Learning Image Similarity from Flickr Groups Using Stochastic Intersection Kernel Machines

Gang Wang[1] Derek Hoiem[2] David Forsyth[2]

[1] Dept. of Electrical and Computer Engineering
University of Illinois at Urbana-Champaign (UIUC)
gwang6@uiuc.edu

[2] Dept. of Computer Science
University of Illinois at Urbana-Champaign (UIUC)

Abstract

Measuring image similarity is a central topic in computer vision. In this paper, we learn similarity from Flickr groups and use it to organize photos. Two images are similar if they are likely to belong to the same Flickr groups. Our approach is enabled by a fast Stochastic Intersection Kernel MAchine (SIKMA) training algorithm, which we propose. This proposed training method will be useful for many vision problems, as it can produce a classifier that is more accurate than a linear classifier, trained on tens of thousands of examples in two minutes. The experimental results show our approach performs better on image matching, retrieval, and classification than using conventional visual features.

1. Introduction

Digital cameras have made it much easier to take photos, but organizing those photos is still difficult. As a result, many people have thousands of photos sitting on their hard disk in some miscellaneous folder. Fortunately, the same digital explosion that created the problem may also supply the solution.

Using online photo sharing sites, such as Flickr, people have organized many millions of photos into hundreds of thousands of semantically themed groups. Our idea is to determine which images are similar and how they are similar by learning from Flickr groups. Simply put, two images are similar in some sense if they are likely to belong to the same group. If we can learn these group membership likelihoods, we can help a user sort through his photo collection by text or image-based query and refine the search with simple feedback. In doing so, we allow flexible, on-the-fly organization of his photo album.

But how can we learn whether a photo is likely to belong to a particular Flickr group? We can easily download thousands of images belonging to the group and many more that do not, suggesting that we train a classifier. Still, the time that it would take to learn hundreds of categories is daunting. We propose a new method for stochastic learning of support vector machines (SVMs) using Histogram Intersection Kernel (HIK). We combine the kernelized stochastic learning algorithm from [14] with the support vector approximation trick [18] proposed for fast classification. The result is an algorithm that is much faster and more accurate than the original stochastic learning algorithm, allowing us to learn from five thousand examples with 3000 features in just 15 seconds. This algorithm will be useful for a wide variety of computer vision algorithms.

Space allows only a brief survey of **related work**. We wish to train a very large scale kernel SVM. There is a good survey of current results in large-scale kernel machine training in [4]. Algorithms are generally of two classes; either one exploits the sparseness of the lagrange multipliers (like SMO [22] and variants), or one uses stochastic gradient descent on the primal problem. Stochastic gradient descent has the advantage that, at each iteration, the gradient is calculated for only a single training example. Very good results can be obtained without touching every example [25, 3]. Kivinen *et al.* describe a method that applies to kernel machines [14]. We use a similar form of incremental training, exploiting a method of Maji *et al.* [18] for very quickly evaluating a histogram intersection kernel. In the same proceedings, [19] uses stochastic gradient descent to learn an additive classifier with a max margin criteria avoiding the need to store any support vectors. With certain regularization, this is an approximation to the histogram intersection kernel SVM.

There is an extensive content-based image management literature, with recent reviews in [15, 8]. Appearance [26] or iconic [13] matching are well established techniques. Clustering images as a way to expose the structure of a collection dates to at least [2, 1]. Relevance feedback has been

used at least since [6]. Annotating images with words to allow word searches to at least [2]. None of these technologies works terribly well. Generally, users are querying for specific objects or object classes [9], and supporting object semantics is difficult. In recent work, Frome *et al.* [11] show that local metrics around examples built using Caltech 101 images give good retrieval behavior. The most relevant work is [24]. We both advocate the use of features composed of category predictions for image retrieval. Our key observation, which differs from [24], is that Flickr provides an organizational structure with thousands of categories that reflect how people like to group images, each with tens of thousands of examples, and our SIKMA classifier allows efficient and accurate learning of these categories.

Our similarity features capture the sense of an image rather well. First, they make an effective feature for supervised learning of new categories, as we show on the PASCAL 2007 dataset. Second, this property generalizes well. We show that our similarity measure can be used for image retrieval on a test dataset *which was not obtained from Flickr*. In particular, we demonstrate marked improvements on image clustering, retrieval, matching and relevance feedback using a large test dataset from Corel. These improvements extend even to categories not well covered by our Flickr groups.

2. Approach

We download thousands of images from many Flickr groups. For each group, we train a kernelized SVM introduced in section 2.1. For a test image, we use the trained group classifiers to predict likely group memberships. We use these predictions to measure similarity (Section 2.2).

2.1. Stochastic Intersection Kernel MAchines (SIKMA)

We train a Stochastic Intersection Kernel MAchine for each Flickr group as the classifier. Suppose we have a list of training examples $\{(x_t, y_t), t = 1, \cdots, T, y_t \in \{-1, +1\}\}$. We aim to learn a decision function $f{:}X \longrightarrow R$, using a kernel machine; this yields $f = \sum_{i=1}^{N} \alpha_i K(x_i, \bullet)$ where K is a kernel function. Then for a test example u, the classification score is $f(u) = \sum_{i=1}^{N} \alpha_i K(x_i, u)$. In a primal method, we obtain the kernel machine by minimizing the regularized empirical risk:

$$\frac{1}{T}\sum_{t=1}^{T} l(f(x_t), y_t) + \frac{\lambda}{2}\|f\|^2 \tag{1}$$

where l is a loss function, in our case the hinge loss $l(f(x_t), y_t) = \max(0, 1 - y_t f(x_t))$. Computing the gradient of the regularized empirical risk involves a sum over all data points, which is very expensive. In a stochastic gradient method, we approximate the gradient by replacing the sum over all examples with a sum over some subset, chosen at random, and then take a step. It is usual to consider a single example. In Kivinen *et al.*'s method [14], one sees this as presenting the training examples to the classifier in some random order, one by one, then updating the classifier at each example to get a set of f, $\{f_0, f_1, ..., f_T\}$. Now assume we have f_{t-1}. When the tth training example comes, we take a step down the gradient of $Q = l(f(x_t), y_t) + \frac{\lambda}{2}\|f\|^2$. By writing $\sigma_t = \begin{cases} 1 \text{ if } y_t f_{t-1}(x_t) < 1 \\ 0 \text{ otherwise} \end{cases}$, we obtain the estimate of the new decision function as (see [14] for the detailed derivation):

$$f_t = (1 - \lambda\eta_t) f_{t-1} + \eta_t \sigma_t y_t K(x_t, \bullet) \tag{2}$$

Where η_t is the step length at the tth step. This update can also be written in terms of the lagrange multipliers for the examples seen to date. In particular, we can write $\alpha_i = (1 - \lambda\eta_t)\alpha_i$ for $i < t$ and $\alpha_t = \eta_t \sigma_t y_t$. We can see that when there are a large number of support vectors (this would happen in large datasets), it is expensive to calculate $f_{t-1}(x_t)$. The NORMA algorithm in [14] keeps a set of support vectors of fixed length by dropping the oldest ones. As we shall see, doing so comes at a considerable cost in accuracy.

Recently, Maji *et al.* [18] show that the support vectors of an intersection kernel machine can be efficiently represented. This trick is exploited to train a fast stochastic intersection without dropping any support vectors.

Write f_{t-1} as $\sum_{i=1}^{N_{t-1}} \alpha_i K(x_i, \bullet)$, where K denotes the histogram intersection kernel. Then

$$f_{t-1}(x_t) = \sum_{i=1}^{N_{t-1}} \alpha_i \sum_{d=1}^{D} \min(x_i(d), x_t(d)) \tag{3}$$

$$= \sum_{d=1}^{D} \sum_{i=1}^{N_{t-1}} \alpha_i \min(x_i(d), x_t(d)) \tag{4}$$

D is the feature dimension. At each dimension d, if we have the sorted values of $x_i(d)$ as $\overline{x_i}(d)$, with the corresponding $\overline{\alpha_i}$, then:

$$\sum_{i=1}^{N_{t-1}} \alpha_i \min(x_i(d), x_t(d)) \tag{5}$$

$$= \sum_{l=1}^{r} \overline{\alpha_l x_l}(d) + x_t(d) \sum_{l=r+1}^{N_{t-1}} \overline{\alpha_l} \tag{6}$$

where $\overline{x_r}(d) \leq x_t(d) < \overline{x_{r+1}}(d)$. As [18], we use M piecewise linear segments to approximate equation 6. Given that feature histograms are normalized, each element of the feature vectors falls in the range of [0 1]. We divide this range to M (not necessarily even) bins, and the starting value of each bin is recorded in vector P.

Notice that the terms of equation 6 contain only partial sums of α, rather than the values. This means that the complexity of representing the kernel machine has to do with these partial sums, rather than the number of support vectors. We can store these sums in tables, and update them efficiently. In particular, we have two tables B_1 and B_2 with dimensions $M \times D$, where M is the number of bins and D is the feature dimension. $B_1(m,d)$ contains the value $\sum_i \alpha_i x_i(d)\sigma_i$, $\sigma_i = 1$ if $x_i(d) < P(m)$ and zero otherwise; $B_2(m,d)$ stores the value $\sum_i \alpha_i \sigma_i$, $\sigma_i = 1$ if $x_i(d) \geq P(m)$ and zero otherwise.

To evaluate the function for $x_t(d)$, we quantize $x_t(d)$ and look up in B_1 and B_2. The two values are interpolated to calculate equation 6. Since the elements of the tables are linear in the lagrange multipliers, updating the tables is straightforward. At the t'th iteration both B_1 and B_2 are multiplied by $1 - \lambda\eta_t$. If σ_t is non-zero, the tables B_1 and B_2 are updated accordingly.

Comparison: The computational complexity to train SIKMA is $O(TMD)$, where T is the number of training examples that are touched, M is the number of quantization bins and D is the feature dimension (compare the computational complexity of the conventional SVM solver at $O(T^2D)$). The space cost is $O(MD)$ (the conventional SVM solver needs $O(T^2)$), and evaluation is $O(D)$ for each test example. Unlike NORMA [14], SIKMA doesn't need to drop examples to maintain efficiency.

2.2. Measuring image similarity

We use the proposed SIKMA training algorithm to train classifiers to predict whether an image is likely to belong to a Flickr group. The set of predictions can be used to predict similarity. We found a simple Euclidean distance between the SVM outputs to work as well as any other (such as cosine distance between probability vectors, where the probability vectors were normalized using Platt's probabilistic outputs algorithm [23]). Once computed, this similarity measure can be used to perform image-based queries or to cluster images. Since we have names (groups) attached to each prediction, we can also sometimes perform text-based queries (e.g., "get images likely to contain people dancing") and determine how two images are similar.

3. Implementation details

3.1. Features

We use four types of features to represent images and train the SVM classifier. The **SIFT** feature [17] is popularly used for image matching and object recognition. We use it to detect and describe local patches. We extract about 1000 patches from each image. The SIFT features are quantized to 1000 clusters and each patch is denoted as a cluster index. Each image is then represented as a normalized histogram of the cluster indices. The **Gist** feature has been proven to be very powerful in scene categorization and retrieval [21]. We represent each image as a 960 dimensions Gist descriptor. We extract **Color** features in the RGB space. We quantize each channel to 8 bins, then each pixel is represented as a integer value range from 1 to 512. Each image is represented as a 512 dimensional histogram by counting all the pixels. The histogram is normalized. We also extract a very simple **Gradient** feature, which can be considered as a global and coarse HOG feature [7]. We divide the image to 4*4 cells, at each cell, we quantize the gradient to 16 bins. The whole image is represented as a 256 dimensional vector.

For each Flickr group, we train four SVM classifiers, one for each of the above four features. We combine the outputs of these four classifiers to be a final prediction on a validation data set. The final prediction is used to measure image similarity.

To compare our results with conventional visual similarity, we use a **Unified** visual feature, obtained by concatenating the above four visual features. Each feature is associated with a weight. The weights are learned on a validation set, where we force the images from the same categories to be close and images from different categories to be far away with the learned weights. Similar methodology can be found in [20]; this feature tends to outperform each separate feature, and so gives a fair appearance baseline.

3.2. Flickr groups

For our experiments, we use 103 Flickr groups that capture a range of common topics. Some large Flickr groups are uninformative (e.g., "10 million photos"), and we ignore them. Groups that we use are organized by objects, such as "aquariums" and "cars", scenes, such as "sunsets" and "urban", and abstract concepts, such as "Christmas" and "smiles". We provide the complete list (paraphrased) below: aquariums, airplanes, American flags, animals, architecture, art, bags, beaches, bees, bikes, birds, boats, bonsai, bottles, bridges, buses, butterflies, cameras, candles, cars, castles, cats, chairs, characters, children, Christmas, churches, concerts, cows, dances, dogs, dolphins, drawings, ducks, eagles, eating, eggs, faces, fashion, ferns, ferrets, fireworks, fishing, flamingos, flowers, fog+rain, food, frogs, fruits, the Golden Gate Bridge, hands, helicopters, horses, ice, insects, laptop lunch, light, lizards, love, macro-flower, monkeys, moons, motorbikes, mountains, mushrooms, painting, pandas, penguins, people, plants, rain, rainbows, rural, sheep, skateboarding, skies, skyscrapers, smiles, snails, sneakers, snow, socks, spiders, sports, squirrels, stairs, sunset, sushi, tea cup, teddy bears, tigers, tomatoes, toys, trains, trees, trucks, turtles, urban, watches, water drops, waterfalls, and weddings.

Note that, while this list is long, we could potentially

make use of thousands of categories, each containing thousands of photographs.

Most groups contain $15,000 \sim 30,000$ images. To train a group classifier, we also sample about 60,000 negative images from other groups. Training each SVM using our SIKMA algorithm takes about 150 seconds per classifier, which tends to have between 5,000 and 8,000 support vectors. This is remarkable, considering that standard batch training is infeasible and that the previously proposed online algorithm would take at least 10 times longer to produce a much less accurate classifier.

4. Experiments

In Section 4.1, we compare our fast histogram intersection SVM training algorithm (SIKMA) to alternatives. We show that our method is much faster and more accurate than a recently proposed stochastic learning method [14]. Our method is nearly as accurate as batch training on small problems involving a few thousand examples and enables training with tens of thousands of examples. For example, our method can train on 80,000 examples in 150 seconds, while batch training requires several hundred seconds to train with 5,000 examples.

In Section 4.2, we evaluate the usefulness of our learned similarity measure in several ways. We show that our similarity measure allows much better image matching in the Corel dataset and improves more with feedback than similarity based on the original image features. We can also perform text-based searches on non-annotated images in some cases.

4.1. SIKMA Training Time and Test Accuracy

We compare batch training, an existing stochastic learning algorithm NORMA [14], our proposed algorithm SIKMA for training SVMs with the histogram intersection kernel and linear SVM on the PASCAL VOC 2007 dataset [10]. We also report average precision results for our 103 Flickr categories.

Classifier Comparison. The batch learning method is implemented in LIBSVM [5]. LIBSVM does not support histogram intersection kernel directly, so we pre-compute the kernel matrix with a Mex function and use LIBSVM to solve it. In NORMA, the learning rate is set to be $\frac{0.71}{\lambda\sqrt{t}}$, and it keeps 500 support vectors at most. λ is validated for each category. In SIKMA, the learning rate is set to be $\frac{1}{\lambda(t+100)}$, and λ is set to be 0.00005. The number of quantization bins is set to be 50. We also compare with linear SVM implemented in LIBSVM, where the parameter C is cross validated.

This experiment is conducted on the PASCAL VOC 2007 image classification task, which has 20 object classes. As features, we use a 3000-bin histogram of quantized SIFT codewords, a standard feature for this task. The average precision (AP) results, training time and test time of different methods are compared in Table 1. Our method achieves similar accuracy to batch training when running for 6 rounds and is more accurate than either NORMA (also with histogram intersection kernel) or batch training of linear SVMs. We were surprised that the online kernelized learner (NORMA) tends to underperform the batch linear SVM. This seems to be due to the approximation of discarding support vectors with low weights and due to difficulty in choosing the learning rate for NORMA, on which we spent considerable effort. By contrast, our method is insensitive to the learning rate and consistently outperforms the linear classifier.

Our algorithm is much faster than NORMA and the batch algorithm. For larger problems, the speedup over batch will increase dramatically, and NORMA will be forced to make larger approximations at great cost to classifier accuracy. The same is true for memory requirements, which would make standard batch training impossible for problems with tens of thousands of examples.

In summary, our SIKMA algorithm makes it easy to train SVMs with the histogram intersection kernel on large datasets. Recent work by Maji et al. [18] makes classification nearly as fast as for linear kernels (this enables our training method). Together, these works are important because histogram intersection kernels tend to provide more accurate classifiers for histogram-based features that are used in many computer vision problems.

Performance on Flickr Categories. In Figure 1, we show average precision for our 103 Flickr categories. These are trained using positive group images (most have $15,000 \sim 30,000$ positive images) as well as about 60,000 negative images sampled from other groups. Each group has 20,900 held out test times: 500 positive and 20,4000 negative sampled from other groups. The average AP over these categories is 0.433.

4.2. Evaluation of Learned Similarity Measure

Our similarity measure consists of Euclidean distance between predictions on Flickr categories (as mentioned earlier, this works as well as other slightly more complicated distances). The quality of the similarity measure is the most important factor in automatic organization, and we evaluate it in several ways. We can rank images according to similarity (image-based query) or cluster a set of images. We can also find images that are likely to belong to particular groups or have certain tags. We can also often say how two images are similar, suggesting the possibility of more intuitive feedback mechanisms. The following experiments are performed on 38,000 images from the Corel dataset (except where noted). Each image has a CD label and a set of keyword annotations, which we treat as ground truth for

	Linear	NORMA (HIK, SV=500)	SIKMA (2 rounds)	SIKMA (6 rounds)	Batch (HIK)
AP	0.362	0.308	0.414	0.436	0.440
training time (seconds)	-	172.4	15.3	46.7	638.0
test time (seconds)	0.5	63.9	3.9	3.9	236.8

Table 1. The AP values, training time and test time of the five SVM training methods are compared on the PASCAL VOC 2007 image classification task. All the values are averaged over the 20 object categories. HIK denotes the histogram intersection kernel. NORMA (HIK, SV=500) denotes HIK is used with the NORMA algorithm and at most 500 support vectors are kept. SIKMA (2 rounds) denotes the SIKMA algorithm visits each training example twice. SIKMA (6 rounds) denotes each training example is visited six times.

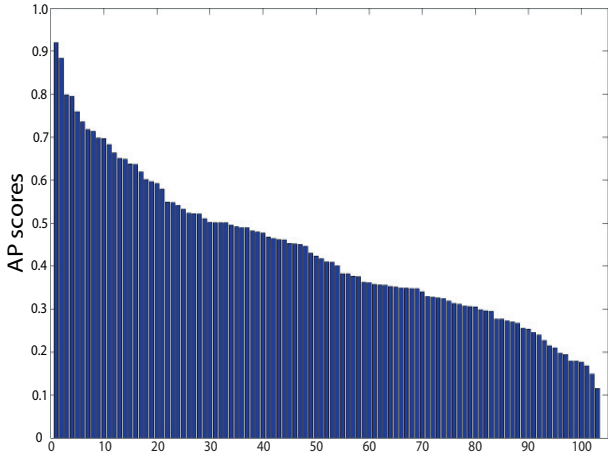

Figure 1. The AP scores of the 103 Flickr groups (categories). For each category, There are 20,900 held-out examples (500 positive). The five groups which get the highest AP values are: laptop lunch, fireworks, pandas, socks and moon; the five groups which get the lowest AP values are: love, art, trees, ice and light.

matching. There are 100 images per CD, and roughly 3-5 keywords per image.

Image Matching. We randomly select 500 images to be queries and rank the remaining Corel images using either our learned similarity or visual feature-based similarity measures. Images that the CD label or at least one keyword in common are considered correct matches; others are incorrect matches. For each query image, we calculate the AP value. Our learned similarity produces an AP of 0.221, averaged over the queries; for feature-based similarity, the average AP is 0.192. Fig. 2 compares the nearest neighbor images of a query image given by the two similarity measures. Images are sorted by similarity in descending order from left to right, top to bottom.

We also randomly select 25 images for keywords "city","house" and "skiing", which don't have corresponding Flickr categories in our collection. For each of these images, we also rank the other Corel images and calculate the AP score. The average AP scores for these categories with the learned similarity is 0.111, 0.120 and 0.097, which is slightly better than for the visual similarity at 0.105, 0.113, 0.096. These results indicate that the learned similarity works best when queries are related to the learned Flickr categories but provides an advantage in out of sample cases as well. As more Flickr categories are learned, fewer queries will be out of sample. Note that 1,000 classifications per second can be performed after computing the features, so it is entirely feasible to use thousands of Flickr categories (downloading tens of millions of images is the main obstacle for training).

Image Matching with Feedback. Using a subset of 10 CDs, we also investigate the effectiveness of simple relevance feedback. We compare the performance of our features with that of pure appearance features in a relevance feedback task. Because users will provide very little feedback (we use 5 positive and 5 negative examples), a good simulation of this task is demanding. We use the same CDs as as [16], which are chosen to provide an unambiguous ground truth: 1 (sunsets), 21 (race cars), 34 (flying airplanes), 130 (African animals), 153 (swimming), 161 (egyptian ruins), 163 (birds and nests), 182 (trains), 276 (mountains and snow) and 384 (beaches). Images are considered to match only if they have the same CD label. We compute average AP over 25 randomly selected queries.

To simulate feedback, after each query, we select the top five negative examples and five randomly chosen positive examples from among the top 50 ranked images and label them according to ground truth. We use this to train a weight vector on our distance function (initially uniform). With the feedback, we aim to minimize the following function:

$$\sum_{i}^{10} y_i w \bullet (x_q - x_i)^2 \tag{7}$$

Subject to $w_d \geq 0, \sum_d w_d = 1$ where x_q is the feature representation of the query image. x_i is the feedback example. y_i is 1 if it is positive, otherwise 0. If we had very extensive feedback, we would have a good estimate of the cost function. With relatively little feedback, the model of cost applies only locally around the current values of w. For this reason, we take a single step down the gradient, then project to the constraints. The scale of the step is chosen on a validation set of 20 queries, and then fixed.

The average AP values on these 25 query images with

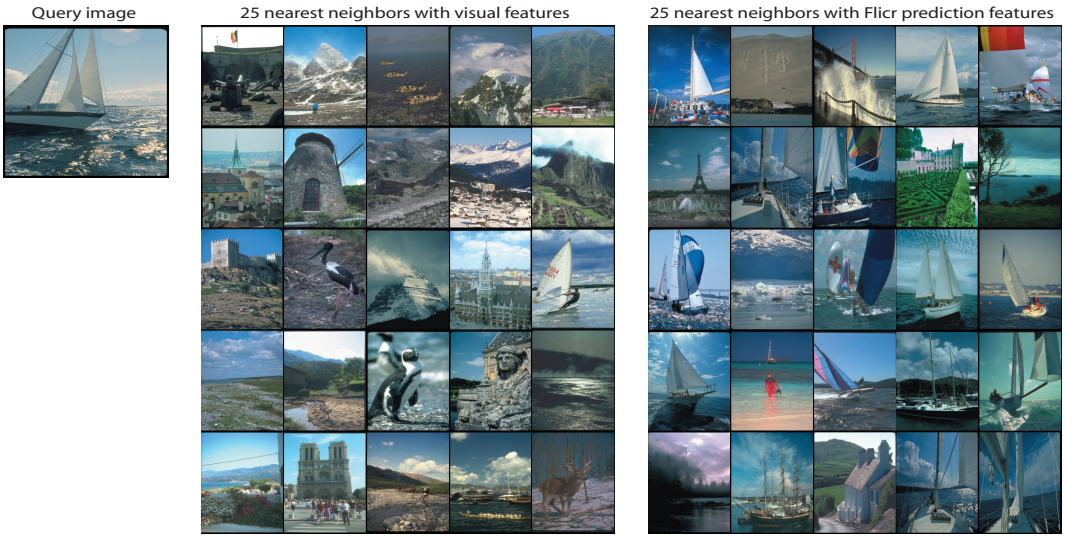

Figure 2. The left column shows a "ship" query image; the center column shows the 25 nearest neighbor images found with visual features; the right column shows the 25 nearest neighbor images found with Flickr prediction features. The rank is from left to right, from top to bottom.

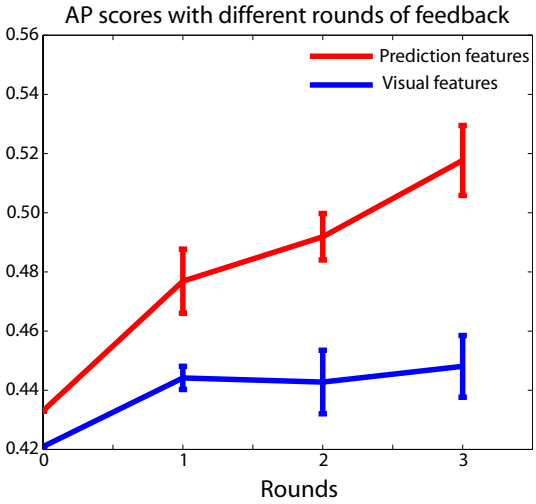

Figure 3. The average AP values with three rounds of feedback. The red line shows the results with Flickr prediction features and the blue line shows the results with visual features.

three rounds of feedback are compared in Fig. 3. Note that the similarity based on Flickr prediction features improves more with each round of feedback than with the visual features. Fig. 4 shows the nearest neighbors images without feedback and with the first round of feedback for a query image. The selected negative images are shown in red and selected positive images are shown in green.

Semantic similarity of image pairs. In Fig. 5, we show six pairs of similar Corel images. The text shows the Flickr groups which both of the images are likely to belong to.

Unsupervised clustering results on Corel data set. Using the same 10 CDs as listed above, we also compare results on clustering. We represent these images with our prediction features (classification scores) and visual feature respectively. We cluster these 1000 images to 15 clusters in an unsupervised way (K-means). Each cluster is labeled with the most common CD label in this cluster. Each image is labeled by the cluster label accordingly. The accuracy of Flickr prediction features is 0.559 and the accuracy of visual features is 0.503.

Text-based queries. Because each Flickr category can be described with several words, we can support text-based queries. When users input a word query, we can find the Flickr group whose description contains such words. The corresponding Flickr group classifier is then used to classify personal photos. The photos with high confidence are returned to users.

We test this on the Corel data set,with two queries "airplane" and "sunset". There are about 38,000 images in total, from which there are 840 "airplane" images and 409 "sunset" images. We rank the images according to the Flickr group classification score. We get an AP value 0.28 for "airplane" and 0.16 for "sunset". In the 100 top ranked images for "airplane", there are 52 true positives; in the 100 top ranked images for "sunset", there are 26 true positives.

The Corel images which are most relevant to "sunset" and "airplane" are shown in Fig. 6 according to the classification scores.

Classification. We can also use our Flickr group predictions as features for classification. In Table 2, we compare our prediction features with visual features. As implemented in [12], for the visual features, we train a chi-square kernel machine with the unified features (chi-square kernel

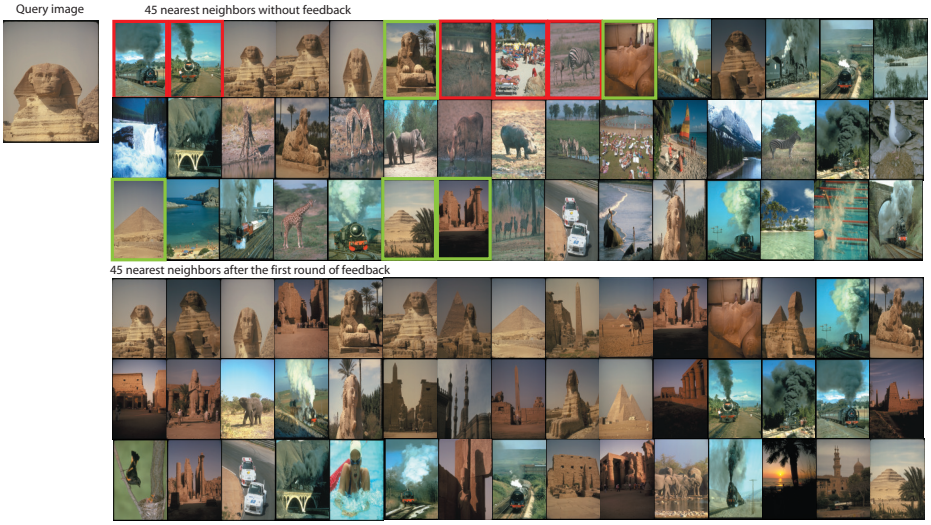

Figure 4. The left column shows the query image; the center column shows the 50 nearest neighbors found with the Flickr prediction features, the five negative images (in red) and five positive images (in green) are selected for feedback; after one round of feedback, we get the 50 nearest neighbors shown in the right column.

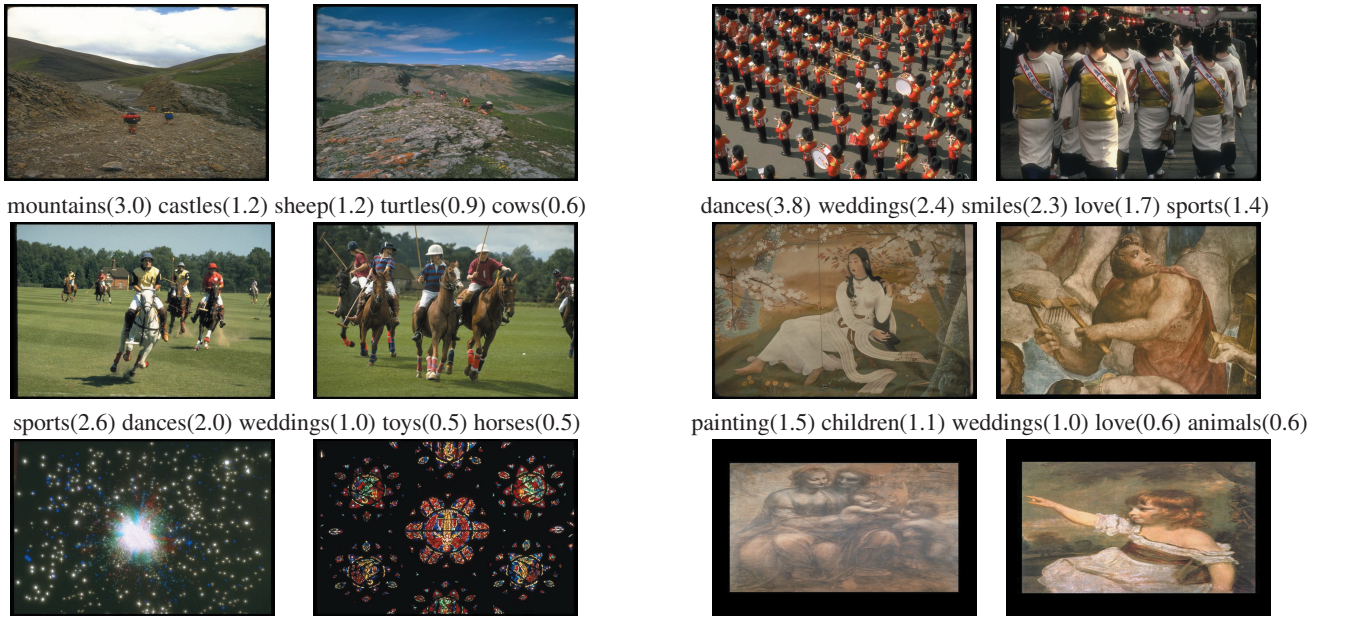

Figure 5. Six pairs of similar Corel images. The text shows the top five Flickr groups which both of the images are likely to belong to. The value for each group in the parenthesis is $100 \times p(\text{group} \mid \text{image1})p(\text{group} \mid \text{image2})$.

is the state-of-the-art for histogram based image classification). Our group predictions features are not histograms, so we have to use a RBF kernel instead. Table 2 shows that our features are usually more effective than the visual features that are used to train the Flickr classifiers. Exceptions are objects that are typically in the background, such as tables, chairs, and bottles.

5. Conclusion

We have proposed SIKMA, an algorithm to quickly train an SVM with the histogram intersection kernel using tens of thousands of training examples. We use SIKMA to train classifiers that predict Flickr group membership. This serves as a basis for image similarity: two images that are likely to belong to the same Flickr groups are considered similar. Our experimental results show that our approach

	aeroplane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow
Visual features	0.647	0.399	0.450	0.540	**0.207**	0.425	0.577	0.388	**0.439**	0.273
Prediction features	**0.650**	**0.443**	**0.486**	**0.584**	0.178	**0.464**	**0.632**	**0.468**	0.422	**0.296**
	table	dog	horse	motorbike	person	plant	sheep	sofa	train	monitor
Visual features	**0.373**	0.343	0.657	0.489	0.749	**0.330**	**0.324**	**0.323**	0.619	0.322
Prediction features	0.208	**0.377**	**0.666**	**0.503**	**0.781**	0.272	0.321	0.268	**0.628**	**0.333**

Table 2. The AP value with Flickr prediction features and visual features on PASCAL 2007 classification for each object class.

Figure 6. The Corel images which are most relevant to the query "airplane", obtained by one-vs-all classification with our SIKMA method, trained on the Flickr airplane group. Images are ranked according to their classifier score.

measures image similarity better than matching with visual features.

6. Acknowledgement

This work was supported in part by the National Science Foundation under IIS -0803603 and in part by the Office of Naval Research under N00014-01-1-0890 as part of the MURI program. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect those of the National Science Foundation or the Office of Naval Research.

References

[1] K. Barnard, P. Duygulu, and D.A. Forsyth. Clustering art. In *CVPR*, pages II:434–441, 2001. 1

[2] K. Barnard and D.A. Forsyth. Learning the semantics of words and pictures. In *ICCV*, pages 408–15, 2001. 1, 2

[3] L. Bottou. Stochastic learning. *Lecture notes in computer science*, pages 146–168, 2004. 1

[4] L. Bottou, O. Chapelle, D. DeCoste, and J. Weston, editors. *Large-Scale Kernel Machines*. 2007. 1

[5] C.C. Chang and C.J. Lin. LIBSVM: a library for support vector machines. *Software available at http://www. csie. ntu. edu. tw/cjlin/libsvm*, 80:604–611, 2001. 4

[6] I.J. Cox, M.L. Miller, S.M. Omohundro, and P.N. Yianilos. Pichunter: Bayesian relevance feedback for image retrieval. In *ICPR*, 1996. 2

[7] N. Dalai, B. Triggs, I. Rhone-Alps, and F. Montbonnot. Histograms of oriented gradients for human detection. *CVPR*, 1, 2005. 3

[8] Ritendra Datta, Dhiraj Joshi, Jia Li, and James Z. Wang. Image retrieval: Ideas, influences, and trends of the new age. *ACM Comput. Surv.*, 40(2):1–60, 2008. 1

[9] P.G.B. Enser. Pictorial information retrieval. *J. of Documentation*, 51(2):126–170, 1995. 2

[10] M. Everingham, L. Van Gool, CKI Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results, 2007. 4

[11] A. Frome, Y. Singer, and J. Malik. Image retrieval and classification using local distance functions. In *Advances in Neural Information Processing*, 2007. 2

[12] Derek Hoiem Gang Wang and David Forsyth. Building text features for object image classification. *CVPR*, 2009. 6

[13] C.E. Jacobs, A. Finkelstein, and D.H. Salesin. Fast multiresolution image querying. In *Proc SIGGRAPH-95*, pages 277–285, 1995. 1

[14] J. Kivinen, AJ Smola, and RC Williamson. Online learning with kernels. *IEEE Transactions on Signal Processing*, 52(8):2165–2176, 2004. 1, 2, 3, 4

[15] Y. Liu, D. Zhang, G. Lu, and W-Y Ma. A survey of content-based image retrieval with high-level semantics. *Pattern Recognition*, 40:262–282, 2007. 1

[16] N. Loeff and A. Farhadi. Scene Discovery by Matrix Factorization. In *ECCV*, 2008. 5

[17] D.G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *IJCV*, 2004. 3

[18] S. Maji, A.C. Berg, and J. Malik. Classification using intersection kernel support vector machines is efficient. In *Proc. CVPR*, 2008. 1, 2, 4

[19] Subhransu Maji and Alexander C. Berg. Max-Margin Additive Classifiers for Detection. *ICCV*, 2009. 1

[20] Ameesh Makadia, Vladimir Pavlovic, and Sanjiv. A new baseline for image annotation. In *ECCV*, 2008. 3

[21] A. Oliva and A. Torralba. Modeling the shape of the scene: a holistic representation of the spatial envelope. *IJCV*, 42, 2001. 3

[22] J.C. Platt. Fast training of support vector machines using sequential minimal optimization. 1999. 1

[23] John C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances in Large Margin Classifiers*, pages 61–74. MIT Press, 2000. 3

[24] N. Rasiwasia, PJ Moreno, and N. Vasconcelos. Bridging the gap: Query by semantic example. *IEEE Transactions on Multimedia*, 9(5):923–938, 2007. 2

[25] S. Shalev-Shwartz, Y. Singer, and N. Srebro. Pegasos: Primal estimated sub-gradient solver for SVM. In *ICML*, 2007. 1

[26] M.J. Swain and D.H. Ballard. Color indexing. *IJCV*, 7(1):11–32, 1991. 1