Simultaneous Active Learning of Classifiers & Attributes via Relative Feedback

Arijit Biswas
University of Maryland, College Park
arijitbiswas87@gmail.com

Devi Parikh
Virginia Tech
parikh@vt.edu

Abstract

Active learning provides useful tools to reduce annotation costs without compromising classifier performance. However it traditionally views the supervisor simply as a labeling machine. Recently a new interactive learning paradigm was introduced that allows the supervisor to additionally convey useful domain knowledge using attributes. The learner first conveys its belief about an actively chosen image e.g. *"I think this is a forest, what do you think?". If the learner is wrong, the supervisor provides an explanation* e.g. *"No, this is too open to be a forest". With access to a pre-trained set of relative attribute predictors, the learner fetches all unlabeled images more open than the query image, and uses them as negative examples of forests to update its classifier. This rich human-machine communication leads to better classification performance. In this work, we propose three improvements over this set-up. First, we incorporate a weighting scheme that instead of making a hard decision reasons about the likelihood of an image being a negative example. Second, we do away with pre-trained attributes and instead learn the attribute models on the fly, alleviating overhead and restrictions of a pre-determined attribute vocabulary. Finally, we propose an active learning framework that accounts for not just the label- but also the attributes-based feedback while selecting the next query image. We demonstrate significant improvement in classification accuracy on faces and shoes. We also collect and make available the largest relative attributes dataset containing 29 attributes of faces from 60 categories.*

1. Introduction

Image classification is one of the core tasks in visual recognition, with many generic applications such as image search [25], as well as niche applications such as recognizing bird [3], animal [14] or leaf [12] species from images. These applications often involve a large number of classes. Learning based methods have achieved a lot of success at these tasks, but typically require a large amount of labeled training data. Collecting labels for images is tedious and expensive especially in large-scale niche applications, which has led to increasing interest in active learning methods [2, 7, 9, 11, 18, 20, 21]. These methods aim to solicit labels from the supervisor on a small but useful set of images, leading to classification performance similar to that of having labeled a larger but random set of images.

However, most existing active learning settings still view the supervisor as an entity to simply get labels from. The supervisor often has much more domain knowledge about an application at hand than just the label, that if communicated to the learner, may allow the learner to learn from even fewer examples. Parkash and Parikh [17] proposed the use of attributes to enable such communication. Attributes [1, 3, 4, 6, 11, 13–16, 19, 22–24] are mid-level semantic concepts such as "furry", "natural", etc. that are shared across related categories. In their work [17], at each iteration in active learning, the learner identifies an image that it would like labeled. However instead of simply demanding the label for the image, the learner first determines its current belief about the image, and conveys it to the supervisor. For instance the learner says, "I think this is a forest, what do you think?". If the learner is wrong, the supervisor identifies a reason that conveys to the learner why it is wrong. The supervisor may say "No, this is not a forest image, because it is too open to be a forest image". The learner is assumed to have access to a set of pre-trained relative attribute [16] predictors. In this case, it uses the "openness" attribute model to find all images from an unlabeled pool that are even more open than the selected query image, and assumes that those must not be forests either. This provides the learner with many more (negative) examples of forest which it uses to update its classifier. Such a rich communication between the supervisor and the learner was shown to lead to faster learning with fewer labeled examples [17]. In this paper, we improve upon [17] in several ways.

First (Section 3.1), instead of simply assuming that all images in the above example more open than the query image are not forests, we intelligently estimate the likelihood of the images not being a forest. This allows us to be robust to potential inaccuracies in the attribute predictors. Images that are significantly more open than the query image are more likely to not be forests than images that are barely

1063-6919/13 $26.00 © 2013 IEEE
DOI 10.1109/CVPR.2013.89

more open. Moreover, if an image is deemed to not be a forest due to several reasons provided by the supervisor across the active learning iterations, this increases our confidence in it not being a forest. We show that a simple weighting scheme that accounts for these intuitions significantly improves performance.

Second (Section 3.2), instead of requiring the learner to have access to a pre-trained set of relative attribute models, we allow the learner to learn the attribute models on the fly, simultaneously with the category models. Requiring access to a domain relevant vocabulary of attributes and corresponding attribute models is very restrictive and can be time consuming to acquire. Instead, we leverage the following observation: when the supervisor says "This query image is too open to be a forest", it not only conveys information about forests, but it also conveys information about the notion of openness. If in previous (or future) iterations the learner has already collected a few example forest images, this feedback from the supervisor indicates that those images must be less open than the query image. This information can be used to build or update the openness attribute model. This allows the supervisor to introduce new attributes as and when necessary without being constrained by a pre-determined vocabulary. It also alleviates the need to collect training data to train a pre-determined set of attributes. The learner and supervisor start with nothing but an unlabeled pool of images, and learn both the categories and attribute models from scratch. This reduced overhead and unconstrained setting truly makes the proposed approach easily and widely accessible.

Finally (Section 3.3), we propose a novel active learning framework that selects the most useful image at each iteration, keeping in mind the specific form of learner-supervisor interaction our setup allows. The attributes-based feedback provided by the supervisor is propagated to many unlabeled images by the learner. Our estimate of the reduction in entropy of the system when a query image is selected accounts for this propagation and results in more efficient learning.

We demonstrate our approach on two domains: faces[1] and shoes. In both cases, we assume that if the learner is correct about its belief about a query image, the supervisor can confirm this. But if the learner is incorrect, the supervisor can indicate that and provide an explanation, but can not identify the correct label of the image. This is relevant in various scenarios like face recognition for surveillance or niche applications like bird [3], animal [14] or leaf [12] species recognition that require expert knowledge. The supervisor can look at example images of the claimed category and easily verify if the claim is correct or not. But the supervisor may not have the time to navigate through a long list of categories, nor have the expertise to do so. Attributes on the other hand may not require as much expert knowledge. For instance, say the learner incorrectly claims a teenager in a surveillance video is Kirabo Smith, and shows previously labeled example images of Kirabo Smith. The supervisor can see from the examples that Kirabo Smith is actually a senior citizen. The supervisor may not know who the person in the video is, but can easily say that the person is too young to be Kirabo Smith.

Most relevant to our work is that of Parkash and Parikh [17]. They introduced the notion of using attributes for providing classifiers feedback. Authors provided a great overview of its relationships to existing trends in computer vision. We do not replicate that discussion here for space considerations. Our work innovates over theirs in three significant ways as described above. We now provide an overview of the use of attributes for classifier feedback as in [17] and then describe our proposed approach.

2. Preliminaries

A supervisor is teaching a machine visual concepts. As learning aid, there is a pool of unlabeled images that the supervisor will label over the course of the learning process for the learner to learn from. At each iteration, the learner picks an image from the unlabeled pool that it finds would be most useful to have labeled. It communicates its own belief to the supervisor in the form of a predicted label for this image. The supervisor either confirms or rejects this label. If rejected, the supervisor communicates an explanation using attributes for why the learner's belief was wrong. The learner incorporates both the label-feedback (whether accepted or rejected) and the attributes-based explanation (if rejected) to update its models. And the process continues.

A binary classifier $h_k(\boldsymbol{x}), k \in \{1 \dots K\}$ is trained for each of the K categories to be learnt. At any point during the learning process, there is an unlabeled pool of images U and a training set $T_k = \{(\boldsymbol{x}^k, y^k)\}$ of labeled images for each classifier. The labeled images $\boldsymbol{x}^k$ can lie in any feature space. The class labels are binary $y^k \in \{+1, -1\}$. In our implementation we use linear SVMs as the binary classifier with probability estimates as output. These probability estimates are normalized across the K classifiers to compute $p_k(\boldsymbol{x}_i)$, the probability that image $\boldsymbol{x}_i$ belongs to class k. The image with the highest entropy of the class distribution p_k is chosen to be the query image $\boldsymbol{x}^q$. We will describe our novel criterion for picking the query image in Section 3.3. We initialize the system with one labeled example from each class. Note that a new class can be added to the system at any time in the learning process simply by providing one positive labeled example.

There are two parts to the supervisor's response to the query image. The first is label-based feedback where the supervisor confirms or rejects the learners predicted label for the actively chosen image instance $\boldsymbol{x}^q$. And the sec-

[1] As part of our experimental setup, we collected and have made publicly available annotations for 29 relative attributes on 60 celebrity faces [13], the largest relative attributes dataset to date.

ond is an attributes-based explanation that is provided if the learners predicted label was incorrect and thus rejected. We now discuss how the attributes-based feedback is incorporated given that the learner has access to a set of attribute predictors. We will describe our approach of learning these models on the fly in Section 3.2.

2.1. Incorporating Attribute-based Explanation

Let's say the learner incorrectly predicts the label of actively chosen image $\boldsymbol{x}^q$ to be l. The supervisor identifies an attribute a_m that he thinks is most appropriate to explain to the learner why $\boldsymbol{x}^q$ does not belong to l. There are two simple forms of explanation. The supervisor can either say "$\boldsymbol{x}^q$ is too a_m to be l" or "$\boldsymbol{x}^q$ is not a_m enough to be l", whichever be the case.

In the former case, the learner computes the strength of a_m in $\boldsymbol{x}^q$ as $r_m(\boldsymbol{x}^q)$, where r_m is a attribute strength predictor for attribute a_m (Section 2.2). The learner identifies all images in the currently unlabeled pool of images U with attribute strength of a_m more than $r_m(\boldsymbol{x}^q)$. Clearly, if $\boldsymbol{x}^q$ is too a_m to be l, all images depicting a higher strength of a_m should not be l either. Hence the training data for class l (T_l) is updated to be $\hat{T}_l = T_l \cup \{(\boldsymbol{x}, -1)\}\ \forall \boldsymbol{x} \in U\ s.t.\ r_m(\boldsymbol{x}) \geq r_m(\boldsymbol{x}^q)$. Similarly, for the latter form of feedback, $\hat{T}_l = T_l \cup \{(\boldsymbol{x}, -1)\}\ \forall \boldsymbol{x} \in U\ s.t.\ r_m(\boldsymbol{x}) \leq r_m(\boldsymbol{x}^q)$. The category-label information is thus propagated to other images in U aside from just $\boldsymbol{x}^q$. We will describe our novel weighting scheme in Section 3.1 that softens this propagation.

2.2. Attribute Predictors

The feedback provided by the supervisor relates the query image actively selected by the learner to the category the learner believes the query image is from. Hence relative attributes [16] are a natural choice. We now describe how these relative attribute predictors are trained offline. We will describe our approach to learning these attribute models on the fly in Section 3.2.

Suppose we have a vocabulary of M attributes $A = \{a_m\}$. These attribute models are learnt using a set of training images $I = \{i\}$ represented in $\mathbb{R}^n$ by feature-vectors $\{\boldsymbol{x_i}\}$. For each attribute, we are given as supervision a set of ordered pairs of images $O_m = \{(\boldsymbol{x}_i, \boldsymbol{x}_j)\}$ such that $(\boldsymbol{x}_i, \boldsymbol{x}_j) \in O_m \implies \boldsymbol{x}_i \succ \boldsymbol{x}_j$, *i.e.* image i has a stronger presence of attribute a_m than j. We wish to learn a ranking function $r_m(\boldsymbol{x_i}) = \boldsymbol{w_m^T x_i}$ for $m = 1, \ldots, M$, such that the maximum number of the following constraints is satisfied: $\forall (\boldsymbol{x}_i, \boldsymbol{x}_j) \in O_m, \boldsymbol{w_m^T x_i} > \boldsymbol{w_m^T x_j}$. A relaxed version of this NP hard problem is solved using a large margin learning to rank formulation similar to that of Joachims [8]. With this, given any image $\boldsymbol{x}$ in our pool of unlabeled images U, we can compute the relative strength of each of the M attributes as $r_m(\boldsymbol{x}) = \boldsymbol{w_m^T x}$.

2.3. Incorporating Label-based Feedback

As described earlier, we consider a scenario where the supervisor can verify if the classifier's prediction of $\boldsymbol{x}^q$ is correct or not. But when incorrect, it would be very time consuming for or beyond the expertise of the supervisor to seek out the correct category label to provide as feedback. Hence the supervisor only confirms or rejects the prediction, and does not provide a correction if rejecting it. While our approach is general, in our implementation we assume that each image belongs to exactly one category. Hence, if the classifier's prediction l is confirmed we have $\hat{T}_l = T_l \cup \{(\boldsymbol{x}^q, +1)\}$, and $\hat{T}_n = T_n \cup \{(\boldsymbol{x}^q, -1)\}\ \forall\ n \neq l$. However, if the classifier's prediction is rejected, we only have $\hat{T}_l = T_l \cup \{(\boldsymbol{x}^q, -1)\}$.

3. Proposed Approach

We now describe our proposed approach that improves upon the set-up presented in the previous section.

3.1. Weighting Scheme for Negative Examples

As described above, Parkash and Parikh [17] use the attributes-based feedback to fetch unlabeled images and add them as negative examples, all with the same weight. However, not all images are equally likely to be negative. In our running example, if the query image is too open to be a forest, images that are significantly more open than the query image are more likely to not be forests than images that are barely more open than the query image. Also, as iterations go by, if the same image is deemed to not be a forest due to several reasons provided by the supervisor on different query images, we should be more confident of the image not being a forest. If one query is too open to be a forest and another future query is not natural enough to be a forest, an image that is both more open than the first query and less natural than the second is more likely to not be a forest than an image that is more open than the first query but also more natural than the second. We develop a straightforward weighting scheme that accounts for these intuitions.

Let $w_Q^l(\boldsymbol{x})$ capture the likelihood at iteration Q that any unlabeled image $\boldsymbol{x}$ does not belong to class l. It is computed using attributes-based feedback accumulated over all past iterations (indexed by q) where the classifier incorrectly predicted the label of the corresponding query image $\boldsymbol{x}^q$ to be l. Without loss of generality, we will assume that for each of these iterations, the supervisor gave the explanation: "$\boldsymbol{x}^q$ is too a_{m^q} to be l". Notice that the attribute selection may have been different at each iteration, indexed by m^q. Then

$$w_Q^l(\boldsymbol{x}) = \sum_{q=1}^{Q} n_q(\boldsymbol{x}) \tag{1}$$

where $n_q(\boldsymbol{x})$ is 0 if:

- l was not the predicted label for $\boldsymbol{x}^q$ at iteration q OR

- l was the predicted label but correctly so (*i.e.* no attributes-based feedback was provided) OR
- $\boldsymbol{x}$ does not have more a_{m^q} than $\boldsymbol{x}^q$ *i.e.* $r_{m^q}(\boldsymbol{x}) < r_{m^q}(\boldsymbol{x}^q)$

otherwise $n_q(\boldsymbol{x})$ is the number of images that lie between $\boldsymbol{x}^q$ and $\boldsymbol{x}$ when all the images (labeled or unlabeled) are sorted by r_{m^q}. If $n_q(\boldsymbol{x})$ is large, the weight corresponding to $\boldsymbol{x}$ is high. $n_q(\boldsymbol{x})$ can also be defined to be the difference in attribute scores between the two images *i.e.* $n_q(\boldsymbol{x}) = r_{m^q}(\boldsymbol{x}) - r_{m^q}(\boldsymbol{x}^q)$. However, since the attribute predictors are trained as ranking functions and not regressors, the difference in scores may be less meaningful.[2] At each iteration, the weights $w_Q^l(\boldsymbol{x})$ for all images $\boldsymbol{x}$ in the dataset are normalized to lie between 0 and 1 for each class l. The weights of the images that have been labeled by the supervisor via labeled-based feedback are always set to the maximum, which is 1. The weights of these instances can be fed into standard SVM solvers.

3.2. Learning Attribute Models On The Fly

We now describe our approach to learning the attribute models on the fly as opposed to using pre-trained attribute predictors as in [17]. Recall that an attribute predictor r_m for an attribute a_m is learnt by using human annotated pairs of images $O_m = \{(\boldsymbol{x}_i, \boldsymbol{x}_j)\}$ such that $(\boldsymbol{x}_i, \boldsymbol{x}_j) \in O_m \implies \boldsymbol{x}_i \succ \boldsymbol{x}_j$ *i.e.* image $\boldsymbol{x}_i$ displays a stronger presence of attributes a_m than image $\boldsymbol{x}_j$. We use the following approach to learn attribute predictors on the fly. At any iteration, if the supervisor says "$\boldsymbol{x}^q$ is too a_m to be l", then the learner fetches all images labeled as l, and appends O_m with additional constraints $\hat{O}_m = O_m \cup \{(\boldsymbol{x}^q, \boldsymbol{x}_j)\}$ $s.t.$ $\boldsymbol{x}_j$ has been labeled as l. For instance, if the supervisor says, "this query image is too open to be a forest", then the learner can fetch all images thus far labeled as forests and realize that all these forest images must be less open than the query image. Similarly, if the supervisor says "$\boldsymbol{x}^q$ is not a_m enough to be l", then $\hat{O}_m = O_m \cup \{(\boldsymbol{x}_j, \boldsymbol{x}^q)\}$ $s.t.$ $\boldsymbol{x}_j$ is labeled as l. We make some notes: 1) If an image in the future is labeled by the supervisor to be forest, O_m will be appended accordingly. 2) As the attribute models are updated at each iteration the weights described in Section 3.1 are recomputed using the updated attribute models. 3) The additional constraints with which the existing ranker already agrees are less likely to influence the updated ranker. Those that are currently violated will play a critical role in its update.

Learning attributes on the fly gives the supervisor flexibility to use whichever attribute he deems fit, and not be severely restricted by a pre-determined vocabulary of attributes. The supervisor can introduce a new attribute at any time in the learning process. Our approach is general and also allows access to a set of pre-trained attributes, which can then be updated on the fly. Moreover, the form of the attributes-feedback conveniently matches the supervision required by a learning to rank formulation to train relative attribute models, allowing us to use it simultaneously as feedback for training classifiers and as annotation for training relative attribute models.

[2] When $\boldsymbol{x}$ is just a bit more open than $\boldsymbol{x}^q$ and there are many images similar to each other that all fall between $\boldsymbol{x}$ and $\boldsymbol{x}^q$, the difference in scores may be more reliable. Empirically, using the difference in scores gives similar but slightly worse results than using the number of images.

3.3. Active Selection of Images

We now describe our novel active selection approach to select the query image at a given iteration. The idea behind active selection is to select an image such that the supervisor's response to it is likely to reduce the uncertainty or entropy of the system the most. The traditional criterion of picking the image with the most entropy of its class distribution p_k (also used in [17]) does not incorporate our relative attributes-based feedback setup. Our contribution is to account for the feedback while efficiently computing the expected reduction in entropy of the system when a query image is labeled. The current entropy of the system is defined to be:

$$H = -\sum_{i=1}^{N}\sum_{k=1}^{K} p_k(\boldsymbol{x}_i)\log(p_k(\boldsymbol{x}_i)) \tag{2}$$

where $p_k(\boldsymbol{x}_i)$ is the probability of image $\boldsymbol{x}_i$ belonging to class k as estimated by the classifier h_k and N is the number of images in the currently unlabeled set U. When an image $\boldsymbol{x}_i$ is chosen to be the query image, there are two possible responses from the supervisor. The first, with probability p^0 is to accept the label predicted by the classifier for $\boldsymbol{x}_i$ (*i.e.* $\boldsymbol{x}_i$ gets labeled with its correct label). Let the resultant entropy of the system be H^0. The second, with probability $p^1 = 1 - p^0$ is to reject the label and provide an attributes-based feedback. The latter has $2M$ options corresponding to each of the M attributes and "too" or "not enough" reasons. Let's say given a rejection response, the chances of the supervisor picking any of the M attributes with the "too" response is p_m^{1+} and with the "not enough" response is p_m^{1-}. Let the resultant entropy of the system be H_m^{1+} and H_m^{1-} respectively. Hence, the expected change in entropy $\Delta H(i)$ of the system when $\boldsymbol{x}_i$ is selected as the query image is

$$H - \left(p^0 H^0 + p^1\left(\sum_{m=1}^{M} p_m^{1+} H_m^{1+} + \sum_{m=1}^{M} p_m^{1-} H_m^{1-}\right)\right) \tag{3}$$

We wish to find $\boldsymbol{x_i}$ with the largest value of $\Delta H(i)$. We estimate p^0, the probability that the classifier's predicted label l for $\boldsymbol{x}_i$ is correct and hence accepted by the supervisor to be $p_l(\boldsymbol{x}_i)$. As stated earlier, $p^1 = 1 - p^0$. To estimate p_m^{1+}, the probability that the supervisor will provide feedback "$\boldsymbol{x}_i$ is too a_m to be l", we use the following intuition. If the strength of a_m in $\boldsymbol{x}_i$ is much larger than

its strength in all images labeled as l so far i.e. $r_m(\boldsymbol{x}_i)$ is much larger than $r_m(\boldsymbol{x}_j)$ where $\boldsymbol{x}_j$ are images labeled as l, then p_m^{1+} will be high. More specifically, say we sort all the images by their attribute values $r_m(\boldsymbol{x})$ in ascending order. Say $n_m(\boldsymbol{x})$ denotes the rank of image $\boldsymbol{x}$ in the sorted list and N_l is the number of images labeled as l so far. If $\boldsymbol{x}_i$ falls very far to the right of all the images labeled as l, $\tilde{n} = \sum_{j=1}^{N_l}(n_m(\boldsymbol{x}_i) - n_m(\boldsymbol{x}_j))$ will be a large positive number, and p_m^{1+} is high. Similarly, if $\boldsymbol{x}_i$ falls very far to the left *i.e.* $\tilde{n}$ is a large negative number, p_m^{1+} is small. And if $\boldsymbol{x}_i$ lies in the middle of the all the images labeled as l *i.e.* $\tilde{n}$ is close to 0, p_m^{1+} is moderate. Similar reasoning holds for p_m^{1-}. Hence, we compute the $2M$ values corresponding to $\tilde{n}$ and their negatives, linearly shift them such that the smallest value is 0, and then divide by the sum to get a distribution over the $2M$ choices.

We now look at estimating the resultant entropies H^0, H_m^{1-} and H_m^{1+} in Equation 3. H^0 is simply H reduced by the entropy of $\boldsymbol{x}_i$ which is $-\sum_{k=1}^{K} p_k(\boldsymbol{x}_i)\log(p_k(\boldsymbol{x}_i))$, because after being labeled as class l, its entropy is 0. Estimating H_m^{1-} and H_m^{1+} is more interesting. When the label is rejected, the supervisor provides attributes-based feedback, which is transferred to many images as negative labels. If the feedback is "$\boldsymbol{x}_i$ is too a_m to be l", then for all images $\boldsymbol{x}_{i'}$ s.t. $r_m(\boldsymbol{x}_{i'}) > r_m(\boldsymbol{x}_i)$, the corresponding class probabilities $p_l(\boldsymbol{x}_{i'})$ would be set to zero[3], and their entropies can be recomputed to determine the changed entropy. However, note that when the supervisor provides this feedback, the relative attribute model r_m is also updated, and so the only way to determine the set $\{\boldsymbol{x}_{i'}\}$ s.t. $r_m(\boldsymbol{x}_{i'}) > r_m(\boldsymbol{x}_i)$ is to re-train r_m. Notice that at each iteration, Equation 3 has to be computed for all N unlabeled images in the dataset. Hence, this brute-force approach would require us to train 2NM ranking functions at each iteration, which would be prohibitively expensive for any reasonably sized dataset.

To reduce the computational burden, we propose an approximate but faster approach for computing H_m^{1-} and H_m^{1+}. Intuitively, instead of having to train 2NM ranking functions, we simply train 2CM ranking functions, where $C << N$, and is obtained by a meaningful clustering of the N images. At a very high-level, this is similar to [5] that clusters images for image retrieval to ensure a good coverage of the entire feature space while simultaneously improving efficiency. Our clustering relies on the following intuition. Anytime the supervisor gives the learner feedback of the form "$\boldsymbol{x}_i$ is too a_m to be l", the added constraints that are obtained to update r_m are of the form $(\boldsymbol{x}_i \succ \boldsymbol{x}_j)$ s.t. $\boldsymbol{x}_j$ is labeled as l. Recall that some of these constraints may be satisfied by the existing ranker, while others may be violated. The latter tend to have more impact on the ranker's update. Notice that all candidate query images $\boldsymbol{x}_i$ that are predicted by the classifier to be from class l and lie between

[3]Weights (Section 3.1) are ignored to simplify entropy computation.

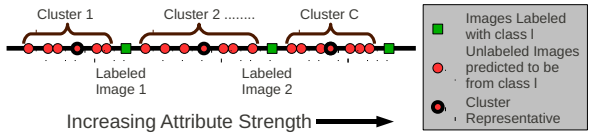


Figure 1: Our clustering approach for fast active learning. Unlabeled images are clustered along each relative attribute using boundaries marked by labeled images of each class. A ranking function is trained for each cluster representative instead of each unlabeled image.

two consecutively ranked labeled images $\boldsymbol{x}_j$ will have the same set of violated constraints, and will presumably lead to similar updated ranking functions. Hence, instead of learning a new ranking function for all N images, we learn a new ranking function only for each group of images that are predicted by the classifier to be from the same class and that fall between two consecutively ranked labeled images of that class. See Figure 1. Each cluster is represented with the median image along the attribute, and a new ranking function is trained using that image instead of each candidate $\boldsymbol{x}_i$ *i.e.* each unlabeled image. All images falling in the same cluster thus share the same ranking functions to compute H_m^{1-} and H_m^{1+} in Equation 3. In our experiments, we find that as a result of our clustering based approximation, we were required to train only $5-7\%$ of ranking functions, while maintaining accuracy (see Figure 4b).

4. Experiments

We experiment with two different domains: faces and shoes. For faces, we use two subsets of the Public Figures Face Dataset (PubFig) [13]: Pubfig-772-8 and Pubfig-900-60. The first is the same as that used by Parkash and Parikh [17] containing 772 images from 8 categories described by 512-d gist concatenated with 30-d color histogram features. The second contains 900 images from 60 categories in the development set of PubFig. This is a significantly more challenging dataset (not used in [17]), and the simple gist and color features do not suffice. Instead, we use the 73-d more sophisticated face features provided by Kumar *et al.* [13] that capture various properties of the faces and were found to provide state-of-the-art face verification performance. For shoes (Shoes-750-10) we use a subset of 750 images from 10 classes of the shoes dataset of Berg *et al.* [1] also used in [10]. As in [10], 960-d gist features are concatenated with 30-d color histogram features.

4.1. Collecting Attributes-based Feedback

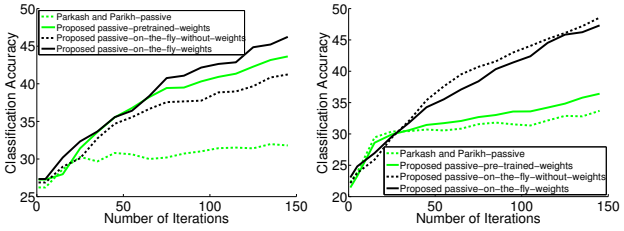
To allow for extensive quantitative evaluations while still using feedback from real users, we collect exhaustive attributes-based feedback offline using human subjects on Amazon Mechanical Turk (MTurk) (as in [17]). Note that the supervisor provides feedback every time a query image

is mis-classified, and this feedback depends on the query image and the predicted label. This would require us to collect (number-of-images $\times$ number-of-categories) number of feedback statements offline. To restrict the amount of data to be collected, we make the simplifying assumption that the feedback depends only on the true label of the query image and the predicted label. This now requires us to collect (number-of-categories $\times$ number-of-categories) feedback statements.

To collect this data, we show 10 MTurk workers example images from a pair of categories. We experiment with two different interfaces. One is where workers are asked to describe in free-form text (using a word or a phrase), the most obvious difference between the categories[4]. The second is where workers are shown a list of attributes one at a time and asked whether the first of the two categories has more of the attribute, less of the attribute, or a similar presence as the second.[5] The attribute with most people agreeing on one category having a stronger presence of the attribute than the other (and fewest people saying the opposite) is the attribute corresponding to the most obvious difference between the two categories. Note that the list of attributes simulates the vocabulary of attributes that the system would end up with at the end of the learning process. In our experiments for testing the system's ability to learn the attributes on the fly, the system does *not* have access to this list ahead of time. To avoid language processing issues that arise when consolidating free-form text from multiple users, we use the latter interface for most of our experiments. Towards the end we show an experiment comparing the two interfaces. In a real setting involving multiple supervisors, we envision a hybrid of the two interfaces: if one of the supervisors introduces a new attribute, it dynamically shows up on the list of available attributes for all supervisors to use. We note that our attribute annotations for the PubFig-900-60 dataset is the largest relative attribute dataset to the best of our knowledge and is publicly available on our webpage.

4.2. Results

We split each of the three datasets into training and testing sets. The training set (usually 65-75% of the total dataset) is used as the unlabeled data that will be labeled by the supervisor with label- and attributes-based feedback over the course of the learning iterations. We evaluate the classifiers every five iterations on the held-out test set. We report average accuracies across 20 random train/test splits. In Table 1 we list the various algorithms we evaluated. The different comparisons allow us to evaluate the role of attributes-based feedback, of the proposed weighting scheme, of the on-the-fly attribute models and of the proposed query image selection approach as compared to the traditional max-entropy active selection approach.



(a) Shoes-750-10 (b) Pubfig-772-8

Figure 2: Impact of our proposed weighting scheme and of learning attribute models on the fly on performance

4.2.1 Weighting Scheme

Figure 2 demonstrates the impact of our proposed weighting scheme (solid vs dashed). Overall, we find that the weighting scheme improves performance significantly, especially for pre-trained attributes.

4.2.2 Attribute Models Trained On The Fly

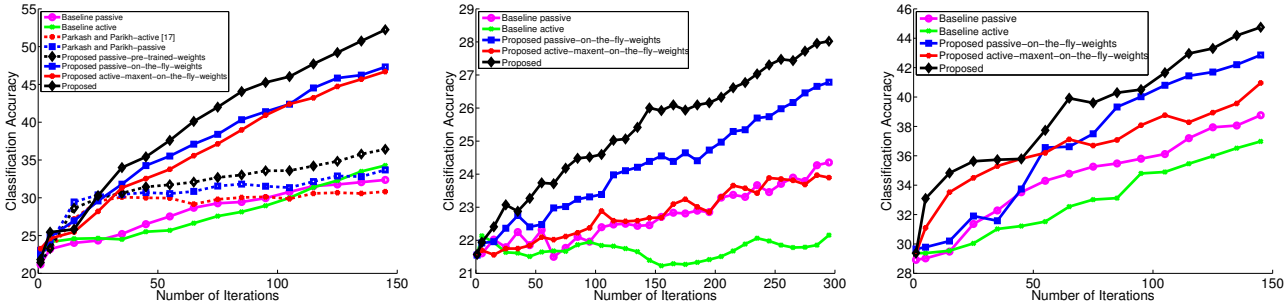
Figure 2 also demonstrates the impact of learning the attribute models on the fly (black vs. green). While the motivation behind learning the attribute models on the fly was added flexibility and reduced over-head of pre-training the attributes, surprisingly, we found that they also improve accuracies significantly. Our analysis revealed that the attribute models learnt on the fly are biased towards adding fewer but more informative negative examples to the classifiers as compared to the pre-trained attribute models. For example, after 300 iterations on the Pubfig-900-60, pre-trained attributes models add on average 415 negative examples while on-the-fly models add 210. Of these, $\sim 1.3\%$ are incorrect (i.e. are in fact positive examples) when using pre-trained attribute model, but only $\sim 0.61\%$ are incorrect when using attribute models trained on the fly. Perhaps more importantly, $\sim 80\%$ of the true positive images are added to the negative side when using pre-trained attributes, while only $\sim 15\%$ for attributes trained on the fly. Further, we evaluated the accuracy of the attribute models at predicting the relative attribute strength in pairs of images. As seen in Figure 4a, we find that the pre-trained attribute models are in fact more accurate on average at predicting the relative order of images. While being worse attribute predictors, the models trained on the fly are better catered towards providing classifier-feedback. Due to their superiority, from here on, we only show comparisons to approaches using attribute models trained on the fly. Note that the passive approach of Parkash and Parikh [17] with at-

[4]We assume that the supervisor is likely to comment on the attribute that makes the predicted category of an image most different from its true category, allowing us to simplify the question posed to MTurk workers.

[5]For PubFig-772-8 we used a list of the 11 attributes used in [16, 17] and for PubFig-900-60 we used a list of 29 attributes capturing the same concepts as those of Kumar *et al*. [13] such as age, gender, race, face-shape, accessories and make-up, etc. For shoes, we used the 10 attributes introduced by Kovashka *et al*. [10] such as colorful, formal, sporty, etc.

Name of Method	Attribute Feedback Used	Weighting Scheme Used	Attributes Learned	Query Image Selection
Baseline passive	no	N/A	N/A	random
Baseline active	no	N/A	N/A	max-entropy
Parkash & Parikh -active [17]	yes	no	pre-trained	max-entropy
Parkash & Parikh -passive	yes	no	pre-trained	random
Proposed passive-pre-trained-weights	yes	yes	pre-trained	random
Proposed passive-on-the-fly-without-weights	yes	no	on-the-fly	random
Proposed passive-on-the-fly-weights	yes	yes	on-the-fly	random
Proposed active-maxent-on-the-fly-weights	yes	yes	on-the-fly	max-entropy
Proposed	yes	yes	on-the-fly	proposed (Section 3.3)

Table 1: Summary of the algorithms that we compare



(a) Pubfig-772-8 (b) Pubfig-900-60 (c) Shoes-750-10

Figure 3: Comparing our proposed approach to various baselines (Table 1)

tribute models trained on the fly is the same as "Proposed passive-on-the-fly-without-weights" in Table 1.

4.2.3 Active Selection

We now evaluate the benefit of our proposed active learning approach. In Figure 3a, we report exhaustive comparisons on PubFig-772-8 (also used in [17]), and show the most critical comparisons on the remaining two datasets in Figures 3b and 3c. We find that attributes-based feedback improves performance significantly. Perhaps somewhat surprisingly, we find that randomly selecting query images performs better than using the maximum entropy criterion used in traditional active learning. We suspect this to be the case due to the nature of our application where the supervisor only accepts or rejects a label for the query image, as opposed to providing the correct label as in traditional labeling tasks. As the classifiers get stronger (around iteration 125 in Figure 3a), the active approach starts to get better than the passive approach. However, we see that our proposed active learning approach significantly outperforms both the traditional active learning criterion and the passive approach on all three datasets.

4.2.4 Comparison to Brute Force

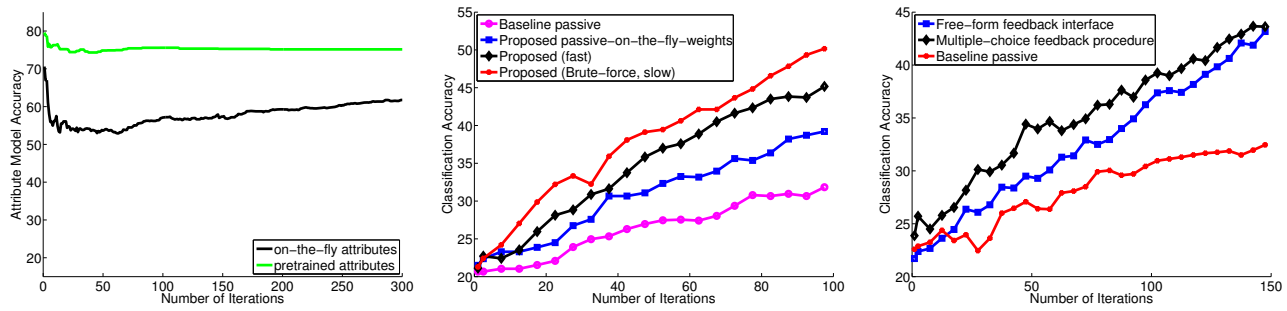
We now analyze how much performance we are loosing due to our clustering approximation (Figure 1). We experiment with a small unlabeled dataset of 80 images from 8 classes in PubFig-772-8. We run the brute-force active selection scheme that involves training all 2NM ranking functions at each iteration and compare it to our proposed approach that requires training 2CM ranking functions. The value of C increases as more images are labeled, making the process most efficient early on when the classifier gains more from each labeled example. On average across the learning iterations, our approach was 4 times faster than brute force on this small dataset without significant loss in performance (Figure 4b). The speed-up factor ($\sim \frac{N}{C}$) was $15 - 20$ on our other larger datasets, and goes up linearly with the average number of images per class in the unlabeled dataset.

4.2.5 Free-form Feedback

As described in Section 4.1, we also collected attributes-based feedback using a free-form interface for PubFig-772-8. The attributes collected were manually grouped (*e.g.* young, old, age, etc. all refer to the same attribute) leading to a final vocabulary of 8 attributes at the end of the learning process. This set-up mimics the scenario where the supervisor can truly use any attribute on the fly. Resultant accuracies are seen in Figure 4c. For sake of comparison, we also show accuracies using the second interface described in Section 4.1 that used annotations on a list of 11 attributes.

4.2.6 Efficient Use of User's Effort

If all the training images *i.e.* 532, 600 and 500 images from Pubfig-772-8, Pubfig-900-60 and Shoes-750-10 respectively were labeled, the classification accuracy on the three datasets would be 84%, 48% and 68%. A labeled image corresponds to a scenario where 1 of the K categories



(a) Quality of attribute predictors (b) Comparison to brute force (c) Free-form attributes-feedback

Figure 4: (a) Attribute models (for PubFig-60-900) learnt on the fly are worse attribute models per say, but are better suited for providing classifier-feedback than pre-trained attribute models (Figure 2). At each iteration, accuracies are shown only for the attributes introduced in the vocabulary by the user so far. (b) Our clustering-based fast active learning approach does not perform significantly worse than the brute-force version of our approach which would be prohibitively slow. (c) A comparison between two interfaces for collecting attributes-based feedback (Section 4.1)

has been identified to be the correct label *i.e.* $K-1$ labels (7, 59 and 9 respectively) have been eliminated for each training image. This corresponds to a total of 3724, 35400 and 4500 labels that need to be eliminated for each dataset. At the end of our 150, 300 and 150 iterations (Figure 3), we find that we have eliminated on average only 13% of these labels, but achieve accuracies that are 63%, 58% and 65% of the accuracy possible using a 100% of labeled training images. This demonstrates efficient use of the user's effort.

5. Conclusion

We introduced three innovations to enhance the use of relative attributes in providing classifiers feedback. We introduced a weighting scheme that intelligently reasons about the likelihood of any unlabeled image being a negative example for a category. We also learn attribute models on the fly, which not only provides increased flexibility to the supervisor with less overhead of pre-training attribute predictors, but also leads to significant improvements in classifier performance. Finally, we introduced a novel active learning criterion that accounts for the specific form of relative attributes-based feedback, leading to improved performance with significantly fewer labeled examples.

Acknowledgment: This work was supported in part by NSF IIS-1115719.

References

[1] T. Berg, A. Berg, and J. Shih. Automatic attribute discovery and characterization from noisy web data. In *ECCV*, 2010.
[2] A. Biswas and D. W. Jacobs. Active image clustering: Seeking constraints from humans to complement algorithms. In *CVPR*, 2012.
[3] S. Branson, C. Wah, B. Babenko, F. Schroff, P. Welinder, P. Perona, and S. Belongie. Visual recognition with humans in the loop. In *ECCV*, 2010.
[4] A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth. Describing objects by their attributes. In *CVPR*, 2009.
[5] M. Ferecatu and D. Geman. Interactive search for image categories by mental matching. In *ICCV*, 2007.
[6] V. Ferrari and A. Zisserman. Learning visual attributes. In *NIPS*, 2007.
[7] P. Jain and A. Kapoor. Active learning for large multi-class problems. In *CVPR*, 2009.
[8] T. Joachims. Optimizing search engines using clickthrough data. In *KDD*, 2002.
[9] A. J. Joshi, F. M. Porikli, and N. P. Papanikolopoulos. Multi-class active learning for image classification. In *CVPR*, 2009.
[10] A. Kovashka, D. Parikh, and K. Grauman. Whittlesearch: Image search with relative attribute feedback. In *CVPR*, 2012.
[11] A. Kovashka, S. Vijayanarasimhan, and K. Grauman. Actively selecting annotations among objects and attributes. In *ICCV*, 2011.
[12] N. Kumar, P. N. Belhumeur, A. Biswas, D. W. Jacobs, W. J. Kress, I. Lopez, and J. V. B. Soares. Leafsnap: A computer vision system for automatic plant species identification. In *ECCV*, 2012.
[13] N. Kumar, A. Berg, P. Belhumeur, and S. Nayar. Attribute and simile classifiers for face verification. In *ICCV*, 2009.
[14] C. H. Lampert, H. Nickisch, and S. Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *CVPR*, 2009.
[15] D. Parikh and K. Grauman. Interactively building a discriminative vocabulary of nameable attributes. In *CVPR*, 2011.
[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] B. Settles, M. Craven, and S. Ray. Multiple-instance active learning. In J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, editors, *NIPS*, 2007.
[19] A. Shrivastava, S. Singh, and A. Gupta. Constrained semi-supervised learning using attributes and comparative attributes. In *ECCV*, 2012.
[20] B. Siddiquie and A. Gupta. Beyond active noun tagging: Modeling contextual interactions for multi-class active learning. In *CVPR*, pages 2979–2986. IEEE, 2010.
[21] S. Vijayanarasimhan and K. Grauman. Large-scale live active learning: Training object detectors with crawled data and crowds. In *CVPR*, 2011.
[22] G. Wang and D. Forsyth. Joint learning of visual attributes, object classes and visual saliency. In *ICCV*, 2009.
[23] G. Wang, D. Forsyth, and D. Hoiem. Comparative object similarity for improved recognition with few or no examples. In *CVPR*, 2010.
[24] J. Wang, K. Markert, and M. Everingham. Learning models for object recognition from natural language descriptions. In *BMVC*, 2009.
[25] J. Weston, S. Bengio, and N. Usunier. Large scale image annotation: learning to rank with joint word-image embeddings. *Machine Learning*, 81(1), 2010.