Enhancing Deep Learning with Visual Interactions

ERIC KROKOS, University of Maryland College Park
HSUEH-CHIEN CHENG, University of Maryland College Park
JESSICA CHANG, Department of Defense
BOHDAN NEBESH, Department of Defense
CELESTE LYN PAUL, Department of Defense
KIRSTEN WHITLEY, Department of Defense
AMITABH VARSHNEY, University of Maryland College Park

Deep learning has emerged as a powerful tool for feature-driven labeling of datasets. However, for it to be effective, it requires a large and finely-labeled training dataset. Precisely labeling a large training dataset is expensive, time consuming, and error-prone. In this paper, we present a visually-driven deep learning approach that starts with a coarsely-labeled training dataset, and iteratively refines the labeling through intuitive interactions that leverage the latent structures of the dataset. Our approach can be used to (a) alleviate the burden of intensive manual labeling that captures the fine nuances in a high-dimensional dataset by simple visual interactions, (b) replace a complicated (and therefore difficult to design) labeling algorithm by a simpler (but coarse) labeling algorithm supplemented by user interaction to refine the labeling, or (c) use low-dimensional features (such as the RGB colors) for coarse labeling and turn to higher-dimensional latent structures, that are progressively revealed by deep learning, for fine labeling. We validate our approach through use cases on three high-dimensional datasets and a user study.

CCS Concepts: •**Human-centered computing** → *Interaction design; Visualization;*

General Terms: Design, Human factors, Algorithms

Additional Key Words and Phrases: deep learning, dimensionality reduction, knowledge discovery, semantic interaction, visual interaction

ACM Reference Format:
Eric Krokos, Hsueh-Chien Cheng, Jessica Chang, Bruce Crabill, Bohdan Nebesh, Celeste Lyn Paul, Karl Reuss, Tripti Sinha, Kirsten Whitley, and Amitabh Varshney, 2017. Enhancing Deep Learning with Visual Interactions *ACM Trans. Interact. Intell. Syst.* 0, 0, Article 0 (June 2017), 28 pages.
DOI: 0000001.0000001

1. INTRODUCTION

Computer-based semantic learning systems have made impressive strides in the last few years, but there remains a striking disparity between the abilities of humans and machines. Current-generation deep-learning systems require thousands of finely-labeled images to train. If a child needed a thick stack of images of cats to learn what a cat looks like, we would be in deep trouble [Matheny 2016]. The goal of this paper is to take the first steps to bridge this gap, by using visual interactions to help enhance the performance of deep learning. Furthering the capabilities of deep learning through interactions can help it emerge as a powerful engine for new discoveries in high-dimensional data.

A challenge with deep learning, as previously mentioned, is that it requires large amounts of precisely annotated and labeled data for training purposes. Recent ad-

Author's addresses: E. Krokos and H. Cheng and A. Varshney, Computer Science Department, University of Maryland College Park; B. Crabil and K. Reuss and T. Sinha, University of Maryland College Park; J. Chang and C. Lyn Paul and B. Nebesh and K. Whitley, Department of Defense.

DOI: 0000001.0000001

vances in data collection and annotation have allowed for the creation of large high-dimensional datasets, consisting of millions of points, which are often used to train deep-learning models. However, in many real-world datasets, the labels and annotations provided may be incomplete or not capture all the distinctions within the data, known and unknown. Precisely annotating a high-dimensional dataset that contains many labels is expensive, time consuming, and error-prone, caused possibly by mislabeling data-points, the unintended omission of precise labeling, or the subjectivity of the annotator(s).

Semi-supervised learning aims to improve classification performance by using labeled and unlabeled examples. In this work we extend semi-supervised-learning to include coarse-labeling that a user can refine based on visual feedback. Since it is often easier and faster to carry out coarse labeling, we expect our approach to be broadly applicable to many more datasets where coming across large amounts of precisely labeled data is difficult.

In this paper, we present our approach to facilitate visually-driven deep learning that enables the refinement of a coarsely-labeled dataset through intuitive interactions by leveraging the latent structures present in high-dimensional datasets. Through the combined efforts of human analysts and deep learning, we hope to not only facilitate discovery of hidden communities and structures within these high-dimensional datasets, but also start to bridge the gap in our understanding of deep learning.

This paper makes the following contributions to interaction-assisted deep-learning for high-dimensional semantic labeling:

— We use permutation-invariant deep neural networks to generate 2D point distributions such that similar points are proximal.
— We iteratively and manually refine the distribution and labeling of data points using visual feedback, which are in-turn used to refine the deep neural networks.
— We facilitate the discovery of detailed latent groups or labels from datasets that only consist of a few high-level or no labels.

2. RELATED WORK

A new focus on bringing human judgment and intuition back into the data exploration process has recently gained popularity. Individually, machines and people are generally good at solving different problems, but the union of their strengths will lead to new and better insights. Our system builds upon the ideas by Turkay *et al.* [2017], who have shown that placing the human into the data analytics process, particularly for high-dimensional data, is beneficial especially when the temporal, perceptual, and cognitive abilities of the user can be leveraged.

We drew inspiration from the work by Endert *et al.* [2012], whose insight was to use a model that is continuously updated with information from user interactions, and of Sacha *et al.* [2017], who have shown that although there exist many tools and techniques for dimensionality reduction, there is a lack of general-purpose tools for human-in-the-loop interactive dimensionality reduction and exploration.

We next present a review of related research, touching on dimensionality reduction, high-dimensional visualization, interactive analysis of high-dimensional data, label generation, active learning, along with comparisons between traditional semi-supervised learning and our goal, label discovery and refinement.

2.1. Dimensionality Reduction

The idea behind dimensionality reduction is to capture the dominant trends in the data and project them to a lower dimension. We use dimensionality-reduction techniques to produce a human comprehensible representation of an input dataset to allow analysts

to visually detect clusters and refine the data labeling. Since the previous work on dimensionality reduction is extensive, we only refer to a small representative set here.

One set of dimensionality reduction techniques is based on Taylor-series expansions. Roweis and Saul [2000] introduced and compared locally linear embedding (LLE) projections of data to those produced by principal component analysis (PCA) and multidimensional scaling (MDS) [Kruskal 1964]. Belkin and Niyogi [2003] introduced Laplacian eigenmaps that generalize the idea of LLE by computing a low-dimensional manifold representation of a high-dimensional data set such that local neighborhood information is preserved in a least-squares sense. The dimensionality reduction by this algorithm produces an approximation that reflects the intrinsic geometric structure of the high-dimensional dataset. The idea of spectral dimensionality reduction has also been used in visual depiction of graph relationships [Koren 2003].

A popular approach, introduced by Maaten and Hinton [2008], is a technique called t-Distributed Stochastic Neighbor Embedding (t-SNE), which is often used to generate visualizations of high-dimensional data in two dimensions. This is done using a modified algorithm based on Stochastic Neighbor Embedding by Taylor *et al.* [2007]. Their approach generates a view that reveals the structure of low-dimensional manifolds within the high-dimensional datasets and visualizes those manifolds using a collection of two-dimensional scatterplots. The algorithm is non-linear, performing different transformations on different regions of the feature space, looking for similarities in these smaller regions. The need to search through and check many different configurations of hyper-parameters just to generate the visualization, such as step and perplexity, can lead to a lot of time being spent simply searching for meaningful and revealing information, while also potentially leading to incorrect interpretations of the underlying data. According to Wattenberg *et al.* [2016], t-SNE does not always produce meaningful representations. For our work, it is crucial that the generated visualizations facilitate intuitive and accurate interpretations of the data.

We next review deep-learning-based dimensionality-reduction techniques. Hinton *et al.* [2006] popularized the idea of using a multilayer neural network to perform dimensionality reduction. Hadsell *et al.* [2006] developed a convolution-based encoder using deep learning to perform dimensionality reduction to create a globally coherent nonlinear function that maps the data evenly on an output space. The advantage of these techniques is that the learning relies only on point-neighborhood relationships and does not require any distance measurements in the high-dimensional input space. More recently, Chen *et al.* [2014] developed a system using deep learning Autoencoders to perform dimensionality reduction on hyper-spectral data. In their approach they combine neighboring pixels to generate spatial feature sets which produce output class labeling after autoencoding and logistic regression. We use a modified version of Hadsell *et al.* [2006] method to enable handling of coarsely labeled data.

Traditional dimensionality reduction and machine learning approaches, such as the ones reviewed above, require fully and accurately labeled data. Further, in contrast to our approach, they are single-shot techniques, with no mechanism to leverage the analyst's domain knowledge or pattern recognition ability to guide, shape, or influence the clustering of the data.

2.2. High-Dimensional Community Visualization

Visualizing high-dimensional data is a core element of our system since our goal is to reveal hidden patterns and communities within. In this section we review a selection of previous recent work on high-dimensional data visualization. A common technique for high-dimensional visualization is multidimensional scaling (MDS), pioneered by Kruskal [1964]. The goal of MDS is to visually group data objects such that similar objects are spatially close to each other and dissimilar data objects are far away, as

determined by a similarity function. The idea of plotting similar objects in close spatial proximity is fundamental to our own work.

One example of using spatial proximity for conveying information is the work by Amir *et al.* [2013] who have created a 2D visualization tool for the analysis of high-dimensional biological-cell data. Each individual cell is represented as a point in a scatter plot, with the positioning of each point calculated using the t-SNE [Maaten and Hinton 2008] algorithm.

One of the most common and easiest-to-interpret visualization techniques is the scatterplot. Dang and Wilkinson [2014] have improved the classical scatter-plots with scagnostics (scatter-plot diagnostics). One of the main problems in using scatter-plots is that as the dimensionality of the dataset increases, the number of generated scatter-plots also increases rapidly. To get around this problem, Dang and Wilkinson developed techniques to enable discovery of hidden structures within a subset of scatter-plots through various transformations.

The work we present here takes inspiration from the previous visualization works mentioned. Our goal is to build upon the foundations of that work and expand the capability of visualization-based investigation by incorporating and interleaving an intelligent system (deep learning) into the exploration process, rather than make an advance in the field of visualization itself.

2.3. Interactive Analysis of High-Dimensional Data

The ability to directly interact with high-dimensional data in order to search for trends or other interesting information is crucial. For our system, we expect a human analyst to understand, interact, and interpret results based on the output of a dimensionality reduction algorithm. In this section we review some of the previous works whose goal was to provide a method to interact in an iterative manner with high-dimensional datasets.

Ip *et al.* [2012] developed a tool that facilitated the visual exploration of hidden spatial structures in volumetric datasets. The tool used a 2D intensity-gradient histogram to enable a user to iteratively search for interesting regions in the 3D volume. The interaction involved manipulating and selecting regions of the generated histogram through normalized cuts. We were inspired by the work done by Ip *et al.* and have also included the usage of normalized cuts to aid in our own segmentation and clustering process.

Chen *et al.* [2017; 2018] developed a system which leverages deep learning to guide user exploration of interesting high-dimensional and temporally changing structures within volumetric cell data. Rather than using the raw features provided by the data, deep learning is able to transform those features into compact and semantically meaningful representations, which better capture and distinguish biological properties, such as boundaries and other components. Using this new feature space, a quantifiable metric of similarity between these deep-learning constructed features may be calculated, simplifying the transfer function used for volume rendering, giving rise to the interesting structures within the original biological data.

Liu *et al.* [2014] developed an application that allows for the interaction and exploration of high-dimensional data presented in low-dimensional space. The primary contribution in their work is the use of distortion-guided manipulations, where a user can select a data-element and then move or delete it, causing point-wise distortion measures to be re-calculated and visualized. As a data-element is moved in the 2D space, a global structural change occurs on the fly, which provides information regarding the relationship between different parts of the 2D projection. Liu *et al.* [2015] further expanded upon the previous work with the inclusion of a subspace view navigation graph. This graph allows for animated transitions between different subspace views to facili-

tate easy comparison between those views. Our work shares a similar goal to Liu *et al.*, exploring high-dimensional data, but their tool is designed for understanding how different variables are related in low-dimensional space, by changing the configurations of the projection, rather than finding new subsets/categories in low-dimensional space. In addition, there is no mechanism in place for influencing the projections based on other input data, such as a newly discovered group or label information.

2.4. Label Generation

For many applications, it is useful to be able to generate labels for various content. For example, these labels could be hashtags for Twitter [Krokos and Samet 2014], keywords for image or scene search [Yee et al. 2003] or for music labeling [Trohidis et al. 2008]. Traditional labeling classification consists of learning from a dataset where each data example is associated with a known single label. For example, if there are only two distinct labels in the dataset, the problem is known as binary classification [Joachims 1998]. In multi-label classification each data example is associated with more than a single label. A common application for multi-label classification is image labeling, online search, and machine learning [Ueda and Saito 2003][Godbole and Sarawagi 2004]. However, the problem this paper tackles is often considered a challenge that many multi-label classification approaches suffer from, which is the lack of large, precisely labeled datasets to train on. Rather, many datasets have missing, or over simplified labels. This is the challenge we tackle in this paper, identifying, differentiating, and re-labeling hidden labels and groups within high-dimensional data.

One way to get around this problem is to pre-train the classification network (in the case of deep learning) in an unsupervised manner [Erhan et al. 2010], without the usage of labels. Erhan *et al.* [2010] have shown that pre-training the classification network allows the layers to focus on and capture the variation and nuances of the data itself, which allows for better regularization and generalization of the network. An example of this is the work done by Hinton and Salakhutdinov [2006], who use an autoencoder to pre-train a deep learning network to learn a low-dimensional embedding of high-dimensional data. The primary purpose of the previous work is to increase the performance of trained models by seeding the weights in the network through pre-training. Instead of modifying the methods behind training networks, we are improving model performance by modifying the training data itself to be more precise and accurate, resulting in a better model. The benefit of our approach is that the result of our technique, an improved training set, can be leveraged by all forms of machine learning which use training data, not just neural networks, as well as other applications where labeled datasets are required.

Lee [2013] developed a technique that uses labeled and unlabeled data to train a deep neural network in a semi-supervised fashion. For the data with unknown labels, a pseudo-label is assigned with the maximum predicted probability after every update cycle, which is then used as if it were the true label. Testing their approach using the MNIST dataset, they were able to develop a deep neural network capable of state-of-the-art classification performance. Similar to the previous work, our system is designed to help discover the true label of the data-point based on the given labeling present in the dataset. As the data points are relabeled, they are placed back into the dataset and are used for re-training the network. However, our technique benefits from the ability of an analyst to use their judgment and domain knowledge to help drive the re-labeling of data, while also benefiting from the suggestions and output of a model. If the model should make a mistake in labeling, a human analyst can step into the loop and correct the mistake, before it propagates and influences the future labeling of data. Lastly, the previously mentioned approach can only assign data points with

a label that already exists in the dataset, and cannot generate new labels or groups, which may be necessary if a large enough portion of the data is without a label.

There has not been much work on including human-in-the-loop interactions to enhance labeling accuracy. Much of the prior work relies on statistical models that are limited by the domain knowledge of the dataset or expert availability. One example of rectifying crowd sourced labeling with expert review is the work by Hung *et al.* [2015]. In their work, they use statistical methods to generate a few meaningful questions about the data, previously labeled by crowd-sourcing, to minimize the amount of expert time needed to correct that labeling. The output of the crowd sourcing and expert refinement of the labeled data may then be used for machine learning. The authors found that they were able to reduce the amount of time needed from domain experts to refine and correct the labeling to achieve near perfect classification. While the impact and usefulness of the previous work cannot be understated, the weakness of the previous method is that there remains a large reliance of crowd sourced information to provide the majority of labels and handle the burden of labeling the data. This may not always be possible, as the domain of the data to be labeled may not be easily understood by the population in general (such as hyper-spectral imagery or computer-network data). The datasets typically targeted for crowd sourcing, and those used in the previous work, are image tagging and sentiment analysis. As the ability of a crowd sourced audience to provide labels reduces, the burden on the expert increases dramatically, and the effectiveness of the previous work diminishes. Our tool is designed to handle high-dimensional and abstract data, that at face value would be daunting to label without additional meta information (such as maps or domain knowledge), and to present that data in an easy-to-interpret-and-manipulate format regardless of the type of data.

2.5. Deep Learning Semi-Supervised Classification

In this section, we cover some of the recent related works which use deep learning for classification tasks. A new approach by Rasmus *et al.* [2015] uses a ladder-network for a classification task, where unsupervised and semi-supervised training methods are combined and used simultaneously to improve the overall training and performance of a model. A ladder network utilizes an autoencoder model, but with additional "skip" connections between the encoder and decoder to transfer details which would otherwise be lost. In their work, the ladder network is treated as both a noisy encoder and a noise decoder, known as a denoising autoencoder (dAE), which also functions as a hierarchical latent variable model. In the dAE, an autoencoder is trained to reconstruct the original observation from a corrupted version, which is compared to a clean version of the original observation run through the encoder. Using this setup, they are able to perform state of the art semi-supervised classification on the MNIST and CIFAR-10 datasets using only a small subset of the data, using only 100 examples and 1000 examples for two different tests, but such that every unique label (type) is present and no single label group is over-represented in the training set. Using a ladder network, the authors demonstrate that they can achieve low test error percentages (high classification accuracy) as compared to previous semi-supervised classification results using extremely small training labeled sets.

Pezeshki *et al.* [2016] conducted an investigation into how the different aspects of a ladder network function, and the influence of different amounts of training data on those components. In their experiments, they removed or reconfigured individual components to learn about their relative importance in the operation of the model. One particularly interesting result reported is that the skip connections between layers become less important as the number of training examples increases, with more emphasis placed on the injection of noise in each layer. The experimental setup included

evaluations using a permutation-invariant MNIST dataset, evaluated on 100 and 1000 training examples, such that every label is equally represented and present. The authors found that using their own configuration of the ladder network, they were able to achieve state of the art classification accuracy.

Our work differs from the previous work in a few ways. The goal of our work is label discovery, to refine the initial coarse labeling provided with a dataset to a more precise labeling, and adding labels which did not previously exist in the dataset. The power of the previous works is their ability to leverage the unsupervised power of deep learning in conjunction with traditional semi-supervised classification. The previous works are able to generate impressive classification accuracy given a small number of training examples, but still require knowledge and examples for each of the unique categories, where our approach does not have this requirement.

2.6. Interactive Intelligent Systems and Active Learning

There have been other systems that combine the abilities of humans and machines to achieve a goal that would be difficult by either alone. The process of continuously updating a model through human interaction, which then produces results for a human to judge and operate with, often called semi-supervised learning, is also known as active learning. The advantage of active learning is that both the machine and human are able to make better decisions as well as continuously update and refine their decisions based on the output from the other. An example of this iterative updating process is the work by Ware *et al.* [2001] who have developed a system for interactively training a machine learning classifier by leveraging the background and domain knowledge of the users. The system used a 2D scatter-plot visualization where two of the many feature attributes selected define the plane axes. By iterating over the different pairs of attributes, the authors found that users were able to create classifiers on par with those of the state of the art by visually partitioning the data and drawing decision boundaries.

A system called CueT by Amershi *et al.* [2011] uses human analysts to train and refine a recommender system for network system triage (linking associated error messages into a common problem). In their work, they build a similarity matrix to group and classify incoming alarms and tickets, which is refined through interactions of the analyst. These recommendations are then sorted and presented at the top of a ticket list and color-coded based on severity. Soto *et al.* [2015] developed a system called ViTA-SSD for presenting and identifying patterns among semi-structured documents for text mining and analysis. This is done by leveraging a learned corpus of important words from the documents along with meta-data to generate document clusters. These 2D clusters are generated using a dimensionality reduction based on a combination of t-Distributed Stochastic Neighbor Embedding (t-SNE) and K-means clustering, which can be refined using a user adjustable distance metric for measuring similarity between documents. These clusters are then visualized using a scatter plot, and can be refined through the selection of particular keywords or documents which an analyst may find useful, to allow further refinement of the documents and clusters relating to the user input to be shown in the following iterations. The ultimate goal is to allow for analysts to discover correlations, trends, and similarities between different sets of documents and topics, such as similarities of documents across different topic domains. The primary limitation of the previously mentioned systems is that they have been engineered to handle a specific or discrete type of data, or to try to help solve a specific kind of problem. In addition, the previous systems are designed to help find correlations or differences between data elements, or to improve the speed with which a user can interact with large amounts of a specific kind of data. While also finding correlations and patterns in high-dimensional data, our system allows for the actual

modification and improvement of a given dataset for future use. Not only does our approach improve the model used within our own tool, it allows for the improvement of any future models trained using the same data.

3. OUR APPROACH

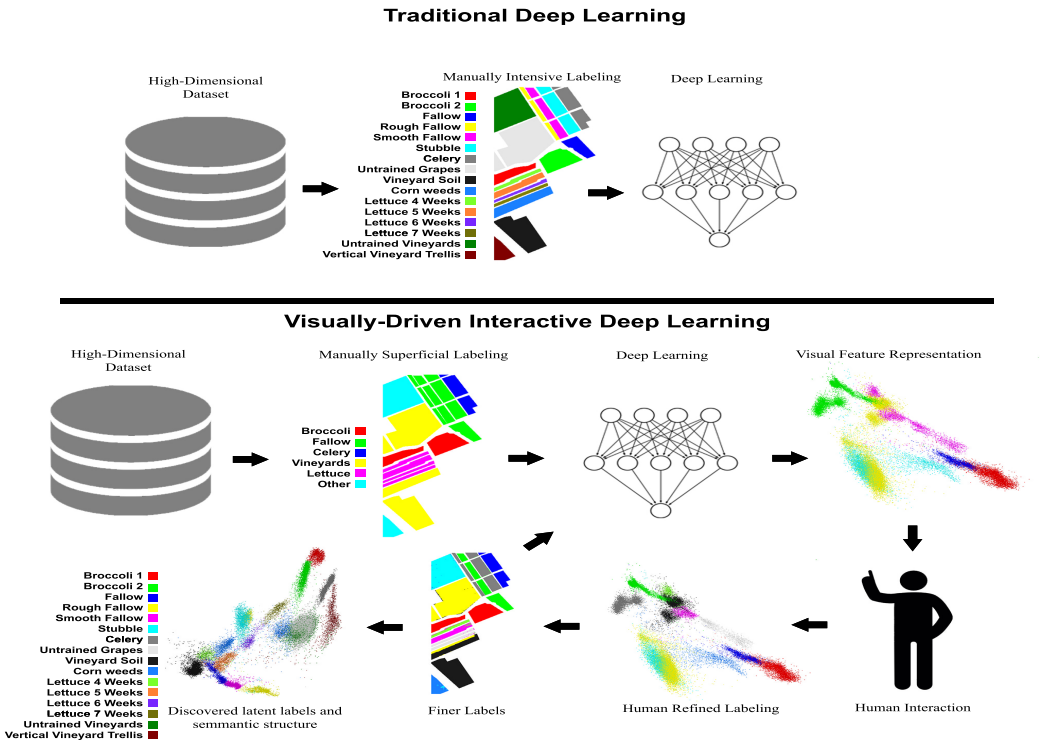


Fig. 1. A brief illustration of the difference between traditional deep learning techniques and our approach. deep learning traditionally requires a large, time-consuming, and precisely labeled dataset for training. For many different reasons, such datasets may be inappropriately labeled. In our approach, we start with coarse labels (that are typically far easier to construct) and then refine them through an iterative process, involving visual interactions and deep learning.

In this paper, we present our approach to facilitate the discovery of latent groups and labels within high-dimensional datasets through spatially meaningful visualizations generated using deep learning. In addition, we also present our technique that allows an analyst to iteratively define and refine the labeling of a dataset to reveal interesting trends and sub-communities. We illustrate our approach with several datasets.

A general overview of our technique as compared to traditional approaches can be seen in Figure 1. The traditional learning paradigm consists of first labeling a large dataset through intensive and tedious work, and then using that constructed dataset to train a model. As stated earlier, this approach can suffer from labeling errors and omissions, which can lead to errors and gaps in classification. In our approach, we take the same datasets along with a simple labeling scheme to train a deep neural network and produce a visually meaningful representation of the dataset and the labels associated with that data. Using that visual representation, an analyst can visually refine the labeling of the data based on patterns and spatial distinctions, which are fed back

into the model to generate a new spatial representation. After a few iterations, our approach can accurately refine the labeling and discover hidden groups within high-dimensional datasets. The end of the iterative process depends on the structure of the data. As the user makes changes to the labeling of various data points, the structure of the presented data changes. As the iterative process continues, the structure changes less and less and eventually converges. Once the structure has converged, the user may conclude the iterative process.

3.1. Point-Distribution Generation

Our goal is to generate a spatially meaningful 2D representation of the data such that data elements with similar features are located close together. To generate the representation, we used Matlab and Python with the deep learning libraries TensorFlow [Abadi et al. 2016] and Keras [Chollet et al. 2015]. Therefore the deep-learning performance reported here is not particularly optimized for time but for flexibility. We developed a Siamese deep neural network inspired by Hadsell *et al.* [2006], which is built ontop of a Variational Autoencoder as inspired by Hinton and Salakhutdinov [2006]. Our visual knowledge discovery program uses 2D and 3D rendering techniques with GPU-accelerated OpenGL and C++. Data processing was performed on a computer with an Intel Xenon 2.6 GHz CPU and a NVIDIA GTX 1080 GPU.

The deep neural network used for dimensionality reduction and revealing hidden clusters is composed of two systems, a Variational Autoencoder, and a Siamese network. For a given dataset, whose labeling may be coarse or absent, the autoencoder first learns a 2D representation based solely on the data. The idea is to let the data itself drive the dimensionality reduction, and to have any emerging groups that come out of the process be based on the features within the data. From this pre-trained representation, the Siamese network then steps in, using the same network structure and weights. Its' goal is to refine that network, by pushing and pulling on these clusters in 2D space, such that similarly labeled groups of points are pulled together, and dissimilar groups are pushed apart. It is this joint process, of leveraging both the labels and features, that is used to generate the 2D distributions. Figure 3 shows the effect of the Siamese network on a distribution of points generated by the Variational Autoencoder.

A visual representation of the network structure and procedure can be seen in Figure 4. The deep neural network structure starts with an input layer with the size of the dimensionality of the input data. From that input layer, that data is passed into a set of three one-dimensional convolutional layers, a 128 filter, 9 feature wide layer, a 64 filter, 6 feature wide layer, and a 64 filter, 3 feature wide layer. All convolutional layers have rectified linear activation functions [Nair and Hinton 2010], also known as Relu, as defined as $\max(0, x)$. The last convolutional layer is then passed into a Flatten layer, which brings the internal data structure back to a 1D representation. From this flattened layer, the data is passed into a Dense layer with two nodes, using the identity activation function $f(x) = x$, which is used to generate the (x, y) coordinates of a given datapoint. This network structure is shared between both the Variational Autoencoder and the Siamese networks.

To update the Variational Autoencoder, reconstruction loss (binary cross-entropy [Hinton and Salakhutdinov 2006]) and Kullback-Leibler (KL) Divergence [1996] are used to refine the network. The Variational Autoencoder is completely unsupervised, relying entirely on the features within the data. The goal is to generate a 2D representation of the data which captures as many of the intrinsic properties of the high-dimensional data, such that from the 2D representation, the original high-dimensional data can be reconstructed accurately. For the Siamese network, two data points are fed into two networks with the same structure and weights, with the weights of the networks updated identically across iterations based on the contrastive loss function.

$$Distance = \sqrt{\sum (x_a - x_b)^2}$$

$$ContrastiveLoss = y * Distance + (1 - y) * \max(0, 1 - Distance)$$

Fig. 2. Equations used in calculating the Constrastive Loss for the Siamese Network

To refine the Siamese network, we use a contrastive loss function using Euclidean distance (as defined in Figure 2), which compares the two output 2D points and checks to see if they belong to the same label, in conjunction with ADADELTA [Zeiler 2012] during training. If the points to the same label, then the loss is the Euclidean distance between them. Otherwise, we adjust the loss to be high (one minus the distance), so that the points are pushed apart. This creates a 2D distribution of points such that points belonging to the same label are spatially distinct to points belonging to different label. We use a batch size of $20,000$ and train for a total of 10 iterations for both the Variational Autoencoder and Siamese networks, values which were chosen as a balance to maintain real-time usability and data distribution integrity. The same network and weights are used and shared across iteration stages. Once the network is finished training, the network is used to generate the 2D distribution of points. After the 2D points are generated, the network weights are saved, to be re-loaded and re-used in the next iteration. For each of the spatial scatter-plot representations, we process, visualize, and allow the user to manipulate the entire dataset.

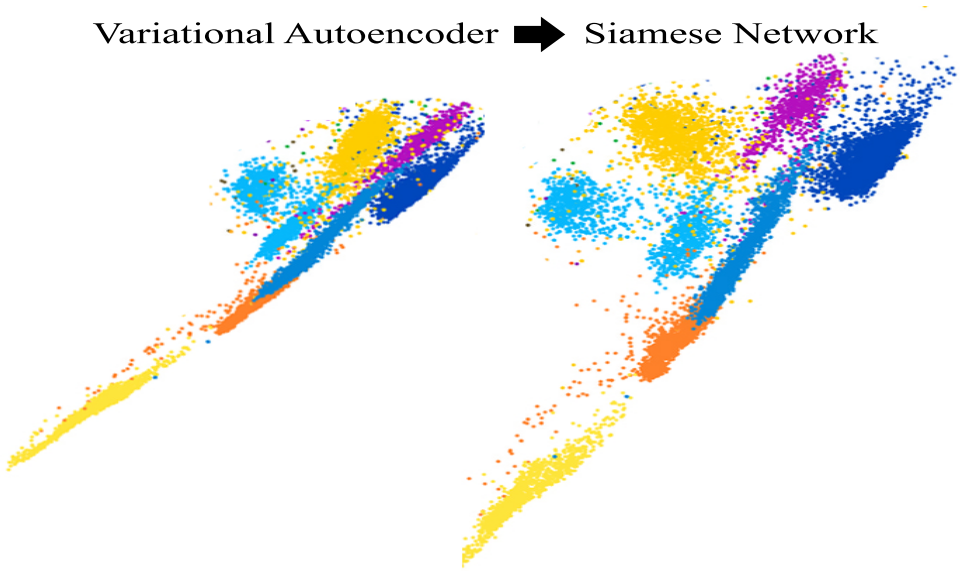


Fig. 3. An example of the result of running the Variational Autoencoder and then refining that result using the Siamese Network. By running the Siamese network after autoencoder, the generated clusters tend to be tighter with more space in-between, making the individual clusters easier to identify. The user has the ability to adjust the number of iterations the Siamese network runs which affects the tightness of the clusters.

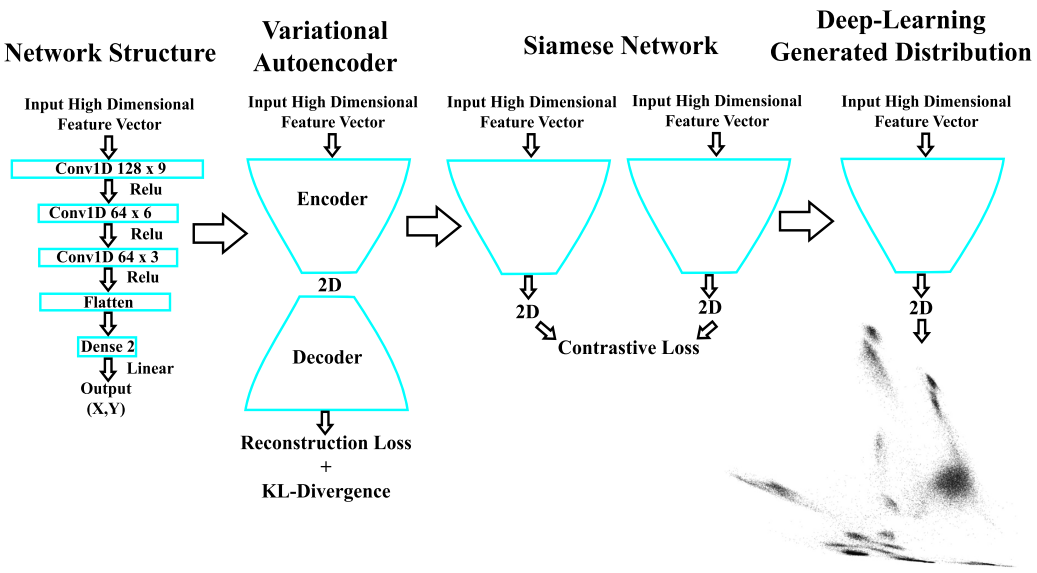


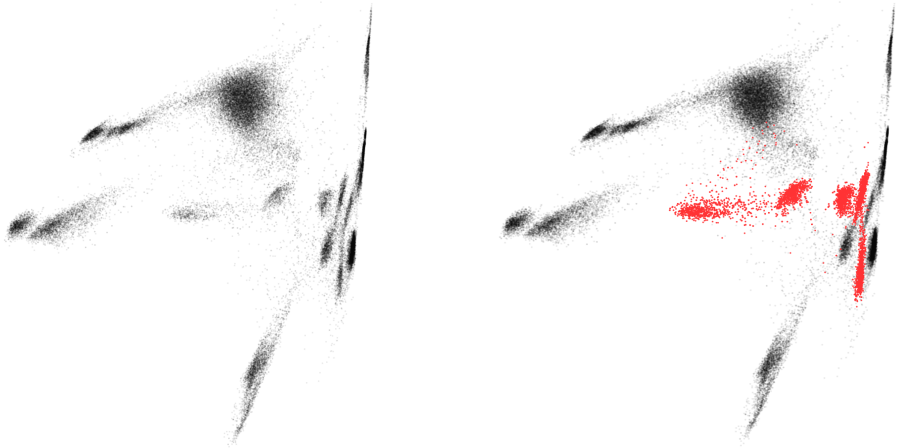
Fig. 4. A visual representation of the network structure used in both the Variational Autoencoder and the Siamese network. The same network structure and weights are shared across the networks. First the Variational Autoencoder runs, and the Siamese network continues using the network weights generated by the autoencoder. After the Siamese network has run, the resulting network is used to generated the 2D distribution of points. The network weights are also saved, reused, and refined in the following iterations.

3.2. Cluster Visualization and Manipulation

Our approach utilizes a 2D scatterplot to visualize the output from the deep learning network. Each data-point in the dataset is assigned a 2D position from the deep learning network and colored based on the group or label it currently belongs to. The first distribution of points is generated using the initial coarse labeling from the dataset. As points are removed and added to new or existing groups, the distribution of the 2D datapoints changes to reflect the changing relationships between them.

We next discuss how the user interacts with the system to modify and reorganize the labeling of the 2D points. Initially, the entire distribution is presented to the user, but such that every point shares the same color (gray), which helps reveal the density of points, as shown in Figure 5(a). When a user clicks on a point, all the points which share the same group as the selected point, become activated (Figure 5(b)), as indicated by becoming colored. Multiple groups of points can be activated in this way. When the user clicks on the white space between points, all selected points/groups become deactivated, and return to the gray color. This scheme allows for an almost unlimited number of groups/labels to be handled by our system, especially since only a relatively few groups will be interacted with at any given time. When a user identifies a potential sub-cluster, they first "activate" the set of points by clicking on one of the points in the region of interest. After clicking, a menu appears, prompting the user with several different options. An example of this menu can be seen in Figure 6. The most basic interaction is "Select Points", which allows for a manual selection of points. When this button is clicked, the cursor changes to a square glyph, 5 pixels in size, to be used like a paint brush select points, as can be seen in Figure 7(b). When points are selected, then turn brighter. For larger selections, the control key may be held, which allows for a drag-and-select circle to appear, which automatically selects all the points

that fall within its boundary that belong to the selected group (Figure 7(c)). To turn these selected points into a new group, the user would then click the “Cut Group” button. These controls allow for manual interaction with the points, but can be tedious to use, and should only be used for fine-tuned selection. For larger changes to the points, we have implemented an automatic partitioning tool, inspired by how humans would naturally segment the points. The “Cut Group” menu option starts a normalized cut operation [Shi and Malik 2000] on the clicked/activated group (if no points are currently selected). From the points belonging to the activated group, a similarity matrix is calculated by comparing the distances between all pairs of points, and then replacing those distance values x with $1.0/x$ to form a correlation matrix. Using this matrix, a diagonal and Laplacian matrix [Anderson Jr and Morley 1985] are computed. The Laplacian matrix is then passed into an eigensolver, which generates the two smallest eigenvalued-eigenvectors. Ignoring the Fiedler vector, the vector whose eigenvalue is zero, the elements of the correlation matrix are reordered using the values of the second-smallest-eigenvalued eigenvector. Next, we compute an appropriate cutting location by minimizing the NCut function as described in the formulas in Figure 8, with cut(A,B) as the total weight of the edges connecting A and B, assoc(A,V) as the total weight of the edges of A in V, and with NCut(A,B) as the cost to cut A and B, normalized to favor roughly equally sized segments within a tolerance. The index which is found to have the minimum NCut value is used as the splitting point, and the group of points which was under the cursor at the time of first click are assigned to a new group. An example of the output from the NCut algorithm can be seen in Figure 9, with the left, smaller group of points being selected and cut from the larger group. The second button, “Select Group” is used to select an entire set of points belonging to a group. This is useful for merging groups or merging a group with a selection of points from another group. To merge a group, the “Merge Group” button is used. This will merge together all selected points into a single group (label). The final button, “Deep Learning”, initializes a call to run the deep learning algorithm which takes the current labeling distribution and retrains the network to generate a new distribution of points.

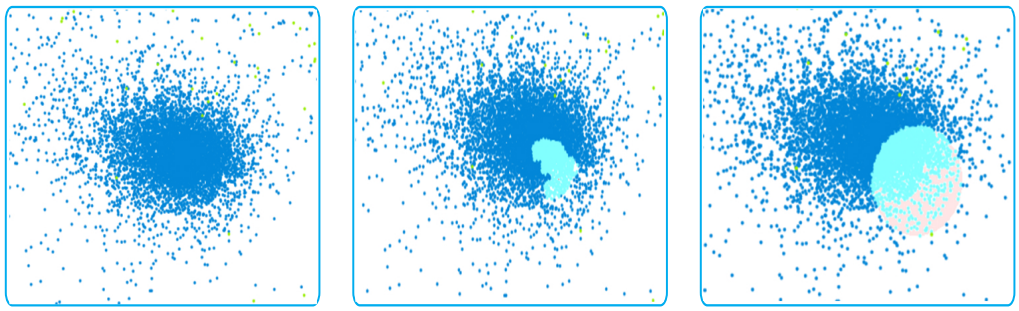


(a) An initial view of the Salinas Valley dataset, with none of the points or groups selected

(b) Same set of points with one group activated.

Fig. 5. Initial view of points, with all points given the same color, which are only assigned a color once selected or activated by the user.

Fig. 6. The menu interface presented when a user selects a point/group.



(a) A set of points before selection

(b) Same set of points with the top section being selected using a "paint brush"

(c) Same set of points selected using a circle selection interface

Fig. 7. Selection of points through manual paint-brush and circle selection interaction.

We have identified a few techniques that often enable an analyst to locate latent structures. The most common indication of a latent structure arises as visually distinct cluster or sets of clusters in the 2D visualization. The distinction could appear as geometric or color separation, or a combination of the two.

An example of such an interaction can be seen in Figure 10(a). A slightly more complicated and common scenario is where the previous example exists, but the second cluster overlaps with a different cluster of another label. This can indicate that either the secondary group belongs to a separate group, such as in Figure 10(b), or that the two clusters belong to the same group and should be merged, as in Figure 10(c). This

$$SimilarityMatrix(x,y) = \frac{\sqrt{(x-y)^2}}{2 * \sigma^2}$$

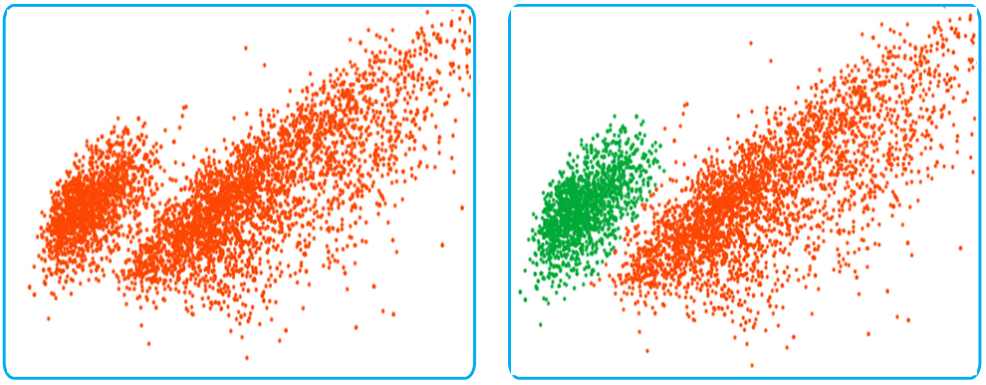
$$CorrelationMatrix = \frac{1.0}{SimilarityMatrix}$$

$$cut(A,B) = \sum_{u \in A, v \in B} w(u,v)$$

$$assoc(A,V) = \sum_{u \in A, t \in V} w(u,t)$$

$$NCut(A,B) = \frac{cut(A,B)}{assoc(A,V)} + \frac{cut(A,B)}{assoc(B,V)}$$

Fig. 8. Equations used to compute normalized cut.



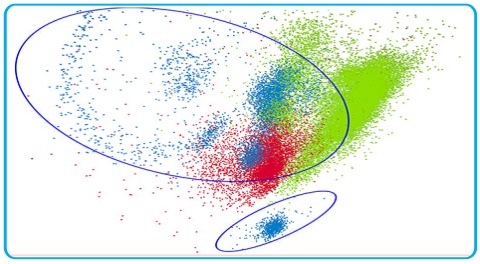
(a) A set of points before Ncut segmentation

(b) Same set of points after Ncut segmentation

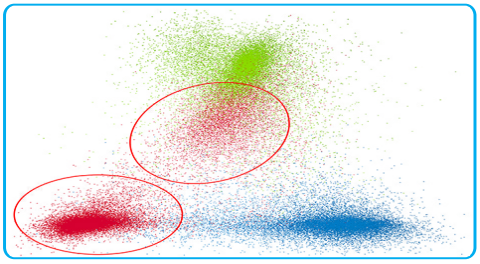
Fig. 9. Segmentation of a set of points using normalized cut (Ncut).

often indicates that the two groups share a common feature subset. A merge may become necessary when trying to extract a hidden cluster, and instead of one clear cluster being extracted, two smaller clusters overlap, indicating that the clusters belong to the same group. Another common scenario is when a bulge exists somewhere on a large cluster, such as in Figure 10(d). Finally, a more complex and rare scenario exists when there are two sub-clusters within one common parent and one child group surrounds the other child group. An example of this can be see in Figure 10(e). This is often represented as a dense set of points surrounded by many disparate points surrounding the central denser group.

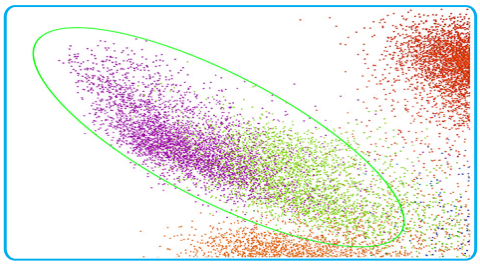
One common problem across all visualizations which perform dimensionality reduction is overlapping points. In high-dimensional space, points may be far apart, but when projected down into 2 or 3 dimensions, they can overlap and obscure one another. Our tool uses two different techniques to handle overlap. First, all points are



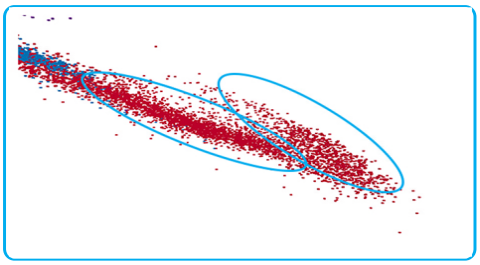
(a) An example of clear cluster separation.



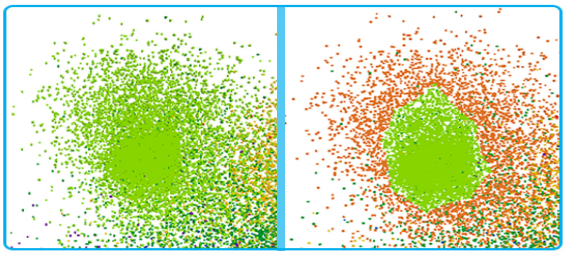
(b) An example of two clusters which are distinct, but the separation boundary is not clearly defined



(c) A merge example, where the green and purple clusters should be combined. This can occur when a larger green cluster from a previous iteration is partially split.



(d) An example of a hidden cluster adjoining to another cluster, typically identified as a bulge protruding off another dense cluster of the same color.

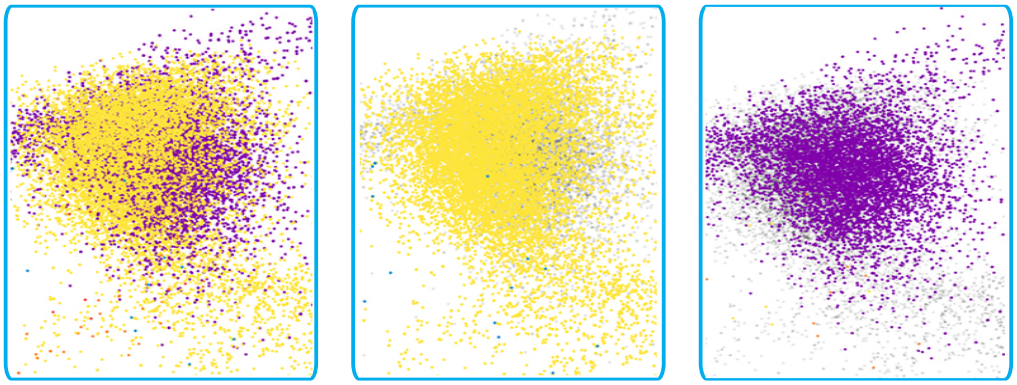


(e) An example of a cluster existing within another cluster. These are typically identified as points of the same label radiating from a dense center.

Fig. 10. Common Interaction Techniques

rendered using transparency, such that the points below can still be seen. Secondly our visualization is designed around the idea that only a few clusters, or sets of points, will be interacted with at any given time. Therefore, when a group is selected, all the points belonging to that group “pop” to the front of the visualization and are given a distinct color, while all inactivated groups retain a high-transparency value and gray color. This allows points of the given group to become easily visible compared to in-

active points or points in the background. Figure 11 demonstrates a simple example, where the purple and yellow points are highly overlapped, and when selected, the other group fades away. Using our tool, we are able to individually select and visualize the independent clusters to see their full extent.



(a) Two clusters, purple and yellow, are obscuring one another

(b) The yellow set of points are selected, and the purple points are greyed-out

(c) The purple set of points are selected, and the yellow points are greyed-out

Fig. 11. Handling of overlapping sets of points.

Throughout this process, it is possible for a user to make an error in labeling, such as mislabeling a few points. This is an inevitable part of the discovery process. If a new group is accidentally created, such that two groups now exist where one group should have only existed, it is easy enough to simply select both groups and merge them into a new group. Second, if a few points are mislabeled, it is very likely that those mislabeled points will appear spatially within the group they were meant to belong to after re-running the point-distribution algorithm, and contrast visually with those points. This contrast of points would then lead the user to merge those two groups together. In addition, this mechanism is often leveraged to tease apart tight groups and to test if a hidden group may exist over multiple iterations (an example of which can be seen in Figure 12). We present an example in Figure 13, where there exist two independent clusters, and a user has accidentally re-assigned half the points of one cluster into the domain of another. Figure 13 shows that our algorithm is resilient against such mistakes, and that while deep learning is taking into account the labels assigned to each data point, the underlying features of the data are also drive the distribution of points, and that the mistakenly labeled points are still co-located with points sharing their true labeling.

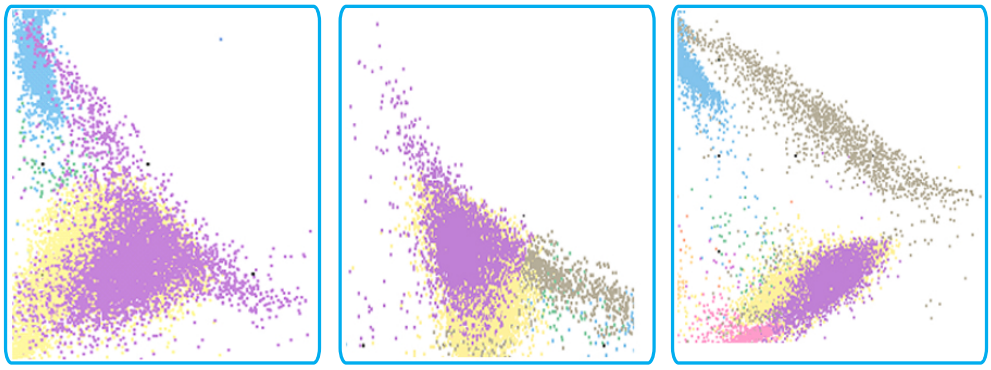
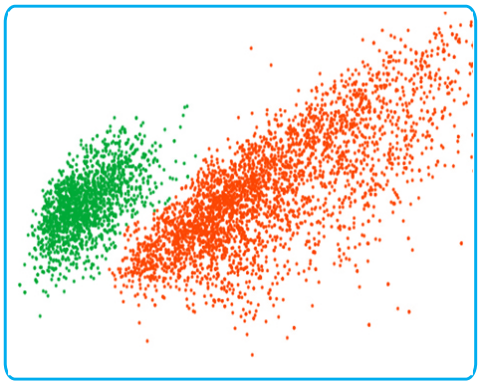
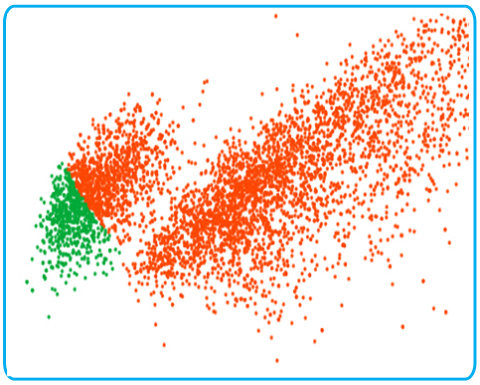


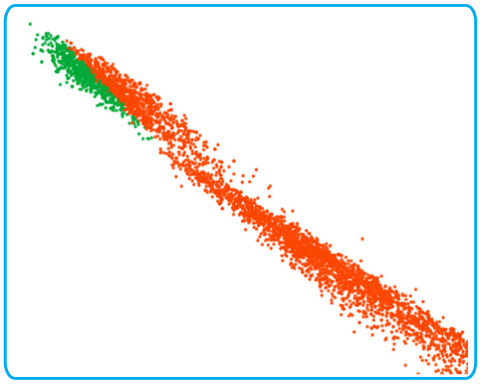
Fig. 12. An example of where a few points are selected from a parent cluster, and over time the points are separated away from the parent cluster to form their own cluster.



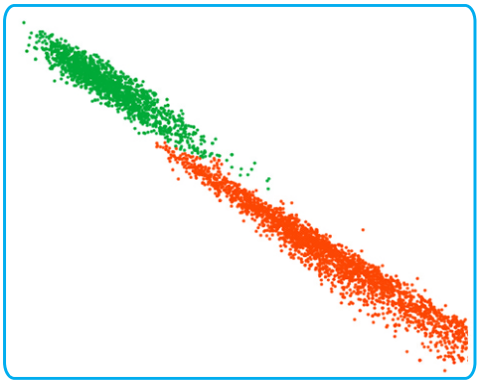
(a) Initial labeling distribution



(b) A labeling mistake where more than half of the points are mis-assigned to a nearby group



(c) The result of regenerating the point distribution using the erroneous labeling



(d) The labeling after correction

Fig. 13. The resilience of the algorithm to labeling mistakes.

There are three main components for each iteration. The first is training the neural network with the current labeling of the dataset to generate the 2D scatterplot. The second component is the creation, modification, and deletion of groups using the visual interface and human interaction. Lastly, the modified labels are fed back into the neural network, where the weights of the network are updated from the previous iterations using the new labeling, and then a new distribution of points are generated.

4. RESULTS

In this section, we present the results of our technique on three different high-dimensional datasets. Our goal was to ease the burden of human labeling and iteratively use deep learning to discover the latent groups and labels within these high-dimensional datasets.

In this section, we will review two of the datasets tested in our system, a hyper-spectral image of Pavia University [Dell'Acqua et al. 2003], and a hyper-spectral scan of Salinas Valley [Landgrebe 2005]. While the datasets we have selected and used for our study are spatial by nature, we disregard the spatial and neighborhood aspects of these datasets and consider each data-point individually during our discovery trials. Many real-world datasets have no natural or obvious spatial component. Therefore, our goal is to generate meaningful spatial representations using our technique that enables the discovery of latent communities for non-spatial datasets. However, it is still possible to use spatial datasets in our approach, but the spatial aspect will act as meta information that is not directly utilized by our technique at present. Throughout this paper, we use the innate spatial distributions of our selected datasets as an easily understandable visual metaphor to help analyze how well we are able to recover the hidden labels, but these visual representations are never used or seen during the iterative discovery process. The only visualizations presented to the user during the iterative discovery process are the constructed scatter-plot spatial representations.

For each dataset, we present a series of figures showing the initial coarse labeling, the feature-space distribution generated through iterative deep learning guided by visual interactions, and the true distribution of labels. Throughout the process, the ground-truth labeling was not used to aid the discovery and refinement process, but is being presented here merely as an illustrative tool. Below we discuss the results for each of the previously presented datasets.

4.1. Pavia University Dataset

The hyperspectral Pavia University dataset consists of a 610×340 pixel image, where each pixel is represented by a 103-dimensional feature vector containing intensity values at different spectral bands. Each pixel has been labeled based on the identity of the surface. There are 9 discrete labels: asphalt, meadows, gravel, trees, metal sheet, soil, bitumen, brick, and shadow. We decided to group those labels into three coarse categories natural surfaces, roads, and buildings, which can be seen in Figure 14. In Figure 14, we show the initial spatial representation of the Pavia University dataset.

Starting from this initial representation, we manually select and re-label points into new clusters, and then generate new point distributions in an iterative manner. Within 3 iterations we were able to generate 9 labels from the initial three coarse labels and improve the labeling accuracy from 67.7% to an accuracy of 88.2%. To test the robustness of our technique, a hold-out test evaluating 10% of the data was conducted, to determine if the hold-out test set could be accurately classified/grouped into the proper cluster when trained on the other 90% of the dataset. For the 10% holdout test-set, 95.3% of those points were correctly assigned to their proper group, confirming the generalization and stability of the technique. There is a lot of overlap between the different "brush" categories; trees, dirt, and meadows. We were unfortunately unable

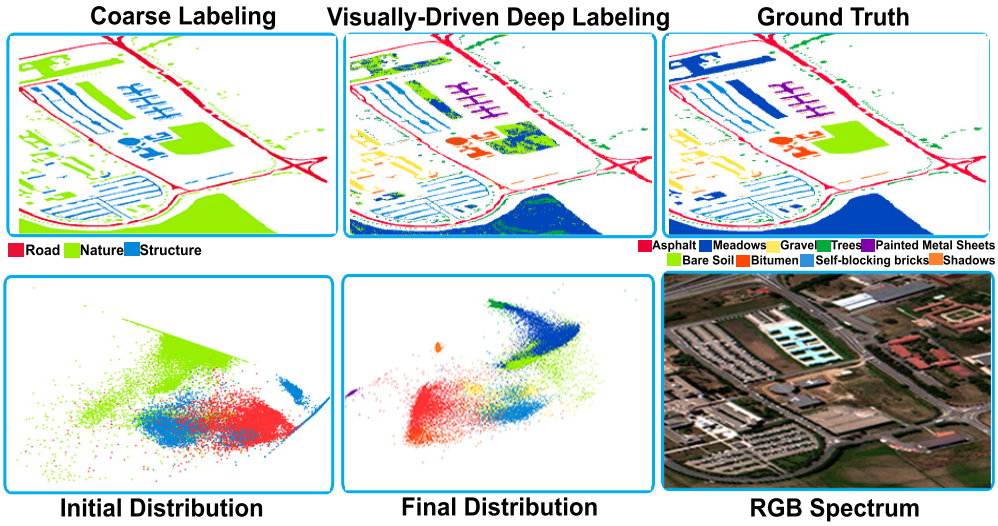


Fig. 14. A comparison of the initial coarse labeling, to the generated refined labels, and the precise (true) labels of the Pavia university dataset after 3 iterations. Starting from the three initial categories: natural surfaces, roads, and buildings, we were able to reconstruct the distribution of the 9 labels with an accuracy of 88.2%.

to differentiate these categories perfectly, but based on the satellite RGB imagery, we believe that these categories are not strictly distinct and therefore, there exist a lot of overlap between those categories. An example of this is the irregular patch of "meadows" in the center of campus that many participants in our user study, presented in more detail later, labeled as an independent group, as seen in Figure 15. The proximity of this cluster in the 2D space indicates that, while this group of points is similar to "meadows" group, there is a clear distinction in their features, as supported by the aerial view. The reconstructed labeling we generated for this dataset can be seen in Figure 14 along with the ground truth. The average time required to run each iteration of the deep neural network was 9 seconds.

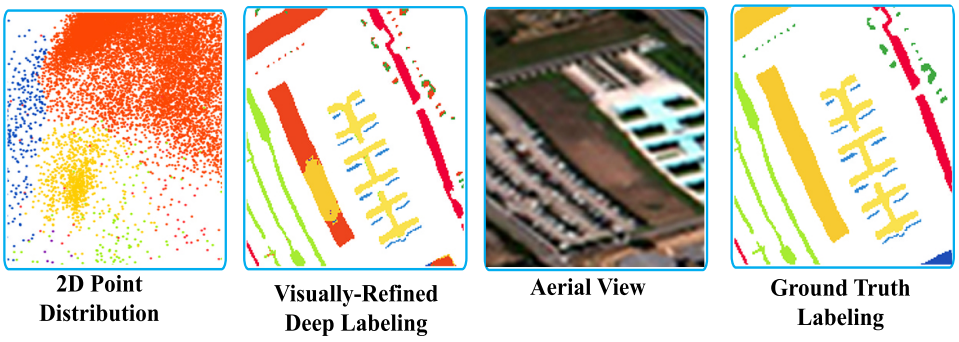


Fig. 15. A comparison between the labeling generated by our system and the ground truth labeling. The labeling generated by our system is driven by the clearly distinct group off the main body of points above it. This difference is supported by a visual difference in the aerial view.

Based on the final data-point distribution as shown in Figure 14, it is possible to further refine the cluster segmentation and labeling, which we discuss more in a later section. The next dataset we present is more complex, involving more dimensions and more hidden labels.

4.2. Salinas Valley

Here we review our results for another hyperspectral dataset, a (224-band) scan of Salinas Valley, California. The dataset consists of a 512×217 image, with each pixel represented by a 224 (204 after removing the water absorption bands) feature vector. The valley consists of fallows, broccoli weeds, stubble, celery, grapes, various vineyards, corn weeds, bare soils, and various stages of lettuce growth for a total of 16 different labels. For our purposes, we clustered those 16 labels into 6 groups, where each group contains similar labels. For example, there were four labels consisting of lettuce at various stages of growth. These were clustered into a single group, which would constitute a reasonable assignment if the growth stages were not known at the time of initial coarse labeling. The 6 groups we created are named broccoli, fallow, celery, lettuce, vineyards, and other. This can be seen in Figure 16. In the bottom left of Figure 16 we show the initial distribution of data points from the Salinas Valley dataset. After three iterations of re-labeling and re-generating the point position distribution, starting from the 6 initial coarse labels and an accuracy of 58.61%, we reconstruct the 16 precise labels with an accuracy of 97.4%. The state-of-the-art technique using the Salinas Valley dataset achieves 97.11% classification accuracy [Kang et al. 2014]. The average amount of processing time required by each iteration of the deep neural network was 10 seconds. Another hold-out test using 10% of the data was conducted using the Salinas Valley dataset, to determine if the test set could be accurately classified/grouped into the proper cluster. For the 10% holdout test-set, trained on the other remaining 90% of data, 100% of those points were correctly assigned to their proper group, also confirming the generalization and stability of the technique.

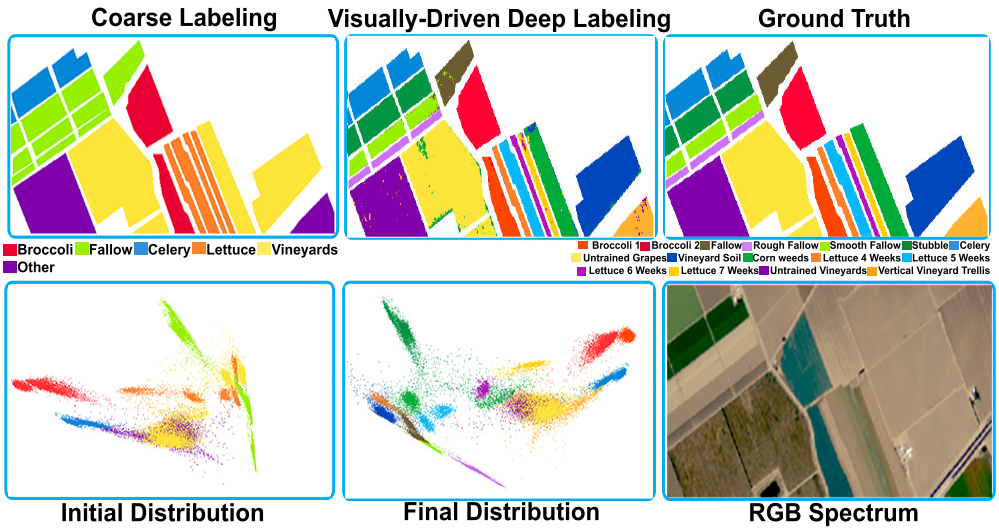


Fig. 16. A comparison of the initial coarse labeling, the generated refined labels, and the precise (true) distribution of labels, of the Salinas valley dataset after five iterations. Starting with 6 initial coarse labels, we were able to reconstruct the distribution of the 16 labels with an accuracy of 97.4%.

4.3. User Study

To demonstrate the usability and capability of our approach and tool, we conducted a small user study, inviting participants to iteratively investigate the two datasets presented earlier in this paper. The participants were informed that their goal was to try to determine if there were any undiscovered groups within the presented visualization, and were briefly shown how to interface with the tool. No information on the number of true clusters, the nature of the data, or how to determine what makes a cluster distinct were provided. Our goal was to simply provide the bare-minimum needed for the participant to use the tool. All participants found the tool to be very intuitive and easy to use. Each participant started from the exact same layout of points, network structure, and initial network weights, and were instructed to continue refining their labeling until they verbally stated that they were satisfied with their groupings. After the study, the participants were explained the true nature of the task, and were astonished with the ease they were able to accomplish this task. The results of the user study are summarized in Figures 17 and 18. Note that the scores for the primary author have been also been added to these plots, showing that the performance between experienced and inexperienced users are comparable. For each dataset, there is a clear iterative upwards improvement in the labeling accuracy of the datasets, suggesting that even users who have had no prior experience working with dimensionality reduction and data analytics were able to not only use our tool, but achieve positive or spectacular results. The solid-line plots show the accuracy of the data-labels with respect to the true, hidden labeling, for each iterative step, showing that the recovered labeling increases in accuracy with each subsequent iteration. The dashed-line shows the amount of time spent on manual interaction during each iteration of the reconstruction process, revealing a general downward trend. This suggests that in the beginning, large changes are being made in labeling, which changes to fine refinements as the iterative labeling process continues. It is important to note that, although we use accuracy as a measure to quantify the success of our system, it is not exactly a proper measure. The goal of our system is hidden label/group discovery, and throughout the rediscovery task, the primary author and participants have found a few sub-groups or hidden groups within the existing datasets, which are clearly differentiated in our visualization (covered further in the next section). The participants, unaware of the underlying metric of accuracy, have over-labeled the datasets, driving down their otherwise high accuracy scores, but in doing so, have been successful in their stated goals and the goals of the presented system.

4.4. Interpreting Discovered Labels

For our testing, we used datasets where the precise labeling of data was complete, and then grouped those precise labels to create coarse labels. Our goal was to reconstruct and rediscover the provided distribution and labeling of the data. Because the full precise labeling (the ground truth) is known, we are able to discern which of the newly discovered labels are associated with each pre-existing label. However, our technique is designed to be used for hidden label/group discovery. A user could start with a group whose label is known, and discover it contains a sub group or is composed of a few sub groups, which could be then found and re-labeled using our technique. It may then be possible to identify what the newly discovered labels correspond to or mean based on additional meta information. For example, we identified through our visualization that the coarse lettuce category from the Salinas Valley dataset had four sub-clusters within it, but it would be very difficult to identify that the reason they are different is due to age, based only on the hyperspectral imagery, and without the additional meta information.

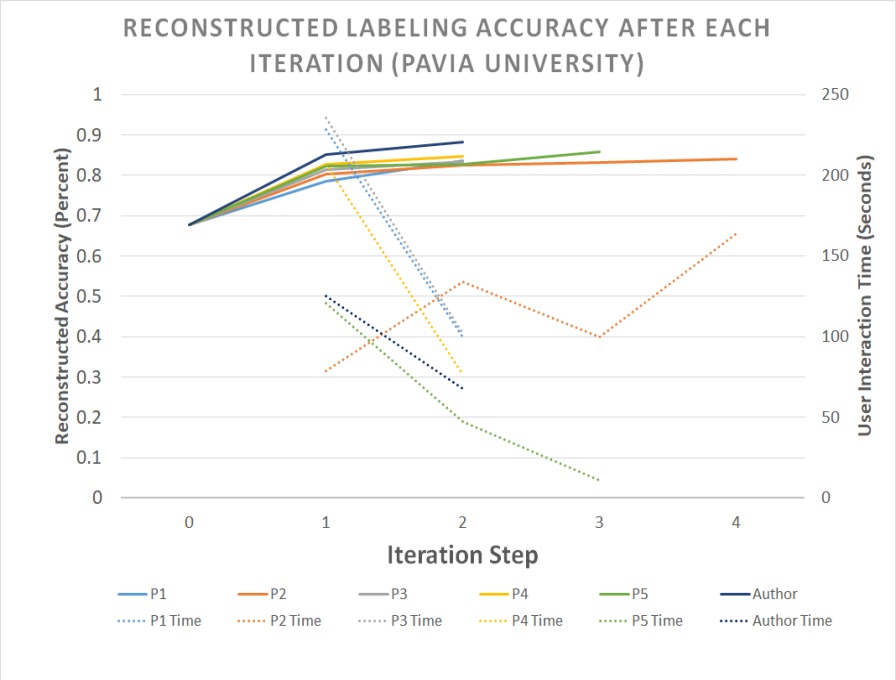


Fig. 17. Participants accuracy and timings for the Pavia University dataset.

In our iterative discovery process on the two previous datasets, it was possible to continue the iterative process to generate more labels, beyond the given ground-truth labeling. For example, in the Pavia University dataset (shown in Figure 14), in addition to the findings presented in Figure 15, we discovered another sub-cluster within one of the larger groups for the painted metal sheets. In Figure 19, we show the result and spatial representation of sub-dividing the purple cluster. By using the additional meta-information (in this case the spatial information) we can see there is a consistent spatial coherence for the newly discovered group. The newly discovered sub-cluster runs along the vertical portions of the roof of these buildings. We may speculate as to why there is a distinct difference here, but unfortunately the precise labeling and aerial view given with the dataset does not reveal any clues as to why there is a consistent feature difference within the group. Another set of interesting discoveries made were in the Salinas Valley dataset (as shown in Figure 16). In Figure 20, we compare the aerial view of the Salinas valley with the labeling generated using our technique and the ground-truth labeling as provided by the dataset. Looking at the aerial view, a discoloration in the field is visible, which clearly does not match the rest of the field, indicating a different surface material is present. The ground truth labeling does not take into account this discoloration, and techniques which use shape or structure to partition or label an image may miss this entirely. Our technique reveals that this surface material was different, and correlates with the aerial view. The absence of these hidden groups within the datasets may be for one of the earlier presented reasons: lack of knowledge of the existence, a simple mistake, or labeling simplification by relying only on the satellite RGB, or overall shape information.

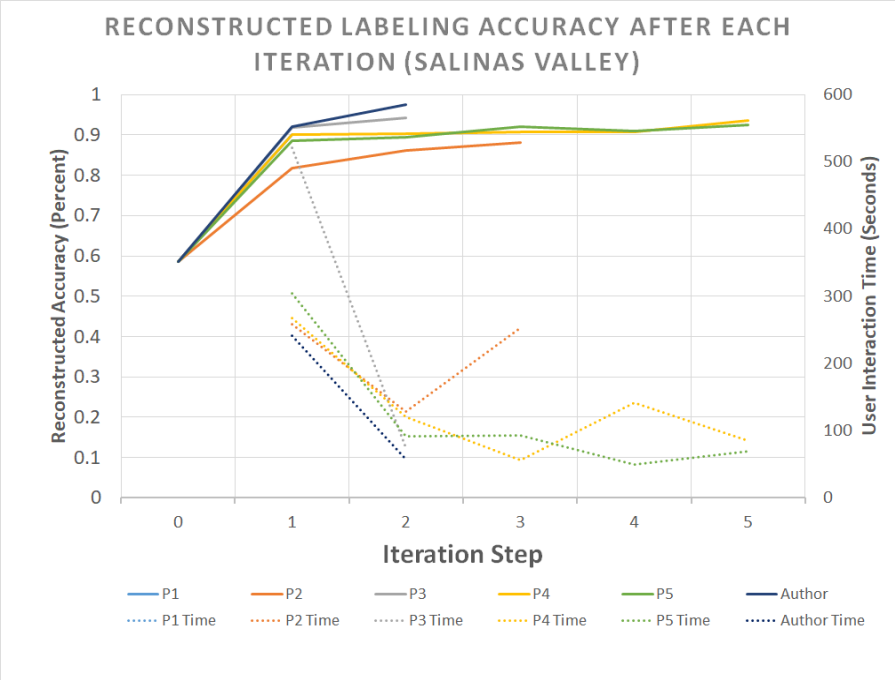


Fig. 18. Participants accuracy and timings for the Salinas Valley dataset.

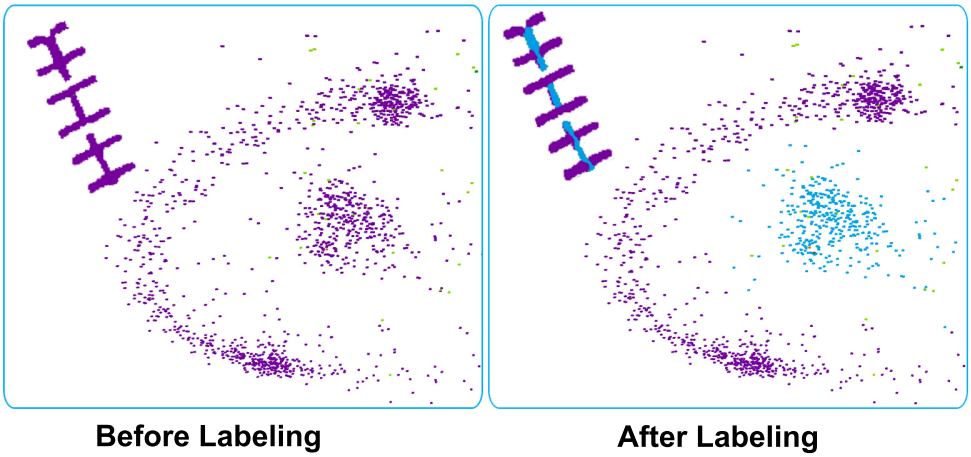


Fig. 19. A potential newly discovered sub-group within the painted metal sheets label in the Pavia University dataset. On the left shows the point distribution and spatial representation of the labeling as generated by our technique. On the right shows another iteration showing if we had sub-divided the current points and the resulting spatial representation.

4.5. DNS Query Dataset

One of the goals of the presented system is to find hidden groups within real-world, large, high-dimensional datasets where there is little to no meta information, such as

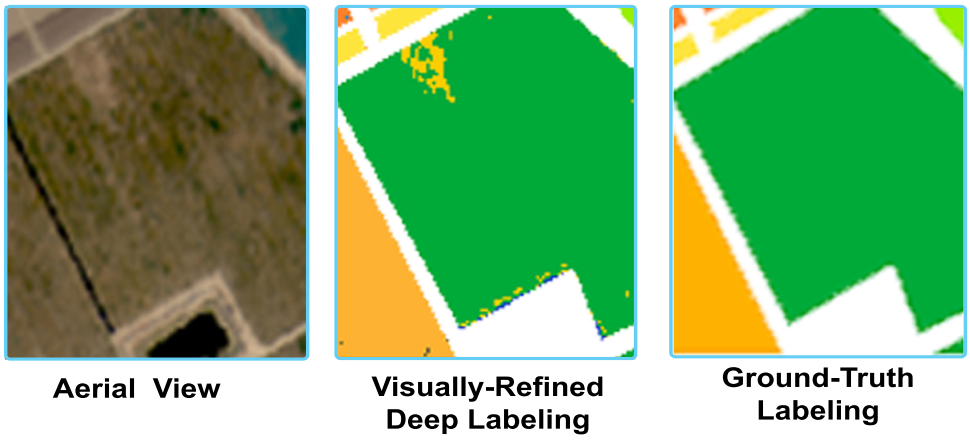


Fig. 20. A potential newly discovered sub-group within the untrained grapes label in the Salinas Valley dataset. The left figure shows the meta information used, the aerial view of the valley. The middle image shows the labeling as generated using our approach. The right image shows the labeling as given by the dataset. The yellow labeling portion in the middle image matches with a discoloring in the aerial view, which suggests that there may be a different material there.

spatial cues, to aid in that process. In this section, was present the results of our system on a real-world dataset from the University of Maryland D-Root DNS server, one of 13 root DNS services which handle DNS requests for the entire internet. DNS stands for Domain Name Service, which is responsible for answering queries, converting human understandable URLs into IP-addresses that the computer can understand. The D-Root DNS service receives billions of such requests per day. As part of their analyses, understanding the nature and distribution of the traffic they receive is crucial. Using this data, our goal was to uncover clusters of queries, to determine if there were any trends in the huge volume of queries the D-Root DNS server receives. To convert the queries into a format that can be passed to deep learning, each query string is converted to a vector of TF-IDF (Term Frequency, Inverse Document Frequency) values, where each value of the feature vector corresponds to a particular character. For our purposes, we process an hour of traffic, which is 7.5 gigabytes for nearly 7 million queries (data points). In Figure 21, we show the evolution of the generated query space throughout the discovery and labeling process, as performed by the primary author. To start, there were no labels (as nothing was known about the dataset), and after 8 iterations, 42 groups/labels were generated (although this process could continue to further refine the groups). To verify the success of our process, we have manually investigated the discovered clusters and have provided each group with a human-understandable label in addition to presenting any trends that exist within a group or across groups. The result of this labeling is shown in Figure 22. Many different groups were discovered, the most salient being those regarding erroneous IP address queries. Using our tool, we were able to identify different configurations of queried IP addresses based on the error of the request. Other interesting discoveries were sets of code fragments and commands being issued over DNS, which is common for command and control of botnets. Lastly, we were able to find different configurations of alphabetic-based queries, ranging from differently sized sets of random characters, to more formed queries with specific domains. Our collaborators at the University of Maryland D-Root server agree that not only have we confirmed the various types of traffic they see, but we have also revealed other types of traffic, that they were unaware of but are now interested in.

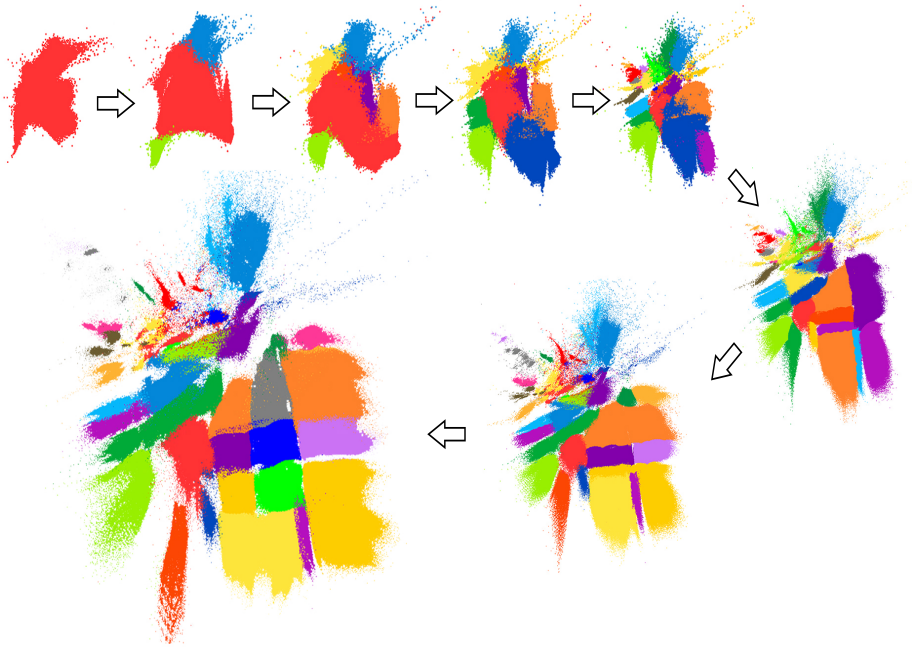


Fig. 21. The evolution of the query-space representation over eight iterations, showing the influence of the iterative labeling. Note that for some clusters certain colors have been re-used due to the high-number of groups.

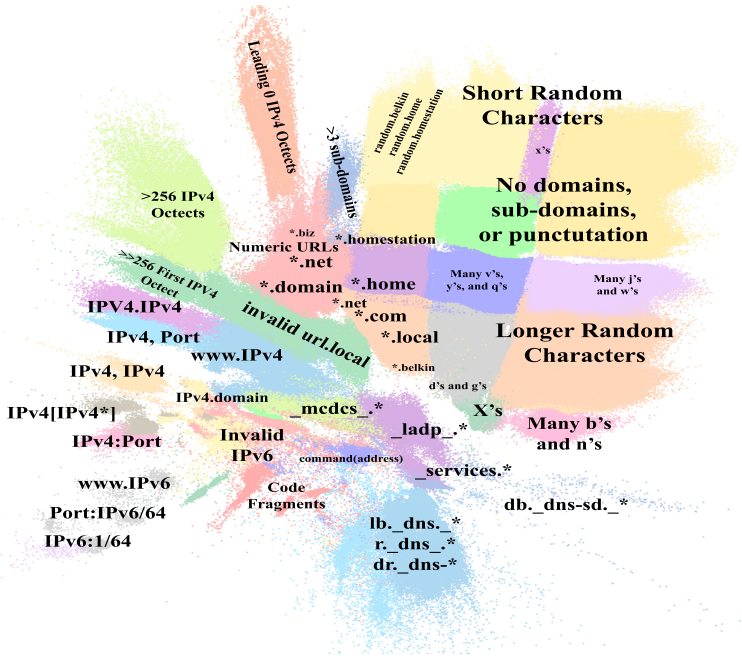


Fig. 22. The output of the DNS query investigation coarsely labeled to identify the meaning of the individual clusters and their relation to each other. Note that for some clusters certain colors have been re-used due to the high-number of groups.

5. DISCUSSION

In recent years there has been an explosion of large high-dimensional datasets. Extracting meaningful information hidden within these high-dimensional datasets is not trivial, and is made more difficult by erroneous and high-level (coarse) labeling. These errors may be caused by subjectivity, lack of time, or misunderstanding of the data. One way of revealing hidden trends or structures within a high-dimensional dataset is to group similar points based on their features. Identifying similar high-dimensional data-points is difficult in part due to the Curse of Dimensionality, which states that the distance between all pairs of high-dimensional points converge, meaning that traditional distance techniques reveal little information. Another difficult aspect of high-dimensional information visualization is deciding on how to show distinctions and similarities between high-dimensional points in an intuitive and insightful way.

In our approach, finer labeling and classification of the data requires more iterations. To reduce the amount of human effort required we have added the point-and-click normalized cut segmentation. This is particularly effective for segmenting large numbers of smaller clusters quickly. Introducing more automated tools such as this to reduce the effort required by a human, but still keeping a human in the loop and giving them the final say, would be an interesting direction of future research.

6. CONCLUSIONS

The current advances and future potential of deep learning is without question. The objective of this paper is to provide a first step to advance our capability and understanding of deep learning. In this paper we presented our approach to facilitate the discovery of latent structures and communities/labels within high-dimensional non-spatial datasets using deep learning. Given a coarsely, or broadly labeled, or unlabeled high-dimensional dataset, we generate a 2D distribution of the data based on the idea that similarly attributed and labeled points should be in close proximity to each other. Through an iterative process, an analyst can select points and assign them to new or existing communities, which is then used in conjunction with deep learning, to refine the 2D spatial distribution of the points, revealing more new information. Supplementing the instinct and ability of analysts to identify patterns with deep learning, we have shown on three different datasets that our technique is able to reconstruct hidden structures and communities. In many previous works, deep learning has been used as a black-box classifier, with little or no human interaction. Our technique, presented here, enables both deep learning and the human analyst to support, refine, and enhance each other through visualization.

7. ACKNOWLEDGEMENTS

We would like to extend our sincere appreciation to the anonymous reviewers who helped us refine this paper that significantly improved its presentation. We would also like to extend our thanks to Karl Reuss, Bruce Crabill, and Tripti Sinha from the University of Maryland D-Root DNS authority for their help in providing real-world capture data, and providing continuous guidance in the development and analysis of our visualization and findings. We appreciate the support of the DOD contract H98230-13-D-0056, NSF grants 14-29404, 15-64212, and the State of Maryland's MPower initiative. Any opinions, findings, conclusions, or recommendations expressed in this article are those of the authors and do not necessarily reflect the views of the research sponsors.

REFERENCES

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, and others. 2016. TensorFlow: A System for Large-Scale Machine Learning.. In *OSDI*, Vol. 16. 265–283.

Shun-ichi Amari, Andrzej Cichocki, and Howard Hua Yang. 1996. A new learning algorithm for blind signal separation. In *Advances in neural information processing systems*. 757–763.

Saleema Amershi, Bongshin Lee, Ashish Kapoor, Ratul Mahajan, and Blaine Christian. 2011. CueT: human-guided fast and accurate network alarm triage. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 157–166.

El-ad David Amir, Kara L Davis, Michelle D Tadmor, Erin F Simonds, Jacob H Levine, Sean C Bendall, Daniel K Shenfeld, Smita Krishnaswamy, Garry P Nolan, and Dana Pe'er. 2013. viSNE enables visualization of high dimensional single-cell data and reveals phenotypic heterogeneity of leukemia. *Nature biotechnology* 31, 6 (2013), 545–552.

William N Anderson Jr and Thomas D Morley. 1985. Eigenvalues of the Laplacian of a graph. *Linear and multilinear algebra* 18, 2 (1985), 141–145.

Mikhail Belkin and Partha Niyogi. 2003. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation* 15, 6 (2003), 1373–1396.

Yushi Chen, Zhouhan Lin, Xing Zhao, Gang Wang, and Yanfeng Gu. 2014. Deep learning-based classification of hyperspectral data. *IEEE Journal of Selected topics in applied earth observations and remote sensing* 7, 6 (2014), 2094–2107.

Hsueh-Chien Cheng, Antonio Cardone, Somay Jain, Eric Krokos, Kedar Narayan, Sriram Subramaniam, and Amitabh Varshney. 2018. Deep-learning-assisted Volume Visualization. *IEEE Transactions on Visualization and Computer Graphics* PP, 99 (22 January 2018), 1–14. DOI:http://dx.doi.org/10.1109/TVCG.2018.2796085

Hsueh-Chien Cheng, Antonio Cardone, Eric Krokos, Bogdan Stoica, Alan Faden, and Amitabh Varshney. 2017. Deep-Learning-Assisted Visualization for Live-Cell Images. In *Proceedings of 2017 IEEE International Conference on Image Processing (ICIP)*. IEEE. DOI:http://dx.doi.org/10.1109/ICIP.2017.8296507

François Chollet and others. 2015. Keras: Deep learning library for theano and tensorflow. *URL: https://keras.io/k* (2015).

Tuan Nhon Dang and Leland Wilkinson. 2014. Transforming scagnostics to reveal hidden features. *IEEE Transactions on Visualization and Computer Graphics* 20, 12 (2014), 1624–1632.

Fabio Dell'Acqua, Paolo Gamba, and Alessio Ferrari. 2003. Exploiting spectral and spatial information for classifying hyperspectral data in urban areas. In *Geoscience and Remote Sensing Symposium, 2003. IGARSS'03. Proceedings. 2003 IEEE International*, Vol. 1. IEEE, 464–466.

Alex Endert, Patrick Fiaux, and Chris North. 2012. Semantic interaction for visual text analytics. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 473–482.

Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. 2010. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research* 11, Feb (2010), 625–660.

Shantanu Godbole and Sunita Sarawagi. 2004. Discriminative methods for multi-labeled classification. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 22–30.

Raia Hadsell, Sumit Chopra, and Yann LeCun. 2006. Dimensionality reduction by learning an invariant mapping. In *Computer vision and pattern recognition, 2006 IEEE computer society conference on*, Vol. 2. IEEE, 1735–1742.

Geoffrey E Hinton and Ruslan R Salakhutdinov. 2006. Reducing the dimensionality of data with neural networks. *Science* 313, 5786 (2006), 504–507.

Nguyen Quoc Viet Hung, Duong Chi Thang, Matthias Weidlich, and Karl Aberer. 2015. Minimizing efforts in validating crowd answers. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*. ACM, 999–1014.

Cheuk Yiu Ip, Amitabh Varshney, and Joseph JaJa. 2012. Hierarchical exploration of volumes using multilevel segmentation of the intensity-gradient histograms. *IEEE Transactions on Visualization and Computer Graphics* 18, 12 (2012), 2355–2363.

Thorsten Joachims. 1998. Text categorization with support vector machines: Learning with many relevant features. In *European conference on machine learning*. Springer, 137–142.

Xudong Kang, Shutao Li, and Jon Atli Benediktsson. 2014. Spectral–spatial hyperspectral image classification with edge-preserving filtering. *IEEE Transactions on Geoscience and Remote Sensing* 52, 5 (2014), 2666–2677.

Yehuda Koren. 2003. On Spectral Graph Drawing. In *Computing and Combinatorics*, Tandy Warnow and Binhai Zhu (Eds.). Lecture Notes in Computer Science, Vol. 2697. Springer Berlin Heidelberg, 496–508. DOI:http://dx.doi.org/10.1007/3-540-45071-8_50

Eric Krokos and Hanan Samet. 2014. A look into twitter hashtag discovery and generation. In *Proceedings of the 7th ACM SIGSPATIAL Workshop on Location-Based Social Networks (LBSN14), Dallas, TX, Nov*.

Joseph B Kruskal. 1964. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika* 29, 1 (1964), 1–27.

David A Landgrebe. 2005. *Signal theory methods in multispectral remote sensing*. Vol. 29. John Wiley & Sons.

Dong-Hyun Lee. 2013. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on Challenges in Representation Learning, ICML*, Vol. 3. 2.

Shusen Liu, Bei Wang, P-T Bremer, and Valerio Pascucci. 2014. Distortion-Guided Structure-Driven Interactive Exploration of High-Dimensional Data. In *Computer Graphics Forum*, Vol. 33. Wiley Online Library, 101–110.

Shusen Liu, Bei Wang, Jayaraman J Thiagarajan, P-T Bremer, and Valerio Pascucci. 2015. Visual Exploration of High-Dimensional Data through Subspace Analysis and Dynamic Projections. In *Computer Graphics Forum*, Vol. 34. Wiley Online Library, 271–280.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research* 9, Nov (2008), 2579–2605.

Jason Matheny. 2016. Intelligence Advanced Research Projects Activity. (2016). 3rd Annual BRAIN Initiative Investigators Meeting, North Bethesda, Maryland.

Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*. 807–814.

Mohammad Pezeshki, Linxi Fan, Philemon Brakel, Aaron Courville, and Yoshua Bengio. 2016. Deconstructing the ladder network architecture. In *International Conference on Machine Learning*. 2368–2376.

Antti Rasmus, Mathias Berglund, Mikko Honkala, Harri Valpola, and Tapani Raiko. 2015. Semi-supervised learning with ladder networks. In *Advances in Neural Information Processing Systems*. 3546–3554.

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

Dominik Sacha, Leishi Zhang, Michael Sedlmair, John A Lee, Jaakko Peltonen, Daniel Weiskopf, Stephen C North, and Daniel A Keim. 2017. Visual interaction with dimensionality reduction: A structured literature analysis. *IEEE Transactions on Visualization and Computer Graphics* 23, 1 (2017), 241–250.

Jianbo Shi and Jitendra Malik. 2000. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence* 22, 8 (2000), 888–905.

Axel J Soto, Ryan Kiros, Vlado Kešelj, and Evangelos Milios. 2015. Exploratory Visual Analysis and Interactive Pattern Extraction from Semi-Structured Data. *ACM Transactions on Interactive Intelligent Systems (TiiS)* 5, 3 (2015), 16.

Graham W Taylor, Geoffrey E Hinton, and Sam T Roweis. 2007. Modeling human motion using binary latent variables. *Advances in neural information processing systems* 19 (2007), 1345.

Konstantinos Trohidis, Grigorios Tsoumakas, George Kalliris, and Ioannis P Vlahavas. 2008. Multi-Label Classification of Music into Emotions.. In *The International Society of Music Information Retrieval*, Vol. 8. 325–330.

Cagatay Turkay, Erdem Kaya, Selim Balcisoy, and Helwig Hauser. 2017. Designing Progressive and Interactive Analytics Processes for High-Dimensional Data Analysis. *IEEE Transactions on Visualization and Computer Graphics* 23, 1 (2017), 131–140.

Naonori Ueda and Kazumi Saito. 2003. Parametric mixture models for multi-labeled text. *Advances in neural information processing systems* (2003), 737–744.

Malcolm Ware, Eibe Frank, Geoffrey Holmes, Mark Hall, and Ian H Witten. 2001. Interactive machine learning: letting users build classifiers. *International Journal of Human-Computer Studies* 55, 3 (2001), 281–292.

Martin Wattenberg, Fernanda Vigas, and Ian Johnson. 2016. How to Use t-SNE Effectively. *Distill* (2016).

Ka-Ping Yee, Kirsten Swearingen, Kevin Li, and Marti Hearst. 2003. Faceted metadata for image search and browsing. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 401–408.

Matthew D Zeiler. 2012. ADADELTA: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701* (2012).