Attribute Adaptation for Personalized Image Search

Adriana Kovashka Kristen Grauman
The University of Texas at Austin
{adriana, grauman}@cs.utexas.edu

Abstract

Current methods learn monolithic attribute predictors, with the assumption that a single model is sufficient to reflect human understanding of a visual attribute. However, in reality, humans vary in how they perceive the association between a named property and image content. For example, two people may have slightly different internal models for what makes a shoe look "formal", or they may disagree on which of two scenes looks "more cluttered". Rather than discount these differences as noise, we propose to learn user-specific attribute models. We adapt a generic model trained with annotations from multiple users, tailoring it to satisfy user-specific labels. Furthermore, we propose novel techniques to infer user-specific labels based on transitivity and contradictions in the user's search history. We demonstrate that adapted attributes improve accuracy over both existing monolithic models as well as models that learn from scratch with user-specific data alone. In addition, we show how adapted attributes are useful to personalize image search, whether with binary or relative attributes.

1. Introduction

Visual attributes are human understandable properties to describe images, e.g., *shiny*, *natural*, or *white*. Recent research explores a variety of applications for attributes, including object recognition [15, 6, 3] and image retrieval [14, 23, 21, 13, 12].

Thus far, training an attribute predictor largely follows the same procedure used for training any image classification system: one collects labeled image exemplars, extracts image descriptors, and applies discriminative learning. The underlying assumption is that an image has a single "true" category label that objective viewers could agree upon. Yet, while this holds for objects (a horse is a horse, of course), an attribute inherently has more leeway. Multiple objective viewers are bound to have slightly different internal models of a visual property. Indeed, researchers collecting attribute-labeled datasets report non-negligible disagreement among human annotators [6, 5, 19].

Figure 1. Visual attribute interpretations vary slightly from viewer to viewer. This is true whether attributes are modeled as categorical or relative properties. For example, 5 viewers *confidently* declare the shoe as formal (left) or more ornamented (right), while 5 others confidently declare the opposite! We propose to adapt attribute models to take these differences in perception into account.

The differences may stem from several factors: the words for attributes are imprecise (when is the cat *overweight* vs. *chubby*?), their meanings often depend on context (the shoe appears *comfortable* for a wedding, but not for running) and even cultures (languages have differing numbers of color words, ranging from two to eleven), and they often stretch to refer to quite distinct object categories (e.g., *pointy* pencil vs. *pointy* shoes). For all such reasons, humans inevitably craft their own definitions for visual attributes. Notably, their definitions vary whether we consider binary or relative attributes (see Fig. 1).

This variability has important implications for any application where a human uses attributes to communicate with a vision system. For example, in image search, a user requests images containing certain attributes [14, 23, 21, 13, 12]; in recognition, a user teaches a system about objects by describing their properties [15, 6, 3, 16, 17]. *Failing to account for user-specific notions of attributes will lead to discrepancies between the user's precise intent and the message received by the system.* Yet, even when training labels are solicited from multiple annotators, existing methods learn only a single "mainstream" view of each attribute, forcing a consensus through majority voting.[1]

We propose to model attributes in a user-specific way, in order to capture the inherent differences in perception. How can we do so efficiently? The most straightforward

[1]We stress this is the case whether using binary [15, 6] or relative [16] attributes. For binary properties, one takes the majority vote on the attribute present/absent label. For relative properties, one takes a majority vote on the attribute more/less label.

approach—to learn one function per attribute and per user, from scratch—is certainly not scalable in most reasonable application settings, and ignores the reality that people do share *some* foundational definition of a visual property.

Instead, we pose attribute learning as an *adaptation* problem. First, we leverage any commonalities in perception to learn a *generic* prediction function, namely, a classifier for a binary attribute (e.g., *pointy*) or a ranking function for a relative attribute (e.g., *pointier than*). Then, we use a small number of user-labeled examples to adapt that model into a *user-specific* prediction function. In technical terms, this amounts to imposing regularizers on the learning objective favoring user-specific model parameters that are similar to the generic ones, while still satisfying the user-specific label constraints [28, 8].

To further lighten the user's labeling load, we introduce two ways to extrapolate beyond the labels explicitly provided by a given user. In the first, we connect relative attribute statements given by the user on multiple different images to obtain new implicit constraints via transitivity. In the second, we detect discrepancies between the system's generic attribute models and the user's perception, and create implicit constraints to correct the models. Both ideas serve to generate additional plausible user-specific labels without directly requesting more labels from the user.

While our adapted attributes are applicable to any task demanding precise human-system communication about visual properties, we focus specifically on their impact for image search. We demonstrate the advantages of personalized retrieval when a user queries for images with multi-attribute keywords or uses attributes to provide relevance feedback on selected reference images. In this context, we show that a user's search history offers a natural source of data for inferring user-specific labels.

To validate our idea, we experiment with 75 unique users on two large datasets. We compare our user-specific adapted attributes to a standard generic "consensus" model, as well as a baseline that trains exclusively with user-specific data. We show that adapting learned models is an efficient way to capture person-dependent interpretations, particularly for fine-grained attribute distinctions where perception varies most. Furthermore, we show that our ideas to extrapolate user-specific labels can successfully mitigate labeling effort. Finally, we demonstrate the practical impact of adapted attributes for personalized search. Throughout, our method's consistent advantages highlight the risk in assuming one attribute model fits all.

2. Related Work

Learning visual attributes Visual attributes, originally introduced in [15, 6], offer a semantic representation shared among objects. Attributes may be expressed categorically, as a property that is either present or absent, or relatively [16], as a property that is present with a certain strength. Attributes are valuable not only for recognition [15, 6, 3, 22, 19], but also for keyword-based image search (e.g., "find images of *smiling Asian* men" [14, 23], or "find images of men *smiling more than/similarly to* this one" [21, 13, 12]). Recent user studies analyze how humans perceive subjective properties like *cool* and *cute* [4], but do not propose vision techniques to account for the subjectivity. Typically discriminative classifiers or ranking algorithms are used to predict attributes. To our knowledge, all prior work assumes monolithic attribute predictors are sufficient, and none attempts to model user-specific perception, as we propose. This includes prior methods that represent attributes relatively [16, 22]; though they permit looser comparative labels, they still assume a single underlying relative concept and learn a single "true" ordering of images.

Transfer learning and adaptation We adapt a generic attribute model to learn a user-specific one. Somewhat analogously, transfer learning work in object recognition leverages previously learned object categories when training a new category for which few labeled images are available (e.g., [24, 1]). Also related are domain adaptation methods (e.g., [20, 10, 9]), which account for the feature distribution mismatch between a source domain (in which objects are learned) and a target domain (in which the objects must be recognized)—for example, allowing a classifier trained on Web images to work well on images taken by a robot [20]. Conceptually our goal is perhaps closer in spirit to speaker-dependent speech recognition. Speaker adaptation methods have long been used in the speech community to adapt parameters of a speaker-independent model to account for an individual's idiosyncrasies (voice, accent, etc.) [7]. We explore existing adaptation formulations for SVMs [28, 8]; using them in our setting is novel, and we introduce methods to infer user-specific training labels for these models.

Modeling users in crowdsourcing Learning from multiple noisy labelers is increasingly important for training data-hungry vision systems. Typically, an image labeling task is "crowdsourced" by submitting it to workers on Mechanical Turk, then aggregating their labels through majority vote. Moving beyond majority vote, recent work discovers the skills and biases of individual workers in order to better infer the true labels [27, 25]. However, while they model each worker's "school of thought", they still aim for consensus and recover a single true label per example. In contrast, we recover an individual user's subjective attribute model from their annotations, by properly adapting a generic model over all previously seen users.

Personalization in information retrieval In information retrieval, personalization involves learning what a given user perceives as relevant, and producing user-specific

search results [18]. Relevance feedback can be mined explicitly or implicitly, by creating user profiles or mining clickthrough data [11]. We incorporate a novel form of implicit cue from visual search. Furthermore, whereas personalization generally entails learning a user-specific relevance function from scratch—there is no "universal" prior on relevance—we leverage a generic model for the attribute as a starting point, and efficiently adapt it towards the user's preferences. As we demonstrate, doing so saves user time.

3. Approach

We first train a *generic* model of an attribute using a large margin learning algorithm and data labeled with majority vote from multiple annotators. This is the "source" model, in transfer learning terms. Then, for a given user, we adapt the parameters of the generic model to account for any user-specific labeled data, while not straying too far from the prior generic model. We refer to the resulting prediction function as an *adapted attribute* or *user-specific attribute*. This is the "target" model, in transfer learning terms.

In the following, we first overview the adaptation learning algorithms we use (Sec. 3.1). Then, we briefly describe how we use the adapted attributes to perform personalized content-based image search (Sec. 3.2). Finally, we explain how we gather explicit and implicit user-specific labeled data (Sec. 3.3).

3.1. Learning Adapted Attributes

We consider two variants of attributes: binary attributes, which entail learning a classifier, and relative attributes, which entail learning a ranking function. For both, we perform adaptation with a large-margin formulation and a regularizer preferring user-specific parameters that do not deviate greatly from the generic parameters.

Adaptation requires that the source and target tasks be related, such that it is meaningful to constrain the target parameters to be close to the source's. Whereas in some transfer problems this requires a "leap of faith" and/or hand crafting (e.g., to specify that bicycle classifiers should transfer well to motorcyles), in our setting the assumption naturally holds. An attribute is semantically meaningful to all annotators, just with (usually slight) perceptual variations among them. Thus, we are assured that the generic model is a valid prior for each novel user we aim to adapt to.

We learn each attribute of interest separately (i.e., one classifier for *white*, another for *pointy*). Similarly, an adapted function is user-specific, with one distinct function for each user. In the following, we do not notate individual attributes or users to avoid subscript clutter.

Let D' denote the set of images labeled by majority vote that are used to learn the generic model. Let $\boldsymbol{x}_i$ denote a feature describing the i-th image (texture, color), and y_i be its label. We assume the labeled examples originate from a pool of possibly many annotators who collectively represent the common denominator in attribute perception. We train a generic attribute $f'(\boldsymbol{x}_i)$ from D'. Let D denote the set of user-labeled images, which is typically disjoint from D'. Both adaptive learning objectives below will take a D and f' as input, and produce an adapted attribute f as output.

Adapting binary attribute classifiers Binary attributes predict whether or not an attribute is present in an image. In this case, the generic data $D'_b = \{\boldsymbol{x}'_i, y'_i\}_{i=1}^{N'}$ consists of N' labeled images, with $y'_i \in \{-1, +1\}$. The subscript b denotes *binary*. Let f'_b denote the generic binary attribute classifier trained with D'_b. For a linear support vector machine (SVM), we have $f'_b(\boldsymbol{x}) = \boldsymbol{x}^T \boldsymbol{w}'_b$. To adapt the parameters $\boldsymbol{w}'_b$ to account for user-specific data $D_b = \{\boldsymbol{x}_i, y_i\}_{i=1}^{N}$, we use the Adaptive SVM [28] objective function:

$$\min_{\boldsymbol{w}_b} \frac{1}{2}\|\boldsymbol{w}_b - \boldsymbol{w}'_b\|^2 + C\sum_{i=1}^{N}\xi_i, \quad (1)$$
$$\text{subject to} \quad y_i \boldsymbol{x}_i^T \boldsymbol{w}_b \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad \forall i$$

where $\boldsymbol{w}_b$ denotes the desired user-specific hyperplane, and C is a constant controlling the tradeoff between misclassification on the user-specific training examples and the regularizer. Note, the objective expands the usual large-margin regularizer $\|\boldsymbol{w}_b\|^2$ to additionally prefer that $\boldsymbol{w}_b$ be similar to $\boldsymbol{w}'_b$.[2] Thus the generic attribute serves as a prior for the user-specific attribute, such that even with small amounts of user-labeled data we can learn an accurate predictor.

The optimal $\boldsymbol{w}_b$ is found by solving a quadratic program to maximize the Lagrange dual objective function. This yields the Adaptive SVM decision function:

$$f_b(\boldsymbol{x}) = f'_b(\boldsymbol{x}) + \sum_{i=1}^{N} \alpha_i y_i \boldsymbol{x}^T \boldsymbol{x}_i, \quad (2)$$

where $\boldsymbol{\alpha}$ denotes the Lagrange multipliers that define $\boldsymbol{w}_b$. Hence, the adapted attribute prediction is a combination of the generic model's prediction and similarities between the novel input $\boldsymbol{x}$ and (selected) user-specific instances $\boldsymbol{x}_i$.

Adapting relative attribute rankers Rather than make a hard decision about attribute presence, relative attributes predict the strength of an attribute in an image [16]. In this case, labels are provided in terms of ordered pairs of examples: $D'_r = \{(\boldsymbol{x}'_{i_1}, \boldsymbol{x}'_{i_2})\}_{i=1}^{N'}$, where the subscript r denotes *relative*. Each pair denotes that image i_1 exhibits the attribute more strongly than image i_2—for example, that i_1 is *pointier* than i_2. Therefore, collecting D'_r requires asking multiple annotators to vote on which of the two images exhibit the attribute more. Implicitly, this corresponds to $y'_{i_1} > y'_{i_2}$, though during training the absolute strengths

[2] See [1] for a variant that separates the transfer and margin regularizers.

are irrelevant—only the comparative values matter. Following [16], we use a Rank SVM [11] approach to train each generic relative attribute. The Rank SVM seeks a hyperplane $\boldsymbol{w}_r'$ that, when used to project all training data, 1) maintains their specified orderings, and 2) keeps a wide margin between the nearest projected points. For a linear ranker, we have $f_r'(\boldsymbol{x}) = \boldsymbol{x}^T \boldsymbol{w}_r'$.

To adapt the parameters $\boldsymbol{w}_r'$ to account for user-specific ordered pairs $D_r = \{(\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2})\}_{i=1}^N$, we use a Ranking Adaptation SVM [8]. It modifies the Rank SVM objective to add a regularizer that, similar to above, prefers that the resulting function stay close to the generic one. Specifically, to learn the adapted ranker, we optimize:

$$\min_{\boldsymbol{w}_r} \frac{1-\delta}{2}\|\boldsymbol{w}_r\|^2 + \frac{\delta}{2}\|\boldsymbol{w}_r - \boldsymbol{w}_r'\|^2 + C\sum_{i=1}^{N}\xi_i \quad (3)$$
$$\text{subject to} \quad \boldsymbol{w}_r^T \boldsymbol{x}_{i_1} - \boldsymbol{w}_r^T \boldsymbol{x}_{i_2} \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad \forall i,$$

where $\boldsymbol{w}_r$ denotes the user-specific hyperplane, and $\delta \in [0, 1]$ is a constant balancing the two regularizers. The constraints reflect that the resulting $\boldsymbol{w}_r$ ought to rank each $\boldsymbol{x}_{i_1}$ higher than its corresponding $\boldsymbol{x}_{i_2}$, with a large margin. Again the solution requires solving a quadratic program [8], and the resulting adapted relative attribute predictor is:

$$f_r(\boldsymbol{x}) = \delta f_r'(\boldsymbol{x}) + \sum_{i=1}^{N} \beta_i \boldsymbol{x}^T(\boldsymbol{x}_{i_1} - \boldsymbol{x}_{i_2}), \quad (4)$$

where $\boldsymbol{\beta}$ denotes the Lagrange multipliers defining $\boldsymbol{w}_r$. Though shown here as linear functions, non-linear decision boundaries and rankers are also possible via kernelization.

Suitability for adapted attributes Having defined the two adaptation methods, we can now reflect on their strengths for our problem. The adaptive formulations integrate the generic model and user-specific data during learning. This is preferable to independently training generic and user-specific models then combining their outputs, which is prone to overfit to the few available user-labeled examples. Intuitively, when optimizing Eqn. 1 or 3, a larger weight on a user-specific support vector $\boldsymbol{x}_i$ is more likely when the generic model f' mispredicts $\boldsymbol{x}_i$, i.e., when $f_b'(\boldsymbol{x}_i) \neq y_i$ or $f_r'(\boldsymbol{x}_{i_1}) \not\succ f_r'(\boldsymbol{x}_{i_2})$. Thus, user-specific instances that deviate from the generic model will have more impact on f. For example, suppose a user mostly agrees with the generic notion of *formal* shoes, but, unlike the average annotator, is also inclined to call loafers *formal*. Then the adapted classifier will likely exploit some user-labeled loafer image(s) with nonzero α_i in Eqn. 2 when predicting whether a shoe would be perceived as formal by that user.

The adaptation strategy promotes efficiency in two ways. First, the human labeling cost is low, since each user only needs to provide a small amount of labeled data. In experiments, we see substantial gains with as few as 12 user-labeled examples (Fig. 3). Second, training time is substantially lower than training each user model from scratch by pooling the generic and user-specific data. We train the generic model once, offline, with a large pool of annotations. Then, the user-specific function is trained with a small amount of new data and the (already fixed) parameters $\boldsymbol{w}'$. This amortizes the "big" generic SVM's training cost—superquadratic in the number of training examples—across all future user-specific functions we learn. The efficiency is especially valuable for personalized search, where we continually adapt a user's attributes as his search history accumulates more user-specific data.

Finally, a more subtle advantage of our model choice is its modularity. The adaptation objectives do not require access to the generic training data. This is convenient, since in practice the data could be proprietary or simply unwieldy to pass around, yet one still would like to avoid learning personal attributes from scratch.

3.2. Personalized Image Search

We next briefly describe how we use the adapted attributes to personalize image search results. Compared to using generic attributes, the personalized results should more closely align with the user's perception, leading to more precise retrieval of relevant images.

For binary attributes, we use the user-specific classifiers to retrieve images that match a multi-attribute query. Similar to [14], the user states "I want images with attributes X, Y, and not Z". For relative attributes, we use the adapted rankers to retrieve images that agree with comparative relevance feedback. Similar to [13], the user states "I want images that show more of attribute X than image A and less of attribute Y than image B", etc. Then, in both cases, the system sorts the database images according to how confidently the adapted attribute predictions agree with the attribute constraints mentioned in the query or feedback. We use the magnitude of classifier/ranker outputs as confidences.

We stress that our contribution is how to adapt attributes, not how to perform search with attributes, which is studied extensively in other work [14, 23, 21, 13, 12]. One can directly incorporate our adapted attributes into any existing attribute-search method.

3.3. Obtaining User-Specific Labeled Data

In order to learn an adapted attribute, we need to populate D with data annotated by the specific user. We present two forms of data collection: explicit and implicit.

Explicit collection Most directly, we ask the user to label a small set of images with the presence/absence of attributes (in the binary case) or pairs of images with comparative labels of the form "Image A is more/less/equally

[attribute name] than Image B" (in the relative case). We track worker IDs on MTurk to keep each user's data separate. We convey the generic attribute meanings via qualification tests.

When collecting labels explicitly, the main consideration is how to select the images that the user should annotate. Intuitively, we want to focus on examples for which his perception is likely to deviate from the generic model. Thus, we take an active learning approach. For binary attributes, we consider two forms. The first uses a margin criterion [26], requesting labels for those N images closest to the generic classifier's hyperplane. For the second, we devise a variant of the query-by-committee criterion, requesting user-specific labels for the N images where the human-given generic labels were most in disagreement.

While we find the margin criterion useful for binary attributes, for relative attributes it is less so. This is likely because it is hard to meaningfully choose which of two images has the attribute "more" when they are very close. Therefore, for relative attributes we adopt a simple diversity-based active selection scheme. We sort the candidate image pairs by their Euclidean distance in feature space, and request user comparisons on an even mix of the most similar, most dissimilar, and those surrounding the median.

Our preliminary experiments indicated that actively obtained user-specific labels result in better models than passively obtained labels, as expected. Thus, we use them in all results. An empirical study of various active selection methods is beyond the scope of this research.

Implicit collection Explicit labels offer the purest cues, but they also place some burden on the user. Therefore, we propose ways to infer "implicit" user-specific labels by mining the user's relative attribute search history. We define two forms based on *transitivity* and *contradictions*.

In the transitivity case, we infer constraints on-the-fly when the user gives feedback of the form "I want images *flatter* than Image A and *less flat* than Image C." Let B denote the user's mental target image—the item he envisions finding in the database. From his feedback, we now know that $f_r(B) > f_r(A)$ and $f_r(B) < f_r(C)$ in terms of *flatness*. By transitivity, we can infer a new user-specific label pair for D_r that requires that $f_r(A) < f_r(C)$.

In the contradictions case, we exploit seeming contradictions in a user's relevance feedback. If he issues statements that appear contradictory according to the current model, he implicitly reveals a discrepancy between his perception and the system's models. For example, if he says, "I want images *whiter* than A and *less white* than B", but the current whiteness model says $f_r(A) \approx f_r(B)$, then in principle the set of images satisfying the user's model is empty.

Contradictions on the *same* attribute, while informative, are bound to be infrequent. Thus, we generalize this idea to the case where contradictions may occur *across* attributes.

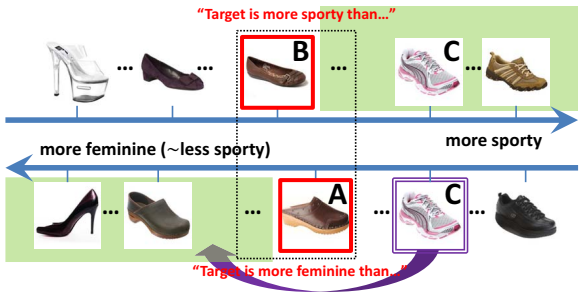


Figure 2. Example illustrating our idea for extracting implicit user-specific labels for a user's search history. See text for details.

We discover which pairs of attributes are strongly correlated or anti-correlated[3]. Now treating strongly (anti-)correlated attributes as the same (opposite) attribute, we detect contradictions as described above, for images A and B that have the same approximate attribute rank. Consider Fig. 2, where *feminine* and *sporty* are strongly anti-correlated. If the user requests images both *more feminine* than A and *more sporty* than B, where A and B are similarly feminine and similarly sporty, he seems to indicate that no images satisfy both constraints (green regions share no images). This suggests his perception on one or both attributes differs from the current model f'_r. For example, perhaps he finds a pink sneaker C (which is high on *sportiness*) more *feminine* than clog A.

For each constraint in a contradictory pair, we select an image C that violates it by a small margin, and create an implicit user-specific pair using A and C in the reverse order of how the current generic attribute ranks them. In Fig. 2, we create a pair "C is more feminine than A". By swapping the order, we correct the attribute model, and the theoretical set of images satisfying the user's mental target is no longer empty (image C is now in both green regions). Thus, we have a better chance to align with the user's perception.

Naturally, getting such labels "for free" carries some risk. We are not guaranteed that a user would agree with all implicit labels, if asked. Our approach can be viewed as a twist on self-training, a semi-supervised learning method in which one trains a classifier with labeled data, then uses it to classify unlabeled examples, and augments the labeled training set with the most confident predictions. As we demonstrate in the results, labels inferred from the user's search history prove to be quite valuable.

4. Experiments

We evaluate adapted attributes in terms of both their generalization accuracy (Sec. 4.1) and their utility for personalized image search (Sec. 4.2).

Compared methods We compare our **user-adaptive** approach to the following three methods:

[3]We say two attributes are strongly correlated if they share at least a third of the images in their top or bottom quartiles.

- **generic**: learns a model from the generic majority vote data D' only. This is how attributes are learned in prior work (e.g. [15, 6, 3, 16, 14, 21, 13]).
- **generic+**: is just like above, but uses more generic data. For every additional user-specific label our method gets, it gets an additional generic label from some other user. This baseline lets us compare the effect of adapting to user-specific data versus simply adding more generic data.
- **user-exclusive**: learns a user-specific model from scratch, without the generic model. It always uses the exact same user-specific data as our method. This baseline lets us see how much our method benefits from regularization with the generic model.

Aside from these distinctions, all methods use the exact same features and learning algorithms.

Datasets and features We use two datasets: Shoes [2], which contains 14,658 online shopping images describable by 10 attributes [13], and SUN Attributes [19], which contains 14,340 scenes. We consider 12 attributes from SUN[4] that appear frequently and are likely to be relevant for image search applications. Shoes has both binary and relative attributes; SUN has only binary. Since SUN comes with annotators' label votes, we use the query-by-committee criterion to solicit user-specific labels. Since Shoes does not, we use the margin criterion (see Sec. 3.3). These datasets represent the largest and most challenging attribute-labeled collections available today, and they allow us to observe the impact of adaptation for both a narrow class of objects (Shoes) as well as a wide domain of scenes (SUN).

To form descriptors $\boldsymbol{x}$ for Shoes, we use the GIST and color histograms provided by [13]. For SUN, we concatenate features provided by [19]: GIST, color, and base HOG and self-similarity. We cross-validate δ and C for all models, per attribute and user.

4.1. Adapted Attribute Accuracy

First we evaluate generalization accuracy: will adapted attributes better agree with a user's perception in novel images? To form a generic model for each dataset, we use 100-200 images (or pairs, in the case of Shoes-R) labeled by majority vote. We collect user-specific labels on 60 images/pairs, from each of 10 (Shoes) or 5 (SUN) workers on MTurk. We reserve 10 random user-labeled images per user as a test set in each run, and show average accuracy and standard error across 300 random splits.

Fig. 3 shows representative results. In the upper-right, we show an averaged result over all datasets, attributes, and users. We plot test accuracy as a function of the amount of additional training data beyond the generic pool. Generic remains flat, as it gets no additional data. For binary attributes, chance is 50%; for relative it is 33%, since there are three possible responses ("more", "less", "equally").

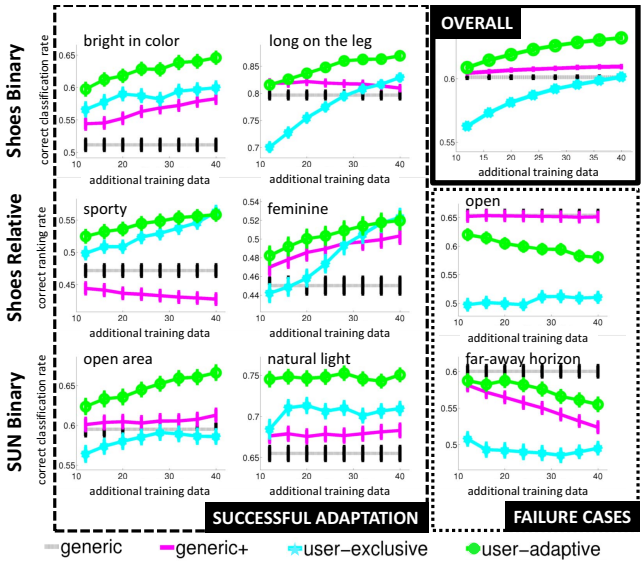


Figure 3. Accuracy as more training data is added.

Our adapted attributes typically outperform all other methods. Our advantage over the generic model supports our main claim: we need to account for users' individual perception when learning attributes. Further, our advantage over the user-exclusive model shows our approach successfully leverages "universal" perception as a prior; learning from scratch is inferior, particularly if very few user-specific labels are available (see leftmost points on plots). With more user-specific labels, the non-adaptive approach can sometimes catch up (see *feminine*), but at the expense of a much higher burden on each user. Finally, the generic+ baseline confirms that our advantage is not simply a matter of having more data available. Generic+ usually gives generic a bump, but much less than user-adaptive. For example, on *bright*, we improve accuracy by up to 26%, whereas generic+ only gains 14%.

We do see some failure cases though. The failures are by definition rather hard to analyze. That's because by focusing on user-specific perception, we lose any ability to filter noisy label responses (e.g., with voting). So, when a user-adapted model misclassifies, we cannot rule out the possibility that the worker himself was *inconsistent with his personal perception* of the attribute in that test case. Nonetheless, we do see a trend in the failure cases—weaker user-exclusive classifiers. As a result, our model can start to underperform generic, pulled down by (what are possibly inconsistent) user responses.

Fig. 4 shows example attribute spectra for three generic

[4]SUN: sailing, vacationing, hiking, camping, socializing, shopping, vegetation, clouds, natural light, cold, open area, far-away horizon. Shoes: pointy, open, bright, ornate, shiny, high, long, formal, sporty, feminine

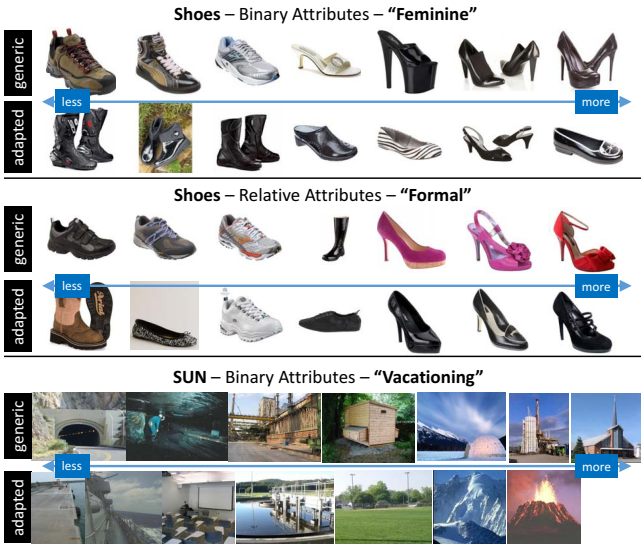
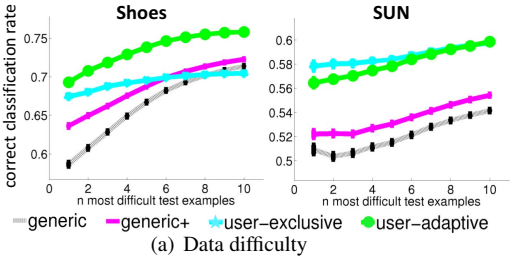


Figure 4. Example learned generic (top row per example) and user-specific (bottom row per example) attribute spectra.



(a) Data difficulty

	Most divergent	All
generic	58.66 (0.35)	71.38 (0.11)
user-exclusive	71.86 (0.33)	70.54 (0.11)
user-adaptive	69.91 (0.29)	75.78 (0.10)

(b) User difficulty

Figure 5. Accuracy as a function of task difficulty. Best on pdf.

and adapted attribute predictions, sorted from least to most. They illustrate how our method captures user-specific nuances in attribute meaning. In the top set, it learns that this user perceives flat fancy shoes to be *feminine*, whereas the generic impression is that high-heeled shoes are more feminine. In the middle set, it learns that for this user, shoes that are darker in color are more *formal*, whereas the generic model says shoes similar but brighter in color are formal. In the bottom set, it learns that this user finds landscapes with mountains more *vacation-like* than other settings.

Fig. 5 analyzes our method's impact as a function of task difficulty, using all 40 training labels. First, we consider test case difficulty (a), as measured by the distance to the binary attribute generic hyperplane; closer instances are more difficult. We sort the 10 test examples per split by difficulty, and average over all attributes and users. We plot accuracy as we add increasingly easy examples to the test set. We see that user-adapted attributes are often strongest when test cases are hardest. This is intuitive, since the intent of our method is to capture what may be subtle, fine-grained perceived differences. For SUN attributes, the user-exclusive model outperforms ours by a small margin for the most difficult examples, likely because binary judgments are hard to make for some of these attributes, making the generic prior less valuable.

Second, in Fig. 5(b) we consider user difficulty on the Shoes Binary dataset, as measured by how often a worker disagrees with the majority. Numbers in parens are standard error over all binary shoe attributes and random splits. The margin between our adaptive method and the generic method is significantly increased for divergent workers (left col) compared to all workers (right col), as the generic model is insufficient when the user has a unique perception. In contrast, our method faithfully captures users' notions.

4.2. Personalized Search with Adapted Attributes

Next we show that correctly capturing attribute perception is important for accurate search. Search is a key application where adapted attributes can alleviate inconsistencies between what the user says, and what the (traditionally majority-vote-trained) machine understands. For all search results, we use the attributes that seem most in need of adaptation, based on our previous results (5 for Shoes, 4 for SUN).

Multi-attribute keyword search First we evaluate multi-attribute keyword queries. We ask 10 MTurkers to label 40 images for each of the attributes. We train all models, then apply them to a test set of 20 held-out images per user. We issue all combinations of 3-attribute queries. Accuracy is the percentage of test images where the binary predictions on all 3 query attributes agree with that user's ground truth.

Fig. 6(a) shows the results, averaged over all users and queries. We see that the generalization power of the adapted attributes translates into the search setting. Our method is substantially better at finding the images relevant to the user. This result demonstrates how our idea can benefit a number of prior binary attribute search systems [14, 23, 21].

Relevance feedback with relative attributes Next we evaluate adapted attributes for relevance feedback. We ask 10 users for whom we have trained user-specific relative attribute models to examine 10 target query images, and tell us whether they exhibit a specified attribute more/less/equally than 20 random reference images. This yields a total of 20 feedback statements per query per user.

Fig. 6(b) shows the averaged results. Since in this scenario the user describes a single (known to us) image, we gauge accuracy in terms of the percentile rank for this target image, i.e., the proportion of database images that the system ranks lower than the correct target image that the

	generic	generic+	user-exclusive	user-adaptive
Shoes-B	31.5 (0.13)	36.3 (0.14)	40.3 (0.15)	**43.6** (0.13)
SUN	34.3 (0.19)	47.3 (0.15)	51.9 (0.24)	**64.5** (0.16)

(a) Multi-attribute keyword search

	generic	generic+	user-exclusive	user-adaptive
Shoes-R	70.96 (0.12)	72.70 (0.10)	72.75 (0.14)	**74.70** (0.12)

(b) Relative attribute search feedback

Figure 6. Personalized image search accuracy

	generic	explicit	+contr	+trans
Shoes-R	70.96 (0.1)	72.58 (0.1)	**74.15** (0.1)	**74.34** (0.1)

Table 1. The benefit of inferring implicit user-specific labels

user was trying to find (higher is better). Again, our personalized search results are best, even notably stronger than the personalized user-exclusive model. To give a concrete sense of significance, our method ranks the target image 7 pages higher than the closest baseline, assuming a webpage fits 40 images per page. This result shows how our idea can improve prior systems for relative attribute search [13, 12].

Learning with inferred labels Finally, we validate our ideas to infer user-specific labels. They apply only to the relative attribute search scenario, so we test on Shoes-R. We replace half of the explicit user-specific labels used for adaptation with all the labels we infer using transitivity or contradictions. Table 1 shows that either inference method boosts search accuracy. The user's target image is on average ranked 6 pages higher using those "free" inferred labels compared to just explicit labels. Note we are getting comparable accuracy to our result in Fig. 6(b), but now with half the user-specific labels.

Conclusions and future work Our main contribution is the idea of adapting attributes to account for user-specific perception. Our approach accommodates both binary and relative properties, and makes it possible to leverage existing labeled datasets as a prior to regularize new user-specific models. Our results on two compelling datasets indicate that 1) people do indeed have varying shades of attribute meaning, 2) transferring generic models makes learning those shades more cost-effective than learning from scratch, and 3) accounting for the differences is essential in image search applications.

Our work suggests a number of interesting future directions. We plan to investigate extensions to detect when an attribute is perceived nearly the same by most users, to avoid requesting user-specific labels unnecessarily. Further, we will explore more ways to gauge internal consistency within a user's set of responses, since self-consistency is critical for adaptation. Finally, we will consider how to discover and exploit structure among multiple users, which could allow learning functions somewhere between monolithic and user-specific.

Acknowledgement This research was supported in part by ONR ATL grant N00014-11-1-0105.

References

[1] Y. Aytar and A. Zisserman. Tabula Rasa: Model Transfer for Object Category Detection. In *ICCV*, 2011.
[2] T. Berg, A. Berg, and J. Shih. Automatic Attribute Discovery and Characterization from Noisy Web Data. In *ECCV*, 2010.
[3] S. Branson, C. Wah, F. Schroff, B. Babenko, P. Welinder, P. Perona, and S. Belongie. Visual Recognition with Humans in the Loop. In *ECCV*, 2010.
[4] W. Curran, T. Moore, T. Kulesza, W.-K. Wong, S. Todorovic, S. Stumpf, R. White, and M. Burnett. Towards Recognizing "Cool": Can End Users Help Computer Vision Recognize Subjective Attributes or Objects in Images? In *IUI*, 2012.
[5] I. Endres, A. Farhadi, D. Hoiem, and D. Forsyth. The Benefits and Challenges of Collecting Richer Object Annotations. In *ACVHL*, 2010.
[6] A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth. Describing Objects by Their Attributes. In *CVPR*, 2009.
[7] J. Gauvain and C.-H. Lee. Maximum A Posterior Estimation For Multivariate Gaussian Mixture Observations Of Markov Chains. *IEEE Transactions on Speech and Audio Processing*, 2, 1994.
[8] B. Geng, L. Yang, C. Xu, and X.-S. Hua. Ranking Model Adaptation for Domain-Specific Search. *IEEE TKDE*, March 2010.
[9] B. Gong, Y. Shi, F. Sha, and K. Grauman. Geodesic Flow Kernel for Unsupervised Domain Adaptation. In *CVPR*, 2012.
[10] R. Gopalan, R. Li, and R. Chellapa. Domain Adaptation for Object Recognition: An Unsupervised Approach. In *ICCV*, 2011.
[11] T. Joachims. Optimizing Search Engines Using Clickthrough Data. In *KDD*, 2002.
[12] A. Kovashka and K. Grauman. Attribute Pivots for Guiding Relevance Feedback in Image Search. In *ICCV*, 2013.
[13] A. Kovashka, D. Parikh, and K. Grauman. WhittleSearch: Image Search with Relative Attribute Feedback. In *CVPR*, 2012.
[14] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar. Describable Visual Attributes for Face Verification and Image Search. *PAMI*, 2011.
[15] C. Lampert, H. Nickisch, and S. Harmeling. Learning to Detect Unseen Object Classes by Between-class Attribute Transfer. In *CVPR*, 2009.
[16] D. Parikh and K. Grauman. Relative Attributes. In *ICCV*, 2011.
[17] A. Parkash and D. Parikh. Attributes for Classifier Feedback. In *ECCV*, 2012.
[18] G. Pasi. Issues in Personalizing Information Retrieval. *IEEE Intelligent Informatics Bulletin*, 2010.
[19] G. Patterson and J. Hays. SUN Attribute Database: Discovering, Annotating, and Recognizing Scene Attributes. In *CVPR*, 2012.
[20] K. Saenko, B. Kulis, M. Fritz, and T. Darrell. Adapting Visual Category Models to New Domains. In *ECCV*, 2010.
[21] W. Scheirer, N. Kumar, P. Belhumeur, and T. Boult. Multi-Attribute Spaces: Calibration for Attribute Fusion and Similarity Search. In *CVPR*, 2012.
[22] A. Shrivastava, S. Singh, and A. Gupta. Constrained Semi-Supervised Learning using Attributes and Comparative Attributes. In *ECCV*, 2012.
[23] B. Siddiquie, R. Feris, and L. Davis. Image Ranking and Retrieval Based on Multi-Attribute Queries. In *CVPR*, 2011.
[24] M. Stark, M. Goesele, and B. Schiele. A Shape-based Object Class Model for Knowledge Transfer. In *ICCV*, 2009.
[25] Y. Tian and J. Zhu. Learning from Crowds in the Presence of Schools of Thought. In *KDD*, 2012.
[26] S. Tong and E. Chang. Support Vector Machine Active Learning for Image Retrieval. In *ACM Multimedia*, 2001.
[27] P. Welinder, S. Branson, S. Belongie, and P. Perona. The Multidimensional Wisdom of Crowds. In *NIPS*, 2010.
[28] J. Yang, R. Yan, and A. G. Hauptmann. Adapting SVM Classifiers to Data with Shifted Distributions. In *ICDM Workshops*, 2007.