MASSACHUSETTS INSTITUTE OF TECHNOLOGY
ARTIFICIAL INTELLIGENCE LABORATORY
and
CENTER FOR BIOLOGICAL AND COMPUTATIONAL LEARNING
DEPARTMENT OF BRAIN AND COGNITIVE SCIENCES

A.I. Memo No. 1635 May, 1998
C.B.C.L Paper No. 162

Sparse Correlation Kernel Analysis and Reconstruction

Constantine P. Papageorgiou, Federico Girosi and Tomaso Poggio

This publication can be retrieved by anonymous ftp to publications.ai.mit.edu.
The pathname for this publication is: ai-publications/1500-1999/AIM-1635.ps

Abstract

This paper presents a new paradigm for signal reconstruction and superresolution, Correlation Kernel Analysis (CKA), that is based on the selection of a sparse set of bases from a large dictionary of class-specific basis functions. The basis functions that we use are the correlation functions of the class of signals we are analyzing. To choose the appropriate features from this large dictionary, we use Support Vector Machine (SVM) regression and compare this to traditional Principal Component Analysis (PCA) for the tasks of signal reconstruction, superresolution, and compression. The testbed we use in this paper is a set of images of pedestrians. This paper also presents results of experiments in which we use a dictionary of multiscale basis functions and then use Basis Pursuit De-Noising to obtain a sparse, multiscale approximation of a signal. The results are analyzed and we conclude that 1) when used with a sparse representation technique, the correlation function is an effective kernel for image reconstruction and superresolution, 2) for image compression, PCA and SVM have different tradeoffs, depending on the particular metric that is used to evaluate the results, 3) in sparse representation techniques, L_1 is not a good proxy for the true measure of sparsity, L_0, and 4) the L_ϵ norm may be a better error metric for image reconstruction and compression than the L_2 norm, though the exact psychophysical metric should take into account high order structure in images.

This report describes research done within the Center for Biological and Computational Learning in the Department of Brain and Cognitive Sciences and at the Artificial Intelligence Laboratory at the Massachusetts Institute of Technology. This research is sponsored by MURI grant N00014-95-1-0600 and by a ONR/ARPA grant under contract N00014-92-J-1879. Additional support is provided by Eastman Kodak Company, Daimler-Benz, Siemens Corporate Research, Inc. and AT&T.

1 Introduction

This paper presents Correlation Kernel Analysis (CKA), a new paradigm for signal reconstruction and compression that is based on the selection of a sparse set of bases from a large dictionary of class-specific basis functions. The concept of sparsity enforces the requirement that, given a certain reconstruction error, we should choose the smallest subset of basis functions that yields a reconstruction with this error. The problem of signal reconstruction is formulated as one where we are given only a small, possibly unevenly sampled, subset of points in a signal where the goal is to accurately reconstruct the entire signal. We also investigate a closely related subject, lossy compression, that is, given an entire signal of N bits, we see how well we can represent the signal with only $M \ll N$ bits of information, using the same general technique.

The signal approximation problem we present assumes that we have prior information about the class of signals we are reconstructing or compressing; this information is in the form of the correlation function of the class of signals to which this signal belongs, as defined by a representative set of signals from this class (Penev and Atick, 1996; Poggio and Girosi, 1998a; Poggio and Girosi, 1998b). For this paper, the signals that we will be looking at are images of pedestrians (Papageorgiou, 1997; Oren, et al., 1997; Papageorgiou, et al., 1998). Using an initial set of pedestrian images, we compute the correlation function and use the pointwise-defined functions as the dictionary of basis functions from which we can reconstruct subsequent out-of-sample images of pedestrians. Our choice of using the correlation kernel can be motivated from a Bayesian point of view. We show that, if we assume a gaussian noise process on our measurements, the kernel to use, in a Bayesian sense, is the correlation kernel.

To approximate or reconstruct an image, rather than using the entire set of correlation-based basis functions comprising the dictionary – this would result in no compression whatsoever – we choose a small subset of the kernels via the criteria of sparsity. We obtain a sparse representation by approximating the signal using the Support Vector Machine (SVM) (Boser, Guyon, and Vapnik, 1992; Vapnik, 1995) formulation of the regression problem. Based on recently reported results (Girosi, 1997; Girosi, 1998), we note that this framework is equivalent to using a modified version of the Basis Pursuit De-Noising (BPDN) approach of Chen, Donoho, and Saunders (1995) to obtaining a sparse representation of a signal.

We push this paradigm further by investigating the use of dictionaries of multiscale basis functions that encode different levels of detail. To obtain a sparse, multiscale approximation of a signal, we use BPDN; this leads to improved reconstruction error and a more sparse representation. We also show that the empirical results highlight a drawback in using traditional formulations of sparsity.

The results presented in this paper can be useful in low-bandwidth videoconferencing, image de-noising, reconstruction in the presence of occlusions, signal approximation from sparse data, as well as in superresolving images. It is important to note that the results are not particular to image analysis; this technique can also be seen as an alternative to traditional means of function approximation and signal reconstruction, such as Principal Components Analysis (PCA), for a wider class of signals.

The paper is organized as follows: in Section 2, we introduce generalized correlation kernels and Section 3 provides Bayesian motivation for our choice of kernels. Section 4 describes the concept of sparsity and presents both the SVM regression and BPDN formulations of this approach. In Section 5, we present results of several image reconstruction experiments using CKA for sparse approximations with the generalized correlation kernels and describe a superresolution reconstruction experiment. Section 6 presents results of image compression experiments and a comparison between SVM and BPDN on this task. In Section 7, we show results of experiments that use a dictionary with basis functions at multiple scales to do lossy image compression using BPDN. Section 8 discusses the error norms that our different reconstruction techniques use and their psychophysical plausibility. Section 9 summarizes our results and presents several observations and open questions.

2 Generalized Correlation Kernels

To reconstruct or compress a function f, we use information about the class of pointwise mean-normalized signals that f is a part of, derived from a set of representative examples from that class. This information is in the form of the correlation function of the signals in the class:

$$R(\mathbf{x},\mathbf{y}) = E[(f_\alpha(\mathbf{x}) - \mu(\mathbf{x}))(f_\alpha(\mathbf{y}) - \mu(\mathbf{y}))] \tag{1}$$

where f_α are instances of the class of functions to which f belongs, $\mathbf{x}$ and $\mathbf{y}$ are coordinates in the 2-dimensional signal, and μ are the point means across the class of functions: $\mu(\mathbf{x}) = E[f_\alpha(\mathbf{x})]$. We can also generate the eigen-decomposition of the symmetric, positive definite correlation matrix by solving

$$\int d\mathbf{x} R(\mathbf{x},\mathbf{y})\phi_n(\mathbf{x}) = \lambda_n \phi_n(\mathbf{y}) \tag{2}$$

where ϕ_n are the eigenvectors and λ_n are the eigenvalues of the system. After generating this decomposition, we can write R in the form,

$$R(\mathbf{x},\mathbf{y}) = \sum_{n=1}^{M} \lambda_n \phi_n(\mathbf{x})\phi_n(\mathbf{y}) \tag{3}$$

where $M \leq \infty$; this result is due to the spectral theorem.
The set of functions ϕ_n are ordered with decreasing positive eigenvalue λ_n and are normalized to form an orthonormal basis for the correlation function of f_α. The classical Principal Component Analysis (PCA) approach approximates a function f as a linear combination of a finite number, M', of the basis functions ϕ_n:

$$f(\mathbf{x}) = \sum_{n=1}^{M'} b_n \phi_n(\mathbf{x}) \tag{4}$$

where the coefficients b_i are determined so as to minimize the L_2 approximation error of f.
Poggio and Girosi (1998a) show that the correlation function R, which is positive definite, induces a Reproducing Kernel Hilbert Space (RKHS) that allows us to approximate the function f as:

$$f(\mathbf{x}) = \sum_{i=1}^{N} c_i R(\mathbf{x},\mathbf{x}_i) \tag{5}$$

where i ranges over pixel locations in the image; R is the reproducing kernel in this space and the norm is:

$$\|f\|_R^2 = \sum_{n=1}^{M} \frac{c_n^2}{\lambda_n} \tag{6}$$

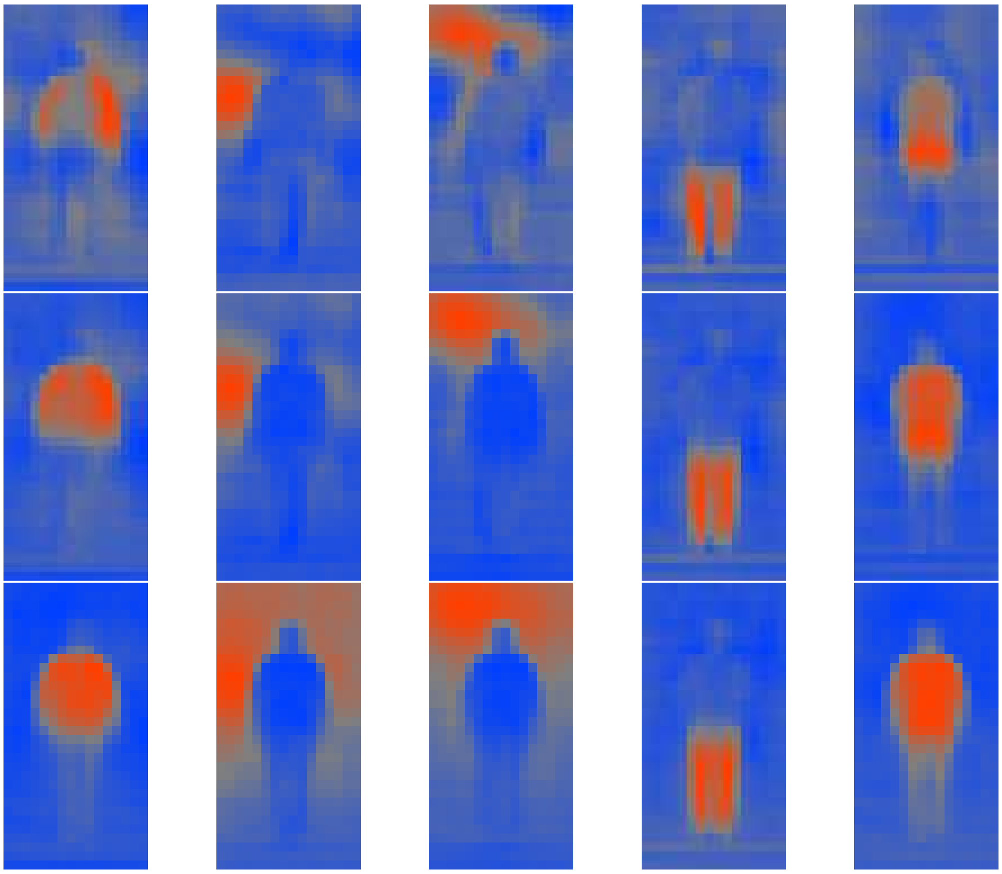


Figure 1: Examples of the correlation kernels we can compute. The kernels shown here are computed from a set of 924 grey-level 128×64 images of pedestrians that have been normalized to the same scale and position in the image. Each column shows the kernels, $R_d((x_1 = a, x_2 = b), \mathbf{y})$, for a specific (a, b) where $d = 0.0$, $d = 0.5$, and $d = 1.0$ in the top, middle, and bottom rows, respectively. These images demonstrate that $d = 1.0$ corresponds to a very smooth kernel, while $d = 0.0$ is highly localized.

We can obtain a wider class of kernels spanning exactly the same space of functions as the correlation function in Equation 3 by varying the degree of λ_n, which in effect controls the prior information regarding the strength of each eigenfunction, an observation due to Penev and Atick (1996). We therefore define the *generalized correlation kernel* as:

$$R_d(\mathbf{x}, \mathbf{y}) = \sum_{n=1}^{M} (\lambda_n)^d \phi_n(\mathbf{x}) \phi_n(\mathbf{y}) \tag{7}$$

and notice that the parameter d controls the locality of the kernel; for small d, R_d approaches a delta function in the space of ϕ_n, and as d gets larger, R_d gets smoother[1].

[1]This particular parameterization is one of many possibilities

Each of these correlation kernels is a function in four variables (x_1,x_2,y_1,y_2) so, to effectively visualize them, we hold the x_1 and x_2 positions constant and vary y_1 and y_2. Figure 1 shows several examples of the kernels generated with varying d, for a set of 924 grey-level 128×64 images of pedestrians that have been normalized to the same scale and position; this database has been used in Papageorgiou (1997), Oren, et al. (1997), and Papageorgiou, et al. (1998). Each column shows $R_d((x_1 = a, x_2 = b), \mathbf{y})$ for an image where, from the top to bottom rows, $d = 0.0$, $d = 0.5$, and $d = 1.0$; for example, the first column shows the kernels for $R_d((11, 10), \mathbf{y})$. The progressive delocalization of the kernels when d is varied from 0.0 to 1.0 is evident in these figures.

3 Bayesian Motivation

Our choice of the correlation function, R, as the kernel can be motivated from a Bayesian perspective; see Wahba (1990) and Poggio and Girosi (1998a) for background material. Consider the general regularization problem:

$$\min_{f \in \mathcal{H}} H[f] = \sum_{i=1}^{N} (y_i - f(\mathbf{x}_i))^2 + \gamma \|f\|_K^2 \tag{8}$$

In a Bayesian interpretation, the data term is a model of the noise and the stabilizer is a prior on the regression function f. If we assume that the data, y_i, are affected by additive independent gaussian noise, then the likelihood has the following form:

$$P(\mathbf{y}|f) \propto e^{-\sum_{i=1}^{N}(y_i - f(\mathbf{x}_i))^2} \tag{9}$$

and, when we use the correlation kernel R, the prior probability is:

$$P(f) \propto e^{-\|f\|_R^2} \propto e^{-\sum_{n=1}^{M} \frac{c_n^2}{\lambda_n}} \tag{10}$$

where $M < \infty$. As shown earlier, this corresponds to a representation of the form:

$$f(\mathbf{x}) = \sum_{n=1}^{M} c_n \phi_n(\mathbf{x}) \tag{11}$$

Thus, the stabilizer measures the Mahalanobis distance of f from the mean signal. This also corresponds to a zero mean multivariate gaussian density on the Hilbert space of functions defined by R and spanned by ϕ_n, e.g., the space spanned by the principal components introduced in Section 2. From a Bayesian point of view, under the assumption of gaussian noise, R is the right kernel to use, whenever it is available. It is important to note that in our SVM and BPDN formulations, we use gaussian priors but do not assume gaussian additive noise in the data.

4 Sparsity

The operational definition of a sparse representation in the context of regression that we will use is the smallest subset of elements from a large dictionary of features such that a linear superposition of these features can effectively reconstruct the original signal. In this paper, we will focus on sparse representations using the correlation kernels introduced in the previous section:

$$f(\mathbf{x}) = \sum_{i=1}^{N'} c_i R(\mathbf{x}, \mathbf{x}_i) \tag{12}$$

where N' is smaller than the size of the signal.
Suppose that we have a large dictionary of core building blocks for a class of signals we are analyzing. Given a new signal of the same class, obtaining a sparse representation of this signal amounts to choosing the smallest subset of building blocks from the dictionary that will allow us to achieve a certain level of performance. It is important to note that comparing representations for sparsity is only fair for a given performance criterion.
Here, we present a brief introduction to the concepts of Support Vector Machine regression and Basis Pursuit De-Noising as they apply to sparse representations; for a more in depth treatment of these subjects, the reader is referred to (Boser, Guyon, and Vapnik, 1992; Vapnik, 1995; Burges, 1998; Chen, Donoho, and Saunders, 1995; Girosi, 1997; Girosi, 1998).

4.1 Support Vector Machine Regression

Given a kernel K that defines a RKHS and with the appropriate choice of the scalar product induced by K, the empirical risk minimization regularization theory framework suggests to minimize the following functional:

$$H[f] = \frac{1}{N} \sum_{i=1}^{N} \| z_i - f(\mathbf{x}_i) \|_{L_2}^2 + \gamma \|f\|_K^2 \tag{13}$$

where $\|f\|_K^2$ is as defined in Section 2. This corresponds to minimizing the sum of the empirical error measured in L_2 and a smoothness functional. The Support Vector Machine regression formulation minimizes a similar functional, differing only in the norm on the data term; instead of using the L_2 norm, the following ϵ-insensitive error function, called the L_ϵ norm, is used:

$$|z_i - f(\mathbf{x}_i)|_\epsilon = \begin{cases} 0 & \text{if } |z_i - f(\mathbf{x}_i)| < \epsilon \\ |z_i - f(\mathbf{x}_i)| - \epsilon & otherwise \end{cases} \tag{14}$$

The functional that is minimized is therefore:

$$H[f] = \frac{1}{N} \sum_{i=1}^{N} |z_i - f(\mathbf{x}_i)|_\epsilon + \gamma \|f\|_K^2 \tag{15}$$

yielding a function of the form:

$$f(\mathbf{x}) = \sum_{i=1}^{N'} c_i R(\mathbf{x}, \mathbf{x}_i) \tag{16}$$

where the coefficients $\mathbf{c}$ are obtained by solving a quadratic programming problem (Vapnik, 1995; Osuna, Freund, and Girosi, 1997; Girosi, 1997). Depending on the value of the sparsity parameter γ, the number of c_i that differ from zero will be smaller than N; the data points associated with the non-zero coefficients are called *support vectors* and it is these support vectors that comprise our sparse approximation.

4.2 Basis Pursuit De-Noising

The Basis Pursuit De-Noising approach of Chen, Donoho, and Saunders (1995) is a means of decomposing a signal into a small number of constituent dictionary elements. The functional that is minimized consists of an error term and a sparsity term and in the case of arbitrary basis functions, ϕ_i, is:

$$E[\mathbf{c}] = \|f(\mathbf{x}) - \sum_{i=1}^{N} c_i \phi_i(\mathbf{x}_i)\|_{L_2}^2 + \lambda \|\mathbf{c}\|_{L_1} \tag{17}$$

In our case, to sparsify Equation 12, the following functional must be minimized (Girosi, 1997; Girosi, 1998):

$$E[\mathbf{c}] = \|f(\mathbf{x}) - \sum_{i=1}^{N} c_i R(\mathbf{x}, \mathbf{x}_i)\|_{L_2}^2 + \lambda \|\mathbf{c}\|_{L_1} \tag{18}$$

yielding an approximation to f that has a similar form to Equation 16. Girosi (1997) shows that if, instead of the L_2 norm, we use the norm induced by R, then Basis Pursuit De-Noising is in fact equivalent to Support Vector Machine regression and identical sparse representations are obtained.

This function minimization is formulated as a quadratic programming problem (see Appendix A) and can be solved using traditional methods. Appendix B presents a decomposition algorithm that allows us to quickly solve this minimization problem even when we have a large dictionary of basis functions.

5 Reconstruction

In the case of image reconstruction and compression when we do not assume any prior knowledge (other than that we are considering images), we can use techniques like JPEG, wavelets, and regularization using a spline or gaussian kernel. The focus of this paper is regularization schemes for the case where we do have statistical information on the class of functions we are reconstructing. When we do have such knowledge, as in the case of the correlational structure of the class to which the image to be compressed belongs, we may be able to obtain better compression by using this information. As described in the introductory sections, we can use the set of basis functions that encode the correlational structure of the class of images we are interesed in reconstructing. For a given image that we would like to approximate, we use these *class-specific* basis functions in the SVM formulation to obtain a sparse subset with which we can encode the image.

The generalized correlation kernels are generated from a training set of 924 grey-level 32×16 images of pedestrians that have been normalized to the same scale and position. We test the correlation kernels and the SVM formulation of function approximation by analyzing the reconstruction of pedestrian images not in the training set and comparing to the widely used PCA technique. The test database of pedestrian images consists of 50 out-of-sample 32×16 grey-level images of frontal and rear views of pedestrians; as in the training set, these images have been normalized such that the pedestrian bodies are aligned in the center of the image and are scaled to the same size.

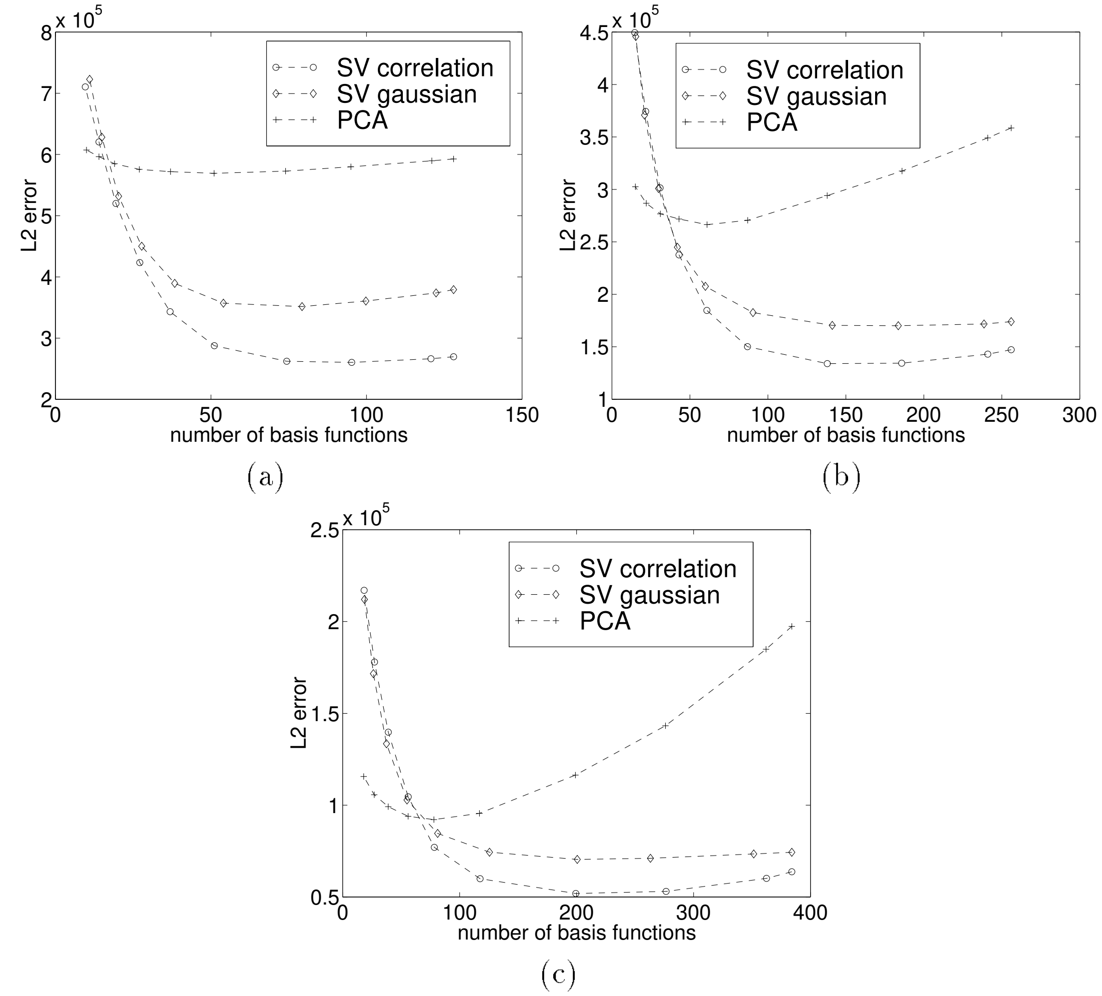


Figure 2: Out-of-sample L_2 reconstruction error comparison between SVM with correlation kernel $R_{1.0}$, SVM with gaussian kernel ($\sigma = 3.0$), and PCA, where the input is a random sampling of the original image. Each of these figures represents a different sized sampling, (a) $\frac{1}{4}$ of the image as input, (b) $\frac{1}{2}$ of the image as input, and (c) $\frac{3}{4}$ of the image as input.

For the SVM experiments, we use the correlation kernel corresponding to $d = 1.0$ as our dictionary of basis functions, so the reconstructed signal will be a sparse linear combination of those basis functions:

$$R_{1.0}(\mathbf{x}, \mathbf{y}) = \sum_{n=1}^{M} \lambda_n \phi_n(\mathbf{x}) \phi_n(\mathbf{y}) \tag{19}$$

To accurately test the reconstruction performance, we need to measure the ability of the technique to reconstruct unseen data and not simply fit the data. For each image in the test set, we randomly partition the pixels into a set that has M pixels – the input set, F_{input} – and a set consisting of the remaining $(N - M)$ pixels – the test set, F_{test}.
In the case of the SVM, to find the sparse set of basis functions that minimizes the error over the input subset, F_{input}, we obtain the coefficients of reconstruction by minimizing:

(a) (b) (c) (d)

Figure 3: Reconstruction comparison for a higher resolution image (64×32) using identical random sets of $\frac{1}{16}$th of the original pixels as input; (a) the original image, (b) PCA reconstruction with 74 basis functions, (c) SVM reconstruction with 74 basis functions ($\epsilon = 10$ for the SVM), (d) locations of the support vectors are denoted as black values. With a small subset of the original image as input, the SVM reconstruction is clearly superior to the PCA reconstruction.

$$H[f] = \frac{1}{M}\sum_{i=1}^{M} |F_{input}(\mathbf{x}_i) - f(\mathbf{x}_i)|_\epsilon^2 + \frac{1}{C}\|f\|_K^2 \tag{20}$$

where,

$$f(\mathbf{x}) = \sum_{i=1}^{M} c_i R(\mathbf{x}, \mathbf{x}_i) \tag{21}$$

The portion of the coefficients, c_i, that will be 0 is determined by the variable C.
For PCA-based reconstruction, we minimize L_2 error over F_{input}:

$$\min_c \sum_{i=1}^{M} \|F_{input}(\mathbf{x}_i) - \sum_{j=1}^{N} c_j \phi_j(\mathbf{x}_i)\|_{L_2}^2 \tag{22}$$

where c_j is given by the dot product between F_{input} and ϕ'_j is taken over the M input points:

$$c_j = \langle F_{input}, \phi'_j \rangle \tag{23}$$

Out-of-sample performance in each case is determined by reconstructing the full image and measuring the error over the pixels in f_{test}. We measure performance as the error achieved with respect to the number of basis functions used in the above formulations (equivalently, reconstruction error versus the sparsity of the representation). In the case of the SVM regression, the number of basis functions is varied by changing the ϵ parameter. To compare with PCA-based reconstruction, for a given ϵ, we use, as the number of principal components (ie. basis functions) for the reconstruction, the number of support vectors found in the SVM formulation. In our experiments, the size of the input set is varied as $\frac{1}{4}N$, $\frac{1}{2}N$, and $\frac{3}{4}N$; error is measured in L_2.

As a benchmark meant to ensure that the performance of the system using SVM with the correlation kernels is not due exclusively to the SVM machinery, we also show the results using SVM with gaussian kernels, yielding approximations of the form:

$$f(\mathbf{x}) = \sum_{i=1}^{M} c_i e^{\left(\frac{x-x_i}{\sigma}\right)^2} \tag{24}$$

where the value of σ is determined empirically over a small set of images and that same σ is used throughout. This setting of sigma for all the tests may be limiting the performance of the SVM with a gaussian kernel; on the other hand, we are also *a priori* fixing the locality parameter, d, in our choice of correlation kernel.

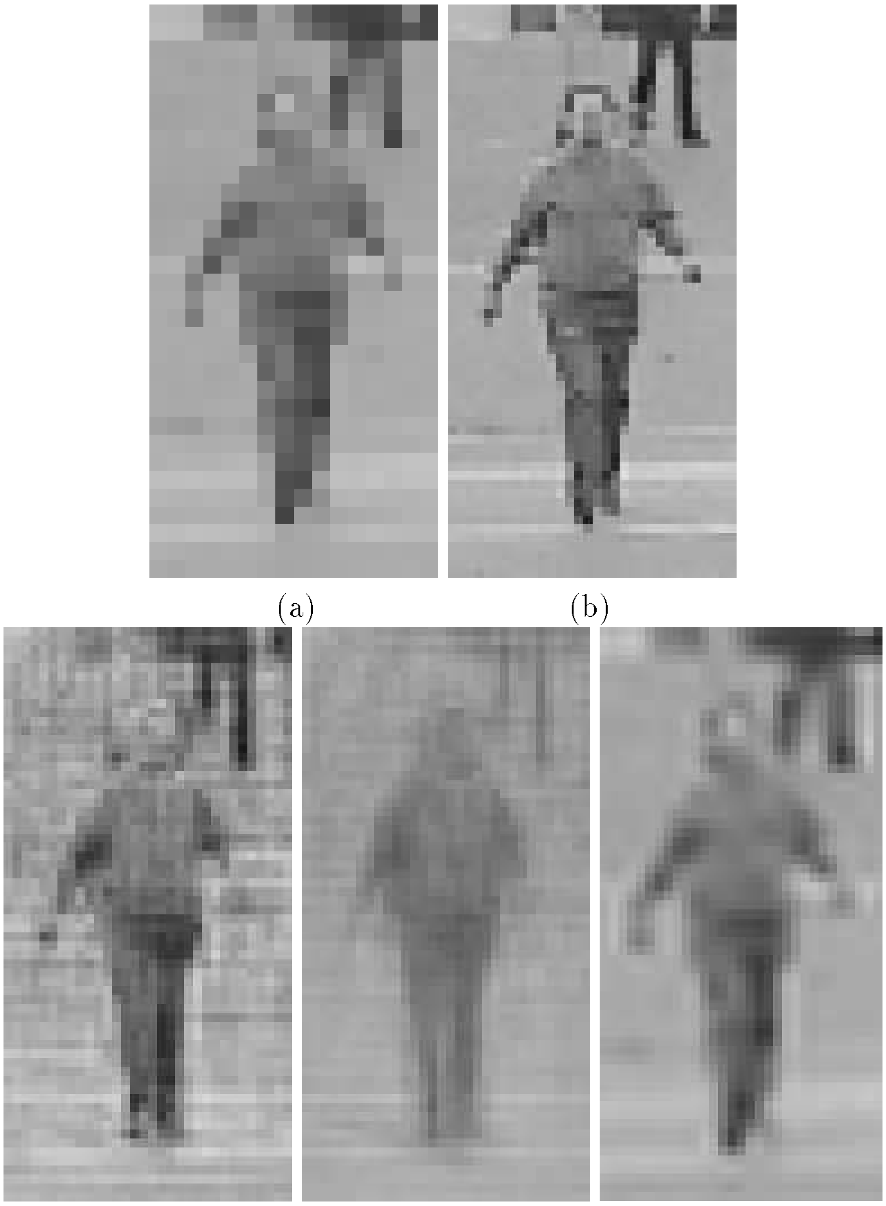
The results of these reconstructions, averaged over the 50 out-of-sample images, are shown in Figures 2a-c for each case of using $\frac{1}{4}$, $\frac{1}{2}$, and $\frac{3}{4}$ of the pixels as input, respectively. The SVM reconstructions using different numbers of basis functions were generated by varying ϵ. From these performance results, we can see that, even though the PCA formulation minimizes L_2 error and SVM regression is minimizing error in the RKHS induced by the epsilon insensitive norm, SVM performs better than PCA even when measuring error in L_2 over out-of-sample test data. Furthermore, SVM with the correlation kernels performs better than SVM with gaussian kernels, showing that the correlation kernels encode important prior information on the pedestrian class. The difference in performance is most pronounced for the reconstructions that use the smallest input set.

Figure 3 presents an extreme case where the input data is a random set of only $\frac{1}{16}$th (6.25%) of the image pixels; here, a higher resolution image (64×32) is used. The SVM reconstruction with correlation kernels recovers more of the structure of the pedestrian than PCA, due to the smoothness preserving properties of the SVM approach to function approximation (Vapnik, 1995).

5.1 Superresolution

To further highlight the generalization power of the SVM reconstruction, we can do an experiment to determine *superresolution* capability, that is, reconstructions at a finer level of detail than was originally present in the image. Superresolution entails approximating a small image with some representation and then sampling that representation at a finer scale to recover the higher resolution image. This could be useful if, for instance, we have an image of a person's face that is too small for us to be able to recognize who it is; after superresolving the image, the details that emerge could allow us to recognize the person.

This is not possible with our generalized correlation kernels since they are discrete kernels generated from high resolution images (64×32) and we cannot subsample them. Therefore, to superresolve a given 32×16 image, we can consider it as a 64×32 image sampled every two pixels in both dimensions and then use the correlation kernel basis functions defined in the high resolution space (64×32) to recover the full high resolution image.



(a) (b)

(c) (d) (e)

Figure 4: Superresolution reconstruction from a low resolution (32×16) sampling; (a) the input 32×16 image, scaled up to 64×32 by direct scaling, (b) the actual 64×32 image, (c) SVM superresolution reconstruction using 272 basis functions from $R_{1.0}$ ($\epsilon = 10$), (d) PCA superresolution reconstruction using 272 basis functions, and (e) cubic spline interpolation.

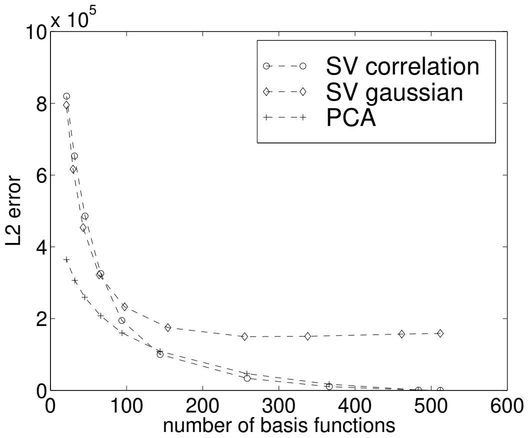
As input to the superresolution technique, we take a low resolution 32×16 image of a pedestrian and reconstruct it at high resolution (64×32). Figure 4 shows (a) an example of a 32×16 image of a pedestrian that has been directly scaled to 64×32 and (b) the true 64×32 pedestrian image. These are compared with (c) the superresolved image, reconstructed at 64×32 using the SVM with correlation kernels $R_{1.0}$, compared against both (d) a PCA reconstruction, and (e) a standard cubic spline interpolation reconstruction (Schumaker, 1981). Given the constraints presented above as well as the fact that the cubic spline interpolation superresolves the image quite well, for this specific experiment, we favor this standard spline technique over the correlation kernels.

6 Compression

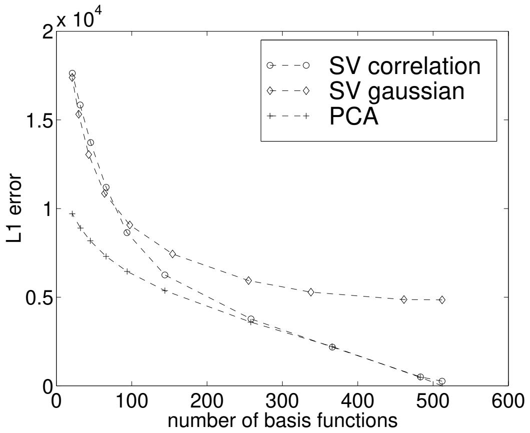
We can also investigate image *compression* using the set of correlation-based basis functions, in the same manner as the reconstruction experiments presented in Section 5. For the task of compression, the goal is to approximate the entire given signal f using as few basis functions as possible. The experiments are run as before; we compare the SVM regularization approach to compression with our benchmark, PCA-based compression. For the SVM approach, we use the correlation kernel with $d = 1.0$ and compare with using SVM with gaussian kernels. Performance is measured as the error achieved for a given number of basis functions. The number of basis functions that are used in the case of SVM regression are varied by changing the ϵ parameter. As in the reconstruction experiments, the number of eigenvectors we use to compare against PCA-based compression is the number of support vectors for given level of ϵ.
Figure 5 plots the reconstruction error against the number of basis functions for three different error norms: L_2, L_1, and L_ϵ. Comparing the SVM and PCA approaches to compression is less conclusive than the reconstruction experiments; the results here depend on the measure of error. PCA performs better when measured in L_2 and L_1 while SVM wins when measured in L_ϵ. The L_2 and L_ϵ results are not surprising; when error is measured in the norm that a technique is minimizing, we would expect that technique to perform better than the others. On the other hand, it is not clear which norm results in a reconstructed image that *appears* more similar to the original image; Section 8 contains a discussion of the different norms.

6.1 Comparing SVM and BPDN

Girosi (1997, 1998) showed that Basis Pursuit De-Noising is equivalent to Support Vector Machines when the L_2 norm in the BPDN formulation is replaced by the norm induced by the regularization kernel. Here, we empirically test the effect of the different error norms in the two approaches by comparing SVM and BPDN reconstruction error when compressing our test set of 50 pedestrian images. Both of these techniques are evaluated using the correlation kernel $R_{1.0}$. Figure 6 graphs the results and indicates that the performance of the two techniques is not identical. For representations using large numbers of basis functions, the performance is comparable, but BPDN obtains more accurate *sparse* approximations, when measured in L_2, to the original image (where the number of basis functions is less than 100). Again, the reason behind this is that we are measuring error in the norm that BPDN is explicitly minimizing.



(a)



(b)

	number of	SVM error with kernel $R_{1.0}$			PCA error		
ϵ	bases	L_2	L_1	L_ϵ	L_2	L_1	L_ϵ
0	512.00	258.45	258.45	258.45	0.00	0.17	0.17
1	483.32	744.37	499.96	122.20	1232.28	476.48	189.88
5	366.35	10495.71	2182.04	89.59	17318.51	2214.40	613.90
10	258.35	33565.92	3772.49	63.33	46563.25	3594.14	743.88
20	144.00	100221.65	6252.31	34.61	108767.78	5368.40	740.31
30	93.73	194718.31	8637.45	19.82	160343.38	6450.33	583.80
40	66.02	325743.27	11201.67	13.98	207841.38	7300.94	458.35
50	45.47	485822.18	13720.92	8.39	259697.64	8181.98	358.18
60	31.59	653269.84	15841.84	5.88	306557.81	8897.46	264.76
70	21.16	819763.43	17634.61	3.90	364437.23	9705.89	218.01

(c)

Figure 5: Comparison of compression error between SVM with correlation kernel $R_{1.0}$, SVM with gaussian kernel, and PCA; (a) L_2 error, (b) L_1 error, and (c) L_ϵ error. The L_ϵ results are presented in tabular format. The L_2 and L_1 results indicate that performance is comparable between SVM with the correlation kernel and PCA for large numbers of basis functions, but the SVM generates better sparse approximations (using less than 100 basis functions).

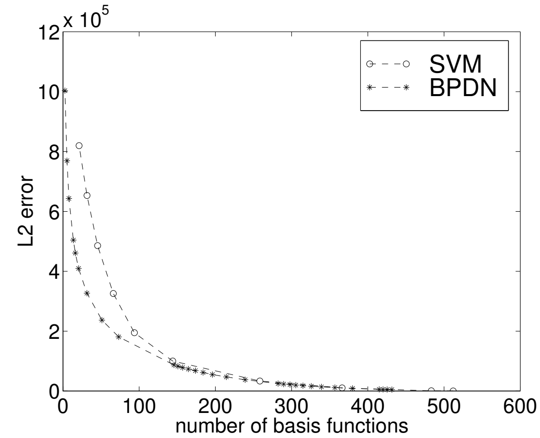


Figure 6: A comparison of SVM and BPDN measuring reconstruction error obtained when representing pedestrian images as a sparse set of correlation-based basis functions ($R_{1.0}$); L_2 reconstruction error is plotted against the number of basis functions found by each technique. The performance of these techniques is comparable for large numbers of basis functions, but BPDN obtains better sparse approximations, measured in L_2, to the original images (number of basis functions < 100).

7 Multiscale Representations

Multiscale representations allow us to represent a signal using successive levels of approximation; lower levels of resolution capture the coarse structure of the signal and finer levels resolution of resolution encode the details. These representations are standard in the signal processing literature (Mallat and Zhang, 1989; Simoncelli and Freeman, 1995; Mallat and Zhang, 1993).
In our image reconstruction experiments, we have focused on approximating a signal using a single kernel with $d = 1.0$, corresponding to coarse scale features. In certain applications, we may be able to derive class-specific basis functions for several scales; this is the case for our generalized correlation kernels where, to vary the locality of the basis functions, we simply change d. We can then use the sparsification paradigm on this larger overcomplete dictionary to obtain a sparse approximation of a given signal with a set of basis functions at several scales. The SVM formulation for multiple scales has not been derived yet, but Basis Pursuit De-Noising can be used with these multiscale dictionaries.
As introduced in Section 4.2, Basis Pursuit De-Noising is an approach to sparsification that minimizes a functional containing an term measuring the approximation error in L_2 using a linear combination of basis functions and a sparsity term in L_1. In our signal and reconstruction experiments, where we have focused on using a set of basis functions ϕ_n that are at a single scale, we would minimize:

$$E[\mathbf{c}] = \|f(\mathbf{x}) - \sum_{i=1}^{N} c_i \phi_i(\mathbf{x}_i)\|_{L_2}^2 + \lambda \|\mathbf{c}\|_{L_1} \tag{25}$$

for some signal f.
We can formulate the BPDN functional for our case of generating a multiscale representation using correlation kernels as follows:

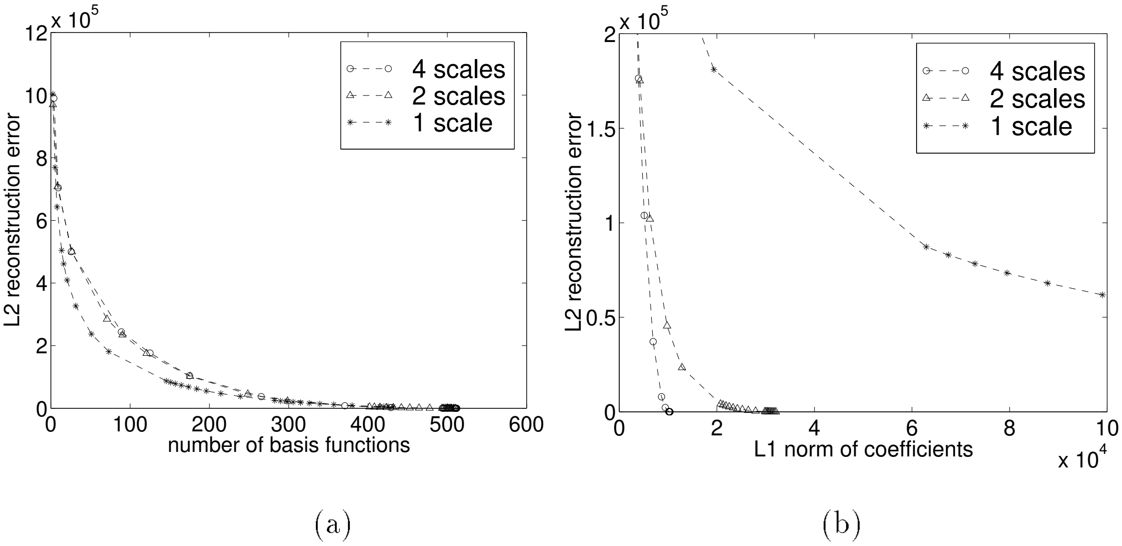


Figure 7: Compression error when using multiscale basis functions with BPDN; (a) L_2 error plotted against the L_0 norm of the coefficients (ie., the number of basis functions), (b) L_2 error plotted against the L_1 norm of the coefficients. These graphs imply that, in the context of sparsity, the L_1 norm is not a good approximation of L_0.

$$E[\mathbf{c}] = \|f(\mathbf{x}) - \sum_{i=1}^{N} \sum_{d=d_1}^{d_D} c_{i,d} R_d(\mathbf{x}, \mathbf{x}_i)\|_{L_2}^2 + \lambda \|\mathbf{c}\|_{L_1} \tag{26}$$

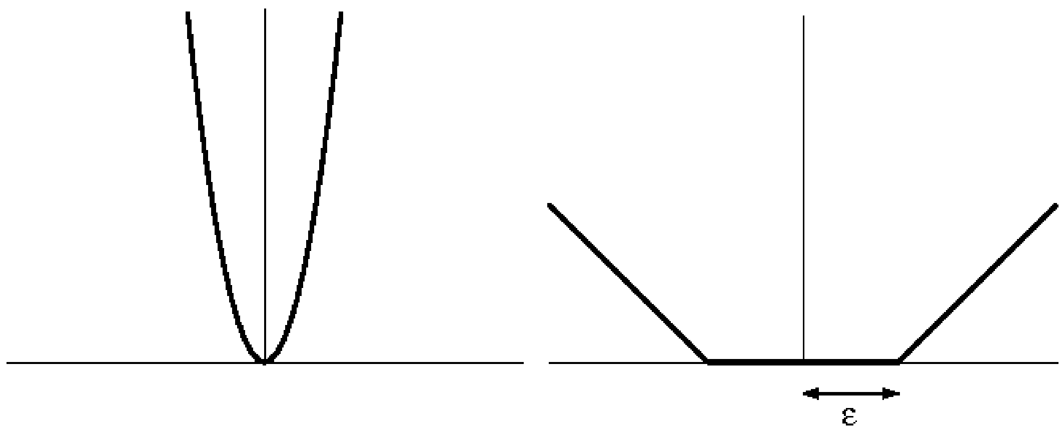
where d ranges over the elements of $\mathbf{D}$, the set of scales we are using.

The experiments compare the performance of the BPDN technique for correlation kernels using various numbers of scales: one scale ($\mathbf{D} = \{1.0\}$), two scales ($\mathbf{D} = \{0.5, 1.0\}$), and four scales ($\mathbf{D} = \{0.0, 0.5, 0.75, 1.0\}$). As before, we run the experiments on our set of 50 out-of-sample images of pedestrians. Figure 7a, which plots the average reconstruction error in L_2 against the number of basis functions used in the compression, seems to indicate that to achieve a certain error rate, fewer scales of basis functions are better. This is counter to our argument for using multiple scales of basis functions since we would expect that, with more scales to choose from, the minimization technique would be able to obtain a better approximation when choosing basis functions from this larger dictionary.

To explain this apparent inconsistency, Figure 7b plots reconstruction error against the L_1 *norm of the coefficients*, which is the measure of sparsity that BPDN minimizes. Here, the desired behavior of the one-, two-, and four-scale reconstructions is evident – for a given level of reconstruction error, starting with a multiscale dictionary affords a more sparse representation. What does this mean?

The true measure of sparsity is the L_0 norm of the coefficients, or the number of basis functions. Since this would lead to an Integer Programming problem which is computationally prohibitive for the number of basis functions we are using, the BPDN formulation approximates L_0 by L_1. These results offer empirical evidence that these norms are in fact very different and L_1 is not a good approximation of L_0.

8 Error Norms for Image Compression



(a) (b)

Figure 8: The two different error norms; (a) L_2 norm, (b) L_ϵ norm.

The techniques for basis selection that we present in this paper use fundamentally different criteria to represent signals, depending on what functional form the error term takes; PCA minimizes the traditional L_2 norm and SVM minimizes L_ϵ, an ϵ-insensitive norm (Pontil, et al., 1998), both plotted in Figure 8. While the vast majority of reports of image processing techniques ascribe to the use of the L_2 norm, it is not clear that this measure of error is the "best" for this particular domain. One important caveat: any pixel-based norm, in particular all L_p, is clearly not the "right" error metric to use since the human visual system takes into account higher order image structure; our discussion focuses on choosing the best norm when we are restricted to a "pixelwise" cost such as L_p or L_ϵ.

In the context of image reconstruction, the L_2 norm penalizes any perturbations from the true value, while the L_ϵ norm does not penalize values that are within ϵ of the true value, but linearly penalizes values lying outside of this region. The difference in these similarity measures is shown in Figure 9; Figure 9a has low L_2 error and high L_ϵ error, relative to 9c, while Figure 9c has high L_2 error and low L_ϵ error, relative to 9a; 9b is the true image. The deviations in Figure 9a seem to stand out more than those in 9c, but 9c has higher L_2 error.

How are we to reconcile this seeming inconsistency in what the traditional L_2 error tells us with what our brain tells us? It is well known that people cannot perceive differences in intensity that are very small (Schade, 1956; Campbell and Robson, 1968; Hess and Howell, 1977). In DeVore, et al. (1992), the authors argue that the L_1 error norm is a more accurate mathematical realization of the norm embedded in the human visual system than the L_2 norm. Fundamental to their hypothesis is the structure of the Contrast Sensitivity Threshold (CST) curve that captures a person's ability to distinguish an oscillating pattern of increasing frequency at different levels of contrast. Their argument determines the value of p for which the L_p norm best fits what the geometry of the CST curve implies; they find that $p = 1$ is the best approximation of the perceptual system's norm.

We can combine their results with the fact that at low contrasts in the middle frequencies of the CST curve it is nearly impossible to distinguish the different bands, implying the existence of

(a) (b) (c)

Figure 9: Examples of images with different types of errors; (a) low L_2 error, high L_ϵ error, relative to image (c); (b) true image; (c) high L_2 error, low L_ϵ error, relative to image (a).

some base threshold. This leads us to postulate that the L_ϵ norm may be a more perceptually accurate norm than L_1, since it encodes both the geometric constraints and threshold evident in the CST curve. In the absence of a psychophysical experiment that investigates this hypothesis, this conjecture is speculation, of course.

9 Conclusion

We have shown that the use of class-specific correlation-based kernels, when combined with the notion of sparsity, results in a powerful signal reconstruction technique. In a comparison to a traditional method of signal approximation, Principal Components Analysis, our approach achieves a more sparse representation for a given level of error.

For signal compression, the difference in performance between the techniques is not easily evaluated; when using different measures of error, we obtain a different "best" system. The choice of a system to use could depend on the characteristics of the different norms. The L_2 norm penalizes any difference in reconstruction. On the other hand, the L_ϵ norm does not penalize differences in the small ϵ-insensitive region around the true value, but linearly penalizes errors outside this region. One way of comparing the L_2, L_1, and L_ϵ norms could be to decide which is a more accurate description of psychophysical measures of similarity between images. Based on the arguments presented in Section 8 and the references cited therein, we postulate that the L_ϵ norm may be the norm we should use in image reconstruction, superresolution, and compression.

Our approach of using a dictionary of class-specific correlation kernels to obtain sparse representation of a signal leads to an interesting question: could this sparse representation that has been generated to *approximate* a signal be used to *classify* different signals? In other words, is the representation of pedestrians via sparse sets of correlation-based basis functions different enough from the representation of other objects (or all other objects), so that it can be used as a model for that class of objects? The representations we generate are derived through an argument that minimizes error for reconstructing the image. This, however, says nothing about the ability of that same representation to be used to differentiate images of different objects. Whether or not this can be done is an open question; Appendix C presents a preliminary discussion of this approach.

10 Acknowledgments

The authors would like to thank the following people for useful discussions and suggestions that helped improve the quality of this work: Sayan Mukherjee, Edgar Osuna, Massimiliano Pontil, and Ryan Rifkin.
The eigen-decomposition of our pedestrian class was generated using routines from *Numerical Recipes in C* (Press, et al., 1992). In our implementation of BPDN, as well as the associated decomposition algorithm (see Section B), we used the LSSOL quadratic programming package from Stanford Business Software, Inc. (Gill, et al., 1986). The cubic spline superresolution reconstruction in Section 5.1 was generated using MATLAB Version 5.2 (The MathWorks, Inc., 1998).

References

[1] M. Bazaraa, H. Sherali, and C. Shetty. *Nonlinear Programming: Theory and Algorithms.* John Wiley & Sons, 2nd edition, 1979.

[2] B. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifier. In *Proceedings of the Fifth Annual ACM Workshop on Computational Learning Theory*, pages 144–52. ACM, 1992.

[3] C.J.C. Burges. A Tutorial on Support Vector Machines for Pattern Recognition. In Usama Fayyad, editor, *Proceedings of Data Mining and Knowledge Discovery*, pages 1–43, 1998.

[4] F.W. Campbell and J.G. Robson. Application of Fourier Analysis to the Visibility of Gratings. *Jounral of Physiology*, 197:551–566, 1968.

[5] S. Chen, , D. Donoho, and M. Saunders. Atomic Decomposition by Basis Pursuit. Technical Report 479, Department of Statistics, Stanford University, May 1995.

[6] P.E. Gill, S.J. Hammarling, W. Murray, M.A. Saunders, and M.H. Wright. User's Guide for LSSOL (Version 1.0). Technical Report SOL 86-1, Stanford University, 1986.

[7] F. Girosi. An Equivalence Between Sparse Approximation and Support Vector Machines. A.I. Memo 1606, MIT Artificial Intelligence Laboratory, 1997. (available via the home-page: http://www.ai.mit.edu/people/girosi/).

[8] F. Girosi. An Equivalence Between Sparse Approximation and Support Vector Machines. *Neural Computation*, 1998. (in press).

[9] R.F. Hess and E.R. Howell. The Threshold Contrast Sensitivity Function in Strabismic Amblyopia: Evidence for a Two Type Classification. *Vision Research*, 17:1049–1055, 1977.

[10] The MathWorks Inc. *Using Matlab*. 1998.

[11] W. Karush. Minima of Functions of Several Variables with Inequalities as Side Conditions. Master's thesis, Department of Mathematics, University of Chicago, 1939.

[12] H.W. Kuhn and A.W. Tucker. Nonlinear Programming. In J. Neyman, editor, *Proceedings of 2nd Berkeley Symposium on Mathematical Statistics and Probability*, 1951.

[13] S. Mallat and Z. Zhang. Matching Pursuit in a time-frequency dictionary. *IEEE Transactions on Signal Processing*, 41:3397–3415, 1993.

[14] S.G. Mallat. A theory for multiresolution signal decomposition: The wavelet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(7):674–93, July 1989.

[15] Sr. O.H. Schade. Optical and Photoelectric Analog of the Eye. *Journal of the Optical Society of America*, 46(9):721–739, September 1956.

[16] M. Oren, C.P. Papageorgiou, P. Sinha, E. Osuna, and T. Poggio. Pedestrian Detection Using Wavelet Templates. In *Computer Vision and Pattern Recognition*, pages 193–99, 1997.

[17] E. Osuna, R. Freund, and F. Girosi. Support Vector Machines: Training and Applications. A.I. Memo 1602, MIT Artificial Intelligence Laboratory, 1997.

[18] C.P. Papageorgiou. Object and Pattern Detection in Video Sequences. Master's thesis, MIT, 1997.

[19] C.P. Papageorgiou, M. Oren, and T. Poggio. A General Framework for Object Detection. In *Proceedings of International Conference on Computer Vision*, 1998.

[20] P. S. Penev and J. J. Atick. Local Feature Analysis: A general statistical theory for object representation. *Neural Systems*, 7(3):477–500, 1996.

[21] T. Poggio and F. Girosi. A Sparse Representation for Function Approximation. *Neural Computation*, 1998. (in press).

[22] T. Poggio and F. Girosi. Notes on PCA, Regularization, Support Vector Machines and Sparsity. A.I. Memo (in press), MIT Artificial Intelligence Laboratory, 1998.

[23] M. Pontil, S. Mukherjee, and F. Girosi. An Interpretation of the $\epsilon-$Insensitivity Loss Function. A.I. Memo, MIT Artificial Intelligence Laboratory, 1998. (in press).

[24] W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery. *Numerical Recipes in C*. Cambridge University Press, 2 edition, 1992.

[25] L.L. Schumaker. *Spline Functions: Basic Theory*. John Wiley and Sons, New York, 1981.

[26] E.P. Simoncelli and W.T. Freeman. The steerable pyramid: A flexible architecture for multi-scale derivative computation. In *2nd Annual IEEE International Conference on Image Processing*. IEEE, October 1995. not sure if it belongs here.

[27] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer Verlag, 1995.

[28] G. Wahba. *Spline Models for Observational Data*. Series in Applied Mathematics, Vol. 59, SIAM, Philadelphia, 1990.

A The BPDN QP Formulation

The Basis Pursuit De-Noising formulation minimizes the following functional:

$$\|f(\mathbf{x}) - \sum_{i=1}^{N} c_i \phi_i(\mathbf{x})\|_{L_2}^2 + \lambda \|\mathbf{c}\|_{L_1} \tag{27}$$

To make the expansion of Equation 27 easier, we decompose $\mathbf{c}$ into its positive and negative coefficients:

$$\mathbf{c} = \mathbf{c}^+ - \mathbf{c}^- \tag{28}$$

where, to enforce the constraint that a coefficient is non-zero in at most one of the vectors, $\mathbf{c}^+$ or $\mathbf{c}^-$, we have:

$$\mathbf{c}^+, \mathbf{c}^- \geq \mathbf{0}$$

$$c_i^+ c_i^- = 0 \qquad \forall i = 1 \ldots N$$

This allows us to write the rewrite the sparsity term as:

$$\|\mathbf{c}\|_{L_1} = \mathbf{1}^T(\mathbf{c}^+ + \mathbf{c}^-) = \sum_{i=1}^{N}(c_i^+ + c_i^-).$$

We therefore expand Equation 27 as:

$$\|f(\mathbf{x})\|^2 - 2\sum_{i=1}^{N} c_i \langle f(\mathbf{x}), \phi_i(\mathbf{x})\rangle + \sum_{i=1}^{N}\sum_{j=1}^{N} c_i c_j \langle \phi_i(\mathbf{x}), \phi_j(\mathbf{x})\rangle + \lambda \mathbf{1}^T(\mathbf{c}^+ + \mathbf{c}^-) \tag{29}$$

Since $\|f(\mathbf{x})\|^2$ is a constant, it does not affect the minimization, so we have:

$$-2\sum_{i=1}^{N} c_i \langle f(\mathbf{x}), \phi_i(\mathbf{x})\rangle + \sum_{i=1}^{N}\sum_{j=1}^{N} c_i c_j \langle \phi_i(\mathbf{x}), \phi_j(\mathbf{x})\rangle + \lambda \mathbf{1}^T(\mathbf{c}^+ + \mathbf{c}^-) \tag{30}$$

Letting:

$$\begin{aligned} y_i &= \langle f(\mathbf{x}), \phi_i(\mathbf{x})\rangle \\ M_{ij} &= \langle \phi_i(\mathbf{x}), \phi_j(\mathbf{x})\rangle \end{aligned}$$

we get:

$$- 2\sum_{i=1}^{N} c_i y_i + \sum_{i=1}^{N}\sum_{j=1}^{N}(c_i^+ - c_i^-)(c_j^+ - c_j^-)\langle\phi_i(\mathbf{x}), \phi_j(\mathbf{x})\rangle + \lambda\mathbf{1}^T(\mathbf{c}^+ + \mathbf{c}^-) \tag{31}$$

Using the following definitions,

$$\begin{aligned}\mathbf{d} &= (\mathbf{c}^+, \mathbf{c}^-)\\ \mathbf{Y} &= (\mathbf{y}, -\mathbf{y})\end{aligned}$$

the first and last terms can be rewritten as:

$$\begin{aligned}&-2\mathbf{c}^T\mathbf{y} + \lambda\mathbf{1}^T\mathbf{c}\\ &= \mathbf{d}^T(\lambda\mathbf{1} - 2\mathbf{Y})\end{aligned}$$

so we have:

$$\sum_{i=1}^{N}\sum_{j=1}^{N}(c_i^+ - c_i^-)(c_j^+ - c_j^-)\langle\phi_i(\mathbf{x}), \phi_j(\mathbf{x})\rangle + \mathbf{d}^T(\lambda\mathbf{1} - 2\mathbf{Y}) \tag{32}$$

Taking:

$$\mathbf{H} = 2\begin{pmatrix} \mathbf{M} & -\mathbf{M} \\ -\mathbf{M} & \mathbf{M} \end{pmatrix}$$

the final form of this QP problem is

$$\text{minimize } \frac{1}{2}\mathbf{d}^T\mathbf{H}\mathbf{d} + \mathbf{d}^T(\lambda\mathbf{1} - 2\mathbf{Y}) \tag{33}$$

subject to the constraints:

$$\mathbf{d} \geq \mathbf{0} \tag{34}$$

We compute the $\mathbf{M}$ matrix by taking the inner products of different basis functions; the basis functions we use are the correlation kernels from Section 2. For notional simplicity, let $R(\cdot)$ refer to the correlation kernel with $d = 1.0$, $Q(\cdot)$ to the kernel with $d = 0.5$, and $P(\cdot)$ to the kernel with $d = 0.0$.

$$\begin{aligned}\int R(\mathbf{x}, \mathbf{x}_i)R(\mathbf{x}, \mathbf{x}_j)dx &= \left(\sum_k \lambda_k\phi_k(\mathbf{x})\phi_k(\mathbf{x}_i)\right)\left(\sum_\ell \lambda_\ell\phi_\ell(\mathbf{x})\phi_\ell(\mathbf{x}_\ell)\right)\\ &= \sum_k\sum_\ell \lambda_k\lambda_\ell\phi_k(\mathbf{x})\phi_k(\mathbf{x}_i)\underbrace{\langle\phi_k(\mathbf{x}), \phi_\ell(\mathbf{x})\rangle}_{\delta_{k\ell}}\\ &= \sum_k \lambda_k^2\phi_k(\mathbf{x}_i)\phi_k(\mathbf{x}_j)\end{aligned}$$

which corresponds to the correlation kernel with $d = 2.0$, ie.

$$\int R(\mathbf{x}, \mathbf{x}_i) R(\mathbf{x}, \mathbf{x}_j) d\mathbf{x} = R_{2.0}(\mathbf{x}_i, \mathbf{x}_j)$$

Similarly, we can show that for corresponding choices of basis functions, Q and P, we get:

$$\int Q(\mathbf{x}, \mathbf{x}_i) Q(\mathbf{x}, \mathbf{x}_j) d\mathbf{x} = R_{1.0}(\mathbf{x}_i, \mathbf{x}_j)$$
$$\int P(\mathbf{x}, \mathbf{x}_i) P(\mathbf{x}, \mathbf{x}_j) d\mathbf{x} = R_{0.0}(\mathbf{x}_i, \mathbf{x}_j)$$

Therefore, the matrix M does not need to be computed on the fly; we can simply store the correlation function of the signal and use this at run-time.

B QP Decomposition Algorithm

For the Basis Pursuit De-Noising approach to the sparsity problem, the size of the quadratic programming problem is directly related to the number of basis functions contained in our dictionary of features. The computational limitations come from the size of the matrix $\mathbf{H}$ in Equation 33; if there are n features in our dictionary, the size of the matrix will be $4n^2$. Even for dictionaries where n is on the order of $O(10^3)$, the amount of space this matrix takes up is immense. We would like to have both a system that uses a rich set of basis functions and one that is computationally tractable; for this we develop an active set method that decomposes the problem into smaller elements, under the expectation that most basis functions will not be included in the final solution.

The algorithm proceeds by first finding a feasible solution in a smaller problem and verifying optimality conditions in the original problem. We then check the optimality conditions for this point; if the solution is not optimal, the smaller problem is modified by substituting in elements that will help reduce the objective function. This process of finding a feasible solution in a smaller problem, checking the optimality of this point, and modifying the problem to push it towards an optimal point, is iterated until an optimal solution is found.

The details regarding the optimality conditions and the actual decomposition algorithm are presented in the rest of this section.

B.1 Optimality Conditions

In general terms, the minimization problem is formulated as follows:

$$\text{minimize } f(\mathbf{d}) \tag{35}$$

subject to the constraints:

$$\begin{aligned} g_1(\mathbf{d}) &\leq \mathbf{0} \\ g_2(\mathbf{d}) &\leq \mathbf{0} \\ &\vdots \\ g_m(\mathbf{d}) &\leq \mathbf{0} \end{aligned} \tag{36}$$

Finding an optimal solution to this problem entails a constrained search in parameter space ($\mathbf{d}$) to minimize the objective function in Equation 35 while maintaining the constraints in Equation 36. A point in space, $\mathbf{d}'$, that satisfies the constraints is called a feasible point. If $\mathbf{H}$ is positive definite, the objective function we are minimizing is strictly convex so a feasible point $\mathbf{d}'$ is an optimal solution if it satisfies a set of conditions called the Karush-Kuhn-Tucker (KKT) conditions (Karush, 1939; Kuhn and Tucker, 1951; Bazaraa, et al., 1979). For the general problem, the KKT conditions are, in addition to the primal feasibility (PF) condition, the following:

$$\begin{array}{lll l} \nabla f(\mathbf{d}) + \sum_{i=1}^{m} v_i \nabla g_i(\mathbf{d}) & = \mathbf{0} & & (DF) \\ v_i & \geq 0 & \forall i = 1, \ldots, m & (DF) \\ v_i g_i(\mathbf{d}) & = 0 & \forall i = 1, \ldots, m & (CS) \end{array} \tag{37}$$

where v_i are the Lagrange multipliers of the problem, DF indicates a dual feasibility condition, and CS indicates the complementary slackness condition.
The QP problem we address is:

$$\text{minimize } \frac{1}{2}\mathbf{d}^T \mathbf{H} \mathbf{d} + \mathbf{d}^T \mathbf{C} \tag{38}$$

subject to the constraints:

$$\begin{array}{ll} \mathbf{d} & \geq \mathbf{0} \\ \mathbf{d} & \leq \mathbf{u} \end{array} \tag{39}$$

which can be placed into the general form as:

$$\begin{array}{lll} -\mathbf{d} & \leq \mathbf{0} & (g_1) \\ \mathbf{d} - u\mathbf{1} & \leq \mathbf{0} & (g_2) \end{array} \tag{40}$$

The formulas for the KKT conditions for this problem are as follows:

$$\begin{array}{ll} \nabla f(\mathbf{d}) + \boldsymbol{\mu} \nabla g_1(\mathbf{d}) + \boldsymbol{\nu} \nabla g_2(\mathbf{d}) & = \mathbf{0} \\ \boldsymbol{\mu} g_1(\mathbf{d}) & = \mathbf{0} \\ \boldsymbol{\nu} g_2(\mathbf{d}) & = \mathbf{0} \\ \boldsymbol{\mu} & \geq \mathbf{0} \\ \boldsymbol{\nu} & \geq \mathbf{0} \end{array} \tag{41}$$

which yield:

$$\begin{array}{ll} [\mathbf{H}\mathbf{d} + \mathbf{C}]_i - \mu_i + \nu_i & = 0 \\ -\mu_i d_i & = 0 \\ \nu_i (d_i - u) & = 0 \\ \mu_i & \geq 0 \\ \nu_i & \geq 0 \\ & \quad \forall i = 1, \ldots, n \end{array} \tag{42}$$

Since in our case $\mathbf{H}$ is positive definite, the objective function we are minimizing is convex and, if the KKT conditions hold for a feasible point, this point is an optimal solution.

B.2 Decomposition Algorithm

For our particular problem, we are interested in obtaining a solution where the number of non-zero elements of $\mathbf{d}$ are small in comparison to the number of zero coefficients; this is exactly the sparsity criterion. The decomposition algorithm we develop will push the objective function down the gradient until a point is reached where the objective function can no longer be decreased.
To start developing the algorithm, we define an index set $\mathbf{I}$ on the variables $\mathbf{d}$ and then partition $\mathbf{I}$ into $[\mathbf{B}, \mathbf{N}]$, such that the optimality conditions are enforced only in the smaller QP problem defined over the variables in $\mathbf{B}$. The vector $\mathbf{d}$ is partitioned into $\mathbf{d_B}$ and $\mathbf{d_N}$, where $d_i = 0 \;\; \forall i \in \mathbf{N}$; our goal is to have $\mathbf{B}$ index the sparse nonzero coefficients.
Since we are looking for a sparse representation, $\mathbf{d_B}$ will have relatively few elements; minimizing this smaller objective function will be efficient. Since we set $d_i = 0 \;\; \forall i \in \mathbf{N}$, the value of the objective function we get by solving the smaller QP problem is equal to the value of the original objective function. For a formal proof showing that improving the cost function defined over the sub-problem strictly improves global cost function we are minimizing, see Osuna, et al., (1997).
After solving the smaller QP problem, we check the KKT conditions to see if this solution is optimal. The KKT conditions postulate that for a solution to be optimal, the following must hold, for each d_i:

$$[\mathbf{Hd} + \mathbf{C}]_i \begin{cases} \geq 0 & \text{if } d_i = 0 \\ = 0 & \text{if } 0 < d_i < u \\ \leq 0 & \text{if } d_i = u \end{cases} \tag{43}$$

This means that, for each coefficient $d_j \;\; j \in \mathbf{B}$, $[\mathbf{Hd}+\mathbf{C}]_j = 0$ must be true. If, for any $d_i \;\; i \in \mathbf{N}$, $[\mathbf{Hd} + \mathbf{C}]_i < 0$, then the addition of d_i to the working set would decrease the objective function – the current solution is not optimal. Hence, we exchange each $d_i \;\; i \in \mathbf{N}$ where $[\mathbf{Hd} + \mathbf{C}]_i < 0$ with a $d_j \;\; j \in \mathbf{B}$ where $d_j = 0$ (and d_j is therefore not contributing to minimizing the objective function); it is easy to see that this pivoting does not change the value of the objective function. The algorithm will move down the gradient until it reaches an optimal solution; the stopping criterion is that there are no more $d_i \;\; i \in \mathbf{N}$ with $[\mathbf{Hd}+\mathbf{C}]_i < 0$. From the KKT conditions, this means that $[\mathbf{Hd} + \mathbf{C}]_i < 0 \;\; \forall i \in \mathbf{B}$ and the solution is therefore optimal.
The decomposition algorithm is as follows:

1. Partition the variables into $d_{\mathbf{B}}$ and $d_{\mathbf{N}}$ such that d_i are fixed to $0 \;\; \forall i \in \mathbf{N}$.

2. Solve the smaller QP problem over $d_{\mathbf{B}}$; since $d_i = 0 \;\; i \in \mathbf{N}$, do not affect the value of the objective function.

3. While there is a $d_i \;\; i \in \mathbf{N}$ such that $[\mathbf{Hd} + \mathbf{C}]_i < 0$ (ie., the contribution of this variable will push down the objective function), we will pivot this with a $d_j = 0 \;\; j \in \mathbf{B}$ (i.e. d_j is not contributing to reducing the objective function). Go to (2) and repeat.

C Classification

The pattern classification problem is one where, instead of approximating a signal, we would like to decide to which class of patterns that signal belongs. For simplicity, let us say that we are interested in a classification problem where there are two classes, $\mathcal{C}_1$ and $\mathcal{C}_2$. We may be able to relate the distinct problems of regression and classification through our use of class-specific basis functions. Specifically, we would like to argue that the features of an object class $\mathcal{C}_1$ that are important for reconstructing elements of that class may be useful for differentiating elements of that class from elements of the other class $\mathcal{C}_2$. More formally, we would like to classify the image $f(\mathbf{x})$ but we also know the generalized correlation function $R(\mathbf{x}, \mathbf{y})$ of the set of similar images $f_\alpha(\mathbf{x})$, from which the correlation function was derived. We can follow the general approach of Penev and Atick (1996) where they use the sparsified kernels computed for regression for classification; in our case, we will use a SVM classifier.

C.1 Using the Regression Kernel R_d for Pattern Classification

Consider the problem of classification applied to images of dimensionality N; here, each real-valued pixel corresponds to one dimension. The goal is to learn a mapping g from points in $\mathcal{R}^N$ to a binary variable, $\mathcal{C}$, that indicates the possible classes. In general, this is a difficult task because the dimensionality of N is usually large. To make this tractable, we can use the notion of sparse representations to compress the "index" space $\mathcal{R}^N$ into a smaller space that accurately approximates the original space. As we have shown in this paper, this can be done using SVM regression or BPDN.

Let us assume that we have found the optimal sparse set of $R_d(\mathbf{x}, \mathbf{x}_i)$ for $i = 1, \ldots, N'$ ($N' << N$) over the set of images f_α. Thus:

$$f_\alpha(\mathbf{x}) = \sum_{i=1}^{N'} a_i^\alpha R(\mathbf{x}, \mathbf{x}_i) \tag{44}$$

where the $\mathbf{x}_i$ are not computed from the specific images; for instance, they could be generated by sparsifying the average image $E[f_\alpha]$; see remark later). Then we can estimate the coefficients $\mathbf{a}^\alpha$ for each image from

$$\mathbf{f}_\alpha = R\mathbf{a}^\alpha \tag{45}$$

($\mathbf{a}^\alpha = R^\dagger \mathbf{f}_\alpha$)[2]. The matrix $R^\dagger$ can be precomputed at the locations $\mathbf{x}_i$ given by the sparsification of the average image.

In many image classification problems there are two classes: the class of images we are interested in, and the class of all other images. The latter class will be associated with a correlation function which is translation invariant and rather "generic". It would be advantageous to use both kernels within the classifier but it is not clear what is the best way to do it.

[2]The coefficients computed in this way are not the correct ones from the point of view of SVM regression.