Towards Visual Explainable Active Learning for Zero-Shot Classification

Shichao Jia, Zeyu Li, Nuo Chen, Student Member, IEEE, and Jiawan Zhang, Senior Member, IEEE

Abstract—Zero-shot classification is a promising paradigm to solve an applicable problem when the training classes and test classes are disjoint. Achieving this usually needs experts to externalize their domain knowledge by manually specifying a class-attribute matrix to define which classes have which attributes. Designing a suitable class-attribute matrix is the key to the subsequent procedure, but this design process is tedious and trial-and-error with no guidance. This paper proposes a visual explainable active learning approach with its design and implementation called semantic navigator to solve the above problems. This approach promotes human-AI teaming with four actions (ask, explain, recommend, respond) in each interaction loop. The machine asks contrastive questions to guide humans in the thinking process of attributes. A novel visualization called semantic map explains the current status of the machine. Therefore analysts can better understand why the machine misclassifies objects. Moreover, the machine recommends the labels of classes for each attribute to ease the labeling burden. Finally, humans can steer the model by modifying the labels interactively, and the machine adjusts its recommendations. The visual explainable active learning approach improves humans' efficiency of building zero-shot classification models interactively, compared with the method without guidance. We justify our results with user studies using the standard benchmarks for zero-shot classification.

Index Terms—Active Learning, Explainable Artificial Intelligence, Human-AI Teaming, Mixed-Initiative Visual Analytics

1 Introduction

Zero-shot classification is a dominant and promising learning paradigm, aiming to recognize instances that are unseen during training. Compared with traditional supervised classification, which needs enormous labeled training data and can only classify instances covered by seen classes, zero-shot classification is more applicable and more like the human reasoning process. Humans can recognize unseen objects once we provide the description of objects. For example, we can teach children to recognize zebras by describing that zebras look like horses with black and white stripes. Zero-shot classification is based on this idea, and due to its practical value, it gains lots of attention in recent
machine learning and computer vision studies [69].

One critical process to achieve zero-shot classification is building a class-attribute matrix (matrix with rows as classes and columns as attributes, see Fig. 2). The attributes are human interpretable semantic words designed by experts, such as big eyes, black hair, etc. We can distinguish classes (including seen and unseen classes) by describing whether or how likely they have the attributes. Once the machine learns the concepts of attributes from seen classes, it can recognize objects from unseen classes leveraging the description using attributes. However, constructing the class-attribute matrix is very challenging. Firstly, this process is hard to be automated with interpretability preserved. Designing attributes usually involves manually picking descriptive words for the images under consideration by domain specialists [21][37]. Yu et al. [73] try to avoid human labor to build the class-attribute matrix automatically. However, the attributes designed by this approach have no concise names and may not be semantically interpreted. Secondly, the attributes used in the zero-shot classification are selective. Not all attributes are necessary and helpful for zero-shot classification, and some are even harmful and lead to wrong predictions [21]. Thirdly, once the analyst comes up with new attributes, he/she needs to go through the tedious and painful process to label every class. In summary, up to now, it can be tricky to design the class-attribute matrix for zero-shot classification using solely machine-centric processes or solely human-centric processes.

One promising approach to the problem is teaming the machine and analysts with the machine guiding the attribute design process. Approaches to alleviating the labeling burden with the machine querying labels from humans, aka active learning [60], have been widely used in traditional supervised learning. While in this paper, instead of asking for labels of instances, the machine seeks attributes to differentiate classes. However, active learning merely recognizes humans as labeling machines so that analysts can not steer the process. It has shown that combining visual interactive labeling with active learning in a mixed-initiative way can accelerate the labeling process [3][4]. Meanwhile, explainability is a crucial factor in promoting human-AI communication and trust, we step back to think about improving human-AI teaming. Considering interpretability is inevitable in making human and machine understandable. Contrastively, humans, describe classes leveraging high-level interpretable attributes, which is not the logic to distinguish objects for the machine. This semantic gap between the machine and the analyst seriously hinders the effectiveness of human-AI teaming.

To solve this technical challenge, we bridge the two spaces of features and attributes into one shared space facilitating mutual coordination. We then design a semantic map to visualize this space as the main interface for cooperative attribute design. This visualization enables analysts to reason the misclassification of the zero-shot model and inspire them to come up with new attributes. Based on this, we further design and implement a mixed-initiative visual analytics system called semantic navigator to facilitate other teaming tasks. The semantic navigator guides analysts with contrastive questions to elicit attributes by comparing two class exemplars. Contrastive questions require analysts to focus on the differences between classes, which is easier than the method to let humans come up with new attributes directly. Meanwhile, we leverage semi-supervised support vector machines to provide label recommendations and enable analysts to interactively provide feedback to the recommendations.

In summary, our technical contributes are as follows:

- We propose a human-AI teaming approach called visual explainable active learning to elicit human knowledge and enable analysts to steer models at the same time.
- We design and implement a visual analytics system called semantic navigator that guides analysts to interactively build zero-shot classification models.
- We evaluate our technique with case studies and user studies. We find the semantic navigator can improve the efficiency of analysts and accuracy of zero-shot models compared with the method without guidance.

2 RELATED WORK

In this section, we describe prior work relevant to our study, including methods of designing visual attributes (Sec. 2.1), interactive classification (Sec. 2.2) and guidance in visual analytics (Sec. 2.3).

2.1 Designing Visual Attributes

Traditionally, visual attributes are designed by manually picking a set of words that are descriptive and distinguishable for the images [21][37]. Therefore, this process needs special expertise and great effort for domain experts, which is expensive for laypersons. To alleviate the burdens, Berg et al. [2] propose an automatic approach to tag images with visual attributes by mining text and image data sampled from the Internet. However, the discovered attributes from text-mining can contain irrelevant semantics for the classification task at hand or may not be separable in the visual feature space. Considering this, Parikh et al. [51] propose to build nameable and discriminative attributes with human-in-the-loop. Based on that, Duan et al. [16] propose a method to discover localized attributes for fine-grained recognition. Yu et al. [23] propose an automatic method to design discriminative category-level attributes. However, the learned attributes will not have concise semantics as the manually specified attributes. Moreover, for unseen classes, this approach needs humans to model the similarity between seen classes and unseen classes, which is more difficult and unintuitive than directly specifying visual attributes of unseen classes. Considering interpretability is inevitable in making human and machine communication and trust, we step back to think about improving human efficiency when designing visual attributes with human-in-the-loop.
2.2 Interactive Classification

Interactive machine learning [39], or machine teaching [41], places a human-centered perspective on the building process of machine learning models. This paradigm leverages end-user involvement to enable rapid, focused, and incremental interaction cycles with various applications [44]. Visualization as the interface between humans and machines plays a crucial role in seamlessly fitting analytics into existing interactive process [19]. We mainly discuss methods of interactive building classification models.

Labeling data instances is an important task for interactive classification. Active learning [60] takes the machine-centric approach to ease the labeling burden by querying labels from users. In contrast, visual-interactive labeling [59] takes the human-centric approach leveraging humans’ ability to identify patterns with interactive visualization. Bernard et al. [15] systematically compare the performance of the two paradigms and summarize different user strategies for visual interactive labeling. Heimerl et al. [29] compare different combinations of active learning and visual interactive learning for document retrieval. Similarly, Höferlin et al. [30] combine the two approaches for video analytics, and Sun et al. [63] present Label-and-Learn system to help analysts understand the classifier’s behavior. Jose Gustavo et al. [50] propose an incremental visual classification method that enables users to steer the classification process by annotating items inside the similarity tree of data. Liu et al. [42] propose a visual analytics method to improve the crowding issue of visualizations while improving the classification model. Xiang et al. [72] solve a similar problem with a different approach called DataTableDebugger. Felix et al. [22] propose exploratory labeling to facilitate label ideation, specification, and refinement with machine-driven recommendations. Our research expands the research space considering attribute ideation guided by the machine and label recommendations with human feedback for zero-shot classification.

Sahoo et al. [57] step first to study interactive zero-shot classification for diagnosing and steering its procedure. However, this process is under the assumption that the class-attribute matrix is provided. In contrast, our research focuses on a more fundamental problem that how to facilitate the constructing process of the class-attribute matrix. Analysts build classification models by describing classes using visual attributes via collaboration with the machine. Another advantage of our method is that the whole process is naturally interpretable when visual features extracted from the deep neural networks are involved. Our technique contributes to another way of interpretable machine learning [13-33]. Previous works either open the black boxes [20, 41, 43, 45, 62, 68] which need analysts to have machine learning knowledge or explain the black boxes using surrogate models [33, 46], which adds another layer of uncertainty.

2.3 Guidance

Guidance is a mixed-initiative process to assist analysts with a set of visual means to close the knowledge gap via the collaboration of the machine. Ceneda et al. [10] present a systematic review of guidance approaches in visual analytics based on the dimensions of the knowledge generation process. We only discuss the most relevant methods that guide model construction and parameter refinement processes.

Dis-Function [7] enables analysts to directly manipulate the data points to express the similarity by the proximity, and machines transform the interactions into the weights of distance functions. This idea has inspired a new interaction paradigm called semantic interaction [17] to infer analysts’ interests by manipulating visual items directly. For example, Endert et al. [18] propose requireSPIRE to steer text models by spatially clustering data points. Moreover, Dowlinget et al. extend semantic interaction to a bidirectional pipeline to infer both the observation and attribute weights [15]. Sacha et al. [55] propose a guided visual analytics system called SOMFlow for exploratory cluster analysis using self-organizing maps. Specifically, SOMFlow provides a ranked list of recommendations and visual cues to guide analysts. Cavallo et al. propose Clustrophile 2 [9] for guided visual clustering analysis. This system guides analysts to explore the large clustering space and fine-tune the clustering parameters. Other techniques [6, 45, 68, 70] aim to provide suggestions to guide the visualization design or graphic design.

Our work focuses on guided supervised classification for zero-shot learning by construing the class-attribute matrix, which has not been explored yet.

3 Human-AI Team for Zero-Shot Classification

3.1 Problem Statement

This section briefly introduces the basic concepts of zero-shot classification and formally states the research problem, which is essential for subsequent discussion.

In zero-shot classification, the classes covered by training data are seen classes, denoted as \( C_t = \{ c^t_i \mid i = 1, ..., N_t \} \). Contrastively, classes for unlabeled testing instances are unseen classes, denoted as \( C_u = \{ c^u_i \mid i = 1, ..., N_u \} \). The seen classes and unseen classes are disjoint, namely \( C_t \cap C_u = \emptyset \). Both training and testing data come from a \( D \) dimensional real space denoted as \( X \in \mathbb{R}^D \), namely feature space. We denote \( D_t = \{ x^t_i \in X \} \) as the training dataset, in which \( x^t_i \) is the labeled instance in the feature space, and \( x^u_i \) is the corresponding class label in the seen classes. Similarly, we denote \( D_u = \{ x^u_i \in X \} \) as the testing dataset, in which \( x^u_i \) is the unlabeled instance in the feature space. Given the training dataset \( D_t \), zero-shot classification aims to learn a classifier \( f(\cdot) : X \rightarrow C_u \) that predicts the labels of testing dataset \( D_u \).

As there are no instances of unseen classes during training, the machine needs some auxiliary information to transfer knowledge from the seen classes to unseen classes. The most widely used auxiliary information are attributes. Attributes are human understandable words to describe objects with meaningful characteristics. These attributes construct a space namely semantic space \( \mathcal{S} \in \mathbb{R}^M (M < D) \), with each dimension being one attribute. Attributes can be binary (0/1) to symbolize their presence, or continuous to represent the degree of confidence. In this paper, we consider binary attributes as the initial step of the research. In the semantic space, each class is represented as a semantic vector namely class prototype. For example, in Fig. 2 there are four visual attributes to describe classes, including black, white, striped, and water. The zebra prototype is specified as \([1, 1, 1, 0]\). We denote the prototypes of seen classes as \( P_t = \{ p^t_i | p^t_i \in \mathcal{S} | i = 1, ..., N_t \} \), and those of unseen classes as \( P_u = \{ p^u_i | p^u_i \in \mathcal{S} | i = 1, ..., N_u \} \). \( P_t \), \( P_u \) are vital to obtain a zero-shot classifier \( f(\cdot) \). Aligning all the class prototypes vertically, we get the class-attribute matrix, denoted as \( S \in \mathbb{R}^{N_t \times M} \). Our paper aims to build the binary class-attribute matrix interactively and train a good function \( f(\cdot) \) as well.

3.2 Visual Explainable Active Learning

Analysts without systematic domain knowledge can not easily transform the implicit knowledge to the structural class-attribute matrix. Although humans make decisions using attributes every minute, this process is usually subconscious and difficult to be characterized. To tackle this issue, we propose visual explainable active learning for zero-shot classification with four actions (ask, explain, recommend, respond) to guide the design process.

Ask. Considering dialogue is a natural way for human interactions, the machine can elicit the attributes by asking questions. A naive approach is to ask for attributes given pictures of specific categories directly. However, this approach is inefficient because direct questions are not well-defined and may elicit irrelevant attributes to the classification.

Consider this, we adopt contrastive questions by letting analysts compare two representative images of different categories and discover attributes that differentiate the two classes. These questions are simple enough but powerful by leveraging humans’ analogical reasoning [24] to discover deeper structural characteristics. Notice that we do not constrain the forms of questions, as there are still limited studies on how different kinds of questions impact the elicitation process [64].

Explain. During the training process, visualization can play a critical role in enabling analysts to interpret and diagnose results from the machine. One fundamental technical problem in this setting is that features from the machine and attributes elicited from the analysts lie in the two different spaces. The semantic gap between the two spaces

© 2021 IEEE. This is the author’s version of the article that has been published in IEEE Transactions on Visualization and Computer Graphics. The final version of this record is available at: [xx.xxxx/TVCG.201x.xxxxxx/3]
hinders the effectiveness of human-AI teaming. It is vital to bridge the two spaces using visualization to enhance the explainability of the results. We solve this problem by mapping the feature space and semantic space into one shared space, and we name the new space as mutual mental space. To explain the zero-shot classifier results in the mutual mental space, we leverage explainable case-based reasoning [38].

Numerous studies [47] have demonstrated that case-based reasoning involving various forms of matching and prototyping is fundamental to effective decision-making strategies. In this paper, we explain the zero-shot classifier results by visualizing the relationships between prototypes and corresponding desirable exemplars (details in Sec. 4.3).

**Recommend and Respond.** Meanwhile, eliciting a new attribute in every interaction loop is just one step towards building the class-attribute matrix. Labeling classes using the attributes is another burden. We enable the machine to recommend labels to fill the blanks in the class-attribute matrix. Moreover, analysts can respond to recommendations by interactively modifying class labels through visualization to close the interaction loop. We call the above approach containing four actions (ask, explain, recommend and respond) as visual active learning for zero-shot classification.

**4 SEMANTIC NAVIGATOR**

**4.1 System Overview**

We instantiate the visual explainable active learning for zero-shot classification as a visual analytic approach called semantic navigator. We first discuss how we construct the mutual mental space as this is the technical fundamentals for visualization. Then we provide the overview of visualization views and interaction loop. Finally, we cover the training and testing process of the underlying zero-shot classifier, as its running process closely relates to visualization and interaction.

**Mutual Mental Space.** We use PCA to project these features into the mutual mental space. These features are usually extracted from a pre-trained deep neural network. PCA decorates the dimensions of features so that we can predict these dimensions independently rather than jointly. Besides, we apply PCA to gain computational benefits by reducing dimensionality, considering the need for real-time interaction.

We denote the PCA projection matrix as $M_{PCA} \in \mathbb{R}^{d \times D} (2 < d < D)$. Notice that we choose the dimensionality $d$ empirically in this paper, and the projected space using PCA is still in high dimensional space. For example, in Sec. 5, we set $d = 500$ when the feature space is in 2048-dimensional space. Besides, $M_{PCA}$ is computed over all the training data and not class-specific. In the mutual mental space, the data for each class usually form a cluster. We assume that each cluster center is our target semantic representation, namely visual exemplars.

Then we map the prototypes into the new space by predicting the visual exemplars from the class prototypes.

**Visualization Views.** The semantic map (Fig. 3(c)) is the critical visualization of the semantic navigator. It explains the status of the zero-shot model, visualizes the label recommendations, and enables feedback interaction from analysts. Each contour in the semantic map presents one class cluster. The interactive zero-shot classification targets to narrow the gap between each pair of the class prototype (yellow dot) and corresponding visual exemplar (black dot). The colors of the striped contours stand for the recommended labels of the current attribute. Besides, the semantic navigator contains three more views: a hint view (Fig. 3(a)), a matrix view (Fig. 3(f)), and a line chart (Fig. 3(g)). The hint view presents a ranked list of questions to elicit attributes from the analyst. Visualizing the class-attribute matrix enables the analyst to scrutinize the externalized knowledge. The line chart presents the training and testing accuracy and enables monitoring status of the current zero-shot classifier.

**Interaction Loop.** The workflow closes the loop with four actions (ask, explain, recommend, respond), seen in Fig. 3(b). The machine asks contrastive questions (what is the most visible attribute between classes A and B except existing attributes) to guide analysts to come up with new attributes. Once the analyst comes up with a new attribute, he/she can interactively specify the positive or negative status of partial classes using lasso interaction. The machine recommends the rest classes to ease the labeling burden. After specifying a new attribute, the analyst can click the submit button. The semantic navigator will...
What is the most visible attribute between classes A and B except attribute $a_1$.

When the class-attribute matrix is not empty, the semantic navigator selects a pair of classes that the zero-shot classifier confuses most. When all categories have different prototypes, the semantic navigator selects a pair of classes (A and B in Fig. 4(a)) that the zero-shot classifier confuses most. We then refer to the existing class-attribute matrix to find which attributes differentiate classes A and B. Suppose there are three attributes $a_1$, $a_2$ and $a_3$. Only attribute $a_1$ differentiates classes A and B while attributes $a_2$ and $a_3$ of classes A and B are the same (see Fig. 4(b)). The question is provided by asking what is the most visible attribute between classes A and B except attribute $a_1$. The hint view ranks the questions by the percentage of misclassifications in the class-attribute matrix.

4.3 Visualization

Design Rationale. We visualize the mutual mental space by projecting the dataset $\{MPCA_x^i\} \cup V \cup \psi(P_i)$ into a 2D plane with two reasons. Firstly, the cluster structure is important to inspire analysts to define attributes since classes with similar semantics will cluster closely. Therefore, adjacent classes usually have the same attributes. Secondly, the dimension reduction can reveal the alignment of the class prototypes and the exemplars. As discussed in Sec. 4.1, the EXEM model learns a mapping function from the class prototypes to exemplars. Therefore, if class prototypes align well with the exemplars, the designed attributes have a good description of classes. We choose the t-SNE [14] as the basic algorithm, as this is a popular nonlinear visualization technique to reveal the cluster structure.

The pipeline of constructing the semantic map can be viewed in Fig. 6. The semantic map abstracts the scatterplot as class-level contours and then projects the prototypes every time the analyst adds a new attribute. Finally, the semantic map enhances the visualization with exemplar-prototype links, nearest neighbors and semantic trajectories.

Fig. 6. A conceptual pipeline of constructing the semantic map. (a) The semantic map abstracts the scatterplot as class-level contours. Meanwhile, it reprojects the prototypes every time the analyst adds a new attribute. (b) The semantic map enhances the visualization with exemplar-prototype links, nearest neighbors and semantic trajectories.
maintaining cognitive continuity is essential. One possible method is training a surrogate model such as a neural network to map from the PCA projection to the 2D t-SNE projection [65]. Then we can only update the positions of class prototypes in every iteration. However, we do not adopt this method because learning a good approximation needs to design a good neural network architecture by empirically choosing different hyperparameters and the number of layers and neurons. Instead we adopt another strategy by fixing the coordinates of \( \{ \{ M_{PCA} P_i \} \} \cup V \), and optimizing t-SNE loss function with stochastic gradient descent algorithm to update the positions of class prototypes \( \psi(P_i) \) in every iteration. This method enables faithful projection and maintains cognitive continuity with fast computation.

**Visual Enhancement.** The semantic map links the class exemplar and its corresponding prototype. This simple design enables case-based reasoning using exemplars and prototypes by comparing their relationships (details in Sec. 4.4). As exemplars and prototypes do not correspond to specific images, we enable the analyst to check their nearest neighbors to estimate their semantics. By examining the nearest images around class exemplars and prototypes, analysts can understand their semantic difference and leverage their inference ability to develop a new attribute hypothesis. Besides, the prototypes will travel around the semantic map and leave different trajectories during the design process. As we will show in Sec. 5.1, the trajectories enable analysts to track the conditions of the model and understand how different attributes influence the model. For example, some prototypes will quickly converge to the visual exemplars after several iterations, which means the designed attributes describe these classes well. However, some prototypes will jump around two or three exemplars before the final convergence, which may infer that these prototypes confuse these classes easily. Therefore, we enable analysts to click class exemplars and show the trajectories of history positions. We name these trajectories as semantic trajectories.

### 4.4 Explanation with the Semantic Map

Now we present how semantic map supports explanations. As discussed above, each contour contains most of the data of that class. Therefore, if the class prototype comes inside the contour, it means the prototype has a high probability of having similar semantics with the exemplar inside the contour (see examples in Fig. 1(a) for classes bobcat, fox, and wolf). We can sanity-check this assumption by examining the nearest neighbor images of exemplars and prototypes. This visual clue is the basic reasoning rule in our design.

When the visual exemplar comes outside the contour, we need to reason why the visual exemplar is not well aligned with its corresponding prototype. Reasoning this requires analysts to consider the context around this class and how class prototypes, contours, and exemplars interact with each other. Consider two classes A and B as the primitive case since it can be extended to cases involved multiple classes. Denote the contour of A and the class prototype of A as \( C(A) \) and \( P(A) \), respectively. The symbolism of \( B \) is the same. \( P(A) \in C(A) \) stands for the class prototype lies inside the contour. We first discuss the situation where contours A and B do not intersect. In terms of whether \( P(A) = P(B) \) and class prototypes inside the contours or not, there are five different visual patterns except the symmetrical cases.

**Pattern 1.** \( P(A) \notin C(A), P(B) \notin C(B), P(A) \notin C(B), P(B) \notin C(A) \) and \( P(A) = P(B) \) (or exchange A and B). Leveraging projection technique in Sec. 4.5, the same data points will be projected at the same place. When class prototypes collapse into one point, it means classes A and B have the same attributes. Since both \( P(A) \) and \( P(B) \) are not inside the corresponding contours, we can infer that there is still a large gap between the prototypes and exemplars. Facing this situation, analysts should compare classes A and B to find the most distinguishable attribute that differentiates them.

**Pattern 2.** \( P(A) \in C(A), P(B) \in C(A) \) and \( P(A) = P(B) \) (or exchange A and B). This pattern shows that class prototype A aligns well with exemplar A. In contrast, prototype B does not align well with exemplar B. Since prototype A and B have the same attributes, analysts should adopt the same strategy as that of pattern 1 to figure out at least one attribute that differentiates A from B. Notice that pattern 1 and pattern 2 frequently happen at the early stage of the design process. When multiple classes are involved, these patterns will result in a star graph (see the subgraph circled by the dashed line in Fig. 2(e) as an example).

**Pattern 3.** \( P(A) \notin C(A), P(B) \notin C(B), P(A) \notin C(B), P(B) \notin C(A) \) and \( P(A) \neq P(B) \) (or exchange A and B). Now analysts find at least one attribute that distinguishes class A and class B. However, the attributes do not describe both classes well. The reason for this pattern is the incompleteness of the attributes. To solve this problem, analysts should closely examine existing attributes and the corresponding exemplar images and check their missing attributes.

**Pattern 4.** \( P(A) \in C(A), P(B) \in C(A) \) and \( P(A) \neq P(B) \) (or exchange A and B). The difference between this pattern and pattern 2 is that the two prototypes are different. However, prototype B is still similar to class A. It means the current attributes of class B are not well described. Therefore, analysts should further explore more attributes that distinguish B from other classes.

**Pattern 5.** \( P(A) \in C(B) \) and \( P(B) \in C(A) \) (or exchange A and B). This pattern shows that the machine wrongly recognizes A as B, and B as A. The reason behind this pattern may come from the errors in the class-attribute matrix. Therefore, analysts should closely examine the attributes to see whether they describe the wrong things.

Hopefully, this problem can be solved by debugging the matrix.

The above five patterns consider the conditions in which \( C(A) \) and \( C(B) \) do not overlap. However, when \( C(A) \) and \( C(B) \) mostly intersect, the situation becomes complex. Since most data points of classes A and B mix together in the feature space, it will be difficult to distinguish the two classes even though the attributes are distinguishable. In this situation, it will be better to improve the features first.

Now we discuss how the machine generates contrastive questions and label recommendations, and how analysts provide feedback to the recommendations through the interaction with the semantic map.

### 4.5 Label Recommendation

Besides providing questions, the semantic navigator highlights the contours of selected classes and provides its attribute label recommendations by coloring class contours using red or blue striped textures. This visual representation mismatches with the analyst’s hypothesis, hence she can provide feedback to the machine with interactions discussed in Sec. 4.6.

Then the machine adjusts the hypothesis correspondingly.

There are many hypotheses given the constraint that some classes are positive and some negative. The machine should provide the most distinguishable attribute hypothesis. Therefore, we leverage the semi-supervised support vector machine (\( S^2VM \)) to do this. Given the training set of labeled data and a working set of unlabeled data, \( S^2VM \) searches for the best labeling of the working set that results in a support vector machine with maximum margin.

However, solving the \( S^2VM \) problem is very challenging for its non-convexity and computational cost [24]. It is still an open problem to scale up \( S^2VM \) for large-scale applications. Using the whole dataset as the training and working set is unrealistic because analysts need instant feedback when generating attribute hypotheses. Considering this, we build \( S^2VM \) using class exemplars rather than the whole datasets as they have already summarized the cluster structure. We solve this problem using quasi-Newton schemes, the state-of-art method namely \( QN-S^2VM \) [26].

For example, suppose the analyst adopts the hint that asks him/her to come up with a new attribute to distinguish classes A and B except for the attributes \( \{ a_1 \} \in \mathbb{Z} \) (see one example in Fig. 3(c)). Note that \( \{ a_2 \} \in \mathbb{Z} \) are the existing attributes that can distinguish classes A and B. Therefore, the question asks for a new attribute to distinguish the two classes except for \( \{ a_1 \} \). Otherwise, the analyst may come up with a re-
We conduct case studies and controlled user studies to evaluate our method.

Analyzing the visual patterns in the semantic map guided by questions, the stopping condition is met. The visualization will be updated as well. A new iteration begins until the model converges.

In this situation, the images are replaced by their names. Once committed, the image names are the nearest neighbors for each unseen class exemplar. We show the images to ease the recall of the semantics of the analyst. These images are the nearest neighbors for each unseen class.

At the same time, this collaboration eases the labeling burden and improves the efficiency of humans.

After generating an attribute hypothesis, the analyst finetunes the attribute specification recommended by the machine. The analyst circles the polar bear and labels it as positive. Once committed, the machine agrees with the analyst and does not change its decision. The result is updated by the machine. The analyst labels the grizzly bear as negative. The machine agrees with the analyst.

Fig. 7. The design process of water attribute: (a) The analyst annotates hamster, bat, mole, and mouse as negative using a lasso to modify the attribute specification recommended by the machine. (b) The machine adjusts the attribute specification. The analyst circles the polar bear and labels it as positive. (c) The result is updated by the machine. The analyst labels the grizzly bear as negative. (d) The machine agrees with the model using the water attribute.

We first demonstrate the effectiveness of the semantic navigator through case studies using the Animals with Attributes2 (AWA2) dataset a standard benchmark for zero-shot classification. AWA2 has 37322 images of 50 kinds of mammals, along with the features extracted from a pre-trained ResNet. We use the standard split of the dataset (40 classes for training and 10 for testing) for the study. Besides, AWA2 provides a class-attribute matrix with 85 binary attributes. This class-attribute matrix is originally collected in a psychological experiment on attribute-based object similarity from human subjects.

Suppose one data scientist Bob wants to build an animal classifier using the AWA2 dataset. He first explores the semantic map to get familiar with the animals. Then he finds that similar animals in the semantic map are close to each other, while different animals are far apart.

Semantic navigator orients the modification process of attributes. Bob checks the hint view and preview a list of recommendations. The machine first asks him what is the most visible attribute between classes otter and horse (Fig. 5). Bob quickly notices that the otter lives in the water, while the horse lives on the ground. Therefore, he decides to add water concept to the empty class-attribute matrix.

The semantic navigator provides an attribute hypothesis shown in the semantic map (Fig. 7(a)). The machine thinks this attribute distinguishes the otter and horse most. However, this attribute does not align with the water concept of what Bob thinks. Therefore, Bob decides to modify the attribute hypothesis. He first circles the bat, mole, hamster, and mouse, and labels these animals as negative (Fig. 7(a)). Then the machine quickly gives feedback to adjust the attribute hypothesis.

The updated attribute hypothesis is shown in Fig. 7(b). Bob notices that the semantic navigator omits the polar bear, as the polar bear usually jumps into the sea to catch fish. Therefore, Bob circles the polar bear and marks it as positive as well (Fig. 7(b)). Then the machine provides feedback and recommends labeling the grizzly bear as positive (Fig. 7(c)). It seems that the machine has a different opinion of the attribute hypothesis compared with that of Bob. However, Bob is quite sure that grizzly bear usually lives in the forest. Therefore, he modifies the hypothesis.

5.1 Case Studies

We conduct case studies using the Animals with Attributes2 (AWA2) dataset a standard benchmark for zero-shot classification. AWA2 has 37322 images of 50 kinds of mammals, along with the features extracted from a pre-trained ResNet. We use the standard split of the dataset (40 classes for training and 10 for testing). AWA2 provides a class-attribute matrix with 85 binary attributes. This class-attribute matrix is originally collected in a psychological experiment on attribute-based object similarity from human subjects.

Suppose one data scientist Bob wants to build an animal classifier using the AWA2 dataset. He first explores the semantic map to get familiar with the animals. Then he finds that similar animals in the semantic map are close to each other, while different animals are far apart.

Semantic navigator orients the modification process of attributes. Bob checks the hint view and preview a list of recommendations. The machine first asks him what is the most visible attribute between classes otter and horse (Fig. 5). Bob quickly notices that the otter lives in the water, while the horse lives on the ground. Therefore, he decides to add water concept to the empty class-attribute matrix.

The semantic navigator provides an attribute hypothesis shown in the semantic map (Fig. 7(a)). The machine thinks this attribute distinguishes the otter and horse most. However, this attribute does not align with the water concept of what Bob thinks. Therefore, Bob decides to modify the attribute hypothesis. He first circles the bat, mole, hamster, and mouse, and labels these animals as negative (Fig. 7(a)). Then the machine quickly gives feedback to adjust the attribute hypothesis.

The updated attribute hypothesis is shown in Fig. 7(b). Bob notices that the semantic navigator omits the polar bear, as the polar bear usually jumps into the sea to catch fish. Therefore, Bob circles the polar bear and marks it as positive as well (Fig. 7(b)). Then the machine provides feedback and recommends labeling the grizzly bear as positive (Fig. 7(c)). It seems that the machine has a different opinion of the attribute hypothesis compared with that of Bob. However, Bob is quite sure that grizzly bear usually lives in the forest. Therefore, he modifies the hypothesis.

© 2021 IEEE. This is the author’s version of the article that has been published in IEEE Transactions on Visualization and Computer Graphics. The final version of this record is available at: xx.xxxx/TVCG.201x.xxxxxxx/
and labels the grizzly bear as negative (Fig. 7(c)). Now, the machine seems to agree with Bob and does not change the hypothesis anymore (Fig. 7(d)). Bob examines the specification closely, then names the hypothesis as water and summit. Then the semantic navigator pops up a window to ask Bob to specify the unseen classes by choosing images with the attribute water. There are ten kinds of unseen animals, including chimpanzee, giant panda, leopard, persian cat, pig, hippopotamus, humpback whale, raccoon, rat, and seal. This task is effortless, and Bob quickly selects the hippopotomus, humpback whale, and seal. Finally, Bob commits the task and retrains the underlying zero-shot model with the water attribute. The final result is shown in Fig. 7(c).

**Semantic navigator directs the design process of the class-attribute matrix step by step.** In the following iterations, Bob is guided by the machine by answering the questions in the hint view. Questions and corresponding answers of the first fifteen attributes specified by Bob are shown in Fig. 8. It seems that it is getting more difficult to answer the questions when the iteration increases. For example, the machine asks Bob what is the most visible attribute between dolphin and killer whale except the white, black, and patches attributes in the final iteration. Nevertheless, with the collaboration between the machine and Bob, semantic prototypes are getting closer to their corresponding class exemplars (Fig. 1(a)). For example, in Fig. 1(a), the semantic trajectory of the sheep presents that the sheep prototype transforms from squirrel and buffalo to sheep and polar bear, and finally gets to the sheep. Most of the semantic prototypes align well with their corresponding class exemplars such as siamese cat, fox, and wolf. Guided by the semantic navigator, Bob creates a better class-attribute matrix with only 15 visual attributes without losing the interpretability. This new class-attribute results in comparable training accuracy (ours: 79.612%, baseline: 89.076%) and higher testing accuracy (ours: 70.966%, baseline: 57.079%) compared with the baseline (EXEM [11] with 85 attributes (see Fig. 1(d)).

### 5.2 User Studies

We conduct user studies to evaluate the semantic navigator using both AWA2 and a subset of CUB200-2011 [CUB200-2011](http://www.vision.caltech.edu/visipedia/CUB-200-2011.html). CUB200-2011 is a fine-grained bird dataset containing 200 bird species and binary 312 attributes with 6033 images. These attributes are collected from MTurk workers guided by 28 attribute questions based on an online tool for bird species identification. The standard split of this dataset contains 150 training categories and 50 testing categories. We select a subset of the CUB200-2011 for evaluation by randomly selecting 40 training classes and ten testing classes from the original training classes and testing classes, respectively. We name the selected data set as SUB-CUB200 for short. The features for both AWA2 and SUB-CUB200 are 2048-dimensional extracted from the last pooling layer of ResNet [ResNet](http://www.vision.caltech.edu/visipedia/CUB-200-2011.html).

206 participants (denote as P1-P26) in a large university volunteer the studies, including nine undergraduates, thirteen masters, and four Ph.D. students. To compare with our system, we design a system without guidance. This system contains only two views, including a matrix view (Fig. 1(e)) and a line chart (Fig. 1(e)) as the semantic navigator. It enables analysts to append a new column and modify cells to specify a new attribute. Whenever the analyst adds a new attribute, the underlying zero-shot model is trained online and the line chart for accuracy is updated.

We divide the participants into two groups (group A and group B) randomly and evenly. Before all the experiments, participants should fill in a questionnaire to provide personal information. Both groups need to conduct two experiments without and with guidance. Participants in group A build one classifier for AWA2 without guidance, while the other one for SUB-CUB200 with guidance. In contrast, group B reverses the datasets. We require participants to accomplish classification tasks with two different datasets to mitigate the anchoring bias if the same dataset is conducted sequentially. Besides, we conduct the two experiments on separate days in case the participants are exhausted.

Before each experiment, we conduct a workshop with each group to explain some functionality of the system without guidance or the semantic navigator and how to build a zero-shot classification model by describing animals using attributes. We do not explain any details of the underlying zero-shot model as participants can complete the task without understanding technical details. They only need to describe animals objectively with attributes that are visually distinguishable. Participants can explore the system and ask any questions during this session. This session lasts about thirty minutes. After each experiment, participants need to fill in a questionnaire to provide some feedback.

During each experiment, every participant has to design 15 attributes at the completion of the task. It takes each participant about 30-45 minutes to complete the task in each experiment. The whole design process is screen recorded. Notice that we adjust the system’s language to their corresponding native language since the foreign words of animals can influence their cognition. We also allow analysts to name the attributes in their native language. Moreover, we translate the attributes later on. After each experiment, the class-attribute matrices are downloaded locally for further analysis. At the end of the user studies, we interview each participant to get qualitative feedback. We summarize the results and our findings in the following.

#### 5.2.1 Quantitative results of zero-shot classification

We compare the results from three settings by calculating the testing accuracy using several state-of-art zero-shot algorithms, including EXEM [11], ESZSL [55], LisGAN [39], VAE [58], and DEM [74] (Fig. 9). Three settings include the method without guidance, method with guidance using the semantic navigator, and method using random strategy. The random strategy simulates 20 persons by randomly selecting 15 attributes from the class-attribute matrix benchmark. Besides testing accuracy, we also evaluate the results based on the average time
The total time starts when the participant enters each experiment and specifies a new attribute, the test accuracy does not change much and some words to describe animals. Everybody can do it. Nevertheless, it is hard for them to complete the task. Participant P2 said:

In the task of AWA2, besides, participants spend more time completing the task of SUB-CUB200 than that of AWA2 regardless of whether with guidance. Moreover, participants without guidance gain comparable performance as the random baseline. However, we also find noticeable performance gain once analysts apply the semantic navigator. On average, the semantic navigator can increase the test accuracy by 4.96% for AWA2 and 4.13% for the SUB-CUB200. This difference can become notable when participants create over five attributes (Fig. 10).

5.2.2 Qualitative feedback

More than half of the participants without guidance mentioned that it is hard for them to complete the task. Participant P2 said: In the beginning, I think this task is effortless as what I have to do is just to find some words to describe animals. Everybody can do it. Nevertheless, I quickly found that it is not as easy as expected. Most time, when I specify a new attribute, the test accuracy does not change much and even decreases. It makes me crazy as I have no idea what's wrong I have done. Participant P5 agrees with it and commented: At the end of the design process, I think my brain is stuck. I do not know what I have specified before. When I want to add a new attribute, I need to go back to check if I have specified it already. Participant P8 also mentioned: It is very boring to create the matrix. When I come to a new attribute, I have to specify the animals individually to check whether they have the attribute. During the design process, I need to pay much focus and specify the matrix carefully. It exhausted me.

However, most participants (18/26) with guidance thought that the semantic navigator is easy to understand and use. P11 mentioned that the questions helped me a lot. You just to follow its guidance to answer the questions only. P23 commented that sometimes the questions are too hard for me. For example, what is the most visible attribute between the otter and the beaver? I know they are different, but I can not find an appropriate word to distinguish them. But the good thing is that you can skip the hard question and find one that you can answer. P15 liked the design of the semantic map, he said at the beginning I did not understand what the black dots mean. But when you explained it as the destinations as the normal map, I quickly understand what I need to do is to let the yellow dots reach the corresponding destinations. I like it as it provides an overview, and I can understand the current status of the model. Meanwhile, the semantic map can tell me where I should pay attention. One participant P17 acknowledged that the label recommendations are beneficial, he said: Whenever I fix the specification, the machine can recommend the rest. It impressed me. However, sometimes we disagreed with each other so that I need to specify the attributes with several interactions. It would be better if it could read my mind more efficiently.

6 Conclusion, Discussion and Future Work

This work focuses on the fundamental problem of designing the class-attribute matrix for zero-shot classification with the mixed-initiative approach. We propose visual explainable active learning with four actions (ask, explain, recommend, and respond) to promote human-AI teaming. Besides, we design and implement a visual analytics system called semantic navigator for interactive zero-shot classification. To justify our method, we conduct case studies and controlled user studies. Results show that the semantic navigator improves analysts’ efficiency for building zero-shot classifiers compared with the method without guidance.

Although the visual explainable active learning approach targets zero-shot classification in this paper, its concept has generalizability. The machine asks, and human answers using attributes can be viewed as a form of human-machine communication. At the same time, the visualization creates a shared space to bridge the low-level feature space and the high-level semantic space, which may help build the shared mental model between the human and machine. Our work contributes a new perspective on how humans and AI interact and collaborate via visual analytics and may inspire researchers to promote better human-AI teamwork. Future work can go beyond animal classification to real-world problems, such as the human-AI teaming approach to medical diagnosis.

In the future, we can further improve human-AI teaming with more effective human-AI communication and further explanations. Firstly, we start our work by considering only binary attributes. Future work can extend binary attributes to relative attributes. For example, the classifier can capture that animal A is furrier than animal B. Using relative attributes can foster more natural human-AI communication and more effective human feedback. Secondly, we quantify the performance of zero-shot classifiers based on the model accuracy. However, one open question is how to know the classifier learns the attributes accurately rather than other related concepts? Recent explainable artificial intelligence research has shifted attention to quantifying concept-level explanations, which can complement our work.
