a traverse path. By clicking on the glyphs, the corresponding segments are added into a list depicted in the top left corner of the view. The orders of the segments in the sequence are highlighted by the right side of the glyphs, and the existing edges are displayed to indicate shared seeds between segments. The list of selected segments can be submitted to generate a traverse path by clicking on the "Create a Path" button.

Building an Optimized Traverse Path: Figure 6 (A) illustrates the process of building an optimized traverse path between two consecutive segments. For two segments that

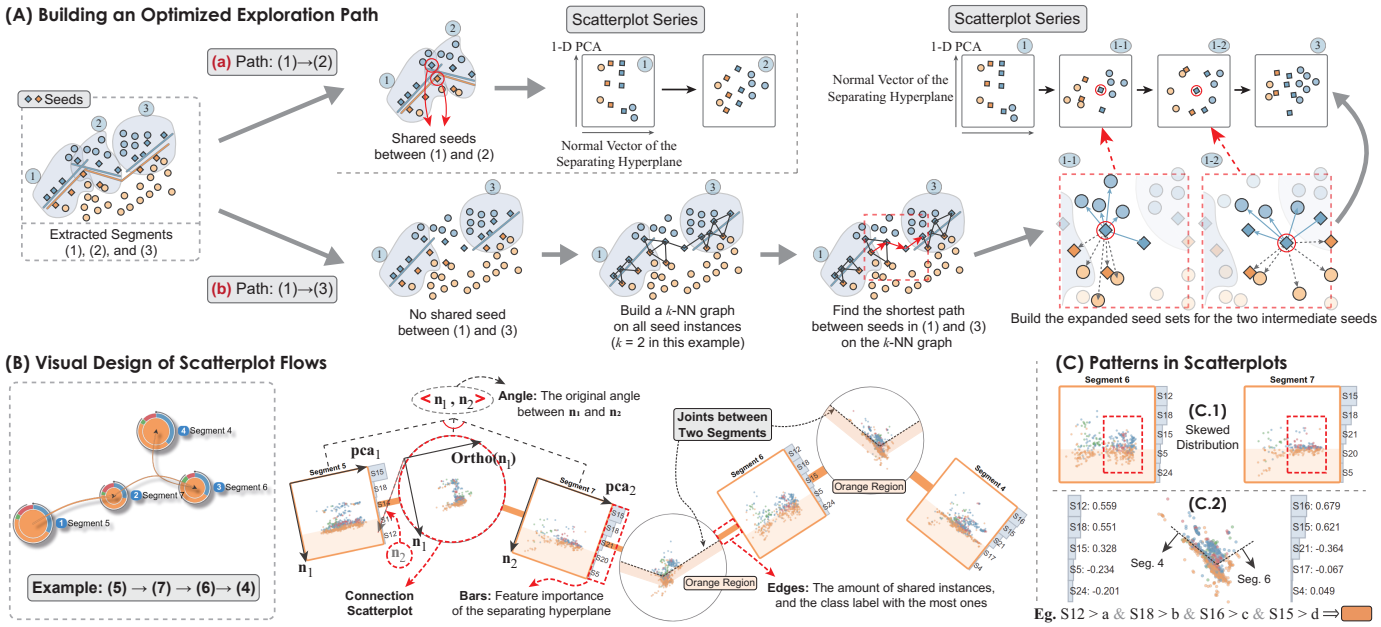


Fig. 6. Planning an exploration path between two locally linear segments. (A) For two segments sharing some seeds such as Segment 1 and 2, the series of scatterplots only consists of the two segments. If the two segments have no shared seeds (Path 1 to 3), a list of intermediate seeds are selected to provide a smooth transition along the decision boundaries. (B) The scatterplots are concatenated in a zig-zag manner. (C) Skewed distributions are found in Segment 6 and 7. An example of the joint of two connected separations (Segment 4 and 6) is illustrated, which can be described as a classification rule.

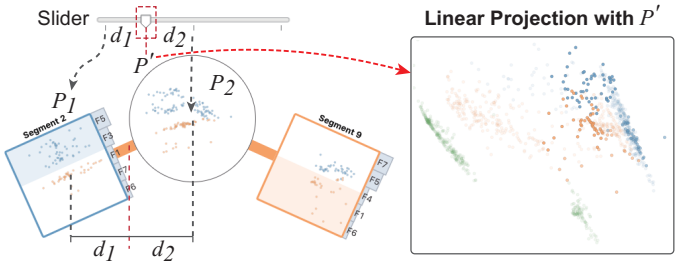


Fig. 7. The touring interaction in the path exploration view. The slider indicates the current location between two projection matrices.

share common seeds, e.g., Path (1)→(2) in Figure 6 (A), we consider the segments to be connected and need no additional bridging transitions. However, if the two segments have no common seeds that connect them, it will be difficult to perceive how the two local separations are joined together. To resolve this issue, we apply an interpolation-based method to provide a smooth transition between the two segments:

1) In the example of Path (1)→(3) in Figure 6 (A), we first build an k-NN graph among all the seed instances in all locally linear segments. The k is set as the minimum value that the seeds are connected.
2) Then, we find the shortest path between the expanded seed sets of Segment (1) and (3) in the k-NN graph.
3) Next, temporary locally linear segments on the intermediate instances (1-1 and 1-2) in the shortest path are generated with the method described in Figure 2 (b).
4) Finally, these new segments built on the intermediate instances are inserted between the original segments.

If there are more than two segments selected by users, the interpolation process will be repeated on every two consecutive segments. We build a traverse path which can depict class separations at every position in the path and provide a continuous transition between segments.

Visualizing the Segment Series: After the traverse path is built, the series of segments are visualized in the path exploration view illustrated in Figure 3 (D). The sequence of originally selected segments is listed on the left side. When clicking on the title of the sequence, the details are represented as a list of scatterplots on the right side (Figure 6 (B)). For the segments in the traverse path, including the newly-inserted intermediate ones, we juxtapose the scatterplots of projections with corresponding side-view projection matrices. A shaded area is drawn under the dots with the color assigned to the corresponding seed class in order to show the class boundary derived from the separating hyperplane. However, one problem of simple juxtaposition is that the connections of separations between consecutive segments are not emphasized. To address this issue, we adopt a concatenation method to highlight the relationship between segments. Between each pair of two consecutive scatterplots, a new connection scatterplot is inserted, which contains all the data instances of the two neighboring scatterplots. The coordinates of the instances are assigned to a projection matrix consisting of two normal vectors from the two scatterplots to simultaneously show the separations in the two corresponding segments. It should be noted that the normal vectors may not be orthogonal to each other and cannot be directly used as a basis of an orthogonal linear projection. We use the Gram-Schmidt process to create an orthogonal basis on the same 2-D plane for the connection scatterplot. Similar to the curved edges in the segment relation view, a triangular layout is adopted to present the actual angle between normal vectors and link the entire series in a zig-zag way.

Interactive Traverse Between Scatterplots: The scatterplots

in the path exploration view provide a static representation of how the separations are distributed and connected. Inspired by the existing traveling methods in high-dimensional space [35], [36], [58], we use a touring interaction with dynamic transitions between locally linear segments. As shown in Figure 7, at the top of the path exploration view, a horizontal slider is employed ranging from the first scatterplot to the last. When the handle is placed between two scatterplots, we compute the current projection matrix P' based on the two side-view projection matrices P_1 and P_2 of the two scatterplots:

$$P' = GramSchmidt(P_1 + \frac{d_1}{d_1 + d_2}(P_2 - P_1))$$

where d_1 and d_2 are the horizontal distances from the handle to the center of the left and right scatterplots, respectively. P' is then applied to the linear projection view where the instances from the two scatterplots are highlighted. The opacity values of instances from the left and the right scatterplots are set as $\frac{d_1}{d_1+d_2}$ and $\frac{d_2}{d_1+d_2}$.

6 Case Study and Expert Interview

In this section, we describe how our approach facilitates the exploration and diagnosis of class separations through two case studies using real-world high-dimensional datasets. We also summarize feedback from two expert users.

6.1 Shuttle Statlog Dataset

In this case study, we explore the space shuttle statlog dataset[1] which contains 8 numerical features and 7 classes. The classes 2 (Fpv Close), 3 (Fpv Open), 6 (Bpv Close), and 7 (Bpv Open) are known outliers with only a few data instances. For purposes of this case study, we have removed these classes from the original dataset. Since the original dataset is too large (58000 instances), we then randomly sampled 400 instances from each of the remaining 3 classes (Rad Flow, High, Bypass), resulting in a dataset with 1200 instances in total. Downsampling was done to improve computation time and reduce clutter in the scatterplots. These limitations are discussed in the conclusion.

Overview of the Separations: After the dataset is loaded into our system, the locally linear segments are automatically extracted from the dataset. We first check the t-SNE and linear projections to get an overall impression of how the classes are distributed (**T2.1**). In the t-SNE projection result, Figure 3 (1), we see that the two green clusters are clearly split from the blue and the orange class, while the other two classes have overlapping regions. The separation pattern is further illustrated in the initial PCA projection result in Figure 3 (2), where a clear linear boundary can be drawn between the green clusters and other instances. Moreover, the boundary between the blue and the orange classes is not strictly linear and consists of two significant linear separations. This indicates that the relationships between the blue/orange classes should be further examined.

Segment Relationships: Next, we inspect the segment relation view to explore the extracted segments and their relationships. In Figure 3 (3), the graph structure shows two communities of segments (**T1.1, T1.2, T1.3**). The green instances are mainly in community (a), while the orange and blue instances lie in community (b) (**T2.3**). The community structure confirms the relationships identified in the t-SNE projection. We discover that the green segments mostly consist of green and orange instances (**T1.3, T2.1**). This indicates that the green class is near the orange class but not close to the blue one, which is different from the t-SNE projection result where part of the green instances is near the blue class. Additionally, there is a green segment far away from the central community of green segments, Figure 3 (4). By hovering on the glyph, we find that the corresponding instances are a group of outliers from the larger green cluster (**T2.1**), which may partially be the reason of non-connectivity in the segment graph.

Investigation of Specific Classes: Based on the segments and their relationships, we further investigate how each class or class pair is separated. Specifically, we address the question that, for a class, are there any decisive features that influence separations in different local regions **(T2.2)**?

- First, for the green classes in community (a), we examine the most important features for each segment by hovering the mouse cursor on each glyph and check the details in the segment detail view, Figure 3 (5). From the feature lists, it can be found that F5, F6, and F7 are the top features, suggesting that these features should be considered when separating green instances with others.
- Next, to explore the relationships between the blue and the orange class, we discover two subgroups in the community (b) that have different essential features: (b.1) where F5 dominates the separation, and (b.2) has the highest weight on F7 (Figure 3 (6)). The pattern of dominant features can be used to summarize classification rules in two different regions. For example, for unlabeled instances in the regions covered by segments in (b.1), we may predict their class labels with the values on F5.
- Furthermore, we explore the transitions between the two subgroups. In Figure 3 (7), the curved link between Segment 2 and 9 indicates an angle between the corresponding separations in two segments. After creating an exploration path from Segment 2 to 9, the scatterplot in the middle shows the joint of two separations where a clear angled transition is depicted from left to right. In practice, when an unlabeled instance falls into the joint region, the dominant features for both separations should be considered to judge the label.

6.2 Wall-following Robot Navigation Dataset

Our second case study explores the wall-following robot navigation dataset [59] that comprises measurements from 24 ultrasound sensors deployed on a robot, Figure 9. The labels of the sensor readings correspond to four commands to the robot, including "Move-Forward", "Slight-Right-Turn", "Sharp-Right-Turn", and "Slight-Left-Turn". As described by Freire et al. [59], the dataset was designed to test whether the navigation task is non-linearly separable. Thus, it is impossible to learn the classification task with a single linear classifier. To further assess our framework, a domain expert in machine learning and robotics was invited to explore the

1. https://archive.ics.uci.edu/ml/datasets/Statlog+(Shuttle)

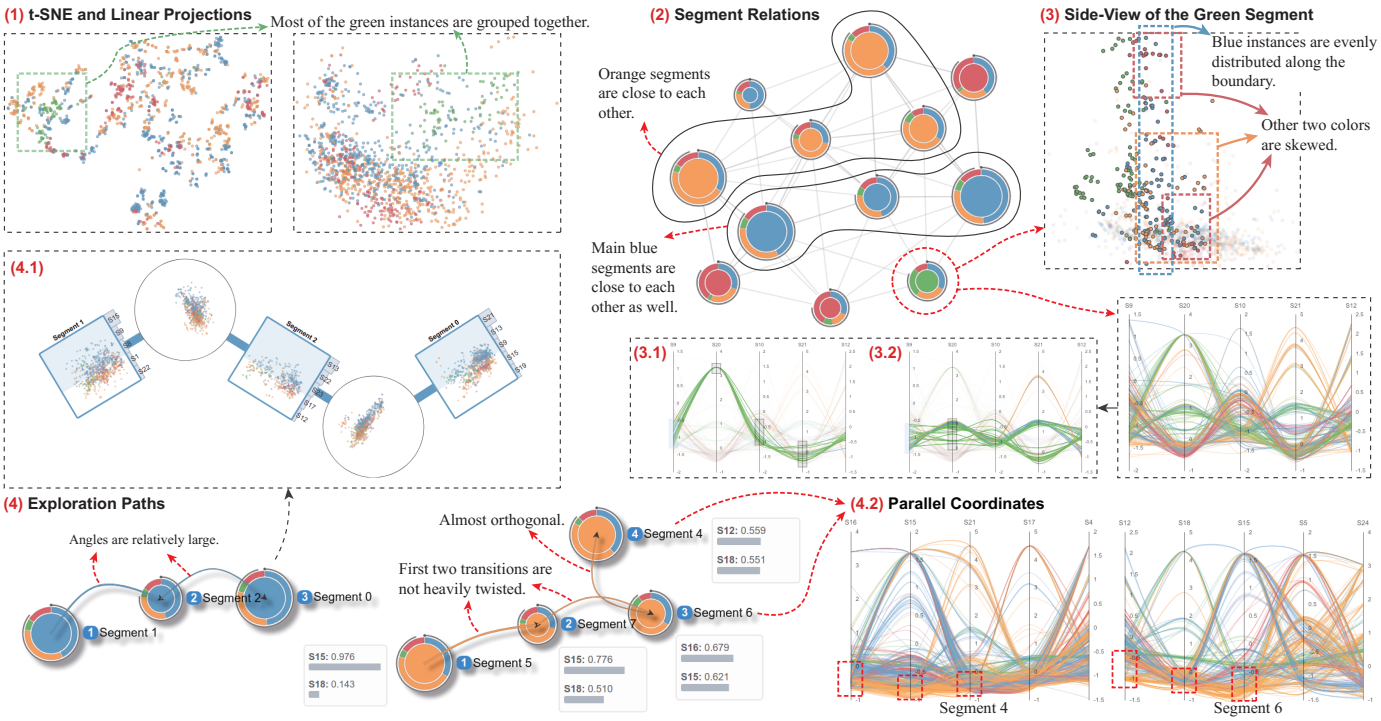


Fig. 8. The result of the wall-following robot sensor data. (1) The two projection results show heavy overlaps between the red, the blue, and the orange classes, while the green instances are roughly grouped together. (2) In the segment relation view, the segments in the blue class and the orange class are relatively close to each other. Only one green segment is constructed, which matches the grouping patterns found in the linear and the t-SNE projection results. (3) The side-view of the green segment is presented, showing the relationships between the green instances and the other classes. The data distribution of the green segment is shown in (3.1) and (3.2). (4) Two paths are selected from the blue and the orange classes, respectively. The corresponding details of the path and two parallel coordinates are presented in (4.1) and (4.2).

dataset with our framework. In the data exploration process, we first introduced the background and the design goals of our framework to the expert, followed by an illustration of the interactive interface and the analytical workflow with the dataset in the first case study. Next, we presented some patterns we have discovered while building the framework to get the expert's verification and explanations. The expert was allowed to explore the datasets freely during the interview process.

Initial Linear and t-SNE Projection Results: Shown in Figure 8 (1), we find that the two projection results show significant overlaps in all four classes of instances (**T2.1**). Notably, there is no clear cluster structure for a specific class shown in the t-SNE projection. The expert confirmed that the heavily overlapped points in the projection results indicate a complex class separations between the four classes. Thus, we need to use the extracted locally linear segments to explore separations from the perspective of local regions.

Segment Relationships: In the segment relation view, Figure 8 (2), there is no salient community structure in the segment graph, which matches the overlapped distributions among all classes depicted in the t-SNE projection (**T1.1**). For the blue and the orange class, their segments in the same class are relatively close to each other and linked with solid edges, i.e., the four orange segments and the three blue segments in Figure 8 (2) (**T1.2, T1.3**). Meanwhile, there is only one segment associated with the green class. The expert commented that the segment layout for the blue and the orange classes might indicate that separations are linked together for each class. In addition, the grouping patterns for the green instances in the t-SNE and the linear projection results may be the reason of why there is only one green segment.

Investigation of Specific Classes: Based on the segment relationships found above, the expert was interested in exploring the detailed patterns of the separations in the three classes, respectively. After the green segment is activated by double-clicking on the glyph, Figure 8 (3), the linear projection result shows a fuzzy separation between the green instances on the left and others on the right **(T2.1)**. It can also be observed that the blue instances are evenly distributed along the entire boundary, while orange ones are mainly located at the bottom side. This indicates that the green class is connected with the blue class in a wide range, and the orange and the red classes only attach the green instances in a restricted region of the segment. The expert further recommended exploring the data distribution of the green segment in the parallel coordinates to explore whether the green instances in the green segment are grouped together. When hovering the mouse pointer on the green segment to activate its details in the segment detail view, two batches of green edges, Figure 8 (3.1) and (3.2), can be highlighted by brushing on the axes, indicating two major clusters in the green class. In addition, the corresponding sensors of the five most important features suggest the front left (S20, S21) and the rear right sensors (S9, S10, S12) are mainly considered when making a "Slight-Left-Turn". The expert suggested that this clue may enlighten the robot designers to further investigate why these two directions are essential for moving left.

Next, we explore the two segment groups in blue and or-

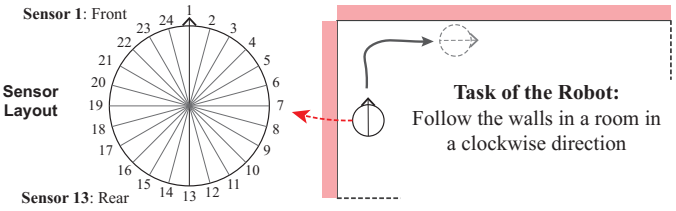


Fig. 9. An illustration of the wall-following robot. (Left) The sensors corresponding to the features S1-S24 are mounted around the robot. (Right) The task of the robot is to follow the wall in a room in a clockwise direction.

ange colors, respectively, Figure 8 (2) (4). When highlighting the edges between the four orange segments, we find that the connections between Segment 5, 7, and 6 are not heavily twisted, while Segment 4 is rotated to another direction. However, it is not the same for the blue group where the top features vary among all the segments (Figure 8 (4.1)). The expert mentioned that the shared top features (S15, S18, S12) among Segment 5, 7, and 6 should be specifically considered for the orange class **(T2.2)**, while there are no common important features in the blue class. Different feature groups should be considered for the local regions covered by the blue segments.

The details of the data distributions near separations and transitions in the two groups can be observed in the corresponding exploration paths **(T2.1, T2.3, T2.4)**. In the path of the orange segments, Figure 6 (C.1), blue and red instances in Segment 7 and 6 are skewed to one side and the remaining green instances on the other side, showing that in these two local regions the major adjacent classes are blue and red which only connect to a part of the instances in the two segments. The skewness does not exist in Segment 5 and 4, where the blue and red instances are evenly distributed along the boundary. However, there is no significant pattern in distributions of non-blue instances but noise-like clutters in the path of the blue segments, Figure 8 (4). Concerning the transitions, the joints in the first two connection scatterplots in the orange path, Figure 6 (B), are relatively flat with the dominant features of S15, S18, and S12. The separations become nearly orthogonal in the third one. The expert discovered that S12, S18, S15, S16, and S21 receive high importance values in identifying orange instances on the bottom side in the local joint region, Figure 6 (C.2). Furthermore, the expert examined the parallel coordinates of Segment 4 and 6 and observed that most of the yellow edges receive relatively low values on the top features. After checking the sensor layout, the expert commented that the local joint region may indicate a situation where the robot is very close to the wall on the left side, i.e., the readings are low from the sensors on the left side. Since the robot is set to follow the wall in a room in a clockwise direction, a "Sharp-Right-Turn" is required to keep the robot away from the wall on the left side.

7 Discussion and Conclusion

In this paper, we present a visual analytics approach for exploring complex non-linear class separations in labeled high-dimensional datasets. In order to achieve a balance between interpretability of the projection results and the ability to describe complex decision boundaries, we use locally linear segments that are automatically extracted from the dataset to represent the class separations. In the visualization stage, the relationships and details of locally linear segments are presented in order to support exploration and investigation of how instances from a specific class are separated from other classes. Users are allowed to check projection results, view the detailed information of segments, and build exploration paths to traverse along the class separation boundaries to observe the transitions between different local regions. We demonstrate the usability of our visual analytics framework through two case studies and domain expert reviews.

Parameter Tuning: When employing the locally linear extraction algorithm, users need to set a few parameters, including the size k of the expanded seed sets, the threshold θ to accept a separating hyperplane found by a linear SVM, and the size t to control the upper limit of possible merges. In general, k and t are responsible for the granularity of segments, and θ presents the tolerance to abandon some less important instances when searching for hyperplanes. We used default settings of $k = 10$, $\theta = 0.9$, and $t = 20$ in the two datasets presented in Section 6. A future extension will be to find optimal parameter settings with hyper-parameter tuning methods widely used in machine learning, such as minimizing the accuracies of covered instances for all segments by grid searching and cross-validation. Fast visual examination methods to check the algorithm output can also speed up the parameter tuning procedure. In the segment relation view, the number of nearest neighbors for segment glyphs may influence the pattern extraction based on the edge structures. We set the number to 5 in the two cases presented in Section 6. In the future, we plan to provide a control widget for adjusting the number of visible edges.

Scalability: One limitation in our visual analytics approach is the scalability with respect to the extraction of the locally linear segments and visual presentation of the elements.

Extraction of the Locally Linear Segments: The performance of the extraction algorithm has two limitations. First, since we utilize the support vectors of non-linear SVMs as the initial seed instances, performance speeds can be limited if there are too many support vectors selected by the SVM model. In the two case studies, it took about 4 minutes and 12 minutes respectively to finish the extraction procedure. A potential solution is to deploy filtering procedures before running the segment extraction algorithm, such as sampling methods to reduce the number of seed instances, or feature selection methods to remove redundant features. Second, the run time to train numerous linear SVM models can be a bottleneck once a large number of seeds are considered. For this issue, we can parallelize the model training process by distributing multiple SVM fitting tasks to different cores or machines.

Visual Design: In our design, a common issue for the scatterplots is the visual clutter caused by large numbers of dots plotted in the view. A similar issue also exists in the segment graph view where the glyphs may be overlapped once too many segments are extracted. To mitigate the visual clutter in the segment graph, we have employed filtering methods in the segment graph view to hide irrelevant glyphs and

edges. In the future, a feasible way to further scale our design in the scatterplots in the projection views is to adopt sampling and abstraction methods to reduce the number of visible dots [60], [61], [62].

Future Work: In the future, we expect to accelerate the implementation of the segment extraction process in order to support real-time exploration of large-scale datasets. By utilizing the topology graphs extracted with existing works [40], [42], [44], we also plan to explore different seed generation methods in the extraction algorithms. Another promising extension is to integrate our approach in the process of diagnosing classification and clustering models. By checking the quality of decision boundaries, users can gain insight into how the models separate instances into different classes (or clusters), how the data instances distribute near decision boundaries, and how separable it is between two or more classes. The insights can further assist the adjustment of the training processes including the selection of proper model types and informative features in order to get better prediction performance.

ACKNOWLEDGMENTS

This work was supported by the U.S.Department of Homeland Security under Grant Award 2017-ST-061-QA0001 and 17STQAC00001-03-03, and the National Science Foundation Program on Fairness in AI in collaboration with Amazon under award No. 1939725. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the U.S. Department of Homeland Security.

REFERENCES

[1] J. Krause, A. Perer, and E. Bertini, "INFUSE: Interactive feature selection for predictive modeling of high dimensional data," *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 12, pp. 1614–1623, 2014.

[2] J. P. Cunningham and Z. Ghahramani, "Linear dimensionality reduction: Survey, insights, and generalizations," *The Journal of Machine Learning Research*, vol. 16, no. 1, pp. 2859–2900, 2015.

[3] S. Gratzl, A. Lex, N. Gehlenborg, H. Pfister, and M. Streit, "LineUp: Visual analysis of multi-attribute rankings," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 12, pp. 2277–2286, 2013.

[4] S. Liu, P. T. Bremer, J. J. Jayaraman, B. Wang, B. Summa, and V. Pascucci, "The Grassmannian atlas: A general framework for exploring linear projections of high-dimensional data," *Computer Graphics Forum*, vol. 35, no. 3, pp. 1–10, 2016.

[5] Y. Koren and L. Carmel, "Visualization of labeled data using linear transformations," *Proceedings of IEEE Information Visualization*, 2003.

[6] Y. Wang, K. Feng, X. Chu, J. Zhang, C.-W. Fu, M. Sedlmair, X. Yu, and B. Chen, "A perception-driven approach to supervised dimensionality reduction for visualization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 5, pp. 1828–1840, 2017.

[7] L. G. Nonato and M. Aupetit, "Multidimensional projection for visual analytics: Linking techniques with distortions, tasks, and layout enrichment," *IEEE Transactions on Visualization and Computer Graphics*, vol. 25, no. 8, pp. 2650–2673, 2019.

[8] J. Xia, F. Ye, W. Chen, Y. Wang, W. Chen, Y. Ma, and A. K. Tung, "LDSScanner: Exploratory analysis of low-dimensional structures in high-dimensional datasets," *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 1, pp. 236–245, 2017.

[9] S. Liu, D. Maljovec, B. Wang, P. T. Bremer, and V. Pascucci, "Visualizing high-dimensional data: Advances in the past decade," *IEEE Transactions on Visualization and Computer Graphics*, vol. 23, no. 3, pp. 1249–1268, 2017.

[10] X. Geng, D. Zhan, and Z. Zhou, "Supervised nonlinear dimensionality reduction for visualization and classification," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 35, no. 6, pp. 1098–1107, 2005.

[11] L. V. D. Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008.

[12] T. K. Kim and J. Kittler, "Locally linear discriminant analysis for multimodally distributed classes for face recognition with a single model image," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 3, pp. 318–327, 2005.

[13] H. Zhang, A. C. Berg, M. Maire, and J. Malik, "SVM-KNN: Discriminative nearest neighbor classification for visual category recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, June 2006, pp. 2126–2136.

[14] L. Ladicky and P. Torr, "Locally linear support vector machines," in *Proceedings of the 28th International Conference on Machine Learning*, 2011, pp. 985–992.

[15] J. Wang and V. Saligrama, "Local supervised learning through space partitioning," *Advances in Neural Information Processing Systems 25*, vol. 004, pp. 91–99, 2012.

[16] M. Fornoni, B. Caputo, and F. Orabona, "Multiclass latent locally linear support vector machines," in *Proceedings of the Asian Conference on Machine Learning*, 2013, pp. 229–244.

[17] J. Wang and V. Saligrama, "Locally-linear learning machines (l3m)," in *Proceedings of the Asian Conference on Machine Learning*, 2013, pp. 451–466.

[18] P. Joia, F. V. Paulovich, D. Coimbra, J. A. Cuminato, and L. G. Nonato, "Local affine multidimensional projection," *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, no. 12, pp. 2563–2571, 2011.

[19] J. B. Tenenbaum, V. de Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–23, dec 2000.

[20] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323 LP – 2326, dec 2000.

[21] F. Paulovich, L. Nonato, R. Minghim, and H. Levkowitz, "Least square projection: A fast high-precision multidimensional projection technique and its application to document mapping," *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, no. 3, pp. 564–575, 2008.

[22] J. Tang, J. Liu, M. Zhang, and Q. Mei, "Visualizing large-scale and high-dimensional data," in *Proceedings of the 25th International Conference on World Wide Web*, 2016, pp. 287–297.

[23] M. Sips, B. Neubert, J. P. Lewis, and P. Hanrahan, "Selecting good views of high-dimensional data using class consistency," *Computer Graphics Forum*, vol. 28, no. 3, pp. 831–838, 2009.

[24] M. Sedlmair and M. Aupetit, "Data-driven evaluation of visual quality measures," *Computer Graphics Forum*, vol. 34, no. 3, pp. 201–210, 2015.

[25] M. Aupetit and M. Sedlmair, "SepMe: 2002 new visual separation measures," in *Proceedings of the IEEE Pacific Visualization Symposium*, 2016.

[26] D. Sacha, L. Zhang, M. Sedlmair, J. A. Lee, J. Peltonen, D. Weiskopf, S. C. North, and D. A. Keim, "Visual interaction with dimensionality reduction: A structured literature analysis," *IEEE Transactions on Visualization and Computer Graphics*, vol. 2626, no. Section 6, pp. 1–1, 2016.

[27] J. H. Lee, K. T. McDonnell, A. Zelenyuk, D. Imre, and K. Mueller, "A structure-based distance metric for high-dimensional space exploration with multidimensional scaling," *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 3, pp. 351–364, 2014.

[28] S. Liu, B. Wang, J. J. Thiagarajan, P. Bremer, and V. Pascucci, "Visual exploration of high-dimensional data through subspace analysis and dynamic projections," *Computer Graphics Forum*, vol. 34, no. 3, 2015.

[29] J. Xia, W. Chen, H. Yumeng, H. Wanqi, H. Xinxin, and D. S. Ebert, "DimScanner: A relation-based visual exploration approach towards data dimension inspection," *Proceedings of the IEEE Visual Analytics Science and Technology*, 2016.

[30] F. Zhou, J. Li, W. Huang, Y. Zhao, X. Yuan, X. Liang, and Y. Shi, "Dimension reconstruction for visual exploration of subspace

clusters in high-dimensional data," *IEEE Pacific Visualization Symposium*, vol. 2016-May, no. 1, pp. 128–135, 2016.

[31] D. H. Jeong, C. Ziemkiewicz, B. Fisher, W. Ribarsky, and R. Chang, "iPCA: An interactive system for pca-based visual analytics," *Computer Graphics Forum*, vol. 28, no. 3, pp. 767–774, 2009.

[32] D. Asimov, "The grand tour: A tool for viewing multidimensional data," *SIAM Journal on Scientific and Statistical Computing*, vol. 6, no. 1, pp. 128–143, 1985.

[33] J. H. Friedman and J. W. Tukey, "A projection pursuit algorithm for exploratory data analysis," *IEEE Transactions on Computers*, vol. C-23, no. 9, 1974.

[34] J. Wenskovitch, I. Crandell, N. Ramakrishnan, L. House, S. Leman, and C. North, "Towards a systematic combination of dimension reduction and clustering in visual analytics," *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 1, pp. 131–141, 2017.

[35] J. E. Nam and K. Mueller, "Tripadvisor N-D: A tourism-inspired high-dimensional space exploration framework with overview and detail," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 2, pp. 291–305, 2013.

[36] B. Wang and K. Mueller, "The subspace voyager: Exploring high-dimensional data along a continuum of salient 3d subspaces," *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 2, pp. 1204–1222, 2018.

[37] F. C. Rodrigues, R. Hirata, and A. C. Telea, "Image-based visualization of classifier decision boundaries," in *Proceedings of the 31st Conference on Graphics, Patterns and Images*, 2019, pp. 353–360.

[38] F. C. Rodrigues, M. Espadoto, R. Hirata, and A. C. Telea, "Constructing and visualizing high-quality classifier decision boundary maps," *Information (Switzerland)*, vol. 10, no. 9, pp. 1–22, 2019.

[39] L. Wasserman, "Topological data analysis," *Annual Review of Statistics and Its Application*, vol. 5, pp. 501–532, 2018.

[40] M. Aupetit and T. Catz, "High-dimensional labeled data analysis with topology representing graphs," *Neurocomputing*, vol. 63, no. SPEC. ISS., pp. 139–169, 2005.

[41] M. Aupetit, "Learning topology with the generative gaussian graph and the em algorithm," *Advances in Neural Information Processing Systems 18*, pp. 83–90, 2006.

[42] P. Gaillard, M. Aupetit, and G. Govaert, "Learning topology of a labeled data set with the supervised generative Gaussian graph," *Neurocomputing*, vol. 71, no. 7, pp. 1283 – 1299, 2008.

[43] Q. Mao, L. Wang, I. W. Tsang, and Y. Sun, "Principal graph and structure learning based on reversed graph embedding," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 11, pp. 2227–2241, 2017.

[44] O. Melnik, "Decision region connectivity analysis: A method for analyzing high-dimensional classifers," *Machine Learning*, vol. 48, no. 1997, pp. 321–351, 2002.

[45] T. Martinetz and K. Schulten, "Topology representing networks," *Neural Networks*, vol. 7, no. 3, pp. 507–522, 1994.

[46] K. N. Ramamurthy, K. Varshney, and K. Mody, "Topological data analysis of decision boundaries with application to model selection," in *Proceedings of the 36th International Conference on Machine Learning*, 2019, pp. 5351–5360.

[47] W. He, B. Li, and D. Song, "Decision boundary analysis of adversarial examples," in *Proceedings of the 6th International Conference on Learning Representations*, 2018.

[48] A. P. Engelbrecht, "Sensitivity analysis for decision boundaries," *Neural Processing Letters*, vol. 10, no. 3, pp. 253–266, 1999.

[49] H. Oiwa and R. Fujimaki, "Partition-wise linear models," in *Proceedings of the International Conference on Neural Information Processing Systems*, 2014.

[50] C. Jose, P. Goyal, P. Aggrwal, and M. Varma, "Local deep kernel learning for efficient non-linear SVM prediction," in *Proceedings of the 30th International Conference on Machine Learning, ICML 2013*, vol. 28, no. PART 2, 2013, pp. 1523–1531.

[51] M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why should i trust you?' Explaining the predictions of any classifier," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.

[52] C. J. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[53] L. Bottou and V. Vapnik, "Local learning algorithms," *Neural Computation*, vol. 4, no. 6, pp. 888–900, 1992.

[54] Z. Xu, M. J. Kusner, K. Q. Weinberger, and M. Chen, "Cost-sensitive tree of classifiers," in *Proceedings of International Conference on Machine Learning*, vol. 28, no. PART 1, 2013, pp. 133–141.

[55] B. Shneiderman, "The eyes have it: A task by data type taxonomy for information visualizations," in *Proceedings of IEEE Symposium on Visual Languages*, Sep. 1996, pp. 336–343.

[56] M. Jacomy, T. Venturini, S. Heymann, and M. Bastian, "ForceAtlas2, a continuous graph layout algorithm for handy network visualization designed for the gephi software," *PLOS ONE*, vol. 9, no. 6, pp. 1–12, 06 2014.

[57] D. J. Lehmann and H. Theisel, "Orthographic star coordinates," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 12, pp. 2615–2624, Dec 2013.

[58] N. Elmqvist, P. Dragicevic, and J. Fekete, "Rolling the dice: Multidimensional visual exploration using scatterplot matrix navigation," *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, no. 6, pp. 1539–1148, 2008.

[59] A. L. Freire, G. A. Barreto, M. Veloso, and A. T. Varela, "Short-term memory mechanisms in neural network learning of robot navigation tasks: A case study," in *Proceedings of the 6th Latin American Robotics Symposium*, 2009, pp. 1–6.

[60] A. Mayorga and M. Gleicher, "Splatterplots: Overcoming overdraw in scatter plots," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 9, pp. 1526–1538, Sep. 2013.

[61] H. Chen, W. Chen, H. Mei, Z. Liu, K. Zhou, W. Chen, W. Gu, and K. Ma, "Visual abstraction and exploration of multi-class scatterplots," *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 12, pp. 1683–1692, Dec 2014.

[62] H. Liao, Y. Wu, L. Chen, and W. Chen, "Cluster-based visual abstraction for multivariate scatterplots," *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 9, pp. 2531–2545, Sep. 2018.

Yuxin Ma is a Postdoctoral Research Associate in the School of Computing, Informatics and Decision Systems Engineering at Arizona State University. He received his B.S. and Ph.D. from Zhejiang University, China in 2012 and 2017, respectively. He worked as an intern researcher in SeSaMe (Sensor-enhanced Social Media) Center, National University of Singapore, in 2016. His current research interests are Explainable AI, Information Visualization, and Visual Analytics.

Ross Maciejewski is an Associate Professor at Arizona State University in the School of Computing, Informatics and Decision Systems Engineering and Director of the Center for Accelerating Operational Efficiency - a Department of Homeland Security Center of Excellence. His primary research interests are in the areas of geographical visualization and visual analytics focusing on homeland security, public health, dietary analysis, social media, criminal incident reports, and the food-energy-water nexus. His work has been recognized through a variety of awards at the IEEE Visual Analytics Contest (2010, 2013, 2015), a best paper award in EuroVis 2017, and a CHI Honorable Mention Award in 2018.