DRAVA: Aligning Human Concepts with Machine Learning Latent Dimensions for the Visual Exploration of Small Multiples

Qianwen Wang
qianwen_wang@hms.harvard.edu
Harvard Medical School
Boston, MA, USA

Sehi L'Yi
sehi_lyi@hms.harvard.edu
Harvard Medical School
Boston, MA, USA

Nils Gehlenborg
nils@hms.harvard.edu
Harvard Medical School
Boston, MA, USA

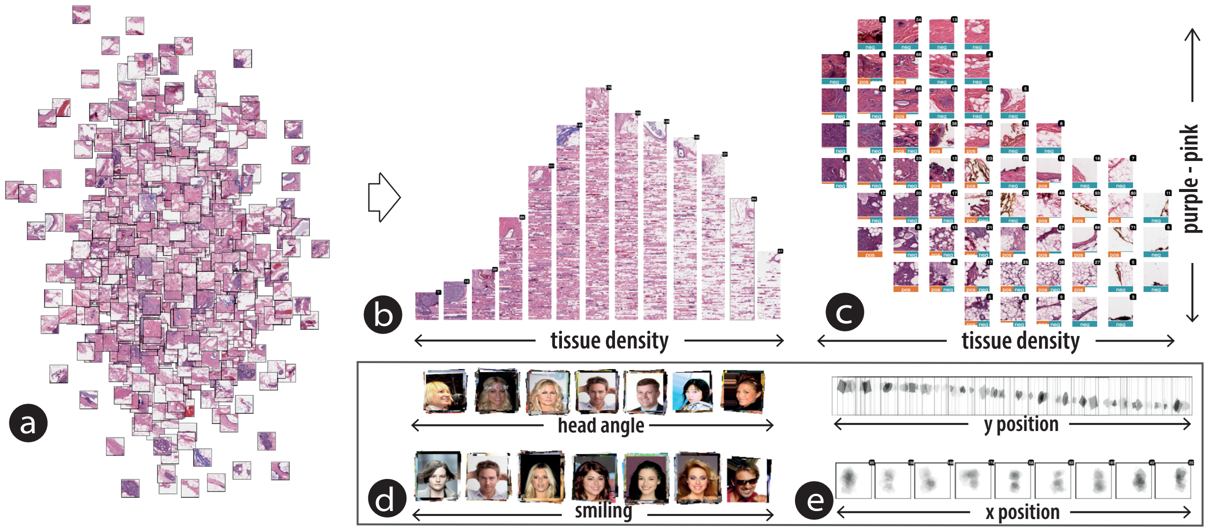

Figure 1: Drava enables concept-driven exploration by aligning semantic latent dimensions with human concepts. (a) UMAP projection of image patches of breast cancer specimens. (b) All image patches are organized and piled up based on the density of tissues. (c) All image patches are grouped into a grid layout according to the tissue density and color. The two visual concepts reveal a strong association of the presentation of invasive ductal carcinomas (IDC), *i.e.*, the orange label. (d-e) More examples.

ABSTRACT

Latent vectors extracted by machine learning (ML) are widely used in data exploration (*e.g.*, t-SNE) but suffer from a lack of interpretability. While previous studies employed disentangled representation learning (DRL) to enable more interpretable exploration, they often overlooked the potential mismatches between the concepts of humans and the semantic dimensions learned by DRL. To address this issue, we propose Drava, a visual analytics system that supports users in 1) relating the concepts of humans with the semantic dimensions of DRL and identifying mismatches, 2) providing feedback to minimize the mismatches, and 3) obtaining data insights from concept-driven exploration. Drava provides a set of visualizations and interactions based on visual piles to help users understand and refine concepts and conduct concept-driven exploration. Meanwhile, Drava employs a concept adaptor model to fine-tune the semantic dimensions of DRL based on user refinement. The usefulness of Drava is demonstrated through application scenarios and experimental validation.

CHI'23, April 23–28, 2023, Hamburg, Germany
© 2023 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-9421-5/23/04.
https://doi.org/10.1145/3544548.3581127

CCS CONCEPTS

• **Human-centered computing** → **Interactive systems and tools**; **Visual analytics**; **Information visualization**.

KEYWORDS

Visual exploration, XAI, Human-AI collaboration, latent space, small multiples

ACM Reference Format:
Qianwen Wang, Sehi L'Yi, and Nils Gehlenborg. 2023. DRAVA: Aligning Human Concepts with Machine Learning Latent Dimensions for the Visual Exploration of Small Multiples. In *CHI'23: ACM CHI Conference on Human Factors in Computing Systems,April 23–28, 2023, Hamburg, Germany*. ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3544548.3581127

1 INTRODUCTION

Presenting analyzed small multiples (*e.g.*, patches of medical images, miniature visualizations of a large genomic sequence) using latent vectors learned by machine learning (ML) models has become a common practice in many visual analytics systems [6, 10]. A latent vector, usually represented as multi-dimensional quantitative values, is a compact representation of the analyzed data to capture relevant information. For example, a 64×64 pixel image can be represented as a 10-dimensional latent vector. Compared with analysis using raw data or human-crafted metrics, latent vectors enable users to organize and explore a large amount of data and conduct analysis tasks, such as finding similar items and identifying outliers, more efficiently.

Even though latent vectors can accurately capture patterns extracted from the analyzed data, they cannot be directly interpreted by humans like the original images or texts. For this reason, latent vectors are usually used to represent the similarity between data items, assuming the latent vectors of two similar items are close in a latent space. For example, dimension reduction methods (*e.g.*, t-SNE [58], UMAP [41]) are widely used to visualize latent vectors in 2D space, showing the similarities and differences among data items. Other prior studies proposed to hierarchically cluster items based on their latent vectors to conduct pattern-driven visual analytics [6]. However, definitions of "similar items" vary depending on analysis tasks, and there is no single definition that can be applied to all scenarios. Even though some prior studies have incorporated user input to learn user's perception of similarity [10, 32] and even to extract human-readable concepts (*e.g.*, gender from face images) [37, 68], visual analytics based on latent vectors still suffers from their limited interpretability.

Disentangled representation learning (DRL) [12, 22] is a promising approach that can provide more explainable latent vectors through unsupervised learning, *i.e.*, without human labels. By disentangling features and encoding them as separated dimensions in the latent vectors, DRL can generate latent vectors whose values carry semantics and can reveal human-understandable concepts, *e.g.*, the value on one dimension indicates whether a person is smiling or not (Figure 1d). We call such dimensions semantic dimensions. Some recent visualization tools [18, 20, 59] have successfully employed DRL in their analysis and demonstrated the effectiveness of DRL. For example, Gou *et al.* [18] used DRL for traffic light images to summarize images based on human-readable concepts, such as color, brightness, and rotation. These studies usually assumed that the learned semantic dimensions can perfectly capture human concepts, and the concepts can be accurately represented by a set of synthesized images. However, these assumptions do not always hold. **Potential mismatches can exist between the semantic latent dimensions learned by ML models and the concepts of humans.** As shown in Figure 2a, one latent dimension correlates to the angle of human head according to the synthesized images. But when using this dimension to organize images, our experiment results show that the model confuses "angle of the head" with "whether part of the face is covered", *e.g.*, covered by a dark shadow or a flower (Figure 2b). Meanwhile, previous studies focus on using DRL to diagnose supervised ML models rather than building an understanding of the data [18, 20]. They provide limited discussion about user needs in understanding and utilizing DRL for concept-driven data exploration.

This study aims to provide a more interpretable and flexible visual exploration of small multiples by better aligning concepts of human users with the semantic latent vectors generated by ML models. We propose Drava, an interactive system that utilizes

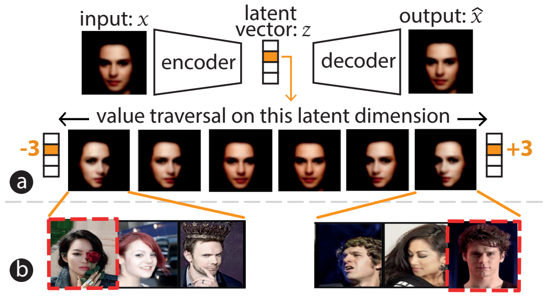

Figure 2: Mismatches between semantic latent dimensions and human concepts (red dashed boxes). (a): Synthesized images through value traversal of a latent dimension. (b): Items with the same latent values as the left- and right-most synthesized images.

Disentangled **R**epresentation learning as **A** **V**isual **A**nalytics approach for concept-driven data exploration. In Drava, a dataset is represented as a set of small multiples [57], *i.e.*, a series of basic charts or graphics that show instances or different slices of the dataset (Figure 1). Hereafter, we call each small multiple as a *data item*. For each data item, DRL learns a multi-dimensional latent vector, certain dimensions of which have semantic meanings. Drava supports an interpretable exploration of these items by supporting users in correlating and aligning the semantic dimensions with human concepts. The interactive visualizations and algorithms in Drava are motivated and guided by a three-step workflow that we propose. Throughout this workflow, users 1) understand ML-learned semantic dimensions and identify their potential mismatches with human concepts, 2) refine and align ML semantic dimensions with human concepts, and 3) generate new knowledge about the analyzed data through concept-driven exploration. Particularly, Drava automatically ranks latent vectors and proposes a concept adaptor that can refine a concept based on human input. Meanwhile, a set of interactions based on visual piles [33] are provided, enabling users to effectively arrange, summarize, and compare items based on human-readable concepts. We demonstrate the usefulness of Drava through experimental validation and four usage scenarios. Drava is available at https://qianwen.info/DRAVA/.

2 BACKGROUND: DISENTANGLED REPRESENTATION LEARNING

DRL is a promising machine learning method that is able to extract interpretable features without human supervision. Given an input item $\mathbf{x}$, the goal of DRL is to learn a vector $\mathbf{z}$ that captures the features of $\mathbf{x}$ in a disentangled manner. For example, as shown in Figure 2a, a DRL model learns to represent an image of a human face using $\mathbf{z}$ and captures the feature "the angle of head" independently in z_1 (*i.e.*, the second dimension of $\mathbf{z}$). Similarly, an area chart can be described as a vector $\mathbf{z}$ where z_0 indicates the height of the chart, z_1 indicates the trend, etc. DRL assumes $\mathbf{x}$ as a joint distribution of independent and dependent generative factors [22]. While these independent factors will be captured in separated dimensions of $\mathbf{z}$ (*i.e.*, semantic dimensions), the dependent factors will remain entangled in other dimensions of $\mathbf{z}$ that are not used for representing the independent factors. In other words, some dimensions of $\mathbf{z}$ will have semantic meanings while others will not. For a precise mathematical definition of disentangled and entangled dimensions, we refer the readers to [8, 21, 22].

A DRL model learns disentangled representations via two loss terms, a reconstruction term and a regularization term. The reconstruction term evaluates the differences between the input item $\mathbf{x}$ and the reconstructed item $\hat{\mathbf{x}}$, encouraging the model to learn $\mathbf{z}$ that capture the main characteristics of the input item. The regularization term encourages disentanglement of the latent vectors. A DRL model is usually constructed by encouraging disentanglement in standard generative models, such as VAE [22] and GAN [12]. The state-of-the-art DRL approaches are largely based on VAE mainly due to their better training stability than GAN-based methods. For example, the loss function of β-VAE is defined as

$$E_{q_\phi(z|x)}[\log p_\theta(x|z)] - \beta D_{KL}(q_\phi(z|x)||p(z)) \tag{1}$$

The first term is a reconstruction loss, and the second term is a regularization for disentanglement. With $\beta > 1$, β-VAE encourages disentangled $\mathbf{z}$ by putting a constraint on the latent bottleneck. Prior studies have proposed various regularization terms for disentanglement. For more details, refer to prior studies [21, 36]. Given its wide popularity, we use β-VAE in Drava with some modifications (subsection 6.1). The proposed framework can be easily adapted to other VAE-based DRL, such as FactorVAE [28] or β-TCVAE [11].

3 RELATED WORK

First, since Drava aims to assist data exploration using explainable latent vectors, it is closely related to **visual analytics on latent vectors** and, more broadly, **visual analytics for ML models** whose hidden layers generate latent vectors of the input data.

Many visual analytics tools have been proposed to support interactive explorations of latent vectors. Dimensionality reduction techniques, such as t-SNE [58], UMAP [41], PCA [1], and their variants [35, 66], are widely used to assist the visualization of latent vectors. Most of them focus on analyzing the latent vectors generated by a specific model [67], such as a convolutional neural network [25, 34, 46], a graph neural network [24], and a recurrent neural network [35, 42, 54]. Other studies aim to provide more generic methods for visually exploring the latent space [7, 37, 51]. Most relevant to our study is LSC [37], which provides comprehensive support for mapping and comparing semantic dimensions in the analysis of latent vectors. However, LSC requires users to manually identify semantic dimensions, either by importing data labels or by interactively grouping items.

Apart from showing latent vectors, previous studies have combined interactive visual analytics with interactive or explainable ML to introduce interpretability into the analysis of latent vectors [18, 23, 68]. Several studies [18, 20, 59] used DRL to extract semantic dimensions and associate model performance with human concepts (*e.g.*, brightness of images, location of objects). The semantic dimensions learned by DRL are directly used without refinement, mostly because they are low-level concepts that can be easily extracted by ML. Jia *et al.* [23] proposed a visual explainable active learning approach that asks users questions and uses their answers to learn explainable attributes that can be used to classify images from unseen classes. Zhao *et al.* [68] proposed a visualization tool where users can explore and label image patches with a certain concept. These labels are used to train a concept extractor network, enabling users to diagnose model predictions using the learned concept.

However, these studies mainly focus on understanding the working mechanism of ML models and improving model performances (*i.e.*, VIS for ML). How to utilize explainable latent vectors for concept-driven data exploration (*i.e.*, XAI for VIS) has not been extensively discussed. Drava is built upon previous visual analytics studies on latent vectors and ML models. Unlike previous studies, Drava focuses on aligning interpretable latent vectors with human concepts to assist concept-driven data exploration.

Second, Drava learns the visual representation and supports **the exploration of small multiples** [57], a series of miniature visualizations that represent different facets, subsets, or instances of a dataset. Current studies in data visual exploration usually present small multiples as points (*e.g.*, [7, 16, 47, 51]), glyphs (*e.g.*, [29, 63]), or images (*e.g.*, [18, 27, 37]) and place them in a grid, a dimension reduction projection, or a data-driven layout. For example, Sharkzor [27] enabled users to interactively organize images and their groups while providing visual cues for groups (*e.g.*, badges). AxiSketcher [29] uses glyph representations and offers sketch-based interactions to flexibly arrange data items in the 2D space. Even though these studies provided valuable insights, they provide limited support in inspecting and summarizing a group of small multiples, which are important to reveal and remove the mismatches between human concepts and ML semantic dimensions. Some interaction techniques have been proposed to better organize small multiples and facilitate the exploration, such as interactive piling [4, 30, 33] and hierarchical clustering [6, 31]. For example, interactive piling is inspired by physical piles and enables users to effectively group, aggregate, browse, and compare small multiples. However, these interactions are usually designed for specific application scenarios and cannot be directly applied to concept-driven exploration. In Drava, we adapt interactive piling to facilitate the concept-driven exploration of small multiples, especially focusing on the interpretation of semantic dimensions, the mismatch identification between ML semantic dimensions and human concepts, and guidance on refining semantic dimensions.

Third, to better guide user exploration and insight generation, researchers have proposed **interactive ML for visual data exploration**, which learns what visual concepts are important to users from user feedback [5, 10, 15, 32, 61]. For example, Behrisch *et al.* [5] trained a classifier to interactively capture users' notion of interestingness when exploring many scatter plots. This classifier is then used to recommend potentially interesting plots and guide the exploration of large multidimensional data. Cai *et al.*[10] provides an interactive tool that empowers users to refine an ML model by communicating what types of similarities are most important when searching certain medical images. Peax [32] proposes an efficient and accurate query of a certain visual pattern in sequential data by learning from users' binary feedback on samples selected through active learning strategy. However, prior studies mainly use interactive ML to assist with similarity queries, *i.e.*, modeling the similarity between items and user-selected targets. Despite the

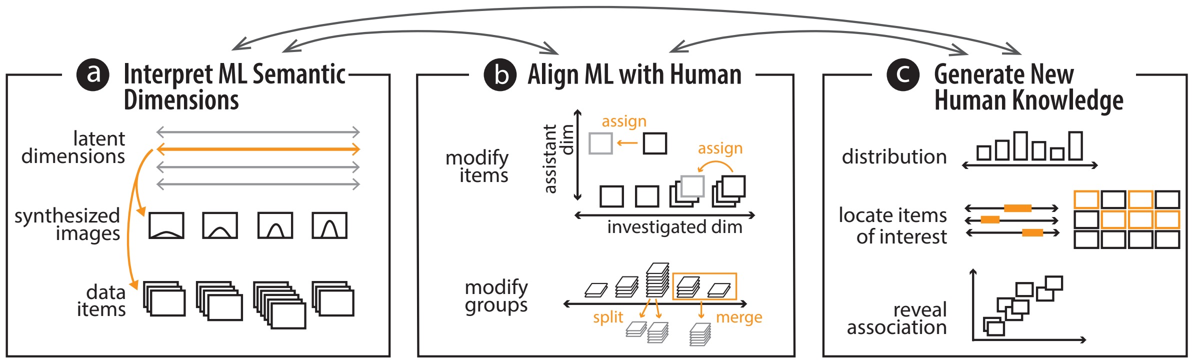

Figure 3: A three-step workflow that guides the application of DRL for the concept-driven exploration of small multiples.

helpful guidance that these studies provide in data exploration, they cannot provide a comprehensive overview of the analyzed data.

Like these approaches, Drava employs learning from user input to provide more precise exploration guidance. Furthermore, Drava provides semantic dimensions and supports summarization, exploration, and analysis based on different visual concepts.

4 WORKFLOW AND TASKS

In this section, we decompose the overall goal of *concept-driven visual exploration using DRL* into three main steps (Figure 3). We discuss the user tasks within each step from two aspects: the characteristics of DRL, as discussed in the DRL literature [8, 22, 28]; and the user needs in visual data exploration, largely informed by the task summarization work in previous studies [16, 33, 37]. These user tasks have been well established in previous studies and can be reused to effectively guide the design of Drava. Moreover, reuses in the task analysis can increase the design quality and reduce expenditure, as recommended in [44, 55, 56].

Step 1: Interpret ML Semantic Dimensions. Since only a subset of the latent dimensions correlates with semantic meanings, users should be assisted to *identify the semantic dimensions efficiently* (**T1.1**). For a specific dimension, users can *interpret its semantic meaning* (**T1.2**) through 1) synthesized images generated by single value traversal of this dimension or 2) data items sorted and grouped by their value in this dimension. A group summary can help users to efficiently understand the semantic meaning of a large number of items, associate it with a human concept, and identify mismatches. Unlike previous studies that group items based on their overall similarities, concept-based analysis requires to group and summarize items based on certain concepts. Therefore, proper aggregations should be provided to *highlight the concept of interest and fade out others* (**T1.3**) when summarizing an item group.

Step 2: Align ML Semantic Dimensions with Human Concepts. Once a mismatch is identified, users modify the semantic dimension to better align it with the human's definition of concepts. Such *refinement should be user-friendly and conducted upon objects that users are familiar with* (**T2.1**), *e.g.*, data items and item groups rather than numerical values of latent dimensions. Meanwhile, *visual cues should be provided to guide and facilitate the user refinement* (**T2.2**), *e.g.*, highlight the items that are grouped wrongly due to a concept mismatch.

Step 3: Generate New Human Knowledge about the Data. Users *explore the data items based on the identified concepts* (**T3.1**) to generate insights about the analyzed items, including the distribution of items on one or multiple visual concepts, the association between different concepts. Such analysis can be further enhanced by *correlating the concepts with other item metadata* (**T3.2**), such as the spatial information and the item labels.

The three steps are interconnected (*i.e.*, arrows in Figure 3). For example, users may directly go to Step 3 from Step 1 if they do not observe obvious mismatches. Users can also go back from Step 3 to Step 2 if they find some semantic dimensions fail to support their analysis tasks and require further refinement. Drava provides a set of dedicated interactive visualizations and algorithms that are closely coupled with this three-step workflow.

5 VISUAL INTERFACE

The user interface of Drava (Figure 4) consists of a *Concept View*, an *Item Browser*, and an optional *Spatial View*. The interactions related to visual piles are based on the design space proposed by Lekschas *et al.* [33], selected, modified, and extended to better reflect tasks described in section 4.

5.1 Concept View

In the *Concept View* (Figure 4a), each latent dimension is visualized as a histogram and a list of synthesized images. The histogram shows the distribution of all items based on their values on the corresponding latent dimension (**T3.1**). Since the exact values of a latent dimension do not have specific meanings, we use synthesized images rather than numbers as the tick labels of the x-axis in the histogram. The synthesized images are generated by the decoder in the DRL model. For one specific latent dimension, the synthesized images are generated using a set of latent vectors whose values only differ on this dimension. These synthesized images illustrate the visual changes associated with the value traversal on the investigated dimension and help users understand its semantics (**T1.2**). Users can filter items based on their values on specific semantic dimensions by clicking on bars of a histogram (Figure 4F).

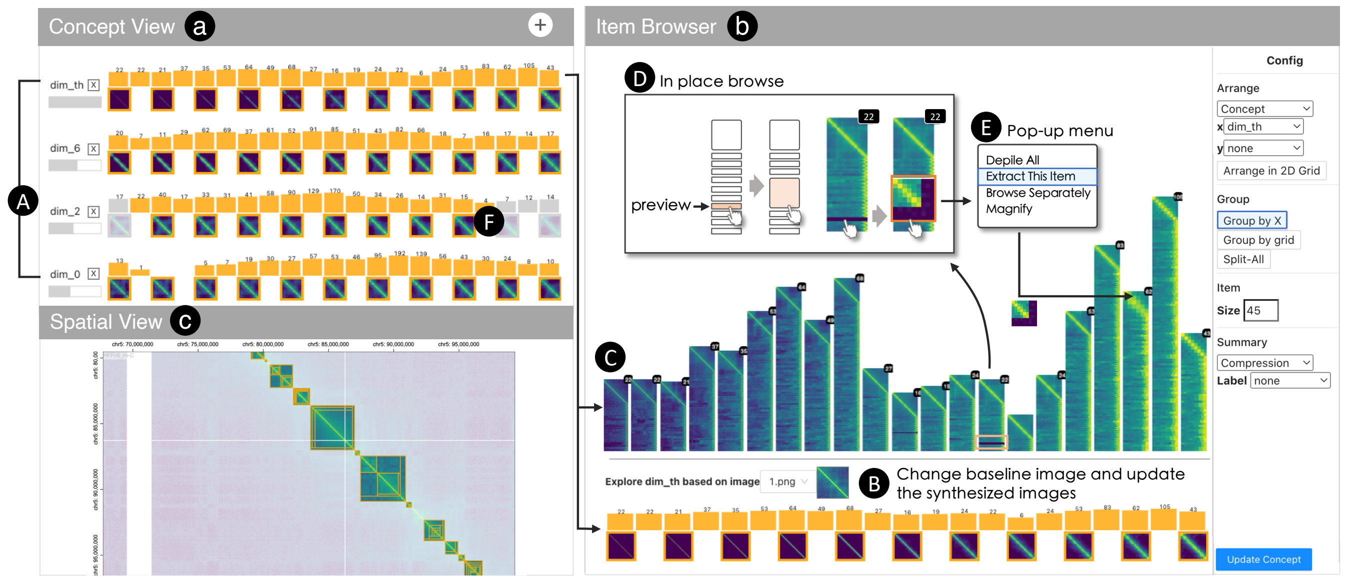

Figure 4: The user interface of Drava. The *Concept View* presents latent dimensions and other metadata as rows. The *Item Browser* enables user exploration of items based on selected concepts from the *Concept View* and can be controlled through the Configuration panel on the right. The *Spatial View* provides context information of the items when applicable.

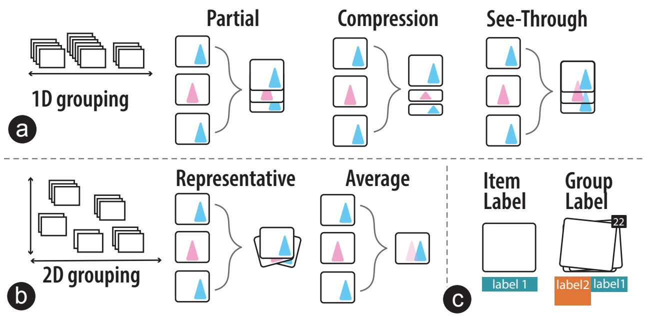

Figure 5: Drava provides various grouping and labeling methods to help users interpret semantic dimension. (a-b) Different approaches to pile up items based on their characteristics. (c) Labels can be added to both items and item groups.

As explained in section 2, only a subset of the latent dimensions are semantic and correlate with human concepts [8, 28]. Therefore, it is important to provide a mechanism that guides users in the exploration of a potentially large number of dimensions. Drava calculates a salience score for each latent dimension (subsection 6.3), indicating how important a particular latent dimension is for the synthesized images. As shown in Figure 4A, all latent dimensions are ranked based on their salience scores, and the normalized score of each latent dimension is visualized by the width of a gray bar (**T1.1**). Users can change the dimension name based on their interpretation of the associated concept to facilitate the following analysis. Users can also remove irrelevant dimensions and add other customized dimensions from the item metadata.

5.2 Item Browser

The *Item Browser* (Figure 4b) layouts all items in a 2D space where users can freely arrange and group items. Arranging items based on their values of certain semantic dimensions enables users to interpret semantic dimensions and understand the item distribution among a certain concept (**T3.1**). A set of synthesized images are added to the $x-$ and/or $y-$axis to guide the interpretation of latent semantic dimensions and the exploration of data items (Figure 4B). Since the visual appearance of the synthesized images largely depends on the latent vector, Drava allows users to select an item and use its latent vectors to generate synthesized images. This interaction enables users to further validate the concept associated with a latent dimension and identify possible mismatches (**T1.2**).

Since the number of items can be large and the items often overlap with each other, effective grouping and summarizing mechanisms are needed. In Drava, users can either manually group items using a lasso selection or automatically group items based on their proximity in the 2D space. Drava provides various methods for summarizing a group of items and revealing abnormal items inside this group (**T1.3**), as shown in Figure 5a-b. When items are arranged horizontally (*i.e.*, 1D grouping), items will be stacked along the vertical direction, and each item will be visualized as an item preview. Users can select the grouping method in the configure panel based on the characteristics of items and concepts.

Labels can also be added to individual items or item groups to incorporate more item metadata into the analysis and investigate their associations with concepts, as shown in Figure 5c (**T3.2**). To investigate more details about an item group, users can browse items by hovering on their item previews (Figure 4D). A pop-up menu, shown upon right-clicking on an item group, enables users to depile this group or browse the items in a separate window.

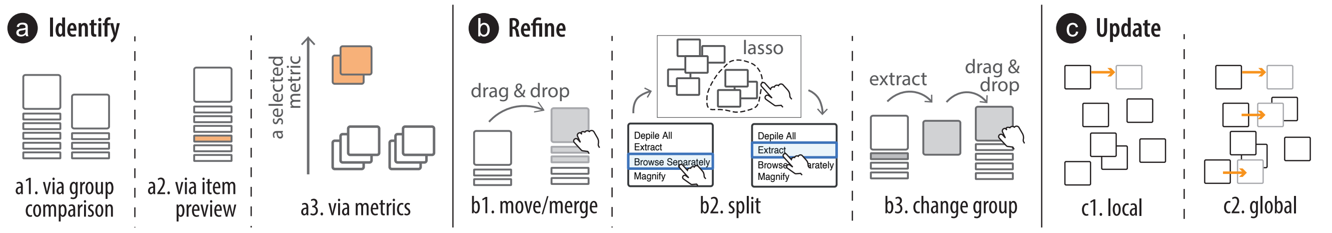

Figure 6: Overview of the interactions and mechanisms that Drava provides for identifying concept mismatches (a), refining items and groups (b), and updating items and underlying models (c).

5.3 Spatial View

The *Spatial View* (Figure 4c) is an optional view for data items that have spatial/context information. For example, in Figure 4, each item indicates a region of interest in a huge genomic interaction matrix and is arranged according to its genomic location. Users can zoom and pan to obtain an overview or inspect further details. The *Spatial View* is coordinated with other views to reveal the correlations between concepts and item context (**T3.2**). When users filter items in the *Concept View*, the corresponding items will fade out in the *Spatial View*.

5.4 User Refinement

Instead of directly modifying the hard-to-interpret latent values, Drava supports refinement towards groups and items (**T2.1**). For one selected semantic dimension D_i, Drava groups items (21 bins by default) based on their values of this dimension d_i to represent the gradual changes associated with this dimension. First, this default group assignment may have inappropriate thresholds, *e.g.*, assigning similar items into two adjacent groups. Therefore, Drava enables users to merge (Figure 6b1) or split (Figure 6b2) groups to construct more meaningful groups according to one concept. More importantly, due to the imperfection of algorithms, the latent values may not accurately depict the concept for certain items, leading to inappropriate x position and group assignment for these items. Users can align the concepts and semantic dimensions by changing the item position (Figure 6b1) and reassigning the group of these items (Figure 6b3).

Several mechanisms are provided to assist users in locating abnormal items and groups (**T2.2**), as shown in Figure 6a. First, users can decide whether to merge or split groups by comparing these groups side by side (a1). Second, Drava enables users to identify abnormal items through previews (a2). For example, as shown in Figure 4C, all items are grouped based on the thickness of their diagonal. Users can locate an abnormal item because its preview is darker than others. Users then examine this item through in-place browsing (Figure 4D), extract it using the pop-up menu (Figure 4E), and drag and drop it to a proper group based its diagonal thickness. Apart from identifying abnormal items through previews, users can also browse a group in a separate window and arrange the items using selected metrics (a3). In our experiments, we found certain metric values are useful in identifying abnormal items, including the reconstruction loss, the deviation of the latent value, the item metadata, and the uncertainty score.

After user refinement, Drava supports two mechanisms, local and global, to update the items and/or the underlying model (Figure 6c). By default, Drava employs a local updating mechanism, which remembers the user refinement, applies it to items with similar latent vectors, but does not modify the underlying model. Similar items are defined by setting a threshold θ to the $L2$ distances of their latent vectors to the that of the refined items. On the contrary, global update initializes and fine-tunes a concept adaptor (subsection 6.2). The values for all other items at this dimension will be updated accordingly by this concept adaptor. The global refinement is triggered by clicking the *update concept* button. Since it is hard for users to label an item with an exact numerical value, global refinement can only be triggered when items are grouped for a certain concept.

6 MODEL SETUP AND IMPLEMENTATION

6.1 Learning Semantic Dimensions using DRL

Our DRL model is based on the β-VAE [22]. The structure of the DRL model is illustrated in Figure 7, encoder (a) and decoder (b). Each convolution block consists of a convolution layer, a batch normalization layer, and a leaky ReLU (Rectified Linear Unit) activation. The decoder architecture is the transpose of the encoder. Following the practice in [53], we do not include Max Pooling layers by setting $stride = 2$ in the convolution layer. All usage scenarios in this paper use this structure and only vary in 1) the number of convolution and transposed convolution blocks, 2) the kernel size and the number of channels of the convolution and transposed convolution layers, and 3) the output size of the fully connected layer (*i.e.*, the number of dimensions for the latent vector).

We use the loss function proposed by Burgess *et al.* [8], which progressively increases the information capacity during the training process. An Adam optimization is used to train the model. Note that we use the mean μ of the normal distribution learned by the encoder rather than the sampled $z \sim \mathcal{N}(\mu, \sigma^2)$ as the latent vector for the input data, which enables deterministic latent values for each data item.

Even though we implement and evaluate Drava using β-VAE, the proposed framework can be easily adapted to other VAE-based DRL models, such FactorVAE [28] and β-TCVAE [11].

6.2 Concept Adaptor

The concept adaptor is a lightweight model that modifies semantic dimensions based on user refinements. For each semantic dimension, one concept adaptor will be generated if users use this

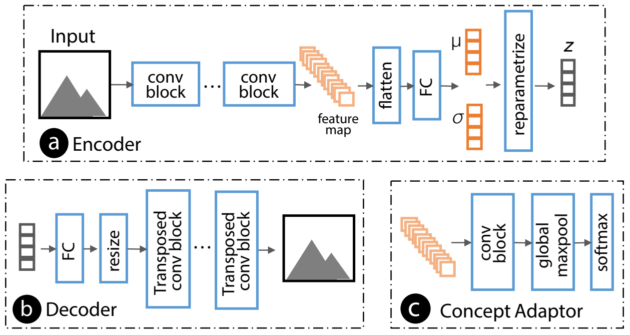

Figure 7: Architecture of the encoder, the decoder, and the concept adaptor.

dimension to arrange items, refine the item groups, and apply a global update. Since it is hard for users to associate the concept with an exact numerical value, the concept adaptor is only used to refine a concept for already grouped items. In other words, the concept adaptor is a multi-class classifier. The concept adaptor uses the feature map generated by the encoder hidden layer as input and predicts the group that the input item should belong to.

Figure 7c illustrates the structure of the concept adaptor. The convolution block contains a convolution layer (kernel size =4, stride =2) and a batch normalization. The convolution layer has n output channels where n equals to the number of item groups. A $n \times 1$ vector will be obtained after a global max pooling layer and then feed into a softmax function. A cross entropy is used to calculate the loss. An Adam optimization is used to train the model.

Once items are grouped based on the values of one dimension, users can initialize a concept adaptor accordingly. The training ends when the validation loss does not decrease. For all the datasets used in section 8, the initialization took less than two minutes on a machine with one Tesla K80 GPU. After users have refined the item groups (*i.e.*, change the classification label) for some items, the concept adaptor will be fine-tuned accordingly. During the fine-tuning, we increase the weight of the items that have been refined by the users. For the back-end models, only the concept adaptor is updated with user refinement, while the encoder and decoder are fixed. For the data items, only the values of the specific latent dimension (*i.e.*, dimension used as the x axis) will be updated by the concept adaptor, while other dimensions will remain the same.

6.3 Dimension Ranking

We rank all latent dimensions based on their importance to help users quickly locate semantic dimensions. Inspired by the salience scores used in interpretable ML [26], we gauge the importance of a latent dimension via the sensitivity of the reconstructed image $\hat{x}$ to changes in the magnitude of a latent dimension z_i. However, directly using the gradients $\partial \hat{x} / \partial z_i$ has several issues. First, it is a local importance score that is calculated for a particular reconstructed image. Second, it is a vector rather than a scalar value and can be hard to compare across. Third, it counts pixel-level differences that are not necessarily consistent with human perception. To solve these issues, we use a simple but effective method, *i.e.*, averaging

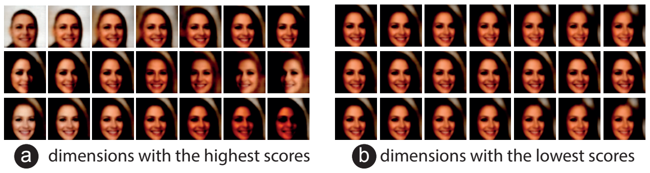
a dimensions with the highest scores
b dimensions with the lowest scores

Figure 8: Dimensions (rows) with different salience scores (a–b). The visual changes are clearer when changing the values of the dimensions with the highest salience scores (a) compared to the dimensions with the lowest scores (b).

the importance score across output dimensions and across a set of sampled latent vectors. To mimic human perception of the synthesized images, we use latent vectors of the synthesized images as samples. Instead of using the reconstructed output, we use the feature maps generated by the second last layer of the decoder, aiming to capture high-level features rather than pixel-to-pixel differences.

$$\frac{1}{m \times n \times k} \sum_{m,n,k} \left| \frac{\partial L_{m,n}(z_k)}{\partial z_{k,i}} \right|$$

Where z_k is the k_{th} sampled latent vector, $z_{k,i}$ is its value at dimension i, $L_{m,n}(z_k)$ is the feature map (m, n) of the second to last decoder layer.

The salience score serves as a useful indicator for semantic dimensions (Figure 8). Theoretically, a dimension with a high salience score is not necessarily equal to a semantic dimension, *e.g.*, a dimension is not semantic but significantly influences the output. However, the DRL model will minimize the existence of such dimensions by disentangling features and encoding them as separate dimensions. Ranking all dimensions based on salience scores can help users exclude many latent dimensions that do not contribute to the output and have little semantic meanings (*e.g.*, Figure 8b).

6.4 Implementation

The implementation of Drava includes a front-end for interactive visualization and a back-end for data storage and the DRL model. The front-end is implemented in TypeScript using React [17], Piling.js [33], and Gosling.js [39]. The visualizations are rendered using SVG, Canvas, and WebGL. The back-end DRL model and concept adaptor are implemented in Python with PyTorch [45]. The front-end and back-end communicate via a Flask [19] web server built in Python. Users can easily apply Drava to their own datasets through two YAML configuration files that configure the back-end model training process and the front-end interface, respectively. The source code and documentation are available at https://qianwen.info/DRAVA/.

7 EXPERIMENTAL VALIDATION

In this section, we evaluated the back-end model in Drava from three aspects: 1) the *representativeness* of the latent vector, 2) the *semantic meaning* of individual latent dimensions, and 3) the improvements from *concept fine-tuning*. Previous studies either focused on assessing the disentanglement of latent dimensions [8, 22, 28] or overlooked the possible mismatches between human concepts and semantic dimensions [18, 20, 59]. Therefore, it is important

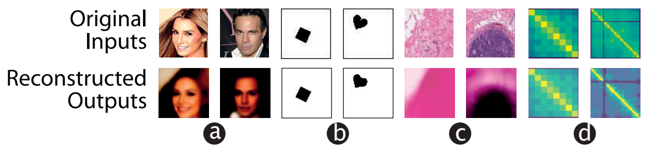

Figure 9: Examples of reconstructed outputs on four different datasets (a–d). The first row shows the original inputs and the second row represents the reconstructed outputs.

dataset	dsprites			CelebA	
concept	pos_x	pos_y	scale	smiling	bangs
random guess	0.333	0.333	0.333	0.5	0.5
Drava (no human refine)	0.87	0.93	0.62	0.70	0.77

Table 1: Semantic meaning of individual latent dimensions. We compare Drava (*i.e.*, using the value of one latent dimension to classify the corresponding concept) with random guesses on five concepts from two datasets. The results show that the latent dimension value could effectively indicate the corresponding concept.

to validate the quality of these semantic latent vectors and their fine-tuning mechanism.

Representativeness of the Latent Vector. We used the reconstruction quality to show whether the latent vectors can capture all the important visual features of the input data. Figure 9 exemplifies the reconstruction quality of the latent vectors for the four datasets used in the application scenarios (section 8). Instead of the absolute similarity or the realism of the reconstructed images, we focused on evaluating whether the reconstructed images are able to capture important concepts. For the relatively simple *dsprites* shapes dataset (b), the model is able to generate images that are very similar to the input data. For more complex datasets (a, c-d), even though some details in the input data are missing, the model can reconstruct salient concepts.

Semantic Meaning of Individual Latent Dimensions. To evaluate whether a single latent dimension can sufficiently depict a concept, we classified items based on their values on a certain semantic dimension and reported the classification accuracy. Specifically, for n classes belonging to a concept, $n-1$ thresholds are learned to classify items. For example, the "smiling" concept has two classes, smiling and not smiling. We first identified a latent dimension D_i that is related to the "smiling" concept. We then classified each item based on whether its value on this dimension d_i is larger or smaller than a threshold thr, which was chosen to maximize the classification accuracy of all items. We used the *dsprites* and the *CelebA* datasets because they have labels for a diverse set of concepts. The results in Table 1 demonstrated that the latent dimension value could effectively represent the corresponding concept but also showed space for further improvement.

Improvements from Concept Fine-tuning. We evaluated the fine-tuning mechanism of the concept adaptor by comparing the classification accuracy of a specific concept before and after user refinement. This evaluation used the "scale" concept from the *dsprites* dataset and the "smiling" and "bangs" concepts from the *CelebA* dataset, because they have relatively low accuracy without any human refinement (Table 1). We chose an active learning method as the baseline for evaluating the concept adaptor. The baseline had the same architecture as the concept adaptor. We used simulated user feedback to obtain reproducible results in a variety of settings. Following the common practices in evaluating interactive machine learning [13] and active learning [49], we simulated user feedback as an oracle (*i.e.*, always providing correct labels to the queried items). Both the concept adaptor and the baseline used the same simulation at each iteration but with different initialization. The active learning baseline is initialized with 5% labels. The concept adaptor is initialized with no labels but the same item groups as that in Table 1. Such an initialization simulates how users would divide items into several groups for a specific concept based on their latent dimension values. At each iteration, N items were refined (for the concept adaptor) or labeled (for the baseline) and models were trained until the validation loss did not decrease, which typically took around 10-20 epochs and less than 20 seconds. We experimented with three metrics for selecting the N items: uncertainty scores of the classification, the standard deviation of the latent dimension value, and differences between the latent dimension value and the classification threshold. We found that refining items with the highest uncertainty score led to the best model performances. Even though we used an oracle to simulate user refinement here, real-world users can easily examine and label these items in Drava by selecting a metric of interest as the y axis in *Item Browser*.

We ran experiments under three settings: N = 1%, 2%, and 5% of the items. A total of 15 iterations were performed for each experiment. The results in Figure 10 were obtained by averaging the results of three experiments. First, the increased accuracy indicated

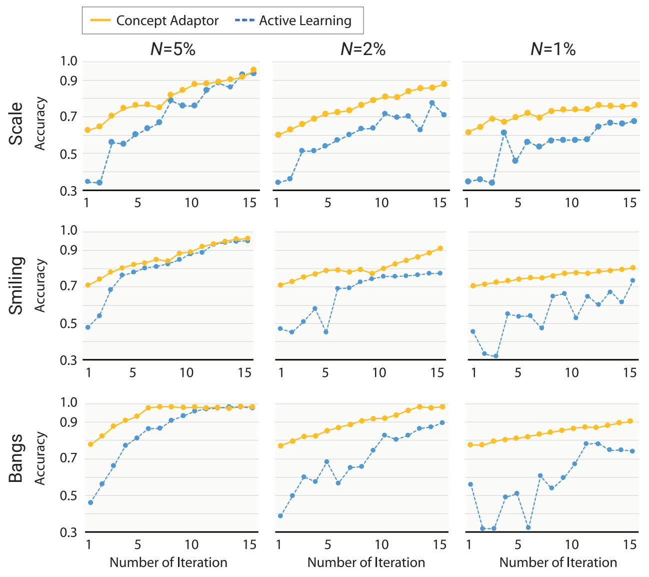

Figure 10: We compared our concept adaptor with active learning (baseline) on three different concepts (*i.e.*, scale, smiling, and bangs) under three conditions (*i.e.*, N = 1%, 2%, 5%). Each line graph shows the accuracy over 15 iterations. The concept adaptor (yellow) overall showed higher accuracy.

that the concept adaptor helped align a concept and a semantic latent dimension. Compared with the baseline, the concept adaptor generated more accurate concepts by leveraging the values of the semantic dimension. Second, the curves of the concept adaptor were more smooth than the baseline, indicating a more stable improvement over iterations. Third, while the concept adaptor and the baseline required the same amount of user effort at each iteration (*i.e.*, the same N and the same user simulation), the concept adaptor required less user effort at the initialization than the baseline (*i.e.*, drawing two or three lasso selections vs. labeling 5% of the items one by one). Fourth, it was not surprising that the difference between the concept adaptor and the baseline model decreased with the increase of N and iteration steps. The advantages of the concept adaptor mainly result from using the semantic dimension values. As more and more items are labeled, these semantic dimensions become less useful in describing a concept.

8 APPLICATION SCENARIOS

In this section, we present four application scenarios of Drava using one simulated dataset and three real-world datasets. For all four application scenarios, the DRL model is trained on the whole dataset with no labels used. The four application scenarios are conducted under collaboration with domain users, including two postdoctoral researchers on computer vision (P1 and P2, both for subsection 8.1 and subsection 8.2), two researchers on genomic analysis (P3 and P4, for subsection 8.3), and a professor on histopathological image analysis (P5, for subsection 8.4). For each application scenario, we first provided a tutorial to introduce the functionalities of Drava. We then demonstrate our analysis and validate our findings with the participants. Participants can freely explore Drava and conduct additional analysis on the provided dataset. We further collected qualitative feedback about Drava from the participants.

8.1 Simple Shapes

Data, Model, and Analysis Overview. This scenario uses the *dsprites* dataset [40], which consists of three types of simple shapes (*i.e.*, square, ellipse, heart) with different scales, positions, and orientations. We uniformly sampled 1,000 items. The DRL model has four convolution blocks, each of which has 32 channels, a kernel of size 4, and a stride of 2. The latent vector has 8 dimensions. Even though this is a simple dataset, it can work as a proxy for more complicated datasets, such as the bounding boxes in object detection or the masks for cell segmentation. In this scenario, we explore the distribution of items according to concepts related to position and size, which are identified, validated, and refined by users.

Arranging Items based on Concepts of Interest. To start with, we display all items in a 2D space using UMAP, a dimension reduction method that is commonly used for visualizing items with latent vectors. While the UMAP successfully put items with similar shapes and scales close to one another, the shape position information is mostly ignored, as shown in Figure 11a. The position information can be important for some analysis tasks, *e.g.*, object detection in autopilot.

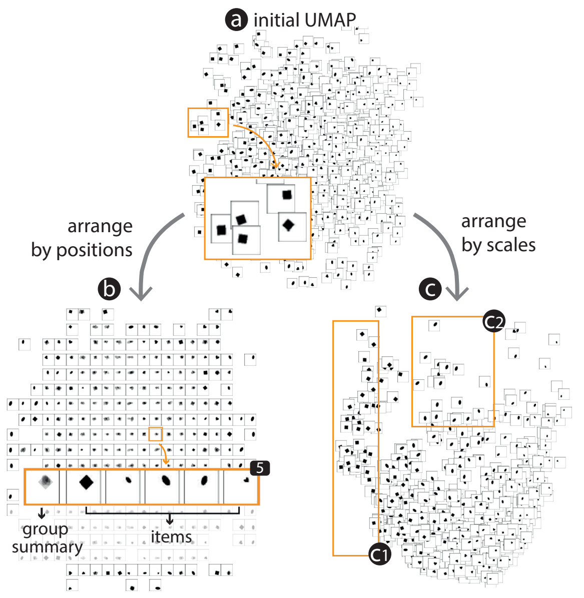

Figure 11: The application scenario using the simple shape data. (a) A UMAP projection puts images together even though the positions of shapes are different. (b) Users can arrange images based on shape positions. (c) Images are arranged based on the scales of shape, but the left-most side is mostly squares (c1), indicating the need for user refinement.

Based on the synthesized images in the *Concept View*, the position-related information is successfully extracted in two top-ranked dimensions, which we rename to `dim_x` and `dim_y` (Figure 1e). As shown in Figure 11b, all items are arranged and grouped based on the x and y position of the shape. We choose the *average* method to summarize a group, which enables us to inspect the positions of shapes without browsing individual items one by one.

Refine a Semantic Dimension. The scale, *i.e.*, size, of the shapes is also a vital piece of information for some analyses and has been successfully extracted in a latent dimension (named as `dim_size`). We verify this semantic dimension in the *Item Browser*, setting `dim_size` as x axis and its deviation σ as the y axis. While all items are sorted based on their size from left to right, we find that items on the left side are all squares (Figure 11c1). We speculate this is because an ellipse or a heart, even with the same scale, is smaller than a square in terms of absolute pixel areas.

To obtain a semantic dimension that better matches the analysis purpose and indicates the scale regardless of shape types, we refine `dim_size` using the concept adaptor. We set the "reconstruction loss" as the y axis to reveal abnormal items (Figure 11c2) and modify the x position of these items. We then group the items into three main groups, indicating large, medium, and small scales, respectively. After clicking the *update concept* button, the concept adaptor is initialized based on our grouping. We further refine these groups using the *browse separately* function, examining each group and updating the group mainly by moving items of ellipse or heart shape from the medium group to the large group. After several updates, we click the *update concept* button again. The concept adaptor is fine-tuned based on the refined item groups and updates the grouping of all items. After several iterations, we obtain three groups that more accurately reflect the scale of shapes without the influence of shape types (refer to section 7 for quantitative results).

8.2 Celebrity Images

Data, Model, and Analysis Overview. This usage scenario uses the celebrity images from the *CelebA* dataset [38]. The DRL model is trained on the complete dataset, and we randomly sample 1,000 items for the exploration in Drava. The DRL model has five convolution blocks, each of which contains a kernel of size 3, a stride of 2, and 32, 64, 128, 256, and 512 channels, respectively. The latent vector has 20 dimensions. In this scenario, we investigate the quality of the *CelebA* dataset based on the diversity, balance, and association of the concepts in this dataset.

Examine Dataset Diversity. Collecting a diverse dataset is important in ML to improve the model performance in real-world deployment and avoid algorithmic discrimination of certain populations [65]. The concepts extracted by Drava offer an effective approach to investigating the diversity of a dataset.

Based on the synthesized images in the *Concept View*, we can affirm that diverse visual concepts exist in the analyzed data items. The analyzed items vary in a number of aspects, including emotional expression, gender, angle, skin color, background color, hair length, and hairstyle. To further verify our interpretation of the semantics of individual dimensions, we can interactively change the latent vector to update the synthesized images and group items based on their latent values at a selected dimension (Figure 1d).

Investigate Dataset Balance. We then analyze the item distribution along individual concepts as dataset imbalance can introduce bias during model training and impair model performance. For example, for the "skin color" concept, a dataset with a large number of items with fair skin and only a small number of items with dark skin can lead to an ML model that has poor performance on the latter. As shown in Figure 12a, we arrange and group items based on `dim_9`, which captures skin color based on the synthesized images. Through browsing items in these groups, we find only the right several groups include people with dark skin (a1), indicating a relatively small portion. When we browse individual items in each group (a2), we can find that this portion is even smaller since the model considers people with dark skin and people with shadows on their faces as similar. This observation implies an imbalance related to skin color, which may introduce a bias into a model trained on it.

Confirm Concept Association. Based on Figure 12a, we suspect a correlation between dark skin and dark background. Such correlations can be treated as causalities by ML models [68] and need to be avoided. We confirm this suspicion by arranging all items using `dim_9` (skin tone) as the x-axis and `dim_16` (background darkness) as the y-axis. The resulting distribution (Figure 12b) dispels our suspicion. Even though the distribution is not uniform, the dataset contains both items that have fair skin and dark background (b1) and items that have dark skin and light background (b2).

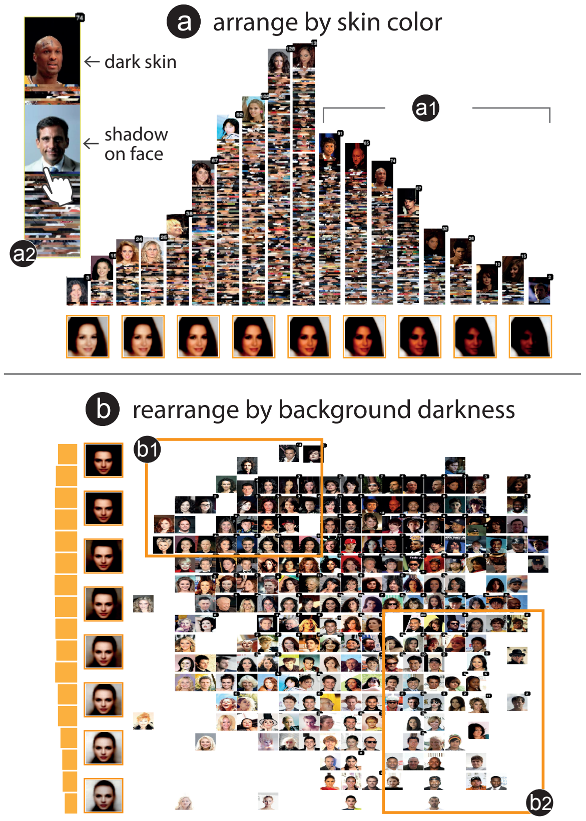

Figure 12: The application scenario using celebrity images. (a) Items are grouped based on a visual concept that is related to skin color. (b) Items are then rearranged by adding another visual concept that is related to the background darkness as the y axis. The dataset contains both items that have fair skin and dark background (b1) and items that have dark skin and light background (b2).

8.3 Genomic Interaction Matrix

Data, Model, and Analysis Overview. This usage scenario uses a genome interaction matrix for the HFFc6 cell line published by Rao *et al.* [48]. The matrices describe the chromatin interactions between different genomic locations, which is related to the physical folding of DNA that affects the regulation of gene expression. In a genome interaction matrix, rows and columns represent genomic locations, and the color intensity indicates the interaction probability between a pair of locations. Experts typically examine regions of interest (ROI) that have unique visual patterns and indicate specific biological events. Since the size of the matrix is huge, *i.e.*, 3 billion × 3 billion for human genomes, this analysis process is often laborious and time-consuming.

We generate small multiples for one specific type of ROI called Topologically Associated Domains (TAD), which are visually represented as squares that are presumably organized hierarchically. We first extract TADs from the interaction matrix using OnTAD [3] and then use the DRL model to generate a latent vector for each TAD. We demonstrate Drava using the 855 TADs extracted from chromosome 5 of the HFFc6 cell line. The DRL model has three convolution blocks with filter sizes of 7, 5, 3 and channel sizes of 32, 64, 128, respectively. The latent vector has 8 dimensions.

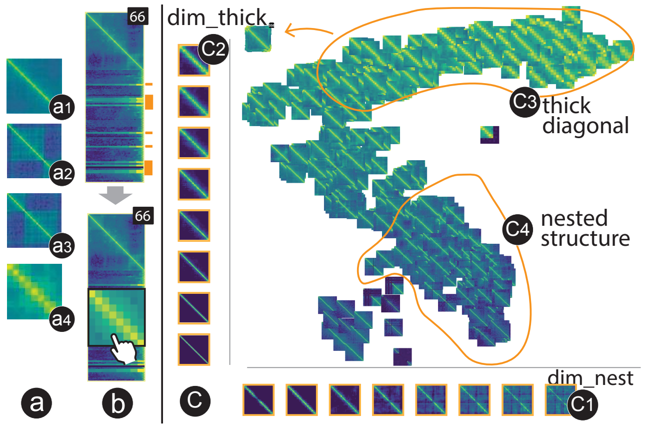

Figure 13: The application scenario using a genomic interaction matrix. (a1-a4) Four representative items vary on three concepts: the thickness of the diagonal (a1 and a4), the presence of nested squares (a1 and a2), and the asymmetric structure of the nested squares (a2 and a3). (b) A group has both the items with nested squares and the items with thick diagonals (orange marks). Each item is displayed upon mouse hovering. (c) Arranging items using `dim_nest` as x axis and `dim_thick` as y axis clearly separates these two concepts.

In this scenario, we investigate different types of TADs by identifying, validating, and refining concepts that correspond to important visual patterns of TADs. Guided by these concepts, we are able to locate different types of TADs and examine the spatial distribution of these TADs on the whole genome.

Understand Data through Concepts. The visual appearance of TADs in a heatmap can serve as effective proxies of the underlying data patterns and biological events [3, 30]. Therefore, by interpreting the visual concepts, we can inspect how the underlying data and the associated biological events vary among the analyzed items. In the *Concept View*, we identified three dimensions of interest. `Dim_7` (renamed as `dim_thick`) indicates the thickness of the diagonal (*e.g.*, an item changing from Figure 13a1 to a4), which is related to the resolution of the TAD on the matrix since we resize all TADs into a fixed pixel size for the DRL model. `Dim_0` indicates the asymmetry of the nested TAD structure (*e.g.*, an item changing from Figure 13a2 to a3). `Dim_6` (renamed as `dim_nest`) corresponds to whether a TAD data item contains additional nested squares (*i.e.*, nested TAD such as a2, a3) or not (*i.e.*, single TAD such as a1, a4). Other dimensions are either hard to interpret because there is little variation in the synthesized images or can not be associated with meaningful domain insights. `Dim_thick` and `dim_nest` are the top two dimensions based on the salience scores, indicating the usefulness of the dimension ranking. The three dimensions (`dim_thick`, `dim_0`, `dim_6`) correspond to important attributes of TADs, as described by An *et al.* [3].

Verify and Refine Concepts. After obtaining a basic understanding of the semantic meaning of each dimension through their synthesized images, we further verify the three concepts one by one through grouping and browsing data items. Interestingly, we find that `dim_nest` confuses the thickness of the diagonal with the nested structure of TADs. As shown in Figure 13b, items are grouped based on `dim_nest` and use "partial" to generate item previews. Users can identify items with thick diagonals from the item preview (as annotated by the orange marks) and examine them in detail by hovering over them. **This issue can hardly be revealed through the synthesized images (Figure 13c1), which are widely used as the only method to interpret semantic meanings in previous literature [18, 59].** This observation shows the importance of further verifying a concept base on data items and the need for user refinement.

Since `dim_thick` can indicate the TAD size, we use it as the y axis to help refine the concept associated with `dim_nest`. As shown in Figure 13c, items arranged in different vertical positions based on their diagonal thickness, enabling successful separation of nested TAD (*e.g.*, a2, a3) from single TADs with thick diagonal (*e.g.*, a4). Users can refine `dim_nest` by a lasso selection on all single TADs that have large `dim_nest` values and moving them to the left-most position (*e.g.*, assigning them a small value for `dim_nest`), as shown in Figure 13c3. The refinement is recorded using the local updating mechanism and applied to similar items.

Locate items of interest. After the refinement, users can easily locate nested TADs in Figure 13C4 through a lasso selection. They can also filter these TADs based on `dim_thick` and `dim_nest` using their histograms. The nested structure in TADs is important to understand the boundary usage in gene regulation [3]. For this purpose, these identified items can be further examined in the *Spatial View* (Figure 4), which reveals the genomic locations of these TADs and associated them with other context information (*e.g.*, chromatin accessibility).

8.4 Breast Cancer Specimen

Data, Model, and Analysis Overview. This usage scenario uses breast histopathology images downloaded from [43]. This dataset contains 277,524 patches (50 × 50 pixels) extracted from stained whole mount slide images of breast cancer specimens from 162 patients scanned at 40x magnification. The DRL model is trained on the whole dataset. In this usage scenario, we explore the 1,745 image patches from one patient. The DRL model has five convolution blocks, each with a kernel of size 3 and 32, 64, 128, 256, and 512 channels, respectively. The latent vector has 12 dimensions. In this scenario, we examine the presence of cancer cells in these items and analyze the performance of a classification model. Specifically, we identify concepts and associate them with domain semantics. We then use these concepts to describe the characteristics of hard-to-classify items.

Interpret Visual Concepts and Assign Domain Semantics. We first visualize all the items using UMAP (Figure 1a). However, the UMAP projection is not ideal since it is based on the overall similarities and considers some irrelevant information, such as the position of tissue patches and the orientation of tissue patches.

Therefore, we check the *Concept View* to find dimensions that can indicate concepts with domain semantics. Based on the synthesized images, we speculate that `dim_5` is related to the density of tissues and `dim_2` is related to the color of the stained tissues. Our interpretation of these two dimensions is further confirmed by examining the grouped items in the *Item Browser*. As shown in

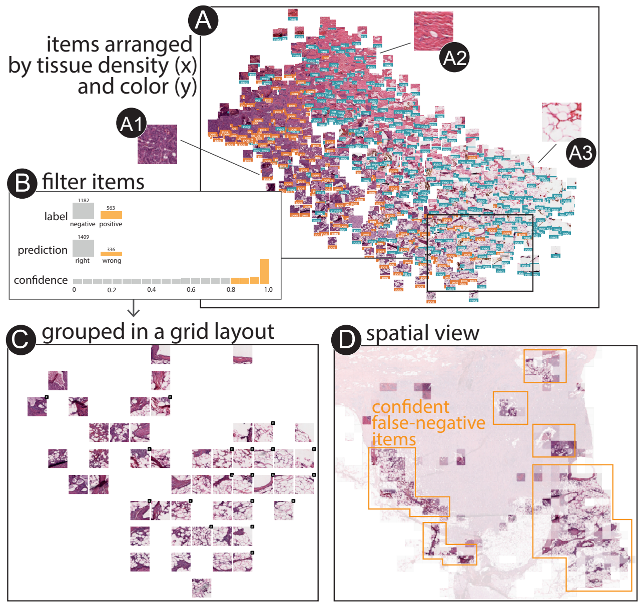

Figure 14: The application scenario using the breast histopathology images. (a) Arranging image patches from breast cancer specimens based on concepts learned by Drava shows a strong association between the presence of IDC (the color of item labels) and the two visual concepts, *i.e.*, the tissue density (a2 vs. a3, the x axis) and the tissue color (a1 vs. a2, the y axis). We further (b) filter these items and (c) display them in a grid layout to identify confident false-positive predictions without visual clutter. (d) The spatial view enables us to locate the items in the original whole-mount slide image.

Figure 1b, when all items are arranged based on dim_5, items on the left side have almost no white space, indicating a high tissue density, while items on the right side have more white spaces, indicating loose tissues or fatty tissues. When all items are arranged based on `dim_2`, items on the left side have a more purple hue while items on the right side have a more pink hue. We then rename `dim_5` as `dim_density` and `dim_2` as `dim_color`.

We arrange all items using `dim_density` as the x axis and `dim_color` as the y axis and then add a label for each item from the item metadata to indicate whether this item contains Invasive Ductal Carcinoma (IDC), a subtype of breast cancer cells (*i.e.*, orange and blue item labels in Figure 14a). As shown in Figure 14a, there is a strong correlation between the presence of IDC and the two visual concepts mapped on the x and y axes. We group items (Figure 1c) to reduce the visual clutter. Items with purple and dense tissues (Figure 14a1) are more likely to contain IDC (*i.e.*, orange labels) while items that are closer to pink (a2) and contain less dense tissue (a3) are less likely to contain IDC (*i.e.*, blue labels). This association is further confirmed by a pathologist. Even though the identification of cancer cells needs to consider a variety of factors, the color and the tissue density are strong indicators of the presence of cancer cells. Cancer cells are typically dense, which leads to less white space, and have larger and darker nuclei than normal cells, which leads to more purple color.

Identify Hard Examples for IDC Identification. Identifying regions in the whole mount slide image (*i.e.*, items in our analysis) with IDC is an important task for pathologists to assign an aggressiveness grade to cancer. Since `dim_dense` and `dim_color` are related to the identification of cancer cells, we further analyzed how they influence on the prediction of IDC in an ML model. We train an IDC classification model by fine-tuning a ResNet34 model, as described in [52], and record the model prediction and confidence score for each item.

Confident wrong predictions and false negatives are more consequential in real-world deployment [9], as patients may fail to receive the treatment they need. Therefore, we are especially interested in false-negative prediction with high confidence scores. We import item metadata to the concept view and filter items accordingly, *i.e.*, ground truth = positive, prediction = negative, confidence score > 0.8, as shown in Figure 14b. According to the *Item Browser* (Figure 14c), the filtered items are close to each other in the *Item Browser*, containing tissues that are not very dense and have a more purple hue. Since items with cancer cells usually contain dense tissues, this may explain why the classification model makes very confident but wrong predictions. We further examine the original spatial positions of these items in the *Spatial View* (see Figure 14d), where other items are faded out with a semi-transparent white mask. We find the items of interest (*i.e.*, non-masked items) are from regions where fatty tissues are surrounded by cancer cells, as shown by the orange boxes. This can explain why these items have many white spaces and only contain a small number of cancer cells. This observation is valuable for understanding and improving this IDC diagnosis model. First, it indicates when and where the IDC prediction model tends to make confident false negative predictions and a double-check from human experts is needed. Second, the training strategy can be modified accordingly (*e.g.*, increasing the sample weight of these loose and purple tissues) to improve the model performance.

8.5 User Feedback

We collected qualitative user feedback about Drava from the collaborated domain users.

Participants commented that Drava provided *"an attractive addition"* (P5) to the current analysis methods. They liked the comprehensive user interaction provided by Drava. P4 commented that *"the item preview is engaging and useful"*. Participants (P1, P4, P5) commented that it is not always easy to interpret a semantic dimension using one set of synthesized images. Therefore, the functionalities to generate synthesized images for a given baseline and to summarize item groups for a certain dimension are helpful. All participants agreed that Drava provided helpful guidance in interpreting and refining the ML semantic dimensions.

The participants also provided valuable suggestions for further improvements. While some dimensions were reported as *"easy to associate with human concepts"*, participants also complained that some dimensions had unclear semantics and were hard to interpret. This issue might be caused by the entangled concepts (section 2) or the quality of the synthesized images. Instead of manually changing baseline images for the synthesized images (Figure 4B), P3 suggested that Drava should recommend several baseline images to facilitate the interpretation of semantic dimensions. P1 and P2 were concerned about the extent to which their refinements will influence the back-end model. P1 stated that refining item groups without updating the back-end model (*i.e.*, a local update) made him *"feel safer and in control"*. Such concerns about automation are consistent with the observations in previous studies [64]. On the other hand, P1 also agreed that the local update can be inefficient and that updating the back-end model is necessary when analyzing a large number of items. P1 and P2 both provided suggestions for improving the global update mechanism, such as annotating how items change after updating the concept adaptor.

9 DISCUSSION

9.1 The Scope of Drava

Dependence on DRL Performance and Data Quality. The concept-driven exploration provided by Drava is based on interpreting, refining, and utilizing semantic dimensions. Therefore, Drava's capability depends on what semantic dimensions a DRL model can learn, which highly relies on the DRL model performance and the data quality [28]. Drava may fail to capture the desired concepts in the semantic dimensions due to the limited capabilities of the model or the low quality of the dataset. We believe that advances in DRL will further empower Drava and provide more opportunities for concept-driven data exploration. Additionally, the concept adaptor in Drava enables users to improve an unsatisfied concept through user refinement. In the worst-case scenario where the desired concepts can not be learned by the DRL model, Drava can serve as a pure interactive active learning tool that learns a concept merely based on user labeling.

Visual Complexity of Concepts. Apart from DRL performance and data quality, whether a concept can be identified in Drava is also related to its visual complexity. Here, a visually complex concept indicates an abstract or subjective concept that has diverse visual representations, which makes it hard to visually summarize and interpret the concept via either the synthesized images or interactive piles. For example, in the *CelebA* dataset, some concepts are simple and have clear visual representations (*e.g.*, black objects near eyes for a "sunglass" concept), but other concepts are rather complex and involve varying visual presentations (*e.g.*, "attractive" can be related to either short or long hair, oval or round face shapes), making it hard to be visually summarized.

Format and Characteristics of Data Items. To achieve the concept-driven exploration in Drava, data items need to fulfill two requirements. First, the data items must be visually perceivable for humans. Image datasets naturally fulfill this requirement. For other types of datasets (*e.g.*, sequences, matrices), a workaround is to visualize the dataset and use the visualization (or segments of the visualization) as data items. For example, in subsection 8.3, we convert a genome interaction matrix dataset into a heatmap visualization and treat each ROI in the heatmap as a data item. Second, these data items need to have similar appearances and share the same concepts, as shown in section 8. Data items with dramatically different appearances not only make it challenging for the ML model to learn and extract concepts but also results in high cognitive loads for users to identify and validate concepts. For example, Drava can not be applied to the ILSVRC dataset [50], which contains diverse images depicting 1,000 different object categories.

9.2 Human Factors in Drava

Human factors play an important role in human-in-the-loop AI tools [2, 10, 65]. Here, we discuss two important human factors in Drava, *i.e.*, cognitive biases and cognitive load, including their impacts, our design considerations for mitigating the impacts, and the limitations of the current design.

Cognitive Bias. ML models do not know what a human concept is. It is the users who associate the concepts of humans with the semantic dimensions of ML. As a result, the interpretation and refinement of the semantic dimensions can be influenced by users' cognitive biases (*e.g.*, confirmation bias, anchoring bias, and availability bias). To facilitate the user interpretation, Drava supports concept validation through various interactions (*e.g.*, changing the baseline image, arranging and piling items) rather than merely relying on the observation of a set of synthesized images. We also plan to support hypothesis generation and testing [60] to further reduce misinterpretation. However, Drava does not have mechanisms that are specifically designed for minimizing cognitive biases. Future studies are needed to systematically investigate the causes of and the solutions for cognitive bias in human–AI collaboration.

Cognitive Load. While more latent dimensions will potentially enable the model to capture more meaningful concepts, it will also increase the cognitive load of users. Drava alleviates this issue by enabling users to rank dimensions based on their salience scores and remove less relevant dimensions. We have successfully tested Drava in application scenarios with at most 32 latent dimensions. A large number of dimensions (*e.g.*, 100) can challenge the cognitive capacity of users and undermine the usability of Drava. Like other hyperparameters in ML, the number of latent dimensions needs to be carefully selected to strike a balance between the representative of the latent dimensions and the cognitive load of the users. Promising directions for reducing the cognitive load include progressively revealing the information [62] and tracking provenance data [14].

9.3 Relation to Dimension Reduction Methods.

The application scenarios present examples where the item arrangement based on a dimension reduction method (*i.e.*, UMAP) fails to fill the analysis needs. Particularly, Drava enables visual exploration and analysis that focuses on the similarity of certain concepts rather than overall similarity. Drava complements the widely used dimension-reduction-based visual exploration tools. Drava is most suitable for analysis scenarios in which data items are similar (*i.e.*, share multiple concepts) and the analysis concentrates on specific concepts. Dimension reduction projection (*e.g.*, t-SNE, UMAP) is still an effective method for visualizing latent vectors, especially when the items form distinct clusters, and when the analysis focuses on overall similarity among items.

9.4 Scalability of Rendering and Interaction.

The rendering scalability of Drava is mainly limited by its rendering engine in the *Item Browser*, which is built upon Piling.js [33]. The *Item Browser* can handle the rendering of and the interaction with 2,000 items with reasonable performance: the *Item Browser* can be initialized in less than 15 seconds and perform the interaction animation in no less than 50 frames per second on a laptop (MacBook Pro, 2020). Data loading is only performed when users open the tool for the first time. Loading depends on the bandwidth of the internet connection and the size of the dataset. It typically takes less than 30 seconds for the four datasets described in the application scenarios. Drava currently does not provide direct support for visualizing and interacting with more than several thousand items. In the future, we plan to further improve its scalability via item sampling and dynamically adjusting the level of detail.

9.5 Limitations of Evaluation

We evaluated Drava on four application scenarios with five domain users. The evaluation demonstrated Drava's capability on different types of datasets, domains, and analysis scenarios. At the same time, we admit the limitations resulting from the selection of participants and the setting of the evaluation. In particular, we only selected five participants in a non-random manner. The evaluation was based on self-reported feedback and included limited independent user exploration. While the evaluation revealed valuable insights and feedback, the generalizability of the results should be treated with caution. In the future, we plan to conduct a user study with a larger group of participants. Apart from assessing the usability of Drava, this user study will help validate the proposed workflow (section 4) and understand user behaviors in human–AI collaboration.

10 CONCLUSION

This paper introduces Drava, a visual analytics system that employs DRL to support the concept-driven exploration of small multiples. Focusing on the ambiguity and imperfection of DRL semantic dimensions, Drava proposes a set of interactive visualizations and algorithms to help users better interpret DRL semantic dimensions, align them with human concepts, and utilize them for visual exploration. The application of Drava for data exploration complements the widely used dimension-reduction-based visual exploration tools, especially for situations where 1) the analyzed items are similar and share multiple visual concepts and 2) the analysis focuses on certain visual concepts rather than the overall similarity. Our application scenarios demonstrate the usefulness of Drava on various datasets and for different analysis purposes. Finally, Drava demonstrates the possibilities of employing XAI techniques to help users better understand data and support visual data exploration across a wide range of domains.

ACKNOWLEDGMENTS

This work was supported by the National Institutes of Health (OT2OD026677, U24CA237617, UM1HG011536, R33CA263666). Q.W. is supported, in part, by Harvard Data Science Initiative Postdoctoral Research Fund.

REFERENCES

[1] Hervé Abdi and Lynne J Williams. 2010. Principal component analysis. *Wiley interdisciplinary reviews: computational statistics* 2, 4 (2010), 433–459.

[2] Saleema Amershi, Maya Cakmak, William Bradley Knox, and Todd Kulesza. 2014. Power to the people: The role of humans in interactive machine learning. *AI Magazine* 35, 4 (2014), 105–120.

[3] Lin An, Tao Yang, Jiahao Yang, Johannes Nuebler, Guanjue Xiang, Ross C Hardison, Qunhua Li, and Yu Zhang. 2019. OnTAD: hierarchical domain structure reveals the divergence of activity among TADs and boundaries. *Genome Biology* 20, 1 (2019), 1–16.

[4] Benjamin Bach, Nathalie Henry-Riche, Tim Dwyer, Tara Madhyastha, J-D Fekete, and Thomas Grabowski. 2015. Small MultiPiles: Piling time to explore temporal patterns in dynamic networks. In *Computer Graphics Forum*, Vol. 34. Wiley Online Library, NJ, USA, 31–40.

[5] Michael Behrisch, Fatih Korkmaz, Lin Shao, and Tobias Schreck. 2014. Feedback-driven interactive exploration of large multidimensional data supported by visual classifier. In *2014 IEEE Conference on Visual Analytics Science and Technology (VAST).* IEEE, NY, USA, 43–52.

[6] Michael Behrisch, Robert Krueger, Fritz Lekschas, Tobias Schreck, Nils Gehlenborg, and Hanspeter Pfister. 2018. Visual Pattern-Driven Exploration of Big Data. In *International Symposium on Big Data Visual and Immersive Analytics (BDVA 18).* IEEE, NY, USA, 1–11.

[7] Angie Boggust, Brandon Carter, and Arvind Satyanarayan. 2022. Embedding comparator: Visualizing differences in global structure and local neighborhoods via small multiples. In *27th International Conference on Intelligent User Interfaces.* ACM, NY, USA, 746–766.

[8] Christopher P Burgess, Irina Higgins, Arka Pal, Loic Matthey, Nick Watters, Guillaume Desjardins, and Alexander Lerchner. 2017. Understanding disentangling in $beta$-VAE. In *International Conference on Machine Learning (ICLR).* International Machine Learning Society, Sydney, Australia, 10 pages.

[9] Tal Burt, KS Button, HHZ Thom, RJ Noveck, and Marcus R Munafò. 2017. The Burden of the "False-Negatives" in Clinical Development: Analyses of Current and Alternative Scenarios and Corrective Measures. *Clinical and Translational Science* 10, 6 (2017), 470–479.

[10] Carrie J Cai, Emily Reif, Narayan Hegde, Jason Hipp, Been Kim, Daniel Smilkov, Martin Wattenberg, Fernanda Viegas, Greg S Corrado, and Martin C Stumpe. 2019. Human-centered tools for coping with imperfect algorithms during medical decision-making. In *Proceedings of the 2019 CHI conference on Human Factors in Computing Systems.* ACM, NY, USA, 1–14.

[11] Ricky TQ Chen, Xuechen Li, Roger B Grosse, and David K Duvenaud. 2018. Isolating sources of disentanglement in variational autoencoders. *Advances in Neural Information Processing Systems* 31 (2018), 11 pages.

[12] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. 2016. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. *Advances in Neural Information Processing Systems* 29 (2016), 10 pages.

[13] Furui Cheng, Mark S Keller, Huamin Qu, Nils Gehlenborg, and Qianwen Wang. 2023. Polyphony: an Interactive Transfer Learning Framework for Single-Cell Data Analysis. *IEEE Transactions on Visualization and Computer Graphics* 29, 1 (2023), 591–601.

[14] Zach Cutler, Kiran Gadhave, and Alexander Lex. 2020. Trrack: A library for provenance-tracking in web-based visualizations. In *2020 IEEE Visualization Conference (VIS).* IEEE, NY, USA, 116–120.

[15] Frederik L Dennig, Tom Polk, Zudi Lin, Tobias Schreck, Hanspeter Pfister, and Michael Behrisch. 2019. FDive: Learning relevance models using pattern-based similarity measures. In *2019 IEEE Conference on Visual Analytics Science and Technology (VAST).* IEEE, NY, USA, 69–80.

[16] K Eckelt, A Hinterreiter, P Adelberger, C Walchshofer, V Dhanoa, C Humer, M Heckmann, C Steinparz, and M Streit. 2022. Visual Exploration of Relationships and Structure in Low-Dimensional Embeddings.. In *OSF Preprint*, Vol. 10.31219/osf.io/ujbrs. Open Society Foundation, SA, 15 pages.

[17] Facebook. 2014. React.js. https://github.com/facebook/react.

[18] Liang Gou, Lincan Zou, Nanxiang Li, Michael Hofmann, Arvind Kumar Shekar, Axel Wendt, and Liu Ren. 2020. VATLD: a visual analytics system to assess, understand and improve traffic light detection. *IEEE Transactions on Visualization and Computer Graphics* 27, 2 (2020), 261–271.

[19] Miguel Grinberg. 2018. *Flask web development: developing web applications with python.* O'Reilly Media, Inc., USA.

[20] Wenbin He, Lincan Zou, Arvind Kumar Shekar, Liang Gou, and Liu Ren. 2021. Where Can We Help? A Visual Analytics Approach to Diagnosing and Improving Semantic Segmentation of Movable Objects. *IEEE Transactions on Visualization and Computer Graphics* 28, 1 (2021), 1040–1050.

[21] Irina Higgins, David Amos, David Pfau, Sebastien Racaniere, Loic Matthey, Danilo Rezende, and Alexander Lerchner. 2018. Towards a definition of disentangled representations. In *arXiv preprint*, Vol. arXiv:1812.02230. Cornell University, USA, 25 pages.

[22] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. 2016. beta-vae: Learning basic visual concepts with a constrained variational framework. In *International Conference on Machine Learning (ICLR).* International Machine Learning Society, NY, USA, 12 pages.

[23] Shichao Jia, Zeyu Li, Nuo Chen, and Jiawan Zhang. 2021. Towards visual explainable active learning for zero-shot classification. *IEEE Transactions on Visualization and Computer Graphics* 28, 1 (2021), 791–801.

[24] Zhihua Jin, Yong Wang, Qianwen Wang, Yao Ming, Tengfei Ma, and Huamin Qu. 2020. GNNLens: A Visual Analytics Approach for Prediction Error Diagnosis

of Graph Neural Networks. In *arXiv preprint*, Vol. arXiv:2011.11048. Cornell University, USA, 17 pages.

[25] Minsuk Kahng, Pierre Y Andrews, Aditya Kalro, and Duen Horng Chau. 2017. Activis: Visual exploration of industry-scale deep neural network models. *IEEE Transactions on Visualization and Computer Graphics* 24, 1 (2017), 88–97.

[26] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, et al. 2018. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *International conference on machine learning*. PMLR, Stockholm, Sweden, 2668–2677.

[27] Hannah Kim, Jaegul Choo, Haesun Park, and Alex Endert. 2015. Interaxis: Steering scatterplot axes via observation-level interaction. *IEEE transactions on visualization and computer graphics* 22, 1 (2015), 131–140.

[28] Hyunjik Kim and Andriy Mnih. 2018. Disentangling by factorising. In *International Conference on Machine Learning*. PMLR, Stockholm, Sweden, 2649–2658.

[29] Bum Chul Kwon, Hannah Kim, Emily Wall, Jaegul Choo, Haesun Park, and Alex Endert. 2016. Axisketcher: Interactive nonlinear axis mapping of visualizations through user drawings. *IEEE transactions on visualization and computer graphics* 23, 1 (2016), 221–230.

[30] Fritz Lekschas, Benjamin Bach, Peter Kerpedjiev, Nils Gehlenborg, and Hanspeter Pfister. 2017. HiPiler: visual exploration of large genome interaction matrices with interactive small multiples. *IEEE Transactions on Visualization and Computer Graphics* 24, 1 (2017), 522–531.

[31] Fritz Lekschas, Michael Behrisch, Benjamin Bach, Peter Kerpedjiev, Nils Gehlenborg, and Hanspeter Pfister. 2020. Pattern-Driven Navigation in 2D Multiscale Visualizations with Scalable Insets. *IEEE Transactions on Visualization and Computer Graphics* 26, 1 (1 1 2020), 611–621.

[32] Fritz Lekschas, Brant Peterson, Daniel Haehn, Eric Ma, Nils Gehlenborg, and Hanspeter Pfister. 2020. Peax: Interactive visual pattern search in sequential data using unsupervised deep representation learning. In *Computer Graphics Forum*, Vol. 39–3. Wiley Online Library, USA, 167–179.

[33] Fritz Lekschas, Xinyi Zhou, Wei Chen, Nils Gehlenborg, Benjamin Bach, and Hanspeter Pfister. 2021. A Generic Framework and Library for Exploration of Small Multiples through Interactive Piling. *IEEE Transactions on Visualization and Computer Graphics* 27, 2 (1 2 2021), 358–368.

[34] Mengchen Liu, Shixia Liu, Hang Su, Kelei Cao, and Jun Zhu. 2018. Analyzing the noise robustness of deep neural networks. In *2018 IEEE Conference on Visual Analytics Science and Technology (VAST)*. IEEE, NY, USA, 60–71.

[35] Shusen Liu, Peer-Timo Bremer, Jayaraman J Thiagarajan, Vivek Srikumar, Bei Wang, Yarden Livnat, and Valerio Pascucci. 2017. Visual exploration of semantic relationships in neural word embeddings. *IEEE Transactions on Visualization and Computer Graphics* 24, 1 (2017), 553–562.

[36] Xiao Liu, Pedro Sanchez, Spyridon Thermos, Alison Q O'Neil, and Sotirios A Tsaftaris. 2022. Learning disentangled representations in the imaging domain. *Medical Image Analysis* 80 (2022), 102516.

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

[38] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. 2015. Deep Learning Face Attributes in the Wild. In *Proceedings of International Conference on Computer Vision (ICCV)*. IEEE, USA, 3730–3738.

[39] Sehi L'Yi, Qianwen Wang, Fritz Lekschas, and Nils Gehlenborg. 2021. Gosling: A Grammar-based Toolkit for Scalable and Interactive Genomics Data Visualization. *IEEE Transactions on Visualization and Computer Graphics* 28, 1 (1 10 2021), 140–150.

[40] Loic Matthey, Irina Higgins, Demis Hassabis, and Alexander Lerchner. 2017. dSprites: Disentanglement testing Sprites dataset. https://github.com/deepmind/dsprites-dataset/.

[41] Leland McInnes, John Healy, Nathaniel Saul, and Lukas Großberger. 2018. UMAP: Uniform Manifold Approximation and Projection. *Journal of Open Source Software* 3, 29 (2018), 861.

[42] Yao Ming, Shaozu Cao, Ruixiang Zhang, Zhen Li, Yuanzhe Chen, Yangqiu Song, and Huamin Qu. 2017. Understanding hidden memories of recurrent neural networks. In *2017 IEEE Conference on Visual Analytics Science and Technology (VAST)*. IEEE, NY, USA, 13–24.

[43] Paul Mooney. 2017. Breast Histopathology Images. https://www.kaggle.com/datasets/paultimothymooney/breast-histopathology-images.

[44] Giulio Mori, Fabio Paternò, and Carmen Santoro. 2002. CTTE: support for developing and analyzing task models for interactive system design. *IEEE Transactions on software engineering* 28, 8 (2002), 797–813.

[45] 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. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.). Curran Associates, Inc., NY, USA, 8024–8035.

[46] Nicola Pezzotti, Thomas Höllt, Jan Van Gemert, Boudewijn PF Lelieveldt, Elmar Eisemann, and Anna Vilanova. 2017. Deepeyes: Progressive visual analytics for designing deep neural networks. *IEEE Transactions on Visualization and Computer Graphics* 24, 1 (2017), 98–108.

[47] Meg Pirrung, Nathan Hilliard, Artëm Yankov, Nancy O'Brien, Paul Weidert, Courtney D Corley, and Nathan O Hodas. 2018. Sharkzor: Interactive deep learning for image triage, sort and summary. In *arXiv preprint*, Vol. arXiv:1802.05316. Cornell University, USA, 4 pages.

[48] Suhas SP Rao, Miriam H Huntley, Neva C Durand, Elena K Stamenova, Ivan D Bochkov, James T Robinson, Adrian L Sanborn, Ido Machol, Arina D Omer, Eric S Lander, et al. 2014. A 3D map of the human genome at kilobase resolution reveals principles of chromatin looping. *Cell* 159, 7 (2014), 1665–1680.

[49] Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Brij B Gupta, Xiaojiang Chen, and Xin Wang. 2021. A survey of deep active learning. *Comput. Surveys* 54, 9 (2021), 1–40.

[50] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. 2015. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)* 115, 3 (2015), 211–252. https://doi.org/10.1007/s11263-015-0816-y

[51] Daniel Smilkov, Nikhil Thorat, Charles Nicholson, Emily Reif, Fernanda B Viégas, and Martin Wattenberg. 2016. Embedding projector: Interactive visualization and interpretation of embeddings. In *arXiv preprint*, Vol. arXiv:1611.05469. Cornell University, USA, 4 pages.

[52] Karthick Sothivelr. 2020. Breast Cancer Classification With PyTorch and Deep Learning. https://medium.com/swlh/.

[53] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. 2014. Striving for simplicity: The all convolutional net. In *ICLR (workshop track)*. International Machine Learning Society, USA, 14 pages.

[54] Hendrik Strobelt, Sebastian Gehrmann, Hanspeter Pfister, and Alexander M Rush. 2017. Lstmvis: A tool for visual analysis of hidden state dynamics in recurrent neural networks. *IEEE Transactions on Visualization and Computer Graphics* 24, 1 (2017), 667–676.

[55] Alistair Sutcliffe. 2000. On the effective use and reuse of HCI knowledge. *ACM Transactions on Computer-Human Interaction (TOCHI)* 7, 2 (2000), 197–221.

[56] Alistair G Sutcliffe and John M Carroll. 1999. Designing claims for reuse in interactive systems design. *International Journal of Human-Computer Studies* 50, 3 (1999), 213–241.

[57] Edward R Tufte, Nora Hillman Goeler, and Richard Benson. 1990. *Envisioning information*. Vol. 126. Graphics press, Cheshire, CT.

[58] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research* 9, 11 (2008), 2579–2605.

[59] Junpeng Wang, Wei Zhang, and Hao Yang. 2020. SCANViz: Interpreting the symbol-concept association captured by deep neural networks through visual analytics. In *2020 IEEE Pacific Visualization Symposium (PacificVis)*. IEEE, NY, USA, 51–60.

[60] Qianwen Wang, William Alexander, Jack Pegg, Huamin Qu, and Min Chen. 2020. HypoML: Visual analysis for hypothesis-based evaluation of machine learning models. *IEEE Transactions on Visualization and Computer Graphics* 27, 2 (2020), 1417–1426.

[61] Qianwen Wang, Zhutian Chen, Yong Wang, and Huamin Qu. 2021. A Survey on ML4VIS: Applying MachineLearning Advances to Data Visualization. *IEEE Transactions on Visualization and Computer Graphics* 28, 12 (2021), 5134–5153.

[62] Qianwen Wang, Zhen Li, Siwei Fu, Weiwei Cui, and Huamin Qu. 2019. Narvis: Authoring Narrative Slideshows for Introducing Data Visualization Designs. *IEEE Transactions on Visualization and Computer Graphics* 25, 1 (jan 2019), 779–788.

[63] Qianwen Wang, Tali Mazor, Theresa A Harbig, Ethan Cerami, and Nils Gehlenborg. 2021. ThreadStates: State-based Visual Analysis of Disease Progression. *IEEE Transactions on Visualization and Computer Graphics* 28, 1 (2021), 238–247.

[64] Qianwen Wang, Yao Ming, Zhihua Jin, Qiaomu Shen, Dongyu Liu, Micah J Smith, Kalyan Veeramachaneni, and Huamin Qu. 2019. Atmseer: Increasing transparency and controllability in automated machine learning. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. ACM, NY, USA, 1–12.

[65] Qianwen Wang, Zhenhua Xu, Zhutian Chen, Yong Wang, Shixia Liu, and Huamin Qu. 2020. Visual analysis of discrimination in machine learning. *IEEE Transactions on Visualization and Computer Graphics* 27, 2 (2020), 1470–1480.

[66] Yinqiao Wang, Lu Chen, Jaemin Jo, and Yunhai Wang. 2021. Joint t-SNE for Comparable Projections of Multiple High-Dimensional Datasets. *IEEE Transactions on Visualization and Computer Graphics* 28, 1 (2021), 623–632.

[67] Jun Yuan, Changjian Chen, Weikai Yang, Mengchen Liu, Jiazhi Xia, and Shixia Liu. 2021. A survey of visual analytics techniques for machine learning. *Computational Visual Media* 7, 1 (2021), 3–36.

[68] Zhenge Zhao, Panpan Xu, Carlos Scheidegger, and Liu Ren. 2021. Human-in-the-loop Extraction of Interpretable Concepts in Deep Learning Models. *IEEE Transactions on Visualization and Computer Graphics* 28, 1 (2021), 780–790.