IEEE VIS Full Paper. To appear in IEEE Transactions on Visualization and Computer Graphics.
DOI: 10.1109/TVCG.2022.3209384. Details: https://bckwon.com/publication/conceptexplainer

ConceptExplainer: Interactive Explanation for Deep Neural Networks from a Concept Perspective

Jinbin Huang, Aditi Mishra, Bum Chul Kwon, *Member, IEEE*, Chris Bryan, *Member, IEEE*

Fig. 1. CONCEPTEXPLAINER provides interactive concept-based explanations on deep neural networks (DNNs) at global/class/instance levels via a coordinated visual analytics interface connected to a backend processing pipeline. A full description of the interface design is given in Sect. 4.4; this image also shows actions taken during the usage scenario described in Sect. 5.2.

Abstract—Traditional deep learning interpretability methods which are suitable for model users cannot explain network behaviors at the global level and are inflexible at providing fine-grained explanations. As a solution, concept-based explanations are gaining attention due to their human intuitiveness and their flexibility to describe both global and local model behaviors. Concepts are groups of similarly meaningful pixels that express a notion, embedded within the network's latent space and have commonly been hand-generated, but have recently been discovered by automated approaches. Unfortunately, the magnitude and diversity of discovered concepts makes it difficult to navigate and make sense of the concept space. Visual analytics can serve a valuable role in bridging these gaps by enabling structured navigation and exploration of the concept space to provide concept-based insights of model behavior to users. To this end, we design, develop, and validate CONCEPTEXPLAINER, a visual analytics system that enables people to interactively probe and explore the concept space to explain model behavior at the instance/class/global level. The system was developed via iterative prototyping to address a number of design challenges that model users face in interpreting the behavior of deep learning models. Via a rigorous user study, we validate how CONCEPTEXPLAINER supports these challenges. Likewise, we conduct a series of usage scenarios to demonstrate how the system supports the interactive analysis of model behavior across a variety of tasks and explanation granularities, such as identifying concepts that are important to classification, identifying bias in training data, and understanding how concepts can be shared across diverse and seemingly dissimilar classes.

Index Terms—Explainable AI, Concept Activation Vectors, Interactive Visual Analytics

1 INTRODUCTION

Deep learning has led to tremendous advances across a number of fields, including speech recognition, medical applications, computer vision, and autonomous vehicles [8]. Unfortunately, the complexity and inherent opaqueness of neural networks imposes significant difficulty in understanding the behavior and inner workings of models [6]. As deep learning is increasingly employed in today's society, it is important that diverse groups of people (e.g. model users "who may have some technical background but lack expertise in neural network development" [19] in AI) are able to understand and interpret the predictions made by AI models [49]—the explainable AI (XAI) subfield supports the development of tools and techniques to support this process [50].

Generally, interpretability of trained neural networks is achieved either by (1) revealing the neural architecture, and how signals flow through the pipeline, or (2) using explainable substitutions to approximate the neural network's mental model on a task [16]. The first option

- *Jinbin Huang, Aditi Mishra and Chris Bryan are with Arizona State University. E-mail: {jhuan196, amishr45, cbryan16}@asu.edu*
- *Bum Chul Kwon is with IBM research. E-mail: bumchul.kwon@us.ibm.com.*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

is straightforward (i.e., making transparent the inner logic and algorithmic functions of the model), but requires users be equipped with deep learning expertise, and is thus inappropriate for model users [28]. In contrast, the latter option must be carefully employed as it comes with inherent information loss and potentially false reflection [9].

One interpretability approach that is suitable for model users is to demonstrate how input perturbation affects model outputs. In image classification, saliency and activation maps [12, 38, 40, 41, 55] measure the importance of the input features (i.e., image pixels) when classifying an image. The drawback is that these approaches only provide *local* or sample-level explanations for individual images. Cross-grained insights which can help users understand *class* or *global*-level model behaviors—e.g., measuring the influence that *ear* and *fur* have on predicting a classification of a *cat*—is not possible [24]. However, such understanding can be important for model users, as it allows them to compare their own mental models of how classification should work to the neural network's mental model [23].

To facilitate such understanding, concept-based methods have recently been introduced by the AI community [15,24,45,51]. For image classification, a concept refers to a group of pixels in a sample that represents an important part of the class object within the image [15]. To be considered as a concept for a class, the concept needs to be easily understandable to human users, consistent within each concept but separable from other concepts, and necessary for the prediction of the class. Importantly, concepts can be used to generate explanations at the different levels of granularity. At a high level, we can compute the concepts that are influential to classes (e.g., *ear*, *fur*, *tail* are likely important to the *cat* class). Locally (for an instance in a class), we can measure how much each concept influences the correct or incorrect classification of the class. When concepts and classes are aggregated together, there is the potential to form a global perspective of the neural network's mental model, to understand the large-scale behavior based on what the network has learned.

Although concepts are appropriate for use as an interpretability technique for model users [24], there has been little broad application of this technique. This is likely because of two shared complexities that make concept generation and presentation a difficult exercise: (1) The common procedure to create concepts is by hand (i.e., manually). To automatically generate the massive concept space that covers an entire neural network is a non-trivial exercise. (2) Structuring, navigating, and probing such a concept space in a way that supports different (global and local) explanation granularities is likewise non-trivial.

To address these two challenges, we propose CONCEPTEXPLAINER, consisting of two primary components: (1) a backend pipeline interactively (and automatically) generates and structures a concept space for a neural network, and (2) a coordinated, visualization-driven frontend lets users interactively explore and probe the concept space. **To the best of our knowledge, CONCEPTEXPLAINER is one of the first interactive visual analytics systems** to support model users employing concept-based explanations to understand deep learning models. At the global level, CONCEPTEXPLAINER can reveal which concepts are broadly influencing the neural network's decision making. At the class level, users can visualize which concepts are influential for that class, and can explore the overlaps of influential concepts which are shared between classes. At the instance level, individual images can be reviewed to see how present/non-present concepts affect the model's correct/incorrect predictions. In developing and evaluating CONCEPTEXPLAINER, we make the following research contributions:

(1) Identifying challenges and goals. Based on an analysis of challenges in XAI for model users, we identify salient design challenges and goals which are important for concept-based explanation, particularly in an interactive visualization context.

(2) A novel visual analytics tool for concept-based explanation. To support the ideated challenges and goals, we design and implement CONCEPTEXPLAINER, an interactive system that automatically generates a concept space for an image classification neural network, and supports model users to explore and probe concept-based explanations at multiple levels of granularity. CONCEPTEXPLAINER is designed based on an iterative prototyping process and validated through multiple evaluations (usage scenarios and a user study).

(3) Empirical findings and generalizable takeaways. Based on the process of designing, implementing, and validating CONCEPTEXPLAINER, we discuss several lessons learned and takeaways about how visualization-driven concept-based explanation can support XAI tasks for model users, such as identifying issues (or biases) in data samples, and how tools like CONCEPTEXPLAINER can be extended in the future (e.g., supporting expert users).

2 RELATED WORK

2.1 Deep Learning Interpretability using Concepts

There are two primary approaches for using concepts to improve deep learning interpretability: (1) training inherently interpretable models with concept-based constraints. (2) constructing post-hoc explanations based on concepts.

Regarding the first approach, Koh et al. [25] proposed Concept Bottleneck Model (CBM), which restricts neural networks to behave in an auto-encoder manner where they first map inputs to human-interpretable concepts and then predict based on those concepts. Chen et al. [5] proposed Concept Whitening (CW) layer as a substitution for the normalization layer (i.e., batch normalization) to achieve higher interpretablity. While these methods employ concepts to offer embedded interpretablity with no extra dependency, they do not provide ways to gain insights into a trained model. Our work differs from this line of research as we seek to interpret fixed models with no alteration.

For the second approach, TCAV [24] provides a means to computationally define a concept and calculate its class-specific influence on model predictions (called the TCAV score). Building upon TCAV, ACE [15] is a technique to automatically extract influential class-specific concepts from a dataset. For a formal description of ACE and TCAV, see Sect. 4.3, where we describe how both techniques are incorporated into CONCEPTEXPLAINER's backend to automatically define and extract concepts.

Similarly, Ge at al. [14] proposed the Visual Reasoning Explanation (VRX) framework that answers interpretability questions such as *Why?* and *Why not?* [28, 29] from a concept perspective by extracting and organizing class-specific concepts using trained structural concept graphs (SCGs) based on pairwise concept relationships. Though VRX provides insightful instance analysis, it requires training an SCG for each class. This makes VRX difficult to implement within interactive scenarios (i.e., as a part of a user interface) due to computational cost.

2.2 Visual Analytics in Concept-based Interpretability

A large amount of neural network interpretability research in the visualization community has utilized (non-concept-based) conventional methods, including visualizing features, saliency and activation maps, and visualizing the model neurons or architectures (e.g., [3, 34, 42]); for more information, there are several recent surveys that discuss XAI from a visualization perspective [19, 52, 53]. For research that employs visualization to support concept-based tasks, there are two recent systems that are highly relevant to CONCEPTEXPLAINER:

CONCEPTEXTRACT [54] utilizes visualization within a human-in-the-loop pipeline for concept extraction and fine-tuning. The system generates initial concept segments using ACE; the user then interactively refines the set of concept images. The user can also participate in refining neural network-based binary concept classifiers by interactively supplying labels. The intent of the system is to counter the potential drawbacks of automatic concept extraction, such as human incomprehensible patches, by incorporating human oversight and manual labeling, with the ultimate goal of enabling efficient handcrafting of high-quality concepts.

NEUROCARTOGRAPHY [35] aims to interactively reveal neuron-concept relations by grouping a model's similarly-activated neurons in the same latent layer by the same set of data instances. The corresponding set of data instances is viewed as a set of concepts. Concepts discovered in this way are layer-dependent. They evolve as layers go deeper. The system is thus capable of letting users analyze concept evolution in the context of the neural architecture, which is potentially beneficial to expert users (e.g., model developers) in debugging.

While these two systems combine concept-based explanations and visual analytics, both differ from CONCEPTEXPLAINER in two important ways: (1) They are designed for expert users; in contrast, we focus on AI model users as defined by Hohman et al.'s survey paper [19]. (2) They support different tasks (creating/refining concepts, and revealing concept/neuron relations); our system is designed to probe and explore concepts as a way to understand model behavior.

2.3 Deep Learning Interpretability for Model Users

The popularity of deep learning in today's society has led to increased demands for XAI techniques that are friendly to model users who have zero or few deep learning background [30]. Desiderata for this type of explanation includes faithfulness, consistency with prior beliefs and generality [4]. Liao et al. [28] proposed a set of design principles for end users, targeting explanation system in form of a question bank, where global explanations for a model's decision making and local explanations for a particular decision on an instance were ranked as top interests. Similarly, Hohman et al. [19], in their categorization of end users (which include model developers/builders, model users, and non-experts) suggest that model users who apply deep learning models to their domain tasks mostly need tools that support exploration of model behaviors at both local and global scale. Concept-based explanations fits these needs well, as they require minimal prior knowledge about neural networks (e.g., it is not necessary to know technical concepts like backpropagation) while remaining intuitive and accessible. CONCEPTEXPLAINER supports probing and exploring of concepts at multiple levels to support interpretability tasks for model users.

3 DESIGN CHALLENGES AND GOALS

Similar to prior visualization design studies in XAI [19, 20, 39], we identify a salient set of design challenges (C1–C4) which are important for concept-based explanation for model users, based on reviewing of state-of-the-art publications that discuss challenges in XAI, concept-based explanations, and AI explanations for model users (see Section 2). We distill these into a set of four design goals (G1–G4), which we use to guide CONCEPTEXPLAINER's development.

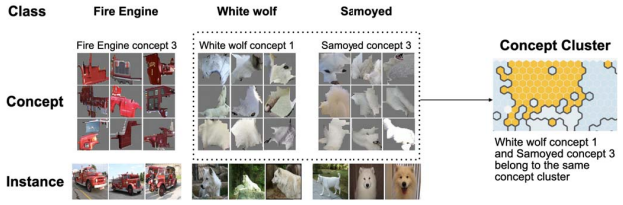

Fig. 2. An example of how classes, concepts, and instances are related. Using our concept extraction and clustering pipeline (described in Sect. 4.3, we show an example concept for each of the *fire engine*, *white wolf*, and *Samoyed* classes.

3.1 Design Challenges

(C1) Extracting human-understandable concepts for classes. There is increasing evidence that neural networks predict based on a combination of the concepts present in instances [10, 18, 22, 33, 48]. It is helpful for the user to understand neural network behavior by demonstrating what concepts a neural network relies on for predictions and the extent of concept influence on predictions. For example, when the neural network classifies a *tiger*, is the *bushy background* concept more influential or the *tiger stripe*? By extracting influential concepts for each class, the user can intuitively understand how the network understands each class and verify if it works in a sensible way. However, discovering concepts and measuring concept influences can be difficult as noise is inevitable and there are different ways to measure influence [54]. In our work, we adopt the ACE method for concept extraction and TCAV method for measuring concept influence, as they are the most widespread approaches for concept-based explanations. We also propose a concept clustering process after concept discovery to structure the concept space and facilitate concept navigation.

(C2) Multi-scale concept visualization for large datasets. System scalability has gained increasing attention in the visualization community. The issue becomes salient when visual interfaces for XAI are concerned because models to be explained are trained on large datasets. Concept explanations also face this issue because the number of concepts discovered grows linearly with the number of classes in the dataset. To demonstrate the concept space and enable users to gain insights from navigating the space, we need more than a naive navigation mechanism which can cause the user to be lost in the deluge of concept information without a clear navigational goal. Apart from that, the data structure of concepts – quantitative influences and images – requires specifically designed visualizations to present.

(C3) Revealing conceptual overlap between classes. The conceptual overlap (i.e., common influential features) between classes can reveal to the user why one class might be easily misclassified as another (e.g., a *Samoyed* classified as a *wolf* due to the influence of a *snowy background* concept). Among a group of classes, understanding what concepts are shared (or not shared) can help discover features that cause confusion [17] and/or serve as the unique "signature" of a class in the neural network's mental model. For example, Fig. 2, shows a network has learned a *white fur* concept from both the *white wolf* and *Samoyed* classes, which provides an idea about the commonalities the network sees between these classes. In contrast, the lack of overlap between *fire engine* and *white wolf* also suggests that the network discriminates animals from vehicles.

(C4) Balancing global explanations and local explanations. Global explanations without much detail are generally easier to understand. However, since global explanations are a summary of local behavior solely revealing global explanations might lead to unfaithful interpretations. To ensure faithfulness of the explanations we need to show information at both global and local levels and integrate the analysis process at both levels in an organic workflow [36]. The user benefits from having a holistic view of neural network behavior at various levels (see Fig. 2) in a consistent concept-based framework so they can (1) recognize contradictory explanations (2) verify their hypothesis at different levels.

3.2 Design Goals

Based on the design challenges C1–C4, we identify four design goals for the CONCEPTEXPLAINER system to support interactive concept-based explanations of neural networks' behavior for model users. Roughly, these goals can be ordered in terms of their granularity: at the global, class, and instance levels of analysis and explanation.

(G1) Navigating the global concept space. We aim to facilitate navigation in the concept space of a large dataset (in our case, we use ImageNet, which contains 1.2 M images for 1000 classes, see Sect. 4.2) (C2). The methodology we use is transferable to other image datasets as long as a trained deep learning model is available. To discover the concept space while preventing noise (C1), we first use ACE algorithm to extract influential concepts from the dataset. To facilitate structured concept navigation we cluster the extracted concepts into multiple concept clusters. The clustering of concepts makes the navigation easier—the user can start their navigation by peeking at various concept clusters to pick up the one they want to explore and get into the cluster to see concepts inside. By gradually understanding the concept clusters the user can understand the concept space. The transition from concept cluster to individual concept reinforces an overview-first-detail-on-demand workflow. To help the user keep track of their investigation process in the concept space and understand the concept clusters by labeling them (taking notes of them) in a user-driven way, annotation functionality should also be available in the system.

(G2) Supporting intra-class concept analysis For a class, the discovered, influential concepts can be represented as a collection of image patches that have similar high dimensional activations. These influential concepts intuitively reveal how the model understands a class. The system needs to present these concepts such that the user can understand what they are and the extent of their influence (C2,

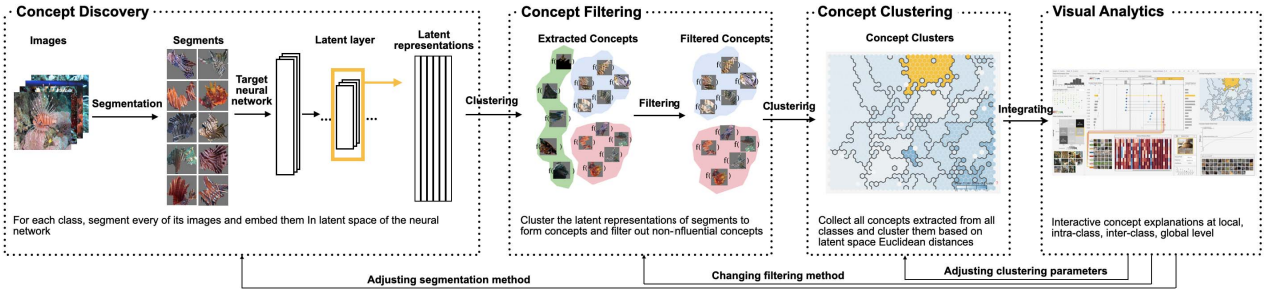

Fig. 3. The workflow for CONCEPTEXPLAINER. In the *concept discovery* stage, class-specific images are segmented, embedded in latent space and clustered to form class-specific concepts. In the *concept filtering* stage, non-influential concepts are filtered based on t-test on their TCAV scores. Concepts are then clustered in the *concept clustering* stage. We employ *visual analytics* in the frontend interface to let users explore and probe concepts at the local, class, and global levels; users can also interactively update parameters on the backend stages. By quickly scanning through concept patches in the visual interface, the user can determine if those patches are comprehensible enough and update the backend interactively (e.g. changing image segmentation methods from 'QuickShift' to 'Slic' or increasing filtering thresholds).

C3). Users can even inspect the most influential concepts to see if the network has learned sensible relations (e.g., noisy concepts should be less influential than relevant concepts).

(G3) Supporting inter-class concept analysis. Understanding conceptual commonalities between classes can help users to understand how the network perceives similar/different classes and pinpoint concept-based root causes for misclassifications. To this end, we seek to demonstrate the links between similar concepts of different classes (C3). It is necessary that inter-class concept analysis go beyond "comparing two classes," as several classes may share commonalities of interest and the user may want to understand how the neural network considers them comprehensively.

(G4) Supporting instance analysis. Only explaining neural network behavior from a global or class level might lead the user to overlook important details, resulting in misinterpretations of how the network should work (C4). To enable detailed inspection, while not overwhelming by revealing too many details, concept influences can be measured for individual instances. Such options keep the analytical framework unified: the user does not need other tools for instance explanations/analysis. In addition, instance-level analysis also helps users investigate edge cases ("*Why did the model misclassify this image?*") by listing influential concepts of the image.

4 SYSTEM DESIGN

Based on the design goals G1–G4, we develop CONCEPTEXPLAINER, an interactive system that provides model users with the ability to probe and analyze concept-based explanations at both the global and local levels. As shown in Fig. 3, the system consists of an integrated backend and frontend. In the **backend**, we design a processing pipeline that leverages concept-extraction methods (specifically, TCAV and ACE) to automatically extract and organize concepts for a given image dataset. Extracted concepts are organized into meaningful clusters to facilitate interactive concept navigation and concept overlap inspection. The **frontend** interface (Fig. 1) consists of four coordinated panels: (1) the header (M) contains controls widgets, including for manipulating the backend processing (setting parameters for the TCAV, ACE, and clustering methods), (2) the left panel (A – C) provides overview, navigation and analysis for classes, (3) the right panel (D – F) provides a clutter-free locality-retaining overview for the concept space and navigation mechanisms reinforcing the overview-first-detail-on-demand mantra. (4) the central panel (G – L) provides support for class/instance level analysis – an overview of class-specific concepts and inter-class concept links combined with a detailed concept inspection view account for both class and instance level model behaviors.

4.1 Iterative Prototyping Process

To facilitate our design process, we employed an iterative prototyping methodology [32]. Over an approximately six months period, we sketched and prototyped a number of user interface and interaction designs. Designs were holistically reviewed and discussed by the project team, with additional feedback solicited from XAI researchers who work on interpretability techniques for model users. Designs deemed suboptimal were either discarded or refactored (Fig. **??** in the Appendix shows two examples of "old" interfaces), while well-received designs were iteratively refined to develop the current system version. As an example of this process' impact, we adopted a three-panel layout in the interface (as shown in Fig. 1 and described in Sect. 4.4). The left **class navigation panel** supports class-based analysis and navigation; in parallel, the right **concept-navigation panel** supports concept-based analysis and navigation. Within each of these panels, users can transit fluently from global perspectives of the model (i.e., visualizing all classes/concepts together) down to analyzing individual instances. Classes, concepts, and instances are synthesized together in the center **class-concept panel**.

4.2 Dataset and Model

To demonstrate a real-world application throughout the rest of this paper, we use the ImageNet dataset [7], which is a well-known image dataset consisting of 1.2M training instances. As a model, we employ GoogLeNet [44], a convolutional neural network consisting of 22 layers. GoogLeNet provides a pre-trained model for ImageNet, which classifies images into 1,000 object categories. Thus, our task is to use CONCEPTEXPLAINER to explain GoogLeNet's behavior in the context of instances and classes for the ImageNet dataset.

4.3 Backend: Concept Extraction and Clustering

We use a combination of TCAV [24] and ACE [15] as the methodological backbone for concept-based explanation. Combining these methods together lets us automatically derive influential concepts for each class predicted by GoogLeNet and the extent of the class' influences. By subsequently clustering the ensemble of derived concepts, we impose a human-understandable structure on the concepts that approximates the model's mental model in classification.

4.3.1 TCAV

TCAV [24], or *testing with concept activation vectors*, is a concept-based explanation method that measures the importance of human understandable concepts to the neural network's inference—e.g., how much a *stripe* concept affects a classifier when predicting a *zebra* image. The TCAV approach views concepts as a set of images of similar traits. A set of images without these traits forms a set of counterexamples (i.e,. non-concept examples).

A concept activation vector, or CAV, is defined as the normal to the hyperplane that linearly separates high-dimensional representations of concept examples from that of non-concept examples. To measure the influence of a concept to an instance's prediction, TCAV computes pixel gradients of the instance at a target layer and compares its direction with the CAV's (both are in the same latent space) by taking inner product of the two. If the gradient vector lines up with the CAV (indicated by a positive inner product) it means the concept is positively

influencing the prediction (*stripe* makes an instance more likely to be a *zebra*). Conversely, if the two vectors part (indicated by negative inner product), the concept negatively influences the prediction.

The TCAV score, which measures conceptual influence, is defined as: $TCAV_{Q_{C,k,l}} = \frac{|\{x \in X_k : S_{C,k,l}(x) > 0\}|}{|X_k|}$ where the fraction of k-class inputs whose l-layer activation vector was positively influenced by concept C. $TCAV_Q(\cdot)$ means quatitatively testing CAVs and $S_{C,k,l}(x)$ is the aforementioned innerproduct of pixel gradients and CAVs. To prevent meaningless concepts (concepts with influence $\sim$0.5), multiple CAVs can be trained using the same set of concept examples against various sets of counter examples. A meaningful concept should lead to TCAV scores that rejects the hypothesis of a 0.5 TCAV score with statistical significance (we use a threshold of $p > 0.01$).

To reduce noisy concepts, we train 20 CAVs for each concept on concept examples (this number can be changed via the frontend) and randomly form counter examples after it is generated using ACE method. TCAV scores are computed for each CAV and averaged.

4.3.2 ACE

TCAV provides a measurement for the influence of human-understandable concepts on the neural network for a specific class, but in the base TCAV approach, forming a set of examples for a concept is a manual process. To automate the process of concept generation and extraction, Ghorbani et al. [15] proposed ACE, or *automatic concept-based explanations*.

ACE extracts a set of concepts for each class the model predicts on, by segmenting the images of the class and grouping segments that form clusters in the high-dimensional space of specific latent layer; the TCAV procedure can be followed to filter out meaningless collections. This combined TCAV-plus-ACE process makes efficient and automated concept generation tractable. In our system, we default to sampling 50 images for each class, and segment them using SLIC [1] with three resolutions (15, 50, and 80 segments) per image. The segments are embedded in the network's "mixed4c" layer. K-means clustering is then performed to cluster their latent representations into 10 concepts (high-dimensional clusters), though each of these parameters is controllable from CONCEPTEXPLAINER's frontend.

4.3.3 Concept Clustering

Running ACE will extract a large ensemble of concepts. For example, we extracted 1,211 concepts from 143 randomly selected classes in ImageNet using the default parameters mentioned above. This concept space is unorganized (i.e., there's no ranking or structure imposed on them), which makes it difficult to review, explore, and compare them.

An additional consideration (or complexity) is that concepts discovered in different classes might be semantically similar (e.g., *snow background* in *husky* and *snowy background* in *wolf*); such similarity needs to be demonstrated because it is useful for understanding the model from a concept perspective.

To enable structured and guided navigation within this the concepts space (G1), we apply clustering to the extracted concepts. The intent is to group semantically similar concepts into the same cluster, which will let us leverage cluster identity as a means to demonstrate conceptual overlap across classes, and to form a meaningful navigation structure that the user can rely on to probe and explore the concept space.

The similarity between two concepts is quantified by their cluster identity (if two concepts come from the same cluster their similarity is 1, otherwise 0). In the frontend, users can choose between k-means and agglomerative clustering, and select associated cluster parameters. Based on feedback from our user study evaluation (see Sect. 6), this provides good flexibility by allowing users to explore different clustering levels and granularities, though the system is extensible to other clustering methods (i.e., as future work) and heuristics. In particular, clustering itself can benefit from the use of XAI techniques, though exploring such activities is beyond the scope of the current paper.

4.4 Frontend: CONCEPTEXPLAINER Interface

Fig. 1 shows the CONCEPTEXPLAINER interface, which consists of four primary panels and several coordinated views (A–N). The top panel provides interactions with the backend, while the bottom three panels support a workflow that includes class analysis (left panel), concept analysis (right panel) and class-concept analysis (central panel).

4.4.1 Class Navigation Panel

The leftmost **class navigation panel** provides three visualizations (A–C) to facilitate navigation and selection of classes of interest for concept exploration. This panel primarily supports exploring the model from a class-based perspective, and allows users to analyze the concept space at a global level (G1), comparatively analyze classes (G2 and G3), and inspect specific instances within a class (G4).

(A) The **class performance view** summarizes model performance. Users can begin exploration by brushing histogram bins to filter out undesired classes (e.g., show only low performance classes).

(B) Brushing updates the **class navigation view**. This view provides a global perspective of all classes using t-SNE [47], which maps a high dimensional representation of classes (computed as the average of latent vectors of all images of the same class) into a two dimensional embedding. To prevent overplotting, we grid the bounding box of the 2D t-SNE point cloud and aggregate classes into "clique" circles, sized by the number of total classes and colored based on average accuracy of classes in the clique (This design is inspired by the Chameleon system [21]). Hovering displays a tooltip (see (O)) showing individual classes belonging to the clique with with a representative image per class. Users can select a class for subsequent analysis.

(C) Selected classes are displayed in the **class confusion matrix view**. The confusion matrix supports inter-class and intra-class analysis by summarizing the classification performance against ground truth labels across the selected classes. Clicking on a cell shows a list of images that belong to the corresponding category of the cell below the confusion matrix. For example, Fig. 1(C) shows the 11 instances of *tiger cat* which were misclassified as *tiger*.

Hovering over an instance shows the concepts that affect the instance's prediction. For example, in Fig. 5(J) (part of usage scenario #3), a *police van* image was misclassified as *ambulance*. Hovering on the instance shows how influential the concepts of these two classes affect the prediction. In the usage scenario, the *ambulance* concepts have slightly higher, influence scores than *police van* concepts, which explains why the model misclassified the image as *ambulance*.

4.4.2 Concept Navigation Panel

The **concept navigation panel** on the right side of the interface contains three views that facilitate navigation of the concept space in a structured way (G1) while providing context during class-concept and instance analysis (G2–G4).

(D) The entire concept space is displayed in a hexagon plot in the **concept navigation view**. The design of this view underwent multiple iterations during the iterative prototyping process. For example, Fig.7 (top) in the Appendix shows an early visualization design of the concept space, based on dimensionality reduction where each concept was plotted as a point. This design had several limitations, including overplotting and difficulty in distinguishing and selecting individual concept clusters.

The hexagon design avoids these issues. Instead of plotting a point cloud, we use the IsoMatch method [13] to plot organize concepts within a hexagonal grid layout like a heatmap. Each concept is visualized as a hex tile, assigned to a unique location such that (a) clutter and overlap is avoided, (b) cluster boundaries are easily demarcated using dark borders, (c) concepts are evenly distributed across the panel's available space, and (d) concept clusters are easily selectable.

(E) Clicking on a cluster in (D) loads it into the **concept cluster detail view**. This chart summarizes the concepts belonging to a concept cluster. Concepts are ordered along the x-axis by their influence scores, which are mapped to the y-axis. Circles are colored using a diverging blue-to-red color scale based on positive (above 0.5) or negative (less than 0.5) influence score; the influence score mid-point (0.5) is shown as a dotted line, and the average influence score over all concepts in the cluster is shown as a solid horizontal line.

(F) Hovering or clicking on a concept in (E) loads it into the **individual concept view**, which loads all image patches for that concept. By skimming through these, the user can quickly gain a semantic understanding of what that concept is. Skimming multiple concepts in this way can help show why they belong to the same concept cluster.

As an additional feature for this view, we provide users with an annotation feature in (D), activated by right clicking a cluster, to input descriptive information about a concept in a concept cluster for future reference. The user is also enabled to save their annotation for later work or upload an existing annotation to start with established insights.

4.4.3 Class-Concept Panel

The central **class-concept panel** synthesizes information from the left (class-focused) and right (concept-focused) panels, supporting interpretation to the underlying model's behavior on the given data at inter-class, intra-class and instance levels (G2–G4). It consists of three views, organized around a workflow going from inter-class comparison, to intra-class comparison, to instance analysis.

(G) The **concept card view**, inspired by Spinner et al.'s information cards [43], provides an overview of selected classes and concepts (e.g., selecting a set of classes from the class navigation view). Each class is represented by a card, with the class name at top. Below each class title, a histogram shows the distribution of influence scores for the concepts in that class. If the histogram skews to one side, it means the concepts are likewise skewed either positively or negatively influential. In contrast, a normal distribution indicates the majority of concepts in that class are closer to neutral (i.e., close to the 0.5 influence mid-point).

Below the histograms, a detailed dot plot shows individual concepts as circles. Circles are organized into rows corresponding to their concept clusters (labeled at left, e.g., the first row "CC9" stands for "concept cluster #9"). Circles located on the same row belong to the same concept cluster (i.e., they would be in the same cluster in the concept navigation view in (D)), even when they are distributed among different classes. The left-to-center-to-right position of a circle (along with its corresponding red-to-white-to-blue color scale) within its class card is based on its influence score along the negative-to-neutral-to-positive scale. Concepts in the same row are also linked via horizontal lines to emphasize their concept cluster grouping (supporting inter-class analysis, G3). This design is inspired by Lex et al.'s work [27]. Clicking or hovering on these links highlights the corresponding concept cluster in gold in the hexagon plot in (D), and loads the specific concept cluster in the detail view in (E).

To support additional concept analysis, we adopted the idea of periphery plots [31] to supply condensed insights to the left and right of the class cards. (I) The box plots to the left summarize the influence scores for the concepts for each row (i.e., for each concept cluster), connecting class-specific concepts to global concept clusters. The length of the box encodes the range of TCAV scores for concepts within a cluster and the vertical line shows the median. (H) To the right, the bar chart shows the number of concepts belonging to that concept cluster, based on the current selection of classes. A higher frequency number indicates that this concept cluster is likely a common characteristic of these classes in the model's mental map.

(J) Clicking the title bar of a class card loads the class in the **class-specific concept view**. This view facilitate intra-class concept inspection (G2). Each row shows one concept, sorted based on decreasing influence score (i.e., the top concept rows are more positively influential to the prediction of the class). Users can review and verify if the model is making inferences about a class based on a set of reasonable concepts. For example, if the user sees a concept relating to an image's background, as opposed to important semantic features (e.g., for a *zebra*, *grassland* as opposed to *stripes*), it may indicate that the neural network is assigning too much weight to the background of the image while ignoring foreground information that might reasonably be important (at least from a human's perspective).

In each row, we present five representative image patches of the corresponding concept so that the user can form a rough idea of what the concept is about. Additional samples can be loaded by clicking the ✚ icon beside each label.

(K) To the right of the class-specific concept view, the **instance influence matrix** visualizes local explanations about the models behavior for individual instances. This heatmap plots each row as a concept (aligned with the concepts to the left in (J)), where each column indicates an image sample in the dataset. Each cell indicates, for a particular instance, how influential that concept was in its prediction, colored using the same red-to-blue color scale from previous views. Above each column, a ✓ or ✗ icon indicates if the prediction was correct.

Instances are sorted based on classification accuracy and confidence (i.e., the left-most column is the correctly classified instance with the highest confidence and the right-most column the misclassified instance with the highest confidence). In Fig. 1, the user has scrolled all the way to right, showing incorrect predictions with the highest confidence.

(L) Clicking on a column loads its instance into the **instance view**. This card provides details about the instance (G4), including its image with concept patches that can be toggled on as overlaid semi-transparent boundaries. The color of the semi-transparent masks matches the colors of concepts in (J), its ground truth labeling, and the model's prediction and confidence. At the bottom of the card, a lollipop chart shows the concept influences specific to that particular instance.

4.4.4 Header Bar Panel

(M) The **header bar** provides control widgets allowing users to update parameters and re-run the backend pipeline. The menu includes selecting a dataset and model, selecting the model layers where the concepts are extracted, setting TCAV and ACE parameters, specifying the concept clustering methods and number of clusters (i.e., the parameter k in k-means). The number of clusters is set by default to the optimal value based on the silhouette score [37]. Users can hover on other numbers of clusters to see their scores as shown in (M).

5 Usage Scenarios

To help demonstrate how ConceptExplainer supports learning about model behavior, we present three usage scenarios exploring GoogLeNet and ImageNet. Each focuses on a different type of exploration and analysis: (#1) for individual classes and their instances, (#2) comparing the concepts between two classes, and (#3) understanding concept commonalities across many classes. We tell these stories from the perspective of Michael, a model user.

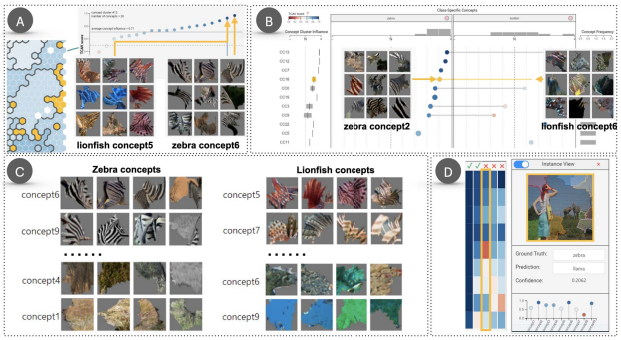

Fig. 4. Usage scenario #1 represents analysis of the *zebra* and *lionfish* classes. A full-size figure is available in the Appendix.

5.1 Usage Scenario #1: Verifying GoogLeNet's knowledge about individual classes and samples

As Fig. 4(A) shows, Michael begins by reviewing clusters in the concept navigation view, focusing on clusters with darker colors (i.e, containing more influential concepts). Loading Cluster #13 in the concept cluster detail view, he realizes the two most-influential concepts (from the *zebra* and *lionfish* classes) look like stripes. He loads the *zebra* and *lionfish* classes into the central class-specific concepts panel for subsequent investigation.

In the class-specific concepts panel, Michael sees several cross-class links between the *lionfish* and *zebra* classes, which further confirms that

conceptual overlap exists (Fig. 4(B)). He sees that the shared concepts primarily relate to stripe patterns. As both zebras and lionfish have prominent stripe patterns, GoogLeNet has learned (or formed a mental model) that stripe patterns are a way to recognize both classes.

Michael next inspects the individual concepts of the *zebra* and *lionfish* classes (Fig. 4(C)). For the *zebra* class, the three most positively influential concepts pertain to stripe patterns. As influence decreases, the concepts become more associated with noise and background features, such as representing desert or grassland. In this case, GoogLeNet is aligning with Michael's mental model, as background information is largely irrelevant to classifying these images as zebras.

Reviewing individual samples using the influence instance matrix and the instance view, Michael notices a misclassified image of *zebra* predicted as the *llama* (Fig. 4(D)). The zebras in this image are very small, and "background concepts" (such as *grasses* and *sky*) are the main concepts present. In other words, the model predicted using less-influential concepts that were present, as stripe concepts were not found. This happens several times in instances predicted as *zebra* or *lionfish*, and the model regularly predicts wrongly in such cases.

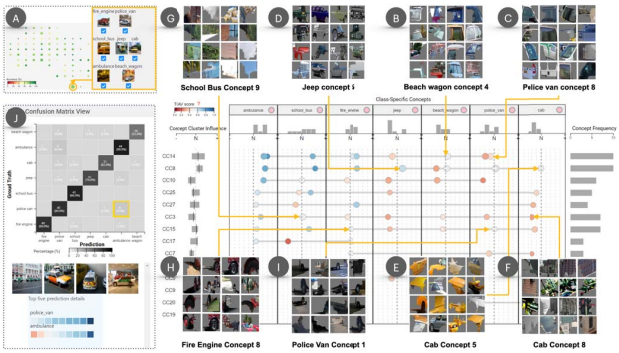

Fig. 5. Usage scenario #3 represents analysis on a group of classes. A full-size figure is available in the Appendix.

5.2 Usage Scenario #2: Spotting unreasonable concepts and data quality issues

Using the class navigation view, Michael looks at other classes similar to *zebra*. Hovering on the circle containing the *zebra* class displays a tooltip, which indicates the *tiger* class (another animal with prominent stripes) is also contained in this clique (Fig 1(O)). An adjacent clique in this view is colored in red (indicating below-average classification accuracy); the tooltip shows that this clique contains a *tiger cat* and *tabby* class. Intrigued, he selects the *tiger*, *tiger cat*, and *tabby* classes.

The class confusion matrix (Fig. 1(C)) view shows many misclassifications between *tiger cat* and *tiger*; for example, there are 11 images of *tiger cat* (ground truth) that were predicted as *tiger* by GoogLeNet. Selecting this cell loads these images; to Michael's eyes, they all indeed appear to be *tigers*, not *tiger cats*. Already, Michael can see that there is a data labeling issue in ImageNet: instances that should have been labeled as *tigers* were wrongly labeled as *tiger cats*.

Most *tiger cat* concepts appear to have negative influences in concept card view (Fig. 1(G)). Michael reviews these in the class-specific concept view (Fig. 1(J)).

He quickly scans through the concepts, and unsurprisingly, many of them seemed to be random or incomprehensible patches that do not form a collective theme. In other words, GoogLeNet did not form a sensible mental model about *tiger cat*. Interestingly, the most negatively influential concept at the bottom row presents a *tiger stripe* pattern, which helps explain why so many *tiger cats* are misclassified as *tigers*.

Because of this, Michael decides to review *tiger cat* instances that contain the *stripe* concept using the instance influence matrix and the instance view (Fig. 1(N)). Reviewing misclassified instances, he again can see that there are several images labeled as *tiger cat* but actually including *tigers* in them. Overall, Michael observes that mislabeled instances tend to include small objects of tigers. This explains why the tiger stripe was not positively influential – because it was not easily detected due to its size.

5.3 Usage Scenario #3: Understanding the neural network's knowledge on a group of classes

In Fig. 5(A), Michael investigates GoogLeNet's global mental model by reviewing the class navigation view. He notices a big circle in the bottom right corner of the visualization, indicating a clique containing many classes. The tooltip shows that the classes (7 in total) are all vehicles. He selects all of them. Reviewing these in the class-specific concepts view, he sees (as expected) many cross-class links in the class-specific concepts panel which connect influential concepts for the various vehicles. This means the neural network has likely learned common features across different cars.

Michael scans through the horizontal links across classes one by one. Fig. 5(B) and (C) show that the highest-ranked concept cluster (labeled CC14, for concept cluster #14) contains window-like patches in the *beach wagon* class, so an important "vehicle commonality" the network has likely learned is "car windows." This hypothesis is confirmed by reviewing across other concepts in this concept cluster (e.g., Fig. 5 (C) also contains window patches from the *police van* class). The next set of links in Fig. 5 (D) and (E) tend to show *car side* concepts, which are relevant features regardless of the vehicle type (in this case, between *jeep* and *cab*. Further investigation reveals both *urban background* in Fig. 5 (F) and (G), and *wheel* as influential concepts in Fig. 5 (H) and (I). Both concepts make sense at a high level ("*Cars have wheels and car pictures are usually taken in urban environments.*"), but interestingly, the *jeep* class does not consider the *urban background* concept as influential. Reviewing *jeep* instances reveals why: many *jeep* pictures are taken with the background in nature (desert or woodland settings); he cannot find *jeep* pictures taken in urban settings.

While many of the concepts make sense upon review, Michael is surprised to realize that GoogLeNet considers *windows* and *car sides* as the most salient discriminators or influencers for predicting cars, while other features like *wheels* or *headlights* are not chosen.

6 User Study

To evaluate ConceptExplainer, we designed a user study to answer two primary questions relating to design goals ("*How well does the system support the design goals (G1–G4)?*") and overall usability ("*What is the overall usability of the system?*"). Our user study consisted of two stages: i) a task stage where participants completed three representative analysis tasks in line with the four design goals listed in Sect. 3.2; and ii) a freeform analysis stage where participants freely probed model behavior using the interface. We recruited ten participants who were model users in deep learning, had not heard of concept-based explanations, and had not used ImageNet and GoogleNet before. We collected both quantitative and qualitative data, which allowed us to robustly evaluate both the ability of ConceptExplainer to support the design goals G1–G4, and also to understand the system's overall usability.

6.1 Study Design

Participants took the following procedure for the study:

(1) Training. Participants completed a short demographics questionnaire. Next, they were given a high-level introduction on TCAV, ACE, and what concept influence means. The study administrator walked the participant through available features and functionalities of ConceptExplainer; participants completed a simple training task to help familiarize themselves with the system. During the training, participants could ask questions at anytime, and were allowed to play around with the interface until they felt comfortable to proceed.

(2) Task. Participants completed the following three tasks (T1-T3):

T1: Inspect the influential concepts for a given class: (1) Identify the comprehensible concepts and state why they are comprehensible. (2) Identify the incomprehensible concepts and state why they are incomprehensible. (3) Review the concept influences; what are the items that make sense to you, and what are the items that do not make sense to you? (4) Take a look at the instances contained in the influence matrix, find interesting cases and describe why they are

interesting. This task primarily focuses *instance-level* and *single class-level analysis*, and supports G1 and G2. Each participant completed T1 on two classes: one high-performance (accuracy > 0.9) and one poor-performance (accuracy $\sim 0.1 - 0.3$); class choices for each participant were randomly selected from a high-performance set (*zebra*, *tench*, *lionfish*) and a poor-performance set (*tiger cat*, *appenzeller*, *seashore*).

T2: Given two classes that are similar and have overlapping misclassifications (i.e., images which should belong to C_2 are classified as C_3, and vice versa): (1) Identify what are the commonalities between their influential concepts. (2) Identify distinguishable influential concepts unique to each class. This task represents *instance-level* and *multi-class analysis*, and supports G2 and G3. The class pairs for T2 were randomly selected from the following sets: {*tiger cat*, *tiger*}, {*eskimo dog*, *siberian husky*}, and {*police van*, *ambulance*}. Note that if *tiger cat* was chosen during T1, it was not an option in T2.

T3: Given a group of seven related classes representing various vehicles (*fire engine*, *police van*, *school bus*, *jeep*, *cab*, *ambulance*, *beach wagon*)): (1) What are the common influential concepts across this group? Reason about the vehicle knowledge that the neural network has learned and what vehicle knowledge it lacks. This task represents *multi-class* and *global-level analysis*, and supports all four design goals (G1–G4). All participants used the same set of seven classes for this task. (Two participants had previously seen the *police van* and *ambulance* classes in T2, however their results were in line with other participants, so we do not believe this caused any study confounds during this task.)

The task order was consistent. Each participant was given a verbal description of the task (which was also available on a sheet of paper) by the administrator. Think-aloud protocol was employed during this (and the following) stage; the administrator listened to verbal utterances to help confirm that the participant was correctly performing the task and to check if the participant's interpretations of tasks and system features was correct. Upon completion of a task, the participant verbally summarized their answer(s) to the administrator.

(3) Freeform Analysis. In this stage, participants conducted an undirected, freeform analysis of ImageNet to gain insights on model behavior using CONCEPTEXPLAINER. Participants were told to use the tool until they were satisfied, but were required to spend at least 10 minutes. To prevent participants from becoming lost or frustrated, we prepared an initial (optional) motivation scenario which was only given if participants asked for guidance: "Check if the neural network is working sensibly on well performing classes." Only one participant asked for this, as we found that the others had ideated their own exploratory goals upon completing the task stage. Participants also verbally reported their thought processes during this stage.

(4) Review. Participants completed a short usability survey to rate various system components using 7-point Likert scales; they were also allowed to provide comments or critiques about the system.

6.2 Participants and Apparatus

We recruited ten participants: nine graduate computer science students at Arizona State University and one analyst with two years experience in a data company (average age = 25.8, SD = 2.08; 8 males, 2 females). Although a couple of the participants had colloquial familiarity with AI/ML, none had expertise in deep learning development or analysis, and all were unfamiliar with concept-based explanation methods. Study duration averaged 86 minutes (SD = 15).

CONCEPTEXPLAINER was shown using Google Chrome in full-screen mode on a 30" monitor at 3840×2160 resolution. Study sessions were conducted in a quiet, office environment with no distractions.

6.3 Study Results

To analyze the study, we first report quantitative ratings from the review stages' usability survey. We next report on qualitative verbal comments and responses given by participants, which were collected via the think-aloud protocol and summary answers during their tasks. To analyze these verbal comments, we used a grounded theory procedure [46] to qualitatively code comments based on assessing how the system promotes insights and supports the design goals G1–G4.

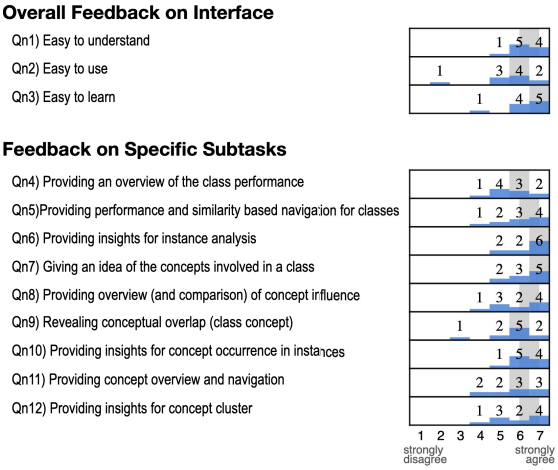

Fig. 6. Participant usability ratings about CONCEPTEXPLAINER based on the survey given during the user study's review stage. Median ratings are indicated in gray.

6.3.1 System Usability Ratings

Fig. 6 summarizes system usability based on participants ratings from the survey completed during the review stage. Ratings are broken into two types: (Q1–Q3) the overall usability and effectiveness of the system, (Q4–Q13) the usability of individual features.

In general, feedback was positive. We highlight that the system was considered easy to use (Q1), learn (Q2), and understand (Q3), and that the individual interface panels, the class navigation panel (Q5–Q6), the class concept view (Q7–Q11), and the concept navigation panel (Q12–Q13) were all positively regarded.

6.3.2 Feedback from Novice Users

The survey ratings show that CONCEPTEXPLAINER achieves good usability. Here, by analyzing the verbal comments and responses of participants, we reflect on the types of insights users can generate with the system, specifically in the context of the design goals (G1–G4).

(G1) Contextualizing concepts using the concept navigation view. The concept navigation view was heavily used and considered positively as a way to explore new concepts. In particular, several participants inspected similar concepts using thi view as a way to verify their intuition or mental model about a concept of interest. "*I'm trying to see what are concepts similar to this one because I want to know if my thought is correct*" (p5). "*This map thing [the concept navigation view] is cool. I can compare this [concept] with similar ones*" (p8). "*The navigation view helps me understand a concept better*" (p9).

(G2) The concept view is effective revealing class-specific concepts. Several participants regarded the class-specific concept view (Fig. 1 (J)) as effective in revealing influential concepts. Specifically, the list of concepts ranked by concept influences provided them with intuitive insights for network behaviors. "*This is helpful because I can immediately see what the neural network learned about the class*" (p3). "*It doesn't make sense that top concepts are background patches...middle concepts should be moved up. It's clear in the view*" (p8).

Participants generally liked the class-specific concept view, and felt most concepts had "good quality" and were easily interpretable. However, participants also sometimes encountered incomprehensible or noisy concepts, which could hinder their analysis. In part, this is a byproduct of the concept generation pipeline used in CONCEPTEXPLAINER's backend, which automatically extracts and defines concepts (bottom up) as opposed to creating them by hand (top down). We discuss this issue in Sect. 7

(G3) Concept links are effective revealing inter-class concept overlap. Participants reported it was easy to identify conceptual links between classes and understand them by using the concept view. "*I can see that concept cluster 5 are green areas in the two classes.*" (p2, analyzing the *tiger cat* and *tiger* classes), "*The first link is white-fur*

stuff." (p5, analyzing the *Eskimo dog* and *Siberian husky* classes), "*I can see silhouette of cars, tire and window [being the common concepts].*" (p7, analyzing the *police van* and *ambulance* classes). As an extension of the current system, two participants (p1, p9) requested the ability to compare two or multiple concepts in parallel.

(G4) Instance analysis was found to be useful for identifying data quality issues. A couple of participants identified data quality issues solely using instance analysis view. "*Clearly, real tigers are mislabeled as tiger cats in the testing set*" (p3). "*Why is there a tiger stripe in tiger_cat_concept_8? It doesn't make sense ... Oh I see why, because there are tiger pictures in ground truths*" (p8).

Although participants in general like the instance analysis views, four participants (p1, p5, p6, p8) mentioned it was hard for them to link concepts with their segments in the image instance view. Three participants (p1, p3, p7) mentioned that instances sometimes seemed counterintuitive. "*Some instances are highly (positively) influenced by the concepts but are still misclassified, why?*" (p7, while analyzing *tiger cat*). "*Why are 'police' letters negatively influential here?*" (p3, analyzing *police van*). "*Why are cab cases all negatively influenced but correctly classified?*" (p1, comparatively analyzing multiple car classes). The current interface does not support answering these sorts of "*why?*" questions; see Sect. 7 for discussion on this.

7 Discussion and Conclusion

Based on the set of conducted usage scenarios and a robust user study, we demonstrate how ConceptExplainer addresses a number of design challenges (C1–C4) and goals (G1–G4) that are important to the problem of concept-based explanation for model users. In particular, model users can effectively explore and reason about model behavior at different granularities (i.e., instance, class, and global levels). Below, we discuss several takeaways and lessons learned from our experiences in designing and evaluating ConceptExplainer.

User Study Takeaways. In general, the user study demonstrated the overall performance of our system from a human-centric perspective, indicating that it provides good usability and successfully supports the design goals (G1–G4), while also illustrating nuances and complexities in concept-based explanations.

One interesting takeaway from our study is that participants tended to use the instance analysis view more than we expected; based on feedback, we see several potential avenues for future extensions to enrich functionality. For example, the current instance analysis view provides instance-level explanations in terms of individual concept influences. A logical next step could be illustrating how the interplay of different concepts influence an instance's prediction. For example, both *snowy background* and *white fur* are positively influential for the *white wolf* class. When these two concepts co-occur in an instance, do they increase the likelihood of a *white wolf* prediction, compared to if only one concept is present? To achieve this type of fine-grained analysis, we plan to adopt techniques such as VRX [14] in future versions of ConceptExplainer.

Serendipitously Supporting an Unexpected Task. Our user study also revealed an explanatory task supported by ConceptExplainer that we did not intentionally support. During the freeform analysis stage, several participants analyzed if the model "understood" a class by quickly tabbing through instances in the class-specific concept view ((J) in Fig. 1). This allowed them to comprehend a rough estimate of the types of instances making up the class, and to use this as a basis to understand the influential concepts that the model has learned for the class. This "fact-checking" action was unexpected to us, but demonstrates the flexibility of the system's visual analytics to support diverse actions to interpret deep learning behavior.

ConceptExplainer's Suitability for Expert Users. To understand ConceptExplainer's potential suitability for experts ("model developers and builders" as defined by Hohman et. al. [19]), we conducted pair analytic sessions [2] and semi-structured interviews with three deep learning experts (Ph.D. students with 3+ years experience in AI research) to formatively assess how the system could be extended to benefit their needs and workflows.

Each opined that the system was easy to understand once the concept method was illustrated and a walk through of the interface was given. To better support expert usage scenarios, it was suggested that concept explanations could be juxtaposed with visualizing the inner architecture and logic of the neural network. Such "opening up the black box," when faceted with concepts, could enable novel ways to connect peculiar or unexpected model behavior. For example, when training a model, tools like ConceptExplainer can be used to identify the conceptual root cause of misclassifications between two similar classes, or to identify issues in ground truth labeling. One expert also suggested we augment the interface to let users focus on revealing concept distinguishability, to more explicitly demonstrate how the neural network discriminates between classes. In the future, we plan to even automate these suggested insights such as computationally determining conceptual root cause of misclassification and recommending most distinguishable concepts between classes based on similarity metrics.

Actionable Insights. Although our system design is centered on helping model users, the design and validation process surfaced additional ways that it could be helpful for model developers. These include: (1) Slice discovery [11], which is spotting underperforming subclasses of a performant class by comparing major influential concepts with data samples and understanding how the training data should be augmented to account for intra-class data imbalance. (2) Exploring counter-intuitive concepts, which—when present at a collective level—might indicate potential issues in the model's architecture or parameterization. For this point, the model user would likely need to collaborate with a model developer who could better investigate the model's neural architecture to confirm and fix these issues.

Current System Limitations and Future Work. Throughout ConceptExplainer's design and evaluation process, we also identified several limitations in the current system, suitable for investigation as future work. As an example, study participants commonly compared between 2–5 classes (with under 30 total concept clusters) in the concept card view (Figure 1(G); it is likely this panel would not scale much beyond this count. Another issue is that some concepts were not comprehensible to study participants. Unfortunately, there is no easy way around this issue; as automatic concept discovery invariably introduces some amount of noise [15]. As novel concept discovery methods (which hopefully reduce noise) are developed, they can be integrated into our system.

Another potential limitation deals with the use of ImageNet, which was tested on concepts generated from 143 (out of 1,000) classes. While ImageNet is a well-validated dataset for image classification tasks, it is possible that scaling to the full set of classes, or testing on additional datasets (e.g., medical images) and accompanying models might produce different user experiences. We plan to investigate issues like these in future iterations of ConceptExplainer.

Concept-based Explanations in Other Domains. Currently, concept-based explanations have primarily been applied in image classification scenarios. A future work can apply the concept-based explanation for bias detection and mitigation [26]. The concept-based explanation can be useful to discover irrelevant features that are associated with classes. Also, our system is optimized for the image classification task, as opposed to other domains (e.g., natural language processing). As a future step, we would like to explore the use of visual analytics as a modality for probing and exploring concept-based explanations applied to other underexplored domains, such as speech recognition or time series prediction. Such domains bring their own complexities; e.g., in image classification, a user can directly observe the pixels in a set of image patches representing a concept, but there are no direct analogs to this process in many other domains. Despite this, visualization may prove to be a key approach for this, due to the power of visualization in being able to graphically render abstract data in ways that reveal patterns and insights.

8 Acknowledgement

This research was supported in part by the U.S. National Science Foundation through grant DUE-2216452.

REFERENCES

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk. Slic superpixels. Technical report, 2010.

[2] R. Arias-Hernandez, L. T. Kaastra, T. M. Green, and B. Fisher. Pair analytics: Capturing reasoning processes in collaborative visual analytics. In *2011 44th Hawaii international conference on system sciences*, pp. 1–10. IEEE, 2011.

[3] A. Boggust, B. Carter, and A. Satyanarayan. Embedding comparator: Visualizing differences in global structure and local neighborhoods via small multiples. In *27th International Conference on Intelligent User Interfaces*, pp. 746–766, 2022.

[4] D. V. Carvalho, E. M. Pereira, and J. S. Cardoso. Machine learning interpretability: A survey on methods and metrics. *Electronics*, 8(8):832, 2019.

[5] Z. Chen, Y. Bei, and C. Rudin. Concept whitening for interpretable image recognition. *Nature Machine Intelligence*, 2(12):772–782, 2020.

[6] S. Dargan, M. Kumar, M. R. Ayyagari, and G. Kumar. A survey of deep learning and its applications: a new paradigm to machine learning. *Archives of Computational Methods in Engineering*, 27(4):1071–1092, 2020.

[7] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

[8] S. Dong, P. Wang, and K. Abbas. A survey on deep learning and its applications. *Computer Science Review*, 40:100379, 2021.

[9] F. K. Došilović, M. Brčić, and N. Hlupić. Explainable artificial intelligence: A survey. In *2018 41st International convention on information and communication technology, electronics and microelectronics (MIPRO)*, pp. 0210–0215. IEEE, 2018.

[10] R. Duggal, S. Freitas, C. Xiao, D. H. Chau, and J. Sun. Rest: Robust and efficient neural networks for sleep monitoring in the wild. In *Proceedings of The Web Conference 2020*, pp. 1704–1714, 2020.

[11] S. Eyuboglu, M. Varma, K. Saab, J.-B. Delbrouck, C. Lee-Messer, J. Dunnmon, J. Zou, and C. Ré. Domino: Discovering systematic errors with cross-modal embeddings. *arXiv preprint arXiv:2203.14960*, 2022.

[12] R. C. Fong and A. Vedaldi. Interpretable explanations of black boxes by meaningful perturbation. In *Proceedings of the IEEE international conference on computer vision*, pp. 3429–3437, 2017.

[13] O. Fried, S. DiVerdi, M. Halber, E. Sizikova, and A. Finkelstein. Isomatch: Creating informative grid layouts. In *Computer graphics forum*, vol. 34, pp. 155–166. Wiley Online Library, 2015.

[14] Y. Ge, Y. Xiao, Z. Xu, M. Zheng, S. Karanam, T. Chen, L. Itti, and Z. Wu. A peek into the reasoning of neural networks: Interpreting with structural visual concepts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2195–2204, 2021.

[15] A. Ghorbani, J. Wexler, J. Y. Zou, and B. Kim. Towards automatic concept-based explanations. In *Advances in Neural Information Processing Systems*, pp. 9273–9282, 2019.

[16] L. H. Gilpin, D. Bau, B. Z. Yuan, A. Bajwa, M. Specter, and L. Kagal. Explaining explanations: An overview of interpretability of machine learning. In *2018 IEEE 5th International Conference on data science and advanced analytics (DSAA)*, pp. 80–89. IEEE, 2018.

[17] Y. Goyal, A. Feder, U. Shalit, and B. Kim. Explaining classifiers with causal concept effect (cace). *arXiv preprint arXiv:1907.07165*, 2019.

[18] Y. He, X. Zhang, and J. Sun. Channel pruning for accelerating very deep neural networks. In *Proceedings of the IEEE international conference on computer vision*, pp. 1389–1397, 2017.

[19] F. Hohman, M. Kahng, R. Pienta, and D. H. Chau. Visual analytics in deep learning: An interrogative survey for the next frontiers. *IEEE transactions on visualization and computer graphics*, 25(8):2674–2693, 2018.

[20] F. Hohman, H. Park, C. Robinson, and D. H. P. Chau. S ummit: Scaling deep learning interpretability by visualizing activation and attribution summarizations. *IEEE transactions on visualization and computer graphics*, 26(1):1096–1106, 2019.

[21] F. Hohman, K. Wongsuphasawat, M. B. Kery, and K. Patel. Understanding and visualizing data iteration in machine learning. In *Proceedings of the 2020 CHI conference on human factors in computing systems*, pp. 1–13, 2020.

[22] M. Jaderberg, A. Vedaldi, and A. Zisserman. Speeding up convolutional neural networks with low rank expansions. *arXiv preprint arXiv:1405.3866*, 2014.

[23] S. Kambhampati, S. Sreedharan, M. Verma, Y. Zha, and L. Guan. Symbols as a lingua franca for bridging human-ai chasm for explainable and advisable ai systems. *arXiv preprint arXiv:2109.09904*, 2021.

[24] B. Kim, M. Wattenberg, J. Gilmer, C. Cai, J. Wexler, F. Viegas, et al. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *International conference on machine learning*, pp. 2668–2677. PMLR, 2018.

[25] P. W. Koh, T. Nguyen, Y. S. Tang, S. Mussmann, E. Pierson, B. Kim, and P. Liang. Concept bottleneck models. In *International Conference on Machine Learning*, pp. 5338–5348. PMLR, 2020.

[26] B. C. Kwon, J. Lee, C. Chung, N. Lee, H.-J. Choi, and J. Choo. DASH: Visual Analytics for Debiasing Image Classification via User-Driven Synthetic Data Augmentation. *Eurographics Conference on Visualization (EuroVis)-Short Papers.*, 2022. doi: 10.2312/evs.20221099

[27] A. Lex, N. Gehlenborg, H. Strobelt, R. Vuillemot, and H. Pfister. Upset: visualization of intersecting sets. *IEEE transactions on visualization and computer graphics*, 20(12):1983–1992, 2014.

[28] Q. V. Liao, D. Gruen, and S. Miller. Questioning the ai: informing design practices for explainable ai user experiences. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pp. 1–15, 2020.

[29] A. Mishra, U. Soni, J. Huang, and C. Bryan. Why? why not? when? visual explanations of agent behavior in reinforcement learning. *CoRR*, abs/2104.02818, 2021.

[30] S. Mohseni, N. Zarei, and E. D. Ragan. A multidisciplinary survey and framework for design and evaluation of explainable ai systems. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 11(3-4):1–45, 2021.

[31] B. Morrow, T. Manz, A. E. Chung, N. Gehlenborg, and D. Gotz. Periphery plots for contextualizing heterogeneous time-based charts. In *2019 IEEE visualization conference (VIS)*, pp. 1–5. IEEE, 2019.

[32] J. Nielsen. Iterative user-interface design. *Computer*, 26(11):32–41, 1993.

[33] C. Olah, N. Cammarata, L. Schubert, G. Goh, M. Petrov, and S. Carter. An overview of early vision in inceptionv1. *Distill*, 5(4):e00024–002, 2020.

[34] C. Park, S. Yang, I. Na, S. Chung, S. Shin, B. C. Kwon, D. Park, and J. Choo. VATUN: Visual Analytics for Testing and Understanding Convolutional Neural Networks. *Eurographics Conference on Visualization (EuroVis)-Short Papers.*, 2021.

[35] H. Park, N. Das, R. Duggal, A. P. Wright, O. Shaikh, F. Hohman, and D. H. P. Chau. Neurocartography: Scalable automatic visual summarization of concepts in deep neural networks. *IEEE Transactions on Visualization and Computer Graphics*, 2021.

[36] M. Ribera and A. Lapedriza. Can we do better explanations? a proposal of user-centered explainable ai. In *IUI Workshops*, vol. 2327, p. 38, 2019.

[37] P. J. Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65, 1987.

[38] A. Saha, A. Subramanya, K. Patil, and H. Pirsiavash. Role of spatial context in adversarial robustness for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 784–785, 2020.

[39] W. Samek, G. Montavon, S. Lapuschkin, C. J. Anders, and K.-R. Müller. Explaining deep neural networks and beyond: A review of methods and applications. *Proceedings of the IEEE*, 109(3):247–278, 2021.

[40] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pp. 618–626, 2017.

[41] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

[42] V. Sivaraman, Y. Wu, and A. Perer. Emblaze: Illuminating machine learning representations through interactive comparison of embedding spaces. In *27th International Conference on Intelligent User Interfaces*, pp. 418–432, 2022.

[43] T. Spinner, U. Schlegel, H. Schäfer, and M. El-Assady. explainer: A visual analytics framework for interactive and explainable machine learning. *IEEE transactions on visualization and computer graphics*, 26(1):1064–1074, 2019.

[44] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.

[45] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826,

2016.

[46] R. Thornberg, K. Charmaz, et al. Grounded theory and theoretical coding. *The SAGE handbook of qualitative data analysis*, 5:153–69, 2014.

[47] L. Van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[48] W. Wen, C. Wu, Y. Wang, Y. Chen, and H. Li. Learning structured sparsity in deep neural networks. *Advances in neural information processing systems*, 29, 2016.

[49] N. Xie, G. Ras, M. van Gerven, and D. Doran. Explainable deep learning: A field guide for the uninitiated. 2020.

[50] F. Xu, H. Uszkoreit, Y. Du, W. Fan, D. Zhao, and J. Zhu. Explainable ai: A brief survey on history, research areas, approaches and challenges. In *CCF international conference on natural language processing and Chinese computing*, pp. 563–574. Springer, 2019.

[51] C.-K. Yeh, B. Kim, S. Arik, C.-L. Li, T. Pfister, and P. Ravikumar. On completeness-aware concept-based explanations in deep neural networks. *Advances in Neural Information Processing Systems*, 33:20554–20565, 2020.

[52] J. Yuan, C. Chen, W. Yang, M. Liu, J. Xia, and S. Liu. A survey of visual analytics techniques for machine learning. *Computational Visual Media*, 7(1):3–36, 2021.

[53] Q.-s. Zhang and S.-C. Zhu. Visual interpretability for deep learning: a survey. *Frontiers of Information Technology & Electronic Engineering*, 19(1):27–39, 2018.

[54] Z. Zhao, P. Xu, C. Scheidegger, and L. Ren. Human-in-the-loop extraction of interpretable concepts in deep learning models. *IEEE Transactions on Visualization and Computer Graphics*, 2021.

[55] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2921–2929, 2016.