CueFlik: Interactive Concept Learning in Image Search

James Fogarty[†], **Desney Tan**[‡], **Ashish Kapoor**[‡], **Simon Winder**[‡]

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

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

ABSTRACT

Web image search is difficult in part because a handful of keywords are generally insufficient for characterizing the visual properties of an image. Popular engines have begun to provide tags based on simple characteristics of images (such as tags for black and white images or images that contain a face), but such approaches are limited by the fact that it is unclear what tags end-users want to be able to use in examining Web image search results. This paper presents CueFlik, a Web image search application that allows end-users to quickly create their own rules for re-ranking images based on their visual characteristics. End-users can then re-rank any future Web image search results according to their rule. In an experiment we present in this paper, end-users quickly create effective rules for such concepts as "product photos", "portraits of people", and "clipart". When asked to conceive of and create their own rules, participants create such rules as "sports action shot" with images from queries for "basketball" and "football". CueFlik represents both a promising new approach to Web image search and an important study in end-user interactive machine learning.

Author Keywords

Web image search, interactive concept learning, CueFlik.

ACM Classification Keywords

H5.2. Information interfaces and presentation: User Interfaces; H1.2. Models and Principles: User/Machine Systems.

INTRODUCTION AND MOTIVATION

Although the number of images available on the Web continues to explode, driven by a combination of technological advances and the development of new uses for that technology, image search on the Web remains a challenging problem. Web image search is difficult in part because a handful of keywords are generally insufficient for characterizing an image. Visually different images can have the same keywords, or visually similar images could be labeled by very different keywords. If a person seeks an image with visual characteristics that cannot be easily expressed in keywords, or if their attempts to use keywords to describe visual characteristics of an image are ineffective, they are generally left to scroll through large numbers of results in search of a desired image.

Widely used Web image search engines have begun to provide tags that can be used to filter results according to certain characteristics. The most common of these can be used to automatically detect and exclude pornographic content from results, but several engines also support queries based on image size (small, medium, or large images), whether an image is in color or in black and white, and whether an image contains a face. The computer vision research community has explored identification of a number of other characteristics of images, such as indoor vs. outdoor scenes [21], city vs. landscape scenes [8], and photos vs. graphics [17]. But it is hard to successfully apply such work to Web image search, in part because it is unclear what concepts will be valuable to Web search users.

In this paper, we present CueFlik, a Web image search application that allows end-users to quickly create their own rules for re-ranking images based on their visual characteristics. To use CueFlik, end-users provide examples of images each rule should match and examples of images the rule should reject. CueFlik learns the common visual characteristics of examples, and the end-user can then re-rank any future Web image search according to the learned concept.

Figure 1 presents an example sequence of interaction with CueFlik surrounding a rule for the concept *Scenic*. An image search for "Mountain" yields reasonable results, but the top results include images that the end-user is not interested in, such as a movie poster and a picture of a dog. They create a new rule by dragging several of the scenic mountain images to the rule panel, then use the rule editing interface to provide positive and negative examples until satisfied with how the rule re-ranks the mountain images. When this rule is then applied, the top results are now all scenic images. Future searches can then be re-ranked using the scenic rule, and examples are shown here for "Water" and for "Scenic" (which includes several photos of a car, the Renault Scénic).

CHI 2008, April 5–10, 2008, Florence, Italy.

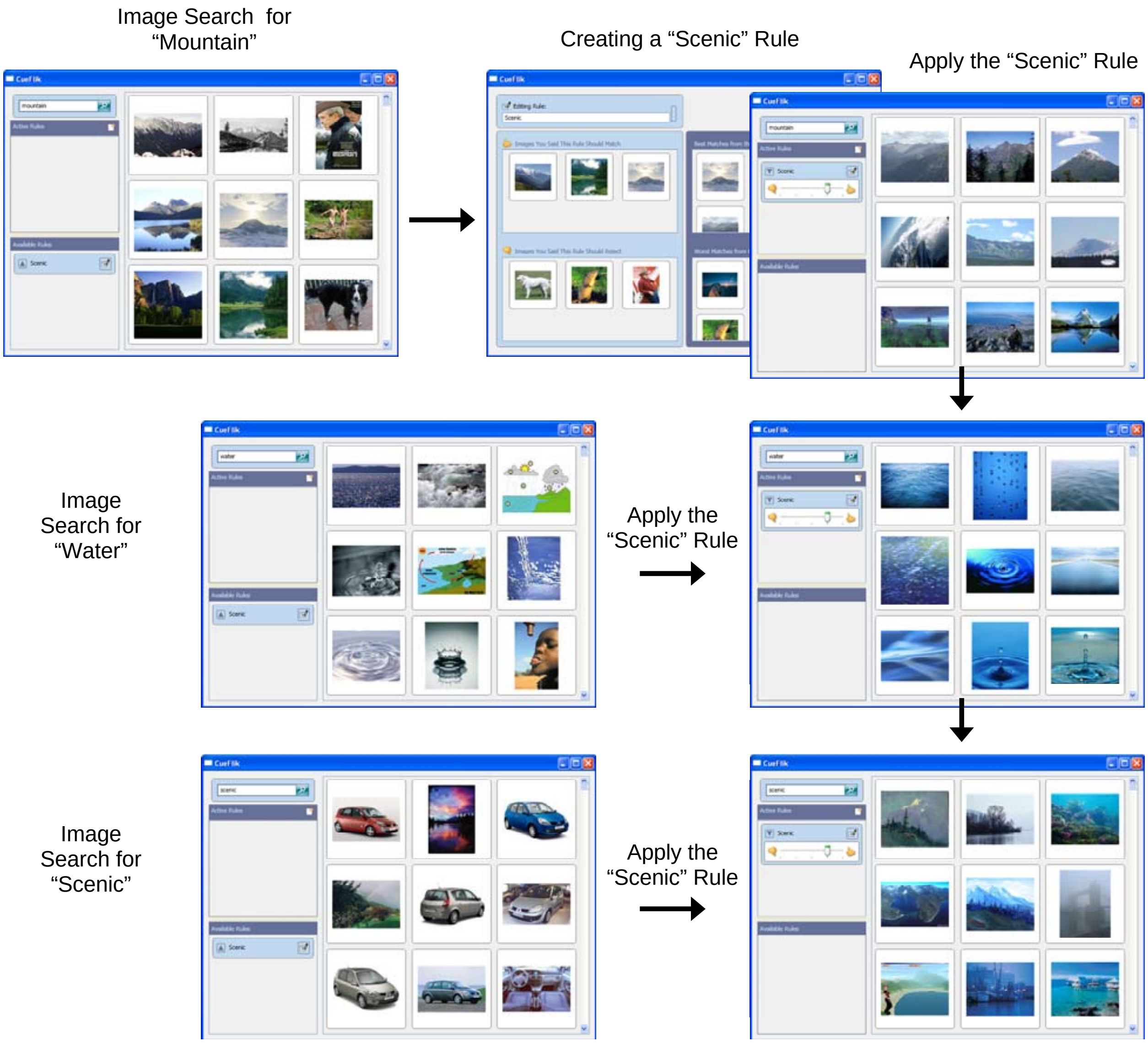

Figure 1. CueFlik enables the interactive creation and application of rules based on the visual properties of images. In this case, a "Scenic" rule is created from a query for "mountain" and then applied to queries "water" and "scenic".

We present an experiment exploring the effectiveness of CueFlik in light of several design decisions that confront such systems. This experiment shows that participants can quickly create effective rules for identifying such concepts as "product photos", "portraits of people", and "clipart". When asked to conceive of and create their own rules, participants create such rules as "sports action shot" with images from queries for "basketball" and "football".

Our experiment also has broader implications for end-user interactive concept learning. An end-user provides CueFlik with examples of images that each rule should match, is shown the rule that CueFlik has learned, and then continues to provide CueFlik with examples of images that should be matched or rejected until they are satisfied with the learned rule. We examine six approaches to presenting this interactive inference process. We find that an interface condition that shows only the best and worst matches (as opposed to showing an entire ranking of a set of matches) leads participants to create better rules, based on fewer examples, in less time. This suggests that such split presentations should be considered in future applications based on interactive concept learning.

The next section briefly discusses some related work. We then present CueFlik's implementation, including details of how CueFlik learns by weighting potential distance metrics. We next discuss our design of an experiment to examine end-user interactive concept learning in CueFlik. The experiment compares six interface conditions to examine their effect on end-user interactive concept learning, then examines how participants re-use rules across queries, how they formulate queries to build rules, and what rules they choose to build on their own. Finally, we discuss CueFlik as a new approach to Web image search and as a case study of end-user interactive concept learning.

RELATED WORK

A number of systems have explored query-by-content approaches to image search [13, 19, 20]. Such systems are based in retrieving images according to low-level features, such as color or texture. A user specifies a desired set of features either abstractly (as in "images with blue in the upper-left corner") or concretely (via an example image), and the system returns images with similar features. While CueFlik also analyses low-level image features, CueFlik is not a query-by-example system. CueFlik learns a rule by analyzing a set of example images, and can then apply that rule to re-rank any future set of images. In contrast, query-by-example generally requires a new demonstration of the desired characteristic with every query.

Other systems have explored approaches to image searching or browsing that are based in clustering or otherwise grouping images according to some criteria [1, 2, 4, 16, 22]. Clustering algorithms are fundamentally based on the specification of a distance metric, as the chosen notion of distance (or, inversely, similarity) controls what images are clustered together. A system's distance metric is generally chosen by the developer, but CueFlik is distinct in that each rule is based in learning a distance metric according to the examples provided by an end-user. This allows CueFlik to learn the relevant visual properties of a user-specified concept.

Yee *et al.* present the use of faceted metadata for image search and browsing [24]. Although CueFlik is currently based in re-ranking query results, our approach to end-user interactive concept learning could likely be applied to faceted search and browsing. The same concerns discussed in our introduction, that it is generally unclear what distinctions between images are best included in Web image search interfaces, also apply to faceted approaches. An interesting possibility, therefore, is supporting end-user interactive specification of facets based on the visual properties of images.

A variety of other work has explored interactive machine learning. For example, Crayons supports the interactive creation of pixel-level classifiers for use in camera-based interfaces [3]. Exemplar supports designer creation of simple sensor-based recognizers through the direct manipulation of a dynamic time warping algorithm [9]. CueTip supports interactive intelligent correction of errors in handwriting recognition [18], while Kristjannson *et al.* examine intelligent interactive correction of an information extraction system [11]. Finally, Arnauld learns an individual user's preferences for how automatically generated interfaces should be presented by examining the tradeoffs and choices that the generation algorithm encounters in creating interfaces for a particular platform [5, 6]. Although this variety of work addresses a wide range of problems using approaches that are very different from the approach taken by CueFlik, all are focused on creating truly usable end-user interfaces that leverage the application of machine learning algorithms and techniques.

CUEFLIK IMPLEMENTATION

CueFlik is currently implemented as a desktop application that retrieves images from a keyword-based Web image search engine. CueFlik enables the re-ranking of image search results according to rules based on visual characteristics of the images. Each rule is defined as a nearest-neighbor classifier, computing a score that indicates how similar an image is to the examples that were used to train that rule. The training of such rules requires learning a distance function from the examples provided by an end-user. In order to help end-users provide informative examples that help CueFlik determine what rule the user is creating, CueFlik implements two active learning criteria. This section discusses each of these aspects of CueFlik.

Image Queries

CueFlik retrieves images using queries to Microsoft's Live Image Search. A format parameter in the query indicates that the engine should return its results list in an XML format, and CueFlik downloads the thumbnail for each image. Due to a limitation of the search service, a maximum of 1000 images are obtained for each query. The visual characteristics of each image are analyzed as they are downloaded, and the resulting images are ranked according to any active rules.

Ranking Image Results

Images are re-ranked by applying a set of end-user created rules. Users can enable and disable rules in their library by moving them between the Active and Available panes of CueFlik's interface (see Figure 1). A slider control on each active rule allows control of the relative weighting of multiple rules. Every active rule computes a score for each image, and scores are multiplied by a weighting between -1 and 1. Images are thus ranked by weighted sum of scores:

$$ImageScore(i) = \sum_{r \in Active\ Rules} weight_r * score_r(i)$$

Applying Concept Rules

Each CueFlik rule is a nearest-neighbor classifier. The rule is defined as a set of positive examples (images illustrating what the rule should match), a set of negative examples (images illustrating what the rule should reject), and a distance metric. Given these, a rule scores a new image by computing the distance between that image and each positive or negative example, then dividing the distance to the nearest positive example by the sum of the distance to the nearest positive and nearest negative example:

$$score_r(i) = 1 - \frac{mindist_P}{mindist_P + mindist_N}$$

Note that $score_r(i)$ ranges between 0 and 1, approaching 1 when i is near a positive example and far from negative examples, having value .5 when i is equally close or far from the nearest positive and negative examples, and approaching 0 when i is near a negative example and far from positive examples. For its distance metric, CueFlik uses a weighted sum of a set of several component distance metrics:

$$Distance(i,j) = \sum_{m \in Metrics} weight_m * distance_m(i,j)$$

Given this approach, the core of CueFlik's ability to re-rank images according to their visual characteristics lies in a set of component distance metrics and CueFlik's ability to learn how to weight those different metrics.

CueFlik's Distance Metrics

CueFlik currently implements image distance metrics based on histograms of the hue, saturation, and luminosity of pixels, an edge histogram, a global shape histogram, and a texture histogram. CueFlik computes and stores these histograms with each image, using them to efficiently compute distances between images.

The hue, saturation, luminosity, and edge histograms are computed over the pixels in each thumbnail image and normalized to account for thumbnails of varying size. Two distance metrics are defined for each histogram. The first is the quadratic distance between two histograms, a measure of histogram distance that accounts for the similarity between different bins in the histogram [13]. In the case of the luminosity histogram, for example, an image entirely of luminance 1.0 is considered more similar to an image entirely of luminance 0.8 than it is to an image entirely of luminance 0.4 (the simpler Euclidean comparison would treat the two images as equally dissimilar from the first, provided the three luminosity values are in different histogram bins). The second metric for each image's histograms is the difference in histogram entropy.

We compute a histogram representing the overall structure of each image by applying a shape descriptor to the entire image [23]. This descriptor sums local image gradients into bins over a log-polar target-shaped region covering the entire image, normalizing the resulting histogram. Similar histograms (using Euclidean distance) correspond to images with similar overall structure, and the descriptor offers a degree of invariance to illumination, translation, scale, and rotation. Less formally, this distance metric will generally indicate that two frontal close-ups of a face are similar and that two driver-side views of a car are similar. It will also generally indicate that a close-up of a face is different from a view of a car. The shape descriptor does not consider color, and so it complements our color histogram metrics.

Finally, we compute a texture histogram that preserves less geometric information than the global shape histogram but allows discrimination between the distribution of structures present in an image without regard to their arrangement. This is a bag-of-words approach [12], and requires sampling a number of patches from the image. For efficiency, we sample on a regular grid of partially overlapping blocks and compute a descriptor for each block [23]. An offline recursive clustering analysis of a large image database is used to learn a set of discriminative textures [14], and a histogram is computed at runtime by resolving each sampled block to a bin based on the identified discriminative textures. Less formally, this metric considers images similar if they contain similar patches. It might consider images of two different city skylines to be similar, while the previously discussed global shape descriptor might consider the two skylines different.

CueFlik's Concept Learning over Distance Metrics

CueFlik learns rules from positive and negative examples [1] of images that rule should match or reject. Given a set of positive examples, there are many concepts a person might be attempting to specify. In most applications of nearest-neighbor algorithms, the developer of a system carefully tunes a distance function based on their knowledge of the problem being solved. In our case, however, we do not know beforehand what notion of similarity will be appropriate for an end-user's rule. If we attempt to treat all distance metrics equally, the curse of dimensionality guarantees a very large number of images will be required in order to specify even the simplest rules [10]. It also seems inappropriate to ask end-users to directly manipulate the weights controlling rule formulation.

CueFlik therefore defines a concept learning problem as a matter of learning a set of weights based on which distance metrics best correspond to the provided examples. CueFlik can learn, for example, whether a set of images are similar because of their color histogram, their global shape descriptor, or a combination of the two. Our method for learning these weights is inspired by the work of Globerson and Roweis [7], and is described next.

Given a set of positive and negative examples, CueFlik learns a set of distance metric weights such that the distance between two images with the same label (positive or negative) is minimized and the distance between two images with different labels is maximized. Specifically, we minimize an objective function that separates the two classes as much as possible while keep examples of the same class close together:

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

The first two terms correspond to within-class distances, thus the minimization of the function favors weights that minimize the distance between data of the same class. The third term considers all examples and favors maximum separation. The combination of terms thus favors weights

[1] In the case where an end-user has provided only positive examples, CueFlik randomly samples negative examples from the current image set under the assumption that, on average, these will serve as reasonable temporary negative examples. These provide a point of comparison against the images the end-user has selected as positive, and they are only used until the end-user provides negative examples.

that collapse each of the classes while maximizing the distance between data with different labels. The function is convex, and the unique global minimum is efficiently found using standard non-linear optimization techniques [15].

Less formally, CueFlik learns what notions of distance are relevant based on the examples that a person has provided. If all of the provided positive examples are mostly yellow, and they have no other common characteristics, and the negative examples are not mostly yellow, CueFlik will learn that hue histogram similarity is the relevant distance (giving it a large weight and other distance metrics small weights). The resulting rule will give high scores to images with hue histograms similar to those of the positive examples used to train the rule. In a situation where the positive examples have multiple characteristics in common, those characteristics will each receive some weighting.

Active Learning in CueFlik

Because it might sometimes be difficult to determine what images an end-user should provide as examples in order to help CueFlik learn the correct concept, CueFlik uses active learning to identify images that are likely to provide the most information about the rule a person is creating. We focus here on the heuristics used to identify images for labeling, while the next section examines the effects of different presentations of identified images.

CueFlik's first active learning heuristic identifies images that, given the current set of learned distance weights, are closest to the boundary between positive and negative. These are the images about which CueFlik is most uncertain, so labeling them provides information within the space defined by the current distance weights. Formally, CueFlik selects images with the smallest value:

$$uncertain(i) = abs(mindist_P - mindist_N)$$

This is complemented by a second heuristic that identifies images that will result in the exploration of new weightings of the distance metrics. Although the active selection of examples for a given distance metric (as with our first heuristic) is a well-explored problem, the active selection of examples to inform the learning of a distance metric has not been well explored. For CueFlik, we have designed a heuristic based on data density and uncertainty, selecting images with the smallest value:

$$activedistance(i) = (mindist_P + mindist_N) * uncertain(i)$$

The intuition behind this heuristic is to find images that are in dense portions of the space (near other labeled examples) but are still very uncertain. The first term captures density, as the distance to positive and negative examples will be lower in dense portions of the space. The second term captures uncertainty, as discussed in the previous paragraph. We select images with low scores, and labeling those images gives CueFlik new information to use in finding a weighting of distance metrics that pushes positive and negative examples away from each other.

EXAMINING INTERACTIVE CONCEPT LEARNING

In order to examine end-user interactive concept learning in CueFlik, we designed an experiment exploring whether end-users would create effective rules, how the presentation of images in and the rule editing panel would affect interaction with CueFlik, and how the presentation of images identified by active learning algorithms would affect interaction with CueFlik.

Interface Conditions

In considering the design of CueFlik as an example of an application based on interactive concept learning, we identified two dimensions of the rule editing interface that deserve careful attention. The first, Editing Presentation, is how the interface presents the effect of a rule on the current image query while that rule is being edited. The second, Active Learning Presentation, is how the interface presents the examples identified by our active learning algorithms. Crossing these two dimensions yields the six interface conditions tested in our experiment.

Editing Presentation

As a rule is being edited, CueFlik needs to present what rule has been learned. Presenting the positive and negative examples that form the basis for the rule is straightforward. The learned distance metric weights are illustrated by showing the images in the current query ranked according to the rule being edited. Editing Presentation considers whether CueFlik should show the entire set of images as they are ranked by the rule that is being edited (Single), or show only a small subset of the images, those that rank at the very top and those that rank at the very bottom (Split). The Single approach provides the end-user with access to the entire set of images from the current query, so they have more images to choose from in training the rule. But the rule is unlikely to ever be completely perfect, so a person may become overly focused on the noisy boundary between positive and negative images, continuing to provide training examples that are no longer noticeably improving a rule. The Split approach avoids this possibility, as end-users can provide training examples only until the small subset of images displayed from the top of the ranking match the desired concept and the small subset of images displayed from the bottom of the ranking are examples of images that have been correctly rejected. This comes at the cost that fewer images are available to choose from when providing training examples.

Active Learning Presentation

A similar tradeoff is explored in considering how to present images selected by CueFlik's active learning algorithms. These images, by definition, will exist in the most uncertain regions of the image query space. They may therefore either help a participant to quickly find effective examples, or their presence may lead a participant to continue providing examples even after they are no longer noticeably improving a rule. Active Learning Presentation considers whether CueFlik should place active learning images in a separate pane (Explicit), randomly distribute active learning images near the top and bottom of a set of results in the hope being visually distinct from the nearby images may lead active learning images to be selected for use as a positive or negative example (Embedded), or rely only upon the ranked query results (None).

Interface Condition Descriptions

Crossing these two dimensions yields the six interface conditions tested in our experiment. They are:

Split-Explicit. Uses three scrollpanes. They present the 50 top-ranked results, 10 results selected by CueFlik's active learning algorithms, and the 50 bottom-ranked results.

Split-Embedded. Uses two scrollpanes. The first presents the 50 top-ranked results, as well as 5 randomly-seeded active learning results. The second presents the 50 bottom-ranked results, as well as 5 randomly-seeded active learning results.

Split-None. Uses two scrollpanes. They present the 50 top-ranked results and the 50 bottom-ranked results.

Single-Explicit. Uses two scrollpanes. 10 active learning results are displayed in one scrollpane. The other displays the remainder of the ranked query images.

Single-Embedded. Uses one scrollpane. Seeds the top and bottom of the rankings with active learning results using the same algorithm as Split-Embedded, then displays the entire modified ranking in one scroll pane.

Single-None. Uses one scrollpane. It displays all of the ranked query images.

Tasks

Participants performed four sets of related tasks. Each session started with Rule Creation, where we tested how well participants were able to create rules with CueFlik. The second task was Rule Transfer, where we tested whether rules created by participants worked well when applied to new queries. The third task was Query Creation, where we observed as participants issued their own search queries to create rules we had specified. The final task was Concept Creation, in which participants used CueFlik in a freeform manner, conceiving of and creating a rule of their own choice using their own Web queries.

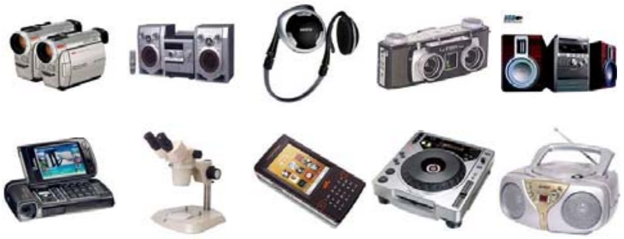

Figure 2. Target images for the concept "product photo" taken from a Web image query "stereo."

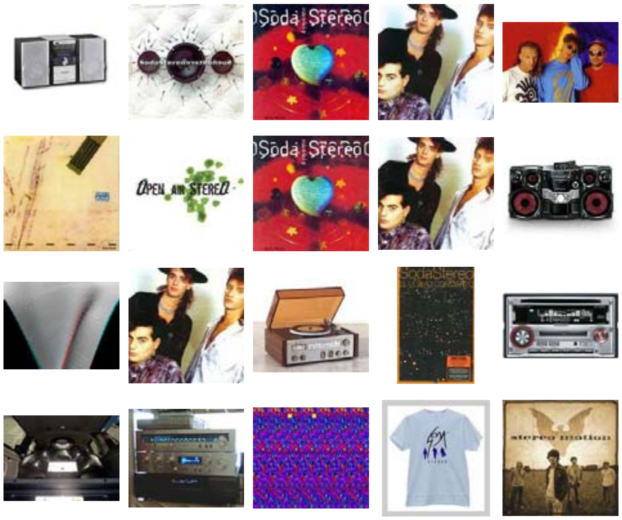

Figure 3. Top 20 results from a Web image query "stereo."

For each trial in the Rule Creation task, participants were given ten target images (printed on a sheet of paper) labeled with a common target property. Figure 2, for example, shows target images for the concept "product photo" in a Web image query for "stereo." The system issued a pre-determined query, and the participant used CueFlik to build a rule that, when applied, would be expected to rank the target images as high as possible. We chose target concepts, search keywords, and target images such that (a) the Web query returned enough results for us to have very near to 1000 images; (b) there were minimal duplicate images within this set of 1000 images; (c) the property of interest applied to between a quarter and a third of the images; and (d) the images matching the property of interest were not already ranked near the top or the bottom of the results. Target images were well distributed across the ranking of the original result set, as we selected one target from each tenth of the images. The target images were filtered from the query results, so participants could not see how their rule ranked the target images. Our final set of concepts included six color-based concepts (red, orange, yellow, green blue, violet) and six non-color concepts (portraits of people, brightly color images, quiet scenery, product images, cluttered images, and clipart). The color concepts were used only in the earliest portion of the experiment, and each was associated with a single query. The non-color concepts were for the bulk of the experiment, and so each was associated with four queries.

The Rule Transfer task was similarly structured around sets of target images for pre-determined queries, with the exception that participants performed sets of three related queries sequentially. The target property was identical across the three queries, but the queries themselves varied significantly. This task was based on the non-color concepts, using keywords participants had not yet seen.

The Query Creation and Concept Creation tasks had users issue their own queries. In the Query creation task, participants created a rule for one of our non-color concepts and were given a small set of stop words they could not use as queries (for example, they could not use the query "clipart" in building the clipart rule). In the Concept Creation task, participants chose their own target concept and issued queries to build the most robust rule they could. We included these tasks to see how users might construct specific rules in the wild, to observe ecologically valid end-to-end usage, and to gather suggestions as to the types of rules end-users might like to construct.

Design and Procedure

We ran participants in pairs, with each participant working on an identical 2.4 GHz dual-core HP4300 machine with a 21" Samsung SyncMaster 214B display. Participants used a Microsoft Optical Intellimouse and Microsoft Ergonomic keyboard for input. Before beginning the experiment, participants received a short tutorial introducing CueFlik. They were informed that some of the interfaces they would see during the experiment would present information slightly differently and told how to interact with each of these. They were led through a small example rule building exercise (a simple rule that favored images with a vertical aspect ratio) and then built a rule on their own (identifying maps within a query for "Seattle"). All participants were able to build a reasonable rule within two minutes.

Participants performed the Rule Creation task using all six interface conditions. The order of conditions was counterbalanced using a Latin square design. They performed two trials with each interface, a color trial followed by a non-color trial. We kept the order of target concepts and queries constant because we did not expect concepts would lead directly to ordering effects and because we wanted to ensure balanced coupling of interface to concept and query. For each trial, participants were given the sheet of paper containing the target images and clicked on a button to begin. Participants were told to perform the task as quickly and accurately as possible, and they clicked a button on the interface to advance to the next trial when they thought their rule was not getting any better. In order to keep the experiment to a reasonable length, we also imposed a 2.5 minute time limit after which the task would self advance. Participants received visual warning 10 seconds before this happened so that they could complete any pending actions. After each trial, a dialog appeared, the participant was given a new page of targets, and they would click on the button to begin the next trial.

Because the remainder of the experiment does not explicitly compare interfaces, each participant completed the final three tasks in a single interface condition (the condition they used in their final Rule Creation trial). Because we had six conditions and there were twelve participants, each interface was used by exactly two participants for the latter three tasks. In the Rule Transfer task, participants created rules for two sets of three related queries. One set was done independently (as in Rule Creation), and one set was done with the rule persisting across the three queries. In the latter condition, the rule persisted and was automatically applied to the result sets for the second and third queries. Participants could choose to accept the rule if it was good enough or could augment it with more examples if they felt they could improve the rule. The task was based on two random non-color concepts that had not yet been used with the current interface condition. The order of the two conditions was counterbalanced across participants.

For the Query Creation task, we chose another random non-color concept that had not yet been used with the current interface condition. During both Query Creation and Concept Creation, we logged the queries that participants issued and took notes on usage behavior. We did not impose the 2.5 minute time limit, and all participants built a reasonable rule within 10 minutes for each of these tasks.

Participants

Twelve individuals from the Greater Puget Sound area volunteered for this experiment. Most were daily computer users, none were colorblind, and all had 20/20 or corrected to 20/20 vision. While recruiting, we screened half of the participants to be image search novices and half to be image search experts. Novices reported performing no more than one image search every week, and experts reported performing more than five searches weekly. The experiment lasted approximately 90 minutes, and participants were given a software gratuity for their time.

One participant experienced multiple software crashes and logs were incomplete for their session. We excluded this participant's data, discarding it at the end of the participant's session, and used a thirteenth participant as a replacement.

RESULTS

We present the results of our experiment in three parts. We first explore the effect of the interface conditions in the Rule Create task. We then examine the Rule Transfer task, comparing the independent creation of rules to the transfer of a previously created rule. We end with qualitative observations based on all of the tasks.

We analyzed all our quantitative data at the summary level, taking the mean of multiple trials when appropriate. Our dependent variables were the number of examples provided to each rule, trial time, and the change in the mean rank of target images. For the latter metric, we calculated the difference between the starting and final rank for each target image for each trial, then took the mean of the differences for the ten target images for each trial. This provides a measure of the quality of the rule based on how it increased the rankings of the participant's target images. As previously noted, target images were filtered from query results so participants could not see how their rule ranked the target images. We also logged how many times the participant switched in and out of edit mode within each

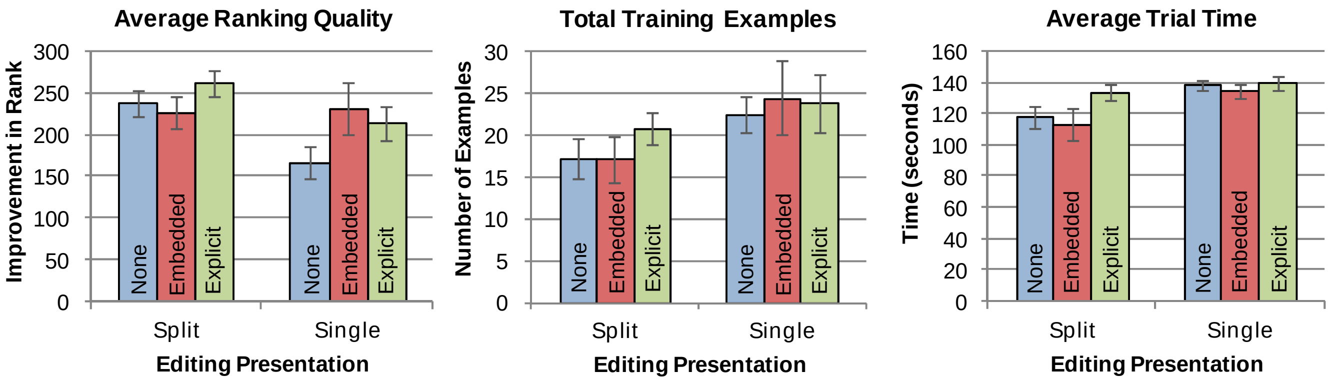

Figure 4. The number of training examples provided, task time, and average improvement in ranking. Participants using a Split Editing Presentation created results of higher quality, using fewer example images, in less time.

trial. We found no effects of gender or expertise, nor any interaction between task and our conditions, so we leave these factors out in our final analyses for simplicity.

Rule Creation Task

We performed a 2 (Editing Presentation: Split vs. Single) x 3 (Active Learning Presentation: Explicit vs. Embedded vs. None) repeated measures analysis of variance (RM-ANOVA) for each of our dependent measures. We found significant effects of Editing Presentation, with Split interface conditions resulting in participants creating rules of higher quality ($F(1,11) = 5.16$, $p \approx .044$) [2], using fewer examples ($F(1,11) = 8.77$, $p \approx .013$), and completing the trial in less time ($F(1,11) = 6.90$, $p \approx .024$) than when using Single condition interfaces. See Figure 4 for plots of our dependent measures. This provides converging evidence that the Split condition of Editing Presentation is more effective for Rule Creation. This performance benefit existed even though participants entered and exited edit mode significantly more in Split conditions ($F(1,11) = 6.04$, $p \approx .024$), perhaps to check the overall quality of a rule as it was applied to all image results (because they could not see the entire set of images results in Split edit conditions).

We also found a main effect of Active Learning Presentation ($F(2,22) = 4.79$, $p \approx .019$), with posthoc tests suggesting that participants took significantly more time in Explicit conditions than Embedded conditions ($p \approx .049$). This is somewhat difficult to interpret, but in the absence of any performance differences, this might be attributed simply to the fairly small size of the scrollpane used to display active learning results, since this led to more scrolling than in other conditions. We saw no other significant effects or interactions.

Rule Transfer Task

We performed paired t-tests for dependent measures in the Rule Transfer task, comparing the Independent Creation condition to the Persistent Rule condition. We found significant main effects, with participants in the Persistent Rule condition providing significantly fewer example images (17.4 images in the Persistent Rule condition vs. 24.2 images in the Independent Creation condition, $t(11) = 2.898$, $p \approx .01$) and spending less time creating their rule (119.0 seconds in the Persistent Rule condition vs. 133.7 seconds in the Independent Creation condition, $t(11) = 3.256$, $p \approx .01$). We found no significant difference in the quality of the resulting rules (an average ranking improvement of 246.1 in the Persistent Rule condition vs. an average ranking improvement of 214.2 in the Independent Creation condition, $t(11) = 1.56$, $p \approx .14$). These results suggest that rules transferred between queries relatively well, and that persistent rules can indeed be more efficient than creating new filters within each query.

Qualitative Results

We now report on participant qualitative experiences, as observed by the experimenter, by examining usage logs, and from participant comments on a post-experiment questionnaire and in a post-experiment debrief.

Participants expressed general satisfaction with the interface itself as well as CueFlik's approach to re-ranking images. Without being prompted, at least five participants explicitly expressed the desire to see an offering like CueFlik in their Web search engines. Three others wanted the software to be able to search through personal picture repositories on their hard drive. All participants were able to learn the interface within the short tutorial and practice trial, each of which ran under 2.5 minutes (because they were implemented using the same timeouts used in the primary task trials). All but two of the participants also commented on how effective the system was in helping them rank the images, making comments like "it seemed very easy to get the pictures I wanted", "it's quick, it's visual", "drag-and-drop was easy", or "it quickly intuits my selections".

[2] As additional validation, we also repeated our analyses using mean average precision, a common information retrieval statistic. The same results reported here were obtained, namely that Split interface conditions resulted in participants creating rules of higher quality ($F(1,11) = 5.53$, $p \approx .038$).

Participants in the Query Creation tasks took varying approaches to formulating queries. Some participants issued sequences of very similar queries, such as "cartoon" followed by "licensed cartoons" and then "cartoon characters". These participants seemed to be searching for more examples that possessed a specific property. This behavior may have been partially prompted by the fact that CueFlik returned only 300 images per query in the Query Creation and Concept Creation tasks (we reduced the number of images returned per query in order to ensure that our current implementation was sufficiently responsive in the face of queries that could not be pre-cached). Other participants issued queries that were relatively diverse, such as "anime" followed by "cartoon" and then "mega man". These participants seemed more concerned with the robustness of the rule and were trying to provide as diverse a set of examples as possible.

When asked to create their own rules in the Concept Creation task, participants chose a number of interesting concepts according to which they wanted to rank images. Example concepts include "sports action shots", "underwater images with fish in them", and "religious iconography". In creating these rules, participants did not seem to distinguish between semantics and visual properties. Although CueFlik currently makes no use of the keywords associated with Web query results and so we would not expect CueFlik to capture semantics, many of these concepts worked surprisingly well. This seems to be because the set of images returned for a given Web query are not homogenously distributed in visual space, and so correlations exists between the visual properties of images and their semantics. Though CueFlik cannot detect the level of sports activity per se, it can identify differences in the visual characteristics of images returned from a Web query for "football" or "basketball". Action shot images might, for example, be close-ups of a person, might contain fairly little of the characteristic colors of the field or court, or might include a large region of solid color corresponding to a player's jersey. Such rules are possible only through a tight coupling of semantic keyword search with CueFlik's visual rules, as current keyword-based engines cannot address such visual concepts and CueFlik does not address semantics. The complementary nature of keyword search and visual rules seems powerful and intriguing, and further exploiting it is a promising direction for future work.

The one situation where we noticed that the lack of distinction between semantics and visual properties was problematic occurred when participants had built a good visual rule, then decided to refine the rule with additional semantic properties. As they provided more examples with related semantics, but explicitly different visual properties, the system would re-rank images in a manner that seemed unpredictable to participants, as they expected that the visual properties would remain fixed and the semantics would only add additional information. Exploring how to express this distinction is another direction for future work.

DISCUSSION

This paper has presented CueFlik, a novel approach to Web image search based in end-user interactive concept learning. CueFlik allows end-users to quickly create rules for re-ranking images according to their visual characteristics. In contrast to query-by-example approaches, CueFlik users can maintain a library of rules that they have developed, applying them to re-rank the results of future Web image searches. While existing clustering-based approaches to image search and browsing are based on pre-determined notions of similarity included by a system's developer, CueFlik rules and their underlying notions of similarity are interactively defined by end-users.

CueFlik explicitly does not attempt to solve computer vision problems related to the semantic recognition of image contents. We instead complement existing keyword-based Web image search functionality with a new approach to re-ranking images according to their visual characteristics. Our approach therefore seems have the best advantages of two approaches, in that (1) semantic image recognition remains a hard research problem, but keywords are available for many images, and (2) the visual characteristics of images can be difficult to describe in keywords, but low-level image features can support example-based approaches to interactive concept learning. Interestingly, our experiment revealed that the results of a Web image search sometimes include correlations between semantics and visual characteristics, so CueFlik may appear to learn a semantic concept even though it is actually learning the visual characteristics of images with those semantics within the particular space defined by the result of a keyword-based Web image search. Promising directions for future work are suggested by this finding and by the possibility of adding keyword-based semantic similarity to the distance metrics considered by CueFlik.

Finally, our examination of strategies for presenting the interactive inference of a concept from examples has important implications for future work addressing end-user interactive machine learning. Interfaces that showed only the best and worst matches for a rule led participants to create better rules, using fewer examples, in less time than interfaces that presented an entire ranking of the current images according to the rule. One explanation for this might be that the Split conditions encouraged participants to focus on whether a rule was mostly correct, stopping when the top and bottom of a ranking corresponded to the desired concept. In contrast, the conditions that presented an entire ranking allowed participants to examine the more uncertain portions of the ranking (the middle images), and may have led participants to find relatively minor inconsistencies in this region of greater uncertainty which then prompted them to continue adding examples to refine their rule. This would explain participants providing more examples and taking more time, and we believe it might also explain the lower quality of the resulting rules. This is because the later training examples (those taken from the more uncertain middle portion of the ranking) may be of lower quality than the initial examples, as something about them has made it difficult for CueFlik to accurately rank them. As more of these types of examples are provided, CueFlik may begin to learn weights that correspond to irrelevant aspects of those images. In the context of such possibilities, our findings suggest careful consideration of how end-user interactive machine learning applications solicit examples and other evidence from end-users, as well as future research on how to determine when an end-user has encountered such a turning point in an interactive machine learning process.

ACKNOWLEDGMENTS

We would like to thank Sumit Basu, Matthew Brown, Mary Czerwinski, Susan Dumais, Eric Horvitz, Gang Hua, Dan Liebling, John Platt, Michael Revow, Ying Shan, Patrice Simard, and Larry Zitnick for both their stimulating discussions and for their support and feedback on this work.

REFERENCES

1. Bederson, B.B. (2001). PhotoMesa: A Zoomable Image Browser Using Quantum Treemaps and Bubblemaps. *Proceedings of the ACM Symposium on User Interface Software and Technology* (UIST 2001), 71-80.
2. Cai, D., He, X., Li, Z., Ma, W.-Y. and Wen, J.-R. (2004). Hierarchical Clustering of WWW Image Search Results Using Visual, Textual, and Link Information. *Proceedings of the ACM Conference on Multimedia* (Multimedia 2004), 952-959.
3. Fails, J.A. and Olsen, D.R. (2003). Interactive Machine Learning. *Proceedings of the International Conference on Intelligent User Interfaces* (IUI 2003), 39-45.
4. Fass, A.M., Bier, E.A. and Adar, E. (2000). PicturePiper: Using a Re-Configurable Pipeline to Find Images on the Web. *Proceedings of the ACM Symposium on User Interface Software and Technology* (UIST 2000), 51-62.
5. Gajos, K. and Weld, D.S. (2004). SUPPLE: Automatically Generating User Interfaces. *Proceedings of the International Conference on Intelligent User Interfaces* (IUI 2004), 93-100.
6. Gajos, K. and Weld, D.S. (2005). Preference Elicitation for Interface Optimization. *Proceedings of the ACM Symposium on User Interface Software and Technology* (UIST 2005), 173-182.
7. Globerson, A. and Roweis, S. (2005). Metric Learning by Collapsing Classes. *Proceedings of the Conference on Neural Information Processing Systems* (NIPS 2005), 451-458.
8. Gorkani, M.M. and Picard, R.W. (1994). Texture Orientation for Sorting Photos 'At a Glance'. *Proceedings of the International Conference on Pattern Recognition* (ICPR 1994), 459-464.
9. Hartmann, B., Abdulla, L., Mittal, M. and Klemmer, S.R. (2007). Authoring Sensor-Based Interactions by Demonstration with Direct Manipulation and Pattern Recognition. *Proceedings of the ACM Conference on Human Factors in Computing Systems* (CHI 2007), 145-154.
10. Hastie, T., Tibshirani, R. and Friedman, J. (2001). *The Elements of Statistical Learning*. Springer.
11. Kristjannson, T., Culotta, A., Viola, P. and McCallum, A. (2004). Interactive Information Extraction with Constrained Conditional Random Fields. *Proceedings of the National Conference on Artificial Intelligence* (AAAI 2004), 412-418.
12. Li, F.-F. and Perona, P. (2005). A Bayesian Hierarchical Model for Learning Natural Scene Categories. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (CVPR 2005), 524-531.
13. Niblack, C.W., Barber, R., Equitz, W., Flickner, M.D., Glasman, E.H., Petkovic, D., Yanker, P., Faloutsos, C. and Taubin, G. (1993). QBIC Project: Querying Images by Content, Using Color, Texture, and Shape. *Proceedings of the Conference on Storage and Retrieval for Image and Video Databases* 173-187.
14. Nistér, D. and Stewénius, H. (2006). Scalable Recognition with a Vocabulary Tree. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (CVPR 2006), 2161-2168.
15. Nocedal, J. and Wright, S.J. (2006). *Numerical Optimization*. Science Press.
16. Platt, J.C., Czerwinski, M. and Field, B.A. (2003). PhotoTOC: Automatic Clustering for Browsing Personal Photographs. *Proceedings of the IEEE Pacific Rim Conference on Multimedia* 6-10.
17. Schettini, R., Ciocca, G., Valsasna, A., Brambilla, C. and De Ponti, M. (2002). A Hierarchical Classification Strategy for Digital Documents. *Pattern Recognition* **35**(8). 1759-1769.
18. Shilman, M., Tan, D.S. and Simard, P. (2006). CueTIP: A Mixed-Initiative Interface for Correcting Handwriting Errors. *Proceedings of the ACM Symposium on User Interface Software and Technology* (UIST 2006), 323-332.
19. Smeulders, A.W.M., Worring, M., Santini, S., Gupta, A. and Jain, R. (2000). Content-Based Image Retrieval at the End of the Early Years. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **22**(12). 1349-1380.
20. Smith, J.R. and Chang, S.-F. (1997). VisualSEEK: A Fully Automated Content-Based Image Query System. *Proceedings of the ACM Conference on Multimedia* (Multimedia 1997), 87-98.
21. Vailaya, A., Figueiredo, M., Jain, A. and Zhang, H.J. (1999). Content-Based Hierarchical Classification of Vacation Images. *Porceedings of the IEEE International Conference on Multimedia Computing and Systems* (ICMCS 1999), 518-523.
22. Wang, S., Jing, F., He, J., Du, Q. and Zhang, L. (2007). IGroup: Presenting Web Image Search Results in Semantic Clusters. *Proceedings of the ACM Conference on Human Factors in Computing Systems* (CHI 2007), 587-596.
23. Winder, S.A. and Brown, M. (2007). Learning Local Image Descriptors. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (CVPR 2007), 1-8.
24. Yee, K.-P., Swearingen, K., Li, K. and Hearst, M. (2003). Faceted Metadata for Image Search and Browsing. *Proceedings of the ACM Conference on Human Factors in Computing Systems* (CHI 2003), 401-408.