Probabilistic Kernel Combination for Hierarchical Object Categorization

Ashish Kapoor
Raquel Urtasun
Trevor Darrell

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2009-16
http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-16.html

January 27, 2009

Probabilistic Kernel Combination for Hierarchical Object Categorization

Ashish Kapoor
Microsoft Research, Redmond
akapoor@microsoft.com

Raquel Urtasun and Trevor Darrell
University of California, Berkeley
{rurtasun,trevor}@icsi.berkeley.edu

Abstract

Recognition of general visual categories requires a diverse set of feature types, but not all are equally relevant to individual categories; efficient recognition arises by learning the potentially sparse features for each class and understanding the relationship between features common to related classes. This paper describes hierarchical discriminative probabilistic techniques for learning visual object category models. Our method recovers a nested set of object categories with chosen kernel combinations for discrimination at each level of the tree. We use a Gaussian Process based framework, with a parameterized sparsity penalty to favor compact classification hierarchies. We exploit structural properties of Gaussian Processes in a multi-class setting to gain computational efficiency and employ evidence maximization to optimally infer kernel weights from training data. Experiments on benchmark datasets show that our hierarchical probabilistic kernel combination scheme offers a benefit in both computational efficiency and performance: we report a significant improvement in accuracy compared to the current best whole-image kernel combination schemes on Caltech 101, as well as a two order-of-magnitude improvement in efficiency.

1. Introduction

A wide range of image descriptors have been proposed for visual object category recognition. Local feature descriptors (c.f. [21, 24]) have been shown to be effective: early models captured appearance and shape variation in a generative probabilistic framework [11], and more recent techniques have typically exploited methods based on SVMs or Nearest Neighbors in a bag-of-visual-words feature space [8, 25, 27, 33]. Several authors have explored correspondence-based kernels and their extensions [15, 20, 30, 33, 4], where the distance between a set of local feature descriptors—potentially including appearance and shape / position—is computed based on associating pairs of descriptors. Efficient intersection and search measures have been recently shown with such schemes [19, 22]. It is, however, likely that no single class of features will suffice to recognize all categories. Recent efforts have highlighted the value of various different approaches for learning weights on features or collections of features [13, 14, 10, 34].

Recent work on combining image descriptors to yield an optimal kernel matrix has shown impressive gains [5, 18, 29]. Learning an optimal weighting over image feature types can lead to a significant boost in classification accuracy as the invariance to various geometric, photometric, and structural transformations most relevant to recognize a category or group of categories can be inferred during training [29]. One disadvantage of these approaches is that instead of optimizing a kernel for the global classification problem they separately learn kernels for each individual class, which can lead to overfitting with small numbers of training examples and is relatively expensive in terms of computation time.

Approaches based on cross-validation [6] address these problems by directly carrying out an exhaustive search over the parameter space and selecting parameters that reduce global error on a validation set. However, this approach is computationally expensive and quickly becomes infeasible as the parameter space grows. Further, all of these methods are non-probabilistic and, with exception of [6], do not consider any hierarchy in classification; in contrast, we provide here a unified probabilistic model that is efficient in terms of computation and performance. Discovering a hierarchy amongst the class labels allows for further specialization of the classification model for different groups of categories [6, 16, 35, 23].

This paper addresses the problem of discovering a discriminative hierarchy for visual object category recognition. We develop a probabilistic multi-kernel recognition framework where kernel combination weights are learned via an evidence maximization criteria and sparse priors can encourage compact and efficient classifiers. Our approach performs weight inference in a multi-class setting and leads to significantly improved performance and computational efficiency when compared to previous techniques.

Initial results on visual category learning using a Gaussian Process (GP) multi-kernel formalism were reported in

[2].[1] Here we present a new method which extends such multi-kernel techniques to include a scheme for extremely fast multi-class training when compared to a traditional ensemble of 1-vs-all classifiers, a regularization term which significantly improves performance especially with small amounts of training data, and a hierarchical learning formulation which discovers nested sets of related classes leading to simpler intermediate classifiers and overall improved performance.

Given labeled training data, our method learns an optimal combination of kernels by maximizing the evidence of the observed object category labels. A significant advantage of GP regression is that leave-one-out estimates can be very efficiently computed, and can be utilized to relatively quickly learn a hierarchy of classes based on a confusion matrix. In contrast, traditional discriminative methods typically require retraining a classifier every time a single example is removed from the training set. Our method exploits the structural properties of GP regression for efficient multi-way classification.

In the experiments reported below, our method performs favorably to the kernel combination schemes of [29] and [5]. We restrict our evaluation to the publically available 'whole-image' kernels (not including the 'ROI kernels' reported in [5])[2]; focusing on the power of kernel combination and hierarchy formation we believe our method offers the best reported performance on Caltech datasets when compared on all available whole-image kernels.

2. Background: Combining Kernels with Gaussian Processes

As Gaussian processes are non-parametric models, their performance in regression and classification tasks depends on the covariance matrix or kernel that captures the similarity between the data points. Let $\mathbf{Y} = [\mathbf{y}_1, \cdots, \mathbf{y}_N]^T$ be the set of class labels. For simplicity in the discussion consider the binary case, $\mathbf{y}_i \in \{-1, 1\}$, and let $\mathbf{X} = [\mathbf{x}_1, \cdots, \mathbf{x}_N]^T$ be the set of input variables $\mathbf{x}_i \in \Re^Q$. The Gaussian Process prior is then defined as

$$p(\mathbf{Y}|\mathbf{X}) = \mathcal{N}(0, \mathbf{K}) \,, \tag{1}$$

with $\mathbf{K}$ the covariance matrix corresponding to a kernel (similarity measure). See [26] for a detailed treatment.

In [17] GP covariance functions based on the Pyramid Match Kernel [15] (and by extension, the Spatial Pyramid Match [20]) were introduced, and results in a supervised and active learning setting were reported. Other kernels used in SVM-based visual category recognition are also generally suitable for use as GP covariances. In [28] the composition of an RBF kernel, which is traditionally used in GP regression and has hyperparameters associated with the mapping, and a PMK kernel was used for human pose estimation.

In a multi-kernel approach the covariance is defined as a linear combination of covariance matrices [2]

$$\mathbf{K} = \sum_{i=1}^{M} \alpha_i \mathbf{K}^{(i)} \tag{2}$$

where $\boldsymbol{\alpha} = \{\alpha_1, \cdots, \alpha_M\}$ are the weight parameters, $\forall i, \alpha_i \geq 0$, and $\exists\ \alpha_i$ suth that $\alpha_i > 0$, at least one α_i is different from zero. The individual covariance matrices $\mathbf{K}^{(i)}$ are restricted to be positive definite, so that their linear combination is also positive definite.

Learning in the Gaussian Process framework consists of estimating the kernel hyperparameters, including the kernel weight parameters in the case of the multi-kernel in Eq. (2). Finding the right set of hyperparameters can be a challenge. Ideally we would like to marginalize over these hyperparameters. While approaches based on Hybrid Monte Carlo have been explored to perform this marginalization [31], such techniques are computationally expensive.

A more computationally efficient alternative is empirical Bayes, where the idea is to maximize the marginal likelihood or evidence. This methodology of tuning the hyperparameter is often called *evidence maximization*. This approach is computationally efficient and unlike cross-validation it does not require a validation set.

When using a Gaussian noise model, the log of the evidence can be written in closed form [26, 2]

$$\begin{aligned} \mathcal{L}(\boldsymbol{\alpha}) &= -\log p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\alpha}) \\ &= \frac{1}{2}\mathbf{Y}^T(\sigma^2\mathbf{I} + \mathbf{K})^{-1}\mathbf{Y} + \frac{1}{2}\log|\sigma^2\mathbf{I} + \mathbf{K}| + C \end{aligned}$$

where C is a constant, σ the noise variance, and $\boldsymbol{\alpha} = \{\alpha_1, \cdots, \alpha_M\}$ is the set of hyperparameters. Learning the parameters by maximum likelihood results in accurate classification [2] when the number of examples is large. However, when the number of training examples is small, and the number of kernels large, learning by maximum likelihood can result in overfitting.

3. Efficient Multi-Class Learning

In this section we describe several aspects of our new method, which extends the above formalism to incorporate a regularization term to avoid overfitting on small training sets, and includes a new algorithm for very efficient kernel combination in the multi-class case. Our method also incorporates a novel hierarchical learning technique, described in the following section.

[1] An anonymized version of this recently accepted journal paper is provided in the supplementary material.

[2] The segmentation implicit in the ROI kernels is shown in [5] to be extremely powerful, yeilding higher overall performance than with whole-image kernels, but these kernels are not publically available for all test splits so we were unable to try them with our method.

3.1. Learning from Small Number of Examples

To avoid overfitting, instead of finding the hyperparameters by maximum likelihood, we assume a prior distribution over the hyperparameters, $p(\boldsymbol{\alpha})$, and choose the maximum-a-posteriori (MAP) estimate. It can be easily shown that various choices of priors lead to different choices of regularization. For instance assuming a Gaussian and a Laplacian prior on $\boldsymbol{\alpha}$ leads to an $L2$ and $L1$ regularized formulation respectively; the latter is well known to enforce a degree of sparsity on the kernel weights. A parameterized form of regularization is known in the statistics literature as the *elastic net* [12]. We can write the optimization as

$$\begin{aligned} &\arg\min_{\boldsymbol{\alpha}} -\log p(\mathbf{Y}|\mathbf{X},\boldsymbol{\alpha}) + \gamma_1||\boldsymbol{\alpha}||_1 + \gamma_2||\boldsymbol{\alpha}||_2 \\ &\text{subject to:} \qquad \alpha_i \geq 0 \;\; \forall i \in \{1,..,M\}. \end{aligned}$$

Here, γ_1 and γ_2 are regularization constants for $L1$ and $L2$ norms respectively. The non-negativity constraints on $\boldsymbol{\alpha}$ ensure that the resulting $\mathbf{K}$ is positive-semidefinite and can be used in a GP formulation (or other kernel-based methods).

This objective can be minimized using non-linear optimization techniques, such as gradient descent. The optimization can be performed with multiple initializations to deal with the fact that we are optimizing a non-convex function and the log evidence has multiple local optima. The gradients of the negative log evidence, $\mathcal{L}(\boldsymbol{\alpha})$, are efficient to compute and can be written as:

$$\begin{aligned} \frac{\delta\mathcal{L}(\boldsymbol{\alpha})}{\delta\alpha_i} = &-\frac{1}{2}\mathbf{Y}^T\mathbf{A}^{-1}\mathbf{K}^{(i)}\mathbf{A}^{-1}\mathbf{Y} + \frac{1}{2}\text{Tr}(\mathbf{A}^{-1}\mathbf{K}^{(i)}) \\ &+ \gamma_i + \gamma_2 \cdot \boldsymbol{\alpha} \end{aligned}$$

where $\mathbf{A} = \sigma^2\mathbf{I} + \mathbf{K}$. In our implementation we use a gradient descent procedure based on the projected BFGS method using line search.

Our Gaussian Process framework provides conditional models that are probabilistic; since GPs marginalize over the feature weights, our model–in contrast to other state-of-the-art discriminative models such as SVMs [5, 29]–is less prone to overfitting. As shown in this paper, this results in a large increase of performance when dealing with small number of examples, greater than 7% in Caltech 101 (for 1 training image per class). GPs have structural properties with regard to multiclass classification and leave-one-out cross-validation that, as shown below, can be exploited to develop highly efficient algorithms for learning kernel combinations as well as discovering hierarchies.

3.2. Learning in Multi-Class Problems

Object categorization is typically a multiclass problem and consequently requires a multiclass extension of the kernel learning framework. Popular techniques include 1-vs-all or 1-vs-1 formulations, where outputs from multiple binary classifiers trained on 1-vs-rest and pairwise classification problems are combined respectively. Learning a kernel introduces additional complexity as the optimization procedure for kernel combination should consider all the labels and result in a single set of global parameters that are informative about the entire classification task. Learning a global set of parameters is in general non-trivial, and learning separate kernels for each binary subproblem has been proposed [29]. Despite the fact that such *classwise* parameterizations offer flexibility in modeling each individual class, these extrategies are more prone to overfitting than global ones when dealing with small number of examples. As shown below, global optimization of the parameters consistently outperforms classwise optimization in our experiments. Furthermore, classwise techniques require solving as many classification tasks as the number of classes; with large datasets such as Caltech-101 and Caltech-256 this means that learning has to be repeated 101 and 256 times respectively. While it is unclear how to overcome these issues in non-probabilistic approaches such as [29], GPs provide a principled and computationally efficient scheme of finding globally optimal parameters.

We first consider a 1-vs-all formulation of GP classifiers, where multiple binary classifiers correspond to each individual class. Similar to binary classifiers we optimize kernels weights by considering the log evidence, however, for the multiclass case we consider a joint log-likelihood over all the classifiers:

$$\mathcal{L}(\boldsymbol{\alpha}) \quad = \quad -\sum_i \log p(\mathbf{Y^{(i)}}|\mathbf{X},\boldsymbol{\alpha}) \; .$$

Here the sum is taken over all the class labels, and $\mathbf{Y}^{(i)}$ are the labels for i-th 1-vs-all problem. This joint likelihood corresponds to a probabilistic model that assumes that given the input images the binary outputs of 1-vs-all problems are independent. Note that, this assumption is well justified as given an image its class label is determined by the image content only. Further, this model allows us to optimize for a global set of kernel parameters that maximize the joint likelihood over all the class labels. Thus, instead of learning a kernel for every individual class, we can learn an optimal parameterization that is globally discriminative.

There are additional computational benefits of the above scheme. Note that in the proposed GP framework, given a test observation $\mathbf{x}_*$, the mean prediction for a binary classifier can be computed as:

$$\bar{y}_*^{(i)} \quad = \mathbf{k}(\mathbf{x}_*)^T\mathbf{A}^{-1}\mathbf{Y}^{(i)} \; , \tag{3}$$

where $\mathbf{k}(\mathbf{x}_*)$ is the kernel computed between the training and test data. The most expensive operation in such computation is the matrix inversion which is independent of the training labels $\mathbf{Y}$. Consequently, once the inverse is

computed, estimating predictions for 1-vs-all models corresponds to a multiplication with the relevant label vectors.

This is a significant advantage since the cost of training all the classifiers in a 1-vs-all formulation is the same as the cost of training a single classifier. This is especially beneficial in cases with large number of classes, and provides a significant advantage over other methods which separately need to train different classifiers per class. This observation readily extends to the kernel learning scenario with multiple classes. As before, the primary operation is a matrix inversion (computing $\mathbf{A}^{-1}$) that is independent of the labels. Thus, learning kernels for multiple class problems using the joint likelihood has similar cost as that of learning a kernel in a binary problem [3]. We show empirically in Section 5 that this scheme is extremely fast when compared to other state-of-the-art methods while providing superior classification performance.

4. Learning a Hierarchy Tree in Multiclass Problems

It is often the case for multiclass problems that some classes are typically more confused than others, e.g., discriminating bikes and motorbikes is more challenging than discriminating bikes and oranges. As a consequence, one can expect an increase of classification accuracy by training classifiers dedicated to discriminate the most similar classes. In this section we introduce an efficient hierarchical classification scheme that utilizes the confusion matrix to generate a hierarchy of classifiers, where at each level of the hierarchy similar classes are group together.

There have been some attempts at learning such hierarchy based on clustering confusion matrices [16, 6], which are typically generated with cross-validation. Leave-one-out is especially attractive method to generate confusion matrices. It is well known that in the limit the leave-one-out error is the unbiased estimator of generalization error [7]; consequently, finding a hierarchy to reduce the error on cross-validation results should lead to more accurate models. However, computing such a matrix can be expensive if a new classifier needs to be trained after each point is removed/replaced in the training set.

The structure of GP regression provides an elegant solution to compute a leave-one-out cross validation matrix. Most of the computation can be shared across each leave-one-out estimate; leave-one-out classifier outputs for all the training points can be found in one simple matrix operation. Specifically, the leave-one-out predictive means $\bar{\mathbf{y}}$ for all training data are given by [26]:

$$\bar{\mathbf{y}}_{LOO} = \bar{\mathbf{y}} - [\mathbf{A}^{-1}\mathbf{Y}]./\text{diag}(\mathbf{A}^{-1}) \tag{4}$$

[3] The computational cost is dominated by the $\mathcal{O}(N^3)$ cost of inverting $\mathbf{A}$, with N the number of examples.

Here, $\bar{\mathbf{y}}$ denotes the predictive mean on the training points when using all the data. Also note that the inverse $\mathbf{A}^{-1}$ was already computed during kernel learning. The computational complexity of this operation is only $\mathcal{O}(N^2)$, with N the number of training points. Further, as mentioned before this readily extends to the multiclass case as $\mathbf{A}$ is unchanged and the inverse can be reused across all the individual classes.

Once the confusion matrix C_m is computed, clusters of classes that are most confusable can be found. In this work, we use self-tuning spectral clustering [32] on the symmetrized confusion matrix

$$C_{sym} = \frac{1}{2}(C_m + C_m^T) \tag{5}$$

We let the self-tuning spectral method discover the correct number of clusters. Further, if a class is perfectly recognized using the leave-one-out scheme we consider it a leaf node at that level. New kernel weights for the root classifier are learned by merging all classes in each cluster; thus, the classifier at the root level learns to classify a data point into the discovered clusters. This process is recursively applied to each cluster of size greater than one until all the clusters have one class. Algorithm 1 describes details of the procedure to learn the object hierarchy. At test time, at each level of the hierarchy we first classify the cluster that each test example belongs to. This process is repeated until we arrive at a leaf of the hierarchy.

Algorithm 1 Learning Object Class Hierarchy

function TreeOut = LearnTree(ListClasses)

if (length(ListClasses) > 1) **then**
  Learn kernel parameters $\boldsymbol{\alpha}$ for ListClasses
  Compute leave-one-out confusion matrix C_m
  Cluster classes using $C_{sym} = (C_m + C_m^T)$
  Learn kernel parameters $\boldsymbol{\alpha}$ using the clustered classes
  TreeOut.nodeparams = $\boldsymbol{\alpha}$
  TreeOut.nodeclasses = Learned clusters
  for i = 1 to number of clusters **do**
    TreeOut.child(i) = LearnTree(Cluster(i))
  end for
else
  TreeOut.nodeparams = ListClasses
end if
Return TreeOut

5. Experiments and Results

In this section we report results demonstrating the computational efficiency and high performance accuracy of probabilistic kernel combination, the ability of the proposed

framework to discover object hierarchies, and the use of recovered hierarchies for improved classification with small numbers of training examples.

Datasets and Implementation Details

We performed experiments on two different datasets that are considered standards for the object categorization task: the Caltech-101 data set and the Caltech-256 data set (which is a superset of Caltech-101). Our experiments use 30 images per class from Caltech-101 dataset (3030 images in total), and are the same as the ones used in [29]. For Caltech-256 [9] we use the kernels of [1]. The training and the test split consists of 10 and 25 images per class.

We consider various shape and appearance features and sampling strategies, which are useful to capture the intra-class variation present in the Caltech-101 and Caltech-256 images. Specifically, we look at the following eight combinations of matching kernels and features:

1. **AppColour:** SIFT descriptors are extracted for each component of the HSV color space representation of the image, with all features sampled on a regular grid and at four fixed scales. These are quantized into visual words, and the pyramid kernel is applied per word in the space of image coordinates. See [5] for details.

2. **AppGray:** Same as AppColour, except features are extracted from the grayscale images.

3. **Shape 180**: Histograms of oriented gradients are matched using a spatial pyramid kernel. Edges are computed using the Canny edge detector followed by Sobel filtering for computing the gradients; the gradients are discretized into orientation histogram bins in the range [0, 180] with soft voting. See [5] for details.

4. **Shape 360:** Same as Shape 180 except that the orientation bins are in the range [0, 360].

5. **GB:** The Geometric Blur feature of [3] is extracted at sampled edge points. For the kernel values, the exact correspondences are computed based on the average minimum distance between points in the two input sets of features, as in [33].

6. **GBdist:** Same as GB, except the feature representation has an additional geometric distortion term.

7. **Dense PMK:** Pyramid Match Kernel (PMK) with uniformly shaped pyramid bins, using SIFT descriptors extracted densely from the images at every 8th pixel from a region of 16 pixels in diameter, with each SIFT descriptor concatenated with its normalized image position. PCA is used to reduce the dimensionality of the SIFT descriptors to 10, yielding features having a total of 12 dimensions. See [15] for details.

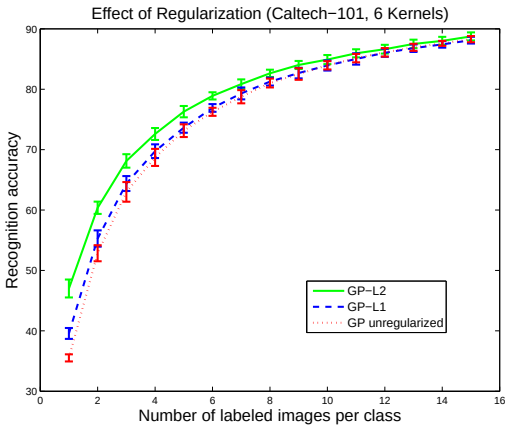


Figure 1. Performance comparison of different regularized and unregularized version of the probabilistic kernel combination scheme. A strong regularization results in significantly higher gains specially when the amount of labeled data is sparse.

8. **Spatial PMK:** The spatial variant of Dense-PMK. We take the same raw SIFT features, but quantize them into visual words, and then build one pyramid per word, each with uniform bins in the space of image coordinates. See [20] for details.

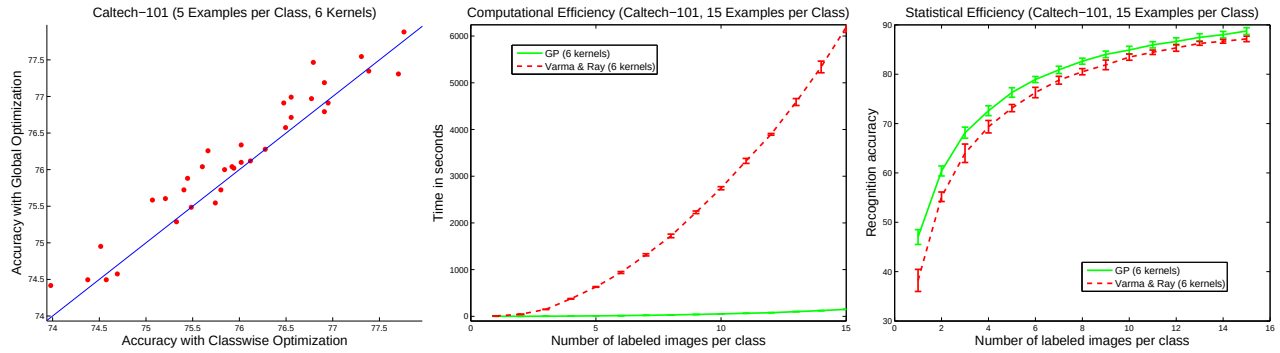
The kernel matrices for both datasets using each of these image descriptors were provided directly by the respective authors. In the experiments below we use either four, six or eight kernels to compare against existing approaches and refer to the group of first four, six and eight kernels from the list mentioned above.

For comparison against Varma and Ray [29], we used code provided by the authors with the parameters setting they provided. For learning kernels in GP models, we randomly initialized the kernel parameters and set the noise model variance to $\sigma = 10^{-10}$. These parameter values worked well; we experimented with other initialization schemes but found that the kernel learning was fairly insensitive to the initialization.

In all our experiments we follow the standard testing protocol, where a given number of training images (say 15) are taken from each class at random, and the rest of the data is used for testing. The mean recognition rate per class is used as a metric of performance. This process is repeated 10 times and the average correctness rate is reported. Finally, all the experiments are performed using MATLAB on a 64-bit windows machine with dual Intel Xeon 3.0 Ghz processors and 8 GB of RAM.

Effect of Regularization

First, we study how regularization of the log evidence effects classification performance. Figure 1 demonstrates recognition performance of different regularized and un-



(a) (b) (c)

Figure 2. (a) Comparison of accuracy obtained when optimizing kernels globally versus class wise. Comparison of (b) computational efficiency and (c) statistical efficiency for learning using GPs and Varma & Ray [29] that uses class wise optimization with SVMs.

Table 1. Time required and accuracy achieved on Caltech-101 using 15 labeled examples per class.

Method	Kernels	Time in Seconds	Accuracy
GP tree	8	305.49 ± 29.10	90.71 ± 0.58
	6	206.87 ± 36.82	89.64 ± 0.54
	4	133.04 ± 7.75	81.46 ± 0.93
GP	8	267.51 ± 3.45	88.81 ± 0.69
	6	149.44 ± 1.35	88.74 ± 0.69
	4	91.31 ± 1.01	80.32 ± 1.16
Varma & Ray[29]	6	6192.05 ± 113.65	87.15 ± 0.56
	4	4463.81 ± 165.84	79.88 ± 0.65
SVM-CV[4][6]	4	≥ 4463.81	81.00 ± 0.80

regularized probabilistic schemes averaged over 10 different splits. As shown in the figure regularization results in a significant improvement in performance, especially L2; the average gain is very high for small number of examples ($\approx 12\%$ for 2 examples per class). With two examples per class, the average accuracy, 47%, is higher than many prior methods that used a much larger number of label examples.

Computational Efficiency and Accuracy

Next, we explore the performance difference when parameters are trained globally versus trained separately for each class. Figure 2(a) shows a scatter plot where each point represents test accuracy obtained on a single train-test split of Caltech-101 data with 5 labeled examples per class. The figure illustrates performance on 35 different train-test splits when combinations of six kernels are learned. Most of the points lie above the diagonal, which suggests that training parameters globally is better than classwise training. To judge the significance of the results we performed a paired-t test and found the performance different to be significant at $p = 10^{-3}$ level. By jointly maximizing parameters for all classes the classifier learns representations that are maximally informative with respect to all the classes simultaneously, as is evident from superior performance across all three choice of kernel combinations. Classwise training has been the method of choice as non-probabilistic alternatives such as SVMs do not have straightforward formulations to optimize the weights globally—however, as described above, GPs avoid this problem by forming the joint likelihood of all the labels given the training data.

We also compare the computational efficiency of our approach with the scheme of Varma and Ray [29]. Figure 2(b) shows the time required to learn the kernel combination with six kernels. The method of Varma and Ray [29] learns kernels separately for each class using the one-vs-all formulation of binary SVMs and takes significantly longer time than the probabilistic combination based on GP. The GP-based approach learns the kernels simultaneously for all the classes and has clear computational advantages vs. training one-vs-all classifiers. In terms of accuracy GP-based formulation significantly outperforms the SVM formulation (see Figure 2(c)).

It has been suggested that performance of SVM based kernel combination can be improved by using cross-validation. Bosch et al. [6] perform a local search for kernel combination parameters around an initial solution found by Varma and Ray [29]. Note that the time required to run such an approach is lower bounded by the time required to first optimize SVM parameters and quickly becomes infeasible as we increase the number of kernels. For example, using the strategy of [6], for eight kernels on Caltech-101 $101 * 21^8 \approx 3.8$ trillion SVMs will need to be trained. Table 1 summarizes comparison of time required and the test accuracy of our approach with other methods. We can easily see that the GP based method has clear computational and statistical advantages.

Hierarchical Classification

We performed experiments to investigate whether learning object class hierarchies can improve classification performance. Table 1 shows the running time and average accuracies obtained by our hierarchical approach. Note that

[4] Accuracy reported from [6] and includes ROI detection.

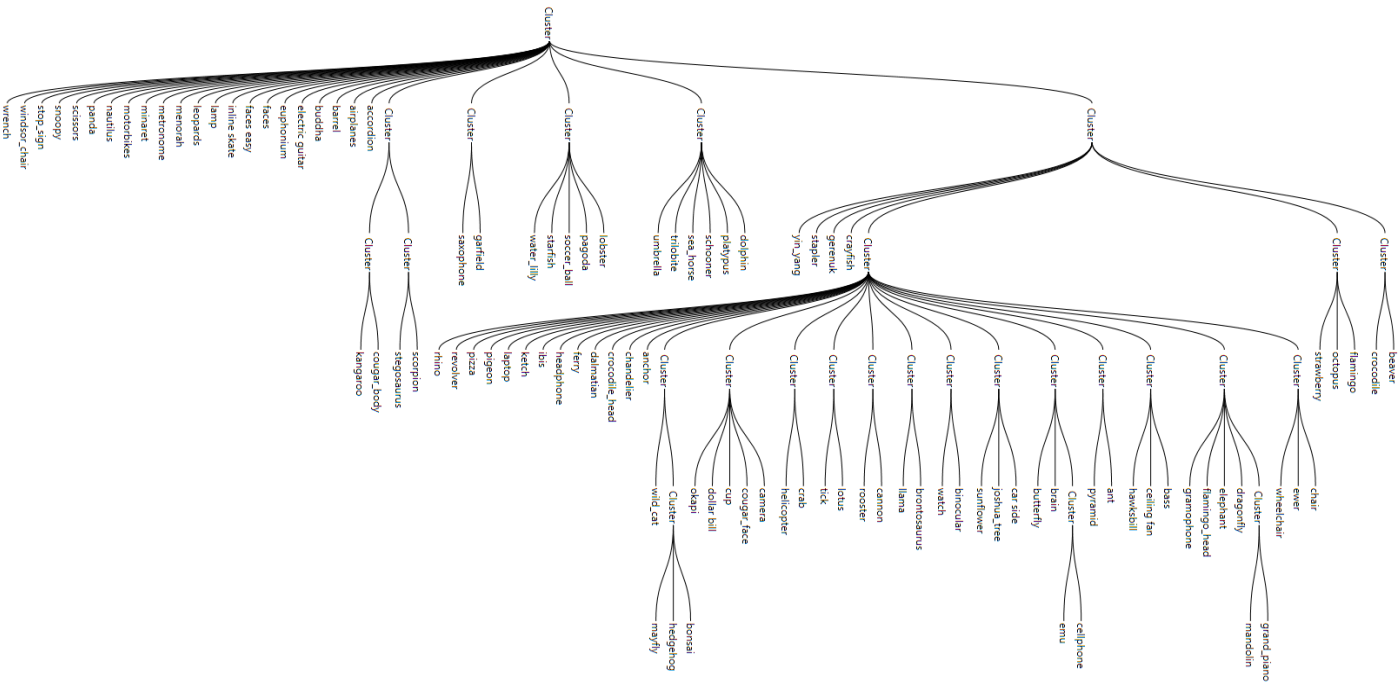


Figure 3. One instance of learnt object class hierarchy using Caltech-101 data (15 examples per class, 8 Kernels)

the running time to learn such a hierarchy is just slightly higher than non-hierarchical probabilistic kernel combination, and is significantly lower than SVMs, while the performance is improved. Figure 3 shows an example of a hierarchy learned by our technique. Note that the learned hierarchy is computed using a confusion matrix; a class is considered a cluster by itself if it does not get confused. Thus, the level at which a class becomes a leaf is an indicator of how difficult a class is. For instance, classes like faces, airplanes, motorbikes are found to be the easiest to recognize and become leaves at the first level. Note, that faces, airplanes and motorbikes comprise of three of four categories of Caltech-4 data and this observation is consistent with other prior work that reported very good accuracies with single descriptors [15]. The hardest categories are grand piano, mandolin, cellphone, emu, bonsai, hedgehog and mayfly. We find some of the categories that have similar visual properties are grouped together. For instance, wheel chair and chair are challenging classes that are grouped at the second to the last level. However, we also note that two categories which are traditionally confused in PMK-based approaches, ketch and schooner, were more easily recognized in the multi-kernel setting; while the PMK and its spatial variant had difficulty classifying those categories, the AppColour, AppGray, Shape 180 and Shape 260 did provide very good discriminative features. We must note that the learned hierarchy is a *discriminative* hierarchy and not a *semantic* one; no clear division along semantic categories is discernable.

Similarly, we also learned a hierarchical model on Caltech-256 data. We used the 4 whole-image kernels (AppColour, AppGray, Shape 180, Shape 260) available from [1], where a single train-test split with 10 training examples and 25 test examples per class is provided. The learning time for a full hierarchical model was 923.12 seconds despite the large number of classes and we obtained a test accuracy of 46.36%. The non-hierarchical model obtained an accuracy of 44.20% (647.61 seconds) outperforming the method of Varma et al. [29] which achieved 43.03% (11.23 hours), using these 4 kernels. Bosch et al. [6] reports higher performance using additional kernel matrices not available to us at this time. Our results confirm that when compared on the same set of input kernel matrices, our method outperforms the kernel combination formulations reported in [29].

Comparison with the State-of-the-Art

Figure 4 shows a comparison of the accuracy obtained with competing whole-image-kernel approaches. We obtain significantly better results than all methods including a non-regularized kernel learning scheme using GPs. The kernel combination of Bosch et al. [6] is slightly worse than ours when tested on the same kernels; we note that their reported performance is significantly higher (98%) when used with the full set of their innovative segmentation-mask (ROI) kernels. We were not able to run our method with the ROI kernels since only one split was publically available. (Each test split requires a recomputed set of kernel matricies cor-

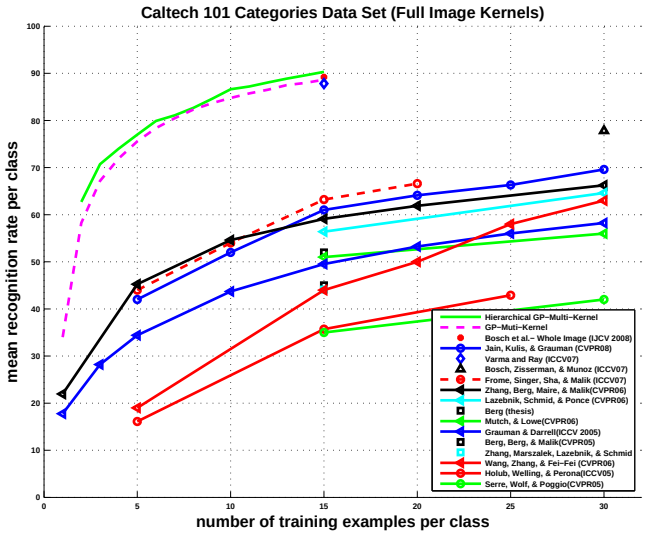


Figure 4. Performance comparison against existing methods on Caltech-101 that use whole image kernels.

responding to selected test image ROI under their scheme). Also, note that we can train a full hierarchy for Caltech-101 in as few as 10 minutes on a single CPU machine. Further, note that we achieve over 64% accuracy with just 2 examples per class which is better than most of the methods that use far more images to train. This suggests the utility of the proposed probabilistic approach to learn good object category models with sparse data.

6. Conclusion

We have presented a discriminative probabilistic framework based on Gaussian Processes that performs kernel combination and learns a hierarchy of visual categories. Besides providing a principled theoretical methodology for regularized kernel combination, the proposed scheme has significant computational advantages. By exploiting the structure of GPs the method can learn globally optimal kernel combinations for multiclass problems and discover hierarchies very quickly. The empirical experiments indicate two orders of magnitude in speed with additional boost in classification accuracy.

We plan extensions of the framework where we both recognize and detect objects from images. By incorporating such *region of interest* detection schemes we should be able to learn better object categorization models. We also plan to extend the model to handle multiple objects in the same image and explore sparse GP techniques for large datasets.

References

[1] *http://www.robots.ox.ac.uk/ vgg/research/caltech/*. 5, 7

[2] Anonymous. Gaussian Processes for object categorization. *In submission IJCV*, 2009. 1, 2

[3] A. Berg and J. Malik. Geometric blur for template matching. In *CVPR*, 2001. 5

[4] A. Bosch, A. Zisserman, and X. Munoz. Representing shape with a spatial pyramid kernel. In *CIVR*, 2007. 1

[5] A. Bosch, A. Zisserman, and X. Munoz. Representing shape with a spatial pyramid kernel. In *CIVR*, 2007. 1, 2, 3, 5

[6] A. Bosch, A. Zisserman, and X. Muoz. Image classification using rois and multiple kernel learning. *In submission IJCV*, 2008. 1, 4, 6, 7

[7] T. Evgeniou, M. Pontil, and A. Elisseeff. Leave one out error, stability, and generalization of voting combinations of classifiers. *Machine Learning*, 2004. 4

[8] B. T. F. Moosmann and F. Jurie. Fast discriminative visual codebooks using randomized clustering forests. In *NIPS*, 2007. 1

[9] L. Fei-Fei, R. Fergus, and P. Perona. One-shot learning of object categories. *IEEE Transaction on Pattern Recognition and Machine Intelligence*, 2006. 5

[10] P. Felzenszwalb, D. McAllester, and D. Ramanan. Discriminatively trained, multiscale, deformable part model. In *CVPR*, 2008. 1

[11] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *CVPR*, 2003. 1

[12] J. H. Friedman. Fast sparse regression and classification. Technical report, Standford, 2008. 3

[13] M. Fritz, B. Leibe, B. Caputo, and B. Schiele. Integrating representative and discriminant models for object category detection. In *ICCV*, 2005. 1

[14] A. Frome, Y. Singer, F. Sha, and J. Malik. Learning globally-consistent local distance functions for shape-based image retrieval and classification. In *ICCV*, 2007. 1

[15] K. Grauman and T. Darrell. The pyramid match kernel: Discriminative classification with sets of image features. In *ICCV*, 2005. 1, 2, 5, 7

[16] G. Griffin and P. Perona. Learning and using taxonomies for fast visual categorization. In *CVPR*, 2008. 1, 4

[17] A. Kapoor, K. Grauman, R. Urtasun, and T. Darrell. Active learning with Gaussian Processes for object categorization. In *ICCV*, 2007. 2

[18] A. Kumar and C. Sminchisescu. Support kernel machines for object recognition. In *ICCV*, 2007. 1

[19] C. H. Lampert, M. B.Blaschko, and T. Hofmann. Beyond sliding windows: Object localization by efficient subwindow search. In *CVPR*, 2008. 1

[20] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, 2006. 1, 2, 5

[21] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2), 2004. 1

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

[23] M. Marszałek and C. Schmid. Constructing category hierarchies for visual recognition. In *European Conference on Computer Vision*, 2008. 1

[24] K. Mikolajczyk and C. Schmid. Indexing Based on Scale Invariant Interest Points. In *ICCV*, 2001. 1

[25] D. Nister and H. Stewenius. Scalable Recognition with a Vocabulary Tree. In *CVPR*, 2006. 1

[26] C. E. Rasmusen and C. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006. 2, 4

[27] J. Sivic and A. Zisserman. Video Google: a text retrieval approach to object matching in videos. In *ICCV*, 2003. 1

[28] R. Urtasun and T. Darrell. Local probabilistic regression for activity-independent human pose inference. In *CVPR*, 2008. 2

[29] M. Varma and D. Ray. Learning the discriminative power-invariance trade-off. In *ICCV*, 2007. 1, 2, 3, 5, 6, 7

[30] C. Wallraven, B. Caputo, and A. Graf. Recognition with local features: the kernel recipe. In *ICCV*, 2003. 1

[31] C. Williams and D. Barber. Bayesian classification with Gaussian Processes. *IEEE Transaction on Pattern Recognition and Machine Intelligence*, 20(12):1342–1351, 1998. 2

[32] L. Zelnik-Manor and P. Perona. Self-tuning spectral clustering. *Eighteenth Annual Conference on Neural Information Processing Systems, (NIPS)*, 2004. 4

[33] H. Zhang, A. Berg, M. Maire, and J. Malik. SVM-KNN: Discriminative nearest neighbor classification for visual category recognition. In *CVPR*, 2006. 1, 5

[34] Q. Zhu, L. Wang, Y. Wu, and J. Shi. Contour context selection for object detection: A set-to-set contour matching approach. In *ECCV*, 2008. 1

[35] A. Zweig and D. Weinshall. Exploiting object hierarchy: Combining models from different category levels. In *ICCV*, 2007. 1