International Journal on Artificial Intelligence Tools
Vol. 16, No. 6 (2007) 967–979
© World Scientific Publishing Company

LARGE SCALE MULTIKERNEL RELEVANCE VECTOR MACHINE FOR OBJECT DETECTION

DIMITRIS TZIKAS*, ARISTIDIS LIKAS† and NIKOLAS GALATSANOS‡
Department of Computer Science, University of Ioannina
Ioannina, 45110, Greece
**tzikas@cs.uoi.gr*
†arly@cs.uoi.gr
‡galatsanos@cs.uoi.gr

The Relevance Vector Machine(RVM) is a widely accepted Bayesian model commonly used for regression and classification tasks. In this paper we propose a multikernel version of the RVM and present an alternative inference algorithm based on Fourier domain computation to solve this model for large scale problems, e.g. images. We then apply the proposed method to the object detection problem with promising results.

Keywords: Relevance vector machine; object detection; image analysis.

1. Introduction

The Relevance Vector Machine (RVM)[1] is a Bayesian treatment of the linear model given by:

$$y(\vec{x}) = \sum_{i=1}^{M} w_i \phi_i(\vec{x}) , \tag{1}$$

where $\{\phi_i(\vec{x})\}_{i=1}^{M}$ is a set of basis functions. Learning on such a model, is the process of estimating the weights $\{w_i\}_{i=1}^{M}$, using a training set $\{(\vec{x_n}, t_n)\}_{n=1}^{N}$. The weights are typically assigned those values that maximize the likelihood of the training set, however the training examples must be significantly more than the parameters in order to achieve good generalization performance. The RVM overcomes this limitation by following Bayesian principles and assuming prior knowledge for the model. Specifically, a suitable hierarchical prior distribution is assumed for the weights of the model, which has most probability mass concentrated in sparse solutions, meaning that it forces most of the weights to be assigned to zero values.[1] This results in pruning basis functions that are not sufficiently supported by the training data. There are several reasons to seek sparse solutions:

- Sparseness automatically adjusts the complexity of the model, thus very complex models may be considered.

- The basis functions that remain on the model provide information about which basis functions are relevant with the data. This may be useful in many applications.
- The output of sparse models is computed very efficiently, since only few basis functions are considered.

In a typical RVM there is one basis function centered at each training example, resulting in the following model:

$$y(\vec{x}) = \sum_{i=1}^{N} w_i \phi(\vec{x} - \vec{x_i}) \,, \tag{2}$$

As in most Bayesian models, inference in RVM is analytically intractable. The most well-known approximations are based on the expectation maximization (EM) algorithm[1] and the variational approximation.[2] Both approaches seem to provide similar results, but the first is generally preferred because it is computationally more efficient. However, time complexity is of order $O(N^3)$ and memory complexity of order $O(N^2)$, making inference on large training sets extremely difficult. The method based on the EM algorithm can be accelerated by incrementally adding basis function to an initially empty model,[3] improving time complexity to order $O(M^3)$, where M is the number of the basis functions that are included in the model. Since the model is sparse, M could be only a small fraction of the total number of basis functions N.

In this paper we use the RVM for modeling images. Unfortunately, the standard RVM training algorithms are too computationally demanding, even for small images. We notice that if the training points x_i lie on a uniform grid, the RVM model given in Eq. (2) can be rewritten as the convolution of the weight vector $\vec{w} = (w_1, \ldots, w_N)^T$ with a vector $\vec{\phi} = (\phi(\vec{x_1}), \ldots, \phi(\vec{x_N}))^T$, which consists of the basis function $\phi(\vec{x})$ evaluated at the training points $\vec{x_i}$. The RVM output can then be written as:

$$\vec{y} = \vec{\phi} * \vec{w} \,, \tag{3}$$

where $\vec{y} = (y(\vec{x_1}), \ldots, y(\vec{x_N}))^T$ is the output of the model evaluated at the training points. In section 2 we present in detail the RVM model and propose an alternative implementation of the EM-based algorithm.[1] Our implementation computes convolutions in the DFT domain, improving both time and memory requirements and allows to train RVM models on high resolution images, with reasonable computational costs. In addition we propose a multikernel RVM model, where more than one types of basis functions is allowed to be simultaneously used. The proposed implementation is evaluated in subsection 2.4.

We then use the proposed algorithm to solve the object detection problem, which is the problem of finding the location of an unknown number of occurrences of a given 'target' image in another given 'observed' image, under the presence of noise. The 'target' may appear significantly different in the observed image, as a result of being scaled, rotated, occluded by other objects, different illumination conditions and other effects.

The most common approaches to solve the object detection problem are variants of the matched filter, such as the phase-only[7] and the symmetric phase-only[8] matched filters. These are based on computing the correlation image between the 'observed' and 'target' images and then using a threshold to determine the locations where the 'target' object is present. Alternatively, the problem can be formulated as an image restoration problem, where the image to restore is considered as an impulse function at the location of the 'target' object. This technique allows many interesting background models to be considered, such as autoregressive models.[9] A different object detection approach, which has been successfully applied on face detection,[10] is to split the observed image in several regions and train a classifier with some features of the target 'object' in order to decide which regions contain the 'target' object.

In section 3 we propose a method for object detection, which is based on training a multikernel RVM model on the 'observed' image. The RVM model consists of two sets of basis functions: basis functions that are used to model the 'target' image and basis functions that are used to model the background. After training the model each 'target' basis function, whose corresponding weight is larger than a specified threshold, is considered as a detected 'target' object. Examples of the RVM based object detection algorithm and a comparison with the autoregressive impulse restoration[9] method are provided in section 4.

2. Large Scale Multikernel RVM

2.1. *RVM for image analysis*

The linear model in (1) is very efficient provided that suitable basis functions ϕ_i are selected and that there exist adequate training examples. Thus, finding a basis function set that describes the data well is an important problem, that is very difficult to solve. In this paper, we use several different types of basis functions $\phi_1, \ldots, \phi_M$ centered at each training point, resulting in the following model:

$$y(\vec{x}) = \sum_{m=1}^{M} \sum_{i=1}^{N} w_{mi} \phi_m(\vec{x} - \vec{x}_i) , \tag{4}$$

where M is the number of different basis function types. Although we use so many basis functions (and therefore parameters), overfitting should not be a concern, because of the sparseness of the RVM model.

In order to model a $N_i \times N_j$ image t using a RVM, we assume that the intensity $t(i,j)$ of the observed image at location (i,j) has been generated from the output $y(i,j)$ of the model at the same location, after addition of independent white noise $\epsilon(i,j)$:

$$t(i,j) = y(i,j) + \epsilon(i,j) , \tag{5}$$

$$\epsilon(x,y) \sim N(0, \beta^{-1})\,, \tag{6}$$

where β is the inverse variance of the noise.

Defining $\vec{t} = (t(1,1), \ldots, t(1,N_j), \ldots, t(N_i,N_j))^T$ to be a vector that contains the intensities of the image pixels in lexicographical order and similarly for $\vec{\epsilon} = (\epsilon(1,1), \ldots, \epsilon(1,N_j), \ldots, \epsilon(N_i,N_j))^T$, Eq. (5) can be rewritten as:

$$\vec{t} = \Phi\vec{w} + \vec{\epsilon} = \sum_{m=1}^{M} \Phi_m \vec{w}_m + \vec{\epsilon}, \tag{7}$$

where Φ is the $N \times (MN)$ design matrix, each column of which is a vector with the values of a basis function at all the training points. The design matrix can be partitioned as $\Phi = (\Phi_1, \ldots, \Phi_M)$, with $\Phi_m = (\vec{\phi_{m1}}, \ldots, \vec{\phi_{mN}})$ being the part of the design matrix corresponding to basis functions of type $\phi_m(\vec{x})$ and $\vec{\phi_{mi}} = (\phi_m(\vec{x_1} - \vec{x_i}), \ldots, \phi_m(\vec{x_N} - \vec{x_i}))^T$ being a vector consisting of the basis function $\phi_m(\vec{x} - \vec{x_i})$ evaluated at all the training points. The weight vector $\vec{w}$ can be similarly partitioned as $\vec{w} = (\vec{w_1}^T, \ldots, \vec{w_M}^T)^T$, with each $\vec{w_m} = (w_{m1}, \ldots, w_{mN})$ consisting of the weights corresponding to basis function $\phi_m(\vec{x})$.

The likelihood of the data set can then be written as:

$$p(\vec{t}|\vec{w}, \beta) = (2\pi)^{-N/2}\beta^{N/2} \exp\left\{-\frac{1}{2}\beta\|t - \Phi\vec{w}\|^2\right\}. \tag{8}$$

Given that the described model has M times more parameters than the training examples are, it is essential to seek a sparse solution. Under the Bayesian framework sparseness is obtained by assigning suitable prior distributions on the parameters. Specifically, independent Gaussian prior distributions with unknown variances are assigned on the weights $\vec{w}$:

$$p(\vec{w}) = \prod_{m=1}^{M}\prod_{i=1}^{N} p(w_{mi}) = \prod_{m=1}^{M}\prod_{i=1}^{N} N(0, \alpha_{mi}^{-1})\,, \tag{9}$$

where α_{mi} is a hyperparameter controlling the inverse variance of the corresponding weight w_{mi}. These hyperparameters are assumed unknown and Gamma hyperpriors are assigned to them. The inverse noise variance β may also be assumed unknown and similarly, a Gamma prior distribution is assigned to it:

$$p(\vec{\alpha}) = \prod_{m=1}^{M}\prod_{i=1}^{N} \Gamma(a,b)\,, \tag{10}$$

$$p(\beta) = \Gamma(c,d)\,, \tag{11}$$

where $\vec{\alpha} = (\alpha_{11}, \ldots, \alpha_{1N}, \ldots, \alpha_{MN})$.

Unfortunately, computation of the posterior distribution of the parameters is analytically intractable and an approximation has to be used. An effective approximation is to consider the posterior distribution of the weights treating the hyperparameters as known parameters and then optimize the hyperparameters.[1] The posterior weight distribution is:

$$p(\vec{w}|\vec{t},\vec{\alpha},\beta) = N(\vec{w}|\vec{\mu},\Sigma)\,, \tag{12}$$

with

$$\Sigma = (\beta\Phi^T\Phi + A)^{-1}\,, \tag{13}$$

$$\vec{\mu} = \beta\Sigma\Phi^T\vec{t}. \tag{14}$$

Then the hyperparameters are set to those values that maximize their posterior distribution given by:

$$p(\vec{\alpha},\beta|t) \propto p(t|\vec{\alpha},\beta)p(\vec{\alpha})p(\beta)\,. \tag{15}$$

The quantity $p(t|\vec{\alpha},\beta)$ is known as the marginal likelihood and is given by:

$$p(\vec{t}|\vec{\alpha},\beta) = (2\pi)^{-MN/2}|\Sigma|^{-1/2}\exp\left\{-\frac{1}{2}(\vec{w}-\vec{\mu})^T\Sigma^{-1}(\vec{w}-\vec{\mu})\right\}\,. \tag{16}$$

Differentiation of (16) leads to the following updates for the hyperparameters:

$$\alpha_{mi} = \frac{1-\alpha_{mi}\Sigma_{ii}}{\mu_{mi}^2}\,, \tag{17}$$

$$\beta = \frac{N-\sum_{i=1}^{N}(1-\alpha_i\Sigma_{ii})}{\|\vec{t}-\Phi\vec{\mu}\|^2}\,. \tag{18}$$

Equivalent updates can also be derived from an EM formulation, treating the weights as hidden variables.

The learning algorithm proceeds by iteratively computing the posterior statistics $\vec{\mu}$, Σ of the weights, given by (13) and (14) and then updating the hyperparameters using (17) and (18). Computation of Σ involves inverting a N–by–N matrix which is an $O(N^3)$ procedure, where N is the initial number of basis functions. During the training process, many of the hyperparameters are set to infinite values and the corresponding basis functions can be punned, allowing computation of the posterior statistics in $O(M^3)$ time, where M is the number of functions that remain in the model. This results in significant speed-up of the latter iterations of the algorithm, however in the first iteration all the basis functions have to be considered and the overall complexity is still $O(N^3)$.

An alternative algorithm[3] starts by assuming an empty model and incrementally adds basis functions in each iteration. Computing the posterior statistics require $O(M^3)$ time, since the full model never has to be considered. However, only one basis function may be considered at each iteration and if this is chosen at random, then the algorithm requires much more iterations to reach convergence. Alternatively, the most significant basis function may be selected at each iteration, but this selection is computationally expensive. Overall, this algorithm is an important improvement, but it still cannot be used for large scale problems, such as modeling images. In this paper we propose an RVM implementation based on DFT computations, that successfully resolves the problem of computational complexity.

2.2. *RVM sparseness*

Since the properties of any Bayesian model are based on the prior distributions that are used, sparseness in the RVM is achieved by assigning to the weights a suitable prior distribution. This is a hierarchical prior, consisting of a Gaussian distribution for the weights $\vec{w}$ (eq.(9)), and a Gamma distribution for the hyperparameters $\vec{\alpha}$ (eq.(10)) that define the inverse variance of the distribution of the weights. It is important that there is a separate hyperparameter that controls the variance of each weight, since this makes the hierarchical prior equivalent to a Student-t prior distribution with $\frac{a}{b}$ variance and $2a$ degrees of freedom:

$$p(\vec{w}) = \int \prod_{i=1}^{M} \left(p(w_i|\alpha_i)p(\alpha_i)\right) d\vec{\alpha} = \prod_{i=1}^{M} \left(\int p(w_i|\alpha_i)p(\alpha_i)d\alpha_i \right)$$

$$= \prod_{i=1}^{M} \left(\int N(w_i|0,\alpha_i)\Gamma(\alpha_i|a,b)d\alpha_i \right) = \prod_{i=1}^{M} St\left(w_i|0,\frac{a}{b}I,2a\right) . \quad (19)$$

It is well known that most probability mass of the Student-t distribution is concentrated on the origin of the axis of definition and along the axis, which explains why this prior distribution produces sparse solutions. Actually, the basis functions that are used in the final model, are those basis functions that are more relevant with the data. For this reason, this prior is also known as automatic relevance determination (ARD) prior,[5] and the points that correspond to basis functions that remain in the final model are called relevance vectors (RV).

2.3. *Multikernel RVM implementation in the DFT domain*

It can be observed that if the training points are the pixels of an image, or generally uniform samples of a signal, then the RVM given by (4) can be written using a convolution as:

$$\vec{y} = \sum_{m=1}^{M} \vec{\phi_m} * \vec{w_m} . \quad (20)$$

Equation (7) still holds, with the additional property that matrices Φ_m are circulant. This is an important property, implying that the product $\Phi_m\vec{w}_m$ is a convolution which can be efficiently computed in the DFT domain by multiplying the DFT $\mathcal{F}_m$ and $\mathcal{W}$ of the basis function ϕ_m and the weight vector w.

$$\mathcal{T}_i = \sum_{m=1}^{M} \mathcal{F}_{mi}\mathcal{W}_{mi} , \quad (21)$$

where $\mathcal{T}$ is the DFT of the observations vector $\vec{t}$. This observation allows computation of the output of the model without using the complete design matrix but only one basis vector, improving memory complexity from $O(N^2)$ to $O(N)$ and time complexity from $O(N^2)$ to $O(N \log N)$.

The posterior statistics of the weights μ and Σ can also be computed in the DFT domain, benefitting from the same advantages. Beginning with (14), the posterior mean of the weights can be found by solving the equation:

$$\Sigma^{-1}\vec{\mu} = \beta\Phi^T\vec{t}, \tag{22}$$

$$(\beta\Phi^T\Phi + A)\vec{\mu} = \beta\Phi^T\vec{t}. \tag{23}$$

Instead of analytically inverting the matrix $\beta\Phi^T\Phi + A$, which is computationally expensive and requires inversion of the large design matrix Φ, we solve equation (23) by using the conjugate gradient method[6] to minimize the following quadratic function:

$$\vec{\mu}^* = \underset{\mu}{argmin}(\vec{\mu}^T(\beta\Phi^T\Phi + A)\vec{\mu} - \vec{\mu}^T\beta\vec{\Phi}^T\vec{t})\,. \tag{24}$$

The quantities $\beta\Phi^T\Phi\vec{\mu}$ and $\beta\Phi^T\vec{t}$ can be easily computed in the DFT domain since Φ is circulant, while computation of $A\vec{\mu}$ is straightforward since A is diagonal. In the ideal case, the conjugate gradient method is guaranteed to find the exact minimum after N iterations. In practice, a very good estimate can be obtained in only a few iterations.

Unfortunately, in order to compute the posterior weight covariance we have to invert the matrix $\beta\Phi^T\Phi + A$, which is a computational burden. Instead, we notice that we only need to compute the diagonal of Σ and consider two possible approximations.

The simplest approximation is to consider only the main diagonal of the matrix $\beta\Phi^T\Phi + A$, and compute Σ_{ii} as:

$$\Sigma_{ii} = (\beta\|\phi\|^2 + \alpha_i)^{-1}\,, \tag{25}$$

with $\phi = (\phi_{11}^T, \ldots, \phi_{1M}^T)^T$. Although this approximation is not valid in general, it has been proved very effective in the experiments, because the matrix A has generally very large values and is the dominant term in $\beta\Phi^T\Phi + A$.

An alternative approximation that has been considered is to approximate the matrix $\beta\Phi^T\Phi + A$ with a circulant matrix and compute Σ_{ii} as:

$$\Sigma_{ii} = (\beta\Phi^T\Phi + \alpha_i I)^{-1} = \frac{1}{N}\sum_{j=1}^{M}(\beta\mathcal{F}_j^2 + \alpha_i)^{-1}\,. \tag{26}$$

Notice, that a different (circulant) approximating matrix has to be inverted for the computation of each element of the diagonal of Σ. For this reason, this approximation requires more computations than the first and may be impractical for large images.

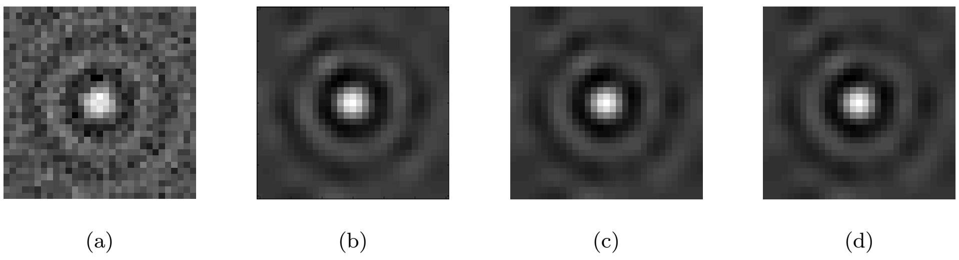


Fig. 1. (a) An artificially generated image with added noise. Estimates of (b) the RVM algorithm and the DFT-RVM algorithm using (c) the diagonal approximation of Eq. (25) and (d) the circulant approximation of Eq. (26).

Table 1. Mean square error of the typical RVM algorithm and the DFT based algorithm with the two approximations for several choices of the kernel width w. Inside parenthesis is the number of relevance vectors for each case.

Algorithm	w = 1	w = 2	w = 4	w = 8
RVM	0.055(229)	0.038(84)	0.040(45)	0.052(70)
DFT-RVM	0.055(249)	0.039(120)	0.048(49)	0.111(12)
DFT-RVM(2)	0.058(234)	0.041(105)	0.077(165)	0.111(190)

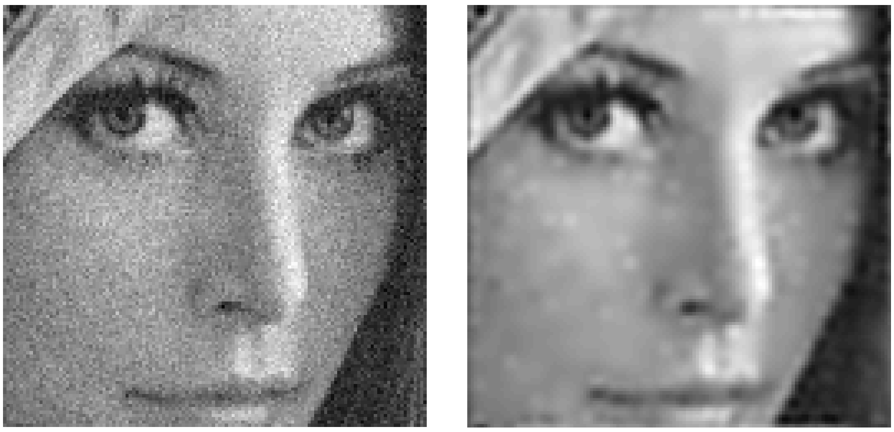
2.4. *Evaluation of the proposed modification*

In order to verify the validity and evaluate the performance of the proposed DFT-based implementation we sampled uniformly the function:

$$t(x,y) = \frac{\sin(\|x+y\|)}{\|x+y\|}, \tag{27}$$

to generate a 30×30 image shown in Fig. 1. We then added white Gaussian noise of variance 0.1 and applied both the typical and the DFT-based algorithm to estimate the parameters of an RVM model, which was then evaluated at each pixel location to produce an estimate of the original image t. Figure 1 shows the estimates obtained using the typical RVM algorithm and the DFT-based algorithm with the two different approximations respectively. Averages over 10 noise realizations of the mean squared error (MSE) of each estimate and the number of relevance vectors are shown in Table 1 for many different widths w of the kernel. We notice that the first (diagonal) approximation typically gives better results than the second (circulant) approximation and it also requires less computations. Also notice that the approximation gives excellent results when the size of the kernel is small, because the matrix Σ is almost diagonal.

Unfortunately, we cannot compare the algorithms for larger images because we cannot apply the typical RVM algorithm on larger datasets. However, we demonstrate the effectiveness of the proposed algorithm on large scale regression problems, by training an RVM model with Gaussian kernels of sizes $w_1 = 2$, $w_2 = 4$ and $w_3 = 8$ on a 256×256 image. The estimated image, shown in Fig. 2, is improved



(a) (b)

Fig. 2. (a) An 128 × 128 image with added gaussian noise. (b) Estimate of the DFT-RVM algorithm using gaussian kernels with variance 2, 4 and 8.

with respect to the initial noisy image, having $ISNR = 2.2$, where $ISNR$ is defined as $ISNR = 10 \log \left(\|f - g\|^2 / \|f - \hat{f}\|^2 \right)$ and is a measure of the improvement in quality of the estimated image with respect to the initial image.

3. Object Detection Using the RVM Model

In the rest of this paper, we present an alternative method for object detection, which is based on training a multikernel RVM model on the 'observed' image. The RVM model consists of two sets of basis functions: basis functions that are used to model the 'target' image and basis functions that are used to model the background. After training the model, each 'target' basis function that remains in the model can be considered as a detected 'target' object. However, if the background basis functions are not flexible enough, 'target' functions may also be used to model areas of the background. Thus, we should consider only 'target' basis functions whose corresponding weight is larger than a specified threshold.

We denote by $\vec{t} = (t(1,1), \ldots, t(1, N_j), \ldots, t(N_i, N_j))^T$ a vector consisting of the intensity values of the pixels of the 'observed' image in lexicographical order. We model this image using the following RVM model:

$$\vec{t} = \sum_{i=1}^{N} w_{ti} \phi_t(\vec{x} - \vec{x_i}) + \sum_{i=1}^{N} w_{bi} \phi_b(\vec{x} - \vec{x_i}) + \vec{\epsilon}, \tag{28}$$

where ϕ_t is the 'target' basis function which is a vector consisting of the intensity values of the pixels of the 'target' image, and ϕ_b is the background basis function, which we selected to be a Gaussian function. After training the RVM model we obtain the vectors $\vec{\mu_t}$ and $\vec{\mu_b}$, which are the posterior weight mean for the kernel and background weights respectively. Ideally, 'target' kernel functions would only be used to model occurrences of the ‘target’ object. However, because the background basis functions are often not flexible enough to model the background accurately, some ‘target’ basis functions may have been used to model the background as well. In order to decide which ‘target’ basis functions actually correspond to ‘target’ occurrences, the posterior ‘target’ weight means are thresholded, and only those that exceed a specified threshold are considered significant:

$$\text{Target exists at location } i \Leftrightarrow |\mu_{ti}| > T\,. \tag{29}$$

Choosing a low threshold may generate false alarms, indicating that the object is present in locations where it actually does not exist. On the other hand, choosing a high threshold may result in failing to detect an existing object. There is no unique optimal value for the threshold, but instead it should be chosen depending on the characteristics of the application.

The Support Vector Machine (SVM)[11] is another sparse learning method that has been used in regression problems. Although it is usually much more computationally efficient than the RVM approach,[12] it requires that the basis functions are valid kernel functions. This limitation, prohibits using SVMs with the proposed object detection method, since the basis functions that we use are the ‘target’ image and are not valid kernel functions.

4. Numerical Experiments

In this section we present experiments that demonstrate the improved performance of the DFT-RVM algorithm compared to autoregressive impulse restoration (ARIR), which is a state-of-the-art method, found to be superior to most existing object detection methods.[9] We first demonstrate two examples where the ‘observed’ images have been constructed by adding the ‘target’ object to a background image and then adding white Gaussian noise. Images consisting of the values of the kernel weights computed with the DFT-RVM algorithm are shown in Fig. 3 and compared with the output of the ARIR method. Notice that because of the RVM sparseness property, the output of the algorithm is zero at most locations where there is no target object. This property of the DFT-RVM detection method, is the main reason for the improved detection performance, which will be more thoroughly evaluated later.

When evaluating a detection algorithm it is important to consider the detection probability P_D, which is the probability that an existing ‘target’ is detected and the probability of false alarm P_{FA}, which is the probability that a ‘target’ is incorrectly detected. Any of these probabilities can be set to an arbitrary value by selecting an appropriate value for the threshold T. The receiver operating characteristics (ROC) curve is a plot of the probability of detection P_D versus the probability of false alarm P_{FA} that provides a comprehensive way to demonstrate the performance of a detection algorithm. However, the ROC curve is not suitable for evaluating object detection algorithms because it only considers if an algorithm has detected

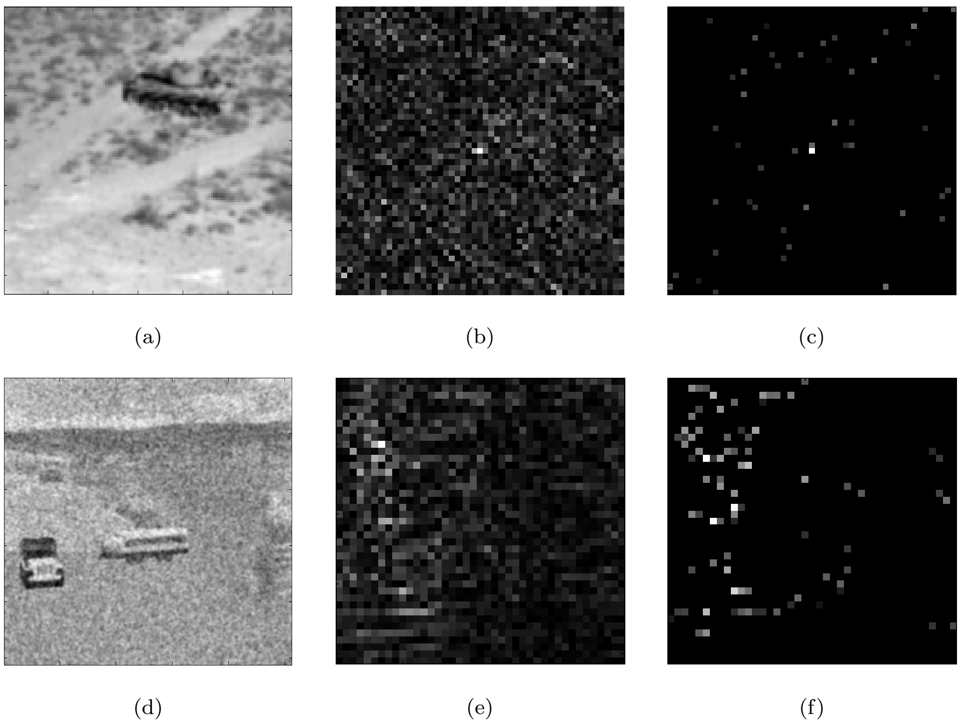


Fig. 3. Two object detection examples. (a) and (d) are the 'observed' images, (b) and (e) are the results of the ARIR algorithm and (c) and (f) are the results of the DFT-RVM algorithm. The target object is the tank in image (a) and the jeep in image (d). In the results, only a small area around the target is shown. In all cases, the the output of both algorithms is maximum at the location of the target. However, at all other locations, where there is no target and the output should ideally be zero, DFT-RVM outperforms the ARIR algorithm, since its output is zero at most locations.

an object or not; it does not consider if the object was detected in the correct location. Instead, we can use the localized ROC (LROC) curve which is a plot of the probability of detection and correct localization P_{DL} versus the probability of false alarm and considers also the location where a 'target' has been detected.

In order to evaluate the performance of the algorithm, we created 50 'observed' images by adding a 'target' image to a random location of the background image, and another 50 'observed' images without the 'target' object. White Gaussian noise of variance $\sigma^2 = 20$ was then added to each 'observed' image, that corresponds to signal to noise ratio $22dB$. The DFT-RVM algorithm was then used to estimate the parameters of an RVM model with a 'target' kernel and a Gaussian background kernel for each 'observed' image, generating 100 kernel weight images. The background basis functions were Gaussian functions of the form $\phi_i(\vec{x}) = \exp(-\frac{1}{r^2}\|\vec{x}-\vec{x_i}\|^2)$ with the width parameter set to $r = 6$. The kernel weight images were then thresholded for many different threshold values and estimates of the probabilities P_{DL} and P_{FA} were computed for each threshold value. Similar experiments were also per-

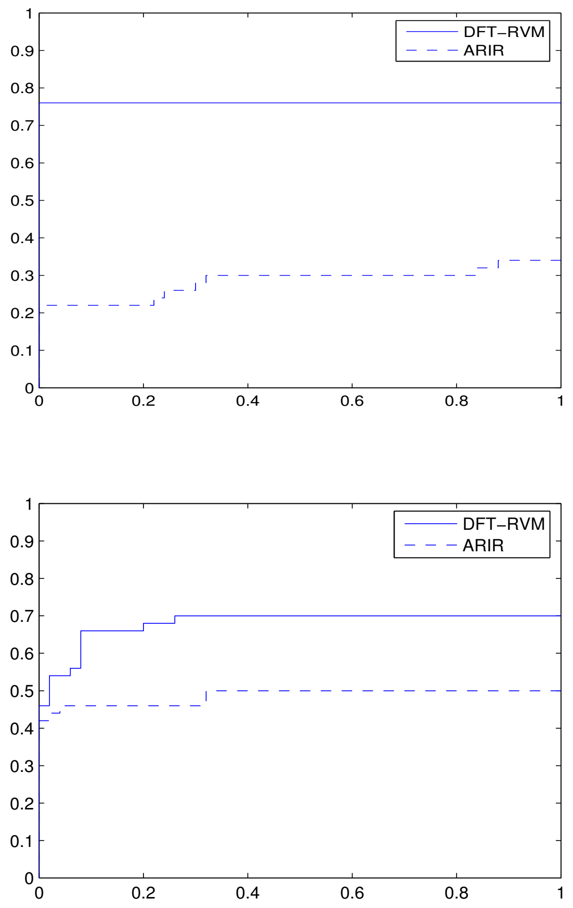


Fig. 4. LROC curves of the ARIR and DFT-RVM algorithms for the two detection problems shown in Fig. 3.

formed for the ARIR algorithm and an LROC curve was plotted for each algorithm. Figure 4 shows the LROC curve of each algorithm for the two cases of background and target images shown in Fig. 3. It can be observed that the area under the LROC curve, which is a common measure of the performance of a detection algorithm, is significantly larger for the DFT-RVM algorithm. Another important observation is that the LROC curve is high for small values of P_{FA}, since usually the threshold is chosen so that only a small fraction of false detections is allowed.

5. Conclusions

We have proposed an approximate but accelerated inference method for training the RVM model on large scale images, based on fast computation of the posterior statistics in the Fourier domain. Experiments on images demonstrate that the proposed approximation allows inference on large scale images, where the typical RVM algorithm is too computationally demanding to run. We then presented an application of the method to the object detection problem. Experimental results indicate that this approach is more robust than existing methods. Furthermore, the proposed technique can be extended to solve the rotation and scaling invariant object detection problem, by optimizing the model with respect to rotation and scaling of the basis functions.

Acknowledgments

This research was co-funded by the European Union in the framework of the program “Pythagoras II” of the “Operational Program for Education and Initial Vocational Training” of the 3rd Community Support Framework of the Hellenic Ministry of Education.

References

1. M. E. Tipping, “Sparse Bayesian learning and the relevance vector machine”, *Journal of Machine Learning Research*, 1:211–244, 2001.
2. A. C. Faul and M. E. Tipping, “A Variational Approach to Robust Regression”, ICANN 2001: 95–102.
3. M. E. Tipping and A. C. Faul, “Fast marginal likelihood maximisation for sparse Bayesian models”, In *Proceedings of Artificial Intelligence and Statistics ’03.*
4. M. Girolami and S. Rogers, “Hierarchical Bayesian Models for Kernel Learning”, ICML 2005.
5. D. J. C. MacKay, “Bayesian methods for backpropagation networks”, In *Models of Neural Networks III*, editors E. Domany, J. L. van Hemmen, and K. Schulten, Chapter 6, pages 211–254. Springer, 1994.
6. J. R. Shewchuk, “An Introduction to the Conjugate Gradient Method Without the Agonizing Pain”, http://www.cs.cmu.edu/∼quake-papers/painless-conjugate-gradient.ps
7. J. L. Horner and P. D. Gianino, “Phase-only matched filtering”, *Applied Optics*, 23(6), 812–816, 1984.
8. Q. Chen, M. Defrise and F. Decorninck, “Symmetric phase-only matched filtering of Fourier-Mellin transforms for image registration and recognition”, *Pattern Recognition and Machine Intelligence*, 12(12), 1156–1198, 1994.
9. A. Abu-Naser, N. P. Galatsanos, M. N. Wernick and D. Shonfeld, “Object recognition based on impulse restoration using the expectation-maximization algorithm”, *Journal of the Optical Society of America*, Vol. 15, No. 9, 2327–2340, September 1998.
10. P. Viola and M. Jones, “Rapid object detection using a boosted cascade of simple features”, IEEE CVPR, 2001.
11. C. Cortes and V. Vapnik, “Support vector networks”, Machine Learning, 20, 1995.
12. Wei Miao Yu, Tie Hua Du and Kah Bin Lim, “Comparison of the support vector machine and relevant vector machine in regression and classification problems”, *Proc. 8th International Conference on Control, Automation, Robotics and Vision*, 6–9 December 2004, KunMing, China.