Data Driven Frequency Mapping for Computationally Scalable Object Detection

Fatih Porikli
Mitsubishi Electric Research Laboratories

Huseyin Ozkan
Boston University

Abstract

Nonlinear kernel Support Vector Machines achieve better generalizations, yet their training and evaluation speeds are prohibitively slow for real-time object detection tasks where the number of data points in training and the number of hypotheses to be tested in evaluation are in the order of millions. To accelerate the training and particularly testing of such nonlinear kernel machines, we map the input data onto a low-dimensional spectral (Fourier) feature space using a cosine transform, design a kernel that approximates the classification objective in a supervised setting, and apply a fast linear classifier instead of the conventional radial basis functions. We present a data driven hypotheses generation technique and a LogistBoost feature selection. Our experimental results demonstrate the computational improvements 20∼100× while maintaining a high classification accuracy in comparison to SVM linear and radial kernel basis function classifiers.

1. Introduction

Discriminative kernel machines are found their way into many classification tasks because they can approximate any decision boundary arbitrarily well with an efficient generalization capability. Among those, Support Vector Machines (SVM) implicitly map the data $\mathbf{x} \in \mathcal{R}^d$ into a dot product space F, called as the feature space, via usually a nonlinear lifting $\phi : \mathcal{R}^d \rightarrow \mathcal{F},\ \mathbf{x} \mapsto \phi(\mathbf{x})$. Although $\mathcal{F}$ can be high-dimensional (infinite for certain liftings such as Gaussian radial basis functions), it is not necessary to explicitly work in that space. Besides, direct access to an infinite dimensional Hilbert space may not be possible. Instead of mapping data via ϕ and computing inner products, we can do it in one operation, leaving the lifting completely implicit, thus, all we need to know is how to compute the modified inner product, so called as kernel $k(\mathbf{x}, \mathbf{y})$.

Mercer's theorem enables characterizing kernels without the lifting function. A symmetric function $k(\mathbf{x}, \mathbf{y})$ can be expressed as an inner product $k(\mathbf{x}, \mathbf{y}) = \langle \phi(\mathbf{x}), \phi(\mathbf{y}) \rangle$ for some ϕ if and only if $k(\mathbf{x}, \mathbf{y})$ is positive semidefinite, that is

$$\int k(\mathbf{x}, \mathbf{y}) g(\mathbf{x}) g(\mathbf{y}) dx dy \geq 0 \quad \forall g. \tag{1}$$

There exists a class of kernels, which can be shown to compute the dot products in associated feature spaces. In other words, any positive definite function $k(\mathbf{x}, \mathbf{y})$ with $\mathbf{x}, \mathbf{y} \in \mathcal{R}^d$ defines an inner product and a lifting ϕ so that the inner product between lifted data points can be computed as $\langle \phi(\mathbf{x}), \phi(\mathbf{y}) \rangle = k(\mathbf{x}, \mathbf{y})$, thus all inner product computations in $\mathcal{F}$ are reduced to the evaluation of the kernel.

SVM based classifiers construct a separating hyperplane in $\mathcal{F}$, which maximizes the margin between the two datasets while minimizing the classification error. It can be shown that using Hinge loss as penalty to errors and for some positive value of a soft margin cost parameter C determining the trade-off between margin maximization and training error minimization, the resulting training problem consists of computing

$$\max_{\alpha} \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j) \tag{2}$$

subject to $0 \leq \alpha_i \leq C$ and $\sum \alpha_i y_i = 0$ where y_i are the labels corresponding to the data points $\mathbf{x}_i$. The classification solution as a result of the above optimization is given as

$$H(\mathbf{x}) = sgn \left(\sum_{i=1}^{l} \alpha_i y_i k(\mathbf{x}, \mathbf{x}_i) \right) \tag{3}$$

and the data points with $\alpha_i > 0$ are called as *support vectors*. One advantage is that the algorithm accesses the data only through evaluations of $k(\mathbf{x}_i, \mathbf{x}_j)$.

One important drawback of SVM is that whenever one wants to train it with a nonlinear kernel, the algorithm scales very poorly with the size of the training data. That is, when the size of the training data is in the order of hundreds of thousands, training phase of the algorithm can take days. To extend learning with kernel machines to large datasets, decomposition methods are often employed. These methods iteratively update a subset of the kernel machine coefficients using coordinate ascent until Karush-Kuhn-Tucker conditions are satisfied within a tolerance factor [1].

978-1-4577-0845-9/11/$26.00

It is shown that linear machines can be trained quickly on large datasets when the dimensionality of the data is small [2, 3, 4]. One way to take advantage of these linear training algorithms for training nonlinear machines is to approximately factor the kernel (Gram matrix consisting of kernel function applied to all pairs of data points) and to treat the columns of the factor matrix as features in a linear machine [5]. The evaluation of the kernel function can also be accelerated using linear random projections by discarding individual entries [6] while obtaining sparse approximations of the true Gram matrix, or entire rows [7, 8] of the Gram matrix while preserving the separability of the data. Alternatively, Taylor approximation methods have been proposed [9]. While these approaches appear to be effective on low-dimensional problems, their results degrades exponentially with the dimensionality of the dataset.

Second important drawback is the computational load of the test phase. For linear machines, testing a given data point is very efficient as it only requires computing one inner product. On the other hand, for nonlinear machines one must apply as many kernel evaluations as the number of the support vectors to compute the projection of the data point onto the separating hyperplane normal. Fast nearest neighbor lookup with kd-trees has been used to approximate multiplication with the Gram matrix for embedding such linear assignment problems [10]. An explicit mapping the data to a low-dimensional Euclidean inner product space using a randomized feature map for approximating shift invariant kernels is also proposed in [11]. The randomized features give an access to fast learning algorithms and also provide a way to quickly evaluate the kernel machine. On the other hand, the proposed random selection scheme is highly sensitive to the sampling, thus the accuracy of the resulting linear learning algorithm varies at each newly added dimension. Besides, it does not take advantage of the available class membership information for classification.

Although the existing works so far can linearize the kernel to speed up the test and training phases, they fail to exploit the class labels. Mostly, the linearization is blind and independent of the context and data prior.

Here, we present a novel algorithm that drastically speeds up testing of nonlinear kernel machines. We factor the kernel function itself, however this factorization is conveniently data driven, and allows us to convert the training and evaluation of a kernel machine into the corresponding operations of a linear machine by mapping data into a relatively low-dimensional feature space that is determined by the distributions of the binary class labels. To our knowledge, this is the first time a data driven representation of a nonlinear kernel into a frequency space is proposed for significantly bringing down the computational load.

We first generate a set of Fourier features that are rich enough to approximate the separating boundary between two classes of data points and the continuous, shift-invariant kernel that we implicitly impose. Our intuition is that a data driven low-dimensional mapping through a set of Fourier features will enable better generalization (robustness to variations) of the classifier while efficiently approximating the correct decision boundary at the same time.

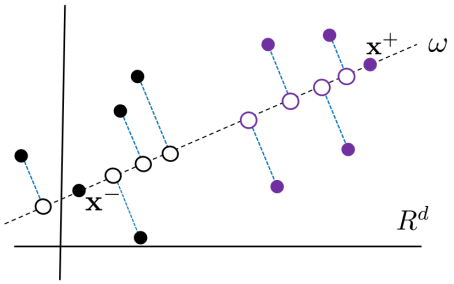

Figure 1. Mapping into Fourier space on an ω vector while considering the separability of the data points.

Instead of relying on the lifting provided by the kernel trick, we explicitly map the data to a low-dimensional Euclidean inner product space using a data driven feature map $\mathbf{z} : \mathcal{R}^d \rightarrow \mathcal{R}^m$ so that the inner product between a pair of transformed points approximates their kernel evaluation $k(\mathbf{x}, \mathbf{y}) = \langle \phi(\mathbf{x}), \phi(\mathbf{y}) \rangle \approx \mathbf{z}(\mathbf{x})^T \mathbf{z}(\mathbf{y})$. Unlike the kernel's lifting ϕ, $\mathbf{z}$ is low-dimensional. Thus, we can simply transform the input with frequency mapping onto $\mathbf{z}$, and then apply linear methods to approximate the decision boundary established through a nonlinear kernel. Fourier feature mapping is illustrated in Fig. 1.

In training, we employ LogitBoost [12], which is a logistic regression on the hyperplane normal, to select features from this set while directly optimizing the classification accuracy as opposed to targeting kernel function approximation. We start with the set of hypotheses, i.e. Fourier features, that are constructed from the labeled training data. Then for each hypothesis, we map the data onto a vector. We apply weighted least squares fitting of the labels to the mapped data and compute a regression error. We select the hypothesis that gives the minimum regression error. After adjusting the weights, we select the next hypothesis.

This way, we collect only useful Fourier features to train a linear machine. In comparison to [11] where features are collected totally random and independently, we take advantage of the available training data and the corresponding labels in a supervised setting.

By approximating the nonlinear separating boundary as an inner product of Fourier transformed features and eliminating non-descriptive features by feature selection, these data driven feature maps give us access to extremely fast algorithms to quickly evaluate the classifier. For example, with the kernel trick, evaluating the radial basis function support vector machine at a test point x requires $O(Nd)$

operations, where N is the number of training points, to compute and retain much of the dataset unless the boundary is simple and sparse. This is often unacceptable for large datasets. On the other hand, after learning a hyperplane ω, a linear machine can be evaluated by simply computing $f(\mathbf{x}) = \omega^T \mathbf{z}(\mathbf{x})$, which requires only $O(m(d+1))$ operations and storage, where m is the number of the selected features. Note that, for most problems $m << d < N$.

Our work can also be considered as kernel design where we naturally construct a data driven kernel as opposed to approximating the kernel. Therefore, the classification accuracy of our algorithm is not upper bounded by a fixed kernel function. In fact, we obtain better classification results in some datasets in addition to significantly decreasing the computational load. Our experiments show that these frequency mapped features, combined with very simple linear learning techniques, compete favorably in speed and accuracy with the state-of-the-art kernel-based classification algorithms.

2. Fourier Features

Bochner's theorem [13] from harmonic analysis on groups characterizes the Fourier transform of a positive finite measure. Given a positive finite Borel measure μ on the real line $\mathbb{L}$, the Fourier transform $f(\omega)$ of μ is the continuous function

$$f(\omega) = \int_{\mathbb{L}} e^{-j\omega x} d\mu(x) \tag{4}$$

The function $f(\omega)$ is a positive definite function, that is the kernel $k(\mathbf{x}, \mathbf{y}) = f(\mathbf{x} - \mathbf{y})$ is positive definite. Bochner's theorem also says the converse is true, i.e. every positive definite function $f(\omega)$ is the Fourier transform of a positive finite Borel measure μ;

Theorem. *A continuous kernel $k(\mathbf{x}, \mathbf{y}) = f(\mathbf{x} - \mathbf{y})$ on $\mathcal{R}^d$ is positive definite if and only if $f(\mathbf{x} - \mathbf{y})$ is the Fourier transform of a non-negative measure.*

When the kernel $k(\mathbf{x}, \mathbf{y})$ is properly scaled, Bochner's theorem guarantees that its Fourier transform $f(\omega)$ is a proper probability distribution:

$$k(\mathbf{x}, \mathbf{y}) = \int_{\mathcal{R}^d} f(\omega) e^{j\omega^T(\mathbf{x}-\mathbf{y})} d\omega = E[e^{j\omega^T(\mathbf{x}-\mathbf{y})}]. \tag{5}$$

In other words, $e^{j\omega^T(\mathbf{x}-\mathbf{y})}$ is an unbiased estimate of $k(\mathbf{x}, \mathbf{y})$ when ω is drawn from the Fourier transform f. Since the kernel is even and real valued, and the probability distribution $f(\omega)$ is purely real, the integrand $e^{j\omega^T(\mathbf{x}-\mathbf{y})}$ may be replaced with $\cos(\omega^T(\mathbf{x} - \mathbf{y}))$. Defining $\mathbf{z}_\omega(\mathbf{x}) = [\cos\left(\omega^T(\mathbf{x})\right) \ \sin\left(\omega^T(\mathbf{x})\right)]^T$ gives a real valued mapping that satisfies the condition $E[\mathbf{z}_\omega(\mathbf{x})^T \mathbf{z}_\omega(\mathbf{y})] = k(\mathbf{x}, \mathbf{y})$, since $\mathbf{z}_\omega(\mathbf{x})^T \mathbf{z}_\omega(\mathbf{y}) = \cos\left(\omega^T(\mathbf{x} - \mathbf{y})\right)$. Each ω maps a data point onto two coefficients as

$$\begin{aligned} \mathbf{z}_\omega(\mathbf{x})^T \mathbf{z}_\omega(\mathbf{y}) &= \cos\left(\omega^T(\mathbf{x} - \mathbf{y})\right) \\ &= [\cos(\omega^T \mathbf{x}) \ \sin(\omega^T \mathbf{x})]^T [\cos(\omega^T \mathbf{y}) \ \sin(\omega^T \mathbf{y})] \end{aligned} \tag{6}$$

It is also possible to show that defining $z_\omega(\mathbf{x}) = \sqrt{2}\cos\left(\omega^T \mathbf{x} + b\right)$ and $\mathbf{z}(x) = n^{-0.5}[z_{\omega_i}]^T$ where n is the cardinality of set $\{\omega_i\}$ and b is a phase parameter uniformly drawn from $[0, 2\pi]$ also gives a real valued mapping, onto a single coefficient this time, that satisfies the condition $E[\mathbf{z}(\mathbf{x})^T \mathbf{z}(\mathbf{y})] = k(\mathbf{x}, \mathbf{y})$. For a set of properly drawn random bases $\{\omega_i\}$ and by law of large numbers,

$$\mathbf{z}(\mathbf{x})^T \mathbf{z}(\mathbf{y}) \approx k(\mathbf{x}, \mathbf{y}). \tag{7}$$

Above, we show that Fourier features can approximate any even and real-valued nonlinear kernel. However, this process is blind to the content and given training data as it does not utilize class label information or density distribution of points. Our goal is not to approximate a prefixed kernel but to find a linear representation of a complex separating boundary via Fourier features that optimize the classification performance. Now, the problem comes how to select a set of ω's such that the final classification performance is optimized. For object detection tasks, a training dataset with binary labels indicating the class memberships is often available. It is desirable to make the best use of these additional prior.

To accomplish this we select the salient Fourier features that minimize a negative binomial log-likelihood of the data as we discuss in the following section.

3. Boosted Feature Selection

Boosting iteratively combines weak classifiers (hypotheses) that favor of those instances misclassified by previous hypotheses. On each round, a distribution of data point weights are updated. The weights of each incorrectly classified data points are increased and the weights of each correctly classified data points are decreased, so that the new classifier focuses more on those examples.

For the binary classification problem we have $y_i \in \{-1, 1\}$. We initialize the weights of the data points at the first round $\beta_n^0 = 1/N$ for all $n = 1, ..., N$. We choose a weak classifierh_t with respect to the weighted data points. The probability of $\mathbf{x}$ being in class 1 is represented by

$$p(\mathbf{x}) = \frac{e^{H(\mathbf{x})}}{e^{H(\mathbf{x})} + e^{-H(\mathbf{x})}} \qquad H(\mathbf{x}) = \frac{1}{2} \sum_{t=1}^{L} h_t(\mathbf{x}). \tag{8}$$

The LogitBoost algorithm learns the set of regression functions $h_t(\mathbf{x})_{t=1..L}$ by minimizing the negative binomial log-likelihood of the data by

$$-\sum_{n=1}^{N} [y_n \log(p(\mathbf{x}_n)) + (1 - y_n) \log(1 - p(\mathbf{x}_n))] \tag{9}$$

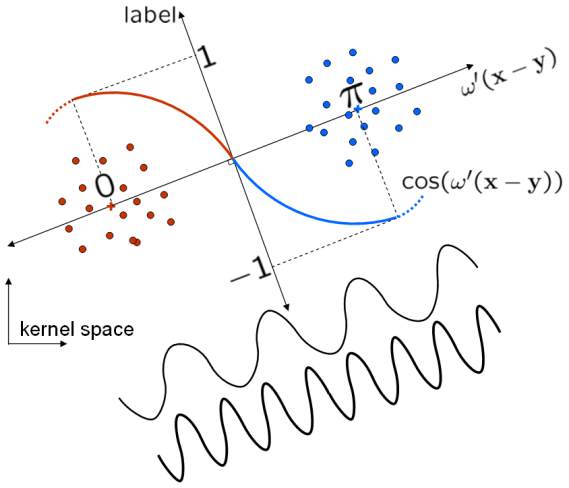

Figure 2. Fourier feature mapping and separation of data points.

through Newton iterations. At the core of the algorithm, LogitBoost fits a weighted least square regression, $h_t(\mathbf{x})$ of training points $\mathbf{x}_n$ to response values λ_n and weights β_n as

$$\lambda_n = \frac{0.5(y_n+1) - p(\mathbf{x}_n)}{p(\mathbf{x}_n)(1-p(\mathbf{x}_n))}, \quad \beta_n = p(\mathbf{x}_n)(1-p(\mathbf{x}_n). \tag{10}$$

The final response of the classifier is $H(\mathbf{x}) = sgn\sum h_t(\mathbf{x})$. At each iteration, a set of hypotheses $S_M : \{\omega_1, ..., \omega_M\}$ are tested. The hypothesis that reduces the negative exponential loss most is combined in the boosted classifier as the current weak classifier, included in the subset of hypotheses S_m, and m iterations are repeated until a performance level is achieved or an upper bound on computational load is reached. Here, m is the dimensionality of transform space and cardinality of the set $S_m \subset S_M$.

An important question is how to determine the set of hypotheses S_M and thus S_m adaptive to the training data. Each hypotheses corresponds to a vector $z_\omega(\mathbf{x}) = \sqrt{2}\cos\left(\omega^T\mathbf{x} + b\right)$ that is desired to be normal to a separating hyperplane between the two classes. The magnitude of this vector represents a space combing frequency.

To obtain the set of hypotheses S_M we apply a generative model based selection scheme. We extract separate probability distribution functions p_- and p_+ that indicate local density for both classes $\{-1, +1\}$. We sample M points from each of these distributions to construct point pairs $\{(\mathbf{x}^-, \mathbf{x}^+)\}_{1..M}$. Each hypothesis corresponds to a pair such that $\omega = \frac{\pi}{|\mathbf{x}^+ - \mathbf{x}^-|^2}(\mathbf{x}^+ - \mathbf{x}^-)$ the combing frequency is $|\omega| = \frac{\pi}{|\mathbf{x}^+ - \mathbf{x}^-|}$, and the corresponding phase shift is $b = -\omega^T\mathbf{x}^+$. In other words, we assign ω as the vector that connects the points of the sampled pair with the corrected norm such that $\cos(\omega^T\mathbf{x}^- + b) = -1$, $\cos(\omega^T\mathbf{x}^+ + b) = 1$, and $\cos(0.5\omega^T(\mathbf{x}^+ - \mathbf{x}^-) + b) = 0$. This means we place one of the separating boundaries (cosine function has infinitely many sign changes, thus separating boundaries) on the middle of both points as illustrated in Fig. 2. It is also possible to apply a weighted discriminant constraints at each iteration by incorporating point weights when we extract the probability density functions p_-, p_+ for tighter fittings.

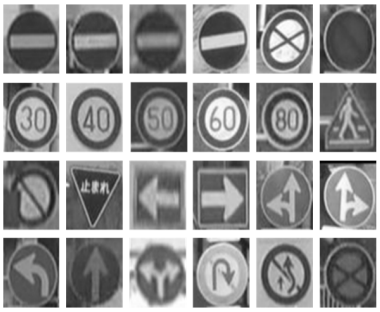

Figure 3. Road signs have different shapes and no color cue is available to help the detector.

After LogitBoost feature selection, we only have the m useful Fourier features in terms of classification. This mapping is nonlinear by nature of Fourier bases and the procedure we use selects m features from a the set S_M. Therefore, we can train linear methods in this new transform space, $\mathbf{x} \rightarrow \mathbf{z}(x) = \sqrt{2}m^{-0.5}[\cos\left(\omega^T\mathbf{x} + b\right)_{1..m}]^T$, returned by the boosted feature selection.

The frequency mapping based feature selection helps to eliminate redundant and irrelevant features for classification. In addition it enhances the generalization capability and speeds up both training process and testing load. Computationally, the above transformation requires only m dot products. As we only use the useful features the set S_M is minimal and so the speed up comes from this process.

It should be noted that, determining the optimum subset is NP-hard as it projects onto a combinatorial problem by nature. Yet, greedy boosting solutions provide satisfactory performance especially when $M \sim N$ without affecting the speed of the testing phase.

4. Experimental Evaluation

Since human detection and road sign detection are two critical applications in video surveillance and vehicle navigation, we demonstrated the potential of our algorithm on these sample tasks.

For road sign detection, we apply a detection window over the gray-level input image. Because the size of the road sign is unknown, the scanning has to be done on multiple scales. We have a total of 20 different sign classes that including square, circular, and triangular shapes and different pictographs as shown in Fig. 3. We select the 10K positive and 100K negative 48×48 samples as the training set from multiple traffic videos. As for the test set, we randomly select another 8386 positive and 100K negative samples.

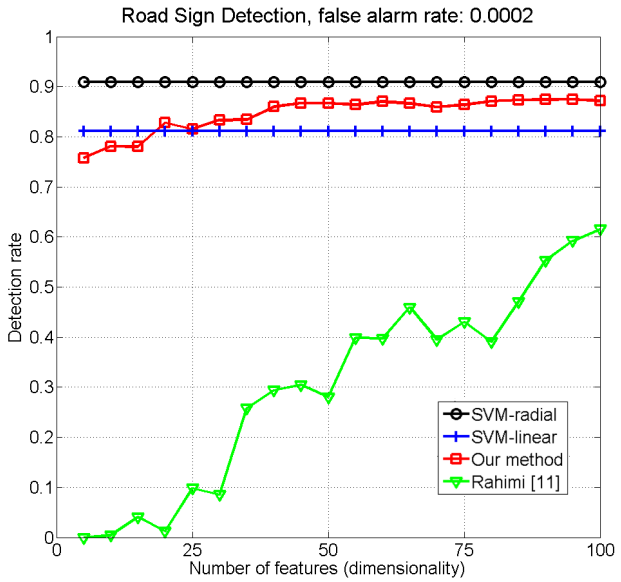

Figure 4. Our method achieves using 45 features 15× speed up in comparison to the SVM-radial, which uses 700 support vectors, with only 4% detection rate loss.

We compute the histogram of oriented gradients (HOG) descriptor for each sample. HOG consists of 6$times$6 blocks (a total of 36 blocks, each is 8×8 in size), and each block is represented by a 9-bin histogram. Concatenating the histograms of these 36 blocks, we obtain a 324 dimensional feature vector for each sample.

A positive (negative) sample is counted as a true detection (false alarm) if is classified as positive. We set the false alarm operation point to 0.0002 and compared the true detection rates for all classifiers. For comparisons, we evaluated several classifiers. (1) SVM-radial kernel with C=19 and σ=1. This classifier gives 91% true detection rate. However, it requires 700 support vectors, which means 700 dot products for every sample. (2) SVM-linear with C=1. This linear classifier gives nearly 81% detection rate and it needs only one dot product computation. (3) SVM-linear for varying number of Fourier features selected completely randomly from the kernel distribution as explained in [11]. (4) Our method: SVM-linear for varying number of Fourier features selected in a data driven manner as explained before.

As shown in Fig. 4, our method quickly outperforms the SVM-linear and using less than 20 features and its performance converges to the SVM-radial as the number of the selected features increases. Table 1 presents the computation times for the classification of 1000 samples from road sign detection dataset using MATLAB code. The SVM-radial implementations are optimized for speed. Our method takes 390ms and achieves 86.6% detection rate. Given the same processing time, the random selection [11] results in 30.5% detection rate in comparison. Its converging behavior is much slower; the random selection is still remains considerably behind using even more than 100 Fourier features. When the number of features is 45, our method is nearly 20× faster than the SVM-radial for only 4.2% detection rate loss requiring only 390ms in comparison to 7.5s.

Figure 5. Samples from human dataset.

The detection rates of our method and [11] sometimes decrease as the number of features increases. This is mainly due to the greedy feature selection mechanism and possibly unintentional overfitting to the training data. Regardless, both methods improve as the number of features increases while our method converges faster.

For human detection, we tested our method on the INRIA datasets where 2416 (685) positive and 10000 (14315) negative samples are included in training (testing) phase. We used a HOG descriptor where each 64×128 sample is represented by 8×4 blocks, for which a 9-bin histogram is computed and normalized. The final feature vector is 288 dimensional. Note that, our feature is much simpler than the one in [14] where each block is represented by 36 coefficients, and each 64×128 sample is represented by 7×15 blocks, resulting in a 3780 dimensional feature vector.

For comparisons, we trained the classifiers at 0.0015 false alarm rate. Since we used a much simpler HOG to demonstrate the strength of our solution, we had to choose a higher false alarm rate to present a statistically meaningful evaluation. Nevertheless, it is always possible to use more blocks and cells, or even another set of features to obtain very high detection rates as in [14]. Our purpose here is to quantify the reduction in the computational complexity while outperforming the SVM-linear and the random selection [11], and converging to the SVM-radial [14] performance.

Figure 6 shows that the proposed frequency mapping

Table 1. Computation Times

	Detection Rate	Time (ms)
SVM-linear	81.1%	10
SVM-radial	90.8%	7500
Rahimi [11]	30.5%	390
Our (45 features)	86.6%	390

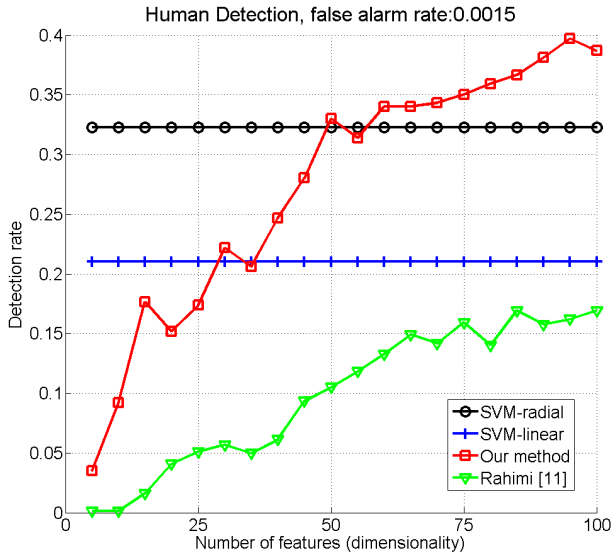

Figure 6. Our method outperforms both random selection and SVM-linear (and even SVM-radial) while processing 140× faster than SVM-radial.

based method outperforms the SVM-linear using less than 30 features, and needs less than 50 features to match the detection accuracy of the SVM-radial, which uses 4871 support vectors. Considering the number of operations, nearly 100× speed up is achieved by our method (∼50 dot products vs. 4871 dot products plus that many exponential operators and additions.

On the other hand, the detection rate of the random selection cannot match the SVM-linear even for 100 features, which indicates the data driven approach is a better choice than fixed (or randomly assigned) kernel approximation on this dataset. More interestingly, using more features results in better detection rates than the SVM-radial. This shows the data driven feature selection acts as an effective kernel design where the dot products in the transform space correspond to a kernel composed of the selected features.

5. Conclusion

We proposed a general purpose data driven feature transformation and selection for support vector machines that provides 20∼100× speed up for classification. Our method adds features in a way that the performance improves at each iteration on the training data and it is expected to follow the same trend on the test data, which has been empirically observed. This means that our method is scalable, i.e., the performance of the classifier can be optimized for the available computational resources, which is not possible using the conventional SVM-linear and SVM-radial algorithms.

Our tests also show that it is possible to optimize the test-time speed for real-time processing.

References

[1] J. Platt, Using sparseness and analytic QP to speed training of Support Vector Machines, *Advances in Neural Information Processing Systems (NIPS)*, 1999.

[2] T. Joachims, Training linear SVMs in linear time, *ACM Conference on Knowledge Discovery and Data Mining (KDD)*, 2006.

[3] M. C. Ferris and T. S. Munson, Interior-point methods for massive Support Vector Machine, *SIAM Journal of Optimization*, 13(3), 783-804, 2003.

[4] S. Shalev-Shwartz, Y. Singer, and N. Srebro, Pegasos: Primal Estimated sub-GrAdient SOlver for SVM, *IEEE International Conference on Machine Learning (ICML)*, 2007.

[5] D. DeCoste and D. Mazzoni, Fast query-optimized kernel machine classification via incremental approximate nearest support vectors, *IEEE International Conference on Machine Learning (ICML)*, 2003.

[6] D. Achlioptas, F. McSherry, and B. Scholkopf, Sampling techniques for kernel methods, *Advances in Neural Information Processing Systems (NIPS)*, 2001.

[7] A. Blum, Random projection, margins, kernels, and feature-selection. *LNCS*, 3940, 52-68, 2006.

[8] P. Drineas and M. W. Mahoney, On the nystrom method for approximating a Gram matrix for improved kernel-based learning, *Conference on Learning Theory (COLT)*, 2005.

[9] C. Yang, R. Duraiswami, and L. Davis, Efficient kernel machines using the improved fast Gauss transform, *Advances in Neural Information Processing Systems (NIPS)*, 2004.

[10] P. Indyk and N. Thaper, Fast image retrieval via embeddings, *International Workshop on Statistical and Computational Theories of Vision*, 2003.

[11] A. Rahimi and B. Recht, Random features for large-scale kernel machines, *Neural Information Processing Systems (NIPS)*, 2007.

[12] J. Friedman, T. Hastie and R. Tibshirani, Additive logistic regression: a statistical view of boosting, *Annals of Statistics*, 28(2), 2000.

[13] W. Rudin, *Fourier Analysis on Groups*, Wiley Classics Library. Wiley-Interscience, New York, 1994.

[14] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005.