Attribute Pivots for Guiding Relevance Feedback in Image Search

Adriana Kovashka and Kristen Grauman

(Supplementary Materials)

In this document, we present additional example search results. We also show the two supplementary results mentioned in the main text: 1) accuracy plots as a function of user feedback time, and 2) the Shoes-1k exhaustive result. We also list the attribute names in each dataset, and give additional details on the experimental setup, as promised in the main text.

1 Dataset Details

The attribute names used in each dataset are as follows:

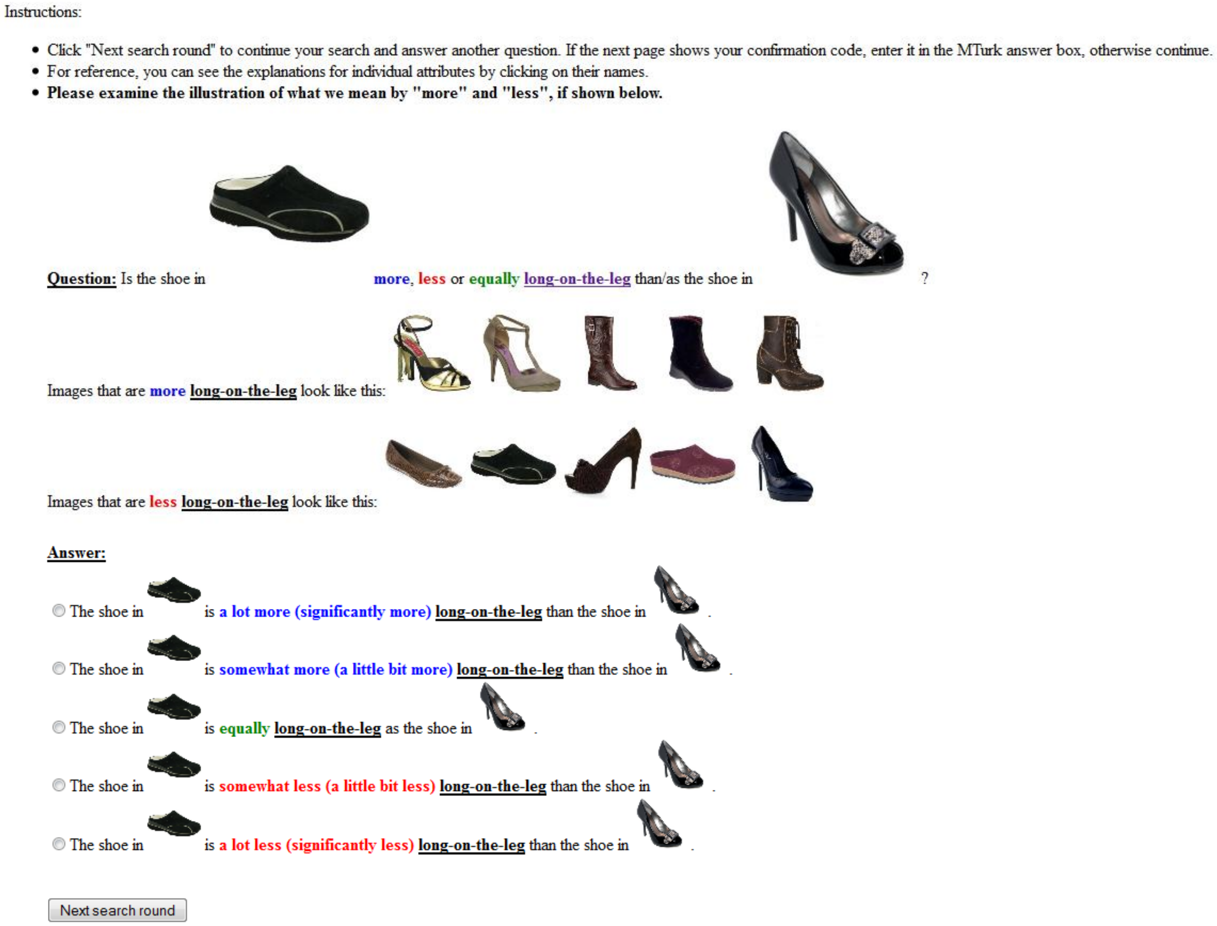
- **Shoes** – pointy at the front, open, bright in color, covered with ornaments, shiny, high at the heel, long on the leg, formal, sporty, feminine
- **Scenes** – natural, open-air, perspective, large objects, diagonal plane, close-depth
- **Faces** – male, white, young, smiling, chubby, visible forehead, bushy eyebrows, narrow eyes, pointy nose, big lips, round face
- **Faces-Unique** – male, white, young, smiling, bushy eyebrows, big lips

For Shoes, we concatenate a 960-dimensional GIST feature vector and a 30-dimensional color feature vector. For Scenes, we use a 512-dimensional GIST vector. For Faces, we concatenate a 512-dimensional GIST vector and a 30-dimensional color vector.

2 Additional Qualitative Results

In Figures 2 through 6, we show how our method and a traditional method which requests feedback on top-ranking images (referred to as Top in the main text) rank images. The top-ranked images after each iteration are shown on the right-hand side, and the user feedback from which the ranking is computed is shown on the left-hand side. The feedback was given by real users on Amazon Mechanical Turk. In each figure, our method is shown first, followed by Top.

The interface used for these experiments is shown in Figure 1.



Instructions:

- Click "Next search round" to continue your search and answer another question. If the next page shows your confirmation code, enter it in the MTurk answer box, otherwise continue.
- For reference, you can see the explanations for individual attributes by clicking on their names.
- **Please examine the illustration of what we mean by "more" and "less", if shown below.**

Question: Is the shoe in **more**, **less** or **equally** **long-on-the-leg** than/as the shoe in ?

Images that are **more** **long-on-the-leg** look like this:

Images that are **less** **long-on-the-leg** look like this:

Answer:

○ The shoe in is **a lot more (significantly more)** **long-on-the-leg** than the shoe in .

○ The shoe in is **somewhat more (a little bit more)** **long-on-the-leg** than the shoe in .

○ The shoe in is **equally** **long-on-the-leg** as the shoe in .

○ The shoe in is **somewhat less (a little bit less)** **long-on-the-leg** than the shoe in .

○ The shoe in is **a lot less (significantly less)** **long-on-the-leg** than the shoe in .

Figure 1: The interface we use for the live user experiments in Figure 5 of the paper.

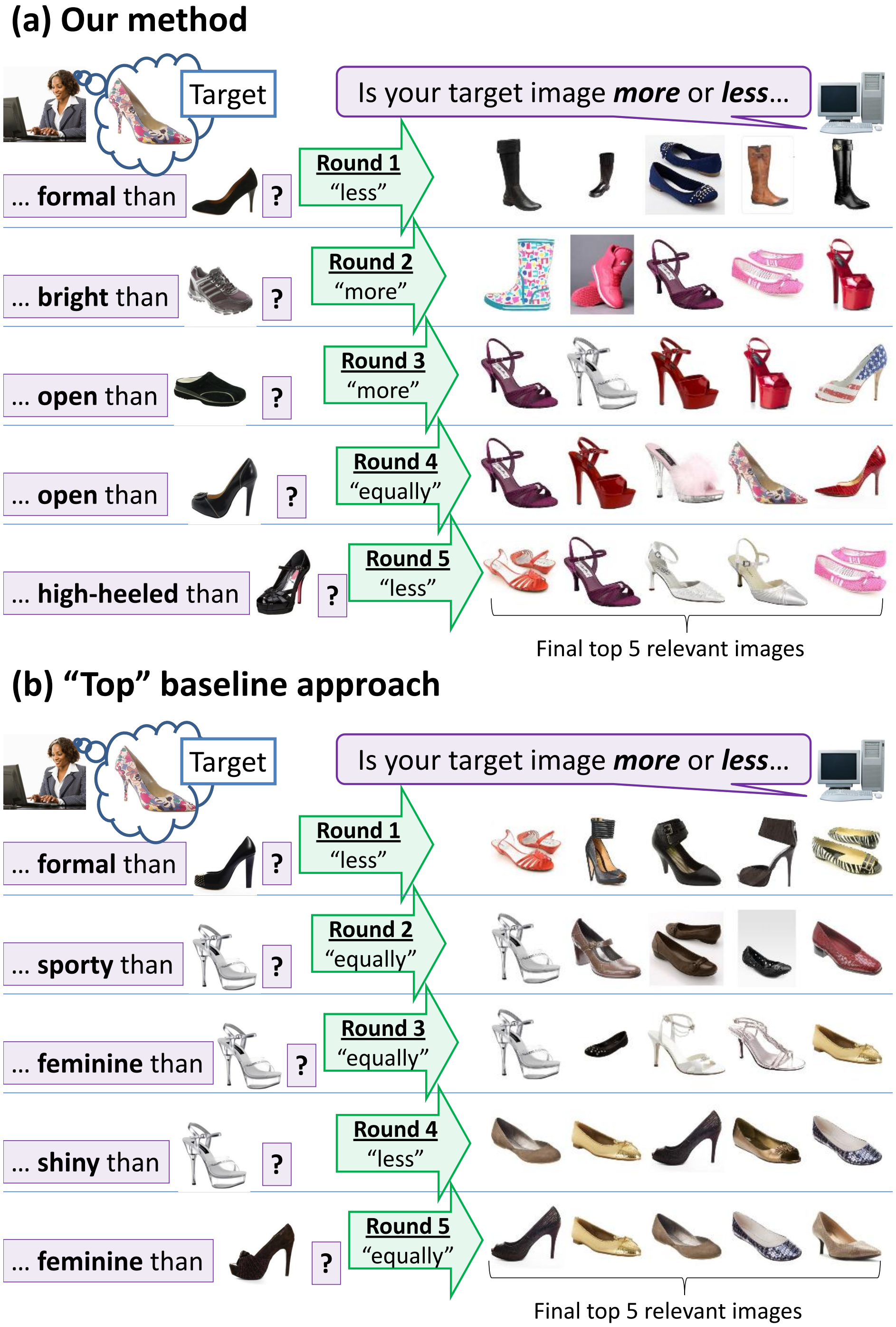


Figure 2: Our method vs Top on a Shoes-1k query. Our method quickly converges on shoes that look like the target (bright high-heeled pointy shoes). Notice how our method asks questions that are crucial in describing the shoe precisely (it is a high-heeled but not formal shoe, and it is more open than other high-heeled shoes). In contrast, Top gets stuck asking questions about the same shoe, and moreover, asking questions whose answers might be redundant (i.e., about sportiness and its opposite femininity).

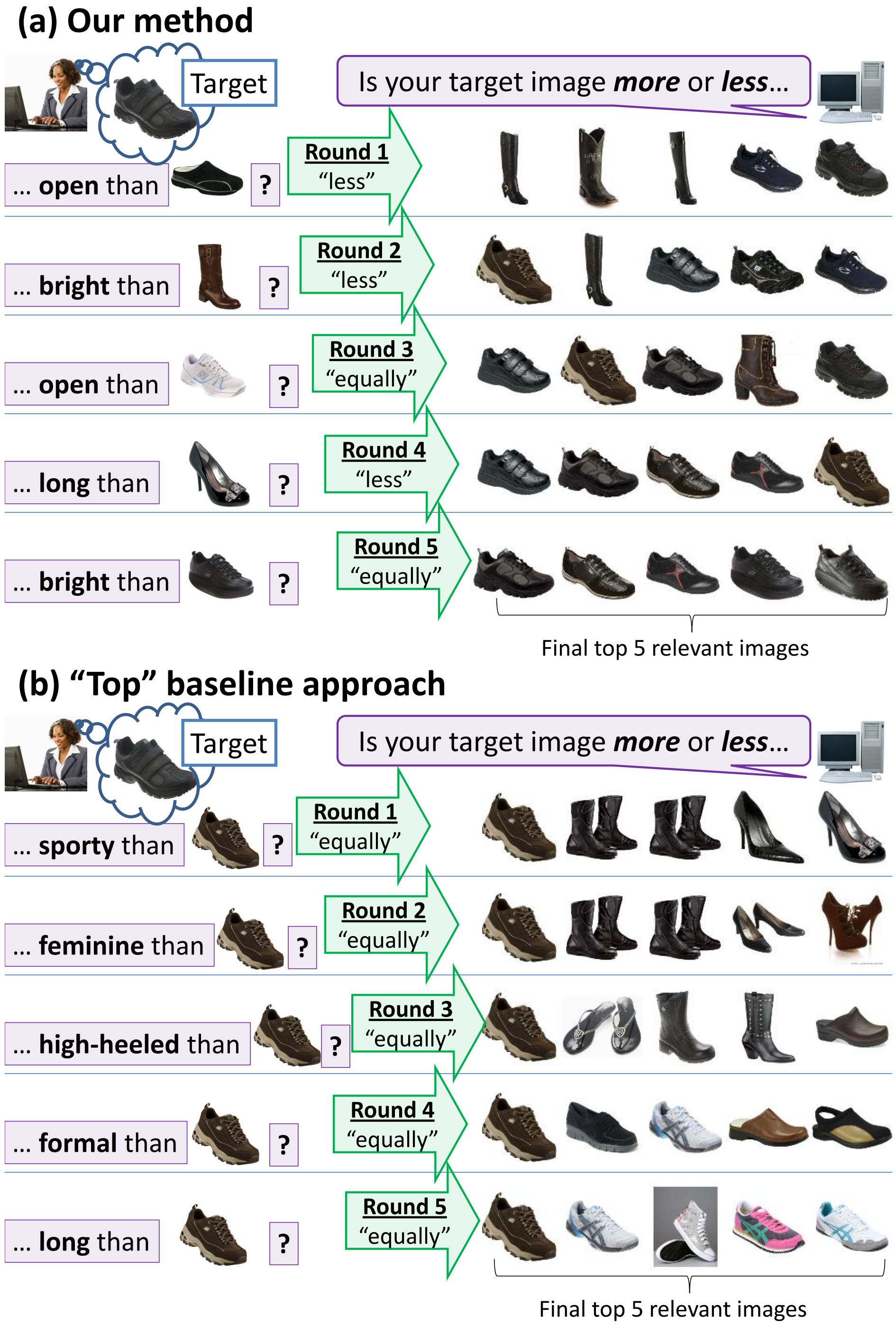


Figure 3: Our method vs Top on another Shoes-1k query. Our method asks questions about the same attribute until it gets the right level of that attribute, such as openness and brightness in this example. It starts out by retrieving images that are too un-open (which can also be described as long on the leg), but then converging on the right amount of openness. It also uses a brighter and a less-bright image to arrive at the right level of brightness. In contrast, Top over-focuses on a single image, with no way to turn its attention to another image (via a change in the image rankings) other than getting lucky and asking about a relevant attribute (e.g., brightness), which does not occur here.

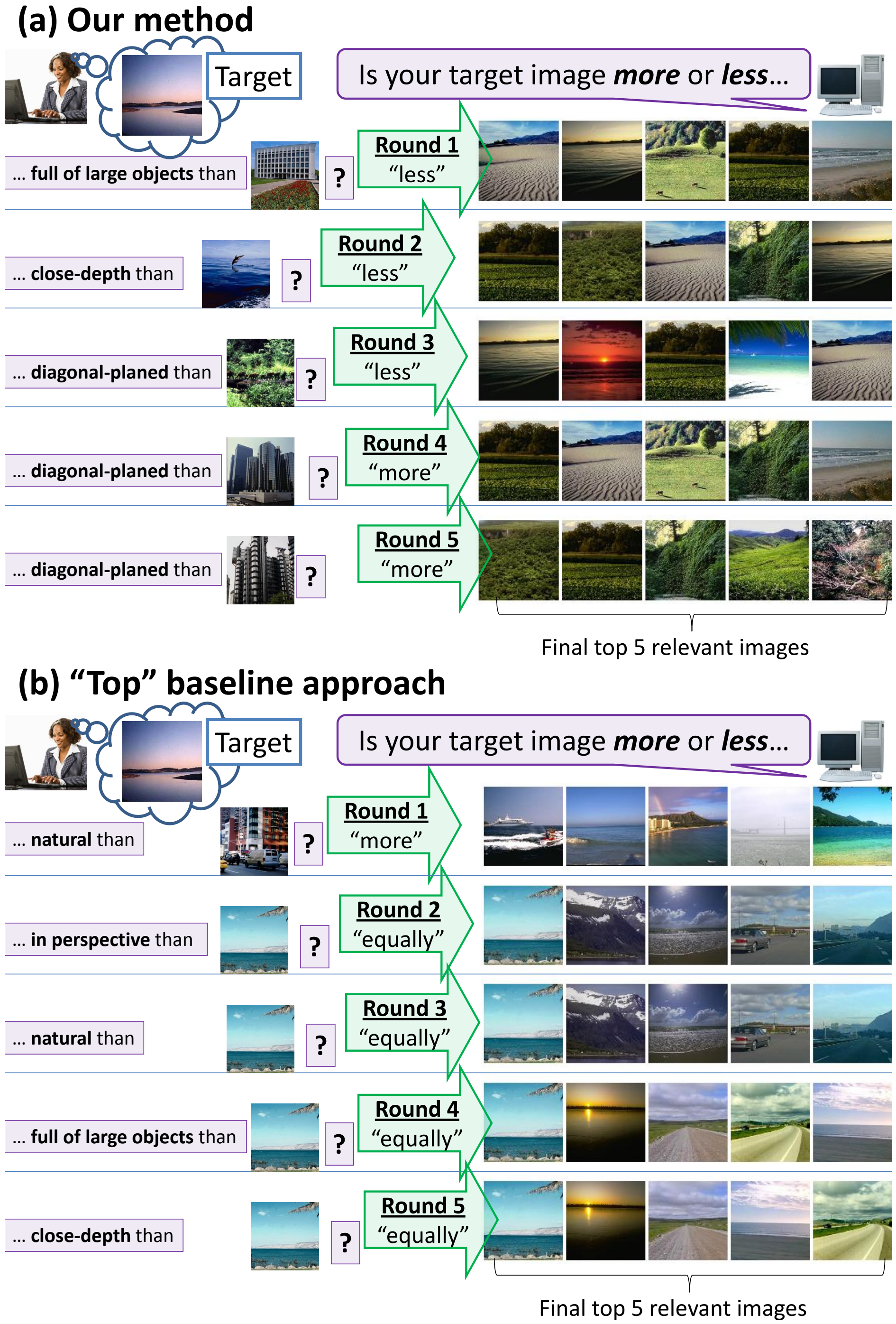


Figure 4: Our method vs Top on a Scenes query. This figure shows a failure of our method. It successfully retrieves images that share many properties with the target, but should have water where they have grass. In contrast, Top finds images that look like the target early on, and mostly preserves its ranking.

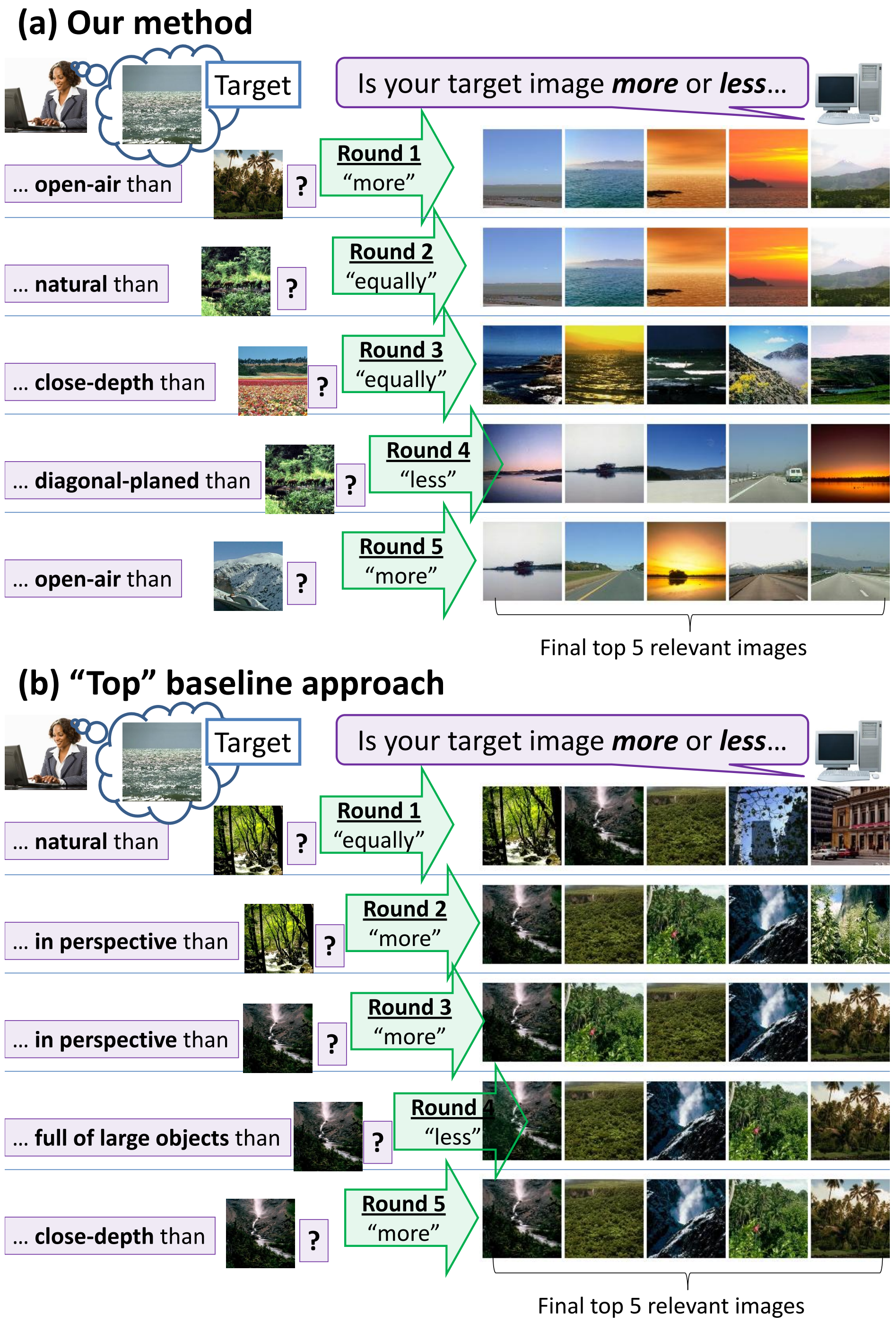


Figure 5: Our method vs Top on another Scenes query. Our method asks about properties that are important for distinguishing the target image from other images, namely open-air. Both methods start out by asking questions about images that do not look like the target, but only our method is able to provide acceptable top results.

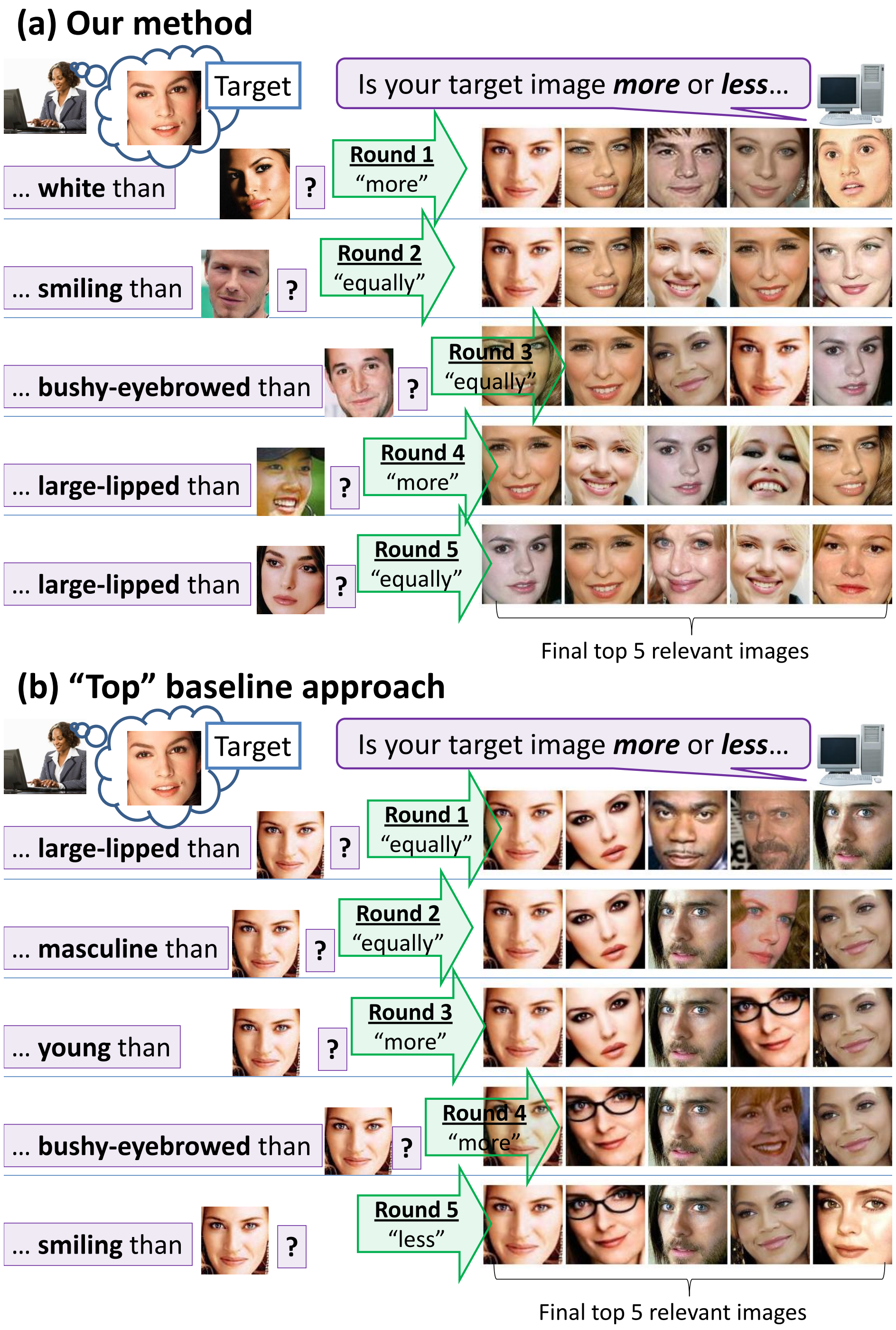


Figure 6: Our method vs Top on a Faces-Unique query. Our method's final ranking includes images of women that more closely resemble the target, and unlike the Top method, it does not erroneously include images of men. Notice how Top's top-ranked images changed little with additional feedback, i.e. some feedback requests were wasteful. In contrast, our method suitably explores the image space.

3 Plotting Rank as a Function of User Time

In Figure 7, we show how the percentile rank of the target image improves as more user time is spent on feedback. This figure is a counterpart to Figure 4 of the main text. Since binary feedback requires slightly less time than relative attribute feedback (we measured 5.5 seconds for binary feedback and 6.4 seconds for relative attribute feedback), the 20 iterations for binary feedback finish faster than the 20 iterations for relative feedback methods. However, due to the large difference between the performance of relative attribute and binary feedback, the trends remain the same as in Figure 4 of the main text.

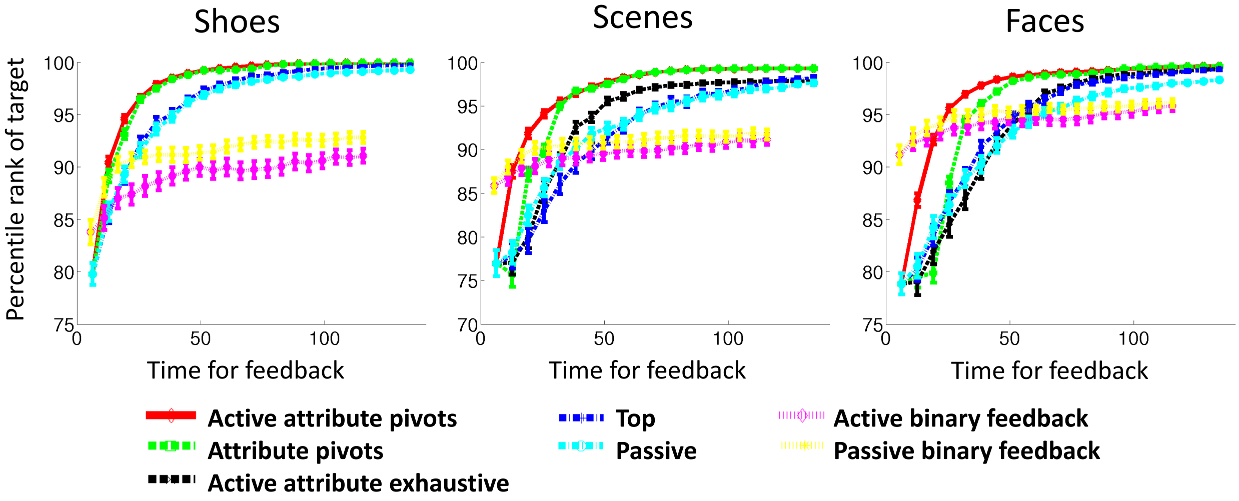


Figure 7: Percentile rank of the target image as a function of the time (in seconds) spent on feedback. The trends are the same as shown in Figure 4 of the main text, with attribute pivots outperforming the alternative methods.

4 Results for Active Attribute Exhaustive on Shoes-1k Dataset

As noted in footnote 3 in the main text, the Active attribute exhaustive baseline is too expensive to run on the entire Shoes dataset. This is why the top two plots in Figure 4 do not show a curve for that baseline (whereas the plots for Scenes and Faces do). Therefore, for completeness, here we show the results for the reduced Shoes-1k dataset. We plot performance as a function of user time in seconds, for consistency with Figure 7.

Figure 8 shows the results. Consistent with the exhaustive method's performance on Scenes and Faces, it ranks the target image poorly compared to our Active attribute pivots method, and shows images which align less well with a ground truth ranking than those shown by our method. Furthermore, our method only takes 0.015 seconds per iteration to make its choices, versus 9.964 seconds for the exhaustive method. We stress that this is not a new result; we could not fit both of these plots in the main text.

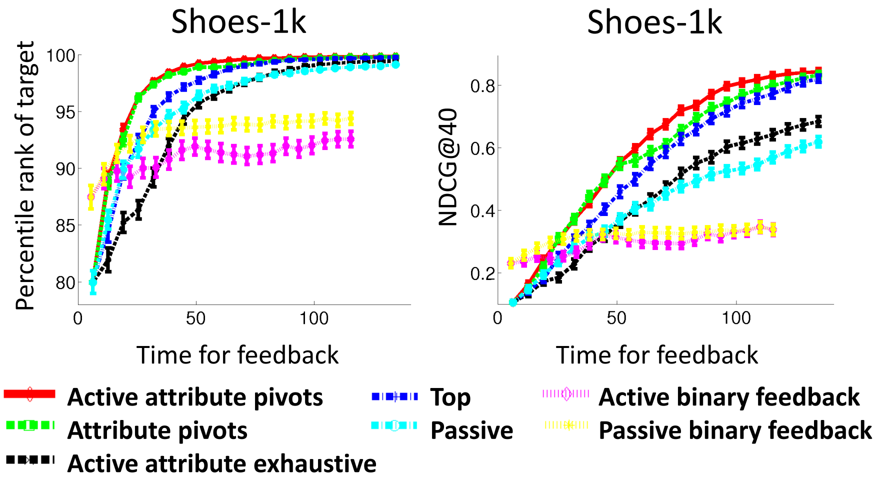


Figure 8: Comparison of our method and the six baselines on the Shoes-1k subset of the Shoes dataset. In particular, our method ranks the target image better than Active attribute exhaustive, and is computationally much faster.

5 Experimental Setup Details

The only difference between live user feedback and simulated feedback results is that in the former case, questions posed by a method are answered by a worker on Amazon Mechanical Turk, whereas in the latter case, we generate the response automatically based on the attribute estimates.

5.1 Results with Simulated Feedback

In order to generate a response automatically, we need two strategies: one for relative attribute feedback, and one for binary feedback.

For relative attribute feedback, we have to answer the question "Is the target image I_t more m than, less m than, or equally m as the pivot image I_{p_m}?", where m is an attribute. To the vector of relative attribute values for each attribute, we add Gaussian noise with $\mu = 0$ and $\sigma = 0.1s$, where s is the standard deviation of values for that attribute. This results in predicted attribute values a'_m for each attribute m. We examine the predicted relative attribute values $a'_m(I_t)$ and $a'_m(I_{p_m})$. If their difference is within a learned threshold, we generate a response of "equally". This threshold is learned from human data. In particular, it is the average of the distances along a_m for training image pairs which have been marked equal in terms of m by human judges, i.e., pairs that belong to the set E_m. We eliminate outlier training image pairs whose distances are more than one standard deviation of the values for the corresponding attribute. If the threshold for equality is exceeded, we give a response of "more" if $a'_m(I_t) > a'_m(I_{p_m})$ and "less" if $a'_m(I_t) < a'_m(I_{p_m})$. The addition of Gaussian noise is to account for the discrepancy between the human-perceived attributes A_m and our predictions a_m.

For binary feedback, we have to answer the question "Is the target image I_t similar to or dissimilar from the exemplar image I_i?" We respond with "similar" if the distance between I_t and I_i in terms of the learned perceptual distance between images is within one standard deviation of the distance between I_t and all other images in the database. Otherwise we respond with "dissimilar". We add Gaussian noise with the same parameters as above to the SVM decision values.

We initialize the binary feedback methods by peeking at the distances between the target image and a pool of 40 images, and selecting the closest image (Euclidean distance in feature space) as a positive and the furthest as a negative. This simulates a user starting the search with feedback on a page of random images. If anything, it is generous to the baseline, since our method gets only one "bit" of feedback at the onset, while the binary feedback baselines get two.

The non-probabilistic baseline to which we refer at the end of Sec. 3.2 and Sec. 3.3 of the main text makes hard decisions to prune images on the branches of a single attribute tree, depending on the user feedback. For this baseline, allowing the user to respond with "equally" means that multiple images will be left at the last explored node of the tree, so many images will all be considered "most relevant", without a way to distinguish between them and rank them. Instead, we perform standard binary search and only allow responses of "more" and "less" for this method.

5.2 Results with Live Users

We submit 50 queries for each dataset. For fairest comparison, we eliminate any queries where one or more methods did not receive 5 complete feedback iterations, leaving 34, 42, and 47 total queries for Shoes-1k, Scenes, and Faces-Unique, respectively.

We stop updating the probabilities of relevance for a method once this method places the target image in the top 40 images.

In order to get richer feedback from users, we allow users to express their confidence in their responses. Specifically, we allow them to say "a lot more" and "a lot less" in addition to "more", "less", and "equally", as a way to express their confidence of an answer. We then give twice the weight to constraints for which the user says "a lot more (less)" when computing the relevance probabilities.

We show users images from the bottom and top of our attribute rankers (see Figure 1), in order to guide their answers and ameliorate the effect of the discrepancy between machine and user understanding of an attribute. For the live experiments, we restrict the set of exemplar images for Shoes-1k and Scenes to 100 images that are diverse in terms of their attribute values. The target image is chosen as one of those 100. We do this in order to ensure that the questions posed to live users are not too difficult and the answers are not too subtle. However, we still always rank all the images according to their relevance for all methods; for this reason, the Top method's exemplar can differ from its top-ranked database image, as seen in the figures.

For the live tests with Faces, we restrict the dataset to unique individuals (as stated in the main text) because the dataset has strong category boundaries (i.e., the different celebrities) and low intra-class variation, so it does not make sense to compare one image of a person to another image of the same person.