Generalized RBF feature maps for Efficient Detection

Sreekanth Vempati[1]
v_sreekanth@research.iiit.ac.in
Andrea Vedaldi[2]
vedaldi@robots.ox.ac.uk
Andrew Zisserman[2]
az@robots.ox.ac.uk
C. V. Jawahar[1]
jawahar@iiit.ac.in

[1] Center for Visual Information Technology
IIIT Hyderabad
Hyderabad, India
http://cvit.iiit.ac.in

[2] Visual Geometry Group
University of Oxford
Oxford, UK
http://www.robots.ox.ac.uk/∼vgg

Abstract

Kernel methods yield state-of-the-art performance in certain applications such as image classification and object detection. However, large scale problems require machine learning techniques of at most linear complexity and these are usually limited to linear kernels. This unfortunately rules out gold-standard kernels such as the generalized RBF kernels (e.g. exponential-χ^2). Recently, Maji and Berg [13] and Vedaldi and Zisserman [20] proposed explicit feature maps to approximate the additive kernels (intersection, χ^2, etc.) by linear ones, thus enabling the use of fast machine learning technique in a non-linear context. An analogous technique was proposed by Rahimi and Recht [14] for the translation invariant RBF kernels. In this paper, we complete the construction and combine the two techniques to obtain explicit feature maps for the generalized RBF kernels. Furthermore, we investigate a learning method using l^1 regularization to encourage sparsity in the final vector representation, and thus reduce its dimension. We evaluate this technique on the VOC 2007 detection challenge, showing when it can improve on fast additive kernels, and the trade-offs in complexity and accuracy.

1 Introduction

In computer vision applications such as object category classification and detection, the gold-standard kernels are the so called *generalized radial-basis function (RBF) kernels* [19, 23]. A typical example is the exponential-χ^2 kernel

$$K(\mathbf{x},\mathbf{y}) = e^{-\frac{1}{2\sigma^2}\chi^2(\mathbf{x},\mathbf{y})}, \quad \chi^2(\mathbf{x},\mathbf{y}) = \frac{1}{2}\sum_{l=1}^{d}\frac{(x_l - y_l)^2}{x_l + y_l}.$$

These kernels combine the benefits of two other important classes of kernels: the homogeneous additive kernels (e.g. the χ^2 kernel) and the RBF kernels (e.g. the exponential kernel). The homogeneous additive kernels are designed to compare histograms, and are particularly useful in computer vision since most descriptors are, in fact, histograms (e.g. SIFT [12],

GIST [17], HOG [7], bag-of-words [6], spatial pyramids [11]). The RBF kernels, on the other hand, can represent local templates which is useful for highly variable visual aspects.

A major problem in the use of the generalized RBF kernels is their computational cost. Training a non-linear SVM with such kernels is typically $O(dN^2)$ or $O(dN^3)$, where N is the number of training examples and d the data dimensionality. Testing the learned SVM is also very expensive, usually $O(dN)$, from the need to compare each novel datum to the support vectors determined during training (and these are usually of order N). In contrast, there exist methods that can train a linear SVM in time $O(dN)$ only (e.g. SVM-perf [10], PEGASOS [16], LIBLINEAR [9]) and testing is only of order $O(d)$ for a linear kernel (since it only involves a scalar product between the learnt weight vector $\mathbf{w}$ and the feature vector of the test image $\mathbf{x}$).

The fact that kernels can be seen as inner product in an appropriate vector space suggests that it may be possible to train and test efficiently SVMs even in the non-linear case. In symbols, for every positive-definite (PD) kernel $K(\mathbf{x},\mathbf{y})$ there exists a *feature map* $\Psi(\mathbf{x})$ such that

$$K(\mathbf{x},\mathbf{y}) = \langle \Psi(\mathbf{x}), \Psi(\mathbf{y}) \rangle \tag{1}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product in feature space. Typically $\Psi(\mathbf{x})$ is infinite dimensional and therefore not suitable for numerical computations. It may however be possible to find an *approximate* feature map $\hat{\Psi}(\mathbf{x})$ that (i) is finite dimensional and that (ii) generates a kernel $\langle \Psi(\mathbf{x}), \Psi(\mathbf{y}) \rangle$ that is a close approximation of (1), i.e.

$$\hat{K}(\mathbf{x},\mathbf{y}) = \langle \hat{\Psi}(\mathbf{x}), \hat{\Psi}(\mathbf{y}) \rangle, \quad \hat{K}(\mathbf{x},\mathbf{y}) \approx K(\mathbf{x},\mathbf{y}). \tag{2}$$

So far approximate feature maps have been proposed for the homogeneous additive kernels [13, 20] and for the RBF kernels [14]. In this paper we complete the story and give *efficient approximated feature maps for the generalized RBF kernels.* Specifically, Sect. 2.1 and Sect. 2.2 review the homogeneous additive and RBF kernels and their feature maps. Sect. 2.3 then reviews the generalized RBF kernels and derives feature maps for them, summarizing results on the approximation error and the computational cost, and Sect. 3 describes learning methods using l^1 regularization to encourage sparsity and improve testing speed. Finally, Sect. 4 compares empirically the various kernels and their approximations for the case of object detectors on the VOC 2007 dataset [8]. We show that the approximate feature map can improve performance significantly over that of the original additive kernels. An alternative to the method we propose is to employ Nyström-type approximations [1, 3, 22] to obtain linear feature maps, but these require a data dependent training step that we avoid here.

2 Kernels and feature maps

This section reviews the additive homogeneous kernels (Sect. 2.1), the RBF kernels (Sect. 2.2), and their feature maps [14, 20]. It then introduces the generalized RBF kernels and gives a finite dimensional approximate feature map for them (Sect. 2.3).

2.1 Additive homogeneous kernels

Common additive homogeneous kernels [20], such as the χ^2, intersection, Jensen-Shannon (JS), Hellinger's, are designed to compare probability distributions. An additive kernel is

defined as

$$K(\mathbf{x},\mathbf{y}) = \sum_{l=1}^{d} k(\mathbf{x}_l, \mathbf{y}_l) \tag{3}$$

where d is the dimension of the input histograms $\mathbf{x},\mathbf{y}$, l is the component (bin) index, and $k(x,y)$ is an homogeneous PD kernel on the non-negative reals $\mathbb{R}_0^+$ (the kernel k is *homogeneous* if $k(cx,cy) = ck(x,y)$ for any $c > 0$). For instance, setting $k(x,y) = \min(x,y)$ in (3) yields the intersection kernel, and setting $k(x,y) = 2xy/(x+y)$ yields the χ^2 kernel.

[20] proposes closed-form approximated feature maps for all common homogeneous kernels. The construction starts by factorizing $k(x,y)$ as

$$k(x,y) = \sqrt{xy}\mathscr{K}(\lambda), \quad \lambda = \log\frac{y}{x}. \tag{4}$$

where $\mathscr{K}(\lambda)$ is called the kernel *signature*, and is a scalar function that fully describes the kernel k. It is then noted that $\mathscr{K}(\lambda)$ is the Fourier transform of a symmetric non-negative measure $\kappa(\omega)\,d\omega$

$$\mathscr{K}(\lambda) = \int_{-\infty}^{\infty} e^{-i\omega\lambda}\,\kappa(\omega)\,d\omega. \tag{5}$$

This in turn can be used to define a feature map $k(x,y) = \langle \Psi(x), \Psi(y) \rangle$ by

$$\begin{aligned} k(x,y) &= \sqrt{xy}\mathscr{K}\left(\log\frac{y}{x}\right) = \int_{-\infty}^{\infty} \left(\sqrt{x\kappa(\omega)}e^{-i\omega\log x}\right)^* \left(\sqrt{y\kappa(\omega)}e^{-i\omega\log y}\right) d\omega \\ &= \int_{-\infty}^{\infty} [\Psi(x)]_\omega^* [\Psi(y)]_\omega\,d\omega, \quad \boxed{[\Psi(x)]_\omega = \sqrt{x\kappa(\omega)}e^{-i\omega\log x}} \end{aligned} \tag{6}$$

An approximate finite feature map $\hat{\Psi}(\mathbf{x})$ can be constructed by sampling and truncating (6):

$$[\hat{\Psi}(x)]_j = \sqrt{L}[\Psi(x)]_{jL}, \quad j = -n,\ldots,n. \tag{7}$$

Due to symmetries of the feature map $\Psi(x)$, (7) reduces to a real vector of dimension $2n+1$ [20]. The explicit form of the feature map in the case of χ^2 is given in Figure 1.

Since for the intersection, χ^2, Jensen-Shannon, and Hellinger's kernels the density $\kappa(\omega)$ can be computed in closed form, the approximated feature map $\hat{\Psi}(x)$ can also be computed from (7) in closed form. A corresponding approximate feature map for an additive homogeneous kernel $K(\mathbf{x},\mathbf{y}) \approx \langle \hat{\Psi}(\mathbf{x}), \hat{\Psi}(\mathbf{y}) \rangle$, is obtained by stacking $\hat{\Psi}(\mathbf{x}_l)$ for each of the d components of $\mathbf{x}$, i.e. $\hat{\Psi}(\mathbf{x}) = \mathrm{stack}_{l=1,\ldots,d} \hat{\Psi}(\mathbf{x}_l)$.

2.2 RBF kernels

The *radial-basis function (RBF) kernels* such as the exponential (Gaussian) kernel are function of the Euclidean distance between vectors $\mathbf{x},\mathbf{y}$:

$$K_{\mathrm{RB}}(\mathbf{x},\mathbf{y}) = k(\|\mathbf{x}-\mathbf{y}\|_2^2)$$

where $k : \mathbb{R}_0^+ \to \mathbb{R}$ is the *kernel profile* [5]. An important example is the exponential kernel

$$K_{\mathrm{RB}}(\mathbf{x},\mathbf{y}) = \exp\left(-\frac{1}{2\sigma^2}\|\mathbf{x}-\mathbf{y}\|_2^2\right).$$

Such kernels are a particular cases of the *translation invariant kernels* that can be written as

$$K_{\text{RB}}(\mathbf{x},\mathbf{y}) = \mathscr{K}_{\text{RB}}(\boldsymbol{\lambda}), \quad \boldsymbol{\lambda} = \mathbf{y}-\mathbf{x}.$$

We may call $\mathscr{K}_{\text{RB}}$ the kernel signature in analogy with the homogeneous case, but here the signature has d-dimensional input $\boldsymbol{\lambda}$. Moreover $\boldsymbol{\lambda} = \mathbf{y}-\mathbf{x}$ has a linear rather than logarithmic dependency on $\mathbf{x}$ and $\mathbf{y}$. As noticed in [14] and analogously to (6), the signature is the Fourier transform of a non-negative symmetric density $\kappa_{\text{RB}}(\boldsymbol{\omega})$ (Bochner theorem) and $\kappa_{\text{RB}}(\boldsymbol{\omega})$ can be used to define a feature map $\Psi_{\text{RB}}(\mathbf{x})$ for the RBF kernel:

$$\mathscr{K}_{\text{RB}}(\boldsymbol{\lambda}) = \int_{\mathbb{R}^d} e^{-i\boldsymbol{\omega}^\top\boldsymbol{\lambda}}\,\kappa_{\text{RB}}(\boldsymbol{\omega})\,d\boldsymbol{\omega}, \qquad \boxed{[\Psi_{\text{RB}}(\mathbf{x})]_{\boldsymbol{\omega}} = \sqrt{\kappa_{\text{RB}}(\boldsymbol{\omega})}e^{-i\boldsymbol{\omega}^\top\mathbf{x}}.} \tag{8}$$

Differently from (7), it is not practical to regularly sample $[\Psi_{\text{RB}}(\mathbf{x})]_{\boldsymbol{\omega}}$ to obtain an approximate feature map due to the high dimensionality of $\boldsymbol{\omega}\in\mathbb{R}^d$. Instead, [14] proposes to use random sampling. Without loss of generality, assume that the kernel is properly normalized, i.e. that $K_{\text{RB}}(\mathbf{x},\mathbf{x}) = 1$. Then $1 = K_{\text{RB}}(\mathbf{x},\mathbf{x}) = \mathscr{K}_{\text{RB}}(0) = \int \kappa_{\text{RB}}(\boldsymbol{\omega})\,d\boldsymbol{\omega}$ implies that $\kappa_{\text{RB}}(\boldsymbol{\omega})$ is non-negative and sums to one. Hence $\kappa_{\text{RB}}(\boldsymbol{\omega})$ can be thought of as a probability density and (8) approximated by an empirical average:

$$\mathscr{K}_{\text{RB}}(\boldsymbol{\lambda}) = E[e^{-i\boldsymbol{\omega}^\top\boldsymbol{\lambda}}] \approx \frac{1}{m}\sum_{j=1}^{m} e^{-i\boldsymbol{\omega}_j^\top\boldsymbol{\lambda}}, \quad \boldsymbol{\omega}_1,\dots,\boldsymbol{\omega}_m \sim \kappa_{\text{RB}}(\boldsymbol{\omega}).$$

The summation can be expressed as $\langle\hat{\Psi}_{\text{RB}}(\mathbf{x}),\hat{\Psi}_{\text{RB}}(\mathbf{y})\rangle$, where the approximated feature map $\hat{\Psi}_{\text{RB}}(\mathbf{x})$ is given by

$$[\hat{\Psi}_{\text{RB}}(\mathbf{x})]_j = \frac{1}{\sqrt{m}}e^{-i\boldsymbol{\omega}_j^\top\mathbf{x}}, \quad j = 1,\dots,m.$$

Note that the vectors $\boldsymbol{\omega}_j\in\mathbb{R}^d$ are randomly sampled from $\kappa_{\text{RB}}(\boldsymbol{\omega})$ and act on the data $\mathbf{x}$ as random projections. Note also that the complex vector $\hat{\Psi}_{\text{RB}}(\mathbf{x})$ can be equivalently written as a $2m$ dimensional real vector in terms of sine and cosine functions [14]. Its form for the exponential kernel is given in Figure 1.

2.3 Generalized RBF kernels

The *generalized RBF kernels* extend the RBF kernels to use a metric not necessarily Euclidean. For our purposes, this is best seen in terms of kernels. Recall that, for any PD kernel $K(\mathbf{x},\mathbf{y})$, the equation

$$D^2(\mathbf{x},\mathbf{y}) = K(\mathbf{x},\mathbf{x}) + K(\mathbf{y},\mathbf{y}) - 2K(\mathbf{x},\mathbf{y}) \tag{11}$$

defines a corresponding squared metric [15]. For instance, from the intersection, χ^2, JS, and Hellinger's kernel one obtains the l^1 distance, and the squared χ^2, JS, and Hellinger's distances respectively. Given an RBF kernel $K_{\text{RB}}(\mathbf{x},\mathbf{y}) = k(\|\mathbf{x}-\mathbf{y}\|_2^2)$, one can then obtain a corresponding generalized variant

$$K_{\text{RB}D^2}(\mathbf{x},\mathbf{y}) = k(D^2(\mathbf{x},\mathbf{y})). \tag{12}$$

Constructing an approximate feature map for (12) involves two steps. First, the feature map (7) can be used to approximate $D^2(\mathbf{x},\mathbf{y})$ as *Euclidean distance* in feature space:

$$\begin{aligned} D^2(\mathbf{x},\mathbf{y}) &\approx \hat{K}(\mathbf{x},\mathbf{x}) + \hat{K}(\mathbf{y},\mathbf{y}) - 2\hat{K}(\mathbf{x},\mathbf{y}) \\ &= \langle\hat{\Psi}(\mathbf{x}),\hat{\Psi}(\mathbf{x})\rangle + \langle\hat{\Psi}(\mathbf{y}),\hat{\Psi}(\mathbf{y})\rangle - 2\langle\hat{\Psi}(\mathbf{x}),\hat{\Psi}(\mathbf{y})\rangle \\ &= \|\hat{\Psi}(\mathbf{x}) - \hat{\Psi}(\mathbf{y})\|_2^2. \end{aligned} \tag{13}$$

Compute a $2m$ dimensional approximate finite feature map for the exponential-χ^2 kernel $K(\mathbf{x},\mathbf{y}) = \exp(-\frac{1}{2\sigma^2}\chi^2(\mathbf{x},\mathbf{y}))$.

Preprocessing: Draw m random vectors $\boldsymbol{\omega}$, sampled from a $(2n+1)d$ isotropic Gaussian of variance $1/\sigma^2$.

Given: A vector $\mathbf{x} \in \mathbb{R}^d$.
Compute: The approximate feature map $\hat{\Psi}_{\text{RB}\chi^2}(\mathbf{x})$

1: Construct the $2n+1$ dimensional vector $\hat{\Psi}(\mathbf{x})$ by setting for $j = 0,\ldots,2n$

$$[\hat{\Psi}(\mathbf{x})]_j = \begin{cases} \sqrt{xL\,\text{sech}(0)}, & j = 0, \\ \sqrt{2xL\,\text{sech}(\pi\frac{j+1}{2}L)}\cos\left(\frac{j+1}{2}L\log x\right) & j > 0 \text{ odd}, \\ \sqrt{2xL\,\text{sech}(\pi\frac{j}{2}L)}\sin\left(\frac{j}{2}L\log x\right) & j > 0 \text{ even}, \end{cases} \tag{9}$$

2: Construct the $2m$ dimensional vector $\hat{\Psi}_{\text{RB}\chi^2}(\mathbf{x})$ by setting for $j = 1,\ldots,2m$

$$[\hat{\Psi}_{\text{RB}\chi^2}(\mathbf{x})]_j = \begin{cases} \frac{1}{\sqrt{m}}\cos\left(\boldsymbol{\omega}_{\frac{j+1}{2}}^\top \hat{\Psi}(\mathbf{x})\right), & j \text{ odd}, \\ \frac{1}{\sqrt{m}}\sin\left(\boldsymbol{\omega}_{\frac{j}{2}}^\top \hat{\Psi}(\mathbf{x})\right), & j \text{ even}. \end{cases} \tag{10}$$

Figure 1: **Feature map for the exponential-χ^2 kernel.** The resulting vector is $2m$ dimensional. Here n controls the χ^2 approximation, and is typically chosen as a small number, e.g. $n = 1$ (and in this case $L \approx 0.8$, see [20] for details on how to choose this parameter). The algorithm requires only two modifications for any other RBF-D^2 kernel. First, (9) should be adjusted according to (6) to match the metric D^2 (closed forms are given in [20]). Second, the projections $\boldsymbol{\omega}_j$ should be sampled from the density $\kappa_{\text{RB}}(\boldsymbol{\omega})$ corresponding to the desired RBF profile as given by (8).

Hence the generalized RBF kernel $K_{\text{RB}D^2}$ can be approximated by the RBF kernel

$$K_{\text{RB}D^2}(\mathbf{x},\mathbf{y}) \approx K_{\text{RB}}(\hat{\Psi}(\mathbf{x}),\hat{\Psi}(\mathbf{y})) = k(\|\hat{\Psi}(\mathbf{x}) - \hat{\Psi}(\mathbf{y})\|_2^2).$$

Second, the random Fourier features (Sect. 2.2) can be used to approximate K_{RB}. Composing the two approximations yields an approximated feature map for the generalized RBF kernel:

$$\boxed{[\hat{\Psi}_{\text{RB}D^2}(\mathbf{x})]_j = e^{-i\boldsymbol{\omega}_j^\top \hat{\Psi}(\mathbf{x})}, \quad j = -n,\ldots,n.} \tag{14}$$

The complete procedure (from measured feature vector $\mathbf{x}$ to approximating feature vector $\hat{\Psi}_{\text{RB}\chi^2}(\mathbf{x})$ for the RBF-χ^2 kernel is given in Figure 1.

Errors and computational cost. The cost of computing the feature map (14) is $O(mnd)$, where m is the number of random Fourier features, n the dimensionality of the feature map (7) of the additive kernel, and d is the dimensionality of the data $\mathbf{x}$. As shown in [20], usually small values of n (e.g. $n = 1,2$) are sufficient to yield good accuracy. In particular, it can be shown (the proof is omitted for brevity) that the error decreases exponentially fast with n for the smooth kernels such as χ^2. Based on (11) and (12), the error in approximating the additive kernel propagates to the RBF-D^2 kernel multiplied by the Lipschiz constant of

the RBF kernel profile k, which is usually small. Overall, it can be shown that the error is dominated by approximating the RBF kernel by the m random Fourier features. In particular, based on the error analysis of [14], $m = \Omega(1/\varepsilon^2)$ random projections are needed to achieve a uniform approximation error ε with any fixed (large) probability.

3 Learning and reducing the weight (w) dimension

A limitation of the random Fourier features is the relatively large number of projections required to obtain good accuracy (Sect. 4). As seen in Sect. 2.3, the accuracy improves only as $O(1/\sqrt{m})$ with the number of projections, and the cost of evaluating the feature map $\hat{\Psi}_{\text{RBD}^2}(\mathbf{x})$ is $O(mdn)$. The parameter n can usually be small (e.g. $n = 1$), but m is often in the order of several thousands. Note, however, that the random Fourier features $\boldsymbol{\omega}_j$ can be "optimized" in several ways. For instance, it is possible to select (e.g. learn) from a large set of such features only the ones that are useful for a *specific* problem.

The simplest way to select useful random Fourier features is to use an appropriate regularizer for SVM training. Recall that in training a linear SVM one solves the problem

$$\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{N}\sum_{i=1}^{N} l(\mathbf{x}_i, y_i; \mathbf{w}), \quad l(\mathbf{x}_i, y_i; \mathbf{w}) = \max\{0, 1 - y_i \langle \mathbf{w}, \hat{\Psi}_{\text{RBD}^2}(\mathbf{x}_i)\rangle\} \tag{15}$$

where $\mathbf{x}_1, \ldots, \mathbf{x}_N$ are training vectors, $y_1, \ldots, y_N \in \{-1, +1\}$ their labels, and $l(\mathbf{x}_i, y_i; \mathbf{w})$ the hinge loss. We consider two other formulations where l^1 regularization is used in order to encourage sparsity in the $\mathbf{w}$ vector, and for which efficient implementations exist in LIBLINEAR [9]:

$$\min_{\mathbf{w}} \tfrac{1}{2}\|\mathbf{w}\|_1 + \tfrac{C}{N}\textstyle\sum_{i=1}^{N} l(\mathbf{x}_i, y_i; \mathbf{w})^2, \tag{16}$$

$$\min_{\mathbf{w}} \tfrac{1}{2}\|\mathbf{w}\|_1 + \tfrac{C}{N}\textstyle\sum_{i=1}^{N} \log(1 + \exp(-y_i \langle \Psi(\mathbf{x}_i), \mathbf{w}\rangle)). \tag{17}$$

these are known respectively as $l^1 l^2$-SVM and l^1-logistic regression. We will refer to these implementations as $\text{SVM}^{\text{sparse}}$ and $\text{LR}^{\text{sparse}}$ respectively, and to the standard SVM of (15) as $\text{SVM}^{\text{dense}}$.

From (10) we see that successive components of $\hat{\Psi}_{2j-1}(\mathbf{x})$, $\hat{\Psi}_{2j}(\mathbf{x})$ of the feature map share the same projection $\boldsymbol{\omega}_j$. Thus if $\mathbf{w}_{2j-1} = \mathbf{w}_{2j} = 0$ the projection $\boldsymbol{\omega}_j$ can be discarded in the evaluation of the SVM $\langle \mathbf{w}, \hat{\Psi}_{\text{RBD}^2}(\mathbf{x})\rangle$. Setting such projections to zero can be encouraged by considering the problem

$$\min_{\mathbf{w}} \frac{1}{2}\sum_{j=1}^{m} \sqrt{\mathbf{w}_{2j-1}^2 + \mathbf{w}_{2j}^2} + \frac{C}{N}\sum_{i=1}^{N} l(\mathbf{x}_i, y_i; \mathbf{w}), \tag{18}$$

This problem is known as Multiple Kernel Learning SVM (MKL-SVM) [2, 18] where we defined a base kernel for each projection $\boldsymbol{\omega}_j$.

Notice that, compared to the MKL formulation (18), the $\text{SVM}^{\text{sparse}}$ and $\text{LR}^{\text{sparse}}$ formulations encourage any coefficient, but not specifically *pairs* of related coefficients, to be zero. In practice, thus, we can remove a projection $\boldsymbol{\omega}_j$ only if both $\mathbf{w}_{2j-1}$ and $\mathbf{w}_{2j}$ are set to zero. This is usually sufficient to discard a very large number of projections, but we would expect an implementation of the MKL formulation to results in an even larger number of discarded projections. We donot investigate MKL any further here though.

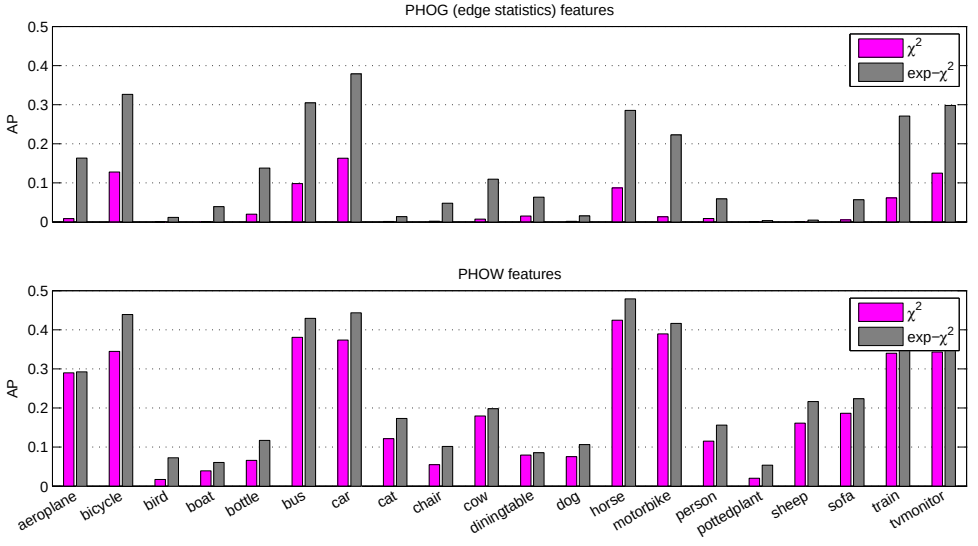

Figure 2: **Effect of RBF kernels on different feature types.** AP of the sliding window object detector [7] for PHOW and PHOG features and χ^2 and exp-χ^2. The PHOG features benefit substantially from the use of the exponential kernel, while the PHOW features are much less sensitive.

4 Experiments

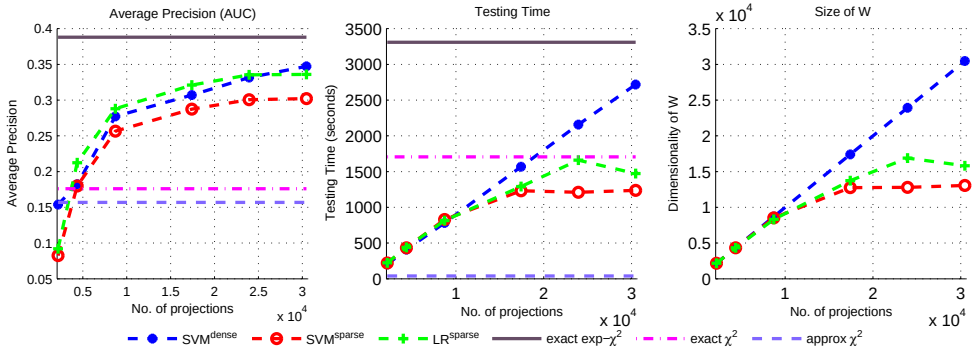

Figure 3: **Additive, exponential, and approximated kernels.** Performance of a car detector. The baselines are the exact exp-χ^2 kernel (corresponding to the third stage of the cascade [7]) and the exact additive χ^2 kernel (second stage). The exponential variant is better by 20% AP. The approximated exp-χ^2 kernel is shown for an increasing number of random Fourier features m (results are averaged over five sets of random projections). The approximation trades-off speed and accuracy. For instance, at 30% AP the approximation is twice as fast as the exact exp-χ^2 kernel and twice as accurate as the exact additive χ^2 kernel. The sparse solutions found by SVM^{sparse} and LR^{sparse} are faster still with only a small impact on performance (and in some case with improved performance).

We evaluate the proposed feature maps as part of the construction of an object detector on the PASCAL VOC 2007 [8] data. The VOC detection challenge involves predicting the bounding box and label of each instance of the target class in several thousand test images.

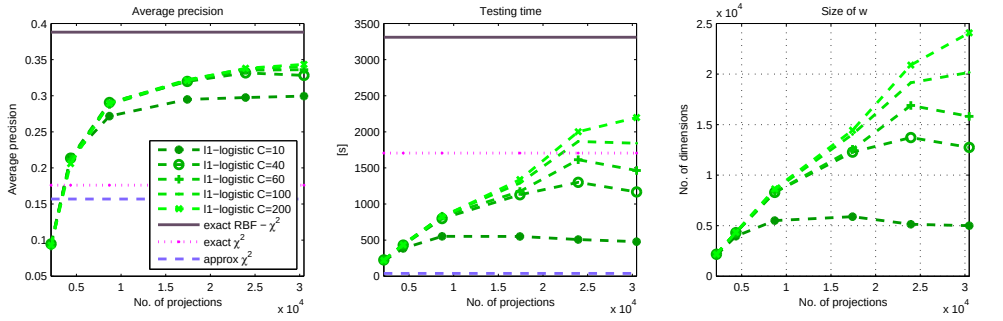

Figure 4: **Sparsity vs performance.** In order to make the approximated random Fourier features more competitive, we perform a random selection of the useful projections. Based on the formulations of Sect. 3, the only parameter that controls sparsity is C, which also controls overfitting. As C is increased from 10 to 200 the AP matches the one of the dense SVM, but also the testing time. For low value of C it is however possible to obtain fairly competitive AP (about 30%, still much better than the exact χ^2 kernel) with a seven-fold increase of speed compared to the exact kernel.

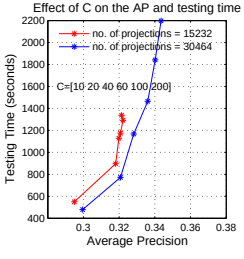

Figure 5: **Effect of C on sparsity.** Both SVM$^{\text{sparse}}$ and the LR$^{\text{sparse}}$ control sparsity through the regularization parameter C. The figure illustrates for LR$^{\text{sparse}}$ the variation of testing time (inversely proportional to the sparsity of the learned weight vector $\mathbf{w}$) and average precision as C is varied. The experiment is averaged over five sets of random projections, and repeated for sets of 15×10^3 and 30×10^3 projections. Notice how searching among more projections finds smaller sets of projections for a given level of accuracy.

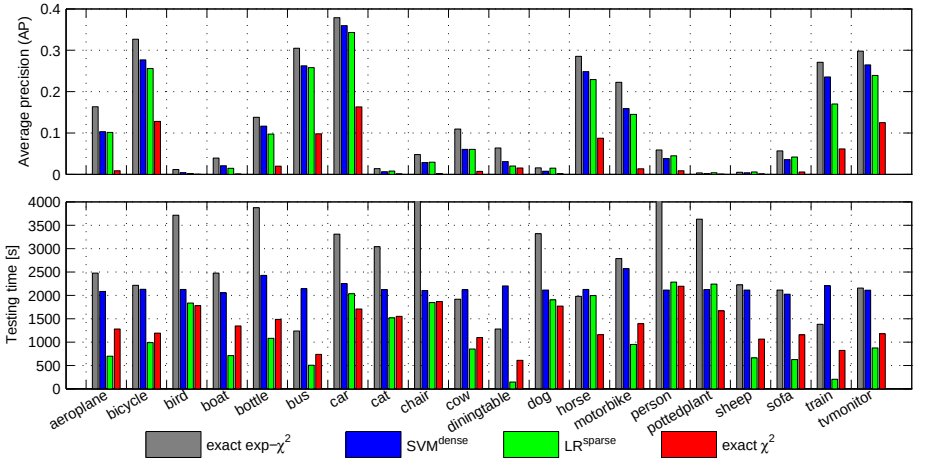

Figure 6: **Effect on all VOC classes** Exact exponential and additive χ^2 kernels for the twenty classes of the VOC 2007 detection challenge, along with their approximations. Top: both the dense and sparse approximations perform nearly as well as the exact kernel. Bottom: the testing time of the approximated sparse SVM is from two to three times better than the dense SVM.

We work on top of the state-of-the-art multiple-stage detector proposed in [21]. The multiple stages are a cascade of a linear, χ^2 and exponential-χ^2 (exp-χ^2) detector. Thus the feature

map (14) can be used to speed-up the third stage of the cascade (exp-χ^2), which is also noted to be the bottleneck in [21].

Features. While [21] uses up to six image descriptors, here we focus only on PHOG and PHOW , since these illustrate the main features of the proposed approach. PHOW are visual words obtained from rotationally invariant SIFT descriptors extracted on a regular grid with five pixels spacing, at four multiple scales (10, 15, 20, 25 pixel radii), zeroing the low contrast ones. Descriptors are then quantized in 300 visual words. PHOG [4] is a sparse multi-scale version of HOG [7]. The Canny edge detector is used to compute an edge map and the underlying image gradient is used to assign an orientation and a weight to each edge pixel. The orientation angle is then quantized in sixteen bins with soft linear assignment and an histogram is computed. Both PHOW and PHOG features are then converted into spatial histograms [11] with 1×1, 2×2, and 4×4 subdivisions in order to characterize each candidate object bounding box B. Overall each region is described by a 6,300 dimensional vector $\mathbf{x}$ for the PHOW features and 336 dimensional for the PHOG features.

Figure 2 compares the performance of the exact kernels with PHOW and PHOG features. A first important observation is that, while the PHOG features benefit substantially from the exponential kernel, the improvements with the PHOW features is much more limited. Our interpretation is as follows: The PHOG features (edglets) are not very discriminative in isolation, but their *combination*, capturing the shape of the object, is. This makes the local exponential kernel particularly important (for its template matching ability). The PHOW features, on the other hand, are quite discriminative in isolation since they match the semi-local SIFT 'footprint', and can be used with an additive kernel that, by operating independently on each component, is more similar to a voting process than to template matching. In this case, just the *presence* of certain visual words is important, not so much their specific combination. Thus less is gained by the exp-χ^2 kernel over the χ^2 kernel. Therefore, in the rest of the experiments we focus only on the PHOG features.

Approximated kernels. Figure 3 compares the exact and approximated exp-χ^2 generalized RBF kernels and the χ^2 additive homogeneous kernel on the object class *car*. The exact exp-χ^2 kernel performs much better than χ^2 for the PHOG features. The dense approximated version, $\text{SVM}^{\text{dense}}$, converges to the exact exp-χ^2 performance as more random projections are added. With around 10^4 projections the approximated exp-χ^2 kernel is already much better than the χ^2 kernel, and it is about seven times faster in testing than the exact exp-χ^2. The sparse SVM, and especially the sparse logistic regression, can further discard up to half of the projections as redundant, without impacting accuracy significantly. In this example, the approximation does not improve training time compared to the exact kernel due to the limited amount of training data; however, the training complexity is just linear, compared to quadratic of the exact kernel, so that the approximate representation would be better for large enough data sets. The exact RBF kernel, which could be approximated by using directly the technique from [14], was also tested but resulted in extremely poor performance (below 11% AP). This is due to the fact that the l^2 metric is a particularly poor match for the PHOG features. Figures 4 and 5 illustrate in more detail the effect of C on the sparsity and speed and their trade-off. It can be seen that an AP performance far superior to that of an exact χ^2 kernel can be achieved at a lower test cost. For example for $C = 10$ and 2×10^4 projections the AP is about twice that of exact χ^2 and it is about three times faster. Figure 6 shows that similar effects hold for all the 20 VOC classes.

5 Conclusions

We have introduced a method to construct a finite dimensional approximate feature map for the generalized RBF kernels. In general, the approximation is independent of the number of support vectors, yields linear training complexity, and may easily be included into an online training framework. We have shown that the finite feature map can be used to speedup testing significantly in a detection task while still yielding an accuracy far superior to that of the additive kernels for certain visual features.

Acknowledgments

We are grateful for financial support from the UKIERI, ONR MURI N00014-07-1-0182 and ERC grant VisRec no. 228180.

References

[1] F. R. Bach and M. I. Jordan. Predictive low-rank decomposition for kernel methods. In *ICML*, 2005.

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

[3] L. Bo and C. Sminchisescu. Efficient match kernels between sets of features for visual recognition. In *Proc. NIPS*, 2009.

[4] A. Bosch, A. Zisserman, and X. Muñoz. Scene classification via pLSA. In *Proc. ECCV*, 2006.

[5] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *PAMI*, 24(5), 2002.

[6] G. Csurka, C. R. Dance, L. Dan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *Proc. ECCV Workshop on Stat. Learn. in Comp. Vision*, 2004.

[7] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proc. CVPR*, 2005.

[8] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html, 2008.

[9] R.-E. Fan, K.-W. Chang, C.-J Hsieh, X.-R. Wang, and C.-J. Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9, 2008.

[10] T. Joachims. Training linear SVMs in linear time. In *Proc. KDD*, 2006.

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

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

[13] S. Maji and A. C. Berg. Max-margin additive classifiers for detection. In *Proc. ICCV*, 2009.

[14] A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *Proc. NIPS*, 2007.

[15] B. Schölkopf. The kernel trick for distances. *Proc. NIPS*, 2001.

[16] S. Shalev-Shwartz, Y. Singer, and N. Srebro. Pegasos: Primal Estimated sub-GrAdient SOlver for SVM. In *Proc. ICML*, 2007.

[17] A. Torralba and A. Oliva. Statistics of natural images categories. *Network: Computation in Neural Systems*, 14, 2003.

[18] M. Varma and D. Ray. Learning the discriminative power-invariance trade-off. In *Proc. ICCV*, 2007.

[19] A. Vedaldi and A. Zisserman. Structured output regression for detection with partial occulsion. In *Proc. NIPS*, 2009.

[20] A. Vedaldi and A. Zisserman. Efficient additive kernels via explicit feature maps. In *Proc. CVPR*, 2010.

[21] A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman. Multiple kernels for object detection. In *Proc. ICCV*, 2009.

[22] C. K. I. Williams and M. Seeger. Using the Nyström method to speed up kernel machines. In *Proc. NIPS*, 2001.

[23] 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.