

Learning to Combine Kernels for Object Categorization

Deyuan Zhang (Corresponding author)

School of Computer Science and Technology, Harbin Institute of Technology

Mailbox 319, Harbin Institute of Technology, China

Tel: 86-451-8641-332-284 E-mail: dyzhang@insun.hit.edu.cn

Bingquan Liu, Chengjie Sun & Xiaolong Wang

School of Computer Science and Technology, Harbin Institute of Technology

Mailbox 319, Harbin Institute of Technology, China

Tel: 86-451-8641-3322 E-mail: {liubq, cjsun, wangxl}@insun.hit.edu.cn

Received: March 17, 2011

Accepted: April 13, 2011

doi:10.5539/cis.v4n3p116

This work was funded in part by the National Natural Science Foundation of China (Grant No. 60973076) and the Special Fund Projects for Harbin Science and Technology Innovation Talents (2010RFXXG003).

Abstract

Kernel classifiers based on the hand-crafted image descriptors proposed in the literature have achieved state-of-the-art results in several dataset and been widely used in image classification systems. Due to the high intra-class and inter-class variety of image categories, no single descriptor could be optimal in all situations. Combining multiple descriptors for a given task is a way to improve the accuracy of the image classification systems. In this paper, we propose a filter framework “Learning to Align the Kernel to its Ideal Form(LAKIF)” to automatically learn the optimal linear combination of multiple kernels. Given the image dataset and the kernels computed on the image descriptors, the optimal kernel weight is learned before the classification. Our method effectively learns the kernel weights by aligning the kernels to their ideal forms, leading to quadratic programming solution. The method takes into account the variation of kernel matrix and imbalanced dataset, which are common in real world image categorization tasks. Experimental results on Graz-01 and Caltech-101 image databases show the effectiveness and robustness of our method.

Keywords: Kernel Methods, Image Classification, Multiple Kernel Learning

1. Introduction

Classifying multiple object categories is a challenging task for computer vision systems. Although many hand-craft features and pattern classification algorithms have been proposed to tackle the problem, it is still far from human vision systems. The main difficulty lies in the large intra and inter class variation caused by issues such as ambiguities from clutter background, various poses, different lighting conditions, possible occlusion, etc. Many image descriptors and the corresponding classification methods have been proposed in the literature, and gained some success in some applications, such as color histogram(Swain and Ballard, 1991) for content based image retrieval, SIFT feature(Lowe, 2004) for object detection. But for the general image classification system, no single feature is sufficient for handling diverse objects of broad categories.

Based on the rigid theories, Support Vector Machines(SVM)(Cortes and Vapnik, 1995) have become popular in classification problems and succeeded in many image classification tasks, especially several object categorization benchmark datasets recently. Since no single image feature can fit every image dataset, combining multiple image descriptors using SVM is a promising research direction. In SVM methodology, combining multiple kernels automatically is a promising direction to improve the recognition accuracy.

Automatically learning the weight of kernel combination has been explored in the literature, roughly categorized into “filter” based or “wrapper” based. “Filter” methods are to optimize a specific kernel evaluation measure before SVM training which is not relevant to the SVM decision function. “Wrapper” methods learn the weight

and the SVM decision function simultaneously, and they are explored in machine learning fields, namely Multiple Kernel Learning(MKL)(Varma and Ray, 2007). The major difference of these methods is their optimized objective function.

Compared to “wrapper” methods, “filter” methods have lower training time complexity, but have far worse test performance. The reason is that the kernel evaluation measure is not related to the SVM decision function, thus the learned kernel weight is affected by the kernel values more easily. For the object categorization task, there exist two main obstacles for “filter” methods. Firstly, because the image features that proposed in the literature are variable, and kernel value of image pairs are much different, thus are more influenced by the variance of the kernel matrix combined; second, the training image dataset is often imbalanced, and most of “filter” methods suffer from imbalanced training dataset.

In this paper, we propose Learning to Align the Kernel to its Ideal Form(LAKIF) framework for learning the optimal combination of kernels. It is a “filter” framework, and our aim is achieving comparable performance with “wrapper” methods while keeping the training efficiency. The basic idea of this paper is to define an ideal form for each kernel, and learn to align the kernel to its ideal form. We propose a novel kernel normalization technique to make the kernel more robust, and develop a simple and fast approach for the kernel matrix learning problem based on quadratic programming. The framework handles imbalanced training data, which is common in object recognition tasks.

2. Related Work

There exist many kernel combination techniques and image classification methods, and only the related linear kernel combination technique is reviewed in the paper. Kernel combination methods roughly fall into three categories.

The first direction is combining multiple kernels using heuristic weights or brute force search. Grauman and Darrell (2005) used Pyramid Match Kernel(PMK) to match a set of orderless features, which can be viewed as combining multiple codebooks using heuristic weights. Lazebnik et al. (2006) proposed Spatial Pyramid Kernel that combines the spatial information of the image. They divide the image into finer grids and combine the grids into the final decision using heuristic weights. Gehler and Nowozin (2009) argue that the averaged kernel or product kernel often yields good results in several benchmark datasets, even though the simplicity of the method. Sande, Gevers and Snoek (2010) use the average kernel to combine several state of the art image features, and achieving the first place in the Classification Task of the PASCAL Visual Object Classes Challenge 2008. The method of fixed kernel weights combination needs the carefully designed kernels, thus resulting in the limited applications. Although using heuristic weights works well for combining some image features, the features is restricted in some identical form and lacked flexibility. In order to make the method more flexible, the weights of the kernels must be determined automatically, not the carefully designed kernels. The simple and naive way is determine the parameter using brute force search. Bosch et al. (2007) search the kernel weight using brute force over validation set and achieves state of the art results on several datasets. The brute force search tries groups of parameters for the best performance. When the number of kernels is large or the image dataset grows, the brute force search is not practical.

The second direction is to optimize a specific kernel evaluation measure before the SVM training. We call these methods “filter” methods because the kernel weight is determined before the SVM training. The idea of the filter methods is to align the current kernel to its “ideal” form, and maximize or minimize the alignment function result in the optimal kernel weights. Cristianini et al. (2002) proposed Kernel Target Alignment (KTA) to learn the ideal kernel. He et al. (2008) proposed a fast kernel learning scheme to learn the optimal weights of the image's spatial layout. The methods above is treats the loss of every kernel value of image pairs identical. Lin et al. (2007) do not learn a global optimal kernel weights, and learn the kernel weights for every training image by optimizing localized kernel alignment.

The third direction is to learn the kernel weight and SVM decision function simultaneously, which we called “wrapper” methods. The “wrapper” methods are explored much in the literature, namely Multiple Kernel Learning (MKL). The major difference of these multiple kernel learning algorithms is the optimization function or optimization methods. Kumar and Smirchisescu (2007) introduced the support kernel machine proposed by Bach et al. (2004) for combining the PMK and SPM kernel automatically. Varma and Ray (2007) think that for a specific classification task, the key is to determine the trade-off of discriminative power and invariance, and proposed a gradient based method to learn the trade-off. Yang et al. (2009) divide each category of training images into groups and trains distinct weights for every group.

3. Learning the combination of sub-kernels

3.1 Problem formulation

Consider binary classification problem with the training images $x_i (i=1,2,\dots,l)$ and their corresponding labels $y_i (y_i \in \{-1,+1\})$. The SVM minimize the following function:

$$\begin{aligned} \min J(\mathbf{w}, b, \xi) &= \frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum_{j=1}^l \xi_j \\ \text{s.t. } y_j (\mathbf{w}^T \Phi(\mathbf{x}_j) + b) &\geq 1 - \xi_j \\ \xi_j &\geq 0 \end{aligned} \quad (1)$$

In SVM methodology, kernel function $K(x_i, x_j) = \phi(x_i)\phi(x_j)$ is employed to compute the similarity of images x_i and x_j . For a tested image x , the decision function of SVM is:

$$f(x) = \text{sign}(\sum_{i=1}^l \alpha_i^* K(x, x_i) + b) \quad (2)$$

where the coefficients α_i^* are obtained by maximizing the following function:

$$W(\alpha) = -\frac{1}{2} \sum_{i=1}^l \sum_{j=1}^l \alpha_i \alpha_j y_i y_j K(x_i, x_j) + \sum_{i=1}^l \alpha_i \quad (3)$$

under the constraints:

$$\sum_{i=1}^l \alpha_i y_i = 0 \quad \text{and} \quad C \geq \alpha_i \geq 0 (i = 1, 2, \dots, l)$$

It is a quadric programming problem, and can be solved very efficiently using decomposition techniques (Joachims, 1999) or SMO algorithm (Platt, 1999).

Given N sub-kernel functions $K_1, \dots, K_N$ computed on different image features, the kernel K can be formulated as:

$$K(x_i, x_j) = \sum_{u=1}^N d_u K_u(x_i, x_j) \quad (4)$$

where $d_u \geq 0$ and $\sum_{u=1}^N d_u = 1$. The weight d_u measures the importance for discrimination. Our task is to learn the weight d_u before training SVM. The following two subsections introduce our framework: kernel normalization technique is described in section 3.2; the kernel weight learning algorithm is proposed in section 3.3.

3.2 Kernel normalization

The kernel itself measures the capacity of the classification performance. In the ideal case, the kernel transformation $K(x_i, x_j) = tK(x_i, x_j) + s (t > 0, s > 0)$ does not affect the decision function. And in practice by tuning the parameter C of SVM, the kernel transformation does not affect the performance. When applied to combining multiple kernels, the filter methods of kernel weight learning suffer from the kernel variation. It is obvious that transforming some sub-kernel functions will change the decision function of the combined kernel K . In order to make the learning deal with this problem, we must normalize each sub-kernel:

$$K_u = n_u \overline{K_u} \quad (5)$$

Where the $\overline{K_u}$ is the normalized form of the kernel K_u , and n_u is the scaling factor. The normalization step should be robust to sub-kernel transformation. That is, when we transform a sub-kernel $K'_u = tK_u + s (t > 0, s > 0)$, the normalized kernel form $\overline{K_u}$ is the same, resulting in the decision function f on combined kernel K should not change. Previous “filter” based kernel matrix learning algorithm can tackle the scaling variation ($K'_u = tK_u (t > 0)$), but the translation variance could not be considered. KTA is invariant to scaling problem, but as (Nguyen and Ho, 2008) points out, KTA is not invariant under data translation in the feature space. (He, Chang and Xie, 2008) normalizes the sub-kernel K_u by dividing K_u by the largest absolute value of K_u . From the equation 2 and 3 we can conclude that kernel translation ($K'_u = K_u + s (s > 0)$) does not change the decision function, which makes He’s normalization formulation suffer from kernel translation.

Kernel value can be treated as measuring the similarity between two features. In the ideal case, all the images

from the same categories should be similar, and those from different categories are dissimilar. Therefore, we define the ideal form of the sub-kernel K_u is:

$$K_u(x_i, x_j) = \begin{cases} a_u & \text{if } y_i = y_j \\ b_u & \text{if } y_i \neq y_j \end{cases} \quad (6)$$

where $a_u, b_u (a_u > b_u)$, and can be chosen using prior knowledge of the image features. For simplicity, a_u is the largest value of the sub-kernel, and b_u be the minimum value. To deal with the scaling problem, we constrain the kernel margin as 1:

$$\begin{aligned} \overline{K_u} &= K_u / (a_u - b_u) \\ \overline{a_u} &= a_u / (a_u - b_u) \\ \overline{b_u} &= b_u / (a_u - b_u) \end{aligned} \quad (7)$$

where $\overline{K_u}$ is the normalized kernel. This normalized kernel is invariant from kernel transformations.

3.3 Learning the kernel combination

After the sub-kernels are normalized, our aim is to learn the weights of normalized sub-kernels:

$$K_{opt}(x_i, x_j) = \sum_{u=1}^N d_u \overline{K_u}(x_i, x_j) \quad (8)$$

We measure how well the sub-kernel $\overline{K_u}$ is aligned to its ideal form using alignment loss. The alignment loss of $\overline{K_u}(x_i, x_j)$ is defined as:

$$\xi_u(x_i, x_j) = \begin{cases} \overline{a_u} - \overline{K_u}(x_i, x_j) & \text{if } y_i = y_j \\ \overline{K_u}(x_i, x_j) - \overline{b_u} & \text{otherwise} \end{cases} \quad (9)$$

The alignment loss defines how well the kernel is closed to its ideal form. Therefore, The total alignment loss of image x_i and x_j is defined as:

$$\xi(x_i, x_j) = \sum_{u=1}^N d_u * \xi_u(x_i, x_j) \quad (10)$$

If the alignment loss is small, it implicates that the kernel K is close to its ideal form. In order to make kernel K closer to its ideal form, we minimize the alignment loss:

$$\sum_{i,j=1}^l t_{ij} * \xi^n(x_i, x_j) + \lambda * \sum_{u=1}^N d_u^2 \quad (11)$$

where t_{ij} is the weight of each image pairs, and λ is a tradeoff parameter to prevent parameter overfitting. t_{ij} is designed to tackle the imbalanced dataset problem. If t_{ij} is a constant, the objective function treats every loss equal contribution, thus overfitted to the class with larger training samples. Therefore, the contribution of the loss of larger class must be penalized. In this paper, t_{ij} is set as $1/(n_i * n_j)$, where n_i is the number of images whose labels equal to label y_i . n is an integer, and we set $n=1$ or $n=2$ in this paper.

Denote d as the column vector of the weights $[d_1, d_2, \dots, d_N]$, and ξ_{ij} as the column vector of alignment loss of each sub-kernels $\{\xi_1(x_i, x_j), \xi_2(x_i, x_j), \dots, \xi_N(x_i, x_j)\}$, when $n=1$, the optimization problem can be rewritten as formula (12), which is name as LAKIF-L1:

$$\begin{aligned} \min \quad & \lambda d^T d + \left(\sum_{i,j=1}^l t_{ij} \xi_{ij}^T \right) d \\ \text{s.t.} \quad & d_j \geq 0 \quad (j = 1, 2, \dots, N) \\ & \sum_{j=1}^N d_j = 1 \end{aligned} \quad (12)$$

When $n=2$, the optimization problem can be written as formula (13), which is named as LAKIF-L2:

$$\begin{aligned}
\min \quad & d^T \left(\sum_{i,j=1}^l t_{ij} \xi_{ij} \xi_{ij}^T \right) d + \lambda d^T d \\
\text{s.t.} \quad & d_j \geq 0 \quad (j = 1, 2, \dots, N) \\
& \sum_{j=1}^N d_j = 1
\end{aligned} \tag{13}$$

Both the optimization problem is quadratic programming problem with N unknowns, which can be solved efficiently using convex programming softwares.

We will show the time complexity of training LAKIF-L1 and LAKIF-L2. For training LAKIF-L1, the complexity for constructing the matrix is $O(l^2N)$, and solving the quadratic programming problem is $O(N^3)$. The complexity of training SVM is $O(l * l_{sv} + l_{sv}^3)$, thus the total training time is $O(l^2N) + O(N^3) + O(l * l_{sv} + l_{sv}^3)$. For training LAKIF-L2, The complexity for constructing the matrix is $O(l^2N^2)$, and the others is the same as LAKIF-L1. Thus the total complexity of training time is $O(l^2N^2) + O(N^3) + O(l * l_{sv} + l_{sv}^3)$. Because most of the popular training algorithms use decomposition method, and update the parameters iteratively, the training time of these methods is affected by the parameter C , stopping criterion, and working set selection. In practice, our method is faster than the svm training.

4. Experiments

In this section we carry out experiments on Graz-01 (Opelt et al., 2006) and Caltech-101 (Li et al., 2006) datasets. We compare our kernel weight learning algorithm with two other weight learning methods: KTA (Cristianini et al., 2002) and second order optimization of Multiple Kernel Learning (MKL) (Chapelle and Rakotomamonjy, 2008) algorithm. The KTA method is a filter based method, and achieved good results on several datasets. The objective function of the MKL is similar to other Multiple Kernel Learning algorithms, but the MKL considers the kernel normalization and use second order information for optimize the function, resulting in one of the fastest solver in the literature.

Before the comparison of the algorithms, we first describe the implementation details of the preprocessing, feature extraction and the sub-kernels used in the paper. In the preprocessing process, images whose width or heights are larger than 300 pixels are scaled to 300 pixels with preserving the aspect ratio of the image. 6 image features and the corresponding kernel matrix are used in the paper, and described below in detail.

1) Color Histogram (CH): Color histogram (Swain and Ballard, 1991) is used to count the distribution of the image pixels, and have been widely used in content based image retrieval systems. Although its simplicity, it is scale and rotation invariant. We uniformly quantize the RGB color space into $4 \times 4 \times 4$ bins, resulting in a 64 bin color histogram.

2) PHOG180: Pyramid Histogram of Orientated Gradients (PHOG) (Bosch et al., 2007) is a global shape representation of an image. The method extracts the edge contours by the Canny edge detector, and computes the histogram of the gradient orientation of the image. The orientation is in the range $[0, 180]$, according to the author's experiment, we use 80 orientation bins.

3) PHOG360: The main difference of PHOG360 and PHOG180 is that the orientation of PHOG360 ranges in $[0, 360]$.

4) DenseSIFT: The SIFT (Lowe, 2004) descriptors are computed on 16×16 pixel patches over a grid with spacing of 8 pixels. The descriptors are then clustered using K-means algorithm to get a 200 bin codebook, and the image is represented as the histograms of codebook. The feature captures the global representation of the image.

5) DoGSIFT: SIFT descriptors are computed on the regions detected by DoG operator, and then clustered into 200 codebooks.

The above 5 features are matched by Spatial Pyramid Match Kernel.

6) Geometric Blur (GB): The geometric blur descriptors are extracted in the image, and we use Varma and Ray (2007)'s method to compute GB kernel.

4.1 Graz-01 dataset

This dataset contains 373 images of the category "bike", 460 images of the category "person", and 273 "background" images. Figure 1 shows some example images of each category. The images are highly complex

with high intra-class variability in scale, viewpoint, color, location and illumination. There is much background clutter in the image. These characteristics make the image classification algorithms difficult for recognizing the categories. The categorization task is Class vs Others. We randomly draw 100 images from the category (bike or person) as positive training images and 100 images as negative images (of which 50 images are sampled from the counter category and 50 other images are sampled from the “background” category). Similar strategy is adopted to obtain test set. In order to test our method for dealing with imbalanced database, we change the ratio of positive images to negative images for training the classifier. We experiment the ratio from 1:5 to 1:1, that is, we choose the 20, 25, 34, 50 and 100 positive images and 100 negative images for training. We report the average accuracy of the classification results showed in Figure 2.

As Figure 2 shows, in most cases our method outperforms all the sub-classifiers and KTA method, and is comparable to MKL method. One exception is using 20 positive “person” training images—all the combination methods fail to improve the accuracy. This is perhaps because the “person” class share some of the sub-classifiers trained by 20 “person” images are overfitting.

Compared to the “bike” category, classifying the “person” category is more challenging because “person” category shares more intra-class variability: the person always wear different color clothes, is more flexible than the bike category, in different poses, in more sophisticated environments. This results in the difficulty of recognizing person category. As Figure 2 shows, the classification accuracy of person category is lower than that of bike category.

4.2 Caltech-101 dataset

The Caltech-101 dataset consists of 101 object categories and an additional class of background images. Although the database has less intra-class variability, it is still a challenging database because of high inter-class variability and the large number of categories. Figure 3 shows some example images of Caltech-101 dataset.

We randomly select 30 images of each category, with 15 images for training and the other 15 images for testing. One-vs-All Strategies is adopted for multiple classification. In order to test our method not suffering from imbalance problem, one versus all strategy is adopted for the multi-class classification. The mean accuracy of all methods is reported in Table 1.

As Table 1 shows, the performance of GB descriptor is much better than that of other descriptors, and when combining these descriptors together, the GB descriptor will give overwhelming weight on the dataset. Therefore we report the performance of combination of both 6 kernels and 5 kernels (without GB). Our method does not suffer from imbalanced database, reaching comparable results to that of MKL method.

4.3 Average training time comparison

In this subsection we report the average training time of these methods on both datasets. We run the experiments on the same computer (Intel Quadcore 2.5 GHz CPU, 4 GB memory, Ubuntu Linux Server 9.10). The average training time is showed in Table 2. For the Graz-01 dataset, we report the average training time of 200 training images. For the Caltech-101 dataset, because we adopt one-vs-all training strategy, so every training stage contains 102 separate SVM training of $102 \times 15 = 1530$ samples. Because LAKIF and KTA methods are filter methods, we report the training time of kernel weight learning and SVM training separately. From Table 2 we can see that when the dataset is small, the training time of our method is about one iteration of SVM training, when the dataset grows large, the training time of our method is less than that of one iteration of SVM training, which makes our method more efficient than MKL method that trains SVM iteratively. MKL method is one of the fastest solver among “wrapper” methods, typically converges within 10 iterations.

5. Conclusion

We develop a novel framework Learning to Align the Kernel to its Ideal Form (LAKIF). LAKIF use a novel kernel normalization technique and learns the parameter by align the kernels to their ideal forms. Our framework considers the imbalanced dataset problem, and use different penalties on the alignment loss of different image pairs. Experimental results show that our method is efficient and robust for object categorization. The performance is comparable to MKL algorithm, while needs much less training time.

References

- Bach, F. R., Lanckriet, G. R. G., & Jordan, M. I. (2004). Multiple kernel learning, conic duality, and the SMO algorithm. In *Proceedings of the twenty-first international conference on Machine learning* (p. 6). Banff, Alberta, Canada: ACM. <http://dx.doi.org/10.1145/1015330.1015424>.
- Bosch, A., Zisserman, A., & Munoz, X. (2007). Representing shape with a spatial pyramid kernel. In

- Proceedings of the 6th ACM international conference on Image and video retrieval (pp. 401-408). <http://dx.doi.org/10.1145/1282280.1282340>.
- Chapelle, O., Haffner, P., & Vapnik, V. (1999). Support vector machines for histogram-based image classification. *Neural Networks, IEEE Transactions on*, 10(5), 1055-1064. <http://dx.doi.org/10.1109/72.788646>.
- Chapelle, O., & Rakotomamonjy, A. (2008). Second order optimization of kernel parameters. In *NIPS Workshop on Automatic Selection of Optimal Kernels*. Presented at the NIPS Workshop on Automatic Selection of Optimal Kernels.
- Cortes, C., & Vapnik, V. (1995). *Support-Vector Networks. Machine Learning*, 20(3), 273-297. <http://dx.doi.org/10.1023/A:1022627411411>
- Cristianini, N., Kandola, J., Elisseeff, A., & Shawe-Taylor, J. (2002). On kernel-target alignment. In *Advances in Neural Information Processing Systems*. 14 (Vol. 14, pp. 373, 367).
- Gehler, P., & Nowozin, S. (nd). On feature combination for multiclass object classification. In *ICCV 2007*. <http://dx.doi.org/10.1109/ICCV.2009.5459169>.
- Grauman, K., & Darrell, T. (2005). The pyramid match kernel: discriminative classification with sets of image features. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on* (Vol. 2, pp. 1458-1465). <http://dx.doi.org/10.1109/ICCV.2005.239>.
- He Junfeng, Chang Shih-Fu, & Xie Lexing. (2008). Fast kernel learning for spatial pyramid matching. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on* (pp. 1-7). <http://dx.doi.org/10.1109/CVPR.2008.4587636>.
- Hoi, S. C. H., Lyu, M. R., & Chang, E. Y. (2006). Learning the unified kernel machines for classification. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 187-196). <http://dx.doi.org/10.1145/1150402.1150426>.
- Joachims, T. (1999). Making large-scale support vector machine learning practical. In *Advances in kernel methods: support vector learning* (pp. 169-184).
- Kumar, A., & Sminchisescu, C. (2007). Support Kernel Machines for Object Recognition. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on* (pp. 1-8). <http://dx.doi.org/10.1109/ICCV.2007.4409065>.
- Lazebnik, S., Schmid, C., & Ponce, J. (2006). Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on* (Vol. 2, pp. 2169-2178). <http://dx.doi.org/10.1109/CVPR.2006.68>.
- Li Fei-Fei, Fergus, R., & Perona, P. (2006). One-shot learning of object categories. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(4), 594-611. <http://dx.doi.org/10.1109/TPAMI.2006.79>.
- Lowe, D. G. (2004). Distinctive Image Features from Scale-Invariant Keypoints. *Int. J. Comput. Vision*, 60(2), 91-110. <http://dx.doi.org/10.1023/B:VISI.0000029664.99615.94>.
- Nguyen, C. H., & Ho, T. B. (2008). *An efficient kernel matrix evaluation measure. Pattern Recognition*, 41(11), 3366-3372. <http://dx.doi.org/10.1016/j.patcog.2008.04.005>.
- Opelt, A., Pinz, A., Fussenegger, M., & Auer, P. (2006). Generic object recognition with boosting. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(3), 416-431. <http://dx.doi.org/10.1109/TPAMI.2006.54>.
- Platt, J. (1999). Fast training of support vector machines using sequential minimal optimization. *Advances in kernel methods: support vector learning*, 185-208.
- Rakotomamonjy, A., Bach, F., Canu, S., & Grandvalet, Y. (2008). SimpleMKL. *Journal of Machine Learning Research*, 9, 2521, 2491.
- Sande, K., Gevers, T., & Snoek, C. (2010). Evaluating Color Descriptors for Object and Scene Recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(9), 1582-1596. <http://dx.doi.org/10.1109/TPAMI.2009.154>.
- Swain, M. J., & Ballard, D. H. (1991). Color indexing. *Int. J. Comput. Vision*, 7(1), 11-32. <http://dx.doi.org/10.1007/BF00130487>.
- Varma, M., & Ray, D. (2007). Learning The Discriminative Power-Invariance Trade-Off. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on* (pp. 1-8).

<http://dx.doi.org/10.1109/ICCV.2007.4408875>.

Yang Jingjing, Li Yuanning, Tian Yonghong, Duan Lingyu, & Gao Wen. (2009). Group-sensitive multiple kernel learning for object categorization. In 2009 IEEE 12th International Conference on Computer Vision (pp. 436-443). <http://dx.doi.org/10.1109/ICCV.2009.5459172>.

Yen-Yu Lin, Tyng-Luh Liu, & Chiou-Shann Fuh. (2007). Local Ensemble Kernel Learning for Object Category Recognition. In *Computer Vision and Pattern Recognition*, 2007. CVPR '07. IEEE Conference on (pp. 1-8). <http://dx.doi.org/10.1109/CVPR.2007.383084>.

Table 1. the categorization performance on Caltech-101 dataset

Feature	Accuracy	
1.CH	27.73%	
2.PH180	47.74%	
3.PH360	49.29%	
4.DenseSIFT	46.14%	
5.DoGSIFT	38.23%	
6.GB	62.42%	
LAKIF-L2 LAKIF-L1 KTA MKL	Without GB	with GB
	61.02%	68.10%
	60.72%	66.34%
	55.43%	65.62%
	61.83%	69.48%

Table 2. The average training time of various methods(seconds)

	Graz-01(200*1)	Caltech-101(1530*102)
LAKIF-L1	0.08+0.08=0.16	39.7+253.1=292.8
LAKIF-L2	0.14+0.08=0.16	106.8+246.3=353.1
KTA	0.07+0.08=0.15	38.2+250.7=288.9
MKL	0.35	1436.9



Figure 1. Some example images of Graz-01 dataset:

Left: "bike" Category; Middle: "person" Category; Right: "Background" Category

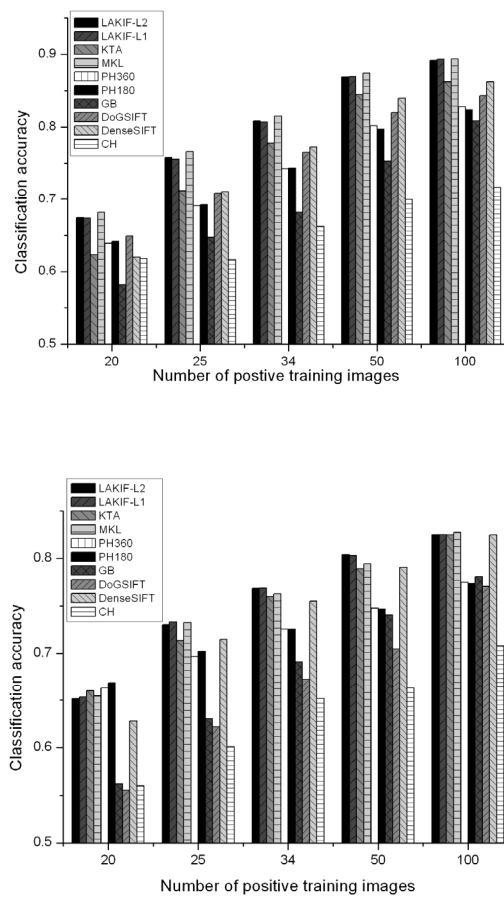


Figure 2. The categorization performance on Graz-01 top: "bike" category; bottom: "person" category

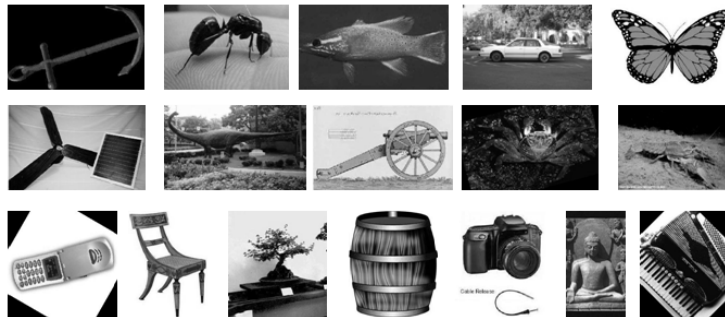


Figure 3. Example images of Caltech-101 dataset