Learning The Discriminative Power-Invariance Trade-Off

Manik Varma
Microsoft Research India
manik@microsoft.com

Debajyoti Ray
Gatsby Computational Neuroscience Unit
University College London
debray@gatsby.ucl.ac.uk

Abstract

We investigate the problem of learning optimal descriptors for a given classification task. Many hand-crafted descriptors have been proposed in the literature for measuring visual similarity. Looking past initial differences, what really distinguishes one descriptor from another is the trade-off that it achieves between discriminative power and invariance. Since this trade-off must vary from task to task, no single descriptor can be optimal in all situations.

Our focus, in this paper, is on learning the optimal trade-off for classification given a particular training set and prior constraints. The problem is posed in the kernel learning framework. We learn the optimal, domain-specific kernel as a combination of base kernels corresponding to base features which achieve different levels of trade-off (such as no invariance, rotation invariance, scale invariance, affine invariance, etc.*) This leads to a convex optimisation problem with a unique global optimum which can be solved for efficiently. The method is shown to achieve state-of-the-art performance on the UIUC textures, Oxford flowers and Caltech 101 datasets.*

1. Introduction

A fundamental problem in visual classification is designing good descriptors and many successful ones have been proposed in the literature [31]. If one looks past the initial dissimilarities, what really distinguishes one descriptor from another is the trade-off that it achieves between discriminative power and invariance. For instance, image patches, when compared using standard Euclidean distance, have almost no invariance but very high discriminative power. At the other extreme, a constant descriptor has complete invariance but no discriminative power. Most descriptors place themselves somewhere along this spectrum according to what they believe is the optimal trade-off.

However, the trade-off between invariance and discriminative power depends on the specific classification task at hand. It varies according to the training data available as well as prior knowledge and thus no single descriptor can be optimal for all tasks. For example, when classifying digits, one would not like to use a fully rotationally invariant descriptor as a 6 would then be mistaken for a 9. If the task was now simplified to distinguishing between just 4 and 9, then it would be preferable to have full rotational invariance if the digits could occur at any arbitrary orientation. However, 4s and 9s are easily confused. Therefore, if a rich enough training corpus was available with digits present at a large number of orientations, then one could revert back to a more discriminative and less invariant descriptor. In this scenario, the data itself would provide the rotation invariance and even nearest neighbour matching of rotationally variant descriptors would do well. As such, even if an optimal descriptor could be hand-crafted for a given task, it might no longer be optimal as the training set size is varied.

Our focus in this paper is on learning the trade-off between invariance and discriminative power for a given classification task. Knowledge of the trade-off can directly lead to improved classification. Perhaps as importantly, it might also provide insights into the nature of the problem being tackled. In addition, knowing how invariances change with varying training set size could be used to learn priors which could be transfered to other closely related problems. Finally, such knowledge can also be used to perform analogous reasoning where images are retrieved on the basis of learnt invariances rather than just image content.

It is often easy to arrive at the broad level of invariance or discriminative power necessary for a particular classification task by visual inspection. However, figuring out the exact trade-off can be more difficult. Let us go back to our example of classifying 4 versus 9. If only rotated copies of both digits were present in the training set then we could conclude that, broadly speaking, rotationally invariant descriptors would be suited to this task. However, what if some of the rotated digits were now scaled by a small factor, just enough to start causing confusion between the two digits? We might now consider moving up to similarity or affine invariant descriptors. However, this might lead to even poorer classification performance as such descriptors

would have lower discriminative power than purely rotationally invariant ones.

An ideal solution would be for every descriptor to have a continuously tunable meta parameter controlling its level of invariance. By varying the parameter, one could generate an infinite set of base descriptors spanning the complete range of the trade-off and, from this set, select the single base descriptor corresponding to the optimal trade-off level. The optimal descriptor's kernel matrix should have the same structure as the *ideal kernel* (essentially corresponding to zero intra-class and infinite inter-class distances) in kernel target alignment [15]. Unfortunately, most descriptors don't have such a continuously tunable parameter.

It is nevertheless possible to discretely sample the levels of invariance and generate a finite set of base descriptors. For instance, by selectively taking the maximum response over scale or orientation or other transformations of a basic filter one can generate base descriptors that are scale invariant, rotation invariant, *etc*. Alternatively, one can even start with different descriptors which achieve different levels of the trade-off. The optimal descriptor can still be approximated, not by selecting one of the base descriptors, but rather by taking their combination. However, approximating the ideal kernel via kernel target alignment is no longer appropriate as the method is not geared for classification.

Our solution instead is to combine a minimal set of base descriptors specifically for classification. The theory is developed in Section 3 but for an intuitive explanation let us return to our 4 versus 9 example. Starting with base descriptors that are rotationally invariant, scale invariant, affine invariant *etc*., our solution is to approximate the optimal descriptor by combining the rotationally invariant descriptor with just the scale invariant one. The combined descriptor would have neither invariance in full. As a result, the distance between a digit and its rotated copy would no longer be zero, but would still be tolerably small. Similarly, small scale changes would lead to increased, small non-zero distances within class. However, the combined distance between classes would also be increased and by a sufficient enough margin to ensure good classification.

2. Related Work

Our work builds on recent advances in kernel learning. It is also related to work on learning distance functions as well as descriptor optimisation and combination.

The goal of kernel learning is to learn a kernel which is optimal for the specified task. Much progress has been made recently in this field and solutions have been proposed based on kernel target alignment [15], multiple kernel learning [3, 24, 35, 42, 52], hyperkernels [34, 45], boosted kernels [14, 20] and other methods [2, 9]. These approaches mainly differ in the cost function that is optimised. Of particular interest are [3, 4, 35, 42] as each learns the optimal kernel for classification as a linear combination of base kernels with positive weights while enforcing sparsity.

The body of work on learning distances [13,17,30,32,37, 39,41] is also relevant to our problem. In addition, boosting has been particularly successful at learning distances and features optimised for classification and related tasks [44]. A recent survey of the state-of-the-art in learning distances can be found in [19].

There has also been a lot of work done on learning invariances in an unsupervised setting, see [22, 43, 49] and references within. In this scenario, an object is allowed to transform over time and a representation invariant to such transformations is learnt from the data. These methods are not directly applicable to our problem as they are unsupervised and generally focus on learning invariances without regard to discriminative power.

One might also try and learn an optimal descriptor directly [21, 27, 36, 48] for classification. However, our proposed solution has two advantages. First, by combining kernels, we never need to work in combined high dimensional descriptor space with all its associated problems. By effective regularisation, we are also able to avoid the over-fitting problem typical of such high dimensional spaces. Second, we are able to combine heterogeneous sources of data, such as shape, colour and texture.

The idea of combining descriptors has been explored in [8,25,33,51]. Unfortunately, these methods are not based on learning. In [25, 51] a fixed combination of descriptors is tried with all descriptors being equally weighted all the time. In [8, 33] a brute force search is performed over a validation set to determine the best descriptor weights.

Finally, the idea of a trade-off between invariance and discriminative power is well known and is explored theoretically in [40]. However, rather than learning the actual trade-off, their proposed randomised invariants solution is to add noise to the training set features. The noise parameters, corresponding to the trade-off, have to be hand tuned. In this paper, we automatically learn both the trade-off as well as the optimal kernel for classification.

3. Learning the Trade-Off

We start with N_k base descriptors and associated distance functions $f_1, \ldots, f_{N_k}$. Each descriptor achieves a different trade-off between discriminative power and invariance on the specified task. The descriptors and distance functions are then "kernelised" to yield base kernels matrices $\mathbf{K}_1, \ldots, \mathbf{K}_{N_k}$. There are many ways of converting distances to inner products and one is free to choose whichever embedding is most suitable. We simply set $\mathbf{K}_k(\mathbf{x}, \mathbf{y}) = \exp(-\gamma_k f_k(\mathbf{x}, \mathbf{y}))$ taking care to ensure that the kernel matrices are strictly positive definite.

Given the base kernels, the optimal descriptor's kernel is approximated as $\mathbf{K}_{\text{opt}} = \sum_k d_k \mathbf{K}_k$ where the weights $\mathbf{d}$

correspond to the trade-off level. The optimisation is carried out in an SVM framework so as to achieve the best classification on the training set, subject to regularisation. We set up the following primal cost function

$$\begin{aligned} \underset{\mathbf{w},\mathbf{d},\boldsymbol{\xi}}{\text{Min}} \quad & \tfrac{1}{2}\mathbf{w}^t\mathbf{w} + C\mathbf{1}^t\boldsymbol{\xi} + \boldsymbol{\sigma}^t\mathbf{d} && (1)\\ \text{subject to} \quad & y_i(\mathbf{w}^t\boldsymbol{\phi}(\mathbf{x}_i) + b) \geq 1 - \xi_i && (2)\\ & \boldsymbol{\xi} \geq 0, \mathbf{d} \geq 0, \mathbf{A}\mathbf{d} \geq \mathbf{p} && (3)\\ \text{where} \quad & \boldsymbol{\phi}^t(\mathbf{x}_i)\boldsymbol{\phi}(\mathbf{x}_j) = \textstyle\sum_k d_k \boldsymbol{\phi}_k^t(\mathbf{x}_i)\boldsymbol{\phi}_k(\mathbf{x}_j) && (4) \end{aligned}$$

The objective function (1) is near identical to the standard l_1 C-SVM objective. Given the misclassification penalty C, it maximises the margin while minimising the hinge loss on the training set $\{(\mathbf{x}_i, y_i)\}$. The only addition is an l_1 regularisation on the weights $\mathbf{d}$ since we would like to discover a minimal set of invariances. Thus, most of the weights will be set to zero depending on the parameters $\boldsymbol{\sigma}$ which encode our prior preferences for descriptors. The l_1 regularisation thus prevents overfitting if many base kernels are included since only a few will end up being used. Also, it can be shown that the quantity $\frac{1}{2}\mathbf{w}^t\mathbf{w}$ is minimised by increasing the weights and letting the support vectors tend to zero. The regularisation prevents this from happening and can therefore be seen as not letting the weights become too large. This could also be achieved by requiring that the weights sum to unity but we prefer not to do this as it restricts the search space.

The constraints are also similar to the standard SVM formulation. Two additional constraints have been incorporated. The first, $\mathbf{d} \geq 0$, ensures that the weights are interpretable and also leads to a much more efficient optimisation problem. The second, $\mathbf{A}\mathbf{d} \geq \mathbf{p}$, with some restrictions, lets us encode our prior knowledge about the problem. The final condition (4) is just a restatement of $\mathbf{K}_{opt} = \sum_k d_k \mathbf{K}_k$ using the non-linear embedding $\boldsymbol{\phi}$.

It is straightforward to derive the corresponding dual problem which turns out to be:

$$\begin{aligned} \underset{\boldsymbol{\alpha},\boldsymbol{\delta}}{\text{Max}} \quad & \mathbf{1}^t\boldsymbol{\alpha} + \mathbf{p}^t\boldsymbol{\delta} && (5)\\ \text{subject to} \quad & 0 \leq \boldsymbol{\delta},\ 0 \leq \boldsymbol{\alpha} \leq C,\ \mathbf{1}^t\mathbf{Y}\boldsymbol{\alpha} = 0 && (6)\\ & \tfrac{1}{2}\boldsymbol{\alpha}^t\mathbf{Y}\mathbf{K}_k\mathbf{Y}\boldsymbol{\alpha} \leq \sigma_k - \boldsymbol{\delta}^t\mathbf{A}_k && (7) \end{aligned}$$

where the non-zero $\boldsymbol{\alpha}$s correspond to the support vectors, $\mathbf{Y}$ is a diagonal matrix with the labels on the diagonal and $\mathbf{A}_k$ is the k^{th} column of $\mathbf{A}$.

The dual is convex with a unique global optimum. It is an instance of a Second Order Cone Program [10] and can be solved relatively efficiently by off-the-shelf numerical optimisation packages such as SeDuMi [1].

However, in order to tackle large scale problems involving hundreds of kernels we adopt the minimax optimisation strategy of [12, 35]. In their method, the primal is reformulated as $\text{Min}_{\mathbf{d}}\, T(\mathbf{d})$ subject to $\mathbf{d} \geq 0$ and $\mathbf{A}\mathbf{d} \geq \mathbf{p}$, where

$$\begin{aligned} T(\mathbf{d}) = \quad \text{Min}_{\mathbf{w},\boldsymbol{\xi}} \quad & \tfrac{1}{2}\mathbf{w}^t\mathbf{w} + C\mathbf{1}^t\boldsymbol{\xi} + \boldsymbol{\sigma}^t\mathbf{d} && (8)\\ \text{subject to} \quad & y_i(\mathbf{w}^t\boldsymbol{\phi}(\mathbf{x}_i) + b) \geq 1 - \xi_i && (9)\\ & \boldsymbol{\xi} \geq 0 && (10) \end{aligned}$$

The strategy is to minimise T using projected gradient descent via the iteration $\mathbf{d}^{n+1} = \mathbf{d}^n - \epsilon^n \boldsymbol{\nabla} T$ taking care to ensure that the constraints $\mathbf{d}^{n+1} \geq 0$ and $\mathbf{A}\mathbf{d}^{n+1} \geq \mathbf{p}$ are satisfied. The important step then is calculating $\boldsymbol{\nabla} T$. In order to do so, we look to the dual of T which is

$$\begin{aligned} W(\mathbf{d}) = \underset{\boldsymbol{\alpha}}{\text{Max}} \quad & \mathbf{1}^t\boldsymbol{\alpha} + \boldsymbol{\sigma}^t\mathbf{d} - \tfrac{1}{2}\textstyle\sum_k d_k \boldsymbol{\alpha}^t\mathbf{Y}\mathbf{K}_k\mathbf{Y}\boldsymbol{\alpha} && (11)\\ \text{subject to} \quad & 0 \leq \boldsymbol{\alpha} \leq C,\ \mathbf{1}^t\mathbf{Y}\boldsymbol{\alpha} = 0 && (12) \end{aligned}$$

By the principle of strong duality $T(\mathbf{d}) = W(\mathbf{d})$. Furthermore, if $\boldsymbol{\alpha}^*$ maximises W, then [7] have shown that W is differentiable if $\boldsymbol{\alpha}^*$ is unique (which it is in our case since all the kernel matrices are strictly positive definite). Finally, as proved in Lemma 2 of [12], W can be differentiated with respect to $\mathbf{d}$ as if $\boldsymbol{\alpha}^*$ did not depend on $\mathbf{d}$. We therefore get

$$\frac{\partial T}{\partial d_k} = \frac{\partial W}{\partial d_k} = \sigma_k - \tfrac{1}{2}\boldsymbol{\alpha}^{*t}\mathbf{Y}\mathbf{K}_k\mathbf{Y}\boldsymbol{\alpha}^* \qquad (13)$$

The minimax algorithm proceeds in two stages. In the first, $\mathbf{d}$ and therefore $\mathbf{K} = \sum d_k \mathbf{K}_k$ are fixed. Since $\boldsymbol{\sigma}^t\mathbf{d}$ is a constant, W is the standard SVM dual with kernel matrix $\mathbf{K}$. Any large scale SVM solver of choice can therefore be used to maximise W and obtain $\boldsymbol{\alpha}^*$. In the second stage, T is minimised by projected gradient descent according to (13). The two stages are repeated until convergence [11] or a maximum number of iterations is reached at which point the weights $\mathbf{d}$ and support vectors $\boldsymbol{\alpha}^*$ have been solved for.

A novel point $\mathbf{x}$ can now be classified as ± 1 by determining $\text{sign}(\sum_i \alpha_i y_i \mathbf{K}_{opt}(\mathbf{x}, \mathbf{x}_i) + b)$. To handle multi-class problems, both 1-vs-1 and 1-vs-All formulations are tried. For 1-vs-1, the task is divided into pairwise binary classification problems and a novel point is classified by taking the majority vote over classifiers. For 1-vs-All, one classifier is learnt per class and a novel point is classified according to its maximal distance from the separating hyperplanes.

4. Experimentation

In this section, we apply our method to the UIUC textures [25], Oxford flowers [33] and Caltech 101 object categorisation [16] databases. Since we would like to test how general the technique is, we assume that no prior knowledge is available and that no descriptor is *a priori* preferable to any other. We therefore set σ_k to be constant for all k and do not make use of the constraints $\mathbf{A}\mathbf{d} \geq \mathbf{p}$ (unless otherwise stated). The only parameters left to be set are C, the misclassification penalty, and the kernel parameters γ_k. These

parameters are not tweaked. Instead, C is set to 1000 for all classifiers and databases and γ_k is set as in [51].

To present comparative results, we tried the Multiple Kernel Learning SDP formulation of [24]. However, as [24] does not enforce sparsity and the results were 5% worse on the Caltech database we didn't explore the method further. Instead, we compare our method to the Multiple Kernel Learning Block l_1 regularisation method of [4] for which code is publicly available. All experimental results are calculated over 20 random train/test splits of the data except for 1-vs-All results which are calculated over 3 splits.

4.1. UIUC textures

The UIUC texture database [25] has 25 classes and 40 images per class. The database contains materials imaged under significant viewpoint variations and also contains fabrics which display folds and have non-rigid surface deformations. *A priori*, it is hard to tell what is the right level of invariance for this database. Affine invariance is probably helpful given the significant viewpoint changes. Higher levels of invariance might also be needed to characterise fabrics and handle non-affine deformations. However, [51] concluded that similarity invariance is better than either scale or affine invariance for this database. Then again, our results indicate that even better performance can be obtained by sticking to rotationally invariant descriptors. This reinforces the observation that it is not always straight forward to pinpoint the required level of invariance.

For this database, we start with a standard patch descriptor having no invariance but then take different transforms to derive 7 other base descriptors achieving different levels of the trade-off. The first descriptor is obtained by linearly projecting the patch onto the MR filters [47] (see Figure 1). Subsequent rotation, scale and similarity invariant descriptors are obtained by taking the maximum response of a basic filter over orientation, scale or both. This is similar to [38] where the maximum response is taken over position to achieve translation invariance. MR filter responses can also be used to derive fractal based bi-Lipschitz (including affine, perspective and non-rigid surface deformations) invariant and rotation invariant descriptors [46]. Finally, patches can directly yield rotation invariant descriptors by aligning them according to their dominant orientation.

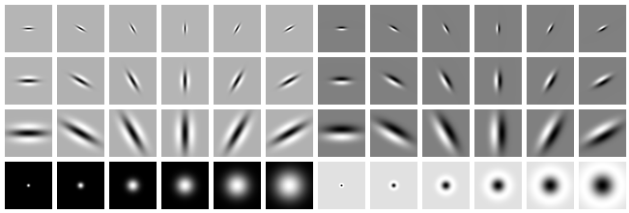
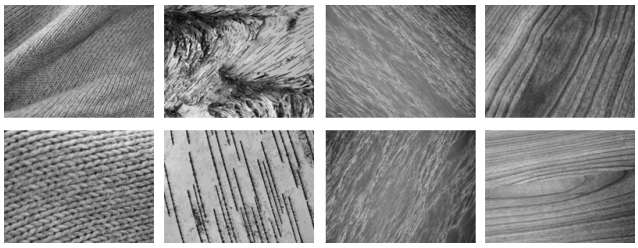


Figure 1. The extended MR8 filter bank.

Invariance	1NN	SVM (1-vs-1)
None (Patch)	$82.39 \pm 1.58\%$	$91.46 \pm 1.13\%$
None (MR)	$82.18 \pm 1.51\%$	$91.16 \pm 1.05\%$
Rotation (Patch)	$97.83 \pm 0.63\%$	$98.18 \pm 0.43\%$
Rotation (MR)	$93.00 \pm 1.04\%$	$96.69 \pm 0.74\%$
Rotation (Fractal)	$94.96 \pm 0.91\%$	$97.24 \pm 0.76\%$
Scale (MR)	$76.77 \pm 1.77\%$	$87.04 \pm 1.57\%$
Similarity (MR)	$90.35 \pm 1.15\%$	$95.12 \pm 0.95\%$
Bi-Lipschitz (Fractal)	$95.40 \pm 0.92\%$	$97.19 \pm 0.52\%$

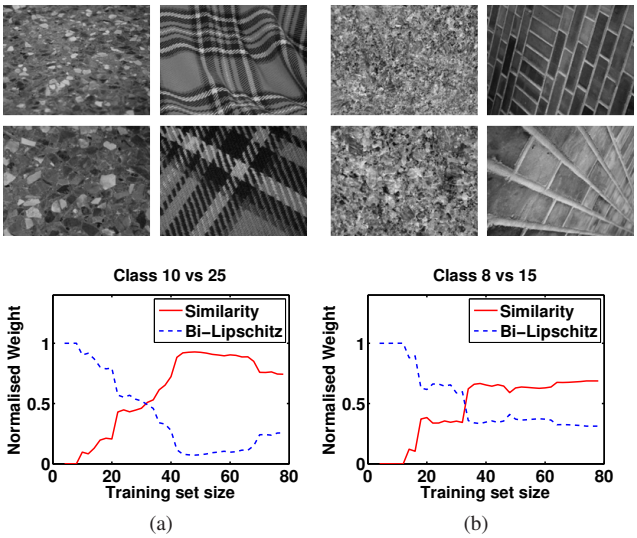
Table 1. Classification results on the UIUC texture dataset. The MKL-Block l_1 method of [4] achieves $96.94 \pm 0.91\%$ for 1-vs-1 classification when combining all the descriptors. Our results are **98.76 ± 0.64%** (1-vs-1) and **98.9 ± 0.68%** (1-vs-All).

For classification, the testing methodology is kept the same as in [51] – 20 images per class are used for training and the other 20 for testing. Table 1 lists the classification results. Our results are comparable to the $98.70 \pm 0.4\%$ achieved by the state-of-the-art [51]. What is interesting is that our performance has not decreased below that of any single descriptor despite the inclusion of specialised descriptors having scale and no invariance. These descriptors have poor performance in general. However, our method automatically sets their weights to zero most of the time and uses them only when they are beneficial for classification. Had the equally weighted combination scheme of [51] been used, these descriptors would have been brought into play all the time and the resulting accuracy drops down to $96.79 \pm 0.86\%$. In each of the 20 train/test splits,



Class 23 Class 3 Class 7 Class 4

Figure 2. 1-vs-1 weights learnt on the UIUC database: Both class 23 and class 3 exhibit significant variation. As a result, bi-Lipschitz invariance gets a very high weight when distinguishing between these two classes while all the other weights are 0. However class 7 is simpler and the main source of variability is rotation. Thus, full bi-Lipschitz invariance is no longer needed when distinguishing between class 23 and class 7. It can therefore be traded-off with a more discriminative descriptor. This is reflected in the learnt weights where rotation invariance gets a high weight of 1.46 while bi-Lipschitz invariance gets a small weight of 0.22. Bi-Lipschitz invariance isn't set to 0 as class 23 would start getting misclassified. However, if class 23 were replaced with the simpler class 4, which primarily has rotations, then bi-Lipschitz invariance is no longer necessary. Thus, when distinguishing class 7 from class 4, rotation invariance is the only feature used.



(a) (b)

Figure 3. Column (a) shows images from classes 10 and 25 and the variation in learnt weights as the training set size is increased for this pairwise classification task. A similar plot for classes 8 and 15 is shown in (b). When the training set size is small, a higher level of invariance (bi-Lipschitz) is needed. As the training set size grows, a less invariant and more discriminative descriptor (similarity) is preferred and automatically learnt by our method. The trends, though not identical, are similar in both (a) and (b) indicating that the tasks could be related. Inspecting the two class pairs indicates that while they are visually distinct, they do share the same types of variations (apart from the fabric crumpling).

learning the descriptors using our method outperformed equally weighted combinations (as well as the MKL-Block l_1 method). Figure 2 shows how the learnt weights correspond visually to the trade-offs between different classes while Figure 3 shows that the learnt weights change sensibly as the training set size is varied.

4.2. Oxford Flowers

The Oxford flowers database [33] contains 17 different categories of flowers and each class has 80 images. Classification is carried out on the basis of vocabularies of visual words of shape, colour and texture descriptors in [33]. The background in each image is removed using graph cuts so as to extract features from the flowers alone and not from the surrounding vegetation. Shape distances between two images are calculated as the χ^2 statistic between the normalised frequency histograms of densely sampled, vector quantised SIFT descriptors [29] of the two images. Similarly, colour distances are computed over vocabularies of HSV descriptors and texture over MR8 filter responses [47].

Cue combination fits well within our framework as one can think of an ideal colour descriptor as being very highly discriminating on the basis of an object's colour but invariant to changes in the object's shape or texture. Similar arguments hold for shape and texture descriptors. We therefore

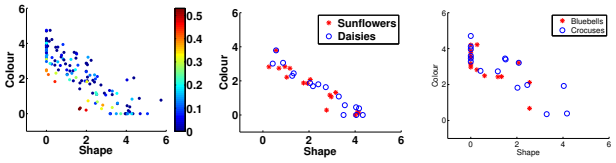
Descriptor	1NN	SVM (1-vs-1)
Shape	$53.30 \pm 2.69\%$	$68.88 \pm 2.04\%$
Colour	$47.32 \pm 2.59\%$	$59.71 \pm 1.95\%$
Texture	$39.36 \pm 2.43\%$	$59.00 \pm 2.14\%$

Table 2. Classification results on the Oxford flowers dataset. The MKL-Block l_1 method of [4] achieves $77.84 \pm 2.13\%$ for 1-vs-1 classification when combining all the descriptors. Our results are $\mathbf{80.49 \pm 1.97\%}$ (1-vs-1) and $\mathbf{82.55 \pm 0.34\%}$ (1-vs-All).

start with shape, colour and texture distances between every image pair, provided directly by the authors of [33]. Testing is also carried out according to the methodology of [33]. Thus, for each class, 40 images are used for training, 20 for validation and 20 for testing. We make no use of the validation set as all our parameters have already been set. Table 2 lists the classification results. Our results are better than the individual base kernels and are also better than the MKL-Block l_1 formulation on each of the 20 train/test splits.

Figure 4 (a) plots the distribution of the learnt shape and colour weights for all 136 pairwise classifiers in the 1-vs-1 formulation. Normalised texture weights are shown as colour codes to emphasise that they are relatively small. Note that the weights don't favour either just shape or just colour features. An entire set of weights is learnt, spanning the full range from shape to colour. While a person could correctly predict which is the more important feature by looking at the images, they would be hard pressed to achieve the precise trade-off.

The relative importance of the learnt 1-vs-1 weights is curious. Shape turns out to be the dominant feature in 38.24% of the pairwise classification tasks, colour in 60.29% and texture in 1.47%. This is surprising, since according to the individual SVM classification results in Table 2, shape is the best single feature and texture is nearly as good as colour. Texture features are probably ignored in our formulation as they are very strongly correlated with shape features (both are edge based). The l_1 regularisation prefers minimal feature sets with small weights and so gives texture either zero or low weights. Forcing the texture weights to be high (by constraining them to be higher than colour using the constraint term $\mathbf{Ad} \geq \mathbf{p}$) improves the overall



(a) (b) (c)

Figure 4. The distribution of the learnt shape and colour weights on the Oxford flowers dataset: (a) 1-vs-1 pairwise weights for all the classes; (b) 1-vs-1 weights for Sunflowers and Daisies; and (c) Bluebells and Crocuses.

Figure 5. 1-vs-1 weights learnt on the Oxford dataset: Dandelions and Wild Tulips are both yellow and therefore colour is a nuisance parameter to which we should be invariant. However, shape is a good discriminator for these flowers. This is reflected in the weights which are learnt to be shape=3.94, colour=0 and texture=0. When the task changes to distinguishing Dandelions from Crocuses, shape becomes a poor discriminator (Crocuses have large variability) but colour becomes good. However, Crocuses also have some yellow which causes confusion. To compensate for this, shape invariance is traded-off for increased discrimination and the learnt weights are shape=0.42, colour=2.46 and texture=0. When distinguishing Cowslips from Irises, all three features are used and the weights are shape=1.48, colour=2.00 and texture=1.36. As can be seen, colour is good at characterising Cowslips (which are always yellow) but not sufficient for distinguishing them from Irises which might also be yellow. Shape and texture features are also not sufficient by themselves due to the large intra-class variability. However, combining all three features in the right proportion leads to good discrimination.

1-vs-1 accuracy marginally to $81.12 \pm 2.09\%$.

A few classes along with their learnt pairwise weights are shown in Figure 5. Keeping one class fixed and varying the other results in the weights changing according to changes in perceptual cues. Since the learnt weights provide a layer of abstraction, one can use them to reason about the given classification problem. For instance, Figure 4 (b) and (c) plot the distribution of all 1-vs-1 weights for Bluebells and Crocuses and Sunflowers and Daisies respectively. The distributions of Bluebells and Crocuses are similar as are that of Sunflowers and Daisies but the two sets are distinct from each other. This shows that these categories form related

Figure 6. Learning related tasks: Bluebells and Crocuses share similar sets of invariances. Apart from some cases, they require higher degrees of shape invariance and can be distinguished well on the basis of their colour. Sunflowers and Daisies are neither distinctive in shape nor in colour from all the other classes. They form another related pair in that sense. Since the flowers in each related pair are visually different, it might be hard to establish such relationships by visual inspection alone.

classification tasks (see Figure 6). Such knowledge could be useful for learning and transferring priors.

Finally, since only three descriptors are used on this database, an exhaustive search can be performed on a validation set for the best combination of weights. However, it was noticed that performing a brute force search over every class pair lead to overfitting. If ties were not resolved properly, the overall classification performance could be as poor as 60%. We therfore enforced that, in the 1-vs-1 formulation, all pairwise classifiers should have the same weights and performed a brute force search again. The best weights resulted in an accuracy of $80.62 \pm 1.65\%$ which is similar to our results. A 1-vs-All brute force search couldn't be performed as it was computationally too expensive.

4.3. Caltech 101 Object Categorisation

The Caltech 101 database [16] contains images of 101 categories of objects as well as a background class. The database is very challenging as it contains classes with significant shape and appearance variations (Ant, Chair) as well as classes with roughly fixed shape but considerably varying appearance (Butterfly, Watch) or vice-versa (Leopard, Panda). We therefore combine 6 shape and appearance features for this dataset.

The first two shape descriptors correspond to equations (1) and (2) in [50] and are based on Geometric Blur [5]. Pairwise image distances for these were provided directly by the authors for the training and test images used in their paper. For the first descriptor, GB, the distance between two images is $f_{\text{GB}}(I_1, I_2) = D^A(I_1 \rightarrow I_2) + D^A(I_2 \rightarrow I_1)$ where $D^A(I_1 \rightarrow I_2) = (1/m) \sum_{i=1}^{m} \min_{j=1..n} \|F_i^1 - F_j^2\|$. F_i^1 and F_j^2 are Geometric Blur features in the two images. The texture term in (1) in [50] is not used. The second descriptor, GBDist, corresponding to (2) in [50]. It is very similar to GB except that D^A now incorporates an additional first-order geometric distortion term.

We also incorporate the four descriptors used in [8]. The two appearance features, AppGray and AppColour, are based on SIFT descriptors sampled on a regular grid. At each point on the grid, SIFT descriptors are computed us-

Descriptor	1NN	SVM (1-vs-1)
GB	$39.67 \pm 1.02\%$	$57.33 \pm 0.94\%$
GBDist	$45.23 \pm 0.96\%$	$59.30 \pm 1.00\%$
AppGray	$42.08 \pm 0.81\%$	$52.83 \pm 1.00\%$
AppColour	$32.79 \pm 0.92\%$	$40.84 \pm 0.78\%$
Shape180	$32.01 \pm 0.89\%$	$48.83 \pm 0.78\%$
Shape360	$31.17 \pm 0.98\%$	$50.63 \pm 0.88\%$

Table 3. Classification results on the Caltech 101 dataset. The MKL-Block l_1 method of [4] achieves $76.55 \pm 0.84\%$ for 1-vs-1 classification when combining all the descriptors. Our results are **78.43** $\pm$ **1.05%** (1-vs-1) and **87.82** $\pm$ **1.00%** (1-vs-All).

(a) (b)

Figure 7. In (a) the three classes, Pizza, Soccer ball and Watch, all consist of round objects and so the learnt weights do not use shape for any of these class pairs. In (b) both Butterfly and Electric guitar have significant within class appearance variation. However, their shape remains much the same within class and is distinct between the classes. As such, only shape is used to distinguish these two classes from each other.

ing 4 fixed scales. These are then vector quantised to form a vocabulary of visual words. Images are represented as a bag of words and similarity between two images is given by the spatial pyramid kernel [26]. While AppGray is computed from gray scale images, AppColour is computed from an HSV representation. The two shape features, Shape180 and Shape360, are represented as histograms of oriented gradients and matched using the spatial pyramid kernel. Gradients are computed using the Canny edge detector followed by Sobel filtering. They are then discretized into the orientation histogram bins with soft voting. The primary difference between the two descriptors is that Shape180 is discretized into bins in the range $[0, 180]$ and Shape360 into $[0, 360]$. Details can be found in [8]. Note that since the gradients are computed at both boundary and texture edges these descriptors represent both local shape and local texture.

To evaluate classification performance, we stick to the methodology adopted in [6, 50]. Thus, 15 images are randomly selected from all 102 class (i.e. including the background) for training and another random 15 for testing. Classification results using each of the base descriptors as well as their combination are given in Table 3 and Figure 7 gives a qualitative feel of the learnt weights.

To compare our results to the state-of-the-art, note that [50] combine shape and texture features to obtain $59.08 \pm 0.37\%$ and [18] combine colour features in addition to get $60.3 \pm 0.70\%$. Kernel target alignment is used by [28] to combine 8 kernels based on shape, colour texture and other cues. Their results are 59.80%. In [23], a performance of 57.83% is achieved by combining 12 kernels using the MKL-Block l_1 method. In [8], a brute force search is performed over a validation set to learn the best combination of their 4 kernels in a 1-vs-All formulation. When training and testing on 15 images for 101 categories (i.e. excluding background) they record an overall classification accuracy of $71.4 \pm 0.8\%$. Using the same 4 kernels but testing on all 102 categories we obtain $79.85 \pm 0.04\%$.

5. Conclusions

In this paper, we developed an approach for learning the discriminative power-invariance trade-off for classification. Starting with base descriptors which achieve different levels of the trade-off, our solution is to combine them optimally in a kernel learning framework. The learnt kernel yields superior classification results while the learnt weights correspond to the trade-off and can be used for meta level tasks such as transfer learning or reasoning about the problem.

Our framework has certain attractive properties. It appears to be general and capable of handling diverse classification problems. No hand tuning of parameters was required. In addition, it can be used to combine heterogeneous sources of data. This is particularly relevant in cases where human intuition about the right levels of invariance might fail – such as when combining audio, video and text. Another advantage is that the method can work with poor, or highly specialised, descriptors. This is again useful in cases when the right levels of invariance are not known *a priori* and we would like to start with many base descriptors. Also, it appears that we get similar (Oxford Flowers) or better (Caltech 101) results as compared to brute force search over a validation set. This is particularly encouraging since a brute force search can be computationally expensive. In addition, in the very small training set size limit, it might not be feasible to hold out training data to form a validation set and one also risks overfitting. Finally, our performance was generally better than that of the MKL-Block l_1 method while also enjoying the advantage of scaling up to large problems as long as efficient solvers for the corresponding single kernel problem are available.

Acknowledgements

We are very grateful to the following for providing kernel matrices and for many helpful discussions: P. Anandan, Anna Bosch, Rahul Garg, Varun Gulshan, Jitendra Malik, Maria-Elena Nilsback, Patrice Simard, Kentaro Toyama, Hao Zhang and Andrew Zisserman.

References

[1] http://sedumi.mcmaster.ca/.

[2] A. Argyriou, C. A. Micchelli, and M. Pontil. Learning convex combinations of continuously parameterized basic kernels. In *COLT*, 2005.

[3] F. R. Bach, G. R. G. Lanckriet, and M. I. Jordan. Multiple kernel learning, conic duality, and the SMO algorithm. In *NIPS*, 2004.

[4] F. R. Bach, R. Thibaux, and M. I. Jordan. Computing regularization paths for learning multiple kernels. In *NIPS*, 2004.

[5] A. Berg and J. Malik. Geometric blur for template matching. In *CVPR*, volume 1, pages 607–614, 2001.

[6] A. C. Berg, T. L. Berg, and J. Malik. Shape matching and object recognition using low distortion correspondence. In *CVPR*, volume 1, pages 26–33, San Diego, California, 2005.

[7] J. F. Bonnans and A. Shapiro. *Perturbation Analysis of Optimization Problems*. 2000.

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

[9] O. Bousquet and D. J. L. Herrmann. On the complexity of learning the kernel matrix. In *NIPS*, pages 399–406, 2002.

[10] S. Boyd and L. Vandenberghe. *Convex Optimization*. 2004.

[11] P. Calamai and J. More. Projected gradient methods for linearly constrained problems. *Mathematical Programming*, 39(1):93–116, 1987.

[12] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee. Choosing multiple parameters for Support Vector Machines. *Machine Learning*, 46:131–159, 2002.

[13] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *CVPR*, volume 1, pages 26–33, 2005.

[14] K. Crammer, J. Keshet, and Y. Singer. Kernel design using boosting. In *NIPS*, pages 537–544, 2002.

[15] N. Cristianini, J. Shawe-Taylor, A. Elisseeff, and J. Kandola. On kernel-target alignment. In *NIPS*, 2001.

[16] L. Fei-Fei, R. Fergus, and P. Perona. One-shot learning of object categories. *IEEE PAMI*, 28(4):594–611, 2006.

[17] A. W. Fitzgibbon and A. Zisserman. On affine invariant clustering and automatic cast listing in movies. In *Proc. ECCV*, volume 3, pages 304–320, Copenhagen, Denmark, 2002.

[18] A. Frome, Y. Singer, and J. Malik. Image retrieval and recognition using local distance functions. In *NIPS*, 2006.

[19] T. Hertz. *Learning Distance Functions: Algorithms and Applications*. PhD thesis, 2006.

[20] T. Hertz, A. Bar-Hillel, and D. Weinshall. Learning a kernel function for classification with small training samples. In *ICML*, pages 401–408, Pittsburgh, USA, 2006.

[21] A. K. Jain and K. Karu. Learning texture discrimination masks. *IEEE PAMI*, 18(2):195–205, 1996.

[22] A. Kannan, N. Jojic, and B. J. Frey. Fast transformation-invariant component analysis. *IJCV*, Submitted.

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

[24] G. R. G. Lanckriet, N. Cristianini, P. Bartlett, L. El Ghaoui, and M. I. Jordan. Learning the kernel matrix with semidefinite programming. *JMLR*, 5:27–72, 2004.

[25] S. Lazebnik, C. Schmid, and J. Ponce. A sparse texture representation using local affine regions. *IEEE PAMI*, 27(8):1265–1278, 2005.

[26] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, volume 2, pages 2169–2178, New York, New York, 2006.

[27] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[28] Y. Y. Lin, T. Y. Liu, and C. S. Fuh. Local ensemble kernel learning for object category recognition. In *CVPR*, 2007.

[29] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

[30] S. Mahamud and M. Hebert. The optimal distance measure for object detection. In *CVPR*, pages 248–255, 2003.

[31] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE PAMI*, 27(10):1615–1630, 2005.

[32] E. Miller, N. Matsakis, and P. Viola. Learning from one example through shared densities on transforms. In *CVPR*, pages 464–471, 2000.

[33] M.-E. Nilsback and A. Zisserman. A visual vocabulary for flower classification. In *CVPR*, volume 2, pages 1447–1454, New York, New York, 2006.

[34] C. S. Ong, A. J. Smola, and R. C. Williamson. Learning the kernel with hyperkernels. *JMLR*, 6:1043–1071, 2005.

[35] A. Rakotomamonjy, F. Bach, S. Canu, and Y. Grandvalet. More efficiency in multiple kernel learning. In *ICML*, 2007.

[36] T. Randen and J. H. Husoy. Optimal filter-bank design for multiple texture discrimination. In *ICIP*, volume 2, pages 215–218, Santa Barbara, California, 1997.

[37] L. Ren, G. Shakhnarovich, J. K. Hodgins, P. Hanspeter, and P. Viola. Learning silhouette features for control of human motion. *ACM Trans. Graph*, 24(4):1303–1331, 2005.

[38] T. Serre, L. Wolf, S. Bileschi, M. Riesenhuber, and T. Poggio. Robust object recognition with cortex-like mechanisms. *IEEE PAMI*, 2007. To appear.

[39] G. Shakhnarovich, P. Viola, and T. J. Darrell. Fast pose estimation with parameter-sensitive hashing. In *ICCV*, pages 750–757, 2003.

[40] X. Shi and R. Manduchi. Invariant operators, small samples, and the bias-variance dilemma. In *CVPR*, volume 2, pages 528–534, 2004.

[41] P. Simard, Y. LeCun, J. Denker, and B. Victorri. Transformation invariance in pattern recognition – tangent distance and tangent propagation. *International Journal of Imaging System and Technology*, 11(2):181–194, 2001.

[42] S. Sonnenburg, G. Raetsch, C. Schaefer, and B. Schoelkopf. Large scale multiple kernel learning. *JMLR*, 7:1531–1565, 2006.

[43] M. W. Spratling. Learning viewpoint invariant perceptual representations from cluttered images. *IEEE PAMI*, 27(5):753–761, 2005.

[44] K. Tieu and P. Viola. Boosting image retrieval. *IJCV*, 56(1-2):17–36, 2004.

[45] I. W. Tsang and J. T. Kwok. Efficient hyperkernel learning using second-order cone programming. *IEEE Trans. Neural Networks*, 17(1):48–58, 2006.

[46] M. Varma and R. Garg. Locally invariant fractal features for statistical texture classification. In *ICCV*, 2007.

[47] M. Varma and A. Zisserman. A statistical approach to texture classification from single images. *IJCV*, 2005.

[48] S. Winder and M. Brown. Learning local image descriptors. In *CVPR*, 2007.

[49] T. Wiskott, L. Sejnowski. Slow feature analysis: Unsupervised learning of invariances. *Neural Computation*, 14(4):715–770, 2002.

[50] H. Zhang, A. Berg, , M. Maire, and J. Malik. SVM-KNN: Discriminative nearest neighbor classication for visual category recognition. In *CVPR*, pages 2126–2136, 2006.

[51] J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid. Local features and kernels for classification of texture and object categories: A comprehensive study. *IJCV*, 2007.

[52] A. Zien and C. S. Ong. Multiclass multiple kernel learning. In *ICML*, pages 1191–1198, 2007.