Learning Image Similarity from Flickr Groups Using Fast Kernel Machines

Gang Wang, *Member, IEEE*, Derek Hoiem, *Member, IEEE*, David Forsyth, *Fellow, IEEE*

Abstract—Measuring image similarity is a central topic in computer vision. In this paper, we propose to measure image similarity by learning from the online Flickr image groups. We do so by: choosing 103 Flickr groups; building a one-vs-all multi-class classifier to classify test images into a group; taking the set of responses of the classifiers as features; calculating the distance between feature vectors to measure image similarity. Experimental results on the Corel dataset and the PASCAL VOC 2007 dataset show that our approach performs better on image matching, retrieval, and classification than using conventional visual features. To build our similarity measure, we need one-vs-all classifiers that are accurate, and can be trained quickly on very large quantities of data. We adopt an SVM classifier with a histogram intersection kernel. We describe a novel fast training algorithm for this classifier: the Stochastic Intersection Kernel MAchine (SIKMA) training algorithm. This method can produce a kernel classifier that is more accurate than a linear classifier, on tens of thousands of examples in minutes.

Index Terms—Image Similarity, Kernel Machines, Stochastic Gradient Descent, Online Learning, Image Classification, Image Organization.

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 folders but do not know or do not have time to organize them. Fortunately, the same digital explosion that created the problem may also supply the solution. Users on online photo sharing sites, like Flickr, have organized many millions of photos into hundreds of thousands of semantically themed groups. These groups expose implicit choices that users make about which images are similar. Flickr group membership is usually less noisy than Flickr tags, because images are screened by group members. Furthermore, Flickr groups can represent richer themes than regular tags and object categories. For example, there is a Flickr group called "No-Flash Night Shots" that highlights shots which are taken at night and involve long exposures. Flickr group membership patterns capture rich notions of image similarity. In this paper, we exploit these group membership patterns to learn a measure of similarity for pairs of test images. More specifically, two images are considered to be similar if they are likely to belong to the same set of groups. We use this measure to help a user query a photo collection with text or with images, and refine this search with simple relevance feedback. In doing so, we allow flexible, on-the-fly interaction between the user and the photo album.

We measure the likelihood that an image belongs to a particular Flickr group using a trained classifier. There are many Flickr groups, each with many examples, so we require an efficient training algorithm. In this paper, we describe a new method to learn support vector machines (SVMs) with a Histogram Intersection Kernel (HIK), which has been shown to outperform linear kernels for histogram-based image classification [14], [19]. Our method combines the kernelized stochastic learning algorithm from [17] with the support vector approximation method [27] 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 dimensional features in just 15 seconds. Our algorithm is also memory-efficient, since training examples are processed one-by-one. Our SVM-HIK training algorithm is useful for a variety of recognition tasks the employ histogram-based features.

An image is represented with a vector, whose components represent the strength of association to each Flickr group, as measured by our trained classifiers. To compute similarity between two images, we compute Euclidean distance between the feature vectors, which reflects that similar images should belong to the same Flickr groups.

Our experiments show that our training algorithm is efficient and provides accurate Flickr group classifiers. Our ultimate goal, however, is to use the Flickr-learned similarities for other image classification and organization tasks. We show that our learned similarity measure outperforms the original visual features for image matching and clustering on the Corel dataset. We also show that the learned similarities lead to more

- *G. Wang is with the School of Electrical and Electronic Engineering, Nanyang Technological University and the Advanced Digital Science Center, Singapore.*
  E-mail: wanggang@ntu.edu.sg
- *D. Hoiem and D. Forsyth are with the Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL, 61801.*

effective relevance feedback [34]. Taking advantage of the text attached to Flickr groups, we also demonstrate effectiveness in retrieval from text queries. Finally, we apply our method to calculate kernel values for the image classification task on the PASCAL VOC 2007 dataset [12]. Our novel similarity kernel outperforms the kernel based on visual features on 12 of 20 object categories.

To summarize, this paper includes two contributions:

(1)We show how to use implicit knowledge of similarity (represented by the Flickr image groups) to train a measure of image similarity. This idea can be generalized. For example, one might use an online music directory to measure music similarity.

(2)We propose a fast and memory-efficient algorithm to train histogram intersection kernel machines, which is applicable to many recognition tasks.

1.1 Related work

Measuring image similarity is an important research problem in computer vision. Traditional approaches usually extract low-level visual features (e.g., texture features [20], [36], [46], shape features [4], [16], and gradient features [25], [9]) from images, and match them using distance metrics such as the Euclidean distance, the Chi-Square distance, the histogram intersection [40], and the EMD distance [33]. Our paper addresses this problem by learning middle-level image representation from Flickr groups. Our method can exploit hundreds of thousands of Flickr images to discover discriminative visual patterns. Meanwhile, our results are interpretable: we can say what makes two images similar by finding out their shared group memberships.

Our work is related to a number of recently emerged papers which build image representation based on the outputs of concept classifiers [32], [18], [45], [22]. Our key novel observation is that Flickr provides an organizational structure with thousands of categories. Our learned similarity from Flickr groups can reflect how people on the Internet tend to group images.

We train Flickr group classifiers using many Internet images. Several previous papers [13], [21], [37], [47] also learn object models from Internet images, but they tend to gather training examples using text-based image search results, which are not reliable. Their approaches have to alternate between finding good examples and updating object models, in order to be robust against noisy images. In contrast, we use images from Flickr groups, which are screened by the Flickr group members and clean enough to produce good classifiers.

We aim to organize images on the fly using the learned image similarity. There is an extensive content-based image management literature, with recent reviews in [10], [23]. Appearance [41] or iconic [15] matching are well established techniques. Clustering images as a way to expose the structure of a collection dates back to at least [3], [2]. Annotating images with words to allow word searches dates back to at least [3]. None of these technologies works terribly well. When managing family albums, face annotation [49], [53] is especially interesting. Automatic face annotation and retrieval is hard. Active human annotation is usually needed to improve the performance. [8], [42] develop approaches to reduce the efforts of human intervention. Different from the above methods, we exploit the organization structure of Flickr image groups to learn image similarity to organize images.

We wish to train a very large scale kernel SVM. This is an important research problem as we have bigger and bigger datasets in computer vision such as the Lableme dataset [35] and the ImageNet dataset [11]. There is a good survey on large-scale kernel machine training in [6]. Algorithms are generally of two classes: one exploits the sparseness of the lagrange multipliers (like SMO [31] and its variants); the other 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 [38], [5]. Kivinen *et al.* describe a method that applies to kernel machines [17]. However, it has to drop support vectors in the process, because of which classification accuracy drops. We exploit the method of Maji *et al.* [27] to quickly evaluate a histogram intersection kernel and efficiently keep existing support vectors. No support vectors have to be dropped, and the training is fast. This proposed training algorithm is further extended to minimize an objective function enforcing similarity constraints in our later work [48].

This paper is an extension of our published conference paper [51]. We have provided more details in the algorithm section. We have also added more experiments, in particular, testing how the performance changes with different number of quantization bins, with more training data, and with different number of Flickr groups, etc.

2 Approach

We consider two images to be similar if people are likely to organize them into the same groups. We have found Flickr groups to be a more meaningful content indicator than other surrounding meta-data, such as tags and captions. Our basic approach is to download images from each group and to train classifiers that distinguish between photos that belong to the group and those that do not. To get reliable classifiers, many (tens of thousands of) training images are used for each group as the training data. Previous training methods with non-linear kernels cannot handle so many training examples. In this paper, we propose a new approach to train kernel machines using a histogram intersection kernel, namely Stochastic Intersection Kernel MAchines (SIKMA). This algorithm is fast and memory efficient. We apply group classifiers to predict the group memberships of each test image. Image similarity is then measured using the Euclidean distance between prediction scores.

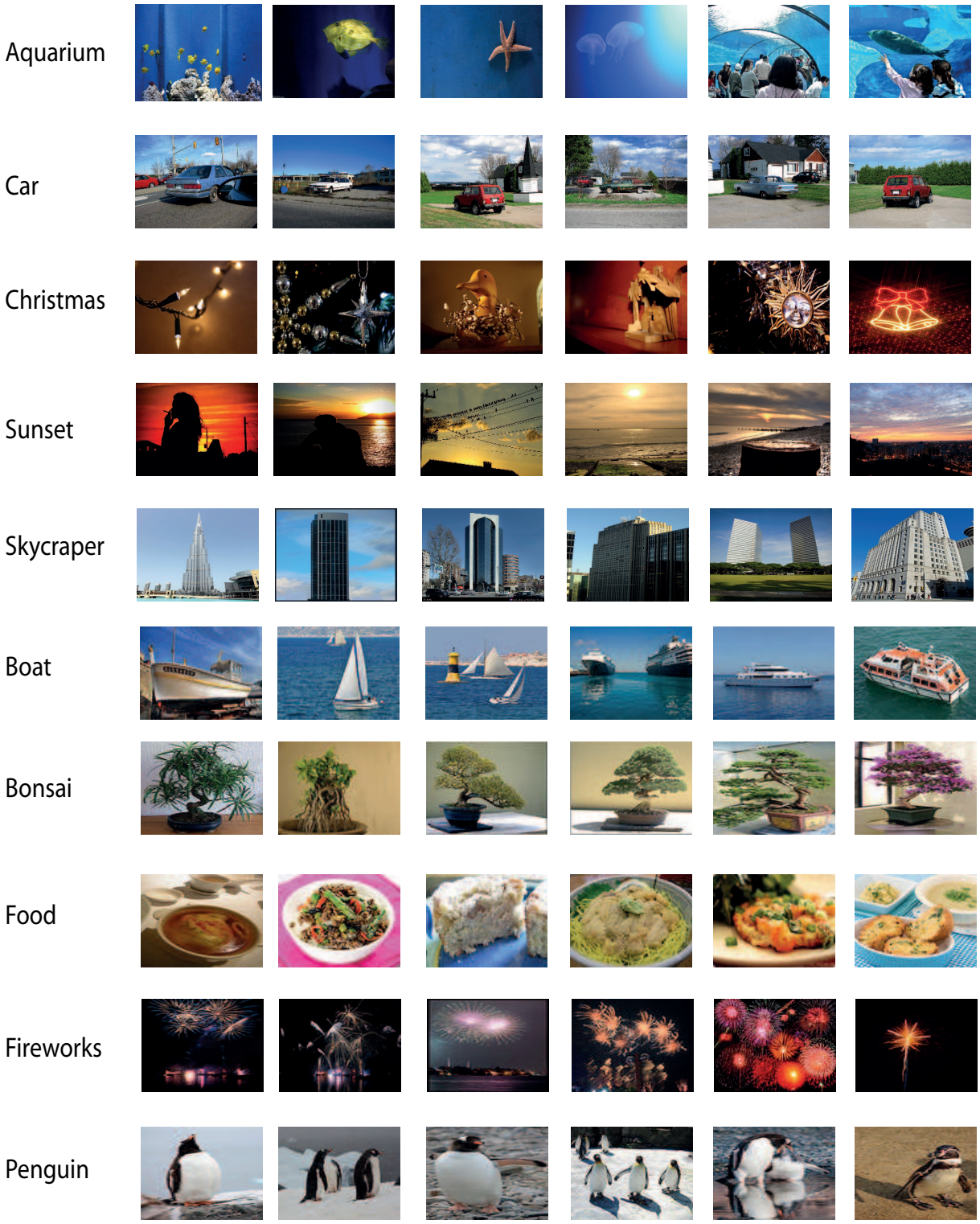


Fig. 1. Image examples from ten Flickr groups. Each row corresponds to a group. These groups are: aquariums, cars, Christmas, sunset, skyscrapers, boat, bonsai, food, fireworks, and penguin.

2.1 Downloading Flickr image groups

We download 103 Flickr image groups that span a wide range of topics, avoiding groups, such as "B&W photography" that do not refer to scene content. Groups that we use include 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.

Some image examples are shown in Fig. 1. Compared to Flickr tags and other metadata, the images within a Flickr group are consistent in theme. Our experiments use 15,000 to 30,000 images in each of the 103 groups.

2.2 Training group classifier using Stochastic Intersection Kernel MAchines (SIKMA)

For each Flickr group, we train a histogram intersection kernel classifier to predict whether a test image belongs to it. Each Flickr group usually contains tens of thousands of images, which can be used as positive examples. To train a discriminative classifier, we also sample a large number of negative images from other groups. We usually use around 80,000 training images to learn each group classifier. Traditional approaches such as SMO [31] cannot handle so many training examples for reasons of computation and memory. In this paper, we describe a new training algorithm called Stochastic Intersection Kernel MAchines (SIKMA). It is an online learning algorithm, meaning that it processes training examples one by one in a sequence and hence does not have the memory issue. Using the histogram intersection kernel, we combine the fast evaluation method developed in Maji et al. [27] with the stochastic gradient descent method, which leads to a very fast training algorithm.

We start by introducing the general stochastic kernel machine framework described in [17]. 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 $\mathbf{u}$, the classification score is $f(\mathbf{u}) = \sum_{i=1}^{N} \alpha_i K(x_i, \mathbf{u})$. In a primal method, we learn the kernel machines by minimizing the regularized empirical risk:

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

where l is a loss function. A hinge-loss $l(f(x_t), y_t) = \max(0, 1 - y_t f(x_t))$, which is used in the conventional support vector machine framework, is also used here. But other loss functions such as log-loss can also be applicable. We describe our approach using the hinge-loss. Equation (1) can be minimized directly using the gradient descent method in the primal space. At the tth iteration, we update f using:

$$f_t = f_{t-1} - \eta_t \frac{\partial L}{\partial f}|_{f=f_{t-1}} \tag{2}$$

η_t is the learning rate at the tth step. Evaluating equation (2) involves calculating $\sum_{t=1}^{T} \frac{\partial l(f(x_t), y_t)}{\partial f}|_{f=f_{t-1}}$, which is very expensive when T is big.

Using the stochastic gradient method, we approximate the gradient by replacing the sum over all examples with a sum over a subset chosen at random, and then take a step. It is usual to consider a single example. In Kivinen *et al.*'s method [17], 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, $\{f_0, f_1, ..., f_T\}$, where f_0 is some initial hypothesis, f_{t-1} is the learned classifier by seeing the first $t-1$ training examples. Now assume we have f_{t-1}. When the tth training example comes, the new objective function is:

$$Q = l(f(x_t), y_t) + \frac{\lambda}{2} \|f\|^2 \tag{3}$$

we update f as:

$$f_t = f_{t-1} - \eta_t \frac{\partial Q}{\partial f}|_{f=f_{t-1}} \tag{4}$$

When l is the hinge-loss, the update becomes

$$\frac{\partial Q}{\partial f}|_{f=f_{t-1}} = \frac{\partial l(f(x_t), y_t)}{\partial f(x_t)} \frac{\partial f(x_t)}{\partial f}|_{f=f_{t-1}} + \lambda f|_{f=f_{t-1}} \tag{5}$$

$$= \frac{\partial l(f_{t-1}(x_t), y_t)}{\partial f_{t-1}(x_t)} K(x_t, \bullet) + \lambda f_{t-1} \tag{6}$$

$$= -\sigma_t K(x_t, \bullet) + \lambda f_{t-1} \tag{7}$$

by writing

$$\sigma_t = \begin{cases} 1 \text{ if } y_t f_{t-1}(x_t) < 1 \\ 0 \text{ otherwise} \end{cases} \tag{8}$$

Then (4) is equivalent to:

$$f_t = (1 - \lambda \eta_t) f_{t-1} + \eta_t \sigma_t y_t K(x_t, \bullet) \tag{9}$$

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 $\alpha_t = \eta_t \sigma_t y_t$. It is hard to detect the convergence of the stochastic gradient descent method. In our implementation, we find it

usually takes tens of thousands of iterations to produce a stable classifier. When there are fewer training instances, we let the algorithm visit each instance multiple times.

When there are a large number of support vectors (this would happen in large datasets), it is expensive to calculate $f_{t-1}(x_t)$ in (8) because it involves calculating kernel values for many pairs of points. The NORMA algorithm developed in [17] keeps a set of support vectors of fixed length by dropping the oldest ones. As pointed out by [17], it is difficult to choose the number of support vectors, which trades off between the memory and computational efficiency and the accuracy. In our experiments, we find dropping support vectors usually produces considerable cost in accuracy.

When a histogram intersection kernel is used, we do not need to drop any support vectors and can still maintain the efficiency. The histogram intersection kernel has a strong performance record in computer vision [14], [19]. Recently, Maji et al. [27] show that the support vectors of an intersection kernel machine can be efficiently represented. We exploit this trick 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; write D as the feature dimension. Then

$$f_{t-1}(x_t) = \sum_{i=1}^{N_{t-1}} \alpha_i \sum_{d=1}^{D} \min(x_i(d), x_t(d)) \tag{10}$$

$$= \sum_{d=1}^{D} \sum_{i=1}^{N_{t-1}} \alpha_i \min(x_i(d), x_t(d)) \tag{11}$$

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:

$$\sum_{i=1}^{N_{t-1}} \alpha_i \min(x_i(d), x_t(d)) \tag{12}$$

$$= \sum_{l=1}^{r} \overline{\alpha_l x_l}(d) + x_t(d) \sum_{l=r+1}^{N_{t-1}} \overline{\alpha_l} \tag{13}$$

where $\overline{x_r}(d) \leq x_t(d) < \overline{x_{r+1}}(d)$. As [27], we use M piecewise linear segments to approximately calculate (13). 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 bins, and the starting value of each bin is recorded in vector P. Note that the bins are not necessarily even: in practice, we assign more bins to the range (e.g., [0 0.01]) which has more data points.

Note that the terms of equation (13) 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 approximately calculate (13). Since the elements of the tables are linear in the lagrange multipliers, updating the tables is straightforward. At the tth iteration both B_1 and B_2 are multiplied by $1 - \lambda\eta_t$. If σ_t (see (8)) is nonzero, the tables B_1 and B_2 are updated accordingly by adding x_t.

Our method involves quantization. Because we quantize each dimension separately and the values are typically small, our approximation does not significantly impact performance when using a moderate number of bins (e.g., 50). In Table 1, we show that very close performance is achieved using our method compared to the exact solution. Maji *et al.* [27] show the similar observation on pedestrian detection using the fast histogram intersection kernel evaluation method.

Computational Complexity: From the above description, we can see that the computational complexity of training the SIKMA is $O(TMD)$, where T is the number of training examples that are touched, M is the quantization bin size, and D is the feature dimension. And the computational complexity of the NORMA algorithm [17] is $O(TPD)$, where P is the number of kept support vectors. P is usually large when there are many features and many training examples, as in our case.

Comparison to Maji *et al.*[26]: Maji *et al.*[26] developed (in parallel) a related method to speed up SVM training. Our method directly minimizes the objective function in the kernel space. Their method first encodes data approximately. Once this is done, the problem becomes linearly separable, and then a fast linear SVM solver such as PEGASOS [38] can be applied.

2.3 Measuring image similarity

For two test images, we use the trained Flickr group classifiers to classify them and get prediction scores. Then the distance between prediction vectors are calculated using the Euclidean distance. We have experimented with metric learning approaches to weight the prediction scores of different Flickr groups, but found the simpler Euclidean distance equally effective.

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 which are likely to contain people dancing") and determine how two images are similar.

3 Features and Implementation Details

Following [50], we adopt four types of features to represent images. The **SIFT** feature [25] is widely used for image matching [39], [30] and object recognition [52]. We employ it to detect and describe local patches in this paper. We extract about 1,000 patches from each image.

The SIFT features are quantized to 1,000 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 [29], [44]. We represent each image as a 960 dimensional Gist descriptor. We extract **Color** features in the RGB space. We quantize each channel to 8 bins, then each pixel is represented as an integer value range from 1 to 512. Each image is then represented as a 512 dimensional histogram. We also extract a very simple **Gradient** feature, which can be considered as a global and coarse HOG feature [9]. 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 produce a final prediction score. We use a validation data set to tune the weight parameters. In the validation procedure, we first randomly generate 10,000 different combinations of weights, then choose the one that maximizes the performance on the validation data set. The final prediction score is used to measure image similarity.

We apply the Euclidean distance to calculate the distance between Flickr prediction vectors and the distance between visual feature vectors. In particular, when we compute distance between visual feature vectors, we weight the four types of features (namely, SIFT; GIST; Color; and Gradient) differently. This is because some features are more discriminative than the others. The weights are learned on a validation set, where we choose weights that force images from the same categories to be close and images from different categories to be far away with the learned weights. Similar approaches have been applied in [28], [50]. This approach is observed to produce a better baseline. Since our method exploits the histogram intersection kernel to train the Flickr group classifier, we also test a second baseline, which calculate the similarity between visual feature descriptors with the histogram intersection. Again, different types of features are weighted differently via validation.

Most groups contain 15,000 to 30,000 images. To train a discriminative group classifier, we sample about 60,000 negative images from other groups. Training each SVM using our SIKMA algorithm takes about 150 seconds. The resulting kernel machine classifier tends to have between 5,000 and 8,000 support vectors. This is remarkable, considering that standard batch training is infeasible and that NORMA [17] would take much 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 [17]. 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 120,000 examples in 80 seconds.

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

4.1 SIKMA Training Time and Test Accuracy

We first compare batch training, an existing stochastic learning algorithm NORMA [17], and our proposed algorithm SIKMA with the histogram intersection kernel and linear SVM on the PASCAL VOC 2007 dataset [12]. We choose this dataset because it is a popular benchmark dataset and has a proper number of images: around 5,000 for training and 5,000 for test. The batch learning method is implemented in LIBSVM [7]. LIBSVM does not support histogram intersection kernel directly, so we precompute the kernel matrix with a Mex function and use LIBSVM to solve it. In NORMA, we set the number of kept support vectors as 500, 1000, and 1500 respectively. λ is validated for each category. Following [5], in SIKMA, the learning rate is set to be $\frac{1}{\lambda(t+100)}$, and λ is set to be 0.00005. For this comparison, the number of quantization bins is set to be 50. We also compare with linear SVM implemented in LIBSVM [7], where the parameter C is also cross validated.

The experiment is conducted on the PASCAL VOC 2007 image classification task, which has 20 object classes. We use a 3000-bin histogram of quantized SIFT codewords as features. The average precision (AP) results, training time and test time of different methods are compared in Table 1. The time of feature extraction is not included to report the training and test time for all the methods. 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. When more support vectors are kept, the NORMA algorithm can achieve better accuracy. However, the training time and test time increase accordingly. Its accuracy is still lower than our SIKIMA algorithm when keeping 1500 support vectors, and it takes more than 10 times longer to train (compared to the SIKMA (3)).

And for larger problems, the speedup of our SIKMA algorithm over batch will increase dramatically, and NORMA will be forced to make larger approximations at cost to classifier accuracy. We run another experiment to compare SIKMA and NORMA on 124, 000 images (including 20,000 bus images and 104,000 non-bus images; all are downloaded from Flickr groups). The test set includes 500 bus images and 20,800 non-bus images. Each image is represented as a 1,000 dimensional "bag

	Linear	NORMA (SV=500)	NORMA (SV=1000)	NORMA (SV=1500)	SIKMA (3)	SIKMA (6)	Batch (HIK)
AP	0.362	0.308	0.374	0.411	0.429	0.436	0.440
training time (seconds)	0.8	172.4	304.4	396.5	23.1	46.7	638.0
test time (seconds)	0.5	63.9	122.4	178.8	3.9	3.9	236.8

TABLE 1
Average precision (AP), training time and test time of different SVM training methods are compared on the PASCAL VOC 2007 image classification task. All the values are averaged over the 20 object categories. The histogram intersection kernel is used except the linear SVM. NORMA (SV=500) denotes the performance of the NORMA algorithm when at most 500 support vectors are kept. NORMA (SV=1000) denotes the performance when 1000 support vectors are kept. And NORMA (SV=1500) denotes the performance when 1500 support vectors are kept. SIKMA (3) denotes the SIKMA algorithm visits each training example three times. SIKMA (6) denotes each training example is visited six times.

	SIKMA	NORMA (SV=1000)	NORMA (SV=2000)	NORMA (SV=3000)
AP	0.598	0.469	0.483	0.500
training time (seconds)	81.5	2556.0	5050.2	7402.6
test time (seconds)	4.2	130.1	256.7	383.9

TABLE 2
Average precision (AP), training time and test time of different training methods are compared on a larger problem (124,00 training images and 21,300 test images). NORMA (SV=1000) denotes the performance when 1000 support vectors are kept; NORMA (SV=2000) denotes the performance when 2000 support vectors are kept; NORMA (SV=3000) denotes the performance when 3000 support vectors are kept.

of words" feature. Each training example is visited twice by both SIKMA and NORMA algorithms. Even keeping 3,000 support vectors, the performance of NORMA is poor, since many support vectors are dropped. Our SIKMA algorithm is 90 times faster compared to NORMA (3000).

When training with SIKMA, we need to use piecewise linear segments to approximate the continuous values. The approximation accuracy is controlled by the number of quantization bins. We perform an experiment to investigate how the performance is affected by the bin size. The results are summarized in Table 3 (each training example is visited three times). Training time and test time go up when we use more levels since we need to update more parameters. The best performance is achieved using 50 bins. There is not much change with more bins (some small drop is observed; it might be because our algorithm randomly picks training examples, hence randomness is introduced).

We apply the SIKMA algorithm to train group classifiers for the 103 Flickr groups listed in Section 2.1. For each Flickr group, there are 15,000 to 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. An AP score is calculated for each group. Fig. 3 shows the AP scores for all the 103 Flickr groups. The average AP over these categories is 0.433. We can see the SIKMA algorithm can produce reliable classifiers, which is very important for measuring image similarity. We use Flickr groups as the resource to learn image similarity partially because Flickr groups have a large number of images, which can be exploited to learn more accurate concept classifiers compared to learning with a small number of training examples. To show that more data helps, we do similar experiments as above, but with different percentages of images from each group as training data (10%, 20%, 50%, and 100% respectively). We keep the test data the same. Average AP values are shown in Fig. 2. Using more data significantly improves the classification performance.

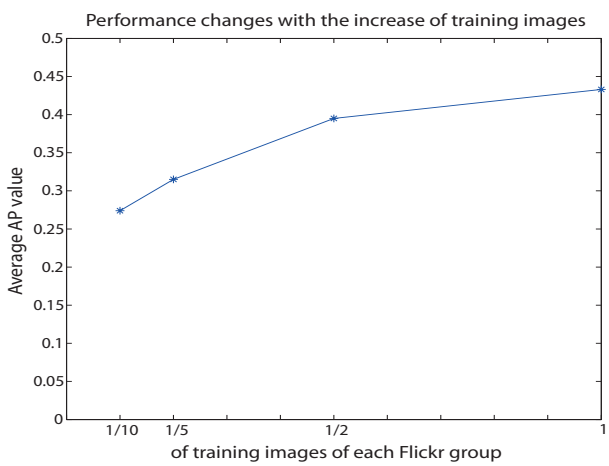


Fig. 2. The average AP values over the 103 Flickr categories. The X-axis denotes different percentages of training images (of each Flickr group) used at each time.

4.2 Evaluation of Learned Flickr Prediction Features

We evaluate our learned Flickr prediction features on image retrieval, on image clustering, and on image

bins	10	20	50	80	120	150	200
AP	0.401	0.410	0.429	0.426	0.427	0.424	0.425
training time (seconds)	9.9	12.8	23.1	25.3	35.0	38.1	54.2
test time (seconds)	2.7	3.2	3.9	4.3	4.6	4.8	5.1

TABLE 3
The AP values, training time, and test time of SIKMA using different number of bins. All the numbers are averaged over the 20 object categories. Each training example is visited for three times in this implementation.

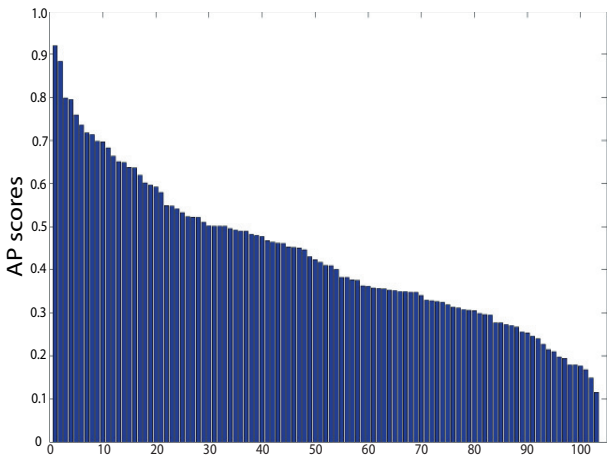


Fig. 3. The AP scores of the 103 Flickr groups (categories). For each category, There are 20,900 held-out examples (500 positive). The five groups which get the highest AP values are: laptop lunch, fireworks, pandas, socks and moon; the five groups which get the lowest AP values are: love, art, trees, ice and light.

classification. The results show that our learned features perform better than the conventional visual features on all these tasks.

4.2.1 Image retrieval and clustering

We measure image similarity by calculating the Euclidean distance between Flickr prediction features. The quality of similarity measurement is the most important factor in automatic organization. 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 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), which has been popularly used to evaluate image retrieval and organization methods [1], [43]. Each image is associated with a CD label and a set of keywords. A CD contains 100 visually similar images. Hence both CD labels and keywords are used as ground truth for matching. Each image has roughly 3-5 keywords.

We first evaluate our method on image query based matching. 30,000 Corel images are used as the test bed, among which 500 images are randomly chosen as query images. For each query image, the other images are ranked using different similarity measures. Images from the same CD in the Corel dataset share the same theme. Hence images that have the same CD label or at least one keyword in common are considered as correct matches; others are incorrect matches. An AP value is calculated for each query image. When using "visual features + Euclidean distance", the averaged AP value over all the query images is 0.110; when using "visual features + histogram intersection", the averaged AP value is 0.114; when using our learned similarity, the average AP value is 0.123. We can see using Flickr groups provides a more accurate measure of similarity. In Fig. 4, we show the 25 nearest neighbor images found for each of 4 image queries. Images are sorted by similarity in descending order from left to right, top to bottom. Two methods are compared: one is "visual features + Euclidean distance"; the other one is our learned similarity (also indicated as "Flickr prediction features"). We can observe that our method can find more relevant nearest neighbors. For example, for the top left "ship" image query, the nearest neighbor images found by our method are more likely to contain ship inside; while the nearest neighbors images found using visual features are more about mountains and animals. This is because our method can learn more powerful features from the Flickr image groups.

It is intuitive that the learned similarity works best when queries are related to the learned Flickr categories. When more Flickr categories are use, 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).

We perform an experiment to investigate how the matching performance changes with the change of the number of the Flickr groups. At each time, we randomly select a subset of Flickr groups (for example, 10). Then in the matching procedure, only the prediction scores of these groups are used to measure image similarity. We calculate the averaged AP value over the 500 query images mentioned above. There is much variation when different subsets of Flickr groups are chosen. We repeat sampling Flickr groups 15 times for each number of groups. The mean and standard deviation values for different numbers of groups are shown in Fig. 5. The performance increases with more Flickr groups. We can also observe that the gain is not so significant when increasing from 70 groups to 100 groups. This may

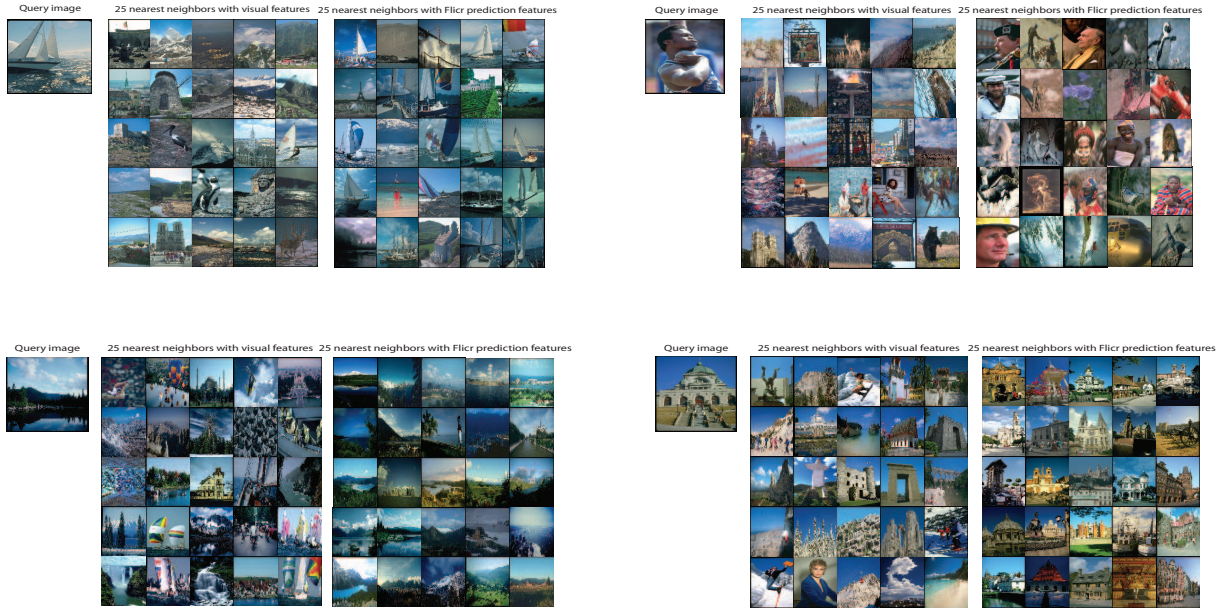


Fig. 4. This figure shows the nearest neighbor images found for each of the 4 image queries. For each query image, the left column shows the query itself; the center column shows the 25 nearest neighbor images found with visual features and the Euclidean distance; the right column shows the 25 nearest neighbor images found with our Flickr prediction features. The rank is from left to right, from top to bottom.

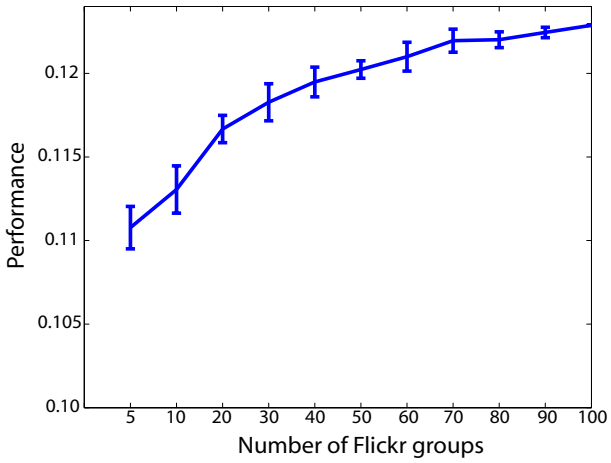


Fig. 5. The AP values of using different number of Flickr groups. For each number, we repeat 15 times to randomly select a subset of groups. Both mean and standard deviation values are shown in the figure.

suggest that Flickr groups share some common visual patterns, learning from a number of Flickr groups is also effective for out of sample image queries.

Image matching is a hard task. Human intervention can be very useful for finding similar images as humans can implicitly or explicitly suggest what features to look at. Relevance feedback is a very popular technique in image retrieval to enforce human intervention. A good similarity measure should allow effective feedback. In our experiment, using a subset of 10 CDs, we investigate the effectiveness of simple relevance feedback with different types of features. 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 in [24], which are chosen to provide 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 when they have the same CD label. We compute the average AP value over 25 randomly selected image queries. For both visual features and Flickr prediction features, we initialize the weights as ones. To simulate feedback, after each query, we select the top five negative examples and five randomly chosen positive examples among the top 50 ranked images and label them according to ground truth. We use this to train a weight vector to produce a new distance metric. This is a very simple metric learning procedure. With the feedback, we aim to minimize the following objective function:

$$\sum_{i}^{10} y_i w \bullet (x_q - x_i)^2 \tag{14}$$

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, and 0 otherwise. 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 over these 25 query images are compared in Fig. 6, with different rounds of relevance feedback. We can see that at the beginning, Flickr prediction features don't work much better than visual features. However, with relevance feedback, prediction

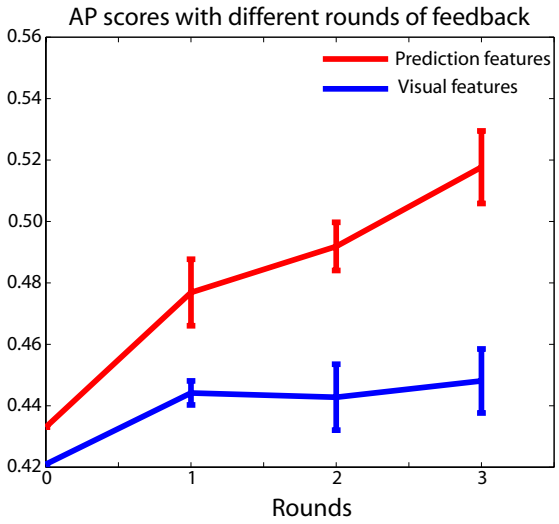


Fig. 6. The average AP values with three rounds of feedback. The red line shows the results with Flickr prediction features and the blue line shows the results with visual features.

features perform much better. This suggests prediction features are more effective for feedback. One possible reason is that Flickr prediction features are more compact (only with 103 dimensions), while visual features have thousands of dimensions. We cannot learn a good distance metric in high dimensional spaces when training examples are scarce.

Fig. 7 shows the nearest neighbor images without relevance feedback and the nearest neighbor images after the first round of relevance feedback for a query image (using our Flickr prediction features). The selected negative images are shown in red and the selected positive images are shown in green. We can see no false positives are found in the 45 nearest neighbor images after only learning with 5 positive and negative images.

One advantage of our method is that it can help us understand what makes two images similar. In Fig. 8, we show six pairs of similar Corel images. The text shows the Flickr groups which both of the images are likely to belong to. For example, the first pair of images are considered similar because they are both likely to belong to the following Flickr groups: mountains, castles, sheep, turtles, and cows.

Good image similarity measures can not only help image matching, but also help image clustering, which is also a popular way to organize images. We perform image clustering experiments on the images from the 10 CDs listed above. We represent these images with our prediction features and visual feature respectively. We cluster these 1000 images into 15 clusters in an unsupervised way (K-means). Each cluster is labeled with the most common CD label in this cluster. Then 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. This shows our learned similarity is also more effective for the image clustering task.

We may want to access images via text based queries (e.g., finding all the images associated with "cat"). In our approach, each Flickr group is described by several key words. When users input a word query, we can find the Flickr group whose description contains such a word. Then the corresponding Flickr group classifier is used to classify all the images to find relevant ones. We test this method on the Corel data set, with two queries "airplane" and "sunset". There are about 38,000 images in total, including 840 "airplane" images and 409 "sunset" images. We rank the images according to the Flickr group classification scores. 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. 9 according to the classification scores.

4.2.2 Classification

We can also use our Flickr group predictions as features for classification. In Table 4, we compare our prediction features with visual features. As implemented in [50], for the visual features, we train a chi-square kernel machine with the unified features (chi-square kernel is the state-of-the-art for histogram based image classification). Our group prediction features are not histograms, so we have to use an RBF kernel. Table 4 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. This shows our Flickr prediction features are also effective for the classification task.

5 Conclusions and Discussions

In this paper, we developed an approach to learn image similarity from Flickr groups. The motivation is that Flickr groups show how people would like to group similar images on the Internet. Then two query images can be considered similar if the are likely to belong to the same set of Flickr groups. We also described 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 memberships. Our experimental results provide strong evidence that such learned image similarity works better on many tasks such as image matching and unsupervised clustering than directly measuring similarity with visual features.

There are several explanations for the success of learned image similarity. First, the predictions are discriminative, and are trained using tens of thousands of positive and negative training images. In our procedure, interesting visual patterns that are beneficial for matching and classification are preserved, while the others are abandoned. By mapping a new query image to this space, we can measure how affinity with these visual

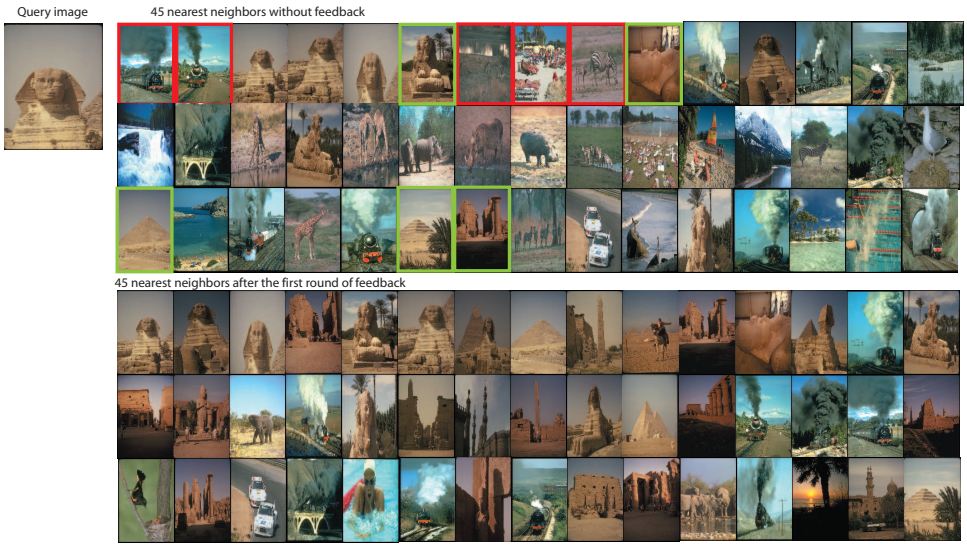
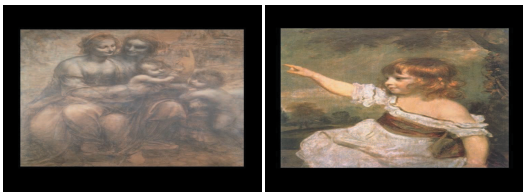


Fig. 7. The left column shows the query image; the center column shows the 45 nearest neighbors found with the Flickr prediction features, the five negative images (in red) and five positive images (in green) are selected for feedback; after one round of feedback, we get the 50 nearest neighbors shown in the right column.

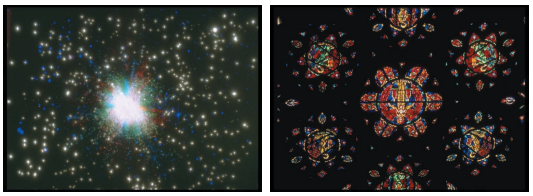
mountains(3.0) castles(1.2) sheep(1.2) turtles(0.9) cows(0.6)

dances(3.8) weddings(2.4) smiles(2.3) love(1.7) sports(1.4)

sports(2.6) dances(2.0) weddings(1.0) toys(0.5) horses(0.5)



painting(2.4) art(1.2) macro-flowers(0.9) hands(0.9) skateboard(0.6)



fireworks(15.6) Christmas(7.6) rain(4.0) water drops(2.5) candles(2.0)

painting(1.5) children(1.1) weddings(1.0) love(0.6) animals(0.6)

Fig. 8. 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 $100 \times p(\text{group} \mid \text{image1})p(\text{group} \mid \text{image2})$.

	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 4
The AP value with Flickr prediction features and visual features on PASCAL 2007 classification for each object class.

Fig. 9. 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.

patterns. Second, the Flickr prediction features are very compact. As in our implementation, they only have 103 dimensions, while traditional visual features may have thousands of dimensions. With compact representation, relevance feedback can be more effective because it deals with fewer parameters, as is confirmed by our experiments. Third, the predictions are trained to agree with human judgements, which seem to be based on semantics. This means that our predictions should be better at predicting concept-based similarity than pure 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] Vladimir Pavlovic Ameesh Makadia and Sanjiv Kumar. A new baseline for image annotation. In *European Conference on Computer Vision*, 2008.

[2] K. Barnard, P. Duygulu, and D.A. Forsyth. Clustering art. In *IEEE Conf. on Computer Vision and Pattern Recognition*, pages II:434–441, 2001.

[3] K. Barnard and D.A. Forsyth. Learning the semantics of words and pictures. In *Int. Conf. on Computer Vision*, pages 408–15, 2001.

[4] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(4):509–522, 2002.

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

[6] L. Bottou. *Large-scale kernel machines*. Mit Pr, 2007.

[7] 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.

[8] J. Cui, F. Wen, R. Xiao, Y. Tian, and X. Tang. EasyAlbum: an interactive photo annotation system based on face clustering and re-ranking. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, page 376. ACM, 2007.

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

[10] R. Datta, D. Joshi, J. Li, and J.Z. Wang. Image Retrieval: Ideas, Influences, and Trends of the New Age. *Pennsylvania State University Technical Report CSE*, pages 06–009, 2006.

[11] J. Deng, W. Dong, R. Socher, L.J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. *CVPR*, 2009.

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

[13] R. Fergus, L. Fei-Fei, P. Perona, and A. Zisserman. Learning object categories from google's image search. In *Proceedings of the 10th International Conference on Computer Vision, Beijing, China*, volume 2, pages 1816–1823, October 2005.

[14] K. Grauman and T. Darrell. The Pyramid Match Kernel: Discriminative Classification with Sets of Image Features. *Proc. ICCV*, 1(2):3, 2005.

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

[16] A.E. Johnson and M. Hebert. Using spin images for efficient object recognition in cluttered 3D scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(5):433, 1999.

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

[18] N. Kumar, A.C. Berg, P.N. Belhumeur, S.K. Nayar, C. Zhou, S. Lin, D. Bitouk, S. Dhillon, O. Cossairt, R. Ramamoorthi, et al. Attribute and Simile Classifiers for Face Verification. *ICCV*, 2009.

[19] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, 2006.

[20] T. Leung and J. Malik. Representing and recognizing the visual appearance of materials using three-dimensional textons. *IJCV*, 43(1):29–44, June 2001.

[21] L.J. Li and L. Fei-Fei. Optimol: automatic online picture collection via incremental model learning. *International journal of computer vision*, 88(2):147–168, 2010.

[22] L.J. Li, H. Su, E.P. Xing, and L. Fei-Fei. Object bank: A high-level image representation for scene classification & semantic feature sparsification. In *Neural Information Processing Systems*, 2010.

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

[24] N. Loeff and A. Farhadi. Scene Discovery by Matrix Factorization. In *Proceedings of the 10th European Conference on Computer Vision: Part IV*, pages 451–464. Springer-Verlag Berlin, Heidelberg, 2008.

[25] D.G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *IJCV*, 60(2):91–110, 2004.

[26] S. Maji and A.C. Berg. Max-Margin Additive Classifiers for Detection. ICCV, 2009.

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

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

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

[30] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *IEEE Conference on Computer Vision and Pattern Recognition, 2007. CVPR'07*, pages 1–8, 2007.

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

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

[33] Y. Rubner, C. Tomasi, and L.J. Guibas. A metric for distributions with applications to image databases. 1998.

[34] Y. Rui, T.S. Huang, M. Ortega, and S. Mehrotra. Relevance feedback: A power tool for interactive content-based image retrieval. *IEEE Transactions on circuits and systems for video technology*, 8(5):644–655, 1998.

[35] B.C. Russell, A. Torralba, K.P. Murphy, and W.T. Freeman. LabelMe: a database and web-based tool for image annotation. *IJCV*, 2008.

[36] C. Schmid and R. Mohr. Local grayvalue invariants for image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(5):530–535, 1997.

[37] F. Schroff, A. Criminisi, and A. Zisserman. Harvesting image databases from the web. *IEEE transactions on pattern analysis and machine intelligence*, 2010.

[38] S. Shalev-Shwartz, Y. Singer, and N. Srebro. Pegasos: Primal estimated sub-gradient solver for SVM. In *Proceedings of the 24th international conference on Machine learning*, pages 807–814. ACM New York, NY, USA, 2007.

[39] N. Snavely, S.M. Seitz, and R. Szeliski. Photo tourism: exploring photo collections in 3D. In *ACM SIGGRAPH 2006 Papers*, page 846. ACM, 2006.

[40] M.J. Swain and D.H. Ballard. Color indexing. *International journal of computer vision*, 7(1):11–32, 1991.

[41] M.J. Swain and D.H. Ballard. Color indexing. *Int. J. Computer Vision*, 7(1):11–32, 1991.

[42] Y. Tian, W. Liu, R. Xiao, F. Wen, and X. Tang. A face annotation framework with partial clustering and interactive labeling. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2007.

[43] K. Tieu and P. Viola. Boosting image retrieval. In *cvpr*, page 1228. Published by the IEEE Computer Society, 2000.

[44] A. Torralba. Contextual priming for object detection. *International Journal of Computer Vision*, 53(2):169–191, 2003.

[45] L. Torresani, M. Szummer, and A. Fitzgibbon. Efficient object category recognition using classemes. *Computer Vision–ECCV 2010*, pages 776–789, 2010.

[46] M. Varma and A. Zisserman. A statistical approach to texture classification from single images. *International Journal of Computer Vision*, 62(1):61–81, 2005.

[47] G. Wang and D. Forsyth. Object image retrieval by exploiting online knowledge resources. In *CVPR*, pages 1–8, 2008.

[48] G. Wang, D. Forsyth, and D. Hoiem. Comparative object similarity for improved recognition with few or no examples. In *CVPR*, 2010.

[49] G. Wang, A. Gallagher, J. Luo, and D. Forsyth. Seeing People in Social Context: Recognizing People and Social Relationships. *Computer Vision–ECCV 2010*, pages 169–182, 2010.

[50] G. Wang, D. Hoiem, and D. Forsyth. Building text features for object image classification. In *CVPR*, 2009.

[51] G. Wang, D. Hoiem, and D. Forsyth. Learning Image Similarity from Flickr Groups Using Stochastic Intersection Kernel Machines. *ICCV*, 2009.

[52] J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid. Local features and kernels for classification of texture and object categories: A comprehensive study. *IJCV*, 73(2):213–238, 2007.

[53] L. Zhang, Y. Hu, M. Li, W. Ma, and H. Zhang. Efficient propagation for face annotation in family albums. In *Proceedings of the 12th Annual ACM international Conference on Multimedia*, pages 716–723. ACM, 2004.

Gang Wang is an Assistant Professor in Electrical and Electronic Engineering at the Nanyang Technological University. He is also a Research Scientist of the Advanced Digital Science Center. He received the B.S. degree from Harbin Institute of Technology, China, in 2005 and the Ph.D. degree from the University of Illinois at Urbana-Champaign, Urbana. His research interests include computer vision and machine learning.

Derek Hoiem is an assistant professor in Computer Science at the University of Illinois at Urbana-Champaign (UIUC). Before joining the UIUC faculty in 2009, Derek completed his Ph.D. in Robotics at Carnegie Mellon University in 2007, advised by Alexei A. Efros and Martial Hebert, and was a postdoctoral fellow at the Beckman Institute from 2007-2008. Derek's research on scene understanding and object recognition has been recognized with a 2006 CVPR Best Paper award, a 2008 ACM Doctoral Dissertation Award honorable mention, and a 2011 NSF CAREER award.

David Forsyth (Fellow, IEEE) received the B.Sc. and M.Sc. degrees in electrical engineering from the University of the Witwatersrand, Johannesburg, South Africa, and the D.Phil. degree from Balliol College, Oxford, U.K. He has published more than 100 papers on computer vision, computer graphics, and machine learning. He is a coauthor (with J. Ponce) of Computer Vision: A Modern Approach (Englewood Cliffs, NJ: Prentice-Hall, 2002). He was a Professor at the University of California, Berkeley. He is currently a Professor at the University of Illinois at Urbana-Champaign. Prof. Forsyth was Program Cochair for IEEE Computer Vision and Pattern Recognition (CVPR) in 2000, General Cochair for CVPR 2006, and Program Cochair for the European Conference on Computer Vision 2008 and the CVPR 2011. He received an IEEE Technical Achievement Award in 2005.