Overview-Based Example Selection in End-User Interactive Concept Learning

Saleema Amershi[†], *James Fogarty*[†], *Ashish Kapoor*[‡], *Desney Tan*[‡]

[†]Computer Science & Engineering
DUB Group, University of Washington
Seattle, WA 98195
{ samershi, jfogarty }@cs.washington.edu

[‡]Microsoft Research
One Microsoft Way
Redmond, WA 98052
{ akapoor, desney }@microsoft.com

ABSTRACT

Interaction with large unstructured datasets is difficult because existing approaches, such as keyword search, are not always suited to describing concepts corresponding to the distinctions people want to make within datasets. One possible solution is to allow end-users to train machine learning systems to identify desired concepts, a strategy known as *interactive concept learning*. A fundamental challenge is to design systems that preserve end-user flexibility and control while also guiding them to provide examples that allow the machine learning system to effectively learn the desired concept. This paper presents our design and evaluation of four new overview-based approaches to guiding example selection. We situate our explorations within CueFlik, a system examining end-user interactive concept learning in Web image search. Our evaluation shows our approaches not only guide end-users to select better training examples than the best-performing previous design for this application, but also reduce the impact of not knowing when to stop training the system. We discuss challenges for end-user interactive concept learning systems and identify opportunities for future research on the effective design of such systems.

ACM Classification: H5.2 [Information interfaces and presentation]: User Interfaces – Interaction styles. H1.2 [Models and principles]: User/Machine Systems – Human factors.

Keywords: End-user interactive machine learning.

INTRODUCTION AND MOTIVATION

The current information explosion fundamentally changes how people live and work with computing: the ease of publishing and sharing has made vast numbers of documents and images available on the Web; the ubiquity of sensing-equipped devices enables near-continuous tracking and monitoring of people and objects; and inexpensive storage allows people to keep personal data and sensing archives of practically unlimited size.

UIST'09, October 4–7, 2008, Victoria, BC, Canada.

Although keyword search is the current preferred method of end-user access to large collections of unstructured content, a fundamental premise of keyword search is the availability of keywords corresponding to the important concepts and distinctions that end-users want to make within such datasets. An end-user seeking items with a characteristic that cannot easily be expressed in keywords, or who finds their attempted formulations ineffective, is left to manually sift through large numbers of results.

There are an increasing number of important problems in which this relevant keyword assumption does not hold, necessitating new strategies for effective end-user interaction with large unstructured datasets. We are currently examining the domain of Web image search, where available keywords are generally insufficient for visually characterizing desired images. Visually distinct images may have the same keywords, and visually similar images may have very different keywords. The computer vision community has explored automated identification of many image characteristics (e.g., *indoor* versus *outdoor* scenes [24], *city* versus *landscape* scenes [10], and *photos* versus *graphical* images [20]). Other work has explored end-user browsing of unstructured data using clustering techniques [4, 5]. In both approaches, the application developer predetermines what distinctions an end-user can make among objects (e.g., by training a particular classifier, by specifying a particular distance metric for use in clustering). Our perspective is that the current challenge is *not* a lack of methods for making such distinctions among classes of objects, but instead that application developers cannot possibly foresee the countless variety of distinctions end-users might want to make within large datasets.

One strategy for enabling effective access to and interaction with large unstructured datasets is thus to allow end-users to interactively train a machine learning system to identify the concepts that are important to them, a strategy known as *end-user interactive concept learning*. A fundamental challenge in end-user interactive concept learning lies in designing for effective end-user interaction with a system as it learns and refines its definition of a concept. On one hand, a traditional active learning approach can meet the needs of a machine learning system by forcing labeling of high-value training examples (i.e., examples for which the system currently has low certainty). However, such an approach creates a frustrating experience that treats an

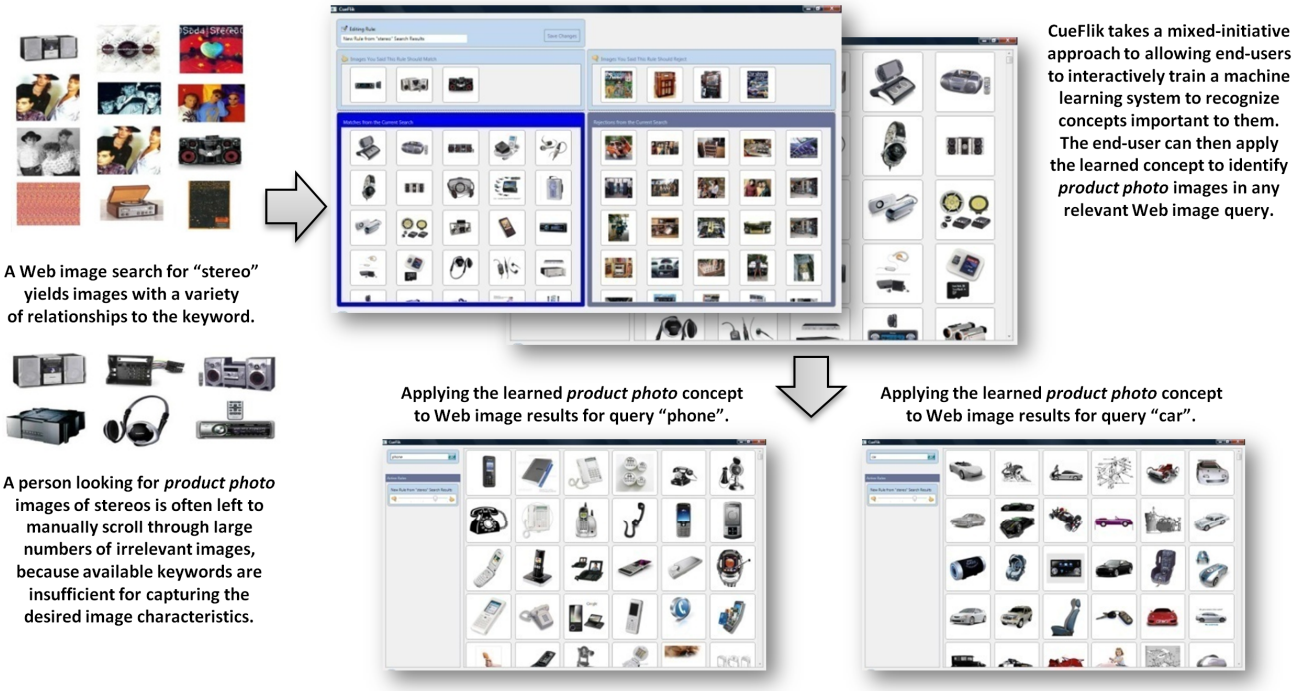

Figure 1: We examine the problem of selecting and presenting sets of examples to improve end-user ability to effectively train an interactive concept learning system. Specifically, we focus on providing overviews by selecting representative images to illustrate the set of images that match or are rejected by an interactively learned concept.

end-user simply as an information oracle. On the other hand, a design that neglects the needs of the machine learning system in favor of end-user flexibility may be equally frustrating if an end-user cannot effectively train the system to recognize a desired concept.

We take a mixed-initiative approach, examining strategies that implicitly assist an end-user in effectively steering a machine learning system while preserving the end-user's flexibility and sense of control. In contrast to a focus on developing machine learning algorithms, we aim to address the important question of how we should actually *interact* with interactive machine learning systems.

This paper is grounded in CueFlik, a system illustrated in Figure 1 and developed to support end-user interactive concept learning in Web image search [8]. As an example, a keyword search for "stereo" yields images with a variety of relationships to the keyword. A developer of a Web image search service cannot reasonably be expected to have included a filter for *product photo* images or the countless other distinctions end-users might desire, so CueFlik allows an end-user to interactively define a *product photo* concept by providing examples of images with and without the desired characteristics. As they do this, CueFlik and the end-user engage in an iterative process of refining the concept definition. CueFlik presents examples that illustrate its current understanding of the concept, and the end-user uses this presentation to decide which additional examples to provide to further refine the concept. The resulting concept can be re-applied to identify *product photo* images not only for the "stereo" query, but also for other relevant queries (e.g., "phone" and "car").

Central to this mixed-initiative approach is the question of *how to select and present sets of examples illustrating the current concept so as to improve end-user ability to effectively train an interactive concept learning system*. Specifically, our contributions are:

- We motivate the study of selecting representative sets of examples that provide an overview of the currently learned concept as a strategy for presenting end-users with the information needed to effectively train interactive concept learning systems.
- We examine four approaches to presenting overviews of the currently learned concept. These are based in *global* overviews that select examples most representative of an entire high-dimensional space, *projected* overviews that identify principal dimensions within a space and then select examples to illustrate data variation along those dimensions, and the use of *neighbors* to better illustrate portions of a space.
- We evaluate our overview-based methods in CueFlik, comparing to the best-performing design from previous studies (which emphasizes high-certainty examples) [8], showing that our new overviews lead end-users to select better training examples throughout their process of training CueFlik to recognize a concept.
- Based on our analyses of our evaluation in CueFlik, we present design challenges for end-user interactive concept learning. By articulating these issues, we provide an important foundation for additional research on the effective design of interactive concept learning systems.

RELATED WORK

The work most closely related to ours is that examining the interactive training of machine learning systems. Fails and Olsen's Crayons system supports interactive training of pixel classifiers for image segmentation in camera-based applications [7]. In their *a CAPpella* system, Dey *et al.* take a demonstration approach to end-user configuration of context-aware applications [6]. Hartmann *et al.*'s Exemplar uses both demonstration and direct manipulation of a dynamic time warping algorithm to support sensor-based recognition [13]. Ritter and Basu examine interactive machine learning in file selection [19]. Although our work is situated in a different domain, these and other systems share our motivation of enabling new applications through more effective interactive training of machine learning systems. The overview-based approaches we develop here, as well as the insights we gain from their evaluation, warrant further examination in these and other domains.

Many researchers have explored mixed-initiative interfaces and interactions in a variety of problems [15]. These include end-user problems like the correction of handwriting recognition errors [21], the correction of information extraction systems [14, 17], and interface customization [3]. This work is generally based in applications carefully tuned according to the application developer's insight into the problem being solved by the end-user. Similarly, we are distinct from prior work examining browsing based on interactive clustering techniques [4, 5]. Such work generally employs a distance metric selected and tuned by the application developer according to their knowledge of the problem. In contrast, our focus on applications where the developer cannot know what distinctions will be important to an end-user magnifies the importance of end-user methods for understanding and manipulating machine learning. Moreover, while prior work has examined model training and feature engineering by developers and other users with deep insight into machine learning algorithms and techniques, alternative methods are needed when end-users are the target audience.

Other prior work has examined interactive machine learning from a machine-centric perspective, focusing on questions related to the underlying learning algorithms. Although traditional active learning systems assume a person is an infallible information oracle, people often have difficulty providing labels out of context [1, 12]. Prior work has thus examined algorithmic compensation for incorrect or incomplete human input [2, 9]. Others have explored alternative strategies for providing examples, including presenting sets of examples [11] or presenting examples from multiple distinct views [18]. Algorithmic work often focuses on minimizing the number of labels required for training, and so evaluations are often conducted offline using fully-labeled datasets in lieu of actual human experiments. In contrast, our focus is on examining all aspects of the problem of how people actually *interact* with interactive machine learning.

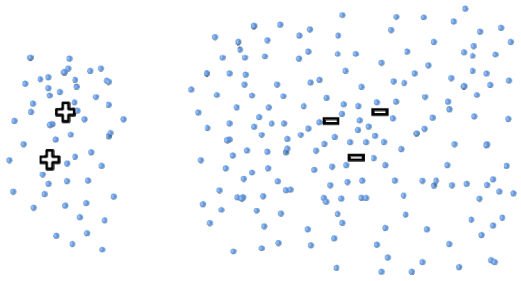

Figure 2: CueFlik learns by finding a distance metric that collapses positive and negative examples together while pushing apart the two classes. The learned metric is then used to rank unlabeled images by their proximity to positive examples.

CUEFLIK BACKGROUND

In order to frame our discussion of overview-based presentation methods, this section briefly recaps some of the most important aspects of prior work on CueFlik [8].

Learning Implementation

Each CueFlik concept is defined as a nearest-neighbor classifier. CueFlik re-ranks the images from a keyword query according to their likelihood of membership in the positive class, defined as the distance to the nearest positive example divided by the sum of the distance to the nearest positive and negative example. Note this requires a method for determining the distance between two images. In most nearest-neighbor applications, this distance metric is carefully tuned by an application developer based on their knowledge of the problem. In our case, however, we cannot know what notion of similarity will be appropriate for an end-user's desired concept. CueFlik's concept learning is therefore formulated as a metric learning problem, wherein the system uses the provided examples to learn a distance metric as a weighted sum of component distance metrics. CueFlik currently includes component metrics based on histograms of pixel hue, saturation, and luminosity, an edge histogram, a global shape histogram, and a texture histogram. Formally, CueFlik minimizes the function:

$$f(weights) = \sum_{i,j \in Pos} D(i,j) + \sum_{i,j \in Neg} D(i,j) + \sum_{i \in All} \ln \sum_{j \in All} e^{-D(i,j)}$$

where $D(i, j)$ is the distance metric computed as a weighted sum of CueFlik's component metrics. The first two terms correspond to within-class distances and favor minimizing distance between examples of the same class. The third considers all examples and favors maximum separation. The combination thus favors weights collapsing each class while maximizing distance between classes. Figure 2 illustrates a hypothetical space learned from two positive examples and three negative examples.

Evaluation of Split Presentation

Our current overview-based methods are motivated in part by a prior study comparing two methods for illustrating the current version of a learned CueFlik concept: a *single* panel method and a *split* panel method [8]. The *single* panel method provided access to the entire set of images, ranked

by their likelihood of membership in the positive class. The *split* method instead showed only a small set of images split across two panels, a best match panel and a worst match panel. The best match panel showed a small number of high-certainty positive images (extremely near positive training examples). The worst match panel showed a small number of high-certainty negative images (extremely near negative training examples).

The study found participants using the *split* presentation created CueFlik concepts of significantly higher quality, using significantly fewer training examples, in significantly less time than participants using the *single* method. This prior study also found no effect of presenting participants with low-certainty examples identified using active learning heuristics, further motivating our interest in mixed-initiative strategies for better guiding end-users to effectively train interactive machine learning systems.

OVERVIEW-BASED METHODS

We believe the current state of the learned concept (i.e., the current learned distance metric) is a critical piece of information that an end-user needs to understand and manipulate to effectively train a desired concept. Although the study reviewed in the previous section showed that the split presentation led participants to better performance, the results mix two possible explanations for the improvement: (1) the use of a split presentation and a much smaller number of examples to illustrate the positive and negative regions during interactive refinement of a learned concept, and (2) that those examples were selected as representative of the positive and negative regions because of their *high-certainty*. We hypothesize that the first of these explanations is indeed important, and that there are likely more effective strategies for selecting small numbers of examples to represent the positive and negative regions.

This paper therefore examines two strategies for selecting small sets of examples to provide an overview of positive and negative regions of a space defined by a learned concept. Both require the separation of the positive region from the negative region, a problem we address using semi-supervised classification. Our first strategy then presents a *global* overview, selecting instances that provide an end-user with the most information about the full set of instances in the positive and negative regions. Our second strategy instead emphasizes *projected* overviews, selecting instances that illustrate variation along major dimensions of the positive and negative regions. We pair both with the use of *neighbors* intended to better illustrate the important aspects of selected instances. This section presents both the intuition behind our strategies and their implementations. Rapid system response to interactive guidance is critical to interactive machine learning systems [7], and our implementations demonstrate techniques to obtain these overviews quickly enough to maintain responsiveness.

Semi-Supervised Classification

Because both of our strategies use split presentations, the first step for both is to separate the unlabeled instances into positive and negative regions. A naïve approach would be to apply a nearest neighbor classifier with the current learned distance metric. The limitations of this can be seen in Figure 2, where some of the unlabeled instances are clearly in the negative cluster but are also closer to a labeled positive example than they are to a labeled negative example (those on the edge nearest the middle). Because we are focused on providing overviews of the positive and negative regions, it is important we preserve such relationships among unlabeled instances.

We do this by employing a semi-supervised classification method that exploits the distribution of both labeled and unlabeled data [25]. Motivated by graph-based methods, we extend nearest neighbor methods to use the underlying data distribution. We create a *k-nearest-neighbor* graph in which each instance is a vertex and edges are created for the k nearest neighbors of each instance (we informally found good performance for many values of k, and arbitrarily set k to 10 in our evaluation). The weight of each such edge is the distance between the corresponding instances according to the current learned distance metric. We then use the graph to compute geodesic distance, the shortest weighted path in the k-nearest-neighbor graph, between labeled instances and each unlabeled instance. The positive region is defined as all instances with a geodesic distance to a labeled positive training example that is less than their geodesic distance to a labeled negative example, and all other instances are included in the negative region. Because geodesic distance considers both labeled and unlabeled data, the resulting separation represents the underlying distribution more reliably than approaches that consider only labeled training examples.

Global Overview Method

Our first strategy aims to provide a *global* overview, selecting a representative set to provide good coverage of the positive and negative regions. Figure 3 presents the intuition behind this, showing red crosses illustrating eight instances selected to provide a *global* overview of Figure 2's negative region (the green triangles are neighbors to these red crosses, as discussed in a later section). The same process is applied separately to create overviews for the positive and negative regions.

More specifically, we approach the problem of creating a *global* overview as a sensor-placement problem [16]. Each instance can be thought of as a possible sensor location where a probe could be placed to sense how well the concept being trained fits the surrounding space. Given a sensor budget (the number of instances we want to use to provide an overview), we choose a set that provides maximum information about the concept the end-user is attempting to train. Intuitively, our approach builds upon the observation that instances that are close to each other can be expected to share similar properties; thus our selection scheme utilizes the principle of maximizing the information theoretic criteria of mutual information. In particular, our aim is to select a subset of instances that

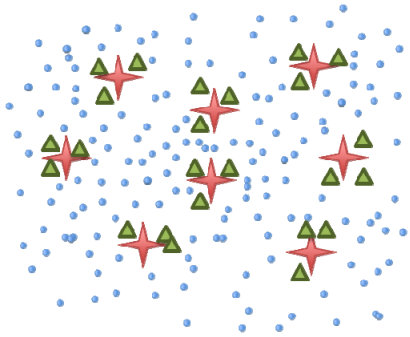

Figure 3: Our *Global* overview considers the entire high-dimensional space and selects examples that are most representative of the full set.

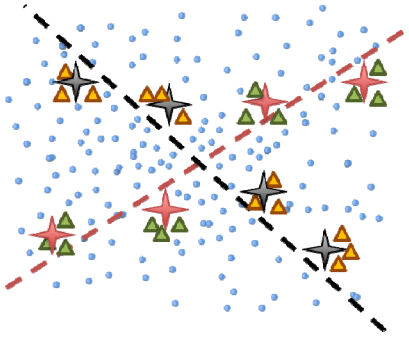

Figure 4: Our *Projected* overview identifies principal dimensions in the space and selects examples to illustrate data variation along those dimensions.

share the highest mutual information with the rest of the high-dimensional space and is therefore most representative of the full set. However, selecting such a set is NP-complete, and a greedy and myopic selection procedure maximizing mutual information gain is the best possible approximation [16]. We therefore employ this approach.

Formally, we consider a Gaussian Process perspective with covariance function (alternatively, kernel function):

$$k_{ij} = e^{\frac{-G(i,j)^2}{Mean(G^2)}}$$

where $G(i, j)$ is geodesic distance in a k-nearest-neighbor graph based on the current learned distance metric, and G^2 is the square of each element in G. Because of the negative exponent, k_{ij} measures similarity between instances i and j and ranges from 0 (infinitely far apart) to 1 (at the same location). The matrix K therefore encodes similarity and how well information flows in the space. Including i in an overview provides a good representation of instances where k_{ij} is high, but is unrepresentative of the portion of the space where k_{ij} approaches zero.

At each step in the greedy selection, given the existing set of selected instances S and unselected instances U, we select the instance that maximizes the gain in mutual information on the remainder of the unselected instances:

$$MI(U-i; S \cup i) - MI(U-i; S)$$

For Gaussian Process models, this can be achieved by selecting the instance i that maximizes:

$$f(i) = \frac{1 - K_{i,S} K_{S,S}^{-1} K_{S,i}}{1 - K_{i,U-i} K_{U-i,U-i}^{-1} K_{U-i,i}}$$

where $K_{S,S}$ is the similarity matrix among S, $K_{U\text{-}i,U\text{-}i}$ is the similarity matrix among U excluding i, and $K_{i,S}$, $K_{S,i}$, $K_{i,U\text{-}i}$, and $K_{U\text{-}i,i}$ are each similarity vectors between i and the respective sets. The numerator characterizes the similarity of i to the previously selected instances and the denominator characterizes the similarity of i to other currently unselected instances. Choosing i that maximizes the ratio selects an instance that is farthest from previously selected instances (most dissimilar to those previously selected) but also central to the unselected instances (most representative of those currently unselected).

Projected Overview Method

Our second strategy presents instances selected after they have been *projected* onto a set of principle dimensions corresponding to the major axes of variation in a dataset. Figure 4 illustrates this, showing two principle dimensions with four instances selected for each (note the total number of representative instances is the same as Figure 3). The approach is similar to Principal Component Analysis or Multi-Dimensional Scaling, except that we want a method that corresponds to the structure of the underlying data (for much the same reasons that we motivated semi-supervised classification). We therefore apply an ISOMAP-based non-linear projection technique [22], which selects principle dimensions using geodesic distance in a k-nearest-neighbor graph. The process of selecting principal dimensions is identical to that used in Multi-Dimensional Scaling, except using geodesic distance.

After obtaining a set of principal dimensions, it might seem that we could apply the same sensor-placement strategy from the previous section to each one-dimensional space to choose instances that illustrate that principal dimension. The problem with such an approach, however, is that it leaves selection completely unconstrained in the other dimensions. Although we are trying to select images instances to illustrate variation along a single principal dimension, images instances selected in this way may do a poor job of conveying the intended variation because they may also vary in many other dimensions. We therefore derive a new scheme to select instances that provide coverage of a single principle dimension but also vary as little as possible in all of the other principal dimensions. We find i maximizing:

$$f(i) = \left(\frac{1 - K_{i,S} K_{S,S}^{-1} K_{S,i}}{1 - K_{i,U-i} K_{U-i,U-i}^{-1} K_{U-i,i}} \right) \left(\frac{1}{1 - \overline{K}_{i,S} \overline{K}_{SS}^{-1} \overline{K}_{S,i}} \right)$$

where K is the similarity matrix for the principal dimension for which we are currently selecting a set of representative examples and $\overline{K}$ is the similarity matrix for all of the other principal dimensions. The left term again prefers instances

farthest from previously selected instances but also central to unselected instances in the current principal dimension. The right term prefers selected instances that are similar to each other in all other dimensions. Maximizing the product selects instances that span a particular dimension and are similar to each other in all other dimensions. In Figure 4, this additional term is responsible for the roughly collinear set of selected instances for each principal dimension.

Using Neighbors to Illustrate Commonalities

There is no shared language between an end-user and CueFlik. For example, CueFlik does not have a notion of a "fuzzy" image and end-users do not conceive of images in terms of the distribution of their edge histograms. Hence, there is no guarantee that an end-user and CueFlik will agree regarding how an image selected for an overview should be interpreted in the context of that overview. As one potential approach to this challenge, we examine the presentation of selected instances together with their nearest *neighbors* (see the green and orange triangles in Figure 3 and Figure 4). The intuition is that presenting a group of similar instances will allow an end-user to recognize what they have in common and thus better interpret an overview. This also introduces the possibility of more quickly training CueFlik by labeling entire groups of similar images as positive or negative in a single action. Our evaluation examines neighbors with both our *global* and *projected* methods.

Downsampling for Performance

Both overviews require inversion of similarity matrix K, an expensive operation. To maintain interactive speeds, we randomly downsample the number of instances in the positive and negative regions before creating overviews. The success of overviews in our evaluation shows this is not problematic, but it is likely important that this be random to preserve the original distribution. Parallel matrix inversion algorithms can leverage the increasing number of cores in a typical computer, and the degree of downsampling could be automatically adjusted to maintain responsiveness on a particular computer. Our evaluation fixed downsampling to 200 images in each of the positive and negative regions. In conditions that display neighbors, the neighbors are taken from the original full set of images.

EXAMINING OVERVIEW-BASED METHODS

We conducted a within-subjects experiment examining overview-based methods for end-user interactive concept learning in CueFlik. The experiment was a 2 (*Global* vs. *Projected*) × 2 (*NoNeighbors* vs. *WithNeighbors*) design with an additional *HighCertainty* baseline condition.

Interface Conditions

We tested a total of five interfaces in our experiment, intentionally holding constant the number of images displayed so results were not confounded by this factor:

Baseline

HighCertainty. Presented the 42 highest-certainty images from both the positive and negative regions. This strategy yielded the best results in prior work [8].

Figure 5: Portions of the interfaces used in each study condition (with images from query "stereo").

Overview

GlobalNoNeighbors. Selected 42 images from both the positive and negative regions using our global method.

GlobalWithNeighbors. Selected 6 images from both the positive and negative regions using our global method, and presented each of them together with 6 neighbors.

ProjectedNoNeighbors. Projected both the positive and negative regions onto 6 principal dimensions accounting for the greatest variance, and selected 7 images to illustrate the variation in each principle dimension.

ProjectedWithNeighbors. Projected both the positive and negative regions onto 3 principal dimensions accounting for the greatest variance, selected 7 images to illustrate the variation in each principle dimension, and presented each together with a single neighbor.

The *HighCertainty* and *Global* interfaces sorted images (and groups of images) in order of their certainty. The *Projected* interfaces sorted principal dimensions by the amount of variance explained by the dimension and then sorted images within each dimension into a spectrum.

Tasks

Participants trained two concepts per interface condition, using queries and corresponding concepts developed for the previous CueFlik evaluation [8]. Each task consisted of approximately 1000 images corresponding to a query (e.g., "drink", "sky") and a set of ten target images printed on a sheet of paper and labeled with a target concept (e.g., "pictures with products on a white background", "pictures with quiet scenery"). Participants were asked to create a concept such that the target images would be highly ranked if they were actually contained in the image set. Target images were originally well distributed across the ranking of the result set so that improvement in their ranking could be used as a measure of the quality of the trained concept. Importantly, the target images were filtered from the result set so that participants could not see how their current learned concept ranked the targets.

Design and Procedure

We ran participants in pairs, with each working on an identical 3.16 GHz quad-core Dell T5400, with 8 GB of RAM and 512 MB Video, running Windows XP, and using 24" Dell monitors at a resolution of 1920×1200. Before beginning, the experimenter demonstrated how to use CueFlik to build a simple rule favoring images with a vertical aspect ratio. We used the *GlobalWithNeighbors* condition for the demonstration so that we could explain that some interfaces would group images and that participants could guide the system by selecting individual images, ctrl-selecting multiple images, or selecting grouped images. Participants then practiced training a simple concept (images of maps within a query for "Seattle") using the first interface condition that they were to interact with during the main portion of the study.

Because we did not expect queries would lead directly to carryover effects, and because we wanted to ensure balanced coupling of interface conditions with queries, we fixed the order of queries in the experiment. We manually categorized target concepts as easy or difficult and then pseudo-randomly selected queries such that the first task in each interface condition would be an easy concept and the second task would be a difficult concept. The ordered query pairs were: "character" (*clipart*) and "sky" (*quiet scenery*), "drink" (*product photo*) and "bill" (*portrait*), "car" (*product photo*) and "crowded" (*cluttered*), "stereo" (*product photo*) and "disco" (*brightly colored*), and "tennis" (*clipart*) and "sea" (*quiet scenery*).

The order of interface conditions was counterbalanced using a Latin square design. Each participant trained the pair of easy and difficult tasks before proceeding to the next interface. For each task, participants were given a printout containing the target images labeled with the target concept and clicked a button to begin. Participants were told to perform each task as quickly and accurately as possible and that they could click a button on the interface to advance to the next task if they felt their concept was not improving. We also imposed a maximum time limit of 4 minutes (indicated 20 seconds prior by a visual warning), after which the task would self-advance. After each task, a dialog appeared giving the participant's final score on the task (computed as the final mean ranking of the targets in the image set, inverted and converted to a percentage). Participants were then given a new printout and began the next task. All actions were timestamped and logged.

After each interface condition, participants were given a short questionnaire asking for their level of agreement (on a 5-point Likert scale) with five short statements regarding the interface they had just used. The questionnaire also asked the participants to briefly explain what they thought CueFlik had been doing each time they gave it more training examples, asked what they thought would help improve the interface they had just used, and asked for any other comments. A different short questionnaire at the end of the experiment asked for overall aspects of CueFlik that participants liked, disliked, or would like improved.

Participants

Twenty people (ten female, ages 20 to 48) volunteered. None were colorblind, and all had 20/20 or corrected to 20/20 vision. We attempted to recruit a balance of image search novices (performing no more than one image search every week) and image search experts (performing more than five searches weekly), obtaining seven self-reported novices and thirteen self-reported experts. The experiment lasted approximately 90 minutes, and participants were given a software gratuity for their time.

RESULTS

We analyze our results using four dependent measures. Consistent with prior work [8], we define *Score* as the mean ranking of the target images by a learned concept. A lower score indicates a higher-quality concept that ranks targets closer to the top of the query results. We define *Time* starting from the button click that began each task. Because participants could label multiple images per action (e.g., by dragging grouped images or ctrl-selecting multiple images or groups), we define *NumImages* and *NumActions* as the number of images or actions used to train a concept.

We perform all of our analyses using mixed-model analyses of variance. All of our models include *Participant* and *Query* as random effects. Modeling *Participant* accounts for any variation in individual performance, and modeling *Query* accounts for any difference in the difficulty of queries or their associated concepts as well as any learning, fatigue, or other potential carryover effects between tasks (because our experiment fixed query order). Throughout this section, we report least-squared means obtained from our mixed-model analyses.

HighCertainty versus *Overview*

We first compared our baseline *HighCertainty* condition to all of our *Overview* methods (i.e., aggregating data for our four *Overview* methods). Mixed-model analyses reveal an apparent tradeoff: participants using *Overview* interfaces create concepts with a significantly better final *Score* (178 vs. 223, $F_{1,170} = 8.86$, $p \approx .003$) but spend significantly more *Time* training their final concept (197 sec vs. 176 sec, $F_{1,170} = 7.72$, $p \approx .006$) in comparison to the *HighCertainty* interface. We found no difference in the *NumImages* or *NumActions* for the final concept.

We observed during our experiments that participants often continued providing additional training examples even when they did not seem to be further improving a concept. This obviously adds to the *Time*, *NumImages*, and *NumActions* used to train a final concept, and we believed it could also negatively impact the *Score* of a final concept. We therefore further analyze the point where participants obtained their *Best* learned concept, defined as the first point at which their *Score* for a task was within 5% of the best *Score* they achieved at any point during that task. This metric detects both well-defined peaks as well as the beginning of gradual-sloping plateaus. Figure 6 plots this perspective, showing the average *BestScore* and *BestTime* for each interface and their corresponding *FinalScore* and *FinalTime*. For the sake of readability, Figure 6 plots *Score Improvement*, obtained by subtracting *Score* at each plot point from the initial *Score*. We now analyze these and the *Decay* from *Best* to *Final* (the time, images, and actions spent continuing to train a model after achieving *BestScore* as well as the resulting negative impact on *FinalScore*).

Mixed-model analyses show that *BestScore* is significantly better for *Overview* interfaces than for *HighCertainty* (128 vs. 149, $F_{1,170} = 6.52$, $p \approx .012$) and that participants reach their *BestScore* in the same amount of *Time* (108 sec vs. 120 sec, $F_{1,170} = 1.11$, $p \approx .294$). We also saw a marginal effect on *NumActions*, with *Overview* interfaces requiring marginally less actions to reach the significantly better *BestScore* (6.39 vs. 8.63, $F_{1,170} = 3.89$, $p \approx .050$). We found no difference in *NumImages* to *BestScore*.

Examining the *Decay* from *Best* to *Final*, a mixed-model analysis shows *Overview* interfaces result in marginally less *ScoreDecay* than the *HighCertainty* interface (49 vs. 74, $F_{1,170} = 3.44$, $p \approx .065$), even though the *Overview* interfaces result in both greater *DecayTime* (89 sec vs. 56 sec, $F_{1,170} = 9.92$, $p \approx .002$) and *DecayActions* (8.03 vs. 5.30, $F_{1,170} = 7.49$, $p \approx .007$) than *HighCertainty*. We found no difference in the number of *DecayImages*.

We defer discussion of freeform feedback, but analyses of our post-condition questionnaire Likert scales found no significant differences for *HighCertainty* versus *Overview*.

Examining *OverviewMethod* and *Neighbors*

To examine differences among our *Overview* interfaces, we exclude data from our baseline *HighCertainty* condition and conduct a series of mixed-model analyses with fixed effects *OverviewMethod* (*Global* vs. *Projected)*, *Neighbors* (*NoNeighbors* vs. *WithNeighbors*), and their interaction *OverviewMethod* × *Neighbors*. In all cases, effects include either *OverviewMethod* or *Neighbors*, but not both and not their interaction. We therefore only report the effecting treatment (i.e., removing the non-effecting treatment and the interaction from each analysis).

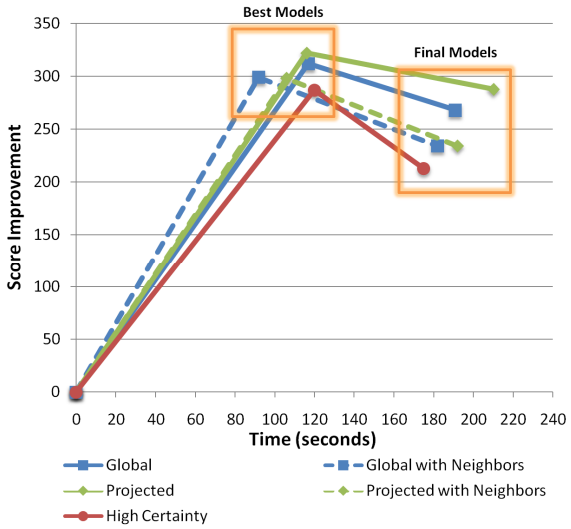

Figure 6: Our *Overview* interfaces guide participants to select better training examples that result in significantly improved *Scores* for their *Best* models and reduce the effect of *Decay* on their *Final* models.

We first examined our *Final* measures, finding interfaces with *NoNeighbors* yielding a significantly better *Final Score* than interfaces *WithNeighbors* (159 vs. 197, $F_{1,131} = 10.7$, $p \approx .001$). We also find that *Global* overviews led participants to spend significantly less *Time* training a *Final* concept than *Projected* overviews (187 sec vs. 206 sec, $F_{1,130} = 10.1$, $p \approx .002$). There were no significant effects on *NumImages* or *NumActions*.

Examining the *Best* concepts shows that interfaces with *NoNeighbors* result in a significantly better *Best Score* than *WithNeighbors* (119 vs. 138, $F_{1,130} = 9.34$, $p \approx .003$). We also find that participants require marginally less time to reach their *BestScore* for interfaces *WithNeighbors* than with *NoNeighbors* (99 sec vs. 117 sec, $F_{1,130} = 3.21$, $p \approx .076$). There were no significant effects on *NumImages* or *NumActions*. This pair of *Neighbors* results is shown in Figure 7, a close-up of the *Best* region from Figure 6. The finding that *NoNeighbors* interfaces yield a better *BestScore* than *WithNeighbors*, together with the finding that *Overview* interfaces yield a better *BestScore* than *HighCertainty*, implies a transitive relationship between *NoNeighbor* interfaces and *HighCertainty*. A mixed-model analysis of variance confirms *NoNeighbors* overviews result in a significantly better *BestScore* than *HighCertainty* (119 vs. 149, $F_{1,91} = 10.67$, $p \approx .002$). Similarly, a mixed-model analysis of variance shows *WithNeighbors* overviews require marginally less *Time* to reach a *BestScore* that is as good as the *HighCertainty BestScore* (99 sec vs. 120 sec, $F_{1,91} = 2.97$, $p \approx .088$).

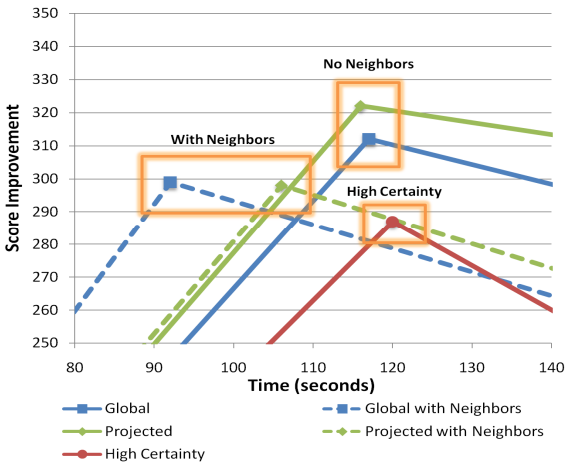

Figure 7: Examining differences among *Overview* interfaces, *NoNeighbors* interfaces guide participants to select training examples that result in significantly better *Best* models, while *WithNeighbors* interfaces result in marginally faster model training.

Examining the *Decay* from *Best* to *Final*, we find only that interfaces *WithNeighbors* resulted in marginally more *DecayActions* than interfaces with *NoNeighbors* (8.79 vs. 7.28, $F_{1,129} = 2.91$, $p \approx .091$). There were no significant effects on *ScoreDecay*, *DecayTime*, or *DecayImages*.

Analyses of our post-condition questionnaire Likert scales showed marginal effects of *Neighbors*. Participants agreed marginally more with "*I felt confused or frustrated when trying to create rules*" for the *WithNeighbors* interfaces than the *NoNeighbors* interfaces (2.5 vs. 2.1, $F_{1,1} = 3.03$, $p \approx .086$) and marginally less with "*I felt that this method was easy to use*" (3.8 vs. 4.1, $F_{1,1} = 2.80$, $p \approx .098$).

DISCUSSION

Our experimental results substantiate our hypothesis that providing an overview of positive and negative regions leads end-users to *select better training examples* when interacting with CueFlik. Using the same amount of *Time*, *NumImages*, and *NumActions*, participants using *Overview* interfaces trained *Best* concepts that were significantly better than those trained using the *HighCertainty* interface, the best-performing interface from prior work [8]. Notably, this prior work shows that the split *HighCertainty* approach performs significantly better than presenting the full image set and found no effect of presenting uncertain images identified using active learning heuristics [8]. Furthermore, although participants using *Overview* interfaces spent significantly more *Time* and *NumActions* in the *Decay* from *Best* to *Final*, their *ScoreDecay* was still marginally less than that of the *HighCertainty* interface. This discussion revisits the impact of *Decay* on all of our interface conditions, but this lower rate of *Decay* per unit of *Time* and *Actions* is further evidence that *Overview* interfaces led participants to *select better training examples*.

Our *WithNeighbors* overviews were intended to provide context in which end-users could better interpret images, as well as the possibility of quickly training a concept by labeling entire groups of similar images in one step. *WithNeighbors* overviews did require marginally less *Time* to obtain a *BestScore* as good as that for *HighCertainty*, but *NoNeighbor* interfaces had significantly better *Scores* for both their *Best* and *Final* concepts. A potential commonality here might be that the *HighCertainty* and *WithNeighbors* interfaces are less effective because they less effectively convey variation with a set of images. In addition to our performance analyses, this interpretation is supported by negative responses to *WithNeighbors* on our questionnaire scales and by participant feedback for these interfaces of the tone "*there were nowhere near enough images – too many were duplicated*" and "*give more options of photos so filtering them will go faster/easier*".

Interestingly, the same emphasis on variation that leads end-users to choose better examples may also make it more difficult to assess how well the current version of a learned concept is performing (because the overview provides more insight into less central portions of a positive or negative region). Participant feedback included many comments of the form "*it was weird, sometimes it would start out doing really well, but as I kept going it did worse*" and "*it is hard to know if more data is better as I should probably stop occasionally to see the results as I am going*". All of the interfaces suffered from some *Decay*: *Overview* interfaces marginally reduced the impact of that decay on concept *Score*, but also had greater *DecayTime* and *DecayActions*. Apart from the finding that *Overview* interfaces had a lower rate of *Decay*, this tension surrounding the effects of emphasizing variation highlights a set of design challenges that warrant further examination in end-user interactive concept learning systems:

Guiding Selection of Effective Training Examples. Prior work has shown the effectiveness of a split presentation of positive and negative regions in CueFlik [8], and we show that overviews of those regions improve end-user selection of effective training examples. Important opportunities remain to examine these results in other applications and to explore how these and other potential strategies interact with the following additional design challenges.

Guiding Selection of Effective Stopping Points. We have identified the importance of helping end-users determine whether to continue providing additional training examples. Interfaces that reduce *DecayTime*, *DecayImages*, and *DecayActions* would positively impact all methods examined in this paper (and further magnify the benefits of our *Overview* methods, given their lower rate of *Decay*). One approach might be to provide lightweight previews or tools for examining multiple alternatives, methods that have proven effective in other domains [23].

Guiding Understanding and Manipulation of Decision Boundaries. It might seem that CueFlik's focus on ranking avoids the matter of a hard confidence threshold, but showing overviews of positive and negative regions requires a boundary between them. Importantly, this challenge likely interacts with the above two. For example,

we are interested in supporting end-user manipulation of how strict a system is with regard to which examples are considered positive. This will affect which examples are included in an overview of the positive region, which might reduce doubt about the current concept quality and allow better choice of a stopping point.

Supporting Effective Strategies. Participants pursued many different strategies in training CueFlik concepts. For example, participants placed varying emphasis on providing positive or negative examples. Some participants always labeled examples one at a time, while others would select a dozen examples before giving them all the same label. There are many opportunities to explore what high-level strategies end-users pursue, which are most effective, and how they can be effectively supported by mixed-initiative systems. For example, if end-users pursue strategies focused on the correction of classification errors (as opposed to a focus on providing canonical examples), a system may be able to gain additional information by analyzing the order in which examples are provided.

These design challenges each have relationships to machine learning problems (e.g., the challenge of selecting an effective stopping point is related to the machine learning problem of identifying overfitting), but a mixed-initiative perspective introduces new opportunities. Our work is based in CueFlik and Web image search, but significant opportunities remain to extend and examine our results in the context of end-user interaction with large unstructured datasets in other domains. By explicitly separating and highlighting these related challenges, we provide a foundation for this additional research on the effective design of end-user interactive machine learning systems.

CONCLUSION

Motivated by the challenges of designing effective interactive concept learning systems, we have examined methods for showing overviews of regions associated with a learned concept. Evaluating these methods in CueFlik, we have shown they guide end-users to select better training examples that result in significantly better *Scores* for their *Best* models and reduce the effect of *Decay* on their *Final* models. These results and our discussion of design challenges for mixed-initiative end-user interactive concept learning provide an important foundation for additional research on the effective design of such systems.

ACKNOWLEDGEMENTS

We would like to thank Morgan Dixon for his assistance creating figures used in this paper. This work was supported in part by the National Science Foundation under grant IIS-0812590.

REFERENCES

1. Baum, E.B. and Lang, K. Query Learning can work Poorly when a Human Oracle is Used. *Proceedings of Neural Networks 1992*.
2. Blum, A., Chalasani, P., Goldman, S.A. and Slonim, D.K. Learning with Unreliable Boundary Queries. *Proceedings of COLT 1995*, pp. 98-107.
3. Bunt, A., Conati, C. and McGrenere, J. Supporting Interface Tailoring Using a Mixed-Initiative Approach. *Proceedings of IUI 2007*, pp. 92-101.
4. Chen, F., Gargi, U., Niles, L., and Schuetze, H. Multi-Modal Browsing of Images in Web Documents. *Proceedings of SPIE Document Recognition and Retrieval VI,* 3651 (1999), pp. 122-133.
5. Cutting, D.R., Karger, D.R., Pedersen, J.O., and Tukey, J.W. Scatter/Gather: A Cluster-Based Approach to Browsing Large Document Collections. *Proceedings of SIGIR 1992*, pp. 318-329.
6. Dey, A.K., Hamid, R., Beckmann, C., Li, I. and Hsu, D. a CAPpella: Programming by Demonstrations of Context-Aware Applications. *Proceedings of CHI 2004*, pp. 33-40.
7. Fails, J.A., Olsen Jr., D.R. Interactive Machine Learning. *Proceedings of IUI 2003*, pp. 39-45.
8. Fogarty, J., Tan. D., Kapoor, A. and Winder, S. CueFlik: Interactive Concept Learning in Image Search. *Proceedings of CHI 2008*, pp. 29-38.
9. Frazier, M., Goldman, S., Mishra, N. and Pitt, L. Learning from a Consistently Ignorant Teacher. *Journal of Computing Systems Science 52*, 3 (1996), 472-492.
10. Gorkani, M.M. and Picard, R.W. Texture Orientation for Sorting Photos 'At a Glance'. *Proceedings of ICPR 1994*, pp. 459-464.
11. Guo, Y. and Schuurmans, D. Discriminative Batch Mode Active Learning. *Proceedings of NIPS 2007*.
12. Gurevich, N., Markovitch, S. and Rivlin, E. Active Learning with Near Misses. *Proceedings of AAAI 2006*, pp. 362-367.
13. Hartmann, B., Abdulla, L., Mittal, M. and Klemmer, S.R. Authoring Sensor-Based Interactions by Demonstration with Direct Manipulation and Pattern Recognition. *Proceedings of CHI 2007*, pp. 145-154.
14. Hoffman, R., Amershi, S., Patel, K., Wu, F., Fogarty, J. and Weld, D.S. Amplifying Community Content Creation with Mixed-Initiative Information Extraction. *Proceedings of CHI 2009*, pp. 1849-1858.
15. Horvitz, E. Principles of Mixed-Initiative User Interfaces. *Proceedings of CHI 1999*, pp. 159-166.
16. Krause, A., Singh, A. and Guestrin, C. Near-optimal Sensor Placements in Gaussian Processes: Theory, Efficient Algorithms and Empirical Studies. *Journal of Machine Learning Research 9*, (2008), 235-284.
17. Kristjannson, T., Culotta, A., Viola, P. and McCallum, A. Interactive Information Extraction with Constrained Conditional Random Fields. *Proceedings of AAAI 2004*, pp. 412-418.
18. Muslea, I., Minton, S. and Knoblock, C.A. Active Learning with Multiple Views. *Journal of AI Research 27*, (2006), 203-233.
19. Ritter, A. and Basu, S. Learning to Generalize for Complex Selection Tasks. *Proceedings of IUI 2009*, pp. 167-176.
20. Schettini, R., Ciocca, G., Valsasna, A., Brambilla, C. and De Ponti, M. A Hierarchical Classification Strategy for Digital Documents. *Pattern Recognition 35*, 8 (2002), 1759-1769.
21. Shilman, M., Tan, D.S. and Simar, P. CueTIP: A Mixed-Initiative Interface for Correcting Handwriting Errors. *Proceedings of UIST 2006*, pp. 323-332.
22. Tenenbaum, J.B., de Silva, V. and Langford, J.C. A Global Geometric Framework for Nonlinear Dimensionality Reduction. *Science 290*, 5500 (2000), 2319-2323.
23. Terry, M. and Mynatt, E.D. Side Views: Persistent, On-Demand Previews for Open-Ended Tasks. *Proceedings of UIST 2002*, pp. 71-80.
24. Vailaya, A., Figueiredo, M., Jain, A. and Zhang, H.J. Content-Based Hierarchical Classification of Vacation Images. *Proceedings of ICMCS 1999*, pp. 518-523.
25. Zhu, X. Semi-supervised learning literature survey. Technical Report 1530, Department of *Computer Sciences, Univ*ersity of Wisconsin, Madison, 2005.