Towards the Usability of Dimensionality Reduction for Visual Analytics

A dissertation

submitted by

Gabriel Raia Appleby

In partial fulfillment of the requirements
for the degree of

Doctor of Philosophy

in

Computer Science

TUFTS UNIVERSITY

May 2024

ADVISOR: Remco Chang

Towards the Usability of Dimensionality Reduction for Visual Analytics

Gabriel Raia Appleby

ADVISOR: Remco Chang

The generation of complex data is increasing rapidly. Data visualization helps users make sense of this data and extract insight. Many methods exist for visualizing low-dimensional data, but high-dimensional data often requires dimensionality reduction. Dimensionality reduction methods allow users to visualize complex data effectively, frequently utilizing as few as two dimensions. However, these techniques are complicated and require careful tuning and understanding. In this dissertation, I introduce novel approaches to constructing arbitrary projections and tools for better understanding and building trust in said projections. I begin with an interview study showcasing the struggles facing practitioners who need to understand and communicate complex data. I then explain three novel contributions towards understanding projections: HyperNP, NNInv, and DimBridge. These methods leverage recent advancements in machine learning to help users tune and correctly understand their projections.

For Dayna.

Acknowledgments

I am eternally grateful to my friends, family, and lab. Each of you has significantly shaped my journey, and I am so fortunate to have met such wonderful people.

To my wife, Dayna, there are no words to express how much you have been my rock throughout this process. Your unwavering support and understanding has been the bedrock of my journey. Thank you for everything.

To my parents, Peter and Stephanie, and to my siblings, David and Kyra, you have supported me throughout my journey. Your constant encouragement has been instrumental in my success. I am truly gifted to have such amazing family members in my life.

No one could ask for better labmates. Ab, Ashley, Brian, Camelia, Dylan, Jen, and Rui, I cannot thank you enough for all the help and direction you have given me over the years. Your contributions have been invaluable, from guidance to support during challenging times.

Remco, I am deeply grateful for your advice and unwavering belief in me. Your encouragement and feedback, which were a guiding force throughout my PhD, were crucial to my success. I would not have reached the finish without your help.

I want to thank my committee, Erik Anderson, Rob Jacob, Fahad Dogar, and James Intriligator, for their guidance throughout this process. And to the rest of the CS department, your collective support and assistance have been instrumental in my success.

Gabriel Raia Appleby

TUFTS UNIVERSITY

May 2024

Contents

List of Tables

List of Figures

Summary of Papers

Portions of this dissertation are based on the following papers:

- Ashley Suh, Gabriel Appleby, Erik W. Anderson, Luca Finelli, Remco Chang, and Dylan Cashman. Are metrics enough? guidelines for communicating and visualizing predictive models to subject matter experts. *IEEE Transactions on Visualization and Computer Graphics*, pages 1–16, 2023

- G. Appleby, M. Espadoto, R. Chen, S. Goree, A. C. Telea, E. W. Anderson, and R. Chang. Hypernp: Interactive visual exploration of multidimensional projection hyperparameters. *Computer Graphics Forum*, 41(3):169–181, 2022

- Mateus Espadoto, Gabriel Appleby, Ashley Suh, Dylan Cashman, Mingwei Li, Carlos Scheidegger, Erik W. Anderson, Remco Chang, and Alexandru C. Telea. Unprojection: Leveraging inverse-projections for visual analytics of high-dimensional data. *IEEE Transactions on Visualization and Computer Graphics*, 29(2):1559–1572, 2023

- Brian Montambault, Gabriel Appleby, Jen Rogers, Camelia D. Brumar, Mingwei Li, and Remco Chang. Dimbridge: Interactive explanation of visual patterns in dimensionality reductions with predicate logic, 2024

Chapter 1

Introduction

Data is everywhere. By 2025, the amount of data "created, captured, copied, and consumed worldwide" is forecasted to reach over 180 zettabytes [Tay]. This data is produced by scientific fields, from information technology to bioinformatics [Buh11]. Data Visualization is a sub-field of Computer Science dedicated to helping users visualize this information overflow. Several techniques exist for summarizing data without visualization, but these methods are not capable of quickly illustrating the full story behind the data. For instance, Anscombe's quartet consists of four datasets with nearly identical descriptive statistics that differ significantly when plotted [Ans73]. Visualization can not only uncover hidden patterns within data, but even the act of visualizing and manipulating visualizations can amplify cognition, helping users make sense of data [CMS99]. Unfortunately, the data is too complex for traditional visualization techniques in many scenarios.

In particular, high-dimensional data presents a unique challenge for visualization. While there are a number of techniques for displaying multidimensional data, very few scale past even twenty dimensions [HG02, FL03, LMW$^+$17]. Much modern data consists of hundreds, if not thousands, of dimensions. As a result, many practitioners turn toward dimensionality reduction (DR) techniques, commonly referred to as projections. These methods represent the high-dimensional data in a lower-dimensional space while preserving some measure of the underlying structure of the data [NA19, EMK$^+$21]. Unfortunately, many of these methods are complicated

and require careful tuning and interpretation to yield accurate results [WVJ16].

This dissertation provides novel algorithms to help users project complex, high-dimensional data. For many domain scientists, their first method of viewing a new dataset, found or created, is to project the data using one of the newer nonlinear dimensionality reduction techniques such as t-SNE [vdMH08] or UMAP [MHSG18]. For both methods, the user is given a litany of choices and hyperparameters, but practitioners usually stick to the defaults. This practice results in a 2D matrix with as many rows as the original dataset and two columns. These are typically treated as X and Y coordinates and plotted on a scatterplot. Sometimes, this can already tell a story about the data—clusters or specific patterns may emerge. But most likely, if the plot tells a story, it is only part of it. Additionally, it may be hard to make sense of the resulting plot since popular nonlinear dimensionality reduction techniques have uninterpretable axes and can contain significant distortions [NA19].

Complicated nonlinear DR techniques have several important hyperparameters. The most relevant to practitioners that use t-SNE [vdMH08] or UMAP [MHSG18] is the hyperparameter that trades off between local and global structure, perplexity, and "number of neighbors", respectively. The hyperparameter "number of neighbors" determines the number of neighboring points considered when attempting to preserve structure in the low-dimensional representation, and perplexity can be viewed as a smooth measure of the same concept. While there might be a magic value for each hyperparameter that allows a practitioner to explore the relationships that they are interested in, they will need to consider multiple values of this hyperparameter to get the whole picture [WVJ16]. This process requires recomputing these costly projections many times, which can be computationally infeasible for many modern large datasets.

As previously mentioned, interpreting the resulting plot will also take a lot of work. Since the axes have no easily explainable meaning (unlike methods such as PCA [Pea01]), users can have a hard time navigating the space, especially reasoning or hypothesizing about the spaces within the resulting plot that are empty or sparsely populated. Not to mention that the rate of change between the distance traveled in

the low-dimensional space and the distance traveled in the high-dimensional space is not uniform within the projection, resulting in small gaps within the projection that can mean a considerable difference between points in the original data space. Finally, these projection methods introduce errors that are not readily apparent upon initial inspection of the resulting projections.

The thesis introduces three methodologies that assist users in both efficiently exploring multiple hyperparameters and gaining a better understanding of their projection. These methodologies are designed to meet the needs of domain practitioners who utilize projection methods, particularly in visual analytics applications.

The first is HyperNP, a technique that allows real-time interactive hyperparameter tuning and exploration of projection methods by training neural network approximations. It can be thought of as a surrogate model for a projection technique such as t-SNE [vdMH08] or UMAP [MHSG18], which alleviates some of the computational burden by approximating the results. The model can be trained on a fraction of the total data instances and hyperparameter configurations that one would like to investigate and can compute projections for new data and hyperparameters at interactive speeds. Most importantly, it can compute projections for hyperparameter values unseen during training without the heavy recomputation inherent to general projection methods. Once a HyperNP model has been trained, the compute time required for the inference step is minimal. This fast computation allows for real-time visualization of large data sets and interactive exploration of different hyperparameter values associated with the projection operator.

NNInv is a machine-learning technique for computing the inverse of any projection. This method helps users explore projections by allowing them to query the projection space for corresponding high-dimensional data points interactively. Applications for this work are numerous. First, inverse projections can be used to explore the "empty" spaces in a 2D projection of high-dimensional data. While the user interactively brushes such spaces, high-dimensional instances corresponding to the visited 2D points are synthesized and displayed, thus allowing them to form a better mental map of how the 2D image represents the entire high-dimensional

data space beyond how a 2D scatterplot represents a high-dimensional dataset. Second, the speed at which NNInv is capable of inference opens the door to accessible "dense-map" visualizations. We can inverse-project each of the 2-D coordinates within the scatterplot, and the resulting high-dimensional representation can be used in downstream calculations for various dense-map visualizations. For instance, we can present a visualization of the rate of change within a projection—how far one travels within the high-dimensional space given a unit of distance in the pixel space. This type of visualization can help users find projection artifacts often present in UMAP [MHSG18] or t-SNE [vdMH08] visualizations.

The third is DimBridge, which is a visual analytics tool that allows users to interact with visual patterns in a projection and retrieve corresponding data patterns. DimBridge supports several interactions, allowing users to perform various analyses, from contrasting multiple clusters to explaining complex latent structures. Leveraging first-order predicate logic, DimBridge identifies subspaces in the original dimensions relevant to a queried pattern and provides an interface for users to visualize and interact with them. DimBridge can directly explain a pattern within a projection, helping users overcome the challenges associated with interpreting visual patterns in projections.

Using these three techniques, practitioners can maximize the value of their projections by exploring their local and global structure in a scalable fashion. They also gain the tools to understand the projection and its distortions better.

The dissertation begins by discussing recent work on dimensionality reduction, works that help users explore and understand the resulting projections, and visual analytics applications that utilize these techniques in Chapter 2. Then, Chapter 3 motivates for the thesis, discussing many opportunities I have had to work with real-world users who needed to make sense of complex data. While only some of these users worked with truly high-dimensional data, all of these experiences have motivated my work and shaped my approach.

Chapter 4 examines an interview study conducted about communication breakdowns between data scientists (DS) and subject matter experts (SMEs). It

begins by motivating the research and explaining the breakdown in communication between data science experts and subject matter experts caused by complicated data and complicated explanations. These data scientists specifically struggled to communicate the results of machine learning results to subject matter experts. As a result, the machine-learning models that had been developed were not being utilized for production, as data scientists were unable to communicate the benefits and risks of their work effectively. Collaborating with a large pharmaceutical organization, we interviewed data scientists and subject matter experts on the challenges of communicating regression results. The resulting interview transcripts were coded to identify not only points of friction but also potential solutions. These findings were consolidated into design guidelines, which helped the organization's data scientists keep to best communication practices and brought a number of these challenges to light within the visualization and machine-learning communities. One of my main takeaways was that understanding complex data is difficult, and complicated visualization techniques are often too complicated to help users make sense of their data. In addition, participants admitted it can be challenging to ask for clarification or help in these scenarios. This overwhelmingly points towards the need for more accessible visualizations that offer more tools for users to understand the visualizations themselves.

Chapter 5 discusses the HyperNP algorithm in depth. Most importantly, it explains how HyperNP can be leveraged by end users and designers of visual analytics applications to help users make sense of their data. It also illustrates the resulting plots that can be produced by HyperNP and quantitatively evaluates the method. Chapter 6 dives into NNInv, illustrating several applications of the process, focusing on both interactive techniques for understanding and static dense-map visualizations it enables. Chapter 7 discusses DimBridge, demonstrating how the tool and predicate logic can be utilized to explain projections. Finally, Chapter 9 explains how HyperNP and NNInv are the first steps towards a unified framework for based dimensionality reduction within visual analytics.

Chapter 2

Related Work

We divide our discussion of related work into five general topics: high-dimension data visualization, projection techniques, hyperparameters within projection methods, making sense of projections, and inverse projections. Further background about data science and domain science collaboration can be found in Section 4.2.

2.1 High-Dimensional Data Visualization

A number of general techniques for multi-dimensional visualization have been presented over the years. One of the most common families of methods is multiline graphs, which plot a number of features either overlaid or stacked vertically over another dimension. An example of this is the parallel coordinates plot [Ins85, HW13], which puts each dimension on a separate axis and draws a line connecting these axes for each instance. Parallel coordinates are often used in concert with sophisticated interaction techniques for selecting groups of data items, e.g., angular brushing of dimensions [HLD02], multi-way brushing for high dimensional pattern discovery [RLS+18], as well as augmented designs that better convey relationships between dimensions [BKS20]. Similar to parallel coordinates, Andrews Curves [And72] transform the data via a projection onto the Fourier basis and, subsequently, can be viewed as a series of curves, one for each data item.

One of the earliest attempts at multi-dimensional plots fit under the um-

brella of small multiples, where a number of 2d plots are used together to represent multi-dimensional data. These include the scatterplot matrix [Har75], the permutation matrix of bar charts [Ber83], and histogram matrices. Small multiples are closely related to the idea of dimension stacking [LWW90], wherein each dimension is broken into histogram-like buckets, with further dimensions recursively partitioned within preceding dimension buckets. Faceting data, or dimensions, across multiple views quickly leads to scalability issues, and thus methods for selecting informative views [WAG05], used in conjunction with visual quality metrics to rank views [BTK11], are common approaches to handle such problems. If limited to a single view, then scatterplots that encode multiple dimensions of data can be designed using star coordinates [Kan00, LT13] or RadViz coordinates [RSRDS15] via the spatial arrangement of axes in the 2D plane, coupled with a scheme for computing 2D coordinates for data items.

A more thorough discussion of these and similar techniques can be found in the following surveys [HG02, FL03, LMW$^+$17]. A central advantage of such high-dimensional data visualization methods is the positional encoding of individual data dimensions. This allows for detected visual patterns to be readily understood in terms of the data dimensions, facilitating the discovery of high-dimensional patterns. Although these techniques can scale in a number of instances, general visualization techniques do not scale well to high-dimensional data, often requiring users to select a subset of dimensions for visual analysis.

2.2 Overview of Projection Techniques

Also called Dimensionality Reduction (DR) methods, projections are techniques that aim to go beyond the aforementioned limitations of high-dimensional visualization techniques [GKW$^+$08, vdMPvdH09, JCC$^+$11, SPN12, SVM14, SZS$^+$16, JZF$^+$09]. Projections scale well both in the number of samples N and dimensions n, enabling their wide use for visualizing high-dimensional data. Let $D = \{\mathbf{x}_i\}$, $\mathbf{x}_i \in \mathbb{R}^n$, $1 \leq i \leq N$ be a n-dimensional dataset. Its N points $\mathbf{x}_i$ (also called samples or observations)

each have n dimensions (also called attributes). Formally put, a projection technique $P : D \to \mathbb{R}^m$ is a function that maps every point $\mathbf{x} \in D$ of a high-dimensional dataset to a low-dimensional counterpart $P(\mathbf{x})$. Typically $m \in \{2, 3\}$, which allows directly depicting the projection $P(D) = \{P(\mathbf{x})|\mathbf{x} \in D\}$ as a 2D or 3D scatterplot, respectively. Projection techniques P aim to preserve the so-called *data structure* between the original dataset D and its low-dimensional counterpart $P(D)$. Hence, by examining this scatterplot, users can reason about data relationships in D. Structure is captured in terms of inter-point distances [RS00, PNML08b, JCC+11], point neighborhoods [vdMH08, MHSG18], or clusters [PM06]. Projections can be further classified as linear [CG15] *or* nonlinear [Yin07, vdMPvdH09]. Linear techniques, such as PCA, are simple and fast to compute, have an intuitive geometric interpretation, and robust association with statistical analysis. Nonlinear techniques, such as UMAP, are generally more computationally expensive but strive to represent local neighborhood information with minimal distortion. There are also a number of projection techniques that generally fit under the RadViz family [HGM+97, ABL+19, PT19]. RadViz is able to visualize multidimensional data in 2D by anchoring each feature around the perimeter of a circle and leverages spring forces from those points to assign each instance a location inside the circle.

Projection techniques can be organized into several taxonomies, thereby providing ways for practitioners to compare and choose a technique based on specific requirements. Van der Maaten *et al.* [vdMPvdH09], and Cunningham *et al.* [CG15] show how P can be computed by several types of non-linear, or respectively linear, optimization methods that offer different tradeoffs between computational scalability and projection quality. Sorzano *et al.* [SVM14], and Engel *et al.* [EHH12] propose two taxonomies of projections based on their implementation aspects, in particular the cost functions they optimize to compute P, and the choices of available optimizers and their computational complexities. Projection techniques are further classified, analyzed, and compared both theoretically and practically in a number of surveys [HG02, BBH12, LMW+17, NA19, EMK+21], to which we refer the reader.

All projection techniques P transform data between the original space D

and the projection space $\mathbb{R}^m$. A projection's effectiveness is a combination of how well $P(D)$ captures data patterns present in D and how (easily) users actually perceive the patterns in $P(D)$. Heulot *et al.* [HFA17], and Nonato and Aupetit [NA19] classify the different types of errors that projection techniques create and how these impact different classes of visual analytics tasks. Separately, several authors propose perception models [AEM11, TBB+10, WFC+18] and visual quality metrics and visualizations thereof [LV09, MLGH13, SA15, AS16, CHAS18] to help users understand the quality of the projections they compute, as determined by the chosen algorithms. Error metrics and subsequent visualization mechanisms include trustworthiness and continuity [VPN+10], false and missing neighbors [MCMT14, MMT15a], and false neighborhoods and tears [Aup07, LA11]. Several works propose benchmarks which measure the quality of projection techniques across a selection of datasets [vdMPvdH09, NA19, EMK+21]. These provide guidelines for practitioners to choose a suitable technique depending on the type of quality metrics they want to maximize for a given dataset type.

Choosing a good projection – one which yields a low projection error on a given family of datasets, is simple to use in terms of parameter setting, is robust to small changes in the data D, and is computationally scalable to large dimensions d and sample counts N – is challenging. Recently, Neural Network Projection (NNP) [EHT20] was proposed as a method to achieve these goals by leveraging deep learning: Given a dataset D, a small subset $D_S \in D$ is chosen and projected by any user-chosen technique P. After a suitable projection $P(D_S)$ is obtained by tuning P's parameters, a fully connected feed-forward neural network is trained to infer $P(D_S)$ from D_S. The trained network is then used to project any data drawn from a similar distribution as D_S. NNP has shown remarkable ability in producing projections that mimic a wide range of techniques P on many types of datasets, with little or no parameter tuning [EHFT20a]. Moreover, NNP is parametric, making it robust to small-scale data changes in D while also providing an out-of-sample capability – that is, NNP learns a *continuous* function $P : \mathbb{R}^n \to \mathbb{R}^m$ with $n << m$ rather than a discrete mapping formed by a non-parametric projection. HyperNP

builds upon NNP, adding the ability to project across multiple hyperparameters after training a single network.

2.3 Hyperparameters of Projection Techniques

The function P depends not only on D, but also on a set of *hyperparameters* $\mathbf{h} = \{h_i\}$. These hyperparameters control several aspects of P as follows. Computing a *global* mapping P between $\mathbb{R}^n$ and $\mathbb{R}^2$ that has overall high quality is, in general, not possible. As such, many projection techniques construct different mappings for different small-scale neighborhoods in $\mathbb{R}^n$. For example, in several projection techniques (including UMAP, Isomap, and others), a scalar parameter h controls the neighborhood size.

Locally Linear Embedding (LLE) [RS00] fits a hyperplane through each point and its nearest neighbors, keeping local relationships linear but allowing the global structure to be nonlinear. Isomap [BS02] tackles the problem of projecting curved manifolds by estimating geodesic distances over neighborhoods and using these as a cost function to derive the projection. Least Squares Projection [PNML08a] projects a subset of landmarks and then uses a fast Laplacian-like operator to map the remaining samples. Two Phase Projection [PSN10], LAMP [JCC+11], and the Piece-wise Laplacian Projection [PEP+11] use a similar approach. t-SNE [vdMH08] preserves neighbors during the projection by minimizing the Kullback-Leibler divergence between neighborhood probabilities in $\mathbb{R}^n$ and $\mathbb{R}^2$. UMAP [MHSG18] models the $\mathbb{R}^n$ manifold over small neighborhoods with a fuzzy topological structure and searches for a 2D representation that has the closest possible fuzzy topological structure. All these neighborhood-based approaches depend on hyperparameters $\mathbf{h}$ that model neighborhood sizes or number (and selection) of landmark points.

All of the discussed works mention that finding good hyperparameter values is challenging. Espadoto *et al.* [EMK+21] recognize this problem and address it by proposing optimal and preset parameter values computed by grid search over the hyperparameter space. However, this exhaustive search is expensive. More impor-

tantly, certain projection methods do not have *globally* optimal presets. Wattenberg *et al.* [WVJ16] illustrate this for t-SNE's perplexity parameter whose variation can create projections that emphasize different aspects of the n-dimensional data. However, experimenting with many hyperparameter values is a costly process, especially for projections that take a long time to compute.

Deep learning methods are effective for scalable DR [HS06, SLZ12]. Kwon *et al.* [KM20a] demonstrated success using a novel autoencoder to generate graph layouts. Their method enables the exploration of the latent space of graph layouts. More recently, NNP [EHT20, EHFT20b] used deep learning to mimic any projection technique P by training on $P(D')$ for a small subset $D' \subset D$. In addition to providing fast computation, such approaches also have the ability to project *out-of-sample* data. Furthermore, due to the neural-network-based approach, inference computations are parameter-free. However, this is not always desirable. As observed by Wattenberg *et al.* [WVJ16], users may want to control $\mathbf{h}$ to gain insight into their data or to illustrate local versus global patterns (e.g., by changing the neighborhood size). Systems such as VisCoDeR [CHAS18] and t-viSNE [CMK20] allow users to explore how changes in hyperparameters affect projections, but their underlying projection methods do not offer the aforementioned benefits of NNP. HyperNP addresses the challenges of hyperparameter exploration and out-of-sample projection in a naturally scalable framework.

2.4 Making Sense of Projections

The use of nonlinear DR [vdMH08, MHSG18] has become an important part of interface design for visual analytics [SZS+16]. In these methods, high-dimensional similarity, such as a data item's proximity to its nearest neighbors, is intended to be faithfully represented as spatial proximity in the 2D projection space. Spatial proximity alone, however, lacks the context for *why* points are positioned, e.g., what makes two data items similar? There are methods for bringing in the context of individual data dimensions exist, e.g., as per-dimension aggregations arranged in multiple

views [SDMT15], dimension-specific spatial groupings in the projection [SSJ+21], or augmenting the projection space with 2D lines [CD18b] or curves [FGS18] that encode large variations in the data space.

Several techniques aim to show errors in this process visually, *i.e.*, areas in $P(D)$ that may miss or not reflect actual structures in D. For example, Stress Maps [SSK10] is a visual analysis tool that displays the local stress values, or how local distance relationships have changed, under a projection algorithm. In order to demonstrate potential changes of $P(D)$ caused by a hypothetical perturbation of the data in D, DimReader [FGS18] utilized a filled contour plot in the background. Probing Projections [SDMT15] allows users to display the value of any attribute through a background heat map and also enables users to correct distance errors in the projection by moving individual points in $P(D)$ on the 2D projection space. A similar technique is proposed by LAMP [JCC+11]. In contrast, Dis-Function [BLBC12] updates the mapping P from the user's dragging of data points to generate new, and hopefully better, projections. Sirius [DWF+19] allows practitioners to investigate both the observations and attributes of a dataset through symmetric projections.

The motivation for better understanding a projection is often driven by the visual patterns found within the scatterplot. In this setting, interpreting the meaning of a cluster of points is a common objective [WCR+17], with numerous methods using a projection as a scaffold for interactively selecting clusters and in turn, visually comparing clusters [CD18a] or performing a contrastive cluster analysis [FKM19, FWZM21]. However, as DR methods are prone to error, cluster analysis methods that convey the distortion induced by DR [SZJ+21, JAL+22] remain important for proper interpretation. Observations made through cluster analysis inform methods for steering DR [XHL+22, EHA+22], often performed in response to imperfections found in projections, reflecting an iterative process of (1) visual analysis of projections and clusters, and (2) data annotation. Although these methods can facilitate the understanding of projections, often, the provided context is limited to summaries over individual data dimensions [SDMT15, CD18a, FWZM21].

NNInv can produce estimations of samples that live within the empty space

of the projection, as well as dense-map visualizations that showcase distortions or errors. This capability is easily combined with HyperNP to help users understand multiple projections across a number of hyperparameter values.

2.4.0.1 Inverse Projection

Inverse-projection can be seen as a function $P^{-1} : \mathbb{R}^m \to \mathbb{R}^n$, which should ideally be the mathematical inverse of a given projection P, *i.e.*, $P^{-1}(P(\mathbf{x})) = \mathbf{x}, \forall \mathbf{x} \in D$. A crucial component of inverse-projections is that they *should* have an out-of-sample ability that can be expressed as a continuous mapping, which is generally not the case for direct (discrete) projections. Thus, P^{-1} can be used to invert points that fall *between* the points of the scatterplot $P(D)$, helping the user to understand what kind of data samples could project at a particular location in $P(D)$. This ability further supports applications such as data augmentation and classifier exploration [MHT18, REHT19].

Inverse-projection is inherently harder than direct projection due to the need for an out-of-sample ability and the fact that P^{-1} needs to synthesize a high number of dimensions d from a lower dimension m. Early on, autoencoders were proposed to jointly infer both P and P^{-1} by deep learning to minimize the projection error from $\mathbb{R}^n$ to $\mathbb{R}^m$ [HS06]. While autoencoders are parametric, the resulting mappings are not always intuitive [vdMPvdH09] and autoencoders can be difficult to train [VGS+20]. Amorim *et al.* approach inverse-projection in iLAMP [dBD+12] by using local affine transformations, following the earlier idea of the LAMP direct projection technique [JCC+11]. Mamani *et al.* also use local affine transformations in their inverse-projections as a part of their work on user-driven feature space transformation [MFNP13]. iLAMP was later extended by leveraging radial basis functions (RBFs) to provide a smoother inverse mapping P^{-1}, which was shown to be useful for data augmentation [ABMC+15]. Kriegeskorte and Mur [KM12] proposed inverse MDS, which infers pairwise dissimilarities from multiple 2D arrangements of items. Cavallo *et al.* [CD18b] used a type of inverse-projection in Praxis, an interactive exploratory analysis tool for high-dimensional data. The authors leveraged

the analytical inverse of PCA in addition to an autoencoder to both project and inverse-project data. Similarly, Zhao *et al.* [ZFF20] used a Grammar Variational Autoencoder (GVAE) [KPHL17] to project and inverse-project data charts for steering exploratory visual analysis. NNInv can be used to inverse-project any projection method, not relying on a tailored projection, and performs faster than other methods (enabling intensive background visualizations like dense maps).

Chapter 3

Motivation

The proliferation of data is occurring at an unprecedented pace. With the right tools for examining this data, practitioners across fields can make discoveries in much shorter time horizons. As discussed in Chapter 2, researchers are developing innovative new techniques for examining high-dimensional data, which has previously been challenging to investigate. My experiences working with users of complex data have underscored the importance of taking a human-centered approach to technique development to ensure we, as a field, are moving toward addressing real-world challenges to adoption and implementation.

I will begin this section with three illustrative stories from my experiences working with academic, industry, and government researchers. In each of these cases, subject matter experts collaborated with data scientists and grappled with the challenges of visualizing complex data. I'll discuss how we addressed a situation where new, better-performing models were not being deployed due to communication challenges at a large pharmaceutical company. At a federally funded research and development center (FFRDC), we talked with practitioners–primarily working on cybersecurity–to understand what sort of visualizations worked for their needs; we discovered the most popular and complex visualizations (node-link diagrams) were largely unsuccessful. Finally, I'll share how I collaborated with organic chemists at Northeastern University to use interactive visualization to speed up the search for novel molecules for material chemistry and pharmacology applications.

This section will conclude by explaining how these collaborations informed my belief that domain researchers need better tools to analyze high-dimensional data and support to employ these tools successfully.

3.1 Communicating Machine-learning Predictions to Subject Matter Experts

At a large pharmaceutical company, data scientists had been investing in developing machine learning models to provide predictions to subject matter experts. Despite these models proving to be better performing than existing strategies, leaders found that they were not being utilized for production. The core obstacle was that data scientists were unable to communicate the benefits and risks of these models effectively. In collaboration with the company, I interviewed data scientists and subject matter experts on the challenges associated with communicating regression results discussed further in Chapter 4). Through these conversations, we identified both points of friction and potential facilitators for future implementation efforts. These findings were consolidated into design guidelines, which helped the company's data scientists implement best practices in communicating with subject matter experts and revealed several challenges within the visualization and machine-learning communities.

What we learned about Subject matter experts' preferences has informed my philosophy around **non-technical** user's visualization needs. The SMEs in our study preferred visualizations valued interaction, and relied on example data instances and explanations to make sense of complicated data. For example, we learned that SMEs reported visual explanations to be helpful for their understanding and expressed a preference for interactive tools over static visualizations. Additionally, they preferred concrete examples of data points visible within the graphics and further explanatory techniques embedded within the visualizations.

However, one of my most critical takeaways from this experience is that challenges with understanding complicated data are likely under-reported. Practitioners were frank about their reluctance to express a lack of comprehension during data

scientists' presentations of new predictive machine learning models. One SME stated, "I think the end-user, a good chunk of the time, would be too embarrassed to say, 'I don't get what you're talking about". Further, SMEs' desire for embedded concrete examples of data points in visualizations and additional explanations showcases that while visualization is a powerful tool, visualizations without adequate user-focused considerations can become too complicated for end-users. Truly high-dimensional datasets exacerbate these issues, as the more complex techniques required add to the complexity.

3.2 Documenting Practitioner Needs with Knowledge Graphs

Collaborators at an FFRDC were interested in identifying challenges associated with using knowledge graphs (KGs) in data analysis. Knowledge graphs have emerged as a popular approach, including with researchers at this FFRDC, to represent complex data for question-answering, recommendation systems, and other practical applications. However, there is a limited understanding of KG users, their challenges, and the limitations of current tools and visualization designs for KGs used in practice. We conducted a series of observational interview studies with KG practitioners–about half of whom are at the FFRDC–to characterize the typical personas of KG users, their expertise, tool usage, the obstacles they face when using KGs, and their unmet visualization needs.

One relevant insight I gained from this research was around complexity and user trust in visualizations. In particular, the most common KG design, node-link diagrams, fail to scale to real-world data. In our analysis, the "Consumer" KG user persona—which includes stakeholders and domain experts—expressed uncertainty in the results as the graphs' complexity level increased. As one KG user stated, "As soon as the complexity of the graph reaches a certain level on the screen, users really tend to shut down and not trust any of it." Given this lack of trust and scalability, consumers ultimately perceived little utility in KG's use for downstream tasks.

3.3 Helping Domain Scientists Understand and Explore Their Data Space

Organic chemists at Northeastern University were interested in speeding up the process of finding new light-activated azoarene photoswitches. Photoswitches are molecules that absorb visible light and are used for materials chemistry, photopharmacology, and catalysis applications. The chemists required a visualization that could help them understand the *space of molecules*, including molecules that have been explored, novel molecules under consideration, and unknown molecules suggested by a machine-learning algorithm. To address their needs, I developed an interactive system with UMAP to help the organic chemists focus their search, locating areas of the space that had been under-explored. A static plot inspired by this process was later created for a publication to communicate to the organic chemistry research community the clusters of molecules that were most promising for future work [MNA+21].

The process of collaborating with the chemists to create UMAP visualizations elucidated specific user interaction challenges with the tool for my future research. The chemists were very explicitly interested in analyzing their data using UMAP to (1) understand the space being searched by an algorithmic process looking for azoarene photoswitches and (2) visually communicate their findings. When we first spoke, they struggled to find the right hyperparameters for UMAP. Offering them a tool that allowed them to quickly view several different plots at once and compare and contrast allowed them to learn about the local and global structure and find a final plot for the paper.

3.4 Takeaways

Each group of practitioners in the examples above dealt with different types of data, whether it be low-dimensional tables or high-dimensional graphs or tables. Throughout these experiences, I observed themes in the users' challenges in making

sense of the data. If users are experiencing obstacles with trusting, interpreting, and communicating low-dimensional visualizations, it follows that addressing these challenges will be foundational to more complex high-dimensional visualizations. The significant opportunities to improve user experience with high-dimensional data visualization that my thesis addresses include:

- **Exploring the Projected Space**: Users often use projection methods to understand their data better. As seen at the pharmaceutical company and Northeastern, examples of individual data points–especially embedded within a specific visualization–would help users understand the space of their projection.

- **Examining Multiple Plots**: Users of non-linear projections often need to explore multiple hyperparameter values to understand their data fully; in neighborhood-based methods, for example, trade-offs exist between preserving local and global structure. As we saw when working with collaborators at Northeastern, practitioners need assistance exploring potential plots outside the defaults.

- **Understanding Errors and Distortions**: Projections are complicated techniques that often necessarily introduce distortions or errors to the resulting plot [NA19]. My work with the pharmaceutical company and FFRDC illustrates that practitioners need guidance to identify and assess distortions or errors so that they can employ complex visualizations confidently.

During my PhD, I developed two novel algorithms and a system for working with projections, HyperNP, NNInv, and DimBridge. HyperNP allows fast approximation of any projection method across several hyperparameter values, making it a perfect tool for scientists hoping to example multiple plots of the same dataset. NNInv provides fast inverse projection, which offers two benefits. NNInv makes it easy to interactively explore the projected space by generating examples of data instances across the empty space of a projection. Additionally, it enables dense-map visualizations that can embed information about distortions and errors directly into the

resulting plot. The system, DimBridge, goes a step further than NNInv, allowing the user to query the projection for explanations directly. It uses predicates to fit this explanation into a low-dimensional space plotted in a SPLOM. Together, these three contributions work towards solving the challenges outlined above.

Chapter 4

Communication

This chapter outlines the challenges inherent to helping users understand complex data. These challenges derive from an interview study focusing on communicating complicated machine-learning results to subject matter experts (SMEs). The study's central motivation was to help data scientists communicate these results more clearly, thereby helping SMEs make informed decisions about deploying these models.

We conducted this study to support a large pharmaceutical company that ascertained that many of their machine-learning models, which data scientists believed outperformed current strategies, were not making it to production due to communication challenges. To address this challenge, we tried to understand how visualization could improve the communication of this complex data.

The rest of this chapter discusses the interview study in depth, explaining the methodology, outcomes, and resulting guidelines in detail. These discussions with subject matter experts and data scientists are not explicitly about high-dimensional data but were paramount to the final philosophy I adopted regarding high-dimensional data in my work.

4.1 Problem Statement

The widespread use of artificial intelligence (AI) has reached far beyond academia. In a recent study with over 70 Fortune 1000 companies, the percentage of firms

investing at least $50 million in AI increased by over 60% from 2018 to 2020 [Alg21]. Surprisingly though, despite increased investments in AI, only **15%** of these firms reported that they had *actually deployed* AI capabilities into widespread production. Similar findings were echoed by Ransbotham et al. in an MIT SMR report: 7 out of 10 companies reported minimal to no impact from their organization's AI use, with only 10% of companies seeing significant financial benefits from implementing AI solutions [RKK+20].

The potential benefits of AI and machine learning (ML) models have driven great interest across a variety of domains, such as healthcare [JJZ+17], biology [Web18], and commercial industries [FS19]. While companies invest millions into AI and ML production, it is evident that a gap still exists between the promise and the actual execution of these models in practice [Alg21]. Data science interviews[1] and qualitative studies [PJ18, HHB20, Ber19] consistently cite *strained communication* between data scientists and subject matter experts as a top barrier to model development and deployment.

Communication between members of different disciplines and expertise is a difficult task that is often facilitated by visualization [Smi12, BKVR+20, KM13]. However, while visualization has acted as a powerful communication medium for broad audiences [SH10], most research in visualization for AI/ML has focused on helping data scientists develop, debug, and improve their models [SKKC18]. We believe that a more fundamental visualization question needs to be answered for AI and ML development: how can we best communicate the strengths and weaknesses of a predictive model's performance to subject matter experts, who ultimately decide what to do with it?

To investigate how visualization can mitigate issues in model communication – which ultimately lead to downstream problems with deployment – we conducted two interview studies with data science practitioners who develop models, and the subject matter experts (SMEs) that must work with the outputs of those models. From a thematic analysis in which we iteratively coded and categorized our first

[1] https://hbr.org/2019/01/data-science-and-the-art-of-persuasion

round of interviews, we present the most prevalent themes related to the challenges and unaddressed communication needs of SMEs in model collaborations.

We find that, while data scientists and SMEs are often concerned about similar issues (*e.g.*, strengths and weaknesses of a model, edge cases, etc.), each group approaches these issues drawing upon very different sets of language, context, and experiences. SMEs reported feeling overwhelmed by data science presentations, discouraged from asking clarifying questions, and skeptical about whether a predictive model actually fits their needs. Consequently, this lack of a common framing results in misunderstandings, miscalculations of a model's risks, and in some cases the failure of a model's adoption. While data visualization was cited as a tool employed by data scientists to better communicate model performance results, our interviews revealed that many commonplace predictive model visualizations (*e.g.*, residual plots) may result in greater confusion by SMEs, particularly when they are not already familiar with technical data science concepts or visualization methods.

To combat this confusion, data scientists diverted to presenting performance metrics, such as explained variance or mean squared error, to illustrate their model's performance in a presentation. However, we observe that the presentation of these metrics alone is insufficient for SMEs who are responsible for making decisions and recommendations based on the model's outputs. We suggest that more easily interpretable visualization methods, such as the use of annotations and illustrative grounded use cases, can more effectively communicate a model's performance to SMEs beyond error metrics. As a result, these methods can provide a clearer picture of the model's impact and allow SMEs to better understand and assess its outcomes.

From our interview findings, we distill a set of model communication guidelines for data scientists to consider when presenting the complexities of a model's performance. We then demonstrate an application of our guidelines in a data science presentation with four visualizations, and elicit feedback from SMEs. We find that the use of visualization enabled SMEs to have "*quick*" and "*instant*" insights beyond text and numbers. Furthermore, when presented on a model's performance using our guidelines, SMEs felt more comfortable asking questions, better equipped to

understand the model's strengths and weaknesses, and ultimately empowered to use their domain expertise to make a more calculated interpretation of the model.

In summary, our major contributions are as follows:

- Two interview studies with data scientists and SMEs who regularly collaborate on predictive models.
- Guidelines that emerge from the analysis of our interviews and review of related literature.
- Example slides and visualizations, as well as an open-source Observable notebook[2] used to generate those visualizations, so that our guidelines are easy to implement for data science presentations.

4.2 Background

For the remainder of this chapter, we broadly refer to any practitioner who works on and builds predictive models as a *data scientist*. We refer to *subject matter experts (SMEs)* as model consumers with primary roles and expertise outside of data science and machine learning. We differentiate between model consumers who take the role as 'executives' (*i.e.* stakeholders) and those whose primary function is as a domain scientist (SME) that works with data scientists to develop and deploy predictive models in their daily work. Although some SMEs may also be stakeholders, we make the distinction in this chapter that an SME has a specialized set of knowledge for a particular scientific domain.

4.2.1 Visualization for Model Interpretability

Effective presentation of a predictive model's performance to domain scientists, SMEs, and other stakeholders is an active topic of research in visualization. Researchers studying explainable AI seek to help users interpret and explain the inferences of AI models by visualizing the internal workings of those models [MZR18, Lip18, WvIP19]. Metrics and principles are posed for explainable AI [RCC+22, HMKL18], guidelines

[2] https://observablehq.com/@ashleysuh/model-comm-vis

for defining *interpretability* are suggested [DGSA20, YHSA20], and visual analytic tools can enhance machine learning and AI transparency [SKKC18, CMJ+20, HSD19]. Hohman et al. conducted an interview study with machine learning experts to understand how interactive interfaces could better support model interpretation for data scientists [HHC+19]. While many of these works are relevant to the research presented here, the concepts of explainability are discussed at too low of a level. Consequently, a majority of ML interpretability support is targeted towards supporting experts in debugging and improving models (*e.g.*, [SKKC18]) and not necessarily in communicating the limitations or weaknesses of models to the stakeholders – or consumers – of ML [SMB19, NDL+23].

From the results of our first interview study (Sections 4.3 and 4.4), we found that solutions were needed to facilitate model communication between data scientists and SMEs at a higher level, particularly when SMEs are responsible for adopting and monitoring those models in practice. To this end, we identify what data scientists and SMEs each find most valuable in the presentation of a model's outcome, and distill common barriers SMEs face when receiving and interpreting presentations on model performance. From these findings, we propose visualization guidelines (Section 4.5) that practitioners can use when communicating models.

4.2.2 Data Science & ML in Practice

The desire and demand for AI/ML at organizations outside of big tech is well documented [Alg21, HB21]. Consequently, there has been a sharp increase in the use (and subsequent risk) of advanced analytics in nontraditional ML domains (*e.g.*, for healthcare and childcare) [Eva00, Alm07, ZLVV21]. However, when introducing AI/ML techniques in practice, it is also necessary to investigate the risks, limitations, edge cases, and weaknesses of a predictive model before its deployment [Eva00, Alm07]. Although strides are continuously being made to improve the transparency of ML models, empirical research shows these techniques are rarely deployed in high stakes domains [HEB+02, CS13, EW19, SRS20]. Close to our work is the recent design study by Zytek et al. [ZLVV21], in which the authors

collaborate with SMEs that use ML techniques in child welfare. In Zytek et al.'s work, an iterative design process combines ML practitioners and childcare experts to identify key challenges in their workflow. The final result is a visual analytics tool to help alleviate interpretability challenges for the domain experts. In contrast to Zytek et al.'s work and similar characterizations of ML interpretability challenges, we instead focus on communication bottlenecks between data scientists and SMEs that prevent model assessment and deployment.

Challenges in ML collaborations directly affect the eventual production of AI/ML solutions, particularly when the model's performance can not be well understood. Seneviratne et al. argue that work is needed to bridge the implementation gap of machine learning by merging ML algorithms into the 'socio-technical' milieu of the organization [SSC20]. Similar work describes widespread systematic issues in the adaption of predictive models at healthcare organizations, citing the mismatch between stakeholders and their understanding of technological innovations [CS13]. Shah et al. suggests that the utility of ML algorithms could be better demonstrated in practice if the end-users could better assess the performance of a predictive model without relying on standard performance metrics [SMB19]. In this chapter, we intentionally study how the performance of a predictive model can be effectively communicated to SMEs through visualization. Visualization and visual communication is often deployed to bridge communication and interpretability gaps, particularly when an audience may not have a similarly technical background [BKVR+20, KM13, Smi12]. This reasoning has been instrumental in guiding us to improve the accessibility of predictive models through the careful application of visualization for data science and SME communication.

4.2.3 Characterizing ML Collaborations

Across enterprise surveys that highlight common challenges for AI/ML collaborations, communication is continuously cited as the most difficult to overcome [Alg21]. Research in ML, visual analytics, and human factors have attempted to characterize the precise nature of challenges surrounding collaborations between ML researchers

and domain scientists. After 6 months of fieldwork with a corporate data science team, Passi and Jackson found that data science practitioners had several frustrations concerning the communication with stakeholders [PJ18], such as the lack of trust from stakeholders when the results or inner workings of an ML model were being discussed. These challenges exist in other sectors outside of data science as well. Hopkins and Booth conducted an interview study with stakeholders from organizations outside of 'Big Tech' to understand how resource constraints challenge ML workflows and development [HB21]. The authors similarly found that language barriers between ML practitioners and stakeholders prevented trust and deployment of models and data science results.

Language barriers between data scientists and SMEs can occur at many stages of the ML collaboration process. Hong et al. conducted an interview study with ML practitioners on model interpretability and found that data scientists had difficulty assessing the prior expertise and knowledge level of SMEs when delivering results, presentations, and insights [HHB20]. Similar in spirit to our work, Mosca et al. presented a study of client-facing data scientists to identify the common communication strategies they employ to translate the (potentially ill-defined) analysis needs of SMEs [MRC+19]. In this paper, we focus primarily on how *predictive models* (rather than general data science findings) can be best communicated with and to SMEs. To this end, we interview both data scientists and SMEs to get a sense of current gaps in their practices. Further, unlike prior work that aims to characterize and offer broad solutions to common ML collaboration challenges, we distill and validate visualization strategies that reduce the burden of presenting and communicating model performance.

In recent work that surveys previous studies on ML collaborations, Suresh et al. suggest that the end-users for ML can be characterized, and thus better understood, beyond standard *expert* versus *non-expert* roles [SGNS21]. Specifically, the authors suggest that consumers of machine learning models (*e.g.*, stakeholders, SMEs) can be categorized by their expertise (formal, instrumental, or personal) and the contexts in which this expertise manifests (ML, data domain, or the general

milieu). To situate our work with Suresh et al.'s framework, we interview *subject matter experts*, not necessarily *stakeholders*, with formal knowledge of a particular data domain (*e.g.*, clinical safety or biology) and personal or instrumental knowledge of machine learning. Motivated by Suresh et al.'s work, we are able to investigate how communication can be improved between data scientists and those with specialized knowledge in a respective data domain (SMEs).

PID	Role	Education	Domain expertise	Experience with data science (1-5)	Frequency working with data (1-5)	Frequency using regression (1-5)
PID1	DS	Doctorate	Data science	(3) Familiar	(5) Every day	(3) 1-3x/week
PID2	DS	Doctorate	Data science	(5) Expert	(4) 1-3x/day	(2) 1-3x/month
PID3	DS	Masters	Data science	(5) Expert	(4) 1-3x/day	(3) 1-3x/week
PID4	DS	Doctorate	Data science	(5) Expert	(5) Every day	(3) 1-3x/week
PID5	DS	Masters	Data science	(4) Quite familiar	(5) Every day	(2) 1-3x/month
PID6	DS	Doctorate	Data science	(5) Expert	(3) 1-3x/week	(5) Every day
PID7	DS	Masters	Data science	(5) Expert	(3) 1-3x/week	(2) 1-3x/month
PID8	SME	Masters	Pharmacovigilance	(4) Quite familiar	(3) 1-3x/week	(2) 1-3x/month
PID9	SME	Masters	Pharmacovigilance	(3) Familiar	(5) Every day	(3) 1-3x/week
PID10	SME	Bachelors	Quality Assurance	(3) Familiar	(3) 1-3x/week	(2) 1-3x/month
PID11	SME	Masters	Commercial	(2) Somewhat familiar	(3) 1-3x/week	(1) Never
PID12	SME	Masters	Finance	(3) Familiar	(5) Every day	(2) 1-3x/month
PID13	SME	Masters	Quality Assurance	(4) Quite familiar	(5) Every day	(3) 1-3x/week

Table 4.1: Demographics for our interviewees in Section 4.3. Participants (DS=data scientist, SME=subject matter expert) self-reported their highest education, domain expertise, level of experience with data science (1=No experience, 2=Somewhat familiar, 3=Familiar, 4=Quite familiar, 5=Expert), as well as their frequency working with data and using regression models (1=Never, 2=1-3x/month, 3=1-3x/week, 4=1-3x/day, 5=Every day).

4.3 Interview Study

Our research explores how communication can be improved between data scientists and subject matter experts when their end-goal is to use a predictive model. We conducted an interview study at a major organization with two participant groups - data scientists who build and present the performance of predictive models, and subject matter experts who make decisions with these models in their daily workflow.

The goal of our interview study was to understand the types of challenges both data scientists and SMEs can experience when assessing predictive models. By comparing and contrasting themes in the responses of each participant group, we could then identify opportunities for visualization to bridge communication gaps

between the two groups and attain a shared understanding.

During the interviews, each participant was asked the same set of questions and given the same prompts (described in Section 4.3.2). Participants' feedback was elicited on a hypothetical modeling scenario, followed by a discussion of their own experiences in past model collaborations. In addition to barriers that affect model communication, we also investigated whether any visualization techniques or presentation styles had been helpful when data scientists communicate the performance of their models.

4.3.1 Participants

We solicited interview participants through email recruitment within the Data and AI division of a large pharmaceutical company. To produce visualization guidelines that could be broadly used by practitioners, we followed the 2017–2021 Kaggle Machine Learning & Data Science Surveys[3] in which *regression* was marked as the most common machine learning method used by practitioners. Therefore, we required all participants to have prior experience with *regression models* as a common baseline in our emails.

In our email exchange, potential participants were informed that the purpose of the interview was to discuss their experiences interpreting and communicating a regression model's performance. When soliciting SMEs, we specifically asked for "SMEs who do not build models themselves, but have prior experience using a regression model, *e.g.*, looking at its predictions, deciding whether to use it, etc." In contrast, when recruiting data scientists, we targeted those that "have worked with, assessed, or communicated the performance of a regression model previously." In total, we interviewed 7 data scientists and 6 subject matter experts in varying departments and data domains. Our participants' demographics, as well as their self-reported level of data science experience and frequency working with regression models, can be found in Table 4.1.

[3] https://www.kaggle.com/c/kaggle-survey-2021

4.3.2 Protocol

All of our interviews were semi-structured, lasting roughly one hour each. Interviews were conducted virtually on Microsoft Teams with audio only. Shortly before each interview, participants were given a copy of the consent form which contained information about the study, its design, and their rights as participants. Each participant verbally consented to the study over a recording and was given an anonymous demographics survey to complete.

Participants were shown a set of prepared slides to elicit feedback on their experience assessing and communicating the performance of regression models. Regardless of the participant's organizational role (data scientist or SME), the same set of questions and prompts were given during the interview. The slide deck used in our interviews, as well as all interview questions asked to participants, are included in our supplemental and can be accessed at https://github.com/TuftsVALT/ModelComm. The three major topics discussed during our interviews were:

1. **What would you need to know about a regression model before recommending its use?** Participants were described a hypothetical scenario in which a recently developed regression model was being presented at their workplace. Participants then told us what they would need to know about the model to recommend its use. Additionally, we asked participants their preference for presentation styles (single slide, in-depth, or interactive), as well as the types of visuals and information they typically see in similar data science presentations.
2. **How have you assessed and communicated a regression model at work previously?** We asked participants to reflect on a regression model that had been previously introduced to their daily work. Once the participant had a particular model in mind, they described how its performance was assessed, communicated, and scrutinized during its development and deployment.
3. **How could communicating a regression model's performance with *data scientists* and *subject matter experts* be improved?** We asked participants

to describe how their future collaborators could better communicate and give feedback on the performance of regression models. Specifically, we asked SMEs what they would like data scientists to communicate to them regarding a model's performance, and data scientists were asked what they would like SMEs to communicate to them.

The topics were designed to engage participants in a broad discussion on factors that influence their trust and interpretation of a regression model's performance and outputs. By discussing past experiences as well as hypothetical scenarios, we could gather responses from both participant groups to identify opportunities for improved communication. Ultimately, we use these findings (detailed in Section 4.4) to inform our guidelines in Section 4.5.

4.3.3 Analysis Methodology

To identify communication challenges that threaten model collaborations between data scientists and SMEs, we conducted a thematic analysis of our interviews. We followed a top down approach, described by Braun and Clarke [BC06], in which we iteratively coded and categorized statements made by participants to discover prevalent themes.

Developing codes for our codebook was guided by literature in qualitative analysis practices [MMKM98, DGMM11] and visualization studies [AZL+18, KPHH12]. At the first stage of our codebook, we began with a set of theory-based codes using prior work that characterizes ML workflows (see Section 4.2.3). In particular, we started with codes we derived from work by Suresh et al. [SGNS21], which includes objectives and tasks distilled from 58 papers across computer science and social sciences on characterizing the stakeholders of machine learning. The first complete draft of our codebook was developed over a 2-hour working session with all authors. In addition to our theory-based codes, we added codes we considered missing based on our initial observations of the interview data. The remainder of our coding process largely followed the procedure described by Alspaugh et al. [AZL+18].

We defined a participant's utterance as an individual response to each of our interview questions, such that short responses and multi-turns taken in conversation (about the same topic) were weighted fairly. At times, participants changed topics (*e.g.*, recounting a previous collaboration or memory based on an interview question), which we counted as a new utterance. Each utterance was coded as a whole and could be assigned one or more codes. The final iteration of our codebook totaled 44 codes. Our full codebook – including a definition of each code, inclusion and exclusion criteria, and an example via participant quotes (following codebook protocols [DGMM11]) – is included as supplemental.

To identify emergent themes, we ranked codes using several metrics including frequency, entropy, as well as the differences and similarities of the distribution of those metrics by group. The goal of this process was to perform an unbiased analysis to discern critical communication challenges, needs, and opportunities most important to both data scientists and SMEs. The three highest-level themes from this process, as deliberated by authors, were:

- Data science presentation styles and visualizations used when illustrating model performance
- Understanding a model's strengths and weaknesses
- Gaps and mismatches in language, context, and comfort experienced by SMEs during data science collaborations

The organization of our codes into high-level themes was conducted during four working sessions with all authors. A full report of our coding procedure (including coding passes and tie-breakers) is included in our supplemental material.

4.4 Interview Findings

In this section, we present the collective findings from our first round of interviews with data scientists and SMEs. We center the discussion around particular sources of friction in communication between the two groups, and identify opportunities to help

SMEs better understand and assess the performance of a predictive model. From our thematic analysis, we identified and grouped challenges related to: (1) presentation and visualization styles, (2) understanding the strengths and weaknesses of a model, and (3) gaps in language, context, and comfort. For each category, we highlight the most common issues that cause model collaborations to suffer. In Section 4.5, we map these challenges (C1-C7) to our proposed model communication guidelines.

4.4.1 Presentation and Visualization Styles

The first theme relates to the communication and illustration of model performance through presentation and visualization styles. SMEs revealed to us that data science presentations, metrics, and visualizations are not as easily interpretable or recognizable as data scientists think: "*Data scientists show a regression curve and it's so normal for them... they don't always realize that people don't understand some of the visuals for these models and what they really mean.*"

C1 **Metrics and visuals provide little insight to SMEs without proper explanation, context, and rationale from data scientists.** Overall, data scientists more frequently brought up using metrics to illustrate and interpret the performance of a model than SMEs. Our usage of the 'metric performance' code had the greatest difference of median tags across any code, with the median data scientist mentioning 'metric performance' 11 times, while the median SME mentioned 'metric performance' only 3.5 times. This finding supports previous literature that suggests data scientists, in practice, too often rely on metrics to communicate model performance [SMB19, SSC20, CS13].

SMEs reported uncertainty towards how error metrics relate to their end-goals for the model, and how metric performance should be interpreted: "*I've seen metric scores when working with data scientists. I've also observed others and myself wanting to know what was behind the score, how it was derived. I would say there's a great deal of skepticism. I think it's something that needs to be described or defined fairly quickly upfront... where does it come from? What's*

the top score, what's the bottom score?" (PID11). When asked how data scientists could make metrics more interpretable, SMEs cited "visuals" as one possible method, so long as they are contextualized: "*Data scientists need to explain what the metrics mean. Visuals can be a good way to explain it. It's more that you can see the results on the graph and you're able to relate the results to the context that we're working in*" (PID12). However, some SMEs found visualizations used by data scientists to be confusing, typically due to lackluster explanations for what was being depicted in the chart.

C2 **Interactive presentations are valued by SMEs, but may displace focus from a data scientist's intended takeaways.** Although data scientists cited metric performance as their primary method for communicating a model's results, a few told us they had prepared demos for presentations in previous projects. These demos consisted of interactive components of the model that could be manipulated by SMEs (e.g., interactive cells in a computational notebook), or visualizations that depicted model performance. In general, SMEs highly valued these interactive and in-depth presentations on a model, 5 out of 6 SMEs told us they would prefer an interactive presentation over a static or short presentation. In some cases, SMEs did not have prior experience with interactive demos or presentations, however, every SME wanted the ability to "play with the model," *e.g.*, by seeing how the model's outputs are affected by particular inputs: "*During this presentation... as I was observing what the data scientists were doing, I was thinking I'd really like to put in my own values to the model, rather than having them pre-defined by somebody. I would like to pop in my own things and see what it returns*" (PID11).

However, when asked whether model performance presentations should be interactive, one data scientist (PID1) told us that SMEs can focus on the wrong aspects of a demo or visualization: "*Sometimes I notice when we show a demo [of the model], which contains some graphs or some nice visualizations, SMEs can be drawn more to the visual aspects of the demo and not how the*

data was collected, or how the model actually performs... They're more drawn towards the visual part, so some key assumptions are not asked... Instead, there's this interest in like, can we color it differently? Can we change the labels?" This misplaced focus on the visual aspects of a presentation, rather than its depicted content, may be one cause for a lack of visualizations found in data science presentations[4].

4.4.2 Understanding Model Strengths and Weaknesses

The second theme focuses on major concerns raised by both data scientists and SMEs in understanding the caveats, edge cases, outliers, and limitations of a model. Many data scientists and SMEs told us that when a model's weaknesses (or limitations) are not fully communicated, it can delay or prevent model adoption – ultimately disappointing the end-users of the model: *"We need to bridge the gap between business expectations, what a machine learning model should be doing, and what realistically a data scientist can or cannot do."*

C3 **SMEs need comprehensive details for a model's weaknesses at all stages of development; data scientists need context for why a model is underperforming.** SMEs unanimously agreed it is essential to know how and when a model may not meet their needs, particularly at the beginning of a collaboration. SMEs generally referred to this information as the "limitations" or "weaknesses" for a model: "*Even at the beginning stages, it's still beneficial for our team to know what the model is all about. What are the strengths of this model, but also what are its weaknesses?... We may re-align our business needs around those strengths and weaknesses. But if there's a business need which the model cannot fulfill, we need to know during a presentation. We can help figure out how to change the approach, or if we should look into other models*" (PID8). Data scientists often remarked that their understanding of a model's weaknesses and limitations hinged on SMEs' ability to communicate

[4] https://hbr.org/2019/01/data-science-and-the-art-of-persuasion

nuanced, domain-specific context (*e.g.*, outliers).

Understanding the weaknesses of a model can also build confidence in SMEs when making decisions from the model's predictions. This is especially critical when the model is being used in a high-risk environment. PID9, who discussed the usability of a regression model that predicts water quality, was primarily concerned with when the model might fail: "*If we're using this model to tell people that water is potable, that means they're going to use it for washing, cooking, cleaning, drinking... and if there's something wrong with it, there are consequences to that decision. You know, there's a human being at the end of it. That's why I'd want as much information about the model as possible. So if I'm going to make a decision about applying this in the real world, then at least I know exactly what its limitations are before making a decision.*"

C4 **Misunderstanding or misrepresenting the data used in training can result in downstream issues.** While both SMEs and data scientists expressed great interest in the technical details for the data (*e.g.*, distributions, summary statistics), only data scientists commonly noted that it was important to know how the data was pre-processed. These pre-processing steps included transformations, splits, and imputation strategies. However, it was unclear whether these procedures are commonly communicated to SMEs – despite the SMEs' domain knowledge for whether those steps seem reasonable to perform. SMEs were also concerned about the quality of the data (*e.g.*, whether data was missing or there were outliers in the data). PID8 stated that data quality issues may at first glance look like legitimate outliers from a model: "*A high number of outliers show up incorrectly... and when you start investigating you identify that it's because of data related or data quality issues.*" In general, both data scientists and SMEs remarked that information about outliers, error, and representations of the data are critical to know when understanding a model's performance – and without this understanding, downstream issues can occur that prevent or halt model adoption (*e.g.*, the

effects of data cascades [SKH+21]).

C5 **Mismatched criteria for model acceptance can lead to unmet expectations for a model's performance.** A common pain point for model collaboration, most often brought up by data scientists, was the lack of clarity in what made a model "good" or "acceptable" to SMEs. On one hand, data scientists felt they had satisfied most of (if not all) the criteria that they established with SMEs at the start of their collaboration: *"SMEs don't necessarily understand the kind of undertaking you performed to improve a model. Perhaps for you it was an awesome result of a multi-month effort and you never thought you could do it. And then they say, isn't it still too much [error]?"* (PID4).

This concept of unrealistic model performance expectations is a common problem experienced by data scientists in machine learning collaborations [PJ18]. However, SMEs expressed their own frustration and confusion when models failed to perform up to their standards when all other criteria seemed to be met: *"We constantly need to revisit these kinds of discussions to understand the model better... There is a perception to us that, OK, we've given data scientists everything they need. The variables, the data points. So why is the model not giving us the expected results?"* (PID8).

4.4.3 Gaps in Language, Context, and Comfort

Finally, we discuss mismatches in language as well as a lack of context and comfort experienced by SMEs in data science collaborations. A common sentiment shared by SMEs was feeling confused, overwhelmed, or anxious when discussing details for a model and its performance – largely due to data scientists' inability to relate to SMEs: "*Something important for data scientists to understand is that they are potentially introducing a new concept to us or a new way of doing things, and sometimes they don't consider that as a reference point... I've heard data scientists comment that some of us don't know what we're doing or that we don't understand data. So there can be a little bit of an aloofness to the role of the data scientist.*"

C6 **Overly technical concepts that are not properly communicated by data scientists may discourage SMEs from asking clarifying questions.** SMEs expressed that they had felt confused or overwhelmed by data science presentations during previous collaborations, largely due to unfamiliar language, metrics, data, and visualizations used by data scientists. One SME noted that, as a result, she did not always feel confident asking questions during data science presentations: "*Sometimes these presentations just go over your head, and I think the end-user a good chunk of the time would be too embarrassed to say, I don't get what you're talking about*" (PID11). In Section 4.7.3, we examine language barriers that may be causing confusion between the two groups.

When we asked data scientists how they approach mismatches in the language spoken, many of them expressed comfort with explaining or defining concepts during presentations. One data scientist told us she includes slides in the 'appendix' portion of presentations with definitions prepared: "*Definitely there are a few people in the audience who won't understand these mathematical terms, so we create slides that explain them in the appendix. If someone has a question on terms, we just take it to those slides*" (PID5). Although back-up slides may help with providing definitions for terminology, it is unclear whether SMEs will always ask the necessary clarification questions to prompt those slides.

In contrast, SMEs told us they prepare pre-read documents to help contextualize unfamiliar concepts to data scientists. These documents included definitions and examples for the SME's domain, their objectives, known data outliers, acronyms/terminology commonly used, and expectations for the model: "*What has worked well for me in the past is sending a document, whether it's an email or Word doc or PowerPoint, kind of outlining what we're seeing, what our expectations are, what our concerns are. Because we all live in our own worlds... Data scientists are good at what they do, but maybe we [SMEs] aren't*

the best at conveying the full idea to someone who's not living and breathing in our space" (PID10).

C7 **A lack of illustrative examples disconnects the model from its proposed use case.** When discussing how data scientists can better connect a model to its proposed use case, SMEs most often requested practical, illustrative, and grounded examples of using the model to meet their objectives. On one hand, SMEs seemed skeptical that a new model being proposed could in fact be better than their current practices, particularly when data scientists do not take the time to compare both approaches: "*If data scientists explained what our [current] method does and how it works, and why and how they've built their model... if they told us the reason why their model could be better than ours, or how it could be more trustworthy, it would really help us understand that, OK, this new approach might actually be better*" (PID8).

Further, without a descriptive or exemplifying translation for how a predictive model relates to the business's needs, SMEs can fail to see its practicality: "*Data science, predictive models,... they're all kind of trendy at the moment. I would be conscientious about using a new model, making sure that the model that we would use aligns with the purpose and objective of what we're trying to accomplish*" (PID9). When a model is being proposed, one SME suggested that stories can be crafted by data scientists to better illustrate its eventual use: "*It can be any story. It can be a weird story. It doesn't need to be real, it doesn't need to have happened yet. Anything that would help us imagine how your model could help us predict a real outcome based on the information in your story*" (PID13).

4.5 Guidelines for Communicating Model Performance

Based on the analysis of our interviews, we derive a set of guidelines that can begin to bridge common communication gaps when data scientists present a model's

performance to subject matter experts. For the sake of clarity, they are grouped broadly by the challenges they address (C1-C7). We note that some guidelines may partially address many findings, and provide justification when that is the case.

The first three guidelines relate to the choice of **presentation and visualization styles** when communicating and presenting model performance to SMEs, with a focus on contextualizing visualizations and metric performance: "*Some people don't have experience with visualization outside of BBC infographics*[5]*. I do realize it can be hard for me to remove my data scientist hat and put myself into the role of somebody who's not looking at a log plot every day.*"

G1: **Provide context for a model's performance by annotating plots with clear stories.** Each annotation serves to decode the intended message of the visualization, beyond the visualized data and legend. Providing a clearcut story in the presentation of a model gives the data scientist more control over the message being communicated, and makes that message more palatable to the SME (**C2, C6**). Annotating visualizations that showcase individual data instances also provides the opportunity to highlight illustrative, concrete examples (**C7**). By guiding the audience through sensible conclusions on a provided visualization, an SME could more quickly arrive at new conclusions with the same visualization, improving visualization literacy [BBG19] over time. That said, building stories from model performance is a nontrivial task and an active area of research; we point to existing work for more in-depth guidance on audience-based AI/ML narratives [WYAL19, SH10].

G2: **For any chart that communicates a model's performance, provide a range of comparisons.** Providing a comparison helps focus the message of a presentation (**C2**), in particular on how the model might improve their current practices (**C3, C7**). It also helps define what qualifies as "good" and "bad" performance (**C5**). In our study, SMEs found that assessing the results of a predictive model's performance is easier if it is compared against their current

[5] https://www.bbc.com/future/tags/infographic

practices, in addition to an interpretable naive baseline model. If possible, an oracle, perfect model (or otherwise ground truth) can also be used for comparison.

G3: **Visually explain the significance of aggregated metrics.** Global metrics such as explained variance or mean absolute error can seem abstract and removed from the use case (**C1**). By sharing a visual medium, data scientists and SMEs can use it as a common language to discuss the significance of the metrics of a model. Showing metrics in visual context can help ground them; for example, visualizing the enveloping ellipse in a correlation scatterplot can give a proxy for the correlation between predicted and actual values. Whenever possible, showing how individual instances relate to the aggregated metric can help connect an SME's understanding of *items* of data with the metric calculated on the full *set* of data, which partially addresses the need for illustrative examples (**C7**).

The second three guidelines address concerns by both data scientists and SMEs in understanding the **caveats, edge cases, outliers, and limitations** of the model: "*If data scientists said, 'when you run these models, here is the area where we think you're going to have the most problems, or the most risk. And here's the explanation for why we think that's happening.'... I think upfront and transparent communication about why we should expect those issues is a very big way for us to build trust and confidence in the model.*"

G4: **Point to outliers in the model's performance and data space with known or plausible explanations.** SMEs have deep knowledge of the data being used to train the model and make predictions, so it is a missed opportunity if outliers are not discussed between SMEs and data scientists (**C3, C4**). The source of outliers and anomalies is often dependent on the scenario, therefore, data scientists should point SMEs to known or potential outliers, and include at least reasonable speculations behind their anomalous behavior. Use visualizations that illustrate the biggest errors in model performance, and show the distribution of features to point to potential outliers.

G5: **Explicitly describe the data used in training and testing a model,**

any pre-processing or transformations performed, and provide examples. The distribution, weighting, correlation, and availability of the data used in the modeling process were notable concerns from both SMEs and data scientists (**C4**). Many data scientists agreed that SMEs provide essential context for the data domain, ultimately leading to improvements in model performance and transparent communication. In addition to explicit communication about the data used in modeling, we point to HCI solutions that can prevent the downstream effects of data related issues [SKH+21].

G6: **Communicate the limitations, weaknesses, and blind spots of the model, especially when suggesting it can replace the existing solution(s).** SMEs want transparent information regarding the limitations of a model (**C3**) with qualitative and quantitative assessments of those errors or weaknesses against their current practices (**C7**). Both SMEs and data scientists noted that they were able to help each other improve a model once the limitation or poor performance of the model was fully understood – *e.g.*, a model overfitting, or a linear model performing poorly on outliers – ultimately leading to higher model acceptance. By understanding the reasoning behind weak performance (data quality, outliers, weak signal), SMEs can better understand the criteria for assessing a model's performance (**C5**).

The last three guidelines address a need for **context and comfort** identified by SMEs: "*You have to make the end-user feel comfortable both in the data scientist's language, and also that if they don't understand something they can ask, what is this?*"

G7: **When articulating results, start slow and offer to speed up; use visualization as a medium when appropriate.** Lack of comfort or understanding should be anticipated in a presentation of data science results, as SMEs have distinct backgrounds (**C6**). Data scientists need to be cognizant of this imbalance, particularly at the beginning of their presentations. SMEs suggested that data scientists could spend more time highlighting aspects of their presentation that

could be considered "obvious," in order to establish a common baseline for the language spoken and understood. Using interpretable visualizations (**G1**) to drive the discussion can make it easier to catch common misunderstandings and ground the discussion.

G8: **Provide a pre-read document or appendix with explanations and illustrations of any models, metrics, or unfamiliar vocabulary.** Since many data science concepts may be new to the audience (**C6**), it is beneficial to provide an option to read through both the definitions and the results, ideally offline before and after a presentation at the SME's own pace. Frisch et al. present details for successful pre-reads [FG20], and Crisan et al. describe how visualization can aid in data science documentation [CFGT20]. If a pre-read is not possible, an appendix slide with definitions should be included and deliberately referenced in the presentation.

G9: **Tie in use cases for the model by illustrating real-life, objective-driven examples.** SMEs need to understand how a model's performance relates to their end-goals before recommending its use (**C7**). Illustrative use cases can include individualized examples that showcase how the prediction of the model is affected by particular inputs (**C2**), as well as examples of the model's predictions being used in real-world applications. Similar to story-telling visualizations as in **G1**, illustrative examples can provide an entry point to engage the audience and provide comfort (**C6**).

4.6 Demonstration of the Guidelines

We demonstrate the use of our guidelines in a data science presentation with SMEs. For this demonstration, we developed two slide-based presentations on distinct regression modeling scenarios: the first scenario *without* our guidelines, and the second scenario *with* our guidelines. We conducted a second round of interviews with the SMEs from our first interview study (Section 4.3), treating the first presentation

as a point of comparison to elicit feedback on our guidelines.

4.6.1 Follow-up Interviews

From our first interview study, a common modeling scenario experienced by SMEs involved meeting with a data scientist to assess whether a newly developed model was ready for deployment or needed to be more performant. Error metrics were typically used by data scientists to depict the model's performance, but these metrics alone failed to capture key aspects of the model, e.g., its risk-benefit ratio. Consequently, SMEs struggled to determine how well the model would perform in real-world situations, or how the model's outputs could lead to meaningful insights.

Therefore, the objectives of our follow-up study were to: (1) understand whether our proposed guidelines could help inform and contextualize the evaluation of a model during a data science presentation, and (2) demonstrate how data scientists can incorporate our guidelines into their own presentations to better communicate the accuracy, strengths, and weaknesses of a model's performance to SMEs.

The slides we presented to SMEs in this follow-up study are available as supplemental material, including all visualizations and data tables seen by participants, and can be accessed at https://github.com/TuftsVALT/ModelComm. Below, we describe the protocol for our demonstration of the guidelines.

Participants: To understand whether our guidelines address the challenges identified from our first interview study (Section 4.3), the six SMEs who participated were contacted for a follow-up study approximately six months later, with four agreeing to participate. Interviews were held virtually in a one-on-one one-hour session. Each session walked through a set of slides describing two distinct modeling scenarios.

Scenarios: Scenarios 1 and 2 described different datasets with similar modeling goals. In both scenarios, participants listened to a presentation of a new model being proposed to replace a model currently in use. For the first scenario, designed as a control setting, a baseline set of information was given to communicate model performance based on common techniques described by the data scientists in our

first interview study, as well as our review of the related literature. In the second scenario, the experimental setting, the presentation of model results was augmented according to our guidelines, including visualizations. Because this follow-up study consisted of only four participants, we did not switch the order of the scenarios presented during the interviews. To balance the ordering, we used a more complex dataset and prediction task in scenario 2.

Datasets: For scenario 1, we used the Auto MPG dataset [DR82] (6 features) to train a regression model to predict a vehicle's miles per gallon (MPG). For scenario 2, we used the California housing dataset [KB97] (9 features) to train a regression model to predict a house's cost. All features in the original datasets were used in training our models.

Models used: SMEs were told that a KNN [FH89] ("baseline") model was already in use, and were asked to assess whether an XGB [CG16] ("new") model should be used instead. In both scenarios, the XGB model had improved performance.

Prediction tasks: Both tasks asked SMEs to choose between the KNN or XGB model. In scenario 1, SMEs were told they would use their chosen model to predict the MPG of a vehicle based on its specifications. In scenario 2, SMEs were told they would use their chosen model to predict the value of a property for a future investment.

4.6.2 Presentations

Scenario 1 (no guidelines): At the start of both scenario presentations, we included a short description of the dataset, the model inputs, and the SME's prediction task. An example of the input data was provided as well as a table of the first 10 prediction results for the KNN (baseline) and XGB (new) models. Following the most standard practices described by data scientists and SMEs in our first interview study, we did not include any visualizations for the first scenario's presentation. Instead, a table of common regression model metrics (R^2, MSE, MAE) were displayed to illustrate the performance of both models.

Scenario 2 (with guidelines): Before starting our second scenario, we explained to SMEs that supplementary information would be included in the next set of slides. This additional information was designed to reflect the guidelines described in Section 4.5; the description of the mapping is shown in Table 4.2. In total, there were four differences (**D1**-**D4**) incorporated into the presentation of our second scenario. This additional information was embedded and explicitly highlighted in our slides, and can be found in our provided supplemental. We briefly describe each below:

D1 Open to questions: An initial slide encouraging SMEs to stop us at any time to ask clarifying questions. (**G7**)

D2 Detailing the data: A table that descriptively defined all features used in training and testing the models. (**G5**)

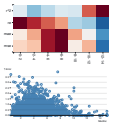
D3 Defining all terminology: High-level descriptions of the modeling algorithms (KNN and XGB) and how they differ in deriving their predictions. Definitions for R^2, MSE, and MAE were supplied. These definitions included how a lower or higher value for each metric translate to relative performance. (**G8**)

D4 Highlighting model weaknesses: Several examples of XGB performing worse than KNN, on both instance-levels and visualized regions of data. Each example included a short description to explain why we believed XGB was predicting worse than KNN for these specific examples, and how this performance would relate to real-world objectives (**G6**, **G9**).

Lastly, we used four different visualizations (**V1**-**V4**) to illustrate differences in performance for the two models. Our chosen visualizations were based on the guidelines and deliberated between all authors; however, we note that a broad design space of visual encodings would pertain to our guidelines. The four chosen for this demonstration were picked to illuminate several key aspects of the guidelines: showing outliers, enabling comparisons, and providing both global and instance-level views into metrics. We considered charts that are recognizable and interpretable to SMEs (*e.g.*, heatmaps, scatterplots, histograms) that can also address these model communication needs.

Every visualization for the XGB (new) model included a comparison visualization to the KNN (baseline) model (**G2**). For each visualization, one or more annotations were included (**G1**) to point to specific outliers (**G4**), differences (**G8**), and conclusions (**G9**) between the two models. We read the annotations to SMEs during the presentation and asked if they had any follow-up questions. Full versions of the visualizations can be seen in Table 4.2, all source code is provided in an Observable notebook[6], and the annotations are included in our supplemental. The visualizations used in our follow-up are described as follows:

(V1) Heatmap by error category and feature distribution:



A heatmap that illustrates aggregate errors (y-axis) across a single feature (x-axis), using a distinct color scale for each row (*i.e.* error metric). The heatmap can help SMEs see how different aggregate measures (**G3**) differ across groups (**G4**) and determine if a model's performance has weaknesses along the selected feature (**G6**). This is especially important when looking at how a model performs across regions of data, ensuring that a model with a slight increase in overall accuracy does not sacrifice performance within a specific subgroup of data. Attached to the heatmap is a one-dimensional scatterplot showing the distribution of the feature used to create the grouping, to allow for viewing individual instances when interpreting aggregate measures.

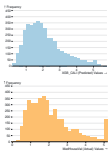
(V2) Correlation scatterplot with confidence ellipse:

A scatterplot with predicted values from the model plotted on the x-axis, and actual values plotted on the y-axis. Behind the plotted points is a 95% confidence interval (or covariance ellipse [Sch18]) that serves as a graphical proxy [FMF13] for the regression metric R^2 (**G3**). Simultaneously displaying the aggregate metric R^2 along with individual instances in the scatterplot provides a visual interpretation for the R^2 metric. In addition, it makes it easy to

[6] https://observablehq.com/@ashleysuh/model-comm-vis

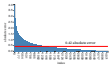
see outliers in predictions and labels (**G4**).

(V3) Histogram for predicted values versus actual values:



Two one-dimensional histograms, each showing the distributions of a model's prediction and actual values, respectively (**G5**). By highlighting the differences in shape of predicted versus actuals, this visualization makes it easy to see if a model might have biases (*e.g.*, missing the high-range outliers found in the ground truth, or missing one of the modes of the distribution of actual values) (**G4**).

(V4) Bar chart with global error:



A bar chart where each bar represents the absolute error of a single instance; bars are sorted by descending error. This chart shows the *shape* of error by the models, and conveys how it apportions error, whether by apportioning it equally among all instances, or confining it to few (**G6**). A red horizontal line is added to represent the mean absolute error, giving more context to the per instance error for the MAE and MSE metrics. (**G3**)

V1 and **V3** were used as part of the presentation of examples where the model performed poorly (**D4**). Any time XGB performed worse than KNN in our examples, we included a short description to explain why we believed XGB was predicting worse than KNN (**G4**).

Feedback questions: At the end of our demonstration, we asked SMEs to evaluate the effectiveness for each of the presentation differences (**D1-D4**) and visualizations used (**V1-V5**). SMEs rated their agreement to the following statement:

> "*I found this difference / visualization to be helpful in interpreting the model's performance.*"

Agreement was made on a 7-point Likert scale of 1 (Strongly Disagree) to 7 (Strongly Agree). Finally, SMEs provided open-ended feedback on the helpfulness of the guidelines.

4.6.3 SME Feedback

4.6.3.1 Additional model information (D1-D4)

In general, all supplemental information for the models (**D1**-**D4**) were considered helpful by SMEs, which is not surprising on its own: contextual information is generally helpful during model communication, as evidenced by challenges C6 and C7 from Section 4.4, and previous work on improving model adoption by SMEs [SMB19].

Ranked by average helpfulness score, **D4** [*Highlighting model weaknesses*] (6.75) was most helpful, reflecting our finding that understanding the strengths and weaknesses of a model (**G6**), as well as seeing objective-driven examples of its predictions (**G9**), impacts and informs decision-making. It was followed by **D2** [*Detailing the data*] (6.5) and **D3** [*Defining all terminology*] (6.25), which provide additional context on the raw data, the models, and the metrics used in a presentation (**G5**). The lowest rated difference **D1** [*Open to questions*] (5.5) was split, with two SMEs rating it as not helpful or unhelpful, and two SMEs rating it as strongly helpful. Below, we distill takeaways from the use of **D1**-**D4**.

Highlighting the model's weaknesses informs decision-making and boosts credibility: SMEs strongly preferred examples of the model performing poorly. PID9 explained that they helped her "*come to the conclusion about whether there's a real performance difference between the two models*," reinforcing our previous finding that SMEs need transparency from data scientists on a model's weaknesses. Similar to challenge C3 identified in Section 4.4, PID8 told us that seeing the model's limitations would help him decide whether the model could be improved, or would be a no-go for future tasks: "*Explanations around why KNN shows such worse results than XGB... That helps us try to see if it's a valid reason, or if it's a straight no-go and we cannot consider the model.*"

Describing the data used in training "beyond numbers" level-sets the presentation: SMEs found that detailed descriptions and examples of the modeling data provided essential context beyond commonplace summary statistics: "*Always show this [data descriptors] before the numbers. Otherwise they're just numbers*"

(PID11). PID8 had similar feedback, remarking that "*these descriptions will help everyone [SMEs] be on the same page as the data scientists*," thus grounding the model and its performance to the real world.

Defining data science terminology, model differences, and error metrics establishes a common framework: Overall, SMEs found the added definitions and descriptions of the models used (KNN and XGB) to be helpful in interpreting how predictions are made. PID9 told us, "*without your descriptions, KNN versus XGB could be anything. General information is helpful.*" Both PID9 and PID11 suggested that walking through examples with "*named attributes*" in a "*nontechnical context*" could be even more helpful (**G8**).

Although some SMEs were already familiar with the metrics used in our presentation, all SMEs found the added description for what 'low' and 'high' error means to be helpful in contextualizing model performance: "*I already know about these [R^2, MSE, etc.], what's good and bad. But of course knowing what high versus low error means is helpful*" (PID11). This finding reinforces challenge C1 and guidelines **G2** and **G3**: "*I don't use R^2 or MSE or any of these on a day to day basis. In our world, we use p-values... It's difficult at times to know in which scenarios lower is better or higher is better*" (PID9).

Encouraging SMEs to ask for clarification helps the presentation feel conversational: The overall sentiment towards our slide that encouraged clarification questions was split. PID11 told us, "*I think I'd already be proactive if there was something I didn't understand.*" On the other hand, PID9 felt the addition made our presentation more conversational: "*The perception it creates when you have this statement is that it's more of an exchange. Without it, it feels like more of an exam.*" PID8 related it to having a two-way, as opposed to one-way, presentation: "*It does give you the liberty that nothing is written in stone, it's not something that's one way. It's kind of a two-way as you go through [the presentation]... You're not just saying, this is what we are doing.*" We believe the additional step by data scientists to remind SMEs that they can always pause the presentation to ask questions will

help encourage those who may not otherwise (**G7**).

4.6.3.2 Model performance visualizations (V1-V4)

Visualizations **V1** [*Heatmap*] (7.0), **V2** [*Scatterplot*] (6.5), and **V3** [*Histogram*] (6.0) were all considered helpful, while **V4** [*Bar chart*] (2.75) was not considered helpful. Overall, we find that **visual comparisons that clearly tell a story are preferred to simple text and metrics.** All SMEs pointed out that with the supplied visualizations, they had "*quick*" or "*instant*" insights into model performance, whereas text or numbers had to be studied. Below, we discuss the qualitative feedback (and overall Likert ratings) received from SMEs on the visualizations shown in our presentation.

The heatmap **V1** was found to be "strongly helpful" in interpreting the models' performance for all participants, partly because of its easy interpretation: "*Easily interpretable since the colors pop out, colors are easier to draw conclusions from and find outliers*" (PID11); "*I can immediately see which model is better and which is worse*" (PID8), and partly because it helped communicate the strengths and weaknesses of the models: "*This one gives you even more detail as to how the models are behaving, where they're having the most errors*" (PID9).

The scatterplot **V2** was found to be useful for understanding the metric R^2, as PID13 noted that the ellipse helped them see the difference in precision for the two models: "*With the ellipse in the middle, it was easy to gauge where each model was better*," but it was also helpful for identifying outliers and alerting risks by PID8 and PID9: "*I can see visually where the difference was, the outliers – the values falling outside of the ellipse on the left side.*"

The histogram **V3** was considered the easiest visualization to grasp, and helped participants see what types of errors the two models had: "*What's really positive is it's very easy to understand*" (PID13); "*It does give a quick reality check in terms of the actuals versus the predicted*" (PID8).

Ratings for the bar chart **V4** ranged from "neutral" to "strongly unhelpful,"

with all participants remarking that the chart did not seem visually distinguishable between the two models, so it was not clear what the chart was trying to communicate: "*The shape of the curves are similar, it's just not jumping out at me what conclusion I'm supposed to take from it*" (PID11). However, it was noted by PID9 that, for a different scenario, the chart could be helpful if the distribution of errors was very different: "*Overall shape was the same and the absolute errors were close. Probably wasn't as useful in the exercise. But there are probably circumstances where it might be helpful as a more differentiating figure.*"

We believe that this illuminates a practical consideration for implementing the guidelines: depending on the dataset and the model predictions, visualizations produced by data scientists may not tell an obvious story. These ambiguous visualizations should not be included in a presentation, as they might be taxing to interpret and could even mislead the audience towards reaching for a false conclusion.

4.6.4 Learning Outcomes from Using the Guidelines

In every circumstance, SMEs found our additional information about the models and explanations for why a model performed poorly to be helpful when choosing between the two models (KNN and XGB). PID8 found the additional information helped him assess potential risks: "[In the second presentation] *I can weigh each of these different parameters against everything and try to take more calculated risk. But with the first presentation, I'm really not sure because I'm just dependent on what limited information is provided and if it clicks, it clicks. If it doesn't, then I'm going to lose everything.*"

As PID8 explained to us – when there is less clarity, it is better to be able to understand where the models perform well and perform poorly, so that risk can be calculated, rather than taking a leap of faith. PID9 similarly told us that "*a more nuanced presentation can give confidence.*" These findings demonstrate the value of guidelines **G4, G5, G6**: weaving outliers, strengths, and weaknesses into the model presentation improves the SME's ability to use their own expertise in their assessment of a model's performance.

All SMEs also provided positive feedback on the use of visualization to communicate model performance rather than simple text or metrics. Annotated examples in the visualizations (**G1**) were recounted by SMEs as particularly helpful in providing quick insights and conclusions to be drawn. PID11 and PID13 both noted that they wanted the ability to pore over the visualizations outside of our presentation to fully digest all of the information: "*I'm often someone who doesn't react to something initially on a screen... And when it's on the screen and I'm listening to someone present I'm back and forth between listening to the person talking and then looking at the slide... I'm not digesting the visuals*" (PID11). SMEs supported the usefulness of providing a pre-read document (**G8**) containing the visualizations shown in our presentation. We believe this also highlights the need for more interactive tools – we discuss further in Section 4.7.2.

Overall, the feedback received from the demonstration of our guidelines during follow-up interviews was positive. We found that **D1**-**D5** were all either helpful or neutral, and provide some evidence that guidelines **G5**, **G6**, **G7**, **G8**, and **G9** can be partially addressed by data scientists with these easy-to-implement additions. SMEs generally found the guideline-improved presentation to be helpful enough that the guidelines have been implemented within their organization. We discuss the institutional impact of our guidelines in Section 4.7.1, and limitations of this follow-up study in Section 4.7.4.

4.7 Discussion

4.7.1 Institutional impact: Guidelines in Practice

The guidelines presented in this chapter were developed in collaboration with a large pharmaceutical organization. The conducted interviews have highlighted not only that multi-disciplinary communication remains a substantial barrier to communication, but also that the effectiveness of collaboration is often overestimated. Additionally, we found that familiarity with advanced visualization techniques varies widely across users and groups, even if they are part of the same organization.

This finding has highlighted the benefits gained by providing additional context to presentations, and slowing the pace of discussions by not just providing time and opportunities for SMEs to ask questions, but by encouraging them to do so.

Our industry partner has already begun to adopt these guidelines in order to address and mitigate the challenges presented in this chapter. These teams are high-performing, multi-disciplinary groups that make regular use of ML models to address domain science questions. Although all collaborators have responded positively to these new guidelines, it is not possible to implement the entire suite of recommendations simultaneously. After in-depth discussions with SMEs, new visualizations will be introduced slowly to allow fluency in one visualization type at a time.

It should be noted that introducing interactivity must be done with care. An initial attempt at providing interactive visualization techniques to the SMEs required multiple steps: the SMEs needed to first adapt to new visualization methods, new interaction techniques, and new data. Only after providing an interactive system that better matched their fluency with visualization techniques was the system adopted and used by the SMEs. Overall, implementation of these guidelines has resulted in a noticeable change in group dynamics: SMEs are more engaged, asking more questions, and are more positive about the collaboration.

4.7.2 Presentation Modalities

There are many modalities for presenting and communicating model performance. In this chapter, we evaluated the use of our visualization guidelines in a PowerPoint presentation, however, SMEs told us in our interview study (Section 4.3) they would also find value in interactive presentations. Interestingly, when we first brought up an "interactive presentation," none of the SMEs we interviewed knew precisely what we meant. Once the interviewer explained the capabilities for an interactive system to communicate model performance, SMEs were immediately drawn to the idea that they could test various inputs to outputs themselves. One of the SMEs we interviewed mentioned that during previous model performance presentations, she

had found herself wondering "*what if we tested another input instead?*" However, she felt this line of communication could be disruptive, and did not think it would be possible to see this type of interaction on-demand.

Of course, there is an indisputable trade-off cost to building interactive tools and interfaces for model performance. There is continued research being done to support effective and accessible interaction for web-based tools [Sie20], computational notebooks [WHS20], reusable ML interfaces [BCH+22], and responsive visualization tools [SLGS22] that are practical for data scientists to use. Adding interaction also introduces costs for the audience (*e.g.*, false discoveries) during a presentation [Lam08]. Further, adding interaction could be overwhelming or ineffective for an audience if they are unfamiliar with common visualization techniques (see Section 4.7.1). To this end, we suggest introducing interaction only after SMEs show comfort and understanding of static visualizations, *e.g.*, those we demonstrated in Section 4.6.

Future work should consider the integration (and effects) of interaction into mainstream data science tools, such as pandas and sklearn. Computational notebooks seem to balance interactivity with the costs of building interactive visualizations, and are historically popular with data scientists [AZL+18]. A handful of data scientists we interviewed mentioned they design computational notebooks with SMEs in mind: inputs to the model can be modified on-demand, certain cells can be collapsed or left open depending on who is being presented to, and so on. As part of our contributions in this chapter, we designed and deployed an interactive Observable notebook that has a variety of static and interactive visualizations inspired by our interview findings.

4.7.3 Discriminative Language Analysis

In addition to our thematic analysis in Sections 4.3 and 4.4, we also performed a discriminative analysis between classes using Scattertext [Kes17] on our interviews. Scattertext is a tool that specializes in visualizing linguistic variation between document categories to investigate the differences between the language of the two groups. The Scattertext plot of the processed interview data can be found in Figure 4.1.

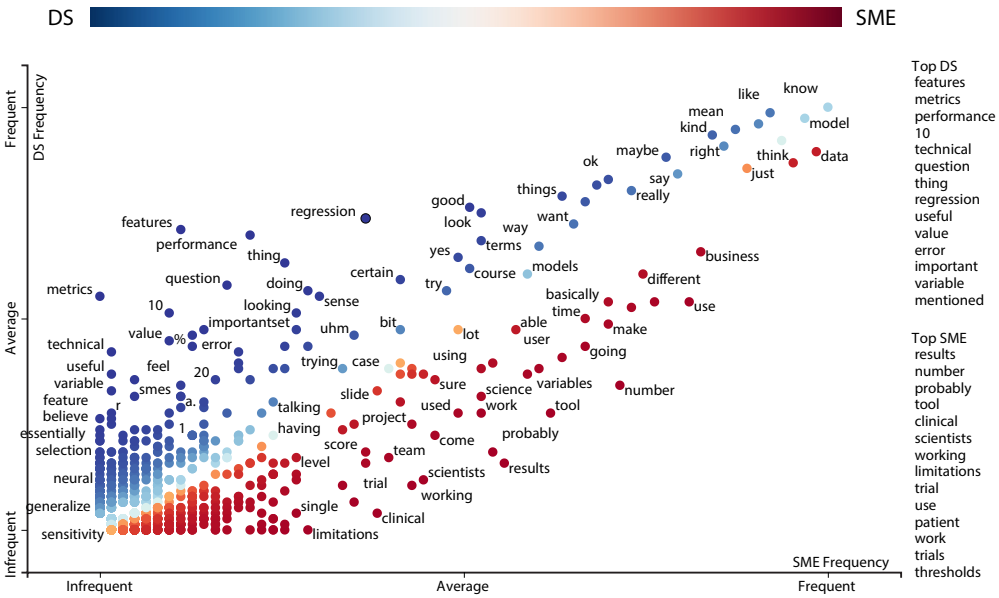


Figure 4.1: Results from using Scattertext [Kes17] on our interview documents. The x-axis represents the ranked word frequency spoken by SMEs (points shown in red), the y-axis represents ranked word frequency spoken by data scientists (points shown in blue), and words shown on the diagonal (colored closer to white) are related to both DS and SMEs. The "Top DS" and "Top SME" columns show the terms most related to each respective participant group. See Section 4.7.3 for more details.

The goal of this disciminative analysis was to determine if the two groups were emphasizing different aspects of model communication, or using different terminology to describe the same concepts. Ultimately, we can begin to understand the vocabulary data scientists should use for broad audiences in presentations, as well as what might be relevant to add to a pre-read document suggested by **G8**.

The Scattertext results in Figure 4.1 contains evidence that the two groups did emphasize different aspects of model communication. Words spoken by data scientists and SMEs with similar frequency tended to be inherent to the collaboration, such as "models," "data," and "explain." Words spoken more frequently by SMEs included terms that related to their own domain: "clinical," "trial," "patient," "user," and "business," illustrating that SMEs tend to ground the discussion of a model within their expertise. On the other hand, data scientists were more likely to use words like "features," "regression," "metrics," and "technical," demonstrating their propensity to discuss models in a more domain agnostic manner and utilize their own familiar terminology or assessment strategies. These findings supply further credence that data scientists could work to ground their explanations within the SMEs' domain expertise in order to increase context and comfort (**G9**).

Combining word frequencies with the interview text yields insight into repeatedly spoken words that are used differently by the two groups. SMEs tended to use "results" holistically to describe the performance, limitations, and weaknesses of the model, while data scientists used more specific terminology to describe model performance. SMEs used "value" in the traditional sense of the word (*e.g.*, "*the model's value in my workplace*"), while data scientists used "value" as a prediction or numerical value – this can lead to confusion across the two groups. Data scientists often used "error" to identify which error SMEs care about, whereas SMEs did not necessarily translate error to the overall goodness of the model. These slight differences in semantics should be kept in mind by data scientists when delivering presentations to SMEs, and can be included in a pre-read document or appendix slide when appropriate (**G8**).

4.7.4 Limitations & Future Work

The requirements gathered during our interview study, in addition to the guidelines demonstrated and tested in our follow-up study, represent only a subset of problems and solutions from professionals collaborating within the same corporation. While we recognize that this limits the generalizability of our findings, we believe that it was necessary to restrict the data scientists and SMEs to be in the same organization, in order to get both sides of the same story. The scope of our studies is also narrowed to both the high funding received towards AI/ML at this corporation, and the high familiarity of data science by our interview participants. Moreover, our interviewees are highly educated professionals (4 PhD, 8 MS, 1 BS) who are considered experts in their field.

Our interview study did not attempt to quantify whether the challenges faced by our data scientists and SMEs are common across a variety of domains, skill levels, and corporations. In prior literature, studies have been done to understand AI/ML industry practices across a multitude of domains [HHB20], in addition to sectors outside of 'Big Tech' [HB21]. In contrast, our study aims to characterize *how* model performance is assessed and communicated (*e.g.*, presentation styles, language used, metrics, visualizations). Our goal was to identify communication bottlenecks that hinder model usage, and suggest easy-to-implement guidelines that can alleviate these issues through visual mediums.

Our study is focused on predictive model performance communication, as opposed to general AI/ML interpretability. While it is possible that the methods suggested by this chapter could be mapped to communicating any predictive model, we did not ask interview participants any specific questions related to other types of models. Previous work has investigated challenges related to workflows on "blackbox" AI models between data scientists and stakeholders (*e.g.*, [PJ18]), however, we observe a lack of prescriptive guidelines for data scientists to consider when communicating their models. Future work should be done to assess the validity of (or extend) our guidelines to other AI/ML models.

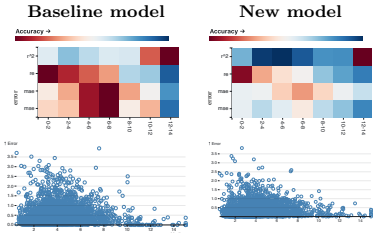
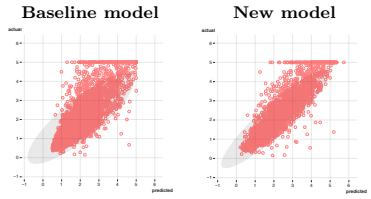
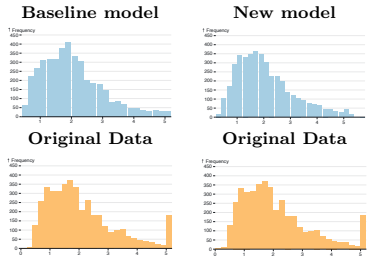
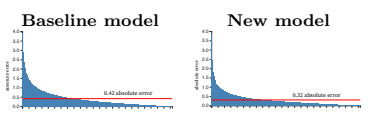
Finally, although a motivation for this study was to help increase the number of models deployed in practice, we could not validate whether our guidelines address this gap. Future work should be done to closely follow a model's development and how it is communicated - aided by our guidelines. Fieldwork by Passi and Jackson [PJ18] begins to tackle this critical problem. It should similarly be documented and evaluated in research whether our suggested guidelines impact a model's eventual adoption in practice.

4.8 Conclusion

In this Chapter, we discussed interviews with data scientists and subject matter experts, identified common gaps in communicating complex data, and suggested broadly applicable solutions for data scientists to use when presenting their results to SMEs. We demonstrated our guidelines by giving a data science presentation to SMEs that followed our guidelines and elicited feedback. Our results indicated that our guidelines help better understand the discussed data.

Learnings from these interviews provide further evidence of the usefulness of HyperNP and NNInv in helping users effectively use their high-dimensional data. SMEs expressed preferences for interactive tools over static presentations; HyperNP improves on existing tools by enabling users to interrogate their data across multiple projection hyperparameters in real-time, getting to act and see the effect on the resulting visualization without time-intensive re-runs. At the same time, NNInv can produce many of the explanatory functions SMEs need to trust and adopt new models. For example, NNInv can produce high-dimensional examples even in spaces without original data and marks errors and distortions within DR visualizations. With additional explanatory functions, users can understand the limitations and feel confident in assessing risks.

Table 4.2: An illustration of how our guidelines (Section 4.5) can be applied in practice to a data science presentation.

	Guidelines	How the guidelines can be implemented	SMEs' feedback on effectiveness
D1	(**G7**) Ask for clarification	Slide instructing participants to ask clarifying questions about data, charts, or any language / vocabulary.	Statement made the presentation feel like "*an exchange, instead of an exam.*"
D2	(**G5**) Data descriptions	Table detailing all data attributes used in training, including units, a short description, and an example for each.	Quickly familiarized SMEs with the data, "*otherwise they're just numbers.*"
D3	(**G8**) Terminology definitions	Comparative explanation for how different regression models make predictions. Definitions for common metrics (e.g., R^2, MSE) were included, with descriptions of high versus low error.	Gave models and metrics context: "*Without descriptions, those [models] could be anything.*"
D4	(**G6**) Show model weaknesses and (**G9**) illustrate how these instances impact real-life objectives.	Specific examples – tied to the prediction task and objective for the model – that illustrate one model performing worse than another, with explanations and context for why.	Objective-driven examples were cited as the most helpful for decision-making by SMEs.
V1	(**G3**) Visually explain multiple metrics and (**G4**, **G6**) see where they perform strongly or poorly. (**G2**) Compare performance to a baseline and (**G1**) see annotated examples.	Heatmap showing the apportionment of error across subgroups of the data, split by metric. Reveals potential bias in the data. 	Easy to interpret the strengths and weaknesses of the models: "*The visual [heatmap] made it easier to draw conclusions, the numbers alone were not as easy to translate what the model means, or how you could apply it if you were really using it [for the task].*"
V2	(**G3**)Visually explain R^2 metric and (**G4**) discover outliers in prediction error. (**G2**) Compare performance to a baseline and (**G1**) see annotated examples.	Correlation scatterplot between predicted & actuals to identify outliers. A containing ellipse provides a visual for the R^2 metric. 	Useful for illuminating R^2. The bounding ellipse helped discern the difference in precision for the two models, as well as identifying outliers and potential risks: "*I found it particularly easy with the ellipse in the middle to gauge where each model was better. I can see visually where the difference was, the outliers.*"
V3	(**G4**) Discover outliers in prediction error and (**G5**) view the difference in distributions. (**G2**) Compare performance to a baseline and (**G1**) see annotated examples.	Paired histograms comparing the distribution of predicted values from the model versus the actual values from the raw data. 	Simplest visualization to grasp, helping SMEs see what types of errors the two models had compared to ground truth: "*Easy to compare the two side by side, the overall shape gives you an idea of the errors that can come up. It's just these two bars at the end each time.*"
V4	(**G3**)Visually explain MAE metric and (**G6**) observe where error is apportioned. (**G2**) Compare performance to a baseline and (**G1**) see annotated examples.	Absolute error plots to contextualize the MAE metric (in red). Shows the density of data with better or worse average error. 	Models were not visually distinguishable, making it unclear what finding was being communicated in the chart: "*Probably wasn't as useful in this exercise. But there are probably circumstances where it might be helpful as a more differentiating figure.*"

Chapter 5

HyperNP

This chapter discusses HyperNP, the first algorithm in our framework for tackling complex high-dimensional data. It enables real-time interactive dimensionality reduction and out-sample projection across important hyperparameters, making it easy for practitioners to understand the effect of hyperparameters and explore their data fully.

5.1 Problem Statement

As data-generating devices and computational resources expand, there is a growing need for interactive visual exploration for high-dimensional data [LMW+17]. An important class of visualization methods for such data is projection, which maps data from a high-dimensional space to a similarity-preserving low-dimensional representation [NA19, EMK+21]. Many projection methods exist, including linear and global techniques such as PCA [Pea01], and non-linear approaches such as t-SNE [vdMH08], UMAP [MHSG18], and Isomap [BS02].

Projections can be sensitive to hyperparameter choices that must be tuned carefully [SH05, WHRS21, WVJ16]. Unfortunately, many projection methods are computationally expensive [NA19], making real-time hyperparameter changes intractable. In order to support a user's exploration of hyperparameters in projections, a method is required that enables real-time interaction with a projection's hyperpa-

rameter values.

In this chapter, we present HyperNP, a deep learning technique that approximates projections across hyperparameters to enable real-time exploration of a projection method's hyperparameters. Similar to previous work on neural network projection (NNP) [EHT20], HyperNP approximates the projection of a single technique (like t-SNE) on a single dataset with the added benefit of out-of-sample projection. Both NNP and HyperNP can be thought of as surrogate models for projection techniques, which attempt to alleviate some of the computational burden by approximating the results. Unlike previous methods, HyperNP can compute projections for hyperparameter values *unseen* during training, without the heavy recomputation inherent to the process for general projection methods. Previous methods [EHT20, EHFT20b] have to refit each hyperparameter value since they only produce a single projection at a single hyperparameter value. HyperNP, on the other hand, produces projections across the range of the hyperparameter value of interest after a single training. Once HyperNP has been trained, the compute time required for the inference step is minimal. We leverage this fast computation to allow real-time visualization of large data sets and interactive exploration of different hyperparameter values associated with the projection operator.

As with most deep learning techniques, HyperNP has two stages: training and inference. Training HyperNP to mimic the outcome of a parameterized projection method has two steps: (1) The hyperparameters are sampled and used to project a subset of the data using the user-selected projection technique. (2) The HyperNP model is trained with a loss function that captures the differences between its prediction and the ground-truth. After training, HyperNP infers the 2D projection of high-dimensional data for any user-selected projection hyperparameter value. At this stage, the user can interact with the hyperparameters using HyperNP, as the effect of these parameters can be estimated by the HyperNP model.

HyperNP is applicable to any projection requiring user-tuned input. This is noteworthy, as it means that the technique can be leveraged for interactive tuning of *any* projection method without reformulation. We demonstrate HyperNP's ability

to learn hyperparameters by applying it to three popular techniques: t-SNE, UMAP, and Isomap. For each of these techniques, we explore one of their most important hyperparameters, and show how HyperNP can help users tune this hyperparameter to find an appropriate projection. This shows that HyperNP learns hyperparameter spaces that cover both continuous domains (t-SNE perplexity) as well as discrete integral domains (k-nearest neighbors value for UMAP and Isomap).

To evaluate HyperNP, we conducted experiments across three datasets to assess its (1) accuracy in approximating projection techniques across a variety of measures, (2) scalability, measured by training time, and (3) interactivity, measured by inference speed. The results demonstrate that HyperNP creates high-quality projections using a fraction of the training data and hyperparameter samples. Projections generated by HyperNP are visually similar to those produced by these projection methods, and perform similarly to the ground-truth in terms of common projection-quality metrics. Further, HyperNP is scalable with respect to the size of data, both in terms of training time and inference time. Training HyperNP with up to 50k data points requires less than 20 minutes. Once trained, HyperNP is fully interactive with 20ms latency for 50k data points and 500ms latency for up to 500k data points. In summary, we claim the following contributions: HyperNP, a deep learning-based **method** able to approximate projections of high-dimensional data across hyperparameter configurations at interactive speeds; **examples** of our method applied to three popular projection techniques (t-SNE, UMAP, and Isomap); and a quantitative **evaluation** of our method highlighting its accuracy, scalability, and interactivity.

5.2 Method

We next detail HyperNP, a technique that can approximate any projection of a high-dimensional dataset at interactive speeds. This allows users to easily experiment with the hyperparameters of the approximated projection method. Similar to NNP [EHT20, EHFT20b], we use a neural network to learn the approximation of

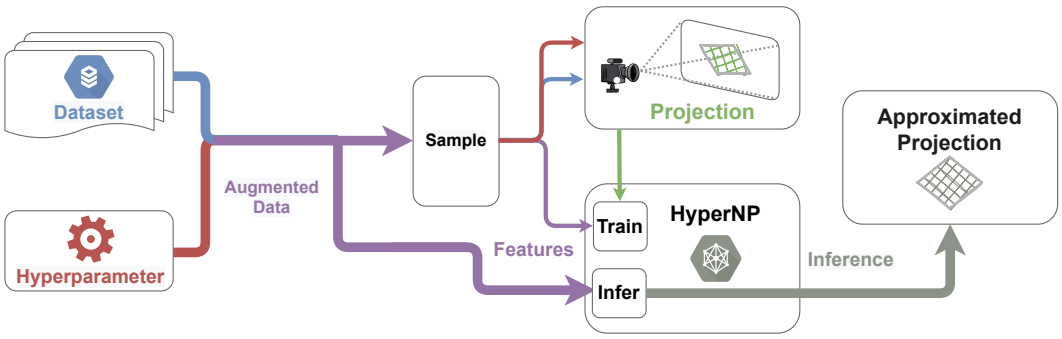


Figure 5.1: Architecture of HyperNP: Training data from a dataset is projected across a range of a hyperparameters of interest belonging to a specific projection method. HyperNP's neural network is trained using a combination of a sample of the dataset to project and the projection methods's hyperparameters as features, and the projections of the dataset sample as target values. After training, HyperNP infers projections that approximate the original projection, but at a fraction of the computational cost. In the figure, line width relates to the number of observations; color shows when data are combined, split, or transformed.

a projection on a dataset. As NNP showed, such networks are accurate in their approximations and can infer a projection in a fraction of the time required by the learned technique. The configuration of the dense network used in this work is different from NNP and can be found in Section 5.2.4.

Figure 5.1 illustrates the process of using HyperNP to approximate a projection. Note that the dataset and the chosen hyperparameters are necessary for the creation of the ground truth projection and for training HyperNP. In addition to computing the projection $\mathbf{y} \in \mathbb{R}^2$ of a sample $\mathbf{x} \in \mathbb{R}^n$, and different from NNP, our trained network takes an additional argument $\mathbf{h}$ that represents the hyperparameters of the projection algorithm. For example, for UMAP, $\mathbf{h}$ is a single value that represents the number of nearest neighbors; for t-SNE, $\mathbf{h}$ is a single value that models perplexity. Performing inference with HyperNP on a data point $\mathbf{x}$ can be written as

$$\mathbf{y} = \hat{P}([\mathbf{x}; \mathbf{h}]) \tag{5.1}$$

where $\hat{P}$ represents HyperNP's neural network, $\mathbf{y}$ is its projection of $\mathbf{x}$, and $[\mathbf{x}; \mathbf{h}]$ is the concatenated vector of the sample and hyperparameter values that serves as input to $\hat{P}$.

5.2.1 Model Training in Theory

To train $\hat{P}$ to approximate a projection method P, we seek to minimize the difference between the projections output by $\hat{P}$ and P. Denoting $\hat{P}$'s parameters by θ, the objective can be expressed as

$$\underset{\theta}{\operatorname{argmin}} \frac{\sum_{\mathbf{x} \in D} \sum_{\mathbf{h} \in H} \|\hat{P}_\theta([\mathbf{x}; \mathbf{h}]) - P(\mathbf{x}, \mathbf{h})\|^2}{|D| \cdot |H|}, \tag{5.2}$$

where D is the input dataset and H is the set of all values of $\mathbf{h}$.

We show empirically (Section 5.5) that training $\hat{P}$ does not require minimizing Eqn. 5.2 for all data points and across all values of $\mathbf{h}$. Depending on the algorithm P that $\hat{P}$ aims to approximate, we can use as little as 20% of all data points in D; for a hyperparameter $\mathbf{h}$, we can sample anywhere from every second to sixteenth value during training. We thus rewrite Eqn. 5.2 as

$$\underset{\theta}{\operatorname{argmin}} \frac{\sum_{\mathbf{x} \in D'} \sum_{\mathbf{h} \in H'} \|\hat{P}_\theta([\mathbf{x}; \mathbf{h}]) - P(\mathbf{x}, \mathbf{h})\|^2}{|D'| \cdot |H'|}, \tag{5.3}$$

where $D' \subset D$ is the training set and $H' \subset H$ is the set of sampled values of the hyperparameters $\mathbf{h}$, respectively.

Consider a hyperparameter h ranging in some interval $[h_{min}, h_{max}]$, which is sampled $(h_{max} - h_{min})/g$ times. In the remainder of the chapter, we refer to g as the '*gap size*'.

5.2.2 Model Training in Practice

Training P follows the typical workflow for training a neural network except for the generation of the training data, which must include the chosen hyperparameter values in addition to the raw features. To illustrate this, consider using HyperNP to explore UMAP's nearest-neighbor hyperparameter. In this example, we aim to train $\hat{P}$ using 20% of the data and a gap size of 6 to sample UMAP's neighborhood size $\mathbf{h}$ from 2 to 20. We begin generating the training data by randomly sampling 20% of the dataset D. We then project the data four times for $\mathbf{h} \in \{2, 8, 14, 20\}$.

We standardize the features and scale the resulting projections between 0 and 1. Once we have all of our projections, we stack the training data four times (once for each projection) and augment it by concatenating the **h** value used in the projection. Notably, the nearest neighbor value is not transformed. Finally, we train the network following usual procedures, using the augmented stacked matrix as input and the UMAP projections as target values. After training, the model can project the entire dataset across the full hyperparameter range.

5.2.3 Stability Considerations

Particular attention must be given to the issue of projection *stability*. Let P be a function of the hyperparameters $\mathbf{h}$ without considering dataset D. Then, we argue that $P(\mathbf{h})$ should be a relatively smooth function – small changes of $\mathbf{h}$ should yield only small changes in $P(\mathbf{h})$. Indeed, if this were not the case, P would confuse users by producing sudden jumps in the scatterplot when varying $\mathbf{h}$. This issue of stability *vs* hyperparameters $\mathbf{h}$ is analogous to the stability of projections *vs* changes in the data D, which was studied in detail for dynamic projections [VGS+20].

As discussed in [VGS+20], not all projection techniques are stable. We next identify and address two types of instabilities. The methods we use to address these instabilities are simple fixes that make this technique available to anyone and work well for our use case. Other solutions to this sort of stability problem can be found in these excellent surveys [TCL+13, ZGZ+19, HMZA21].

Seeding instabilities: Many stochastic techniques start from a random 2D scatterplot which is iteratively updated to minimize cost [vdMH08, JCC+11]. This makes P not smooth, since even with *no* parameter changes, P depends on the random seed (see *e.g.* the discussion in [EHT20], Figure 11). We address this as follows: Let $\mathbf{h}_i$ be the samples of the hyperparameter $\mathbf{h}$ used during training, in ascending order. We construct the training projection $P_i = P(D, \mathbf{h}_i)$ as usual, *i.e.*, running P with random seeding. Next, we construct P_{i+1} similarly, but use P_i to initialize the low-dimensional embedding of P_{i+1}. A similar strategy was used in [RFaT16] to construct stable t-SNE projections for dynamic datasets.

Eigenanalysis instabilities: Many projection techniques, (e.g. PCA [Pea01]), project data along eigenvectors which are agnostic towards dimension direction, or *sign*. One consequence of using such techniques is that the resulting projections can easily 'flip' along one of the 2D x or y axes upon slight changes in the input data or hyperparameters. Such drastic changes convey no meaning and should be eliminated. We address this problem using a similar solution to pose-invariance used in multimedia retrieval [BBFd07] for comparing arbitrarily rotated shapes. We compute the mean square error (MSE) between a training projection P_i and the previous projection P_{i-1} for all four possible mirrorings and pick the configuration with the lowest MSE for P_i. This minimizes undesired changes between P_i and P_{i-1} during training, making $\hat{P}$ more stable.

5.2.4 Implementation and Tuning

We implemented the dense NN used by HyperNP with Tensorflow [AAB+15] and Keras [C+15]. We used Keras-Tuner's [OBL+19] Bayesian Optimization to find the appropriate number of layers, neurons per layer, and regularization. The tuning consisted of 512 trials, with the first 40 using random search as initial points. The search space included the number of layers (two to six), number of neurons per layer (32 to 512 in steps of 32), dropout probability [SHK+14] (0, 0.25, or 0.5), and whether or not to use batch normalization [IS15]. The final configuration used was a network with layer sizes $|V_1| = 320, |V_2| = 256, |V_3| = 352, |V_4| = 2$. We used ReLU [NH10] activation for all layers except the final one (V_4) where no activation function was used (linear). Between all layers we used batch normalization followed by a dropout probability of 0.25. We chose mean absolute error as the loss function and the ADAM [KB17] optimizer, following NNP [EHT20]. For more on the architecture of the network, see the supplemental material.

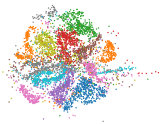
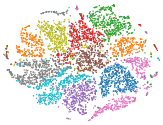
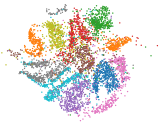
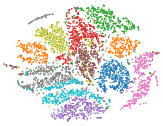
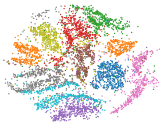
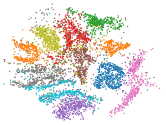
perplexity	t-SNE	HyperNP		
		Gap=2	Gap=4	Gap=8
5				
15				
25				

Table 5.1: t-SNE and HyperNP projections for three different perplexity values. First column: MNIST projected with t-SNE projections at perplexity values of $p = \{5, 15, 25\}$. The following three columns: HyperNP results when trained with gap values of 2, 4, and 8. First, as perplexity increases (within this range) in t-SNE, the global structures become more distinguishable. Further, notice that HyperNP can learn the t-SNE projections that result in visually similar plots to emulate the properties of perplexity, even when the hyperparameter is sampled sparsely (i.e. as the gap size increases from 2 to 8). All projections were trained using 40% of the original dataset.

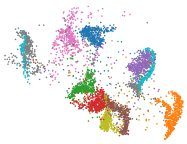
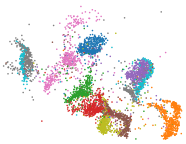
k	UMAP	HyperNP		
		Gap=2	Gap=4	Gap=8
5				
15				
25				

Table 5.2: UMAP and HyperNP projections for three different values of the nearest neighbors parameter. First column: MNIST projected with UMAP projections at nearest neighbor values of $k = \{5, 15, 25\}$. The following three columns: HyperNP results when trained with gap values of 2, 4, and 8. As the number of nearest neighbors increase in UMAP, the global structure is prioritized, losing some fine detail structure. HyperNP is able to capture these changes, properly capturing this-trade off even with a large sampling gap size. All projections were trained using 40% of the original dataset.

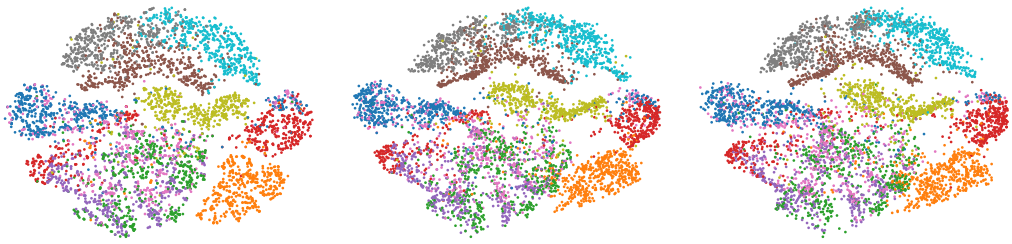
5.3 Data

We use three publicly available high-dimensional datasets that are reasonably large (thousands of samples) and have non-trivial data structure. **MNIST** [LCB10]: 70K images of handwritten digits from 0 to 9, rendered as 28x28-pixel grayscale images, flattened to 784-element vectors; **FashionMNIST** [XRV17]: 70K images of 10 clothing piece types, rendered and flattened as for MNIST; **GloVe** [PSM14]: 70K GloVe (Global Vectors for Word Representation) vectors sampled from the 400k 300-dimensional vectors trained on Wikipedia 2014 + Gigaword 5.

5.4 Applications

We illustrate the value of HyperNP by showing how it approximates changes in the common hyperparameters provided by three popular projection methods: t-SNE, UMAP, and Isomap.

5.4.1 Exploring Perplexity in t-SNE



(a) Ground Truth (b) Train Predictions (c) Test Predictions

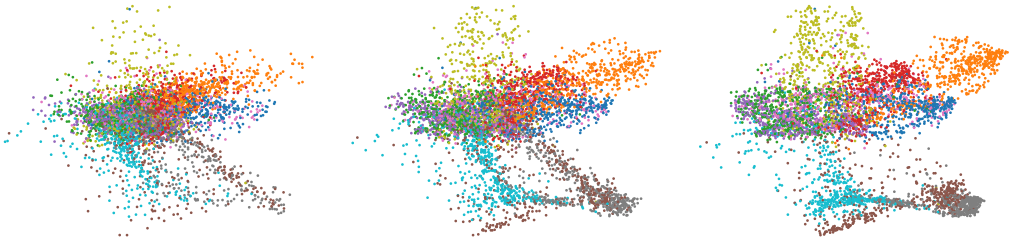
Figure 5.2: HyperNP approximation of the t-SNE projections of the FashionMNIST dataset. From left to right: (a) t-SNE projection of 20% data that is then used to train HyperNP, (b) HyperNP projection of the same data used in (a), (c) HyperNP projection of test data instances unseen during training. This result suggests that HyperNP is adequately learning the t-SNE projection using just 20% of the data as these three images are visually similar.

t-SNE is a variation of Stochastic Neighbor Embedding [HR03] which takes a probabilistic approach to preserving point-wise similarity when mapping high-dimensional data to a low-dimensional (typically 2D) space. First, a Gaussian

distribution is constructed over the data samples so that similar sample pairs have a higher value than dissimilar ones. Another distribution is then constructed for the 2D space. t-SNE then places each sample in 2D by minimizing the Kullback-Leibler divergence between the two Gaussian Distributions. The bandwidth of the Gaussian kernels is chosen so that the perplexity of the high dimensional distribution is equal to a user-defined *perplexity* hyperparameter. This adapts the size of the Gaussian kernel to the underlying density of the input data – smaller kernels are used in dense areas.

As Wattenberg *et al.* [WVJ16] point out, the value of the perplexity hyperparameter in t-SNE strongly affects the produced 2D embedding. Simply put, as perplexity increases, the importance of global features in the data also increases. Yet, the computational cost of robustly exploring the effects of many perplexity values is prohibitive for large data. Figure 5.2 shows how HyperNP can appropriately project out-of-sample data instances. This minimizes training time for each realized projection configuration, $\hat{P}([\mathbf{x}; \mathbf{h}])$, allowing HyperNP to scale to large data with only moderate training cost.

While speed of both training and inference are important to the overall performance of HyperNP, care must be taken to ensure that the projection $\hat{P}$ is adequately learned. As Figure 5.2 shows, HyperNP has learned the t-SNE projection of the FashionMNIST dataset. The image created using only 20% of the data



(a) k=3 (b) k=3.5 (c) k=4

Figure 5.3: Using HyperNP to explore values of UMAP's nearest-neighbor hyperparameter, k, on FashionMNIST. From left to right: (a) $k = 3.0$, (b) $k = 3.5$, (c) $k = 4.0$. While non-natural values for the parameter k are not valid, we show that HyperNP is able to smoothly interpolate between meaningful values without sacrificing projection quality.

(randomly sampled) is similar to the others. This provides strong evidence that using only a small data fraction for training is enough to faithfully represent a chosen projection configuration.

Besides out-of-sample inference for *data*, HyperNP can perform out-of-sample inference for *hyperparameter* values. Table 5.1 shows how HyperNP explores different perplexity values for the MNIST dataset. Looking across rows, Table 5.1 shows that not only does HyperNP maintain similar clustering (columns 2-4) to the ground truth projection (column 1) for hyperparameters not seen in training, but also that increasing the gap size has only a marginal impact on the overall projection quality. This sampling strategy accelerates the initial training of the model without sacrificing quality.

Both Wattenberg *et al.* [WVJ16] and Silva *et al.* [ST02] argue for a thorough inspection of hyperparameters when interpreting projected data. Yet this critical task is often skipped due to its high computational cost. Not only does HyperNP provide the means to explore hyperparameters, but it does this for large datasets and without substantial quality loss. The increased interactivity allows users to better understand how the perplexity hyperparameter influences the final rendering of their data.

5.4.2 Exploring k Nearest Neighbors in UMAP

UMAP is a projection technique based on ideas from topological data analysis and manifold learning. It is implemented using weighted graph techniques [MHSG18] derived from the notion of fuzzy simplicial sets. First, UMAP constructs a k-nearest-neighbor (kNN) graph, whose edge weights model the probability that the edge exists. The graph describes the locally connected manifold that the data is assumed to exist on, and to guide a force-directed graph layout in low dimensions. After initialization using spectral methods, a low dimensional graph is constructed from the projected points. The points are repositioned such that the low-dimensional graph approximates the original weighted graph in high dimension. The objective function minimized by the force directed layout is the total cross entropy of all edge

probabilities, ensuring that the two graphs have similar topologies.

As with other nearest-neighbor based projections, a key hyperparameter of UMAP is k, the number of considered neighbors [WHRS21]. Varying k changes the local scale that the high dimensional manifold considers planar. Features that exist at smaller scale than the neighborhood size are thus lost, with larger features being better preserved in the projection. Silva *et al.* [ST02] provide a discussion of the importance of k with respect to feature size for projection techniques using manifold learning approaches.

Typically, methods that use kNN graphs require all data to be integrated into the graph prior to projection. One challenge of manifold learning based projection techniques is the projection of out-of-sample data which does not exist in the kNN graph used to determine relationships across data points. HyperNP is able to project out-of-sample data without first embedding new points in the existing kNN graph. Table 5.2 shows how HyperNP maintains projection quality of out-of-sample data and also the importance of hyperparameter selection for different gap sizes: smaller neighborhoods lose projection quality faster as the gap size increases.

Care must be taken to ensure that the gap size used to train HyperNP does not overwhelm the smaller values for neighborhood size. The tradeoff between gap size and parameter lower bound is not unique to UMAP, but must be kept in mind for any manifold learning technique. As k changes in the low end of its range, the resulting change in the neighborhood topology is large: changing k from 3 to 4 yields a larger change than changing k from 20 to 21. Projection error increases as small values of k move away from the training k values, which is expected, as the projection change most rapidly for low k.

Unlike t-SNE's perplexity hyperparameter, $k \in \mathbb{N}$. Still, HyperNP is able to both learn and infer all $k \in \mathbb{R}$ values, as shown in Figure 5.3. That is, HyperNP can smoothly interpolate between the strict $k \in \mathbb{N}$ values.

(a) Ground Truth (b) Train Predictions (c) Test Predictions

Figure 5.4: HyperNP approximation of the Isomap projections of the FashionMNIST dataset. From left to right: (a) Isomap projection of 20% data that is then used to train HyperNP, (b) HyperNP projection of the same data used in (a), (c) HyperNP projection of data points unseen during training. This result suggests that HyperNP is adequately learning the Isomap projection using just 20% of the data as these three images are visually similar.

5.4.3 Exploring Neighborhood Size in Isomap

Isomap is a technique that projects data points by first learning the manifold structure in high dimensions via kNN graphs [BS02]. As with UMAP, the most often changed hyperparameter of Isomap is the number of nearest neighbors to consider while building the kNN graph, k. However, unlike UMAP's notion of probabilistic edge weights (See Section 5.4.2), Isomap uses distances as edge weights. These weights enable methods like Dijkstra's Algorithm [Dij59] to approximate a geodesic distance along the induced manifold for each pair of points. After pair-wise geodesic distances are approximated, classical MDS methods are employed to construct the final low-dimensional embedding.

As both UMAP and Isomap make use of manifold learning and represent the manifold structure through kNN graphs, both methods rely on the value of the hyperparameter k to set the importance of global and local feature importance. Silva et al. provide a full discussion of the importance of this hyperparameter with respect to feature size for the class of projection techniques using manifold learning approaches [ST02]. As the underlying functionality of our method does not depend on any computational or theoretical framework unique to a given projection method, new methods are easily adapted to HyperNP's model. As shown in Figure 5.4, HyperNP can approximate Isomap with accuracy. These results are comparable to

the same experiment performed using UMAP (See Figure 5.2). Additional attention must be paid to the value of k with respect to gap size in order to ensure projection quality remains high across all projection configurations for both UMAP and Isomap.

5.5 Evaluation

We evaluate HyperNP's performance through quantitative projection quality metrics and training and inferencing speed.

5.5.1 Performance: Quantitative Metrics

We measure the quality of HyperNP by the following metrics commonly used in the projection literature [EMK+21]. For more information on metrics and experiments, see the supplemental material.

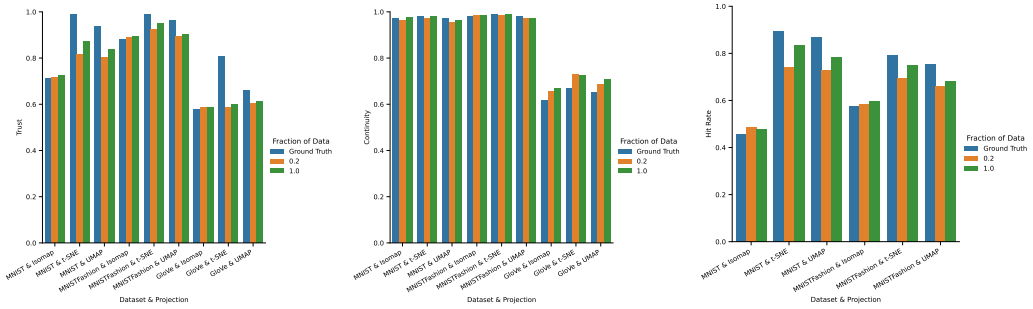
Trustworthiness [VK06] measures the fraction of points in D that are also close in $P(D)$. It tells us how much we can trust the local patterns in a projection (*e.g.,* clusters) to represent actual data patterns. In the definition (Table 5.3), $U_i^{(K)}$ is the set of points that are among the K nearest neighbors of point i in 2D but not among the K nearest neighbors of point i in $\mathbb{R}^n$; and $r(i,j)$ is the rank of point j in the ordered-set of nearest neighbors of i in 2D. We choose $K = 7$ in our experiments, in line with [vdMPvdH09, MMT15b, EMK+21].

Continuity [VK06] measures the fraction of close points in $P(D)$ that are also close in D. In the definition (See Table 5.3), $V_i^{(K)}$ is the set of points that are among the K nearest neighbors of point i in $\mathbb{R}^n$ but not among the K nearest neighbors in 2D; and $\hat{r}(i,j)$ is the rank of point j in the ordered set of nearest neighbors of i in $\mathbb{R}^n$. Similar to trustworthiness we choose $K = 7$ following convention.

Hit Rate [PNML08b] measures the proportion of the neighbors of a point in a projection that have the same label as the point. In the definition (See Table 5.3), $W_i^{(K)}$ is the set of points that are among the K nearest neighbors of point i in 2D that share the same label as point i. For ease of computation we calculate hit rate on a uniform sampling of 10% of the data, unlike the previous two metrics.

Metric	Definition	Range
Trustworthiness	$1 - \frac{2}{NK(2n-3K-1)} \sum_{i=1}^{N} \sum_{j \in U_i^{(K)}} (r(i,j) - K)$	$[0, \mathbf{1}]$
Continuity	$1 - \frac{2}{NK(2n-3K-1)} \sum_{i=1}^{N} \sum_{j \in V_i^{(K)}} (\hat{r}(i,j) - K)$	$[0, \mathbf{1}]$
Hit Rate	$\frac{1}{N} \sum_{i=1}^{N} \lvert W_i^K \rvert / K$	$[0, \mathbf{1}]$

Table 5.3: Projection quality metrics used in our evaluation with optimal values in bold in the Range column.



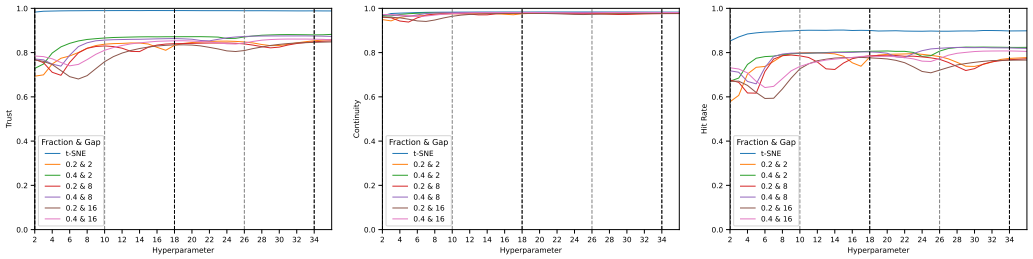
(a) Trustworthiness (b) Continuity (c) Hit Rate

Figure 5.5: In this figure, we present the trustworthiness, continuity, and hit rate scores associated with the ground truth projections as well as HyperNP trained using a gap size of 16, and data fractions of 0.2 and 1.0. We have averaged the scores over all of the hyperparameter values from 2 to 50. Comparing the ground truth score to HyperNP's score for each dataset and projection combination illustrates that HyperNP is able to perform close to the ground truth even under extreme sampling conditions (i.e., when only 20% of the data is used for training).

We next explore different hyperparameter settings to determine their effect on HyperNP's quality, namely the gap size and the data fraction used for training (see Section 5.2). We first examine the metrics in aggregate, by taking their averages across hyperparamter values. For each dataset and projection combination, we fix the gap size and training fraction and compute the mean trustworthiness and continuity values across either perplexity or k, from two to fifty. We present these results in Figure 5.5. Notice that across most dataset and projection combinations within the figure, our method produces trustworthiness, continuity, and hit rate scores within a few percentage points of the ground truth projection method. Decreasing the size of the training data by decreasing data fraction from 1.0 to 0.2 (orange bars versus blue bars in Figure 5.5), a 5-fold decrease, shows only a minimal loss in scores with the benefit of decreased training time. These results inspire confidence; they show that HyperNP is able to infer projection locations of out-of-sample data, generalizing

from a small subset of the entire dataset.

In Figure 5.6, we plot trustworthiness and continuity for UMAP and HyperNP trained with UMAP, varying the value of the hyperparameter k. We further vary the data fraction and gap size used to train HyperNP. In this experiment, HyperNP performs similarly to the ground truth UMAP for continuity, and at worst within approximately 85% of hit rate and 80% of trustworthiness. The relatively similar continuity scores, and different trust scores suggest that approximated projections have the same level of false neighbors as the real projections, but have more missing neighbors. False neighbors are points close in low dimensional space that were not close in the high dimensional space, while missing neighbors are the opposite [NA19].



(a) Trustworthiness (b) Continuity (c) Hit Rate

Figure 5.6: Trustworthiness, continuity, and hit rate for HyperNP trained with t-SNE, on the MNIST dataset, for different hyperparameter values. Light gray vertical lines show parameter values used for training with a gap size of 8; dark gray ones show parameter values used for training with a gap size of 16. This dataset and projection combination yielded the greatest difference in trust scores between ground truth and HyperNP out of all of the combinations tested. Despite this, quality does not decrease substantially even during the dips far from sampled points. It is worth noting that some visual samples from the purple line representing a gap of 8 and a data fraction of .4 can be in the last column of Table 5.2.

5.5.2 Performance: Training and Inferencing Speeds

Training Speed: Figure 5.7 shows training times for HyperNP for different gap sizes and data fraction values. The figure illustrates the trade-offs in training speed for different data fractions and gap sizes. As detailed in Section 5.2, the neural network will be trained on the number of sampled instances times the number of hyperparameters. For our experiments we chose what we expect an average case

would be, examining hyperparameter values from 2 to 50. It should be noted that Figure 5.7 shows overall training time for HyperNP as a function of dataset size. The number of training instances is determined not just by this value, but the number of hyperparameter values and fraction of this number used.

As shown in Figure 5.7, in extreme cases when the gap size is too low, training HyperNP could take up to 80 minutes for a dataset with 50,000 points. However, as seen in Section 5.5.1 using a data fraction value of 0.4 (i.e., sampling rate of 40%) and gap size of 8 produces HyperNP projections of reasonable quality. With these configurations, most datasets with less than 50,000 points can be trained within 20 minutes. The time to train HyperNP in Figure 5.7 includes the time to project the ground truth data using Multicore-TSNE [Uly16] with four cores.

The HyperNP model is trained with a batch size of 32 in Figure 5.7. We note that the batch size will often have a considerable effect on not only the wall time it will take a network to converge, but also the final performance of the model. A more thorough discussion of this topic can be found in a number of papers [KMN+17, ML18, HLT19]. The training speed tests were executed on a 2-way, 16-core Intel Xeon Silver 4216@2.1 GHz with 256 GB of RAM, and an NVidia GeForce 1080 Ti GPU.

Inferencing Speed: We evaluate the inferencing speed of HyperNP. Table 5.4 shows the HyperNP's inference run-time using a batch size of $20k$ instances. We see that HyperNP remains interactive with $500k$ data points, performing inference under 500ms [LH14]. Overall, the speed of HyperNP scales linearly with the number

Number of Instances	Seconds
56,000	0.192241
168,000	0.264158
336,000	0.370889
504,000	0.479788
616,000	0.551125
728,000	0.624992
840,000	0.695622
952,000	0.766805
1,064,000	0.839778

Table 5.4: HyperNP's inference speed

of data points projected. The inferencing speed tests were executed on a 8-core Intel

i7-6700k@4 GHz with 64 GB of RAM, and an NVidia GeForce 1080 GPU, with 8GB of VRAM.

5.6 Using HyperNP to Approximate iPCA

Beyond approximating projection methods, HyperNP can have additional applications in visual analytics given its use of a neural network. In this section, we describe how HyperNP can be used to approximate iPCA [JZF+09], an interactive visual analytics technique for supporting exploration of high-dimensional data with PCA. iPCA is a technique that allows users to interactively adjust the weighted contributions of data dimensions and observe the effects in a PCA projection. Similar to HyperNP, iPCA allows a user to interactively manipulate projections using sliders in real time.

Figure 5.8 shows HyperNP's approximation of iPCA. The training of HyperNP is the same as described in Section 5.2. In Eqn. 5.1, $\mathbf{h}$ now is a n-dimensional vector that represents the scaling factors in iPCA where n is the number of data dimensions, each controlled by a slider in iPCA. We see that HyperNP is successful in emulating iPCA: interactions with the sliders in HyperNP produce the same projections as iPCA while retaining the interactive speed. This is important in terms of genericity of HyperNP. iPCA is designed specifically around PCA, using Online SVD [Bra03] to reduce the computation cost of singular value decomposition from $O(N^2)$ to $O(n^2)$ for N samples of n dimensions. As a result, it is hard to replace iPCA with another projection technique while maintaining interactivity. In contrast, doing so with HyperNP is simple – replacing PCA by its HyperNP approximation requires just changing the function P in Eqn. 5.3 from PCA to another projection technique. This supports the use of HyperNP to generalize high-dimensional data exploration techniques (like iPCA) beyond their associated projection methods. As future work, we will investigate other interaction techniques in visual analytics that can be made more generalizable with the use of HyperNP.

5.7 Discussion, Limitations, and Future Work

In this section, we discuss both conceptual and practical considerations of using HyperNP. We conclude by presenting the limitations of HyperNP and some possible future research directions.

5.7.1 Sampling and Exploring Hyperparameters

This chapter shows HyperNP is able to approximate important hyperparameters associated with a projection technique. Exploring projection hyperparameters is exemplified in Tables 5.1 and 5.2. These images highlight how HyperNP captures much of the structure of the original projections using only a fraction of the data for training of the system. Similar to interpolation, increasing the gap size during HyperNP training introduces additional deviation from ground truth. These errors are most evident with low values of perplexity and k for tSNE and UMAP, respectively. At these small hyperparameter values, the number of samples naturally defining the local structure in the embedding space is likely insufficient to provide enough signal for HyperNP to approximate robustly. However, as the hyperparameter values increase, the local topology captured by the methods is less influenced by small changes to the hyperparmeter, allowing HyperNP to better approximate the projection, even when using larger gap sizes.

5.7.2 Conceptual and Practical Considerations of HyperNP

Using deep learning to approximate projections raises some important considerations. Since a neural network does not consider the semantics of a hyperparameter, it can generate projections using invalid configurations, see *e.g.* the use of non-integral k values for UMAP (Section 5.4.2). In this case, HyperNP infers the projection by interpolating the hyperparameter setting between integral values used during training. While formally, this goes beyond the assumptions of UMAP (*i.e.*, k is an integer), this interpolation has yielded reasonable results for all k values, with no negative consequences noticed (see Figure 5.3).

The hyperparameter sampling strategy is also an important factor for HyperNP. While t-SNE's perplexity and UMAP's k-nearest neighbors are both hyperparameters, they respond differently to sampling. Figure 5.6 shows how quality changes when hyperparameters move farther from training values. For t-SNE, neither continuity nor trustworthiness change much. Still, while these metrics show a dip in the middle of large gaps (farthest from k training values), we still get high quality especially past the smaller hyperparameter values. This tells us that the gap size used to train HyperNP has greater impact for UMAP than for t-SNE; yet, even a gap size of 16 only marginally changes the projection quality. Summarizing, choosing the sample step (gap size) of hyperparameters needs to be studied individually for different projection techniques.

5.7.3 Limitations and Future Work

In this section we present potential limitations of HyperNP and discuss avenues to further develop this work.

Multiple Hyperparameters: Section 5.5 discusses the training time for HyperNP for a single hyperparameter. This time can grow exponentially as the number of hyperparameters $|\mathbf{h}|$ increases. If the complexity of training a traditional network is $O(\mathrm{K})$, the complexity of training HyperNP with one hyperparameter is $O(\mathrm{K} \cdot |H'|)$ with $|H'|$ as described in Eqn. 5.3. For $|\mathbf{h}|$ hyperparameters, this becomes $O(\mathrm{K} \cdot |H'|^{|\mathbf{h}|})$. Further work is needed to develop methods to reduce the training cost of HyperNP for projection algorithms with many hyperparameters. Additionally, an interface for intuitively exploring all hyperparameters associated with a projection method would need to be developed.

Hyperparameter Selection: Due to the current limitation of HyperNP in learning multiple hyperparameters, the user must carefully choose the most important hyperparameter to explore. In this chapter we select the hyperparameters that affect the tradeoff between global and local preservation as the user's primary exploration dimension, following conventions established by Wattenberg et al. [WVJ16]. Prior work like VisCoDeR [CHAS18] and t-viSNE [CMK20] allow users to explore the

effect of other hyperparameters that can affect the quality of the projections. We leave the evaluation of HyperNP on other hyperparameters for future work.

Training Data: Like any deep learning technique, HyperNP is influenced by the quality of training data. As detailed in Section 5.5.1, performance on the GloVe dataset is lower than the other two datasets, but performance relative to the ground truth projections is still high. HyperNP faithfully reconstructs the training data; however, the quality of a projection is directly related to the quality of the initial projections used during training. Even if the training data as a whole is of good quality, it is possible to introduce bias through sampling. Unintentional bias is introduced by any sampling strategy (to create training sets) that results in a different distribution than the dataset at large. We note that the problem of sampling training data is not unique to our method. We leave as future work strategies to mitigate its effects on approximating projections.

Training data quality is important when considering performance in the context of outliers and large data. Datasets containing outliers pose the same challenges for HyperNP as they do for the both the underlying projection technique being used and the deep learning architecture learning the projection. Exploring novel methods to mitigate the impact of outliers is beyond the scope of this work, but advances in this topic are easily integrated with the system. Since HyperNP is, at its core, a deep learning implementation, increasing the dataset size is generally considered to be an important step to improve performance. However, as discussed above, maintaining a similar distribution in the training data with respect to the overall dataset is important and must be considered when deciding when to update the HyperNP model representing the projection operator.

Smoothness Perception: The technique described in Section 5.2.3 maintains projection stability and visual coherence between frames as users interactively change a hyperparameter value. When combined with the high interactivity of HyperNP, this may induce a false sense of continuity concerning the *properties of the underlying learned projection method*. For example, consider approximating UMAP with HyperNP (Section 5.4.2). When the hyperparameter k changes from k to $k+1$,

there is no guarantee that the two learned manifolds share correspondences, so the perception of smoothness created by HyperNP may be misleading. Conversely, for iPCA (Section 5.6), weight changes should result in a smooth animation as PCA rotates the underlying coordinate system to maximize the projected data variance. In summary, the built-in perception of smoothness of HyperNP should not always be interpreted as reflecting mathematical smoothness properties of the learned projections. Additionally, the neural network architecture learning the projection operator enables the use of potentially unintended parameter values. For example, a non-integer value of k in the UMAP operator is possible to approximate using HyperNP. Of course, the interpretation of a non-integer number of neighbors is challenging.

5.8 Conclusion

In this chapter, we discussed HyperNP, a deep learning approach to approximating projections that enables real-time interactive hyperparameter exploration. We showed that HyperNP can learn to infer projections of several techniques, for a wide range of parameter values, and different real-world datasets, using only a small fraction of the data and hyperparameter values. HyperNP performs in real-time for reasonably large datasets, can be generalized to any projection technique (and hyperparameters), and produces stable animations of the resulting projections upon parameter variations. This enables users to explore the structure of a dataset interactively, getting a better understanding of the dataset itself as well as the effect of the hyperparameter in question. We demonstrated HyperNP through the exploration of the key hyperparameters of t-SNE, UMAP, and Isomap.

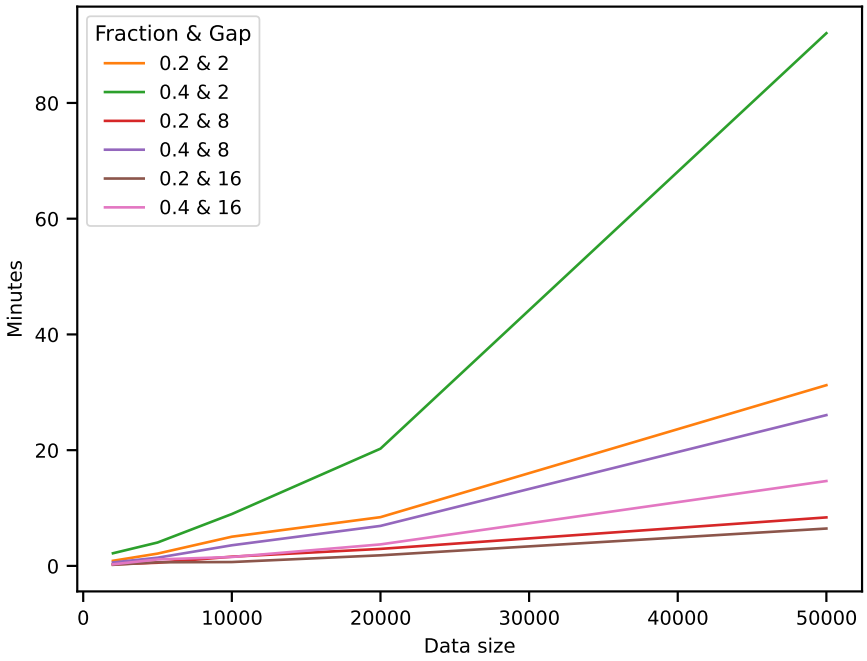


Figure 5.7: HyperNP training speed with the MNIST dataset and t-SNE training projection using different amounts of training data (data fractions of 0.2 and 0.4) and gap sizes (2, 8, and 16). Smaller gap sizes result in larger $|\mathbf{h}|$ and thus longer training times. For smaller datasets (less than 20k), HyperNP can be fully trained within 10 to 20 minutes.

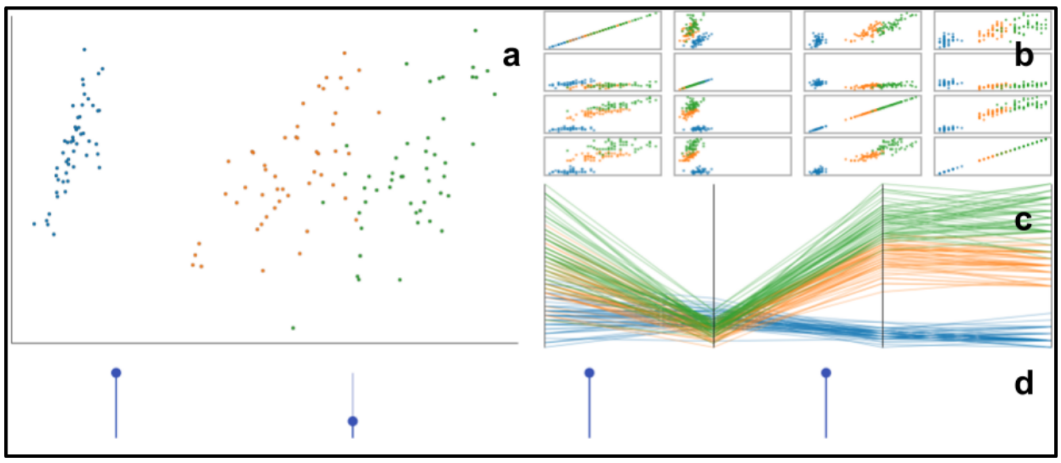


Figure 5.8: A HyperNP implementation of iPCA. Each slider (d) corresponds to a dimension of the input dataset (Iris) [And35]. Adjusting a slider scales a feature from 100% to 0%. This is reflected in the parallel coordinates plot (c) and scatterplot matrix (b). HyperNP re-projects (a) the dataset in real time as the user interacts with the sliders (Section 5.5.2).

Chapter 6

Unprojection

This chapter discusses NNInv, an algorithm for inverse projection fast enough to enable real-time high-quality inversion. Most importantly, this enables hypothesis generation that allows users to reason about full data instances within the empty space of their projections and enables fast, dense-map backgrounds that illustrate distortions and errors within the projection. Thus enabling users to understand their complicated data and projection better.

6.1 Problem Statement

High-dimensional data is created at unprecedented rates by scientific fields as diverse as information technology, bioinformatics, and astronomy [Buh11]. As a result, there is a growing need for visualization and interaction methods for high-dimensional data. A common choice is to project the high-dimensional data to two dimensions using methods such as t-SNE [vdMH08], PCA [Pea01], LLE [RS00], or UMAP [MHSG18], among the many other existing options [EMK+21].

Projections allow better insight into the overall structure of data and can be enriched by interactions that allow users to reason about the corresponding high-dimensional data by selecting, brushing, and querying the 2D scatterplots they create. For example, t-SNE has allowed computational biologists to investigate human genetic data, revealing otherwise obfuscated population stratification [LCYH17].

However, any projection technique will create errors when mapping complex and high-dimensional datasets to a low number of dimensions [MCMT14, NA19]. Moreover, projections are often complex algorithms, so the way they map the high-dimensional data to low-dimensional space can be difficult for users to fully interpret. As such, additional mechanisms need to be complement projections to empower users to better explore the high-dimensional data.

Recently attention has turned to *inverse-projection*, a process that allows one to compute the inverse mapping from the projection space back to the original high-dimensional data space [dBD+12]. Also called back-projections, these methods help users to explore projections by allowing a user to interactively query the projection space to find high-dimensional data points. These points correspond to specific locations in the low-dimensional projection. Inverse-projections are also instrumental in explaining the decision boundaries of machine learning classifiers [REHT19] and data augmentation scenarios [dBD+12]. In contrast to the many existing projection techniques [EMK+21], only a handful of inverse projection algorithms exist, including iLAMP [dBD+12] and its extension that uses radial basis functions (RBFs) [ABMC+15]. Algorithms like iLAMP and RBF are quite slow, and have multiple free parameters, making it hard to use them in interactive data exploration scenarios [REHT19].

In this chapter, we present NNInv, a technique for computing the inverse of any projection using a deep learning approach. Our idea is inspired by the recent work of Espadoto *et al.* [EHFT20a] that demonstrates that deep learning can learn to imitate the style of any projection technique, and is parametric and stable to data changes (and thereby offers out-of-sample capabilities). Following their approach, we show that NNInv is a scalable, robust, high-quality inverse projection method which supports multiple applications.

Using NNInv, we introduce three use cases across a number of well-known datasets to illustrate how the use of inverse-projection can improve the user's interaction, exploration, and understanding of high-dimensional data in a 2D visualization. Additionally, we provide an evaluation of iLAMP, RBF, and NNInv in terms of

scalability and accuracy. To this end, we provide a novel visualization for evaluating the joint quality of a pair of inverse-projection and direct-projection methods. We make a point of studying the NNInv inverse-projection method on two synthetic datasets with well-known topology (*i.e.*, a 3D sphere dataset and a 3D swissroll dataset), allowing us to illustrate the behaviors of the learned inverse function.

Applications for this work are numerous. First, we use inverse-projections to explore the "empty" spaces in a 2D projection of high-dimensional data. While the user interactively brushes such spaces, high-dimensional instances corresponding to the visited 2D points are synthesized and displayed, thus allowing one to form a better mental map of how the 2D image represents the entire high-dimensional data *space*, beyond how a 2D scatterplot represents a high-dimensional *dataset*.

Second, we present a visualization of the decision boundary of an ensemble classifier. Visualizing cluster boundaries can help users see patterns within the data and the behaviors of the classifiers (see the classifier comparison by Scikit-Learn [SL]). We show that the use of an inverse projection method such as NNInv makes it possible to visualize this important information with high-dimensional data (see Fig. 6.2). Finally, we introduce a gradient map visualization to help users find projection artifacts. This method highlights regions where the projection shrinks and expands the relationships between points by visualizing the rate of change between the learned 2D embedding and the original high-dimensional space. We show that NNInv provides an alternative approach to helping the user "see" the high-dimensional space in 2D.

In summary, the main contributions of this chapter are:

- A deep learning approach to inverse-projection, which is fast enough to be used at scale.
- A comparison to existing inverse-projection methods.
- An exploration of the behavior of NNInv on datasets with well-known topology.
- Two novel visualizations for evaluating inverse-projection methods.

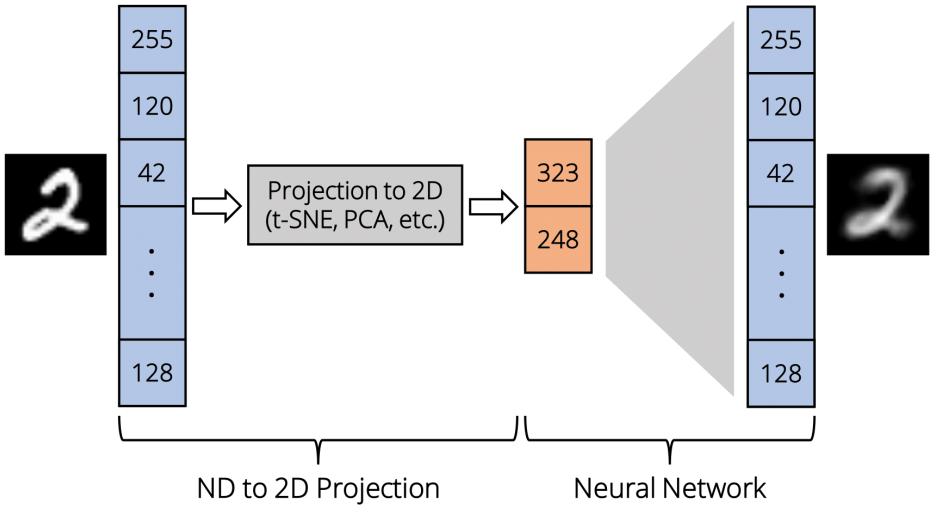


Figure 6.1: End-to-end pipeline of direct and inverse projections. A high-dimensional vector representing an image of the number two is fed into a projection technique. The orange numbers are the 2D projection of this vector. The inverse projection NNInv takes this 2D representation and yields a high-dimensional vector corresponding to it.

- A showcase of visual exploration techniques enabled by inverse-projection.

6.2 Background

Let us recall the notation detailed in Section 2.2, and Section 2.4.0.1. We also briefly discuss latent spaces as it relates to this chapter in Sections 6.2.1 and 6.2.1.1.

6.2.1 Latent Spaces with Neural Networks

Recent developments in machine learning and AI have shown that deep learning approaches are both accurate and flexible when used and trained properly [GBC17]. In general, neural network encoders work by learning a mapping from the input data space $\mathbb{R}^d$ to a lower dimensional representation $\mathbb{R}^m$ called the *latent space*. This mapping, conceptually similar to our projection P, is often difficult to interpret as the latent dimensions are abstract. Moreover, the neural network's operation is harder to understand than the equivalent operation of a typical projection function P.

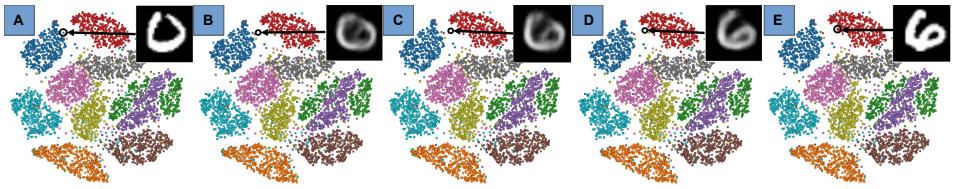


Figure 6.2: Example of back projection-enabled interpolation in the original space, as a user explores regions between original data points. As the user moves the mouse from a point representing the digit 0 (A) to a point representing the digit 6 (E), the pixel under the mouse is used as input to the back projection. B-D show how, as the user moves closer to the original point, the recovered high-dimensional data is meaningfully interpolated.

6.2.1.1 Interpreting the Latent Space

Interactive visualization tools have been developed to help with analysis tasks that give a better understanding of latent spaces. In particular, when a neural network model has a generative component (*e.g.* autoencoders and Generative Adversarial Networks), its latent space can be explained by bringing data points back to the original space via its generative component. Liu *et al.* [LJLH19] presented a latent space cartography (LSC) visual analysis system for vector space embeddings. The LSC system was created to address common interpretation tasks for latent spaces. It provides a means to both quantify attribute vector uncertainty and compare multiple attribute vectors. Spinner *et al.* [SKGD18] also used latent spaces to visually compare autoencoders with variational autoencoders. A number of techniques have been developed in order to try to disentangle the latent features of autoencoders [HMP+17, KM19, CLGD19]. A recent work by Gou *et al.* moved these advances forward within a full visual analytics system for traffic light detection [GZL+20]. Additional visualizations making use of, and explaining, latent spaces are discussed in a recent survey [GTdS+18].

6.3 Learning the Inverse Projection

Figure 6.1 shows the operation of NNInv. Given a dataset $D \subset \mathbb{R}^d$, of N points, let $P(D) = \{P(\mathbf{x}_i) | \mathbf{x}_i \in D\}$, $1 \leq i \leq N$ be its projection by any user-chosen projection method P. In practice, $P(D)$ is a $m = 2$ dimensional scatterplot, so $P(D) \in \mathbb{R}^2$.

NNInv constructs an approximation $B : \mathbb{R}^2 \rightarrow \mathbb{R}^d$ of the inverse P^{-1} of P by using deep learning. Let

$$\hat{\mathbf{x}} = B(\boldsymbol{\theta}, \mathbf{y}) \tag{6.1}$$

be a d-dimensional point $\hat{\mathbf{x}}$ inferred by the neural network B from a 2D point $\mathbf{y}$. Here, $\boldsymbol{\theta}$ are the learned parameters of the function B (*i.e.*, the weights of the network). To train the model, we minimize the loss between each predicted $\hat{\mathbf{x}}_i$ and true $\mathbf{x}_i$ within the training set ($D_s \subset D$, $P_s \subset P(D)$) using some loss function.

6.3.1 Data

We used five different datasets across our evaluation and proposed applications.

MNIST: This dataset [LCB10] has $N = 70K$ grayscale images of hand-drawn digits, zero through nine. Each image is at a resolution of 28×28. The images have been translated so that the center of mass of the pixels is at the center of the image. The MNIST dataset is commonly used to illustrate and measure the quality of projection techniques [vdMH08, vdMPvdH09, EMK+21, EHT20, ERH+19].

Fashion-MNIST: This dataset [XRV17] is constructed in the same manner as the original MNIST dataset, but contains pictures of different items of clothing. It was designed as a slightly more difficult replacement for the MNIST dataset.

Blobs: This synthetic dataset has $N = 70K$ points sampled from a Gaussian distribution with 5 different centers (clusters) in $d = 50$ dimensions.

Sphere: This dataset consists of $N = 8K$ points uniformly sampled from a 3D unit sphere. It allows us to clearly demonstrate the behaviour of the projection techniques included, and more importantly, offer a simple illustration of our gradient map visualization.

Swiss Roll: This dataset consists of $N = 70K$ points sampled from a densely-sampled 2D patch which was smoothly mapped to a "roll" in 3D. It is commonly used to gauge the capability of projections to "unroll" the data back to its 2D configuration [dBD+12, JCC+11, BS02].

6.3.2 Implementation

Dataset	L_1	L_2	L_3	L_4	MAE	STD
Blobs	64	128	256	512	0.036941	3e-06
	128	256	512	1024	0.036944	2.5e-05
	256	512	1024	2048	0.036945	2.1e-05
	640	1280	1280	640	0.03695	3.3e-05
	240	240	240	240	0.036961	3.1e-05
MNIST	128	256	512	1024	0.06241	0.000425
	640	320	320	640	0.062606	0.000113
	640	320	320	640	0.062787	0.000551
	480	480	480	480	0.06303	5e-05
	480	480	480	480	0.063168	0.000258
FashionMNIST	1024	2048	4096	8192	0.072804	0.000411
	1280	2560	2560	1280	0.072873	0.000136
	512	1024	2048	4096	0.073108	0.000268
	1280	640	640	1280	0.073209	6.5e-05
	256	512	1024	2048	0.073214	0.00064
Swiss	64	128	256	512	0.011698	0.000489
	256	512	1024	2048	0.012288	0.001286
	640	1280	1280	640	0.013136	0.000805
	160	320	320	160	0.013209	0.001
	640	320	320	640	0.013929	0.001523

Table 6.1: Top five configurations per dataset sorted by lowest mean absolute error (MAE), computed by averaging three different runs. Columns L_i show the number of neurons used in the respective hidden layers.

We next describe the design and tuning of the neural network used to learn the inverse projection. Following [EMH19], and also the method used to tune NNP [EHFT20a], we used grid search to explore different architecture configurations: total number of neurons, neurons per layer, and dropout values.

We ran the grid search across the four datasets introduced in Sec. 6.3.1. As the direct projection P, we used t-SNE, which was earlier shown to be the hardest projection from a set of nine different projections to mimic via deep learning [EHT20]. Hence, we believe that t-SNE is also a hard challenge to invert via NNInv. We varied the training-set size $|D_s|$ between 5250, 10500, 21000, and 42000 samples. To account for variation in random initialization of the neural network weights, we ran each configuration three times and averaged the results into a single error score. We

measure quality via mean absolute error (MAE) $\frac{1}{N}\sum_{i=1}^{N}|x_i - \hat{x}_i|$, and also provide its standard deviation across the three runs. Training is stopped automatically on convergence, defined as the moment when the validation loss stops decreasing. We next discuss the hyperparameters investigated.

Network Architectures: We restricted ourselves to fully-connected layers and used four hidden layers $(L_1, \ldots, L_4)$ in each configuration. We varied the network shape and number of neurons in each layer. The total number of neurons ν in each network varied between 240, 480, 960, 1920, 3840, 7680, and 15360. We experimented with four network shapes (see Table 6.2).

Shape	$\lvert L_1\rvert$	$\lvert L_2\rvert$	$\lvert L_3\rvert$	$\lvert L_4\rvert$
straight	$\nu/4$	$\nu/4$	$\nu/4$	$\nu/4$
wide	$\nu/6$	$\nu/3$	$\nu/3$	$\nu/6$
bottleneck	$\nu/3$	$\nu/6$	$\nu/6$	$\nu/3$
fan-out	$\nu/15$	$\nu/7.5$	$\nu/3.84$	$\nu/1.875$

Table 6.2: Shapes of the tested networks for learning NNInv.

Activation Functions: We used a ReLU activation function for all hidden layers. Since the input data D is normalized such that each of the d dimensions ranges over $[0, 1]$, we used a sigmoid activation function on the output layer.

Regularization: We used both early stopping and dropout, with dropout probabilities of 0.125, 0.25, 0.5. Experiments showed dropout was not generally effective. We believe that this is due to the fact that overfitting is unlikely given that we used smaller networks and early stopping.

Loss Function: For the loss function we used MAE.

Optimizer: We used the Adam optimizer [KB17], given its good performance with NNP [EHT20, EHFT20a].

Table 6.1 shows the MAE and standard deviation (STD) results for the

top-five configurations of the 1536 tested ones, *i.e.*, the ones obtaining lowest error. The full results including all configurations tested is available in the supplemental material. The best architectures for each dataset either had the same number of neurons in each layer, or used a widening architecture which doubles or quadruples the number of neurons in each successive layer.

Our results suggest that smaller architectures can be used other than the original architecture from Espadoto *et al.* [ERH+19]. The only dataset that performs better with more than 1920 neurons is the FashionMNIST dataset. For this dataset, we obtain a slightly lower MAE when using 7680 neurons. However, as in most cases observed, the error decrease is negligible compared to the increase in complexity (network size). Summarizing our findings from the tested datasets, we offer suggestions for future experimentation: (1) networks should follow a straight or fan-out style shape as described in Table 6.2, and (2) even relatively small networks can perform exceptionally well at this task as seen in Table 6.1.

6.4 Applications of Inverse Projection in Visual Analytics

Traditional error calculations may tell something of the overall loss, *i.e.*, going from high-dimensional space to 2D and back to the original space. However, a robust analysis of an inverse-projection technique must include more than just this type of error. This section focuses on a qualitative evaluation of inverse projections using applications that are of interest to the visualization community. In particular, we explore three use cases of inverse-projection in visual analytics: (1) direct interpolation of high-dimensional data using the 2D screen, (2) leveraging the generation of high-dimensional data across the screen to per-pixel color classifier agreement, and (3) using the generated high-dimensional data to illustrate high gradient areas of the projection.

6.4.1 Case Study 1: Dynamic Imputation

One shortcoming of the current use of projection methods is that the projections are "one-way streets." From a user interaction and exploration standpoint, the most that a user can do using such techniques is to select a data point in the (2D) visualization and look up the original values of that point in high-dimensional space. Due to this limitation, the user's exploration of the data is restricted. For example, the user would have no easy way of knowing why two data points appear close to each other in the 2D space, or what other data points, if they existed, would appear near or between these points.

6.4.1.1 Example with MNIST

In this case study, we demonstrate the use of inverse-projection to perform "dynamic imputation." The inspiration for this case study comes from recent works by Cavallo *et al.* [CD18b] that explores inverse-projection with PCA and autoencoders, and Kwon *et al.* [KM20b] that generates graph layouts from the user's interactions in a 2D latent space.

Consider Figure 6.2: The user can select projected data points in a 2D visualization (of the MNIST dataset) and see their original values (see Figure 6.2A and Figure 6.2E), similar to traditional visual analytics systems. However, with the use of inverse-projection, the user can *also* select an *"empty"* space between these data points (see the three inner images). The inverse-projection function implicitly performs imputation (*i.e.*, generating a new data point) when performing inference over the 2D pixel location to find its position in high-dimensional space.

Since the inference step of a trained neural network is fast, this computation can be done in a web browser and be made fully interactive using mouse hovering. In this example, the computation time of inverse-projecting a point in tensorflow.js on an Intel i7-8650U CPU is below 10 milliseconds. With this high degree of interactivity, the user can quickly explore both the high-dimensional dataset as well as the high-dimensional *space* (between data points) itself. Figure 6.3 further showcases the

ability of using inverse-projection to interpolate between clusters. Here, the images furthest to the left and right represent two visually distinct objects (*i.e.*, a pair of pants and a dress, and the digits 6 and 5). The images in between are interpolations generated by the inverse-projection algorithm.

6.4.1.2 Evaluation of the Inverse Projection

Using this framework, we can also visually evaluate the quality of the inverse-projection algorithm. Specifically, instead of selecting an "empty" pixel, a user can select the 2D position of an existing data point. We can then compare the original values of the data point with the values generated by the inverse-projection algorithm. For example, the two images on the upper left side of Figure 6.4 are two different styles of pants from the original Fashion-MNIST dataset. The two images directly below, on the lower left side, are those generated by the inverse-projection. Similarly, images on the upper right side of Figure 6.4 are from the original MNIST dataset, and images on the lower right side of Figure 6.4 are generated. In both cases, the generated images are "blurrier" than the originals. However, it is shown that the inverse-projection function has successfully learned the important visual features of these images and can reproduce them with high fidelity.

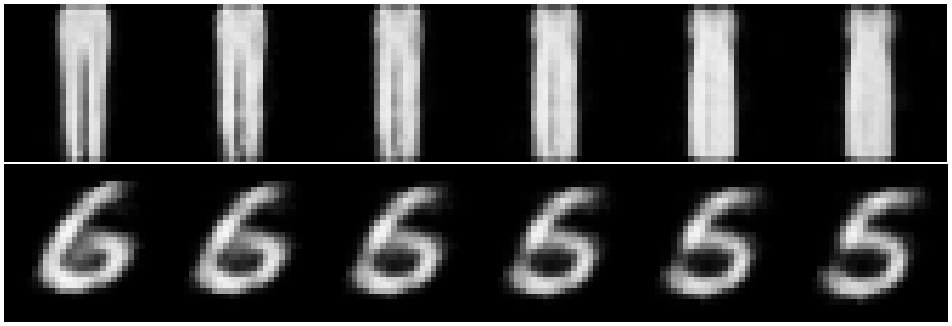


Figure 6.3: Images generated when moving from one cluster to another within a projection, and feeding the x and y coordinates into NNInv. Top row: FashionMNIST, moving from class "pants" to class "dress"; bottom row: MNIST, moving from class 6 to class 5.

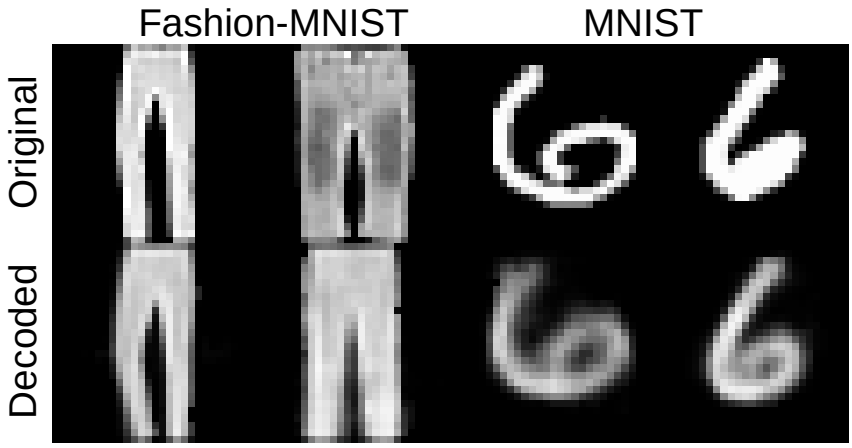


Figure 6.4: Data points from the test portion of both datasets. The images labeled as "Generated" illustrate the inverse-projection corresponding to the projection of the images labeled as "original".

6.4.1.3 Implications to Visual Analytics

Although we used two relatively simple image datasets in this case study (MNIST and Fashion-MNIST) for illustrative purposes, the use of inverse-projection for dynamic imputation should be extendable to visual analysis of other high-dimensional datasets, including temporal data, geographic data, and tabular data. As such, having an accurate inverse-projection function in a visual analytics system can allow the system designer and the user to explore high-dimensional data in ways that have not been possible. For example, in the context of business analysis, the use of inverse-projection for data imputation can serve as a *"hypothesis generator"* (*e.g.*, Figure 6.2). With inverse-projection, the user can interpolate between the 0 and the 6 from the original data, and use inverse-projection to generate hybrid examples between. While the generated data points are estimates of the inverse-projection function, they may nonetheless serve as potential hypotheses for an analyst to further explore.

6.4.2 Case Study 2: Model Agreements

Previous work in defining and interpreting back projections has shown that creating dense pixel maps in the 2D projection space can provide additional insight into the behavior of classification type tasks [ERH+19, MHT18, REHT19]. Figure 6.5 shows

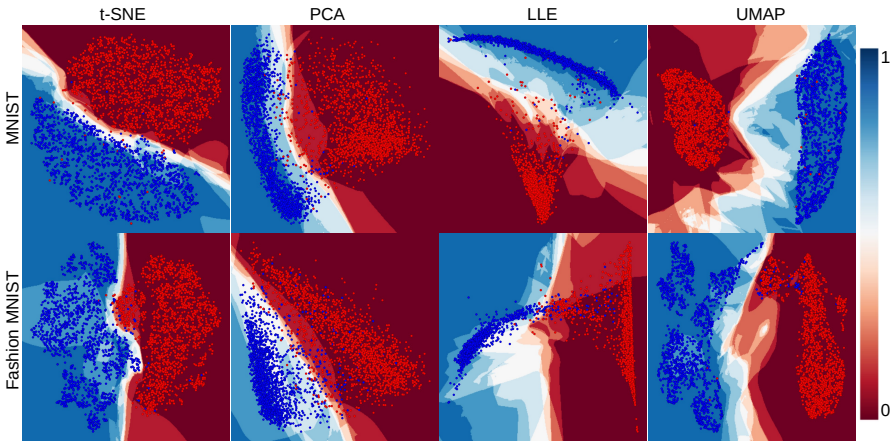


Figure 6.5: Classifier agreement map. Top row shows the result of classification of two digits in the MNIST data (digits 1 and 7), bottom shows classification of two objects in the MNIST-Fashion data (handbags and shirts). Color in these images denote agreement between 9 classifiers. Red represents agreements for class 1 and blue for class 2. More saturated colors indicate higher agreement. In between clusters where agreement is low, the colors tend to be desaturated (white).

how this concept is extended to highlight regions of lower classification agreement.

6.4.2.1 Example with MNIST and Fashion-MNIST

We demonstrate ensemble classification confidence by creating dense pixel maps to show the classifier agreement of two of the ten digits in the MNIST dataset (digits 1 and 7) and two of the objects in the Fashion-MNIST dataset (handbags and shirts). While there is nothing preventing the technique from being extended to multiclass classification, as in previous work [ERH+19, MHT18, REHT19], we limit ourselves to binary classification. In both cases, we begin by inverse-projecting each screen pixel to learn its position in the high-dimensional space. This high-dimensional point is then put through some number (greater than one) of classification methods. Since our dataset only contains two classes, each of the classifiers will simply assign a data point to one class or the other. We then color the pixel based on the number of classifiers that predicted each class. As shown in Figure 6.5, we color a pixel bright blue if the majority of classifiers predicted class one, and bright red if the majority of classifiers predicted class two. In between these two extremes, pixels are colored by decreasing the amount of saturation such that complete disagreement between

the models results in a white pixel – that is, half of the classifiers says the data point inverse-projected from the respective pixel belongs to class 1, while the other half says the point belongs to class 2.

The ensemble is formed by nine classifiers, namely Logistic regression, Linear SVM, SVM with radial basis function, K-Nearest Neighbors, Gaussian Process, Decision Tree, Random Forest, Adaboost, Gaussian Naive Bayes, and Quadratic Discriminant Analysis. These classifiers represent a diverse number of classification algorithms, including linear and non-linear methods. The output from the nine classifiers is used to generate the images in Figure 6.5, where not only can we see the class memberships of each point, we can also see the shape of the decision boundaries. For example, we can combine the use of inverse-projection for visualizing decision

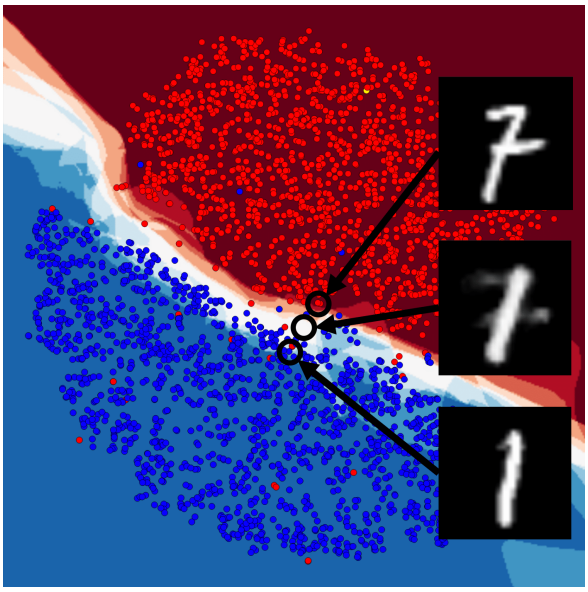


Figure 6.6: Visual inspection of points in the decision boundary as yielded by the ensemble classifier and drawn using back projection.

boundaries with its use for dynamic imputation, resulting in an interactive visual exploring system for understanding the uncertainty of the classifiers.

As an illustrative example, consider the differences between t-SNE and UMAP in Figure 6.5. When only considering the separability of the two clusters, one would likely assume that UMAP outperforms t-SNE, especially for the MNIST dataset (top row of Figure 6.5). However, when inspecting the decision boundaries, it becomes

less clear that the separability affects the classifiers' abilities to distinguish data points from one class to another. Specifically, in the t-SNE example, although the separation between the two clusters in the MNIST data is small, the boundaries are sharp and clean. Conversely, while UMAP produces high separation between the two clusters, there are disagreements between the classifiers in that space.

6.4.2.2 Implications to Visual Analytics

While there have been a number of proposed methods for illustrating the decision boundaries for classifiers of high-dimensional data [MWV15, SGH15, REHT19], our proposed use of inverse-projection offers an alternative that can be more flexible for visual analytics systems. As illustrated in Figure 6.6, the user can hover over areas with low model agreement (*e.g.*, pixels that are white or near white), and see what characteristics of the data might cause the classifier models to disagree.

In the context of designing new visual analytics techniques, the use of inverse-projection to help users better understand the behaviors of machine learning models can prove to be invaluable. Colloquially referred to as Explainable AI (or XAI), visualization researchers have been active in developing novel visualization and interaction techniques that can help a user understand, debug, and improve a complex machine learning model. While the space of XAI is large, we posit that the inverse-projection technique can contribute to this broad space of research.

6.4.3 Case Study 3: Gradient Map Visualization

Related to Explainable AI (XAI), one of the primary use cases for multidimensional projection is the visualization and interaction of data that exists in high-dimensional spaces that humans have difficulty interpreting. Unfortunately, a side effect of projection is the loss of information. To help mitigate the consequences of information loss imposed by projection, most techniques strive to maintain local relationships. In other words, they seek to preserve the relative distances between neighboring data points in the high-dimensional space in the two-dimensional projection. Of course, keeping these relationships intact after projecting is not always possible.

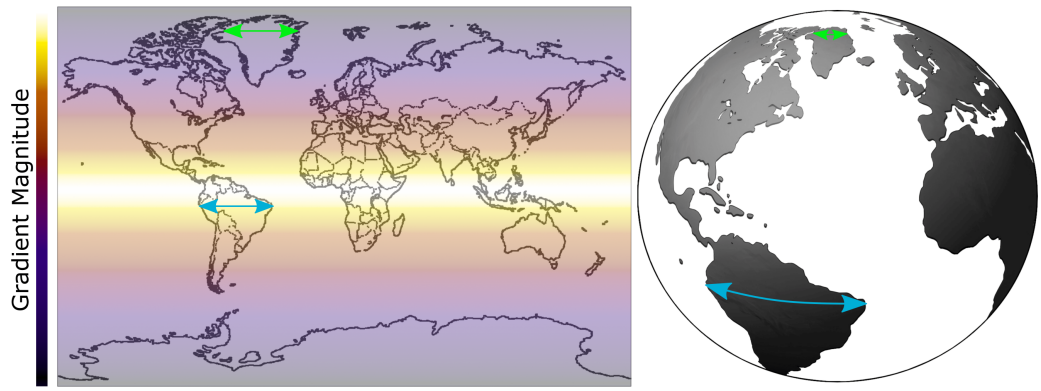


Figure 6.7: Two equal length line segments (green and blue) are placed on a Mercator projection of the Earth overlaid with its gradient image. After inverse projection, the lengths of these segments are dramatically different. The degree of change corresponds to the difference in the pseudo total derivative, shown by the gradient image overlay in the projection. Note that the green segment (in a region of low gradient) shrinks *vs* the blue segment (in a high gradient region around the equator). This figure is an illustrative example of how gradient images function (note that the figure does not reflect the actual gradients of the sphere).

Using the concepts of data imputation, a more holistic view of how a projection represents the spatial relationships between data points is presented. The ability to determine high-dimensional coordinates from a projected point in 2D enables a more complete investigation of the consequences of selecting a given projection technique by inspecting its *gradient image* (see Fig. 6.9). This image D is a 2D scalar field representing a pseudo total derivative of inverse projection function B computed using central differences as

$$D_x(\mathbf{y}) = \frac{B(\mathbf{y} + (w, 0)) - B(\mathbf{y} - (w, 0))}{2w}$$

$$D_y(\mathbf{y}) = \frac{B(\mathbf{y} + (0, h)) - B(\mathbf{y} - (0, h))}{2h}$$

$$D(\mathbf{y}) = \sqrt{\|D_x(\mathbf{y})\|^2 + \|D_y(\mathbf{y})\|^2}$$

where $\mathbf{y}$ is a point in the 2D projection space and w and h are a pixel's width and height, respectively. In summary, regions of a projection with large gradient values illustrate where the high-dimensional distance is changing most rapidly with respect to the low-dimensional distance. Figure 6.8 demonstrates how values on either side

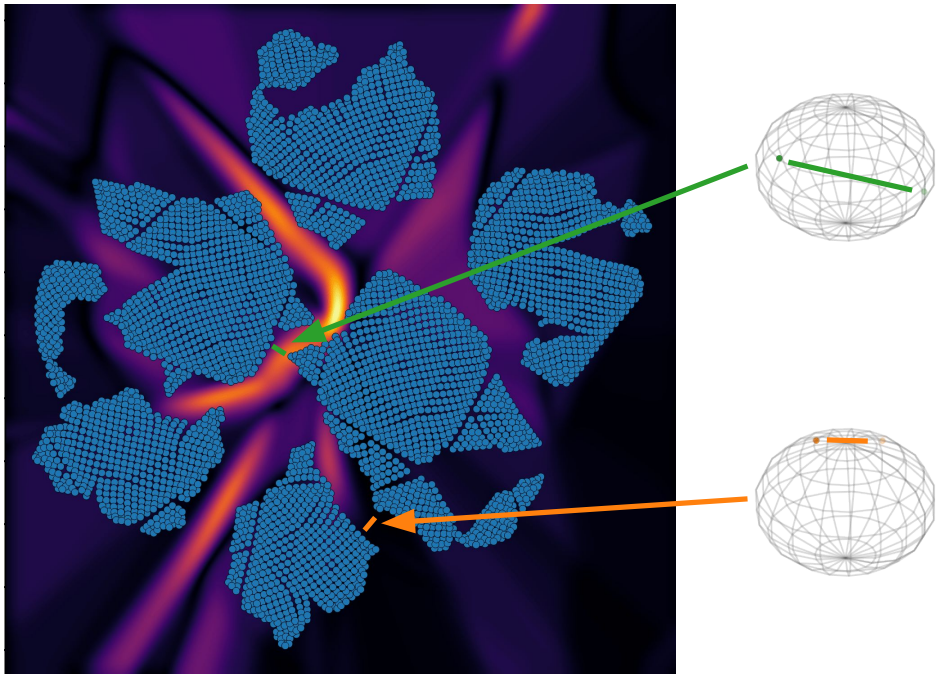


Figure 6.8: An illustration of how illuminated areas in the gradient map show areas where a small change in the low-dimensional space signifies a large change in the high-dimensional space. On the left, there is a 2-dimensional t-SNE projection of a 3-dimensional sphere, with a gradient map underlaid. Two similar distances are highlighted, one that crosses a highlighted area (green line), and one that crosses a dark area (orange line). On the right, two spheres are drawn, one for the green area, and one for the orange area. The dots on the spheres are the two closest points to the green and orange lines in the projection. The points closest to either end of the green line on the projection are much further away on the sphere than the points closest to either end of the orange line.

of large gradient values map to larger distances in the original data space, compared to values on either side of small gradient values. While the above method uses simple finite differences, any method for computing the gradient magnitude of B is appropriate.

6.4.3.1 Example with Sphere Data

Figure 6.7 shows how, under a standard projection for parameterization, even a simple three-dimensional sphere is transformed into a stretched and squeezed plane. Here, two equal length lines are placed on the parameterized plane at different locations. When each line segment is inverse-projected to recover the coordinates on the sphere, it is clear that the relative lengths of each segment have changed considerably. The

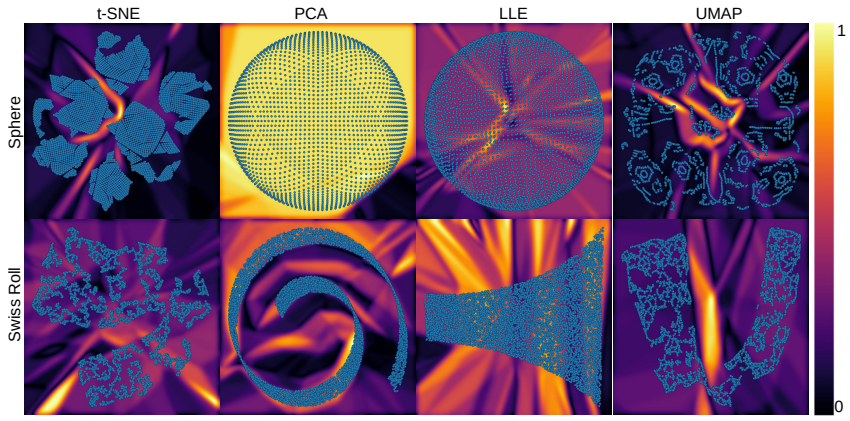


Figure 6.9: Gradient maps for 3D sphere dataset (top row) and 3D swissroll dataset (bottom row), showing the pseudo total derivative of the neighborhoods of all inverse-projected data points. Blue dots represent the points projected from three dimensions to two. The gradient at each pixel is determined by inverse-projecting each pixel's neighborhood and computing the gradient in the recovered high-dimensional space. Overall, the gradient maps show areas in the projections with high rates of change, suggesting peaks or valleys in the embedding space.

degree of change is more completely understood when it is observed in the context of the gradient map overlaid onto this plane. In this case, areas towards the poles of the globe intuitively have a gradient that approaches zero, while the equator will have the highest gradient. Thus, a line segment back projected from the two-dimensional plane will necessarily shrink; however, a similar line segment positioned near the equator will grow in length.

In the top row of Figure 6.9, a uniformly sampled three-dimensional sphere was projected to a two-dimensional plane using t-SNE, PCA, LLE, and UMAP. In these images, the sample points on the sphere are represented by blue dots, while the background is colored with the gradient image. In the cases of t-SNE, LLE, and UMAP, the projections maintain similar gradient characteristics with respect to neighboring points. However, there are some points that project to regions of high gradient. These regions are inevitable, as tears in the three-dimensional sphere are required in order to represent it on a plane. Conversely, PCA is a linear projection method that does not seek to preserve neighborhood information between data points. As a result, the gradient map under the data points is constant and reflects the planar nature of the projection space.

6.4.3.2 Implications to Visual Analytics

The gradient maps shown in Figure 6.9 illustrate the use of inverse-projection to help users see the quality of the projection. It is relevant to note that the gradient maps do not show the topology of an embedding space created using a projection function, which is the goal of works like Stress Maps [SSK10]. Instead, these gradient maps represent the *reconstructed* embedding space by the inverse-projection function. In some cases the inverse-projection function does not perfectly recover the original embedding space. For example, the top row of the PCA column in Figure 6.9 shows the reconstruction of a plane created by PCA in the 3D sphere dataset. Notice that the reconstructed surface is not perfectly linear as should be the case of PCA projections.

As such, we consider the gradient map as a debugging mechanism similar to tools in the XAI community for debugging machine learning models. In particular, the gradient map can help data scientists and visual analytics researchers to better understand the effect of projection and inverse-projection when visualizing high-dimensional data. For example, the top left image in Figure 6.9 shows the projection of a 3D sphere using t-SNE. The intense colors denote sharp discontinuities between the parts of the 3D sphere separated by t-SNE. This information has illustrative values and can be used to help a user better understand the behaviors of a projection function such as t-SNE.

6.5 Evaluation

We present an empirical evaluation of the inverse projection function described in the previous section. In the following, we split each dataset D into a training set D_s and a test set D_p. We train NNInv using the pair (D_s, P_s). Within Sections 6.5.1, 6.5.3, and 6.5.4 we restrict P to t-SNE, but also explore PCA, LLE, and UMAP in Section 6.5.2. We next evaluate the quality of NNInv using various error metrics computed using D_p and P_s. We next discuss our method in terms of scalability (Sec. 6.5.4), quantitative assessment of quality (Sec. 6.5.1), qualitative assessment of

quality (Sec. 6.5.2), and our novel inverse-projection error map (Sec. 6.5.3).

6.5.1 Quantitative Assessment of Quality

Besides being fast, we want an inverse-projection to be *accurate*. That is, given some ground truth pair ($\mathbf{x} \in \mathbb{R}^d, \mathbf{y} = P(\mathbf{x}) \in \mathbb{R}^2$), *unseen* during training, we want $B(\mathbf{y})$ to be as close as possible to $\mathbf{x}$. This follows the same idea as normalized stress metrics used to gauge the quality of projections in the literature [SVM14, vdMPvdH09] and also classical validation of inference models in machine learning. We measure quality in our case by computing the average inverse-projection mean square error $MSE = \|\mathbf{x} - B(P(\mathbf{x}))\|^2/|D_p|$ over the test set D_p. The closer MSE is to zero, the better B is. While we minimized MAE in our loss function, we report MSE here to enable easier comparison to earlier papers [dBD$^+$12, ABMC$^+$15].

Figure 6.10 shows the MSE for our three datasets, two projections (t-SNE and UMAP), and three tested inverse projections (iLAMP, RBF, and NNInv). We also consider several training-set sizes $|D_s|$ to show how MSE depends on the training amount. For Blobs, a relatively easy-to-project synthetic dataset, all methods have essentially zero error except RBF. MNIST and FashionMNIST show similar behavior: Our method achieves consistently one of the lowest errors. Errors are larger for these real-world complex datasets than for the synthetic Blobs, which is expected.

6.5.2 Qualitative Exploration

We explore NNInv's performance on two well-understood synthetic datasets, Sphere and Swiss Roll (Sec. 6.3.1). Simple datasets where the projections are well understood give us greater ability to reason about the inverse projection. In particular it is easier to understand how error is distributed across a dataset, as well as which projections will incur higher error. To illustrate this, we once again split the datasets into training (D_s) and test sets (D_p), this time having 75 and 25 percent of the total data, respectively. We then plot the projections of test portion (P_p) of these two datasets in Fig. 6.11 and color the points by the root-mean-squared error between the inverse-projections ($B(P_p)$) and the true high-dimensional data (D_p). Error

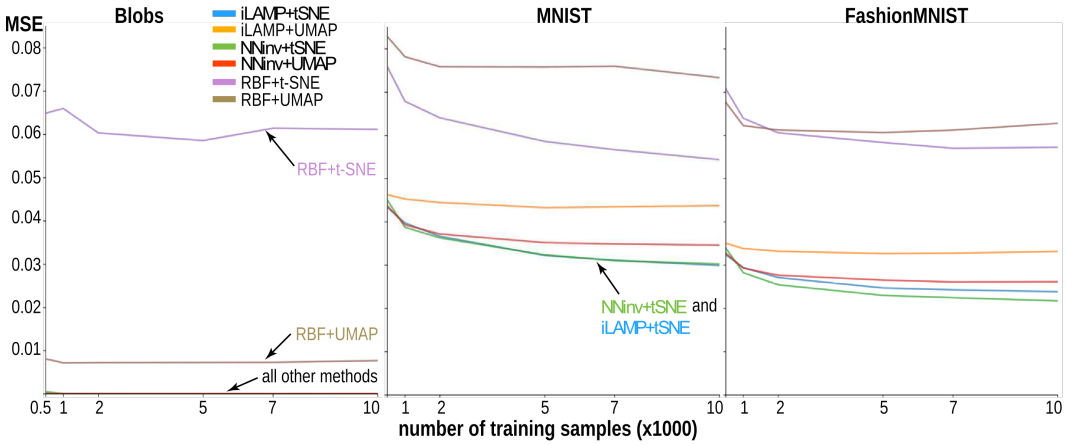


Figure 6.10: Mean square error of iLAMP, RBF, and NNInv inverse projections (from [ERH+19]). This graphic serves a dual purpose, illustrating how MSE depends on training-set size, and NNInv's ability to achieve one of the lowest errors amongst two other inverse-projection methods.

colormaps are normalized within each image, so that we can better see error variations *within* a given projection. Hence, colors cannot be compared across rows or columns of Fig. 6.11.

When analyzing inverse projection results, we must remember that the concept of error encompasses inaccuracies and faults in *both* the projection and the inverse-projection methods. For example, linear techniques like PCA will have a substantially different error profile compared to non-linear techniques such as t-SNE, LLE, or UMAP.

For the sphere, t-SNE and UMAP are able to peel away the surface, and error seems to congregate along the edges of the structures that make up the peel. In contrast, PCA and LLE end up with a slice out of the sphere causing the largest error in the center of their slices. For the Swiss Roll dataset, t-SNE, LLE, and UMAP are able to remove the swirl, with UMAP and t-SNE making similar ribboned shapes and LLE unraveling to an rectangle with perspective. In contrast, PCA keeps the general shape of the spiral, causing a speckling of high error throughout the whole structure.

Projecting high-dimensional space down to 2D is inherently lossy, and each method will project the data to 2D differently. This difference is not only visual –

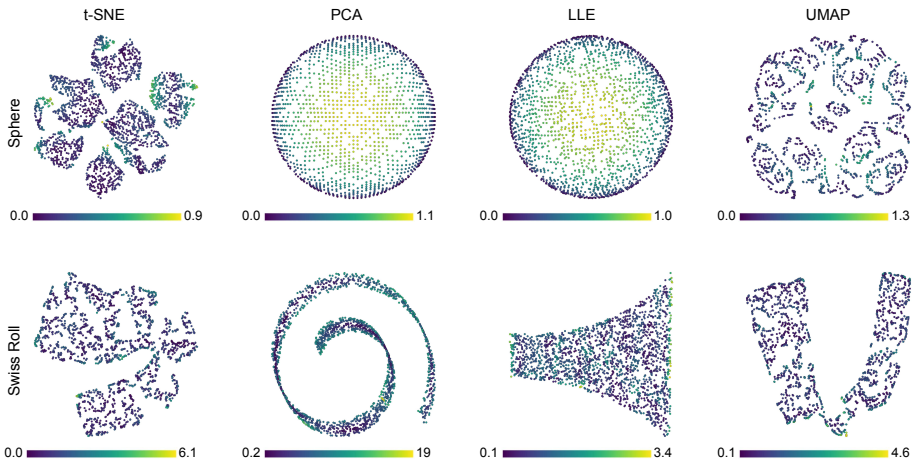


Figure 6.11: Validation maps showing inverse-projection error. Top row: Validation of the sphere dataset under four different projections. Bottom row: Validation of the swiss roll dataset under the same projections. Reconstruction error is computed by comparing the inverse-projected location with the original point's location. Each figure is scaled separately to highlight how the error distributed within that figure, so figures cannot be directly compared. The min and max error of each figure can be found below each plot on either end of a color legend.

each projection method emphasizes certain aspects of the data. As a result, different techniques throw away different portions and amounts of the high-dimensional data when performing the projection. This means that certain projection techniques will be easier to inverse-project than others.

For example, PCA does not aim to prevent overdrawing or projecting different points to the same two-dimensional location. As such, several data points can be projected to the same position in 2D space, making it impossible to correctly learn an inverse. In contrast, t-SNE and other non-linear techniques work to maintain local neighborhood relationships; when projecting a set of points, they try to preserve the relative distances in the projection that exist in the original space. In cases where there is poor preservation of inter-cluster distances, NNInv remains a valuable tool. If an area in the projection is shrunk or expanded relative to the high-dimensional space, the rate of change between inverse-projections will either increase or decrease respectively. When the interpolation moves very quickly, NNInv may be less useful for tasks like dynamic imputation (Sec. 6.4.1), but NNInv can help identify these spots with gradient maps (Sec. 6.4.3). The properties of each projection technique

inform and define the types of errors exhibited during the inverse-projection process.

As Fig. 6.11 shows, one consequence of PCA projecting multiple distant points to a small region on a 2D plane is that the inverse-projected points will likely be erroneous. In this case, the error increases as the distance in the high-dimensional space increases between points co-located on the 2D projection. Conversely, for t-SNE and UMAP, the non-linear projections distort the input geometry, often into shapes that no longer resemble the topology of the original data. In return, the inverse-projected data points from 2D back to the high-dimensional space are much closer to their original positions, resulting in significantly smaller total error. In other words, better grouping by similarity as well as better separation of points will make inverse-projection easier.

6.5.3 Dense map of inverse projection error

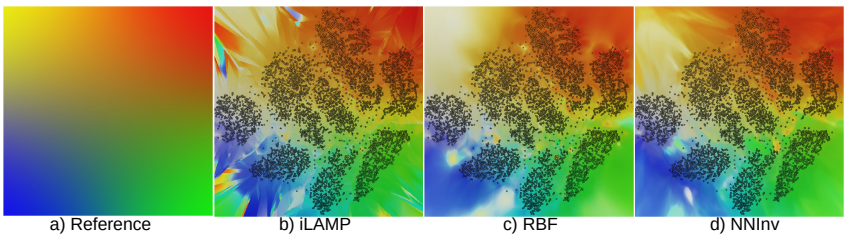


Figure 6.12: Differences of "round-trip errors" with three inverse techniques (iLAMP, RBF, and NNinv). See Sec. 6.5.3 for an in-depth discussion.

Evaluation of inverse-projection methods often uses error metrics defined for direct projections such as stress or reprojection error [dBD$^+$12, ABMC$^+$15]. However, the above metrics only gauge the error at the locations of projection points $P(D)$. The same is actually the case for all errors for direct projections we are aware of – they only gauge how good a (direct) projection is at the locations of the scatterplot points. As explained earlier in Sec. 2.4.0.1, the key use-case of inverse projections is the *out-of-sample* one, where one inversely projects *different* points than $P(D)$.

We next propose a validation approach that considers the out-of-sample case, *i.e.*, evaluates the quality of B at all points in $\mathbb{R}^2$. We proceed as follows. Given

a dataset D, we construct $P(D)$ as usual given a user-chosen projection technique P, and use $(D, P(D))$ to train our inverse projection B. Next, we discretize the projection space $\mathbb{R}^2$ using a pixel grid with a given resolution R, in our case $R = 400$. Then, for every pixel $\mathbf{y}$, we compute the pixel $\mathbf{y}' = P(B(\mathbf{y}))$ given by the "round trip" of back projecting it to $\mathbb{R}^d$ and next projecting it again to $\mathbb{R}^2$. To perform this, we must assume that P is *parametric*. Then, ideally, $\mathbf{y} = \mathbf{y}'$ for all pixels $\mathbf{y}$. This way, we can assess an inverse projection error also for points in $\mathbb{R}^2$ which do not correspond to projections of points in our given dataset D.

We visualize the round-trip errors as a dense map as follows. We create a hue image by bilinear interpolation of four different hues (Fig. 6.12a). Next, we color every pixel $\mathbf{y}$ by the hue of the round-trip pixel $\mathbf{y}'$ and set its luminance to $\|\mathbf{y} - \mathbf{y}'\|$. Dense map areas which show the same color gradient as Fig. 6.12a have, thus, low inverse-projection errors. Bright areas and/or hue differences from this gradient show large projection errors. Scatterplot points are colored in the same way, but use a slightly lower brightness value to avoid confusion with the map pixels. Figures 6.12b-d show the error maps for iLAMP, RBF, and NNInv for the inverse projection of the MNIST dataset projected by t-SNE. We see that NNInv creates a color gradient which is close to the reference one, has minimal discontinuities, and has few bright spots. Hence, NNInv can inverse-project the entire 2D space without introducing large amounts of error.

6.5.4 Scalability in Training and Inference

Scalability implies the effort required to *train* our method and, separately, the effort needed to *infer* $B(T)$ as function of the size of the dataset Y to inversely project. Concerning training, Table 6.3 shows the number of training epochs needed to obtain convergence (defined as in Sec. 6.3.2) as function of the training set size $|D_s|$, for all three considered datasets and P = t-SNE. Columns 2..4 indicate averages for multiple runs created by randomly sampling D_s from the entire dataset D. Overall, we obtain convergence for roughly 150 epochs for all datasets and training-set sizes.

Figure 6.13 shows the inference speed for all three datasets. Speed does not

Training set size $\lvert D_s \rvert$	Average # epochs for each dataset D			Row averages
	Blobs	Fashion-MNIST	MNIST	
500	268.0	214.0	213.5	192.5
1000	190.5	129.0	147.5	149.0
2000	153.0	112.0	111.0	112.5
5000	103.0	120.5	138.0	127.5
7000	127.0	118.5	151.0	144.0
10000	82.0	124.5	142.5	146.5
column avg	153.9	136.4	150.6	145.3

Table 6.3: Training effort until convergence.

depend on the projection method P – once NNInv is trained, its performance is linear in the number of inversely-projected samples. When computing inference speed, we inversely project *any* point in $\mathbb{R}^2$ and not just points in $P(D)$. Indeed, for assessing speed, we do not need ground-truth information. Moreover, in real use cases, one would inversely project *unseen* data, for which such ground-truth information is not available. We see that both RBF and iLAMP have a superlinear behavior, while NNInv (our method) is basically linear. NNInv is roughly one magnitude order faster than RBF and nearly two magnitude orders faster than iLAMP for 40K samples or more. This speed-up is crucial for applications that need to inversely project hundreds of thousands of samples (or more), like in the construction of dense maps [MHT18, ERH+19] and the maps in Sec. 6.4.2 and 6.4.3. NNInv constructs such maps in *seconds*, while iLAMP and RBF need (tens of) minutes, making human-in-the-loop usage of such methods impossible in visual analytics scenarios – one of the key reasons why dense maps are built in the first place. This scalability is one of the most important advantages of NNInv.

All experiments were run on a 4-core Intel E3-1240 v6 at 3.7 GHz with 64 GB RAM and an NVidia GeForce GTX 1070 GPU with 8 GB VRAM, and the code was implemented in Python 3 using the Keras library (keras.io).

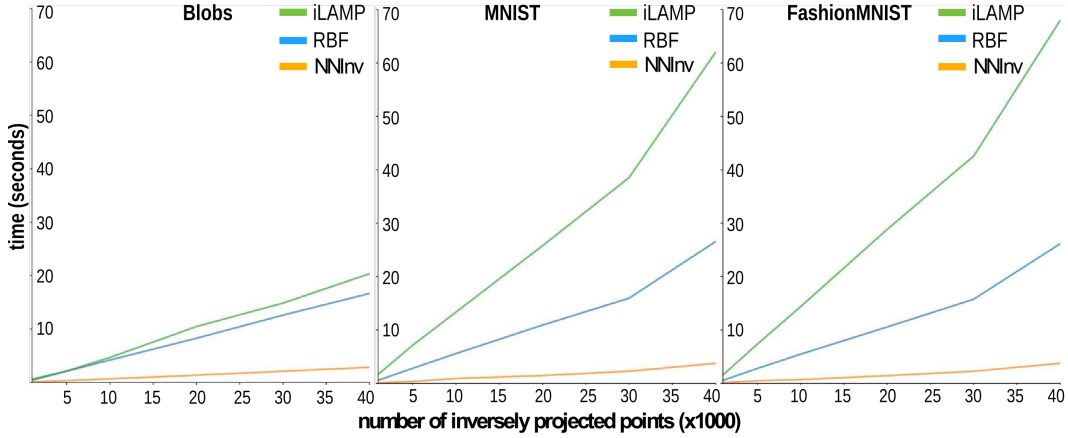


Figure 6.13: Inverse-projection speed *vs* number of samples [ERH+19]. Inverse-projection speed is of great importance for applications that require the inverse-projection of many samples, as is the case for the aforementioned classifier and gradient maps (Secs. 6.4.2 and 6.4.3).

6.6 Limitations

NNInv is scalable, accurate, and relatively smooth, as shown in Sec. 6.5. Yet, using a neural network does have its disadvantages [Ben13]. A neural network (1) requires a particular threshold of good quality training examples, (2) can be computationally expensive to train, and (3) can be generally hard to interpret. In Sec. 6.5.1 we show acceptable mean squared error with as few as 500 training examples, and caution that below that threshold, our technique will not perform as successfully. In all of the examples in this chapter, the projections NNInv are trained on are *good quality* projections, obtained by choosing reasonable values for the projection's hyperparameters. Good quality projections are generally more likely to have the qualities (as described in Sec. 6.5.2) required for accurate inverse-projection. While NNInv is useful in helping to interpret projections (*e.g.*, Fig. 6.2), it can be difficult to reason about NNInv itself, since neural networks are hard to interpret in general. That is, our metrics show that NNInv performs better as it can approximate nonlinear patterns, but it is not obvious how NNInv does this. We leave the explainability of NNInv's improved performance to future work.

6.7 Discussion and Future Work

Future inverse-projection research can take several interesting directions. Of particular relevance is the discussion in Sec. 6.5.2 regarding the properties of projection techniques and their inverses.

As Sec. 6.5.2 shows, when discussing the invertibility of projection functions, we find that not all projection methods are equally suitable for the inverse-projection method: PCA is worse than t-SNE or UMAP because multiple data points can be projected into the same 2D pixel.

Interestingly, a type of projection that is designed specifically for its invertibility is the encoder portion of an autoencoder. When trained together with the decoder, the entire process optimizes for the recoverability of data points from input to output. Yet, a user would have a hard time understanding the embedding of a regular encoder because there is no intentionally designed structure in the embedding space created with an encoder. Also, there are no guarantees about neighborhood preservation or relative distance preservation.

The tradeoff between understandability of the latent space created by a projection and the appropriateness of the projection for learning its inverse is interesting. On one hand, a projection technique may sacrifice some information to create a more insightful, or more spatially intuitive, visualization. Yet, the use of inverse-projection can lead to novel visualization and interaction techniques that can better help the user explore and understand a high-dimensional space. Further steps should be taken to find a happy medium between these two extremes, whether that be autoencoders with some cost for occlusion, or spacing items too far apart, or a projection technique with a greater loss for discarding information.

Accessible and fast inverse projections will have far-reaching impacts on visual analytics (VA) systems that use projections. We believe that a deep learning approach to inverse projection is especially accessible given today's robust ecosystem for neural network development [C$^+$15, OBL$^+$19, AAB$^+$15, PGM$^+$19]. We hope that future works along this line of research continue to leverage approachable

methods and libraries that ease adoption for tool builders. The most potential use-case is hypothesis generation made possible by dynamic imputation (Sec. 6.4.1), but several different augmentations exist, *e.g.*, adding extra information showing how models understand the data space in the same vein as classifier agreement maps (Sec. 6.4.2), or helping projection users to better understand the underlying structure as in gradient maps (Sec. 6.4.3). We are particularly interested in how combinations of these techniques, as the hypothesis generation paired with gradient map style backgrounds, can help users who are less familiar with projection techniques make sense of overview projections in VA applications.

Lastly, we believe there are several applications of this technique that should be explored further. Projections and inverse-projections can be used to explore the space of different 2D charts that have themselves been projected to 2D (in a manner similar to ChartSeer [ZFF20]), and data that is often modeled on graphs, such as molecular data.

6.8 Conclusion

In this chapter, we presented NNInv, a deep learning approach to learning the inverse of projection functions. Similar to existing works such as iLAMP and RBF, NNInv is agnostic of the projection used, *i.e.*, it can learn to invert any projection algorithm (such as PCA, t-SNE, UMAP, LLE, etc.). NNInv uses a trained neural network to learn the approximate mapping from a given 2D scatterplot produced by a projection algorithm to the corresponding high-dimensional data. We find that NNInv can be more accurate than iLAMP and RBF on both synthetic and real-world datasets, and is more scalable to large datasets: Once trained, NNInv can perform inferencing within less than 10 milliseconds when running in a browser on a laptop, which makes NNInv a more suitable technique than iLAMP and RBF for interactive visualizations. Lastly, we show the potential of NNInv for analysis tasks such as hypothesis generation, classifier agreement, and gradient visualization These are three areas are important to helping users understand complex high-dimensional

data and serve as evidence to the possibility of the broad applicability of NNInv in high-dimensional data exploration and analysis.

Chapter 7

DimBridge

In this chapter, we detail DimBridge, a system that leverages predicate logic to bridge the projection and the data spaces. The system allows users to highlight perceived patterns in the projection space. DimBridge then identifies the relevant data dimensions and a relevant interval of values for each dimension to concisely explain the user-selected visual pattern across the sequence of selections. The resulting subspace, defined by the selected data dimensions and intervals, is then visualized using a scatterplot matrix (SPLOM). This simple workflow produces profound visualization that helps users explore and make sense of the patterns in a projection.

7.1 Problem Statement

The demand for interactive visual exploration techniques for high-dimensional data has grown significantly, given the substantial increase in the generation of such data over the last decade [LMW+17]. Among these techniques, dimensionality reduction (DR) stands out as one of the most prominent methods for visualizing high-dimensional data. Well-known DR techniques such as MDS [KW78], PCA [Pea01], t-SNE [vdMH08], and UMAP [MHSG18] are capable of generating low-dimensional projections of data, where the spatial proximity of points encodes a measure of similarity between data items [NA19, EMK+21]. These techniques offer 2D repre-

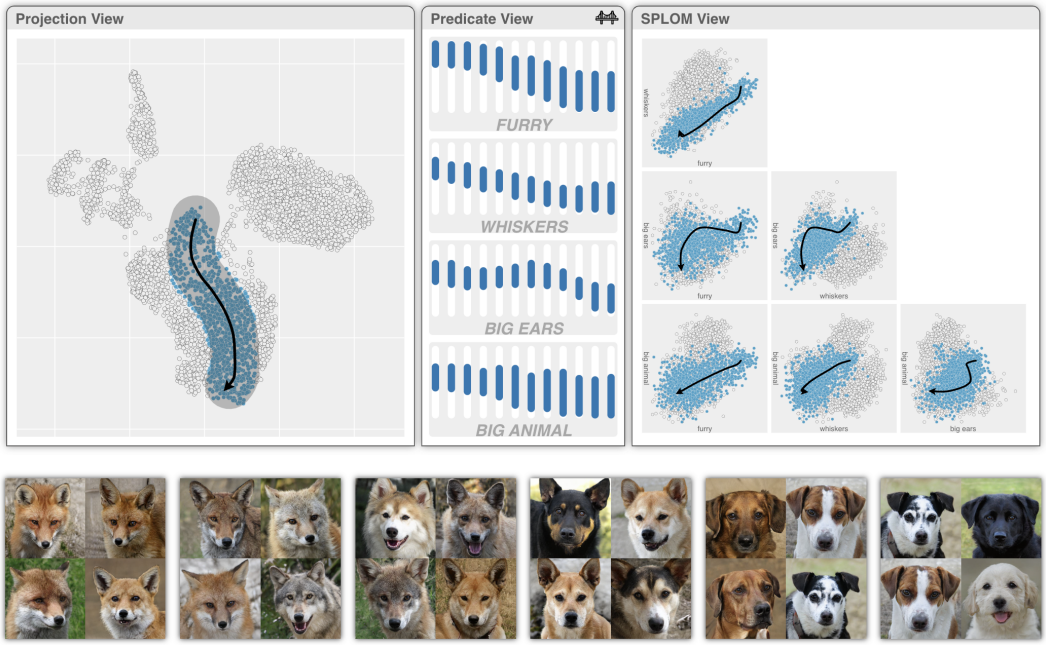


Figure 7.1: DimBridge is a system that helps users understand visual patterns in dimensionality reduction-based 2D projections. Within an interface, users can brush perceived patterns in a projection (left), and DimBridge computes *predicates* in response (middle) – compact explanations of the user's selection expressed in terms of the data space. The data dimensions that compose a predicate help place the data in a context well-understood by users, shown as a scatterplot matrix (right).

sentations of high-dimensional data, facilitating users in identifying data patterns in the high-dimensional space more readily.

However, despite the utility of DR methods in understanding *what* data items are similar, few provide insight into *why* the data are similar. As a result, users often need to rely on patterns that they perceive in the projection, e.g., clusters of points, spatial density, or distinctive shapes, to infer characteristics of the original high-dimensional data [SG17, BSIM14, SMT13]. While some of these patterns may indeed reflect underlying data characteristics, they can also be influenced by noise, artifacts stemming from the DR process [MCMT14, NA19, KB19], or even the propensity of humans to see visual patterns where none exist [WVJ16, Ell18, WFG$^+$12]. Modern visualization tools and interaction techniques offer some assistance in understanding projections, such as helping users discern DR distortions [Aup07, LA11, SDMT15], examining or contrasting perceived clusters [LWCC18, MJEG21, EHA$^+$23], or gen-

erating new plausible data points in the projection [dBD+12, ABMC+15, EAS+23].

In this chapter, we present DimBridge, building upon this existing work to contribute a new way to explore and understand perceived visual patterns in a DR projection. The workflow for DimBridge is designed to be as simple as possible. As shown in Figure 7.1, a user interactively highlights a pattern in the projection space and DimBridge (1) identifies a subset of data dimensions and (2) an interval of values for each dimension that concisely explain the user-selected visual pattern. The resulting subspace, defined by the selected data dimensions and intervals, is then visualized using a scatterplot matrix (SPLOM). This visualization enables a user to observe the selected visual pattern in a reduced context of the original high-dimensional data space – a version of the task known as *"Mapping Synthesized [Data] to Original Dimensions"* in DR task taxonomies [BSIM14, NA19]. By connecting the observed visual patterns in the projection space to the original data space, DimBridge helps users understand the meaning of a visual pattern in a familiar context.

DimBridge supports several user interactions (select, select and contrast, select and draw) for identifying visual patterns in a projection, including clusters, outliers, and shapes. Once a pattern is selected, DimBridge uses specially designed predicate induction algorithms to learn the subset of data dimensions and an interval for each dimension that most effectively explains the visual pattern. DimBridge is equipped with two predicate induction algorithms, a novel Predicate Regression algorithm tailored for speed and scalability, complemented by the PIXAL algorithm for accuracy [MBBC22]. Both algorithms allow for a smoothness constraint, which enables DimBridge to support a novel select and draw interaction. Within this framework, predicates serve as a conceptual bridge connecting the low-dimensional (i.e., the projection space) and the high-dimensional (i.e., the original data space) space.

To demonstrate DimBridge's efficacy, we showcase the system on numerous datasets, demonstrate its utility within a case study, and iteratively evaluate its functionality with domain experts. With these examples, we also hope to demonstrate that DimBridge provides a means for users to explain projections using the original

data space, solving key challenges in interpretability associated with conventional projection-based high-dimensional data visualization.

In summary, our work makes the following contributions:

- We introduce DimBridge, a system that uses predicate logic to *bridge* the projection and the data spaces, helping users to make sense of patterns in 2D projections of high-dimensional data.
- We introduce a new interaction design for selecting visual patterns within DR projections and a new algorithm for generating predicates with smoothness constraints given a selected visual pattern (Predicate Regression).
- We evaluate DimBridge through several showcases with different domain applications, a case study, and evaluations with researchers from materials science and pharmaceutical drug discovery.

7.2 Interpreting DR Results: Design Goal and Tasks

Patterns identified in conventional DR visualizations are challenging to interpret because they lose the underlying semantics of the original dimensions, such as the context for why points are considered similar. Our work addresses these challenges by bridging the projection space and the underlying data space. Specifically, DimBridge facilitates the interpretation of patterns found in DR results through interactive querying of the projection, enabling retrieval and visualization of relevant subspaces of the original dimensions.

We break down the process of identifying and interpreting patterns into three high-level tasks. These tasks are the foundational requirements used to develop the DimBridge system.

(T1) Identify and query visual patterns in a projection

Dimensionality reduction (DR) transforms high-dimensional data into low-dimensional forms to facilitate visualization and analysis while preserving important structures

and relationships. A common approach is to reduce the data to two dimensions and visualize the resulting patterns using a scatterplot. DimBridge should support the user in identifying potential patterns of interest and facilitate direct engagement with the projection, enabling users to explore by selecting patterns that capture their interest. There are multiple patterns that a user can identify within a projection that have unique approaches to selection and interaction. DimBridge must allow the user to identify and select each of the following patterns:

Clusters: Identifying clusters is a key application of DR projections [SZS+16], as it allows a way to classify similar and dissimilar points based on attributes. Although it can be intuitive to identify distinct groups of data points, it is not directly clear why they formed or how they differ by visualizing the projection alone.

Outliers: Identifying outliers is closely linked to recognizing clusters, yet determining if outliers in a 2D projection accurately reflect those in the high-dimensional data is challenging based solely on the projection. While outliers may be identified based on distance in the projection, it is not clear how this distance translates to the original dimensions.

Spatial Density: Identifying clusters is not always straightforward, especially with varied point densities [HE11, SHGF16]. Unlike clear-cut clusters, density across a projection can change gradually, complicating the task of selecting a group of points for detailed analysis. For example, points just outside a selected dense area might still belong to that group, reflecting the challenge of making precise selections.

Shapes: DR projections are also used to uncover hidden low-dimensional structures in data [SZS+16], such as identifying a low-dimensional manifold that explains the data distribution. These structures often appear as specific shapes within the projection, such as a curve slicing through a group of points.

(T2) Retrieve and visualize relevant subspaces of the original data

Understanding visual patterns in the DR results requires explaining them in the context of the data's original dimensions. Traditional visualization techniques,

however, struggle with handling high-dimensional data effectively. Given a visual pattern in the DR results queried by the users, DimBridge must be able to retrieve relevant subspaces of the original data. For this subspace to be visualized effectively, it must have a relatively small number of dimensions.

(T3) Evaluate the visual pattern in context

While the goal is to explain a visual pattern in the original dimensions, distortions introduced by the DR algorithm can result in patterns in the DR results that do not correspond to patterns in the original dimensions. Beyond simply retrieving relevant subspaces, DimBridge must help users evaluate a selected pattern given a retrieved subspace.

7.3 Example Use Case and System Overview

We provide a simple use case of DimBridge to illustrate how it addresses the design goals. Figure 7.1 shows the example of a user with a set of animal images where each image is labeled with 14 numeric attributes (e.g., furry, whiskers, etc.). After performing dimensionality reduction using UMAP and visualizing the results, the user identifies a cluster of data points that form an unusual curved shape in the *Projection View*.

The user investigates this pattern by selecting the points at the top of the cluster by drawing a bounding box, and then dragging the selected area to the bottom, following the cluster's curved shape (T1). Given the selection, DimBridge identifies a set of (1) dimensions and (2) intervals for each dimension that most concisely explain the visual pattern (T2). These dimensions are visualized using a scatterplot matrix in the *SPLOM View*, with the corresponding intervals shown for each dimension in the *Predicate View* (T3). With these visualizations, the user can make the following observations:

- The selected shape can be explained using just 4 of the original 14 dimensions.
- The shape most closely correlates with the "furry" and the "whiskers" dimensions.

- The top of the shape represents very furry animals with whiskers and big ears. Towards the bottom of the shape, the animals become larger, are less furry, and have small ears and no whiskers.
- There is a downward shift in the intervals for "big ears" where the shape curves to the right, corresponding to a tipping point where the animals' ears become increasingly smaller.
- Confirming these observations, the user can see that data sampled along the shape represent animals from foxes to wolves, wolf-like dogs, large dogs, and finally smaller puppies.

To support this workflow, DimBridge models the user's interaction as a set of continuous brushes. In the example shown in Figure 7.1, the shape drawn by the user is discretized into 12 brushes, shown as 12 intervals for each dimension in the Predicate View. To help the user make sense of this selection, DimBridge uses a novel predicate induction algorithm to identify a set of dimensions and intervals that best explain the brushes. The algorithm enforces a **smoothness constraint**, such that the dimensions and the intervals in each of the predicates are consistent with the others.

In section 7.4 we describe our predicate induction engine. Section 7.5 discusses the design of the visualization interface and interaction techniques. We demonstrate the utility of DimBridge with 3 examples and 2 use cases with collaborators in drug discovery and material science domains.

7.4 Predicate Induction Engine

The central motivation of this work is to support users' interpretion of patterns found in the DR results in the context of the original data dimensions. The predicate induction engine facilitates this by generating predicates that best explain, in the original dimensions, a visual pattern identified by the user.

7.4.1 Bridging DR and data patterns with predicates

DimBridge uses first-order predicate logic as a "bridge" between patterns in the DR results and relevant subspaces of the original dimensions. This allows users to capitalize on the strengths of both spaces: patterns can be identified visually in the DR results, and their semantics can be recovered in the original dimensions.

A first-order predicate, Φ, is defined as a conjunction of one or more clauses, each consisting of a data dimension and a minimum and maximum value. The clause ϕ_j is defined for the j-th dimension, with minimum and maximum values denoted ϕ_j^{min} and ϕ_j^{max}. A predicate contains all data points with values that fall within the ranges defined by each clause. Alternatively, a predicate can be thought of as a function that takes a data point, x_i, as input and outputs a binary label: 1 if the data point falls within the value ranges defined by each clause, and 0 otherwise:

$$\Phi(x_i) = \mathbb{1}[\phi_j^{min} \leq x_{ij} \leq \phi_j^{max},\ \forall \phi_j \in \Phi] \tag{7.1}$$

7.4.2 Generating predicates from user interactions

The predicate induction engine enables DimBridge to leverage predicates as a bridge between patterns in the DR results and original dimensions by generating predicates that closely match the users selection. First, we derive a set of background points, B, and pattern points, P, from a users interaction. Second, we define the combined dataset $X = B \cup P$ and associated binary labels, $Y = \{\mathbb{1}[x_i \in P] | x_i \in X\}$. Finally, we use a predicate induction algorithm to identify predicates that contain a set of data points, represented by the binary labels $\Phi(X) = \{\Phi(x_i) | x_i \in X\}$, that closely match the data points selected by the user, represented by Y.

In our implementation of DimBridge, we consider two predicate induction algorithms. The first is the recursive predicate induction (RPI) algorithm used by the PIXAL system [MBBC22]. This algorithm constructs predicates from the ground up, starting with a set of single clause predicates that are iteratively refined.

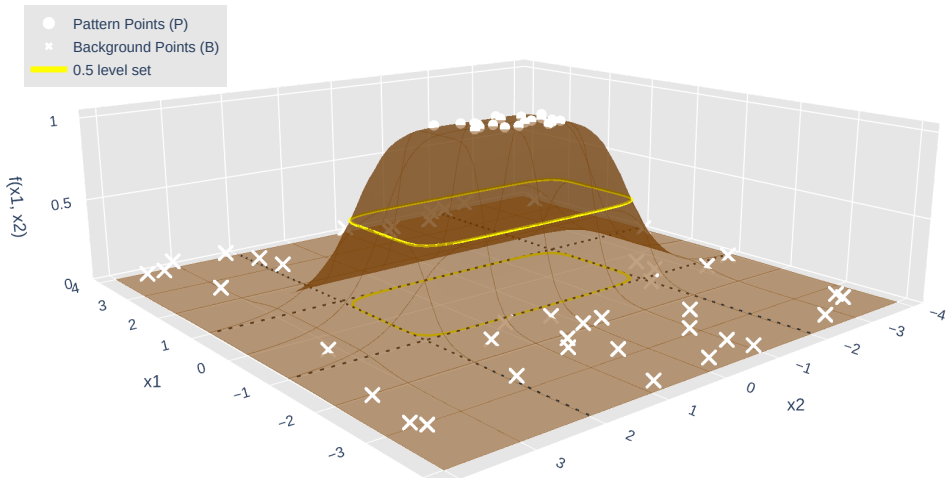


Figure 7.2: Illustration of differentiable proxy function (Equation 7.5) centered at $\boldsymbol{\mu} = (0, 0)$ with $\mathbf{a} = (1, 1/2)$ and $b = 7$.

A predicate is scored at each step, and continuously refined until the score stops improving. For DimBridge, we score predicates using the F1 score calculated between $\Phi(X)$ and Y, balancing false positives (points that are in the predicate but not the user's selection) and false negatives (points that are in the user's selection but not the predicate). This results in multiple, overlapping predicates, each representing a candidate subspace. While this may be suitable for offline applications, this is too slow for an interactive system.

The second is a novel algorithm, Predicate Regression that works by approximating high dimensional bounding cuboids via a differentiable proxy function.

7.4.2.1 Predicate Regression

Rather than generating and evaluating predicates iteratively, the Predicate Regression algorithm defines a differentiable loss function based on a reparameterization of first-order predicates, which it then minimizes to find a single "best" predicate given the X and Y defined by the user.

Note that when a predicate clause fully cover the data extent along certain dimension, the clause always return true. Therefore, even though a predicate defines a subspace that only includes dimensions in the clause, we can expand it to all dimensions, with the range being strictly greater than the data extent:

$$\Phi(x_i) = \mathbb{1}[\phi_j^{min} \leq x_{ij} \leq \phi_j^{max},\ \forall j = 1 \dots M] \tag{7.2}$$

with $\phi_j^{max} \geq \max_i\{x_{ij}\}$ and $\phi_j^{min} \leq \min_i\{x_{ij}\}$ when $\phi_j \notin \Phi$.

This expansion gives us opportunity to optimize all predicate clauses simultaneously. Moreover, we can define a predicate with the parameters $\boldsymbol{\mu}$ and $\mathbf{r}$, denoting a midpoint and range for each dimension (j) respectively:

$$\mu_j = \frac{\phi_j^{max} + \phi_j^{min}}{2},\ \ r_j = \frac{\phi_j^{max} - \phi_j^{min}}{2} \tag{7.3}$$

This reparameterization allows us to consider whether a data point is contained by a predicate not only as a binary label, but a continuous probability:

$$Pr(\Phi(x_i) = 1|\mathbf{r}, \boldsymbol{\mu}, b) := \frac{1}{1 + \sum_{j=1}^{M} |\frac{1}{r_j} \cdot (x_{ij} - \mu_j)|^b} \tag{7.4}$$

Generating a Predicate for a Single Brush: Geometrically, the probability gives a rounded bump function (see Figure 7.2) of x_i centering at $\boldsymbol{\mu}$, where b is a fixed parameter controlling the steepness of the bump. Moreover, the 0.5 level set is enclosed by the predicate bounding box, $\prod_{j=1}^{M}[\mu_j - r_j, \mu_j + r_j]$, along each data feature dimension j. To find the optimal predicate via optimizing the parameters, we rewrite $1/r_j$ as a_j and view the probability as a differentiable function of a_j and μ_j:

$$f(x_i|\mathbf{a}, \boldsymbol{\mu}, b) = \frac{1}{1 + \sum_{j=1}^{M} |a_j \cdot (x_{ij} - \mu_j)|^b} \tag{7.5}$$

Given an X and Y defined by a user's selection, the loss function, binary cross entropy (BCE), is defined as:

$$\mathcal{L}_{bce}(\mathbf{a}, \boldsymbol{\mu}|X, Y) = \frac{1}{N}\sum_{i=1}^{N} y_i \log(f(x_i)) + (1 - y_i)\log(1 - f(x_i))$$

Recall that a_j is the inverse of the range r_j, and enlarging the range beyond data extent (i.e. forcing $a_j \to 0$) effectively eliminates the corresponding clause in the predicate conjunction. Therefore, selection of features in the predicate can be achieved through an L_1 regularization, $||\mathbf{a}||_1$. Eventually, minimizing loss functions over $\mathbf{a}$ and $\boldsymbol{\mu}$ gives a predicate in the form $\prod_{j=1}^{M}[\mu_j^* - 1/a_j^*, \mu_j^* + 1/a_j^*]$, where

$$\mathbf{a}^*, \boldsymbol{\mu}^* = \underset{\mathbf{a},\boldsymbol{\mu}}{\operatorname{argmin}}\, \mathcal{L}_{bce}(\mathbf{a}, \boldsymbol{\mu}) + \gamma_1 \cdot ||\mathbf{a}||_1 \tag{7.6}$$

and γ_1 controls the strength of the predicate sparsity.

Generating Predicates for Multiple Continuous Brushes: One requirement of a predicate induction algorithm, when it comes to the interactive nature of DimBridge, is the consistency and smoothness of results it gives from consecutive interactions. For example, one may expect smoothly changing clauses when fine-tuning queries, or brushing curves over a region. For these cases, a constraint can be added to the objective function that encourages smoothness in the resulting predicates.

The draw interaction results in a discretized sequence of selections derived from the user's gesture. Given a sequence X_t, Y_t for $t = 1...T$, optimizing

$$\sum_{t=1}^{T} \mathcal{L}_{bce}(\mathbf{a_t}, \boldsymbol{\mu_t}|X_t, Y_t) \tag{7.7}$$

gives a sequence of predicates independent to one another. To encourage consistency and continuity between consecutive predicates, a smoothness loss function is

introduced:

$$\mathcal{L}_{smooth} = \sum_{t=2}^{T} \gamma_a \cdot ||\mathbf{a_t} - \mathbf{a_{t-1}}||^2 + \gamma_\mu \cdot ||\boldsymbol{\mu_t} - \boldsymbol{\mu_{t-1}}||^2 \tag{7.8}$$

where γ_a and γ_μ control the strength of consistency and continuity between consecutive predicates.

In entirety, the loss function becomes

$$\mathcal{L} = \sum_{t=1}^{T} \mathcal{L}_{bce}(\mathbf{a_t}, \boldsymbol{\mu_t} | X_t, Y_t) + \sum_{t=1}^{T} \gamma_1 \cdot ||\mathbf{a_t}||_1 \tag{7.9}$$

$$+ \sum_{t=2}^{T} \gamma_a \cdot ||\mathbf{a_t} - \mathbf{a_{t-1}}||^2 + \sum_{t=2}^{T} \gamma_\mu \cdot ||\boldsymbol{\mu_t} - \boldsymbol{\mu_{t-1}}||^2 \tag{7.10}$$

7.5 Visual Interface

The DimBridge interface consists of three coordinated views: The *Projection View*, the *Predicate View*, and the *SPLOM View*. By interacting with these views, users can query patterns in the DR results, view and modify predicates, visualize relevant subspaces of the original dimensions, and evaluate the queried pattern in context.

Users can employ the Projection View to visualize and interact with patterns through a scatterplot displaying the DR results. The Predicate View allows users to view and adjust predicates generated by the Predicate Induction Engine in response to these interactions. Finally, the SPLOM View allows users to visualize the data in a subspace of the original dimensions defined by these predicates.

Predicates are generated with the goal of matching a user's selection as closely as possible. However, distortions introduced by the DR algorithm will results in some data points that are in the user's selection not being included in the predicate, and some data points in the predicate not being in the user's selection. These mismatches are illustrated using color in the Projection and SPLOM Views. Similar to prior work [JAL+22], this provides DimBridge users with a transparent indication of whether a perceived cluster can be adequately explained by a first-order predicate in the original dimensions.

7.5.1 Visualizing Data

In the Projection View, the DR results are visualized with a scatterplot. A user can interact directly with the scatterplot to identify interesting patterns by brushing the related data points. Brushing in the Projection View is supported by two basic interactions. Users can *select* a region of the projection by drawing a bounding box or lasso around that region, or *draw* a selected region between one location in the scatterplot and another. These two brushing interactions allow DimBridge users to select a wide variety of interesting visual patterns that might appear in the DR results:

Select: A user can select a group of data points (P) that are distinct from the rest of the dataset (B) using the bounding box or lasso. In response, the Predicate Induction Engine will attempt to generate predicates that distinguish P from B in the original dimensions. This can be used to identify **clusters** that are unique relative to the rest of the data due to their shape or positioning. Similarly, this can be used to identify groups of global **outliers** that are spatially distant from the rest of the data, as well as regions with lower or higher **spatial density**.

Select and Contrast: Rather than comparing to the rest of the data, a user can use the bounding box or lasso to select another group of data points (B) to compare to. This interaction is used to explain the difference between two **clusters**, local **outliers** and their neighbors, or to explain variation in **spatial density** relative to a local region.

Select and Draw: If a group of data points forms a distinct, continuous **shape**, a user can use the bounding box or lasso to select a group of data points to define the shape's starting point. Then, they can drag the selected region to the shape's endpoint, drawing through the rest of the data points along the way. In response, the Predicate Induction Engine will return a series of predicates that distinguish the selected points (P) at multiple intervals along the shape.

In the SPLOM View, a subspace of the original dimensions is visualized with a scatterplot matrix. Rather than visualizing every dimension, the SPLOM View

includes only those dimensions that are included in a predicate. This reduces clutter in the visualization and helps a user focus only on the dimensions of relevance.

7.5.2 Visualizing Predicates

In the Predicate View, a user can visualize and interact with predicates generated by the induction engine. This view lists each dimension included in a predicate and displays its value ranges. Value ranges have multiple possible representations, depending on the user's interaction:

Select: Each dimension is displayed next to a horizontal bar whose total length represents its full range of values. A segment of each bar is highlighted, representing the range of values defined by the predicate (Fig. 7.3). Users can directly modify the predicate by clicking and dragging either endpoint to adjust the range of the predicate (Fig. 7.7, 7.8).

Select and Contrast: Instead of one value range, two ranges (one for each selection) are displayed for each dimension (Fig. 7.4).

Select and Draw: A user's draw gesture is discretized into a set of discrete selections. Multiple value ranges (one for each selection) are displayed vertically to preserve space (Fig. 7.1).

The set of data points that fall within the predicate are highlighted in both the Projection View and the SPLOM View. Situated between the Projection View and the SPLOM View, the Predicate View acts as a conceptual bridge between a pattern in the DR results and patterns in the original dimensions. Users can modify a predicate by adjusting the value ranges or adding or removing a dimension to observe how these changes impact the Projection and SPLOM Views.

Points in the Projection and SPLOM Views are categorized and assigned a color based on this predicate. Points within the user's selection and the predicate (true positives) are shown in purple, those in the predicate but not the selection (false positives) are shown in red, and those in the selection but not the predicate (false negatives) are shown in blue. All other points (true negatives) are shown in

grey. Categories update dynamically in response to interactions in the Projection (updating the selection) and Predicate (updating the predicate) Views.

7.6 Showcases

In this section, we provide three showcases using DimBridge to analyze image data, motion-capture data, and scientific data. Each of these showcases highlights a user's task in exploring and making sense of high-dimensional data.

7.6.1 Understanding the Space of a Generative Model

As generative AI becomes more commonplace in today's society, understanding a generative model's affordances is vital for instilling model trust [VBF+23, FSB+23]. We employ DimBridge for analyzing the output space of a generative vision model to demonstrate the range of patterns one encounters in DR projections and to illustrate how DimBridge provides the necessary context for making sense of these patterns. More specifically, we consider the StyleGAN3 [KAL+21] image generation model, which aims to generate an image collection depicting animals. We define a collection of textual phrases as attributes – 14 in total – that describe visual appearances one would expect of animals, e.g. "furry", "size", and descriptions of behavior, e.g. "excited", or "suspicious looking". For each phrase, we use CLIP [RKH+21] to score the attribute against a given generated image. We take this set of continuously-valued attributes as our high-dimensional space alongside the generated images to verify findings.

Explaining a Cluster: DimBridge demonstrates cluster analysis in a UMAP projection, highlighting discernible groups within the data, as seen in the scatterplot (Fig. 7.3). Upon brushing one of the clusters, DimBridge derives the predicate that best distinguishes those data items from the rest of the dataset. The predicate reveals the data items represent large, spotted animals with distinct facial expressions, identified as cheetahs upon reviewing selected images. A scatterplot matrix (Fig. 7.3.3-4) contextualizes this, showing cheetahs as uniquely furry and spotted within the

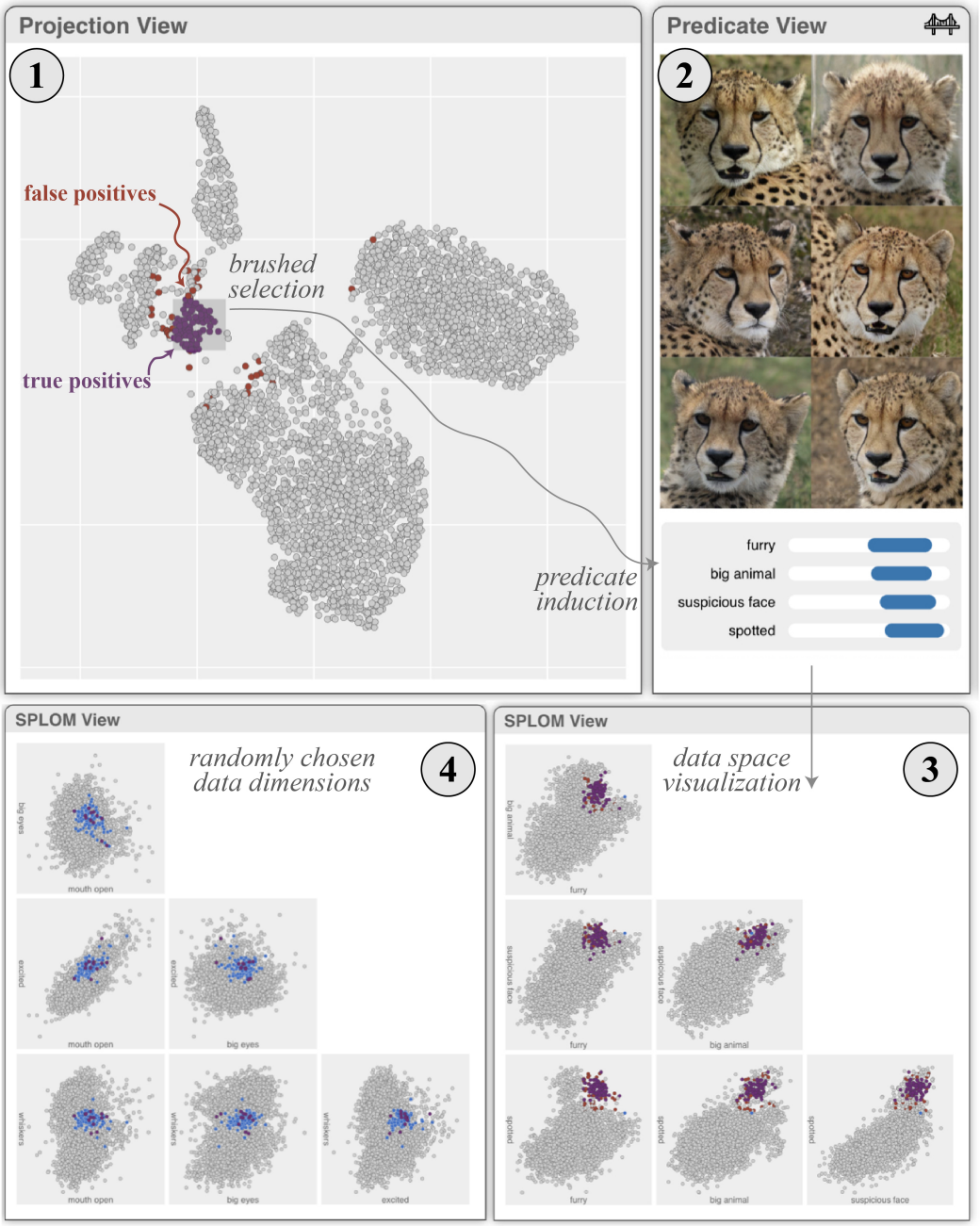


Figure 7.3: We show how DimBridge allows one to better understand a potential cluster within the output space of a generative vision model. Upon performing a brush in the scatterplot (1), DimBridge finds a predicate comprised of 4 attributes (2) that, combined, help distinguish cheetahs from other animals (3), e.g. a big animal with spotted features. In comparison, highlighting brushed data points in randomly chosen four features (4) does not help in distinguishing key features of cheetahs from other animals.

Figure 7.4: We show how DimBridge allows one to better contrast one region of the dimensionality reduction plot from another. Upon brushing two regions, DimBridge finds a predicate that explains the two regions from the rest of the data points. DimBridge finds that while both kittens (blue) and puppies (orange) are not big animals, kittens have whiskers, and puppies have bigger ears.

dataset. We also find that there are a small number of data items that are false positives (colored in red), e.g. animals that fall within the predicate. Some of these data items are close to the brushed region and we observe these are mostly cheetahs, indicating that the perceived cluster that was brushed was slightly imprecise.

As a baseline for comparison, we chose four dimensions and plotted the user's selection in the SPLOM (Fig. 7.3-4). Plotting data on these dimensions hardly contextualizes the user's selection against the dataset, whereas the algorithmically defined predicates allow one to reason about *why* this cluster is discriminative, highlighted in the scatterplot views (Fig. 7.3-3).

Contrasting Two Regions in Context: Beyond comparing a cluster with the overall dataset, users can also compare one cluster to another. In Fig. 7.4, DimBridge highlights the differences between kitten and puppy clusters in the DR projection, distinguishing them by features such as "whiskers" and "big ears" and noting they are both categorized as smaller than other animals in the dataset.

7.6.2 Understanding Progression in Motion Captures

Biomechanical data, such as cyclic motion recordings, are crucial in orthopedic, rehabilitation, and sports research [HSMHW16]. The Multivariate Gait Dataset [HHW22] captures the motion of walking humans in terms of 6 joints angles ({left, right} $\times$ {ankle, knee, hip}) for 101 timestamps and 10 repetitions, among 10 subjects and under 3 bracing conditions ({unbraced, knee brace, and ankle brace}). Here we selected 2 subjects from the 10 in the original dataset for illustration. The dimensionality reduction plot is generated via UMAP using only the 6 angular features, excluding the explicit encoding of timestamps, repetitions, subject IDs, and bracing conditions. In the DR projection scatterplot (Fig. 7.5), we identified three groups of loops corresponding to the three bracing conditions, each containing two overlapping repetition bundles. The difference between the two bundles comes from the subject, as shown in Fig. 7.5. Coloring the DR plot by attributes helps understand data one attribute at a time, but simultaneously analyzing multiple continuous attributes can be challenging. In a motion capture dataset, displaying all 6 joint angles in a SPLOM creates a cognitive load due to the $\binom{6}{2}$ possible subplots. However, DimBridge generates predicates that highlight a few key views, greatly easing the visual workload in exploratory data analysis.

Understanding Progression Over a Curve: Brushing over a segment of one group in the direction of time flow, the predicate induction algorithm recognizes that the segment belongs to a single subject under a particular bracing condition (Fig. 7.6). It further summarizes the progression, in the direction pointed by arrows on the plot, as an increase in left and right ankle angles with a slight decrease in angles on left and right knees. The SPLOM View confirms that this summary indeed distinguishes the brushed segment from the rest of data items.

7.6.3 Examining Populations within a Diabetes Study

Healthcare remains one of the fastest-growing industries in the U.S. [BoLS]. The industry is rapidly modernizing, and producing data at an ever-increasing rate [RC18].

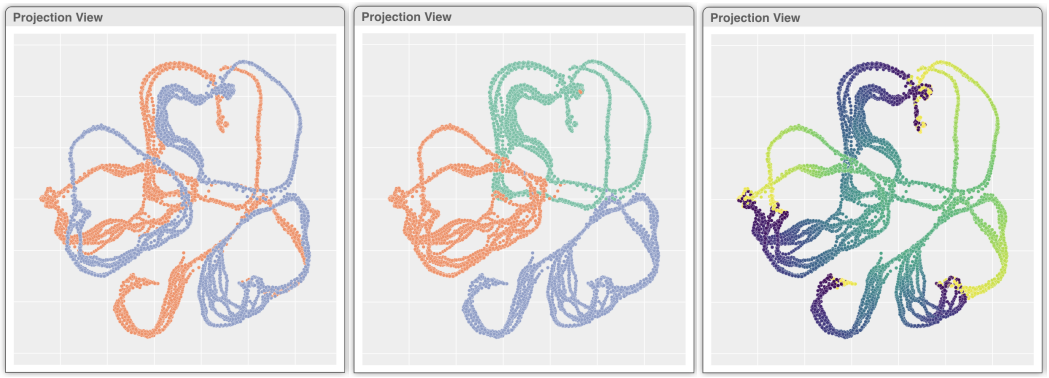


Figure 7.5: DR plot of the Motion Capture dataset. **Left:** color by subject. **Middle:** color by bracing conditions. **Right:** color by time.

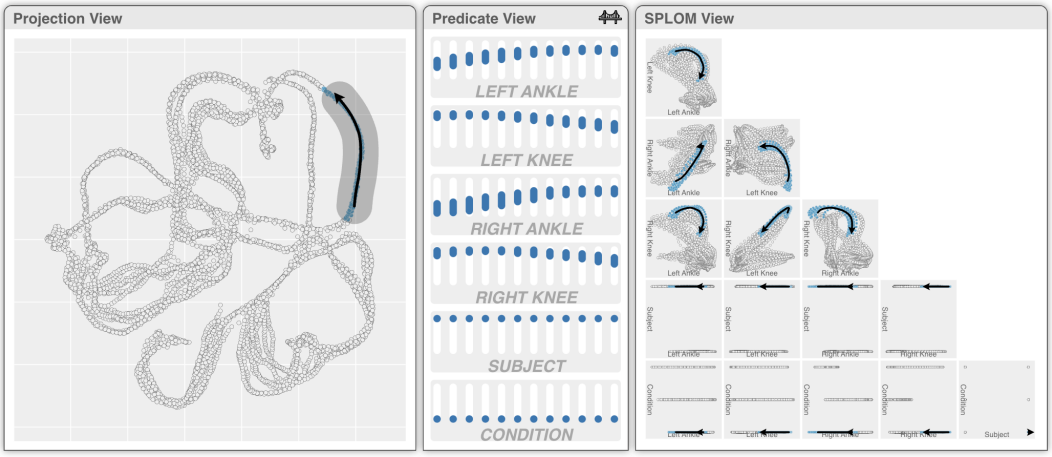


Figure 7.6: DimBridge shows that the curve following the flow of time in the figure captures only one subject and condition, and the segment represents a period with increased angles on left and right ankles and slightly decreased angles on left and right knees.

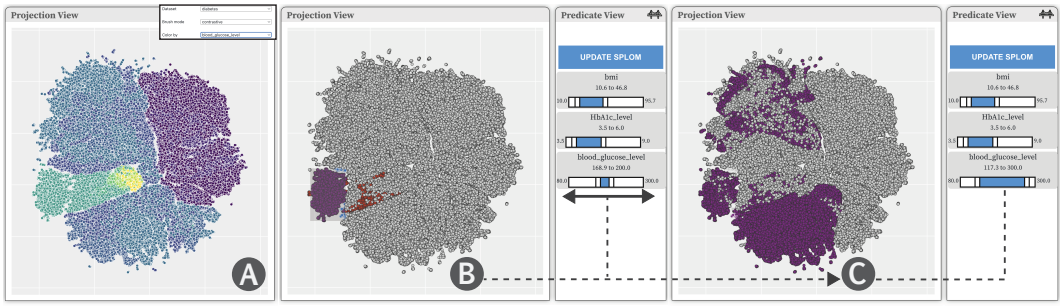


Figure 7.7: A system screenshot showing the projection and predicate components: (A) Users start by coloring the projection according to a feature, here Blood Glucose level. (B) A domain practitioner explores a data subset, creating a predicate based on their selection. (C) They then adjust the Blood Glucose range to a meaningful one, observing changes in the point distribution.

There is a need for new techniques that can help make sense of this influx of data.

In this use case, we examine a dataset focused on diabetes. We examine nine features, as well as a quantitative measure of how far the disease has progressed following a year. The nine features are age, body mass index, blood pressure, total serum cholesterol, low-density lipoproteins, high-density lipoproteins, total cholesterol, serum triglycerides level, and blood sugar level.

Specifying a Known Population: We demonstrate how a practitioner with domain knowledge can use DimBridge to manually adjust ranges within returned predicates to find specific populations within the projection. The user begins by adding relevant features to those used in the projection and SPLOM in the data-space visualization. In this case, the user is interested in BMI, hbA1c level, blood glucose level, and age.

As seen in Fig. 7.7-A, they can color the projection by values for blood glucose values, finding a subset of the data of interest in the lower left of the projection. They can then generate initial predicates from the brush selection they have made to the projection (Fig. 7.7-B).

Interactive Predicate Refinement: The user can then adjust the blood glucose level ranges in the predicate view to focus on particular states of the condition (Fig.7.7-B). By manipulating these ranges, the system dynamically updates the projection view, highlighting the data points that fall within the newly specified thresholds. This process is visualized in Fig.7.7-C, where the adjusted range causes a redistribution of the highlighted points, providing immediate visual feedback on the change.

The ability to update these ranges is not only important for identifying distinct diabetic populations but also for exploring the complex interplay between glucose levels and other biomarkers. Upon updating the ranges to the desired distribution, the SPLOM can also be updated by clicking the "Update SPLOM" button, further allowing users to examine the pairwise relationships between the blood glucose levels and other variables such as BMI, HbA1c, and age. This refined analysis can surface

more subtle correlations or patterns, which might go unnoticed with a more static approach.

7.7 Case Study: Investigating Properties of the Mn1-xGexTe Alloy

The case study presents an application of DimBridge to materials science, focusing on research to better understand the local structure of non-periodic materials such as alloys, that are difficult to model with traditional methods. Stoichiometric analysis in materials science often relies on determining the proportions of elements and their interactions. For the case study in question, our collaborators used sophisticated sampling techniques recommended by Novick et al. [NNG+23], along with density functional theory (DFT) — a quantum mechanical method that calculates the properties of materials based on electron behavior — to explore the structure of the Mn1-xGexTe alloy as it changes with the chemistry. This process created around 500 theoretical model representations, to explore the material's structure with different combinations of Manganese (Mn) and Germanium (Ge). These models are grouped into five types based on their Mn to Ge ratios: $Mn_{0.125}Ge_{0.875}Te$, $Mn_{0.2}Ge_{0.8}Te$, $Mn_{0.25}Ge_{0.75}Te$, $Mn_{0.3}Ge_{0.7}Te$, $Mn_{0.375}Ge_{0.625}Te$.

7.7.1 Theoretical Models and Empirical Validation: Current Analysis Methods

Characterizing theoretical models is crucial for understanding how atoms bond and arrange themselves and the local structure, which directly relates to the properties of the material. For non-periodic materials like alloys, characterization involves theoretical and empirical methods to simulate atom distributions. Standard analysis involves comparing these detailed computational predictions with actual experimental data.

For the data used in this case study, the real-world empirical data used to confirm the theoretical models were obtained from the Spallation Neutron Source

at Oak Ridge National Lab Researchers traditionally visualize and analyze data through computational simulations, graphical representations, and empirical data comparisons. In this area of research, Pair Distribution Functions (PDF) plots [YG11] are used to understand and compare the local atomic structure of different alloy compositions to one another and the experimental data Signals identified in the data are referred to as "peaks", which we will refer to in the subsequent sections. Generating lower-dimensional representations of the data is straightforward, but connecting these to the underlying science driving the clustering is very difficult with our collaborators' current workflow.

7.7.2 Analyzing the Mn1-xGexTe Alloy with Dimbridge

Our collaborators were interested in using DimBridge as an exploratory analysis tool to understand the similar characteristics shared by the computationally generated theoretical models for the Mn1-xGexTe alloy that align with the experimental data from Oak Ridge National Lab. Specifically, they asked the following questions: **(1)** How does changing the proportion of manganese (Mn) affect the way the alloy's structure shifts from a slanted, diamond-like shape to a straight-edged, cube-like shape? **(2)** Are the shapes formed by Mn atoms similar across the alloy space? For example, if we look at shapes with 6 out of 12 manganese atoms in their outer layer, do they look the same from one Mn-Ge mixture to another?

7.7.3 Walkthrough of Analysis with Dimbridge

During our collaborator's use of DimBridge, it became apparent that the system is not only a tool for bridging dimensional spaces but a catalyst for testing hypotheses and identifying areas for rich exploration.

Our collaborator for this case study, a graduate student at an affiliate university, studies how varying the ratio of Mn and Ge in alloy samples induces a transition between crystal structures, to better understand the relationship between local atomic coordination, global crystal structure, and material properties. Having a deep familiarity with the data, she familiarized herself with DimBridge through

a process of generating hypotheses about clusters, finding an area of interest, and refining both her mental model of the data and investigating strategies for subsequent data runs.

Her exploration and analysis with DimBridge progressed in three distinct iterations, defined by a revised dataset tailored to a new investigative strategy. All three of these iterations used datasets generated from the five subsets but differed in the properties for these subsets. As mentioned previously, subsets varied in Mn to Ge combinations: $Mn_{0.125}Ge_{0.875}Te$, $Mn_{0.2}Ge_{0.8}Te$, $Mn_{0.25}Ge_{0.75}Te$, $Mn_{0.3}Ge_{0.7}Te$, $Mn_{0.375}Ge_{0.625}Te$.

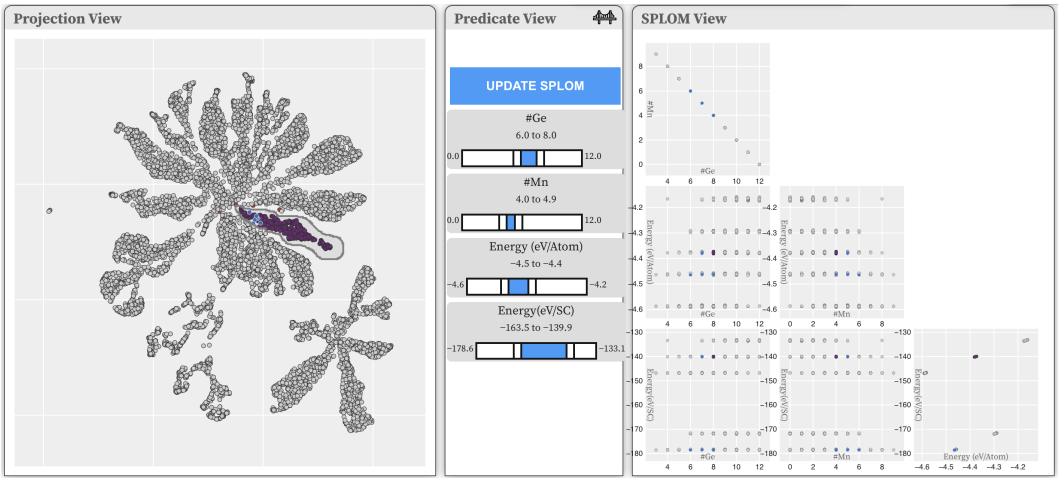


Figure 7.8: UI of DimBridge with cluster selected, showing the division of points into clean groups in the SPLOM. This indicates to our collaborator the strong influence Mn has on the clustering of data.

7.7.3.1 Phase one: Investigating Mn's Influence on Separation

The first phase of exploration involved a broad exploration of all five subsets in which she experimented with various hyperparameters to investigate their impact on data separation, including adjusting and observing the effects of removing attributes with predictable influences on the dataset's division into clusters, specifically investigate whether the Mn coordination shell was important.

Between the removal of attributes, the projection would be color-coded again to confirm separations she had predicated would be present, such as the division of

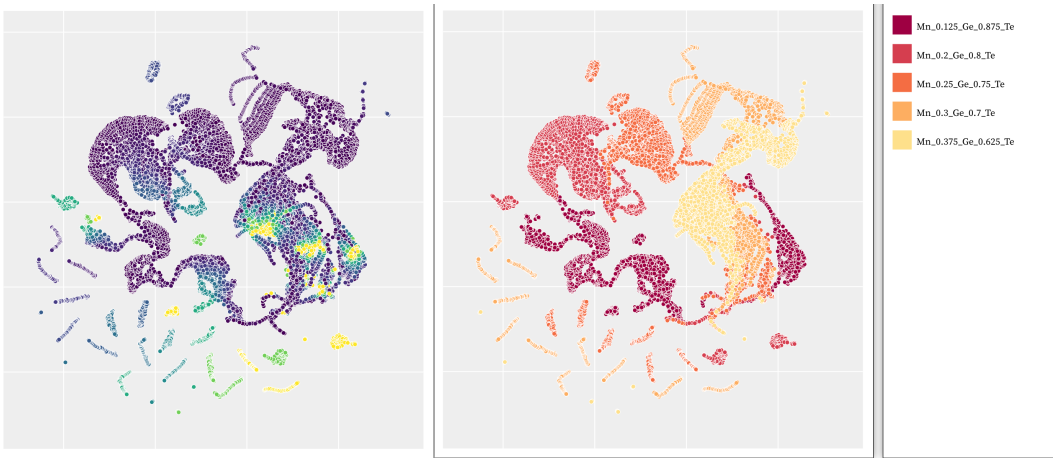


Figure 7.9: Projection view showing temperature dependencies color-coded by attributes. The left is color-coded by temperature and the right is color-coded by the data subset indicating the varying combinations of Mn and Ge.

each subset into largely disparate clusters. She would then brush clusters of interest to inspect any correlations between attributes of interest or confirm that the clusters she was inspecting resulted from something uninteresting, such as the SPLOM in Figure 7.8, which were so separated by the proportion of Mn and Ge, it generated results uninformative to our collaborator.

7.7.3.2 Phase two: Investigating Temperature Dependencies

Her findings indicated that the number of Mn atoms significantly influenced data separation in lower-dimensional spaces, prompting her to adjust her analysis method to focus on the relationships between different attributes and her experimental data. In her next analysis phase, she investigated how temperature affects each alloy combination (Fig 7.9), focusing on attributes that offer insights into the Pair Distribution Function (PDF) plots. These plots helped her link local atomic structures, seen as peaks in the PDF, to the properties of various alloys. By examining the projection's top curve to see how values change within interesting loop shapes, she confirmed that with increasing temperature, the data points tend to converge (Fig. 7.10-A).

She also makes note of the small, isolated worm-like clusters outside of the larger shape. Drawing a shape over one of the small clusters, she can see that these

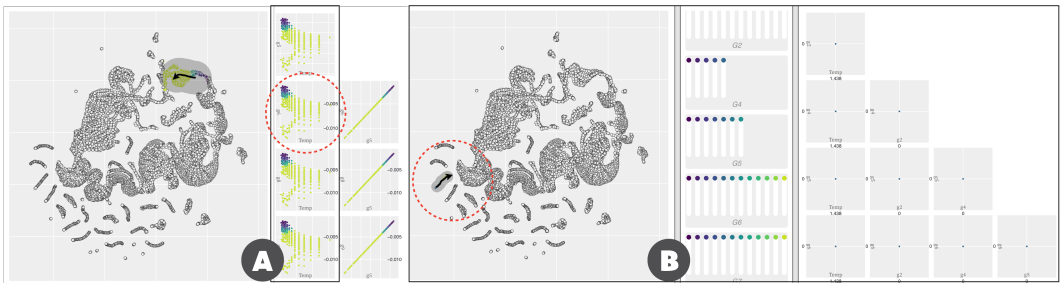


Figure 7.10: (A) DimBridge explains a trail within the alloy temperature prediction dataset. The SPLOM in the data view shows curves merging as the temperature increases. (B) View of the data subset of the drawn shape selection, indicating that the cluster selection contains only one temperature value.

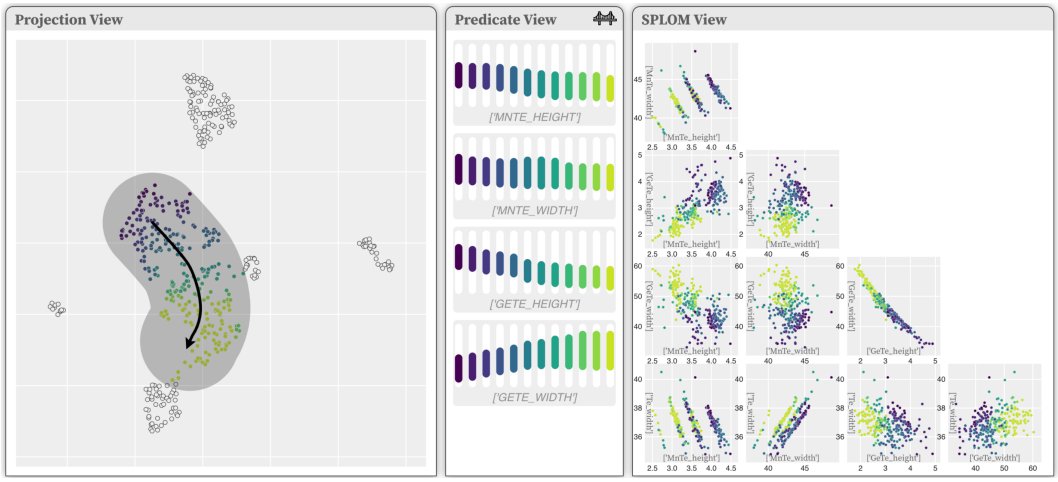


Figure 7.11: DimBridge views showing the attributes of the drawn shape selection of the larger cluster. In the SPLOM, you can see the correlation between Te height and Mn height.

small clusters are composed of a single temperature (Fig. 7.10-B). She noted, "well, something is definitely happening here! For the [$Mn_{0.3}Ge_{0.7}Te$] datasets, it looks like it's making worms grouped by temperature for a bunch of different POSCARs. I assume that's the case for the [$Mn_{0.25}Ge_{0.75}Te$], and [$Mn_{0.2}Ge_{0.}Te$], worms as well." Color coding the projection by combination, she can confirm her assumption on the data subsets.

7.7.3.3 Phase Three: Investigating Subsets of Interest

In the next stage of her exploration, our collaborator creates a dataset for these subsets in question. Instead of the temperature dependencies, she focuses on the

outputs of her existing model to get a clearer tie back to her experiments. This dataset, smaller and more targeted, focuses on subsets that previously formed unique, worm-like clusters. Building on her analysis suggesting Mn influences Te positions, she first colors the projection by $MnGe_r$. After identifying an interesting cluster, she examines its predicate ranges and potential relationships. Upon observing correlations, she draws a shape down the larger cluster (Fig. 7.11), leading to a key observation: the height and shape of MnTe peaks significantly impact Te peak locations and heights, showing a stronger correlation with MnTe peaks than with GeTe/Te peaks. This consistency across clusters suggests that while germanium (Ge) scarcely affects Te atom positions, manganese (Mn) plays a pivotal role, altering them noticeably. "I really love seeing this because it's showing me how influential the Mn is in the PDF patterns."

Revisiting the questions posed in the previous section: **How does changing the proportion of manganese (Mn) affect the way the alloy's structure shifts from a slanted, diamond-like shape to a straight-edged, cube-like shape?** Our collaborator found that Mn significantly influences the alloy's structure, noting a clear correlation between the intensities of Mn-Te and Te-Te peaks across the alloy space. This observation aids in understanding how Ge atoms cause local distortions, affecting the alloy's appearance, especially in low Mn concentrations, where Ge tends to shift the structure towards a more rhombohedral form. **Are the shapes formed by manganese (Mn) atoms similar across the alloy space?** In the initial exploration stage, projections showed clustering by the number of Mn atoms in the coordination shell, regardless of composition. This suggests that the proximity of Mn atoms within their shells is more crucial than the overall elemental mix in the unit.

7.7.4 Observations

We observed a few notable things during our collaborator's interaction with DimBridge. Her process was highly iterative, moving from broad to more pointed in the data and hypotheses, often switching between several datasets. The adaptability

of DimBridge proved to be a significant asset, for switching between one dataset to another, adding and subtracting columns used in the projection and data view from the system's UI, and getting immediate feedback on selections of interest. Particularly important was the ability to dynamically update the projection and data spaces, as this allowed her to rapidly iterate on her hypotheses and direction of analysis, as well as account for unexpected things, such as a cluster defined by attributes that were not expected. As well, our collaborator found DimBridge's functionality to draw a shape was extremely valuable for understanding the change in properties between one data subset to another, especially during the second phase of the analysis of temperature dependencies as seen in Figure 7.10-A.

7.8 Discussion

DimBridge is designed to support the user in making sense of visual patterns in dimensionality reduction projections. In this section, we reflect on emergent insights, implications of DimBridge's value beyond its original design goals, limitations, and some opportunities for future work.

7.8.1 DimBridge Design Considerations: Why SPLOM?

We chose to use a Scatter Plot Matrix (SPLOM) due to its efficacy for visualizing patterns in high dimensional space. This could involve the characterization of subspaces or the relationships between attribute values, which are both important tasks for our collaborators. Additionally, the SPLOM view allows you to visualize the predicate value ranges of the relevant attributes in all possible pairings. For these reasons, we chose SPLOM over alternative high dimensional techniques, detailed in 2.1. However, future work will explore other methods of visualization to make DimBridge more flexible for a wider variety of datasets and tasks.

7.8.2 The Value of Flexibility

Along with creating a flexible analysis environment, it is also important to consider how we can enhance DimBridge to be more adaptable in terms of its composition and features. Below, we outline several opportunities for doing so.

Beyond Projection Visualizations: Not all data and tasks benefit from the exploration of a projection. The projection view can be substituted for other methods of visualizing an overview or summary of the dataset, depending on the analysis tasks. Considering an example in the context of material science, imagine researchers are seeking to develop a new alloy with high strength and corrosion resistance for aerospace applications. They use a connectivity matrix to explore potential candidates, where the matrix includes a variety of known alloys, each characterized by their mechanical and chemical properties. Selecting cells or submatrices are entry points for understanding the properties of nearest neighbors of a selected cell, or defining ranges of desired properties to highlight relevant cells in the matrix. Or in a more general context, one could also imagine a scenario where the projection view might contain geospatial data, and patterns of location could be explored across relevant dimensions. Future work will explore the use of predicates as a bridge between high-dimensional data and alternative low-dimensional representations, beyond just projections.

Beyond Axis-Aligned Dimensions: DimBridge currently assumes the original (axis-aligned) data dimensions in the data-space visualization as they are the most interpretable. However, this design requirement can be lifted for advanced users who can understand complex data dimensions. Although we illustrated DimBridge using the original data dimensions in Section 7.7, we observe that the predicate induction could also have been performed using the principal components with the SPLOM showing the data with principal components as the data dimensions. The resulting SPLOM visualization will be less interpretable, but the use of principle components (and possibly other non-linear dimensions) can result in predicates that better fit the user-selected data.

Adapting Predicate Induction: DimBridge's predicate induction engine uses the RPI and Predicate Regression algorithms, representing two extremes of the accuracy/scalability tradeoff. Future work exploring predicate induction algorithms that take a more balanced approach to this tradeoff could be beneficial. Additionally, the predicate induction algorithm can be further tuned to the task of understanding DR results by explicitly accounting for distortions introduced by the DR algorithm. "Distortion aware" predicate induction could more effectively identify patterns in the original dimensions by adjusting for distortions in regions brushed by a user.

7.9 Conclusion

In this chapter we present DimBridge, a system that *bridge*s a projection space with the original data space using first-order predicate logic. DimBridge connects patterns observed in the projection space to the original data space, helping users understand the pattern within the familiar data space. This decreases the likelihood of false discoveries resulting from spurious structure within the projection. DimBridge is agnostic to the projection algorithm, the visualization technique used within the data space, and the predicate induction algorithm itself. We illustrate three showcases of DimBridge within scientific data, motion-capture data, and imagery data. Finally, we evaluated the utility of DimBridge with a domain expert who found the design to be helpful in her workflow. This demonstrates that DimBridge can be help users explore and make sense of their projections.

Chapter 8

Discussion

Throughout this thesis, we have discussed some problems with projection methods, as well as some algorithms and systems to help users make sense of projections. This chapter discusses other problems with projections and some directions for future work.

8.1 Further Limitations of Projections

Projections can produce an informative lower-dimensional representation of data that can guide users in exploring some aspect or structure of their data. Unfortunately, they also have limitations. In this thesis, we have focused on some tools that help users overcome some limitations, but some have not been mentioned. In this section, we discuss yet unmentioned limitations, as well as when a projection may not be the appropriate tool. We begin with the most important limitations: (1) intrinsic dimensionality and (2) visual and cognitive scalability.

Intrinsic dimensionality does not warrant a long discussion but must be mentioned. Practitioners have a tendency to believe that dimensionality reduction methods can somehow ignore a dataset's intrinsic dimensionality, but this is not the case. Van Der Maaten and Hinton [vdMH08] put it bluntly: "...it is by definition impossible to fully represent the structure of intrinsically high-dimensional data in two or three dimensions." Luckily, much of the data that practitioners are interested

in projecting are not intrinsically high-dimensional, and we do not necessarily have to fully represent the structure to have a useful projection. That being said, this cannot be ignored and should be considered when evaluating how to represent and gain insight from complex data.

We have discussed the computational scalability of projection methods at length in Chapter 5 and offered HyperNP as a possible solution. The next hurdles are visual and cognitive scalability, assuming the data can be represented in a lower-dimensional space.

We begin by discussing visual scalability, which we use to mean the ability to represent the projection within a scatterplot. Note that this differs from what we would refer to as the cognitive scaling of the representation (i.e., even if I can perfectly represent it on a 2D canvas, can a user cognitively process the information within a reasonable amount of time). Projections in scatterplots visually scale well with the number of features in a dataset since we usually project down to two or three dimensions regardless of the number of features. The same cannot be said for the number of instances. Representing each instance as a point inevitably leads to overplotting after a certain point. There are mechanisms within popular nonlinear projection methods to deal with overplotting, such as specifying a minimum distance between points [MHSG18]. However, these methods are necessarily further distorting the projection.

Even if the overplotting issues are solved in a satisfactory manner, there is still the issue of cognitive scalability. Representing each instance as a point will eventually be too much information to process in a reasonable amount of time. Some rendering methods, agnostic to the projection method, can help with both visual and cognitive scalability, but these tend to be aggregation methods. The two most common are what we will refer to as point-density methods and bucket methods. Point-density methods begin with individual points but change the opacity to represent the number of over-plotted points [dat24]. On the other hand, bucket-based visualizations turn the scatterplot into fixed areas such as hexagons and color each hexagon by the number of points [Hun07]. The bucket method can be thought of as a generalization

of the point-density method. One can constrain the size of the visual area representing the bucket to as small as one likes, eventually returning to the point-density method. The larger the buckets, the greater the loss of information in terms of the density at each point, whereas at the extreme of a bucket per point, the visualization tends to fail in terms of cognitive complexity—it is just too much information to digest. There is a sweet spot for most visualizations, but several further questions arise. The most significant two are how one contends with labeled data if the visual channel is being used for density (and there are some clever existing answers to that, but again, they tend to increase cognitive complexity) and, more importantly, projecting to buckets brings us much closer to clustering than dimensionality reduction.

This brings us to the next consideration: when should one use projection methods? Why not just use clustering methods? Well, the answer depends highly on the task and projection method. Brehmer et al. [BSIM14] discuss tasks for projections, splitting them into *dimension* and *cluster* tasks.

Dimension-based tasks are split into *name* and *map* tasks. Put succinctly, *name* means assigning semantics to a dimension, and *map* means finding the composition of original dimensions within the newly synthesized dimensions. These tasks are often globally impossible for popular nonlinear techniques such as UMAP or t-SNE. PCA, a linear method that produces an orthogonal projection where the variance of the projection is maximized, allows users to achieve these tasks more easily. However, it is also unclear what higher-level tasks these *dimension* tasks support, as they mostly allow the practitioner to make better sense of the plot. The one particular advantage of some linear methods like PCA is that they allow the user to reason about distances within the plot efficiently, which we have shown is not always so simple for nonlinear projections (see Chapter 6).

For *cluster*-based tasks, practitioners would be better off using clustering methods directly for any static-based visualization. This completely sidesteps a number of the interpretability problems with nonlinear projection methods. The exception to this line of thinking is interactive visualization.

Regardless of visual and cognitive scalability issues, there is much to be gained

from using dimensionality reduction results as the center of a coordinated view in visual analytics systems. Take, for example, the selection of individual instances and groups and the interplay between them, as we have seen in DimBridge. The types of interactions to support these sorts of selections, as well as the ability to hover over instances for details-on-demand is intuitive to users. As discussed above, scatterplots run into issues of cognitive scalability, but there are many visual analytics applications where this is not the case, and projection methods are a strong tool to be leveraged. That said, a number of current tools fail to scale computationally and cannot be used with real-world datasets. Currently, tools like HyperNP can help with computational scalability. In the future, though, things like webAssembly combined with libraries like cuML [RPN20] might solve the computational scalability completely as computational scalability starts to exceed visual or cognitive scalability.

In conclusion, while interactive projections are helpful in various cases, most static nonlinear projection visualizations would be better off as a dendrogram of a clustering algorithm or something similar. There is an argument to be made that t-SNE or UMAP might give one some sense of when clusters are close or when a point may be between two clusters, but the way these projections are constructed, it is tough to attain this sort of information from these plots. This brings us to future work.

8.2 Future Work

The two major thrusts for future work are projection redesign and a unified platform for neural-network-driven projection.

There are two angles from which a projection redesign is worth considering: (1) the visual representation of the projection and (2) the algorithm that generates the embedding. We have briefly discussed that projections can be in dimensions higher than the regular 2D projection. Projecting data to a lower-dimensional space without having to go all the way to 2D to obtain a good visual representation would help tremendously to assuage many cases of potential higher intrinsic dimensionality.

Work needs to be done to explore which existing techniques would work best for which tasks and whether a completely new design is necessary. For instance, parallel coordinates work very well up through twenty or so dimensions, so a projection to twenty dimensions could easily be explored in a PCP plot. Unfortunately, it is very hard to understand the overall distance between two instances across a long parallel coordinates plot, and they do not scale well in terms of the number of instances without augmentation. Would they be a good fit for the results of a projection method? Scatterplots may be the only appropriate home for current projection methods, as they were designed with scatterplots in mind. The next question is, how would one design a projection method explicitly for other forms of multi-dimensional visualization (such as a PCP)? Additionally, in cases where the number of instances is so great that an aggregation technique is necessary, the projection method could simply comprise creating clusters and then organizing the clusters themselves by distance. The simplest path forward to test the idea would be to combine existing algorithms. One could cluster the instances using DBSCAN [EKSX96] and then use the centroid of each cluster as an instance for UMAP [MHSG18] in order to determine a position within a scatterplot for each cluster. This would likely yield worse results than purpose-built methodologies, but an acceptable algorithm to begin to explore the appropriate visual representation.

The other major thrust for future work revolves around creating a platform for neural-network-driven projection. As discussed in Chapters 5 and 6, using a neural network as a backbone for projection techniques has many advantages. We will touch on another here: it creates an easily unified framework for embedding these techniques within visual analytics applications. This is quite a practical advantage, as most developers are now familiar enough to create a HyperNP or NNInv model using high-level frameworks such as PyTorch or Keras, and the tooling coming out of the recent investments in AI and ML makes embedding these models within applications and then deploying them, a breeze. Basing a dimensionality toolkit around neural network surrogate models also allows one to easily incorporate new DR techniques by simply training a new model. At the same time, rapidly

advancing technology that speeds up training and inference while decreasing the size of models lets the framework scale easily with little developer investment. HyperNP significantly improves the exploration aspect of NNInv, as it makes it easier to explore new plots that will lead to different possible interpolations between points. These combine even better to augment DimBridge since NNInv can be used to unproject centroids of selections within DimBridge or even between two selections or along a trail. This thesis contributes the first steps towards transitioning dimensionality reduction within visual analytics to a neural-network-driven framework. However, future work will be necessary to create a unified platform for practitioners.

Chapter 9

Conclusion

This thesis aims to aid practitioners in understanding projections of their high-dimensional data. It examines some of the challenges practitioners face when presented with complicated data and visualizations and contributes three novel contributions towards easing this process.

First, we consider and distill challenges experienced by practitioners across a variety of fields in Chapter 3. These challenges are condensed to three major themes relevant to dimensionality reduction: the ability to (1) explore the projected space, (2) examine multiple plots, and (3) understand errors and distortions. We then discuss three contributions that work towards alleviating these major challenges.

Second, we design HyperNP, a method of approximating any projection technique, adding out-of-sample capabilities and the ability to project across hyper-parameters in real-time. HyperNP can help users explore the projected space but primarily assists with allowing them to quickly and easily examine multiple plots. We demonstrate HyperNP applied to three popular projection techniques (t-SNE, UMAP, and Isomap), qualitatively showcasing the similarity to the underlying projection methods and discussing the implications for these three methods. Finally, we produce a quantitative evaluation of HyperNP, examining the quality of results, the scalability of training, and the interactivity of its inference.

Next, we contribute NNInv, a neural-network-based approach to inverse projection. NNInv can help practitioners explore the projected space and generate

hypotheses about instances that might exist between ground-truth data. It can also be used to generate dense-map visualizations that illuminate errors and distortions within the plot. We compare NNInv to existing inverse-projection methods, showing it superior in speed and quality. We also demonstrate NNInv by exploring how it behaved on well-known datasets and provide a showcase of some of the techniques made possible by a quick and accurate inverse-projection method. Finally, we also provide two novel visualizations for evaluating the quality of inverse-projection methods.

The final contribution is the system DimBridge, which allows users to directly query a projection plot for explanations of selections using predicates. DimBridge primarily focuses on allowing users to accurately explore the projected space using first-order predicate logic, but its ability to generate an explanation of a selection can also be used to understand errors and distortions within a plot. We also introduce a novel interaction design that facilitates the selection of visual patterns in DR projections and an algorithm that generates predicates with smoothness constraints. We evaluate DimBridge through several showcases with different domain applications, a case study, and evaluations with researchers from materials science and pharmaceutical drug discovery.

These three contributions focus on addressing the three challenges outlined above and represent a major push toward improving the usability of dimensionality reduction methods.

Bibliography

[AAB+15] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[ABL+19] Marco Angelini, Graziano Blasilli, Simone Lenti, Alessia Palleschi, and Giuseppe Santucci. Towards enhancing radviz analysis and interpretation. In *2019 IEEE Visualization Conference (VIS)*, pages 226–230. IEEE, 2019.

[ABMC+15] Elisa Amorim, Emilio [Vital Brazil], Jesús Mena-Chalco, Luiz Velho, Luis Gustavo Nonato, Faramarz Samavati, and Mario [Costa Sousa]. Facing the high-dimensions: Inverse projection with radial basis functions. *Computers & Graphics*, 48:35 – 47, 2015.

[AEC+22] G. Appleby, M. Espadoto, R. Chen, S. Goree, A. C. Telea, E. W. Anderson, and R. Chang. Hypernp: Interactive visual exploration of

multidimensional projection hyperparameters. *Computer Graphics Forum*, 41(3):169–181, 2022.

[AEM11] Georgia Albuquerque, Martin Eisemann, and Marcus Magnor. Perception-based visual quality measures. In *Proceedings of the 2011 IEEE Conference on Visual Analytics Science and Technology*, pages 13–20, 2011.

[Alg21] Algorithmia. 2021 enterprise trends in machine learning, 2021.

[Alm07] June S Almenoff. Innovations for the future of pharmacovigilance. *Drug safety*, 30(7):631–633, 2007.

[And35] Edgar Anderson. The irises of the gaspe peninsula. *Bulletin of the American Iris Society*, page 2–5, 1935.

[And72] D. F. Andrews. Plots of high-dimensional data. *Biometrics*, 28(1):125–136, 1972.

[Ans73] F. J. Anscombe. Graphs in statistical analysis. *The American Statistician*, 27(1):17–21, 1973.

[AS16] Michael Aupetit and Michael Sedlmair. Sepme: 2002 new visual separation measures. In *Proceedings of the 2016 IEEE Pacific Visualization Symposium*, pages 1–8, 2016.

[Aup07] Michaël Aupetit. Visualizing distortions and recovering topology in continuous projection techniques. *Neurocomputing*, 70(7-9):1304–1330, 2007.

[AZL+18] Sara Alspaugh, Nava Zokaei, Andrea Liu, Cindy Jin, and Marti A Hearst. Futzing and moseying: Interviews with professional data analysts on exploration practices. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):22–31, 2018.

[BBFd07] Henk Blanken, Henk Ernst Blok, Ling Feng, and Arjen P. de Vries, editors. *Multimedia Retrieval*. Data-Centric Systems and Applications. Springer, Netherlands, 2007.

[BBG19] Katy Börner, Andreas Bueckle, and Michael Ginda. Data visualization literacy: Definitions, conceptual frameworks, exercises, and assessments. *Proceedings of the National Academy of Sciences*, 116(6):1857–1864, 2019.

[BBH12] K. Bunte, M. Biehl, and B. Hammer. A general framework for dimensionality reducing data visualization mapping. *Neural Computation*, 24(3):771–804, 2012.

[BC06] Virginia Braun and Victoria Clarke. Using thematic analysis in psychology. *Qualitative research in psychology*, 3(2):77–101, 2006.

[BCH+22] Alex Bäuerle, Ángel Alexander Cabrera, Fred Hohman, Megan Maher, David Koski, Xavier Suau, Titus Barik, and Dominik Moritz. Symphony: Composing interactive interfaces for machine learning. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, pages 1–14, 2022.

[Ben13] Yoshua Bengio. Deep learning of representations: Looking forward. In *International Conference on Statistical Language and Speech Processing*, pages 1–37. Springer, 2013.

[Ber83] Jacques Bertin. *Semiology of graphics*. University of Wisconsin Press, Madison, Wis., 1983.

[Ber19] Scott Berinato. Data science and the art of persuasion. *Harvard Business Review*, 97(1):126–137, 2019.

[BKS20] Jinwook Bok, Bohyoung Kim, and Jinwook Seo. Augmenting parallel coordinates plots with color-coded stacked histograms. *IEEE Trans-*

actions on Visualization and Computer Graphics, 28(7):2563–2576, 2020.

[BKVR+20] Michael Böttinger, Helen-Nicole Kostis, Maria Velez-Rojas, Penny Rheingans, and Anders Ynnerman. Reflections on visualization for broad audiences. In *Foundations of Data Visualization*, pages 297–305. Springer, 2020.

[BLBC12] Eli T Brown, Jingjing Liu, Carla E Brodley, and Remco Chang. Disfunction: Learning distance functions interactively. In *2012 IEEE conference on visual analytics science and technology (VAST)*, pages 83–92. IEEE, 2012.

[BoLS] Bureau of Labor Statistics. Healthcare Occupations : Occupational Outlook Handbook: : U.S. Bureau of Labor Statistics — bls.gov. https://www.bls.gov/ooh/healthcare/home.htm. [Accessed 29-Mar-2023].

[Bra03] Matthew Brand. Fast online svd revisions for lightweight recommender systems. In *Proceedings of the 2003 SIAM International Conference on Data Mining*, pages 37–46, 2003.

[BS02] Mukund Balasubramanian and Eric L. Schwartz. The isomap algorithm and topological stability. *Science*, 295(5552):7–7, 2002.

[BSIM14] Matthew Brehmer, Michael Sedlmair, Stephen Ingram, and Tamara Munzner. Visualizing dimensionally-reduced data: Interviews with analysts and a characterization of task sequences. In *Proceedings of the Fifth Workshop on Beyond Time and Errors: Novel Evaluation Methods for Visualization*, pages 1–8, 2014.

[BTK11] Enrico Bertini, Andrada Tatu, and Daniel Keim. Quality metrics in high-dimensional data visualization: An overview and systematiza-

tion. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2203–2212, 2011.

[Buh11] Peter Buhlmann. *Statistics for high-dimensional data : methods, theory and applications*. Springer series in statistics. Springer, Heidelberg ; New York, 2011.

[C+15] François Chollet et al. Keras. https://keras.io, 2015.

[CD18a] Marco Cavallo and Çağatay Demiralp. Clustrophile 2: Guided visual clustering analysis. *IEEE transactions on visualization and computer graphics*, 25(1):267–276, 2018.

[CD18b] Marco Cavallo and Çağatay Demiralp. A visual interaction framework for dimensionality reduction based data exploration. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, page 1–13, New York, NY, USA, 2018. Association for Computing Machinery.

[CFGT20] Anamaria Crisan, Brittany Fiore-Gartland, and Melanie Tory. Passing the data baton: A retrospective analysis on data science work and workers. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):1860–1870, 2020.

[CG15] John P. Cunningham and Zoubin Ghahramani. Linear dimensionality reduction: Survey, insights, and generalizations. *Journal of Machine Learning Research*, 16(89):2859–2900, 2015.

[CG16] Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA, 2016. ACM.

[CHAS18] René Cutura, Stefan Holzer, Michaël Aupetit, and Michael Sedlmair. VisCoDeR: A tool for visually comparing dimensionality reduction

algorithms. In *Proceedings of the 26th European Symposium on Artificial Neural Networks*, 2018.

[CLGD19] Ricky T. Q. Chen, Xuechen Li, Roger Grosse, and David Duvenaud. Isolating sources of disentanglement in variational autoencoders, 2019.

[CMJ+20] Angelos Chatzimparmpas, Rafael M Martins, Ilir Jusufi, Kostiantyn Kucher, Fabrice Rossi, and Andreas Kerren. The state of the art in enhancing trust in machine learning models with the use of visualizations. In *Computer Graphics Forum*, volume 39, pages 713–756. Wiley Online Library, 2020.

[CMK20] Angelos Chatzimparmpas, Rafael M. Martins, and Andreas Kerren. t-viSNE: Interactive assessment and interpretation of t-sne projections. *IEEE Transactions on Visualization and Computer Graphics*, 26(8):2696–2714, 2020.

[CMS99] Stuart K. Card, Jock D. Mackinlay, and Ben Shneiderman, editors. *Readings in information visualization: using vision to think*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1999.

[CS13] Kathrin Cresswell and Aziz Sheikh. Organizational issues in the implementation and adoption of health information technology innovations: an interpretative review. *International journal of medical informatics*, 82(5):e73–e86, 2013.

[dat24] holoviz/datashader, April 2024. original-date: 2015-12-23T18:02:20Z.

[dBD+12] E. P. dos Santos Amorim, E. V. Brazil, J. Daniels, P. Joia, L. G. Nonato, and M. C. Sousa. ilamp: Exploring high-dimensional spacing through backward multidimensional projection. In *2012 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 53–62, 2012.

[DGMM11] Jessica T DeCuir-Gunby, Patricia L Marshall, and Allison W McCulloch. Developing and using a codebook for the analysis of interview data: An example from a professional development research project. *Field methods*, 23(2):136–155, 2011.

[DGSA20] Brittany Davis, Maria Glenski, William Sealy, and Dustin Arendt. Measure utility, gain trust: Practical advice for xai researchers. In *2020 IEEE Workshop on TRust and EXpertise in Visual Analytics (TREX)*, pages 1–8. IEEE, 2020.

[Dij59] Edsger W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271, Dec 1959.

[DR82] David Donoho and Ernesto Ramos. Primdata: data sets for use with prim-h. Technical report, Version for second (15-18, Aug, 1983) Exposition of Statistical Graphics Technology, by American Statistical Association, 1982.

[DWF+19] M. Dowling, J. Wenskovitch, J. T. Fry, S. Leman, L. House, and C. North. Sirius: Dual, symmetric, interactive dimension reductions. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):172–182, 2019.

[EAS+23] Mateus Espadoto, Gabriel Appleby, Ashley Suh, Dylan Cashman, Mingwei Li, Carlos Scheidegger, Erik W. Anderson, Remco Chang, and Alexandru C. Telea. Unprojection: Leveraging inverse-projections for visual analytics of high-dimensional data. *IEEE Transactions on Visualization and Computer Graphics*, 29(2):1559–1572, 2023.

[EHA+22] Klaus Eckelt, Andreas Hinterreiter, Patrick Adelberger, Conny Walchshofer, Vaishali Dhanoa, Christina Humer, Moritz Heckmann, Christian Steinparz, and Marc Streit. Visual exploration of relationships and structure in low-dimensional embeddings. *IEEE Transactions on Visualization and Computer Graphics*, 2022.

[EHA+23] Klaus Eckelt, Andreas Hinterreiter, Patrick Adelberger, Conny Walchshofer, Vaishali Dhanoa, Christina Humer, Moritz Heckmann, Christian Steinparz, and Marc Streit. Visual exploration of relationships and structure in low-dimensional embeddings. *IEEE Transactions on Visualization and Computer Graphics*, 29(7):3312–3326, 2023.

[EHFT20a] Mateus Espadoto, Nina S. T. Hirata, Alexandre X. Falcão, and Alexandru C. Telea. Improving neural network-based multidimensional projections. In Andreas Kerren, Christophe Hurter, and José Braz, editors, *Proceedings of the 15th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, VISIGRAPP 2020, Volume 3: IVAPP, Valletta, Malta, February 27-29, 2020*, pages 29–41. SCITEPRESS, 2020.

[EHFT20b] Mateus Espadoto., Nina S. T. Hirata., Alexandre X. Falcão., and Alexandru C. Telea. Improving neural network-based multidimensional projections. In *Proceedings of the 15th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, pages 29–41, 2020.

[EHH12] Daniel Engel, Lars Hüttenberger, and Bernd Hamann. A Survey of Dimension Reduction Methods for High-dimensional Data Analysis and Visualization. In *Proceedings of the Visualization of Large and Unstructured Data Sets: Applications in Geospatial Planning, Modeling and Engineering - IRTG 1131 Workshop 2011*, volume 27 of *OpenAccess Series in Informatics*, pages 135–149, 2012.

[EHT20] Mateus Espadoto, Nina Sumiko Tomita Hirata, and Alexandru C Telea. Deep learning multidimensional projections. *Information Visualization*, 19(3):247–269, 2020.

[EKSX96] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, page 226–231. AAAI Press, 1996.

[Ell18] Geoffrey Ellis. *Cognitive biases in visualizations*. Springer, 2018.

[EMH19] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural architecture search: A survey. *Journal of Machine Learning Research*, 20(55):1–21, 2019.

[EMK+21] Mateus Espadoto, Rafael M. Martins, Andreas Kerren, Nina S. T. Hirata, and Alexandru C. Telea. Toward a quantitative survey of dimension reduction techniques. *IEEE Transactions on Visualization and Computer Graphics*, 27(3):2153–2173, 2021.

[ERH+19] Mateus Espadoto, Francisco Caio Maia Rodrigues, Nina S. T. Hirata, Roberto Hirata Jr., and Alexandru C. Telea. Deep Learning Inverse Multidimensional Projections. In Tatiana von Landesberger and Cagatay Turkay, editors, *EuroVis Workshop on Visual Analytics (EuroVA)*. The Eurographics Association, 2019.

[Eva00] Stephen JW Evans. Pharmacovigilance: a science or fielding emergencies? *Statistics in medicine*, 19(23):3199–3209, 2000.

[EW19] Ezekiel J Emanuel and Robert M Wachter. Artificial intelligence in health care: will the value match the hype? *Jama*, 321(23):2281–2282, 2019.

[FG20] Bob Frisch and Cary Greene. What it takes to run a great virtual meeting, 2020.

[FGS18] Rebecca Faust, David Glickenstein, and Carlos Scheidegger. Dimreader: Axis lines that explain non-linear projections. *IEEE transactions on visualization and computer graphics*, 25(1):481–490, 2018.

[FH89] Evelyn Fix and J. L. Hodges. Discriminatory analysis. nonparametric discrimination: Consistency properties. *International Statistical Review / Revue Internationale de Statistique*, 57(3):238–247, 1989.

[FKM19] Takanori Fujiwara, Oh-Hyun Kwon, and Kwan-Liu Ma. Supporting analysis of dimensionality reduction results with contrastive learning. *IEEE transactions on visualization and computer graphics*, 26(1):45–55, 2019.

[FL03] M. C. Ferreira de Oliveira and H. Levkowitz. From visual data exploration to visual data mining: a survey. *IEEE Transactions on Visualization and Computer Graphics*, 9(3):378–394, 2003.

[FMF13] Michael Friendly, Georges Monette, and John Fox. Elliptical insights: understanding statistical methods through elliptical geometry. *Statistical Science*, 28(1):1–39, 2013.

[FS19] Jason Furman and Robert Seamans. Ai and the economy. *Innovation policy and the economy*, 19(1):161–191, 2019.

[FSB+23] Felix Friedrich, Patrick Schramowski, Manuel Brack, Lukas Struppek, Dominik Hintersdorf, Sasha Luccioni, and Kristian Kersting. Fair diffusion: Instructing text-to-image generation models on fairness. *arXiv preprint arXiv:2302.10893*, 2023.

[FWZM21] Takanori Fujiwara, Xinhai Wei, Jian Zhao, and Kwan-Liu Ma. Interactive dimensionality reduction for comparative analysis. *IEEE Transactions on Visualization and Computer Graphics*, 28(1):758–768, 2021.

[GBC17] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2017.

[GKW+08] Alexander N Gorban, Balázs Kégl, Donald C Wunsch, Andrei Y Zinovyev, et al. *Principal manifolds for data visualization and dimension reduction*, volume 58. Springer, 2008.

[GTdS+18] R. Garcia, A. Telea, B. da Silva, J. Torresen, and J. Comba. A task-and-technique centered survey on visual analytics for deep learning model engineering. *Computers and Graphics*, 77:30–49, 2018.

[GZL+20] L. Gou, L. Zou, N. Li, M. Hofmann, A. K. Shekar, A. Wendt, and L. Ren. Vatld: A visual analytics system to assess, understand and improve traffic light detection. *IEEE Transactions on Visualization and Computer Graphics*, pages 1–1, 2020.

[Har75] J.A. Hartigan. Printer graphics for clustering. *Journal of Statistical Computation and Simulation*, 4(3):187–213, 1975.

[HB21] Aspen Hopkins and Serena Booth. Machine learning practices outside big tech: How resource constraints challenge responsible development. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, pages 134–145, 2021.

[HE11] Christopher Healey and James Enns. Attention and visual memory in visualization and computer graphics. *IEEE transactions on visualization and computer graphics*, 18(7):1170–1188, 2011.

[HEB+02] Joan Harvey, George Erdos, Helen Bolam, Michael AA Cox, John NP Kennedy, and David T Gregory. An analysis of safety culture attitudes in a highly regulated environment. *Work & stress*, 16(1):18–36, 2002.

[HFA17] Nicolas Heulot, Jean-Daniel Fekete, and Michael Aupetit. Visualizing dimensionality reduction artifacts: An evaluation, 2017.

[HG02] P. Hoffman and G. Grinstein. A survey of visualizations for high-dimensional data mining. *Information Visualization in Data Mining and Knowledge Discovery*, 104:47–82, 2002.

[HGM+97] P. Hoffman, G. Grinstein, K. Marx, I. Grosse, and E. Stanley. Dna visual and analytic data mining. In *Proceedings. Visualization '97 (Cat. No. 97CB36155)*, pages 437–441, 1997.

[HHB20] Sungsoo Ray Hong, Jessica Hullman, and Enrico Bertini. Human factors in model interpretability: Industry practices, challenges, and needs. *Proceedings of the ACM on Human-Computer Interaction*, 4(CSCW1):1–26, 2020.

[HHC+19] Fred Hohman, Andrew Head, Rich Caruana, Robert DeLine, and Steven M Drucker. Gamut: A design probe to understand how data scientists understand machine learning models. In *Proceedings of the 2019 CHI conference on human factors in computing systems*, pages 1–13, 2019.

[HHW22] Nathaniel Helwig and Elizabeth Hsiao-Wecksler. Multivariate Gait Data. UCI Machine Learning Repository, 2022.

[HLD02] Helwig Hauser, Florian Ledermann, and Helmut Doleisch. Angular brushing of extended parallel coordinates. In *IEEE Symposium on Information Visualization, 2002. INFOVIS 2002.*, pages 127–130. IEEE, 2002.

[HLT19] Fengxiang He, Tongliang Liu, and Dacheng Tao. Control batch size and learning rate to generalize well: Theoretical and empirical evidence. In *Proceedings of the Advances in Neural Information Processing Systems*, volume 32, 2019.

[HMKL18] Robert R Hoffman, Shane T Mueller, Gary Klein, and Jordan Litman. Metrics for explainable ai: Challenges and prospects. *arXiv preprint arXiv:1812.04608*, 2018.

[HMP+17] I. Higgins, Loïc Matthey, A. Pal, C. Burgess, Xavier Glorot, M. Botvinick, S. Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *ICLR*, 2017.

[HMZA21] Xiaoshui Huang, Guofeng Mei, Jian Zhang, and Rana Abbas. A comprehensive survey on point cloud registration, 2021.

[HR03] Geoffrey E Hinton and Sam Roweis. Stochastic neighbor embedding. In *Proceedings of the Advances in Neural Information Processing Systems*, volume 15, 2003.

[HS06] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[HSD19] Fred Hohman, Arjun Srinivasan, and Steven M Drucker. Telegam: Combining visualization and verbalization for interpretable machine learning. In *2019 IEEE Visualization Conference (VIS)*, pages 151–155. IEEE, 2019.

[HSMHW16] Nathaniel E Helwig, K Alex Shorter, Ping Ma, and Elizabeth T Hsiao-Wecksler. Smoothing spline analysis of variance models: A new tool for the analysis of cyclic biomechanical data. *Journal of biomechanics*, 49(14):3216–3222, 2016.

[Hun07] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing In Science & Engineering*, 9(3):90–95, 2007.

[HW13] Julian Heinrich and Daniel Weiskopf. State of the art of parallel coordinates. *Eurographics (State of the Art Reports)*, pages 95–116, 2013.

[Ins85] Alfred Inselberg. The plane with parallel coordinates. *The Visual Computer*, 1(2):69–91, Aug 1985.

[IS15] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning*, page 448–456. JMLR.org, 2015.

[JAL+22] Hyeon Jeon, Michael Aupetit, Soohyun Lee, Hyung-Kwon Ko, Youngtaek Kim, and Jinwook Seo. Distortion-aware brushing for interactive cluster analysis in multidimensional projections. *arXiv preprint arXiv:2201.06379*, 2022.

[JCC+11] Paulo Joia, Danilo Coimbra, Jose A. Cuminato, Fernando V. Paulovich, and Luis G. Nonato. Local affine multidimensional projection. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2563–2571, 2011.

[JJZ+17] Fei Jiang, Yong Jiang, Hui Zhi, Yi Dong, Hao Li, Sufeng Ma, Yilong Wang, Qiang Dong, Haipeng Shen, and Yongjun Wang. Artificial intelligence in healthcare: past, present and future. *Stroke and Vascular Neurology*, 2(4):230–243, 2017.

[JZF+09] Dong Hyun Jeong, Caroline Ziemkiewicz, Brian Fisher, William Ribarsky, and Remco Chang. iPCA: An interactive system for pca-based visual analytics. *Computer Graphics Forum*, 28(3):767–774, 2009.

[KAL+21] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. *Advances in Neural Information Processing Systems*, 34:852–863, 2021.

[Kan00] Eser Kandogan. Star coordinates: A multi-dimensional visualization technique with uniform treatment of dimensions. In *Proceedings of the IEEE information visualization symposium*, volume 650, page 22. Citeseer, 2000.

[KB97] R. Kelley Pace and Ronald Barry. Sparse spatial autoregressions. *Statistics & Probability Letters*, 33(3):291–297, 1997.

[KB17] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[KB19] Dmitry Kobak and Philipp Berens. The art of using t-sne for single-cell transcriptomics. *Nature communications*, 10(1):5416, 2019.

[Kes17] Jason Kessler. Scattertext: a browser-based tool for visualizing how corpora differ. In *Proceedings of ACL 2017, System Demonstrations*, pages 85–90, Vancouver, Canada, 2017. Association for Computational Linguistics.

[KM12] Nikolaus Kriegeskorte and Marieke Mur. Inverse mds: Inferring dissimilarity structure from multiple item arrangements. *Frontiers in Psychology*, 3:245, 2012.

[KM13] Robert Kosara and Jock Mackinlay. Storytelling: The next step for visualization. *Computer*, 46(5):44–50, 2013.

[KM19] Hyunjik Kim and Andriy Mnih. Disentangling by factorising, 2019.

[KM20a] Oh-Hyun Kwon and Kwan-Liu Ma. A deep generative model for graph layout. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):665–675, 2020.

[KM20b] Oh Hyun Kwon and Kwan Liu Ma. A deep generative model for graph layout. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):665–675, 1 2020.

[KMN+17] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima, 2017.

[KPHH12] Sean Kandel, Andreas Paepcke, Joseph M. Hellerstein, and Jeffrey Heer. Enterprise data analysis and visualization: An interview study. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2917–2926, 2012.

[KPHL17] Matt J Kusner, Brooks Paige, and José Miguel Hernández-Lobato. Grammar variational autoencoder. In *International Conference on Machine Learning*, pages 1945–1954. PMLR, 2017.

[KW78] Joseph B Kruskal and Myron Wish. *Multidimensional scaling*, volume 11. Sage, 1978.

[LA11] Sylvain Lespinats and Michaël Aupetit. Checkviz: Sanity check and topological clues for linear and non-linear mappings. *Computer Graphics Forum*, 30(1):113–125, 2011.

[Lam08] Heidi Lam. A framework of interaction costs in information visualization. *IEEE transactions on visualization and computer graphics*, 14(6):1149–1156, 2008.

[LCB10] Yann LeCun, Corinna Cortes, and CJ Burges. MNIST handwritten digit database. *ATT Labs [Online]*, 2, 2010.

[LCYH17] Wentian Li, Jane E. Cerise, Yaning Yang, and Henry Han. Application of t-sne to human genetic data. *Journal of bioinformatics and computational biology*, 15 4:1750017, 2017.

[LH14] Zhicheng Liu and Jeffrey Heer. The effects of interactive latency on exploratory visual analysis. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2122–2131, 2014.

[Lip18] Zachary C Lipton. The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 16(3):31–57, 2018.

[LJLH19] Yang Liu, Eunice Jun, Qisheng Li, and Jeffrey Heer. Latent space cartography: Visual analysis of vector space embeddings. In *Computer Graphics Forum*, volume 38, pages 67–78. Wiley Online Library, 2019.

[LMW+17] S. Liu, D. Maljovec, B. Wang, P. Bremer, and V. Pascucci. Visualizing high-dimensional data: Advances in the past decade. *IEEE Transactions on Visualization and Computer Graphics*, 23(3):1249–1268, 2017.

[LT13] Dirk J Lehmann and Holger Theisel. Orthographic star coordinates. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2615–2624, 2013.

[LV09] John A. Lee and Michel Verleysen. Quality assessment of dimensionality reduction: Rank-based criteria. *Neurocomputing*, 72(7):1431–1443, 2009. Advances in Machine Learning and Computational Intelligence.

[LWCC18] Hongsen Liao, Yingcai Wu, Li Chen, and Wei Chen. Cluster-based visual abstraction for multivariate scatterplots. *IEEE Transactions on Visualization and Computer Graphics*, 24(9):2531–2545, 2018.

[LWW90] Jeffrey LeBlanc, Matthew O. Ward, and Norman Wittels. Exploring n-dimensional databases. In *Proceedings of the 1st Conference on Visualization '90*, VIS '90, page 230–237, Washington, DC, USA, 1990. IEEE Computer Society Press.

[MAR+24] Brian Montambault, Gabriel Appleby, Jen Rogers, Camelia D. Brumar, Mingwei Li, and Remco Chang. Dimbridge: Interactive explanation of visual patterns in dimensionality reductions with predicate logic, 2024.

[MBBC22] Brian Montambault, Camelia D. Brumar, Michael Behrisch, and Remco Chang. Pixal: Anomaly reasoning with visual analytics, 2022.

[MCMT14] Rafael Messias Martins, Danilo Barbosa Coimbra, Rosane Minghim, and A.C. Telea. Visual analysis of dimensionality reduction quality for parameterized projections. *Computers & Graphics*, 41:26–42, 2014.

[MFNP13] G. M. H. Mamani, F. M. Fatore, L. G. Nonato, and F. V. Paulovich. User-driven feature space transformation. *Computer Graphics Forum*, 32(3pt3):291–299, 2013.

[MHSG18] Leland McInnes, John Healy, Nathaniel Saul, and Lukas Grossberger. Umap: Uniform manifold approximation and projection. *Journal of Open Source Software*, 3(29):861, 2018.

[MHT18] F. C. M. Rodrigues, R. Hirata, and A. C. Telea. Image-based visualization of classifier decision boundaries. In *2018 31st SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*, pages 353–360, 2018.

[MJEG21] Wilson E. Marcílio-Jr, Danilo M. Eler, and Rogério E. Garcia. Contrastive analysis for scatterplot-based representations of dimensionality reduction. *Computers & Graphics*, 101:46–58, 2021.

[ML18] Dominic Masters and Carlo Luschi. Revisiting small batch training for deep neural networks, 2018.

[MLGH13] Bassam Mokbel, Wouter Lueks, Andrej Gisbrecht, and Barbara Hammer. Visualizing the quality of dimensionality reduction. *Neurocomputing*, 112:109–123, 2013. Advances in artificial neural networks, machine learning, and computational intelligence.

[MMKM98] Kathleen M MacQueen, Eleanor McLellan, Kelly Kay, and Bobby Milstein. Codebook development for team-based qualitative analysis. *Cam Journal*, 10(2):31–36, 1998.

[MMT15a] R. Martins, R. Minghim, and A. C. Telea. Explaining neighborhood preservation for multidimensional projections. In *Proc. CGVC*, pages 121–128. Eurographics, 2015.

[MMT15b] Rafael Messias Martins, Rosane Minghim, and Alexandru C. Telea. Explaining Neighborhood Preservation for Multidimensional Projections. In Rita Borgo and Cagatay Turkay, editors, *Computer Graphics and Visual Computing*. The Eurographics Association, 2015.

[MNA+21] Fatemah Mukadum, Quan Nguyen, Daniel M. Adrion, Gabriel Appleby, Rui Chen, Haley Dang, Remco Chang, Roman Garnett, and Steven A. Lopez. Efficient discovery of visible light-activated azoarene photoswitches with long half-lives using active search. *Journal of Chemical Information and Modeling*, 61(11):5524–5534, 11 2021.

[MRC+19] Abigail Mosca, Shannon Robinson, Meredith Clarke, Rebecca Redelmeier, Sebastian Coates, Dylan Cashman, and Remco Chang. Defining an Analysis: A Study of Client-Facing Data Scientists. In Jimmy Johansson, Filip Sadlo, and G. Elisabeta Marai, editors, *EuroVis 2019 - Short Papers*. The Eurographics Association, 2019.

[MWV15] M. A. Migut, M. Worring, and C. J. Veenman. Visualizing multidimensional decision boundaries in 2D. *Data Mining and Knowledge Discovery*, 29(1):273–295, January 2015.

[MZR18] Sina Mohseni, Niloofar Zarei, and Eric D Ragan. A survey of evaluation methods and measures for interpretable machine learning. *arXiv preprint arXiv:1811.11839*, 1, 2018.

[NA19] Luis Gustavo Nonato and Michaël Aupetit. Multidimensional projection for visual analytics: Linking techniques with distortions, tasks, and layout enrichment. *IEEE Transactions on Visualization and Computer Graphics*, 25(8):2650–2673, 2019.

[NDL+23] Vasileios Nittas, Paola Daniore, Constantin Landers, Felix Gille, Julia Amann, Shannon Hubbs, Milo Alan Puhan, Effy Vayena, and Alessandro Blasimme. Beyond high hopes: A scoping review of the 2019–2021 scientific discourse on machine learning in medical imaging. *PLOS Digital Health*, 2(1):e0000189, 2023.

[NH10] Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, page 807–814, 2010.

[NNG+23] Andrew Novick, Quan Nguyen, Roman Garnett, Eric Toberer, and Vladan Stevanović. Simulating high-entropy alloys at finite temperatures: An uncertainty-based approach. *Physical Review Materials*, 7(6):063801, 2023.

[OBL+19] Tom O'Malley, Elie Bursztein, James Long, François Chollet, Haifeng Jin, Luca Invernizzi, et al. Kerastuner. https://github.com/keras-team/keras-tuner, 2019.

[Pea01] Karl Pearson. LIII. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.

[PEP+11] F.V. Paulovich, D.M. Eler, J. Poco, C.P. Botha, R. Minghim, and L.G. Nonato. Piece wise laplacian-based projection for interactive data exploration and organization. *Computer Graphics Forum*, 30(3):1091–1100, 2011.

[PGM+19] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala.

Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[PJ18] Samir Passi and Steven J Jackson. Trust in data science: Collaboration, translation, and accountability in corporate data science projects. *Proceedings of the ACM on Human-Computer Interaction*, 2(CSCW):1–28, 2018.

[PM06] F. V. Paulovich and R. Minghim. Text map explorer: a tool to create and explore document maps. In *Proc. IEEE IV*, pages 245–251, 2006.

[PNML08a] F. V. Paulovich, L. G. 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*, 14(3):564–575, 2008.

[PNML08b] Fernando V. Paulovich, Luis G. Nonato, Rosane Minghim, and Haim Levkowitz. Least square projection: A fast high-precision multidimensional projection technique and its application to document mapping. *IEEE Transactions on Visualization and Computer Graphics*, 14(3):564–575, 2008.

[PSM14] Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543, October 2014.

[PSN10] Fernando V. Paulovich, Claudio T. Silva, and Luis G. Nonato. Two-phase mapping for projecting massive data sets. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1281–1290, 2010.

[PT19] Lucas de Carvalho Pagliosa and Alexandru C Telea. Radviz++: Improvements on radial-based visualizations. In *Informatics*, volume 6, page 16. Multidisciplinary Digital Publishing Institute, 2019.

[RC18] Blagoj Ristevski and Ming Chen. Big data analytics in medicine and healthcare. *Journal of Integrative Bioinformatics*, 15(3):20170030, 2018.

[RCC+22] Cynthia Rudin, Chaofan Chen, Zhi Chen, Haiyang Huang, Lesia Semenova, and Chudi Zhong. Interpretable machine learning: Fundamental principles and 10 grand challenges. *Statistic Surveys*, 16:1–85, 2022.

[REHT19] Francisco C. M. Rodrigues, Mateus Espadoto, Roberto Hirata, and Alexandru C. Telea. Constructing and visualizing high-quality classifier decision boundary maps. *Information*, 10(9):280, Sep 2019.

[RFaT16] Paulo E. Rauber, Alexandre X. Falcão, and Alexandru C. Telea. Visualizing time-dependent data using dynamic t-sne. In *Proceedings of the Eurographics / IEEE VGTC Conference on Visualization: Short Papers*, EuroVis '16, page 73–77, Goslar, DEU, 2016.

[RKH+21] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[RKK+20] S. Ransbotham, S. Khodabandeh, D. Kiron, F. Candelon, M. Chu, and B. LaFountain. Expanding ai's impact with organizational learning, 2020.

[RLS+18] Richard C Roberts, Robert S Laramee, Gary A Smith, Paul Brookes, and Tony D'Cruze. Smart brushing for parallel coordinates. *IEEE*

transactions on visualization and computer graphics, 25(3):1575–1590, 2018.

[RPN20] Sebastian Raschka, Joshua Patterson, and Corey Nolet. Machine learning in python: Main developments and technology trends in data science, machine learning, and artificial intelligence. *arXiv preprint arXiv:2002.04803*, 2020.

[RS00] Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

[RSRDS15] Manuel Rubio-Sánchez, Laura Raya, Francisco Diaz, and Alberto Sanchez. A comparative study between radviz and star coordinates. *IEEE transactions on visualization and computer graphics*, 22(1):619–628, 2015.

[SA15] M. Sedlmair and M. Aupetit. Data-driven evaluation of visual quality measures. *Computer Graphics Forum*, 34(3):201–210, 2015.

[SAA+23] Ashley Suh, Gabriel Appleby, Erik W. Anderson, Luca Finelli, Remco Chang, and Dylan Cashman. Are metrics enough? guidelines for communicating and visualizing predictive models to subject matter experts. *IEEE Transactions on Visualization and Computer Graphics*, pages 1–16, 2023.

[Sch18] Carsten Schelp. An alternative way to plot the covariance ellipse. https://carstenschelp.github.io/2018/09/14/Plot_Confidence_Ellipse_001.html, 2018. Accessed: 2022-03-24.

[SDMT15] Julian Stahnke, Marian Dörk, Boris Müller, and Andreas 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.

[SG17] Alper Sarikaya and Michael Gleicher. Scatterplots: Tasks, data, and designs. *IEEE transactions on visualization and computer graphics*, 24(1):402–412, 2017.

[SGH15] Alexander Schulz, Andrej Gisbrecht, and Barbara Hammer. Using Discriminative Dimensionality Reduction to Visualize Classifiers. *Neural Processing Letters*, 42(1):27–54, August 2015.

[SGNS21] Harini Suresh, Steven R Gomez, Kevin K Nam, and Arvind Satyanarayan. Beyond expertise and roles: A framework to characterize the stakeholders of interpretable machine learning and their needs. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–16, 2021.

[SH05] Chao Shao and Houkuan Huang. Selection of the optimal parameter value for the isomap algorithm. In *Proceedings of the 4th Mexican International Conference on Advances in Artificial Intelligence*, MICAI'05, page 396–404, 2005.

[SH10] Edward Segel and Jeffrey Heer. Narrative visualization: Telling stories with data. *IEEE transactions on visualization and computer graphics*, 16(6):1139–1148, 2010.

[SHGF16] Danielle Albers Szafir, Steve Haroz, Michael Gleicher, and Steven Franconeri. Four types of ensemble coding in data visualizations. *Journal of vision*, 16(5):11–11, 2016.

[SHK+14] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014.

[Sie20] Carson Sievert. *Interactive web-based data visualization with R, plotly, and shiny*. CRC Press, 2020.

[SKGD18] Thilo Spinner, Jonas Körner, Jochen Görtler, and Oliver Deussen. Towards an interpretable latent space: an intuitive comparison of autoencoders with variational autoencoders. In *Proceedings of the Workshop on Visualization for AI Explainability 2018 (VISxAI)*, 2018.

[SKH+21] Nithya Sambasivan, Shivani Kapania, Hannah Highfill, Diana Akrong, Praveen Paritosh, and Lora M Aroyo. "everyone wants to do the model work, not the data work": Data cascades in high-stakes ai. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, pages 1–15, New York, NY, USA, 2021. Association for Computing Machinery.

[SKKC18] Dominik Sacha, Matthias Kraus, Daniel A Keim, and Min Chen. Vis4ml: An ontology for visual analytics assisted machine learning. *IEEE transactions on visualization and computer graphics*, 25(1):385–395, 2018.

[SL] ScikitLearn.org. Classifier comparison. https://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html. Retrieved April 30, 2020.

[SLGS22] Harini Suresh, Kathleen M Lewis, John Guttag, and Arvind Satyanarayan. Intuitively assessing ml model reliability through example-based explanations and editing model inputs. In *27th International Conference on Intelligent User Interfaces*, IUI '22, page 767–781, New York, NY, USA, 2022. Association for Computing Machinery.

[SLZ12] Andre Stuhlsatz, Jens Lippel, and Thomas Zielke. Feature extraction with deep neural networks by a generalized discriminant analysis. *IEEE Transactions on Neural Networks and Learning Systems*, 23(4):596–608, 2012.

[SMB19] Nigam H Shah, Arnold Milstein, and Steven C Bagley. Making machine learning models clinically useful. *Jama*, 322(14):1351–1352, 2019.

[Smi12] Mark Smiciklas. *The power of infographics: Using pictures to communicate and connect with your audiences*. Que Publishing, 2012.

[SMT13] M. Sedlmair, T. Munzner, and M. Tory. Empirical guidance on scatterplot and dimension reduction technique choices. *IEEE TVCG*, pages 2634–2643, 2013.

[SPN12] C. T. Silva, F. V. Paulovich, and L. G. Nonato. User-centered multidimensional projection techniques. *Computing in Science Engineering*, 14(4):74–81, 2012.

[SRS20] Smadar Shilo, Hagai Rossman, and Eran Segal. Axes of a revolution: challenges and promises of big data in healthcare. *Nature medicine*, 26(1):29–38, 2020.

[SSC20] Martin G Seneviratne, Nigam H Shah, and Larry Chu. Bridging the implementation gap of machine learning in healthcare. *BMJ Innovations*, 6(2):45–47, 2020.

[SSJ+21] Jan-Tobias Sohns, Michaela Schmitt, Fabian Jirasek, Hans Hasse, and Heike Leitte. Attribute-based explanation of non-linear embeddings of high-dimensional data. *IEEE Transactions on Visualization and Computer Graphics*, 28(1):540–550, 2021.

[SSK10] Christin Seifert, Vedran Sabol, and Wolfgang Kienreich. Stress maps: Analysing local phenomena in dimensionality reduction based visualisations. In *EuroVAST@ EuroVis*, 2010.

[ST02] Vin de Silva and Joshua B. Tenenbaum. Global versus local methods in nonlinear dimensionality reduction. In *Proceedings of the 15th*

International Conference on Neural Information Processing Systems, NIPS'02, page 721–728, 2002.

[SVM14] C. O. S. Sorzano, J. Vargas, and A. Pascual Montano. A survey of dimensionality reduction techniques, 2014.

[SZJ+21] Guodao Sun, Sujia Zhu, Qi Jiang, Wang Xia, and Ronghua Liang. Evosets: tracking the sensitivity of dimensionality reduction results across subspaces. *IEEE Transactions on Big Data*, 8(6):1566–1579, 2021.

[SZS+16] Dominik Sacha, Leishi Zhang, Michael Sedlmair, John A Lee, Jaakko Peltonen, Daniel Weiskopf, Stephen C North, and Daniel A Keim. Visual interaction with dimensionality reduction: A structured literature analysis. *IEEE transactions on visualization and computer graphics*, 23(1):241–250, 2016.

[Tay] Petroc Taylor. Data growth worldwide 2010-2025. https://www.statista.com/statistics/871513/worldwide-data-created/. Accessed: 2024-3-1.

[TBB+10] Andrada Tatu, Peter Bak, Enrico Bertini, Daniel Keim, and Joern Schneidewind. Visual quality metrics and human perception: An initial study on 2d projections of large multidimensional data. In *Proceedings of the International Conference on Advanced Visual Interfaces*, page 49–56, 2010.

[TCL+13] Gary K.L. Tam, Zhi-Quan Cheng, Yu-Kun Lai, Frank C. Langbein, Yonghuai Liu, David Marshall, Ralph R. Martin, Xian-Fang Sun, and Paul L. Rosin. Registration of 3d point clouds and meshes: A survey from rigid to nonrigid. *IEEE Transactions on Visualization and Computer Graphics*, 19(7):1199–1217, 2013.

[Uly16] Dmitry Ulyanov. Multicore-tsne. https://github.com/DmitryUlyanov/Multicore-TSNE, 2016.

[VBF+23] Helena Vasconcelos, Gagan Bansal, Adam Fourney, Q Vera Liao, and Jennifer Wortman Vaughan. Generation probabilities are not enough: Exploring the effectiveness of uncertainty highlighting in ai-powered code completions. *arXiv preprint arXiv:2302.07248*, 2023.

[vdMH08] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008.

[vdMPvdH09] Laurens van der Maaten, Eric O. Postma, and Jaap van den Herik. Dimensionality reduction: A comparative review. 2009.

[VGS+20] E. F. Vernier, R. Garcia, I. P. da Silva, J. L. D. Comba, and A. C. Telea. Quantitative evaluation of time-dependent multidimensional projection techniques. *Computer Graphics Forum*, 39(3):241–252, 2020.

[VK06] J. Venna and S. Kaski. Visualizing gene interaction graphs with local multidimensional scaling. In *Proceedings of the 14th European Symposium on Artificial Neural Networks*, pages 557–562, 2006.

[VPN+10] J. Venna, J. Peltonen, K. Nybo, H. Aidos, and S. Kaski. Information retrieval perspective to nonlinear dimensionality reduction for data visualization. *JMLR*, 11:451–490, 2010.

[WAG05] Leland Wilkinson, Anushka Anand, and Robert Grossman. Graph-theoretic scagnostics. In *Information Visualization, IEEE Symposium on*, pages 21–21. IEEE Computer Society, 2005.

[WCR+17] John Wenskovitch, Ian Crandell, Naren Ramakrishnan, Leanna House, and Chris North. Towards a systematic combination of dimension reduction and clustering in visual analytics. *IEEE transactions on visualization and computer graphics*, 24(1):131–141, 2017.

[Web18] Sarah Webb. Deep learning for biology. *Nature*, 554(7690):555–558, 2018.

[WFC+18] Yunhai Wang, Kang Feng, Xiaowei Chu, Jian Zhang, Chi-Wing Fu, Michael Sedlmair, Xiaohui Yu, and Baoquan Chen. A perception-driven approach to supervised dimensionality reduction for visualization. *IEEE Transactions on Visualization and Computer Graphics*, 24(5):1828–1840, 2018.

[WFG+12] Johan Wagemans, Jacob Feldman, Sergei Gepshtein, Ruth Kimchi, James R Pomerantz, Peter A Van der Helm, and Cees Van Leeuwen. A century of gestalt psychology in visual perception: Ii. conceptual and theoretical foundations. *Psychological bulletin*, 138(6):1218, 2012.

[WHRS21] Yingfan Wang, Haiyang Huang, Cynthia Rudin, and Yaron Shaposhnik. Understanding how dimension reduction tools work: An empirical approach to deciphering t-sne, umap, trimap, and pacmap for data visualization. *Journal of Machine Learning Research*, 22(201):1–73, 2021.

[WHS20] Yifan Wu, Joseph M Hellerstein, and Arvind Satyanarayan. B2: Bridging code and interactive visualization in computational notebooks. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*, pages 152–165, 2020.

[WvIP19] Hilde JP Weerts, Werner van Ipenburg, and Mykola Pechenizkiy. A human-grounded evaluation of shap for alert processing. *arXiv preprint arXiv:1907.03324*, 2019.

[WVJ16] Martin Wattenberg, Fernanda Viégas, and Ian Johnson. How to use t-SNE effectively. *Distill*, 2016.

[WYAL19] Danding Wang, Qian Yang, Ashraf Abdul, and Brian Y Lim. Designing theory-driven user-centric explainable ai. In *Proceedings of the*

2019 CHI conference on human factors in computing systems, pages 1–15, 2019.

[XHL+22] Jiazhi Xia, Linquan Huang, Weixing Lin, Xin Zhao, Jing Wu, Yang Chen, Ying Zhao, and Wei Chen. Interactive visual cluster analysis by contrastive dimensionality reduction. *IEEE Transactions on Visualization and Computer Graphics*, 29(1):734–744, 2022.

[XRV17] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

[YG11] Callum A Young and Andrew L Goodwin. Applications of pair distribution function methods to contemporary problems in materials chemistry. *Journal of Materials Chemistry*, 21(18):6464–6476, 2011.

[YHSA20] Fumeng Yang, Zhuanyi Huang, Jean Scholtz, and Dustin L Arendt. How do visual explanations foster end users' appropriate trust in machine learning? In *Proceedings of the 25th International Conference on Intelligent User Interfaces*, pages 189–201, 2020.

[Yin07] H. Yin. Nonlinear dimensionality reduction and data visualization: A review. *Intl. Journal of Automation and Computing*, 4(3):294–303, 2007.

[ZFF20] Jian Zhao, Mingming Fan, and Mi Feng. Chartseer: Interactive steering exploratory visual analysis with machine intelligence. *IEEE Transactions on Visualization and Computer Graphics*, pages 1–1, 2020.

[ZGZ+19] Hao Zhu, Bin Guo, Ke Zou, Yongfu Li, Ka-Veng Yuen, Lyudmila Mihaylova, and Henry Leung. A review of point set registration: From pairwise registration to groupwise registration. *Sensors*, 19(5), 2019.

[ZLVV21] Alexandra Zytek, Dongyu Liu, Rhema Vaithianathan, and Kalyan Veeramachaneni. Sibyl: Understanding and addressing the usability challenges of machine learning in high-stakes decision making. *IEEE Transactions on Visualization and Computer Graphics*, 28(1):1161–1171, 2021.